CHURN AND MARKETING CAMPAIGN

Djenabou DIALLO

Import packages

```
library(readr)
library(ggplot2)
library(lattice)
library(caret)
library(rpart)
library(ROCR)
library(randomForest)
library(gbm)
library(foreach)
library(dplyr)
library(naniar)
library(corrplot)
library(tinytex)
tinytex::install_tinytex()
```

Load the three data sets

```
in13 <- read_csv("in13.csv")
data1 <- read_csv("data1.csv")
an13 <- read_csv("an13.csv")
```

Merge the three data sets in one big data base called 'data'

We renamed the variable Codcliente in the base 'in13' to codcliente in order to have the same name in all the data bases.

```
colnames(in13)[colnames(in13) == "CodCliente"] <- "codcliente"
data <- merge(merge(data1, an13, by = 'codcliente'), in13, by = 'codcliente')
```

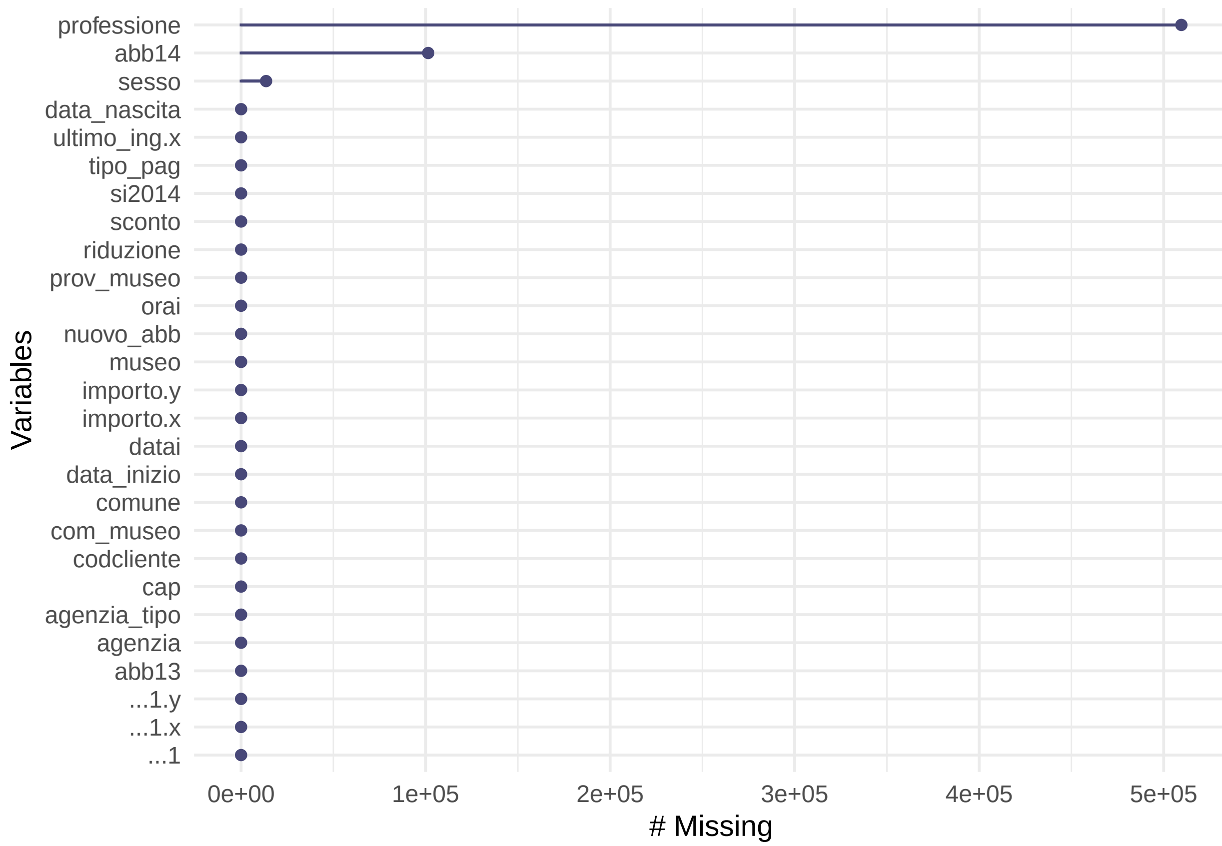
Handling missing values

We check for the number of missing values in each variable

```r
NA_count <- colSums(is.na(data))
print(sum(NA_count))
```

```
## [1] 624531
```

```r
gg_miss_var(data)
```



Replacing the missing values

sesso : we replaced the NA's with a new category 'I' for inconnu(unknown) abb14 : we replaced the NA's with the median date data_nascita : we only have 9 NA's that we replaced with the median date

```r
data$sesso[is.na(data$sesso)] <- 'I'
data$sesso <- factor(data$sesso, levels = c("F", "M", "I"))
median_date <- median(data$abb14, na.rm = TRUE)
data$abb14[is.na(data$abb14)] <- median_date
median_data <- median(data$data_nascita, na.rm = TRUE)
data$data_nascita[is.na(data$data_nascita)] <- median_date
```

