Thera Bank - Loan Purchase Modeling

Adeleke Dare

8/31/2020

The objective of this exercise is that Thera Bank want to increase the asset of the bank by increasing the borrowers base (asset customers) to bring in more loan business that will lead to earning through interest on the loan. The problem the model is trying solve is to identify the potential customers who have a higher probability of purchasing the loan. We fitted four machine learning algorithms and selected the best algorithm to explain relevant features to Staff promotion using the provided data set.

Let’s start by loading the packages and data set into R

library(readxl)  
library(tidyverse)

library(gridExtra)

library(recipes)

library(caret)

library(rpart)  
library(cluster)  
library(Rtsne)  
library(rpart.plot)  
library(randomForest)

library(AUC)

library(lime)

library(corrr)  
library(tidyquant)

library(pROC)

library(party)

# Import the dataset   
loan <- read\_excel("C:/Users/DHREY/Desktop/R-ass/Thera-Bank\_Personal\_Loan\_Modelling-dataset-1.xlsx", sheet = 2)  
  
str(loan)

## tibble [5,000 x 14] (S3: tbl\_df/tbl/data.frame)  
## $ ID : num [1:5000] 1 2 3 4 5 6 7 8 9 10 ...  
## $ Age (in years) : num [1:5000] 25 45 39 35 35 37 53 50 35 34 ...  
## $ Experience (in years): num [1:5000] 1 19 15 9 8 13 27 24 10 9 ...  
## $ Income (in K/month) : num [1:5000] 49 34 11 100 45 29 72 22 81 180 ...  
## $ ZIP Code : num [1:5000] 91107 90089 94720 94112 91330 ...  
## $ Family members : num [1:5000] 4 3 1 1 4 4 2 1 3 1 ...  
## $ CCAvg : num [1:5000] 1.6 1.5 1 2.7 1 0.4 1.5 0.3 0.6 8.9 ...  
## $ Education : num [1:5000] 1 1 1 2 2 2 2 3 2 3 ...  
## $ Mortgage : num [1:5000] 0 0 0 0 0 155 0 0 104 0 ...  
## $ Personal Loan : num [1:5000] 0 0 0 0 0 0 0 0 0 1 ...  
## $ Securities Account : num [1:5000] 1 1 0 0 0 0 0 0 0 0 ...  
## $ CD Account : num [1:5000] 0 0 0 0 0 0 0 0 0 0 ...  
## $ Online : num [1:5000] 0 0 0 0 0 1 1 0 1 0 ...  
## $ CreditCard : num [1:5000] 0 0 0 0 1 0 0 1 0 0 ...

The data contains 5000 observations and 14 variables. The response variable “Personal Loan” is seen as a numeric likewise all other predictor variable. The data dictionary revealed that Securities Account, CD (certificate of deposit) Account, Online banking and CreditCard are Yes or No type of variables, hence, the need to convert them to factor or classification variables. Treating them as numeric will undermine the finding.

**Exploratory Data Analysis**

The summary function will be basically used for univariate analysis.

summary(loan)

## ID Age (in years) Experience (in years) Income (in K/month)  
## Min. : 1 Min. :23.00 Min. :-3.0 Min. : 8.00   
## 1st Qu.:1251 1st Qu.:35.00 1st Qu.:10.0 1st Qu.: 39.00   
## Median :2500 Median :45.00 Median :20.0 Median : 64.00   
## Mean :2500 Mean :45.34 Mean :20.1 Mean : 73.77   
## 3rd Qu.:3750 3rd Qu.:55.00 3rd Qu.:30.0 3rd Qu.: 98.00   
## Max. :5000 Max. :67.00 Max. :43.0 Max. :224.00   
##   
## ZIP Code Family members CCAvg Education   
## Min. : 9307 Min. :1.000 Min. : 0.000 Min. :1.000   
## 1st Qu.:91911 1st Qu.:1.000 1st Qu.: 0.700 1st Qu.:1.000   
## Median :93437 Median :2.000 Median : 1.500 Median :2.000   
## Mean :93153 Mean :2.397 Mean : 1.938 Mean :1.881   
## 3rd Qu.:94608 3rd Qu.:3.000 3rd Qu.: 2.500 3rd Qu.:3.000   
## Max. :96651 Max. :4.000 Max. :10.000 Max. :3.000   
## NA's :18   
## Mortgage Personal Loan Securities Account CD Account   
## Min. : 0.0 Min. :0.000 Min. :0.0000 Min. :0.0000   
## 1st Qu.: 0.0 1st Qu.:0.000 1st Qu.:0.0000 1st Qu.:0.0000   
## Median : 0.0 Median :0.000 Median :0.0000 Median :0.0000   
## Mean : 56.5 Mean :0.096 Mean :0.1044 Mean :0.0604   
## 3rd Qu.:101.0 3rd Qu.:0.000 3rd Qu.:0.0000 3rd Qu.:0.0000   
## Max. :635.0 Max. :1.000 Max. :1.0000 Max. :1.0000   
##   
## Online CreditCard   
## Min. :0.0000 Min. :0.000   
## 1st Qu.:0.0000 1st Qu.:0.000   
## Median :1.0000 Median :0.000   
## Mean :0.5968 Mean :0.294   
## 3rd Qu.:1.0000 3rd Qu.:1.000   
## Max. :1.0000 Max. :1.000   
##

From the above result, it was deduced that there are issues in our data set that have to be treated before moving forward.

1. Family members variable has missing values. This will be treated by using impute with mean method
2. Age, Education, Income and Year of Experience will be categorized into difference level as it is advisable to treat this kind of variable like that.
3. Variables like Personal Loan, Securities Account, CD account, Online, Credit card will be preferred in categorical variable rather than numeric since they are "YES" or "NO" type of response.
4. Un-useful variables like ID, zip code will be removed from our data set
5. Using the quantile values likewise the difference between mean and median, there are outlier in the data set which is Mortgage.
6. Most variables have to be renamed. Age (in years), Experience (in years), Income (in K/month) etc.
7. Education has been stated in the data dictionary to be the following Levels. 1: Undergrad; 2: Graduate; 3: Advanced/Professional hence it will also be categorized.
8. The code below will be used in treating and transforming our data set.

# Imputing mean value to fill the missing value in family members  
loan$`Family members`[is.na(loan$`Family members`)] <- mean(loan$`Family members`, na.rm = T)  
  
# Categorizing and renaming the variable Age  
loan<- loan %>% mutate(agegroup = case\_when(`Age (in years)` >= 18 & `Age (in years)` <= 35 ~ '1', `Age (in years)` >= 36 & `Age (in years)` <= 52 ~ '2', `Age (in years)` >= 53 & `Age (in years)` <= 100 ~ '3'))  
  
loan$agegroup<- factor(loan$agegroup, labels=c("Young", "Middle-Aged","Old"))  
  
# Categorizing and renaming the variable Income  
loan<- loan %>% mutate(income = case\_when(`Income (in K/month)` >= 1 & `Income (in K/month)` <= 15 ~ '1', `Income (in K/month)` >= 16 & `Income (in K/month)` <= 30 ~ '2', `Income (in K/month)` >= 31 & `Income (in K/month)` <= 75 ~ '3', `Income (in K/month)` >= 76 & `Income (in K/month)` <= 300 ~ '4'))  
  
loan$income <- factor(loan$income, labels = c("Lower Class","Working Class","Lower Middle Class","Upper Middle Class"))

# Categorizing and renaming the variable Personal Loan  
loan <- loan %>% mutate(personalLoan = case\_when(`Personal Loan` == 0 ~ '1', `Personal Loan` == 1 ~ '2'))  
loan$personalLoan <- factor(loan$personalLoan, labels = c("No", "Yes"))

#Let convert the numeric variable to factor since they are YES(1) or NO(0), hence, Categorical variable is advisable  
loan$Online <- as.factor(loan$Online)  
loan$CreditCard <- as.factor(loan$CreditCard)  
loan$`Securities Account` <- as.factor(loan$`Securities Account`)  
loan$`CD Account` <- as.factor(loan$`CD Account`)  
loan$Education <- as.factor(loan$Education)  
  
