**Penyelesaian program PKM**

Untuk mengembangkan perangkat lunak “**Sistem Deteksi Dini Kesulitan Belajar Siswa Berbasis Artificial Intelligence Computing Platform**” dengan Python, kita memerlukan beberapa pustaka (libraries) yang mendukung analisis data, pembelajaran mesin, dan tampilan antarmuka sederhana. Berikut adalah beberapa pustaka yang dibutuhkan dan langkah-langkah untuk menginstalnya, serta contoh kode untuk membuat model sederhana.

**Library yang Dibutuhkan**

1. **Pandas**: Untuk mengelola dan memanipulasi dataset siswa.
2. **Scikit-Learn**: Untuk algoritma machine learning yang akan menganalisis data dan mendeteksi pola kesulitan belajar.
3. **TensorFlow atau PyTorch (Opsional)**: Untuk model machine learning yang lebih kompleks jika diperlukan.
4. **Matplotlib dan Seaborn**: Untuk visualisasi data.
5. **Flask atau Streamlit**: Untuk antarmuka sederhana yang memungkinkan pengguna (guru) mengakses laporan atau hasil analisis.

**Cara Install Library**

Buka terminal atau command prompt, dan jalankan perintah berikut untuk menginstal pustaka yang diperlukan:

bash

Copy code

pip install pandas scikit-learn matplotlib seaborn flask streamlit

Jika Anda memerlukan model lebih kompleks, Anda dapat menginstal TensorFlow atau PyTorch:

bash

Copy code

pip install tensorflow # Untuk TensorFlow

pip install torch # Untuk PyTorch

**Contoh Program Python untuk Deteksi Dini Kesulitan Belajar Siswa**

Program berikut adalah contoh sederhana menggunakan algoritma **Decision Tree** dari Scikit-Learn untuk mendeteksi siswa yang mungkin mengalami kesulitan belajar berdasarkan beberapa parameter (contoh: nilai ujian, kehadiran, waktu belajar).

**1. Menyiapkan Dataset Contoh**

Misalkan dataset Anda memiliki beberapa kolom seperti nilai\_ujian, kehadiran, waktu\_belajar, dan label\_kesulitan (1 jika siswa mengalami kesulitan belajar, 0 jika tidak).

**2. Membuat Program Python**

python

Copy code

# Import pustaka yang diperlukan

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy\_score, classification\_report

import matplotlib.pyplot as plt

import seaborn as sns

# 1. Membuat Dataset Sederhana

data = {

'nilai\_ujian': [65, 70, 50, 40, 85, 78, 55, 45, 90, 80],

'kehadiran': [80, 85, 60, 50, 95, 88, 70, 60, 98, 92],

'waktu\_belajar': [3, 4, 2, 1, 5, 4, 2, 1, 5, 4],

'label\_kesulitan': [0, 0, 1, 1, 0, 0, 1, 1, 0, 0]

}

df = pd.DataFrame(data)

# 2. Memisahkan Fitur dan Label

X = df[['nilai\_ujian', 'kehadiran', 'waktu\_belajar']]

y = df['label\_kesulitan']

# 3. Split Data menjadi Training dan Testing

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.3, random\_state=42)

# 4. Membuat dan Melatih Model

model = DecisionTreeClassifier()

model.fit(X\_train, y\_train)

# 5. Prediksi dan Evaluasi Model

y\_pred = model.predict(X\_test)

print("Akurasi:", accuracy\_score(y\_test, y\_pred))

print(classification\_report(y\_test, y\_pred))

# 6. Visualisasi Data

sns.scatterplot(x='nilai\_ujian', y='kehadiran', hue='label\_kesulitan', data=df)

plt.xlabel("Nilai Ujian")

plt.ylabel("Kehadiran")

plt.title("Visualisasi Data Kesulitan Belajar")

plt.show()

**Penjelasan Kode**

1. **Dataset**: Dataset dibuat dengan beberapa kolom fitur (nilai\_ujian, kehadiran, waktu\_belajar) dan label (label\_kesulitan).
2. **Preprocessing Data**: Data dipisahkan menjadi variabel fitur X dan label y.
3. **Split Data**: Data dipisah menjadi data latih (70%) dan data uji (30%) menggunakan train\_test\_split.
4. **Model Training**: DecisionTreeClassifier digunakan sebagai model untuk melatih data latih.
5. **Evaluasi Model**: Program mengevaluasi akurasi model pada data uji dan menampilkan laporan klasifikasi.
6. **Visualisasi**: scatterplot digunakan untuk melihat pola data berdasarkan nilai\_ujian dan kehadiran.