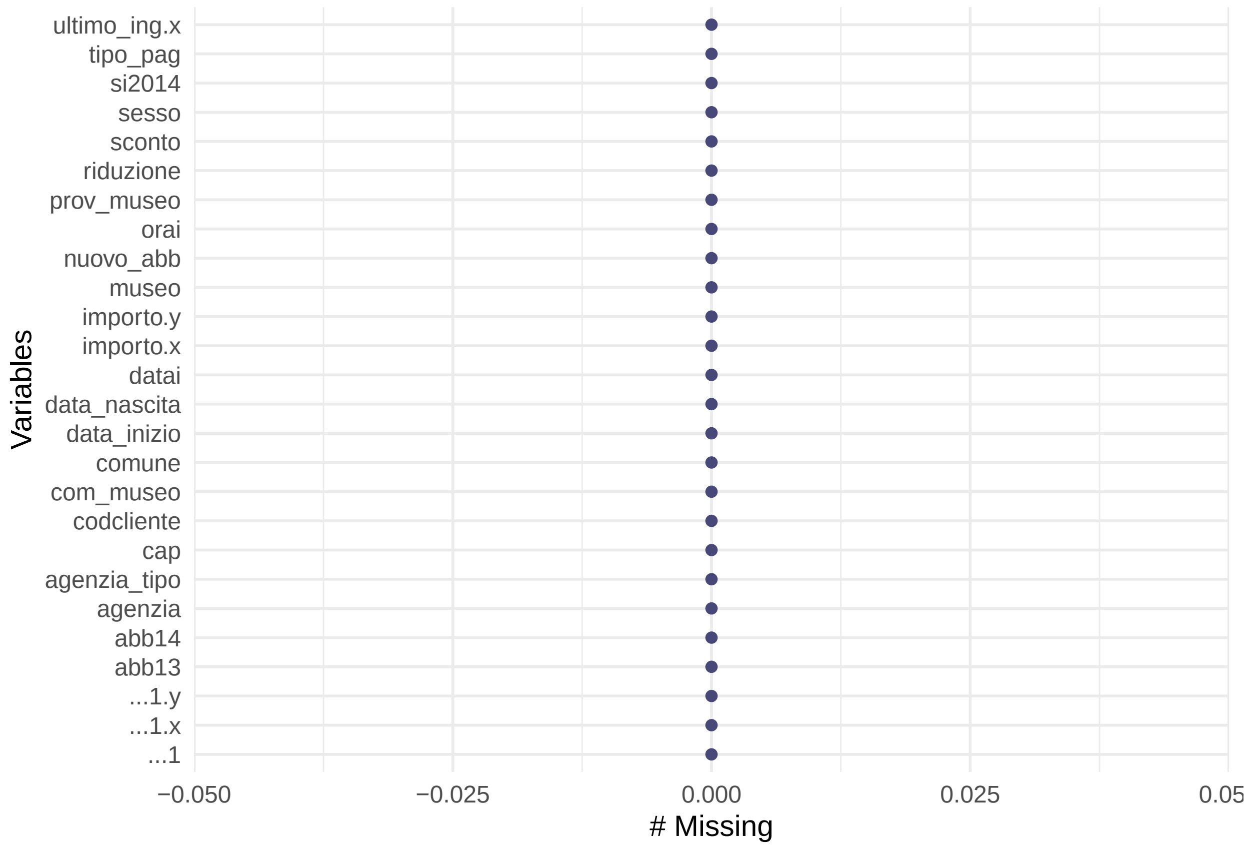
Deleted variables

professione : we deleted the column because it only contained missing values

```r
data <- data[, -which(names(data) == "professione")]
```

We verify the NA's in the data base

```r
gg_miss_var(data)
```



Variable processing

We transformed the following variables in dummies : sconto : we created two categories 0="NESSUNO SCONTO", 1="OTHER" tipo_pag : we created four categories 0="BANCOMAT/CARTA DI CREDITO" , 1="CONTANTI", 2="NESSUN PAGAMENTO", 3="ACQUISTO ONLINE" nuovo_abb : we created two categories 0= "NUOVO ABBONATO", 1="VECCHIO ABBONATO"

```r
data$sconto <- ifelse(data$sconto == "NESSUNO SCONTO", 0, 1)
data$sconto <- as.factor(data$sconto)

data$tipo_pag[data$tipo_pag == "BANCOMAT" | data$tipo_pag == "CARTA DI CREDITO"] <- 0
data$tipo_pag[data$tipo_pag == "CONTANTI"] <- 1
data$tipo_pag[data$tipo_pag == "NESSUN PAGAMENTO"] <- 2
data$tipo_pag[data$tipo_pag == "ACQUISTO ON-LINE"] <- 3
data$tipo_pag <- as.factor(data$tipo_pag)
```

```
data$nuovo_abb[data$nuovo_abb == "NUOVO ABBONATO"] <- 0
data$nuovo_abb[data$nuovo_abb==  "VECCHIO ABBONATO"] <- 1
data$nuovo_abb <- as.factor(data$nuovo_abb)
```

Target Distribution

```
table(data$si2014)/dim(data)[1]
```

```
##
##         0         1
## 0.1987633 0.8012367
```

```
data$si2014 <- as.factor(data$si2014)
```

Variable selection

We selected the variables that we used for the models and we stock them in a new data frame called 'df'

```
df <- data[c("si2014","sesso","abb13","data_nascita","sconto", "nuovo_abb", "tipo_pag",
             "importo.x")]
```

Create training and test data set

```
set.seed(4210)
perc_train_set <- 0.7
index <- createDataPartition(df$si2014,
                             p=perc_train_set,
                             list = FALSE)

train <- df[index,]
test <- df[-index,]
```

Decide which features we do need and define a formula

```
columns <- colnames(df)
target <- "si2014"
features <- columns[!columns %in% c(target)]

print(paste(paste(target," ~"), paste(features, collapse = " + ")))
```

```
## [1] "si2014  ~ sesso + abb13 + data_nascita + sconto + nuovo_abb + tipo_pag + importo.x"
```

```
formula <- as.formula(paste(paste(target," ~"), paste(features, collapse = " + ")))
```

Modelisation

CART Modeling

```
cart_model <- rpart(formula,
                    method="class", data=train,
                    parms = list(prior = c(.20,.80), split = "information"))

prediction_cart <- predict(cart_model,test,type = "class")

confusionMatrix(prediction_cart,test$si2014)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction      0      1
##          0   3215   2179
##          1  27170 120309
##
##                Accuracy : 0.808
##                  95% CI : (0.806, 0.81)
##     No Information Rate : 0.8012
##     P-Value [Acc > NIR] : 1.326e-11
##
##                   Kappa : 0.1274
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.10581
##             Specificity : 0.98221
##          Pos Pred Value : 0.59603
##          Neg Pred Value : 0.81577
##              Prevalence : 0.19876
##          Detection Rate : 0.02103
##    Detection Prevalence : 0.03528
##       Balanced Accuracy : 0.54401
##
##        'Positive' Class : 0
##
```

