KOMPUTA : Jurnal Ilmiah Komputer dan Informatika
Vol. xx, No. x, mmmm yyyy, ISSN : 2089-9033, e- ISSN : 2715-7849
PAGE 4
DOI: 10.34010/komputa.vxxix.xxxx

Rangkuman Materi Perkuliahan Machine Learning Kelompok-8

Rahma Fitria Tunnisa1, Rahmawati2

[1,2] Sistem Informasi, Sekolah Tinggi Manajemen Informatika Komputer (STMIK) Tazkia
E-mail : 1 241572010009.tunnisa@student.stmik.tazkia.ac.ai
2 241572010012.rahmawati@student.stmik.tazkia.ac.ai

Abstrak

Perkembangan teknologi kecerdasan buatan, khususnya *Machine Learning* (ML), telah menjadi pilar utama dalam transformasi digital di berbagai sektor industri. Artikel ini menyajikan tinjauan komprehensif terhadap materi pembelajaran *Machine Learning* dari tingkat dasar hingga teknik optimasi tingkat lanjut. Fokus pembahasan mencakup paradigma belajar induktif, representasi data dalam memori, algoritma klasifikasi berbasis pohon (*Decision Tree*), hingga pemodelan regresi menggunakan pendekatan numerik. Melalui analisis algoritma ID3, C4.5, dan *Stochastic Gradient Descent* (SGD), artikel ini mendemonstrasikan bagaimana data mentah diproses menjadi pengetahuan terstruktur melalui model matematis. Hasil tinjauan ini menunjukkan bahwa pemahaman mendalam terhadap struktur data dan teknik optimasi merupakan kunci utama dalam membangun model ML yang efisien dan akurat. Implementasi menggunakan bahasa Python dengan dukungan pustaka NumPy dan Pandas memperkuat fleksibilitas dalam eksperimentasi model.

Kata kunci: *Machine Learning*, *Decision Tree*, ID3, *Linear Regression*, *Stochastic Gradient Descent*.

Summary of Machine Learning Lecture Material for Group 8

Abstract

The development of artificial intelligence technology, particularly machine learning (ML), has become a key pillar of digital transformation across various industrial sectors. This article presents a comprehensive overview of machine learning materials, from basic to advanced optimization techniques. The discussion focuses on inductive learning paradigms, in-memory data representation, tree-based classification algorithms (Decision Trees), and regression modeling using numerical approaches. Through an analysis of the ID3, C4.5, and Stochastic Gradient Descent (SGD) algorithms, this article demonstrates how raw data is processed into structured knowledge through mathematical models. The review demonstrates that a thorough understanding of data structures and optimization techniques is key to building efficient and accurate ML models. The implementation uses Python with support for the NumPy and Pandas libraries, enhancing flexibility in model experimentation.

Keywords: Machine Learning, Decision Tree, ID3, Linear Regression, Stochastic Gradient Descent.

- **Pendahuluan**

Naskah dikirim: dd-mm-yyyy; direvisi dd-mm-yyyy; diterima: dd-mm-yyyy

Machine Learning merupakan disiplin ilmu yang memungkinkan sistem komputer untuk belajar dari data (*experience*) guna menyelesaikan tugas tertentu (*task*) tanpa instruksi eksplisit, dengan performa yang terukur (*performance measure*). Di era data raya (*big data*), kemampuan untuk mengekstraksi pola tersembunyi dari dataset yang kompleks menjadi kompetensi yang sangat krusial bagi praktisi teknologi informasi.

Penulisan jurnal ini bertujuan untuk mendokumentasikan dan merangkum konsep-konsep fundamental yang dipelajari selama empat belas pertemuan perkuliahan. Cakupan materi dimulai dari pengenalan paradigma belajar, di mana *Inductive Learning* menjadi landasan dalam pembentukan hipotesis berdasarkan observasi data spesifik. Selanjutnya, pembahasan diarahkan pada representasi data dalam memori komputer, yang sangat menentukan efisiensi algoritma, terutama pada struktur data pohon (*tree*).

Selain aspek teoritis, artikel ini juga membahas implementasi teknis menggunakan ekosistem Python. Pembahasan mengenai algoritma klasifikasi seperti pohon keputusan (ID3 dan C4.5) serta algoritma regresi linear memberikan gambaran mengenai evolusi solusi ML, mulai dari logika berbasis aturan (*rule-based*) hingga optimasi berbasis kalkulus melalui *Stochastic Gradient Descent*. Dengan memahami alur dari *data preprocessing* hingga evaluasi model, diharapkan pembaca mendapatkan peta jalan yang jelas dalam pengembangan aplikasi berbasis kecerdasan buatan.

● Metodologi

2.1 Metode Pembelajaran dan Pendekatan Studi

Metodologi yang digunakan dalam penulisan artikel ini mengacu pada pendekatan **studi implementasi Machine Learning berbasis pembelajaran terstruktur** pada mata kuliah Machine Learning. Pendekatan ini mengombinasikan pemahaman konseptual, praktik komputasi, serta evaluasi berbasis studi kasus data nyata. Proses pembelajaran dirancang secara bertahap, dimulai dari pengenalan konsep dasar Artificial Intelligence dan Machine Learning, hingga implementasi model Machine Learning klasik dan Deep Learning melalui evaluasi Ujian Tengah Semester (UTS) dan Ujian Akhir Semester (UAS).

Pendekatan ini bersifat **applied learning**, di mana mahasiswa tidak hanya mempelajari teori, tetapi juga menerapkannya langsung pada dataset pendidikan yang relevan. Evaluasi pembelajaran dilakukan melalui dua tahapan utama, yaitu implementasi algoritma **Support Vector Machine (SVM)** pada UTS dan **Multilayer Perceptron (MLP)** pada UAS. Kedua tahapan ini menjadi dasar metodologi eksperimen yang dijelaskan pada bagian selanjutnya.

2.2 Dataset dan Lingkungan Pengembangan

Dataset yang digunakan pada seluruh tahapan metodologi adalah **Higher Education Students Performance Evaluation Dataset** yang diperoleh dari **UCI Machine Learning Repository**. Dataset ini terdiri dari **145 data mahasiswa** dengan **31 fitur** yang mencakup aspek demografis, latar belakang keluarga, kebiasaan belajar, serta aktivitas akademik mahasiswa. Variabel target berupa **nilai akhir mahasiswa** yang diklasifikasikan ke dalam delapan kelas, yaitu *Fail, DD, DC, CC, CB, BB, BA,* dan *AA*.

Seluruh eksperimen dilakukan menggunakan bahasa pemrograman **Python** dengan dukungan pustaka *scikit-learn* untuk implementasi algoritma Machine Learning klasik dan *TensorFlow Keras* untuk implementasi Deep Learning. Lingkungan pengembangan menggunakan *Jupyter Notebook* atau *Visual Studio Code* yang mendukung eksplorasi data, pemodelan, dan evaluasi model secara interaktif.

2.3 Pra-pemrosesan Data

Tahapan pra-pemrosesan data dilakukan untuk memastikan kualitas data sebelum digunakan dalam proses pelatihan model. Tahapan ini meliputi:

1. **Pembersihan Data**, yaitu pemeriksaan nilai kosong (*missing values*) dan duplikasi data.

2. **Transformasi Data Kategorikal**, dengan menerapkan teknik *Label Encoding* untuk mengonversi data kategorikal menjadi representasi numerik.

3. **Normalisasi Fitur**, menggunakan teknik *Feature Scaling* agar seluruh fitur memiliki skala yang seragam.

    - Pada implementasi UTS digunakan **StandardScaler**, sedangkan pada implementasi UAS digunakan **Min-Max Scaling**, disesuaikan dengan karakteristik algoritma yang digunakan.

4. **Pembagian Data**, dengan rasio **80% data latih dan 20% data uji** menggunakan teknik *Stratified Sampling* untuk menjaga proporsi kelas pada data yang bersifat tidak seimbang (*imbalanced*).

Tahapan pra-pemrosesan ini bertujuan untuk meningkatkan stabilitas dan kinerja model Machine Learning yang digunakan.

2.4 Implementasi UTS: Support Vector Machine

Pada tahap Ujian Tengah Semester (UTS), metode **Support Vector Machine (SVM)** digunakan sebagai model klasifikasi utama. SVM dipilih karena memiliki kemampuan generalisasi yang baik pada data berdimensi tinggi dan sering digunakan pada studi *Educational Data Mining*. Model SVM diimplementasikan dengan beberapa variasi kernel, yaitu *Linear, Radial Basis Function (RBF), Polynomial,* dan *Sigmoid*.

Optimasi parameter dilakukan menggunakan **Grid Search Cross-Validation** untuk menentukan kombinasi parameter terbaik, seperti nilai regularisasi CCC, parameter kernel, dan derajat polinomial. Evaluasi performa model dilakukan menggunakan metrik **akurasi, presisi, recall, dan F1-score**. Selain itu, dilakukan **uji statistik McNemar** untuk menguji signifikansi perbedaan performa antara model SVM dan model baseline.

2.5 Implementasi UAS: Multilayer Perceptron

Pada tahap Ujian Akhir Semester (UAS), digunakan metode **Multilayer Perceptron (MLP)** sebagai pengembangan dari pendekatan Machine Learning klasik menuju Deep Learning. Model MLP dibangun menggunakan arsitektur *feedforward neural network* dengan beberapa *hidden layer* yang berfungsi untuk menangkap hubungan non-linear yang kompleks dalam data pendidikan.

Model MLP menggunakan fungsi aktivasi **ReLU** pada *hidden layer* dan **Softmax** pada *output layer* untuk klasifikasi multikelas. Proses pelatihan dilakukan menggunakan **optimizer Adam** dan fungsi *loss Sparse Categorical Crossentropy*. Untuk mencegah *overfitting*, diterapkan teknik **dropout** dan *early stopping*. Evaluasi performa dilakukan menggunakan metrik yang sama dengan implementasi SVM agar hasil dapat dibandingkan secara objektif.

2.6 Alur Umum Metodologi Pembelajaran Machine Learning

Secara umum, alur metodologi pembelajaran dan implementasi Machine Learning pada penelitian ini dapat dirangkum sebagai berikut:

1. Pengumpulan dan pemahaman dataset pendidikan.

2. Pra-pemrosesan dan transformasi data.
3. Implementasi model Machine Learning (SVM pada UTS).
4. Implementasi model Deep Learning (MLP pada UAS).
5. Evaluasi performa model menggunakan metrik evaluasi yang relevan.
6. Analisis hasil untuk mendukung sistem peringatan dini akademik.

Alur ini mencerminkan proses pembelajaran Machine Learning yang sistematis dan aplikatif, sesuai dengan tujuan mata kuliah dan kebutuhan analisis data pendidikan.

● Hasil Dan Pembahasan

3.1 Pertemuan 1 – Pengantar Machine Learning: Definisi, Konsep Dasar, dan Relevansi Industri

3.1.1 Deskripsi Singkat

Pertemuan pertama membahas pengantar Machine Learning sebagai bagian dari Artificial Intelligence yang berfokus pada kemampuan sistem komputer untuk belajar dari pengalaman. Materi mencakup posisi Machine Learning dalam taksonomi AI, definisi formal Machine Learning menurut Tom Mitchell, peran data sebagai experience, serta pentingnya penguasaan fundamental dalam menghadapi perkembangan teknologi dan kebutuhan industri. Selain itu, dibahas pula relevansi Machine Learning dalam dunia kerja, tantangan penerapan di industri, serta pentingnya pembelajaran berkelanjutan (long life learning).

3.1.2 Penjelasan Detail

Machine Learning merupakan salah satu cabang utama dari Artificial Intelligence yang bertujuan memungkinkan sistem komputer meningkatkan kinerjanya secara otomatis berdasarkan pengalaman. Dalam konteks perkuliahan ini, Machine Learning diposisikan sebagai fondasi utama bagi pengembangan sistem cerdas yang adaptif dan berbasis data.

Perkembangan industri saat ini menunjukkan bahwa Machine Learning menjadi dasar bagi berbagai profesi strategis, seperti Machine Learning Engineer, AI Engineer, Data Scientist, dan Data Analyst. Profesi-profesi tersebut tidak hanya menuntut kemampuan penggunaan perangkat lunak, tetapi juga pemahaman konsep fundamental seperti probabilitas, statistika, dan matematika. Oleh karena itu, pembelajaran Machine Learning di perguruan tinggi diarahkan untuk membangun pemahaman konseptual yang kuat sebelum memasuki tahap implementasi lanjutan.

Definisi formal Machine Learning yang digunakan dalam perkuliahan ini mengacu pada Tom Mitchell, yang menyatakan bahwa suatu program komputer dikatakan belajar dari pengalaman apabila kinerjanya pada suatu tugas meningkat seiring bertambahnya pengalaman tersebut. Definisi ini menekankan tiga komponen utama, yaitu experience (E), task (T), dan performance measure (P). Sebuah sistem hanya dapat dikatakan “belajar” apabila terdapat peningkatan performa yang terukur, bukan sekadar perubahan perilaku.

Dalam Machine Learning, experience umumnya direpresentasikan dalam bentuk data. Data didefinisikan sebagai fakta yang telah terjadi dan dikumpulkan dari proses observasi atau pencatatan. Tidak semua data bersifat sempurna, karena dalam praktik sering dijumpai noise yang dapat memengaruhi kualitas pembelajaran. Data inilah yang digunakan untuk membentuk pengetahuan, yang dapat berupa model, hipotesis, atau pola tertentu. Pengetahuan yang dihasilkan harus konsisten terhadap data, dan dapat berubah apabila data yang mendasarinya juga berubah.

Konsep pembelajaran dalam Machine Learning memiliki kemiripan dengan proses belajar manusia. Sebagai contoh, dalam konteks akademik, ujian dapat dipandang sebagai task, nilai sebagai performance measure, dan proses belajar sebagai experience. Apabila nilai meningkat setelah mahasiswa memperoleh pengalaman belajar tambahan, maka proses pembelajaran telah terjadi. Analogi ini memperjelas cara kerja Machine Learning secara konseptual.

Selain aspek teknis, penerapan Machine Learning di industri juga menghadapi berbagai tantangan non-teknis, seperti keamanan data, kepatuhan terhadap regulasi privasi (misalnya PDP dan GDPR), serta biaya infrastruktur. Oleh karena itu, penerapan Machine Learning tidak dapat dilakukan secara sembarangan tanpa mempertimbangkan aspek hukum, etika, dan keberlanjutan sistem.