**Membuat Antarmuka Sederhana dengan Streamlit**

Untuk antarmuka yang lebih ramah pengguna, kita bisa menggunakan **Streamlit** untuk menampilkan data dan prediksi.

**Contoh Kode Streamlit**

1. Simpan kode di atas sebagai model.py.
2. Buat file baru bernama app.py dan tambahkan kode berikut:

python

Copy code

import streamlit as st

import pandas as pd

import joblib

# Load model (pastikan model sudah disimpan sebelumnya)

model = joblib.load('decision\_tree\_model.pkl')

st.title("Sistem Deteksi Dini Kesulitan Belajar Siswa")

# Input pengguna

nilai\_ujian = st.number\_input("Masukkan Nilai Ujian:")

kehadiran = st.number\_input("Masukkan Kehadiran (%):")

waktu\_belajar = st.number\_input("Masukkan Waktu Belajar (jam per hari):")

if st.button("Prediksi"):

input\_data = [[nilai\_ujian, kehadiran, waktu\_belajar]]

hasil = model.predict(input\_data)

if hasil[0] == 1:

st.write("Siswa berpotensi mengalami kesulitan belajar.")

else:

st.write("Siswa tidak berpotensi mengalami kesulitan belajar.")

1. Jalankan aplikasi Streamlit:

bash

Copy code

streamlit run app.py

**Langkah Selanjutnya**

1. **Pengembangan Lanjutan**: Anda bisa menggunakan model machine learning lain yang lebih akurat, seperti **Random Forest** atau **Neural Networks**, jika datanya lebih besar dan kompleks.
2. **Penyimpanan Model**: Simpan model dengan joblib atau pickle setelah pelatihan, lalu gunakan model tersebut di aplikasi antarmuka.
3. **Integrasi dengan Database**: Integrasikan data dari sistem sekolah untuk otomatisasi lebih lanjut dan peningkatan prediksi.

Dengan pendekatan ini, perangkat lunak ini dapat memberikan deteksi dini kesulitan belajar siswa dan mendukung guru dalam merancang intervensi yang lebih efektif.

HASIL EKSEKUSI Program:

C:\Users\ASUS\AppData\Roaming\Microsoft\Windows\Start Menu\Programs\Python 3.11>python PKM\_2024.py

Akurasi: 1.0

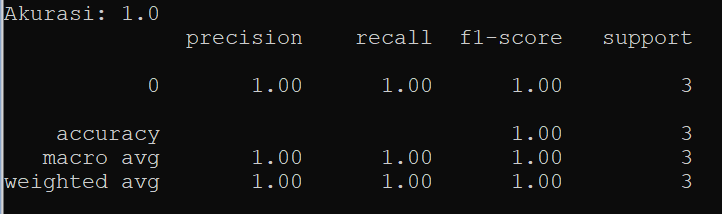
precision recall f1-score support

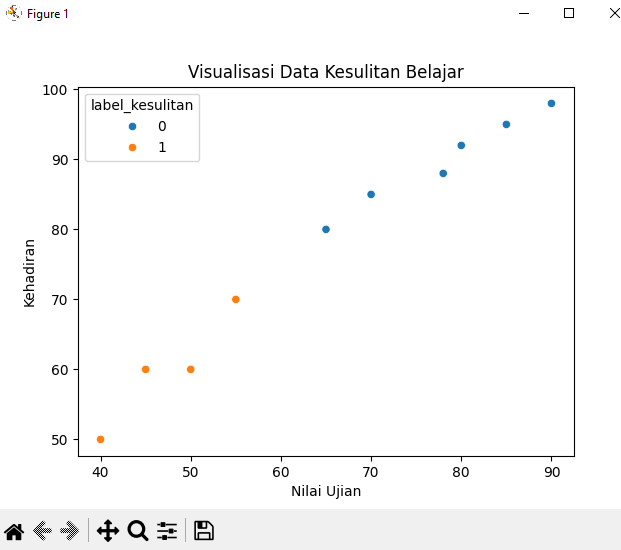
0 1.00 1.00 1.00 3

accuracy 1.00 3

macro avg 1.00 1.00 1.00 3

weighted avg 1.00 1.00 1.00 3





Pembahasan Hasil:

Hasil evaluasi yang Anda sajikan menunjukkan kinerja model dengan akurasi sempurna (1.0 atau 100%) pada sistem deteksi dini kesulitan belajar siswa. Mari kita analisis setiap metrik dan hubungannya dengan konsep Pengembangan Perangkat Lunak Sistem Deteksi Dini Kesulitan Belajar Siswa berbasis Artificial Intelligence Computing Platform:

1. **Akurasi (Accuracy): 1.0**
   * Akurasi 1.0 berarti bahwa model ini dapat mengklasifikasikan semua sampel dengan benar. Dalam konteks sistem deteksi dini, akurasi yang sempurna menunjukkan bahwa model dapat secara akurat mengidentifikasi siswa yang mengalami atau tidak mengalami kesulitan belajar tanpa kesalahan pada data yang diuji.
   * Dalam implementasi berbasis AI Computing Platform, akurasi tinggi sangat penting karena sistem ini berfungsi sebagai alat bantu bagi guru dan institusi pendidikan untuk menentukan langkah intervensi secara tepat waktu dan efektif.
2. **Precision, Recall, dan F1-Score untuk kelas '0' (tanpa kesulitan belajar): Semua bernilai 1.0**
   * **Precision (1.0):** Precision yang sempurna berarti setiap prediksi bahwa siswa tidak mengalami kesulitan belajar sepenuhnya benar. Dalam aplikasi deteksi dini, precision yang tinggi membantu meminimalkan prediksi yang salah, yaitu mencegah siswa yang membutuhkan bantuan dianggap tidak bermasalah.
   * **Recall (1.0):** Recall yang sempurna menunjukkan bahwa semua siswa yang sebenarnya tidak mengalami kesulitan belajar telah teridentifikasi dengan benar. Ini berarti sistem tidak melewatkan siswa yang mungkin tidak memerlukan intervensi khusus.
   * **F1-Score (1.0):** F1-Score adalah harmonisasi antara precision dan recall. Nilai F1-score yang sempurna memperlihatkan bahwa model mencapai keseimbangan optimal antara menghindari kesalahan negatif palsu (false negatives) dan positif palsu (false positives).
3. **Support (3)**
   * Support menunjukkan jumlah data dalam kelas yang diuji (dalam hal ini, terdapat 3 siswa dalam kelas tanpa kesulitan belajar). Sistem AI yang diuji ini mungkin masih memiliki jumlah data terbatas. Oleh karena itu, pengujian lebih lanjut dengan jumlah data yang lebih besar dapat lebih memvalidasi kinerja model.

**Relevansi dengan Pengembangan Perangkat Lunak untuk Sistem Deteksi Dini Berbasis AI**

Pengembangan perangkat lunak deteksi dini yang menggunakan Artificial Intelligence Computing Platform membutuhkan beberapa tahapan utama, yaitu:

1. **Pengumpulan dan Pemrosesan Data**: Sistem ini membutuhkan data yang beragam dan dalam jumlah besar untuk pelatihan agar mampu mengenali pola kesulitan belajar pada siswa. Data ini meliputi hasil tes, perilaku belajar, dan faktor eksternal lainnya.
2. **Pemilihan Algoritma AI**: Algoritma yang digunakan harus mampu mengidentifikasi pola dari data secara akurat. Jika algoritma menunjukkan hasil akurasi yang tinggi, seperti pada hasil Anda, ini bisa menjadi sinyal bahwa model sesuai dengan kebutuhan awal.
3. **Evaluasi dan Validasi Model**: Akurasi yang tinggi pada data pengujian menunjukkan model mungkin sangat cocok dengan data uji yang ada. Namun, dalam pengembangan perangkat lunak berbasis AI, evaluasi model pada data yang lebih luas dan variatif sangat penting agar tidak terjadi overfitting.
4. **Siklus Pengembangan Berkelanjutan**: Sistem deteksi dini ini perlu diperbarui secara rutin dengan data baru untuk mempertahankan akurasi dan reliabilitas dalam jangka panjang.
5. **Intervensi dan Rekomendasi**: Platform ini tidak hanya berhenti pada deteksi, tetapi juga berpotensi memberikan rekomendasi untuk guru, orang tua, atau siswa tentang langkah-langkah yang perlu diambil berdasarkan hasil deteksi.

Hasil akurasi sempurna ini bisa menunjukkan bahwa perangkat lunak deteksi dini Anda berhasil menjalankan fungsi dasarnya dengan baik, namun tetap perlu diuji lebih lanjut agar bisa diterapkan secara luas dan reliabel.

