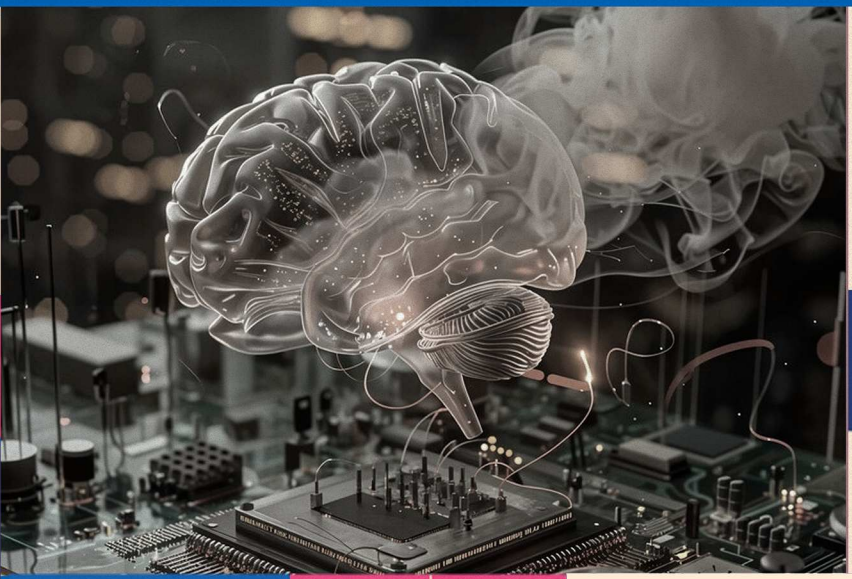
KECERDASAN KOMPUTASIONAL

Victor Benny Alexsius Pardosi, S.Kom., M.Sc
Ir. Heliza Rahmania Hatta, S.Kom., M.Kom., IPP
Dr. Ir. N. Tri S. Saptadi, S.Kom., MT., MM., IPM
Joko Tri Haryanta, ST., MT
Bulkis Kanata, ST., MT
Dr. Widyastuti Andriyani, S.Kom., M.Kom
Dr. Ilham, ST., MT
Mustovia Azahro, S.T., M.T
Ir. Muhammad Azhar Irwansyah, ST., M.Eng
Dr. Ir. Masduki Zakarijah, M.T



CV REY MEDIA GRAFIKA

KECERDASAN KOMPUTASIONAL

Penulis:
Victor Benny Alexsius Pardosi, S.Kom., M.Sc
Ir. Heliza Rahmania Hatta, S.Kom., M.Kom., IPP
Dr. Ir. N. Tri S. Saptadi, S.Kom., MT., MM., IPM
Joko Tri Haryanta, ST., MT
Bulkis Kanata, ST., MT
Dr. Widyastuti Andriyani, S.Kom., M.Kom
Dr. Ilham, ST., MT
Mustovia Azahro, S.T., M.T
Ir. Muhammad Azhar Irwansyah, ST., M.Eng
Dr. Ir. Masduki Zakarijah, M.T

CV.REY MEDIA GRAFIKA
PUBLISHER

KECERDASAN KOMPUTASIONAL

Penulis :
Victor Benny Alexsius Pardosi, S.Kom., M.Sc
Ir. Heliza Rahmania Hatta, S.Kom., M.Kom., IPP
Dr. Ir. N. Tri S. Saptadi, S.Kom., MT., MM., IPM
Joko Tri Haryanta, ST., MT
Bulkis Kanata, ST., MT
Dr. Widyastuti Andriyani, S.Kom., M.Kom
Dr. Ilham, ST., MT
Mustovia Azahro, S.T., M.T
Ir. Muhammad Azhar Irwansyah, ST., M.Eng
Dr. Ir. Masduki Zakarijah, M.T

Penyunting dan Desain Cover :
Paput Tri Cahyono

Ukuran:
x hal + 196 hal; 14,8cm x 21cm

Diterbitkan Oleh :

CV.REY MEDIA GRAFIKA
PUBLISHER

Jln.Melati, BKG. Palapa, Blok.T No.6
Batam - Indonesia 29432
Email : reymediagrafika.rgm@gmail.com

ISBN : 978-623-8609-57-4
IKAPI: 010/Kepri/2022
Terbitan:

KATA PENGANTAR

Syukur *alhamdulillah* penulis haturkan kepada Allah Swt. yang senantiasa melimpahkan karunia dan berkah-Nya sehingga penulis mampu merampungkan karya ini tepat pada waktunya, sehingga penulis dapat menghadirkannya dihadapan para pembaca. Kemudian, tak lupa *shalawat* dan salam semoga senantiasa tercurah limpahkan kepada Nabi Muhammad SAW, para sahabat, dan ahli keluarganya yang mulia.

Kecerdasan komputasional mencakup berbagai metode yang diilhami oleh proses biologis dan sosial dalam memecahkan masalah-masalah kompleks yang sulit diselesaikan dengan cara tradisional. Dengan berkembangnya teknologi, pendekatan ini telah diterapkan di berbagai bidang, seperti pengolahan data, optimisasi, pengambilan keputusan, hingga kecerdasan buatan.

Buku ini mengupas berbagai teknik dalam kecerdasan komputasional, seperti jaringan saraf tiruan, algoritma genetika, sistem fuzzy, dan swarm intelligence, yang dijelaskan secara rinci dengan contoh penerapan nyata di dunia industri dan penelitian. Penyusunan buku ini dilakukan dengan bahasa yang mudah dipahami, sehingga diharapkan dapat

membantu mahasiswa, praktisi, dan peneliti untuk memahami konsep-konsep tersebut secara mendalam.

Penulis menyampaikan terima kasih yang tak terhingga bagi semua pihak yang telah berpartisipasi. Terakhir seperti kata pepatah bahwa" Tiada Gading Yang Tak Retak" maka penulisan buku ini juga jauh dari kata sempurna, oleh karena itu penulis sangat berterima kasih apabila ada saran dan masukkan yang dapat diberikan guna menyempurnakan buku ini di kemudian hari.

2024

Penulis

DAFTAR ISI

BAB I
KONSEP DASAR KECERDASAN BUATAN DAN *SOFT COMPUTING*

1.1. Definisi dan Ruang Lingkup Kecerdasan Buatan (AI)

Definisi Kecerdasan Buatan (AI): Kecerdasan Buatan (AI) merujuk pada kemampuan sistem komputer atau mesin untuk meniru kecerdasan manusia dalam menyelesaikan tugas-tugas tertentu. AI bertujuan untuk menciptakan mesin yang dapat berpikir, belajar, dan beradaptasi seperti manusia. Ini melibatkan pengembangan algoritma dan model yang memungkinkan komputer untuk melakukan tugas-tugas seperti pemrosesan bahasa alami, pengenalan pola, pengambilan keputusan, dan pemecahan masalah.

Ruang Lingkup AI: AI mencakup berbagai subbidang dan teknik yang dirancang untuk mensimulasikan berbagai aspek kecerdasan manusia. Beberapa area utama dalam ruang lingkup AI meliputi:

- Pembelajaran Mesin (Machine Learning): Teknik di mana komputer belajar dari data untuk membuat prediksi atau keputusan tanpa

diprogram secara eksplisit. Contohnya termasuk algoritma klasifikasi, regresi, dan *Clustering*.

- Pembelajaran Mendalam (Deep Learning): Subset dari pembelajaran mesin yang menggunakan jaringan saraf tiruan dengan banyak lapisan untuk memodelkan data yang kompleks. Ini sering digunakan dalam pengenalan gambar, pemrosesan bahasa alami, dan tugas-tugas lain yang memerlukan pemahaman tingkat tinggi.
- Pemrosesan Bahasa Alami (Natural Language Processing, NLP): Bidang yang berfokus pada interaksi antara komputer dan bahasa manusia. Ini termasuk penerjemahan mesin, analisis sentimen, dan chatbots.
- Robotika: Pengembangan robot yang dapat melakukan tugas fisik atau interaksi dengan lingkungan secara otonom. Ini melibatkan integrasi AI untuk navigasi, pengenalan objek, dan pengambilan keputusan.
- Sistem Pakar: Sistem berbasis komputer yang meniru kemampuan keputusan seorang ahli di bidang tertentu, menggunakan basis pengetahuan dan aturan inferensi untuk menyelesaikan masalah.

- Visi Komputer: Teknologi yang memungkinkan komputer untuk "melihat" dan memproses gambar atau video. Ini termasuk deteksi objek, pengenalan wajah, dan pelacakan gerakan.

1.2. Sejarah dan Evolusi AI

Awal Mula AI:

- 1950-an: Kecerdasan Buatan mulai dikembangkan sebagai disiplin ilmiah pada tahun 1950-an. Alan Turing, seorang ilmuwan komputer Inggris, memperkenalkan Tes Turing pada tahun 1950 sebagai cara untuk mengukur kemampuan mesin untuk menunjukkan perilaku yang tidak dapat dibedakan dari manusia. Konferensi Dartmouth pada tahun 1956 diadakan oleh John McCarthy, Marvin Minsky, Nathaniel Rochester, dan Claude Shannon, yang dianggap sebagai titik awal resmi dari bidang AI.

1960-an hingga 1970-an:

- Eksplorasi Awal: Pada periode ini, AI awalnya difokuskan pada pengembangan sistem berbasis aturan dan permainan seperti catur. Sistem pakar awal, seperti DENDRAL dan MYCIN,

dikembangkan untuk aplikasi spesifik dalam analisis kimia dan diagnosis medis.

1980-an:

- Musim Dingin AI Pertama: Kemajuan dalam AI melambat pada 1970-an dan awal 1980-an, periode yang dikenal sebagai "musim dingin AI" karena kurangnya kemajuan yang signifikan dan kesulitan dalam mencapai ekspektasi yang tinggi. Namun, perkembangan dalam teknik seperti jaringan saraf tiruan dan algoritma pembelajaran mendalam mulai mendapatkan momentum pada akhir dekade ini.

1990-an hingga 2000-an:

- Kebangkitan AI: Pada tahun 1990-an, AI mengalami kebangkitan kembali berkat kemajuan dalam komputasi, data besar, dan algoritma. Pencapaian penting termasuk kemenangan Deep Blue, komputer catur IBM, atas juara dunia Garry Kasparov pada tahun 1997. Pengembangan sistem seperti ASR (Automatic Speech Recognition) dan aplikasi AI komersial menjadi lebih umum.

2010-an hingga Sekarang:

- Era Deep Learning dan AI Modern: Revolusi deep learning dan pemrosesan data besar pada 2010-an mempercepat kemajuan dalam AI. Model-model seperti AlexNet, yang memenangkan kompetisi ImageNet pada tahun 2012, memicu ledakan dalam aplikasi pengenalan gambar dan pemrosesan bahasa alami. AI saat ini digunakan dalam berbagai aplikasi sehari-hari, termasuk asisten virtual seperti Siri dan Alexa, sistem rekomendasi, dan mobil otonom.

Masa Depan AI:

- Tren Terbaru: AI terus berkembang dengan kemajuan dalam teknik-teknik seperti transfer learning, reinforcement learning, dan AI yang lebih interpretatif. Penelitian sedang dilakukan untuk mengatasi tantangan etika dan sosial, seperti bias dalam algoritma dan dampak AI pada pekerjaan.

AI telah berkembang dari eksperimen akademis menjadi bagian integral dari kehidupan modern, dengan aplikasi yang menjangkau hampir setiap aspek

industri dan kehidupan sehari-hari.

1.3. Definisi dan Tujuan *Soft Computing*

Definisi *Soft Computing*: *Soft Computing* adalah pendekatan komputasi yang dirancang untuk menangani masalah yang tidak dapat diselesaikan dengan metode komputasi tradisional yang kaku dan presisi. Berbeda dengan metode komputasi keras yang memerlukan model matematis yang ketat dan eksak, *Soft Computing* lebih fleksibel dan mampu menangani ketidakpastian, imprecision, dan approximasi dalam proses komputasi. *Soft Computing* mencakup berbagai teknik yang dapat beradaptasi dengan data yang tidak pasti dan tidak lengkap.

Tujuan *Soft Computing*:

1. Menangani Ketidakpastian: *Soft Computing* dirancang untuk bekerja dengan data yang tidak lengkap atau tidak pasti, menawarkan solusi ketika model matematis yang tepat sulit diterapkan.
2. Menghasilkan Solusi Aproksimasi: Tujuannya adalah untuk menemukan solusi yang cukup baik dalam waktu yang wajar, meskipun tidak selalu optimal atau eksak.

3. Meningkatkan Fleksibilitas: Teknik *Soft Computing* memungkinkan penyesuaian terhadap data yang berubah-ubah dan situasi yang kompleks, yang tidak dapat diatasi dengan metode komputasi keras.
4. Integrasi dengan Teknik Lain: *Soft Computing* sering digunakan bersama teknik lain untuk memperbaiki kinerja sistem dan menangani berbagai jenis masalah yang tidak dapat dipecahkan secara efisien dengan metode konvensional.

1.4. Perbandingan *Soft Computing* dengan *Hard Computing*

Hard Computing:

- Definisi: *Hard Computing*, atau komputasi keras, mengacu pada metode tradisional yang berbasis pada prinsip matematika dan logika formal. Teknik ini memerlukan model matematis yang tepat dan eksak.
- Kelebihan:
    - Dapat memberikan solusi yang sangat presisi dan akurat jika model yang tepat tersedia.

    - Berguna untuk masalah yang dapat didefinisikan dengan jelas dan dipecahkan dengan algoritma yang ditetapkan.
- Kekurangan:
    - Kurang fleksibel dalam menangani ketidakpastian dan data yang tidak lengkap.
    - Memerlukan model matematis yang ketat, yang tidak selalu praktis untuk semua aplikasi.

Soft Computing:

- Definisi: *Soft Computing* melibatkan teknik yang lebih fleksibel dan toleran terhadap ketidakpastian dan ketidakakuratan. Ini mencakup pendekatan seperti logika fuzzy, jaringan saraf tiruan, dan algoritma evolusi.
- Kelebihan:
    - Mampu menangani data yang tidak pasti dan tidak lengkap.
    - Lebih fleksibel dan adaptif terhadap berbagai jenis masalah.

- Dapat mengintegrasikan beberapa teknik untuk menghasilkan solusi yang lebih baik.

- Kekurangan:
  - Solusi yang dihasilkan mungkin tidak selalu optimal atau eksak.
  - Hasil dapat bervariasi tergantung pada parameter dan konfigurasi yang digunakan.

1.5. Elemen Utama *Soft Computing*

1. Jaringan Saraf Tiruan (Neural Networks):

   Jaringan saraf tiruan meniru struktur dan fungsi otak manusia, menggunakan neuron buatan untuk memproses informasi. Jaringan ini digunakan untuk pembelajaran dan pengenalan pola, serta dapat mengadaptasi dan belajar dari data.

2. Logika Fuzzy:

   Logika fuzzy adalah metode yang digunakan untuk menangani ketidakpastian dan variabilitas dalam data dengan memperkenalkan derajat keanggotaan. Ini memungkinkan sistem untuk membuat

keputusan berdasarkan informasi yang tidak jelas atau tidak pasti.

3. Algoritma Evolusi:

   Algoritma evolusi, termasuk algoritma genetika, menggunakan prinsip-prinsip evolusi biologis seperti seleksi, crossover, dan mutasi untuk mencari solusi optimal dalam ruang pencarian yang besar. Teknik ini berguna untuk optimasi dan pemecahan masalah kompleks.

4. Pemrograman Genetik (Genetic Programming):

   Pemrograman genetik adalah metode evolusi yang digunakan untuk menghasilkan program komputer atau model matematis. Ini melibatkan evolusi struktur program untuk menyelesaikan tugas tertentu dengan cara yang adaptif dan fleksibel.

5. Simulated Annealing:

   Simulated annealing adalah teknik optimasi yang meniru proses annealing dalam metalurgi untuk menemukan solusi optimal. Ini melibatkan eksplorasi ruang solusi dengan probabilitas yang menurun seiring waktu untuk menghindari perangkap lokal.

6. Algoritma Swarm Intelligence:

Algoritma swarm intelligence terinspirasi oleh perilaku kolektif hewan seperti burung atau ikan. Teknik ini menggunakan prinsip-prinsip kolaborasi dan interaksi antar individu dalam kelompok untuk menyelesaikan masalah optimasi.

1.6. Tren dan Masa Depan Kecerdasan Buatan dan *Soft Computing*

Tren Terkini dalam Kecerdasan Buatan (AI)

1. Pembelajaran Mendalam (Deep Learning) dan Model Transformer:

   Model Transformer seperti BERT dan GPT-4 telah merevolusi pemrosesan bahasa alami dengan kemampuan untuk memahami konteks dan nuansa bahasa dengan sangat baik. Deep learning terus berkembang dengan penggunaan model besar yang lebih efisien dan teknik pelatihan yang lebih canggih.

2. AI Generatif:

   AI generatif, yang mencakup model seperti Generative Adversarial Networks (GANs), menghasilkan data baru yang mirip dengan data pelatihan. Ini digunakan untuk menghasilkan

gambar, teks, dan musik baru, serta dalam simulasi dan desain produk.

3. Kecerdasan Buatan Terintegrasi:

   AI semakin sering diintegrasikan ke dalam berbagai aplikasi dan perangkat sehari-hari. Contoh termasuk asisten pribadi cerdas, sistem rekomendasi yang lebih baik, dan kendaraan otonom yang menggunakan AI untuk navigasi dan pengambilan keputusan.

4. AI Etis dan Bertanggung Jawab:

   Ada peningkatan fokus pada pengembangan AI yang etis, dengan perhatian khusus pada bias algoritma, transparansi, dan tanggung jawab. Organisasi dan pemerintah sedang mengembangkan pedoman dan regulasi untuk memastikan bahwa AI digunakan dengan cara yang adil dan bertanggung jawab.

5. AI dalam Kesehatan dan Bioteknologi:

   AI digunakan untuk menganalisis data medis, memprediksi penyakit, dan merancang obat-obatan baru. Teknologi seperti pengenalan gambar medis dan analisis genomik didorong oleh kemajuan AI, membantu dokter dan peneliti dalam diagnosis dan penelitian.

6. Komputasi Kuantum dan AI:

Penelitian dalam komputasi kuantum bertujuan untuk meningkatkan kapasitas komputasi secara eksponensial, yang dapat mempercepat proses pelatihan model AI dan menyelesaikan masalah yang saat ini terlalu kompleks untuk komputer klasik.

1.7. Tren Terkini dalam *Soft Computing*

1. Integrasi Metode *Soft Computing*:
   Ada tren menuju integrasi berbagai metode *Soft Computing*, seperti kombinasi logika fuzzy dengan jaringan saraf tiruan untuk meningkatkan fleksibilitas dan kinerja sistem dalam menghadapi masalah kompleks.
2. Optimasi Berbasis Metaheuristik:
   Teknik optimasi berbasis metaheuristik, seperti algoritma swarm intelligence (misalnya, Particle Swarm Optimization), semakin digunakan untuk menyelesaikan masalah optimasi yang sulit dipecahkan dengan metode tradisional.
3. Pengembangan Model Fuzzy dan Hybrid:
   Model fuzzy dan teknik hybrid yang menggabungkan logika fuzzy dengan metode lain seperti pembelajaran mesin atau optimasi evolusi semakin populer untuk menangani

ketidakpastian dan kompleksitas dalam sistem yang dinamis.

4. *Soft Computing* dalam IoT dan Smart Systems:
   Soft Computing digunakan untuk mengelola data besar dan membuat keputusan dalam sistem Internet of Things (IoT) dan sistem pintar. Ini termasuk penggunaan teknik fuzzy untuk manajemen dan pengendalian perangkat yang terhubung dalam jaringan.
5. Penerapan di Industri dan Keuangan:
   Soft Computing diterapkan dalam industri dan keuangan untuk prediksi pasar, manajemen risiko, dan kontrol kualitas. Teknik seperti logika fuzzy dan algoritma evolusi digunakan untuk mengatasi ketidakpastian dan variabilitas dalam data industri dan keuangan.

Masa Depan Kecerdasan Buatan dan *Soft Computing*

1. Kecerdasan Buatan Umum (AGI):
   Meskipun masih dalam tahap awal penelitian, pengembangan Kecerdasan Buatan Umum (AGI) yang dapat melakukan tugas dengan kecerdasan yang setara dengan manusia di berbagai domain terus menjadi area fokus. AGI berpotensi

membawa perubahan besar dalam cara kita berinteraksi dengan teknologi.

2. Peningkatan Efisiensi dan Etika:

   Masa depan AI akan melibatkan peningkatan efisiensi algoritma dan perangkat keras, serta perhatian yang lebih besar pada etika penggunaan AI. Ini termasuk mengembangkan teknologi yang dapat menjelaskan keputusan mereka (explainable AI) dan memastikan penggunaan AI yang adil dan etis.

3. Integrasi dengan Teknologi Baru:

   Integrasi AI dan *Soft Computing* dengan teknologi baru seperti blockchain dan komputasi kuantum dapat membuka peluang baru dan meningkatkan kemampuan sistem dalam mengelola data dan membuat keputusan yang lebih baik.

4. Pengembangan AI Kecil dan Edge AI:

   Ada dorongan menuju pengembangan AI yang dapat berjalan pada perangkat kecil dan di edge (tepi jaringan), seperti perangkat IoT. Ini akan memungkinkan pengolahan data secara real-time dan pengambilan keputusan langsung di tempat tanpa perlu mengirim data ke cloud.

5. Revolusi dalam Kesehatan dan Bioteknologi:

AI dan *Soft Computing* diharapkan membawa revolusi lebih lanjut dalam bidang kesehatan dan bioteknologi, dengan aplikasi yang lebih personalisasi dalam pengobatan, serta inovasi dalam penelitian dan pengembangan bioteknologi.

1.8. Kecerdasan Komputasional dan Cloud *Computing*

Seiring dengan perkembangan teknologi, kebutuhan untuk mengolah data dalam jumlah besar dan menjalankan algoritma kecerdasan buatan secara efisien semakin meningkat. Cloud *Computing* telah menjadi solusi utama dalam mengatasi tantangan tersebut. Cloud *Computing* tidak hanya menyediakan infrastruktur komputasi yang kuat, tetapi juga mendukung berbagai layanan dan platform yang dirancang khusus untuk mengoptimalkan Kecerdasan Komputasional dan *Soft Computing*.

- Peran Cloud *Computing* dalam Kecerdasan Buatan

  Kecerdasan Komputasional, termasuk teknik seperti jaringan saraf tiruan, pembelajaran mesin, dan logika fuzzy, memerlukan daya komputasi yang besar, terutama ketika

digunakan untuk memproses data dalam skala besar. Cloud *Computing* menawarkan beberapa manfaat utama bagi pengembangan dan penerapan kecerdasan buatan, di antaranya:

1. Skalabilitas: Cloud *Computing* memungkinkan penyesuaian sumber daya komputasi sesuai kebutuhan. Ketika beban kerja meningkat, misalnya dalam pelatihan model pembelajaran mendalam dengan dataset yang besar, cloud menyediakan infrastruktur yang dapat di-skala secara dinamis.
2. Akses ke Data dalam Skala Besar: Kecerdasan buatan membutuhkan data untuk melatih dan mengembangkan model yang cerdas. Dengan cloud *Computing*, penyimpanan data tidak lagi menjadi hambatan karena cloud menawarkan penyimpanan yang praktis, aman, dan mudah diakses.
3. Kecepatan dan Efisiensi: Dengan cloud, organisasi dapat menjalankan proses komputasi yang membutuhkan waktu lama secara lebih cepat. Algoritma AI dapat berjalan lebih efisien karena

didukung oleh prosesor, GPU, atau TPU yang dioptimalkan untuk komputasi berat.

- Platform AI di Cloud *Computing*

  Penyedia layanan cloud besar, seperti Amazon Web Services (AWS), Google Cloud, dan Microsoft Azure, menawarkan berbagai platform dan alat khusus untuk pengembangan kecerdasan buatan. Platform ini memberikan kemampuan kepada pengguna untuk membangun, melatih, dan menyebarkan model AI tanpa perlu mengelola infrastruktur komputasi yang kompleks secara manual.

  1. Amazon SageMaker: Sebuah platform di AWS yang memungkinkan pengembang untuk membangun, melatih, dan menerapkan model pembelajaran mesin secara cepat. SageMaker menyediakan alat-alat untuk mempercepat pengembangan AI dan memungkinkan kolaborasi lebih mudah di antara tim-tim data science.
  2. Google AI Platform: Di bawah naungan Google Cloud, platform ini memberikan dukungan untuk pembelajaran mesin end-

to-end. Google AI Platform mendukung framework populer seperti TensorFlow, menjadikannya pilihan yang baik untuk membangun aplikasi cerdas.

3. Azure Machine Learning: Platform berbasis cloud dari Microsoft yang menyediakan layanan untuk membangun, melatih, dan mengimplementasikan model kecerdasan buatan secara mudah dengan integrasi yang baik ke dalam ekosistem produk Microsoft.

- Kecerdasan Komputasional dalam Lingkungan Cloud

  Salah satu kekuatan cloud *Computing* adalah kemampuannya untuk mendukung pengembangan dan penerapan kecerdasan komputasional dalam berbagai lingkungan. Contohnya, banyak aplikasi berbasis Internet of Things (IoT) yang memanfaatkan kecerdasan komputasional untuk memproses data sensor secara real-time, dan cloud *Computing* menyediakan infrastruktur yang dapat menangani volume data ini dengan cepat.

