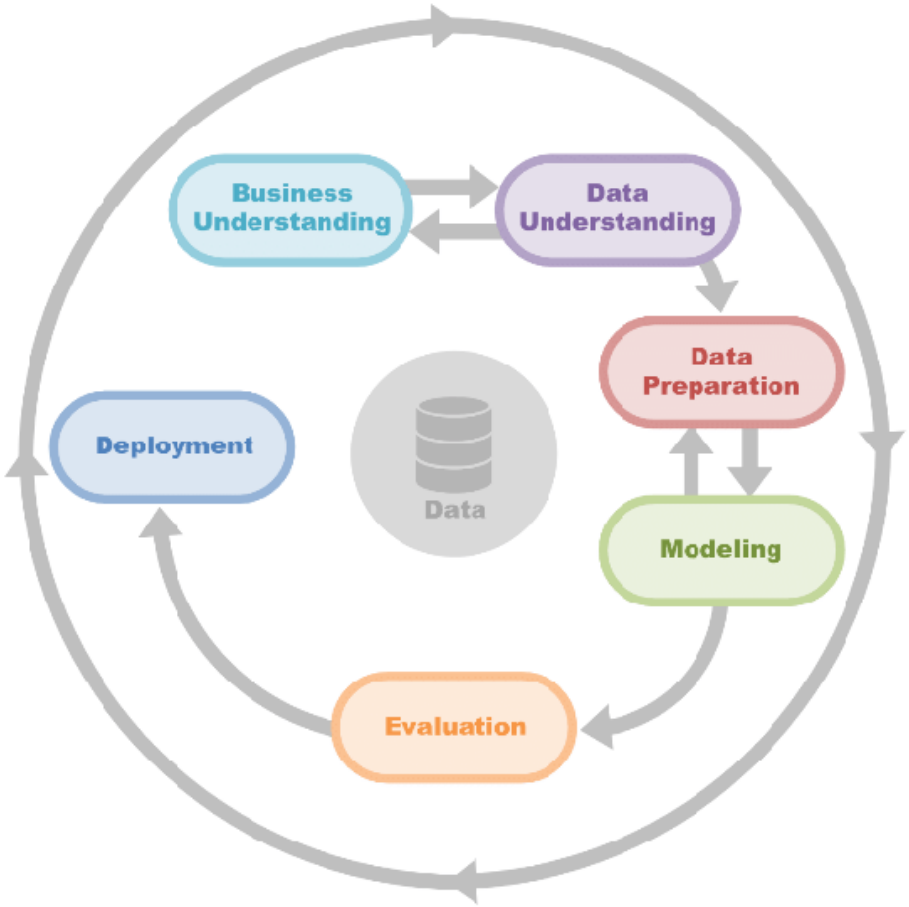
Modul 1
Praktikum Data Mining

PROGRAM STUDI SAINS DATA
FAKULTAS SAINS
INSTITUT TEKNOLOGI SUMATERA
2025

Pengantar Data Mining

Data mining adalah proses menemukan pola, tren, dan informasi berguna dari kumpulan data besar dengan menggunakan teknik statistik, algoritma, dan metode analisis. Tujuannya untuk mengidentifikasi informasi yang mungkin tidak terlihat secara langsung atau yang tersembunyi dalam data tersebut. Proses ini sering digunakan untuk membantu membuat keputusan bisnis, memahami perilaku pelanggan, atau mengidentifikasi peluang baru. Proses data mining umumnya melibatkan beberapa langkah yang dirancang untuk mengambil data mentah dan mengubahnya menjadi informasi yang berguna.



