DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

SCHOOL OF ENGINEERING AND TECHNOLOGY

SHARDA UNIVERSITY , GREATER NOIDA

STROKE PREDICTION

A project submitted

In partial fulfillment of the requirements for the degree of

Bachelor of Technology in Computer Science and Engineering

by

AKHIL SIBI (2018008191)

KEVIN SABU (2018004754)

ANIKET SHUKLA (2018006541)

Supervised by

Jyotsna Seth , Asst. Professor (CSE)

May, 2022

CERTIFICATE

This is to Certify that the Report titled "STROKE PREDICTION" submitted by "AKHIL SIBI (2018008191), KEVIN SABU (2018004754), ANIKET SHUKLA(2018006541)" to Sharda University, in fulfillment of the requirements of the degree of "Bachelor of Technology," is the record of Bonafide final year project work carried out by them in the "Department of Computer Science and Engineering, School of Engineering and Technology, Sharda University”. This Project's results/findings have not been submitted in part or in full to any other University/Institute for the award of any other Degree/Diploma.

Signature of the Supervisor
Name: Jyotsna Seth
Designation: Asst. Professor (CSE)

Signature of Head of Department

Name: Prof. (Dr.) Nitin Rakesh

Place:

Date:

Signature of External Examiner

Date:

ACKNOWLEDGEMENT

A Major project provides an excellent opportunity for learning and self-development. We consider ourselves extremely fortunate and privileged to have had so many excellent people guide us through the completion of this project.

First and foremost , we would like to thank Dr. Nitin Rakesh, HOD, CSE who gave
us an opportunity to undertake this project.

My grateful thanks to Jyotsna mam for consistently guiding us throughout our Research work and sharing her wisdom and insights, which enabled us to make the right decisions along the way in implementing our project and writing our journal paper. We do not know where we would have been without her help.

CSE department monitored our progress and arranged all facilities to make our research work easier. We choose this moment to acknowledge their contribution gratefully.

Name and signature of Students:

[illegible]

Akhil Sibi (2018008191)

[illegible]

Kevin Sabu (2018004754)

Aniket.

Aniket Shukla (2018006541)

ABSTRACT

Stroke is a medical emergency that occurs when the blood supply to a portion of our brain is cut off, preventing brain tissue from receiving oxygen. Without oxygen, brain cells and tissue become damaged and die within minutes. The person enters a state of inactivity that necessitates rest and, in the worst-case scenario, coma or death. Stroke occurs in older people, but studies show that the risk increases with age and that it can occur at any age. People who lead a certain lifestyle, such as smoking or drinking, increase their chances of having a stroke. As a result, there is a need for early detection of stroke so that we can diagnose patients with stroke and save their lives. In our research work , we explored various machine learning approaches such as XGBoost, Random Forest, LightGBM, Catboost , Multilayer Perceptron , KNeighbours, and others to train, tune, and test these approaches or models on high feature attributes from the stroke dataset, and we discover that XGBoost is the Best Model for Predicting Stroke, with an Accuracy Score of 94.6183 %.

Index Terms: Stroke , Machine learning , XGBoost , LightGBM , CatBoost

TABLE OF CONTENTS

TITLE PAGE……………………………………………………………………………........1

CERTIFICATE………………………………………………………………………….....2

ACKNOWLEDGEMENT…………………………………………………………………3

ABSTRACT………………………………………………………………………………4

LIST OF SYMBOLS AND ABBREVIATIONS…………………………………………7

LIST OF TABLES AND FIGURES……………………………………………………....8

CHAPTER 1: INTRODUCTION…………………………………………………………10

1.1 Problem definition and motivation ……………………………………………...………..…11

1.2 Project overview……………………………………………………………………...11

1.3 Possible risks……………………………………………………………...……….12

1.4 Software requirement Specifications…………………………………………..…….13

1.5 Hardware Specifications……………………………………………………………..16

1.6 Software Specifications………………………………………………………………17

CHAPTER 2: LITERATURE SURVEY…………………………………………………18

2.1 Existing system…………………………………………………………………..….18

2.2 Basic machine learning concepts……………………………………………...……23

2.3 Classifiers description………………………………………………………...…….24

2.4 Proposed approach……………………………………………………………….…25

2.5 Feasibility Study……………………………………………………………………..27

2.6 Risk management…………………………………………………………..…..28

CHAPTER 3: DESIGN ……………………..……………………………….………………...29

3.1 Flow chart………………………………………………………….………….....29

3.2 Diagrams……………………………………………..……………….……………..30

3.3 SDLC model……………………………………………………………….....…….31

3.4 User interface design………………………………………………………..….…..…..32

3.5 Data integrity and constraints…………………………………………………………32

3.6 Testing process……………………………………………………………………….32

3.7 Verification and validation……………………………………………………………..33

CHAPTER 4: METHODOLOGY……………………………………………………………………34

4.1 Data description…………………………………………………………………34

4.2 Data visualization ………………………………………………………………...35

4.3 Data preprocessing……………………………………………………………...36

4.4 Training………………………………………………………………………….38

4.5 Tuning……………………………………………………………………………40

CHAPTER 5: RESULTS………………..………………………………………………….43

CHAPTER 6 : CONCLUSION……………………………………………………………51

6.1 Performance estimation………………………………………………………..51

6.2 Usability of product / system…………………………………………………...51

6.3 Further Improvement…………………………………………..…………………………52

CHAPTER 7: REFERENCES………………………………………………………….………..53

LIST OF SYMBOLS AND ABBREVIATIONS

Abbreviation	*Definition*
SDLC	Software development life cycle
ML	Machine learning
DL	Deep learning
AI	Artificial intelligence
SVM	Support vector machine
GBM	Gradient boosting model

LIST OF TABLES AND FIGURES

Table 1.1 hardware requirements……………………………………………………….16

Table 1.2 software requirements……………………………………………………..17

Table 2.1 survey of research works on stroke prediction…………………………………18

Table 5.1 comparative analysis of tuned classifiers………………………………………50

Figure 1 proposed stroke prediction system……………………………………..………..25

Figure 2 Flowchart of the Prediction system……………………………………...…..29

Figure 3 Level 0 DFD of Stroke prediction system……………………………...…….31

Figure 4 Import python libraries…………………………………………………………...34

Figure 5 Dataset description……………………………………………………….....34

Figure 6 Countplot of stroke number………………………………………………………35

Figure 7 Label encoders……………………………………………………………………36

Figure 8 Splitting the dataset and and smote implementation……………………………..37

Figure 9 Importing the classifiers for modeling…………………………………………...38

Figure 10 Training the models……………………………………………………………..39

Figure 11 Compiling the results of training and testing……………………………………39

Figure 12 Tuning of the models with the desired parameters………………………………40

Figure 13 Testing tuned XGBoost……………………………………………………....41

Figure 14 Testing tuned Random forest…………………………………………………...41

Figure 15 Testing tuned CatBoost……………………………………………………....41

Figure 16 Testing tuned LightGBM…………………………………………………………41

Figure 17 Pie Chart on number of stroke affected people……………………………..43

Figure 18 Count plot on gender in relation to stroke……………………………………..43

Figure 19 Plot of smoking status……………………………………………………………44

Figure 20 Hypertension in relation to stroke……………………………………………...44

Figure 21 Work type in relation to stroke…………………………………………………45

Figure 22 Heart disease effect on stroke occurrence……………………………………...45

Figure 23 BMI plotting……………………………………………………………………...46

Figure 24 Correlation of features…………………………………………………………46

Figure 25 Comparative Analysis of Classifiers after training and testing…………….….47

Figure 26 Tuned Random forest model performance……………………………………49

Figure 27 Tuned CatBoost performance…………………………………………………49

Figure 28 Tuned LightGBM performance………………………………………………49

Figure 29 Tuned XGBoost performance…………………………………………..….49

CHAPTER 1: INTRODUCTION

Stroke is the second leading cause of death worldwide, accounting for approximately 11% of all deaths, according to the World Health Organization (WHO). It causes difficulty walking and speaking, as well as facial paralysis or numbness, and the patient's life expectancy ranges from 1 to 5 years. Strokes are classified into three types: transient ischemic attack (TIA), ischemic stroke, and hemorrhagic stroke, with ischemic stroke accounting for the majority of strokes (87 percent) [12]. With the ongoing new diseases or viruses such as covid 19, it has been revealed in research articles or journals that stroke can also occur from covid 19 implications and thus it is a deadly medical situation that can impose a serious threat to life if not handled or approached carefully with the dedicated doctors help and guidance and also we should watch for the early symptoms of stroke such as drooping on one side of the face and difficulty speaking etc in older people.

The early prediction of stroke incidence is a critical component of our research work because it allows us to avoid the worst-case scenario and provide proper treatment at the right time. The dataset is open source, taken from Kaggle and attributed to the Author: 'fedesoriano'[11] . As a result, this dataset is used to predict whether a patient is likely to have a stroke based on input parameters such as gender, age, various diseases, and smoking status. Each row of data in the table contains pertinent information about the patient. We balance the dataset by removing duplicate or null values and SMOTE. To deal with categorical values, we use label encoding. We also perform Data Visualization of Feature attributes to gain a better understanding of the relationship between features and stroke attack.

Our Proposed approach is the comparative analysis of various machine learning classifiers that are trained, tested, and tuned with hyperparameters and finding out which is the best classifier. Choosing the classifiers for modeling the prediction system will be based on a review of past research works, which will provide us with the essential information and knowledge to move forward with the project.

1.1 PROJECT DEFINITION AND MOTIVATION

The project's goal is to better understand the risk factors that contribute to stroke and to compare various machine learning approaches for detecting stroke. Stroke is a very serious medical condition that can lead to death of the patients in the worst-case scenarios and in order to avoid such situations , we with our technical prowess have used our knowledge and experience of machine learning and data science to go ahead with this remarkable and ambitious project of the early prediction of stroke in the suspected patients for the benefit of mankind and saving lives based on careful study of the patient's clinical features from the dataset.We investigated and determined the risk factors impacting stroke, displayed great visualizations of these risk factors, which increased our understanding of stroke, and then proceeded with our prediction system modeling.

This project aims to increase awareness and understanding of stroke among older people and younger generations, as well as how new age technologies such as machine learning and data science techniques can aid in the early detection of stroke and save lives by deploying this prediction system in a real-world setting in hospitals for patient diagnosis. Saving lives is our first priority with this research, therefore we followed all of the norms and ethics and rigorously trained and evaluated the models for accurate and exact stroke outcomes that may be used in patient diagnosis.

1.2 PROJECT OVERVIEW

Analysis of stroke occurrence by studying the dataset, identification of stroke risk factors by the data visualization through the python libraries like seaborn and matplotlib and then after , modeling of the prediction system with various machine learning approaches and comparison of performance of models on the different testing parameters .

1.2.1 PROJECT DESCRIPTION

Using Google Colab Notebook, imported dataset is cleaned and preprocessed , visualizations are created for risk factors identification , and classifiers are trained, tuned, and tested on various parameters for modeling of the prediction system. Classifiers are chosen for our research work with the help of an appropriate literature review, which can be seen in the later chapter, and some of them are new and unexplored machine learning classifiers that will enrich our knowledge and provide us with new findings that can or will be shared with the scientific community.

1.2.2 EXPECTED OUTCOME

The results of the stroke prediction research will be a comparative study of the tuned models on the testing dataset, indicating the best model for real-time prediction and engineered for deployment in medical centers based on their accuracy score and ROC - AUC.

1.3 POSSIBLE RISKS

Because research is difficult to conduct and implement, one person or a team member may encounter various risks associated with the project, which may hinder or reduce the quality of the work done and prevent the work from reaching its full potential and extent with new observations and possibilities. These risks can stem from a variety of events encountered during the course of our project, such as:

1.3.1 CLASS IMBALANCE

Imbalance between the majority and minority classes in the dataset can lead to lower accuracy and performance, and because the stroke dataset is imbalanced in nature, it is balanced during the preprocessing process using SMOTE, an oversampling technique used to synthetically increase the number of minority classes to balance the dataset. Without balancing the dataset prior to training and testing the models, we risk having poor performance and accuracy, which will impair the quality and enrichment of the findings.

1.3.2 LOWER ACCURACY SCORE

While putting the trained models to the test. Sometimes it is discovered that they have lesser accuracy, which can be damaging to the progress of the research job, thus we perform hyperparameter tuning of every model produced by their desired parameters such as learning rate, random state, eval metric, and n estimators, among others. Tuning is an important element of our study, and we do it carefully by picking the desired parameters while taking into account the impact on the performance of the classifiers in the model.

1.4 SOFTWARE REQUIREMENTS SPECIFICATIONS

Software requirements specification (SRS) is a Technical document that specifies the functionality of a piece of software or a product to stakeholders and customers. It is completed prior to the design process, and the document discusses the product's functional and nonfunctional needs, who is targeted for the product launch based on a market survey, and how the product will contribute to the company's growth and profitability. While developing the prediction system, we considered the various types of requirements, such as functional and nonfunctional requirements, as well as how they interact with one another, in order to create a personalized and realistic prediction system that will accurately predict the chances of stroke when given patient data. It is written in simple layman language and is concise, consistent, unambiguous, and traceable in nature, so that stakeholders and the developer team can refer to the SRS document at all stages of the software development life cycle to ensure that everything is going as planned and that the product or software's quality and standards are maintained and SRS is critical in the early stages of project development because they guide the company or team members to make the right decisions throughout the development of the product or software. Similarly, we have set our own requirements for our research work and followed them consistently so that we can have the same good quality and fairness of the work ensured and maintained throughout the work. They are listed below.:

1.4.1 FUNCTIONAL REQUIREMENTS

Functional requirements are the ones that specify how the system should behave, function, and perform in order to achieve the desired results. They are the crucial element in a SRS document because they tell the stakeholders what the product is supposed to do in terms of functionality and how the proposed product is different from similar products in the market in terms of novelty and market disruption. Functional requirements also assist the developer team and the manager in referring to these requirements on a regular basis to ensure that they are building the product correctly and in the right direction so that there are no problems, bugs, or errors in the built software or product later on in the SDLC and in our research work these are the functional requirements that we kept in mind while developing the prediction system:

1.4.1.1 ACCURACY SCORE

The Accuracy score is the most crucial component of our prediction system, and it must be exact and correct for doctors to make the correct diagnosis based on the results. To achieve the desired accuracy, machine learning classifiers must be properly trained, tuned, and tested on a balanced dataset, and thus for the integrity of our research work, the prediction system must have an accuracy of more than 90% while making predictions on the testing dataset or the inputs provided, because accuracy matters a lot in the field of medicine or any other, and so good accuracy ensures the software's reliability to make the right predictions.

