Classifier for Predicting Stroke Risks

Jueshen Hou

2022-12-09

Contents

List of Tables

Abstract

Stroke, defined as a brain disease that happens when the blood supply to part of the brain was blocked or when a blood vessel in the brain breaks, is a leading cause of death and a major cause of severe disability among adults in the United States. According to CDC, more than 795000 people suffer from stroke every year in the United States. While being regarded as a factor that leads to serious consequences and even mortality, stroke is preventable and treatable (CDC). Therefore, it is important to conduct screening on the population to identify people who have a higher risk of having stroke and provide preventive support to these people. The primary goal of this project is to build a classifier for medical professionals to conduct screening for the risk of stroke. Three techniques, the k-Nearest Neighbors, penalized regression and bagged forest was implemented to build this classifier and their performance metrics were compared. We found the model built through the bagged forest has a better performance in predicting the stroke risk status of people and suggest using it to build the classifier, while improvements can still be made through collecting more data to mitigate the impact caused by imbalanced data set.

Section 1: Data and Motivation

Our goal for this project is to build a classifier that helps to predict whether someone is at a higher or lower risk of having a stroke based on the health condition, health measures, and related information provided. This classifier can be utilized to assist in the prevention of stroke events along with other testing techniques and facilitate the diagnosis process through providing a preliminary screening, which can help medical professionals to identify people who are more likely to have a stroke, and implement preventive care if needed.

The data that we are dealing with consists of 5110 rows and 12 columns. The data set records the 12 features(4 numeric features and 8 categorical features) of 5110 patients who have either had suffered or have not suffered from stroke. We have information on the patient's biological gender, the age of each patient in years, whether a patient has heart disease/hypertension, marital status, smoking status, stroke status, and other related information. For other features not mentioned, please refer to the following link for a detailed description of features in this data set (Fedesoriano). Link: https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset

Below is a table of all the features and their descriptions from the data set.

Table 1: Features and Descriptions

Features	Descriptions
id	unique identifier
gender	Male, "Female" or "Other"
age	age of the patient
hypertension	0 if the patient doesn't have hypertension, 1 if the patient has hypertension
heart_disease	0 if the patient doesn't have any heart diseases, 1 if the patient has a heart disease
ever_married	"No" or "Yes"
work_type	children, "Govt_jov", "Never_worked", "Private" or "Self-employed"
Resident_type	Rural or "Urban"
avg_glucose	average glucose level in blood
bmi	body mass index
smoking_status	formerly smoked, "never smoked", "smokes" or "Unknown"(Unknown means that the information is unavailable for this patient)
stroke	1 if the patient had a stroke or 0 if not

Section 2: Data Cleaning and Exploratory Data Analysis (EDA)

Data Cleaning

We started with converting the data types of several features to more appropriate types.

Table 2: Features Need to Convert to Other Type

Features	Type
hypertension	integer
heart_disease	integer
stroke	integer
bmi	character

There are three features that need to be converted from numeric data to categorical data, which are hypertension, heart_disease, and stroke. These three features were converted from numeric type to categorical type. Other categorical features that are originally in character type were also converted into factor type for the model building later.

We then check for missing in the data set. The result was shown in the table below.

Table 3: Number of Missing Values

x
201

Figure 2.1 Graph of Missing Values in Each Column

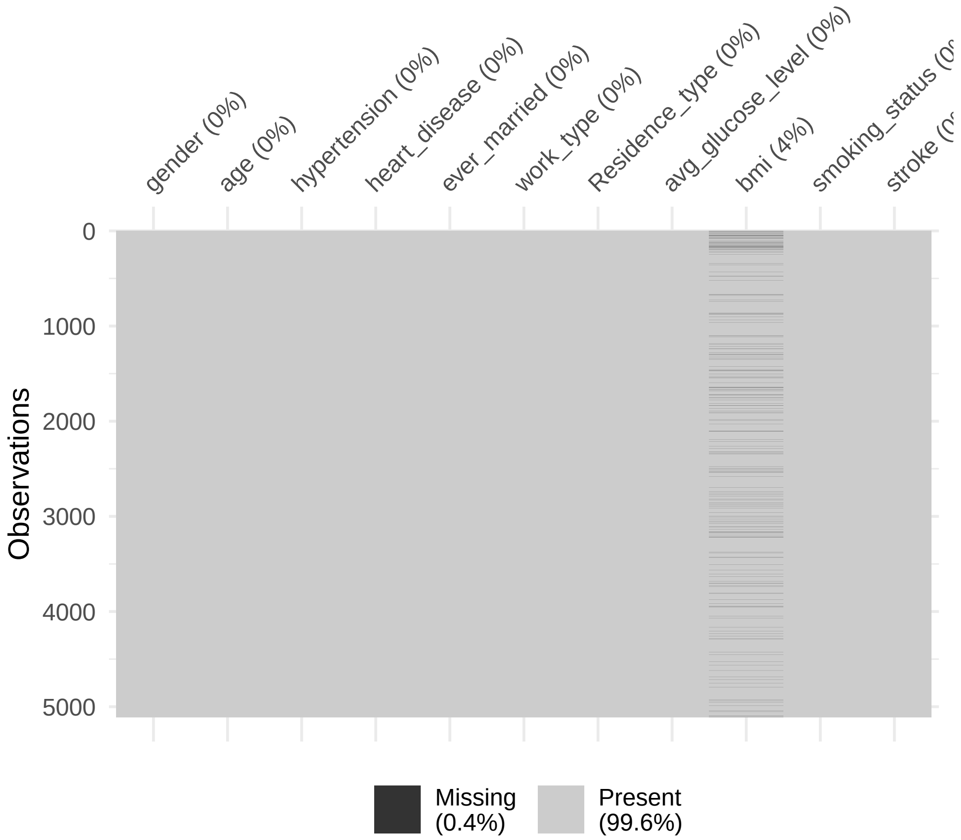
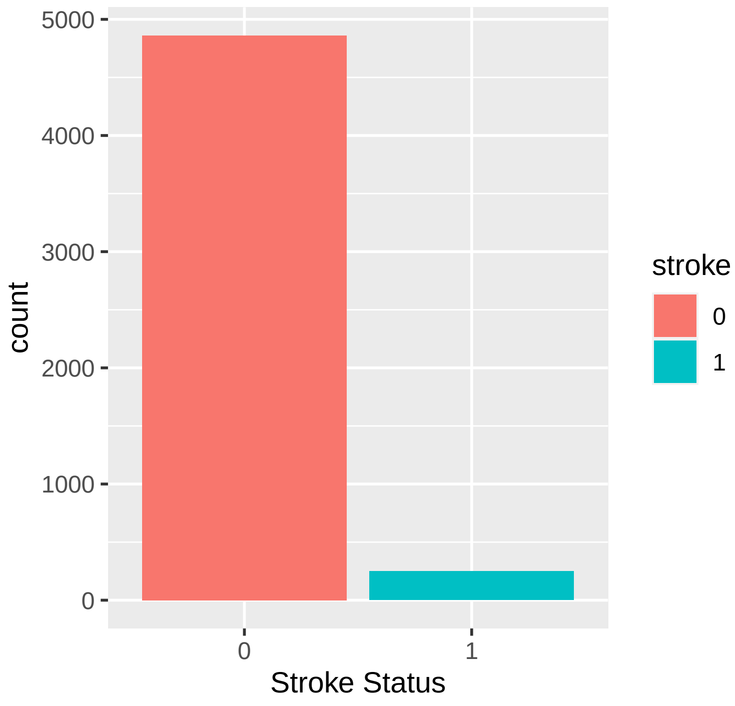


