

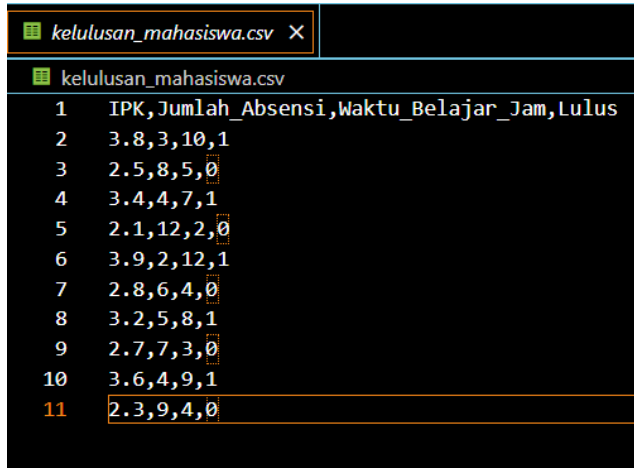
LAPORAN MACHINE LEARNING

PERTEMUAN KE 4

Nama : Edo Aditya Saputra

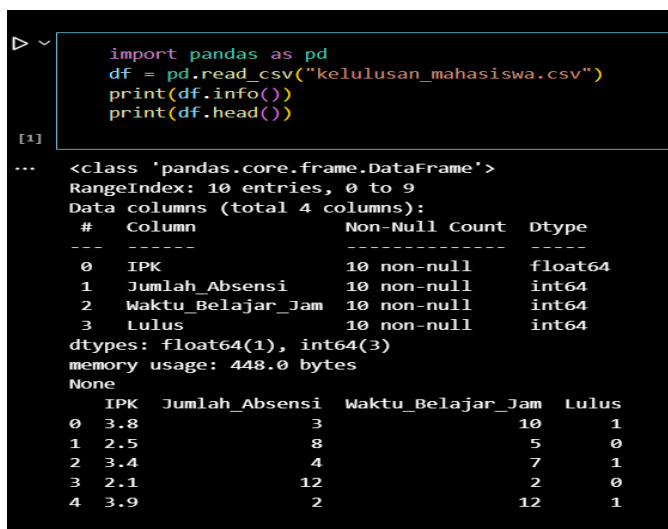
Kelas : 05TPLE017

1.



	IPK	Jumlah_Absensi	Waktu_Belajar_Jam	Lulus
1	IPK	Jumlah_Absensi	Waktu_Belajar_Jam	Lulus
2	3.8	3	10	1
3	2.5	8	5	0
4	3.4	4	7	1
5	2.1	12	2	0
6	3.9	2	12	1
7	2.8	6	4	0
8	3.2	5	8	1
9	2.7	7	3	0
10	3.6	4	9	1
11	2.3	9	4	0

Pada tahap awal, berhasil dibuat dataset berformat CSV menggunakan VSCode. Dataset ini berisi informasi mahasiswa, meliputi kolom IPK, Jumlah Absensi, Waktu Belajar per Hari, dan status kelulusan. Seluruh data diketik dengan rapi, menggunakan koma sebagai pemisah, dan disimpan dengan nama kelulusan_mahasiswa.csv. Pembuatan dataset ini merupakan langkah penting karena menjadi dasar bagi proses analisis dan pemodelan selanjutnya.



```
import pandas as pd
df = pd.read_csv("kelulusan_mahasiswa.csv")
print(df.info())
print(df.head())
```

[1]

