Data Analysis Project Using the Diabetes DataSet

Source of data: https://www.kaggle.com/uciml/pima-indians-diabetes-database

File Format: CSV (comma separated value) format

About this Dataset

Context: This dataset is originally from the National Institute of Diabetes and Digestive and Kidney Diseases. The objective of the dataset is to diagnostically predict whether or not a patient has diabetes, based on certain diagnostic measurements included in the dataset. Several constraints were placed on the selection of these instances from a larger database. In particular, all patients here are females at least 21 years of age of Pima Indian heritage

In [1]:
```
# IMPORT PANDAS LIBRARY
import pandas as pd
```

In [2]:
```
#LODAING THE DATASET
file = 'diabetes.csv'
```

In [3]:
```
#READING THE DATASET
df = pd.read_csv(file)
```

After reading the dataset, I used **dataframe.head(n)** method to check the top n rows of the dataframe, where n is an integer.

In [4]:
```
# view the first 5 rows using dataframe.head() method
print("The first 5 rows of the dataframe")
df.head(5)
```

```
The first 5 rows of the dataframe
```

Out[4]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	O
0	6	148	72	35	0	33.6	0.627	50	
1	1	85	66	29	0	26.6	0.351	31	
2	8	183	64	0	0	23.3	0.672	32	
3	1	89	66	23	94	28.1	0.167	21	
4	0	137	40	35	168	43.1	2.288	33	

Also, I used **dataframe.tail(n)P** method to check the bottom n rows of the dataframe, where n is an integer.

In [5]:
```
# view the last 5 rows using dataframe.head() method
print("The last 5 rows of the dataframe")
df.tail(5)
```

```
The last 5 rows of the dataframe
```

Out[5]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
763	10	101	76	48	180	32.9	0.171	63
764	2	122	70	27	0	36.8	0.340	27
765	5	121	72	23	112	26.2	0.245	30
766	1	126	60	0	0	30.1	0.349	47
767	1	93	70	31	0	30.4	0.315	23

To view the dimensions of the dataframe, I used the **.shape** parameter.

In [9]:
```
print ('The number of rows and columns of the dataframe')
df.shape
```

```
The number of rows and columns of the dataframe
(768, 9)
```

Out[9]:

Content of the dataset: The datasets consists of several medical predictor variables and one target variable, Outcome. Predictor variables includes the number of pregnancies the patient has had, their BMI, insulin level, age etc.

And We have 768 rows and 9 columns. The first 8 columns represent the features and the last column represent the target/label.

Statistical Overview of dataset

In [11]:
```
df.info()

#To print information about the DataFrame including the index dtype and columns, non-nu
```

```
<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
```

In [12]:
```
df.describe()

#To view some basic statistical details like percentile, mean, standard deviation.
```

Out[12]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreel
count	768.000000	768.000000	768.000000	768.000000	768.000000	768.000000	76
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	

Identify and handle missing values

In [13]:
```
missing_data = df.isnull()
missing_data.head(5)
```

Out[13]:

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age
0	False	False	False	False	False	False	False	False
1	False	False	False	False	False	False	False	False
2	False	False	False	False	False	False	False	False
3	False	False	False	False	False	False	False	False
4	False	False	False	False	False	False	False	False

Count missing values in each column

In [14]:
```
for column in missing_data.columns.values.tolist():
    print(column)
    print (missing_data[column].value_counts())
    print("")
```

```
Pregnancies
False    768
Name: Pregnancies, dtype: int64

Glucose
False    768
Name: Glucose, dtype: int64
```

```
BloodPressure
False    768
Name: BloodPressure, dtype: int64

SkinThickness
False    768
Name: SkinThickness, dtype: int64

Insulin
False    768
Name: Insulin, dtype: int64

BMI
False    768
Name: BMI, dtype: int64

DiabetesPedigreeFunction
False    768
Name: DiabetesPedigreeFunction, dtype: int64

Age
False    768
Name: Age, dtype: int64

Outcome
False    768
Name: Outcome, dtype: int64
```

Check Correct data format

In [16]:
```
df.dtypes
```

Out[16]:
```
Pregnancies                 int64
Glucose                     int64
BloodPressure               int64
SkinThickness               int64
Insulin                     int64
BMI                       float64
DiabetesPedigreeFunction  float64
Age                         int64
Outcome                     int64
dtype: object
```

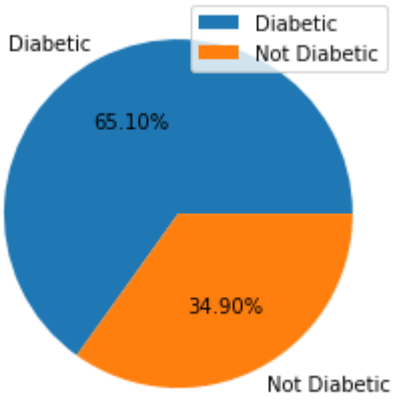
Data Visualization

P.S: The major visualization of data was done in PowerBI

In [18]:
```
# import libraries
import matplotlib.pyplot as plt
%matplotlib inline
```

In [19]:
```
labels= 'Diabetic','Not Diabetic'
plt.pie(df['Outcome'].value_counts(),labels=labels,autopct='%0.02f%%')
```

```
plt.legend()
plt.show()
```



Results: The basic data visualization shows that 65.10% of the sample population are diabetic. While 34.90% of the sample population are not diabetic

Save New data into csv

The new data was saved in CSV and exported to PowerBI for advanced visualization (dashboards).

In [20]:
```
#Save Data in CSV format
df.to_csv('New_diabetes_data.csv')
```