Churn Prediction Model for Telecom Companies

Group\_8

2022-12-14

setwd("~/Desktop/R/64036-002/Group\_proj") #set working directory  
Customer <- read.csv("Churn\_Train.csv") #load the data  
load("Customers\_To\_Predict.RData")  
summary(Customer)

## state account\_length area\_code international\_plan  
## Length:3333 Min. :-209.00 Length:3333 Length:3333   
## Class :character 1st Qu.: 72.00 Class :character Class :character   
## Mode :character Median : 100.00 Mode :character Mode :character   
## Mean : 97.32   
## 3rd Qu.: 127.00   
## Max. : 243.00   
## NA's :501   
## voice\_mail\_plan number\_vmail\_messages total\_day\_minutes total\_day\_calls  
## Length:3333 Min. :-10.000 Min. : 0.0 Min. : 0.0   
## Class :character 1st Qu.: 0.000 1st Qu.: 149.3 1st Qu.: 87.0   
## Mode :character Median : 0.000 Median : 190.5 Median :101.0   
## Mean : 7.333 Mean : 418.9 Mean :100.3   
## 3rd Qu.: 16.000 3rd Qu.: 237.8 3rd Qu.:114.0   
## Max. : 51.000 Max. :2185.1 Max. :165.0   
## NA's :200 NA's :200 NA's :200   
## total\_day\_charge total\_eve\_minutes total\_eve\_calls total\_eve\_charge  
## Min. : 0.00 Min. : 0.0 Min. : 0.0 Min. : 0.00   
## 1st Qu.:24.45 1st Qu.: 170.5 1st Qu.: 87.0 1st Qu.:14.14   
## Median :30.65 Median : 209.9 Median :100.0 Median :17.09   
## Mean :30.63 Mean : 324.3 Mean :100.1 Mean :17.08   
## 3rd Qu.:36.84 3rd Qu.: 257.6 3rd Qu.:114.0 3rd Qu.:20.00   
## Max. :59.64 Max. :1244.2 Max. :170.0 Max. :30.91   
## NA's :200 NA's :301 NA's :200 NA's :200   
## total\_night\_minutes total\_night\_calls total\_night\_charge total\_intl\_minutes  
## Min. : 23.2 Min. : 33.0 Min. : 1.040 Min. : 0.00   
## 1st Qu.:167.3 1st Qu.: 87.0 1st Qu.: 7.530 1st Qu.: 8.50   
## Median :201.4 Median :100.0 Median : 9.060 Median :10.30   
## Mean :201.2 Mean :100.1 Mean : 9.054 Mean :10.23   
## 3rd Qu.:235.3 3rd Qu.:113.0 3rd Qu.:10.590 3rd Qu.:12.10   
## Max. :395.0 Max. :175.0 Max. :17.770 Max. :20.00   
## NA's :200 NA's :200 NA's :200   
## total\_intl\_calls total\_intl\_charge number\_customer\_service\_calls  
## Min. : 0.00 Min. :0.000 Min. :0.000   
## 1st Qu.: 3.00 1st Qu.:2.300 1st Qu.:1.000   
## Median : 4.00 Median :2.780 Median :1.000   
## Mean : 4.47 Mean :2.762 Mean :1.561   
## 3rd Qu.: 6.00 3rd Qu.:3.270 3rd Qu.:2.000   
## Max. :20.00 Max. :5.400 Max. :9.000   
## NA's :301 NA's :200 NA's :200   
## churn   
## Length:3333   
## Class :character   
## Mode :character   
##   
##   
##   
##