# Labeling of levels in education variable  
loan$Education <- factor(loan$Education, labels = c("Undergrad", "Graduate", "Advanced/Professional"))  
  
# Rename of variables to get rid of the space  
loan <- rename(loan, securitiesAcct = `Securities Account`)  
loan <- rename(loan, CDAcct = `CD Account`)  
loan <- rename(loan, familyMember = `Family members`)

# Categorizing the year of experience to difference levels and labeling   
  
# Labeling of levels in Experience variable  
loan$Exp\_Agegroup<- factor(loan$Exp\_Agegroup, labels=c("0yrs\_Exp", "Between 1-10yrs", "Between 11-20yrs","Between 21-30yrs", "Between 31-40yrs","Between 41-50yrs"))  
  
# Remove the variable that have been transformed and the useless variables  
loan <- loan[, -1] # ID  
loan <- loan[, -1] # Age  
loan <- loan[, -1] # Experience  
loan <- loan[, -2] # Zip code  
loan <- loan[, -1] # Income  
loan <- loan[, -5] # Personal Loan  
str(loan)

## tibble [5,000 x 12] (S3: tbl\_df/tbl/data.frame)  
## $ familyMember : num [1:5000] 4 3 1 1 4 4 2 1 3 1 ...  
## $ CCAvg : num [1:5000] 1.6 1.5 1 2.7 1 0.4 1.5 0.3 0.6 8.9 ...  
## $ Education : Factor w/ 3 levels "Undergrad","Graduate",..: 1 1 1 2 2 2 2 3 2 3 ...  
## $ Mortgage : num [1:5000] 0 0 0 0 0 155 0 0 104 0 ...  
## $ securitiesAcct: Factor w/ 2 levels "0","1": 2 2 1 1 1 1 1 1 1 1 ...  
## $ CDAcct : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 1 ...  
## $ Online : Factor w/ 2 levels "0","1": 1 1 1 1 1 2 2 1 2 1 ...  
## $ CreditCard : Factor w/ 2 levels "0","1": 1 1 1 1 2 1 1 2 1 1 ...  
## $ agegroup : Factor w/ 3 levels "Young","Middle-Aged",..: 1 2 2 1 1 2 3 2 1 1 ...  
## $ income : Factor w/ 4 levels "Lower Class",..: 3 3 1 4 3 2 3 2 4 4 ...  
## $ personalLoan : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 1 1 1 2 ...  
## $ Exp\_Agegroup : Factor w/ 6 levels "0yrs\_Exp","Between 1-10yrs",..: 2 3 3 2 2 3 4 4 2 2 ...

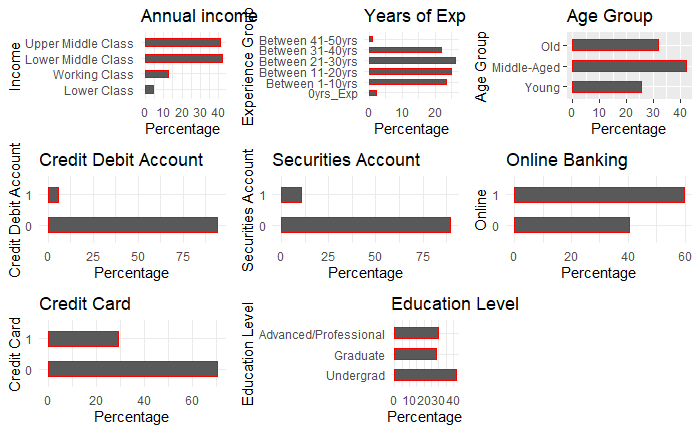
summary(loan)

## familyMember CCAvg Education Mortgage   
## Min. :1.000 Min. : 0.000 Undergrad :2096 Min. : 0.0   
## 1st Qu.:1.000 1st Qu.: 0.700 Graduate :1403 1st Qu.: 0.0   
## Median :2.000 Median : 1.500 Advanced/Professional:1501 Median : 0.0   
## Mean :2.397 Mean : 1.938 Mean : 56.5   
## 3rd Qu.:3.000 3rd Qu.: 2.500 3rd Qu.:101.0   
## Max. :4.000 Max. :10.000 Max. :635.0   
## securitiesAcct CDAcct Online CreditCard agegroup   
## 0:4478 0:4698 0:2016 0:3530 Young :1274   
## 1: 522 1: 302 1:2984 1:1470 Middle-Aged:2130   
## Old :1596   
##   
##   
##   
## income personalLoan Exp\_Agegroup   
## Lower Class : 225 No :4520 0yrs\_Exp : 118   
## Working Class : 640 Yes: 480 Between 1-10yrs :1171   
## Lower Middle Class:2093 Between 11-20yrs:1253   
## Upper Middle Class:2042 Between 21-30yrs:1301   
## Between 31-40yrs:1103   
## Between 41-50yrs: 54

The above is the new summary of our data set after the exploratory data analysis.

Let’s examine the distribution of the data set using graph. Percentage Value will be used for Classification variables While Central Tendency will be used for Continuous or numeric variable using the ggplot2 package.

#Classification variables distribution  
expr <- ggplot(loan, aes(x=Exp\_Agegroup)) + ggtitle("Years of Experience") + xlab("Experience Group") + geom\_bar(aes(y = 100\*(..count..)/sum(..count..)), width = 0.5, color = 'red') + ylab("Percentage") + coord\_flip() + theme\_minimal() + scale\_fill\_manual(values = c("red","blue","green","yellow"))  
  
agegroup <- ggplot(loan, aes(x=agegroup)) + ggtitle("Age Group") + xlab("Age Group") + geom\_bar(aes(y = 100\*(..count..)/sum(..count..)), width = 0.5, colour = 'red') + ylab("Percentage") + coord\_flip()  
  
cdAcct <- ggplot(loan, aes(x=CDAcct)) + ggtitle("Credit Debit Account") + xlab("Credit Debit Account") + geom\_bar(aes(y = 100\*(..count..)/sum(..count..)), width = 0.5, colour = 'red') + ylab("Percentage") + coord\_flip() + theme\_minimal()  
  
secAcct <- ggplot(loan, aes(x=securitiesAcct)) + ggtitle("Securities Account") + xlab("Securities Account") + geom\_bar(aes(y = 100\*(..count..)/sum(..count..)), width = 0.5, colour = 'red') + ylab("Percentage") + coord\_flip()+ theme\_minimal()  
  
online <- ggplot(loan, aes(x=Online)) + ggtitle("Online Banking") + xlab("Online") + geom\_bar(aes(y = 100\*(..count..)/sum(..count..)), width = 0.5, colour = 'red') + ylab("Percentage") + coord\_flip() + theme\_minimal()  
  
creditCard <- ggplot(loan, aes(x=CreditCard)) + ggtitle("Credit Card") + xlab("Credit Card") + geom\_bar(aes(y = 100\*(..count..)/sum(..count..)), width = 0.5, colour = 'red') + ylab("Percentage") + coord\_flip() + theme\_minimal()  
  
education <- ggplot(loan, aes(x=Education)) + ggtitle("Education Level") + xlab("Education Level") + geom\_bar(aes(y = 100\*(..count..)/sum(..count..)), width = 0.5, colour = 'red') + ylab("Percentage") + coord\_flip() + theme\_minimal()  
  
Income <- ggplot(loan, aes(x= income)) + ggtitle("Annual income") + xlab("Income") + geom\_bar(aes(y = 100\*(..count..)/sum(..count..)), width = 0.5, colour = 'red') + ylab("Percentage") + coord\_flip() + theme\_minimal()  
  
grid.arrange(Income, expr, agegroup, cdAcct, secAcct, online, creditCard, education, ncol = 3)

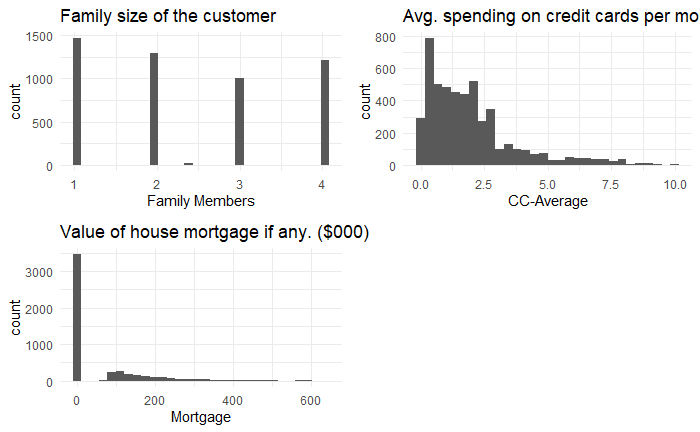


From the above, there are variables with multi-category levels: income, Experience group, Age group. One-hot encoding will be applied to the categorical variable.

