Diabetes Logistic Regression

Kaia Lindberg

12/5/2021

Set up

```
# Import libraries
library(tidyverse)
library(ROCR)
```

```
# Load data
Data <- read.csv("diabetes2.csv", header=T)
print(head(Data))
```

```
##   Pregnancies Glucose BloodPressure SkinThickness Insulin  BMI
## 1           6     148            72            35       0 33.6
## 2           1      85            66            29       0 26.6
## 3           8     183            64             0       0 23.3
## 4           1      89            66            23      94 28.1
## 5           0     137            40            35     168 43.1
## 6           5     116            74             0       0 25.6
##   DiabetesPedigreeFunction Age Outcome
## 1                    0.627  50       1
## 2                    0.351  31       0
## 3                    0.672  32       1
## 4                    0.167  21       0
## 5                    2.288  33       1
## 6                    0.201  30       0
```

The data has 768 rows and 9 columns. One of these columns (Outcome) will be our response variable for our logistic regression and the others are potential predictors.

```
# Check dimensions of data
print(dim(Data)) # 768 rows and 9 columns
```

```
## [1] 768   9
```

```
# Display names of all columns
print(colnames(Data))
```

```
## [1] "Pregnancies"              "Glucose"
## [3] "BloodPressure"            "SkinThickness"
## [5] "Insulin"                  "BMI"
## [7] "DiabetesPedigreeFunction" "Age"
## [9] "Outcome"
```

Before we go any further with our analysis we will split our data into a training and test set. We've chosen a random seed of "123" for reproducibility. We will do all further analysis, visualization, and model building using our training data and then use our test data to evaluate our model's performance on unseen data.

```
# Randomly split data into two halves
set.seed(123) # For reproducability
sample<-sample.int(nrow(Data), floor(.50*nrow(Data)), replace = F)
train<-Data[sample, ] # Training data
test<-Data[-sample, ] # Test data
```

Data Cleaning

```
# Check for missing values in each column
colSums(is.na(train))
```

```
##              Pregnancies                  Glucose            BloodPressure 
##                        0                        0                        0 
##            SkinThickness                  Insulin                      BMI 
##                        0                        0                        0 
## DiabetesPedigreeFunction                      Age                  Outcome 
##                        0                        0                        0
```

It appears that there are no missing values.

```
# Summary of columns
print(summary(train))
```

```
##   Pregnancies        Glucose      BloodPressure    SkinThickness  
##  Min.   : 0.000   Min.   :  0.0   Min.   :  0.00   Min.   : 0.0  
##  1st Qu.: 1.000   1st Qu.:100.0   1st Qu.: 62.00   1st Qu.: 0.0  
##  Median : 3.000   Median :119.0   Median : 72.00   Median :23.0  
##  Mean   : 3.802   Mean   :120.9   Mean   : 68.92   Mean   :20.8  
##  3rd Qu.: 6.000   3rd Qu.:139.0   3rd Qu.: 80.00   3rd Qu.:33.0  
##  Max.   :17.000   Max.   :199.0   Max.   :110.00   Max.   :99.0  
##     Insulin            BMI        DiabetesPedigreeFunction      Age       
##  Min.   :  0.00   Min.   : 0.00   Min.   :0.0890           Min.   :21.00  
##  1st Qu.:  0.00   1st Qu.:27.40   1st Qu.:0.2400           1st Qu.:24.00  
##  Median : 40.00   Median :31.80   Median :0.3730           Median :29.00  
##  Mean   : 80.05   Mean   :32.09   Mean   :0.4734           Mean   :33.41  
##  3rd Qu.:130.00   3rd Qu.:36.73   3rd Qu.:0.6220           3rd Qu.:41.00  
##  Max.   :846.00   Max.   :67.10   Max.   :2.3290           Max.   :70.00  
##     Outcome     
##  Min.   :0.000  
##  1st Qu.:0.000  
##  Median :0.000  
##  Mean   :0.349  
##  3rd Qu.:1.000  
##  Max.   :1.000
```

This summary shows us the distribution of each variable. For example, we can see that the patients in the dataset range from age 21 to 70 with a median of 29 and mean of 33.4. Since we mean age is greater than the median age we can tell that the age distribution is skewed right. From this summary of the data we also observe some non-sensical values. For example some participants have a recorded BMI of 0 or an insulin value of 0, which cannot occur logically. Although there were no missing values above, we suspect that 0 was zero was recorded in place of missing data.The only variables for which a zero makes sense are pregnancies and outcome. For any other variable we'll fill zero with unknown and then evaluate the best way to handle unknowns (i.e. replace unknown with mean). We will do this for both the train and test sets even though we won't be using the test data until later.

```
# List of columns where zero is non-sensical (i.e. zero indicates unknown)
# All predictor columns other than pregnancies
zero_unknown_cols <-c("Glucose", "BloodPressure", "SkinThickness",
                      "Insulin", "BMI", "DiabetesPedigreeFunction",
                      "Age")
train[,zero_unknown_cols] <- replace(train[,zero_unknown_cols], train[,zero_unknown_cols]==0,NA)
test[,zero_unknown_cols] <- replace(test[,zero_unknown_cols], test[,zero_unknown_cols]==0,NA)
```

```
# Check for missing values in each column again
round(colSums(is.na(train))/dim(train)[1],2)
```

```
##              Pregnancies                  Glucose            BloodPressure
##                     0.00                     0.01                     0.05
##            SkinThickness                  Insulin                      BMI
##                     0.29                     0.48                     0.01
## DiabetesPedigreeFunction                      Age                  Outcome
##                     0.00                     0.00                     0.00
```

Now we can see that a large portion of some columns have missing data, especially Insulin with 48% missing and SkinThickness with 29%. Now that we've identified the missing values we need to handle them before we continue with our analysis. For now I am going to remove the variable with over 25% missing because I don't think they would be reliable predictors with so much missing data.

```
# Remove variables with high percent missing
train <- select(train, -c('SkinThickness', 'Insulin'))
```

For the remaining columns I will impute the missing values with the median for that column.

```
# For the remaining missing values (<5% in any column) I'll impute with the median
fill_missing_cols <-c("Glucose", "BloodPressure", "BMI",
                      "DiabetesPedigreeFunction", "Age")
for(i in fill_missing_cols) {
  train[ , i][is.na(train[ , i])] <- median(train[ , i], na.rm=TRUE)
}

# Do the same for the test data
for(i in fill_missing_cols) {
  test[ , i][is.na(test[ , i])] <- median(test[ , i], na.rm=TRUE)
}
```