1.4.1.2 RESULTS

Prediction systems should be able to provide accurate results in a short period of time, in terms of seconds, so that doctors can make diagnoses based on the prediction system's test results. As a result, machine learning models should be rigorously trained, tested, and fine-tuned so that the prediction system can provide accurate results in real-world situations.

1.4.2 NORMAL REQUIREMENTS

These are the consumer's requirements, as well as their expectations of how the product will benefit them with their day-to-day medical requirements and as a result they are as follows:

1.4.2.1 FASTER RESULTS WITH GREATER ACCURACY

Machine learning models are extensively trained and tested before being tweaked with the appropriate set of parameters to deliver precise accuracy scores on the testing dataset or inputs provided, and they are ready for real-world situations and deployment.

1.4.2.2 TRUSTED GENUINE RESULTS

There is no manipulation of the prediction system's results based on the patient's data for incorrect and misleading claims, and entire transparency and regulations are kept and ensured so that the research effort can serve mankind and have a healthy and transformational impact on people's lives. So, with the consumer's needs in mind, we carried out the project in a fair and transparent manner.

1.4.3 NON FUNCTIONAL REQUIREMENTS

Non-functional requirements are the ones that inform us about the system's attributes and design constraints, such as availability, security, reliability, and usability, among others. Non-functional requirements ensure that the system is capable of carrying out the functional requirements in real-world conditions in terms of system design and capacity, and so they are equally as significant as function requirements and should be kept in mind in the research effort, as follows:

1.4.3.1 RELIABILITY

There should be no manipulation with test results within the prediction system, and the test findings on stroke should always be accurate to the point so that the doctor can make the correct diagnosis. They should be trustworthy enough that the patient can make the necessary changes to their lifestyle and health on the doctor's advice if they are at danger of having a stroke in the future.

1.4.3.2 PERFORMANCE REQUIREMENTS

When patient information is used as an input to the prediction system, test results should be provided in seconds; thus, carefully training and tuning the classifiers is critical, taking into account the speed and execution of the results if the system is deployed in a real-world setting, such as a hospital. In this fast-paced world and in the medical profession, performance matters, and failure to load the data or lag in the prediction system will only slow the diagnosis of patients who are at risk of having a stroke in the future.

1.4.3.3 AVAILABILITY

When a prediction system is used in real-world situations in hospitals for the diagnosis of stroke patients, it should be operational 24 hours a day, seven days a week so that doctors can use it to check the results of early stroke patients and decide whether the patients require medical attention or need to stay in the hospital for further observation.

1.4.3.4 ABILITY OF LEARNING

Machine learning models should be trained on a large dataset of stroke patients to aid in the identification of risk variables associated with stroke. Datasets should be gathered from credible sources, and models should be properly tuned with the appropriate set of parameters so that they work well on testing data or patient input.

1.5. HARDWARE SPECIFICATIONS

Table 1.1 hardware requirements.

Components	Specifications
Operating System	Windows 11
Processor	Intel i5 8th generation processor
Memory (RAM)	8 GB RAM , 1TB HDD
Graphics	Nvidia Geforce MX 150 graphics card.

Our study hardware is a simple, efficient laptop with an i5 8th gen processor, a minimal graphic card, and 8gb ram. Good internet access ensured the seamless operation of the study for research and training of the models in Google colab notebook, and university access to extra hardware resources such as computer labs, etc. facilitated the growth of our work in the field of stroke.

1.6. SOFTWARE SPECIFICATIONS

Table 1.2 software requirements.

Google Colab Notebook
Python programming language
Import machine learning classifier’s through scikit-learn or else pip install
Data visualization libraries like matplotlib , seaborn etc
Hyperparameter tuning through GridSearch

Instead of Jupyter notebook, we chose Google Colab notebook, which is cloud-based and provides free access to powerful GPUs in the cloud. The notebook also comes preinstalled with the necessary python libraries for machine learning, such as scikit learn, pandas, numpy, matplotlib, and so on, so we don't have to worry about installing them via pip.

CHAPTER 2: LITERATURE SURVEY

A literature survey entails conducting a thorough analysis of previous research works on your chosen topic by various authors and studying their approach, advantages, and limitations in order to come up with your own novelty for the topic and do something unique that can be shared with the scientific community.

2.1 EXISTING SYSTEM

We did a thorough analysis of past studies in stroke prediction from a range of well-known sources, including IEEE, ScienceDirect, Research Gate, Scopus, and others, and compared them based on findings, algorithms used with accuracy, benefits, and limitations. The study helped us determine the most common risk factors for stroke occurrence. These past research works inspired us to create our prediction system, and the approaches used in these previous works aided us in investigating alternative powerful machine learning algorithms such as XGBoost, LightGBM, and CatBoost, among others.

Table 2.1 survey of research works on stroke prediction.

AUTHORS	SOURCE	FINDINGS	ALGORITHMS	ADVANTAGES	LIMITATIONS
M. S. Singh and P. Choudhary [1]	IEEE Xplore	A comparison of their proposed AI-based approach with other methods trained on the CHS dataset.	The proposed Classification model is built with a Decision Tree for feature selection, a PCA for dimensionality reduction, and a back propagation neural network for Classification, and it predicts Stroke with an accuracy of 97.70%.	The research work is straightforward, highlighting the steps in developing their proposed approach, which can be used to train other high-level machine learning models.	The research work inspires our own work, but the only limitation is that alternative higher level deep learning techniques are not explored, which could have further enriched their observations with new insights.

A. Sudha et al. [2]	International journal of Computer applications	A comparison of three approaches: Decision Tree, Naive Bayes, and Neural Network trained and tested on a dataset of 909 patient records with eight important patient data attributes.	In comparison to decision trees and Naive Bayes classifiers, neural networks are more accurate.	The steps involved in the development of a prediction model were clearly explained in the paper, from data collection to processing to result analysis. They have clearly demonstrated the logic behind the selection of these three models, as well as the factors influencing stroke, such as hypertension, weight, and headaches, among others.	Alternative techniques should have been explored in order to strengthen and cover all aspects of the research.
C. H.Lin et al. [3]	Science Direct	A comparison of SVM, Random Forest, artificial neural network, and hybrid ann trained on Taiwan Stroke Registry (TSR) data from stroke patients since 2006.	ML techniques achieve 0.94 AUC in both ischemic and hemorrhagic stroke. The prediction ability was improved to 0.97 AUC by including follow-up data.	The study also identifies key features from ischemic and hemorrhagic stroke datasets that contribute to stroke incidence.	The majority of prediction errors were found in patients with more severe strokes, according to error analysis.
T. Liu et al. [4]	Science Direct	On a medical dataset containing 43,400 records of potential patients, an automated hyperparameter optimization approach is used to develop the model, and random forest regression is used to handle missing values.	Hybrid machine learning approach - 71.4%	The approach proposed in this paper demonstrated a significant reduction in the misdiagnosis rate for stroke prediction.	One limitation is class imbalance in the dataset, and because the resulting accuracy is low, alternative machine learning classifiers should have been investigated to further improve the accuracy that can lead to precise prediction of stroke in patients.

H. Ahmed et al. [5]	Research gate	A comparison of machine learning models built on Apache Spark with the MLib library	Logistic Regression - 76% Random Forest - 90% Decision Tree - 79% Linear SVM - 76%	The Apache Spark APIs made dataset balancing and model formation extremely simple and efficient.	There were only a few machine learning approaches used.
C. S. Nwosu et al. [6]	Research gate	Over 1000 experiments, the classification accuracy of neural networks, decision trees, and random forests was compared on a dataset of electronic health records.	Decision Tree - 74.31% Random Forest 74.53% Multilayer perceptron 75.02%	The research work is impressive, and the dataset contains 29072 patient's health records with 12 attributes, from which they chose 10 for training the models.	The accuracy is insufficient, and alternative machine learning classifiers should have been investigated to improve overall accuracy.
P.Chantamit-o-pas et al. [7]	Researchgate	A comparison of Naive Bayes, SVM, and feedforward ann on the UCI Heart dataset for stroke prediction.	Mean Values are Naive Bayes : 49% SVM :47.78% Deep learning: 36.73%	According to research, the deep learning approach uses data obtained from patient health records to aid in the discovery of previously unknown knowledge, which is critical in diagnosis and decision making.	While developing the model, alternative machine and deep learning approaches should have been investigated.
Y. Wu and Y . Fang [8]	IJERPH	On the basis of data from an elderly Chinese population, a comparison of machine learning approaches such as Random forest, svm, and logistic regression was performed, as well as the identification of	AUC Regularized Logistic Regression - 0.72 Random Forest - 0.78	The comparison of model performance on imbalanced and balanced datasets suggests that the balanced dataset is important in model accuracy score and that using SMOTE technique helps in the class imbalance of the	To further cover all aspects of the research, alternative machine learning approaches should have been investigated.

		risk factors influencing stroke in older Chinese people.		dataset.	
K. A. Mahesh et al. [9]	Research gate	Comparison of decision trees, naive bayes, and artificial neural networks based on their AUC - ROC curves.	During ROC comparative analysis, it is discovered that they have similar AUC.	The great thing about their work is that they also built a simple GUI and integrated the models so that when the user fills in the new inputs in the GUI, it predicts the stroke accuracy and provides personalized warnings to optimize their lifestyle.	Using the AUC-ROC curve to compare machine learning models may yield unreliable results, so we can consider other performance indicators such as Accuracy Score, Precision, Recall, and F1 Score, among others.
S. N. Min et al. [10]	European Neurology Journal	Using a large pool of enrollee data from the Korea National Health Insurance Service database (NHIS), the research team was able to conduct an extensive study on risk factors affecting stroke and use a machine learning approach to train on the dataset features and provide real-time predictions on stroke incidence..	Logistic Regression was developed and trained on the risk factors and it was found that the model was able to distinguish between normal subjects and stroke patients in 65% of cases.	Given access to a large pool of data, they were able to conduct a comprehensive study on stroke risk factors such as hypertension, heart disease, and lifestyle factors, among others, and they claimed that the model developed can be used in clinical settings to estimate the probability of stroke in a year	The only limitation is that additional algorithms should have been investigated for developing the model in the form of a comparative analysis study because logistic regression performs poorly in comparison to powerful classifiers such as Random Forest, XGBoost, LightGBM, and so on.

Amini L et al. [13]	International journal of preventive medicine	A comparison of the C4.5 decision tree algorithm and K Nearest neighbor on a dataset of 807 patients from two reputable hospitals in Iran, and these algorithms are trained on risk factors such as hyperlipidemia, diabetes, and smoking, among others.	C4.5 - 95.42% Knn - 94.18%	Extensive investigation of stroke risk factors in Iranian patients, as well as the significance of data mining mechanisms using Weka software.	For new observations such as increased accuracy score and performance, large amounts of data must be collected, and alternative machine algorithms must be investigated.
J. N. Heo et al.[16]	AHA Journals	On the basis of the ASTRAL score for ischemic stroke, a comparison of deep neural networks, random forests, and logistic regression was performed on a cohort of 2604 patients.	AUC NN - 0.888 RF-0.857 LR - 0.849	The AUC of NN is higher than its ASTRAL score, whereas the AUC of RF and LR is lower than their respective ASTRAL scores, indicating that deep learning neural networks are suitable for predicting complex patient outcomes.	Because it was a single-center study with limited availability of variables from other sources, model predictions will differ each time it is trained and tested.
E. M. Alanazi et al. [17]	JMIR Publications	On the NHANES dataset with three distinct types of selection methods, a comparison of J48, random forest, naive bayes, and BayesNet algorithms was performed.	Random forest achieves a 96 percent accuracy in data resampling selection, followed by decision tree at 93 percent, BayesNet at 87 percent, and Naive Bayes at 82 percent.	Identification of stroke risk factors such as age, gender, creatinine, and white blood cell count, as well as a comparison of the three selection approaches, with the algorithms trained on the data resampling selection method achieving a greater accuracy.	Alternative machine learning algorithms should have been investigated, and due to the poor quality of the NHANES data, it is difficult to distinguish between the types of stroke in patients: ischemic or hemorrhagic stroke.

B. Letham et al.[18]	The Annals of Applied Statistics	On the basis of their performance on the MDCD database, the bayesian rule list, CHADS2 score, and CHA_2DS_2-VASc score were compared.	AUC BRL point - 75.6%, $CHADS_2$ - 72.1%, CHA_2DS_2-VASc -67.7%	Bayesian rule list method described in this paper seeks the "sweet spot" between accuracy and compatibility and has the shortest training time due to its speed. The study's goal was to make the results more understandable to domain experts and to make the inner workings of the models used easier to understand.	The accuracies are much lower, and they haven't even crossed the 80% threshold and other models should be explored to advance the research.

2.2 BASIC MACHINE LEARNING CONCEPTS

We will briefly discuss the machine learning concepts that will be employed in our proposed method or system so that one has a clear knowledge of what is going on in our prediction system and is familiar with the technical jargon.

2.2.1 BAGGING

It is an ensemble learning method in which the predictions of various decision trees are combined using a majority voting principle to give a final prediction that is accurate in nature.