**Data Exploration**

xtabs(~churn+state, data = Customer) #churn distribution by state

## state  
## churn AK AL AR AZ CA CO CT DC DE FL GA HI IA ID IL IN KS KY LA MA MD ME MI MN  
## no 49 72 44 60 25 57 62 49 52 55 46 50 41 64 53 62 57 51 47 54 53 49 57 69  
## yes 3 8 11 4 9 9 12 5 9 8 8 3 3 9 5 9 13 8 4 11 17 13 16 15  
## state  
## churn MO MS MT NC ND NE NH NJ NM NV NY OH OK OR PA RI SC SD TN TX UT VA VT WA  
## no 56 51 54 57 56 56 47 50 56 52 68 68 52 67 37 59 46 52 48 54 62 72 65 52  
## yes 7 14 14 11 6 5 9 18 6 14 15 10 9 11 8 6 14 8 5 18 10 5 8 14  
## state  
## churn WI WV WY  
## no 71 96 68  
## yes 7 10 9

**Comment**  
Early observations for churn customers:   
1) There is 19% of churn rate in the training data   
2) AR, KS, MA, MD, ME, MI, MN, MS, MT, NC, NJ, NV, NY, OH, OR, SC, TX, WA have higher churn rate.

**Data Preparation**

#Clean and transform the data   
Customer <- Customer[, -c(1:3)] #delete state, account length and area code  
  
#Change negative data in account\_length and number\_vmail\_message to positive  
Customer <- Customer%>%  
 #mutate(account\_length = ifelse(account\_length <0, abs(account\_length), account\_length))%>%  
 mutate(number\_vmail\_messages = ifelse(number\_vmail\_messages <0, abs(number\_vmail\_messages), number\_vmail\_messages))  
  
#Change binary columns to 0 and 1  
Customer$international\_plan <- ifelse(Customer$international\_plan=="yes", 1, 0)  
Customer$voice\_mail\_plan <- ifelse(Customer$voice\_mail\_plan=="yes", 1, 0)  
  
#Change data attribute from character to factor, the data is coded as 1 as no and 2 as yes  
Customer$churn <- as.factor(Customer$churn)   
  
#impute missing values with mean  
Customer[, c(3:10, 11:16)] <- Customer[, c(3:10, 11:16)]%>%  
 mutate\_if(is.numeric, function(x) ifelse(is.na(x), median(x, na.rm = T), x))

**Data partition**

#Partition the given training data into 70% training data and 30% testing data  
set.seed(111)  
index\_train <- createDataPartition(Customer$churn, p=0.7, list= F)  
Cust\_train <- Customer[index\_train, ]  
Cust\_test <- Customer[-index\_train, ]

**Run logistic regression model**

set.seed(1)  
log\_model <- glm(churn~., data = Cust\_train, family = 'binomial')

**Run knn model**

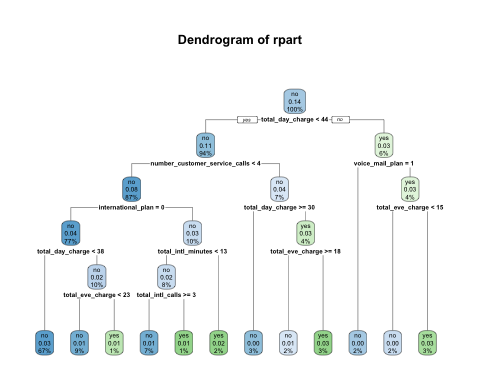
set.seed(2)  
knn\_model <- train(data = Cust\_train, churn~., method = "knn", metric = "Accuracy",   
 trControl= trainControl(), tuneGrid = NULL, tuneLength = 3)

**Run NB model**

library(e1071)  
set.seed(3)  
nb\_model <- naiveBayes(churn~., data = Cust\_train)  
Predict\_test\_labels\_nb <- predict(nb\_model, Cust\_test, type = "raw")

**Run Decision Tree**

set.seed(4)  
library(rpart)  
library(rpart.plot)  
#agnes or hclust object does not work with later prediction  
dt\_model <- rpart(churn~., data = Cust\_test,method = "class") #class for binary  
rpart.plot(dt\_model, extra = 110, main = "Dendrogram of rpart")