Summarize the results with AUROC

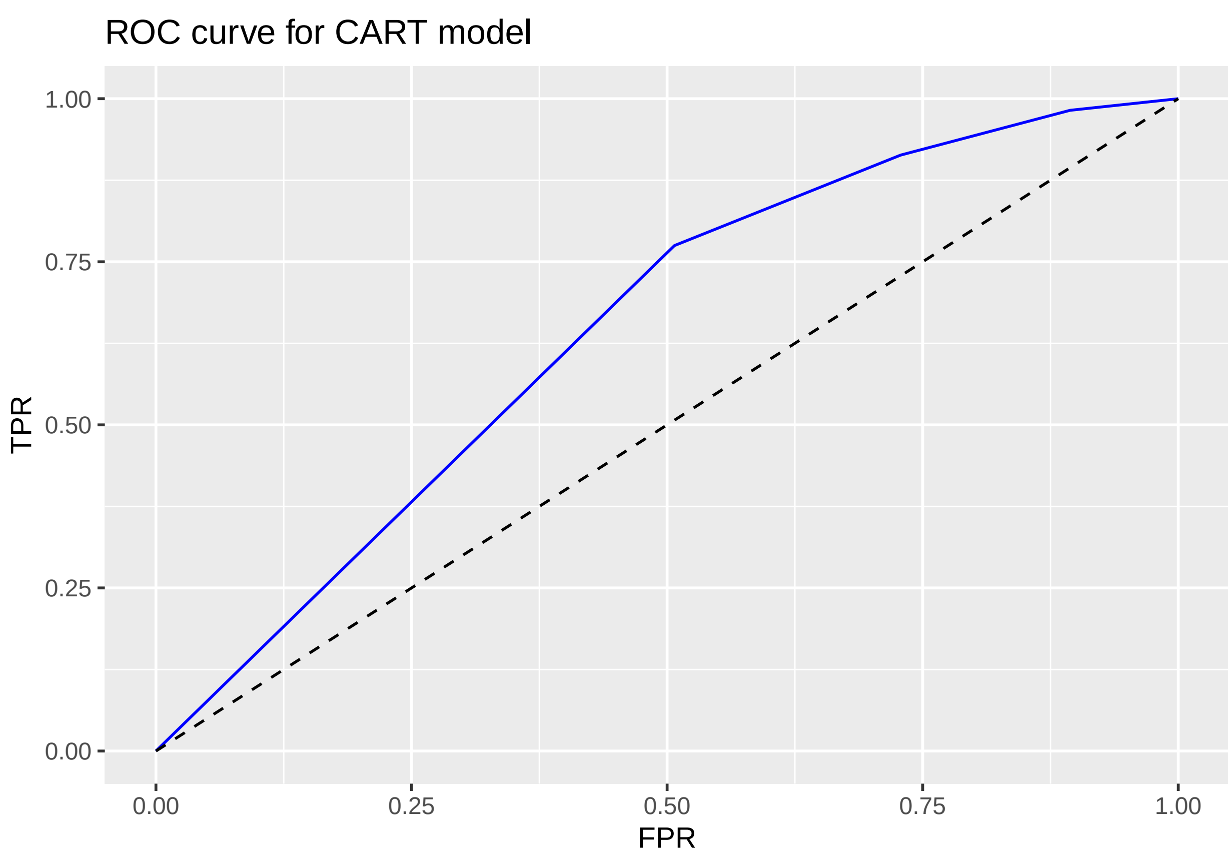
```
prediction_prob_cart <- predict(cart_model, test, type = 'prob')[,2]
roc_cart <- ROCR::prediction(predictions = prediction_prob_cart,
                              labels = test$si2014)
perf.roc_cart <- performance(roc_cart, measure = "tpr", x.measure = "fpr")
perf.auc_cart <- performance(roc_cart, measure = "auc")
ROC_df_cart <- data.frame(unlist(perf.roc_cart@x.values),
```

```r
                            unlist(perf.roc_cart@y.values))
colnames(ROC_df_cart) <- c("fpr","tpr")

print(paste("AUC (CART) -->",format((perf.auc_cart@y.values)[[1]]*100,digits = 4),"%"))
```

```
## [1] "AUC (CART) --> 64.53 %"
```