Our outcome variable (Diabetes) is currently numeric as it is labeled as 0/1. We'll convert this to a factor and label the outcomes so that R treats it as a categorical response.

```
# Create diabetes factor column with labels from outcome column
train$Outcome<-factor(train$Outcome)
levels(train$Outcome) <- c("No", "Yes")

# For the test data too
test$Outcome<-factor(test$Outcome)
levels(test$Outcome) <- c("No", "Yes")
```

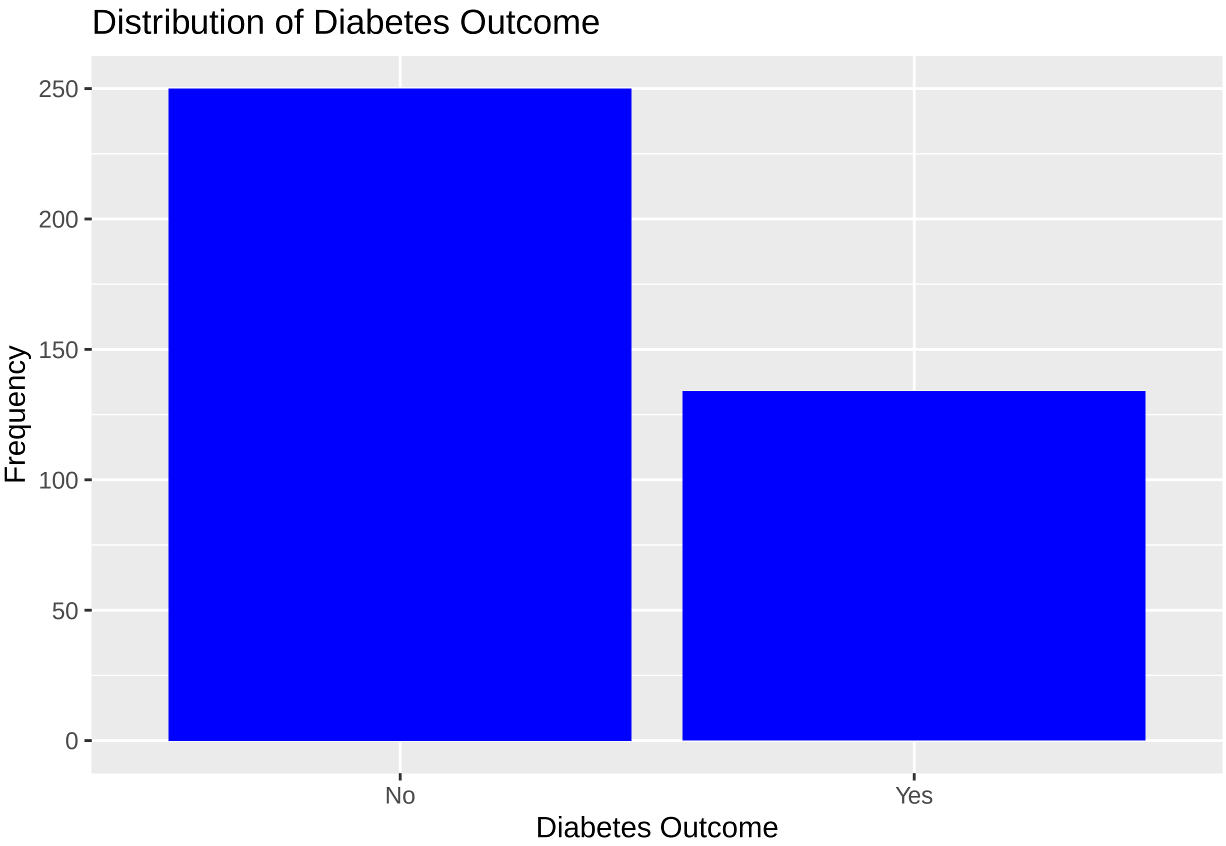
Analysis and Visualization

Distribution of Reponse Variable

```
# Calculate proportion of patients with diabetes
outcome_prop <- round(prop.table(table(train$Outcome)),2)
outcome_prop
```

```
##
##   No  Yes
## 0.65 0.35
```

```
# Bar plot of distribution
ggplot(train, aes(x=Outcome)) +
  geom_bar(fill="blue") +
  labs(x="Diabetes Outcome", y="Frequency", title="Distribution of Diabetes Outcome")
```

