EXST 7141 Midterm Project: Stroke Prediction

Dina Dinh, Jalen Amos, and Marcellus Lewis

2023-10-22

Contents

Packages Needed

Here are the packages used in this analysis

```
library(easypackages)
libraries("tidyverse","boot","randomForest","psych","AUC","MASS","car","viridis","caret","ggplot2")
```

Data Description

In this document, we will describe the results of the analysis for the Stroke Prediction dataset from the website [link]https://www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset/data. Kaggle is an online platform and community that focuses on data science, machine learning, and artificial intelligence by providing access to data, tools, and a platform for collaboration. The Stroke Prediction Dataset is used for research and analysis to understand the factors contributing to strokes and to build predictive models for identifying individuals at risk of experiencing a stroke. It includes qualitative variables like a person's marital status, and quantitative variables like body mass index (BMI), for each individual (1). The data consists of 5,110 observations with 12 quantitative and qualitative variables. The 12 variables are the following:

1. ID: An anonymous identifier for each individual.
2. Gender: The gender of the individual (e.g., Male, Female, Other).
3. Age: The age of the individual.
4. Hypertension: Indicates whether the individual has hypertension.
5. Heart Disease: Indicates whether the individual has heart disease.
6. Marital Status: The marital status of the individual (e.g., Married or Single).
7. Work Type: The type of work the individual is engaged in.
8. Residence Type: Whether the individual lives in an urban or rural area.
9. Average Glucose Level: The average glucose level of the individual.

10. BMI (Body Mass Index): The Body Mass Index of the individual.
11. Smoking Status: The smoking status of the individual (e.g., Smoker, Non-smoker, etc.).
12. Stroke: The binary response variable, indicating whether the individual had a stroke (0 = No, 1 = Yes).

Importing and Cleaning the Data

ID has been removed for analysis purposes. Most of the qualitative variables were re-coded to be binary. Given the large sample size of the data, missing values were removed. We also changed the categorical data to be factors and not numeric since most are binary. None of the categorical variables have high levels for further evaluation. The observation for gender = Other, was removed for this data.

```
data <- read.csv("../Data/Raw/healthcare-dataset-stroke-data.csv")

data <- data %>%
              drop_na()  %>%
    #select(!(id)) %>%
    mutate(gender = as.factor(gender)) %>%
    mutate(Residence_type = as.factor(Residence_type)) %>%
    mutate(ever_married = as.factor(ever_married))%>%
    mutate(work_type = as.factor(work_type)) %>%
    mutate(smoking_status = as.factor(smoking_status))%>%
    mutate(heart_disease = as.factor(heart_disease))%>%
    mutate(hypertension = as.factor(hypertension))%>%
    mutate(stroke = as.factor(stroke))


data<- data[data$gender != "Other", ]

data$gender <- droplevels(data$gender)
data <- data[, -1]
str(data)
```

```
## 'data.frame':    4908 obs. of  11 variables:
##  $ gender           : Factor w/ 2 levels "Female","Male": 2 2 1 1 2 2 1 1 1 1 ...
##  $ age              : num  67 80 49 79 81 74 69 78 81 61 ...
##  $ hypertension     : Factor w/ 2 levels "0","1": 1 1 1 2 1 2 1 1 2 1 ...
##  $ heart_disease    : Factor w/ 2 levels "0","1": 2 2 1 1 1 2 1 1 1 2 ...
##  $ ever_married     : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 2 1 2 2 2 ...
##  $ work_type        : Factor w/ 5 levels "children","Govt_job",..: 4 4 4 5 4 4 4 4 4 2 ...
##  $ Residence_type   : Factor w/ 2 levels "Rural","Urban": 2 1 2 1 2 1 2 2 1 1 ...
##  $ avg_glucose_level: num  229 106 171 174 186 ...
##  $ bmi              : num  36.6 32.5 34.4 24 29 27.4 22.8 24.2 29.7 36.8 ...
##  $ smoking_status   : Factor w/ 4 levels "formerly smoked",..: 1 2 3 2 1 2 2 4 2 3 ...
##  $ stroke           : Factor w/ 2 levels "0","1": 2 2 2 2 2 2 2 2 2 2 ...
```

Analyzing the Dataset

Summary Statistics

```
age.summary = data.frame(describe(data$age))
rownames(age.summary) <- "age"
bmi.summary = data.frame(describe(data$bmi))
rownames(bmi.summary) <- "bmi"
gluc.summary = data.frame(describe(data$avg_glucose_level))
rownames(gluc.summary) <- "avg_glucose_level"
num.summary = rbind(age.summary,bmi.summary,gluc.summary)
num.summary
```

```
##                   vars    n      mean       sd median  trimmed      mad   min
## age                  1 4908  42.86881 22.55613  44.00 43.17693 26.68680  0.08
## bmi                  1 4908  28.89456  7.85432  28.10 28.34422  6.96822 10.30
## avg_glucose_level    1 4908 105.29740 44.42555  91.68 97.00922 25.85654 55.12
##                      max  range       skew   kurtosis        se
## age                82.00  81.92 -0.1193767 -0.9890687 0.3219677
## bmi                97.60  87.30  1.0544181  3.3550926 0.1121131
## avg_glucose_level 271.74 216.62  1.6136322  1.9019220 0.6341333
```

```
table(data$gender)
```

```
##
## Female   Male
##   2897   2011
```

```
table(data$hypertension)
```

```
##
##    0    1
## 4457  451
```

```
table(data$heart_disease)
```

```
##
##    0    1
## 4665  243
```

```
table(data$ever_married)
```

```
##
##   No  Yes
## 1704 3204
```

```
table(data$work_type)
```

```
##
##      children      Govt_job  Never_worked       Private Self-employed
##           671           630            22          2810           775
```

```
table(data$Residence_type)
```

```
##
## Rural Urban
##  2418  2490
```

```
table(data$smoking_status)
```

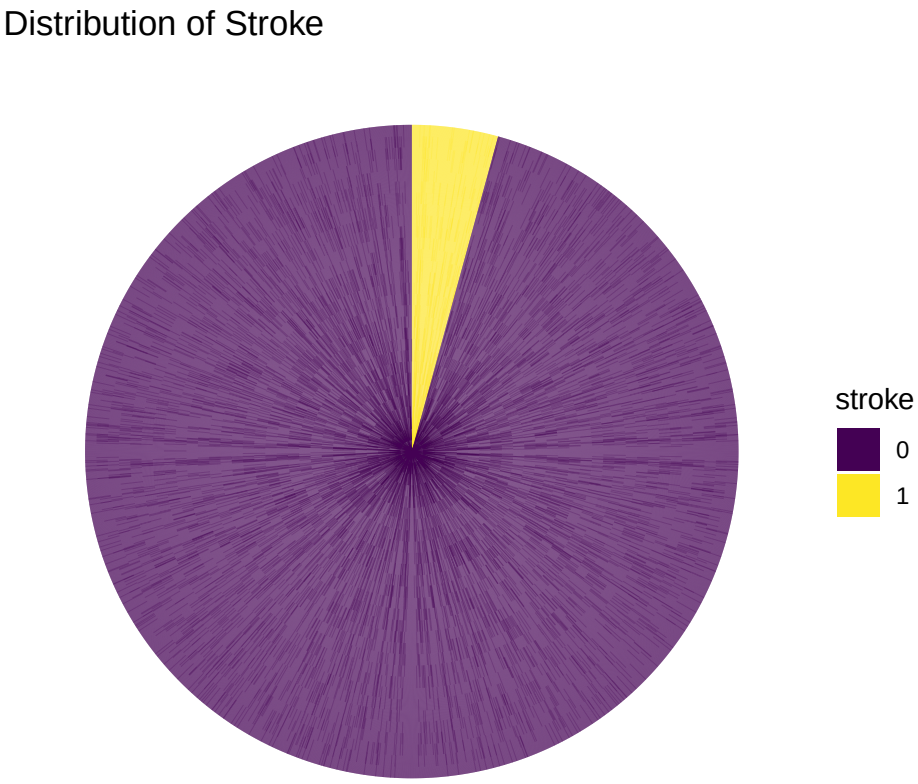
```
##
## formerly smoked    never smoked          smokes         Unknown
##             836            1852             737            1483
```

Correlation of Numeric Explanatory Variables, Distribution of Response Variable, and Outliers in the Dataset

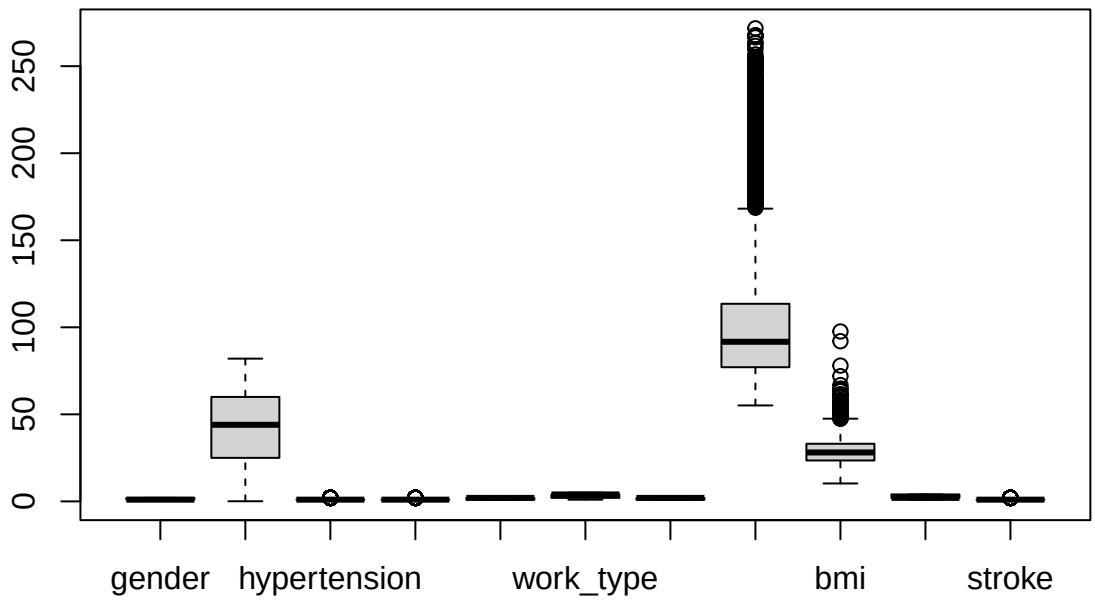
```
cor(data[sapply(data,is.numeric)],use = "pairwise.complete.obs")
```

```
##                        age avg_glucose_level       bmi
## age              1.0000000         0.2359996 0.3333142
## avg_glucose_level 0.2359996         1.0000000 0.1756717
## bmi              0.3333142         0.1756717 1.0000000
```

```
ggplot(data, aes(x="", y="", fill=stroke)) +
  geom_bar(stat="identity", width=1) +
  ggtitle("Distribution of Stroke") +
  coord_polar("y", start=0)+
  scale_fill_viridis(discrete = T) +
theme_void() # remove background, grid, numeric labels
```



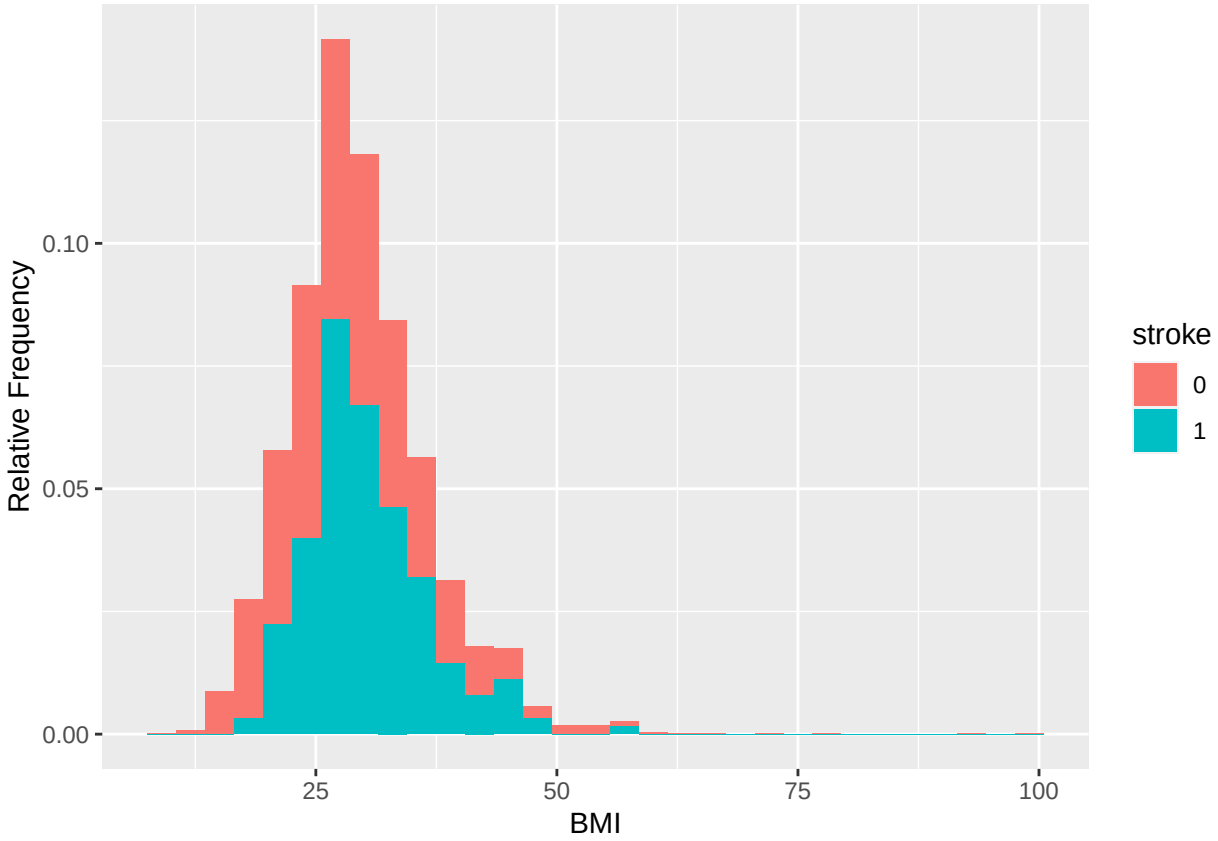

```
boxplot(data)
```



We can see that none of the numeric explanatory variables are highly correlated with each other. However, there is a high imbalance between stroke patients and non-stroke patients.We can also see many outliers in the average glucose level and BMI.

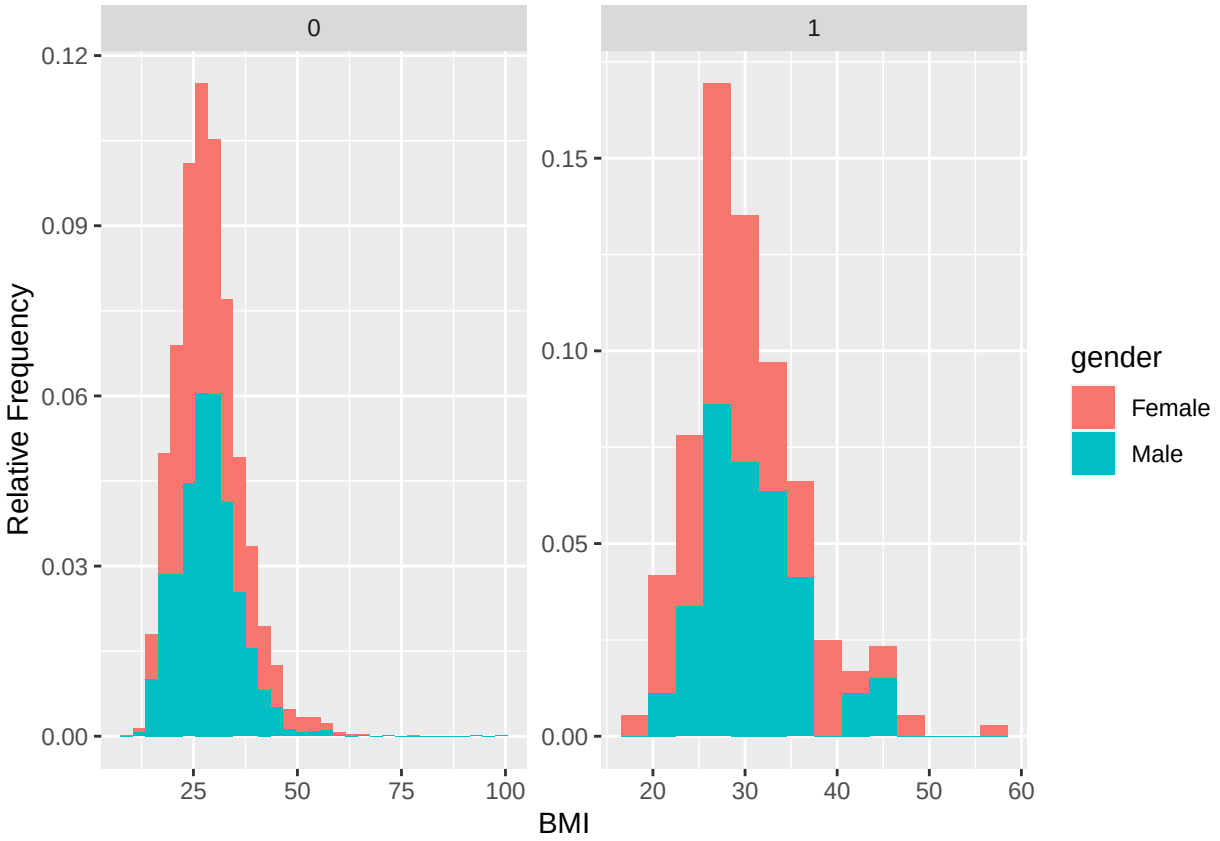
Histogram of BMI

```
ggplot(data, aes(x = bmi, fill = stroke)) +
  geom_histogram(binwidth = 3, aes(y = ..density..)) +
  labs(x = "BMI", y = "Relative Frequency")
```