ROC plot

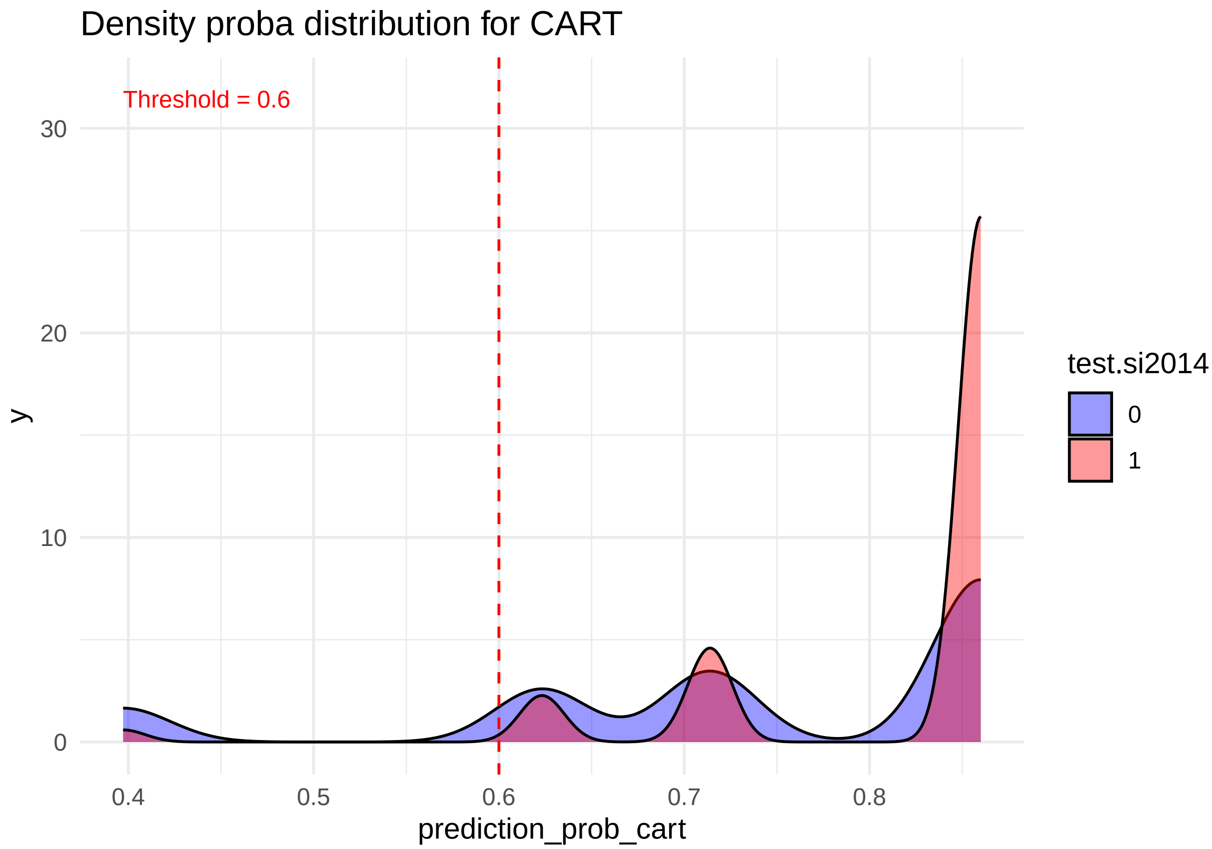
```r
xline <- seq(0,1,0.02)
yline <- seq(0,1,0.02)
xyline <- data.frame(xline,yline)

ggplot() +
  geom_line(data=ROC_df_cart, aes(x=fpr, y=tpr), color = "blue") +
  geom_line(data=xyline, aes(x=xline, y=yline), color='black',linetype = "dashed") +
  xlab("FPR") + ylab("TPR") +
  ggtitle("ROC curve for CART model")
```



Cheking the prediction

```r
plot <- data.frame(prediction_prob_cart, test$si2014)
threshold <- 0.6
ggplot(data=plot, aes(x=prediction_prob_cart, group=test.si2014, fill=test.si2014)) +
  geom_density(adjust=1.5, alpha=.4) +
```

```r
  geom_vline(xintercept=0.6, linetype="dashed", color="red")+
  ggtitle("Density proba distribution for CART")+
  scale_fill_manual(values = c("0" = "blue", "1" = "red")) +
  theme_minimal()+
  annotate("text", x = min(plot$prediction_prob_cart), y =  max(density(plot$prediction_prob_cart)$y),
           label = paste("Threshold =", threshold), color = "red", size = 3, hjust = 0, vjust = 1)
```



We chose the threshold = 0.6 because the two probability densities intersect approximately at the 0.6 point. This means that observations with a prediction probability greater than 0.6 are as likely to be in class 1 as in class 0. By choosing this threshold, we minimize the risk of classifying a class 0 observation as class 1.

RandomForest Modeling

```r
output <- capture.output({
rf_model <- randomForest(formula,
                         data = train,
                         do.trace = TRUE,
                         sampsize=c(1000),
                         mtry = 5,
                         ntree = 500,
                         importance = TRUE,
                         verbose = FALSE)
})
prediction_rf <- predict(rf_model,test,type = "class")
```

Confusion Matrix

```
confusionMatrix(prediction_rf,test$si2014)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction      0      1
##          0   4482   2518
##          1  25903 119970
##
##                Accuracy : 0.8141
##                  95% CI : (0.8121, 0.816)
##     No Information Rate : 0.8012
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.1786
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.14751
##             Specificity : 0.97944
##          Pos Pred Value : 0.64029
##          Neg Pred Value : 0.82243
##              Prevalence : 0.19876
##          Detection Rate : 0.02932
##    Detection Prevalence : 0.04579
##       Balanced Accuracy : 0.56347
##
##        'Positive' Class : 0
##
```

