Exploratory Data Analysis (EDA)

1. Introduction

Exploratory Data Analysis (EDA) is the process of scrutinizing datasets to summarize their fundamental characteristics, often utilizing statistical graphics and other visualization methods. The aim of EDA is to uncover patterns within the data to derive meaningful insights. To achieve this objective, exploration is conducted on a particular dataset to identify, understand, and categorize its features.

For instance, let's consider the Pima Indians diabetes dataset. In the subsequent sections, we will delve into the exploration process as outlined.

1.1 Getting the data.

The dataset is readily available in the UCI repository and can be downloaded from the following;

URL. https://www.kaggle.com/datasets/uciml/pima-indians-diabetes-database/

1.1.1 Dataset Context

The source of this dataset is the National Institute of Diabetes and Digestive and Kidney Diseases. The dataset's goal is to predict, diagnostically, whether a patient has diabetes or not, utilizing specific diagnostic measurements provided within it. Certain criteria were applied when selecting these instances from a larger database. Specifically, all the patients included in this dataset are females of at least 21 years of age, and they belong to the Pima Indian heritage.

1.1.2 Dataset Data dictionary:

Below is the attribute information:

- Pregnancies: Number of times pregnant
- Glucose: Plasma glucose concentration a 2 hours in an oral glucose tolerance test
- Blood pressure: Diastolic blood pressure (mm Hg)
- SkinThickness: Triceps skinfold thickness (mm)
- Insulin 2-Hour serum insulin (mu U/ml) test
- BMI: Body mass index (weight in kg/(height in m)^2)
- DiabetesPedigreeFunction: A function that scores the likelihood of diabetes based on family
- history
- Age: Age in years

- Outcome: Class variable (0: the person is not diabetic or 1: the person is diabetic)

Now that we have gained some insight into our dataset and the objective of our analysis (which is to explore the patterns and trends of diabetes within the Pima Indians population), let's delve directly into the analysis

1.2 Objective

The first and foremost important thing in EDA is setting a goal. Our goal is to draw meaningful inferences which in turn will be used in identifying diabetes in a person.

2. Implementation

To achieve the stated objective, we need well set environment for data analysis. In this case, we shall use ANACONDA platform. Using Jupyter.

2.1 Reading Data

Import all the related libraries

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

Python libraries

importing file

```
df=pd.read_csv("diabetes.csv")
```

Csv file

We assign our dataset as df

2.2 Understanding the data

Taking time to go through the data to understand its characteristics and gather information on general knowledge with in regard to the field that re very important during the analysis.

2.2.1 Reviewing the data frame info

```
df.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
```

This gives information on dataset variables, their data types, helps us Identify if there are missing values. We can also tell if the variables are assigned the right data types. In this case, the data types are assigned correctly and there are no nonvalues.

```
df.columns
```

```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

This outputs all the variables or features in our df

2.2.2. View 5 records to better understand the data

```
df.head()
```

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

Returns the first five rows of the dataset. We can also see that all the data is numerical with our target variable being categorical with a numerical attributes of 0 and 1.

From the table, we can point out zero attributes. This variables cannot be zero in the real world. From the table,

Insulin has zero in rows 1, 2, 3, SkinThickness in row 3. In this case we take the values as outliers which ought to be dropped when cleaning the data.

2.2.3. Describe various properties of the data

In this section we produce the descriptive statistics.

```
df.describe()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	768.000000	768.000000	768.000000	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	0.471876	33.240885	0.348958
std	3.369578	31.972618	19.355807	15.952218	115.244002	7.884160	0.331329	11.760232	0.476951
min	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.078000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	0.000000	0.000000	27.300000	0.243750	24.000000	0.000000
50%	3.000000	117.000000	72.000000	23.000000	30.500000	32.000000	0.372500	29.000000	0.000000
75%	6.000000	140.250000	80.000000	32.000000	127.250000	36.600000	0.626250	41.000000	1.000000
max	17.000000	199.000000	122.000000	99.000000	846.000000	67.100000	2.420000	81.000000	1.000000

We get descriptive statistics for all the variables which are all numerical.

2.2.4. Understanding the outcomes

Outcome is our target variable.

```
df["Outcome"].unique()
```

```
array([1, 0], dtype=int64)
```

This indicates that there are 0 and 1 attributes for outcome. 0 for non-diabetic and 1 for diabetic

```
df["Outcome"].value_counts()
```

```
Outcome
0    500
1    268
Name: count, dtype: int64
```

This shows that there are 268 people having diabetes and 500 who don't have the disease.

