Predicting Stroke through Medical and Demographic Data

Seth Singson-Robbins
Fordham Univesrity
ssingsonrobbins@fordham.edu

Abstract—This paper investigates a way to create a classification model from a medical dataset if someone has already had a stroke with the value of the model being to help predict the people's likelihood of having a stroke in the future to help with prevention. Classification will be based on a mix of demographic and medical information of the patients within the dataset and the goal of the model to have a strong recall rate when predicting if future patients have already had a stroke. The model will utilize several classification algorithms from Python's Ski-Kit Learn package [1] along with a range of sampling techniques and missing data strategies from the Pandas [2] and ImLearn[3] packages. The result of the model is evaluated using 20% of the dataset as test data and looking and picking the model that has the highest Area Under the ROC curve. The final model will have a probability threshold picked that maximizes the True Positive rate (which in the results is set to 0.8) while minimizing the False Positive rate.

I. Introduction

According to the Center for Disease Control and Prevention, about 695,000 patients suffer from a stroke every year in the United States (4) and killing over 150,000 people. This making up one out of every 19 deaths from all causes in 2019 (5). Although age and family history are major but unpreventable factors, Harvard Medical School has determined that there are many ways to prevent stroke including weight loss, quitting smoking, and exercise (6). Knowing what types of people have already had a stroke could help predict if someone could have a stroke in the future allowing for focusing the patients more susceptible to having a stroke to receive more resources and programs that would help prevent them from having a stroke in the future.

Predicting stroke can get complicated which is why we do not have a set-in stone model already in place. According to Amarenco P., author of another research paper on stroke classification, there are over 150 known causes of stroke and, even among those 150 causes, there are four subcategories of what is currently classified as a stroke making it a much more complicated medical issue than a single disease. They also discovered that, of the patients who were discovered to have had a stroke, doctors were not able to determine the cause of the stroke in 25-39% of the patients.(7) Issues like this complicate prevention as stroke is a broad category but not easily identifiable.

II. Methodology

This section looks at the additional work done to run the model. This includes how the data was collected, what preprocessing was done to the data, training versus test data, feature selection, oversampling techniques, hyperparameters, and what models were tested for final evaluation.

A. Data Collection

Data was collected from the Kaggle database and contains 5,110 examples of patients who either did or did not already have a stroke. (8) The features available were a wide range that included medical and demographic information of the patient that included, but not limited to, gender, age, whether they smoke, and other cardio medical conditions the patient has such as hypertension and heart disease. This information has proven indicative of people more likely to have had a stroke and, in aggregate, could lead to a stronger classification model. Within the dataset, 5% of the patients have already had a stroke, leading to an imbalanced dataset. Discussions on the methods of how the data was rebalanced will be addressed in the oversampling section in part E.

B. Preprocessing

Given the nature of the data and the models used for classification, preprocessing of the dataset is required for many models to run and for the models to both converge efficiently and have stronger predictability.

Due to the nature of how Ski-Kit was designed, many models require that all the features be numerical or binary and, if not done prior to running the model, will cause the model to error out. To account for this, the Pandas package has a function called 'Get Dummies' [2] which changes all categorical features in a dataset into binary features. This means that, for each category within a feature, a new feature is created within the dataset setting it to 1 for each patient that identifies in that category and setting it to 0 otherwise. This means that there are a lot more features in the dataset but that they still portray the same information for analysis and modeling.

Additionally, the features have varying numerical ranges. For example, the average Glucose Level has a range from 45.28 to 271.74 which is much wider than the new dummy categories that are 0 or 1. These differences can lead to the features with wider ranges to have more influence on the modeling which can hurt the final performance of the models due to the bias. It could also lead to some of the models like Logistic regression to not converge due to the nature of the how the model determines the best fit. The python package Ski-Kit Learn has a function called 'Standard Scaler' [(1)] which updates the features to have the same range, changing the values to be around the mean of 0 and each feature value to show the variation from the mean via a standard deviation of 1. This makes all the features have consistent ranges with the range portraying the variability of an example compared to the rest of the dataset. This removes the bias the features could have due to inconsistent range and keeps the important information retained in the dataset for each patient. All the preprocessing techniques make for more efficient models and data analysis.

