Analyzing the Performance of Stroke Prediction using ML Classification Algorithms

Gangavarapu Sailasya[1], Gorli L Aruna Kumari[2]
Department of Computer Science and Engineering
GITAM Institute of Technology, GITAM (Deemed to be University)
Visakhapatnam, Andhra Pradesh – 530045

***Abstract*—A Stroke is a health condition that causes damage by tearing the blood vessels in the brain. It can also occur when there is a halt in the blood flow and other nutrients to the brain. According to the World Health Organization (WHO), stroke is the leading cause of death and disability globally. Most of the work has been carried out on the prediction of heart stroke but very few works show the risk of a brain stroke. With this thought, various machine learning models are built to predict the possibility of stroke in the brain. This paper has taken various physiological factors and used machine learning algorithms like Logistic Regression, Decision Tree Classification, Random Forest Classification, K-Nearest Neighbors, Support Vector Machine and Naïve Bayes Classification to train five different models for accurate prediction. The algorithm that best performed this task is Naïve Bayes that gave an accuracy of approximately 82%.**

Keywords*—Stroke; machine learning; logistic regression; decision tree classification; random forest classification; k-nearest neighbors; support vector machine; Naïve Bayes classification

I. Introduction

According to the Centers for Disease Control and Prevention (CDC), stroke is the fifth-leading cause [1] of death in the United States. Stroke is a non-communicable infection that is liable for around 11% of total deaths. Consistently, over 795,000 individuals in the United States experience the ill effects of a stroke [2]. It is the fourth significant reason for death in India.

With the advancement of technology in the medical field, predicting the occurrence of a stroke can be made using Machine Learning. The algorithms present in Machine Learning are constructive in making an accurate prediction and give correct analysis. The works previously performed on stroke mostly include the ones on Heart stroke prediction. Very less works have been performed on Brain stroke. This paper is based on predicting the occurrence of a brain stroke using Machine Learning. The key components of the approaches used and results obtained are that among the five different classification algorithms used Naïve Bayes has best performed obtaining a higher accuracy metric. The limitation with this model is that it is being trained on textual data and not on real time brain images. The paper shows the implementation of six Machine Learning classification algorithms. This paper can be further extended to implementing all the current machine learning algorithms.

A dataset is chosen from Kaggle [3] with various physiological traits as its attributes to proceed with this task. These traits are later analyzed and used for the final prediction. The dataset is initially cleaned and made ready for the machine learning model to understand. This step is called Data Preprocessing. For this, the dataset is checked for null values and fill them. Then Label encoding is performed to convert string values into integers followed by one-hot encoding, if necessary.

After Data Preprocessing, the dataset is split into train and test data. A model is then built using this new data using various Classification Algorithms. Accuracy is calculated for all these algorithms and compared to get the best-trained model for prediction.

After training the model and calculating the accuracy, an HTML page and a Flask application are developed. The web application is for the user to enter the values for prediction. The flask application is a framework that connects the trained model and the web application. After proper analysis, the paper concludes which algorithm is most appropriate for the prediction of stroke.

II. Literature Survey

In [4], stroke prediction was made on Cardiovascular Health Study (CHS) dataset using five machine learning techniques. As an optimal solution, the authors used a combination of the Decision Tree with the C4.5 algorithm, Principal Component Analysis, Artificial Neural Networks, and Support Vector Machine. But the CHS Dataset taken for this work had a smaller number of input parameters.

In [5], stroke prediction has been carried out from the social media posts posted by people. In this particular work, the authors have used the DRFS method to find the various symptoms associated with stroke disease. The usage of Natural Language Processing to extract the text from the social media posts adds up to the overall execution time of the model which is not desirable.

In [6], the authors have performed the task of stroke prediction by using an improvised random forest algorithm. This was used to analyze the levels of risks obtained with the strokes. As suggested by the authors, this method is said to have performed better when compared to the existing algorithms. This particular research is limited to very few types of strokes and cannot be used for any new stroke type in the future.

Research paper [7] shows that the model was trained using Decision Tree, Random Forest, and Multi-layer perceptron for stroke prediction. The obtained accuracies for the three methods were quite close, with slight differences. The calculated accuracy for Decision Tree was 74.31%, Random Forest was 74.53%, and Multi-layer perceptron was 75.02%.

