Subject: Data Intensive Computing
Subject Code: CSE 4/587
Professor: Eric Mikida
School of Engineering and Applied Sciences
University at Buffalo, Buffalo, NY 14260

Project Phase 1

PROJECT TITLE	DATE
Analysis and Prediction of stroke in Human body	10/17/2022

PROJECT TEAM MEMBERS	
Briyana Rana	UBIT Number: 50442498 UB Name: brana
Yashesh Pandya	UBIT Number: 50442549 UB Name: yasheshp

1.) Problem Statement:-

a) Motivation and problem:-

i) Stroke is the second biggest cause of death worldwide, accounting for around 11% of all fatalities, according to the World Health Organization (WHO).

ii) We need a prediction mechanism tool with which we can assume weather a person is likely to get stroke or not.

b) Solution and impact:-

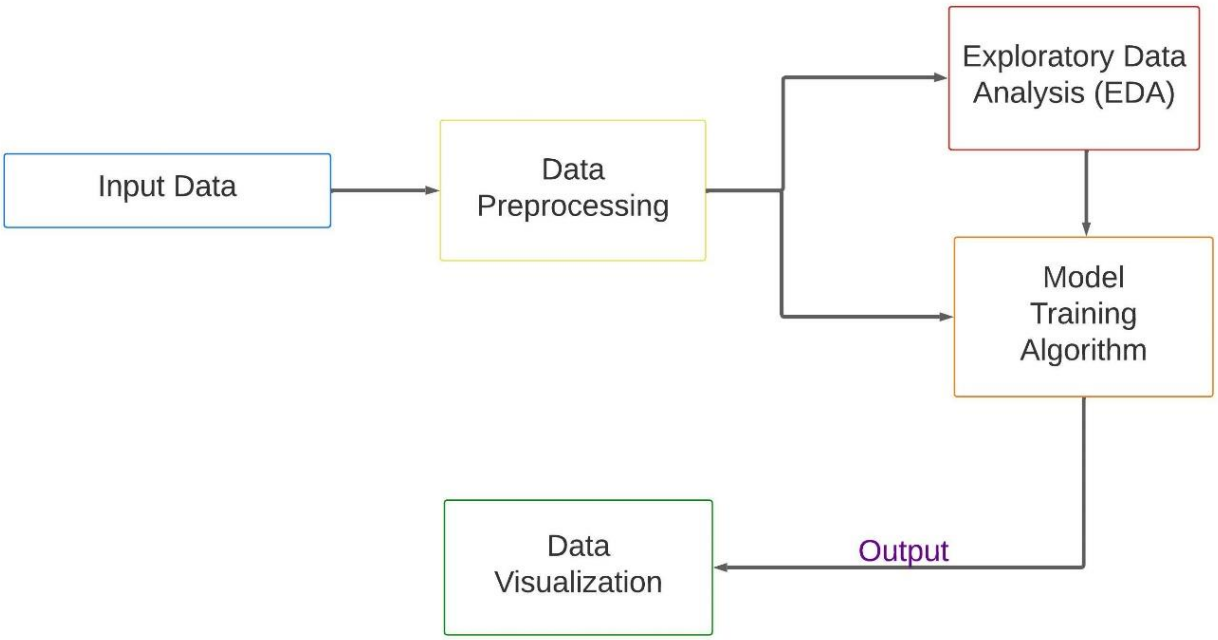
i) Based on input characteristics including gender, age, numerous illnesses, and smoking status, this dataset is used to determine whether a patient is likely to get a stroke. Each row of the data contains pertinent patient information.

ii) So using ML models we will develop a prediction mechanism to predict stroke.

2.) Data Source:-

- We have taken the unclean dataset from Kaggle.
- https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset
- This dataset contains 5110 observations with 12 attributes.
- The column names are id, gender, age, hypertension, heart_disease, ever_married, work_type, Residence_type, avg_glucose_level, bmi, smoking_status, stroke.

column name	Description
Id	It is unique id
Gender	Gender of person
Age	Age of person
hypertension	Hypertension binary feature
heart_disease	Heart disease binary feature
ever_married	Person married or not
work_type	Work type of the patient
Residence_type	Residence type of the patient
avg_glucose_level	Average glucose level in blood
bmi	Body mass index
smoking_status	Smoking status of person
stroke	Stroke status in binary



3.) Data cleaning/Processing: -

1. Removal of unwanted rows:-

- Count the value of “gender” and found 2991 females, 2111 males and 1 other.

```
gender :
Female    2991
Male      2111
Other        1
Name: gender, dtype: int64
```

- We dropped the row which have other value in gender field.

```
df1= df[df.gender != 'Other']
```

```
df1['gender'].value_counts()
```

```
9]: Female    2991
    Male      2111
    Name: gender, dtype: int64
```

2. Remove null value:-

- Remove the null value from the “avg_glucose_level” field.

```
df1['avg_glucose_level']=df1['avg_glucose_level'].dropna()
```

3. Filling NULL values with mean:-

- Replace missing values in “bmi” field with the most frequent value (i.e 28.893237)

```
df1['bmi']=df1['bmi'].fillna(28.893237)
```

4. Simple imputer:-

- Replace missing value in “gender” with most frequent value.

```
from sklearn.impute import SimpleImputer
Imp = SimpleImputer(missing_values= np.NaN,strategy='most_frequent')
df2 = Imp.fit_transform(df1)
```

5. Label Encoding for Residence_type field:-

- Performed label encoding on “gender” and “Residence_type” field.
- For female we have used 0 and for male we have used 1.

```
from sklearn.preprocessing import LabelEncoder
L = LabelEncoder()
```

```
df3['gender'] = L.fit_transform(df3['gender'])
```

```
gender :
0    2998
1    2111
Name: gender, dtype: int64
```