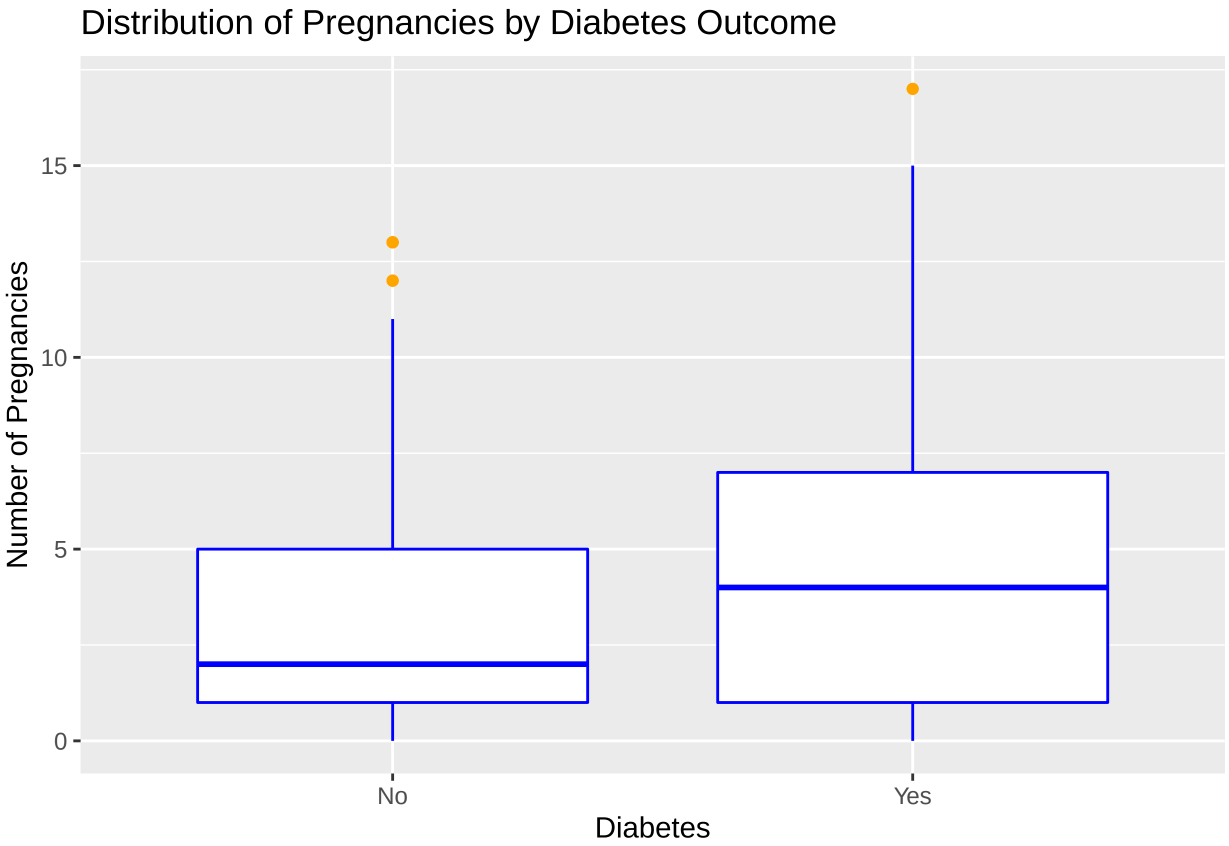


About 35% of patients in this data set have diabetes while 65% do not. More don't have diabetes, but this is not a huge imbalance.

Visualizaling Response and Potential Predictors

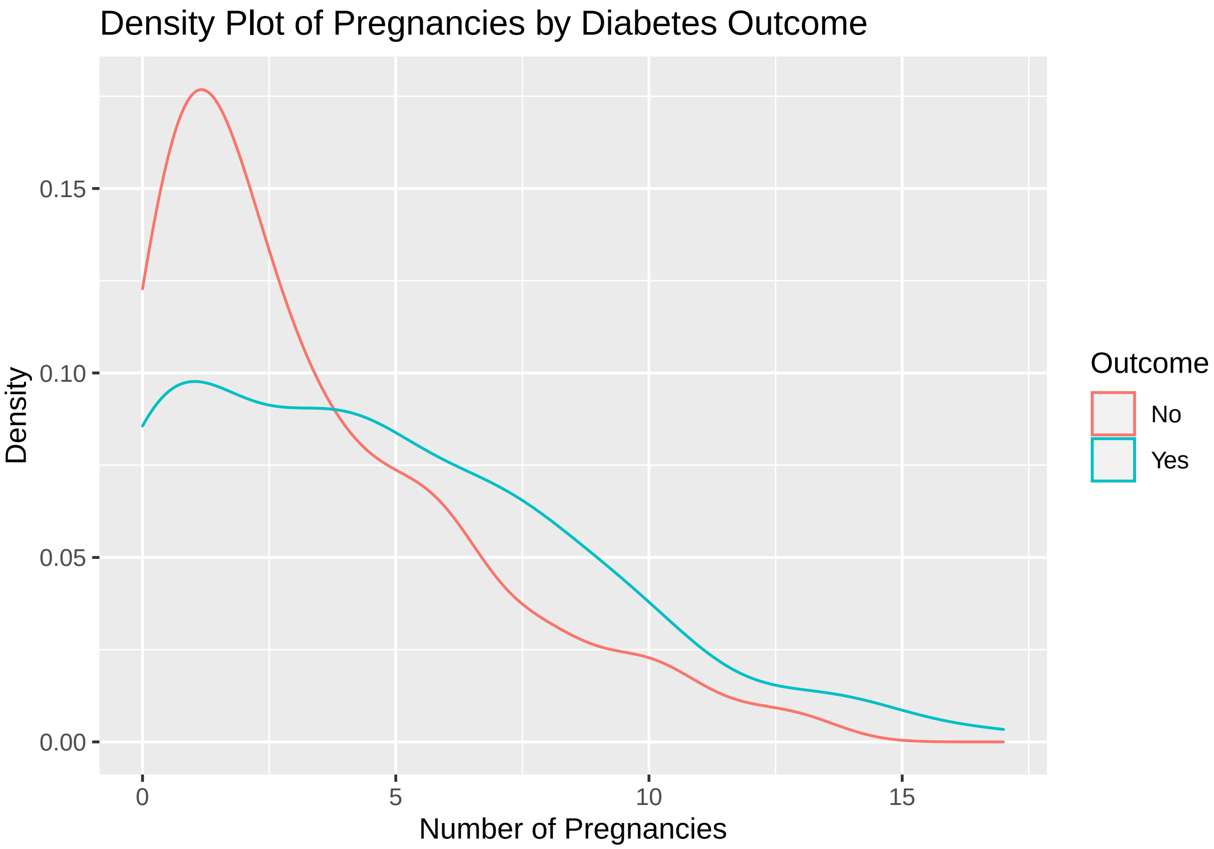
Pregnancies

```
# Side by side box plot
ggplot(train, aes(x=Outcome, y=Pregnancies))+
  geom_boxplot(color = "blue", outlier.color = "orange") +
  labs(x="Diabetes", y="Number of Pregnancies", title="Distribution of Pregnancies by Diabetes Outcome")
```



