STROKE RISK PREDICTION

Team Members:

19BDS0029 (Tushar Singla)
19BDS0042 (Samarth Gupta)
19BDS0050 (Mrigank Sachan)

Report submitted for the
First Project Review of

Course Code: CSE3045
Predictive Analytics

Slot: A2

Professor: Dr. Ilanthenral Kandasamy

1. Introduction:

Stroke is a blood coagulation or drains in the cerebrum, which can make super durable harm that affects versatility, perception, sight or correspondence. Stroke is considered as clinical pressing circumstance and can cause long haul neurological harm, difficulties and frequently death. Stroke is the secondary driving reason for death and primary driving reason for grown-up inability worldwide with 400-800 strokes per 100,000, 15 million new acute strokes every year, 28,500,000 disability adjusted life-years and 28-30-day case fatality ranging from 17% to 35%. The burden of stroke will probably deteriorate with stroke and heart illness related deaths projected to increment to 5,000,000 out of 2020, contrasted with 3,000,000 of every 1998. This will be an aftereffect of proceeding with wellbeing and segment progress bringing about expansion in vascular infection risk factors and population of older people. Passing from stroke is because of comorbidities or potential difficulties. Stroke can occur in anyone regardless of race, gender or age however the chances of having a stroke increase if an individual has certain risk factors that can cause a stroke. The most effective way to safeguard oneself as well as other people is to comprehend individual gamble and how to oversee it. Complexities of stroke might emerge at various time-frames. Every 4 minutes someone passes away due to stroke, but up to 80% of it can be stopped if we can predict the presence of stroke in its early stage. The goal of this project is to work on the best machine learning algorithm to predict such strokes and prevent the number of deaths due to it.

2. Literature Review Summary Table

Authors and Year	*Title (Study)*	*Concept / Theoretical model/ Framework*	*Methodology used/ Implementation*	*Dataset details/ Analysis*	*Relevant Finding*	*Limitations/ Future Research/ Gaps identified*
Matthew Chun, Robert Clarke, Benjamin J. Cairns (May 2021)	Stroke risk prediction using machine learning	This research compares Cox models, machine learning (ML), and ensemble models.	Compared discrimination and calibration of Cox regression, logistic regression, supporting vector machines, random survival forests, gradient boosted trees (GBT), and multilayer perceptron, benchmarking performance against the 2017 Framingham Stroke Risk Profile	The dataset included 143 risk factor indicators in addition to incident stroke cases and a time-to-event for each stroke event.	Among the included study participants, the mean (SD) age was 51.9 (10.6) years and 59% were women. The incidence of stroke was higher in men than in women (9.5% vs 7.9%) and varied by more than 5-fold between the 10 study areas.	Atrial fibrillation (AF), which is commonly included in risk scores for CVD and stroke,10,13 was not recorded in the CKB and could not be included in the models.
Vamsi Bandi, Debnath Bhattacharyya, Divya M. (2020)	Prediction of Brain Stroke Severity Using Machine Learning	A Stroke Prediction (SPN) algorithm is proposed by using the improvised random forest in analysing the levels of risks obtained within the strokes.	Algorithm 1: Random Forest Algorithm 2: AdaBoost: Boosting is the technique of combining all the weak classifiers under a single strong classifier.	The data set used in this research work includes a total of 4,799 subjects which contains 3,123 males and 1,676 females.	The Random Forest classifier was found out to be the best performer with an accuracy of 94.23%, 92.16% sensitivity, 95.07 specificities, 0.04% low error rate results.	In this research work, the authors have done only a classification of various types of strokes. This leads to improper classification and we do not attain decent accuracy to predict the stroke severity. So, there is a gap identified for predicting the risk levels of stroke factors which are low, moderate, high and severe.

Sergio Penafi El, Nelson Baloia, Horacio Sanson, Jose Pino (Novemb er 2020)	Predicting Stroke Risk with an Interpretabl e Classifier	In this work, they presented the development of a prediction method which not only outperforms some other existing ones but also gives information about the most probable causes of high stroke risk. It is based on the Dempster-Shafer theory of plausibility.	For the case of disease history, the data is presented like a log showing the time when a patient was diagnosed with certain disease. This representation cannot be inputted directly into the vector; instead, we can consider to have a column for each disease and mark it as 1 if the patient had the disease in the past and 0 if not.	The dataset had 27876 patients' records, 22140 of them did not have strokes (79.4%), and 5736 had (20.6%).	The model performanc e was tested using a 5-fold experiment over the dataset. For each fold, the following indicators were calculated: accuracy, sensitivity, specificity, F1 macro, and the area under the ROC curve.	One of the limitations of all proposed representations is the lack of interpretability; the authors emphasize that this is an important aspect in the medical field but currently its coverage is very low among existing proposed solutions.
Xuemen g Li, Di Bian, Jinghui Y, Mei Li3 and Dongshe ng Zhao (2019)	Using machine learning models to improve stroke risk level classificatio n methods of China national stroke screening	In this paper, they have used 2017 national stroke screening data to develop stroke risk classification models based on machine learning algorithms to improve the classification efficiency.	Training set and test sets were separated and then they performed logistic regression model, Naïve Bayesian model, Bayesian network model, decision tree model, neural network model, random forest model, bagged decision tree model, voting model and boosting model with decision trees to classify stroke risk levels.	They used five different heart disease datasets, four breast cancer datasets, two diabetes datasets, two liver disease datasets and one hepatitis dataset.	Ten-fold cross validation method was used to evaluate results. Random Forest: Precision: 97.3% Recall: 98.44% F1-score: 97.88 AUC: 99.94%	Unknown factors which can cause stroke cannot be tracked right now with the current technology and hence the model is not that reliable right now. This paper does not analyse the classification level of stroke levels.

Nugroho Sinung Adi, Richas Farhany, Rafidah Ghina, Herlina Napitupulu (2021)	Stroke Risk Prediction Model Using Machine Learning	The research uses three machine learning algorithm models, including Naive Bayes, Decision Tree, and Random Forest. The prediction uses patient health history as the attribute in each mode.	The procedure performed in testing the algorithms consists of six steps. Including (1) collecting data, (2) data labeling, (3) data preparation, (4) data partitioning, (5) data classification, and (6) evaluation. Knime Analytics Platform was used.	The stroke dataset was used to classify stroke predictions and it was taken from Kaggle in September 2021. It consists of attributes including gender, age, hypertension, etc.	The highest accuracy value was obtained by Random Forest with a result of 94.26%.	This model does not consider the factors such as recent strenuous activity and occupation.

