Success of Bank Telemarketing

29/05/23

Importing all the libraries

```
library(rpart)
library(rpart.plot)
library(randomForest)
library(caret)
```

Data Exploration

```
# Set working directory to file location
setwd("D:/JCU/Semester/2023 SP51 trisemester 2/MA3405 Statistical Data Mining
for Big Data/CAPSTONE PROJECT")

# Read 'bank-additional-full.csv' file
Data <- read.csv('bank-additional-full.csv', header = TRUE, sep = ";")

# Summary of the data
summary(Data)
```

```
##       age            job              marital           education
##  Min.   :17.00   Length:41188       Length:41188       Length:41188
##  1st Qu.:32.00   Class :character   Class :character   Class :character
##  Median :38.00   Mode  :character   Mode  :character   Mode  :character
##  Mean   :40.02
##  3rd Qu.:47.00
##  Max.   :98.00
##    default            housing              loan             contact
##  Length:41188       Length:41188       Length:41188       Length:41188
##  Class :character   Class :character   Class :character   Class :character
##  Mode  :character   Mode  :character   Mode  :character   Mode  :character
##
##
##
##     month           day_of_week           duration         campaign
##  Length:41188       Length:41188       Min.   :   0.0   Min.   : 1.000
##  Class :character   Class :character   1st Qu.: 102.0   1st Qu.: 1.000
##  Mode  :character   Mode  :character   Median : 180.0   Median : 2.000
##                                        Mean   : 258.3   Mean   : 2.568
##                                        3rd Qu.: 319.0   3rd Qu.: 3.000
##                                        Max.   :4918.0   Max.   :56.000
##      pdays          previous       poutcome          emp.var.rate
##  Min.   :  0.0   Min.   :0.000   Length:41188       Min.   :-3.40000
##  1st Qu.:999.0   1st Qu.:0.000   Class :character   1st Qu.:-1.80000
##  Median :999.0   Median :0.000   Mode  :character   Median : 1.10000
##  Mean   :962.5   Mean   :0.173                      Mean   : 0.08189
```

```
##  3rd Qu.:999.0   3rd Qu.:0.000                      3rd Qu.: 1.40000
##  Max.   :999.0   Max.   :7.000                      Max.   : 1.40000
##  cons.price.idx  cons.conf.idx     euribor3m      nr.employed
##  Min.   :92.20   Min.   :-50.8   Min.   :0.634   Min.   :4964
##  1st Qu.:93.08   1st Qu.:-42.7   1st Qu.:1.344   1st Qu.:5099
##  Median :93.75   Median :-41.8   Median :4.857   Median :5191
##  Mean   :93.58   Mean   :-40.5   Mean   :3.621   Mean   :5167
##  3rd Qu.:93.99   3rd Qu.:-36.4   3rd Qu.:4.961   3rd Qu.:5228
##  Max.   :94.77   Max.   :-26.9   Max.   :5.045   Max.   :5228
##       y
##  Length:41188
##  Class :character
##  Mode  :character
##
##
##

# Structure of the data
str(Data)

## 'data.frame':    41188 obs. of  21 variables:
##  $ age           : int  56 57 37 40 56 45 59 41 24 25 ...
##  $ job           : chr  "housemaid" "services" "services" "admin." ...
##  $ marital       : chr  "married" "married" "married" "married" ...
##  $ education     : chr  "basic.4y" "high.school" "high.school" "basic.6y"
...
##  $ default       : chr  "no" "unknown" "no" "no" ...
##  $ housing       : chr  "no" "no" "yes" "no" ...
##  $ loan          : chr  "no" "no" "no" "no" ...
##  $ contact       : chr  "telephone" "telephone" "telephone" "telephone"
...
##  $ month         : chr  "may" "may" "may" "may" ...
##  $ day_of_week   : chr  "mon" "mon" "mon" "mon" ...
##  $ duration      : int  261 149 226 151 307 198 139 217 380 50 ...
##  $ campaign      : int  1 1 1 1 1 1 1 1 1 1 ...
##  $ pdays         : int  999 999 999 999 999 999 999 999 999 999 ...
##  $ previous      : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ poutcome      : chr  "nonexistent" "nonexistent" "nonexistent"
"nonexistent" ...
##  $ emp.var.rate  : num  1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 1.1 ...
##  $ cons.price.idx: num  94 94 94 94 94 ...
##  $ cons.conf.idx : num  -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -36.4 -
36.4 -36.4 ...
##  $ euribor3m     : num  4.86 4.86 4.86 4.86 4.86 ...
##  $ nr.employed   : num  5191 5191 5191 5191 5191 ...
##  $ y             : chr  "no" "no" "no" "no" ...
```

