Department of Electrical and Computer Engineering

North-South University

Senior Design Project

Machine Learning Powered Stroke Predicting Web Application

Eshan Ahmed 1712918642

Redwan Ahmed 1721279042

Shahriar Hossain 1712852642

Faculty Advisor:

Md. Shahriar Hussain

Lecturer

ECE Department

Summer, 2021

LETTER OF TRANSMITTAL

May 2022

Md. Shahriar Hussain
Lecturer
Department of Electrical and Computer Engineering
North South University

Subject: Submission of the Final Project Report

Dear Sir,
Assalamu Alaikum, Sir. This is to inform you that we have been doing our final project report under the academic curriculum as a course CSE499A and CSE499B. This project report is to fulfill a Computer Science and Engineering (CSE) Bachelor's degree requirements. It has been proved very effective as we completed our final project.

We have completed our final project report on "Machine Learning Powered Stroke Predicting Web Application." In the report, we have tried to accommodate your valuable comments and suggestions.

Thank you for your kind cooperation. Without your support, we would not have completed this report. So, we are submitting the final version of the final project report and requesting your acceptance.

Regards,

.....
Eshan Ahmed,
Redwan Ahmed,
Shahriar Hossain,
ECE Department
North South University, Bangladesh

APPROVAL

Eshan Ahmed (ID# 1712918642), Shahriar Hossain(ID #1712852642) and Redwan Ahmed (ID # 1721279042) from Electrical and Computer Engineering Department of North South University, have worked on the Project titled “Machine Learning Powered Stroke Predicting Web Application.” under the supervision of Md. Shahriar Hussain partial fulfillment of the requirement for the degree of Bachelors of Science in Engineering and has been accepted as satisfactory.

Supervisor’s Signature

.....
Md. Shahriar Hussain

Lecturer
Department of Electrical and Computer Engineering
North South University

Chairman’s Signature

.....
Dr. Mohammad Rezaul Bari

Associate Professor
Department of Electrical and Computer Engineering
North South University

DECLARATION

We solemnly declare that the project report on "MACHINE LEARNING POWERED STROKE PREDICTING WEB APPLICATION" is based on our work carried out during our study under the supervision of Md Shahriar Hussain. We assert that the statements made and conclusions generated are an outcome of our research work. No part of this work was submitted elsewhere partially or fully to award any other degree or diploma. Any material reproduced in this project has been adequately acknowledged.

Students' Names Signatures

1. Eshan Ahmed

_ _ _ _ _ _ _ _ _ _ _ _ _ _

2. Redwan Ahmed

_ _ _ _ _ _ _ _ _ _ _ _ _ _

3. Shahriar Hossain

_ _ _ _ _ _ _ _ _ _ _ _ _ _

ACKNOWLEDGEMENT

We would like to express our most profound appreciation to all those who provided us the possibility to complete this report. Special gratitude to our final year project supervisor, Mr. Hussain, whose contribution in stimulating suggestions and encouragement helped us coordinate our project, especially in writing this report.

We are grateful to all of the lecturers who have assisted us in completing the project with our full knowledge by offering technical knowledge and ethical assistance.

We are grateful to the North South University, ECE Department for providing this course CSE499.

We appreciate our family's support.

ABSTRACT

Machine Learning Powered Stroke Predicting Web Application

Stroke is one of the major causes of severe health issues and death cases in the world. A chance of strokes may develop gradually in one person's body. Early prediction of stroke is useful for prevention, so it is possible to reduce the chance of stroke by taking regular checkups. The main objective of our project is to detect the possibility of stroke by machine learning which is a subfield of artificial intelligence. This machine learning part will be a significant part of our work. Another important component of our work is that we built a web platform based on machine learning. Here users will be able to make predictions simply by putting some values on our stroke prediction form. Again, the required values are user-friendly, so one will not need to go through any unnecessary hassle to collect them. All of the required attributes are related to one's lifestyle and overall general health condition. After getting the user's required attributes' value, our system will analyze those by our machine learning model, which we built earlier. Then our system will send a response if the user has any chance of having a stroke or not. Among all attributes or features, some crucial attributes play a significant role in having a stroke in the future. Our system will also prompt the user to predict which aspects of their health they should pay attention to. Our system will also store users' input attributes and their corresponding prediction results. This will help our users to keep track of their health conditions. We built and tested our whole system with utmost precision so that it can help our users to predict stroke before it happens.

TABLE OF CONTENT

List of Figures

List of Tables

Chapter 1
Overview

1.1 Introduction

Imagine a world where there is a one less fatal disease. We can use our knowledge in computer science to create a better world for us and our future generations. This project aims to fulfill this vision as we implement machine learning to solve real-world problems. According to the Global Burden of Disease study, stroke is one of the leading causes of death and morbidity worldwide and a significant health problem. A stroke occurs when there is an obstruction of the blood vessels that supply blood to the brain.[1] Our goal in this project is to use machine learning techniques to detect the probability of stroke. Machine learning (ML), an application of artificial intelligence, may use multiple imaging features, including those even invisible to humans with consistent accuracy. Since we work with a large dataset, the machine learning algorithm helps visualize the data and find correlations to generate accurate results.

1.2 Project Description

The main objective of our work is to create a machine learning model and a website for our users. This machine learning model is the backbone of our project as it will play a vital role in predicting stroke. Our machine model is integrated into a web application. Our user will make predictions from that website by putting the required attributes.

1.3 Project Purpose

This project aims to detect the possibility of a stroke to take early prevention measures and minimize the risk. If the stroke is detected as early as possible, there is a high chance that patients can take early preventative measures to minimize fatal damage. Implementing this project using Machine learning practices is also a challenge, so our purpose lies there. We want to give something back to our society, and this project is one of the best ways to do it.

1.4 Report Outline

Every chapter contains different aspects of the work that we did. Here we will discuss how the report is organized. In each chapter individual aspects are briefly described and how it contributes to the whole work.

Chapter 1	In this chapter, we presented a brief overview of our work. The motive of the work, usage of machine learning in the medical field, the importance of machine learning, and most importantly how machine learning can solve our problem in predicting stroke is discussed here.
Chapter 2	Here all the related works that were done before are discussed briefly. Many individual projects were done to solve this kind of problem. Though the goal is the same, their methods of working are different. The difference can exist in terms of analysis, attribute selection, algorithm selection, and many other things.

Chapter 3	Here we discussed our theoretical background. The actual work that is done behind the scenes is discussed here. In machine learning, many mathematical operations are done specially the statistics branch that gave a solid foundation for machine learning. Here we also showed how the implementation of statistics is done in a programmatic way and the tools or frameworks that are needed for machine learning
Chapter 4	This chapter is the core aspect of our work. Here we discussed each step that we did to build this model. From data collection to choose the best model for our work is discussed here. Here the implementation of required steps for machine learning are vividly shown.
Chapter 5	Results are shown in this chapter that we got from our work. Our results varied for different models. The comparisons between the models were also done. The outcome we got after the web application is also

	shown here.
Chapter 6	This is the conclusion part. Here we concluded everything about the final decisions for building the system, project development phases, advancements, and ethical and professional responsibilities.

Chapter 2
Related work

There are several papers and work on stroke. Since stroke treatment is not risk-free, physicians only proceed with treatment when the potential benefits outweigh the perceived risks. For this reason, tools that can help predict the patient's functional outcome using only data available when the patient gets admitted to the hospital are handy because they can help inform the treatment decision. Our work is to help patients before they are even admitted to the hospitals. The medical community has made several efforts to create some advancements to minimize the risk of stroke. We have read papers where similar works were conducted to identify a stroke.

