Technical Report

A Straightforward Analysis of the Pima Native American Dataset

Shukry Zablah, Jack Koravos, Jasmine Horan, and Jayne Vogelzang

December 17, 2018

Contents

0.1 Setup

```r
library(dplyr)
library(tidyr)
library(caTools)
library(mosaic)
library(ggplot2)
library(ggcorrplot)
library(ROCR)
library(caret)
library(tibble)
library(car)
library(ggthemes)
```

0.2 Introduction

To see how the data files were created you can check out Appendix B or go to the actual source file in the repository.

Let's familiarize ourselves with the dataset.

0.2.1 Global Characteristics

```
names(PIMA)
```

```
## [1] "pregnancies"              "glucoseConcentration"
## [3] "bloodPressure"            "skinThickness"
## [5] "insulin"                  "bmi"
## [7] "diabetesPedigreeFunction" "age"
## [9] "hasDiabetes"
```

These are the variable names. You can check out a description for each variable in the codebook file in the references folder in the github repository.

```
dim(PIMA)
```

```
## [1] 392   9
```

We have 768 observations with 8 features and our response variable.

Let's take a look at some of the observations to familiarize ourselves with the dataset.

```
head(PIMA)
```

```
##    pregnancies glucoseConcentration bloodPressure skinThickness insulin
## 4            1                   89            66            23      94
## 5            0                  137            40            35     168
## 7            3                   78            50            32      88
## 9            2                  197            70            45     543
## 14           1                  189            60            23     846
## 15           5                  166            72            19     175
##     bmi diabetesPedigreeFunction age hasDiabetes
## 4  28.1                    0.167  21           0
## 5  43.1                    2.288  33           1
## 7  31.0                    0.248  26           1
## 9  30.5                    0.158  53           1
## 14 30.1                    0.398  59           1
## 15 25.8                    0.587  51           1
```

Descriptive statistics for the variables in our dataset are provided below.

```
summary(PIMA)
```

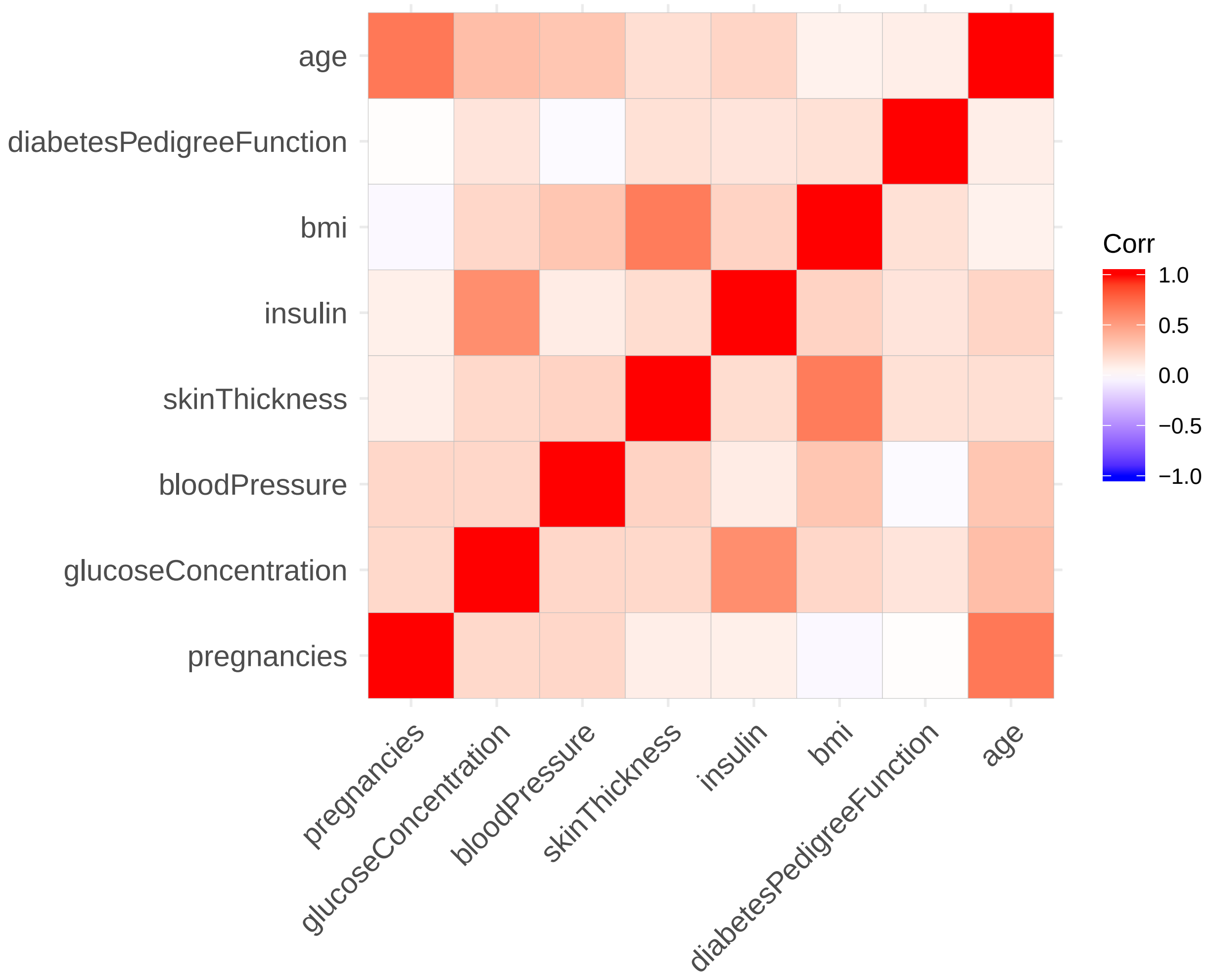
```
##   pregnancies     glucoseConcentration bloodPressure    skinThickness
##  Min.   : 0.000   Min.   : 56.0        Min.   : 24.00   Min.   : 7.00
##  1st Qu.: 1.000   1st Qu.: 99.0        1st Qu.: 62.00   1st Qu.:21.00
##  Median : 2.000   Median :119.0        Median : 70.00   Median :29.00
##  Mean   : 3.301   Mean   :122.6        Mean   : 70.66   Mean   :29.15
##  3rd Qu.: 5.000   3rd Qu.:143.0        3rd Qu.: 78.00   3rd Qu.:37.00
##  Max.   :17.000   Max.   :198.0        Max.   :110.00   Max.   :63.00
##     insulin            bmi        diabetesPedigreeFunction      age
##  Min.   : 14.00   Min.   :18.20   Min.   :0.0850           Min.   :21.00
##  1st Qu.: 76.75   1st Qu.:28.40   1st Qu.:0.2697           1st Qu.:23.00
##  Median :125.50   Median :33.20   Median :0.4495           Median :27.00
##  Mean   :156.06   Mean   :33.09   Mean   :0.5230           Mean   :30.86
##  3rd Qu.:190.00   3rd Qu.:37.10   3rd Qu.:0.6870           3rd Qu.:36.00
##  Max.   :846.00   Max.   :67.10   Max.   :2.4200           Max.   :81.00
##  hasDiabetes
##  0:262
##  1:130
##
##
##
##
```

We will not include our univariate analysis in this document, however, an important finding to keep in mind is that all the variables except glucose concentration and blood pressure were skewed to the right. These skewed distributions mean that most of our population is young. To generalize our findings this sample of about 400 women must be representative of the population of PIMA women as a whole. We proceed under this assumption.

0.3 Multivariate Analysis

Before fitting any models we explore any relationships between the features in our dataset. Let's focus on the features.

```
PIMA %>%
  select(-hasDiabetes) %>%
  cor() %>%
  ggcorrplot()
```



In the correlation plot above we can see that there are some featured that are correlated. This is a hint that we might not need both features of a correlated pair in our model as they are likely to not add valuable information. We can see that the strongest correlated features are:

