SC324 Final Project: Predicting Strokes

Qinbo Liu, Julian Zhu, Matt Welch, & Dhruv Joshi

May 19, 2021

1. Introduction

Worldwide, stroke is the second leading cause of death and the third leading cause of disability [1]. In the United States, stroke accounts for 150,000 deaths every year, or 1 in every 19 deaths from all causes. A model that yields an accurate prediction of stroke would thus be a valuable tool to prevent strokes and save lives.

This project leverages statistical learning techniques to make predictions on strokes, using a dataset from Kaggle [2], the author is unknown. The dataset contains 5110 observations with 10 predictors. The Framingham study has identified several important factors contributing to strokes, such as age, the use of antihypertensive therapy, cigarette smoking, and prior cardiovascular disease [3]. The dataset used in this project includes predictors directly related to these factors, such as age, an indicator on hypertension, an indicator on heart disease, average glucose levels, gender, and BMI (body mass index) of the observations. These predictors give a rough estimate of the observation's body status. The dataset also includes indicators on whether the observation has married, a dummy variable describing whether they work in the private sector, government, or are self-employed, and their residence (rural vs urban). These predictors give the social profile of the observations. Finally, the dataset contains a variable stroke that indicates whether a stroke has happened to the observation.

The goal of this study is, based on the 10 variables, to predict whether stroke has occurred for a particular input. Such a model can, for example, be used to predict

if a stroke would happen for an individual based on the individual's personal information and serve as a potential warning of risks of stroke.

2. Exploratory Descriptives

Among 5110 observations in the datasets, 4909 observations contain valid values for both the potential explanatory variables and the response variable. The remaining 201 observations contain a not-applicable BMI. To ensure that we have complete data for each sample that the model will use, the project uses only the first 4909 observations.

Some categorical variables have an extremely imbalanced number of samples for each category. 4458 samples do not have hypertension in comparison to 451 who do. 4666 samples do not have heart disease in comparison to 243 who do. 4700 out of 4909 samples did not have a stroke. These imbalances may restrict us from fully actualizing the predictive power of each of these variables. The distribution of BMI values and average glucose values are also clearly skewed to the right. We should probably use bootstrapping to better mimic the actual distribution.

