Bank_Personal_Loan

Sourav Dutta

13/06/2020

Executive summary

One of the leading bank wants to increase its asset customers (borrowers) base by offering Personal Loan. Majority of the Bank customers are from liability (depositors) customer base. Bank wants to cross sell its Personal Loan product to this set of customers to diversify its business to Asset.

A campaign that the bank ran last year for liability customers showed a healthy conversion rate of over 9% success. This has encouraged the retail marketing department to devise campaigns to better target marketing to increase the success ratio with a minimal budget.

The department wants to build a model that will help them identify the potential customers who have a higher probability of purchasing the loan. Then, they want to run multi-touch journey campaigns to increase the conversion rate and hence ROI.

Data

The file contains data of 5000 customers. The data include customer demographic information (age, income, etc.), the customer's relationship with the bank (mortgage, securities account, etc.), and the customer response to the last personal loan campaign (Personal Loan). Among these 5000 customers, only 480 (= 9.6%) accepted the personal loan that was offered to them in the earlier campaign

Objectives

To predict the propensity to buy a personal loan

Data Attribute Information

1) ID : Customer ID
2) Age : Customer's age in completed years
3) Experience : #years of professional experience
4) Income : Annual income of the customer ($000)
5) ZIP Code : Home Address ZIP code.
6) Family : Family size of the customer
7) CCAvg : Avg. spending on credit cards per month ($000)
8) Education : Education Level. 1: Undergrad; 2: Graduate; 3: Advanced/Professional
9) Mortgage : Value of house mortgage if any. ($000)
10) Personal Loan : Did this customer accept the personal loan offered in the last campaign?
11) Securities Account : Does the customer have a securities account with the bank?

12) CD Account : Does the customer have a certificate of deposit (CD) account with the bank?
13) Online : Does the customer use internet banking facilities?
14) Credit card : Does the customer use a credit card issued by Bank

Approches

1) Loading required Packages
2) Loading data
3) Data exploration & Preparation

a) Check observations and variables
b) Check variables class
c) Check missing data
d) Remove data where customer age is less than 18 or More than 65
e) Removing wrong data entry (where experience is mentioned as negative)
f) Converting necessary variables from integer to Factor
g) Plotting the variables to understand the predictors
h) Creating correlation matrix
i) Creating dummy variables

4) Test and training data set creation
5) Different modeling approaches:

a) Logistic Regression Model
b) K-nearest neighbors (kNN)
c) RandomForest model
d) Rborist model
e) rpart model

6) Results
7) Conclusion

Loading necessary Packages

```
if(!require(tidyverse)) install.packages("tidyverse", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: tidyverse

## -- Attaching packages ------------------------------------------ tidyverse 1.3.0 --

## v ggplot2 3.3.1     v purrr   0.3.4
## v tibble  3.0.1     v dplyr   1.0.0
## v tidyr   1.1.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## -- Conflicts --------------------------------------------- tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```r
if(!require(caret)) install.packages("caret", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: caret

## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```r
if(!require(data.table)) install.packages("data.table", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: data.table

##
## Attaching package: 'data.table'

## The following objects are masked from 'package:dplyr':
##
##     between, first, last

## The following object is masked from 'package:purrr':
##
##     transpose
```

```r
if(!require(hexbin)) install.packages("hexbin", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: hexbin
```

```r
if(!require(corrplot)) install.packages("corrplot", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: corrplot

## corrplot 0.84 loaded
```

```r
if(!require(RColorBrewer)) install.packages("RColorBrewer", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: RColorBrewer
```

```r
if(!require(dummies)) install.packages("dummies", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: dummies

## dummies-1.5.6 provided by Decision Patterns
```

```r
if(!require(e1071)) install.packages("e1071", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: e1071
```

```r
if(!require(Rborist)) install.packages("Rborist", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: Rborist
```

```
## Rborist 0.2-3
```

```
## Type RboristNews() to see new features/changes/bug fixes.
```

```r
if(!require(randomForest)) install.packages("randomForest", repos = "http://cran.us.r-project.org")
```

```
## Loading required package: randomForest
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
##
## Attaching package: 'randomForest'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

```
## The following object is masked from 'package:ggplot2':
##
##     margin
```

Loading data

```r
download.file("https://raw.githubusercontent.com/souravdutta20/CYO_Project/master/Bank_Personal_Loan_Mo

Bank_data<- read.csv("Bank_Personal_Loan_Modelling.csv")
head(Bank_data)
```

```
##   ID Age Experience Income ZIP.Code Family CCAvg Education Mortgage
## 1  1  25          1     49    91107      4   1.6         1        0
## 2  2  45         19     34    90089      3   1.5         1        0
## 3  3  39         15     11    94720      1   1.0         1        0
## 4  4  35          9    100    94112      1   2.7         2        0
## 5  5  35          8     45    91330      4   1.0         2        0
## 6  6  37         13     29    92121      4   0.4         2      155
##   Personal.Loan Securities.Account CD.Account Online CreditCard
## 1             0                  1          0      0          0
## 2             0                  1          0      0          0
## 3             0                  0          0      0          0
## 4             0                  0          0      0          0
## 5             0                  0          0      0          1
## 6             0                  0          0      1          0
```

Data Exploration and Preparation

Checking observations and variables

```r
# Check observation & Variables
dim(Bank_data)
```

```
## [1] 5000   14
```

```r
# To see top 6 records
head(Bank_data)
```

```
##   ID Age Experience Income ZIP.Code Family CCAvg Education Mortgage
## 1  1  25          1     49    91107      4   1.6         1        0
## 2  2  45         19     34    90089      3   1.5         1        0
## 3  3  39         15     11    94720      1   1.0         1        0
## 4  4  35          9    100    94112      1   2.7         2        0
## 5  5  35          8     45    91330      4   1.0         2        0
## 6  6  37         13     29    92121      4   0.4         2      155
##   Personal.Loan Securities.Account CD.Account Online CreditCard
## 1             0                  1          0      0          0
## 2             0                  1          0      0          0
## 3             0                  0          0      0          0
## 4             0                  0          0      0          0
## 5             0                  0          0      0          1
## 6             0                  0          0      1          0
```

```r
# See the end 6 record
tail(Bank_data)
```

```
##        ID Age Experience Income ZIP.Code Family CCAvg Education Mortgage
## 4995 4995  64         40     75    94588      3   2.0         3        0
## 4996 4996  29          3     40    92697      1   1.9         3        0
## 4997 4997  30          4     15    92037      4   0.4         1       85
## 4998 4998  63         39     24    93023      2   0.3         3        0
## 4999 4999  65         40     49    90034      3   0.5         2        0
## 5000 5000  28          4     83    92612      3   0.8         1        0
##      Personal.Loan Securities.Account CD.Account Online CreditCard
## 4995             0                  0          0      1          0
## 4996             0                  0          0      1          0
## 4997             0                  0          0      1          0
## 4998             0                  0          0      0          0
## 4999             0                  0          0      1          0
## 5000             0                  0          0      1          1
```

```r
# How many unique customers present in database
n_distinct(Bank_data$ID)
```

```
## [1] 5000
```

```r
# get Descriptive Statistics
summary(Bank_data)
```

```
##        ID            Age          Experience       Income          ZIP.Code    
##  Min.   :   1   Min.   :23.00   Min.   :-3.0   Min.   :  8.00   Min.   : 9307  
##  1st Qu.:1251   1st Qu.:35.00   1st Qu.:10.0   1st Qu.: 39.00   1st Qu.:91911  
##  Median :2500   Median :45.00   Median :20.0   Median : 64.00   Median :93437  
##  Mean   :2500   Mean   :45.34   Mean   :20.1   Mean   : 73.77   Mean   :93152  
##  3rd Qu.:3750   3rd Qu.:55.00   3rd Qu.:30.0   3rd Qu.: 98.00   3rd Qu.:94608  
##  Max.   :5000   Max.   :67.00   Max.   :43.0   Max.   :224.00   Max.   :96651  
##      Family          CCAvg          Education        Mortgage    
##  Min.   :1.000   Min.   : 0.000   Min.   :1.000   Min.   :  0.0  
##  1st Qu.:1.000   1st Qu.: 0.700   1st Qu.:1.000   1st Qu.:  0.0  
##  Median :2.000   Median : 1.500   Median :2.000   Median :  0.0  
##  Mean   :2.396   Mean   : 1.938   Mean   :1.881   Mean   : 56.5  
##  3rd Qu.:3.000   3rd Qu.: 2.500   3rd Qu.:3.000   3rd Qu.:101.0  
##  Max.   :4.000   Max.   :10.000   Max.   :3.000   Max.   :635.0  
##  Personal.Loan   Securities.Account   CD.Account          Online      
##  Min.   :0.000   Min.   :0.0000     Min.   :0.0000   Min.   :0.0000  
##  1st Qu.:0.000   1st Qu.:0.0000     1st Qu.:0.0000   1st Qu.:0.0000  
##  Median :0.000   Median :0.0000     Median :0.0000   Median :1.0000  
##  Mean   :0.096   Mean   :0.1044     Mean   :0.0604   Mean   :0.5968  
##  3rd Qu.:0.000   3rd Qu.:0.0000     3rd Qu.:0.0000   3rd Qu.:1.0000  
##  Max.   :1.000   Max.   :1.0000     Max.   :1.0000   Max.   :1.0000  
##    CreditCard   
##  Min.   :0.000  
##  1st Qu.:0.000  
##  Median :0.000  
##  Mean   :0.294  
##  3rd Qu.:1.000  
##  Max.   :1.000
```

```r
# To check variables' class
lapply(Bank_data,class)
```

```
## $ID
## [1] "integer"
## 
## $Age
## [1] "integer"
## 
## $Experience
## [1] "integer"
## 
## $Income
## [1] "integer"
## 
## $ZIP.Code
## [1] "integer"
## 
## $Family
## [1] "integer"
## 
```

```
## $CCAvg
## [1] "numeric"
##
## $Education
## [1] "integer"
##
## $Mortgage
## [1] "integer"
##
## $Personal.Loan
## [1] "integer"
##
## $Securities.Account
## [1] "integer"
##
## $CD.Account
## [1] "integer"
##
## $Online
## [1] "integer"
##
## $CreditCard
## [1] "integer"
```

```
str(Bank_data)
```

```
## 'data.frame':    5000 obs. of  14 variables:
##  $ ID                : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Age               : int  25 45 39 35 35 37 53 50 35 34 ...
##  $ Experience        : int  1 19 15 9 8 13 27 24 10 9 ...
##  $ Income            : int  49 34 11 100 45 29 72 22 81 180 ...
##  $ ZIP.Code          : int  91107 90089 94720 94112 91330 92121 91711 93943 90089 93023 ...
##  $ Family            : int  4 3 1 1 4 4 2 1 3 1 ...
##  $ CCAvg             : num  1.6 1.5 1 2.7 1 0.4 1.5 0.3 0.6 8.9 ...
##  $ Education         : int  1 1 1 2 2 2 2 3 2 3 ...
##  $ Mortgage          : int  0 0 0 0 0 155 0 0 104 0 ...
##  $ Personal.Loan     : int  0 0 0 0 0 0 0 0 0 1 ...
##  $ Securities.Account: int  1 1 0 0 0 0 0 0 0 0 ...
##  $ CD.Account        : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Online            : int  0 0 0 0 0 1 1 0 1 0 ...
##  $ CreditCard        : int  0 0 0 0 1 0 0 1 0 0 ...
```

checking missing data if any

```
# to check missing data
Bank_data[!complete.cases(Bank_data),]
```

```
##  [1] ID                 Age                Experience         Income
##  [5] ZIP.Code           Family             CCAvg              Education
##  [9] Mortgage           Personal.Loan      Securities.Account CD.Account
## [13] Online             CreditCard
## <0 rows> (or 0-length row.names)
```

Remove data where customer age is less than 18 or More than 65