Data Preprocessing

```
# Check for missing values.
missing_counts <- colSums(is.na(Data))
missing_counts
```

```
##            age            job        marital      education        default
##              0              0              0              0              0
##        housing           loan        contact          month    day_of_week
##              0              0              0              0              0
##       duration       campaign          pdays       previous       poutcome
##              0              0              0              0              0
##   emp.var.rate cons.price.idx  cons.conf.idx      euribor3m    nr.employed
##              0              0              0              0              0
##              y
##              0
```

```
# No missing values in data set.

# Convert categorical variables to factors
categorical_cols <- c("job", "marital", "education", "default", "housing",
"loan", "contact", "month", "day_of_week", "poutcome", "y")
Data[categorical_cols] <- lapply(Data[categorical_cols], as.factor)

# Split the data into training and testing sets (80% for training, 20% for
testing)
set.seed(123)
train_index <- sample(nrow(Data), 0.8 * nrow(Data))
train_data <- Data[train_index, ]
test_data <- Data[-train_index, ]
dim(train_data)
```

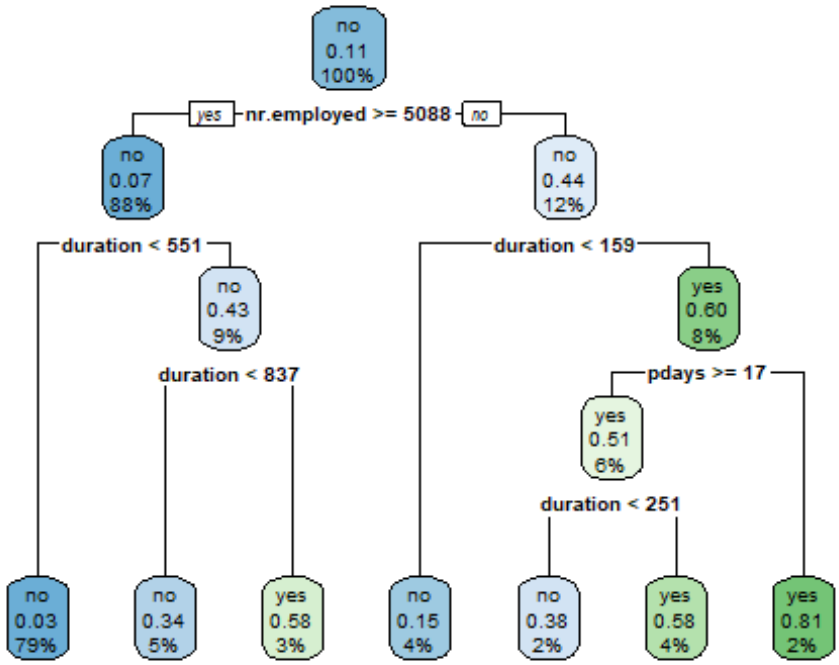
```
## [1] 32950    21
```

```
dim(test_data)
```

```
## [1] 8238   21
```

Decision Tree

```
# Build the decision tree model
tree_model <- rpart(y ~ ., data = train_data, method = "class")
tree_predictions <- predict(tree_model, newdata = test_data, type = "class")
# Visualize the decision tree
rpart.plot(tree_model)
```

