**Stroke Prediction Using Machine Learning Techniques**

**BootCamp on Data Science**

**and Tools**

Submitted

To

**CRC-Training**

**By**

**AYUSH GUPTA**

**Roll Number: 2002900100042**

**Under the guidance of**

**Mr. Gaurav Kansal / Mr. Gopal Gupta**

**TABLE OF CONTENTS**

Page No.

DECLARATION ii

CERTIFICATE iii

ACKNOWLEDGEMENTS iv

ABSTRACT v

LIST OF TABLES viii

LIST OF FIGURES ix

**CHAPTER-1 INTRODUCTION 1**

1.1 Problem Definition 1

1.2 Motivation 1

1.3 Objective of the Project 1

1.4 Scope of the Project 1

1.5 Need of Work 1

**CHAPTER 2 RELATED WORK 2**

**CHAPTER 3 PROPOSED METHODOLOGY 3**

3.1 Dataset Description 3

3.2 Methods 4

3.3 Hardware / Software Requirements 10

3.3 Our Methodology 10

**CHAPTER 4 EXPERIMENT AND RESULT ANALYSIS 29**

**CHAPTER 5 CONCLUSION 34**

5.1 Discussion 34

5.2 Future Work 34

**REFERENCES 35**

**CERTIFICATE**

This is to certify that Project Report entitled **“Brain Stroke Prediction Using Machine Learning Techniques”** which is submitted by **Ayush Gupta** in partial fulfilment of the requirement for the “Bootcamp on Data Science and Tools” in Department of CRC-Training of ABES Institute of Technology, is a record of the candidate own work carried out by him under my/our supervision.

**Mr. Gaurav Kansal**

**Mr. Gopal Gupta**

**Date:**

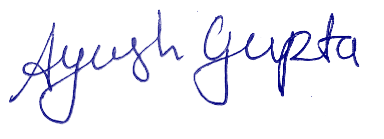
**ACKNOWLEDGEMENT**

*I am delighted to present this report on the “Bootcamp on Data Science and Tools” that I completed during my third year of B.Tech. I would like to express my sincere gratitude to Mr. Gopal Gupta and Mr. Gaurav Kansal for their constant support and guidance throughout the course of my work. Their constant motivation has been a source of inspiration for me. It is only due to their efforts that my endeavours have seen the light of day.*

*I would also like to acknowledge the contribution of the team members of CRC-Training for their full support and assistance during the development of the project.*

*I would also like to acknowledge the motivation of the Department of Computer Science and Engineering, ABES Institute of Technology, for providing me the opportunity to undergo training at CRC-Training.*

*Signature:*

**

*Name : Ayush Gupta*

*Roll No.:2002900100042*

*Date :*

**ABSTRACT**

Brain stroke is a critical medical condition that results in the sudden disruption of blood supply to the brain, leading to severe damage and sometimes even death. It has become a major health concern worldwide, and the incidence of brain stroke is increasing due to lifestyle changes and an aging population. Predicting brain stroke accurately has become a complicated and challenging task, requiring the analysis of vast amounts of medical data.

Machine learning techniques can provide an efficient way to predict brain stroke. This research aims to predict brain stroke using machine learning techniques and analyse the algorithms used for the prediction. The paper proposes a robust system that uses a dataset with 11 attributes, including gender, age, hypertension, heart disease, ever married, work type, residence type, average glucose level, BMI, smoking status, and previous stroke history. Different algorithms, including SVM, Naïve Bayes, Logistic Regression, Decision Tree, Random Forest, and KNN, are implemented to predict brain stroke accurately.

Logistic Regression produced the best result with an accuracy of 95.3%. The paper also presents a comparative analysis of all the algorithms used. The research uses model validation techniques to design the best suitable model for the current scenario. Early prediction of brain stroke can save many lives, and this research aims to design an accurate prediction system that overcomes the limitations of existing systems. The study provides insights into the most critical factors contributing to brain stroke and proposes an efficient system for predicting and preventing brain stroke.

**LIST OF TABLES**

|  |  |  |
| --- | --- | --- |
| Table No. | Table Name | Page no |
| 4.1 | Accuracy Percentage of different algorithms | **33** |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| Fig. No. | Fig. Name | Page No. |
| 3.1 | Logistic Regression Graph | **5** |
| 3.2 | Flow-chart of Decision Tree | **5** |
| 3.3 | Working of Decision Tree | **6** |
| 3.4 | Working of Random Forest | **7** |
| 3.5 | Working of KNN Classifier | **8** |
| 3.6 | Working of SVM | **9** |
| 3.7 | Graph Of Naive Bayes Classifier |  |
| 3.8 | Importing numpy and pandas | **10** |
| 3.9 | read\_csv() | **10** |
| 3.10 | describe() function | **10** |
| 3.11 | info() function | **11** |
| 3.12 | head() function | **11** |
| 3.13 | isnull() function | **12** |
| 3.14 | Converting into categorical values | **12** |
| 3.15 | Attributes after conversion into categorical types | **13** |
| 3.16 | Count of people vs. Gender | **13** |
| 3.17 | Count of people vs. Age | **14** |
| 3.18 | Count of people vs. Hyper Tension | **14** |
| 3.19 | Count of people vs. Heart Disease | **15** |
| 3.20 | Count of people vs. Ever Married | **15** |
| 3.21 | Count of people vs. Work Type | **16** |
| 3.22 | Count of people vs. Residence Type | **16** |
| 3.23 | Count of people vs. Avg Glucose Level | **16** |
| 3.24 | Count of people vs. Body Mass Index | **17** |
| 3.25 | Count of people vs. Smoking Status | **18** |
| 3.26 | HEATMAP | **18** |
| 3.27 | BMI vs Age | **19** |
| 3.28 | Average Glucose Level vs. Age | **20** |
| 3.29 | BMI vs. Average Glucose Level | **20** |
| 3.30 | Age vs. Heart Disease | **21** |
| 3.31 | Age vs. Hypertension | **22** |
| 3.32 | Features or Target |  |
| 3.33 | train\_test\_split() function | **23** |
| 3.34 | Import MinMaxScaler() | **23** |
| 3.35 | MinMaxScaler() function on Train data | **24** |
| 3.36 | MinMaxScaler() function on Test data | **25** |
| 4.1 | Importing Logistic Regression | **29** |
| 4.2 | Training Model | **29** |
| 4.3 | Accuracy Percentage | **29** |
| 4.4 | Importing DecisionTreeClassifier | **30** |
| 4.5 | Training the model | **30** |
| 4.6 | Accuracy Percentage | **30** |
| 4.7 | Importing RandomForestClassifier | **31** |
| 4.8 | Training the model | **31** |
| 4.9 | Accuracy Percentage | **31** |
| 4.10 | Importing KNeighboursClassifier | **32** |
| 4.11 | Training the model | **32** |
| 4.12 | Accuracy Percentage | **32** |
| 4.13 | Importing SVM | **33** |
| 4.14 | Training the model | **33** |
| 4.15 | Accuracy Percentage | **33** |
| 4.16 | Importing naiveBayesClassifier | **34** |
| 4.17 | Training the model | **34** |
| 4.18 | Accuracy Percentage | **34** |
| 4.19 | Accuracy Percentage of different algorithms | **35** |

**CHAPTER 1**

**INTRODUCTION**

* 1. **Problem Definition: -**
* Brain stroke is a major cause of death, accounting for 11% of global deaths.
* Therefore to identify people having heart disease we are using data analysis and ML model for prediction.
* We are predicting whether a person has heart disease or not through various ml models.
  1. **Motivation:**
* As mentioned in our problem statement, this model aims to predict the occurrence of brain stroke in individuals.
* With the current state of the world, brain stroke is one of the leading causes of mortality and morbidity worldwide.
* Utilizing data analysis and machine learning, this model uses a set of 12 attributes to identify individuals at risk for brain stroke and enable early intervention.
* Early detection and treatment of brain stroke is crucial to reduce its impact and improve the patient's quality of life.
* Through this model, we aim to contribute to reducing the burden of brain stroke on the healthcare system and society as a whole.
  1. **Objective of the Project:** The Project is based on a machine learning model that can predict if a person has a stroke or not.
  2. **Scope of the Project:** The scope of the project is very vast, as it can be used in early detection of stroke which helps in reducing the death rate due to brain stroke.
  3. **Need of Work: -** The objective of this work is early detection of stroke. It improves accuracy and helps individuals to get personalized treatments. It further reduces healthcare costs.

**CHAPTER 2**

**RELATED WORK**

In this section, we will list the papers that were reviewed to obtain relevant information about the dataset and its attributes for the project. These papers offered valuable insights into the data, such as its distinguishing features and characteristics. By reviewing these papers, we have acquired a foundational understanding of the dataset and its attributes, which will aid us in developing the project.

* Govindarajan et al. [3] conducted a study to categorize stroke disorder using a text mining combination and a machine learning classifier and collected data for 507 patients. They used various machine learning approaches for training purposes using ANN, and the SGD algorithm gave them the best value, which was 95%.
* Amini et al. [1] conducted research to predict stroke incidence, collected 807 healthy and unhealthy subjects 11in their study categorized 50 risk factors for stroke, diabetes, cardiovascular disease, smoking, hyperlipidemia, and alcohol use. They used two techniques that had the best accuracy from c4.5 decision tree algorithm, and it was 95%, and for K-nearest neighbor, the accuracy was 94%
* Cheng et al. [4] published a report on the estimation of the ischemic stroke prognosis. In their analysis, 82 ischemic stroke patient data were used, two ANN models were used to find precision, and 79% and 95% were used.
* Cheon et al. [5] performed a study to predict stroke patient mortality. In their study, they used 15099 patients to identify stroke occurrence. They used a deep neural network approach to detect strokes. The authors used PCA to extract medical record history and predict stroke. They have got an area under the curve (AUC) value of 83%.
* Singh et al. [6] performed a study on stroke prediction applied to artificial intelligence. In their research, they used a different method for predicting stroke on the cardiovascular health study (CHS) dataset. And they took the decision tree algorithm to feature extract to principal component analysis. They used a neural network classification algorithm to construct the model they got 97% accuracy.
* Chin et al. [7] performed a study to detect an automated early ischemic stroke. In their study, the main purpose was to develop a system using CNN to automated primary ischemic stroke. They collected 256 images to train and test the CNN model. Their CNN method has given 90% accuracy.
* Sung et al. [2] performed a study to develop a stroke severity index. They collected 3577 patient’s data with acute ischemic stroke. For their predicting models, they used various data mining techniques and linear regression. Their prediction feature got the best result from the k-nearest neighbor model (95% CI).
* Monteiro et al. [8] performed a study to get a functional outcome prediction of ischemic stroke using machine learning. In their research, they apply this technique to a patient who was passing three months after admission. They got the AUC value above 90%.
* Kansadub et al. [9] performed a study to predict stroke risk. In the study, the authors employed Naive Bayes, Decision Tree, and Neural Network to analyze data to predict stroke. In their study, they used accuracy and AUC as their pointer’s assessment. All of this algorithm, they classified decision tree and naive Bayes gave the most accurate.
* Adam et al. [10] performed a study to classify ischemic stroke. They used two models: a k-nearest neighbor and a decision tree algorithm to classified ischemic stroke. In their research, the decision tree algorithm was more usable for medical specialists who used it to classify stroke.

**CHAPTER 3**

**PROPOSED METHODOLOGY**

* 1. **Dataset Description**

The dataset for our Brain Stroke prediction was downloaded from Kaggle.

According to the World Health Organization (WHO) stroke is the 2nd leading cause of death globally, responsible for approximately 11% of total deaths.

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 relavant information about the patient.

The data set has 5110 rows.

The number of attributes it possesses is 12. These are listed below:

**ID**: - The ID serves as a distinctive marker for each individual's information.

**Gender:** - This characteristic identifies the gender of a person, whether it is "male", "female", or other.

**Age: -** This attribute tell the age of the patient.

**Hypertention: -** If the patient is not suffering from hypertension, the value assigned is 0, while if the patient has hypertension, the value assigned is 1.