Table 4: Stroke Status of Rows with Missing Value

Var1	Freq
0	161
1	40

Table 3 and Figure 2.1 show that 201 rows of missing values were found from the data set, which was only present in the column "bmi". According to Table 4, there are 161 patients who suffered from stroke events and 40 patients who did not suffer from stroke in patients with their BMI value missing(Here, 0 = the patient have not had stroke, 1 = the patient had a stroke). We decide to remove these rows with missing values in BMI.The data set after the removal of missing data contains 4909 rows of data.

Figure 2.2 Classes of Stroke



We can see from Figure 2.2 that the class of stroke status was imbalanced, therefore, we utilized the technique called Synthetic Minority Oversampling Technique(SMOTE) to synthesize data points for the minority class, therefore increasing the proportion of the minority class in the data set and enable us to build a usable classifier.

The SMOTE technique is an oversampling technique that utilizes the technique called k-Nearest Neighbors algorithm to synthesize data. In this project, we choose to use SMOTE_NC, which is a modification of SMOTE that is able to handle data sets with both categorical and numeric features. For numeric features in the data set, SMOTE_NC randomly chooses k data points from the minority class in the data set and synthesizes new data points on the line segment between each of the two data points. For categorical features, SMOTE_NC selects the most frequent category of the nearest neighbor data points and assigns it to the new synthesized data point (Imbalanced Learn).

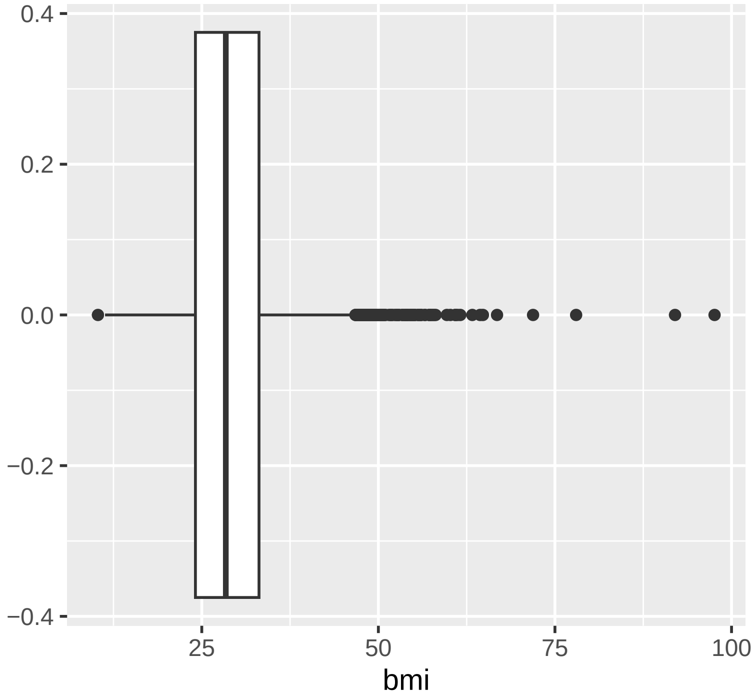
This technique allows us to mitigate the imbalance in the data set with less concern on the issue of overfitting in regular oversampling that make the model giving better predictions on training data but failed to make correct prediction on new data (Wijaya).

An experimental package created by Dongyuan Wu from github was utilized to conduct SMOTE_NC for our data set that consists of both categorical and numeric features (Wu).

EDA

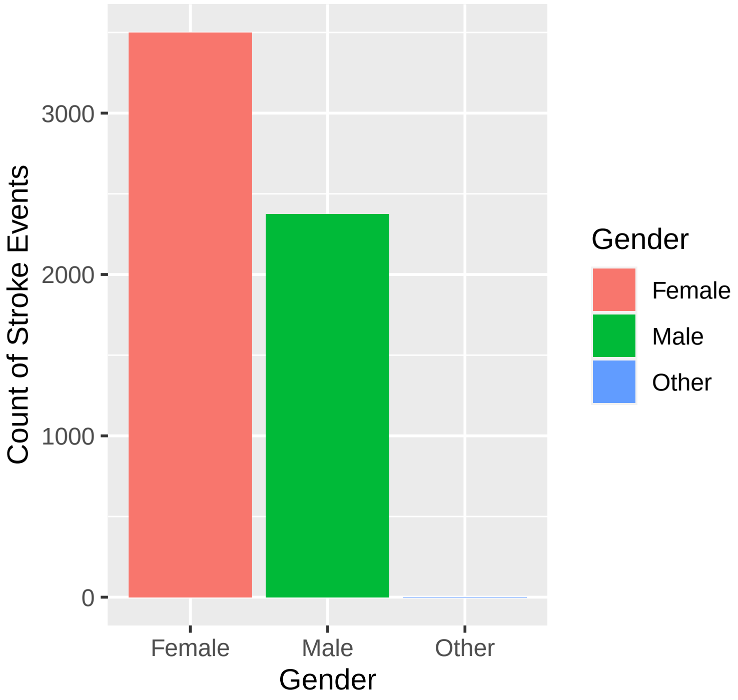
We begin exploratory data analysis to explore characteristics of features in the cleaned data set.

Figure 2.3 Distribution of BMI



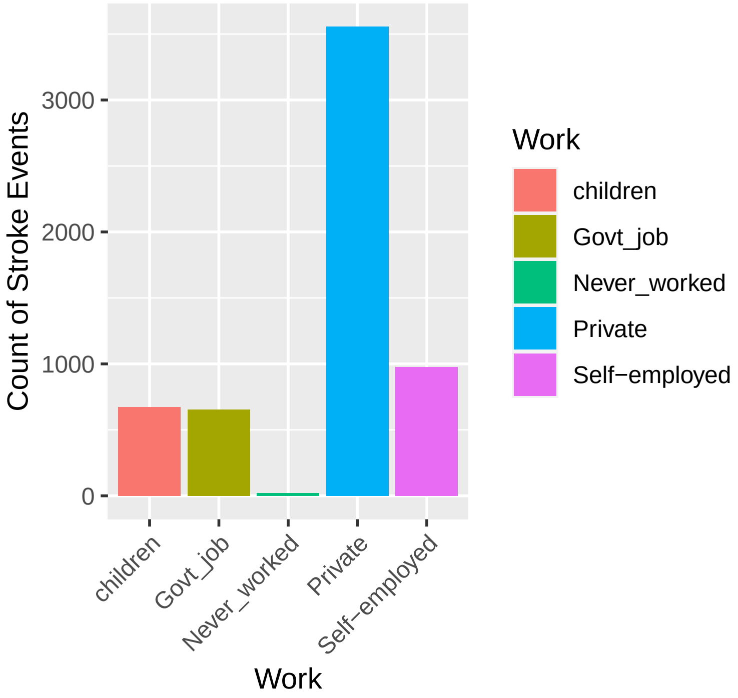
We can see from Figure 2.3 that the distribution of patients' BMI is slightly rightly skewed.

