Cara Mudah Menganalisis Big Data

Dr. Agus Wibowo, M.Kom, M.Si, MM.

YAYASAN PRIMA AGUS TEKNIK

Cara Mudah Menganalisis Big Data

Dr. Agus Wibowo, M.Kom, M.Si, MM.

YAYASAN PRIMA AGUS TEKNIK

PENERBIT :
YAYASAN PRIMA AGUS TEKNIK
Jl. Majapahit No. 605 Semarang
Telp. (024) 6723456. Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

ISBN 978-623-8642-29-8 (PDF)

Cara Mudah Menganalisis Big Data

Penulis :
Dr. Agus Wibowo, M.Kom, M.Si, MM.

ISBN : 978-623-8642-29-8

Editor :
Dr. Joseph Teguh Santoso, S.Kom., M.Kom.

Penyunting :
Dr. Mars Caroline Wibowo. S.T., M.Mm.Tech

Desain Sampul dan Tata Letak :
Irdha Yunianto, S.Ds., M.Kom

Penebit :
Yayasan Prima Agus Teknik Bekerja sama dengan
Universitas Sains & Teknologi Komputer (Universitas STEKOM)

Anggota IKAPI No: 279 / ALB / JTE / 2023

Redaksi :
Jl. Majapahit no 605 Semarang
Telp. 08122925000
Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

Distributor Tunggal :
Universitas STEKOM
Jl. Majapahit no 605 Semarang
Telp. 08122925000
Fax. 024-6710144
Email : info@stekom.ac.id

KATA PENGANTAR

Selamat datang di buku "***Cara Mudah Menganalisis Big Data***". Di era digital saat ini, big data telah menjadi salah satu aset paling berharga bagi organisasi dan individu. Dengan volume, kecepatan, dan variasi data yang terus meningkat, kemampuan untuk menganalisis dan mengekstrak wawasan dari big data menjadi keterampilan yang sangat diperlukan. Buku ini dirancang untuk memberikan pemahaman yang mendalam dan praktis tentang analisis big data, dengan pendekatan yang mudah dipahami, bahkan bagi pemula sekalipun.

Buku ini dibagi menjadi 18 bab, dimulai dengan **Pengantar Big Data** yang memberikan gambaran umum tentang konsep dan pentingnya big data dalam pengambilan keputusan. Selanjutnya, kami akan memandu Anda untuk **Memulai dengan R**, sebuah bahasa pemrograman yang populer untuk analisis data.

Dalam bab-bab berikutnya, Anda akan belajar tentang **Eksplorasi Data**, **Persiapan Data**, dan **Pemikiran Statistik**, yang merupakan langkah-langkah penting sebelum melakukan analisis lebih lanjut. Kami juga akan memperkenalkan Anda pada konsep **Pembelajaran Mesin** dan teknik-teknik seperti **Pengurangan Dimensionalitas**, **Pengelompokan**, dan **Analisis Keranjang Belanja** yang dapat digunakan untuk menggali informasi berharga dari dataset besar.

Setiap bab dirancang untuk memberikan penjelasan yang jelas dan contoh praktis, sehingga Anda dapat menerapkan pengetahuan yang diperoleh langsung dalam proyek analisis data Anda. Dari **Regresi** hingga **Jaringan Syaraf** dan **Pembelajaran Ensemble**, buku ini mencakup berbagai metode analisis yang dapat digunakan untuk memahami dan memprediksi pola dalam data.

Kami berharap buku ini dapat menjadi panduan yang berguna bagi Anda dalam perjalanan menganalisis big data. Semoga Anda menemukan informasi yang bermanfaat dan dapat mengaplikasikannya dalam praktik. Selamat membaca dan selamat berpetualang dalam dunia big data!

Semarang, Agustus 2024
Penulis

Dr. Agus Wibowo, M.Kom, M.Si, MM

DAFTAR ISI

BAB 1
PENGANTAR BIG DATA

1.1 APA ITU BIG DATA?

Big Data adalah sejumlah besar data Terstruktur, Semiterstruktur, dan Tidak Terstruktur yang berpotensi untuk ditambang sebagai informasi di mana catatan Individu tidak lagi penting dan hanya data agregat yang penting. Data menjadi Big Data ketika sulit untuk diproses menggunakan teknik tradisional.

1.2 KARAKTERISTIK BIG DATA

Ada banyak karakteristik Big Data. Mari saya bahas beberapa di antaranya di sini.

1. **Volume:** Big Data menyiratkan volume data yang sangat besar yang dihasilkan oleh Sensor, Mesin yang dikombinasikan dengan ledakan internet, media sosial, e-commerce, perangkat GPS, dll.
2. **Kecepatan:** Ini menyiratkan laju aliran data seperti pengguna Facebook menghasilkan 3 juta like per hari dan sekitar 450 juta tweet dibuat per hari oleh pengguna.
3. **Keragaman:** Berkaitan dengan jenis format dan dapat diklasifikasikan menjadi 3 jenis:
    - Terstruktur – RDBMS seperti Oracle, MySQL, Sistem lama seperti Excel, Access
    - Semi-Terstruktur – Email, Tweet, File log, Ulasan pengguna
    - Tidak Terstruktur – Foto, Video, File audio.
4. **Kebenaran:** Berkaitan dengan bias, gangguan, dan ketidaknormalan dalam data. Jika kita menginginkan wawasan yang berarti dari data ini, kita perlu membersihkannya terlebih dahulu.
5. **Validitas:** Berkaitan dengan kesesuaian dan ketepatan data karena validitas data sangat penting untuk membuat keputusan.
6. **Volatilitas:** Berkaitan dengan berapa lama data valid karena data yang valid saat ini mungkin tidak valid beberapa menit atau beberapa hari kemudian.

1.3 MENGAPA BIG DATA PENTING?

Keberhasilan organisasi tidak hanya terletak pada seberapa baik mereka dalam menjalankan bisnis mereka tetapi juga pada seberapa baik mereka dapat menganalisis data mereka dan memperoleh wawasan tentang perusahaan mereka, pesaing mereka, dll. Big data dapat membantu Anda dalam mengambil keputusan yang tepat pada waktu yang tepat. Mengapa tidak RDBMS? Skalabilitas merupakan masalah utama dalam RDBMS, sangat sulit untuk mengelola RDBMS ketika persyaratan atau jumlah pengguna berubah. Satu masalah lagi dengan RDBMS adalah kita perlu memutuskan struktur basis data di awal dan membuat perubahan apa pun nanti mungkin menjadi tugas yang sangat besar. Saat menangani Big data, kita memerlukan fleksibilitas dan sayangnya, RDBMS tidak dapat menyediakannya.

1.4 ISTILAH ANALITIK

Analitik adalah salah satu dari sedikit bidang yang memiliki banyak istilah berbeda yang digunakan oleh banyak orang dan banyak dari istilah ini terdengar mirip satu sama lain tetapi digunakan dalam konteks yang berbeda. Ada beberapa istilah yang terdengar sangat berbeda satu sama lain tetapi semuanya serupa dan dapat digunakan secara bergantian. Seseorang yang baru mengenal Analisis diharapkan akan bingung dengan banyaknya terminologi yang ada di bidang ini.

Analisis adalah proses memecah masalah menjadi bagian-bagian yang lebih sederhana dan menggunakan kesimpulan berdasarkan data untuk mengambil keputusan. Analisis bukanlah alat atau teknologi, melainkan cara berpikir dan bertindak. Analisis Bisnis menentukan penerapan Analisis dalam bidang Bisnis. Ini mencakup Analisis Pemasaran, Analisis Risiko, Analisis Penipuan, Analisis CRM, Analisis Loyalitas, Analisis Operasional, serta Analisis SDM. Dalam bisnis, Analisis digunakan dalam semua jenis industri seperti Analisis Keuangan, Analisis Kesehatan, Analisis Ritel, Analisis Telekomunikasi, Analisis Web. Analisis Prediktif semakin populer akhir-akhir ini Vs. Sifat retrospektif seperti OLAP dan BI, analitik deskriptif adalah untuk mendeskripsikan atau mengeksplorasi semua jenis data. Eksplorasi data dan Persiapan Data sangat penting untuk sangat bergantung pada analitik deskriptif. Analitik Big Data adalah istilah baru yang digunakan untuk Menganalisis data tidak terstruktur dan big data seperti terabyte atau bahkan petabyte data. Big Data adalah kumpulan data apa pun yang tidak dapat dianalisis dengan alat konvensional.

1.5 JENIS ANALITIK

Analitik dapat diterapkan pada begitu banyak masalah dan dalam begitu banyak industri yang berbeda sehingga penting untuk meluangkan waktu untuk memahami ruang lingkup analitik dalam bisnis.

Mengklasifikasikan berbagai jenis analitik. Kita akan melihat lebih dekat pada 3 klasifikasi analitik yang luas:

1. Berdasarkan Industri.
2. Berdasarkan fungsi Bisnis
3. Berdasarkan jenis wawasan yang ditawarkan.

Mari kita mulai dengan melihat industri tempat penggunaan analitik sangat lazim. Ada industri tertentu yang selalu menghasilkan sejumlah besar data seperti kartu Kredit dan barang konsumen. Industri-industri ini termasuk yang pertama mengadopsi analitik. Analisis sering diklasifikasikan berdasarkan industri tempat analisis tersebut diterapkan, oleh karena itu Anda akan mendengar istilah seperti analisis asuransi, analisis ritel, analisis web, dan sebagainya. Kita bahkan dapat mengklasifikasikan analisis berdasarkan fungsi bisnis tempat analisis tersebut digunakan. Klasifikasi analisis berdasarkan fungsi dan dampak bisnis adalah sebagai berikut:

- Analisis Pemasaran
- Analisis Penjualan dan SDM
- Analisis rantai pasokan,

- dan sebagainya

Ini bisa menjadi daftar yang cukup panjang karena analisis memiliki prospek untuk memengaruhi hampir semua aktivitas bisnis dalam organisasi besar. Namun, cara yang paling populer untuk mengklasifikasikan analisis adalah berdasarkan apa yang dapat kita lakukan. Semua informasi dikumpulkan dari berbagai industri dan departemen. Yang perlu kita lakukan adalah mengiris dan memotong data dengan berbagai cara, mungkin melihatnya dari berbagai sudut atau dimensi, dll.

Seperti yang Anda lihat, analisis deskriptif mungkin merupakan jenis analisis yang paling sederhana untuk dilakukan karena analisis ini menggunakan informasi yang ada dari masa lalu untuk memahami keputusan saat ini dan diharapkan dapat membantu memutuskan sumber tindakan yang efektif di masa mendatang. Namun, karena relatif mudah dipahami dan diterapkan, analisis deskriptif sering dianggap sebagai saudara kembar analisis yang tidak mencolok. Namun, analisis deskriptif juga sangat kuat potensinya dan dalam sebagian besar situasi bisnis, analisis deskriptif dapat membantu mengatasi sebagian besar masalah. Pengecer sangat tertarik untuk memahami hubungan antara produk. Mereka ingin tahu apakah orang tersebut membeli produk A, apakah ia juga cenderung membeli produk B atau produk C. Ini disebut analisis afinitas produk atau analisis asosiasi dan umumnya digunakan dalam industri ritel. Analisis ini juga disebut analisis keranjang belanja dan digunakan untuk merujuk pada serangkaian teknik yang dapat diterapkan untuk menganalisis keranjang belanja atau transaksi. Pernahkah Anda bertanya-tanya mengapa susu diletakkan tepat di bagian belakang toko sementara majalah dan permen karet diletakkan tepat di dekat kasir? Hal ini karena melalui analitik, pengecer menyadari bahwa saat berjalan jauh ke bagian belakang toko untuk mengambil barang-barang penting, Anda mungkin tergoda untuk mengambil barang lain dan juga karena majalah dan permen karet adalah barang yang murah dan mudah dibeli. Anda memutuskan untuk memasukkannya ke dalam keranjang belanja karena harganya tidak terlalu mahal dan Anda mungkin telah memperhatikannya saat mengantre di konter.

Analisis Prediktif bekerja dengan mengidentifikasi pola dan data historis, lalu menggunakan statistik untuk membuat kesimpulan tentang masa depan. Pada tingkat yang sangat sederhana, kami mencoba memasukkan data ke dalam pola tertentu dan jika kami yakin data mengikuti pola tertentu, maka kami dapat memprediksi apa yang akan terjadi di masa mendatang. Mari kita coba dan lihat contoh lain yang melibatkan analisis prediktif dalam industri telekomunikasi. Sebuah perusahaan telekomunikasi besar memiliki akses ke semua jenis informasi tentang kebiasaan menelepon pelanggannya:

Berapa banyak waktu yang mereka habiskan di telepon? Berapa banyak panggilan internasional yang mereka lakukan? Apakah mereka lebih suka SMS atau menelepon nomor di luar kota mereka?

Ini adalah informasi yang dapat diperoleh murni melalui observasi atau analisis deskriptif. Namun, perusahaan tersebut, yang lebih penting, ingin mengetahui pelanggan mana yang

berencana untuk meninggalkan dan menjalin hubungan baru dengan pesaing mereka. Ini akan menggunakan informasi historis tetapi bergantung pada pemodelan dan analisis prediktif untuk mendapatkan hasil. Ini adalah analisis prediktif. Sementara analisis deskriptif adalah alat yang sangat ampuh. Ia masih memberi kita informasi hanya tentang masa lalu sedangkan, pada kenyataannya, perhatian utama sebagian besar pengguna akan selalu menjadi masa depan. Seorang pemilik hotel ingin memprediksi berapa banyak kamarnya yang akan ditempati minggu depan. CEO Perusahaan Farmasi ingin mengetahui obat mana yang sedang diuji yang kemungkinan besar akan berhasil. Di sinilah analisis prediktif jauh lebih berguna. Selain alat-alat ini, ada jenis analisis ketiga, yang muncul baru-baru ini, mungkin baru berusia satu dekade. Ini disebut analisis preskriptif. Analisis preskriptif melampaui analisis prediktif dengan tidak hanya memberi tahu Anda apa yang sedang terjadi tetapi juga apa yang mungkin terjadi dan yang terpenting apa yang harus dilakukan terhadapnya. Analisis preskriptif juga dapat memberi tahu Anda tentang dampak dari keputusan ini, yang menjadikan analisis preskriptif begitu canggih. Contoh bidang bisnis yang dapat menggunakan analisis preskriptif adalah industri penerbangan atau jaringan jalan nasional. Analisis preskriptif dapat memprediksi kemacetan jalan yang efektif, atau mengidentifikasi jalan yang dapat menerapkan tol untuk memperlancar lalu lintas. Untuk melihat bagaimana analisis preskriptif berfungsi dalam industri penerbangan, mari kita lihat contoh berikut.

Maskapai penerbangan selalu mencari cara untuk mengoptimalkan rute mereka demi efisiensi maksimum. Ini dapat menghemat miliaran dolar, tetapi tidak semudah itu untuk dilakukan. Dengan lebih dari 50 juta penerbangan komersial di dunia setiap tahun, itu berarti ada satu penerbangan setiap detik. Rute penerbangan sederhana antara dua kota, misalnya, San Francisco dan Boston, memiliki kemungkinan 2000 pilihan rute. Jadi, industri penerbangan sering kali mengandalkan analisis preskriptif untuk memutuskan apa, yang mana, dan bagaimana mereka harus menerbangkan pesawat mereka untuk menekan biaya dan meningkatkan laba. Jadi, kami telah menelaah secara mendalam analisis deskriptif, prediktif, dan preskriptif. Fokus kursus ini adalah analisis deskriptif. Menjelang akhir, kami juga akan meluangkan waktu untuk memahami beberapa teknik pemodelan prediktif yang lebih populer.

1.6 SIKLUS HIDUP ANALITIK

Siklus hidup Analytics memiliki beberapa tahap dan banyak orang menggambarkannya dengan berbagai cara, tetapi ide keseluruhannya tetap sama. Berikut ini mari kita pertimbangkan beberapa tahap siklus hidup proyek Analytics berikut.

- Identifikasi Masalah
- Perumusan Hipotesis
- Pengumpulan Data
- Eksplorasi Data
- Persiapan/Manipulasi Data
- Perencanaan/Pembuatan Model
- Validasi Model

- Evaluasi/Pemantauan Hasil

1. Identifikasi Masalah: Masalah adalah situasi yang dinilai sebagai sesuatu yang perlu diperbaiki. Tugas kita adalah memastikan bahwa kita memecahkan masalah yang tepat, mungkin bukan masalah yang disajikan kepada kita oleh klien. Apa yang benar-benar perlu kita pecahkan?

Terkadang pernyataan masalah yang kita dapatkan dari bisnis sangat lugas. Misalnya:

- Bagaimana cara mengidentifikasi pelanggan yang paling berharga?
- Bagaimana cara memastikan bahwa saya meminimalkan kerugian akibat produk yang tidak tersedia di rak?
- Bagaimana cara mengoptimalkan inventaris saya?
- Bagaimana cara mendeteksi pelanggan yang cenderung gagal membayar tagihan?

Ini adalah pernyataan masalah yang lugas dan tidak ada kebingungan mengenai apa yang ingin kita capai dengan proyek analitis. Namun, setiap saat pernyataan bisnis kita mungkin tidak mengarah pada identifikasi masalah yang jelas. Terkadang, pernyataan bisnis bersifat sangat tingkat tinggi dan oleh karena itu Anda perlu meluangkan waktu dengan bisnis tersebut untuk memahami kebutuhan dan mendapatkan konteksnya. Anda mungkin perlu memecah masalah tersebut menjadi beberapa sub-masalah untuk mengidentifikasi persyaratan penting. Anda mungkin perlu memikirkan kendala yang perlu disertakan dalam solusi.

Mari kita ambil contoh untuk ini. Misalkan Anda bekerja di perusahaan kartu kredit, dan bisnis tersebut memberi tahu Anda bahwa ini adalah pernyataan masalah yang mereka ingin Anda lihat, yaitu "Kami ingin menerima aplikasi perawatan kredit hanya dari pelanggan yang baik" Sekarang dari perspektif bisnis, apakah ini pernyataan bisnis yang valid? Tentu saja, pada tingkat yang sangat tinggi, ini adalah persyaratan bisnis yang valid. Namun, untuk tujuan Anda yaitu membangun solusi untuk menjawab pertanyaan ini, apakah ini pernyataan yang sangat valid atau apakah ini merupakan titik awal yang cukup untuk analisis data? Tidak. Karena, ada beberapa masalah dengan pernyataan bisnis seperti ini, yaitu, kami ingin menerima aplikasi perawatan kredit hanya dari pelanggan yang baik. Mari kita lihat masalah dengan pernyataan masalah itu. Saya ingin menerima aplikasi perawatan kredit hanya dari pelanggan yang baik. Salah satu masalah yang paling jelas dengan pernyataan itu adalah siapa yang dimaksud pelanggan yang baik?

Jika Anda memiliki pengetahuan tentang industri kartu kredit, salah satu jawaban untuk pelanggan yang baik adalah orang-orang yang tidak pernah gagal membayar. Artinya, Anda menggunakan kartu kredit dan membayar perusahaan kartu kredit tepat waktu. Namun, definisi lain dari pelanggan yang baik adalah orang-orang yang tidak membayar tepat waktu. Mengapa demikian? Karena, jika Anda tidak membayar tepat waktu, perusahaan kartu kredit memiliki kesempatan untuk mengenakan bunga yang tinggi pada saldo kartu kredit Anda. Pelanggan seperti ini disebut revolver. Siapa sebenarnya pelanggan yang baik bagi perusahaan kartu kredit? Apakah pelanggan yang membayar tepat waktu? Atau pelanggan yang gagal bayar dan tidak membayar tepat waktu. Jawabannya bisa jadi keduanya adalah pelanggan yang baik. Bagaimana mungkin? Itu benar-benar tergantung pada perspektif Anda.

Jika Anda tertarik untuk meminimalkan risiko, jika Anda bekerja di fungsi risiko perusahaan kartu kredit, definisi Anda tentang pelanggan yang baik adalah pelanggan yang membayar tepat waktu, pelanggan yang tidak gagal bayar. Padahal, jika Anda melihat pendapatan, maka perspektif Anda tentang pelanggan yang baik bisa jadi adalah orang-orang yang menghabiskan banyak uang dengan kartu kredit dan tidak membayarnya kembali. Mereka memiliki saldo bergulir yang tinggi. Sekarang, sebagai seorang analis, siapa yang memutuskan siapa pelanggan yang baik? Ketika perusahaan kartu kredit memberi Anda pernyataan bisnis yang mengatakan bahwa kami ingin menerima aplikasi kartu kredit hanya dari pelanggan yang baik. Tahukah Anda bahwa mereka melihat risiko atau pendapatan? Itu benar-benar tergantung pada kepentingan bisnis; itu tergantung pada tujuan bisnis untuk tahun itu. Bahkan, pelanggan yang baik tahun ini mungkin menjadi pelanggan yang buruk tahun depan. Inilah mengapa penting untuk mendapatkan konteks atau pernyataan masalah sebelum memulai analisis. Namun, ini bukan satu-satunya masalah dengan pernyataan masalah ini.

Masalah lain adalah memikirkan tentang keputusan, yaitu, dapatkah Anda benar-benar bersikeras menerima aplikasi yang baik atau dapatkah Anda bersikeras menyetujui aplikasi yang baik. Apakah keputusan tersebut pada tahap aplikasi atau tahap persetujuan? Dapatkah Anda benar-benar mengendalikan aplikasi agar menjadi baik atau dapatkah Anda mengendalikan keputusan untuk memungkinkan hanya pelanggan yang baik yang datang kepada Anda? Masalah lain dengan pernyataan masalah ini adalah kita hanya ingin menerima aplikasi kartu kredit dari pelanggan yang baik. Apakah realistis bagi Anda untuk berasumsi bahwa Anda akan memiliki solusi yang tidak akan pernah menerima pelanggan yang buruk? Sekali lagi, itu bukan hasil yang realistis. Kembali ke definisi masalah kita, yaitu, jika ada masalah bisnis, saya ingin mendapatkan pelanggan yang baik sebagai perusahaan kartu kredit. Bagaimana Anda membingkai masalah itu menjadi sesuatu yang dapat diatasi dengan pendekatan analitis?

Salah satu caranya adalah dengan menambahkan hal-hal spesifik ke dalam pernyataan masalah. Jadi, pikirkan tentang hasil yang spesifik, terukur, dapat dicapai, realistis, dan tepat waktu yang dapat Anda lampirkan ke dalam pernyataan masalah itu. Itulah sebabnya kami menekankan bahwa Anda perlu memahami konteks bisnis secara menyeluruh dan berbicara dengan bisnis yang Anda tangani untuk masalah yang tepat. Bagaimana saya dapat menambahkan hal-hal spesifik ke dalam pernyataan masalah ini? Mari kita asumsikan bahwa saya melihatnya dari perspektif risiko, karena tahun ini perusahaan kartu kredit saya berfokus pada pengurangan risiko portofolio. Jadi, saya dapat memiliki berbagai pernyataan masalah bisnis. Misalnya, mengurangi kerugian akibat gagal bayar kartu kredit setidaknya 30 persen dalam 12 bulan pertama pasca penerapan strategi baru.

Mengembangkan algoritma untuk menyaring aplikasi yang tidak memenuhi kriteria yang ditetapkan pelanggan yang baik yang akan mengurangi gagal bayar hingga 20 persen dalam 3 bulan ke depan. Mengidentifikasi strategi untuk mengurangi gagal bayar hingga 20 persen dalam tiga bulan ke depan dengan memberikan pelanggan yang berisiko opsi pembayaran tambahan. Kami telah memutuskan bahwa definisi masalah yang baik adalah

sesuatu yang kami tangani dari perspektif risiko. Namun, untuk pernyataan bisnis yang sama, kami sekarang memiliki tiga pernyataan masalah berbeda yang menangani tiga hal berbeda. Sekali lagi, yang mana dari ini yang harus saya pilih sebagai titik awal untuk analisis saya? Haruskah saya mengidentifikasi strategi untuk pelanggan saya yang sudah ada atau haruskah saya mengidentifikasi calon pelanggan baru? Sekali lagi, ini adalah sesuatu yang mungkin didorong oleh kebutuhan bisnis. Jadi, penting untuk terus berbicara dengan bisnis untuk memastikan bahwa ketika Anda memulai proyek analitik, Anda menangani pernyataan masalah yang tepat.

Menemukan masalah yang didefinisikan dengan jelas sering kali didorong oleh penemuan – Mulailah dengan definisi konseptual dan melalui analisis (akar penyebab, analisis dampak, dll.) Anda membentuk dan mendefinisikan ulang masalah dalam hal isu. Masalah menjadi diketahui ketika seseorang mengamati adanya ketidaksesuaian antara keadaan yang ada dan keadaan yang seharusnya. Masalah dapat diidentifikasi melalui:

- Studi perbandingan/pembandingan
- Pelaporan kinerja - penilaian kinerja saat ini terhadap sasaran dan tujuan
- Analisis SWOT – penilaian kekuatan, kelemahan, peluang, dan ancaman
- Keluhan/Survei

Terkadang hal yang kita anggap sebagai masalah bukanlah masalah yang sebenarnya, jadi untuk mengetahui masalah yang sebenarnya, diperlukan penyelidikan. Analisis Akar Masalah adalah metode penyelidikan yang efektif – analisis ini membantu mengidentifikasi apa, bagaimana, dan mengapa sesuatu terjadi.

Mari kita pertimbangkan tingkat pergantian karyawan di organisasi kita yang meningkat. Kita perlu mencari tahu *Five Why* yang mengacu pada praktik bertanya, lima kali, mengapa masalah itu ada untuk sampai ke akar penyebab masalah:

- Mengapa Karyawan pindah ke pekerjaan lain?
- Mengapa Karyawan tidak puas?
- Mengapa Karyawan merasa bahwa mereka dibayar rendah?
- Mengapa pemberi kerja lain membayar gaji yang lebih tinggi?
- Mengapa Permintaan untuk karyawan seperti itu meningkat di pasar?

Pertanyaan Dasar untuk Ditanyakan dalam Mendefinisikan Masalah:

- Siapa yang menyebabkan masalah?
- Siapa yang terkena dampak masalah ini?
- Apa yang akan terjadi jika masalah ini tidak diselesaikan?
- Apa dampaknya? Di mana dan Kapan masalah ini terjadi?
- Mengapa masalah ini terjadi?
- Bagaimana seharusnya proses tersebut bekerja?
- Bagaimana orang-orang saat ini menangani masalah tersebut?

2. Merumuskan hipotesis: Uraikan masalah dan rumuskan hipotesis. Susun pertanyaan yang perlu dijawab atau topik yang perlu dieksplorasi untuk memecahkan masalah.

- Kembangkan daftar lengkap semua kemungkinan masalah yang terkait dengan masalah tersebut.

- Kurangi daftar lengkap dengan menghilangkan duplikat dan menggabungkan masalah yang tumpang tindih.
- Gunakan pembangunan konsensus untuk membuat daftar masalah utama.

3. Pengumpulan Data: Untuk menjawab pertanyaan kunci dan memvalidasi hipotesis, pengumpulan informasi yang realistis diperlukan. Bergantung pada jenis masalah yang dipecahkan, teknik pengumpulan data yang berbeda dapat digunakan. Pengumpulan data merupakan tahap penting dalam penyelesaian masalah - jika bersifat dangkal, bias, atau tidak lengkap, analisis data akan sulit dilakukan.

Teknik Pengumpulan Data:

- Menggunakan data yang telah dikumpulkan oleh orang lain
- Secara sistematis memilih dan mengamati karakteristik orang, objek, atau peristiwa.
- Pertanyaan lisan kepada responden, baik secara individu maupun kelompok.
- Mengumpulkan data berdasarkan jawaban yang diberikan oleh responden dalam bentuk tertulis.
- Memfasilitasi diskusi bebas tentang topik tertentu dengan kelompok peserta terpilih.

4. Eksplorasi Data: Sebelum analisis data formal dapat dilakukan, analis harus mengetahui berapa banyak kasus yang ada dalam kumpulan data, variabel apa saja yang disertakan, berapa banyak observasi yang hilang, dan hipotesis umum apa yang mungkin didukung oleh data tersebut. Eksplorasi awal kumpulan data membantu menjawab pertanyaan-pertanyaan ini dengan membiasakan analis tentang data yang sedang mereka kerjakan.

Analis biasanya menggunakan visualisasi untuk eksplorasi data karena memungkinkan pengguna untuk melihat sebagian besar fitur yang relevan dari kumpulan data mereka dengan cepat dan mudah. Dengan melakukan ini, pengguna dapat mengidentifikasi variabel yang mungkin memiliki observasi yang menarik. Dengan menampilkan data secara grafis melalui diagram sebar atau diagram batang, pengguna dapat melihat apakah dua atau lebih variabel berkorelasi dan menentukan apakah keduanya merupakan kandidat yang baik untuk analisis lebih mendalam.

5. Persiapan Data: Data datang kepada Anda dalam bentuk yang tidak mudah dianalisis. Kita perlu membersihkan data dan memeriksa konsistensinya, manipulasi data yang ekstensif diperlukan untuk menganalisisnya.

Langkah-langkah Persiapan Data dapat meliputi:

- Mengimpor data
- Identifikasi Variabel/Membuat
- Variabel baru
- Memeriksa dan meringkas data
- Memilih subset data
- Memilih dan mengelola variabel.
- Menggabungkan data
- Membagi data menjadi beberapa set data.
- Pengobatan nilai yang hilang
- Pengobatan outlier

Identifikasi Variabel: Pertama, identifikasi variabel Prediktor (Input) dan Target (output). Kemudian, identifikasi tipe data dan kategori variabel.

Analisis Univariat: Pada tahap ini, kita akan mengeksplorasi variabel satu per satu. Metode untuk melakukan analisis Univariat akan bergantung pada apakah tipe variabel tersebut kategorikal atau kontinu. Mari kita lihat metode dan ukuran statistik ini untuk variabel kategorikal dan kontinu secara individual.

Variabel Kontinu: Dalam kasus variabel kontinu, kita perlu memahami kecenderungan sentral dan penyebaran variabel. Ini diukur menggunakan berbagai metode visualisasi metrik statistik.

Variabel Kategorikal: Untuk variabel kategorikal, kita menggunakan tabel frekuensi untuk memahami distribusi setiap kategori. Kita juga dapat membaca sebagai persentase nilai di bawah setiap kategori. Ini dapat diukur menggunakan dua metrik, Hitungan dan Persen terhadap setiap kategori.

6. Pembuatan Model: Ini benar-benar seluruh proses pembuatan solusi dan penerapan solusi. Sebagian besar waktu proyek dihabiskan dalam tahap implementasi solusi. Satu hal menarik yang perlu diingat dengan pendekatan analitis adalah bahwa pendekatan analitis saat Anda membangun model, model analitis, merupakan proses yang sangat berulang karena tidak ada yang namanya solusi akhir atau solusi yang sempurna. Biasanya, Anda akan menghabiskan waktu membangun beberapa model pada beberapa solusi sebelum sampai pada solusi terbaik yang akan digunakan oleh bisnis.

7. Ada banyak cara untuk mengambil keputusan dari perspektif bisnis. Analisis adalah salah satu caranya. Ada cara lain untuk mengambil keputusan. Bisa jadi pengambilan keputusan berdasarkan pengalaman. Bisa juga pengambilan keputusan berdasarkan intuisi. Dan tidak setiap saat Anda akan selalu memilih pendekatan analitis. Namun, dalam jangka panjang, masuk akal untuk membangun kemampuan analitis karena hal itu mengarah pada pengambilan keputusan yang lebih objektif. Namun pada dasarnya jika Anda ingin data untuk mendorong pengambilan keputusan, Anda perlu memastikan bahwa Anda telah berinvestasi dalam pengumpulan data yang tepat untuk memungkinkan pengambilan keputusan Anda melalui data.

8. Evaluasi/Pemantauan Model: Ini adalah proses yang berkelanjutan yang pada dasarnya ditujukan untuk melihat efektivitas solusi dari waktu ke waktu. Ingatlah bahwa pendekatan pemecahan masalah analitis, yang berbeda dari pendekatan pemecahan masalah standar. Kita perlu mengingat poin-poin ini:

- Ada keyakinan yang jelas pada data untuk mendorong identifikasi solusi.
- Kita menggunakan teknik analitis berdasarkan teori numerik.
- Anda perlu memiliki pemahaman yang baik tentang konsep teoritis untuk situasi bisnis guna membangun solusi yang layak.

Artinya, Anda perlu memiliki pemahaman yang baik tentang situasi bisnis dan konteks bisnis, serta pengetahuan yang kuat tentang pendekatan analitis dan mampu menggabungkan konsep, menghasilkan solusi yang dapat diterapkan. Di beberapa industri, laju perubahan sangat tinggi. Jadi, solusi menua dengan sangat cepat. Di industri lain, laju perubahan mungkin

tidak setinggi itu dan saat Anda membangun solusi, Anda mungkin memiliki waktu 2-3 tahun di mana solusi Anda berfungsi dengan baik, tetapi setelah itu perlu disesuaikan untuk mengelola kondisi bisnis yang baru. Namun, cara untuk menilai apakah solusi Anda berfungsi atau tidak adalah dengan memeriksa efektivitas solusi secara berkala. Anda perlu melacak ketergantungan dari waktu ke waktu dan Anda mungkin perlu membuat perubahan kecil untuk mengembalikan solusi ke jalurnya. Terkadang, Anda mungkin harus membangun seluruh solusi dari awal karena lingkungan telah berubah begitu dramatis sehingga solusi yang Anda bangun tidak lagi sesuai dengan konteks bisnis saat ini.

1.7 KESALAHAN UMUM DALAM BERPIKIR ANALITIS

Definisi masalah klien mungkin tidak tepat. Dia mungkin tidak memiliki pengetahuan dan pengalaman yang Anda miliki. Karena sebagian besar masalah tidaklah unik, Kami mungkin dapat menguatkan masalah dan kemungkinan solusi terhadap sumber lain. Solusi terbaik untuk suatu masalah sering kali terlalu sulit untuk diterapkan oleh klien. Jadi berhati-hatilah dalam merekomendasikan solusi optimal untuk suatu masalah. Sebagian besar penjelasan memerlukan beberapa tingkat konsiliasi untuk pelaksanaan.

BAB 2
MEMULAI DENGAN R

2.1 PENDAHULUAN

R adalah bahasa pemrograman untuk analisis dan pelaporan statistik. R adalah bahasa pemrograman sederhana yang mencakup banyak fungsi untuk Analisis Data, memiliki fasilitas penanganan dan penyimpanan data yang efektif. R menyediakan fasilitas grafis untuk analisis dan pelaporan data. Saya meminta Anda untuk menginstal R dan R studio yang dapat diunduh secara gratis. Di sini, dalam buku saya, saya menggunakan kode yang ditulis dalam R studio. Untuk bekerja di R studio, Anda harus memiliki R di bagian belakang, jadi silakan kunjungi situs CRAN dan instal versi terbaru R sesuai dengan sistem operasi Anda. Jadi, mari kita mulai (Rock and Roll) dengan R-Studio:

Saat pertama kali membuka R-studio, Anda akan melihat empat jendela.

1. *Skrip:* Berfungsi sebagai area untuk menulis dan menyimpan kode R
2. *Ruang kerja:* Mencantumkan kumpulan data dan variabel dalam lingkungan R
3. *Plot:* Menampilkan plot yang dihasilkan oleh kode R
4. Konsol Menyediakan riwayat kode R yang dieksekusi dan output.

2.2 OPERASI DASAR DI R

1. Ekspresi:

jika Anda hanya bekerja dengan angka, R dapat digunakan sebagai kalkulator tingkat lanjut, cukup ketik

```
4+5
dan tekan enter, Anda akan mendapatkan nilai 9.
```

R dapat melakukan perhitungan matematika tanpa kewajiban untuk menyimpannya dalam suatu objek.

Hasilnya dicetak pada konsol.

```
Coba hitung hasil perkalian 2 angka atau lebih (* adalah operator
perkalian).
6*9 # Anda akan mendapatkan 54.
```

Apa pun yang ditulis setelah tanda # akan dianggap sebagai komentar dalam R.

R mengikuti aturan BODMAS untuk melakukan operasi matematika.

Ketik perintah berikut dan pahami perbedaannya.

```
20-15*2 # Anda akan mendapatkan -10
(20-15)*2 #di sini Anda akan mendapatkan 10
```

Hati-hati membagi nilai apa pun dengan 0 akan memberi Anda inf (tak terhingga). ketik perintah ini di konsol dan centang

```
8/0
```

Operasi matematika ini dapat digabungkan menjadi rumus panjang untuk mencapai tugas tertentu.

2. Nilai Logika:

Beberapa ekspresi mengembalikan "nilai logika": TRUE atau FALSE. (dikenal sebagai nilai "Boolean"). Lihatlah ekspresi yang memberi kita nilai logika:

```
6<9 #TRUE
```

3. Variabel:

Kita dapat menyimpan nilai ke dalam variabel untuk mengaksesnya nanti.

```
X <- 48 #untuk menyimpan nilai dalam x.
Y <- "YL, Prasad" (Jangan lupa tanda kutip)
```

Sekarang X dan Y adalah objek yang dibuat dalam R, dapat digunakan dalam ekspresi di posisi hasil asli.

Coba panggil X dan Y hanya dengan mengetikkan nama objek

```
Y # [1] "YL, Prasad"
```

Kita perlu mengingat bahwa R peka huruf besar/kecil. Jika Anda menetapkan nilai untuk huruf besar X dan memanggil huruf kecil x maka akan muncul kesalahan.

Coba bagi X dengan 2 (/ adalah operator pembagian) # Anda akan mendapatkan 24 sebagai jawaban Kita dapat menetapkan ulang nilai apa pun ke variabel kapan saja. Tetapkan "Lakshmi" ke Y. Y <- "Lakshmi"

Kita dapat mencetak nilai variabel hanya dengan mengetikkan namanya di konsol. Coba cetak nilai Y saat ini.

Jika Anda menulis kode ini, selamat! Anda menulis kode pertama di R dan membuat objek.

4. Fungsi:

Kita dapat memanggil fungsi dengan mengetikkan namanya, diikuti oleh argumen ke fungsi tersebut dalam tanda kurung.

Coba fungsi sum, untuk menjumlahkan beberapa angka. Masukkan:

```
sum(1, 3, 5) #9
```

Kita menggunakan fungsi sqrt untuk mendapatkan akar kuadrat dari 16.

```
sqrt(16) #4
16^.5 #juga memberikan jawaban yang sama dengan 4
```

Transformasi akar kuadrat adalah transformasi yang paling banyak digunakan bersama dengan transformasi log dalam persiapan data. Ketik perintah berikut dan periksa jawabannya

```
log(1) #0
log(10) #2.302585
log10(100) # ini akan mengembalikan 2 karena log dari 100 ke basis
10 adalah 2.
```

kapan pun jika Anda ingin mengakses jendela bantuan, Anda dapat mengetik perintah berikut

```
help(exp)
?exp
```

Jika Anda menginginkan beberapa contoh fungsi, berikan perintah ini:

```
example(log)
```

R memungkinkan seseorang untuk menyimpan lingkungan ruang kerja, termasuk variabel dan pustaka yang dimuat, ke dalam file data .R menggunakan fungsi save.image(). File data .R yang ada dapat dimuat menggunakan fungsi load.image().

5. File

Perintah R dapat ditulis dan disimpan dalam file teks biasa (dengan ekstensi ".R") untuk dieksekusi nanti.

Asumsikan bahwa kita menyimpan beberapa contoh skrip, Kita dapat membuat daftar file di direktori saat ini dari dalam R, dengan memanggil fungsi list.files.

```
list.files()
```

2.3 MENYIAPKAN DIREKTORI KERJA

Sebelum mempelajari R lebih dalam, ada baiknya untuk menyiapkan direktori kerja guna menyimpan semua berkas, skalar, vektor, kerangka data, dsb. Untuk itu, pertama-tama kita perlu mengetahui direktori saat ini yang digunakan R secara default. Untuk memahaminya, ketik perintah:

```
getwd() # [1] "C:/Users/admin/Documents"
```

Sekarang saya ingin menetapkan folder R data sebagai direktori kerja yang terletak di drive D. Untuk melakukannya, saya akan memberikan perintah:

```
setwd("D:/R data")
```

Tekan enter (Klik Ikon Kirim) untuk memastikan bahwa perintah Anda telah dijalankan dan direktori kerja telah ditetapkan. Kita menetapkan folder R data di drive D sebagai direktori kerja. Ini bukan berarti kita membuat sesuatu yang baru di sini, tetapi hanya menetapkan tempat sebagai direktori kerja, di sinilah semua berkas akan ditambahkan. untuk memeriksa apakah direktori kerja telah diatur dengan benar berikan perintah:

```
getwd()
```

2.4 STRUKTUR DATA DALAM R

Struktur data adalah antarmuka ke data yang diatur dalam memori komputer. R menyediakan beberapa jenis struktur data yang masing-masing dirancang untuk mengoptimalkan beberapa aspek penyimpanan, akses, atau pemrosesan.

Contoh Struktur Data: 1.Vektor 2.Matriks 3.Faktor 4.Bingkai Data

1. Vektor

Vektor merupakan blok penyusun dasar untuk data dalam R. Variabel R sebenarnya adalah vektor. Vektor hanya dapat terdiri dari nilai-nilai dalam kelas yang sama. Pengujian untuk vektor dapat dilakukan menggunakan fungsi is.vector().

Namanya mungkin terdengar menakutkan, tetapi vektor hanyalah daftar nilai. Nilai vektor dapat berupa angka, string, nilai logika, atau tipe lainnya, selama semuanya bertipe sama.

Jenis Vektor: **Integer, Numerik, Logika, Karakter, Kompleks.**

R menyediakan fungsionalitas yang memungkinkan pembuatan dan manipulasi vektor dengan mudah. Kode R berikut mengilustrasikan cara membuat vektor menggunakan fungsi gabungan, c()

atau operator titik dua, :,

Mari kita buat vektor angka:

```
c(4,7,9)
```

Fungsi c (c adalah kependekan dari Combine) membuat vektor baru dengan menggabungkan daftar nilai. Buat vektor dengan string:

```
c('a','b','c')
```

Vektor Deret

Kita dapat membuat vektor dengan notasi start:end untuk membuat deret. Mari kita buat vektor dari deret bilangan bulat dari 5 hingga 9.

5:9 # Membuat vektor dengan nilai dari 5 hingga 9:

Kita bahkan dapat memanggil fungsi seq. Mari kita coba melakukan hal yang sama dengan seq:

```
seq(5,9)
```

Akses Vektor

Setelah membuat vektor dengan beberapa string di dalamnya dan menyimpannya, Kita dapat mengambil nilai individual dalam vektor hanya dengan memberikan indeks numeriknya dalam tanda kurung siku.

```
sentence <- c('Learn', 'Data', 'Analytics') sentence[3] # [1]
                          "Analytics"
```

Kita dapat menetapkan nilai baru dalam vektor yang sudah ada. Coba ubah kata ketiga menjadi "Science":

```
sentence [3] <- "Science"
```

Jika Anda menambahkan nilai baru ke vektor, vektor akan tumbuh untuk mengakomodasinya. Mari tambahkan kata keempat:

```
sentence [4] <- 'Oleh YL, Prasad'
```

Kita dapat menggunakan vektor di dalam tanda kurung siku untuk mengakses beberapa nilai. Coba dapatkan kata pertama dan keempat:

```
sentence[c(1, 4)]
```

Ini berarti Anda dapat mengambil rentang nilai. Dapatkan kata kedua hingga keempat:

```
kalimat [2:4]
```

Kita dapat menetapkan rentang nilai, hanya dengan menyediakan nilai dalam vektor. kalimat [5:7] <- c('at', 'PRA', 'Analytix') # untuk menambahkan kata 5 hingga 7 Coba akses kata ketujuh dari vektor kalimat:

```
kalimat[7]
```

Nama Vektor

Mari kita buat vektor 3-item, dan simpan dalam variabel peringkat. Kita dapat menetapkan nama ke elemen vektor dengan meneruskan vektor kedua yang diisi dengan nama ke fungsi penugasan nama, seperti ini:

```
peringkat <- 1:3
```

nama(peringkat) <- c("pertama", "kedua", "ketiga") peringkat
Kita dapat menggunakan nama untuk mengakses nilai vektor.
peringkat["pertama"]

2. Matriks

Matriks dalam R adalah kumpulan elemen homogen yang disusun dalam 2 dimensi. Matriks adalah vektor dengan atribut dim, yaitu vektor integer yang memberikan jumlah baris

dan kolom. Fungsi dim(), nrow(), dan ncol menyediakan atribut matriks. Baris dan kolom dapat memiliki nama, dimnames(), rownames(), colnames(). Mari kita lihat dasar-dasar bekerja dengan matriks, membuatnya, mengaksesnya, dan memplotnya. Mari kita buat matriks setinggi 3 baris dan selebar 4 kolom, dengan semua bidangnya disetel ke 0. Sampel <- matriks (0, 3, 4)

Konstruksi Matriks

Kita dapat membuat matriks secara langsung dengan elemen data, konten matriks diisi sepanjang orientasi kolom secara default. Lihat kode berikut, konten Sampel diisi dengan kolom secara berurutan. Sampel <- matriks (1:20, nrow=4, ncol=5)

Akses Matriks

Untuk mendapatkan nilai dari matriks, Anda hanya perlu memberikan dua indeks, bukan satu. Mari kita cetak matriks Sampel kita:

```
print (Contoh)
```

Coba dapatkan nilai dari baris kedua di kolom ketiga Contoh:

```
Contoh [2,3]
```

Kita bisa mendapatkan seluruh baris matriks dengan menghilangkan indeks kolom (tetapi tetap menggunakan koma).

Coba dapatkan Baris ketiga:

```
Contoh [3,] # [1] 3 7 11 15
```

Untuk mendapatkan seluruh kolom, hilangkan indeks baris. Dapatkan kolom keempat:

```
Contoh [,4] # [1] 13 14 15 16
```

3. Faktor

Ketika kita ingin data dikelompokkan berdasarkan kategori, R memiliki tipe khusus yang disebut faktor untuk melacak nilai-nilai yang dikategorikan ini. Faktor adalah vektor yang elemen-elemennya dapat mengambil salah satu dari sekumpulan nilai tertentu. Misalnya, "Jenis Kelamin" biasanya hanya akan mengambil nilai "Pria", "Wanita", dan "NA". Sekumpulan nilai yang dapat diambil oleh elemen-elemen faktor disebut levelnya.

Membuat Faktor

Untuk mengkategorikan nilai, cukup berikan vektor ke fungsi faktor:

```
gender <- c('male', 'female', 'male', 'NA', 'female') types <-
                        factor(gender)
                        print(gender)
```

Anda akan melihat daftar mentah string, nilai berulang, dan semuanya. Sekarang cetak faktor types:

```
print(types)
```

Mari kita lihat integer yang mendasarinya. Berikan faktor ke fungsi as.integer:

```
as.integer(types) # [1] 2 1 2 3 1
```

Anda hanya bisa mendapatkan level faktor dengan fungsi levels:

```
levels(types) # [1] "female" "male" "NA"
```

4. Bingkai Data

Bingkai data menyediakan struktur untuk menyimpan dan mengakses beberapa variabel dengan tipe data yang mungkin berbeda. Karena fleksibilitasnya dalam menangani banyak tipe data, kerangka data merupakan format masukan yang lebih disukai untuk banyak fungsi pemodelan yang tersedia di R. Mari kita buat tiga objek individual bernama Id, Gender, dan Age dan gabungkan ketiganya menjadi satu set data.

```
Id <- c(101, 102, 103, 104, 105)
Gender <- c('male', 'female', 'male', 'NA', 'female')
Age <- c(38,29,NA,46,53)
```

Id, Gender, dan Age adalah tiga objek individual, R memiliki struktur yang dikenal sebagai kerangka data yang dapat menggabungkan semua variabel ini dalam satu tabel atau lembar kerja Excel. Kerangka data memiliki jumlah kolom tertentu, yang masing-masing diharapkan berisi nilai dari tipe tertentu. Kerangka data juga memiliki jumlah baris yang tidak dapat ditentukan - set nilai terkait untuk setiap kolom.

Mudah untuk membuat set data, cukup panggil fungsi data.frame, dan berikan Id, Gender, dan Age sebagai argumen. Tetapkan hasil ke set data Test:

```
Test <- data.frame(Id, Gender, Age)
```

Print Test untuk melihat isinya:

```
print(Test)
fix(Test) #Untuk melihat set data objek ini
```

Akses Data Frame: Mudah untuk mengakses bagian-bagian individual dari data frame. Kita bisa mendapatkan kolom-kolom individual dengan memberikan nomor indeksnya dalam tanda kurung ganda. Coba dapatkan kolom kedua (Gender) dari Test:

```
Test[[2]]
```

Anda bisa memberikan nama kolom sebagai string dalam tanda kurung ganda agar lebih mudah dibaca

```
Test[["Age"]]
```

Kita bahkan bisa menggunakan notasi singkat: nama data frame, tanda dolar, dan nama kolom tanpa tanda kutip.

```
Test$Gender
```

2.5 MENGIMPOR DAN MENGEKSPOR DATA

Sering kali kita perlu mendapatkan data dari file eksternal seperti file Teks, lembar excel, dan file CSV. Untuk melakukan ini, R diberi kemampuan untuk memuat data dari file eksternal dengan mudah.

Lingkungan Anda mungkin memiliki banyak objek dan nilai, yang dapat Anda hapus menggunakan kode berikut:

```
rm(list=ls())
```

Fungsi rm() memungkinkan Anda menghapus objek dari lingkungan tertentu.
Mengimpor file TXT: Jika Anda memiliki file teks berformat .txt atau dibatasi tab, Anda dapat mengimpornya dengan mudah menggunakan fungsi dasar R read.table().

```
setwd("D:/R data")
Inc_ds <- read.table("Income.txt")
```

Untuk file yang menggunakan string pemisah selain koma, Anda dapat menggunakan fungsi read.table. Argumen sep= mendefinisikan karakter pemisah, dan Anda dapat menentukan karakter tab dengan "\t". Panggil read.table pada "Inc_tab.txt", menggunakan pemisah tab:

```
read.table("Inc_tab.txt ", sep="\t")
```

Perhatikan tajuk kolom "V1","V2" dan "V3"? Baris pertama tidak secara otomatis diperlakukan sebagai tajuk kolom dengan read.table. Perilaku ini dikontrol oleh argumen tajuk. Panggil read.table lagi, atur tajuk ke TRUE:

```
Inc_th <- read.table("Inc_tab.txt", sep="\t", tajuk=TRUE)
fix(Inc_th)
```

Mengimpor Berkas CSV: Jika Anda memiliki berkas yang memisahkan nilai dengan koma, biasanya Anda berurusan dengan berkas .csv. Anda dapat memuat konten berkas CSV ke dalam bingkai data dengan meneruskan nama berkas ke fungsi read.CSV.

read.CSV(“Employee.csv”) #Untuk melakukan ini, R mengharapkan keberadaan berkas kita di direktori kerja. 2.5 Mengekspor file menggunakan fungsi write.table: Fungsi write.table menghasilkan file data. Argumen pertama menentukan bingkai data mana di R yang akan diekspor. Argumen berikutnya menentukan file yang akan dibuat. Pemisah default adalah spasi kosong tetapi pemisah apa pun dapat ditentukan dalam opsi sep=. Karena kami tidak ingin menyertakan nama baris, kami diberi opsi row.names=FALSE. Pengaturan default untuk opsi quote adalah menyertakan tanda kutip di sekitar semua nilai karakter, yaitu, di sekitar nilai dalam variabel string dan di sekitar nama kolom. Seperti yang telah kami tunjukkan dalam contoh, sangat umum untuk tidak menginginkan tanda kutip saat membuat file teks.

```
write.table(Employee, file=”emp.txt”, row.names = FALSE, quote =
FALSE)
```

BAB 3
EKSPLORASI DATA

3.1 PENDAHULUAN

Setiap kali kita hendak membuat model, sangat penting untuk memahami data dan menemukan wawasan tersembunyi dari data tersebut. Keberhasilan proyek analisis data memerlukan pemahaman mendalam tentang data tersebut. Eksplorasi data akan membantu Anda membuat model yang akurat jika Anda melakukannya secara terencana. Sebelum analisis data formal dapat dilakukan, analis harus mengetahui berapa banyak kasus yang ada dalam kumpulan data, variabel apa yang disertakan, berapa banyak observasi yang hilang dalam kumpulan data tersebut. Langkah-langkah eksplorasi data meliputi Memahami kumpulan data dan variabel, Memeriksa atribut data, Mengenali dan menangani nilai yang hilang, outlier, Memahami penyajian dasar data, dll. Aktivitas eksplorasi data meliputi studi data dalam hal ukuran statistik dasar dan pembuatan grafik dan plot untuk memvisualisasikan dan mengidentifikasi hubungan dan pola.

Eksplorasi awal kumpulan data membantu menjawab pertanyaan-pertanyaan ini dengan membiasakan analis tentang data yang sedang mereka kerjakan. Pertanyaan dan pertimbangan tambahan untuk langkah pengkondisian data meliputi, Apa saja sumber datanya? Apa saja bidang targetnya? Seberapa bersih datanya? Seberapa konsisten konten dan berkasnya? Sebagai Ilmuwan Data, Anda perlu menentukan sejauh mana data tersebut berisi nilai yang hilang atau tidak konsisten dan apakah data tersebut berisi nilai yang menyimpang dari normal dan Menilai konsistensi tipe data. Misalnya, jika tim mengharapkan data tertentu berupa angka, konfirmasikan bahwa data tersebut berupa angka atau campuran string alfanumerik dan teks. Tinjau konten kolom data atau masukan lainnya, dan periksa untuk memastikannya masuk akal. Misalnya, jika proyek melibatkan analisis tingkat pendapatan, pratinjau data untuk mengonfirmasi bahwa nilai pendapatan positif atau apakah dapat diterima untuk memiliki nilai nol atau negatif. Cari bukti kesalahan sistematis. Contohnya termasuk umpan data dari sensor atau sumber data lain yang rusak tanpa ada yang menyadarinya, yang menyebabkan nilai data tidak valid, salah, atau hilang. Tinjau data untuk mengukur apakah definisi data sama untuk semua pengukuran. Dalam beberapa kasus, kolom data digunakan kembali, atau kolom berhenti diisi, tanpa perubahan ini diberi anotasi atau tanpa orang lain diberitahu. Setelah tim mengumpulkan dan memperoleh setidaknya beberapa set data yang diperlukan untuk analisis selanjutnya, langkah yang berguna adalah memanfaatkan alat visualisasi data untuk melihat pola tingkat tinggi dalam data yang memungkinkan seseorang untuk memahami karakteristik tentang data dengan sangat cepat. Salah satu contohnya adalah menggunakan visualisasi data untuk memeriksa kualitas data, seperti apakah data tersebut berisi banyak nilai yang tidak diharapkan atau indikator lain dari data yang tidak akurat. Contoh lainnya adalah Skewness, seperti jika sebagian besar data sangat bergeser ke satu nilai atau akhir dari suatu kontinum. Visualisasi Data memungkinkan pengguna untuk menemukan area yang menarik, memperbesar, dan memfilter untuk

menemukan informasi yang lebih terperinci tentang area tertentu dari data, lalu menemukan data terperinci di balik area tertentu. Pendekatan ini memberikan tampilan tingkat tinggi dari data dan banyak informasi tentang set data tertentu dalam waktu yang relatif singkat.

3.2 PEDOMAN DAN PERTIMBANGAN

Tinjau data untuk memastikan bahwa perhitungan tetap konsisten dalam kolom atau di seluruh tabel untuk bidang data tertentu. Misalnya, apakah nilai seumur hidup pelanggan berubah di beberapa titik di tengah pengumpulan data? Atau jika bekerja dengan keuangan, apakah perhitungan bunga berubah dari sederhana menjadi majemuk di akhir tahun? Apakah distribusi data tetap konsisten di semua data? Jika tidak, tindakan apa yang harus diambil untuk mengatasi masalah ini? Nilai granularitas data, rentang nilai, dan tingkat agregasi data.

Apakah data mewakili populasi yang diminati? Untuk data pemasaran, jika proyek difokuskan pada penargetan pelanggan usia mengasuh anak, apakah data mewakili itu atau apakah penuh dengan warga senior dan remaja? Untuk variabel terkait waktu, apakah pengukurannya harian, mingguan, bulanan? Apakah itu cukup baik? Apakah waktu diukur dalam detik di mana-mana? Atau apakah dalam milidetik di beberapa tempat? Tentukan tingkat ketelitian data yang diperlukan untuk analisis, dan nilai apakah tingkat stempel waktu saat ini pada data memenuhi kebutuhan tersebut.