3.1.3 Mind Map / Taksonomi Machine Learning

Secara konseptual, posisi Machine Learning dalam Artificial Intelligence dapat digambarkan sebagai berikut:

Taksonomi ini menunjukkan bahwa Machine Learning merupakan bagian dari Artificial Intelligence yang berfokus pada pembelajaran berbasis data, berbeda dengan pendekatan AI lain yang berbasis pencarian atau penalaran simbolik.

3.1.4 Pseudocode Proses Pembelajaran Machine Learning

Berikut disajikan pseudocode sederhana untuk menggambarkan proses pembelajaran Machine Learning berdasarkan definisi Tom Mitchell:

```
Input:
    Dataset D (experience)
    Task T
    Performance Measure P

Process:
    Inisialisasi model M
```

```
    Hitung performa awal P_old

    While performa belum optimal:
        Ambil data dari D
        Latih model M untuk menyelesaikan task T
        Hitung performa baru P_new

        If P_new > P_old:
            Perbarui model M
            P_old ← P_new
        Else:
            Simpan kesalahan sebagai pembelajaran

Output:
    Model M terlatih dengan performa yang meningkat
```

Pseudocode ini menunjukkan bahwa proses pembelajaran Machine Learning bersifat iteratif dan bergantung pada peningkatan performa berdasarkan pengalaman.

3.1.5 Rumus Matematis (LaTeX)

Definisi pembelajaran Machine Learning dapat direpresentasikan secara matematis sebagai hubungan antara task, experience, dan performance measure, yaitu:

$$P(T) = f(E)$$

yang menyatakan bahwa performa pada suatu task merupakan fungsi dari pengalaman yang dimiliki sistem.

Selain itu, peningkatan performa dapat dinyatakan sebagai:

$$P(T)baru > P(T)lama$$

Rumus ini menegaskan bahwa suatu sistem dikatakan belajar apabila terjadi peningkatan performa setelah memperoleh pengalaman tambahan.

3.2 Pertemuan 2 – Artificial Intelligence, Machine Learning, dan Penerapannya pada Sistem Karbon dan Kepatuhan Regulasi

3.2.1 Deskripsi Singkat

Pertemuan kedua membahas keterkaitan antara Artificial Intelligence (AI), Machine Learning (ML), dan Deep Learning dalam konteks penerapan dunia nyata, khususnya pada sistem pengelolaan karbon, pajak karbon, dan validasi proyek lingkungan. Materi menekankan bahwa AI tidak hanya berfungsi sebagai alat pemrograman, tetapi juga sebagai sistem otomasi, validasi data, serta penunjang kepatuhan regulasi (compliance) yang terintegrasi dengan sistem nasional maupun internasional.

3.2.2 Penjelasan Detail

AI dalam Isu Lingkungan dan Karbon

Perubahan iklim global berdampak signifikan terhadap kenaikan permukaan laut, degradasi lingkungan, serta meningkatnya risiko bencana alam. Salah satu strategi mitigasi yang banyak diterapkan adalah melalui proyek karbon, seperti reforestasi dan rehabilitasi hutan. Dalam skema ini, perusahaan tidak memiliki lahan secara langsung, melainkan mengklaim kemampuan serapan karbon berdasarkan sertifikat resmi dalam periode waktu tertentu.

Artificial Intelligence berperan sebagai alat bantu untuk memastikan bahwa klaim karbon tersebut valid, akurat, dan dapat dipertanggungjawabkan secara ilmiah maupun hukum. Tanpa dukungan sistem berbasis AI, proses validasi klaim karbon berpotensi rawan manipulasi data dan pelanggaran regulasi.

AI untuk Validasi Proyek Karbon

AI digunakan dalam beberapa tahapan utama validasi proyek karbon. Pertama, validasi penanaman pohon dilakukan melalui analisis citra menggunakan teknik computer vision untuk memastikan jenis, genus, dan spesies pohon sesuai dengan klaim yang diajukan. Kedua, validasi lokasi proyek dilakukan menggunakan citra satelit untuk memverifikasi kepemilikan lahan, jenis tutupan lahan, serta status hukum wilayah tersebut. Ketiga, sistem AI digunakan untuk mengestimasi daya serap karbon dalam satuan CO_2 ekivalen, yang menjadi dasar penerbitan sertifikat karbon.

Pendekatan ini memungkinkan proses validasi dilakukan secara lebih objektif, konsisten, dan efisien dibandingkan pemeriksaan manual.

Pajak Karbon dan Pasar Karbon

Penerapan pajak karbon menyebabkan perbedaan harga barang antarnegara, terutama bagi produk yang tidak melakukan mitigasi emisi. Di Indonesia, mekanisme perdagangan karbon difasilitasi melalui IDX Carbon, sementara pada tingkat internasional terdapat lembaga sertifikasi seperti VERRA. AI digunakan untuk menghitung jejak karbon produk, menentukan kewajiban pajak karbon, serta mengotomatisasi proses transaksi dan pencatatan sertifikat karbon.

Compliance dan Regulasi

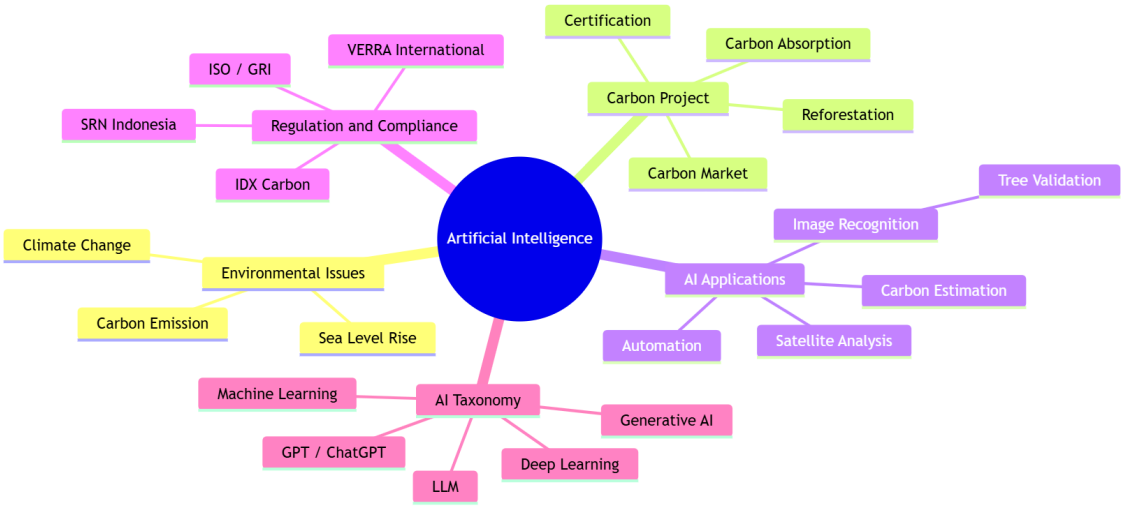
Dalam penerapan industri, sistem AI wajib mematuhi berbagai regulasi nasional dan internasional, seperti Sistem Registri Nasional (SRN), ISO 27001, Global Reporting Initiative (GRI), serta standar pasar karbon internasional. AI berfungsi untuk memastikan bahwa seluruh proses, data, dan sertifikat yang dihasilkan memenuhi prinsip governance, risk, and compliance (GRC), sehingga mengurangi risiko hukum dan reputasi perusahaan.

Posisi Machine Learning dalam Taksonomi AI

Materi ini juga menegaskan hierarki konsep dalam Artificial Intelligence. Machine Learning merupakan bagian dari AI yang bersifat induktif, berbeda dengan expert system yang berbasis aturan (rule-based). Pendekatan induktif memungkinkan sistem belajar langsung dari data, sehingga lebih adaptif terhadap perubahan kondisi lingkungan dan regulasi.

3.2.3 Mind Map / Taksonomi Artificial Intelligence

Hubungan konseptual antara AI, ML, dan teknologi turunannya dapat digambarkan sebagai berikut:



3.2.4 Pseudocode Sistem Berbasis AI pada Proyek Karbon

Pseudocode 1: Validasi Penanaman Pohon

```
Input:
    Image of planted tree

Process:
    Detect tree object from image
    Identify tree species using AI model

If identified species matches claimed species then
    Validation status = valid
Else
    Validation status = invalid
End If

Output:
    Validation status
```

Pseudocode 2: Validasi Lokasi Proyek Karbon

```
Input:
    Satellite image
```

Claimed project location

Process:
Analyze land cover type
Check land ownership
Verify legal status

If land is forest AND legally allowed then
Location status = approved
Else
Location status = rejected
End If

Output:
Location status

Pseudocode 3: Estimasi dan Klaim Karbon
Input:
Validated project data

Process:
Calculate carbon absorption capacity
Convert to CO2 equivalent
Register certificate into system

Output:
Carbon certificate

3.2.5 Rumus Matematis (LaTeX)

Estimasi serapan karbon dapat direpresentasikan secara matematis sebagai:

$$Ctotal = \sum_{i=1}^{n} Ci$$

dengan:

- CtotalC_{total}Ctotal : total serapan karbon proyek
- CiC_iCi : serapan karbon tiap unit pohon atau area
- nnn : jumlah unit yang divalidasi

Konversi ke satuan CO_2 ekivalen dinyatakan sebagai:

$$CO2eq = Ctotal \times \alpha$$

di mana α\alphaα merupakan faktor konversi karbon ke CO_2 ekivalen sesuai standar internasional.

3.3 Pertemuan 3 – Inductive Learning, Deductive Learning, dan Pembentukan Model pada Machine Learning

3.3.1 Deskripsi Singkat

Pertemuan ketiga membahas konsep fundamental pembelajaran mesin, khususnya perbedaan antara pembelajaran induktif (*inductive learning*) dan deduktif (*deductive learning*). Selain itu, materi mengulas definisi formal Machine Learning, peran data sebagai *experience*, serta proses pembentukan model atau hipotesis. Sebagai contoh konkret, diperkenalkan algoritma Find-S yang menggambarkan bagaimana sebuah model machine learning dibangun secara induktif dari data berlabel.

3.3.2 Penjelasan Detail

Inductive Learning dan Deductive Learning

Artificial Intelligence memiliki komponen *learning* yang secara umum terbagi menjadi dua pendekatan utama, yaitu pembelajaran induktif dan deduktif.

Pembelajaran induktif bergerak dari data yang bersifat khusus (spesifik) menuju pengetahuan yang bersifat umum (general). Machine Learning termasuk ke dalam pendekatan ini karena model dibangun berdasarkan data hasil observasi atau eksperimen, tanpa aturan eksplisit yang ditentukan sejak awal. Sistem belajar dengan cara mengamati pola dari data dan melakukan generalisasi.

Sebaliknya, pembelajaran deduktif bergerak dari pengetahuan umum menuju kasus khusus. Pendekatan ini umum digunakan pada *expert system*, di mana aturan, logika, atau fungsi keputusan telah didefinisikan terlebih dahulu oleh pakar. Sistem tidak belajar dari data, tetapi mengeksekusi aturan yang telah tersedia.

Dengan karakteristik tersebut, Machine Learning dikategorikan sebagai pembelajaran induktif karena sistem memperoleh pengetahuan langsung dari data, bukan dari aturan yang dituliskan secara eksplisit.

Definisi Formal Machine Learning

Definisi Machine Learning yang digunakan dalam perkuliahan ini mengacu pada definisi klasik oleh Tom Mitchell. Sebuah sistem dikatakan belajar apabila sistem tersebut:

1. Mengerjakan suatu *task* (T),
2. Kinerjanya diukur menggunakan *performance measure* (P),
3. Kinerjanya meningkat seiring bertambahnya *experience* (E).

Konsep ini dapat dianalogikan dengan proses akademik mahasiswa, di mana ujian berperan sebagai *task*, proses belajar sebagai *experience*, dan nilai sebagai *performance measure*. Jika nilai meningkat setelah mahasiswa memperoleh lebih banyak pengalaman belajar, maka proses pembelajaran telah terjadi. Prinsip ini identik dengan mekanisme kerja Machine Learning.

Data, Experience, dan Knowledge

Dalam Machine Learning, data didefinisikan sebagai fakta yang sesuai dengan kenyataan. Data tersebut berperan sebagai *experience* yang digunakan sistem untuk belajar. Hasil dari proses pembelajaran ini disebut *knowledge*, yang dapat berbentuk model, hipotesis, atau pola (*pattern*).

Hubungan antara data dan pengetahuan bersifat adaptif. Ketika data berubah, maka hipotesis atau model yang dihasilkan juga dapat berubah. Oleh karena itu, pengetahuan dalam Machine Learning bersifat stokastik dan dinamis, bukan deterministik mutlak.

Model dan Hipotesis

Model dalam Machine Learning dapat direpresentasikan sebagai fungsi pendekatan f′(x)f'(x)f′(x) yang berusaha mendekati fungsi asli f(x)f(x)f(x) milik pakar. Karena bersifat pendekatan, model selalu memiliki kesalahan (*error*). Besarnya error diukur menggunakan *performance measure* seperti *Mean Squared Error* (MSE) atau error rata-rata lainnya.

Apabila nilai error berada di bawah ambang batas yang dapat diterima, maka model dianggap layak untuk digunakan dalam proses prediksi atau pengambilan keputusan.

Representasi Data dan Fitur

Satu data (*instance*) terdiri dari sejumlah fitur (*features* atau *dimensions*). Contoh fitur yang sering digunakan dalam dataset klasik meliputi kondisi cuaca, suhu, kelembaban, dan kecepatan angin. Data yang dilengkapi dengan label digunakan pada *supervised learning*, sedangkan data tanpa label digunakan pada *unsupervised learning*.