- Similarly, if person belongs to urban then it is 1 and 0 for rural.

```
from sklearn.preprocessing import LabelEncoder
L = LabelEncoder()
```

```
df3['Residence_type'] = L.fit_transform(df3['Residence_type'])
```

```
Residence_type :
1    2596
0    2513
Name: Residence_type, dtype: int64
```

6. Label Encoding for work_type field:-

- Performed label encoding on “work_type” field.
- If a person has private job then it is 2, for self-employed it is 3, for children it is 4, for government job it is 0 and lastly for never worked it is 1.

```
from sklearn.preprocessing import LabelEncoder
L = LabelEncoder()
```

```
df3['work_type'] = L.fit_transform(df3['work_type'])
```

```
work_type :
2    2924
3     819
4     687
0     657
1      22
Name: work_type, dtype: int64
```

7. Label Encoding for smoking _status field:-

- Performed label encoding on “smoking_status” field.
- If a person never smoked then it is 2, for unknown it is 0, for formerly smoked it is 1 and lastly for smokes it is 3.

```
from sklearn.preprocessing import LabelEncoder
L = LabelEncoder()
```

```
df3['smoking_status'] = L.fit_transform(df3['smoking_status'])
```

```
smoking_status :
2    1892
0    1544
1     884
3     789
Name: smoking_status, dtype: int64
```

8. Ceiling on bmi:-

- As the data in column “bmi” was not in standard form, Ceiling was performed.

```
df3['bmi']=df3['bmi'].apply(np.ceil)
```

- Earlier it was in the non-standard form

bmi
36.6
28.893237
32.5
34.4

- Now, it is converted into a standard form

bmi
37
29
33
35

9. Renaming:-

- In the dataset there was an attribute named ‘ever_married’, which did not make any sense hence we changed the name of the attribute from ‘ever_married’ to ‘married’.

```
df.rename(columns= {'ever_married':'married'},inplace= True)
```

10. Dropping Unwanted column:-

- As observed in the correlation matrix , attribute name ‘id’ was highly uncorrelated with the target value ‘stroke’ , so we decided to drop it to reduce the dimension of our dataset.

```
df1=df.drop(columns='id',axis=1)
```

11. One Hot Encoding:-

- First we performed Label Encoding on married and later we did one hot encoding. This will ensure that machine learning does not assume that higher numbers are more important.

```
from sklearn.preprocessing import OneHotEncoder
```

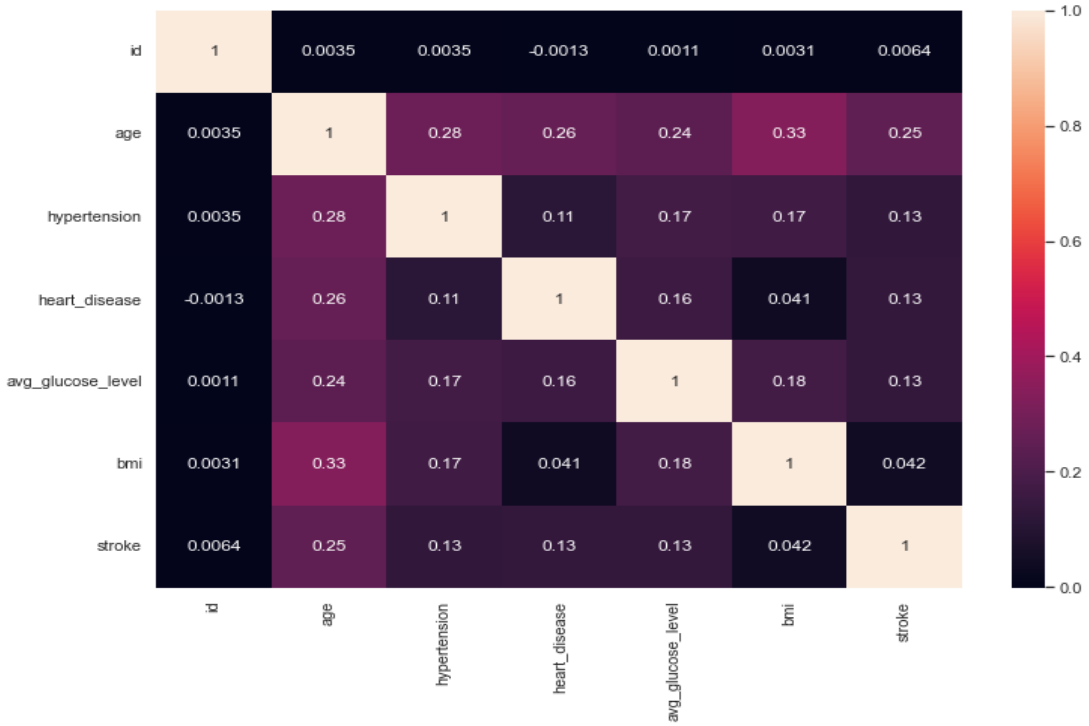
```
H = OneHotEncoder()
```

```
data =pd.DataFrame(H.fit_transform(df3[['married']]).toarray())
```