  Cloud *Computing* juga memungkinkan sistem kecerdasan komputasional untuk diterapkan

dalam skala besar, di mana data yang berasal dari berbagai sumber dapat diproses dan dianalisis secara terdistribusi. Hal ini memberikan keuntungan yang signifikan dalam aplikasi seperti pemantauan lingkungan, pengenalan wajah dalam sistem keamanan, dan pengelolaan lalu lintas berbasis AI.

- Biaya dan Efisiensi dalam Penggunaan Cloud

  Dengan menggunakan cloud, biaya untuk menjalankan aplikasi Kecerdasan Komputasional dapat diminimalkan. Perusahaan atau pengembang individu tidak perlu berinvestasi dalam infrastruktur fisik yang mahal karena cloud *Computing* menawarkan model pembayaran berdasarkan penggunaan (pay-as-you-go). Ini berarti bahwa pengguna hanya membayar sumber daya yang mereka gunakan, baik untuk penyimpanan data, pemrosesan komputasi, atau pelatihan model.

  Keunggulan ini menjadikan cloud *Computing* sebagai pilihan ideal bagi berbagai entitas, mulai dari perusahaan rintisan hingga perusahaan besar, yang ingin memanfaatkan kekuatan kecerdasan buatan tanpa harus

berinvestasi dalam perangkat keras secara besar-besaran.

- Tantangan dan Peluang Kecerdasan Komputasional di Cloud

  Meskipun cloud *Computing* menawarkan banyak manfaat bagi kecerdasan komputasional, ada juga tantangan yang perlu diatasi, seperti masalah keamanan data, privasi, dan ketergantungan pada penyedia layanan cloud. Namun, di sisi lain, integrasi kecerdasan buatan dan cloud membuka banyak peluang untuk inovasi di berbagai industri, seperti kesehatan, manufaktur, transportasi, dan lainnya.

BAB II
JENIS MASUKAN DATA

2.1. Data Teks

Data teks adalah salah satu jenis masukan data yang paling umum dan melibatkan informasi yang disajikan dalam bentuk kata, kalimat, atau paragraf. Data teks dapat berasal dari berbagai sumber, termasuk dokumen, artikel, email, postingan media sosial, dan lainnya. Dalam konteks kecerdasan komputasional, data teks seringkali memerlukan pemrosesan khusus untuk dapat digunakan dalam analisis dan aplikasi berbasis algoritma.

Jenis Data Teks

1. Data Teks Terstruktur

   Data teks terstruktur adalah data yang memiliki format atau skema yang konsisten, sehingga mudah diolah dan dianalisis. Biasanya, data ini disimpan dalam format seperti basis data atau tabel.

2. Data Teks Tidak Terstruktur

   Data teks tidak terstruktur tidak mengikuti format yang tetap dan sering kali terdiri dari teks bebas tanpa struktur yang jelas. Jenis data

ini memerlukan teknik pemrosesan teks yang lebih kompleks untuk dianalisis.

Aplikasi Data Teks dalam Kecerdasan Komputasional

1. Pencarian Informasi

   Menggunakan algoritma untuk mencari dan mengidentifikasi informasi relevan dari kumpulan data teks besar.
2. Analisis Ulasan dan Feedback

   Menganalisis ulasan pelanggan untuk mengidentifikasi tren dan pola yang dapat mempengaruhi keputusan bisnis.
3. Pengenalan Entitas Named Entity Recognition (NER)

   Mengidentifikasi dan mengklasifikasikan entitas seperti nama orang, lokasi, dan organisasi dalam teks.

2.2. Data Numerik

Data numerik adalah jenis data yang disajikan dalam bentuk angka dan dapat digunakan untuk perhitungan matematis dan statistik. Data ini seringkali menjadi dasar untuk analisis kuantitatif dalam berbagai aplikasi, termasuk kecerdasan komputasional,

penelitian ilmiah, dan analisis bisnis.

Jenis Data Numerik

1. Data Diskrit

   Data diskrit terdiri dari nilai-nilai yang terpisah atau terbatas, dan tidak dapat dibagi menjadi unit yang lebih kecil. Data ini biasanya diperoleh dari penghitungan dan memiliki jarak yang tetap antar nilai.

2. Data Kontinu

   Data kontinu adalah data yang dapat memiliki nilai dalam rentang yang kontinu dan dapat dibagi menjadi unit yang lebih kecil. Data ini diperoleh dari pengukuran dan memiliki nilai yang dapat memiliki banyak titik desimal.

Aplikasi Data Numerik dalam Kecerdasan Komputasional

1. Model Prediktif

   Membangun model yang dapat memprediksi hasil masa depan berdasarkan data numerik historis.

2. Analisis Tren

   Menganalisis pola dan tren dalam data numerik untuk membuat keputusan berbasis data.

3. Optimasi

Menggunakan data numerik untuk mencari solusi optimal dalam masalah kompleks.

2.3. Data Waktu

Data waktu adalah jenis data yang berkaitan dengan pengukuran waktu dan tanggal. Data ini sering digunakan dalam analisis untuk mengidentifikasi pola, tren, dan siklus yang terkait dengan waktu. Dalam kecerdasan komputasional, data waktu sangat penting untuk analisis time series, forecasting, dan perencanaan.

Jenis Data Waktu

1. Tanggal dan Waktu

   Data ini mencatat informasi spesifik tentang tanggal dan waktu tertentu. Format data ini sering kali termasuk hari, bulan, tahun, serta jam, menit, dan detik.

2. Durasi dan Interval

   Data ini mengukur rentang waktu atau interval antara dua titik waktu. Durasi sering diukur dalam satuan seperti detik, menit, jam, atau hari.

Aplikasi Data Waktu dalam Kecerdasan Komputasional

1. Analisis Tren dan Musiman

   Mengidentifikasi dan menganalisis pola musiman dalam data untuk merencanakan strategi bisnis atau merespons perubahan pasar.
2. Forecasting

   Memprediksi nilai masa depan berdasarkan data historis, yang sangat penting untuk perencanaan dan pengambilan keputusan.
3. Monitoring dan Alerting

   Mengawasi data waktu secara real-time untuk mendeteksi dan memberikan peringatan tentang peristiwa yang tidak biasa atau anomali.

2.4. Data Kategorikal

Data kategorikal adalah jenis data yang membagi informasi ke dalam kategori atau kelompok tertentu. Data ini tidak berbentuk angka melainkan representasi dari atribut atau karakteristik yang dapat dikelompokkan. Data kategorikal sering digunakan dalam analisis statistik dan pembelajaran mesin untuk mengelompokkan, membandingkan, dan memahami pola dalam data.

Jenis Data Kategorikal

1. Data Kategorikal Nominal

Data kategorikal nominal adalah data yang membagi informasi ke dalam kategori tanpa urutan atau hierarki tertentu. Setiap kategori adalah unik dan tidak memiliki hubungan yang lebih besar dengan kategori lain.

2. Data Kategorikal Ordinal

   Data kategorikal ordinal adalah data yang membagi informasi ke dalam kategori yang memiliki urutan atau peringkat tertentu. Meskipun kategorinya memiliki urutan, jarak antara kategori tidak selalu dapat diukur dengan tepat.

Aplikasi Data Kategorikal dalam Kecerdasan Komputasional

1. Klasifikasi

   Menggunakan data kategorikal sebagai fitur untuk membangun model klasifikasi yang dapat memprediksi kategori atau label.

2. Segmentasi Pasar

   Mengelompokkan pelanggan atau produk ke dalam segmen berdasarkan kategori untuk strategi pemasaran yang lebih baik.

3. Analisis Survey

Menganalisis hasil survey yang melibatkan data kategorikal untuk memahami pola responden dan membuat keputusan berbasis data.

2.5. Data Gambar

Data gambar adalah jenis data yang berisi informasi visual yang direpresentasikan dalam bentuk piksel. Data ini mencakup berbagai format gambar digital seperti JPEG, PNG, GIF, dan TIFF. Dalam konteks kecerdasan komputasional, data gambar sering digunakan dalam aplikasi seperti pengenalan objek, pemrosesan gambar, dan visi komputer.

Jenis Data Gambar

1. Gambar Berwarna

   Gambar yang menyimpan informasi warna dalam format RGB (Red, Green, Blue). Setiap piksel pada gambar memiliki nilai intensitas untuk setiap komponen warna.

2. Gambar Hitam Putih

   Gambar yang hanya memiliki dua nilai warna, hitam dan putih, atau berbagai tingkat abu-abu. Gambar ini biasanya lebih sederhana dibandingkan gambar berwarna.

3. Gambar Binari

Gambar yang hanya memiliki dua nilai piksel (misalnya, 0 dan 1), sering digunakan dalam aplikasi seperti pemrosesan dokumen dan segmentasi gambar.

Aplikasi Data Gambar dalam Kecerdasan Komputasional

1. Pengenalan Wajah
   Menganalisis dan mengenali wajah individu dalam gambar untuk aplikasi keamanan, otentikasi, atau analisis demografi.
2. Pengawasan dan Monitoring
   Menggunakan kamera dan analisis gambar untuk memantau area dan mendeteksi aktivitas yang mencurigakan atau kondisi darurat.
3. Diagnostik Medis
   Menganalisis gambar medis seperti CT scan atau MRI untuk diagnosis penyakit atau kelainan.

2.6. Data Suara

Data suara adalah jenis data yang berisi informasi audio yang direkam atau dihasilkan dalam bentuk sinyal suara. Data ini sering digunakan dalam aplikasi seperti pengenalan suara, pemrosesan audio, dan komunikasi. Dalam kecerdasan komputasional, data

suara dapat digunakan untuk analisis lebih lanjut dalam berbagai konteks, termasuk asisten virtual, analisis sentimen, dan identifikasi speaker.

Jenis Data Suara

1. Audio Berfrekuensi Rendah

   Sinyal audio yang memiliki frekuensi rendah, seperti suara manusia atau alat musik yang menghasilkan frekuensi bass.
2. Audio Berfrekuensi Tinggi

   Sinyal audio yang memiliki frekuensi tinggi, seperti suara cericit atau suara alat musik dengan frekuensi tinggi.
3. Audio Monofonik

   Audio yang direkam atau diputar melalui satu saluran, sering kali digunakan untuk komunikasi satu arah.
4. Audio Stereofonik

   Audio yang direkam atau diputar melalui dua saluran (kiri dan kanan), menciptakan efek kedalaman dan ruang.

Aplikasi Data Suara dalam Kecerdasan Komputasional

1. Asisten Virtual

Menggunakan data suara untuk memahami dan merespons perintah suara pengguna.

2. Pengenalan Ucapan

   Mengonversi ucapan menjadi teks untuk aplikasi seperti transkripsi otomatis atau kontrol suara.

3. Pendeteksian Anomali

   Menganalisis suara untuk mendeteksi anomali atau kejadian tertentu, seperti mendeteksi kerusakan mesin berdasarkan suara.

2.7. Data Video

Data video adalah data visual yang terdiri dari serangkaian gambar (frame) yang ditampilkan secara berurutan pada kecepatan tertentu (biasanya dalam satuan frame per second atau FPS), sehingga menciptakan ilusi gerakan. Dalam kecerdasan komputasional, data video digunakan dalam berbagai aplikasi seperti pengenalan wajah, pelacakan objek, analisis gerak, dan sistem pengawasan. Video merupakan data yang kompleks karena menggabungkan informasi visual (gambar) dengan elemen temporal (waktu).

Komponen Data Video

1. Frame

Frame adalah gambar individu dalam urutan video. Setiap frame adalah snapshot statis, dan ketika frame ini diputar berurutan dengan kecepatan tertentu, mereka menciptakan ilusi gerakan.

2. Audio

   Banyak video juga mengandung data audio yang sinkron dengan visual, menyediakan informasi suara yang mendukung atau menjelaskan konten visual.

3. Dimensi Spasial dan Temporal

   Dimensi spasial mengacu pada informasi visual dalam setiap frame (panjang, lebar, warna), sementara dimensi temporal mengacu pada urutan waktu di mana frame tersebut ditampilkan.

Aplikasi Data Video dalam Kecerdasan Komputasional

1. Pengawasan dan Keamanan

   Menggunakan data video untuk memantau area tertentu dan mendeteksi aktivitas yang mencurigakan atau anomali. Sistem ini sering dilengkapi dengan algoritma pendeteksian gerakan dan pelacakan objek.

2. Pengenalan Aktivitas

   Menganalisis gerakan dalam video untuk mengenali aktivitas tertentu seperti berjalan, berlari, atau melompat. Aplikasi ini banyak digunakan dalam bidang olahraga, keamanan, dan analisis perilaku.

3. Kendaraan Otonom

   Menggunakan video untuk memantau lingkungan sekitar kendaraan otonom, mengenali rambu lalu lintas, mendeteksi kendaraan lain, dan merencanakan rute.

2.8. Data Sensor

Data sensor adalah informasi yang dihasilkan oleh perangkat sensor, yang digunakan untuk mendeteksi dan mengukur kondisi fisik atau perubahan lingkungan. Sensor mengubah fenomena fisik, seperti suhu, cahaya, gerakan, tekanan, atau kelembaban, menjadi sinyal elektronik yang dapat dianalisis lebih lanjut. Data sensor banyak digunakan dalam berbagai aplikasi mulai dari Internet of Things (IoT), sistem pengawasan, peralatan medis, hingga sistem otomatisasi industri.

Jenis-Jenis Data Sensor

1. Data Sensor Suhu

Mengukur suhu lingkungan atau objek menggunakan sensor suhu seperti termokopel atau termistor.

2. Data Sensor Tekanan

   Mengukur tekanan gas atau cairan dalam suatu sistem. Sensor tekanan sering digunakan dalam aplikasi hidrolik dan pneumatik.

3. Data Sensor Cahaya

   Mengukur intensitas cahaya atau radiasi optik. Sensor cahaya digunakan dalam aplikasi seperti kontrol pencahayaan otomatis dan deteksi cahaya dalam fotografi.

4. Data Sensor Kelembaban

   Mengukur tingkat kelembaban dalam udara atau bahan lain. Sensor ini penting dalam pengendalian iklim dan pemantauan lingkungan.

5. Data Sensor Akselerasi

   Mengukur percepatan atau perubahan kecepatan suatu objek. Data ini penting dalam aplikasi pengukuran gerak, seperti smartphone dan perangkat pelacakan kebugaran.

6. Data Sensor Getaran

Mengukur getaran atau osilasi suatu objek. Sensor ini banyak digunakan dalam pemeliharaan mesin dan prediksi kerusakan.

7. Data Sensor Gerak

   Mengukur perubahan posisi atau gerakan suatu objek. Digunakan dalam sistem keamanan dan otomatisasi.

8. Data Sensor Gas

   Mengukur konsentrasi gas di udara, digunakan untuk pemantauan lingkungan atau deteksi kebocoran gas.

Aplikasi Data Sensor dalam Kecerdasan Komputasional

1. Internet of Things (IoT)

   Data sensor adalah inti dari IoT, di mana perangkat yang terhubung mengumpulkan, memproses, dan berbagi data sensor untuk berbagai aplikasi seperti rumah pintar, kota pintar, dan sistem industri otomatis.

2. Otomasi Industri

   Data sensor digunakan untuk memantau dan mengontrol proses produksi di pabrik. Sensor memberikan informasi real-time tentang status mesin, bahan, dan lingkungan produksi.

3. Sistem Pengawasan dan Keamanan

   Sensor digunakan untuk mendeteksi aktivitas atau perubahan dalam lingkungan untuk tujuan pengawasan dan keamanan.

BAB III
GAMBARAN PROSES TRANSFORMASI DATA

3.1. Pengantar Transformasi Data

Transformasi data adalah proses kritis dalam alur kerja data yang melibatkan konversi data dari satu format atau struktur ke format atau struktur lain. Proses ini tidak hanya mencakup perubahan format data tetapi juga mencakup pembersihan, pemrosesan, dan integrasi data dari berbagai sumber (Fitria and Rozci, 2023). Tujuan utama dari transformasi data adalah memastikan bahwa data yang akan digunakan untuk analisis dan pemodelan sudah bersih, relevan, dan siap untuk digunakan. Dalam dunia di mana data berasal dari berbagai sumber dengan format yang berbeda-beda, transformasi data menjadi langkah esensial untuk mengintegrasikan dan menormalkan data tersebut.

Proses transformasi data sering dimulai dengan pengumpulan data dari berbagai sumber. Sumber data ini bisa berasal dari database relasional, file teks, *API web*, sensor *IoT*, atau bahkan data yang dihasilkan oleh

pengguna. Pengumpulan data yang efektif memerlukan pemahaman mendalam tentang struktur dan format data di masing-masing sumber, serta cara mengakses dan mengekstrak data tersebut. Tantangan utama dalam tahap ini adalah memastikan integritas dan kualitas data yang dikumpulkan sehingga siap untuk diproses lebih lanjut.

Setelah data terkumpul, langkah selanjutnya adalah pembersihan data. Pembersihan data mencakup identifikasi dan perbaikan kesalahan dalam data, seperti nilai yang hilang, duplikasi, dan inkonsistensi. Proses ini melibatkan berbagai teknik seperti pengisian nilai yang hilang, penghapusan duplikasi, dan koreksi data yang salah. Pembersihan data adalah langkah yang sangat penting karena data yang kotor atau tidak akurat dapat mempengaruhi hasil analisis dan pemodelan secara signifikan. Perhatian yang cermat dan teliti terhadap *detail* selama tahap ini sangat diperlukan untuk memastikan kualitas data.

Normalisasi data adalah tahap berikutnya dalam proses transformasi data. Normalisasi melibatkan penyesuaian skala dan distribusi data agar sesuai dengan kebutuhan analisis atau model yang akan digunakan (Febtiawan *et al.*, 2024). Teknik normalisasi umum termasuk scaling, standardisasi, dan

transformasi logaritmik atau eksponensial. Normalisasi membantu dalam mengurangi bias yang disebabkan oleh perbedaan skala dalam data dan memastikan bahwa semua fitur data memiliki kontribusi yang seimbang dalam analisis. Dengan normalisasi yang tepat, model komputasional dapat lebih akurat dan efisien dalam memproses data.

Integrasi data merupakan langkah terakhir dalam transformasi data, di mana data dari berbagai sumber digabungkan untuk membentuk satu set data yang terpadu dan konsisten. Integrasi data melibatkan pencocokan dan penggabungan data berdasarkan kunci atau atribut yang sama, serta penanganan konflik data yang mungkin muncul (Hoffman, 2017). Proses ini memungkinkan analisis yang lebih komprehensif dan mendalam dengan memanfaatkan informasi dari berbagai sumber. Dengan integrasi data yang baik, organisasi dapat menggabungkan data internal dan eksternal untuk mendapatkan wawasan yang lebih kaya dan informatif.

Secara keseluruhan, transformasi data adalah fondasi yang penting dalam setiap proyek analisis data atau pemodelan komputasional. Tanpa transformasi data yang tepat, kualitas dan kegunaan data dapat terdegradasi, yang pada akhirnya dapat mempengaruhi

keputusan yang diambil berdasarkan analisis data tersebut. Pemahaman mendalam tentang teknik dan proses transformasi data adalah keterampilan yang sangat berharga bagi para profesional data dan peneliti di bidang kecerdasan komputasional.

Gambar 3.1 *SQL Database* Transformasi Data
Sumber: https://bss.mediabpr.com/2018/09/sql-database-transformasi-data.html

3.2. Gambaran Proses Transformasi Data

1. Pengumpulan Data

   Pengumpulan data merupakan tahap pertama dalam proses transformasi data, di mana data dikumpulkan dari berbagai sumber baik internal maupun eksternal (Aslamiyah, 2022). Sumber data bisa sangat beragam, termasuk database relasional, *file* teks, sensor *Internet of*

Things (IoT), *data web*, dan *API*. Setiap sumber data memiliki format dan struktur yang berbeda, sehingga proses pengumpulan data memerlukan teknik yang tepat untuk mengekstraksi dan menggabungkan data tersebut dengan benar. Misalnya, data dari database dapat diekstraksi menggunakan *query SQL*, sementara data dari *API web* mungkin memerlukan penggunaan permintaan *HTTP* dan pemrosesan *JSON* atau *XML*. Tantangan utama dalam tahap ini adalah memastikan bahwa data yang dikumpulkan memiliki integritas dan kualitas yang memadai untuk analisis lebih lanjut.

Pengumpulan data melibatkan validasi data untuk memastikan bahwa data yang diambil akurat dan sesuai dengan kebutuhan analisis. Validasi ini bisa mencakup pemeriksaan terhadap format data, konsistensi data, dan kesesuaian data dengan skema yang diharapkan. Proses pengumpulan data yang efektif akan menghasilkan data yang siap untuk tahap transformasi berikutnya, mengurangi risiko kesalahan dan kekurangan data yang dapat menghambat proses analisis.

2. Pembersihan Data

   Pembersihan data adalah tahap kritis berikutnya dalam proses transformasi data, di mana data yang telah dikumpulkan dibersihkan dari berbagai kesalahan dan ketidakkonsistenan (Adu-ManuSarpong and Kingsley Arthur, 2013). Nilai yang hilang, duplikasi, dan kesalahan data adalah beberapa masalah umum yang sering ditemukan dalam data mentah. Proses pembersihan data melibatkan teknik-teknik seperti pengisian nilai yang hilang menggunakan imputasi, penghapusan atau penggabungan duplikasi data, serta koreksi kesalahan data berdasarkan aturan bisnis atau referensi eksternal. Pembersihan data bertujuan untuk meningkatkan kualitas data sehingga dapat digunakan dengan lebih akurat dan andal dalam analisis dan pemodelan.

   Tahap ini juga mencakup identifikasi dan penghapusan *data outlier* atau data yang tidak konsisten yang dapat mempengaruhi hasil analisis. Dengan data yang bersih, risiko bias dan kesalahan dalam analisis dapat diminimalkan. Pembersihan data membantu memastikan bahwa data sesuai dengan format

dan standar yang diinginkan, memudahkan proses analisis dan pemodelan berikutnya. Sebuah *dataset* yang bersih dan konsisten adalah fondasi dari setiap analisis data yang sukses, memberikan kepercayaan yang lebih besar terhadap hasil yang dihasilkan.

3. Normalisasi Data

   Normalisasi data adalah proses mengubah skala data ke rentang yang konsisten untuk memastikan distribusi yang seragam dan memudahkan proses analisis (Mulyati *et al.*, 2018). Dalam banyak kasus, data yang berasal dari berbagai sumber dapat memiliki skala yang berbeda, sehingga perlu dinormalisasi agar analisis dapat dilakukan secara lebih akurat. Teknik normalisasi yang umum digunakan termasuk *scaling*, standardisasi, dan transformasi logaritmik atau eksponensial. Misalnya, dalam proses *scaling*, data diubah ke dalam rentang tertentu, seperti 0 hingga 1, untuk menghilangkan perbedaan skala antar variabel.

   Normalisasi data sangat penting dalam analisis dan pemodelan karena membantu dalam mengurangi bias yang disebabkan oleh

perbedaan skala antar fitur data. Dengan normalisasi yang tepat, semua fitur data dapat berkontribusi secara seimbang dalam analisis, meningkatkan akurasi dan efisiensi model komputasional. Selain itu, normalisasi juga membantu dalam mempercepat konvergensi algoritma pembelajaran mesin dan mengurangi risiko *overfitting*. Dengan memastikan bahwa data dinormalisasi dengan baik, kita dapat menghasilkan model yang lebih andal dan dapat diandalkan.

4. Integrasi Data

   Integrasi data adalah tahap terakhir dalam proses transformasi data, di mana data dari berbagai sumber digabungkan untuk membentuk satu set data yang terpadu dan konsisten (Widarti, Bahri and Widyanto, 2018). Proses ini melibatkan pencocokan dan penggabungan data berdasarkan kunci atau atribut yang sama, serta penanganan konflik data yang mungkin muncul. Integrasi data memungkinkan analisis yang lebih komprehensif dan mendalam dengan memanfaatkan informasi dari berbagai sumber. Misalnya, data penjualan dari berbagai cabang

toko dapat diintegrasikan untuk memberikan gambaran menyeluruh tentang kinerja bisnis secara keseluruhan.

Tantangan utama integrasi data adalah mengatasi perbedaan dalam format, struktur, dan semantik data dari berbagai sumber. Proses memerlukan teknik cermat untuk memastikan bahwa data yang digabungkan konsisten dan tidak ada redundansi atau konflik. Integrasi data yang baik membuat organisasi menggabungkan data internal dan eksternal untuk mendapatkan wawasan yang lebih kaya dan informatif. Integrasi data yang efektif memungkinkan pengambilan keputusan baik dan strategis, meningkatkan efisiensi operasional, dan memberikan keunggulan kompetitif bagi organisasi.

3.3. Teknik-Teknik Transformasi Data

1. Transformasi Linier dan Non-Linier

   Transformasi linier dan non-linier adalah teknik yang digunakan untuk mengubah skala data guna memastikan data memiliki distribusi yang seragam dan sesuai untuk analisis. Transformasi linier mencakup metode seperti normalisasi dan

standardisasi (Sinulingga, Fatichah and Yuniarti, 2016). Normalisasi mengubah data ke dalam rentang tertentu, seperti 0 hingga 1, sehingga semua nilai data berada dalam skala yang konsisten. Standardisasi, di sisi lain, mengubah data sehingga memiliki mean nol dan deviasi standar satu, yang bermanfaat untuk algoritma pembelajaran mesin yang sensitif terhadap skala data. Transformasi ini membantu mengurangi bias yang disebabkan oleh perbedaan skala antar fitur data.