Figure 2.4 Stroke Event Across Gender

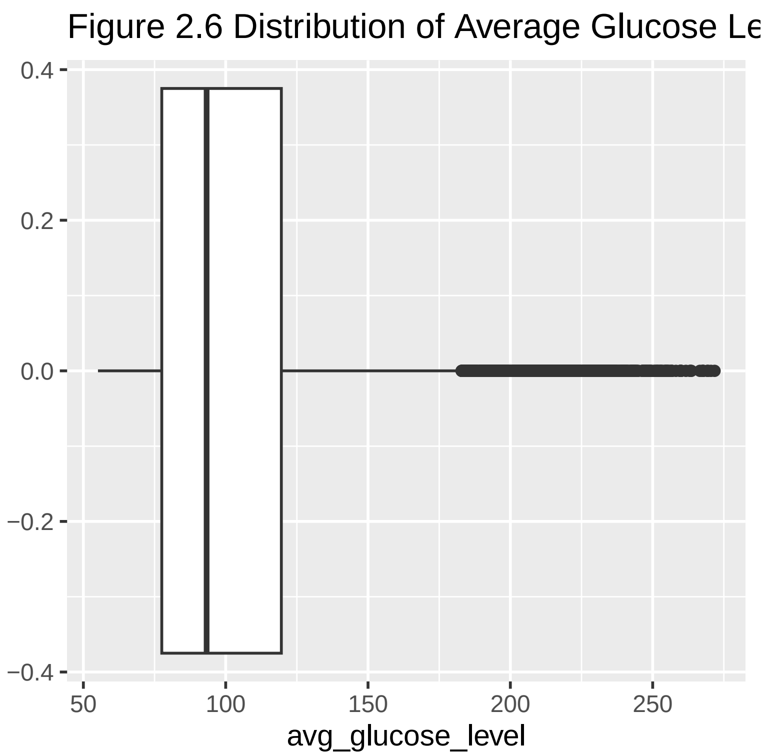


We can see from Figure 2.4 that there are more female patients among all patients who had stroke. Additionally, one patient with gender “Other” exist in the data set.

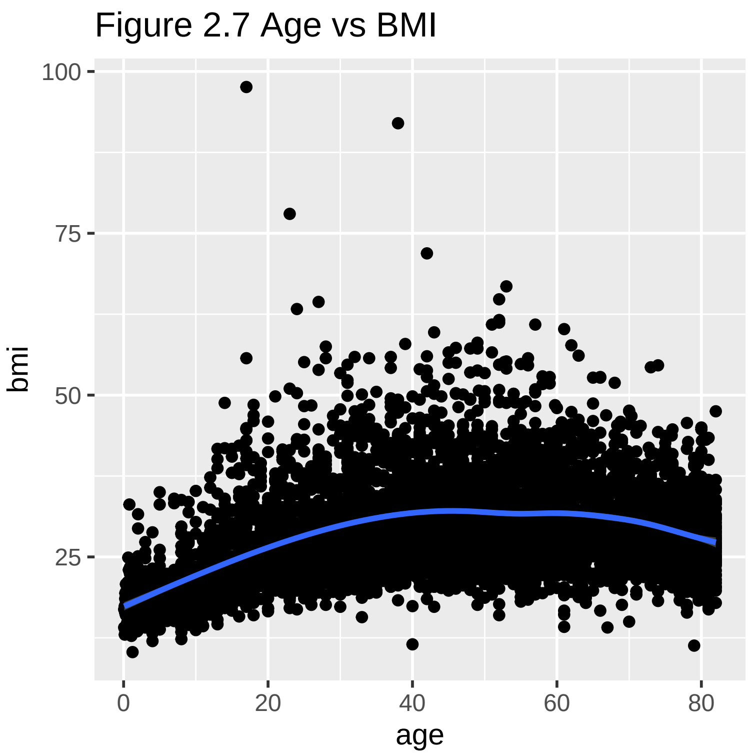
Figure 2.5 Stroke Event Across Work Type



We can see from Figure 2.5 that their are more patients who had stroke that has private as their work type.