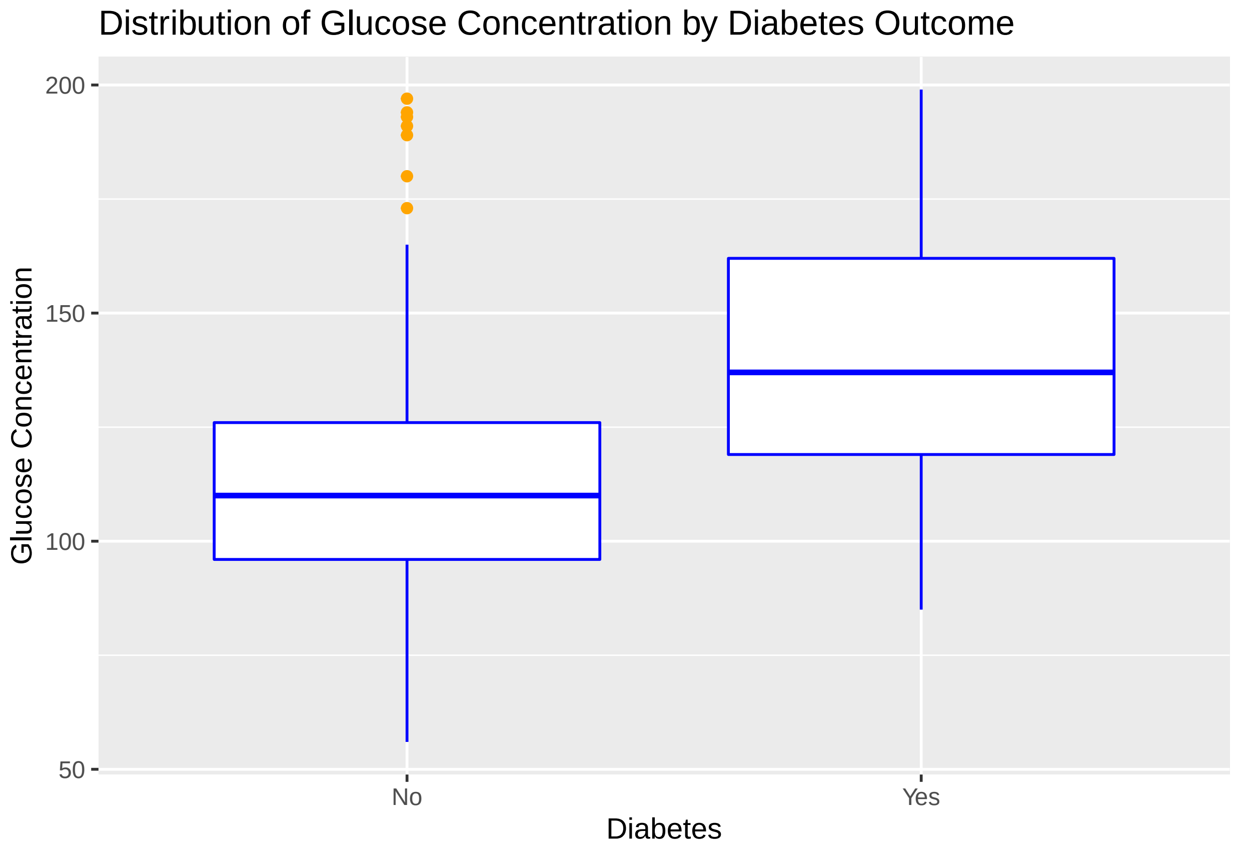
There is a fair amount of overlap in these box plots, but they do suggest that patients with diabetes tend to have higher number of pregnancies. We also observe this in the density plots where patients without diabetes are more likely to have had small numbers of or no pregnancies.

```
# Density plot
ggplot(train, aes(x=Pregnancies, color=Outcome)) +
  geom_density() +
  labs(x="Number of Pregnancies", y="Density", title="Density Plot of Pregnancies by Diabetes Outcome")
```



### Glucose

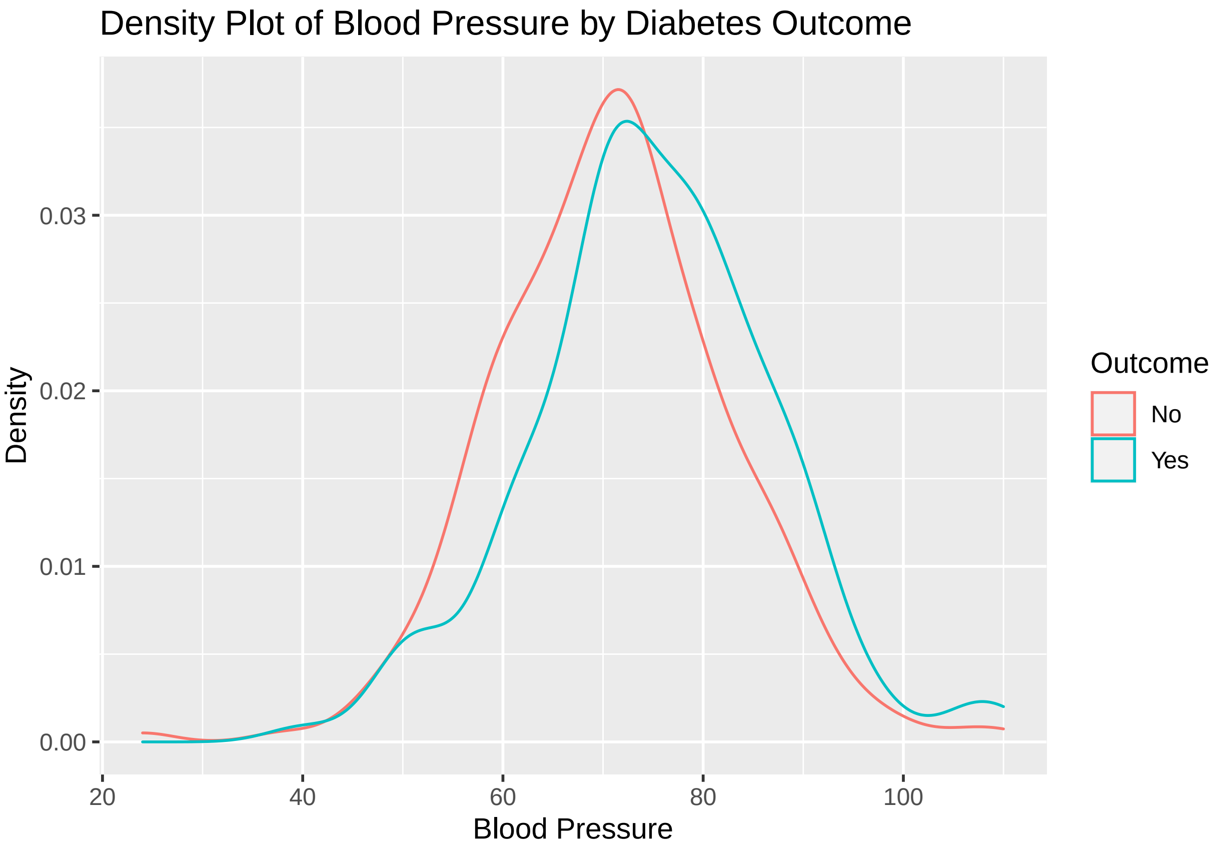
```
# Side by side box plot
ggplot(train, aes(x=Outcome, y=Glucose))+
  geom_boxplot(color = "blue", outlier.color = "orange") +
  labs(x="Diabetes", y="Glucose Concentration", title="Distribution of Glucose Concentration by Diabetes
```



These box plots (especially from Q1 to Q3) have little overlap so we suspect that glucose will be predictive in identifying which patients have diabetes. From this visual we can see that patients with diabetes tend to have higher glucose concentrations.