```r
# checking customers with Age<=18 and Age>65 (As Bank doesn't want to target them)

data_less18_ge65<-Bank_data%>% filter(Age<=18 | Age> 65)
nrow(data_less18_ge65)
```

```
## [1] 36
```

```r
# Removing customers with age less than 18 and greater than 65
Bank_data <- subset(Bank_data,Age>18 & Age< 65)
Bank_data %>% filter(Age<=18 | Age> 65)
```

```
##  [1] ID                 Age                Experience         Income
##  [5] ZIP.Code           Family             CCAvg              Education
##  [9] Mortgage           Personal.Loan      Securities.Account CD.Account
## [13] Online             CreditCard
## <0 rows> (or 0-length row.names)
```

```r
nrow(Bank_data)
```

```
## [1] 4884
```

Removing wrong data entry (where experience is mentioned as negative)

```r
# checking customers with negative experience

negative_exp<-Bank_data %>% filter(Experience<0)
nrow(negative_exp)
```

```
## [1] 52
```

```r
# Removing those 52 rows (issue with data having experience less than 0)
Bank_data <- subset(Bank_data,Experience>0)
nrow(Bank_data)
```

```
## [1] 4766
```

```r
Bank_data %>% filter(Experience<0)
```

```
##  [1] ID                 Age                Experience         Income
##  [5] ZIP.Code           Family             CCAvg              Education
##  [9] Mortgage           Personal.Loan      Securities.Account CD.Account
## [13] Online             CreditCard
## <0 rows> (or 0-length row.names)
```

Removing ZIP.Code from data, as this is not going to be considered as a predictor

```r
dim(Bank_data)
```

```
## [1] 4766   14
```

```r
Bank_data <- Bank_data %>% select(-ZIP.Code)
dim(Bank_data)
```

```
## [1] 4766   13
```

Deleting rows with missing values if any

```r
Bank_data<- na.omit(Bank_data)
dim(Bank_data)
```

```
## [1] 4766   13
```

Converting necessary variables from integer to Factor

```r
# Changing the class of variables
Bank_data$Personal.Loan <- as.factor(Bank_data$Personal.Loan)
Bank_data$Securities.Account <-as.factor(Bank_data$Securities.Account)
Bank_data$CD.Account <-as.factor(Bank_data$CD.Account)
Bank_data$Online <-as.factor(Bank_data$Online)
Bank_data$CreditCard<-as.factor(Bank_data$CreditCard)
Bank_data$Family<-as.factor(Bank_data$Family)
Bank_data$Education<-as.factor(Bank_data$Education)
str(Bank_data)
```

```
## 'data.frame':    4766 obs. of  13 variables:
##  $ ID                : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Age               : int  25 45 39 35 35 37 53 50 35 34 ...
##  $ Experience        : int  1 19 15 9 8 13 27 24 10 9 ...
##  $ Income            : int  49 34 11 100 45 29 72 22 81 180 ...
##  $ Family            : Factor w/ 4 levels "1","2","3","4": 4 3 1 1 4 4 2 1 3 1 ...
##  $ CCAvg             : num  1.6 1.5 1 2.7 1 0.4 1.5 0.3 0.6 8.9 ...
##  $ Education         : Factor w/ 3 levels "1","2","3": 1 1 1 2 2 2 2 3 2 3 ...
##  $ Mortgage          : int  0 0 0 0 0 155 0 0 104 0 ...
##  $ Personal.Loan     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ Securities.Account: Factor w/ 2 levels "0","1": 2 2 1 1 1 1 1 1 1 1 ...
##  $ CD.Account        : 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 ...
```

```r
summary(Bank_data)
```

```
##        ID            Age          Experience        Income       Family
##  Min.   :   1   Min.   :25.00   Min.   : 1.00   Min.   :  8.0   1:1425
##  1st Qu.:1251   1st Qu.:36.00   1st Qu.:11.00   1st Qu.: 39.0   2:1236
##  Median :2490   Median :45.00   Median :20.00   Median : 64.0   3: 951
##  Mean   :2495   Mean   :45.35   Mean   :20.13   Mean   : 73.8   4:1154
##  3rd Qu.:3731   3rd Qu.:55.00   3rd Qu.:29.00   3rd Qu.: 98.0
##  Max.   :5000   Max.   :64.00   Max.   :40.00   Max.   :224.0
##      CCAvg        Education    Mortgage       Personal.Loan Securities.Account
##  Min.   : 0.00   1:2019     Min.   :  0.00   0:4307        0:4270
##  1st Qu.: 0.70   2:1336     1st Qu.:  0.00   1: 459        1: 496
##  Median : 1.50   3:1411     Median :  0.00
##  Mean   : 1.94              Mean   : 57.16
##  3rd Qu.: 2.60              3rd Qu.:102.00
##  Max.   :10.00              Max.   :635.00
##  CD.Account Online   CreditCard
##  0:4480     0:1908   0:3372
##  1: 286     1:2858   1:1394
##
##
##
##
```

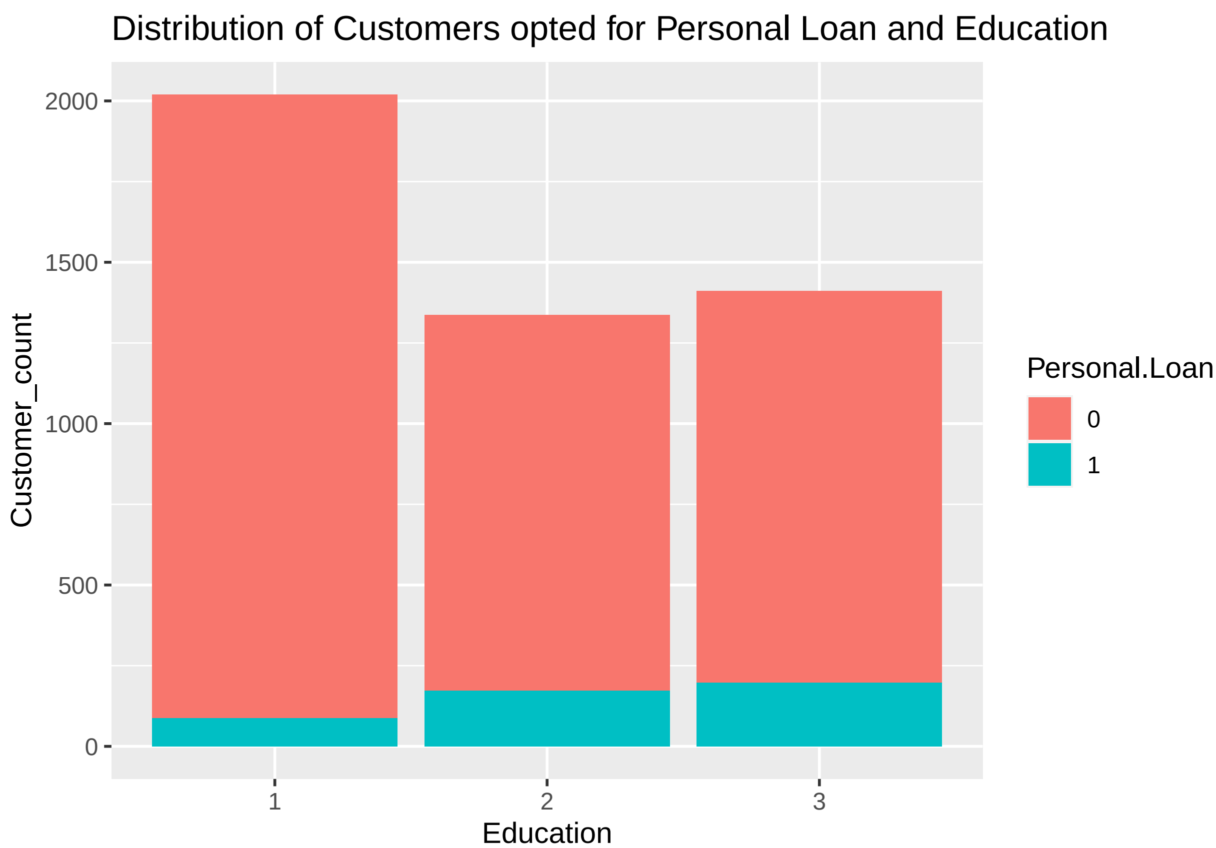
Plotting the variables to understand the predictors

```r
# 1. Personal Loan Distribution

  Bank_data %>% group_by(Personal.Loan)%>%
summarise(Customer_count=n())%>% ggplot(aes(Personal.Loan,Customer_count,fill=Personal.Loan))+
geom_bar(stat="identity")+ggtitle("Distribution of Customers opted for Personal Loan")
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

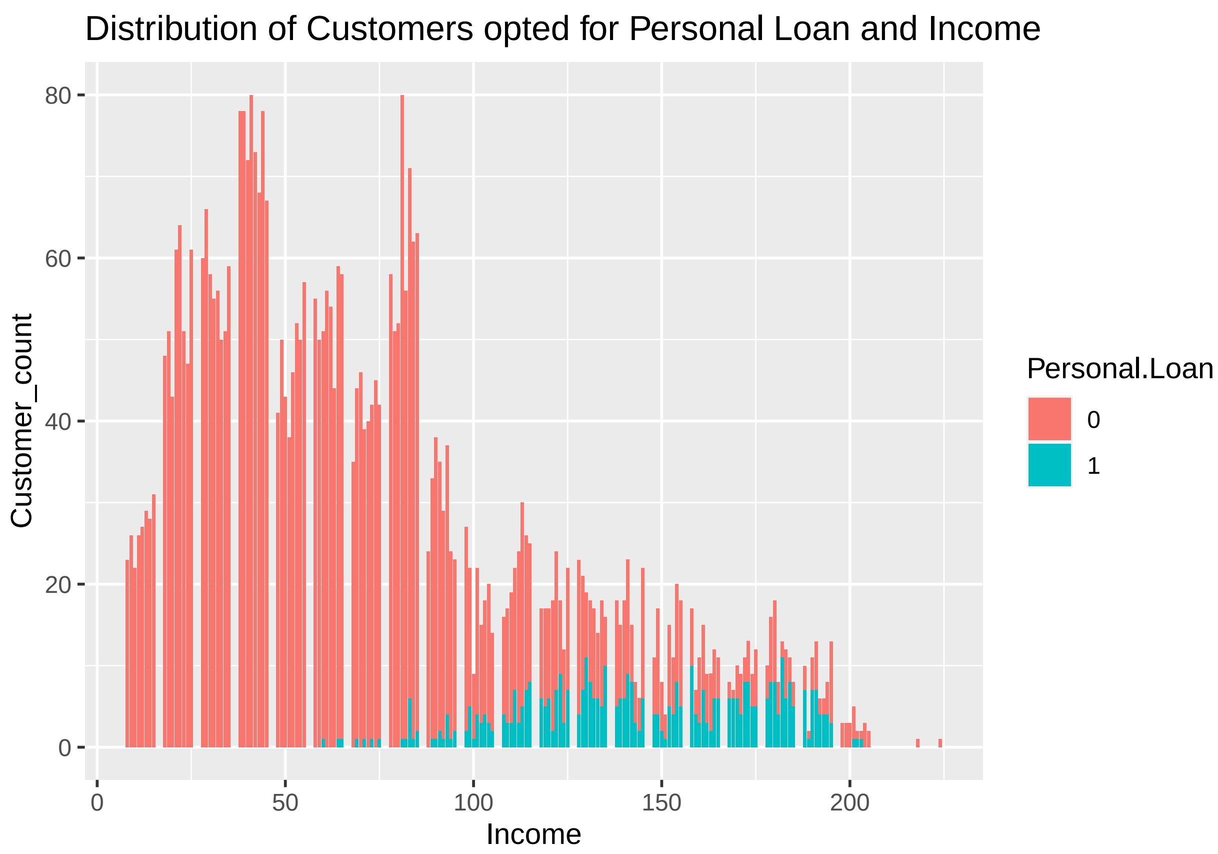

```
# 2. Personal Loan  & Education Distribution

  Bank_data %>% group_by(Personal.Loan,Education)%>%
    summarise(Customer_count=n())%>% ggplot(aes(Education,Customer_count,fill=Personal.Loan))+
    geom_bar(stat="identity")+ggtitle("Distribution of Customers opted for Personal Loan and Education")
```

```
## `summarise()` regrouping output by 'Personal.Loan' (override with `.groups` argument)
```