Miguel Monteiro proposed an ML system for stroke prediction. They have used similar algorithms like SVM, Linear regression, and Decision Tree.[2]

Another work is performed on stroke and neurovascular medicine by Hamidreza Saber and Melek Somai. They described various tools and techniques in predictive analytics. It will analyze the trends in applying these techniques over recent years. They have concluded that Future

predictive modeling will likely combine a wide range of data points such as images, biomarkers, and continuous real-time monitoring using wearable devices. This deluge of data would feed advanced deep learning and neural network algorithms to accurately predict the risk of health conditions such as aneurysm rupture or stroke treatment [3].

Hamann Janne makes stroke predictions for patients with MCA-M1 occlusions and early thrombectomy. This paper has included 222 patients with acute ischemic stroke due to the middle cerebral artery. (MCA)-M1 occlusion who received EVT. They used clinical variables and region of interest (ROI) based magnetic resonance imaging (MRI) features as predictors. They developed different machine learning models and quantified their prediction performance by the area under the curve (AUC) of receiver operator characteristics (ROC) curves and the Brier score.[4]

Hendrikus J. A. van Os performed a work where he focuses on patients from the Multicenter Randomized Clinical Trial of Endovascular Treatment for Acute Ischemic Stroke in the Netherlands (MR CLEAN) Registry, an observational cohort of LVO patients treated with EVT. They applied the following machine learning algorithms: Random Forests, Support Vector Machine, Neural Network, and Super Learner and compared their predictive value with classic logistic regression models using various variable selection methodologies.[5]

Yafei Wu made a stroke Prediction among the Older Chinese people. A paper on a study aimed to develop machine learning models for predicting stroke with imbalanced data in an elderly population in China. Data were obtained from a prospective cohort that included 1131 participants (56 stroke patients and 1075 non-stroke participants) in 2012 and 2014. Data balancing techniques including random over-sampling (ROS), random under-sampling (RUS), and synthetic minority over-sampling technique (SMOTE) were used to process the imbalanced data in this study. Machine learning methods such as regularized logistic regression (RLR), support vector machine (SVM), and random forest (RF) were used to predict stroke with demographic, lifestyle, and clinical variables.[6]

Ching-Heng Lin evaluated many machine learning methods to predict stroke using the Nationwide disease Registry of Taiwan. This study used the Taiwan Stroke Registry (TSR), which has prospectively collected data from stroke patients since 2006. Three known ML models (support vector machine, random forest, and artificial neural network) and a hybrid artificial neural network were implemented and evaluated by 10-time repeated hold-out with 10-fold cross-validation.[7]

A Practical Approach of Different Programming Techniques to Implement a Real-time Application using Django By Sebastian Stigler & Marina Burdack is a paper on implementing Django for real-time applications. Django is one of the popular high-level Python web frameworks to represent ML work, and we learned a lot from this paper. In this paper, they

described what kind of scaling options Python offers to implement the data processing component of the web application. They also described which of these options they have implemented and what architectural choices they made. Finally, they presented the mathematical foundations used to determine the scaling of the application.[8]

Django Website for Disease Prediction using Machine Learning by Sayali Kulkarni, Isha Sawant, Megha Shinde, Vishal Sonar, and Vaishali Latke. The work described in this is very similar to our work. Therefore we were able to gain valuable insights on accomplishing our work. The paper describes in detail the necessary steps for implementing the Django website. It helps us see a clear picture of who will be the users and get more knowledge about the architecture used to implement the Django framework.[9]

Chapter 3
Theoretical Background

3.1 Theoretical Overview

In simple words, machine learning combines statistics and its implementation with a programming language/framework/tool. As we know, it is broadly used to build prediction or forecasting systems. The idea behind this ML system is pretty similar and straightforward. A model or system needs a dataset that contains a wide suitable variety of relevant data. A particular incident occurs based on some attributes or features. We need to understand and analyze how these features impact an incident to happen. Furthermore, a major essential component of machine learning kicks in, which is statistics [10]. It helps us analyze the data and better understand why things are happening in a certain way. We got a mathematical foundation, but now it needs to be implemented somehow to find a way to express itself programmatically. For that, now we need a programming language or some tools that will help us to apply statistical formulas or methods. There are two major parts for any machine learning application: statistics and another tool or programming language for implementing statistics.

3.2 Fundamental Statistics

Statistics is a field of mathematics used to collect, analyze, forecast, and visualize data. There are many statistical methods to learn about the behavior or pattern of a dataset[11]. From the perspective of machine learning, a dataset is divided into two parts. One is features or attributes; another is label or target. Target or label is a part that represents the result of the incident that occurred. This is the part that we predict for or are mainly interested in. Another part of the dataset is a feature. This part is used to analyze, visualize so that it can produce an ideal outcome. There can be many variations in the feature, and the primary key point is to produce or predict the correct label for it [12]. Now our goal is to create a system to learn the features and their corresponding label so that after learning, we can give our system or model some feature or attribute as input and expect a perfect outcome as a label. In order to perform well, we have to make sure that our model learns all the variations or incidents as profoundly as possible from the dataset. Here is an example formula of the XGBoost algorithm, as shown below.

$$f(x) \approx f(a) + f'(a)(x-a) + \frac{1}{2}f''(a)(x-a)^2$$

$$\mathcal{L}^{(t)} \simeq \sum_{i=1}^{n} [l(y_i, \hat{y}^{(t-1)}) + g_i f_t(\mathbf{x}_i) + \frac{1}{2} h_i f_t^2(\mathbf{x}_i)] + \Omega(f_t)$$

Equation 3.1: XGBoost Formula [13]

This is the formula in equation 3.1 for the famous algorithm XGBoost, which certainly has created some waves in machine learning. Though it has many parameters, it primarily functions

on two, which can give optimal performance. n_estimators and learning rate are these two essential features.

n_estimators: The number of trees we will create before taking the maximum number of voting for predictions. A higher number of trees will give better performance.

Learning-rate: One characteristic of gradient boosting trees is that they are quick to learn and easily overfit the data. So, we need a method to slow down this learning process. That is why the learning rate is used. We have to choose the optimal point for selecting the learning rate.

3.3 Implementation of statistics with a programming language

A programming language/framework is needed for a system that will progressively and efficiently help us apply statistical methods. We must choose the correct language and efficient tools for building ML models that support us with an extensive collection of variant tools and methods. A language is needed to implement statistics and choose the correct framework and tools to perform mathematical operations quickly and efficiently. Indeed, we can start with any language and implement them from scratch, but it will not be suitable to reinvent the wheel. There are many statistical and mathematical methods or steps that are required for building an ML model. Data collection, coefficient correlation, stratified sampling, filling missing values with mean, visualizing data, train-test splitting, and many other operations are needed to build an ideal ML system [14]. Moreover, as a programming language for implementing these methods, the most commonly used language is Python. It has many libraries, frameworks, modules that have excellent support and a vast community. Python supports NumPy, pandas, Matplotlib, Scikit-learn, pickle, and many other tools. Pandas and NumPy for pre-processing, Scikit-learn for

train-test splitting and model training, Matplotlib for data visualization, and many other tools are available for performing necessary steps. In the next section, steps of mathematical operation and the usage of corresponding libraries will be briefly discussed.

3.4 Implementation of ML model into web application

We will integrate our final ML model into a web application to make predictions from the website. We will use Django, which is a popular framework of python. Django's MVT architecture will help us to manage different parts of the application. Model, view, and template are the three crucial aspects of Django. The model will handle the database part, and View will deal with all the business logic and functionality. Then the template will finally render the desired Html template for us.