**Heart disease**: - The binary variable assigned to a patient indicates their heart disease status, with 0 indicating no heart disease and 1 indicating the presence of heart disease.

**Marital Status: -** This attribute takes the marital status and tells whether the individual is Married or not married.

**Work type: -** The classification of an individual's profession can be determined by this characteristic, which may encompass "Private job," "Government job," "never worked," "self-employed," or being a child.

**Residence type: -** This attribute is indicative of whether a person resides in a rural or urban locality.

**Average glucose level:** - It consists of the average blood sugar level of the person.

**Body mass index: -** The person's body mass index is indicated by this attribute.

**Smoking status:** - "formerly smoked", "never smoked", "smokes" or "Unknown"

**Stroke: -** 1 if the patient had a stroke or 0 if not

\*Note: "Unknown" in smoking\_status means that the information is unavailable for this patient

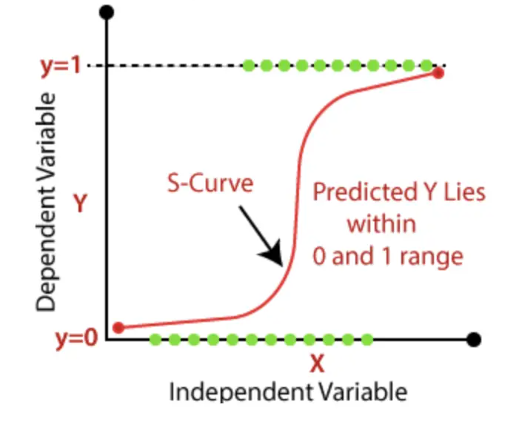
**3.2 Methods:**

We began by downloading the dataset. We then analyzed the dataset's attributes to understand what they were and how they were related.

We then used matplotlib and seaborn to visualize the data so that we could see how the attributes were distributed. Finally, we applied various machine learning models to the data to see how well they could predict the target variable.

**LOGISTIC REGRESSION:**

* Logistic regression is a supervised machine learning algorithm that predicts a categorical dependent variable based on a given set of independent variables. It is used for classification problems, where the outcome is a categorical value such as Yes or No, 0 or 1, or true or false. The algorithm predicts the output as probabilistic values between 0 and 1.
* Unlike linear regression, which is used for regression problems, logistic regression fits an "S" shaped logistic function to predict the maximum values of 0 or 1. This curve indicates the likelihood of a particular event, such as whether a cell is cancerous or not.
* Logistic regression is significant because it can provide probabilities and classify new data using both continuous and discrete datasets. It can also determine the most effective variables for classification.



**Fig. 3.1: Logistic Regression Graph**

There are three main types of logistic regression:

* **Binary logistic regression:** This type of logistic regression is used when the dependent variable has only two possible outcomes, such as "pass" or "fail", "yes" or "no", or "0" or "1".
* **Multinomial Logistic Regression:** This type of logistic regression is used when the dependent variable has three or more possible outcomes, but the outcomes are not ordered. For example, you could use multinomial logistic regression to predict whether a customer will buy a "cat", "dog", or "sheep".
* **Ordinal Logistic Regression** This type of logistic regression is used when the dependent variable has three or more possible outcomes, and the outcomes are ordered. For example, you could use ordinal logistic regression to predict whether a customer will rate a product as "good", "bad", or "neutral".

**Advantages:**

* **Easy to implement:** Logistic regression is a relatively simple algorithm to implement, making it a good choice for beginners.
* **Interpretable:** The results of logistic regression are easy to interpret, making it a good choice for understanding the relationships between variables.
* **Efficient to train:** Logistic regression is a relatively efficient algorithm to train, making it a good choice for large datasets.
* **Widely used by data analysts and scientists:** Logistic regression is a widely used algorithm by data analysts and scientists, making it a good choice for working with data.

**Disadvantages:**

* **Not able to handle a large number of categorical features/variables:** Logistic regression is not able to handle a large number of categorical features/variables, as this can lead to overfitting.
* **Constructs linear boundaries:** Logistic regression constructs linear boundaries, which may not be accurate if the relationship between the dependent and independent variables is nonlinear.

**DECISION TREE:**

Decision trees are a supervised learning method that can be used for both classification and regression problems. They are commonly used for classification tasks, where the goal is to predict a class label for each data point. Decision trees work by creating a tree-like structure, where each node represents a feature of the data, each branch represents a decision rule, and each leaf node represents a class label.

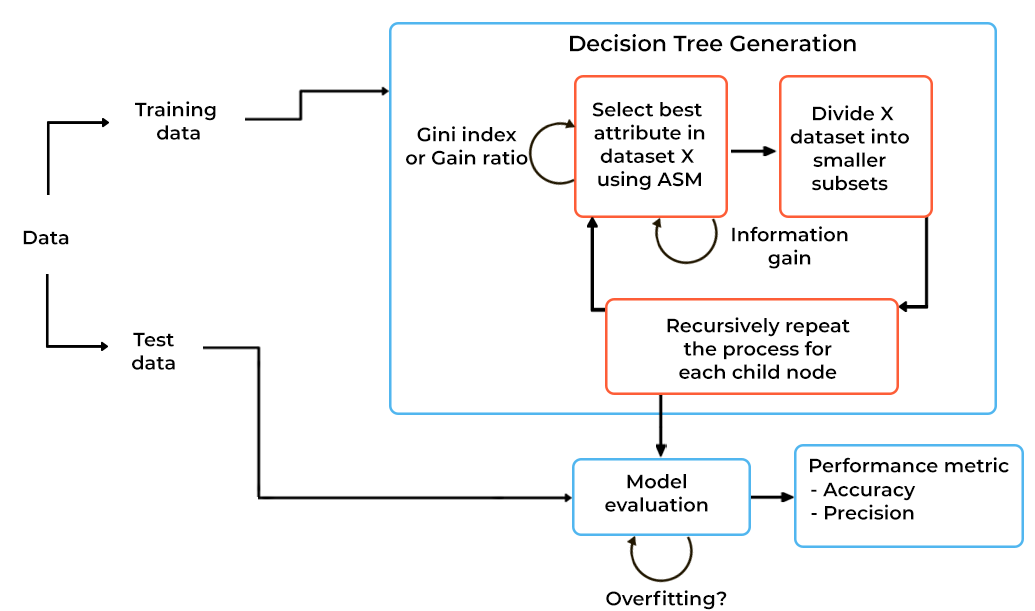
The tree is created by recursively splitting the data into smaller and smaller subsets until each subset contains only data points of a single class.

Decision nodes are responsible for making decisions and have multiple branches. In contrast, leaf nodes are the final outputs resulting from those decisions and do not have any additional branches. The decisions made by decision nodes are based on the features present in the given dataset. A decision tree is a visual representation that illustrates all the potential solutions to a problem or decision based on specified conditions. The CART (Classification and Regression Tree) algorithm is utilized to construct a decision tree.



**Fig. 3.2: Flow-chart of Decision Tree**

How does the Decision Tree work?

Decision trees employ a range of algorithms to determine whether a node should be partitioned into two or more sub-nodes. The creation of sub-nodes enhances the homogeneity of the resulting sub-nodes. In simpler terms, the purity of the node with respect to the target variable is increased. The decision tree then proceeds to split into sub-nodes.

**Fig. 3.3: Working of Decision Tree**

**Advantages:**

* **Simple to comprehend and visualize:** Decision trees are easy to understand because they are represented as a tree-like structure. This makes it easy to see how the data is being split and how the predictions are being made.
* **Can effectively identify non-linear patterns:** Decision trees can identify non-linear patterns in data, which means they can be used to make predictions about data that is not well-represented by linear models.
* **Users don't need to perform as much data pre-processing:** Decision trees do not require as much data pre-processing as other machine learning algorithms. This means that you can use decision trees to make predictions about data that is not well-prepared.
* **Decision trees are useful for feature engineering tasks:** Decision trees can be used for feature engineering tasks, such as filling in missing values or selecting variables. This makes them a valuable tool for data scientists.
* **The algorithm's non-parametric nature allows it to avoid making assumptions about data distribution:** Decision trees are non-parametric, which means they do not make assumptions about the data distribution. This makes them a valuable tool for data scientists who are working with data that is not well-represented by parametric models.

**Disadvantages:**

* **Decision trees are prone to overfitting noisy data:** Decision trees are based on a greedy algorithm, which means that they make decisions based on the best available information at the time. However, this can lead to overfitting, which is when the model learns the noise in the data instead of the underlying patterns.
* **Slight variations in data can lead to different decision trees:** Decision trees are also prone to instability, which means that slight variations in the data can lead to different decision trees being created. This can make it difficult to trust the predictions of the model.
* **The algorithm may exhibit bias when dealing with imbalanced datasets:** Decision trees can also be biased, which means that they are more likely to make predictions that favor one class over another. This is a problem when the data is imbalanced, which means that there are more data points in one class than in another.

**RANDOM FOREST:**

Random forest is a supervised learning algorithm that can be used for both classification and regression problems. It is based on the idea of ensemble learning, which is a technique that combines multiple models to improve the performance of the overall model. In the case of random forest, the individual models are decision trees.

Decision trees are a type of supervised learning algorithm that can be used to make predictions about data. They work by recursively splitting the data into smaller and smaller subsets until each subset contains only data points of a single class. The splitting is done by selecting the feature that best separates the data into two classes.

Random forest works by creating a number of decision trees, each of which is trained on a different subset of the data. The predictions of the individual trees are then averaged to produce the final prediction. This helps to reduce overfitting, which is a problem that can occur when a model is too closely tied to the training data.

**Why Random Forest?**

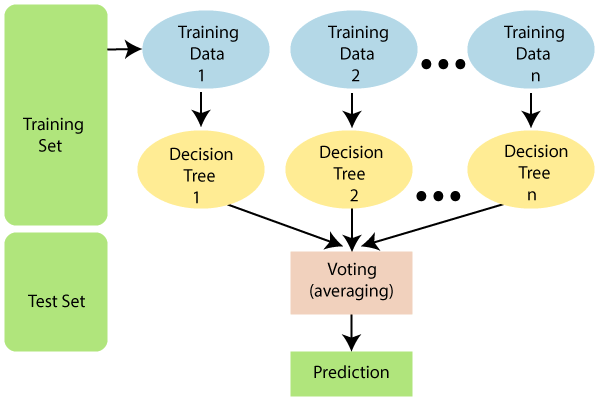
Here are some reasons why the Random Forest algorithm is advantageous:

* **It is robust to overfitting.** Overfitting is a problem that can occur when a model learns the noise in the data instead of the underlying patterns. Random forest is less likely to overfit because it uses a set of decision trees, each of which is trained on a different subset of the data. This helps to reduce the risk of overfitting.
* **It is able to handle missing data.** Random forest is able to handle missing data because it uses a technique called imputation. Imputation is a process of filling in missing values with estimates from the other data points. This helps to improve the accuracy of the model.
* **It is able to generalize to new data.** Random forest is able to generalize to new data because it is a non-parametric algorithm. Non-parametric algorithms do not make assumptions about the data distribution. This makes them more robust to changes in the data distribution.

**How does the algorithm work?**

The Random Forest algorithm is a type of ensemble learning that blends several decision trees to create predictions. The following is a succinct description of how the Random Forest algorithm works:

* Data Preparation: This involves furnishing a training dataset to the algorithm, which is then utilized to train the decision trees. The dataset is partitioned into two parts, one for training and the other for testing purposes.
* Random Sampling: In order to avoid overfitting, the algorithm utilizes the method of random sampling to choose a random subset of the training dataset for each decision tree. This ensures that each tree is trained on a unique subset of data.
* Building Decision Trees: The algorithm then constructs several decision trees from the chosen subset of data. Using a recursive process, the decision trees are created by splitting the data into smaller and smaller subsets based on various features until a stopping criterion is reached. A stopping criterion could be the maximum depth of the tree, the minimum number of samples in a leaf node, or a minimum reduction in impurity.
* Predictions: The algorithm traverses the decision trees and takes the majority vote of the predicted values to make predictions on the test dataset.
* Aggregating Predictions: The final step of the Random Forest algorithm is to combine the predictions of all the decision trees. For classification problems, the majority vote is taken as the final prediction. For regression problems, the average of the predicted values is taken.