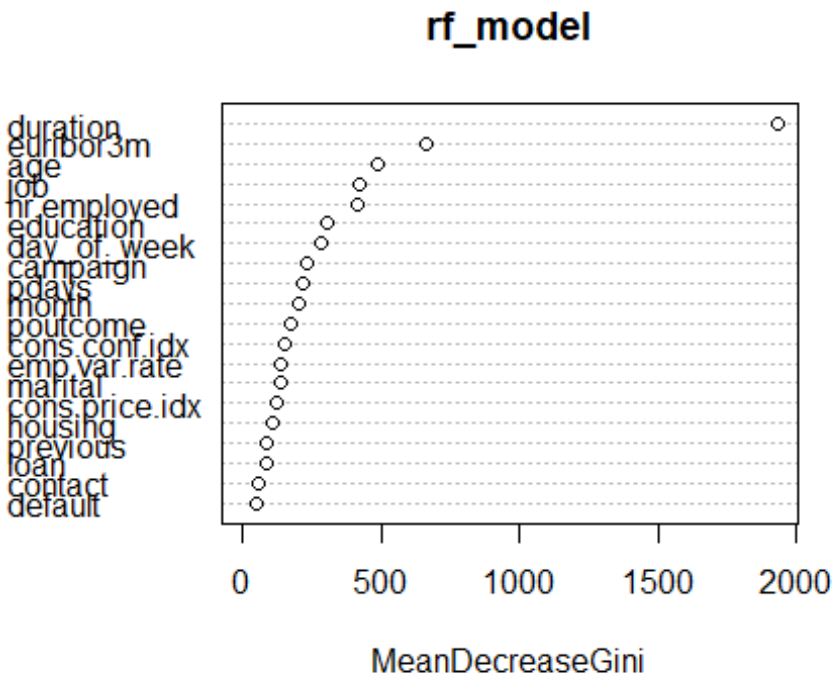


Logistic Regression

```
# Build the logistic regression model
logit_model <- glm(y ~ ., data = train_data, family = "binomial")

# Make predictions using the logistic regression model
logit_predictions <- predict(logit_model, newdata = test_data, type =
"response")
logit_predictions <- ifelse(logit_predictions > 0.5, "yes", "no")

# Convert predicted variable to factor with same levels as actual variable
logit_predictions <- factor(logit_predictions, levels = levels(test_data$y))
```

Random Forest

```
# Build the Random Forest
rf_model <- randomForest(y ~ ., data = train_data)
rf_predictions <- predict(rf_model, newdata = test_data, type = "class")

# Variable Importance Plot for Random Forest
varImpPlot(rf_model)
```



Model Evaluation

```
# Evaluate the performance of the models
tree_confusion <- confusionMatrix(tree_predictions, test_data$y)
logit_confusion <- confusionMatrix(logit_predictions, test_data$y)
rf_confusion <- confusionMatrix(rf_predictions, test_data$y)

# Display Confusion Matrices
print(tree_confusion)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   no  yes
##        no  7115  398
##        yes  248  477
##
##                Accuracy : 0.9216
##                  95% CI : (0.9156, 0.9273)
##     No Information Rate : 0.8938
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5532
##
##  Mcnemar's Test P-Value : 4.564e-09
##
##             Sensitivity : 0.9663
```

```
##             Specificity : 0.5451
##          Pos Pred Value : 0.9470
##          Neg Pred Value : 0.6579
##              Prevalence : 0.8938
##          Detection Rate : 0.8637
##    Detection Prevalence : 0.9120
##       Balanced Accuracy : 0.7557
##
##        'Positive' Class : no
##

print(logit_confusion)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction   no  yes
##        no  7168  489
##        yes  195  386
##
##                Accuracy : 0.917
##                  95% CI : (0.9108, 0.9228)
##     No Information Rate : 0.8938
##     P-Value [Acc > NIR] : 9.052e-13
##
##                   Kappa : 0.4867
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9735
##             Specificity : 0.4411
##          Pos Pred Value : 0.9361
##          Neg Pred Value : 0.6644
##              Prevalence : 0.8938
##          Detection Rate : 0.8701
##    Detection Prevalence : 0.9295
##       Balanced Accuracy : 0.7073
##
##        'Positive' Class : no
##

print(rf_confusion)

## Confusion Matrix and Statistics
##
##           Reference
## Prediction   no  yes
##        no  7121  403
##        yes  242  472
##
##                Accuracy : 0.9217
```

```
##                  95% CI : (0.9157, 0.9274)
##     No Information Rate : 0.8938
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5512
##
##  Mcnemar's Test P-Value : 2.977e-10
##
##             Sensitivity : 0.9671
##             Specificity : 0.5394
##          Pos Pred Value : 0.9464
##          Neg Pred Value : 0.6611
##              Prevalence : 0.8938
##          Detection Rate : 0.8644
##    Detection Prevalence : 0.9133
##       Balanced Accuracy : 0.7533
##
##        'Positive' Class : no
##
```