Gambar 1. Proses Data Mining

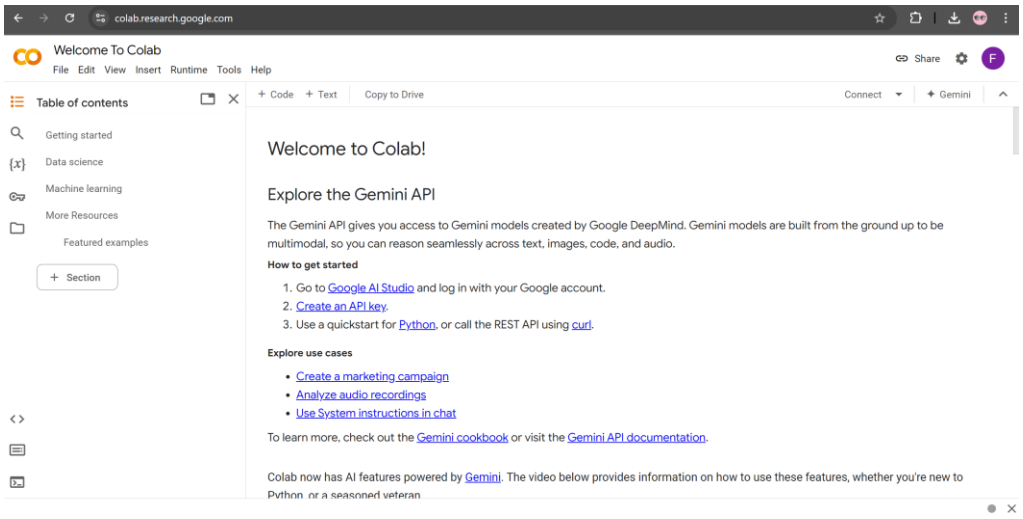
Dalam proses data mining, tahapan pertama yang harus dilakukan adalah Data Understanding dan Data Preparation. Untuk itu Praktikum pertama mata kuliah Data Mining akan membahas tentang Data Understanding dan konsep dasar Data Preparation (Data Preprocessing).

Tujuan Praktikum

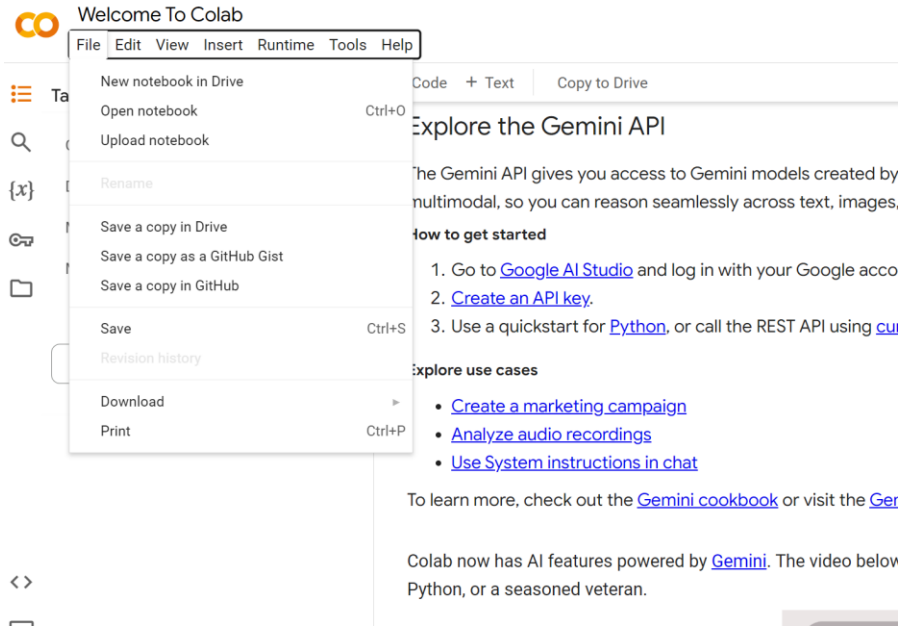
1. Memahami konsep dasar tentang data mining
2. Mempelajari konsep Data Understanding yang mencakup mendeskripsikan data (objek data, type data, atribute data, baris dan kolom) dan eksplorasi data
3. Mempelajari konsep dasar Data Preparation yang mencakup verifikasi kualitas data (Data Preprocessing).

Pada praktikum Data Mining, akan digunakan IDE Python Service dari Google Colab. Praktikan dipersilahkan menyiapkan akun google agar dapat mengakses Google Colab. Langkah-langkah membuka Google Colab dapat dilakukan dengan cara berikut:

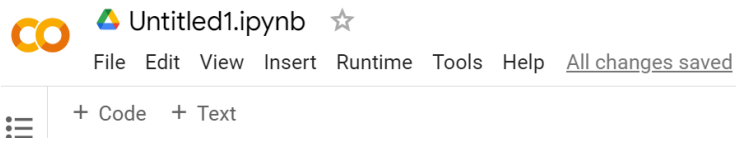
- Pergi ke laman platfrom google colabs yaitu https://colab.research.google.com/
- Akan ditampilkan laman google colabs seperti berikut