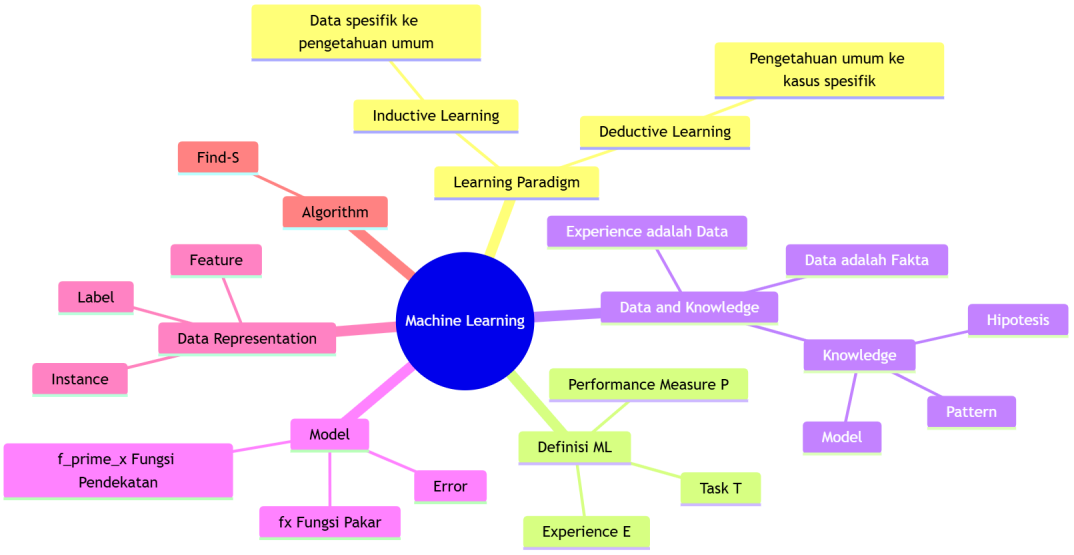
Algoritma Find-S

Find-S merupakan algoritma *supervised learning* sederhana yang bertujuan mencari hipotesis paling spesifik yang konsisten dengan seluruh data positif. Algoritma ini hanya memperhatikan data berlabel positif dan mengabaikan data negatif.

Model yang dihasilkan berupa pola atribut dengan nilai spesifik atau simbol "?". Simbol "?" menyatakan bahwa nilai atribut dapat berupa apa saja dan tetap diterima oleh model. Algoritma ini memberikan gambaran awal mengenai proses generalisasi dalam Machine Learning.

3.3.3 Mind Map / Taksonomi Konseptual

Secara konseptual, materi pertemuan ketiga dapat dirangkum sebagai berikut:



3.3.4 Pseudocode Algoritma Find-S

```
Input:
   Training data D dengan label positif dan negatif

Initialize:
   Hypothesis H = nilai paling spesifik (Ø)

For each instance x in D:
   If x is positive:
      If H is empty:
         H = x
      Else:
         For each attribute i in H:
            If H[i] ≠ x[i]:
               H[i] = "?"
   Else:
      Ignore instance

Output:
   Hypothesis H
```

3.3.5 Rumus Matematis (LaTeX)

Representasi model Machine Learning sebagai fungsi pendekatan dapat dituliskan sebagai:

$$f'(x) \approx f(x)$$

Kesalahan prediksi model dapat diukur menggunakan *Mean Squared Error* (MSE):

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (yi - y\hat{}i)^{2}$$

dengan:

- yiy_iyi : nilai aktual
- y^i\hat{y}_iy^i : nilai prediksi model
- nnn : jumlah data

3.4 Pertemuan 4 – Algoritma Supervised Learning, Decision Tree, dan Evaluasi Model Machine Learning

3.4.1 Deskripsi Singkat

Pertemuan keempat membahas lanjutan *supervised learning* dengan fokus pada perbedaan antara algoritma, metode, dan model dalam Machine Learning. Materi menekankan penggunaan Decision Tree sebagai model klasifikasi serta algoritma ID3 dan C4.5 sebagai pembentuk struktur pohon keputusan. Selain itu, dibahas pentingnya evaluasi model menggunakan data *training* dan *testing*, serta dampak kesalahan prediksi terhadap proses bisnis dan pengambilan keputusan di dunia nyata.

3.4.2 Penjelasan Detail

Supervised Learning dan Ragam Algoritma

Supervised learning merupakan pendekatan pembelajaran mesin yang menggunakan data berlabel sebagai dasar pembentukan model. Dalam praktiknya, terdapat berbagai algoritma supervised learning yang dikembangkan untuk kebutuhan dan karakteristik data yang berbeda.

Beberapa kategori utama algoritma supervised learning meliputi algoritma berbasis hipotesis sederhana, algoritma berbasis vektor, algoritma berbasis pohon keputusan, algoritma berbasis probabilistik, serta algoritma berbasis jaringan saraf. Dari berbagai pendekatan tersebut, Decision Tree menjadi salah satu model yang paling mudah dipahami karena memiliki representasi visual yang intuitif dan logika pengambilan keputusan yang menyerupai cara berpikir manusia.

Algoritma, Metode, dan Model

Dalam Machine Learning, terdapat perbedaan konseptual yang penting antara metode, algoritma, dan model. Metode merupakan pendekatan atau strategi umum dalam membangun sistem pembelajaran. Algoritma adalah prosedur komputasi yang digunakan untuk menghasilkan model berdasarkan data. Model merupakan hasil akhir dari proses pembelajaran yang merepresentasikan pengetahuan yang dipelajari sistem.

Output dari Machine Learning bukanlah kode program, melainkan model, yang dapat berupa pohon keputusan, vektor parameter, matriks bobot, atau model probabilistik. Model inilah yang kemudian diintegrasikan ke dalam sistem untuk melakukan prediksi terhadap data baru.

Decision Tree sebagai Model Machine Learning

Decision Tree adalah model Machine Learning yang memiliki struktur hierarkis menyerupai pohon. Struktur ini terdiri dari *root node* sebagai simpul awal yang mewakili atribut paling signifikan, *internal node* sebagai simpul pengambilan keputusan berdasarkan nilai atribut, serta *leaf node* sebagai simpul akhir yang merepresentasikan kelas atau keputusan.

Decision Tree bekerja dengan cara membagi data secara rekursif berdasarkan atribut yang paling berpengaruh terhadap kelas target. Proses ini menghasilkan model yang bersifat interpretatif, mudah divisualisasikan, dan dapat dijelaskan secara logis.

Algoritma ID3 dan Information Gain

ID3 (*Iterative Dichotomiser 3*) merupakan algoritma yang digunakan untuk membangun Decision Tree dengan memilih atribut terbaik berdasarkan nilai Information Gain. Information Gain mengukur seberapa besar suatu atribut mampu mengurangi ketidakpastian data.

Langkah umum algoritma ID3 meliputi perhitungan entropy dataset, perhitungan entropy untuk setiap atribut, perhitungan information gain, pemilihan atribut dengan gain tertinggi sebagai root, dan pengulangan proses hingga pohon keputusan terbentuk. Atribut dengan nilai information gain terbesar dianggap paling signifikan dalam menentukan kelas target.

Kelas, Fitur, dan Target

Dalam dataset supervised learning, fitur (*feature*) merupakan atribut atau variabel input, sedangkan kelas atau label adalah nilai yang ingin diprediksi. Target dapat bersifat diskrit (kategorikal) maupun kontinu (numerik).

Contoh klasik yang sering digunakan adalah *Play Tennis Dataset*, di mana fitur meliputi *outlook*, *temperature*, *humidity*, dan *wind*, sedangkan labelnya adalah keputusan bermain tenis (*yes* atau *no*).

Evaluasi Model dan Confusion Matrix

Model Machine Learning harus dievaluasi untuk mengetahui kualitas dan tingkat keandalannya. Salah satu metode evaluasi yang umum digunakan adalah *split validation*, yaitu membagi dataset menjadi data training dan data testing.

Hasil prediksi dianalisis menggunakan confusion matrix yang terdiri dari *True Positive*, *True Negative*, *False Positive*, dan *False Negative*. Kesalahan prediksi memiliki dampak langsung terhadap proses bisnis, khususnya pada sistem keamanan dan deteksi penipuan, karena dapat menyebabkan kerugian finansial, hilangnya kepercayaan pengguna, hingga risiko hukum.

Overfitting dan Underfitting

Decision Tree juga digunakan untuk menjelaskan dua kondisi penting dalam Machine Learning, yaitu *underfitting* dan *overfitting*. Underfitting terjadi ketika model terlalu sederhana sehingga tidak mampu menangkap pola data, sedangkan overfitting terjadi ketika model terlalu kompleks dan hanya cocok pada data training.

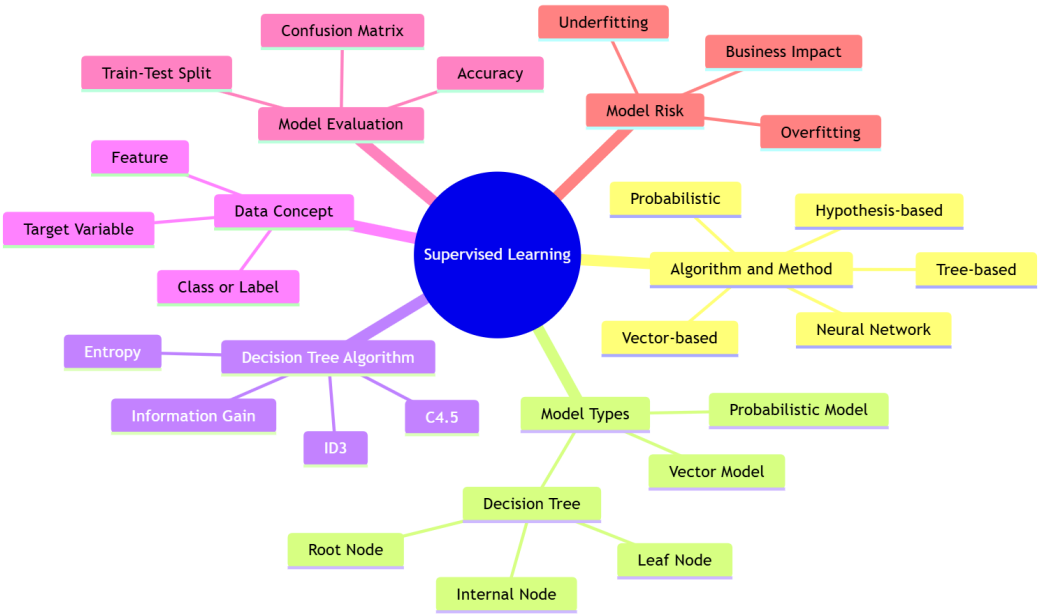
Keseimbangan antara kompleksitas model dan kemampuan generalisasi menjadi kunci keberhasilan penerapan Machine Learning dalam sistem nyata.

Implementasi dan Alat Pendukung

Dalam praktik, pengolahan data sering menggunakan pustaka seperti Pandas untuk memuat dan memanipulasi dataset. Notebook seperti Jupyter Notebook digunakan untuk eksperimen dan validasi awal, sedangkan sistem produksi membutuhkan implementasi yang lebih terstruktur, stabil, dan siap digunakan secara nyata.

3.4.3 Mind Map / Taksonomi Konseptual

Secara konseptual, materi pertemuan keempat dapat dirangkum sebagai berikut:



3.4.4 Pseudocode Pembentukan Decision Tree (ID3)

Input:
Dataset D dengan fitur dan label

If semua data dalam D memiliki label yang sama then
Return leaf node dengan label tersebut
End If

If tidak ada atribut tersisa then
Return leaf node dengan label mayoritas
End If

For setiap atribut A dalam D:
Hitung entropy(D)
Hitung entropy(D | A)
Hitung Information Gain(A)
End For

Pilih atribut dengan Information Gain terbesar sebagai root

For setiap nilai v dari atribut terpilih:
Buat subset Dv
Rekursif bangun subtree dari Dv
End For

Output:
Decision Tree

3.4.5 Pseudocode Evaluasi Model dengan Confusion Matrix
Input:
Model M
Data testing T

For setiap data x dalam T:
Prediksi kelas y_pred menggunakan M
Bandingkan dengan kelas asli y_true
Simpan hasil ke confusion matrix
End For

Hitung:
Accuracy
Error rate

Output:
Confusion matrix dan nilai evaluasi

3.4.6 Rumus Matematis (LaTeX)

Entropy dataset didefinisikan sebagai:

$$Entropy(S) = -\sum_{i=1}^{c} pilog2(pi)$$

Information Gain untuk atribut AAA dihitung sebagai:

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)\sum} \frac{|S|}{|Sv|} Entropy(Sv)$$

dengan:

- pip_ipi : proporsi kelas ke-iii
- SvS_vSv : subset data dengan nilai atribut vvv
- ∣S∣|S|∣S∣ : jumlah total data

3.5 Pertemuan 5 – Pembangunan Struktur Tree dan Algoritma ID3 dalam Machine Learning

3.5.1 Deskripsi Singkat

Pertemuan kelima membahas keterkaitan antara struktur data tree dalam ilmu komputer dan decision tree dalam machine learning, khususnya algoritma ID3 (Iterative Dichotomiser 3). Materi difokuskan pada bagaimana pohon keputusan direpresentasikan dalam memori komputer menggunakan konsep node, pointer, dan relasi hierarkis, serta bagaimana proses pembelajaran supervised learning memanfaatkan perhitungan entropy dan information gain untuk memilih atribut terbaik. Studi kasus *Play Tennis Dataset* digunakan untuk menggambarkan pembentukan decision tree secara bertahap hingga menghasilkan aturan klasifikasi yang deterministik.

3.5.2 Penjelasan Detail

Struktur Data Tree dalam Komputasi

Tree merupakan struktur data hierarkis yang banyak digunakan dalam ilmu komputer untuk merepresentasikan hubungan bertingkat. Struktur ini terdiri dari *root* sebagai node paling atas, *node* sebagai elemen penyimpan data, *child* sebagai node turunan, serta *leaf* sebagai node tanpa anak.

Dalam implementasi komputer, setiap node disimpan sebagai objek di memori dan memiliki alamat memori yang umumnya direpresentasikan dalam bentuk heksadesimal. Relasi antar node tidak disimpan secara fisik berurutan, melainkan melalui referensi atau pointer yang menunjuk ke alamat memori node lain. Berdasarkan jumlah anak, tree dapat diklasifikasikan menjadi binary tree, yang memiliki maksimal dua anak, dan N-ary tree, yang jumlah anaknya tidak dibatasi. Konsep inilah yang menjadi fondasi bagi decision tree dalam machine learning.