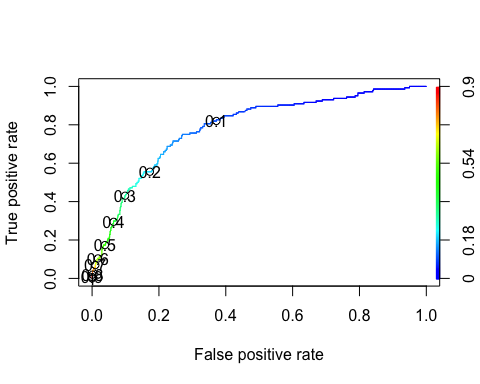


**Model Testing**

#Test the logistic regression model and return in probability  
log\_test\_prob <- predict(log\_model, Cust\_test, type = "response")  
#log\_test <- cbind(Cust\_test, log\_test\_prob)  
  
#Test the knn model  
knn\_test\_prob <- predict(knn\_model, Cust\_test, type = "prob")  
  
#Test the nb model  
nb\_test\_prob <- predict(nb\_model, Cust\_test, type = "raw")  
  
#Test the dt model- (predict does not apply to "hclust" or "agnes" object)  
dt\_test\_prob <- predict(dt\_model, Cust\_test, type = "prob")

**Model Comparison: Thresholding, best cutoff point, confusion table and ROC**

#logistic regression   
pred\_log\_test <- prediction(log\_test\_prob, Cust\_test$churn)#create prediction obj  
  
#TPR FPR plot  
roc\_perf\_log\_test <- performance(pred\_log\_test, measure = "tpr", x.measure = "fpr")  
plot(roc\_perf\_log\_test,colorize=TRUE,print.cutoffs.at=seq(0.1,by=0.1))



#TPR/FPR cutoff graph<br>   
  
#cut-off trade-off between FPR and TPR, we want to reduce False Negative  
cost\_perf = performance(pred\_log\_test, "cost")   
pred\_log\_test@cutoffs[[1]][which.min(cost\_perf@y.values[[1]])]#best cutoff 0.539

## 47   
## 0.774522

#Logistic regression AUC value  
auc.perf = performance(pred\_log\_test, measure = "auc")  
auc.perf@y.values

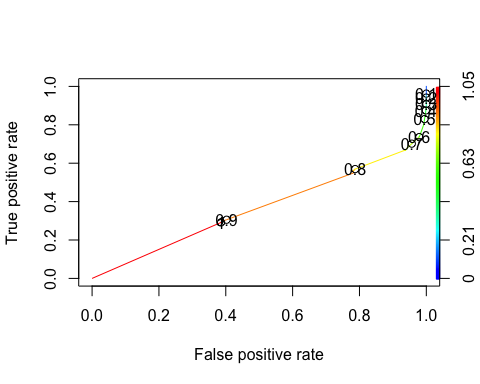
## [[1]]  
## [1] 0.7860624

#Confusion table  
confusionMatrix(as.factor(ifelse(log\_test\_prob>0.5064, "yes", "no")), Cust\_test$churn, positive = "yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction no yes  
## no 825 119  
## yes 30 25  
##   
## Accuracy : 0.8509   
## 95% CI : (0.8272, 0.8724)  
## No Information Rate : 0.8559   
## P-Value [Acc > NIR] : 0.6927   
##   
## Kappa : 0.1864   
##   
## Mcnemar's Test P-Value : 5.626e-13   
##   
## Sensitivity : 0.17361   
## Specificity : 0.96491   
## Pos Pred Value : 0.45455   
## Neg Pred Value : 0.87394   
## Prevalence : 0.14414   
## Detection Rate : 0.02503   
## Detection Prevalence : 0.05506   
## Balanced Accuracy : 0.56926   
##   
## 'Positive' Class : yes   
##

**Logistic Regression Metric**  
True Positive (TP) = 26   
True Negative (TN) = 835   
False Positive (FP) = 20   
False Negative (FN) = 118   
Miscalculations = 138   
Accuracy = 86.19%   
Sensitivity = 18.06%   
Specificity = 97.66%

