ML1_FinalProject

Yunwon Kim

```
library(tidyverse)
```

```
## Warning: package 'tidyverse' was built under R version 4.0.5
```

```
## Warning: package 'ggplot2' was built under R version 4.0.5
```

```
## Warning: package 'tibble' was built under R version 4.0.5
```

```
## Warning: package 'tidyr' was built under R version 4.0.5
```

```
## Warning: package 'readr' was built under R version 4.0.5
```

```
## Warning: package 'dplyr' was built under R version 4.0.5
```

```
## Warning: package 'forcats' was built under R version 4.0.5
```

```
library(ggplot2)
library(readxl)
library(caret)
```

```
## Warning: package 'caret' was built under R version 4.0.5
```

```
library(ROSE)
```

```
## Warning: package 'ROSE' was built under R version 4.0.5
```

```
library(car)
```

```
## Warning: package 'car' was built under R version 4.0.5
```

```
library(ROCR)
```

```
## Warning: package 'ROCR' was built under R version 4.0.5
```

```
emp <- read_xlsx("Employee_Data_Project.xlsx")
```

```
table(emp$Attrition)
```

```
##
##   No  Yes
## 3699  711
```

Data Cleaning

```
emp <- emp %>% mutate(Attrition01 = ifelse(Attrition == "Yes", 1,
                                    ifelse(Attrition == "No", 0, 0)))
emp <- emp %>%
mutate(Income = case_when(Income>=66000 & Income <=95000~ "Average",
Income > 96000 ~ "Above Average",
TRUE ~ "Below Average"))
```

```
#remove NA's
emp <- emp %>% subset(NumCompaniesWorked != "NA"& TotalWorkingYears != "NA"&
                      EnvironmentSatisfaction != "NA"&
                      JobSatisfaction != "NA")
emp$NumCompaniesWorked <- as.numeric(emp$NumCompaniesWorked)
emp$TotalWorkingYears <- as.numeric(emp$TotalWorkingYears)
emp$EnvironmentSatisfaction <- as.numeric(emp$EnvironmentSatisfaction)
emp$JobSatisfaction <- as.numeric(emp$JobSatisfaction)
emp$BusinessTravel <- as.factor(emp$BusinessTravel)
emp$Gender <- as.factor(emp$Gender)
emp$MaritalStatus <- as.factor(emp$MaritalStatus)
emp$JobLevel <- as.factor(emp$JobLevel)
emp$JobSatisfaction <- as.factor(emp$JobSatisfaction)
emp$EnvironmentSatisfaction <- as.factor(emp$EnvironmentSatisfaction)
emp$Attrition <- as.factor(emp$Attrition)
emp$Attrition01 <- as.factor(emp$Attrition01)
emp$Income <- as.factor(emp$Income)
```

Data Partition

```
set.seed(123)

index <- createDataPartition(emp$Attrition01, p=0.7, list=FALSE)
train <- emp[index,]
test <- emp[-index,]
```

```
#check for missing values
sum(is.na(train))
```

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

Bagging

- Alex is doing this part.

```
library(randomForest)
```

```
## Warning: package 'randomForest' was built under R version 4.0.4
```

```
#Clean Data
train_clean <- train %>% select(Attrition01,Age,BusinessTravel,DistanceFromHome ,Education,Gende
r,JobLevel,MaritalStatus,Income,NumCompaniesWorked ,StandardHours,TotalWorkingYears,TrainingTime
sLastYear,YearsAtCompany ,YearsWithCurrManager,EnvironmentSatisfaction,JobSatisfaction)

test_clean <- test %>% select(Attrition01,Age,BusinessTravel,DistanceFromHome ,Education,Gender,
JobLevel,MaritalStatus,Income,NumCompaniesWorked ,StandardHours,TotalWorkingYears,TrainingTimesL
astYear,YearsAtCompany ,YearsWithCurrManager,EnvironmentSatisfaction,JobSatisfaction)
```

```
under_train <- ovun.sample(Attrition01 ~Age+BusinessTravel+DistanceFromHome +Education+Gender+Jo
bLevel+MaritalStatus+Income+NumCompaniesWorked +StandardHours+TotalWorkingYears+TrainingTimesLas
tYear+YearsAtCompany +YearsWithCurrManager+EnvironmentSatisfaction+JobSatisfaction,
                     data = train_clean,
                     method = "under", N = 1000)$data
table(under_train$Attrition)
```