Representasi Node dalam Pemrograman

Setiap node dalam struktur tree umumnya memiliki tiga komponen utama, yaitu data yang disimpan, referensi ke parent, serta daftar child. Representasi ini memungkinkan sistem melakukan proses traversal tree, pembentukan struktur hierarkis, dan penyimpanan model machine learning ke dalam memori komputer.

Dalam konteks machine learning, decision tree bukan sekadar konsep visual, melainkan struktur data nyata yang disimpan dan diproses oleh sistem untuk melakukan prediksi terhadap data baru.

Supervised Learning, Label, dan Target

Decision tree termasuk ke dalam pendekatan supervised learning, di mana setiap data latih memiliki label sebagai acuan pembelajaran. Label atau class umumnya digunakan untuk data kategorikal, seperti *Yes/No* atau *Lulus/Tidak Lulus*, sedangkan istilah target sering digunakan untuk data kontinu atau numerik, seperti harga rumah atau nilai prediksi.

Perbedaan ini penting karena memengaruhi jenis permasalahan machine learning, apakah berupa klasifikasi atau regresi, serta memengaruhi evaluasi model yang digunakan.

Algoritma ID3

ID3 merupakan algoritma pembentuk decision tree yang bekerja berdasarkan prinsip pengurangan ketidakpastian data. Algoritma ini menggunakan entropy sebagai ukuran ketidakmurnian dataset dan information gain sebagai kriteria pemilihan atribut terbaik.

Tujuan utama ID3 adalah memilih atribut dengan nilai information gain terbesar sebagai *root* atau node lanjutan, kemudian mengulangi proses yang sama secara rekursif hingga setiap node menjadi murni atau tidak ada atribut yang tersisa.

Perhitungan Entropy

Entropy digunakan untuk mengukur tingkat ketidakpastian dalam suatu dataset. Jika seluruh data berada dalam satu kelas, maka entropy bernilai nol. Sebaliknya, jika data terbagi secara seimbang ke dalam beberapa kelas, maka entropy mencapai nilai maksimum.

Perhitungan entropy dilakukan dengan menghitung proporsi masing-masing kelas, kemudian menjumlahkan hasil perkalian proporsi tersebut dengan logaritma basis dua dari proporsi itu sendiri. Nilai entropy inilah yang menjadi dasar perhitungan information gain.

Information Gain

Information gain menunjukkan seberapa besar suatu atribut mampu mengurangi ketidakpastian data. Semakin besar nilai information gain suatu atribut, semakin penting atribut tersebut dalam proses pemisahan kelas.

Dalam algoritma ID3, atribut dengan nilai information gain terbesar dipilih sebagai root atau node utama karena dianggap paling informatif dalam menentukan kelas target.

Studi Kasus: Play Tennis Dataset

Play Tennis Dataset merupakan dataset klasik yang sering digunakan untuk menjelaskan cara kerja decision tree. Dataset ini memiliki fitur *Outlook*, *Temperature*, *Humidity*, dan *Wind*, dengan label *Play Tennis* (Yes/No).

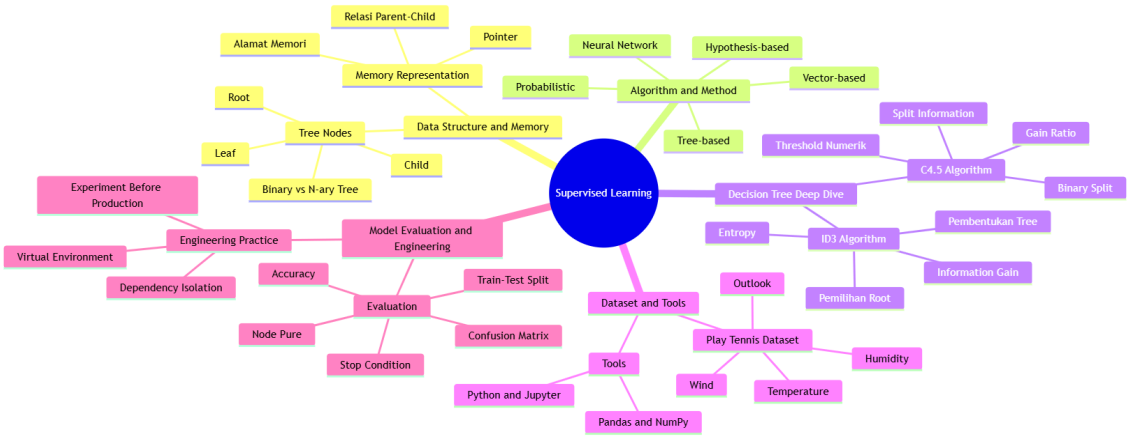
Hasil perhitungan information gain menunjukkan bahwa atribut *Outlook* memiliki gain tertinggi sehingga dipilih sebagai root. Atribut *Humidity* dan *Wind* digunakan pada level berikutnya, sedangkan *Temperature* tidak digunakan karena kontribusinya paling kecil. Dari struktur tree tersebut, dihasilkan aturan klasifikasi yang bersifat deterministik, seperti:

- Jika *Outlook* = *Overcast* maka keputusan adalah *Yes*
- Jika *Outlook* = *Sunny* maka keputusan ditentukan oleh *Humidity*

- Jika *Outlook* = *Rain* maka keputusan ditentukan oleh *Wind*

3.5.3 Mind Map / Taksonomi Konseptual

Secara konseptual, materi pertemuan kelima dapat dirangkum sebagai berikut:



3.5.4 Pseudocode Struktur Node Tree

```
Node:
  data
  parent
  children (list)
```

3.5.5 Pseudocode Algoritma ID3

```
ID3(dataset, attributes):
  if semua data dalam satu kelas:
    return leaf dengan label tersebut

  if attributes kosong:
    return leaf dengan kelas mayoritas

  hitung entropy dataset
  hitung information gain setiap atribut
  pilih atribut dengan gain terbesar sebagai root

  buat node root
  untuk setiap nilai atribut terpilih:
    subset ← filter dataset berdasarkan nilai
    if subset kosong:
      tambahkan leaf dengan kelas mayoritas
    else:
```

child ← ID3(subset, attributes - atribut_terpilih)
tambahkan child ke root

return root

3.5.6 Rumus Matematis (LaTeX)

Entropy dataset didefinisikan sebagai:

$$Entropy(S) = -\sum_{i=1}^{c} pilog2(pi)$$

Information Gain untuk atribut AAA dihitung sebagai:

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S|}{|Sv|} Entropy(Sv)$$

dengan:

- pip_ipi adalah proporsi data pada kelas ke-iii
- SvS_vSv adalah subset data dengan nilai atribut vvv

3.6 Pertemuan 6 – Pengembangan Decision Tree: ID3, C4.5, Data Numerik, dan Implementasi Awal

3.6.1 Deskripsi Singkat

Pertemuan keenam membahas pendalaman algoritma Decision Tree dengan fokus pada perhitungan lanjutan entropy dan information gain, serta keterbatasan algoritma ID3 ketika dihadapkan pada fitur numerik (kontinu). Materi diperluas dengan pengenalan algoritma C4.5 yang mampu menangani data numerik melalui mekanisme binary split dan gain ratio. Selain aspek teoretis, pertemuan ini juga memperkenalkan penggunaan Jupyter Notebook dan virtual environment Python sebagai sarana eksplorasi, validasi komputasi, dan praktik awal implementasi machine learning.

3.6.2 Penjelasan Detail

Review Algoritma Decision Tree dan ID3

Decision Tree bertujuan membangun model klasifikasi dengan memilih atribut paling signifikan sebagai *root*, kemudian membagi dataset secara bertahap hingga setiap node mencapai kondisi murni (*pure*). Algoritma ID3 menjalankan proses ini dengan menghitung entropy dataset, entropy setiap atribut, dan information gain untuk menentukan atribut terbaik.

Entropy digunakan untuk mengukur tingkat ketidakpastian distribusi kelas, sedangkan information gain mengukur seberapa besar kontribusi suatu atribut dalam mengurangi ketidakpastian tersebut. Atribut dengan nilai information gain tertinggi dipilih sebagai node keputusan.

Entropy dan Information Gain Lanjutan

Nilai entropy bergantung pada distribusi kelas dalam dataset. Jika seluruh data berada dalam satu kelas, entropy bernilai nol. Sebaliknya, jika data terbagi seimbang ke dalam beberapa kelas, entropy mencapai nilai maksimum. Information gain dihitung sebagai selisih antara entropy total dataset dan entropy subset yang dihasilkan oleh suatu atribut, dengan mempertimbangkan proporsi data pada masing-masing subset.

Pendekatan ini memungkinkan sistem memilih atribut yang paling efektif dalam memisahkan kelas target pada setiap tahap pembentukan tree.

Pembentukan Node Lanjutan (Recursive Splitting)

Setelah root ditentukan, dataset dibagi menjadi beberapa subset berdasarkan nilai atribut tersebut. Untuk setiap subset, proses perhitungan entropy dan information gain diulang secara rekursif. Proses berhenti apabila node telah murni atau tidak ada atribut yang tersisa.

Tidak semua atribut selalu digunakan dalam pembentukan tree. Jika suatu node sudah mencapai kondisi murni, maka proses pembelajaran pada cabang tersebut dihentikan meskipun masih terdapat atribut lain yang belum digunakan.

Keterbatasan ID3 pada Data Numerik

Algoritma ID3 bekerja optimal pada data diskrit atau kategorikal. Ketika dihadapkan pada data numerik kontinu, seperti *sepal length* atau *petal width* pada Iris Dataset, ID3 tidak dapat langsung menentukan cabang keputusan karena tidak tersedia nilai kategori yang eksplisit.

Keterbatasan ini mendorong pengembangan algoritma decision tree yang lebih fleksibel, salah satunya adalah C4.5.

Algoritma C4.5 dan Binary Split

C4.5 merupakan pengembangan dari ID3 yang mampu menangani data numerik dengan melakukan binary split. Pada pendekatan ini, fitur numerik dibagi menjadi dua kelompok berdasarkan nilai ambang (*threshold*), yaitu nilai yang lebih kecil atau sama dengan threshold dan nilai yang lebih besar dari threshold.

Nilai threshold ditentukan dengan menghitung titik tengah antara nilai minimum dan maksimum suatu fitur, kemudian mengevaluasi performa pemisahan data menggunakan kriteria evaluasi tertentu.

Gain Ratio dan Split Information

Berbeda dengan ID3 yang hanya menggunakan information gain, C4.5 menggunakan gain ratio sebagai kriteria pemilihan atribut. Gain ratio diperoleh dengan membagi information gain dengan split information, yaitu ukuran seberapa besar atribut membagi data tanpa memperhatikan label kelas.

Penggunaan gain ratio bertujuan mencegah bias terhadap atribut yang memiliki banyak nilai unik. Nilai gain ratio berada pada rentang 0 hingga 1, dan atribut dengan nilai gain ratio tertinggi dipilih sebagai node keputusan.

Studi Kasus: Iris Dataset

Iris Dataset merupakan contoh dataset dengan fitur numerik dan target diskrit. Dataset ini memiliki empat fitur numerik dan tiga kelas target. Proses pembentukan decision tree dengan C4.5 dilakukan dengan menghitung entropy total kelas, menentukan threshold untuk setiap fitur, melakukan binary split, serta menghitung information gain dan gain ratio.

Fitur dengan nilai gain ratio tertinggi dipilih sebagai root, kemudian proses dilanjutkan secara rekursif hingga terbentuk struktur tree yang mampu mengklasifikasikan data secara akurat.

Implementasi Awal dengan Jupyter Notebook

Jupyter Notebook digunakan sebagai media eksplorasi dan validasi perhitungan algoritma decision tree. Notebook memungkinkan eksekusi kode secara bertahap per sel, sehingga memudahkan proses pengujian, visualisasi, dan koreksi kesalahan logika.

Pendekatan ini umum digunakan dalam tahap awal pengembangan model machine learning sebelum sistem diimplementasikan ke lingkungan produksi.

Virtual Environment dan Praktik Rekayasa Perangkat Lunak

Virtual environment digunakan untuk mengisolasi dependensi Python dan mencegah konflik antar pustaka. Pendekatan ini mencerminkan praktik rekayasa perangkat lunak yang baik, di mana lingkungan eksperimen dipisahkan dari sistem utama. Perbedaan ini juga menegaskan batas antara aktivitas *data science* yang bersifat eksploratif dan *engineering* yang menuntut stabilitas sistem.

3.6.3 Mind Map / Taksonomi Konseptual

Materi pertemuan keenam dapat dirangkum dalam struktur konseptual berikut:

3.6.4 Pseudocode Perhitungan Entropy

```
Function CalculateEntropy(dataset, target):
    hitung jumlah tiap kelas
    hitung probabilitas tiap kelas
    entropy = 0

    For setiap kelas:
        entropy += -p * log2(p)

    Return entropy
```

3.6.5 Pseudocode Information Gain

```
Function InformationGain(dataset, attribute, target):
    total_entropy = CalculateEntropy(dataset, target)
    weighted_entropy = 0

    For setiap nilai attribute:
        subset = filter dataset
        weight = ukuran subset / ukuran dataset
        weighted_entropy += weight * CalculateEntropy(subset, target)

    gain = total_entropy - weighted_entropy
    Return gain
```

3.6.6 Pseudocode Binary Split (C4.5)

```
Function BinarySplit(dataset, attribute):
    min_value = nilai minimum attribute
    max_value = nilai maksimum attribute
    threshold = (min_value + max_value) / 2

    left = data dengan attribute <= threshold
    right = data dengan attribute > threshold

    Return left, right
```

3.6.7 Pseudocode Gain Ratio

```
Function GainRatio(dataset, attribute, target):
    gain = InformationGain(dataset, attribute, target)
    split_info = hitung split information attribute

    If split_info == 0:
        Return 0

    Return gain / split_info
```

3.6.8 Rumus Matematis (LaTeX)

Entropy:

$$Entropy(S) = -\sum_{i=1}^{c} pilog2(pi)$$

Information Gain:

$$Gain(S, A) = Entropy(S) - \sum_{v \in Values(A)} \frac{|S|}{|Sv|} Entropy(Sv)$$

Split Information:

$$SplitInfo(S, A) = - \sum_{v \in Values(A)} \frac{|S|}{|Sv|} log2(\frac{|S|}{|Sv|})$$

Gain Ratio:

$$GainRatio(S, A) = \frac{Gain(S,A)}{SplitInfo(S,A)}$$

3.7 Pertemuan 7 – Linear Regression Lanjutan, Implementasi Numerik, dan Stochastic Gradient Descent

3.7.1 Deskripsi Singkat

Pertemuan ketujuh membahas pendalaman **Linear Regression** dari sisi konseptual, analitik, dan numerik. Materi mencakup hubungan persamaan garis lurus dengan fungsi regresi, perhitungan parameter regresi secara analitik, serta implementasi numerik menggunakan **Python dan NumPy**. Selain itu, diperkenalkan **Stochastic Gradient Descent (SGD)** sebagai metode optimasi iteratif untuk mempelajari parameter model. Diskusi juga mencakup perbandingan struktur data **list**, **array**, dan **NumPy array**, serta pengaruh **epoch** dan **learning rate** terhadap performa model regresi.

