**R Script**

**#Data Preparation and Cleaning**

**#Census dataset**

census <- census[,-c(1,5:23)] popullation <- census[,-1]

colnames(popullation)[2] <- "Popullation"

**#Referendum Results dataset**

referendum <- EU.referendum.result.data[,-c(1:4,12,13,15:19,7)]

**#General elections results 2015**

results1 <- RESULTS.FOR.ANALYSIS[,-c(1,3,4,5,6,7)] results2 <- results1[,-c(4:20,22:40,42:46)] results3 <- results2[,-c(7:21,23,25:50)] results4 <- results3[,-c(9:41,43:50)] results <- results4[-c(10:17,2,3)] results <- results[,-3]

**#Treating the NAs**

colnames(results)[1] <- "Area"

Results\_NA\_count <- apply(is.na(results), 2, sum)

Results\_NA\_count

Results\_NA\_perc <- Results\_NA\_count / dim(results)[1] \* 100

Results\_NA\_perc

results[is.na(results$C), 'C'] = '1' results[is.na(results$Green), 'Green'] = '1' results[is.na(results$Lab), 'Lab'] = '1' results[is.na(results$LD), 'LD'] = '1' results[is.na(results$UKIP), 'UKIP'] = '1' colnames(results)[1] <- "Area"

**#Income and Tax dataset**

Income\_and\_Tax <- NS\_Table\_3\_15\_1516[,-c(2:15,19,20,23,16)] colnames(Income\_and\_Tax)[1] <- "Area" colnames(Income\_and\_Tax)[2] <- "Mean\_Income" colnames(Income\_and\_Tax)[3] <- "Median\_Income" colnames(Income\_and\_Tax)[4] <- "Mean\_Tax" colnames(Income\_and\_Tax)[5] <- "Median\_Tax"

Income\_and\_Tax <- na.omit(Income\_and\_Tax)

Income\_and\_Tax$Area <- factor(Income\_and\_Tax$Area)

Income\_and\_Tax$Median\_Tax <- as.numeric(as.character(Income\_and\_Tax$Median\_Tax))

Income\_and\_Tax$Median\_Income <- as.numeric(as.character(Income\_and\_Tax$Median\_Income))

Income\_and\_Tax$Mean\_Tax <- as.numeric(as.character(Income\_and\_Tax$Mean\_Tax))

Income\_and\_Tax$Mean\_Income <- as.numeric(as.character(Income\_and\_Tax$Mean\_Income))

**#Library dplyr - Join the datasets together**

leave <- inner\_join(referendum,results) leave <- inner\_join(Income\_and\_Tax,leave)

leave <- inner\_join(popullation,leave)

leave$Area <- factor(leave$Area) leave$C <- as.numeric(as.character(leave$C)) leave$Green <- as.numeric(as.character(leave$Green)) leave$UKIP <- as.numeric(as.character(leave$UKIP)) leave$Lab <- as.numeric(as.character(leave$Lab)) leave$LD <- as.numeric(as.character(leave$LD))

**#Exploratory Data Analysis**

**#Histograms**

summary(leave)

par(mfrow = c(2,2))

hist(

leave$Median\_Income, xlab = 'Median Income in $1000',

ylab = 'Frequency',

main = 'Median Income Per Consituency', col = rgb(0, 1, 0), border = 'white', xaxt = 'n'

)

axis(1, at = seq(0, 65000, 5000), seq(0, 65, 5)) abline(v = median(leave$Median\_Income,), lty = 2) legend('topright', 'median Income', lty = 2, bty = 'n')

hist(

leave$Median\_Tax, xlab = 'Median Tax in $1000', ylab = 'Frequency',

main = 'Media Tax Per Consituency',

col = rgb(0, 1, 0), border = 'white', xaxt = 'n'

)

axis(1, at = seq(0, 65000, 5000), seq(0, 65, 5))

abline(v = median(leave$Median\_Tax,), lty = 2)

legend('topright', 'median Tax', lty = 2, bty = 'n')

hist(

leave$Mean\_Income, xlab = 'Mean Income in $1000', ylab = 'Frequency',

main = 'Mean Income Per Consituency', col = rgb(0, 1, 0), border = 'white',

xaxt = 'n'