Here is the distribution of the numeric variables

# Numeric variables distribution

familyMem <- ggplot(loan, aes(x= familyMember)) + ggtitle("Family size of the customer") + xlab("Family Members") + geom\_histogram() + theme\_minimal()  
  
ccAvg <- ggplot(loan, aes(x= CCAvg)) + ggtitle("Avg. spending on credit cards per month. ($000)") + xlab("CC-Average") + geom\_histogram() + theme\_minimal()  
  
mortgage <- ggplot(loan, aes(x= Mortgage)) + ggtitle("Value of house mortgage if any. ($000)") + xlab("Mortgage") + geom\_histogram() + theme\_minimal()  
  
grid.arrange(familyMem, ccAvg, mortgage, ncol = 2)



The family member is widely spread unlike CC-Average and Mortgage which are both skewed to the right. The skewed variables will be transformed using log.

**CLUSTERING ALGORITHM: PARTITIONING AROUND MEDOIDS (PAM)**

By clustering, we mean to find the similarity in our data. Since this data set is of mixed variables that is consist of numeric and categorical variable, hence the use of k-mean is not advisable. PAM clustering algorithm (partitioning around medoids) as well as silhouette coefficient to select optimal number of clusters will be used in our clustering analysis. Packages cluster and Rtsne are the R packages used for the analysis.

The Gower distance which is available in R using daisy()function from the cluster package fits well with the [k-medoids algorithm](https://en.wikipedia.org/wiki/K-medoids). k-medoid is a classical partitioning technique of clustering that clusters the data set of n objects into k clusters known a priori.

Interpretation: There are basically two ways to investigate the results of such a clustering exercise, in order to derive some business-relevant interpretation.

1. Summary of each cluster, using summary() function in R.
2. Visualization in a lower dimensional space, with [t-SNE](https://lvdmaaten.github.io/tsne/), using Rtsne() function in R. t-Distributed Stochastic Neighbor Embedding (t-SNE) is a technique for dimensionality reduction that is particularly well suited for the visualization of high-dimensional datasets.

Most similar and dissimilar clients according to Gower distance:

#' Compute Gower distance  
 gower\_dist <- daisy(loan, metric = "gower")  
  
 gower\_mat <- as.matrix(gower\_dist)  
  
# Print most similar clients  
 loan[which(gower\_mat == min(gower\_mat[gower\_mat != min(gower\_mat)]), arr.ind = TRUE) [1, ], ]

## # A tibble: 2 x 12  
## familyMember CCAvg Education Mortgage securitiesAcct CDAcct Online CreditCard  
## <dbl> <dbl> <fct> <dbl> <fct> <fct> <fct> <fct>   
## 1 4 1.7 Graduate 103 0 0 1 0   
## 2 4 1.7 Graduate 104 0 0 1 0   
## # ... with 4 more variables: agegroup <fct>, income <fct>, personalLoan <fct>,  
## # Exp\_Agegroup <fct>

# Print most dissimilar clients  
 loan[which(gower\_mat == max(gower\_mat[gower\_mat != max(gower\_mat)]), arr.ind = TRUE)[1, ], ]

## # A tibble: 2 x 12  
## familyMember CCAvg Education Mortgage securitiesAcct CDAcct Online CreditCard  
## <dbl> <dbl> <fct> <dbl> <fct> <fct> <fct> <fct>   
## 1 4 0.9 Undergrad 0 1 1 1 1   
## 2 1 7 Advanced~ 541 0 0 0 0   
## # ... with 4 more variables: agegroup <fct>, income <fct>, personalLoan <fct>,  
## # Exp\_Agegroup <fct>

In business situation, we usually search for a number of clusters both meaningful and easy to remember, i.e. 2 to 8 maximums. The silhouette figure helps us identify the best option(s).

sil\_width <- c(NA)  
 for (i in 2:8) {  
 pam\_fit <- pam(gower\_dist, diss = TRUE, k = i)  
 sil\_width[i] <- pam\_fit$silinfo$avg.width  
 }  
  
 plot(1:8, sil\_width,  
 xlab = "Number of clusters",  
 ylab = "Sihouette Width")  
 lines(1:8, sil\_width)



*6 clusters have the highest silhouette width therefore, let’s pick k = 6*

**Interpretation:**

Summary of each cluster

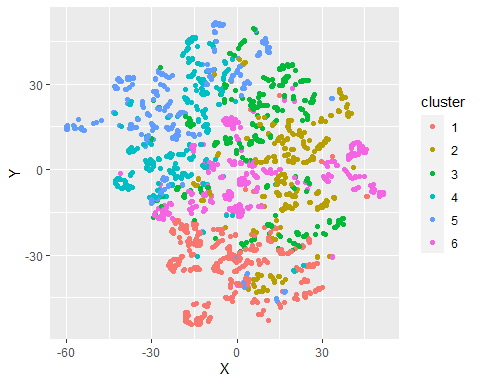
# 6 clusters has the highest silhouette width  
 k <- 6  
 pam\_fit <- pam(gower\_dist, diss = TRUE, k)  
 pam\_results <- loan %>%  
 mutate(cluster = pam\_fit$clustering) %>%  
 group\_by(cluster) %>%  
 do(the\_summary = summary(.))  
 pam\_results$the\_summary

## [[1]]  
## familyMember CCAvg Education Mortgage   
## Min. :1.000 Min. : 0.0 Undergrad :547 Min. : 0.00   
## 1st Qu.:1.000 1st Qu.: 0.8 Graduate :233 1st Qu.: 0.00   
## Median :2.000 Median : 1.8 Advanced/Professional:190 Median : 0.00   
## Mean :2.363 Mean : 2.2 Mean : 63.39   
## 3rd Qu.:4.000 3rd Qu.: 2.9 3rd Qu.:105.75   
## Max. :4.000 Max. :10.0 Max. :612.00   
## securitiesAcct CDAcct Online CreditCard agegroup   
## 0:871 0:900 0:332 0:679 Young :962   
## 1: 99 1: 70 1:638 1:291 Middle-Aged: 7   
## Old : 1   
##   
##