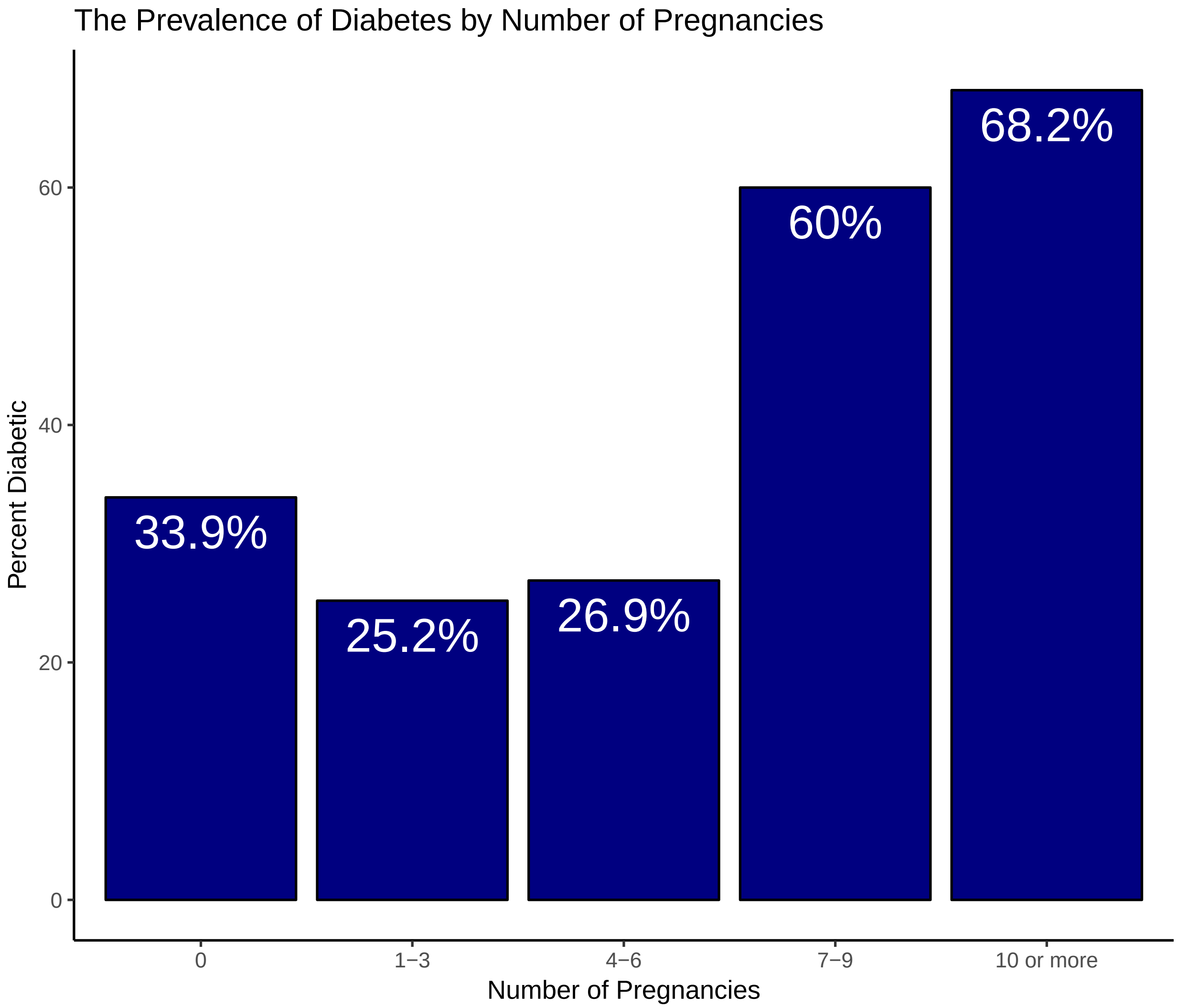
- insulin and glucose concentration: 0.581223
- age and pregnancies: 0.6796085
- bmi and skin thickness: 0.6643549

0.3.1 Visualizations

We decided to include the following visualizations that were helpful in observing trends and relationships between the variables:

0.3.1.1 Percentages of Diabetes in Binned Pregnancies Groups

```
pregnancies_labels <- c("0", "1-3", "4-6", "7-9", "10 or more")
PIMA %>%
  mutate(pregnancies = cut(pregnancies,
                           breaks = c(-.5,.5,3.5,6.5,9.5, 50),
                           labels = pregnancies_labels)) %>%
  group_by(pregnancies, hasDiabetes) %>%
  summarize(count = n()) %>%
  spread(hasDiabetes, count) %>%
  mutate(percentage = round(100*(`1`/(`0`+`1`)),1)) %>%
  ggplot(aes(x = pregnancies, y = percentage, label = paste0(percentage, "%"))) +
    geom_bar(stat = "identity", fill = "navy", color = "black") +
    geom_text(vjust=1.5, size = 7, color="white") +
    labs(x = "Number of Pregnancies",
         y = "Percent Diabetic",
         title = "The Prevalence of Diabetes by Number of Pregnancies") +
    scale_x_discrete(limits = pregnancies_labels)
```

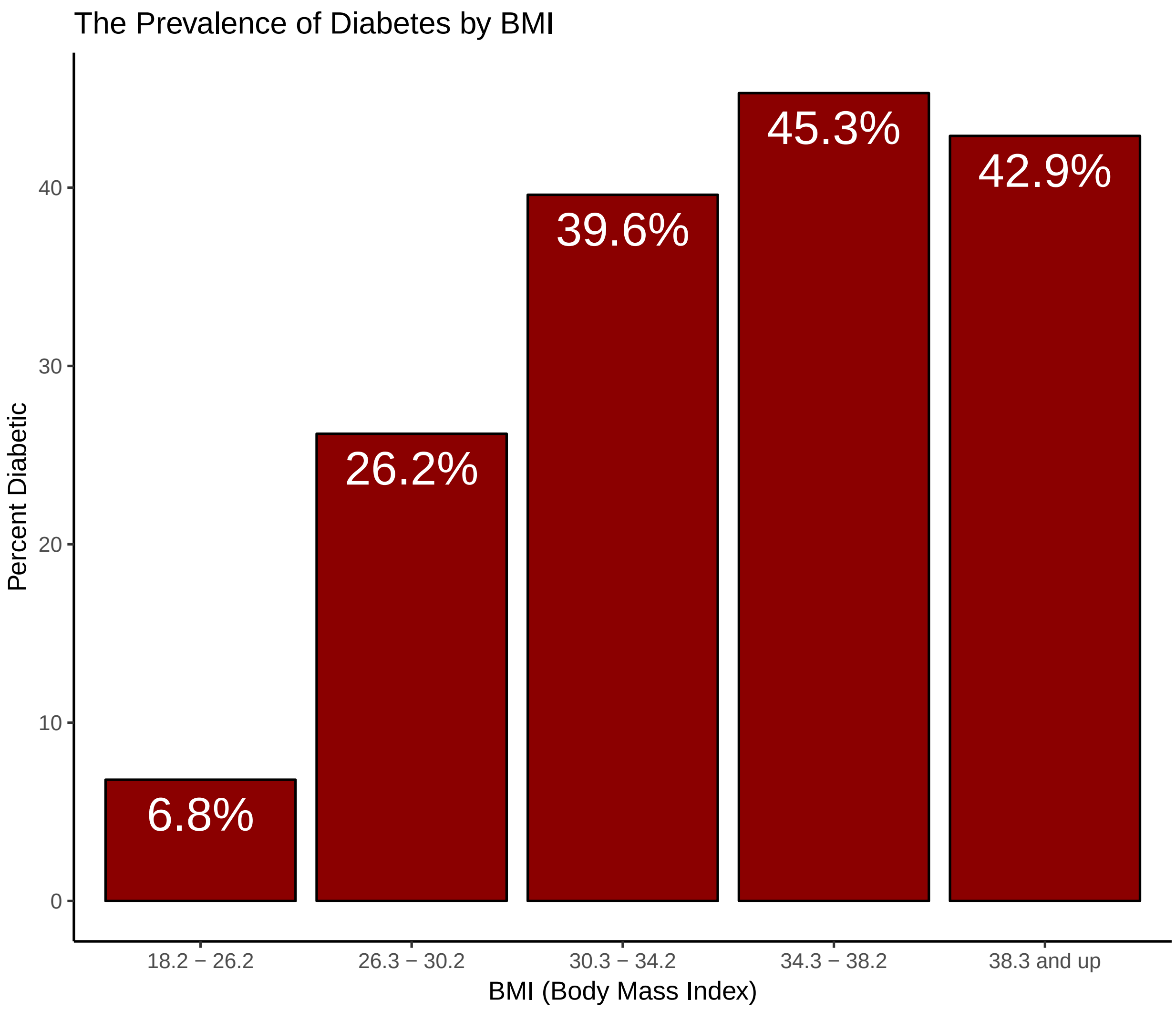


In the graph above we can see that the number of pregnancies is correlated with the percentage of people that have diabetes. In particular, there is a significant increase in percentage of diabetic PIMA women in the groups of women that had 7 or more pregnancies. In the context of this study however, the number of pregnancies might not be as an important feature as it appears to be. Recall the correlation of pregnancies and age for our observations. The correlation between those two variables is 0.6796085. The value supports the idea that both variables are probably not going to make it to the final model. Later on we will test which variable explains more of the variation of our response.

Observation: You might be wondering why the first column in the visual is higher than the expected value. A possible explanation for this comes from not being able to discern missing values from the actual data (both coded with 0). To read more about this go to Appendix B.