)

axis(1, at = seq(0, 65000, 5000), seq(0, 65, 5)) abline(v = median(leave$Mean\_Income,), lty = 2) legend('topright', 'Mean Income', lty = 2, bty = 'n')

hist(

leave$Mean\_Tax, xlab = 'Mean Tax in $1000', ylab = 'Frequency', main = 'Mean Tax Per Consituency',

col = rgb(0, 1, 0), border = 'white',

xaxt = 'n'

)

axis(1, at = seq(0, 65000, 5000), seq(0, 65, 5)) abline(v = median(leave$Mean\_Tax,), lty = 2)

legend('topright', 'Mean Tax', lty = 2, bty = 'n')

plot(

leave$Pct\_Leave ~ leave$Median\_Income,

ylab = 'Percentage of Leave', xlab = 'Median Income', main = 'Leave vs Income', pch = 0.15

)

abline(v = median(leave$Median\_Income,), lty = 2) legend('topright', 'Median Income', lty = 2, bty = 'n') dev.off()

plot(

leave$Pct\_Leave ~ leave$Median\_Tax, ylab = 'Percentage of Leave', xlab = 'Median Tax', main = 'Leave vs Tax', pch = 0.15

)

abline(v = median(leave$Median\_Tax,), lty = 2)

legend('topright', 'Median Tax', lty = 2, bty = 'n')

**#Principal Component Analysis**

par(mfrow = c(2,2))

plot(density(leave[, 3]), main = names(leave)[3], xlab = names(leave)[3]) plot(density(leave[, 4]), main = names(leave)[4], xlab = names(leave)[4]) plot(density(leave[, 5]), main = names(leave)[5], xlab = names(leave)[5]) plot(density(leave[, 6]), main = names(leave)[6], xlab = names(leave)[6]) pc\_leave <- prcomp(leave[,-1], center = T, scale. = T) pc\_leave\_var <- pc\_leave$sdev^2 pc\_leave\_var

pc\_leave\_PEV <- pc\_leave\_var / sum(pc\_leave\_var) pc\_leave\_PEV png(file = "percent") plot(pc\_leave) dev.off()

plot( cumsum(pc\_leave\_PEV),

ylim = c(0,1), xlab = 'PC', ylab = 'Cumulative PEV', pch = 20, col = 'orange'

)

abline(h = 0.8, col = 'red', lty = 'dashed') pc\_leave\_loadings <- pc\_leave$rotation pc\_leave\_loadings

colvector = c('red', 'orange', 'yellow', 'green', 'cyan', 'blue') labvector = c('PC1', 'PC2', 'PC3') barplot( pc\_leave\_loadings[,c(1:3)],

beside = T,

yaxt = 'n', names.arg = labvector, col = colvector, ylim = c(-1,1), border = 'white',

ylab = 'loadings'

)

axis(2, seq(-1,1,0.1)) legend( 'bottomright', bty = 'n', col = colvector,

pch = 15,

row.names(pc\_leave\_loadings)

)

par(mfrow = c(3,1)) biplot( pc\_leave, scale = 0,

col = c('grey40','orange')

)

**#Biplot**

biplot( pc\_leave, choices = c(1,3), scale = 0, col = c('grey40','orange')

)

biplot( pc\_leave, choices = c(2,3), scale = 0,

col = c('grey40','orange')

)

leave\_clear <- leave[,-c(2,8,10,11,12,14,7)] summary(pc\_leave\_clear)

pc\_leave\_clear <- prcomp(leave\_clear[,-1], center = T, scale. = T)

pc\_leave\_var <- pc\_leave\_clear$sdev^2 pc\_leave\_var

pc\_leave\_PEV <- pc\_leave\_var / sum(pc\_leave\_var) pc\_leave\_PEV plot(pc\_leave\_clear) png(file = "percent") plot( cumsum(pc\_leave\_PEV),

ylim = c(0,1), xlab = 'PC', ylab = 'Cumulative PEV', pch = 20,

col = 'orange'

)

abline(h = 0.8, col = 'red', lty = 'dashed') pc\_leave\_loadings <- pc\_leave\_clear$rotation pc\_leave\_loadings opar <- par()

colvector = c('red', 'orange', 'yellow', 'green', 'cyan', 'blue') labvector = c('PC1', 'PC2', 'PC3') barplot( pc\_leave\_loadings[,c(1:3)],

beside = T,

yaxt = 'n', names.arg = labvector, col = colvector, ylim = c(-1,1), border = 'white', ylab = 'loadings'