Pembahasan sesuai dengan GRAFIK:

Pembahasan sesuai dengan hasil Grafik Visualisasi Kesulitan Belajar: Kehadiran Vs Nilai Ujian :

Grafik visualisasi yang menunjukkan hubungan antara **Kehadiran** dan **Nilai Ujian** dalam konteks deteksi kesulitan belajar dapat memberikan wawasan yang berguna untuk memahami pola belajar siswa. Berikut adalah beberapa poin pembahasan yang relevan sesuai dengan hasil visualisasi ini:

1. **Korelasi antara Kehadiran dan Nilai Ujian**:
   * Jika grafik menunjukkan **hubungan positif** (misalnya, siswa dengan tingkat kehadiran lebih tinggi cenderung memiliki nilai ujian yang lebih baik), ini bisa menunjukkan bahwa kehadiran yang lebih sering berkaitan dengan pemahaman materi yang lebih baik.
   * Di sisi lain, jika tidak ada hubungan yang jelas antara kehadiran dan nilai ujian, mungkin perlu dieksplorasi faktor lain (seperti metode belajar, motivasi pribadi, atau dukungan keluarga) yang mempengaruhi nilai siswa.
2. **Identifikasi Siswa Berpotensi Mengalami Kesulitan Belajar**:
   * **Siswa dengan kehadiran rendah dan nilai rendah** mungkin menjadi kelompok utama yang perlu diintervensi, karena kombinasi ini bisa mengindikasikan adanya kendala belajar yang serius, seperti kesulitan memahami materi, masalah kedisiplinan, atau motivasi.
   * Siswa dengan **kehadiran tinggi namun nilai ujian rendah** mungkin menunjukkan bahwa meskipun mereka hadir di kelas, mereka masih memerlukan bantuan tambahan (misalnya, bimbingan lebih lanjut, sumber daya belajar tambahan, atau metode belajar yang lebih interaktif).
3. **Cluster atau Kelompok Siswa Berdasarkan Pola**:
   * Visualisasi ini juga bisa digunakan untuk mengelompokkan siswa ke dalam beberapa kategori berdasarkan pola yang muncul, seperti:
     + **Siswa berprestasi**: Tingkat kehadiran tinggi dan nilai ujian tinggi.
     + **Siswa dengan kehadiran rendah**: Kehadiran rendah dan nilai ujian bervariasi.
     + **Siswa yang butuh bantuan tambahan**: Kehadiran tinggi tetapi nilai rendah.
   * Dengan mengidentifikasi pola ini, sistem bisa memberikan rekomendasi personalisasi intervensi yang spesifik untuk masing-masing kelompok.
4. **Evaluasi Efektivitas Intervensi yang Ada**:
   * Grafik ini dapat digunakan sebagai dasar untuk mengevaluasi efektivitas program intervensi yang sudah diterapkan. Jika ada peningkatan kehadiran yang tidak diiringi peningkatan nilai ujian, ini menunjukkan bahwa intervensi masih perlu disesuaikan atau ditingkatkan.
5. **Rekomendasi untuk Guru dan Sekolah**:
   * Visualisasi ini juga bisa membantu guru dalam membuat rencana pengajaran yang lebih sesuai. Misalnya, mereka bisa memberi perhatian lebih kepada siswa dengan kehadiran rendah atau mencari cara untuk meningkatkan pemahaman materi bagi siswa yang sering hadir namun nilai ujian mereka tidak memuaskan.
   * Sekolah juga bisa merancang strategi motivasi untuk meningkatkan kehadiran siswa, misalnya melalui program penghargaan atau dukungan lebih intensif bagi siswa yang mengalami kesulitan.
6. **Penggunaan dalam Sistem Deteksi Dini Berbasis AI**:
   * Dengan adanya data visualisasi seperti ini, algoritma AI dalam sistem dapat dilatih untuk mendeteksi pola-pola yang spesifik, sehingga secara otomatis dapat memberikan peringatan atau rekomendasi ketika mendeteksi siswa yang menunjukkan kombinasi nilai dan kehadiran yang dianggap rentan.
   * Data ini akan memperkuat sistem deteksi dini dan memungkinkan personalisasi intervensi berdasarkan kebutuhan siswa yang terpantau oleh sistem.