Summarize the results with AUROC

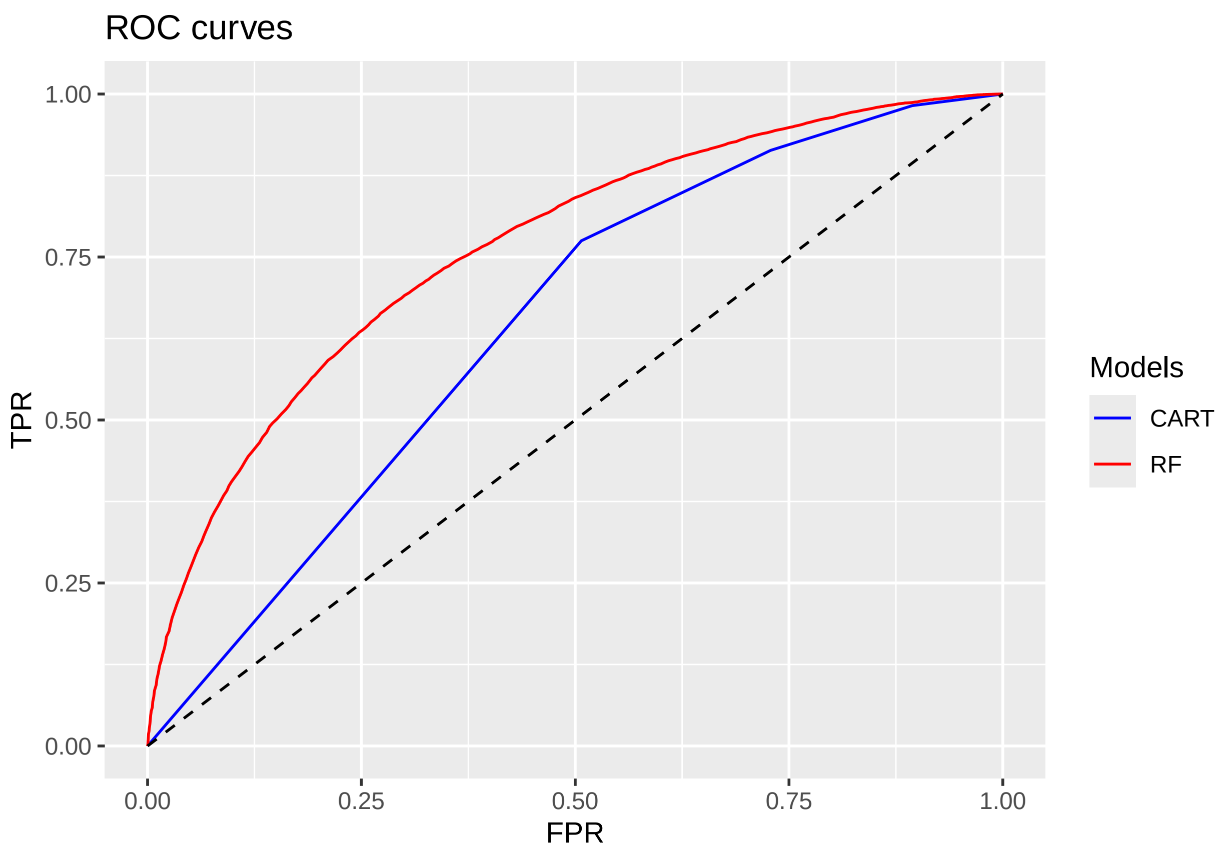
```
prediction_prob_rf <- predict(rf_model, test, type = 'prob')[,2]
roc_rf <- ROCR::prediction(predictions = prediction_prob_rf,
                           labels = test$si2014)
perf.roc_rf <- performance(roc_rf, measure = "tpr", x.measure = "fpr")
perf.auc_rf <- performance(roc_rf, measure = "auc")
ROC_df_rf <- data.frame(unlist(perf.roc_rf@x.values),
                        unlist(perf.roc_rf@y.values))
colnames(ROC_df_rf) <- c("fpr","tpr")

print(paste("AUC (RandomForest) -->",format((perf.auc_rf@y.values)[[1]]*100,digits = 4),
            "%"))
```

```
## [1] "AUC (RandomForest) --> 76.33 %"
```

ROC plot

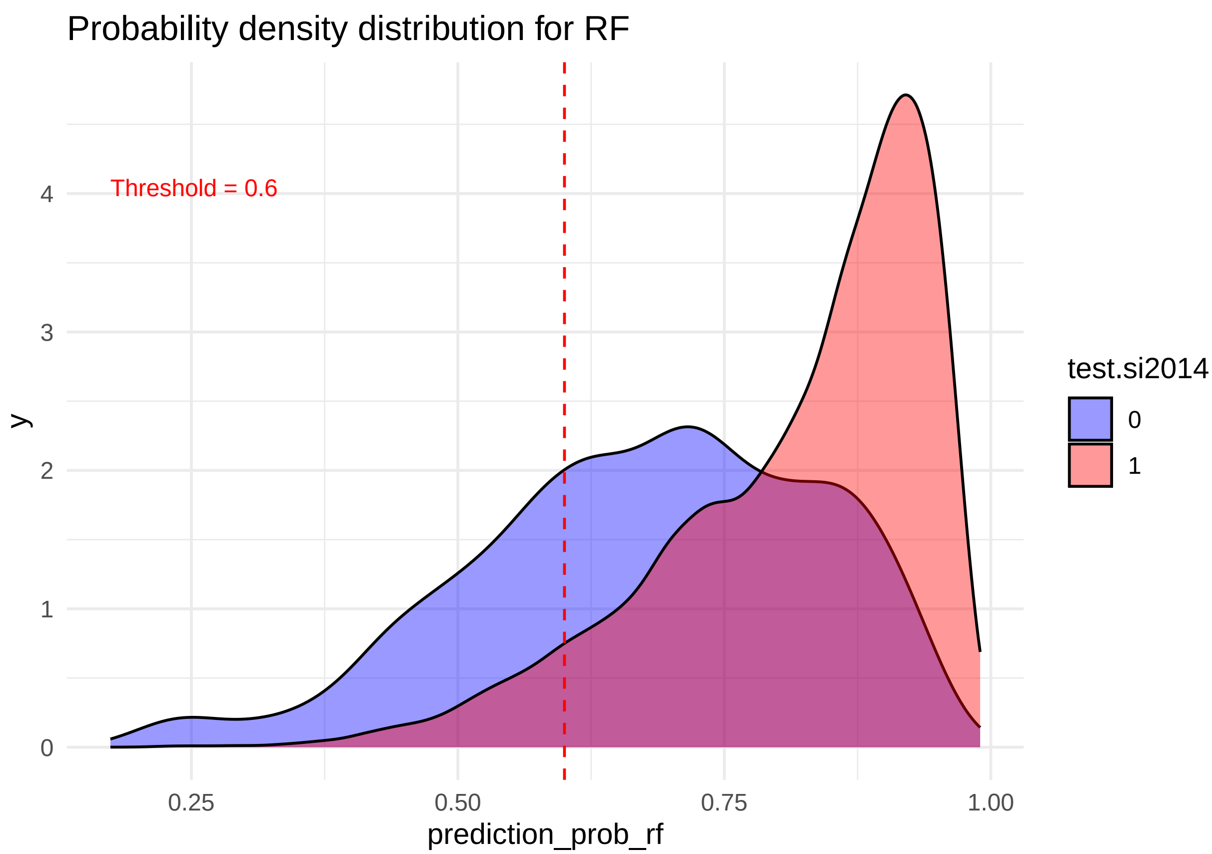
```
ggplot() +
  geom_line(data=ROC_df_cart, aes(x=fpr, y=tpr, color="CART")) +
```

```
  geom_line(data=ROC_df_rf, aes(x=fpr, y=tpr, color="RF")) +
  geom_line(data=xyline, aes(x=xline, y=yline), color='black',linetype = "dashed") +
  xlab("FPR") + ylab("TPR") +
  scale_colour_manual("Models",
                      values=c("CART"="blue","RF"="red")) +
  ggtitle("ROC curves")
```