2.2.2 BOOSTING

It is an ensemble learning method in which weak classifiers, such as decision trees, are built sequentially by reducing the error of previously added models to create a strong classifier and thus increasing the overall performance and accuracy of the predictions.

2.2.3 GRADIENT BOOSTING

It is similar to boosting, but it employs Gradient descent to minimize the value of the cost function of previously added models.

2.2.4 DECISION TREE

Decision tree is a supervised ML technique in which there is a graphical flow chart representation of possible solutions to a decision based on certain conditions and consisting of two components: Decision node and leaves. Decision node represents the question or a problem and leaf node denotes the possible solutions.

2.3 CLASSIFIER'S DESCRIPTION

We will try to explain in layman's term the background of the classifier's used in the research work :

2.3.1 XGBOOST

It is a superior variant of gradient boosting algorithm which has the following features: Regularization , Parallelization and tree pruning that extremely boost the performance of the model on the structured dataset and give greater accuracy.

2.3.2 RANDOM FOREST

Random forest is a Bagging-based classifier that consists of a large number of decision trees that are trained on the dataset and their results are aggregated to provide the final prediction.

2.3.3 LIGHTGBM

It is a variant of the gradient boosting algorithm developed by Microsoft in which decision trees grow leaf wise and it employs Histogram based splitting, GOSS, and EFB and unlike Xgboost , it can handle Categorical data and we don't have to do encodings.

2.3.4. CATBOOST

CatBoost, like LightGBM, provides accurate predictions with default parameters, is fast and scalable, and, unlike Xgboost, can handle categorical values.

2.3.5. ADABOOST

It is the first boosting algorithm developed, in which Decision stumps (half trees) are added sequentially by updating the weights of previously added models to create a strong learner.

2.3.6 SUPPORT VECTOR MACHINE

SVM is a supervised machine learning algorithm that divides the n-dimensional feature space by a decision boundary called a hyperplane that is created using support vector points. When a new data point is added to the model, SVM can easily categorize it using a hyperplane.

2.3.7 KNEIGHBOURS

Knn is a simple supervised ML technique in which the class of a data point is determined by the neighboring points through majority voting principle technique. It is non parametric and we can set the number of neighbors for the data point by controlling the value of k parameter (k = 1 to 5).

2.3.8 MULTILAYER PERCEPTRON

It's a simple neural network with three layers: input, hidden, and output. It's feedforward in nature, and it's trained using backpropagation. We can simply import it from sklearn.neural_network library and use it according to our purpose.

2.3.9. LOGISTIC REGRESSION

It's a simple supervised learning strategy for predicting the value of a categorical or discrete variable with values ranging from 0 to 1 or in the form of yes or no. Its activation function is the sigmoid function, and it is used to solve multi classification problems.

2.3.10. BERNOULLINB AND GAUSSIANNB

Both models are based on the naive bayes classifier, with the gaussian nb classifier being used when the data is continuous and follows a gaussian distribution, and the bernoulli nb classifier being used when the data is boolean and follows a bernoulli distribution.

2.4 PROPOSED APPROACH

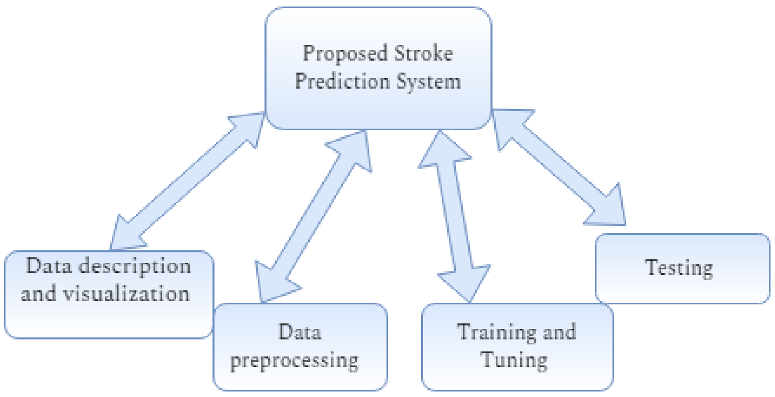

Proposed Stroke Prediction System
Data description and visualization
Data preprocessing
Training and Tuning
Testing

Figure 1. proposed stroke prediction system

Our proposed method is to do a comparison study of several machine learning classifiers that have been trained, tested, and tweaked with hyperparameters to decide which classifier is the best. The identification of risk factors from datasets utilizing visualizations and training machine learning algorithms on these characteristics is crucial for successful prediction and understanding of stroke incidence. Our proposed approach ensures that the research activity coincides with novelty and that the end result is reached so that we may share the results with the scientific community in the form of a research paper and deploy the prediction system in real circumstances for the doctors to diagnose stroke patients.

2.4.1. DATA COLLECTION

A comprehensive dataset is imported from a reputed data source eg a website and used in our research work that contains the stroke patient's clinical features that helps in the identification of risk factors of stroke occurrence and helps us in visualizing the key features of stroke through plotting of count plots , pie chart , histogram etc using python libraries like matplotlib , seaborn etc.

2.4.2. DATA PREPROCESSING

Crucial element of our research work is the preprocessing of the dataset by removing the duplicate values of the features , label encoders to handle categorical values and smote to balance the dataset by oversampling the minority classes. Preprocessing ensures the models built will perform very well with impressive results and no errors or bugs can cause the models to falsely predict or make wrong predictions.

2.4.3. TRAINING AND TUNING

Importing the machine learning classifiers for training using scikit learn python library and training them on the training dataset and after training , further tuning them using the desired parameters to improve the accuracy and performance of the models. Tuning is an integral part of machine learning models being built so that even though the performance of models is poor while training and testing earlier , tuning with the desired parameters of each one of the classifiers will ensure the increase in training accuracy with the best parameter and after , testing will give impressive results from the earlier ones and so tuning of the models in the prediction with right parameters is a very important part of our research work.

2.4.4. TESTING

Testing the tuned models on the testing data to check their accuracy score and ROC AUC score and find out which is the best model for stroke prediction and so this is the last deciding step in the prediction system

2.5 FEASIBILITY STUDY

The Feasibility study was completed very early in the project's development in order to assess the practicability of the proposed system and whether it would be successful in execution and delivery or not. We have covered all the aspects in the study ranging from Technical and Financial areas and have come to the conclusion that the research work is feasible to conduct and it will be a great progress in dealing with stroke and prediction of the stroke through early onset symptoms to save people’s lives.

2.5.1 TECHNICAL FEASIBILITY

Because we are using google colab notebook, which is cloud-based, we have free access to gpu in cloud for model training and we don't have to download additional software because colab notebook already comes with python, tensorflow, pytorch, and so on. To sign in and access the colab notebooks, we need a working laptop or computer, an internet connection, and a google account.So from a technical standpoint , we have the required hardware and software requirements that will aid in the progress of our research work and will lead to a successful completion of our prediction system.

2.5.2 FINANCIAL FEASIBILITY

As seen in the Technical feasibility, since the work is done in a free tier cloud notebook, we only need to budget for a working internet connection and a simple , efficient working laptop or pc that will aid in the making of our prediction system and also we have free access to the computer lab of the university that gives us an upper hand in the research work and save us great time and resource

2.6 RISK MANAGEMENT

Risks can arise while making our prediction system that can impact the performance of the models. These risks can range from low accuracy and ROC AUC scores etc and so to manage these risks , we have to balance the dataset in preprocessing using smote , hyperparameter tuning is essential during training so that to drastically increase the accuracy of the models on the testing dataset.

CHAPTER 3: DESIGN

Designing the prediction system is a key component of the planning phase prior to implementation and coding since it allows us to envision how the system works, its behavior, and the actions required. We also mention and talk about the SDLC in our research work and testing of the prediction system.

3.1 FLOW CHART

Using an online diagram software called diagrams.net , we have designed a basic layout of our step by step making of the prediction system that showcases the methodology followed

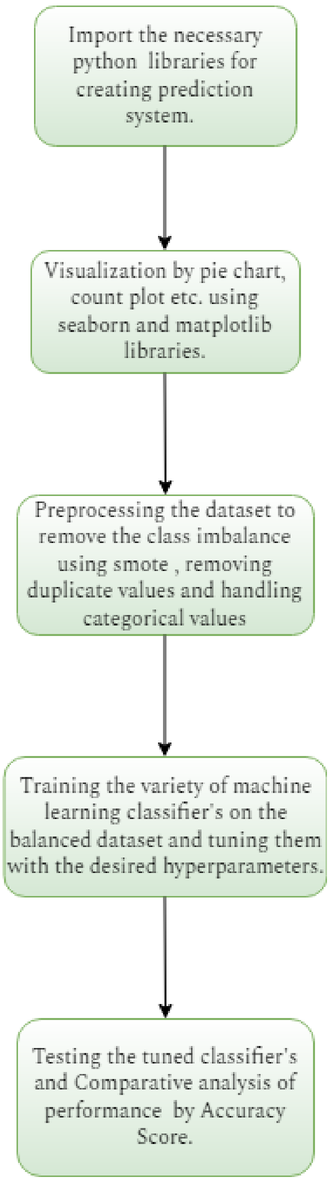

Import the necessary python libraries for creating prediction system.
Visualization by pie chart, count plot etc. using seaborn and matplotlib libraries.
Preprocessing the dataset to remove the class imbalance using smote , removing duplicate values and handling categorical values
Training the variety of machine learning classifier's on the balanced dataset and tuning them with the desired hyperparameters.
Testing the tuned classifier's and Comparative analysis of performance by Accuracy Score.

Figure 2. Flowchart of the Prediction system

Figure 2 represents the flowchart which depicts the flow of the project from start to finish in a sequential manner and everything is to be completed step by step so that we can get the desired result. These steps are clear, well thought out and concise enough that we have built the prediction system from scratch by following the methodology highlighted in the flowchart that ensured the positive success rate of the research work and ensured insightful , rich observations from the study that led to writing the research paper for a reputed journal.

3.2 DIAGRAMS

Diagrams aided in visualizing the prediction system and the outcome of our study endeavor, resulting in:

3.2.1 ER AND CLASS DIAGRAMS

Our project work is purely research oriented in nature, with a comparative analysis of a variety of machine learning classifiers on a stroke dataset imported from a reputable data science website, and thus there is no database on our project for the time being and so there is no ER diagram for the same reason and for the class diagrams , in machine learning oriented research work there is no classes or objects , we import the necessary python libraries and models and then go on with making our classification , regression or prediction model for the research purpose and outcome.

3.2.2 DATA FLOW DIAGRAM

Data flow diagrams (DFDs) are graphical representations of system level designs that show the ins and outs of the flow of information or data and how the results will be generated from the inputs provided to the system. DFDs are very useful in the planning phase of SDLC, as well as in the requirement gathering phase, where visualization of the system's operation at various stages of DFD can guide the development of the product or software.

3.2.2.1 DFD 0

DFD level 0 is the typical context diagram of the entire system that represents the software or a product and depicts the straightforward flow of information from input to output via a standard process. Therefore we have shown a simple dfd 0 level diagram for our stroke prediction system that can help in understanding the inner workings of the system.

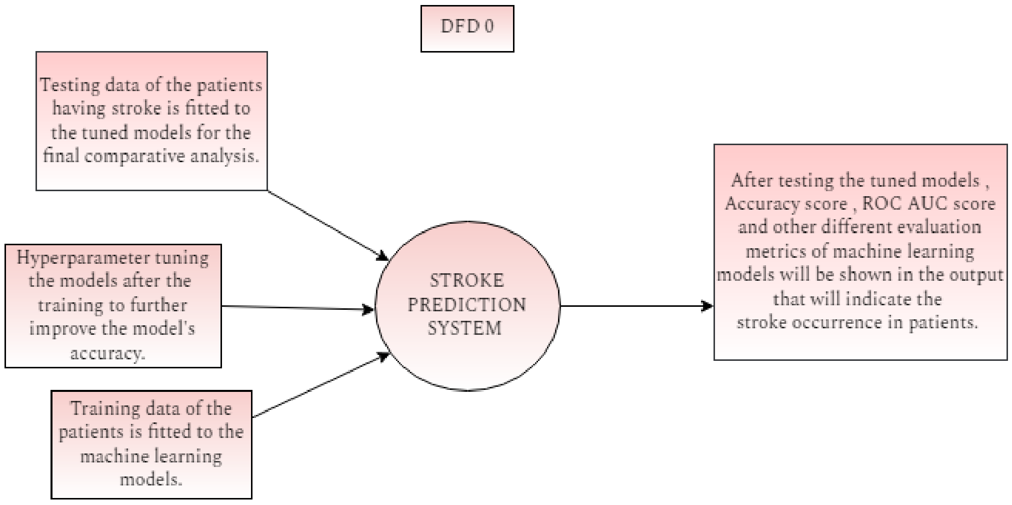

DFD 0
Testing data of the patients having stroke is fitted to the tuned models for the final comparative analysis.
Hyperparameter tuning the models after the training to further improve the model's accuracy.
Training data of the patients is fitted to the machine learning models.
STROKE PREDICTION SYSTEM
After testing the tuned models , Accuracy score , ROC AUC score and other different evaluation metrics of machine learning models will be shown in the output that will indicate the stroke occurrence in patients.

Figure 3. Level 0 DFD of Stroke prediction system

In the figure 3 , we can see the detailed flow of input and output at a basic level of our prediction system and it helps in guiding the development of software or product.

3.3 SDLC model

For project management, we used Agile approach in which we divided the creation of the prediction system into several iterations , as shown in the flowchart, resulting in improved collaboration and project execution. Its advantages include efficient project management , promotes greater team collaboration and ensures work life balance and productivity of the employees.Agile model is very much useful and better than the traditional waterfall , prototype and rad models because agile can be applied for big enterprise projects keeping in mind the resources , time , and people's well being and that’s why agile is used in big technology companies like Facebook , Apple , TCS , HCL and many others. Therefore for our own research work , we felt the need to use Agile for proactive research work progress, scheduling and managing the work between the teammates and reaching the set goals on time.