0.3.1.2 Percentages of Diabetes in Binned BMI Groups

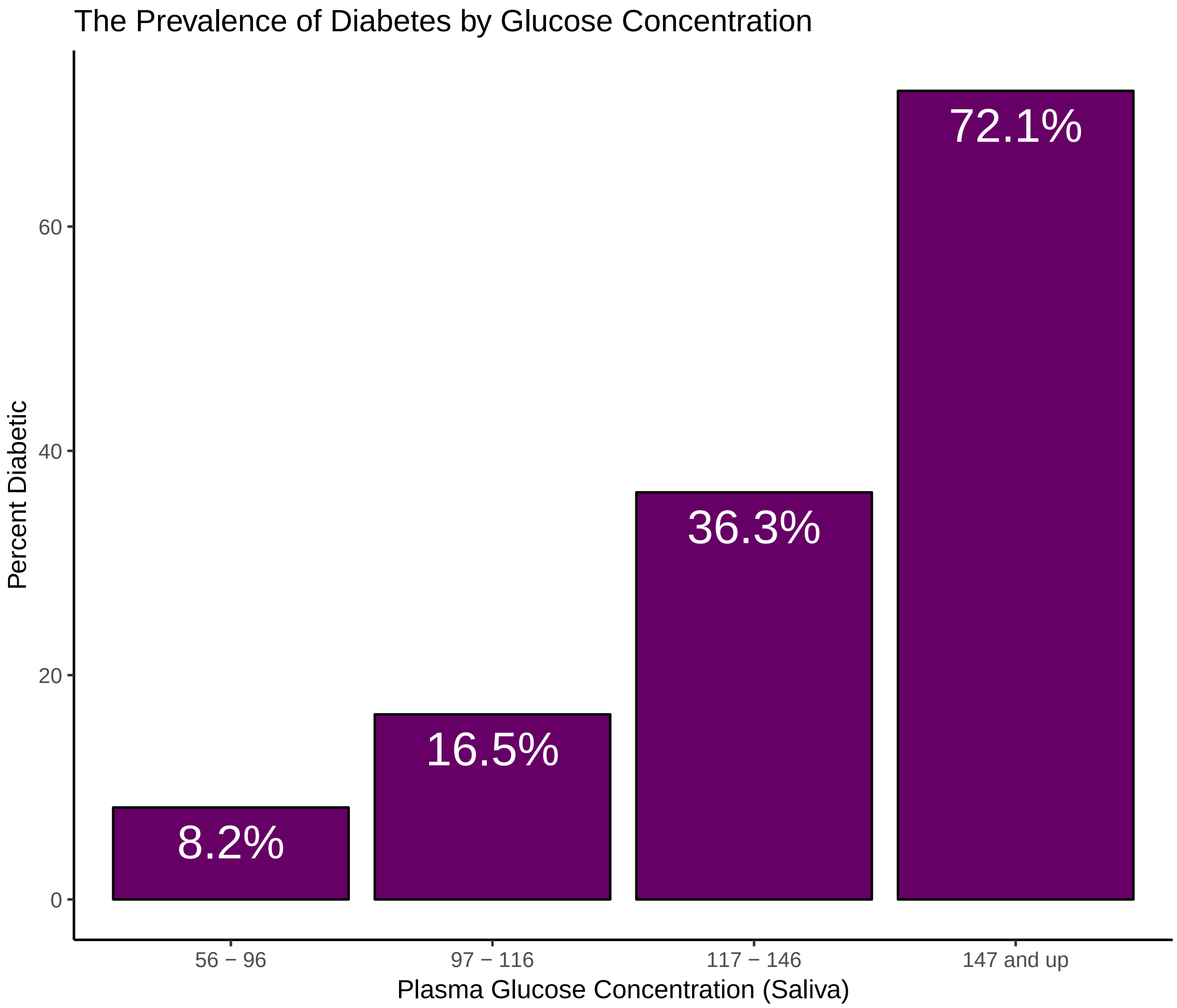
```
bmi_labels <- c("18.2 - 26.2", "26.3 - 30.2",
                  "30.3 - 34.2", "34.3 - 38.2", "38.3 and up")
PIMA %>%
  mutate(bmi = cut(bmi,
                           breaks = c(0,26.2, 30.2, 34.2, 38.2, 100),
                           labels = bmi_labels)) %>%
  group_by(bmi, hasDiabetes) %>%
  summarize(count = n()) %>%
  spread(hasDiabetes, count) %>%
  mutate(percentage = round(100*(`1`/(`0`+`1`)),1)) %>%
  ggplot(aes(x = bmi, y = percentage, label = paste0(percentage, "%"))) +
    geom_bar(stat = "identity", fill = "dark red", color = "black") +
    geom_text(vjust=1.5, size = 7, color="white") +
    labs(x = "BMI (Body Mass Index)",
         y = "Percent Diabetic",
         title = "The Prevalence of Diabetes by BMI") +
    scale_x_discrete(limits = bmi_labels)
```



In this graph we can see a strong positive relationship between the bmi range and the percentage of diabetic women in the group. For reference, The three rightmost columns are women that would be considered obese (as given by bmi cutoffs). There is a slight dip in the group of women with the highest bmi, which is too small of a signal to delve into, but could be explained if the bmi of a women was related with whether they have diabetes or not only up to a certain score. Perhaps past a score, BMI does not indicate a higher chance of having diabetes.

0.3.1.3 Percentages of Diabetes in Binned Glucose Concentration Groups

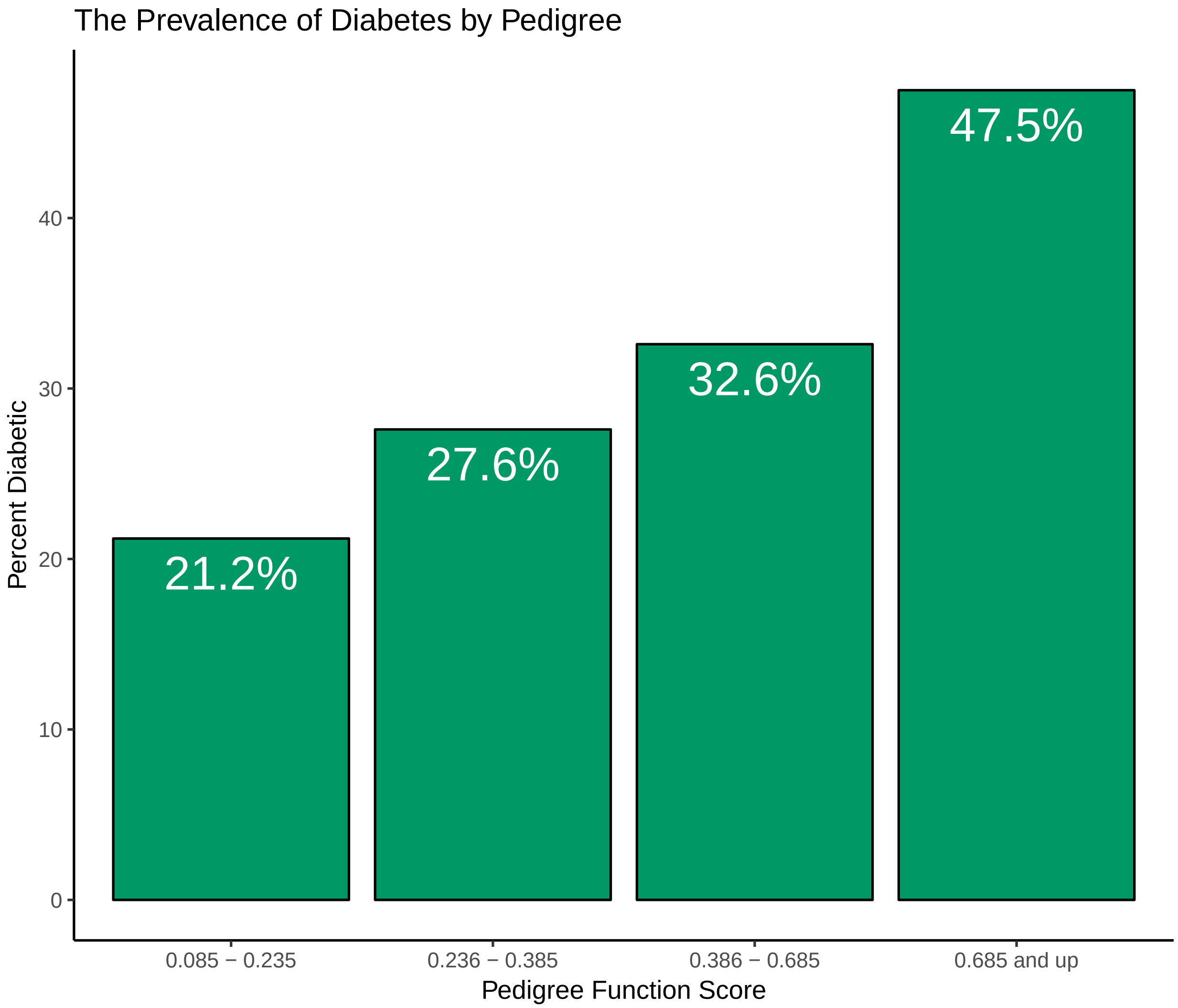
```
glucose_labels <- c("56 - 96", "97 - 116",
                    "117 - 146", "147 and up")
PIMA %>%
  mutate(glucoseConcentration = cut(glucoseConcentration,
                                  breaks = c(50, 96, 116, 146, 200),
                                  labels = glucose_labels)) %>%
  group_by(glucoseConcentration, hasDiabetes) %>%
  summarize(count = n()) %>%
  spread(hasDiabetes, count) %>%
  mutate(percentage = round(100*(`1`/(`0`+`1`)),1)) %>%
  ggplot(aes(x = glucoseConcentration,
             y = percentage,
             label = paste0(percentage, "%"))) +
    geom_bar(stat = "identity", fill = "#660066", color = "black") +
    geom_text(vjust=1.5, size = 7, color="white") +
    labs(x = "Plasma Glucose Concentration (Saliva)",
         y = "Percent Diabetic",
         title = "The Prevalence of Diabetes by Glucose Concentration") +
    scale_x_discrete(limits = glucose_labels)
```



Again, we have a very strong positive relationship between the bins of plasma glucose concentration in saliva and the percentage of women who are diabetic in those groups. This relationship indicates that glucose concentration is a good variable to include in our model.

0.3.1.4 Percentages of Diabetes in Binned Diabetes Pedigree Function

```
pedigree_labels <- c("0.085 - 0.235", "0.236 - 0.385",
                     "0.386 - 0.685", "0.685 and up")
PIMA %>%
  mutate(diabetesPedigreeFunction = cut(diabetesPedigreeFunction,
                           breaks = c(0,0.235, 0.385, 0.685, 3),
                           labels = pedigree_labels)) %>%
  group_by(diabetesPedigreeFunction, hasDiabetes) %>%
  summarize(count = n()) %>%
  spread(hasDiabetes, count) %>%
  mutate(percentage = round(100*(`1`/(`0`+`1`)),1)) %>%
  ggplot(aes(x = diabetesPedigreeFunction,
             y = percentage,
             label = paste0(percentage, "%"))) +
    geom_bar(stat = "identity", fill = "#009966", color = "black") +
    geom_text(vjust=1.5, size = 7, color="white") +
    labs(x = "Pedigree Function Score",
         y = "Percent Diabetic",
         title = "The Prevalence of Diabetes by Pedigree") +
    scale_x_discrete(limits = pedigree_labels)
```



The diabetes pedigree function score bins also show to be related to a larger percentage of women who have diabetes. This is another variable that will be useful to include in our model, especially because it has a low correlation with the glucose concentration feature. (cor = 0.1401802).

In the next section we will start training and evaluating models that will help us assess the risk of diabetes for individual observations.