This paper suggests that Multi-layer perceptron is more accurate than the other two methods. Accuracy score was the only metric used for calculating the performance that might not always give favorable results.

Research carried out in [8] shows the implementation of machine learning model to predict heart stroke. They used various machine learning techniques like Decision tree, Naïve Bayes, SVM to build the model and later compared their performance. They obtained a maximum accuracy of 60% from the used algorithms which is pretty less.

In [9], the authors have used different data mining classification techniques to predict the possibility of a stroke. The dataset was taken from the Ministry of National Guards Health Affairs Hospitals, Kingdom of Saudi Arabia. The three classification algorithms used were C4.5, Jrip and Multi layers perceptron (MLP). With these algorithms, the model obtained an accuracy of around 95%. Even though the paper claims to obtain an accuracy of 95%, the time taken for training and predicting is higher as the authors have used a combination of complex algorithms.

Research carried out in [10], suggests the usage of three different algorithms to predict the possibility of stroke. These algorithms are Naïve Bayes, Decision Tree, and Neural Networks. This paper concluded that the Decision tree has the highest accuracy (about 75%) of the other two algorithms. But this model could not suit the real-world examples based on the values obtained from the confusion matrix.

In [11], the researchers have performed stroke prediction on Cardiovascular Health Study (CHS) dataset. They proposed a novel automatic feature selection algorithm that selects robust features based on their proposed conservative mean. They have combined this method with the Support Vector Machine algorithm for better efficiency. But this resulted in the generation of a number of vectors that tend to reduce the performance of the model.

Research in [12] proposes the prediction of thrombo-embolic stroke disease using Artificial Neural Networks. The method used for prediction was the Back-propagation algorithm. This model was able to get an accuracy of around 89%. But Neural Networks need more time to be trained and require higher processing time because of the complex structure with increasing number of neurons.

III. System Methodology

To proceed with the implementation, different datasets were considered from Kaggle. Out of all the existing datasets, an appropriate dataset was collected for model building.

After collecting the dataset, next step lies in preparing the dataset to make the data more clear and easily understood by the machine. This step is called as Data preprocessing. This step includes handling of missing values, handling imbalanced data and performing label encoding that are specific for this particular dataset.

Now that the data is preprocessed, it is ready for model building. For model building, preprocessed dataset along with machine learning algorithms are required. Logistic Regression, Decision Tree Classification algorithm, Random Forest Classification algorithm, K-Nearest Neighbor algorithm, Support Vector Classification and Naïve Bayes Classification algorithm are used. After building six different models, they are compared using five accuracy metrics namely Accuracy Score, Precision Score, Recall Score, F1 Score and Receiver Operating Characteristic (ROC) curve.

The comparison of the models gives the best model in terms of the accuracy metrics to proceed with the deployment phase. For deploying the model, an HTML page is developed to make it user-friendly for the user to enter the input parameters and get the result. The parameters entered by the user are sent to the model using a flask application which is basically a python framework that links the web application and the model together. The model takes the input parameters, predicts the output and returns the result to the flask application. Now this flask will display that result on the web page for the user to check the result.

The flow chart of the proposed system's methodology is in Fig. 1.