Chapter 4
System Design

4.1 Data Collection and preprocessing

This step solely consists of collecting the suitable dataset and preprocessing the data so that our model can learn the right and ideal pattern. The overall performance of the model depends on the preprocessing part [15]. If this part is fixed in the right way, we can ensure that the model will perform well in almost any situation. Here we want to make our dataset so that it will be flexible enough for any machine learning algorithm to train and learn the pattern. Dataset Collection: We need to collect data from the right and authentic source. The dataset should come from real-world and actual incidents, not made-up random data from an unexpected source. This is where we especially paid attention while gathering the dataset. We collected ours from Kaggle, one of the largest websites for data science resources, materials, and datasets. Of course, the source was verified and authentic. Many other users also used this source for machine learning. There were some categorical or text data as well as numerical data. We cannot visualize or analyze text data mathematically to convert text data into numeric data. Filling with missing values: The dimension of our data is 5110 X 12. Although the data sizes were perfect, there were some

missing values for BMI, which is one of the crucial attributes for stroke prediction. There were 35-40 missing values in the attribute. As the number was not that much, we decided to fill missing values with mean, which is also an efficient technique for fewer missing values.

Any other attributes did not contain any missing values. Coefficient correlation is used to analyze how an attribute is correlated to the label. It determines how strongly or weakly the attribute is related to the label. It shows the result as positive correlation and negative correlation. Furthermore, the range of the value will be between –1 to +1. The more the value moves towards +1, that is a positive correlation, and the more the value moves towards –1, which is a negative correlation. In our case, age and hypertension were strongly positively correlated with stroke. These are the attributes that are used as features in our dataset.

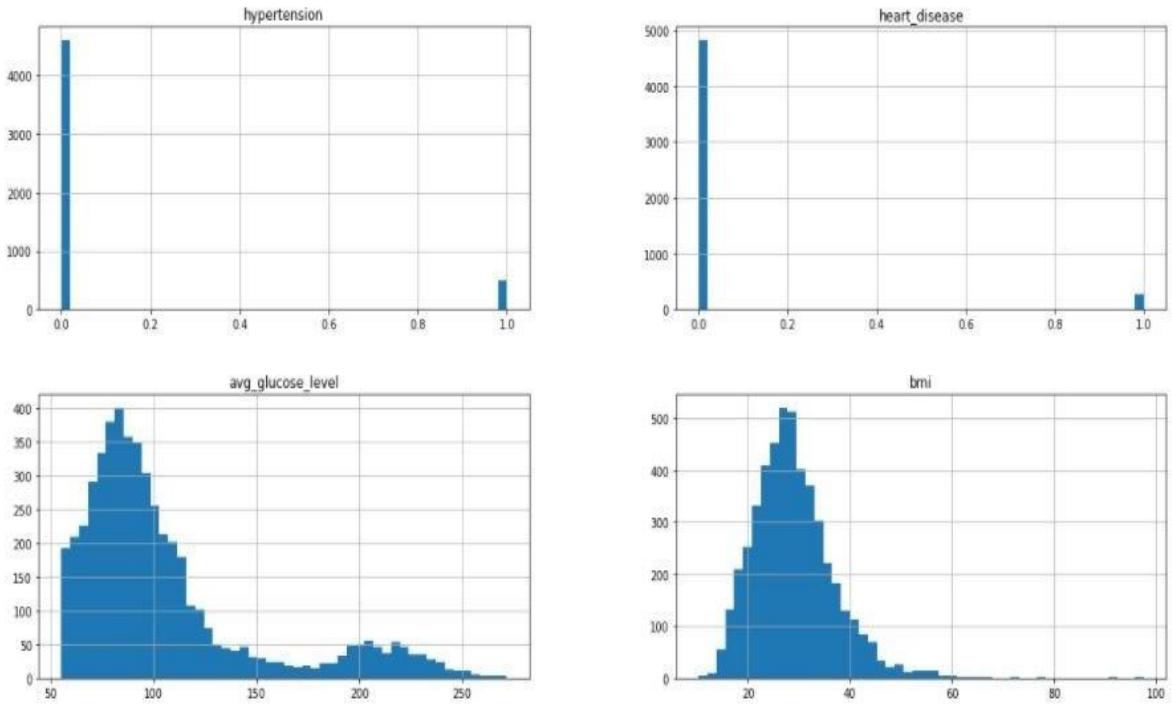


Figure 4.1.1 Important attributes of our dataset as a graphical representation

In figure 4.1.1, there are graphs of some essential attributes in our dataset like hypertension, heart disease, glucose level, and BMI that play a vital role in creating a situation for stroke. Here, hypertension and heart disease are shown as binary data. The high number of hypertension and heart disease cases in the "negative" column is because most values for stroke are negative in our dataset. Plotting glucose level and BMI also shows the rise and fall of their values with their corresponding number of rows or values.

4.2 Train-Test splitting

Train-Test splitting of the dataset is another critical step for creating a machine learning model. This is fixed to train our model with a part of our dataset and then test with another part of our dataset. Typically, train-Test splitting is done at 80% and 20% ratios. The train set will comprise 80% data, and the test set will comprise 20% data of the same dataset. We train our model with the train set. Then we provide our test dataset to model to predict the outcomes. The main idea for separating the train-test is how our model will work against data that it had not learned.

4.3 Stratified Sampling

It seems pretty logical and straightforward that to train-test split our data. We can just take 80% of it and keep the 20% to perform the prediction test later. However, we need to understand that we cannot just randomly take a chunk from the dataset. We need to make sure that the data we are splitting into two parts. These two parts should represent the whole dataset, including the variation and noises [16]. If we randomly take our train set, it may perform well in few situations, but it will fail in most situations because it may not have seen any other types or variations present in the dataset. If we train our model based on an arbitrarily chosen part from the dataset, it will not taste variations. So, the essence of stratified sampling is to split our dataset to contain a pattern of all possible variations among the attributes with their corresponding target or label. Now we can sample this based on attributes. In our case, we are performing binary classification, which means our model will predict only in terms of “yes” or “no.” So, we will want to distribute our positive or negative outcomes at the same ratio in both train and test datasets. Moreover, that is what we did in our work. We divided it into the same ratio.

4.4 Model Training

After pre-processing and train-test splitting, the most awaited step is creating our model with different algorithms. Multiple ML algorithms are available to work with. However, we will perform binary classification, so we will work with a few chosen algorithms that have been proven efficient for this type of classification. Logistic regression, Decision Tree, SVM,

XGBoost Classifier, Naïve Bayes are the few algorithms that perform well with binary classification [17].

4.5 Pipelining and feature scaling

This is a small step that has to be done to train our model. We create a pipeline and scale up our attributes (excluding labels) to fit into the model. We scaled up features using the standardization method where the values of our features will remain in a particular range and will be easier for an algorithm to train on. ”Pipeline” and “fit_transform” are the two methods we used for pipelining, which is again an efficient tool of the Scikit-learn framework.

4.6 Evaluation of models

At the training phase, we created models with the algorithms mentioned above. So, here our idea of choosing a model was based on only the RMSE method. It is a well-known method for evaluation. So, the lower the RMSE value, the better the performance. So, we chose the lowest RMSE model among them. Logistic regression (RMSE-0.22) and Decision Tree (0.30) performed better based on RMSE. We tested our model against the test dataset in the next phase, which was completely unknown to the model. For both of the models, RMSE was lower, and accuracy was higher (92%+). However, the problem arose when we checked the prediction outcomes. It was all “negative” in terms of outcome, which is not acceptable at all. The accuracy

may be higher, but it is not ideal for creating a model that only predicts one type of outcome. We cannot use logistic or Decision Tree Regression despite having a lower RMSE and better accuracy. Now in search of better performance, we applied the XGboost Classifier algorithm. It is a decision-tree-based ensemble Machine Learning algorithm that uses gradient boosting.