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10 entries, 0 to 9
Data columns (total 4 columns):
 #   Column              Non-Null Count  Dtype  
---  --
 0   IPK                  10 non-null    float64
 1   Jumlah_Absensi       10 non-null    int64  
 2   Waktu_Belajar_Jam    10 non-null    int64  
 3   Lulus                10 non-null    int64  
dtypes: float64(1), int64(3)
memory usage: 448.0 bytes
None
```

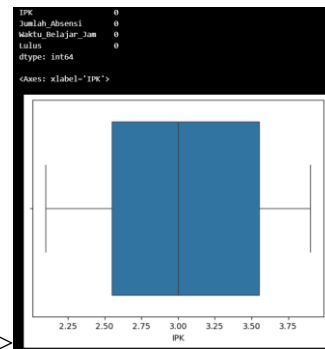
	IPK	Jumlah_Absensi	Waktu_Belajar_Jam	Lulus
0	3.8	3	10	1
1	2.5	8	5	0
2	3.4	4	7	1
3	2.1	12	2	0
4	3.9	2	12	1

2.

Setelah dataset berhasil dibuat, langkah berikutnya adalah melakukan collection atau pengambilan data menggunakan Python dan library Pandas. File kelulusan_mahasiswa.csv dibuka dengan Pandas, dan informasi dasar dataset ditampilkan menggunakan perintah `df.info()` serta lima baris pertama dengan `df.head()`. Dari hasil ini, dapat dipastikan bahwa semua kolom terbaca dengan benar, tipe data sesuai, dan tidak ada kesalahan penulisan. Langkah ini penting untuk memastikan bahwa data siap untuk tahap pembersihan dan analisis lebih lanjut.

```
print(df.isnull().sum())
df = df.drop_duplicates()

import seaborn as sns
sns.boxplot(x=df['IPK'])
```



3. hasil ->

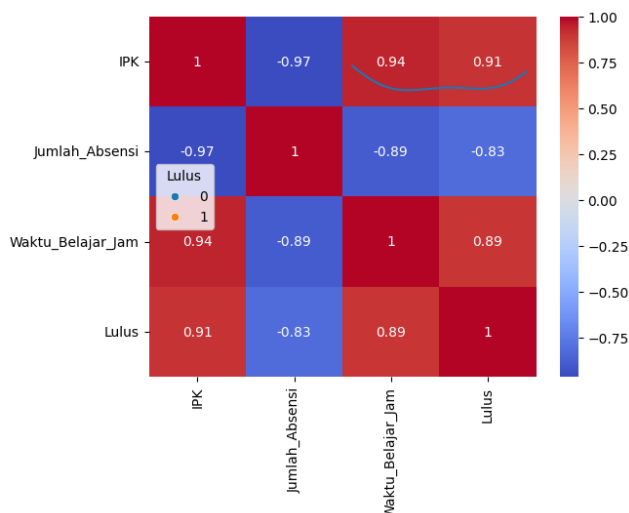
Tahap cleaning dilakukan untuk memastikan dataset bebas dari kesalahan dan siap dianalisis. Pertama, memeriksa apakah ada nilai yang hilang menggunakan `df.isnull().sum()`, dan hasilnya menunjukkan semua data lengkap tanpa missing value. Selanjutnya, data duplikat dihapus dengan `df.drop_duplicates()` agar tidak mengganggu analisis. Outlier pada kolom IPK diidentifikasi menggunakan boxplot, meskipun pada dataset yang sangat kecil ini outlier tidak signifikan dan langkah ini menjadi kurang relevan. Dengan tahap cleaning ini, dataset menjadi lebih rapi dan konsisten, namun perlu dicatat bahwa beberapa prosedur seperti deteksi outlier atau pengisian missing value lebih optimal jika dataset lebih besar.