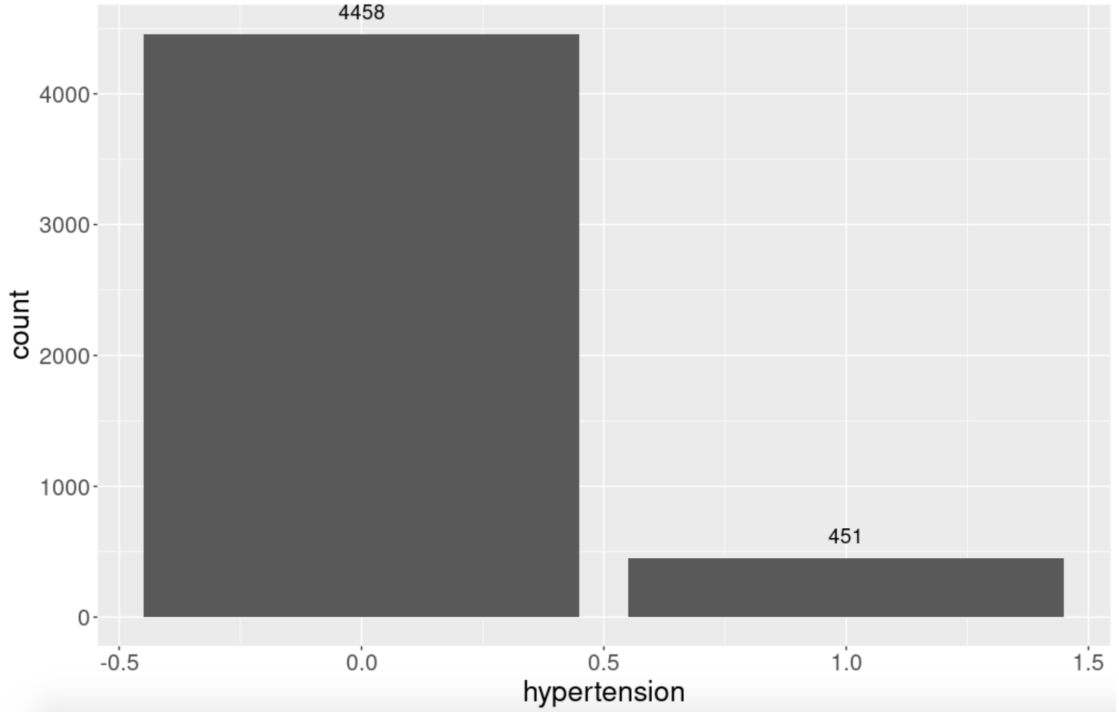

Figure 1: Bar chart of the indicator of hypertension; 1 represents “yes”, 0 represents “no”.

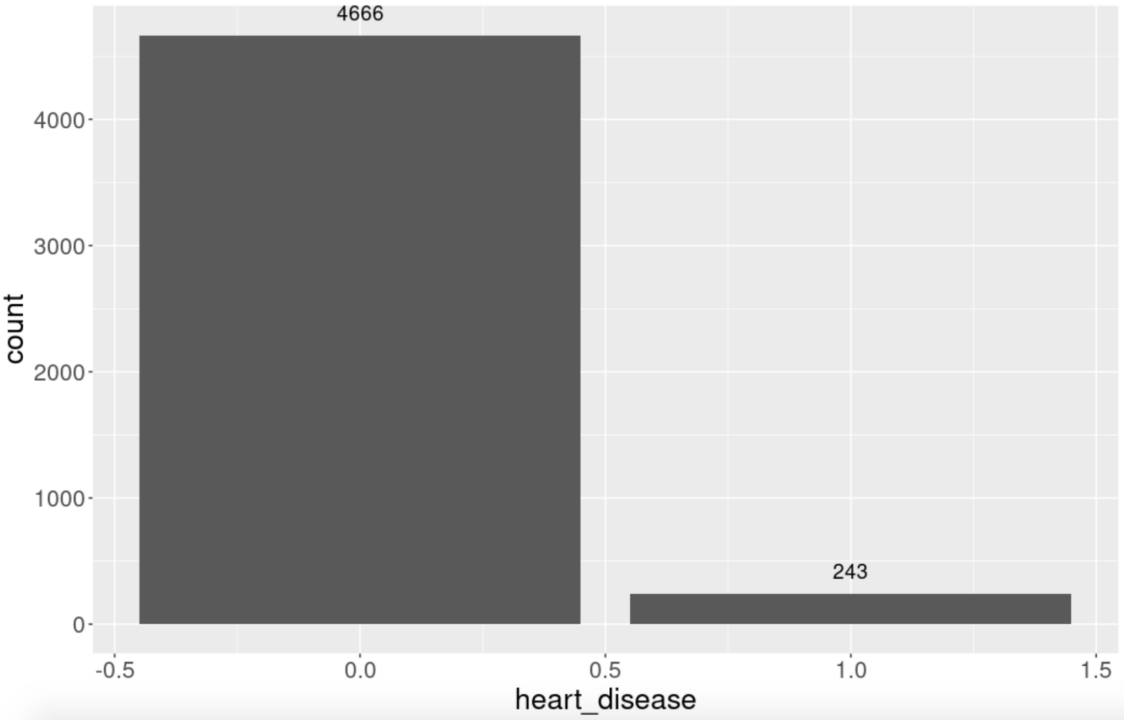

Figure 2: Bar chart of the indicator of heart disease; 1 represents “yes”, 0 represents “no”.