0.4 Modeling

This section contains anything related to finding the best model for predicting the probability of having diabetes. The general layout is as follows:

- Create baseline
- Form a model with one predictor
- Form a model with all predictors
- Optimize and select significant predictors

At every point we will evaluate the models on the test set so that we get an idea of how much improvement our work is generating.

Observation: If you want to see if our models satisfy their assumptions, go over to Appendix C.

0.4.1 Baseline

In machine learning it is necessary to have a baseline in order to see if our predictive models are of any use. Here, we chose to take that baseline as the percentage of people that have diabetes in our train set (almost equal to that of the whole dataset).

```
tally(~ hasDiabetes, data = train)
```

```
## hasDiabetes
##   0   1
## 196  98
```

Any model we choose has to have an accuracy higher than 98/(98+196) = 33%. Otherwise, we are better off just guessing based on the population percentage.

0.4.2 Simple Classifier

From our previous analysis glucose concentration looks like the variable that is more predicitive of whether a woman has diabetes or not. Thus we create a simple logistic regression model based on this sole predictor.

0.4.2.1 Training

```
Fit_LR_Simple <- train(hasDiabetes ~ glucoseConcentration, data = train,
                 method = "glm",
                 trControl = trainControl(method = "none"))
clf_LR_Simple <- with(Fit_LR_Simple, finalModel)
summary(clf_LR_Simple)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.2242  -0.7460  -0.4776   0.7116   2.3714
```

```
##
## Coefficients:
##                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)           -6.261262   0.741942  -8.439  < 2e-16 ***
## glucoseConcentration  0.043892   0.005626   7.801 6.14e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 374.27  on 293  degrees of freedom
## Residual deviance: 288.26  on 292  degrees of freedom
## AIC: 292.26
##
## Number of Fisher Scoring iterations: 4
```

The model confirms that the glucoseConcentration is a significant predictor. This simple model has an AIC of 292.26. To better understand how good this value is we now we evaluate the model on the test set and see how well it performs.

0.4.2.2 Model Evaluation on Test Set

```
predict_LR_Simple <- predict(clf_LR_Simple, type = 'response', newdata = test)

with(test,
     table(hasDiabetes, predict_LR_Simple > 0.3))
```

```
##
## hasDiabetes FALSE TRUE
##           0    47   19
##           1    10   22
```

We can see in the confusion matrix that our accuracy is $(47 + 22)/(47 + 19 + 10 + 22) = 0.7041$. This is with a cutoff of 0.3.

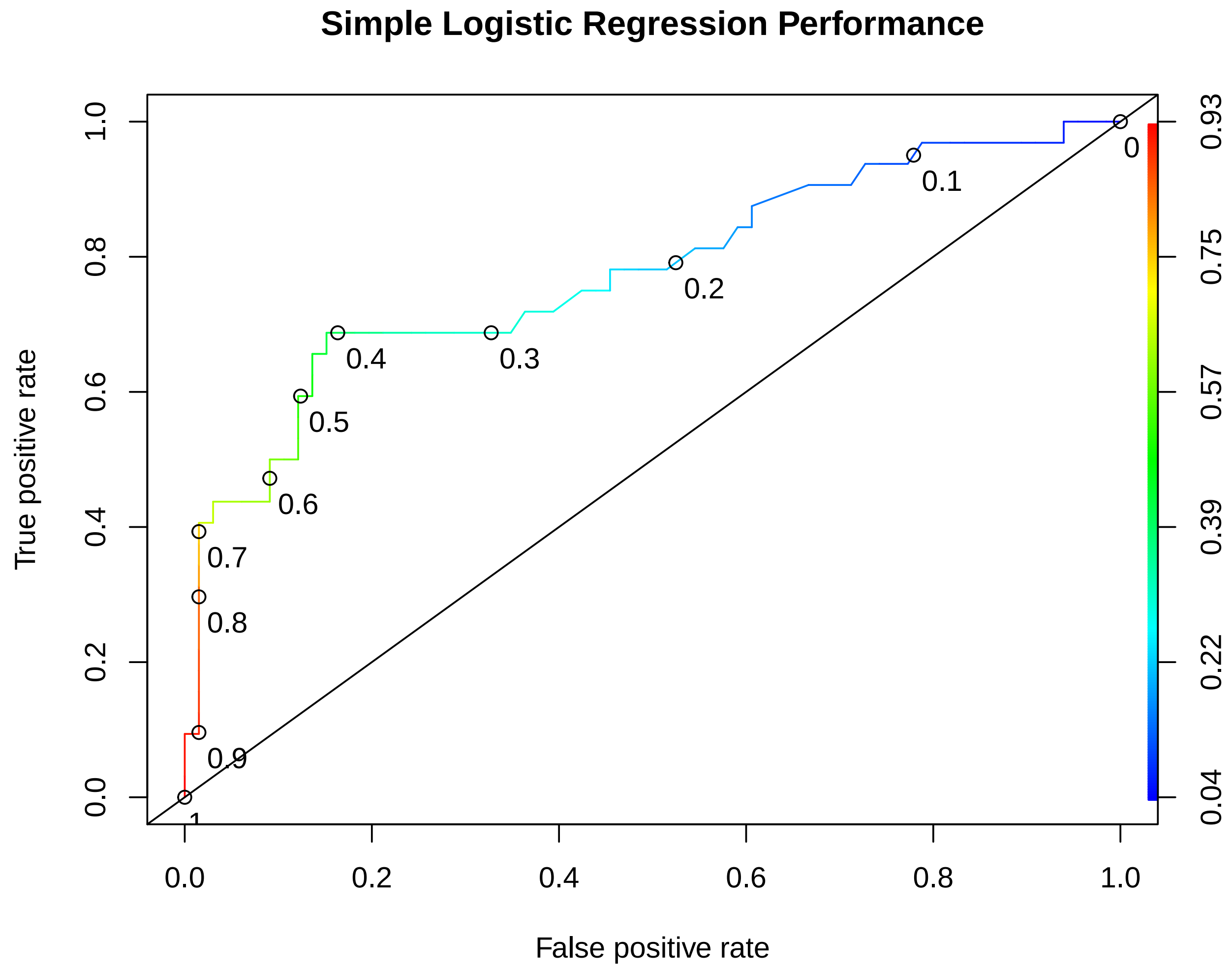
Observation: The cutoff of 0.3 was selected in order to reduce the number of false negatives of our model. We are prepared to sacrifice a tiny percentage of accuracy if it means that our model will be more useful when it comes to not misclassifying people with diabetes as not diabetic. It can be adjusted to create a more sensitive model.

0.4.2.3 Model Performance Visualization

```
ROCRpred_LR_Simple <- with(test,
                 prediction(predict_LR_Simple, hasDiabetes))

ROCRperf_LR_Simple <- performance(ROCRpred_LR_Simple, 'tpr','fpr')

plot(ROCRperf_LR_Simple,
     colorize = TRUE,
     print.cutoffs.at = seq(0,1,0.1),
     text.adj = c(-0.2,1.7),
     main = "Simple Logistic Regression Performance"); abline(0,1)
```



We will compare these ROC curves to determine how the models specificity/sensitivity improves or worsens as we make our model more complicated, and how the overall accuracy changes. For now, we have to keep in mind that the closer the ROC curve is to the upper leftmost corner of the graph, the better the model is.

Let's see if we can do better than this.

0.4.3 Full Classifier

A logistic regression model with all the predictors should not be the final model we choose. Our previous multivariate analysis has shown that there are relationships between the variables that have to be ommitted from the final model. However, to know what variables we won't include, we will evaluate the full model.

0.4.3.1 Training

```
set.seed(1)
Fit_LR_Full <- train(hasDiabetes ~ ., data = train,
                     method = "glm",
                     trControl = trainControl(method = "none"))
clf_LR_Full <- with(Fit_LR_Full, finalModel)
```

```
#this is the same model as:
####glm(hasDiabetes ~ ., family = binomial(link = "logit"), data = train)
#there are advantages to familiarizing with the caret machine learning package

summary(clf_LR_Full)
```

```
##
## Call:
## NULL
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.8732  -0.6344  -0.3526   0.5918   2.2488
##
## Coefficients:
##                             Estimate Std. Error z value Pr(>|z|)
## (Intercept)               -1.090e+01  1.516e+00  -7.190 6.48e-13 ***
## pregnancies                1.310e-01  6.738e-02   1.944  0.05192 .
## glucoseConcentration       4.080e-02  6.740e-03   6.054 1.41e-09 ***
## bloodPressure              2.284e-03  1.367e-02   0.167  0.86732
## skinThickness             -3.725e-06  2.118e-02   0.000  0.99986
## insulin                   -1.764e-03  1.544e-03  -1.142  0.25325
## bmi                        9.483e-02  3.472e-02   2.731  0.00631 **
## diabetesPedigreeFunction   1.472e+00  4.967e-01   2.963  0.00305 **
## age                        2.227e-02  2.099e-02   1.061  0.28881
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 374.27  on 293  degrees of freedom
## Residual deviance: 251.84  on 285  degrees of freedom
## AIC: 269.84
##
## Number of Fisher Scoring iterations: 5
```

0.4.3.2 Model Evaluation on Test Set

```
predict_LR_Full <- predict(clf_LR_Full, type = 'response', newdata = test)

with(test,
     table(hasDiabetes, predict_LR_Full > 0.3))
```

```
##
## hasDiabetes FALSE TRUE
##           0    46   20
##           1     7   25
```

