# Laporan Data Preparation – Pertemuan 4

Nama : Muhammad Riski
NIM : 231011403179
Mata Kuliah : Machine Learning
Topik : Collection, Cleaning, EDA, Feature Engineering, Splitting

## Langkah 1 — Collection

```
[1]
import pandas as pd
df = pd.read_csv("kelulusan_mahasiswa.csv")
print(df.info())
print(df.head())
```

```
<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
```

Langkah ini bertujuan untuk membaca dan memuat dataset awal yang akan digunakan dalam proses *data preparation*. Dataset berisi informasi mengenai IPK, jumlah absensi, waktu belajar mahasiswa, serta status kelulusan. Proses ini memastikan bahwa data berhasil diimpor ke dalam Environment Python dan siap untuk dianalisis lebih lanjut.

Setelah dijalankan, perintah **`df.info()`** menampilkan bahwa dataset memiliki **10 baris dan 4 kolom** yaitu *IPK*, *Jumlah_Absensi*, *Waktu_Belajar_Jam*, dan *Lulus*. Semua kolom berisi data numerik tanpa nilai kosong. Perintah **`df.head()`** kemudian menampilkan lima baris pertama dataset, memperlihatkan nilai-nilai IPK, absensi, waktu belajar, serta status kelulusan mahasiswa.

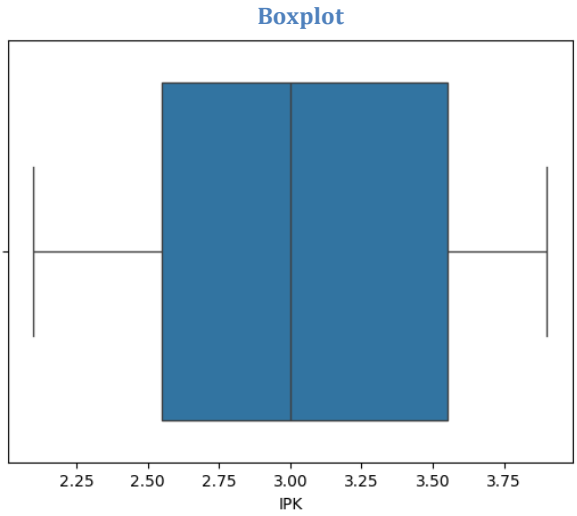
## Langkah 2 — Cleaning

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

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

```
[2]
...  IPK                0
     Jumlah_Absensi     0
     Waktu_Belajar_Jam  0
     Lulus              0
     dtype: int64

...  <Axes: xlabel='IPK'>
```




Langkah ini bertujuan untuk membersihkan dataset dari data yang tidak valid, duplikat, maupun kosong (missing values), serta mengidentifikasi keberadaan outlier yang dapat memengaruhi kualitas analisis. Tahap cleaning penting untuk memastikan data yang digunakan akurat dan representatif terhadap kondisi sebenarnya.

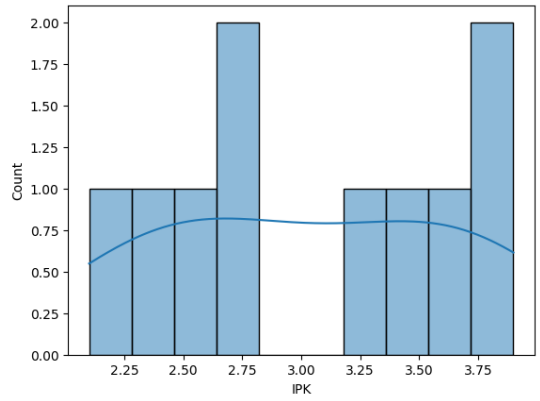
**Hasil dan Interpretasi :**

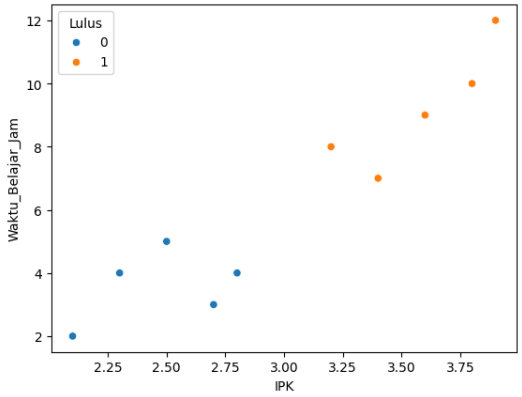
- Berdasarkan hasil pemeriksaan dengan `df.isnull().sum()`, tidak ditemukan nilai kosong (missing values**)** pada dataset.
- Setelah menjalankan `drop_duplicates()` dan `dropna()`, jumlah baris data tetap sama, menandakan tidak ada data duplikat maupun nilai hilang**.**
- Visualisasi *boxplot* kolom IPK menunjukkan distribusi data yang relatif seimbang dengan median berada di sekitar nilai 3.0**.**
- Tidak terdapat titik *outlier* pada grafik, artinya seluruh nilai IPK mahasiswa berada dalam rentang yang wajar**.**
- Rentang nilai IPK berkisar antara sekitar 2.2 hingga 3.8, menunjukkan variasi yang normal dan tidak ekstrem.