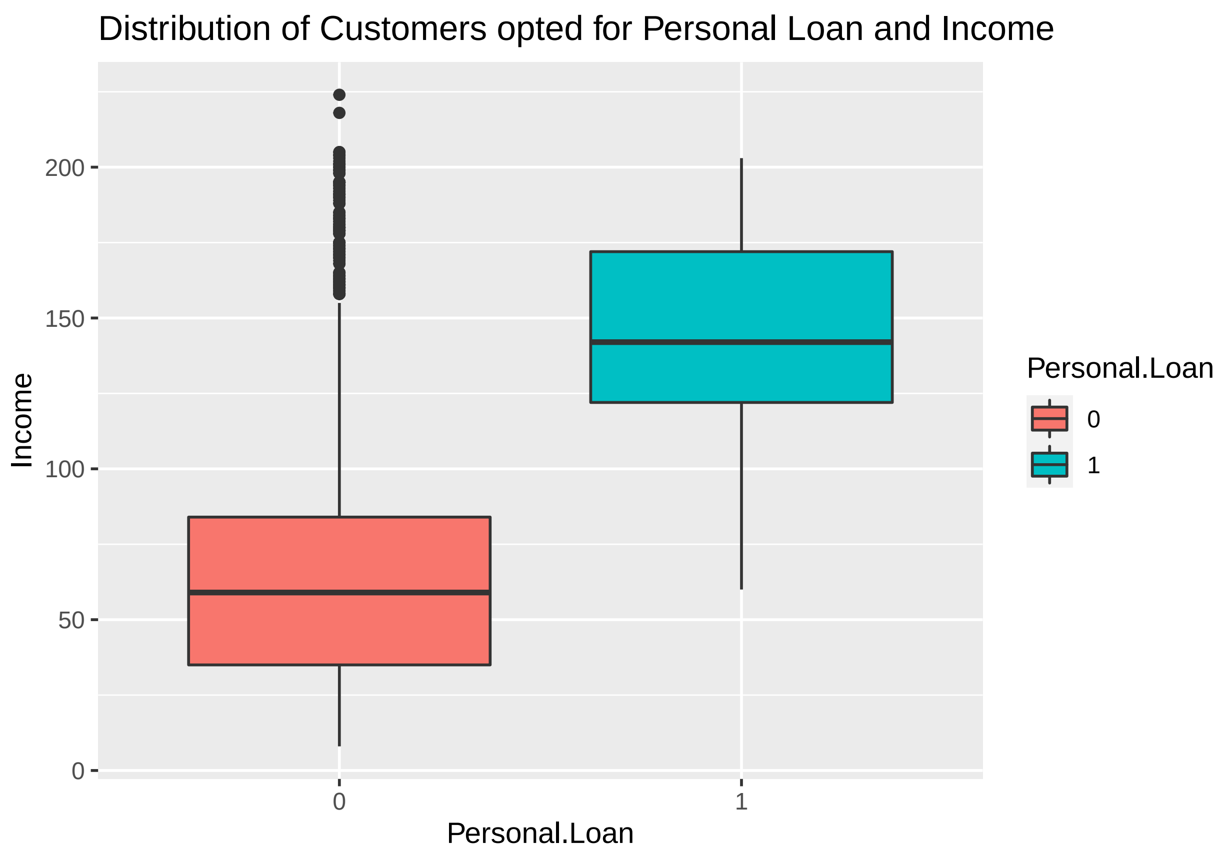



```r
# 3. Personal Loan  & Income Distribution

Bank_data %>% group_by(Personal.Loan,Income)%>%
  summarise(Customer_count=n())%>% ggplot(aes(Income,Customer_count,fill=Personal.Loan))+
  geom_bar(stat="identity")+ggtitle("Distribution of Customers opted for Personal Loan and Income")
```

```
## `summarise()` regrouping output by 'Personal.Loan' (override with `.groups` argument)
```

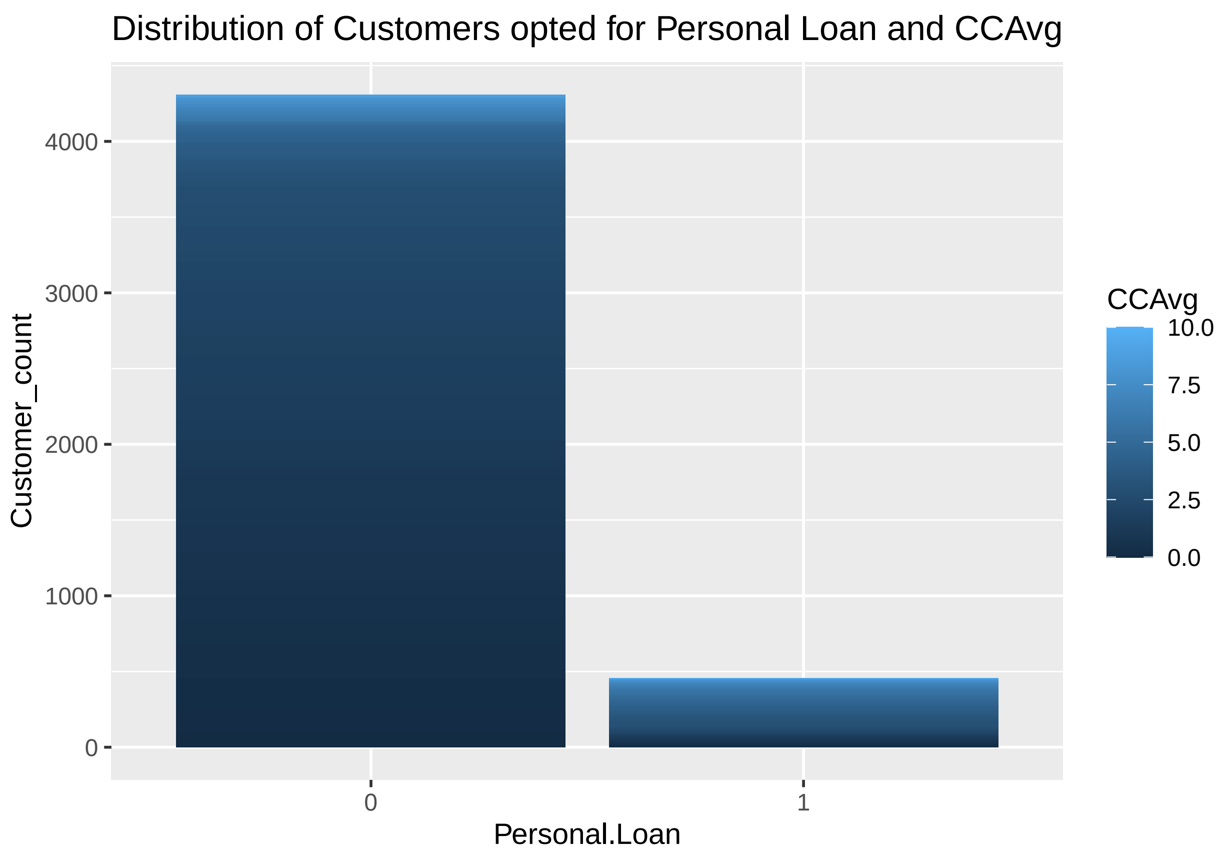



```
Bank_data %>%ggplot(aes(Personal.Loan,Income,fill=Personal.Loan))+geom_boxplot()+ggtitle("Distribution o
```

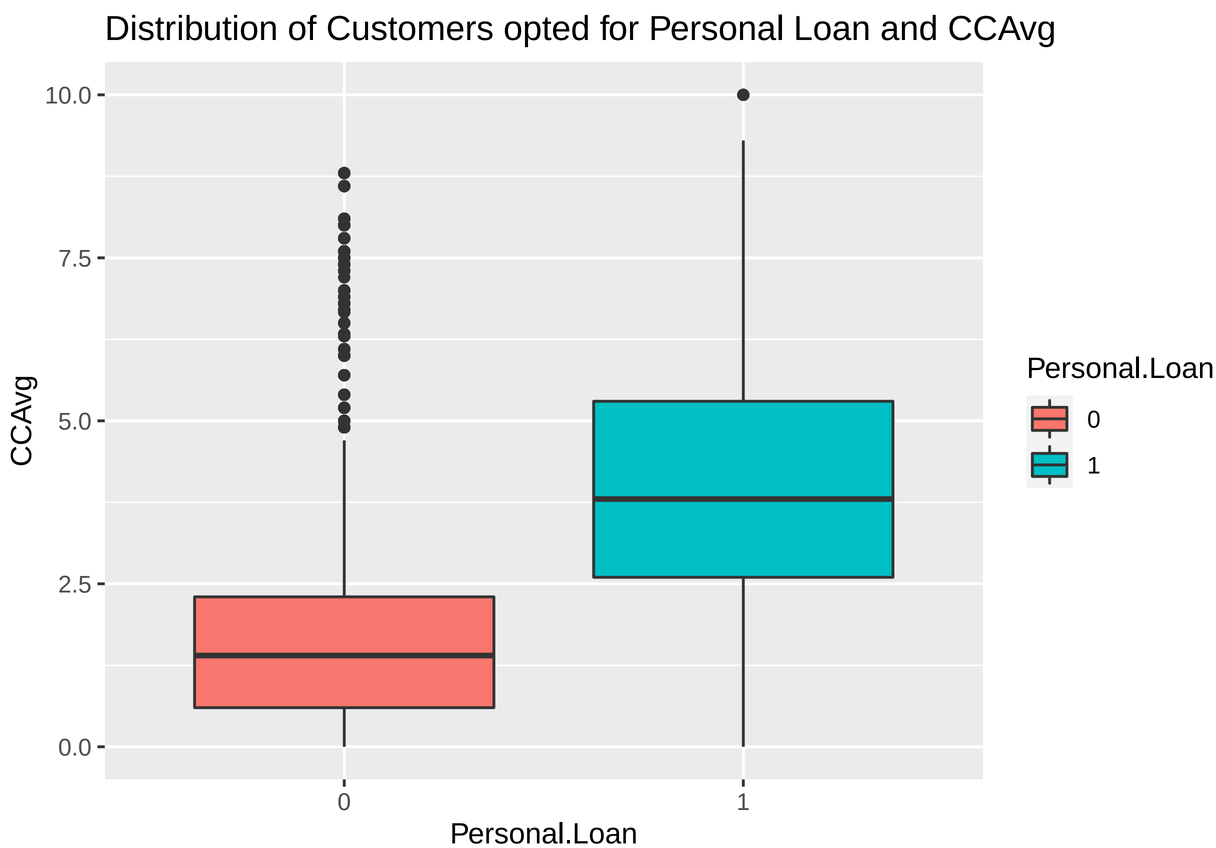



```r
# 4. Personal Loan and CCAvg Distribution
  Bank_data %>% group_by(Personal.Loan,CCAvg)%>%
    summarise(Customer_count=n())%>% ggplot(aes(Personal.Loan,Customer_count,fill=CCAvg))+
    geom_bar(stat="identity")+ggtitle("Distribution of Customers opted for Personal Loan and CCAvg")
```

```
## `summarise()` regrouping output by 'Personal.Loan' (override with `.groups` argument)
```