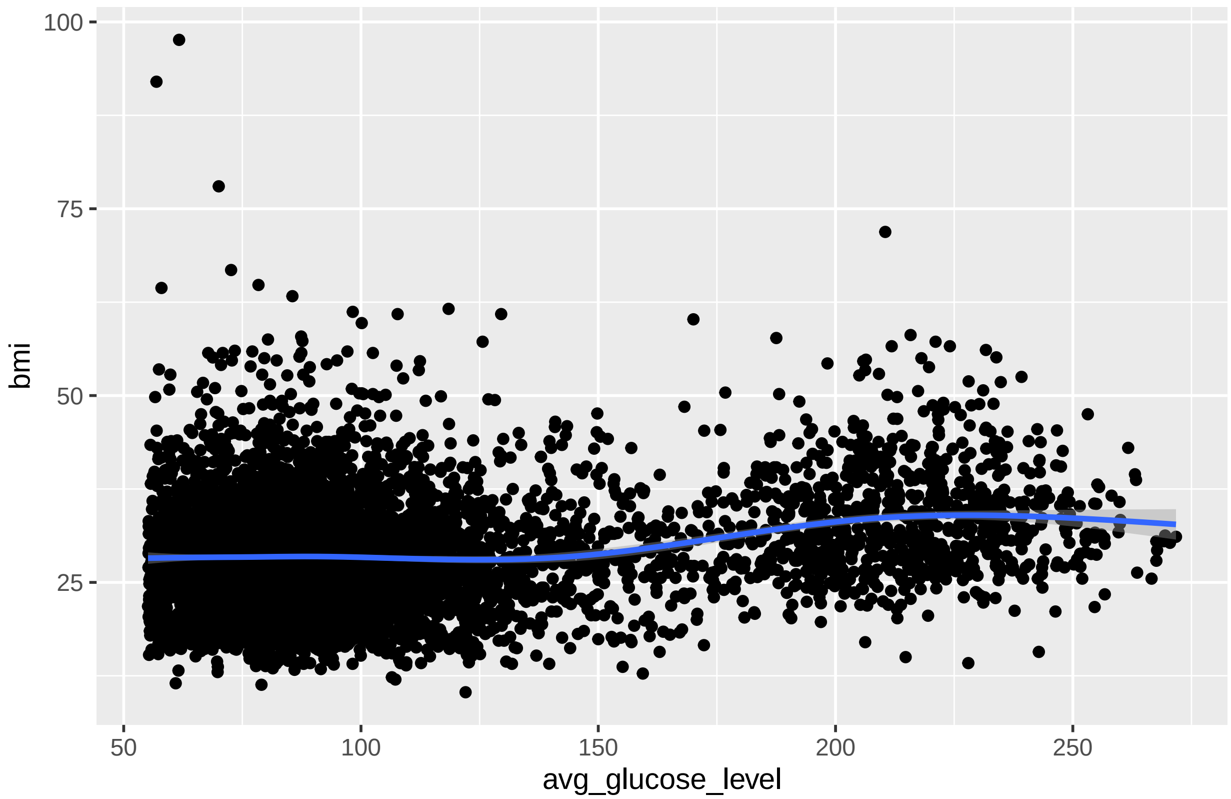


We can see from Figure Figure 2.6 that the distribution of average glucose level of patients is rightly skewed.

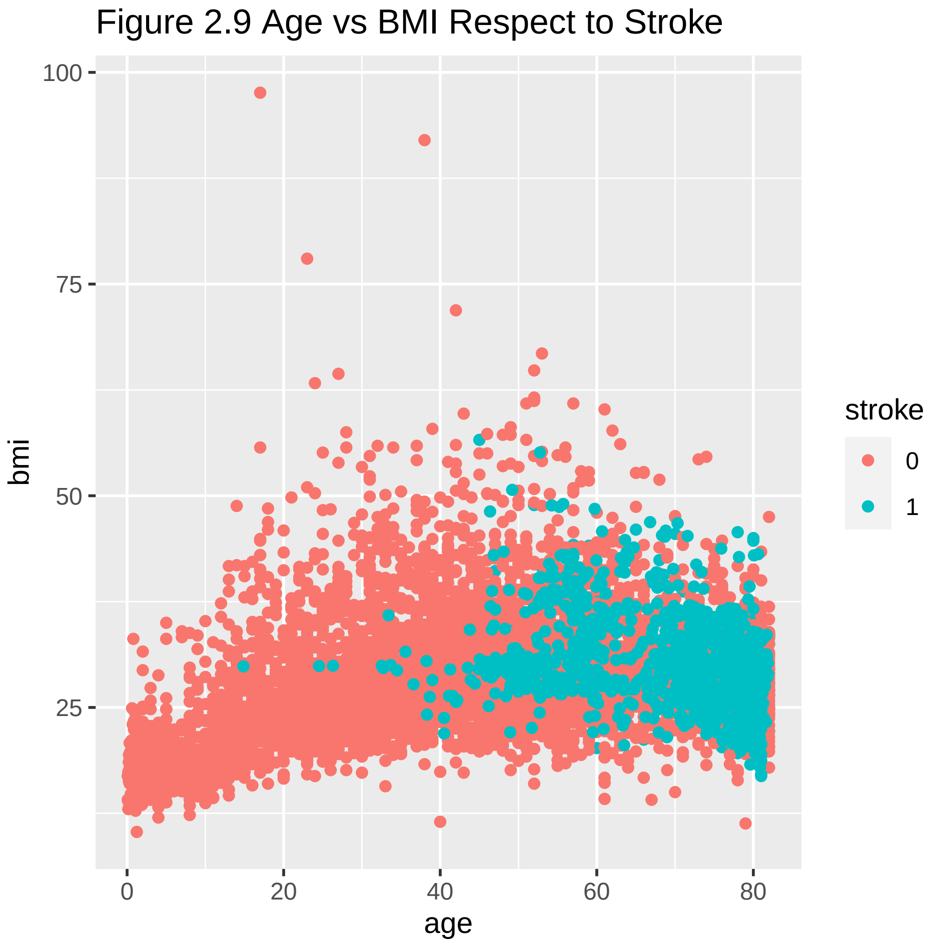


We can see from Figure 2.7 that patients' age and their BMI may have non linear correlation.

Figure 2.8 BMI and Blood Glucose Level



We can see from Figure 2.8 that patients' BMI and their average blood glucose level may have weak non linear correlation.



We can see from Figure 2.9 that there are more older adults who had stroke.

Section 3: Method 1: k Nearest Neighbors(kNN)

Section 3.1: Introduction

We begin building the classifier by implementing a technique called k-Nearest Neighbors(kNN) to predict the stroke status of the patients. It uses the stroke status of the k nearest neighboring data points to predict the stroke status of any given data point in the data set. We employed this technique to predict patients' stroke status since it does not have explicit training before making predictions, therefore allowing the data of new patients to be added without the need of training the model again (Kumar).

Sectio 3.2: Method

Figure 10 is an illustration of the kNN technique with only age and BMI as the features utilized in kNN, where 0 = the patient have not had a stroke, 1 = the patient had a stroke (For demonstration purposes, we only sampled 400 rows of data from the cleaned stroke data set).