Transformasi non-linier, seperti logaritmik dan eksponensial, digunakan ketika data memiliki distribusi yang tidak normal atau terdapat perbedaan yang signifikan antar nilai data. Transformasi logaritmik, misalnya, dapat digunakan untuk mengatasi skewness atau distribusi yang sangat miring, dengan mengompres rentang data yang besar ke dalam skala yang lebih sempit. Transformasi eksponensial, sebaliknya, dapat memperbesar perbedaan antar nilai data untuk mengungkap pola yang tidak terlihat pada skala asli. Penggunaan transformasi non-linier ini membantu dalam membuat data lebih sesuai

untuk analisis statistik dan pemodelan komputasional.

2. Pengkodean Kategorikal

   Pengkodean kategorikal adalah proses mengubah data kategorikal menjadi format numerik agar dapat digunakan dalam algoritma pembelajaran mesin (Maghfiroh, Findawati and Indahyanti, 2023). Data kategorikal adalah data yang memiliki nilai dalam bentuk kategori atau label, seperti warna, jenis kelamin, atau kategori produk. Teknik pengkodean yang umum digunakan termasuk *one-hot encoding* dan *label encoding*. One-hot encoding mengubah setiap kategori menjadi vektor biner terpisah, di mana setiap vektor mewakili satu kategori dengan nilai 1 pada posisinya dan 0 pada posisi lainnya. Teknik ini sangat efektif untuk data yang memiliki sejumlah kecil kategori.

   Label encoding, di sisi lain, mengubah setiap kategori menjadi nilai numerik yang unik. Misalnya, kategori "merah", "biru", dan "hijau" dapat dikodekan sebagai 1, 2, dan 3. Meskipun teknik ini lebih sederhana dan efisien dalam hal penggunaan memori, perlu hati-hati dalam penggunaannya karena dapat memberikan

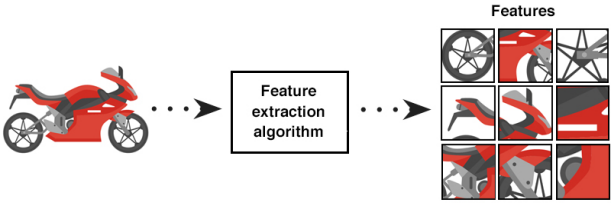
bobot numerik yang tidak diinginkan pada kategori tertentu, yang dapat mempengaruhi model pembelajaran mesin. Pemilihan teknik pengkodean yang tepat tergantung pada karakteristik data dan algoritma yang digunakan, serta perlu mempertimbangkan potensi bias yang mungkin timbul dari pengkodean tersebut.

3. Ekstraksi Fitur

   Ekstraksi fitur adalah proses mengidentifikasi dan mengekstrak fitur penting dari data mentah untuk meningkatkan efisiensi dan akurasi model komputasional (Hidayat *et al.*, 2021). Fitur adalah representasi atau atribut dari data yang dapat digunakan oleh algoritma pembelajaran mesin untuk memahami dan memprediksi pola. Proses ekstraksi fitur dapat melibatkan teknik-teknik seperti pemilihan fitur, pembangkitan fitur, dan pengurangan dimensi. Pemilihan fitur mencakup pemilihan subset fitur yang paling relevan dan informatif dari data asli, yang dapat mengurangi kebisingan dan kompleksitas model.

   Pembangkitan fitur melibatkan pembuatan fitur baru dari data mentah melalui kombinasi atau

transformasi fitur yang ada. Misalnya, dalam analisis teks, fitur seperti frekuensi kata atau bigram dapat dihasilkan dari data teks mentah untuk menangkap informasi penting. Pengurangan dimensi, seperti Principal Component Analysis (PCA), digunakan untuk mengurangi jumlah fitur dengan mempertahankan informasi yang paling penting. Teknik-teknik ekstraksi fitur ini membantu dalam meningkatkan kinerja model dengan fokus pada fitur-fitur yang paling relevan dan mengurangi kompleksitas data yang harus diproses oleh algoritma pembelajaran mesin.



Gambar 3.2 Ekstraksi Fitur dan Ekstraksi Ciri Citra

Sumber: https://pemrogramanmatlab.com/2024/03/05/metode-ekstraksi-fitur-dalam-pengolahan-citra/

3.4. Alat dan Platform untuk Transformasi Data

1. Apache Spark

   Apache Spark adalah platform komputasi klaster yang cepat dan serbaguna, yang banyak digunakan untuk pemrosesan data skala besar. Spark menawarkan suatu kerangka kerja terdistribusi yang memungkinkan pemrosesan data dalam jumlah besar secara paralel, sehingga meningkatkan kecepatan dan efisiensi transformasi data (Aminudin and Cahyono, 2019). Spark memiliki komponen-komponen seperti Spark SQL untuk pengolahan data terstruktur, Spark Streaming untuk pemrosesan data real-time, dan MLlib untuk pembelajaran mesin. Salah satu keunggulan utama Spark adalah kemampuannya untuk menangani data dalam berbagai format dan sumber, baik itu dari HDFS, Cassandra, HBase, atau S3. Dengan fitur-fitur ini, Apache Spark menjadi alat yang sangat powerful untuk pengumpulan, pembersihan, normalisasi, dan integrasi data dalam skala besar.

   Spark menyediakan API yang mendukung berbagai bahasa pemrograman seperti Python, Java, Scala, dan R, yang memudahkan

pengembang untuk mengintegrasikan Spark ke dalam alur kerja mereka. Kemampuan Spark untuk melakukan pemrosesan dalam memori (in-memory processing) juga meningkatkan kecepatan transformasi data secara signifikan dibandingkan dengan sistem berbasis disk. Dengan fitur-fitur canggih dan performa tinggi, Apache Spark menjadi pilihan utama bagi banyak perusahaan dalam mengelola dan memproses data besar secara efisien.

2. Pandas dalam Python

   Pandas adalah pustaka data manipulasi yang sangat populer dalam ekosistem Python, yang menyediakan struktur data dan alat analisis data yang kuat dan mudah digunakan (Hermanto *et al.*, 2023). Pandas memungkinkan pengguna untuk membaca data dari berbagai sumber seperti CSV, Excel, SQL databases, dan JSON, serta melakukan berbagai operasi transformasi data dengan mudah. Struktur data utama dalam Pandas, DataFrame, memungkinkan pengguna untuk melakukan operasi seperti filtering, grouping, pivoting, dan merging dengan sintaks yang intuitif dan efisien. Pandas sangat berguna untuk pembersihan

data, normalisasi, dan eksplorasi data sebelum melakukan analisis atau pemodelan lebih lanjut. Pandas juga mendukung integrasi dengan pustaka lain dalam ekosistem Python seperti NumPy, Matplotlib, dan scikit-learn, yang memungkinkan alur kerja data yang terintegrasi dari pembersihan hingga visualisasi dan pemodelan. Kemudahan penggunaan dan fleksibilitas Pandas menjadikannya alat yang sangat berguna bagi data scientist dan analis data dalam menangani berbagai tugas transformasi data. Dengan dokumentasi yang komprehensif dan komunitas pengguna yang aktif, Pandas terus berkembang dan menjadi salah satu alat terdepan dalam pengolahan data di Python.

3. Alat ETL: Talend

   Talend adalah salah satu alat ETL (*Extract*, *Transform*, *Load*) terkemuka yang menyediakan solusi untuk integrasi data, transformasi, dan manajemen data. Talend menawarkan antarmuka grafis yang intuitif untuk merancang dan mengelola alur kerja ETL, memungkinkan pengguna untuk menghubungkan berbagai sumber data, membersihkan data, mengubah

skala data, dan memuat data ke berbagai target seperti data *warehouses*, databases, dan aplikasi *cloud* (Andriansyah, 2022). *Talend* mendukung berbagai konektor untuk sumber data yang berbeda, termasuk database relasional, file flat, *API web*, dan aplikasi *SaaS*, yang membuatnya fleksibel dan dapat diandalkan dalam berbagai skenario integrasi data.

Keunggulan Talend terletak pada kemampuan otomatisasi dan orkestrasi proses ETL, yang memungkinkan pengguna untuk menjadwalkan dan memantau alur kerja secara efektif. Selain itu, *Talend* menyediakan fitur-fitur canggih seperti *data quality tools*, *master data management*, dan *big data integration*, yang membantu organisasi untuk memastikan integritas dan konsistensi data mereka. Dengan solusi *end-to-end* untuk manajemen data, Talend menjadi pilihan populer bagi perusahaan yang membutuhkan alat ETL yang kuat dan dapat diskalakan.

4. Alat ETL: Informatica

   Informatica adalah alat ETL lainnya yang terkenal dengan kemampuan integrasi datanya yang kuat dan fitur-fitur yang komprehensif

untuk transformasi data. *Informatica PowerCenter* adalah produk utama yang digunakan oleh banyak perusahaan besar untuk mengelola alur kerja ETL. *Informatica* menyediakan lingkungan pengembangan visual untuk merancang, mengimplementasikan, dan mengelola proses ETL, dengan dukungan untuk berbagai sumber data dan target, termasuk *databases*, data *warehouses*, dan aplikasi *cloud*. Informatica juga menawarkan fitur-fitur canggih seperti *data profiling*, *data masking*, dan *data quality*, yang membantu memastikan bahwa data yang diproses memiliki kualitas tinggi dan sesuai dengan standar keamanan (Gupta, 2019).

Salah satu keunggulan utama Informatica adalah skalabilitas dan performanya dalam menangani volume data yang besar dan kompleksitas tinggi. Informatica mendukung arsitektur terdistribusi dan paralel, yang memungkinkan pemrosesan data yang efisien dalam lingkungan *big data*. Dengan dukungan untuk integrasi data *real-time* dan *batch*, serta kemampuan orkestrasi dan manajemen alur kerja yang canggih, *Informatica* menjadi alat

yang sangat berharga bagi organisasi yang membutuhkan solusi integrasi data yang handal dan dapat diskalakan..

3.5. Kesimpulan

Proses transformasi data adalah langkah krusial dalam siklus kehidupan data yang memastikan data siap untuk analisis dan pemodelan. Proses ini terdiri dari beberapa tahapan utama: pengumpulan data, pembersihan data, normalisasi data, dan integrasi data. Setiap tahapan memiliki tujuan spesifik dan tantangan tersendiri yang harus diatasi untuk menghasilkan data yang berkualitas tinggi. Pengumpulan data melibatkan pengambilan data dari berbagai sumber dengan memastikan integritas dan kualitas data tersebut. Pembersihan data bertujuan untuk menghilangkan kesalahan, duplikasi, dan nilai yang hilang, sehingga data yang dihasilkan lebih akurat dan dapat diandalkan.

Normalisasi data membantu dalam mengubah skala data ke dalam rentang yang konsisten, yang sangat penting untuk analisis dan pemodelan yang lebih akurat. Teknik normalisasi seperti scaling dan standardisasi memastikan bahwa data memiliki distribusi yang seragam, mengurangi bias yang disebabkan oleh perbedaan skala antar fitur.

Selanjutnya, integrasi data menggabungkan data dari berbagai sumber untuk membentuk satu set data yang terpadu dan konsisten. Proses ini memungkinkan analisis yang lebih komprehensif dengan memanfaatkan informasi dari berbagai sumber dan memastikan bahwa data yang dihasilkan lebih kaya dan informatif.

Berbagai teknik transformasi data, termasuk transformasi linier dan non-linier, pengkodean kategorikal, dan ekstraksi fitur, digunakan untuk mengubah dan meningkatkan data agar lebih siap digunakan dalam analisis dan pemodelan. Transformasi linier dan non-linier mengubah skala data untuk memastikan distribusi yang seragam, sementara pengkodean kategorikal mengubah data kategorikal menjadi format numerik yang lebih mudah diproses oleh algoritma pembelajaran mesin. Ekstraksi fitur mengidentifikasi dan mengekstraksi fitur penting dari data mentah untuk meningkatkan efisiensi dan akurasi model.

Alat dan platform seperti *Apache Spark*, *Pandas* dalam *Python*, *Talend*, dan *Informatica* memainkan peran penting dalam memfasilitasi proses transformasi data. Alat-alat ini menyediakan fungsi yang memudahkan proses pengumpulan, pembersihan,

normalisasi, dan integrasi data, serta mendukung berbagai bahasa pemrograman dan sumber data. Dengan memanfaatkan alat dan teknik transformasi data yang tepat, organisasi dapat memastikan bahwa data yang mereka gunakan untuk analisis dan pemodelan adalah bersih, relevan, dan siap digunakan, yang pada akhirnya membantu dalam pengambilan keputusan yang lebih baik dan strategis.

BAB IV
VEKTOR FITUR

4.1. Konsep Dasar Vektor Fitur

Vektor Fitur adalah representasi numerik dari data yang digunakan dalam berbagai algoritma kecerdasan komputasional dan machine learning. Vektor fitur adalah salah satu elemen terpenting dalam proses pembelajaran mesin karena mereka mewakili karakteristik atau atribut dari data dalam bentuk yang dapat dipahami oleh komputer.

1. Pengertian dan Fungsi

   Vektor fitur adalah kumpulan nilai-nilai (fitur) yang mewakili properti dari sebuah objek, peristiwa, atau fenomena dalam bentuk numerik. Setiap fitur biasanya berhubungan dengan satu aspek tertentu dari data, seperti warna dalam pengolahan citra, atau frekuensi kata dalam pengolahan teks. Vektor fitur ini kemudian digunakan oleh algoritma machine learning untuk mempelajari pola dan hubungan antar data. Contoh: Dalam pemrosesan teks, vektor fitur dapat terdiri dari frekuensi kata, jumlah karakter, atau panjang kalimat.

2. Representasi Data dalam Bentuk Vektor

   Data yang digunakan dalam kecerdasan komputasional dapat sangat bervariasi, baik dalam bentuk teks, gambar, atau suara. Vektor fitur berfungsi untuk mengubah data ini ke dalam bentuk yang seragam, yaitu array atau matriks angka. Ini memungkinkan algoritma untuk memahami dan memproses data secara konsisten.

   Setiap dimensi dalam vektor fitur mewakili satu karakteristik atau atribut spesifik. Misalnya, dalam pengenalan objek menggunakan pengolahan citra, setiap fitur mungkin mewakili tingkat intensitas cahaya pada pixel tertentu.

3. Hubungan dengan Data Multidimensi

   Vektor fitur sering kali merepresentasikan data multidimensi, di mana setiap dimensi merepresentasikan satu karakteristik atau atribut dari data. Dalam konteks yang lebih sederhana, ini seperti merepresentasikan individu dengan sejumlah karakteristik yang berbeda seperti tinggi badan, berat badan, usia, dll. Jumlah dimensi dalam vektor fitur sama dengan jumlah karakteristik yang digunakan untuk menggambarkan objek tersebut.

Dimensi ini dapat berupa data numerik (seperti angka nyata), data kategorikal (seperti jenis kelamin atau tipe kendaraan), atau bahkan hasil dari transformasi data (seperti hasil dari ekstraksi fitur). Semakin tinggi dimensi vektor fitur, semakin kompleks data yang direpresentasikan, namun terlalu banyak dimensi juga dapat menyebabkan masalah seperti overfitting dan meningkatnya waktu komputasi.

4.2. Ekstraksi Fitur

Ekstraksi fitur adalah proses yang digunakan untuk mengidentifikasi dan mengekstrak informasi penting dari data mentah, yang kemudian direpresentasikan dalam bentuk vektor fitur. Tujuan utama dari ekstraksi fitur adalah untuk mengubah data yang kompleks menjadi serangkaian fitur atau atribut yang lebih sederhana namun tetap mencakup informasi yang relevan untuk pembelajaran mesin atau kecerdasan komputasional. Fitur yang diekstraksi dapat berupa pola, hubungan, atau karakteristik tertentu yang akan digunakan oleh model untuk membuat prediksi atau keputusan.

1. Tujuan dan Pentingnya Ekstraksi Fitur

Ekstraksi fitur memainkan peran krusial dalam memastikan kinerja model yang baik. Proses ini membantu untuk:

- Mengurangi dimensionalitas data: Data mentah sering kali memiliki dimensi yang sangat tinggi, seperti dalam pengolahan citra atau teks. Ekstraksi fitur membantu menyederhanakan data menjadi dimensi yang lebih rendah tanpa kehilangan informasi penting.
- Meningkatkan relevansi informasi: Hanya fitur yang relevan dengan tugas pembelajaran yang dipertahankan, sementara fitur yang kurang relevan atau berisik diabaikan.
- Memfasilitasi pemrosesan lebih efisien: Data yang disederhanakan melalui ekstraksi fitur dapat diproses lebih cepat oleh algoritma pembelajaran mesin, mengurangi beban komputasi.
- Mencegah overfitting: Dengan mengurangi jumlah fitur yang digunakan, ekstraksi fitur juga membantu menghindari overfitting, di mana model menjadi terlalu spesifik

terhadap data pelatihan dan gagal untuk menggeneralisasi pada data baru.

2. Proses Ekstraksi Fitur

   Ekstraksi fitur dilakukan dengan memilih dan mengidentifikasi fitur-fitur penting dari data mentah. Proses ini melibatkan beberapa langkah, termasuk:

   - Preprocessing data: Meliputi pembersihan data, normalisasi, atau transformasi awal untuk memastikan kualitas data yang akan diekstraksi.
   - Penerapan teknik ekstraksi fitur: Menggunakan metode statistik atau teknik lain untuk menemukan pola dan atribut penting.
   - Validasi fitur: Mengukur efektivitas fitur yang diekstraksi untuk memastikan bahwa mereka relevan dengan tugas pembelajaran yang diinginkan.

3. Metode Ekstraksi Fitur

   Berbagai teknik dan metode dapat digunakan untuk mengekstraksi fitur tergantung pada jenis data dan masalah yang dihadapi:

   a. Principal Component Analysis (PCA)

- PCA adalah metode ekstraksi fitur yang digunakan untuk mereduksi dimensi data sambil mempertahankan variansi maksimum yang terkandung di dalamnya. PCA melakukan transformasi linear untuk memproyeksikan data ke dalam ruang yang lebih kecil, di mana setiap sumbu baru (komponen utama) menggambarkan variansi terbesar dalam data.

b. Linear Discriminant Analysis (LDA)
   - LDA adalah teknik ekstraksi fitur yang mirip dengan PCA, tetapi bertujuan untuk memaksimalkan separasi antar kelas dalam data yang diberi label. Ini sangat efektif dalam tugas klasifikasi, karena LDA menekankan fitur yang membantu memisahkan kelas yang berbeda.

c. *Independent Component Analysis* (ICA)
   - ICA adalah teknik untuk memisahkan sinyal independen dari data yang kompleks. ICA sering

digunakan dalam pemrosesan sinyal, seperti dalam pengenalan suara atau analisis gambar.

d. Wavelet Transform
    - Metode ini digunakan untuk mengekstraksi fitur dari data yang mengandung pola frekuensi yang berubah-ubah, seperti sinyal atau citra. Ini sering digunakan dalam aplikasi pengolahan citra dan suara.

e. Autoencoders (Neural Networks)
    - Autoencoders adalah jenis jaringan saraf tiruan yang digunakan untuk mereduksi dimensi dan mengekstraksi fitur dari data. Dalam arsitektur autoencoder, data dimampatkan ke dalam lapisan tersembunyi, dan fitur penting diekstraksi selama proses pembelajaran tanpa pengawasan (unsupervised learning).

f. Histogram of Oriented Gradients (HOG)
    - Teknik ini digunakan dalam pengolahan citra untuk mengekstraksi fitur berbasis tepi

dan bentuk objek. HOG adalah fitur yang sering digunakan dalam deteksi objek, terutama dalam pengenalan wajah dan kendaraan.

g. Term Frequency-Inverse Document Frequency (TF-IDF)
    - Dalam pemrosesan teks, TF-IDF adalah metode populer untuk mengekstraksi fitur dari teks. Ini mengukur pentingnya suatu kata dalam dokumen relatif terhadap koleksi dokumen secara keseluruhan.

4. Ekstraksi Fitur dalam Berbagai Domain
    - Pengolahan Citra: Fitur yang diekstraksi bisa berupa tepi, tekstur, warna, atau bentuk objek. Teknik-teknik seperti HOG, PCA, dan Wavelet Transform sering digunakan.
    - Natural Language Processing (NLP): Dalam pengolahan teks, fitur seperti frekuensi kata, distribusi kata, atau representasi vektor dari kata-kata (word embeddings) diekstraksi untuk tugas seperti klasifikasi teks atau analisis sentimen.

- Pengolahan Sinyal: Ekstraksi fitur dilakukan dengan mengekstrak karakteristik dari sinyal seperti frekuensi, amplitudo, atau pola temporal.
- Audio Recognition: Fitur suara seperti Mel-Frequency Cepstral Coefficients (MFCC) sering digunakan dalam pengenalan suara dan pengolahan audio.

5. Tantangan dalam Ekstraksi Fitur
    - Curse of Dimensionality: Meskipun ekstraksi fitur dapat membantu mengurangi dimensi data, terlalu banyak fitur atau dimensi yang tidak relevan masih bisa mengarah pada masalah komputasi yang berat dan menurunnya kinerja model.
    - Overfitting: Jika fitur yang diekstraksi tidak cukup representatif atau terlalu detail, model dapat belajar untuk mengenali pola hanya pada data pelatihan, dan tidak generalisasi dengan baik pada data baru.
    - Pemilihan Fitur yang Tepat: Memilih fitur yang paling relevan sangat penting untuk menghindari redundansi atau kebisingan dalam data.

4.3. Representasi Vektor Fitur

Representasi vektor fitur adalah cara mengekspresikan data dalam bentuk vektor numerik yang digunakan dalam berbagai algoritma pembelajaran mesin dan kecerdasan komputasional. Setiap dimensi dalam vektor fitur merepresentasikan suatu atribut atau karakteristik data, yang memungkinkan algoritma untuk melakukan proses analisis dan pengambilan keputusan.

1. Definisi dan Peran Vektor Fitur

   Vektor fitur adalah kumpulan angka yang mengandung informasi penting dari data mentah yang diolah, seperti gambar, teks, atau suara. Representasi ini sangat penting karena memungkinkan mesin atau algoritma untuk "membaca" data dalam format yang bisa diproses. Tanpa representasi vektor fitur yang baik, algoritma machine learning tidak akan mampu menangkap pola yang relevan dalam data.

   Misalnya, dalam pengolahan teks, setiap kata dalam dokumen dapat direpresentasikan oleh vektor angka yang menunjukkan frekuensi atau hubungan semantik dengan kata-kata lain. Di sisi lain, dalam pengolahan citra, setiap gambar

dapat direpresentasikan oleh vektor fitur yang menyimpan informasi tentang warna, tekstur, dan bentuk dari gambar tersebut.

2. Fitur Numerik, Kategorikal, dan Kombinasi

   Terdapat beberapa jenis data yang dapat direpresentasikan dalam vektor fitur:

   - Fitur numerik: Fitur ini berupa angka dan biasanya digunakan untuk merepresentasikan informasi yang berkaitan dengan nilai kontinu, seperti suhu, berat, atau jumlah. Misalnya, dalam data harga rumah, fitur seperti luas rumah atau jumlah kamar bisa direpresentasikan sebagai fitur numerik.
   - Fitur kategorikal: Fitur ini berupa kategori atau label yang tidak memiliki urutan atau skala, seperti jenis kelamin, warna, atau tipe kendaraan. Fitur kategorikal sering direpresentasikan melalui one-hot encoding, di mana setiap kategori diubah menjadi vektor biner yang menunjukkan kehadiran atau ketidakhadiran kategori tertentu.
   - Fitur gabungan: Seringkali, dataset berisi kombinasi fitur numerik dan kategorikal. Dalam kasus ini, kedua jenis fitur ini diubah

menjadi vektor dan digabungkan menjadi representasi fitur lengkap yang dapat digunakan oleh algoritma pembelajaran mesin.

3. Normalisasi dan Standardisasi Vektor Fitur

   Untuk memastikan bahwa vektor fitur diolah secara efektif, penting untuk melakukan normalisasi atau standardisasi fitur. Kedua teknik ini membantu menyamakan skala fitur yang berbeda dan mencegah satu fitur mendominasi yang lain.

   - Normalisasi: Proses ini mengubah semua fitur ke dalam rentang tertentu, biasanya antara 0 dan 1. Ini dilakukan dengan mengurangi nilai minimum dari fitur dan kemudian membagi dengan selisih antara nilai maksimum dan minimum. Normalisasi bermanfaat saat semua fitur memiliki satuan yang berbeda dan perlu distandarkan ke skala yang sama.

   Formula normalisasi:

$$x_{new} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

   - Standardisasi: Standardisasi mengubah data sehingga memiliki distribusi normal dengan mean nol dan standar deviasi satu. Ini

bermanfaat saat model machine learning mengandalkan distribusi data yang seragam.