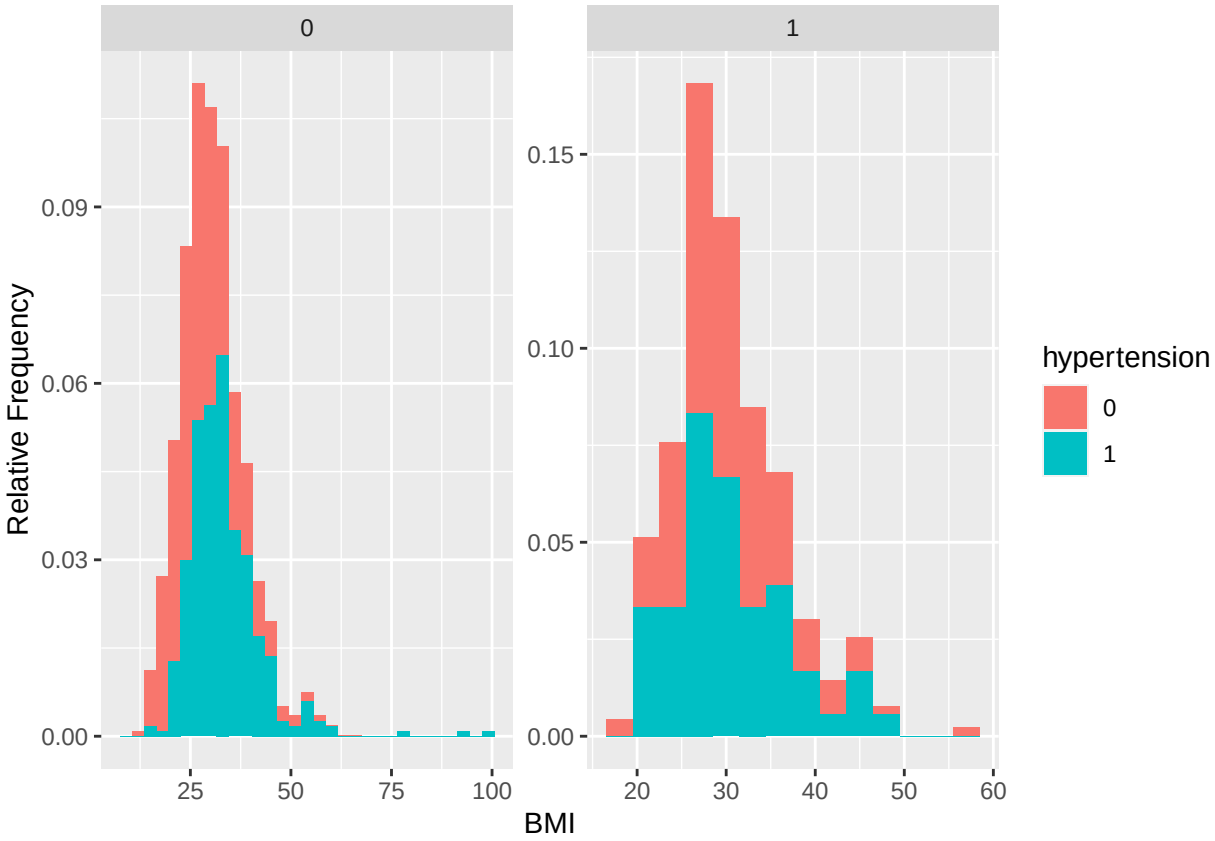
BMI by stroke status

```
ggplot(data, aes(x = bmi, fill = gender)) +
  geom_histogram(binwidth = 3, aes(y = ..density..)) +
  facet_wrap(~stroke, scales = "free") +
  labs(x = "BMI", y = "Relative Frequency")
```



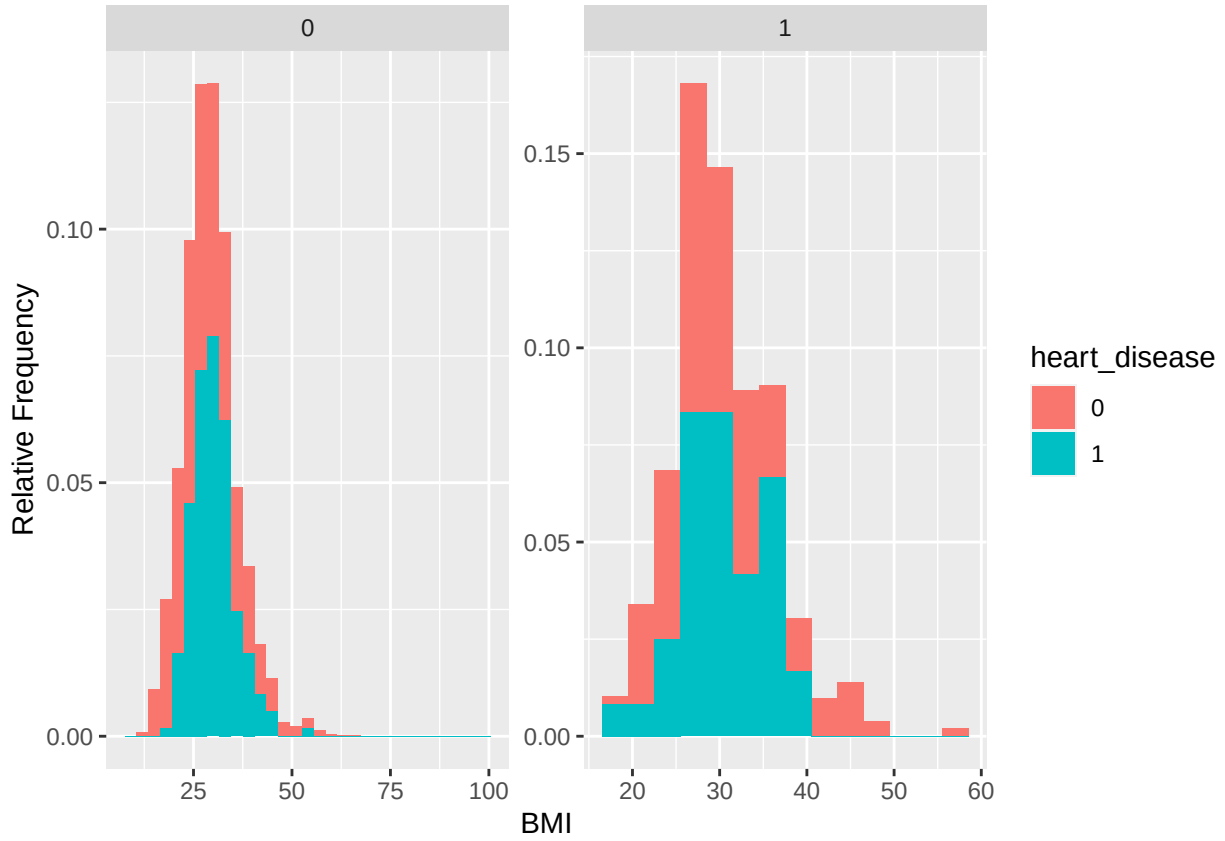
Here is a histogram of BMI by stroke status overlapped by gender. There does not seem to be a relationship between BMI and stroke status for males and females.

```
ggplot(data, aes(x = bmi, fill = hypertension)) +
  geom_histogram(binwidth = 3, aes(y = ..density..)) +  # Adjust binwidth and fill color as needed
  facet_wrap(~stroke, scales = "free") +  # Facet by binomial_var
  labs(x = "BMI", y = "Relative Frequency")
```



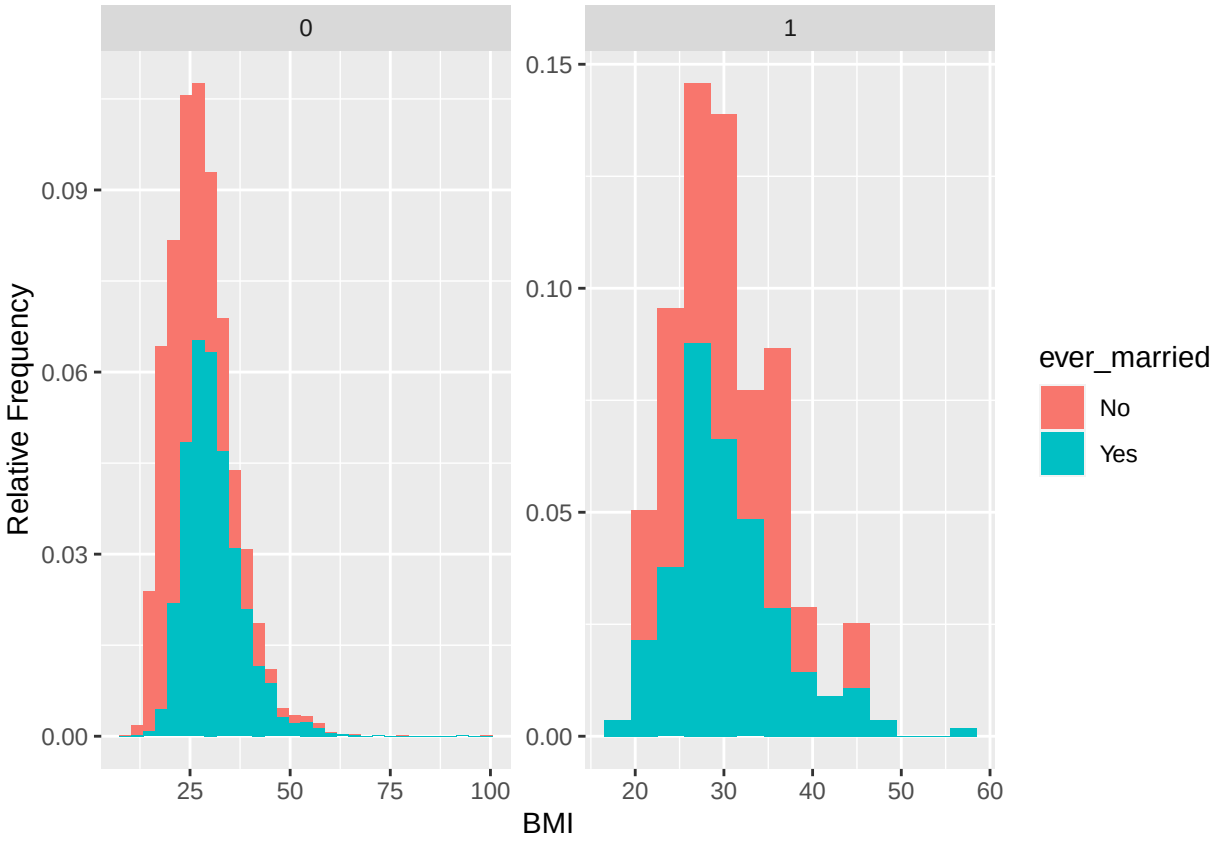
Histogram of BMI by stroke status overlapped by hypertension. There doesn't look to be an obvious relationship between BMI and stroke for people that have hypertension and don't.

```
ggplot(data, aes(x = bmi, fill = heart_disease)) +
  geom_histogram(binwidth = 3, aes(y = ..density..)) +  # Adjust binwidth and fill color as needed
  facet_wrap(~stroke, scales = "free") +  # Facet by binomial_var
  labs(x = "BMI", y = "Relative Frequency")
```



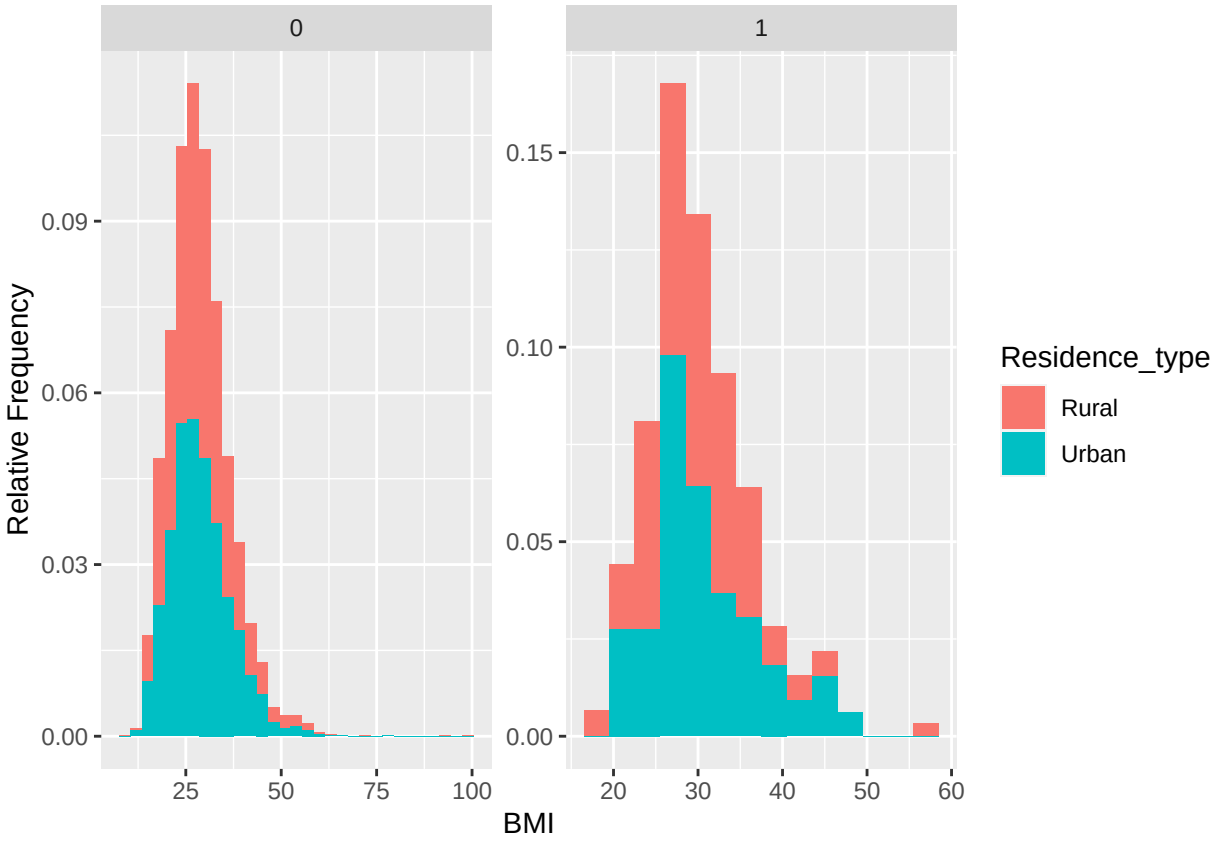
Histogram of BMI by stroke status overlapped by heart disease. Data doesn't seem to show a difference in stroke status by heart disease.

```
ggplot(data, aes(x = bmi, fill = ever_married)) +
  geom_histogram(binwidth = 3, aes(y = ..density..)) +  # Adjust binwidth and fill color as needed
  facet_wrap(~stroke, scales = "free") +  # Facet by binomial_var
  labs(x = "BMI", y = "Relative Frequency")
```



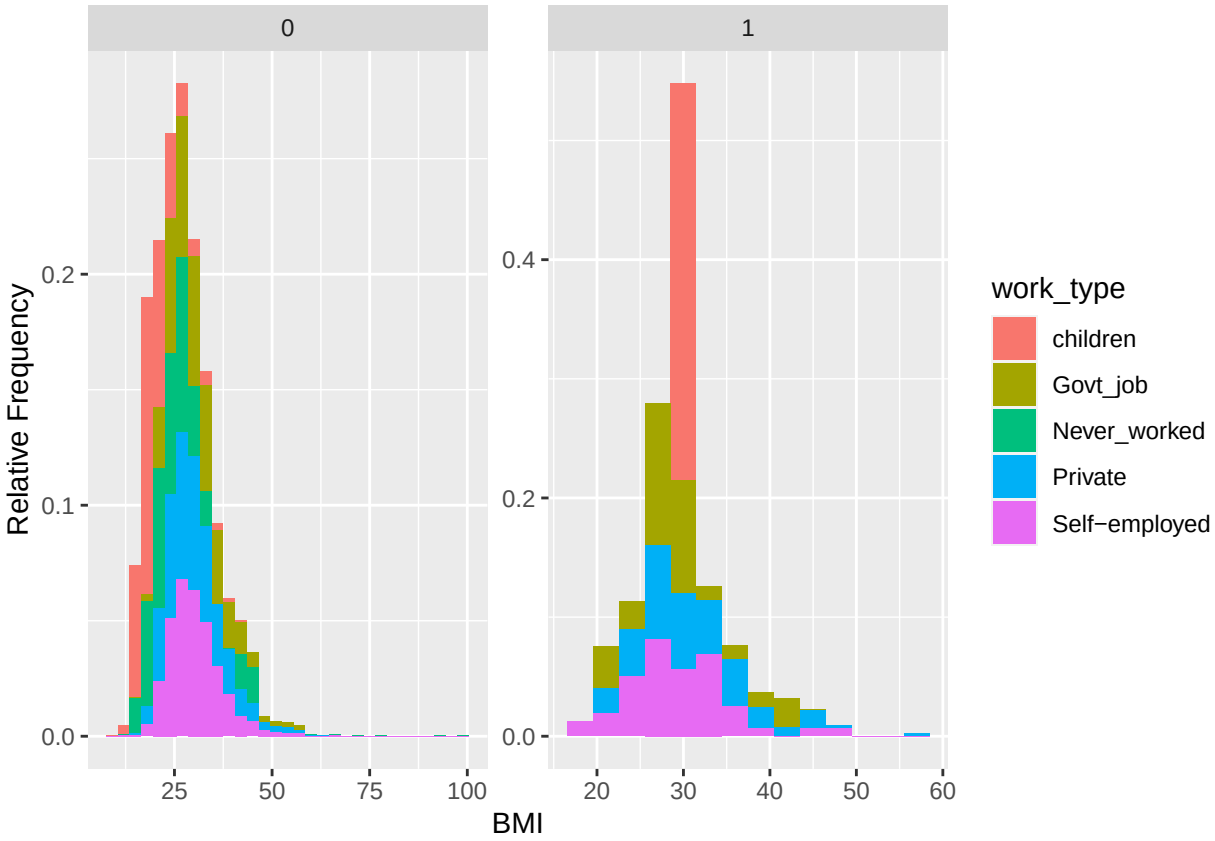
Histogram of BMI by stroke status overlapped by ever married. Marriage doesn't seem to differ between people who had strokes or not.

```
ggplot(data, aes(x = bmi, fill = Residence_type)) +
  geom_histogram(binwidth = 3, aes(y = ..density..)) +  # Adjust binwidth and fill color as needed
  facet_wrap(~stroke, scales = "free") +  # Facet by binomial_var
  labs(x = "BMI", y = "Relative Frequency")
```



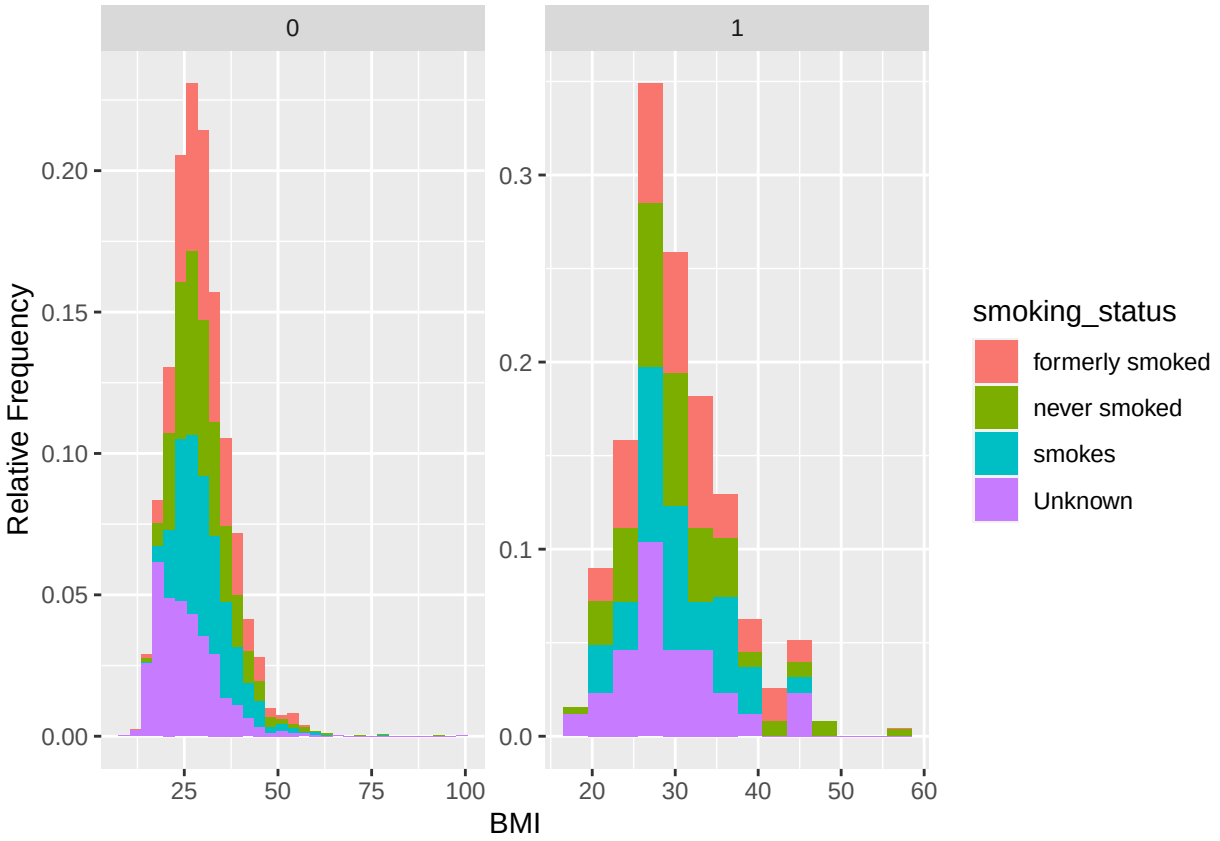
Histogram of BMI by stroke status overlapped by Residence type. There doesn't seem to be much of a difference between the two.

```
ggplot(data, aes(x = bmi, fill = work_type)) +
  geom_histogram(binwidth = 3, aes(y = ..density..)) +  # Adjust binwidth and fill color as needed
  facet_wrap(~stroke, scales = "free") +  # Facet by binomial_var
  labs(x = "BMI", y = "Relative Frequency")
```



Histogram of BMI by stroke status overlapped by work type.