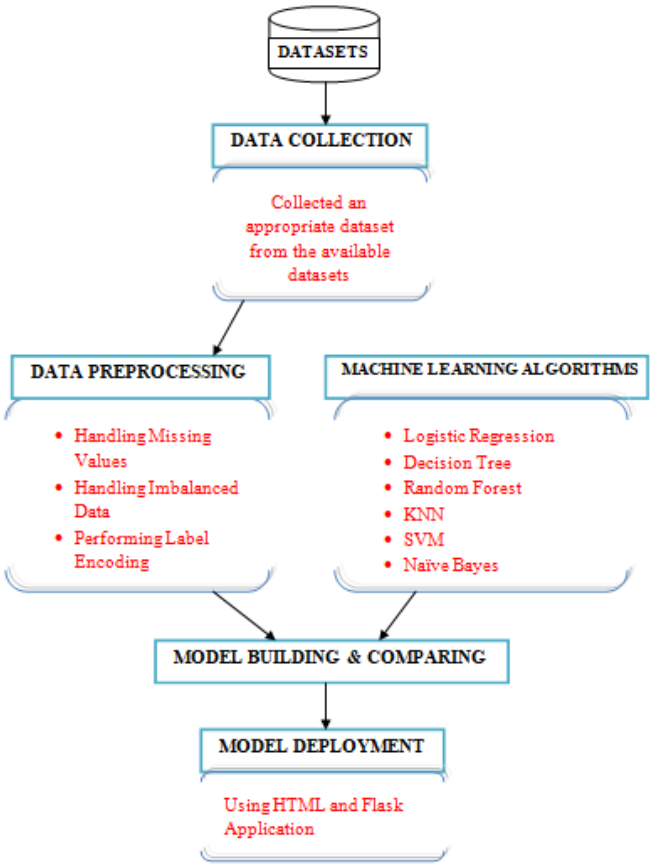

Fig. 1. Proposed System's Flow Chart.

IV. Implementation

The implementation of this project is as follows.

A. Dataset

The dataset for stroke prediction is from Kaggle [3]. This particular dataset has 5110 rows and 12 columns. The columns have'id','gender','age','hypertension',heart_disease','ever_married', 'work_type', 'Residence_type','avg_glucose_level', 'bmi', 'smoking_status' and 'stroke' as the main attributes. The output column 'stroke' has the value as either '1' or '0'. The value '0' indicates no stroke risk detected, whereas the value '1' indicates a possible risk of stroke. This dataset is highly imbalanced as the possibility of '0' in the output column ('stroke') outweighs that of '1' in the same column. Only 249 rows have the value '1' whereas 4861 rows with the value '0' in the stroke column. For better accuracy, data pre-processing is performed to balance the data. The dataset discussed above is summarized in Table 1.

TABLE I. Stroke Dataset

Attribute Name	**Type *(Values)***	**Description**
1. id	Integer	A unique integer value for patients
2. gender	String literal *(Male, Female, Other)*	Tells the gender of the patient
3. age	Integer	Age of the Patient
4. hypertension	Integer *(1, 0)*	Tells whether the patient has hypertension or not
5. heart_disease	Integer *(1, 0)*	Tells whether the patient has heart disease or not
6. ever_married	String literal *(Yes, No)*	It tells whether the patient is married or not
7. work_type	String literal *(children, Govt_job, Never_worked, Private, Self-employed)*	It gives different categories for work
8. Residence_type	String literal *(Urban, Rural)*	The patient's residence type is stored
9. avg_glucose_level	Floating point number	Gives the value of average glucose level in blood
10. bmi	Floating point number	Gives the value of the patient's Body Mass Index
11. smoking_status	String literal *(formerly smoked, never smoked, smokes, unknown)*	It gives the smoking status of the patient
12. stroke	Integer *(1, 0)*	Output column that gives the stroke status

B. Data Preprocessing

Data Preprocessing is required before model building to remove the unwanted noise and outliers from the dataset, resulting in a deviation from proper training. Anything that interrupts the model from performing with less efficiency is taken care of in this stage. After collecting the appropriate dataset, the next step lies in cleaning the data and making sure that it is ready for model building. The dataset taken has 12 attributes, as mentioned in Table I. Firstly, the column 'id' is dropped because its existence does not make much difference in model building. Then the dataset is checked for null values and filled if any found. In this case, the column 'bmi' has null values filled with the mean of the column data. After removing the null values from the dataset, the next task is Label Encoding.

C. Label Encoding

Label encoding encodes the string literals in the dataset into integer values for the machine to understand them. As the machine is usually trained in numbers, the strings have to be converted into integers. There are five columns in the collected dataset that have strings as their data type. On performing label encoding, all the strings get encoded, and the entire dataset becomes a combination of numerals.

D. Handling Imbalanced Data

The dataset chosen for the task of stroke prediction is highly imbalanced. The entire dataset has 5110 rows, of which 249 rows are suggesting the occurrence of a stroke and 4861 rows having the possibility of no stroke. The graphical representation of the imbalance is in Fig. 2. Training a machine-level model with such data might give accuracy, but other accuracy metrics like precision and recall are shallow. If such imbalanced data is not handled, the results are not accurate, and the prediction is inefficient. Therefore, to get an efficient model, this imbalanced data is to be first handled. For this purpose, the method of undersampling is used. Undersampling [13] balances the data wherein the majority class is undersampled to match the minority class. In this case, the class with a value as '0' is undersampled for the class with the value' 1'. So after undersampling the resulting dataset will have 249 rows with value ‘0’ and 249 rows with value ‘1’. The graphical representation of the output column in the resulting dataset is as shown in Fig. 3.