Formula standardisasi:

$$z = \frac{x - \mu}{\sigma}$$

4. Tantangan dalam Representasi Vektor Fitur
   - Curse of Dimensionality: Ketika jumlah fitur terlalu banyak, model bisa menjadi terlalu kompleks dan mengalami overfitting. Terlalu banyak dimensi juga bisa memperlambat proses pelatihan dan mengurangi akurasi prediksi.
   - Redundansi dan Korelasi Antar Fitur: Fitur-fitur yang sangat berkorelasi bisa menambah redundansi, yang berarti informasi yang sama diwakili beberapa kali. Ini dapat menurunkan efisiensi model. Oleh karena itu, teknik seperti PCA (Principal Component Analysis) sering digunakan untuk mereduksi dimensi tanpa kehilangan informasi penting.
   - Kualitas Fitur: Fitur yang diekstraksi dari data harus cukup representatif untuk memungkinkan model menemukan pola yang tepat. Fitur yang tidak relevan atau

tidak bermakna akan menghasilkan model yang tidak akurat.

5. Penerapan dan Studi Kasus
    - Dalam sistem rekomendasi, representasi vektor fitur pengguna dan item digunakan untuk menyarankan produk yang paling sesuai. Setiap pengguna diwakili oleh vektor fitur yang berisi preferensi mereka, dan item seperti film atau produk direpresentasikan dengan fitur seperti genre atau kategori.
    - Dalam pengolahan citra medis, gambar MRI atau CT scan direpresentasikan dalam vektor fitur yang mencakup fitur morfologis atau intensitas piksel, yang kemudian digunakan untuk mendeteksi penyakit atau anomali.

4.4. Implementasi Vektor Fitur dalam Kecerdasan Komputasional

Implementasi vektor fitur dalam kecerdasan komputasional adalah penerapan konsep vektor fitur dalam berbagai algoritma dan model untuk analisis, pengolahan, dan pemodelan data. Vektor fitur merupakan representasi numerik dari data yang

memungkinkan model pembelajaran mesin dan teknik kecerdasan buatan untuk memproses dan memahami informasi dengan cara yang terstruktur.

1. Peran Vektor Fitur dalam Kecerdasan Komputasional

   Vektor fitur menyediakan format yang terstruktur untuk data mentah, yang memudahkan algoritma untuk mempelajari pola, melakukan klasifikasi, regresi, atau tugas-tugas lainnya. Representasi ini memungkinkan berbagai teknik kecerdasan komputasional untuk memproses data secara efektif dan menghasilkan keputusan atau prediksi.

2. Implementasi dalam Algoritma Klasifikasi

   Dalam algoritma klasifikasi, vektor fitur digunakan untuk mengidentifikasi kelas dari data input. Beberapa contoh algoritma klasifikasi yang memanfaatkan vektor fitur adalah:

   - K-Nearest Neighbors (KNN): Algoritma ini mengklasifikasikan data berdasarkan kedekatannya dengan data pelatihan. Vektor fitur dari data uji dibandingkan dengan vektor fitur data pelatihan menggunakan metrik jarak (seperti Euclidean distance)

untuk menentukan kelas yang paling umum di antara tetangga terdekat.

- Support Vector Machines (SVM): SVM menggunakan vektor fitur untuk menemukan hyperplane yang memisahkan data ke dalam kelas-kelas yang berbeda. Fitur-fitur tersebut diubah menjadi ruang dimensi tinggi untuk menemukan hyperplane optimal yang memisahkan kelas-kelas tersebut.

3. Implementasi dalam Algoritma Regresi

   Dalam algoritma regresi, vektor fitur digunakan untuk memprediksi nilai kontinu berdasarkan input data. Beberapa algoritma regresi yang menggunakan vektor fitur adalah:

   - Regresi Linier: Vektor fitur digunakan untuk menghitung parameter model (koefisien) yang memetakan hubungan antara fitur dan variabel target. Regresi linier mengasumsikan hubungan linear antara fitur dan target.
   - Regresi Polinomial: Menggunakan vektor fitur untuk merepresentasikan data dalam bentuk polinomial untuk menangkap hubungan non-linear antara fitur dan target.

4. Implementasi dalam *Clustering*

   Clustering adalah teknik yang digunakan untuk mengelompokkan data berdasarkan kesamaan fitur. Vektor fitur memainkan peran penting dalam menentukan kemiripan antara data:

   - K-Means *Clustering*: Algoritma ini mengelompokkan data menjadi k cluster berdasarkan kedekatan vektor fitur dengan centroid cluster. Vektor fitur digunakan untuk menghitung jarak antara titik data dan centroid.
   - Hierarchical *Clustering*: Menggunakan vektor fitur untuk membangun hierarki cluster melalui penggabungan atau pemisahan iteratif berdasarkan kesamaan fitur.

5. Implementasi dalam Jaringan Saraf Tiruan (Neural Networks)

   Vektor fitur adalah input langsung ke lapisan pertama dari jaringan saraf tiruan (neural networks). Dalam proses pelatihan, fitur ini digunakan untuk menghitung aktivasi neuron dan pembaruan bobot. Beberapa contoh implementasi meliputi:

- Feedforward Neural Networks (FNN): Menggunakan vektor fitur sebagai input untuk menghasilkan output melalui jaringan berlapis dengan fungsi aktivasi non-linear.
- Convolutional Neural Networks (CNN): Dalam pengolahan citra, vektor fitur dari setiap patch citra diproses melalui lapisan konvolusi untuk mendeteksi pola lokal.

6. Implementasi dalam Pengolahan Teks dan NLP
   Dalam pemrosesan bahasa alami (NLP), vektor fitur digunakan untuk merepresentasikan kata atau dokumen:
   - Bag of Words (BoW): Setiap kata dalam dokumen direpresentasikan sebagai vektor frekuensi kata. Vektor fitur ini digunakan untuk analisis teks dan klasifikasi dokumen.
   - Term Frequency-Inverse Document Frequency (TF-IDF): Menghitung bobot fitur kata berdasarkan frekuensi kemunculannya dalam dokumen dan keseluruhan korpus untuk menilai kepentingan kata dalam konteks.
7. Implementasi dalam Pengolahan Citra

Dalam pengolahan citra, vektor fitur digunakan untuk merepresentasikan berbagai karakteristik gambar:

- Histogram of Oriented Gradients (HOG): Menghitung histogram gradien dalam jendela kecil gambar untuk mendeteksi fitur tekstural dan struktural.
- SIFT (Scale-Invariant Feature Transform): Mengekstraksi fitur titik kunci dari gambar yang invariant terhadap skala dan rotasi.

BAB V
PRINCIPAL COMPONENT ANALYSIS

5.1. Definisi dan Tujuan *Principal Component Analysis*

Definisi Principal Component Analysis (PCA)

Principal Component Analysis (PCA) adalah teknik statistika untuk mengurangi dimensi data dengan mempertahankan informasi penting. PCA melakukan transformasi data ke dalam komponen utama, yang merupakan kombinasi linier variabel asli, sehingga data dapat direpresentasikan dalam dimensi lebih rendah. Teknik ini berguna untuk menyederhanakan data yang memiliki banyak variabel berkorelasi.

5.1.1. Tujuan PCA

Tujuan utama dari Principal Component Analysis (PCA) adalah menyederhanakan data kompleks dengan mengurangi jumlah dimensi, sambil mempertahankan sebagian besar informasi penting. Secara spesifik, PCA bertujuan untuk mereduksi dimensi, menghilangkan redundansi, mempermudah visualisasi, mengurangi derau, dan menyederhanakan proses pembelajaran mesin.

5.1.2. Proses Reduksi Dimensi

Reduksi dimensi adalah proses menyederhanakan set data yang memiliki banyak fitur atau variabel menjadi representasi dengan jumlah dimensi yang lebih sedikit tanpa kehilangan informasi penting secara signifikan. Dalam konteks Principal Component Analysis (PCA), reduksi dimensi dicapai dengan mengidentifikasi komponen utama yang mewakili variabilitas terbesar dalam data. Proses ini tidak hanya mengurangi kompleksitas data tetapi juga membantu meningkatkan kinerja algoritma pembelajaran mesin dan analisis data.

5.1.3. Langkah-Langkah Proses Reduksi Dimensi dengan PCA

1. Standardisasi Data

   Standardisasi data merupakan proses normalisasi setiap variabel sehingga setiap fitur memiliki rata-rata nol dan varians bernilai satu. Proses ini dilakukan untuk memastikan bahwa semua fitur memiliki bobot yang sama dalam analisis. Dengan demikian, fitur berskala lebih besar tidak akan mendominasi proses PCA, sehingga

hasil akhirnya terdistribusi secara normal dan akurat. Persamaan (5.1) menunjukkan formula normalisasi data.

$$z = \frac{x - \bar{x}}{\sigma} \quad (5.1)$$

Keterangan:

x = data asli

$\bar{x}$ = rata-rata data asli

σ = standar deviasi dari data tersebut.

2. Menghitung Matriks Kovarians

   Matriks kovarians sebagaimana pada persamaan (5.2) mengukur seberapa besar variabel-variabel saling berkorelasi satu sama lain. Jika dua variabel berkorelasi tinggi, maka salah satu dari keduanya bisa diwakili oleh komponen utama.

$$\boldsymbol{C} = \frac{1}{n-1}\sum_{i=1}^{n}(x_i - \bar{x})(x_i - \bar{x})^T \quad (5.2)$$

   Keterangan:

$\boldsymbol{C}$ = matriks kovarians

n = jumlah observasi

$X\ dan\ \bar{X}$ = vektor data asli dan rata-rata vektor data asli

3. Menghitung *Nilai eigen* dan *Vektor eigen*

   Nilai eigen (λ) memberikan informasi tentang seberapa banyak varians dalam data yang diwakilkan oleh masing-masing komponen utama. Sementara itu, *vektor eigen* (**v**) menentukan arah komponen utama.

   Komponen utama pertama (principal component 1) memiliki *nilai eigen* terbesar, yang berarti mewakilkan bagian terbesar dari varians dalam data. Komponen utama kedua mewakilkan varians terbesar kedua, dan seterusnya.

   Untuk menghitung *Nilai eigen* (λ) digunakan persamaan karakteristik dari matriks kovarians:

$$det(\boldsymbol{C} - \lambda I) = 0 \quad (5.3)$$

   Keterangan:

   $\boldsymbol{C}$ = matriks kovarians

λ = matriks nilai eigen yang dipilih

I = matriks identitas

Nilai eigen diperoleh dengan substitusi matriks kovarians ke dalam persamaan (5.3). Selanjutnya *vektor eigen* (**v***)* diperoleh dengan mensubstitusi nilai-nilai eigen ke dalam persamaan (5.4)

$$(\boldsymbol{C} - \lambda I)\boldsymbol{v} = \mathbf{0} \qquad (5.4)$$

Keterangan:

$\boldsymbol{C}$ = matriks kovarians

λ = matriks nilai eigen yang dipilih

I = matriks identitas

$\boldsymbol{v}$ =matriks vektor eigen

4. Memilih Komponen Utama (*Principal Components*)

   Dari kumpulan vektor eigen yang diperoleh, kita dapat memilih beberapa komponen utama (Principal Components) yang mewakili sebagian besar varians dalam set data. Pemilihan ini berdasarkan Nilai eigen, semakin besar nilai eigen maka semakin signifikan komponen utama tersebut.

Sebaiknya memilih komponen-komponen utama yang secara kumulatif mewakili sekitar 90-95% dari varians total.

5. Merepresentasikan Data dalam Ruang Dimensi Lebih Rendah

   Mentransformasi data asli ke dimensi lebih rendah menggunakan matriks proyeksi yang terdiri dari vektor eigen komponen utama. Data hasil transformasi memiliki dimensi lebih sedikit namun tetap mempertahankan sebagian besar informasi penting. Persamaan (5.5) menunjukkan reduksi dimensi.

$$Y = X \,.\, W \tag{5.5}$$

   Keterangan:

   X = data asli

   W = matriks vektor eigen yang dipilih

   Y = data yang direduksi dimensinya

5.2. Penggunaan Matriks Kovarians

Dalam analisis data multidimensi, seperti yang dilakukan pada Principal Component Analysis (PCA),

matriks kovarians digunakan untuk memahami korelasi antara semua variabel dalam set data. Matriks ini membantu mengidentifikasi hubungan linier antara variabel yang penting dalam proses reduksi dimensi.

5.2.1. Fungsi Matriks Kovarians dalam PCA

Matriks kovarians adalah inti dari PCA yang digunakan untuk mengetahui korelasi antara variabel-variabel dalam set data. Dengan menghitung matriks kovarians, PCA dapat menemukan pola dalam data yang menggambarkan arah variabilitas terbesar, yang kemudian diwakili oleh komponen utama (Principal Components). Komponen utama ini adalah kombinasi linier dari variabel asli, yang mengarahkan variabilitas terbesar dalam data.

Langkah-langkah penggunaan matriks kovarians dalam PCA adalah sebagai berikut:

1. Membangun Matriks Kovarians

   Matriks kovarians dibangun dari set data yang sudah distandardisasi. Matriks kovarians C untuk dua variabel X dan Y didefinisikan pada persamaan (5.6) berikut:

$$C(X,Y) = \frac{1}{n-1}\sum_{i=1}^{n}(X_i - \bar{X}) - (Y_i - \bar{Y}) \quad (5.6)$$

Keterangan:

$\bar{X}\ dan\ \bar{Y}$ = rata-rata variabel X dan Y

n = jumlah observasi

Y = data yang direduksi dimensinya

Untuk set data yang memiliki lebih dari dua variabel, matriks kovarians menjadi matriks simetris dengan dimensi m x m, di mana m adalah jumlah variabel dalam set data.

Sebagai contoh, untuk set data dengan tiga variabel X_1, X_2, X_3 matriks kovariansnya akan berbentuk:

$$C = \begin{bmatrix} cov(X_1,X_1) & cov(X_1,X_2) & cov(X_1,X_3) \\ cov(X_2,X_1) & cov(X_2,X_2) & cov(X_2,X_3) \\ cov(X_3,X_1) & cov(X_3,X_2) & cov(X_3,X_3) \end{bmatrix}$$

Dimana

$cov(X_1,X_1), cov(X_2,X_2), dan\ cov(X_3,X_3)$ adalah varians masing-masing variabel, sementara elemen-elemen lain menunjukkan kovarians antara pasangan variabel.

2. Menemukan Arah Variabilitas Terbesar

   Menemukan arah variabilitas terbesar dalam data dilakukan dengan menghitung *Nilai eigen* dan *vektor eigen* dari matriks kovarians.

   Vektor eigen dari matriks kovarians adalah vektor yang menunjukkan arah dari komponen utama, sedangkan Nilai eigen menunjukkan besarnya variasi data sepanjang arah tersebut. Komponen utama pertama (principal component 1) adalah arah dengan variabilitas terbesar dalam data, yang ditentukan oleh vektor eigen dengan nilai eigen terbesar.

3. Reduksi Dimensi

   Dengan memilih beberapa komponen utama yang memiliki Nilai eigen terbesar, data dapat direpresentasikan dalam ruang berdimensi lebih rendah.

   Contoh Penggunaan Matriks Kovarians dalam PCA

   Misalkan kita memiliki set data yang terdiri dari dua variabel, yaitu suhu dan kecepatan angin suatu wilayah. Langkah pertama dalam PCA adalah menghitung matriks kovarians dari variabel-variabeltersebut. Jika hasil matriks kovarians menunjukkan bahwa kedua variabel

memiliki kovarians positif yang tinggi, maka hal tersebut menunjukkan bahwa ada hubungan linear positif di antara variabel tersebut, yakni ketika suhu naik maka kecepatan angin cenderung juga meningkat.

Dengan menghitung nilai eigen dan vektor eigen dari matriks kovarians tersebut, kita dapat menentukan komponen utama yang menjelaskan sebagian besar hubungan antara suhu dan kecepatan angin, dan kemudian kita dapat mengurangi data menjadi satu dimensi saja yang menyimpan informasi yang paling relevan.

Keuntungan Menggunakan Matriks Kovarians dalam PCA

1. Mengidentifikasi Hubungan Linier

   Matriks kovarians membantu mengidentifikasi hubungan linier antara variabel-variabel. Hubungan ini penting untuk menentukan arah di mana variabilitas terbesar dalam data terjadi, yang kemudian digunakan untuk menentukan komponen utama.
2. Mengurangi Redundansi Data

Dengan memahami korelasi antara variabel, kita dapat mengurangi redundansi dalam data. Jika dua variabel sangat berkorelasi, salah satunya mungkin dapat dihilangkan tanpa kehilangan banyak informasi. Proses ini penting dalam reduksi dimensi.

3. Memudahkan Visualisasi dan Analisis
   Setelah data direduksi dimensinya, hasilnya lebih mudah divisualisasikan dan dianalisis. Matriks kovarians berperan penting dalam menyaring variabel-variabel yang memberikan sedikit kontribusi terhadap variabilitas keseluruhan data.

5.3. Vektor Eigen dan Nilai Eigen

5.3.1. Definisi Vektor Eigen dan Nilai Eigen

Vektor eigen dan nilai eigen adalah konsep matematis yang sangat penting dalam analisis data, terutama dalam bidang *machine learning* dan kecerdasan buatan. Dalam konteks PCA, keduanya digunakan untuk memahami struktur data dan melakukan reduksi dimensi dengan cara yang efisien.

a. Vektor eigen adalah vektor yang tidak berubah arah ketika transformasi linear

diterapkan padanya. Dengan kata lain, saat kita menerapkan transformasi pada sebuah vektor eigen, hasilnya hanyalah pemanjangan atau pemendekan vektor tersebut, bukan perubahan arah.

b. Nilai eigen adalah skalar yang terkait dengan vektor eigen dan menunjukkan seberapa besar vektor eigen tersebut diperpanjang atau dipendekkan selama transformasi. Jika nilai eigen positif dan besar, maka vektor eigen diperpanjang dengan skala besar. Jika nilai eigen mendekati nol, maka transformasi hampir tidak mengubah panjang vektor.

Secara matematis, hubungan antara vektor eigen dan nilai eigen dinyatakan pada persamaan (5.7) berikut:

$$(\boldsymbol{C} - \lambda I)\boldsymbol{v} = 0$$

(5.7)

Keterangan:

$\boldsymbol{C}$ = matriks kovarians

$\boldsymbol{v}$ = vektor eigen

λ = nilai eigen

I = matriks identitas

Persamaan (5.7) menunjukkan bahwa ketika matriks C diterapkan pada vektor eigen ($\boldsymbol{v}$), hasilnya hanyalah pemanjangan atau pemendekan sebesar λ.

5.3.2. Peran Vektor eigen dan Nilai eigen dalam PCA

Dalam PCA, kita ingin menemukan arah di mana data memiliki variansi terbesar. Vektor eigen dan nilai eigen sangat penting dalam proses ini, karena mereka memberikan cara untuk menentukan arah dan besarnya variabilitas dalam data.

a. Menentukan Arah Variasi Terbesar (Vektor eigen)

   Setelah matriks kovarians dari set data dihitung, langkah selanjutnya adalah menentukan arah dalam ruang data di mana variansi terbesar terjadi. Arah ini ditentukan oleh vektor eigen. Setiap vektor eigen menggambarkan suatu arah spesifik dalam ruang data, yang disebut komponen utama

(*principal component*). Vektor eigen dari matriks kovarians adalah vektor yang menunjukkan arah dari variabilitas terbesar dalam data.

b. Menentukan Besarnya Variasi (Nilai eigen)
Nilai eigen memberi informasi tentang seberapa besar variabilitas di sepanjang arah yang ditentukan oleh vektor eigen. Komponen utama yang memiliki nilai eigen terbesar adalah yang paling penting karena menjelaskan bagian terbesar dari variasi dalam data. Nilai eigen yang lebih kecil menunjukkan arah di mana variasi dalam data lebih sedikit dan mungkin tidak begitu penting untuk analisis lebih lanjut.

5.3.3. Keuntungan Menggunakan Vektor eigen dan Nilai eigen

a. Reduksi Dimensi yang Efisien
Dengan memanfaatkan vektor eigen dan Nilai eigen, kita dapat mengurangi dimensi set data secara efisien. Data yang memiliki banyak variabel dapat disederhanakan menjadi beberapa komponen utama yang

tetap mempertahankan sebagian besar informasi penting.

b. Meningkatkan Interpretabilitas

Setelah reduksi dimensi, set data menjadi lebih mudah diinterpretasikan. Komponen utama yang dihasilkan oleh vektor eigen membantu dalam visualisasi dan pemahaman pola dalam data yang sebelumnya tersembunyi.

c. Memaksimalkan Variansi

Proses PCA memastikan bahwa komponen utama yang dipilih memaksimalkan variansi dalam data. Ini memungkinkan kita untuk menangkap fitur atau karakteristik yang paling signifikan dari set data dalam dimensi yang lebih kecil.

5.4. Aplikasi PCA dalam Pengolahan Data

PCA adalah metode statistik yang sangat populer untuk reduksi dimensi dalam set data besar dan kompleks. Teknik ini secara luas digunakan di berbagai bidang seperti pembelajaran mesin, pengolahan gambar, biologi, ekonomi, serta pengenalan pola. PCA membantu dalam menyederhanakan set data tanpa menghilangkan informasi penting, menjadikannya alat

yang sangat efisien dalam analisis data. Beberapa aplikasi utama PCA dalam pengolahan data diantaranya: pengurangan dimensi data, visualisasi data, praproses data dalam machinelLearning, pengolahan gambar dan *computer vision*, pengolahan sinyal, pengenalan pola, analisis data biomedis dan pengolahan data keuangan.

5.5. PCA untuk Visualisasi Data

PCA adalah teknik yang sangat efektif untuk reduksi dimensi, yang juga secara luas digunakan untuk visualisasi data. Ketika kita bekerja dengan set data yang memiliki banyak dimensi (misalnya, puluhan atau bahkan ribuan fitur), sulit untuk memvisualisasikan data dan mengidentifikasi pola tersembunyi. PCA membantu mengatasi masalah ini dengan memproyeksikan data ke dalam ruang berdimensi lebih rendah, biasanya dua atau tiga dimensi, sehingga data dapat divisualisasikan dalam grafik sederhana. Teknik ini memudahkan interpretasi dan analisis, terutama dalam konteks *Clustering*, deteksi outlier, dan pengenalan pola.

1. Reduksi Dimensi untuk Visualisasi

   Dalam banyak aplikasi, data yang dianalisis berada dalam dimensi yang tinggi. Misalnya,

sebuah set data yang mengandung citra beresolusi tinggi dapat memiliki ribuan piksel sebagai fitur. Visualisasi data dalam ruang dimensi tinggi tidak mungkin dilakukan secara langsung karena keterbatasan manusia dalam memahami dimensi lebih dari tiga. PCA memberikan solusi dengan mengurangi jumlah dimensi sambil tetap mempertahankan informasi penting. PCA bekerja dengan menemukan komponen utama yang mengandung variansi terbesar dari data dan memproyeksikan data ke arah komponen tersebut.

Contoh: Bayangkan kita memiliki set data genetik yang mengandung ekspresi ribuan gen. Dengan menerapkan PCA, kita dapat mereduksi ribuan dimensi ini menjadi dua atau tiga komponen utama yang mempertahankan sebagian besar informasi penting dari set data. Ini memungkinkan kita untuk memvisualisasikan data genetik tersebut dalam bentuk scatter plot sederhana.

2. Visualisasi dengan Komponen Utama

   Setelah PCA diterapkan, komponen utama pertama dan kedua yang dihasilkan dapat

digunakan untuk memvisualisasikan set data. PCA memproyeksikan data dari dimensi tinggi ke ruang berdimensi lebih rendah, biasanya ke dua dimensi yang mewakili komponen utama pertama (PC1) dan komponen utama kedua (PC2). Scatter plot dari PC1 dan PC2 sering digunakan untuk memvisualisasikan distribusi data dan pola yang mungkin tidak terlihat di dimensi asli.

Contoh: Dalam pengelompokan (*Clustering*), PCA memungkinkan kita melihat kluster data yang mungkin terbentuk berdasarkan dua atau tiga komponen utama. Misalnya, dalam data yang melibatkan pelanggan suatu toko, PCA dapat memetakan pelanggan berdasarkan perilaku belanja mereka, dan kita dapat melihat kluster yang menggambarkan segmen pasar yang berbeda.

Dengan memvisualisasikan data dalam dua atau tiga dimensi, kita bisa dengan mudah melihat apakah data membentuk kluster, atau apakah ada outlier yang tampak jelas dalam data.

3. Deteksi *Outlier*

   Selain digunakan untuk visualisasi kluster, PCA juga sangat efektif dalam mendeteksi *outlier*

atau data yang tidak mengikuti pola umum dalam set data. Outlier sering kali dapat memengaruhi hasil analisis data atau model pembelajaran mesin, sehingga penting untuk mendeteksinya sejak awal. Dengan memproyeksikan data ke dalam dua atau tiga dimensi utama, *outlier* menjadi lebih mudah terlihat sebagai titik yang terletak jauh dari distribusi data utama.

Contoh: Dalam sebuah set data yang digunakan untuk menganalisis kebiasaan keuangan, PCA dapat membantu mengidentifikasi pelanggan yang memiliki pola pengeluaran yang sangat berbeda dari mayoritas pelanggan lainnya. *Outlier* tersebut mungkin menunjukkan adanya penipuan atau perilaku keuangan yang tidak normal.

4. Visualisasi untuk Data Multikelas

   Dalam klasifikasi dengan banyak kelas, visualisasi menggunakan PCA sangat berguna untuk memahami bagaimana data dari kelas yang berbeda didistribusikan. Setelah PCA mereduksi dimensi data, kita dapat memplot data dari masing-masing kelas menggunakan warna atau simbol yang berbeda untuk

mengidentifikasi pola atau tumpang tindih antar kelas.