Apakah data tersebut terstandar/ternormalisasi? Apakah skalanya konsisten? Jika tidak, seberapa konsisten atau tidak teratur data tersebut? Ini adalah pertimbangan umum yang harus menjadi bagian dari proses berpikir saat tim mengevaluasi kumpulan data yang diperoleh untuk proyek tersebut. Menjadi sangat berpengetahuan tentang data akan menjadi penting saat tiba saatnya untuk membangun dan menjalankan model di kemudian hari dalam proses tersebut.

Periksa Bagian Data Dari Dataset

```
setwd("D:/R data")
Employee <- read.csv("Employee.csv") fix(Employee)
print(Employee)
```

Fungsi print() mencantumkan konten bingkai data (atau objek lainnya). Konten dapat diubah menggunakan fungsi edit().

```
edit(Employee)
```

Periksa Dimensi Data

Gunakan dim() untuk memperoleh dimensi bingkai data (jumlah baris dan jumlah kolom). Outputnya adalah vektor.

```
Dim(Employee)
```

Gunakan nrow() dan ncol() untuk memperoleh jumlah baris dan jumlah kolom. Kita dapat memperoleh informasi yang sama dengan mengekstrak elemen pertama dan kedua dari vektor output dari dim().

```
nrow(Employee) ncol(Employee)
```

Periksa Fitur dan pahami data dengan mencetak beberapa baris pertama menggunakan fungsi head(). Secara default R mencetak 6 baris pertama. Kita dapat menggunakan head() untuk memperoleh n observasi pertama dan tail() untuk memperoleh n observasi terakhir; secara default, n = 6. Ini adalah perintah yang bagus untuk memperoleh gambaran intuitif tentang seperti apa data tersebut tanpa mengungkapkan seluruh set data, yang dapat memiliki jutaan baris dan ribuan kolom.

```
head(Employee)
```

Jika kita ingin memilih hanya beberapa baris, kita dapat menentukan jumlah baris tersebut. Memilih Baris(Observations)

```
Samp <- Employee[1:3,] head(mydata) # 6 baris pertama dari set data
head(mydata, n = 10) # 10 baris pertama dari set data
head(mydata, n = -10) # Semua baris kecuali 10 terakhir
tail(mydata) # 6 baris terakhir
tail(mydata, n = 10) # 10 baris terakhir
tail(mydata, n= -10) # Semua baris kecuali 10 pertama
```

Nama variabel atau nama kolom

```
names(Employee)
```

Periksa Bagian Deskriptor Dari Dataset

Bagian deskriptor berarti metadata (data tentang data);

```
str(Karyawan)
```

Fungsi str() menyediakan struktur kerangka data. Fungsi ini mengidentifikasi tipe data integer dan numerik (ganda), variabel faktor dan level, serta beberapa nilai pertama untuk setiap variabel. Dengan menjalankan kode di atas, kita dapat memperoleh informasi tentang atribut Variabel termasuk Nama Variabel, Tipe, dll. “num” menunjukkan bahwa variabel “count” adalah numerik (kontinu), dan “Factor” menunjukkan bahwa variabel tersebut kategoris dengan kategori atau level.

Perintah sapply(Karyawan, kelas) akan mengembalikan nama dan kelas (misalnya, numerik, integer, atau karakter) dari setiap variabel dalam kerangka data.

```
sapply(Karyawan, kelas)
```

Untuk memperoleh semua kategori atau level dari variabel kategoris, gunakan fungsi levels().

```
levels(Karyawan)
```

3.3 MEMBUAT VISUALISASI

Kita biasanya menggunakan visualisasi data untuk eksplorasi data karena memungkinkan pengguna untuk melihat sebagian besar fitur yang relevan dari kumpulan data dengan cepat dan mudah. Visualisasi membantu kita mengidentifikasi variabel yang mungkin memiliki hubungan yang menarik. Dengan menampilkan data secara grafis melalui diagram sebar atau diagram batang, kita dapat melihat apakah dua atau lebih variabel berkorelasi dan menentukan apakah keduanya merupakan kandidat yang baik untuk analisis lebih mendalam. Cara yang berguna untuk memahami pola dan ketidakkonsistenan dalam data adalah melalui analisis data eksploratif dengan visualisasi. Visualisasi memberikan tampilan data yang ringkas yang mungkin sulit dipahami hanya dari angka dan ringkasan saja. Variabel x dan y dari data kerangka data dapat divisualisasikan dalam diagram atau plot yang dengan mudah menggambarkan hubungan antara dua variabel. Visualisasi membantu kita membuat berbagai jenis grafik seperti:

1. Histogram
2. Diagram Pai
3. Diagram Batang/Garis
4. Diagram Kotak
5. Diagram Sebar

Histogram: Anda dapat membuat histogram dengan fungsi hist(x) di mana x adalah vektor numerik nilai yang akan diplot. Opsi freq=FALSE memplot kerapatan probabilitas, bukan frekuensi. Opsi breaks= mengontrol jumlah bin.

```
# Histogram Sederhana
hist(Karyawan$Gaji)
# Histogram Berwarna dengan Jumlah Bin yang Berbeda
hist(Karyawan$Gaji, breaks=12, col=”merah”)
# Histogram yang dihamparkan
hist(Karyawan$Gaji, breaks=”FD”, col=”hijau”)
hist(Karyawan$Gaji [Karyawan$JenisKelamin==”Pria”], breaks=”FD”,
col=”abu-abu”, add=TRUE)
legend(“kanan atas”, c(“Wanita”,”Pria”), fill=c(“hijau”,”abu-abu”))
```

Diagram Pai: Diagram Pai dibuat dengan fungsi pie(x, labels=) di mana x adalah vektor numerik non-negatif yang menunjukkan luas setiap irisan dan labels= mencatat vektor karakter nama untuk irisan tersebut. # Bagan Pai Sederhana

```
Items_sold <- c(10, 12,4, 16, 8)
```

```
Lokasi <- c("Hyderabad", "Bangalore", "Kolkata", "Mumbai", "Delhi")
pie(Items_sold, labels = Location, main="Bagan Pai Lokasi")
# Bagan Pai 3D yang Dieksplorasi
library(plotrix)
pie3D(slices,labels=lbls,explode=0.1, main="Bagan Pai Negara")
```

Bagan Batang/Garis

Bagan Garis/Bagan Garis umumnya lebih disukai ketika kita menganalisis tren yang menyebar selama periode waktu tertentu. Bagan garis juga cocok untuk membuat bagan yang perlu membandingkan perubahan relatif dalam kuantitas di beberapa variabel (seperti waktu).

#Untuk membuat bagan garis sederhana:

```
Plot(, type=l)
```

Boxplots

Boxplots dapat dibuat untuk variabel individual atau untuk variabel berdasarkan grup. Formatnya adalah boxplot(x, data=), di mana x adalah rumus dan data= menunjukkan kerangka data yang menyediakan data. Contoh rumusnya adalah y~group di mana boxplot terpisah untuk variabel numerik y dibuat untuk setiap nilai grup. Tambahkan varwidth=TRUE untuk membuat lebar boxplot proporsional dengan akar kuadrat ukuran sampel. Tambahkanhorizontal=TRUE untuk membalik orientasi sumbu.

Boxplot Gaji Berdasarkan Pendidikan

boxplot(Gaji~Pendidikan,data=Karyawan, main="Gaji Berdasarkan Pendidikan", xlab="Pendidikan", ylab="Gaji")

boxplot adalah kata kunci untuk membuat boxplot. Plot dibuat antara Gaji Karyawan dan Tingkat Pendidikan. Keberadaan outlier dalam set data diamati sebagai titik di luar kotak.

Scatterplot: Ada banyak cara untuk membuat scatterplot di R. Fungsi dasarnya adalah plot(x, y), di mana x dan y adalah vektor numerik yang menunjukkan titik (x,y) yang akan diplot.

```
# Scatterplot Sederhana attachment(Karyawan)
plot(Usia, Gaji, main="Scatterplot pada Usia vs Gaji",
xlab="Usia", ylab="Gaji ", pch=19)
```

3.4 DATA YANG HILANG

Mari kita lihat bagaimana data yang hilang dapat dideteksi dalam fase eksplorasi data dengan visualisasi. Secara umum, analis harus mencari anomali, memverifikasi data dengan pengetahuan domain, dan memutuskan pendekatan yang paling tepat untuk membersihkan data. Pertimbangkan skenario di mana bank melakukan analisis data pemegang rekeningnya untuk mengukur retensi pelanggan.

`rowSums(is.na(mydata))` # Jumlah yang hilang per baris

```
colSums(is.na(mydata)) # Jumlah yang hilang per kolom/variabel # Ubah ke data yang hilang
mydata[mydata$age==”& “,”age”] <- NA mydata[mydata$age==999,”age”] <- NA
```

Fungsi complete.cases() mengembalikan vektor logis yang menunjukkan kasus mana yang lengkap.

daftar baris data yang memiliki nilai yang hilang

```
mydata[!complete.cases(mydata),]
```

Fungsi na.omit() mengembalikan objek dengan penghapusan nilai yang hilang secara berurutan. # Membuat kumpulan data baru tanpa data yang hilang

```
mydata1 <- na.omit(mydata)
```

Nilai Ekstrem: Pengamatan ekstrem adalah yang menarik dan layak mendapat perhatian kita karena lebih dari sekadar outlier normal di ujung kurva lonceng. Ini adalah yang mendistorsi distribusi ke bentuk F yang ditunjukkan sebelumnya.

Kotak Diagram untuk Mendeteksi Outlier: Outlier adalah skor yang sangat berbeda dari data lainnya. Ketika kita menganalisis data, kita harus menyadari nilai-nilai tersebut karena mereka membiaskan model yang kita sesuaikan dengan data. Contoh yang baik dari bias ini dapat dilihat dengan melihat model statistik sederhana seperti mean. Misalkan sebuah film mendapat rating dari 1 sampai 5. Tujuh orang menonton film tersebut dan memberi rating film tersebut dengan rating 2, 5, 4, 5, 5, 5, dan 5. Semua rating ini, kecuali satu, cukup mirip (umumnya 5 dan 4), tetapi rating pertama cukup berbeda dari yang lainnya. Ratingnya 2.

Ini adalah contoh outlier. Kotak-kotak memberi tahu kita sesuatu tentang distribusi skor. Kotak-kotak menunjukkan yang terendah (garis horizontal bawah) dan tertinggi (garis horizontal atas). Jarak antara garis horizontal terendah dan tepi terendah kotak berwarna adalah rentang di mana 25% skor terendah jatuh (disebut kuartil bawah). Kotak (area berwarna) menunjukkan 50% skor tengah (dikenal sebagai rentang interkuartil); yaitu 50% skor lebih besar dari bagian terendah area berwarna tetapi lebih kecil dari bagian atas area berwarna. Jarak antara tepi atas kotak berwarna dan garis horizontal atas menunjukkan rentang di mana 25% skor teratas jatuh (kuartil atas). Di tengah kotak berwarna ada garis horizontal yang sedikit lebih tebal. Ini mewakili nilai median. Seperti histogram, mereka juga memberi tahu kita apakah distribusinya simetris atau miring. Untuk distribusi simetris, kumis di kedua sisi kotak memiliki panjang yang sama. Terakhir, Anda akan melihat beberapa lingkaran kecil di atas setiap diagram kotak.

Ini adalah kasus yang dianggap sebagai outlier. Setiap lingkaran memiliki angka di sebelahnya yang memberi tahu kita di baris editor data mana kasus tersebut harus ditemukan. Diagram kotak digunakan secara luas untuk memeriksa keberadaan outlier dalam set data. Dua fakta penting yang harus diingat untuk diagram kotak adalah

1. Jumlah observasi dalam set data harus setidaknya sebanyak lima.
2. Jika ada lebih dari satu kategori dalam set data, harus diurutkan menurut kategori.

Set data yang berisi nilai 5 siswa dalam mata pelajaran Bahasa Inggris dan Sains ada dalam format CSV. Diagram kotak adalah kata kunci untuk membuat diagram kotak. Plot dibuat

antara nilai yang diperoleh siswa dan subjek. Keberadaan outlier dalam set data diamati sebagai titik di luar kotak. Kami ingin fokus pada momen menarik di pinggiran, yang dikenal sebagai outlier dan mengapa hal itu bisa menjadi penting. Ketika outlier menjadi pengamatan ekstrem di kiri atau kanan, hal itu dapat mengubah asumsi yang dibuat oleh ahli statistik pada pengaturan studi tentang perilaku populasi yang direkrut, yang dapat membahayakan pembuktian survei dan akhirnya menyebabkan kegagalan yang mahal. Pengamatan ekstrem adalah yang menarik dan layak mendapat perhatian kita karena lebih dari sekadar outlier normal di ujung kurva lonceng. Ini adalah yang mendistorsi distribusi ke bentuk F.

BAB 4
PERSIAPAN DATA

4.1 PENDAHULUAN

Persiapan data mencakup langkah-langkah untuk mengeksplorasi, melakukan praproses, dan menyusun data sebelum pemodelan dan analisis. Memahami data secara terperinci sangat penting untuk keberhasilan proyek. Kita harus memutuskan cara mengondisikan dan mengubah data untuk memasukkannya ke dalam format yang memudahkan analisis selanjutnya. Kita mungkin perlu melakukan visualisasi data untuk membantu kita memahami data, termasuk tren, outlier, dan hubungan antar variabel data. Persiapan data cenderung menjadi langkah yang paling padat karya dan Faktanya, tim biasanya menghabiskan setidaknya 50% dari waktu proyek dalam fase kritis ini. Jika tim tidak dapat memperoleh cukup data dengan kualitas yang memadai, tim mungkin tidak dapat melakukan langkah-langkah selanjutnya dalam proses siklus hidup.

Persiapan Data mengacu pada proses pembersihan data, normalisasi set data, dan melakukan transformasi pada data. Sebagai langkah penting dalam Siklus Hidup Analisis Data, pengondisian data dapat melibatkan banyak langkah rumit untuk menggabungkan atau menyatukan set data atau sebaliknya mendapatkan set data ke dalam status yang memungkinkan analisis dalam fase selanjutnya. Pengondisian data sering kali dipandang sebagai langkah praproses untuk analisis data karena melibatkan banyak operasi pada kumpulan data sebelum mengembangkan model untuk memproses atau menganalisis data.
Langkah-langkah persiapan data meliputi:

1. Membuat Variabel Baru.
2. Mengelompokkan Data, Menghapus observasi duplikat dalam kumpulan data.
3. Memformat.
4. Menyimpan, Menghapus, Mengganti Nama, Memberi Label.
5. Pemrosesan Bersyarat.
6. Fungsi.
7. Menggabungkan Kumpulan Data.
8. Mentransposisi Data.

4.2 MEMBUAT VARIABEL BARU

Mari kita ambil dataset Karyawan, kita memiliki rincian Pendapatan bulanan mereka, kita ingin menghitung kenaikan Gaji mereka dan menghitung Gaji Baru.
Kita menggunakan operator penugasan (<-) untuk membuat variabel baru.

```
setwd("D:/R data")
Karyawan <- read.csv("Karyawan.csv") fix(Karyawan)
```

Buat variabel bernama Bonus dengan kenaikan gaji sebesar 8%

```
Karyawan$Bonus <- Karyawan$Gaji*.08 Karyawan$Berita <-
Karyawan$Gaji + Karyawan$Bonus fix(Karyawan)
```

4.3 MENGURUTKAN DATA

Untuk mengelompokkan bingkai data di R, gunakan fungsi order(). Secara default, pengurutan adalah ASCENDING. Tambahkan variabel pengurutan di awal. Dengan tanda minus untuk menunjukkan urutan DESCENDING.

Urutkan berdasarkan Usia

```
Agesort <- Employee[order(Usia),]
```

#Pengurutan dengan Beberapa Variabel, urutkan berdasarkan Jenis Kelamin dan Usia

```
Mul_sort <- Employee[order(Jenis Kelamin, Usia),]
```

Dengan menjalankan kode di atas, kita mengurutkan bingkai data berdasarkan Jenis Kelamin sebagai preferensi pertama dan Usia sebagai preferensi kedua.

#Urutkan berdasarkan Jenis Kelamin (ascending) dan Usia (descending)

```
Rev_sort <- Employee[order(Jenis Kelamin, -Usia),] detach(Employee)
```

4.4 MENGIDENTIFIKASI DAN MENGHAPUS DATA YANG DUPLIKASI

Kita dapat menghapus data duplikat menggunakan fungsi duplicated() dan unique() serta fungsi distinct dalam paket dplyr. Fungsi duplicated() mengembalikan vektor logika yang TRUE-nya menentukan elemen mana dari vektor atau bingkai data yang merupakan duplikat. Diberikan vektor berikut:

```
Cust_Id <- c(101, 104, 104, 105, 104, 105)
```

Untuk menemukan posisi elemen duplikat di x, kita dapat menggunakan ini:

```
duplicated(Cust_Id)
```

Kita dapat mencetak elemen duplikat dengan menjalankan kode berikut.

```
Cust_Id [duplicated(Cust_Id)]
```

Jika Anda ingin menghapus elemen duplikat dan hanya mendapatkan nilai unik, gunakan !duplicated(), di mana ! adalah negasi logika:

```
Uniq_Cust<- Cust_Id [!duplicated(Cust_Id)]
```

Dalam tugas sehari-hari, kita perlu membuat, memodifikasi, memanipulasi, dan mengubah data agar data siap untuk Analisis dan Pelaporan. Kita menggunakan beberapa Fungsi untuk sebagian besar manipulasi data. Keakraban dengan fungsi-fungsi ini dapat membuat pemrograman jauh lebih mudah. Kita dapat menghapus baris duplikat dari kerangka data berdasarkan nilai kolom, sebagai berikut:

```
# Hapus duplikat berdasarkan kolom Work_Balance
Uniq_WB <- Employee [!duplicated(Employee$ Work_Balance), ]
```

Anda dapat mengekstrak elemen unik sebagai berikut:

```
unique(Cust_Id)
```

Anda juga dapat menerapkan unique() pada bingkai data, untuk menghapus baris duplikat sebagai berikut:

```
unique(Employee)
```

Fungsi distinct() dalam paket dplyr dapat digunakan untuk menyimpan hanya baris unik/berbeda dari bingkai data. Jika ada baris duplikat, hanya baris pertama yang dipertahankan. Ini adalah versi efisien dari fungsi dasar R unique().

Paket dplyr dapat dimuat dan diinstal sebagai berikut:

```
install.packages("dplyr") library("dplyr")
```

Hapus baris duplikat berdasarkan semua kolom:

```
distinct(Employee)
```

Hapus baris duplikat berdasarkan kolom (variabel) tertentu: Hapus baris duplikat berdasarkan JobSatisfaction.

```
discrect(Employee, JobSatisfaction)
```

Hapus baris duplikat berdasarkan JobSatisfaction dan Perf_Rating.

```
discrect(Employee, JobSatisfaction, Perf_Rating)
```

4.5 MEMFILTER PENGAMATAN BERDASARKAN KONDISI

```
Age_Con <- Employee[which(Employee$Age < 40), ]
```

Saat memfilter pengamatan berdasarkan variabel karakter, kita perlu menyematkan string dalam tanda kutip.

```
Sex_Con <- Employee[which(Employee$Gender ="male"), ]
```

Filter Pengamatan berdasarkan beberapa kondisi

```
Mul_Con <-Employee[which(Employee $Gender=='Female' & Employee
$Age < 30),]
```

Pemilihan menggunakan Fungsi Subset

Fungsi subset() adalah cara termudah untuk memilih variabel dan pengamatan. Dalam contoh berikut, kita memilih semua baris yang memiliki nilai age lebih besar dari atau sama dengan 50 atau age kurang dari 30. Kita mempertahankan kolom Emp_Id dan Age.

Menggunakan fungsi subset

```
Test <- subset(Employee, Age >= 50 | Age < 30, select=c(Emp_Id,
Age))
```

Pemrosesan Bersyarat:

Sering kali kita perlu memproses data berdasarkan kondisi tertentu, dan pengambilan keputusan merupakan bagian penting dari pemrograman. Hal ini dapat dicapai dengan menggunakan pernyataan if...else bersyarat.

Sintaks Pernyataan If:

```
if (test_expression) { statement }
```

Kita dapat membuat variabel baru menggunakan if then else. Kita ingin mempromosikan produk kita hanya kepada orang-orang yang berpenghasilan lebih dari 40K.

```
Employee$Promosi <- ifelse(Employee$Gaji>40000,"Promosikan
Produk","Jangan Promosikan Produk")
fix(Karyawan)
```

#Di sini, Kita memeriksa apakah elemen Employee$Gaji lebih besar dari 40000. Jika suatu elemen lebih besar dari 40000, maka ia akan menetapkan nilai Promosikan Produk ke Employee$Promosi, dan jika tidak lebih besar dari 40000, maka ia akan menetapkan nilai Jangan Promosikan Produk ke Employee$Promosi.

Kita ingin menetapkan nilai Rendah, Sedang, dan Tinggi ke variabel Sal_Grp. Untuk melakukan ini, kita dapat menggunakan pernyataan ifelse() bersarang:

```
Employee$Sal_Grp      <-ifelse(Employee$Salary      >20000      &
Employee$Salary<50000," Medium",
ifelse(Employee$Salary >= 50000, "High","Low"))
```

Sekarang ini mengatakan, pertama-tama periksa apakah setiap elemen vektor Salary >20000 dan <50000. Jika ya, tetapkan Medium ke Sal_Grp. Jika tidak, maka evaluasi pernyataan ifelse() berikutnya, apakah Salary>50000. Jika ya, tetapkan Sal_Grp nilai High. Jika tidak ada di antara keduanya, maka tetapkan Low.

4.6 FORMATTING

Ini umumnya digunakan untuk meningkatkan tampilan output, kita dapat menggunakan format yang sudah ada (yang ditentukan Sistem) dan bahkan membuat format khusus untuk mengelompokkan nilai dengan cara yang efisien. Dengan menggunakan ini, kita dapat mengelompokkan data dengan berbagai cara tanpa harus membuat set data baru. Bayangkan, kita melakukan survei pada Produk Baru (A/c): Secara keseluruhan puas - 1 - Sangat rendah 2- Rendah 3. Oke, 4- Baik 5- Sangat Baik. Meskipun ada banyak cara untuk menulis solusi khusus, analis harus terbiasa dengan prosedur tujuan khusus yang dapat mengurangi kebutuhan untuk pengkodean khusus. R akan memperlakukan faktor sebagai variabel nominal dan faktor yang diurutkan sebagai variabel ordinal. Anda dapat

menggunakan opsi dalam fungsi factor() dan ordered() untuk mengontrol pemetaan bilangan bulat ke string.
Kita dapat menggunakan fungsi factor untuk membuat label nilai Anda sendiri.

```
setwd("D:/R data")
Shop <- read.csv("Shopping.csv") fix(Shop)
```

Variabel merek dalam set data Shop dikodekan 1, 2, 3.: Kita ingin melampirkan label nilai 1=Samsung, 2=Hitachi, 3=Bluestar.

```
Shop$Brand <- factor(Shop$Brand, levels = c(1,2,3), labels =
c("Samsung", "Hitachi", "Bluestar"))
```

variabel y dikodekan 1,2, 3,4 dan 5 # kita ingin melampirkan label nilai 1 - Sangat rendah 2-Rendah 3-Oke 4-Baik 5-Sangat Baik.

```
Shop$Overall_Sat <- ordered(Shop$Overall_Sat, levels =
c(1,2,3,4,5), labels = c("Sangat Rendah", "Rendah", "Oke",
"Baik", "Sangat Baik"))
```

Terkadang Anda mungkin ingin membuat variabel kategoris baru dengan mengklasifikasikan observasi menurut nilai variabel kontinu. Misalkan Anda ingin membuat variabel baru bernama Age.Cat, yang mengklasifikasikan orang sebagai "Muda", "Dewasa", dan "Tua" menurut Usia mereka. Orang yang berusia kurang dari 35 tahun digolongkan sebagai Muda, orang yang berusia antara 35 dan 60 tahun digolongkan sebagai Dewasa, dan orang yang berusia lebih dari 60 tahun digolongkan sebagai Tua.

```
Employee$Age.Cat<-cut(Employee$Age, c(18,35, 60,90), c("Muda",
"Dewasa", "Tua"))
```

4.7 MENYIMPAN, MENGHAPUS, MENGGANTI NAMA, MEMBERI LABEL

Pilih variabel

```
myvars <- c("Merek", "Keamanan", "Tampilan") Sub_shop <-
Shop[myvars] fix(Sub_shop)
```

Kecualikan variabel ke-4 dan ke-6

```
Sub_data <- Shop[c(-4,-6)] fix(Sub_data)
```

Ubah nama secara interaktif

```
fix(Toko)
```

hasil disimpan saat ditutup # Ubah nama secara terprogram library(reshape)

```
Ren_Shop <- rename(Toko, c(Keamanan="Keamanan"))
```

Memberi Label pada Variabel: Kita dapat menetapkan label variabel di var.labels ke kolom-kolom dalam data bingkai data menggunakan fungsi label dari Paket Hmisc.

```
install.packages("Hmisc") library("Hmisc")
```

```
label(Shop[["Overall_Sat"]]) <- "Kepuasan Pelanggan Secara
Keseluruhan" label(Shop[["Tampilan"]]) <- "Tampilan dan Nuansa
Produk"
label(Shop)
```

4.8 FUNGSI

Fungsi mengembalikan nilai dari perhitungan atau manipulasi sistem yang memerlukan nol atau lebih argumen. Fungsi dibuat dengan menggunakan kata kunci function. Sintaks dasar definisi fungsi R adalah sebagai berikut:

New_Var <- Function_Name(Argument1, Argument2,...N).

Fungsi dikenali dengan penggunaan nama fungsi, diikuti langsung oleh argumen fungsi, dipisahkan dengan koma, dan diapit tanda kurung. Namun, jumlah argumen yang diperlukan dan opsional bervariasi. Beberapa fungsi hanya memiliki satu argumen yang diperlukan. Yang lain memiliki satu argumen yang diperlukan dan satu atau lebih argumen opsional. Dalam kebanyakan kasus, urutan argumen penting. Beberapa fungsi tidak mengambil argumen, dalam hal ini diperlukan satu set tanda kurung null.

Fungsi Karakter:

Fungsi toupper, tolower: Fungsi ini mengubah huruf besar argumen. Nama <- 'Ramya Kalidindi' #Menetapkan nilai ke variabel

```
upcf <- toupper (Nama)
locf <- tolower(Nama)
```

Fungsi trimws: Sering kali, data yang kami terima mungkin berisi spasi yang tidak diinginkan, dan kami ingin menghapusnya agar data kami bersih. Kami menggunakan fungsi trimws untuk menangani spasi kosong pada string.

```
Nama <- " Y Lakshmi Prasad "
Nama_Trimmed <- trimws(Nama, which = c("both", "left", "right"))
```

Fungsi substr: Fungsi ini digunakan untuk Mengekstrak karakter dari variabel string. Argumen untuk substr() menentukan vektor input, posisi karakter awal dan posisi karakter akhir. Parameter terakhir bersifat opsional. Jika dihilangkan, semua karakter setelah lokasi yang ditentukan di spasi kedua akan diekstrak. `Req_Name <- substr(Name, 6, 12)`

Buat variabel string dari variabel numerik `stringx <- as.character(numericx)`

```
typeof(stringx) typeof(numericx)
```

Fungsi typeof() dapat digunakan untuk memverifikasi tipe objek, nilai yang mungkin termasuk logika, integer, double, kompleks, karakter, raw, list, NULL, closure (fungsi), spesial, dan bawaan.

Buat variabel numerik dari variabel string: Argumen dalam fungsi as.numeric, integer adalah jumlah karakter dalam string, sedangkan desimal adalah spesifikasi opsional tentang berapa banyak karakter yang muncul setelah desimal.

```
numericx <- as.numeric(stringx)
typeof(stringx) typeof(numericx)
```

Tugas: Hitung kenaikan sebesar 10% dan dapatkan gaji baru untuk setiap Id.

```
Num_data <- data.frame(Id = c(101,102,103), Salary
=c(40700,12000,37000))
sapply(Num_data, mode)
Num_data$Char_sal <- as.character(Num_data$Salary)
typeof(Num_data$Char_sal)
Num_data$Saln <- as.numeric(Num_data$Char_sal)
typeof(Num_data$Saln)
Num_data$Bonus <- Num_data$Saln*.10 fix(Num_data)
Num_data$New_Sal <- Num_data$Saln+Num_data$Bonus
```

Fungsi Numerik:

Fungsi Abs: Kita menggunakan fungsi ini untuk Mendapatkan nilai absolut; a=5

```
aba <- abs(a) Val <- -28.86
Req <- abs(Val))
```

Nilai Floor(Base) dan ceiling(Top):

```
X=43.837 #Nilai ini terletak di antara 43 dan 44
Flx=floor(X) #Nilai Base
Cilx=ceiling(X) #Nilai Top
```

Fungsi Round:

```
Val=43.837
Rval1=round(Val) #44
Rval2=round(Val,digits=2) #43.84
```

Fungsi MAX: Mengembalikan nilai terbesar yang tidak hilang dari daftar

```
X=max(2,6,NA,8,0)
```

Fungsi Min: Mengembalikan nilai terkecil yang tidak hilang dari daftar

```
Y=min(2,6,1,7,0,-2)
```

Fungsi Sum: Mengembalikan Jumlah (Total) dari nilai

```
Y=Sum(9,8,.,9)
```

Mean Fungsi: Dihitung dengan menjumlahkan nilai dan membaginya dengan jumlah nilai dalam rangkaian data. Fungsi mean() digunakan untuk menghitungnya di R.

Buat vektor dan cari mean-nya.

```
x <- c(7,3,NA,4,18,2,54,8)
res_mean <- mean(x) print(res_mean)
```

Cari mean dengan mengurangi nilai NA.

```
resmean_na <- mean(x,na.rm = TRUE) print(resmean_na)
```

Fungsi Median: Nilai paling tengah dalam rangkaian data disebut median. Fungsi median() digunakan di R untuk menghitung nilai ini.

Buat vektor dan Cari median.

```
x <- c(12,7,3,4.2,18,2,54,-21,8,-5)
median.result <- median(x) print(median.result)
```

Fungsi Tanggal/Waktu: Fungsi Tanggal/Waktu adalah serangkaian fungsi yang mengembalikan bagian dari nilai tanggal waktu, tanggal, atau waktu atau mengonversi nilai numerik menjadi nilai tanggal atau waktu R. Fungsi ini berguna untuk mengekstrak tanggal dan waktu dari nilai tanggal waktu atau mengonversi nilai bulan, hari, dan tahun yang terpisah menjadi nilai tanggal R.

Fungsi Sys.Date, date: Fungsi ini mengembalikan tanggal hari ini dari jam sistem, yang tidak memerlukan Argumen.

`Sys.Date()` mengembalikan tanggal hari ini.

`date()` mengembalikan tanggal dan waktu saat ini.

cetak tanggal hari ini

```
today <- Sys.Date()
format(today, format="%B %d %Y") format(today, format="%m %d
%Y") format(today, format="%m %d %y")
```

Anda dapat mengamati bahwa tanggal sistem ditampilkan dengan cara yang berbeda saat kita mengubah format.

Kita perlu menemukan format yang lebih sesuai dengan kebutuhan kita dan melakukannya.

Mengonversi Karakter ke Tanggal: Anda dapat menggunakan fungsi `as.Date()` untuk mengonversi data karakter ke tanggal. Formatnya adalah `as.Date(x,"format")`, di mana x adalah data karakter dan format memberikan format yang sesuai. Format default adalah yyyy-mm-dd

```
Testdts <- as.Date(c("1982-07-12", "1975-03-01"))
```

Gunakan as.Date() untuk mengonversi string menjadi tanggal

```
Testdts <- as.Date(c("1982-07-12", "1975-03-01"))
```

jumlah hari antara 7/12/82 dan 03/01/75

```
hari <- Testdts [1] - Testdts [2]
```

4.9 MENGGABUNGKAN SET DATA

Membaca data dari dua atau lebih set data dan memprosesnya dengan Menambahkan Baris, Menambahkan Kolom, Menggabungkan.

Menambahkan Baris: Menggabungkan set data pada dasarnya berarti menumpuk satu set data di atas yang lain, yaitu, jika diberikan dua set data, semua rekaman dari set data kedua akan ditambahkan ke akhir set data pertama. Saat menggabungkan set data, kita akan mengharapkan set data memiliki struktur yang identik tetapi konten yang berbeda. Yang kami maksud dengan struktur adalah tabel akan memiliki nama kolom yang sama dan kolom akan memiliki tipe yang sama (numerik atau karakter). Jika kolom ada di satu atau lebih set data tetapi tidak ada di yang lain, kolom tersebut disertakan dalam semua rekaman keluaran tetapi dengan nilai yang hilang untuk semua rekaman di tabel yang tidak memiliki kolom tersebut.

Fungsi rbind memungkinkan Anda untuk melampirkan satu set data ke bagian bawah yang lain, yang dikenal sebagai menambahkan atau menggabungkan set data. Ini berguna saat Anda ingin menggabungkan dua kumpulan data yang berisi observasi berbeda untuk variabel yang sama. Saat menggunakan fungsi rbind, kita perlu memastikan bahwa setiap kumpulan data berisi jumlah variabel yang sama dan semua nama variabel cocok. Anda mungkin perlu menghapus atau mengganti nama beberapa variabel dalam proses tersebut. Variabel tidak perlu disusun dalam urutan yang sama dalam kumpulan data, karena fungsi rbind secara otomatis mencocokkannya berdasarkan nama. Fungsi rbind tidak mengidentifikasi duplikat atau mengurutkan data. Anda dapat melakukannya dengan fungsi unik dan urutan.

```
Sale1 <- data.frame(Id_Pelanggan = c(101,103,105),
Jumlah_Pengeluaran = c(1700,1200,3700),
Pur_dt = c(20-03-2015,20-03-2015,20-03-2015))
fix(sale1)
sale2 <- data.frame(Id_Pelanggan = c(103,104,105,108),
Pur_dt = c(22-03-2015,22-03-2015,22-03-2015,22-03-2015),
Jumlah_Pengeluaran = c(1800,3400,2500,3200))
fix(sale2)
saleall <-rbind(sale1, sale2)
fix(saleall)
```

Perhatikan bahwa kumpulan data baru berisi semua data asli dalam urutan asli, termasuk dua salinan data untuk Cust_Id 103 dan 105. Kita dapat menambahkan dua atau lebih kumpulan data dengan cara yang sama.

Menambahkan Kolom: Fungsi cbind menempelkan satu set data ke sisi yang lain. Ini berguna jika data dari baris yang sesuai dari setiap set data termasuk dalam observasi yang sama. Anda hanya dapat menggunakan fungsi cbind untuk menggabungkan set data yang memiliki jumlah baris yang sama.

Menggabungkan Set Data dengan Variabel Umum: Fungsi merge memungkinkan Anda untuk menggabungkan dua set data dengan mencocokkan observasi menurut nilai variabel umum. Pertimbangkan
set data sale1 dan loc1. Set data memiliki variabel umum yang disebut Cust_Id, yang dapat digunakan untuk mencocokkan observasi yang sesuai.

```
loc1 <- data.frame(Cust_Id = c(101,102,103,104,105),
```

```
Lokasi
=c("Hyderabad","Bangalore","Chennai","Hyderabad","Bangalore"))
Merdata <- merge(sale1,loc1)
fix(Merdata)
```

Fungsi merge mengidentifikasi variabel dengan nama yang sama dan menggunakannya untuk mencocokkan observasi. Dalam contoh ini, kedua set data berisi variabel bernama Cust_Id, jadi R secara otomatis menggunakan variabel ini untuk mencocokkan observasi.

```
All_mer <- merge(sale1,loc1,all=T)
fix(All_mer)
```

Saat Anda menggabungkan dua set data dengan fungsi merge, R secara otomatis mengecualikan observasi yang tidak cocok yang hanya muncul di salah satu set data. Argumen all, all.x, dan all.y memungkinkan Anda mengontrol cara R menangani observasi yang tidak cocok. Untuk menyimpan semua pengamatan yang tidak cocok, tetapkan argumen all ke T:

```
L_mer <- merge(sale1,loc1,all.x=T) fix(L_mer)
R_mer <- merge(sale1,loc1,all.y=T) fix(R_mer)
```

Untuk menggabungkan dua bingkai data (himpunan data) secara horizontal, gunakan fungsi merge. Dalam kebanyakan kasus, Anda menggabungkan dua bingkai data dengan satu atau beberapa variabel kunci umum (misalnya, gabungan dalam).

gabungkan dua bingkai data dengan ID

```
total <- merge(data frameA,data frameB,by="ID")
```

Menambahkan Baris: Untuk menggabungkan dua bingkai data (himpunan data) secara vertikal, gunakan fungsi rbind. Dua bingkai data harus memiliki variabel yang sama, tetapi tidak harus dalam urutan yang sama.

```
total <- rbind(data frameA, data frameB)
```

4.10 MENGUBAH SET DATA

Membentuk ulang set data juga dikenal sebagai Memutar, Mengubah, atau Mengubah set data. Biasanya berlaku untuk set data yang telah dilakukan pengukuran berulang, fungsi mengubah bentuk digunakan untuk mengubah orientasi data. Namun, sebelum melakukan perubahan bentuk, kita perlu mengajukan pertanyaan berikut kepada diri kita sendiri:

Apa yang harus tetap sama Variabel mana yang harus naik

Variabel mana yang harus turun

Variabel mana yang harus berada di tengah

#Mengubah Data

```
setwd("D:/R data")
Vitals <- read.csv("Vitals.csv") fix(Vitals)
T_vitals<-reshape( Vitals, direction="wide", v.names="Result",
timevar="VS_Test", idvar="Pat_Id")
fix(T_vitals)
```

Karena Pat_Id ditentukan dalam idvar, ia tetap berada pada posisi yang sama, semua variabel lainnya diubah. Gunakan argumen v.names untuk menentukan variabel yang ingin Anda pisahkan ke dalam kolom yang berbeda. Gunakan argumen timevar untuk menentukan variabel yang menunjukkan kolom tempat nilai tersebut berada. Gunakan argumen idvar untuk menentukan variabel mana yang digunakan untuk mengelompokkan data bersama-sama.

BAB 5
PEMIKIRAN STATISTIK

5.1 PENDAHULUAN

Statistik adalah ilmu yang berhubungan dengan pengumpulan, Klasifikasi, analisis, dan Interpretasi fakta numerik dan penggunaan teori probabilitas untuk memaksakan keteraturan pada agregat data. Mari kita lihat beberapa Masalah Bisnis yang dihadapi oleh seorang Pengusaha, di mana ia membutuhkan Metode Statistik untuk menyelesaikannya.

Di mana kita harus membuka toko ritel baru kita?
Seberapa besar tempat yang harus kita sewa?
Berapa banyak orang yang harus saya pekerjakan untuk toko ini?
Berapa tingkat inventaris yang tepat untuk setiap produk?
Bagaimana cara meningkatkan nilai pelanggan dan pendapatan keseluruhan?
Bagaimana cara mengembangkan produk baru yang sukses?
Haruskah kita menerima pesanan daring atau tidak?
Berapa banyak yang harus kita investasikan dalam periklanan?
Bagaimana cara mengurangi biaya operasional?

5.2 ISTILAH STATISTIK

Sebelum menyelesaikan masalah yang disebutkan di atas, mari kita lihat beberapa istilah statistik yang membuat kita nyaman untuk menghadapi skenario ini.

Populasi: Populasi adalah sekumpulan Item lengkap yang memiliki setidaknya satu kesamaan.

Sampel: Subset dari populasi yang dipilih untuk analisis. Parameter: Ukuran yang dihitung pada seluruh populasi

Statistik: Ukuran yang dihitung pada sampel.

Statistik Deskriptif: Ini digunakan untuk mendeskripsikan atau meringkas data dengan cara yang bermakna dan berguna. Kita dapat mendeskripsikan data dengan berbagai cara seperti ukuran kecenderungan sentral, Ukuran penyebaran, ukuran lokasi dan bentuk distribusi. Ukuran Deskriptif memberikan pemahaman yang lebih baik tentang data dan dapat menyajikan gambaran keseluruhan data, statistik ini meliputi Rata-rata, Modus, Median, Minimum, Maksimum, Varians, Simpangan baku, Skewness, Kurtosis, dll.

Statistik Inferensial: Metode yang menggunakan teori probabilitas untuk menyimpulkan sifat-sifat populasi dari analisis sifat-sifat sampel data yang diambil darinya.

Statistik Prediktif: Metode yang berkaitan dengan prediksi probabilitas masa depan berdasarkan data historis.

Statistik Preskriptif: Metode memungkinkan kita untuk menentukan sejumlah tindakan yang mungkin dan membimbing kita menuju solusi yang optimal.

Variabel Acak: Variabel yang nilainya dapat berubah karena peluang.

Bias: Memberikan preferensi yang tidak adil pada satu hal dibandingkan yang lain.

Variabel: Variabel adalah karakteristik atau atribut yang dapat diukur atau dihitung.

Data dapat diklasifikasikan menjadi 2 Jenis:

1. *Data Kualitatif:* Jika kita dapat mengelompokkan data ke dalam sejumlah kelompok, kita menyebut data tersebut sebagai data Kualitatif. Jika tidak ada urutan antara kategori, kita menyebut variabel tersebut sebagai variabel Nominal. Jika kategori dapat diurutkan, kita menyebut variabel tersebut sebagai variabel ordinal.
2. *Data Kuantitatif:* Ini adalah pengukuran yang dinyatakan dalam angka, tetapi tidak semua angka bersifat kuantitatif seperti nomor ponsel dan kode pos, yang tidak dapat kita tambahkan atau kurangi.

5.3 SKALA PENGUKURAN

Ini adalah cara untuk mengkategorikan berbagai jenis variabel.

Skala Nominal: Skala ini memenuhi sifat identitas pengukuran. Mari kita ambil Gender sebagai contoh. Individu dapat diklasifikasikan sebagai "laki-laki" atau "perempuan", tetapi tidak ada nilai yang mewakili "gender" lebih atau kurang dari yang lain. Agama dan ras adalah contoh lain dari variabel yang biasanya diukur pada skala nominal.

Skala Ordinal: Skala ini memiliki sifat identitas dan besaran. Setiap nilai pada skala ordinal memiliki makna yang unik, dan memiliki hubungan yang teratur dengan setiap nilai lain pada skala tersebut. Mari kita ambil contoh bagaimana Anda menilai film tersebut?. Kita mendapatkan tanggapan Sangat baik, baik, Rata-rata, Buruk, dll.

Skala Interval: Skala ini memiliki sifat identitas, besaran, dan interval yang sama. Contoh sempurna dari skala interval adalah skala Fahrenheit untuk mengukur suhu. Skala ini terdiri dari unit suhu yang sama sehingga perbedaan antara 40 dan 50 derajat Fahrenheit sama dengan perbedaan antara 50 dan 60 derajat Fahrenheit. Dengan skala interval, Anda juga mengetahui seberapa besar atau kecilnya.

Skala Rasio: Skala pengukuran ini memenuhi keempat sifat pengukuran: identitas, besaran, interval yang sama, dan nol mutlak. Misalnya, jika berat suatu benda adalah 80 kilogram, kita dapat mengatakan bahwa benda ini dua kali lipat dari benda yang beratnya 40 kilogram. Variabel seperti Tinggi, Usia, Berat memiliki makna yang unik, dapat diurutkan, unit sepanjang skala sama satu sama lain, dan ada nol mutlak.

5.4 TEKNIK SAMPEL

Teknik pengambilan sampel adalah metode yang digunakan untuk mengambil sampel dari populasi. Ada berbagai metode untuk pengambilan sampel. Statistik sampel adalah karakteristik sampel, statistik sampel dapat digunakan sebagai estimasi titik untuk parameter populasi.

Memilih Sampel Acak Sederhana (SRS):

Tidak bias: Setiap unit memiliki peluang yang sama untuk dipilih dalam sampel.

Independen: Pemilihan satu unit tidak memengaruhi pemilihan unit lain.

SRS adalah standar emas yang digunakan untuk mengukur semua sampel lainnya.

Memilih Kerangka Sampel:
Kerangka sampel hanyalah daftar item untuk mengambil sampel.
Apakah kerangka sampel mewakili populasi?
Daftar yang tersedia mungkin berbeda dari daftar yang diinginkan:
Misalnya kami tidak memiliki daftar pelanggan yang tidak membeli dari toko.
Terkadang, tidak ada kerangka sampel yang komprehensif:
Saat meramalkan masa depan. Jadi, daftar lengkap penerimaan tawaran kartu kredit belum ada.

Kelemahan Umum dalam Pengambilan Sampel:

Mengumpulkan data hanya dari sukarelawan (sampel respons sukarela): misalnya ulasan daring (maps.google.com, tripadvisor.com)
Memilih responden yang mudah didapatkan (sampel praktis): – misalnya memilih untuk melakukan survei di mal In-Orbit
Tingkat nonrespons yang tinggi (lebih dari 70%): – mis. survei CEO/CIO tentang beberapa tren industri

Variasi sampel:

Rata-rata sampel bervariasi dari satu sampel ke sampel lainnya
Rata-rata sampel dapat (dan kemungkinan besar) berbeda dari rata-rata populasi
Rata-rata sampel adalah variabel acak
Teorema Batas Pusat (CLT) & distribusi rata-rata sampel:
Distribusi rata-rata sampel akan normal ketika distribusi data dalam populasi normal. Jika tidak, kami menganggapnya mendekati normal bahkan jika distribusi data dalam populasi tidak normal jika ukuran sampel "cukup besar". CLT Valid Ketika setiap titik data dalam sampel independen dari yang lain dan ukuran sampel cukup besar.
Seberapa Besar yang Cukup Besar?
Itu tergantung pada distribusi data – terutama simetrinya dan keberadaan outlier
Jika data cukup simetris dan memiliki sedikit outlier, bahkan sampel yang lebih kecil pun baik-baik saja. Jika tidak, kita perlu sampel yang lebih besar
Ukuran sampel 30 dianggap cukup besar, tetapi itu mungkin/mungkin tidak memadai

Distribusi Sampel dan Teorema Batas Pusat:

Berapa banyak pelanggan baru yang akan saya peroleh jika saya membuka toko di area ini?
Berapa tingkat persediaan yang tepat untuk e-reader baru kita?
Apa dampak kehabisan stok pada perilaku konsumen?
Berapa tingkat bunga yang harus kita tetapkan untuk pinjaman ini?
Apakah kualitas kita akan meningkat setelah penugasan konsultasi?
Berapa jumlah waktu yang dihabiskan oleh calon pelanggan kita di web?
Apakah waktu tunggu pesanan kita berkurang setelah penggabungan?
Berapa banyak pinjaman seperti itu yang gagal bayar di masa lalu?
Berapa jumlah jam kerja yang dibutuhkan untuk menyelesaikan proyek seperti itu?

Pengantar Teori Probabilitas: Probabilitas digunakan di seluruh bisnis untuk mengevaluasi risiko pengambilan keputusan. Setiap keputusan yang kita buat memiliki beberapa kemungkinan kegagalan, jadi analisis probabilitas dilakukan secara formal dan informal.

Sebagian besar dari kita menggunakan probabilitas dengan dua kondisi:

1. Kapan satu peristiwa atau lainnya akan terjadi
2. Di mana dua atau lebih peristiwa akan terjadi bersamaan

Mari kita pahami ini dari contoh toko perhiasan, pada Hari Raya. Berapa probabilitas permintaan hari ini akan melebihi penjualan rata-rata kita? Berapa probabilitas permintaan akan melebihi penjualan rata-rata kita dan lebih dari 20% tenaga penjualan kita tidak akan masuk kerja?

Variabel Acak:

Variabel acak menggambarkan probabilitas untuk hasil numerik masa depan yang tidak pasti dari suatu proses acak.

Variabel ini adalah variabel karena dapat mengambil salah satu dari beberapa nilai yang mungkin. Variabel ini acak karena ada beberapa peluang yang terkait dengan setiap nilai yang mungkin.

Independen: Ketika nilai yang diambil oleh satu variabel acak tidak memengaruhi nilai yang diambil oleh variabel acak lainnya: misalnya, lemparan dua dadu.

Dependen: Ketika nilai dari satu variabel acak memberi kita informasi lebih banyak tentang variabel acak lainnya: misalnya Tinggi dan berat siswa.

5.5 TEORI KEMUNGKINAN

Pendekatan Klasik: Probabilitas suatu kejadian sama dengan Jumlah hasil di mana kejadian tersebut terjadi dibagi dengan jumlah total kemungkinan hasil.

Pendekatan Frekuensi Relatif: Ketika melempar koin, awalnya rasio sejumlah sisi kepala terhadap sejumlah percobaan akan tetap tidak stabil. Ketika jumlah percobaan meningkat, rasio tersebut menyatu menjadi angka tetap (misalkan 0,5).

Pendekatan Probabilitas Subjektif: Pendekatan ini didasarkan pada pengalaman masa lalu dan intuisi individu. Sebagian besar keputusan manajerial berkaitan dengan situasi yang spesifik dan unik.

Distribusi Probabilitas: Distribusi probabilitas adalah aturan yang mengidentifikasi kemungkinan hasil dari variabel acak dan menetapkan probabilitas untuk masing-masing.

Distribusi diskrit memiliki jumlah nilai yang terbatas: misalnya nilai nominal kartu, pengalaman kerja siswa dibulatkan ke bulan terdekat.

Distribusi kontinu memiliki semua nilai yang mungkin dalam beberapa rentang: misalnya penjualan per bulan di toko eceran, detak jantung pasien di rumah sakit. Distribusi kontinu lebih mudah dipahami dan merupakan perkiraan yang baik ketika ada sejumlah besar nilai yang mungkin

Distribusi Probabilitas Diskrit: Misalkan Anda secara acak memilih kartu dari setumpuk kartu. Berapa probabilitas kartu ini akan

Lebih besar dari 7?

Sama dengan atau lebih besar dari 6?

Lebih kecil dari 3?

Lebih besar dari 4 dan kurang dari 8?

Penjualan harian TV layar datar besar di sebuah toko (X):
Berapa probabilitas penjualan?
Berapa probabilitas penjualan setidaknya tiga TV?
Nilai Harapan atau Rata-rata: Nilai harapan atau rata-rata (μ) dari variabel acak adalah rata-rata tertimbang dari nilai-nilainya, probabilitas berfungsi sebagai bobot. Berapa jumlah rata-rata Jam tangan yang terjual per hari?
Varians dan Simpangan Baku: Keduanya merupakan ukuran variasi atau ketidakpastian dalam variabel acak.
Varians (σ2): Rata-rata tertimbang dari deviasi kuadrat dari rata-rata, Probabilitas berfungsi sebagai bobot, Satuan adalah kuadrat dari satuan variabel.
Simpangan baku (σ): Akar kuadrat dari varians, Memiliki satuan yang sama dengan variabel
Distribusi Binomial: Distribusi binomial menggambarkan data diskrit yang dihasilkan dari eksperimen yang dikenal sebagai proses Bernoulli. Pelemparan koin yang adil sejumlah kali adalah proses Bernoulli dan hasil dari lemparan tersebut dapat direpresentasikan oleh distribusi probabilitas binomial. Keberhasilan atau kegagalan orang yang diwawancarai pada tes bakat juga dapat dijelaskan oleh proses Bernoulli. Di sisi lain, distribusi frekuensi masa pakai lampu neon di pabrik akan diukur pada skala jam yang berkesinambungan dan tidak akan memenuhi syarat sebagai distribusi binomial. Fungsi massa probabilitas, rata-rata, dan varians adalah sebagai berikut:

Karakteristik Distribusi Binomial

Hanya ada dua kemungkinan hasil: kepala atau ekor, ya atau tidak, sukses atau gagal. Setiap proses Bernoulli memiliki karakteristik probabilitasnya sendiri. Ambil situasi di mana secara historis tujuh persepuluh dari semua orang yang melamar jenis pekerjaan tertentu lulus ujian pekerjaan. Kita akan mengatakan bahwa karakteristik probabilitas di sini adalah 0,7, tetapi kita dapat menggambarkan hasil pengujian kita sebagai Bernoulli hanya jika kita merasa yakin bahwa proporsi mereka yang lulus ujian (0,07) tetap konstan dari waktu ke waktu. Pada saat yang sama, hasil dari satu pengujian tidak boleh memengaruhi hasil pengujian lainnya.
Distribusi Poisson: Distribusi Poisson digunakan untuk menggambarkan sejumlah proses, termasuk distribusi panggilan telepon yang melalui sistem panel listrik, permintaan pasien untuk layanan di lembaga kesehatan, kedatangan truk dan mobil di gerbang tol, dan jumlah kecelakaan di persimpangan jalan.
Semua contoh ini memiliki elemen yang sama: Semuanya dapat dijelaskan oleh variabel acak diskrit yang bernilai integer (0, 1, 2, 3, 4, dan seterusnya). Jumlah pasien yang tiba di rumah sakit dalam interval waktu tertentu adalah 0, 1, 2, 3, 4, 5, atau bilangan bulat lainnya. Demikian pula, jika Anda menghitung jumlah mobil yang tiba di gerbang tol di jalan raya selama periode 10 menit, jumlahnya akan menjadi 0, 1, 2, 3, 4, 5, dan seterusnya.
Fungsi massa probabilitas, rata-rata, dan varians adalah sebagai berikut:

Karakteristik Distribusi Poisson

Jika kita perhatikan contoh sejumlah mobil, maka jumlah rata-rata kendaraan yang datang per jam sibuk dapat diperkirakan dari data lalu lintas sebelumnya.

Jika kita membagi jam sibuk menjadi interval satu detik, kita akan menemukan pernyataan berikut ini benar.
Probabilitas bahwa tepat satu kendaraan akan tiba di satu bilik per detik adalah angka yang sangat kecil dan konstan untuk setiap interval satu detik.
Probabilitas bahwa dua atau lebih kendaraan akan tiba dalam interval satu detik sangat kecil sehingga kita dapat menetapkannya sebagai nilai nol.
Jumlah kendaraan yang tiba dalam interval satu detik tertentu tidak bergantung pada waktu terjadinya interval satu detik tersebut selama jam sibuk.
Jumlah kedatangan dalam interval satu detik tidak bergantung pada jumlah kedatangan dalam interval satu detik lainnya.

5.6 DISTRIBUSI NORMAL

Apa itu Distribusi Normal?
Bagaimana melakukan perhitungan probabilitas yang terkait dengan distribusi normal?
Apa saja berbagai sifat penting dari distribusi normal?

Dasar-dasar Distribusi Normal:
Grafik pdf (fungsi kerapatan probabilitas) adalah kurva berbentuk lonceng. Variabel acak normal mengambil nilai dari - hingga +. Ia simetris dan berpusat di sekitar rata-rata (yang juga merupakan median dan modus).
Setiap distribusi normal dapat ditentukan hanya dengan dua parameter – rata-rata (μ) dan deviasi standar (σ).
Kita menulisnya sebagai X~N(μ,σ2).
Distribusi normal memiliki aplikasi di banyak bidang administrasi bisnis. Misalnya:
Teori portofolio modern umumnya mengasumsikan bahwa pengembalian portofolio aset yang terdiversifikasi mengikuti distribusi normal. Dalam manajemen operasi, variasi proses sering kali terdistribusi secara normal. Dalam manajemen sumber daya manusia, kinerja karyawan terkadang dianggap terdistribusi secara normal.
Apakah Distribusinya Normal?
Kondisi berikut harus dipenuhi oleh distribusi agar menjadi distribusi normal:
Rata-rata, median, dan modus harus hampir sama. Simpangan baku harus rendah.
Kemiringan dan kurtosis harus mendekati nol.
Median harus terletak tepat di antara kuartil atas dan bawah.
Plot Probabilitas Normal: Plot probabilitas normal adalah teknik grafis untuk pengujian normalitas: menilai apakah kumpulan data terdistribusi secara normal atau tidak. Di sini, pada dasarnya kita membandingkan probabilitas kumulatif yang diamati dengan probabilitas kumulatif teoritis. Jika data yang diamati benar-benar berasal dari distribusi normal, maka kita akan mendapatkan garis lurus.
Untuk distribusi normal, 68,2% data berada dalam rentang satu simpangan baku. (rata-rata - simpangan baku, rata-rata + simpangan baku).

Penyimpangan dari Normalitas: Bagaimana kita dapat mengatakan bahwa distribusi normal adalah perkiraan data yang wajar? Kita dapat melihat data 1. Lebih dari satu modus yang menunjukkan data berasal dari kelompok yang berbeda, 2. Data Tidak memiliki simetri, 3. Nilai ekstrem yang tidak biasa. Jika salah satu dari hal ini diamati, kita dapat mengatakan bahwa datanya tidak normal. Kita dapat mengidentifikasi perbedaan ini dengan melihat 1. Pemeriksaan visual histogram 2. Ringkasan numerik seperti Skewness dan Kurtosis 3. Ringkasan grafis (plot Kuantil Normal).