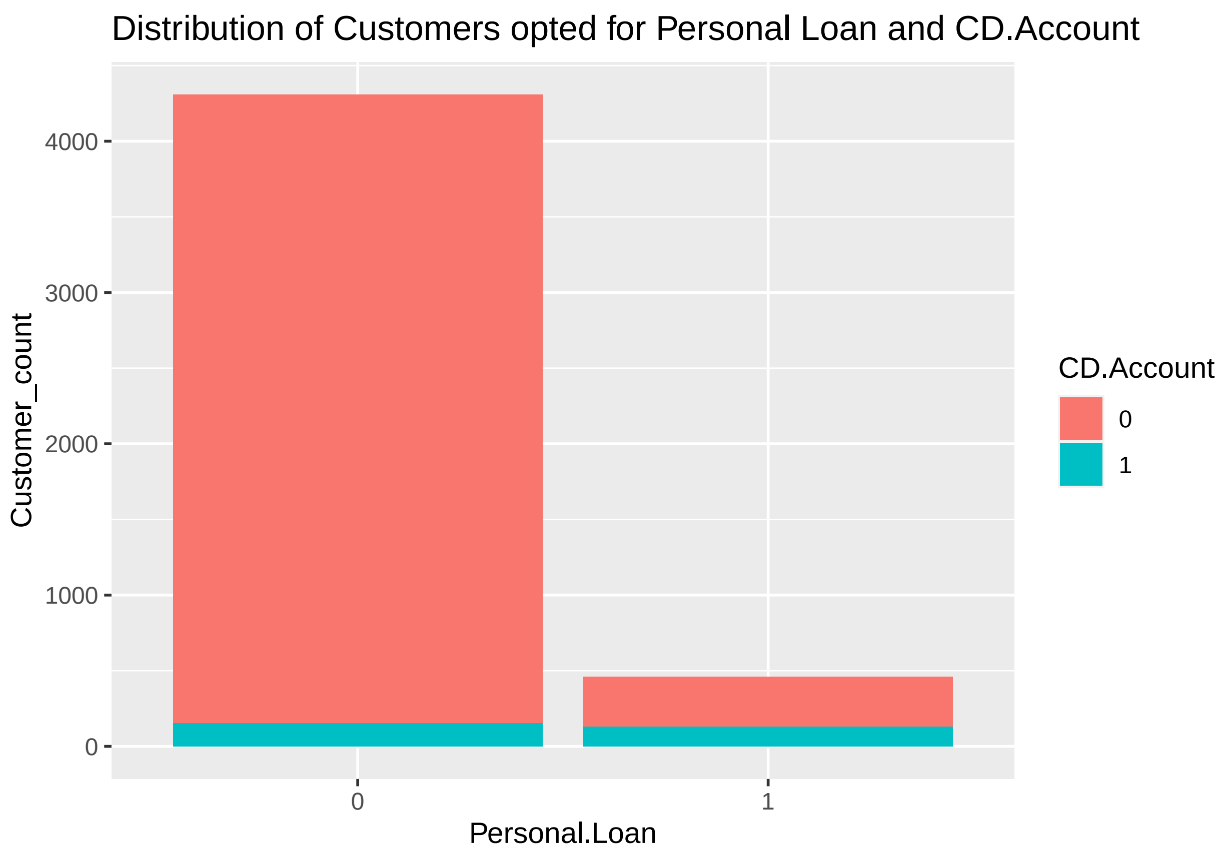



```
Bank_data %>%ggplot(aes(Personal.Loan,CCAvg, fill=Personal.Loan))+geom_boxplot()+ggtitle("Distributio
```



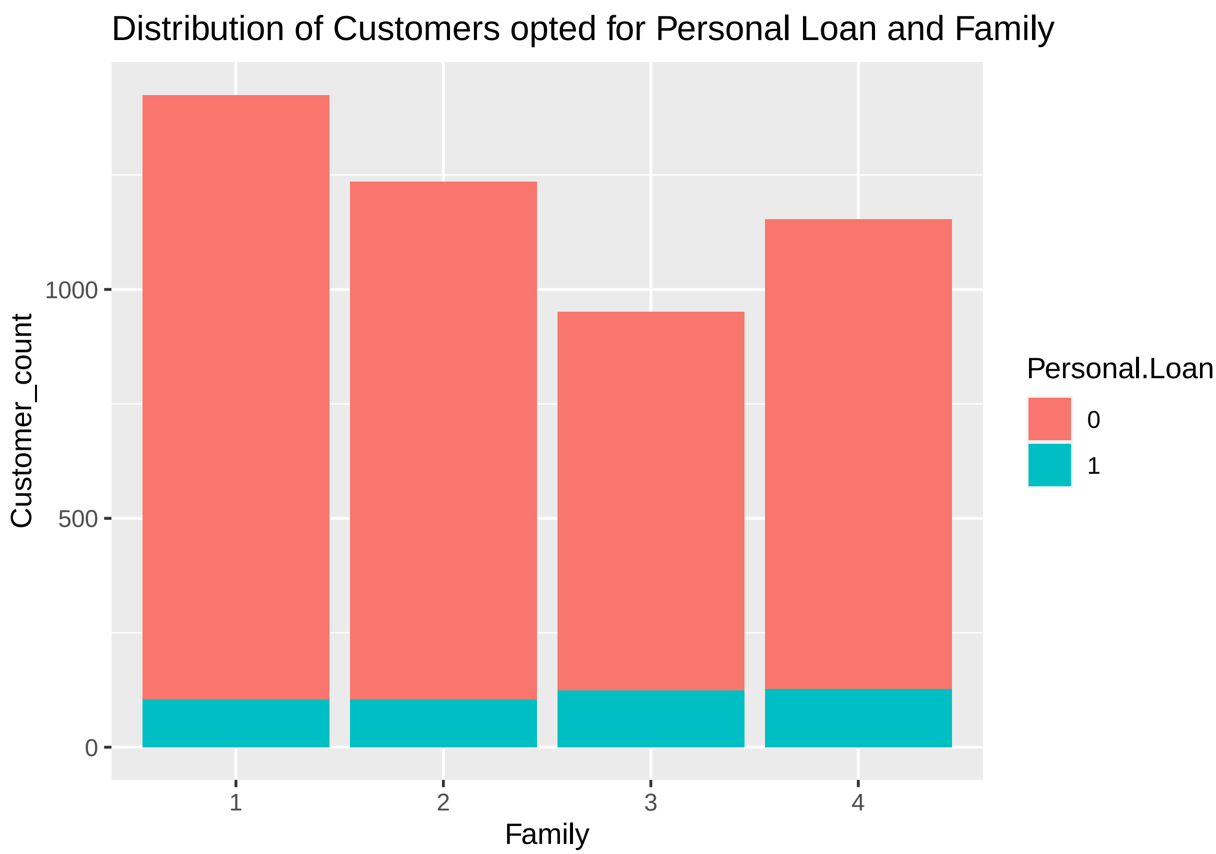

```r
#5.Personal Loan and CD.Account Distribution

Bank_data %>% group_by(Personal.Loan,CD.Account)%>%
  summarise(Customer_count=n())%>% ggplot(aes(Personal.Loan,Customer_count,fill=CD.Account))+
  geom_bar(stat="identity")+ggtitle("Distribution of Customers opted for Personal Loan and CD.Account
```

```
## `summarise()` regrouping output by 'Personal.Loan' (override with `.groups` argument)
```




```
# 6. Family and Personal.Loan
  Bank_data %>% group_by(Personal.Loan,Family)%>%
    summarise(Customer_count=n())%>% ggplot(aes(Family,Customer_count,fill=Personal.Loan))+
    geom_bar(stat="identity")+ggtitle("Distribution of Customers opted for Personal Loan and Family")
```

```
## `summarise()` regrouping output by 'Personal.Loan' (override with `.groups` argument)
```



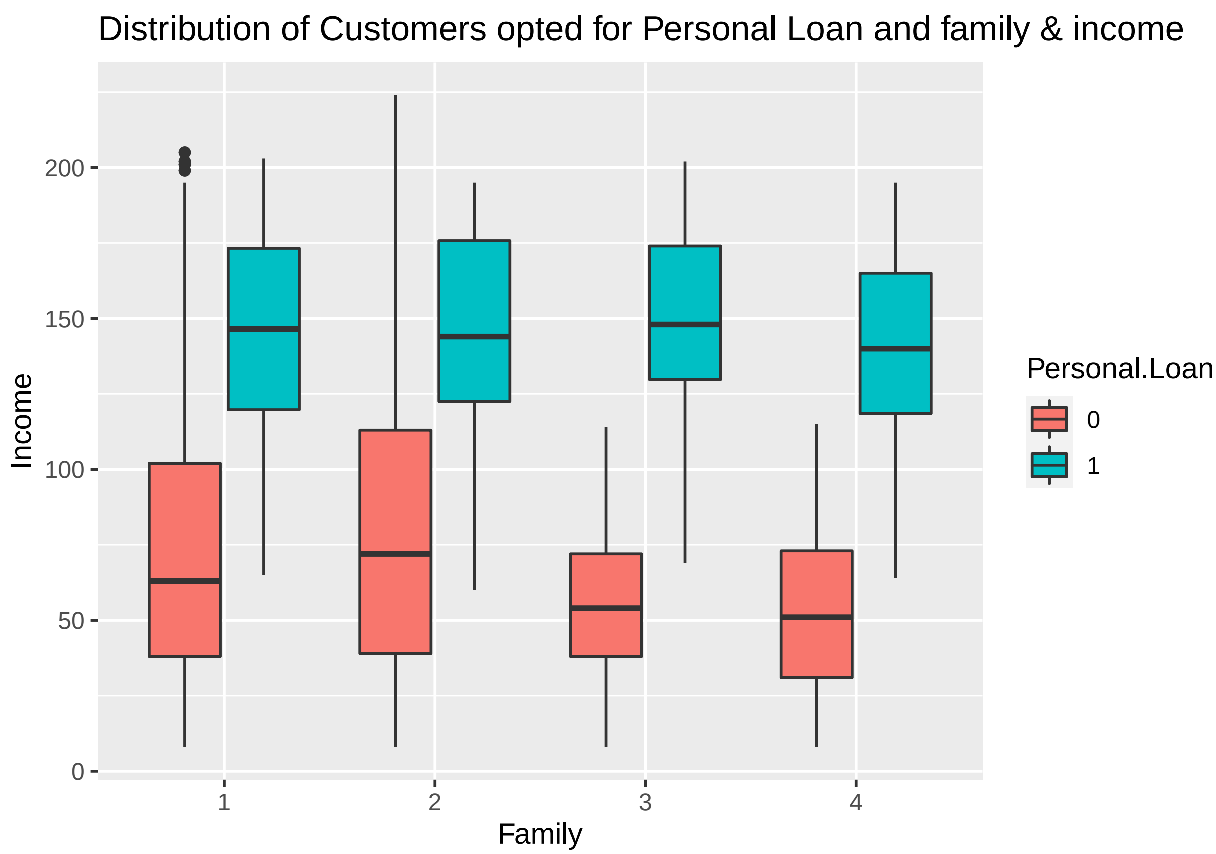

```r
# 7. Online & Personal.Loan
Bank_data %>% group_by(Personal.Loan,Online)%>%
  summarise(Customer_count=n())%>% ggplot(aes(Online,Customer_count,fill=Personal.Loan))+
  geom_bar(stat="identity")+ggtitle("Distribution of Customers opted for Personal Loan and Online")
```

```
## `summarise()` regrouping output by 'Personal.Loan' (override with `.groups` argument)
```


```r
#8. Credit card & Personal Loan
Bank_data %>% group_by(Personal.Loan,CreditCard)%>%
  summarise(Customer_count=n())%>% ggplot(aes(CreditCard,Customer_count,fill=Personal.Loan))+
  geom_bar(stat="identity")+ggtitle("Distribution of Customers opted for Personal Loan and Credit car
```

```
## `summarise()` regrouping output by 'Personal.Loan' (override with `.groups` argument)
```


```
# 9. Family , Income and personal Loan

Bank_data %>%ggplot(aes(Family,Income, fill=Personal.Loan))+geom_boxplot()+ggtitle("Distribution of Cu
```



