Stroke Prediction Using Machine Learning Classification Methods

Hamza Al-Zubaidi
Computer Engineering Department
Princess Sumaya University for Technology
Amman, Jordan
ham20180659@std.psut.edu.jo

Mohammed Dweik
Computer Engineering Department
Princess Sumaya University for Technology
Amman, Jordan
moh20190113@std.psut.edu.jo

Amjed Al-Mousa
Computer Engineering Department
Princess Sumaya University for Technology
Amman, Jordan
a.almousa@psut.edu.jo

Abstract— **Based on machine learning, this paper aims to build a supervised model that can predict the presence of a stroke in the near future based on certain factors using different machine learning classification methods. The predictions resulting from this model can save many lives or give people hints on how they can protect themselves from the risk. The models obtained from this research are just a tool that doctors can use; thus, it does not take the role of doctors. The model was trained on a dataset that contains the factors or features that affect stroke disease. The correlation values were calculated to know how much a particular feature affects the target feature (having a stroke) or if other features are affected by it. After all, the model was tested on a set of samples to measure the accuracy of the trained model. Finally, multiple models were produced using different algorithms (classifiers), but the model that produced the best accuracy, precision, recall, and F1-score of 94%-95% is based on the Random Forest classifier.**

***Keywords*—Machine learning, Stroke, Ensemble methods, Unbalanced dataset, Classification.**

I. INTRODUCTION

According to [1], Stroke is a disease that affects the arteries leading to and within the brain; it occurs when a blood vessel that carries oxygen and nutrients to the brain is either blocked by a clot or bursts (or ruptures); thus, part of the brain will not be able to get the blood and oxygen it needs which leads to damage the brain cells and cause a stroke. Stroke has many types, which can cause death or at least affect the body in a certain way. For example, if it occurs toward the back of the brain, that will most likely lead to some disabilities in vision.

Based on [1] and [2], Stroke is the 2nd leading cause of death globally; responsible for approximately 11% of total deaths, and it is the No. 5 cause of death and a leading cause of disability in the United States. The good news is that 80% of strokes are preventable, leading to one of this research's primary goals: protection from Stroke.

The model focuses on predicting if the person can have a stroke shortly or detecting if the patient has a stroke or not based on multiple factors (aka features). The dataset used for this research was fetched from [3], and the Spanish Data Scientist Fedesoriano built it. The original dataset contains 12 features. Each one of these features has a relation with the desired output (aka label) which is having a stroke or not, and contributes to producing the final result in one way or another. Another important note about this dataset is that it has 5110 samples, where 95% of these samples have considered not having a stroke, and just 5% have considered having a stroke. These percentages indicate that this dataset is unbalanced, and this problem will be considered in this research.

Several machine learning classification methods will be used to get high accuracy, precision, and recall, starting with the Random Forest (RF) classifier and passing through Decision Tree (DT), Logistic Regression (LR), Support Vector Machine (SVM), and ending with ensemble methods as hard and soft voting classifiers.

II. RELATED WORK

Machine Learning has been used successfully in predicting several diseases, like Diabetes [4] and heart disease [5]. In addition, several attempts in the machine learning field to build a stroke predictor module using different classification methods. The research presented in [6] used the same dataset with the chosen features used for this paper. This research used the Undersampling technique to handle the unbalancing issue and six different algorithms (classifiers): Naïve Bayes, SVM, K-Nearest Neighbors, RF, DT, and LR, to obtain convenient accuracy, precision, recall, and F1-score. The Naïve Bayes classifier has achieved the best results with accuracy, precision, recall, and F1-score of 82%, 79.2%, 85.7%, and 82.3%, respectively. The Naïve Bayes classifier achieved the best results, so it is convenient to have the highest percentage of area under the ROC curve, which equals 82%.

The second research presented in [7] used the same dataset with the same chosen features, which will be used for this paper, and it used a SMOTE technique to handle the unbalancing issue. Note that the "unknown" class in the "smoking status" feature was considered a non-null class. It also used three classifiers to obtain high results: RF, K-Nearest Neighbors, and Logistic Regression. Accurate results were achieved using the random forest classifier (RFC) with accuracy, Precision, Recall, and F1-score equal to 96%.

The third research presented in [8] used the same dataset but with some differences in chosen features. Nine features were chosen out of twelve to obtain the final results. The discarded features were the "id", "BMI", and "ever married" features.

2022 International Arab Conference on Information Technology (ACIT) | 979-8-3503-2024-4/22/$31.00 ©2022 IEEE | DOI: 10.1109/ACIT57182.2022.10022050