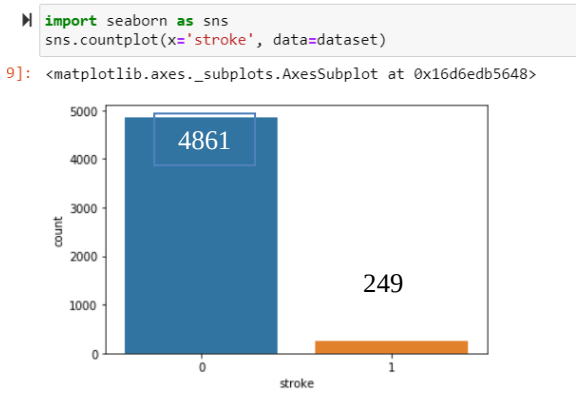

Fig. 2. Before Undersampling.

```
import seaborn as sns
sns.countplot(x='stroke', data=under_sample)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x16d6ee28508>
```

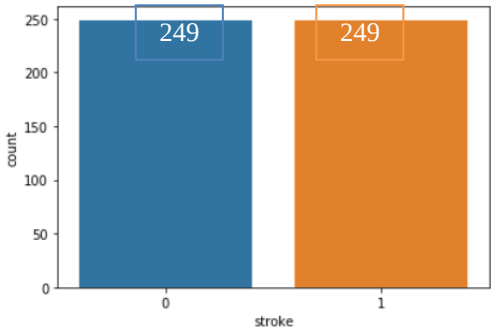

Fig. 3. After Undersampling.

V. Model Building

A. Splitting the Data

After completing data preprocessing and handling the imbalanced dataset, the next step is building the model. The undersampled data is split into training and testing data for better accuracy and efficiency for this task keeping the ratio as 80% training data and 20% testing data. After splitting, various classification algorithms are used to train the model. The classification algorithms used for this purpose are Logistic Regression, Decision Tree Classification algorithm, Random Forest Classification, K-Nearest Neighbors Classification, Support Vector Machine and Naïve Bayes Classification.

B. Classification Algorithms

1) Logistic regression: Logistic Regression [14] is a supervised learning algorithm used for predicting the probability of the output variable. This algorithm is the best fit when the output variable has binary values (0 or 1). As the output attribute in the dataset has only two possible values, Logistic Regression is opted. After performing this algorithm on the dataset, the accuracy obtained is 78%. Efficiency of this algorithm can also be found by using various other accuracy metrics like precision score and recall score. Those two scores obtained in this case are equal, having a value of 77.6%. The F1 Score obtained with this algorithm is 77.6%. The Receiver operating characteristic (ROC) curve for Logistic Regression is 78% as shown in Fig. 4.

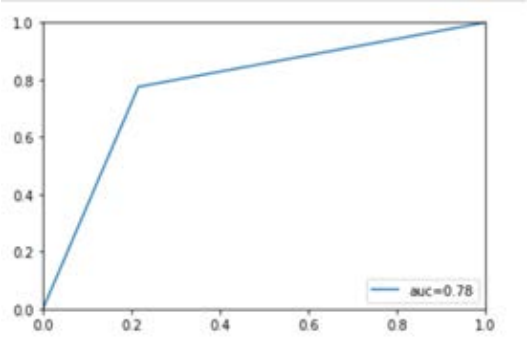

Fig. 4. ROC Curve for Logistic Regression.

2) Decision tree classification: Decision Tree classification [15] is to solve both Regression and classification problems. This algorithm is also a supervised learning method wherein the input variables already have their corresponding output variable. It is a tree-like structure. In this algorithm, the data continuously splits according to a particular parameter. A decision tree has two parts: Decision Node and Leaf node. The data splits at the former node, and the latter is the node that gives the outcome. In this case of stroke prediction, the decision tree classification algorithm obtained an accuracy of 66%, which is less than that obtained using logistic Regression. Similar to logistic Regression, the precision and recall scores are the same and equal to 77.6%. The F1 Score obtained with this algorithm is 77.6%. The Receiver operating characteristic (ROC) curve for Decision Tree Classification is 66% as shown in Fig. 5.