3. Objective of the project: Stroke is the subsequent driving reason for death worldwide and stays a significant wellbeing trouble both for the people and for the public medical care frameworks. Possibly modifiable risk factors for stroke incorporate hypertension, heart infection, diabetes, and dysregulation of glucose digestion, atrial fibrillation, and way of lifestyle factors. Thus, the objective of our project is to apply standards of Machine Learning and AI over enormous dataset collections to actually foresee the stroke in light of possibly modifiable risk factors. The next objective is to provide the user with a web interface to check whether there is a risk of stroke or not by entering the required input values.

4. Innovation component in the project: This project involves the working of 5 different machine learning algorithms to predict the risk of stroke. The usage of 5 different ML algorithms and then deploying the web application based on it is innovative as it will help to predict the risk of stroke very accurately. After finding the best algorithm, an application is created by deploying it on the web. On entering the input values such as gender, age, BMI, etc. the web application works on the best performance algorithm and calculates whether there is a risk of stroke or not. The algorithm diagnoses the risk of stroke based on the given inputs.

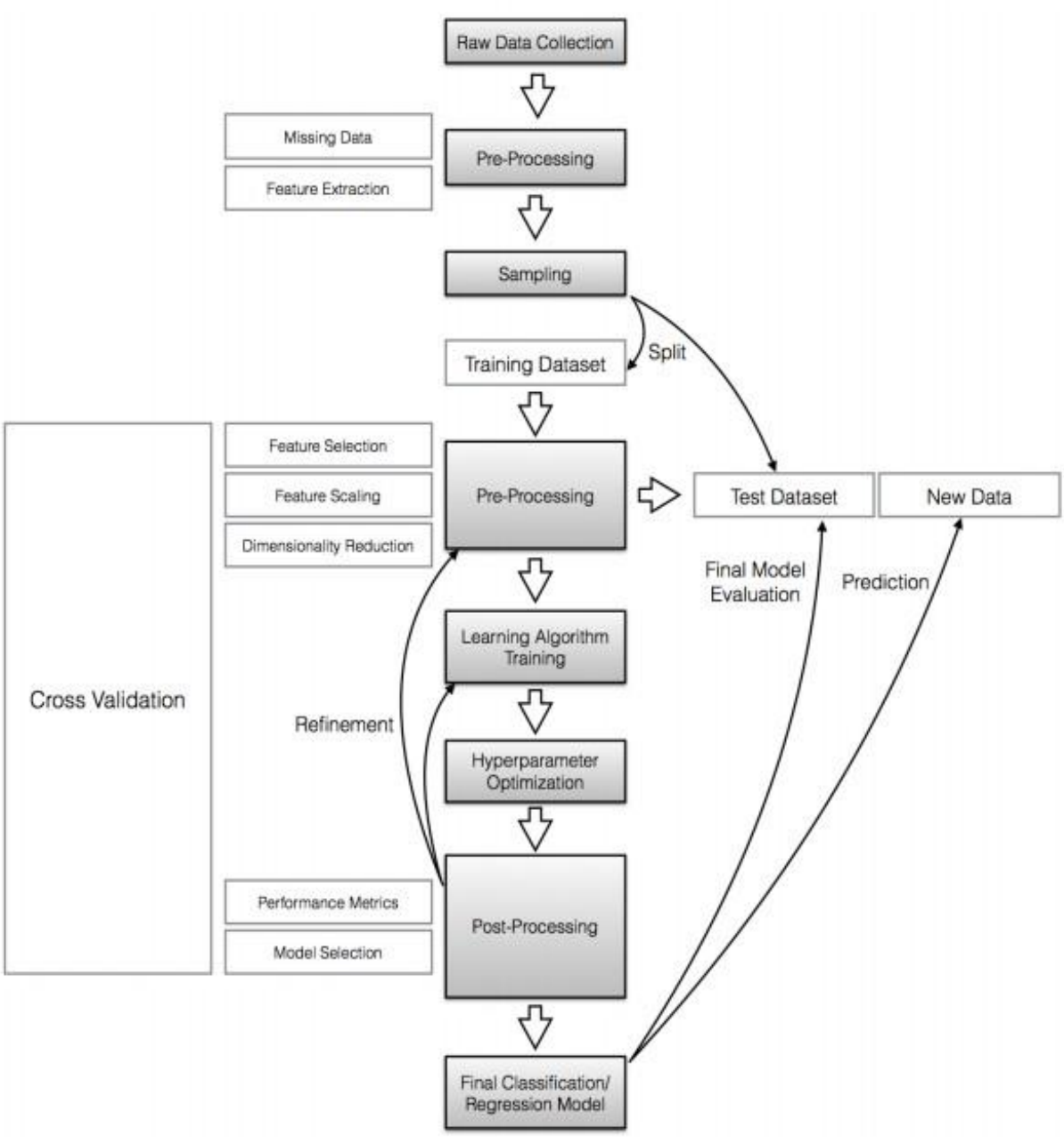
5. Work done and implementation

a. Methodology adapted: Our project uses machine learning algorithms such as Decision Tree, Logistic Regression, KNN, Random Forest, Support Vector Machine. The process starts with collection of data and pre-processing it. The dataset is collected from Kaggle and it includes 12 attributes. Exploratory Data Analysis is performed for the important variables and data is visualized. The pre-processing includes removal of