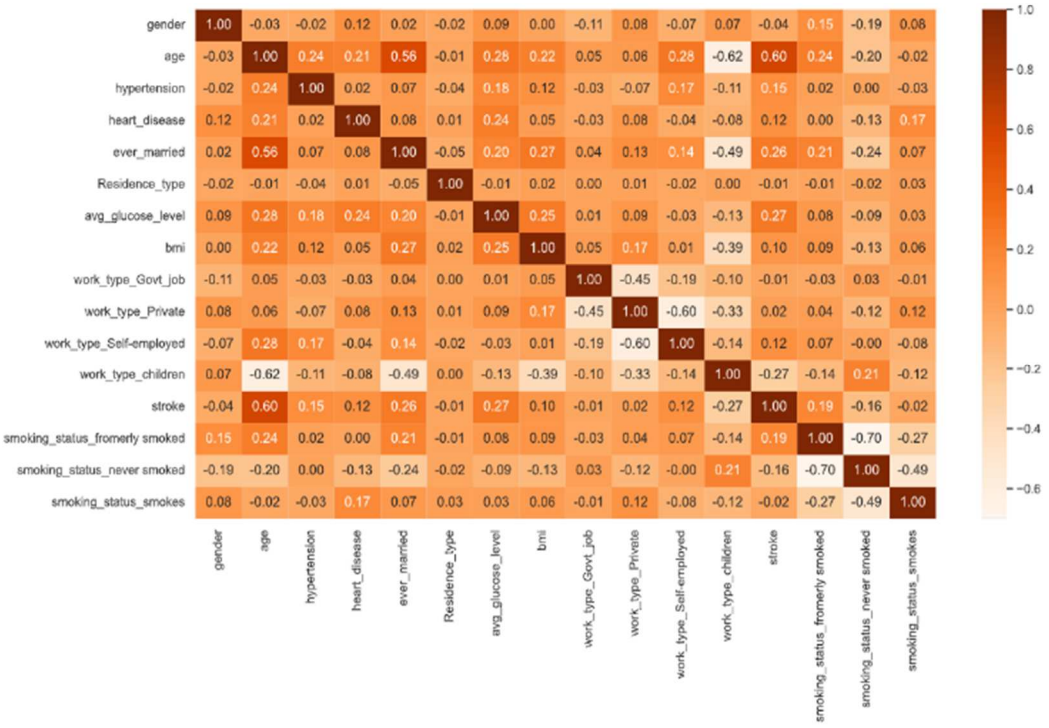

Figure 1: Correlation matrix between features

It also used the SMOTE technique to handle the unbalancing issue and used a set of eight different classifiers. Some of them are interesting to be used, such as the neural network that got an accuracy of 82.4% and the XGBoost classifier that got an accuracy of 91.5%. However, it is still not the best since the RFC got an accuracy of 92.3%.

The fourth research presented in [9] used a different dataset with 4,799 samples and ten features (different features compared with the dataset's features of this paper). The dataset used is more balanced compared with the dataset of this research. It used nine different classifiers: DT, Gaussian Naïve Bayes, LR, Linear SVM, Poly SVM, RGB SVM, RF, AdaBoost, and AdaBoost with SGD. Some of these algorithms produced surprisingly good results, such as the AdaBoost classifier, which got an accuracy, sensitivity, and specificity of 86.87%, 95.78%, and 72.47%, respectively, but still the best results came from the RFC, which produced an accuracy, sensitivity, and specificity of 94.23%, 92.16%, and 95.07% respectively.

III. Experimental Setup

The primary purpose of this research is to build a supervised model which can predict the presence of Stroke shortly based on a dataset that contains real cases. Most of these cases will be used to train the model, and some will be test cases to measure the quality of the model to generalize it to any other cases and consider it a reliable tool.

A. Dataset Attribute Information

The stroke prediction dataset fetched from [3] has 5110 samples and twelve features, and one of these features, "id" will be discarded since it does not have any logical meaning. Table 1 shows the features of this dataset after processing (which will be discussed in the data processing section). Note that the number of features increased to 16 due to encoding two categorical features into multiple binary features. The first one is the "work-type" feature that was encoded to four binary features that are "Work-Govt-Job" which represents a government job, "Work-Private", and "Work-Children" which represents housewife women (having children). The second one is the "smoking-status" feature. It was encoded to three binary features that are "Smoking-Smokes", "Smoking-Formerly", and "Smoking-Never".

Table 1: Stroke Prediction Dataset Attributes Information

Attribute	Type	Information
Gender	Discrete	Male = 1, Female = 0
Age	Continuous	Scaled from 0-1
Hypertension	Discrete	High = 1, Low = 0
Heart Disease	Discrete	Yes = 1, No = 0
Ever Married	Discrete	Yes = 1, No = 0
Residence Type	Discrete	Urban = 1, Rural = 0
Avg Glucose Level	Continuous	Scaled from 0-1
BMI	Continuous	Scaled from 0-1
Work-Govt-Job	Discrete	Yes = 1, No = 0
Work-Private	Discrete	Yes = 1, No = 0
Work-Self-Employed	Discrete	Yes = 1, No = 0
Work-Children	Discrete	Yes = 1, No = 0
Smoking-Smokes	Discrete	Yes = 1, No = 0
Smoking-Formerly	Discrete	Yes = 1, No = 0
Smoking-Never	Discrete	Yes = 1, No = 0
Stroke	Discrete	Yes = 1, No = 0

Each feature has a relation with the desired output called a correlation. Correlation is a metric that explains how one or more features are related to each other, so each peer of features can have a proportional or inversely proportional relation. Figure 1 indicates the correlation between each peer of features, and table 2 shows the correlation with the outcome, stroke diagnosis.

Table 2: Correlation values with the target.