Finding from plotting

1) Graduate & Advanced/Professional have opted for loans compared to undergraduate
2) Customers with Higher income (>$100K) opted for Personal Loan
3) Customers who opted for Personal loan have higher CCAVg
4) Family size doesn't have any significant impact
5) Customers having online accounts comparatively opted for Personal loan
6) Customers having credit card has less personal Loan
7) Family with income less than $100k is more unlikely to take the loan

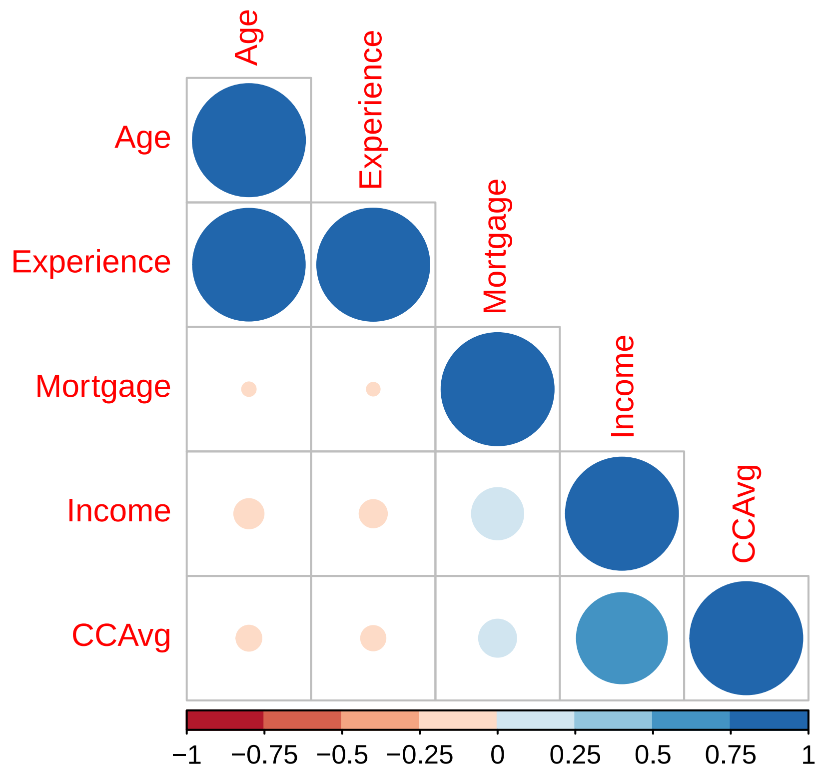
Creating correlation matrix

```r
# Correlation Matrix

  cor_dat <- Bank_data %>% select(Age, Experience, Income, CCAvg, Mortgage)

  head(cor_dat)
```

```
##   Age Experience Income CCAvg Mortgage
## 1  25          1     49   1.6        0
## 2  45         19     34   1.5        0
```

```
## 3  39          15      11    1.0         0
## 4  35           9     100    2.7         0
## 5  35           8      45    1.0         0
## 6  37          13      29    0.4       155
```

```r
temp <- cor_dat %>% select(one_of("Age", "Experience", "Income", "CCAvg", "Mortgage"
                                 )) %>% as.matrix()
M <- cor(temp, use = "pairwise.complete.obs")

corrplot(M, order = "hclust", addrect = 2, type = "lower", col = brewer.pal(n = 8, name = "RdBu"))
```




```r
cor(data.frame(x = Bank_data$Income, y = Bank_data$CCAvg))
```

```
##           x         y
## x 1.0000000 0.6461494
## y 0.6461494 1.0000000
```

```r
cor(data.frame(x = Bank_data$Age, y = Bank_data$Experience))
```

```
##           x         y
## x 1.0000000 0.9935402
## y 0.9935402 1.0000000
```

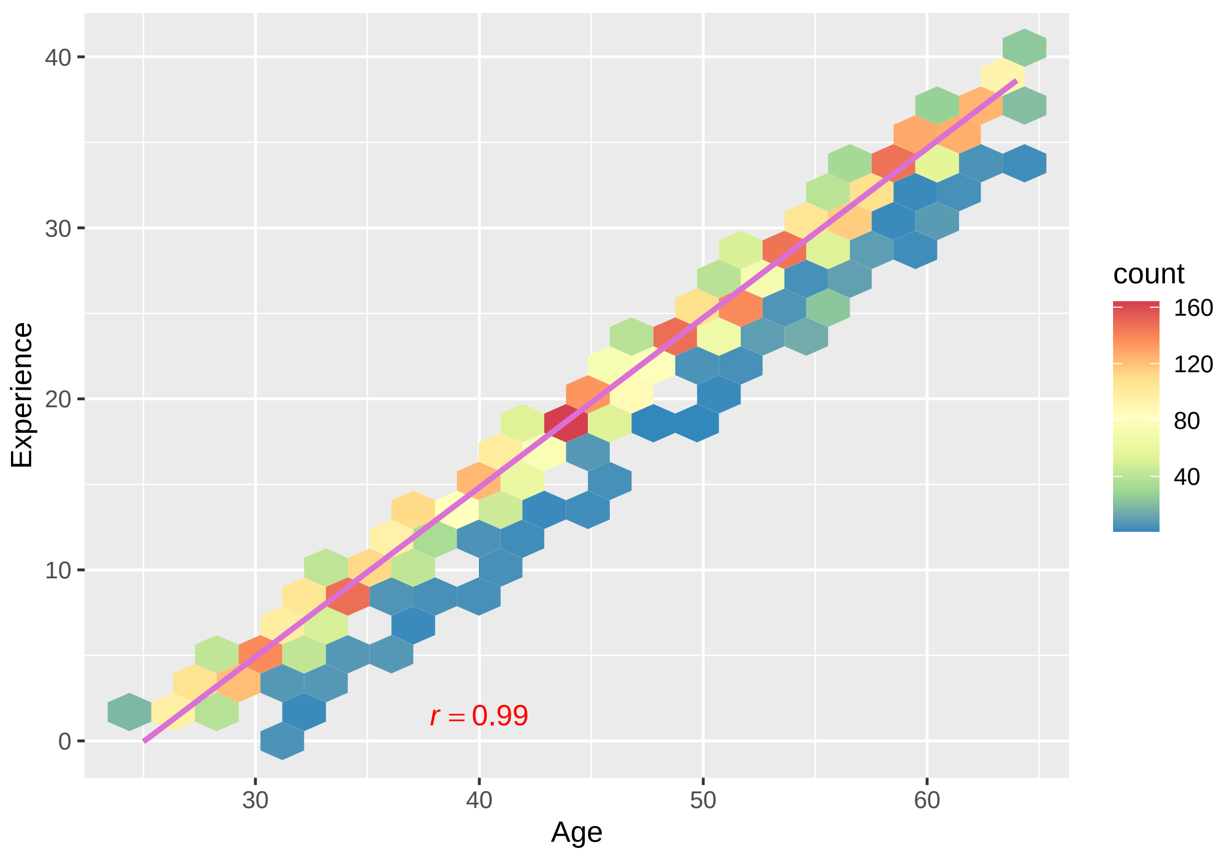
```
get_cor <- function(df){
  m <- cor(df$x, df$y, use="pairwise.complete.obs");
  eq <- substitute(italic(r) == cor, list(cor = format(m, digits = 2)))
  as.character(as.expression(eq));
}
```

Observation: Income and CCAvg are highly correlated and Age and Experience also highly correlated

Age vs Experience

```
cor_dat %>%
    ggplot(aes(Age, Experience)) + stat_bin_hex(bins = 20) + scale_fill_distiller(palette = "Spectral")
    stat_smooth(method = "lm", color = "orchid", size = 1) +
    annotate("text", x = 40, y = 1.5, label = get_cor(data.frame(x = cor_dat$Age, y = cor_dat$Experience
                 parse = TRUE, color = "red", size = 4) + ylab("Experience") + xlab("Age")
```

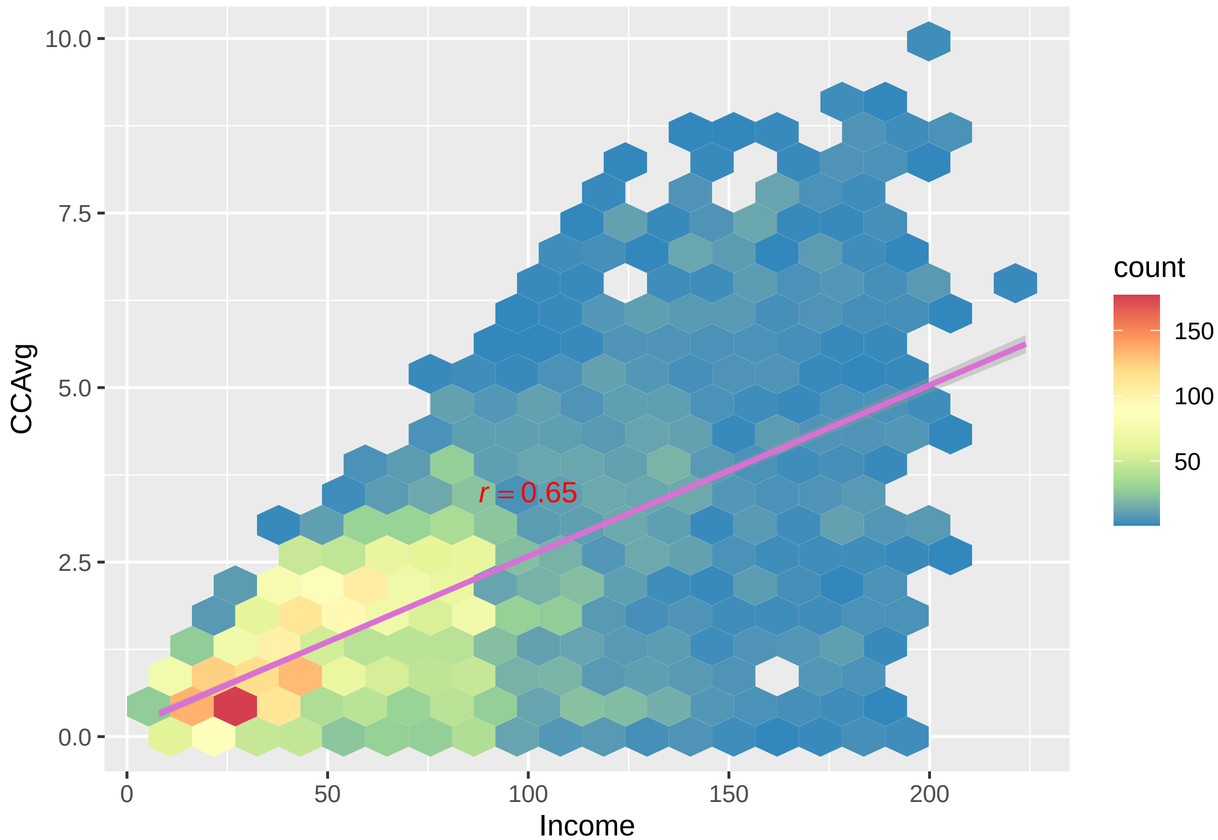
```
## `geom_smooth()` using formula 'y ~ x'
```



Income vs CCAvg

```r
cor_dat %>%
    ggplot(aes(Income, CCAvg)) + stat_bin_hex(bins = 20) + scale_fill_distiller(palette = "Spectral") +
    stat_smooth(method = "lm", color = "orchid", size = 1) +
    annotate("text", x = 100, y = 3.5, label = get_cor(data.frame(x = cor_dat$Income, y = cor_dat$CCAvg
             parse = TRUE, color = "red", size = 4) + ylab("CCAvg") + xlab("Income")
```

```
## `geom_smooth()` using formula 'y ~ x'
```