)

axis(2, seq(-1,1,0.1)) legend( 'bottomright', bty = 'n', col = colvector,

pch = 15,

row.names(pc\_leave\_loadings)

)

par(mfrow = c(2,2)) png(file = "percent") biplot( pc\_leave\_clear, scale = 0,

col = c('grey40','orange')

)

biplot( pc\_leave\_clear, choices = c(1,3), scale = 0,

col = c('grey40','orange')

)

biplot( pc\_leave\_clear, choices = c(2,3), scale = 0,

col = c('grey40','orange')

)

pca3d::pca3d(pc\_leave\_clear, show.labels = T)

summary(pc\_leave)

summary(pc\_leave\_clear)

plot(pc\_leave, type = "l")

**#Clustering**

**#Hierarchical Clustering**

dist\_leave <- dist(leave\_clear[,-1], method = 'euclidian')

# then apply complete linkage png(file = "percent")

hc\_leave <- hclust(dist\_leave, method = 'ward.D') hc\_leave

dist\_leave <- dist(leave\_clear[,-1], method = 'euclidian') hc\_leave\_complete <- hclust(dist\_leave, method = 'complete')

**#Plot the associated dendrogram**

plot(hc\_leave, hang = -0.1, labels = leave\_clear$Area) png(file = "percent")

plot(hc\_leave\_complete, hang = -0.1, labels = leave\_clear$Area)

**Evaluation of cluster results**

**#Silhouette Plot**

sil\_hc\_leave <- cluster::silhouette(hc\_cluster\_id\_leave, dist\_leave)

sil\_hc\_leave\_complete <- cluster::silhouette(hc\_cluster\_id\_leave\_complete, dist\_leave)

opar <- par() par(mfrow = c(2,1)) png(file = "percent") plot(sil\_hc\_leave) png(file = "percent") plot(sil\_hc\_leave\_complete) par(opar)

leave\_clear["Outcome"] <- 0

leave\_clear$Outcome[ leave\_clear$Pct\_Leave > 50] <- "1" leave\_clear$Outcome[ leave\_clear$Pct\_Leave < 50] <- "0"

leave\_clear$Outcome <- as.numeric(leave\_clear$Outcome)

**#Neural network**

set.seed(2018) MinMax <- function(x){ tx <- (x - min(x)) / (max(x) - min(x)) return(tx)

}

leave\_minmax <- apply(leave\_clear[,-1], 2, MinMax)

leave\_minmax <- as.data.frame(leave\_minmax) n\_rows <- nrow(leave\_minmax)

training\_idx <- sample(n\_rows, n\_rows \* 0.7) training\_leave\_minmax <- leave\_minmax[training\_idx,] test\_leave <- leave\_minmax[-training\_idx,]

leave\_formula = Outcome ~ Mean\_Income + Median\_Income + Mean\_Tax + Median\_Tax + Pct\_Turnout + C + Lab + G reen + UKIP + LD

png(file = "p")

**#Library neural net**

**#Trying several nn architecture**

leave\_nn\_64 <- neuralnet(leave\_formula, hidden = c(1,2), data = training\_leave\_minmax,linear.output = FALSE) leave\_nn\_65 <- neuralnet(leave\_formula, hidden = c(2,1), data = training\_leave\_minmax,linear.output = FALSE) leave\_nn\_66 <- neuralnet(leave\_formula, hidden = c(1,1), data = training\_leave\_minmax, linear.output = FALSE) leave\_nn\_67 <- neuralnet(leave\_formula, hidden = c(2,2), data = training\_leave\_minmax,linear.output = FALSE) leave\_nn\_68 <- neuralnet(leave\_formula, hidden = c(2,3), data = training\_leave\_minmax,linear.output = FALSE) leave\_nn\_69 <- neuralnet(leave\_formula, hidden = c(4,4), data = training\_leave\_minmax, linear.output = FALSE) pred\_leave\_nn\_64 <- compute(leave\_nn\_64, test\_leave[,-13]) pred\_leave\_nn\_65 <- compute(leave\_nn\_65, test\_leave[,-13]) pred\_leave\_nn\_66 <- compute(leave\_nn\_66, test\_leave[,-13]) pred\_leave\_nn\_67 <- compute(leave\_nn\_67, test\_leave[,-13]) pred\_leave\_nn\_68 <- compute(leave\_nn\_68, test\_leave[,-13]) pred\_leave\_nn\_69 <- compute(leave\_nn\_69, test\_leave[,-13])