Contoh: Dalam klasifikasi data bunga dari set data Iris, PCA dapat digunakan untuk memproyeksikan empat fitur (panjang kelopak, lebar kelopak, panjang daun, dan lebar daun) menjadi dua komponen utama. Dengan scatter plot dari dua komponen utama ini, kita dapat dengan mudah melihat bagaimana tiga spesies bunga terpisah dalam ruang dua dimensi.

Dengan teknik ini, PCA memberikan pemahaman visual tentang sejauh mana setiap kelas data saling berdekatan atau terpisah, yang sangat berguna dalam penilaian model klasifikasi.

5. Penerapan PCA untuk Data Gambar dan Video

   Dalam bidang pengolahan gambar dan video, PCA digunakan untuk memvisualisasikan gambar yang memiliki ratusan atau ribuan piksel sebagai fitur. PCA dapat mereduksi jumlah piksel ini menjadi beberapa komponen utama yang tetap mempertahankan struktur penting dari gambar. Hal ini juga berlaku dalam pengolahan video, di mana setiap frame dalam video dapat diperlakukan sebagai data

berdimensi tinggi, dan PCA membantu memproyeksikan informasi visual tersebut ke dalam ruang berdimensi lebih rendah.

Contoh: Dalam pengenalan wajah, PCA dapat digunakan untuk mereduksi dimensi gambar wajah dan hanya mempertahankan fitur-fitur yang paling penting. Hasil proyeksi PCA ini dapat divisualisasikan untuk menunjukkan variabilitas utama di antara gambar-gambar wajah dalam set data.

6. Keuntungan Visualisasi Menggunakan PCA

   Visualisasi data menggunakan PCA memiliki beberapa keuntungan utama:

   - Pemahaman Pola yang Lebih Baik: PCA memungkinkan kita untuk melihat pola dan kluster dalam data yang mungkin tersembunyi dalam dimensi asli yang lebih tinggi.
   - Deteksi Anomali: Outlier dan anomali dalam data lebih mudah diidentifikasi setelah PCA mereduksi dimensi data.
   - Mengurangi Overfitting: Dengan mengurangi jumlah fitur sebelum visualisasi, PCA membantu mencegah overfitting, terutama ketika data yang

divisualisasikan akan digunakan untuk model pembelajaran mesin.

- Praproses Data untuk Algoritma Lain: Visualisasi dengan PCA sering kali digunakan sebagai langkah awal sebelum menerapkan algoritma pembelajaran mesin lain, seperti *Clustering* atau klasifikasi.

7. Keterbatasan Visualisasi dengan PCA

   Meskipun PCA sangat berguna untuk visualisasi, terdapat beberapa keterbatasan yang perlu diperhatikan:

   - Non-linearitas: PCA adalah metode linier, sehingga tidak selalu cocok untuk data yang memiliki hubungan non-linier antara fitur-fiturnya. Dalam kasus seperti ini, metode lain seperti t-SNE atau UMAP mungkin lebih efektif.
   - Informasi Hilang: Meskipun PCA menjaga variansi terbesar dalam data, ada informasi lain yang mungkin hilang dalam proses reduksi dimensi. Proyeksi data ke dalam dua atau tiga dimensi mungkin tidak cukup untuk menangkap seluruh kompleksitas set data asli.

BAB VI
LINEAR DISCRIMINANT ANALYSIS (*LDA*)

6.1. Pendahuluan

6.1.1. Kecerdasan Komputasional dan Perannya pada *Linear Discriminant Analysis* (LDA)

Kecerdasan komputasional merupakan cabang dari kecerdasan buatan yang berfokus pada pengembangan algoritma yang mampu meniru cara berpikir manusia dalam menyelesaikan masalah yang kompleks. Sistem ini biasanya menggunakan pendekatan berbasis adaptasi, pembelajaran, dan pengoptimalan untuk menganalisis dan memahami data yang kompleks serta dinamis. Kecerdasan komputasional melibatkan metode-metode seperti jaringan saraf tiruan, algoritma genetika, *fuzzy logic*, dan metode statistik seperti *Linear Discriminant Analysis* (LDA) (Floares, 2012; Tharwat *et al.*, 2017).

Dalam konteks pemrosesan data yang kompleks, LDA memiliki peran yang signifikan dalam membedakan dan mengklasifikasikan data berdasarkan fitur-fitur yang relevan. LDA digunakan sebagai teknik pengurangan dimensi

yang bertujuan untuk mencari proyeksi linear yang memaksimalkan separasi antara kelas-kelas data yang berbeda. Metode ini sangat berguna ketika data yang dianalisis memiliki banyak dimensi atau variabel, yang dapat menyebabkan masalah dalam efisiensi pemrosesan dan akurasi klasifikasi. LDA bekerja dengan menemukan garis atau ruang yang memaksimalkan jarak antar kelas dan meminimalkan variasi dalam kelas yang sama.

Dalam kecerdasan komputasional, LDA sering digunakan dalam aplikasi pengenalan pola, seperti pengenalan wajah, di mana sistem harus mampu membedakan antara individu yang berbeda berdasarkan fitur visual yang kompleks (Tang *et al.*, 2014; Alexander, Irizarry and Bravo, 2023). Dengan memproyeksikan data ke ruang dimensi yang lebih rendah, LDA memungkinkan sistem kecerdasan komputasional untuk bekerja lebih efisien tanpa kehilangan banyak informasi penting. Peran LDA dalam kecerdasan komputasional juga meliputi pengelompokan data yang tersebar dalam dimensi tinggi menjadi kategori-kategori yang lebih sederhana, yang mempermudah proses analisis dan prediksi. LDA memungkinkan pemodelan data yang lebih

terstruktur, yang sangat penting dalam mengatasi masalah ketidakpastian dan ketidakpastian yang sering terjadi dalam analisis data kompleks. Kecerdasan komputasional melalui LDA tidak hanya meningkatkan akurasi pemrosesan data tetapi juga membantu dalam mengoptimalkan proses pengambilan keputusan dalam berbagai aplikasi praktis, seperti dalam sistem pengenalan wajah, diagnosis medis, dan klasifikasi teks (Sun and Scanlon, 2019; Füller *et al.*, 2022; Villegas-Ch, Gaibor-Naranjo and Sanchez-Viteri, 2024).

6.1.2. *Linear Discriminant Analysis* (LDA) sebagai Metode dalam Pengelompokan Data

Linear Discriminant Analysis (LDA) merupakan salah satu metode statistik yang digunakan dalam pengelompokan data dan pengurangan dimensi. LDA dirancang untuk memisahkan data ke dalam kelompok-kelompok berdasarkan karakteristik tertentu, dengan tujuan utama memaksimalkan separasi antara kelas yang berbeda. Sebagai metode pengelompokan data, LDA berperan penting dalam mengklasifikasikan data multivariat, khususnya ketika data yang

digunakan memiliki banyak fitur yang saling terkait. LDA mencari proyeksi linear yang memaksimalkan jarak antar kelas, sekaligus meminimalkan variasi dalam kelas yang sama. Proses ini dilakukan dengan menghitung matriks kovarians antar kelas dan dalam kelas untuk menemukan ruang proyeksi terbaik yang memungkinkan separasi optimal antara kelompok data yang berbeda (Sodhi and Lal, 2013; Chen *et al.*, 2020; Nuraini *et al.*, 2023).

Dalam proses pengelompokan, LDA bekerja dengan cara mengasumsikan bahwa distribusi data dalam setiap kelas mengikuti distribusi normal multivariat. LDA kemudian memodelkan distribusi tersebut untuk menghitung garis pemisah antar kelas. LDA tidak hanya memproyeksikan data ke ruang yang lebih rendah, tetapi juga menjaga hubungan antar fitur yang ada, sehingga informasi penting tidak hilang dalam proses pengelompokan. LDA sangat cocok digunakan ketika data memiliki struktur linear dan distribusi yang relatif seragam antar kelas. Penggunaan LDA banyak ditemukan dalam berbagai aplikasi praktis. Contoh, dalam pengenalan pola seperti pengenalan wajah, LDA dapat memproyeksikan gambar wajah ke ruang

dimensi yang lebih rendah namun tetap mempertahankan karakteristik penting yang membedakan satu individu dengan individu lainnya.

LDA juga diterapkan dalam klasifikasi teks dan pengelompokan dokumen, di mana LDA membantu mengidentifikasi kategori yang berbeda berdasarkan pola-pola kata. LDA merupakan alat yang efektif dalam pemrosesan data berukuran besar dan kompleks, memberikan solusi yang efisien untuk masalah klasifikasi dengan hasil yang terstruktur dan mudah diinterpretasikan (Suadaa and Purwarianti, 2016; Mirończuk and Protasiewicz, 2018; Kim and Gil, 2019; Dogra *et al.*, 2022, 2022).

6.1.3. LDA pada Kecerdasan Komputasional

Penggunaan *Linear Discriminant Analysis* (LDA) dalam kecerdasan komputasional memiliki peran yang signifikan, terutama dalam tugas-tugas klasifikasi dan pengenalan pola. Kecerdasan komputasional, yang mencakup teknik-teknik seperti jaringan saraf tiruan, algoritma evolusi, dan metode berbasis logika fuzzy, sering kali harus menangani data yang sangat besar dan kompleks.

LDA menjadi alat yang sangat efektif untuk mengurangi dimensi data sambil mempertahankan informasi yang relevan untuk tujuan klasifikasi. Dengan memproyeksikan data ke ruang dimensi yang lebih rendah, LDA memfasilitasi pemrosesan yang lebih cepat tanpa mengorbankan akurasi hasil, menjadikannya sangat berguna dalam berbagai aplikasi praktis (Barnova *et al.*, 2023). LDA membantu membedakan antara kelas-kelas yang berbeda dengan memaksimalkan jarak antara kelompok-kelompok data yang berbeda dalam ruang fitur.

Dalam pengenalan wajah, misalnya, LDA dapat memisahkan karakteristik wajah individu yang berbeda dan memproyeksikannya ke ruang yang lebih kecil, sehingga sistem dapat dengan cepat mengidentifikasi identitas seseorang berdasarkan data citra yang diinput. LDA juga sering digunakan dalam klasifikasi teks, di mana algoritma ini mampu mengelompokkan dokumen atau teks ke dalam kategori tertentu berdasarkan pola kata yang muncul. Dengan cara ini, LDA membantu sistem kecerdasan komputasional dalam tugas pengelompokan dokumen, seperti klasifikasi berita atau analisis sentimen.

Penggunaan LDA dalam kecerdasan komputasional tidak hanya membantu dalam mengurangi kompleksitas data, tetapi juga meningkatkan akurasi dan efisiensi dalam pemrosesan. LDA memungkinkan sistem untuk lebih mudah menemukan struktur data yang tersembunyi, memberikan hasil klasifikasi yang lebih andal dalam situasi di mana data memiliki banyak fitur atau variabel (Farkhod *et al.*, 2021; Romero *et al.*, 2022; Barnova *et al.*, 2023; Vora *et al.*, 2023).

6.2. Teori Dasar LDA

6.2.1. Konsep Dasar LDA

Linear Discriminant Analysis (LDA) adalah metode statistik yang dirancang untuk memisahkan atau mendiskriminasi kelas-kelas yang berbeda dalam dataset. Tujuan utama LDA adalah menemukan kombinasi fitur yang menghasilkan pemisahan optimal antara dua atau lebih kelas (Farag and Elhabian, 2008; Graf, Zeldovich and Friedrich, 2024). Dalam konteks pemisahan kelas, LDA bekerja dengan cara memaksimalkan jarak antara rata-rata (mean)

kelas yang berbeda, sambil meminimalkan variabilitas di dalam kelas yang sama.

Ini dilakukan dengan menghitung varians antar-kelas dan dalam-kelas, di mana LDA berupaya memaksimalkan varians antar-kelas agar kelas lebih terpisah, sekaligus meminimalkan varians dalam-kelas untuk mengelompokkan data dalam satu kelas secara lebih kompak. LDA menggunakan kombinasi linear dari fitur input untuk memisahkan kelas-kelas melalui apa yang disebut diskriminasi linear. Diskriminasi ini diwujudkan dalam bentuk garis atau hyperplane yang memisahkan ruang multidimensi menjadi wilayah-wilayah yang berbeda, memungkinkan identifikasi kelas yang berbeda berdasarkan nilai proyeksi data ke ruang baru tersebut (Farag and Elhabian, 2008; Tang *et al.*, 2014).

Gambar 6.1 Konsep Dasar LDA

Proses ini melibatkan perhitungan mean dan kovarians kelas, diikuti dengan penentuan sumbu optimal untuk memisahkan kelas-kelas. LDA juga berguna untuk pengurangan dimensi, di mana data diubah ke dalam ruang yang lebih kecil namun tetap mempertahankan informasi penting terkait pemisahan kelas. LDA melakukan pengurangan dimensi dengan menghasilkan proyeksi data ke ruang dengan dimensi $C - 1$, di mana C adalah jumlah kelas. LDA bekerja dengan asumsi bahwa setiap kelas mengikuti distribusi Gaussian dan memiliki kovarians yang sama, serta kelas-kelas dapat dipisahkan secara linear (Heintz, Francart and Bertrand, 2023; Kak, 2023).

Asumsi-asumsi ini menjadikan LDA sangat efektif pada dataset yang memiliki pemisahan linear dan distribusi yang relatif seragam antar kelas. Dimensi $C - 1$ dalam konteks Linear Discriminant Analysis (LDA) merujuk pada jumlah dimensi dalam ruang baru setelah data diproyeksikan untuk memisahkan kelas-kelas. Untuk memahami ini, penting untuk menyadari bahwa dalam LDA, data dari ruang fitur asli (yang mungkin memiliki banyak dimensi) diproyeksikan ke ruang yang lebih rendah di mana pemisahan

antar kelas menjadi lebih jelas. C adalah jumlah kelas dalam dataset**.** Misalnya, jika memiliki dataset dengan 3 kelas berbeda, maka $C = 3$. jika $C = 3$, LDA akan memproyeksikan data ke dalam ruang 2 dimensi (karena $C - 1 = 3 - 1 = 2$).

6.2.2. Linear Discriminant Analysis (LDA) dan Kecerdasan Komputasional

Linear Discriminant Analysis (LDA) merupakan salah satu metode yang banyak digunakan dalam kecerdasan komputasional, terutama untuk klasifikasi data dan pengurangan dimensi. LDA memanfaatkan kombinasi linear fitur-fitur untuk memaksimalkan pemisahan antar kelas dalam dataset, sehingga sangat efektif untuk diterapkan pada berbagai aplikasi kecerdasan komputasional yang membutuhkan analisis data yang kompleks.

Salah satu penerapan utama LDA dalam kecerdasan komputasional adalah dalam klasifikasi data, di mana LDA bekerja untuk memisahkan kelas-kelas berdasarkan data yang ada (Nasser *et al.*, 2024). Dalam konteks klasifikasi, LDA berperan penting dalam membedakan kelas yang berbeda, misalnya dalam pengenalan wajah, pengolahan

citra medis, atau klasifikasi teks. LDA digunakan untuk menemukan garis atau hyperplane yang memisahkan data dari beberapa kelas sehingga keputusan klasifikasi dapat diambil dengan lebih akurat. Selain klasifikasi, LDA juga memiliki aplikasi dalam pengenalan pola, pengelompokan data, dan pengurangan dimensi. Pada pengenalan pola, LDA membantu mendeteksi dan mengklasifikasikan pola-pola yang ada di dalam dataset, misalnya dalam pengenalan karakter tulisan tangan.

Pengelompokan data menggunakan LDA dapat mempermudah analisis dalam menemukan kelompok yang homogen dalam data. LDA sering digunakan dalam pengurangan dimensi untuk mengurangi jumlah fitur tanpa kehilangan informasi yang relevan, sehingga mempermudah pemrosesan data yang lebih efisien. Beberapa contoh nyata implementasi LDA termasuk pengenalan wajah, di mana LDA membantu memisahkan fitur wajah yang penting untuk membedakan individu.

Dalam klasifikasi teks, LDA digunakan untuk memproyeksikan data teks ke dalam ruang dengan dimensi lebih rendah untuk klasifikasi dokumen.

Dalam pengolahan citra, LDA mempermudah analisis citra dengan mengekstraksi fitur utama yang membedakan kategori-kategori objek dalam gambar (Zheng, 2022).

6.3. Keuntungan dan Keterbatasan LDA

Linear Discriminant Analysis (LDA) memiliki sejumlah keunggulan yang membuatnya populer dalam banyak aplikasi pembelajaran mesin dan kecerdasan komputasional. Salah satu keunggulan utama LDA adalah interpretasinya yang sederhana. LDA memberikan pemahaman yang jelas tentang bagaimana data diproyeksikan ke dalam ruang baru, di mana kelas-kelas yang berbeda dipisahkan secara optimal.

Hal ini membuat LDA sangat berguna ketika hasil yang dapat diinterpretasikan secara langsung oleh manusia diperlukan. Keunggulan lainnya adalah efisiensi LDA dalam menangani data yang terstruktur dan berlabel, terutama ketika kelas-kelas data terdistribusi secara linear. LDA memaksimalkan pemisahan antar kelas dengan mengoptimalkan varians antar-kelas dan meminimalkan varians dalam-kelas.

Ini menjadikannya sangat efisien untuk klasifikasi dalam situasi di mana data memiliki distribusi yang cukup terstruktur dan mengikuti asumsi distribusi Gaussian (normal) (Haufe *et al.*, 2014; Zheng, 2017). LDA juga bekerja dengan baik pada dataset yang memiliki dimensi yang lebih besar, dengan memproyeksikan data ke dalam ruang dimensi yang lebih rendah tanpa mengorbankan informasi penting. LDA memiliki keterbatasan, terutama ketika berhadapan dengan data nonlinear. Karena LDA bekerja berdasarkan asumsi bahwa kelas-kelas dapat dipisahkan secara linear, metode ini akan kesulitan jika data tidak memiliki pemisahan yang jelas secara linear. LDA kurang efektif ketika harus menangani multiklasifikasi yang rumit, terutama jika kelas-kelas tersebut memiliki distribusi yang tidak normal atau jika varians antar kelas tidak seragam. LDA lebih efektif dibandingkan metode lain, seperti Principal Component Analysis (PCA), ketika label kelas tersedia dan pemisahan kelas adalah tujuan utama.

LDA bekerja optimal dalam klasifikasi yang terstruktur dengan baik dan data terdistribusi normal. Namun, untuk data nonlinear atau

multiklasifikasi yang kompleks, metode seperti Support Vector Machines (SVM) atau Neural Networks mungkin lebih efektif karena mereka mampu menangani data dengan pola yang lebih rumit (Haufe *et al.*, 2014; Mandhala, Sujatha and Devi, 2015; Wang *et al.*, 2017; Yaqoob, Musheer Aziz and verma, 2023).

Tabel 6.1 Keunggulan dan Keterbatasan LDA

Aspek	Keunggulan LDA	Keterbatasan LDA
Interpretasi	Interpretasi yang sederhana, hasil mudah dipahami manusia	Kesulitan dalam menangani data nonlinear
Efisiensi	Efisiensi tinggi pada data terstruktur dan berlabel	Kurang efektif pada multiklasifikasi yang rumit
Distribusi Data	Kemampuan menangani data dengan distribusi	Tidak efektif jika varians antar kelas tidak seragam

	normal (Gaussian)	
Dimensi Data	Mampu mengurangi dimensi tanpa kehilangan informasi penting	Membutuhkan data dengan pemisahan kelas linear yang jelas
Kapan Lebih Efektif	Lebih efektif ketika label kelas tersedia dan pemisahan kelas adalah tujuan utama	Tidak cocok untuk data yang memiliki pola kompleks dan nonlinear
Alternatif Metode	Lebih baik dari PCA dalam klasifikasi yang terstruktur dengan baik	Metode lain seperti SVM atau Neural Networks lebih efektif untuk data nonlinear

6.4. Pengembangan dan Integrasi LDA

Pengembangan dan integrasi Linear Discriminant Analysis (LDA) dengan algoritma lain seperti Support Vector Machine (SVM) dan Neural Networks telah terbukti meningkatkan performa klasifikasi. LDA digunakan sebagai langkah awal untuk pengurangan dimensi dan ekstraksi fitur, memproyeksikan data ke ruang yang lebih rendah sambil memaksimalkan separasi antar kelas. Setelah itu, algoritma seperti SVM dan Neural Networks dapat diterapkan untuk melakukan klasifikasi yang lebih kompleks dan non-linear. Kombinasi ini menghasilkan peningkatan efisiensi komputasi dan akurasi, membuatnya ideal untuk berbagai aplikasi kecerdasan komputasional, seperti pengenalan pola dan pengenalan wajah.

1. Kombinasi LDA dan Support Vector Machine (SVM)
    a. Dalam pendekatan ini, LDA digunakan sebagai teknik pengurangan dimensi atau ekstraksi fitur sebelum mengaplikasikan SVM sebagai classifier.

       Langkah-langkah Formulasi:

       $$y = W_{LDA^x}^T \quad \ldots\ldots\ldots\ldots\ldots (1)$$

       Di mana:

    $\mathcal{X}$adalah data asli

W_DLA adalah matriks proyeksi LDA yang terdiri dari vektor eigen (w_(1,) w_(2,)…,w_(d))

y adalah data yang telah diproyeksikan ke ruang fitur berdimensi rendah

b. SVM untuk Klasifikasi: Setelah proyeksi menggunakan LDA, data y kemudian diberikan ke SVM untuk klasifikasi. SVM berusaha mencari hyperplane yang memisahkan kelas-kelas secara optimal:

$$W_{SVM}^{T}\, y + b = 0 \quad \dots\dots\dots\dots\dots (2)$$

Di mana:

W_(SVM) adalah vektor bobot SVM yang mendefinisikan hyperplane pemisah.

b adalah bias

y adalah data hasil proyeksi dari LDA

2. Kombinasi LDA dan Neural Networks

Dalam kombinasi ini, LDA berperan sebagai langkah awal ekstraksi fitur sebelum data dimasukkan ke Neural Networks (NN) untuk melakukan klasifikasi non-linear yang lebih kompleks.

Langkah-langkah formulasi:

a. LDA untuk Pengurangan Dimensi: Sama seperti pada kombinasi dengan SVM, LDA digunakan untuk memproyeksikan data ke ruang fitur berdimensi rendah:

$$y = W_{LDA^x}^T \ \ldots\ldots\ldots\ldots\ldots (3)$$

Di mana x adalah data asli, dan y adalah data yang telah diproyeksikan ke ruang dimensi lebih rendah oleh LDA.

b. Neural Networks untuk Klasifikasi: Hasil proyeksi LDA y kemudian diberikan sebagai input ke Neural Network. Neural Network akan melakukan klasifikasi berdasarkan lapisan-lapisan neuron yang dihubungkan secara non-linear. Neural Network dengan satu lapisan tersembunyi memiliki fungsi aktivasi sebagai berikut:

$$h = f\ (W_{NN}^T\ y + b) \ldots\ldots\ldots\ldots\ldots (4)$$

Di mana:

W_(NN) adalah matriks bobot dari input ke lapisan tersembunyi

b adalah bias

f (.) adalah fungsi aktivasi (misalnya,ReLU atau sigmoid)

y adalah data hasil proyeksi dari LDA

Output dari lapisan terakhir menghasilkan probabilitas kelas k:

$$\hat{y}_k = g\ (W_{out}^T h + b_{out}) \dots\dots\dots\dots\dots (5)$$

Di mana:

g (.) adalah fungsi aktivasi untuk output (misalnya so kelas)

W_out adalah bobot dari lapisan tersembunyi ke output

b_{out}adalah bias untuk output

Kombinasi ini menggunakan LDA untuk melakukan pengurangan dimensi, dan kemudian Neural Networks digunakan untuk

belajar pola non-linear yang lebih kompleks pada data yang diproyeksikan, pada kombinasi LDA + SVM, LDA digunakan untuk mereduksi dimensi, dan SVM digunakan untuk menemukan hyperplane optimal yang memisahkan kelas-kelas. Pada kombinasi LDA + Neural Networks, LDA digunakan untuk mereduksi dimensi sebelum memproses data melalui lapisan-lapisan non-linear Neural Networks untuk klasifikasi yang lebih kompleks.

6.5. Implementasi LDA dalam Proyek Kecerdasan Komputasional

Implementasi Linear Discriminant Analysis (LDA) dalam proyek kecerdasan komputasional dapat dilihat dalam berbagai aplikasi, termasuk pengenalan wajah, diagnosis medis, dan klasifikasi teks. Salah satu studi kasus nyata yang menonjol adalah penerapan LDA dalam pengenalan wajah untuk sistem keamanan berbasis biometrik. Pada proyek ini, tujuan utamanya adalah untuk meningkatkan keakuratan sistem dalam mengidentifikasi individu dari gambar wajah, terutama dalam lingkungan dengan variasi pencahayaan dan ekspresi wajah yang berbeda (Haufe *et al.*, 2014; Wajah

et al., 2019; Alahmadi, Hussain and Aboalsamh, 2022). Dalam implementasinya, LDA digunakan sebagai metode ekstraksi fitur untuk mengurangi dimensi data gambar wajah. Setiap gambar wajah awalnya memiliki dimensi tinggi (ratusan hingga ribuan piksel), sehingga memerlukan pengurangan dimensi agar pemrosesan komputasi lebih efisien tanpa kehilangan informasi penting.