Blood Pressure

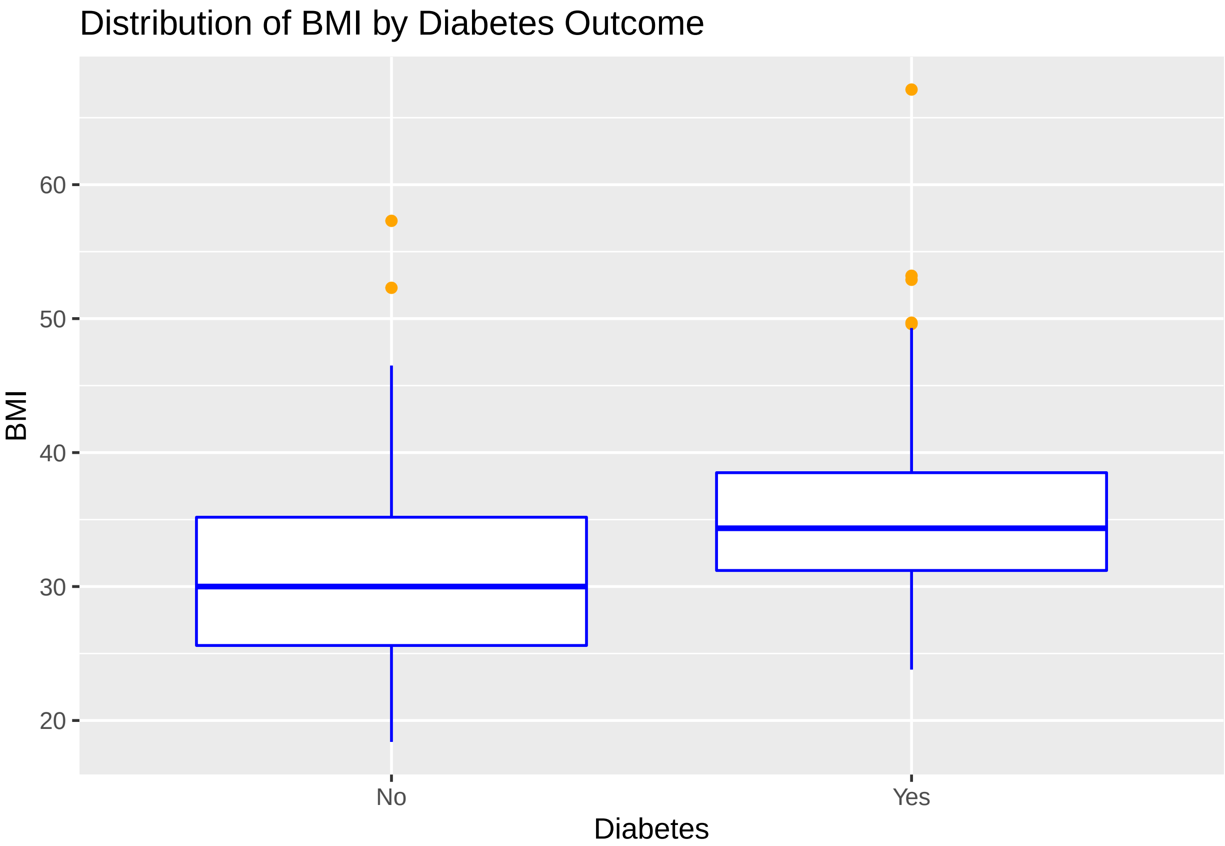
```
# Density plot
ggplot(train, aes(x=BloodPressure, color=Outcome)) +
  geom_density() +
  labs(x="Blood Pressure", y="Density", title="Density Plot of Blood Pressure by Diabetes Outcome")
```



There is quite a bit of overlap in the density plots of blood pressure for those that do and do not have diabetes. While there is a lot of overlap we do observe that those with diabetes tend to have slightly higher blood pressure than those that do not.

BMI

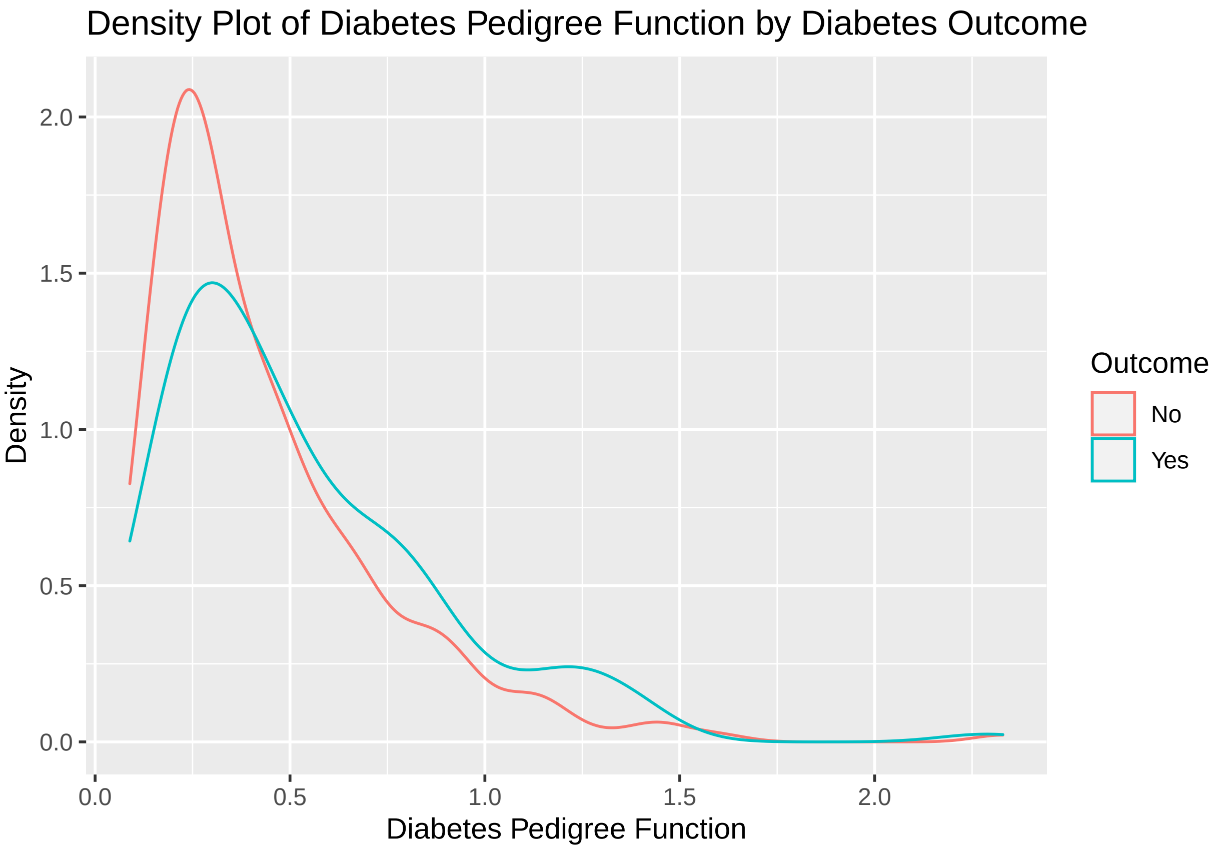
```
# Side by side box plot
ggplot(train, aes(x=Outcome, y=BMI))+
  geom_boxplot(color = "blue", outlier.color = "orange") +
  labs(x="Diabetes", y="BMI", title="Distribution of BMI by Diabetes Outcome")
```