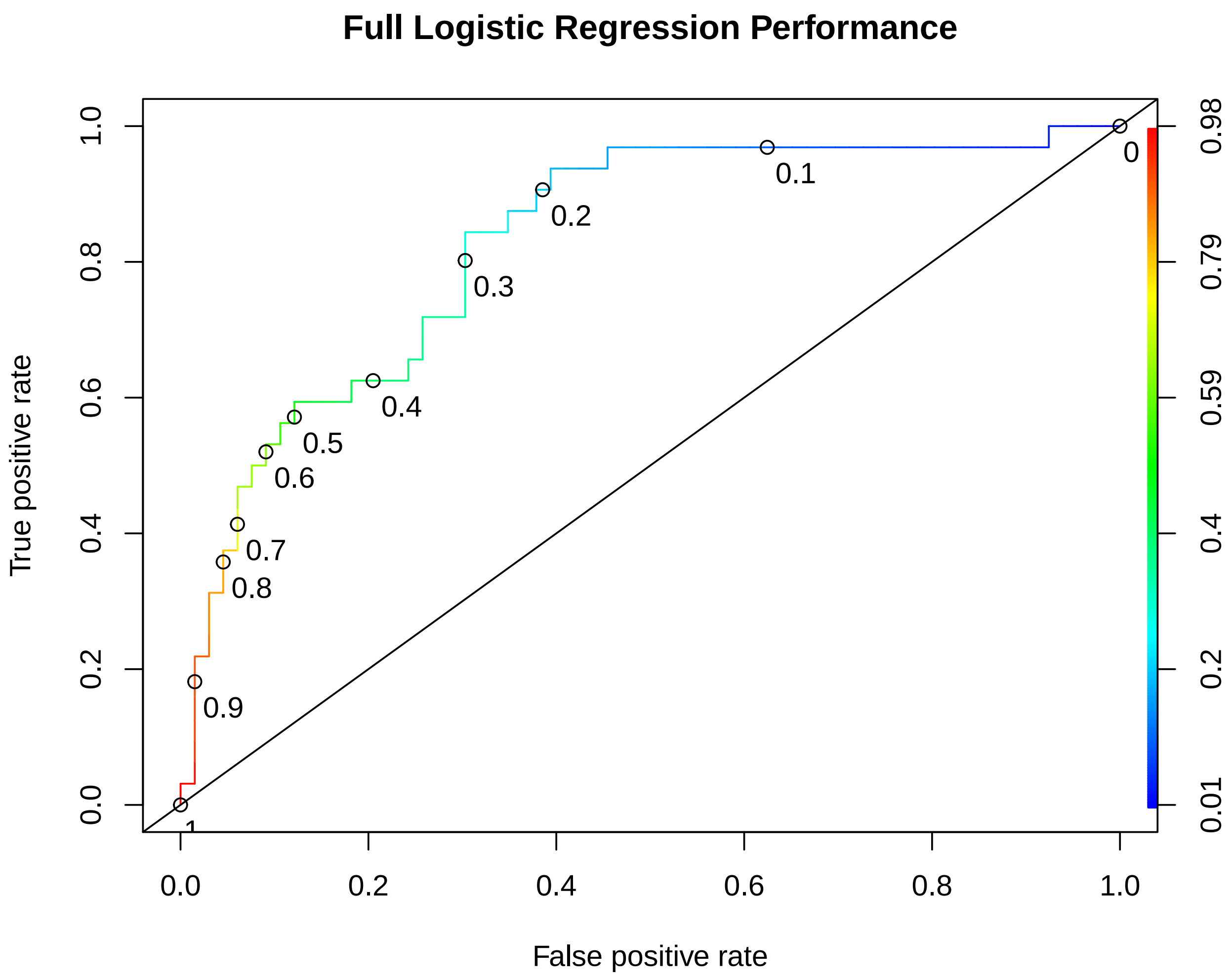
We can see in the confusion matrix that our accuracy is $(46 + 25)/(46 + 20 + 7 + 25) = 0.7245$. This is with a cutoff of 0.3.

0.4.3.3 Model Performance Visualization

```
ROCRpred_LR_Full <- with(test,
                 prediction(predict_LR_Full, hasDiabetes))

ROCRperf_LR_Full <- performance(ROCRpred_LR_Full, 'tpr','fpr')

plot(ROCRperf_LR_Full,
     colorize = TRUE,
     print.cutoffs.at = seq(0,1,0.1),
     text.adj = c(-0.2,1.7),
     main = "Full Logistic Regression Performance"); abline(0,1)
```



This model performed similarly to the simple model. The ROC curve looks a little more desirable for the full model, but as we have mentioned, the full model has some variables that we do not want to include in the final model.

Let's have a look at the overall importance of all our varibles derived from the significance values from the glm output (scaled to add up to 100).

```
varImp(clf_LR_Full, scale = TRUE) %>%
  rownames_to_column("Variable") %>%
  arrange(desc(Overall)) %>%
  mutate(Importance = Overall/sum(Overall)*100)
```

```
##                   Variable      Overall   Importance
## 1     glucoseConcentration 6.0540015916 37.690242765
## 2 diabetesPedigreeFunction 2.9630522472 18.446998540
## 3                      bmi 2.7311698103 17.003373987
## 4              pregnancies 1.9438255309 12.101624858
## 5                  insulin 1.1424876745  7.112756275
## 6                      age 1.0607464317  6.603861912
## 7            bloodPressure 0.1670576402  1.040046475
## 8            skinThickness 0.0001759146  0.001095187
```

From here we get an idea of what variables are the most predictive of the probability of having diabetes. Their seems to be tiers of importance. First, glucoseConcentration is significantly more important than the other variables. After, comes the pedigree function score and the bmi. This is good news for us because bmi is easy to calculate. Then comes number of pregnancies. And after that there are some weaker variables, the weakest of all being skinThickness, which adds next to no predictive ability to the model. Let's look at how we can use these results to optimize our model.

0.4.4 Optimized Classifier

In this step we optimize the logistic regression to include only the variables that are useful in predicting the risk of diabetes for our observations.

0.4.4.1 Training

By performing a stepwise training we will be able to get the model that maximizes the AIC score and balances the number of predictors that are included.

```
clf_LR_Op <- MASS::stepAIC(glm(hasDiabetes ~ 1, family = binomial(link='logit'),
                     data=train),
                   list(upper = ~ pregnancies +
                          glucoseConcentration +
                          bloodPressure +
                          skinThickness +
                          insulin +
                          bmi +
                          diabetesPedigreeFunction +
                          age),
                   direction="both",
                   trace = FALSE)
summary(clf_LR_Op)
```

```
##
## Call:
## glm(formula = hasDiabetes ~ glucoseConcentration + bmi + diabetesPedigreeFunction +
##     pregnancies, family = binomial(link = "logit"), data = train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.1901  -0.6546  -0.3663   0.6179   2.3511
##
## Coefficients:
```

```
##                           Estimate Std. Error z value Pr(>|z|)
## (Intercept)              -10.069964   1.258507  -8.002 1.23e-15 ***
## glucoseConcentration       0.038765   0.005836   6.642 3.09e-11 ***
## bmi                        0.090308   0.024662   3.662 0.000250 ***
## diabetesPedigreeFunction   1.495673   0.486638   3.073 0.002116 **
## pregnancies                0.179393   0.049746   3.606 0.000311 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 374.27  on 293  degrees of freedom
## Residual deviance: 254.61  on 289  degrees of freedom
## AIC: 264.61
##
## Number of Fisher Scoring iterations: 5
```

Here is the path that the model training took to arrive to that selection of variables.

```r
with(clf_LR_Op, anova)
```

```
## Stepwise Model Path
## Analysis of Deviance Table
##
## Initial Model:
## hasDiabetes ~ 1
##
## Final Model:
## hasDiabetes ~ glucoseConcentration + bmi + diabetesPedigreeFunction +
##     pregnancies
##
##
##                          Step Df  Deviance Resid. Df Resid. Dev      AIC
## 1                                                293   374.2703 376.2703
## 2      + glucoseConcentration  1 86.007138       292   288.2632 292.2632
## 3                       + bmi  1 11.098825       291   277.1644 283.1644
## 4                       + age  1 12.336045       290   264.8283 272.8283
## 5 + diabetesPedigreeFunction  1  7.809291       289   257.0190 267.0190
## 6               + pregnancies  1  3.809478       288   253.2096 265.2096
## 7                       - age  1  1.395952       289   254.6055 264.6055
```

Note that the model added age, but then after adding pregnancies it took a step backwards and eliminated age. This confirms our initial thought that either age or pregnancies would make it to the final model but not both. It turns out that pregnancies is a better predictor for the probability of having diabetes.

0.4.4.2 Model Evaluation on Train Set

```r
predict_LR_Op <- predict(clf_LR_Op, type = 'response', newdata = test)

with(test,
     table(hasDiabetes, predict_LR_Op > 0.3))
```

```
##
## hasDiabetes FALSE TRUE
##           0    47   19
##           1     6   26
```