Attribute	Correlation Value
Age	0.598189
Avg Glucose Level	0.265909
Ever Married	0.262617
Smoking-Formerly	0.190377
Hypertension	0.146461
Heart Disease	0.124375
Work_Self_Employed	0.123229
BMI	0.095948
Work-Private	0.023700
Work_Gov_Job	-0.013140
Residence_type	-0.014053
Smoking-Smokes	-0.018508
Gender	-0.040173
Smoking-Never	-0.158343
Work-Children	-0.270147

A. Dataset Preprocessing

The stroke prediction dataset has many issues that must be accessed to achieve the best possible results; otherwise, these issues may negatively affect the accuracy of results. The below points discuss several issues discovered in this dataset and how it was accessed.

I. Deleting meaningless features: "ID" feature has no logical meaning; thus, it was deleted.

II. Handling numerical null values: BMI feature contains 201 null values out of 5110. To replace these null values with convenient values, the mean value of the "BMI" column was calculated, and null values were replaced.

III. Handling categorical data (features): Five features out of eleven are not numerical. Most machine learning algorithms cannot handle categorical data; thus, converting categorical data (features) to numerical data is a significant step in data processing. "Ever married", "gender", and "Residence type" features have just two classes, and these classes can be directly switched by one and zero since this encoding (label encoding) can still represent a logical meaning for these features. "Work Type" and "smoking status" have more than two classes, each with an independent meaning; thus, their classes were converted to binary features using the One Hot Encoding method.

IV. Deleting unnecessary features: The "Work-Type-Never-Worked" feature is one of the encoded classes of the "Work Type" feature. It has just 22 corresponding samples out of 5110; thus, it is considered an unnecessary feature and was dropped with its corresponding rows. The number of remaining samples in the dataset is 5088 samples.

V. Scaling features: Scaling "age", "avg-glucose", and "BMI" features using a min-max scalar since there are many algorithms that depend on the distance between points (samples), such as the SVM algorithm that will be used later to classify the stroke diagnosis, the accuracy of this algorithm is negatively affected by far distances between values; thus, scaling these features will enhance the accuracy out of this algorithm. The min-max scalar formula is given by equation 1 [10].

$$\text{Scaled } x = (x - \text{Min})/(\text{Max} - \text{Min}) \quad (1)$$

where "x" is the sample's feature value. This formula leads to scale values between zero and one.

VI. Handling unbalanced data (label): "Stroke" feature (label) is considered the desired output from this project, the data of this label is unbalanced, which means there are majority and minority classes, and the difference between them is too big, to be specific, the majority class which is represented by zero has 4838 corresponding samples out of 5088 samples while the minority class which one represents has just 249 corresponding samples out of 5088 samples, this unbalancing will lead to a bad accuracy of predictions; thus, this problem must be addressed.

According to [11], unbalanced data can be addressed using multiple techniques, such as the Undersampling technique, which is done by deleting a certain number of samples from the majority class; thus, the data will then be balanced, but in most cases, this approach may lead to a loss of information especially in this dataset since a considerable number of data must be deleted to reach balancing. Another approach that can be used is the Random over-sampling technique by duplicating random samples from the minority class [11]. This approach may lead to overfitting the training data since 40% or 45% of the training data are just duplicated from other samples [12].

Based on [13], SMOTE (Synthetic Minority Over-sampling Technique) is the technique that is used in this research to address this problem in this dataset. This technique works based on the K-nearest neighbor algorithm, so it takes the samples from the minority class and then calculates the distances between them, then identifies new points (samples) which will be located at the line segments between the original points; thus, the new samples are not duplicates, but they are near from minority class's samples. Figure 2 shows the difference between the majority and minority classes before using SMOTE technique, and Figure 4 shows this difference after using it. Figures 3 and 5 illustrate that difference by showing the distribution of the dataset's samples based on "age" and "avg glucose level" features

before and after using SMOTE technique. The SMOTE technique was done using all 16 features, and the main purpose of Figures 3 and 5 is to illustrate the idea behind the result.

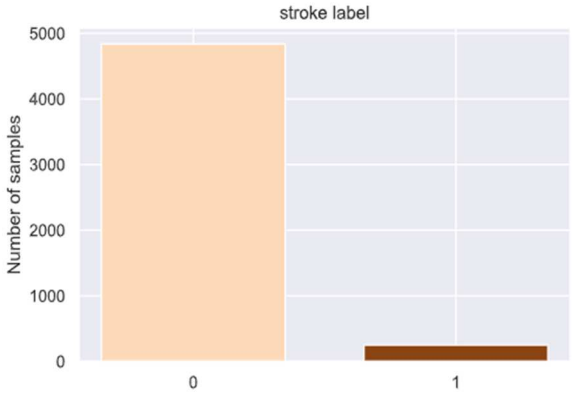

Figure 2: Majority and minority classes in stroke label.