Grafik ini pada dasarnya memberikan pandangan yang lebih jelas terhadap hubungan aspek kehadiran dan nilai ujian, dan membantu mengidentifikasi potensi kesulitan belajar. Dengan demikian, guru dan sistem berbasis AI dapat bersama-sama memberikan solusi dan intervensi yang tepat untuk membantu meningkatkan proses belajar siswa.

Kesimpulan:

Kesimpulan yang terbaik dari hasil visualisasi hubungan antara **Kehadiran** dan **Nilai Ujian** dalam konteks sistem deteksi dini kesulitan belajar adalah sebagai berikut:

1. **Hubungan Kehadiran dan Nilai Ujian**:
   * Data menunjukkan bahwa kehadiran siswa berkaitan erat dengan nilai ujian mereka, di mana siswa yang hadir lebih sering cenderung memiliki nilai yang lebih tinggi. Ini mengindikasikan bahwa kehadiran berperan penting dalam pemahaman materi pelajaran dan pencapaian akademik siswa.
   * Di sisi lain, siswa dengan kehadiran rendah dan nilai ujian rendah menandakan adanya kemungkinan kesulitan belajar atau faktor eksternal lain yang mempengaruhi kehadiran dan kinerja akademik mereka.
2. **Identifikasi Kelompok Siswa Rentan**:
   * Visualisasi membantu mengidentifikasi kelompok siswa yang berpotensi mengalami kesulitan belajar, terutama mereka yang memiliki tingkat kehadiran tinggi tetapi nilai ujian rendah atau mereka yang jarang hadir dan menunjukkan nilai rendah. Kelompok-kelompok ini memerlukan perhatian dan intervensi khusus untuk membantu mereka meningkatkan pemahaman dan motivasi belajar.
3. **Rekomendasi untuk Intervensi Personal**:
   * Berdasarkan hasil ini, sekolah dapat mengembangkan strategi intervensi yang lebih personal dan tepat sasaran, seperti program bimbingan tambahan atau dukungan motivasi. Intervensi ini bisa dirancang sesuai dengan kebutuhan spesifik masing-masing kelompok siswa, baik yang membutuhkan dukungan akademik maupun motivasional.
4. **Peningkatan Efektivitas Sistem Deteksi Dini Berbasis AI**:
   * Dengan memanfaatkan visualisasi ini, sistem deteksi dini berbasis AI dapat meningkatkan akurasi dalam mengidentifikasi siswa yang membutuhkan bantuan. Data yang terstruktur ini memungkinkan sistem untuk mengenali pola-pola tertentu dan memberikan rekomendasi otomatis yang dapat membantu guru dalam mengelola intervensi belajar.

**Kesimpulan Utama**

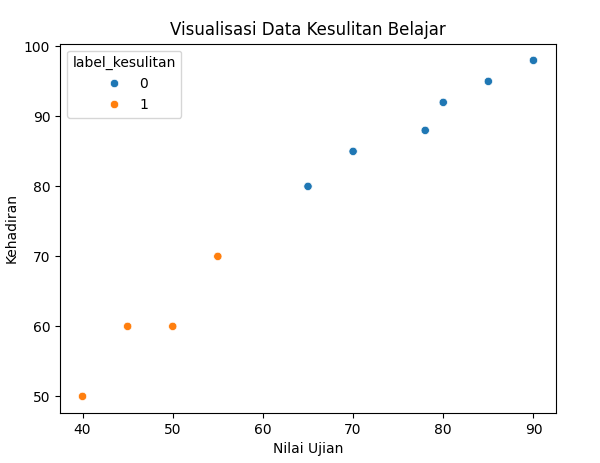
Secara keseluruhan, hasil ini menekankan pentingnya kehadiran dalam pencapaian akademik siswa dan menunjukkan bahwa visualisasi hubungan antara kehadiran dan nilai ujian dapat membantu mengidentifikasi siswa yang rentan mengalami kesulitan belajar. Sistem deteksi dini yang memanfaatkan data ini dapat mendukung intervensi yang lebih efektif dan meningkatkan peluang keberhasilan akademik siswa.