3.7.2 Penjelasan Detail

Konsep Dasar Linear Regression

Linear Regression bertujuan memodelkan hubungan linier antara variabel input XXX dan output YYY. Model regresi linear satu variabel dapat dinyatakan sebagai:

$$y = mx + c$$

atau dalam notasi machine learning:

$$f(x) = w1x + w0$$

Parameter mmm atau w1w_1w1 merepresentasikan gradien (kemiringan garis), sedangkan ccc atau w0w_0w0 merupakan konstanta (intersep). Nilai gradien menentukan karakteristik garis regresi, di mana nilai gradien yang lebih besar menghasilkan garis yang lebih curam, dan nilai gradien yang lebih kecil menghasilkan garis yang lebih landai.

Error dan Pendekatan Garis Regresi

Pada data nyata, titik-titik observasi jarang terletak tepat pada satu garis lurus. Oleh karena itu, Linear Regression berusaha mencari garis terbaik yang meminimalkan selisih antara nilai prediksi dan nilai aktual. Selisih ini disebut **error** dan menjadi dasar dalam proses optimasi parameter model.

Pendekatan Analitik Linear Regression

Pendekatan analitik menghitung parameter regresi secara langsung menggunakan rumus statistik berbasis nilai rata-rata dan operasi penjumlahan (∑\sum∑). Metode ini bersifat deterministik dan efisien untuk regresi satu variabel dengan ukuran data kecil.

Namun, pendekatan analitik memiliki keterbatasan ketika diterapkan pada:

- Data berdimensi tinggi
- Dataset berukuran besar
- Model dengan struktur kompleks

Kondisi tersebut mendorong penggunaan pendekatan numerik.

Implementasi Regresi Linear dengan NumPy

NumPy digunakan dalam implementasi regresi linear karena mendukung operasi vektor dan matriks secara efisien. Data dari file CSV dikonversi menjadi **NumPy array**, kemudian dipisahkan menjadi variabel input dan output.

Keunggulan pendekatan ini meliputi:

- Tidak memerlukan *looping* eksplisit
- Mendukung *vectorized computation*
- Lebih cepat dan ringkas secara sintaks

Dengan NumPy, proses prediksi dan perhitungan error dapat dilakukan secara simultan terhadap seluruh data.

Perbedaan List, Array, dan NumPy Array

Struktur data memiliki pengaruh signifikan terhadap performa komputasi numerik:

- **List (Python)** bersifat dinamis dan dapat menyimpan berbagai tipe data
- **Array (Java)** bersifat statis dan hanya menyimpan satu tipe data
- **NumPy array** bersifat homogen dan mendukung operasi matematis langsung

NumPy array menjadi pilihan utama dalam machine learning karena efisiensinya dalam komputasi numerik dan dukungan operasi matriks.

Analitik vs Numerik

Terdapat dua pendekatan utama dalam menyelesaikan masalah regresi:

1. **Pendekatan Analitik**
   Menggunakan rumus matematis yang bersifat langsung dan deterministik.

2. **Pendekatan Numerik**
   Menggunakan iterasi dan optimasi bertahap untuk mendekati solusi optimal.

Pendekatan numerik menjadi fondasi bagi algoritma optimasi seperti Gradient Descent.

Stochastic Gradient Descent (SGD)

SGD merupakan metode optimasi numerik yang bekerja dengan memperbarui parameter model secara iteratif berdasarkan error prediksi. Proses dimulai dengan inisialisasi parameter secara acak, kemudian dilakukan pembaruan parameter pada setiap data dan setiap epoch.

Karakteristik utama SGD meliputi:

- **Epoch** merepresentasikan jumlah iterasi terhadap seluruh dataset
- **Learning rate** menentukan besar langkah pembaruan parameter
- Semakin banyak epoch, model berpotensi semakin mendekati solusi optimal

Namun, pemilihan epoch dan learning rate yang tidak tepat dapat menyebabkan model tidak konvergen atau mengalami overfitting.

Pengaruh Epoch terhadap Model

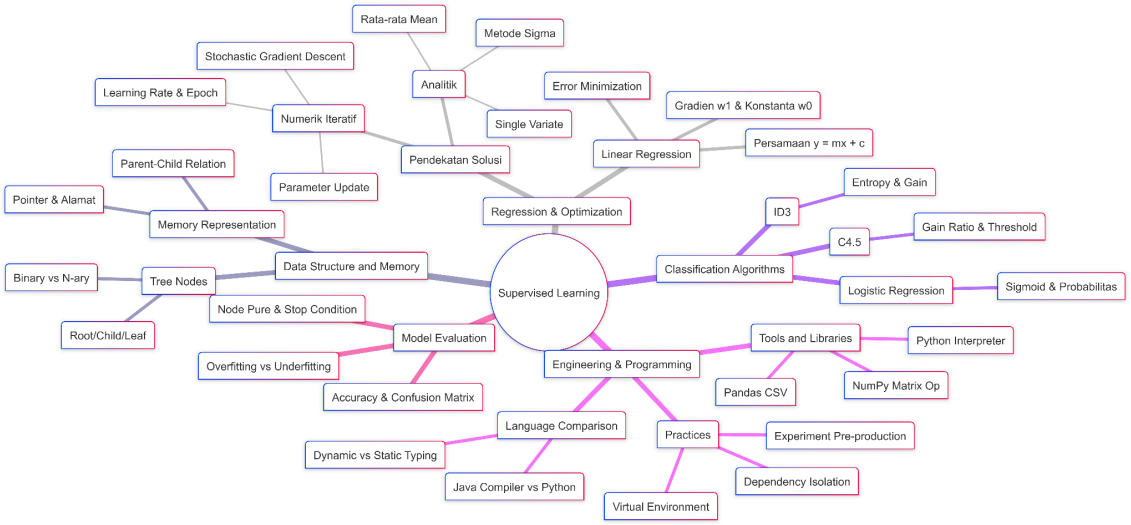
Eksperimen menunjukkan bahwa jumlah epoch sangat memengaruhi hasil regresi:

- Epoch kecil menghasilkan model yang belum optimal
- Epoch menengah menghasilkan garis regresi terbaik
- Epoch terlalu besar berpotensi menurunkan performa model

Hal ini menunjukkan bahwa optimasi numerik bersifat eksperimental dan memerlukan evaluasi berulang.

3.7.3 Mind Map / Taksonomi Konseptual

Struktur konseptual pertemuan ketujuh dapat dirangkum sebagai berikut:



3.7.4 Pseudocode Regresi Linear Analitik

```
HitungGradien(X, Y):
    m = (n * sum(X*Y) - sum(X) * sum(Y)) /
        (n * sum(X^2) - (sum(X))^2)
    return m

HitungKonstanta(X, Y, m):
    c = mean(Y) - m * mean(X)
    return c
```

3.7.5 Pseudocode Prediksi Linear Regression

```
Prediksi(x, m, c):
    return m * x + c
```

3.7.6 Pseudocode Stochastic Gradient Descent

```
SGD(X, Y, learning_rate, max_epoch):
    inisialisasi m dan c secara acak

    for epoch dari 1 sampai max_epoch:
        for setiap data (xi, yi):
            y_pred = m * xi + c
            error = y_pred - yi

            grad_m = error * xi
            grad_c = error

            m = m - learning_rate * grad_m
            c = c - learning_rate * grad_c

    return m, c
```

3.7.7 Rumus Matematis (LaTeX)

Model Linear Regression:

$$\hat{y} = w1x + w0$$

Fungsi Error (Mean Squared Error):

$$MSE = \frac{1}{n} = \sum_{i=1}^{n} (yi - \hat{y}i)2$$

Update Parameter pada SGD:

$$w1 = w1 - \alpha \frac{\partial MSE}{\partial w1}$$

$$w0 = w0 - \alpha \frac{\partial MSE}{\partial w0}$$

3.8 Pertemuan 8 – Paradigma Pemrograman dan Dasar Fundamental Machine Learning

3.8.1 Deskripsi Singkat

Pertemuan kedelapan membahas pergeseran paradigma dari pemrograman konvensional menuju Machine Learning (ML) sebagai pendekatan baru dalam penyelesaian masalah komputasi yang kompleks. Materi menekankan perbedaan mendasar antara pemrograman eksplisit dan pembelajaran mesin, di mana ML memungkinkan sistem untuk mengekstraksi aturan secara mandiri dari data. Fokus utama pertemuan ini adalah pemahaman terhadap komponen fundamental Machine Learning, yaitu Data, Aturan (Rules), dan Jawaban (Answers), serta hubungan ketiganya dalam berbagai skema pembelajaran.

3.8.2 Penjelasan Detail

Paradigma Pemrograman Konvensional dan Machine Learning

Dalam pemrograman tradisional, pengembang harus mendefinisikan seluruh aturan secara eksplisit untuk memperoleh output tertentu. Pendekatan ini efektif untuk permasalahan terstruktur, namun menjadi kurang optimal ketika menghadapi sistem yang kompleks, dinamis, dan memiliki banyak variabel. Machine Learning hadir sebagai solusi dengan membalik alur kerja pemrograman, yaitu dengan menyediakan data dan jawaban, kemudian sistem secara mandiri mempelajari pola atau aturan yang menghubungkan keduanya. Pendekatan ini menjadi fondasi utama pengembangan Kecerdasan Buatan modern.

Metodologi dan Paradigma dalam Machine Learning

Metodologi Machine Learning melibatkan pemahaman mengenai bagaimana data direpresentasikan, diproses, dan digunakan oleh algoritma untuk membentuk model prediktif. Data direpresentasikan dalam bentuk vektor fitur, sementara aturan dihasilkan melalui proses training berdasarkan hubungan antara data input dan jawaban yang diberikan.

Perbandingan Paradigma Pemrograman

Secara konseptual, perbedaan antara pemrograman konvensional dan Machine Learning dapat dijelaskan

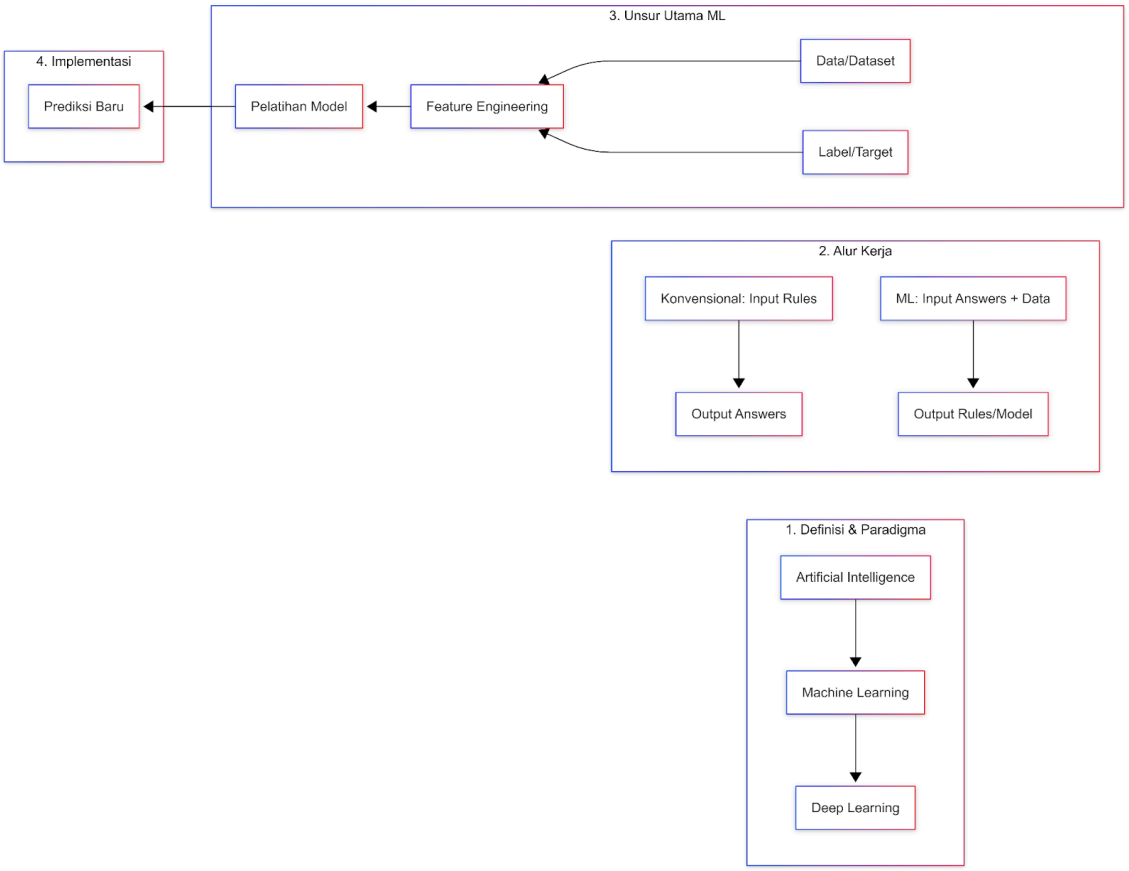
sebagai berikut:

- Pemrograman Konvensional: Data dan aturan diberikan secara eksplisit untuk menghasilkan jawaban.
- Machine Learning: Data dan jawaban diberikan untuk menghasilkan aturan atau model secara otomatis.

Pendekatan ini memungkinkan sistem Machine Learning beradaptasi terhadap data baru dan pola yang sebelumnya tidak didefinisikan secara manual.