C. Training and Evaluation

Several models and methods to set up the data are necessary to find the optimal model to help predict if someone has had a stroke. To help with the evaluation of the numerous models, 20% of the dataset was separated from the training data to help with the final evaluation of all the models. This data was stratified, so the proportion of patients who have had a stroke versus the number of patients who have not had a stroke in both the training set and test set match the original dataset.

D. Feature Selection

Some of the features in the dataset need to be addressed before modeling can begin. The feature 'ID' is irrelevant to classification as it is just a number assigned to each patient. Thus, this feature is removed.

The feature BMI has a significant number of null values (4% of the data). Null values cannot be evaluated in some models, so this needs to be remedied. One method is to remove the BMI feature entirely from the modeling process. While this would solve the null value issue, it is not always optimal as the BMI field could be indicative to if someone has had a stroke. In this dataset, the patient is 4.7 times more likely to already have had a stroke than patients where the BMI is known.

The other option is to nest an additional classification model to predict the values of patients that do not have their BMI information available. This was done with a Linear regression model utilizing the other features available. This can get complicated so, to minimize complexity of the model, we will assume that a simple Logistic regression is accurate and just test both predicting the BMI field and removing the BMI feature as two separate groups of models and utilize the test data to decide on the final model.

E. Oversampling Techniques

The dataset is unbalanced with people who have not had a stroke making up 5% of the dataset. This could lead to major issues in modeling as the people who have not had a stroke will make up most of the datapoints and will have a much stronger influence on the algorithm. If this occurs, the final models will have a much lower recall rate and means people who already have had a stroke are much less likely to be identified in the model. Due to the low number of examples of patients who have already had a stroke, oversampling is the best solution to remedy this. There are several techniques that could be done to help create more examples, and three methods which will be used and compared in the final evaluation of the models. All three methods utilize Python's imbalanced-learn package which is made to help create balanced datasets and leads to a 50/50 split between people who have and not already have had a stroke. [(3)] Additionally, datasets that have no oversampling will also be tested to confirm if oversampling is optimal to have better performance than the three oversampling methods.

The first method is to oversample the data with replacement. This is done by separating out the examples of patients who have already had a stroke and sampling with replacement until the number of examples to equal to the number of patients who did not already have a stroke. This is the cleanest way to oversample the data but could lead to overfitting since the examples of people who already had a stroke will have a large proportion of influence on an individual basis.

The second method utilizes the SMOTE algorithm. The algorithm, which stands for Synthetic Minority Oversampling Technique, utilizes the n-nearest neighbors algorithm to find additional artificial samples within the class and utilizing them as additional examples to train the model.

The third method utilize the ADASYN, or Adaptive Synthetic, algorithm. This algorithm is very similar to SMOTE with the main difference being that it adds additional random noise into each new synthetic example allowing for more variance between the artificially created data points.

SMOTE and ADASYN can be very useful in preventing overfitting and help the model create additional datapoints that help create a more balanced model without too much over presentation from individual patients. However, they can be inaccurate, especially for datasets with many features, due to classes possibly overlapping and leading to new examples that, in the real world, would be outside the boundaries of what would be identified as a patient who already had a stroke but did not (False Positive) as noted in Giorgio Pilotti's article. [(9)] Due to the pros and cons of each method, we will be testing all three methods in our modeling and utilizing the test data to determine which model to use as our final classification model.

H. Hyperparameters

For each of the models used for classification, hyperparameters are used to help ensure that the model has the best predictive power and thus need to be tuned. To ensure that the right hyperparameter is used, the training set went through

cross validation through Ski-kit Learn's GridSearchCV. [2] This separates out the training set into 5 equally sized datasets (20% of the total dataset for each subset) and uses each subset as the test set while training the rest of the data to see how it does in 5 different model runs. GridSearchCV then finds the best hyperparameter based on a given metric which for our models was the ROC-AUC curve. It also provides the best score for the best hyperparameter to see how it would perform on the training data. For each run, Ski-kit Learn also has a pipeline method that ensures that the oversampling is done after the data has been separated out and only on the training set to ensure that the oversampling does not influence and bias the stand in test set.