3.3.1 Why is the approach used ?

We can perform increments or sprints by agile on a weekly basis that ensure the smooth delivery of the project execution from requirement planning , design , implementation , testing and review etc. This ensures the rich quality of the project delivery having less bugs and that meets the customer and stakeholders expectations.

Agile methodology also has variations to it which are catered to different styles of Project management in various companies : Scrum , Kanban and Extreme Programming (XP) etc.

3.4 USER INTERFACE DESIGN

Actually, there is no user interface for the time being because the work is purely research-oriented, with the emphasis on training, tuning, and testing a wide range of machine learning models, and finally compiling the results in the form of a comparative analysis that will provide us with the final results of the models in terms of accuracy score and other evaluation metrics on the stroke dataset. In terms of future work, we can choose for an embedded or a computer system user interface that will assist doctors in typing in patient features into the machine, and the output will be generated from the best model that we select from the comparison analysis.

3.5 DATA INTEGRITY & CONSTRAINTS

Our project work is purely research oriented in nature, with a comparative analysis of a variety of machine learning classifiers on a stroke dataset imported from a reputable data science website, and thus there is no database on our project for the time being, but as previously stated, if we want to deploy the model in an embedded or computer system, we will definitely go for the database that will store the information of the patients in real time and the database will follow the data integrity and constraints imposed on it throughout so that we can maintain the ACID properties.

3.6 TESTING PROCESS

Testing the prediction system is an important component of our research since it ensures that the prediction is running properly and producing amazing results based on the testing dataset or patient input provided into the system.

3.6.1 SYSTEM TESTING

After training and tuning the machine learning models, they are rigorously tested on the testing dataset on the basis of accuracy, ROC AUC score, and other evaluation metrics that aid in the comparative analysis of the models and allow us to determine which model or approach is the best for stroke prediction.

3.7 VERIFICATION AND VALIDATION

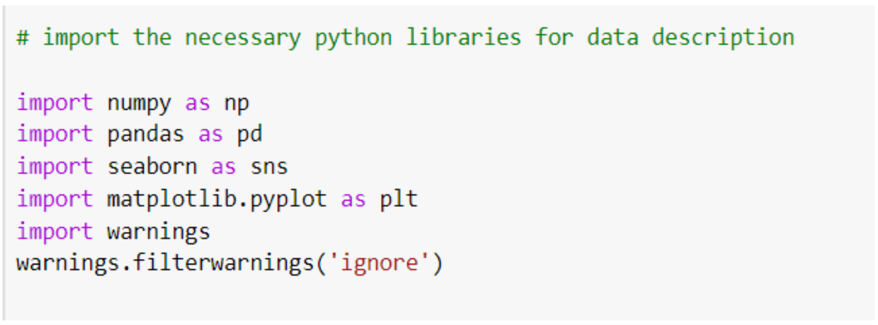
We have verified the prediction system at all phases of our SDLC by following to the SRS document and DFD's, and we have maintained the high quality of our research effort by sticking to the rules, ethics, and transparency. The prediction system will function as expected because there will be rigorous training and tuning of a variety of machine learning models that produce impressive results on the testing dataset in terms of accuracy score and other evaluation metrics that will provide an idea of stroke occurrence in early suspected patients.

Validation ensures that the product or software is the right fit for the company, research work, or society and will have a long-term impact on people's lives, as our vision for the stroke prediction system has been mentioned in the earlier chapters of the project report in Introduction and Literature review, and we have also showcased and demonstrated our prediction system to the college faculty members who were impressed with our results.

CHAPTER 4: METHODOLOGY

Following the design of the prediction system, we proceeded with the implementation by following the research stages outlined in order to reach our study objective of comparison analysis.

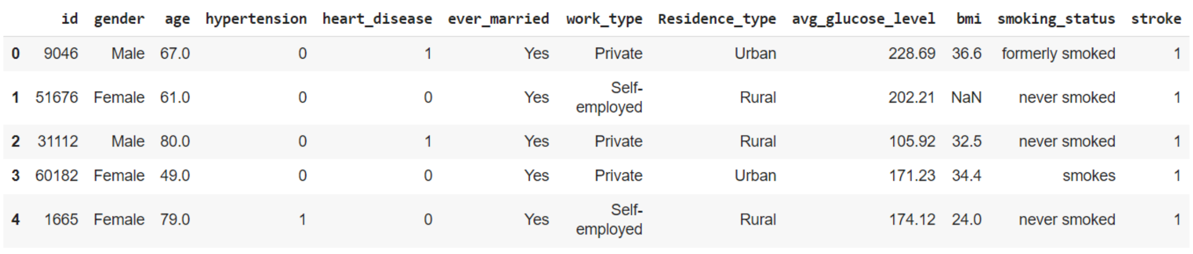
4.1. DATA DESCRIPTION



```
# import the necessary python libraries for data description

import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import warnings
warnings.filterwarnings('ignore')
```

Figure 4. Import python libraries

As seen in the figure 4 ,we import the necessary python libraries for building our prediction system like Pandas , matplotlib etc. for reading and analyzing the dataset and visualizing the key risk factors affecting stroke occurrence. These libraries are very essential for data description and visualization purposes and they provide the base of the prediction system. For our modeling, we use the Stroke Prediction Dataset from Kaggle [11] , which contains 5110 Patient Records with feature characteristics related to the patient's lifestyle, such as BMI, Heart disease, hypertension, smoking, gender, and so on.This dataset is rich in clinical aspects of patients, which, when carefully examined, aids in the identification of risk factors for stroke and allows us to analyze stroke occurrence. The dataset is uneven and contains duplicate values, which we will clean and balance during the data preprocessing stage before training the models.



	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

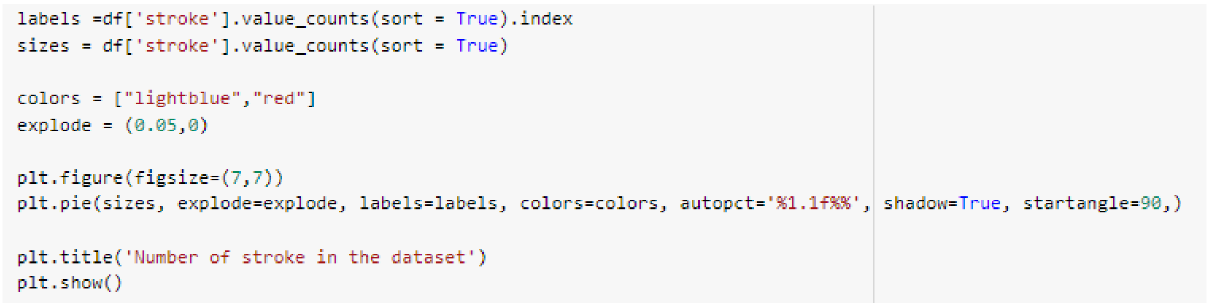
Figure 5. Dataset description.

Figure 5 highlights the dataset features of the patient's health that tell us about the stroke attack and how each feature plays an important role in understanding and predicting stroke.

It has been observed that there are null values in the bmi column in doing the data description process and so we will replace the null values with the average values of the bmi. We also see categorical values in the dataset in the gender , ever_married , work_type , Residence_type and smoking_status columns that during data preprocessing , should be converted to numerical values using label encoders etc.

4.2. DATA VISUALIZATION

To highlight the relationship between the plotted features, we developed amazing visuals using matplotlib, seaborn libraries, and other tools. The major purpose is to raise awareness about stroke and how different clinical variables in the dataset play a significant effect in stroke incidence. We have exhibited and come to the realization of how the stroke impacts people living with diverse lifestyles, demographics, and backgrounds by performing count plots, pie charts, bar plots, and so on of the different clinical aspects like bmi, smoking status, work type, and gender.



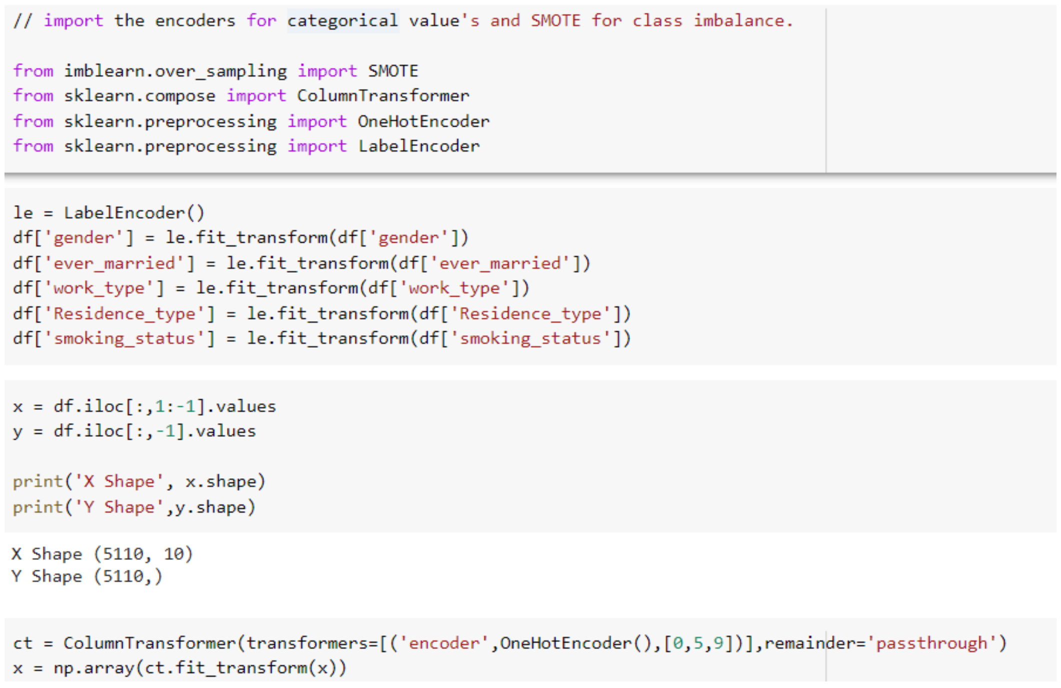
```
labels =df['stroke'].value_counts(sort = True).index
sizes = df['stroke'].value_counts(sort = True)

colors = ["lightblue","red"]
explode = (0.05,0)

plt.figure(figsize=(7,7))
plt.pie(sizes, explode=explode, labels=labels, colors=colors, autopct='%1.1f%%', shadow=True, startangle=90,)

plt.title('Number of stroke in the dataset')
plt.show()
```

Figure 6. Countplot of stroke number

In the above figure 6 , we have made a count plot around the number of stroke affected patients in the dataset that will give us the number of people who are positive (1) and affected with stroke and others who are negative (0).

4.3 DATA PREPROCESSING

Data preprocessing is a very important and crucial step in data science and machine learning because the dataset is often imbalanced in nature, has duplicate and null values, and so it's a must to preprocess and clean the dataset very efficiently without losing the original data so that we can use it wisely in machine and deep learning models while training, testing, and deployment in real time.



```
// import the encoders for categorical value's and SMOTE for class imbalance.

from imblearn.over_sampling import SMOTE
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder
from sklearn.preprocessing import LabelEncoder
```

```
le = LabelEncoder()
df['gender'] = le.fit_transform(df['gender'])
df['ever_married'] = le.fit_transform(df['ever_married'])
df['work_type'] = le.fit_transform(df['work_type'])
df['Residence_type'] = le.fit_transform(df['Residence_type'])
df['smoking_status'] = le.fit_transform(df['smoking_status'])
```

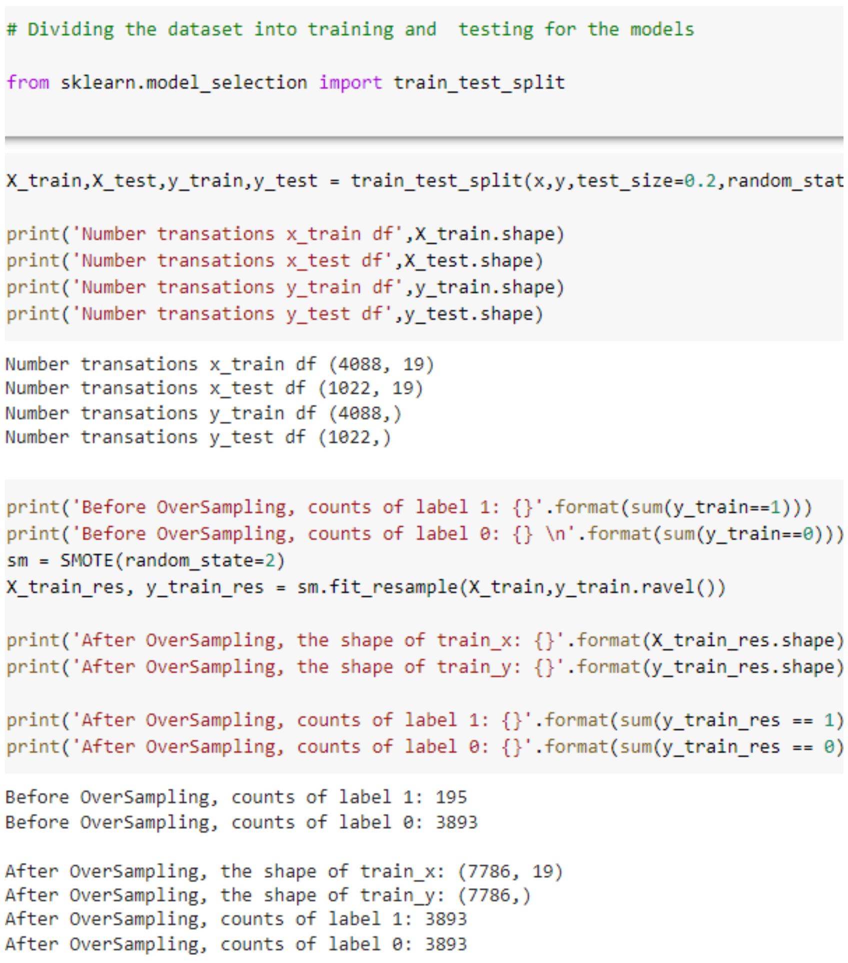
```
x = df.iloc[:,1:-1].values
y = df.iloc[:,-1].values

print('X Shape', x.shape)
print('Y Shape',y.shape)
```

```
X Shape (5110, 10)
Y Shape (5110,)
```

```
ct = ColumnTransformer(transformers=[('encoder',OneHotEncoder(),[0,5,9])],remainder='passthrough')
x = np.array(ct.fit_transform(x))
```