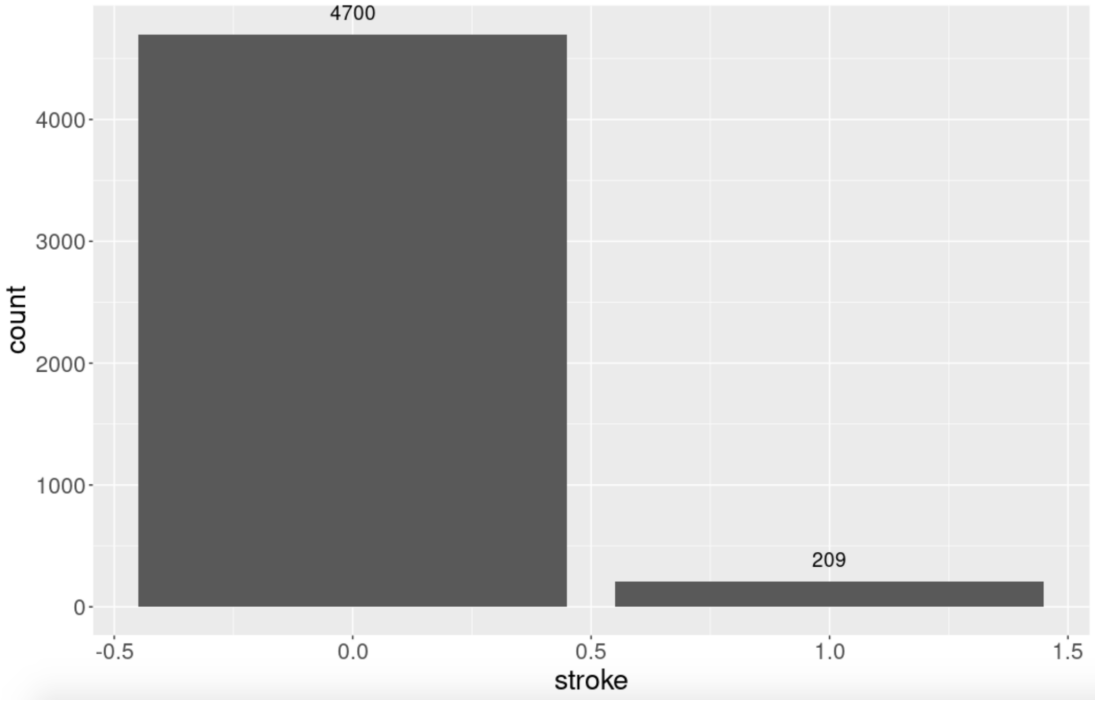

Figure 3: Bar chart of the indicator of stroke; 1 represents “yes”, 0 represents “no”.

