DOUGLAS COLLEGE

Diabetes Dataset Analysis

CSIS4290: Special Topics in Data Analytics

Instructor: Parissa Ahmadi Abkenari

Group Members:

‘Aafrin Zahid Memon

Akintunde Akinro

Bruno do Nascimento Beserra

DOUGLASCOLLEGE

Douglas College respectfully acknowledges that our campuses are located on the unceded traditional and ancestral lands of the Coast Salish Peoples, including the territories of the q̓íc̓əy̓ (Katzie), q’ʷa:n̓ƛ’ən̓ (Kwantlen), kʷikʷəƛ̓əm (Kwikwetlem), xʷməθkʷəy̓əm (Musqueam), and qiqéyt (Qayqayt) First Nations.

Contents

- Dataset Introduction
  - Data Visualizations
  - Exploratory Data Analysis
  - Data Cleaning
- Creating Train, Test datasets
  - Feature Selection
    - LASSO
    - RIDGE
    - Correlation
- Models Analysis
  - Decision Tree
  - Naive Bayes
  - K-Nearest Neighbors
  - Support Vector Machine
  - OLS Regression
  - GLM Regression
  - Bagging
  - XGBoost
  - Random Forest
- Models Comparison
- Research Findings
- Conclusion

Dataset Introduction

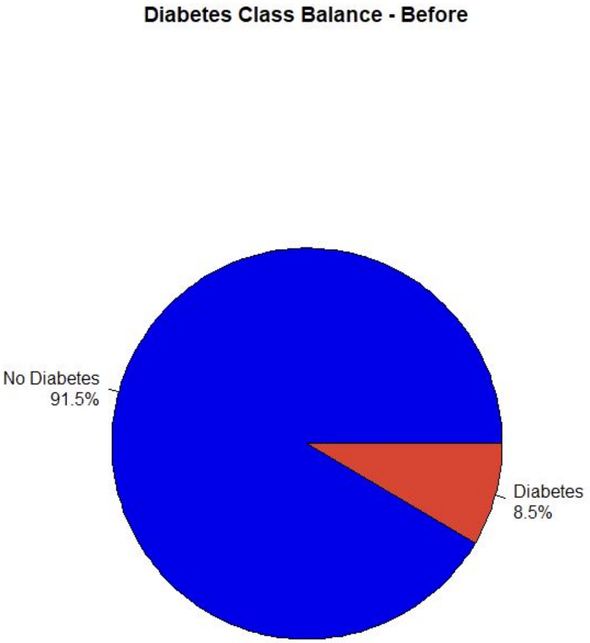
- Diabetes Dataset

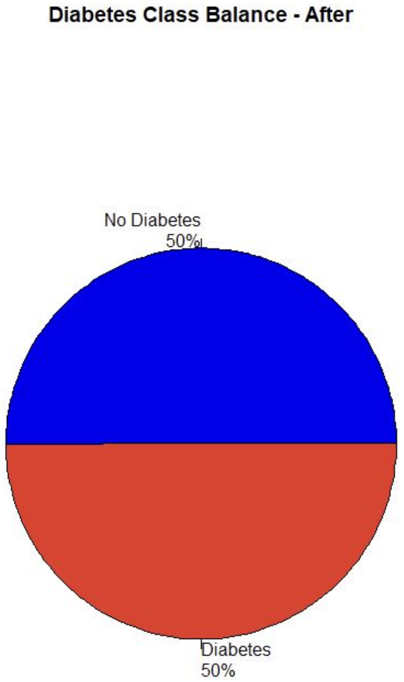
- The Diabetes dataset is a collection of medical and demographic data from patients, along with their diabetes status (positive or negative).

- The data includes features such as age, gender, body mass index (BMI), hypertension, heart disease, smoking history, HbA1c level, and blood glucose level.

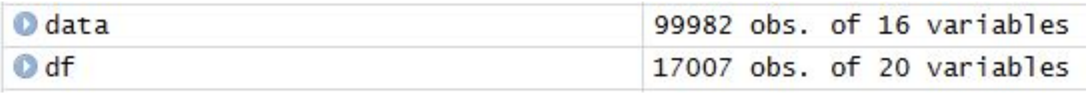
- Source: Kaggle

Data Visualization - Target Balance Comparison


Diabetes Class Balance - Before

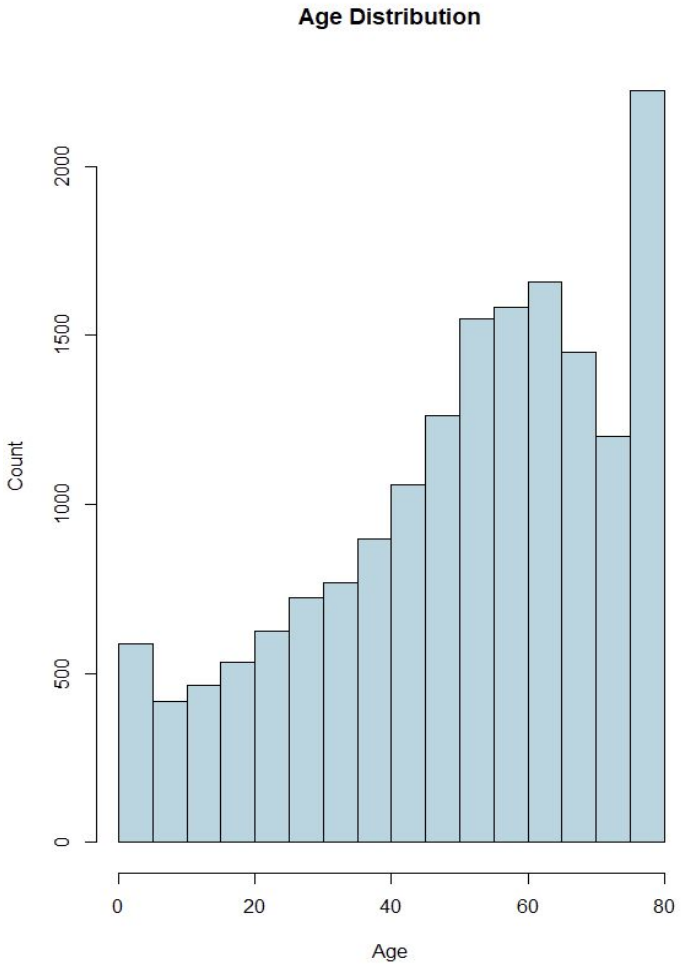

Diabetes Class Balance - After

Size of dataset was 99982 before the balancing process and 17007 after it.

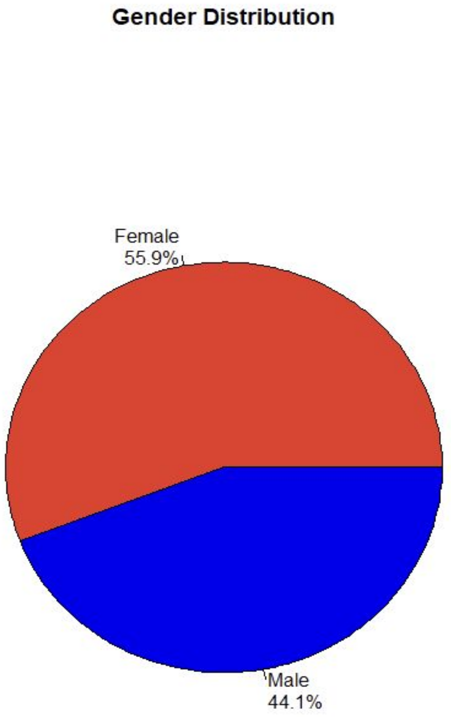


```
data    99982 obs. of 16 variables
df      17007 obs. of 20 variables
```

Data Visualization - Age and Gender Proportion

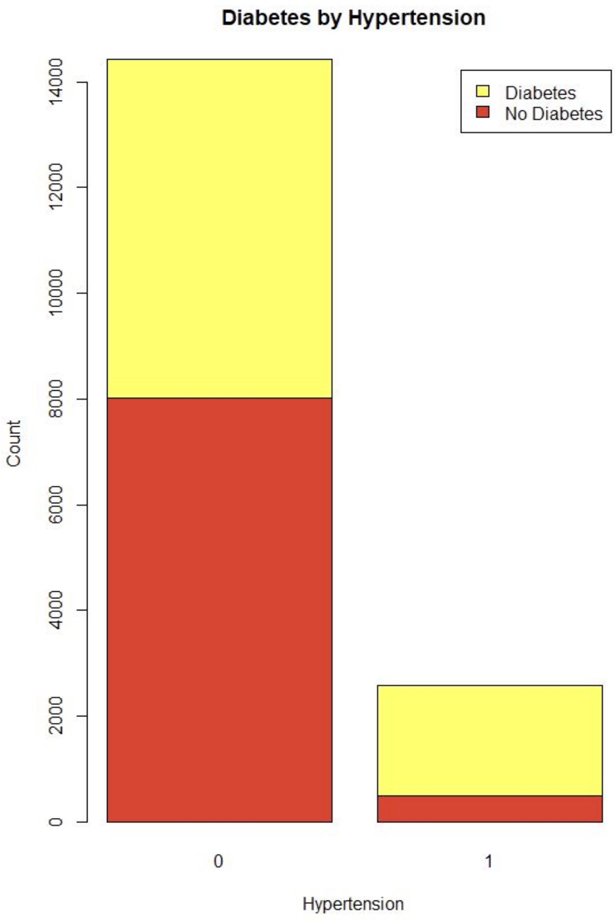


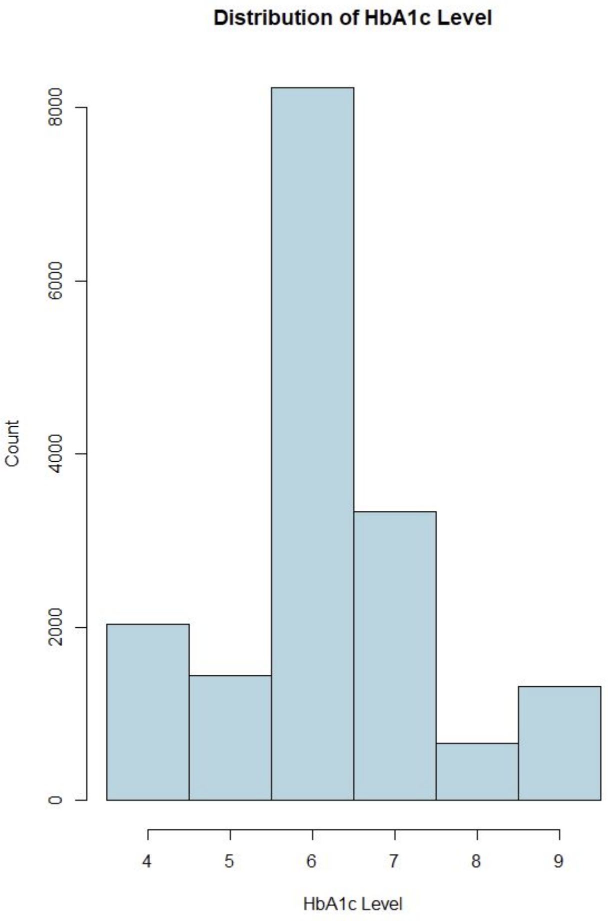
Age Distribution



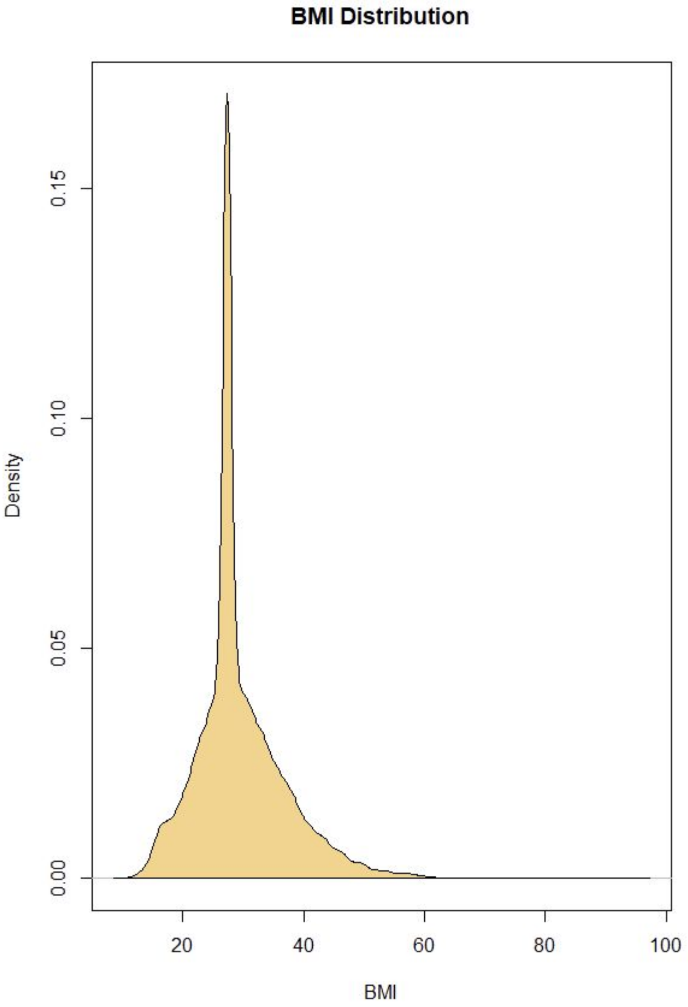
Gender Distribution

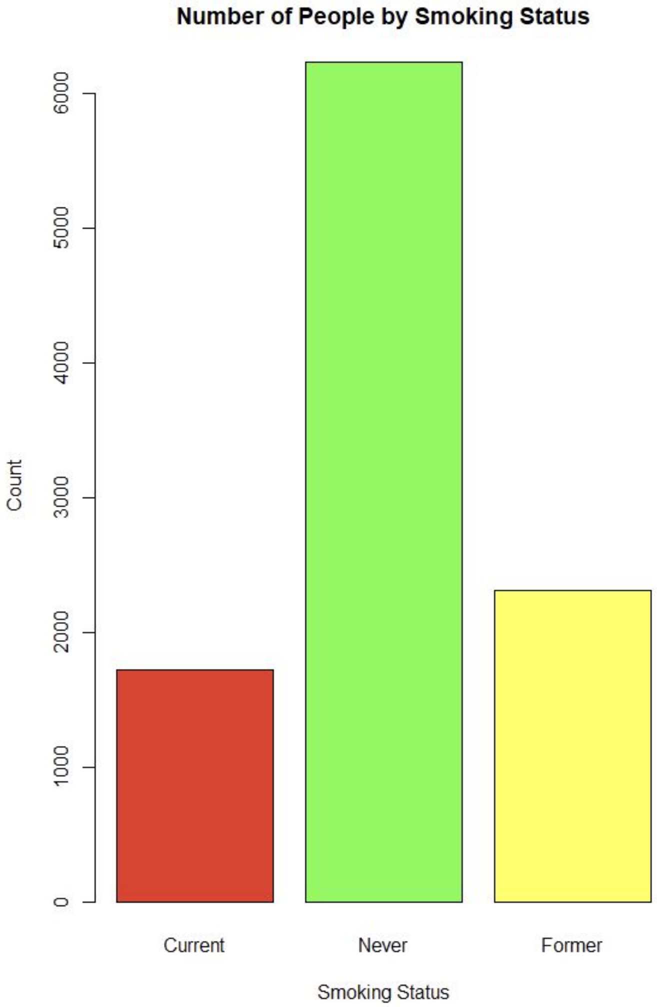
Data Visualization - Diabetes Risk Factors