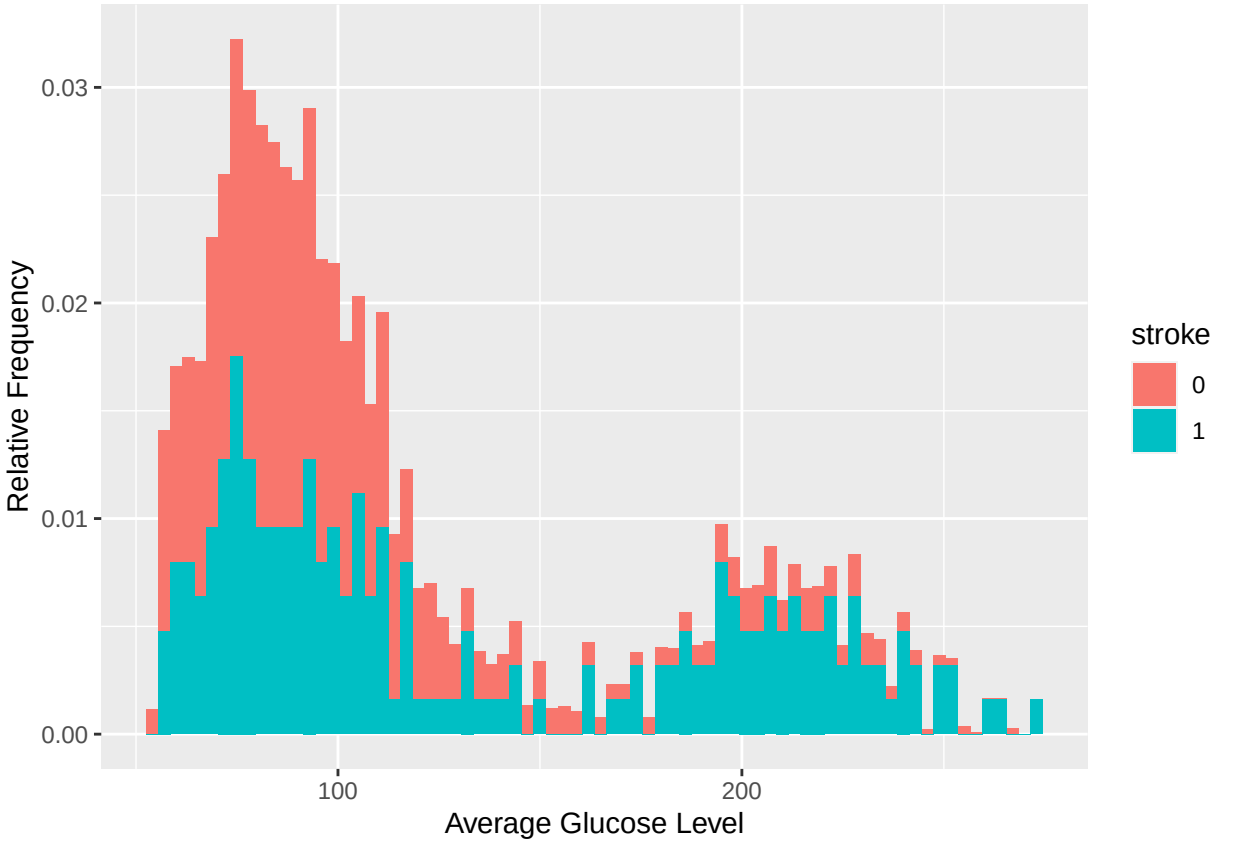
```
ggplot(data, aes(x = bmi, fill = smoking_status)) +
  geom_histogram(binwidth = 3, aes(y = ..density..)) +  # Adjust binwidth and fill color as needed
  facet_wrap(~stroke, scales = "free") +  # Facet by binomial_var
  labs(x = "BMI", y = "Relative Frequency")
```



Histogram of BMI by stroke status overlapped by smoking status.

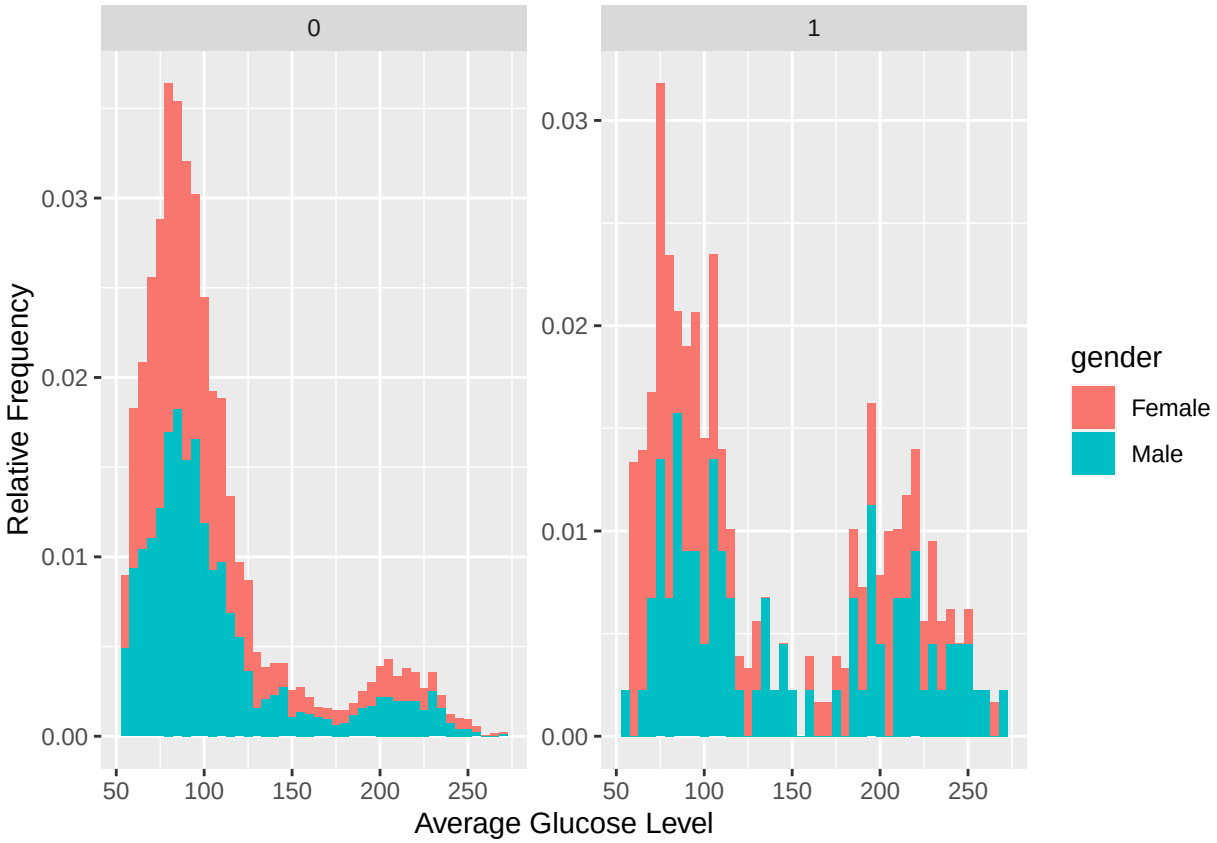
Histogram of Average Glucose Levels

```
ggplot(data, aes(x = avg_glucose_level, fill = stroke)) +
  geom_histogram(binwidth = 3, aes(y = ..density..)) +
  labs(x = "Average Glucose Level", y = "Relative Frequency")
```



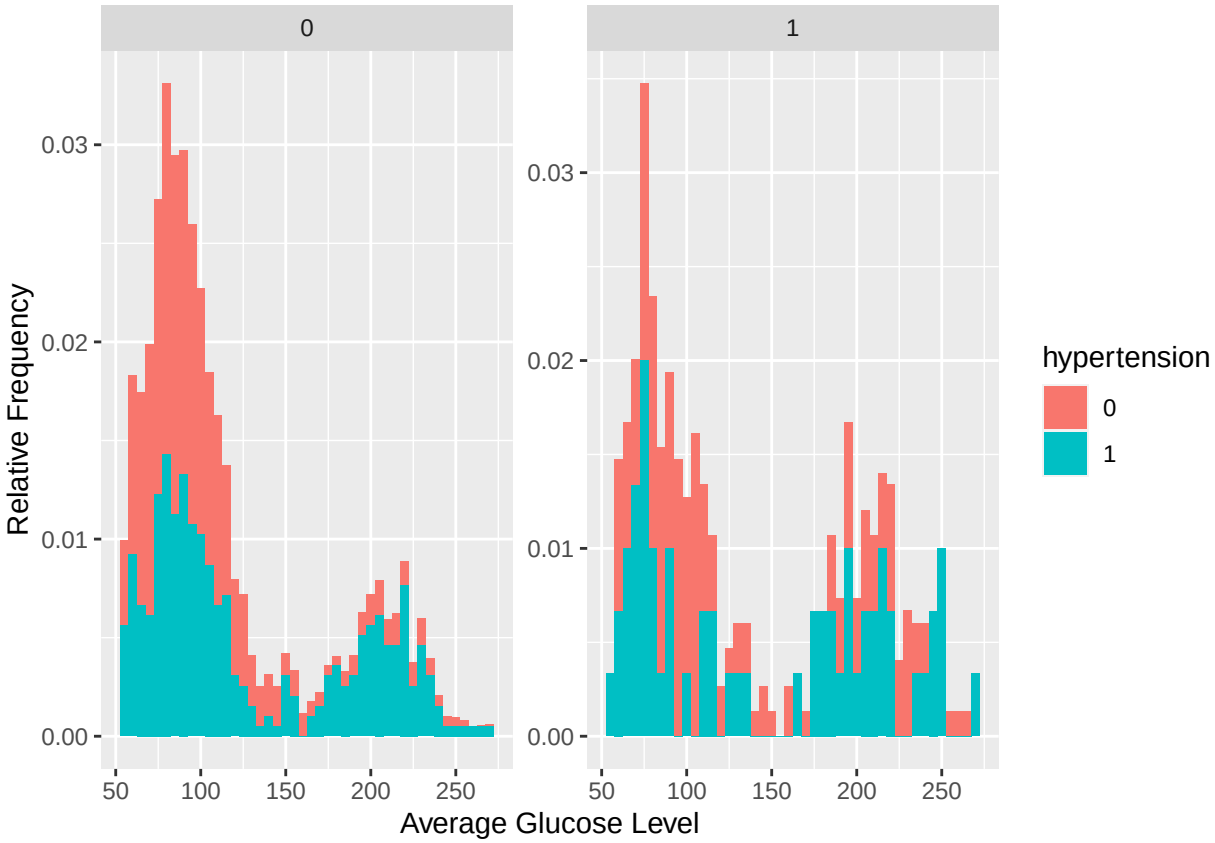
Histogram of average glucose level by stroke status where not much difference can be seen between the average glucose level by stroke status

```
ggplot(data, aes(x = avg_glucose_level, fill = gender)) +
  geom_histogram(binwidth = 5, aes(y = ..density..)) +  # Adjust binwidth and fill color as needed
  facet_wrap(~stroke, scales = "free") +  # Facet by binomial_var
  labs(x = "Average Glucose Level", y = "Relative Frequency")
```



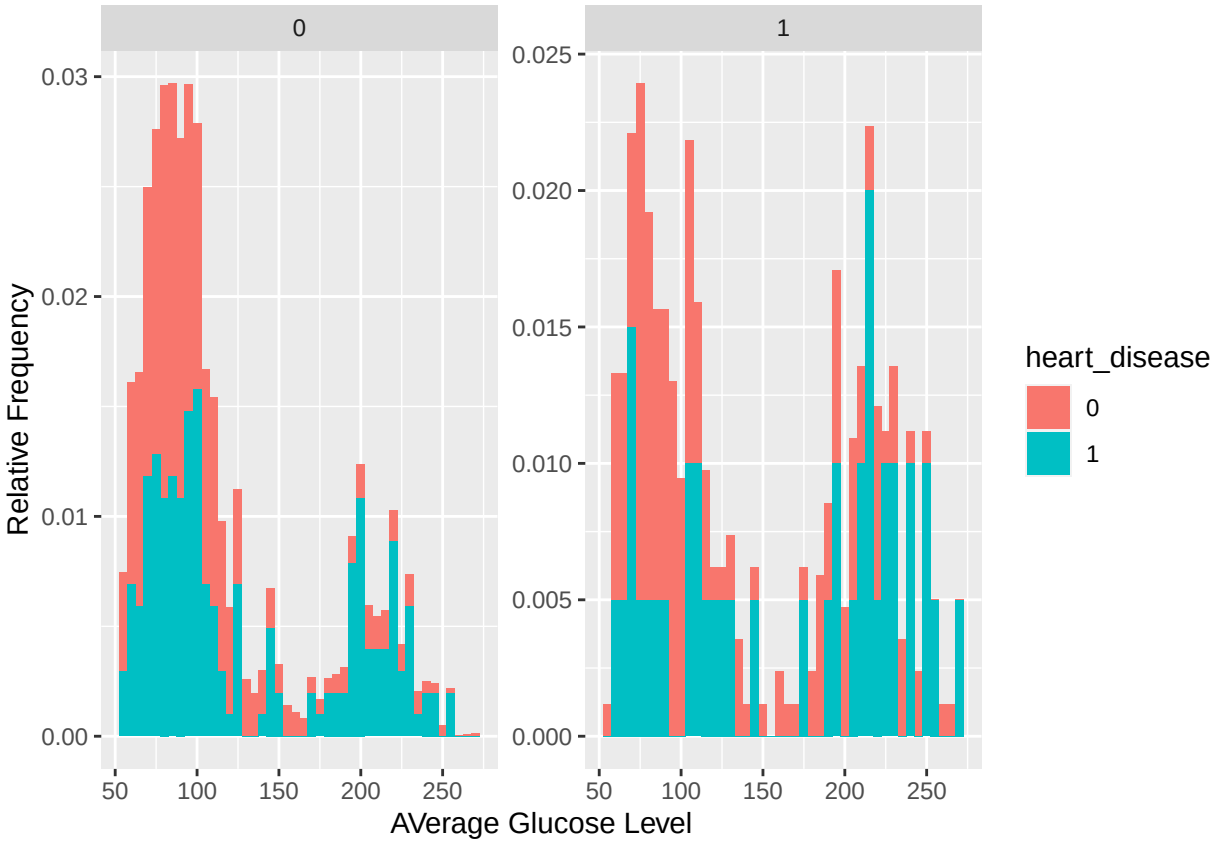
Histogram of average glucose level by stroke status overlapped by gender. Hard to get anything from this when the stroke=1 histogram looks more discrete, but not much difference.

```
ggplot(data, aes(x = avg_glucose_level, fill = hypertension)) +
  geom_histogram(binwidth = 5, aes(y = ..density..)) +  # Adjust binwidth and fill color as needed
  facet_wrap(~stroke, scales = "free") +  # Facet by binomial_var
  labs(x = "Average Glucose Level", y = "Relative Frequency")
```