In prediction problems and artificial networks, it tends to outperform all other algorithms or frameworks. This algorithm worked well on prediction. Furthermore, especially the outcome, we desired that it should predict both types of outcome. This model not only predicted both types of values as an outcome but also provided a higher accuracy. We achieved the highest optimal parameter for prediction by tweaking its model parameters, where it learned profoundly and broadly as much as possible.

4.7 Attribute combination and dimension Reduction

In order to increase accuracy with more variation of correct output, we tried our attribute combination and dimension reduction.

Attribute Combination:

We can know exactly which attributes will impact strongly to predict the output by coefficient correlation. We took all the strongly correlated attributes to make combinations among them to create a more powerful attribute. Not only that, but we created the combination by addition and multiplication. In our case, blood pressure, BMI, and glucose level were strongly correlated with output.

Dimension Reduction:

We dropped some rows and attributes to reduce unnecessary data that is meaningless. We dropped the smoking attribute as it contained some faulty and missing values. Then we again trained our model, which gave us an accuracy of 94.91%. Something we also tried by reducing the row. But when we reduced rows based on statistical analysis, we failed to achieve any good accuracy. As a result of row reduction, our model could not learn the pattern of the system and ended with poor performance, So we decided not to reduce any row. We just canceled the smoking parameter.

4.8 Parameter Tuning

Parameter Tuning is done by changing the model's internal default parameter values to achieve optimized accuracy. We changed our model's parameter, which is n_estimator. We tried out different values of n_estimators by using loop and assigned values for this parameter dynamically. Furthermore, we tried these from 10 to 50000. We got a maximum accuracy of 94.5% with variation in output. We got this accuracy and increased prediction results when n_estimators = 1000. However, we got better accuracy than this. Nevertheless, those models were producing biased output. We had to pay attention that our model is better at predicting both

kinds of binary values. So, for n_estimators =1000 and learning_rate = 0.3, we got the best accuracy and improved results.

4.9 Export the model

After the end ML part, we needed to export our final model to use it in a web application. We chose our model and tuned its parameters for better results. Now it is time to use our model. We used joblib, which is a popular package of python, to make the object of this model. This object will now work as a portable version of our model. We can now integrate our files into our web application and help us make predictions from outside.

4.10 Design The Database

We are using Django's ORM to create our database. This is Django's default architecture of the model. We have to set up our database for patients and their corresponding datasets. We will need three tables for database setup.

1. Patient Information
2. Dataset that patient will provide to predict

3. Prediction Information

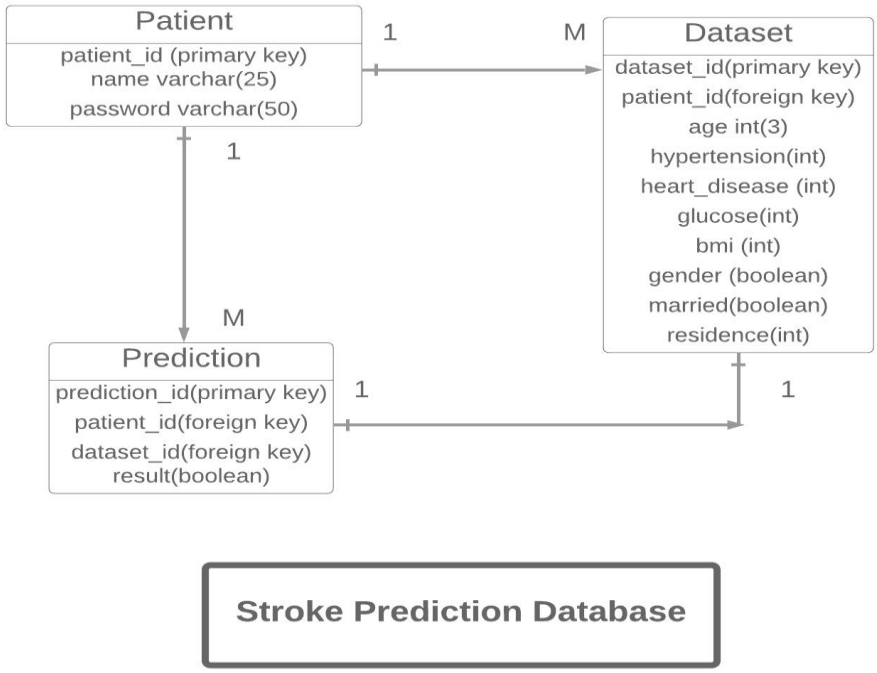


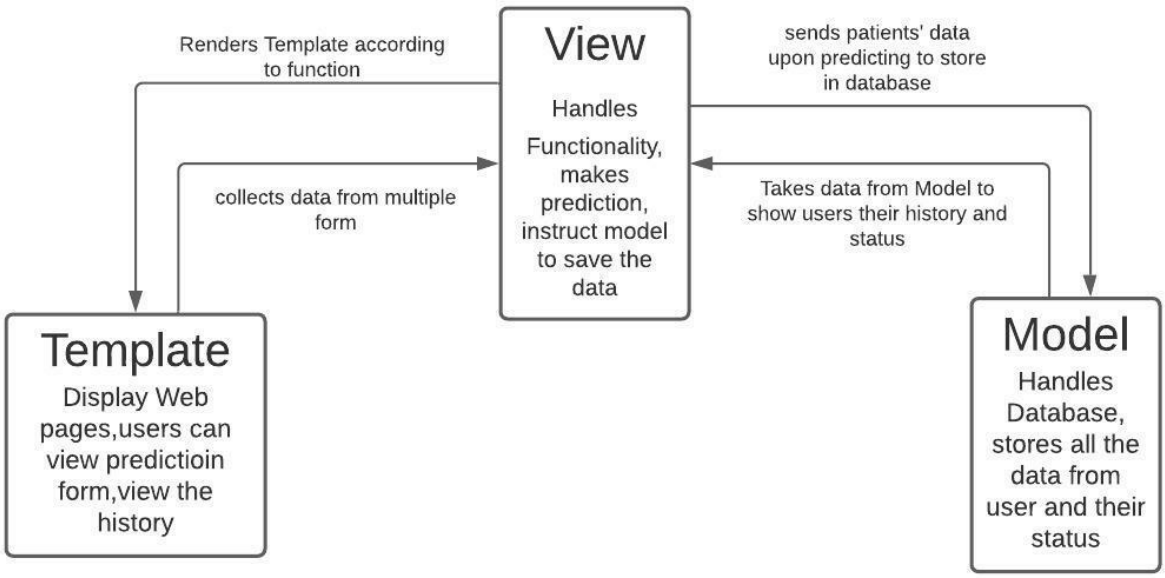
Figure 4.10.1: ER diagram of our web application's database

In figure 4.10.1, these are the required three tables of our website. The patient and the dataset have one too many relationships here. As one patient can have multiple datasets, but a single dataset should only belong to one patient. For this same reason, there are one too many relationships between patient and prediction. Lastly, there is a one-to-one relationship between dataset and prediction because one dataset will be responsible for only one prediction, and one prediction result will come from one dataset only.

4.11 Basic Signup, Sign in and Sign out functionality

Before making any predictions, users have to first sign up and sign in like any other website. Users can sign out after making predictions and viewing their data. Django provides all these functionalities by default. So, we will be using the default methods of Django to implement these features.