##   
## income personalLoan Exp\_Agegroup cluster   
## Lower Class : 38 No :842 0yrs\_Exp : 52 Min. :1   
## Working Class :114 Yes:128 Between 1-10yrs :909 1st Qu.:1   
## Lower Middle Class:275 Between 11-20yrs: 8 Median :1   
## Upper Middle Class:543 Between 21-30yrs: 0 Mean :1   
## Between 31-40yrs: 0 3rd Qu.:1   
## Between 41-50yrs: 1 Max. :1   
##   
## [[2]]  
## familyMember CCAvg Education Mortgage   
## Min. :1.000 Min. : 0.000 Undergrad :155 Min. : 0.00   
## 1st Qu.:1.000 1st Qu.: 0.500 Graduate :133 1st Qu.: 0.00   
## Median :2.000 Median : 1.000 Advanced/Professional:544 Median : 0.00   
## Mean :2.304 Mean : 1.313 Mean : 47.32   
## 3rd Qu.:3.000 3rd Qu.: 2.000 3rd Qu.:100.00   
## Max. :4.000 Max. :10.000 Max. :410.00   
## securitiesAcct CDAcct Online CreditCard agegroup   
## 0:750 0:818 0:675 0:602 Young :123   
## 1: 82 1: 14 1:157 1:230 Middle-Aged:681   
## Old : 28   
##   
##   
##   
## income personalLoan Exp\_Agegroup cluster   
## Lower Class : 57 No :803 0yrs\_Exp : 22 Min. :2   
## Working Class :158 Yes: 29 Between 1-10yrs :106 1st Qu.:2   
## Lower Middle Class:528 Between 11-20yrs:534 Median :2   
## Upper Middle Class: 89 Between 21-30yrs:168 Mean :2   
## Between 31-40yrs: 0 3rd Qu.:2   
## Between 41-50yrs: 2 Max. :2   
##   
## [[3]]  
## familyMember CCAvg Education Mortgage   
## Min. :1.000 Min. :0.000 Undergrad :132 Min. : 0.00   
## 1st Qu.:2.000 1st Qu.:0.670 Graduate :610 1st Qu.: 0.00   
## Median :3.000 Median :1.300 Advanced/Professional: 98 Median : 0.00   
## Mean :2.814 Mean :1.485 Mean : 45.61   
## 3rd Qu.:4.000 3rd Qu.:2.000 3rd Qu.: 95.00   
## Max. :4.000 Max. :9.000 Max. :590.00   
## securitiesAcct CDAcct Online CreditCard agegroup   
## 0:759 0:803 0:194 0:599 Young :127   
## 1: 81 1: 37 1:646 1:241 Middle-Aged:592   
## Old :121   
##   
##   
##   
## income personalLoan Exp\_Agegroup cluster   
## Lower Class : 42 No :788 0yrs\_Exp : 18 Min. :3   
## Working Class :129 Yes: 52 Between 1-10yrs :118 1st Qu.:3   
## Lower Middle Class:521 Between 11-20yrs:106 Median :3   
## Upper Middle Class:148 Between 21-30yrs:595 Mean :3   
## Between 31-40yrs: 0 3rd Qu.:3   
## Between 41-50yrs: 3 Max. :3   
##   
## [[4]]  
## familyMember CCAvg Education Mortgage   
## Min. :1.000 Min. :0.000 Undergrad :490 Min. : 0.00   
## 1st Qu.:1.000 1st Qu.:0.700 Graduate :134 1st Qu.: 0.00   
## Median :2.000 Median :1.600 Advanced/Professional: 91 Median : 0.00   
## Mean :2.137 Mean :2.017 Mean : 61.99   
## 3rd Qu.:3.000 3rd Qu.:2.800 3rd Qu.: 94.00   
## Max. :4.000 Max. :9.300 Max. :601.00   
## securitiesAcct CDAcct Online CreditCard agegroup   
## 0:638 0:682 0:500 0:518 Young : 12   
## 1: 77 1: 33 1:215 1:197 Middle-Aged: 39   
## Old :664   
##   
##   
##   
## income personalLoan Exp\_Agegroup cluster   
## Lower Class : 38 No :622 0yrs\_Exp : 10 Min. :4   
## Working Class : 99 Yes: 93 Between 1-10yrs : 1 1st Qu.:4   
## Lower Middle Class:126 Between 11-20yrs: 2 Median :4   
## Upper Middle Class:452 Between 21-30yrs:169 Mean :4   
## Between 31-40yrs:512 3rd Qu.:4   
## Between 41-50yrs: 21 Max. :4   
##   
## [[5]]  
## familyMember CCAvg Education Mortgage   
## Min. :1.000 Min. :0.000 Undergrad :149 Min. : 0.00   
## 1st Qu.:2.000 1st Qu.:0.700 Graduate :186 1st Qu.: 0.00   
## Median :3.000 Median :1.400 Advanced/Professional:470 Median : 0.00   
## Mean :2.708 Mean :1.512 Mean : 48.85   
## 3rd Qu.:4.000 3rd Qu.:2.000 3rd Qu.:100.00   
## Max. :4.000 Max. :8.200 Max. :587.00   
## securitiesAcct CDAcct Online CreditCard agegroup   
## 0:725 0:749 0:123 0:558 Young : 41   
## 1: 80 1: 56 1:682 1:247 Middle-Aged: 5   
## Old :759   
##   
##   
##   
## income personalLoan Exp\_Agegroup cluster   
## Lower Class : 36 No :759 0yrs\_Exp : 16 Min. :5   
## Working Class :103 Yes: 46 Between 1-10yrs : 30 1st Qu.:5   
## Lower Middle Class:540 Between 11-20yrs: 0 Median :5   
## Upper Middle Class:126 Between 21-30yrs:145 Mean :5   
## Between 31-40yrs:591 3rd Qu.:5   
## Between 41-50yrs: 23 Max. :5   
##   
## [[6]]  
## familyMember CCAvg Education Mortgage   
## Min. :1.000 Min. : 0.000 Undergrad :623 Min. : 0.00   
## 1st Qu.:1.000 1st Qu.: 1.100 Graduate :107 1st Qu.: 0.00   
## Median :2.000 Median : 2.685 Advanced/Professional:108 Median : 0.00   
## Mean :2.035 Mean : 3.051 Mean : 71.22   
## 3rd Qu.:3.000 3rd Qu.: 4.600 3rd Qu.:113.50   
## Max. :4.000 Max. :10.000 Max. :635.00   
## securitiesAcct CDAcct Online CreditCard agegroup   
## 0:735 0:746 0:192 0:574 Young : 9   
## 1:103 1: 92 1:646 1:264 Middle-Aged:806   
## Old : 23   
##   
##   
##   
## income personalLoan Exp\_Agegroup cluster   
## Lower Class : 14 No :706 0yrs\_Exp : 0 Min. :6   
## Working Class : 37 Yes:132 Between 1-10yrs : 7 1st Qu.:6   
## Lower Middle Class:103 Between 11-20yrs:603 Median :6   
## Upper Middle Class:684 Between 21-30yrs:224 Mean :6   
## Between 31-40yrs: 0 3rd Qu.:6   
## Between 41-50yrs: 4 Max. :6

Here one can attempt to derive some common patterns for clients within a cluster. As an example, cluster 1 is made of “Undergrad x tertiary x no securitiesAcct x no CDAcct ,” clients, cluster 2 is made of “Advanced/Professional x no CreditCard x no CDAcct” clients, etc.

tsne\_obj <- Rtsne(gower\_dist, is\_distance = TRUE)  
 tsne\_data <- tsne\_obj$Y %>%  
 data.frame() %>%  
 setNames(c("X", "Y")) %>%  
 mutate(cluster = factor(pam\_fit$clustering))  
  
 ggplot(aes(x = X, y = Y), data = tsne\_data) + geom\_point(aes(color = cluster))



Colors are mostly located in similar areas, confirming the relevancy of the segmentation.

To start the preprocessing, let’s split our data set into training and testing sets

#Splitting, Testing, and Training  
  
set.seed(0)  
split <- sample(seq\_len(nrow(loan)),  
 size = floor(0.70 \* nrow(loan)))  
train\_set <- loan[split, ]  
test\_set <- loan[-split, ]  
dim(train\_set); dim(test\_set)

## [1] 3500 12

## [1] 1500 12

We split the data into train of 70% and test of 30% and also for the sake of reproducibility we set the seed as 0.

**The Preprocessing**

The recipe() function implements the preprocessing steps while the bake() function processes the data by following the steps in the recipe() function.

loan\_recipe <- recipe(personalLoan ~ ., data = train\_set) %>% step\_log(Mortgage, signed = TRUE) %>% step\_log(CCAvg, signed = TRUE) %>% step\_dummy(all\_nominal(), -all\_outcomes()) %>% step\_center(all\_predictors(), -all\_outcomes()) %>% step\_scale(all\_predictors(), -all\_outcomes()) %>% prep(data = train\_set)

Step\_log() to log transform “Mortgage, CCAvg”, A logical indicating whether to take the signed log, If TRUE the offset argument will be ignored. step\_dummy() to convert categorical variables to dummy variables. Step\_center() to mean-center the data and step\_scale() to scale the data. The centering and scaling were done for the sake of improving numerical stability.