4.) Exploratory Data Analysis (EDA):-

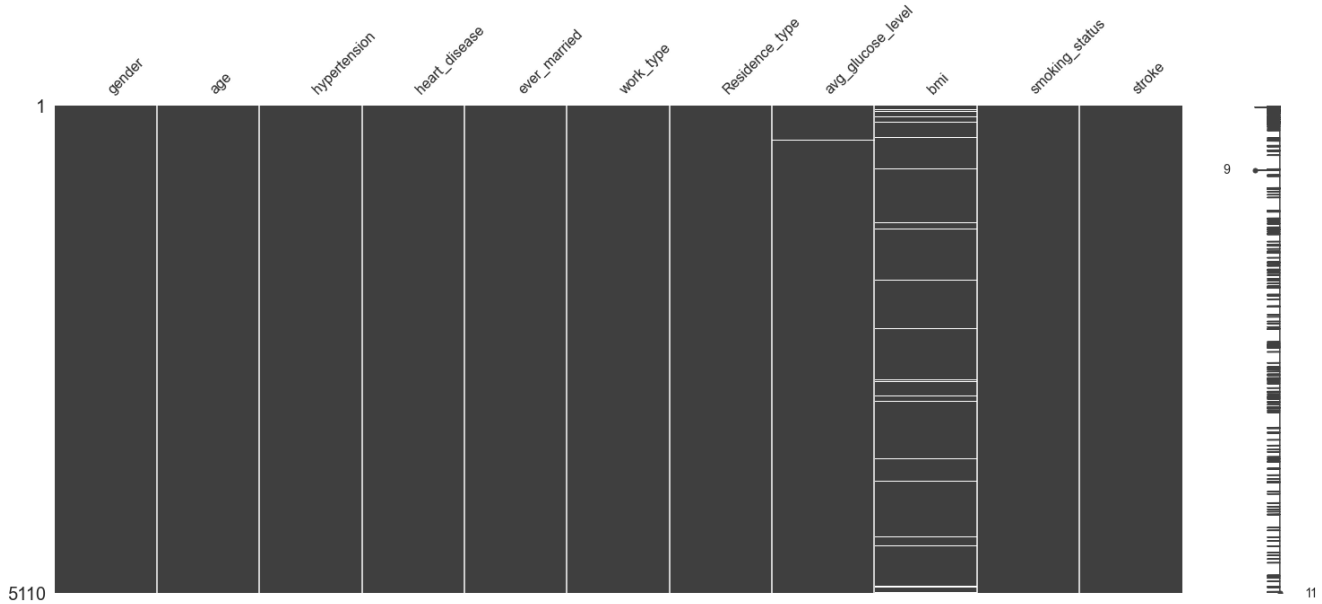
- Exploratory Data Analysis (EDA) is a way of evaluating data sets to highlight their key features, frequently utilizing statistical graphics and other techniques for data visualization.
- EDA differs from traditional hypothesis testing in that it is primarily used to explore what the data can tell us beyond the formal modeling. A statistical model can be utilized or not.
- EDA aims to provide all the particular things that an analyst would wish to extract from a data collection while maximizing the analyst's understanding into a data set and into its underlying structure.

1. Correlation matrix:



- Among all non-categorical attributes, a correlation matrix is formed w.r.t the target attribute [‘STROKE’], as id is having very less correlation with stroke, it was dropped from the dataset. And rest were considered.