Creating dummy variables

Creating dummy variables for Family and Education as both have multiple levels

```r
Bank_data_new<- dummy.data.frame(Bank_data,name=c("Family","Education"),sep=".")
```

```
## Warning in model.matrix.default(~x - 1, model.frame(~x - 1), contrasts = FALSE):
## non-list contrasts argument ignored

## Warning in model.matrix.default(~x - 1, model.frame(~x - 1), contrasts = FALSE):
## non-list contrasts argument ignored
```

```r
str(Bank_data_new)
```

```
## 'data.frame':    4766 obs. of  18 variables:
##  $ ID                : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Age               : int  25 45 39 35 35 37 53 50 35 34 ...
##  $ Experience        : int  1 19 15 9 8 13 27 24 10 9 ...
##  $ Income            : int  49 34 11 100 45 29 72 22 81 180 ...
##  $ Family.1          : int  0 0 1 1 0 0 0 1 0 1 ...
##  $ Family.2          : int  0 0 0 0 0 0 1 0 0 0 ...
##  $ Family.3          : int  0 1 0 0 0 0 0 0 1 0 ...
##  $ Family.4          : int  1 0 0 0 1 1 0 0 0 0 ...
##  $ CCAvg             : num  1.6 1.5 1 2.7 1 0.4 1.5 0.3 0.6 8.9 ...
##  $ Education.1       : int  1 1 1 0 0 0 0 0 0 0 ...
##  $ Education.2       : int  0 0 0 1 1 1 1 0 1 0 ...
##  $ Education.3       : int  0 0 0 0 0 0 0 1 0 1 ...
##  $ Mortgage          : int  0 0 0 0 0 155 0 0 104 0 ...
##  $ Personal.Loan     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ Securities.Account: Factor w/ 2 levels "0","1": 2 2 1 1 1 1 1 1 1 1 ...
##  $ CD.Account        : 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 ...
##  - attr(*, "dummies")=List of 2
##   ..$ Family   : int [1:4] 5 6 7 8
##   ..$ Education: int [1:3] 10 11 12
```

```r
Bank_data_new$Family.1<- as.factor(Bank_data_new$Family.1)
Bank_data_new$Family.2<- as.factor(Bank_data_new$Family.2)
Bank_data_new$Family.3<- as.factor(Bank_data_new$Family.3)
Bank_data_new$Family.4<- as.factor(Bank_data_new$Family.4)
Bank_data_new$Education.1<- as.factor(Bank_data_new$Education.1)
Bank_data_new$Education.2<- as.factor(Bank_data_new$Education.2)
Bank_data_new$Education.3<- as.factor(Bank_data_new$Education.3)

str(Bank_data_new)
```

```
## 'data.frame':    4766 obs. of  18 variables:
##  $ ID                : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ Age               : int  25 45 39 35 35 37 53 50 35 34 ...
##  $ Experience        : int  1 19 15 9 8 13 27 24 10 9 ...
##  $ Income            : int  49 34 11 100 45 29 72 22 81 180 ...
##  $ Family.1          : Factor w/ 2 levels "0","1": 1 1 2 2 1 1 1 2 1 2 ...
##  $ Family.2          : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 2 1 1 1 ...
##  $ Family.3          : Factor w/ 2 levels "0","1": 1 2 1 1 1 1 1 1 2 1 ...
##  $ Family.4          : Factor w/ 2 levels "0","1": 2 1 1 1 2 2 1 1 1 1 ...
##  $ CCAvg             : num  1.6 1.5 1 2.7 1 0.4 1.5 0.3 0.6 8.9 ...
##  $ Education.1       : Factor w/ 2 levels "0","1": 2 2 2 1 1 1 1 1 1 1 ...
##  $ Education.2       : Factor w/ 2 levels "0","1": 1 1 1 2 2 2 2 1 2 1 ...
##  $ Education.3       : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 2 1 2 ...
##  $ Mortgage          : int  0 0 0 0 0 155 0 0 104 0 ...
##  $ Personal.Loan     : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 1 1 2 ...
##  $ Securities.Account: Factor w/ 2 levels "0","1": 2 2 1 1 1 1 1 1 1 1 ...
##  $ CD.Account        : 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 ...
##  - attr(*, "dummies")=List of 2
##   ..$ Family   : int [1:4] 5 6 7 8
##   ..$ Education: int [1:3] 10 11 12
```

```r
dim(Bank_data_new)
```

```
## [1] 4766   18
```

Test and training data set creation

Removing ID as we are not going to use this as predictor

```r
# Removing ID as this is not going to use in our Model
Bank_data_new <- Bank_data_new %>% select(-ID)
dim(Bank_data_new)
```

```
## [1] 4766   17
```

```r
# Test and Train data set creation
test_index <- createDataPartition(y = Bank_data_new$Personal.Loan, times = 1, p = 0.2,
                                  list = FALSE)
train_set <- Bank_data_new[-test_index,]
test_set <- Bank_data_new[test_index,]

dim(train_set)
```

```
## [1] 3812   17
```

```r
dim(test_set)
```

```
## [1] 954  17
```

```r
head(train_set)
```

```
##   Age Experience Income Family.1 Family.2 Family.3 Family.4 CCAvg Education.1
## 2  45         19     34        0        0        1        0   1.5           1
## 3  39         15     11        1        0        0        0   1.0           1
## 4  35          9    100        1        0        0        0   2.7           0
## 5  35          8     45        0        0        0        1   1.0           0
## 6  37         13     29        0        0        0        1   0.4           0
## 7  53         27     72        0        1        0        0   1.5           0
##   Education.2 Education.3 Mortgage Personal.Loan Securities.Account CD.Account
## 2           0           0        0             0                  1          0
## 3           0           0        0             0                  0          0
## 4           1           0        0             0                  0          0
## 5           1           0        0             0                  0          0
## 6           1           0      155             0                  0          0
```

```
## 7           1           0        0           0                  0          0
##   Online CreditCard
## 2      0          0
## 3      0          0
## 4      0          0
## 5      0          1
## 6      1          0
## 7      1          0
```

Different Model building approaches

Model 1:Logistic Regression

```
y<- train_set$Personal.Loan
  fit_glm <- glm(y~Age+Experience+Income+CCAvg+Family.1+Family.2+Family.3+Family.4+Education.1+Educatio
  p_hat_logistic <- predict(fit_glm, test_set,type = "response")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type == :
## prediction from a rank-deficient fit may be misleading
```

```
  y_hat_logistic <- factor(ifelse(p_hat_logistic > 0.5, 1, 0))
  confusionMatrix(data = y_hat_logistic, reference = test_set$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 851  30
##          1  11  62
##
##                Accuracy : 0.957
##                  95% CI : (0.9421, 0.969)
##     No Information Rate : 0.9036
##     P-Value [Acc > NIR] : 4.305e-10
##
##                   Kappa : 0.7283
##
##  Mcnemar's Test P-Value : 0.004937
##
##             Sensitivity : 0.9872
##             Specificity : 0.6739
##          Pos Pred Value : 0.9659
##          Neg Pred Value : 0.8493
##              Prevalence : 0.9036
##          Detection Rate : 0.8920
##    Detection Prevalence : 0.9235
##       Balanced Accuracy : 0.8306
##
##        'Positive' Class : 0
##
```

```r
confusionMatrix(data = y_hat_logistic, reference = test_set$Personal.Loan)$overall["Accuracy"]
```

```
##  Accuracy
## 0.9570231
```

```r
summary(fit_glm)
```

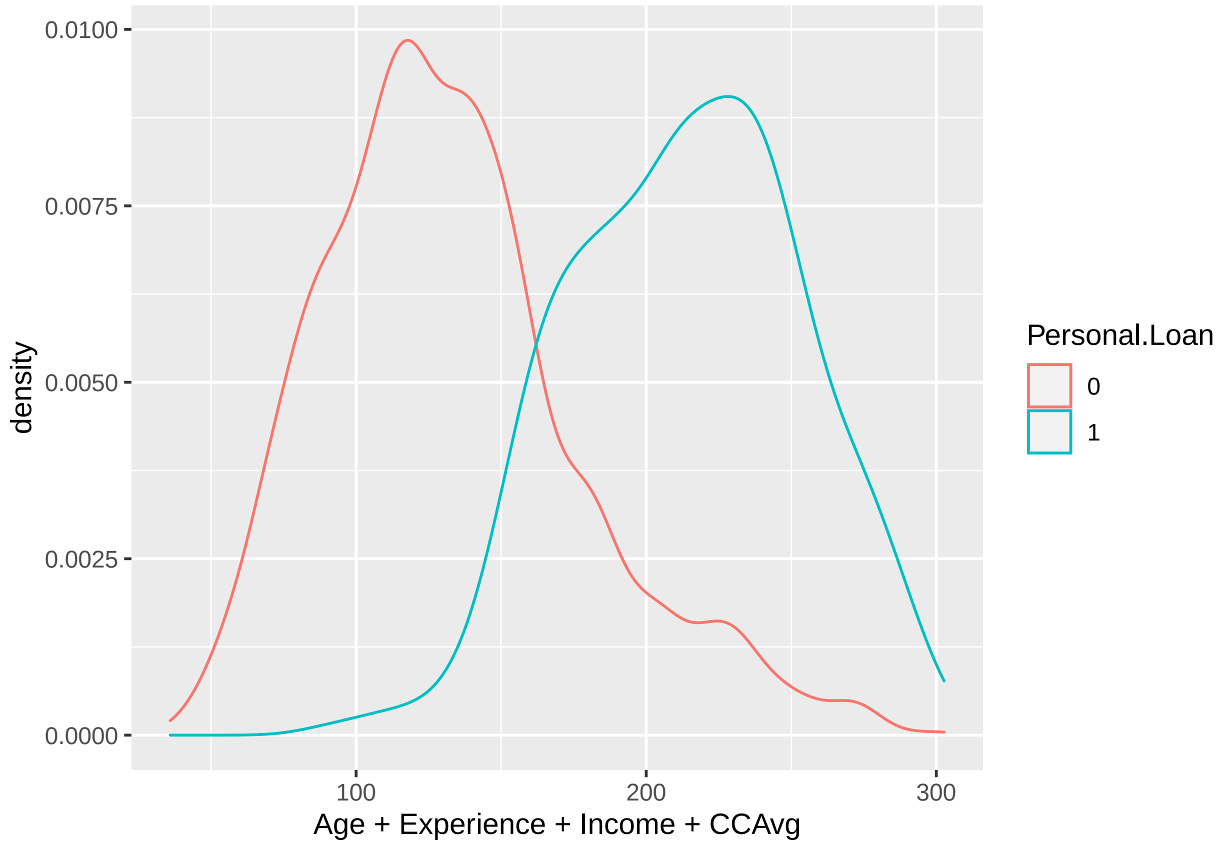
```
##
## Call:
## glm(formula = y ~ Age + Experience + Income + CCAvg + Family.1 +
##     Family.2 + Family.3 + Family.4 + Education.1 + Education.2 +
##     Education.3 + Mortgage + Securities.Account + CD.Account +
##     Online + CreditCard, family = "binomial", data = train_set)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -3.0798  -0.1763  -0.0603  -0.0163   4.1849
##
## Coefficients: (2 not defined because of singularities)
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)         -5.2407913  2.0917790  -2.505 0.012231 *
## Age                 -0.0792335  0.0790791  -1.002 0.316366
## Experience           0.0893503  0.0788287   1.133 0.257015
## Income               0.0648412  0.0037284  17.391  < 2e-16 ***
## CCAvg                0.1718849  0.0543946   3.160 0.001578 **
## Family.11           -1.6018200  0.2804290  -5.712 1.12e-08 ***
## Family.21           -1.7371625  0.2884497  -6.022 1.72e-09 ***
## Family.31            0.3809903  0.2604462   1.463 0.143512
## Family.41                   NA         NA      NA       NA
## Education.11        -4.3679819  0.3337356 -13.088  < 2e-16 ***
## Education.21        -0.2107100  0.2232794  -0.944 0.345320
## Education.31                NA         NA      NA       NA
## Mortgage             0.0004889  0.0007103   0.688 0.491242
## Securities.Account1 -0.8030008  0.3542622  -2.267 0.023409 *
## CD.Account1          3.7482962  0.4099441   9.143  < 2e-16 ***
## Online1             -0.8234541  0.1960963  -4.199 2.68e-05 ***
## CreditCard1         -0.8571351  0.2545400  -3.367 0.000759 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2415.44  on 3811  degrees of freedom
## Residual deviance:  833.93  on 3797  degrees of freedom
## AIC: 863.93
##
## Number of Fisher Scoring iterations: 8
```

```r
## anova chisq
anova(fit_glm,test="Chisq")
```

```
## Analysis of Deviance Table
```

```
##
## Model: binomial, link: logit
##
## Response: y
##
## Terms added sequentially (first to last)
##
##
##                   Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                              3811    2415.44
## Age                1     0.84      3810    2414.59 0.3585306
## Experience         1     0.01      3809    2414.58 0.9165821
## Income             1   906.06      3808    1508.53 < 2.2e-16 ***
## CCAvg              1     2.90      3807    1505.63 0.0886890 .
## Family.1           1    44.09      3806    1461.54 3.132e-11 ***
## Family.2           1   157.81      3805    1303.73 < 2.2e-16 ***
## Family.3           1     0.19      3804    1303.53 0.6589341
## Family.4           0     0.00      3804    1303.53
## Education.1        1   360.69      3803     942.84 < 2.2e-16 ***
## Education.2        1     1.02      3802     941.82 0.3115208
## Education.3        0     0.00      3802     941.82
## Mortgage           1     1.40      3801     940.42 0.2364624
## Securities.Account 1     4.80      3800     935.62 0.0285388 *
## CD.Account         1    74.73      3799     860.89 < 2.2e-16 ***
## Online             1    14.50      3798     846.40 0.0001404 ***
## CreditCard         1    12.46      3797     833.93 0.0004149 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
train_set %>% ggplot(aes(Age+Experience+Income+CCAvg, color = Personal.Loan)) + geom_density()
```




```r
# creating Model result table
model_result<-tibble(method = "Logistic Regression", Accuracy = confusionMatrix(data = y_hat_logistic
)

model_result
```

```
## # A tibble: 1 x 2
##   method              Accuracy
##   <chr>                  <dbl>
## 1 Logistic Regression    0.957
```