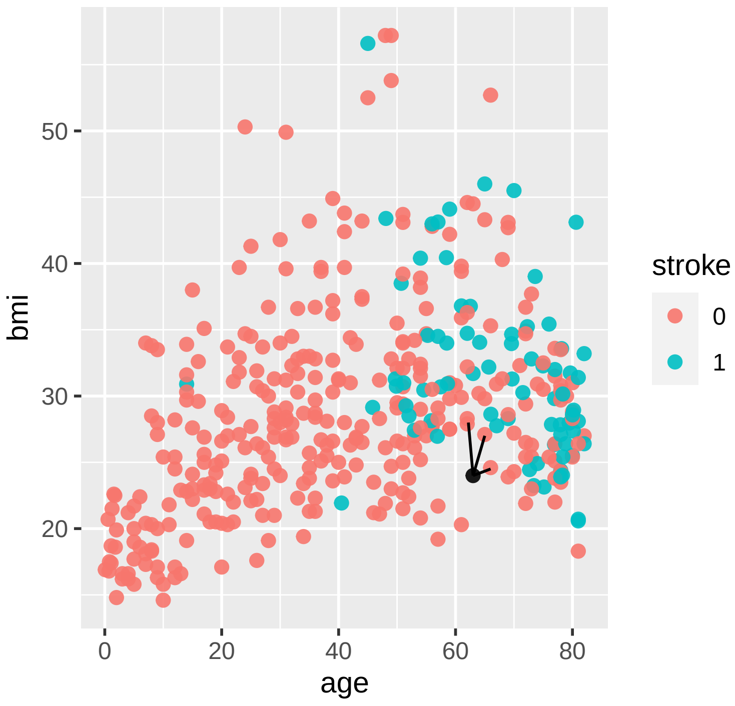
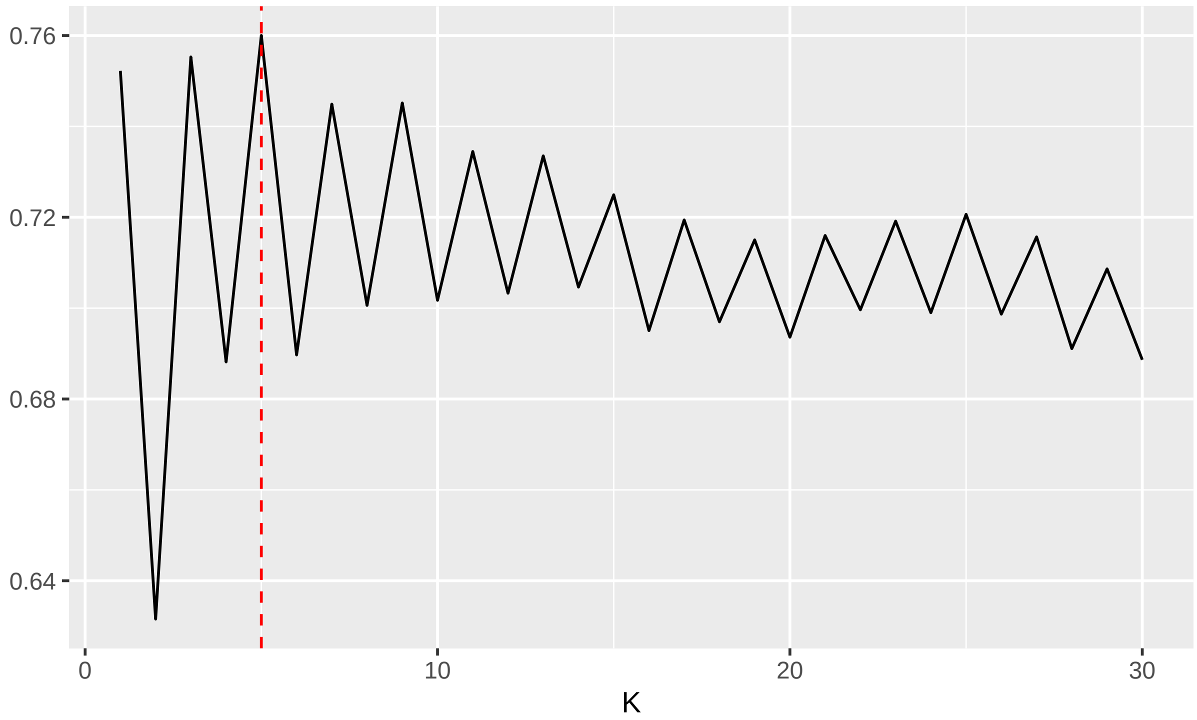


Figure 3.2.1 Illustration of kNN

In Figure 3.2.1, we have a new data point representing a patient who is 63 years old with BMI equals to 24(shown as the black dot on the graph).If we set the k value for k Nearest Neighbors as 3. The kNN algorithm will find the stroke status of the 3 nearest data point to the data point of this patient, as shown in Figure 3.1 with black line segments connecting 3 neighbor points and the black dot. The algorithm the assigns the stroke status of majority of the 3 neighbor points as the predicted stroke status of this patient. Since all 3 neighbor points has 0 as their stroke status, this patient was predicted that he/she/them have not had a stroke.

We utilized 10-fold Cross Validation to assess the predictive accuracy of kNN on predict stroke of patients in the stroke data set. To do this we randomly divided the stroke data set into 10 folds with equal size, where each fold contains $\frac{1}{10}$ of the rows in stroke data set. We then started with treating fold 1 as the new test set of stroke data and the remaining 9 folds as the training data. We repeated this process for 10 times to train the model and obtain predictions on the training data. This method was used along with the process of determine the best k value for kNN.That is, we conduct 10-fold Cross Validation for each k value, and calculate the sensitivity (true positive rate, percent of patients who had strokes that we correctly predicted on their higher stroke risk), specificity (true negative rate, percent of patients who have not had stroke that we correctly predicted) and the geometric mean of them. The geometric mean is calculated by taking the square root of sensitivity and specificity, the closer the two metrics are, the larger the geometric mean. The k value that provides the largest geometric mean is regarded as the best k value for this prediction through kNN. A line graph of geometric mean with respect to each k value was used to indicate the best k value for our prediction.

Figure 3.2.2 GeoMean Graph of kNN with Best k Shown
(10 Fold Cross Validation)



Geometric Mean, ReD Line at K= 5

Figure 3.2.2 shows that k=5 is the best k value for the 10-fold Cross Validation, as it provides the largest geometric mean in k values from 1-30, therefore providing the most balance between sensitivity (true positive rate) and specificity (true negative rate) of the prediction.

Section 3.3: Results

We assesses the predictive accuracy of kNN through the following predictive metric: sensitivity (true positive rate),specificity (true negative rate), accuracy, and classification error rate (CER). The sensitivity indicates the percentage of patients who had stroke that we correctly predicted on their higher risk of having stroke in all patients who had stroke. The specificity indicates the percentage of patients who have not had stroke that we correctly predicted on their lower risk of having stroke in all patients who have not suffered from stroke. Accuracy indicates the percentage of patients we correctly predicted on in all patients. CER indicates the percentage of patients that we failed to make correct prediction on whether they had stroke in all patients.

Below is a confusion matrix of the prediction using kNN with k value equals to 5, where rows are predicted stroke status and columns are actual stroke status of patients.

Table 5: Confusion Matrix of Predicting Stroke (10-fold Cross Validation with k=5)

	0	1
0	4343	572
1	357	603

From Table 5, we have:

Table 6: Performance Metrics kNN (k=5)

Sensitivity	Specificity	Accuracy	CER
0.514	0.928	0.844	0.156

According to the result obtained from 10-fold Cross Validation, kNN allowed us to reach a sensitivity of 0.514 and a specificity of 0.928, and an overall accuracy of 0.844 on predicting patients in the cleaned data set on whether each of them are at higher or lower risk of having stroke. Despite the accuracy of 0.83 which indicates we made accurate predictions for 83% of the patients on their risk of having a stroke, the sensitivity of our prediction revealed that we only predicted accurately for 51.4% of the patients on their higher risk of having a stroke. Since the kNN algorithm only accept numeric features,the 8 remaining categorical features were not used for the prediction. Thus, other techniques that can utilize all available features might be needed for offering better prediction on risk of having stroke.

Section 4: Method 2: Ridge Regression

Section 4.1: Introduction

The kNN algorithm we used in Section 3 was unable to utilize all available features for the prediction of risk of stroke. Additionally, Figure 2.7 and 2.8 from the EDA section indicates that non linear correlations exist among the three numeric features in the data set. Therefore, we decided to implement penalized regression techniques, along with logistic regression, to build the second classifier that not only utilizes both numeric and categorical features in making predictions, but also mitigates the impact of multicollinearity (correlations between features) in the prediction.