LDA memproyeksikan data wajah ke ruang berdimensi rendah yang memaksimalkan separasi antar kelas (wajah individu yang berbeda) sambil meminimalkan variabilitas dalam kelas (wajah dari individu yang sama). Setelah penerapan LDA, model pengenalan wajah dilatih menggunakan algoritma klasifikasi seperti Support Vector Machine (SVM). Hasil dari pendekatan ini menunjukkan peningkatan performa yang signifikan dibandingkan dengan model tanpa reduksi dimensi. Tingkat akurasi pengenalan wajah meningkat dalam skenario dengan variasi pencahayaan dan ekspresi yang berbeda. Dampak penerapan LDA sangat terlihat dalam efisiensi komputasi dan ketepatan klasifikasi.

Dengan pengurangan dimensi, waktu komputasi berkurang secara signifikan, memungkinkan sistem untuk memproses data lebih cepatLDA meminimalkan

overfitting dengan menghilangkan fitur yang tidak relevan, sehingga model menjadi lebih generalisasi dalam menghadapi data baru. Implementasi LDA dalam proyek ini tidak hanya meningkatkan kecepatan sistem, tetapi juga meningkatkan akurasi, yang menjadikannya pilihan ideal untuk aplikasi kecerdasan komputasional berbasis klasifikasi.

6.6. Kesimpulan

Linear Discriminant Analysis (LDA) menawarkan manfaat signifikan dalam kecerdasan komputasional, terutama dalam pengurangan dimensi data dan klasifikasi yang lebih efektif. LDA memaksimalkan separasi antar kelas, membuatnya ideal untuk pengenalan pola, pengenalan wajah, dan diagnosis medis. Keuntungan utama LDA adalah kemampuannya untuk mengurangi kompleksitas komputasi sambil mempertahankan fitur yang relevan, yang mengarah pada peningkatan akurasi dan efisiensi model. Arah penelitian di masa depan mencakup integrasi LDA dengan metode kecerdasan komputasional lainnya seperti Deep Learning dan Support Vector Machine (SVM). Penelitian lebih lanjut dapat mengeksplorasi penggunaan LDA sebagai teknik pra-pemrosesan untuk meningkatkan performa model non-linear yang lebih

kompleks. Pengembangan variasi LDA yang lebih adaptif dalam menangani data non-linear akan menjadi fokus yang menarik, terutama di era data besar dan kecerdasan buatan yang terus berkembang.

BAB VII
INDEPENDENT COMPONENT ANALYSIS (*ICA*)

7.1. Konsep Dasar *Independent Component Analysis* (ICA)

Independent Component Analysis (ICA) adalah teknik statistik yang bertujuan untuk memisahkan sinyal atau data campuran menjadi komponen-komponen independen yang mendasarinya. ICA sangat berguna dalam situasi di mana sinyal yang diamati merupakan campuran linear dari beberapa sumber independen yang tidak diketahui. Contoh klasik dari aplikasi ICA adalah masalah "cocktail party," di mana sejumlah mikrofon menangkap campuran suara dari berbagai sumber, dan tugas ICA adalah memisahkan suara-suara tersebut menjadi sumber aslinya.

ICA berbeda dengan metode seperti Principal Component Analysis (PCA) karena tidak hanya mencari representasi tak berkorelasi dari data, tetapi juga memastikan bahwa komponen-komponen tersebut secara statistik independen satu sama lain. Independen dalam konteks ini berarti bahwa informasi tentang satu

komponen tidak memberikan informasi tentang komponen lainnya.

Prinsip Dasar dan Asumsi ICA:

a. Kemandirian (Independence): Sumber-sumber sinyal yang mendasari diasumsikan saling independen secara statistik.
b. Non-Gaussianity: ICA mengandalkan properti bahwa kombinasi linear dari dua atau lebih variabel acak non-Gaussian lebih mendekati distribusi Gaussian dibandingkan dengan variabel acak asalnya.
c. Campuran Linear: Diasumsikan bahwa data yang diamati adalah campuran linear dari sumber-sumber independen.
d. Jumlah Sumber: Jumlah sumber independen setidaknya sama dengan atau kurang dari jumlah sinyal yang diamati.

7.2. Algoritma *Independent Component Analysis* (*ICA*)

Independent Component Analysis (ICA) adalah teknik yang bertujuan untuk memisahkan campuran sinyal menjadi komponen-komponen independen. Dalam konteks ini, "independen" berarti bahwa komponen-komponen tersebut tidak memiliki

informasi statistik yang berlebihan satu sama lain. ICA digunakan dalam berbagai bidang, seperti pemrosesan sinyal, pengolahan citra, dan analisis data biomedis.

ICA beroperasi berdasarkan asumsi bahwa sinyal yang diamati adalah campuran linear dari beberapa sumber independen yang tidak diketahui. Tujuan ICA adalah menemukan matriks pemisah yang dapat mengurai sinyal campuran menjadi komponen-komponen yang mendekati sumber asli.

Algoritma *Independent Component Analysis* (ICA) adalah, sebagai berikut :

1. FastICA

   Algoritma FastICA adalah salah satu metode yang paling populer dan efisien untuk mengimplementasikan ICA. Dikembangkan oleh Aapo Hyvärinen dan Erkki Oja, FastICA menggunakan metode fixed-point iteration untuk menemukan komponen independen.

   - Prinsip Kerja: Algoritma ini memaksimalkan non-Gaussianity dari komponen dengan menggunakan fungsi kontrastif, seperti negentropy. Melalui serangkaian iterasi, FastICA menyesuaikan vektor bobot untuk menemukan proyeksi yang menghasilkan komponen independen.

- Keunggulan: Cepat dan dapat digunakan pada dataset besar. Selain itu, memiliki konvergensi yang baik dibandingkan metode lain.
- Langkah-Langkah:
    1. Whitening: Data yang diamati di-whitened untuk menghilangkan korelasi antar komponen.
    2. Optimasi Fungsi Kontrastif: Melalui iterasi fixed-point, vektor bobot disesuaikan untuk memaksimalkan non-Gaussianity.
    3. Pemutakhiran Bobot: Hasil akhir adalah komponen independen yang dipisahkan.

2. Infomax ICA

   Algoritma Infomax dikembangkan oleh Bell dan Sejnowski. Algoritma ini didasarkan pada prinsip memaksimalkan informasi keluaran dari jaringan saraf tiruan (neural network).

   - Prinsip Kerja: Infomax memaksimalkan entropi keluaran dari jaringan saraf yang dirancang untuk memetakan sinyal campuran ke komponen independen.

Dengan memaksimalkan entropi, algoritma ini memaksimalkan kemandirian komponen.

- Keunggulan: Cocok untuk pemrosesan sinyal dengan distribusi non-Gaussian.
- Langkah-Langkah:
    1. Pembelajaran Jaringan Saraf: Jaringan saraf tiruan dilatih untuk memetakan sinyal campuran ke komponen independen.
    2. Optimasi Entropi: Parameter jaringan diperbarui untuk memaksimalkan entropi keluaran.
    3. Pemutakhiran Bobot: Menghasilkan pemisahan komponen independen.

3. Maximum Likelihood Estimation (MLE)

   Pendekatan Maximum Likelihood Estimation dalam ICA berusaha menemukan komponen independen dengan memaksimalkan likelihood dari data yang diamati.

   - Prinsip Kerja: MLE menggunakan distribusi probabilitas untuk memperkirakan parameter model yang memaksimalkan kemungkinan menghasilkan data yang diamati. Dengan cara ini, sumber-sumber

independen dipisahkan berdasarkan model probabilistik.

- Keunggulan: Memberikan pendekatan yang solid secara statistik dan dapat digunakan dalam berbagai aplikasi, termasuk yang melibatkan noise.
- Langkah-Langkah:
    1. Pemodelan Distribusi: Membuat model distribusi probabilitas untuk data.
    2. Estimasi Parameter: Menggunakan data yang diamati untuk memperkirakan parameter model yang memaksimalkan likelihood.
    3. Pemutakhiran Bobot: Menghasilkan pemisahan sumber-sumber independen.

4. Algoritma Negentropy

   Algoritma ini beroperasi berdasarkan konsep negentropy, yaitu ukuran seberapa jauh distribusi suatu variabel acak dari distribusi Gaussian.

   - Prinsip Kerja: Dengan memaksimalkan negentropy, ICA berusaha memisahkan komponen yang paling non-Gaussian.

Algoritma ini sering digunakan dalam kombinasi dengan metode seperti FastICA.

- Keunggulan: Efektif dalam memisahkan komponen non-Gaussian.
- Langkah-Langkah:
    1. Pengukuran Negentropy: Mengukur negentropy dari komponen data yang diamati.
    2. Optimasi Fungsi: Melalui iterasi, mencari proyeksi yang memaksimalkan negentropy.
    3. Pemutakhiran Bobot: Menghasilkan komponen independen yang terpisah.

Algoritma ICA, seperti FastICA, Infomax, dan MLE, memberikan metode yang efektif untuk memisahkan komponen independen dari sinyal campuran. Setiap algoritma memiliki kelebihan dan kekurangannya sendiri, tergantung pada sifat data dan tujuan analisis. Pemilihan algoritma yang tepat dan pemahaman mendalam tentang prinsip ICA sangat penting untuk keberhasilan implementasinya.

7.3. Implementasi ICA dalam Kecerdasan Komputasional

Independent Component Analysis (ICA) telah menjadi alat penting dalam kecerdasan komputasional, terutama dalam tugas-tugas yang melibatkan pemisahan sumber sinyal dan ekstraksi fitur. Implementasi ICA dalam konteks ini melibatkan pemisahan data kompleks menjadi komponen independen yang dapat digunakan untuk analisis lebih lanjut, pengenalan pola, atau sebagai input untuk algoritma pembelajaran mesin.

Langkah-Langkah Implementasi ICA :

1. Preprocessing Data

   Normalisasi: Data sering kali dinormalisasi untuk memastikan bahwa setiap variabel memiliki skala yang sama.

   Pengurangan Dimensi: Metode seperti Principal Component Analysis (PCA) dapat digunakan untuk mengurangi dimensi data, mempermudah pemisahan komponen oleh ICA.

2. Penerapan Algoritma ICA

   Algoritma ICA seperti FastICA atau Infomax digunakan untuk memisahkan data campuran menjadi komponen independen.

Algoritma ini memaksimalkan independensi statistik antara komponen dengan mengoptimalkan fungsi objektif seperti negentropy atau kurtosis.

3. Pemilihan dan Evaluasi Komponen

   Komponen-komponen independen yang dihasilkan dianalisis untuk menentukan relevansi dan interpretabilitasnya dalam konteks aplikasi.

   Dalam kecerdasan komputasional, komponen ini dapat digunakan sebagai fitur dalam model pembelajaran mesin atau sebagai sinyal bersih dalam pemrosesan sinyal.

4. Validasi dan Interpretasi Hasil

   Setelah pemisahan, penting untuk memvalidasi apakah komponen yang dihasilkan benar-benar independen dan memiliki makna.

   Dalam beberapa kasus, komponen-komponen ini digunakan untuk rekonstruksi sinyal asli atau untuk analisis lanjutan.

Aplikasi ICA dalam Kecerdasan Komputasional:

a. Pemrosesan Sinyal dan Pengenalan Pola: ICA digunakan untuk memisahkan sinyal yang tumpang tindih, seperti dalam analisis EEG

untuk mengidentifikasi pola aktivitas otak yang spesifik.

b. Pengurangan Dimensi dalam Pembelajaran Mesin: ICA digunakan untuk mengekstraksi fitur dari data multivariat, membantu meningkatkan kinerja model pembelajaran mesin.
c. Pengolahan Citra: ICA dapat digunakan untuk menghilangkan artefak atau noise dari gambar, serta untuk ekstraksi fitur dalam pengenalan objek.
d. Analisis Data Medis dan Biomedis: Dalam analisis data seperti EEG dan fMRI, ICA membantu memisahkan sinyal otak dari noise dan artefak lainnya, memungkinkan analisis lebih dalam terhadap aktivitas otak.

7.4. Keterbatasan dan Tantangan *Independent Component Analysis* (ICA):

Meskipun *Independent Component Analysis* (ICA) merupakan metode yang kuat untuk pemisahan sinyal dan ekstraksi fitur, terdapat beberapa keterbatasan dan tantangan yang perlu diperhatikan saat menerapkannya dalam analisis data dan kecerdasan komputasional.

1. Asumsi Kemandirian yang Kuat

- Kemandirian Statistik: ICA mengasumsikan bahwa sumber-sumber yang ingin dipisahkan benar-benar independen secara statistik. Dalam praktiknya, asumsi ini tidak selalu terpenuhi karena sumber-sumber mungkin memiliki tingkat ketergantungan tertentu.
- Kesulitan dalam Kasus Non-Independen: Jika sumber-sumber tidak sepenuhnya independen, hasil pemisahan oleh ICA mungkin tidak akurat, yang dapat mengurangi keandalan interpretasi hasil.

2. Kesulitan dalam Identifikasi Unik
    - Skala dan Urutan Tidak Diketahui: ICA hanya dapat mengidentifikasi komponen independen hingga skala dan urutan yang tidak diketahui. Artinya, komponen yang dihasilkan bisa dalam urutan acak dan memiliki skala yang berbeda dari sumber asli.
    - Masalah Ambiguitas: Tanpa informasi tambahan, sulit menentukan urutan dan orientasi komponen yang dihasilkan oleh ICA.

3. Sensitivitas terhadap Parameter dan Kondisi Awal
    - Kondisi Awal: Algoritma ICA seringkali sensitif terhadap kondisi awal dan parameter yang digunakan, yang dapat mempengaruhi konvergensi dan hasil akhir.
    - Parameter Algoritma: Pemilihan fungsi objektif dan parameter seperti jumlah iterasi dan kecepatan pembelajaran dapat mempengaruhi performa algoritma ICA.
    - Konvergensi ke Solusi Lokal: Algoritma optimisasi yang digunakan dalam ICA, seperti FastICA, dapat mengalami konvergensi ke solusi lokal, bukan solusi global, terutama jika kondisi awal tidak dipilih dengan baik.
4. Kesulitan dengan Data Berdimensi Tinggi
    - Kompleksitas Komputasi: Saat bekerja dengan data berdimensi tinggi, seperti dalam pemrosesan citra atau data genomik, kompleksitas komputasi ICA dapat menjadi sangat tinggi.
    - Pengurangan Dimensi yang Diperlukan: Untuk mengurangi beban komputasi, seringkali diperlukan langkah pengurangan

dimensi sebelum menerapkan ICA, yang dapat menyebabkan hilangnya informasi penting.

5. Asumsi Campuran Linear
    - Pembatasan pada Model Linear: ICA mengasumsikan bahwa data yang diamati adalah campuran linear dari sumber independen. Dalam banyak kasus nyata, campuran bisa bersifat non-linear, yang membatasi efektivitas ICA dalam memisahkan komponen.
    - Keterbatasan dalam Kasus Non-Linear: Untuk situasi di mana hubungan antara sumber dan sinyal yang diamati bersifat non-linear, ICA mungkin tidak memberikan hasil yang memuaskan.
6. Keterbatasan dalam Pemisahan Sinyal
    - Jumlah Komponen yang Terbatas: ICA memerlukan jumlah komponen yang dipisahkan (sinyal asli) setidaknya sama dengan jumlah sinyal campuran yang diamati. Jika sumber lebih banyak daripada sinyal yang diamati, ICA tidak dapat memisahkan semua sumber tersebut.

- Noise dan Artefak: ICA juga sensitif terhadap noise dan artefak dalam data. Keberadaan noise yang signifikan dapat mengganggu proses pemisahan komponen dan mengurangi kualitas hasil.

7. Validasi dan Interpretasi Hasil
    - Evaluasi Kualitas Komponen: Menilai kualitas dan relevansi komponen independen yang dihasilkan oleh ICA bisa jadi sulit. Tidak selalu jelas komponen mana yang bermakna atau sesuai dengan fenomena nyata.
    - Keterbatasan Interpretabilitas: Dalam beberapa kasus, komponen yang dihasilkan mungkin tidak memiliki interpretasi yang jelas, terutama jika data bersifat kompleks dan multidimensi.

ICA adalah alat yang ampuh untuk pemisahan sinyal dan ekstraksi fitur, tetapi keberhasilan penggunaannya sangat bergantung pada pemahaman terhadap asumsi dan keterbatasan yang ada. Pengguna harus berhati-hati dalam menginterpretasikan hasil dan mempertimbangkan alternatif atau metode tambahan jika data tidak sepenuhnya memenuhi asumsi.

BAB VIII
METODE CLUSTERING

8.1. Teori Dasar *Clustering*

1. Definisi dan Konsep Dasar *Clustering* adalah metode untuk mengelompokkan data ke dalam kelompok atau cluster sehingga objek dalam satu cluster lebih mirip satu sama lain dibandingkan dengan objek di cluster lain. Tujuan utama dari *Clustering* adalah untuk menemukan struktur yang tersembunyi dalam data.
2. Kriteria Evaluasi *Clustering* Evaluasi *Clustering* penting untuk menilai kualitas hasil *Clustering*. Beberapa kriteria evaluasi yang umum digunakan meliputi:
    - Internal Evaluation: Mengukur kualitas *Clustering* berdasarkan informasi dari data yang sudah dikelompokkan. Contoh metode internal adalah Silhouette Score, Davies-Bouldin Index, dan Dunn Index.
    - External Evaluation: Mengukur seberapa baik *Clustering* yang dihasilkan sesuai

dengan label yang sudah diketahui sebelumnya. Contoh metode eksternal adalah Purity, Normalized Mutual Information (NMI), dan Rand Index.

3. Jenis-Jenis *Clustering*
    - *Clustering* Berbasis Centroid: Mengelompokkan data berdasarkan titik pusat atau centroid. Contoh metode: K-Means, K-Medoids.
    - *Clustering* Hierarkis: Membentuk hierarki cluster berdasarkan hubungan antar data. Ada dua jenis utama: Agglomerative (penggabungan) dan Divisive (pembagian).
    - *Clustering* Berbasis Model: Menggunakan model probabilistik untuk menentukan cluster. Contoh metode: Gaussian Mixture Models (GMM).
    - *Clustering* Berbasis Density: Mengelompokkan data berdasarkan kepadatan data dalam ruang fitur. Contoh metode: DBSCAN (Density-Based Spatial *Clustering* of Applications with Noise).
    - *Clustering* Berbasis Jaringan: Menggunakan struktur jaringan atau graf untuk

Clustering. Contoh metode: Self-Organizing Maps (SOM).

8.2. Metode *Clustering*

Metode *Clustering* digunakan untuk mengelompokkan data tanpa label menjadi beberapa kelompok berdasarkan karakteristik atau kemiripan di antara data tersebut. Ada berbagai pendekatan *Clustering*, yang dibedakan berdasarkan cara mereka menemukan dan mengelompokkan data. Berikut adalah beberapa metode *Clustering* utama:

1. *Clustering* Berbasis Centroid

   Metode ini membagi data ke dalam cluster yang diwakili oleh centroid (titik pusat). Setiap data akan dimasukkan ke dalam cluster yang memiliki centroid terdekat.

   - K-Means: Algoritma yang membagi data ke dalam k cluster, di mana setiap cluster diwakili oleh centroid (rata-rata data dalam cluster). Proses ini dilakukan dengan langkah iteratif: inisialisasi centroid, alokasi data ke centroid terdekat, dan pembaruan centroid berdasarkan data yang dialokasikan.

- K-Medoids: Mirip dengan K-Means, tetapi menggunakan medoid (data aktual sebagai pusat cluster), yang lebih tahan terhadap data outlier.
- Fuzzy C-Means: Setiap data memiliki derajat keanggotaan terhadap setiap cluster, yang memungkinkan data menjadi bagian dari lebih dari satu cluster secara parsial.

2. *Clustering* Hierarkis

   Clustering hierarkis menghasilkan pohon dendrogram yang menunjukkan hubungan hirarkis antara data, di mana cluster terbentuk melalui penggabungan atau pembagian secara bertahap.

   - Hierarchical Agglomerative *Clustering* (HAC): Dimulai dengan setiap data sebagai cluster individual dan menggabungkannya berdasarkan kedekatan hingga semua data membentuk satu cluster besar. Pengukuran jarak antara cluster bisa menggunakan metode single linkage, complete linkage, atau average linkage.
   - Divisive *Clustering*: Kebalikan dari HAC, metode ini memulai dengan satu cluster

yang berisi semua data dan membaginya secara bertahap menjadi cluster yang lebih kecil.

3. *Clustering* Berbasis Model

   Metode ini mengasumsikan bahwa data berasal dari model probabilistik tertentu dan bertujuan untuk menemukan parameter model tersebut.

   - Gaussian Mixture Models (GMM): Mengasumsikan bahwa data berasal dari campuran distribusi Gaussian. GMM menggunakan Expectation-Maximization (EM) untuk mengoptimalkan parameter dari distribusi Gaussian dan menentukan cluster.
   - Hidden Markov Models (HMM): Sering digunakan dalam pengolahan sinyal dan data sekuensial. HMM mengelompokkan data berdasarkan probabilitas transisi antarstate tersembunyi dalam data.

4. *Clustering* Berbasis Density

   Metode ini menemukan cluster dengan mendeteksi area dengan kepadatan data yang tinggi.

   - DBSCAN (Density-Based Spatial *Clustering* of Applications with Noise): Algoritma ini

membentuk cluster berdasarkan kepadatan lokal dari data. DBSCAN juga mampu mengidentifikasi noise (data yang tidak masuk dalam cluster manapun).

- OPTICS (Ordering Points To Identify the *Clustering* Structure): Mirip dengan DBSCAN, tetapi dapat menangani cluster dengan kepadatan yang berbeda-beda.

5. *Clustering* Berbasis Jaringan

   Metode ini menggunakan konsep jaringan atau graf untuk menemukan cluster.

   - Self-Organizing Maps (SOM): Peta topologis yang digunakan untuk memvisualisasikan data berdimensi tinggi dengan memproyeksikannya ke dimensi yang lebih rendah, sambil mempertahankan struktur hubungan antar data.
   - ART (Adaptive Resonance Theory): Model *Clustering* yang mampu menyesuaikan diri dengan perubahan data tanpa perlu mengulang pembelajaran dari awal, cocok untuk data sekuensial.

8.3. Metode *Clustering* Lanjutan

Metode *Clustering* lanjutan berfokus pada pengembangan algoritma yang lebih kompleks atau spesifik untuk menangani data yang lebih rumit, data berukuran besar, atau aplikasi khusus. Beberapa metode lanjutan ini didasarkan pada optimasi, teknik pembelajaran mesin modern, atau dirancang untuk skenario data khusus seperti data sekuensial atau bertingkat. Berikut adalah penjelasan beberapa metode *Clustering* lanjutan:

1. *Clustering* dengan Algoritma Metaheuristik

   Metaheuristik adalah pendekatan optimasi global yang sering digunakan untuk mencari solusi *Clustering* optimal di ruang solusi yang besar dan kompleks. *Clustering* dengan algoritma ini biasanya digunakan untuk mengatasi keterbatasan algoritma *Clustering* konvensional.

   - Genetic Algorithm (GA): Metode ini terinspirasi dari teori evolusi biologis dan digunakan untuk mengoptimalkan pembagian data ke dalam cluster. Individu dalam populasi adalah solusi *Clustering*, yang dikombinasikan dan dimutasi untuk mencari solusi optimal.

- Simulated Annealing: Proses optimasi ini meniru proses pendinginan logam, di mana solusi *Clustering* diperbaiki secara bertahap, dan perubahan acak (mutasi) diberikan kesempatan untuk menghindari solusi lokal.
- Particle Swarm Optimization (PSO): Terinspirasi dari perilaku sosial kawanan burung atau ikan, metode ini digunakan untuk mencari pusat cluster secara global dengan iterasi antar partikel yang bergerak di ruang solusi.

2. *Clustering* pada Data Besar (Big Data)

   Clustering data berukuran besar membutuhkan pendekatan khusus karena metode konvensional tidak efisien dalam menangani jumlah data yang besar atau dimensi yang tinggi. *Clustering* Big Data sering kali memanfaatkan teknik distribusi dan parallel *Computing* untuk mengurangi waktu komputasi.

   - MapReduce K-Means: Algoritma K-Means yang diadaptasi untuk platform MapReduce yang mendistribusikan proses komputasi ke beberapa node dalam cluster komputasi,

seperti Hadoop, guna mengelola data dalam skala besar.

- Mini-Batch K-Means: Versi K-Means yang lebih efisien, memproses batch kecil data pada setiap iterasi daripada keseluruhan dataset, sehingga cocok untuk dataset besar.
- *Clustering* dengan Spark: Apache Spark menyediakan platform untuk melakukan *Clustering* dalam skala besar, dengan algoritma seperti K-Means yang dioptimalkan untuk lingkungan distribusi.

3. *Clustering* dalam Data Bertingkat (Hierarchical *Clustering* Lanjutan)

   Pada *Clustering* hierarkis lanjutan, metode ini dikembangkan lebih jauh untuk menangani data dengan struktur bertingkat atau multi-level. Pengembangan terbaru pada *Clustering* hierarkis melibatkan:

   - BIRCH (Balanced Iterative Reducing and *Clustering* using Hierarchies): Algoritma yang mendukung *Clustering* untuk data besar dengan menggunakan struktur hierarkis dan mengurangi data melalui *Clustering* fase awal.

- CURE (*Clustering* Using Representatives): Algoritma ini memperluas *Clustering* hierarkis dengan memilih beberapa titik representatif untuk mendeskripsikan setiap cluster, sehingga lebih tahan terhadap outlier.