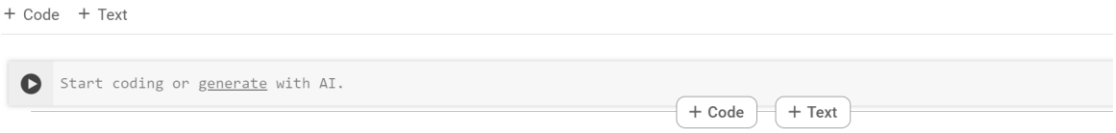
- Klik File lalu pilih New notebook in Drive



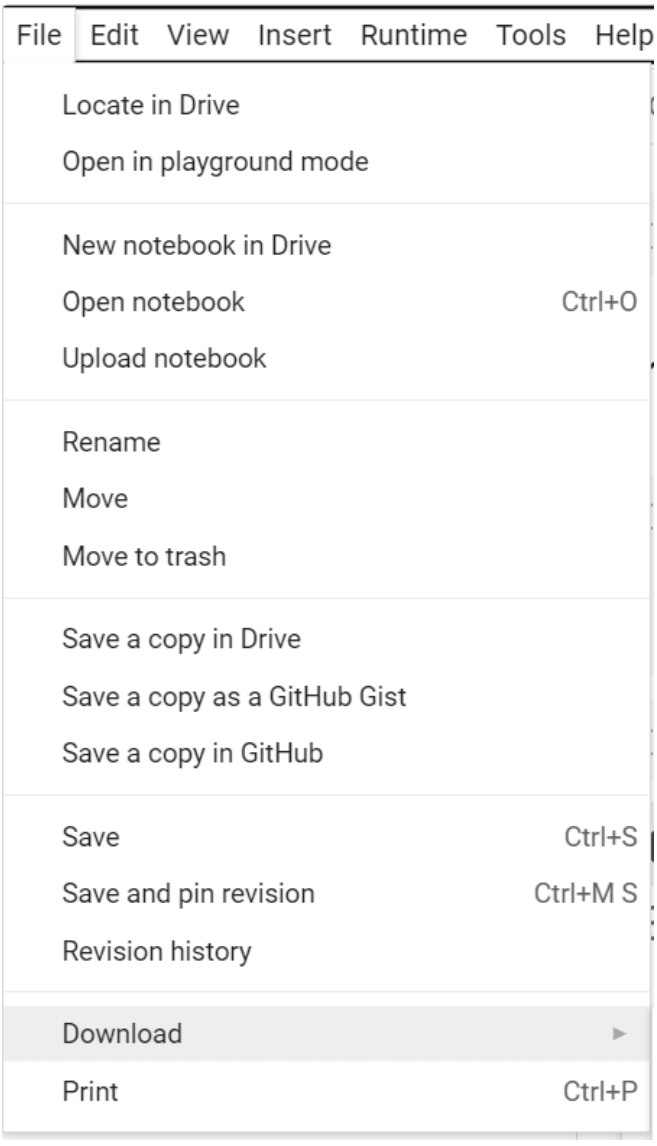
- Ubahlah nama notebook pada “Untitled1.ipynb” menjadi “Praktikum1_DataMining_NIM.ipynb”

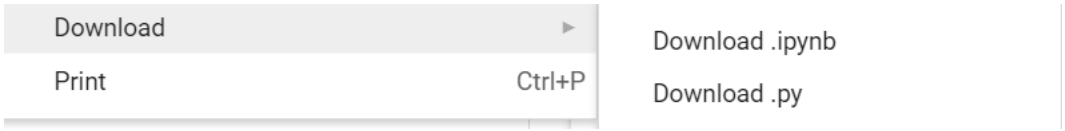


- Klik + Code untuk menambahkan baris coding



- Silahkan mulai Praktikumnya, lakukan perubahan nama untuk setiap kali praktikan melakukan praktikum dengan mengubah angka pada nama sesuai pertemuan praktikum.
- Setelah praktikum pastikan praktikan mendownload codingan beserta hasilnya dengan cara ke menu File – Lalu Downlod – pilih download “.ipynb”





- Selanjutnya file hasil kodingan dapat digunakan untuk tugas praktikum atau yang lainnya.