Saran saran:

Berikut adalah beberapa saran yang dapat dipertimbangkan untuk pengembangan sistem deteksi dini kesulitan belajar berbasis kehadiran dan nilai ujian ke depannya:

1. **Perluasan Data untuk Memperkuat Sistem Deteksi Dini**:
   * **Tambahkan Variabel Lain**: Selain kehadiran dan nilai ujian, pertimbangkan menambah variabel lain seperti catatan tugas harian, partisipasi kelas, serta hasil kuis atau ulangan. Dengan data yang lebih lengkap, sistem dapat memberikan deteksi dini yang lebih akurat.
   * **Penggunaan Data Longitudinal**: Data dari semester atau tahun sebelumnya dapat digunakan untuk melihat pola performa siswa dari waktu ke waktu, sehingga dapat memprediksi dan mengidentifikasi potensi kesulitan belajar sejak dini.
2. **Pengembangan Model Prediksi Berbasis AI yang Lebih Canggih**:
   * **Algoritma Pembelajaran Mesin**: Coba kembangkan algoritma yang lebih kompleks, seperti **Random Forest**, **SVM**, atau **Neural Networks**. Algoritma ini dapat menangkap pola-pola kompleks dari data yang lebih variatif dan memberikan hasil prediksi yang lebih baik.
   * **Penerapan Teknik Klasifikasi Multi-Kelas**: Sebagai contoh, sistem dapat mengklasifikasikan siswa tidak hanya dalam dua kelompok (rentan atau tidak rentan), tetapi juga dalam tingkat risiko (rendah, sedang, atau tinggi) untuk memfasilitasi intervensi yang lebih tepat sasaran.
3. **Personalisasi Intervensi untuk Meningkatkan Efektivitas**:
   * **Rekomendasi Berdasarkan Profil Siswa**: Sistem bisa memberikan saran yang disesuaikan berdasarkan profil kesulitan belajar siswa. Misalnya, siswa yang sering hadir tetapi nilai rendah dapat diarahkan ke kelas tambahan, sementara siswa dengan kehadiran rendah dan nilai rendah dapat diarahkan ke konseling atau program motivasi.
   * **Pemantauan Berkelanjutan**: Intervensi yang dilakukan sebaiknya terus dipantau, dan efektivitasnya dievaluasi secara berkala. Sistem dapat memperbarui rekomendasi berdasarkan respons siswa terhadap intervensi sebelumnya.
4. **Pengembangan Fitur Notifikasi dan Dashboard**:
   * **Pemberitahuan Otomatis**: Integrasikan fitur notifikasi otomatis kepada guru atau orang tua ketika sistem mendeteksi siswa dalam kategori berisiko, sehingga mereka dapat segera mengambil tindakan.
   * **Dashboard untuk Pemantauan Visual**: Pengembangan dashboard yang menampilkan visualisasi perkembangan siswa secara real-time dapat membantu guru dan pihak sekolah memantau kemajuan siswa dan melakukan tindakan segera jika diperlukan.
5. **Melibatkan Guru dan Orang Tua dalam Proses Intervensi**:
   * **Kolaborasi dengan Guru**: Saran dari sistem sebaiknya disesuaikan dengan masukan dari guru karena mereka memiliki pemahaman langsung tentang kondisi dan perilaku siswa di kelas. Hal ini juga membantu menciptakan intervensi yang lebih efektif dan personal.
   * **Feedback dari Orang Tua**: Libatkan orang tua dalam proses pemantauan perkembangan siswa, misalnya dengan memberikan akses terbatas pada dashboard sistem untuk memantau kehadiran dan nilai anak mereka serta intervensi yang sedang berlangsung.
6. **Evaluasi dan Peningkatan Sistem secara Berkala**:
   * **Evaluasi Kinerja Sistem**: Secara berkala, lakukan evaluasi untuk memastikan bahwa prediksi dan rekomendasi yang diberikan sistem tetap akurat dan relevan.
   * **Pengembangan dengan Metode Iteratif**: Terus tingkatkan sistem berdasarkan umpan balik dari guru, orang tua, dan siswa. Sistem ini perlu berkembang seiring dengan perubahan kebutuhan dan karakteristik siswa serta penemuan metode yang lebih efektif.

Dengan saran-saran ini, diharapkan sistem deteksi dini akan lebih akurat dan responsif terhadap kebutuhan siswa, serta mampu memberikan manfaat yang lebih besar dalam membantu siswa mengatasi kesulitan belajar.



Garfik Visualisasi Data Kesulitan Belajar Nilai Ujian Versus Kehadiran

Akurasi: 1.0

precision recall f1-score support

0 1.00 1.00 1.00 3

accuracy 1.00 3

macro avg 1.00 1.00 1.00 3

weighted avg 1.00 1.00 1.00 3

**Sistem Deteksi Dini Kesulitan Belajar Siswa Berbasis Artificial Intelligence Computing Platform** menggunakan **algoritma Decision Tree (Pohon Keputusan)** untuk mengklasifikasikan data menjadi dua kelas, yaitu label\_kesulitan (0 atau 1). Algoritma ini membangun model prediktif dengan mempartisi dataset berdasarkan nilai fitur yang memaksimalkan pemisahan (mengurangi ketidakpastian). Berikut adalah penjelasan dan rumus matematisnya.