Section 4.2: Method

We uses logistic regression model with ridge regularization to create the prediction model for predicting risk stroke status, as it provides the effect of shrinkage. In other words, shrinking the coefficients of correlated features towards 0 while still maintaining all the features in the cleaned data set when building the model, thus restrains the impact from them and mitigates the effect caused by multicollinearity (correlated features in the data set). To do this we need to find the tuning parameter that provides the best shrinking effect.

The general form of our model is the following(where $Y_i \sim Bernoulli(\pi_i)$):

$$log(\frac{\boldsymbol{\pi_i}}{1-\boldsymbol{\pi_i}}) = \boldsymbol{X_D\beta} + \boldsymbol{\epsilon}$$

, where π_i is the probability that a patient has higher risk of having stroke. The $\boldsymbol{X_D}$ is our design matrix that consists of all the features remained in the data set after data cleaning.

The deviance of the logistic regression model is defined as a measurement of how much the fitted logistic regression model deviates from a model that perfectly predicts each of the observation. In other words, deviance refers to the goodness of fit. Therefore, the smaller the deviance is, the better the model fits on the observed response (Kjytay and Kjytay).

For using ridge regression along with logistic regression, we uses Deviance, plus the penalty term $\lambda\boldsymbol{\hat{\beta}^T\hat{\beta}}$, which constrains the regression coefficient when correlated feature exist.

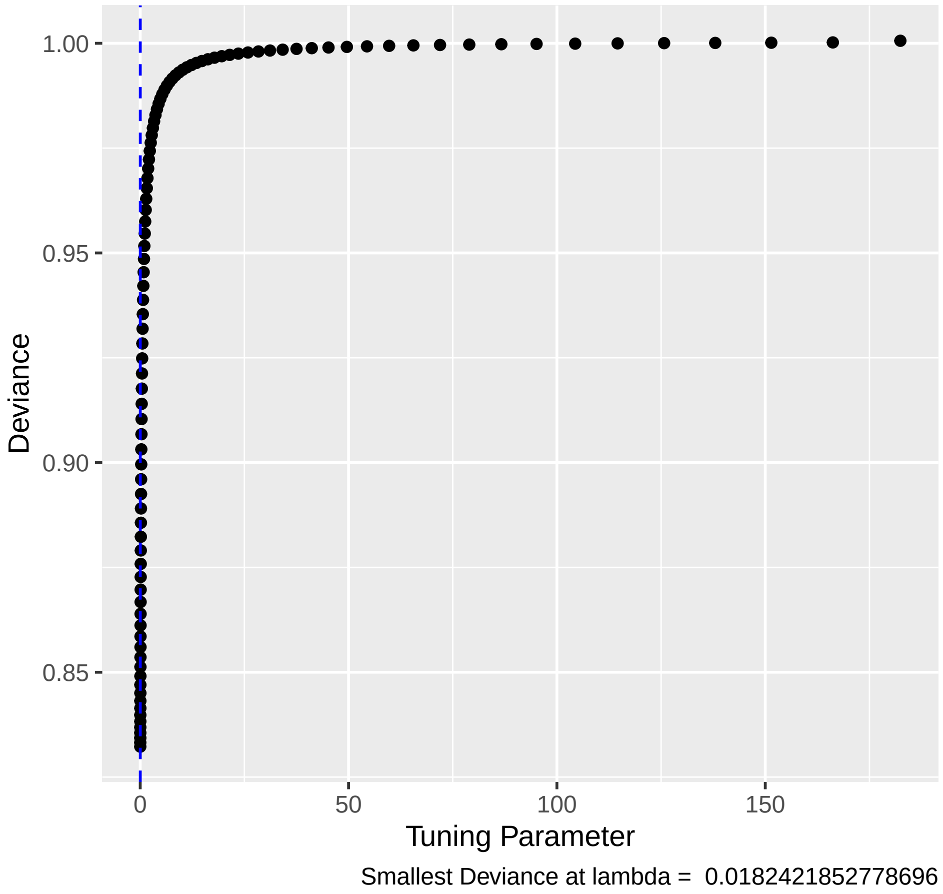
$$Deviance + \lambda\boldsymbol{\hat{\beta}^T\hat{\beta}}$$

Our goal is to find $\boldsymbol{\beta}$ coefficients that minimizes the following:

$$Deviance + \lambda\hat{\boldsymbol{\beta}}^{\mathbf{T}}\hat{\boldsymbol{\beta}} = -2log(f(Y|\hat{\boldsymbol{\beta}})) + \lambda\hat{\boldsymbol{\beta}}^{\mathbf{T}}\hat{\boldsymbol{\beta}}$$

In order to implement ridge regression ,we need to find the value of tuning parameter λ that provides the most ideal shrinkage to the model. We use the deviance to assess the predictive accuracy of the models that uses different values of λ. The logistic regression is said to be more accurate when has smaller deviance. The model with smaller deviance will perform better on predicting observed response, therefore lead to a higher predictive accuracy (Kjytay and Kjytay). To find the most ideal value of lambda, we utilize 10-fold Cross Validation along with each value of lambda from 1 to 50 (increment by 0.05 for each time) to determine the value of lambda that correspond to the lowest deviance among all the lambda values we tested.This lambda value will be used to train the logistic regression model with ridge regularization.

Figure 4.2.1 Change of Deviance vs Lambda (Ridge)



Smallest Deviance at lambda = 0.0182421852778696

Table 7: Deviance of Model with Chosen Lambda (Ridge)

	Lambda	Deviance
100	0.0182422	0.692653

We can see from Figure 4.2.1 and Table 7 that the lowest deviance (0.69) occurred when the model is using $\lambda = 0.0182421$ as the tuning parameter of model. Therefore, we select $\lambda = 0.0182421$ and fit the the final logistic regression model with ridge regularization.

Below is the confusion matrix of prediction using logistic regression with ridge regularization with default threshold,where rows are predicted stroke status and columns are actual stroke status of patients.

Table 8: Confusion Matrix for Prediction of Stroke Via Ridge (Threshold Unadjusted)

	0	1
0	4528	744
1	172	431

From Table 8, we have:

Table 9: Performance Metrics Ridge (Threshold Unadjusted)

Sensitivity	Specificity	Accuracy	CER
0.367	0.963	0.844	0.156

We can see from Table 9 that although our ridge regression model reaches an overall predictive accuracy of 0.844, we only reached a sensitivity of 0.366, meaning that among all of the patients who had stroke, we are only able to accurately predict 36.6 % them on their higher risk of suffering from stroke. This is caused by the imbalanced nature of our data set (Brownlee).