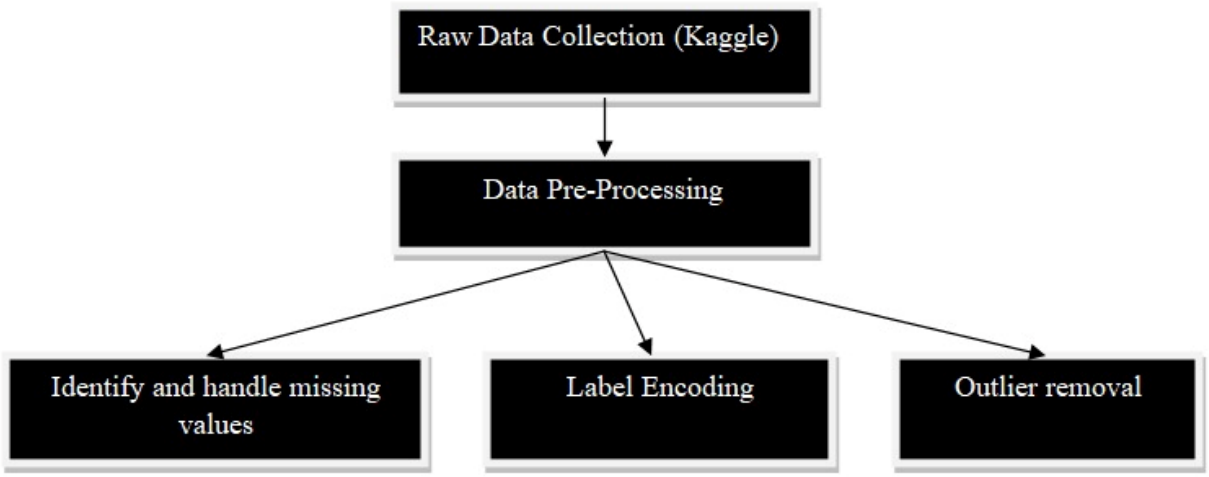
missing data, label encoding and feature extraction. Outliers are also removed during this process. The dataset is then divided into training and testing data. We have considered 80 % - 20 % ratio. After this, the algorithm is applied to the data and the performance measures are compared.

The hardware requirements for this project includes a Windows/Linux PC. The software requirements include Python, Jupyter Notebook, Pandas, Seaborn and other libraries.

By the end of the project the expected architecture is as follows:



The architecture based on our current progress is:



b. Dataset used:

a. The dataset used is Stroke-Prediction dataset. It has been obtained from Kaggle. This dataset is used to predict whether a patient is likely to get stroke based on the input parameters like gender, age, various diseases, and smoking status. Each row in the data provides relevant information about the patient. The data contains 5110 observations with 12 attributes. He attributes include – id, gender, age, hypertension, heart disease, ever married, work type, Residence type, avg glucose level, bmi, smoking status, stroke.

b. The project is not based on any paper but the topic is inspired from a research paper from CMRIT, Bangalore where the authors have worked on "Prediction of Stroke Using Machine Learning". They have worked on the algorithms like Naïve Bayes, Artificial Neural Network to predict the risk. We have used other ML algorithms like SVM, Logistic Regression, KNN, Random Forest to compute the accuracy and then deploy a web application to predict the stroke risk by computing the input values.

c. This project differs from the reference project by working on other ML algorithms like Decision Tree, Logistic Regression, KNN, Random Forest, Support Vector Machine. The aim is to compare these 5 algorithms and find out the best performance metrics to successfully predict the stroke risk. A web application is also developed to check the risk by entering the input values like age, gender, BMI, etc.

c. Tools to be used:

The tools required are Python, Jupyter Notebook and libraries like Matplotlib, Seaborn, Pandas. RStudio is used for data visualization.

Python language is used to write the code for Machine Learning algorithms. Jupyter notebook is used as a platform to write the Python code.

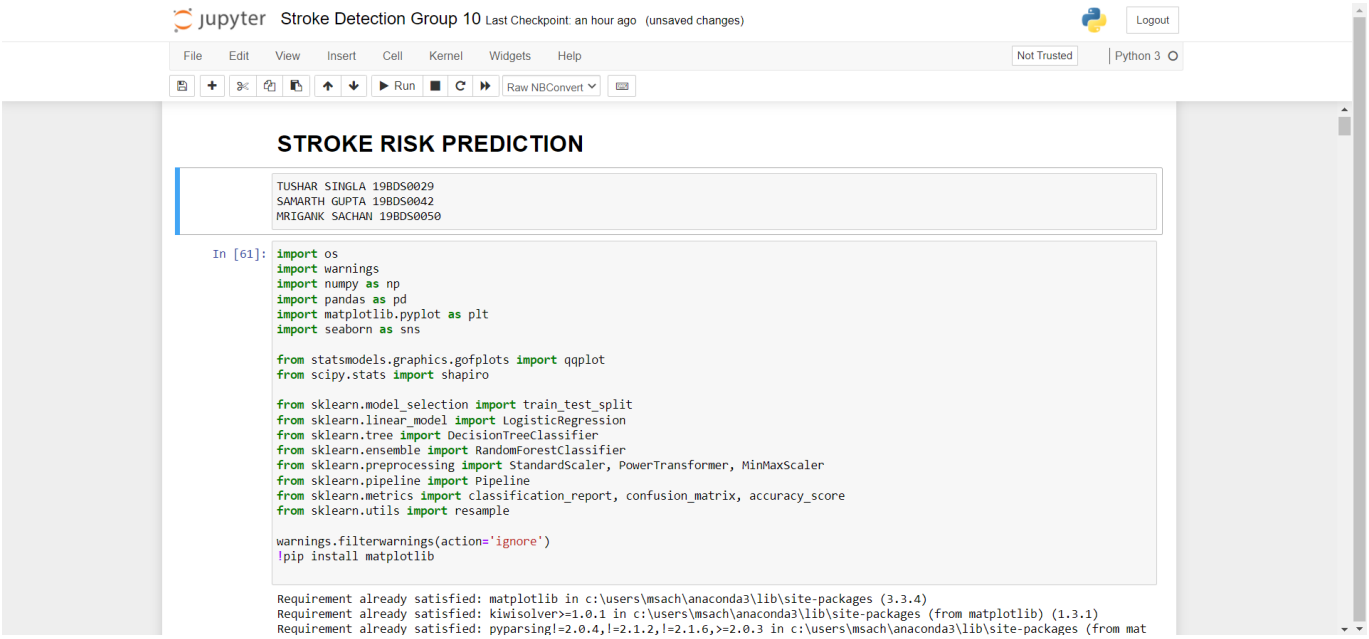
Matplotlib is a plotting library for the Python programming language. It provides an object-oriented API for embedding plots into applications.

Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.

Pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series.