2. Displaying null values in the dataset with the help of white lines.

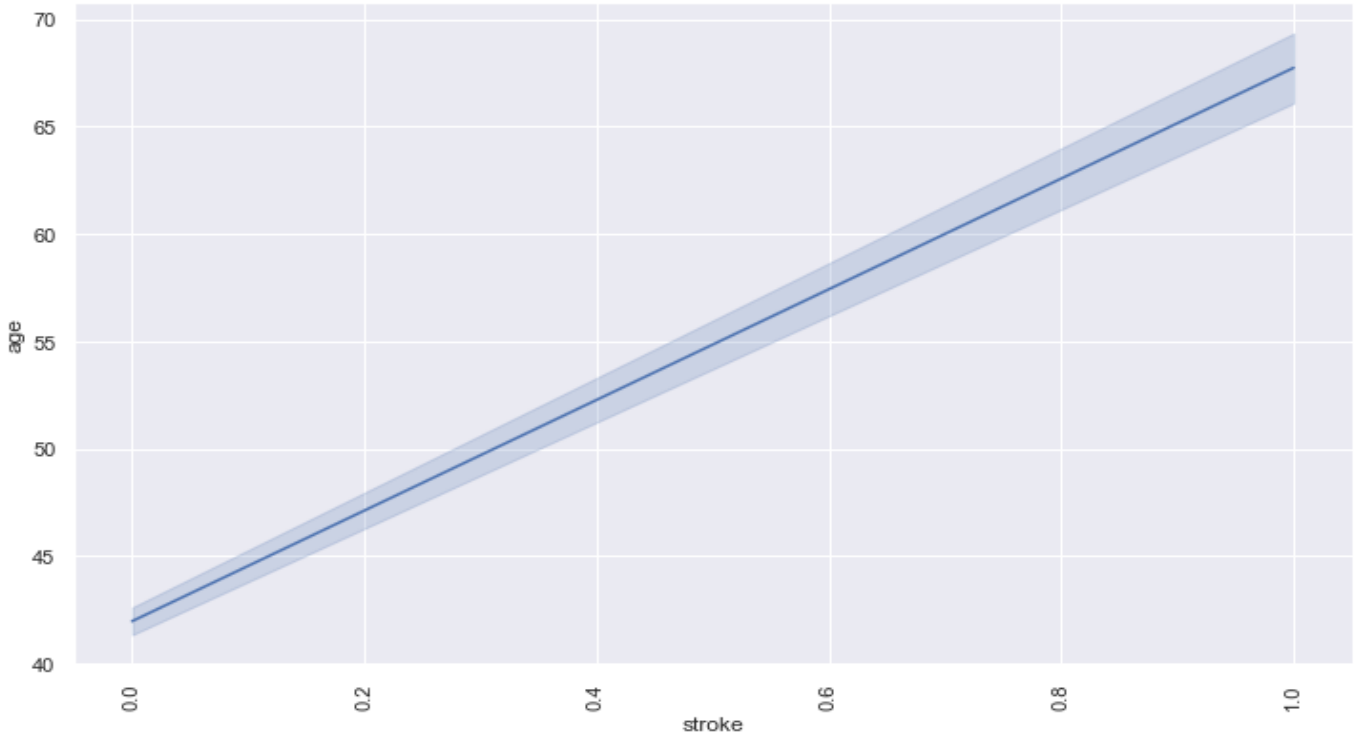


3. Statistical Model:

	id	age	hypertension	heart_disease	avg_glucose_level	bmi	stroke
count	5110.000000	5110.000000	5110.000000	5110.000000	5104.000000	4909.000000	5110.000000
mean	36517.829354	43.226614	0.097456	0.054012	106.162541	28.893237	0.048728
std	21161.721625	22.612647	0.296607	0.226063	45.307349	7.854067	0.215320
min	67.000000	0.080000	0.000000	0.000000	55.120000	10.300000	0.000000
25%	17741.250000	25.000000	0.000000	0.000000	77.222500	23.500000	0.000000
50%	36932.000000	45.000000	0.000000	0.000000	91.850000	28.100000	0.000000
75%	54682.000000	61.000000	0.000000	0.000000	114.160000	33.100000	0.000000
max	72940.000000	82.000000	1.000000	1.000000	271.740000	97.600000	1.000000

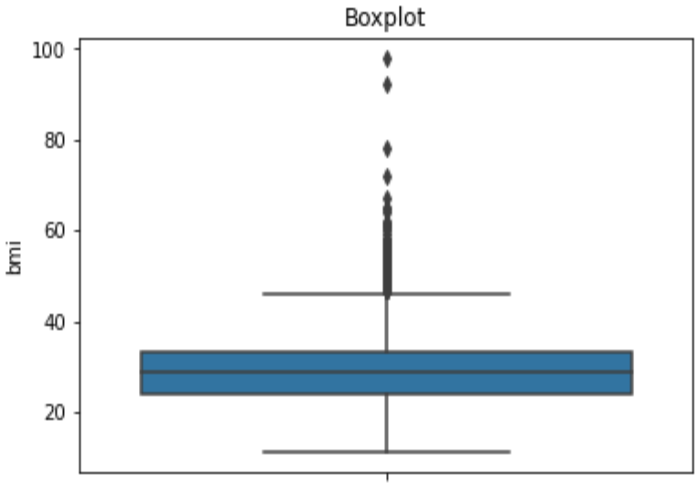
4. Line plot:

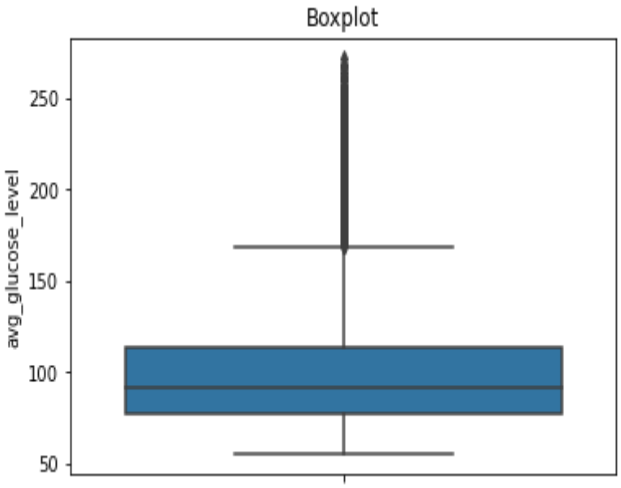
- Here statistical relation is shown among the non-categorical attributes.
- As [‘Age’] is highly correlated with our target attribute [‘stroke’], a line plot is shown which clearly indicates that strokes observed in people were after the age of 40 and they are linearly dependent on each other.