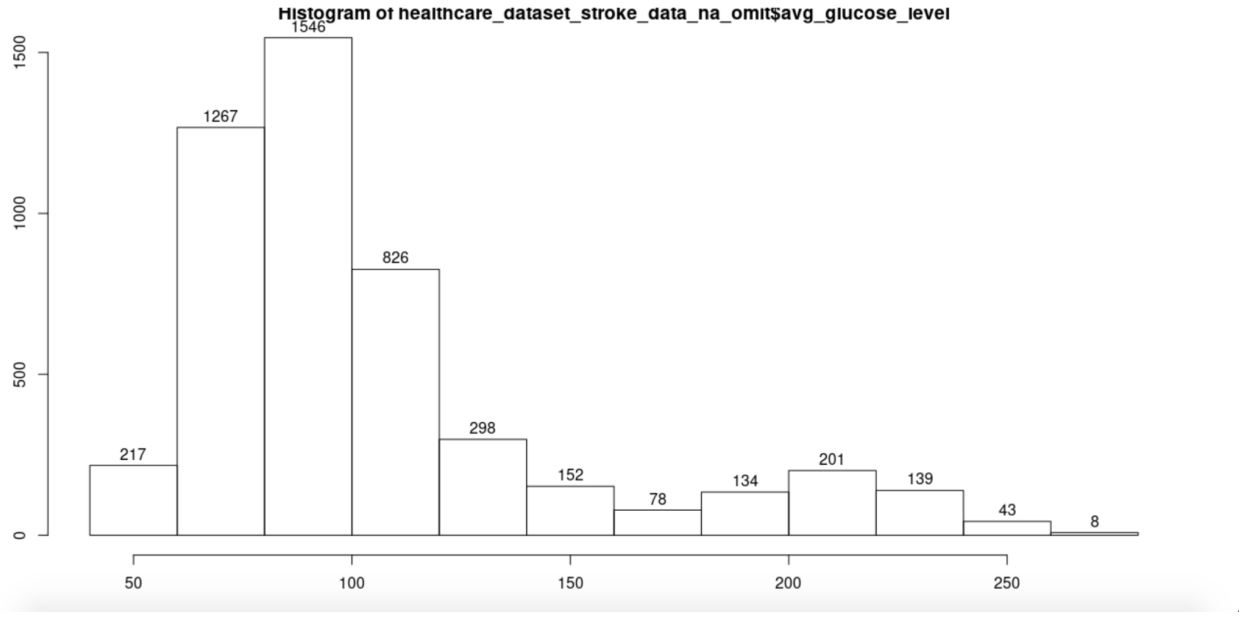

Figure 4: Histogram of average glucose level

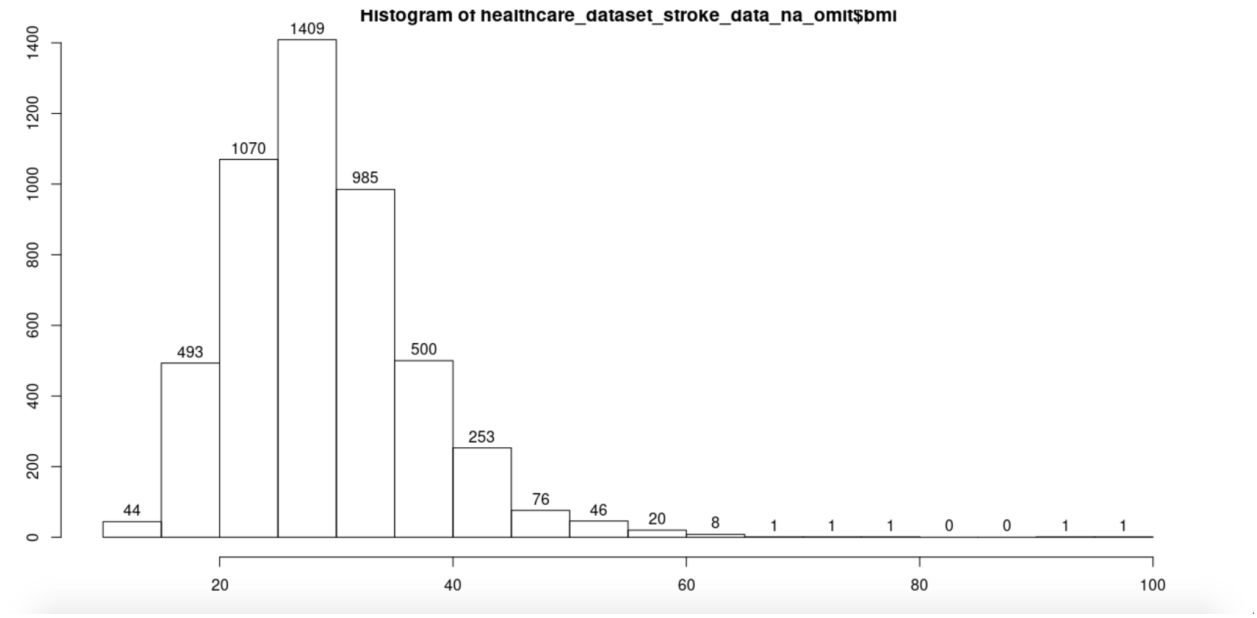

Figure 5: Histogram of BMI (Body Mass Index)