**Fig. 3.4: Working of Random Forest**

**Advantages:**

* Random forests are a reliable and precise technique because they use multiple decision trees. This means that the model is less likely to make mistakes because it is not relying on a single tree.
* Random forests are not prone to overfitting because the predictions of the individual trees are averaged together. This helps to reduce the bias in the model.
* Random forests can be used for both classification and regression problems.

## Disadvantages:

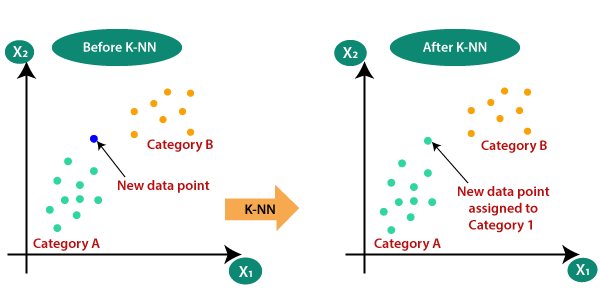
* Random forests are a slow method of generating predictions because they use multiple decision trees. Each time a prediction is required, all trees in the forest must make a prediction for the same input, followed by voting. This entire process is time-intensive.

**KNN CLASSIFIER :**

* The K-Nearest Neighbors algorithm is a basic supervised learning algorithm that classifies new data points by finding the most similar existing data points. It does this by storing all available data and then classifying new data based on its similarity to the existing data. This means that K-NN can easily classify new data into appropriate categories. While K-NN can be used for both regression and classification, it is primarily used for classification problems.
* K-NN is a non-parametric algorithm that does not make assumptions about the data it is used on. This means it can work with a variety of data types without having to be specifically trained for each type. It is also known as a lazy learner algorithm because it does not immediately learn from the data it is trained on. Instead, it stores the data and then classifies new data based on how similar it is to the stored data. This can be computationally expensive, but it makes K-NN a very accurate algorithm.

**The KNN algorithm comprises the following fundamental steps:**

1. Calculate the distance between a specific data point and all the remaining data points in the dataset.
2. Find the data points in the dataset that are most similar to the given data point, using the distance metric that is used.
3. Find the labels that are associated with the data points in the dataset that are most similar to the given data point.
4. Assign a label to the given data point based on the most common label among the K data points that are most similar to the given data point.



**Fig. 3.5: Working of KNN Classifier**

**SUPPORT VECTOR MACHINE ALGORITHM :**

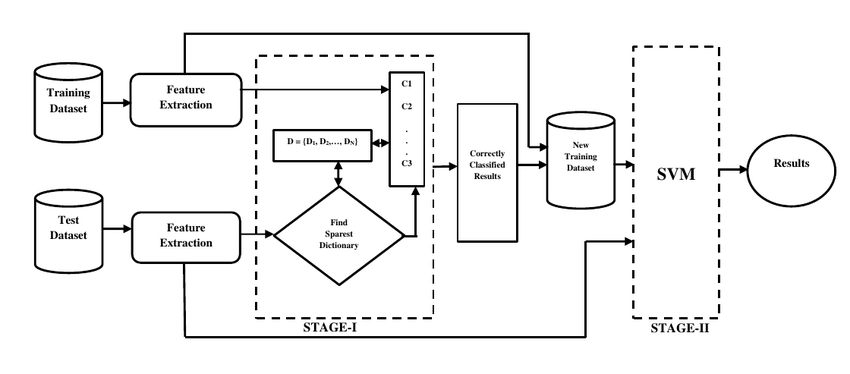
Support Vector Machines (SVM) are a supervised learning algorithm that can be used for both classification and regression problems. However, they are mainly used for classification tasks.

The goal of SVM is to create the optimal hyperplane, which is a line or decision boundary that can effectively separate data points into distinct classes in n-dimensional space. This hyperplane allows the SVM algorithm to easily assign new data points to the correct category.

SVM selects the most important points or vectors that help to create the hyperplane. These important instances are referred to as support vectors, and the algorithm is named Support Vector Machine because of their use.

**There are two main types of SVM:**

* **Linear SVM -** The Linear Support Vector Machine (SVM) is a perfect fit for datasets that can be divided into two classes using a straight line, which is why a Linear SVM classifier is used.
* **Non-Linear SVM** - Non-linear SVMs are used for data that is not linearly separable. They map the data into a higher dimension, where it can be separated by a hyperplane.

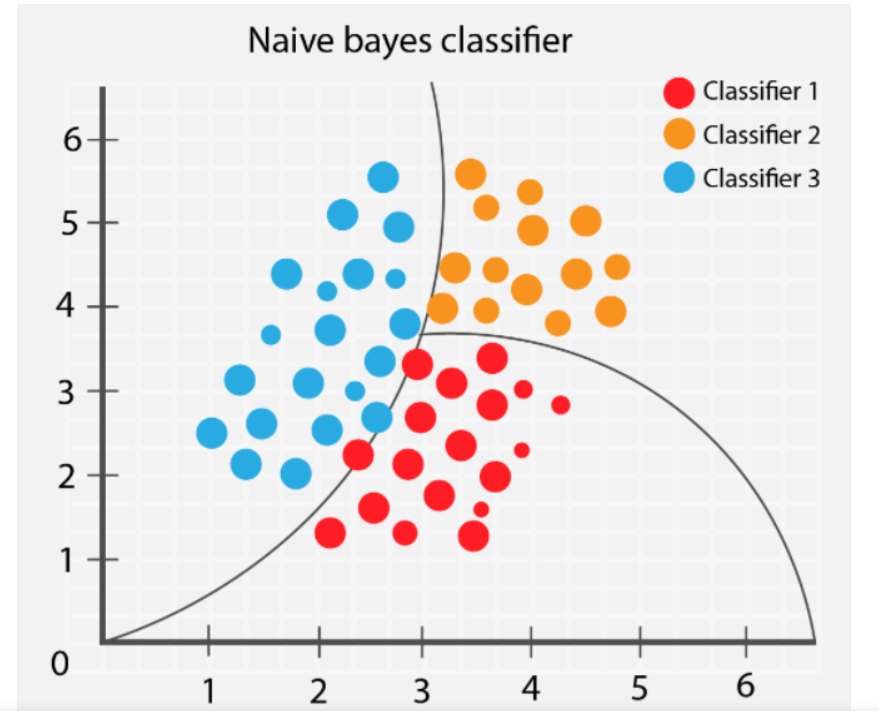


**Fig. 3.6: Working of SVM**

**NAIVE BAYES CLASSIFIER :**

The Naïve Bayes algorithm is a supervised learning technique that uses Bayes' theorem for classification problems. It is mainly used in text classification problems that involve large, high-dimensional training datasets. The Naïve Bayes Classifier is a simple and effective classification algorithm that allows for the creation of fast machine learning models that can make quick predictions.

Naïve Bayes is a probabilistic classifier that makes predictions based on the probability of an object being a member of a particular class. It is frequently used in a variety of applications, such as spam filtering, sentiment analysis, and article classification.



**Fig. 3.7: Graph of Naive Bayes Classifier**

**3.3 Hardware / Software Requirements**

**Minimums Hardware Requirements:**

RAM: 2 GB

Processor: Intel Core 2 Duo

Hard disk: 50GB

**Minimums Software Requirements:**

Pandas: 0.24.2

Numpy; 1.16.4

Matplotlib: 3.1.0

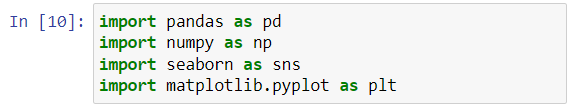
Seaborn: 0.9.0

Python: 3.7

Scikit-Learn: 0.21.2

**3.4 Our Methodology:**

* We began by downloading the Stroke Prediction dataset from Kaggle. Then we loaded the data set into Jupyter Notebook



**Fig. 3.8: Importing numpy and pandas**

We first imported the following Python libraries:

**NumPy:**  a numerical Python package used for working with arrays.

**Pandas:**a high-level open-source data analysis and manipulation tool for Python.

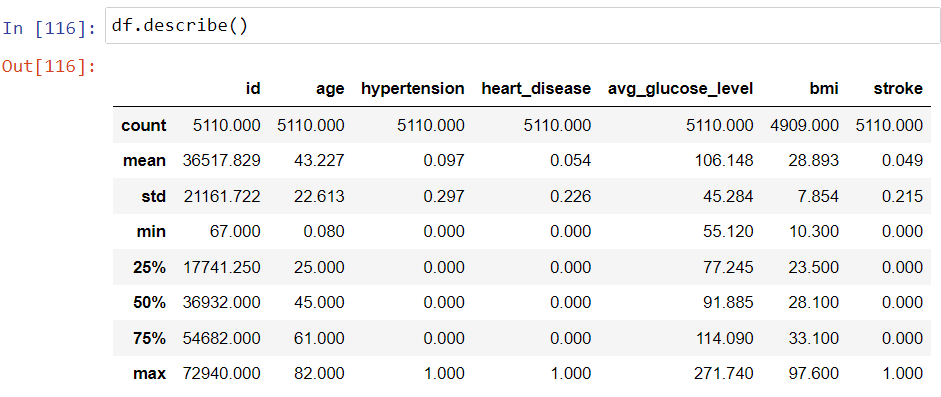
**Seaborn:** a Python data visualization library based on matplotlib. Seaborn provides a number of features that make it easy to create beautiful and informative visualizations.

**Matplotlib:** a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib is a powerful tool that can be used to create a wide variety of visualizations.



**Fig. 3.9: read\_csv()**

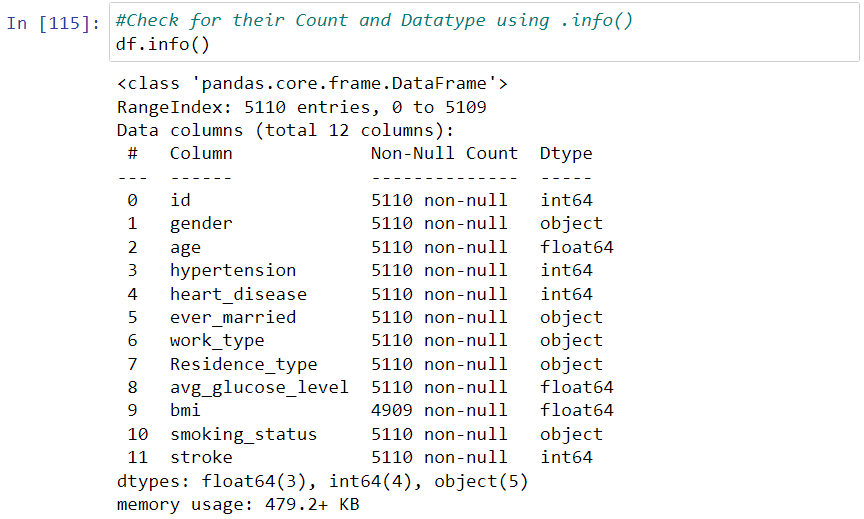
* **Describing the data:**



**Fig. 3.10: describe() function**

Call the **`describe()`** method on the dataframe to get a statistical summary of the data.

* In total, there exist 7 integer attributes.