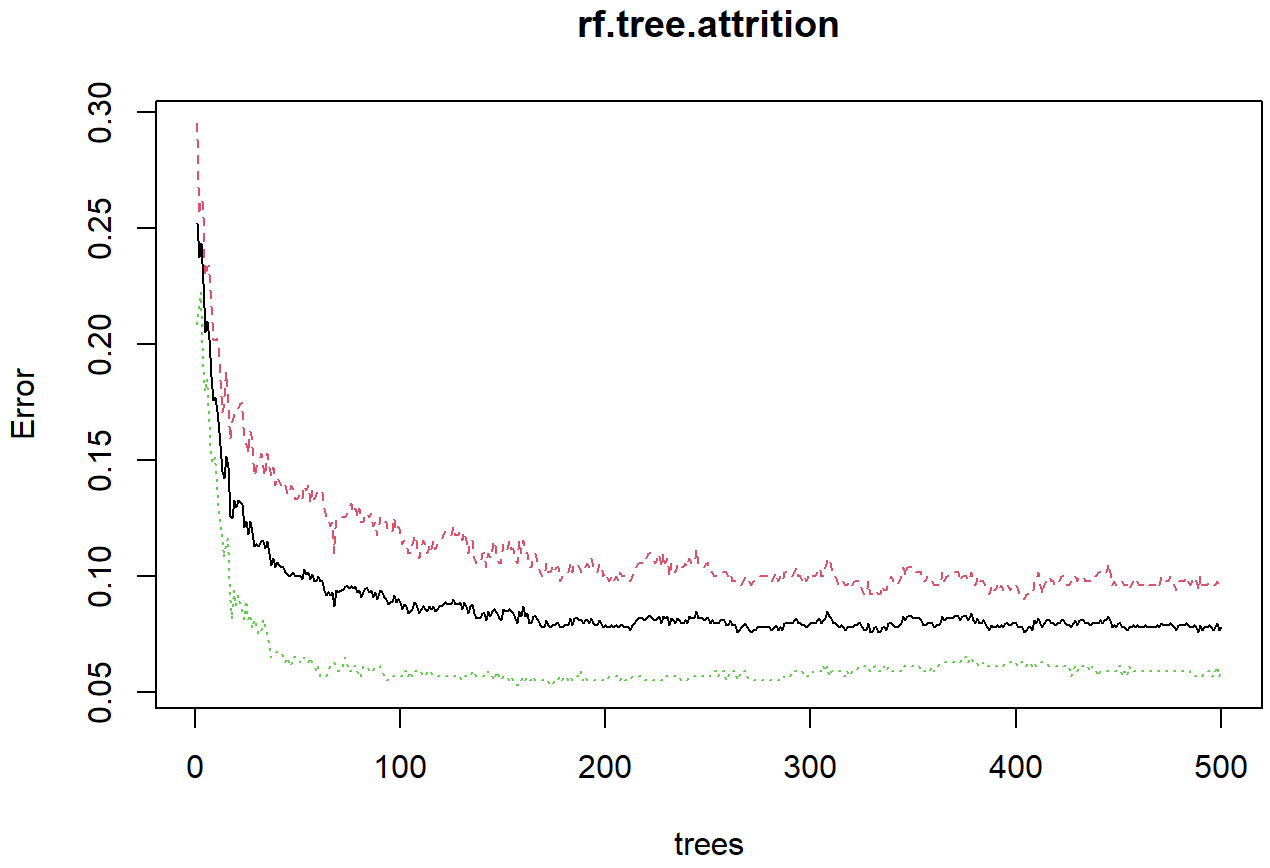
```
##
##   0   1
## 510 490
```

Random Forest

```
set.seed(123)
rf.tree.attrition <- randomForest(Attrition01~Age+BusinessTravel+DistanceFromHome+Education+Gend
er+JobLevel+MaritalStatus+Income+NumCompaniesWorked +StandardHours+TotalWorkingYears+TrainingTim
esLastYear+YearsAtCompany +YearsWithCurrManager+EnvironmentSatisfaction+JobSatisfaction, data=un
der_train, ntree=500)
rf.tree.attrition
```

```
##
## Call:
##  randomForest(formula = Attrition01 ~ Age + BusinessTravel + DistanceFromHome +      Educatio
n + Gender + JobLevel + MaritalStatus + Income +      NumCompaniesWorked + StandardHours + Total
WorkingYears +      TrainingTimesLastYear + YearsAtCompany + YearsWithCurrManager +      Environ
mentSatisfaction + JobSatisfaction, data = under_train,      ntree = 500)
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 7.8%
## Confusion matrix:
##     0   1 class.error
## 0 461  49  0.09607843
## 1  29 461  0.05918367
```

```
plot(rf.tree.attrition)
```



- For the plot above, the black line corresponds to out of bag error, the green line is the error for class 1, meaning error for Attrition=Yes, and the red line is the error for class 0, meaning error for Attrition=No.
- Model Tuning

```
#Finding out the number of trees that minimizes the error
set.seed(123)
ntrees <- which.min(rf.tree.attrition$err.rate[,1])
rf.tree.attrition <- randomForest(Attrition01~Age+BusinessTravel+DistanceFromHome +Education+Gen
der+JobLevel+MaritalStatus+Income+NumCompaniesWorked +StandardHours+TotalWorkingYears+TrainingTi
mesLastYear+YearsAtCompany +YearsWithCurrManager+EnvironmentSatisfaction+JobSatisfaction, data=u
nder_train, ntree=ntrees)
rf.tree.attrition
```