Baking the recipe object using the bake() from the recipe package.

train\_bake <- bake(loan\_recipe, new\_data = train\_set)  
test\_bake <- bake(loan\_recipe, new\_data = test\_set)  
glimpse(train\_bake)

## Rows: 3,500  
## Columns: 20  
## $ familyMember <dbl> 1.4030701, -1.2117020, -1.2117020, ...  
## $ CCAvg <dbl> 0.16027192, -0.93443653, -0.6234795...  
## $ Mortgage <dbl> 1.4889331, -0.6663839, -0.6663839, ...  
## $ personalLoan <fct> No, No, No, No, No, No, No, No, No,...  
## $ Education\_Graduate <dbl> -0.6186625, 1.6159285, -0.6186625, ...  
## $ Education\_Advanced.Professional <dbl> 1.5047519, -0.6643715, 1.5047519, -...  
## $ securitiesAcct\_X1 <dbl> -0.3369767, 2.9667164, -0.3369767, ...  
## $ CDAcct\_X1 <dbl> -0.2454945, -0.2454945, -0.2454945,...  
## $ Online\_X1 <dbl> -1.2223859, 0.8178385, 0.8178385, 0...  
## $ CreditCard\_X1 <dbl> -0.640784, -0.640784, 1.560142, -0....  
## $ agegroup\_Middle.Aged <dbl> 1.1585856, -0.8628748, -0.8628748, ...  
## $ agegroup\_Old <dbl> -0.6876982, -0.6876982, 1.4537108, ...  
## $ income\_Working.Class <dbl> -0.3835659, -0.3835659, 2.6063689, ...  
## $ income\_Lower.Middle.Class <dbl> 1.1646951, 1.1646951, -0.8583485, 1...  
## $ income\_Upper.Middle.Class <dbl> -0.820273, -0.820273, -0.820273, -0...  
## $ Exp\_Agegroup\_Between.1.10yrs <dbl> -0.5508643, 1.8148105, -0.5508643, ...  
## $ Exp\_Agegroup\_Between.11.20yrs <dbl> 1.7384260, -0.5750687, -0.5750687, ...  
## $ Exp\_Agegroup\_Between.21.30yrs <dbl> -0.597946, -0.597946, -0.597946, -0...  
## $ Exp\_Agegroup\_Between.31.40yrs <dbl> -0.534986, -0.534986, 1.868674, -0....  
## $ Exp\_Agegroup\_Between.41.50yrs <dbl> -0.1004894, -0.1004894, -0.1004894,...

Let begin our modeling.

**Cross Validation:**

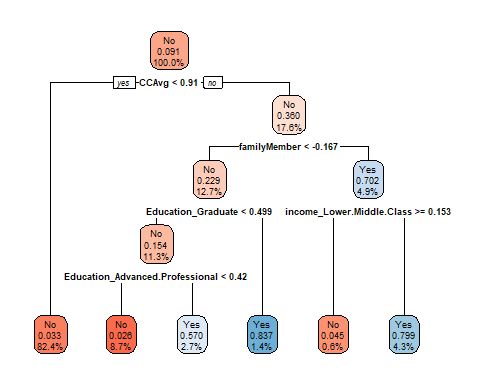
This is simply for resampling that is it involves fitting the same statistical method multiple times using different subsets of the data. Repeated k-fold Cross Validation will be used. number is the number of resampling iterations, repeats is the number of complete sets of folds to compute 5 – 10 is advisable, we chose 10

# Cross validation  
cv.ctrl <- trainControl(method = "repeatedcv", repeats = 10, number = 3)

**Fitting CART (Decision Tree without Pruning)**

#Applying CART <plot the tree> on the training set   
# Decision Tree with control point using Rpart Package and the plot   
rpart.plot(rpart(formula = personalLoan~., data = train\_bake, method = "class", control = rpart.control(maxdepth = 4)), box.palette="RdBu", digits = -3)

The rpart package is used for the fitting and rpart.plot for the tree plotting. The code above is for the tree plotting. Formula is the response variable in our case personalLoan, data is the train data, method to indicate our data type which is categorical hence, class is used, control is for controlling the depth of the tree and digit is for approximation of figures in the node.



The tree is of 4 max depth. The root node started with ‘No’ response to the dependent variable personalLoan and at 100%. The splitting started with Credit card average, when less than 0.91 is ‘No’, the decision node indicated ‘No’ that is the customers percentage that reject the loan is 17.6%, the next splitting is family member, it is less than -0.167, Yes means 12.7% of customer is No to personal loan and 4.9 % of customer indicated Yes when it is No etc.

# The above is the training code for the decision tree without pruning

treeWithCP <- train(form = personalLoan~., data = train\_bake, method="rpart", control = rpart.control(maxdepth = 4), trControl=cv.ctrl)  
  
#Predict value at any point and The confusion Matrix  
treeWithCp\_pred <- predict(treeWithCP, test\_bake, type = "raw")  
confusionMatrix(treeWithCp\_pred, test\_bake$personalLoan)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 1311 79  
## Yes 26 84  
##   
## Accuracy : 0.93   
## 95% CI : (0.9159, 0.9424)  
## No Information Rate : 0.8913   
## P-Value [Acc > NIR] : 2.183e-07   
##   
## Kappa : 0.5785   
##   
## Mcnemar's Test P-Value : 3.881e-07   
##   
## Sensitivity : 0.9806   
## Specificity : 0.5153   
## Pos Pred Value : 0.9432   
## Neg Pred Value : 0.7636   
## Prevalence : 0.8913   
## Detection Rate : 0.8740   
## Detection Prevalence : 0.9267   
## Balanced Accuracy : 0.7479   
##   
## 'Positive' Class : No   
##

For the decision tree without pruning (treeWithCP) model, we have gotten an accuracy of 93%. The confusion matrix has a type false negative of 79 which is known as type II error and false positive of 26 also known as Type I error.

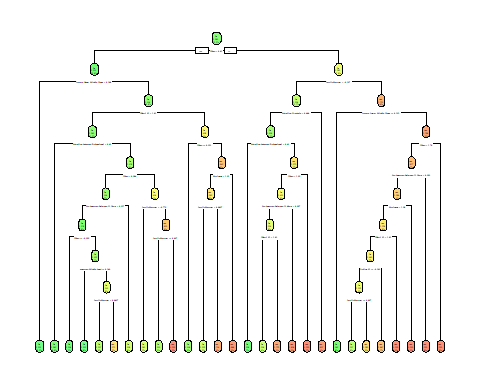
Let’s examine the performance of the decision tree with pruning.

**Fitting CART (Decision Tree) with Pruning**

This determines a nested sequence of subtrees of the supplied object by recursively snipping off the least important splits, based on the complexity parameter (cp). Let do the fitting below;

#Full tree without pruning and the plot  
fullTree <- rpart(formula = personalLoan~., data = train\_bake, method = "class", control = rpart.control(cp = 0))  
rpart.plot(fullTree, box.palette="GnYlRd", digits = -3)

The below tree is the full tree without prune or complexity parameter which is essentially difficult to interpret.