2.2.5. To check if there are any null values in the data set

```
df.isnull().sum()
```

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

The zeros from this table indicates that there are no missing values. If there were non-values we would have imputed them with mean/ median and just drop them.

2.3. Cleansing the Data

Errors can evolve from duplicated data, missing values, incorrect attributes, inconsistency and irrelevant data or outliers.

We have already observed outliers.

2.3.1. Filtering Non-zero records

We previously observed zeros in variables that can never be zero. We take them as outliers.

We drop all the records from the data frame where either Insulin or Blood Pressure or Skin Thickness or BMI, or Glucose levels are 0, through the following codes

```
df[df['Insulin']!=0]
df[df['Glucose']!=0]
df[df['BloodPressure']!=0]
df[df['SkinThickness']!=0]
df[df['BMI']!=0]
```

2.3.2. Confirming that Zero-valued records are deleted

We run the following commands to see if the above rows have been deleted from the data.

```
df.shape
```

```
(392, 9)
```

Initially the rows were 768 and now they are 329. The column remains constant.

This is another proof that the records were dropped

```
df.describe()
```

	Pregnancies	Glucose	BloodPressure	SkinThickness	Insulin	BMI	DiabetesPedigreeFunction	Age	Outcome
count	392.000000	392.000000	392.000000	392.000000	392.000000	392.000000	392.000000	392.000000	392.000000
mean	3.301020	122.627551	70.663265	29.145408	156.056122	33.086224	0.523046	30.864796	0.331633
std	3.211424	30.860781	12.496092	10.516424	118.841690	7.027659	0.345488	10.200777	0.471401
min	0.000000	56.000000	24.000000	7.000000	14.000000	18.200000	0.085000	21.000000	0.000000
25%	1.000000	99.000000	62.000000	21.000000	76.750000	28.400000	0.269750	23.000000	0.000000
50%	2.000000	119.000000	70.000000	29.000000	125.500000	33.200000	0.449500	27.000000	0.000000
75%	5.000000	143.000000	78.000000	37.000000	190.000000	37.100000	0.687000	36.000000	1.000000
max	17.000000	198.000000	110.000000	63.000000	846.000000	67.100000	2.420000	81.000000	1.000000

We no longer have 0 as the minimum values for Glucose BloodPressure SkinThickness Insulin BMI

2.3.2. Checking if there are duplicated records

```
df.duplicated().sum()
```

```
0
```

This 0 indicates that there are no duplicates.

Having cleaned the data we can now proceed to analysis…..

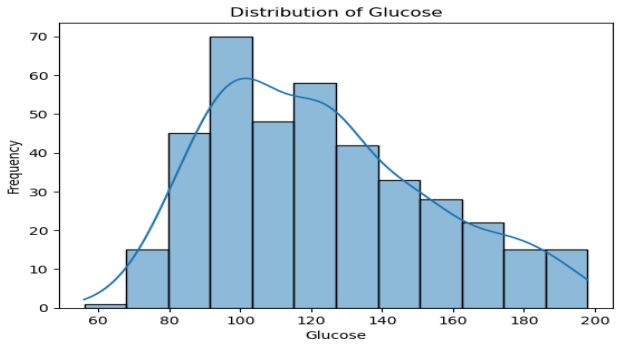
2.4. Analyzing the Data

2.4.1 Univariate analysis

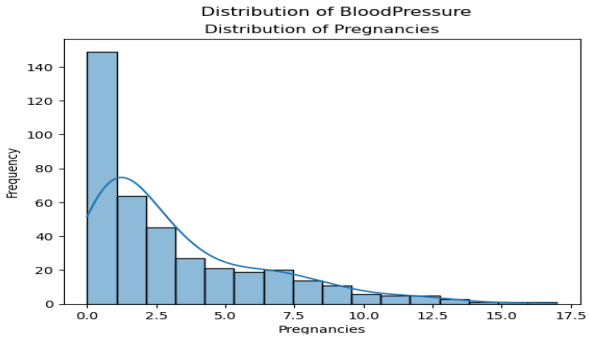
Visualization of each attribute to show its distribution. We start with histogram.

```
Index(['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin',
       'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome'],
      dtype='object')
```