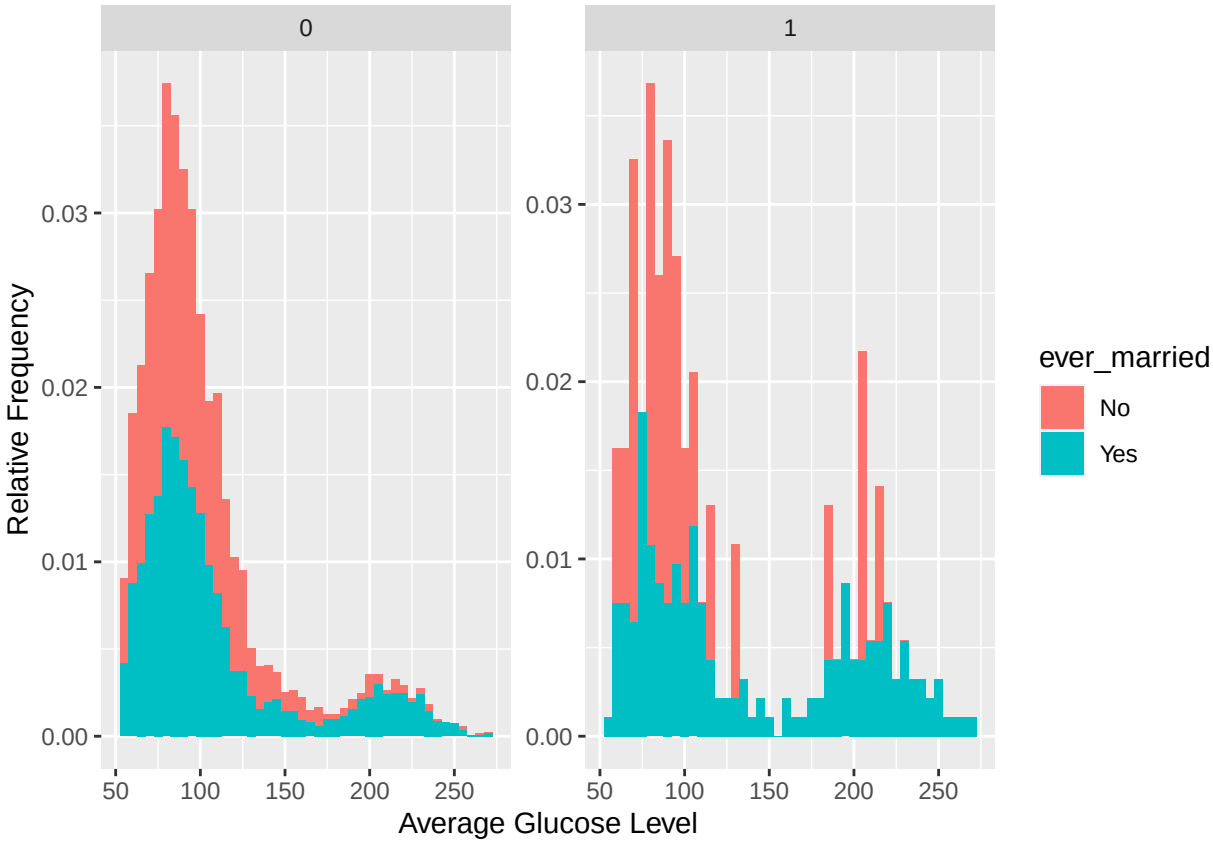
Histogram of average glucose level by stroke status overlapped by hypertension, but not much difference.

```
ggplot(data, aes(x = avg_glucose_level, fill = heart_disease)) +
  geom_histogram(binwidth = 5, aes(y = ..density..)) +  # Adjust binwidth and fill color as needed
  facet_wrap(~stroke, scales = "free") +  # Facet by binomial_var
  labs(x = "AVerage Glucose Level", y = "Relative Frequency")
```



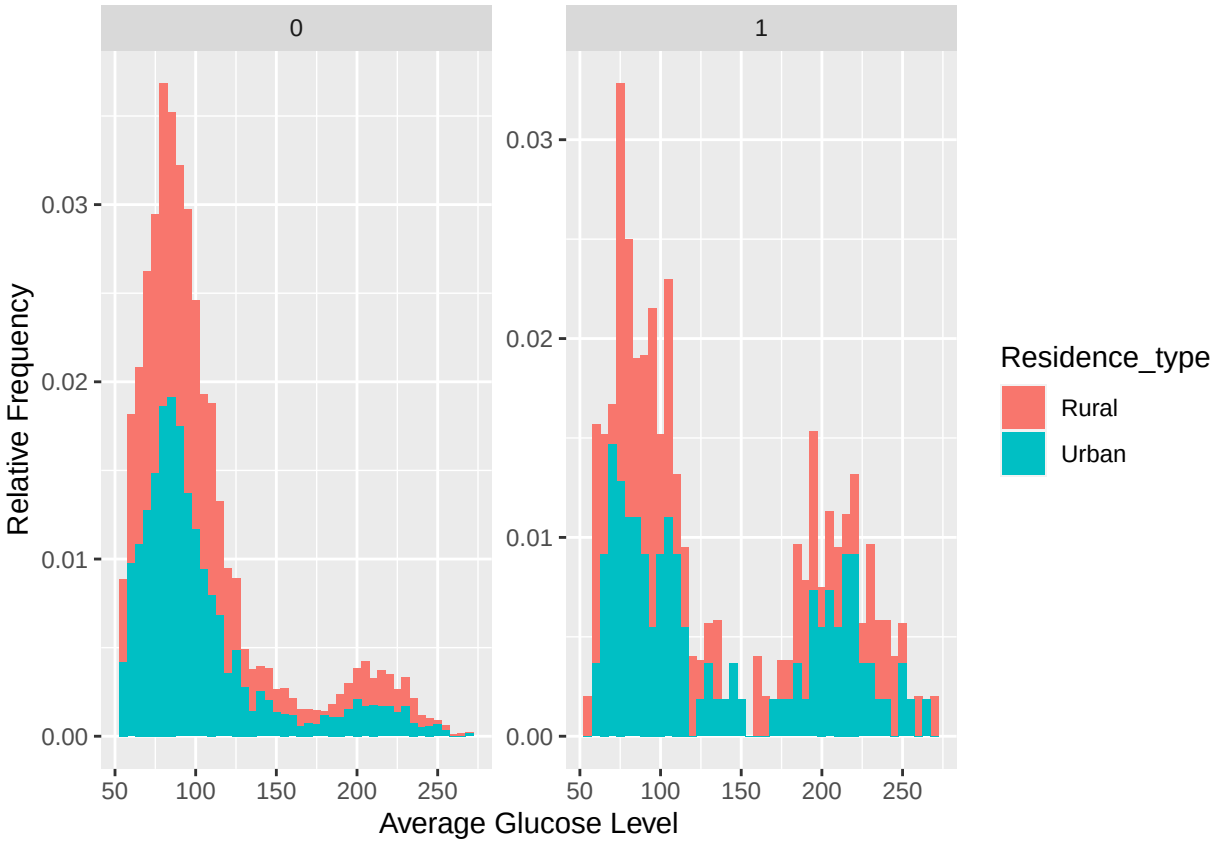
Histogram of average glucose level by stroke status overlapped by heart disease, but not seeing much difference.

```
ggplot(data, aes(x = avg_glucose_level, fill = ever_married)) +
  geom_histogram(binwidth = 5, aes(y = ..density..)) +  # Adjust binwidth and fill color as needed
  facet_wrap(~stroke, scales = "free") +  # Facet by binomial_var
  labs(x = "Average Glucose Level", y = "Relative Frequency")
```



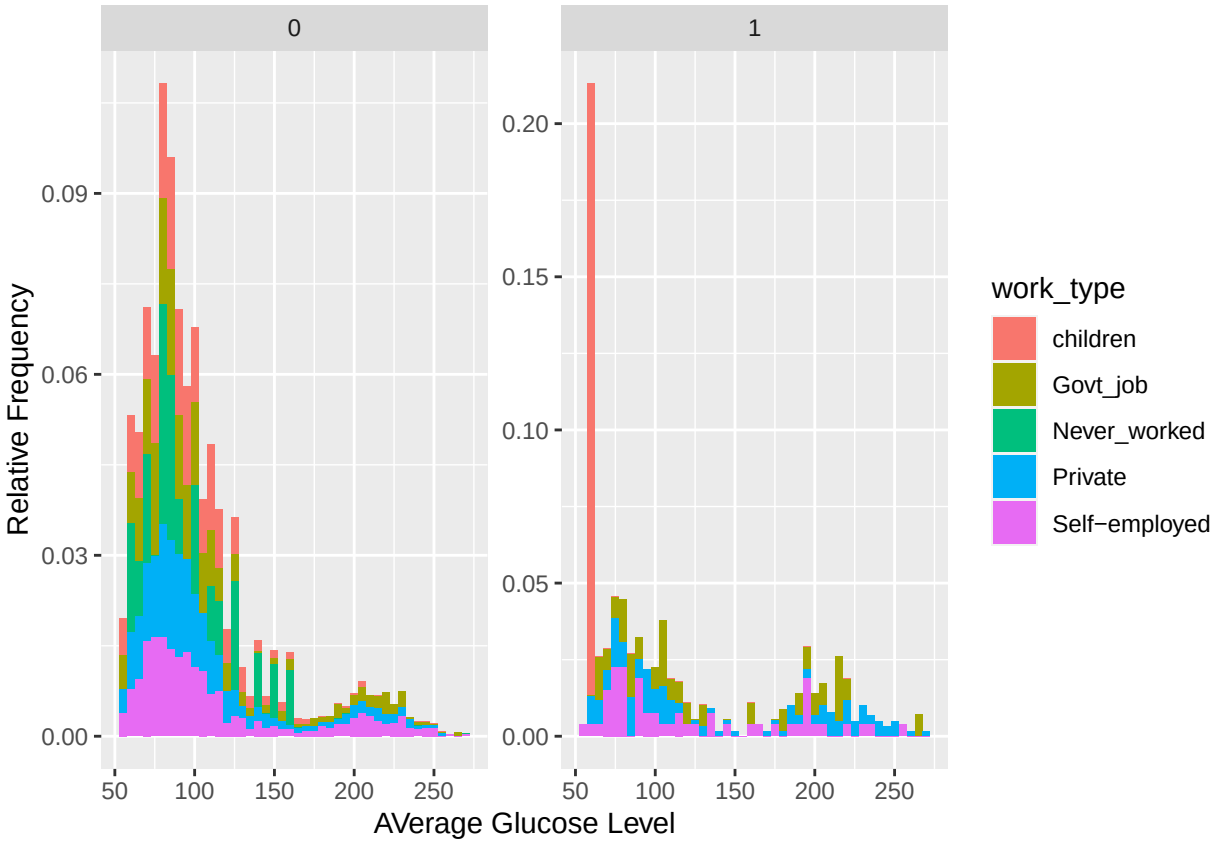
Histogram of average glucose level by stroke status overlapped by ever married, but not much difference seen.

```
ggplot(data, aes(x = avg_glucose_level, fill = Residence_type)) +
  geom_histogram(binwidth = 5, aes(y = ..density..)) +  # Adjust binwidth and fill color as needed
  facet_wrap(~stroke, scales = "free") +  # Facet by binomial_var
  labs(x = "Average Glucose Level", y = "Relative Frequency")
```



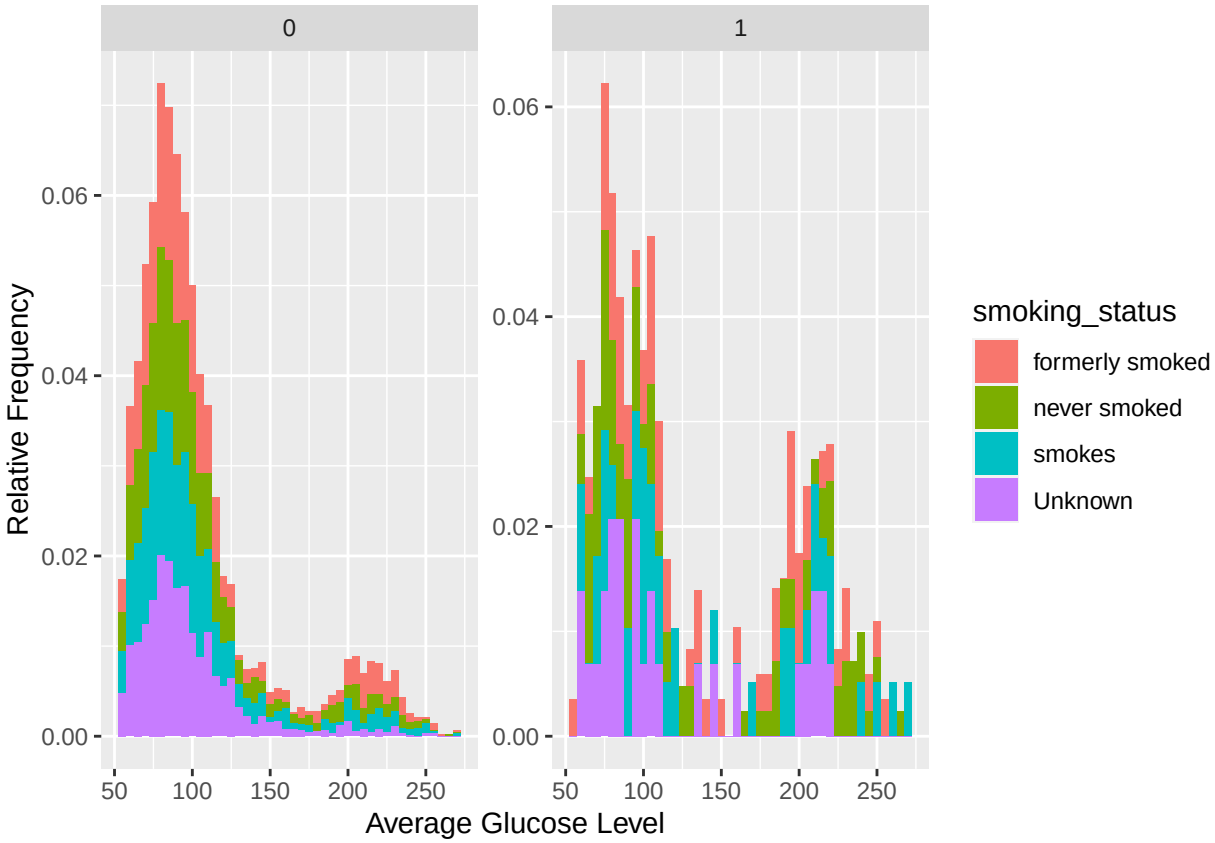
Histogram of average glucose level by stroke status overlapped by Residence type. Hard to see a difference here.

```
ggplot(data, aes(x = avg_glucose_level, fill = work_type)) +
  geom_histogram(binwidth = 5, aes(y = ..density..)) +  # Adjust binwidth and fill color as needed
  facet_wrap(~stroke, scales = "free") +  # Facet by binomial_var
  labs(x = "AVerage Glucose Level", y = "Relative Frequency")
```



Histogram of average glucose level by stroke status overlapped by work type.

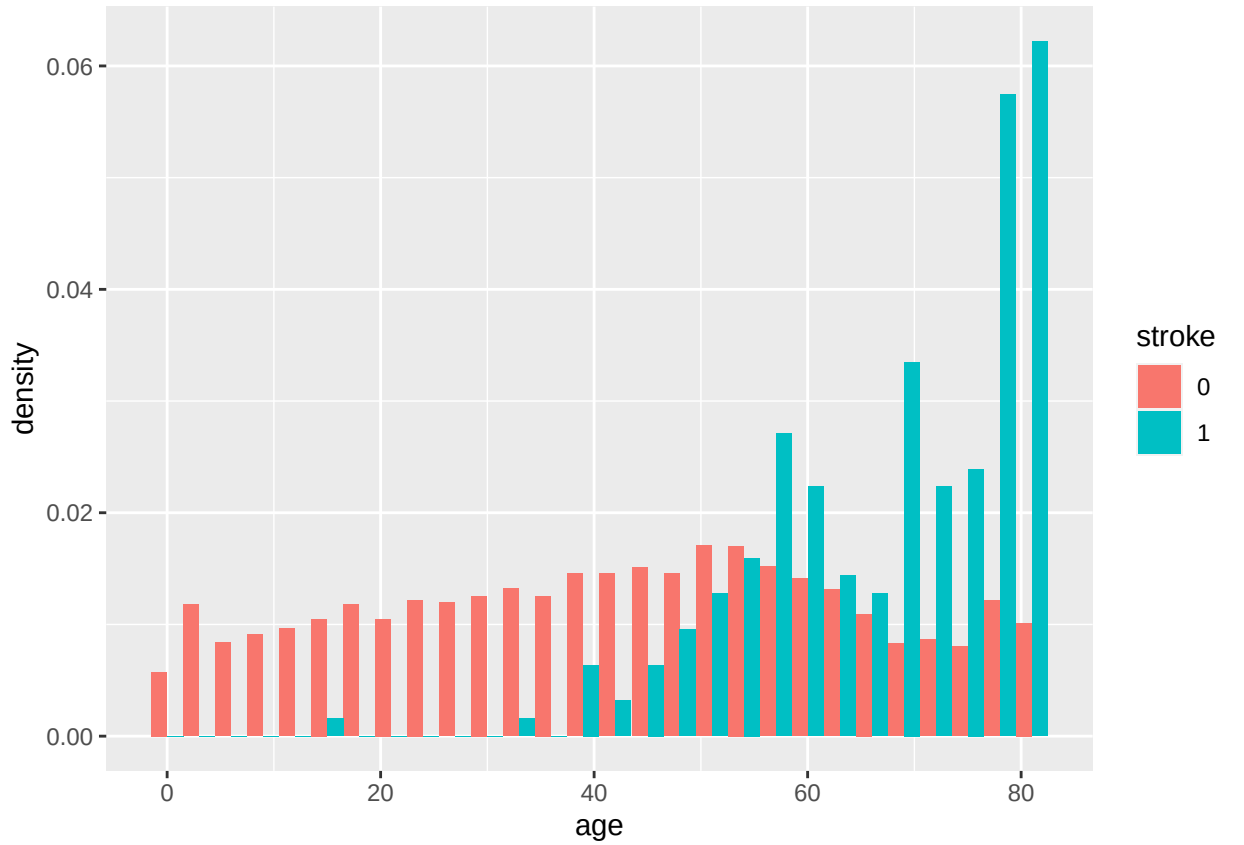
```
ggplot(data, aes(x = avg_glucose_level, fill = smoking_status)) +
  geom_histogram(binwidth = 5, aes(y = ..density..)) +  # Adjust binwidth and fill color as needed
  facet_wrap(~stroke, scales = "free") +  # Facet by binomial_var
  labs(x = "Average Glucose Level", y = "Relative Frequency")
```



Histogram of average glucose level by stroke status overlapped by smoking status.

Histogram of Age

```
ggplot(data, aes(x = age, fill = stroke)) +
  geom_histogram(binwidth = 3, aes(y = ..density..), position = "dodge")
```



Skewed left histogram in terms of age for the stroke patients, seemingly indicating that the age of patients who encountered a stroke is higher than those who did not.

Prediction Models

We decided that Generalized Linear Model and Random Forest are the best prediction models for our dataset.