1. *Ukuran kecenderungan sentral:* Ada tiga cara untuk menemukan nilai sentral: Rata-rata aritmatika, Median, dan Modus.

   Rata-rata atau mean dihitung dengan menemukan jumlah data penelitian dan membaginya dengan jumlah total data. Menentukan denyut jantung merupakan bagian penting dari kondisi medis. Berikut vektor yang berisi jumlah detak jantung.

   ```
   beats <- c(94, 83, 84, 93, 82, 78, 98, 84)
   ```

   Cara cepat untuk menilai data ini adalah dengan mendapatkan rata-rata detak. Ahli statistik menyebutnya "mean". Sebut fungsi mean dengan vektor beats. mean (beats) barplot(beats)

   Jika kita menggambar garis pada plot yang mewakili mean, kita dapat dengan mudah membandingkan berbagai nilai dengan rata-rata. Fungsi abline dapat mengambil parameter h dengan nilai untuk menggambar garis horizontal atau parameter v untuk garis vertikal. Saat dipanggil, fungsi ini memperbarui plot sebelumnya.

   Gambar garis horizontal melintasi plot pada mean:

   ```
   abline(h=mean(beats))
   ```

   Median adalah nilai tengah dalam sekumpulan data. Median dihitung dengan terlebih dahulu mengatur data dalam urutan numerik lalu mencari nilai di tengah daftar. Mari kita ambil contoh nilai yang diperoleh sekelompok siswa. Asumsikan bahwa ujian telah dilakukan untuk nilai 50.

   ```
   marks <- c(14, 13, 14, 23, 42, 24, 47, 18)
   mean(marks)
   ```

   Mari kita lihat bagaimana mean baru ini muncul pada grafik yang sama. barplot(nilai)

   ```
   abline(h = mean(nilai))
   ```

   Mungkin benar secara faktual untuk mengatakan bahwa siswa kita memiliki nilai rata-rata 24,375, tetapi mungkin juga menyesatkan. Untuk situasi seperti ini, mungkin lebih berguna untuk membicarakan nilai "median". Median dihitung dengan mengurutkan nilai dan memilih yang di tengah.

   Panggil fungsi median pada vektor:

   ```
   median(nilai)
   ```

   Mari kita tunjukkan median pada plot. Gambar garis horizontal melintasi plot di median.

   ```
   abline (h=median(nilai))
   ```

   Modus adalah angka yang paling sering muncul dalam kumpulan data.

2. *Ukuran Dispersi:* Kita bahkan ingin mengetahui cara menyebarkan data dari nilai pusat, yaitu mean. Dalam kasus ini, kita ingin melihat ukuran dispersi seperti Rentang, Varians, Simpangan Baku.

   Rentang: Untuk memperoleh rentang, Anda mengurangi angka terkecil dari angka terbesar. Varians: Berasal dari jumlah selisih kuadrat setiap data dari rata-rata aritmatika data.

   Simpangan Standar: Ambil akar kuadrat varians, kita akan mendapatkan Simpangan Standar data. Ahli statistik menggunakan konsep "simpangan standar" dari rata-rata untuk menggambarkan rentang nilai tipikal untuk kumpulan data. Untuk sekelompok angka, konsep ini menunjukkan seberapa besar perbedaannya dari nilai rata-rata. Untuk menghitung simpangan standar, Anda menghitung rata-rata nilai, lalu kurangi rata-rata dari setiap angka dan kuadratkan hasilnya, lalu rata-ratakan kuadrat tersebut, dan ambil akar kuadrat dari rata-rata tersebut.

   Ambil vektor dengan nilai gaji orang yang bekerja di suatu departemen.

   ```
   gaji <- c(46000, 50000, 35000, 30000, 44800, 45000, 10200, 15000)
   barplot(gaji)
   nilaiRata-rata <- rata-rata(gaji)
   ```

   Mari kita lihat plot yang menunjukkan nilai rata-rata:

   ```
   abline(h=nilairata-rata)
   ```

   Untuk menghitung simpangan baku, kita menggunakan fungsi sd. Sekarang mari kita panggil sd pada vektor gaji, dan tetapkan hasilnya ke variabel simpangan baku.

   ```
   deviasi <- sd(gaji)
   ```

   Kita akan menambahkan garis pada plot untuk menunjukkan satu simpangan baku di atas rata-rata

   ```
   abline(h = nilairata-rata + simpangan baku)
   ```

   Sekarang coba tambahkan garis pada plot untuk menunjukkan satu simpangan baku di bawah rata-rata (bagian bawah rentang normal):

   ```
   abline(h = nilairata-rata - simpangan baku)
   ```

3. *Ukuran Lokasi:* Untuk memahami data dengan lebih baik, kita mengamati ukuran lokasi seperti kuartil, desil, dan persentil yang membuat data menjadi 4, 10, dan 100 bagian.
4. *Bentuk distribusi:* Ada dua statistik yang terkait dengan bentuk, Skewness, dan Kurtosis.

   Skewness: Mendeteksi apakah data simetris terhadap nilai pusat distribusi. Jika histogram memiliki ekor kiri yang panjang, kita sebut data tersebut condong negatif, dan jika histogram memiliki ekor kanan yang panjang, kita dapat mengatakan bahwa data tersebut condong positif.

   Kurtosis: Ini adalah ukuran yang dapat mengatakan seberapa datar atau memuncaknya data tersebut. Jika nilai kurtosis positif, kita dapat memahami bahwa data tersebut leptokurtik (memuncak), jika nilainya negatif, data tersebut platikurtik (datar). Nilai kurtosis untuk Distribusi Mesokurtik adalah nol (Normal).

Distribusi normal bersifat asimetris, distribusi probabilitas kontinu yang secara unik ditentukan oleh mean dan deviasi standar. Setiap distribusi normal dapat diubah menjadi distribusi normal standar (skor Z).

5.7 MENDAPATKAN STATISTIK DESKRIPTIF

Untuk menghitung statistik tertentu untuk setiap variabel dalam kumpulan data secara bersamaan, gunakan fungsi sapply jika kumpulan data memiliki nilai yang hilang, lalu tetapkan argumen na.rm ke T:

```
sapply(Kesehatan, rata-rata, na.rm=T)
```

Kita dapat mengamati beberapa peringatan di jendela konsol, ini karena, Jika salah satu variabel dalam kumpulan data Anda non-numerik, fungsi sapply berperilaku tidak konsisten. Di sini kita mencoba menghitung nilai maksimum untuk setiap variabel dalam kumpulan data Kesehatan. R mengembalikan pesan kesalahan karena beberapa variabel dalam kumpulan data adalah variabel faktor. Untuk menghindari masalah ini, kecualikan variabel non-numerik apa pun dari kumpulan data dengan menggunakan fungsi subset braket.

Jika kita ingin mengelompokkan nilai variabel numerik menurut tingkat faktor dan menghitung statistik untuk setiap kelompok, kita dapat menggunakan fungsi tapply atau agregat.

```
tapply(Kesehatan$Usia, Kesehatan$JenisKelamin, rata-rata)
```

Kita juga dapat menggunakan fungsi agregat untuk meringkas variabel menurut kelompok. Menggunakan fungsi agregat memiliki keuntungan karena Anda dapat meringkas beberapa variabel kontinu secara bersamaan.

```
aggregate(Karyawan$Gaji~JenisKelamin, Karyawan, rata-rata)
```

Sekali lagi, Anda juga dapat menggunakan lebih dari satu variabel pengelompokan. Misalnya, untuk menghitung rata-rata gaji untuk setiap kombinasi jenis kelamin dan pendidikan untuk kumpulan data Karyawan:

```
aggregate(Gaji~JenisKelamin+Pendidikan, Karyawan, rata-rata)
```

Untuk meringkas dua atau lebih variabel kontinu secara bersamaan, masukkan variabel tersebut di dalam fungsi cbind.

```
aggregate(cbind(Gaji,Usia)~Level, Karyawan, rata-rata)
```

Dapatkan *Frekuensi tabulasi silang:* Tabulasi silang atau tabel kontingensi adalah jenis tabel yang menampilkan distribusi frekuensi variabel pada baris dan yang lainnya pada kolom. Tabel ini banyak digunakan dalam Analisis Bisnis karena menyediakan hubungan timbal balik antara variabel. Mari kita buat tabel kontingensi dengan menggunakan fungsi tabel pada faktor `Health$Gender` dan `Health$Response`. Anda dapat membuat tabel frekuensi menggunakan fungsi `table()`, tabel proporsi menggunakan fungsi `prop.table()`, dan frekuensi marginal menggunakan `margin.table()`.

```
# buat tabel kontingensi berdasarkan faktor Gender dan Respons
     Health_table <- table(Health$Gender,Health$Response)
     Health_table margin.table(Health_table, 1)

# Frekuensi A (dijumlahkan atas B)
     margin.table(Health_table, 2)

# Frekuensi B (dijumlahkan atas A)
     prop.table(Health_table)

# Persentase sel
     prop.table(Health_table, 1)

# Persentase baris
     prop.table(Health_table, 2)
```

Fungsi Ringkasan: Fungsi `summary()` menyediakan beberapa statistik deskriptif, seperti mean dan median, tentang variabel seperti Tinggi dalam kerangka data Kesehatan. Untuk menghasilkan ringkasan semua variabel dalam kumpulan data, gunakan fungsi ringkasan. Fungsi ini meringkas setiap variabel dengan cara yang sesuai untuk kelasnya. Untuk variabel numerik, fungsi ini memberikan nilai rata-rata, median, rentang, dan rentang interkuartil. Untuk variabel faktor, fungsi ini memberikan angka dalam setiap kategori. Jika suatu variabel memiliki nilai yang hilang, fungsi ini akan memberi tahu Anda berapa banyak nilai yang hilang. `summary(Health)` akan memberikan gambaran umum tentang distribusi setiap kolom.

Fungsi ringkasan menghasilkan semua statistik deskriptif yang terkait dengan variabel tinggi dalam kumpulan data Health. Normalitas suatu distribusi menyiratkan adanya elemen simetri yang terkait dengan distribusi tersebut. Skewness dan Kurtosis dari kumpulan data terjadi di sekitar nol. Analisis dasar menghasilkan hasil bahwa variabel tinggi terdistribusi secara normal dalam kumpulan data Health.

5.8 MEMPEROLEH STATISTIK INFERENSIAL

Statistik Inferensial mengacu pada Penarikan kesimpulan tentang populasi berdasarkan data sampel. *Interval Kepercayaan:* Saat melakukan Analisis Statistik, kita perlu menjawab pertanyaan-pertanyaan berikut:

Bagaimana cara memberikan estimasi interval (interval kepercayaan) untuk parameter populasi seperti mean?

Bagaimana cara menyesuaikan estimasi interval jika simpangan baku populasi tidak diketahui?

Bagaimana cara menghitung interval kepercayaan untuk proporsi populasi?

Berapa ukuran sampel yang harus dikumpulkan untuk mendapatkan lebar estimasi interval yang diinginkan?

Pengujian Hipotesis:

1. Haruskah saya mempekerjakan satu programmer lagi untuk proyek ini?
2. Haruskah kita membuka toko ritel baru di lokasi X?
3. Haruskah kita mempekerjakan perusahaan konsultan ini?
4. Haruskah kita mengakuisisi maskapai penerbangan ini?
5. Haruskah kita berinvestasi dalam iklan daring?
6. Haruskah kita menaikkan suku bunga pinjaman ini?
7. Haruskah kita memasuki pasar ritel India?

Ketika kita memecahkan pertanyaan semacam ini, kita mungkin perlu mencari jawaban untuk beberapa pertanyaan lain seperti:

Bagaimana dan kapan merumuskan hipotesis tentang parameter populasi?
Bagaimana mengukur kekuatan bukti?
Apa itu kesalahan Tipe I dan Tipe II?

Bagaimana menyusun hipotesis: Hipotesis adalah posisi awal yang terbuka untuk pengujian dan penolakan berdasarkan bukti kuat yang merugikan. Keyakinan awal disebut hipotesis nol (H0). Umumnya status quo menyatakan Jangan lakukan apa pun. Negasinya disebut hipotesis alternatif (HA, Ha, H1). Sering kali klaim yang harus diuji, atau perubahan yang harus dideteksi menyatakan Lakukan sesuatu. Kedua hipotesis tersebut saling eksklusif dan menyeluruh.
Proses Pengujian Hipotesis: Dimulai dengan Hipotesis tentang Parameter Populasi. Parameter tersebut dapat berupa rata-rata, proporsi, atau yang lainnya. Kumpulkan informasi dari sampel yang dipilih secara acak dan hitung statistik sampel yang sesuai. Kami Menolak/Tidak Menolak Hipotesis berdasarkan informasi sampel jika sangat tidak konsisten dengan hipotesis nol? Jika ya, maka hipotesis ditolak. Contoh Program Loyalitas Supermarket: Sebuah supermarket berencana meluncurkan program loyalitas jika program tersebut menghasilkan rata-rata pembelanjaan per pembeli lebih dari $120 per minggu. Sampel acak yang terdiri dari 80 pembeli yang terdaftar dalam program percontohan menghabiskan rata-rata $130 dalam seminggu dengan deviasi standar $40. Haruskah program loyalitas diluncurkan?

Proses Pengujian

Mulailah dengan mengasumsikan bahwa H0 (biasanya status quo) benar?: misalnya Saya yakin pengeluaran akan kurang dari atau sama dengan $120.
Kuantifikasi apa yang dimaksud dengan "bukti yang cukup kuat" untuk menolak H0: misalnya Probabilitas menemukan rata-rata sampel harus kurang dari 0,05
Kumpulkan bukti yang akan digunakan untuk menguji H0: misalnya Uji coba menghasilkan pengeluaran rata-rata $130 dalam sampel 80 pelanggan
Hitung probabilitas untuk mengamati bukti yang diberikan atau yang lebih kuat, misalnya Probabilitas maksimum untuk mendapatkan sampel sebesar $130 atau lebih di bawah H0 adalah 0,01
Simpulkan dan ambil tindakan yang tepat? :misalnya Bukti cukup kuat (0,01 < 0,05) untuk menolak H0, lalu luncurkan kartu.

Saat membuat kesimpulan, Anda dapat membuat dua jenis kesalahan:

Keputusan/Realitas	Jangan tolak H0	Tolak H0
H0 benar	Keputusan yang benar	Kesalahan tipe I
H0 salah	Kesalahan tipe II	Keputusan yang benar

Probabilitas melakukan kesalahan Tipe-I sama dengan nilai-p. Nilai-α dapat diartikan sebagai probabilitas yang dapat diterima untuk membuat kesalahan Tipe-I (juga disebut tingkat signifikansi). Hipotesis adalah asumsi tentang parameter populasi yang tunduk pada pengujian dan penolakan berdasarkan bukti. Uji hipotesis berlaku ketika manajer memiliki posisi tertentu pada parameter populasi yang perlu ditolak untuk mengambil tindakan. Seorang ilmuwan data biasanya menargetkan kesalahan tipe-I yang disebut tingkat signifikansi. Jika probabilitas yang dihitung dari sampel yang diberikan kurang dari tingkat signifikansi di bawah hipotesis nol, ia menolak hipotesis nolnya dan membuat perubahan yang diperlukan.

5.9 UJI CHI-KUADRAT

Uji Asosiasi Chi-Kuadrat: Uji asosiasi chi-kuadrat membantu menentukan apakah dua atau lebih variabel kategoris terkait. Uji ini memiliki hipotesis nol bahwa variabel tersebut independen dan hipotesis alternatif bahwa variabel tersebut tidak independen

Uji ini hanya cocok jika ada cukup data, yang umumnya didefinisikan sebagai semua sel tabel yang memiliki jumlah yang diharapkan setidaknya lima. Untuk tabel 2 arah, Anda dapat menggunakan chisq.test (tabel saya) untuk menguji independensi variabel baris dan kolom. Secara default, nilai-p dihitung dari distribusi chi-kuadrat asimptotik dari statistik uji.

```
chisq.test(Health$Treatment,Health$Response)
```

Karena nilai-p kurang dari tingkat signifikansi 0,05, kita dapat menolak hipotesis nol dan menyatakan bahwa keduanya adalah variabel yang terkait.

Uji Eksak Fisher: Uji eksak Fisher digunakan untuk menguji hubungan antara dua variabel kategoris yang masing-masing memiliki dua level. Uji ini dapat digunakan bahkan ketika data yang tersedia sangat sedikit. Uji ini memiliki hipotesis nol bahwa kedua variabel tersebut independen dan hipotesis alternatif bahwa keduanya tidak independen.

```
fisher.test(Health$Treatment,Health$Response)
```

Hasil uji disertai dengan interval kepercayaan 95 persen untuk rasio peluang. Anda dapat mengubah ukuran interval dengan argumen conf.level:

```
fisher.test(Health$Treatment,Health$Response,conf.level=0.99)
```

`fisher.test(x)` menyediakan uji independensi yang tepat. x adalah tabel kontingensi dua dimensi dalam bentuk matriks.

Menganalisis Variabel Kontinu:

Saat menganalisis variabel kontinu, kita mungkin perlu menjawab beberapa pertanyaan.

Bagaimana cara membandingkan rerata dua populasi menggunakan observasi berpasangan?
Kapan dan bagaimana cara membandingkan rerata dua populasi menggunakan sampel independen?
Bagaimana cara menguji perbedaan dalam dua proporsi populasi?

Contoh program penurunan berat badan

Seorang ahli gizi ingin menilai efek program diet terorganisir terhadap berat badan peserta. Dia secara acak memilih 60 peserta program diet dan mengukur berat badan mereka (dalam kg) sebelum mendaftar dalam program dan segera setelah program selesai. Berdasarkan bukti ini, apakah program diet baru efektif dalam mengurangi berat badan? Sebuah jaringan kesehatan dapat merekomendasikan diet rendah kalori konvensional secara gratis atau dapat merekomendasikan diet baru dengan membayar biaya lisensi. Perusahaan telah menentukan bahwa membayar biaya lisensi layak dilakukan jika mereka dapat memperoleh cukup banyak anggota tambahan, yang mungkin terjadi jika diet baru mengurangi berat badan rata-rata hingga 3 kg atau lebih dibandingkan dengan diet rendah kalori konvensional. Perusahaan mengumpulkan data penurunan berat badan dari dua sampel acak sederhana orang, salah satunya menjalani diet baru dan yang lainnya menjalani diet konvensional selama 6 bulan.

5.10 UJI-T

Uji T satu sampel digunakan untuk membandingkan nilai rata-rata sampel dengan nilai konstan yang dilambangkan m0. Uji ini memiliki hipotesis nol bahwa rata-rata populasi sama dengan m0, dan hipotesis alternatif bahwa rata-rata populasi tidak sama.

Uji t satu sampel

```
setwd("D:/R data")
WR_Trt <- read.csv("Wt_red.csv") fix(WR_Trt)
OS_ttest <- WR_Trt[which(WR_Trt$Treatment=="Dummy Pill"),]
OS_ttest$Change <- OS_ttest$Before-OS_ttest$After
fix(OS_ttest)
OS_tt_res<- t.test(OS_ttest$Change, mu=3)
```

Argumen mu memberikan nilai yang ingin Anda gunakan untuk membandingkan rata-rata sampel. Argumen ini bersifat opsional dan memiliki nilai default 0. Secara default, R melakukan uji dua sisi. Untuk melakukan uji satu sisi, tetapkan argumen alternatif ke "lebih besar" atau "lebih kecil". Untuk menyesuaikan ukuran interval, gunakan argumen `conf.level`:

```
t.test(OS_ttest$Change, mu=1, alternative="lebih besar")
t.test(OS_ttest$Change, mu=1, conf.level=0.99)
```

Uji-t dua sampel digunakan untuk membandingkan nilai rata-rata dari dua sampel independen, untuk menentukan apakah keduanya diambil dari populasi dengan rata-rata

yang sama. Uji-t ini memiliki hipotesis nol bahwa kedua rata-rata tersebut sama, dan hipotesis alternatif bahwa keduanya tidak sama.

Untuk melakukan uji-t dua sampel dengan data dalam bentuk bertumpuk, gunakan perintah: `t.test(values~groups, dataset)`, di mana values adalah nama variabel yang berisi nilai data dan groups adalah variabel yang berisi nama sampel. Jika variabel pengelompokan memiliki lebih dari dua level, maka Anda harus menentukan dua kelompok yang ingin Anda bandingkan.

```
t.test(WR_Trt$Change~WR_Trt$Treatment, WR_Trt, Treatment %in%
c("Old_Trt", "Test_Drug"))
```

Secara default, R menggunakan estimasi varians terpisah saat melakukan uji t dua sampel dan uji t berpasangan. Jika Anda yakin varians untuk kedua kelompok tersebut sama, Anda dapat menggunakan estimasi varians gabungan. Untuk menggunakan estimasi varians gabungan, tetapkan argumen var.equal ke T.

Uji T berpasangan: Uji t berpasangan digunakan untuk membandingkan nilai rata-rata untuk dua sampel, di mana setiap nilai dalam satu sampel sesuai dengan nilai tertentu dalam sampel lainnya. Ia memiliki hipotesis nol bahwa kedua rata-rata tersebut sama, dan hipotesis alternatif bahwa keduanya tidak sama.

```
# paired t-test
t.test(WR_Trt$Before,WR_Trt$After,paired=T)
```

Pengukuran sebelum dan sesudah pada subjek yang sama merupakan hal yang wajar dan memungkinkan, dalam hal ini, kami menggunakan uji berpasangan.

Distribusi Sampling Rata-rata Dua Sampel: Dua distribusi sampling rata-rata adalah normal asalkan kondisi Teorema Batas Pusat terpenuhi secara terpisah untuk 1. Independensi seperti Siapa yang ada dalam sampel tidak memengaruhi siapa lagi yang ada dalam sampel itu dan Siapa yang ada dalam sampel tidak memengaruhi siapa yang ada dalam sampel lainnya. 2. Kondisi ukuran seperti Jumlah observasi dalam setiap sampel harus melebihi 10 kali nilai absolut Kurtosis dan 10 kali Skewness kuadrat dalam sampel itu.

Contoh: Proporsi pelaku diet yang menurunkan berat badan: Misalkan metrik alternatif untuk mengukur kinerja program diet adalah proporsi peserta yang telah kehilangan lebih dari 3 KG. Cara terbaik untuk membandingkan rata-rata dua distribusi adalah dengan menggunakan observasi berpasangan jika memungkinkan. Selisih rata-rata observasi sampel berpasangan mengikuti distribusi normal menurut Teorema Batas Pusat. Ketika observasi berpasangan tidak memungkinkan, kami menggunakan sampel independen dan merumuskan hipotesis tentang perbedaan antara dua rata-rata. Penting untuk memastikan bahwa subjek secara acak ditugaskan ke dua sampel untuk menghindari kesalahan perancu. Pendekatan serupa dapat digunakan untuk menguji perbedaan proporsi antara dua.

5.11 ANALISIS VARIANS (ANOVA)

Analisis varians memungkinkan Anda membandingkan rata-rata tiga atau lebih sampel independen. Pendekatan ini cocok jika nilai diambil dari distribusi normal dan jika variansnya

kira-kira sama di setiap kelompok. Hipotesis nol untuk pengujian ini adalah bahwa rata-rata untuk semua kelompok adalah sama, dan hipotesis alternatifnya adalah bahwa rata-rata berbeda untuk setidaknya satu pasang kelompok.
Mari kita pikirkan pertanyaan-pertanyaan berikut dan coba jawab dengan studi kasus.

Mengapa analisis varians (ANOVA) diperlukan untuk membandingkan rata-rata populasi?
Apa prinsip jumlah kuadrat?
Bagaimana cara melakukan uji ANOVA?
Analisis tindak lanjut apa yang harus dilakukan jika uji ANOVA signifikan?

Studi Kasus: Program penurunan berat badan: Misalkan ahli gizi ingin melakukan evaluasi komparatif terhadap tiga program diet. Ia secara acak menugaskan sejumlah peserta yang sama untuk masing-masing program ini dari kumpulan relawan yang sama. Misalkan rata-rata penurunan berat badan di setiap kelompok (lengan) percobaan adalah 4 kg, 7 kg, 5,4 kg. Apa yang dapat disimpulkannya? Di sini, Dua jenis variasi penting. Tidak setiap individu dalam setiap program akan merespons program diet secara identik. Lebih mudah untuk mengidentifikasi variasi di seluruh program jika variasi dalam program lebih kecil, Oleh karena itu metode ini disebut Analisis Varians (ANOVA). Memformalkan intuisi di balik variasi. Yang lebih mengejutkan dan berguna adalah: Jumlah Kuadrat Total (SST), Jumlah Kuadrat Perlakuan (SSTR), Jumlah Kuadrat Kesalahan (SSE)
Uji statistik untuk kesetaraan mean:
n subjek dibagi rata ke dalam kelompok r
Hipotesis:H0: $\mu 1 = \mu 2 = \mu 3 = \ldots = \mu r$. Tolak hipotesis nol jika nilai-p < α.

Anda dapat melakukan analisis varians dengan fungsi aov. Perintah tersebut berbentuk:

```
aov(Change~Treatment, WR_Trt)
```

Hasil analisis terdiri dari banyak komponen yang tidak ditampilkan secara otomatis oleh R. Jika Anda menyimpan hasil ke objek seperti yang ditunjukkan di sini, Anda dapat menggunakan fungsi lebih lanjut untuk mengekstrak berbagai elemen output:

```
aovobject<-aov(Change~Treatment, WR_Trt)
```

#Setelah Anda menyimpan hasil sebagai objek, Anda dapat melihat tabel ANOVA dengan fungsi anova:

```
anova(aovobject)
```

#Untuk melihat koefisien model, gunakan fungsi coef:

```
coef(aovobject)
```

#Untuk melihat interval kepercayaan untuk koefisien, gunakan fungsi confint:

```
confint(aovobject)
```

Anova Satu Arah (Desain Acak Lengkap)

```
fit <- aov(Change~Treatment, WR_Trt)
```

Tabel ANOVA: Jika kita menolak hipotesis nol bahwa semua rata-rata sama, kemungkinan Anda membuat kesalahan kurang dari 2,5%. Dapatkah kita simpulkan bahwa Uji Obat-diet lebih efektif daripada diet Lama? Tingkat variasi antara dan dalam kelompok menentukan kekuatan bukti terhadap hipotesis nol bahwa rata-rata semua kelompok adalah sama. Jumlah total deviasi kuadrat (di sekitar rata-rata besar) sama dengan jumlah deviasi kuadrat galat (di sekitar rata-rata kelompok masing-masing) ditambah jumlah deviasi kuadrat perlakuan (rata-rata kelompok di sekitar rata-rata besar). Uji ANOVA membandingkan rata-rata kuadrat perlakuan dengan rata-rata kuadrat galat. Jika rasio ini "secara signifikan" lebih besar, kita dapat menolak hipotesis nol bahwa rata-ratanya sama.

BAB 6
PENGANTAR PEMBELAJARAN MESIN

6.1 PENDAHULUAN

Pernahkah Anda berpikir bahwa Mesin dapat belajar dan mengatur pekerjaan Anda secara lebih konsisten daripada Anda?
Pernahkah Anda memikirkan pertanyaan-pertanyaan ini?
Tugas apa yang dapat dilakukan mesin dengan baik tetapi tidak dapat dilakukan manusia atau sebaliknya?
Apa artinya Pembelajaran Mesin?
Bagaimana pembelajaran terkait dengan kecerdasan?
Dapatkah manusia benar-benar menciptakan Mesin Cerdas yang dapat mengungguli Manusia dalam Banyak hal?
Apa artinya menjadi cerdas?
Apakah Anda percaya bahwa suatu mesin akan pernah dibangun yang mengungkap kecerdasan?
Apa artinya menjadi sadar?
Dapatkah seseorang menjadi cerdas dan tidak sadar atau sebaliknya?

Ketika kita melihat banyak data, kita tidak yakin apa yang harus dicari dan apa yang ada di dalamnya, dan apa saja yang akan ditemukan. Ingatlah ini, pembelajaran bukan hanya sikap terhadap kehidupan, tetapi juga sikap terhadap penambangan data dan pembelajaran mesin, kita selalu mendekati data dan pembelajaran mesin dengan sikap ini yang akan membawa Anda sangat jauh.

Mari kita bahas filosofi pembelajaran, kita belajar dengan banyak cara. Kita belajar dengan cara mengasimilasi. Kita membaca banyak buku, kita menonton banyak video, kita mendengarkan lagu, ini adalah asimilasi. Hal-hal yang kita pelajari perlu diterapkan, jika tidak, kita akan lupa. Kita menerapkannya dengan melakukan dan berdiskusi. Buku ini akan berisi skenario teoritis dan praktis. Kita akan mencoba menerapkan beberapa hal di sini ke kumpulan data aktual. Anda dapat menggunakan bahasa apa pun yang Anda inginkan, apa pun alat favorit Anda (Saya menggunakan R sebagai Alat). Kita menerapkan hal-hal yang telah kita pelajari. Setelah selesai menerapkan, kita dapat mengadaptasi apa pun yang telah Anda pelajari dan menciptakan sesuatu yang baru. Setelah menyelesaikan buku ini, saya ingin semua orang mencoba hal-hal ini dalam memecahkan masalah bisnis. Mari kita lihat beberapa pernyataan. Pengetahuan adalah apa yang tersisa setelah fakta-fakta dilupakan. Buku ini bukan tentang mempelajari rumus-rumus tertentu, tetapi buku ini benar-benar tentang konsep-konsep. Dalam buku ini, saya akan membahas topik-topik dari berbagai domain, berbagai masalah yang dipecahkan dengan menggunakan pembelajaran mesin. Buku ini akan membantu Anda memahami berbagai algoritma pembelajaran mesin yang kita gunakan dalam industri untuk memecahkan masalah bisnis. Ada tiga I yang akan membuat produk

hebat, mari kita lihat radio dan TV, ada produk hebat pada masanya, tetapi sekarang mari kita lihat kualitas apa yang membuat sebuah produk hebat.

I (Interface) pertama adalah Antarmuka produk. Apakah saya perlu membaca manual untuk mengoperasikan produk atau seorang pria berusia 5 tahun atau bahkan 70 tahun dapat mengoperasikan produk saya? Kotak pencarian Google adalah contoh antarmuka yang hebat.

I (infrastructure) Berikutnya adalah Infrastruktur, kita membangun produk bukan untuk PC, tetapi untuk planet ini. Terjadi pergeseran paradigma, sebelumnya orang membangun produk seperti Windows O/S, Outlook, semuanya ditujukan untuk PC. Jika Anda melihat LinkedIn, YouTube, Google, Facebook, ini adalah produk yang dibangun untuk dunia, yang ditujukan untuk digunakan oleh miliaran orang di seluruh dunia.

I (Intelligent) Ketiga untuk membuat produk hebat adalah Kecerdasan. Jika Anda melihat pencarian web di Google dan mengetikkan kueri, ada beberapa saran otomatis. Saat Anda melihatnya, Anda merasa Google membaca pikiran Anda, video YouTube saat Anda menonton dan saat menyarankan video terkait, Anda merasa produk itu cerdas. LinkedIn, Amazon, Netflix saat Anda menggunakannya dan melihat rekomendasinya, kami merasa mereka sangat cerdas. Fitur ini dikenal sebagai Kecerdasan Buatan yang tanpanya produk-produk tersebut mungkin tidak akan berhasil. Jadi, setiap kali Anda berpikir untuk membangun produk baru, pikirkan dengan cara ini, bahwa produk itu harus memiliki ketiga I di dalamnya. Buku saya membahas bagian Kecerdasan dan kami selalu membahas cara membuat produk Cerdas dengan menggunakan Pembelajaran Mesin.

6.2 MENGAPA PEMBELAJARAN MESIN SEKARANG?

Pembelajaran mesin, Kecerdasan Buatan, Penambangan Data, Analisis data besar semuanya tampak serupa dan hampir sama. Mungkin ada sedikit perbedaan dalam pendekatan dan tumpang tindih di antara semuanya, tetapi yang perlu Anda pahami adalah semuanya sama. Pembelajaran mesin adalah istilah tradisional yang digunakan dan kami menggunakan istilah yang sama.

Izinkan saya memberikan perspektif tentang Pembelajaran mesin: Izinkan saya mengambil contoh pemeringkatan halaman web. Itu adalah proses mengirimkan kueri ke mesin pencari, yang kemudian menemukan halaman web yang relevan dengan kueri dan mengembalikannya dalam urutan relevansinya. Untuk mencapai tujuan ini, mesin pencari perlu 'mengetahui' halaman mana yang relevan dan halaman mana yang cocok dengan kueri. Pengetahuan tersebut dapat diperoleh dari struktur tautan halaman web, kontennya, frekuensi pengguna mengikuti tautan yang disarankan dalam kueri. Pemfilteran kolaboratif adalah aplikasi lain dari pembelajaran mesin, yang digunakan oleh toko e-commerce seperti Amazon secara ekstensif untuk menarik pengguna agar membeli barang tambahan. Mari kita lihat penyaringan spam, kita tertarik pada jawaban ya/tidak mengenai apakah email berisi informasi yang relevan atau tidak. Hal ini sangat bergantung pada pengguna: bagi seorang pelancong yang sering bepergian, email dari maskapai penerbangan yang memberi tahu dia tentang diskon terbaru mungkin terbukti sebagai informasi yang berharga, sedangkan bagi banyak penerima lainnya, hal ini mungkin terbukti lebih merepotkan. Untuk mengatasi

masalah ini, kami ingin membangun sistem yang mampu mempelajari cara mengklasifikasikan email baru. Mari kita lihat diagnosis kanker, ia memiliki struktur umum yang berdasarkan data histologis jaringan pasien, kita dapat menyimpulkan apakah pasien sehat atau tidak. Di sini, kita diminta untuk membuat jawaban ya/tidak berdasarkan serangkaian pengamatan.

Kita semua bekerja untuk perusahaan yang berbeda, kita melihat sejumlah data tertentu, jika Anda mundur dan melihat apa yang dilakukan dunia, sungguh luar biasa bahwa mereka mengumpulkan banyak data, jika Anda melihat urutan gen, proyek genom manusia, orang-orang mengumpulkan urutan gen setiap organisme, itu adalah urutan sepanjang satu miliar yang perlu Anda analisis sekarang Anda dapat membayangkan berapa banyak data itu. Setiap kali Anda menggesek kartu kredit atau debit, Anda menciptakan banyak sekali data. Setiap kali Anda membeli atau menjual saham, titik data dihasilkan. Setiap kali Anda menulis buku atau dokumen hukum, atau setiap kali Anda mengirim satelit, satelit ini mengumpulkan semua jenis data.

Mari saya jelaskan sedikit tentang Big Data, hampir 200 juta tweet terjadi setiap hari dan ada sekitar 500 juta akun Twitter. Pengguna YouTube mengunggah 100 jam video setiap menit, di internet 800 situs web baru dibuat setiap menit, Facebook memproses 100 terabyte data setiap hari, ada 30 miliar konten yang dibagikan setiap bulan yang menjadi 30+ petabyte data pengguna. Google memproses 20 petabyte data sehari, ini tentang merayapi web, mengindeks web, dll. Wal-Mart lebih dari 1 juta transaksi pelanggan setiap jam, ini adalah dunia yang benar-benar baru tempat kita tinggal Lembar kerja Excel tidak akan cukup, Anda tidak dapat memuat data ke PC Anda, memproses data dengan PC Anda, dan membuat wawasan tentang data itu, tidak mungkin. Mari kita pikirkan bagaimana kita sampai di sini? Kita tidak sampai di sini secara kebetulan. Banyak hal yang harus terjadi untuk sampai di sini. Ledakan data ini dimungkinkan oleh sensor yang lebih baik. Ketika saya mengatakan sensor, apa pun yang mengumpulkan titik data adalah sensor. Termometer adalah sensor. GPRS di mobil adalah sensor. Bahkan kita adalah sensor. Setiap kali kita menghasilkan titik data. Ketika Anda menelepon dari ponsel, Anda menciptakan data. Setiap kali Anda menggesek kartu, Anda menghasilkan data. Saya ingin Anda membayangkan dunia baru tempat kita tinggal.

Setelah mengumpulkan data, ketika kita memperoleh beberapa statistik, kita masih dapat menghasilkan banyak statistik, laporan, dan histogram. Namun, itu bukanlah kecerdasan. Itu adalah kalkulasi dengan kekuatan kasar. Kita perlu menyadari semua hal yang kita lakukan. Pakar pembelajaran mesin berpikir tentang hal-hal futuristik. Mereka mengatakan pernyataan seperti teknologi kita. Mesin kita adalah bagian dari kemanusiaan kita. Kita menciptakannya untuk memperluas diri kita. Itulah keunikan manusia. Pernyataan ini berlaku sepanjang masa, jika Anda melihat manusia purba, mereka menciptakan peralatan untuk berburu, mereka menciptakan api, mereka menciptakan roda, kita ingin melihat yang sangat jauh sehingga menciptakan teleskop, kita ingin melihat yang sangat kecil sehingga menciptakan mikroskop, apa pun yang kita ciptakan seperti pesawat terbang, derek, adalah untuk memperluas diri kita. Satu hal yang belum kita kembangkan, hal yang selalu ingin kita ciptakan dan masih belum berhasil adalah otak kita. Ketika saya melihat suatu masalah, bagaimana cara menyelesaikannya, ketika kita melihat sebuah Gambar, bagaimana kita

mengenalinya? Sekarang, tujuan kita adalah menciptakan sebuah mesin yang cerdas, bukan hanya kalkulator, bukan hanya menyimpan banyak data, bukan hanya program yang rumit, tetapi bagaimana cara membuat mesin Cerdas secerdas otak manusia.

Mari kita lihat evolusi komputer, Bayangkan komputer pertama telah diciptakan, Anda perlu masuk ke dalam komputer, menghubungkan kabel, itu adalah pemrograman. Kemudian muncullah PC IBM, lalu laptop, dari satu komputer untuk universitas menjadi satu komputer di setiap rumah, berkat IBM dan Microsoft, memiliki PC di rumah adalah visi mereka.

Apa evolusi komputasi selanjutnya? Komputasi awan adalah evolusi selanjutnya. Ini adalah pusat data, motherboard terhubung satu demi satu di rak, kita tidak memerlukan monitor dan keyboard untuk mengendalikannya. Semua data yang dihasilkan oleh Amazon, YouTube, LinkedIn, Twitter, Facebook, dll. berada di pusat data. Apa masa depan? Masa depan adalah apa yang kita sebut sebagai komputer kuantum, melampaui jenis komputasi tradisional, yang menggunakan status kuantum atom, untuk komputasi dan menyimpan informasi. Seluruh pusat data dapat masuk ke dalam komputer kuantum ini, Anda dapat membayangkan masa depan komputasi. Bandingkan komputer pertama dengan komputer kuantum, keduanya terlihat kikuk, tetapi yang terakhir satu triliun kali lebih cepat daripada komputer pertama. Dalam beberapa hari ke depan kita dapat memiliki miniatur komputer kuantum dan pusat data komputer kuantum, pikirkan apa yang akan terjadi. Anda perlu mengingat ini seperti daya komputasi tumbuh pada tingkat eksponensial yang sangat besar, dan kita dapat menangani apa pun datanya dan berapa pun ukurannya.

Mari kita lihat evolusi teknologi informasi. Saya membaginya menjadi tiga bagian, yang pertama adalah Era Pengindeksan, di mana yang harus kita lakukan hanyalah mencari cara untuk mengumpulkan data, menyimpannya dalam basis data sedemikian rupa sehingga kita dapat mengambilnya dengan kueri SQL. Itulah Era Pengindeksan, di sinilah bahkan mesin pencari modern termasuk dalam kategori ini, Bahkan pencarian Google adalah sistem pengindeksan yang sangat besar, kuncinya adalah kata dan nilainya adalah dokumen yang berisi semua kata, tetapi pada dasarnya itu tidak lebih dari sekadar sistem SQL. Orang-orang menanyakan pertanyaan dasar seperti dapatkah saya mengambil rata-rata semua orang yang bekerja di suatu departemen, atau pada teknologi tertentu dengan tahun pengalaman tertentu. Di situlah sistem pengindeksan berakhir. Era berikutnya adalah era interpretasi, di era ini, kita berpikir seperti apakah saya dapat menafsirkan data, dengan cara yang sangat menarik, apakah ada pola tersembunyi dalam data, apakah data mengatakan sesuatu yang tidak saya ketahui, inilah yang akan kita pelajari dalam pembelajaran tanpa pengawasan. Ini tentang melampaui kueri, dan membiarkan data berbicara tentang apa yang dimilikinya karena saya akan mengajukan pertanyaan lebih dari sekadar kueri sederhana. Kita melihat banyak contoh tentang hal ini di seluruh buku ini.

Era berikutnya adalah era kecerdasan, di sinilah kita mulai mengambil keputusan. Dapatkah saya menggunakan data saya, tidak hanya sebagai basis data, dapatkah saya juga membuat prediksi tentang masa depan?, Apakah pelanggan ini akan berhenti berlangganan?, Siapa orang yang mungkin Anda rekomendasikan berikutnya di LinkedIn?, Apa produk baru yang harus saya buat untuk pasar?, Bagaimana saya dapat membuat keputusan berdasarkan

pengalaman masa lalu saya, dalam semua data yang saya miliki?. Di era kecerdasan, pembelajaran terbimbing dan pengoptimalan akan menurun. Ada dua tujuan untuk pembelajaran mesin. Sekarang kita memahami bahwa kita berada di dunia yang kaya data, dan kita perlu melakukan sesuatu untuk menangani data ini. Pembelajaran mesin adalah salah satu cara untuk melakukannya.

6.3 BAGAIMANA PEMBELAJARAN MESIN BEKERJA?

Untuk memahami hal ini, kita perlu memikirkan tentang Apa hakikat pikiran? Apakah kita ingin memahami cara kerja pikiran? Apa itu kecerdasan? Apa itu pembelajaran? Apa itu berpikir? Misalnya, saya dapat membangun sistem OCR untuk membaca karakter teks dari sebuah buku, tetapi apakah sistem tersebut memahami teks tersebut. Bisakah kita beralih dari membaca teks ke memahami teks?, Bisakah kita beralih dari mendengarkan ke mendengar dan memahami audio?, Bisakah ponsel, kamera video yang Anda gunakan untuk mengambil gambar atau video dapat memahaminya dan menafsirkannya seperti yang Anda inginkan? Bagaimana kita dapat memperluas mesin ke tingkat ini? Itulah salah satu tujuan Kecerdasan Buatan. Pada tahun 1950, Alan Turing menemukan tes ini, bagaimana kita bisa mengatakan bahwa mesin itu cerdas? Ia menemukan tes menarik yang disebut tes Turing. Pada dasarnya, tes ini dilakukan jika seseorang mengobrol dengan dua kotak obrolan. Salah satu kotak obrolan terhubung ke komputer dan yang lainnya dengan orang sungguhan. Tujuan orang A adalah menentukan mana yang mesin dan mana yang mesin. Jika orang A tidak dapat mengatakan bahwa ia mengobrol dengan mesin atau orang, maka kita dapat mengatakan bahwa kita telah mencapai tingkat kecerdasan buatan. Saat ini, kita berbicara tentang pencarian Google. Kita sangat berhati-hati dalam memilih kata kunci, menjaga urutannya, karena Google hanya dapat memahami kueri berdasarkan kata kunci. Google tidak dapat memahami pertanyaan dan melupakan percakapan. Kita ingin berevolusi dari kata kunci menjadi pertanyaan jawaban dan bahkan percakapan. Perusahaan seperti Google berupaya memecahkan masalah ini dan menciptakan mesin yang sebaik otak manusia.

Tujuan kedua dari pembelajaran mesin adalah mengambil keputusan dari data. Saya akan memberikan beberapa contoh.

Di posisi mana Iklan/Halaman akan ditampilkan untuk kueri tertentu?
Haruskah saya menyetujui pinjaman rumah ini atau tidak?
Video mana yang akan ditayangkan berikutnya di YouTube?

Semua pertanyaan ini perlu dijawab oleh perusahaan, ini bukan seperti satu keputusan besar, ini semua adalah keputusan mikro, yang perlu diambil beberapa kali. Tujuannya adalah untuk mengambil keputusan dengan cara yang lebih sistematis dan lebih terorganisasi.

Anda harus melihat perusahaan Anda dan melihat keputusan seperti apa yang diambil perusahaan Anda, apakah berdasarkan firasat, apakah berdasarkan studi kasus sebelumnya, atau benar-benar didorong oleh data?. Kita perlu memeriksa apakah kita memiliki cukup data untuk membuat keputusan, bagaimana kita mengukur dan memvalidasi keputusan, inilah tujuan buku ini untuk menjadikan Anda ilmuwan data yang lebih baik, yang dapat

mengumpulkan data, mengubahnya menjadi masalah penelitian, mengembangkan model, menerapkan model, dll., Ilmu data adalah kombinasi dari tiga hal: 1. Data 2. Algoritma ML 3. Pengetahuan Domain. Kita semua mungkin kuat dalam satu atau dua hal dan kita perlu memilih yang lain untuk menjadi ilmuwan data. Bayangkan jika Anda lulusan ilmu komputer, Anda akan senang berurusan dengan Algoritma, menerapkan algoritma pada kumpulan data yang diberikan kepada Anda oleh klien, tetapi ketika Anda datang ke dunia nyata, berurusan dengan data merupakan masalah besar. Mari kita coba memahami kata Data Mining, Mining adalah proses mencari berlian di antara tumpukan batu yang sangat besar. Faktanya, data tersebut acak, dalam banyak hal Anda bahkan tidak dapat membayangkannya. Kita perlu memahami di sini bahwa data tidak sesederhana yang kita asumsikan. Ini seperti anak Anda yang berusia 2 tahun. Anda menyukainya, tetapi dia melakukan banyak hal yang tidak Anda sukai. Dia tidak berperilaku seperti yang Anda inginkan. Mari kita asumsikan bahwa kita kuat dalam algoritma data dan pembelajaran mesin, kita tetap perlu tahu cara menerapkan semua ini ke situasi dunia nyata karena data dalam domain yang berbeda berperilaku berbeda. Pengetahuan domain memainkan peran yang sangat penting dalam Ilmu Data. Hanya ketika Anda kuat dalam domain, Anda akan memahami parameter mana yang harus dicari?, Fitur seperti apa yang harus dicari?. Jadi selama bertahun-tahun pengalaman, saya memahami bahwa ketiga hal ini memainkan peran penting dalam Ilmu Data. Orang-orang yang kuat dalam ketiga hal ini akan disebut sebagai ilmuwan data. Semoga setelah membaca buku ini Anda akan menjadi kuat dalam ketiga hal tersebut.

6.4 JENIS PEMBELAJARAN MESIN

Anda telah diberi banyak data, tetapi tidak diberi tahu apa yang harus dicari? Sekarang Anda bisa lebih kreatif. Bayangkan saja Anda menyediakan kertas, spidol, dan semua barang lainnya untuk anak Anda dan memintanya menggambar sesuatu. Dia bisa sangat kreatif dan menggambar apa pun yang dia suka dan menjelaskan kepada Anda gambar apa itu. Ini adalah Pembelajaran Tanpa Pengawasan. Dalam hal ini, kami mencoba menemukan struktur dan tata bahasa dalam data sehingga kami dapat memahami data tersebut. Dalam pembelajaran tanpa pengawasan, yang kami kerjakan adalah data yang tidak berlabel.

Bayangkan saya memberi Anda semua video YouTube, video mana yang ditonton berapa kali, dan siapa yang menonton video mana tetapi tidak memberi tahu apa yang harus dilakukan dengan video itu, maka itu menjadi masalah tanpa pengawasan.

Bayangkan anak Anda kembali kepada Anda dan bertanya apa yang harus digambar? Jika Anda memberikan masukan berikutnya seperti menggambar Pohon atau menggambar Rumah, maka itu menjadi pembelajaran dengan pengawasan. Itu berarti Pembelajaran dengan pengawasan adalah ketika Anda tahu apa yang harus dicari. Dalam pembelajaran terbimbing, kami telah memberi label pada data dan kami mencoba menemukan strukturnya serta variabel mana yang menyebabkan variabel lainnya (Struktur dan Kausalitas) dengan membangun Model.

Saat Anda membangun model, yang mengatakan prediksi Penipuan, Prediksi Perpindahan Pelanggan, prediksi video YouTube berikutnya yang akan ditonton pelanggan,

prediksi rekomendasi LinkedIn, Prediksi Kanker, saat Anda memberikan masalah yang sangat spesifik maka itu menjadi pembelajaran Terbimbing.

Mari kita pikirkan tentang pembelajaran Tanpa Pengawasan, Bayangkan Anda memulai bisnis baru, Anda telah melalui berbagai studi kasus dan menemukan 20 masalah CRM (Manajemen Hubungan Pelanggan), Sekarang kita dapat menangani 20 Masalah CRM tersebut dan mungkin merasa senang karena Anda menemukan jawaban untuk semua masalah tersebut. Namun, dapatkah Anda merasa nyaman dengan semua itu? Apakah kita dijamin tidak akan ada masalah lain? Apa masalah dengan pendekatan ini? Mari kita bayangkan semua studi kasus berada di negara-negara barat, dan Anda ingin menerapkannya di India. Masalah di India akan sangat rumit dan bisa sangat berbeda.

Bayangkan saja, Mengapa Amazon tidak dapat bekerja di India sebagaimana adanya? karena di India, Fabmart (banyak orang percaya Flipkart) menciptakan konsep baru yang disebut bayar di tempat (COD), tanpa opsi itu, Amazon tidak akan sukses di India. Jadi, kami harus rendah hati dengan pengetahuan domain Anda. Kami perlu memeriksa apakah data mengatakan sesuatu kepada Anda, di situlah kami membuka diri dan berkata, izinkan saya melakukan pembelajaran tanpa pengawasan. Izinkan saya melihat apa yang dikatakan data di luar apa yang saya pikirkan.

Pembelajaran semi-supervised: Dalam jenis pembelajaran ini, Anda diberikan data berlabel dan tidak berlabel, jumlah data berlabel lebih sedikit dibandingkan dengan jumlah total data. Pembelajaran semi-supervised adalah tentang bagaimana Anda memanfaatkan pembelajaran tanpa pengawasan dalam kombinasi dengan data berlabel untuk membangun model yang lebih baik.

Pembelajaran Aktif: Katakanlah kita memiliki 2000 contoh data berlabel, proses pelabelan sangat mahal, saya harus memilih orang berikutnya dari 10 juta contoh yang masih belum berlabel, perlu memilih 20 gambar berikutnya karena saya hanya memiliki kendala seperti pendanaan dan waktu, contoh mana yang harus saya beri label terlebih dahulu? Karena itu akan menentukan model berikutnya yang akan Anda bangun. Ini disebut pembelajaran aktif.

Pembelajaran penguatan: Dalam jenis pembelajaran ini, kita tidak membangun model yang memetakan A ke B, tetapi kita membangun model yang memetakan seluruh rangkaian tindakan menjadi suatu hasil. Seperti bermain catur, setiap posisi koin akan menentukan apakah Anda akan menang atau kalah. Anda tidak dapat mengevaluasi satu langkah saja, tetapi Anda dapat mengevaluasi seluruh permainan sebagai satu kesatuan.

Bayangkan skenario bank yang mengevaluasi nasabahnya berdasarkan urutan tindakan dari pembukaan rekening hingga penutupan rekening. Serangkaian tindakan tersebut akan membuat nasabah menjadi loyal atau berhenti berlangganan. Akhirnya, saat nasabah berhenti berlangganan, Anda akan menyadari bahwa satu atau beberapa tindakan dalam urutan tersebut salah. Itu berarti Anda mengerjakan serangkaian keputusan, bukan satu keputusan tunggal.

Mari saya ambil contoh kata kunci Google Adwords, Anda masuk, membuka bisnis, membuka akun Adwords, dan menentukan kata kunci yang ingin kami gunakan untuk bisnis

saya. Sekarang banyak penipuan terjadi. Apa yang terjadi? Orang menggunakan kata kunci seperti Mahatma Gandhi untuk toko mainan. Mengapa? Nasabah menerapkan trik ini? Anda tahu itu kueri yang bagus. Saat kueri yang bagus muncul, iklan saya akan muncul, tetapi tidak relevan.

Sekarang ada algoritme pembelajaran mesin yang bagus yang menentukan apakah kata kunci relevan dengan bisnis atau tidak. Namun, terkadang keputusannya sangat tidak jelas. Algoritme pembelajaran mesin tidak dapat memberi Anda solusi. Anda harus mengambil keputusan akhir karena model pembelajaran mesin tidak dapat memutuskan. Saat itulah banyak pembelajaran aktif terjadi. Jadi, kita perlu memahami bahwa tidak ada sistem yang murni digerakkan oleh pembelajaran mesin, kita selalu menggabungkan pembelajaran mesin dengan pemikiran manusia. Kita melakukan otomatisasi paling banyak, tetapi ketika mesin tidak yakin tentang apa yang harus dilakukan, kita menggunakan pemikiran manusia.

BAB 7
PENGURANGAN DIMENSIONALITAS

7.1 PENDAHULUAN

Jika saya memiliki ratusan variabel, tidaklah mudah untuk membuat diagram sebar dan mencari tahu hubungan antar variabel. Untuk memahami data, apa yang dapat kita lakukan selain diagram sebar? Kita melakukan pengurangan dimensionalitas, dengan cara yang berprinsip dan salah satu algoritma yang paling umum digunakan yang disebut analisis komponen utama. Dengan sejumlah besar variabel, matriks dispersi mungkin terlalu besar untuk dipelajari dan ditafsirkan dengan benar. Akan ada terlalu banyak korelasi berpasangan antara variabel untuk dipertimbangkan. Tampilan data secara grafis mungkin juga tidak terlalu membantu jika kumpulan data sangat besar. Oleh karena itu, untuk menafsirkan data dalam bentuk yang lebih bermakna, perlu untuk mengurangi jumlah variabel menjadi beberapa kombinasi linier data yang dapat ditafsirkan. Setiap kombinasi linier akan sesuai dengan komponen utama. Ketika kita menghadapi situasi di mana kita memiliki sekumpulan fitur yang sangat besar dengan lebih sedikit titik data, dalam situasi ini pemasangan model dapat menghasilkan daya prediksi yang lebih rendah. Ini disebut Kutukan Dimensionalitas. Di sini solusinya bisa berupa menambah lebih banyak titik data atau mengurangi ruang fitur. Ini disebut reduksi dimensionalitas.

7.2 TEKNIK REDUKSI DIMENSI YANG BERBEDA

Ketika kita berhadapan dengan data yang sangat besar, kita tidak yakin tentang kegunaan informasi yang dikumpulkan. Jadi, kita cenderung menghapus beberapa variabel dengan asumsi bahwa itu tidak benar-benar berguna. Ini mungkin bukan pendekatan yang tepat karena ada beberapa teknik yang tersedia untuk menggabungkan variabel-variabel ini bersama-sama dan membuat Faktor atau Komponen Utama yang baru.

Ada banyak teknik yang dapat kita gunakan untuk reduksi Dimensionalitas seperti

1. Analisis Faktor,
2. Analisis Komponen Utama,
3. Analisis Diskriminan. Dll.

Kita akan mempelajari algoritma ini sekarang, Mari kita pahami mengapa kita melakukan analisis komponen utama. Analisis komponen utama adalah salah satu metode yang digunakan untuk memahami struktur dalam data, bentuk data, kovariansi data, yang tidak mungkin dilakukan dengan diagram sebar sederhana.

Analisis faktor berguna dalam kasus-kasus berikut:

Ketika kita memiliki sejumlah besar variabel dalam set data kita dan kita perlu mengurangi jumlah ini. Sebelum melakukan regresi atau analisis klaster pada set data dengan variabel berkorelasi. Saat menganalisis hasil survei di mana respons terhadap banyak pertanyaan cenderung sangat berkorelasi.

Sebelum melakukan reduksi Dimensionalitas, kita perlu memeriksa apakah reduksi dimensionalitas diperlukan atau tidak, dengan memeriksa Multikolinearitas. Kita melakukan reduksi dimensionalitas ketika asumsi OLS dilanggar karena Multikolinearitas. Dalam reduksi dimensionalitas, kita dapat mengambil salah satu dari 1. Pendekatan Ekstraksi Fitur atau 2. Pendekatan Pemilihan Fitur.

7.3 MULTIKOLINEARITAS

Multikolinearitas berarti variabel independen sangat berkorelasi satu sama lain. Dalam analisis regresi, asumsi penting adalah bahwa model regresi tidak boleh dihadapkan dengan masalah Multikolinearitas.