Figure 7. Label encoders

As shown in Figure 7, the first important step is to import the necessary libraries for preprocessing, such as LabelEncoder, OneHotEncoder, and ColumnTransformer, which will handle the Categorical values of the dataset and convert them to numerical, and finally we have SMOTE, which will balance the dataset for model training and improved performance. We will use the label encoder to convert the categorical values of the gender, ever married, work type, residence type, and smoking status columns to numerical values so that the machine learning models can train efficiently. Then, following the label encoder, we apply OneHotEncoder and Column transformer for additional processing before passing the dataset to the smote approach.



```
# Dividing the dataset into training and  testing for the models

from sklearn.model_selection import train_test_split
```

```
X_train,X_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_stat

print('Number transations x_train df',X_train.shape)
print('Number transations x_test df',X_test.shape)
print('Number transations y_train df',y_train.shape)
print('Number transations y_test df',y_test.shape)
```

```
Number transations x_train df (4088, 19)
Number transations x_test df (1022, 19)
Number transations y_train df (4088,)
Number transations y_test df (1022,)
```

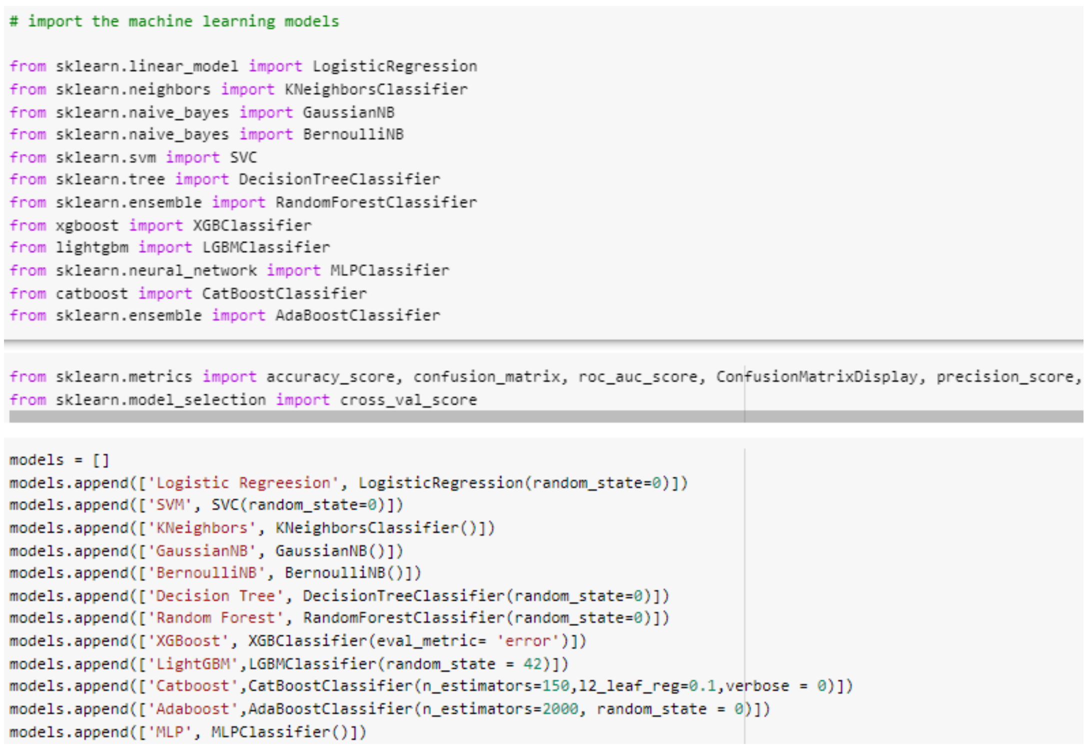
```
print('Before OverSampling, counts of label 1: {}'.format(sum(y_train==1)))
print('Before OverSampling, counts of label 0: {} \n'.format(sum(y_train==0)))
sm = SMOTE(random_state=2)
X_train_res, y_train_res = sm.fit_resample(X_train,y_train.ravel())

print('After OverSampling, the shape of train_x: {}'.format(X_train_res.shape)
print('After OverSampling, the shape of train_y: {}'.format(y_train_res.shape)

print('After OverSampling, counts of label 1: {}'.format(sum(y_train_res == 1)
print('After OverSampling, counts of label 0: {}'.format(sum(y_train_res == 0)
```

```
Before OverSampling, counts of label 1: 195
Before OverSampling, counts of label 0: 3893

After OverSampling, the shape of train_x: (7786, 19)
After OverSampling, the shape of train_y: (7786,)
After OverSampling, counts of label 1: 3893
After OverSampling, counts of label 0: 3893
```

Figure 8. Splitting the dataset and and smote implementation

We divide the dataset into Training and Testing Datasets using the train test split function of the sklearn.model selection package so that we can train the models on one set of the dataset for training and another unobserved dataset just for testing the performance and accuracy of the trained models. As seen in the figure 8, label 1 has a count of 195, forming the minority class, whereas label 0 has a count of 3893, forming the majority class.So this is the class imbalance that we have in the dataset between the majority and minority class and so machine learning models train more towards the majority class and minority class has a little effect on performance and so therefore we use SMOTE [14] , an oversampling technique to remove the class imbalance and oversample the minority class to get a balanced dataset that can be used for model training. Oversampling the minority class of label 1 with the synthetic values will lead to the count of label 1 equal to count of label 0 and therefore the dataset is balanced in nature and can be passed successfully to the training of the models.

4.3. TRAINING



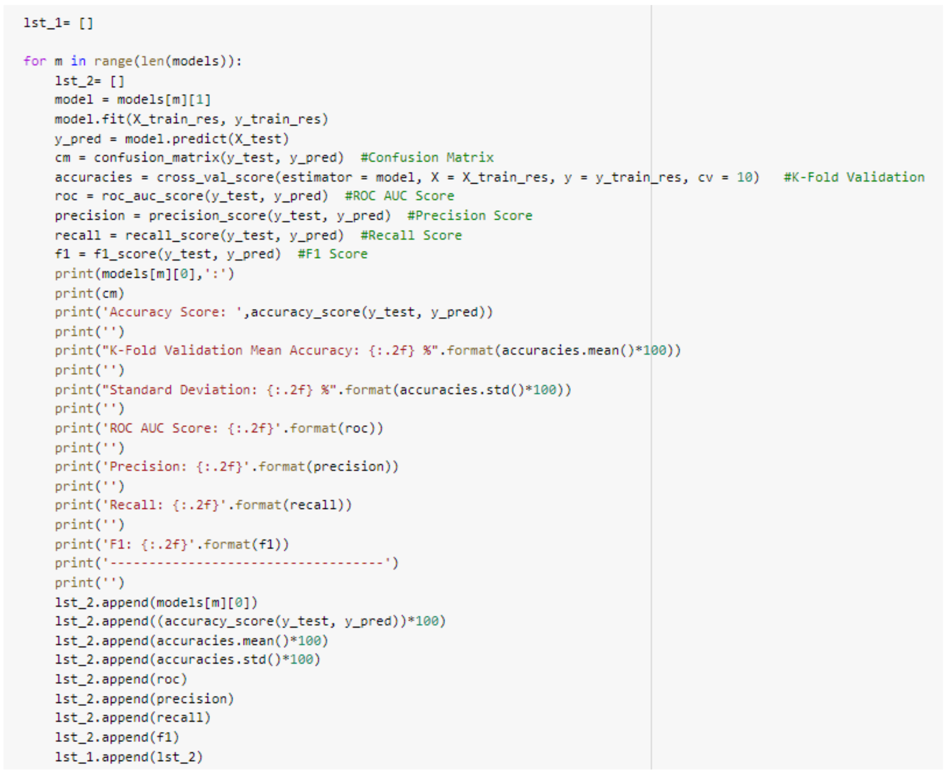
```
# import the machine learning models

from sklearn.linear_model import LogisticRegression
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.naive_bayes import BernoulliNB
from sklearn.svm import SVC
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from xgboost import XGBClassifier
from lightgbm import LGBMClassifier
from sklearn.neural_network import MLPClassifier
from catboost import CatBoostClassifier
from sklearn.ensemble import AdaBoostClassifier
```

```
from sklearn.metrics import accuracy_score, confusion_matrix, roc_auc_score, ConfusionMatrixDisplay, precision_score,
from sklearn.model_selection import cross_val_score
```

```
models = []
models.append(['Logistic Regreesion', LogisticRegression(random_state=0)])
models.append(['SVM', SVC(random_state=0)])
models.append(['KNeighbors', KNeighborsClassifier()])
models.append(['GaussianNB', GaussianNB()])
models.append(['BernoulliNB', BernoulliNB()])
models.append(['Decision Tree', DecisionTreeClassifier(random_state=0)])
models.append(['Random Forest', RandomForestClassifier(random_state=0)])
models.append(['XGBoost', XGBClassifier(eval_metric= 'error')])
models.append(['LightGBM',LGBMClassifier(random_state = 42)])
models.append(['Catboost',CatBoostClassifier(n_estimators=150,l2_leaf_reg=0.1,verbose = 0)])
models.append(['Adaboost',AdaBoostClassifier(n_estimators=2000, random_state = 0)])
models.append(['MLP', MLPClassifier()])
```

Figure 9. Importing the classifiers for modeling

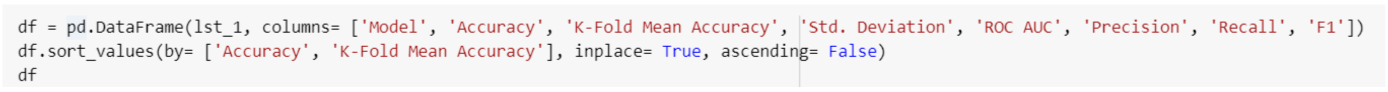
As seen in the code snippet highlighted in figure 9, we import numerous types of machine learning classifiers from the sklearn library or pip install the relevant ones, which are as follows: XGBoost[15], LightGBM[19], CatBoost[20], Random Forest, AdaBoost, MLP, SVM, Logistic Regression, Decision Tree, KNeighbours, BernoulliNB, and GaussianNB. We will train and test each model, calculating its Accuracy Score, ROC AUC Score, Precision Score, Recall, F1 Score, and so on. All of these classifiers were chosen for our prediction system and comparative analysis after careful consideration, which included conducting an appropriate literature survey of previous research works, as seen in the earlier section of the project report, and exploring alternative machine learning classifiers that had not previously been used in stroke research, such as Xgboost, LightGBM, Catboost, and Adaboost, among others.



```python
lst_1= []

for m in range(len(models)):
    lst_2= []
    model = models[m][1]
    model.fit(X_train_res, y_train_res)
    y_pred = model.predict(X_test)
    cm = confusion_matrix(y_test, y_pred)  #Confusion Matrix
    accuracies = cross_val_score(estimator = model, X = X_train_res, y = y_train_res, cv = 10)   #K-Fold Validation
    roc = roc_auc_score(y_test, y_pred)  #ROC AUC Score
    precision = precision_score(y_test, y_pred)  #Precision Score
    recall = recall_score(y_test, y_pred)  #Recall Score
    f1 = f1_score(y_test, y_pred)  #F1 Score
    print(models[m][0],':')
    print(cm)
    print('Accuracy Score: ',accuracy_score(y_test, y_pred))
    print('')
    print("K-Fold Validation Mean Accuracy: {:.2f} %".format(accuracies.mean()*100))
    print('')
    print("Standard Deviation: {:.2f} %".format(accuracies.std()*100))
    print('')
    print('ROC AUC Score: {:.2f}'.format(roc))
    print('')
    print('Precision: {:.2f}'.format(precision))
    print('')
    print('Recall: {:.2f}'.format(recall))
    print('')
    print('F1: {:.2f}'.format(f1))
    print('-----------------------------------')
    print('')
    lst_2.append(models[m][0])
    lst_2.append((accuracy_score(y_test, y_pred))*100)
    lst_2.append(accuracies.mean()*100)
    lst_2.append(accuracies.std()*100)
    lst_2.append(roc)
    lst_2.append(precision)
    lst_2.append(recall)
    lst_2.append(f1)
    lst_1.append(lst_2)
```

Figure 10. Training the models

As seen in the code snippet highlighted in figure 10 , we append the models to a 1d array one by one with their desired parameters like random_state , eval_metric , number of leaves and n_estimators etc for the training purposes and we fit the model with the training dataset , train them and then after , we test the models on the testing dataset and check the models performance on different parameters like accuracy score , roc auc score , precision , recall , f1 score , standard deviation etc. Eventually the most important parameter to check during the testing is the accuracy score of each one of the models.



```python
df = pd.DataFrame(lst_1, columns= ['Model', 'Accuracy', 'K-Fold Mean Accuracy', 'Std. Deviation', 'ROC AUC', 'Precision', 'Recall', 'F1'])
df.sort_values(by= ['Accuracy', 'K-Fold Mean Accuracy'], inplace= True, ascending= False)
df
```