The RStudio IDE is a set of integrated tools designed to be more productive with R and Python. R is a free software environment for statistical computing and graphics. It compiles and runs on a wide variety of UNIX platforms, Windows and MacOS.

d. Screenshot and Demo along with Visualization:



jupyter Stroke Detection Group 10 Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run | Raw NBConvert

```
Requirement already satisfied: matplotlib in c:\users\msach\anaconda3\lib\site-packages (3.3.4)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\msach\anaconda3\lib\site-packages (from matplotlib) (1.3.1)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.3 in c:\users\msach\anaconda3\lib\site-packages (from mat
plotlib) (2.4.7)
Requirement already satisfied: numpy>=1.15 in c:\users\msach\anaconda3\lib\site-packages (from matplotlib) (1.20.1)
Requirement already satisfied: pillow>=6.2.0 in c:\users\msach\anaconda3\lib\site-packages (from matplotlib) (8.2.0)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\msach\anaconda3\lib\site-packages (from matplotlib) (2.8.1)
Requirement already satisfied: cycler>=0.10 in c:\users\msach\anaconda3\lib\site-packages (from matplotlib) (0.10.0)
Requirement already satisfied: six in c:\users\msach\anaconda3\lib\site-packages (from cycler>=0.10->matplotlib) (1.15.0)
```

In [29]:
```
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
plt.rcParams['figure.figsize'] = (5, 5)
```

In [32]:
```
data=pd.read_csv(r'\Users\msach\Desktop\Stroke-Risk-Prediction-using-Machine-Learning-master\Stroke-Risk-Prediction-using-Machine
```

jupyter Stroke Detection Group 10 Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run | Raw NBConvert

DATASET PREVIEW!

In [35]:
```
data
```

Out[35]:

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke
0	9046	Male	67.0	0	1	Yes	Private	Urban	228.69	36.6	formerly smoked	1
1	51676	Female	61.0	0	0	Yes	Self-employed	Rural	202.21	NaN	never smoked	1
2	31112	Male	80.0	0	1	Yes	Private	Rural	105.92	32.5	never smoked	1
3	60182	Female	49.0	0	0	Yes	Private	Urban	171.23	34.4	smokes	1
4	1665	Female	79.0	1	0	Yes	Self-employed	Rural	174.12	24.0	never smoked	1
...	...	...	...	...	...	...	...	...	...	...	...	...
5105	18234	Female	80.0	1	0	Yes	Private	Urban	83.75	NaN	never smoked	0
5106	44873	Female	81.0	0	0	Yes	Self-employed	Urban	125.20	40.0	never smoked	0
5107	19723	Female	35.0	0	0	Yes	Self-employed	Rural	82.99	30.6	never smoked	0
5108	37544	Male	51.0	0	0	Yes	Private	Rural	166.29	25.6	formerly smoked	0
5109	44679	Female	44.0	0	0	Yes	Govt_job	Urban	85.28	26.2	Unknown	0

5110 rows × 12 columns

In [ ]:
```
```

jupyter Stroke Detection Group 10 Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run | Raw NBConvert

Exploratory data analysis

In [41]:
```
data.shape
```

Out[41]: (5110, 12)

In [50]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5110 entries, 0 to 5109
Data columns (total 12 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   id                 5110 non-null   int64
 1   gender             5110 non-null   object
 2   age                5110 non-null   float64
 3   hypertension       5110 non-null   int64
 4   heart_disease      5110 non-null   int64
 5   ever_married       5110 non-null   object
 6   work_type          5110 non-null   object
 7   Residence_type     5110 non-null   object
 8   avg_glucose_level  5110 non-null   float64
 9   bmi                4909 non-null   float64
 10  smoking_status     5110 non-null   object
 11  stroke             5110 non-null   int64
dtypes: float64(3), int64(4), object(5)
memory usage: 479.2+ KB
```

jupyter Stroke Detection Group 10 Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run Raw NBConvert

CHECK IF ANY DATA IS NULL

In [57]:
```
data.isnull().sum()
```

Out[57]:
```
id                   0
gender               0
age                  0
hypertension         0
heart_disease        0
ever_married         0
work_type            0
Residence_type       0
avg_glucose_level    0
bmi                201
smoking_status       0
stroke               0
dtype: int64
```

Lets fill Null Values

In [77]:
```
def heatmapper(data, method, target_col, annot=False, normality=False):
    # create a correlation matrix
    correlation_df = data.corr(method=method)
    correlation_mask = np.triu(np.ones_like(correlation_df))

    # create the data for the bar plot
    bar_df = correlation_df[target_col]
    bar_df = bar_df[bar_df.index != target_col].sort_values(ascending=True).reset_index()

    # plot the heat map
    if normality:
        fig, (ax1, ax2, ax3) = plt.subplots(1, 3, figsize=(18, 6))
```

jupyter Stroke Detection Group 10 Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run Raw NBConvert

Lets fill Null Values

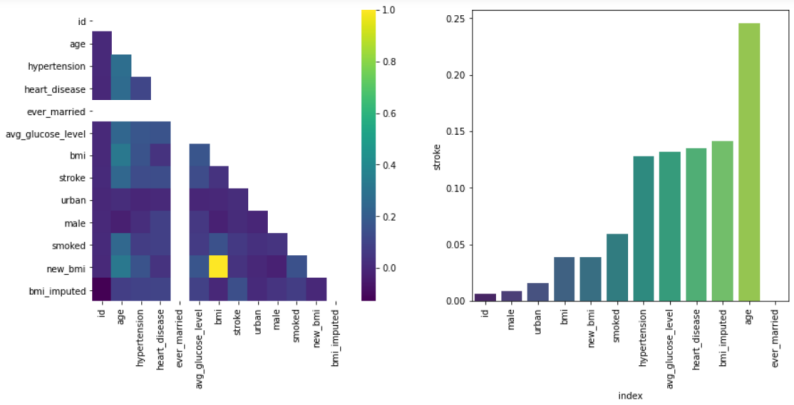
In [77]:
```
def heatmapper(data, method, target_col, annot=False, normality=False):
    # create a correlation matrix
    correlation_df = data.corr(method=method)
    correlation_mask = np.triu(np.ones_like(correlation_df))

    # create the data for the bar plot
    bar_df = correlation_df[target_col]
    bar_df = bar_df[bar_df.index != target_col].sort_values(ascending=True).reset_index()

    # plot the heat map
    if normality:
        fig, (ax1, ax2, ax3) = plt.subplots(1, 3, figsize=(18, 6))
    else:
        fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(14, 6))
    sns.heatmap(data=correlation_df, mask=correlation_mask, annot=annot, cmap='viridis', ax=ax1)