4. *Clustering* Berbasis Spektral

   Metode spektral memanfaatkan informasi spektrum (nilai eigen) dari matriks kesamaan data untuk melakukan *Clustering*. Pendekatan ini berguna ketika struktur cluster tidak berbentuk globular atau linier seperti pada K-Means.

   - Spectral *Clustering*: Berdasarkan analisis nilai eigen dari matriks kedekatan, metode ini memetakan data ke dalam ruang baru yang lebih mudah untuk dikelompokkan. *Clustering* spektral efektif untuk memisahkan data yang tidak dapat dipisahkan secara linier.

5. *Clustering* Berdasarkan Pembelajaran Mendalam (Deep Learning)

   Dalam beberapa tahun terakhir, metode deep learning digunakan untuk meningkatkan hasil

Clustering, terutama untuk data yang kompleks dan berdimensi tinggi.

- Deep Embedded *Clustering* (DEC): Kombinasi autoencoder dengan K-Means untuk melakukan *Clustering* pada representasi laten data yang didapatkan dari jaringan saraf dalam.
- Variational Autoencoder (VAE) *Clustering*: Menggunakan VAE untuk mempelajari distribusi laten data dan melakukan *Clustering* pada representasi tersembunyi tersebut.

6. *Clustering* Ensemble

   Clustering ensemble adalah metode yang menggabungkan hasil dari beberapa algoritma *Clustering* untuk mendapatkan solusi yang lebih stabil dan akurat. Teknik ini membantu meningkatkan konsistensi hasil *Clustering*.

   - Cluster Ensembles: Menggabungkan beberapa hasil *Clustering* dengan menggunakan metode seperti voting atau optimasi untuk mencapai solusi yang lebih baik.

8.4. Aplikasi *Clustering*

Clustering adalah teknik yang sangat penting dan digunakan secara luas dalam berbagai bidang. Dengan mengelompokkan data menjadi beberapa cluster berdasarkan kemiripan atau kedekatan, metode ini membantu mengidentifikasi pola, struktur tersembunyi, atau hubungan dalam data yang sebelumnya tidak diketahui. Berikut adalah beberapa aplikasi penting dari metode *Clustering*:

1. *Clustering* dalam Pengolahan Citra

   Clustering digunakan untuk pengelompokan dan pengolahan citra, seperti segmentasi citra dan pengenalan objek.

   - Segmentasi Citra: *Clustering* membagi citra ke dalam beberapa segmen berdasarkan warna, tekstur, atau fitur lain. Metode seperti K-Means atau *Clustering* berbasis spektral sering digunakan untuk memisahkan bagian citra yang berbeda (misalnya, langit dan tanah pada foto luar ruangan).
   - Pengenalan Pola: *Clustering* digunakan dalam pengenalan pola untuk mengidentifikasi objek dalam citra. Misalnya, *Clustering* hierarkis dapat

mengidentifikasi pola yang lebih kompleks dalam data citra medis atau satelit.

2. *Clustering* dalam Analisis Teks

   Dalam pengolahan teks, *Clustering* sering digunakan untuk analisis dan pengelompokan dokumen.

   - Pengelompokan Dokumen: *Clustering* dapat digunakan untuk mengelompokkan dokumen berdasarkan topik yang mirip. Misalnya, algoritma seperti K-Means atau Latent Dirichlet Allocation (LDA) dapat mengelompokkan artikel berita atau penelitian akademik berdasarkan kata-kata yang sering muncul.
   - Penyaringan Informasi: Dalam mesin pencari atau platform berbasis informasi, *Clustering* membantu menyaring dan mengelompokkan konten yang relevan untuk pengguna berdasarkan preferensi dan pola pencarian.

3. *Clustering* dalam Data Biologis

   Clustering digunakan secara luas dalam bioinformatika untuk mengelompokkan data genetik, sekuens DNA, dan ekspresi gen.

- Analisis Ekspresi Gen: *Clustering* diterapkan untuk mengelompokkan gen berdasarkan pola ekspresi gen pada kondisi yang berbeda. Hal ini membantu peneliti memahami regulasi gen dan mengidentifikasi gen yang terlibat dalam penyakit tertentu.
- *Clustering* Data Sekuens DNA: Metode *Clustering* seperti DBSCAN digunakan untuk menemukan pola dalam sekuens DNA yang dapat membantu dalam identifikasi spesies atau hubungan evolusi.

4. *Clustering* dalam Data Sosial Media

   Clustering digunakan untuk mengelompokkan pengguna, konten, atau interaksi pada platform media sosial.

   - Pengelompokan Pengguna: *Clustering* memungkinkan pengelompokan pengguna dengan preferensi atau perilaku yang mirip. Ini penting untuk personalisasi konten dan rekomendasi.
   - Analisis Komunitas: *Clustering* berbasis graf seperti algoritma Louvain dapat digunakan untuk menemukan komunitas dalam jaringan sosial, seperti pengguna yang

sering berinteraksi satu sama lain atau berbagi ketertarikan yang sama.

5. *Clustering* dalam Analisis Pasar

   Dalam bidang pemasaran, *Clustering* digunakan untuk mengelompokkan konsumen berdasarkan perilaku, preferensi, atau demografi.

   - Segmentasi Pasar: *Clustering* membantu memisahkan konsumen ke dalam segmen pasar yang berbeda untuk menargetkan iklan atau produk yang spesifik. Misalnya, K-Means digunakan untuk mengelompokkan pelanggan berdasarkan kebiasaan belanja atau preferensi produk.
   - Rekomendasi Produk: Metode *Clustering* digunakan untuk memberikan rekomendasi produk berdasarkan kesamaan dengan pembelian sebelumnya atau pola perilaku konsumen lain yang serupa.

6. *Clustering* dalam Data Keuangan

   Clustering diterapkan dalam analisis keuangan untuk menemukan pola dalam perilaku konsumen atau kinerja pasar.

   - Deteksi Penipuan: *Clustering* digunakan untuk mendeteksi transaksi yang

mencurigakan atau penipuan. Dengan mengidentifikasi cluster transaksi normal, transaksi yang menyimpang dapat ditandai sebagai anomali.

- Analisis Portofolio: *Clustering* membantu dalam mengelompokkan aset ke dalam kategori yang berbeda berdasarkan performa dan risiko, sehingga membantu manajer portofolio dalam pengambilan keputusan investasi.

BAB IX
METODE JARINGAN SARAF TIRUAN

9.1. Konsep Dasar Jaringan Saraf Tiruan

Jaringan Saraf Tiruan (Artificial Neural Networks, ANN) adalah model komputasi yang terinspirasi oleh struktur dan fungsi otak manusia. Mereka digunakan untuk memecahkan berbagai masalah dalam kecerdasan buatan (AI) dan pembelajaran mesin (machine learning). Konsep dasar jaringan saraf tiruan mencakup struktur, fungsi, dan proses pelatihan yang memungkinkan model ini untuk belajar dari data dan membuat prediksi.

1. Struktur Dasar Jaringan Saraf Tiruan

   Jaringan saraf tiruan terdiri dari beberapa komponen utama yang membentuk struktur modelnya:

   - Neuron: Unit dasar dari jaringan saraf tiruan, mirip dengan neuron biologis. Setiap neuron menerima sinyal dari neuron lain, mengolah informasi, dan mengirimkan sinyal ke neuron berikutnya.
   - Lapisan (Layer): Jaringan saraf tiruan dibagi menjadi beberapa lapisan:

- Lapisan Input: Menerima data mentah dari luar. Setiap neuron di lapisan ini merepresentasikan fitur dari data.
- Lapisan Tersembunyi (Hidden Layer): Terletak di antara lapisan input dan output. Neuron di lapisan ini memproses data dan mengekstraksi fitur-fitur penting. Jaringan dapat memiliki satu atau lebih lapisan tersembunyi.
- Lapisan Output: Menghasilkan output akhir dari jaringan. Jumlah neuron di lapisan ini bergantung pada jenis masalah yang dipecahkan (misalnya, jumlah kelas dalam klasifikasi).

2. Fungsi Aktivasi

   Fungsi aktivasi digunakan untuk menentukan output dari neuron. Fungsi ini memperkenalkan non-linearitas ke dalam model, memungkinkan jaringan saraf untuk mempelajari hubungan yang kompleks dalam data. Beberapa fungsi aktivasi umum meliputi:

- Sigmoid: Fungsi yang menghasilkan output dalam rentang (0, 1). Sering digunakan dalam lapisan output untuk masalah klasifikasi biner.

  $$\sigma(x) = \frac{1}{1 + e^{-x}}$$

- ReLU (Rectified Linear Unit): Fungsi aktivasi yang umum digunakan dalam lapisan tersembunyi. Menghasilkan output x jika x>0x > 0x>0, dan 0 jika x≤0x \leq 0x≤0.

  $$\text{ReLU}(x) = \max(0, x)$$

- Tanh (Hyperbolic Tangent): Fungsi yang menghasilkan output dalam rentang (-1, 1). Digunakan untuk mempercepat konvergensi pelatihan.

  $$\tanh(x) = \frac{e^{x} - e^{-x}}{e^{x} + e^{-x}}$$

3. Proses Pelatihan

   Pelatihan jaringan saraf tiruan melibatkan penyesuaian bobot (weights) dan bias (biases) untuk meminimalkan kesalahan (error) dalam prediksi. Proses ini melibatkan beberapa langkah utama:

   - Forward Propagation: Data input diproses melalui lapisan-lapisan jaringan untuk

menghasilkan output. Setiap neuron menghitung outputnya dengan menggunakan bobot, bias, dan fungsi aktivasi.

- Loss Function: Mengukur seberapa baik atau buruk prediksi model dibandingkan dengan nilai target yang sebenarnya. Contoh fungsi loss termasuk Mean Squared Error (MSE) untuk regresi dan Cross-Entropy Loss untuk klasifikasi.
- Backward Propagation: Algoritma ini menghitung gradien dari fungsi loss terhadap bobot dan bias dengan menggunakan aturan rantai (chain rule). Gradien ini digunakan untuk memperbarui bobot dan bias melalui metode optimasi.
- Optimasi: Metode seperti Gradient Descent digunakan untuk memperbarui bobot dan bias dengan cara yang meminimalkan fungsi loss. Variasi dari Gradient Descent termasuk Stochastic Gradient Descent (SGD) dan algoritma adaptif seperti Adam.

9.2. Metode Jaringan Saraf Tiruan

Metode Jaringan Saraf Tiruan (Artificial Neural Network Methods) merujuk pada berbagai pendekatan dan teknik yang digunakan untuk mengembangkan dan melatih model jaringan saraf tiruan. Jaringan saraf tiruan merupakan alat yang kuat dalam pembelajaran mesin, digunakan untuk memecahkan masalah yang melibatkan pola, pengenalan, dan prediksi. Berikut adalah penjelasan tentang beberapa metode utama dalam jaringan saraf tiruan.

1. Jaringan Saraf Feedforward (Feedforward Neural Networks)

   Jaringan saraf feedforward adalah jenis jaringan saraf yang paling sederhana dan terdiri dari lapisan-lapisan neuron yang terhubung secara satu arah:

   - Arsitektur: Terdiri dari tiga jenis lapisan: lapisan input, lapisan tersembunyi, dan lapisan output. Data bergerak dari lapisan input melalui lapisan tersembunyi hingga mencapai lapisan output tanpa loop.
   - Pelatihan: Menggunakan algoritma backpropagation untuk mengupdate bobot berdasarkan kesalahan prediksi. Proses ini melibatkan forward propagation untuk

menghitung output dan backward propagation untuk menghitung gradien.

- Aplikasi: Digunakan dalam berbagai aplikasi seperti klasifikasi, regresi, dan pengenalan pola.

2. Jaringan Saraf Konvolusi (Convolutional Neural Networks, CNN)

   Jaringan saraf konvolusi dirancang khusus untuk memproses data yang memiliki grid-like topology, seperti citra:

   - Arsitektur: Memanfaatkan lapisan konvolusi untuk mengekstraksi fitur dari data input. Lapisan konvolusi diikuti oleh lapisan pooling untuk mengurangi dimensi fitur dan mengurangi komputasi.
   - Fungsi: Menangkap pola lokal dalam citra dengan menggunakan filter konvolusi. Teknik ini memungkinkan jaringan untuk belajar fitur dari gambar seperti tepi, sudut, dan tekstur.
   - Aplikasi: Umumnya digunakan dalam pengolahan citra, pengenalan objek, dan visi komputer.

3. Jaringan Saraf Recurrent (Recurrent Neural Networks, RNN)

Jaringan saraf recurrent dirancang untuk menangani data sekuensial dengan mempertimbangkan urutan temporal:

- Arsitektur: Memiliki neuron yang terhubung dalam bentuk loop, memungkinkan informasi dari langkah sebelumnya untuk mempengaruhi langkah berikutnya.
- Variasi: Termasuk Long Short-Term Memory (LSTM) dan Gated Recurrent Units (GRU), yang dirancang untuk mengatasi masalah vanishing gradient dan menangkap ketergantungan jangka panjang dalam data sekuensial.
- Aplikasi: Digunakan dalam pemrosesan bahasa alami (NLP), prediksi deret waktu, dan analisis sekuens.

4. Jaringan Saraf Generatif (*Generative Neural Networks*)

   Jaringan saraf generatif digunakan untuk menghasilkan data baru yang mirip dengan data pelatihan:

   - Generative Adversarial Networks (GANs): Terdiri dari dua jaringan yang bersaing—generator dan discriminator. Generator menciptakan data palsu, sedangkan

discriminator berusaha membedakan antara data palsu dan asli.

- Variational Autoencoders (VAEs): Menggunakan pendekatan probabilistik untuk menghasilkan data baru dengan memodelkan distribusi data dan melakukan sampling dari distribusi tersebut.
- Aplikasi: Digunakan dalam pembuatan gambar, sintesis data, dan pengolahan audio.

5. Jaringan Saraf Deep Learning (Deep Learning Neural Networks)

   Deep learning melibatkan penggunaan jaringan saraf dengan banyak lapisan tersembunyi untuk menangkap representasi data yang lebih kompleks:

   - Arsitektur: Menggunakan banyak lapisan tersembunyi untuk meningkatkan kapasitas model dalam mempelajari fitur dan pola yang kompleks dari data.
   - Teknik Pelatihan: Menggunakan teknik optimasi dan regularisasi untuk mengatasi overfitting dan meningkatkan kinerja model.

- Aplikasi: Meliputi berbagai bidang seperti pengenalan suara, deteksi objek, dan analisis teks.

9.3. Teknik Pembelajaran dan Optimasi dalam Jaringan Saraf Tiruan

Teknik pembelajaran dan optimasi adalah aspek penting dalam pelatihan jaringan saraf tiruan. Mereka berfungsi untuk mengoptimalkan proses pelatihan, mengurangi kesalahan, dan meningkatkan kinerja model. Berikut adalah penjelasan tentang beberapa teknik utama dalam pembelajaran dan optimasi.

1. Gradient Descent

   Gradient Descent adalah algoritma optimasi dasar yang digunakan untuk meminimalkan fungsi loss dengan memperbarui bobot model berdasarkan gradien dari fungsi loss.

   - Proses:
     a. Forward Propagation: Menghitung output dari model dan fungsi loss berdasarkan prediksi dan label yang sebenarnya.
     b. Backward Propagation: Menghitung gradien dari fungsi loss terhadap bobot model.

c. Update Bobot: Memperbarui bobot dengan mengurangi gradien yang dihitung dari fungsi loss.

- Variasi:
    a. Batch Gradient Descent: Menggunakan seluruh dataset untuk menghitung gradien.
    b. Stochastic Gradient Descent (SGD): Menggunakan satu sampel data secara acak untuk menghitung gradien.
    c. Mini-Batch Gradient Descent: Menggunakan subset dari data (mini-batch) untuk menghitung gradien.

2. Adam (Adaptive Moment Estimation)

   Adam adalah algoritma optimasi yang menggabungkan keuntungan dari dua teknik optimasi, yaitu Momentum dan RMSprop.

   - Proses:
       a. Moment Estimation: Menghitung estimasi rata-rata dan kuadrat gradien (momentum) untuk setiap parameter.

b. Adaptif Laju Pembelajaran: Menyesuaikan laju pembelajaran untuk setiap parameter berdasarkan estimasi momentumnya.

- Keunggulan:
  a. Mempercepat konvergensi.
  b. Mengurangi kebutuhan untuk penyesuaian laju pembelajaran manual.
  c. Bekerja dengan baik pada berbagai jenis masalah dan dataset.

3. Regularisasi

   Regularisasi adalah teknik yang digunakan untuk mencegah overfitting dengan menambahkan penalti pada kompleksitas model.

   - Jenis-jenis Regularisasi:
     - L1 Regularization (Lasso): Menambahkan penalti proporsional terhadap nilai absolut bobot. Menghasilkan sparsitas (bobot nol) dalam model.

$$\mathrm{L1}(w) = \lambda \sum_i |w_i|$$

- L2 Regularization (Ridge): Menambahkan penalti proporsional terhadap kuadrat bobot. Mencegah bobot menjadi terlalu besar.

$$\mathrm{L2}(w) = \lambda \sum_i w_i^2$$

- Dropout: Secara acak menonaktifkan neuron selama pelatihan untuk mencegah ketergantungan yang terlalu kuat pada neuron tertentu.

4. Batch Normalization

   Batch Normalization adalah teknik untuk menormalkan aktivasi neuron di setiap lapisan untuk meningkatkan kecepatan pelatihan dan stabilitas.

   - Proses:
     a. Normalisasi Aktivasi: Menghitung rata-rata dan deviasi standar dari aktivasi dalam mini-batch.
     b. Penyesuaian Skala dan Pergeseran: Menggunakan parameter learnable untuk mengubah distribusi aktivasi setelah normalisasi.
   - Keunggulan:

a. Mengurangi masalah internal covariate shift.
b. Mempercepat pelatihan dan mengurangi kebutuhan untuk penyesuaian hyperparameter.

5. Early Stopping

   Early Stopping adalah teknik untuk menghentikan pelatihan model sebelum selesai berdasarkan kinerja pada data validasi.

   - Proses:
     a. Pengawasan Kinerja: Mengamati metrik kinerja pada data validasi selama pelatihan.
     b. Hentikan Pelatihan: Menghentikan pelatihan jika kinerja pada data validasi tidak membaik setelah sejumlah epoch.
   - Keunggulan:
     a. Mengurangi risiko overfitting.
     b. Menghemat waktu pelatihan.

6. Learning Rate Scheduling

   Learning Rate Scheduling adalah teknik untuk menyesuaikan laju pembelajaran selama pelatihan untuk mencapai konvergensi yang lebih baik.

- Metode:
    - Decay: Mengurangi laju pembelajaran secara bertahap selama pelatihan.
    - Cyclical Learning Rates: Mengubah laju pembelajaran dalam rentang tertentu selama pelatihan.
    - Warm-up: Memulai dengan laju pembelajaran rendah dan meningkatkannya seiring berjalannya waktu.
- Keunggulan:
    - Mempercepat konvergensi.
    - Membantu mencapai solusi yang lebih baik.

BAB X
MULTILAYER PERCEPTRON (*MLP*)

10.1. Definisi *Multilayer Perceptron* (MLP)

Multilayer Perceptron (MLP) adalah jenis jaringan saraf tiruan yang terdiri dari beberapa lapisan neuron, termasuk lapisan input, satu atau lebih lapisan tersembunyi, dan lapisan output. MLP menggunakan fungsi aktivasi non-linear, seperti ReLU atau sigmoid, untuk memungkinkan jaringan mempelajari hubungan kompleks dalam data. Proses pelatihan MLP melibatkan algoritma backpropagation, di mana kesalahan dari prediksi dihitung dan digunakan untuk mengupdate bobot melalui gradien descent. MLP efektif untuk berbagai tugas seperti klasifikasi dan regresi, tetapi memerlukan pemilihan dan tuning hyperparameter yang hati-hati untuk mencapai performa optimal. Keunggulan MLP terletak pada kemampuannya untuk menangkap representasi data yang mendalam, tetapi bisa rentan terhadap overfitting tanpa teknik regularisasi yang tepat.

10.2. Struktur Dasar *Multilayer Perceptron* (MLP)

Struktur dasar *Multilayer Perceptron* (MLP) terdiri dari beberapa komponen utama yang bekerja bersama untuk memproses dan menganalisis data:

1. Lapisan Input: Ini adalah lapisan pertama dalam MLP yang menerima data input. Setiap neuron dalam lapisan ini mewakili fitur dari data. Misalnya, dalam citra, neuron di lapisan input dapat mewakili pixel.
2. Lapisan Tersembunyi: Terdapat satu atau lebih lapisan tersembunyi di antara lapisan input dan output. Neuron di lapisan tersembunyi memproses informasi yang diterima dari lapisan input melalui fungsi aktivasi non-linear. Lapisan tersembunyi bertanggung jawab untuk mengekstraksi fitur dan pola kompleks dari data.
3. Lapisan Output: Lapisan terakhir yang menghasilkan output dari jaringan. Jumlah neuron di lapisan output bergantung pada jenis masalah yang dipecahkan. Dalam klasifikasi, neuron biasanya mewakili kelas yang mungkin; dalam regresi, neuron menghasilkan nilai kontinu.

4. Neuron dan Fungsi Aktivasi: Setiap neuron dalam MLP menggabungkan input yang diterima, dikalikan dengan bobot, dan ditambah dengan bias. Hasilnya diproses melalui fungsi aktivasi non-linear seperti ReLU atau sigmoid untuk menghasilkan output neuron.
5. Bobot dan Bias: Bobot menentukan kekuatan koneksi antara neuron, sementara bias membantu menggeser fungsi aktivasi. Kedua parameter ini dioptimalkan selama pelatihan untuk meminimalkan kesalahan prediksi.

10.3. Tuning dan Validasi Model *Multilayer Perceptron* (MLP)

Tuning dan validasi adalah langkah penting dalam *pengembangan* model MLP untuk memastikan bahwa model bekerja dengan baik dan tidak overfitting. Berikut adalah penjelasan mengenai proses ini:

1. Pemilihan Hyperparameter
    - Jumlah Lapisan Tersembunyi dan Neuron: Jumlah lapisan tersembunyi dan neuron dalam setiap lapisan adalah hyperparameter utama. Lebih banyak lapisan dan neuron dapat meningkatkan kapasitas model tetapi juga risiko overfitting. Tuning jumlah

lapisan dan neuron sering kali memerlukan eksperimen dan cross-validation.

- Laju Pembelajaran (Learning Rate): Menentukan seberapa besar perubahan bobot dalam setiap iterasi pelatihan. Laju pembelajaran yang terlalu tinggi dapat menyebabkan model tidak stabil, sementara laju pembelajaran yang terlalu rendah dapat membuat pelatihan sangat lambat.
- Fungsi Aktivasi: Fungsi aktivasi yang digunakan, seperti ReLU, sigmoid, atau tanh, mempengaruhi cara model memproses informasi. Pilihan fungsi aktivasi bisa mempengaruhi kinerja model.
- Ukuran Batch: Menentukan jumlah sampel yang digunakan untuk menghitung gradien sebelum melakukan update bobot. Ukuran batch yang lebih kecil dapat mempercepat pelatihan dan memperbaiki generalisasi, tetapi juga bisa meningkatkan variabilitas dalam pembaruan gradien.
- Epoch: Jumlah iterasi pelatihan yang dilakukan pada seluruh dataset. Terlalu banyak epoch dapat menyebabkan

overfitting, sedangkan terlalu sedikit epoch dapat menyebabkan underfitting.

- Regulasi: Teknik seperti dropout, L1, dan L2 regularization digunakan untuk mengurangi overfitting. Dropout secara acak mematikan neuron selama pelatihan, sedangkan L1 dan L2 regularization menambahkan penalti pada bobot yang besar.

2. Cross-Validation
    - K-Fold Cross-Validation: Data dibagi menjadi kkk subset (fold). Model dilatih pada k−1k-1k−1 subset dan diuji pada subset yang tersisa. Proses ini diulang kkk kali, dengan setiap subset menjadi data validasi satu kali. Cross-validation memberikan estimasi yang lebih andal dari kinerja model dibandingkan dengan pembagian data tunggal.
    - Stratified K-Fold: Versi dari k-fold cross-validation di mana pembagian dilakukan dengan mempertimbangkan distribusi label. Ini berguna untuk dataset yang tidak seimbang.
3. Early Stopping

- Teknik Early Stopping: Menghentikan pelatihan ketika kinerja pada data validasi tidak lagi membaik atau mulai memburuk. Ini mencegah overfitting dengan memastikan bahwa model tidak dilatih lebih lama dari yang diperlukan.
- Monitor Metode: Biasanya memantau metrik seperti fungsi loss atau akurasi pada data validasi selama pelatihan. Jika metrik tidak menunjukkan peningkatan dalam beberapa epoch, pelatihan dapat dihentikan lebih awal.

4. Evaluasi Kinerja Model
    - Metrik Evaluasi: Setelah model dilatih, kinerjanya dievaluasi menggunakan metrik seperti akurasi, presisi, recall, F1 score untuk klasifikasi, atau Mean Squared Error (MSE) untuk regresi. Metrik ini memberikan indikasi seberapa baik model melakukan tugas yang diinginkan.
    - Data Test: Model diuji pada data yang tidak terlihat sebelumnya (data test) untuk mengevaluasi generalisasi. Hasil dari data test memberi gambaran tentang bagaimana model akan berfungsi dalam situasi nyata.