Among the 10 potential predictors, age might be the most useful predictor with a correlation coefficient of 0.23 with the occurrence of stroke. Predictors such as hypertension, heart disease, and average glucose level are also useful as they have the same correlation coefficient of 0.14 with the response. Marriage history also has a correlation coefficient of 0.11 with the response. Smoking status is the only predictor with a negative correlation with the response, but the impact might be small given that the value of the correlation coefficient is -0.08. Some predictors also have a strong correlation with each other. Age and marriage history have a correlation coefficient of 0.68. The correlation between age and work type and that between marriage history and work type are also strong, given the corresponding correlation coefficients are 0.54 and 0.43 respectively. The correlation matrix notices some relatively strong negative correlations between smoking status and predictors such as age, marriage history, work type, and bmi. Whether a sample has hypertension, whether the sample has heart disease, and the average glucose level also correlates with age with correlation

coefficients 0.27, 0.26, and 0.24 respectively. The model created in this project should try to minimize the impact of these correlations and tease out the independent effect of each predictor on the response variable.

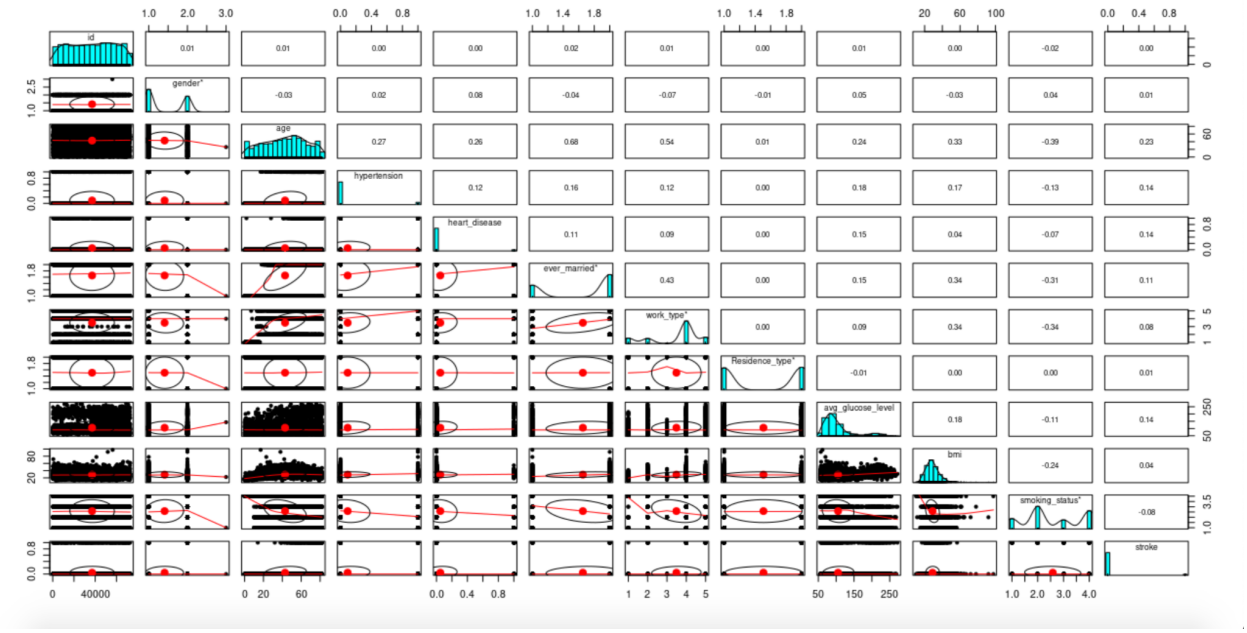

Figure 6: Correlation matrix of all variables.

3. Analysis

See the Appendix for relevant code.