3) Random forest classification: The following classification algorithm chosen is Random Forest Classification [16]. Random Forests are composed of multiple independent decision trees trained independently on a random subset of data. These trees are generated at the time of training, and the outputs are obtained from each decision tree. For the final prediction from this algorithm, a method called "voting" takes place. This method means that each decision tree votes for an output class (in this case, the two classes are: 'stroke' and 'no stroke'). The random forest chooses the class with the maximum number of votes as the final prediction. The accuracy obtained by training the model using this particular algorithm is 73%. The precision and recall scores are 72% and 73.5%, respectively. The F1 Score obtained with this algorithm is 72.7%. The Receiver operating characteristic (ROC) curve for Random Forest Classification is 73% as shown in Fig. 6.

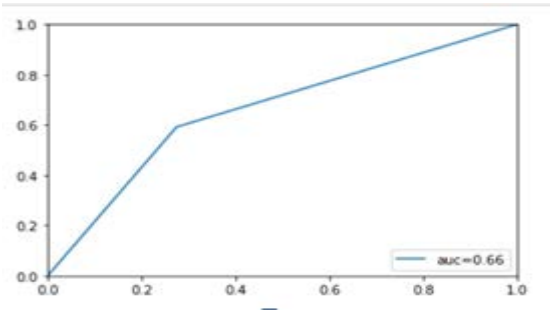

Fig. 5. ROC Curve for Decision Tree Classification.

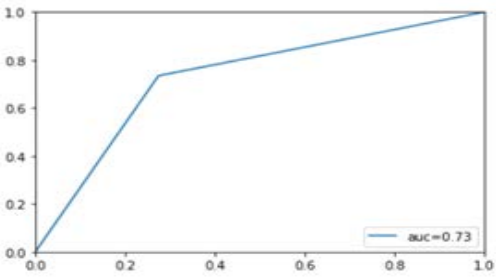

Fig. 6. ROC Curve for Random Forest Classification.

4) K-nearest neighbors classification: Another algorithm used for classification is K-Nearest Neighbors (KNN). It is also a supervised learning technique. KNN [17] is a lazy algorithm that would not train immediately on giving the dataset. Instead, it stores the dataset, and at the time of classification, it acts on the dataset. The working principle of KNN is to find similarities between the new case (or data) and available data and then map the new case into the category that is most similar to the available categories. The accuracy obtained is 80%. Precision and recall scores are 77.4% and 83.7%, respectively. The F1 Score obtained with this algorithm is 80.4%. The Receiver operating characteristic (ROC) curve for KNN is 80% as shown in Fig. 7.

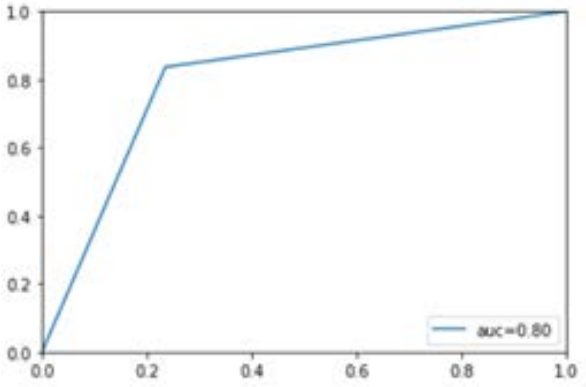

Fig. 7. ROC Curve for KNN.

5) Support vector machine: It is a supervised learning technique that can be associated with learning algorithms to analyze the data for both classification and regression. Support Vector Machine (SVM) [18] scales relatively well to high dimensional data. For this particular dataset, the algorithm obtained an accuracy of 80%, with precision and recall score being 78.6% and 83.8%. The F1 score is 81.1% for this algorithm. The Receiver operating characteristic curve (ROC) for Support Vector Classification is 80% as shown in Fig. 8.

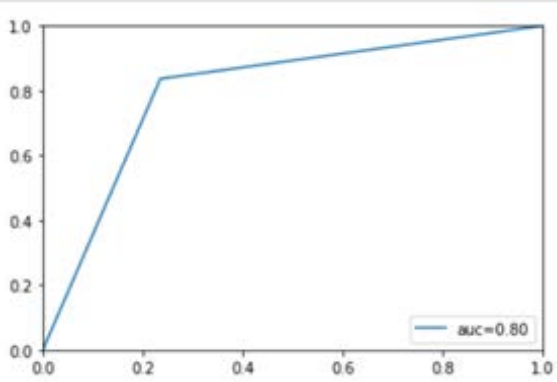

Fig. 8. ROC Curve for Support Vector Machine.