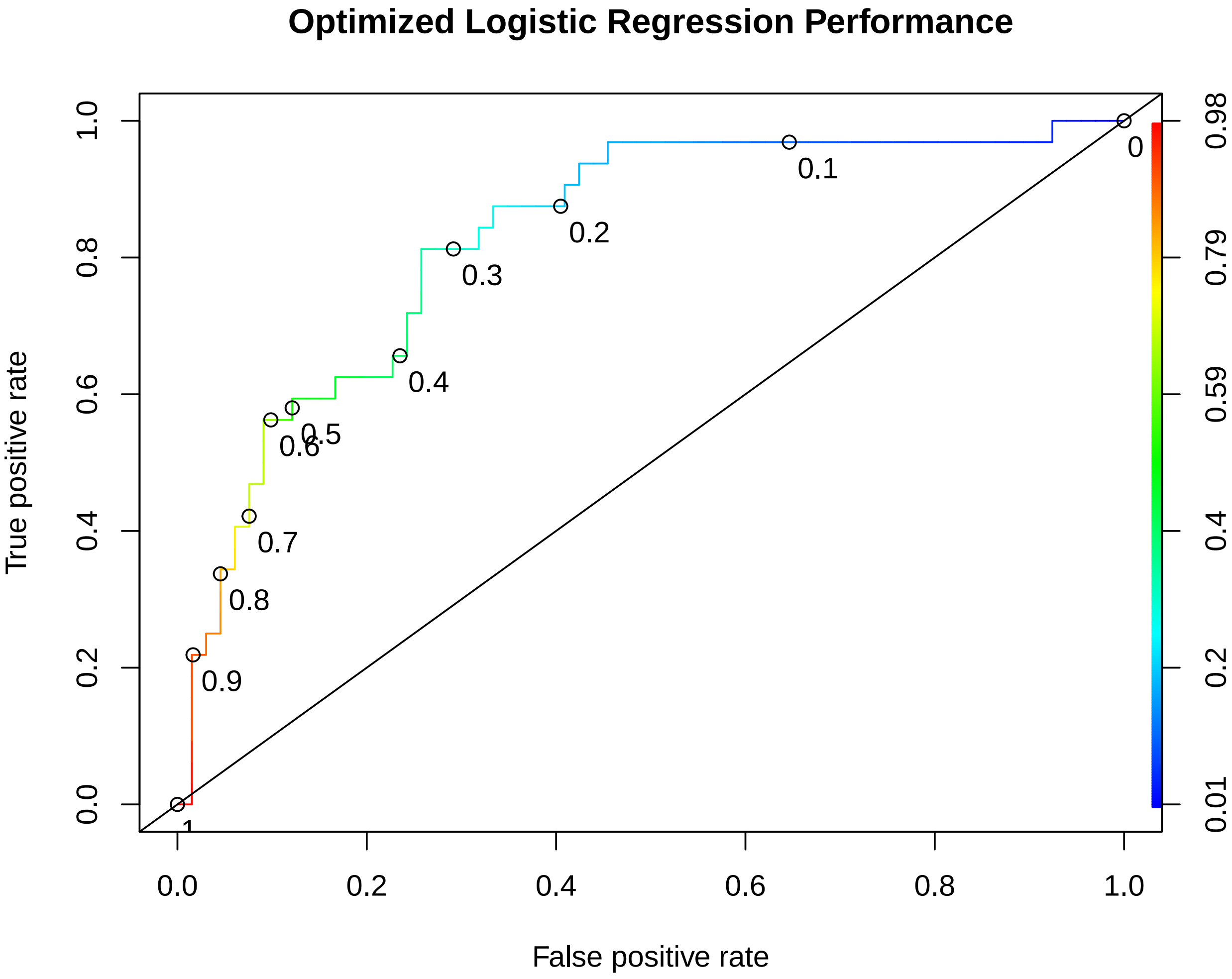
We can see in the confusion matrix that our accuracy is $(47 + 26)/(47 + 26 + 6 + 19) = 0.7449$. This is with a cutoff of 0.3.

0.4.4.3 Model Performance Visualization

```
ROCRpred_LR_Op <- with(test,
                 prediction(predict_LR_Op, hasDiabetes))

ROCRperf_LR_Op <- performance(ROCRpred_LR_Op, 'tpr','fpr')

plot(ROCRperf_LR_Op,
     colorize = TRUE,
     print.cutoffs.at = seq(0,1,0.1),
     text.adj = c(-0.2,1.7),
     main = "Optimized Logistic Regression Performance"); abline(0,1)
```



In the ROC curve we can see that our model is also good (the curve is away from the diagonal). This model has an accuracy of 0.7449.

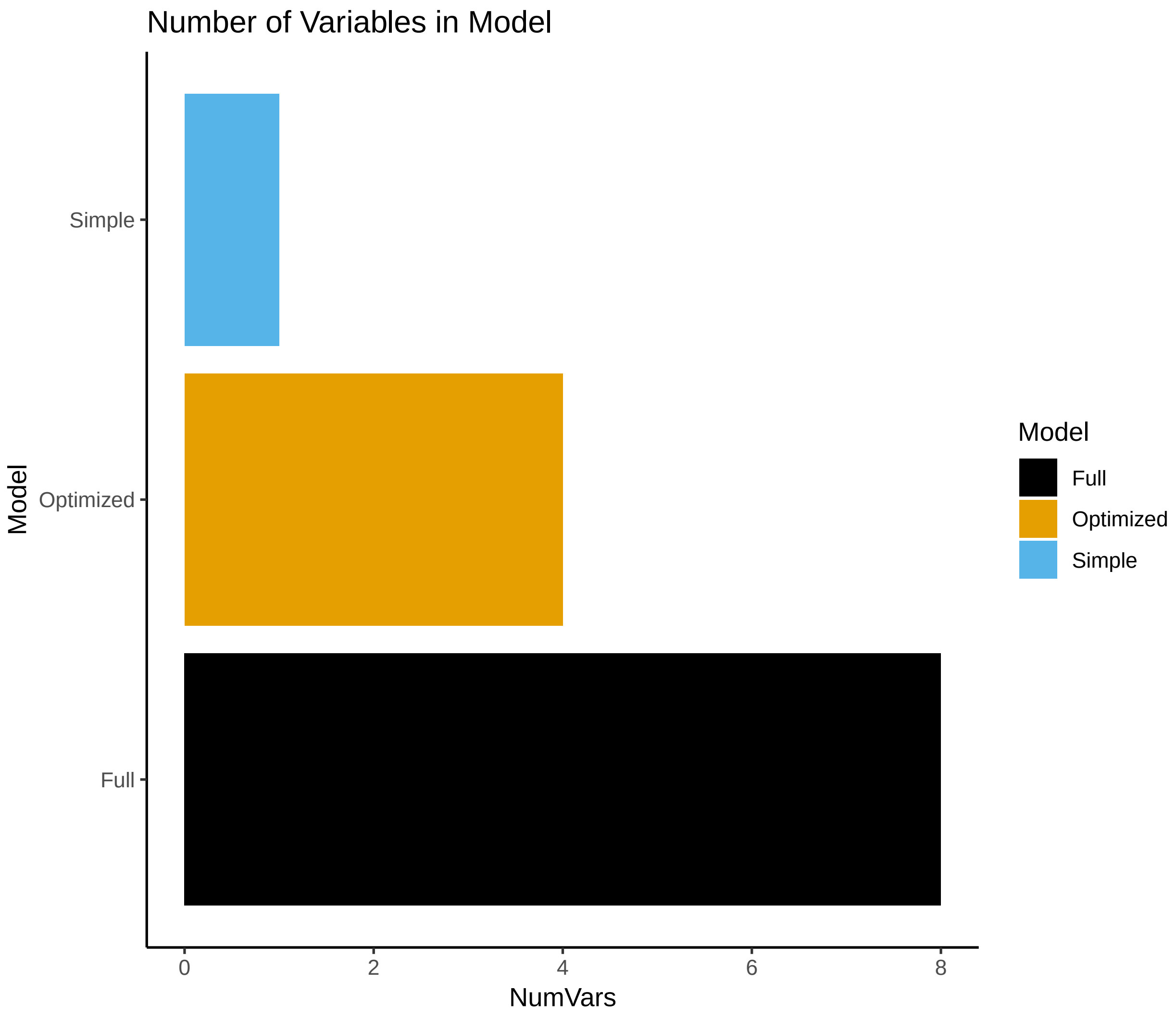
In the repository you can look at other things we tried out that performed similarly to the logistic regressions, but don't have the advantage of being as simple.

0.5 Results

We have created three models. The summary statistics for those three models are the following:

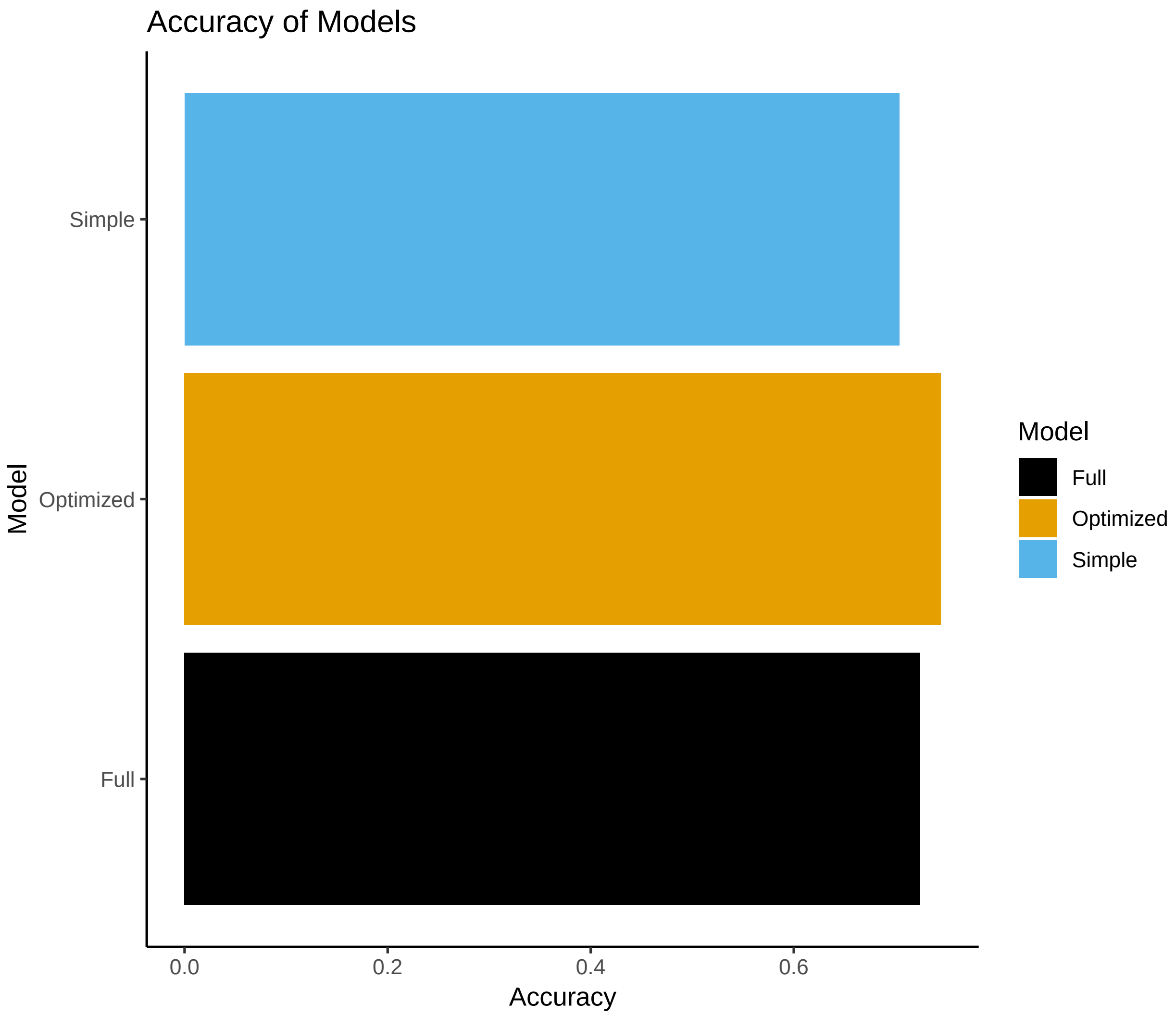
```
Results <- tibble(Model = factor(c("Simple", "Full", "Optimized")),
       AIC = c(clf_LR_Simple$aic, clf_LR_Full$aic, clf_LR_Op$aic),
       Accuracy = c(0.7041, 0.7245, 0.7449),
       FalseNegativeRate = c(10/(10+22), 7/(7+25), 6/(6+26)),
       NumVars = c(1, 8, 4))
```

```
ggplot(Results, aes(x = Model, y = NumVars, fill = Model)) +
  geom_bar(stat = "identity") +
  labs(title = "Number of Variables in Model") +
  coord_flip() +
  scale_fill_colorblind()
```