3.8.3 Mind Map / Taksonomi Materi dan Alur Belajar

Struktur konseptual materi pada pertemuan kedelapan meliputi:



3.8.4 Analisis Skema Pembelajaran Machine Learning

Berdasarkan materi yang dipelajari, Machine Learning dibagi ke dalam beberapa kategori utama, antara lain:

- Supervised Learning, yang memiliki target atau label yang jelas dan bertujuan untuk prediksi serta klasifikasi, seperti deteksi spam dan prediksi harga.

- Unsupervised Learning, yang tidak memiliki target dan bertujuan untuk pengelompokan data (clustering), seperti segmentasi pelanggan.

3.8.5 Analisis Komparatif Skema Pembelajaran

Dalam implementasinya, ML dibagi menjadi beberapa kategori utama:

Kriteria	**Supervised Learning**	**Unsupervised Learning**
Label (Y)	Memiliki Target yang Jelas	Tidak Memiliki Target
Tujuan	Prediksi dan Klasifikasi	Pengelompokan (Clustering)
Contoh Kasus	Deteksi Spam, Prediksi Harga	Segmentasi Pelanggan

3.8.6 Implementasi Dasar (Pseudocode & Python)

Untuk memahami bagaimana "Aturan" dicari secara otomatis, berikut adalah logika algoritma (pseudocode) dan implementasi teknisnya:

Pseudocode: Pelatihan Model Linear

```
BEGIN
  SET dataset_input = [-1.0, 0.0, 1.0, 2.0, 3.0, 4.0]
  SET dataset_target = [-3.0, -1.0, 1.0, 3.0, 5.0, 7.0]

  CREATE model WITH 1 neuron
  DEFINE optimizer AS 'stochastic gradient descent'
  DEFINE loss_function AS 'mean squared error'

  FOR epoch FROM 1 TO 500
      TRAIN model USING dataset_input AND dataset_target
  END FOR

  PREDICT value FOR input 10.0
  DISPLAY prediction
END
```

Sample Code: Implementasi Python

```
import numpy as np
import tensorflow as tf
from tensorflow.keras import layers
```

```
# 1. Menyiapkan Data dan Jawaban (Paradigma ML)
xs = np.array([-1.0, 0.0, 1.0, 2.0, 3.0, 4.0], dtype=float)
ys = np.array([-3.0, -1.0, 1.0, 3.0, 5.0, 7.0], dtype=float)

# 2. Membangun Model Sederhana (1 Neuron)
model = tf.keras.Sequential([
    layers.Dense(units=1, input_shape=[1])
])

# 3. Kompilasi Model
model.compile(optimizer='sgd', loss='mean_squared_error')

# 4. Proses Training (Mencari Aturan)
model.fit(xs, ys, epochs=500, verbose=0)

# 5. Prediksi (Menguji Aturan yang ditemukan)
print("Prediksi untuk x=10 adalah:", model.predict([10.0]))
```

3.9 Pertemuan 9 – Analisis Multivariat dan Implementasi Neural Network

3.9.1 Deskripsi Singkat

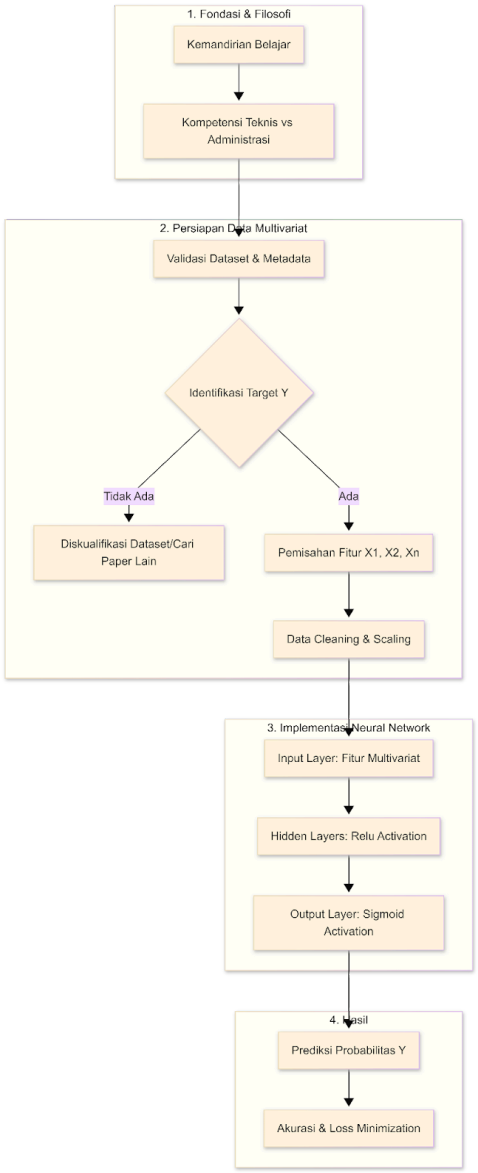
Pertemuan kesembilan membahas pentingnya tahap pra-pemrosesan data dan identifikasi variabel target dalam analisis multivariat sebelum implementasi model Neural Network. Melalui studi kasus dataset reservasi hotel, materi menekankan bahwa pemahaman metadata dan penentuan variabel dependen merupakan prasyarat utama keberhasilan model machine learning, khususnya pada klasifikasi biner.

3.9.2 Penjelasan Detail

Analisis multivariat menuntut pemahaman hubungan antar banyak variabel input terhadap satu variabel target. Kesalahan umum yang sering terjadi adalah penggunaan dataset tanpa identifikasi target yang jelas, sehingga model gagal menjalankan fungsinya. Pada Neural Network, kombinasi linear fitur multivariat diolah melalui bobot dan bias, kemudian dipetakan ke fungsi aktivasi non-linear seperti Sigmoid untuk menghasilkan probabilitas keluaran.

3.9.3 Mind Map / Taksonomi Konseptual

Struktur konseptual pertemuan ini meliputi:



3.9.4 Identifikasi Dataset

Studi kasus menggunakan dataset reservasi hotel dengan struktur multivariat sebagai berikut:

Variabel	Tipe	Peran	Deskripsi
LeadTime	Numerik	Fitur (X_1)	Jarak waktu pemesanan
StayDuration	Numerik	Fitur (X_2)	Durasi menginap

AdultCount	Numerik	Fitur (X_3)	Jumlah tamu dewasa
IsCanceled	Biner	**Target (**Y**)**	Status pembatalan (0/1)

3.9.5 Analisis Komparatif

Perbedaan antara pendekatan tradisional dan *Neural Network* dirangkum dalam tabel di bawah ini:

Kriteria	**Statistik Tradisional**	**Neural Network**
Jumlah Variabel	Terbatas (Univariat)	Masif (Multivariat)
Pola Data	Linear	Non-Linear & Kompleks
Sifat Model	Transparan	*Black Box*

3.9.6 Implementasi Kode Program

Implementasi dilakukan menggunakan pustaka TensorFlow dan Scikit-Learn untuk memastikan data siap diproses.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
import tensorflow as tf
from tensorflow.keras import layers

# Pembersihan dan Pembagian Data
df = pd.read_csv('hotel_data.csv').dropna()
X = df[['LeadTime', 'StayDuration', 'AdultCount']].values
y = df['IsCanceled'].values

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)

# Normalisasi Fitur
scaler = StandardScaler()
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)

# Konstruksi Model Neural Network
model = tf.keras.Sequential([
    layers.Dense(16, activation='relu', input_shape=(3,)),
    layers.Dense(8, activation='relu'),
    layers.Dense(1, activation='sigmoid')
])
```

model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

3.10 Pertemuan 10 – Analisis dan Implementasi Algoritma Komputasi

3.10.1 Deskripsi Singkat

Pertemuan kesepuluh membahas konsep analisis algoritma dengan fokus pada efisiensi waktu dan penggunaan memori. Materi ini menekankan pentingnya pemilihan algoritma yang tepat berdasarkan karakteristik data dan skala permasalahan. Selain itu, digunakan notasi matematika formal untuk merepresentasikan beban komputasi serta pseudocode dan implementasi program sebagai bentuk penerapan praktis.

3.10.2 Penjelasan Detail

Analisis algoritma merupakan bagian fundamental dalam ilmu komputer, karena sebuah program yang menghasilkan keluaran benar belum tentu efisien secara komputasi. Efisiensi diukur melalui kompleksitas waktu dan ruang yang menunjukkan bagaimana sumber daya komputasi bertumbuh seiring bertambahnya ukuran data masukan. Dalam konteks sistem berskala besar, perbedaan kompleksitas algoritma dapat berdampak signifikan terhadap performa dan stabilitas sistem.

3.10.3 Notasi Matematis (LaTeX)

Dalam dunia informatika, efisiensi adalah parameter krusial. Dosen menekankan bahwa sebuah program yang "berhasil jalan" belum tentu merupakan program yang "baik". Perbedaan antara algoritma $O(n)$ dan $O(n^2)$ dapat menentukan kelangsungan operasional sebuah sistem berskala besar.

Berikut adalah representasi formal dari konsep yang dibahas di kelas:

2.10.3.1 Kompleksitas Waktu (Big O Notation)

Definisi formal kompleksitas waktu dinyatakan sebagai:

$$f(n) = O(g(n)) \iff \exists c > 0, n_0 > 0 \text{ s.t. } 0 \leq f(n) \leq c \cdot g(n), \forall n \geq n_0$$

3.10.3.2 Fungsi Rekursif (Master Theorem)

Untuk algoritma *divide and conquer*, hubungan rekurensi dinyatakan dengan:

$$T(n) = aT\left(\frac{n}{b}\right) + f(n)$$

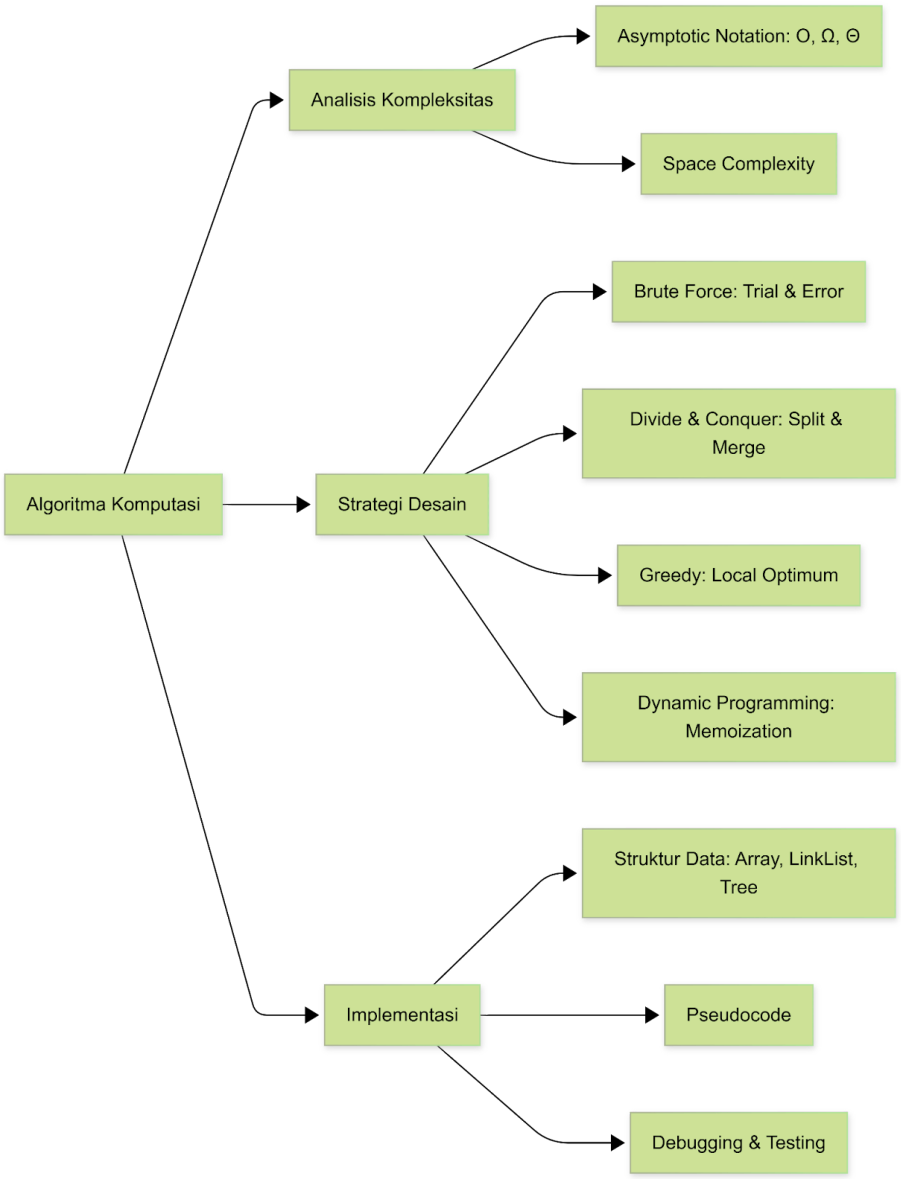
Dimana a adalah jumlah sub-masalah, b adalah faktor pembagi, dan $f(n)$ adalah biaya overhead.

3.10.4 TABEL PERBANDINGAN KOMPLEKSITAS

Notasi	**Nama**	**Contoh Algoritma**	**Skalabilitas**
$O(1)$	Constant	Mengakses elemen Array	Sangat Baik
$O(\log n)$	Logarithmic	Binary Search	Sangat Baik
$O(n)$	Linear	Linear Search	Baik
$O(n \log n)$	Linearithmic	Merge Sort, Quick Sort	Cukup
$O(n^2)$	Quadratic	Bubble Sort, Insertion Sort	Buruk (pada n besar)
$O(2^n)$	Exponential	Rekursi Fibonacci Naif	Sangat Buruk

3.10.5 TAKSONOMI & MINDMAP (MERMAID)

Hierarki strategi penyelesaian masalah menurut penjelasan dosen:



3.10.6 PENJELASAN DETAIL

3.10.6.1 Analisis Kasus (Best, Average, Worst Case)

- **Worst Case (O):** Menjamin batas atas waktu eksekusi. Skenario terburuk yang mungkin terjadi.
- **Best Case (Ω):** Skenario tercepat, biasanya terjadi saat input sudah dalam keadaan ideal (misal: sudah terurut).
- **Average Case (Θ):** Ekspektasi rata-rata penggunaan sumber daya dalam kondisi umum.

3.10.6.2 Trade-off Ruang dan Waktu

Dosen menjelaskan bahwa dalam komputasi sering kali terdapat pertukaran antara memori dan kecepatan. Menggunakan memori tambahan (caching/memoization) seringkali dapat memangkas waktu proses

secara signifikan.

3.10.8 SAMPLE CODE & PSEUDOCODE

3.10.8.1. Pseudocode: Merge Sort (Divide & Conquer)