6) Naïve bayes classification: It is also a supervised learning technique. A Naïve Bayes classifier [19] assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. It is based on Bayes Theorem. This algorithm follows the principle that 'every feature or attribute classified is independent of one another.' With this algorithm, the accuracy obtained was 82%, with the precision score being 79.2% and recall score being 85.7%. The F1 Score obtained with this algorithm is 82.3%. The Receiver operating characteristic (ROC) curve for Naïve Bayes Classification is 82% as shown in Fig. 9.

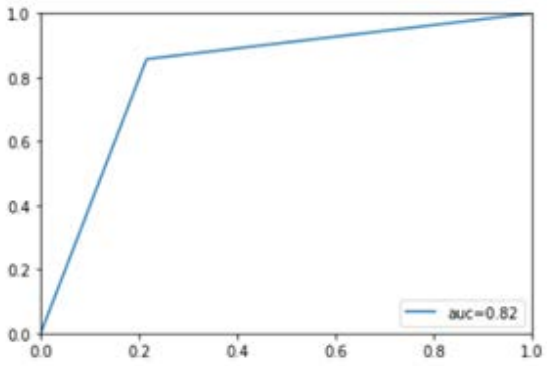

Fig. 9. ROC Curve for Naïve Bayes Classification.

After model building, it can be concluded that Naïve Bayes has performed better when compared to other algorithms. So, the model trained using Naïve Bayes classification is dumped using pickle. The next task is to develop a web application and a flask application to enter the input parameters.

The web page is built using simple HTML code. This application has an input form that will take the entered input values from the user to predict the occurrence of stroke. When the user clicks on the 'Check Here' button, the entered parameters are given to the flask application. A part of the HTML page is shown in Fig. 10.

The flask application which is basically a python code is the bridge between the web page and the trained machine learning model as shown in Fig. 11. The input values are sent to the flask application which in sends the values to the model for prediction.

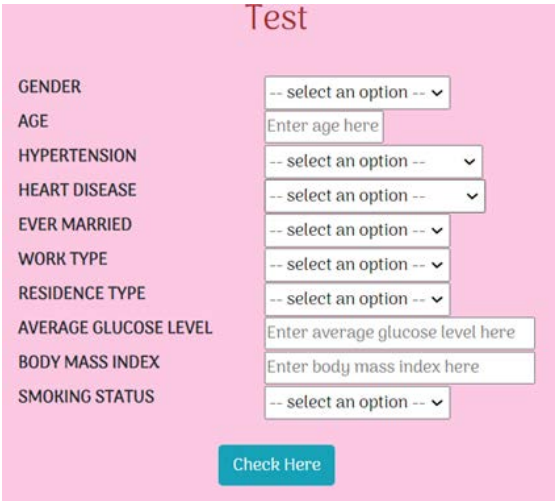

Fig. 10. Input Form in HTML Code.

```
from flask import Flask, request, render_template
import numpy as np
import pickle

model=pickle.load(open('stroke_model.pkl','rb'))

app = Flask(__name__)
@app.route('/')
def home():
    return render_template("home.html")
```

Fig. 11. Flask Application Linking the Model and Web Page.

Once the machine learning model gets the input parameters, it predicts the output. The obtained result then is reflected back on the web page through the flask application for the user to see the prediction.

VI. Conclusion

Stroke is a critical medical condition that should be treated before it worsens. Building a machine learning model can help in the early prediction of stroke and reduce the severe impact of the future. This paper shows the performance of various machine learning algorithms in successfully predicting stroke based on multiple physiological attributes. Out of all the algorithms chosen, Naïve Bayes Classification performs best with an accuracy of 82%. The comparison of accuracies obtained from various algorithms is as shown in Fig. 12. Among all the precision, recall and F1 scores obtained, Naïve Bayes has performed better. The comparison of Precision score, recall score and F1 score is as shown in Fig. 13, Fig. 14 and Fig. 15.