4.12 Backend functionality to make predictions and store the data



Complete flow of our web application with django's architecture

Figure 4.12.1:Complete web flow of website with Django's architecture

Django's MVT architecture helped us to build our desired website fast and securely. In figure 4.12.1, we can see how our website is working with the important components of Django. Model, view, and template are used to run different parts of the system. View component is triggering the template component to render necessary HTML files. The template is the only component that will be directly used or viewed by users. After receiving the data from the user view component, it will implement its business logic to predict and store the values. The pickled object that we used to export our final model will reside in the view component. The view component's responsibility is to predict with that pickled file and send the prediction result to the template component. It will also save the dataset to the model which handles our database. Once the values are saved, users can view their inserted value and the corresponding result on the view history page. When the user requests the history, the view component will retrieve all the data associated with that user ID from the model. It will be shown by rendering the assigned HTML page, which is the sole responsibility of the template component.

4.13 Suggest some advice upon prediction

We already figured out our strongly correlated data with the help of correlation. Blood pressure, BMI, and glucose level are the important factors that are responsible for a stroke. We separated both stroke and non-stroke patients to find out the mean and standard deviation of these particular attributes. So, we analyzed our dataset to find out the actual values that are responsible

for a stroke. We find out that those with glucose levels more than 132, BMI over 25, and high blood pressure are vulnerable. Suppose a patient inputs abnormal values for these attributes while predicting. We prompt him/her with a warning message to take the necessary steps to keep these aspects normal. With a prediction result, these suggestions will undoubtedly help a patient to prevent a deadly stroke.

Chapter 5

Result and Analysis

5.1 Project Illustration

The main goal of our prediction system predicts the right outcome efficiently. Now, in terms of illustration or the usage of our project, it is necessary to represent it in a user-friendly way. From the user's perspective, our patient will just have to input the necessary parameters. These parameters are the attributes or features of our dataset, which we deeply analyzed for creating this system. So, after inputting the necessary parameters, the system will give a result as an outcome. The result will be positive or negative. As a result, our patients will know if they risk having a stroke in the future.

5.2 System Design

Here is a simple overview of our system design. Our prediction system will be consisting of typical machine learning procedures. Data pre-processing, train-test splitting, pipelining, model training, evaluation of models, finally choosing the most reliable and accurate one. In data preprocessing, we figured coefficient correlation and visualized them, stratified sampling for

train-test split, standardization for pipelining, and feature scaling. RMSE, confusion matrix, accuracy as an evaluation method. Different types of classification algorithms for our model.

In terms of tools, we used pandas, NumPy, Matplotlib for data analysis and visualization. We used the Scikit-learn framework for applying different classification and evaluation methods.

5.3 Result

In the training phase, we tried out five algorithms. In table 5.0.1, here the algorithms and their corresponding RMSE are given.

Logistic Regression	Decision Tree Regression	KNN	Naive Bayes	SVM
0.220	0.301	0.307	0.37	0.22

Table 5.0.1: Different models with their corresponding RMSE value

We created different models with these algorithms. For evaluation, we chose the RMSE method. Here we decided to choose the model with low RMSE. However, all of the model's RMSE was low. We chose a Logistic and Decision Tree. SVM's RMSE is also low and similar to logistic regression. These two algorithms are prevalent for most machine learning problems, and they

tend to work better in any situation [18]. Moreover, the main reason we took these two algorithms was that the RMSE was low.

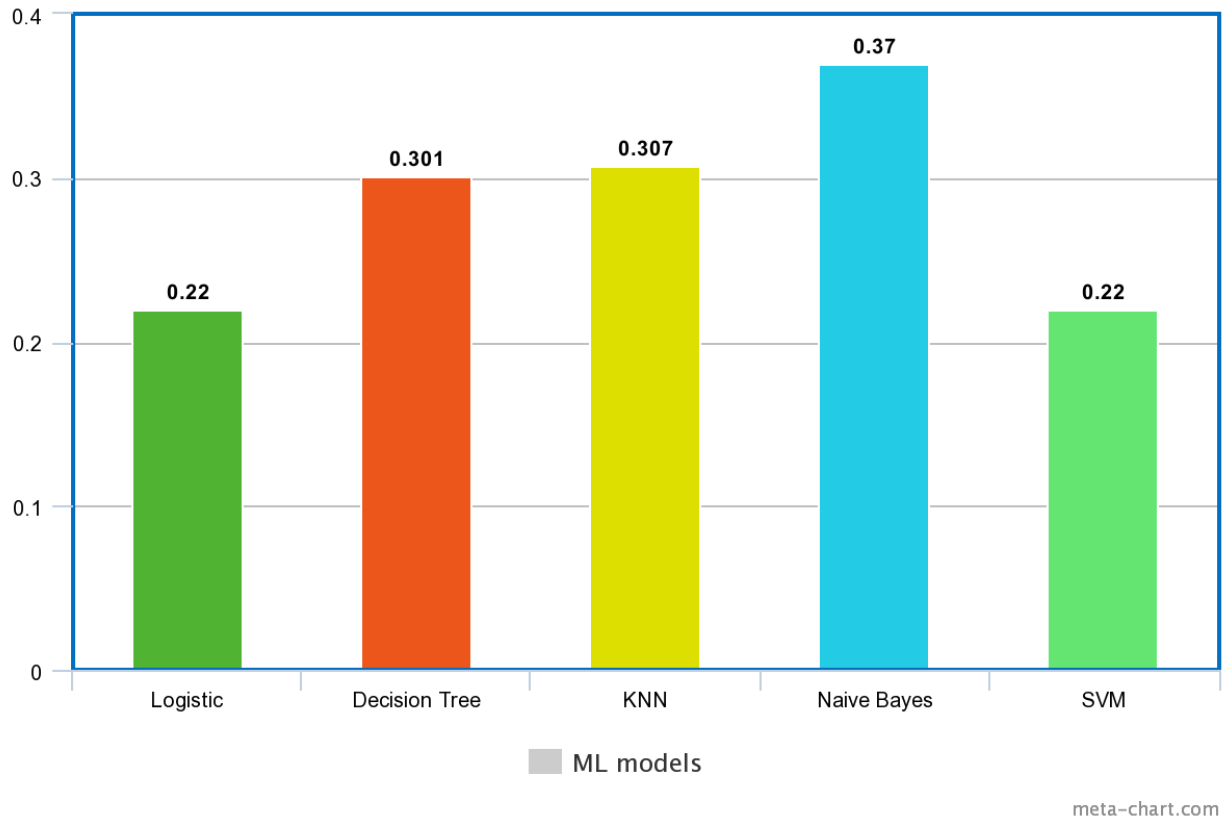


Figure:5.0.1 Visualization of the different models with their corresponding RMSE values

In figure 5.0.1: we can visualize the RMSE values. It shows how one model's RMSE differs from others. Although the difference is subtle, our primary goal is to go with the lowest RMSE.

Now in the prediction phase with the test dataset, we tested with these two models. Their accuracy and RMSE are given.

	Logistic Regression	Decision Tree Regression
Accuracy	95%	94%
RMSE	0.221	0.306

Table 5.0.2: Accuracy and RMSE of Logistic Regression and Decision Tree Regression

In table 5.0.2, we can see our accuracy and RMSE for both models, which are excellent. Our workflow was acceptable to this point, but problems arose when we found out the confusion matrix. The confusion matrix simply tells us the number of right and wrong predictions of positive and negative values. Here it returned no prediction for positive values. It predicted only negative outcomes. This problem occurred because our dataset was imbalanced. The majority of the values were negative. So, more negative values and algorithms' working principle was heavily biased to produce only one type of outcome.

```
confusion_matrix(Y_test,final_predictions)
```

```
array([[972,   0],
       [ 50,   0]], dtype=int64)
```

Figure 5.0.2: Confusion matrix for logistic regression

The confusion matrix of logistic regression here in figure 5.0.2 shows only true negative and false negative values. There are no positive values for not even false positives in the prediction.

```
: confusion_matrix(Y_test,final_predictions)

: array([[970,   2],
         [ 50,   0]], dtype=int64)
```

Figure 5.0.3: Confusion matrix for decision tree regression

In figure 5.0.3, we observe a similar outcome for decision tree regression, just like logistic regression. There are only negative predictions and only two false positives.