#KNN   
pred\_knn\_test <- prediction(knn\_test\_prob[,1], Cust\_test$churn)  
#plot TPR - FPR  
roc\_perf\_knn\_test <- performance(pred\_knn\_test, measure = "tpr", x.measure = "fpr")  
plot(roc\_perf\_knn\_test,colorize=TRUE,print.cutoffs.at=seq(0.1,by=0.1))



#Calculate ROC value for binary classifier  
roc.curve(Cust\_test$churn, knn\_test\_prob[,1], plotit= F)

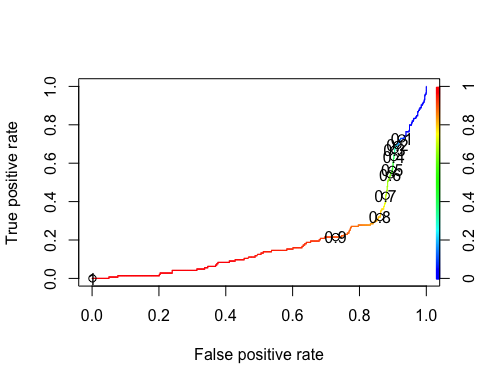
## Area under the curve (AUC): 0.635

confusionMatrix(as.factor(ifelse(knn\_test\_prob[,1]>0.94, "yes", "no")), Cust\_test$churn, positive = "yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction no yes  
## no 525 102  
## yes 330 42  
##   
## Accuracy : 0.5676   
## 95% CI : (0.5362, 0.5986)  
## No Information Rate : 0.8559   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : -0.0569   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.29167   
## Specificity : 0.61404   
## Pos Pred Value : 0.11290   
## Neg Pred Value : 0.83732   
## Prevalence : 0.14414   
## Detection Rate : 0.04204   
## Detection Prevalence : 0.37237   
## Balanced Accuracy : 0.45285   
##   
## 'Positive' Class : yes   
##

**KNN Metric**  
True Positive (TP) = 32   
True Negative (TN) = 585   
False Positive (FP) = 270   
False Negative (FN) = 112   
Miscalculations = 382   
Accuracy = 61.76%   
Specificity = 68.42%   
Sensitivity = 22.22%

#Naive Bayes   
pred\_nb\_test <- prediction(nb\_test\_prob[,1], Cust\_test$churn)  
roc\_perf\_nb\_test <- performance(pred\_nb\_test, measure = "tpr", x.measure = "fpr")  
plot(roc\_perf\_nb\_test,colorize=TRUE,print.cutoffs.at=seq(0.1,by=0.1))



#Calculate ROC value for binary classifier  
roc.curve(Cust\_test$churn, nb\_test\_prob[,1], plotit= F)

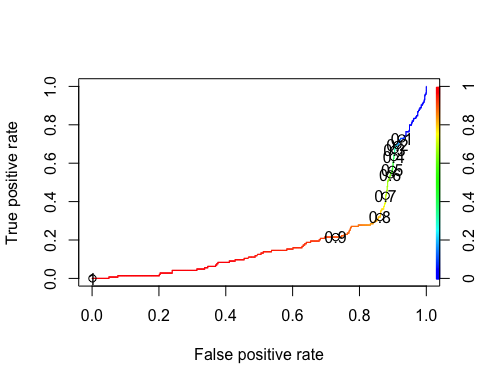
## Area under the curve (AUC): 0.808

confusionMatrix(as.factor(ifelse(nb\_test\_prob[,1]>0.95, "yes", "no")), Cust\_test$churn, positive = "yes")