Each model has its own separate hyperparameters. These will be covered in more detail in the next Final Models section and the final hyperparameters used are shown in Figure 1 of the results section.

F. Final Models

For the actual classification model, we will utilize three different data mining models to looking to see how each performs: Logistic regression, neural networks, and random forests. All three of these will come from Python's Ski-kit Learn package. [1]

Logistic classification utilizes all the features to create a multi-dimensional line that splits the data between the two categories. The model has a hyperparameter 'C' which determines the inverse of the regularization strength of a model with a small number having stronger regularization. Several options were tested by multiples of 10 from 1e-5 to 1e2. This is a best fit model and does not manipulate the space of the features which limits the expressiveness and accuracy of the model. However, it is one of the simpler data mining models and could lead to strong results and limits overfitting. It also shows the slope that each feature contributes to the model which helps with describing the features and could contribute to further insights into how the model determines its classification.

Neural networks use multiple layers of information from combinations of the features finding better ways to both manipulate the space of the features and find how the interactions between the features make an impact on how the examples are classified. The model has the learning rate hyperparameter tuned to show how quickly each iteration affects the final model. The options tested were in multiples of 10 from 1e-4 to 1. While this can lead to stronger predictability it could also lead to overfitting.

Random forests are an ensemble model meaning that it utilizes many data mining models and aggregates them into a final model. This model specifically creates many decision trees varying the features and the training examples used for analysis splitting up features by binary splits until every case in the dataset can be categorized via leaves and pathways which are the decisions. Random forests take all the decision trees and aggregates the models via a majority vote to categorize future examples. This model has two hyperparameters: n estimators and max depth. N estimators determines how many decision trees are created for the final analysis while the max depth is the maximum number of splits each decision tree can have. The options for n estimators tested were 10, 100, 500, and 1,000 while the max_depth tested were 1, 2, 5, 10, 20, 50, 100, and 1,000. While ensemble classification models help lower variance and minimize overfitting, both neural networks and random forests are a black box which prevents insights and analysis of how individual features help determine what category is chosen. All three models will be used and evaluated to help determine the final model.

Between how to approach the BMI feature, oversampling methods, and the three data mining models, a total of 24 models will be created and evaluated to help determine which combination of methods best predicts if someone in the dataset has already have had a stroke.

III. Results

The results of the model have been evaluated by looking at the model's training performance as well as the test set performance. Due to an imbalanced dataset and to help conduct a more accurate evaluation of the models to classify future examples (and with the final model being tuned to a lower probability threshold than its default of 1), the main metric used is the Area Under the Curve of a Receiver Operating Characteristic, or ROC curve. The ROC curve plots two metrics, the True Positive Rate, and the False Positive Rate, on a 2D line chart. The two metrics complement each other when shifting the sensitivity of the model's probability threshold for the final classification models. As the True Positive Rate increases, the False Positive rate can also increase, and the ROC curve plots this relationship with the intention of finding a threshold that balances the two metrics finding the best final predictor of future examples. We will do this after determining which is the best model to utilize for classification.

The Area Under the Curve, or AUC, show how strong the model is at predicting classification on the models' varying thresholds and is based on the ROC curve. It looks at the area under the curve with a higher score indicating that the various True Positive rates are less likely to have a negative impact on the False Positive rate. If the model chooses the category at random, the expected ROC would be a straight 45-degree line. The AUC for models of random guessing would have a score of 0.5, so any score higher than 0.5 would be an improvement on the model which is why this chosen as the model's baseline comparison. We will be evaluating the performance of each model in terms of the ROC-AUC on the test data for each model and, once the final model is determined, used the ROC-AUC to pick the best threshold point of the model for classification of future patients. The final performance will compare the random guessing baseline to the final model based on its AUC and the False Positive rates of both models based on the chosen True Positive rate of predicting when someone already has had a stroke in the test data.