Now we need a different approach to solve this problem. We either hyper-tune the parameters of these algorithms that we created our model with or select a more efficient and modern algorithm like GBM, XGBoost, or lightGBM. We went with the second option. We decided to apply the XGBoost algorithm to create our model. Its working principle is far better and more reliable than other algorithms [19]. We got this result after creating a model with XGBoost and testing it with the test dataset.

```
Accuracy:  0.9373776908023483
array([[950,  22],
       [ 42,   8]], dtype=int64)
```

Figure 5.0.4: Confusion matrix and accuracy for XGBoost

Figure 5.0.4 confusion matrix of XGBoost represents a perfect and efficient result despite having a lower accuracy than logistic and decision tree regression. There are both positive and negative predictions rather than producing one type of outcome. Here we have 22 false positives and eight true positive values.

With XGBoost, we got our desired output which logistic or Decision Tree Regression failed to achieve. From the confusion matrix, we can see that it predicted the outcomes in a progressive manner. It also gave us great accuracy.

The reason XGBoost worked so better than other algorithms is because of its working principle and parameters. It is a decision-tree-based ensemble Machine Learning algorithm that uses a gradient boosting mechanism. Three forms of gradient boosting support it. Gradient boosting, Stochastic gradient boosting, and Regularized gradient boosting [20]. The primary two parameters that play an essential role in XGBoost are n-estimators and learning rate.

We tried with different values of the two parameters mentioned above and reached an optimal point. Our chosen n-estimator is 50000, and the learning rate is 0.4, which produced an optimal solution. The effectiveness of XGBoost and the working principle is way more potent than other

algorithms we tried. So, we will choose XGBoost Classifier as our final model for a robust result and absolute reliability.

Attribute Combination Performance

We combined our strongly correlated datasets to create some more strong attributes that will help us produce better accuracy.

```
Accuracy:  0.949119373776908
array([[968,   4],
       [ 48,   2]])
```

Figure 5.0.5: Accuracy and Confusion matrix after combining attributes

In figure 5.0.5, we can see accuracy and confusion matrix after attribute combination. It gave us a nice accuracy of 94.91%, but the predicted results are not that satisfactory. It produced only 2 true positive outcomes from the system.

Dimension Reduction Performance

```
Accuracy:  0.62426614481409
array([[596, 376],
       [  8,  42]])
```

Figure 5.0.6: Accuracy and Confusion matrix after dimension reduction

We tried to create our model by reducing some low impactful rows from the dataset. In figure 5.0.6, it is visible that the accuracy we got is not acceptable at all.

Parameter tuning performance

```
Accuracy:  0.9452054794520548
array([[957,  15],
       [ 41,   9]])
```

Figure 5.0.7: Accuracy and Confusion matrix after parameter tuning

In figure 5.0.7, we can see that both accuracy and confusion matrix are satisfactory. We got an excellent accuracy of 94.52% and an acceptable amount of positive and negative outcomes. We got 957 true negatives and nine true positives with an increased accuracy which is far better than our previous outcomes.For n_estimator = 1000 and learning_rate = 0.3 we achieved this outcome.