If we uses the default setting 0.5 as the threshold used on predicting whether a patient has higher risk of suffering from stroke,the sensitivity (true positive rate) of our prediction model is going to be low while the specificity is going to be high since majority of the observations we use to build the model consist of patients who have not suffered from stroke (Brownlee). To mitigate the issue and reach our goal of predicting if a patient has higher risk of suffering from stroke, we utilizes ROC curve to find the ideal threshold of the prediction (Google Developers Machine Learning Crash Course).

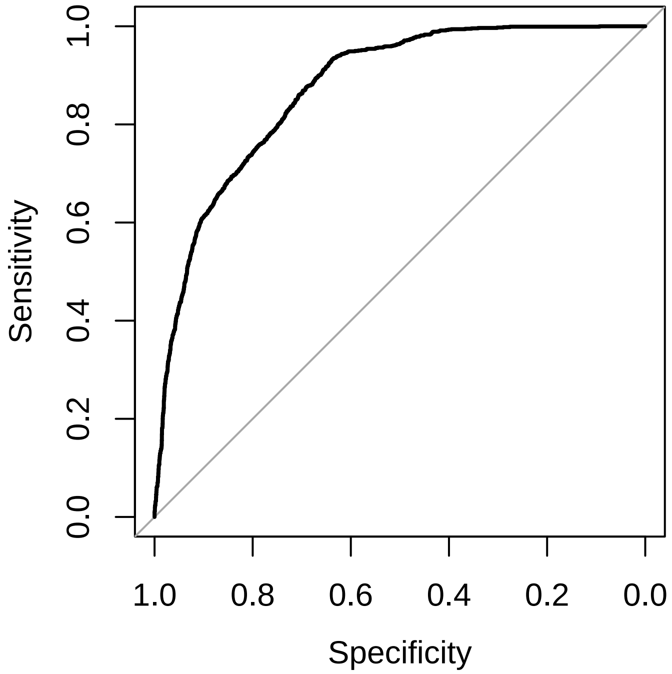


Figure 4.2.2 ROC Curve Ridge Regressio

Table 10: Threshold With Ideally Balanced Sensitivity and Specificity and Largest Geometric Mean

	Threshold	Sensivity	Spec	GMean
3482	0.1913847	0.8604255	0.7057447	0.7792565

We can see from Table 10 that threshold = 0.191465 provides us a balance between sensitivity and specificity of the prediction model, since it has the largest largest geometric mean that indicate the ideal balance between sensitivity and specificity of the prediction model.

Section 4.3: Results

Below is a confusion matrix of prediction on whether a patient in our cleaned data set has higher risk of having stroke (rows in the matrix are predicted stroke status and columns are actual stroke status of patients)

Table 11: Confusion Matrix for Prediction of Stroke Via Ridge (Threshold Adjusted)

	0	1
0	3317	164
1	1383	1011

From Table 11,we have;

Table 12: Performance Metrics Ridge Regression (Threshold Adjusted)

Sensitivity	Specificity	Accuracy	CER
0.86	0.706	0.737	0.263

We can see from Table 12 that logistic regression with ridge regularization reached a sensitivity of 0.86, meaning the model is able to correctly predict for 86% of the patients on their higher risk of having stroke in all patients who had stroke in our cleaned data set. The specificity of our model is 0.706, which indicates that the model is able correctly predict for 70.6% of patients on their lower risk of having stroke in all patients with have not had stroke. An overall accuracy of 0.737 indicates that the model correctly predicts the stroke risk status for 73.7% of the patients in our cleaned data set. A CER of 0.263 indicates that the model incorrectly predicted the stroke risk status of 26.3% of the patients.

The logistic regression with ridge regularization enabled us to predict more accurately than kNN on identifying people with higher stroke risk. However, the increased CER on prediction would lead to more incorrect prediction on people' stroke risk. Therefore, further techniques may be needed for mitigating this issue and developing better classifier on stroke risk prediction.

Section 5: Method 3: Bagged Forest

Section 5.1: Introduction

Although the ridge regression model we used in Section 4 reached a higher predictive accuracy on predicting patients with higher risk on having stroke, the classification error rate of the prediction model was 0.263, meaning the risk status of 26.3% of the patients from the cleaned data set was incorrectly predicted. Thus, we introduce the technique called Bagged Forest to build the third classifier and attempt to mitigate this situation. As it may provide higher accuracy on the prediction on stroke risk status. To do this we need to discuss about the classification tree in the next section, which are the components of the bagged forest.

Section 5.2: Method

Below is a sample decision tree created based on features including gender, age, hypertension, heart disease status, marriage status and stroke (response variable).

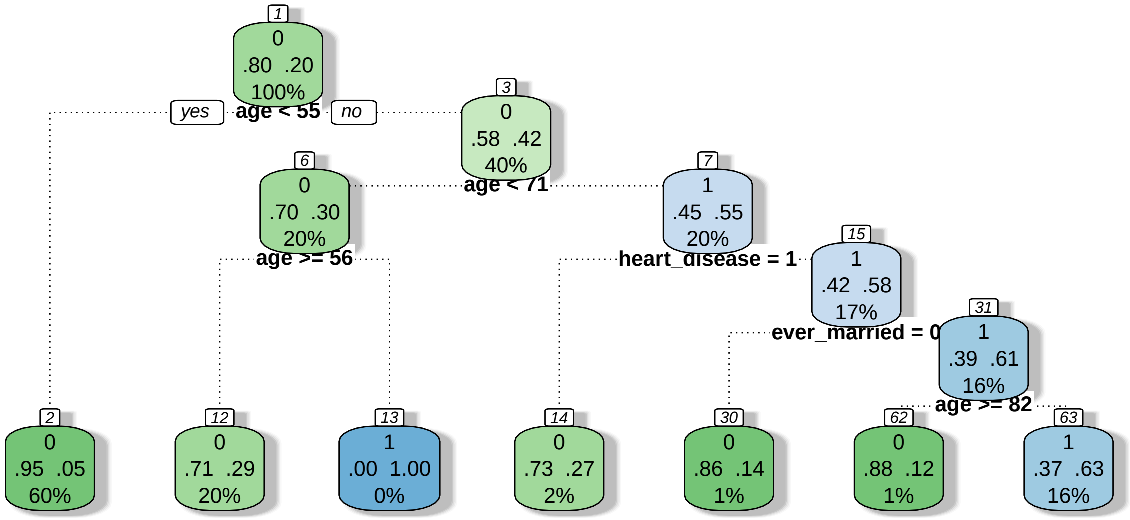


Figure 5.2.1 Sample Classification Tree (Only Use 6 Features)

At the beginning of this tree, we put the patients of the entire cleaned data set as the “root” of the tree, then we split the root of the tree into two nodes based on the features we used, which distribute the patients in the cleaned data set into the two nodes based on their class on this specific feature. The split was decided based on the way that gives us the smallest Gini Index (A measure of a split’s effectiveness of correctly classifying a patient into a node in the tree). This process was repeated until we get the lowest Gini Index, which indicates the most ideal classifying ability of a classification tree (Dash).