```
print(df.describe())
sns.histplot(df['IPK'], bins=10, kde=True)
sns.scatterplot(x='IPK', y='Waktu_Belajar_Jam', data=df, hue='Lulus')
sns.heatmap(df.corr(), annot=True, cmap="coolwarm")
```

	IPK	Jumlah_Absensi	Waktu_Belajar_Jam	Lulus
count	10.000000	10.000000	10.000000	10.000000
mean	3.030000	6.000000	6.400000	0.500000
std	0.639531	3.055050	3.306559	0.527046
min	2.100000	2.000000	2.000000	0.000000
25%	2.550000	4.000000	4.000000	0.000000
50%	3.000000	5.500000	6.000000	0.500000
75%	3.550000	7.750000	8.750000	1.000000
max	3.900000	12.000000	12.000000	1.000000

<Axes: >

4.



Setelah dataset dibersihkan, tahap selanjutnya adalah Exploratory Data Analysis (EDA) untuk memahami karakteristik data. Statistik deskriptif dihitung menggunakan `df.describe()`, sehingga terlihat nilai rata-rata, minimum, maksimum, dan standar deviasi dari setiap kolom numerik. Distribusi IPK divisualisasikan dengan histogram, sedangkan hubungan antara IPK dan Waktu Belajar diperiksa menggunakan scatterplot yang diberi warna berdasarkan status kelulusan mahasiswa. Selain itu, heatmap korelasi dibuat untuk melihat hubungan antar fitur. Dari EDA ini, dapat diamati pola umum, seperti mahasiswa dengan IPK tinggi cenderung belajar lebih lama, serta hubungan fitur yang potensial untuk model prediksi kelulusan. Namun, karena dataset sangat kecil, beberapa visualisasi seperti histogram dan heatmap menjadi kurang informatif dan harus diinterpretasikan dengan hati-hati.

	processed_kelulusan.csv
1	IPK,Jumlah_Absensi,Waktu_Belajar_Jam,Lulus,Rasio_Absensi,IPK_x_Study
2	3.8,3,10,1,0.21428571428571427,38.0
3	2.5,8,5,0,0.5714285714285714,12.5
4	3.4,4,7,1,0.2857142857142857,23.8
5	2.1,12,2,0,0.8571428571428571,4.2
6	3.9,2,12,1,0.14285714285714285,46.8
7	2.8,6,4,0,0.42857142857142855,11.2
8	3.2,5,8,1,0.35714285714285715,25.6
9	2.7,7,3,0,0.5,8.100000000000001
10	3.6,4,9,1,0.2857142857142857,32.4
11	2.3,9,4,0,0.6428571428571429,9.2
12	

5.

Pada tahap Feature Engineering, beberapa fitur baru dibuat untuk meningkatkan kemampuan analisis dan prediksi kelulusan mahasiswa. Fitur pertama adalah Rasio_Absensi, yang menghitung proporsi jumlah absensi terhadap total pertemuan, sehingga mempermudah perbandingan antar mahasiswa. Fitur kedua adalah IPK_x_Study, hasil perkalian antara IPK dan Waktu Belajar, yang bertujuan menangkap interaksi antara prestasi akademik dan intensitas belajar. Dataset dengan fitur baru kemudian disimpan ke file processed_kelulusan.csv untuk tahap berikutnya.

```

from sklearn.model_selection import train_test_split

x = df.drop('Lulus', axis=1)
y = df['Lulus']

x_train, x_temp, y_train, y_temp = train_test_split(
    x, y, test_size=0.3, stratify=y, random_state=42
)

x_val, x_test, y_val, y_test = train_test_split(
    x_temp, y_temp, test_size=0.5, random_state=42
)

print(x_train.shape, x_val.shape, x_test.shape)

```

7] ✓ 2.7s

.. (7, 5) (1, 5) (2, 5)

6.

Langkah terakhir dalam persiapan data adalah membagi dataset menjadi subset untuk training, validation, dan testing. Dataset dibagi dengan proporsi 70% untuk training, 15% untuk validation, dan 15% untuk testing, menggunakan metode stratified split agar proporsi kelas kelulusan tetap terjaga di setiap subset. Variabel input (fitur) dan target (Lulus) dipisahkan sebelum proses split. Masalah utama muncul karena dataset sangat kecil (10 baris) dan stratified split mencoba mempertahankan proporsi kelas di subset validation dan testing. Dengan jumlah baris per kelas yang sangat sedikit, stratify menyebabkan subset kecil mungkin tidak memiliki representasi yang cukup untuk salah satu kelas, atau bahkan menimbulkan error.

Salah satu solusi sederhana adalah **menghilangkan parameter stratify=y_temp** saat membagi dataset. Dengan cara ini, pembagian data tetap acak, dan walaupun proporsi kelas tidak dijaga persis sama, proses split tetap berjalan tanpa error. Pendekatan ini lebih cocok untuk dataset kecil karena tidak menuntut distribusi kelas yang presisi di subset yang hanya berisi 1–2 baris. Namun, perlu dicatat bahwa metode ini bisa membuat distribusi kelas di validation atau test sedikit berbeda dari dataset asli, sehingga hasil evaluasi model harus diinterpretasikan dengan hati-hati.