We will first look at the performance of the models on their own training datasets through Cross-Validation. Figure 1 shows the hyper-parameters used based on having the strongest ROC-AUC. These vary by the type of dataset and model that

was used and the final model will utilize what is listed as the top parameter.

Pre-Processing	Hyperparameters Used		
	Logistic Regression	***Neural Network***	***Random Forest***
BMI Predicted Oversampling with Replacement	C: 10	Learning Rate: 0.0001	Estimators: 100 Max Depth: 5
BMI Predicted SMOTE Oversampling	C: 0.1	Learning Rate: 0.0001	Estimators: 100 Max Depth: 2
BMI Predicted ADASYN Oversampling	C: 1	Learning Rate: 0.0001	Estimators: 500 Max Depth: 5
BMI Predicted No Oversampling	C: 10	Learning Rate: 0.0001	Estimators: 500 Max Depth: 5
BMI Removed Oversampling with Repalcement	C: 10	Learning Rate: 0.0001	Estimators: 1000 Max Depth: 5
BMI Removed SMOTE Oversampling	C: 1	Learning Rate: 0.0001	Estimators: 500 Max Depth: 5
BMI Removed ADASYN Oversampling	C: 0.01	Learning Rate: 0.0001	Estimators: 500 Max Depth: 5
BMI Removed No Oversampling	C: 1	Learning Rate: 0.0001	Estimators: 100 Max Depth: 5

Figure 2 shows the ROC-AUC performance of each of the data sets and models based on the top hyperparameter. Across the board logistic regression had the strongest AUC curve.

Figure 2. AUC Performance on Training Data from Cross Validation

Pre-Processing	ROC-AUC from Training Data from Cross Validation		
	Logistic Regression	***Neural Network***	***Random Forest***
BMI Predicted Oversampling with Replacement	0.839	0.812	0.825
BMI Predicted SMOTE Oversampling	0.837	0.809	0.807
BMI Predicted ADASYN Oversampling	0.836	0.798	0.807
BMI Predicted No Oversampling	0.840	0.825	0.833
BMI Removed Oversampling with Repalcement	0.838	0.803	0.820
BMI Removed SMOTE Oversampling	0.834	0.797	0.807
BMI Removed ADASYN Oversampling	0.834	0.785	0.808
BMI Removed No Oversampling	0.839	0.820	0.827

To avoid picking a model that would produce overfitting and to ensure that the data would work for future patients, the training models were modeled out and then the data was predicted on the test data to see how the modeled performed. The results were very similar to the cross validation set which is logical as it also had data separated from the rest of the data as each validation set ran. The strongest model was Logistic Regression when no oversampling methods were used, and the BMI was predicted with a ROC-AUC of 0.841. It is interesting that, even though the dataset is very unbalanced, oversampling methods did not help with performance.

Figure 3. AUC Performance on Test Data

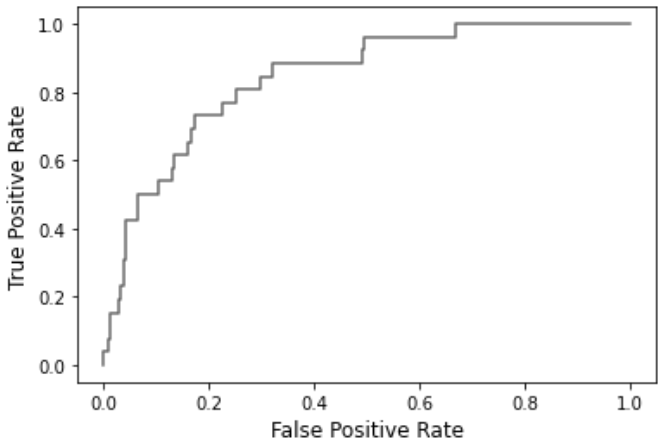
Pre-Processing	ROC-AUC on Training Data		
	Logistic Regression	***Neural Network***	***Random Forest***
BMI Predicted Oversampling with Replacement	0.835	0.816	0.836
BMI Predicted SMOTE Oversampling	0.838	0.818	0.812
BMI Predicted ADASYN Oversampling	0.835	0.820	0.813
BMI Predicted No Oversampling	0.841	0.840	0.836
BMI Removed Oversampling with Repalcement	0.818	0.805	0.813
BMI Removed SMOTE Oversampling	0.817	0.808	0.801
BMI Removed ADASYN Oversampling	0.811	0.813	0.801
BMI Removed No Oversampling	0.821	0.832	0.816