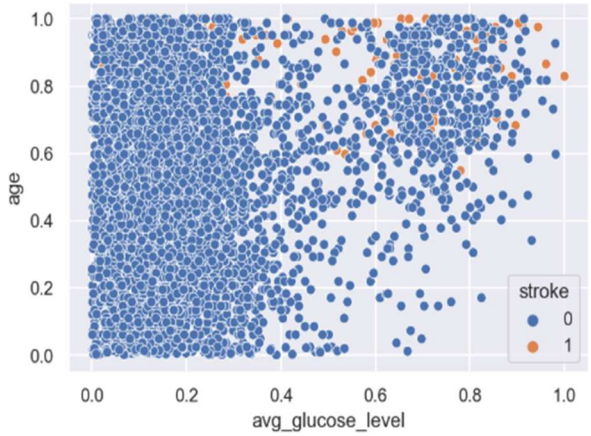

Figure 3: Distribution of data based on "age" and "avg glucose level" features

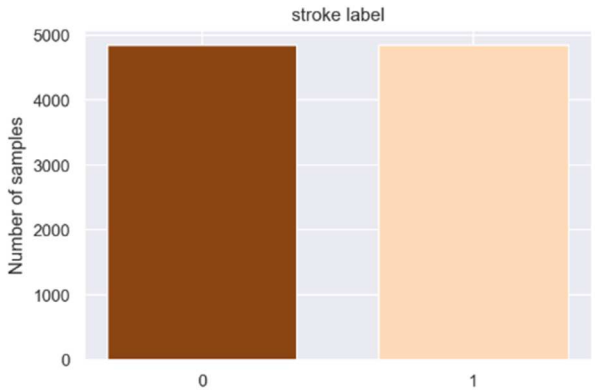

Figure 4: Stroke label's classes after balancing the data.

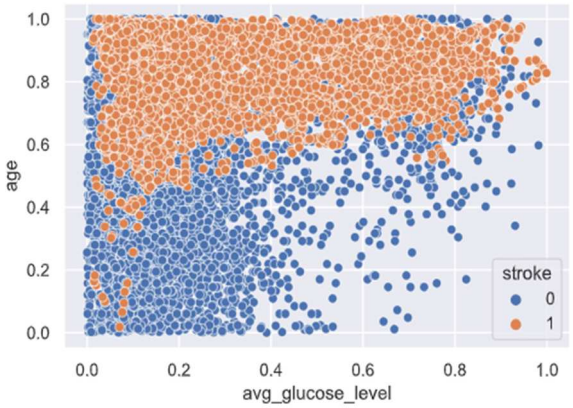

Figure 5: Data distribution based on "age" and "avg glucose level" features after balancing the data.

VII. Handling categorical null values: "Smoking_status" feature has a class of "Unknown" with 2491 corresponding samples. This class is not logical since the primary purpose of this research is to predict the presence of Stroke in the near future; thus, to fix this problem efficiently, a RFC will be built to predict or switch the "unknown" class of the 2491 samples with one of the other three classes (smokes, never smoked, formerly smoked).

The first step is encoding the "smoking_status" feature using the label encoding method. The second step is to convert the sample's values in class "0", which represents the "Unknown" class, to NULL values. The third step is to build the model using the RFC, where the samples with non-null values represent the training set, and the samples with null values represent the test set.

The cross-validation method was used to produce a test label since there is no test label. The cross-validation score approach was used to calculate the accuracy based on five tests (CV=5). Each test has a different validation set (test label). This approach calculates the average accuracy based on each test's resulting accuracy. The average accuracy produced is 70% which is not too high, but note that at this point, the target is just to minimize the error; thus, 70% accuracy is still much better than filling the null values with the mode or deleting their corresponding rows. The fourth step is decoding the "smoking_status" feature to its original categorical classes to encode them to multiple binary features using One-Hot Encoding.

After processing the data and addressing problems, it is time to overview the data distribution (histogram) based on each feature, as shown in Figure 6. These histograms can make detecting any outliers much more straightforward, which may negatively affect the accuracy of predictions.