Mengapa Multikolinearitas menjadi masalah?: Jika tujuan penelitian adalah untuk melihat bagaimana variabel independen memengaruhi variabel dependen, dan jika variabel penjelas ini sangat berkorelasi, sulit untuk menentukan variabel spesifik mana yang memengaruhi variabel dependen. Cara lain untuk melihat masalah Multikolinearitas adalah: nilai P uji-t individual dapat menyesatkan. Artinya nilai P bisa tinggi yang berarti variabel tersebut tidak penting, meskipun variabel tersebut penting.

Bagaimana Mendeteksi Multikolinearitas?: Faktor Inflasi Varians (VIF) - Ini memberikan indeks yang mengukur seberapa besar varians (kuadrat dari deviasi standar estimasi) dari koefisien regresi yang diestimasi meningkat karena kolinearitas. Interpretasi VIF: Jika faktor inflasi varians dari variabel prediktor adalah 5, ini berarti varians untuk koefisien variabel prediktor tersebut adalah 5 kali lebih besar daripada jika variabel prediktor tersebut tidak berkorelasi dengan variabel prediktor lainnya.

Studi Kasus

Asumsikan bahwa Anda menganalisis suatu Produk (Mesin Cuci), Anda mengumpulkan informasi dari berbagai pengguna di seluruh negara dengan mengajukan pertanyaan berikut: Beri peringkat pada skala 1-5 (1: Sangat Rendah, 5: - Sangat Tinggi)

Seberapa Bagus Produk tersebut?
Seberapa Nyaman Anda dalam menggunakan produk tersebut?
Seberapa sering Anda menghadapi kesulitan dalam menggunakan produk tersebut?
Seberapa sering Anda menghubungi layanan pelanggan?
Bagaimana tanggapan dari Call Center?
Seberapa puas Anda dengan produk kami?

Anda ingin mengetahui apakah variabel-variabel ini memiliki masalah Multikolinearitas atau tidak. Kita dapat melakukannya dengan menjalankan model lm dan dengan menghitung VIF (Faktor Inflasi Varians).

#Langkah 1: Baca data

```
setwd('D:/R data')
Cus.drt <- read.csv("Cus_satis.csv", header=T)
```

#Langkah 2: Cari tahu masalah Multikolinearitas

```
Model = lm(Keseluruhan ~ ., data=Cus.drt)
```

```
Rsq = ringkasan(Model)$r.kuadrat vif = 1/(1 - Rsq)
vif
```

Karena nilai vif yang diperoleh kurang dari 5, kita dapat mengatakan bahwa masalah Multikolinearitas tidak ada.

7.4 ANALISIS KOMPONEN UTAMA

PCA: Jika jumlah fitur yang kita miliki lebih sedikit, kita dapat menggunakan diagram sebar untuk mengevaluasinya. Namun, bayangkan bagaimana Anda akan menjelajahi data 100 dimensi. Dapatkah kita membuat diagram sebar berpasangan bahkan untuk ini? Di sini, kita masih ingin mengetahui seperti apa struktur datanya. Apa hal berikutnya selain diagram sebar? Itulah yang kita sebut Proyeksi, Analisis Komponen Utama.

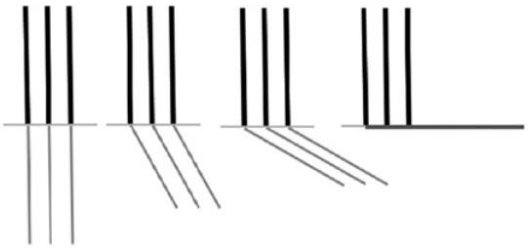
Mari kita bahas apa itu PCA? dan mengapa sangat berguna? Mari kita pahami jika bukan PCA, teknik apa lagi yang dapat digunakan?. PCA adalah teknik yang digunakan untuk memahami Struktur data, Bentuk berdasarkan kovariansi data.

Mari kita pahami apa itu Proyeksi? Kita cukup sering menggunakan istilah proyeksi ini, jadi mari kita pahami apa itu. Izinkan saya mengambil beberapa contoh untuk menjelaskan istilah ini. Bayangkan kita berada di lapangan kriket, melihat tiga tunggul yang kita gunakan dalam kriket. Ini adalah struktur 3 dimensi. Bayangkan kita mengambil senter, memfokuskan cahaya pada tunggul dan mengamati bayangan di sisi lain. Ini disebut proyeksi. Jika kita mengubah sumber cahaya dengan sudut yang berbeda, kita mendapatkan proyeksi yang berbeda. Data masih sama, yang kita lakukan adalah memproyeksikannya ke arah yang berbeda untuk mendapatkan Dimensi yang berbeda. Asumsikan bahwa jika kita memproyeksikannya secara horizontal pada sudut 180 derajat kita mendapatkan garis lurus sebagai proyeksi. Sekarang pahami dimensi garis lurus. Itu adalah 1 dimensi. Tetapi kita perlu berpikir, apa kehilangan informasi ketika kita memproyeksikan data pada sudut 180 derajat? Jika data asli telah dihapus setelah proyeksi, dapatkah kita merekonstruksi data? Berapa banyak informasi yang hilang? Ini adalah pertanyaan yang perlu kita tanyakan pada diri sendiri sebelum memproyeksikan data.

PCA adalah transformasi ortogonal linier yang mengubah data ke sistem koordinat baru sehingga varians terbesar oleh setiap proyeksi data berada pada koordinat pertama, varians terbesar kedua pada koordinat kedua, dan seterusnya. PCA menggunakan proyeksi ortogonal dari variabel yang sangat berkorelasi ke sekumpulan nilai variabel yang tidak berkorelasi secara linier yang disebut komponen utama. Jumlah komponen utama kurang dari atau sama dengan jumlah variabel asli. Transformasi linier ini didefinisikan sedemikian rupa sehingga komponen utama pertama memiliki varians yang paling besar. Ini memperhitungkan sebanyak mungkin variabilitas dalam data dengan mempertimbangkan fitur yang sangat berkorelasi.

Sekarang mari kita pahami terminologinya:

1. Arah proyeksi: Sumber cahaya dan sudut cahaya
2. Data mentah dalam ruang 3d: tunggul sebenarnya, yang tetap dalam ruang 3 dimensi.
3. Data yang diproyeksikan dalam ruang 2-D: bayangan di ujung lainnya.



Bergantung pada arah proyeksi, bayangan akan berubah tetapi data tidak. Jika saya mengubah cahaya lebih jauh, saya mendapatkan proyeksi yang berbeda. Jadi, proyeksi yang kita dapatkan adalah fungsi dari mana cahaya datang dan bayangan selalu berada di sisi lain. Bayangkan kita memproyeksikan tiga tunggul dari empat sudut yang berbeda dan kita mendapatkan Proyeksi A, B, C, D. Sekarang bagaimana kita dapat memutuskan proyeksi mana di A, B, C, D yang merupakan Proyeksi terbaik? Kita mengukur kebaikan proyeksi berdasarkan seberapa banyak informasi yang dapat dipertahankannya. Proyeksi yang mempertahankan informasi maksimum disebut proyeksi terbaik. Asumsikan bahwa kita menganggap 'A' sebagai proyeksi terbaik? Apa yang dilakukan proyeksi A secara berbeda dari yang lain? Apa yang terjadi ketika kita melakukan proyeksi lain adalah kita kehilangan informasi. Sebenarnya, data mentah ada dalam 3 dimensi, lalu kita melakukan proyeksi, proyeksi tersebut merupakan perkiraan data mentah. Setelah proyeksi, kita mendapatkan ruang 2 dimensi yang berarti kita kehilangan beberapa informasi karena kita beralih dari 3D ke 2D.

Sasaran pengurangan dimensionalitas, proyeksi adalah kehilangan informasi seminimal mungkin. Jadi, jawaban yang benar bergantung pada kriteria yang kita gunakan. Di sini kriteria kita adalah meminimalkan hilangnya informasi atau proyeksi yang mempertahankan informasi sebanyak mungkin dalam data mentah, adalah proyeksi terbaik. Dalam definisi itu, dalam Proyeksi A adalah yang Terbaik karena mempertahankan struktur maksimum dalam data mentah. Tentu saja, kita akan kehilangan beberapa struktur tetapi kerugiannya tetap minimum. Dalam kasus D, jika mereka hanya menunjukkan proyeksi tetapi bukan data mentah, kita tidak dapat merekonstruksi data mentah. Itu berarti kita tidak dapat mengetahui apa data mentah itu. Kita akan berpikir bahwa sebenarnya data mentah adalah garis lurus. Jadi kita kehilangan banyak informasi dalam D dan sangat sedikit info yang dipertahankan dalam D. Jadi, ingat ini, setiap kali kita memproyeksikan data, kita seharusnya kehilangan informasi, tetapi proyeksi di mana kita memiliki jumlah kehilangan informasi minimum dianggap sebagai proyeksi terbaik. Jadi, kita dapat mendefinisikan gagasan proyeksi terbaik sebagai proyeksi di mana kita kehilangan informasi seminimal mungkin. Mari kita ambil satu contoh lagi, bayangkan data berbentuk bola saat kita memproyeksikan data 2D, kita akan mendapatkan data yang diproyeksikan dalam ruang 1D. Di sini kita kehilangan beberapa informasi karena titik-titik pada garis horizontal yang sama akan muncul sebagai satu titik di layar. Jadi kita kehilangan perbedaan. Karena kita tidak dapat membedakan perbedaan antara dua titik data yang berbeda pada garis horizontal yang sama. Namun, kita dapat mengurangi dimensionalitas dari 2D menjadi 1D. Bayangkan jika kita meletakkan cahaya pada arah yang berbeda, kita mendapatkan proyeksi yang berbeda. Mari kita asumsikan jika kita memiliki tiga proyeksi A, B, C, beri tahu saya Proyeksi Terbaik? Tidak, kita tidak dapat mengatakannya karena semuanya sama. Setiap kali data berbentuk bola, itu berarti data tersebut terdistribusi secara merata dan kedua fitur tersebut sama sekali tidak berkorelasi, itu

berarti fitur tersebut sama sekali tidak bergantung satu sama lain. Dalam hal ini, apa pun yang Anda lakukan dalam hal proyeksi, Anda akan kehilangan jumlah informasi yang sama. Itulah sebabnya kami mempelajari Kovariansi karena kami ingin mencapai keadaan, di mana data menjadi sangat bulat sehingga semua proyeksi sama-sama baik atau sama-sama buruk. Secara konseptual, bisa ada cara yang tak terbatas untuk memproyeksikan data, tetapi saat kita memproyeksikan data secara ortogonal, kita menangkap informasi dari berbagai jenis titik data, dan berbagai jenis struktur. Dalam kasus pertama, saya menangkap sebaran vertikal dan di kasus lain, saya menangkap sebaran horizontal. Apa yang saya tangkap dari satu sisi benar-benar berbeda dari apa yang saya tangkap dari sisi yang lain. Kedua proyeksi ini jika digabungkan akan menangkap informasi lengkap dari data mentah.

Jadi, idenya adalah ketika saya mendapatkan proyeksi pertama tempat saya menangkap sebagian besar data, proyeksi terbaik berikutnya yang akan memberi saya info lain yang hilang dalam proyeksi pertama. Itu harus ortogonal terhadap proyeksi pertama. Karena datanya dalam 2D, saya hanya perlu 2 proyeksi untuk menangkap informasi lengkap dalam data. Bahkan melihat dua proyeksi lainnya, jumlah info yang ditangkap dalam kedua proyeksi itu sama persis. Izinkan saya memberi Anda contoh lain dari proyeksi ortogonal ketika kita menggunakan proyeksi ortogonal yang berbeda yang berada pada sudut 90 derajat satu sama lain, kedua proyeksi itu ortogonal yang berarti informasi yang ditangkap oleh satu proyeksi sama sekali berbeda dari informasi yang ditangkap oleh proyeksi lain, keduanya digabungkan bersama-sama saya akan mendapatkan informasi yang lengkap. Kita dapat mengatakan proyeksi itu baik ketika proyeksi memperhitungkan bentuk, dan mempertahankan varians. Proyeksi yang mempertahankan varians maksimum kita sebut sebagai Komponen Prinsipal Pertama, arah di mana varians maksimum dipertahankan. Di sini, Varians berarti jumlah Informasi yang dipertahankan. Bayangkan jika fitur tersebut tidak memiliki varians, maka fitur tersebut tidak memiliki konten informasi di dalamnya. Jadi pelestarian varians adalah salah satu kriteria dalam proyeksi terbaik. Komponen utama pertama menangkap varians maksimum dan apa pun varians yang tersisa akan ditangkap oleh Komponen utama kedua dan seterusnya. Sekarang bayangkan jika Anda memiliki data 100 dimensi, Komponen utama pertama tetap menangkap varians maksimum dan komponen utama kedua yang ortogonal terhadap yang pertama akan menangkap varians terbaik kedua. Kemudian apa pun yang tersisa akan ditangkap oleh yang ketiga yang akan ortogonal terhadap dua Komponen utama pertama dan seterusnya. Dalam PCA, kami mencoba menemukan dalam urutan berurutan serangkaian proyeksi, sehingga Komponen utama pertama menangkap varians maksimum dan seterusnya hingga akhir.

Jika data saya memiliki N dimensi, saya dapat memiliki paling banyak N komponen utama. Mari kita bayangkan titik data 2 dimensi, terdapat X titik data di wilayah tersebut, Mu adalah rata-rata titik data tersebut. (Mu(X)). X(n) adalah titik data ke-n dalam titik data X, dan jika kita memproyeksikannya bersama-sama, konsep kita adalah mempertahankan varians maksimum. Mari kita asumsikan bahwa kita memproyeksikan cahaya dari dua sumber, dan Mu(Y) adalah versi proyeksi dari Rata-rata dan Y(n) adalah versi proyeksi dari titik data ke-N.

Bayangkan, kita melihat proyek dari kedua sisi, kita dapat memahami bahwa proyeksi pertama memiliki varians maksimum yang dipertahankan jika dibandingkan dengan yang kedua. Sekarang kita harus menghitung berapa rata-rata yang diproyeksikan Mu(Y), yaitu N atas semua Y(n). Tujuannya adalah untuk memaksimalkan pelestarian varians. Saya melihat ruang yang diproyeksikan dan mencari tahu proyeksi mana yang lebih baik. Proyeksi tidak lain adalah mengambil data mentah, mengalikannya dengan vektor, dan itu memberi Anda satu angka, yang memberi Anda satu proyeksi. Y(n) adalah angka tunggal yang memberi tahu saya seberapa jauh dari titik itu dan seterusnya. Sekarang jika saya ingin menghitung rata-rata, yang saya katakan adalah 1 atas N, rata-rata atas semua nilai Y. Ini hanya contoh satu titik, tetapi pada kenyataannya kita memproyeksikan semua titik ruang dan mengambil nilai rata-rata, dari titik-titik itu dan Jika saya mengembangkan persamaan W, transpos Xn (W adalah vektor linier, yang memberi tahu saya arah proyeksi,) saya keluarkan dari penjumlahan, dan bagian yang tersisa tidak lain adalah rata-rata. Rata-rata dapat dilakukan dengan dua cara, pertama Anda memproyeksikan semua titik data dan mengambil rata-rata, atau mengambil rata-rata dan memproyeksikan semua titik nanti. Dalam kedua situasi, Anda akan mendapatkan hasil yang sama. Sekarang, apa varians dalam ruang Y? Varians tidak lain adalah mengambil setiap titik data, Seberapa jauh dari rata-rata, jika Anda memiliki data bersama dengan arah awan, Anda mencoba untuk mempertahankan atau memaksimalkan varians, karena bentuk yang Anda dapatkan lebih panjang, itulah yang ingin kita lakukan, jika saya mengganti nilai Y, Y tidak lain adalah W transpos X di sini, jika saya melakukan sedikit matematika di sini, yang terjadi adalah, W transpos kovariansi dikali W, ini menjadi ruang kovariansi. Jadi Idenya adalah ini menjadi kovariansi, pada dasarnya apa yang kita lakukan adalah kita mengambil W transpos dikali kovariansi dikali W dan ini menjadi, menjadi satu angka, angka ini akan jauh lebih rendah di sini, angka yang sama akan jauh lebih tinggi dalam kasus lain. Jadi, idenya adalah jika Anda ingin memaksimalkan Varians, temukan W yang memaksimalkan Varians.

Sekarang ini menjadi masalah optimasi, beginilah cara kita menjelaskannya, Anda mengambil matriks kovariansi data, Anda memiliki data mentah seperti data empat dimensi, Anda mengambil rata-rata semua kolom, kita mendapatkan empat angka, ambil matriks kovariansi, matriks itu disebut W, Anda ingin menemukan W itu, yang merupakan arah, sehingga varians dimaksimalkan dalam ruang yang diproyeksikan. ketika Anda melakukannya pada dasarnya, Anda memecahkan masalah vektor eigen, dari kovariansi. Sekarang Anda mungkin telah memahami secara konseptual apa yang coba kita lakukan. Kita mencoba menemukan arah yang memaksimalkan pelestarian varians. Kita menemukan W yang memaksimalkan dan itu sepenuhnya terkait dengan matriks kovariansi data.

Mari kita lihat ini dalam kasus dataset yang memiliki 8 variabel numerik dan 500 observasi. Di sini Anda bisa mendapatkan matriks 8*8, dan ketika kita melihat kovariansinya, diagonalnya adalah varians dari setiap fitur, apa penyebaran fitur itu sendiri, itu adalah matriks 8*8. Kami telah mengambil penjumlahan dari 500 titik, masing-masing adalah penjumlahan ukuran 8*8, mu dari x dan mu dari x adalah titik 8 dimensi, pusat dari semua titik, Xn adalah titik data ke-n, pada dasarnya Anda mengambil rata-rata dari 500 matriks 8*8, dan itu memberi Anda matriks kovariansi. Ketika Anda mengambil vektor eigen, itu menjadi

komponen utama pertama. Ketika Anda menghitung nilai eigen pada dasarnya, inilah yang memberi Anda 8 vektor eigen. Ingat data mentah memiliki 8 dimensi, jadi proyeksinya juga akan menjadi 8. Jadi, kita memperoleh total 8 komponen utama, dan untuk setiap komponen utama, ini juga memberi Anda Nilai Eigen, yang menyatakan ke arah mana varians yang dimiliki setiap komponen utama, kita dapat melihat komponen utama pertama yang mempertahankan jumlah varians yang sangat besar.

Dalam PCA, yang coba kita lakukan adalah, kita mengambil seluruh varians data, kita mencoba untuk menekan semua varians pada komponen utama pertama dari data, jelas, kita tidak bisa mendapatkan semua varians pada komponen utama pertama, tetapi kita dapat mencoba untuk memaksimalkannya. Itulah yang kita katakan setelah melihat hasilnya, kita mencoba untuk mendapatkan varians maksimum yang mungkin pada komponen utama pertama, kemudian yang kedua dan seterusnya. Jadi, Idenya adalah data 8 dimensi memberi Anda 8 komponen utama, tetapi komponen utama pertama akan memiliki kovariansi maksimum. Kalau Anda akan mengambil kedelapan dimensi tersebut, maka kita akan mencakup semua varians. Namun, data tersebut memiliki beberapa gangguan, tidak apa-apa jika kehilangan sejumlah informasi, penting untuk menjaga jumlah informasi yang tepat, dan jenis informasi yang tepat. Oleh karena itu, kita memproyeksikan data pada 2 atau 3 komponen utama.

Mari kita lihat beberapa Contoh tanpa pengawasan:

Dalam PCA, proyeksi adalah satu jenis struktur, mungkin berpikir bahwa data memiliki 200 dimensi atau 1000 dimensi, kolom-kolom tersebut muncul bukan karena data memiliki kolom-kolom tersebut, kolom-kolom tersebut muncul karena seseorang memutuskan bahwa kolom tersebut penting untuk dimiliki. Selalu baik untuk bersikap konservatif dan memiliki lebih banyak kolom di awal karena Anda tidak tahu bagaimana Anda akan menggunakannya nanti. Perusahaan taksi dapat mengumpulkan data Anda seperti berapa lama Anda dapat menunggu, bagian kota tempat Anda tinggal, jam berapa Anda mulai bekerja, dll. Mereka tidak tahu cara menggunakannya sekarang, tetapi mungkin mereka akan menggunakannya nanti. Umumnya, kami mengumpulkan lebih banyak kolom untuk bersikap konservatif karena nanti kami tidak boleh berkata oh! Saya seharusnya mengumpulkan bidang data itu, itulah sebabnya kami menyebut pengumpulan data sebagai seni, itu adalah seni berpikir fitur. Tetapi ketika Anda mengumpulkan lebih banyak kolom, kumpulan data Anda menjadi lebih tebal, sekarang kami perlu mencari tahu apa struktur data yang sebenarnya. Jadi PCA berguna di sana. Studi Kasus: Sebuah organisasi perangkat lunak menghadapi masalah pengurangan karyawan dan mereka ingin mengetahui alasan karyawan meninggalkan organisasi mereka. Mereka mengumpulkan berbagai dimensi dan ingin memadatkannya menjadi beberapa fitur sehingga mereka dapat berkonsentrasi pada masalah tersebut. Sekarang tugas kita adalah mencari tahu apakah kita dapat mengurangi dimensi atau tidak. Kita ingin mencari tahu berapa banyak dimensi yang cukup untuk memuat setidaknya 80% varians dalam data.

Analisis faktor eksploratori (EFA) adalah teknik umum dalam ilmu sosial untuk menjelaskan varians antara beberapa variabel terukur sebagai sekumpulan variabel laten yang

lebih kecil. EFA sering digunakan untuk mengonsolidasikan data survei dengan mengungkap pengelompokan (faktor) yang mendasari pertanyaan individual.

7.5 ANALISIS FAKTOR YANG DILAKUKAN

#Langkah 1: Baca data

```
setwd('D:/R data')
Emp.fa <- read.csv("Emp_satis.csv", header=T)
```

#Langkah 2: Instal dan muat paket

```
install.packages("psych")
library(psych)
```

#Langkah 3: Jelajahi data

```
head(Emp.fa)
```

#tampilkan contoh data `dim(Emp.fa)`

#periksa dimensi `str(Emp.fa)`

#tampilkan struktur data `fix(Emp.fa)`

#Langkah 4: Persiapan Data

```
fa.req <- subset(Emp.fa, select=-c(Empid,Overall))
```

#Langkah 5: Hitung dan tampilkan matriks korelasi

```
corMat <- cor(fa.req) corMat
```

Langkah 6: Lakukan Analisis Faktor

Gunakan fa() untuk melakukan analisis faktor eksplorasi sumbu utama miring dan simpan solusi ke variabel R. Untuk memperoleh solusi faktor, kita akan menggunakan fungsi fa() dari paket psych, yang menerima argumen utama berikut.

```
Mod.fa <- fa(r = corMat, nfactors = 3, rotate = "varimax",fm="ml")
```

Di mana,

#r: matriks korelasi

#nfactors: jumlah faktor yang akan diekstraksi (default = 1)

#rotate: salah satu dari beberapa metode rotasi matriks, seperti "varimax" atau "oblimin"

#fm: salah satu dari beberapa metode pemfaktoran, seperti "pa" (sumbu utama) atau "ml" (kemungkinan maksimum)

Perhatikan bahwa beberapa metode rotasi dan pemfaktoran tersedia saat melakukan EFA. Metode rotasi dapat dideskripsikan sebagai ortogonal, yang tidak memungkinkan faktor yang dihasilkan untuk dikorelasikan, dan miring, yang memungkinkan faktor yang dihasilkan untuk dikorelasikan. Metode pemfaktoran dapat dijelaskan sebagai umum, yang digunakan ketika tujuannya adalah untuk mendeskripsikan data dengan lebih baik, dan komponen, yang digunakan ketika tujuannya adalah untuk mengurangi jumlah data.

#Langkah 7: Menampilkan keluaran solusi

```
Mod.fa
```

7.6 MELAKUKAN ANALISIS KOMPONEN UTAMA

#Langkah 1: Membaca data

```
setwd('D:/R data')
Emp.drt <- read.csv("Emp_satis.csv", header=T)
```

#Langkah 2: Menjelajahi data

```
head(Emp.drt)
```

#menampilkan contoh data `dim(Emp.drt)`

#memeriksa dimensi `str(Emp.drt)`

#menampilkan struktur data `fix(Emp.drt)`

```
colnames(Emp.drt)
```

#Langkah 3: Persiapan Data

```
Emp.req <- subset(Emp.drt, select=-c(Empid))
```

#Langkah 4: Menemukan masalah Multikolinearitas

```
Model = lm(Keseluruhan ~ ., data=Emp.req)
Rsq = ringkasan(Model)$r.kuadrat vif = 1/(1 - Rsq)
vif
```

Karena nilai vif yang diperoleh lebih besar dari 5, kita dapat mengatakan bahwa Multikolinearitas hadir.

Jadi, kami memutuskan untuk melakukan reduksi Dimensionalitas.

#Langkah 5: Persiapan Data

```
Emp.pro <- subset(Emp.drt, select=-c(Empid,Overall))
```

#Langkah 6: Buat Komponen Utama

```
Emp.pca <- prcomp(Emp.pro,center=TRUE,scale=TRUE) print(Emp.pca)
```

#Langkah 7: Buat Plot Layar

```
plot(Emp.pca, type="lines")
```

#Metode ringkasan menjelaskan pentingnya PC. Baris pertama menjelaskan lagi deviasi standar yang terkait dengan setiap PC. Baris kedua menunjukkan proporsi varians dalam data yang dijelaskan oleh setiap komponen sementara baris ketiga menjelaskan proporsi kumulatif varians yang dijelaskan.

#Langkah 8: Lakukan ringkasan

```
summary(Emp.pca)
```

BAB 8
PENGELOMPOKAN

8.1 PENDAHULUAN

Pengelompokan adalah proses pengorganisasian objek ke dalam kelompok yang anggotanya serupa dalam beberapa hal, yang berkaitan dengan pencarian Struktur dalam kumpulan Data Tak Berlabel. Oleh karena itu, klaster adalah kumpulan objek yang "Serupa" di antara mereka dan "Tidak serupa" dengan objek yang termasuk dalam klaster lain. Ambil contoh perusahaan Laptop yang mempromosikan merek laptop terbarunya. Mereka ingin menyesuaikan model iklan, orang yang tinggal di satu bagian kota berbeda dengan orang yang tinggal di bagian kota lainnya. Kita perlu mengirim iklan yang berbeda ke orang yang berbeda. Apa yang kita lakukan di sini? Kita melakukan pengelompokan secara intuitif. Kita dapat memulai dengan satu atau dua pesan dan kita dapat melakukannya hingga ke iklan individual, yaitu iklan yang dipersonalisasi. Jadi, pengelompokan adalah hal yang kita lakukan saat kita ingin membuat kelompok karena Anda tidak dapat membuat satu pesan untuk setiap orang, itu terlalu mahal di satu ujung spektrum, dan Anda tidak dapat mengirim pesan yang sama ke semua orang karena terlalu kasar. Sekarang apa yang akan Anda lakukan? Anda melakukan sesuatu di tengah yang disebut pengelompokan?

Sekarang pikirkan tentang pusat layanan pelanggan dari perusahaan taksi. Mereka ingin tahu apa saja masalah CRM, mereka mulai dengan beberapa masalah dan solusi, lalu menyempurnakan rangkaian masalah berdasarkan masalah CRM. Pikirkan tentang ini, bagaimana mereka mengetahui masalah ini bahkan sebelum perusahaan itu berdiri? Mereka mungkin telah mengambil masalah yang diangkat dari perusahaan taksi sebelumnya, lalu menjadi masalah klasifikasi. Lalu bagaimana Anda menemukan hal-hal baru yang muncul di pasar Anda? Kami melakukan pengelompokan. Berbagai jenis teknik pengelompokan:

1. Pengelompokan Eksklusif (Pengelompokan Partisi): Pengelompokan K-Means
2. Pengelompokan Tumpang Tindih: Pengelompokan Fuzzy C-Means
3. Pengelompokan Hirarkis: Atas-Bawah (Divisif), Bawah-Atas (Aglomeratif)
4. Pengelompokan Probabilistik: Campuran Gaussian
5. Pengelompokan Spektral

8.2 PENGELOMPOKAN PARTISISIONAL

Pengelompokan Partisional dapat berupa Keras atau Lunak, dalam Pengelompokan Keras seperti Anda mengelompokkan titik data ke satu fitur klaster tertentu. Karena termasuk dalam klaster ini, maka tidak dapat termasuk dalam klaster lain. Dalam Pengelompokan Lunak, kami menganggap titik data tampak seperti termasuk dalam klaster ini tetapi bahkan memiliki fitur klaster lain sehingga sebagian termasuk dalam klaster ini dan sebagian lagi termasuk dalam klaster lain. Jadi, izinkan saya menerapkan kedua strategi tersebut ke titik data ini, yang disebut pengelompokan lunak. Awalnya, saat Anda membuat klaster, Anda tidak tahu seberapa akurat pusat klaster tersebut, jadi Anda tidak ingin melakukan pengelompokan

keras. Jika data Anda memiliki terlalu banyak gangguan, Anda tidak ingin melakukan pengelompokan keras, karena Anda tidak yakin.

Hirarkis: Ini bukan teknik pengelompokan, ini sebenarnya adalah filosofi dalam melihat data, data Anda selalu memiliki hierarki. Tidak peduli di domain mana Anda berada, bisa jadi urutan gen, keuangan, asuransi, selalu ada hierarki. Dunia ini terbuat dari hierarki. Hirarki adalah bagian integral dari alam, Otak memahami hierarki. Setiap kali seseorang memberi Anda data, pertanyaan pertama yang harus ditanyakan adalah apa hierarki di dalamnya? Model mental Anda selalu di sini adalah hierarki. Bagaimana Anda menemukan hierarki itu terserah Anda. Tingkat hierarki apa yang akan Anda gunakan terserah Anda. Seberapa banyak tingkat yang Anda pedulikan dalam hierarki itu terserah Anda. Bayangkan saja sistem menu di pusat panggilan telekomunikasi jika Anda memiliki masalah ini, tekan 1 dan jika Anda memiliki masalah itu tekan 2 dan seterusnya.., bahkan setelah menekan 1 Anda akan mendapatkan lebih banyak opsi menu dan seterusnya. Hirarki adalah proses bawaan dalam sistem bisnis, organisasi, dan bahkan dalam Data. Pembelajaran tanpa pengawasan adalah tentang mempelajari Struktur data, ketika kita berbicara tentang struktur, yang dimaksud sebenarnya adalah hierarki. Anda mungkin bertanya kepada saya, seperti jika semuanya ada dalam hierarki, lalu mengapa kita memerlukan Pengelompokan Partisi? Jika kita ingin melakukan pendekatan pengelompokan bottom-up, kita harus mengetahui jarak berpasangan antara semua pasangan titik data satu sama lain. Apakah mungkin untuk sejumlah besar titik data? Tidak, jadi, jika kita melakukan pendekatan Top-down, kita perlu melakukan pengelompokan Partisi di setiap tahap pengelompokan. Idenya di sini adalah, izinkan saya membangun 5 kluster terlebih dahulu karena saya tidak dapat memiliki sistem menu yang berjalan hingga 100. Jadi, saya akan membangun 5 kluster terlebih dahulu, lalu saya membangun sub-kluster karena saya tahu data mengikuti Hirarki. Itu berarti saya menggunakan kedua metode pengelompokan dalam hal ini, tetapi model konseptual saya adalah hierarki. Itu berarti kita menggunakan pengelompokan Partisi sebagai bagian dari pengelompokan Top-Down. Inilah yang diharapkan Industri dari seorang Ilmuwan Data. Pengelompokan K-Means bergantung pada:

1. Anda harus mengetahui jumlah klaster
2. Tergantung pada inisialisasi yang Anda temui, dapatkan klaster yang berbeda
3. Bergantung pada fungsi jarak, Anda mungkin mendapatkan klaster yang berbeda.

Ketika saya mengatakan fungsi jarak, tidak ada yang namanya fungsi jarak, Anda harus membuat fungsi jarak Anda sendiri, seperti saat pergi piknik, Anda perlu membawa meja piknik bersama Anda karena kita tidak akan pergi ke restoran. Seperti halnya, Anda perlu membawa fungsi jarak Anda sendiri ke bagian pembelajaran mesin. Ini adalah bagian yang sangat penting, semakin baik fungsi jarak yang dapat Anda definisikan, semakin baik pula algoritme pembelajaran mesin yang dapat Anda buat. Pengelompokan K-means tidak peduli bagaimana Anda mendefinisikan fungsi Jarak. Jika semuanya adalah alat, kita tidak perlu melakukan kursus ini dengan begitu banyak tekanan. Apa peran Anda sebenarnya sebagai ilmuwan data? Di situlah keterampilan kita akan muncul.

Sekarang pikirkan tentang bagaimana Anda akan mendefinisikan, fungsi jarak antara dua resume? Senioritas adalah salah satunya, keahlian adalah salah satunya, perusahaan tempat ia bekerja, universitas tempat ia kuliah adalah yang lainnya dan seterusnya. Pikirkan tentang fungsi jarak antara satu profil orang dengan profil orang lainnya, berapa jarak antara dua film? aktor, sutradara, musik Anda dapat mendefinisikan fungsi jarak yang sangat rumit. Jadi, salah satu peran terpenting ilmuwan data adalah mendefinisikan fungsi jarak. Semua kreativitas muncul dalam hal-hal ini, tidak ada yang akan mendefinisikan hal-hal ini untuk Anda, karena Anda lebih mengetahui domainnya, Anda lebih mengetahui datanya, dan Anda lebih mengetahui distribusinya, tugas Anda adalah mendefinisikan hal-hal ini.

Sekarang jenis pengelompokan ketiga disebut pengelompokan spektral, Sering kali data yang datang kepada kita dalam bentuk ini, seperti tabel, di mana setiap baris adalah titik data dan kita memiliki fitur khusus sebagai kolom atau dimensi. Anda dapat menggunakan pengelompokan K-means atau pengelompokan aglomeratif di sini hanya ketika Anda mendefinisikan jarak antara dua titik data. Oleh karena itu, fungsi jarak penting, tetapi sering kali Anda mendapatkan data dengan tipe yang berbeda, yang memiliki kemiripan di antara berbagai hal. Kita tidak akan mengetahui seperti apa tampilan datanya, mereka hanya akan memberi tahu kita bahwa ini mirip dengan itu. Dalam kasus tersebut, data tampak seperti grafik di mana setiap simpul adalah titik data, tidak memiliki fiturnya sendiri, yang kita ketahui hanyalah jarak antara titik data.

Tujuan dalam pengelompokan adalah meminimalkan jarak antara titik dan perwakilannya secara intuitif. Bayangkan sistem demokrasi, karena tidak semua orang dapat duduk di Parlemen, kami mengirimkan perwakilan kami, sehingga perwakilan tersebut paling dekat dengan semua titik data. Gunakan analogi itu untuk memikirkan pengelompokan k-means bahwa vektor rata-rata adalah perwakilan dari sekelompok orang atau sekelompok titik data dan vektor rata-rata yang baik harus dekat dengan semua titik yang diwakilinya.

Bagaimana kita menggunakan analogi ini untuk mendefinisikan fungsi Tujuan?
Apa saja parameternya?

Ada dua jenis parameter, 1. Parameter Asosiasi yang mengatakan bagaimana Anda menetapkan klaster ke titik data, dan 2. Bagaimana Anda memperbarui vektor rata-rata? Mari kita lihat Rumus Pengelompokan;

$$J(\mathbf{m}, \mathbf{z}) = \sum_{n=1}^{N} \sum_{k=1}^{K} Z_{n,k} \Delta(\mathrm{X}_n, m_k)$$

Dalam rumus, Anda dapat melihat dua simbol Penjumlahan, satu untuk Titik data dan satu lagi untuk Klaster, ini tidak ada hubungannya dengan Jumlah dimensi. Di sini saya menggunakan fungsi jarak Euclidean.

Saat melakukan pengelompokan K-means. Kita perlu memahami bahwa kita telah memilih seorang perwakilan untuk sebuah kelompok ketika ia pindah dari kelompok kita ke

kelompok lain, ia menjadi perwakilan untuk klaster baru dan orang lain akan menjadi perwakilan untuk kelompok yang tertinggal. Idenya adalah kita dapat memiliki sejumlah perwakilan, tetapi saya ingin melakukannya dengan cara yang optimal. Anggap ini sebagai masalah ayam dan telur, jika kita memutuskan asosiasi maka kita dapat menghitung rata-rata, atau memutuskan rata-rata dan menghitung asosiasi. Ini adalah dua masalah yang terkait dan kita tidak tahu harus mulai dari mana. Kita dapat memulai dengan menetapkan pusat klaster secara acak, atau kita dapat memulai dengan menetapkan lokasi acak dalam data, sebagai rata-rata klaster, salah satu dari keduanya adalah titik awal yang valid. Seperti ini: Diberikan Delta - temukan Rata-rata atau Diberikan Rata-rata temukan Delta. Ketika masalah ini terpecahkan, akan muncul tahap di mana mean dan delta tidak akan berubah, itulah yang kita sebut sebagai konvergensi.

Pengelompokan adalah algoritma iteratif, kita mulai dengan beberapa vektor mean pada awalnya, dan inisialisasi acak, kita akan membahas lebih lanjut inisialisasi nanti, lalu apa yang terjadi? Apa saja titik yang menuju m1 dan apa saja titik yang seharusnya menuju m2? Kemudian pertanyaan kedua, mengingat mean dalam iterasi p, kita ingin menemukan penugasan klaster. Ketika saya mengatakan Iterasi, itu dimulai dari mean, menemukan delta yang tepat, dan kemudian menggunakan delta untuk menemukan mean berikutnya. Jadi ini adalah satu persamaan. Awalnya, t adalah 0 yang berarti itu adalah iterasi ke-0, dan perlahan-lahan kita tingkatkan t.

Ketika kita memiliki titik data, kita mengambil titik-titik tersebut sebagai vektor dan menghitung jarak, menetapkan titik data ke vektor, berdasarkan jarak. Kemudian memperbarui nilai mean, dan melanjutkan untuk iterasi berikutnya. Idenya adalah untuk memulai dengan beberapa cara, kami menggunakan cara rata-rata untuk menetapkan klaster, dan kemudian kami mengambil penetapan klaster untuk menghitung cara rata-rata. Kemudian kami mengulangi lagi, dan lagi hingga konvergen. Ini disebut langkah Ekspektasi. Dalam langkah maksimisasi, dikatakan bahwa mengingat ini adalah penetapan, maksimalkan fungsi objektif.

Langkah E: Pusat Klaster --> Penetapan Klaster
Langkah M: Penetapan Klaster --> Pusat Klaster.

Pikirkan saja tentang Berjalan: Jika kita mengambil satu kaki, itu adalah E-curam dan kemudian kita mendaratkan kaki dan melepas kaki lainnya yang merupakan Langkah M. Pada titik berjalan mana pun, Anda tidak dapat mengangkat kedua kaki atau mendaratkan kedua kaki.

Mari kita jawab dua pertanyaan lagi:

Bagaimana kita memilih K?
Bagaimana kita Menginisialisasi?

Bagaimana kita menetapkan perwakilan secara optimal ke titik data? Saat memecahkan masalah pengoptimalan ini, tidak ada jaminan bahwa ada satu solusi yang baik. Sekarang

Bayangkan bahwa tujuan Anda adalah mendaki bukit terdekat, atau tujuannya adalah jatuh ke lembah terdekat. Anda bisa jatuh untuk menemukan titik terendah atau Anda memanjat untuk menemukan bukit terdekat. Jika saya ingin mendaki bukit, saya akan berjalan ke arah yang memberi saya gradien maksimum, dan saya akan mencapai satu puncak, tetapi jika saya mulai di titik lain, saya akan mencapai puncak lainnya, dan sebagian besar algoritme pembelajaran mesin memiliki jenis masalah ini. Dalam banyak kasus, fungsi pengoptimalan seperti ini, ia memiliki banyak titik optimal lokal yang salah satunya adalah titik optimal global.

Apa yang dijamin oleh kluster K-Means di mana pun Anda memulai, saya akan menjamin titik optimal lokal, yang tidak dapat dijamin adalah bahwa Anda akan mencapai titik optimal global. Oleh karena itu, Inisialisasi memainkan peran yang sangat penting dalam pengelompokan K-Means. Bahkan jika Anda melakukan pengelompokan K-Means sejuta kali, tidak ada yang dapat menjamin Anda solusi Optimal global. Yang dapat kita lakukan dalam pengelompokan K-Means adalah mencoba dengan titik inisialisasi yang berbeda dan menemukan titik optimal, dan di antara titik optimal lokal tersebut, Anda dapat memilih yang terbaik. Jadi, setiap kali kita melakukan pengelompokan, kita perlu mencoba menyebarkan titik awal di antara semua titik data.

Inisialisasi Titik Pertama Terjauh: Ini adalah teknik yang digunakan untuk menangani masalah inisialisasi. Dalam metode ini, kita memilih rata-rata terlebih dahulu dan memilih titik terjauh dari rata-rata, dan menggunakannya sebagai pusat klaster pertama. Sekarang anggap ini sebagai benih, dan hitung titik terjauh dari benih baru ini dan anggap titik itu sebagai pusat klaster kedua, sekarang saya memiliki dua pusat klaster yang menjamin penyebaran data maksimum. Sekarang saya memerlukan benih ketiga yang harus terjauh dari kedua benih, kita perlu mengatakan jarak dari kedua benih harus maksimum, kita hitung skor semua titik sedemikian rupa sehingga benih baru berada pada jarak maksimum dari kedua titik dan kemudian kita pilih benih keempat yang harus berada pada jarak maksimum dari ketiga titik data.

Kita baik-baik saja ketika berhadapan dengan 4 atau 5 klaster tetapi secara real time kita mungkin perlu berurusan dengan ratusan dimensi dan jutaan pengamatan, lalu bagaimana Anda akan memutuskan Jumlah klaster? Saya dapat memiliki semua titik data saya dalam satu klaster atau membuat klaster untuk setiap titik data. Di sini, kedua pendekatan tersebut tidak berguna. Sasaran di sini adalah menemukan jumlah klaster yang tepat. Ada dua cara untuk memikirkannya. Cara ini didasarkan pada masalah bisnis yang sedang kita hadapi. Jika saya memberikan tiga pilihan seperti apakah Anda ingin membuat 2, 5, atau 200 klaster? Keputusan ini didasarkan pada kebutuhan bisnis kita. Jika kita mengatakan bahwa kita dapat menangani 5 jenis keluhan karena kita memiliki sumber daya yang terbatas, menggunakan 200 klaster dan menangani semua keluhan secara terpisah tidak dapat dilakukan oleh kita karena biayanya sangat mahal, izinkan saya mengelompokkan 200 konsep tersebut ke dalam 5 klaster sehingga 5 adalah jumlah yang wajar. Kita dapat menggunakan aturan praktis, yaitu sebagian dari akar kuadrat data. Akar kuadrat dari jumlah titik data. Kita bahkan dapat memutuskan jumlah klaster yang tepat dengan keputusan yang didorong oleh pembelajaran mesin. Ada jumlah klaster yang tepat, untuk setiap kumpulan data tetapi pertanyaannya

adalah apakah kita dapat menemukannya atau tidak. Jumlah klaster yang tepat merupakan masalah penting karena jika melebihi jumlah tersebut akan menjadi gangguan, jika di bawah jumlah tersebut Anda tidak akan menangkap struktur apa pun (Sinyal). Kita perlu berpikir saat melakukan pengelompokan PCA atau K-Means, berapa banyak gangguan dan berapa banyak sinyal yang ada dalam data.

Kita dapat menggunakan statistik Gap untuk mengetahui jumlah klaster yang tepat, kita coba dengan rentang klaster, katakanlah dari 3 hingga 20 untuk setiap klaster, Anda menggunakan algoritma pengambilan sampel titik Farthest First atau beberapa inisialisasi acak, apa pun yang Anda lakukan, Anda akan mendapatkan beberapa statistik. Plot statistik ini dan hitung gap dan itu akan memberi tahu Anda jumlah klaster yang tepat. Kita mengikuti proses ini sepanjang waktu, Gap tidak boleh negatif karena yang diharapkan mengasumsikan gangguan acak sepenuhnya atau distribusi yang sepenuhnya seragam. Dengan mengasumsikan jumlah titik data yang didistribusikan secara seragam, itu adalah struktur terburuk yang dapat kita miliki, jadi itulah yang digunakan untuk menghitung yang diharapkan, yang akan selalu lebih rendah daripada yang diamati. Jadi kami berharap bahwa data akan memiliki beberapa struktur yang berada di luar distribusi data acak.

8.3 PENGELOMPOKAN HIRARKIS

Dalam jenis ini, kami memiliki dua pendekatan seperti 1. Atas ke bawah (Divisif) 2. Bawah ke atas (Aglomeratif). Dalam pendekatan atas ke bawah, kami mengambil seluruh set sebagai satu klaster tunggal dan membaginya menjadi dua klaster, kemudian dari kedua klaster kami membagi klaster lebih lanjut, dan seterusnya. Saat membuat pengelompokan di setiap langkah, kami mengambil pendekatan pengelompokan K-means dalam kasus data multivariat dan pengelompokan bulat dalam kasus data teks. Jika kami langsung membuat 4-5 klaster sekaligus, itu disebut pengelompokan terpartisi sedangkan pendekatan membaginya langkah demi langkah ini disebut pengelompokan divisif, seperti pertama-tama kami membuat dua klaster dari klaster akar dan kemudian membagi kedua klaster menjadi dua bagian masing-masing. Pendekatan ini disebut pengelompokan hierarkis atas ke bawah dan kami terus melakukannya dari bawah ke bawah dan seterusnya. Sekarang Anda mungkin bertanya, jika kita menggunakan pendekatan K-means dan pendekatan Top-Down, apakah kita dijamin akan mendapatkan jumlah klaster yang sama dengan entitas yang sama, pada akhirnya? Jawabannya adalah Tidak. Hirarki adalah cara alami di mana data muncul, bila memungkinkan kita harus berpikir secara hierarkis, kita selalu dapat menggunakan pengelompokan Partisi, di setiap level, tetapi pada akhirnya kita melakukan pengelompokan hierarkis.

Mari kita lihat jenis pengelompokan lainnya yaitu, pengelompokan hierarkis aglomeratif, ini sangat banyak digunakan dalam skenario berikut. Bayangkan ketika titik data tidak terlalu besar, itu berarti kita tidak memiliki terlalu banyak titik data tetapi kita menginginkan klaster yang benar-benar bagus, dalam hal ini, orang lebih suka menggunakan pengelompokan aglomeratif. Mari kita ambil contoh beberapa urutan gen, kita mengelompokkan gen bersama-sama dan mereka muncul dengan klaster. Bayangkan ada

beberapa titik data, saya ingin dapat mengelompokkannya dan mengaturnya, saya menggunakan pengelompokan Aglomeratif. Saya bisa mendapatkannya, baik dengan metode top down atau dengan menggunakan metode bottom-up. Jadi saya tidak akan memberi tahu Anda metode mana yang saya gunakan. Idenya adalah kita dapat mengatur data pada simpul daun, menjadi hierarki.

Mari kita ambil contoh angka 0-9 dan ingin menemukan titik data yang lebih mirip. Pertama, kita ambil dua angka yang terlihat sangat mirip. Ada gagasan yang sangat jelas bahwa 1 dan 7 terlihat mirip. Jadi, saya gabungkan keduanya. Jadi, saat saya menggabungkannya, saya mendapatkan klaster berukuran dua. Saya memiliki n-2 klaster berukuran 1. Saya dapat memperlakukan setiap titik data sebagai klaster. Awalnya, saat saya tidak melakukan apa pun, setiap titik data adalah klaster. Setiap kali kita berpikir tentang pengelompokan, kita dapat memikirkan berapa jumlah klaster terkecil yang dapat kita miliki dan berapa jumlah klaster terbesar yang dapat kita miliki? Jadi, angka terkecil bisa jadi 1 dan angka terbesar bisa jadi jumlah titik data. Klaster aglomeratif membantu kita membangun klaster dari n hingga satu dengan menggabungkan klaster dari bawah ke atas. Anda akan mendapatkan spektrum semua klaster yang mungkin antara 1 dan N.

8.4 MENILAI KECENDERUNGAN KLUSTER

Sebelum menerapkan metode pengelompokan apa pun pada dataset, pertanyaan yang muncul adalah, Apakah dataset tersebut berisi klaster yang melekat?. Masalah besar dalam pembelajaran mesin tanpa pengawasan adalah bahwa metode pengelompokan akan mengembalikan klaster meskipun data tidak berisi klaster apa pun. Dengan kata lain, jika Anda menerapkan analisis pengelompokan secara membabi buta pada dataset, data tersebut akan terbagi menjadi klaster. Oleh karena itu, sebelum memilih pendekatan pengelompokan, kita perlu memutuskan apakah dataset tersebut berisi klaster yang bermakna atau tidak. Jika ya, berapa banyak klaster yang ada. Proses ini didefinisikan sebagai penilaian kecenderungan pengelompokan atau kelayakan analisis pengelompokan. Kita dapat menggunakan metode statistik dan visual untuk menilai kecenderungan pengelompokan. Dapat dilihat bahwa, algoritma k-means dan pengelompokan hierarkis memaksakan klasifikasi pada dataset yang terdistribusi secara seragam secara acak meskipun tidak ada klaster yang bermakna di dalamnya. Metode penilaian kecenderungan pengelompokan digunakan untuk menghindari masalah ini. Metode untuk menilai kecenderungan pengelompokan: Penilaian kecenderungan pengelompokan menentukan apakah kumpulan data tertentu berisi kluster yang bermakna. Kita dapat menggunakan dua metode untuk menentukan kecenderungan pengelompokan: 1) statistik (statistik Hopkins) dan 2) metode visual (algoritma Visual Assessment of Cluster Tendency (VAT)). Statistik Hopkins: Statistik Hopkins digunakan untuk menilai kecenderungan pengelompokan kumpulan data dengan mengukur probabilitas bahwa kumpulan data tertentu dihasilkan oleh distribusi data yang seragam. Dengan kata lain, statistik ini menguji keacakan spasial data. Nilai H sekitar 0,5 berarti data tersebut saling berdekatan, sehingga data D terdistribusi secara seragam. Hipotesis nol dan hipotesis alternatif didefinisikan sebagai berikut. Hipotesis nol adalah kumpulan data D terdistribusi secara seragam (yaitu, tidak ada

kluster yang bermakna) dan hipotesis alternatif adalah kumpulan data D tidak terdistribusi secara seragam (yaitu, berisi kluster yang bermakna). Jika nilai statistik Hopkins mendekati nol, maka kita dapat menolak hipotesis nol dan menyimpulkan bahwa kumpulan data D secara signifikan merupakan data yang dapat dikelompokkan. Fungsi R untuk menghitung statistik Hopkins: Fungsi hopkins() dapat digunakan untuk mengevaluasi kecenderungan pengelompokan secara statistik dalam R.

#1. Baca file csv untuk memuat data

```
setwd("D:/R data")
Clus_ds <- read.csv("Clus_sample.csv", header=TRUE) head(Clus_ds)
str(Clus_ds) fix(Clus_ds)
```

#2. Cari tahu kemampuan pengelompokan dengan Metode visual: Buat diagram sebar

```
plot(CAD~Neuro,Clus_ds)
```

Hitung statistik Hopkins untuk kumpulan data

```
set.seed(123)
hopkins(Clus_stan, n = nrow(Clus_stan)-1)
```

#Jika nilai H jauh di bawah ambang batas 0,5, maka kumpulan data tersebut sangat mampu dikluster.

8.5 MENENTUKAN JUMLAH KLUSTER

```
setwd("D:/R data")
Clus_ds <- read.csv("Cluster.csv", header=TRUE) Clus_req <- Clus_ds[-
c(1,8)]
Clus_stan <- scale(Clus_req)
wssplot <- function(Clus_stan, nc=5, seed=1234){
wss <- (nrow(Clus_stan)-1)*sum(apply(Clus_stan,2,var)) for (i in 2:nc){
set.seed(seed)
wss[i] <- sum(kmeans(Clus_stan, centers=i)$withinss)} plot(1:nc, wss,
type="b", xlab="Jumlah Kluster", ylab="Jumlah dalam grup squares")}
```

#Parameter data adalah kumpulan data numerik yang akan dianalisis, nc adalah jumlah maksimum klaster yang akan dipertimbangkan, dan seed adalah seed angka acak.

```
wssplot(Clus_stan) library(NbClust) set.seed(1234)
nc <- NbClust(Clus_stan, min.nc=2, max.nc=5, method="kmeans")
table(nc$Best.n[1,])
barplot(table(nc$Best.n[1,]),
xlab="Jumlah Klaster", ylab="Kriteria", main="Pilih Jumlah
Klaster dengan 26 Kriteria")
```

8.6 STUDI KASUS

Sebuah Rantai rumah sakit multispesialis yang menangani spesialisasi Kardiologi, Endokrinologi, Psikiatri, dan Urologi ingin membuat beberapa strategi khusus untuk Meningkatkan Bisnis mereka dan memanfaatkan sumber daya mereka secara optimal.

Sekarang tugas Anda adalah membuat Cluster dan menyarankan strategi apa yang harus diikuti di setiap cluster.

#Langkah 1: Instal dan Muat Paket yang diperlukan

```
if(!require(clustertend)) install.packages("clustertend")
library(clustertend)
```

#Langkah 2: Muat data

```
setwd("D:/R data")
Clus_ds <- read.csv("Cluster.csv", header=TRUE)
```

#Langkah 3: Jelajahi data

```
fix(Clus_ds) head(Clus_ds) str(Clus_ds)
```

#Langkah 4: Hapus Variabel kategoris seperti Id dan State

```
Clus_req <- Clus_ds[-c(1,8)]
```

#Langkah 5: Penghapusan secara berurutan

```
Clus_req <- na.omit(Clus_req) summary(Clus_req)
str(Clus_req)
```

#Langkah 6: Lakukan Normalisasi

```
Clus_stan <- scale(Clus_req) # standarisasi variabel
head(Clus_stan)
summary(Clus_stan)
```

 # Langkah 7: Tentukan jumlah klaster

```
library(NbClust) set.seed(1234)
nc <- NbClust(Clus_stan, min.nc=2, max.nc=5, method="kmeans")
table(nc$Best.n[1,])
barplot(table(nc$Best.n[1,]),
xlab="Jumlah Klaster", ylab="Kriteria", main="Pilih Jumlah
Klaster dengan 26 Kriteria")
```

#Langkah 8: Hitung jarak: Pengelompokan Hirarkis Ward

```
dt <- dist(Clus_stan, method = "euclidean")
# matriks jarak fit.hc <- hclust(dt, method="ward.D2")
```

#Langkah 9: Buat dendrogram klaster dengan metode keterkaitan lengkap

```
plot(fit.hc)
```
#display dendrogram

#Langkah 10: Potong pohon menjadi 4 klaster

```
groups <- cutree(fit.hc, k=4)
```

#Langkah 11: Buat dendrogram klaster dengan batas merah di sekitar 4 klaster

```
rect.hclust(fit.hc, k=4, border="red")
```

#Langkah 12: Lakukan Analisis Klaster K-Means

```
set.seed(1234)
fit.km <- kmeans(Clus_stan, 4, nstart=25)
```

#Langkah 13: Tentukan jumlah Titik Data di setiap klaster

```
fit.km$size
```

#Langkah 14: Tampilkan pusat

```
fit.km$centers
```

#Langkah 15: Tampilkan klaster

```
fit.km$cluster
```

#Langkah 16: Dapatkan rata-rata klaster

```
aggregate(Clus_stan,by=list(fit.km$cluster),FUN=mean)
```

#Langkah 17: Tambahkan penugasan klaster ke yang asli dataset

```
Clus_Fin <- data.frame(Clus_ds,fit.km$cluster) fix(Clus_Fin)
```

#Langkah 18: Buat Plot Klaster

```
library(cluster)
clusplot(Clus_Fin, fit.km$cluster, color=TRUE, shade=TRUE,
labels=4, lines=0)
```

BAB 9
ANALISIS KERANJANG BELANJA

9.1 PENDAHULUAN

Analisis Keranjang Belanja memiliki beberapa nama seperti Penemuan Pola/Penambangan Pola/Aturan Asosiasi/Analisis Keranjang Belanja/Penambangan Kumpulan Barang. Analisis ini memberikan wawasan tentang produk mana yang cenderung dibeli bersamaan dan mana yang paling mudah dipromosikan. Pembelajaran aturan asosiasi adalah teknik Penambangan Data yang populer untuk menemukan hubungan yang menarik antara variabel dalam basis data yang besar.

Apa itu Penemuan Pola?