Optimizing Random Forest

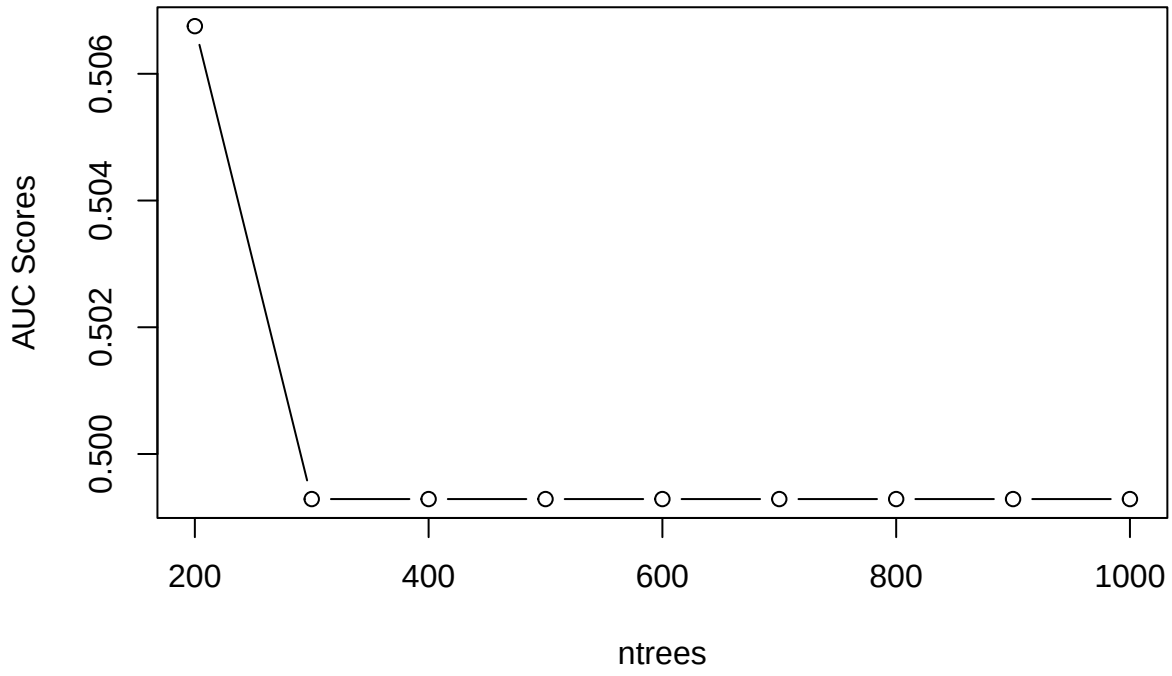
First, we want to tune the number of trees, m, and depth to maximize area under curve (AUC) score.

Finding Optimal Number of Trees

Finding the optimal number of trees to maximize AUC score.

```
trees <- seq(from = 200, to = 1000, by = 100)
z <- length(trees)
rf.max.auc1 <- vector(mode = "numeric", length = z)
N = nrow(data)
for (i in 1:z) {
  set.seed(1)
  index <- sample(1:N, size = N*.7, replace = F)
  train_x <- data[index, 1:10]
```

```
  test_x <- data[-index, 1:10]
  train_y <- data[index, 11]
  test_y <- data[-index, 11]
  train <- data[index,]
  test <- data[-index,]
  rf_model1 <- randomForest(stroke~.,data = train, xtest = test_x,
                      ytest=test_y,ntree=trees[i], keep.forest = TRUE)
  rf_predicted_class1 <- rf_model1$test$predicted
  rf.max.auc1[i] <- max(auc(roc(rf_predicted_class1,test_y)))
}
optimal.ntrees <- trees[which.max(rf.max.auc1)]
plot(trees, rf.max.auc1, type = "b", xlab = "ntrees", ylab = "AUC Scores")
```

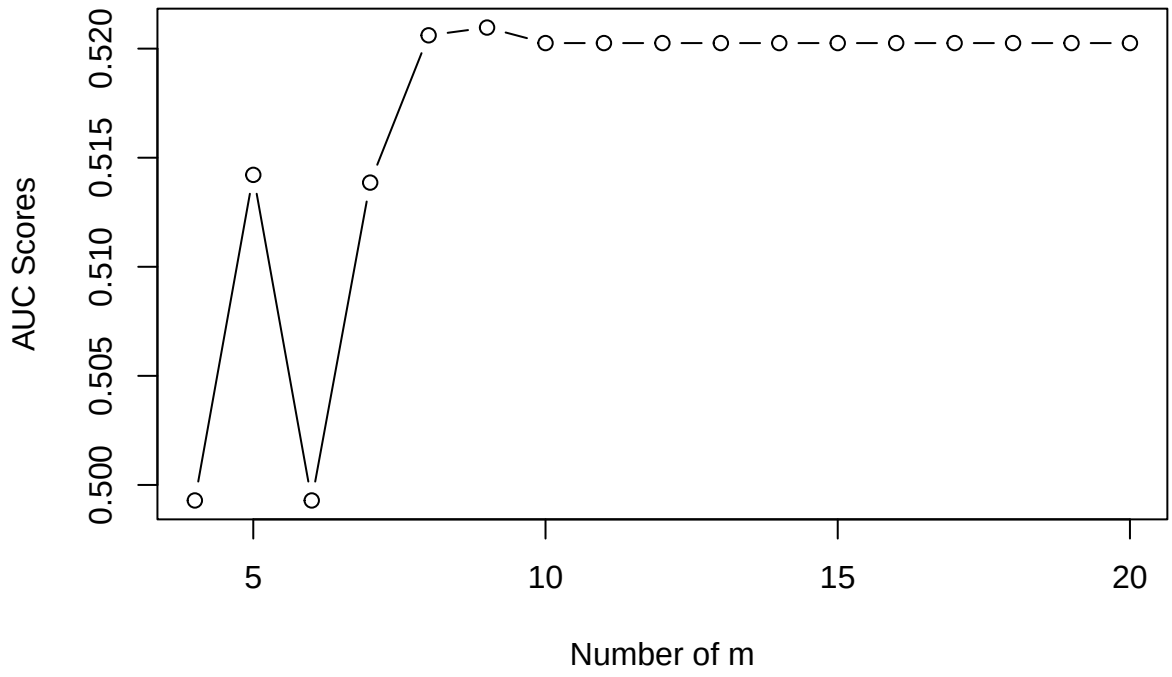


Finding Optimal m

We will now find the optimal m using the optimal number of trees based on AUC score. As we know, too large m will have high correlation and low bias.

```
m <- seq(from = 4, to = 20, by = 1)
h <- length(m)
rf.max.auc2 <- vector(mode = "numeric", length = h)
for (i in 1:h) {
  set.seed(1)
  index <- sample(1:N, size = N*.7, replace = F)
```

```
  train_x <- data[index, 1:10]
  test_x <- data[-index, 1:10]
  train_y <- data[index, 11]
  test_y <- data[-index, 11]
  train <- data[index,]
  test <- data[-index,]
  rf_model2 <- randomForest(stroke~.,data = train, xtest = test_x,
                    ytest=test_y, mtry = m[i],
                    ntree=optimal.ntrees, keep.forest = TRUE)
  rf_predicted_class2 <- rf_model2$test$predicted
  rf.max.auc2[i] <- max(auc(roc(rf_predicted_class2,test_y)))
}
optimal.m <- m[which.max(rf.max.auc2)]
plot(m, rf.max.auc2, type = "b", xlab = "Number of m", ylab = "AUC Scores")
```

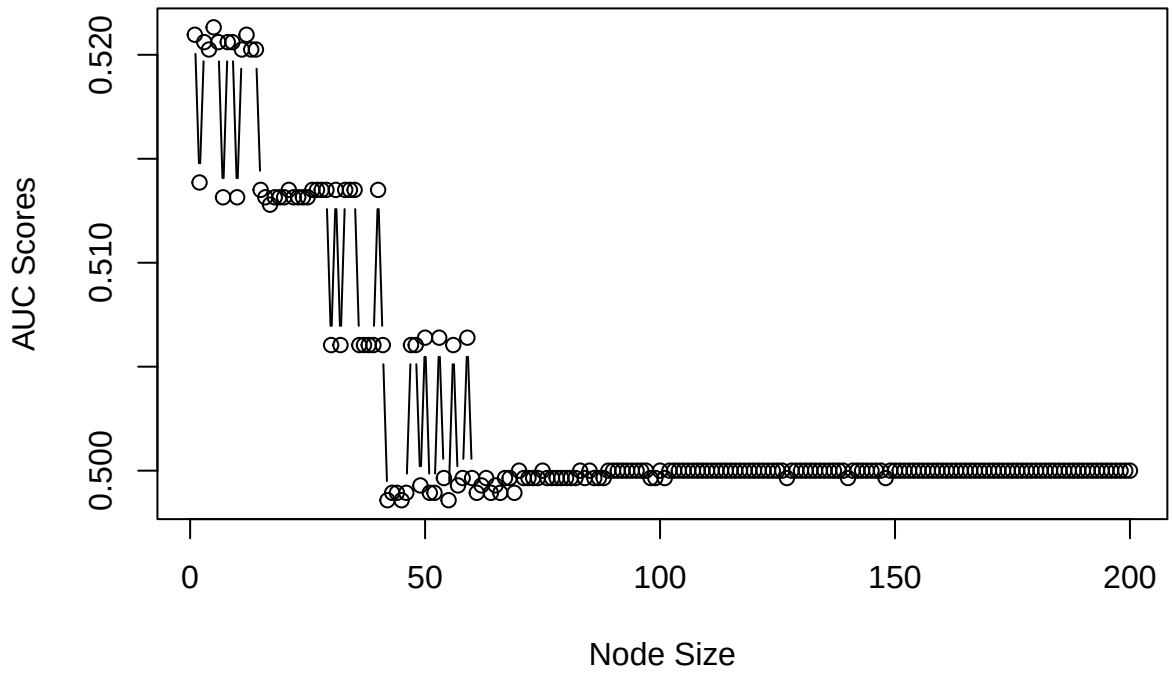


Finding Optimal Tree Depth

We will now find the optimal tree depth using the optimal number of trees and optimal m based on AUC score. The smaller the node size, the deeper the tree.

```
depth <- seq(from = 1, to = 200, by = 1)
d <- length(depth)
rf.max.auc3 <- vector(mode = "numeric", length = d)
for (i in 1:d) {
```

```
  set.seed(1)
  index <- sample(1:N, size = N*.7, replace = F)
  train_x <- data[index, 1:10]
  test_x <- data[-index, 1:10]
  train_y <- data[index, 11]
  test_y <- data[-index, 11]
  train <- data[index,]
  test <- data[-index,]
  rf_model3 <- randomForest(stroke~.,data = train, xtest = test_x,
                    ytest=test_y, mtry = optimal.m, nodesize = depth[i],
                    ntree=optimal.ntrees, keep.forest = TRUE)
  rf_predicted_class3 <- rf_model3$test$predicted
  rf.max.auc3[i] <- max(auc(roc(rf_predicted_class3,test_y)))
}
optimal.depth <- depth[which.max(rf.max.auc3)]
plot(depth, rf.max.auc3, type = "b", xlab = "Node Size", ylab = "AUC Scores")
```



Comparing General Linear Model (GLM) and Optimized Random Forest (RF)

Here we have a loop that computes misclassification rates and AUC scores for both GLM and RF 20 times, and each time using a different seed and training and test set.

```
n=20
log_misclass_rate <- vector(mode = "numeric", length = n)
rf_misclass_rate <- vector(mode = "numeric", length = n)
log_auc <- vector(mode = "numeric", length = n)
rf_auc <- vector(mode = "numeric", length = n)
for (i in 1:n){
  set.seed(i)
  index <- sample(1:N, size = N*.7, replace = F)
  train_x <- data[index, 1:10]
  test_x <- data[-index, 1:10]
  train_y <- data[index, 11]
  test_y <- data[-index, 11]
  train <- data[index,]
  test <- data[-index,]
  logitmodel <- glm(stroke ~., family = binomial(link = logit), data = train)

  pred_logitmod <- predict(logitmodel, newdata = test, type = "response")
  log_predicted_class <- ifelse(pred_logitmod > 0.5, 1, 0)
  log_predicted_class <- factor(log_predicted_class, levels = c(0, 1))

  log_error <- table(actual = test_y, predicted = log_predicted_class)

  log_misclass_rate[i] <- 1-sum(diag(log_error))/sum(log_error)

  rf_model <- randomForest(stroke~.,data = train, xtest = test_x,
                     ytest=test_y, mtry = optimal.m, nodesize = optimal.depth,
                     ntree=optimal.ntrees, keep.forest = TRUE)
  rf_predicted_class <- rf_model$test$predicted
  rf_error <- table(actual = test_y, predicted = rf_predicted_class)
  rf_misclass_rate[i] <- 1-sum(diag(rf_error))/sum(rf_error)

  log_auc[i] <- auc(roc(log_predicted_class, test_y))
  rf_auc[i] <- auc(roc(rf_predicted_class,test_y))
}
```

Confusion Matrix

GLM

```
log.cm = confusionMatrix(log_predicted_class,test_y, mode = "everything", positive ="1")
log.cm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1413   60
##          1    0    0
##
##                Accuracy : 0.9593
##                  95% CI : (0.9479, 0.9688)
```

```
##     No Information Rate : 0.9593
##     P-Value [Acc > NIR] : 0.5343
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : 2.599e-14
##
##             Sensitivity : 0.00000
##             Specificity : 1.00000
##          Pos Pred Value :     NaN
##          Neg Pred Value : 0.95927
##               Precision :      NA
##                  Recall : 0.00000
##                      F1 :      NA
##              Prevalence : 0.04073
##          Detection Rate : 0.00000
##    Detection Prevalence : 0.00000
##       Balanced Accuracy : 0.50000
##
##        'Positive' Class : 1
##
```

RF

```
rf.cm = confusionMatrix(rf_predicted_class, test_y, mode = "everything", positive ="1")
rf.cm
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 1408   59
##          1    5    1
##
##                Accuracy : 0.9566
##                  95% CI : (0.9449, 0.9664)
##     No Information Rate : 0.9593
##     P-Value [Acc > NIR] : 0.7278
##
##                   Kappa : 0.0231
##
##  Mcnemar's Test P-Value : 3.472e-11
##
##             Sensitivity : 0.0166667
##             Specificity : 0.9964614
##          Pos Pred Value : 0.1666667
##          Neg Pred Value : 0.9597819
##               Precision : 0.1666667
##                  Recall : 0.0166667
##                      F1 : 0.0303030
##              Prevalence : 0.0407332
##          Detection Rate : 0.0006789
```

```
##    Detection Prevalence : 0.0040733
##       Balanced Accuracy : 0.5065640
##
##        'Positive' Class : 1
##
```

Summary Results Comparing Predictions of GLM and RF

This shows a table of misclassification rate and AUC score between GLM and RF as well as the summary of them.

```
results_misclass = data.frame(Log_Error = log_misclass_rate, RF_Error = rf_misclass_rate)
results_auc = data.frame(Log_AUC = log_auc, RF_AUC = rf_auc)
summary_results_misclass=describe(results_misclass)
summary_results_auc = describe(results_auc)
results_misclass
```

```
##     Log_Error   RF_Error
## 1  0.04412763 0.04548540
## 2  0.04480652 0.04548540
## 3  0.03937542 0.04344874
## 4  0.04684318 0.04752206
## 5  0.04820095 0.05227427
## 6  0.03530210 0.03598099
## 7  0.04548540 0.04616429
## 8  0.03598099 0.03665988
## 9  0.04344874 0.04616429
## 10 0.04141208 0.04548540
## 11 0.04480652 0.04616429
## 12 0.04276986 0.04480652
## 13 0.04005431 0.04141208
## 14 0.03530210 0.03665988
## 15 0.04412763 0.04412763
## 16 0.03801765 0.03937542
## 17 0.04752206 0.04955872
## 18 0.04344874 0.04480652
## 19 0.04073320 0.04276986
## 20 0.04073320 0.04344874
```

```
results_auc
```

```
##      Log_AUC    RF_AUC
## 1  0.5149254 0.5213212
## 2  0.5000000 0.5140854
## 3  0.5166667 0.5065640
## 4  0.5071429 0.5000000
## 5  0.5000000 0.5045458
## 6  0.5000000 0.5089117
## 7  0.5000000 0.4996444
## 8  0.5000000 0.5178116
## 9  0.5000000 0.4985806
## 10 0.5000000 0.4978754
```

```
## 11 0.4996449 0.4989347
## 12 0.5000000 0.5065181
## 13 0.5083333 0.4996461
## 14 0.5094340 0.5178116
## 15 0.4992908 0.5144546
## 16 0.5085757 0.5078700
## 17 0.5000000 0.5057173
## 18 0.5076923 0.5069821
## 19 0.5079795 0.5228769
## 20 0.5000000 0.5065640
```

```
summary_results_misclass
```

```
##           vars  n mean sd median trimmed mad  min  max range  skew kurtosis se
## Log_Error    1 20 0.04  0   0.04    0.04   0 0.04 0.05  0.01 -0.33    -1.04  0
## RF_Error     2 20 0.04  0   0.04    0.04   0 0.04 0.05  0.02 -0.33    -0.43  0
```

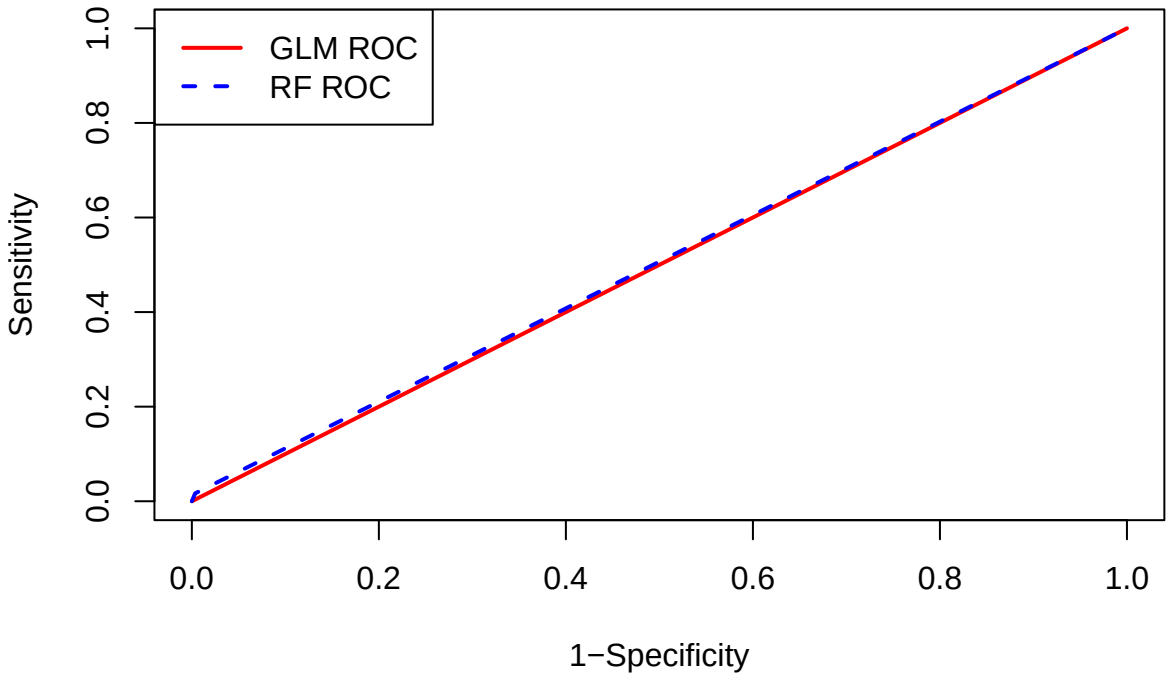
```
summary_results_auc
```

```
##         vars  n mean   sd median trimmed  mad min  max range skew kurtosis se
## Log_AUC    1 20 0.50 0.01   0.50    0.50 0.00 0.5 0.52  0.02 0.87    -0.62  0
## RF_AUC     2 20 0.51 0.01   0.51    0.51 0.01 0.5 0.52  0.03 0.44    -1.11  0
```

Random forest and General Linear Model displayed similar misclassification rates that shows they both performed well; however, we noticed most of the data was stroke=0. Therefore, misclassification rate is not good for gauging the performance of each prediction model. AUC scores from ROC must be used to determine the performance of each model.