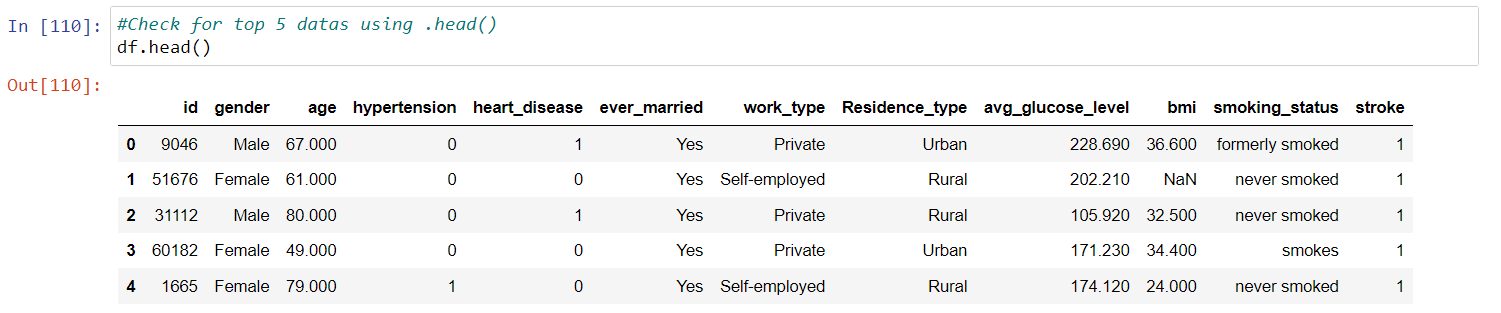


**Fig. 3.11: info() function**

The **info()** function provides a brief overview of a DataFrame, including the number of rows and columns, the data types of each column, the number of non-null values, and the total memory usage.

* There are 5 object variable
* There are 4 int64 variable
* There are 3 float64 variables
* There are 5110 rows entries

The total size of data set is 479.2+ KB

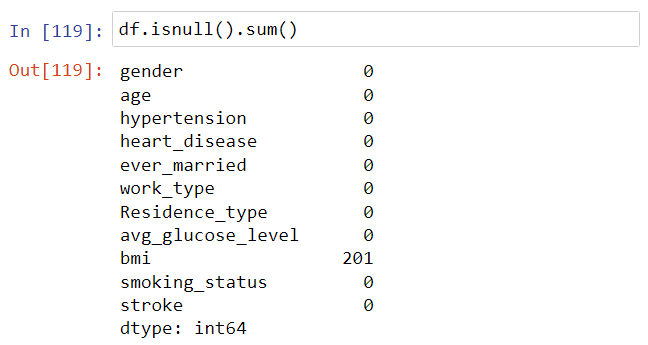


**Fig. 3.12: head() function**

The **head()** method returns the first few rows of a DataFrame. The number of rows can be specified, or it will default to 5 rows

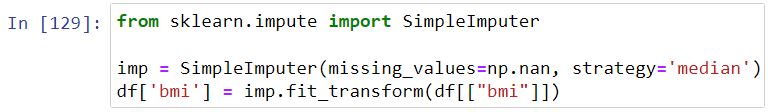
* **CHECKING NULL VALUES FOR EACH COLUMN**

Checking for missing values by using **isnull()** method.



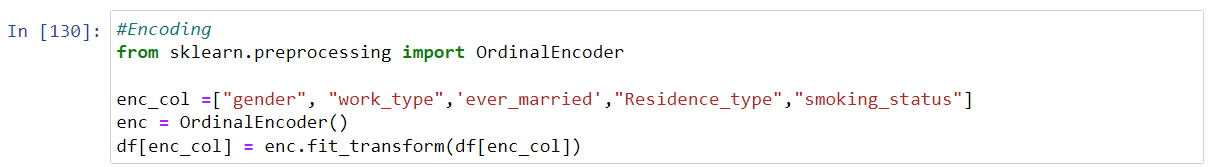
**Fig. 3.13: isnull() function**

* There are total 201 missing values in bmi attribute.
* **Imputing missing BMI values with its median value**

****

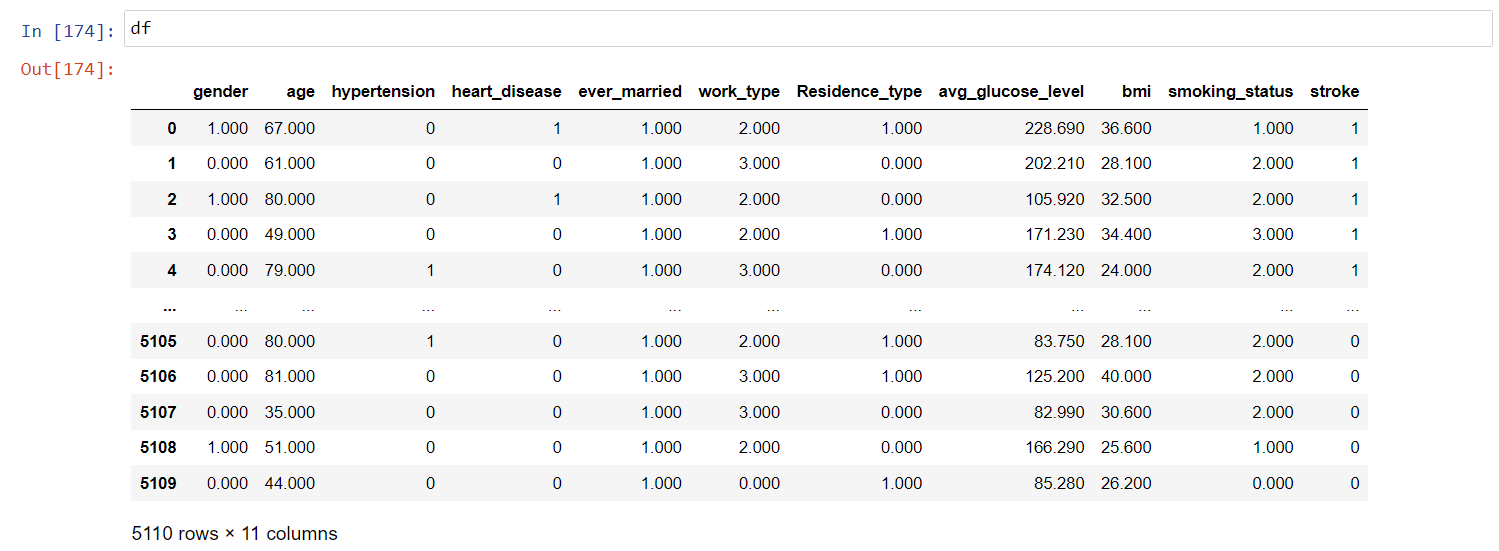
* **CONVERSION INTO CATEGORICAL VARIABLES:**

Assigning numerical values to categorical variables for data analysis purposes.



**Fig. 3.14: Converting into categorical values**

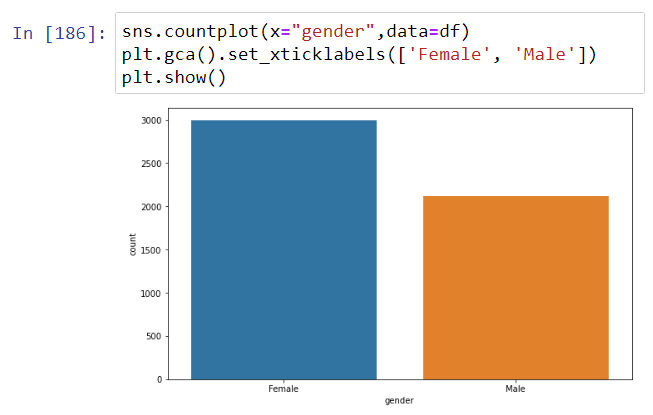
Dataset with numerically coded value for each categorical value in columns.



**Fig. 3.15: Attributes after conversion into categorical types**

* **ANALYSING EACH ATTRIBUTE**

**GENDER :**

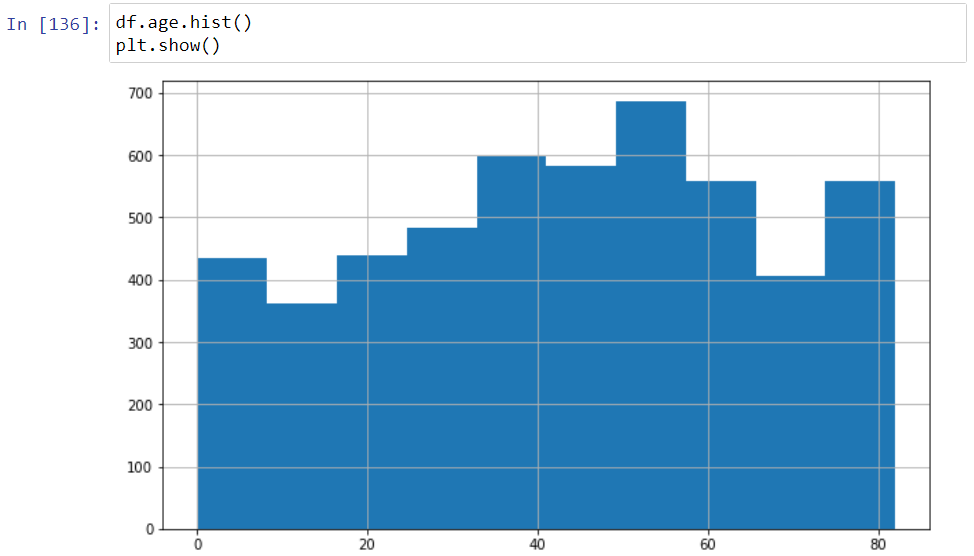


**Fig. 3.16: Count of people vs. Gender**

From the above graph:-

* Majority of candidates are female in the above dataset.

**AGE :**

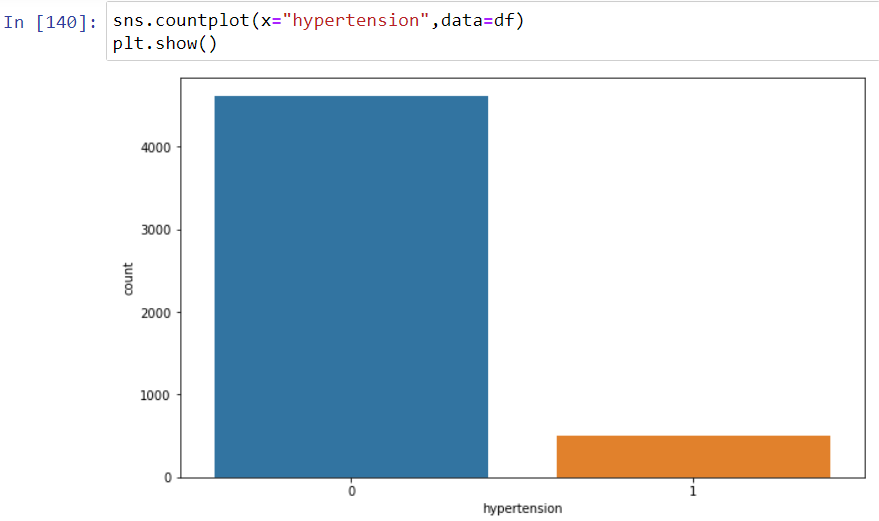


**Fig. 3.17: Count of people vs. Age**

From the above graph:-

* The visualization above displays how the ages are distributed among the entries in our dataset.
* The ages range from 0 to 82 years old.
* It has the majority of entries between the ages of 32 and 66 years.

