LFS\_Algorithms

Md. Azad Pervez

2023-11-05

# Define the file path and Read CSV

full\_dataset\_path <- "C:/820WF/LFS\_2017\_2023/lfs\_17\_23.csv"  
LFS\_data <- read.csv(full\_dataset\_path)

# Select a smaller subset of data from 2021

# Select only the rows where SURVYEAR is greater than or equal to 2021  
LFS\_Selected\_2021\_onwards <- LFS\_data[LFS\_data$SURVYEAR >= 2021, c("LFSSTAT", "PROV", "AGE\_12", "SEX", "MARSTAT", "EDUC", "IMMIG")]

#Dependent and Independent Variables

##Dependent Variable: LFSSTAT  
  
# LFSSTAT (Labour force status) contains the following:  
#1 Employed, at work  
#2 Employed, absent from work  
#3 Unemployed  
#4 Not in labour force  
  
#For the purpose of data analysis the dependent variable is combined into two levels  
# 1 = 1 = Active Employment  
# 2 = 2,3,4 = Inactive employment  
  
## Independent variables: Six variables taken as independent variables:  
  
# prov = Province  
# age\_12 = Five-year age group of respondent  
# sex = Sex of respondent  
# marstat = Marital status of respondent  
# educ = Highest educational attainment  
# immig = Immigrant status  
  
  
#Algorithms  
  
# The algorithms that are used are Decision Tree, Random Forest and Naive Bayes  
# Cross-validation is used to assess the performance of the models  
# Accuracy, Precision and Recalls are used to see how the models performed

# Load the Libraries

library(randomForest)

## Warning: package 'randomForest' was built under R version 4.3.1

## randomForest 4.7-1.1

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

library(caret)

## Warning: package 'caret' was built under R version 4.3.1

## Loading required package: ggplot2

##   
## Attaching package: 'ggplot2'

## The following object is masked from 'package:randomForest':  
##   
## margin

## Loading required package: lattice

library(rpart)  
library(e1071)

## Warning: package 'e1071' was built under R version 4.3.1

# Combine the four LFSSTAT categories into two levels  
LFS\_Selected\_2021\_onwards$LFSSTAT <- ifelse(LFS\_Selected\_2021\_onwards$LFSSTAT == 1, 1, 2)

# Algorithm 1: Decision Tree

# Prepare the Data  
LFS\_Selected\_2021\_onwards$LFSSTAT <- as.factor(LFS\_Selected\_2021\_onwards$LFSSTAT)  
  
# cross-validation settings  
control\_ndt <- trainControl(method = "cv", number = 5) # 5-fold cross-validation  
  
# Create a smaller training subset  
set.seed(123)  
  
sample\_size <- floor(0.1 \* nrow(LFS\_Selected\_2021\_onwards)) # Use 10% of the data  
train\_index\_ndt <- sample(1:nrow(LFS\_Selected\_2021\_onwards), sample\_size)  
smaller\_training\_data\_ndt <- LFS\_Selected\_2021\_onwards[train\_index\_ndt, ]  
testing\_data\_ndt <- LFS\_Selected\_2021\_onwards[-train\_index\_ndt, ]  
  
# Train a Decision Tree Model with cross-validation  
tree\_model\_ndt <- train(LFSSTAT ~ ., data = smaller\_training\_data\_ndt, method = "rpart", trControl = control\_ndt)  
  
# Make Predictions with class labels  
tree\_predictions\_class <- predict(tree\_model\_ndt, testing\_data\_ndt)  
  
# confusion matrix  
confusion\_tree <- confusionMatrix(tree\_predictions\_class, testing\_data\_ndt$LFSSTAT)  
  
# Accuracy, precision and recall  
accuracy\_tree <- confusion\_tree$overall['Accuracy']  
precision\_tree <- confusion\_tree$byClass['Pos Pred Value']  
recall\_tree <- confusion\_tree$byClass['Sensitivity']  
  
# Print the confusion matrix and results  
print(confusion\_tree)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 1 2  
## 1 1345635 563124  
## 2 211556 828258  
##   
## Accuracy : 0.7373   
## 95% CI : (0.7368, 0.7378)  
## No Information Rate : 0.5281   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.4657   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.8641   
## Specificity : 0.5953   
## Pos Pred Value : 0.7050   
## Neg Pred Value : 0.7965   
## Prevalence : 0.5281   
## Detection Rate : 0.4564   
## Detection Prevalence : 0.6474   
## Balanced Accuracy : 0.7297   
##   
## 'Positive' Class : 1   
##

cat("Accuracy:", accuracy\_tree, "\n")

## Accuracy: 0.7372695

cat("Precision:", precision\_tree, "\n")

## Precision: 0.704979

cat("Recall:", recall\_tree, "\n")