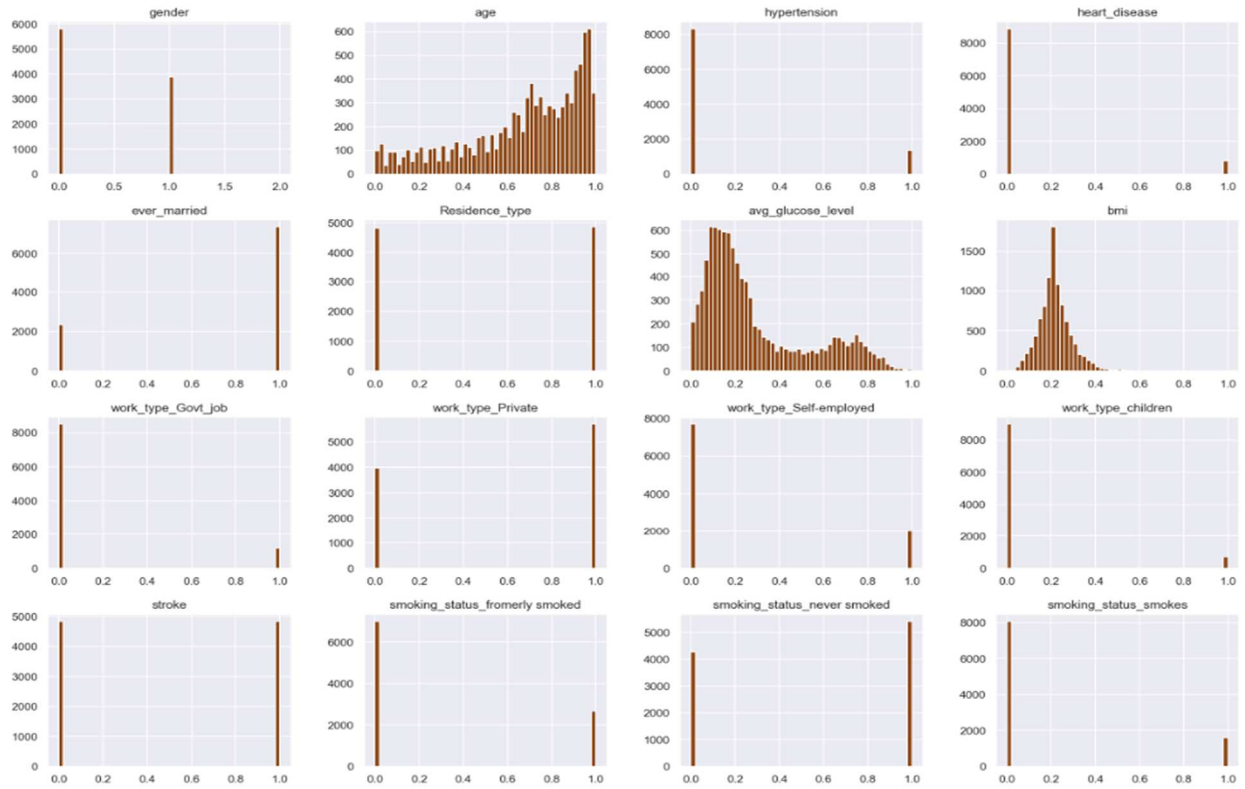

Figure 6: Histograms of all features.

IV. Machine Learning Algorithm

To build models, the data was divided into 80% as a training set and 20% as a test set in a stratified fashion to ensure the even distribution of data. It is now the time to build models using different machine learning algorithms with different parameters to achieve the best results. Six models were built using five different classifiers that are:

A. Decision Tree Classifier (DTC)

The first model was built using DTC. This classifier works by building a tree starting with the root node (feature) and then splitting it into sub-nodes based on certain conditions. Each node represents a specific feature, then the process of splitting will continue until reaching leaf nodes, representing the possible predictions out of its corresponding subtree based on previous satisfied conditions. The bottleneck of this classifier is how to define which attributes must be used in each step and which one must be chosen as a tree's root. Cost functions are considered a technique that can be used to detect how much bad the algorithm is; thus, the attributes and thresholds which minimize the cost function of the DTC must be chosen. Several metrics, such as Gini and Entropy, contribute to calculating the cost function.

Gini impurity measures the frequency at which any dataset element will be mislabeled when randomly labeled. In other words, it represents how much the data is pure (refers to the same class) [14]. When Gini in a specific node equals zero, then all samples in this node are of one class, indicating that the current node is a leaf node. Equation (2) represents the Gini impurity.

$$Gini = 1 - \sum_j P_j^2 \quad (2)$$

Entropy is a measure of information that indicates the disorder of the features with the target. Entropy needs more computational power than Gini, but most of the time, it produces better results. Equation (3) represents the entropy metric [14].

$$Entropy = -\sum_j P_j . log_2 P_j \quad (3)$$

The target is to build modules that can achieve the best accuracies for this research. The computation power is not a big deal since the dataset is relatively small. The grid search technique was used to choose between Gini or Entropy, and the winner was Entropy.

B. Random Forest Classifier

The second model was built using RFC. This classifier is a scaled version of the DTC, in other words, an ensemble of DT. It works by creating a specific amount of DT using different combinations of thresholds and features, then classifying each instance using all trees, the class with the highest votes is considered the predicted class. Figure 7 illustrates this point. The RF can obtain more accurate results than DT in most cases. Also, the RFC can avoid overfitting since it uses multiple trees, but in terms of computational power, the DTC requires much less computational power than RF; thus, obtaining accurate results using DT can lead to dispensing RFC.

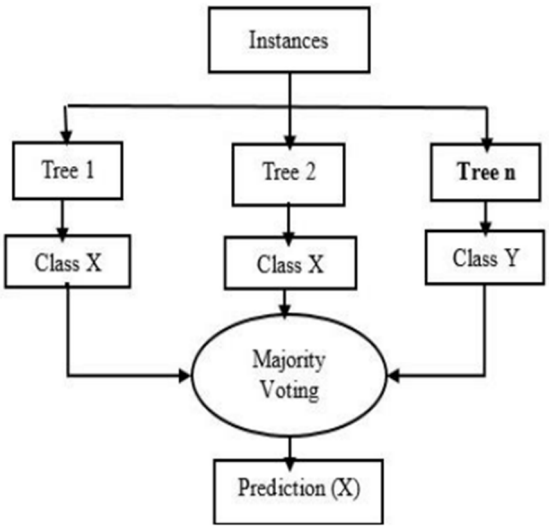

Figure 7: Random Forest Classifier [4]

C. Logistic Regression Classifier

The third model was built using Logistic Classifier. This classifier works by modeling the probability of a discrete (binary) outcome which can be one/zero, yes/no, or true/false. It can be considered a linear regression for classification problems [15]. It takes any range of values as input and then converts it to a probability between zero and one using the sigmoid function, which is represented by equation 4. After that, it classifies the instance based on the resulting probability. If it is greater or equals 0.5, the sample will be classified as class "1". Otherwise, it will be classified as class "0".