Data Preprocessing

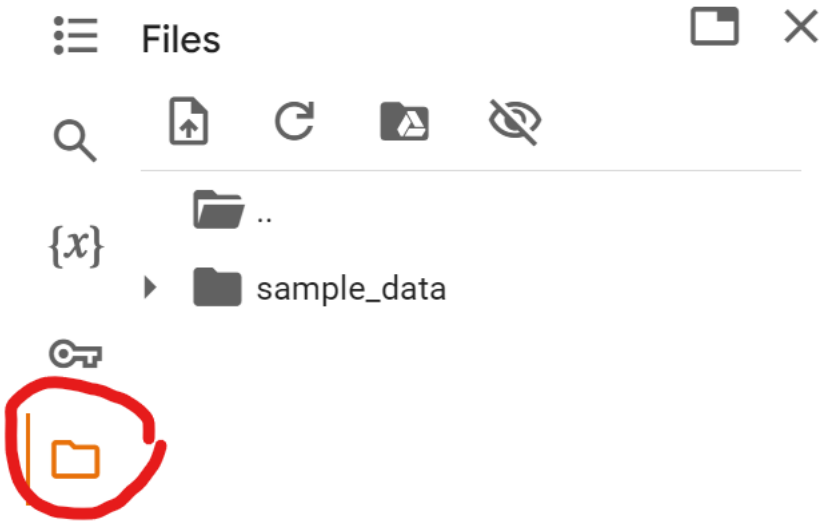
Proses Data Mining seperti Data Understanding dan Data Preparation menggunakan Python seringkali menggunakan beberapa pustaka seperti Pandas, NumPy, Matplotlib, dan Seaborn. Tabel 1 merupakan penjelasan terkait masing-masing pustaka.

Tabel 1. Pustaka Python untuk Preprocessing Data

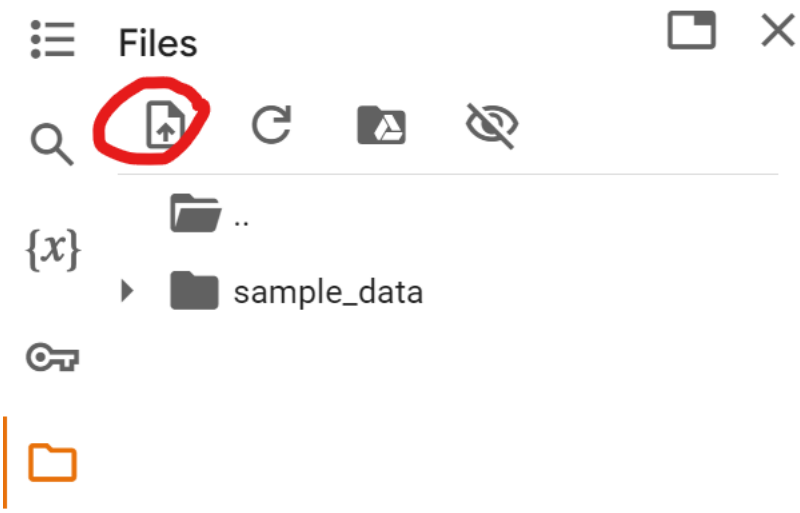
Kategori	Library	Fungsi Utama	Contoh Penggunaan
Numerik	`NumPy`	Operasi numerik dan array multidimensi	`np.array([1, 2, 3])`
Manipulasi Data	`Pandas`	Membaca, mengolah, dan menganalisis data tabular (CSV, Excel)	`pd.read_csv('data.csv')`
Visualisasi	`Matplotlib`	Plot dasar (line, scatter, bar)	`plt.plot(x, y)`
	`Seaborn`	Visualisasi statistik yang menarik	`sns.boxplot(x='kolom', y='nilai', data=df)`

Berikut tahapan-tahapan dalam Data Understanding dan Data Preparation.

1. Upload datasets ke Google Colab



Klik gambar file seperti gambar diatas lalu klik Upload to session storage



Lalu pilih file yang diunduh pada link https://drive.google.com/file/d/1Mfv4xwkcwDLHn5H0YlbFtf8wWzYnyB7s/view?usp=sharing

2. Mengimport Pustaka

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

3. Load datasets yang sudah diupload

```
df=pd.read_csv("diabetes.csv") #membuka file csv
```

4. Menampilkan 5 baris pertama dari Data Frame

```
df.head() #Menampilkan 5 baris pertama dari Data Frame
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	7	159	64	0	0	27.4	0.294	40	0
1	0	180	66	39	0	42.0	1.893	25	1
2	1	146	56	0	0	29.7	0.564	29	0
3	2	71	70	27	0	28.0	0.586	22	0
4	7	103	66	32	0	39.1	0.344	31	1

5. Menampilkan 5 baris terbawah