**#Evaluating nn results**

leave\_results <- data.frame( actual = test\_leave$Outcome, nn\_64 = pred\_leave\_nn\_64$net.result, nn\_65 = pred\_leave\_nn\_65$net.result,

nn\_66 = pred\_leave\_nn\_66$net.result

)

leave\_results <- data.frame( actual = test\_leave$Outcome, nn\_67 = pred\_leave\_nn\_67$net.result, nn\_68 = pred\_leave\_nn\_68$net.result,

nn\_69 = pred\_leave\_nn\_69$net.result

)

cor(leave\_results[,'actual'], leave\_results[,c("nn\_64","nn\_65", "nn\_66")]) cor(leave\_results[,'actual'], leave\_results[,c("nn\_67","nn\_68", "nn\_69")]) leave\_results <- data.frame( actual = test\_leave$Outcome, nn\_65 = pred\_leave\_nn\_65$net.result

)

leave\_results <- data.frame( actual = test\_leave$Outcome, nn\_65 = pred\_leave\_nn\_65$net.result

)

temptest <- subset(test\_leave, select = c("Mean\_Income", "Median\_Income", "Mean\_Tax", "Median\_Tax", "Pct\_Turn out", "C", "Lab", "Green", "UKIP", "LD"))

head(temptest)

nn\_results <- compute(leave\_nn\_65,temptest) nn\_results <- compute(leave\_nn\_69,temptest) nn\_results

results <- data.frame(actual = test\_leave$Outcome, prediction = nn\_results$net.result) results

roundedresults <- sapply(results, round,digits=0) roundedresultsdf = data.frame(roundedresults) table(leave\_results)

predict\_testNN = (pred\_leave\_nn\_65$net.result \* (max(test\_leave$Outcome) - min(test\_leave$Outcome))) + min(tes t\_leave$Outcome)

predict\_testNN = (pred\_leave\_nn\_64$net.result \* (max(test\_leave$Outcome) - min(test\_leave$Outcome))) + min(tes t\_leave$Outcome)

predict\_testNN = (pred\_leave\_nn\_66$net.result \* (max(test\_leave$Outcome) - min(test\_leave$Outcome))) + min(tes t\_leave$Outcome)

predict\_testNN = (pred\_leave\_nn\_67$net.result \* (max(test\_leave$Outcome) - min(test\_leave$Outcome))) + min(tes t\_leave$Outcome)

predict\_testNN = (pred\_leave\_nn\_68$net.result \* (max(test\_leave$Outcome) - min(test\_leave$Outcome))) + min(tes t\_leave$Outcome)

predict\_testNN = (pred\_leave\_nn\_69$net.result \* (max(test\_leave$Outcome) - min(test\_leave$Outcome))) + min(tes t\_leave$Outcome)

plot(test\_leave$Outcome, predict\_testNN, col='blue', pch=16, ylab = "Predicted Outcome NN", xlab = "Real Outcome

")

abline(0,1)

**#Calculating RMSE and MAE for nn performance evaluation**

RMSE.NN = (sum((test\_leave$Outcome - predict\_testNN)^2) / nrow(test\_leave)) ^ 0.5 RMSE.NN

1-mae(pred\_leave\_nn\_64$net.result \* (max(test\_leave$Outcome) - min(test\_leave$Outcome))) + min(test\_leave$Ou tcome)

1-mae(pred\_concrete\_nn\_65$net.result \* (max(test\_concrete\_minmax$Outcome) - min(test\_concrete\_minmax$Out come))) + min(test\_concrete\_minmax$Outcome)

1-mae(pred\_concrete\_nn\_66$net.result \* (max(test\_concrete\_minmax$Outcome) - min(test\_concrete\_minmax$Out come))) + min(test\_concrete\_minmax$Outcome)

1-mae(pred\_concrete\_nn\_67$net.result \* (max(test\_concrete\_minmax$Outcome) - min(test\_concrete\_minmax$Out come))) + min(test\_concrete\_minmax$Outcome)

1-mae(pred\_concrete\_nn\_68$net.result \* (max(test\_concrete\_minmax$Outcome) - min(test\_concrete\_minmax$Out come))) + min(test\_concrete\_minmax$Outcome)

1-mae(pred\_concrete\_nn\_69$net.result \* (max(test\_concrete\_minmax$Outcome) - min(test\_concrete\_minmax$Out come))) + min(test\_concrete\_minmax$Outcome)