Data preparation

We began by removing 201 observations in the data that contain N/A values (only the BMI variable has), leaving us with 4909 observations. Then, we converted the following indicator and categorical variables into factors: gender, hypertension, heart_disease, ever_married, work_type, Residence_type, smoking_status, and stroke. Additionally, we converted the BMI variable into a numeric type: its N/A values are stored as strings instead of numbers, so the entire column is initially parsed as a string type.

Next, we observed that the positive stroke observations were quite sparse: 209 observations were labeled as having a stroke, and the remaining 4700 observations were labeled as not having a stroke. To alleviate our models' biases against classifying positive stroke values, we aimed to balance our training set. To that end, we began by selecting a random sample without replacement of 209 negative stroke cases. Then, our training set contains a random sample of 250 observations (60% of 418) selected without replacement from the 209 positive stroke cases and the 209 previously selected negative stroke cases. Thus, our training set contains 250 observations randomly selected from a balanced sample of positive and negative stroke cases, and our test set contains the remaining 4659 observations, most of which are negative stroke cases.

As a result of this procedure, our models were trained on a balanced data set of positive and negative stroke cases and tested on data in which the positive stroke cases were sparse. While this leads to more false positives, it ensures our models do not simply predict every case is a negative stroke to maximize accuracy.

Multiple Logistic Regression

We first fit a multiple logistic regression model using all of the variables in the training set to determine which variables were significant predictors of stroke. From that, we determined that the variables age, hypertension, ever_married, and work_type were the most significant, so we fit a new multiple logistic regression model only using those variables as predictors of stroke. To predict using the test set, we selected a threshold of 0.5 on predicted probabilities from the logistic regression model, so that a predicted value larger than 0.5 indicates a positive stroke case. On the testing data, this model resulted in a 30.0% test error rate, with the following confusion matrix:

Truth --- Prediction	No Stroke	Stroke

No Stroke	3,191	15
Stroke	1,384	69

Classification Tree

We first fit a classification tree using all of the variables in the training set to predict stroke. Then, using cross-validation, we automatically found the optimal number of terminal nodes for the training data to construct a pruned tree to predict stroke. On the testing data, this model resulted in a 26.1% test error rate, with the following confusion matrix:

Truth --- Prediction	No Stoke	Stroke
No Stroke	3,387	28
Stroke	1,188	56

Linear SVM

We fit multiple linear SVM models using all of the variables in the training set to predict stroke, with a different cost parameter for each model. In the end, we determined that a cost parameter of 1 produced the optimal training set results. On the testing data, this model resulted in a 32.4% test error rate, with the following confusion matrix:

Truth --- Prediction	No Stroke	Stroke

No Stroke	3,077	11
Stroke	1,498	73

4. Conclusion

Strokes have a significant impact worldwide. The effect of stroke is significant in South Asia due to genetic predisposition and developed countries like the US despite their medical strength. Thus mitigating stroke risk is an important task to improve the lives of many people across the world. Considering stroke mitigation can be active steps/medication or more long-term routes such as correction of diet and lifestyle, people at high risk should start consulting healthcare professionals.

Given the biased dataset and only 250 training observations, we see significant results from our models. Accuracy is not a useful metric for this problem as the dataset is quite biased. Instead, we consider the specificity and sensitivity of the models to see how they perform. We see that the class tree model has the highest specificity at 74% and the linear SVM model has the highest sensitivity at 87%. However, they both sacrifice the other statistic to achieve these results. On the other hand, multiple logistic regression performs moderately on both statistics with 70% specificity and 82% sensitivity.

Model	Specificity	Sensitivity
Multiple Logistic Regression	69.7%	82.1%
Classification Tree	74.0%	66.7%
Linear SVM	86.9%	67.3%

With this important problem, it is important to reconsider our objective, which is to create a model that lets people know when they should begin seeking medical advice. Thus, considering our models have a high sensitivity (linear SVM has a sensitivity of 87% and a specificity of 67%), we see that this helps rule out people. That is if the prediction is negative, the person being considered does not need to seek medical attention promptly. This allows people to enter basic information such as age, smoking status, marital status, BMI, etc, and receive a prompt prediction that if negative suggests they do not need to seek medical advice promptly.