Graphing AUC Scores of GLM and RF

```
log_spec <- AUC::specificity(log_predicted_class,test_y)$measure
log_sens <- AUC::sensitivity(log_predicted_class,test_y)$measure
rf_spec <- AUC::specificity(rf_predicted_class,test_y)$measure
rf_sens <- AUC::sensitivity(rf_predicted_class,test_y)$measure
plot(1-log_spec,log_sens, xlab = "1-Specificity", ylab = "Sensitivity", type = "l", col = "red", lwd = 1
lines(1-rf_spec, rf_sens, col = "blue", lty = 2, lwd = 2)
legend("topleft", legend = c("GLM ROC", "RF ROC"), lty = c(1,2), col = c("red", "blue"), lwd = 2)
```



We see that both lines are basically equivalent to the chance line, which is not good! RF does do a bit better. *Not shown but before optimizing RF, RF actually did a little bit worse than GLM*

Variable Importance and Multicollinearity

Let's look at the importance of variables for each model and multicollinearity.

```
vif(logitmodel)
```

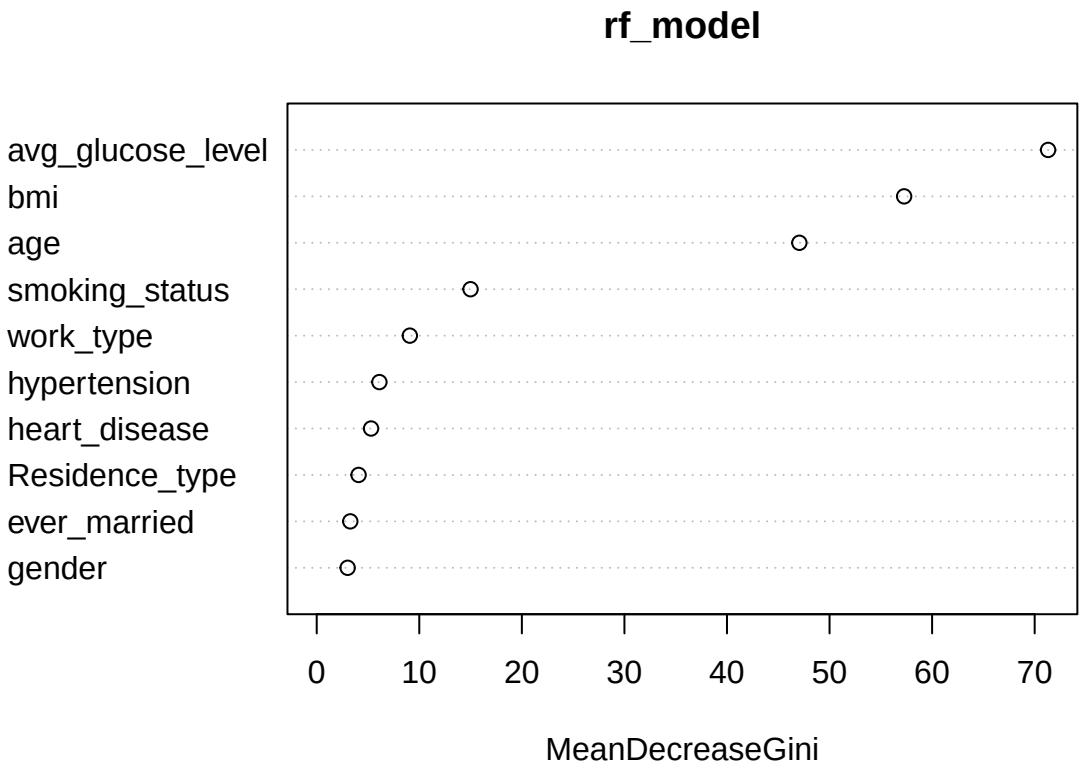
```
##                      GVIF Df GVIF^(1/(2*Df))
## gender           1.038062  1        1.018853
## age              1.237665  1        1.112504
## hypertension     1.059567  1        1.029353
## heart_disease    1.090178  1        1.044116
## ever_married     1.023673  1        1.011767
## work_type        1.078646  4        1.009508
## Residence_type   1.009350  1        1.004664
## avg_glucose_level 1.107348  1        1.052306
## bmi              1.097994  1        1.047852
## smoking_status   1.092557  3        1.014863
```

```
anova(logitmodel)
```

```
## Analysis of Deviance Table
##
```

```
## Model: binomial, link: logit
##
## Response: stroke
##
## Terms added sequentially (first to last)
##
##
##                   Df Deviance Resid. Df Resid. Dev
## NULL                                3434    1226.51
## gender             1    0.001      3433    1226.51
## age                1  234.617      3432     991.90
## hypertension       1   16.620      3431     975.27
## heart_disease      1    4.445      3430     970.83
## ever_married       1    0.635      3429     970.19
## work_type          4    3.437      3425     966.76
## Residence_type     1    0.273      3424     966.49
## avg_glucose_level  1    9.758      3423     956.73
## bmi                1    0.019      3422     956.71
## smoking_status     3    3.368      3419     953.34
```

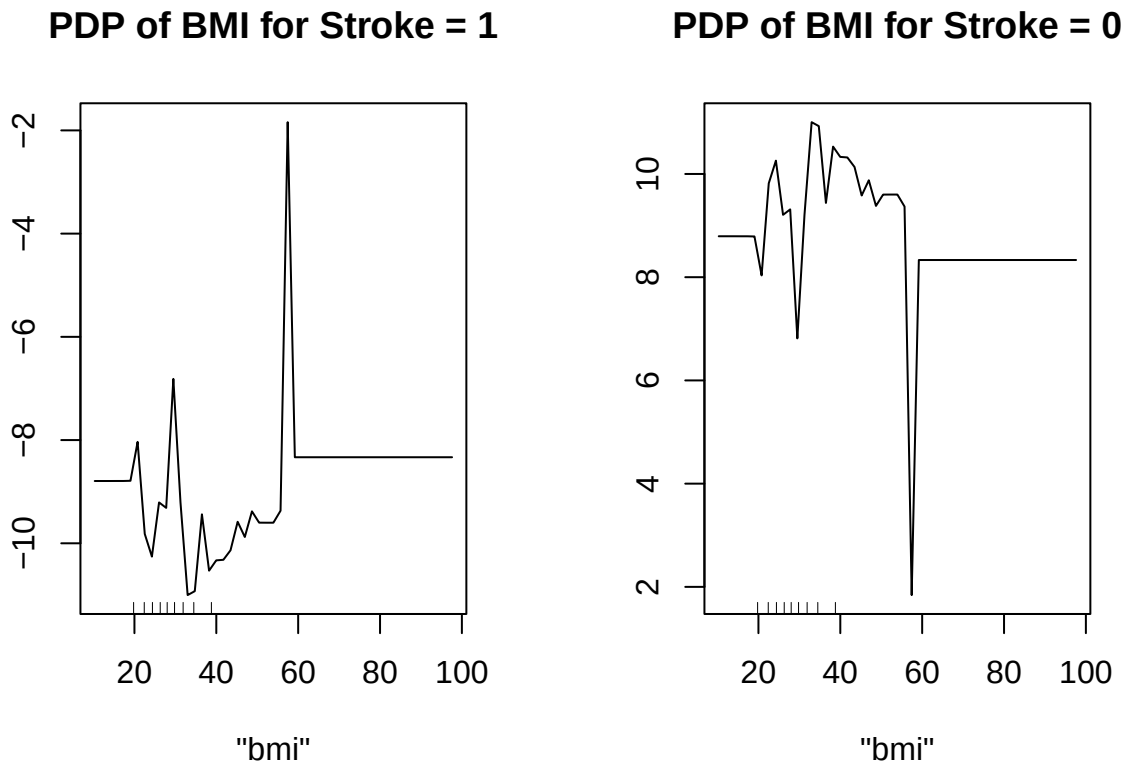
```
varImpPlot(rf_model, sort = T)
```



Average glucose level, BMI, and age have very high importance in the RF model, whereas, age, hypertension, and average glucose level are important in the GLM. Using deviance is a better gauge of variable importance for the GLM. There's also no multicollinearity issues.

Partial Dependency Plots

```
par(mfrow=c(1,2))
partialPlot(rf_model, train, 'bmi', which.class = '1',
            main = "PDP of BMI for Stroke = 1")
partialPlot(rf_model, train, 'bmi', which.class = '0',
            main = "PDP of BMI for Stroke = 0")
```

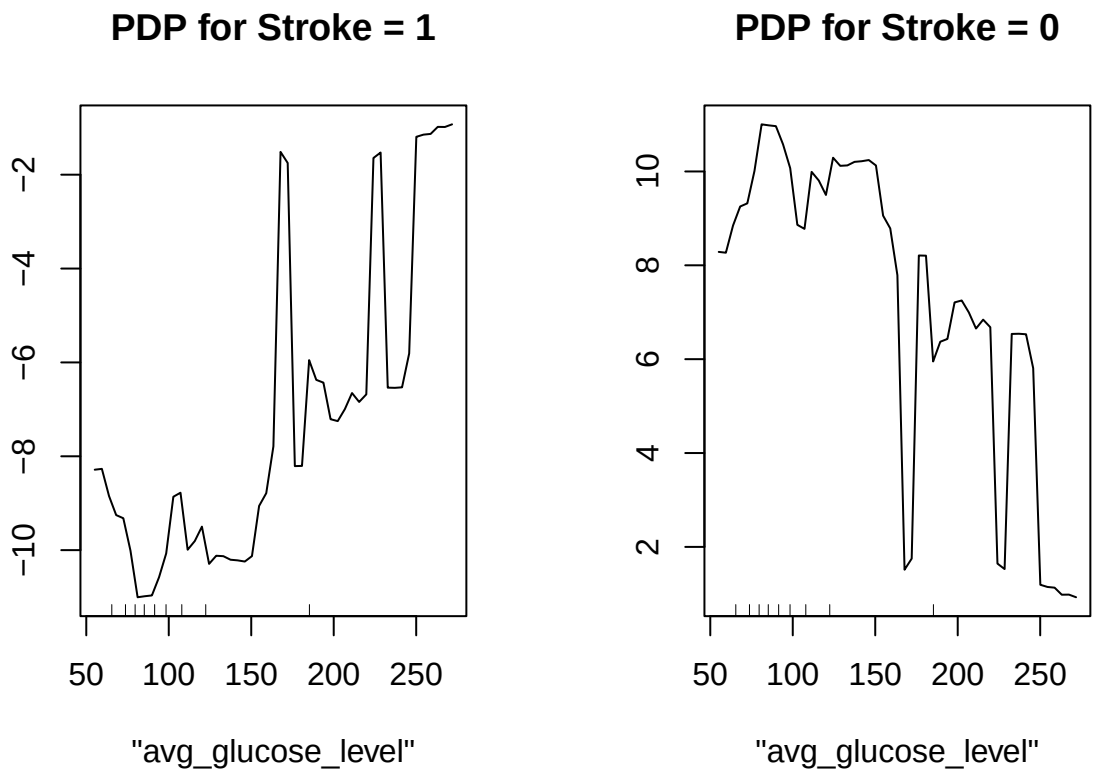


Partial Dependency Plot for BMI, the positive trend in stroke = 1 means that as BMI increases the model is more likely to predict the patient as a stroke patient.

```
par(mfrow=c(1,2))
partialPlot(rf_model, train, 'age', which.class = '1',
            main = "PDP of Age for Stroke = 1")
partialPlot(rf_model, train, 'age', which.class = '0',
            main = "PDP of Age for Stroke = 0")
```

Partial Dependency Plot for age, the positive trend in stroke = 1 means that as age increases the model is more likely to predict the patient as a stroke patient.

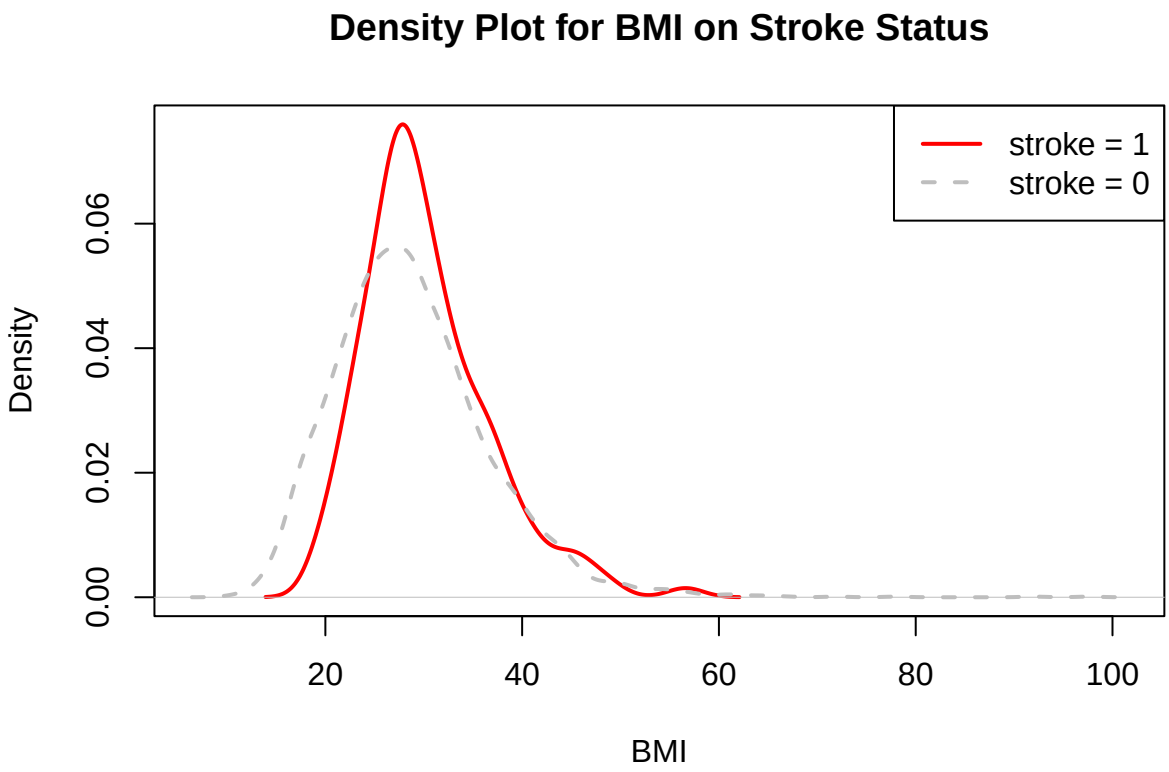
```
par(mfrow=c(1,2))
partialPlot(rf_model, train, 'avg_glucose_level', which.class = '1',
            main = "PDP for Stroke = 1")
partialPlot(rf_model, train, 'avg_glucose_level', which.class = '0',
            main = "PDP for Stroke = 0")
```



Partial Dependency Plot for average glucose level, the positive trend in stroke = 1 means that as average glucose level increases the model is more likely to predict the patient as a stroke patient.

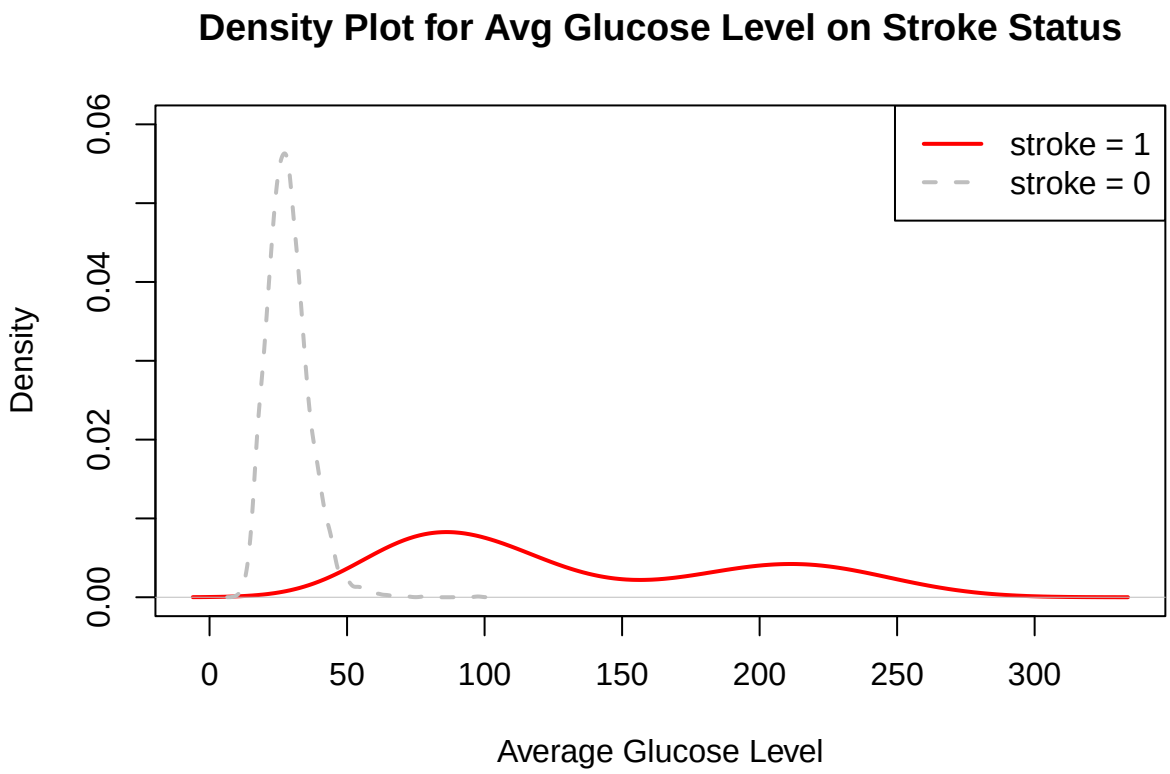
Density Plots

```
bmi_1 <- density(train$bmi[which(train$stroke=='1')])
bmi_0 <- density(train$bmi[which(train$stroke=='0')])
#par(mfrow=c(1,2))
plot(bmi_1, xlim=range(bmi_1$x,bmi_0$x), ylim=range(bmi_1$y,bmi_0$y), col = "red",
     main = "Density Plot for BMI on Stroke Status",
     xlab = "BMI", lwd=2)
lines(bmi_0, col = "grey", lwd= 2, lty = 2)
legend("topright", legend=c("stroke = 1", "stroke = 0"), lty = c(1,2),
       col = c("red", "grey"), lwd=2)
```