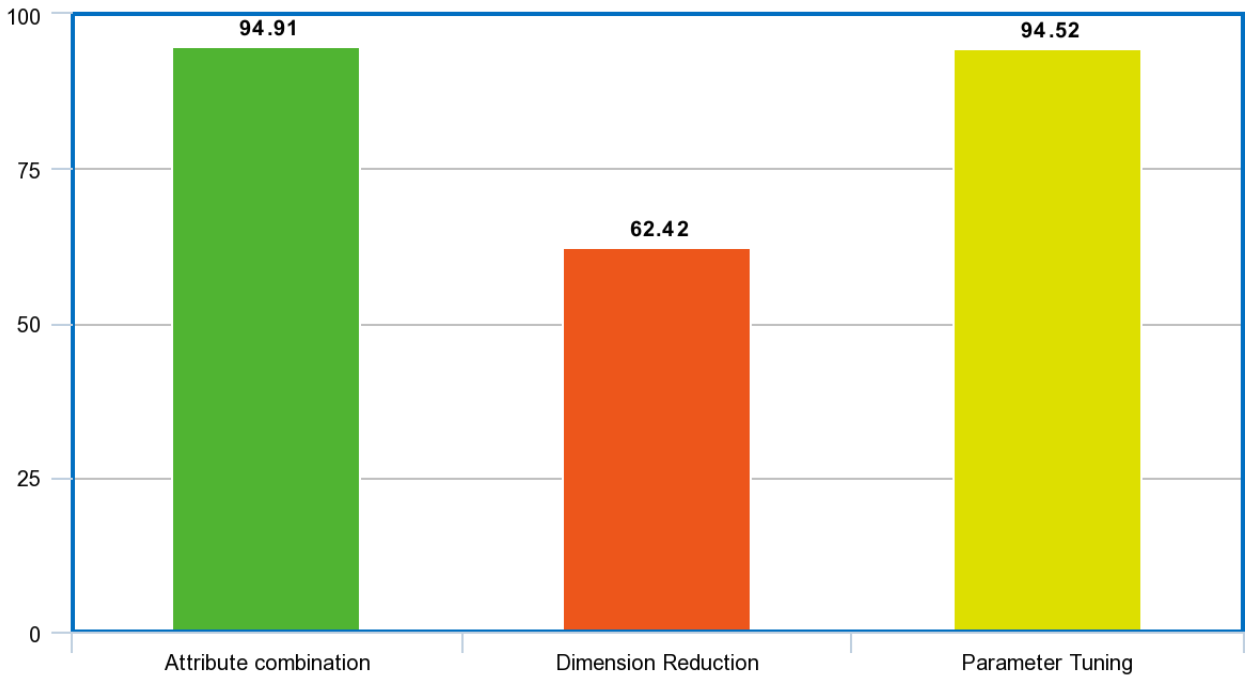


Figure 5.0.8: Visual Representation of optimizing methods' accuracy

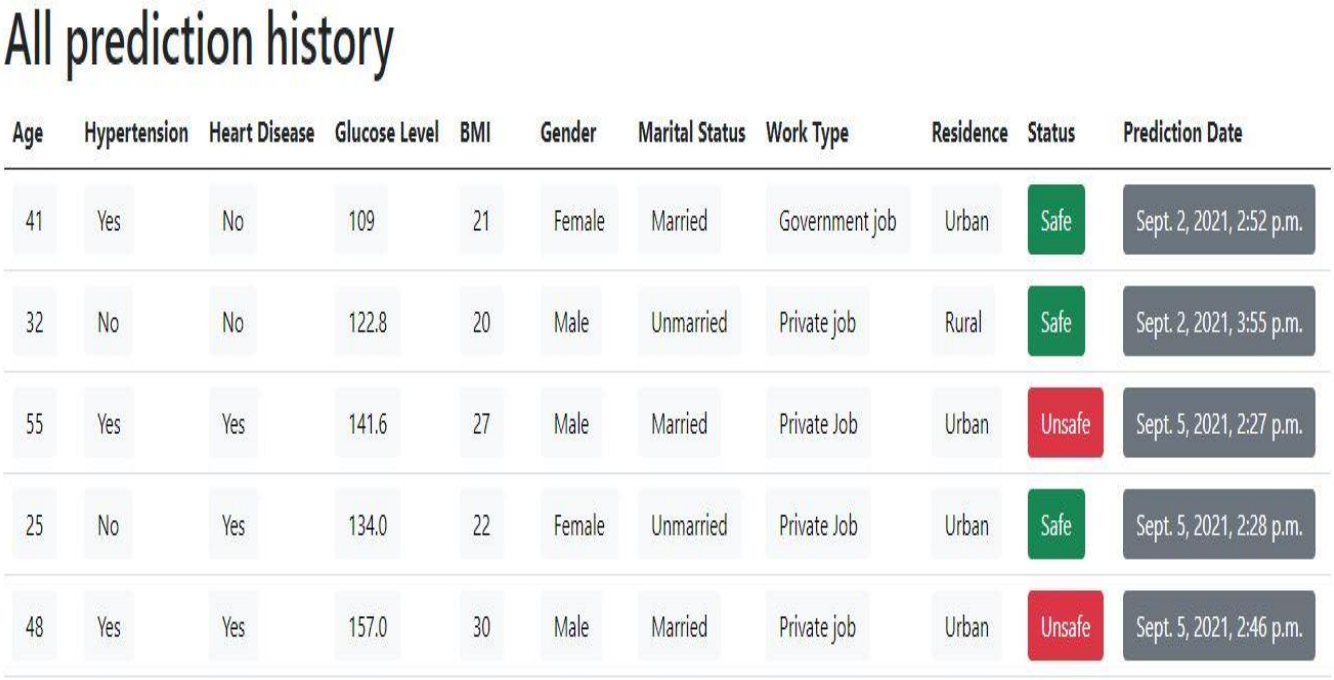
In Figure 5.0.8, the bar chart shows the dimension reduction method provides an inferior performance compared to the other two methods. Attribute combination and Parameter tuning both performed better. Now along with accuracy, we will evaluate these methods in terms of the right number of outcomes as well with the help of a confusion matrix.

	Accuracy	**Correct prediction count from the confusion matrix**
Attribute Combination	94.91%	968 true negative and 2 true positive
Dimension Reduction	62.42%	596 true negative and 42 true positive
Parameter Tuning	94.52%	957 true negative and 9 true positive

Table 5.0.3: Accuracy and correct prediction count after model creation

From table 5.0.3, we can observe that in terms of accuracy attribute combination method wins. It has an accuracy of 94.91%. The parameter tuning method also gives an accuracy of 94.52 %, which is slightly lower than the attribute combination. On the other hand, in terms of correct prediction count, the parameter tuning method is better. It predicted 957 true negatives and nine

true positive values, where the attribute combination method predicted 968 true negatives and two true positives. So, parameter tuning gave us a better result than other methods.



All prediction history

Age	Hypertension	Heart Disease	Glucose Level	BMI	Gender	Marital Status	Work Type	Residence	Status	Prediction Date
41	Yes	No	109	21	Female	Married	Government job	Urban	Safe	Sept. 2, 2021, 2:52 p.m.
32	No	No	122.8	20	Male	Unmarried	Private job	Rural	Safe	Sept. 2, 2021, 3:55 p.m.
55	Yes	Yes	141.6	27	Male	Married	Private Job	Urban	Unsafe	Sept. 5, 2021, 2:27 p.m.
25	No	Yes	134.0	22	Female	Unmarried	Private Job	Urban	Safe	Sept. 5, 2021, 2:28 p.m.
48	Yes	Yes	157.0	30	Male	Married	Private job	Urban	Unsafe	Sept. 5, 2021, 2:46 p.m.

Figure 5.0.9 Prediction history page of a patient

Prediction history is shown in figure 5.0.9. This page shows all the input attributes and their corresponding status or prediction. It also shows the date when the user made his/her prediction. With this history, the user can easily track his/her health status from time to time.

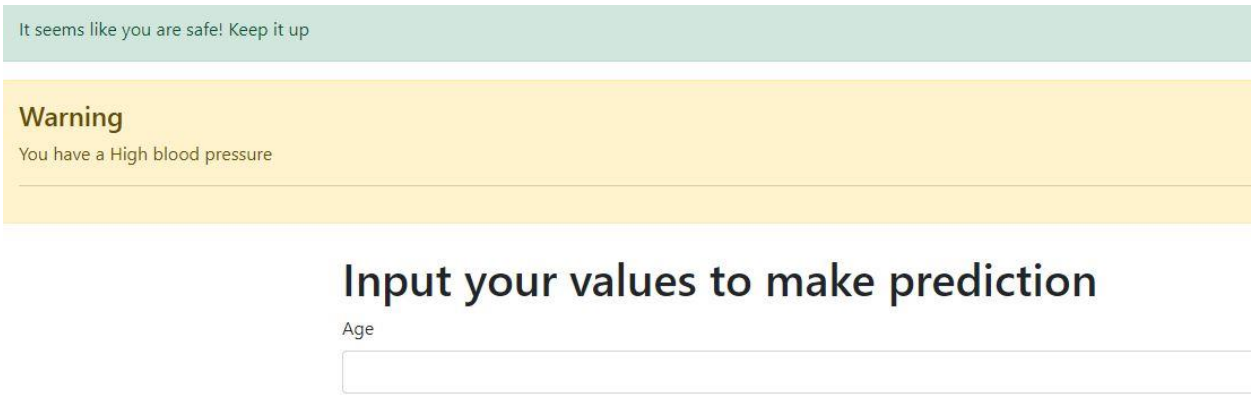


Figure 5.0.10 Displaying results and suggestions upon making a prediction

This is what the user will see after predicting, as shown in figure 5.0.10. After putting all the attributes, the system will prompt the user with a message if he/she has any chance of having a stroke. Then the system will recommend some suggestions based on what the user has input. If there is an abnormality in his/her attributes, it will be shown with a warning message.

5.4 Summary

Among the three models which we tried against our test dataset, XGBoost performed well. It predicted well in terms of both accuracies and variations of the outcome. With this model, we got an accuracy of 93.7%. It also predicted different possible outcomes. This can assure us that in the

future, it will be giving correct predictions for individual users. After performing post-model creation optimization, our accuracy increased from 93.7% to 94.5%. Our web application is also working fine, providing accurate results as expected.

We can now be assured that our work is done with total efficiency. Here is a comparison table of the group members and their contributions.

Eshan Ahmed	Redwan Ahmed	Shahriar Hossain
Dataset collection, preprocessing, pipeline and feature scaling, Training models with the algorithm, Testing with the dataset, evaluation, model optimization with different methods, analysis of data for the suggestion, database design, login, registration, logout, displaying data on the history page, adding suggestion feature, connecting model with backend by joblib and other backend functionalities.	Dataset Collection, model training, frontend(login, registration, homepage)	Dataset collection, user interactive prediction form, converting numeric values to actual value in history page, suggestion feature

Chapter 6
Conclusion

6.1 Summary of Work

In our project, we will start working with a dataset. Therefore, the first task is to acquire the data which we got from Kaggle [21]. We also analyzed the data, applying factorization, correlation, and graphical representation of data. After that, we divided the data for testing and training our model. Our most important work started when we had to choose a suitable algorithm for our model. After successfully evaluating the model, we will integrate our model into a web platform. So, users can quickly get a simple interface to test their possibility of stroke.

6.2 Project Development Phases

Altogether there are six stages in our project. The first stage was data collection which was gathered from actual patient information found in Kaggle. After that, we started processing and visualizing the data using Pandas, NumPy, and Matplotlib. After completing those tasks, our data was ready for work, so we initiated train test splitting, dividing the data for training and testing our model. Next, we started Stratified Sampling to eliminate sampling bias in a test set. Our next

task was pipelining and Feature scaling. We created various models using different algorithms and evaluated the results using the RMSE scores. After that, we created our final model with XGBoost and optimized it to achieve the best accuracy. Once our model was optimized, we integrated it into the Django web application. So, users can make predictions directly from the website. We also made our website users convenient to make their experience hassle-free.