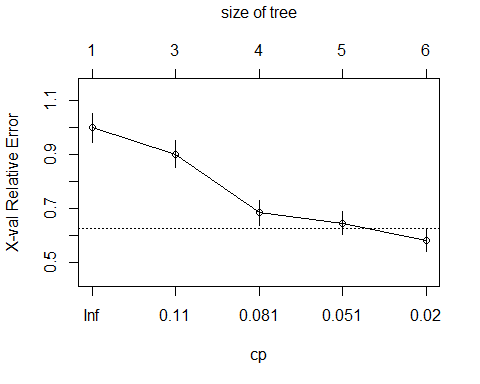
# To carry out pruning, let find the value of cp at which Cross Validation error is at minimum

treeWithCP <- rpart(formula = personalLoan~., data = train\_bake, method = "class", control = rpart.control(maxdepth = 4))  
  
printcp(fullTree)

##   
## Classification tree:  
## rpart(formula = personalLoan ~ ., data = train\_bake, method = "class",   
## control = rpart.control(cp = 0))  
##   
## Variables actually used in tree construction:  
## [1] agegroup\_Middle.Aged CCAvg   
## [3] CDAcct\_X1 Education\_Advanced.Professional  
## [5] Education\_Graduate Exp\_Agegroup\_Between.11.20yrs   
## [7] Exp\_Agegroup\_Between.21.30yrs Exp\_Agegroup\_Between.31.40yrs   
## [9] familyMember income\_Lower.Middle.Class   
## [11] income\_Upper.Middle.Class Mortgage   
## [13] Online\_X1   
##   
## Root node error: 317/3500 = 0.090571  
##   
## n= 3500   
##   
## CP nsplit rel error xerror xstd  
## 1 0.10883281 0 1.00000 1.00000 0.053562  
## 2 0.10410095 2 0.78233 0.88328 0.050631  
## 3 0.06309148 3 0.67823 0.68139 0.044909  
## 4 0.04100946 4 0.61514 0.64038 0.043623  
## 5 0.02839117 6 0.53312 0.55205 0.040674  
## 6 0.01261830 7 0.50473 0.50473 0.038980  
## 7 0.00946372 8 0.49211 0.52366 0.039668  
## 8 0.00788644 11 0.46372 0.53312 0.040007  
## 9 0.00630915 15 0.43218 0.52997 0.039895  
## 10 0.00078864 23 0.38170 0.52681 0.039782  
## 11 0.00000000 27 0.37855 0.54890 0.040564

The above are cp values at various levels with error and cross validation error(xerror)

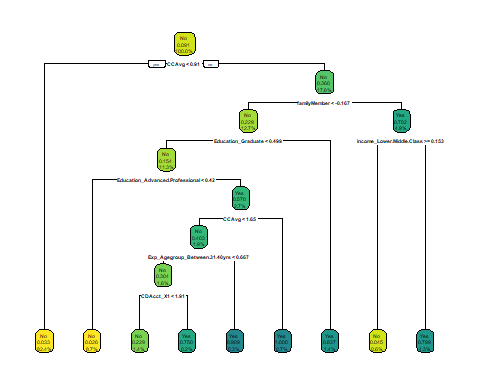
plotcp(treeWithCP)



From the above graph the minimum complexity parameter is below 0.02. Let calculate the minimum below.

# Calculate the Minimum cp

mincp <- treeWithCP$cptable[which.min(treeWithCP$cptable[, "xerror"]), "CP"]  
  
# Prune the tree  
prunedTree <- prune(fullTree, cp = mincp)  
rpart.plot(prunedTree, box.palette = "YlGnBl", digits = -3)



The tree is of 7 max depth after pruning. The root node started with ‘No’ response to the dependent variable personalLoan and at 100%. The splitting started with Credit card average, when less than 0.91 is ‘No’, the decision node indicated ‘No’ that is the customers percentage that reject the loan is 17.6%, the next splitting is family member, it is less than -0.167, Yes means 12.7% of customer is No to personal loan and 4.9 % of customer indicated Yes when it is No etc.

prunedTree <- train(form = personalLoan~., data = train\_bake, cp = mincp, trControl=cv.ctrl)  
  
#Predict value at any point of pruned tree  
prunedTree\_pred <- predict(prunedTree, test\_bake, type = "raw")  
confusionMatrix(prunedTree\_pred, test\_bake$personalLoan)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 1334 45  
## Yes 3 118  
##   
## Accuracy : 0.968   
## 95% CI : (0.9578, 0.9763)  
## No Information Rate : 0.8913   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.8137   
##   
## Mcnemar's Test P-Value : 3.262e-09   
##   
## Sensitivity : 0.9978   
## Specificity : 0.7239   
## Pos Pred Value : 0.9674   
## Neg Pred Value : 0.9752   
## Prevalence : 0.8913   
## Detection Rate : 0.8893   
## Detection Prevalence : 0.9193   
## Balanced Accuracy : 0.8608   
##   
## 'Positive' Class : No   
##

For the decision tree with pruning (pruneTree) model, we have gotten an accuracy of 96.8% which is better compare to decision tree without pruning. The confusion matrix has a false negative of 45 which is known as type II error and false positive of 3 also known as Type I error.

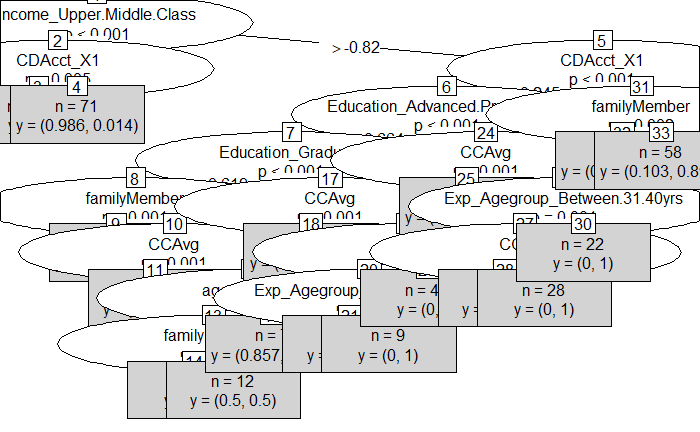
Let’s examine the performance of the Ctree function from party package on decision tree and compare with rpart package with pruning decision tree.

**Fitting CART (Decision Tree) using ctree**

Conditional inference trees(ctree) estimate a regression relationship by binary recursive partitioning in a conditional inference framework. Roughly, the algorithm works as follows:

1. Test the global null hypothesis of independence between any of the input variables and the response (which may be multivariate as well). Stop if this hypothesis cannot be rejected. Otherwise select the input variable with strongest association to the response. This association is measured by a p-value corresponding to a test for the partial null hypothesis of a single input variable and the response.
2. Implement a binary split in the selected input variable.
3. Recursively repeat steps 1) and 2).

fit.ctree <- ctree(personalLoan~., data = train\_bake)  
plot(fit.ctree, type = 'simple')



fit.ctree <- train(form = personalLoan~., data = train\_bake, method = "ctree", trControl=cv.ctrl)  
Ctree\_pred = predict(fit.ctree, newdata=test\_bake)  
confusionMatrix(Ctree\_pred, test\_bake$personalLoan)

## Confusion Matrix and Statistics  
## Reference  
## Prediction No Yes  
## No 1324 58  
## Yes 13 105  
##   
## Accuracy : 0.9527   
## 95% CI : (0.9407, 0.9629)  
## No Information Rate : 0.8913   
## P-Value [Acc > NIR] : < 2.2e-16   
## Kappa : 0.722   
##   
## Mcnemar's Test P-Value : 1.772e-07   
##   
## Sensitivity : 0.9903   
## Specificity : 0.6442   
## Pos Pred Value : 0.9580   
## Neg Pred Value : 0.8898   
## Prevalence : 0.8913   
## Detection Rate : 0.8827   
## Detection Prevalence : 0.9213   
## Balanced Accuracy : 0.8172   
##   
## 'Positive' Class : No   
##

For the decision tree with ctree model, we have gotten an accuracy of 95.27% which is less good compare to decision tree with pruning. The confusion matrix has a false negative of 58 which is known as type II error and false positive of 13 also known as Type I error. Let’s examine the performance of the Random Forest

**Fitting Random Forest**

mtry <- sqrt(ncol(train\_bake)) # Number of variables randomly sampled as candidates at each split  
tunegrid <- expand.grid(.mtry=mtry)  
rf<- train(form = personalLoan~., data=train\_bake, method="rf", metric="Accuracy", tuneGrid=tunegrid, trControl=cv.ctrl)  
rf\_pred<-predict(rf, test\_bake, type="raw")  
confusionMatrix(rf\_pred, test\_bake$personalLoan)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction No Yes  
## No 1332 56  
## Yes 5 107  
##   
## Accuracy : 0.9593   
## 95% CI : (0.9481, 0.9688)  
## No Information Rate : 0.8913   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.7566   
##   
## Mcnemar's Test P-Value : 1.535e-10   
##   
## Sensitivity : 0.9963   
## Specificity : 0.6564   
## Pos Pred Value : 0.9597   
## Neg Pred Value : 0.9554   
## Prevalence : 0.8913   
## Detection Rate : 0.8880   
## Detection Prevalence : 0.9253   
## Balanced Accuracy : 0.8264   
##   
## 'Positive' Class : No   
##

The mtry argument specifies number of variables randomly sampled as candidates at each split. In our own case, we want it to be the square root of the number of variables in our training data set. The method=“rf” specifies that the random forest model should be fitted.