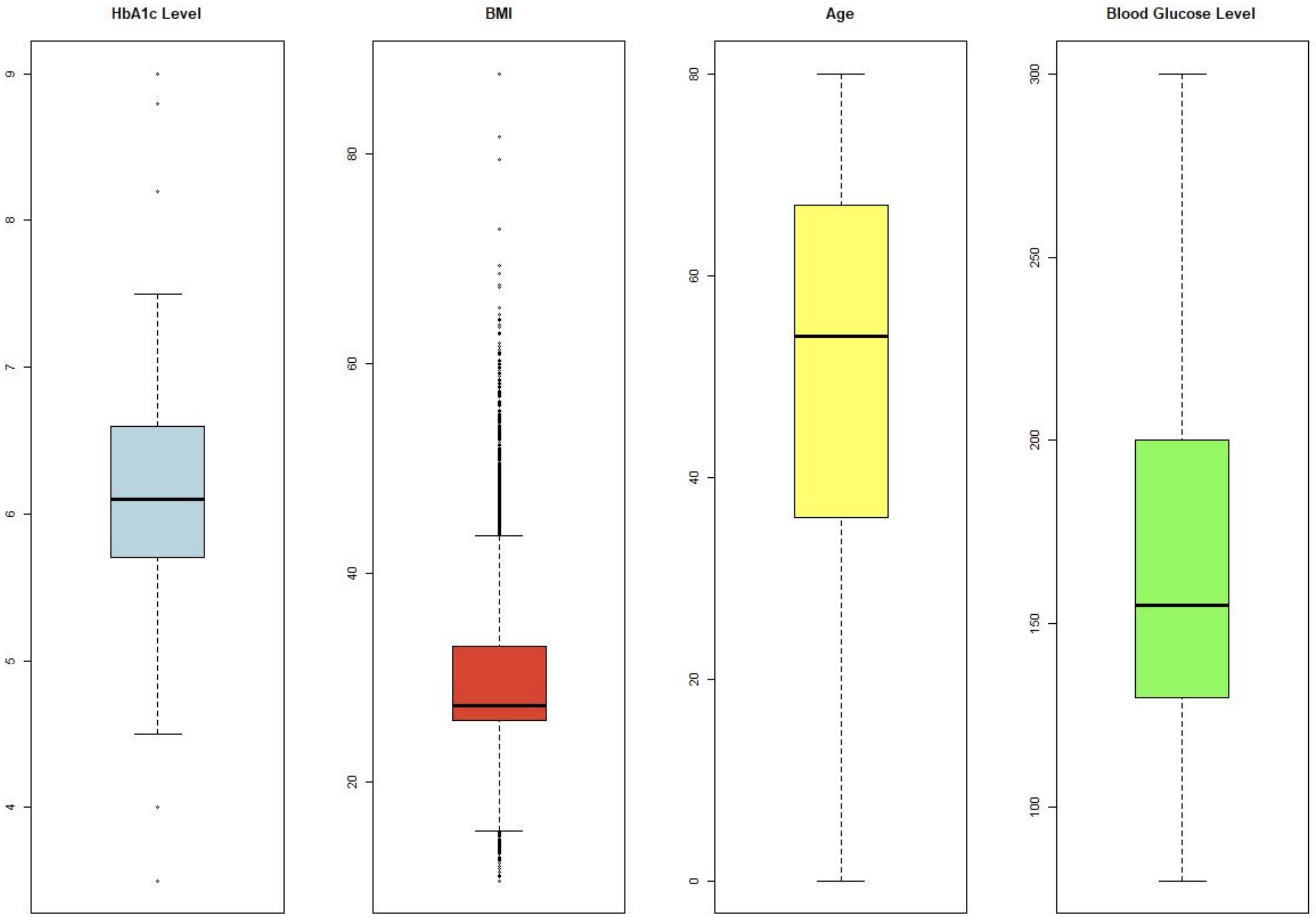


Data Visualization - BMI and Smoking Distribution

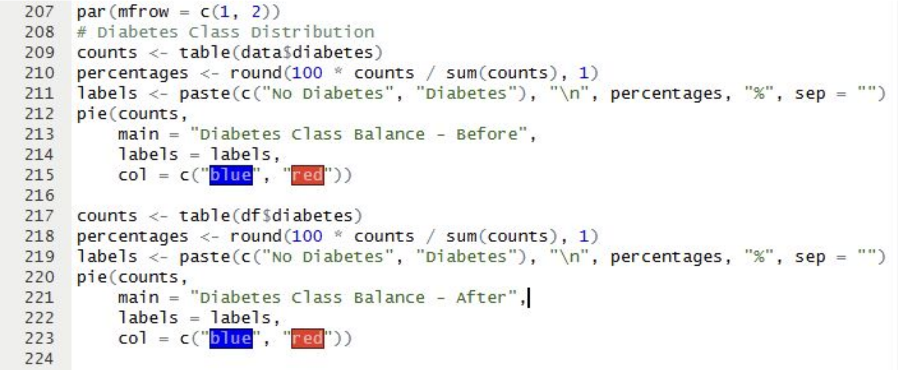

BMI Distribution


Number of People by Smoking Status

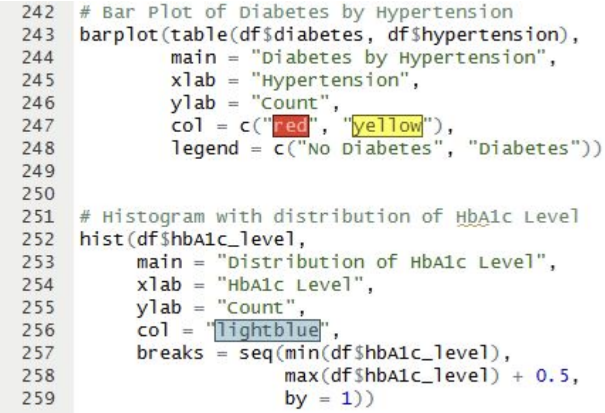
Data Visualization - Box plots



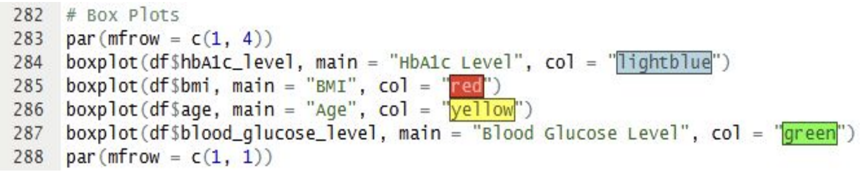
Data Visualization - Codes



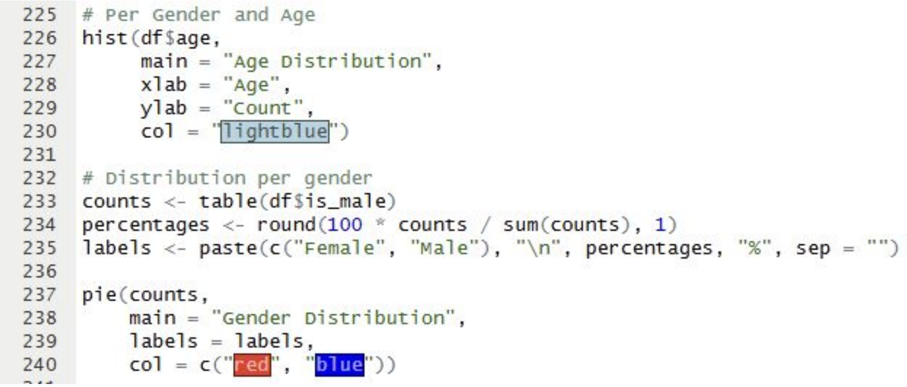
```
par(mfrow = c(1, 2))
# Diabetes Class Distribution
counts <- table(data$diabetes)
percentages <- round(100 * counts / sum(counts), 1)
labels <- paste(c("No Diabetes", "Diabetes"), "\n", percentages, "%", sep = "")
pie(counts,
    main = "Diabetes Class Balance - Before",
    labels = labels,
    col = c("blue", "red"))

counts <- table(df$diabetes)
percentages <- round(100 * counts / sum(counts), 1)
labels <- paste(c("No Diabetes", "Diabetes"), "\n", percentages, "%", sep = "")
pie(counts,
    main = "Diabetes Class Balance - After",
    labels = labels,
    col = c("blue", "red"))

```



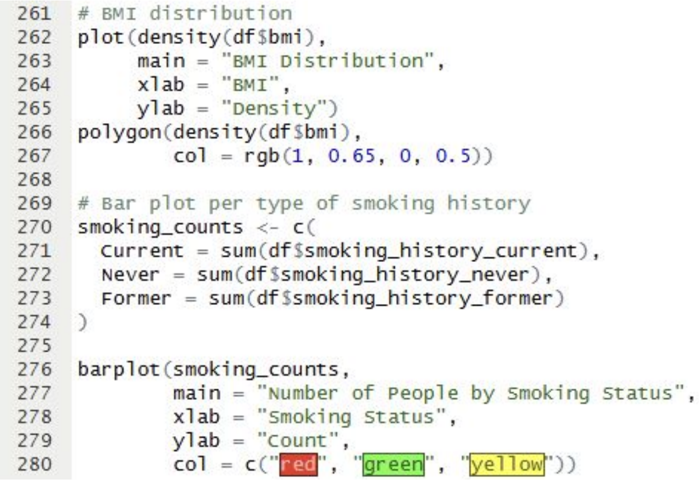
```
# Bar Plot of Diabetes by Hypertension
barplot(table(df$diabetes, df$hypertension),
        main = "Diabetes by Hypertension",
        xlab = "Hypertension",
        ylab = "Count",
        col = c("red", "yellow"),
        legend = c("No Diabetes", "Diabetes"))


# Histogram with distribution of HbA1c Level
hist(df$hbA1c_level,
     main = "Distribution of HbA1c Level",
     xlab = "HbA1c Level",
     ylab = "Count",
     col = "lightblue",
     breaks = seq(min(df$hbA1c_level),
                  max(df$hbA1c_level) + 0.5,
                  by = 1))
```



```
# Box Plots
par(mfrow = c(1, 4))
boxplot(df$hbA1c_level, main = "HbA1c Level", col = "lightblue")
boxplot(df$bmi, main = "BMI", col = "red")
boxplot(df$age, main = "Age", col = "yellow")
boxplot(df$blood_glucose_level, main = "Blood Glucose Level", col = "green")
par(mfrow = c(1, 1))
```



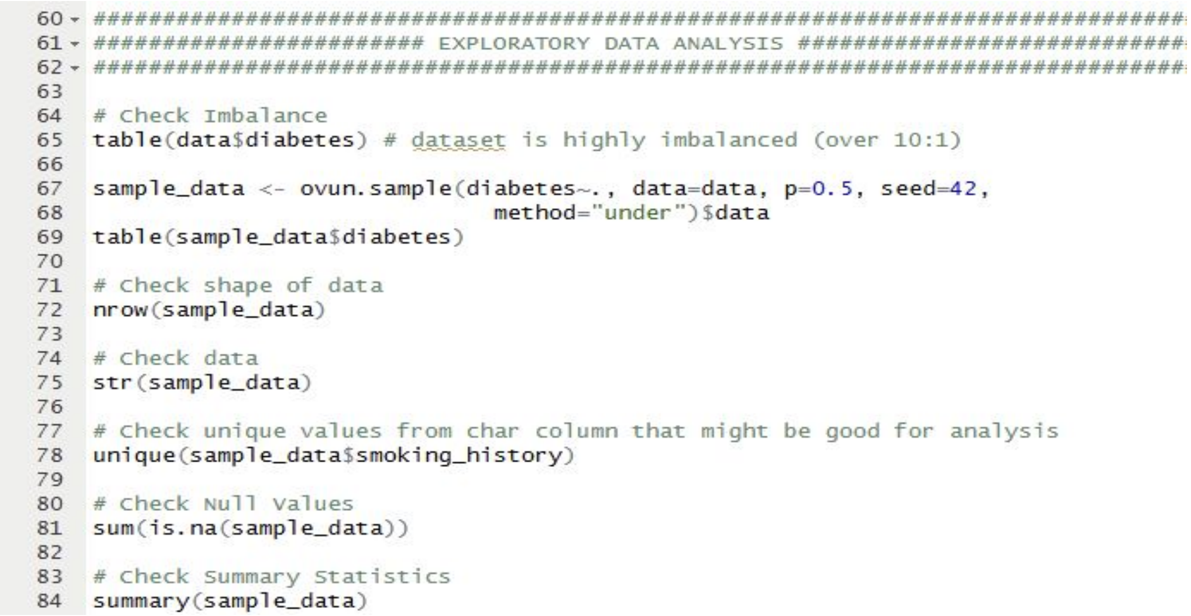
```
# Per Gender and Age
hist(df$age,
     main = "Age Distribution",
     xlab = "Age",
     ylab = "Count",
     col = "lightblue")

# Distribution per gender
counts <- table(df$is_male)
percentages <- round(100 * counts / sum(counts), 1)
labels <- paste(c("Female", "Male"), "\n", percentages, "%", sep = "")

pie(counts,
    main = "Gender Distribution",
    labels = labels,
    col = c("red", "blue"))
```



```
# BMI distribution
plot(density(df$bmi),
     main = "BMI Distribution",
     xlab = "BMI",
     ylab = "Density")
polygon(density(df$bmi),
        col = rgb(1, 0.65, 0, 0.5))

# Bar plot per type of smoking history
smoking_counts <- c(
  Current = sum(df$smoking_history_current),
  Never = sum(df$smoking_history_never),
  Former = sum(df$smoking_history_former)
)

barplot(smoking_counts,
        main = "Number of People by Smoking Status",
        xlab = "Smoking Status",
        ylab = "Count",
        col = c("red", "green", "yellow"))
```