Checking for prediction

```
plot_rf <- data.frame(prediction_prob_rf, test$si2014)
threshold <- 0.6

ggplot(data = plot_rf, aes(x = prediction_prob_rf, group = test.si2014, fill = test.si2014)) +
  geom_density(adjust = 1.5, alpha = 0.4) +
  geom_vline(xintercept = 0.6, linetype = "dashed", color = "red") +
  ggtitle("Probability density distribution for RF")+
  scale_fill_manual(values = c("0" = "blue", "1" = "red")) +
  theme_minimal() +
  annotate("text", x = min(plot_rf$prediction_prob_rf), y = max(density(plot_rf$prediction_prob_rf)$y),
           label = paste("Threshold =", threshold), color = "red", size = 3, hjust = 0, vjust = 1)
```



The optimal threshold is the point where the two probability densities cross. At this point, the two classes are the most difficult to distinguish. By choosing this threshold at 0.6 , we minimize the risk of classification error.

Logistic Regression Modeling

```
logit_model <- glm(si2014 ~ sesso + abb13 + data_nascita + sconto + nuovo_abb + tipo_pag + importo.x,
                   data = train,
                   family = binomial)
```

```
## Warning: glm.fit: fitted probabilities numerically 0 or 1 occurred
```

```
prediction_logit <- predict(logit_model, test, type = "response")

predicted_classes <- ifelse(prediction_logit > 0.5, 1, 0)
predicted_classes <- factor(predicted_classes, levels = levels(test$si2014))
confusionMatrix(predicted_classes, test$si2014)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 2819 2078
```

```
##          1  27566 120410
##
##                Accuracy : 0.8061
##                  95% CI : (0.8041, 0.8081)
##     No Information Rate : 0.8012
##     P-Value [Acc > NIR] : 9.686e-07
##
##                   Kappa : 0.1107
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.09278
##             Specificity : 0.98304
##          Pos Pred Value : 0.57566
##          Neg Pred Value : 0.81371
##              Prevalence : 0.19876
##          Detection Rate : 0.01844
##    Detection Prevalence : 0.03203
##       Balanced Accuracy : 0.53791
##
##        'Positive' Class : 0
##
```

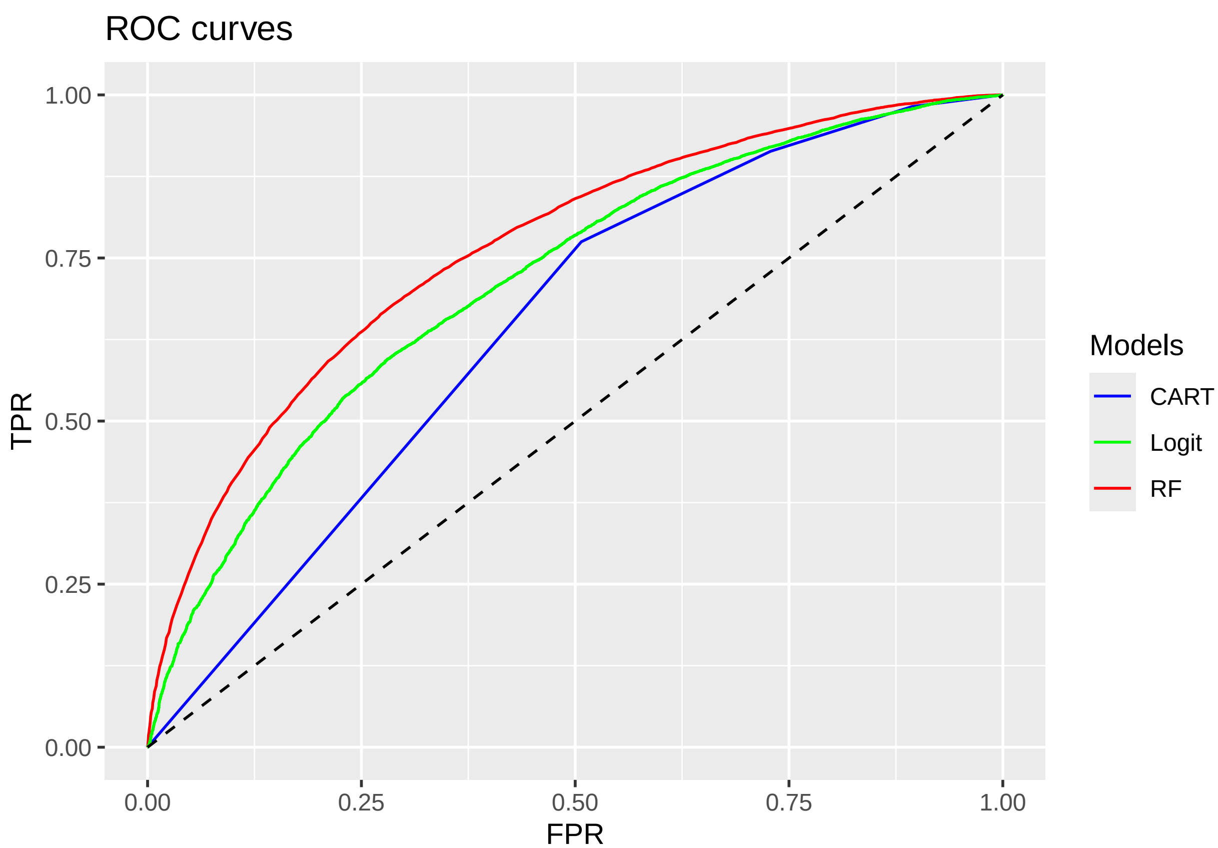
Summarize the results with AUROC

```
prediction_prob_logit <- predict(logit_model, test, type = 'response')
roc_logit <- ROCR::prediction(predictions = prediction_logit, labels = test$si2014)
perf.roc_logit <- performance(roc_logit, measure = "tpr", x.measure = "fpr")
perf.auc_logit <- performance(roc_logit, measure = "auc")
ROC_df_logit <- data.frame(unlist(perf.roc_logit@x.values), unlist(perf.roc_logit@y.values))
colnames(ROC_df_logit) <- c("fpr", "tpr")

print(paste("AUC (Logistic Regression) -->", format((perf.auc_logit@y.values)[[1]] * 100, digits = 4),
```

```
## [1] "AUC (Logistic Regression) --> 71.37 %"
```