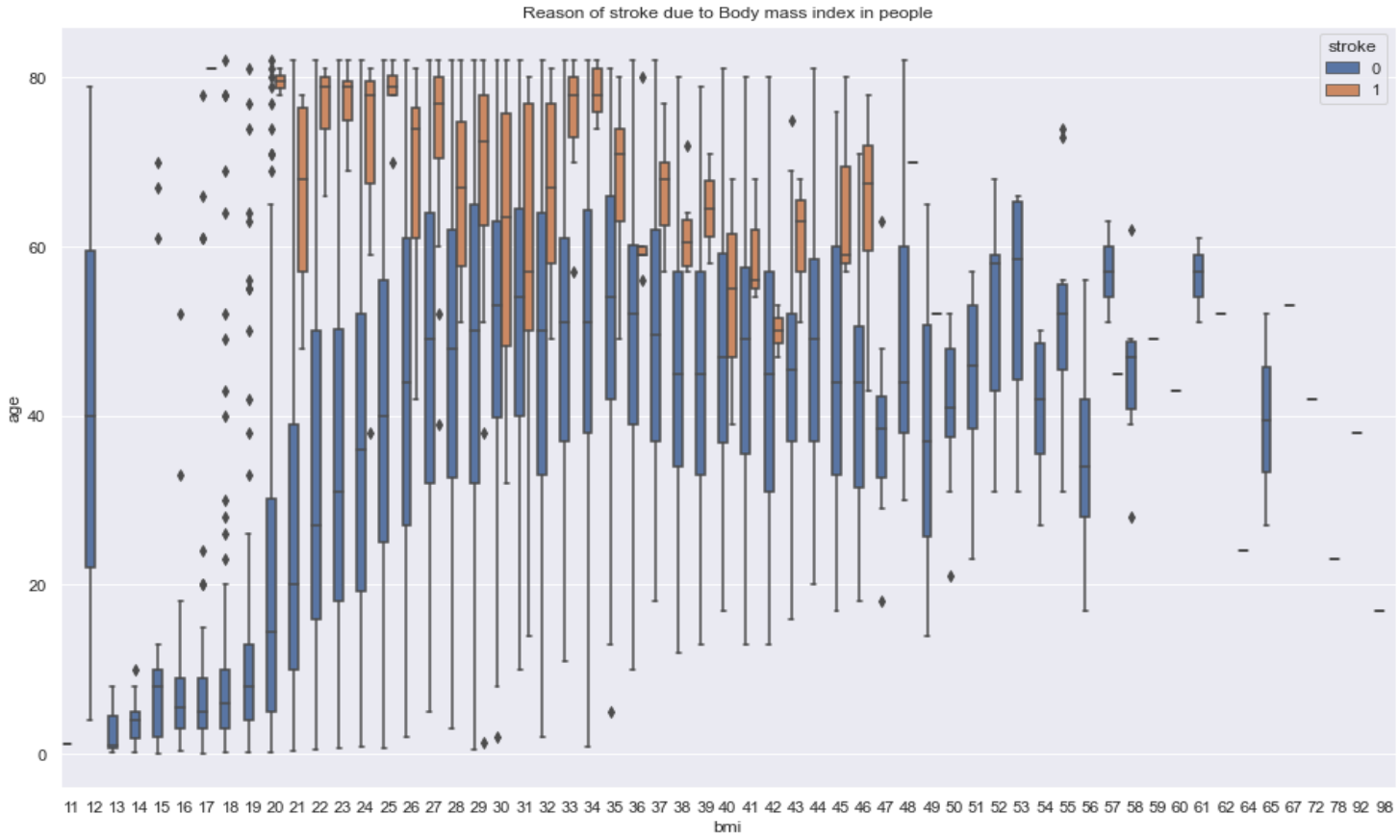


5. Box Plot for the dataframe:

- To observe if the attributes [‘bmi’,’avg_glucose_level’,’age’] have any outliers, boxplot for each is displayed below:

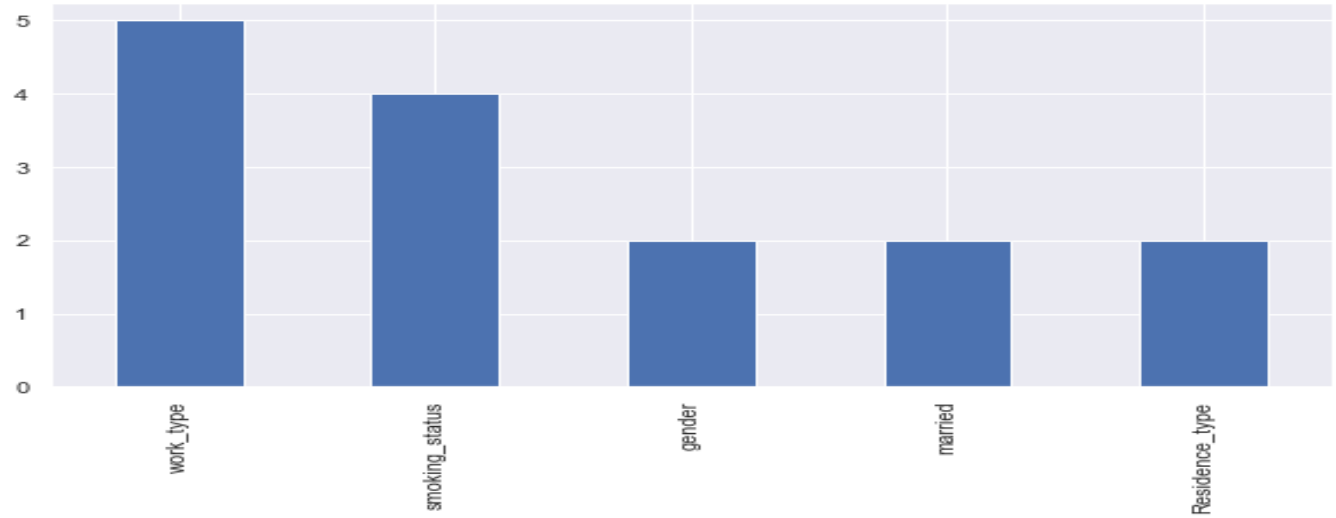






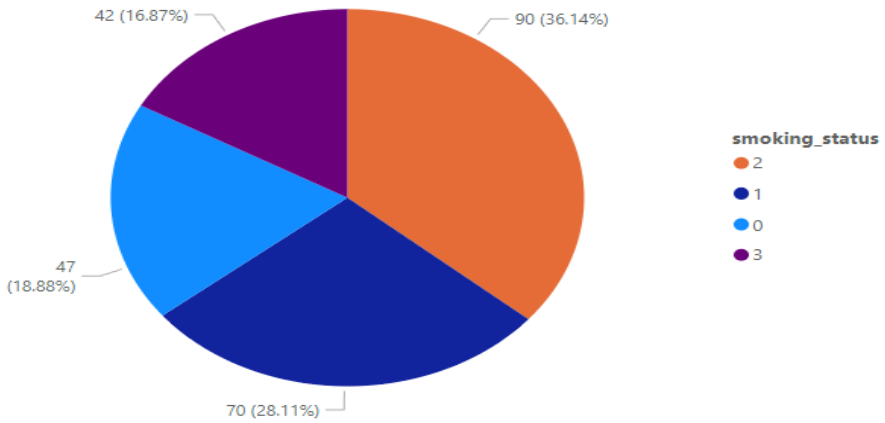
6. Label Encoding:

- To display the number of labels assigned to each categorical attribute , barplot is used as below:



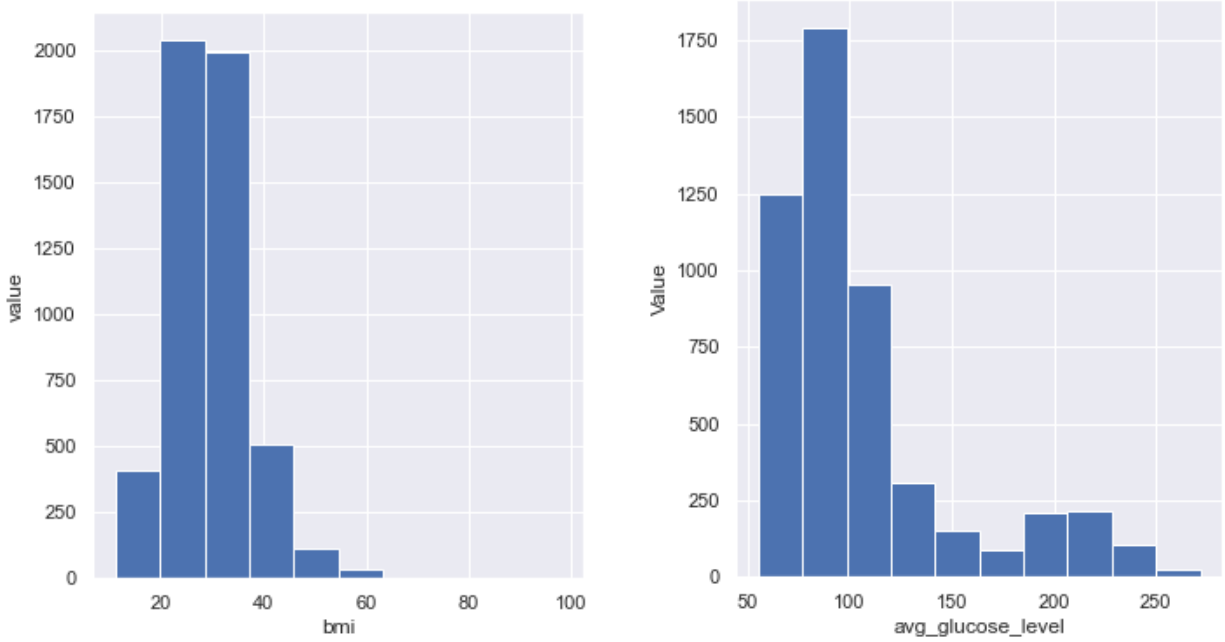
7. Pie chart:

- Here the smoking status 2 indicates that the person never smoked but still had stroke which covers almost 36% of total data. And 28% people formerly smoked and had stroke.
- This plot was created in Microsoft Power BI.



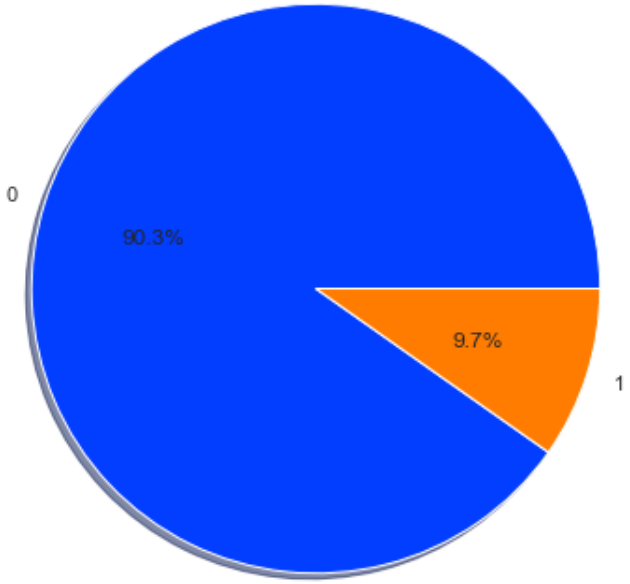
8. Histogram:

- Histograms are created to observe the distribution followed by [‘bmi’,’avg_glucose_level’].



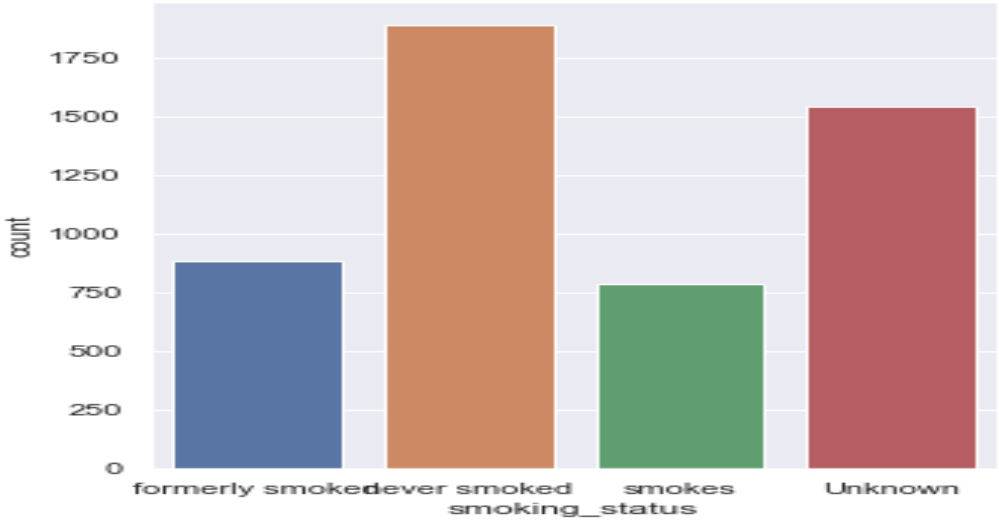
9. Pie chart:

- Here 1 indicates if the patient is having hypertension and 0 for not.



