Stroke Prediction with Machine Learning and Neural Networks: A Comprehensive Predictive Intelligence Framework

Ashik Sathiya, Tanav Thanjavuru

College of Information Sciences & Tech., Pennsylvania State University, State College, PA 16801, USA

abs6607@psu.edu

tzt5285@psu.edu

ABSTRACT

Stroke has a negative impact on society, and efforts have been made to improve stroke management and diagnosis. By using technology to systematically store and analyze patients' medical records, caregivers can better manage patients. This study analyzed various risk factors in electronic health records to predict strokes. Through statistical techniques and principal component analysis, the study found that age, heart disease, average glucose level, and hypertension are the most important factors for stroke prediction. A perceptron neural network using these four factors provides the best accuracy and miss rate compared to using all available input features and other algorithms. As the dataset is imbalanced, the study used sub-sampling techniques to create a balanced dataset for analysis.

Keywords: Machine Learning, Stroke Prediction, Patient Outcomes, Neural Networks

INTRODUCTION

Machine learning can play a crucial role in identifying strokes in patients by analyzing various patient data such as medical history, physical symptoms, and imaging results. There are several improvements that can come out of ML in health care. Firstly is early detection. ML algorithms can analyze large volumes of patient data and detect patterns that may indicate a potential stroke. Next is risk assessment. ML algorithms can analyze patient data to determine their risk of developing a stroke. By considering various risk factors such as age, sex, medical history, and lifestyle habits, an ML model can accurately predict a patient's likelihood of suffering from a stroke. Finally is treatment planning. ML algorithms can assist doctors in determining the best treatment plan for stroke patients. By analyzing patient data such as medical history, imaging results, and other factors, an ML model can suggest appropriate treatments based on the patient's specific needs.

This paper analyzes patient Electronic Health Records (EHR) to identify key risk factors necessary for stroke prediction. To achieve this goal, the patient records were analyzed from a publicly available dataset and used various machine learning algorithms to build predictive models for early detection of a stroke based on certain risk factors. The results showed that the machine learning models were able to accurately predict the occurrence of a stroke in a patient with a high degree of accuracy. The identified key risk factors that were highly predictive of stroke, including age, average glucose level, and smoking status. Overall, this study demonstrates the potential of machine learning techniques to aid in the early detection and prevention of strokes, which could ultimately lead to improved patient outcomes and reduced healthcare costs.

RELATED WORK

In the paper [1], authors analyzed patient attributes and their importance in predicting the

occurrence of stroke. Their dataset was released by Mckinsey and contained Electronic Health Records (EHR). They used a Learning Vector Quantization (LVQ) model and found that age, heart disease, average glucose level, and hypertension are highly important features in predicting the occurrence of stroke. The authors also computed the CHADS2 score, which is a stroke risk score, and found that patients with a CHADS2 score greater than 2 have a higher probability of stroke occurrence.

Principal Component Analysis (PCA) was used to analyze the dataset, and it was found that neural networks work best for stroke prediction using four features, namely age, heart disease, average glucose level, and hypertension. Actual features were found to provide better accuracy and lower miss rate compared to using principal components as inputs. Finally, the distribution of accuracy values obtained from the top four features used in the study was presented. Several machine learning models were used including Random Forest, Decision Tree, SVM, Neural Network, and LASSO. The results of these models were able to get accuracies of up to 80%

DATASET

The use of EHRs for building accurate stroke prediction models has gained significant attention due to the large amounts of data available in such records. A publicly available dataset of EHRs released by Kaggle [4] was utilized. The dataset contains EHR records of patients, and the output response is a binary state indicating whether the patient has suffered a stroke or not.

The electronic health records dataset used in this study contains 5110 total rows and includes patient information such as gender, age, hypertension, heart disease, ever_married status, work type, residence type, average glucose level, body mass index, smoking status, and stroke occurrence. Each record in the dataset is uniquely identified by an ID. The gender attribute can have three values: "Male", "Female", or "Other". The age attribute denotes the patient's age, and hypertension is indicated by a binary variable where 1 represents the presence of hypertension, while 0 indicates its absence. Similarly, heart disease is a binary variable where 1 represents the presence of heart disease and 0 represents its absence. The ever_married attribute indicates whether the patient is married or not, and work_type denotes the type of work the patient is engaged in, including, "Never_worked", "Private", or "Self-employed". The Residence_type attribute can have two values, "Rural" or "Urban". The average glucose level in the blood is recorded in the avg_glucose_level attribute, while the body mass index is represented by the bmi attribute. Finally, the smoking_status attribute indicates the smoking status of the patient, which can be "formerly smoked", "never smoked", "smokes", or "Unknown". The stroke attribute is a binary variable that indicates whether the patient has had a stroke, where 1 represents the occurrence of a stroke and 0 represents no stroke.The dataset has a total of 12 input features, including gender, age, hypertension, heart disease, marital status, occupation type, residence type, and average glucose level.