```
ALGORITHM MergeSort(A[0..n-1])
  // Input: Array A dengan n elemen
  // Output: Array A yang terurut secara ascending
  IF n > 1
      copy A[0..floor(n/2)-1] to B[0..floor(n/2)-1]
      copy A[floor(n/2)..n-1] to C[0..ceil(n/2)-1]
      MergeSort(B)
      MergeSort(C)
      Merge(B, C, A)
```

3.10.8.2. Implementasi Python: Binary Search

```
def binary_search(arr, target):
  low = 0
  high = len(arr) - 1

  while low <= high:
      mid = (low + high) // 2
      if arr[mid] == target:
          return mid
      elif arr[mid] < target:
          low = mid + 1
      else:
          high = mid - 1

  return -1 # Tidak ditemukan

# Contoh penggunaan
data_terurut = [2, 5, 8, 12, 16, 23, 38, 56, 72, 91]
hasil = binary_search(data_terurut, 23)
print(f"Elemen ditemukan pada indeks: {hasil}")
```

3.11 Pertemuan 11 – Neural Networks dan Deep Learning

3.11.1 Deskripsi Singkat

Pertemuan kesebelas membahas transisi dari model regresi linear menuju arsitektur Neural Networks (NN) dan pengenalan awal Deep Learning. Fokus utama materi meliputi struktur neuron univariate dan multivariate, konsep bobot (*weight*), bias, fungsi aktivasi, serta perkembangan jaringan saraf dari model sederhana hingga arsitektur multi-layer yang lebih kompleks.

3.11.2 Penjelasan Detail

Neural Network merupakan model komputasi yang terinspirasi dari sistem saraf biologis manusia. Secara matematis, NN dapat dipandang sebagai fungsi yang memetakan input ke output melalui kombinasi transformasi linier dan non-linier. Pemahaman terhadap bobot dan bias menjadi fondasi utama sebelum mempelajari mekanisme optimasi dan pembelajaran lanjutan seperti *backpropagation*.

Pembahasan dimulai dengan penguatan konsep regresi linear sebagai bentuk neuron paling sederhana, kemudian diperluas ke model perceptron dan Multi-Layer Perceptron (MLP). Selanjutnya, diperkenalkan konsep Deep Learning yang memanfaatkan banyak lapisan tersembunyi (*hidden layers*) untuk mempelajari representasi data secara hierarkis.

3.11.3 NOTASI MATEMATIKA (LATEX)

Berikut adalah formulasi matematis yang digunakan untuk merepresentasikan neuron tunggal:

3.11.3.1. Penjumlahan Terbobot (Weighted Sum)

Output dasar dari sebuah neuron sebelum fungsi aktivasi adalah:

$$z = \sum_{i=0}^{n} (x_i \cdot w_i) + b$$

Atau dalam bentuk matriks:

$$z = XW + b$$

3.11.3.2. Fungsi Aktivasi (Activation Function)

Untuk memperkenalkan non-linearitas, digunakan fungsi aktivasi (misal Sigmoid):

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

3.11.3.3. Perhitungan Univariate vs Multivariate

- **Univariate:** Hanya memiliki satu input x_1 dan satu bobot w_1.
- **Multivariate:** Memiliki banyak input x_n yang diakumulasikan ke dalam satu unit output.

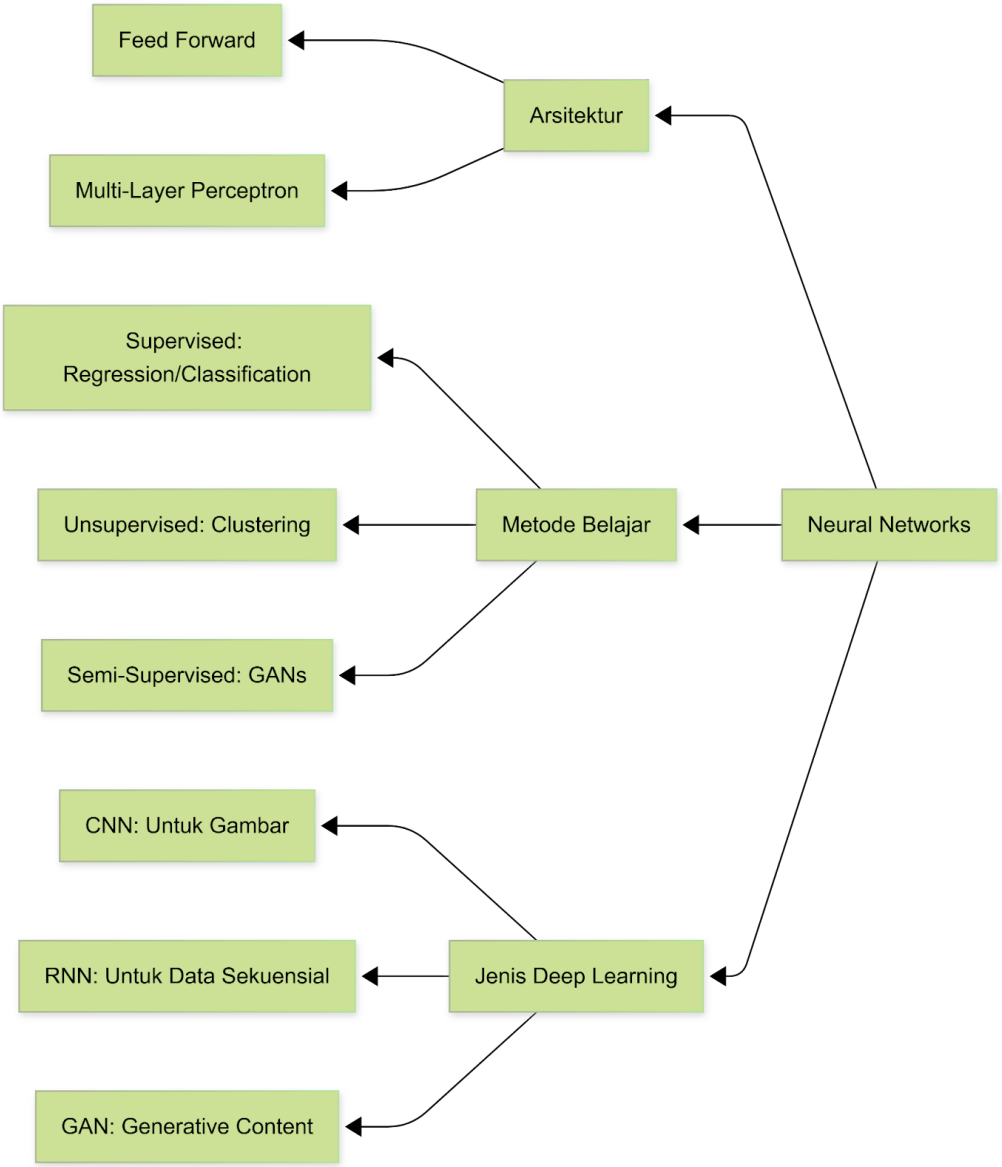
3.11.4. TABEL PERBANDINGAN MODEL

Dosen membandingkan beberapa model yang dibahas berdasarkan kompleksitas dan fungsinya:

Model	Input	Layer	Fungsi Utama

Linear Regression	Tunggal/Banyak	0 (Langsung)	Prediksi angka kontinu
Simple Perceptron	Banyak	1 (Single)	Klasifikasi biner sederhana
Multi-Layer (MLP)	Banyak	> 1 (Hidden)	Klasifikasi masalah non-linear
Deep Learning	Sangat Besar	Banyak (Deep)	Pengenalan pola kompleks (Gambar/Suara)

3.11.5. TAKSONOMI & MINDMAP (MERMAID)



3.11.6 Mekanisme Feed Forward

Pada proses *feed forward*, data mengalir dari *input layer* menuju *output layer*. Setiap neuron menghitung penjumlahan terbobot dari input, menambahkan bias, kemudian menerapkan fungsi aktivasi. Pada jaringan multi-layer, output dari satu lapisan menjadi input bagi lapisan berikutnya.

3.11.7 Deep Learning dan Model Generatif

Perbedaan utama antara Neural Network konvensional dan Deep Learning terletak pada kedalaman jaringan. Deep Learning mampu melakukan *feature extraction* secara otomatis dari data tidak terstruktur seperti citra dan audio. Selain itu, diperkenalkan konsep *Generative Adversarial Networks (GAN)* yang terdiri dari dua jaringan—Generator dan Discriminator—yang dilatih secara kompetitif untuk menghasilkan data baru yang menyerupai data asli.

3.11.8. SAMPLE CODE & PSEUDOCODE

3.11.8.1. Pseudocode: Proses Forward Propagation

```
ALGORITHM ForwardPass(X, W, b)
   // X: Input features
   // W: Weights
   // b: Bias

   z = DOT_PRODUCT(X, W) + b
   output = SIGMOID(z)

   RETURN output
END ALGORITHM
```

3.11.8.2. Implementasi Python (Basic Neuron)

```
import numpy as np

def sigmoid(x):
   return 1 / (1 + np.exp(-x))

def simple_neuron(inputs, weights, bias):
   # Menghitung dot product dan menambah bias
   z = np.dot(inputs, weights) + bias
   return sigmoid(z)

# Data Input (Misal: 3 fitur)
inputs = np.array([1.0, 2.0, 3.0])
weights = np.array([0.2, 0.8, -0.5])
bias = 2.0

output = simple_neuron(inputs, weights, bias)
print(f"Output Neuron: {output:.4f}")
```

3.12 Pertemuan 12 – Manajemen Proyek dan Workflow Machine Learning

3.12.1 Deskripsi Singkat

Pertemuan kedua belas membahas manajemen proyek Machine Learning dengan fokus pada standardisasi luaran akademik untuk UTS dan UAS. Materi menekankan bagaimana mengelola satu proyek penelitian secara berkelanjutan, mulai dari pendekatan Supervised Learning hingga Deep Learning, dalam satu repositori GitHub yang terstruktur dan terdokumentasi dengan baik.

3.12.2 Penjelasan Detail

Dalam pertemuan ini, dosen menekankan bahwa keberhasilan proyek Machine Learning tidak hanya ditentukan oleh performa model, tetapi juga oleh kerapihan workflow, kejelasan hipotesis, serta kemampuan mempertanggungjawabkan pemilihan metode secara ilmiah. Mahasiswa diarahkan untuk menyusun proyek dengan pendekatan saintifik yang selaras dengan format publikasi Jurnal Komputa.

Pendekatan yang dianjurkan adalah menjaga kesinambungan antara UTS dan UAS, di mana studi kasus dan dataset yang sama dapat digunakan untuk membandingkan performa metode klasik dan metode Deep Learning secara objektif.

3.12.3. NOTASI MATEMATIKA

Dalam penyusunan laporan, dosen mewajibkan adanya formalisasi masalah, terutama pada bagian evaluasi dan perbandingan model:

3.12.3.1. Representasi Hipotesis Model

Model yang dibangun harus didefinisikan secara matematis, misalnya pada Regresi atau Klasifikasi Sederhana:

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \cdots + \theta_n x_n$$

3.12.3.2. Fungsi Kerugian (Loss Function)

Mahasiswa harus menjelaskan fungsi optimasi yang digunakan, seperti MSE untuk regresi:

$$J(\theta) = \frac{1}{2m} \sum_{i=1}^{m} (h_\theta(x^{(i)}) - y^{(i)})^2$$

3.12.4. TABEL STRUKTUR LUARAN TUGAS BESAR

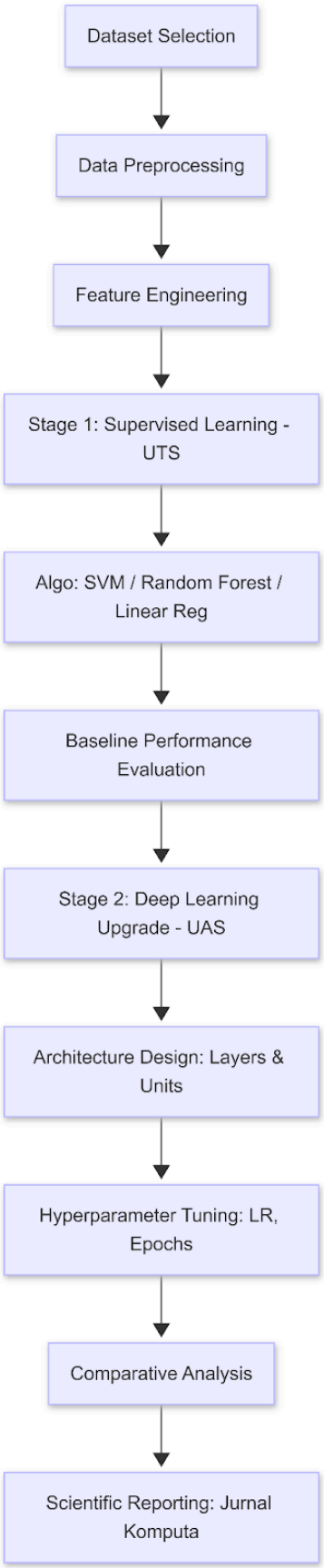
Dosen memberikan rincian folder GitHub yang harus dipenuhi:

Komponen	Folder UTS (Supervised)	Folder UAS (Deep Learning)
Dataset	CSV/Dataset Reference	Updated Dataset (jika ada)
Kode	.ipynb (Supervised)	.ipynb (Deep Learning)
Proposal	Rancangan Awal	Updated Proposal (Hipotesis DL)
Laporan	PDF Jurnal Komputa	PDF Jurnal Komputa (Updated)

Repositori	GitHub Link	GitHub Link (Folder Berbeda)

3.12.5. TAKSONOMI PENGEMBANGAN MODEL (MERMAID)

Berikut adalah alur pengerjaan yang disarankan oleh dosen:


Dataset Selection
Data Preprocessing
Feature Engineering
Stage 1: Supervised Learning - UTS
Algo: SVM / Random Forest / Linear Reg
Baseline Performance Evaluation
Stage 2: Deep Learning Upgrade - UAS
Architecture Design: Layers & Units
Hyperparameter Tuning: LR, Epochs
Comparative Analysis
Scientific Reporting: Jurnal Komputa

3.12.6 Strategi "Satu Paper Berkelanjutan"

Dosen menyarankan strategi "satu paper" apabila studi kasus dan dataset yang digunakan pada UTS dan UAS sama. Dengan pendekatan ini, mahasiswa dapat:

- Membandingkan performa algoritma klasik dan Deep Learning
- Menyajikan analisis eksperimen yang lebih komprehensif
- Menghasilkan satu karya ilmiah yang lebih matang dan siap publikasi

3.12.7 Pentingnya Komunitas dan Portofolio

Selain aspek teknis, dosen menekankan pentingnya keaktifan mahasiswa dalam komunitas teknologi seperti PyCon (Python Conference). Keterlibatan dalam komunitas dinilai dapat memperluas wawasan, meningkatkan kualitas portofolio, serta membuka peluang karier di industri teknologi.

berkenalan dengan praktisi senior.

3.12.8. SAMPLE CODE & PSEUDOCODE

3.12.8.1. Pseudocode: Struktur Folder Otomatis