Patients with diabetes tend to have higher BMIs than those without.

Diabetes Pedigree Function

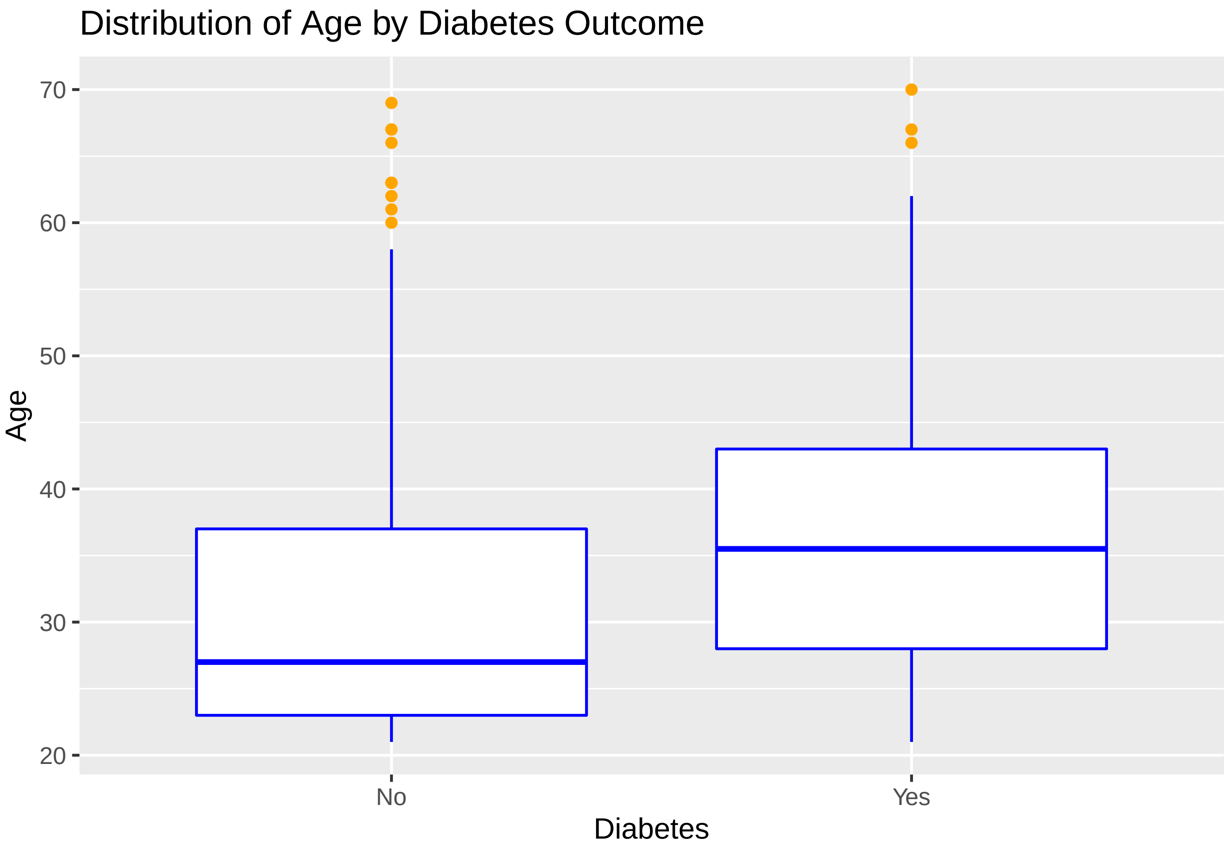
```
# Density plot
ggplot(train, aes(x=DiabetesPedigreeFunction, color=Outcome)) +
  geom_density() +
  labs(x="Diabetes Pedigree Function", y="Density", title="Density Plot of Diabetes Pedigree Function by
```



Those with lower Diabetes Pedigree Functions (less family history of diabetes???) appear less likely to have diabetes.

Age

```
# Side by side box plot
ggplot(train, aes(x=Outcome, y=Age))+
  geom_boxplot(color = "blue", outlier.color = "orange") +
  labs(x="Diabetes", y="Age", title="Distribution of Age by Diabetes Outcome")
```