EXPLORATORY DATA ANALYSIS

Exploratory Data Analysis (EDA) is a process in data analysis as a part of the initial exploration and visualization of data to understand its patterns, distributions, relationships, and potential outliers. The main goal of EDA is to discover insights and relationships in data that will help with data modeling. EDA is often used to examine large datasets and identify potential trends or patterns that may not be apparent through traditional statistical analysis. For the Exploratory Data Analysis on our dataset, we focused on two main visualizations. The first visualization is a heatmap, a graphical representation of a matrix

that contains correlation values of a dataset. The second visualization that was focused on is a histogram, a bar chart that displays the frequencies or counts of values within predefined intervals or bins, used in our research to find potential feature imbalances.

The first step of our Exploratory Data Analysis was to use a heatmap to check the correlation between different features in our dataset. A heatmap is a graphical representation of data where colors are used to show the relative values of different features across a dataset. It is often used in data visualization to display large amounts of data in an easily digestible and visually appealing way.

Figure 1:

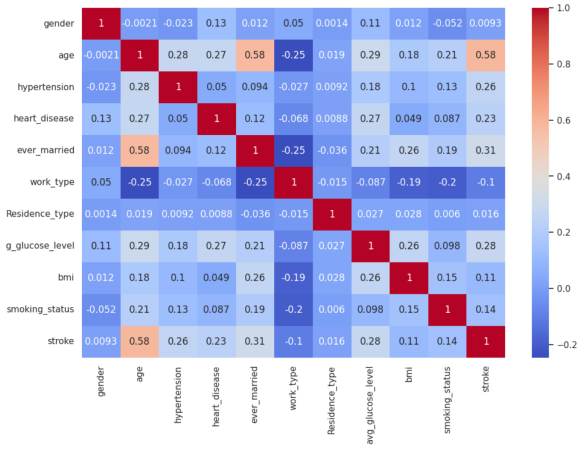
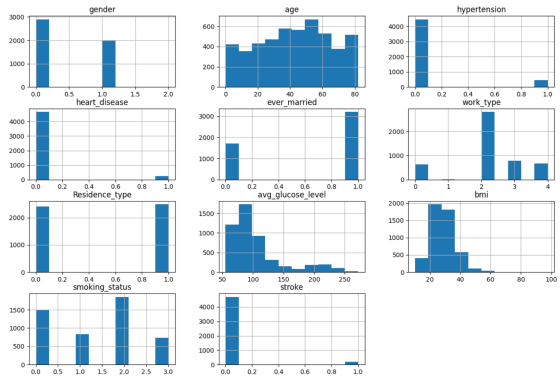


Figure 1, displayed above, is a visual representation of all of the features used in our dataset, and their correlations. The highest correlation was found between age and stroke, with a value of 0.58. The next highest correlation was between marriage and stroke, with a value of 0.31, followed by the correlation between average glucose level and stroke, which had a value of 0.28. Other factors found to have significant correlations with stroke include hypertension, with a correlation value of 0.26, and heart disease, with a correlation value of 0.23. These correlations can provide valuable insights into the relationships between different attributes and the likelihood of a stroke occurrence. This figure can also tell us which features will be playing the highest importance in influencing the final classification score of our machine learning models.

The next step of our Exploratory Data Analysis was to create a histogram to visualize the spreads of each of our features in the dataset. Each bar within the graph shows the frequency of the number of data points falling within that range. Histograms are useful for showing the shape of the data distribution, including information about the central tendency, spread, and skewness of the data. They can also help identify outliers and patterns in the data.

Figure 2:



In Figure 2, displayed above the histogram was plotted of our dataset to find any disparities within the distribution of each of our features. In the Average glucose level and hypertension graphs it was determined that both of the distributions were skewed right. In this case it is not important to fix the skewed information because it needs the outliers in the stated columns to improve the performance of our model. It was also determined that the target variable - stroke is very imbalanced compared to how it was ideally intended to look. With this information, this issue can be addressed in our data preprocessing.