Exploratory Data Analysis

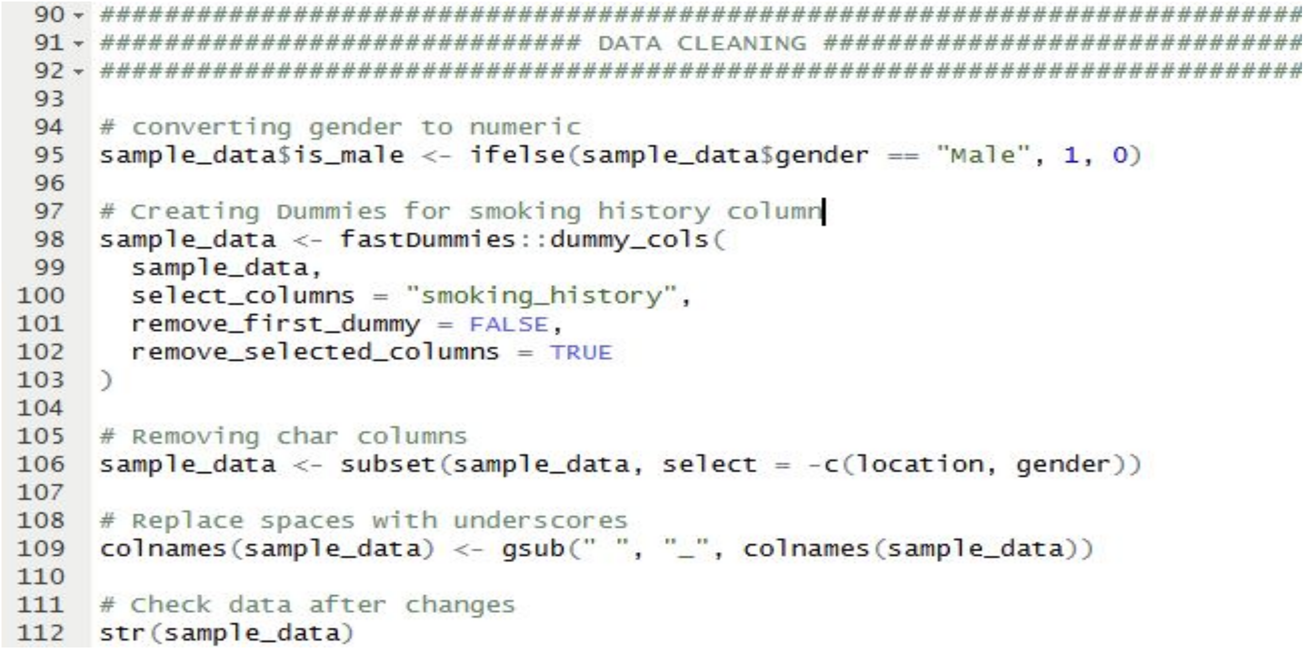
- Check Imbalance in dataset, shape, values from attributes, statistics, and missing values.



```
###############################################################################
######################## EXPLORATORY DATA ANALYSIS ############################
###############################################################################

# Check Imbalance
table(data$diabetes) # dataset is highly imbalanced (over 10:1)

sample_data <- ovun.sample(diabetes~., data=data, p=0.5, seed=42,
                              method="under")$data
table(sample_data$diabetes)

# Check shape of data
nrow(sample_data)

# Check data
str(sample_data)

# Check unique values from char column that might be good for analysis
unique(sample_data$smoking_history)

# Check Null Values
sum(is.na(sample_data))

# Check Summary Statistics
summary(sample_data)
```

Data Cleaning

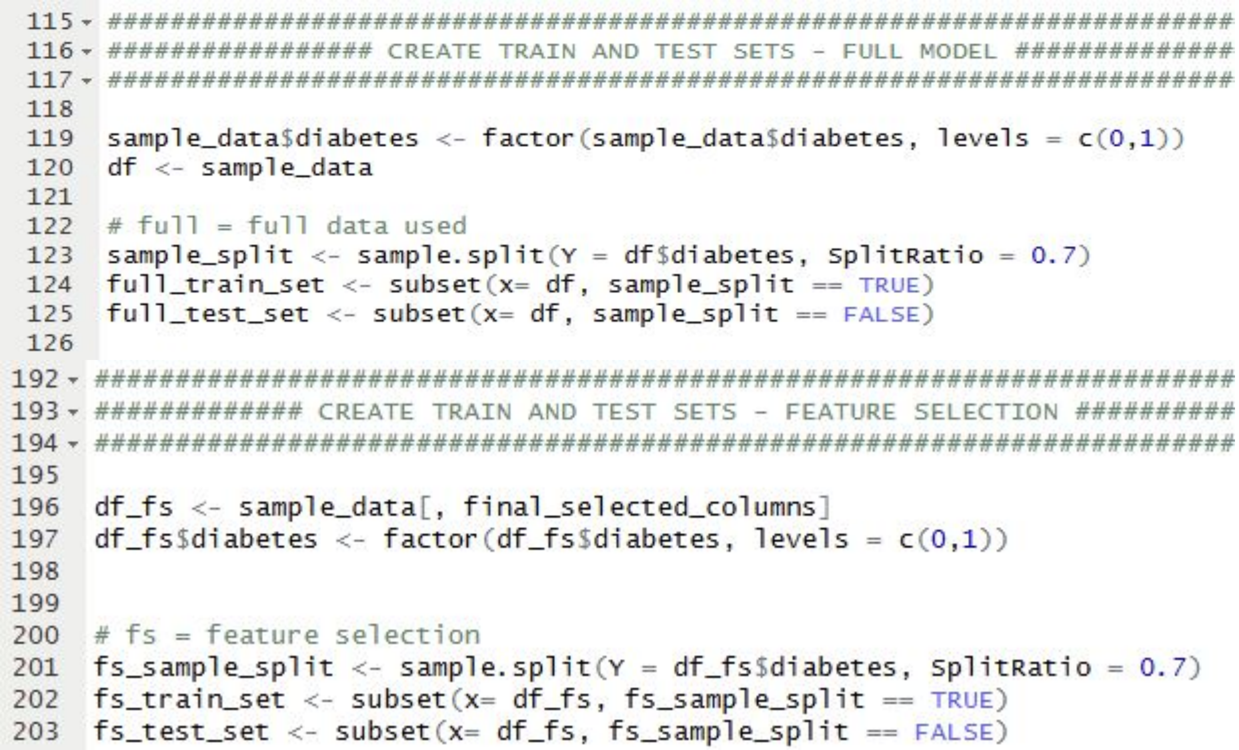
- Converted gender to numeric, created dummies for smoking history, remove char columns, and fixed naming convention from attributes.



```r
###########################################################################
############################## DATA CLEANING ##############################
###########################################################################

# converting gender to numeric
sample_data$is_male <- ifelse(sample_data$gender == "Male", 1, 0)

# Creating Dummies for smoking history column
sample_data <- fastDummies::dummy_cols(
  sample_data,
  select_columns = "smoking_history",
  remove_first_dummy = FALSE,
  remove_selected_columns = TRUE
)

# Removing char columns
sample_data <- subset(sample_data, select = -c(location, gender))

# Replace spaces with underscores
colnames(sample_data) <- gsub(" ", "_", colnames(sample_data))

# Check data after changes
str(sample_data)
```

Creating Train, Test datasets

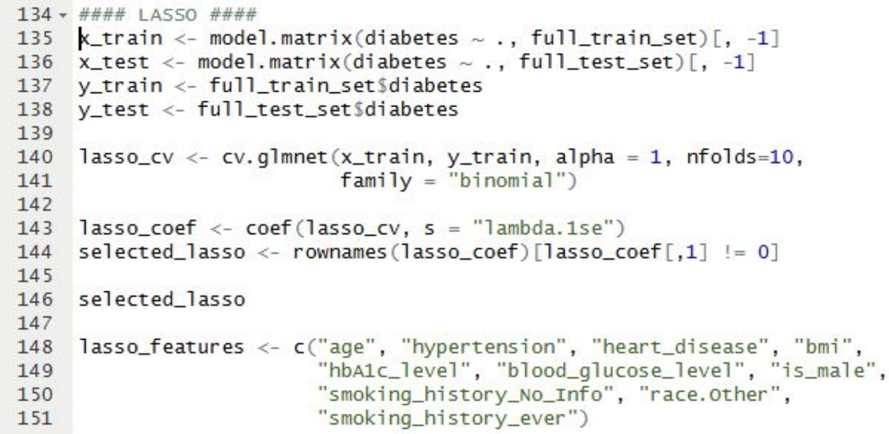
- We used two sets in our project, one with all attributes named ‘full’ and one with the selected features named ‘fs’. Both with ratio 70:30.



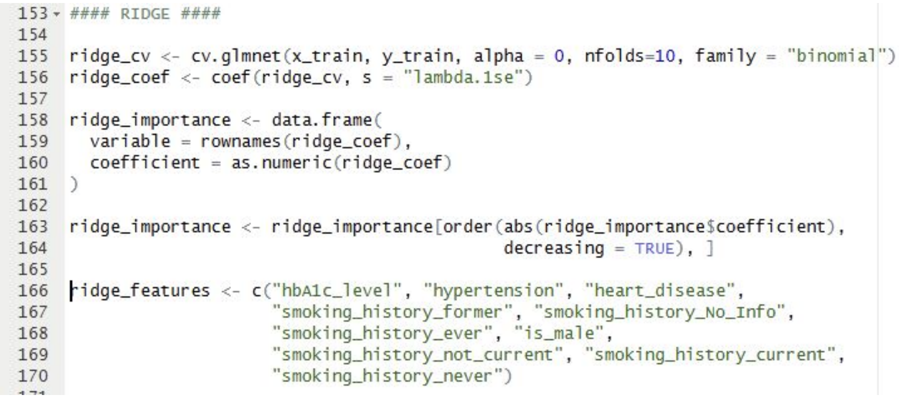
```
########################################################################
################# CREATE TRAIN AND TEST SETS - FULL MODEL ##############
########################################################################

sample_data$diabetes <- factor(sample_data$diabetes, levels = c(0,1))
df <- sample_data

# full = full data used
sample_split <- sample.split(Y = df$diabetes, SplitRatio = 0.7)
full_train_set <- subset(x= df, sample_split == TRUE)
full_test_set <- subset(x= df, sample_split == FALSE)

########################################################################
############# CREATE TRAIN AND TEST SETS - FEATURE SELECTION ##########
########################################################################

df_fs <- sample_data[, final_selected_columns]
df_fs$diabetes <- factor(df_fs$diabetes, levels = c(0,1))


# fs = feature selection
fs_sample_split <- sample.split(Y = df_fs$diabetes, SplitRatio = 0.7)
fs_train_set <- subset(x= df_fs, fs_sample_split == TRUE)
fs_test_set <- subset(x= df_fs, fs_sample_split == FALSE)
```

Feature Selection

- Used to reduce overfitting, and handle multicollinearity
- Used Correlation, LASSO and RIDGE Techniques to decide the better attributes to keep in our model.



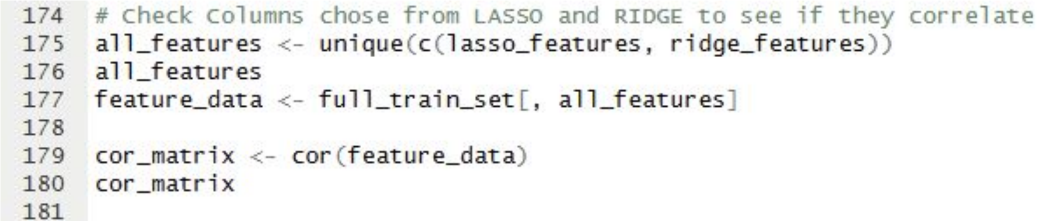
```r
#### LASSO ####
x_train <- model.matrix(diabetes ~ ., full_train_set)[, -1]
x_test <- model.matrix(diabetes ~ ., full_test_set)[, -1]
y_train <- full_train_set$diabetes
y_test <- full_test_set$diabetes

lasso_cv <- cv.glmnet(x_train, y_train, alpha = 1, nfolds=10,
                      family = "binomial")

lasso_coef <- coef(lasso_cv, s = "lambda.1se")
selected_lasso <- rownames(lasso_coef)[lasso_coef[,1] != 0]

selected_lasso

lasso_features <- c("age", "hypertension", "heart_disease", "bmi",
                    "hbA1c_level", "blood_glucose_level", "is_male",
                    "smoking_history_No_Info", "race.Other",
                    "smoking_history_ever")
```



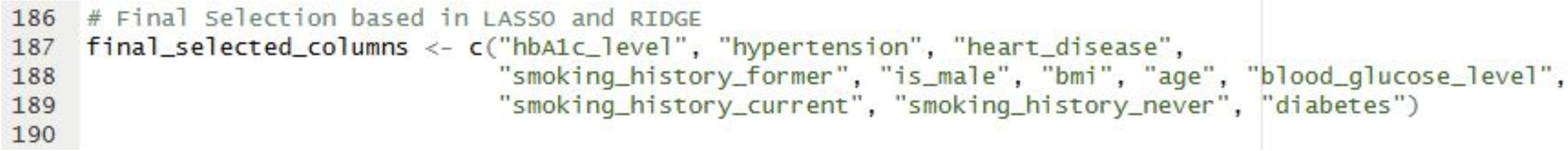
```r
#### RIDGE ####

ridge_cv <- cv.glmnet(x_train, y_train, alpha = 0, nfolds=10, family = "binomial")
ridge_coef <- coef(ridge_cv, s = "lambda.1se")

ridge_importance <- data.frame(
  variable = rownames(ridge_coef),
  coefficient = as.numeric(ridge_coef)
)

ridge_importance <- ridge_importance[order(abs(ridge_importance$coefficient),
                                           decreasing = TRUE), ]

ridge_features <- c("hbA1c_level", "hypertension", "heart_disease",
                    "smoking_history_former", "smoking_history_No_Info",
                    "smoking_history_ever", "is_male",
                    "smoking_history_not_current", "smoking_history_current",
                    "smoking_history_never")
```

Feature Selection

- We did the correlation between attributes selected from LASSO and RIDGE.
- None of the attributes presented correlation higher than 47%, hence all of them could be added.
- but for our study doesn't make sense consider some of the columns even though they could be added.