Pola: Seperangkat item, subsekuens, atau substruktur yang sering muncul bersamaan dalam satu kumpulan data. Pola mewakili properti intrinsik dan penting dari kumpulan data. Penemuan pola adalah Mengungkap pola dari kumpulan data yang sangat besar. Penemuan Pola digunakan untuk menemukan keteraturan yang melekat dalam satu kumpulan data dan dapat bertindak sebagai dasar untuk banyak tugas penambangan data penting seperti Asosiasi, Korelasi, Analisis Deret Waktu, Analisis Kausalitas, Analisis Klaster, Penambangan pola sekuensial dan Struktural. Apa itu pola?: Pola adalah sekumpulan item yang sering muncul bersamaan dalam satu set data, yang merepresentasikan properti penting dari set data.

Kita dapat menggunakan aturan asosiasi untuk menjawab pertanyaan semacam ini:

Produk apa yang sering dibeli bersamaan?
Apa pembelian berikutnya setelah membeli Produk Tertentu?
Urutan kata apa yang mungkin membentuk frasa dalam korpus ini?
Apa saja item yang sering dibeli pelanggan ini?
Berapa jumlah rata-rata item per pesanan?
Apa item yang paling umum ditemukan dalam pesanan satu item?
Karena ia membeli Laptop, kapan ia akan membeli printer?
Karena ia membeli iPhone6, apakah ia akan tertarik pada iPhone7?
Situs mana yang ia kunjungi, lalu ke mana ia pindah, berapa lama ia menghabiskan waktu di sana?
Penyakit apa yang lebih umum pada individu dengan urutan gen spesifik ini?
Pelanggan yang telah membeli produk ini, produk lain apa yang cenderung mereka beli?
Berapa jumlah rata-rata pesanan per pelanggan? Berapa jumlah rata-rata item unik per pesanan?
Kombinasi klaim asuransi yang tidak biasa dapat menjadi tanda penipuan. Bagaimana saya dapat menemukannya?
Riwayat pasien medis dapat memberikan indikasi komplikasi yang mungkin terjadi berdasarkan kombinasi perawatan tertentu. Seberapa sering kejadian buruk ini terjadi?

Tujuan dari aturan asosiasi adalah untuk melihat hubungan yang menarik di antara item-item. Setiap aturan yang tidak terungkap berbentuk X → Y, yang berarti bahwa ketika item X diamati, item Y juga diamati. Dalam kasus ini, sisi kiri (LHS) dari aturan tersebut adalah X, dan sisi kanan (RHS) dari aturan tersebut adalah Y.

Mari kita ambil daftar beberapa transaksi yang diberikan dalam berkas Transact.txt dari sebuah toko kelontong yang berdekatan dengan sebuah Pusat kebugaran.

Kita mungkin ingin menjawab pertanyaan-pertanyaan berikut:

Apa dua item yang lebih mungkin dibeli bersama daripada dua item lainnya?

Produk mana yang tidak pernah dibeli dengan Jam?

Pertanyaan-pertanyaan tersebut dapat dijawab dengan mengamati data secara manual.

Sekarang masalah sebenarnya adalah Bagaimana kita menghasilkan aturan-aturan ini secara otomatis pada data yang besar?

9.2 TERMINOLOGI PENEMUAN POLA

Itemset: Setiap transaksi yang berisi satu atau beberapa item. Ini juga dikenal sebagai Itemset. Istilah itemset mengacu pada kumpulan item atau entitas individual yang berisi beberapa jenis hubungan. Ini bisa berupa sekumpulan item yang dibeli bersama dalam satu transaksi, sekumpulan profil yang dicari di LinkedIn dalam satu sesi atau sekumpulan hyperlink yang diklik oleh satu pengguna dalam sesi tertentu.

K-Itemset: Sebuah itemset yang berisi k item disebut set k-item. Kita menggunakan kurung kurawal seperti {item 1,item 2,… item k} untuk menunjukkan set k-item.

Support: Salah satu komponen utama dari aturan Asosiasi adalah dukungan. Diberikan itemset X, support X adalah persentase transaksi yang berisi X.

Dukungan absolut (jumlah) X:

Frekuensi atau jumlah kemunculan itemset X.

Dukungan relatif, s:

Fraksi transaksi yang berisi X (yaitu, probabilitas bahwa transaksi berisi X)

Untuk 1 Itemset: Support= Frekuensi(X)/N

Untuk 2 Itemset: Support= Frekuensi(X,Y)/N

Dalam contoh yang diberikan, Jika, Roti muncul 4 kali dalam 5 transaksi, artinya, 4/5 (80%) dari semua transaksi berisi itemset {Roti}, maka support {roti} adalah 0,8. Demikian pula, Jika 60% dari semua transaksi berisi itemset {Roti, Susu}, maka support {Roti, Susu} adalah 0,6.

Sebuah itemset X disebut frequent jika support X tidak kurang dari ambang batas minsup (dilambangkan sebagai σ). Sebuah itemset frequent memiliki item yang muncul bersama-sama lebih dari kriteria support minimum. Jika support minimum ditetapkan pada 0,5, setiap itemset dapat dianggap sebagai frequent itemset jika support dari frequent itemset harus lebih besar dari atau sama dengan support minimum. Dalam contoh kita, {Bread} dan {Bread,Milk} dianggap frequent itemset pada support minimum 0,5.

Keyakinan: Keyakinan didefinisikan sebagai ukuran kepastian atau kesetiaan yang terkait dengan setiap aturan yang ditemukan. Secara matematis, keyakinan adalah persentase transaksi yang berisi X dan Y dari semua transaksi yang berisi X.
Keyakinan, c: Probabilitas bersyarat bahwa transaksi yang berisi X juga berisi Y.
Keyakinan = Frekuensi(X,Y) / Frekuensi (X)

Misalnya, jika {Roti, Pisang, Susu} memiliki dukungan 0,20 dan {Roti, Pisang} juga memiliki dukungan 0,20, keyakinan aturan {Roti,Pisang}→{Susu} adalah 1, yang berarti 100% dari waktu pelanggan membeli Roti dan Pisang, juga membeli Susu. Oleh karena itu, aturan tersebut benar untuk 100% transaksi yang berisi Roti dan Pisang. Suatu hubungan menjadi menarik ketika algoritme mengidentifikasi hubungan dengan ukuran keyakinan yang lebih besar dari atau sama dengan keyakinan minimum. terdapat kendala untuk keyakinan, karena dapat mengidentifikasi aturan yang menarik dari semua aturan kandidat, hanya mempertimbangkan anteseden (X) dan kemunculan bersama X dan Y; tidak mempertimbangkan konsekuensi aturan (Y). Jadi, keyakinan tidak dapat menentukan apakah suatu aturan mengandung implikasi sebenarnya dari hubungan tersebut atau apakah aturan tersebut murni kebetulan. Terkadang X dan Y dapat secara statistik independen tetapi tetap menerima skor keyakinan yang tinggi. Lift dapat mengatasi masalah ini. Lift: Lift mengukur seberapa sering X dan Y muncul bersamaan daripada yang diharapkan jika keduanya secara statistik independen satu sama lain. Lift adalah ukuran seberapa banyak X dan Y benar-benar terkait daripada terjadi bersamaan secara kebetulan.

$$\text{Lift} = \frac{\text{Support}(X, Y)}{\text{Support}(X) \times \text{Support}(Y)}$$

Lift bernilai 1 jika X dan Y secara statistik independen satu sama lain. Sebaliknya, lift X → Y yang lebih besar dari 1 menunjukkan bahwa ada beberapa nilai pada aturan tersebut. Nilai lift yang lebih besar menunjukkan efektivitas yang lebih besar dari asosiasi antara X dan Y.

Dengan asumsi 10 transaksi, dengan {Roti, Mentega} muncul dalam 4 transaksi di antaranya, {Roti} muncul dalam 4 transaksi, dan {Mentega} muncul dalam 5 transaksi, maka Lift (Roti → Mentega) = 0,4/(0,4*0,5) = 2,0

Jika {Mentega, Sereal} muncul dalam 3 transaksi di antaranya, {Sereal} muncul dalam 4 transaksi, dan {Mentega} muncul dalam 5 transaksi, maka Lift (Mentega → Sereal) = 0,3/(0,5*0,4) = 1,5. Dengan mengamati hal ini, kita dapat mengatakan bahwa Roti dan Mentega memiliki asosiasi yang lebih kuat daripada Sereal dan Mentega.

9.3 ALGORITMA APRIORI

Algoritma Apriori menggunakan pendekatan iteratif untuk mengungkap itemset yang sering muncul dengan terlebih dahulu menentukan semua kemungkinan 1-itemset, misalnya, {Roti}, {Sereal}, {Mentega} dan mengidentifikasi mana di antara mereka yang sering muncul. Mari kita ambil kriteria dukungan minimum yang ditetapkan pada 0,5, algoritma mengidentifikasi dan mempertahankan itemset yang muncul dalam setidaknya 50% dari

semua transaksi dan menolak itemset yang memiliki dukungan kurang dari 0,5, proses ini disebut Pemangkasan.

Prinsip pemangkasan apriori: Jika ada itemset yang jarang muncul, superset-nya bahkan tidak boleh dibuat. Pada iterasi berikutnya, 1-itemset yang sering muncul yang diidentifikasi dipasangkan menjadi 2-itemset ({Roti, sereal}, {Roti, Mentega}, …) dan dievaluasi lagi untuk menemukan 2-itemset yang sering muncul di antara mereka. Ini adalah proses iteratif, pada setiap iterasi, algoritma memeriksa apakah kriteria dukungan dapat dipenuhi atau tidak. Jika kriteria tersebut terpenuhi, algoritme akan mengembangkan itemset, mengulang proses tersebut hingga tidak ada lagi dukungan atau hingga itemset mencapai panjang yang telah ditentukan sebelumnya.

Asumsi Apriori: Misalkan {Roti, Mentega} sering muncul. Karena setiap kemunculan Roti, Mentega mencakup Roti dan Mentega, maka Roti dan Mentega juga harus sering muncul. Jadi, jika k-itemset sering muncul, semua subsetnya (k-1, k-2 itemset) juga sering muncul.

Langkah-langkah dalam pelaksanaan algoritma Apriori:

Garis Besar Apriori (berdasarkan level, pembuatan kandidat dan pengujian)
Awalnya, pindai DB sekali untuk mendapatkan 1-itemset yang sering
Ulangi
Hasilkan itemset kandidat dengan panjang (k+1) dari itemset yang sering dengan panjang k
Uji kandidat terhadap DB untuk menemukan itemset yang sering (k+1)
Tetapkan k := k +1
Hingga tidak ada set yang sering atau kandidat yang dapat dihasilkan
Kembalikan semua itemset yang sering diturunkan

Keuntungan Algoritma Apriori:

1. Menggunakan properti itemset yang besar.
2. Mudah diparalelkan.
3. Mudah diimplementasikan.

Kekurangan Algoritma Apriori:

Menganggap basis data transaksi berada di memori. Memerlukan hingga m pemindaian basis data. Algoritma Apriori bisa sangat lambat dan hambatannya adalah pembuatan kandidat. Misalnya, jika DB transaksi memiliki 104 frequent 1-itemset, mereka akan menghasilkan 107 kandidat 2-itemset bahkan setelah menggunakan downward closure. Untuk menghitungnya dengan sup lebih dari minsup, basis data perlu dipindai di setiap level. Ia memerlukan (n +1) pemindaian, di mana n adalah panjang pola terpanjang.

Metode untuk Meningkatkan Efisiensi Apriori:

1. Penghitungan itemset berbasis hash: K-itemset yang jumlah bucket hashing-nya di bawah ambang batas tidak dapat sering muncul
2. Pengurangan transaksi: Transaksi yang tidak berisi k-itemset yang sering muncul tidak berguna dalam pemindaian berikutnya

3. Pemartisian: Setiap itemset yang berpotensi sering muncul dalam DB harus sering muncul setidaknya di salah satu partisi DB
4. Pengambilan sampel: penambangan pada subset data yang diberikan, ambang batas dukungan yang lebih rendah + metode untuk menentukan kelengkapan
5. Penghitungan itemset dinamis: tambahkan itemset kandidat baru hanya jika semua subsetnya diperkirakan sering muncul

Ada banyak aplikasi Aturan Asosiasi yang meliputi:

- Analisis keranjang belanja
- Penempatan produk secara fisik atau logis dalam kategori produk terkait,
- Pemasaran silang, (Penjualan Silang, Penjualan Naik)
- Desain Katalog,
- Analisis kampanye penjualan,
- Program promosi,
- Analisis aliran klik atau analisis log Web,
- Analisis urutan biologis,
- kartu loyalitas program
- sistem rekomendasi seperti Amazon, Facebook, LinkedIn, dan Netflix.

9.4 STUDI KASUS

Mari kita ambil data transaksi di dekat pusat kebugaran pada dini hari. Mari kita pahami apa saja pembelian yang sering dilakukan oleh pelari atau pejalan kaki dan temukan hubungan di antara keduanya sehingga pemilik toko ingin menyediakan barang-barang yang dibutuhkan untuk pelanggannya.

Langkah 1: Muat set data

```
setwd("D:/R data")
basket <-read.table("Transact.txt", header=TRUE, sep="\t")
```

Langkah 2: Lihat 10 observasi pertama

```
head(basket, n=10)
```

Langkah 3: Pahami struktur dan bagian deskriptor data

```
str(basket)
```

Langkah 4: Ubah variabel numerik menjadi faktor jika perlu

```
fac <- c(1,2,4)
basket[,fac] <- lapply(basket[,fac],factor) str(basket)
```

Langkah 5: Pisahkan data

```
dt <- split(basket$Products, basket$ID)
```

Langkah 6: Muat paket dan pustaka yang diperlukan

```
if(!require(arules)) install.packages("arules")
```

Kita sekarang siap untuk menambang beberapa aturan. Anda harus selalu melewati dukungan dan keyakinan minimum yang diperlukan. Asumsikan bahwa Kami ingin menetapkan dukungan minimum ke 0,3 dan keyakinan minimum 0,8. Kami ingin menunjukkan 5 aturan teratas

Langkah 7: Dapatkan Aturan

```
rules <- apriori(basket, parameter = list(supp = 0,3, conf = 0,8))
```

Langkah 8: Ubah data ke tingkat transaksi

```
dt2 = as(dt,"transactions") summary(dt2)
inspect(dt2)
```

Langkah 9: Temukan Item yang Paling Sering

```
itemFrequency(dt2, type = "relative")
itemFrequencyPlot(dt2,topN = 5)
```

Langkah 10: Gabungkan dan ringkas aturan:

Jika aturan terlalu panjang. Kita dapat meringkas aturan dengan menambahkan parameter "maxlen" ke fungsi Apriori Anda:

```
rules = apriori(dt2, parameter=list(support=0.3, confidence=0.8))
rules = apriori(dt2, parameter=list(support=0.3, confidence=0.8,
minlen = 3)) rules = apriori(dt2, parameter=list(support=0.3,
confidence=0.8, maxlen = 4))
```

Langkah 11: Ubah aturan menjadi kerangka data

```
rules3 = as(rules, "data.frame")
write(rules, "D:\\rules.csv", sep=",")
```

Langkah 12: Tampilkan hanya aturan produk tertentu

```
inspect( subset( rules, subset = rhs %pin% "Bread" ))
```

Langkah 13: Tampilkan 10 aturan teratas

```
options(digits=2)
inspect(rules[1:10])
```

Langkah 14: Dapatkan Informasi Ringkasan

```
summary(rules)
```

Langkah 15: Urutkan aturan sesuai keinginan kita, tulis aturan yang paling relevan terlebih dahulu berdasarkan confidence atau lift

```
rules<-sort(rules, by="confidence", decreasing=TRUE) rules<-
sort(rules, by="lift", decreasing=TRUE)
```

Langkah 16: Hapus Aturan yang Tidak Diperlukan

Terkadang, aturan akan berulang. Sebagai analis, Anda dapat memilih untuk menghapus item dari kumpulan data. Atau, Anda dapat menghapus aturan yang berulang yang dihasilkan.

```
subset.matrix <- is.subset(rules, rules)
subset.matrix[lower.tri(subset.matrix, diag=T)] <- NA redundant
<- colSums(subset.matrix, na.rm=T) >= 1 which(redundant)
rules.pruned <- rules[!redundant] rules<-rules.pruned
```

Langkah 17: Bersihkan Aturan

```
rules3$rules=gsub("\\{", "", rules3$rules)
rules3$rules=gsub("\\}", "", rules3$rules)
rules3$rules=gsub("\"", "", rules3$rules)
```

Langkah 18: Pisahkan Aturan

```
library(splitstackshape)
Rules4=cSplit(rules3, "rules","=>")
names(Rules4)[names(Rules4) == 'rules_1'] <- 'LHS'
Rules5=cSplit(Rules4, "LHS",",")
Rules6=subset(Rules5,            select=            -c(rules_2))
names(Rules6)[names(Rules6) == 'rules_3'] <- 'RHS'
```

Langkah 19: Menargetkan Item:

Sekarang setelah kita tahu cara membuat aturan, batasi outputnya, katakanlah kita ingin menargetkan item untuk membuat aturan. Ada dua jenis target yang mungkin menarik bagi kita yang diilustrasikan dengan contoh "Roti":

Apa yang kemungkinan akan dibeli pelanggan sebelum mereka membeli "Roti"

```
rules<-apriori(data=dt, parameter=list(supp=0.5,conf = 0.8),
appearance = list(default="lhs",rhs="Roti"), control =
list(verbose=F)) rules<-sort(rules,
decreasing=TRUE,by="confiability")
inspect(rules[1:4])
```

Apa yang kemungkinan akan dibeli pelanggan jika mereka membeli "Roti"

```
rules<-apriori(data=dt, parameter=list(supp=0.5,conf = 0.7),
appearance = list(default="rhs",lhs="Roti"), control =
list(verbose=F)) rules<-sort(rules,
decreasing=TRUE,by="confidential")
inspect(rules[1:3])
```

Langkah 20: Visualisasi:

Terakhir, kita ingin memetakan aturan dalam grafik. Untuk ini, kita memerlukan paket "arulesViz". #Instal paket arulesViz

```
library(arulesViz)
plot(rules,method="graph",interactive=TRUE,shading=NA)
```

BAB 10
ESTIMASI KEPADATAN KERNEL

10.1 PENDAHULUAN

Saat kita melihat data, hal pertama yang ingin kita cari tahu adalah struktur datanya. Untuk itu, kita membuat histogram data berdasarkan satu atau beberapa fitur. Fitur juga dikenal sebagai Dimensi, Variabel, Kolom, jadi kita menggunakan istilah-istilah ini secara bergantian. Histogram mengungkapkan banyak hal, seperti titik data mana yang lebih menonjol dan mana yang kurang. Kita dapat menyebutnya sebagai Fungsi Kepadatan Probabilitas karena hanya dengan melihat histogram, kita dapat mengetahui berapa probabilitas titik data tertentu muncul. Ini adalah estimasi Kepadatan satu dimensi. Namun, Histogram satu dimensi saja tidak cukup. Yang benar-benar kita inginkan adalah estimasi kepadatan gabungan.

Apakah Anda mendengar cerita tentang lima orang buta yang menyentuh Gajah dan mendeskripsikannya? Apa yang mereka lakukan di sana? Setiap orang buta dalam cerita tersebut menjelaskan satu dimensi gajah dan akhirnya kita bisa mendapatkan kesimpulan karena tidak satu pun dari mereka yang mendeskripsikan gajah secara lengkap. Ketika kita melihat satu dimensi, kita mungkin mendapat kesan yang mungkin tidak benar tentang data tersebut. Jika kita melihat data yang sama dari dimensi lain, saya mungkin mendapat kesan yang sama sekali berbeda. Sekarang distribusi sebenarnya dapat berupa kombinasi keduanya di ruang gabungan. Tujuan estimasi Kepadatan adalah untuk mengetahui Distribusi Gabungan. Biarkan saya memberi Anda satu contoh lagi untuk memahami estimasi kepadatan. Mari kita ambil kartu kredit kita sebagai contoh dan kita memiliki penggunaan normal kartu kredit selama bertahun-tahun. Jika seseorang mencuri kartu kredit Anda, ia mungkin menggunakan kartu tersebut sedikit berbeda karena ia tidak tahu bagaimana kita menggunakan kartu tersebut dan di mana kita lebih sering menggunakan kartu tersebut. Sekarang Bayangkan Bagaimana perusahaan kartu kredit mendeteksi perilaku yang sedikit tidak biasa atau sedikit tidak normal ini? Bagaimana kita menangkapnya? Apa yang dilakukan model tersebut?. Biarkan saya mengambil satu contoh lagi, Kami diberi iklan di Google untuk kata kunci tertentu, Biasanya orang mungkin mengeklik iklan itu pada tingkat tertentu berdasarkan hari tertentu dan jumlah kueri dan berapa kali iklan itu muncul, dll. Berdasarkan ini, kami mendapatkan sekitar sejumlah klik. Mari kita bayangkan saya punya pesaing. Pesaing saya ingin uang saya terbuang sia-sia. Yang mungkin dilakukannya adalah, ia dapat membuat robot dengan menulis program yang terus mengeklik iklan berkali-kali, setiap kali ada pertanyaan yang diajukan pada poin tertentu. Sekarang, apa yang Anda harapkan dari Google? Anda berharap Google mengetahui perilaku abnormal ini, dan dari lokasi mana klik terjadi. Di sinilah tepatnya estimasi Kernel Density berguna.

Prinsip Data yang sangat penting pertama adalah Data memiliki Hirarki. Hal kedua adalah data memiliki Struktur dan Noise. Data akan selalu memiliki Struktur dan Noise. Tujuan kami dalam Pembelajaran Tanpa Pengawasan (Estimasi Kepadatan Kernel) adalah untuk

mengetahui bagian mana dari data yang memiliki struktur dan bagian mana dari data yang memiliki Noise. Ketika kami melakukan PCA, kami mempertahankan beberapa komponen Principal dan kami membuang beberapa komponen lainnya karena kami merasa bahwa beberapa komponen memiliki struktur di dalamnya dan yang lainnya memiliki Noise. Bagaimana kami memutuskan berapa banyak komponen principal yang perlu kami pertahankan? Bagaimana kami memutuskan berapa banyak cluster yang perlu kami buat? Itulah panggilan kami dan itulah keindahan ilmu Data. Di sini tujuan kami adalah untuk mengetahui cara mengkomplekskan model yang Anda inginkan sehingga menangkap sinyal maksimum dan bukan noise. Jika kita memutuskan bahwa hanya ada satu perilaku pengguna kartu kredit, kita gagal menangkap struktur dalam perilaku kartu kredit, karena ada berbagai jenis orang dengan berbagai jenis perilaku kartu kredit seperti bagaimana remaja menggunakan kartu dan bagaimana ibu rumah tangga menggunakan kartu dan bagaimana seorang pelajar menggunakan kartu, dll. Berdasarkan domain yang Anda hadapi, Anda perlu memutuskan struktur dan rasio gangguan dan berapa banyak data atau gangguan yang dapat kita harapkan di sini? Mari saya ambil contoh sederhana untuk ini, jika kita berurusan dengan data teks, jika kita melihat sekelompok makalah ilmiah, kita dapat mengharapkan lebih banyak data dan lebih sedikit gangguan di sana karena makalah tersebut ditulis dengan baik dan tidak akan ada kesalahan ejaan dan tata bahasa. Tetapi jika kita berurusan dengan data twitter, kita dapat memiliki banyak kesalahan ejaan. Berdasarkan ini, kita perlu membuat panggilan tentang berapa banyak data yang perlu kita tangkap dan seberapa rumit model saya. Gangguan dapat bersifat alami dan gangguan dapat disengaja. Dalam skenario kartu kredit, penipuan adalah gangguan yang disengaja dan di twitter, itu adalah gangguan yang tidak disengaja.

10.2 APA ITU KERNEL?

Kernel adalah jenis khusus fungsi kepadatan probabilitas (PDF) dengan properti tambahan yang harus bernilai non-negatif, genap, dan riil.

Apa itu Estimasi Kepadatan Kernel?: Estimasi kepadatan kernel adalah metode non-parametrik untuk memperkirakan fungsi kepadatan probabilitas (PDF) dari variabel acak kontinu. Estimasi ini bersifat non-parametrik karena tidak mengasumsikan distribusi yang mendasari variabel tersebut. Pendekatan estimasi kepadatan kernel mengatasi diskretitas pendekatan histogram dengan memusatkan fungsi kernel halus pada setiap titik data lalu menjumlahkannya untuk mendapatkan estimasi kepadatan. Pada dasarnya, pada setiap titik data, fungsi kernel dibuat dengan titik data di tengahnya, ini memastikan bahwa kernel simetris terhadap data. PDF kemudian diestimasi sebagai rata-rata titik data yang diamati untuk membuat perkiraan halus, data untuk memastikan bahwa PDF memenuhi properti PDF berikut. Setiap nilai PDF yang mungkin, yaitu fungsi, adalah non-negatif. Integral pasti PDF atas set pendukungnya sama dengan 1.

Perkiraan Kepadatan Kernel adalah jumlah tonjolan, yang ditetapkan untuk setiap titik data, dan ukuran tonjolan tersebut mewakili probabilitas yang ditetapkan untuk lingkungan nilai di sekitar titik data tersebut. Setiap tonjolan berpusat pada titik data, dan menyebar

secara simetris untuk menutupi nilai-nilai tetangga titik data tersebut. Setiap kernel memiliki lebar pita, dan menentukan lebar tonjolan. Lebar pita yang lebih besar menghasilkan tonjolan yang lebih pendek dan lebih lebar yang menyebar lebih jauh dari pusat dan menetapkan lebih banyak probabilitas untuk nilai-nilai tetangga.

10.3 LANGKAH DALAM MEMBUAT ESTIMASI KEPADATAN KERNEL

1. Pilih kernel, yang umum seperti normal (Gaussian), seragam (persegi panjang).
2. Pada setiap titik data, buat fungsi kernel berskala di mana K() adalah fungsi kernel yang Anda pilih. Parameter h disebut bandwidth, lebar jendela, atau parameter penghalusan.
3. Tambahkan semua fungsi kernel berskala individual dan bagi dengan n, ini menempatkan probabilitas 1/n untuk setiap Xi. Ini juga memastikan bahwa estimasi kepadatan kernel terintegrasi ke 1 atas set pendukungnya.

Memilih Bandwidth:

Memilih bandwidth adalah langkah tersulit dalam membuat estimasi kepadatan kernel yang baik yang menangkap distribusi variabel yang mendasarinya. Kita dapat mengikuti aturan sederhana seperti:

1. H kecil menghasilkan deviasi standar kecil, dan kernel menempatkan sebagian besar probabilitas pada titik data. Kita Menggunakan ini ketika ukuran sampel besar dan data dikemas rapat.
2. H yang besar menghasilkan simpangan baku yang besar, dan kernel menyebarkan lebih banyak probabilitas dari titik data ke nilai-nilai tetangganya. Kita menggunakan ini ketika ukuran sampel kecil dan datanya jarang. Fungsi density() di R menghitung nilai estimasi kepadatan kernel. Menerapkan fungsi plot() ke objek yang dibuat oleh density() akan memplot estimasi. Menerapkan fungsi summary() ke objek akan mengungkapkan statistik yang berguna tentang estimasi. Kita menggunakan metode ini untuk mengetahui struktur dan tata bahasa dalam data. Jika kita mendengarkan sepotong musik, kita dapat mengatakan bahwa itu dari Rahman atau Ilaya raja. Bagaimana kita bisa melakukan ini? Otak kita telah menyimpan musik dan menemukan struktur dalam urutan tertentu. Ngomong-ngomong, kita melakukan banyak estimasi kepadatan di otak kita.

Mari kita asumsikan bahwa semua mobil di dunia memiliki warna dan bentuk yang sama, kita melihat banyak data, mencoba untuk mengompresnya menjadi sebuah model. Kita tidak dapat mengingat semua mobil tetapi kita masih dapat mengenali mobil baru karena kita melihat banyak data, kita mengenali seperti apa mobil itu dan kemudian ketika saya melihat data lain atau objek baru saya dapat mengatakan ini adalah mobil. Apa yang terjadi? Apa yang dilakukan otak Anda? Anda melihat banyak data, otak Anda membuat model, dan setiap kali Anda melihat titik data baru, Anda dapat memahami bahwa itu adalah mobil. Untuk ini model Anda sudah cukup setiap kali Anda tidak memerlukan data.

Kami menggunakan teknik KDE bahkan untuk menemukan outlier. Serta untuk mengetahui berapa banyak sinyal dan noise yang ada, oleh karena itu kami dapat menentukan

seberapa rumit modelnya. Semakin rumit modelnya, kami mulai menangkap noise, jika modelnya terlalu sederhana, kami tidak dapat menangkap strukturnya. Jadi, kami perlu memiliki kompleksitas model yang tepat. Estimasi kepadatan adalah yang memiliki banyak hal psikologis yang kami lakukan. Mari kita bayangkan situasi di mana Anda ingin memberikan hadiah Unik kepada istri Anda. Bagaimana dia bisa merasakan/mengukur keunikannya? Dia mengambil semua hadiah yang mungkin dan menghitung probabilitasnya, jika probabilitasnya lebih besar, maka kita menyebutnya normal. Jika probabilitasnya sangat langka, maka kita menyebutnya Unik/Baru.

Estimasi kepadatan kernel digunakan dalam

1. Penipuan kartu kredit
2. Deteksi intrusi
3. Penipuan penjual
4. Perilaku terorisme

Estimasi kepadatan digunakan dalam memecahkan masalah klasifikasi bahkan. Bayangkan kita mengambil urutan gen dan mencari tahu apakah dia penderita diabetes atau tidak, bayangkan kompleksitas modelnya. Kita mungkin memiliki data urutan gen 1 juta pasien diabetes dan 1 juta urutan gen non-diabetes. Bayangkan kompleksitasnya di sini. Kami mengumpulkan urutan gen dari 1 juta pasien diabetes dan 1 juta orang non-diabetes dan kami mencoba mencari tahu bagian mana dari urutan gen yang harus saya lihat dan Bagian mana dari urutan gen yang perlu saya modifikasi saat bayi lahir sehingga dia tidak dapat mengembangkan diabetes sepanjang hidupnya. Jika kita tahu cara membangun Estimasi Kepadatan Kernel, kita dapat melakukan hal yang rumit tersebut. Kami mencoba mencari tahu urutan mana yang lebih umum pada penderita diabetes, dan bagaimana saya dapat memodifikasi urutan ini saat lahir sehingga mereka tidak akan pernah menderita diabetes selama sisa hidup mereka. Untuk melakukan ini, kami mencoba memahami estimasi Kepadatan, lalu kami dapat membangun pengklasifikasi yang rumit.

Pelatihan: Kami mungkin telah melihat banyak data, memahami berbagai jenis dan urutan gen yang memampatkannya menjadi bentuk. Berdasarkan ini, kami mencoba membuat model.

Pemberian skor: Setelah membuat model, terapkan model ke titik data baru, seperti saat titik data baru terlihat, kami dapat mengatakan bahwa ini adalah urutan gen diabetes atau bukan. Kami bahkan dapat mengatakan berapa probabilitasnya bahwa ia akan menderita diabetes.

Meskipun histogram secara fungsional sesuai untuk menampilkan data biaya kuliah yang luas, memeriksa distribusi secara visual dapat membuat frustrasi karena pemirsa dipaksa untuk secara mental Menghubungkan Titik-titik di antara tempat sampah. Selain itu, ukuran tempat sampah dan titik tengah yang dipilih tampaknya menyembunyikan tiga subkelompok dalam distribusi. Grafik dapat ditingkatkan dengan menambahkan estimasi kepadatan kernel (KDE) karena histogram dapat secara visual membuat frustrasi dan menyesatkan, terutama saat tempat sampah atau titik tengah tidak berukuran atau ditempatkan dengan tepat. Estimasi kepadatan kernel memberikan garis halus untuk diikuti, namun, mirip dengan "ukuran bin" yang dapat mendistorsi distribusi data dalam histogram. Pemilihan bandwidth merupakan bagian penting dari estimasi kepadatan kernel. Dalam sebagian besar teknik

estimasi kepadatan kernel, kernel memiliki bentuk dan bandwidth yang sama. Meskipun ada beberapa metode pemilihan bandwidth yang secara otomatis memilih ukuran kernel yang sesuai, yang paling populer adalah metode Sheather-Jones Plug-In (SJPI), yang digunakan sebagai default dalam prosedur KDE.

Metode pemilihan bandwidth lainnya:

- Simple Normal Reference (SNR),
- SNR with Inter-quartile range (SNRQ),
- Silverman's Rule of Thumb (SROT) dan
- Over smoothed (OS).

Memahami Estimasi Kepadatan Kernel dengan studi kasus: Mari kita lihat data Jewel, kita pergi ke toko perhiasan dan mengamati ornamen dari berbagai logam dan Model. dimensi/fitur/variabel/kolom. Mari kita lihat dataset Jewel, pada data di atas, kita ingin mengerjakan dua Fitur/variabel/dimensi/kolom yaitu, Logam dan Model. Logam dapat mengambil tiga nilai diskrit (Nominal) yaitu, Perak, Emas, dan Platina, Model bahkan dapat mengambil tiga nilai Reguler, Mewah, Antik. Jadi, kita sebut ini sebagai masalah klasifikasi. Apa yang kita coba lakukan di sini adalah kita ingin mencari tahu P(Mewah, Emas) dan seterusnya. Jika saya dapat melakukan ini untuk semua kombinasi, kita dapat mengatakan kita membangun sebuah model. Dan membuang data karena kita membuat model, probabilitas semua kombinasi.

Mari kita lakukan ini:

P(Fancy, Platinum)= Jumlah(Fancy, Platinum)/Jumlah Total Pelanggan =(15/150)=0,10, jadi kita dapat mengatakan estimasi kepadatan sebenarnya tentang penghitungan dan normalisasi. Jika kita memiliki nilai riil, kita masukkan ke dalam bin dan buat histogram, lalu normalisasikan.

Mari kita pikirkan untuk membuat sebuah model:

Berapakah P(Reguler, Perak) dan P(Antik, Emas)? Jika saya menghitung semua parameter, selesai sudah. Sekarang, apakah penting seberapa besar Toko Permata dan berapa banyak Pelanggan di sana? Jika kita membangun model, kita dapat menggunakannya di toko kecil atau toko yang sangat besar. Jika datanya sangat besar, model saya akan jauh lebih tangguh. Tidak ada yang namanya kompleksitas dalam data, kita dapat memiliki kompleksitas dalam model tetapi tidak dalam data.

Setiap kali kita berpikir untuk membuat model untuk KDE, kita perlu bertanya pada diri sendiri beberapa pertanyaan:

1. Berapa banyak parameter yang ada di sini?
2. Berapa banyak parameter bebas yang ada?
3. Apakah kita memiliki kendala?

Kita memiliki 9 (3*3) parameter dalam contoh ini dan kendala di sini adalah bahwa mereka harus berjumlah satu. Jadi, parameter bebasnya adalah 8 (Jumlah parameter-Jumlah kendala) (9-1). Parameter Bebas: Untuk memahami parameter bebas, mari saya ambil contoh melempar koin. Jika saya ingin memperkirakan Probabilitas, cukup bagi saya untuk mengetahui probabilitas Kepala. Saya tidak perlu mengetahui probabilitas ekor karena kita

tahu bahwa P(ekor) + P(kepala) sama dengan 1. Jadi, setiap kali kita memiliki kendala, jumlah parameter bebas berkurang. Jadi saya dapat menghitung P(ekor)=1- P(kepala).
Kendala: Jika melempar dadu enam sisi: Probabilitas memperoleh 1: 1/6, Probabilitas memperoleh 2: 1/6,...,Probabilitas memperoleh 6: 1/6. Itu berarti keenam probabilitas harus berjumlah 1. Ini adalah kendalanya.

Mari kita pahami cara menghitung probabilitas: Jika Anda seorang manusia, merasa khawatir dengan semua hal yang terjadi di sekitar Anda, Anda menggunakan Probabilitas Gabungan. (Kecemasan) Jika Anda seorang manusia, sama sekali tidak khawatir tentang apa pun yang terjadi di sekitar Anda, Anda menggunakan Probabilitas Perkiraan. (Depresi).

Probabilitas Gabungan (Menganggap bahwa semuanya berkorelasi): M=M1M2...Mn
Probabilitas Perkiraan (Menganggap bahwa semuanya Independen): M=M1+M2+...Mn.

Dalam kasus pertama, terlalu rumit dan dalam kasus kedua sangat sederhana sehingga Anda bahkan tidak dapat menggunakannya dalam Pembelajaran Mesin. Seni pembelajaran mesin adalah menemukan model kompleksitas yang tepat. Bayangkan seseorang diminta untuk mencari tahu kota mana yang disukai untuk orang lanjut usia di India dan Anda mengumpulkan Informasi berikut seperti Usia, Pendapatan, Tingkat Pendidikan, Lalu Lintas, Polusi, Suhu, dll. dari berbagai kota. Sekarang kita perlu menjawab pertanyaan seperti Bangalore jauh lebih nyaman jika dibandingkan dengan Delhi dalam hal polusi? Lalu bagaimana dengan lalu lintas? Apakah preferensi kota bergantung pada usia atau pendapatan? Estimasi Kepadatan Kernel berguna dalam situasi ini.

Mari saya ambil contoh lain, Dalam sebuah Wawancara, Manajer ingin memilih mereka yang sangat ahli dalam pengkodean, ia memberi tugas kepada orang-orang dan menjelaskan metodologi yang harus diikuti, Ia mengumpulkan waktu yang dibutuhkan dalam hitungan menit untuk mencapai tugas itu. Setelah membuat histogram dan mengamati plotnya, kita dapat memahami bahwa jika banyak programmer yang membutuhkan waktu 15-20 menit untuk menyelesaikan Tugas. Itu berarti bahwa jika ada yang membutuhkan waktu lebih dari 40 menit (Outlier), Kita dapat mengatakan bahwa mereka tidak cocok untuk pekerjaan programmer.

10.4 STUDI KASUS

Mari kita pahami hal ini dengan studi kasus diabetes di mana beberapa wanita hamil diberi Glukosa secara oral dan nilai glukosa plasma dikumpulkan setelah satu jam untuk memeriksa apakah wanita tersebut memiliki masalah diabetes akibat kehamilan atau tidak?. Rata-rata dan simpangan baku "OGTT" dikumpulkan dan kita harus membuat plot Kepadatan. Dari sini, kita melihat bahwa, dalam kumpulan data ini, kasus diabetes dikaitkan dengan tingkat "OGTT" yang lebih tinggi. Hal ini akan diperjelas dengan plot fungsi kepadatan yang diestimasikan. Plot ini menunjukkan estimasi kepadatan p(OGTT | diabetes=1), p(OGTT | diabetes=0), dan p(OGTT). Estimasi kepadatan adalah estimasi kepadatan kernel menggunakan kernel Gaussian. Artinya, fungsi kepadatan Gaussian ditempatkan pada setiap

titik data, dan jumlah fungsi kepadatan dihitung pada rentang data. Fungsi kernel menentukan bentuk tonjolan sementara lebar jendela h menentukan lebarnya.

#Langkah 1: Tetapkan direktori kerja dan Baca data

```
setwd('D:/R data')
Diabdata <- read.csv("Diab.csv", header=T)
```

#Langkah 2: Buat objek yang diperlukan dari data Diab

```
OGTT <- Diabdata[, 'OGTT']
d0 <- Diabdata[, 'Diabetic'] == 'No' d1 <- Diabdata[,
'Diabetic'] == 'Yes'
```

#Langkah 3: Buat plot kepadatan

```
plot(density(OGTT[d0]),bty="n",lwd=2, col='blue',
xlim=c(10,250), xlab="Oral Glucose Tolerance Test(OGTT)",
ylab='estimate p(OGTT)',
main="Distribusi orang berdasarkan OGTT")
lines(density(OGTT[d1]),col="#FFCCCC")
```

#Langkah 4: Mempercantik Plot:

#Pilih dan tambahkan kode warna HTML yang terlihat transparan pada plot

```
polygon(density(OGTT),col="#FFCCCC")
```

#Tambahkan garis untuk mean

```
abline(v=mean(OGTT))
```

#Tambahkan garis putus-putus berwarna abu-abu gelap dan lebih lebar untuk median

```
abline(v=median(OGTT),lwd=2,lty=3,col="#999999")
```

BAB 11
REGRESI

11.1 PENDAHULUAN

Analisis regresi adalah teknik statistik yang digunakan untuk menyimpulkan besaran dan arah hubungan kausal yang mungkin terjadi antara pola yang diamati dan variabel yang diasumsikan memiliki dampak pada pola yang diamati. Regresi akan memberi tahu Anda apakah hubungan tersebut ada? Terserah analis untuk memutuskan apakah ada hubungan kausal atau tidak.

Statistik – ini adalah statistik inferensial. Ini (regresi) adalah pendekatan matematis dan variabel yang memengaruhi pola semuanya adalah sampel acak dari populasi yang mendasarinya.

Besaran - Ukuran dampak, Arah - Positif atau Negatif

Kausal – Hubungan Sebab dan Akibat - Misalnya: Besaran curah hujan harus berdampak pada hasil panen, tetapi hasil panen tidak memengaruhi curah hujan

Studi Kasus: Mari kita mulai dengan sebuah masalah, Anda bekerja untuk sebuah perusahaan asuransi, dan Anda ingin memahami apa yang menyebabkan kematian di jalan raya untuk membantu menetapkan premi dengan tepat. Kita mulai dengan memikirkan tentang apa saja kemungkinan yang memengaruhi kematian. Alkohol, Bulan, Pengemudi, Kemacetan, Apa lagi?. Ada banyak alasan lain untuk kematian di jalan raya. Sangat sulit untuk menganalisis semua kemungkinan kematian di setiap negara di dunia. Katakanlah kita memutuskan untuk mengumpulkan data yang mudah didapat, kita dapat memperoleh kumpulan data yang berisi: Kematian per bulan

Apakah hari itu akhir pekan atau hari kerja?
Jumlah pengemudi berlisensi
Jumlah kecelakaan per bulan
Jumlah kendaraan yang ditempuh per mil jalan (Kemacetan)
Pengemudi di bawah pengaruh alkohol atau tidak

Sekarang, dengan data ini, kita ingin menganalisis hubungan antara variabel yang tersedia dan kematian untuk mengusulkan cara menetapkan premi yang memperhitungkan faktor-faktor yang berpotensi memengaruhi kematian. Apa saja cara yang mungkin untuk menilai hubungan tersebut? Kita dapat menggunakan visualisasi grafis atau korelasi.

11.2 MENGAPA REGRESI?

Teknik regresi dapat diterapkan pada masalah ini untuk memahami dampak variabel yang tersedia pada kematian. Keuntungan menggunakan teknik regresi adalah bahwa dengan menggunakan teknik regresi, kita akan dapat menilai dampak setiap faktor dengan mempertimbangkan dampak faktor lain yang juga diambil secara bersamaan. Pertama, jika

kita percaya bahwa beberapa faktor berdampak pada Kematian, maka pada dasarnya kita mendalilkan bahwa kematian adalah fungsi dari faktor-faktor yang diidentifikasi.
Secara matematis direpresentasikan sebagai berikut:- *Kematian = f (Pengemudi, Kemacetan)*
Setelah kita mengetahui bahwa Beberapa faktor berdampak pada kematian dependen, maka kita harus mengetahui bagaimana kematian dipengaruhi oleh Jumlah Pengemudi dan kemacetan. Solusi untuk masalah kita adalah Regresi. Untuk saat ini, mari kita batasi diri kita pada kasus di mana kita hanya mengkhawatirkan satu variabel yang memengaruhi kematian secara linier. Jadi, kita berhipotesis bahwa Kematian meningkat seiring dengan peningkatan jumlah pengemudi berlisensi di suatu negara bagian. Kematian = f (pengemudi), di mana f positif
Regresi Linier Sederhana: Model Regresi Linier Sederhana biasanya dilambangkan dengan:
$Y = \beta 0 + \beta 1X + e$ Di mana $\beta 0$=Intercept, $\beta 1$ = Slope, e = Error. Kita perlu memperkirakan beta sehingga kita dapat memahami hubungan antara Y dan X.
Variabel Dependen: Y: Variabel yang Diprediksi: Variabel yang perilakunya kita hipotesiskan dapat dijelaskan atau Dipengaruhi oleh faktor lain.
Variabel Independen: X(s): Prediktor yang kita hipotesiskan memengaruhi variabel dependen.
Koefisien Beta: Estimasi besarnya dampak perubahan prediktor pada variabel yang diprediksi.
Kesalahan: (e): Dampak variabel tak teramati pada variabel dependen biasanya dihitung sebagai perbedaan antara nilai prediksi Y yang diberikan fungsi regresi yang diestimasi dan nilai aktual Y.
Jika B = 0 maka Y adalah konstanta sehingga tidak ada hubungan antara Y dan X, karena, apa pun perubahan X, Y tidak berubah.
Apa yang terjadi ketika intersep = 0?. Garis Regresi melewati Titik Asal. (X α Y)
Apa yang terjadi ketika Kemiringan = 0?. Garis regresi akan sejajar dengan sumbu X (Tidak ada hubungan antara variabel dependen dan variabel independen).
Kembali ke contoh kematian, jika kita berasumsi untuk saat ini bahwa kita hanya memiliki data tentang jumlah pengemudi, maka model regresi kita akan menjadi,
*Kematian = ß0 + ß1 * Jumlah Pengemudi + e*

Jika kita memperkirakan nilai β, kita akan dapat memahami seberapa kuat dampak jumlah pengemudi terhadap jumlah kematian, dan melihat apa yang dapat dilakukan untuk mengurangi kematian. Ada banyak cara untuk memperkirakan koefisien beta. Untuk saat ini, kita akan fokus pada salah satu yang paling intuitif: Ordinary Least Squares.

11.3 REGRESI KUADRAT TERKECIL BIASA (OLS)

Teknik Regresi Kuadrat Terkecil Biasa memperkirakan koefisien pada variabel yang dihipotesiskan memiliki dampak pada variabel yang diminati dengan mengidentifikasi garis yang meminimalkan jumlah perbedaan kuadrat antara titik-titik pada garis yang diestimasikan dan nilai aktual dari variabel Independen.

Koefisien: Beta (ß1 dan ß0)
Meminimalkan: Terkecil (nilai)
Jumlah Perbedaan Kuadrat: Kuadrat residual

Garis yang diestimasikan: Garis Regresi
Nilai Aktual: Nilai dalam set data

Jelas, kita dapat membuat banyak garis lurus yang masing-masing akan mencakup beberapa titik. Karena garis lurus tidak dapat mengenai semua titik, salah satu cara memilih garis adalah dengan mengidentifikasi garis yang akan menjelaskan sebagian besar variasi dalam Y, atau dengan kata lain, memiliki kesalahan paling sedikit. Regresi kuadrat terkecil biasa menemukan garis tersebut dengan melihat residu (atau perbedaan antara titik-titik pada setiap garis dan Y aktual) dan meminimalkan jumlah kuadratnya.

Residu menangkap kesalahan dalam garis yang diestimasikan (perbedaan antara nilai estimasi garis dan nilai kehidupan nyata). Setelah garis diestimasikan, maka β0 adalah intersep garis tersebut, dan demikian pula β1 adalah kemiringan garis tersebut. Bagaimana kita benar-benar memperkirakan garis "terbaik"?

Kita dapat yakin bahwa berdasarkan data, garis estimasi kuadrat terkecil biasa meminimalkan kesalahan lebih dari garis lain yang kita pilih. Apakah ada garis lurus yang dapat mengenai semua titik? Estimasi OLS: Regresi Kuadrat Terkecil Biasa menemukan garis tersebut dengan melihat residu (atau perbedaan antara titik-titik pada setiap garis dan Y aktual) yang meminimalkan jumlah kuadratnya.

Mengapa jumlah kuadrat?: Untuk mengatasi, Perbedaan Positif dan Negatif Secara Matematis, minimalkan

$$Q = \sum_{i=1}^{N} (Y_i - \beta_o - \beta_i, X_i)^2$$

Dengan menggunakan kalkulus Diferensial kita akan memperoleh,

$$\beta_1 = \frac{\sum X_i Y_i - \sum X_i \sum Y_i}{n \sum X_i^2 - \sum (X_i)^2}$$

$$\beta_1 = \frac{\sum X_i^2 \sum Y_i - \sum X_i \sum X_i Y_i}{n \sum X_i^2 - \sum (X_i)^2}$$

Estimasi ini disebut estimasi kuadrat terkecil biasa. Dan garis lurus yang direpresentasikan oleh $Y = \beta_0 + \beta_1 X$ akan menjadi garis kuadrat terkecil di sini. Estimasi ini disebut estimasi Kuadrat Terkecil Biasa. Garis estimasi ini meminimalkan kesalahan lebih dari garis lain yang kita pilih. Setelah kita memperkirakan koefisien, kita memiliki persamaan seperti ini,

$$Kematian = Estimasi\ intersep + Koefisien\ Beta * Jumlah\ Pengemudi.$$

Ingat, ini adalah garis yang paling sesuai, tetapi garis ini tidak akan mencakup setiap titik pada diagram sebar. Jika kita menghitung nilai Kematian menggunakan nilai aktual Jumlah Pengemudi yang kita lihat dalam data, kita menghitung Kematian yang "Diprediksi". Selisih antara nilai kematian yang diprediksi dan nilai kematian aktual dalam data untuk setiap nilai sejumlah pengemudi adalah residu. Dengan menggunakan kumpulan data kematian, jalankan regresi sederhana dan cari tahu intersep (A) dan Kemiringan (B). Oleh karena itu, garis regresi yang diestimasikan adalah, $Kematian = A + B * Jumlah\ pengemudi$
Koefisien Beta: Untuk setiap peningkatan satuan Jumlah Pengemudi, kami memperkirakan akan terjadi peningkatan kematian dengan jumlah tertentu. Artinya, untuk peningkatan 100 pengemudi, kami memperkirakan akan terjadi peningkatan kematian dengan jumlah ini. Tanda positif pada koefisien pada sejumlah pengemudi menyiratkan hubungan positif antara Jumlah Pengemudi dan Kematian.
Nilai-P: H0 – Tidak ada dampak jumlah pengemudi (sumbu X) terhadap kematian (sumbu Y)
Nilai-P: Jika nilai-p < 0,05 maka koefisien signifikan pada tingkat 5%.

Semakin rendah nilai-p, semakin besar kemungkinan untuk menolak hipotesis H0. Jika nilai-P jauh lebih kecil dari 0,05, kami menolak H0. Akhirnya, kita dapat menyimpulkan bahwa tidak ada pengemudi yang memiliki dampak signifikan secara statistik terhadap kematian.
Meskipun persamaan regresi adalah persamaan garis lurus terbaik yang mungkin, bagaimana kita menilai efektivitas model keseluruhan? Salah satu caranya adalah dengan melihat ukuran “Keterjelasan”, yaitu, seberapa banyak variabel dependen Y yang dijelaskan oleh X? Atau, cara yang lebih baik untuk menjelaskannya adalah, seberapa banyak varians dalam Y yang dijelaskan oleh X?

Cara matematis untuk menghitungnya adalah:

$$R^2 \equiv 1 - \frac{SS_{err}}{SS_{tot}}$$

Dimana

$$\mathrm{SS_{err}} = \Sigma_i (\mathrm{y_i} - \mathrm{f_i})^2$$

$$\mathrm{SS_{tot}} = \Sigma_i (\mathrm{y_i} - \tilde{y})^2$$

Hitung nilai R-Square misalnya jika R2 = 0,9399, artinya 93,99% variasi variabel fatalitas dijelaskan oleh variasi variabel jumlah pengemudi. Semakin tinggi R2, semakin tinggi variasi variabel Dependen (Y) yang dijelaskan oleh variasi variabel Independen (X).

Buat model, dengan R2 terbaik yang memungkinkan dengan data yang Anda miliki dengan mencoba berbagai kombinasi variabel yang Anda miliki. R2 adalah satu cara untuk memvalidasi model, tetapi bukan satu-satunya cara untuk memvalidasi model. R2 juga

meningkat dengan penambahan variabel, baik relevan atau tidak, jadi lebih baik menggunakan ukuran R2 yang disesuaikan.

R2 yang disesuaikan: R2 yang disesuaikan melihat dampak variabel signifikan dalam suatu model. Bahkan jika Anda memiliki variabel yang tidak signifikan dalam model, R2 akan naik, tetapi R2 yang disesuaikan akan naik hanya jika model tersebut berisi variabel signifikan.

11.4 REGRESI BARIS BERGANDA

Sekarang, mari kita beralih ke kasus di mana kita mengharapkan beberapa variabel berdampak pada variabel dependen tertentu. Ini jelas terjadi dalam kehidupan nyata. Selama kita mengharapkan hubungan linear antara setiap variabel independen dan variabel dependen, kita dapat menggunakan teknik Kuadrat Terkecil untuk menghasilkan penduga koefisien beta. Kita akan kembali memperkirakan garis melintasi beberapa dimensi yang akan meminimalkan jumlah residual kuadrat.

Mari kita pikirkan tiga variabel yang akan memengaruhi Kematian yaitu Kemacetan, Pengemudi, dan Alkohol.

Kemacetan: seberapa padat jalan-jalan Kota

Pengemudi: Jumlah pengemudi berlisensi di Kota.

Alkohol: 0 menunjukkan pengemudi tidak mengonsumsi alkohol dan 1 menunjukkan pengemudi mengonsumsi alkohol.

Persamaan untuk estimasi OLS adalah:

$$Kematian = \beta_0 + \beta_1 * Kemacetan + \beta_2 * Pengemudi + \beta_3 * Alkohol + e$$

Sekarang, kita perlu memperkirakan 4 koefisien Beta: β_0, β_1, β_2, dan β_3. Kita akan menggunakan pendekatan OLS yang sama dengan meminimalkan jumlah residu kuadrat di berbagai dimensi.

Koefisien pengemudi akan berubah karena sekarang kita juga mengendalikan dampak kemacetan dan Alkohol. Jika tujuannya adalah untuk memprediksi dampak pengemudi, maka kita tidak memasukkan variabel kemacetan ke dalam model. Dalam kasus ini, kita dapat membuang variabel kemacetan dan menjalankan kembali model hanya dengan variabel yang signifikan.

11.5 MEMBANGUN MODEL DAN VALIDASI

Langkah 1: Instal dan Muat Paket yang Diperlukan:

```
library(ggplot2) library(caret) library(lattice)
```

Langkah 2: Muat data

```
Fataldata =read.csv(file="D:\\R data\\Fatalities.csv",
header=TRUE, sep=",")
```

Langkah 3: Jelajahi data

```
str(Fataldata)
fix(Fataldata)
```

Langkah 4: Siapkan data

4.1 Mengonversi variabel kategoris ke faktor

```
Fataldata$Weekend = as.factor(Fataldata$Weekend)
Fataldata$Alcohol = as.factor(Fataldata$Alcohol) Fataldata$Month
= as.factor(Fataldata$Month)
```

4.2 Hapus variabel dependen

```
Fataldata_a = subset(Fataldata, select = -c(Kematian))
```

4.3 Mengidentifikasi variabel numerik

```
numericdata <- Fataldata_a[sapply(Fataldata_a, is.numeric)]
```

4.4 Menghitung Korelasi

```
descrCor <- cor(numericdata)
highlyCorrelated <- findCorrelation(descrCor, cutoff=0.4)
```

4.5 Mengidentifikasi Nama Variabel dari Variabel yang Sangat Berkorelasi

```
highlyCorCol <- colnames(numericdata)[highlyCorrelated]
```

4.6 Mencetak atribut yang sangat berkorelasi

```
highlyCorCol
```

4.7 Menghapus variabel yang sangat berkorelasi dan membuat kumpulan data baru

```
dat3 <- Fataldata[, -which(colnames(Fataldata) %in%
highlyCorCol)] dim(dat3)
str(dat3)
```

Langkah 5: Membangun Model Regresi Linier

```
fit0 = lm(Kematian ~ Pengemudi, data=Fataldata)
fit2 = lm(Kematian ~ Akhir Pekan, data=Data Fatal)
fit3 = lm(Kematian ~ Akhir Pekan+Pembalap, data=Data Fatal)
fit = lm(Kematian ~ ., data=Data Fatal)
```

Langkah 6: Periksa Kinerja Model

```
summary(fit) summary(fit0)
```

6.1 Mengekstrak Koefisien

```
summary(fit)$coeff
```

6.2 Mengekstrak nilai Rsquared

```
summary(fit)$r.squared
```

6.3 Mengekstrak Adj. Nilai Rsquared

```
summary(fit)$adj.r.squared
```

6.4 Pemilihan Bertahap berdasarkan AIC

```
library(MASS)
step <- stepAIC(fit, direction="both") summary(step)
```

6.5 Pemilihan Mundur berdasarkan AIC

```
step <- stepAIC(fit, direction="backward") summary(step)
```

6.6 Pemilihan Maju berdasarkan AIC

```
step <- stepAIC(fit, direction="forward") summary(step)
```

6.7 Pemilihan Bertahap dengan BIC

```
n = dim(dat3)
stepBIC = stepAIC(fit,k=log(n)) summary(stepBIC)
```

Langkah 7: Diagnostik Model

Kita perlu memeriksa validitas asumsi utama model regresi linier. Ini mengacu pada distribusi istilah kesalahan model, yaitu varians homogen, normalitas, dan independensi. Analisis residual yang diamati dapat membantu mengevaluasi kemungkinan asumsi ini. Memeriksa observasi yang tidak biasa dan berpengaruh adalah bagian lain dari diagnostik regresi.