Now that the model has been picked, the threshold point needs to be determined. The threshold looks at, assuming the information from the model is accurate and given the information of a new patient, what is the probability that the patient would be classified as already having a stroke. Usually, models assume a threshold of 1 for models but this could lead to very poor performance due to such a high threshold.

Picking the threshold is very important and can vary by what the model is portraying. In the medical field, True Positive needs to be as high as possible as, if they do not detect if someone has a specific medical condition, it could lead to people getting sicker or even death. For public health, conversing resources in very important and a high False Positive rate could lead to a lack of confidence in the modeling

and usually would mean that additional tests would need to be done on patients which could be both expensive or invasive for either the patient or the organization. Figure 4 shows the ROC-AUC curve of the model below. As one can see, there is a huge drop off in False Positive Rate performance when the True Positive Rate is above 0.8 so the final model will utilize 0.8 for the model's final probability threshold and for the final evaluation.

Figure 4. ROC-AUC Curve of Final Model



When the True Positive Rate is picked at 0.8, it leads to the threshold to be set to 0.550 and the test data having a False Positive rate of 0.25. Even though the False Positive rate is still substantia, a model that uses random guessing and a ROC-AUC of 0.5, the True Positive Rate of 0.8 would lead to a False Positive Rate of 0.8. This means that this model has a 0.55 lower False Positive rate than the random guessing model and is substantially better at predicting if someone has already had a stroke from the dataset.

IV. Related Work

There have been many additional analysis and classification models created for determining risk of stroke and even for this dataset. Some of the analysis on this dataset included more advanced modeling like boosting and Principal Component Analysis that would help create a more precise model but is beyond the scope of this paper where we try to have more simpler models to help create final classification models.

Additionally, there have been many research studies that go beyond just looking at if someone has a stroke. Jae C. Chang of the NCBI found a way to diagnosed different types of strokes, which was separated into phenotypes, utilizing an array of medical information about the patient such as vascular damage and hemorrhages. [(10)] Louis R. Caplan created a model that assigns a grade (0-3) to determine the cause of a stroke occurring based on the information provided.[(11)] Amarenco went even deeper at identifying 150 different types of strokes with an array of additional medical data provided by the patients.[(7)] A lot of these studies look at digging into the different types of strokes and classifying them which help with deeper medical studies but not a lot of classification models have been created to do a broader analysis to determine if someone is more susceptible of a stroke, something that would be more helpful when deep medical knowledge of the patient is not available without extensive testing which is usually the case.

V. Conclusion

From all the models used, the model that yielded the best results was the Logistic Regression model when the null values for BMI were predicted and oversampling was performed by stratified sampling of the patients who already had a stroke with replacement. This led to a ROC-AUC of 0.841. When picking a True Positive rate of 0.8, it led to a probability threshold of the model to be chosen at 0.550 and a False Positive rate of 0.25, a 0.55 improvement than random guessing.

Developing a classification model like this has a lot of use cases in the medical, public health, and even insurance industries. It can be used to help determine if someone is more likely to have a stroke in the future and could lead to life saving changes in their life through programs that could lead to weight loss or quitting smoking to treating other but relevant medical conditions like heart disease. These programs could be very expensive, but modeling can help determine who would have the highest likelihood of benefiting from the program leading to lifesaving programs at a minimal cost. The public health community can allocate resources more effectively and insurance companies can determine that these preventative measures have a strong ROI due to lower insurance claim costs, so they are more likely to cover the cost of these programs to mitigate risk of paying out for hospital bills due to stroke in the future.