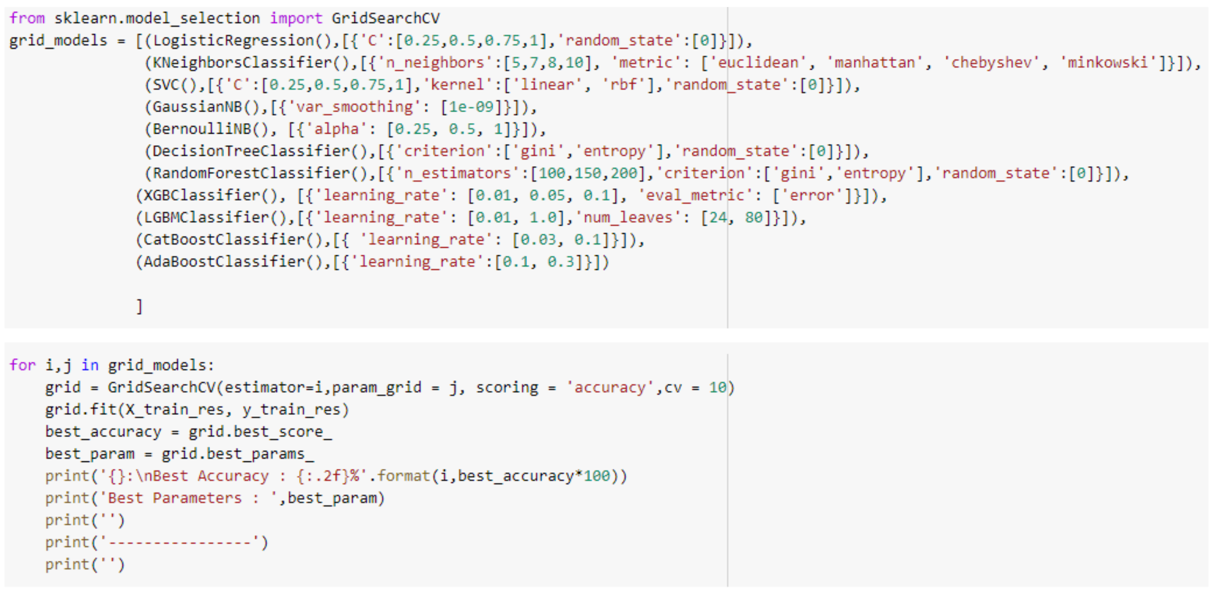
Figure 11. Compiling the results of training and testing

So then after training and testing , we will do the comparative analysis of the performance of the models on the basis of their accuracy and different parameters used while testing with the help of pandas dataframe that will make a simple table with the models name , accuracy and other parameters and then the it will be sorted in a descending order so that to check which are the top performing models as seen in the figure 11.

However, even though the performance of the models is comparatively good, we will not stop there, and in order to further improve the performance and accuracy of the existing models, we will have to do hyperparameter tuning of the models to further boost the overall quality of the prediction system and research work.

4.4. TUNING

To improve the accuracy and performance of the models even further, we will train them again on the dataset with the desired hyperparameters which will take some time to train. After Fine-tuning the models, we Fit the Testing data to see which one is the best. Hyperparameters chosen can range from Learning rate , evaluation metrics etc.

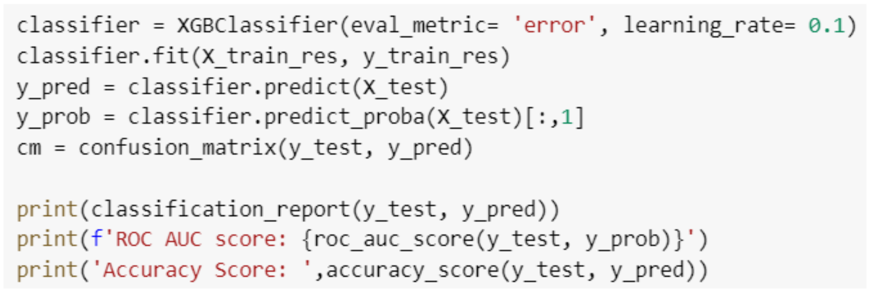


```
from sklearn.model_selection import GridSearchCV
grid_models = [(LogisticRegression(),[{'C':[0.25,0.5,0.75,1],'random_state':[0]}]),
               (KNeighborsClassifier(),[{'n_neighbors':[5,7,8,10], 'metric': ['euclidean', 'manhattan', 'chebyshev', 'minkowski']}]),
               (SVC(),[{'C':[0.25,0.5,0.75,1],'kernel':['linear', 'rbf'],'random_state':[0]}]),
               (GaussianNB(),[{'var_smoothing': [1e-09]}]),
               (BernoulliNB(), [{'alpha': [0.25, 0.5, 1]}]),
               (DecisionTreeClassifier(),[{'criterion':['gini','entropy'],'random_state':[0]}]),
               (RandomForestClassifier(),[{'n_estimators':[100,150,200],'criterion':['gini','entropy'],'random_state':[0]}]),
              (XGBClassifier(), [{'learning_rate': [0.01, 0.05, 0.1], 'eval_metric': ['error']}]),
              (LGBMClassifier(),[{'learning_rate': [0.01, 1.0],'num_leaves': [24, 80]}]),
              (CatBoostClassifier(),[{ 'learning_rate': [0.03, 0.1]}]),
              (AdaBoostClassifier(),[{'learning_rate':[0.1, 0.3]}])

              ]
```

```
for i,j in grid_models:
    grid = GridSearchCV(estimator=i,param_grid = j, scoring = 'accuracy',cv = 10)
    grid.fit(X_train_res, y_train_res)
    best_accuracy = grid.best_score_
    best_param = grid.best_params_
    print('{}:\nBest Accuracy : {:.2f}%'.format(i,best_accuracy*100))
    print('Best Parameters : ',best_param)
    print('')
    print('----------------')
    print('')
```

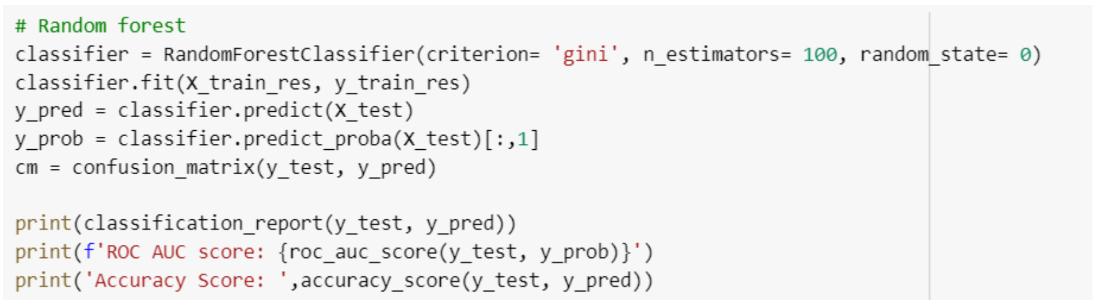
Figure 12. Tuning of the models with the desired parameters

So the Tuning of the models take a lot of time and we have carefully select the parameters of each and every parameter and in the figure 12 , so for example while tuning the XGBMClassifier() , we will tune it with an appropriate learning parameter and eval metric to improve the overall performance and accuracy and so like Xgboost , all these other models will be carefully tuned keeping in mind the improved performance and we will append these tuned models to an array named grid_models and after that we will do gridsearchcv on the grid_models which is a searching technique that will loop through the parameters defined and will choose the best one parameter for the model that will be used for training of the model once again and while training the model , we will only check the accuracy and best parameter in the output during runtime.



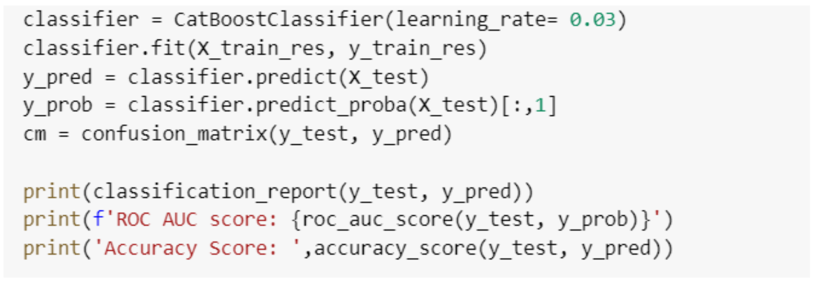
```
classifier = XGBClassifier(eval_metric= 'error', learning_rate= 0.1)
classifier.fit(X_train_res, y_train_res)
y_pred = classifier.predict(X_test)
y_prob = classifier.predict_proba(X_test)[:,1]
cm = confusion_matrix(y_test, y_pred)

print(classification_report(y_test, y_pred))
print(f'ROC AUC score: {roc_auc_score(y_test, y_prob)}')
print('Accuracy Score: ',accuracy_score(y_test, y_pred))
```

Figure 13. Testing tuned XGBoost



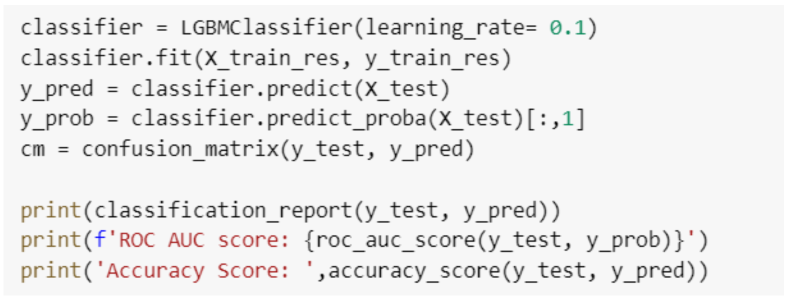
```
# Random forest
classifier = RandomForestClassifier(criterion= 'gini', n_estimators= 100, random_state= 0)
classifier.fit(X_train_res, y_train_res)
y_pred = classifier.predict(X_test)
y_prob = classifier.predict_proba(X_test)[:,1]
cm = confusion_matrix(y_test, y_pred)

print(classification_report(y_test, y_pred))
print(f'ROC AUC score: {roc_auc_score(y_test, y_prob)}')
print('Accuracy Score: ',accuracy_score(y_test, y_pred))
```

Figure 14. Testing tuned Random forest



```
classifier = CatBoostClassifier(learning_rate= 0.03)
classifier.fit(X_train_res, y_train_res)
y_pred = classifier.predict(X_test)
y_prob = classifier.predict_proba(X_test)[:,1]
cm = confusion_matrix(y_test, y_pred)

print(classification_report(y_test, y_pred))
print(f'ROC AUC score: {roc_auc_score(y_test, y_prob)}')
print('Accuracy Score: ',accuracy_score(y_test, y_pred))
```

Figure 15. Testing tuned CatBoost



```
classifier = LGBMClassifier(learning_rate= 0.1)
classifier.fit(X_train_res, y_train_res)
y_pred = classifier.predict(X_test)
y_prob = classifier.predict_proba(X_test)[:,1]
cm = confusion_matrix(y_test, y_pred)

print(classification_report(y_test, y_pred))
print(f'ROC AUC score: {roc_auc_score(y_test, y_prob)}')
print('Accuracy Score: ',accuracy_score(y_test, y_pred))
```

Figure 16. Testing tuned LightGBM

After tuning the models , we come to a conclusion that Random forest , CatBoost , XGBoost and LGBM are having the best performance in terms of training accuracy and so it is obvious that for our final comparative analysis of the models on the testing dataset , we will test these four models mentioned and not take the other ones as seen in the figures 13 - 16.

Testing will be done on the basis of their accuracy score and ROC AUC score which will clearly reveal to us which is the best classifier for model building in the future where the system will be used in the real time deployment in hospitals.

The research methodology should be followed step by step, with careful consideration given to clean code, visualizations, model development, and tuning, so that we may have a good prediction system with minimal error, and all protocols, ethics, and regulations must be followed. These research steps can be replicated by anyone with access to a laptop, Google colab notebook, and a working internet so that they, too, can gain insight and awareness on stroke and understand how, with the power of machine learning, they can also do the same innovations in the field of stroke prediction and ease the diagnosis of the patients.

CHAPTER 5: RESULT

So, first and foremost, we'd want to highlight the visualizations and plottings of the dataset features that were crucial in the connection of the features with stroke occurrence.

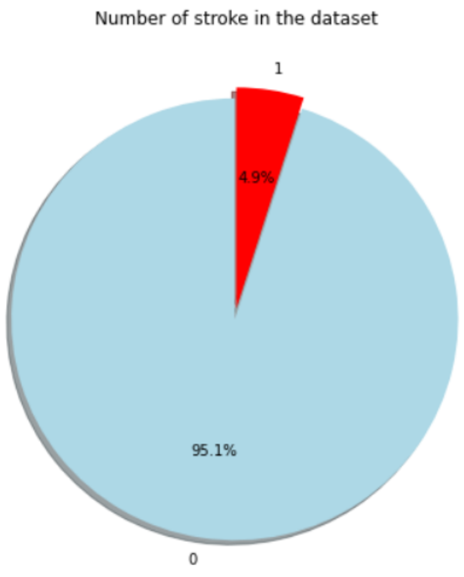


Figure 17. Pie Chart on number of stroke affected people

In this figure 17 is a pie chart plot of the number of people suffering from stroke in the dataset , we come to know only 4.9% suffer from stroke and are positive while 95.1% are negative with stroke and so it gives us a glimpse on the number of the stroke patients.