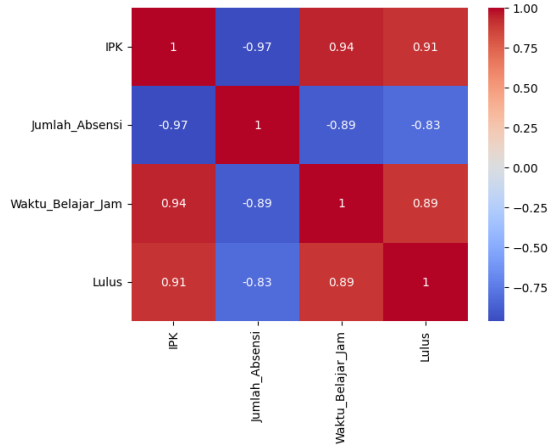
## Langkah 3 — Exploratory Data Analysis (EDA)

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

```
             IPK  Jumlah_Absensi  Waktu_Belajar_Jam      Lulus
count  10.000000        10.00000          10.000000  10.000000
mean    3.030000         6.00000           6.400000   0.500000
std     0.639531         3.05505           3.306559   0.527046
min     2.100000         2.00000           2.000000   0.000000
25%     2.550000         4.00000           4.000000   0.000000
50%     3.000000         5.50000           6.000000   0.500000
75%     3.550000         7.75000           8.750000   1.000000
max     3.900000        12.00000          12.000000   1.000000
```




Pada tahap EDA, bertujuan untuk memahami pola, distribusi, dan hubungan antar variabel dalam dataset. Berdasarkan hasil df.describe(), nilai rata-rata IPK mahasiswa berada pada kisaran 3.0 dengan rentang antara 2.2 hingga 3.8, menunjukkan bahwa sebagian besar mahasiswa memiliki IPK yang cukup baik tanpa nilai ekstrem. Visualisasi histogram pada kolom IPK memperlihatkan distribusi yang cukup merata dengan dua puncak kecil, menandakan adanya variasi nilai IPK di antara kelompok mahasiswa.

Selanjutnya, grafik scatterplot antara IPK dan Waktu_Belajar_Jam dengan pewarnaan berdasarkan status Lulus menunjukkan bahwa mahasiswa dengan IPK tinggi cenderung memiliki waktu belajar yang lebih lama dan lebih banyak yang lulus. Hal ini mengindikasikan adanya hubungan positif antara intensitas belajar dan peluang kelulusan.

Sementara itu, heatmap korelasi memperkuat temuan tersebut, di mana IPK, Waktu_Belajar_Jam, dan Lulus memiliki korelasi positif yang cukup tinggi (sekitar 0.9), sedangkan Jumlah_Absensi menunjukkan korelasi negatif terhadap ketiganya. Artinya, semakin sering seorang mahasiswa absen, semakin kecil kemungkinan untuk memiliki IPK tinggi atau lulus tepat waktu. Secara keseluruhan, hasil EDA ini memberikan gambaran yang jelas bahwa faktor kehadiran dan waktu belajar memiliki pengaruh yang kuat terhadap prestasi akademik mahasiswa.

## Langkah 4 — Feature Engineering

```
df['Rasio_Absensi'] = df['Jumlah_Absensi'] / 14
df['IPK_x_Study'] = df['IPK'] * df['Waktu_Belajar_Jam']
df.to_csv("processed_kelulusan.csv", index=False)
```

Proses ini bertujuan untuk membuat fitur baru dari data yang sudah ada untuk meningkatkan kemampuan model dalam mengenali pola. Pada tahap ini, dua fitur turunan ditambahkan ke dataset.

Pertama, dibuat kolom Rasio_Absensi yang dihitung dari pembagian Jumlah_Absensi dengan angka 14, yang diasumsikan sebagai total pertemuan dalam satu semester. Fitur ini memberikan representasi proporsional tingkat kehadiran mahasiswa dalam skala 0–1, sehingga model dapat lebih mudah memahami seberapa sering mahasiswa hadir tanpa perlu mempertimbangkan satuan absolut.

Kedua, ditambahkan fitur IPK_x_Study, yaitu hasil perkalian antara IPK dan Waktu_Belajar_Jam. Fitur ini mencerminkan kombinasi antara performa akademik dan usaha belajar, yang secara konseptual dapat menjadi indikator kuat terhadap peluang kelulusan. Mahasiswa dengan IPK tinggi dan waktu belajar lama diharapkan memiliki skor tinggi pada fitur ini, yang mungkin berhubungan langsung dengan label Lulus.