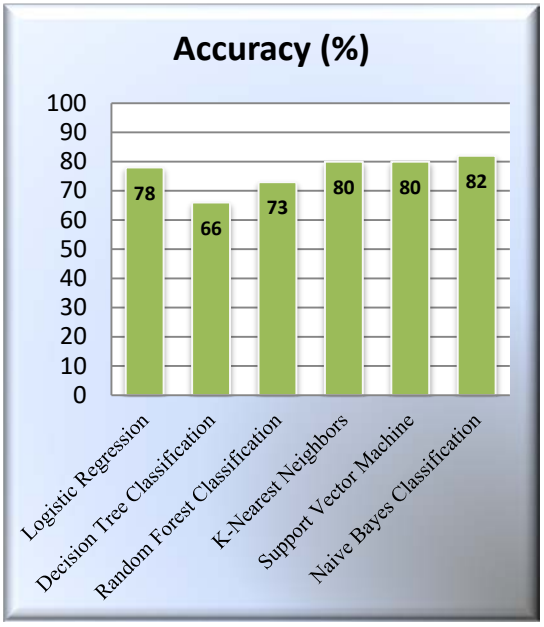

Fig. 12. Comparing the Accuracies of ML Algorithms.

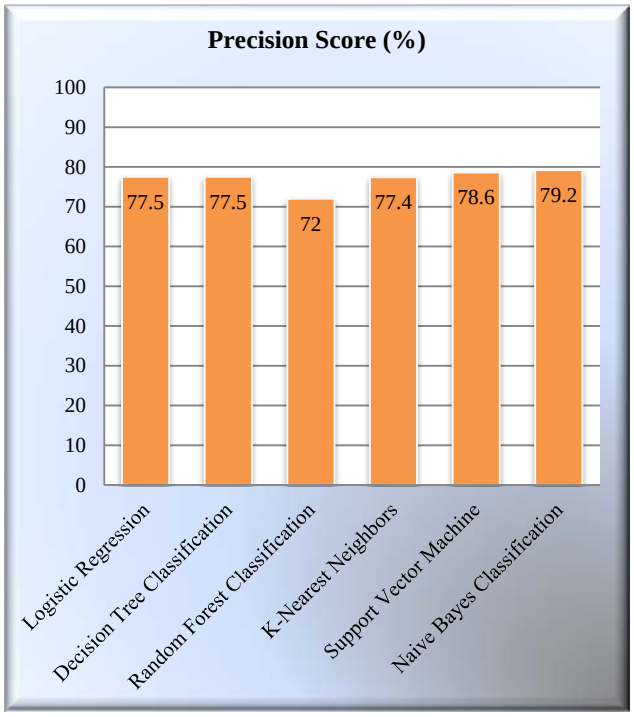

Fig. 13. Comparing the Precision Scores of ML Algorithms.

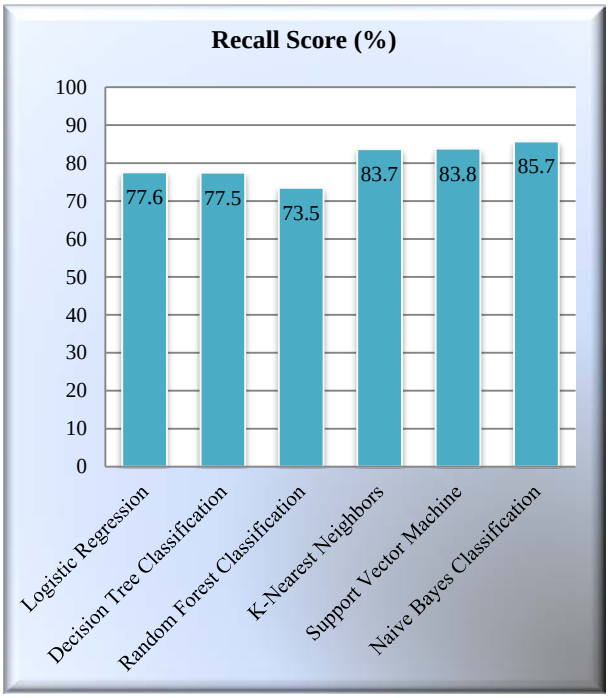

Fig. 14. Comparing the Recall Scores of ML Algorithms.