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction no yes  
## no 412 124  
## yes 443 20  
##   
## Accuracy : 0.4324   
## 95% CI : (0.4014, 0.4638)  
## No Information Rate : 0.8559   
## P-Value [Acc > NIR] : 1   
##   
## Kappa : -0.1974   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.13889   
## Specificity : 0.48187   
## Pos Pred Value : 0.04320   
## Neg Pred Value : 0.76866   
## Prevalence : 0.14414   
## Detection Rate : 0.02002   
## Detection Prevalence : 0.46346   
## Balanced Accuracy : 0.31038   
##   
## 'Positive' Class : yes   
##

**Naive Bayes Metric**  
True Positive (TP) = 14   
True Negative (TN) = 471   
False Positive (FP) = 384   
False Negative (FN) = 130   
Miscalculations = 514   
Accuracy = 48.55%   
Specificity = 55.09%   
Sensitivity = 9.72%

#decision tree (dt): create prediction object for ROCR evaluation   
pred\_dt\_test <- prediction(dt\_test\_prob[,1], Cust\_test$churn)  
roc\_perf\_dt\_test <- performance(pred\_dt\_test, measure = "tpr", x.measure = "fpr")  
plot(roc\_perf\_nb\_test,colorize=TRUE,print.cutoffs.at=seq(0.1,by=0.1))



#Calculate ROC value for binary classifier  
roc.curve(Cust\_test$churn, dt\_test\_prob[,1], plotit= F)

## Area under the curve (AUC): 0.866

confusionMatrix(as.factor(ifelse(dt\_test\_prob[,1]>0.967, "yes", "no")), Cust\_test$churn, positive = "yes")

## Warning in confusionMatrix.default(as.factor(ifelse(dt\_test\_prob[, 1] > : Levels  
## are not in the same order for reference and data. Refactoring data to match.

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction no yes  
## no 855 144  
## yes 0 0  
##   
## Accuracy : 0.8559   
## 95% CI : (0.8325, 0.8771)  
## No Information Rate : 0.8559   
## P-Value [Acc > NIR] : 0.5222   
##   
## Kappa : 0   
##   
## Mcnemar's Test P-Value : <2e-16   
##   
## Sensitivity : 0.0000   
## Specificity : 1.0000   
## Pos Pred Value : NaN   
## Neg Pred Value : 0.8559   
## Prevalence : 0.1441   
## Detection Rate : 0.0000   
## Detection Prevalence : 0.0000   
## Balanced Accuracy : 0.5000   
##   
## 'Positive' Class : yes   
##

**Decision Tree Metric**  
True Positive (TP) = 0   
True Negative (TN) = 836   
False Positive (FP) = 19   
False Negative (FN) = 144   
Miscalculations = 163   
Accuracy = 83.68%   
Specificity = 97.78%   
Sensitivity = 0.0%

**Conclusion**  
Decision Tree and Logistic Regression model have good performance in accuracy, ROC value and specificity. LR model has a better sensitivity but DT has better ROC value which means the model is better. We will choose to apply DT on the test data.

**Test Data Prediction**

#updating the binary variables  
Customers\_To\_Predict$international\_plan <- ifelse(Customers\_To\_Predict$international\_plan =="yes", 1, 0)  
Customers\_To\_Predict$voice\_mail\_plan <- ifelse(Customers\_To\_Predict$voice\_mail\_plan =="yes", 1, 0)  
  
#apply DT model on the test data  
customer\_test <- predict(dt\_model, Customers\_To\_Predict, type = "prob")  
x <- cbind(Customers\_To\_Predict, customer\_test)  
  
#set cutoff value  
x$prob <- ifelse(x$'no'>0.967, "no", "yes")  
Customers\_To\_Predict$churn\_prob <- x$prob

write\_xlsx(Customers\_To\_Predict, "Customer\_to\_Predict")

**Conclusion**  
The test data has been updated with additional variable “churn\_prob” which provide the probability of churn for each customer. The model aims to reduce false negatives and tolerates more on false positives. It will cost more on the company to miss a churning customer than to mis-classify un-churning customers. If provided with the promotion cost, cost for false positive and false negative, we can further calculate the total saving cost for the company.