For the random forest model, we have gotten an accuracy of 95.93% which is less good compare to decision tree with pruning. The confusion matrix has a false negative of 56 which is known as type II error and false positive of 5 also known as Type I error.

**ROC Curve for the Fitted Models**

# ROC Curve for Deciosion Tree without pruning using rpart package  
response1 <- predictor1 <- c()  
response1 <- c(response1, test\_bake$personalLoan)  
predictor1<- c(predictor1, treeWithCp\_pred)  
roc1 <- plot.roc(response1, predictor1, main="ROC Curve for the Fitted Models",ylab="True Positive Rate",xlab="False Positive Rate", percent=F, col="red", print.auc=TRUE)

## Setting levels: control = 1, case = 2

## Setting direction: controls < cases

The AUC, GINI and KS for tree without pruning

auc(roc1)

## Area under the curve: 0.7479

2\*auc(roc1)-1

## [1] 0.4958909

ks.test(response1, predictor1)

## Warning in ks.test(response1, predictor1): p-value will be approximate in the  
## presence of ties

##   
## Two-sample Kolmogorov-Smirnov test  
##   
## data: response1 and predictor1  
## D = 0.035333, p-value = 0.3063  
## alternative hypothesis: two-sided

# ROC Curve for Decision Tree with Pruning using rpart package  
response2 <- predictor2 <- c()  
response2 <- c(response2, test\_bake$personalLoan)  
predictor2 <- c(predictor2, prunedTree\_pred)  
par(new=T)  
roc2 <- plot.roc(response2, predictor2, ylab="True Positive Rate",xlab="False Positive Rate", percent=F, col="blue", print.auc=TRUE)

## Setting levels: control = 1, case = 2  
## Setting direction: controls < cases

The AUC, GINI and KS for tree with prunnig

auc(roc2)

## Area under the curve: 0.8608

2\*auc(roc2)-1

## [1] 0.7216826

ks.test(response2, predictor2)

## Warning in ks.test(response2, predictor2): p-value will be approximate in the  
## presence of ties

##   
## Two-sample Kolmogorov-Smirnov test  
##   
## data: response2 and predictor2  
## D = 0.028, p-value = 0.599  
## alternative hypothesis: two-sided

# ROC Curve for Decision Tree using Ctree in Party package  
response3 <- predictor3 <- c()  
response3 <- c(response3, test\_bake$personalLoan)  
predictor3<- c(predictor3, Ctree\_pred)  
par(new=T)  
roc3 <- plot.roc(response3, predictor3, ylab="True Positive Rate",xlab="False Positive Rate", percent=F, col="peachpuff")

## Setting levels: control = 1, case = 2  
## Setting direction: controls < cases

"The AUC, GINI and KS for ctree"

The AUC, GINI and KS for ctree

auc(roc3)

## Area under the curve: 0.8172

2\*auc(roc3)-1

## [1] 0.6344485

ks.test(response3, predictor3)

## Warning in ks.test(response3, predictor3): p-value will be approximate in the  
## presence of ties

##   
## Two-sample Kolmogorov-Smirnov test  
##   
## data: response3 and predictor3  
## D = 0.03, p-value = 0.5095  
## alternative hypothesis: two-sided

# ROC Curve for Random Forest  
response4<- predictor4 <- c()  
response4 <- c(response4, test\_bake$personalLoan)  
predictor4 <- c(predictor4, Ctree\_pred)  
par(new=T)  
roc4 <- plot.roc(response4, predictor4, ylab="True Positive Rate",xlab="False Positive Rate", percent=F, col="darkseagreen4")

## Setting levels: control = 1, case = 2  
## Setting direction: controls < cases

"The AUC, GINI and KS for Random forest"

The AUC, GINI and KS for Random forest

auc(roc4)

## Area under the curve: 0.8172

2\*auc(roc4)-1

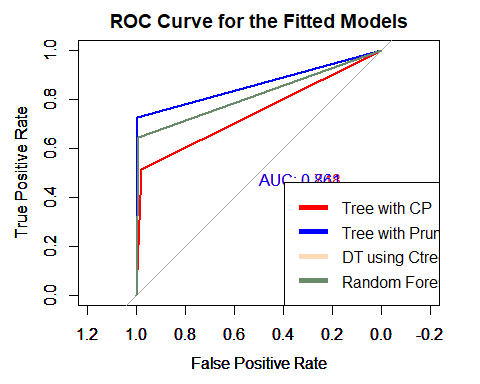
## [1] 0.6344485

ks.test(response4, predictor4)

## Warning in ks.test(response4, predictor4): p-value will be approximate in the  
## presence of ties

##   
## Two-sample Kolmogorov-Smirnov test  
##   
## data: response4 and predictor4  
## D = 0.03, p-value = 0.5095  
## alternative hypothesis: two-sided

legend("bottomright", legend = c("Tree with CP", "Tree with Pruning", "DT using Ctree", "Random Forest"), col = c("red", "blue","peachpuff", "darkseagreen4"),lwd = 5)



The Decision Tree with Pruning has the highest accuracy as seen from the output above, also  
from the ROC curve, the Decision Tree with Pruning model has the largest area under the curve 86.08% and the GINI coefficient is 0.7216826. Going forward, the Decision Tree with Pruning algorithm is recommended. Let’s examine the Decision Tree with Pruning model and the most influential features locally using LIME package.

**Model Explanation using the Lime Package**

explainer <- lime::lime(x = train\_bake,  
 model = prunedTree,  
 quantile\_bins = FALSE  
 )

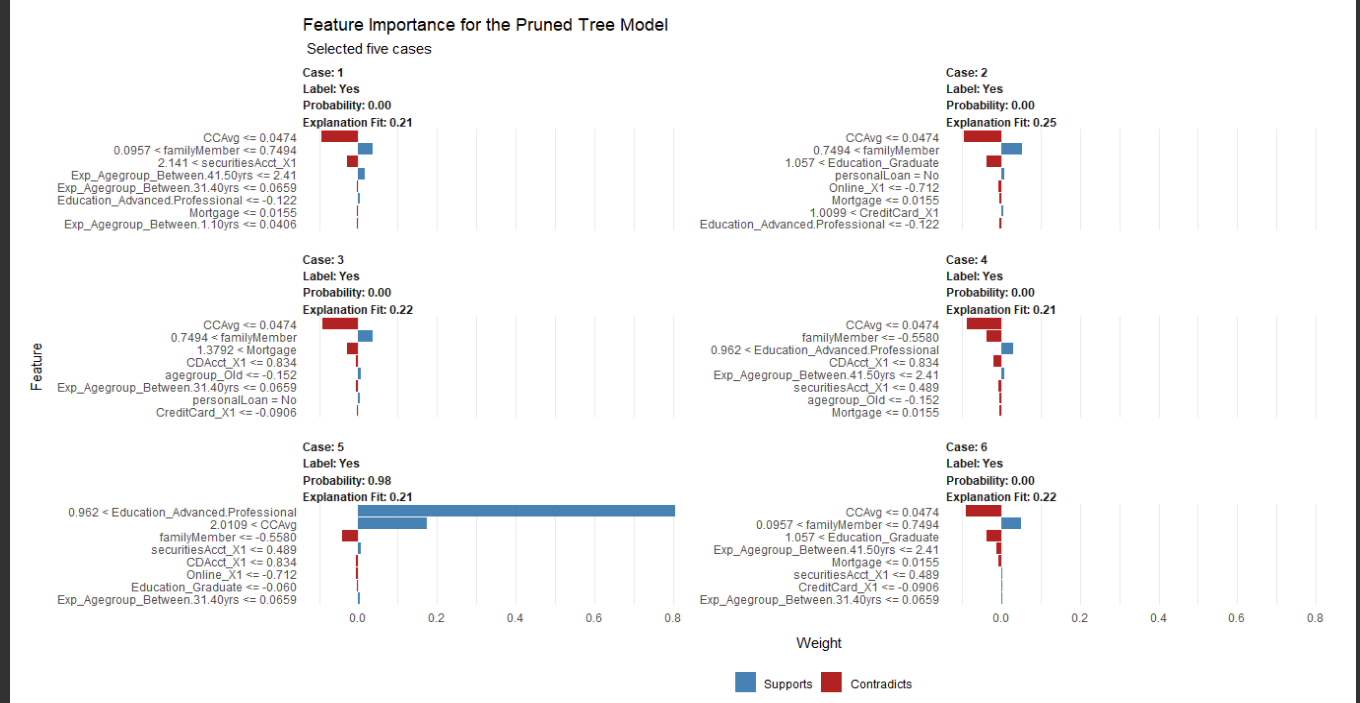
we create an explainer using the lime() function, which only takes the model we intend to explain which is the Decision Tree with Pruning(prunedTree) model and the train data set. We set the quantile\_bins= FALSE.