```
df.tail() #menampilkan 5 baris terbawah
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
15	0	105	64	41	142	41.5	0.173	22	0
16	2	84	0	0	0	0.0	0.304	21	0
17	8	133	72	0	0	32.9	0.270	39	1
18	5	44	62	0	0	25.0	0.587	36	0
19	2	141	58	34	128	25.4	0.699	24	0

6. Menampilkan baris “n” secara random

```
df.sample(5) #menampilkan baris "n" secara random
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
9	8	176	90	34	300	33.7	0.467	58	1
14	0	146	82	0	0	40.5	1.781	44	0
13	0	100	88	60	110	46.8	0.962	31	0
7	1	101	50	15	36	24.2	0.526	26	0
4	7	103	66	32	0	39.1	0.344	31	1

7. Memeriksa jumlah baris dan kolom

```
df.shape #memeriksa jumlah baris dan kolom
```

```
(20, 9)
```

8. Memilih salah satu kolom dari datasets

`df["nama_atribut"]`

```
df["BMI"] #ambil kolom BMI
```

	BMI
0	27.4
1	42.0
2	29.7
3	28.0
4	39.1
5	0.0
6	19.4
7	24.2
8	24.4
9	33.7
10	34.7
11	23.0
12	37.7
13	46.8

9. Memilih baris ke-m sampai ke-n

`df.iloc[m:n+1]`

```
df.iloc[1:4]  #memilih baris ke 1 sampai 3
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
1	0	180	66	39	0	42.0	1.893	25	1
2	1	146	56	0	0	29.7	0.564	29	0
3	2	71	70	27	0	28.0	0.586	22	0

Kode df.iloc[1:4] mengambil baris dari index ke-1 sampai ke-3 (ingat kembali bahwa index dimulai dari 0). Sehingga, data yang diambil yaitu:

Baris index 1: Baris kedua.

Baris index 2: Baris ketiga.

Baris index 3: Baris keempat.

Index 4 tidak terpilih karena batas akhir tidak termasuk.

10. Filter Data Sesuai Kondisi

```
filtered = df[df["Age"] > 35]

filtered
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
0	7	159	64	0	0	27.4	0.294	40	0
9	8	176	90	34	300	33.7	0.467	58	1
10	7	150	66	42	342	34.7	0.718	42	0
12	7	187	68	39	304	37.7	0.254	41	1
14	0	146	82	0	0	40.5	1.781	44	0
17	8	133	72	0	0	32.9	0.270	39	1
18	5	44	62	0	0	25.0	0.587	36	0

Kode ini buat filter data berdasarkan kondisi:

- `df["Age"]>35`

  Cek baris mana saja yang nilai Age-nya lebih dari 35. Hasilnya boolean (True/False) per baris.

- `df[Condition]`

  Ambil semua baris yang kondisinya True (Age > 35).

Hasil akhirnya: DataFrame baru, hanya baris yang sesuai filter.

11. Melihat informasi dasar tentang dataset

```
df.info() #melihat informasi dasar tentang dataset
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20 entries, 0 to 19
Data columns (total 9 columns):
 #   Column                   Non-Null Count  Dtype  
---  ------                   --------------  -----  
 0   Pregnancies              20 non-null     int64  
 1   Glucose                  20 non-null     int64  
 2   BloodPressure            20 non-null     int64  
 3   SkinThickness            20 non-null     int64  
 4   Insulin                  20 non-null     int64  
 5   BMI                      20 non-null     float64
 6   DiabetesPedigreeFunction 20 non-null     float64
 7   Age                      20 non-null     int64  
 8   Outcome                  20 non-null     int64  
dtypes: float64(2), int64(7)
memory usage: 1.5 KB
```

12. Menampilkan jenis data tiap kolom

```
df.dtypes #menampilkan jenis data tiap kolom
```

	0
Pregnancies	int64
Glucose	int64
BloodPressure	int64
SkinThickness	int64
Insulin	int64
BMI	float64
DiabetesPedigreeFunction	float64
Age	int64
Outcome	int64

dtype: object

13. Melihat statistik deskriptif dari data numerik