5. Pengaturan Hyperparameter Lanjutan
    - Grid Search: Menguji berbagai kombinasi hyperparameter dalam ruang pencarian yang telah ditentukan untuk menemukan set parameter terbaik.
    - Random Search: Mengambil sampel secara acak dari ruang hyperparameter untuk menemukan kombinasi yang baik. Ini sering kali lebih efisien daripada grid search.
    - Bayesian Optimization: Menggunakan teknik probabilistik untuk mengoptimalkan hyperparameter dengan lebih efisien, terutama untuk ruang parameter yang besar dan kompleks.

10.4. Aplikasi *Multilayer Perceptron* (MLP)

Multilayer Perceptron (MLP) adalah jenis jaringan saraf tiruan yang sangat serbaguna dan digunakan dalam berbagai aplikasi di berbagai domain. Berikut adalah beberapa aplikasi utama MLP:

1. Klasifikasi

   MLP sering digunakan untuk masalah klasifikasi, di mana tujuan adalah untuk mengidentifikasi kategori atau label dari data input. Contoh aplikasi klasifikasi meliputi:

- Pengenalan Citra: Mengklasifikasikan gambar ke dalam kategori yang berbeda, seperti deteksi objek dalam foto, pengenalan wajah, atau klasifikasi jenis tanaman berdasarkan citra daun.
- Pengolahan Teks: Klasifikasi teks untuk analisis sentimen, identifikasi topik, atau deteksi spam dalam email.
- Diagnosis Medis: Mengklasifikasikan data medis untuk diagnosis penyakit, seperti mendiagnosis kanker berdasarkan citra medis atau hasil tes laboratorium.

2. Regresi

   MLP juga digunakan untuk masalah regresi, di mana tujuan adalah untuk memprediksi nilai kontinu. Contoh aplikasi regresi meliputi:

   - Prediksi Harga: Memprediksi harga rumah, harga saham, atau harga barang berdasarkan fitur-fitur terkait.
   - Peramalan Waktu: Mengestimasi nilai masa depan dalam deret waktu, seperti peramalan permintaan produk atau cuaca.

3. Pengenalan Pola

MLP dapat digunakan untuk mengenali pola dalam data. Contoh aplikasi pengenalan pola meliputi:

- Pengenalan Suara: Mengidentifikasi kata atau frasa dari sinyal suara untuk aplikasi pengenalan suara seperti asisten virtual atau sistem transkripsi otomatis.
- Pengenalan Handwriting: Menerjemahkan tulisan tangan ke dalam teks digital, seperti dalam sistem OCR (Optical Character Recognition) untuk dokumen yang dipindai.

4. Pengolahan Citra

   Dalam pengolahan citra, MLP dapat digunakan untuk berbagai tugas, termasuk:

   - Segmentasi Citra: Mengidentifikasi dan memisahkan area atau objek yang berbeda dalam citra, seperti memisahkan organ dalam gambar medis atau objek dalam foto.
   - Enhancement dan Filtering: Meningkatkan kualitas citra atau mengurangi noise dengan memproses citra untuk aplikasi seperti pengolahan video atau restorasi gambar.

5. Pengolahan Bahasa Alami (NLP)

   Dalam NLP, MLP digunakan untuk tugas-tugas seperti:

- Penerjemahan Mesin: Menerjemahkan teks dari satu bahasa ke bahasa lain.
- Analisis Sentimen: Mengidentifikasi dan mengklasifikasikan sentimen dalam teks, seperti menentukan apakah ulasan produk positif atau negatif.

6. Kontrol dan Prediksi

   MLP digunakan dalam sistem kontrol dan prediksi untuk berbagai aplikasi, seperti:

   - Otomatisasi Industri: Mengontrol proses industri atau mesin dengan memprediksi dan menyesuaikan parameter operasi untuk efisiensi maksimum.
   - Prediksi Anomali: Mengidentifikasi anomali atau kegagalan dalam sistem, seperti dalam pemantauan mesin atau deteksi penipuan.

7. Rekomendasi dan Personalisasi

   MLP dapat digunakan dalam sistem rekomendasi untuk memberikan saran yang relevan kepada pengguna, seperti:

   - Sistem Rekomendasi: Menyediakan rekomendasi produk atau konten berdasarkan preferensi pengguna, seperti dalam e-commerce atau platform streaming.

- Personalisasi Konten: Menyesuaikan konten yang ditampilkan kepada pengguna berdasarkan perilaku dan preferensi mereka.

10.5. Kelebihan dan Kekurangan *Multilayer Perceptron* (MLP)

Multilayer Perceptron (MLP) memiliki berbagai kelebihan dan kekurangan yang mempengaruhi aplikasinya dalam berbagai masalah pembelajaran mesin. Berikut adalah penjelasan mengenai kelebihan dan kekurangan MLP:

Kelebihan MLP

1. Kemampuan untuk Menangkap Pola Non-Linear

   MLP menggunakan fungsi aktivasi non-linear di lapisan tersembunyi, memungkinkan jaringan untuk memodelkan hubungan yang kompleks dan non-linear dalam data. Ini membuat MLP efektif untuk menangani berbagai masalah yang tidak dapat dipecahkan dengan model linear sederhana.
2. Fleksibilitas Arsitektur

   MLP dapat diubah untuk menyesuaikan berbagai jenis data dan tugas dengan mengubah jumlah lapisan tersembunyi, jumlah neuron di

setiap lapisan, dan jenis fungsi aktivasi. Fleksibilitas ini memungkinkan MLP untuk diterapkan pada berbagai masalah.

3. Kemampuan Belajar Representasi

   Dengan beberapa lapisan tersembunyi, MLP dapat belajar representasi hierarkis dari data. Ini berarti MLP dapat menangkap fitur tingkat tinggi yang berguna untuk tugas-tugas seperti klasifikasi dan regresi.

4. Penggunaan Fungsi Aktivasi Non-Linear

   Fungsi aktivasi seperti ReLU, sigmoid, atau tanh memungkinkan MLP untuk memperkenalkan non-linearitas dalam model, yang dapat meningkatkan kinerja pada berbagai aplikasi, termasuk pengenalan pola dan prediksi.

5. Komunitas dan Alat Pengembangan

   MLP didukung oleh berbagai alat dan pustaka perangkat lunak seperti TensorFlow, Keras, dan PyTorch, yang memudahkan implementasi dan pelatihan model MLP. Ada juga banyak sumber daya dan dokumentasi yang tersedia untuk membantu dalam pengembangan MLP.

10.6. Kekurangan MLP

1. Overfitting

MLP dapat mengalami overfitting, terutama jika model terlalu kompleks dengan banyak lapisan dan neuron, dan jika data pelatihan tidak cukup banyak. Overfitting terjadi ketika model belajar terlalu baik pada data pelatihan, tetapi tidak dapat menggeneralisasi dengan baik pada data baru.

2. Kebutuhan Komputasi yang Tinggi

   Pelatihan MLP, terutama dengan banyak lapisan dan neuron, memerlukan sumber daya komputasi yang signifikan dan waktu pelatihan yang lama. Ini bisa menjadi kendala, terutama pada dataset besar atau ketika menggunakan perangkat keras dengan kapasitas terbatas.

3. Kesulitan dalam Pemilihan Hyperparameter

   MLP memerlukan pemilihan dan tuning hyperparameter yang hati-hati, seperti jumlah lapisan tersembunyi, jumlah neuron per lapisan, dan laju pembelajaran. Menemukan kombinasi optimal dari hyperparameter sering kali memerlukan eksperimen yang ekstensif dan proses pencarian.

4. Kebutuhan Data yang Besar

   Untuk mencapai kinerja yang baik, MLP sering memerlukan jumlah data yang besar. Jika data

pelatihan tidak mencukupi, model mungkin tidak belajar dengan baik atau tidak memberikan hasil yang memadai.

5. Interpretabilitas Terbatas

   Model MLP, terutama dengan banyak lapisan tersembunyi, dapat menjadi "kotak hitam" yang sulit untuk diinterpretasikan. Ini berarti bahwa sulit untuk memahami bagaimana model membuat keputusan atau memprediksi hasil.

6. Training Instability

   Pelatihan MLP dapat menjadi tidak stabil, terutama jika laju pembelajaran tidak diatur dengan benar atau jika fungsi aktivasi menyebabkan masalah seperti vanishing gradients. Teknik seperti normalisasi batch dan inisialisasi bobot yang baik sering diperlukan untuk meningkatkan stabilitas pelatihan.

DAFTAR PUSTAKA

Abdi, H., & Williams, L. J. (2010) 'Principal Component Analysis' *Wiley Interdisciplinary Reviews: Computational Statistics, 2*(4), 433–459. https://doi.org/10.1002/wics.101.

Adler, N., & Golany, B. (2001) 'Evaluation of deregulated airline networks using data envelopment analysis combined with Principal Component Analysis with an application to Western Europe' *European Journal of Operational Research, 132*(2), 260–273. https://doi.org/10.1016/S0377-2217(00)00143-9.

Adu-ManuSarpong, K. and Kingsley Arthur, J. (2013) 'A Review of Data Cleansing Concepts Achievable Goals and Limitations', International Journal of Computer Applications, 76(7), pp. 19–22. Available at: https://doi.org/10.5120/13259-0737.

Aggarwal, C. C. (2018). Neural Networks and Deep Learning: A Textbook. *Springer*.

Alahmadi, A., Hussain, M. and Aboalsamh, H. (2022) 'LDA-CNN: Linear Discriminant Analysis Convolution Neural Network for Periocular Recognition in the Wild', Mathematics, 10(23). Available at: https://doi.org/10.3390/math10234604.

Alexander, T.A., Irizarry, R.A. and Bravo, H.C. (2023) 'Capturing discrete latent structures: choose LDs

over PCs', Biostatistics, 24(1), pp. 1–16. Available at: https://doi.org/10.1093/biostatistics/kxab030.

Alpaydin, E. (2014). Introduction to Machine Learning. MIT Press.

Aminudin, A. and Cahyono, E.B. (2019) 'Pengukuran Performa Apache Spark dengan Library H2O Menggunakan Benchmark Hibench Berbasis Cloud *Computing*', Jurnal Teknologi Informasi dan Ilmu Komputer, 6(5), p. 519. Available at: https://doi.org/10.25126/jtiik.2019651520.

Analytics Vidhya. (n.d.) 'Understanding PCA and Its Applications in Machine Learning', Retrieved from https://www.analyticsvidhya.com.

Andriansyah, D. (2022) 'Implementasi Extract-Transform-Load (ETL) Data Warehouse Laporan Harian Pool', Jurnal Teknik Informatika, 8(2), pp. 45–49. Available at: https://doi.org/10.51998/jti.v8i2.486.

Aslamiyah, S. (2022) Komputer Bisnis Menghadapi Era Digital, Penerbit Pustaka Aksara. Surabaya: Penerbit Pustaka Aksara. Available at: https://repository.pustakaaksara.co.id/media/publications/559130-komputer-bisnis-menghadapi-era-digital-76ad7b5f.pdf.

Barnova, K. et al. (2023) 'Implementation of artificial intelligence and machine learning-based methods in brain–computer interaction', Computers in Biology and Medicine, 163(March), p. 107135. Available at:

https://doi.org/10.1016/j.compbiomed.2023.107135.

Bishop, C. M. (2006). Pattern Recognition and Machine Learning. Springer.

Bro, R., & Smilde, A. K. (2014) 'Principal Component Analysis' *Analytical Methods, 6*(9), 2812–2831. https://doi.org/10.1039/C3AY41907J.

Chen, R.C. et al. (2020) 'Selecting critical features for data classification based on machine learning methods', Journal of Big Data, 7(1). Available at: https://doi.org/10.1186/s40537-020-00327-4.

Cichocki, A., & Amari, S. (2002). Adaptive Blind Signal and Image Processing: Learning Algorithms and Applications. John Wiley & Sons.

Dogra, V. et al. (2022) 'A Complete Process of Text Classification System Using State-of-the-Art NLP Models', Computational Intelligence and Neuroscience, 2022. Available at: https://doi.org/10.1155/2022/1883698.

Farag, A.A. and Elhabian, S.Y. (2008) 'A Tutorial on Data Reduction Linear Discriminant Analysis (LDA)', CVIP Lab, (September), p. 47. Available at: www.cvip.uofl.edu.

Farkhod, A. et al. (2021) 'Lda-based topic modeling sentiment analysis using topic/document/sentence (Tds) model', Applied Sciences (Switzerland), 11(23). Available at: https://doi.org/10.3390/app112311091.

Febtiawan, E.P. et al. (2024) 'Forecasting Produksi Energi Photovoltaic Menggunakan Algoritma Random Forest Classification', 5(4), pp. 1053–1062. Available at: https://doi.org/10.47065/josh.v5i4.5514.

Fitria, E.R. and Rozci, F. (2023) 'Penerapan Metode Regresi Least Absolute Shrinkage and Selection Operator (LASSO) dan Regresi Linier untuk Memprediksi Tingkat Kemiskinan di Indonesia', Jurnal Ilmiah Sosio Agribis, 22(2), p. 123. Available at: https://doi.org/10.30742/jisa22220222620.

Floares, A. (2012) 'Computational intelligence', Computational Intelligence, (January 2002), pp. 1–212. Available at: https://doi.org/10.19044/esj.2018.v14n21p56.

Froelicher, D., et al. (2023) 'Scalable and Privacy-Preserving Federated Principal Component Analysis', *ArXiv*. https://arxiv.org/abs/2304.04523.

Füller, J. et al. (2022) 'How AI revolutionizes innovation management – Perceptions and implementation preferences of AI-based innovators', Technological Forecasting and Social Change, 178(March), p. 121598. Available at: https://doi.org/10.1016/j.techfore.2022.121598.

Géron, A. (2019). *Hands-on machine learning with scikit-learn, keras, and tensorflow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media.

Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.

Graf, R., Zeldovich, M. and Friedrich, S. (2024) 'Comparing linear discriminant analysis and supervised learning algorithms for binary classification—A method comparison study', Biometrical Journal, 66(1), pp. 1–20. Available at: https://doi.org/10.1002/bimj.202200098.

Gupta, A. (2019) 'A Complete Reference for Informatica Power Center ETL Tool', International Journal of Trend in Scientific Research and Development, Volume-3(Issue-2), pp. 1063–1070. Available at: https://doi.org/10.31142/ijtsrd19045.

Han, J., Kamber, M., & Pei, J. (2011) *Data Mining: Concepts and Techniques* (3rd ed.), Morgan Kaufmann.

Hassabis, D., Silver, D., & Lee, S. (2017). "Mastering the game of Go without human knowledge". Nature, 550, 354-359.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). The Elements of Statistical Learning: Data Mining, Inference, and Prediction. Springer.

Haufe, S. et al. (2014) 'On the interpretation of weight vectors of linear models in multivariate neuroimaging', NeuroImage, 87, pp. 96–110. Available at: https://doi.org/10.1016/j.neuroimage.2013.10.067.

Heintz, N., Francart, T. and Bertrand, A. (2023) 'Minimally Informed Linear Discriminant Analysis: training an LDA model with unlabelled data', pp. 1–13. Available at: http://arxiv.org/abs/2310.11110.

Hermanto, K. et al. (2023) 'Penggunaan Python Untuk Menganalisis Pola Penyebaran Covid-19 Di Masa Pandemi', Journal of Student Development Information System (JoSDIS), 3(2), pp. 62–75.

Hidayat, R. et al. (2021) 'Penerapan Metode Pembelajaran Menggunakan Ekstraksi Fitur dan Algoritma Klasifikasi untuk Identifikasi Pengenalan Iris', Jurnal Teknik Komputer AMIK BSI, 7(2), pp. 151–157.

Hoffman, D.W. (2017) 'Implementasi Web Service pada Integrasi Data Akademik dengan Replika Pangkal Data Dikti', Telematika, 14(1), pp. 1–11.

Hyvärinen, A., Karhunen, J., & Oja, E. (2001). *Independent Component Analysis*. John Wiley & Sons.

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. (2016). *Deep Learning*. MIT Press.

Jolliffe, I. T. (2002) *Principal Component Analysis* (2nd ed.), Springer.

Jurafsky, D., & Martin, J. H. (2021). Speech and Language Processing. *Prentice Hall*.

Kak, A. (2023) 'Dimensionality Reduction with PCA , LDA , and Autoencoders — Constructing Optimal Subspaces', (August).

Kim, S.W. and Gil, J.M. (2019) ‘Research paper classification systems based on TF-IDF and LDA schemes’, Human-centric *Computing* and Information Sciences, 9(1). Available at: https://doi.org/10.1186/s13673-019-0192-7.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). "ImageNet Classification with Deep Convolutional Neural Networks". Advances in Neural Information Processing Systems, 25.

Lee, T. W. (1998). *Independent Component Analysis: Theory and Applications*. Kluwer Academic Publishers.

Maghfiroh, A., Findawati, Y. and Indahyanti, U. (2023) ‘Klasifikasi Penipuan pada Rekening Bank menggunakan Pendekatan Ensemble Learning’, Building of Informatics, Technology and Science (BITS), 4(4), pp. 1883–1891. Available at: https://doi.org/10.47065/bits.v4i4.3212.

Mandhala, V.N., Sujatha, V. and Devi, B.R. (2015) ‘Scene classification using support vector machines’, Proceedings of 2014 IEEE International Conference on Advanced Communication, Control and *Computing* Technologies, ICACCCT 2014, 63(3), pp. 1807–1810. Available at: https://doi.org/10.1109/ICACCCT.2014.7019421.

Manning, C. D., Raghavan, P., & Schütze, H. (2008). Introduction to Information Retrieval. *Cambridge University Press*.

Mardia, K. V. (1979) *Multivariate Analysis* Academic Press.

Mardia, K. V., Kent, J. T., & Bibby, J. M. (1979) *Multivariate Analysis* Academic Press.

MATLAB. (n.d.) 'Principal Component Analysis Documentation', Retrieved from https://mathworks.com.

Mirończuk, M.M. and Protasiewicz, J. (2018) 'A recent overview of the state-of-the-art elements of text classification', Expert Systems with Applications, 106, pp. 36–54. Available at: https://doi.org/10.1016/j.eswa.2018.03.058.

Mitchell, T. M. (1997). *Machine Learning*. McGraw-Hill.

Mulyati, S. et al. (2018) 'Normalisasi Database Dan Migrasi Database Untuk Memudahkan Manajemen Data', Sebatik, 22(2), pp. 124–129. Available at: https://doi.org/10.46984/sebatik.v22i2.319.

Murphy, K. P. (2012). Machine Learning: A Probabilistic Perspective. MIT Press.

Nasser, A.B. et al. (2024) 'Depth linear discrimination-oriented feature selection method based on adaptive sine cosine algorithm for software defect prediction', Expert Systems with Applications, 253(February), p. 124266. Available at: https://doi.org/10.1016/j.eswa.2024.124266.

Nielsen, M. A. (2015). *Neural Networks and Deep Learning: A Textbook*. Determination Press.

Nuraini, R. et al. (2023) 'Tomato Ripeness Detection Using Linear Discriminant Analysis Algorithm with CIELAB and HSV Color Spaces', Building of Informatics, Technology and Science (BITS), 5(2), pp. 523–531. Available at: https://doi.org/10.47065/bits.v5i2.4192.

Romero, J.A. et al. (2022) 'Linear discriminant analysis reveals hidden patterns in NMR chemical shifts of intrinsically disordered proteins', PLoS Computational Biology, 18(10), pp. 1–22. Available at: https://doi.org/10.1371/journal.pcbi.1010258.

Russell, S., & Norvig, P. (2016). Artificial Intelligence: A Modern Approach. Pearson.

scikit-learn. (n.d.) 'Principal Component Analysis (PCA)' Retrieved from https://scikit-learn.org.

Shlens, J. (2014) 'A Tutorial on Principal Component Analysis', *ArXiv*. https://arxiv.org/abs/1404.1100.

Singh, M., Pant, M., Kong, L., Alijani, Z., & Snasel, V. (2023) 'A PCA-based fuzzy tensor evaluation model for multi-criteria decision-making' *Axioms, 11*(3), 140. https://doi.org/10.3390/axioms11030140.

Sinulingga, S., Fatichah, C. and Yuniarti, A. (2016) 'Pengenalan Wajah Menggunakan Two Dimensional Linear Discriminant Analysis Berbasis Optimasi Feature Fusion Strategy', JATISI (Jurnal Teknik Informatika dan Sistem Informasi, 3(1), pp. 1–11. Available at: http://jurnal.mdp.ac.id/index.php/jatisi/article/view/59.

Sodhi, K.S. and Lal, M. (2013) 'Face Recognition Using PCA , LDA and Various Distance Classifiers', Journal of Global Research in Computer Science, 4(3), pp. 30–35.

Stone, J. V. (2004). *Independent Component Analysis: A Tutorial Introduction*. MIT Press.

Suadaa, L.H. and Purwarianti, A. (2016) 'Combination of Latent Dirichlet Allocation (LDA) and Term Frequency-Inverse Cluster Frequency (TFxICF) in Indonesian text *Clustering* with labeling', 2016 4th International Conference on Information and Communication Technology, ICoICT 2016 [Preprint]. Available at: https://doi.org/10.1109/ICoICT.2016.7571885.

Sun, A.Y. and Scanlon, B.R. (2019) 'How can Big Data and machine learning benefit environment and water management: A survey of methods, applications, and future directions', Environmental Research Letters, 14(7). Available at: https://doi.org/10.1088/1748-9326/ab1b7d.

Suriakin, M., Kanata, B., & Suta Wijaya, I. G. P. (2014) Ekstraksi Ciri Wajah Manusia Menggunakan Algoritma Principal Component Analysis (PCA) untuk Sistem Pengenalan Wajah, *Dielektrika, 1*(1), 16-23. ISSN 2086-9487.

Tang, L. et al. (2014) 'A new method combining LDA and PLS for dimension reduction', PLoS ONE, 9(5). Available at: https://doi.org/10.1371/journal.pone.0096944.

Tharwat, A. et al. (2017) 'Linear discriminant analysis: A detailed tutorial', AI Communications, 30(2), pp. 169–190. Available at: https://doi.org/10.3233/AIC-170729.

Towards Data Science. (n.d.) 'Principal Component Analysis (PCA) Explained', Retrieved from https://towardsdatascience.com.

Villegas-Ch, W., Gaibor-Naranjo, W. and Sanchez-Viteri, S. (2024) 'Application of Deep Learning Techniques for the Optimization of Industrial Processes Through the Fusion of Sensory Data', International Journal of Computational Intelligence Systems, 17(1). Available at: https://doi.org/10.1007/s44196-024-00596-4.

Vora, L.K. et al. (2023) Artificial Intelligence in Pharmaceutical Technology and Drug Delivery Design, Pharmaceutics. Available at: https://doi.org/10.3390/pharmaceutics15071916.

Wajah, D. et al. (2019) 'Face Detection Using Linear Discriminant Analysis (Lda) Method and Support Vector Machine (Svm)', JOINCS (Journal of Informatics, Network, and Computer Science), 1(2), pp. 1–4. Available at: https://doi.org/10.21070/joincs.v1i2.521.

Wang, Q. et al. (2017) 'Convolutional 2D LDA for nonlinear dimensionality reduction', IJCAI International Joint Conference on Artificial Intelligence, 0, pp. 2929–2935. Available at: https://doi.org/10.24963/ijcai.2017/408.

Widarti, E., Bahri, S. and Widyanto, A. (2018) 'Integrasi Sistem Informasi Akademik Terpadu (SIAKAT) Dengan Feeder Pddikti', JITU : Journal Informatic Technology And Communication, 2(3), pp. 10–18.

Yaqoob, A., Musheer Aziz, R. and verma, N.K. (2023) 'Applications and Techniques of Machine Learning in Cancer Classification: A Systematic Review', Human-Centric Intelligent Systems, 3(4), pp. 588–615. Available at: https://doi.org/10.1007/s44230-023-00041-3.

Zadeh, L. A. (1994). *Soft Computing* and Fuzzy Logic. World Scientific.

Zheng, F. (2022) 'Facial Expression Recognition Based on LDA Feature Space Optimization', Computational Intelligence and Neuroscience, 2022. Available at: https://doi.org/10.1155/2022/9521329.

Zheng, S. (2017) 'Machine Learning: Several Advances in Linear Discriminant Analysis, Analysis, Multi-View Regression and Support Vector Machine.

KECERDASAN KOMPUTASIONAL

Kecerdasan komputasional mencakup berbagai metode yang diilhami oleh proses biologis dan sosial dalam memecahkan masalah-masalah kompleks yang sulit diselesaikan dengan cara tradisional. Dengan berkembangnya teknologi, pendekatan ini telah diterapkan di berbagai bidang, seperti pengolahan data, optimisasi, pengambilan keputusan, hingga kecerdasan buatan.

Buku ini mengupas berbagai teknik dalam kecerdasan komputasional, seperti jaringan saraf tiruan, algoritma genetika, sistem fuzzy, dan swarm intelligence, yang dijelaskan secara rinci dengan contoh penerapan nyata di dunia industri dan penelitian. Penyusunan buku ini dilakukan dengan bahasa yang mudah dipahami, sehingga diharapkan dapat membantu mahasiswa, praktisi, dan peneliti untuk memahami konsep-konsep tersebut secara mendalam.

IKAPI
IKATAN PENERBIT INDONESIA
CV REY MEDIA GRAFIKA
EMAIL:
REYMEDIAGRAFIKA.RGM@GMAIL.COM
ISBN 978-623-8609-57-4