Assign3-2\_Shah

2024-01-25

options(repos = c(CRAN = "https://cran.rstudio.com/"))  
library(readr)  
library(dplyr)

##   
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':  
##   
## filter, lag

## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

bank\_data <- read\_csv("/Users/manishashah/Desktop/UniversalBank\_Ensemble(1).csv")

## Rows: 5000 Columns: 9

## ── Column specification ────────────────────────────────────────────────────────  
## Delimiter: ","  
## dbl (9): Age, Experience, Income, Family, Education, Mortgage, Personal Loan...  
##   
## ℹ Use `spec()` to retrieve the full column specification for this data.  
## ℹ Specify the column types or set `show\_col\_types = FALSE` to quiet this message.

# data preprocessing  
bank\_data <- bank\_data %>%  
 rename(Personal\_Loan = `Personal Loan`,   
 Securities\_Account = `Securities Account`) %>%  
 mutate\_at(vars(Personal\_Loan, Securities\_Account, CreditCard), as.factor)

# 1. Decision Tree  
library(rpart)  
fit\_tree <- rpart(Personal\_Loan ~ ., data = bank\_data, method = "class")  
fit\_tree

## n= 5000   
##   
## node), split, n, loss, yval, (yprob)  
## \* denotes terminal node  
##   
## 1) root 5000 480 0 (0.90400000 0.09600000)   
## 2) Income< 113.5 4021 84 0 (0.97910967 0.02089033) \*  
## 3) Income>=113.5 979 396 0 (0.59550562 0.40449438)   
## 6) Education< 1.5 635 69 0 (0.89133858 0.10866142)   
## 12) Family< 2.5 566 0 0 (1.00000000 0.00000000) \*  
## 13) Family>=2.5 69 0 1 (0.00000000 1.00000000) \*  
## 7) Education>=1.5 344 17 1 (0.04941860 0.95058140) \*

# 2. Bagging  
library(ipred)  
fit\_bagging <- bagging(Personal\_Loan ~ ., data = bank\_data, nbagg = 100, coob = TRUE)  
fit\_bagging

##   
## Bagging classification trees with 100 bootstrap replications   
##   
## Call: bagging.data.frame(formula = Personal\_Loan ~ ., data = bank\_data,   
## nbagg = 100, coob = TRUE)  
##   
## Out-of-bag estimate of misclassification error: 0.0184

# 3. Boosting  
library(gbm)

## Loaded gbm 2.1.9

## This version of gbm is no longer under development. Consider transitioning to gbm3, https://github.com/gbm-developers/gbm3

fit\_boosting <- gbm(Personal\_Loan ~ ., data = bank\_data, distribution = "bernoulli", n.trees = 100, verbose = FALSE)  
fit\_boosting

## gbm(formula = Personal\_Loan ~ ., distribution = "bernoulli",   
## data = bank\_data, n.trees = 100, verbose = FALSE)  
## A gradient boosted model with bernoulli loss function.  
## 100 iterations were performed.  
## There were 8 predictors of which 0 had non-zero influence.

# 4. Random Forest  
library(randomForest)

## randomForest 4.7-1.1

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

##   
## Attaching package: 'randomForest'

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

fit\_rf <- randomForest(Personal\_Loan ~ ., data = bank\_data)  
fit\_rf

##   
## Call:  
## randomForest(formula = Personal\_Loan ~ ., data = bank\_data)   
## Type of random forest: classification  
## Number of trees: 500  
## No. of variables tried at each split: 2  
##   
## OOB estimate of error rate: 1.78%  
## Confusion matrix:  
## 0 1 class.error  
## 0 4515 5 0.001106195  
## 1 84 396 0.175000000

# now create caroline's data  
caroline <- data.frame(Age = 32, Experience = 6,Income = 200,Family = 3,Education = 3, Mortgage = 0, Securities\_Account = factor(1, levels = levels(bank\_data$Securities\_Account)), CreditCard = factor(1, levels = levels(bank\_data$CreditCard)))  
  
# let's do a predictions  
predict\_tree <- predict(fit\_tree, newdata = caroline, type = "class")  
predict\_bagging <- predict(fit\_bagging, newdata = caroline, type = "class")  
predict\_boosting <- predict(fit\_boosting, newdata = caroline, n.trees = 100, type = "response")  
predict\_rf <- predict(fit\_rf, newdata = caroline, type = "class")  
  
# Print the prediction  
print(list(Decision\_Tree = predict\_tree, Bagging = predict\_bagging, Boosting = predict\_boosting, Random\_Forest = predict\_rf))

## $Decision\_Tree  
## 1   
## 1   
## Levels: 0 1  
##   
## $Bagging  
## [1] 1  
## Levels: 0 1  
##   
## $Boosting  
## [1] NaN  
##   
## $Random\_Forest  
## 1   
## 1   
## Levels: 0 1

# Caroline’s profile with four different models - Decision Tree, Bagging, Random Forest, and Boosting - each model gives the same prediction: Caroline is likely to accept a personal loan. This unanimous prediction across different models suggests that Caroline’s details match well with those of customers who usually accept loans from Universal Bank. This consistent prediction across various models makes the result more reliable. For the bank, this information is very useful. They can focus their loan marketing on customers like Caroline, likely improving their chances of success. This example shows how combining different models in prediction, known as ensemble modeling, is useful in banking for making better marketing decisions.