```
df.describe() #melihat statistik deskriptif dari data numerik
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outc
count	20.000000	20.000000	20.000000	20.000000	20.000000	20.000000	20.000000	20.000000	20.00
mean	3.550000	119.750000	60.900000	20.250000	73.350000	28.770000	0.590400	31.400000	0.25
std	3.136794	39.803299	23.463073	18.970545	114.471911	12.371746	0.469685	9.981035	0.44
min	0.000000	44.000000	0.000000	0.000000	0.000000	0.000000	0.173000	21.000000	0.00
25%	1.000000	97.000000	57.500000	0.000000	0.000000	24.350000	0.301500	23.500000	0.00
50%	2.000000	105.000000	66.000000	18.000000	0.000000	28.850000	0.479000	29.500000	0.00
75%	7.000000	147.000000	70.500000	35.250000	114.500000	38.050000	0.615000	39.250000	0.25
max	8.000000	187.000000	90.000000	60.000000	342.000000	46.800000	1.893000	58.000000	1.00

14. Menampilkan deskripsi data berupa nilai perhitungan statistik pada atribut tertentu

`df['nama_attribute'].describe()`

```
df['Pregnancies'].describe() #menampilkan deskripsi data berupa nilai perhitungan statistik pada atribut tertentu
```

	Pregnancies
count	20.000000
mean	3.550000
std	3.136794
min	0.000000
25%	1.000000
50%	2.000000
75%	7.000000
max	8.000000

dtype: float64

15. Menampilkan komposisi semua data

```
df.value_counts() #menampilkan komposisi semua data
```

Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome	count
0	100	88	60	110	46.8	0.962	31	0	1
	105	64	41	142	41.5	0.173	22	0	1
	146	82	0	0	40.5	1.781	44	0	1
	180	66	39	0	42.0	1.893	25	1	1
1	73	50	10	0	23.0	0.248	21	0	1
	101	50	15	36	24.2	0.526	26	0	1
	103	80	11	82	19.4	0.491	22	0	1
	146	56	0	0	29.7	0.564	29	0	1
2	71	70	27	0	28.0	0.586	22	0	1
	84	0	0	0	0.0	0.304	21	0	1
	141	58	34	128	25.4	0.699	24	0	1
5	44	62	0	0	25.0	0.587	36	0	1

16. Menampilkan komposisi data suatu atribut

```
df['nama_attribute'].value_counts()
```

```
df['Insulin'].value_counts() #menampilkan komposisi data suatu atribut
```

Insulin	count
0	11
82	1
36	1
23	1
300	1
342	1
304	1
110	1
142	1
128	1

dtype: int64

Selanjutnya akan dilakukan data preprocessing yaitu verifikasi data seperti identifikasi nilai yang hilang, identifikasi duplikat data, identifikasi outlier, dan identifikasi data imbalance. Berikut cara untuk mendeteksi semua masalah data yang biasanya terdapat pada data mentah/*raw data*.

1. Identifikasi nilai yang hilang (Missing Values)

```
df.isnull().sum() #menampilkan jumlah nilai yang hilang di setiap kolom
```

	0
Pregnancies	0
Glucose	0
BloodPressure	0
SkinThickness	0
Insulin	0
BMI	0
DiabetesPedigreeFunction	0
Age	0
Outcome	0

dtype: int64

2. Identifikasi Duplikat Data

`df.groupby('nama_attribut')[' nama_attribut'].agg("count")`

```
df.groupby('BMI')['BMI'].agg("count") #identifikasi duplikat data pada attribut tertentu
```

	BMI
BMI	
0.0	2
19.4	1
23.0	1
24.2	1
24.4	1
25.0	1
25.4	1
27.4	1
28.0	1
29.7	1
32.9	1
33.7	1
34.7	1

```
duplikat = df[df.duplicated(keep=False)] #identifikasi baris duplikat
print("Baris Duplikat:")
print(duplikat)
```