$$Sigmoid\ Function = \frac{1}{1+e^{x}} \quad (4)$$

D. Support Vector Machine

The fourth model was built using the SVM algorithm. SVM works by finding the suitable margin that separates different classes. In this case, it will separate into two different classes, which are having a stroke and does not have a stroke. To find the best margin, it uses the cross-validation method, so it tries many different solutions and then chooses the best solution to deal with outliers and misclassification points. The number of features represents the dimension which SVM will work with. Still, in some cases, it is hard to find a suitable margin in the current dimensions due to the data distribution, so SVM increases the number of dimensions to find an image of each point in the new dimension and then find the suitable margin.

SVM has a kernel to find a relation between points (samples) to transform the data to a higher dimension. There are multiple kernels, such as linear kernel (not useful on high-dimensional data) and Radial kernel (useful on high-dimensional data). The Grid search technique was used to choose the best "C" and gamma. "C" is the penalty parameter, representing misclassification or error term; if it is too high, it may lead to overfitting the training data. Gamma sets if the far points from the margin can affect the value of it or just the near points. When gamma is high, only near points are considered. And when it is low, even far away points are also considered. The best estimator out of grid search has the parameters C = 10000 and gamma = 1. Note that the "RBF" kernel was used since it can deal with high-dimensional data.

E. Voting Classifier

The fifth and sixth models were built using the voting classifier. This classifier is an ensemble method since it uses multiple classifiers to achieve the highest possible accuracies. Each chosen classifier predicts the target by giving a probability to each class for each instance. There are two main types of voting which are hard and soft voting. The hard voting classifier classifies the instance based on the number of times it is classified to a specific class using the contributed classifiers without caring about the confidence of classification. In other words, the differences in probabilities between classes are not considered. The classification in the soft voting classifier is based on the average probability; thus, each classifier's confidence or the differences between probabilities will take place in the final decision. This research used RF, DT, LR, and SVM classifiers to build a voting classifier.

V. Results

A set of metrics and the confusion matrix were used as quality metrics for the results. These metrics are accuracy score, precision, recall, and F1-score. The most critical metric is the recall of class "1" since it depends on the false negative cases. In other words, telling someone that he most likely won't have a stroke when he can have it is considered the most critical decision; thus, it is important to keep the recall of class "1" high.

The produced confusion matrices were normalized as percentages per row. The grid search technique was used to choose between "Gini" and "entropy" metrics for the DT model. The confusion matrix of this model is shown in Figure 8.

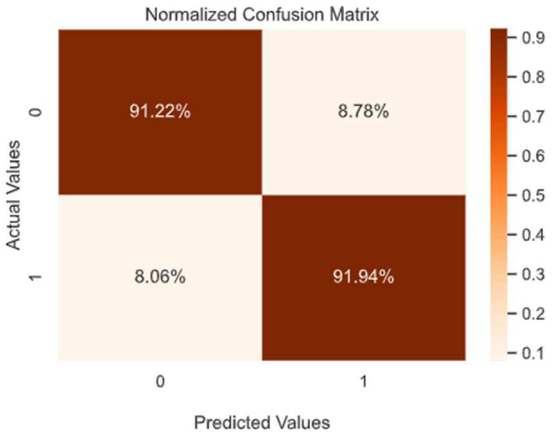

Figure 8: Confusion matrix of Decision Tree Classifier

The second model was built using the RFC and got the best results out of all the other four algorithms. The grid search technique was used for tuning the parameters of the algorithm. Figure 9 shows the confusion matrix of this model.

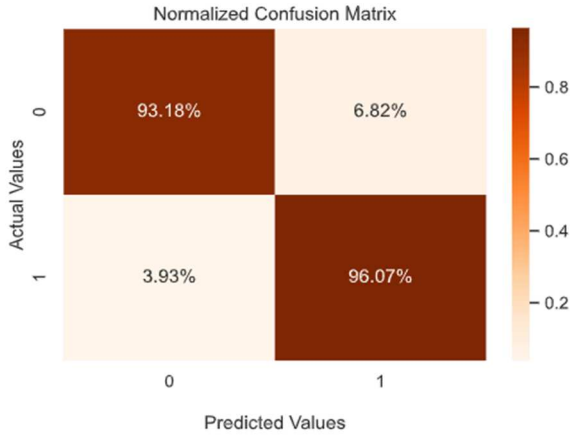

Figure 9: Confusion matrix of Random Forest Classifier

The third model was built using a LR classifier. The results of this model had the lowest quality. Grid search is not considered an efficient technique with this algorithm since it does not have critical parameters that can affect the results; thus, using this algorithm will waste computation power. It was tested on training data to ensure no overfitting in the model, and the results were almost the same. Figure 10 shows the confusion matrix of this model.