**#Create an extra column for classification**

leave\_clear["Outcome"] <- 0

leave\_clear$Outcome[ leave\_clear$Pct\_Leave > 50] <- "Leave" leave\_clear$Outcome[ leave\_clear$Pct\_Leave < 50] <- "Remain" leave\_clear$Outcome <- as.factor(leave\_clear$Outcome)

**#Decision Tree # Library tree**

set.seed(2018)

n\_rows <- nrow(leave\_clear) training\_idx <- sample(n\_rows, n\_rows \* 0.7) training\_leave <- leave\_clear[training\_idx,] test\_leave <- leave\_clear[-training\_idx,]

leave\_formula = Outcome ~ Mean\_Income + Median\_Income + Mean\_Tax + Median\_Tax + Pct\_Turnout + C + Lab + G reen + UKIP + LD

tree\_leave <- tree(leave\_formula, data = training\_leave) summary(tree\_leave)

plot(tree\_leave) text(tree\_leave, pretty = 0)

cv\_leave <- cv.tree(tree\_leave, FUN=prune.misclass) cv\_leave\_table <- data.frame( size = cv\_leave$size,

error = cv\_leave$dev

)

plot( cv\_leave, xaxt = 'n',

yaxt = 'n'

)

axis(1, seq(1,max(cv\_leave\_table$size))) axis(2, seq(50,150,5))

**#Prune the decision tree**

pruned\_tree\_size <- cv\_leave\_table[which.min(cv\_leave\_table$error), 'size'] pruned\_tree\_leave <- prune.misclass(tree\_leave, best = pruned\_tree\_size) summary(pruned\_tree\_leave)

tree\_leave\_pred <- predict(tree\_leave, test\_leave[,-13], type= "class")

pruned\_tree\_leave\_pred <- predict(pruned\_tree\_leave, test\_leave[,-13], type= "class") leave\_results <- data.frame( actual = test\_leave$Outcome, unpruned = tree\_leave\_pred, pruned = pruned\_tree\_leave\_pred)

unpruned\_results\_table <- table(leave\_results[,c('actual', 'unpruned')]) unpruned\_results\_table

pruned\_results\_table <- table(leave\_results[,c('actual', 'pruned')]) pruned\_results\_table

acc\_unpruned <- sum(diag(unpruned\_results\_table)) / sum(unpruned\_results\_table) acc\_unpruned

acc\_pruned <- sum(diag(pruned\_results\_table)) / sum(pruned\_results\_table) acc\_pruned

**#Performance evaluation Decision Tree and Random Forest**

**#Library caret**

set.seed(2018)

leave\_formula <- reformulate(names(training\_leave[, -13]), response = 'Outcome')

ctrl\_parameters <- trainControl(method = 'CV', number = 10) ctrl\_parameters <- trainControl(method = 'CV', number = 30) modelLookup('rpart')

leave\_tree\_perf <- train(leave\_formula, data = training\_leave, method = "rpart", trControl = ctrl\_parameters) leave\_tree\_perf png(file = "p")

plot(leave\_tree\_perf)

modelLookup('rf')

leave\_rf <- train(leave\_formula, data = training\_leave, method = "rf", trControl = ctrl\_parameters) leave\_rf plot(leave\_rf) leave\_tree\_predict <- cbind( actual = test\_leave$Outcome,

predicted = predict(leave\_tree\_perf, test\_leave[, -13], type = 'raw'), predict(leave\_tree\_perf, test\_leave[, -13], type = 'prob')

)

leave\_rf\_predict <- cbind( actual = test\_leave$Outcome,

predicted = predict(leave\_rf, test\_leave[, -13], type = 'raw'), predict(leave\_rf, test\_leave[, -13], type = 'prob')

)

leave\_rf\_predict

plot(leave\_rf\_predict)

tree\_confmat <- confusionMatrix(data = leave\_tree\_predict$predicted, reference = leave\_tree\_predict$actual, positi ve = "Leave")

rf\_confmat <- confusionMatrix(data = leave\_rf\_predict$predicted, reference = leave\_rf\_predict$actual, positive = "Le ave")

tree\_confmat

rf\_confmat

leave\_models\_prob <- data.frame( tree = leave\_tree\_predict$Leave,

rf = leave\_rf\_predict$Leave

)

leave\_label <- data.frame( tree = leave\_tree\_predict$actual,

rf = leave\_rf\_predict$actual

)

opar <- par() par(pty = 's')

png(file = "p")