7.1 Memeriksa Outlier: Paket mobil berisi uji outlier Bonferroni yang hanya menghitung dan menilai Outlier.

```
library(car)
outlierTest(stepBIC) # Outlier - Uji Bonferroni
```

7.2 Periksa Normalitas: Histogram dan diagram kotak juga cocok untuk memeriksa normalitas, bersama dengan statistik deskriptif seperti kemiringan dan kurtosis, misalnya.

```
hist(residuals(fit))
boxplot(residuals(fit))
```

Normalitas Residual:# qq plot untuk residual yang distudentisasi

```
qqPlot(fit, main="QQ Plot")
```

Uji normalitas Shapiro-Wilks: #Normalitas Residual (Harus > 0,05). Hipotesis nol adalah bahwa residual memiliki distribusi normal. Nilai-p dari statistik uji besar dalam contoh ini. Dengan demikian, hipotesis nol tidak ditolak.

```
res=residuals(stepBIC,type="pearson") shapiro.test(res)
```

7.3 Autokorelasi: Statistik Durbin–Watson adalah statistik uji yang digunakan untuk mendeteksi keberadaan autokorelasi (hubungan antara nilai yang dipisahkan satu sama lain oleh jeda waktu tertentu) dalam residual (kesalahan prediksi) dari analisis regresi. Nilai p menunjukkan bahwa tidak ada bukti kesalahan berkorelasi, tetapi hasilnya harus dilihat dengan skeptis karena penghilangan nilai yang hilang.

Uji Kesalahan Autokorelasi

```
durbinWatsonTest(stepBIC)
```

7.4 Multikolinearitas: Kami memeriksa VIF dari semua variabel untuk Menguji Multikolinearitas. Faktor inflasi varians (VIF) mengukur seberapa besar varians koefisien regresi yang diestimasi meningkat dibandingkan dengan ketika variabel prediktor tidak terkait secara linier. VIF berguna untuk menggambarkan seberapa besar Multikolinearitas (korelasi antara prediktor) yang ada dalam analisis regresi. Multikolinearitas bermasalah karena dapat meningkatkan varians koefisien regresi, sehingga menjadi tidak stabil dan sulit diinterpretasikan.

Kita dapat menggunakan panduan berikut untuk menginterpretasikan VIF: Jika VIF kurang dari atau sama dengan 1, kita dapat mengatakan bahwa prediktor tidak berkorelasi. Jika VIF antara 1 hingga 5, prediktor berkorelasi sedang dan jika VIF lebih besar dari 5, kita dapat menyimpulkan bahwa prediktor berkorelasi tinggi.

Mengevaluasi Multikolinearitas

```
vif(stepBIC)
```

faktor inflasi varians

7.5 Homoskedastisitas (Varians konstan): Asumsi penting dari regresi linier adalah bahwa tidak boleh ada heteroskedastisitas residual. Artinya, varians residual tidak boleh meningkat

dengan nilai variabel respons yang disesuaikan. Merupakan hal yang rutin untuk memeriksa heteroskedastisitas residual setelah membangun model regresi linier. Karena kita ingin memeriksa apakah model yang dibangun tidak dapat menjelaskan beberapa pola dalam variabel respons (Y), yang akhirnya muncul dalam residual. Hal ini akan menghasilkan model regresi yang tidak efisien yang dapat menghasilkan prediksi aneh di kemudian hari saat kita benar-benar menggunakan model tersebut.

Ada beberapa pengujian yang dapat kita gunakan untuk memeriksa ada atau tidaknya heteroskedastisitas: 1. Pengujian Breush-Pagan dan 2. Pengujian NCV.

Uji Breush Pagan

```
HS_test1 <- bptest(fit)#Uji Breusch-Pagan
```

Uji NCV

```
HS_test2 <- ncvTest(fit) #Uji Skor Varians Tidak Konstan
```

Baik HS_test1 maupun HS_test2 memiliki nilai-p kurang dari tingkat signifikansi 0,05, oleh karena itu kita dapat menolak hipotesis nol bahwa varians residual bersifat konstan dan menyimpulkan bahwa heteroskedastisitas memang ada. Kita dapat memperbaiki masalah ini dengan menggunakan dua metode: 1. Bangun kembali model dengan prediktor baru. 2. Lakukan transformasi variabel seperti transformasi Box-Cox.

7.6 Transformasi Box-Cox: Transformasi Box-Cox adalah transformasi matematis variabel untuk membuatnya mendekati distribusi normal. Melakukan transformasi Box-Cox dari variabel Y sering kali memecahkan masalah Heteroskedastisitas.

```
Death_Boxcox <- caret::BoxCoxTrans(Fataldata$Deaths)
print(Death_Boxcox)
```

Model untuk membuat variabel yang ditransformasikan dari box-cox sudah siap. Sekarang mari kita terapkan pada `Fataldata$Deaths` dan tambahkan ke kerangka data baru. Tambahkan variabel yang ditransformasikan ke Fataldata.

```
Fataldata <- cbind(Fataldata, Deaths_new=predict(Death_Boxcox,
Fataldata$ Deaths))
```

Data yang ditransformasikan untuk model regresi baru kita sudah siap. Mari kita buat model dan periksa heteroskedastisitas.

```
Fatal_bc <- lm(Deaths_new ~ ., data=Fataldata)
```

Lakukan uji Breusch-Pagan sekali lagi untuk memeriksa apakah masalah heteroskedastisitas telah diatasi atau belum.

```
bptest(Fatal_bc)
```

Karena Nilai-P yang diperoleh lebih besar dari 0,05, kita dapat mengatakan bahwa residualnya sekarang Homoskedastisitas.

7.7 Pengamatan yang Berpengaruh: Pengamatan yang berpengaruh adalah pengamatan untuk perhitungan statistik yang penghapusannya dari kumpulan data akan mengubah hasil perhitungan secara signifikan. Pengamatan yang berpengaruh didefinisikan sebagai pengamatan yang mengubah kemiringan garis. Dengan demikian, titik-titik yang berpengaruh memiliki pengaruh besar pada kesesuaian model. Jadi, kita perlu memeriksa pengamatan yang berpengaruh dalam kumpulan data kita. Ukuran jarak Cook adalah kombinasi dari efek residual dan leverage. tujuan dari pengukuran jarak Cook adalah deteksi observasi yang

berpengaruh dan deteksi pengaruh gabungan dari outlier, baik dalam variabel respons Y maupun variabel penjelas X.

#Plot D Cook: identifikasi nilai D > 5/(n-k-1)

```
cutoff <- 5/((nrow(Fataldata)-length(fit$coefficients)-2))
plot(fit, which=5, cook.levels=cutoff)
```

#Kepentingan Relatif

```
library(relaimpo) calc.relimp(stepBIC)
```

#Lihat Nilai yang Diprediksi

```
pred = predict(stepBIC,Fataldata)
```

#Lihat Nilai Aktual vs. yang Diprediksi

```
finaldata = cbind(Fataldata,pred)
print(head(subset(finaldata, select = c(Deaths,pred))))
```

Langkah 8: Menghitung RMSE

```
rmse <- sqrt(mean((Fataldata$Deaths - pred)^2)) print(rmse)
```

#Menghitung Rsquared secara manual

```
y = Fataldata[,c("Kematian")]
R.squared = 1 - sum((y-pred)^2)/sum((y-mean(y))^2)
print(R.squared)
```

#Menghitung Adj. Rsquared secara manual

```
n = dim(Fataldata)[1]
p = dim(summary(stepBIC)$coeff)[1] - 1 adj.r.squared = 1 - (1 -
R.squared) * ((n - 1)/(n-p-1)) print(adj.r.squared)
```

BAB 12
REGRESI LOGISTIK

12.1 PENDAHULUAN

Banyak masalah penelitian memiliki hasil dikotomis, apakah pelanggan akan berhenti berlangganan atau tidak, apakah pinjaman akan dibayar atau gagal bayar, apakah pasien menderita kanker atau tidak, dan seterusnya. Biasanya, pertanyaan-pertanyaan ini ditangani dengan regresi kuadrat terkecil biasa (OLS) atau analisis fungsi Diskriminan linier. Namun, ini ditemukan kurang ideal untuk menangani hasil dikotomis karena asumsi statistiknya yang ketat, seperti linearitas, normalitas, dan kontinuitas untuk regresi OLS dan normalitas multivariat dengan varians dan kovariansi yang sama untuk analisis Diskriminan. Regresi logistik memperluas gagasan regresi linier ke situasi di mana variabel dependen Y, bersifat kategoris. Kita dapat menganggap variabel kategoris sebagai pembagian observasi ke dalam kelas-kelas. Misalnya, jika Y menunjukkan apakah pelanggan tertentu cenderung membeli suatu produk (1) atau tidak cenderung membeli (0), kita memiliki variabel kategoris dengan 2 kategori atau kelas (0 dan 1). Hipotesis regresi linier bisa jauh lebih besar dari 1 atau jauh lebih kecil dari nol dan karenanya permulaan menjadi sulit.

Dalam regresi logistik, kita mengambil dua langkah: 1. Temukan estimasi probabilitas untuk masuk ke setiap kelas. Kasus ketika Y = 0 atau 1, probabilitas untuk masuk ke kelas 1, P(Y=1) dan 2. Gunakan nilai batas pada probabilitas ini untuk mengklasifikasikan setiap kasus dalam salah satu kelas. Misalnya dalam kasus biner, batas 0,5 berarti bahwa kasus dengan estimasi probabilitas P(Y=1) > 0,5 diklasifikasikan sebagai masuk ke kelas 1, sedangkan kasus dengan P(Y=0) < 0,5 diklasifikasikan sebagai masuk ke kelas 0. Batas tidak perlu ditetapkan pada 0,5. Ketika peristiwa dalam pertanyaan adalah peristiwa dengan probabilitas rendah, nilai batas yang lebih tinggi dari rata-rata, meskipun di bawah 0,5 mungkin cukup untuk mengklasifikasikan. Menentukan nilai batas adalah 'Seni' bukan sains. Analisis regresi logistik digunakan untuk prediksi variabel kategoris (Binomial, Ordinal) menggunakan campuran prediktor kontinu dan diskrit. Regresi Logistik digunakan ketika variabel Dependen: Kategoris dan variabel Independen bersifat Kontinu atau Kategoris. Regresi Logistik digunakan ketika tujuan penelitian difokuskan pada apakah suatu peristiwa terjadi atau tidak, bukan pada saat peristiwa itu terjadi, yaitu informasi tentang rentang waktu tidak digunakan. Di sini, alih-alih membangun model prediktif untuk "Y (Respons)" secara langsung, pendekatan tersebut memodelkan Log Odds (Y); oleh karena itu dinamakan Logistik atau Logit.

Contoh:

Default - Kartu Kredit
Respon - Surat Langsung
Akuisisi - Pelanggan
Rekomendasi – Pembelian

Beberapa Jenis Regresi Logistik

1. **Logit Biner:**
   Digunakan ketika variabel respons bersifat biner atau dikotomis. Variabel ini hanya memiliki 2 hasil, misalnya Baik vs Buruk, Ya vs Tidak.
2. **Logit Multinomial:**
   Digunakan ketika variabel respons memiliki lebih dari 2 hasil, dan Hasil tidak dapat diurutkan dengan cara apa pun, misalnya Pilihan minuman, pilihan tempat wisata.
3. **Logit Terurut:**
   Digunakan ketika variabel respons memiliki lebih dari 2 hasil, dan
   Hasil dapat diurutkan dengan cara yang bermakna, misalnya Tinggi / Sedang / Rendah, Sangat Setuju / Setuju / Tidak Setuju / Sangat Tidak Setuju
   Studi Kasus: Mari kita pertimbangkan sampel nasabah yang diberi pinjaman Perumahan oleh bank. Kami ingin membuat model yang menilai dampak beberapa faktor terhadap kelayakan pinjaman Perumahan. Kami memiliki data berikut yang tersedia:
   Id
   Usia Pemohon
   Jenis Kelamin
   Pengalaman dalam Tahun
   Pendapatan Bulanan dalam Ribuan Jumlah Keluarga
   Tingkat Pendidikan Pinjaman_Disetujui atau tidak

   Salah satu cara untuk menilai dampak faktor-faktor pada kelayakan pinjaman pribadi adalah dengan membuat model regresi
   Kelayakan Pinjaman = f (Pendapatan, Usia, Pendidikan)
   Apa persamaan OLS untuk model regresi tersebut?
   *Kelayakan (Y) = β0+ β1 *Pendapatan + β2 * Usia + β3 * Pendidikan +e*

12.2 MENGAPA REGRESI LOGISTIK?

Probabilitas Pinjaman_Disetujui tidak linier

Kita melihat bahwa hampir tidak ada seorang pun yang disetujui pinjaman pada kisaran rendah, dan Hampir semua orang disetujui pinjaman pada kisaran Pendapatan tinggi. Perubahan dalam probabilitas disetujui pinjaman pada kisaran Pendapatan rendah dan tinggi adalah minimal, sedangkan, di tengah kisaran, perubahan probabilitasnya besar. Berapa nilai Y? Jika kita menggunakan model regresi linier, nilai yang diprediksi tidak terbatas (- hingga +). Namun, dalam kasus ini, nilai probabilitas dibatasi pada 0 hingga 1. Salah satu cara untuk menyelesaikan masalah ini adalah dengan mengambil rasio peluang (p/1-p) lalu mengambil logaritma.

$p/(l-p)$ - dapat mengambil nilai dari 0 hingga ,

$log(p/1-p)$ dapat mengambil nilai dari - hingga +

Konsep matematika yang mendasari regresi logistik adalah logit, logaritma natural dari rasio peluang.

Apa itu "Rasio Peluang"?: Ini adalah istilah statistik standar yang menunjukkan probabilitas keberhasilan terhadap probabilitas kegagalan. Jika probabilitas keberhasilan adalah 0,75, maka rasio peluang = (0,75/0,25) = 3. Dengan kata lain, ada peluang keberhasilan 3:1. Transformasi dari probabilitas ke peluang adalah transformasi monotonik, yang berarti peluang meningkat seiring dengan peningkatan probabilitas atau sebaliknya. Peluang berkisar dari 0 hingga 1. Peluang berkisar dari 0 hingga positif tak terhingga.

Transformasi dari peluang ke logaritma peluang adalah transformasi logaritma. Sekali lagi, ini adalah transformasi monotonik. Artinya, semakin besar peluang, semakin besar logaritma peluang dan sebaliknya.

Mengapa kita bersusah payah melakukan transformasi dari peluang ke logaritma peluang? Salah satu alasannya adalah karena biasanya sulit untuk memodelkan variabel yang memiliki rentang terbatas, seperti peluang. Transformasi ini merupakan upaya untuk mengatasi masalah rentang terbatas. Transformasi ini memetakan peluang yang berkisar antara 0 dan 1 ke logaritma peluang yang berkisar dari negatif tak terhingga hingga positif tak terhingga. Alasan lainnya adalah bahwa di antara semua pilihan transformasi yang tak terhingga, logaritma peluang adalah salah satu yang paling mudah dipahami dan ditafsirkan. Transformasi ini disebut transformasi logit.

Bentuk Model Regresi Logistik

Oleh karena itu, Bentuk Modelnya adalah:

- Aditif: log (p/1-p) = Y= β0+ β1 *Pendapatan
- Istilah perkalian: (p/1-p) = e β0 * e β1 * Pendapatan
- Nilai Y tidak dibatasi pada 0 dan 1.
- Transformasi logaritma memiliki hubungan linier dengan prediktor (perubahan satuan pada X akan menyebabkan perubahan % tetap pada logY)
- Dalam hal Y: perubahan satuan pada X akan menyebabkan perubahan eβ perkalian pada Y (pengganda peluang)
- %perubahan pada Y kira-kira =100 * (e β -1) (untuk nilai koefisien yang kecil)

Jika kita menginginkan model dalam hal p:

$$P = \frac{e^{\left(\beta_0+\beta_1 X_1+\beta_2 X_2+\ldots \beta_k X_k\right)}}{1-e^{\left(\beta_0+\beta_1 X_1+\beta_2 X_2+\ldots \beta_k X_k\right)}}$$

Di mana:

P - Probabilitas kejadian

β0 - parameter intersep (nilai variabel dependen, ketika variabel independen (x) sama dengan nol)

X - Kumpulan variabel independen (prediktor)

βk adalah parameter kemiringan (perubahan variabel independen untuk perubahan unit dalam variabel prediktor).

12.3 SET DATA PINJAMAN PERUMAHAN

Sebuah Bank ingin memahami faktor-faktor yang memengaruhi kelayakan untuk pinjaman Perumahan, berdasarkan data historis nasabahnya yang ada.

Penyusunan Data untuk Regresi Logistik meliputi:

Pengodean Variabel Respons:

Variabel respons (atau variabel target) perlu diubah menjadi 1/0. Kodekan "Pinjaman Perumahan yang Disetujui" sebagai "1" dan "Pinjaman Perumahan yang Ditolak" sebagai "0". Perlakuan nilai yang hilang - menggunakan aturan logis.

Pendeteksian outlier - untuk memastikan kita tidak memiliki nilai yang sangat miring. Multikolinearitas - dua variabel independen tidak memberikan informasi yang serupa. Transformasi variabel - kita memiliki transformasi variabel yang bermakna tergantung pada cakupan penelitian dan pemodelan. Statistik deskriptif - Ukuran dasar kecenderungan sentral perlu dikeluarkan untuk memvalidasi apakah data yang benar digunakan untuk pemodelan. Pembagian Data: Membagi sampel menjadi 2 sub-sampel, 1. Sampel pengembangan (Pelatihan), 2. Sampel validasi. Sampel pengembangan adalah Sampel yang digunakan untuk membangun model regresi Logistik. Sampel validasi digunakan untuk Estimasi yang diperoleh dari sampel pengembangan akan diuji di sini untuk perbandingan dan memeriksa kekokohan model. Sampel Seimbang: Idealnya: Proporsi 1 terhadap 0 tidak boleh kurang dari 2%. Jika "peristiwa langka proporsi 1:< 2% - oversample. Pertahankan peristiwa langka dalam sampel sebagaimana adanya, dan kurangi non-peristiwa. Pendekatan buatan ini tidak mengubah bentuk model yang melekat. Ini berdampak pada suku konstan atau intersep dan harus diperbaiki setelah model difinalisasi. Faktor Koreksi Sampel Seimbang: Dalam kasus ini, kita memiliki bentuk model regresi logistik sebagai

```
Log(pi/1-pi) = a + blX1+ b2X2 + b3X3.
```

Estimasi Parameter Regresi Logistik

Estimasi untuk Logistik: Koefisien untuk persamaan Logistik diestimasi menggunakan teknik yang dikenal sebagai Estimasi Kemungkinan Maksimum (MLE). MLE adalah metode estimasi yang populer karena tidak memiliki asumsi distribusi yang mendasarinya. Ketika distribusi dasar dari istilah kesalahan adalah normal, estimasi MLE mirip dengan estimasi OLS. OLS seperti banyak distribusi lainnya adalah kasus khusus MLE.

12.4 KINERJA MODEL REGRESI LOGISTIK

Untuk mengevaluasi kinerja model regresi logistik, kita harus mempertimbangkan beberapa metrik.

1. ***Uji Rasio Kemungkinan:*** Model yang dibuat oleh regresi logistik dianggap lebih sesuai dengan data jika menunjukkan kesesuaian yang baik dengan lebih sedikit prediktor. Dalam uji rasio kemungkinan, kita membandingkan kemungkinan data di bawah model lengkap terhadap model dengan lebih sedikit prediktor. Menghapus variabel prediktor dari model hampir selalu akan membuat model kurang sesuai, tetapi perlu untuk menguji apakah perbedaan yang diamati dalam kesesuaian model signifikan secara

statistik. Kami menganggap Hipotesis Nol (H0) menyatakan bahwa model yang direduksi itu benar, nilai-p untuk statistik kecocokan model secara keseluruhan yang kurang dari 0,05 kami menolak hipotesis nol. Uji rasio kemungkinan dapat dilakukan di R menggunakan fungsi lrtest() dari paket lmtest atau menggunakan fungsi ANOVA() di basis.

```
model<- glm(Loan_sanctioned~ Age+Experience+Income+Family,
data=loan_train, family = binomial)
model2<- glm(Loan_sanctioned~ Income+Family,
data=loan_train, family = binomial) anova(model, model2,
test ="Chisq") library(lmtest)
lrtest(model, model2)
```

Kami memperoleh Nilai yang jauh lebih besar dari 0,05, Jadi, kami gagal menolak hipotesis nol, dan menyimpulkan bahwa menghilangkan Usia dan Pengalaman dari model tidak memiliki dampak apa pun pada kinerja model.

2. ***Pentingnya Variabel:*** Kami menggunakan fungsi varImp dalam paket caret untuk menilai kepentingan relatif prediktor individual dalam model, kami juga dapat melihat nilai absolut dari statistik-t untuk setiap parameter model.

```
library(caret) varImp(model) varImp(model2)
```

Dengan melihat nilai-nilai tersebut, kami dapat mengatakan bahwa Pendapatan dan Jumlah Keluarga memainkan peran Penting dalam memprediksi apakah Pinjaman akan disetujui atau tidak. Jadi, kami dapat menghapus Usia dan Pengalaman dari model.

3. ***Validasi Nilai yang Diprediksi (Tingkat Klasifikasi):*** Ini melibatkan penggunaan estimasi model untuk memprediksi nilai pada set pelatihan dan membandingkan variabel target yang diprediksi dengan nilai yang diamati untuk setiap pengamatan. *Matriks Kebingungan:* Matriks Kebingungan adalah tabel yang sering digunakan untuk menggambarkan kinerja model klasifikasi pada set data uji yang nilai sebenarnya diketahui.

<table>
<tr><td colspan="2" rowspan="2">Confusion matrix</td><td colspan="2">Target</td><td colspan="2" rowspan="2"></td></tr>
<tr><td>Positif</td><td>Negatif</td></tr>
<tr><td rowspan="2">Model</td><td>Positif</td><td></td><td></td><td>Positif predicitive value</td><td>a/(a+b)</td></tr>
<tr><td>negatf</td><td></td><td></td><td>Positif predicitive value</td><td>d/(c+d)</td></tr>
<tr><td colspan="2" rowspan="2"></td><td>Sensitif</td><td>Spesifik</td><td colspan="2" rowspan="2">Akurasi=(a+d)/(a+b+c+d)</td></tr>
<tr><td>a/(a+c)</td><td>d/(b+d)</td></tr>
</table>

Akurasi: Proporsi jumlah total prediksi yang benar.

Akurasi = (A+D)/ (A+B+C+D)

Nilai Prediktif Positif atau Presisi: Proporsi kasus positif yang diidentifikasi dengan benar.

Presisi = A/(A+B)

Nilai Prediktif Negatif: Proporsi kasus negatif yang diidentifikasi dengan benar.

NPV = D/(C+D)

Sensitivitas atau Recall atau True Positive rate: proporsi kasus positif aktual yang diidentifikasi dengan benar.

TPR = A/(A+C)

Spesifisitas atau True Negative Rate: proporsi kasus negatif aktual yang diidentifikasi dengan benar.

TNR = D/(B+D)

4. ***Kurva Karakteristik Operasional Penerima (Kurva ROC):*** Karakteristik operasi penerima adalah ukuran kinerja pengklasifikasi. Dengan menggunakan proporsi titik data positif yang dianggap positif dengan benar dan proporsi titik data negatif yang dianggap positif secara keliru, kami menghasilkan grafik yang menunjukkan tradeoff antara tingkat prediksi yang benar dengan tingkat prediksi yang salah. Pada akhirnya, kami memperhatikan area di bawah kurva ROC, atau AUROC. Metrik tersebut berkisar antara 0,50 hingga 1,00, dan nilai di atas 0,80 menunjukkan bahwa model tersebut berfungsi dengan baik dalam membedakan antara dua kategori yang mencakup variabel target kami. # Hitung AUC untuk memprediksi Loan_sanctioned dengan variabel Income

```
library(ROCR)
pred <- prediction(predict, train_data$Loan_sanctioned)
perf <- performance(pred, 'tpr','fpr')
plot(perf, colorize = TRUE, text.adj = c(-0.2,1.7))plot(f1,
col="red")
auc <- performance(pred, measure = "auc") auc <-
auc@y.values[[1]]
auc
```

Kami memperoleh nilai auc (Area Under Curve) yang lebih besar dari 90% sehingga kami dapat menyimpulkan bahwa model kami bekerja dengan sangat baik dan model kami tervalidasi. Kurva ROC secara virtual independen dari tingkat respons. Ini karena kurva tersebut memiliki dua sumbu yang keluar dari perhitungan kolom matriks kebingungan. Pembilang dan penyebut dari sumbu x dan y akan berubah pada skala yang sama jika terjadi pergeseran tingkat respons. Inilah keuntungan menggunakan kurva ROC.

5. ***Root Mean Squared Error (RMSE):*** RMSE adalah metrik evaluasi paling populer yang digunakan dalam masalah regresi. Metrik ini mengikuti asumsi bahwa kesalahan tidak bias dan mengikuti distribusi normal. Sifat 'kuadrat' metrik ini membantu memberikan hasil yang lebih kuat yang mencegah pembatalan nilai kesalahan positif dan negatif. Metrik ini menghindari penggunaan nilai kesalahan absolut yang sangat tidak diinginkan dalam perhitungan matematika. RMSE sangat dipengaruhi oleh nilai outlier. Oleh karena itu, kita perlu memastikan bahwa kita menghapus outlier dari kumpulan data kita sebelum menggunakan metrik ini.
6. ***AIC (Akaike Information Criteria):*** AIC adalah ukuran kecocokan yang menghukum model untuk jumlah koefisien model. Oleh karena itu, kita selalu lebih memilih model dengan nilai AIC minimum.

7. ***K-Fold Cross Validation:*** Dalam metode ini, kita mempartisi data menjadi k segmen berukuran sama (disebut 'lipatan'). Satu lipatan ditahan untuk validasi sementara lipatan k-1 lainnya digunakan untuk melatih model dan kemudian digunakan untuk memprediksi variabel target dalam data pengujian kami. Proses ini diulang k kali, dengan kinerja setiap model dalam memprediksi set hold-out dilacak menggunakan metrik kinerja seperti akurasi. Variasi validasi silang yang paling umum adalah validasi silang 10 kali lipat. Validasi silang k-lipat banyak digunakan untuk memeriksa apakah suatu model overfit atau tidak. Untuk k kecil, kami memiliki bias seleksi yang lebih tinggi tetapi varians rendah dalam kinerja. Untuk k besar, kami memiliki bias seleksi kecil tetapi varians tinggi dalam kinerja.
   Uji Hosmer-Lemeshow: Statistik Homer-Lemeshow, yang dihitung pada data setelah pengamatan disegmentasi menjadi kelompok berdasarkan probabilitas prediksi yang sebanding. Ini meneliti apakah proporsi kejadian yang diamati mirip dengan probabilitas kejadian yang diprediksi dalam subkelompok set data menggunakan uji chi square. Nilai kecil dengan nilai-p besar menunjukkan kesesuaian yang baik terhadap data, sedangkan nilai besar dengan nilai-p di bawah 0,05 menunjukkan kesesuaian yang buruk.

12.5 MEMBANGUN REGRESI LOGISTIK

#Langkah 1: Memuat Data ke R:

```
setwd("D:/R data") loandata=read.csv(file="Housing_loan.csv",
header=TRUE)
```

#Langkah 2: Persiapan Data:

Hapus kolom ID & Gender dari data

```
loandata2=subset(loandata, select=-c(ID, Gender)) fix(loandata2)
```

Langkah 3: Buat variabel Dummy:

#Variabel "Pendidikan" memiliki lebih dari dua kategori (1: Sarjana, 2: Pascasarjana, 3: Lanjutan/Profesional), Jadi kita perlu membuat variabel dummy untuk setiap kategori untuk dimasukkan ke dalam analisis. #Instal & Muat paket "dummies" untuk membuat variabel dummy

```
install.packages("dummies") library(dummies)
Edu_dum=dummy(loandata2$Education) head(Edu_dum)
loandata3=subset(loandata2,select=-c(Education))
loandata4=cbind(loandata3,Edu_dum) head(loandata4)
```

#Langkah 4: Standardisasi Data:

Standarisasi data menggunakan metode 'Range'

```
install.packages("vegan") library(vegan)
loandata5=decostand(loandata4,"range")
```

#Langkah 5: Siapkan set data Latih & Uji:

#Ambil sampel acak 80% dari rekaman untuk data latih.

```
train = sample(1:1000,800) train_data = loandata5[train,]
nrow(train_data)
```

#Ambil sampel acak 20% dari rekaman untuk data uji

```
test = (1:1000) [-train] test_data = loandata5[test,]
nrow(test_data)
```

#Langkah 6: Ringkasan Data untuk variabel respons "Loan_sanctioned":

```
table(loandata5$Loan_sanctioned) #Total Data
table(train_data $Loan_sanctioned) #Train Data
table(test_data$Loan_sanctioned) #Test Data
```

#Langkah 7: Bangun model regresi Logistik

```
model<- glm(Loan_sanctioned~ Age+Experience+ Income+Family+
Education1+Education2+ Education3, data=train_data, family =
binomial)
```

Langkah 8: Periksa Ringkasan model dan Evaluasi model dengan memperoleh kebingungan matrix

```
summary(model)
predict <- predict(model, type = 'response')
table(train_data$Loan_sanctioned, predict > 0.5)
```

Langkah 9: Buat Kurva ROC dan Periksa Area di Bawah Kurva

```
library(ROCR)
pred <- prediction(predict, train_data$Loan_sanctioned) perf <-
performance(pred, 'tpr','fpr')
plot(perf, colorize = TRUE, text.adj = c(-0.2,1.2))
```

#Langkah 10: Coba dengan kombinasi variabel yang berbeda dan evaluasi kinerja model

```
model2<- glm(Loan_sanctioned~ Age+Experience+Income+Family,
data=loan_train, family = binomial)
summary(model2)
predict <- predict(model2, type = 'response') #matriks
kebingungan table(loan_train$Loan_sanctioned, predict > 0,5)
model3<- glm(Loan_sanctioned~ Income+Family,
data=loan_train, family = binomial) summary(model3)
predict <- predict(model3, type = 'response')
```

#matriks kebingungan

```
table(loan_train$Loan_sanctioned, predict > 0,5)
```

BAB 13
POHON KEPUTUSAN

13.1 PENDAHULUAN

Pohon keputusan merupakan metode yang ampuh untuk klasifikasi dan prediksi serta untuk memfasilitasi pengambilan keputusan dalam masalah keputusan berurutan. Kita dapat menggunakan tiga jenis pohon keputusan, yaitu:

1. Untuk merekomendasikan tindakan berdasarkan urutan simpul informasi,
2. Pohon Klasifikasi dan Regresi, dan
3. Pohon Kelangsungan Hidup

Namun, sebagian besar waktu kita menggunakan pohon keputusan untuk memecahkan masalah klasifikasi. Mari kita lihat beberapa Masalah dan pahami apakah itu masalah Pengelompokan atau masalah klasifikasi.

1. Jenis halaman di Web?
2. Apakah pinjaman pribadi harus disetujui atau tidak?
3. Jenis Pelanggan pusat perbelanjaan?
4. Apakah pelanggan ini akan membeli produk atau tidak?
5. Apa tempat berikutnya yang ingin dikunjungi wisatawan?
6. Apakah sinyal radio itu supernova, katai putih, atau raksasa merah?
7. Jenis orang di akun Facebook/LinkedIn Anda?
8. Jenis email di kotak masuk Anda?
9. Artikel ini tentang Olahraga, Hiburan, politik atau sains?
10. Jenis gen dalam Genom Manusia?

Apa perbedaan antara pengelompokan dan klasifikasi?

Setiap kali Anda tidak diberi target untuk diprediksi, Jika saya katakan ini adalah sekumpulan halaman web, saya hanya memberikan informasi itu, Maka kita dapat berpikir tentang struktur data, seperti jenis halaman apa saja yang ada, biar saya melakukan pengelompokan dan mencari tahu.

Pertanyaan berikutnya adalah berapa banyak jenis pelanggan yang ada, kita tidak yakin tentang ini. Jika kita tidak yakin tentang targetnya, Jika kita hanya mendapatkan data Input, X, dan bukan Y maka kita akan mencoba melakukan sesuatu seperti pengelompokan, PCA atau teknik pembelajaran tanpa pengawasan lainnya. Namun setiap kali Anda diberi Input dan Output untuk diprediksi, biar saya katakan ini adalah riwayat transaksi yang memprediksi Penipuan dan bukan Penipuan, Ini adalah email yang memberi tahu kita apakah itu promosi spam atau utama. Ketika saya mengajukan pertanyaan yang jelas, klasifikasikan ke dalam salah satu kategori, maka itu adalah Masalah klasifikasi. Bagaimana cara kerjanya di dunia nyata? Mari saya ambil contoh Uber Cab. Ketika mereka memulai bisnis, mereka mulai mendapatkan umpan balik dari pelanggan. Mereka tidak tahu jenis umpan balik apa yang mereka dapatkan. Jadi, mereka melakukan pengelompokan. Bayangkan mereka berkumpul di lima kluster.

Setelah mereka menemukan lima kluster, mereka akan menyiapkan proses untuk apa yang harus dilakukan untuk setiap kluster. Sekarang, kluster ini menjadi label kelas. Umpan balik berikutnya yang masuk harus dipetakan ke salah satu dari kelas tersebut. Jika tidak, maka harus menjadi kelas yang terpisah. Jika termasuk dalam salah satu grup yang sudah dibuat, maka Anda tahu apa yang harus dilakukan. Jika tidak, Anda perlu membuat proses di sekitarnya. Jadi, yang perlu kita pahami adalah, apakah strukturnya sudah ditemukan atau belum. Bayangkan keluhan pelanggan masuk ke satu alamat email. Lalu, yang perlu kita lakukan adalah mengklasifikasikannya. Kita dapat mengambil semua 100 ribu keluhan dan mengelompokkannya menjadi 5-6 jenis keluhan. Bagaimana Anda memetakan data mentah ke dalam jenis-jenis? Untuk menemukan jenis-jenisnya, kita perlu pengelompokan. Setelah kita mengetahui jenis-jenisnya, kita melakukan klasifikasi. kita bisa mendapatkan keluhan baru di luar jenis yang diberikan lalu kita membuat kelas baru.

Klasifikasi Deskriptif dan Diskriminatif: Ketika orang melihat data, separuh dari Anda melakukan satu hal dan separuh lainnya melakukan hal yang lain. Sebagian dari Anda mungkin membuat model untuk setiap kelas, seperti apa struktur dan bentuk kelas tersebut. Model ini disebut model deskriptif yang berarti Anda mendeskripsikan seperti apa tampilan data. Separuh lainnya berkata, hei, bagaimana cara membedakan keduanya. Saya tidak peduli tentang bentuk yang satu versus yang lain, saya peduli tentang batas yang satu versus yang lain. Tugas klasifikasi diskriminatif bukanlah mengkhawatirkan bentuk kelas, tetapi batasnya. Berpikirlah seperti klasifikasi deskriptif adalah kementerian dalam negeri, mereka peduli tentang bentuk negara di dalamnya. Seperti ini adalah dua negara, semua orang tertarik dengan negaranya, tetapi klasifikasi diskriminatif seperti kementerian pertahanan. Mereka lebih fokus pada batas. Keduanya melihat data yang sama tetapi dengan cara yang berbeda. Mari saya ambil dataset saya, saya telah melakukan rekayasa fitur, Sekarang jika saya ambil satu kelas, saya hanya mengurus satu kelas, menghitung mean dan matriks kovariansi kelas itu, dan melakukan hal yang sama untuk dua kelas lainnya maka saya memiliki tiga parameter ini. Ketika saya menghitung mean dan kovariansi satu kelas saya hanya mempertimbangkan kelas itu dan tidak peduli dengan kelas lainnya. Matriks kovariansi akan menggambarkan bentuk dan mean akan menggambarkan lokasi. Jika saya menggunakan jenis pengklasifikasi lain, Regresi logistik, maka ia akan mencoba memisahkan kelas-kelas. Di sini keduanya melakukan hal yang sama tetapi dengan cara yang berbeda.

13.2 APA ITU KLASIFIKASI?

Dalam klasifikasi, kita ingin mengklasifikasikan data, sedemikian rupa sehingga satu kelompok berisi titik-titik dari satu kelas, dan kelompok lain berisi titik-titik dari kelas lain. Kita ingin membagi ruang menjadi beberapa wilayah, yang harus murni berkenaan dengan label kelas saya. Mari saya ambil contoh India yang dibagi menjadi beberapa negara bagian, apa yang mereka gunakan sebagai kriteria. Mereka menggunakan Bahasa karena mereka ingin menghasilkan kesamaan sehingga titik-titik dalam satu kelompok serupa satu sama lain dan titik-titik dalam kelompok lain serupa satu sama lain.

Klasifikasi adalah membagi ruang (Fitur) menjadi wilayah murni yang ditetapkan untuk setiap kelas. Jika kita memiliki ruang seperti sekelompok titik, kita dapat memindahkan ruang tersebut, tetapi kita dapat memindahkan partisi tersebut. Kita memiliki ruang fitur, yang berasal dari semua rekayasa fitur yang telah Anda lakukan, kita memastikan bahwa mereka dinormalisasi, dll., Dalam ruang fitur tersebut, kita memiliki sekelompok titik data, dan kita ingin membuat batas keputusan. Batas ini akan membagi ruang menjadi wilayah yang lebih kecil, sekarang berapa banyak batas keputusan yang dapat saya pilih? Apa saja kemungkinan cara saya dapat membagi ruang tersebut? Jawabannya adalah Tak Terbatas. Saya dapat menggunakan jumlah cara yang tak terbatas untuk membagi ruang tersebut menjadi dua wilayah. Dari semua cara tak terbatas dari sejumlah partisi tersebut, saya harus menemukan satu partisi yang memenuhi kriteria, yang memaksimalkan kriteria tersebut.

Apa kriterianya? Kriterianya adalah bahwa setiap wilayah harus semurni mungkin, berkenaan dengan label kelas. tetapi kita tidak akan pernah mendapatkan partisi yang sempurna. Alasannya mungkin:

- Ruang fitur mungkin tidak lengkap, kita mungkin telah melewatkan fitur penting.
- Bisa jadi ada outlier,
- Noise pada Fitur,
- Model mungkin tidak sempurna,
- Noise pada Label, dengan semua wilayah ini kita mungkin tidak mendapatkan model yang sempurna.

Itulah alasan mengapa pekerjaan kami sebagai ilmuwan data sangat menantang. Kami memiliki daftar alasan mengapa model kami tidak berfungsi. Ketika kami mengklasifikasikan ruang ke dalam beberapa wilayah, dan masing-masing memiliki rangkaian kemurniannya sendiri. Beberapa kelas memiliki kemurnian lebih tinggi dibandingkan dengan yang lain. Apakah saya bisa melakukannya dengan lebih baik? Saya bisa membaginya menjadi wilayah yang lebih kecil dengan melihat lebih banyak fitur. Saya bisa membuat batasan nonlinier, saya akan membuat model yang lebih kompleks untuk melakukannya dengan lebih baik. Saya bisa melakukan banyak hal untuk meningkatkan pengklasifikasi ini. Bayangkan seseorang telah memberikan kumpulan data ini kepada Anda untuk dikerjakan, apa model paling sederhana yang dapat Anda pikirkan? Itu adalah batas keputusan dari garis lurus, tetapi apa masalahnya dengan batas sederhana, itu tidak semurni yang Anda inginkan. Jadi, idenya adalah kesederhanaan tidak berarti kemurnian atau akurasi. Apa yang dapat Anda lakukan sekarang? Jika saya menulis ini secara matematis, jenis model apakah ini? Sigma(W0+EWiXi) ini akan memberikan garis, dengan garis dengan tingkat kompleksitas model ini, Anda hanya dapat melakukan sebanyak ini. Hanya akan ada satu yang optimal jika saya membatasi Anda pada model linear.

Sekarang kita katakan tidak, Linear tidak cukup baik, kita butuh sesuatu yang lebih kompleks, apa yang harus kita lakukan sekarang? Kita ambil regresi Logistik dan buat lebih kompleks dengan menambahkan fitur nonlinear. Bukan hanya lebih banyak fitur, kita dibatasi oleh fitur tetapi dapatkah kita menggabungkan fitur-fitur tersebut bersama-sama. Sekarang saya dapat mengatakan sesuatu seperti Sigma(W0+EWiXi + EWijXiXJ) sekarang kita

mendapatkan model yang lebih kompleks, Berapa banyak parameter yang saya miliki sekarang? W0 --> 1, EWiXi -- > n, EWijXiXJ --> N pilih 2 yang berarti ketika saya meningkatkan kompleksitas dengan menambahkan lebih banyak parameter kita dapat membangun Batasan keputusan yang lebih kompleks.

Bayangkan situasi di mana kita membangun model dengan polinomial derajat 2 (Orde kedua kita memiliki kuadrat). Dan model lain dengan polinomial derajat 9 karena ada banyak infeksi hampir 9, jadi itu adalah polinomial derajat 9 yang bekerja dengan baik, dibandingkan dengan polinomial derajat pertama yang merupakan pengklasifikasi linier.

Dapatkah saya melakukannya dengan lebih baik? Bayangkan jika saya mengambil 100 derajat polinomial, Ada prinsip yang sangat penting yang perlu kita pahami di sini, sekarang pikirkan yang mana dari ketiga model ini yang merupakan model yang masuk akal? Di mana Anda akan berhenti? Jelas, tidak ada yang ingin menggunakan polinomial derajat 100 meskipun itu memberikan hasil yang lebih akurat.

Biarkan saya berbicara tentang prinsip umum yang tidak bergantung pada masalah. Prinsip kami mengatakan haruskah saya mengalahkan data hingga menyerah? Jika kita melakukan yang ketiga, kita tidak dapat menjelaskan modelnya, mengapa kita melakukan itu? Masalah utamanya adalah kita melatih data pelatihan secara berlebihan, yaitu kita menghafal data pelatihan, kita mungkin kehilangan kemampuan untuk menggeneralisasi karena kita telah menghafal. Saya mungkin ahli dalam kuis GK, tetapi saya mungkin tidak berpikir dengan benar, itu berarti saya memiliki ingatan yang baik tetapi tidak menggeneralisasi modelnya. Biasanya kita memilih model yang tidak terlalu sederhana atau terlalu rumit, di suatu tempat di tengah adalah model yang benar, dan seni ilmu data adalah untuk mencari tahu apa tingkat kompleksitas yang tepat dalam model. Saya dapat memberi Anda pohon keputusan atau regresi logistik, Anda masih dapat membuat model yang lebih kompleks atau kurang kompleks, di mana Anda berhenti adalah pilihan Anda. Ketika Anda melihat model, model harus melakukan hal yang benar, tidak boleh terlalu sederhana atau terlalu rumit. kita perlu memahami tradeoff. Ini yang kita sebut sebagai Rasio Sinyal terhadap Derau. Setiap data memiliki beberapa struktur (Sinyal) dan beberapa Derau. kita harus membuat model cukup kompleks untuk menangkap struktur tetapi bukan derau.

Generalisasi: Kemampuan untuk memprediksi atau menetapkan label pada observasi baru berdasarkan model yang dibangun dari pengalaman sebelumnya. Inilah yang kami sebut sebagai kompleksitas dan akurasi model. Yang biasa kami lakukan, kami mengambil data, pada data pelatihan jika kami terus meningkatkan kompleksitas model, akurasi meningkat hingga maksimum. Namun, itu bukan tujuan kami. Tujuannya adalah untuk melihat Validasi atau set pengujian yang merupakan data yang Tidak Terlihat, pada titik tertentu, hasil Validasi akan mulai menurun. Ini adalah titik di mana Anda harus berhenti meningkatkan kompleksitas karena tidak berkinerja dengan baik pada set data validasi. Jika kami melatih sesuatu secara berlebihan, model akan mengingat tetapi tidak akan belajar. Jadi, Pelatihan berlebihan adalah sesuatu yang perlu kami khawatirkan. Kami akan kembali ke hal ini, setiap kali kami mempelajari model.

Setiap kali Anda membuat model, kami perlu memahami apa harapan klien. Kami tertarik untuk membuat model umum yang akurat. Akurasi bukan hanya kriteria, tetapi kami perlu melihat interpretabilitas model. Jika model Anda terlalu rumit, model tersebut tidak dapat diinterpretasikan, bahkan saat itu pun itu menjadi masalah. Maka Anda mungkin memiliki model yang sangat akurat, model yang sangat tergeneralisasi, tetapi bukan model yang sangat dapat diinterpretasikan. Itu kriteria lainnya. Kriteria lainnya adalah penilaian waktu nyata.

Jika Anda memiliki model yang sangat rumit, yang memerlukan waktu satu detik untuk menilai satu titik data, Maka sekali lagi Anda perlu mengambil jalan pintas. Oleh karena itu, kita berhadapan dengan semua kendala ini. Saya menginginkan generalisasi yang baik, saya menginginkan interpretabilitas, dan hasil yang tinggi. Sekarang model apa yang harus saya Bangun? Itulah sebabnya, tidak jelas apa jawaban yang benar. Itu tergantung pada semua kendala ini.

Jika kita mengambil dua situasi di sini untuk memahami hal ini, Bayangkan kita sedang membangun model untuk deteksi penipuan, Kriteria untuk model tersebut adalah akurasi. Jika saya memutuskan untuk menghentikan kartu atau melepaskan kartu, saya tidak memiliki persyaratan untuk menjelaskannya kepada seseorang. Di sini interpretabilitas tidak penting, akurasilah yang Penting. Kemudian izinkan saya mengambil situasi lain di mana kita sedang membangun model untuk pemeringkatan kredit. ketika saya melakukan pemeringkatan kredit, setiap kali kami menolak pinjaman, kami harus memberikan tiga alasan utama mengapa kami menolak pinjaman tersebut. Di sini, interpretabilitas lebih penting dalam model pemeringkatan kredit. Oleh karena itu, model yang digunakan dalam model pemeringkatan kredit sangat berbeda dengan model yang digunakan dalam mendeteksi penipuan kartu kredit.

Kita perlu mengajukan pertanyaan berikut setiap kali kita membuat Model.

Apa Sifat Batasan Keputusan pengklasifikasi?
Apa Kompleksitas Batasan Keputusan pengklasifikasi?
Bagaimana cara mengendalikan kompleksitas pengklasifikasi?
Bagaimana cara mengetahui kapan pengklasifikasi cukup kompleks/Bagaimana cara memilih pengklasifikasi yang tepat untuk digunakan?

Model Berbasis Pohon: Partisi rekursif adalah alat dasar dalam penambangan data. Ini membantu kita menjelajahi struktur sekumpulan data, sambil mengembangkan aturan keputusan yang mudah divisualisasikan untuk memprediksi hasil kategoris (pohon klasifikasi) atau berkelanjutan (pohon regresi). Pohon klasifikasi dan regresi dapat dibuat melalui paket rpart.

13.3 LANGKAH DALAM MEMBUAT POHON KEPUTUSAN

Mengembangkan Pohon

```
rpart(formula, data=, method=,control=)where,
```

Formula dalam format outcome ~ predictor1+predictor2+predictor3+ect.
Data = menentukan kerangka data

Method = “class” untuk pohon klasifikasi, “anova” untuk pohon regresi
Control = parameter opsional untuk mengendalikan pertumbuhan pohon.
Misalnya, `control=rpart.control(minsplit=30, cp=0.001)` mengharuskan jumlah minimum observasi dalam sebuah simpul menjadi 30 sebelum mencoba pemisahan dan bahwa pemisahan harus mengurangi keseluruhan ketidaksesuaian dengan faktor 0,001 (faktor kompleksitas biaya) sebelum dicoba.

Memeriksa hasil

Fungsi-fungsi berikut membantu kita untuk memeriksa hasil. printcp(fit) #tampilkan tabel cp

`plotcp(fit)` #plot hasil validasi silang

`rsq.rpart(fit)` #plot perkiraan R-kuadrat dan galat relatif untuk pemisahan yang berbeda (2 plot).

label hanya sesuai untuk metode "anova".

`print(fit)` #cetak hasil.

`summary(fit)` #hasil terperinci termasuk pemisahan pengganti

`plot(fit)` #plot pohon keputusan

`text(fit)` #beri label plot pohon keputusan

`post(fit, file=)` #buat plot postscript dari pohon keputusan

Memangkas pohon

Prune mengembalikan pohon untuk menghindari overfitting data. Biasanya, Anda ingin memilih ukuran pohon yang meminimalkan galat validasi silang, kolom xerror yang dicetak oleh `printcp()`. Memangkas pohon ke ukuran yang diinginkan menggunakan `prune(fit, cp= )`. Secara khusus, gunakan `printcp()` untuk memeriksa hasil kesalahan yang divalidasi silang, pilih parameter kompleksitas yang terkait dengan kesalahan minimum, dan tempatkan parameter tersebut ke dalam fungsi `prune()`.

13.4 MEMBANGUN MODEL POHON KEPUTUSAN PADA DATASET PINJAMAN PERUMAHAN

Mari kita gunakan data Pinjaman Perumahan untuk memprediksi apakah pinjaman akan disetujui atau tidak berdasarkan Usia, Pengalaman, Pendapatan, Jumlah Keluarga, dan Tingkat Pendidikan.

Langkah 1: Instal dan Muat Paket yang diperlukan

```
install.packages(“rpart”)
library(rpart)
library(dummies)
?rpart
```

Langkah 2: Memuat Data ke R:

```
loandata=read.csv(file=”D:\\R data\\Housing_loan.csv”,
header=TRUE, sep=”,”)
fix(loandata)
```

Langkah 3: hapus kolom ID, kolom Gender dari data

```
loandata2=subset(loandata, select=-c(ID, Gender))
fix(loandata2)
```

Langkah 4: Buat variabel Dummy

```
Edu_dum =dummy(loandata2$Education)
loandata3=subset(loandata2,select=-c(Education))
fix(loandata3)
loandata4=cbind(loandata3,Edu_dum)
fix(loandata4)
```

Langkah 5: Standarisasi data menggunakan 'Range' metode

```
install.packages("vegan") library(vegan)
loandata_stan=decostand(loandata4,"range")
fix(loandata_stan)
```

Langkah 6: Perbaiki seed untuk mendapatkan data yang sama di setiap waktu

```
set.seed(123)
```

Langkah 7: Ambil sampel acak 60% dari rekaman untuk data train

```
train = sample(1:1000,600) loan_train = loandata_stan[train,]
```

Ambil sampel acak 40% dari rekaman untuk data uji

```
test = (1:1000) [-train] loan_test = loandata_stan[test,]
table(loandata_stan$Loan_sanctioned)
table(loan_train$Loan_sanctioned)
table(loan_test$Loan_sanctioned)
```

#Hapus objek yang tidak diperlukan

```
rm(loandata2, loandata3, loandata4,loandata_stan, Edu_dum, test,
train)
```

Langkah 8: Bangun Pohon Keputusan

```
fit <- rpart(Loan_sanctioned~ Age +Experience+ Income +Family+
Education1+Education2+Education3, data=loan_train,
method="class", control=rpart.control(minsplit=10, cp=0.001))
```

#rumus dalam format outcome ~ `predictor1+predictor2+predictor3+etc.`

#data= Menentukan kerangka data

#method= "class" untuk pohon klasifikasi, "anova" untuk pohon regresi #control= Parameter opsional untuk mengendalikan pertumbuhan pohon.

control= rpart.control(minsplit=10, cp=0.001) mengharuskan jumlah minimum pengamatan dalam sebuah simpul menjadi 10 sebelum mencoba #pemisahan dan bahwa pemisahan harus mengurangi keseluruhan ketidaksesuaian dengan faktor 0,001 (faktor kompleksitas biaya) sebelum dicoba.

Langkah 9: Tampilkan hasil

```
printcp(fit) # tampilkan hasil
plotcp(fit) # visualisasikan hasil validasi silang
summary(fit) # ringkasan terperinci dari pemisahan
```

Langkah 10: Plot pohon

```
plot(fit, uniform=TRUE, main="Pohon Klasifikasi untuk Pinjaman Perumahan") text(fit, use.n=TRUE, all=TRUE, cex=.8)
```

Langkah 11: Pangkas pohon

```
pfit<-prune(fit,cp=fit$cptable[which.min(fit$cptable[,"xerror"]),"CP"])
```

#Pangkas kembali pohon untuk menghindari data yang terlalu pas. Anda mungkin ingin memilih ukuran pohon yang meminimalkan kesalahan validasi silang, kolom xerror yang dicetak oleh printcp().

Pohon inferensi bersyarat dapat dibuat dengan menggunakan paket party yang menyediakan pohon regresi nonparametrik untuk respons nominal, ordinal, numerik, tersensor, dan multivariat. Anda dapat membuat pohon regresi atau klasifikasi melalui fungsi ctree(rumus, data=). Jenis pohon yang dibuat akan bergantung pada variabel hasil (faktor nominal, faktor terurut, numerik, dsb.). Pertumbuhan pohon didasarkan pada aturan penghentian statistik, jadi pemangkasan tidak diperlukan.

BAB 14
KLASIFIKASI K-NEAREST NEIGHBOR

14.1 PENDAHULUAN

Sebelum membahas KNN, mari kita pahami perbedaan antara Memori dan Pembelajaran. Memori adalah proses pencatatan, penyimpanan, dan pengambilan informasi. Dalam Memori, kita menyimpan data dan selalu mengaksesnya kembali kapan pun diperlukan. Pembelajaran adalah proses atau perilaku memperoleh pengetahuan. Pembelajaran bukan sekadar perolehan dan penyimpanan informasi, tetapi kemampuan untuk mengimplementasikan informasi dan memanfaatkannya dalam situasi praktis. Jika kita membangun model yang merupakan versi data yang jauh lebih terkompresi, maka kita akan melupakan data tersebut, kita akan menggunakan satu-satunya model di masa mendatang. Begitulah cara kerja otak kita. Jadi, kita dapat mengatakan pembelajaran adalah proses yang akan mengubah perilaku selanjutnya. Memori, di sisi lain, adalah kemampuan untuk mengingat pengalaman masa lalu.

K-Nearest Neighbor adalah algoritma sederhana yang menyimpan semua kasus yang tersedia dan mengklasifikasikan kasus baru dengan suara mayoritas dari k tetangganya. Algoritma ini memisahkan titik data yang tidak berlabel ke dalam kelompok yang terdefinisi dengan baik. Pengklasifikasi KNN adalah salah satu pengklasifikasi yang paling sederhana dan indah. Bayangkan diagram sebar dengan titik data kelas merah muda dan biru, di mana, titik data merah muda lebih menonjol di pojok kiri atas, dan biru lebih menonjol di pojok kanan bawah, tetapi ada banyak gangguan dalam data. seperti itulah data sebenarnya. Sekarang, jika saya memberi Anda titik data baru dan meminta Anda untuk mengklasifikasikannya. Apa yang akan Anda lakukan? Anda mulai mencari titik terdekat ke titik data baru yang diberikan berdasarkan jaraknya, Asumsikan yang terdekat adalah merah muda. Apa yang akan Anda lakukan? Anda akan menetapkan titik data baru ini ke merah muda. Ini disebut 1 Tetangga Terdekat.

Kira-kira seperti ini, Anda ingin menonton film, Anda menelepon salah satu teman Anda, dan bertanya apakah film itu bagus atau buruk. Apa pun yang dia katakan, Anda akan melakukannya. Apakah itu yang akan Anda lakukan? Tidak, lalu apa yang akan Anda lakukan? Kami menelepon lebih banyak orang, online dan membaca ulasan, Anda menonton cuplikannya dan kemudian Anda memutuskan, benar. Jadi proses pengambilan keputusan bergantung pada masukan eksternal, semakin banyak masukan yang Anda dapatkan, semakin kuat keputusan Anda. Daripada menggunakan 1 Tetangga Terdekat, apa yang dapat saya lakukan? Saya dapat menggunakan 2 tetangga terdekat. Sekarang saya menelepon 2 teman. Orang pertama berkata, lihat saja dan yang lain berkata jangan lihat. Sekarang saya bahkan lebih bingung. Jadi, dalam K tetangga terdekat, kami menghindari penggunaan K sebagai angka genap. Jika saya menggunakan 3 tetangga terdekat, angkanya akan berubah lagi berdasarkan jumlah poin mayoritas. Jika saya menggunakan lebih jauh lagi, Labelnya dapat berubah. dan seterusnya.