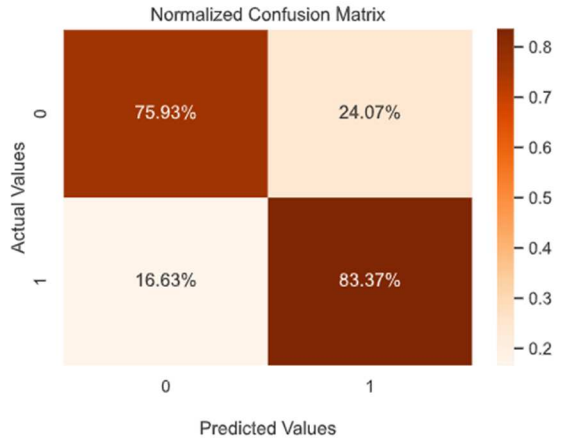

Figure 10: Confusion matrix of Logistic Regression Classifier

The fourth model was built using the SVM classifier. The grid search technique was used to obtain the best values of "C" and "gamma" parameters without overfitting the training data. The confusion matrix of this model is shown in Figure 11.

The fifth model was built using the Hard-Voting classifier. DT, RF, LR, and SVM are the participating algorithms in voting. There is no need for the grid search technique since each type (hard and soft) was implemented independently. Figure 12 shows the confusion matrix of this model.

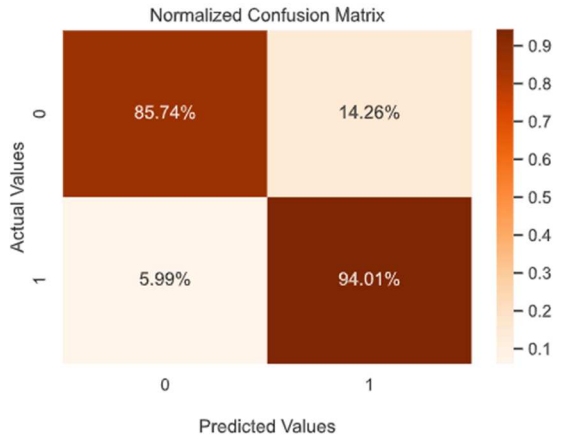

Figure 11: Confusion matrix of Support Vector Machine

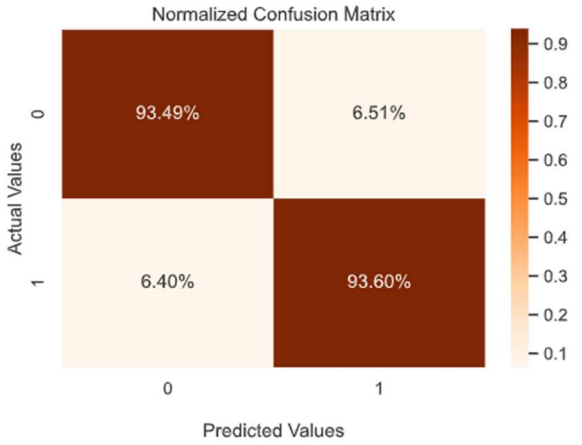

Figure 12: Confusion matrix of Hard Voting Classifier

The sixth model was built using the Soft Voting classifier. The confusion matrix of this model is shown in figure 13.

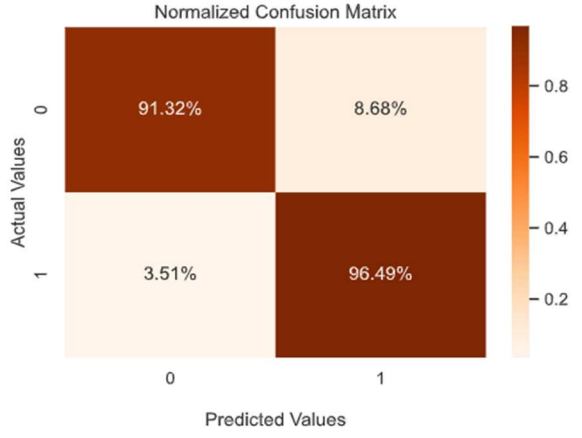

Figure 13: Confusion matrix of Soft Voting Classifier

Table 3 shows the quality measurement metrics of each class for each classifier (model). It involves the accuracy score, precision, recall, and F1 score.

Table 3: Quality measurement metrics of each model.

Classifier	Class	Accuracy Score	Precision	Recall	F1-score
Decision Tree	1	91.6%	91%	92%	92%
	0	91.6%	92%	91%	92%
Random Forest	1	94.6%	93%	96%	95%
	0	94.6%	96%	93%	95%
Logistic Regression	1	79.7%	78%	83%	80%
	0	79.7%	82%	76%	79%
Support Vector Machine	1	90%	87%	94%	90%
	0	90%	93%	86%	89%
Hard Voting	1	93.5%	93%	94%	94%
	0	93.5%	94%	93%	94%
Soft Voting	1	93.9%	92%	96%	94%
	0	93.9%	96%	91%	94%

Another way to compare different models is using the ROC curve representing the relation of true positive rate versus false positive rate. Figure 14 shows the ROC curve of all models along with their areas. Note that the optimal value for the area is 1.

Note that the ROC curve requires the percentages of predictions to produce a smooth curve. This illustrates why some models have a sharp edge because these models were built using algorithms that cannot efficiently deal with or do not deal with probabilities, such as hard voting.