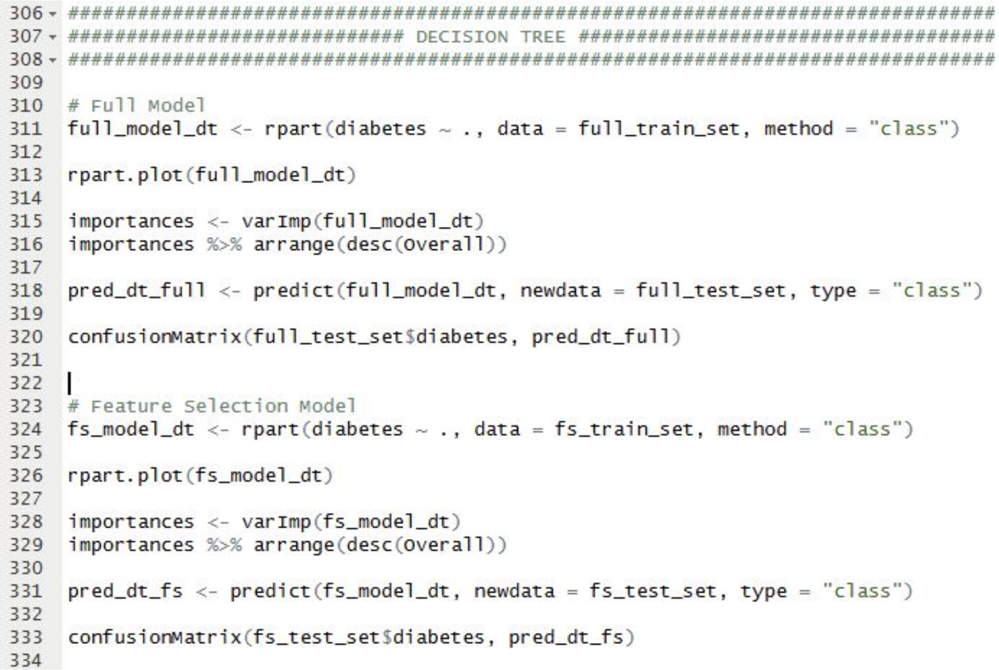


```
# Check Columns chose from LASSO and RIDGE to see if they correlate
all_features <- unique(c(lasso_features, ridge_features))
all_features
feature_data <- full_train_set[, all_features]

cor_matrix <- cor(feature_data)
cor_matrix

```

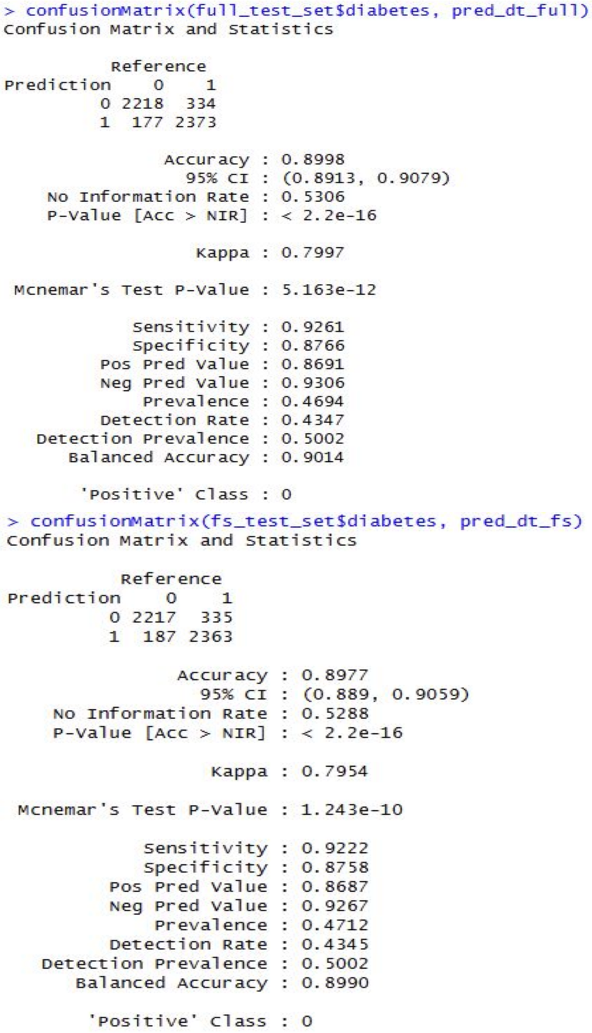


```
# Final Selection based in LASSO and RIDGE
final_selected_columns <- c("hbA1c_level", "hypertension", "heart_disease",
                            "smoking_history_former", "is_male", "bmi", "age", "blood_glucose_level",
                            "smoking_history_current", "smoking_history_never", "diabetes")

```

Models Analysis - Decision Tree



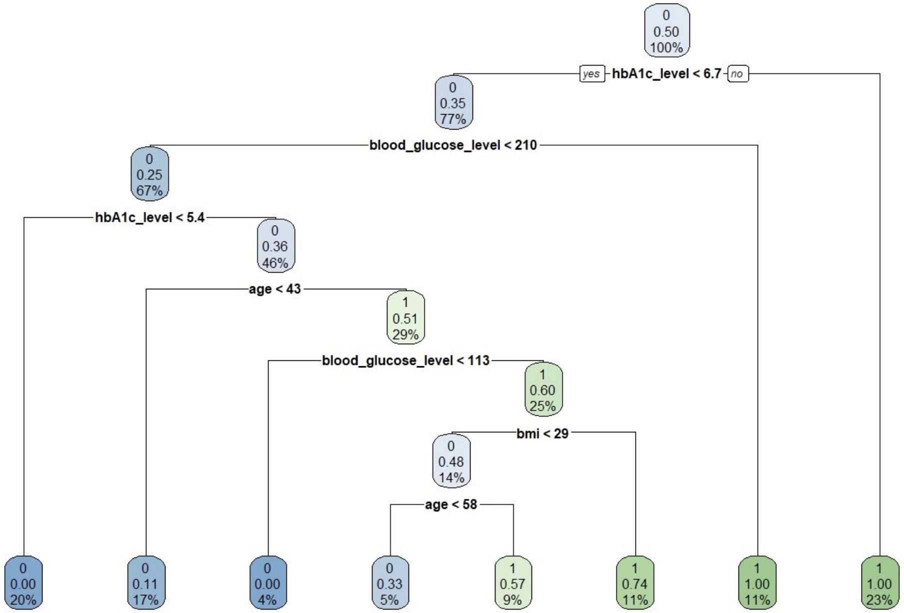
```
##########################################################################
############################ DECISION TREE ###############################
##########################################################################

# Full Model
full_model_dt <- rpart(diabetes ~ ., data = full_train_set, method = "class")

rpart.plot(full_model_dt)

importances <- varImp(full_model_dt)
importances %>% arrange(desc(Overall))

pred_dt_full <- predict(full_model_dt, newdata = full_test_set, type = "class")

confusionMatrix(full_test_set$diabetes, pred_dt_full)


# Feature Selection Model
fs_model_dt <- rpart(diabetes ~ ., data = fs_train_set, method = "class")

rpart.plot(fs_model_dt)

importances <- varImp(fs_model_dt)
importances %>% arrange(desc(Overall))

pred_dt_fs <- predict(fs_model_dt, newdata = fs_test_set, type = "class")

confusionMatrix(fs_test_set$diabetes, pred_dt_fs)

```



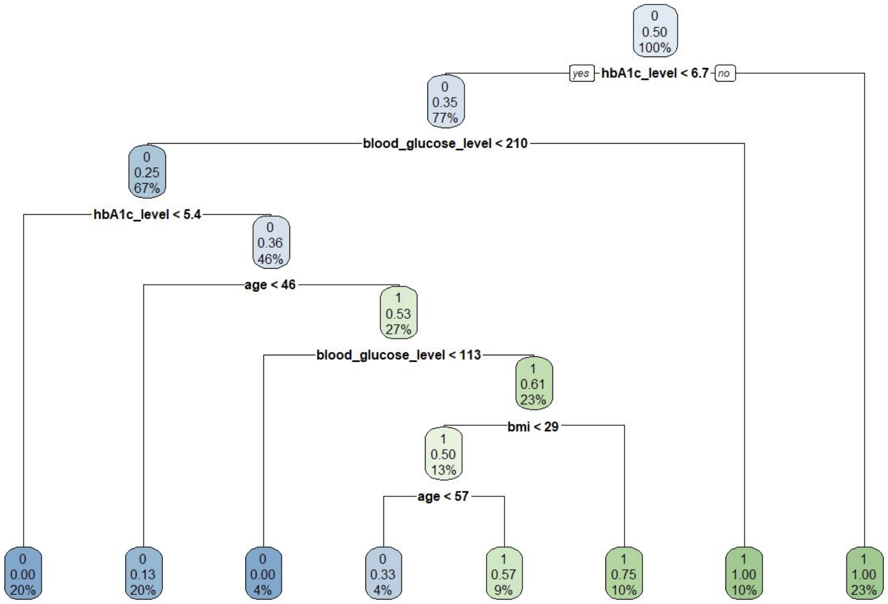
```
> confusionMatrix(full_test_set$diabetes, pred_dt_full)
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2218  334
         1  177 2373

               Accuracy : 0.8998
                 95% CI : (0.8913, 0.9079)
    No Information Rate : 0.5306
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7997

 Mcnemar's Test P-Value : 5.163e-12

            Sensitivity : 0.9261
            Specificity : 0.8766
         Pos Pred Value : 0.8691
         Neg Pred Value : 0.9306
             Prevalence : 0.4694
         Detection Rate : 0.4347
   Detection Prevalence : 0.5002
      Balanced Accuracy : 0.9014

       'Positive' Class : 0
> confusionMatrix(fs_test_set$diabetes, pred_dt_fs)
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2217  335
         1  187 2363

               Accuracy : 0.8977
                 95% CI : (0.889, 0.9059)
    No Information Rate : 0.5288
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7954

 Mcnemar's Test P-Value : 1.243e-10

            Sensitivity : 0.9222
            Specificity : 0.8758
         Pos Pred Value : 0.8687
         Neg Pred Value : 0.9267
             Prevalence : 0.4712
         Detection Rate : 0.4345
   Detection Prevalence : 0.5002
      Balanced Accuracy : 0.8990

       'Positive' Class : 0
```

Models Analysis - Decision Tree

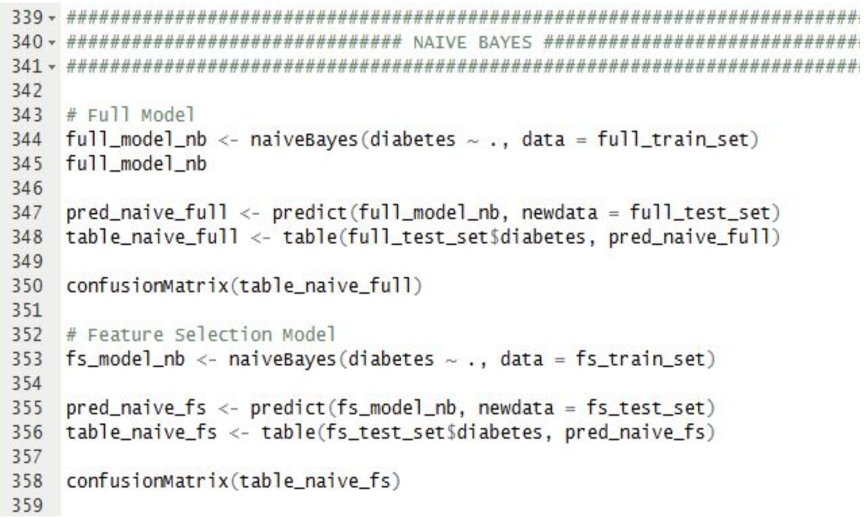
Decision Tree Full Model



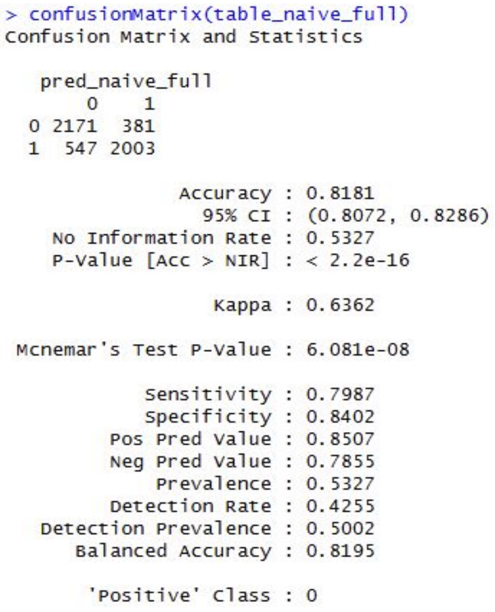
Decision Tree Feature Selection Model



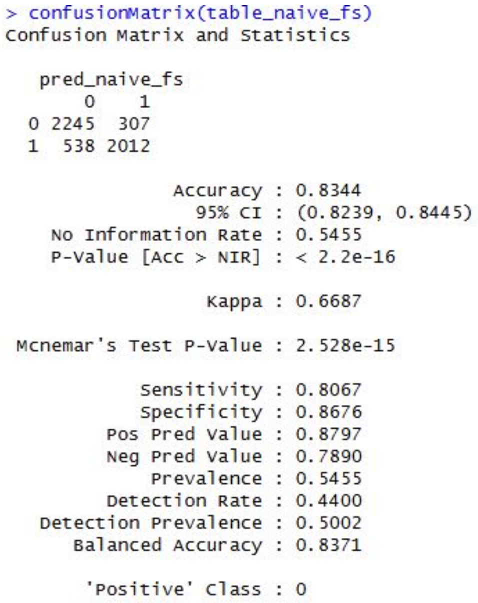
Models Analysis - Naive Bayes



```r
##########################################################################
############################### NAIVE BAYES ##############################
##########################################################################

# Full Model
full_model_nb <- naiveBayes(diabetes ~ ., data = full_train_set)
full_model_nb

pred_naive_full <- predict(full_model_nb, newdata = full_test_set)
table_naive_full <- table(full_test_set$diabetes, pred_naive_full)

confusionMatrix(table_naive_full)

# Feature Selection Model
fs_model_nb <- naiveBayes(diabetes ~ ., data = fs_train_set)

pred_naive_fs <- predict(fs_model_nb, newdata = fs_test_set)
table_naive_fs <- table(fs_test_set$diabetes, pred_naive_fs)

confusionMatrix(table_naive_fs)