**HYPERTENSION:**

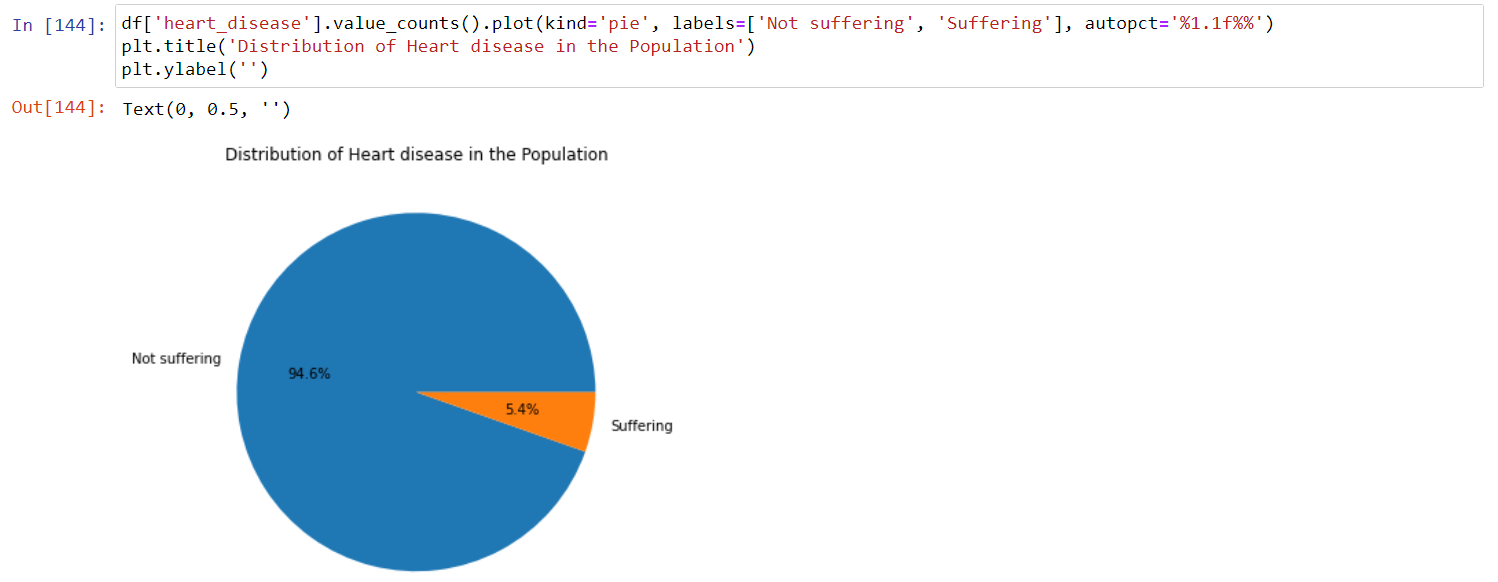


**Fig. 3.18: Count of people vs. Hypertension**

From the above graph:-

* Majority of candidates are not suffering from hypertension.
* A large proportion (90.3%) of the population is not afflicted by hypertension, while a small proportion (9.7%) is experiencing hypertension.

**HEART DISEASE:**

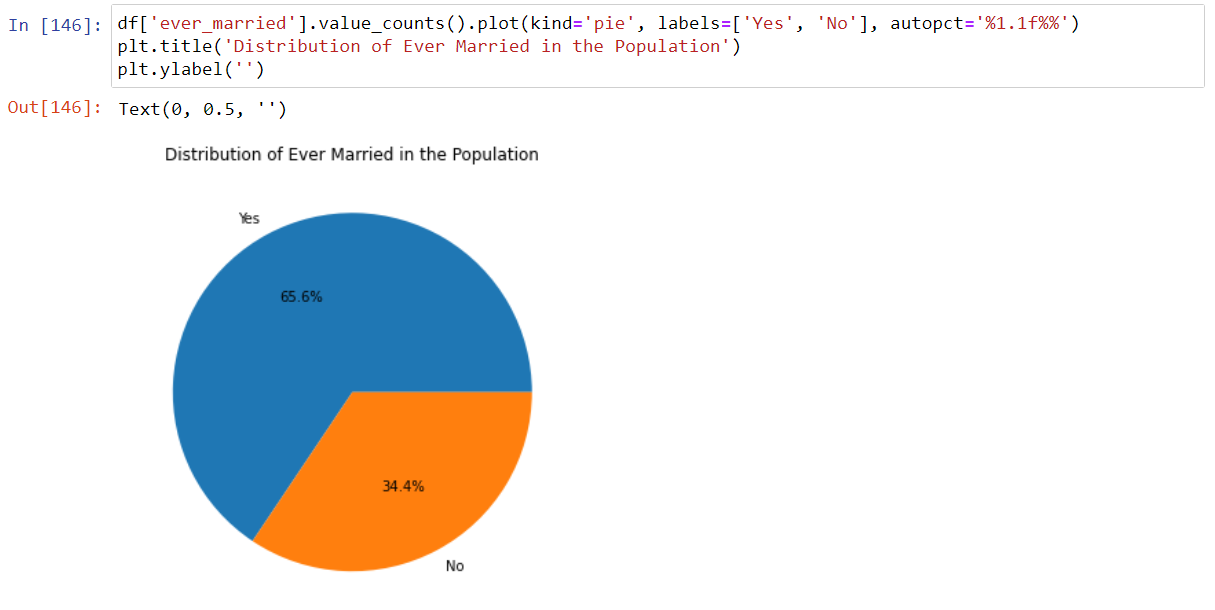


**Fig. 3.19: Count of people vs. Heart Disease**

From the above graph:-

* The majority of candidates are free from Heart Disease.
* Out of the total population, a mere 5.4% are suffering from heart disease, leaving a significant 94.6% without any affliction.

## EVER MARRIED:

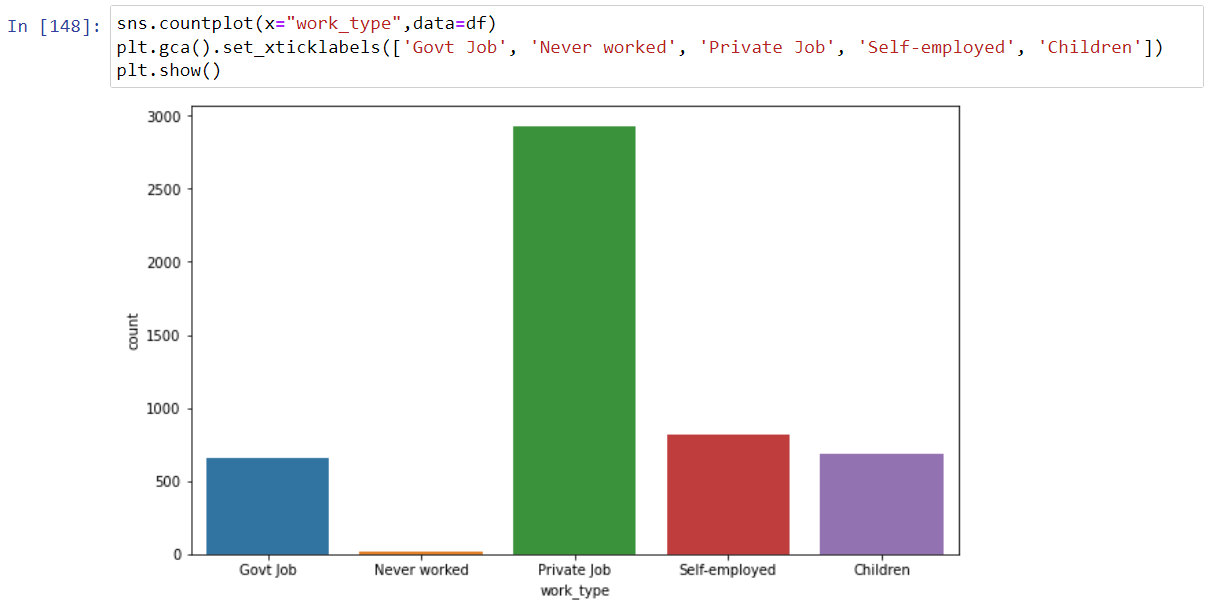


**Fig. 3.20: Count of people vs. Ever Married**

From the above graph:-

* The pie chart indicates that a majority of the population (65.6%) have been married.
* While a minority (34.4%) have never been married.

## WORK TYPE:

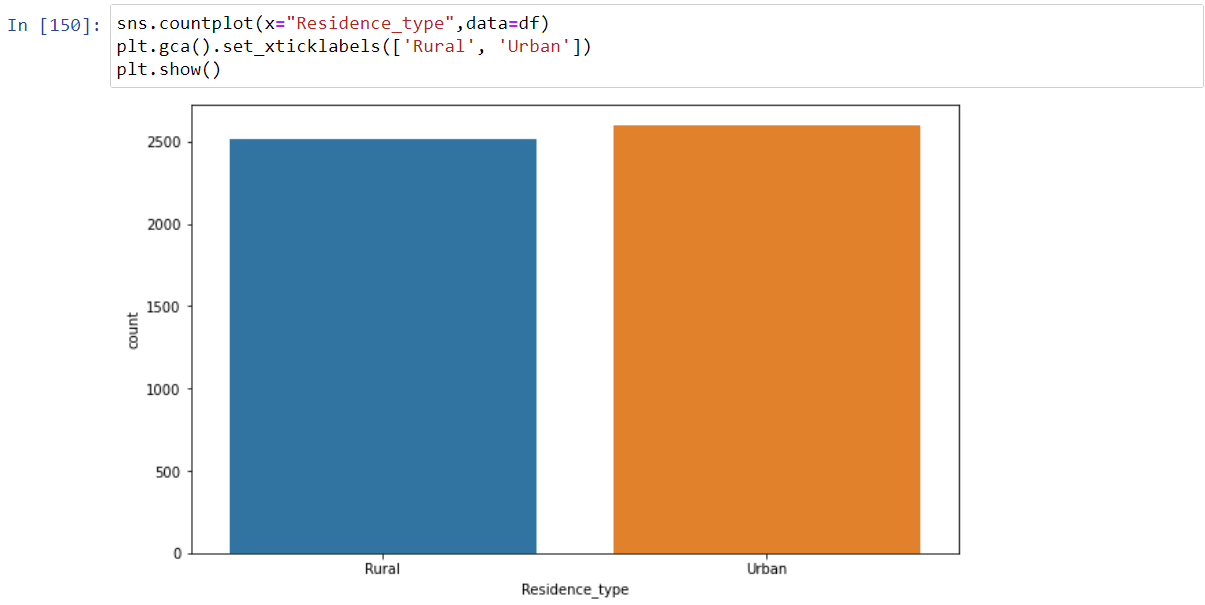
****

**Fig. 3.21: Count of people vs. Work Type**

From the above graph:-

* The largest group of individuals in the dataset work in the private sector, with a count of 2925.
* Self-employed individuals are the next most common group, with a count of 819.
* The smallest groups in the dataset are those who have never worked (22), government job employees (657), and children (687).

**RESIDENCE TYPE:**

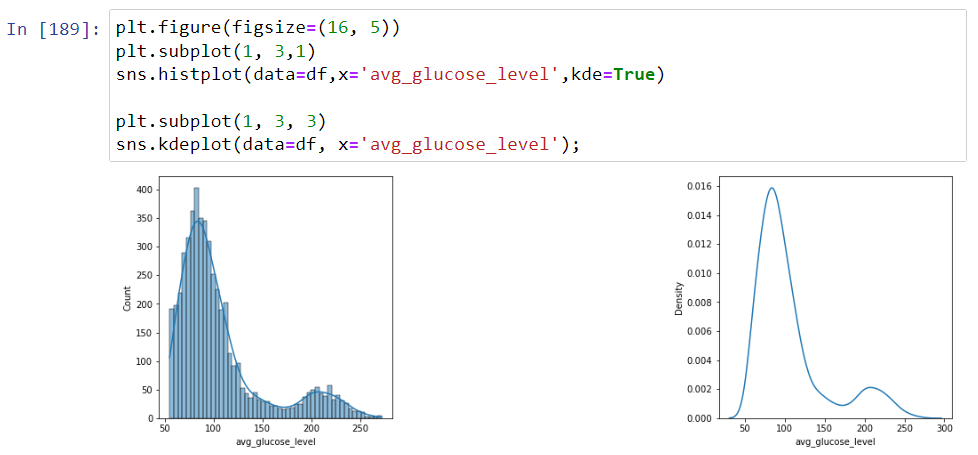


**Fig. 3.22: Count of people vs. Residence Type**

From the above graph:-

* The data shows that the number of individuals living in urban areas is almost equal to those living in rural areas, with urban residents accounting for 50.8% of the population and rural residents accounting for 49.2%.
* The "Urban" and "Rural" categories are almost evenly distributed, with no significant difference in the number of individuals living in each type of area.

