Assignment 1 - Data Wrangling I (Data Science)

Atharva Taras (TE A - 73)

1. Import all the required Python Libraries.

In [1]:

```python
import pandas as pd
```

2. Locate open-source data from the web (e.g. https://www.kaggle.com (https://www.kaggle.com)).

Dataset - https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database?resource=download (https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database?resource=download)

3. Load the Dataset into pandas dataframe.

In [2]:

```python
data = pd.read_csv('diabetes.csv')
```

4. Data Preprocessing

In [3]:

```python
data.head()
```

Out[3]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunct
0	6	148	72	35	0	33.6	0.
1	1	85	66	29	0	26.6	0.
2	8	183	64	0	0	23.3	0.
3	1	89	66	23	94	28.1	0.
4	0	137	40	35	168	43.1	2.

In [4]:

```python
data.shape
```

Out[4]:

```
(768, 9)
```

In [5]:

```
data.describe()
```

Out[5]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	Diab
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	
mean	3.845052	120.894531	69.105469	20.536458	79.799479	31.992578	
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	

In [6]:

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 768 entries, 0 to 767
Data columns (total 9 columns):
 #   Column                    Non-Null Count  Dtype  
---  ------                    --------------  -----  
 0   Pregnancies               768 non-null    int64  
 1   Glucose                   768 non-null    int64  
 2   BloodPressure             768 non-null    int64  
 3   SkinThickness             768 non-null    int64  
 4   Insulin                   768 non-null    int64  
 5   BMI                       768 non-null    float64
 6   DiabetesPedigreeFunction  768 non-null    float64
 7   Age                       768 non-null    int64  
 8   Outcome                   768 non-null    int64  
dtypes: float64(2), int64(7)
memory usage: 54.1 KB
```

5. Data Formatting and Data Normalization

In [7]:

```
data.isnull().sum()
```

Out[7]:

```
Pregnancies                 0
Glucose                     0
BloodPressure               0
SkinThickness               0
Insulin                     0
BMI                         0
DiabetesPedigreeFunction    0
Age                         0
Outcome                     0
dtype: int64
```

In [8]:

```
(data['SkinThickness'] == 0).sum()
```

Out[8]:

```
227
```

In [9]:

```
data = data.replace({'SkinThickness': {0: data['SkinThickness'].mean()}})
```

In [10]:

```
data['SkinThickness'][:10]
```

Out[10]:

```
0    35.000000
1    29.000000
2    20.536458
3    23.000000
4    35.000000
5    20.536458
6    32.000000
7    20.536458
8    45.000000
9    20.536458
Name: SkinThickness, dtype: float64
```

6. Turn categorical variables into quantitative variables in Python.

BMI Categorical Data Source - CDC.gov (https://www.cdc.gov/obesity/basics/adult-defining.html#:~:text=If%20your%20BMI%20is%20less,falls%20within%20the%20obesity%20range.)

1 - Underweight
2 - Healthy Weight
3 - Overweight
4 - Obese

In [11]:

```
def bmi_category(BMI):

    if BMI <= 18.5:
        return 1

    elif BMI <= 25:
        return 2

    elif BMI <= 30:
        return 3

    else:
        return 4
```

In [12]:

```
tmplist = []

for value in data['BMI']:
        tmplist.append(bmi_category(value))

data['BMI Category'] = tmplist
```

In [20]:

```
print(data['BMI'][:5], '\n\n', data['BMI Category'][:5])
```

```
0    33.6
1    26.6
2    23.3
3    28.1
4    43.1
Name: BMI, dtype: float64

 0    4
1    3
2    2
3    3
4    4
Name: BMI Category, dtype: int64
```

In [ ]:

```

```