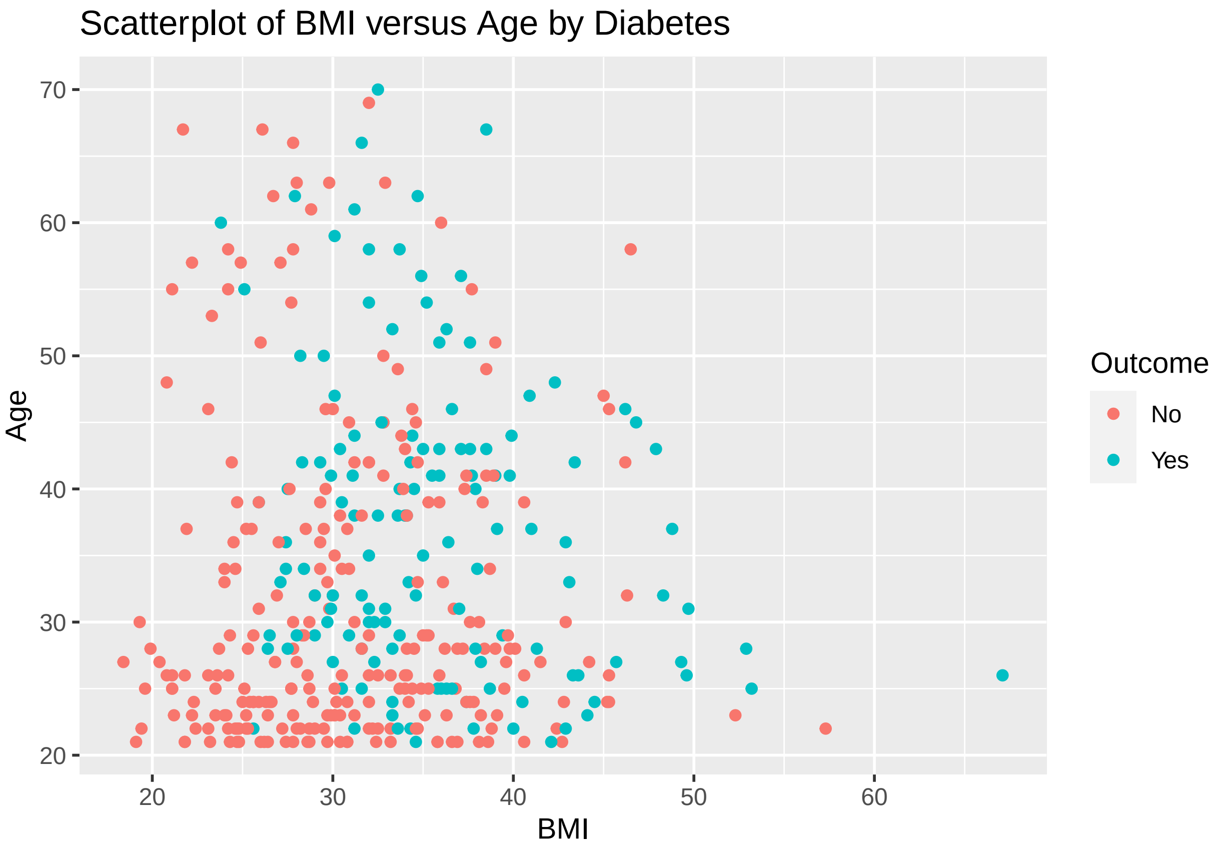


The median age of patients with diabetes is higher than those that do not have diabetes.

Multiple Predictors and Response

Next, we created scatter plots between pairs of potential predictors and colored the dots with the diabetes outcome. From this scatter plot of BMI versus age there does seem to be a relationship with diabetes where the women with diabetes tend to be higher BMIs and older as well.

```r
# Scatter plot of potential predictors (colored by response)
# bmi and blood pressure colored by diabetes
ggplot(train, aes(x=BMI, y=Age, color=Outcome)) +
  geom_point() +
  labs(x="BMI", y="Age", title="Scatterplot of BMI versus Age by Diabetes")
```



Logistic Regression Model

Fit Initial Model

```
# Fit initial regression model
full <- glm(Outcome~., family="binomial", data=train)
summary(full)
```

```
##
## Call:
## glm(formula = Outcome ~ ., family = "binomial", data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.4453  -0.7416  -0.4513   0.7522   2.0878
##
## Coefficients:
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -8.066680   1.040066  -7.756 8.77e-15 ***
## Pregnancies               0.084191   0.042291   1.991   0.0465 *
## Glucose                   0.032040   0.004985   6.428 1.30e-10 ***
## BloodPressure            -0.015570   0.012356  -1.260   0.2076
## BMI                       0.097023   0.021665   4.478 7.53e-06 ***
## DiabetesPedigreeFunction  0.514970   0.394550   1.305   0.1918
## Age                       0.022129   0.012858   1.721   0.0852 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 496.74  on 383  degrees of freedom
## Residual deviance: 375.18  on 377  degrees of freedom
## AIC: 389.18
##
## Number of Fisher Scoring iterations: 4
```

Based on the summary of this initial logistic regression model we observe that Glucose and BMI and highly significant in predicting who has diabetes. These two variables had the largest differences in the density and box plots above so it is not surprising that they are highly predictive of diabetes. Pregnancies is also significant at a 5% significance level. The other predictors, Blood Pressure, Diabetes Pedigree Function, and Age do not add as much value given all of the other predictors are fit in the model. This does not mean that these three variables are not related to diabetes, it only means that they may not add value given that the other variables are also included in the model.

Test Hypothesis on Subset of Parameters

We will test whether we can drop all three of these predictors. For this test our null hypothesis will be $H_0 : \beta_{BloodPressure} = \beta_{DiabetesPredigreeFunction} = \beta_{Age} = 0$ and the alternative is H_A : at least one β in H_0 is non-zero.

```r
reduced <- glm(Outcome~Pregnancies + Glucose + BMI, family="binomial", data=train)
summary(reduced)
```

```
##
## Call:
## glm(formula = Outcome ~ Pregnancies + Glucose + BMI, family = "binomial",
##     data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.0937  -0.7426  -0.4572   0.7804   2.1535
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -8.037254   0.884026  -9.092  < 2e-16 ***
## Pregnancies  0.106819   0.036251   2.947  0.00321 **
## Glucose      0.033449   0.004828   6.928 4.27e-12 ***
## BMI          0.084303   0.019571   4.307 1.65e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 496.74  on 383  degrees of freedom
## Residual deviance: 381.41  on 380  degrees of freedom
## AIC: 389.41
##
## Number of Fisher Scoring iterations: 4
```

All of the predictors in this three variable model are statistically significant.

```
# Test statistic
test_stat <- reduced$deviance - full$deviance
# P-value
p_value <- 1 - pchisq(test_stat, 3)
p_value
```

```
## [1] 0.1011126
```

The p-value of 0.10 is not less than our significance level of 0.05 so we fail to reject the null hypothesis. We don't have enough evidence to say that any of the three variables we dropped in the reduced model have a non-zero coefficient and thus we can use our reduced model with just Pregnancies, Glucose, and BMI to predict diabetes.