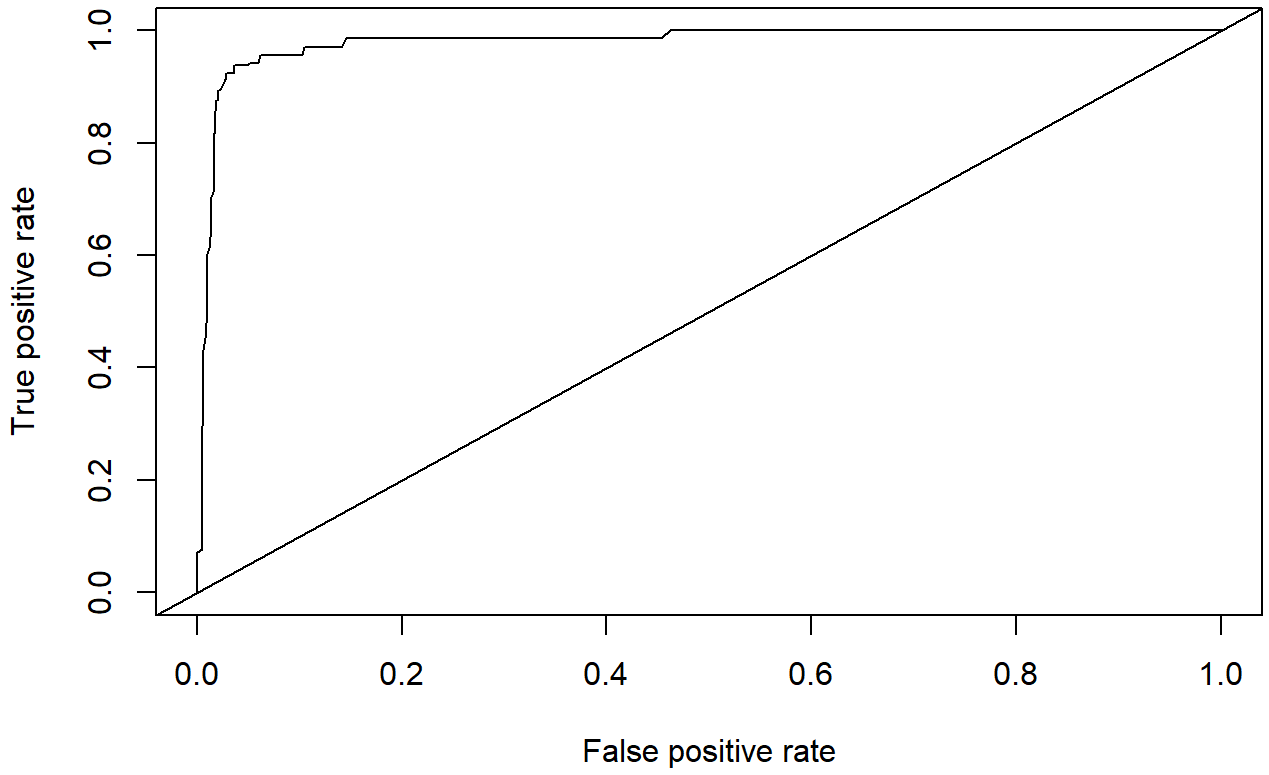
```
##
## Call:
##  randomForest(formula = Attrition01 ~ Age + BusinessTravel + DistanceFromHome +      Educatio
n + Gender + JobLevel + MaritalStatus + Income +      NumCompaniesWorked + StandardHours + Total
WorkingYears +      TrainingTimesLastYear + YearsAtCompany + YearsWithCurrManager +      Environ
mentSatisfaction + JobSatisfaction, data = under_train,      ntree = ntrees)
##                Type of random forest: classification
##                      Number of trees: 264
## No. of variables tried at each split: 4
##
##         OOB estimate of  error rate: 7.6%
## Confusion matrix:
##     0   1 class.error
## 0 461  49  0.09607843
## 1  27 463  0.05510204
```

```
rf.trees.predict <- predict(rf.tree.attrition, newdata=test_clean,
type="class")
confusionMatrix(rf.trees.predict, test_clean$Attrition01, positive="1")
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 987   9
##          1 104 200
##
##                Accuracy : 0.9131
##                  95% CI : (0.8964, 0.9278)
##     No Information Rate : 0.8392
##     P-Value [Acc > NIR] : 3.904e-15
##
##                   Kappa : 0.7279
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9569
##             Specificity : 0.9047
##          Pos Pred Value : 0.6579
##          Neg Pred Value : 0.9910
##              Prevalence : 0.1608
##          Detection Rate : 0.1538
##    Detection Prevalence : 0.2338
##       Balanced Accuracy : 0.9308
##
##        'Positive' Class : 1
##
```