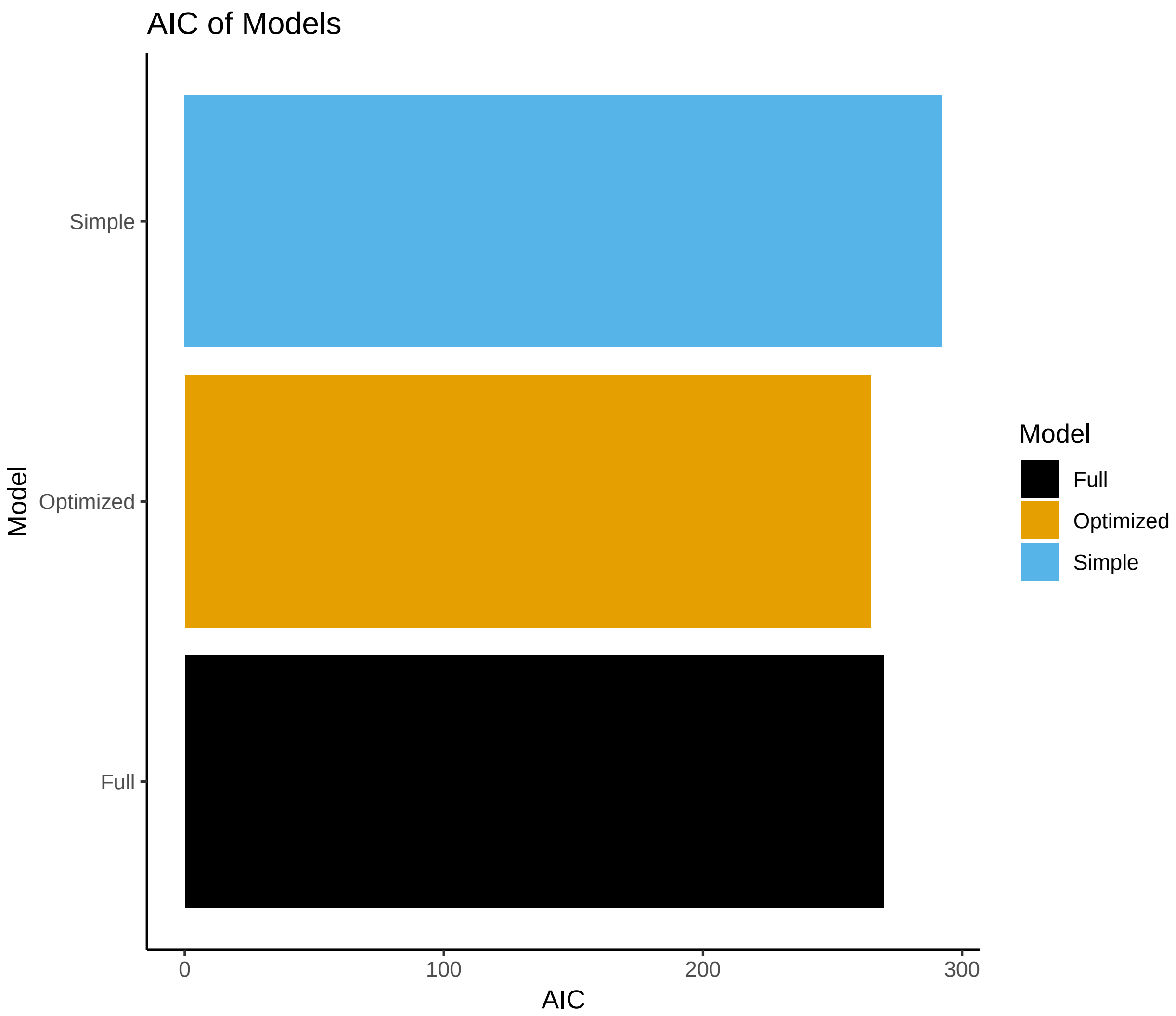
These are the three models that we are comparing. Notice that the number of variables adds complexity to the model, and it becomes harder for individuals to use it to predict the risk of diabetes. In this regard, the simple model has its advantages, but that is why we create the optimized model, because it has parts of the advantages of being simple and still being a good model for predicting probability of having diabetes.

```
ggplot(Results, aes(x = Model, y = Accuracy, fill = Model)) +
  geom_bar(stat = "identity") +
  labs(title = "Accuracy of Models") +
  coord_flip() +
  scale_fill_colorblind()
```



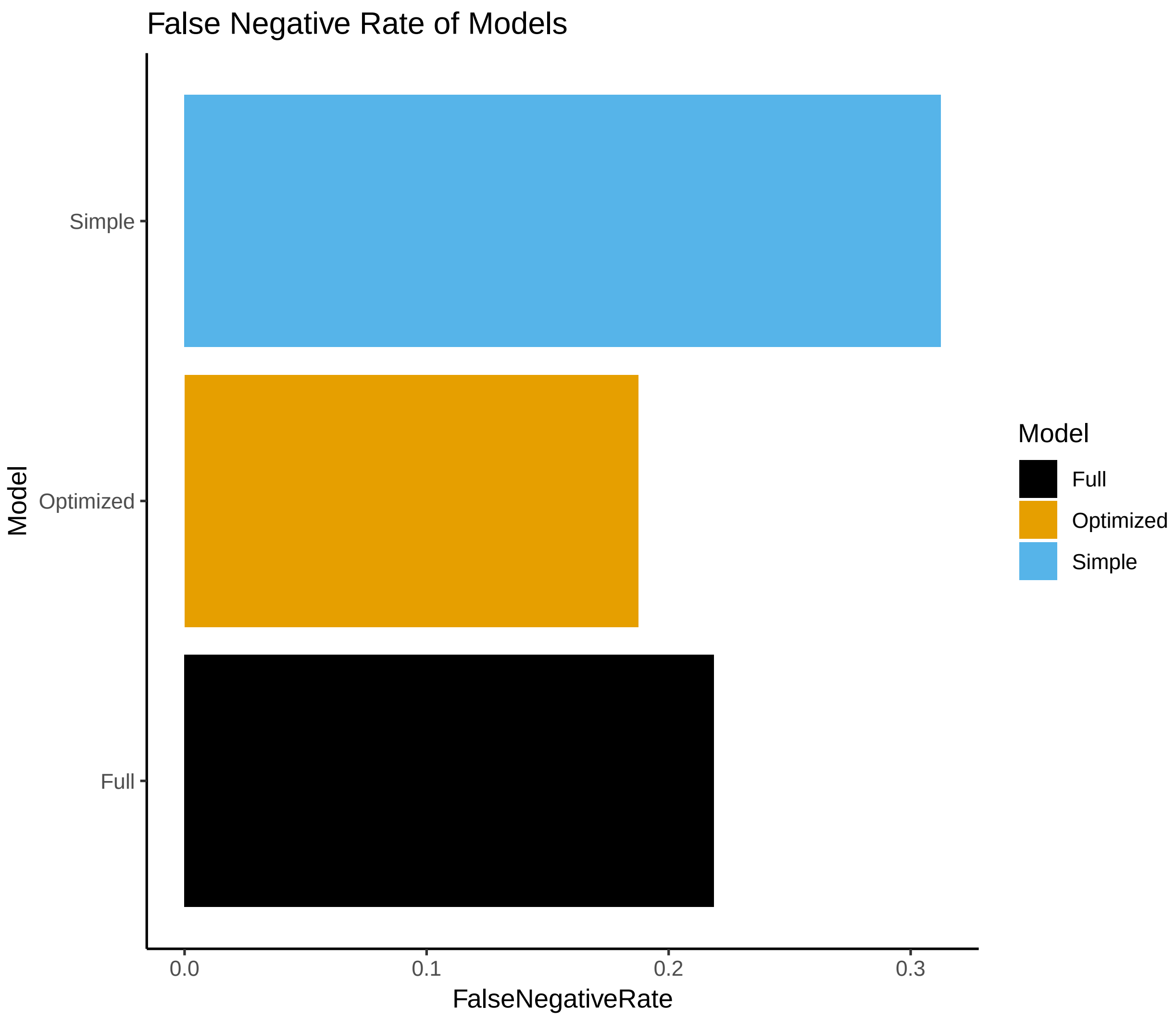
Our most accurate model was the optimized model. However, note that the accuracies for all the models are similar.

```
ggplot(Results, aes(x = Model, y = AIC, fill = Model)) +
  geom_bar(stat = "identity") +
  labs(title = "AIC of Models") +
  coord_flip() +
  scale_fill_colorblind()
```



A lower AIC score is a better score. Again, the best model is the optimized model.

```
ggplot(Results, aes(x = Model, y = FalseNegativeRate, fill = Model)) +
  geom_bar(stat = "identity") +
  labs(title = "False Negative Rate of Models") +
  coord_flip() +
  scale_fill_colorblind()
```



Here there is a slightly more sizeable difference between the performance of our best model and the simple and full model. Our optimized model has a false negative rate of 18.8% compared to 21.9% of the full model and an uncomfortable 31.2% of the Simple model.