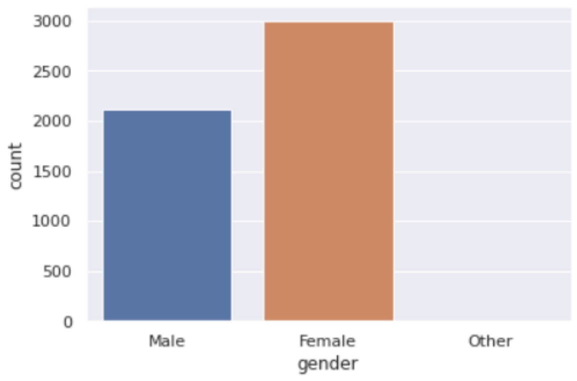


Figure 18. Count plot on gender in relation to stroke

In this countplot as seen in the figure 18 , we try to see the stroke on the basis of gender and how it affects us and it is clear from the dataset by visualization that patients are mostly female , 3000 in number and male patients with stroke are 2110 in number and so it is learnt from this females have a higher chance of contracting the stroke than males and so we should take a look in our lifestyles and improve our physical well being for a brighter stroke less future.

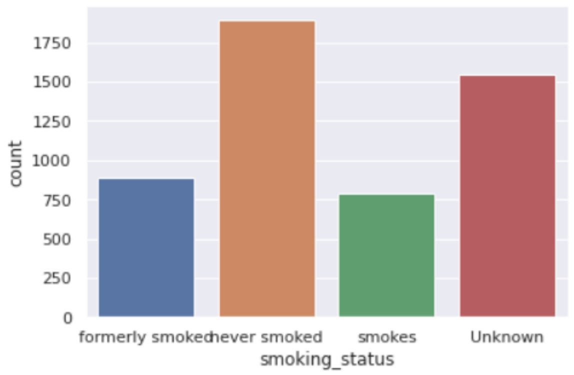


Figure 19. Plot of smoking status

Figure 19 implies that those people who smoke have good chances of having strokes over an older age because smoking contains nicotine which is very much harmful for our health and smoking is addictive in nature ie if you smoke one cigarette , then you will get the urge to smoke more and then it becomes a daily habit and it's hard to quit smoking and with time , your body undergoes several metabolic changes , your lungs get damaged , heart problems and you become weaker with time and that can play a vital role in stroke occurrence at the age 40 - 50 and so its advisable to quit smoking so to have a brighter future.

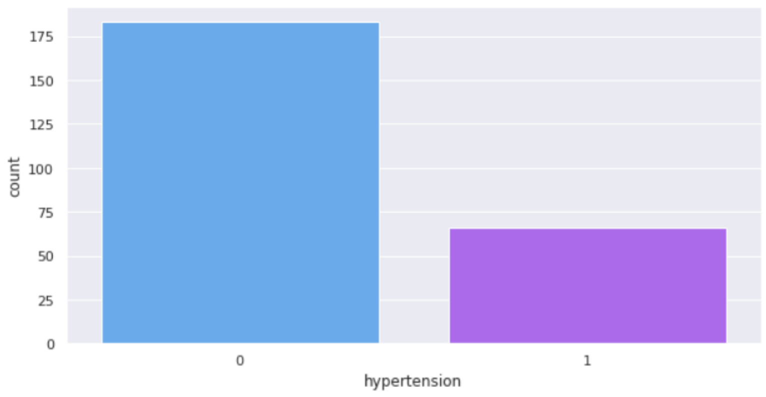


Figure 20. Hypertension in relation to stroke

Hypertension is related with high blood pressure that can lead to severe headaches , heart disease and nosebleed among many others and it is clearly revealed that from the countplot in the figure 20 that a considerable number of stroke patients suffer from hypertension and so a pattern is seen that hypertension along with smoking status can also play a vital factor in stroke incidence and so what needs to be done here is the lifestyle choices of the patients by taking food in low cholesterol , salt and fat that can help in the overall metabolism of the body and maintain homeostasis.

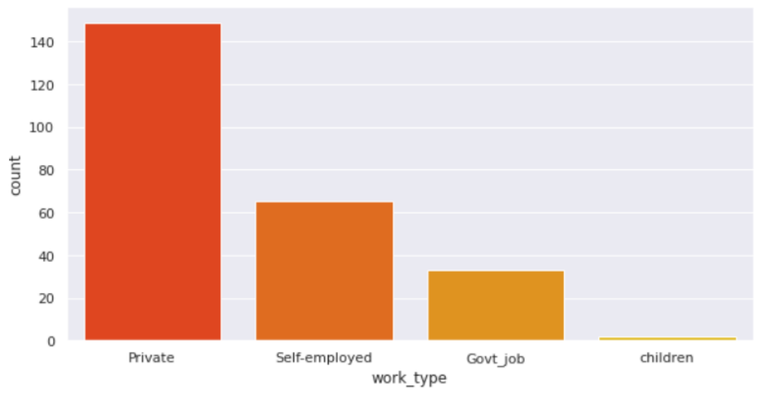


Figure 21. Work type in relation to stroke

In this plot as highlighted in the figure 21, we divide the stroke patients on the basis of their work_type or occupation and we have come to a new revelation that the people who are working in a private sector are more likely to get a stroke than those who belong to self employed , govt job or house people because it's the amount of stress from the workplace of private sector that can cause a rise in high blood pressure which is related to hypertension and heart disease and that will ultimately lead to stroke incidence and so they should indulge physical exercise and activities in gym and also should look after their mental well being so that they can have a longer life span and are not riddled with medical problems in an older age.

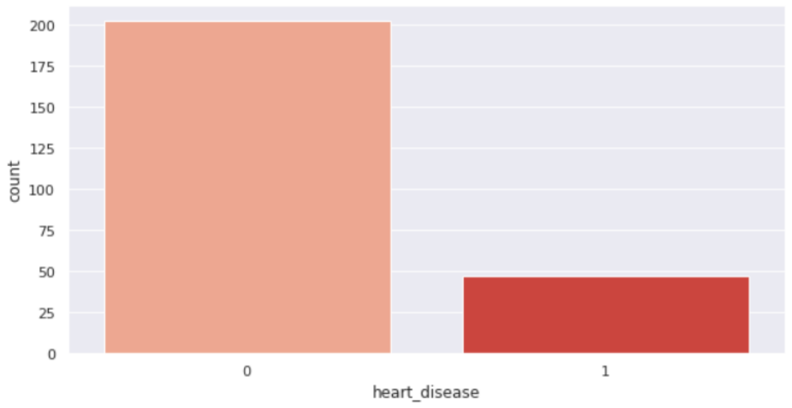


Figure 22. Heart disease effect on stroke occurrence

As seen in the figure 22 , heart disease is also prevalent risk factor is also prevalent in stroke patients that arises from the carefree lifestyle of the people who indulge in smoking , alcohol , no exercise and even though there are various types of heart diseases , heart disease and stroke can go hand in hand and so early symptoms of heart disease should be monitored so that we can avoid a stroke a later stage and so lifestyle choices can play a vital role in avoiding such medical risks at a later stage of life.

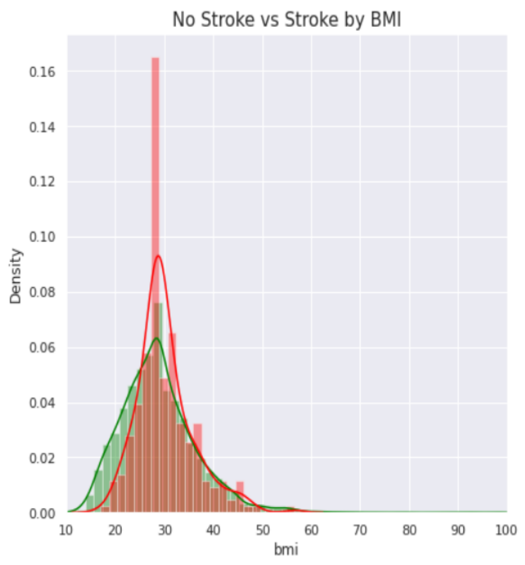


Figure 23. BMI plotting.

Figure 23 implies that people having a higher Body mass index (BMI) index have a good chance of contracting stroke because having more weight or obesity can cause cholesterol that can lead to fat accumulation in blood vessels which can medical problems like heart diseases , hypertension and ultimately to stroke and than can damage your later life with family persons and so maintain a body mass index below the predefined level and try to workout and take a look at your dietary intake which will ultimately help in avoiding such stroke scenarios.

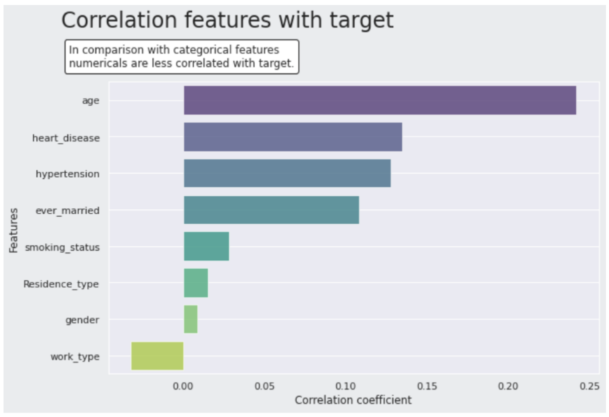
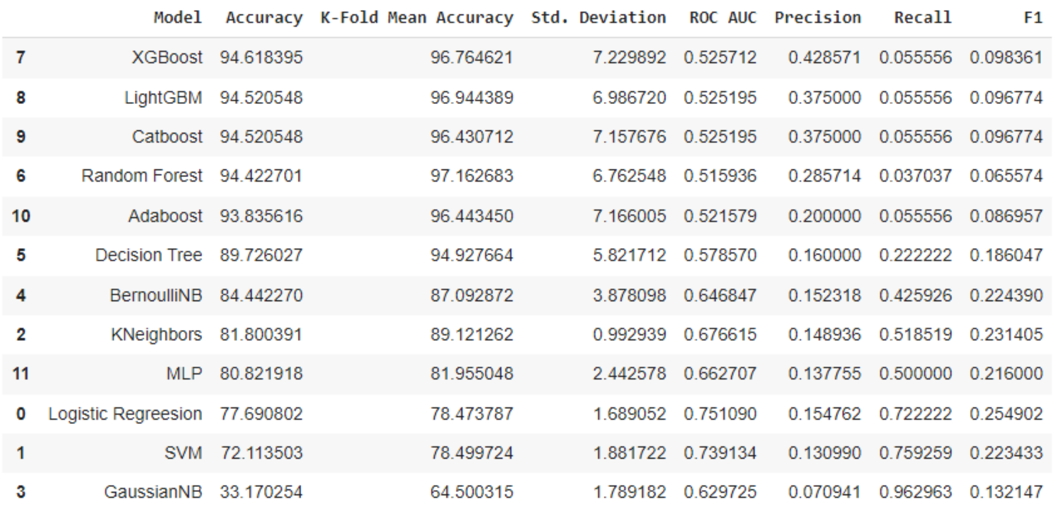


Figure 24. Correlation of features.

In figure 24 , we have demonstrated the correlation of the clinical features or the risk factors that contribute to stroke from the dataset and as seen in the previous plottings and now this , we have come to know that with age , chances of stroke increases followed by heart Disease , hypertension and marital status etc and so with age , one should be cautious of their lifestyle habits , particularly in the age group of 30 - 50 because its a sensitive age group of occurrence of variety of medical diseases that can ultimately lead to stroke in people because of their lifestyle choices and so one should take a considerable look at themselves and their physical well being and try to start working out as soon as possible because for stroke there is no cure and when it happens to you , you can die in rarest of cases , go into a coma or you can suffer paralysis of arms or legs or face and ultimately it can just destroy your life in terms of your career , family and health. So there is a sense of responsibility in each one of us to look after ourselves and try to avoid having medical conditions by making healthy choices.

As a result, the primary goal of creating these visualizations was to understand the patient's health and lifestyle factors that contribute to stroke attack, and by demonstrating such plottings in our research work, we can clearly create awareness among the people about the highlighted risk factors associated with stroke and what one should do to protect their life. Matplotlib and seaborn are fantastic Python libraries that substantially aided in the ease of charting clinical features from patient datasets and helped us to advance our research work with fresh insights and so we can go to the results from modeling of the prediction system in the following pages.



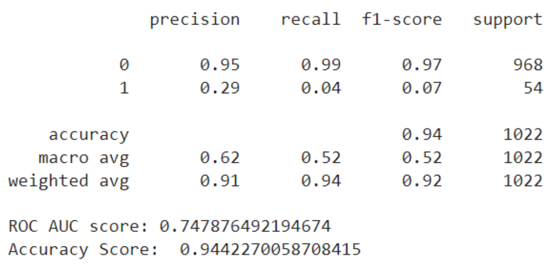
	Model	Accuracy	K-Fold Mean Accuracy	Std. Deviation	ROC AUC	Precision	Recall	F1
7	XGBoost	94.618395	96.764621	7.229892	0.525712	0.428571	0.055556	0.098361
8	LightGBM	94.520548	96.944389	6.986720	0.525195	0.375000	0.055556	0.096774
9	Catboost	94.520548	96.430712	7.157676	0.525195	0.375000	0.055556	0.096774
6	Random Forest	94.422701	97.162683	6.762548	0.515936	0.285714	0.037037	0.065574
10	Adaboost	93.835616	96.443450	7.166005	0.521579	0.200000	0.055556	0.086957
5	Decision Tree	89.726027	94.927664	5.821712	0.578570	0.160000	0.222222	0.186047
4	BernoulliNB	84.442270	87.092872	3.878098	0.646847	0.152318	0.425926	0.224390
2	KNeighbors	81.800391	89.121262	0.992939	0.676615	0.148936	0.518519	0.231405
11	MLP	80.821918	81.955048	2.442578	0.662707	0.137755	0.500000	0.216000
0	Logistic Regreesion	77.690802	78.473787	1.689052	0.751090	0.154762	0.722222	0.254902
1	SVM	72.113503	78.499724	1.881722	0.739134	0.130990	0.759259	0.223433
3	GaussianNB	33.170254	64.500315	1.789182	0.629725	0.070941	0.962963	0.132147

Figure 25. Comparative Analysis of Classifiers after training and testing

As stated earlier in the methodology , In the figure 25 we compare the models performance with accuracy , ROC- AUC , K - Fold Mean Accuracy, Precision , recall , f1 score and standard deviation in which we have to only mind accuracy score and roc auc score and it is observed that XGBoost is the top performing model in terms of all the performance metrics highlighted having an accuracy score of 94.61% and roc auc score of 0.52 followed by LightGBM , Catboost , Random Forest and many others etc.