Bagaimana Anda mengendalikan kompleksitas di sini? Apa model sederhana dan model kompleks dalam K tetangga terdekat? Mereka mengambil data mentah yang berwarna merah muda dan biru. Dengan asumsi itu adalah contoh pengujian. Mereka menggunakan nilai K tertentu dan berkata jika saya menggunakan tiga tetangga terdekat, apa label saya? Merah muda atau biru?. Ini adalah ruang dan kami membaginya menjadi 2 Wilayah. Tidak seorang pun mengatakan wilayah harus bagus dan indah. Sekarang wilayah biru memiliki kemurnian dan titik apa pun yang termasuk biru akan ditetapkan ke wilayah biru.

Sekarang, apa yang terjadi ketika K meningkat. Beberapa titik telah berubah, seiring dengan peningkatan K, kita mendapatkan batas yang lebih halus dan model kita menjadi semakin kuat terhadap gangguan. Sebelumnya, model terlalu banyak merespons gangguan. Apa yang dipelajari di sini? Semakin tinggi K, semakin kuat modelnya dan semakin tidak rumit modelnya. Model tersebut tampak hampir seperti regresi linier. Kita perlu mempertimbangkan beberapa pertanyaan saat membangun pengklasifikasi. Apa saja parameternya? Apakah kita mempelajari parameter apa pun?

Dalam regresi logistik, apakah kita mempelajari parameter apa pun? Bobot adalah parameter saya. Hal-hal yang saya pelajari. Dalam K-means Clustering, pusat klaster adalah parameter, dan K adalah Hiperparameter. Dalam K Nearest Neighbor, K disebut hiperparameter. Itu adalah sesuatu yang Anda berikan kepada sistem untuk mengendalikan kompleksitas. Dalam kasus KNN, tidak ada parameter. Tidak ada Model. Oleh karena itu, dalam KNN waktu pelatihan adalah Nol karena tidak mempelajari apa pun. Itulah sebabnya model ini disebut sebagai model nonparametrik.

Apa kelemahan KNN ini? Berapa lama Anda perlu menilai titik data? Bayangkan jika saya memberi Anda 100 ribu titik data dalam set pelatihan, dan mengatakan ini adalah titik data baru yang masuk, apa yang perlu Anda lakukan? Anda perlu mengukur jarak dari semua 100 ribu titik, lalu mengurutkannya dan memilih K teratas, melihat apa labelnya, latihan itu adalah urutan N, dalam waktu pelatihan. Apakah Anda akan menggunakan pengklasifikasi seperti itu, untuk keputusan waktu nyata?. Saat memilih teknik Pemodelan, kita tidak hanya melihat keakuratannya, tetapi juga melihat waktu komputasi. Jika saya perlu membangun model yang perlu dilatih dengan sangat cepat, Data saya berubah sangat cepat, maka Anda memerlukan model yang tidak memiliki waktu pelatihan sama sekali. Kuncinya di sini adalah bagaimana Anda mendefinisikan fungsi jarak Anda. mendefinisikan fungsi jarak adalah tugas yang paling sulit. Bayangkan dua variabel seperti usia dan Pendapatan, bagaimana Anda mendefinisikan jarak antara keduanya?

Jarak antara dua profil LinkedIn, Jarak antara dua urutan gen, dll., dua sinyal suara, dua dokumen di web, dua tweet, dua film, dan seterusnya. Mari saya ambil contoh sederhana dari urutan gen kanker, kita memiliki sekitar 10 ribu urutan gen pasien kanker dan 1 juta urutan gen pasien non-kanker. Ketika titik data baru masuk, yang perlu kita putuskan adalah kita perlu memeriksa semua miliar urutan gen tersebut, mencocokkannya dengan semua urutan gen, dan menghitung jarak di antara mereka, menemukan K dan seterusnya, ini cukup sulit dilakukan. Jadi, saat memilih teknik pemodelan, kita perlu memikirkan, Apa itu K?, Apa itu Fungsi jarak?, Apakah saya dapat mencetak skor Lebih Cepat sekarang? semua hal ini.

Masalah lain dengan KNN adalah, jarak sebenarnya tidak akan diperhitungkan. Saya hanya mengatakan K Teratas. Saya tidak mengatakan Seberapa Jauh. Jadi, ini adalah satu masalah lagi dengan KNN karena kehilangan informasi jarak. Satu masalah lagi dengan KNN adalah tidak kuat terhadap derau. Kecuali jika Anda meningkatkan K terlalu banyak, itu tidak terlalu kuat terhadap derau.

14.2 BAGAIMANA CARA MEMILIH NILAI K YANG TEPAT?

Memilih jumlah tetangga terdekat, yaitu menentukan nilai k, memegang peranan penting dalam menentukan kemanjuran model. Dengan demikian, pemilihan k akan menentukan seberapa baik data dapat digunakan untuk menggeneralisasikan hasil algoritma kNN. Nilai k yang besar memiliki manfaat yang mencakup pengurangan varians karena data yang tidak jelas, efek sampingnya adalah mengembangkan bias yang menyebabkan pembelajar cenderung mengabaikan pola yang lebih kecil yang mungkin memiliki wawasan yang berguna.

Algoritma kNN – Kelebihan dan Kekurangan

Kelebihan: Algoritma ini sangat tidak bias dan tidak membuat asumsi awal tentang data yang mendasarinya. Karena sifatnya yang sederhana dan efektif, algoritma ini mudah diimplementasikan dan telah memperoleh popularitas yang baik.

Kekurangan: Algoritma kNN memang tidak membuat model karena tidak ada proses abstraksi yang terlibat. Ya, proses pelatihannya sangat cepat karena data disimpan kata demi kata, tetapi waktu prediksinya cukup tinggi dengan wawasan yang berguna yang terkadang hilang. Oleh karena itu, membangun algoritma ini memerlukan waktu untuk diinvestasikan dalam persiapan data guna memperoleh model yang kuat.

14.3 MEMBANGUN MODEL KNN PADA DATASET DIABETES

Mendeteksi Diabetes: Pembelajaran mesin banyak digunakan dalam industri farmasi, khususnya dalam mendeteksi Diabetes dan Kanker. Mari kita lihat proses membangun model ini menggunakan algoritma kNN dalam Pemrograman R.

Langkah 1: Pengumpulan data

Kita akan menggunakan set data pasien diabetes untuk mengimplementasikan algoritma KNN dan dengan demikian menginterpretasikan hasilnya. Set data terdiri dari 500 observasi dan 7 variabel sebagai berikut: Dalam kehidupan nyata, ada lusinan parameter penting yang diperlukan untuk mengukur probabilitas Diabetes, tetapi untuk tujuan penyederhanaan, mari kita bahas 7 di antaranya.

```
Pat_Id
Gender
OGTT
DBP
BMI
Age
Diabetic
```

Langkah 2: Eksplorasi Data

`setwd("D:/R data")` # Mari kita impor berkas data 'Diab.csv'. Perintah ini digunakan untuk menunjuk ke folder yang berisi berkas yang dibutuhkan.

`Diabdata <- read.csv("Diab.csv", header = TRUE,stringsAsFactors = FALSE)` #Perintah ini mengimpor kumpulan data yang dibutuhkan dan menyimpannya ke kerangka data Diabdata.

`stringsAsFactors = FALSE` #Perintah ini membantu mengonversi setiap vektor karakter ke faktor di mana pun masuk akal.

`str(Diabdata)` #Kita menggunakan perintah ini untuk melihat apakah data terstruktur atau tidak. Kita menemukan bahwa data terstruktur dengan 7 variabel dan 500 observasi. Jika kita mengamati kumpulan data, variabel pertama '`Pat_Id`' bersifat unik dan dapat dihapus karena tidak memberikan informasi yang berguna.

Langkah 3: Persiapan Data

Hapus variabel pertama (`Pat_Id`) dari kumpulan data. `Diabdata <- Diabdata[-1]`
Kumpulan data berisi pasien yang telah didiagnosis menderita Diabetes atau NonDiabetic #Sekilas atribut Diabetic melalui pembagian. `tabel(Diabdata$Diabetic)`
Variabel Diabetic adalah variabel target kita, yaitu variabel ini akan menentukan hasil diagnosis berdasarkan variabel numerik lainnya. #Jika Anda ingin memeriksa pembagian persentase atribut Diabetic, Anda dapat meminta tabel proporsi:

`round(prop.table(table(Diabdata$Diab)) * 100, digits = 1)` #Pemahaman Mendalam tentang Data Anda `summary(Diabdata)`
#Anda juga dapat menyempurnakan ringkasan ringkasan Anda dengan menambahkan atribut tertentu

```
summary(Diabdata[c("OGTT", "BMI")])
```

Langkah 4: Normalisasi:

Fitur ini sangat penting karena skala yang digunakan untuk nilai setiap variabel mungkin berbeda. Praktik terbaik adalah menormalkan data dan mengubah semua nilai ke skala umum. #Kita dapat melakukan normalisasi fitur, dengan terlebih dahulu membuat fungsi normalisasi Anda sendiri:

```
normalize <- function(x) { num <- x - min(x)
denom <- max(x) - min(x) return (num/denom)
}
Diab_norm <- as.data.frame(lapply(Diabdata[3:6], normalize))
summary(Diab_norm)
fix(Diab_norm)
```

Mari kita periksa menggunakan variabel 'BMI' apakah data telah dinormalisasi.

```
summary(Diab_norm$BMI)
```

Langkah 5: Membuat set data pelatihan dan pengujian:

Algoritma kNN diterapkan pada set data pelatihan dan hasilnya diverifikasi pada set data pengujian. Untuk ini, kita akan membagi set data menjadi 2 bagian dengan rasio 60: 40 untuk set data pelatihan dan pengujian. Anda dapat menggunakan rasio yang berbeda sama sekali tergantung pada persyaratan bisnis. #Set Pelatihan dan Pengujian

```
set.seed(1234)
Indicator <- sample(2, nrow(Diab_norm), replace=TRUE,
prob=c(0.6, 0.4))
```

#Kita kemudian dapat menggunakan sampel yang disimpan dalam variabel Indicator untuk menentukan set pelatihan dan pengujian Anda:

```
Diab.training <- Diab_norm[Indicator==1, 1:4] Diab.test <-
Diab_norm[Indicator ==2, 1:4]
```

Variabel target kita adalah 'Diabetic' yang belum kita sertakan dalam set data pelatihan dan pengujian kita.

```
Diab.trainLabels <- Diabdata[Indicator ==1, 7]
Diab.testLabels <- Diabdata[Indicator ==2, 7]
```

Langkah 6: Melatih model pada data:

Fungsi `knn()` perlu digunakan untuk melatih model yang memerlukan paket 'class' untuk diinstal. Fungsi `knn()` mengidentifikasi k-tetangga terdekat menggunakan jarak Euclidean, di mana k adalah angka yang ditentukan pengguna.

```
install.packages("class") library(class)
```

Sekarang kita siap menggunakan fungsi `knn()` untuk mengklasifikasikan data uji. Mari kita Bangun Pengklasifikasi. Untuk membangun pengklasifikasi, kita perlu mengambil fungsi `knn()` dan menambahkan beberapa argumen ke dalamnya.

```
Diab_pred <- knn(train = Diab.training, test = Diab.test, cl =
Diab.trainLabels, k=11) Diab_pred
```

Nilai untuk k umumnya dipilih sebagai akar kuadrat dari jumlah observasi. `knn()` mengembalikan nilai faktor label yang diprediksi untuk setiap contoh dalam set data uji yang kemudian ditetapkan ke kerangka data `Diab_pred`. Hasil dari perintah ini adalah vektor faktor dengan kelas yang diprediksi untuk setiap baris data uji.

Langkah 7: Evaluasi Model Anda:

Untuk memeriksa kinerja model, Kita dapat mengimpor paket gmodels:

```
install.packages("gmodels")
```

#Jika Anda telah menginstal paket ini, Anda cukup memasukkan

```
library(gmodels)
```

#Kemudian kita membuat tabulasi silang atau tabel kontingensi.

```
CrossTable(x = Diab.testLabels, y = Diab_pred, prop.chisq=FALSE)
```

Tidak ada kasus Negatif Palsu (FN) yang berarti tidak ada kasus yang tercatat yang sebenarnya bersifat Diabetik tetapi diprediksi sebagai NonDiabetik. FN jika ada menimbulkan potensi ancaman karena alasan yang sama dan fokus utama untuk meningkatkan akurasi model

adalah mengurangi FN. Akurasi total model adalah 60 % ((TN+TP)/35) yang menunjukkan bahwa mungkin ada peluang untuk meningkatkan kinerja model

Langkah 8: Meningkatkan kinerja model

Hal ini dapat diperhitungkan dengan mengulangi langkah 3 dan 4 dan dengan mengubah nilai k. Umumnya, nilai tersebut adalah akar kuadrat dari pengamatan dan dalam kasus ini, kami mengambil k=10 yang merupakan akar kuadrat sempurna dari 100. Nilai k dapat berfluktuasi dalam dan di sekitar nilai 10 untuk memeriksa peningkatan akurasi model. Cobalah dengan nilai pilihan Anda untuk meningkatkan akurasi. Kita perlu mengingat hal itu untuk menjaga nilai FN serendah mungkin.

BAB 15
PENGKLASIFIKASI BAYESIAN

15.1 PENDAHULUAN

Teorema Bayes adalah teorema matematika yang digunakan untuk mencari tahu hubungan antara data dan kelas. Kita perlu memikirkan jika ini adalah data, maka apa yang seharusnya menjadi kelas. Di sini kita mencoba melakukan pemetaan antara data dan kelas.

1. Berapa probabilitas pelanggan akan merespons dengan tawaran dan kupon ini.
2. Berapa probabilitas Anda menderita kanker atau tidak kanker dengan urutan gen tertentu?

Mari saya bahas aturan Bayes. Mari kita pahami mengapa hal ini menjadi hal yang penting.

$$P(\text{Class}|Data) = \frac{P(Class)P(Data|Class)}{P(Data)}$$

P(Class) → Kelas Prior
P(Data|Class) → Kemungkinan Data yang diberikan Kelas
P(Data) → Data Prior(Marginal)
P(Kelas|Data) → Probabilitas Posterior(Probabilitas kelas setelah melihat data)

Mari saya bahas tentang sifat yang sangat sederhana dari distribusi probabilitas gabungan. Jika saya katakan, P(a,b), saya dapat menuliskannya dalam dua jenis, sebagai P(a) dikali p(b jika a) atau P(b) dikali p(b jika a). Mari kita pahami ini dengan sebuah contoh. Bayangkan Anda seorang dokter, Anda sedang menunggu pasien berikutnya datang. Apakah Anda sudah tahu apa penyakitnya? Bisakah Anda menebak apa penyakitnya? Apa pun yang kita ketahui tanpa melihat data, kita sebut itu sebagai prior. Ketika saya meminta untuk mewawancarai seorang kandidat di sebuah organisasi perangkat lunak, tanpa melihat kandidat tersebut, tanpa melihat resume-nya, apa yang dapat saya katakan apakah kandidat tersebut akan dipekerjakan atau tidak? Bagaimana saya melakukannya? Saya melihat semua orang, yang menghadiri wawancara dan berapa fraksi dari mereka yang dipekerjakan. Ini disebut probabilitas prior.

Dokter yang sama telah melihat banyak pasien di masa lalu dan mendiagnosis mereka dengan benar, maka kita memiliki ekspektasi pada titik data untuk mengklasifikasikan mereka ke dalam kelas tersebut (Kemungkinan data memberikan kelas). Kemudian saya perlu mempertimbangkan bahkan berapa probabilitas pasien seperti itu akan datang kepada saya. (Data Prior Marginal). Berdasarkan informasi ini, saya perlu memprediksi masa depan. Dengan titik data baru ini, beri tahu saya berapa probabilitas kelas ini. Itulah yang coba kita lakukan. Jadi, teorema Bayes bukanlah sulap matematika probabilitas bersyarat. Ini benar-benar cara untuk menghubungkan pengalaman masa lalu Anda dengan prediksi masa depan Anda. Ada

banyak sekali teknik pemodelan hanya berdasarkan teorema Bayes. Sebagian besar pembelajaran mesin menghabiskan waktu pada Data Likelihood yang diberikan kelas. Kemungkinan Maksimum berarti berapa kemungkinan data ini berasal dari kelas ini, dan kami hanya mencoba untuk memaksimalkannya. Ada dua cara untuk memikirkan teorema Bayesian.

1. Keputusan Kemungkinan Maksimum
2. Keputusan probabilitas posterior Maksimum.

15.2 BERPIKIR SEPERTI SEORANG BAYESIAN

Ini hanya menerapkan teori probabilitas kita dalam pengambilan keputusan. Saya akan memberi Anda sebuah skenario, kita akan mengubah kalimat-kalimat bahasa Inggris tersebut menjadi probabilitas dan probabilitas bersyarat dan kita akan menghitung probabilitas posterior. Bayangkan Anda adalah produsen mesin tersebut, Ambil seratus orang yang Anda tahu menderita kanker dan ujilah mereka, jika pernyataan tersebut mengatakan mesin tersebut akan memberikan hasil positif yang Benar sebanyak 95% dari waktu.

```
P(Uji_positif|Mengidap_Kanker)=.95, P(Uji_Negatif|Mengidap_Kanker) =.05
```

Itu berarti meskipun mesin tersebut mengatakan negatif dan Anda menderita kanker, itu tidak berarti Anda tidak menderita kanker. Apa yang dimaksud dengan data dan Kelas di sini? Hasil uji adalah data, Kanker tidak ada kanker adalah outputnya.

Bayangkan Anda adalah produsen mesin tersebut, Ambillah seratus orang yang Anda tahu tidak menderita kanker dan ujilah mereka. Jika mesin tersebut memberikan hasil negatif yang Benar sebanyak 90% dari waktu.

```
P (Tes_Negatif | Tidak_Kanker) = 0,9, P (Tes_Positif | Tidak_Kanker) = 0,1
```

Dan, hanya 0,8% dari seluruh populasi yang menderita kanker:

```
P(Mengidap_Kanker) = 0,008, P (Tidak_Kanker) = 0,992
```

Di sini kita berurusan dengan dua variabel: 1. apakah pasien menderita kanker atau tidak, 2. Hasil Tes Positif atau Negatif.

Sekarang, saya ingin bertanya, berapa probabilitas awal bahwa tesnya positif? Untuk menjawabnya, kita perlu menghitung dua hal: 1. pasien menderita kanker, apakah mesin menunjukkan positif atau tidak, 2. pasien tidak menderita kanker, apakah mesin menunjukkan positif atau tidak.

```
= [P (Tes_positif | Menderita_Kanker) * P(Menderita_Kanker) + P
(Tes_Positif | Tidak_Kanker) * P (Tidak_Kanker) ]
=0,95*0,008 + 0,1*0,992 = 0,1068
```

Berapa probabilitas awal bahwa tes tersebut negatif?

```
= [P (Tes_Negatif | Menderita_Kanker) * P(Menderita_Kanker) + P
(Tes_Negatif | Tidak_Kanker) * P (Tidak_Kanker) ]
=0,05*0,008 + 0,9*0,992 = 0,8932
```

Beginilah cara kami berpikir seperti seorang Bayesian. Saya tidak dapat mengukur berapa kali hasilnya akan positif. Itu tergantung pada jenis pasien yang diuji. Kita tidak dapat hanya mengamati data dengan mengabaikan label kelas.

Mari saya ambil contoh, saya ingin memprediksi apakah Anda akan bersin hari ini? Tergantung pada apa penyebab bersin dan apakah Anda akan mengungkap penyebab tersebut, barulah kita dapat menentukan apakah Anda bersin atau tidak.

Sekarang, izinkan saya mengajukan pertanyaan lain. Jika seorang pasien baru datang dan hasil tesnya positif, kita terapkan teorema Bayes. Berapa probabilitas kanker jika hasil tesnya positif?

```
P (Has_Cancer | Test_Positive)
= [P(Test_positive|Has_cancer)*P(Has_Cancer)/P(Test_Positive)]
= [ (0,95 * 0,008) / 0,1068]
= 0,07116
```

Berapa probabilitas kanker jika hasil tesnya negatif?

```
P (Has_Cancer | Test_Negative)
= [P(Test_Negative|Has_cancer)*P(Has_Cancer)/P(Test_Negattive)]
= [ (0,05 * 0,008) / 0,8932]
= 0,00045
```

Bahkan bukti mengatakan bahwa pasien tersebut menderita kanker, tetapi kita tidak perlu khawatir karena priornya sangat rendah. Untuk menghitung NaiveBayes, kita perlu mengetahui cara menghitung Mean dan kovarians serta perhitungan probabilitas Bayes.

Jarak Mahalanobis: Ketika Anda mengambil data, Ambil PCA dari data, apa yang Anda lakukan ketika Anda mengambil PCA dari data, Anda pada dasarnya menghilangkan kovarians di dalamnya dan kemudian Anda menghitung jaraknya, sehingga menjadi Jarak Mahalanobis. Batas Keputusan Pengklasifikasi Bayesian: Ingat bahwa kita membahas 2 jenis pengklasifikasi pada bab sebelumnya, 1. Bentuk kelas 2. Batas keputusan. Dalam Batas Keputusan Pengklasifikasi Bayesian, kami menggunakan keduanya. Yang kami lakukan adalah membuat satu Gaussian untuk setiap kelas, dan jika keduanya sama, di situlah batas keputusan kami.

15.3 NAIVE BAYES

NaiveBayes adalah kelas pengklasifikasi Bayesian yang sangat penting.

Kemandirian Bersyarat: kita mengetahui Kemandirian antara dua variabel. Jika saya katakan dua peristiwa terjadi dan keduanya benar-benar acak, apa yang dapat kita katakan tentang probabilitas gabungan? P(A, B)= P(A)* P(B).

Misalkan saja kita masuk ke ruang dokter dan kita mengatakan kita demam dan nyeri tubuh. Menurut Anda, apakah kedua gejala ini independen? Tidak, ada penyebab umum untuk

kedua gejala di atas. Dokter mencoba mencari tahu penyebab umum dari semua gejala independen. Katakanlah infeksi virus menyebabkan kedua gejala ini. Jadi, Infeksi virus adalah penyebabnya. Kemudian Kita dapat menulis pernyataan ini seperti ini: `P (Demam, Nyeri tubuh | Virus) = P (Demam | Virus) * P (Nyeri tubuh | Virus)` inilah yang disebut Kemandirian Bersyarat. Kembali lagi ke teorema Bayes, saya ingin mencari tahu probabilitas Infeksi Virus jika pasien mengalami Demam dan Pegal-pegal.
`P (Virus | Demam, Pegal-pegal) = [P (Demam, Pegal-pegal | Virus) * P (Virus)] / P(Demam, Pegal-pegal).`

15.4 MEMBANGUN NAIVE BAYES DARI DATASET DIABETES

Langkah 1: Dapatkan Data Anda:

#load dalam set data dengan perintah berikut:

```
setwd("D:/R data")
Diabds <- read.csv("Diab.csv", header = TRUE) fix(Diabds)
```

Langkah 2: Ketahui Data Anda:

Adalah ide yang lebih baik untuk memeriksa set data dengan menjalankan

```
head(Diabds) str(Diabds) names(Diabds) summary(Diabds)
```

Langkah 3: Siapkan Data Anda:

Hapus variabel pertama(Pat_Id) dari set data.

```
Diabdata <- Diabds[-1]
```

Langkah 4: Muat Paket yang diperlukan:

Instal dan Muat paket e1071 `install.packages('e1071',dependencies=TRUE) library(e1071)`

Langkah 5: Pisahkan kumpulan data

```
Diab_test = sample(1:nrow(Diabdata),200) Diab_train =
setdiff(1:nrow(Diabdata),Diab_test)
```

Langkah 6: Bangun Model

```
Diab_nb =
naiveBayes(Diabdata[Diab_train,2:5],Diabdata[Diab_train,6])
Dia_res = predict(Diab_nb,Diabdata[Diab_test,2:5])
```

Langkah 7: Tampilkan matriks kebingungan

```
table(Dia_res,Diabdata[Diab_test,6])
```

Langkah 8: Hitung akurasi

```
cm_Diab = tabel(Dia_res,Diabdata[Diab_test,6])
```

#hitung jumlah nilai diagonal dalam matriks

```
benar = jumlah(diag(cm_Diab)) akurasi = benar / jumlah(cm_Diab)
```

BAB 16
JARINGAN SARAF

16.1 PENDAHULUAN

Jaringan Saraf (NN) memiliki neuron dan lapisan dalam arsitekturnya. 3 lapisan dalam Jaringan Saraf adalah Lapisan Input, Lapisan Tersembunyi, dan Lapisan Output. Setiap lapisan ini memiliki unit atau neuron di dalamnya.

Jaringan Saraf Manusia: Unit komputasi dasar dalam sistem saraf adalah sel saraf atau neuron. Neuron memiliki 1. Dendrit (input) 2. Badan sel 3. Akson (output). Neuron menerima input dari neuron lain, Input dijumlahkan dan, Setelah input melebihi level kritis, neuron melepaskan pulsa listrik yang berjalan melalui badan, menuruni akson, ke neuron berikutnya. Peristiwa lonjakan ini juga disebut depolarisasi dan diikuti oleh periode refraktori, di mana neuron tidak dapat menyala. Ujung akson hampir menyentuh dendrit atau badan sel neuron berikutnya. Transmisi sinyal listrik dari satu neuron ke neuron berikutnya dilakukan oleh neurotransmitter, zat kimia yang dilepaskan dari neuron pertama dan mengikat reseptor di neuron kedua. Hubungan ini disebut sinaps. Sejauh mana sinyal dari satu neuron diteruskan ke neuron berikutnya bergantung pada banyak faktor, misalnya jumlah neurotransmitter yang tersedia, susunan reseptor, jumlah neurotransmitter yang diserap kembali, dll.

Mari kita pahami algoritma EM. Dalam pengelompokan K-means, yang kita lakukan adalah menetapkan kluster ke titik data dan menghitung rata-ratanya. Setelah menghitung rata-rata, kita menetapkan kembali titik data. Karena kita tidak mengetahui modelnya, kita mulai dari suatu tempat dan bergerak maju dari sana.

Aplikasi jaringan saraf:

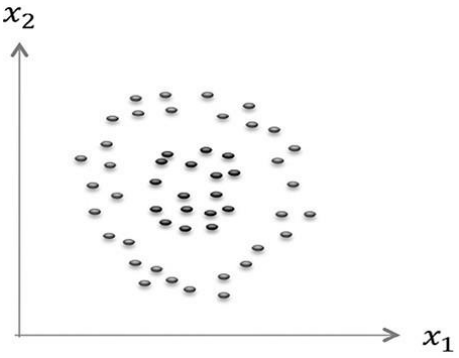
1. Robotika – Navigasi, Pengenalan Penglihatan
2. Kedokteran – Menyimpan catatan medis
3. Pengenalan Ucapan
4. Prediksi pasar saham
5. Kompresi data
6. Pemrosesan gambar
7. Pengenalan wajah
8. Pelacakan posisi kabin atau truk
9. Pemrosesan Sinyal
10. Mengenali karakter tulisan tangan.

16.2 MENGAPA JARINGAN SYARAF?

Jaringan saraf adalah algoritma terobosan dalam pembelajaran mesin. Pertama, mari kita pahami mengapa regresi logistik tidak cukup? Jika terdapat lebih dari 2 kelas, sejauh ini kami telah menyarankan untuk melakukan hal berikut: Tetapkan satu simpul keluaran untuk setiap kelas, Tetapkan nilai target setiap simpul menjadi 1 jika merupakan kelas yang benar dan 0 jika tidak, Gunakan jaringan linier dengan fungsi kesalahan kuadrat rata-rata. Ada

beberapa masalah dengan metode ini. Pertama, terdapat kesenjangan antara definisi fungsi kesalahan dan penentuan kelas. Kesalahan minimum tidak selalu menghasilkan jaringan dengan jumlah prediksi benar terbesar.

Interpretasi Baru: Keluaran yi diinterpretasikan sebagai probabilitas bahwa i merupakan kelas yang benar. Ini berarti bahwa keluaran setiap simpul harus berada di antara 0 dan 1. Jumlah keluaran semua simpul harus sama dengan 1.



Mari kita ambil masalah XOR, Gerbang EXCLUSIVE-OR adalah gerbang dua masukan, satu keluaran, ini adalah contoh klasik mengapa kita memerlukan jaringan saraf. XOR. Ambil saja mekanisme sakelar dua arah di rumah kita, jika kita menghidupkan atau mematikan sakelar, sakelar dianggap mati. Jika salah satu menyala, sakelar dianggap menyala. Dikatakan jika keduanya nol atau keduanya satu, kita mendapatkan nilai X dan salah satu nol/Satu, kita mendapatkan nilai Y. Gerbang XOR menghasilkan 0 saat kedua masukan cocok. Saat mencari pola bit tertentu atau urutan PRN dalam urutan data yang sangat panjang, serangkaian gerbang XOR dapat digunakan untuk membandingkan serangkaian bit dari urutan data terhadap urutan target secara paralel. Jumlah keluaran 0 kemudian dapat dihitung untuk menentukan seberapa baik urutan data cocok dengan urutan target.

Berapa banyak garis yang perlu kita buat untuk membedakan titik data seperti ini? Minimal dua karena dengan satu saya tidak dapat sepenuhnya membedakan data. Di sini kita perlu memahami bahwa regresi logistik tidak cukup karena kompleksitas data lebih besar daripada kompleksitas model.

Sekarang, garis seperti apa yang dapat saya miliki?

Apakah saya memiliki solusi yang unik?

Kita dapat memperoleh solusi yang berbeda. Sebenarnya ada empat solusi berbeda untuk ini. Salah satunya diberikan dalam gambar, setidaknya dengan dua garis ini, saya dapat melakukan sesuatu. Apa pun yang kita lakukan, pada akhirnya yang saya lakukan adalah saya membagi ruang menjadi beberapa wilayah murni.

Bagaimana pun saya melakukannya, apakah saya melakukannya dengan memperbaiki Gaussian, atau campuran Gaussian, dengan dua Gaussian per kelas, Bayangkan jika saya harus menyelesaikan masalah yang sama dengan pengklasifikasi deskriptif, apa yang akan saya lakukan, apakah satu Gaussian cukup per kelas? Tidak, satu Gaussian tidak cukup. Dalam kedua kasus, baik Anda menggunakan deskriptif atau diskriminatif, keduanya independen, sifat data mengatakan, satu Gaussian tidak cukup dan satu garis tidak cukup. Jadi, oleh karena itu, saya memerlukan dua Gaussian atau dua garis dalam pengklasifikasi diskriminatif. Ketika kita mengatakan ada beberapa minimum lokal, yang kita maksud adalah intinya adalah satu minimum lokal, yang berarti itu akan memberi saya jawaban yang bagus, jika Anda melihat yang lain, itu adalah serangkaian parameter lain yang juga akan memberi saya daerah yang

sangat murni, itu adalah minimum lokal lainnya. Jadi, kita dapat memahami bahwa ada beberapa solusi untuk masalah ini. Itulah yang membuat algoritme semacam ini menantang.

Mari saya ambil satu persamaan per baris, Sekarang saya punya dua baris, apakah itu cukup? Baris pertama hanya mengatakan apakah Anda berada di sisi ini atau sisi itu. Itu tidak mengatakan apa kelasnya. Baris di atas juga mengatakan apakah Anda berada di sisi ini atau sisi itu. Itu bahkan tidak mengatakan apa kelasnya. Kita butuh seseorang untuk mendengarkan kedua baris dan menggabungkan output, lalu mengatakan apa kelas akhirnya. Seperti itulah bentuk jaringan saraf. Ini adalah jaringan saraf yang paling sederhana, ini adalah satu neuron yang sama dengan satu garis, ini adalah neuron lain yang merupakan garis lain, dan lihat bobotnya, garis ini menyatakan 2 kali X1 ditambah 2 kali X2 dikurangi 1 yang menghasilkan satu garis. Garis lainnya dapat dijelaskan dengan cara yang sama. Poin di atas seperti manajer tim, mendengarkan semua orang dan mengambil output mereka, dan menimbangnya sesuai dengan itu dan mendorong mereka maju. Ini adalah keputusan akhir yang diambilnya. Ingat kita membahas Hierarki, seperti setiap insinyur melakukan satu hal, dan kita membutuhkan seseorang untuk menggabungkan output mereka dan memberikan keputusan akhir.

Sekarang kita mengerti mengapa kita membutuhkan jaringan saraf. Mari kita pelajari cara menafsirkannya. Jaringan saraf tidak lain hanyalah lebih dari satu regresi logistik dan penggabung di atasnya. Ini juga disebut Perceptron multilayer. Mari kita berpikir dengan cara ini seperti jika titik data baru berada di atas garis pertama dan di bawah garis kedua, maka kita katakan itu Merah. jika tidak, Anda adalah Biru. Umumnya ketika kita membuat aturan dan keputusan, itu adalah sekumpulan Pernyataan Dan. Mari kita bahas sekali lagi, misalnya jika Anda berada di sisi garis ini dan sisi garis lainnya, lalu apa Jawabannya? Mari kita lihat garis yang berarti Anda berada di atas garis bawah. Dan di bawah garis atas, jika kedua kondisi ini benar, maka Anda berwarna merah. Dalam sistem berbasis aturan, ini diberikan kepada Anda, Anda tidak mempelajarinya, dan kita hanya membuat aturan dari pernyataan tersebut. Namun dalam jaringan Neural, kita membuat aturan dengan cara ini. Dalam pohon keputusan, seluruh jalur juga merupakan aturan. Mari saya hubungkan ini dengan situasi perawatan kesehatan, jika Anda mengalami Kedinginan dan Demam, masalahnya mungkin Demam Malaria.

Jaringan saraf juga disebut sebagai Perceptron berlapis-lapis. Jika saya memiliki data mentah di mana satu kelompok titik tertanam dalam kelompok titik lain, Sekarang beri tahu saya berapa banyak baris yang saya butuhkan? Sekarang saya butuh setidaknya tiga atau lebih. Apa yang kita katakan? Setiap unit tersembunyi adalah baris. Cara penafsirannya adalah setiap kali kita menambahkan unit, itu adalah satu baris, unit tersembunyi lainnya adalah baris lain dan seterusnya. Bergantung pada kompleksitas masalahnya, saya dapat menambahkan lebih banyak unit tersembunyi dan itu dapat memberi saya lebih banyak baris. Dan kemudian kita menambahkan konstanta. Ingat apa yang kita katakan sebelumnya tentang neuron, sifat neuron tidak berubah, inputnya dapat berubah. Neuron di tengah mengambil info dari dua unit kiri dan ambang batas ada di atas. Aktivasi dan sakelar membentuk neuron. Neuron paling kanan tidak mengambil input mentah. Ini adalah model linier umum, ia mengambil titik tengah

sebagai input. Ia tidak mengetahui apa pun tentang data mentah, yang berarti peran setiap neuron sangat jelas, mereka tahu apa yang seharusnya dilakukan dan apa yang tidak.

Elemen komputasi dasar kita (neuron model) sering disebut simpul atau unit. Ia menerima masukan dari beberapa unit lain, atau mungkin dari sumber eksternal. Setiap masukan memiliki bobot terkait w, yang dapat dimodifikasi untuk memodelkan pembelajaran sinaptik. Unit tersebut menghitung beberapa fungsi f dari jumlah tertimbang masukannya:

$$y_i = f\left(\sum_j w_{ij} y_j\right)$$

- Jumlah tertimbang $\left(\sum_j w_{ij} y_j\right)$ disebut **input neto** ke unit i, sering ditulis neti.
- Perhatikan bahwa w_{ij} merujuk pada bobot dari unit j ke unit i (bukan sebaliknya).
- Fungsi f adalah fungsi aktivasi unit. Dalam kasus paling sederhana, f adalah fungsi identitas, dan output unit hanyalah input netonya. Ini disebut unit linear.

Jika saya memberi Anda arsitektur jaringan neural, Katakanlah saya memiliki D input, dan kita selalu memiliki konstanta X0 yang selalu 1. Selalu ada suku konstan per lapisan. dan kita memiliki H unit tersembunyi, dan kelas C di bagian akhir. Mari kita pahami dalam kasus Data Diabetes di mana input dan output tetap. Kita tidak dapat melakukan apa pun tentang hal itu. Apa yang dapat kita ubah jumlah unit tersembunyi dan jumlah lapisan tersembunyi? Itu memberi Anda kompleksitas. Jika kompleksitas berlanjut ke tingkat tertentu, akurasi meningkat.

Sekarang beri tahu saya berapa banyak bobot yang sedang saya pelajari. untuk satu unit tersembunyi, saya memiliki W0 dan jumlah Input yang berarti bobot D+1. Lalu, berapa banyak unit tersembunyi yang saya miliki. H hidden units. yaitu jumlah parameter dari sini ke sini, untuk hidden unit di lapisan berikutnya, dihitung sekali lagi, dengan cara yang sama, untuk end class, dihitung seperti (H+1)C. jadi jawaban akhirnya adalah (D+1) H + (H+1)C. dengan cara ini kita dapat menghitung parameter untuk kompleksitas apa pun. Biar saya jelaskan lagi dengan kata-kata sederhana, Jaringan Neural terhubung sepenuhnya, pikirkan neuron pertama, terhubung ke semua input D, setiap input memiliki bobot, pikirkan ini sebagai g(W0+W1X1 +... WdXd) ini adalah regresi logistik. ini memberi saya output dari satu neuron.

Berapa banyak parameter yang ada? berapa derajat kebebasannya? Berapa banyak parameter yang dipelajarinya?. Ia mempelajari (D+1) dan itu berlaku untuk unit lain, karena semuanya mempelajari bobot D+1, jadi (D+1)H. Sekarang logika yang sama diterapkan ke lapisan berikutnya, dan seterusnya.

Mari kita ambil C unit output, di sini inputnya akan menjadi unit tersembunyi dari lapisan sebelumnya (H+1) dan saya memiliki C unit tersebut. Jadi C(H+1). Dengan cara ini kita dapat menghitung sejumlah parameter untuk kompleksitas apa pun.

Misalnya kita memiliki 10.000 piksel di retina Anda, itu adalah input Anda, beri tahu saya berapa banyak kemiringan garis yang dapat Anda beri tahu. Dan bayangkan sebuah jam tangan, berapa banyak kemiringan yang dapat kita deteksi secara visual, yaitu 60. Kita bahkan

dapat melakukan gradasi halus, tetapi katakanlah 60. Itu berarti dari 10.000 input Anda mendapatkan 60 unit tersembunyi. Sekarang jika saya katakan, tidak hanya garis lurus, bagaimana dengan kurva, Anda dapat mendeteksi semua jenis kurva, Anda dapat mendeteksi kurva dengan orientasi yang berbeda, saya dapat terus dan terus dan saya dapat memiliki lebih dari jumlah input, sama halnya di lapisan berikutnya, jika saya katakan lingkaran, oval, segitiga, dan lainnya, jika kita naik dalam hierarki itu tidak berarti Anda mengurangi jumlah fitur. Itu sebenarnya ledakan kombinatorial. Dengan sejumlah kecil hal, tingkat kombinasinya terus meningkat. Inilah yang sebenarnya dilakukan jaringan dalam. Jaringan ini menangani begitu banyak kompleksitas.

Itulah yang dikatakan alam, sifatnya sangat kompleks, jika hewan atau manusia harus memahami kompleksitas sebanyak ini, Kita memerlukan jaringan saraf yang sangat kompleks, tetapi jaringan ini tidak dapat menciptakan neuron yang berbeda, jadi dikatakan Bagaimana saya dapat menciptakan jaringan yang kompleks, dengan blok penyusun yang sama dan yang harus dilakukannya adalah mengubah blok penyusunnya. Itulah keindahan jaringan saraf. Setiap neuron masih melakukan hal yang sama. Yang dilakukannya hanyalah kombinasi linier dari masukannya, diikuti oleh fungsi logistik. Semua miliar neuron melakukan hal yang persis sama. Dengan kesederhanaan sebanyak itu, Anda mendapatkan kompleksitas sebanyak ini. Karena arsitekturnya berbeda.

Anda mungkin bertanya kepada saya bagaimana dengan klasifikasi jenis gradien. Ini adalah jenis regresi logistik yang memiliki gradien. Mari kita bayangkan awalnya garis-garis ini sangat acak jika kita melihat bagaimana tampilannya sebelum pelatihan, ini sangat kacau. Kita bahkan tidak akan tahu garis mana yang menjadi garis spesifik ini. Kita memerlukan mekanisme untuk melakukan transisi garis yang halus. Sekarang Bayangkan sebuah tim yang baru direkrut, kita berikan mereka 5 masalah, dan berkata saya butuh seseorang untuk mendeteksi garis vertikal, seseorang untuk mendeteksi garis horizontal, dan mereka semua berjuang untuk melakukan apa. Jadi, akhirnya, seseorang berkata biarkan saya mencari tahu garis ini dan Anda pergi dan mencari tahu garis lainnya, tanpa kita semua melakukan hal yang benar, kita tidak dapat melakukan klasifikasi. Perilaku itu muncul dari anak-anak yang awalnya acak menjadi seorang jenius.

16.3 MENENTUKAN JUMLAH LAPISAN DAN UNIT TERSEMBUNYI?

Ini adalah masalah penting, bagaimana Anda memutuskan jumlah klaster dalam pengelompokan K-means, Bagaimana Anda memutuskan kedalaman pohon keputusan, Bagaimana Anda memutuskan K dalam K-Nearest Neighbors, Bagaimana Anda memutuskan lebar jendela parzan, ini semua adalah hiperparameter. Ini mengendalikan kompleksitas model. Itu adalah sesuatu yang harus Anda berikan.

Kita tidak memutuskan jumlah Input dan kita tidak memutuskan jumlah Output, dan kita bahkan tidak memutuskan bobot. Kita mempelajari bobotnya. Semakin kompleks modelnya, semakin tinggi akurasinya, tetapi Anda perlu memberikan kompleksitas yang optimal.

Orang bertanya kepada saya seperti apa model yang tepat untuk digunakan? Itu bukan pertanyaan yang tepat, pertanyaan yang tepat adalah apakah Anda peduli dengan interpretasi output atau tidak? Mari saya ambil dua situasi. Pertama, mereka ingin mendeteksi penipuan, dan mereka tidak peduli bahwa hal itu harus dijelaskan kepada konsumen, mengapa kita menyebutnya penipuan. Di sini, interpretasi model tidaklah penting, yang penting adalah keakuratan model. Dalam kasus ini, kita menggunakan jaringan saraf karena sangat sulit untuk menafsirkannya.

Jika Anda peduli dengan Interpretasi, kita perlu menggunakan pohon keputusan. Mari saya ambil contoh lain. Dalam kasus lain, jika Anda menolak pinjaman seseorang, Anda perlu memberikan alasannya. Anda tidak bisa begitu saja menolak pinjaman, Anda harus memiliki alasan yang sah. Oleh karena itu, di sini interpretasi model sangat penting. Anda tidak dapat membangun jaringan saraf. Meskipun dengan biaya keakuratan yang lebih rendah, kita menggunakan pohon keputusan karena interpretabilitasnya tinggi. Jadi, saat memutuskan antara pohon keputusan dan jaringan saraf, Anda perlu bertanya mana yang lebih penting bagi Anda, keakuratannya atau interpretabilitasnya.

Ada kriteria lain seperti itu. Kriterianya adalah seberapa cepat model Anda berubah? Jika Anda berhadapan dengan lingkungan yang sangat dinamis, satu model yang Anda buat untuk musim panas tidak akan berfungsi untuk musim dingin, maka Anda perlu membangun ulang model Anda, jadi kriterianya adalah dapatkah saya membangun ulang model tersebut dengan cepat? Berapa lama waktu yang dibutuhkan untuk membangun model tersebut?

Kriteria lainnya adalah, apakah keputusan Anda bersifat real time atau batch? Jika keputusan tersebut bersifat real time, Anda harus membangun model yang dapat memproses data dengan cepat dan mengambil keputusan. Pengambilan keputusan tentang penipuan kartu kredit harus berupa keputusan real time. Namun, jika Anda membangun model pemeringkatan kredit, Anda dapat menggunakan pohon keputusan. Lalu, Anda mungkin bertanya satu pertanyaan lagi, dapatkah saya menggunakan KNN? Mungkin sangat akurat, tetapi akan memakan banyak waktu, karena dalam KNN, ia menghitung jarak dari semua titik, sehingga tidak dapat digunakan secara real time. Jadi, pertanyaan yang tepat bukanlah model mana yang bagus? Pertanyaan yang tepat adalah kriteria mana yang Anda pedulikan? Izinkan saya memberi tahu Anda empat kriteria apa yang perlu Anda perhatikan.

1. Pengambilan keputusan secara real-time atau tidak?
2. Akurasi yang Anda butuhkan tinggi atau tidak?
3. Seberapa cepat model Anda harus dibangun kembali?
4. Apakah interpretabilitas penting atau tidak?

Pikirkan empat pertanyaan ini dan kemudian Anda putuskan teknik pemodelan mana yang perlu Anda gunakan.

Ide kompleksitas seperti ini, Bayangkan apa yang terjadi jika mereka menambahkan satu lapisan tersembunyi lagi di atas lapisan yang sudah ada ini. Jika Anda menginginkan lebih banyak kompleksitas, arah lain yang harus ditempuh adalah Menambah jumlah lapisan.

Mari kita pahami persamaannya, Output dari lapisan sebelumnya unit ke-j dari lapisan sebelumnya, memberi Anda jumlah tertimbang, melakukan logistik di atasnya, memberi Anda

output dari lapisan berikutnya. Ini adalah persamaan rekursif, karena g di lapisan n, mengarah ke g di n+1. Ini terus berlanjut ke arah depan, inilah yang dilakukan neuron tetapi tempatnya berada membuat perbedaan.

Kita telah membahas model linear umum. Kita telah mengatakan bahwa jika kita ingin melakukan regresi linear, kita tidak ingin melakukannya hanya pada X. Kita dapat melakukannya pada fungsi X mana pun. Itulah yang dilakukan unit tersembunyi, bukan unit input. Jadi, ini masih model linear. Saya memulai J dari 0 karena W0 juga merupakan bagian dari persamaan linear ini. Saya tidak ingin mengatakan W0 ditambah sesuatu. Jadi, kita meletakkan angka 1 di sini dan langsung memulai dengan W0. Jadi, kita dapat mengatakan bahwa ini adalah model linear. Kita tidak dapat berurusan dengan linear. Kita perlu memastikan neuron kita terbatas. Bayangkan, jika neuron Anda benar-benar dapat bekerja sangat tinggi, otak Anda akan meledak. Jadi, kita mempertahankan fungsi logistik. Ujung yang lebih tinggi menjadi datar. Jadi, fungsi logistik pada kombinasi linear mengarah ke output neuron. Kita dapat menunjukkan algoritma jaringan Neural seperti berikut:

$$Z_k^{(\ell+1)} = f\left(\sum_{j=0}^{S_\ell} Z_j^{(\ell)} W_{jk}^{(\ell)}\right)$$

di mana:

$\boldsymbol{Z_k^{(\ell+1)}}$ → Aktivasi Neuron ke-k pada Lapisan Berikutnya

$\boldsymbol{f}$ → Fungsi Aktivasi

$\boldsymbol{Z_j^{(\ell)}}$ → Aktivasi Neuron ke-j pada Lapisan Saat Ini

$\boldsymbol{W_{jk}^{(\ell)}}$ → Bobot(Parameter)

Mari saya ulangi hal yang sama, ada beberapa komponen yang berbeda, Anda dapat memetakannya ke struktur neuron, jadi inputnya adalah komponen-komponen ini, dendrit yang berasal dari lapisan sebelumnya, semuanya dikumpulkan, dikirim melalui Axon, ke sisi lain, setelah fungsi aktivasi, ini adalah output neuron yang menuju neuron masa depan. Neuron tidak tahu apa yang harus dilakukan dengan outputnya, ia hanya berkata jika Anda terhubung dengan saya, inilah yang akan Anda dapatkan. Tugas fungsi aktivasi adalah memastikan bahwa hal yang linier tidak berjalan tanpa batas, dalam arah positif atau negatif, jadi ia hanya membatasi hal itu secara sistematis, ada berbagai jenis fungsi aktivasi.

Lihat saja bentuknya, semuanya melakukan hal yang sama, apa pun input yang datang, inputnya bisa sangat besar, saya ingin membatasi rentang output itu menjadi 0 hingga 1. Oleh karena itu, kita memerlukan sesuatu untuk mengikat output. Itulah sebabnya kita memasukkan regresi logistik sejak awal. Jadi bentuknya berubah. Semua fungsi ini melakukan hal yang sama dengan sedikit variasi, konsepnya hampir mendekati batas, Anda harus mampu melakukan sesuatu yang halus, sehingga tidak terlihat seperti Perceptron yang sulit, dan jauh dari batas Anda harus mampu memuat nilai antara 0 hingga 1. Jika kedua properti itu ada, maka kita dapat mendefinisikan semua jenis fungsi aktivasi tersebut, ini semua berasal dari

ilmu saraf, mereka melakukan banyak eksperimen untuk mencari tahu, apa fungsi yang tepat. Satu masalah dengan jaringan saraf adalah, saya tahu cara melatih bobot lapisan ini, semakin internal lapisan tersebut, semakin sulit untuk memperbaikinya karena bergantung pada begitu banyak hal yang akhirnya terjadi, yang mengarah ke benar atau salah.

Pikirkan ini dengan cara ini, jika saya meminta anak itu untuk mengenali A versus 4, neuron yang mendeteksi garis berperan, neuron yang mendeteksi karakter A berperan, neuron itu akan mendapatkan masukan langsung, anak itu mengatakan A dan itu bukan A. Jadi, dia harus memperbaiki dirinya sendiri, tetapi fiksasi itu akan lebih jauh ke belakang, dan berkata apakah Anda mendeteksi garis dengan benar. mungkinkah itu alasan Anda tidak dapat mendeteksi A dengan benar? Beginilah cara kerja backpropagation. Ini adalah penemuan besar dalam jaringan saraf, yang membuat jaringan saraf bahkan dapat melatih simpul internal. jika itu satu lapisan, itu mudah. Apa yang kita lakukan, ketika kita bangun di siang hari, kita mengambil data yang kita ajukan, dan kita mendapatkan umpan balik, kita mengambil umpan balik, dan melakukan backpropagation dan belajar, dan seterusnya.

Mari kita lihat hal yang sama pada data Housing_loan, dalam data Housing_loan berapa banyak input yang kita miliki, yaitu 4, dan kita memiliki sebuah konstanta, kemudian saya membangun sebuah model, dan saya memiliki 3 output, saya tidak tahu berapa banyak unit tersembunyi yang benar-benar saya butuhkan, bisa jadi kurang atau lebih dari kelas, kita bereksperimen dan melatih jaringan saraf untuk dataset Housing_loan. Kemudian saya mengambil satu contoh di baris ini, ia memiliki beberapa fitur, neuron-neuron sudah memiliki beberapa bobot, berdasarkan bobot-bobot tersebut ia akan memberi saya beberapa aktivasi di sini, yang akan memberi tahu saya apakah saya berada di sisi ini atau sisi itu dari garis, dan kita memiliki tiga kelas, Bayangkan bahwa saya memutuskan bahwa titik baru adalah kelas 3. Seperti apa outputnya, saya mengharapkan 1 di tempat kelas itu dan 0 di dua kelas lainnya. beginilah cara Anda mengubah masalah klasifikasi menjadi masalah jaringan saraf. Karena neuron belum sempurna, neuron tersebut mungkin memberi Anda 0,9, 0,4, 0,6 untuk tiga kelas, karena neuron tersebut belum terlatih sepenuhnya, maka kita akan menemukan kesalahan dan kita perlu melakukan backpropagation kesalahan tersebut, oleh karena itu hal tersebut disebut backpropagation kesalahan. Di sini, dalam kasus ini, saya perlu melakukan backpropagation kesalahan sebesar -0,9 karena kita selalu mengatakan target dikurangi aktual (0–0,9) adalah kesalahan. Jadi bobot yang menyebabkan kesalahan begitu tinggi sekarang akan turun. Dalam kasus kedua, bobot pun akan turun tetapi tidak sebanyak situasi pertama. Dan dalam kasus ketiga, kita mendapatkan 0,4 jadi kita meningkatkan bobot sehingga aktivasi menjadi lebih tinggi sekarang. Jadi kesalahan harus disebarkan dari atas ke bawah dan output harus dari bawah ke atas. Mari kita ambil contoh Google yang digunakan untuk mendeteksi gambar, mereka mungkin memiliki 100-an lapisan. Hal itu menjadi jauh lebih rumit untuk ditafsirkan, tetapi ia melakukan sesuatu yang sangat hebat. Setiap kali Anda menggunakan jaringan saraf, pertama-tama pelajari data secukupnya, putuskan seberapa rumitnya, dengan cara itu kita mendapatkan titik awal yang baik, dan asumsikan bahwa tiga tampak cukup baik, lalu Anda coba dengan 2 dan 4 genap, dan lihat apakah hasilnya membaik. Jika membaik pada 4, saya coba dengan 5 genap dan seterusnya. Ini masalah coba-coba.

Itulah alasan pekerjaan ilmuwan data sedikit menantang, ia dapat kembali ke analisis mentah, membangun model yang sangat mentah, lalu melihat kembali dan belajar serta membuat model lagi dan seterusnya. Terobosan besar yang sebenarnya adalah bagaimana Anda mempelajari bobot ini? Jumlah beban yang masuk ke cara ini proporsional dengan keterlibatan unit tersebut dalam keputusan. Idenya adalah semakin tinggi bobot Anda, semakin banyak kesalahan yang harus Anda tanggung.

Mari saya ambil skenario bisnis, seberapa besar kesalahan yang dibuat direktur ini sebanding dengan beberapa hal seperti berapa total kesalahan dan apa yang telah dilakukannya dengan unit ini dan berapa besar kesalahan yang dilakukan orang lain dan apa keterlibatannya dalam keputusan itu, bersama-sama berapa jumlah koreksi yang perlu dilakukan unit ini? Propagasi balik pertama-tama mengakumulasi kesalahan dan mengirimkannya ke pendahulu saya. Saya melakukan hal yang sama kepada junior saya karena berdasarkan komitmen mereka, saya berkomitmen kepada V.P. Seperti ini, forward pass adalah mengakumulasi info dan mendistribusikan dan backward propagasi dalam mengakumulasi kesalahan dan mendistribusikan. Lihatlah keindahan jaringan saraf, neuron yang sama melakukan hal yang persis sama, tetapi cara mereka terhubung, mereka terus meneruskan info dan kesalahan dan terus belajar. Jadi bobotnya terus belajar.

Ini adalah terobosan dan karena ini kita semua memiliki kartu kredit saat ini. Bayangkan jika kita tidak menerapkan jaringan saraf untuk masalah deteksi penipuan, bank akan menutup sistem kartu kredit. Jaringan saraf tidak memiliki status, pada dasarnya mengambil satu masukan dan memberi Anda satu keluaran, dan mengambil kesalahan itu dan melakukan propagasi balik. Ia tidak tahu apa pun tentang contoh berikutnya. Ingat IID (Independent and Identically Distributed), neuron tidak memiliki status, ia melakukan gerakan mundur dan maju.

Dalam banyak situasi lain, Anda memerlukan status, misalnya, Anda ingin memutuskan seberapa keras Anda ingin berbicara dengan teman Anda, itu tergantung di mana Anda berada. Di sini, status adalah di ruangan mana Anda berada dan seberapa jauh atau dekat dia dengan Anda. Pada dasarnya, pikirkan status dalam lingkungan di mana status sebelumnya juga berkontribusi pada keluaran berikutnya. Dalam situasi di mana jaringan saraf biasa tidak berfungsi, mereka menciptakan sesuatu yang baru yang disebut jaringan saraf berulang. Dalam hal ini, apa yang dilakukannya? Ia mengambil masukan saat ini, ia menghasilkan status, tetapi kemudian ia mengingat status yang sekali lagi menjadi masukan, sekarang kita katakan ini masukan Anda saat ini, ini status Anda, bersama-sama kita akan memutuskan apa yang seharusnya menjadi keluaran berikutnya.