Model 2: K-nearest neighbors (kNN)

k=1

```r
# k=1
knn_fit<-knn3(y~Age+Experience+Income+CCAvg+Family.1+Family.2+Family.3+Family.4+Education.1+Education

y_hat_knn<-predict(knn_fit,test_set,type="class")

confusionMatrix(data = y_hat_knn, reference = test_set$Personal.Loan)
```

```
## Confusion Matrix and Statistics
```

```
##
##           Reference
## Prediction   0   1
##          0 812  58
##          1  50  34
##
##                Accuracy : 0.8868
##                  95% CI : (0.865, 0.9062)
##     No Information Rate : 0.9036
##     P-Value [Acc > NIR] : 0.9624
##
##                   Kappa : 0.3241
##
##  Mcnemar's Test P-Value : 0.5006
##
##             Sensitivity : 0.9420
##             Specificity : 0.3696
##          Pos Pred Value : 0.9333
##          Neg Pred Value : 0.4048
##              Prevalence : 0.9036
##          Detection Rate : 0.8512
##    Detection Prevalence : 0.9119
##       Balanced Accuracy : 0.6558
##
##        'Positive' Class : 0
##
```

```
  confusionMatrix(data = y_hat_knn, reference = test_set$Personal.Loan)$overall["Accuracy"]
```

```
##  Accuracy
## 0.8867925
```

To check the improvement, considering k=400

```
knn_fit<-knn3(y~Age+Experience+Income+CCAvg+Family.1+Family.2+Family.3+Family.4+Education.1+Education.2

  y_hat_knn<-predict(knn_fit,test_set,type="class")

  confusionMatrix(data = y_hat_knn, reference = test_set$Personal.Loan)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 862  92
##          1   0   0
##
##                Accuracy : 0.9036
##                  95% CI : (0.883, 0.9216)
##     No Information Rate : 0.9036
##     P-Value [Acc > NIR] : 0.5277
##
```

```
##                  Kappa : 0
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 1.0000
##             Specificity : 0.0000
##          Pos Pred Value : 0.9036
##          Neg Pred Value :    NaN
##              Prevalence : 0.9036
##          Detection Rate : 0.9036
##    Detection Prevalence : 1.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : 0
##
```

```
confusionMatrix(data = y_hat_knn, reference = test_set$Personal.Loan)$overall["Accuracy"]
```

```
##  Accuracy
## 0.9035639
```

optimum k value selection

```
ks<-seq(1,400,2)

 accuracy<-map_df(ks,function(k){

   knn_fit<-knn3(y~Age+Experience+Income+CCAvg+Family.1+Family.2+Family.3+Family.4+Education.1+Educati

   y_hat_knn<-predict(knn_fit,test_set,type="class")

   test_error<-confusionMatrix(data = y_hat_knn, reference = test_set$Personal.Loan)$overall["Accuracy
   tibble(test=test_error)
 }

   )
 # pick the k that maximizes accuracy using the estimate built on test data
 ks[which.max(accuracy$test)]
```

```
## [1] 35
```

```
max(accuracy$test)
```

```
## [1] 0.9056604
```

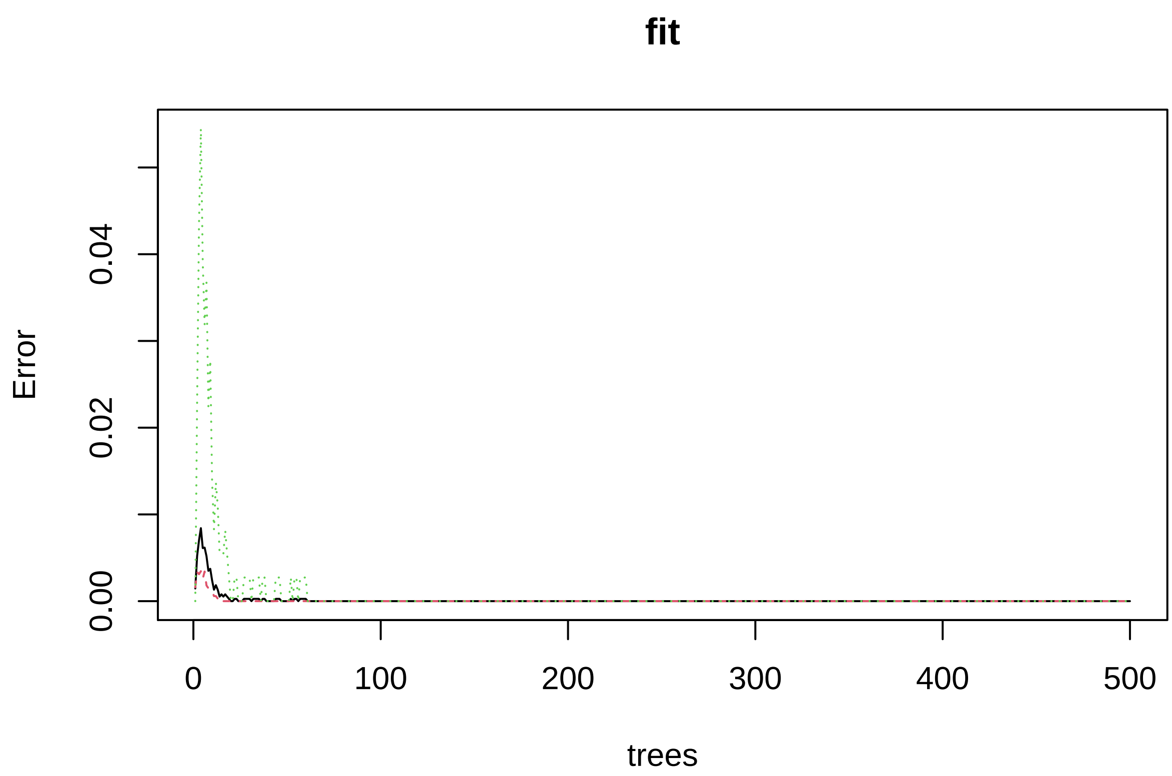
update model_result table

```r
model_result <- bind_rows (model_result ,tibble(method = "K-nearest neighbors (kNN)", Accuracy = max(ac
 model_result
```

```
## # A tibble: 2 x 2
##   method                      Accuracy
##   <chr>                          <dbl>
## 1 Logistic Regression            0.957
## 2 K-nearest neighbors (kNN)      0.906
```

Model 3:RandomForest

```r
fit <- randomForest(y~., data = train_set)
  plot(fit)
```




```r
# General model
  train_rf <- randomForest(Personal.Loan ~ ., data=train_set)
  confusionMatrix(predict(train_rf, test_set), test_set$Personal.Loan)$overall["Accuracy"]
```

```
##  Accuracy
## 0.9874214
```

update model_result table

```r
model_result <- bind_rows (model_result ,tibble(method = "RandomForest General", Accuracy = confusionMa
  model_result
```

```
## # A tibble: 3 x 2
##   method                    Accuracy
##   <chr>                        <dbl>
## 1 Logistic Regression          0.957
## 2 K-nearest neighbors (kNN)    0.906
## 3 RandomForest General         0.987
```

Model 4:Rborist

```r
# use cross validation to choose parameter
  train_rf_2 <- train(Personal.Loan ~ .,
                      method = "Rborist",
                      tuneGrid = data.frame(predFixed = 2, minNode = c(3, 50)),
                      data = train_set)
  confusionMatrix(predict(train_rf_2, test_set), test_set$Personal.Loan)$overall["Accuracy"]
```

```
##  Accuracy
## 0.9737945
```

```r
  control <- trainControl(method="cv", number = 5, p = 0.8)
  grid <- expand.grid(minNode = c(1,5) , predFixed = c(10, 15, 25, 35, 50))
  train_rf <-  train(Personal.Loan ~ .,
                     data=train_set,
                     method = "Rborist",
                     nTree = 50,
                     trControl = control,
                     tuneGrid = grid,
                     nSamp = 5000)
```

```
## Warning: model fit failed for Fold1: minNode=1, predFixed=25 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold1: minNode=5, predFixed=25 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold1: minNode=1, predFixed=35 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold1: minNode=5, predFixed=35 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold1: minNode=1, predFixed=50 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count
```

```
## Warning: model fit failed for Fold1: minNode=5, predFixed=50 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold2: minNode=1, predFixed=25 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold2: minNode=5, predFixed=25 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold2: minNode=1, predFixed=35 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold2: minNode=5, predFixed=35 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold2: minNode=1, predFixed=50 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold2: minNode=5, predFixed=50 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold3: minNode=1, predFixed=25 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold3: minNode=5, predFixed=25 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold3: minNode=1, predFixed=35 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold3: minNode=5, predFixed=35 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold3: minNode=1, predFixed=50 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold3: minNode=5, predFixed=50 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold4: minNode=1, predFixed=25 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold4: minNode=5, predFixed=25 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold4: minNode=1, predFixed=35 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count

## Warning: model fit failed for Fold4: minNode=5, predFixed=35 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count
```

```
## Warning: model fit failed for Fold4: minNode=1, predFixed=50 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count
```

```
## Warning: model fit failed for Fold4: minNode=5, predFixed=50 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count
```

```
## Warning: model fit failed for Fold5: minNode=1, predFixed=25 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count
```

```
## Warning: model fit failed for Fold5: minNode=5, predFixed=25 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count
```

```
## Warning: model fit failed for Fold5: minNode=1, predFixed=35 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count
```

```
## Warning: model fit failed for Fold5: minNode=5, predFixed=35 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count
```

```
## Warning: model fit failed for Fold5: minNode=1, predFixed=50 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count
```

```
## Warning: model fit failed for Fold5: minNode=5, predFixed=50 Error in Rborist.default(x, y, predFixe
##   'predFixed' must be positive integer <= predictor count
```