**AVERAGE GLUCOSE LEVEL:**

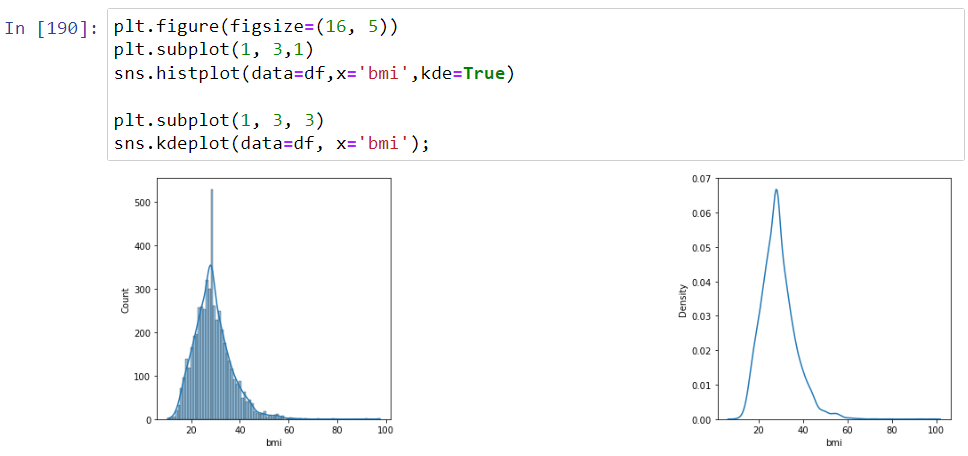


**Fig. 3.23: Count of people vs. Average glucose level**

From the above graph:-

* The distribution of the average glucose level data is positively skewed, with a skewness of 1.5722839, and a kurtosis of 1.6804785, indicating that the data is relatively more spread out than a normal distribution.

**BODY MASS INDEX:**

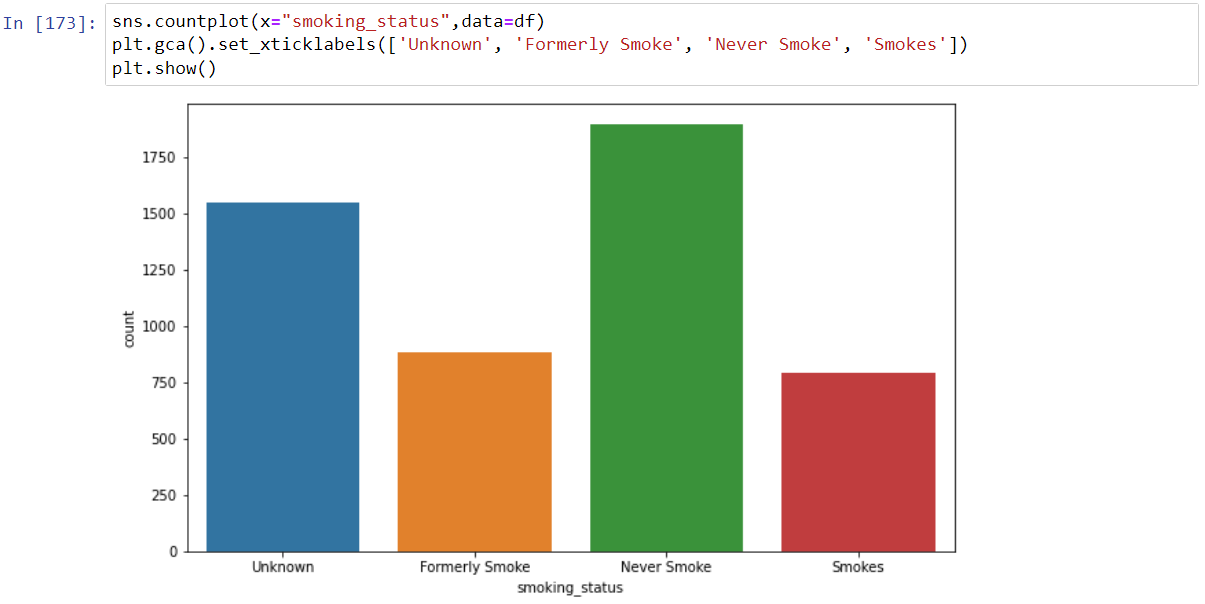


**Fig. 3.24: Count of people vs. Body mass index**

From the above graph:-

* The majority of people in the dataset are overweight or obese.
* The distribution of BMI values is slightly skewed to the right.
* There are some people with lower BMI values in the dataset, but the majority of people have higher BMI values.

**SMOKEING STATUS:**

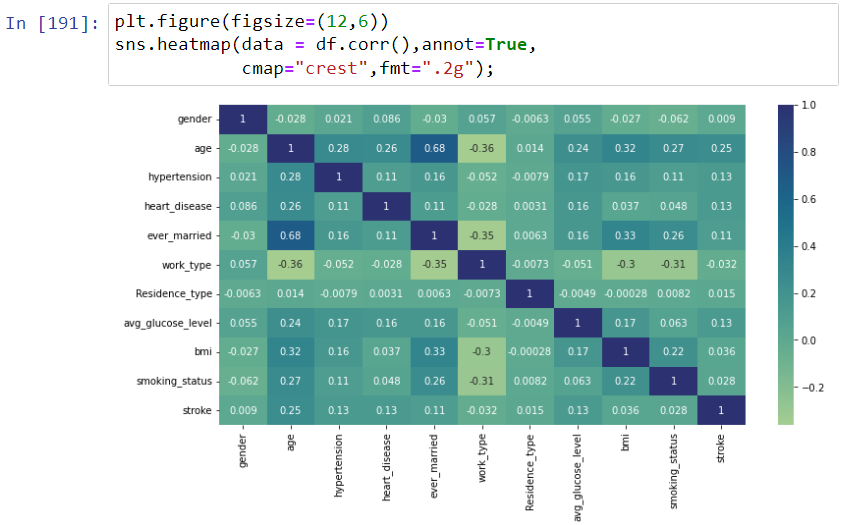


**Fig. 3.25: Count of people vs. Smoking Status**

From the above graph:-

* The majority of people in the dataset have never smoked.
* A significant number of people in the dataset have smoked in the past, but have since quit.
* A small number of people in the dataset currently smoke.

**HEATMAP :**

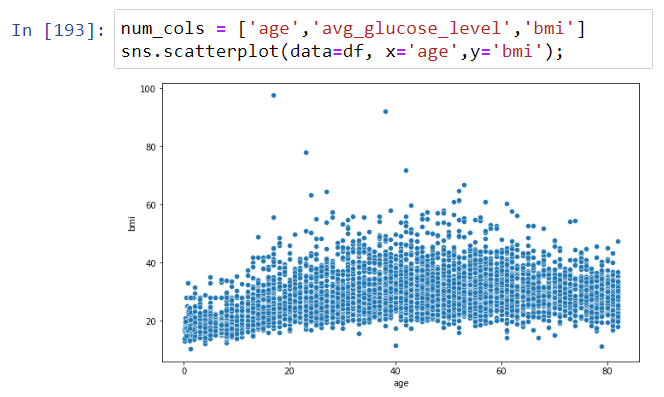


**Fig. 3.26: HEATMAP**

* The correlation matrix of all attributes is shown in a heat map.
* The darker the color, the weaker the correlation.
* Most parts of the heat map are dark.
* Indicating that the attributes are poorly correlated.

**Visualizing relationship between various attributes:**

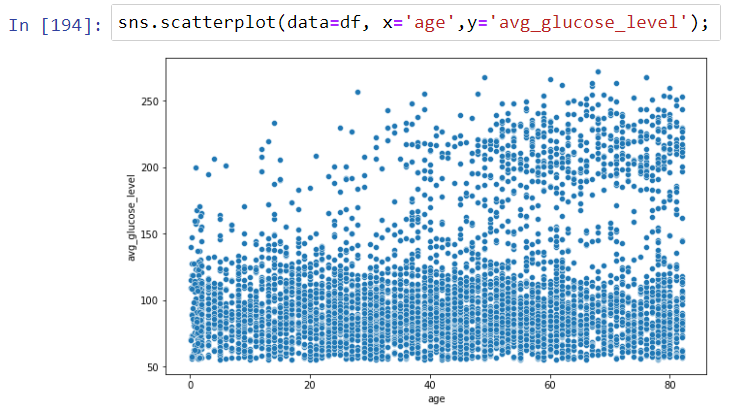
* Age group more prone to a higher BMI -



**Fig. 3.27: BMI vs. Age**

This graph depicts that BMI is high in patients of age between 20-60.

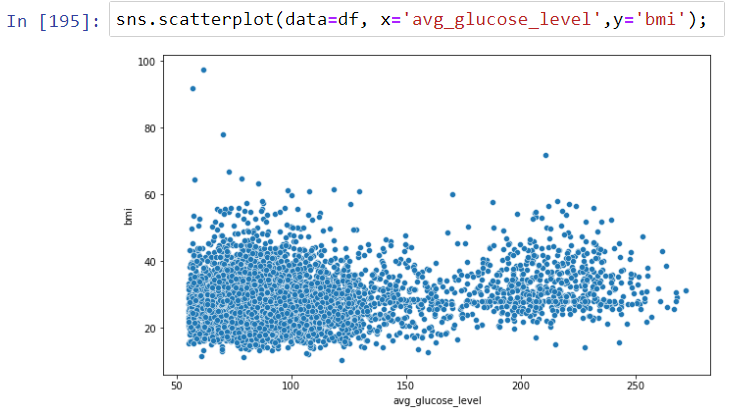
* Which Age group has higher Average glucose level -



**Fig. 3.28: Average glucose level vs. Age**

According to data, high Average glucose level is recorded in patient of age above 50.

* How is the average glucose level of a person related to BMI-

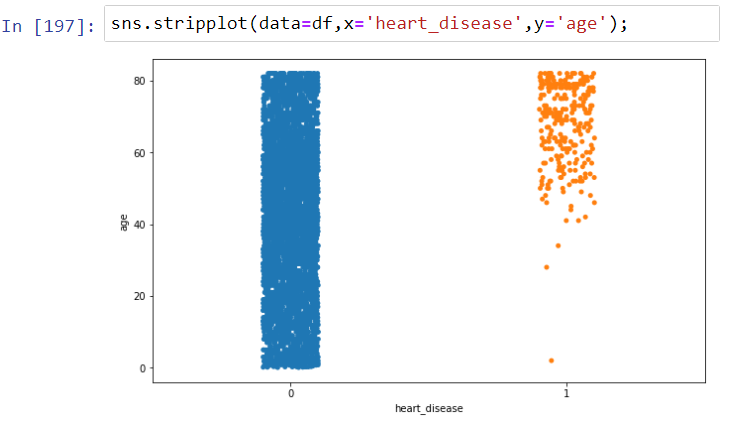


**Fig. 3.29: BMI vs. Average glucose level**

This graph depicts that if glucose level is low then it is more chance that your bmi level is high according to this data.

.