Figure 3

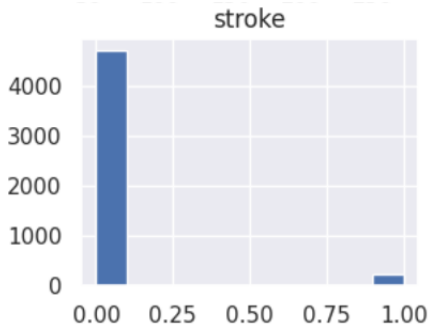
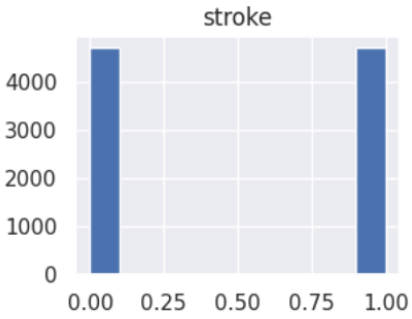


Figure 3 is a zoomed in version of the stroke visualization from the histogram in figure 2. In the graphic, it is clear that there are many more cases of negative strokes compared to positive strokes in the cleaned dataset. To implement oversampling with the target variable, we use the RandomOverSampler from the imblearn library.

Figure 4



In figure 4, displayed above, this is what the target variable distribution looks like after running the RandomOverSampler. This fixes the issue of the imbalanced data, and our data can now be passed into the machine learning model.

MACHINE LEARNING

Now that the data is cleaned and the dataset is balanced, we can now make our predictions. We decided to use two Machine Learning techniques – Neural Network and Ensemble Learning. The Neural Network architecture consists of several layers of

DATA PREPROCESSING:

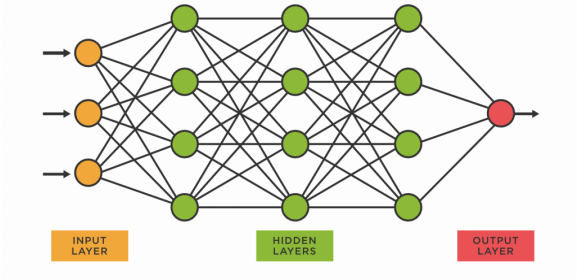
In our data preprocessing stage, we want to clean the data and make it as high quality as possible before sending it through a machine learning model. Data preprocessing is the process of cleaning, transforming, and preparing raw data for analysis. The purpose of data preprocessing is to improve the quality of data, remove inconsistencies, and make the data suitable for prediction and analysis. The first step we take to clean our data is to drop all of the nulls. We do this so that we do not have to fill the null data points with values which could be inaccurate. After dropping our nulls, we are left with around 4,909 rows of data.

The next step taken in data cleansing is to encode all of our categorical variables. Label encoding is a process of converting categorical variables into numerical values, also known as codes. The main purpose of label encoding is to convert categorical data into a numerical format that can be used as input for machine learning models. We use a Python Library called Label Encoder as a part of the sklearn module. The categories that we had to encode were ‘Gender’, ‘Ever Married’, ‘Work Type’, ‘Smoking Status’, and ‘Residence Type’. Now that these values are numerical we can pass them into a machine learning model.

The last step taken was to address the problem described in figure 2, with the imbalance target variable. To address this issue, we use oversampling. Oversampling is a technique used in machine learning to balance imbalanced datasets by increasing the number of instances in the minority class. The minority class is the class with fewer instances than the majority class, which in this case is the value of 1 in the stroke column which means positive stroke. This needs to be done because imbalance data can cause problems for machine learning algorithms that are trained on imbalanced datasets.

interconnected nodes that processes the input data and generates output predictions. Neural networks can be used for a wide variety of tasks, including image and speech recognition, natural language processing, and prediction tasks, in our case stroke prediction in patients.

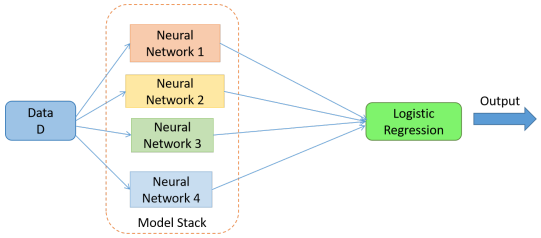
Figure 5



Neural Network Architecture

Figure 5 illustrates the Neural Network Architecture. Ensemble learning is another machine learning technique and approach we took in our data modeling that combines multiple models to improve its accuracy. Several different models are trained on the same dataset, and their predictions are combined to generate a final prediction. Ensemble learning can be used to improve the accuracy of machine learning models, as it can help reduce the impact of individual model biases and errors. It is often applied in cases where accuracy is crucial. Some examples where it is applied are financial forecasting, medical diagnosis, or risk prediction.