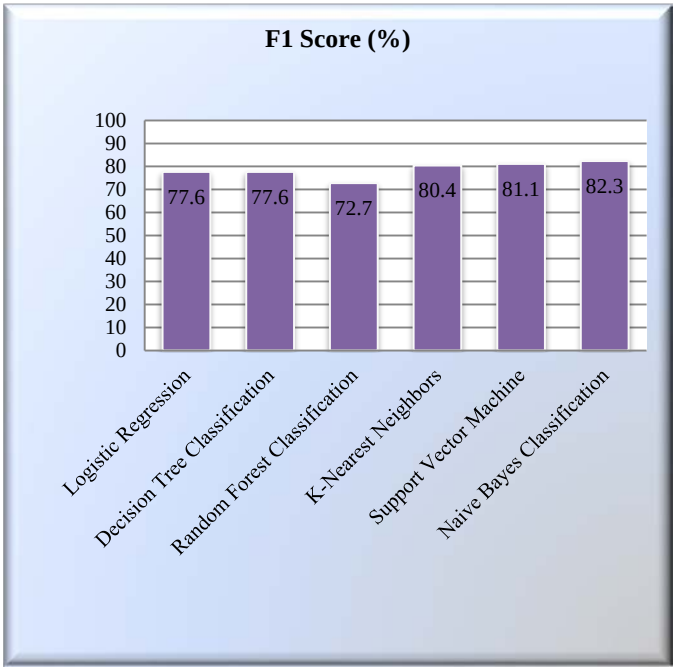

Fig. 15. Comparing the F1 Scores of ML Algorithms.

This paper suggests the implementation of various Machine learning algorithms on the dataset taken. This project can be further extended by training the model using Neural Networks. The comparison of the performance can be done by taking more accuracy metrics into consideration. This work is limited to textual data, which might not always be accurate for stroke prediction. Collecting a dataset consisting of images such as Brain CT scans to predict the possibility of stroke would be more efficient in the future.

REFERENCES

[1] Concept of Stroke by Healthline.

[2] Statistics of Stroke by Centers for Disease Control and Prevention..

[3] Dataset named 'Stroke Prediction Dataset' from Kaggle: https://www.kaggle.com/fedesoriano/stroke-prediction-dataset.

[4] Singh, M.S., Choudhary, P., Thongam, K.: A comparative analysis for various stroke prediction techniques. In: Springer, Singapore (2020).

[5] Pradeepa, S., Manjula, K. R., Vimal, S., Khan, M. S., Chilamkurti, N., & Luhach, A. K.: DRFS: Detecting Risk Factor of Stroke Disease from Social Media Using Machine Learning Techniques. In Springer (2020).

[6] Vamsi Bandi, Debnath Bhattacharyya, Divya Midhunchakkravarthy: Prediction of Brain Stroke Severity Using Machine Learning. In: International Information and Engineering Technology Association (2020).

[7] Nwosu, C.S., Dev, S., Bhardwaj, P., Veeravalli, B., John, D.: Predicting stroke from electronic health records. In: 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society IEEE (2019).

[8] Fahd Saleh Alotaibi: Implementation of Machine Learning Model to Predict Heart Failure Disease. In: International Journal of Advanced Computer Science and Applications (IJACSA) (2019).

[9] Ohoud Almadani, Riyad Alshammari: Prediction of Stroke using Data Mining Classification Techniques. In: International Journal of Advanced Computer Science and Applications (IJACSA) (2018).

[10] Kansadub, T., Thammaboosadee, S., Kiattisin, S., Jalayondeja, C.: Stroke risk prediction model based on demographic data. In: 8th Biomedical Engineering International Conference (BMEiCON) IEEE (2015).

[11] Aditya Khosla, Yu Cao, Cliff Chiung-Yu Lin, Hsu-Kuang Chiu, Junling Hu, Honglak Lee:An Integrated Machine Learning Approach to Stroke Prediction. In: Proceedings of the 16th ACM SIGKDD international conference on Knowledge discovery and data mining (2010).

[12] Shanthi, D., Sahoo, G., Saravanan, N.: Designing an artificial neural network model for the prediction of thrombo-embolic stroke. Int. J. Biometric Bioinform. (IJBB) (2009).

[13] 7 Techniques to Handle Imbalanced Data – Kdnuggets.

[14] Documentation for Logistic Regression from Scikit-learn.org.

[15] Documentation for Decision Tree Classification from Scikit-learn.org.

[16] Documentation for Random Forest Classification from Scikit-learn.org.

[17] Documentation for K-Nearest Neighbor from Scikit-learn.org.

[18] Documentation for Support Vector Machine from Scikit-learn.org.

[19] Documentation for Naïve Bayes Classification Algorithm from Scikit-learn.org.