6.3 Ethical and Professional Responsibilities

It is easy to understand that nearly every act or thought can be taken as both moral and immoral at the same time. However, when we work with medical data, we should respect and protect the confidential data of every patient. We should give them enough privacy to use their data in medical technology. When we collect data from any patient, We should take their consent and make them aware of which data we are using. They should know the risk of disclosing any confidential information like personal health information. So we only take the health data and do not have their name or address in our dataset.

6.4 Environmental impact

Our project might not have a direct impact on the environment. However, we are indirectly trying to impact by helping people identify stroke in early stages, which helps the environment. Healthy people make up a good environment. Therefore the goal of our work is to ensure the safety and wellbeing of the people and their environment. Imagine a world where the stroke is not one of the leading causes of death. Not everyone is aware of the risks and fatalities of stroke.

It may be something that people do not care about, but anyone can have a stroke at any point in their life. Our work can give accurate predictions of the chances of stroke and we can help people who are not aware of it. We want our work to impact these and help people live a stroke-free life in a safe and healthy environment.

6.5 Project Sustainability

Our system can help people as well as doctors in dealing with stroke. As we worked with real-life patient data, we are confident that our model can accurately predict stroke chances. With capabilities like this, our work is sustainable and is usable by patients as well as doctors. We can increase scalability by adding more data into our system, which will help train our model better and increase the accuracy of our predictions. We can expand our work by working closely with doctors and experts who will surely benefit from the information we provide. This project will be sustainable in the long run as long as stroke disease is one of the deadliest diseases in the world.

References

[1].Cleveland Clinic. 2021. Stroke: Causes, Prevention. [online] Available at: https://my.clevelandclinic.org/health/diseases/5601-stroke-understanding-stroke.

[2].Miguel Monteiro, Ana Catarina Fonseca, Ana Teresa Freitas, Teresa Pinho e Melo, Alexandre P. Francisco, Jose M. Ferro, and Arlindo L. Oliveira, Using Machine Learning to Improve the Prediction of Functional Outcome in Ischemic Stroke Patients. https://doi.org/10.1109/TCBB.2018.2811471

[3].Hamidreza Saber, Melek Somai, Gary B. Rajah, Fabien Scalzo & David S. Liebeskind.Predictive analytics and machine learning in stroke and neurovascular medicine.https://doi.org/10.1080/01616412.2019.1609159

[4].Hamann Janne , Herzog Lisa, Wehrli Carina, Dobrocky Tomas, Bink Andrea, Piccirelli Marco, Panos Leonidas, Kaesmacher Johannes, Fischer , Stippich Christoph4 , Luft Andreas , Grall Jan , Arnold Marcel, West Roland , Sick Beate, egener Susanne. Machine learning-based outcome prediction in stroke patients with MCA-M1 occlusions and early thrombectomy.DOI:

10.1111/ene.14651

[5].Hendrikus J. A. van Os, Lucas A. Ramos, Adam Hilbert , Matthijs van Leeuwen , Marianne A. A. van Walderveen , Nyika D. Kruyt , Diederik W. J. Dippel , Ewout W. Steyerberg, Irene C.

van der Schaaf, Hester F. Lingsma, Wouter J. Schonewille, Charles B. L. M. Majoie, Silvia D. Olabarriaga , Koos H. Zwinderman, Esmee Venema, Henk A. Marquering, Marieke J. H. Wermer.Predicting Outcome of Endovascular Treatment for Acute Ischemic Stroke: Potential Value of Machine Learning Algorithms.doi:10.3389/fneur.2018.00784

[6]. Yafei Wu and Ya Fang.Stroke Prediction with Machine Learning Methods among Older Chinese.doi:10.3390/ijerph17061828

[7]. Ching-Heng Lin, Kai-Cheng Hsu, Kory R. Johnson, Yang C. Fann, Chon-Haw Tsai, Yu Sun, Li-Ming, Lien, Wei-Lun Chang, Po-Lin Chen, Cheng-Li Lin, Chung Y. Hsu.Evaluation of Machine Learning Methods to Stroke Outcome Prediction Using a Nationwide Disease Registry.doi:10.1016/j.cmpb.2020.105381

[8] Athensjournals.gr, 2021. [Online]. Available: https://www.athensjournals.gr/sciences/2020-7-1-4-Stigler.pdf. [Accessed: 23- Sep- 2021].

[9]Jusst.org, 2021. [Online]. Available: https://jusst.org/wp-content/uploads/2021/06/Django-Website-for-Disease-Prediction-using-Machine-Learning.pdf. [Accessed: 23- Sep- 2021].

[10] Jakobs, D. (2019). „BEFORE WE UNDERSTAND WHAT WE ARE DOING, WE NEED TO KNOW HOW WE THINK". Protection Of Cultural Heritage, (8), 139-150. https://doi.org/10.35784/odk.1075

[11] White, S., & Ivie, R. (2010). What Can We Learn about Diversity from Statistics on Acoustics?. Acoustics Today, 6(4), 7. https://doi.org/10.1121/1.3533341

[12] Xu, J., Liu, J., Yin, J., & Sun, C. (2016). A multi-label feature extraction algorithm via maximizing feature variance and feature-label dependence simultaneously. Knowledge-Based Systems, 98, 172-184. https://doi.org/10.1016/j.knosys.2016.01.032

[13]"Simple Math About XGBoost", Medium, 2021. [Online]. Available: https://towardsdatascience.com/simple-math-about-xgboost-b9a924aaae9f#:~:text=In%20XGBoost%2C%20it%20uses%20the,%3A%20(g1%2Bg2%2B%E2%80%A6. [Accessed: 23- Sep- 2021].

[14]. Jamshidian, M., & Bentler, P. (1999). ML Estimation of Mean and Covariance Structures with Missing Data Using Complete Data Routines. Journal Of Educational And Behavioral Statistics, 24(1), 21. https://doi.org/10.2307/1165260

[15] Verboven, S., Hubert, M., & Goos, P. (2012). Robust preprocessing and model selection for spectral data. Journal Of Chemometrics, 26(6), 282-289. https://doi.org/10.1002/cem.2446.

[16]. Mehare, D. (2018). Introduction to TF-IDF: To Represent Importance of Keyword within the whole Dataset. International Journal For Research In Applied Science And Engineering Technology, 6(3), 2321-2323. https://doi.org/10.22214/ijraset.2018.3369

[17] S A, S. (2021). Comparative Study of Naive Bayes, Gaussian Naive Bayes Classifier and Decision Tree Algorithms for Prediction of Heart Diseases. International Journal For Research In Applied Science And Engineering Technology, 9(3), 475-486. https://doi.org/10.22214/ijraset.2021.33228

[18] Di Porto, F. (2020). Creating Better Disclosure Norms through Machine Learning Algorithms. SSRN Electronic Journal. https://doi.org/10.2139/ssrn.3705967

[19] XGBOOST?, W., & Khandelwal, P. (2021). Light GBM vs XGBOOST: Which algorithm takes the crown. Analytics Vidhya. Retrieved 23 May 2021, from https://www.analyticsvidhya.com/blog/2017/06/which-algorithm-takes-the-crown-light-gbm-vs-xgboost/

[20] Danandeh Mehr, A. (2021). Drought classification using gradient boosting decision tree. ActaGeophysica.https://doi.org/10.1007/s11600-021-00584-8

[21]"Stroke Prediction Dataset", Kaggle.com, 2021. [Online]. Available: https://www.kaggle.com/fedesoriano/stroke-prediction-dataset. [Accessed: 23- Sep- 2021].