```
ALGORITHM SetupProject()
   CREATE_DIRECTORY "Project_AI"
   CREATE_DIRECTORY "Project_AI/UTS"
   CREATE_DIRECTORY "Project_AI/UAS"

   // Inisialisasi Git
   RUN "git init"
   CREATE_FILE ".gitignore"

   PRINT "Struktur proyek siap untuk dipush ke GitHub"
END ALGORITHM
```

3.13 Pertemuan 13 – Optimasi Parameter dan Etika Profesional dalam Machine Learning

3.13.1 Deskripsi Singkat

Pertemuan ketiga belas membahas optimasi parameter pada model Machine Learning serta pentingnya etika dan komunikasi profesional dalam praktik kecerdasan buatan. Fokus utama diarahkan pada pemahaman teori di balik pemilihan parameter, khususnya *learning rate* dan mekanisme konvergensi, sebagai kompetensi wajib bagi seorang Machine Learning engineer profesional.

3.13.2 Penjelasan Detail

Pada sesi ini, dosen menyoroti permasalahan umum yang sering terjadi pada mahasiswa, yaitu kecenderungan menggunakan parameter default atau menyalin kode dari tutorial tanpa memahami dasar matematisnya. Melalui studi kasus dan simulasi diskusi profesional, mahasiswa diajak memahami bahwa

setiap nilai parameter harus dapat dipertanggungjawabkan secara teoritis, terutama saat berhadapan dengan klien atau atasan yang memiliki latar belakang teknis yang kuat.

Diskusi diawali dengan review tugas kelompok terkait algoritma Support Vector Machine (SVM), kemudian dilanjutkan dengan sesi tanya jawab mendalam mengenai alasan pemilihan nilai parameter tertentu. Pendekatan ini bertujuan melatih mahasiswa agar mampu menjelaskan model tidak hanya dari sisi implementasi, tetapi juga dari sudut pandang matematis dan etika profesional.

3.13.3. NOTASI MATEMATIKA

Dosen menjelaskan logika di balik pembaruan bobot yang sering menjadi pertanyaan klien:

3.13.3.1. Mekanisme Gradient Descent

Perubahan bobot ditentukan oleh gradien error terhadap bobot tersebut:

$$w_{t+1} = w_t - \eta \cdot \nabla J(w_t)$$

Di mana η (atau α) adalah **Learning Rate**. Jika η terlalu besar, model akan berosilasi; jika terlalu kecil, model akan lambat.

3.13.3.2. Parameter SVM (Contoh Kasus)

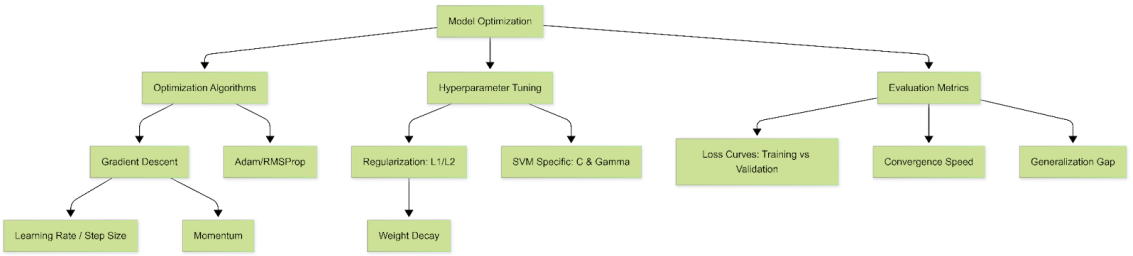
Dalam SVM, terdapat parameter C (Regularization) yang menyeimbangkan antara margin lebar dan kesalahan klasifikasi:

$$\min \frac{1}{2}\|w\|^2 + C\sum \xi_i$$

3.13.4. TABEL ANALISIS PARAMETER

Parameter	**Fungsi Utama**	**Dampak Jika Terlalu Tinggi**
Learning Rate	Kecepatan penyesuaian bobot	Model tidak pernah konvergen (divergen)
Epochs	Jumlah iterasi seluruh dataset	Berisiko terjadi *Overfitting*
Batch Size	Jumlah data per update bobot	Beban memori tinggi, proses lambat
C (SVM)	Toleransi kesalahan	Model menjadi terlalu sensitif (*Overfit*)

3.13.5. TAKSONOMI KOMUNIKASI TEKNIS (MERMAID)



3.13.6 Studi Kasus: Menghadapi VP Data

Dosen memberikan ilustrasi kasus ketika seorang engineer dihadapkan pada pertanyaan kritis seperti: *"Mengapa learning rate yang digunakan bernilai 0.01?"*. Jawaban yang dapat diterima bukanlah berdasarkan tutorial, melainkan berdasarkan analisis kestabilan proses pelatihan, seperti pola konvergensi fungsi kerugian dan perilaku gradien.

3.13.7. SAMPLE CODE & PSEUDOCODE

3.13.7.1. Implementasi Penyesuaian Parameter (Python)

```
from sklearn.svm import SVC

# Contoh setting parameter manual berdasarkan teori
model = SVC(C=1.0, kernel='rbf', gamma='scale')

# Penjelasan:
# C=1.0 adalah default, tapi kita bisa naikkan jika ingin
# model lebih ketat terhadap kesalahan.
model.fit(X_train, y_train)
```

3.14 Pertemuan 14 – Analisis Metode Clustering: K-Means dan Gaussian Mixture Model

3.14.1 Deskripsi Singkat

Pertemuan keempat belas membahas teknik *unsupervised learning* khususnya metode clustering, dengan fokus pada perbandingan antara algoritma K-Means dan Gaussian Mixture Model (GMM). Pembahasan menekankan perbedaan paradigma pengelompokan berbasis jarak dan berbasis probabilitas serta implikasinya terhadap karakteristik data.

3.14.2 Penjelasan Umum

Clustering merupakan metode pembelajaran tanpa label yang bertujuan mengelompokkan data berdasarkan kesamaan karakteristik internal. Dalam sesi ini, dosen menggunakan pendekatan konseptual dan visual untuk menjelaskan bagaimana struktur alami data dapat diidentifikasi melalui pola kepadatan dan jarak. Analogi "gunung" pada histogram digunakan untuk membantu mahasiswa memahami keberadaan kelompok data yang terbentuk secara alami.

3.14.3. NOTASI MATEMATIKA (LATEX)

3.14.3.1. Objektif K-Means (Minimize Inertia)

Tujuan K-Means adalah meminimalkan jumlah kuadrat jarak antara titik data x dan pusat cluster c:

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \|x_i^{(j)} - c_j\|^2$$

3.14.3.2. Fungsi Densitas Probabilitas GMM (Gaussian)

GMM mengasumsikan data berasal dari campuran beberapa distribusi Gaussian:

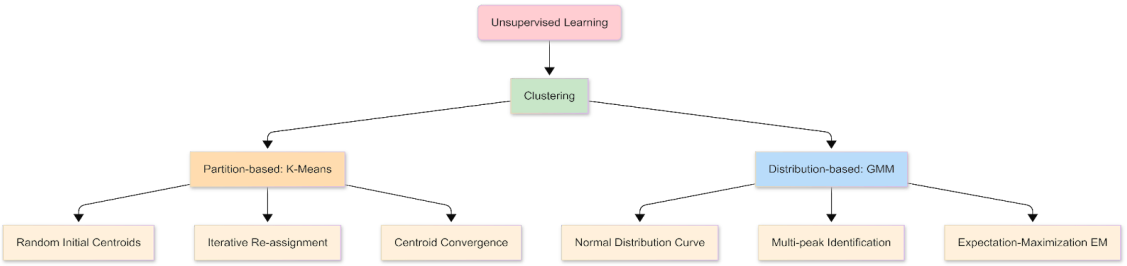
$$P(x) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x|\mu_k, \Sigma_k)$$

Dimana π_k adalah bobot pencampuran, μ_k adalah rata-rata, dan Σ_k adalah kovarians.

3.14.4. TABEL PERBANDINGAN CLUSTERING MENDALAM

Fitur	**K-Means**	**Gaussian Mixture Model (GMM)**
Logika Dasar	Jarak Euclidean (Pusat Massa)	Distribusi Probabilitas (Statistik)
Jenis Kelompok	*Hard Clustering* (Data masuk 1 kelompok)	*Soft Clustering* (Probabilitas masuk kelompok)
Bentuk Geometri	Cenderung bulat/lingkaran	Bisa elips, miring, atau memanjang
Kelebihan	Sangat cepat dan simpel	Akurat untuk data yang tumpang tindih
Kekurangan	Sensitif terhadap pencilan (*outliers*)	Lebih berat secara komputasi

3.14.5. TAKSONOMI CLUSTERING (MERMAID)



3.14.6 Pembahasan Teknis

3.14.6.1 Mekanisme Pergerakan Centroid pada K-Means

Proses K-Means berlangsung secara iteratif:

1. Inisialisasi centroid secara acak
2. Penentuan cluster berdasarkan jarak terdekat
3. Pembaruan posisi centroid
4. Proses berhenti saat konvergensi tercapai

Metode ini efektif namun rentan terhadap *outliers* dan inisialisasi awal.

3.14.6.2 Analogi “Gunung” pada GMM

GMM bekerja dengan mengidentifikasi kepadatan data melalui distribusi Gaussian. Ketika data memiliki beberapa puncak frekuensi (misalnya nilai ujian 60 dan 90), GMM mampu memodelkan masing-masing puncak sebagai distribusi terpisah, bahkan ketika saling tumpang tindih secara spasial.

3.14.7. SAMPLE CODE & PSEUDOCODE

3.14.7.1. Pseudocode: K-Means yang Disempurnakan

```
ALGORITHM KMeans(Dataset D, integer k)
  Initialize k centroids (C1, C2, ..., Ck) randomly
  WHILE centroids are changing DO:
    FOR each point x in D:
      Assign x to cluster j where distance(x, Cj) is minimum
    FOR each cluster j from 1 to k:
      Cj = mean of all points assigned to cluster j
  RETURN centroids and assignments
```

3.14.7.2. Implementasi Python: GMM dengan Visualisasi Konsep

```
from sklearn.mixture import GaussianMixture
import numpy as np
```

```
# Membuat data buatan (Synthetic Data)
# Cluster 1: Mean=0, Cluster 2: Mean=5
data = np.concatenate([
    np.random.normal(0, 1, 300),
    np.random.normal(5, 1.5, 300)
]).reshape(-1, 1)

# Fitting GMM
gmm = GaussianMixture(n_components=2, covariance_type='full')
gmm.fit(data)

# Output parameter yang dipelajari model
for i in range(2):
    print(f"Cluster {i+1}: Mean = {gmm.means_[i][0]:.2f}, Var = {gmm.covariances_[i][0][0]:.2f}")
```

- **Kesimpulan**

Berdasarkan tinjauan materi perkuliahan dari pertemuan pertama hingga ke-14, dapat disimpulkan bahwa *Machine Learning* adalah perpaduan harmonis antara statistika, struktur data, dan optimasi matematika. Pembelajaran dimulai dengan pemahaman dasar mengenai representasi data dan paradigma induktif, yang kemudian berkembang menjadi teknik klasifikasi kompleks menggunakan algoritma pohon keputusan. Penggunaan algoritma ID3 dan penyempurnaannya pada C4.5 menunjukkan pentingnya metrik seperti *Information Gain* dan *Gain Ratio* dalam menangani variasi tipe data (diskrit dan numerik).

Lebih lanjut, transisi menuju *Linear Regression* dan optimasi numerik melalui *Stochastic Gradient Descent* (SGD) menegaskan bahwa model ML yang modern sangat bergantung pada kemampuan mesin untuk melakukan iterasi pembaruan parameter guna meminimalkan error secara efisien. Secara teknis, penggunaan Python sebagai bahasa utama terbukti efektif karena sifatnya yang dinamis dan didukung oleh pustaka pemrosesan matriks yang kuat. Praktik *engineering* seperti penggunaan *virtual environment* tetap menjadi standar wajib untuk memastikan isolasi dependensi dan reproduksibilitas model dalam lingkungan produksi.

Daftar Pustaka

[1] T. M. Mitchell, *Machine Learning*, New York: McGraw-Hill, 1997.

[2] E. Alpaydin, *Introduction to Machine Learning*, 4th ed. Cambridge, MA: MIT Press, 2020.

[3] Nurhayati, S.; and Immanudin, I., “Penerapan Logika Fuzzy Mamdani untuk Prediksi Pengadaan Peralatan Rumah Tangga Rumah Sakit,” *Komputika: Jurnal Sistem Komputer*, vol. 8, no. 2, 2019.

[4] S. Wiyono and T. Abidin, "Comparative Study of KNN, SVM and Decision Tree for Student Performance Evaluation," *Jurnal Ilmiah Komputa*, 2019.

[5] J. Brownlee, *Optimization for Machine Learning*, Machine Learning Mastery, 2021.

[6] STMIK Tazkia, *Modul Perkuliahan Machine Learning Pertemuan 1-14*, Bogor: STMIK Tazkia, 2024.