    # plot the bar graph
    sns.barplot(x='index', y='stroke', data=bar_df, palette='viridis', ax=ax2)
    ax2.set_xticklabels(bar_df['index'], rotation=90)

    if normality:
    # check if the feature is guassian
        probability=0.05

        _, p = shapiro(data[target_col])
        qqplot(data[target_col], line='s', ax=ax3)
        if p > probability:
            ax3.set_title(f'P value of {p} > {probability} -> Gaussian')
        else:
            ax3.set_title(f'P value of {p} <= {probability} -> Not gaussian')
    else:
        pass
```

```
    # create the data for the bar plot
    bar_df = correlation_df[target_col]
    bar_df = bar_df[bar_df.index != target_col].sort_values(ascending=True).reset_index()

    # plot the heat map
    if normality:
        fig, (ax1, ax2, ax3) = plt.subplots(1, 3, figsize=(18, 6))
    else:
        fig, (ax1, ax2) = plt.subplots(1, 2, figsize=(14, 6))
    sns.heatmap(data=correlation_df, mask=correlation_mask, annot=annot, cmap='viridis', ax=ax1)

    # plot the bar graph
    sns.barplot(x='index', y='stroke', data=bar_df, palette='viridis', ax=ax2)
    ax2.set_xticklabels(bar_df['index'], rotation=90)

    if normality:
    # check if the feature is guassian
        probability=0.05

        _, p = shapiro(data[target_col])
        qqplot(data[target_col], line='s', ax=ax3)
        if p > probability:
            ax3.set_title(f'P value of {p} > {probability} -> Gaussian')
        else:
            ax3.set_title(f'P value of {p} <= {probability} -> Not gaussian')
    else:
        pass

    return plt.show()


heatmapper(data, 'pearson', 'stroke')
```

jupyter Stroke Detection Group 10 Last Checkpoint: an hour ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

Run Raw NBConvert



jupyter Stroke Detection Group 10 Last Checkpoint: an hour ago (autosaved) Logout

File Edit View Insert Cell Kernel Widgets Help Not Trusted Python 3

Run Raw NBConvert

In [93]:
```
def convert_binary_features(data):
    # convert the binary fields into 1/0
    data['ever_married'] = np.where(data['ever_married'] == 'Yes', 1, 0)
    data['urban'] = np.where(data['Residence_type'] == 'Urban', 1, 0)
    data['male'] = np.where(data['gender'] == 'Male', 1, 0)

    # Convert if ever smoked into a binary field
    smoking_list = ['formerly smoked', 'smokes']
    data['smoked'] = np.where(data['smoking_status'].isin(smoking_list), 1, 0)

    # BMI is the only null column, for now try imputing the mean value
    median_bmi = data['bmi'].median()
    mean_bmi = data['bmi'].mean()
    data['new_bmi'] = np.where(data['bmi'].isnull(), mean_bmi, data['bmi'])
    data['bmi_imputed'] = np.where(data['bmi'].isnull(), 1, 0)

    # get dummies for work features
    work_dummies = pd.get_dummies(data['work_type'], drop_first=True, prefix='work')
    data = pd.merge(left=data,
                    right=work_dummies,
                    how='left',
                    left_index=True,
                    right_index=True)


    # standardise glucose and BMI
    data['glucose'] = (data['avg_glucose_level'] - data['avg_glucose_level'].mean()) / data['avg_glucose_level'].std()
    data['body_mass_index'] = (data['new_bmi'] - data['new_bmi'].mean()) / data['new_bmi'].std()
    data['age'] = np.log(data['age'])

    # if you have a glucose level over 200 you are v likley to be diabetic - increasing your rish
    data['diabetes'] = np.where(data['avg_glucose_level'] > 200, 1, 0)

    # remove old columns from dataset
```

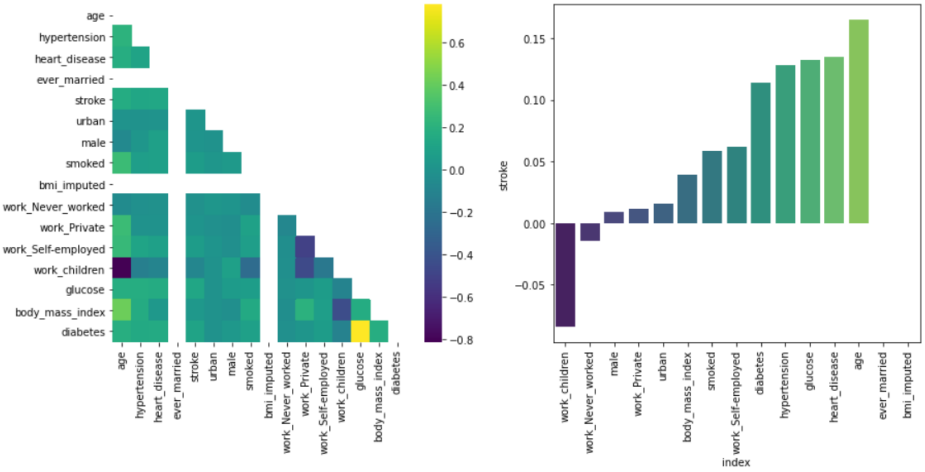
```
    # if you have a glucose level over 200 you are v likley to be diabetic - increasing your rish
    data['diabetes'] = np.where(data['avg_glucose_level'] > 200, 1, 0)