10. Countplot:

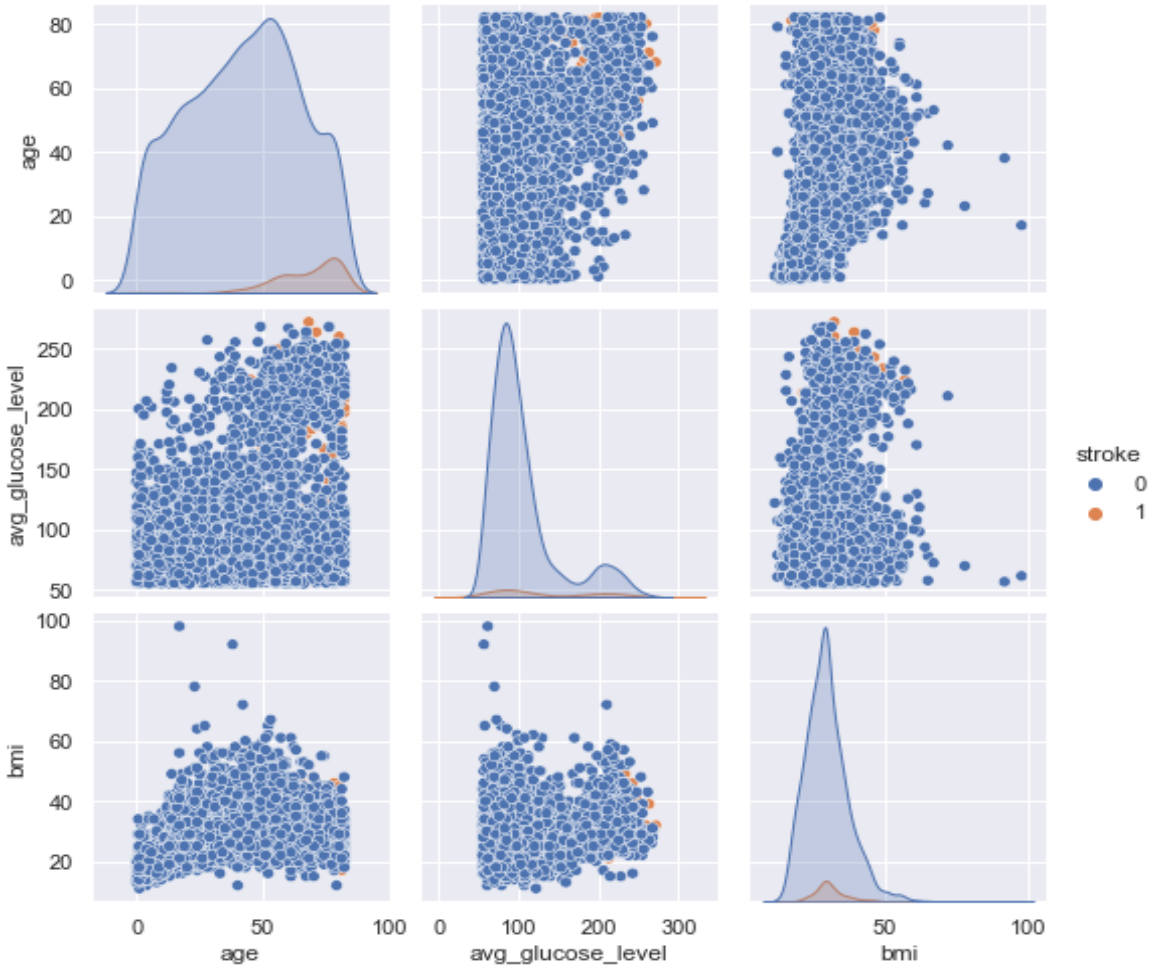
- In this countplot we can observe smoking status which is represented by different colors.
- Here, blue represents the formerly smoked, orange is for never smoked, green is for smokes and re represents the unknown.



11. Pairplot:

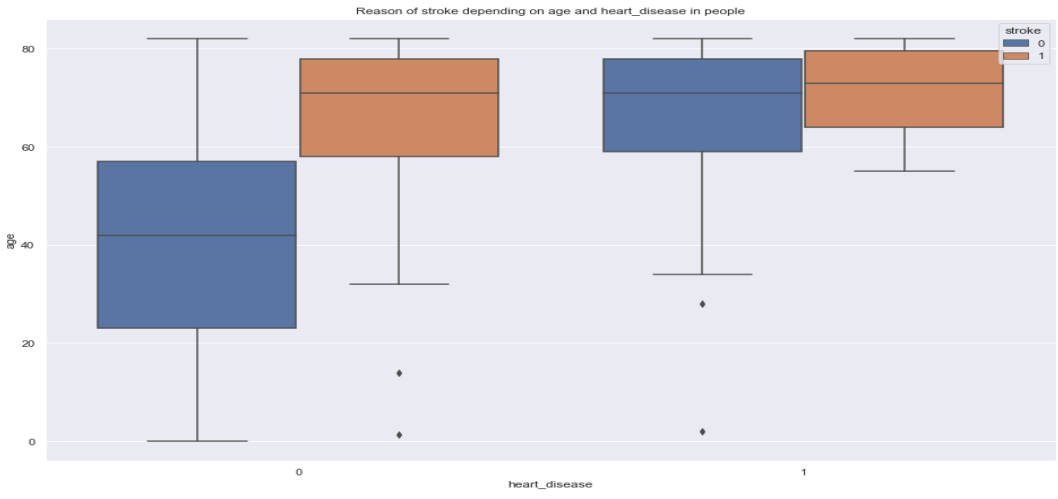
- In a dataset, a pairplot displays pairwise connections. Each data variable is shared over one row and one column on the y-axis and one column and row on the x-axis, respectively, in the grid of axes created by the pairplot function.