- The above output is our final result for random forest. Accuracy = 0.9685, etc.. Let me know if you need help with interpretating or understanding the result.

```
require(ROCR)
tree.attrition.predict2 <- predict(rf.tree.attrition, test_clean,
type="prob")
predROC <- prediction(tree.attrition.predict2[,2],
test_clean$Attrition01)
perfROC <- performance(predROC, "tpr", "fpr")
plot(perfROC)
abline(a=0, b=1)
```




```
#Calculate the area under the curve
perfROC <- performance(predROC, "auc")
perfROC@y.values[[1]]
```

```
## [1] 0.9793986
```

```
varImp(rf.tree.attrition)%>% arrange(desc(Overall))
```

```
##                          Overall
## Age                    57.375139
## TotalWorkingYears      53.629785
## YearsAtCompany         51.376746
## DistanceFromHome       47.968157
## YearsWithCurrManager   37.455916
## NumCompaniesWorked     35.369135
## EnvironmentSatisfaction 31.773235
## TrainingTimesLastYear  29.765292
## JobLevel               29.712792
## JobSatisfaction        29.618471
## MaritalStatus          27.732694
## Education              23.039365
## BusinessTravel         19.632843
## Income                 14.507939
## Gender                  8.412019
## StandardHours           0.000000
```

```
varImpPlot(rf.tree.attrition)
```