    # remove old columns from dataset
    drop_cols = ['Residence_type', 'gender', 'bmi', 'work_type', 'smoking_status',
                 'new_bmi', 'avg_glucose_level', 'id']
    data = data.drop(drop_cols, axis=1)
    return data


df = convert_binary_features(data)
data.head()
```

Out[93]:

	id	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke	urban	n
0	9046	Male	67.0	0	1	0	Private	Urban	228.69	36.600000	formerly smoked	1	1	
1	51676	Female	61.0	0	0	0	Self-employed	Rural	202.21	28.893237	never smoked	1	0	
2	31112	Male	80.0	0	1	0	Private	Rural	105.92	32.500000	never smoked	1	0	
3	60182	Female	49.0	0	0	0	Private	Urban	171.23	34.400000	smokes	1	1	
4	1665	Female	79.0	1	0	0	Self-employed	Rural	174.12	24.000000	never smoked	1	0	

jupyter Stroke Detection Group 10 Last Checkpoint: an hour ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted | Python 3
Run Code

In [146]:
```
heatmapper(df, 'pearson', 'stroke', annot=False)
```



In [147]:
```
data['bmi'].value_counts()
```

Out[147]:
```
28.893237    201
28.700000     41
28.400000     38
26.700000     37
26.100000     37
             ...
48.000000      1
49.400000      1
47.400000      1
46.600000      1
54.000000      1
Name: bmi, Length: 419, dtype: int64
```

In [148]:
```
data['bmi'].describe()
```

Out[148]:
```
count    5110.000000
mean       28.893237
std         7.698018
min        10.300000
25%        23.800000
50%        28.400000
75%        32.800000
max        97.600000
Name: bmi, dtype: float64
```

In [149]:
```
data['bmi'].fillna(data['bmi'].mean(),inplace=True)
```

jupyter Stroke Detection Group 10 Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run Code

In [150]:
```
data['bmi'].describe()
```

Out[150]:
```
count    5110.000000
mean       28.893237
std         7.698018
min        10.300000
25%        23.800000
50%        28.400000
75%        32.800000
max        97.600000
Name: bmi, dtype: float64
```

In [151]:
```
data.isnull().sum()
```

Out[151]:
```
gender               0
age                  0
hypertension         0
heart_disease        0
ever_married         0
work_type            0
Residence_type       0
avg_glucose_level    0
bmi                  0
smoking_status       0
stroke               0
urban                0
male                 0
smoked               0
new_bmi              0
bmi_imputed          0
dtype: int64
```

jupyter Stroke Detection Group 10 Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run Code

In [162]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5110 entries, 0 to 5109
Data columns (total 16 columns):
 #   Column             Non-Null Count  Dtype  
---  ------             --------------  -----  
 0   gender             5110 non-null   object 
 1   age                5110 non-null   float64
 2   hypertension       5110 non-null   int64  
 3   heart_disease      5110 non-null   int64  
 4   ever_married       5110 non-null   int32  
 5   work_type          5110 non-null   object 
 6   Residence_type     5110 non-null   object 
 7   avg_glucose_level  5110 non-null   float64
 8   bmi                5110 non-null   float64
 9   smoking_status     5110 non-null   object 
 10  stroke             5110 non-null   int64  
 11  urban              5110 non-null   int32  
 12  male               5110 non-null   int32  
 13  smoked             5110 non-null   int32  
 14  new_bmi            5110 non-null   float64
 15  bmi_imputed        5110 non-null   int32  
dtypes: float64(4), int32(5), int64(3), object(4)
memory usage: 539.1+ KB
```

In [167]:
```
data
```

jupyter Stroke Detection Group 10 Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run Code

In [167]:
```
data
```

Out[167]:

	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke	urban	male
0	Male	67.0	0	1	0	Private	Urban	228.69	36.600000	formerly smoked	1	1	1
1	Female	61.0	0	0	0	Self-employed	Rural	202.21	28.893237	never smoked	1	0	0
2	Male	80.0	0	1	0	Private	Rural	105.92	32.500000	never smoked	1	0	1
3	Female	49.0	0	0	0	Private	Urban	171.23	34.400000	smokes	1	1	0
4	Female	79.0	1	0	0	Self-employed	Rural	174.12	24.000000	never smoked	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
5105	Female	80.0	1	0	0	Private	Urban	83.75	28.893237	never smoked	0	1	0
5106	Female	81.0	0	0	0	Self-employed	Urban	125.20	40.000000	never smoked	0	1	0
5107	Female	35.0	0	0	0	Self-employed	Rural	82.99	30.600000	never smoked	0	0	0
5108	Male	51.0	0	0	0	Private	Rural	166.29	25.600000	formerly smoked	0	0	1
5109	Female	44.0	0	0	0	Govt_job	Urban	85.28	26.200000	Unknown	0	1	0

5110 rows × 16 columns

jupyter Stroke Detection Group 10 Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run Code

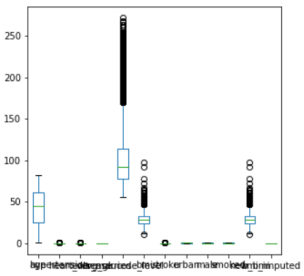
Outlier Removation

In [174]:
```
from matplotlib.pyplot import figure
figure(num=None, figsize=(8, 6), dpi=800, facecolor='w', edgecolor='k')
```

Out[174]:
```
<Figure size 6400x4800 with 0 Axes>
```

<Figure size 6400x4800 with 0 Axes>

In [178]:
```
data.plot(kind='box')
plt.show()
```



jupyter Stroke Detection Group 10 Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run Code

Label Encoding

In [181]:
```
data.head()
```

Out[181]:

	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke	urban	male	sn
0	Male	67.0	0	1	0	Private	Urban	228.69	36.600000	formerly smoked	1	1	1	
1	Female	61.0	0	0	0	Self-employed	Rural	202.21	28.893237	never smoked	1	0	0	
2	Male	80.0	0	1	0	Private	Rural	105.92	32.500000	never smoked	1	0	1	
3	Female	49.0	0	0	0	Private	Urban	171.23	34.400000	smokes	1	1	0	
4	Female	79.0	1	0	0	Self-employed	Rural	174.12	24.000000	never smoked	1	0	0	

In [185]:
```
from sklearn.preprocessing import LabelEncoder
enc=LabelEncoder()
```

In [192]:
```
gender=enc.fit_transform(data['gender'])
```

In [198]:
```
smoking_status=enc.fit_transform(data['smoking_status'])
```

In [204]:
```
work_type=enc.fit_transform(data['work_type'])
Residence_type=enc.fit_transform(data['Residence_type'])
ever_married=enc.fit_transform(data['ever_married'])
```

In [207]:
```
data['work_type']=work_type
```

jupyter Stroke Detection Group 10 Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

Run Code

In [212]:
```
data['ever_married']=ever_married
data['Residence_type']=Residence_type
data['smoking_status']=smoking_status
data['gender']=gender
```

In [ ]:

In [218]:
```
data
```

Out[218]:

	gender	age	hypertension	heart_disease	ever_married	work_type	Residence_type	avg_glucose_level	bmi	smoking_status	stroke	urban	male
0	1	67.0	0	1	0	2	1	228.69	36.600000	1	1	1	1
1	0	61.0	0	0	0	3	0	202.21	28.893237	2	1	0	0
2	1	80.0	0	1	0	2	0	105.92	32.500000	2	1	0	1
3	0	49.0	0	0	0	2	1	171.23	34.400000	3	1	1	0
4	0	79.0	1	0	0	3	0	174.12	24.000000	2	1	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
5105	0	80.0	1	0	0	2	1	83.75	28.893237	2	0	1	0
5106	0	81.0	0	0	0	3	1	125.20	40.000000	2	0	1	0
5107	0	35.0	0	0	0	3	0	82.99	30.600000	2	0	0	0
5108	1	51.0	0	0	0	2	0	166.29	25.600000	1	0	0	1
5109	0	44.0	0	0	0	0	1	85.28	26.200000	0	0	1	0

5110 rows × 16 columns

jupyter Stroke Detection Group 10 Last Checkpoint: an hour ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3

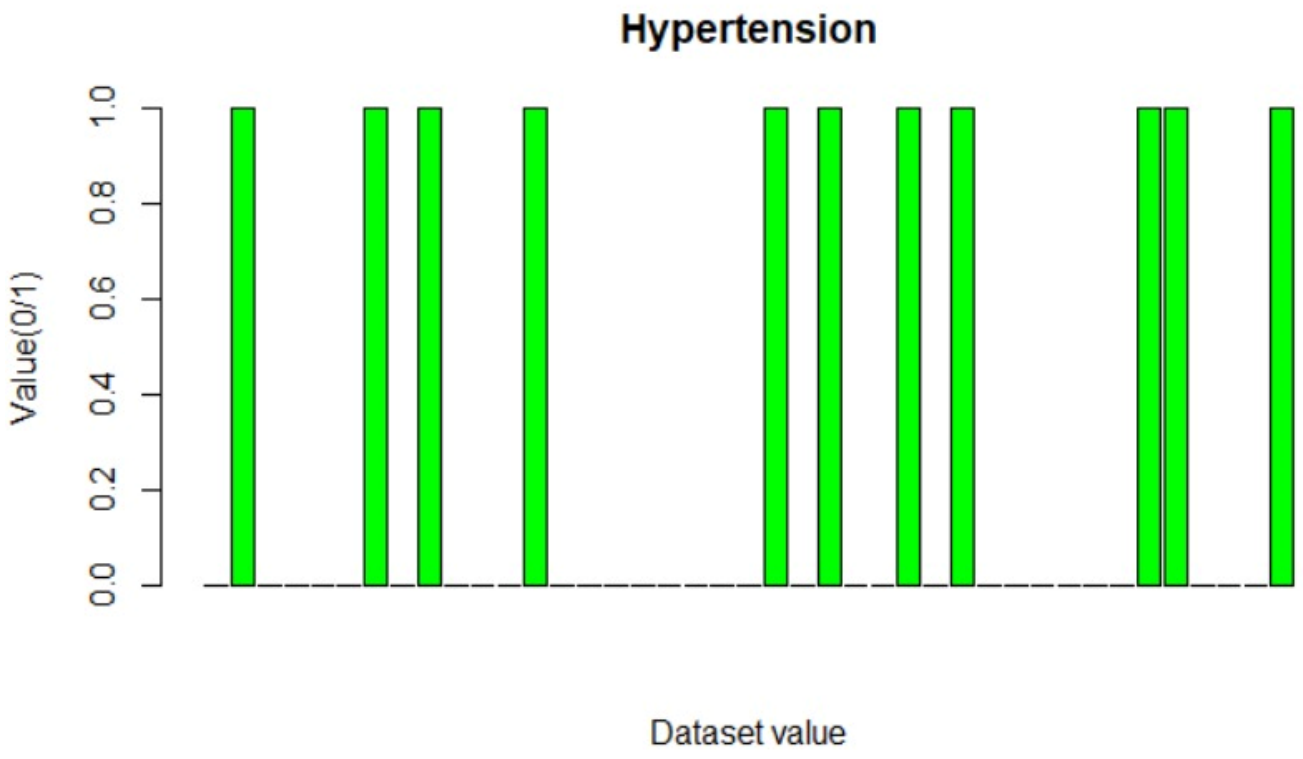
Run Code

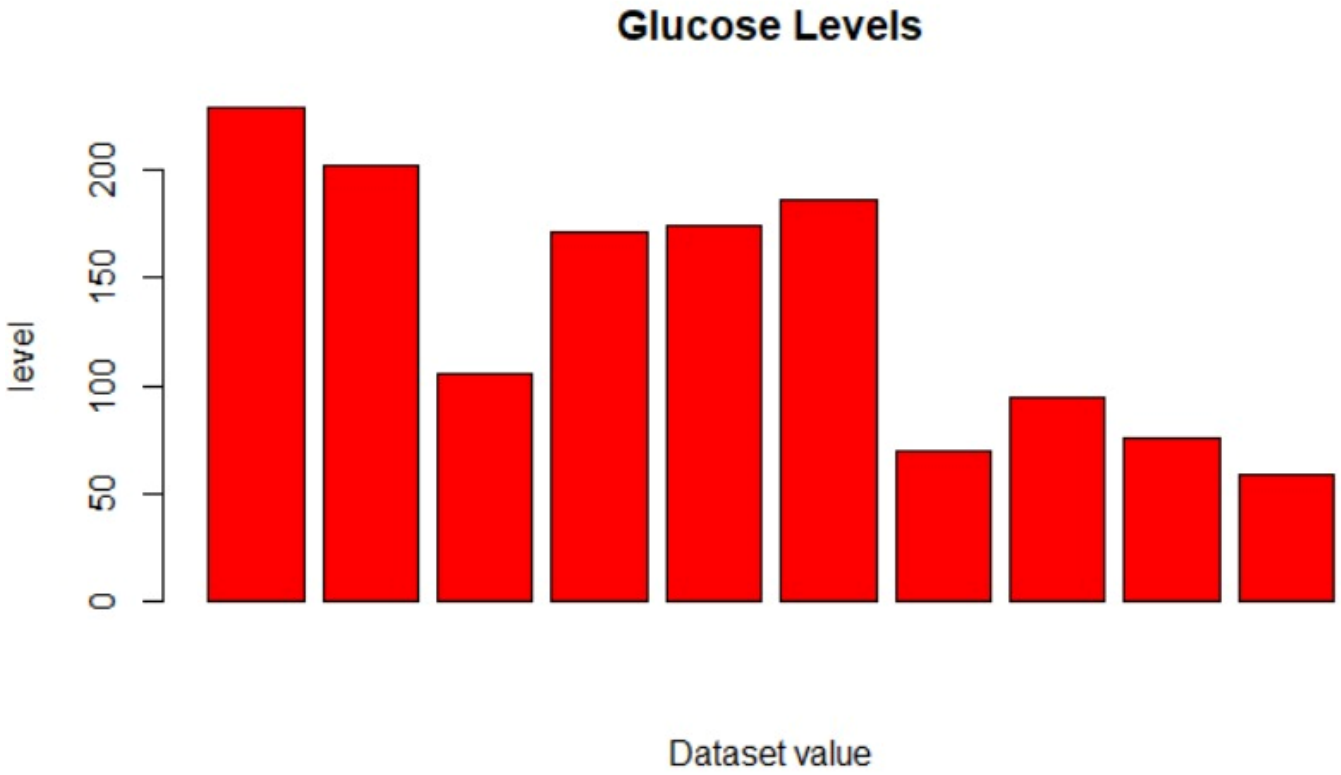
In [226]:
```
data.info()
```

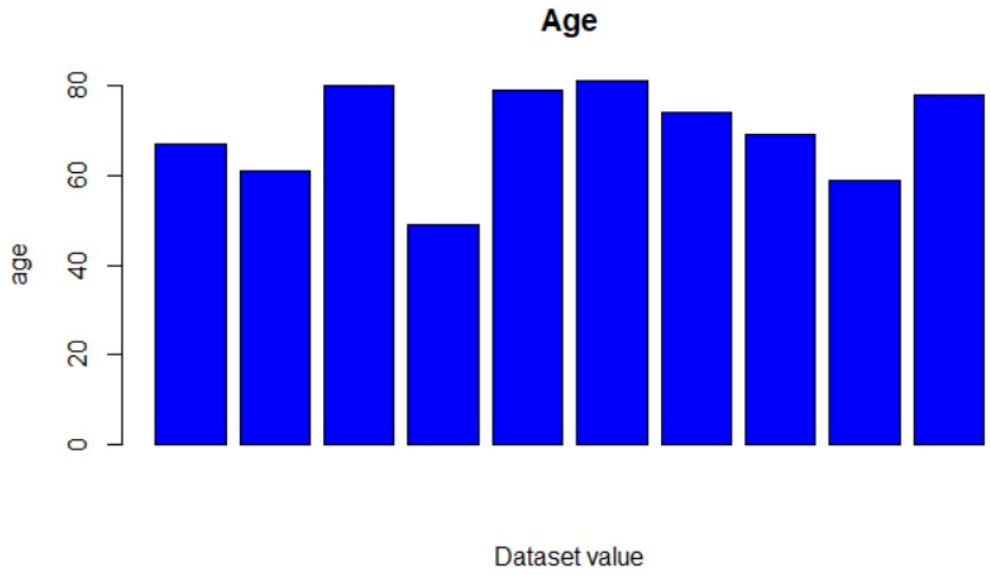
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5110 entries, 0 to 5109
Data columns (total 16 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   gender             5110 non-null   int32
 1   age                5110 non-null   float64
 2   hypertension       5110 non-null   int64
 3   heart_disease      5110 non-null   int64
 4   ever_married       5110 non-null   int64
 5   work_type          5110 non-null   int32
 6   Residence_type     5110 non-null   int32
 7   avg_glucose_level  5110 non-null   float64
 8   bmi                5110 non-null   float64
 9   smoking_status     5110 non-null   int32
 10  stroke             5110 non-null   int64
 11  urban              5110 non-null   int32
 12  male               5110 non-null   int32
 13  smoked             5110 non-null   int32
 14  new_bmi            5110 non-null   float64
 15  bmi_imputed        5110 non-null   int32