Setiap kali Anda memiliki masalah pembelajaran berurutan seperti Anda belajar memprediksi pasar saham, Anda perlu mengingat status sebelumnya, dan jika Anda ingin memprediksi kata berikutnya dalam urutan kata, Anda perlu mengingat status tersebut. setiap kali Anda memiliki suatu keadaan yang terjadi, ada suatu lingkaran umpan balik internal yang kembali. karena ini adalah bagian memori dalam pembelajaran. seperti otak kita memiliki gagasan tentang memori. akan ada hal yang membusuk yang terlibat di sini karena saya tidak dapat memberikan bobot yang sama pada pembelajaran kemarin dan pembelajaran satu

bulan yang lalu. itu seperti pembusukan eksponensial. pikirkan tentang ini, Anda tahu apa yang Anda lakukan hari ini pagi tetapi Anda tidak tahu apa yang Anda lakukan 5 hari yang lalu. jadi keadaan Anda selalu terkini.

Jenis lain dari jaringan saraf adalah jaringan kompresi, Jika kita mengambil gambar dengan kamera, saya akan mendapatkan gambar dalam format png yang sangat besar, lalu apa yang harus saya lakukan, saya mengompresnya. ketika Anda mengompresnya tidak cukup baik, Anda seharusnya dapat mendekompresinya. hanya dengan begitu Anda mendapatkan gambar jpeg. Di sini dalam situasi ini, ia tidak mempelajari apa pun yang disebut diawasi, ia tidak mengatakan X ke Y apa yang dikatakannya adalah, X ke versi terkompresi dari X, dan kemudian tidak terkompresi, sekarang input dan output seharusnya sama. jika kompresinya bagus, yang mereka lakukan adalah mengambil angka yang sama, memasukkan angka yang sama ke dalam output, lalu mempelajari jaringan, ini adalah teknik pembelajaran tanpa pengawasan karena tidak ada variabel Y. Anda dapat menggunakan ini di mana pun Anda menggunakan PCA, PCA adalah proyeksi linier Jaringan saraf terkompresi adalah proyeksi non-linier. Ini seperti penyandi ucapan, telepon kita menggunakan ini, suara Anda pertama-tama dikompresi, lalu ditransmisikan di saluran, lalu orang lain menerimanya, ia memiliki penyandi-dekoder yang sama, ia menggunakan dekodernya untuk mendekompresinya, dan apa yang Anda dengar adalah sedikit variasi dari yang asli.

Di sini kita tidak mempelajari pemetaan antara sesuatu dengan sesuatu yang lain. seperti kita memetakan fitur ke label kelas. tetapi yang ini adalah masalah kompresi yang berarti, Anda harus mempelajari kompresi dan dekompresi sehingga efek keseluruhan kesalahan rekonstruksi keseluruhan harus diminimalkan.

Tingkat pembelajaran adalah parameter lain yang harus kita tentukan seperti kita menentukan unit Tersembunyi. Umumnya, kita mulai dengan laju pembelajaran yang kecil dan jika kita merasa sangat lambat maka kita tingkatkan laju pembelajarannya. Namun pada titik tertentu, jika kita meningkatkan laju pembelajaran terlalu banyak, ia akan menjadi overtrained, jadi kita perlu menemukan laju pembelajaran yang optimal. Pertimbangan penting adalah laju pembelajaran μ, yang menentukan seberapa banyak kita mengubah bobot w pada setiap langkah. Jika μ terlalu kecil, algoritme akan membutuhkan waktu lama untuk konvergen. Sebaliknya, jika μ terlalu besar, kita mungkin akan terpental di sekitar permukaan kesalahan di luar kendali, algoritme akan menyimpang.

16.4 MEMBANGUN JARINGAN SYARAF PADA DATASET HOUSING_LOAN

Langkah 1: Instal dan Muat paket yang diperlukan

```
install.packages('neuralnet') library("neuralnet")
library(dummies)
library(vegan)
```

Langkah 2: Memuat Data ke R:

```
loandata=read.csv(file="D:\\R
data\\Housing_loan.csv",header=TRUE, sep=",")
fix(loandata)
```

Langkah 3: Hapus kolom ID dari data

```
loandata2=subset(loandata, select=-c(ID))
fix(loandata2)
Edu_dum =dummy(loandata2$Education)
loandata3=subset(loandata2,select=-c(Education))
fix(loandata3) loandata4=cbind(loandata3,Edu_dum)
fix(loandata4)
```

Langkah 4: Standarkan data menggunakan metode

```
'Range' loandata_stan=decostand(loandata4,"range")
fix(loandata_stan)
```

Tetapkan seed untuk mendapatkan data yang sama di setiap waktu

```
set.seed(123)
```

Langkah 5: Ambil sampel acak 60% dari rekaman untuk data train

```
train = sample(1:1000,600)
loan_train = loandata_stan[train,]
```

Ambil sampel acak 40% dari rekaman untuk data uji

```
test = (1:1000) [-train]
loan_test = loandata_stan[test,]
table(loandata_stan$Loan_sanctioned)
table(loan_train$Loan_sanctioned)
table(loan_test$Loan_sanctioned)
rm(loandata2,loandata3,loandata4,loandata_stan,Edu_dum,test,train)
```

Langkah 6: Bangun Jaringan Syaraf

```
nn <- neuralnet(Loan_sanctioned~ Usia+Pengalaman+
  Pendapatan+Keluarga+Pendidikan1+Pendidikan2+ Pendidikan3,
  data=loan_train, hidden=c(2,3))
out <- cbind(nn$covariate, nn$net.result[[1]]) fix(out)
dimnames(out)=list(NULL,c
  ("Usia","Pengalaman","Pendapatan","Keluarga","Pendidikan1","
  Pendidikan2", "Pendidikan3","nn-output"))
```

Langkah 7: Lihat rekaman teratas dalam set data

```
head(out) plot(nn)
```

Langkah 8: Persiapan data untuk matriks klasifikasi

```
p=as.data.frame(nn$net.result)
colnames(p)="pred"
pred_class <- factor(ifelse(p$pred > 0.5, 1, 0))
a <- table(pred_class, loan_train$Loan_sanctioned)
recall <- a[2,2]/(a[2,1]+a[2,2])*100
```

Langkah 9: Mengambil kolom yang dibutuhkan dari data

```
test_data2=subset(loan_test, select=-c(Loan_sanctioned))
```

```
new.output <- compute(nn,covariate=test_data2)
p=as.data.frame(new.output$net.result)
colnames(p)=”pred”
pred_class <- factor(ifelse(p$pred > 0.5, 1, 0))
a <- table(pred_class,loan_test$Loan_sanctioned)
recall <- a[2,2]/(a[2,1]+a[2,2])*100
recall
```

Langkah 10: Bermain dengan struktur node yang berbeda (3), (2,2), (4,3)

Kita dapat mengambil lapisan tersembunyi yang berbeda dan masukan tersembunyi dan buat model, periksa persentase akurasi dan recall-nya. Kemudian selesaikan model berdasarkan akurasi dan recall pada dataset Validasi (Pengujian).

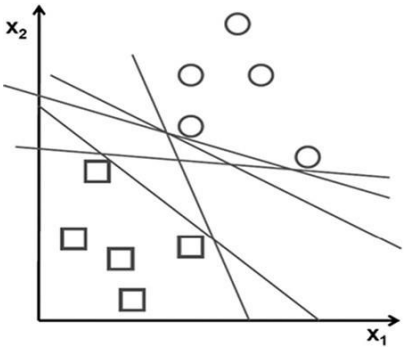
BAB 17
MESIN VEKTOR PENDUKUNG

17.1 PENDAHULUAN

Mesin Vektor Pendukung (SVM) adalah alat klasifikasi dan prediksi regresi yang menggunakan teori pembelajaran mesin untuk memaksimalkan akurasi prediktif sekaligus secara otomatis menghindari over-fit terhadap data. Setiap kali kita berpikir untuk membuat model, kita memikirkan beberapa hal seperti kompleksitas, determinisme, pengambilan sampel, fitur, dll.

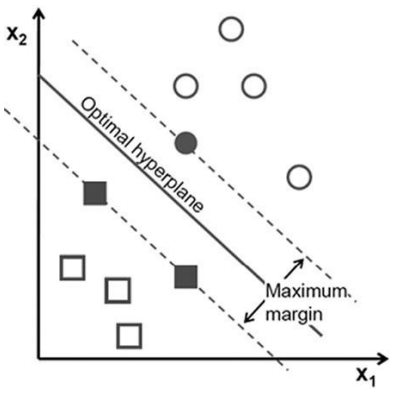
Kompleksitas: Dalam teknik pemodelan yang baik, kita harus dapat mengendalikan kompleksitasnya, dalam kasus campuran Gaussian, kompleksitas adalah jumlah komponen, sedangkan kompleksitas jaringan Neural adalah jumlah unit tersembunyi dan lapisan tersembunyi.

Determinisme: Saya ingin algoritme menghasilkan hal yang sama berulang-ulang dengan data pelatihan yang sama. Bayangkan algoritme lain bergantung pada titik awal, ia menuju ke tempat lain. Itu bukan properti yang diinginkan, tetapi itulah yang terjadi ketika Anda memiliki model yang kompleks. Perseptron, jaringan neural, K-means Clustering memiliki masalah yang sama, Ia tidak menghasilkan hasil yang sama setiap saat.



Bidang Hiper Optimal: Masalah klasifikasi dapat dibatasi pada perenungan masalah dua kelas tanpa kehilangan keumuman. Dalam masalah ini, tujuannya adalah untuk membedakan dua kelas dengan fungsi yang diinduksi dari contoh yang disajikan. Tujuannya adalah untuk menghasilkan pengklasifikasi yang akan bekerja dengan baik pada contoh yang tidak terlihat, yaitu mengumumkan dengan baik. Di sini ada banyak pengklasifikasi linier potensial yang dapat memisahkan data, tetapi hanya ada satu yang memaksimalkan margin (memaksimalkan jarak antara margin dan titik data terdekat dari setiap kelas). Pengklasifikasi linier ini disebut bidang hiper pemisah yang optimal.

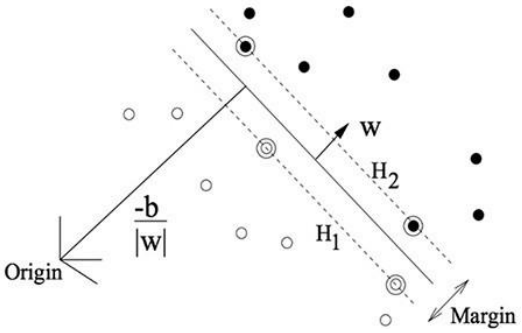
Misalnya saya memiliki kumpulan data dengan masalah dua kelas, saya ingin membuat pengklasifikasi, dan saya menggunakan Perceptron sebagai pengklasifikasi. Perceptron di mana pun ia memulai, ia terus belajar dan segera setelah ia berhenti membuat kesalahan, ia berhenti belajar. Kita dapat memiliki banyak perceptron (Tak Terbatas) untuk kumpulan data yang memiliki masalah dua kelas. Semua perceptron ini memiliki biaya kesalahan klasifikasi yang sama yaitu Nol. Jadi semua ini bisa menjadi model yang valid. Tapi apa yang salah dengan mereka? Hal pertama adalah Mereka tidak deterministik, itu berarti setiap kali ketika saya mulai di tempat lain, itu memberi saya hasil yang berbeda. Ada satu masalah



lagi, ada beberapa titik yang benar-benar dekat dengan batas. Jadi itu adalah model yang rapuh. Jika saya mendapatkan titik data yang sedikit berbeda, Itu mungkin membuat kesalahan besar. Itu bukan yang kita inginkan. Kami ingin model menjadi lebih kuat. Sekarang karena kita memiliki masalah ini, kita harus mencoba sesuatu yang lebih baik daripada model yang ada Itulah Bagaimana SVM muncul. Sekarang pikirkan secara intuitif, di mana batas keputusan seharusnya berada. Itu harus jauh dari kedua titik data, itu berarti di tengah titik data terdekat. Batas keputusan harus kuat. Perubahan kecil dan noise seharusnya tidak memengaruhi batas keputusan.

Bayangkan ini adalah dua desa, dan Anda ingin membuat jalan di antara keduanya. Jalan harus selebar mungkin. Namun, ada kendala jika saya ingin membangun jalan yang lebar, tetapi saya tidak ingin merusak rumah-rumah. Yang kami coba lakukan di sini adalah membangun batas yang sekuat mungkin, tanpa merusak rumah-rumah di kedua sisinya. Apa pengklasifikasi yang baik? Saya ingin pengklasifikasi linier sehingga saya ingin menggambar garis di kedua sisi dan berhenti di tempat rumah pertama berada. Lalu, lihat lebar jalan ini. Ini juga disebut pengklasifikasi margin maksimum.

Di sini, tidak semua titik data penting. Titik data yang dekat dengan bataslah yang penting. Pertanyaannya, dapatkah Anda menemukan rumah-rumah yang dapat saya buat jalannya? Mari kita lihat bagaimana seorang matematikawan memecahkan masalah pembelajaran mesin. Jika saya memiliki hiperbidang dan ingin mengukur jarak tegak lurus antara titik asal dan garis, maka jaraknya akan menjadi sebesar ini (- B/|w|).



$$\mathbf{x}_i \cdot \mathbf{w} + b \geq +1 \quad \text{for } y_i = +1$$
$$\mathbf{x}_i \cdot \mathbf{w} + b \leq -1 \quad \text{for } y_i = -1$$

Jadi saya ingin menemukan W dan B, yang kita coba di sini adalah jika kita dapat menemukan jalan tengah, jika Anda memilih +1 di sini, atau -1 di sini, seharusnya tidak ada apa pun (Tidak Ada Rumah) di antaranya. Jadi persamaannya terlihat seperti ini, Xi (Titik data terdekat), untuk semua rumah yang berada di satu sisi jalan, kondisi ini harus berlaku (W+B >= +1) untuk semua rumah yang berada di sisi lain jalan, kondisi lainnya harus berlaku. (W+B <= -1). Sekarang ingat apa yang dilakukan Perceptron, Perceptron mengatakan bahwa Xi W + W0 adalah >0 atau <0. Di Perceptron, segera setelah Anda melewati batas, Anda berada di sisi lain. Tidak ada gagasan tentang margin. Itu hanya garis. Di SVM, yang kita lakukan adalah kita memerlukan margin kesalahan, jadi Alih-alih mengatakan 0, kita akan meletakkan 1 dan -1 di sini. Itulah satu-satunya perbedaan antara Perceptron yang memiliki Margin Nol Versus SVM yang memiliki margin Maksimum, yang berarti kita meningkatkan ketebalan Perceptron. Jika kita memasukkan kedua persamaan di atas dalam satu persamaan, kita memperoleh Yi(Xi.W+B)-1 >=0. Dalam pembelajaran terbimbing, kita selalu mendefinisikan fungsi objektif dan menyelesaikannya. Fungsi objektif memiliki dua bagian. (Maksimalkan, Kendala). Di sini kita mencoba memaksimalkan margin di bawah kendala bahwa Tidak ada kerusakan pada Rumah.

Sekarang beri tahu saya berapa banyak kendala di sini? Mari kita kembali ke situasi pemrograman linier. Dalam pemrograman linier, kita menggambar sekumpulan garis, yang dianggap sebagai kendala linier. Oleh karena itu, kita pikir itu harus berada di satu sisi garis. Saya tidak dapat menggambar garis selebar mungkin karena saya memiliki kendala bahwa saya tidak dapat memecahkan rumah. Saya memiliki Kendala N (Jumlah Total Titik Data) sekarang karena saya tidak dapat memecahkan rumah mana pun.

Dalam diagram awal, Semua garis mungkin tetapi kita perlu menyempurnakan fungsi objektif saya lebih lanjut untuk mendapatkan garis unik atau solusi optimal. Garis yang terjauh dari kedua sisi adalah yang terbaik karena memaksimalkan margin. Ketika kita merumuskan masalah, itu hanya berarti bahwa jika ini adalah garisnya, apa marginnya, apa kendalanya, dan kemudian kita selesaikan untuk apa garisnya. Ini seperti Biarkan X menjadi solusinya, dan selesaikan untuk X.

Jika kita melihat matematika, Geometri mengatakan Jika saya menggambar garis tegak lurus dari titik asal ke garis mana pun, maka panjangnya adalah (-b/w). kemudian ketika kita melihat dua garis lainnya yang satu di atas dan yang lainnya di bawah, panjangnya akan menjadi (1-b/w) dan (-1-b/w), Margin akan menjadi satu persamaan dikurangi persamaan lainnya. B dibatalkan dan yang kita dapatkan adalah 2/W.

$$Objective: \frac{1}{2}\|w\|^2$$
$$Constraint: y_i(x_i \cdot w + b) - 1 \geq 0 \; \forall_i$$

Di sini kita memiliki fungsi objektif dan serangkaian kendala. Kendala tersebut adalah jumlah titik data karena kita tidak dapat memecah rumah atau desa mana pun, setiap titik data (Rumah) adalah kendala saya.

Tujuan kita adalah mengoptimalkan fungsi objektif dengan kendala yang diberikan. Dalam situasi ini, Lagrange mengusulkan Pengganda yang menyatakan bahwa kita perlu membayar sejumlah penalti karena melanggar setiap kendala. Misalkan jika Anda melanggar kendala ke-i, Anda membayar penalti Alpha I Ci(x). Alfa tersebut disebut Pengganda Lagrange. Kemudian kita jumlahkan semua penalti.

Dalam algoritma Support Vector Machine, kita dapat menerapkan tiga trik:

1. Konversi Primal ke Dual
2. Variabel Slack
3. Memilih Kernel

Coba bayangkan fungsi objektif kita adalah untuk memaksimalkan margin sehingga melanggar kendala akan meminimalkan nilai. Jadi, kita perlu mengurangi penjumlahan keseluruhan penalti dari fungsi Objektif yang ingin kita maksimalkan. Jadi akhirnya persamaan tersebut akan terlihat seperti ini:

17.2 KONVERSI PRIMAL KE DUAL

Pengali Lagrange: Mari saya ambil contoh sederhana agar Anda dapat memahami seluruh teori ini. Bayangkan istri Anda menelepon Anda dan mengatakan bahwa dia

berencana untuk pergi berbelanja pada pukul 19.30. Sekarang tujuan Anda adalah tiba di rumah lebih awal. Ini adalah fungsi tujuan Anda. Anda memiliki serangkaian kendala seperti menyelesaikan pekerjaan lebih awal, jangan mengemudi terlalu cepat, jangan menerobos lampu lalu lintas, jangan menabrak kendaraan apa pun, dll., jika Anda melanggar salah satu kendala ini, Anda akan dikenai penalti. Penalti tersebut berbeda untuk setiap kendala. Jadi, akhirnya, kita jumlahkan semua penalti dan kurangi itu dari fungsi tujuan yang ingin Anda maksimalkan. Yang perlu kita pikirkan adalah apakah kita melanggar kendala atau tidak saat mencapai tujuan kita. Jika kita tidak melanggar kendala apa pun, penalti kita akan menjadi nol.

$$L_p \equiv \frac{1}{2}\|w\|^2 - \sum_{i=1}^{l} \alpha_i y_i (x_i \cdot w + b) + \sum_{i=0}^{l} \alpha_i$$

Masalah primal Lagrange terlihat seperti di atas dan yang perlu kita pahami adalah kita ingin mengubah masalah primal ini menjadi masalah dual sehingga saya dapat menyingkirkan w dan b.

Karena jika saya mengetahui alpha, saya dapat menghitung w dan b. Seperti ini, Jika saya mengetahui rumah-rumah mana yang tidak boleh saya rusak saat membangun jalan, saya dapat merencanakan jalan saya sesuai dengan itu. Bentuk dual persamaan terlihat seperti ini di mana kita tidak memiliki w dan b.

$$L_D = \sum_i \alpha_i - \frac{1}{2}\sum_{i,j} \alpha_i \alpha_j y_i y_j x_i \cdot x_j$$

Hal penting yang perlu diingat adalah Kita dapat memaksimalkan margin dengan mengidentifikasi titik-titik data (rumah) yang akan membentuk batas, pada dasarnya ini disebut vektor pendukung. Ini adalah vektor karena ini adalah titik-titik dalam ruang berdimensi tinggi dan ini adalah titik-titik (vektor) yang mendukung bidang (Batas Keputusan).

Kita memiliki satu kendala untuk satu titik data dan setelah kita menyelesaikannya, ini menghasilkan sekumpulan nilai alfa untuk setiap titik data. Nilai-nilai alfa ini mengatakan bahwa titik-titik ini dekat dengan batas keputusan atau tidak. Hal penting yang perlu dipahami di sini adalah sebagian besar nilai alfa adalah nol karena sebagian besar titik data berada di dalam batas (sebagian besar rumah berada di dalam desa). Yang perlu kita pahami di sini adalah semakin tinggi nilai alfa, semakin tinggi titik itu berada di jalan.

17.3 VARIABEL SLACK

Sampai sekarang kita membahas data yang dapat dipisahkan secara linier. Sekarang pikirkan tentang apa yang terjadi jika titik-titik data tidak dapat dipisahkan secara linier. Bayangkan saja data dunia nyata yang biasanya tidak dapat dipisahkan secara linier. Di sini kita perlu menerapkan satu trik lagi yang disebut Variabel Slack. Jika set pelatihan tidak dapat

dipisahkan secara linear, kita menggunakan pendekatan standar untuk memungkinkan margin keputusan yang besar untuk membuat beberapa kesalahan untuk beberapa titik data seperti outlier atau contoh yang tidak jelas yang berada di dalam atau di sisi margin yang salah. Kita kemudian membayar biaya untuk setiap contoh yang salah diklasifikasikan, yang bergantung pada seberapa jauh contoh tersebut dari memenuhi persyaratan margin. Untuk menerapkan ini, kita memperkenalkan variabel slack. Nilai bukan nol untuk ξi memungkinkan xi untuk tidak memenuhi persyaratan margin dengan biaya yang proporsional dengan nilai ξi.

Masalah pengoptimalan kemudian memperdagangkan seberapa besar margin yang dapat dibuat versus berapa banyak titik yang harus dipindahkan untuk memungkinkan margin ini. Margin dapat kurang dari 1 untuk titik xi dengan menetapkan ξi > 0 tetapi kemudian seseorang membayar penalti Cξi dalam minimisasi karena telah melakukan itu. Jumlah ξi memberikan batas atas pada jumlah kesalahan pelatihan. SVM margin lunak meminimalkan kesalahan pelatihan yang dipertukarkan dengan margin. Parameter C adalah istilah regularisasi, yang menyediakan cara untuk mengendalikan overfitting: saat C menjadi besar, tidak menarik untuk tidak menghormati data dengan mengorbankan pengurangan margin geometris; saat C kecil, mudah untuk memperhitungkan beberapa titik data dengan penggunaan variabel longgar dan menempatkan margin tebal sehingga memodelkan sebagian besar data.

Amati titik data di atas dan batasan yang diberikan. Ada kemungkinan jalan dengan titik data ini, tetapi

$$X_i \cdot w + b \geq +1 - \mathcal{E}_i \; untuk \; y_i = +1$$
$$X_i \cdot w + b \leq -1 + \mathcal{E}_i \; untuk \; y_i = -1$$
$$\mathcal{E}_i \geq 0 \forall_i$$

Model macam apa ini?

- Apa modelnya?
- Apa saja parameternya?
- Apa itu kompleksitas?

17.4 TRIK KERNEL

Jika data bersifat nonlinier, artinya jika data tidak dapat dipisahkan dengan garis lurus, maka kita ingin SVM memproyeksikan data ke ruang dimensi yang lebih tinggi agar memungkinkan pemisahan secara linier atau melakukan pemisahan linier. Ini disebut trik kernel.

17.5 MESIN VEKTOR PENDUKUNG PADA DATASET PINJAMAN PERUMAHAN

```
setwd("D:/R data")
```

#Memuat Data ke R:

```
loandata=read.csv(file="Housing_loan.csv", header=TRUE)
```

#Persiapan Data: Hapus ID kolom dari data

```
loandata2=subset(loandata, select=-c(ID))
fix(loandata2)
```

#Variabel "Pendidikan" memiliki lebih dari dua kategori, (1: Sarjana, 2: Pascasarjana, 3: Lanjutan/Profesional), #jadi kita perlu membuat variabel dummy untuk setiap kategori untuk dimasukkan ke dalam analisis.buat variabel dummy untuk variabel kategori

#"Pendidikan" dan tambahkan variabel dummy tersebut ke data asli.

```
install.packages("dummies") library(dummies)
```

#Instal & Muat paket "dummies" untuk membuat variabel dummy

```
Edu_dum=dummy(loandata2$Education)
head(Edu_dum)
loandata3=subset(loandata2,select=-c(Education))
loandata4=cbind(loandata3,Edu_dum)
head(loandata4)
```

#Standardisasi Data: Standarisasi data menggunakan metode 'Range'

```
install.packages("vegan")
library(vegan)
loandata5=decostand(loandata4,"range")
```

#Siapkan set data train & test, Ambil sampel acak 60% dari record untuk data train

```
train = sample(1:1000,600)
train_data = loandata5[train,]
nrow(train_data)
```

#Ambil sampel acak 40% dari record untuk data test

```
test = (1:1000) [-train]
test_data = loandata5[test,]
nrow(test_data)
```

#Ringkasan Data untuk variabel respons "Loan_sanctioned":

```
table(loandata5$Loan_sanctioned)
```

#Data Latih

```
table(train_data $Loan_sanctioned)
```

#Data Uji

```
table(test_data$Loan_sanctioned)
```

#Klasifikasi menggunakan SVM:

```
install.packages("e1071") library(e1071)
```

#Instal & Muat paket e1071 untuk melakukan analisis SVM.

```
x = subset(train_data, select = -Loan_sanctioned)
y = as.factor(train_data$Loan_sanctioned)
?svm
model=svm(x,y,method="C-classification",kernel="linear",cost=10,
gamma=0.1
```

#***Kernel:*** Kernel yang digunakan dalam pelatihan dan prediksi. Anda dapat mempertimbangkan untuk mengubah beberapa parameter berikut, tergantung pada jenis kernel.

Cost: biaya pelanggaran kendala (default: 1)-ini adalah konstanta 'C' dari istilah regularisasi dalam formulasi Lagrange.

Gamma: parameter yang diperlukan untuk semua kernel kecuali linear. ringkasan(model)

 # Uji dengan data kereta

```
pred = prediksi(model, x)
tabel(pred, y)
```

 # Uji dengan data uji

```
a = subset(data_uji, pilih = -Loan_sanctioned)
b = as.faktor(data_uji$Loan_sanctioned)
pred= prediksi(model, a)
tabel(pred, b)
model2 = svm(x,y, metode = "Klasifikasi-C", kernel = "radial",
biaya = 10, gamma = 0,1) ringkasan(model2)
```

 # Uji dengan Data Train

```
pred = prediksi(model, x) tabel(pred, y)
```

 # Uji dengan data uji

```
pred = prediksi(model, a) tabel(pred, b)
```

BAB 18
PEMBELAJARAN ENSEMBLE

18.1 PENDAHULUAN

Mari kita beralih ke dimensi lain dari teknik pemodelan, yang disebut metode ensemble. Sampai sekarang kita membahas model individual, dan kita melihat cara menangani peningkatan kompleksitas, tetapi mari kita pikirkan apa yang harus dilakukan jika model ini tidak cukup baik? Jika SVM linear tidak cukup, kita beralih ke SVM polinomial derajat 2 dan kemudian derajat 3. Jika ini tidak cukup, kita beralih ke SVM nonlinier dengan kernel RBF. Ada cara untuk terus meningkatkan kompleksitas, tetapi seperti yang kita lihat, peningkatan kompleksitas akan meningkatkan akurasi hingga satu tingkat tertentu saja. Dalam metode ini, kita terus meningkatkan kompleksitas.

Area besar lainnya untuk meningkatkan kinerja model adalah merekayasa fitur yang lebih baik. Anda mungkin berkata bahwa saya mengekstrak sejumlah fitur, dan membangun sebuah model. Saya membangun model terbaik dengan serangkaian fitur ini. Saya tidak dapat melakukan yang lebih baik dari itu. Mari saya kembali ke data saya, tingkatkan fitur saya, tambahkan beberapa fitur lagi, sekali lagi bangun model yang kompleks, dan siklus ini terus berlanjut. dengan fitur mentah, kita mungkin memerlukan model yang kompleks, tetapi dengan rekayasa fitur yang lebih baik, kita mungkin memerlukan model yang sederhana. Mari kita ambil pendekatan lain untuk meningkatkan kinerja model kita, yang disebut pembelajaran ensemble. Dalam hal ini, alih-alih mempelajari model yang kompleks, kita mempelajari banyak model sederhana dan menggabungkannya. Itulah pendekatan untuk meningkatkan kompleksitas model secara keseluruhan.

Ensemble tidak lain hanyalah sekelompok hal sebagai satu koleksi. Sejauh ini yang kita lakukan adalah, kita mengambil beberapa masukan, kita mengekstrak beberapa fitur, kita melatih model, kita meningkatkan kompleksitas model, dan memperoleh keluaran. Ensemble adalah teknik untuk menggabungkan banyak pembelajar yang lemah dalam upaya menghasilkan pembelajar yang kuat. Dalam statistik dan pembelajaran mesin, metode ensemble menggunakan beberapa model untuk memperoleh kinerja prediktif yang lebih baik daripada yang dapat diperoleh dari salah satu model utama. Istilah ensemble biasanya diperuntukkan bagi metode yang menghasilkan beberapa hipotesis menggunakan pembelajar dasar yang sama. Mengevaluasi prediksi ensemble biasanya memerlukan lebih banyak komputasi daripada mengevaluasi prediksi satu model, sehingga ensemble dapat dianggap sebagai cara untuk mengimbangi algoritma pembelajaran yang buruk dengan melakukan banyak komputasi tambahan. Algoritma cepat seperti pohon keputusan umumnya digunakan dengan ensemble.

18.2 BAGGING

Bagging adalah teknik yang digunakan untuk mengurangi varians prediksi kita dengan menggabungkan hasil beberapa pengklasifikasi yang dimodelkan pada berbagai sub-sampel (Pengambilan Sampel Data) dari set data yang sama.

Membuat Beberapa Set Data: Pengambilan sampel dilakukan dengan penggantian pada data asli dan set data baru dibentuk. Set data baru dapat memiliki sebagian kolom maupun baris, yang umumnya merupakan hiperparameter dalam model bagging. Ini membantu dalam membuat model yang tangguh, yang tidak mudah mengalami overfitting. Kita Membangun Beberapa Pengklasifikasi pada setiap set data, dan prediksi dibuat.

Pengklasifikasi Gabungan: Prediksi semua pengklasifikasi digabungkan menggunakan nilai rata-rata atau modus tergantung pada masalah Bisnis. Nilai gabungan sebagian besar lebih tangguh daripada model tunggal. Semakin banyak model, semakin baik kinerjanya daripada semakin sedikit jumlahnya. Secara hipotetis dapat ditunjukkan bahwa varians prediksi gabungan dikurangi menjadi 1/n (n: jumlah pengklasifikasi) dari varians asli.

Langkah-langkah dalam agregasi Bootstrap (Bagging):

- Kita mulai dengan ukuran sampel N
- Kita membuat sejumlah besar sampel dengan ukuran yang sama. Sampel baru dihasilkan dari set data Pelatihan menggunakan metode pengambilan sampel dengan penggantian. Jadi, sampel tersebut tidak identik dengan sampel asli.
- Kita mengulanginya berkali-kali, mungkin 1000 kali, dan untuk setiap sampel bootstrap ini, kita menghitung rata-ratanya, yang disebut estimasi bootstrap.
- Buat Histogram dengan estimasi ini, jika memberikan estimasi bentuk distribusi rata-rata, dari situ kita dapat mengetahui seberapa besar fluktuasi rata-rata.

Prinsip utama bootstrap adalah menyediakan cara untuk mensimulasikan pengamatan berulang dari populasi yang tidak diketahui menggunakan sampel yang diperoleh sebagai dasar. Kita mengambil n*<n sampel dari D dengan penggantian (berarti sampel yang sama dapat diambil dalam beberapa pengambilan). Kita menerapkan pengklasifikasi C1 dan mengulanginya dengan sampel yang berbeda dan pengklasifikasi baru C2. Model 'm' dipasang menggunakan m sampel bootstrap di atas dan digabungkan dengan merata-ratakan output untuk regresi atau pemungutan suara untuk klasifikasi.

18.3 RANDOM FOREST

Random Forest adalah metode pembelajaran Ensemble untuk klasifikasi dan regresi, dengan membangun beberapa pohon keputusan. Random Forest cukup cepat dan mudah digunakan. Random Forest dapat menangani data yang jarang dan dengan Random Forest kita dapat mengatasi masalah overfitting. Random Forest dapat mengambil subset (sampel) data yang berbeda dengan penggantian dan bahkan dapat mengambil sampel fitur, artinya Random Forest melakukan Pengambilan Sampel Data (Observasi) serta pengambilan sampel Fitur (Variabel). Akhirnya, keputusan diambil dengan pemungutan suara mayoritas.

Dalam pohon keputusan, satu pohon keputusan dibangun dan dalam algoritma random forest, banyak pohon keputusan dibangun selama proses berlangsung. Suara dari

masing-masing pohon keputusan dipertimbangkan dalam memutuskan kelas akhir dari suatu kasus atau objek, ini disebut proses ensemble. Karena banyak pohon keputusan dibangun dan digunakan dalam proses algoritma Random Forest, maka disebut Forest. Kita tahu bahwa bingkai data memiliki dua dimensi 1. Baris dan 2. Kolom. Untuk sebuah bangunan, pohon keputusan, sampel kerangka data dipilih dengan penggantian bersama dengan pemilihan subset Kolom untuk setiap pohon keputusan. Baik pengambilan sampel kerangka data (Pengambilan Sampel Data) dan pemilihan subset variabel (Pengambilan Sampel Fitur) dilakukan secara acak. Jadi, kami menyebutnya Hutan Acak. Hutan acak meningkatkan akurasi prediktif dengan menghasilkan sejumlah besar pohon bootstrap berdasarkan sampel variabel acak, mengklasifikasikan kasus menggunakan setiap pohon di "hutan" baru ini, dan memutuskan hasil prediksi akhir dengan menggabungkan hasil di semua pohon.

18.4 MEMBANGUN MODEL MENGGUNAKAN RANDOM FOREST

Langkah 1: Instal dan Muat Paket dan Pustaka yang Diperlukan

```
install.packages('randomForest')
library(randomForest)
```

Langkah 2: Baca data dan buat kerangka data.

```
Diab<-read.csv(file="D:/R data/Diab.csv",header = T)
```

Langkah 3: Jelajahi kerangka data

```
fix(Diab)
str(Diab)
```

Langkah 4: Tetapkan benih untuk membuat hasil yang dapat direproduksi

```
set.seed(4848)
```

Langkah 5: Buat Set Data Pelatihan dan Pengujian Train(70%), Test(30%).

#Ambil sampel acak 70% dari rekaman untuk data Train

```
train = sample(1:500,350)
train_data = Diab[train,]
nrow(train_data)
```

#Ambil sampel acak 30% dari rekaman untuk data uji

```
test = (1:500)
[-train] test_data = Diab[test,]
nrow(test_data)
```

Langkah 6: Bangun Model dengan menggunakan Algoritma Random Forest

```
fit <- randomForest(as.factor(Diabetic) ~ Gender + Age + OGTT +
  DBP + BMI, data=train_data, importance=TRUE, ntree=400)
```

Langkah 7: Periksa variabel apa yang penting:

```
varImpPlot(fit)
```

Langkah 8: Validasi Model Anda dengan Memprediksi data yang tidak terlihat

```
Prediction <- predict(fit, test_data)
Final <- data.frame(Id = test_data$Pat_Id, Diabetic = Prediksi)
fix(Final)
```

Langkah 9: Jika Anda tidak puas dengan hasil model, Anda dapat mencoba pohon inferensi bersyarat, yang membuat keputusan menggunakan uji statistik alih-alih ukuran kemurnian.

```
install.packages('party')
library(party)
fit <- cforest(as.factor(Diabetic) ~ Gender + Age + OGTT + DBP +
  BMI, data=train_data, controls=cforest_unbiased(ntree=700,
  mtry=3))
Prediksi <- predict(fit, test_data, OOB=TRUE, type = "response")
```

Di random forest, kami memaksa model untuk memprediksi klasifikasi kami dengan mengubah sementara variabel target kami ke faktor dengan hanya dua level menggunakan as.factor(). Argumen importance=TRUE memungkinkan kami untuk memeriksa pentingnya variabel, dan argumen ntree menentukan berapa banyak pohon yang ingin kami kembangkan. Jika Anda bekerja dengan kumpulan data yang lebih besar, Anda mencoba dengan jumlah pohon yang lebih sedikit, atau membatasi kompleksitas setiap pohon menggunakan nodesize serta mengurangi jumlah baris yang diambil sampelnya dengan sampsize. Anda juga dapat mengganti jumlah variabel default untuk dipilih dengan mtry, tetapi defaultnya adalah akar kuadrat dari jumlah total yang umumnya berfungsi dengan baik. Kita dapat menggunakan replace Mengambil Benar dan Salah dan menunjukkan apakah akan mengambil sampel dengan/tanpa penggantian proximity Apakah akan menghitung ukuran proximity antara baris opsi bingkai data.

Estimasi Kesalahan Out of Bag (OOB): Di hutan acak, tidak perlu validasi silang atau set pengujian terpisah untuk mendapatkan estimasi kesalahan set pengujian yang tidak bias. Setiap pohon dibangun menggunakan sampel bootstrap yang berbeda dari data. Sekitar 1/3 dari kasus tidak dimasukkan ke dalam sampel bootstrap dan tidak digunakan dalam konstruksi pohon Kth. Masukkan setiap kasus yang tidak dimasukkan dalam konstruksi pohon kth ke bawah pohon kth untuk mendapatkan klasifikasi. Dengan cara ini, klasifikasi set pengujian diperoleh untuk setiap kasus di sekitar 1/3 pohon. Di akhir pengujian, ambil j sebagai kelas yang memperoleh suara terbanyak setiap kali kasus n adalah oob. Proporsi waktu j tidak sama dengan kelas n sebenarnya yang dirata-ratakan pada semua kasus adalah estimasi kesalahan oob, ini telah terbukti tidak bias dalam banyak pengujian.

Kekurangan dalam Pentingnya Variabel Hutan Acak: Hutan Acak sangat populer sebagai teknik pemilihan variabel. Namun, ia juga memiliki beberapa kekurangan. Jika variabel independen memiliki tipe yang berbeda, ukuran pentingnya variabel hutan acak dapat menyesatkan. Jika semua variabel independen bersifat kategoris tetapi memiliki kategori yang berbeda, ukuran pentingnya variabel hutan acak dapat menyesatkan. Untuk mengatasi kedua masalah di atas, kita harus menggunakan hutan inferensi bersyarat, yaitu cforest. Jika variabel

independen saling berkorelasi, ukuran kepentingan variabel random forest dapat menyesatkan. Bahkan, conditional forest tidak menghilangkan masalah Multikolinearitas sepenuhnya. Ia memecahkan masalah kolinearitas sampai batas tertentu.

18.5 BOOSTING

Boosting adalah cara yang lebih sistematis untuk meningkatkan kinerja melalui penggabungan berbagai pengklasifikasi. Pertimbangkan untuk membuat pengklasifikasi tiga komponen untuk masalah dua kategori melalui boosting.

1. Pilih secara acak n1 < n sampel dari D tanpa penggantian untuk mendapatkan D1 dan latih pembelajar lemah C1.
2. Pilih n2 < n sampel dari D dengan setengah dari sampel salah diklasifikasikan oleh C1 untuk mendapatkan D2 dan latih pembelajar lemah C2.
3. Pilih semua sampel yang tersisa dari D yang tidak disetujui C1 dan C2 dan latih pembelajar lemah C3. Pengklasifikasi akhir adalah suara pembelajar lemah.
4. Jika perkiraan bootstrap yang direplikasi benar, maka bagging akan mengurangi varians tanpa mengubah bias. Dalam praktiknya, bagging dapat mengurangi bias dan varians. Untuk pengklasifikasi bias tinggi, bagging dapat mengurangi bias dan untuk pengklasifikasi varians tinggi, bagging dapat mengurangi varians.

Adaboost: Dalam Adaboost, alih-alih mengambil sampel setiap saat, bobot ditetapkan untuk setiap sampel. Bobot adalah probabilitas sampel untuk dipilih dalam pengklasifikasi. Bobot ini bekerja dengan pengklasifikasi biner yang lebih baik daripada lemparan koin acak (kesalahan kurang dari 0,5). Idenya adalah untuk menyesuaikan bobot sedemikian rupa sehingga rekaman yang salah diklasifikasikan akan dipilih untuk tingkat klasifikasi kedua oleh pengklasifikasi kedua. Jadi, jika pengklasifikasi membuat kesalahan dalam memprediksi variabel, bobot variabel meningkat. Kemudian pengklasifikasi didefinisikan sebagai kombinasi linier dari semua pengklasifikasi yang lemah. Hasilnya adalah mode prediksi semua pengklasifikasi.

Misalnya, kita memiliki kumpulan data (x1,y1),…(xm,ym) di mana x adalah nilai dan y adalah apakah data tersebut dipilih oleh pengklasifikasi. Jadi, y mengambil -1 atau 1. Kita mulai dengan bobot awal, Kita memilih pengklasifikasi sedemikian rupa sehingga kesalahan sehubungan dengan distribusi minimal dan kurang dari 0,5. Pengklasifikasi yang lemah memiliki kesalahan kurang dari 50% tetapi masih belum memuaskan (50% merupakan kasus lemparan koin dan karenanya kesalahan lebih dari 50% tidak diperbolehkan bahkan untuk pengklasifikasi yang lemah).

$$\alpha_t = \frac{1}{2} log \frac{1-\epsilon_t}{\epsilon_t} \geq 0$$

Di sini, ini adalah kesalahan pengklasifikasi individu. Setelah kita mengklasifikasikan dengan pengklasifikasi lemah pertama, kita memperbarui bobot sedemikian rupa sehingga

$$D_{t+1}(i) = \frac{D_t(i)}{Z_t} * \begin{cases} e^{-\alpha_t} jika\ h_t(x_i) = y_i \\ e^{\alpha_t}\ jika\ h_t(x_i) \neq y_i \end{cases}$$

$$= \frac{D_t(i) e^{-\alpha_t y_t h_t(x_i)}}{Z_t}$$

Zt adalah faktor normalisasi untuk memastikan bahwa D selalu merupakan distribusi. Jadi, jika bidang tertentu diklasifikasikan dengan benar, suku eksponensialnya rendah (karena y positif). Oleh karena itu, bobotnya rendah. Untuk bidang yang salah diklasifikasikan, y negatif dan karenanya eksponensialnya besar dan karenanya bobotnya naik. Kami melanjutkan dengan semua pengklasifikasi yang lemah dan pada setiap tahap, kami memilih pengklasifikasi yang meminimalkan kesalahan. Dengan cara ini, ini adalah algoritma greedy.

Pada iterasi awal, boosting terutama merupakan metode pengurangan bias dan pada iterasi selanjutnya, tampaknya terutama merupakan metode pengurangan varians. Kesalahan pelatihan didefinisikan pada setiap putaran sebagai fraksi pengamatan yang salah diklasifikasikan.

$$\frac{\sum_{i=1}^{m} (H(x_i) \neq y_i)}{m}$$

Kesalahan pelatihan turun secara eksponensial dengan cepat. Pilih αt yang meminimalkan Zt. Tidak ada parameter yang harus disetel (kecuali jumlah putaran). Cepat, sederhana, dan mudah diprogram. Dilengkapi dengan serangkaian jaminan teoritis (misalnya, kesalahan pelatihan, kesalahan pengujian). Daripada mencoba merancang algoritme pembelajaran yang akurat di seluruh ruang, kita dapat fokus menemukan algoritme pembelajaran dasar yang hanya perlu lebih baik daripada acak. *Algoritme ini dapat mengidentifikasi outlier*: yaitu contoh yang salah diberi label atau yang secara inheren ambigu dan sulit dikategorikan. Namun, kinerja aktual dari boosting bergantung pada data dan pembelajar dasar. Boosting tampaknya sangat rentan terhadap gangguan.

Bagging dan Boosting: Bagging selalu menggunakan resampling daripada reweighting. Bagging tidak mengubah distribusi atas contoh atau kesalahan label, tetapi sebaliknya selalu menggunakan distribusi seragam. Dalam membentuk hipotesis akhir, bagging memberikan bobot yang sama untuk setiap hipotesis yang lemah.

18.6 MEMBANGUN MODEL MENGGUNAKAN ADABOOST

 # Langkah 1: Memuat Data ke R:

```
setwd("D:/R data") loandata=read.csv(file="Housing_loan.csv",
header=TRUE)
```

 # Langkah 2: Persiapan Data:

Hapus kolom ID & Gender dari data

```
loandata2=subset(loandata, select=-c(ID, Gender))
fix(loandata2)
```

 # Langkah 3: Buat variabel dummy untuk variabel kategorikal "Pendidikan" dan tambahkan variabel dummy tersebut ke data asli.

#Instal & Muat paket "dummies" untuk membuat variabel dummy

```
install.packages("dummies")
library(dummies)
Edu_dum=dummy(loandata2$Education)
head(Edu_dum)
loandata3=subset(loandata2,select=-c(Education))
loandata4=cbind(loandata3,Edu_dum)
fix(loandata4)
```

 # Langkah 4: Standardisasi Data:

Standarisasi data menggunakan metode 'Range'

```
install.packages("vegan")
library(vegan)
loandata5=decostand(loandata4,"range")
```

 # Langkah 5: Siapkan set data train & test

#Ambil sampel acak 60% dari rekaman untuk data train

```
train = sample(1:1000,600)
train_data = loandata5[train,]
nrow(train_data)
```

#Ambil sampel acak 40% dari rekaman untuk data uji

```
test = (1:1000) [-train]
test_data = loandata5[test,]
nrow(test_data)
```

 # Langkah 6: Ringkasan Data untuk variabel respons "Loan_sanctioned":

#Total Data

```
table(loandata5$Loan_sanctioned)
```

#Train Data

```
table(train_data $Loan_sanctioned)
```

#Test Data

```
table(test_data$Loan_sanctioned)
```

 # Langkah 7: Klasifikasi menggunakan Adaboost:

#Instal & Muat paket — ada untuk melakukan analisis SVM.

```
install.packages("ada")
library(ada)
x = subset(train_data, select = -Loan_sanctioned)
```

```
y = as.factor(train_data$Loan_sanctioned)
Ada_20=ada(x,y,iter=20,nu=1,loss="logistic", type="discrete")
summary(Ada_20)
```

 # Langkah 8: Tambahkan set data pengujian

```
a = subset(test_data, select=-Loan_sanctioned)
b = as.factor(test_data$Loan_sanctioned)
Ada_t20=addtest(Ada_20,a,b)
pred = predict(Ada_t20, a)
table(pred, b)
```

 # Langkah 9: Plot Ada_t20

```
plot(Ada_t20,TRUE,TRUE)
```

 # Langkah 10: Coba dengan 50 Iterasi

```
Ada_50=ada(x,y,iter=50,nu=1,loss="logistic", type="discrete")
summary(Ada_50)
Ada_t50=addtest(Ada_50,a,b)
pred = predict(Ada_t50, a)
table(pred, b)
```

#Plot Ada_50

```
plot(Ada_t50,TRUE,TRUE)
```

#Langkah 11: Coba dengan Iterasi yang berbeda seperti iter=100,500,1000 untuk memeriksa akurasi dan memperbaiki model.frame()

DAFTAR PUSTAKA

Abbott, Dean. 2014. Applied Predictive Analytics: Principles and Techniques for the Professional Data Analyst. Hoboken, NJ: Wiley.

Ariely, Dan. 2008. Predictably Irrational: The Hidden Forces That Shape Our Decisions. New York, NY: HarperCollins.

Barocas, Solon, dan Andrew D. Selbst. 2016. Big Data's Disparate Impact. New York, NY: New York University Press.

Berson, Alex, dan Stephen J. Smith. 2010. Data Warehousing, Data Mining, & OLAP. New York, NY: McGraw-Hill Education.

Bihani, Pradeep, dan Atul Kumar. 2018. Big Data and Predictive Analytics: Solutions and Applications. New Delhi: Wiley.

Bruce, Peter, Andrew Bruce, dan Peter Gedeck. 2016. Practical Statistics for Data Scientists: 50 Essential Concepts. Sebastopol, CA: O'Reilly Media.

Buyya, Rajkumar, Cyrus Shahabi, dan Suresh Chandra Satapathy. 2020. Big Data Analytics: Tools and Techniques for Insights and Optimization. Amsterdam: Elsevier.

Caffo, Brian. 2017. Data Science for Economists. Cambridge: MIT Press.

Casad, Joe M. 2014. Data Science for Dummies. Hoboken, NJ: Wiley.

Chen, Shyi-Ming, dan Zongmin Ma. 2017. Big Data Analytics and Decision Making: Fundamentals and Techniques. Berlin: Springer.

Choi, Seung-Kyu, dan Jae-Hyuk Yang. 2019. Fundamentals of Big Data and Analytics: A Comprehensive Overview. Cambridge: Cambridge University Press

Cukier, Kenneth, dan Viktor Mayer-Schönberger. 2014. The Big Data Revolution: How Data Analytics Is Transforming the Future of Business. London: Penguin Books.

Davenport, Thomas H., dan Jill Dyché. 2013. Big Data at Work: Dispelling the Myths, Uncovering the Opportunities. Boston, MA: Harvard Business Review Press.

Downey, Allen B. 2016. Think Stats: Probability and Statistics for Programmers. Sebastopol, CA: O'Reilly Media.

Ekstrand, Michael D., John Riedl, dan Dan Cosley. 2011. Recommender Systems: Challenges and Opportunities. New York, NY: ACM Press.

Foreman, John W. 2014. Data Smart: Using Data Science to Transform Information into Insight. Hoboken, NJ: Wiley.

Franks, Bill. 2012. Taming the Big Data Tidal Wave: Finding Opportunities in Huge Data Sets. Hoboken, NJ: Wiley.

Gorve, Kiran. 2013. Big Data Analytics: A Hands-On Approach. Delhi: Pearson.

Hu, Xiaojin, dan Sanjay Chaudhuri. 2015. Big Data Analysis: Methodologies and Applications. New York, NY: Springer.

Jones, Elizabeth A. 2018. Big Data Analytics: A Practical Guide for Managers. London: Routledge.

Kelleher, John D., dan Brendan Tierney. 2018. Data Science: An Introduction. Cambridge: Cambridge University Press.

Kumar, Anil, dan Bhanu Prakash. 2017. Big Data Analytics: A Comprehensive Guide for Beginners. New Delhi: Wiley.

Lang, Michael, dan Peter H. Han. 2017. Big Data Analytics in Healthcare: Methods and Applications. New York, NY: Springer.

Lee, John, dan Hwang Chun. 2019. Big Data and Cloud Computing: A Practical Guide. New York, NY: Springer.

Leskovec, Jure, Anand Rajaraman, dan Jeffrey Ullman. 2014. Mining of Massive Datasets. Cambridge: Cambridge University Press.

Li, Guojun, dan Zhiqiang Wei. 2018. Big Data Computing and Communications: A Comprehensive Guide. London: Wiley.

Li, Kuan, dan Yi Li. 2017. Big Data Management: Concepts, Techniques, and Applications. Berlin: Springer.

Linoff, Gordon S., dan Michael S. Berry. 2011. Data Mining Techniques: For Marketing, Sales, and Customer Relationship Management. Hoboken, NJ: Wiley.

Long, J.D. 2021. Fundamentals of Data Visualization: A Primer on Creating Interactive Data Graphics. Boston, MA: O'Reilly Media.

Loshin, David. 2013. Big Data Analytics: From Strategic Planning to Enterprise Integration with Tools, Techniques, NoSQL, and Graph. Burlington, MA: Morgan Kaufmann.

Lucas, Michael W. 2020. Data Science from Scratch: First Principles with Python. New York, NY: No Starch Press.

Malec, Michael S. 2019. Advanced Big Data Analytics: Techniques and Technologies. Chicago, IL: Springer.

Marr, Bernard. 2015. Big Data in Practice: How 45 Successful Companies Used Big Data Analytics to Deliver Extraordinary Results. Chichester: Wiley.

Mayer-Schönberger, Viktor, dan Kenneth Cukier. 2013. Big Data: A Revolution That Will Transform How We Live, Work, and Think. Boston, MA: Eamon Dolan/Houghton Mifflin Harcourt.

Miller, James D. 2014. Big Data: A Practical Guide for Managers. London: Kogan Page.

Minelli, Michael, Michele Chambers, dan Ambiga Dhiraj. 2013. Big Data, Big Analytics: Emerging Business Intelligence and Analytic Trends for Today's Businesses. Hoboken, NJ: Wiley.

Müller, Andreas C., dan Sarah Guido. 2016. Introduction to Machine Learning with Python: A Guide for Data Scientists. Sebastopol, CA: O'Reilly Media.

O'Reilly, Tim. 2017. Data-Driven: Creating a Data Culture. Sebastopol, CA: O'Reilly Media.

Pal, Sankar K., dan Paul G. Ranjan. 2015. Data Mining and Predictive Analytics. Berlin: Springer.

Perlich, Claudia A., Foster Provost, dan Tom Fawcett. 2017. Data Science for Business: What You Need to Know about Data Mining and Data-Analytic Thinking. New York, NY: O'Reilly Media.

Queneau, Claude T. E. 2020. Introduction to Data Science: Big Data, Data Mining, and Machine Learning. New York, NY: Springer.

Ronanki, Rajeev. 2016. Big Data: Concepts, Technology, and Architecture. Berlin: Springer.

Siegel, Eric. 2013. Predictive Analytics: The Power to Predict Who Will Click, Buy, Lie, or Die. Hoboken, NJ: Wiley.

Stikeleather, Jim. 2017. Data Analytics for Beginners: How to Get Started with Data Analysis. New York, NY: Independently Published.

Tan, Pang-Ning, Michael Steinbach, dan Vipin Kumar. 2013. Introduction to Data Mining. Boston, MA: Pearson.

White, Tom. 2015. Hadoop: The Definitive Guide. Sebastopol, CA: O'Reilly Media.

Witten, Ian H., Eibe Frank, dan Mark A. Hall. 2016. Data Mining: Practical Machine Learning Tools and Techniques. Cambridge: Morgan Kaufmann.

Xu, Hui, dan Yan Liu. 2020. Data Science: Theories and Applications. New York, NY: Elsevier.

Zhou, Lei, dan Qun Liu. 2016. Advanced Big Data Analytics: Theory and Practice. Berlin: Springer.

Zikopoulos, Paul, dan Chris Eaton. 2011. Understanding Big Data: Analytics for Enterprise Class Hadoop and Streaming Data. New York, NY: McGraw-Hill.

Cara Mudah Menganalisis Big Data

Dr. Agus Wibowo, M.Kom, M.Si, MM.

BIO DATA PENULIS

Penulis memiliki berbagai disiplin ilmu yang diperoleh dari Universitas Diponegoro (UNDIP) Semarang. dan dari Universitas Kristen Satya Wacana (UKSW) Salatiga. Disiplin ilmu itu antara lain teknik elektro, komputer, manajemen dan ilmu sosiologi. Penulis memiliki pengalaman kerja pada industri elektronik dan sertifikasi keahlian dalam bidang Jaringan Internet, Telekomunikasi, Artificial Intelligence, Internet Of Things (IoT), Augmented Reality (AR), Technopreneurship, Internet Marketing dan bidang pengolahan dan analisa data (komputer statistik).

Penulis adalah pendiri dari Universitas Sains dan Teknologi Komputer (Universitas STEKOM) dan juga seorang dosen yang memiliki Jabatan Fungsional Akademik Lektor Kepala (Associate Professor) yang telah menghasilkan puluhan Buku Ajar ber ISBN, HAKI dari beberapa karya cipta dan Hak Paten pada produk IPTEK. Sejak tahun 2023 penulis tercatat sebagai Dosen luar biasa di Fakultas Ekonomi & Bisnis (FEB) Universitas Diponegoro Semarang. Penulis juga terlibat dalam berbagai organisasi profesi dan industri yang terkait dengan dunia usaha dan industri, khususnya dalam pengembangan sumber daya manusia yang unggul untuk memenuhi kebutuhan dunia kerja secara nyata.

YAYASAN PRIMA AGUS TEKNIK

PENERBIT :
YAYASAN PRIMA AGUS TEKNIK
Jl. Majapahit No. 605 Semarang
Telp. (024) 6723456. Fax. 024-6710144
Email : penerbit_ypat@stekom.ac.id

ISBN 978-623-8642-29-8 (PDF)
9 786238 642298