Given that this is a simple classification model, there are several things that could be done for further modeling and research. Boosting and other models could be joined together to create a stronger ensemble model. Feature selection at the beginning such as Principal Component Analysis could have been used to help determine which features could be used in the model and minimize bias from redundant or irrelevant features creating more noise. If access to the patients is available for future research, one can look to see if, for the patients that were misdiagnosed as already having had a stroke, if they have a stroke sometime soon and thus proves that this is also a good model for determining if people have already had a stroke. According to the CDC, 185,000, or about 1 in 4 strokes were recurrent meaning that these patients have already had a stroke before this latest one.[1] Prevention measures could also be tested on this population to see if, given we know that someone has already had a stroke, do the prevention measures and programs help decrease the probability that they will get a stroke in the future compared to people who were not in these prevention programs. Further analysis is very important to show both the validity of this classification model and the how useful it is in the real world in making a positive difference on this deadly disease.

VI. References

(1) “Pandas Python Package Documentation,” pandas. [Online]. Available: https://pandas.pydata.org/. [Accessed: 07-Dec-2021].

(2) “Skikit-Learn Python Package Documentation,” scikit. [Online]. Available: https://scikit-learn.org/stable/. [Accessed: 07-Dec-2021].

(3) “Imbalanced-Learn Python Package Documentation,” imbalanced. [Online]. Available: https://imbalanced-learn.org/stable/. [Accessed: 07-Dec-2021].

(4) “Stroke facts,” Centers for Disease Control and Prevention, 25-May-2021. [Online]. Available: https://www.cdc.gov/stroke/facts.htm. [Accessed: 07-Dec-2021].

(5) “Know the facts about stroke,” Centers for Disease Control and Prevention, 03-May-2021. [Online]. Available: https://www.cdc.gov/stroke/facts_stroke.htm. [Accessed: 07-Dec-2021].

(6) “7 things you can do to prevent a stroke,” Harvard Health, 26-Jun-2020. [Online]. Available: https://www.health.harvard.edu/womens-health/8-things-you-can-do-to-prevent-a-stroke. [Accessed: 07-Dec-2021].

(7) P. Amarenco, J. Bogousslavsky, L. R. Caplan, G. A. Donnan, and M. G. Hennerici, “Classification of stroke subtypes,” Cerebrovascular Diseases, 03-Apr-2009. [Online]. Available: https://www.karger.com/Article/Fulltext/210432. [Accessed: 07-Dec-2021].

(8) Fedesoriano, “Stroke prediction dataset,” Kaggle, 26-Jan-2021. [Online]. Available: https://www.kaggle.com/fedesoriano/stroke-prediction-dataset. [Accessed: 07-Dec-2021].

(9) G. Pilotti, “Oversampling and undersampling: Adasyn vs Enn,” Medium, 17-Feb-2020. [Online]. Available: https://medium.com/quantyca/oversampling-and-undersampling-adasyn-vs-enn-60828a58db39. [Accessed: 07-Dec-2021].

(10) J. C. Chang, “Stroke classification: Critical role of unusually large von Willebrand factor multimers and tissue factor on clinical phenotypes based on novel ‘Two-path unifying theory’ of hemostasis,” Clinical and applied thrombosis/hemostasis : official journal of the International Academy of Clinical and Applied Thrombosis/Hemostasis, 2020. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC7427029/. [Accessed: 08-Dec-2021].

(11) L. R. CaplanMD, L. R. Caplan, Louis R. Caplan From the Beth Israel Deaconess Medical Center, and C. to L. R. Caplan, “Stroke classification,” Stroke, 16-Dec-2010. [Online]. Available: https://www.ahajournals.org/doi/full/10.1161/strokeaha.110.594630. [Accessed: 08-Dec-2021].