Thus we choose the optimized model.

0.6 Conlusion

Overall, the three models created were similarly accurate, had similar scores, but the advantage of 12.4 percentage points between the false negative rates of the simple and the optimized models suggest that we should be using the optimized model. As said earlier, we would not want to misclassify the women that actually have diabetes into the category of women that don't have diabetes.

The dataset we explored was limited to 21 year old Pima Native American women. This can be generalized to the population of Pima women if we assume that the sample was representative. However, generalizing these results to Pima Native American men, or other larger American populations would be extrapolating the results.

Additionally, the fact that these variables are predictive of whether an individual has diabetes does not mean that there is a causal link between them.

Despite these comments, we offer a set of recommendations based on the findings of this study.

1) Encourage physical activity. Create programs that involve multiple sessions of physical engagement in order to maintain a healthy bmi.
2) Promote the idea of knowing your family's medical history, if other members of your family have diabetes then your risk of diabetes is going to be higher.
3) Distribute educational materials showing the relationship between the number of pregnancies and the risk of having diabetes.
4) Invest in blood/saliva glucose monitors and allow people free access to them. (Blood glucose and saliva glucose levels have been shown to be strongly correlated).

0.7 Appendix A: Installation Instructions

To install the required packages you need to run the following command:

```
install.packages(c(
  "dplyr",
  "tidyr",
  "caTools",
  "mosaic",
  "ggplot2",
  "ggcorrplot",
  "tibble",
  "MASS",
  "ROCR",
  "caret",
  "car",
  "ggthemes"
))
```

To run this command you can paste it to your R console. You can also remove those that are already installed if you know what you are doing.

0.8 Appendix B: Data Preprocessing

First the file is read in.

```
PIMA <- read.csv("https://pmatheson.people.amherst.edu/Pima.dat", header = FALSE)
```

Variable names are then assigned.

```
names(PIMA) <- c("pregnancies", "glucoseConcentration",
                    "bloodPressure", "skinThickness", "insulin",
                    "bmi", "diabetesPedigreeFunction", "age", "hasDiabetes")
```

Any variable type corrections are done next.

```
PIMA <- PIMA %>%
  mutate(hasDiabetes = as.factor(hasDiabetes))
```

The data at this point would have missing data, but since it is coded as 0 we have to mark it. This is done in this step.

```
PIMA <- PIMA %>%
  mutate(glucoseConcentration = ifelse(glucoseConcentration == 0,
                                       NA_integer_, glucoseConcentration)) %>%
  mutate(bloodPressure = ifelse(bloodPressure == 0, NA_integer_, bloodPressure)) %>%
  mutate(skinThickness = ifelse(skinThickness == 0, NA_integer_, skinThickness)) %>%
  mutate(insulin = ifelse(insulin == 0, NA_integer_, insulin)) %>%
  mutate(bmi = ifelse(bmi == 0, NA_integer_, bmi))
```

Once we have marked the missing data, since we don't perform any imputation, we drop those observations.

```
PIMA <- PIMA %>%
  drop_na()
```

Finally, in order to prepare for our modeling section, we split our data set and creat our train and our test sets.

```r
set.seed(100)
split <- with(PIMA,
              sample.split(hasDiabetes, SplitRatio = 0.75))

train <- subset(PIMA, split == TRUE)
test <- subset(PIMA, split == FALSE)
```

0.9 Appendix C: Checking Model Assumptions

In this appendix we verify that the assumptions of logistic regression are met.

First, we can safely assume that the observations are independent.

Second, we check if there is multicollinearity.

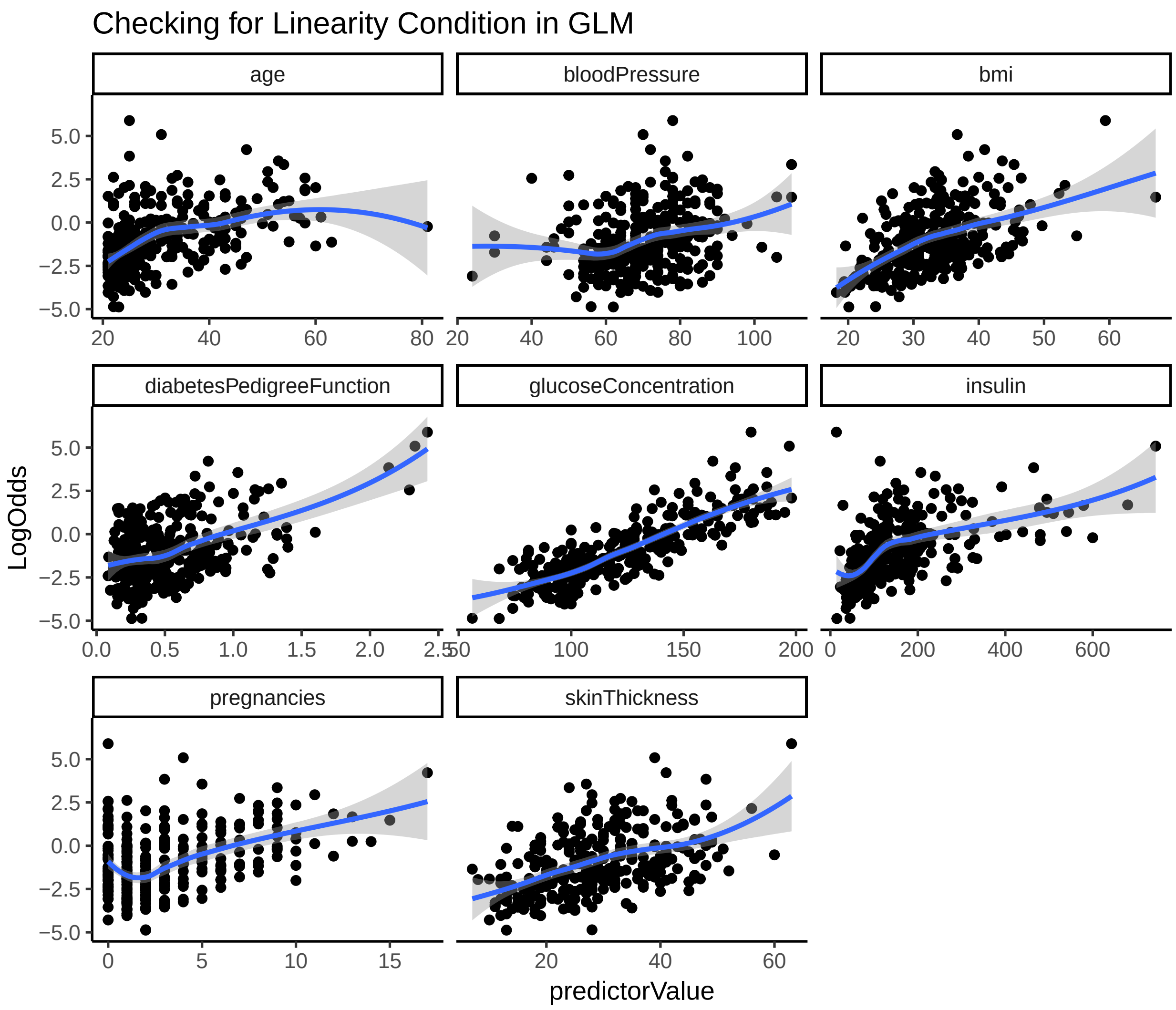
```r
vif(clf_LR_Op)
```

```
##     glucoseConcentration                      bmi diabetesPedigreeFunction 
##                 1.014944                 1.083773                 1.040180 
##              pregnancies 
##                 1.106276 
```

In general a vif value of 5 is indicative of a problem with collinearity. All the predictors have vif values less than 5.

Lastly, we check if the predictors have a linear relationship to the logit of the outcome's probability (this is the log odds of the outcome's probability).

```r
train %>%
  mutate(LogOdds = predict(clf_LR_Op)) %>%
  select(-hasDiabetes) %>%
  gather(key = "predictor", value = "predictorValue", -LogOdds) %>%
  ggplot(aes(x = predictorValue, y = LogOdds)) + geom_point() +
    facet_wrap(~predictor, scales = "free_x") +
    geom_smooth(method = "loess") +
    labs(title = "Checking for Linearity Condition in GLM")
```



We can see that all the variables are linearly related to the log odds of the outcomes. Thus this condition is satisfied.

Lastly, we have a large enough sample size.

Hence, all conditions are satistfied for our logistic regression model.