Let’s examine factors that were important to being promoted by selecting five cases in our test data set.

explanation <- lime::explain(test\_bake[1:6, ],  
 explainer = explainer,  
 n\_features = 8,  
 feature\_select = "highest\_weights",  
 labels = "Yes"  
)

The explain() function helps in explaining the explainer we set above. We set feature\_select = “highest\_weights” because we are interested in features with the highest absolute weight. We set n\_features = 8 because we want to see the eight most important features in the Decision Tree with Pruning(prunedTree) model. Finally, we set the labels = “Yes” because we are interested in cases where the personal loan is taken by customer.

plot\_features(explanation) +  
labs(title = "Feature Importance for the Pruned Tree Model",  
subtitle = " Selected five cases")



For most of the cases, negative impact on the personal loan is obvious except case 5 where Education\_AdvancedProfessional support with significant value likewise CCAvg (Average Credit Card). The LIME only provides local interpretation which means that we are only interpreting the Decision Tree with Pruning(prunedTree) model on a case by case basis. Let’s examine the global interpretation of the Decision Tree with Pruning(prunedTree) model, understanding the features that are important on a global perspective using the Corrr package.

train\_bake$personalLoan<-as.numeric(train\_bake$personalLoan)  
global\_perspective <- train\_bake %>%   
 correlate() %>%   
 focus(personalLoan) %>%  
 rename(Variable = rowname) %>%  
 arrange(abs(personalLoan)) %>%  
 mutate(feature = as.factor(Variable))

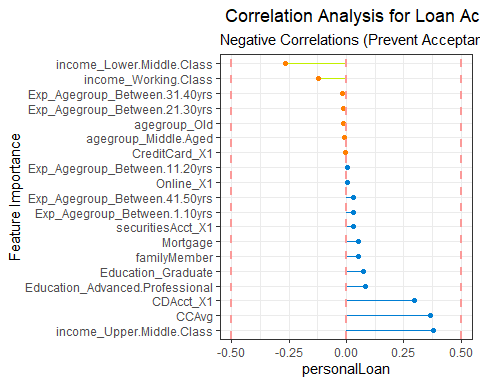
##   
## Correlation method: 'pearson'  
## Missing treated using: 'pairwise.complete.obs'

global\_perspective

## # A tibble: 19 x 3  
## Variable personalLoan feature   
## <chr> <dbl> <fct>   
## 1 Exp\_Agegroup\_Between.11.20yrs 0.00277 Exp\_Agegroup\_Between.11.20yrs   
## 2 Online\_X1 0.00421 Online\_X1   
## 3 CreditCard\_X1 -0.00502 CreditCard\_X1   
## 4 agegroup\_Middle.Aged -0.0107 agegroup\_Middle.Aged   
## 5 agegroup\_Old -0.0124 agegroup\_Old   
## 6 Exp\_Agegroup\_Between.21.30yrs -0.0124 Exp\_Agegroup\_Between.21.30yrs   
## 7 Exp\_Agegroup\_Between.31.40yrs -0.0181 Exp\_Agegroup\_Between.31.40yrs   
## 8 Exp\_Agegroup\_Between.41.50yrs 0.0283 Exp\_Agegroup\_Between.41.50yrs   
## 9 Exp\_Agegroup\_Between.1.10yrs 0.0287 Exp\_Agegroup\_Between.1.10yrs   
## 10 securitiesAcct\_X1 0.0318 securitiesAcct\_X1   
## 11 Mortgage 0.0505 Mortgage   
## 12 familyMember 0.0530 familyMember   
## 13 Education\_Graduate 0.0717 Education\_Graduate   
## 14 Education\_Advanced.Professional 0.0819 Education\_Advanced.Professional  
## 15 income\_Working.Class -0.121 income\_Working.Class   
## 16 income\_Lower.Middle.Class -0.267 income\_Lower.Middle.Class   
## 17 CDAcct\_X1 0.297 CDAcct\_X1   
## 18 CCAvg 0.365 CCAvg   
## 19 income\_Upper.Middle.Class 0.381 income\_Upper.Middle.Class

Let’s visualize this correlation to enable us identify variables that are relevant to Staff Promotion.

global\_perspective %>% ggplot(aes(x = personalLoan, y = fct\_reorder(Variable, desc(personalLoan)))) + geom\_point() + geom\_segment(aes(xend = 0, yend = Variable), color = palette\_dark()[[6]], data = global\_perspective %>% filter(personalLoan > 0)) + geom\_point(color = palette\_dark()[[6]], data = global\_perspective %>% filter(personalLoan > 0)) + geom\_segment(aes(xend = 0, yend = Variable), color = palette\_dark()[[10]], data = global\_perspective %>% filter(personalLoan < 0)) + geom\_point(color = palette\_light()[[10]], data = global\_perspective %>% filter(personalLoan < 0)) + geom\_vline(xintercept = 0, color = palette\_light()[[8]], size = 1, linetype = 2) + geom\_vline(xintercept = -0.5, color = palette\_light()[[8]], size = 1, linetype = 2) + geom\_vline(xintercept = 0.5, color = palette\_light()[[8]], size = 1, linetype = 2) +  
theme\_bw() + labs(title = " Correlation Analysis for Loan Acceptance",subtitle = paste("Negative Correlations (Prevent Acceptance),","Positive Correlations (Support Acceptance)"),y = "Feature Importance")



The features with the blue lines revealed the right customers who have a higher probability of purchasing the loan while the variables with yellow lines revealed otherwise. From this correlation plot, we can see the features that contribute positively to accepting personal loan and those that prevent it.

**Suggestion**

The suggestion to Thera Bank is to channel the retail marketing department to devise campaign toward customers with the following features; Income (Upper Middle Class), that have Certificate of Deposit Account, and worth mention Education (Advanced Professional), Average spending on credit cards. By doing this will lead to minimal budget and increasing the asset base of the bank.

**Conclusion**

In conclusion, we applied machine learning techniques to examine factors which can classify the right customers who have a higher probability of purchasing the loan based on the given data set. We started by splitting the dataset into 70% training and 30% test datasets. We implemented four machine learning algorithms namely: Decision Tree without pruning, Decision Tree with pruning, Decision Tree using Ctree, Random Forest. The models were implemented using rpart, randomForest and party Package in R. The performance of the trained models was evaluated on the test data set and evaluation metrics such as Accuracy and ROC curve were used. The results of the performance metrics showed that Decision Tree with pruning perform better than other machine learning models. The LIME function was used to explain the important features of the Decision Tree with pruning locally while we used correlation analysis to gain a globalized understanding of important features of the Decision Tree with pruning model.