```

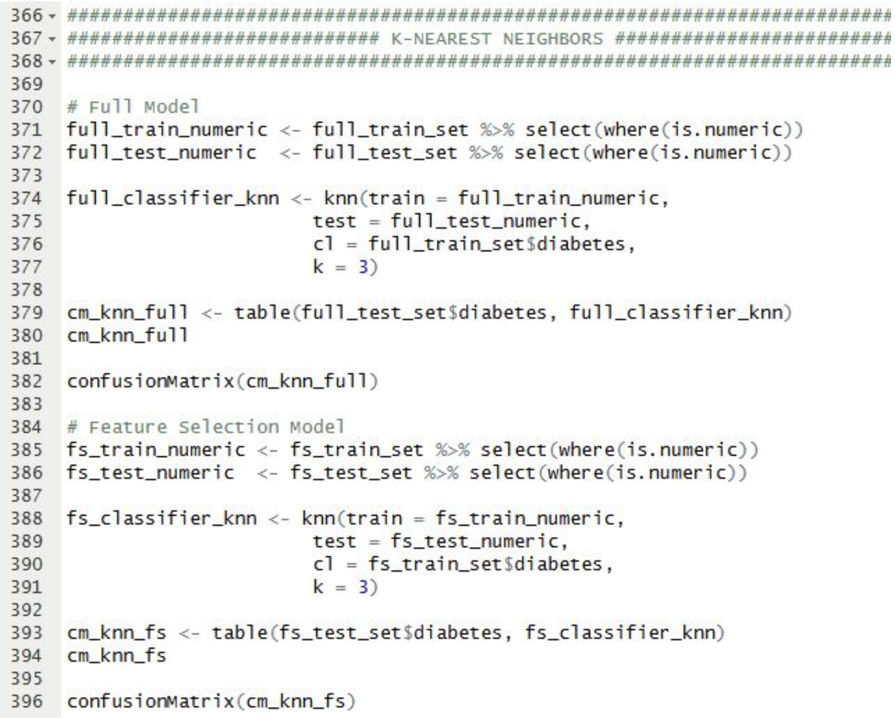


```
> confusionMatrix(table_naive_full)
Confusion Matrix and Statistics

   pred_naive_full
       0    1
  0 2171  381
  1  547 2003

               Accuracy : 0.8181
                 95% CI : (0.8072, 0.8286)
    No Information Rate : 0.5327
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6362

 Mcnemar's Test P-Value : 6.081e-08

            Sensitivity : 0.7987
            Specificity : 0.8402
         Pos Pred Value : 0.8507
         Neg Pred Value : 0.7855
             Prevalence : 0.5327
         Detection Rate : 0.4255
   Detection Prevalence : 0.5002
      Balanced Accuracy : 0.8195

       'Positive' Class : 0
```

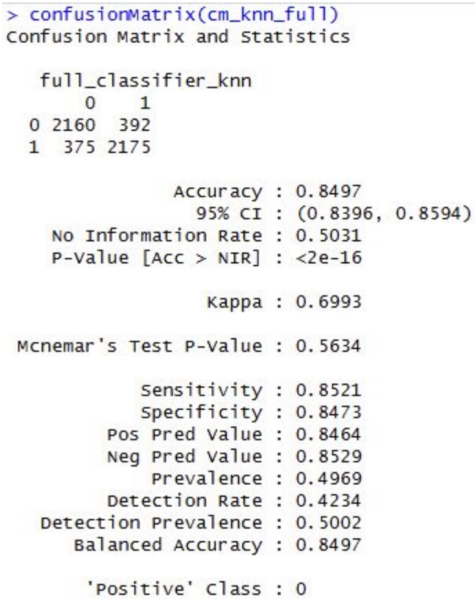


```
> confusionMatrix(table_naive_fs)
Confusion Matrix and Statistics

   pred_naive_fs
       0    1
  0 2245  307
  1  538 2012

               Accuracy : 0.8344
                 95% CI : (0.8239, 0.8445)
    No Information Rate : 0.5455
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6687

 Mcnemar's Test P-Value : 2.528e-15

            Sensitivity : 0.8067
            Specificity : 0.8676
         Pos Pred Value : 0.8797
         Neg Pred Value : 0.7890
             Prevalence : 0.5455
         Detection Rate : 0.4400
   Detection Prevalence : 0.5002
      Balanced Accuracy : 0.8371

       'Positive' Class : 0
```

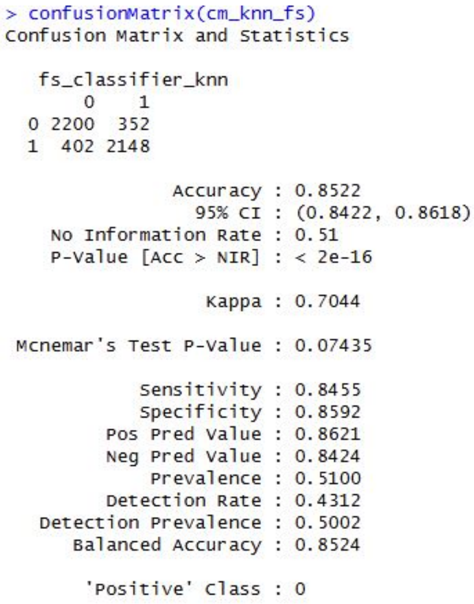
Models Analysis - K-Nearest Neighbors



```
##########################################################################
############################ K-NEAREST NEIGHBORS #########################
##########################################################################

# Full Model
full_train_numeric <- full_train_set %>% select(where(is.numeric))
full_test_numeric  <- full_test_set %>% select(where(is.numeric))

full_classifier_knn <- knn(train = full_train_numeric,
                      test = full_test_numeric,
                      cl = full_train_set$diabetes,
                      k = 3)

cm_knn_full <- table(full_test_set$diabetes, full_classifier_knn)
cm_knn_full

confusionMatrix(cm_knn_full)

# Feature Selection Model
fs_train_numeric <- fs_train_set %>% select(where(is.numeric))
fs_test_numeric  <- fs_test_set %>% select(where(is.numeric))

fs_classifier_knn <- knn(train = fs_train_numeric,
                      test = fs_test_numeric,
                      cl = fs_train_set$diabetes,
                      k = 3)

cm_knn_fs <- table(fs_test_set$diabetes, fs_classifier_knn)
cm_knn_fs

confusionMatrix(cm_knn_fs)
```

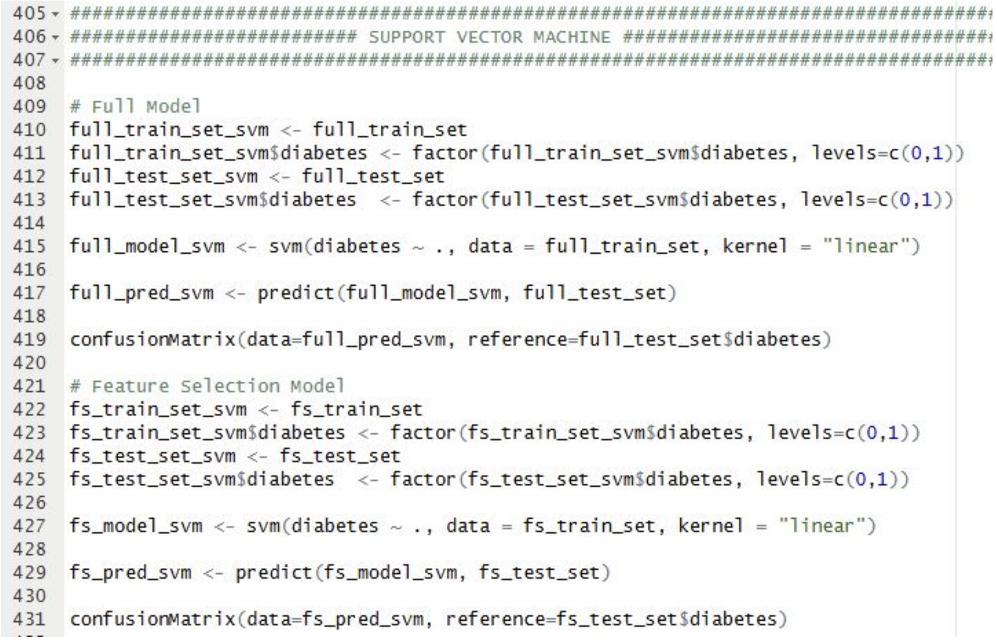


```
> confusionMatrix(cm_knn_full)
Confusion Matrix and Statistics

   full_classifier_knn
        0    1
  0  2160  392
  1   375 2175

               Accuracy : 0.8497
                 95% CI : (0.8396, 0.8594)
    No Information Rate : 0.5031
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.6993

 Mcnemar's Test P-Value : 0.5634

            Sensitivity : 0.8521
            Specificity : 0.8473
         Pos Pred Value : 0.8464
         Neg Pred Value : 0.8529
             Prevalence : 0.4969
         Detection Rate : 0.4234
   Detection Prevalence : 0.5002
      Balanced Accuracy : 0.8497

       'Positive' Class : 0
```

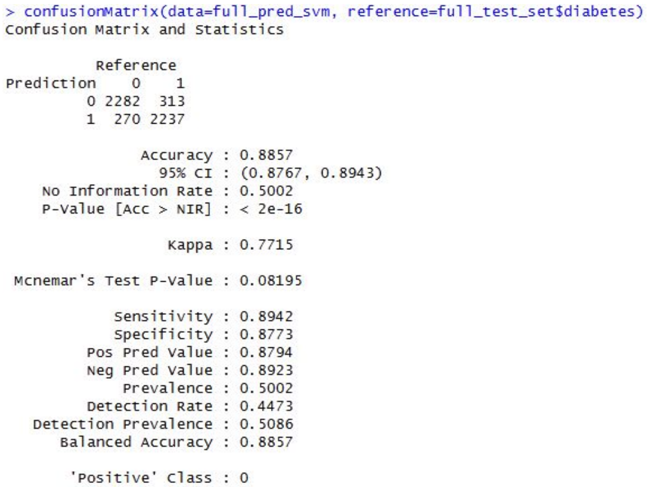


```
> confusionMatrix(cm_knn_fs)
Confusion Matrix and Statistics

   fs_classifier_knn
        0    1
  0  2200  352
  1   402 2148

               Accuracy : 0.8522
                 95% CI : (0.8422, 0.8618)
    No Information Rate : 0.51
    P-Value [Acc > NIR] : < 2e-16

                  Kappa : 0.7044

 Mcnemar's Test P-Value : 0.07435

            Sensitivity : 0.8455
            Specificity : 0.8592
         Pos Pred Value : 0.8621
         Neg Pred Value : 0.8424
             Prevalence : 0.5100
         Detection Rate : 0.4312
   Detection Prevalence : 0.5002
      Balanced Accuracy : 0.8524

       'Positive' Class : 0
```

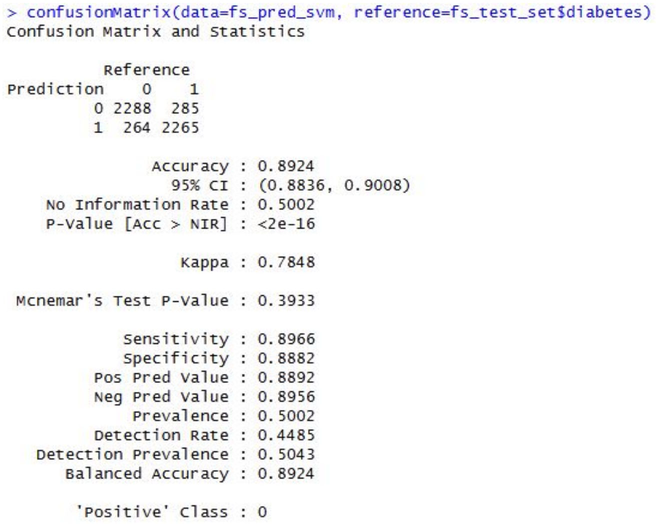
Models Analysis - Support Vector Machine



```
###############################################################################
######################### SUPPORT VECTOR MACHINE ##############################
###############################################################################

# Full Model
full_train_set_svm <- full_train_set
full_train_set_svm$diabetes <- factor(full_train_set_svm$diabetes, levels=c(0,1))
full_test_set_svm <- full_test_set
full_test_set_svm$diabetes  <- factor(full_test_set_svm$diabetes, levels=c(0,1))

full_model_svm <- svm(diabetes ~ ., data = full_train_set, kernel = "linear")

full_pred_svm <- predict(full_model_svm, full_test_set)

confusionMatrix(data=full_pred_svm, reference=full_test_set$diabetes)

# Feature Selection Model
fs_train_set_svm <- fs_train_set
fs_train_set_svm$diabetes <- factor(fs_train_set_svm$diabetes, levels=c(0,1))
fs_test_set_svm <- fs_test_set
fs_test_set_svm$diabetes  <- factor(fs_test_set_svm$diabetes, levels=c(0,1))

fs_model_svm <- svm(diabetes ~ ., data = fs_train_set, kernel = "linear")

fs_pred_svm <- predict(fs_model_svm, fs_test_set)

confusionMatrix(data=fs_pred_svm, reference=fs_test_set$diabetes)
```