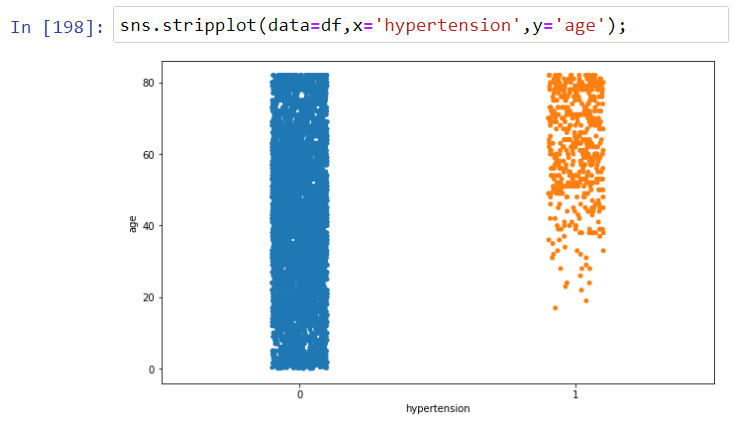
### **Is there a correlation between age and heart disease** -



**Fig. 3.30: Age vs. Heart Disease**

This graph depicts that almost all heart disease people are above 50, which is obvious.

### **What are the effects of hypertension on the body** -



**Fig. 3.31: Age vs. Hypertension**

This graph depicts that Hypertension disease in people of above 50.

**APPLYING VARIOUS MACHINE LEARNING ALGORITHMS :**

We then implement six machine learning models on the dataset: :-

Logistic Regression

Decision Tree

Random Forest

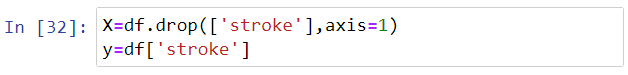
KNN Classifier

Support Vector Machine

Naive Bayes Classifier

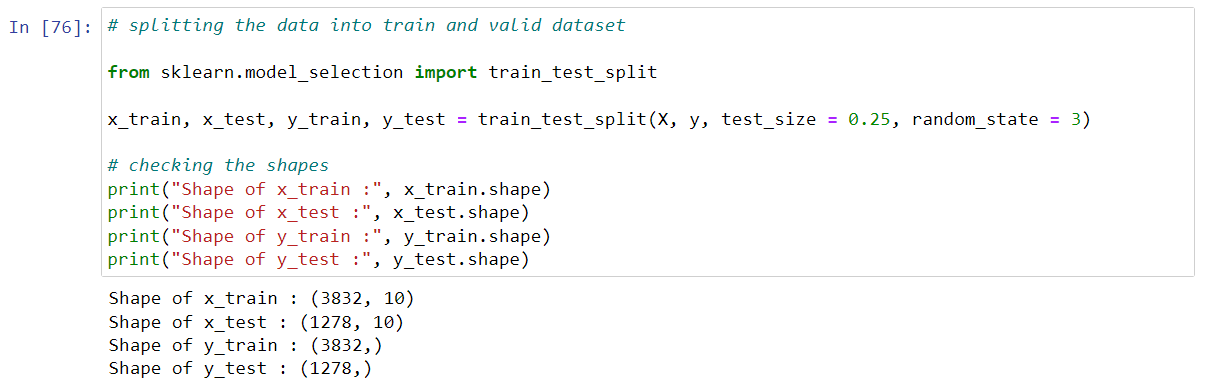
After comparing the accuracy scores of the models, we select one model to use for prediction.

**Dividing the data into features and the target variable**

****

**Fig. 3.32: Features or Target**

We can create testing and training sets for both dependent and independent variables by importing the **train\_test\_split()** function from the **sklearn.model\_selection** module.



**Fig. 3.33: train\_test\_split() function**

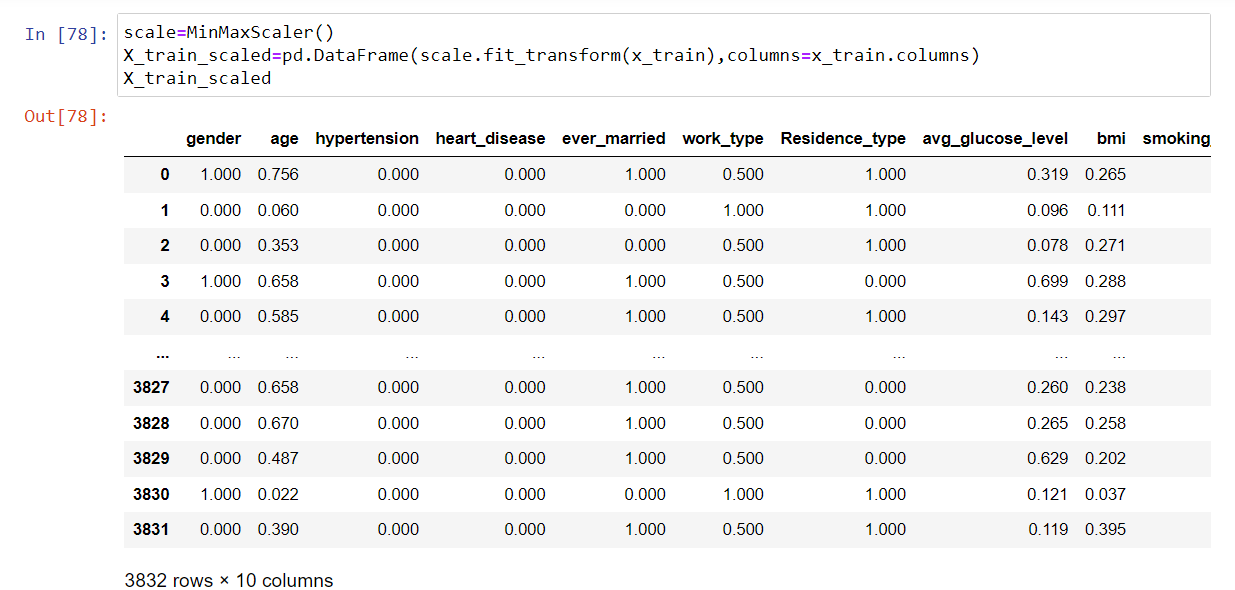
**SCALE THE DATA**

* We are using Min-Max Scaler



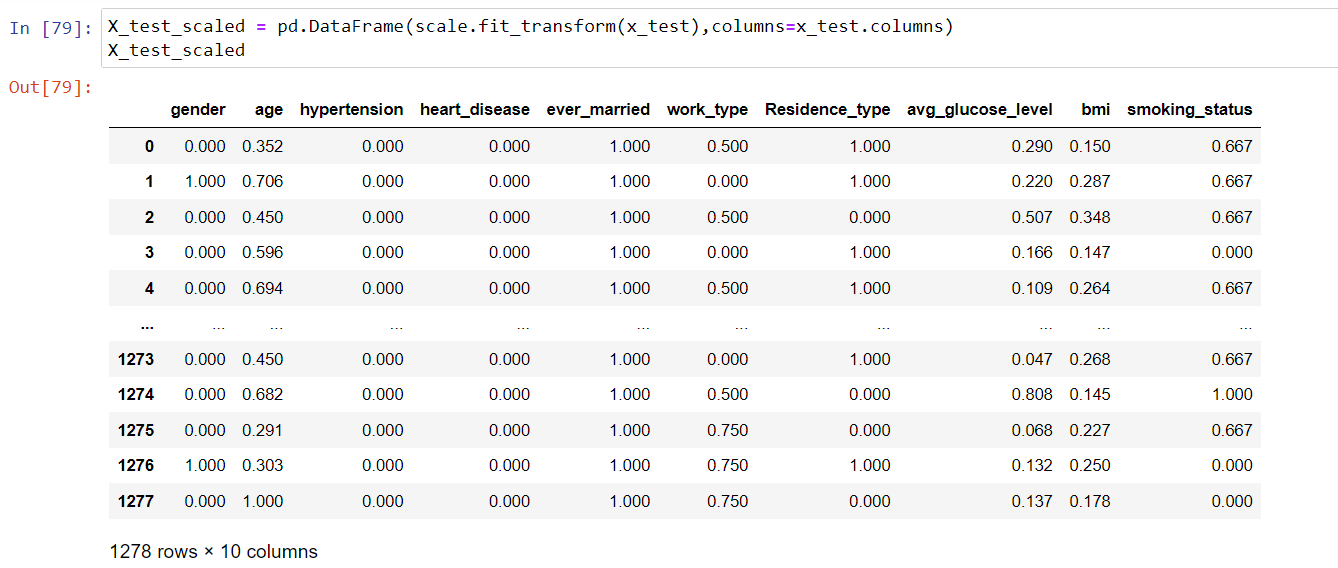
**Fig. 3.34: import MinMaxScaler()**

Dataframe after scaling the train data -



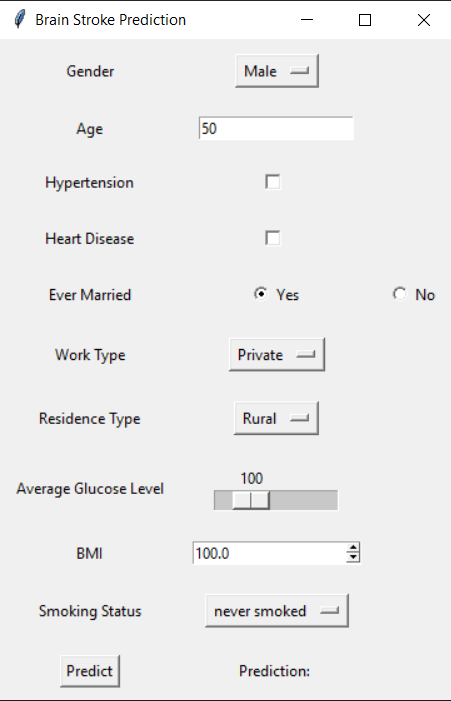
**Fig. 3.35: MinMaxScaler() function on Train data**

Scaling the test dataframe -

****

**\ Fig. 3.36: MinMaxScaler() function on Test data**

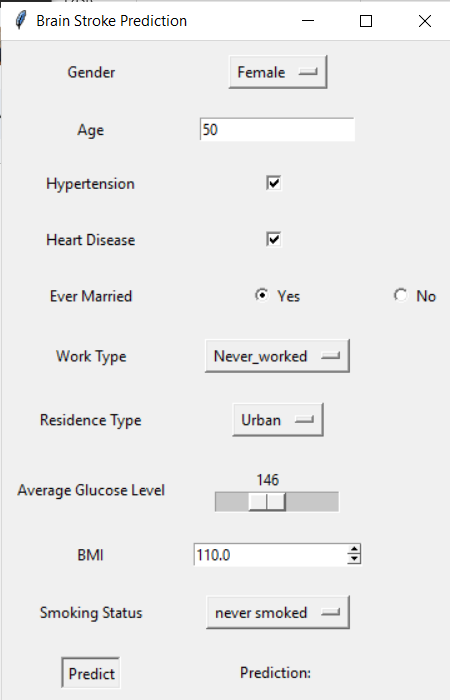
**APPLYING GUI**



**Fig. 3.37: GUI window**

Here is our GUI:

* It has 3 Drop down boxes to choose different categories for each attributes.
* There are also 2 Input boxes for Age and BMI.
* After entering all the details we have to click on Submit button.



**Fig. 3.38: Entering input**

**CHAPTER 4**

**EXPERIMENT AND RESULT ANALYSIS**

We experimented with six machine learning algorithms and selected the LOGISTIC REGRESSIONclassifier based on its accuracy score.

**LOGISTIC REGRESSION :**Logistic regression is a statistical technique that is used to predict binary outcomes.

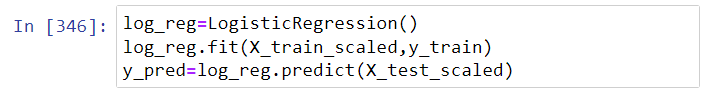
It uses the sigmoid function to generate probability estimates for different classes.