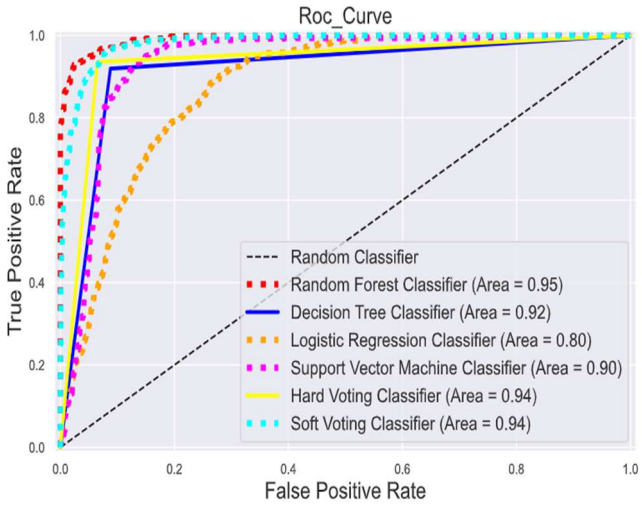

Figure 14: ROC curve for all models

VI. CONCLUSION

A stroke is a severe medical illness that causes a loss of brain functions. Machine learning models can predict Stroke in its early stages and reduce the risk of its consequences. This paper proposes different Machine Learning solutions to predict Stroke based on several features and successfully give a prediction with high accuracy. This paper tested different types of Machine learning algorithms on the selected dataset. The RF algorithm gave the highest accuracy of 94.7% of all tested algorithms. RF also achieved the highest precision, recall, and F1-Score.

The future step includes training the models using Neural Networks that can raise the accuracy and decrease the error by considering more accuracy metrics. Training the models on a dataset containing images of brain scans may be more efficient in the future.

REFERENCES

[1] "About stroke." www.stroke.org. [Online]. Available: https://www.stroke.org/en/about-stroke. [Accessed: 27-May-2022].

[2] "Stroke, Cerebrovascular accident," *World Health Organization*. [Online]. Available: http://www.emro.who.int/health-topics/stroke-cerebrovascular-accident/index.html. [Accessed: 27-May-2022].

[3] *Stroke Prediction Dataset,* fedesoriano, Data Scientist at Kaggle, 2021. [Online]. Available: https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset

[4] N. Abdulhadi and A. A. Al-Mousa, "Diabetes Detection Using Machine Learning Classification Methods," in 2021 International Conference on Information Technology (ICIT), Amman, 2021.

[5] R. Atallah and A. A. Al-Mousa, "Heart Disease Detection Using Machine Learning Majority Voting Ensemble Method," in 2nd International Conference on new Trends in Computing Sciences (ICTCS), Amman, 2019.

[6] Sailasya, G., & Kumari, G. L. A. "Analyzing the performance of stroke prediction using ML classification algorithms*." International Journal of Advanced Computer Science and Applications(IJACSA)*, Vol. 12, No.6, 2021, pp.539-545.

[7] Islam, M.M., Akter, S., Rokunojjaman, M., Rony, J.H., Amin, A. and Kar, S., 2021. "Stroke prediction analysis using machine learning classifiers and feature technique." *International Journal of Electronics and Communications Systems*, Vol. 1, No.2, 2021, pp.17-22.

[8] J.Tavares, "Stroke prediction through Data Science and Machine Learning Algorithms", 2021, DOI: 10.13140/RG.2.2.33027.43040.

[9] Bandi, V., Bhattacharyya, D., Midhunchakkravarthy, D. "Prediction of brain stroke severity using machine learning". *Revue d'Intelligence Artificielle*, Vol. 34, No.6, 2020, pp. 753-761, doi.org/10.18280/ria.340609

[10] S. Ozdemir and D. Susarla, "Feature engineering made easy," *O'Reilly Online Learning*. [Online]. Available: https://www.oreilly.com/library/view/feature-engineering-made/9781787287600/aa5580ee-6fb7-4ac2-a1fe-369d95b70168.xhtml. [Accessed: 27-May-2022].

[11] J. Brownlee, "Random oversampling and undersampling for imbalanced classification," *Machine Learning Mastery*, 04-Jan-2021. [Online]. Available: https://machinelearningmastery.com/random-oversampling-and-undersampling-for-imbalanced-classification/. [Accessed: 27-May-2022].

[12] D. Cummins and F. Dernoncourt, "Opinions about oversampling in general, and the smote algorithm in particular," *Cross Validated*, 01-May-1964. [Online]. Available: https://stats.stackexchange.com/questions/234016/opinions-about-oversampling-in-general-and-the-smote-algorithm-in-particular. [Accessed: 27-May-2022].

[13] "Smote explained for Noobs – synthetic minority over-sampling technique line by line: Rich Data," *Rich Data*, 04-Aug-2021. [Online]. Available: https://rikunert.com/smote_explained. [Accessed: 27-May-2022].

[14] "Decision trees: Gini vs entropy", *Quantdare*, 13-Dec-2020. [Online]. Available: https://quantdare.com/decision-trees-gini-vs-entropy/. [Accessed: 27-May-2022].

[15] "Logistic regression," *Logistic Regression - an overview | ScienceDirect Topics*. [Online]. Available: https://www.sciencedirect.com/topics/computer-science/logistic-regression. [Accessed: 27-May-2022].