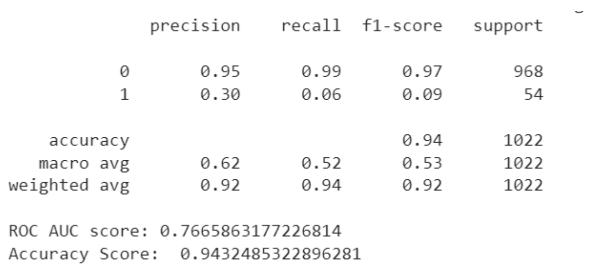
It is revealed through this that Gradient boosting models perform very well in stroke prediction and training and testing , they have showed impressive results like in the case of Xgboost and LightGBM and it's achieved because they follow the principles of boosting which comprises of ensemble of decision trees and with gradient descent as used to decrease the value of loss function. We can see the performance of MLP which is a simple neural network in the above figure but it's forth last in terms of performance during comparative analysis and so we can see a pattern here that machine learning classifiers perform very well in comparison to neural networks and that's why we initially didn't choose CNN , RNN and GAN because the dataset is in structured tabular format and therefore machine learning model works very well with such type of dataset unlike CNN and RNN which work well on image and temporal dataset like in the case of face recognition or time series like stock prediction.

After this Tuning comes, which is an integral part of our research work that aims to further enrich the models accuracy and give us new observations on the performance of the models on the testing dataset .Therefore , we tune each one of the Classifiers with their Desired parameters (Learning rate , n_estimators , eval_metrics , number of leaves etc) using gridsearchcv so that we can further improve their performance and so after , we train them once again on the training dataset to note their training accuracy and it is seen that that tuning with training takes a lot of time because of gridsearchcv that loops through all of the defined parameters of the each one of the models to find the best parameter or estimator that is added or fitted to the the model for training once again and so after training we observe that these four models which are XGBoost , Random forest , CatBoost and LightGBM stand out from others while achieving higher testing accuracy with their best defined parameter and so it stands to reason that we will only test these four to determine which model is the best for real-time stroke prediction and for comparative analysis purposes and so we fit the tuned and trained model on the testing dataset once again to see the improved accuracy score and roc auc score.



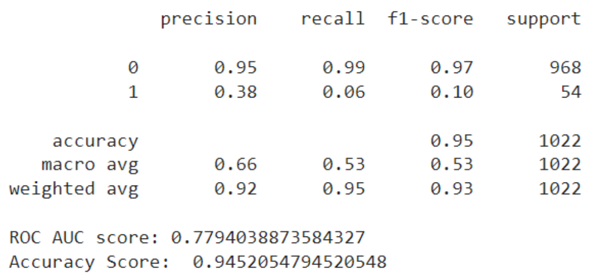
```
              precision    recall  f1-score   support

           0       0.95      0.99      0.97       968
           1       0.29      0.04      0.07        54

    accuracy                           0.94      1022
   macro avg       0.62      0.52      0.52      1022
weighted avg       0.91      0.94      0.92      1022

ROC AUC score: 0.747876492194674
Accuracy Score:  0.9442270058708415
```

Figure 26. Tuned Random Forest model performance



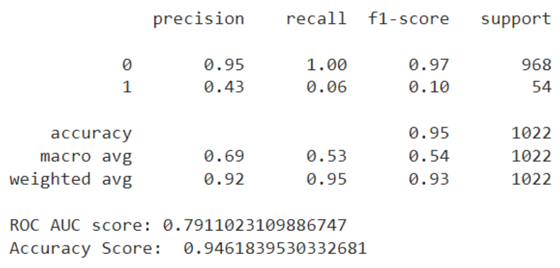
```
              precision    recall  f1-score   support

           0       0.95      0.99      0.97       968
           1       0.30      0.06      0.09        54

    accuracy                           0.94      1022
   macro avg       0.62      0.52      0.53      1022
weighted avg       0.92      0.94      0.92      1022

ROC AUC score: 0.7665863177226814
Accuracy Score:  0.9432485322896281
```

Figure 27. Tuned CatBoost performance



```
              precision    recall  f1-score   support

           0       0.95      0.99      0.97       968
           1       0.38      0.06      0.10        54

    accuracy                           0.95      1022
   macro avg       0.66      0.53      0.53      1022
weighted avg       0.92      0.95      0.93      1022

ROC AUC score: 0.7794038873584327
Accuracy Score:  0.9452054794520548
```

Figure 28. Tuned LightGBM performance



```
              precision    recall  f1-score   support

           0       0.95      1.00      0.97       968
           1       0.43      0.06      0.10        54

    accuracy                           0.95      1022
   macro avg       0.69      0.53      0.54      1022
weighted avg       0.92      0.95      0.93      1022

ROC AUC score: 0.7911023109886747
Accuracy Score:  0.9461839530332681
```

Figure 29. Tuned XGBoost performance

From Figure 26 to 29 , we see the performance of these 4 Tuned models on the Testing data once again and so it is clear from the new observations that XGBoost is the best performing model on the stroke dataset during the comparative analysis and it achieves an impressive all rounder accuracy score of 94.61% and roc auc score of 0.79 and it is followed by LightGBM , Random forest and Catboost and their results are also neatly compiled in the below table for more clarity and so in conclusion these are the final results from our research work of comparative analysis on stroke dataset for the prediction system in which we have tried all the machine learning approaches and in the XGBoost emerged as the winner and best classifier for modeling of the prediction system.

Table 5.1 comparative analysis of tuned classifiers.

Model	**Accuracy Score**
Xgboost	**94.61 %**
LightGBM	**94.52%**
Random forest	**94.42 %**
Catboost	**94.32 %**

Table 5.1 displays the accuracy of the four tuned models throughout testing, revealing that Xgboost is the best model in terms of accuracy score. XGBoost wins our comparison because it is based on gradient boosting, which is an ensemble learning method that uses gradient descent to decrease the value of the loss function, and it adheres to the principles of regularization, tree pruning, and parallelization, giving it a competitive advantage over LightGBM, CatBoost, and Random forest.

CHAPTER 6: CONCLUSION

So, to summarize, for our prediction system, we imported the dataset from Kaggle, cleaned it by removing duplicate or null values, and then displayed insightful visualizations in the form of Plots that revealed our understanding of features influencing the stroke. After that, we preprocessed the dataset using label encoding and other encoders to handle the categorical values to numerical, and then SMOTE was used to balance the dataset for modeling and maintaining consistency. We trained the models, which consisted of a variety of machine learning classifiers, on the training dataset and then did a comparative analysis of the results with the different parameters for which the models were tested. After that, we did hyperparameter tuning of models with the defined, desired parameters of each one of the classifiers and then trained them again, checked their training accuracy, and then selected the best performing four models, which are: XGboost , Random forest , CatBoost and LightGM for testing and fitting on the testing dataset for the final comparative analysis.

6.1. PERFORMANCE ESTIMATION

On testing the tuned models , we find that XGBoost (94.61 percent) is the best performing model, followed by LightGBM (94.52 percent), Random Forest (94.42 percent), and CatBoost (94.41 percent) (94.32 percent). This is why XGBoost is always at the top of the machine learning classifiers, and it has been used many times in renowned data science competitions and research work because of its unique boosting architecture with the necessary modifications that gives it superior speed and performance over structured datasets, and so for our prediction system, by the order of comparative analysis, XGBoost is the top contender and classifier for our prediction system to use.

6.2. USABILITY OF PRODUCT / SYSTEM

XGBoost Model should be integrated in real-time stroke prediction applications and hardware systems where it will produce accurate results since it has been vigorously trained , tuned and tested on the stroke dataset and has shown an impressive allrounder results while testing and so whatever may the patient input or type of structured data feed and fitted to the XGBoost model on deployment for testing , it can accurately give out the results with precision and those results can used by the doctors and physicians for diagnosis of the early stroke patients who are showing the symptoms and put them in the immediate care of the hospital and medication and prescriptions.

6.3. FURTHER IMPROVEMENT

We can further improve the quality of our research work and observations by increasing the size of our dataset with addition of the new stroke patients with their clinical features added to the table so that XGBoost model can learn upon new data of the patients added with training and testing which can improve the overall performance of the prediction system or else we can also train and test the XGBoost from the other data sources or data centers of different demographics , countries , regions so that the model can learn from new type of data unseen before and can be used in almost all type of medical situations related to stroke diagnosis so that we can have richer results and insights with new possibilities and observations.

CHAPTER 7: REFERENCES

[1] M. S. Singh and P. Choudhary, “Stroke prediction using artificial intelligence,” 2017 8th Ind. Autom. Electromechanical Eng. Conf. IEMECON 2017, pp. 158–161, Oct. 2017, doi: 10.1109/IEMECON.2017.8079581.

[2] A. Sudha, P. Gayathri, and N. Jaisankar, “Effective Analysis and Predictive Model of Stroke Disease using Classification Methods,” Int. J. Comput. Appl., vol. 43, no. 14, pp. 26–31, 2012, doi: 10.5120/6172-8599.

[3] C. H. Lin et al., “Evaluation of machine learning methods to stroke outcome prediction using a nationwide disease registry,” Comput. Methods Programs Biomed., vol. 190, p. 105381, Jul. 2020, doi: 10.1016/J.CMPB.2020.105381.

[4] T. Liu, W. Fan, and C. Wu, “A hybrid machine learning approach to cerebral stroke prediction based on imbalanced medical dataset,” Artif. Intell. Med., vol. 101, p. 101723, Nov. 2019, doi: 10.1016/J.ARTMED.2019.101723.

[5] H. Ahmed, S. F. Abd-El Ghany, E. M. G. Youn, N. F. Omran, and A. A. Ali, “Stroke prediction using distributed machine learning based on apache spark,” Int. J. Adv. Sci. Technol., vol. 28, no. 15, pp. 89–97, 2019, doi: 10.13140/RG.2.2.13478.68162.

[6] C. S. Nwosu, S. Dev, P. Bhardwaj, B. Veeravalli, and D. John, “Predicting Stroke from Electronic Health Records,” Proc. Annu. Int. Conf. IEEE Eng. Med. Biol. Soc. EMBS, pp. 5704–5707, Jul. 2019, doi: 10.1109/EMBC.2019.8857234.

[7] P. Chantamit-O-Pas and M. Goyal, “Prediction of stroke using deep learning model,” Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics), vol. 10638 LNCS, no. May, pp. 774–781, 2017, doi: 10.1007/978-3-319-70139-4_78.

[8] Y. Wu and Y. Fang, “Stroke Prediction with Machine Learning Methods among Older Chinese,” Int. J. Environ. Res. Public Health, vol. 17, no. 6, Mar. 2020, doi: 10.3390/IJERPH17061828.

[9] K. A. MAHESH, N. H. SHASHANK, S. SRIKANTH, and M. A. THEJAS, “Prediction of Stroke Using Machine Learning,” no. June, pp. 1–9, 2020, [Online]. Available: https://www.researchgate.net/publication/342437236_Prediction_of_Stroke_Using_Machine_Learning.

[10] S. N. Min, S. J. Park, D. J. Kim, M. Subramaniyam, and K. S. Lee, “Development of an Algorithm for Stroke Prediction: A National Health Insurance Database Study in Korea,” Eur. Neurol., vol. 79, no. 3–4, pp. 214–220, May 2018, doi: 10.1159/000488366.

[11] “Stroke Prediction Dataset | Kaggle.” https://www.kaggle.com/fedesoriano/stroke-prediction-dataset (accessed Apr. 03, 2022).

[12] https://www.cdc.gov/stroke/index.htm

[13] Amini L, Azarpazhouh R, Farzadfar MT, Mousavi SA, Jazaieri F, Khorvash F, et al. Prediction and control of stroke by data mining. Int J Prev Med 2013;Suppl 2: S245-9

[14] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “SMOTE: Synthetic minority over-sampling technique,” J. Artif. Intell. Res., vol. 16, pp. 321–357, 2002, doi: 10.1613/JAIR.953.

[15] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min., vol. 13-17-August-2016, pp. 785–794, Aug. 2016, doi: 10.1145/2939672.2939785.

[16] J. N. Heo, J. G. Yoon, H. Park, Y. D. Kim, H. S. Nam, and J. H. Heo, “Machine Learning-Based Model for Prediction of Outcomes in Acute Stroke,” Stroke, vol. 50, no. 5, pp. 1263–1265, May 2019, doi: 10.1161/STROKEAHA.118.024293.

[17] E. M. Alanazi, A. Abdou, and J. Luo, “Predicting Risk of Stroke From Lab Tests Using Machine Learning Algorithms: Development and Evaluation of Prediction Models,” JMIR Form. Res., vol. 5, no. 12, Dec. 2021, doi: 10.2196/23440.

[18] B. Letham, C. Rudin, T. H. McCormick, and D. Madigan, “Interpretable classifiers using rules and bayesian analysis: Building a better stroke prediction model,” Ann. Appl. Stat., vol. 9, no. 3, pp. 1350–1371, 2015, doi: 10.1214/15-AOAS848.

[19] G. Ke et al., “LightGBM: A Highly Efficient Gradient Boosting Decision Tree,” doi: 10.5555/3294996.

[20] J. T. Hancock and T. M. Khoshgoftaar, “CatBoost for big data: an interdisciplinary review,” J. Big Data, vol. 7, no. 1, Dec. 2020, doi: 10.1186/S40537-020-00369-8.