Test Whether Model is Useful

Next, we'll test whether our reduced model is useful in predicting diabetes. In other words, can we drop all coefficients from our model? We'll start from our reduced model given the results from our hypothesis test above. For this test our null hypothesis will be $H_0 : \beta_{Pregnancies} = \beta_{Glucose} = \beta_{BMI} = 0$ and the alternative is H_A : at least one β in H_0 is non-zero.

```
# Test statistic
test_stat2 <- reduced$null.deviance - reduced$deviance
# P-value
p_value2 <- 1 - pchisq(test_stat2, 3)
p_value2
```

```
## [1] 0
```

The p-value for this hypothesis is 0 so we reject the null hypothesis and conclude that at least one of predictors has a non-zero coefficient and thus this logistic regression model is useful in estimating the odds of developing diabetes.

Interpreting Model's Coefficients

```
reduced
```

```
##
## Call:  glm(formula = Outcome ~ Pregnancies + Glucose + BMI, family = "binomial",
##     data = train)
##
## Coefficients:
## (Intercept)  Pregnancies      Glucose          BMI
##    -8.03725      0.10682      0.03345      0.08430
##
## Degrees of Freedom: 383 Total (i.e. Null);  380 Residual
## Null Deviance:       496.7
## Residual Deviance: 381.4     AIC: 389.4
```

Our estimated logistic regression equation is: $log(\frac{\hat{\pi}}{1-\hat{\pi}}) = -8.03725 + 0.10682 * Pregnancies + 0.03345 * Glucose + 0.08430 * BMI$. All three of these predictors have positive coefficients, suggesting that for larger number of pregnancies, higher glucose concentration, and higher BMI the odds of having diabetes goes up, with all other variables held constant.

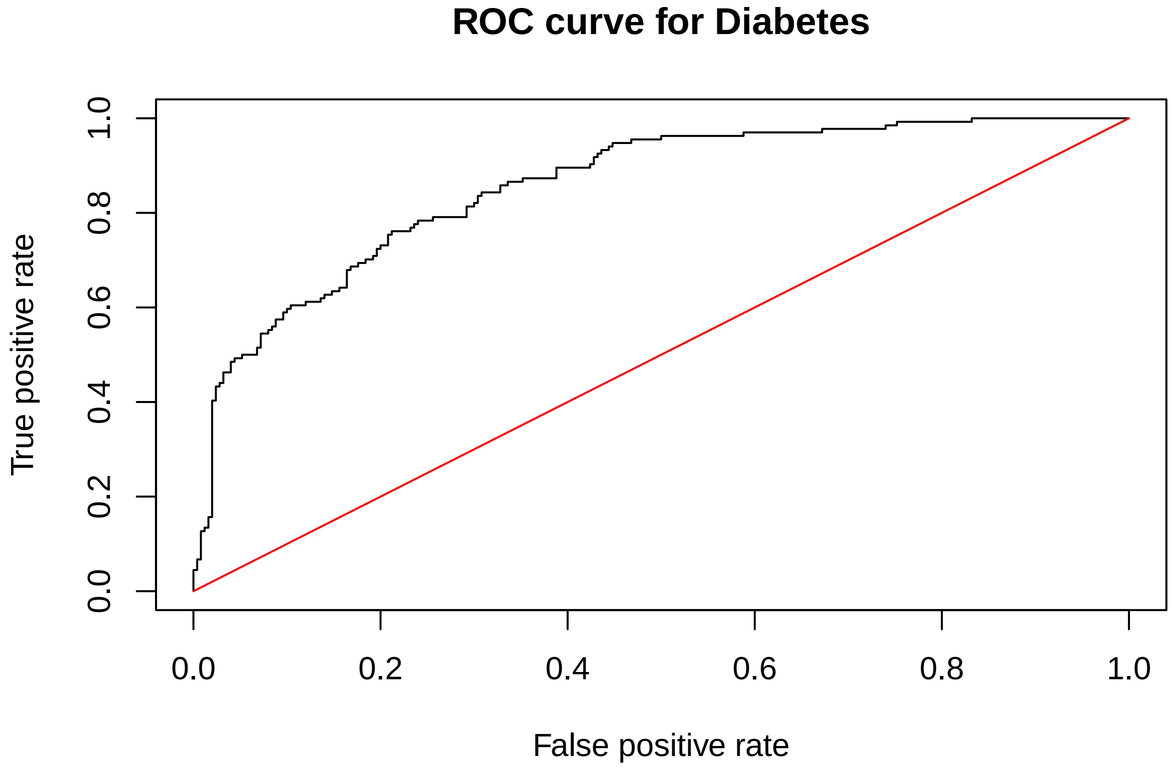
The estimated coefficient for pregnancies is 0.10682. This means that for each additional pregnancies the log odds that the woman will have diabetes increases by 0.10682 while controlling for glucose concentration and BMI. In other words, for each one unit increase in pregnancies the odds that the woman has diabetes is multiplied by 1.112734, while holding the other variables constant.

Model Evaluation

Use model built on training data to estimate the probabilities for observations in the unseen test data set.

Plot ROC curve

```
# Predicted diabetes rate
preds <- predict(reduced, newdata=test, type="response")
# Transform input data
rates <- prediction(preds, test$Outcome)
# Store true positive and false positive rates
roc_result <- performance(rates, measure="tpr", x.measure="fpr")
# Plot roc curve and overly diagonal (random)
plot(roc_result, main = "ROC curve for Diabetes")
lines(x=c(0,1), y=c(0,1), col="red")
```



The ROC curve lies above the straight diagonal line, suggesting that this model identifies/classifies people who have diabetes better than random.

Calculate AUC

```
auc <- performance(rates, measure="auc")
auc@y.values
```

```
## [[1]]
## [1] 0.856597
```

The value of AUC for the ROC curve above is 0.856597 Since this value is greater than 0.5 the model does better than random for classifying who develops diabetes.

Create Confusion Matrix

TODO QUESTION: what do we want to use for a threshold? I just did 0.5 for now, but maybe in predicting diabetes, due to the health implications, we may be more concerned about false negatives (classify someone as not having diabetes when they do have it) so we've chose a threshold lower than 0.5. With this lower threshold we'll have a lower false negative rate (as desired), but as a result would have a higher false positive rate.

```
threshold = 0.5 # Define threshold
table(test$Outcome, preds>threshold)
```

```
##
##       FALSE TRUE
##   No    226   24
##   Yes    55   79
```

```
# Accuracy
(226+79)/(226+24+55+79)
```

```
## [1] 0.7942708
```

```
# TPR
79/(55+79)
```

```
## [1] 0.5895522
```

```
# TNR
226/(226+24)
```

```
## [1] 0.904
```

At a threshold of 0.5, the models' overall accuracy is 79% with a true positive rate of 59% and a false positive rate of 90.4%.