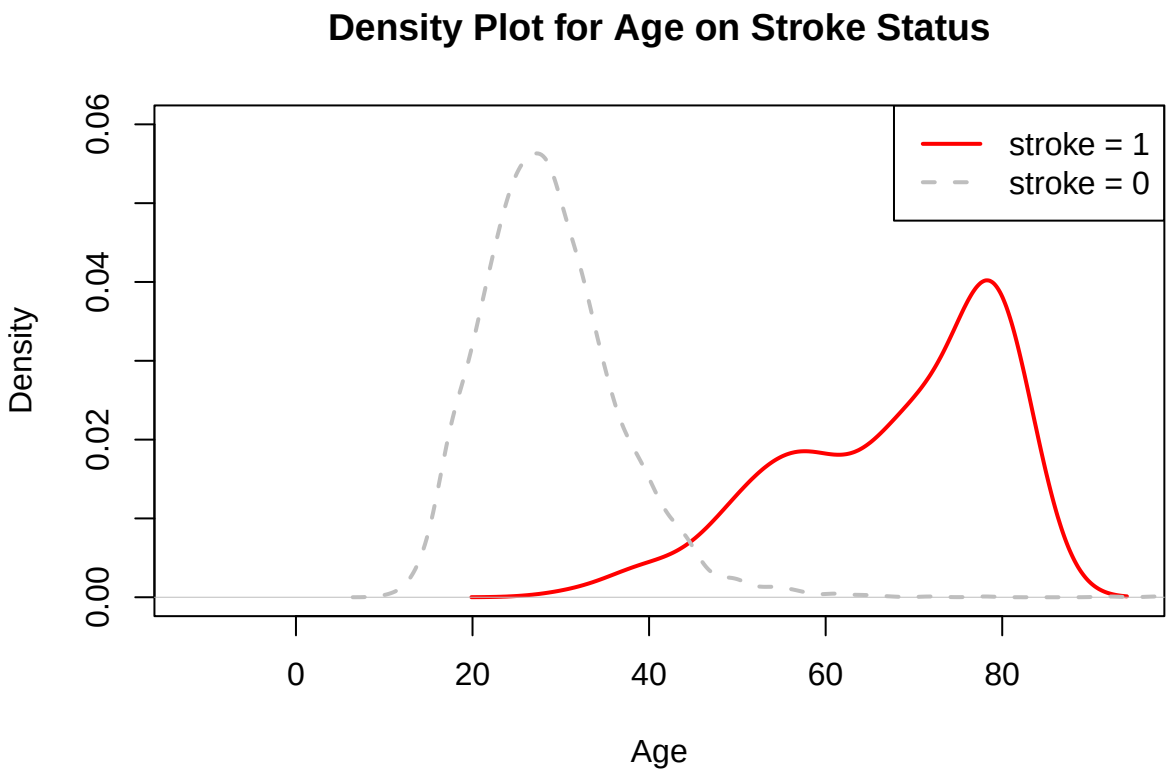
We can see that the BMI of stroke patients are a bit larger than non-stroke patients, but there's no clear threshold.

```
gluc_1 <- density(train$avg_glucose_level[which(train$stroke=='1')])
gluc_0 <- density(train$avg_glucose_level[which(train$stroke=='0')])
#par(mfrow=c(1,2))
plot(gluc_1, xlim=range(gluc_1$x,gluc_0$x), ylim=range(0,0.06), col = "red",
     main = "Density Plot for Avg Glucose Level on Stroke Status",
     xlab = "Average Glucose Level", lwd=2)
lines(bmi_0, col = "grey", lwd= 2, lty = 2)
legend("topright", legend=c("stroke = 1", "stroke = 0"), lty = c(1,2),
       col = c("red", "grey"), lwd=2)
```



We can see that the average glucose level is a lot lower for non-stroke patients with a clear threshold around 50.

```
age_1 <- density(train$age[which(train$stroke=='1')])
age_0 <- density(train$age[which(train$stroke=='0')])
#par(mfrow=c(1,2))
plot(age_1, xlim=range(age_1$x,age_0$x), ylim=range(0,0.06), col = "red",
     main = "Density Plot for Age on Stroke Status",
     xlab = "Age", lwd=2)
lines(bmi_0, col = "grey", lwd= 2, lty = 2)
legend("topright", legend=c("stroke = 1", "stroke = 0"), lty = c(1,2),
       col = c("red", "grey"), lwd=2)
```



Stroke patients tend to be older in age than non-stroke patient with a threshold around age 45 which can clearly be seen in the density plot.

Looking into MatchIt

After seeing the discrepancy between the misclassification rate and AUC score of both General Linear Model and RF model, we noticed that our dataset is severely imbalanced. There were only 209 stroke patients out of 4,908 patients which is 4.26%. To address this issue, we tried the nearest neighbor matching method using the MatchIt package in R. MatchIt is used to address issues when you have an unbalanced dataset by calculating propensity score (2). Propensity score is a method used to balance explanatory variables between treatment (stroke = 1) and control (stroke = 0) groups, reducing bias and enabling more valid causal inferences in observational studies. Propensity scores are calculated using General Linear Model to estimate the probability of being assigned the treatment based on observed explanatory variables. These scores are used for matching or weighting to create balanced treatment and control groups.

First, let's look at the initial imbalance of the data prior to matching which is why method = NULL. We set distance = "glm" for generalized linear model, which implements General Linear Model by default.

```
library(MatchIt)
match <- matchit(stroke ~ ., data = data,
                 method = NULL, distance = "glm")
summary(match)
```

```
##
## Call:
```

```
## matchit(formula = stroke ~ ., data = data, method = NULL, distance = "glm")
##
## Summary of Balance for All Data:
##                               Means Treated Means Control Std. Mean Diff.
## distance                             0.1377        0.0384          0.9596
## genderFemale                         0.5742        0.5910         -0.0340
## genderMale                           0.4258        0.4090          0.0340
## age                                 67.7129       41.7638          2.0922
## hypertension0                        0.7129        0.9168         -0.4506
## hypertension1                        0.2871        0.0832          0.4506
## heart_disease0                       0.8086        0.9568         -0.3767
## heart_disease1                       0.1914        0.0432          0.3767
## ever_marriedNo                       0.1100        0.3577         -0.7915
## ever_marriedYes                      0.8900        0.6423          0.7915
## work_typechildren                    0.0048        0.1426         -1.9969
## work_typeGovt_job                    0.1340        0.1281          0.0172
## work_typeNever_worked                0.0000        0.0047         -0.0701
## work_typePrivate                     0.6077        0.5710          0.0751
## work_typeSelf-employed               0.2536        0.1536          0.2297
## Residence_typeRural                  0.4785        0.4933         -0.0297
## Residence_typeUrban                  0.5215        0.5067          0.0297
## avg_glucose_level                  134.5714      103.9954          0.4895
## bmi                                 30.4713       28.8244          0.2602
## smoking_statusformerly smoked        0.2727        0.1658          0.2401
## smoking_statusnever smoked           0.4019        0.3763          0.0523
## smoking_statussmokes                 0.1866        0.1485          0.0977
## smoking_statusUnknown                0.1388        0.3094         -0.4937
##                               Var. Ratio eCDF Mean eCDF Max
## distance                          3.0748    0.3527   0.5549
## genderFemale                           .    0.0168   0.0168
## genderMale                             .    0.0168   0.0168
## age                               0.3102    0.2517   0.5304
## hypertension0                          .    0.2039   0.2039
## hypertension1                          .    0.2039   0.2039
## heart_disease0                         .    0.1482   0.1482
## heart_disease1                         .    0.1482   0.1482
## ever_marriedNo                         .    0.2477   0.2477
## ever_marriedYes                        .    0.2477   0.2477
## work_typechildren                      .    0.1378   0.1378
## work_typeGovt_job                      .    0.0059   0.0059
## work_typeNever_worked                  .    0.0047   0.0047
## work_typePrivate                       .    0.0367   0.0367
## work_typeSelf-employed                 .    0.0999   0.0999
## Residence_typeRural                    .    0.0148   0.0148
## Residence_typeUrban                    .    0.0148   0.0148
## avg_glucose_level                 2.1102    0.1290   0.2532
## bmi                               0.6405    0.0442   0.1697
## smoking_statusformerly smoked          .    0.1069   0.1069
## smoking_statusnever smoked             .    0.0257   0.0257
## smoking_statussmokes                   .    0.0381   0.0381
## smoking_statusUnknown                  .    0.1707   0.1707
##
## Sample Sizes:
##           Control Treated
```

```
## All          4699     209
## Matched      4699     209
## Unmatched       0       0
## Discarded       0       0
```

Values of standardized mean differences and eCDF statistics close to zero and values of variance ratios close to one indicate good balance, and here many of them are far from their ideal values.

Nearest Neighbor Matching Method

Now, we begin by briefly demonstrating 1:1 nearest neighbor (NN) matching by the propensity score, which is appropriate for estimating the average treatment effect in the treated (ATT). One by one, each treated unit is paired with a control unit that has the closest propensity score to it. Any remaining control units are left unmatched and excluded from further analysis.

```
match_nn <- matchit(stroke ~ ., data = data,
                 method = "nearest", distance = "glm")
match_nn
```

```
## A matchit object
##  - method: 1:1 nearest neighbor matching without replacement
##  - distance: Propensity score
##              - estimated with logistic regression
##  - number of obs.: 4908 (original), 418 (matched)
##  - target estimand: ATT
##  - covariates: gender, age, hypertension, heart_disease, ever_married, work_type, Residence_type, avg
```

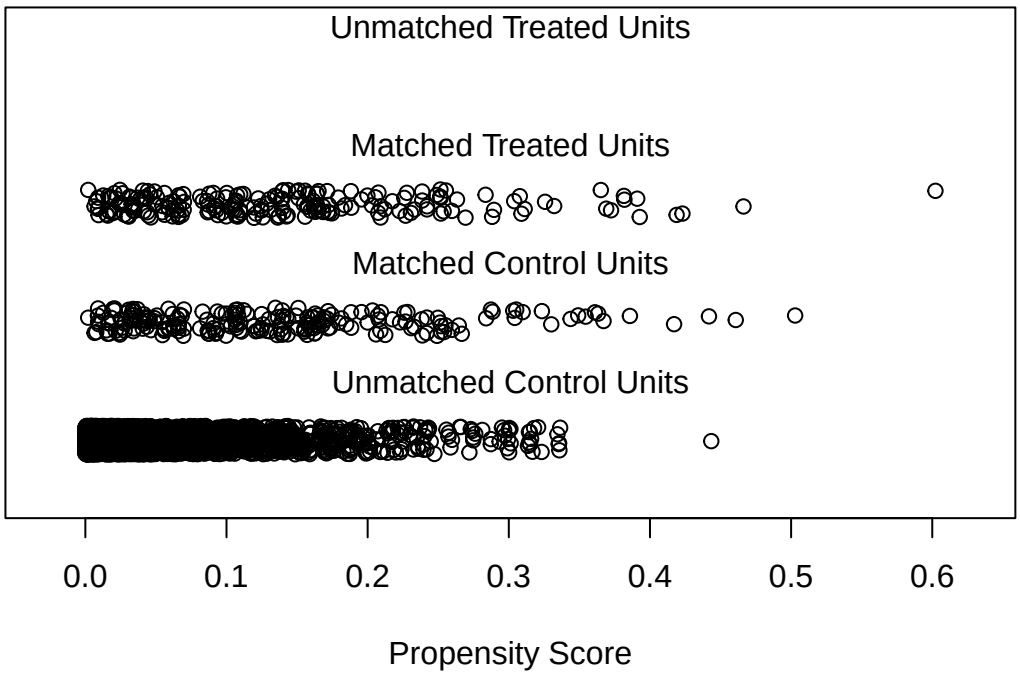
```
summary(match_nn, un = F)
```

```
##
## Call:
## matchit(formula = stroke ~ ., data = data, method = "nearest",
##     distance = "glm")
##
## Summary of Balance for Matched Data:
##                               Means Treated Means Control Std. Mean Diff.
## distance                             0.1377        0.1365          0.0111
## genderFemale                         0.5742        0.5407          0.0677
## genderMale                           0.4258        0.4593         -0.0677
## age                                 67.7129       67.6938          0.0015
## hypertension0                        0.7129        0.6890          0.0529
## hypertension1                        0.2871        0.3110         -0.0529
## heart_disease0                       0.8086        0.8660         -0.1460
## heart_disease1                       0.1914        0.1340          0.1460
## ever_marriedNo                       0.1100        0.0957          0.0459
## ever_marriedYes                      0.8900        0.9043         -0.0459
## work_typechildren                    0.0048        0.0000          0.0693
## work_typeGovt_job                    0.1340        0.1148          0.0562
## work_typeNever_worked                0.0000        0.0000          0.0000
## work_typePrivate                     0.6077        0.6268         -0.0392
## work_typeSelf-employed               0.2536        0.2584         -0.0110
## Residence_typeRural                  0.4785        0.4498          0.0575
```

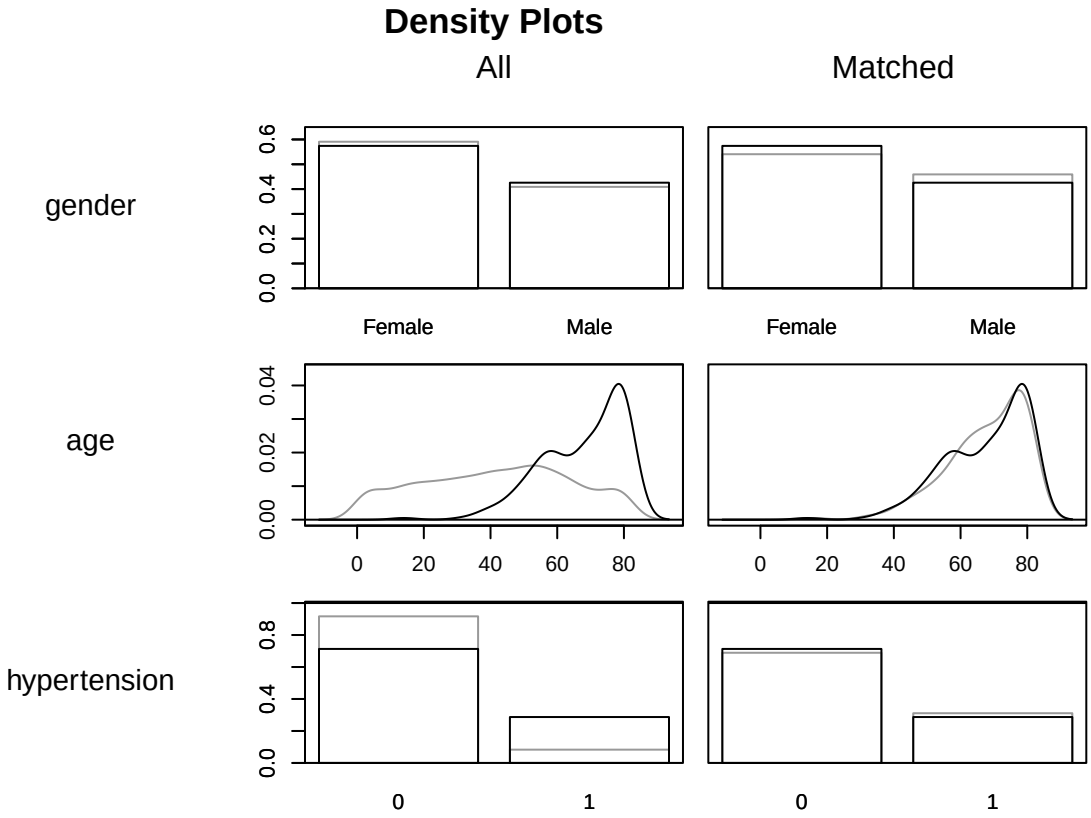
```
## Residence_typeUrban                  0.5215        0.5502         -0.0575
## avg_glucose_level                  134.5714      136.8124         -0.0359
## bmi                                 30.4713       30.8187         -0.0549
## smoking_statusformerly smoked        0.2727        0.2823         -0.0215
## smoking_statusnever smoked           0.4019        0.4354         -0.0683
## smoking_statussmokes                 0.1866        0.1627          0.0614
## smoking_statusUnknown                0.1388        0.1196          0.0554
##                               Var. Ratio eCDF Mean eCDF Max Std. Pair Dist.
## distance                          1.0679    0.0003   0.0239          0.0143
## genderFemale                           .    0.0335   0.0335          0.9967
## genderMale                             .    0.0335   0.0335          0.9967
## age                               1.1035    0.0095   0.0670          0.4753
## hypertension0                          .    0.0239   0.0239          0.7932
## hypertension1                          .    0.0239   0.0239          0.7932
## heart_disease0                         .    0.0574   0.0574          0.6081
## heart_disease1                         .    0.0574   0.0574          0.6081
## ever_marriedNo                         .    0.0144   0.0144          0.5963
## ever_marriedYes                        .    0.0144   0.0144          0.5963
## work_typechildren                      .    0.0048   0.0048          0.0693
## work_typeGovt_job                      .    0.0191   0.0191          0.6462
## work_typeNever_worked                  .    0.0000   0.0000          0.0000
## work_typePrivate                       .    0.0191   0.0191          0.9211
## work_typeSelf-employed                 .    0.0048   0.0048          0.8688
## Residence_typeRural                    .    0.0287   0.0287          1.1685
## Residence_typeUrban                    .    0.0287   0.0287          1.1685
## avg_glucose_level                 0.9502    0.0164   0.0670          0.9260
## bmi                               0.8899    0.0131   0.0670          1.0633
## smoking_statusformerly smoked          .    0.0096   0.0096          0.7950
## smoking_statusnever smoked             .    0.0335   0.0335          0.9076
## smoking_statussmokes                   .    0.0239   0.0239          0.7246
## smoking_statusUnknown                  .    0.0191   0.0191          0.4983
##
## Sample Sizes:
##           Control Treated
## All          4699     209
## Matched       209     209
## Unmatched    4490       0
## Discarded       0       0
```