Our models perform well considering the small size of the training data. To improve the results, however, we considered methods that would allow us to make inferences through the significant amount of data we could not use as a result of the biased dataset. The methods considered were oversampling and SMOTE to create more data points for the positive case. Another method would include using more flexible models to better fit the train data and thus predict positives on the test data despite the biased dataset. These alternatives would allow us to use many more train examples to perform better on the test set.

References

[1] https://www.who.int/bulletin/volumes/94/9/16-181636/en/

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

[3] P. A. Wolf, R. B. D'Agostino, A. J. Belanger, and W. B. Kannel. Probability of stroke: a risk profile from the Framingham study. Stroke, 22:312–318, March 1991.

Appendix

Exploration code:

```
library(ISLR)
library(psych)
library(ggplot2)

#correlation before omitting NA
pairs.panels(healthcare_dataset_stroke_data)
bmi.omit = as.numeric(healthcare_dataset_stroke_data$bmi)
healthcare_dataset_stroke_data$bmi <- c(bmi.omit)
summary(healthcare_dataset_stroke_data)

#Create a data frame that omit NA
healthcare_dataset_stroke_data_na_omit = na.omit(healthcare_dataset_stroke_data)
healthcare_dataset_stroke_data_na_omit <-
subset(healthcare_dataset_stroke_data_na_omit, select = -bmi.omit)
summary(healthcare_dataset_stroke_data_na_omit)

#Correlation after omitting NA
pairs.panels(healthcare_dataset_stroke_data_na_omit)

#Variable histograms and distributions
#bar plot: gender
```

```
gender = ggplot(data=healthcare_dataset_stroke_data_na_omit, aes(x=gender)) +
  geom_bar() +
  geom_text(stat='count', aes(label=..count..), vjust=-1, size = 5)
gender + theme(text = element_text(size = 20))

#histogram: age
par(mfrow=c(1,1))
hist(healthcare_dataset_stroke_data_na_omit$age, labels=TRUE)

#bar plot: hypertension
hypertension = ggplot(data=healthcare_dataset_stroke_data_na_omit,
aes(x=hypertension)) +
  geom_bar() +
  geom_text(stat='count', aes(label=..count..), vjust=-1, size = 5)
hypertension + theme(text = element_text(size = 20))

#bar plot: heart_disease
heart_disease = ggplot(data=healthcare_dataset_stroke_data_na_omit,
aes(x=heart_disease)) +
  geom_bar() +
  geom_text(stat='count', aes(label=..count..), vjust=-1, size = 5)
heart_disease + theme(text = element_text(size = 20))

#bar plot: ever_married
ever_married= ggplot(data=healthcare_dataset_stroke_data_na_omit,
aes(x=ever_married)) +
  geom_bar() +
  geom_text(stat='count', aes(label=..count..), vjust=-1, size = 5)
ever_married + theme(text = element_text(size = 20))

#bar plot: work_type
work_type=ggplot(data=healthcare_dataset_stroke_data_na_omit, aes(x=work_type)) +
  geom_bar() +
  geom_text(stat='count', aes(label=..count..), vjust=-1, size = 5)
work_type+ theme(text = element_text(size = 20))
```

```
#bar plot: Residence_type
Residence_type=ggplot(data=healthcare_dataset_stroke_data_na_omit,
aes(x=Residence_type)) +
  geom_bar() +
  geom_text(stat='count', aes(label=..count..), vjust=-1, size = 5)
Residence_type+ theme(text = element_text(size = 20))

#histogram: avg_glucose_level
hist(healthcare_dataset_stroke_data_na_omit$avg_glucose_level, labels=TRUE)

#histogram: bmi
hist(healthcare_dataset_stroke_data_na_omit$bmi,labels=TRUE)

#bar plot: smoking_status
smoking_status=ggplot(data=healthcare_dataset_stroke_data_na_omit,
aes(x=smoking_status)) +
  geom_bar() +
  geom_text(stat='count', aes(label=..count..), vjust=-1, size = 5)
smoking_status+ theme(text = element_text(size = 20))

#bar plot: stroke
stroke=ggplot(data=healthcare_dataset_stroke_data_na_omit, aes(x=stroke)) +
  geom_bar() +
  geom_text(stat='count', aes(label=..count..), vjust=-1, size = 5)
stroke + theme(text = element_text(size = 20))
```