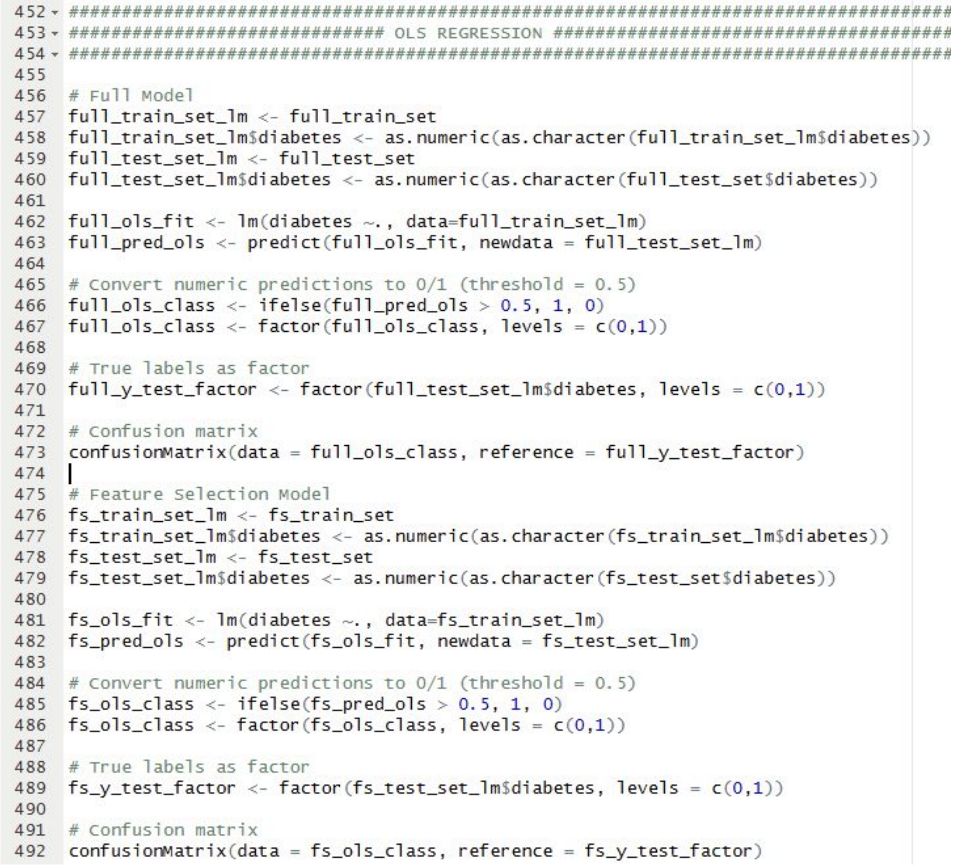


```
> confusionMatrix(data=full_pred_svm, reference=full_test_set$diabetes)
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2282  313
         1  270 2237

               Accuracy : 0.8857
                 95% CI : (0.8767, 0.8943)
    No Information Rate : 0.5002
    P-Value [Acc > NIR] : < 2e-16

                  Kappa : 0.7715

 Mcnemar's Test P-Value : 0.08195

            Sensitivity : 0.8942
            Specificity : 0.8773
         Pos Pred Value : 0.8794
         Neg Pred Value : 0.8923
             Prevalence : 0.5002
         Detection Rate : 0.4473
   Detection Prevalence : 0.5086
      Balanced Accuracy : 0.8857

       'Positive' Class : 0
```

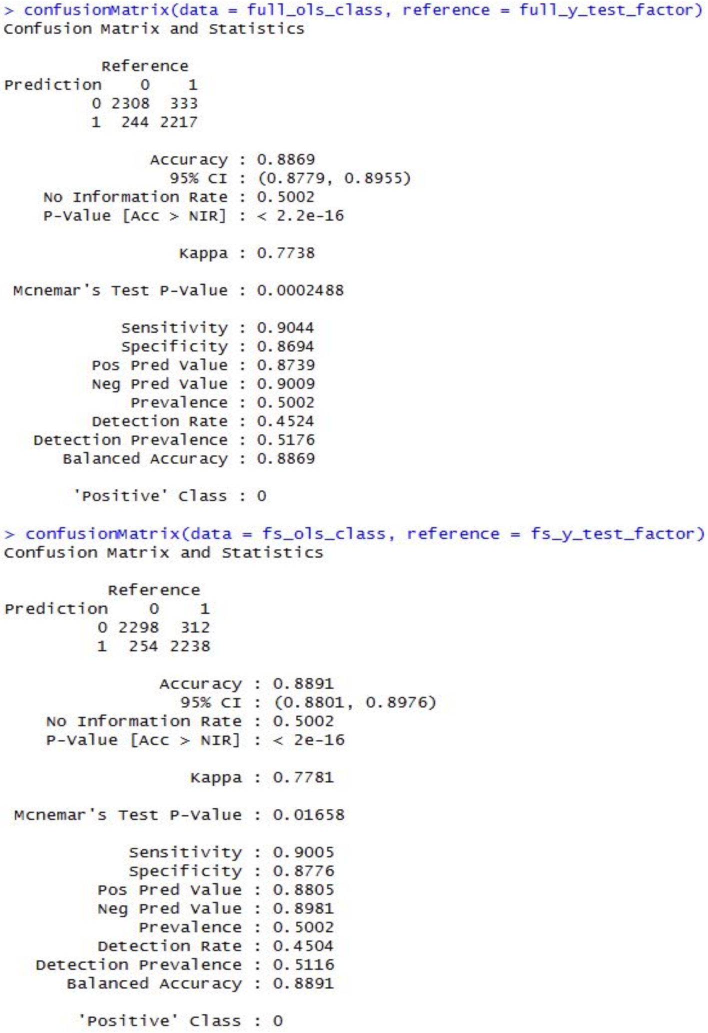


```
> confusionMatrix(data=fs_pred_svm, reference=fs_test_set$diabetes)
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2288  285
         1  264 2265

               Accuracy : 0.8924
                 95% CI : (0.8836, 0.9008)
    No Information Rate : 0.5002
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.7848

 Mcnemar's Test P-Value : 0.3933

            Sensitivity : 0.8966
            Specificity : 0.8882
         Pos Pred Value : 0.8892
         Neg Pred Value : 0.8956
             Prevalence : 0.5002
         Detection Rate : 0.4485
   Detection Prevalence : 0.5043
      Balanced Accuracy : 0.8924

       'Positive' Class : 0
```

Models Analysis - OLS Regression

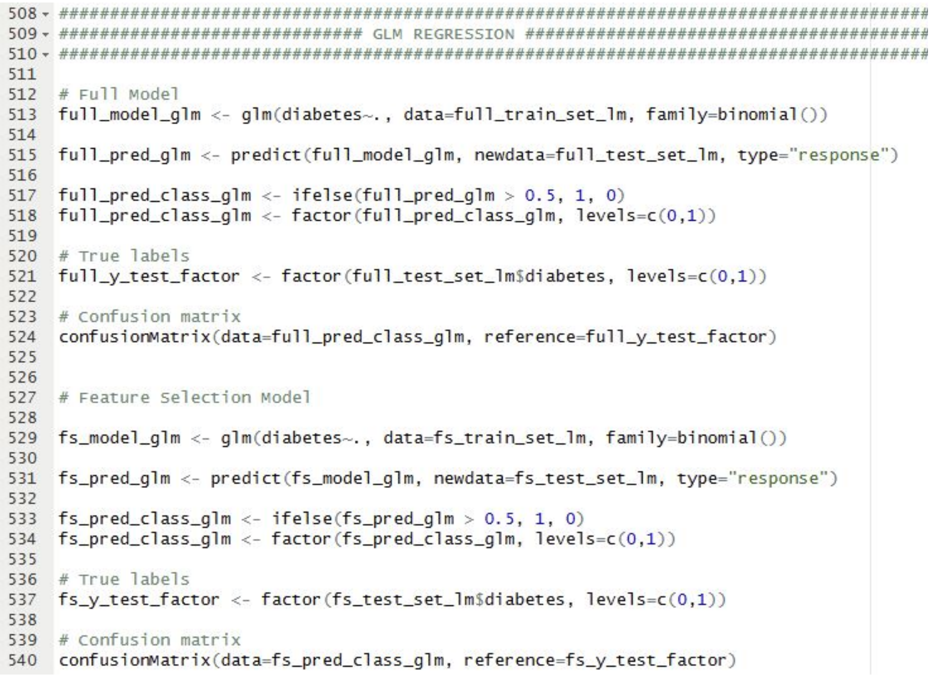


```
################################################################################
############################## OLS REGRESSION ##################################
################################################################################

# Full Model
full_train_set_lm <- full_train_set
full_train_set_lm$diabetes <- as.numeric(as.character(full_train_set_lm$diabetes))
full_test_set_lm <- full_test_set
full_test_set_lm$diabetes <- as.numeric(as.character(full_test_set$diabetes))

full_ols_fit <- lm(diabetes ~., data=full_train_set_lm)
full_pred_ols <- predict(full_ols_fit, newdata = full_test_set_lm)

# Convert numeric predictions to 0/1 (threshold = 0.5)
full_ols_class <- ifelse(full_pred_ols > 0.5, 1, 0)
full_ols_class <- factor(full_ols_class, levels = c(0,1))

# True labels as factor
full_y_test_factor <- factor(full_test_set_lm$diabetes, levels = c(0,1))

# Confusion matrix
confusionMatrix(data = full_ols_class, reference = full_y_test_factor)

# Feature Selection Model
fs_train_set_lm <- fs_train_set
fs_train_set_lm$diabetes <- as.numeric(as.character(fs_train_set_lm$diabetes))
fs_test_set_lm <- fs_test_set
fs_test_set_lm$diabetes <- as.numeric(as.character(fs_test_set$diabetes))

fs_ols_fit <- lm(diabetes ~., data=fs_train_set_lm)
fs_pred_ols <- predict(fs_ols_fit, newdata = fs_test_set_lm)

# Convert numeric predictions to 0/1 (threshold = 0.5)
fs_ols_class <- ifelse(fs_pred_ols > 0.5, 1, 0)
fs_ols_class <- factor(fs_ols_class, levels = c(0,1))

# True labels as factor
fs_y_test_factor <- factor(fs_test_set_lm$diabetes, levels = c(0,1))

# Confusion matrix
confusionMatrix(data = fs_ols_class, reference = fs_y_test_factor)
```

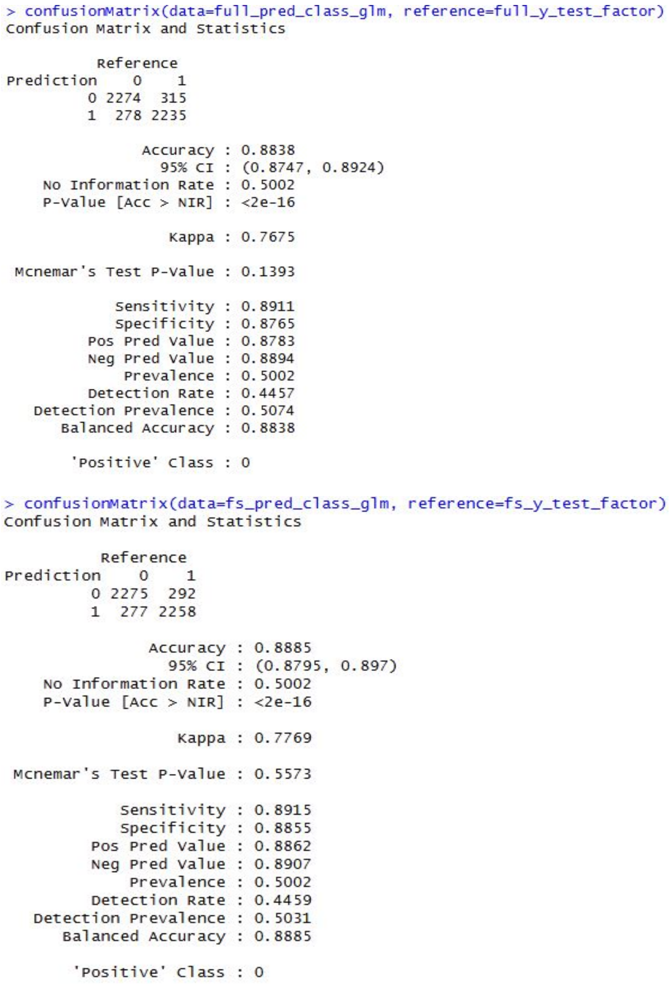


```
> confusionMatrix(data = full_ols_class, reference = full_y_test_factor)
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2308  333
         1  244 2217

               Accuracy : 0.8869
                 95% CI : (0.8779, 0.8955)
    No Information Rate : 0.5002
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.7738

 Mcnemar's Test P-Value : 0.0002488

            Sensitivity : 0.9044
            Specificity : 0.8694
         Pos Pred Value : 0.8739
         Neg Pred Value : 0.9009
             Prevalence : 0.5002
         Detection Rate : 0.4524
   Detection Prevalence : 0.5176
      Balanced Accuracy : 0.8869

       'Positive' Class : 0

> confusionMatrix(data = fs_ols_class, reference = fs_y_test_factor)
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2298  312
         1  254 2238

               Accuracy : 0.8891
                 95% CI : (0.8801, 0.8976)
    No Information Rate : 0.5002
    P-Value [Acc > NIR] : < 2e-16

                  Kappa : 0.7781

 Mcnemar's Test P-Value : 0.01658

            Sensitivity : 0.9005
            Specificity : 0.8776
         Pos Pred Value : 0.8805
         Neg Pred Value : 0.8981
             Prevalence : 0.5002
         Detection Rate : 0.4504
   Detection Prevalence : 0.5116
      Balanced Accuracy : 0.8891

       'Positive' Class : 0
```