However, a single decision tree is prone to overfitting, which results in a high variance on the classifier. In other words, the tree model is trained to tightly to the data set we use to create it, and made it unusable to predict for the stroke status of new patients in screening (Liberman).

The technique of bagged forest consists of a series of decision trees (in this project, classification trees). The bagged forest utilizes the bagging (abbreviation of Bootstrap Aggregation) process, which randomly draws out rows of data from our cleaned data set, and creates classification trees with using the data from each bootstrapping process. All the features in the data set will be used to create each classification tree. When using the bagged forest to predict the risk of stroke of a patient, each tree in the forest will make a prediction based on the data, and the majority of the prediction result will be assigned to the patient as the predicted stroke risk status. This “voting” process helps to mitigates the issue of overfitting and decrease the variance.

```
## pred_bag
##    0    1
## 4806 1069
```

Section 5.3 Results

Below is the confusion matrix of prediction on the set of data that is not been used for growing each tree in the forest, respectively. This is named as the "Out of Bag" data, abbreviated as "OOB" data. The rows in the matrix are predicted stroke status and columns are actual stroke status of patients.

Table 13: Confusion Matrix of Bagged Forest

	0	1
0	4510	296
1	190	879

From Table 13, we have:

Table 14: Performance Metrics Bagged Forest

Sensitivity	Specificity	Accuracy	CER
0.748	0.96	0.917	0.083

We can see from Table 14 that the bagged forest allowed us to reach a sensitivity of 0.748, meaning in all patients who had stroke in the OOB data of this bagged forest model, the model can correctly predict for 74.8% of the patients on their higher risk of having stroke. The specificity of the prediction is 0.96, which indicated that the model can correctly predict for 96% of patients on their lower stroke risk in OOB data of this bagged forest model. An overall accuracy of 0.917 indicates that the model is able to correctly predict the stroke risk status of 96% of patients in the OOB data of this bagged forest. A CER of 0.083 means the model incorrectly predicts the stroke risk status of 8.3% of the patient in the OOB data of this bagged forest.

Conclusion

We have conducted kNN, logistic regression with ridge regularization method, and bagged forest to build the model for predicting stroke risk.

Table 15: Compare 3 Techniques on Performance Metrics

techniques	Sensitivity	Specificity	Accuracy	CER
kNN with k=5	0.514	0.928	0.844	0.156
Logistic Regression With Ridge Regularization	0.860	0.706	0.737	0.263
Bagged Forest	0.748	0.960	0.917	0.083

We can see from Table 14 that logistic regression with ridge regularization has the highest sensitivity (0.86) with relatively low specificity (0.706), while kNN has the lowest sensitivity (0.514). Bagged Forest outperformed logistic regression and kNN on the overall accuracy and CER of the prediction (0.917 and 0.083, respectively). Despite the highest sensitivity that logistic regression with ridge regularization has, we would recommend using the bagged forest model as the classifier for identifying people with a higher risk of having a stroke during screening, since it has the lowest classification error rate and highest specificity while maintaining a relatively high ability to identify people with a higher risk of having stroke. Additionally, the bagged forest will be able to perform better in the real-life setting when predicting new data from people. The limitation we found is that models constructed through all 3 techniques are impacted by the imbalanced nature of our data set, we recommend collecting more data to resolve this issue.

Works Cited

“About Stroke.” Centers for Disease Control and Prevention, Centers for Disease Control and Prevention, 2 Nov. 2022, https://www.cdc.gov/stroke/about.htm.

“Stroke Facts.” Centers for Disease Control and Prevention, Centers for Disease Control and Prevention, 14 Oct. 2022, https://www.cdc.gov/stroke/facts.htm.

Fedesoriano. “Stroke Prediction Dataset.” Kaggle, 26 Jan. 2021, https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset/code.

Imbalanced Learn. “2. Over-Sampling#.” 2. Over-Sampling - Version 0.10.0.dev0, https://imbalanced-learn.org/dev/over_sampling.html#smote-variants.

Wijaya, Cornellius Yudha. “5 Smote Techniques for Oversampling Your Imbalance Data.” Medium, Towards Data Science, 24 May 2022, https://towardsdatascience.com/5-smote-techniques-for-oversampling-your-imbalance-data-b8155bdbe2b5.

RolandRoland.”Convert All Data Frame Character Columns to Factors.” Stack Overflow, 17 Dec. 2013, https://stackoverflow.com/questions/20637360/convert-all-data-frame-character-columns-to-factors.

Wu, Dongyuan. “Dongyuanwu/RSBID: Resampling Strategies for Binary Imbalanced Datasets Version 0.0.2.0000 from Github.” Version 0.0.2.0000 from GitHub, 18 July 2022, https://rdrr.io/github/dongyuanwu/RSBID/.

Singh, Deepika. “Deepika Singh.” Pluralsight, 12 Nov. 2019, https://www.pluralsight.com/guides/finding-relationships-data-with-r.

Reka Solymosi (maintained and updated by David Buil-Gil and Nicolas Trajtenberg). “Making Sense of Crim Data.” Chapter 3 Week 3, 8 Nov. 2022, https://maczokni.github.io/MSCD_labs/week3.html.

Kumar, Naresh. “Advantages and Disadvantages of kNN Algorithm in Machine Learning.” Advantages and Disadvantages of kNN Algorithm in Machine Learning, http://theprofessionalspoint.blogspot.com/2019/02/advantages-and-disadvantages-of-kNN.html.

Maximum Likelihood Estimation, https://sta279-s22.github.io/slides/lecture_6.html.

Kjytay, and Kjytay. “What Is Deviance?” Statistical Odds & Ends, 27 Mar. 2019, https://statisticaloddsandends.wordpress.com/2019/03/27/what-is-deviance/.

Brownlee, Jason. “A Gentle Introduction to Threshold-Moving for Imbalanced Classification.” MachineLearningMastery.com, 4 Jan. 2021, https://machinelearningmastery.com/threshold-moving-for-imbalanced-classification/.

“Classification: Roc Curve and AUC | Machine Learning | Google Developers.” Google, Google,https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc#:~:text=An%20ROC%20curve%20(receive

“Decision Tree Introduction with Example.” GeeksforGeeks, 10 Nov. 2022, https://www.geeksforgeeks.org/decision-tree-introduction-example/.

Liberman, Neil. “Decision Trees and Random Forests.” Medium, Towards Data Science, 21 May 2020, https://medium.com/towards-data-science/decision-trees-and-random-forests-df0c3123f991.

Dash, Shailey. “Decision Trees Explained-Entropy, Information Gain, Gini Index, CCP Pruning..” Medium, Towards Data Science, 2 Nov. 2022, https://towardsdatascience.com/decision-trees-explained-entropy-information-gain-gini-index-ccp-pruning-4d78070db36c.