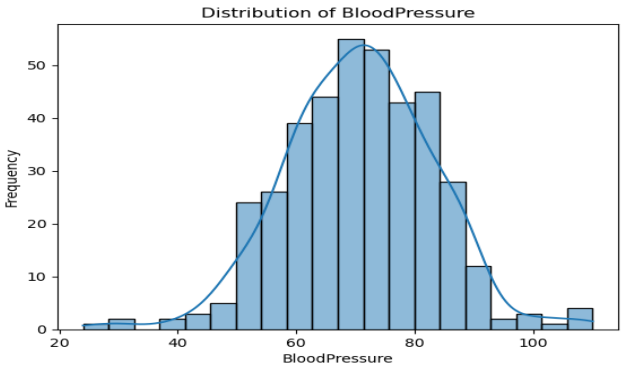
```python
for column in df.columns:
    sns.histplot(df[column],kde =True)  #use kde to better understand the shape of histogram for better analysis
    plt.title(f'Distribution of {column}')
    plt.xlabel(column)
    plt.ylabel('Frequency')
    plt.show()
```

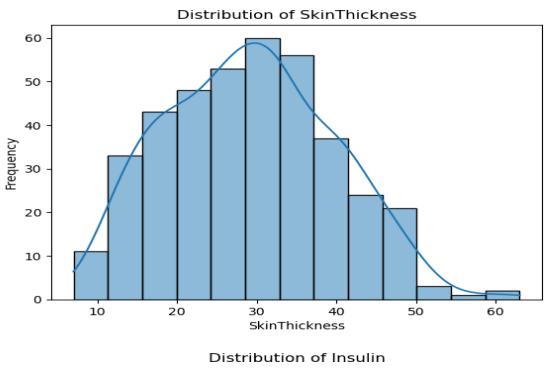


Glucose tend to be fairly symmetrical.

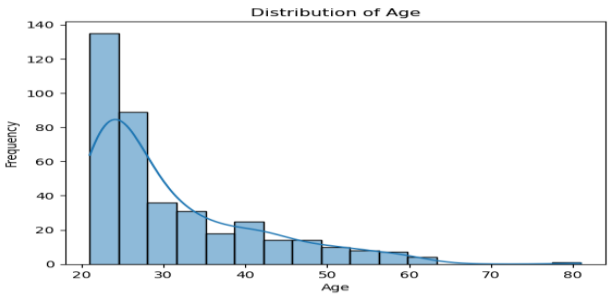


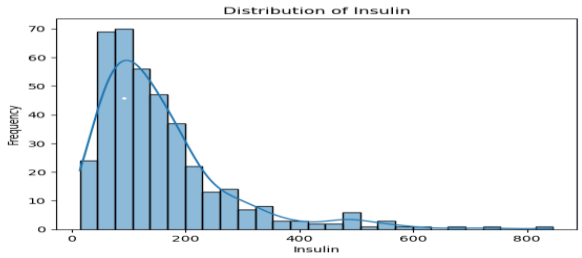
The distribution of pregnancies is right skewed with a higher number rate at 0-3 pregnancies.

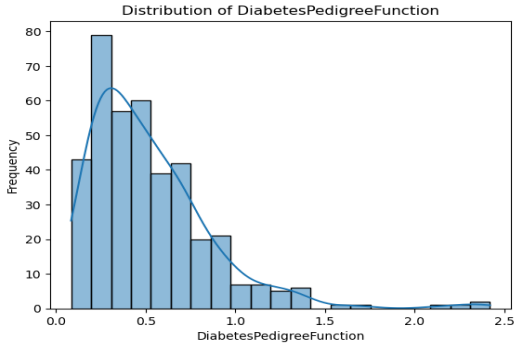


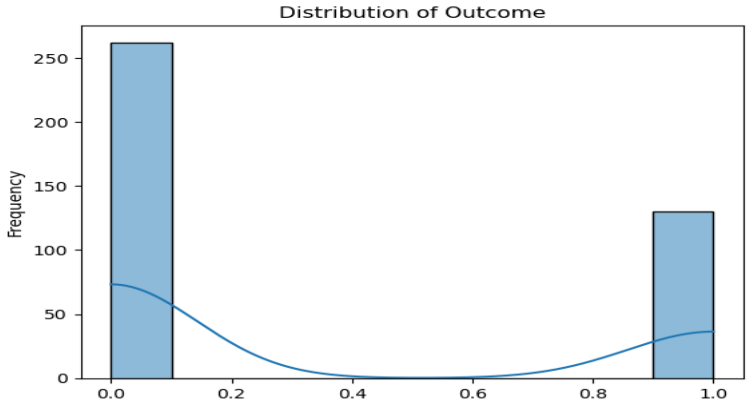


Distribution of Insulin


Distribution of Age
Frequency
Age


Distribution of Insulin
Frequency
Insulin


Distribution of DiabetesPedigreeFunction
Frequency
DiabetesPedigreeFunction


Distribution of Outcome
Frequency

2.4.2 Bivariate Analysis