dtypes: float64(4), int32(8), int64(4)
memory usage: 479.2 KB
```

In [ ]:

In [ ]:


Hypertension
Value(0/1)
0.0
0.2
0.4
0.6
0.8
1.0
Dataset value


Glucose Levels
level
0
50
100
150
200
Dataset value



6. Expected Results

By the end of the project, we would be selecting the best algorithm with highest accuracy to predict the risk of stroke prediction based on the stroke prediction dataset. We also aim to develop a web application to instantly prompt the user if he or she has a probability of stroke based on the input values like age, gender, smoking status, BMI, hypertension and other similar parameters.

7. References - IEEE std.

Adi, N. S., Farhany, R., Ghina, R., & Napitupulu, H. (2022). *Stroke Risk Prediction Model Using Machine Learning*. 56–60. https://doi.org/10.1109/ICAIBDA53487.2021.9689731

Bandi, V., Bhattacharyya, D., & Midhunchakkravarthy, D. (2020). Prediction of brain stroke severity using machine learning. *Revue d'Intelligence Artificielle*, *34*(6), 753–761. https://doi.org/10.18280/RIA.340609

Chun, M., Clarke, R., Cairns, B. J., Clifton, D., Bennett, D., Chen, Y., Guo, Y., Pei, P., Lv, J., Yu, C., Yang, L., Li, L., Chen, Z., & Zhu, T. (2021). Stroke risk prediction using machine learning: a prospective cohort study of 0.5 million Chinese adults. *Journal of the American Medical Informatics Association : JAMIA*, *28*(8), 1719–1727. https://doi.org/10.1093/JAMIA/OCAB068

Li, X., Bian, D., Yu, J., Li, M., & Zhao, D. (2019). Using machine learning models to improve stroke risk level classification methods of China national stroke screening. *BMC Medical Informatics and Decision Making*, *19*(1), 1–7. https://doi.org/10.1186/S12911-019-0998-2/TABLES/5

Penafiel, S., Baloian, N., Sanson, H., & Pino, J. A. (2021). Predicting Stroke Risk with an Interpretable Classifier. *IEEE Access*, *9*, 1154–1166. https://doi.org/10.1109/ACCESS.2020.3047195