- Here we have used three fields to generate pairplot (i.e “age”, “avg-glucouse”, “bmi”). Where orange represent stroke occurred and 0 represent for not.



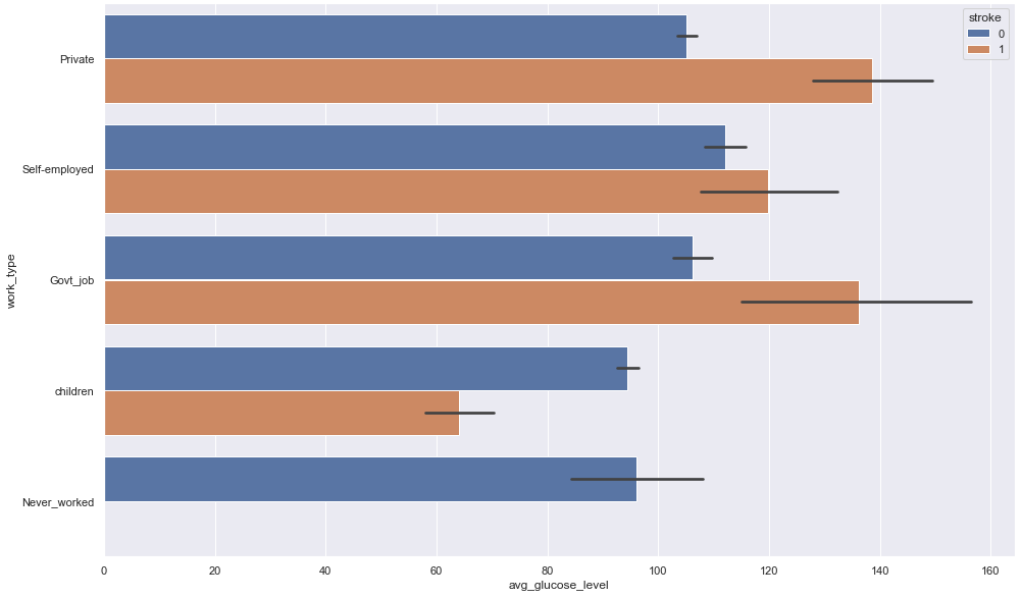
12. Boxplot:

- As observed in the boxplot, it is seen that stroke is observed in people mainly after the age of 60 and with some heart disease.

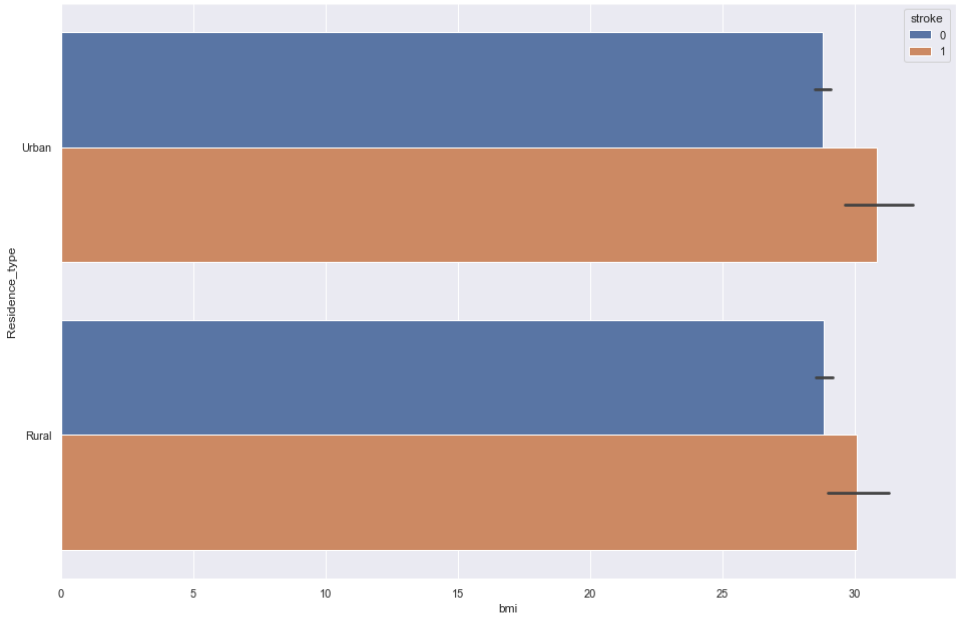


13. Barplot:

- As displayed in the horizontal barplot people working in private/ government job has highest rate of strokes observed on the other hand people who never worked had no strokes.



- Similarly, a barplot displaying relation between “bmi” and “Residence_type” w.r.t target attribute “stroke”.



Conclusion:

- The dataset obtained from an open source Kaggle is processed and is analyzed through various relations within the attributes and with its target distribute “stroke” in the form of dashboards using platform Power BI & libraries like matplotlib, pyplot and seaborn to retrieve meaningful information from our dataset. So that our model can predict if the patient with following attributes have stroke or not.