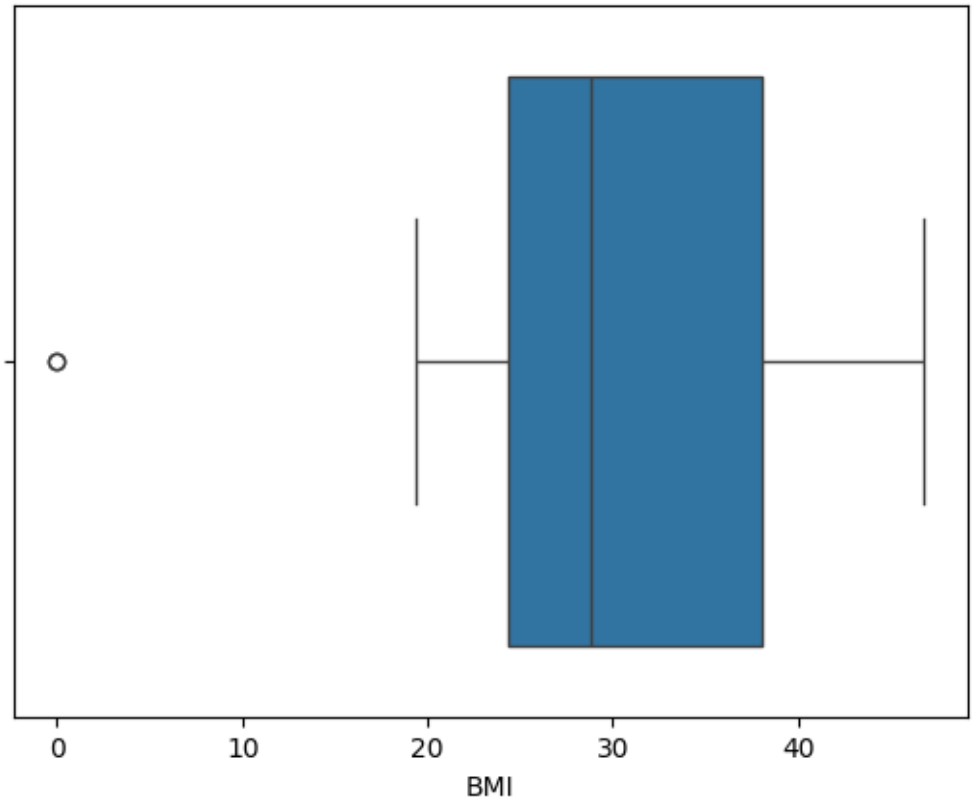
```
Baris Duplikat:
Empty DataFrame
Columns: [Pregnancies, Glucose, BloodPressure, SkinThickness, Insulin, BMI, DiabetesPedigreeFunction, Age, Outcome]
Index: []
```

3. Identifikasi Outlier

```
correlation_matrix = df.corr() #menghitung korelasi antar kolom
sns.heatmap(correlation_matrix) #menampilkan matriks korelasi sebagai heatmap
print(correlation_matrix) #menampilkan nilai korelasi
plt.show() #menampilkan plot
sns.boxplot(x='BMI',data=df) #menampilkan data sebagai boxplot
```

```
                          Pregnancies   Glucose  BloodPressure  SkinThickness  \
Pregnancies                  1.000000  0.241438      -0.034254      -0.098839
Glucose                      0.241438  1.000000       0.299956       0.253455
BloodPressure               -0.034254  0.299956       1.000000       0.447852
SkinThickness               -0.098839  0.253455       0.447852       1.000000
Insulin                      0.322928  0.500616       0.332613       0.615901
BMI                         -0.118628  0.408949       0.831167       0.618654
DiabetesPedigreeFunction    -0.463170  0.322062       0.312071       0.171401
Age                          0.596106  0.550986       0.499109       0.063654
Outcome                      0.462655  0.536493       0.290329       0.266971

                            Insulin       BMI  DiabetesPedigreeFunction  \
Pregnancies                0.322928 -0.118628                 -0.463170
Glucose                    0.500616  0.408949                  0.322062
BloodPressure              0.332613  0.831167                  0.312071
SkinThickness              0.615901  0.618654                  0.171401
Insulin                    1.000000  0.317674                 -0.113869
BMI                        0.317674  1.000000                  0.422357
DiabetesPedigreeFunction -0.113869  0.422357                  1.000000
```



Berdasarkan grafik boxplot, terdapat outlier pada datasets tersebut.

4. Identifikasi data tidak berimbang (Data Imbalanced)

```
df['Outcome'].unique() #menampilkan variabel pada atribut Outcome
```

```
array([0, 1])
```

```
df['Outcome'].value_counts() #menampilkan komposisi pada atribut Outcome
```

Outcome	count
0	15
1	5

dtype: int64

Setelah mengidentifikasi missing values, duplikat data, outlier data dan data imbalance, selanjutnya akan dilakukan penanganan seperti data cleaning, data integrasi, data transformation, dan data reduksi untuk memperbaiki data error tersebut. Data cleaning, data integrasi, data transformation, dan data reduksi akan dijelaskan pada praktikum selanjutnya.