Figure 6



Ensemble Learning Architecture

NEURAL NETWORK

Our neural network model is a sequential model. Sequential models work well when there is a natural order to the input data. The output depends on the sequence of the input data. The architecture for the sequential model has its nodes organized linearly. In a sequential model, the output of one layer is fed as an input to the next layer in the sequence, with each layer performing its own computations on the input data. Sequential models can be built using a different variety of neural network layers. In our model we used dense layers.

Dense layers are a type of building block used in deep learning to process the input data. Every neuron in a dense layer is connected to every neuron in the previous layer. It calculates a weighted sum of the inputs plus a bias term, and then applies an activation function to produce an output. The activation function used for the first layer was reLU, and for the final output layer, Sigmoid. The reLU activation function returns the input if it is positive, and 0 if it is negative while the sigmoid function returns a value between 0 and 1, which can be interpreted as a probability. These weights and biases are adjusted during training to learn the patterns in the input data. The more dense layers you stack

together, the more complex patterns the network can learn.

To put it all together, our model is first defined as a Sequential object, which is a type of architecture that linearly stacks layers. The first layer in the model is a Dense layer with 12 neurons and applies the ReLU activation function to the output. The second layer is another Dense layer with 8 neurons, which also uses the ReLU activation function. The final layer is a Dense layer with 1 neuron, which uses the sigmoid activation function. Our model is then trained for 100 epochs with a batch size of 64. This means it passes through the training data a 100 times and the model is updated after every 64 examples.

ENSEMBLE LEARNING

As previously mentioned, Ensemble learning is a machine learning technique that combines multiple models to improve the overall predictive performance. In our case we combined cross validation and Xg boost. We first used cross validation to find the most optimal hyper parameters for our XGBoost Classifier. By using cross-validation to evaluate the performance of different hyperparameter settings, it is possible to select the settings that result in the best performance on unseen data. GridSearch Cross Validation is a technique used in machine learning to find the most optimal combination of hyperparameters for a given model. It exhaustively searches a predefined set of hyper parameters, and tests each combination using cross-validation. It then returns the combination that results in the best performance on the validation set. The optimal parameters after using GridSearchCV were (Learning Rate - 0.3, Max_Depth - 9, Alpha - 0.1). Alpha is L1 regularization term on weights. Increasing this value makes models more conservative. Max_Depth is the maximum depth of a tree. Increasing this value makes the model more complex and likely to be overfit. 0 indicates no limit. The learning rate controls the rate at which the boosting algorithm learns. These are the optimal parameters we used for the XGBoost Classifier.

The XGBoost Classifier is used for classification problems. In our case we will classify whether or not a patient will have a stroke. It combines the predictions of multiple weak decision trees to produce a strong classifier. The algorithm iteratively builds decision trees, where each tree attempts to correct the errors from the previous tree. The mean squared error and accuracy are then calculated to evaluate the model's performance.

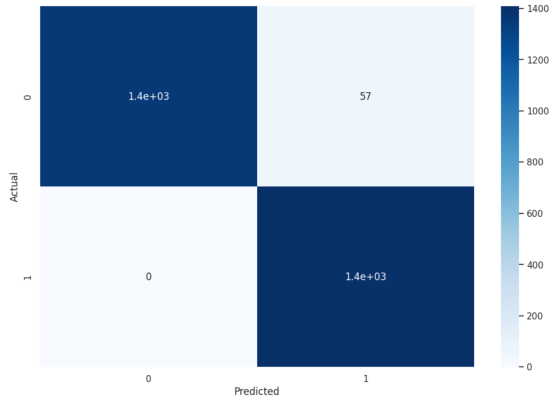
RESULTS

After examining the accuracy and Mean Squared Error of our Neural Network Model and Ensemble Learning Models, we were able to see which model deemed better results. Our Neural Network resulted in a Mean Squared Error of 0.1489 and an Accuracy of 79.67%. Our Ensemble Learning model resulted in a Mean Squared Error of 0.0195 and an Accuracy of 98.05%. After examining these results, it is clear that Ensemble Learning was the better performing model.

To validate our results, we created a Confusion Matrix for the XGBoost Model. A confusion matrix is a 2x2 Matrix that is used to evaluate and visualize the

performance of a machine learning model. It shows the True Positive, True Negative, False Positive, and False Negative and compares the predicted results to the actual results.