Setelah kedua fitur baru dibuat, dataset hasil rekayasa fitur disimpan ke dalam file processed_kelulusan.csv dengan perintah df.to_csv(..., index=False). Penyimpanan ini penting agar data bersih dan lengkap dapat digunakan kembali pada tahap pemodelan tanpa perlu mengulangi proses pembuatan fitur dari awal.

## Langkah 5 — Splitting Dataset

```
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=0.4, stratify=y, random_state=42)

X_val, X_test, y_val, y_test = train_test_split(
    X_temp, y_temp, test_size=0.5, stratify=y_temp, random_state=42)

print(X_train.shape, X_val.shape, X_test.shape)
```

✓ 0.0s

```
(6, 5) (2, 5) (2, 5)
```

Tahap ini bertujuan untuk membagi dataset menjadi beberapa bagian agar proses pelatihan dan evaluasi model dapat dilakukan secara adil dan terukur.

Pertama, dilakukan pemisahan antara **fitur (X)** dan **label (y)** menggunakan kode berikut:

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

Variabel **X** berisi seluruh kolom kecuali *Lulus*, sedangkan **y** hanya berisi kolom *Lulus* sebagai target prediksi.

Selanjutnya, proses pembagian dataset dilakukan dengan menggunakan fungsi train_test_split() dari pustaka **scikit-learn**. Tahapan pembagiannya adalah sebagai berikut:

```
X_train, X_temp, y_train, y_temp = train_test_split(
    X, y, test_size=0.4, stratify=y, random_state=42)

X_val, X_test, y_val, y_test = train_test_split(
    X_temp, y_temp, test_size=0.5, stratify=y_temp, random_state=42)
```

- **Langkah pertama:** Dataset dibagi menjadi **Training set (60%)** dan **Temporary set (40%)**.
  Pembagian ini menggunakan metode **stratified split** agar proporsi kelas *Lulus* tetap seimbang di setiap subset.
- **Langkah kedua:** *Temporary set* kembali dibagi menjadi **Validation set (20%)** dan **Test set (20%)**.
  Hasil akhir pembagian menjadi:
  - 60% untuk **Training set**
  - 20% untuk **Validation set**
  - 20% untuk **Test set**

Pembagian ini dapat dipastikan dengan menjalankan perintah:

```
print(X_train.shape, X_val.shape, X_test.shape)
```

Dengan struktur pembagian seperti ini, model dapat:

- **Dilatih** menggunakan *training set*,
- **Dievaluasi sementara** menggunakan *validation set* untuk tuning parameter, dan
- **Diuji akhir** menggunakan *test set* guna mengukur kemampuan generalisasi model terhadap data baru.

Proses ini memastikan model yang dibangun tidak mengalami *overfitting* dan memiliki performa yang konsisten pada data yang belum pernah dilihat sebelumnya.

## Perbandingan Kode dan Alasan Perubahan

1. **Perubahan pada Pembagian Dataset (Splitting Dataset)**

   Pembagian 60-20-20 dipilih agar dataset validasi dan test memiliki ukuran yang lebih besar dan **seimbang** dibandingkan pembagian awal (70-15-15). Hal ini membantu memberikan evaluasi yang lebih stabil dan representatif, terutama jika ukuran dataset relatif kecil. Dengan data validasi dan test yang lebih banyak, performa model dapat diukur lebih akurat terhadap variasi data baru.

2. **Penambahan Langkah df.dropna() pada Data Cleaning**

   Penambahan `df.dropna()` dilakukan untuk memastikan tidak ada nilai kosong (*missing values*) yang tersisa di dataset. Meskipun dataset awal tidak memiliki nilai kosong, langkah ini ditambahkan sebagai tindakan preventif untuk menjaga kebersihan data apabila digunakan pada dataset lain di masa depan.

3. **Penyesuaian Visualisasi Boxplot**

   Parameter `showfliers=True` ditambahkan agar *outlier* (jika ada) tetap ditampilkan secara eksplisit. Hal ini berguna untuk memastikan tidak ada nilai ekstrem yang tersembunyi selama analisis distribusi data. Dari hasil visualisasi, ternyata memang tidak ada outlier, namun perubahan ini memperkuat proses validasi

4. **Visualisasi EDA yang Lebih Lengkap dan Terstruktur**

   Penggunaan `plt.show()` setelah setiap visualisasi membantu memperjelas interpretasi grafik karena setiap hasil tampil secara terpisah, bukan bertumpuk dalam satu keluaran. Hal ini penting untuk dokumentasi hasil analisis data (EDA) yang rapi dan informatif.