## Recall: 0.8641425

# Algorithm 2: Random Forest

# Split the dataset into a training set and a test set  
set.seed(123)  
trainIndex <- createDataPartition(LFS\_Selected\_2021\_onwards$LFSSTAT, p = 0.7, list = FALSE)  
  
# the data set is large so the dataset is shrinked into a smaller one  
smaller\_training\_data <- LFS\_Selected\_2021\_onwards[sample(nrow(LFS\_Selected\_2021\_onwards), nrow(LFS\_Selected\_2021\_onwards) / 10), ]  
  
# the testing data  
testing\_data <- LFS\_Selected\_2021\_onwards[-trainIndex, ]  
  
# cross-validation  
control\_rf <- trainControl(method = "cv", number = 5) # 5-fold cross-validation  
  
# Train the Random Forest model using cross-validation  
model\_rf <- train(factor(LFSSTAT) ~ ., data = smaller\_training\_data, method = "rf", trControl = control\_rf, ntree = 25)  
  
# Make predictions using the model  
predictions\_rf <- predict(model\_rf, testing\_data)  
  
# Ensure both are factors with the same levels  
predictions\_rf <- as.factor(predictions\_rf)  
testing\_data$LFSSTAT <- as.factor(testing\_data$LFSSTAT)  
  
# Confusion matrix  
confusion\_rf <- confusionMatrix(predictions\_rf, testing\_data$LFSSTAT)  
  
# Calculate accuracy, precision, and recall  
accuracy\_rf <- confusion\_rf$overall['Accuracy']  
precision\_rf <- confusion\_rf$byClass['Pos Pred Value']  
recall\_rf <- confusion\_rf$byClass['Sensitivity']  
  
# Print the confusion matrix and results  
print(confusion\_rf)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 1 2  
## 1 441958 166901  
## 2 77001 296997  
##   
## Accuracy : 0.7518   
## 95% CI : (0.751, 0.7527)  
## No Information Rate : 0.528   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.497   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.8516   
## Specificity : 0.6402   
## Pos Pred Value : 0.7259   
## Neg Pred Value : 0.7941   
## Prevalence : 0.5280   
## Detection Rate : 0.4497   
## Detection Prevalence : 0.6195   
## Balanced Accuracy : 0.7459   
##   
## 'Positive' Class : 1   
##

cat("Accuracy:", accuracy\_rf, "\n")

## Accuracy: 0.7518439

cat("Precision:", precision\_rf, "\n")

## Precision: 0.7258791

cat("Recall:", recall\_rf, "\n")

## Recall: 0.8516241

## Algorithm 3: Naive Bayes

LFS\_Selected\_2021\_onwards$LFSSTAT <- as.factor(LFS\_Selected\_2021\_onwards$LFSSTAT)  
  
# cross-validation  
control\_nb <- trainControl(method = "cv", number = 5) # 5-fold cross-validation  
  
# Create a smaller training subset  
set.seed(123)  
sample\_size <- floor(0.1 \* nrow(LFS\_Selected\_2021\_onwards)) # Use 10% of the data  
train\_index\_nb <- sample(1:nrow(LFS\_Selected\_2021\_onwards), sample\_size)  
smaller\_training\_data\_nb <- LFS\_Selected\_2021\_onwards[train\_index\_nb, ]  
testing\_data\_nb <- LFS\_Selected\_2021\_onwards[-train\_index\_nb, ]  
  
# Train a Naive Bayes Model with cross-validation  
nb\_model <- train(LFSSTAT ~ ., data = smaller\_training\_data\_nb, method = "naive\_bayes", trControl = control\_nb)  
  
# Make Predictions  
nb\_predictions <- predict(nb\_model, testing\_data\_nb)  
  
# confusion matrix  
confusion\_nb <- confusionMatrix(nb\_predictions, testing\_data\_nb$LFSSTAT)  
  
# Calculate accuracy, precision, and recall  
accuracy\_nb <- confusion\_nb$overall['Accuracy']  
precision\_nb <- confusion\_nb$byClass['Pos Pred Value']  
recall\_nb <- confusion\_nb$byClass['Sensitivity']  
  
# Print the confusion matrix and results  
print(confusion\_nb)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 1 2  
## 1 1344864 548450  
## 2 212327 842932  
##   
## Accuracy : 0.742   
## 95% CI : (0.7415, 0.7425)  
## No Information Rate : 0.5281   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.4756   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.8636   
## Specificity : 0.6058   
## Pos Pred Value : 0.7103   
## Neg Pred Value : 0.7988   
## Prevalence : 0.5281   
## Detection Rate : 0.4561   
## Detection Prevalence : 0.6421   
## Balanced Accuracy : 0.7347   
##   
## 'Positive' Class : 1   
##

cat("Accuracy:", accuracy\_nb, "\n")

## Accuracy: 0.7419847

cat("Precision:", precision\_nb, "\n")

## Precision: 0.7103227

cat("Recall:", recall\_nb, "\n")

## Recall: 0.8636474