ROC plot

```
ggplot() +
  geom_line(data=ROC_df_cart, aes(x=fpr, y=tpr, color="CART")) +
  geom_line(data=ROC_df_rf, aes(x=fpr, y=tpr, color="RF")) +
  geom_line(data=ROC_df_logit, aes(x=fpr, y=tpr, color="Logit")) +
  geom_line(data=xyline, aes(x=xline, y=yline), color='black',linetype = "dashed") +
  xlab("FPR") + ylab("TPR") +
  scale_colour_manual("Models",
                      values=c("CART"="blue","RF"="red","Logit"="green")) +
  ggtitle("ROC curves")
```

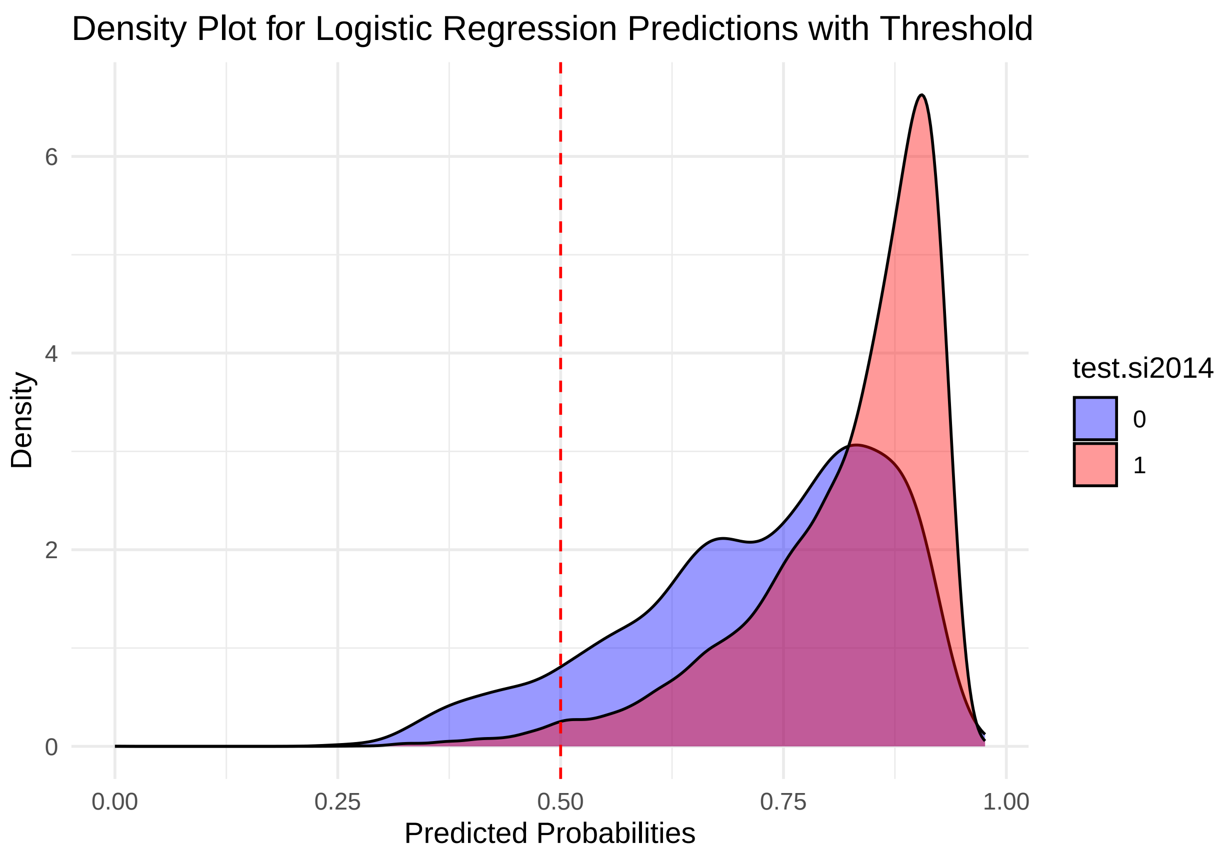


Checking for prediction

```
plot_logit <- data.frame(prediction_logit, test$si2014)
threshold <- 0.5  # You can adjust the threshold as needed
```

```
plot_logit <- data.frame(prediction_logit, test$si2014)
threshold <- 0.5

ggplot(data = plot_logit, aes(x = prediction_logit, group = test.si2014, fill = test.si2014)) +
  geom_density(adjust = 1.5, alpha = 0.4) +
  geom_vline(xintercept = threshold, linetype = "dashed", color = "red", size = 0.5) +
             xlab("Predicted Probabilities") +
             ylab("Density") +
             ggtitle("Density Plot for Logistic Regression Predictions with Threshold") +
             scale_fill_manual(values = c("0" = "blue", "1" = "red")) +

             theme_minimal()
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Here we have chosen 0.5 as the optimum threshold because it minimizes the classification error. This threshold is a compromise between minimizing false positives and minimizing false negatives.

PART 2

Profit curve

We create a new data frame to stock the test data that we will use

```
testdata <-test
cost<-0.2
fix_value <-10
```

We convert the variable 'si2014' as a numeric variable

```
testdata$si2014 <- as.numeric(as.character(testdata$si2014))
```

Calculation of the consumer value for each consumer.

For the following section for the variable 'si2014' we consider 1 as churner and 0 as non churner.