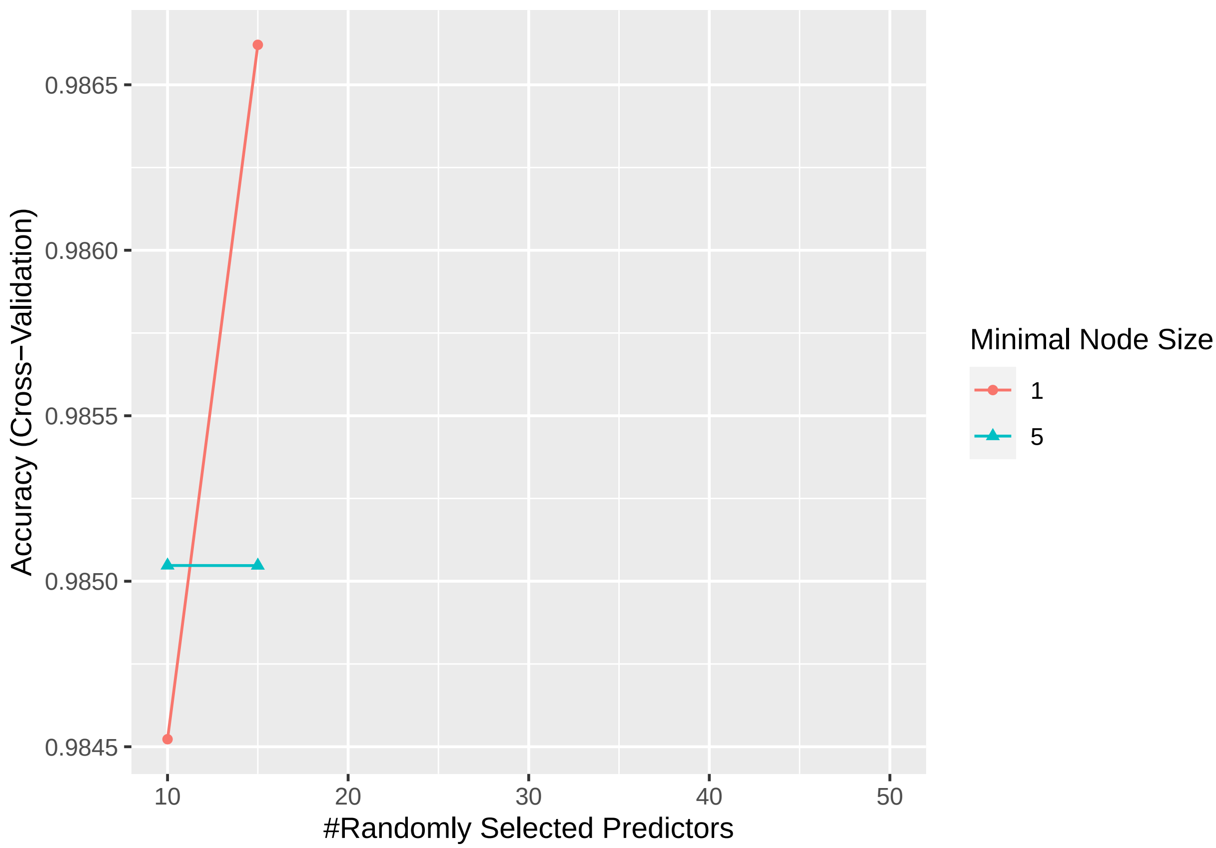
```
## Warning in nominalTrainWorkflow(x = x, y = y, wts = weights, info = trainInfo, :
## There were missing values in resampled performance measures.
```

```
## Warning in train.default(x, y, weights = w, ...): missing values found in
## aggregated results
```

```r
ggplot(train_rf)
```

```
## Warning: Removed 6 rows containing missing values (geom_point).
```

```
## Warning: Removed 6 row(s) containing missing values (geom_path).
```




```r
train_rf$bestTune
```

```
##   predFixed minNode
## 2        15       1
```

Best fit

```r
# best fit
grid <- expand.grid(minNode = train_rf$bestTune$minNode , predFixed = train_rf$bestTune$predFixed)

train_rf_best <-  train(Personal.Loan ~ .,
                   data=train_set,
                   method = "Rborist",
                   nTree = 1000,
                   trControl = control,
                   tuneGrid = grid
                   )

confusionMatrix(predict(train_rf_best, test_set), test_set$Personal.Loan)$overall["Accuracy"]
```

```
##  Accuracy
## 0.9874214
```

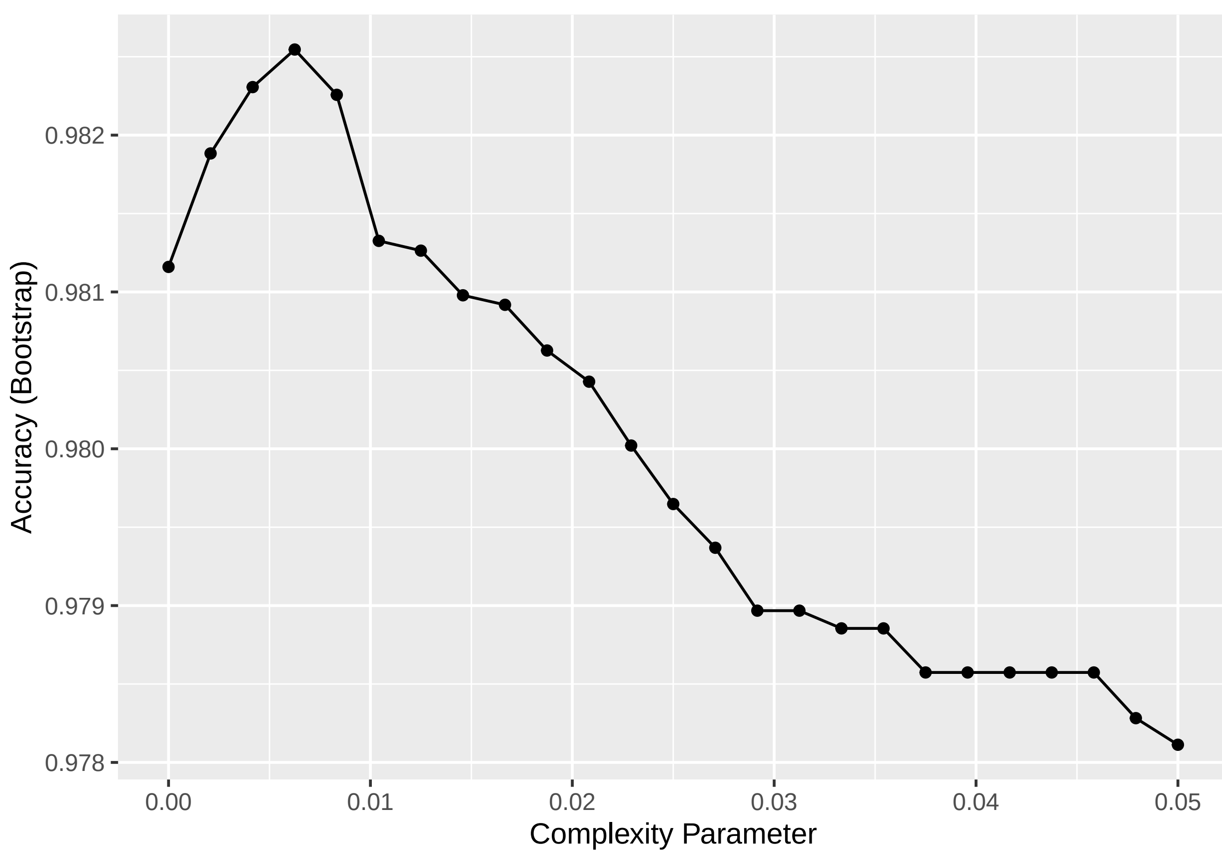
update model_result table

```r
model_result <- bind_rows (model_result ,tibble(method = "Rborist Grid tune", Accuracy =   confusionMat
  model_result
```

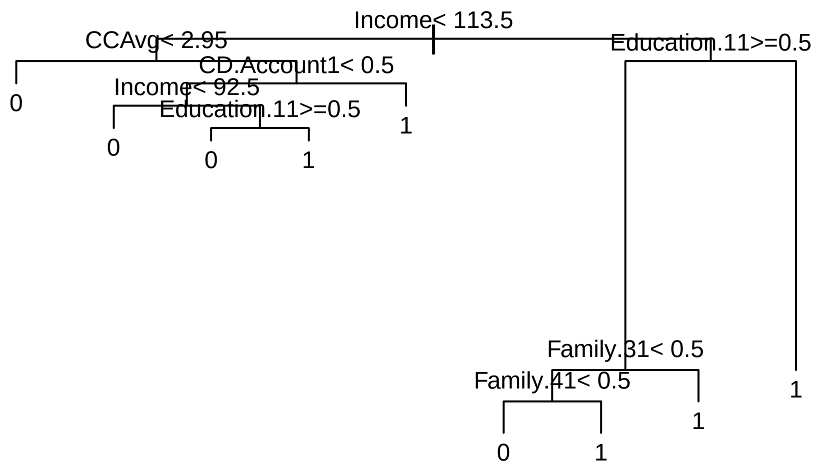
```
## # A tibble: 4 x 2
##   method                     Accuracy
##   <chr>                         <dbl>
## 1 Logistic Regression           0.957
## 2 K-nearest neighbors (kNN)     0.906
## 3 RandomForest General          0.987
## 4 Rborist Grid tune             0.987
```

Model 5:rpart

```r
train_rpart <- train(Personal.Loan ~ ., method = "rpart", tuneGrid = data.frame(cp = seq(0, 0.05, len =
  ggplot(train_rpart)
```




```r
  plot(train_rpart$finalModel, margin = 0.1)
  text(train_rpart$finalModel, cex = 0.75)
```




```
confusionMatrix(predict(train_rpart, test_set), test_set$Personal.Loan)$overall["Accuracy"]
```

```
##  Accuracy
## 0.9842767
```

update model_result table

```
model_result <- bind_rows (model_result ,tibble(method = "rpart Decision Tree", Accuracy =   confusionMa
  model_result
```

```
## # A tibble: 5 x 2
##   method                    Accuracy
##   <chr>                        <dbl>
## 1 Logistic Regression          0.957
## 2 K-nearest neighbors (kNN)    0.906
## 3 RandomForest General         0.987
## 4 Rborist Grid tune            0.987
## 5 rpart Decision Tree          0.984
```

Conclusion

To build a propensity model for Personal loan to help the bank to increase its retail customer base (asset) have started with Bank_data which has 5000 obs and 14 variables. As Bank is not going to provide Personal loans for those who are below 18 years and above 65 years, I have removed those customers. After removing , we have 4884 observations.Also, I removed 52 customers who have negative experience in the data set.After validating data, ID and ZIP code have been removed from data set as those are not going to contribute to build the model as predictors. I also introduced dummy variables for Education and Family as those have more than one levels. I have followed five modeling approaches to build my propensity model.

a) Logistic Regression Model
b) K-nearest neighbors (kNN)
c) RandomForest model
d) Rborist model
e) rpart model

Please refer the respective accuracy for each models.

```
model_result
```

```
## # A tibble: 5 x 2
##   method                     Accuracy
##   <chr>                         <dbl>
## 1 Logistic Regression          0.957
## 2 K-nearest neighbors (kNN)    0.906
## 3 RandomForest General         0.987
## 4 Rborist Grid tune            0.987
## 5 rpart Decision Tree          0.984
```

Basis on this Propensity model output, Bank can adopt the propensity model to create segmentation for campaign and provide Personal loan offers through specific set of campaigns (preferably customer journey campaign). This will help bank to increase its asset customer base and help to diversify its business towards Asset management domain.