**Fig. 4.1: Importing Logistic Regression**

Ones the above module is imported then instantiate its object

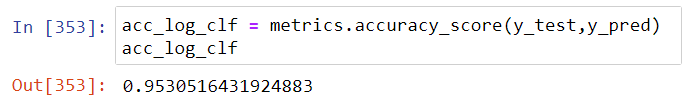
Applying the logistic regression on the test data



**Fig. 4.2: Training Model**

After executing the above code,

Checking the accuracy of our Model:



**Fig. 4.3: Accuracy Percentage**

This results to be **95.3 %**

**DECISION TREE:**

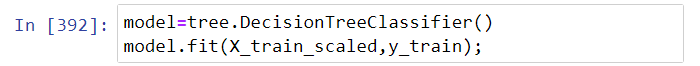
Decision trees are a type of machine learning algorithm that can be used for both classification and regression tasks. Decision trees work by recursively splitting the data into smaller and smaller groups, based on the attribute value that maximizes information gain. This process continues until all of the data points in a group are identical. Once the tree has been created, it can be used to make predictions by following the branches of the tree until a leaf node is reached. The leaf node will contain the predicted value for the data point.



**Fig. 4.4: Importing Decision Tree Classifier**

Ones the above module is imported then instantiate its object

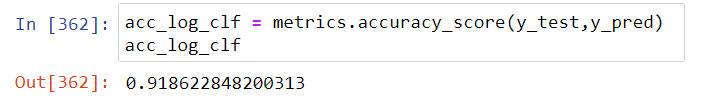
Applying the logistic regression on the test data.



**Fig. 4.5: Training the model**

After executing the above code,

Checking the accuracy of our Model:



**Fig. 4.6: Accuracy Percentage**

This results to be **91.8 %**

**RANDOM FOREST:**

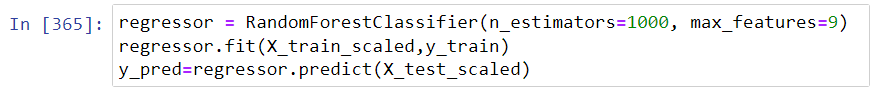
Random forest is an ensemble learning method that uses multiple decision trees to make predictions. Each decision tree is trained on a random subset of the data, and the predictions of the trees are combined using a voting method. This approach helps to reduce overfitting and improve the accuracy of the predictions.



**Fig. 4.7: Importing Random Forest Classifier**

Ones the above module is imported then instantiate its object

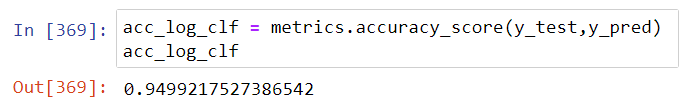
Applying the logistic regression on the test data.



**Fig. 4.8: Training the model**

After executing the above code,

Checking the accuracy of our Model:



**Fig. 4.9: Accuracy Percentage**

This results to be **94.9 %**

**KNN CLASSIFIER:**

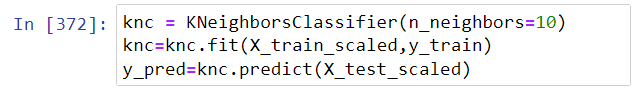
The k-nearest neighbors (KNN) algorithm is a non-parametric machine learning algorithm that can be used for both classification and regression tasks. It works by finding the k most similar data points to a new data point, and then using the labels of those k data points to predict the label of the new data point. The value of k is a hyperparameter that needs to be selected by the user.



**Fig. 4.10: Importing KNeighboursClassifier**

Ones the above module is imported then instantiate its object

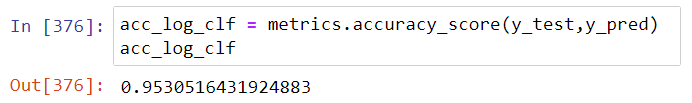
Applying the logistic regression on the test data.



**Fig. 4.11: Training the model**

After executing the above code,

Checking the accuracy of our Model:



**Fig. 4.12: Accuracy Percentage**

This results to be **95.3 %**

**SUPPORT VECTOR MACHINE:**

Support vector machines (SVMs) are a type of supervised machine learning algorithm that can be used for both classification and regression tasks. They work by finding a hyperplane that separates the different classes of data points in a given dataset. The hyperplane is chosen to maximize the margin or the distance between the closest data points of each class.

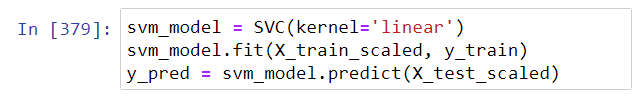
This approach helps to improve the accuracy of the predictions and reduce the risk of overfitting.



**Fig. 4.13: Importing SVM**

Ones the above module is imported then instantiate its object

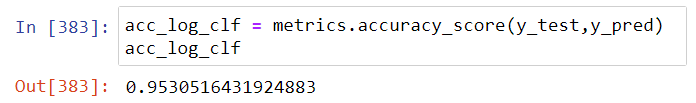
Applying the logistic regression on the test data.



**Fig. 4.14: Training the model**

After executing the above code,

Checking the accuracy of our Model:



**Fig. 4.15: Accuracy Percentage**

This results to be **95.3 %**

**NAIVE BAYES:**

Naive Bayes is a type of probabilistic machine learning algorithm that can be used for both classification and regression tasks. It works by calculating the probability of each class given the input features, and then selecting the class with the highest probability as the predicted class. Naive Bayes is computationally efficient and requires less training data compared to other algorithms.

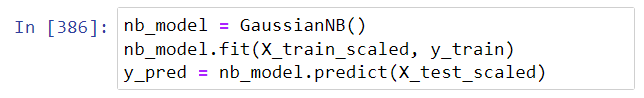
It is called "naive" because it makes the assumption that the features are independent of each other. This is often not the case in real-world data, but it can still be a useful algorithm in many situations.



**Fig. 4.16: Importing Naive Bayes Classifier**

Ones the above module is imported then instantiate its object

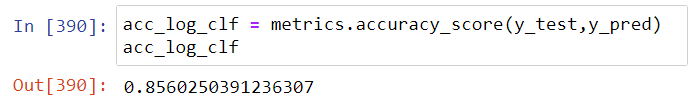
Applying the logistic regression on the test data.



**Fig. 4.17: Training the model**

After executing the above code,

Checking the accuracy of our Model:



**Fig. 4.18: Accuracy Percentage**

This results to be **85.6 %**

**Table 4.19: Accuracy Percentage of different algorithms**

|  |  |
| --- | --- |
| MACHINE LEARNING ALGORITHM | ACCURACY PERCENTAGE |
| Logistic Regression | 95.3 |
| Decision Tree | 91.8 |
| Random Forest | 94.9 |
| KNN Classifier | 95.3 |
| Support Vector Machine | 95.3 |
| Naive Bayes | 85.6 |

It was found that logistic regression the most effective algorithm for the given dataset after applying various machine learning algorithms and analysing the results.

The algorithm was tested by entering random rows of values in a GUI of tkinter and then getting an output. The output was found to be accurate and consistent with the expected results.

**CHAPTER 5**

**CONCLUSION**

* 1. **Discussion**

In conclusion, we were able to achieve an accuracy of 95.3% with our machine learning model. This was accomplished by using a logistic regression algorithm on a dataset of 1278 instances. The model was able to correctly predict 1218 instances of the positive class and 0 instances of the negative class. This suggests that the model is very accurate and has the potential to be used in a variety of applications.

Some of the limitations of the model include the fact that it was trained on a relatively small dataset. This could lead to overfitting, which is when the model learns the patterns in the training data too well and is unable to generalize to new data. To address this limitation, we could collect more data and train the model on a larger dataset.

Overall, we are pleased with the results of our project. We believe that the model has the potential to be used in a variety of applications, and we are excited to continue working on improving it.

* 1. **Future Work**

Future work on our project are as follows:

* + Increase the accuracy.
  + Make the machine learning model much better.
  + Make a web version of the project and easy to use.

**REFERENCES**

**Downloading data set:**

<https://archive.ics.uci.edu/ml/machine-learning-databases/adult/>

<http://abhigyan.dataritz.com/mod/folder/view.php?id=602>

**Loading and analyzing**

<https://pandas.pydata.org/>

<https://numpy.org/>

**Pre-processing**

<https://towardsdatascience.com/data-preprocessing-in-python-b52b652e37d5>

<https://www.geeksforgeeks.org/data-preprocessing-machine-learning-python/>

<https://scikit-learn.org/stable/modules/preprocessing.html>

<https://www.tutorialspoint.com/machine_learning_with_python/machine_learning_with_python_data_preprocessing_analysis_visualization.htm>

<https://www.datacamp.com/courses/preprocessing-for-machine-learning-in-python>

**Data Visualization**

<https://matplotlib.org/tutorials/index.html>

<https://python-graph-gallery.com/seaborn/>

**Machine Learning Algorithm**

<https://www.datacamp.com/community/tutorials/understanding-logistic-regression-python>

<https://www.datacamp.com/community/tutorials/random-forests-classifier-python>

<https://www.datacamp.com/community/tutorials/decision-tree-classification-python>

<https://www.datacamp.com/community/tutorials/k-nearest-neighbor-classification-scikit-learn>

**GUI- Tkinter**

<https://realpython.com/python-gui-tkinter/>

[1] L. Amini, R. Azarpazhouh, M. T. Farzadfar, S. A. Mousavi, F. Jazaieri, F. Khorvash, R. Norouzi, and N. Toghianfar, “Prediction and control of stroke by data mining,” International Journal of Preventive Medicine, vol. 4, no. Suppl 2, pp. S245–249, May 2013.

[2] S.-F. Sung, C.-Y. Hsieh, Y.-H. Kao Yang, H.-J. Lin, C.-H. Chen, Y.- W. Chen, and Y.-H. Hu, “Developing a stroke severity index based on administrative data was feasible using data mining techniques,” Journal of Clinical Epidemiology, vol. 68, no. 11, pp. 1292–1300, Nov. 2015.

[3] P. Govindarajan, R. K. Soundarapandian, A. H. Gandomi, R. Patan, P. Jayaraman, and R. Manikandan, “Classification of stroke disease using machine learning algorithms,” Neural Computing and Applications, vol. 32, no. 3, pp. 817–828, Feb. 2020.

[4] C.-A. Cheng, Y.-C. Lin, and H.-W. Chiu, “Prediction of the prognosis of ischemic stroke patients after intravenous thrombolysis using artificial neural networks,” Studies in Health Technology and Informatics, vol. 202, pp. 115–118, 2014.

[5] S. Cheon, J. Kim, and J. Lim, “The Use of Deep Learning to Predict Stroke Patient Mortality,” International Journal of Environmental Research and Public Health, vol. 16, no. 11, 2019.

[6] M. S. Singh and P. Choudhary, “Stroke prediction using artificial intelligence,” in 2017 8th Annual Industrial Automation and Electromechanical Engineering Conference (IEMECON), Aug. 2017, pp. 158–161.

[7] C. Chin, B. Lin, G. Wu, T. Weng, C. Yang, R. Su, and Y. Pan, “An automated early ischemic stroke detection system using CNN deep learning algorithm,” in 2017 IEEE 8th International Conference on Awareness Science and Technology (iCAST), Nov. 2017, iSSN: 2325- 5994.

[8] M. Monteiro, A. C. Fonseca, A. T. Freitas, T. Pinho e Melo, A. P. Francisco, J. M. Ferro, and A. L. Oliveira, “Using Machine Learning to Improve the Prediction of Functional Outcome in Ischemic Stroke Patients,” IEEE/ACM Transactions on Computational Biology and Bioinformatics, vol. 15, pp. 1953–1959, Nov. 2018.

[9] T. Kansadub, S. Thammaboosadee, S. Kiattisin, and C. Jalayondeja, “Stroke risk prediction model based on demographic data,” in 2015 8th Biomedical Engineering International Conference (BMEiCON), Nov. 2015, pp. 1–3.

[10] S. Y. Adam, A. Yousif, and M. B. Bashir, “Classification of Ischemic Stroke using Machine Learning Algorithms,” International Journal of Computer Applications, vol. 149, no. 10, pp. 26–31, Sep. 2016.