Analysis code:

```
library(e1071)
library(tree)

analysis = function() {
    set.seed(1)
```

```r
#=============================================================================
# load data, remove NA data
stroke = read.csv('healthcare-dataset-stroke-data.csv', header=T)
stroke = stroke[stroke$bmi != 'N/A',]

# convert col types
factor_cols = c('gender', 'hypertension', 'heart_disease', 'ever_married',
                'work_type', 'Residence_type', 'smoking_status', 'stroke')
stroke[,factor_cols] = lapply(stroke[,factor_cols], as.factor)

stroke$bmi = as.numeric(stroke$bmi) # bmi initially parsed as strings

# many more no stroke observations;
# randomly subset no-stroke observations to match yes-stroke count
stroke_index = 1:dim(stroke)[1]
no_stroke_index = stroke_index[stroke$stroke != '1']
yes_stroke_index = stroke_index[stroke$stroke == '1']

# keep all yes-stroke observations and subset of no-strokes
balance_index = c(yes_stroke_index,
                  sample(no_stroke_index, length(yes_stroke_index)))

# randomize kept values
balance_index = sample(balance_index)

# do train/test split based on the balanced data, then include
# remaining no-stroke data in the test set
rows = dim(stroke)[1]
train_rows = as.integer(length(balance_index) * 0.6)
train_vec = sample(balance_index, train_rows)
stroke_train = stroke[train_vec,]
stroke_test = stroke[-train_vec,]

#=============================================================================
# fit logistic regression model with subset of parameters
```

```
glm_fit = glm(stroke ~ age + hypertension + ever_married + work_type,
              data=stroke_train, family=binomial)

# construct predicted categories for testing data
glm_probs = predict(glm_fit, stroke_test, type='response')
glm_preds = rep('0', rows - train_rows)
glm_preds[glm_probs > 0.5] = '1'

# evaluate the testing data results
print('============================')
print(table(glm_preds, stroke_test$stroke))
print(mean(glm_preds == stroke_test$stroke)) # correct rate
print(mean(glm_preds != stroke_test$stroke)) # error rate

#=============================================================================
# fit initial classification tree
tree_fit = tree(stroke ~ ., stroke_train, model=T)

# find and construct optimal tree with cross validation;
# pruned tree has optimal terminal nodes, matching minimum cv_tree$dev
cv_tree = cv.tree(tree_fit, FUN=prune.misclass)
terminal_nodes = cv_tree$size[which.min(cv_tree$dev)]
prune_tree = prune.misclass(tree_fit, best=terminal_nodes)

# evaluate the testing data results
prune_tree_preds = predict(prune_tree, stroke_test, type='class')
print('============================')
print(table(prune_tree_preds, stroke_test$stroke))
print(mean(prune_tree_preds == stroke_test$stroke)) # correct rate
print(mean(prune_tree_preds != stroke_test$stroke)) # error rate

#=============================================================================
# fit linear SVM
svm_fit = svm(stroke ~ ., data=stroke_train,
              scale=F, kernel='linear', cost=1)
```

```
    # evaluate the testing data results
    svm_preds = predict(svm_fit, stroke_test)
    print('============================')
    print(table(svm_preds, stroke_test$stroke))
    print(mean(svm_preds == stroke_test$stroke)) # correct rate
    print(mean(svm_preds != stroke_test$stroke)) # error rate
}

analysis()
```