Figure 7



Confusion Matrix

The confusion matrix in figure 7 shows us that there were 1400 TP, 1400 TN, 0 FN, and only 57 FP. Overall our model performed very well.

ENDPOINT DEPLOYMENT

FastAPI was used to create an API endpoint for testing. The expected input format was defined in the 'Stroke_Prediction' class. It included gender, age, hypertension, heart_disease, ever_married, work_type, Residence_type, avg_glucose_level, bmi, and smoking_status in a JSON format. The endpoint then loads the XGBoost machine learning model stored as a pickle file when it starts up. Once the endpoint is started, the XGBoost model is loaded into memory. When a POST request is made to the endpoint, the input data is used to make a prediction using the loaded model, and the predicted output is returned in the response. Using this deployment, we were able to validate our model with different inputs to make sure our predictions were working as expected.

NOVELTY/FUTURE WORK

Although the Ensemble Learning Model performed very well, there were still potential improvements that could have been made that could have improved overall performance. Firstly, instead of simply dropping the nulls within the dataset during the data cleaning process, an alternative approach could have been made. This could be imputing missing values by filling in the missing values with some values that are derived from the other data points. This could involve filling the nulls with a mean or median value .The possible outcome of this would be more accurate accuracy and MSE because 200+ rows of null of data are not being dropped. There would be more data for training. With more data, models can be trained more accurately, and can lead to better model fitting and, as a result, improved accuracy in the predictions.

Another idea for the future is to use principal component analysis (PCA) on the dataset. PCA is a statistical technique used to reduce the dimensionality of a dataset while retaining as much of the original information as possible. The process takes place by transforming the original data into a new set of variables, known as principal components, which are linear combinations of the original variables. By reducing the number of variables needed to explain the variability in the data, PCA can improve the accuracy of the model by reducing the

potential for overfitting.

The final idea would be to utilize undersampling rather than oversampling when attempting to balance the data set. Undersampling is a technique used in machine learning that is used to balance uneven datasets by keeping all of the data in the minority class and decreasing the size of the majority class. Undersampling can potentially improve model performance by reducing the impact of the majority class and reduce overfitting by preventing the model from memorizing the majority class.

CONCLUSION

By using machine learning and neural networks in an attempt to classify data points to predict the result of a stroke occurrence, we were able to predict at a 98% accuracy. With results like these, medical professionals can successfully predict strokes using certain risk factors. This result could lead to better patient outcomes and the further study of risk factors and their potential to cause a stroke. Medical professionals could also affirm their own diagnoses and get a better perspective on the patient. By using this algorithm, hospitals or government agencies could promote healthy lifestyles and be able to push initiatives to help reduce the amount of strokes in general. This research also proves to be a stepping stone in bridging the crossroad between artificial intelligence and the medical field. Since strokes were able to be predicted using machine learning, there are many other types of diseases or medical conditions which could have the potential to include machine learning for patient diagnosis. Some of these diseases could be Cancer, diabetes, heart disease, Alzheimer's disease, and mental health disorders. All of our work is available in [5].

REFERENCES

[1] A nationwide deep learning pipeline to predict stroke and COVID-19 death in atrial fibrillation. Alex Handy, Angela Wood, Cathie Sudlow, Christopher Tomlinson, Frank Kee, Johan H Thygesen, Mohammad Mamouei, Reecha Sofat, Richard Dobson, Samantha Ip, Spiros Denaxas, medRxiv 2021.12.20.21268113; doi: https://doi.org/10.1101/2021.12.20.21268113

[2] A predictive analytics approach for stroke prediction using machine learning and neural networks. Soumyabrata Dev, Hewei Wang, Chidozie Shamrock Nwosu, Nishtha Jain, Bharadwaj Veeravalli, Deepu John, Healthcare Analytics, Volume 2, 2022, 100032, ISSN 2772-4425, https://doi.org/10.1016/j.health.2022.100032

[3] Stroke risk prediction using machine learning: a prospective cohort study of 0.5 million Chinese adults. Matthew Chun, Robert Clarke, Benjamin J Cairns, David Clifton, Derrick Bennett, Yiping Chen, Yu Guo, Pei Pei, Jun Lv, Canqing Yu, Ling Yang, Liming Li, Zhengming Chen, Tingting Zhu, the China Kadoorie Biobank Collaborative Group, Journal of the American Medical Informatics Association, Volume 28, Issue 8, August 2021, Pages 1719–1727, https://doi.org/10.1093/jamia/ocab068

[4]
https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset

[5]
https://github.com/Tanav1/340W_Final_Presentation

.