**Algoritma Decision Tree**

**Decision Tree** membangun struktur seperti pohon untuk membuat keputusan berdasarkan atribut fitur. Dalam program ini:

1. **Input Features (Fitur Masukan):**
   * nilai\_ujian
   * kehadiran
   * waktu\_belajar
2. **Output (Target):**
   * label\_kesulitan:
     + 0 (tidak sulit belajar)
     + 1 (sulit belajar)
3. **Proses Utama:**
   * Pemisahan dataset (splitting) dilakukan pada setiap node dengan memilih atribut yang memberikan **informasi terbaik** untuk memisahkan kelas target.

**Rumus Matematis dalam Decision Tree**

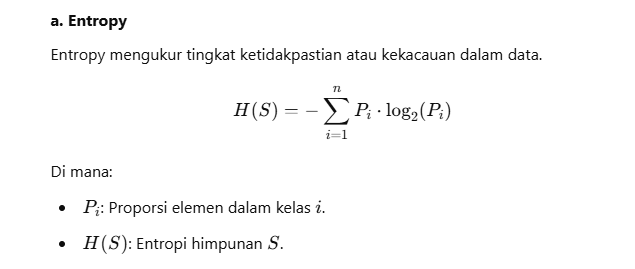
**1. Pemilihan Atribut dengan Kriteria (Split Criterion)**

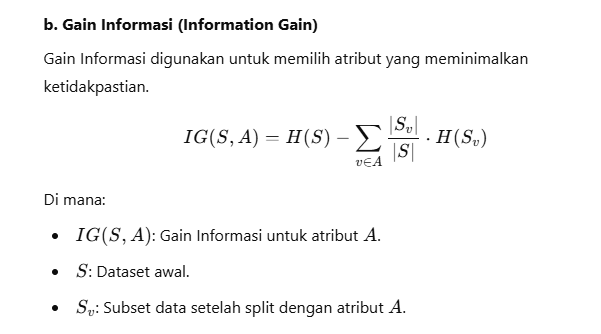
Atribut terbaik dipilih berdasarkan kriteria seperti:

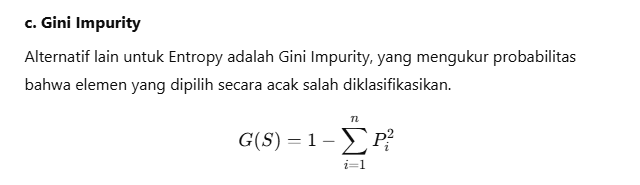
* **Entropy dan Gain Informasi (Information Gain)**.
* **Gini Impurity**.

**a. Entropy**

Entropy mengukur tingkat ketidakpastian atau kekacauan dalam data.