```
testdata$consumer_value <- ifelse(testdata$si2014 ==0,-testdata$importo.x+fix_value,fix_value)
```

Profit calculation

```
testdata$profit <- testdata$consumer_value-0.2
```

Adding predictions from RandomForest, Logit and CART models to a test data set

```
testdata$rf_prediction<-unlist(roc_rf@predictions)
testdata$logit_prediction<-unlist(roc_logit@predictions)
testdata$cart_prediction<-unlist(roc_cart@predictions)
```

Assignment of rankings based on the predictions of the RandomForest, Logistic Regression and CART models to the test data.

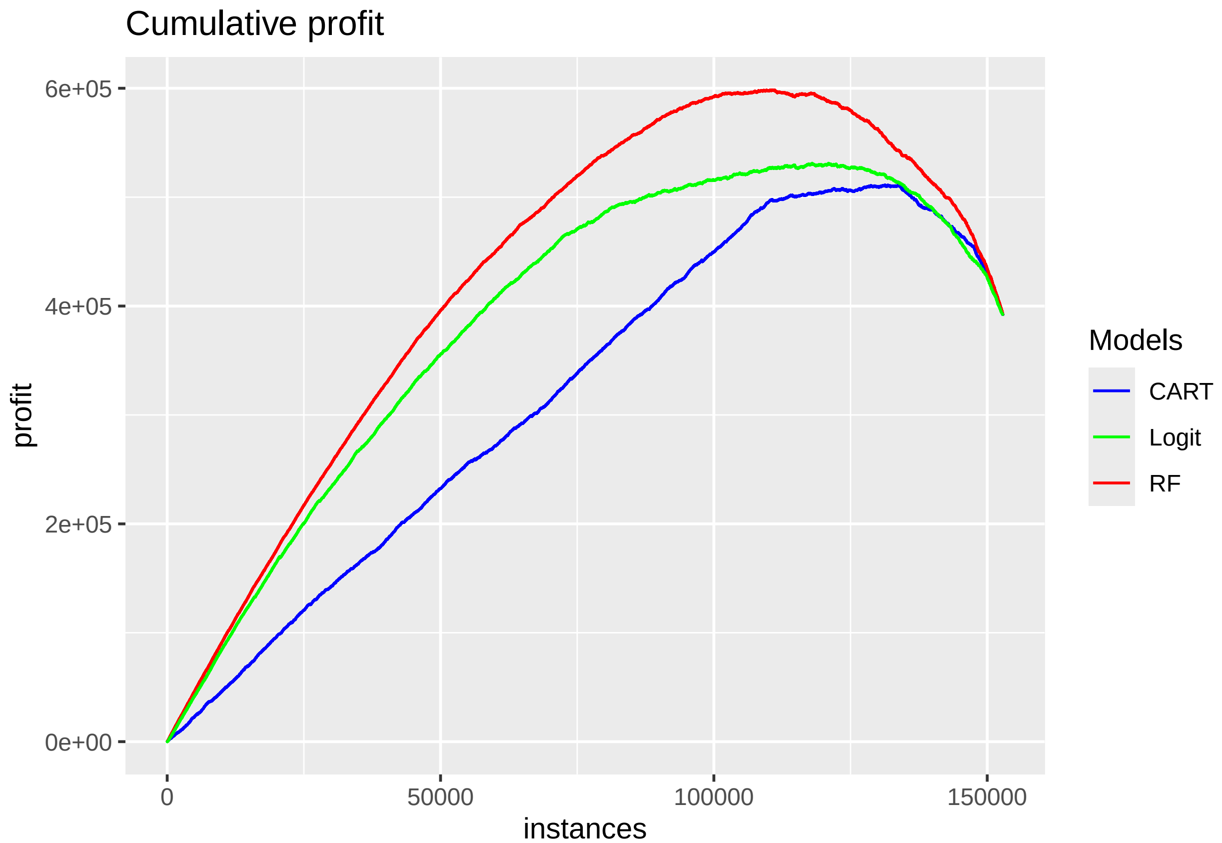
```
testdata<- testdata %>% mutate(rf_rank = rank(desc(rf_prediction),ties.method = "first"))
testdata<- testdata %>% mutate(logit_rank = rank(desc(logit_prediction), ties.method = "first"))
testdata<- testdata %>% mutate(cart_rank = rank(desc(cart_prediction),ties.method = "first"))
```

Calculation of cumulative profit

```
testdata <- transform(testdata[order(testdata$cart_rank, decreasing = F), ], cumsum_cart = ave(profit, 
testdata <- transform(testdata[order(testdata$rf_rank, decreasing = F), ], cumsum_rf = ave(profit, FUN=
testdata <- transform(testdata[order(testdata$logit_rank, decreasing = F), ], cumsum_logit = ave(profit
```

Drawing a cumulative profit curve for the models

```
ggplot() +
  geom_line(data=testdata, aes(x=cart_rank, y=cumsum_cart, color="CART")) +
  geom_line(data=testdata, aes(x=rf_rank, y=cumsum_rf, color="RF")) +
  geom_line(data=testdata, aes(x=logit_rank, y=cumsum_logit, color="Logit")) +
  xlab("instances") + ylab("profit") +
  scale_colour_manual("Models",
                      values=c("CART"="blue","RF"="red","Logit"="green")) +
  ggtitle("Cumulative profit")
```



The graph shows that the RandomForest model generates the highest cumulative profit, followed by the Logit model and the CART model. The RandomForest model starts to generate a positive cumulative profit earlier than the other two models. The CART model generates the smallest cumulative profit, but tends to converge with the other two models as it is trained on more data.