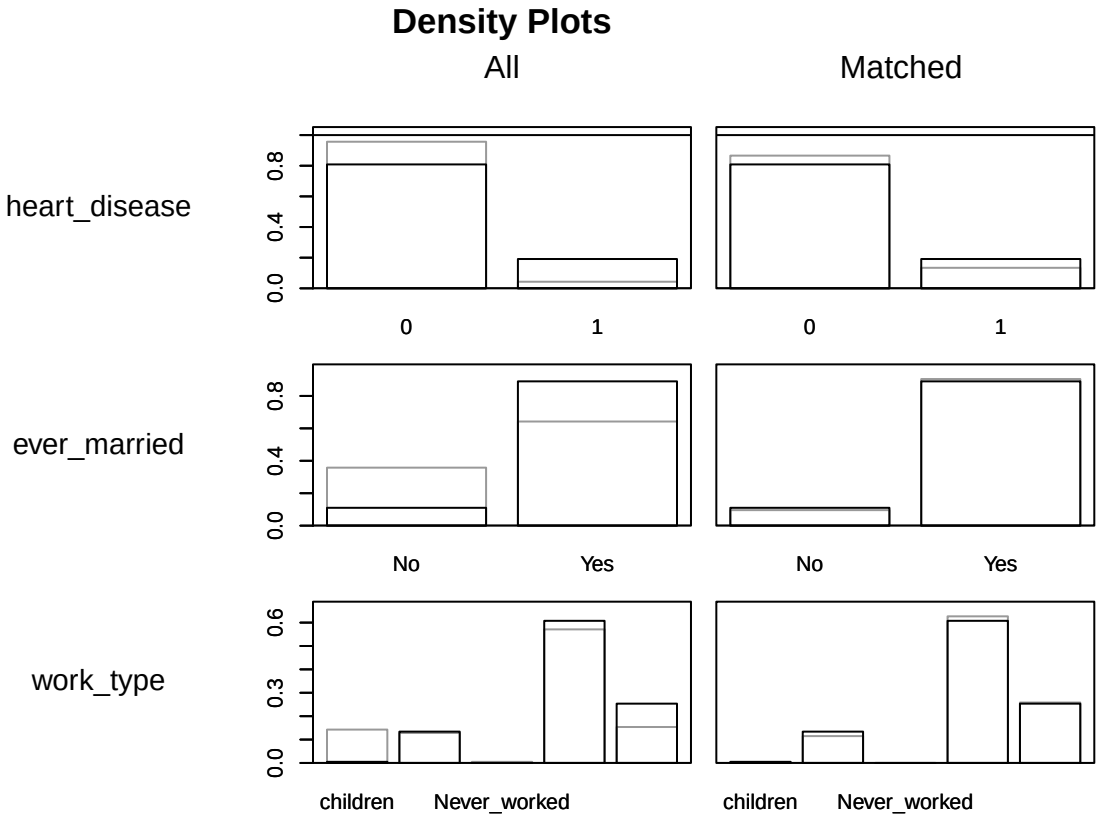
Visualizing the Propensity Score

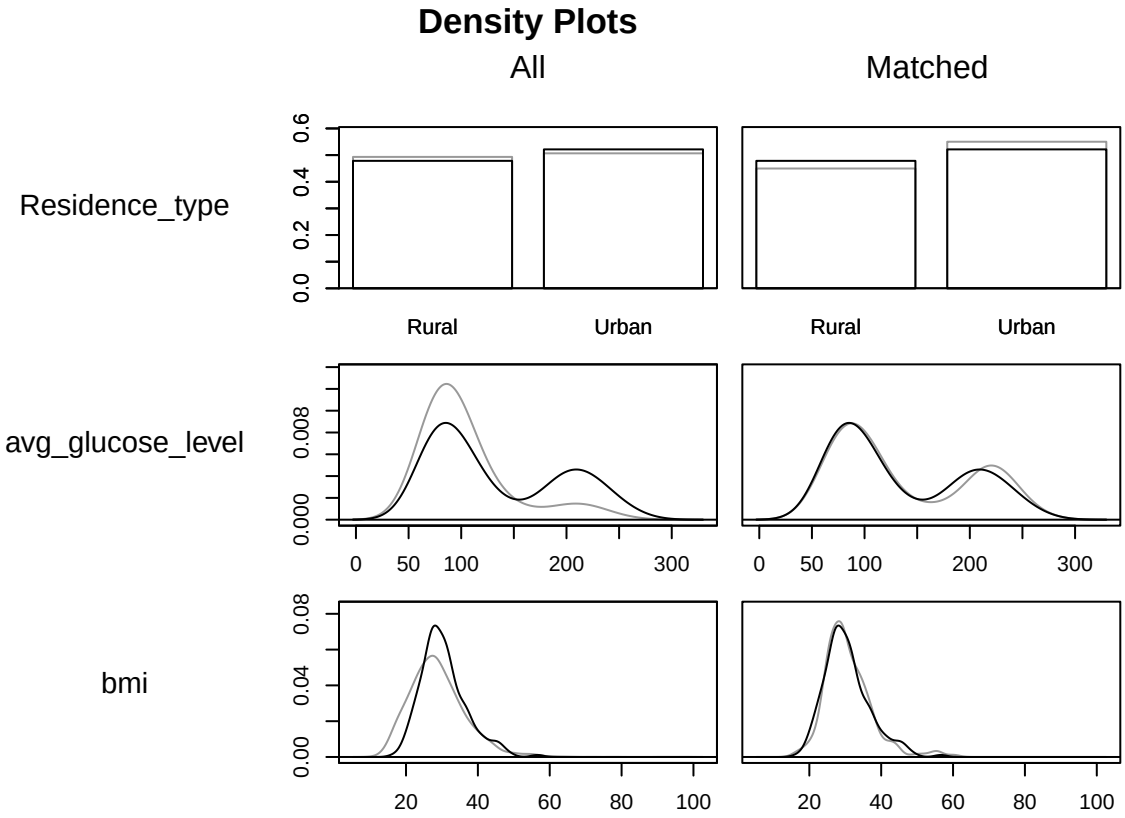
```r
plot(match_nn, type = "jitter", interactive = FALSE)
```




```
plot(match_nn, type = "density", interactive = FALSE,
     which.xs = ~ gender+age+hypertension+heart_disease+ever_married+work_type+Residence_type+avg_glucos
```


Density Plots
All
Matched
gender
age
hypertension
0.0
0.2
0.4
0.6
Female
Male
0.00
0.02
0.04
0
20
40
60
80
0.0
0.4
0.8
0
1


Density Plots
All
Matched
heart_disease
ever_married
work_type
0.0
0.4
0.8
0.3
0.6
0
1
No
Yes
children
Never_worked


Density Plots
All
Matched
Residence_type
avg_glucose_level
bmi
0.0
0.2
0.4
0.6
Rural
Urban
0.000
0.008
0
50
100
200
300
0.00
0.04
0.08
20
40
60
80
100

Based these plots, I believe nearest neighbor propensity score matching is sufficient in balancing the data.

Modeling with the Matched Data

Because matching adds three extra columns (weights, distance, subclass) to the data when computing propensity score, those columns were removed in the regression and prediction analysis.

```
data_match_nn <- match.data(match_nn)
data_match_nn <- data_match_nn[, c("stroke", setdiff(names(data_match_nn), "stroke"))]

n=3
nn = nrow(data_match_nn)
nn_log_misclass_rate <- vector(mode = "numeric", length = n)
nn_rf_misclass_rate <- vector(mode = "numeric", length = n)
nn_log_auc <- vector(mode = "numeric", length = n)
nn_rf_auc <- vector(mode = "numeric", length = n)
for (i in 1:n){
  set.seed(i)
index_nn <- sample(1:nn, size = nn*.7, replace = F)
  nn_train_x <- data_match_nn[index_nn, 2:11]
  nn_test_x <- data_match_nn[-index_nn, 2:11]
  nn_train_y <- data_match_nn[index_nn, 1]
  nn_test_y <- data_match_nn[-index_nn, 1]
  nn_train <- data_match_nn[index_nn, 1:11]
  nn_test <-data_match_nn[-index_nn, 1:11]
```

```
nn_log <- glm(stroke ~., family = binomial(link = logit), data = nn_train)
summary(nn_log)

nn_pred_logitmod <- predict(nn_log, newdata = nn_test, type = "response")
  nn_log_predicted_class <- ifelse(nn_pred_logitmod > 0.5, 1, 0)
  nn_log_predicted_class <- factor(nn_log_predicted_class, levels = c(0, 1))

  nn_log_error <- table(actual = nn_test_y, predicted = nn_log_predicted_class)

  nn_log_misclass_rate[i] <- 1-sum(diag(nn_log_error))/sum(nn_log_error)
  nn_log_auc[i] <- auc(roc(nn_log_predicted_class, nn_test_y))

   nn_rf_model <- randomForest(stroke~.,data = nn_train, xtest = nn_test_x,
                    ytest=nn_test_y, mtry=optimal.m, nodesize=optimal.depth,
                    ntree=optimal.ntrees, keep.forest = TRUE)
  nn_rf_predicted_class <- nn_rf_model$test$predicted
  nn_rf_error <- table(actual = nn_test_y, predicted = nn_rf_predicted_class)
  nn_rf_misclass_rate[i] <- 1-sum(diag(nn_rf_error))/sum(nn_rf_error)

  nn_rf_auc[i] <- auc(roc(nn_rf_predicted_class,nn_test_y))
}
```

Summary Results Comparing Predictions of GLM and RF After Matching

```
nn_results_misclass = data.frame(Log_Error = nn_log_misclass_rate, RF_Error = nn_rf_misclass_rate)
nn_results_auc = data.frame(Log_AUC = nn_log_auc, RF_AUC = nn_rf_auc)
nn_summary_results_misclass=describe(nn_results_misclass)
nn_summary_results_auc = describe(nn_results_auc)
nn_results_misclass
```

```
##   Log_Error  RF_Error
## 1 0.5555556 0.5238095
## 2 0.5396825 0.5238095
## 3 0.5079365 0.5238095
```

```
nn_results_auc
```

```
##     Log_AUC    RF_AUC
## 1 0.4561404 0.4759725
## 2 0.4598485 0.4750000
## 3 0.4920635 0.4761905
```

```
nn_summary_results_misclass
```

```
##           vars n mean   sd median trimmed  mad  min  max range  skew kurtosis
## Log_Error    1 3 0.53 0.02   0.54    0.53 0.02 0.51 0.56  0.05 -0.21    -2.33
## RF_Error     2 3 0.52 0.00   0.52    0.52 0.00 0.52 0.52  0.00   NaN      NaN
##              se
## Log_Error 0.01
## RF_Error  0.00
```

```
nn_summary_results_auc
```

```
##         vars n mean   sd median trimmed  mad  min  max range  skew kurtosis
## Log_AUC    1 3 0.47 0.02   0.46    0.47 0.01 0.46 0.49  0.04  0.37    -2.33
## RF_AUC     2 3 0.48 0.00   0.48    0.48 0.00 0.48 0.48  0.00 -0.33    -2.33
##           se
## Log_AUC 0.01
## RF_AUC  0.00
```

Nearest neighbor method matched 418 observations. General Linear Model and RF was performed on the matched data. General Linear Model performance were misclassification score = 0.5343915 and AUC score = 0.4693508. RF performance were misclassification score = 0.5238095 and AUC score = 0.475721. Both models performed very poorly.

Analyzing Based on Prevalence

In the United States, each year 795,000 suffer from a stroke (3). If we divide that number by the US population, there is about a 0.24% chance of US citizens getting a stroke per year (4). The test set contains 1473 observations. If we multiply the number of rows by the prevalence, it will give us the number of stroke GLM and RF should predict the test set to have which is about 3-4 stroke patients.

```
log_sorted_indices <- order(-pred_logitmod)
us_stroke = 795000/332915073
prevelance = round((nrow(test))*us_stroke)
log_top_prob <- pred_logitmod[log_sorted_indices[1:prevelance]]
print(paste("The highest probability of stroke patient using GLM is", paste(round(log_top_prob, digits =
```

```
## [1] "The highest probability of stroke patient using GLM is 0.47 and 0.464 and 0.431 and 0.417"
```

```
rf_prob <- data.frame(rf_model$test$votes)
rf_prob <- rf_prob$X1
rf_sorted_indices <- order(-rf_prob)
rf_top_prob <- rf_prob[rf_sorted_indices[1:prevelance]]
print(paste("The highest probability of stroke patient using RF is", paste(round(rf_top_prob, digits = 3
```

```
## [1] "The highest probability of stroke patient using RF is 0.765 and 0.665 and 0.64 and 0.585"
```

```
rf_stroke_pred <- sum(rf_top_prob > 0.5)
rf_prev_comparison = c(rf_stroke_pred,prevelance)
log_stroke_pred <- sum(log_top_prob > 0.5)
log_prev_comparison = c(log_stroke_pred, prevelance)
print(paste("The number of predicted stroke using RF vs prevalence of stroke in test set is", paste(rf_
```

```
## [1] "The number of predicted stroke using RF vs prevalence of stroke in test set is 4 vs 4"
```

```
print(paste("The number of predicted stroke using GLM vs prevalence of stroke in test set is", paste(log
```

```
## [1] "The number of predicted stroke using GLM vs prevalence of stroke in test set is 0 vs 4"
```

RF predicted 4 stroke patient and GLM predicted none when hypothetically there should have been 3-4. The optimized RF performed really well in terms of prevalence; however the GLM did poorly. *Not shown but before optimizing RF, RF only predicted 1 stroke patient*

Model Performance Based on Variable Reduction for GLM

Using stepAIC to create the reduced model, the reduced model was analyzed based on prediction performance.

```
reduced_log <- stepAIC(logitmodel, direction = "both")
```

```
## Start:  AIC=985.34
## stroke ~ gender + age + hypertension + heart_disease + ever_married +
##     work_type + Residence_type + avg_glucose_level + bmi + smoking_status
##
##                     Df Deviance     AIC
## - work_type          4   956.52  980.52
## - smoking_status     3   956.71  982.71
## - bmi                1   953.35  983.35
## - Residence_type     1   953.49  983.49
## - ever_married       1   953.55  983.55
## - gender             1   953.56  983.56
## <none>                   953.34  985.34
## - heart_disease      1   955.79  985.79
## - avg_glucose_level  1   962.58  992.58
## - hypertension       1   965.14  995.14
## - age                1  1056.50 1086.50
##
## Step:  AIC=980.52
## stroke ~ gender + age + hypertension + heart_disease + ever_married +
##     Residence_type + avg_glucose_level + bmi + smoking_status
##
##                     Df Deviance     AIC
## - smoking_status     3   960.11  978.11
## - bmi                1   956.52  978.52
## - Residence_type     1   956.64  978.64
## - gender             1   956.74  978.74
## - ever_married       1   956.92  978.92
## <none>                   956.52  980.52
## - heart_disease      1   959.29  981.29
## + work_type          4   953.34  985.34
## - avg_glucose_level  1   965.73  987.73
## - hypertension       1   968.14  990.14
## - age                1  1068.00 1090.00
##
## Step:  AIC=978.11
## stroke ~ gender + age + hypertension + heart_disease + ever_married +
##     Residence_type + avg_glucose_level + bmi
##
##                     Df Deviance     AIC
## - bmi                1   960.12  976.12
## - Residence_type     1   960.26  976.26
## - gender             1   960.31  976.31
## - ever_married       1   960.64  976.64
## <none>                   960.11  978.11
## - heart_disease      1   963.33  979.33
## + smoking_status     3   956.52  980.52
```

```
## + work_type          4   956.71  982.71
## - avg_glucose_level  1   969.71  985.71
## - hypertension       1   973.05  989.05
## - age                1  1073.31 1089.31
##
## Step:  AIC=976.12
## stroke ~ gender + age + hypertension + heart_disease + ever_married +
##     Residence_type + avg_glucose_level
##
##                     Df Deviance     AIC
## - Residence_type     1   960.27  974.27
## - gender             1   960.31  974.31
## - ever_married       1   960.64  974.64
## <none>                   960.12  976.12
## - heart_disease      1   963.35  977.35
## + bmi                1   960.11  978.11
## + smoking_status     3   956.52  978.52
## + work_type          4   956.73  980.73
## - avg_glucose_level  1   969.95  983.95
## - hypertension       1   973.11  987.11
## - age                1  1075.16 1089.16
##
## Step:  AIC=974.27
## stroke ~ gender + age + hypertension + heart_disease + ever_married +
##     avg_glucose_level
##
##                     Df Deviance     AIC
## - gender             1   960.46  972.46
## - ever_married       1   960.75  972.75
## <none>                   960.27  974.27
## - heart_disease      1   963.47  975.47
## + Residence_type     1   960.12  976.12
## + bmi                1   960.26  976.26
## + smoking_status     3   956.64  976.64
## + work_type          4   956.90  978.90
## - avg_glucose_level  1   970.19  982.19
## - hypertension       1   973.26  985.26
## - age                1  1075.49 1087.49
##
## Step:  AIC=972.46
## stroke ~ age + hypertension + heart_disease + ever_married +
##     avg_glucose_level
##
##                     Df Deviance     AIC
## - ever_married       1   960.94  970.94
## <none>                   960.46  972.46
## - heart_disease      1   963.54  973.54
## + gender             1   960.27  974.27
## + Residence_type     1   960.31  974.31
## + bmi                1   960.46  974.46
## + smoking_status     3   956.88  974.88
## + work_type          4   957.09  977.09
## - avg_glucose_level  1   970.23  980.23
## - hypertension       1   973.52  983.52
```

```
## - age                1  1076.02 1086.02
##
## Step:  AIC=970.94
## stroke ~ age + hypertension + heart_disease + avg_glucose_level
##
##                     Df Deviance     AIC
## <none>                   960.94  970.94
## - heart_disease      1   963.95  971.95
## + ever_married       1   960.46  972.46
## + gender             1   960.75  972.75
## + Residence_type     1   960.82  972.82
## + bmi                1   960.94  972.94
## + smoking_status     3   957.25  973.25
## + work_type          4   957.32  975.32
## - avg_glucose_level  1   970.86  978.86
## - hypertension       1   974.03  982.03
## - age                1  1107.06 1115.06
```

```
r_log_pred <- predict(reduced_log, newdata = test, type = "response")
r_log_pred_class <- ifelse(r_log_pred > 0.5, 1,0)
length(r_log_pred_class[r_log_pred_class == "1"])
```

```
## [1] 2
```

```
r_log_auc <- auc(roc(r_log_pred_class,test_y))
```

Because the reduced GLM model predicted 2 stroke patients and the AUC score is 0.4992923, the reduced GLM also doesn't really perform well but better than the full model in terms of prevalence.

Conclusion

Random Forest and General Linear Model are two of the more common and favorable models when dealing with binary classification predictions. In this analysis, we saw how poorly both of them performed on this dataset based on AUC scores. This makes us believe that because the prevalence of having a stroke is so rare, a much larger dataset or other/more explanatory variables must be used in order for RF or GLM to spot the tiniest differences between stroke patients and non-stroke patients.
When using larger or more complex datasets, RF is most likely the better option because of the many ways to optimize RF as we saw the difference between the optimized RF performance and the un-optimized. However, on smaller to moderately sized datasets, GLM is easier to interpret and likely to perform just as well.

References

1. Fedesoriano. "Stroke Prediction Dataset." Kaggle, 26 Jan. 2021, www.kaggle.com/datasets/fedesoriano/stroke-prediction-dataset/data.
2. Greifer, Noah. Matchit: Getting Started, 12 Oct. 2023, cran.r-project.org/web/packages/MatchIt/vignettes/MatchIt.html.
3. Jose Vega MD, PhD. "Interesting and Surprising Facts and Statistics about Stroke." Verywell Health, Verywell Health, 6 Apr. 2022, www.verywellhealth.com/facts-and-statistics-about-stroke-3146382.

4. “U.S. Population 1950-2023.” MacroTrends, www.macrotrends.net/countries/USA/united-states/population#:~:text=The%20current%20population%20of%20U.S.%20in%202021%20is,2019%20was%20329%2C064%2C917%2C%20a%200.6%25%20increase%20from%202018. Accessed 22 Oct. 2023.