Models Analysis - GLM Regression

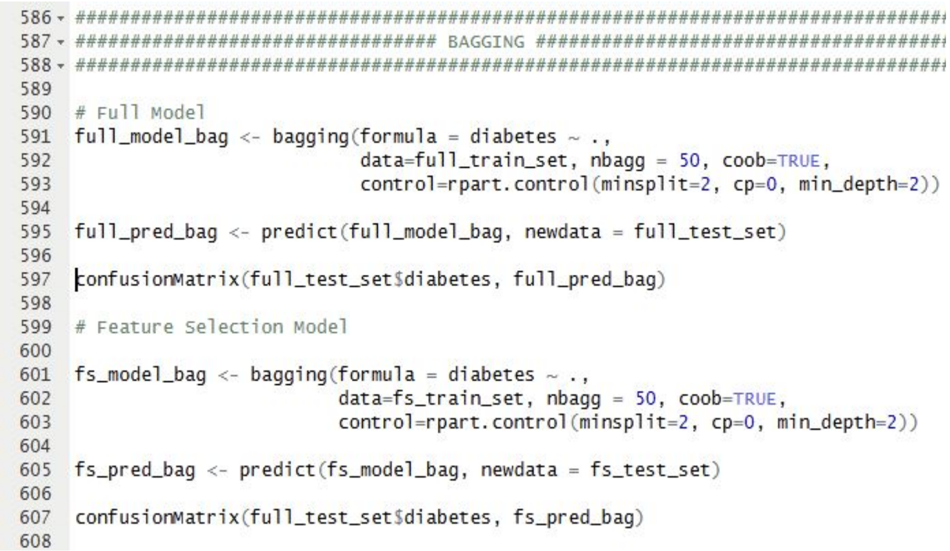


```
######################################################################################
############################## GLM REGRESSION ########################################
######################################################################################

# Full Model
full_model_glm <- glm(diabetes~., data=full_train_set_lm, family=binomial())

full_pred_glm <- predict(full_model_glm, newdata=full_test_set_lm, type="response")

full_pred_class_glm <- ifelse(full_pred_glm > 0.5, 1, 0)
full_pred_class_glm <- factor(full_pred_class_glm, levels=c(0,1))

# True labels
full_y_test_factor <- factor(full_test_set_lm$diabetes, levels=c(0,1))

# Confusion matrix
confusionMatrix(data=full_pred_class_glm, reference=full_y_test_factor)


# Feature Selection Model

fs_model_glm <- glm(diabetes~., data=fs_train_set_lm, family=binomial())

fs_pred_glm <- predict(fs_model_glm, newdata=fs_test_set_lm, type="response")

fs_pred_class_glm <- ifelse(fs_pred_glm > 0.5, 1, 0)
fs_pred_class_glm <- factor(fs_pred_class_glm, levels=c(0,1))

# True labels
fs_y_test_factor <- factor(fs_test_set_lm$diabetes, levels=c(0,1))

# Confusion matrix
confusionMatrix(data=fs_pred_class_glm, reference=fs_y_test_factor)
```

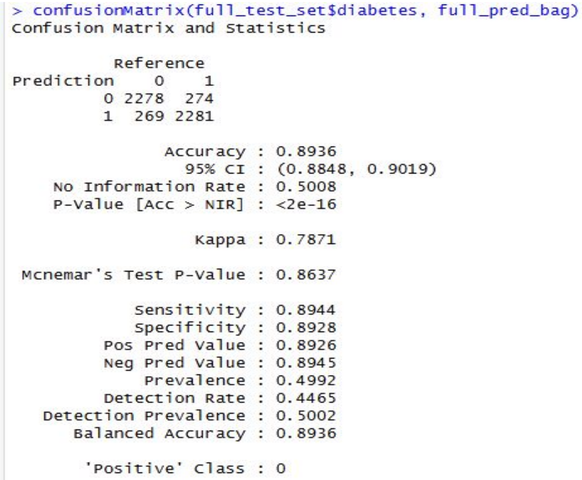


```
> confusionMatrix(data=full_pred_class_glm, reference=full_y_test_factor)
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2274  315
         1  278 2235

               Accuracy : 0.8838
                 95% CI : (0.8747, 0.8924)
    No Information Rate : 0.5002
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.7675

 Mcnemar's Test P-Value : 0.1393

            Sensitivity : 0.8911
            Specificity : 0.8765
         Pos Pred Value : 0.8783
         Neg Pred Value : 0.8894
             Prevalence : 0.5002
         Detection Rate : 0.4457
   Detection Prevalence : 0.5074
      Balanced Accuracy : 0.8838

       'Positive' Class : 0

> confusionMatrix(data=fs_pred_class_glm, reference=fs_y_test_factor)
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2275  292
         1  277 2258

               Accuracy : 0.8885
                 95% CI : (0.8795, 0.897)
    No Information Rate : 0.5002
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.7769

 Mcnemar's Test P-Value : 0.5573

            Sensitivity : 0.8915
            Specificity : 0.8855
         Pos Pred Value : 0.8862
         Neg Pred Value : 0.8907
             Prevalence : 0.5002
         Detection Rate : 0.4459
   Detection Prevalence : 0.5031
      Balanced Accuracy : 0.8885

       'Positive' Class : 0
```

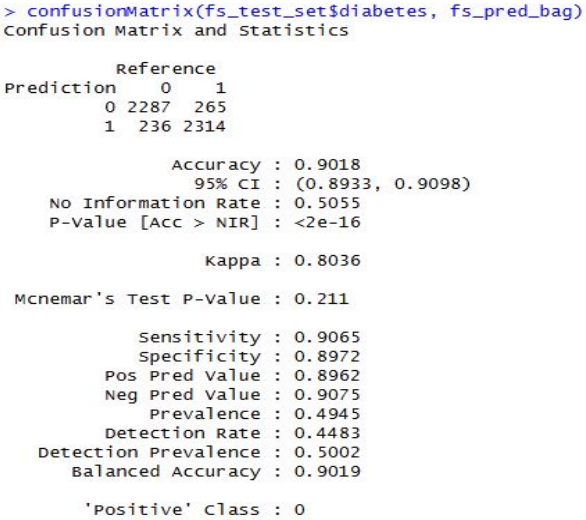
Models Analysis - Bagging



```
################################################################################
################################## BAGGING ####################################
################################################################################

# Full Model
full_model_bag <- bagging(formula = diabetes ~ .,
                          data=full_train_set, nbagg = 50, coob=TRUE,
                          control=rpart.control(minsplit=2, cp=0, min_depth=2))

full_pred_bag <- predict(full_model_bag, newdata = full_test_set)

confusionMatrix(full_test_set$diabetes, full_pred_bag)

# Feature Selection Model

fs_model_bag <- bagging(formula = diabetes ~ .,
                        data=fs_train_set, nbagg = 50, coob=TRUE,
                        control=rpart.control(minsplit=2, cp=0, min_depth=2))

fs_pred_bag <- predict(fs_model_bag, newdata = fs_test_set)

confusionMatrix(full_test_set$diabetes, fs_pred_bag)
```

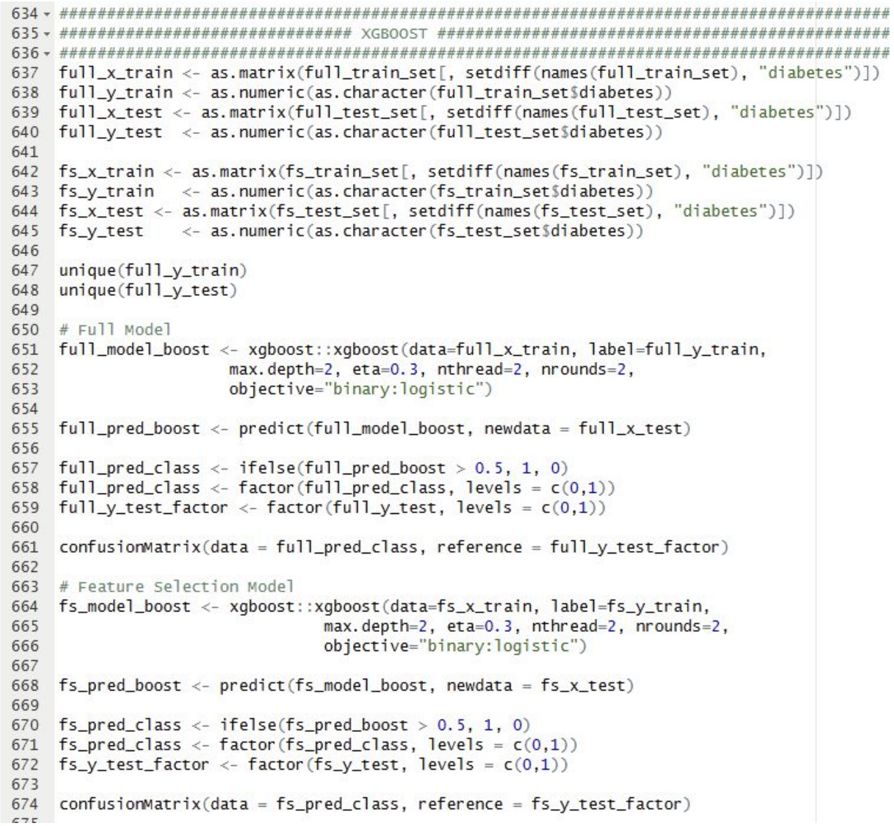


```
> confusionMatrix(full_test_set$diabetes, full_pred_bag)
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2278  274
         1  269 2281

               Accuracy : 0.8936
                 95% CI : (0.8848, 0.9019)
    No Information Rate : 0.5008
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.7871

 Mcnemar's Test P-Value : 0.8637

            Sensitivity : 0.8944
            Specificity : 0.8928
         Pos Pred Value : 0.8926
         Neg Pred Value : 0.8945
             Prevalence : 0.4992
         Detection Rate : 0.4465
   Detection Prevalence : 0.5002
      Balanced Accuracy : 0.8936

       'Positive' Class : 0
```

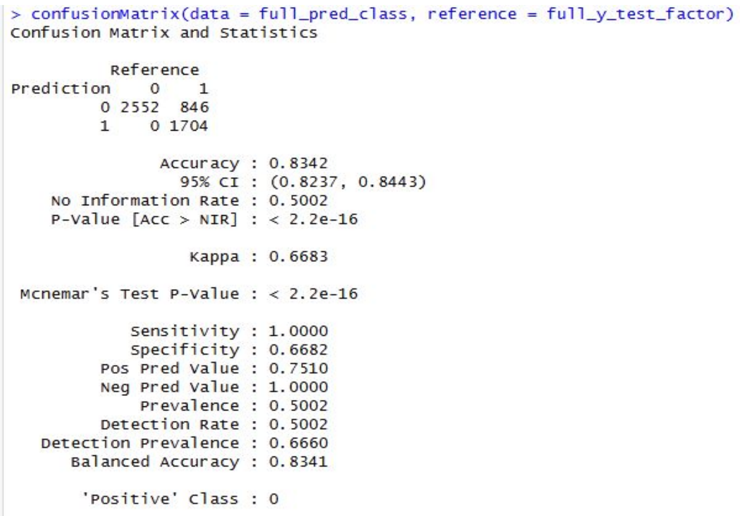


```
> confusionMatrix(fs_test_set$diabetes, fs_pred_bag)
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2287  265
         1  236 2314

               Accuracy : 0.9018
                 95% CI : (0.8933, 0.9098)
    No Information Rate : 0.5055
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.8036

 Mcnemar's Test P-Value : 0.211

            Sensitivity : 0.9065
            Specificity : 0.8972
         Pos Pred Value : 0.8962
         Neg Pred Value : 0.9075
             Prevalence : 0.4945
         Detection Rate : 0.4483
   Detection Prevalence : 0.5002
      Balanced Accuracy : 0.9019

       'Positive' Class : 0
```

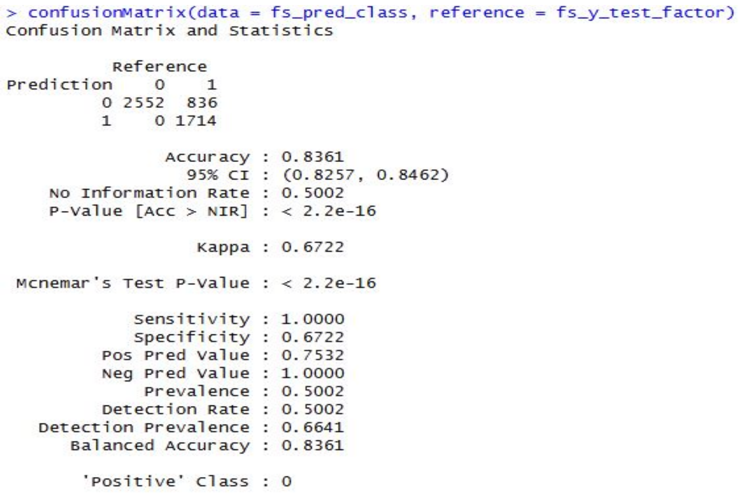
Models Analysis - XGBoost



```
####################################################################################
############################## XGBOOST #############################################
####################################################################################
full_x_train <- as.matrix(full_train_set[, setdiff(names(full_train_set), "diabetes")])
full_y_train <- as.numeric(as.character(full_train_set$diabetes))
full_x_test <- as.matrix(full_test_set[, setdiff(names(full_test_set), "diabetes")])
full_y_test  <- as.numeric(as.character(full_test_set$diabetes))

fs_x_train <- as.matrix(fs_train_set[, setdiff(names(fs_train_set), "diabetes")])
fs_y_train   <- as.numeric(as.character(fs_train_set$diabetes))
fs_x_test <- as.matrix(fs_test_set[, setdiff(names(fs_test_set), "diabetes")])
fs_y_test    <- as.numeric(as.character(fs_test_set$diabetes))

unique(full_y_train)
unique(full_y_test)

# Full Model
full_model_boost <- xgboost::xgboost(data=full_x_train, label=full_y_train,
                  max.depth=2, eta=0.3, nthread=2, nrounds=2,
                  objective="binary:logistic")

full_pred_boost <- predict(full_model_boost, newdata = full_x_test)

full_pred_class <- ifelse(full_pred_boost > 0.5, 1, 0)
full_pred_class <- factor(full_pred_class, levels = c(0,1))
full_y_test_factor <- factor(full_y_test, levels = c(0,1))

confusionMatrix(data = full_pred_class, reference = full_y_test_factor)

# Feature Selection Model
fs_model_boost <- xgboost::xgboost(data=fs_x_train, label=fs_y_train,
                            max.depth=2, eta=0.3, nthread=2, nrounds=2,
                            objective="binary:logistic")

fs_pred_boost <- predict(fs_model_boost, newdata = fs_x_test)

fs_pred_class <- ifelse(fs_pred_boost > 0.5, 1, 0)
fs_pred_class <- factor(fs_pred_class, levels = c(0,1))
fs_y_test_factor <- factor(fs_y_test, levels = c(0,1))

confusionMatrix(data = fs_pred_class, reference = fs_y_test_factor)
```