**#Random Forest #Library Random Forest**

print(leave\_rf)

rf\_leave <- randomForest(leave\_formula, ntree = 500, importance = T, data = training\_leave) png(file = "p") plot(rf\_leave)

rf\_leave\_pred <- predict(rf\_leave, test\_leave[,-1], type= "class") rf\_leave\_pred <- predict(rf\_leave, training\_leave[,-1], type= "class") rf\_results\_table <- table(rf = rf\_leave\_pred, actual = test\_leave$Outcome) rf\_results\_table <- table(rf = rf\_leave\_pred, actual = training\_leave$Outcome) rf\_results\_table

acc\_rf <- sum(diag(rf\_results\_table)) / sum(rf\_results\_table) acc\_rf

**#Performance Evaluation Random Forest**

leave\_rf\_predict <- cbind( actual = test\_leave$Outcome, predicted = predict(rf\_leave, test\_leave[, -13], type = 'response'), predict(rf\_leave, test\_leave[, -13], type = 'response')

)

plot(leave\_rf\_predict)

#Support vector machines

qplot(Percent.Leave, Median\_Income, data = leave\_clear, colour = Outcome) qplot(C, Lab, data = leave\_clear, colour = Outcome)

**#SVMs**

**#Kernel linear**

**#Library e1071**

mymodel <- svm(Outcome ~ Mean\_Income + Median\_Income + Mean\_Tax + Median\_Tax + Pct\_Turnout + C + Lab + G reen + UKIP + LD, data = leave\_clear, kernel = "linear")

mymodel <- svm(Outcome ~ Mean\_Income + Median\_Income + Mean\_Tax + Median\_Tax + Pct\_Turnout + C + Lab + G reen + UKIP + LD, data = leave\_clear, kernel = "polynomial")

mymodel <- svm(Outcome ~ Mean\_Income + Median\_Income + Mean\_Tax + Median\_Tax + Pct\_Turnout + C + Lab + G reen + UKIP + LD, data = leave\_clear, kernel = "radial")

summary(mymodel)

plot(mymodel , data = leave\_clear,

Median\_Income~LD,

)

pred <- predict(mymodel, leave\_clear)

tab <- table(Predicted = pred, Actual = leave\_clear$Outcome) tab

1-sum(diag(tab)/sum(tab))

sum(diag(tab)/sum(tab))

**#Kernel Stigmoid**

mymodel <- svm(Outcome ~ Mean\_Income + Median\_Income + Mean\_Tax + Median\_Tax + Percent.Turnout + C + La b + Green + UKIP + LD, data = leave\_clear, kernel = "sigmoid") summary(mymodel)

plot(mymodel , data = leave\_clear,

Median\_Income~LD,

)

pred <- predict(mymodel, leave\_clear)

tab <- table(Predicted = pred, Actual = leave\_clear$Outcome) tab

sum(diag(tab)/sum(tab)) 1-sum(diag(tab)/sum(tab))

**#Tuning**

set.seed(2018)

tmodel <- tune(svm,Outcome ~ Mean\_Income + Median\_Income + Mean\_Tax + Median\_Tax + Pct\_Turnout + C + Lab + Green + UKIP + LD, data = leave\_clear, ranges = list(epsilon = seq(0,1,0.1), cost = 2^(2:5)))

plot(tmodel) summary(tmodel)

bestmodel <- tmodel$best.model summary(bestmodel)

png(file = "p")

plot(bestmodel , data = leave\_clear,

C~Pct\_Leave,

)

pred <- predict(mymodel, leave\_clear)

tab <- table(Predicted = pred, Actual = leave\_clear$Outcome) tab

1-sum(diag(tab)/sum(tab))

sum(diag(tab)/sum(tab))

p1 <- predict(rf\_leave, data = training\_leave)

head(p1)

head(training\_leave$Outcome) confusionMatrix(p1, training\_leave$Outcome) p2 <- predict(rf\_leave, test\_leave) confusionMatrix(p2, test\_leave$Outcome)

plot(rf\_leave)

legend('topright', colnames(rf\_leave$err.rate), bty = 'n', lty = c(1,2,3), col = c(1:3)) png(file = "p")

varImpPlot(rf\_leave, type = 1)