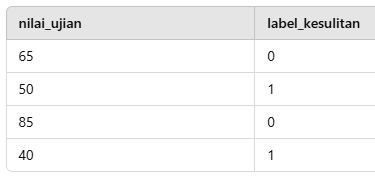


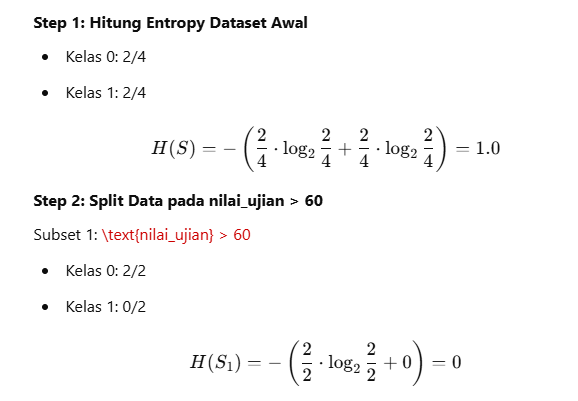


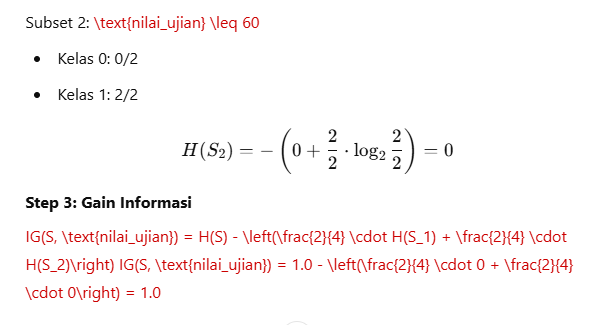


**Contoh Perhitungan Sederhana:**

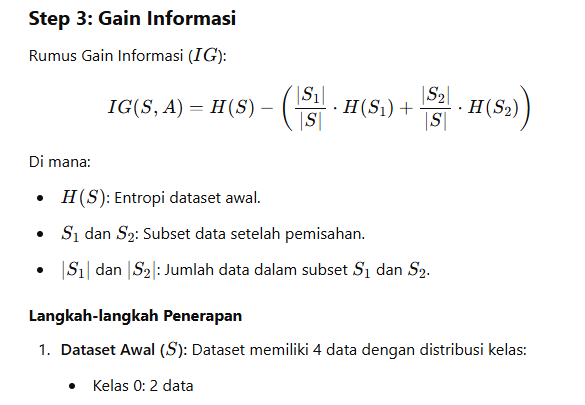
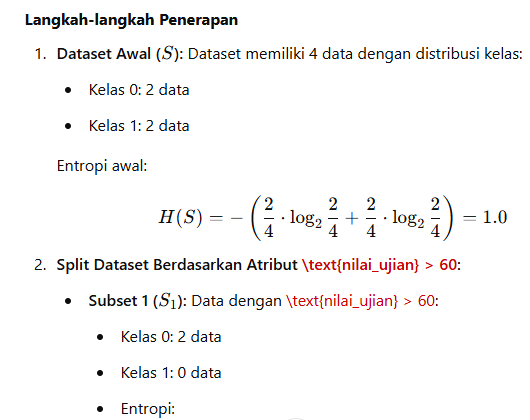
Misalkan dataset kecil seperti ini:

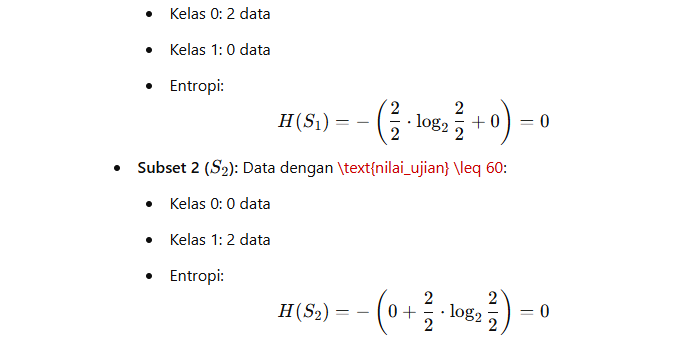


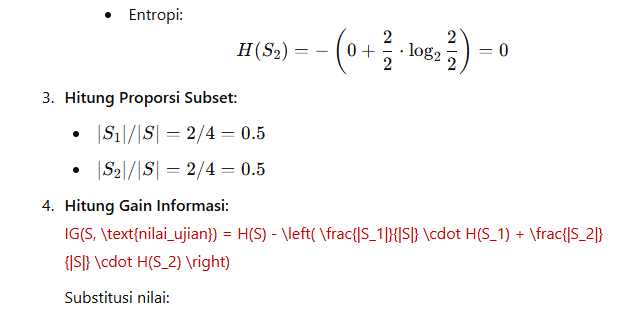


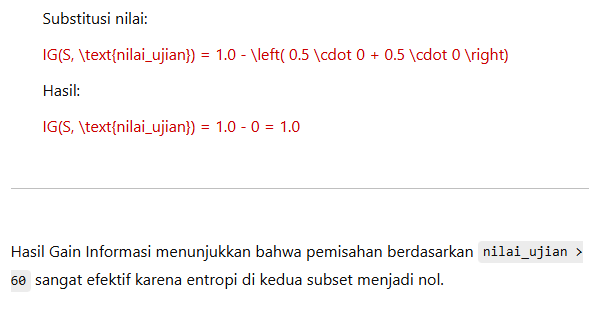


Step 3 diperbaiki Menjadi:









**Output Akhir**

* Algoritma memilih split terbaik berdasarkan nilai tertinggi dari Gain Informasi atau Gini Impurity.
* Model terlatih memberikan akurasi 100% karena dataset kecil dan tidak ada noise.