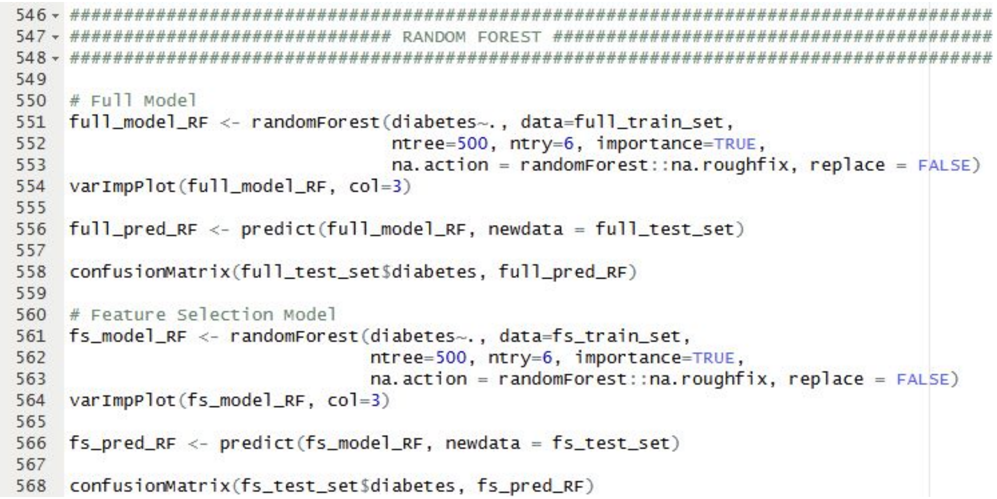


```
> confusionMatrix(data = full_pred_class, reference = full_y_test_factor)
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2552  846
         1    0 1704

               Accuracy : 0.8342
                 95% CI : (0.8237, 0.8443)
    No Information Rate : 0.5002
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6683

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 1.0000
            Specificity : 0.6682
         Pos Pred Value : 0.7510
         Neg Pred Value : 1.0000
             Prevalence : 0.5002
         Detection Rate : 0.5002
   Detection Prevalence : 0.6660
      Balanced Accuracy : 0.8341

       'Positive' Class : 0
```

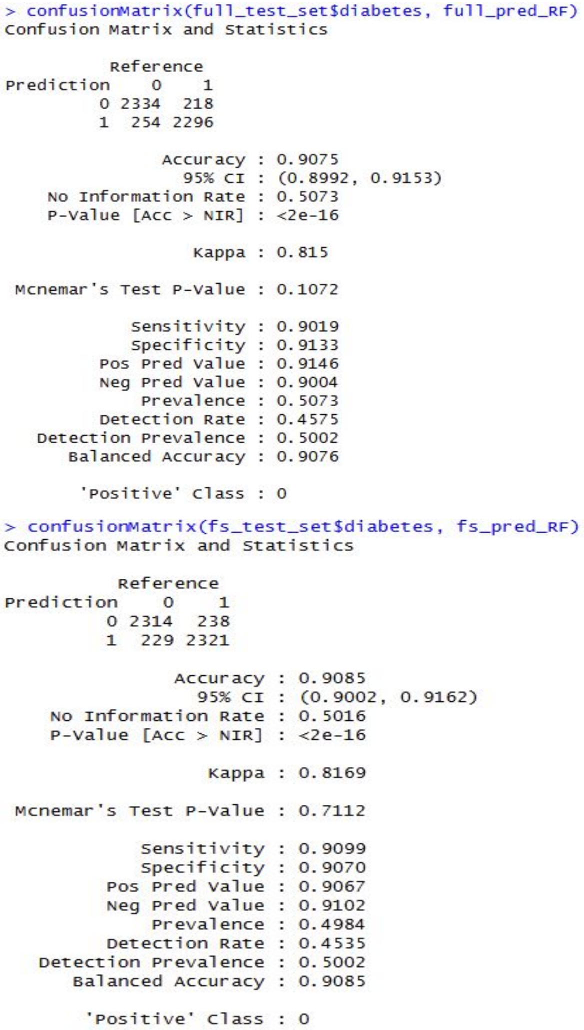


```
> confusionMatrix(data = fs_pred_class, reference = fs_y_test_factor)
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2552  836
         1    0 1714

               Accuracy : 0.8361
                 95% CI : (0.8257, 0.8462)
    No Information Rate : 0.5002
    P-Value [Acc > NIR] : < 2.2e-16

                  Kappa : 0.6722

 Mcnemar's Test P-Value : < 2.2e-16

            Sensitivity : 1.0000
            Specificity : 0.6722
         Pos Pred Value : 0.7532
         Neg Pred Value : 1.0000
             Prevalence : 0.5002
         Detection Rate : 0.5002
   Detection Prevalence : 0.6641
      Balanced Accuracy : 0.8361

       'Positive' Class : 0
```

Models Analysis - Random Forest

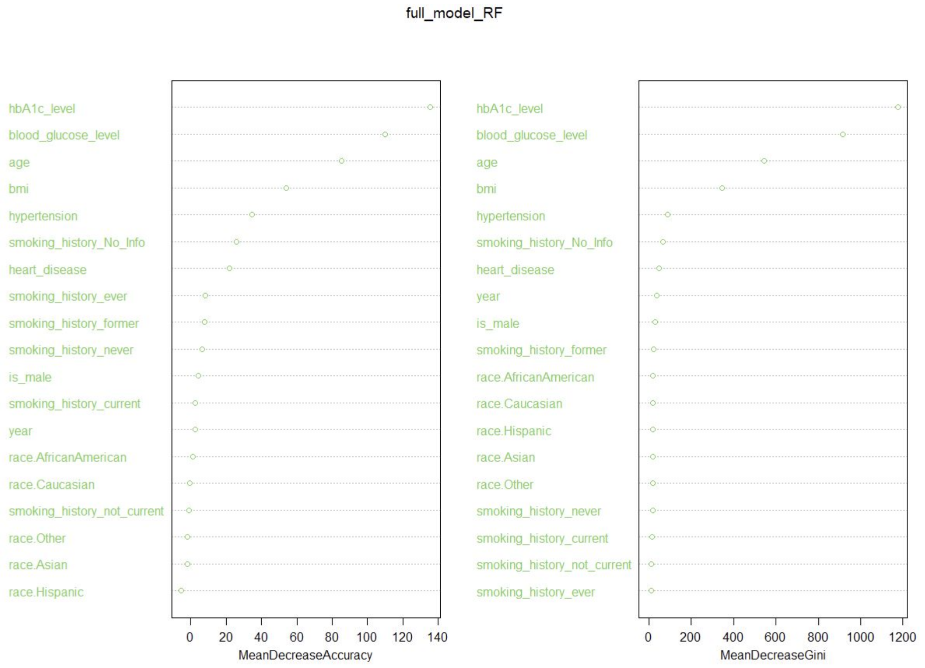


```
###################################################################################
############################## RANDOM FOREST ######################################
###################################################################################

# Full Model
full_model_RF <- randomForest(diabetes~., data=full_train_set,
                              ntree=500, ntry=6, importance=TRUE,
                              na.action = randomForest::na.roughfix, replace = FALSE)
varImpPlot(full_model_RF, col=3)

full_pred_RF <- predict(full_model_RF, newdata = full_test_set)

confusionMatrix(full_test_set$diabetes, full_pred_RF)

# Feature Selection Model
fs_model_RF <- randomForest(diabetes~., data=fs_train_set,
                            ntree=500, ntry=6, importance=TRUE,
                            na.action = randomForest::na.roughfix, replace = FALSE)
varImpPlot(fs_model_RF, col=3)

fs_pred_RF <- predict(fs_model_RF, newdata = fs_test_set)

confusionMatrix(fs_test_set$diabetes, fs_pred_RF)
```

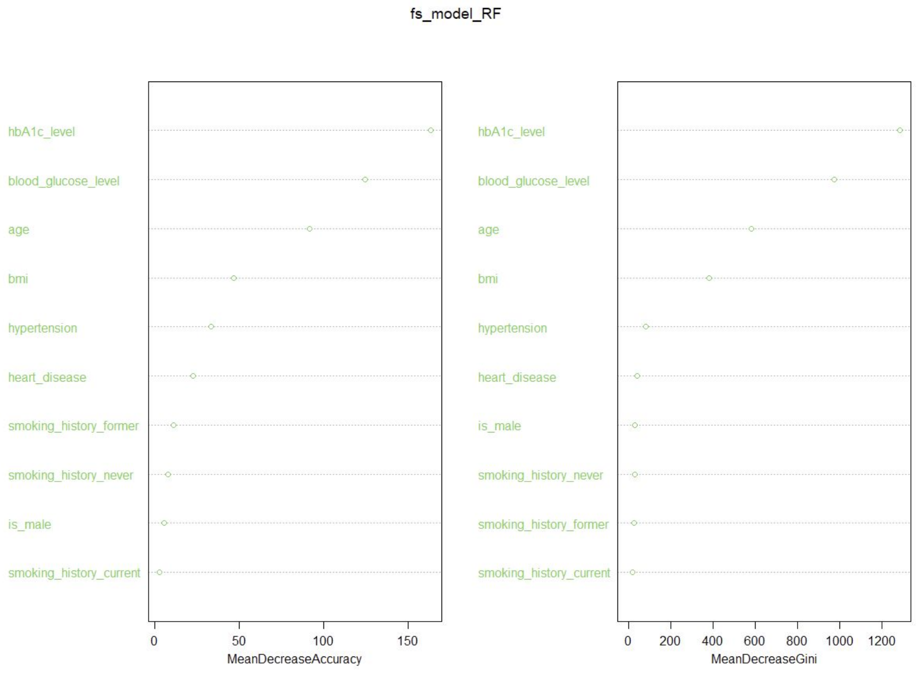


```
> confusionMatrix(full_test_set$diabetes, full_pred_RF)
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2334  218
         1  254 2296

               Accuracy : 0.9075
                 95% CI : (0.8992, 0.9153)
    No Information Rate : 0.5073
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.815

 Mcnemar's Test P-Value : 0.1072

            Sensitivity : 0.9019
            Specificity : 0.9133
         Pos Pred Value : 0.9146
         Neg Pred Value : 0.9004
             Prevalence : 0.5073
         Detection Rate : 0.4575
   Detection Prevalence : 0.5002
      Balanced Accuracy : 0.9076

       'Positive' Class : 0

> confusionMatrix(fs_test_set$diabetes, fs_pred_RF)
Confusion Matrix and Statistics

          Reference
Prediction    0    1
         0 2314  238
         1  229 2321

               Accuracy : 0.9085
                 95% CI : (0.9002, 0.9162)
    No Information Rate : 0.5016
    P-Value [Acc > NIR] : <2e-16

                  Kappa : 0.8169

 Mcnemar's Test P-Value : 0.7112

            Sensitivity : 0.9099
            Specificity : 0.9070
         Pos Pred Value : 0.9067
         Neg Pred Value : 0.9102
             Prevalence : 0.4984
         Detection Rate : 0.4535
   Detection Prevalence : 0.5002
      Balanced Accuracy : 0.9085

       'Positive' Class : 0
```

Models Analysis - Random Forest



full_model_RF



fs_model_RF

Model Comparison

Model	Accuracy	Kappa
Decision Tree - Full	89.98%	0.7997
Decision Tree - FS	89.77%	0.7954
Naive Bayes - Full	81.81%	0.6362
Naive Bayes - FS	83.44%	0.6687
KNN - Full	84.97%	0.6993
KNN - FS	85.22%	0.7044
SVM - Full	88.57%	0.7715
SVM - FS	89.24%	0.7848
Linear Regression - Full	88.69%	0.7738
Linear Regression - FS	88.91%	0.7781

Model	Accuracy	Kappa
Logistic Regression - Full	88.38%	0.7675
GLM - FS	88.85%	0.7769
RF - Full	90.75%	0.815
RF - FS	90.85%	0.8169
Bagging - Full	89.36%	0.7871
Bagging - FS	90.18%	0.8036
XGBoost - Full	83.42%	0.6683
XGBoost - FS	83.61%	0.6722

Research Findings

- Random Forest was the model with best accuracy (~91%)
- The model describes correctly 82% of the data according to the kappa metric
- Accordingly to the feature importance plot, hemoglobin A1c and Blood glucose level heavy indicates if the patience has diabetes or not, followed by age and bmi
- It provides a good balance for between catching disease cases while minimizing unnecessary follow-up tests

Research Findings

- The model performance is statistically significant as it has extremely low p-value

- The model has a good recall value which is important for this study case where we want to avoid false negatives as much as possible

- Random Forest was the best one due to its capacity to handling complex non-linear relationships, less hyperparameter sensitivity, and robust to overfitting

Bibliography

- Classes Notes
- https://www.geeksforgeeks.org/machine-learning/what-are-the-advantages-and-disadvantages-of-random-forest/
- https://www.geeksforgeeks.org/machine-learning/confusion-matrix-machine-learning/

Group Members:

‘Aafrin Zahid Memon
Akintunde Akinro
Bruno do Nascimento Beserra

THANK YOU.

DOUGLASCOLLEGE