LFS\_Algorithms

Azad

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 and Combine Categories

# Select only the relevant variables  
LFS\_Selected <- LFS\_data[LFS\_data$SURVYEAR >= 2023, 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

# Combine the four LFSSTAT categories into two levels

LFS\_Selected$LFSSTAT <- ifelse(LFS\_Selected$LFSSTAT == 1, 1, 2)

# 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

# Algorithm 1: Decision Tree

# Prepare the Data  
LFS\_Selected$LFSSTAT <- as.factor(LFS\_Selected$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)) # Use 10% of the data  
train\_index\_ndt <- sample(1:nrow(LFS\_Selected), sample\_size)  
smaller\_training\_data\_ndt <- LFS\_Selected[train\_index\_ndt, ]  
testing\_data\_ndt <- LFS\_Selected[-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 383773 154626  
## 2 60111 231855  
##   
## Accuracy : 0.7414   
## 95% CI : (0.7405, 0.7423)  
## No Information Rate : 0.5346   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.472   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.8646   
## Specificity : 0.5999   
## Pos Pred Value : 0.7128   
## Neg Pred Value : 0.7941   
## Prevalence : 0.5346   
## Detection Rate : 0.4622   
## Detection Prevalence : 0.6484   
## Balanced Accuracy : 0.7322   
##   
## 'Positive' Class : 1   
##

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

## Accuracy: 0.7413944

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

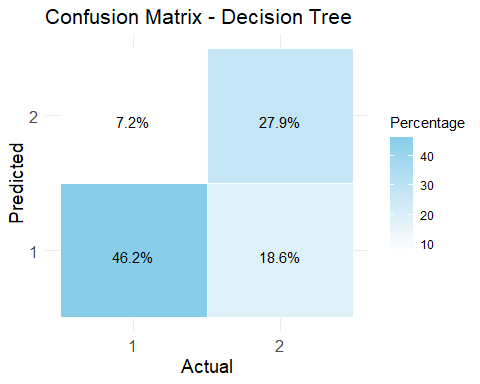
## Precision: 0.7128041

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

## Recall: 0.8645795

#Visualize Decision Tree Confusion Matrix

# Convert the confusion matrix to a data frame for plotting  
conf\_matrix\_tree\_df <- as.data.frame(as.table(confusion\_tree))  
  
# Calculate percentages for a more informative plot  
conf\_matrix\_tree\_df$Percentage <- conf\_matrix\_tree\_df$Freq / sum(conf\_matrix\_tree\_df$Freq) \* 100  
  
# Plot the confusion matrix using ggplot2  
conf\_plot\_dt <- ggplot(conf\_matrix\_tree\_df, aes(x = Reference, y = Prediction, fill = Percentage)) +  
 geom\_tile(color = "white") +  
 geom\_text(aes(label = sprintf("%.1f%%", Percentage)), vjust = 1) +  
 scale\_fill\_gradient(low = "white", high = "skyblue") + # Adjust the color scale  
 theme\_minimal() +  
 labs(title = "Confusion Matrix - Decision Tree",  
 x = "Actual",  
 y = "Predicted") +  
 theme(axis.text = element\_text(size = 12), # Adjust text size  
 axis.title = element\_text(size = 14), # Adjust axis title size  
 plot.title = element\_text(size = 16)) # Adjust plot title size  
  
# Display the plot  
print(conf\_plot\_dt)



# Algorithm 2: Random Forest

# Split the dataset into a training set and a test set  
set.seed(123)  
trainIndex <- createDataPartition(LFS\_Selected$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[sample(nrow(LFS\_Selected), nrow(LFS\_Selected) / 10), ]  
  
# the testing data  
testing\_data <- LFS\_Selected[-trainIndex, ]  
  
# cross-validation  
control\_rf <- trainControl(method = "cv", number = 10) # 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 128557 48229  
## 2 19342 80659  
##   
## Accuracy : 0.7559   
## 95% CI : (0.7543, 0.7575)  
## No Information Rate : 0.5343   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.5023   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.8692   
## Specificity : 0.6258   
## Pos Pred Value : 0.7272   
## Neg Pred Value : 0.8066   
## Prevalence : 0.5343   
## Detection Rate : 0.4645   
## Detection Prevalence : 0.6387   
## Balanced Accuracy : 0.7475   
##   
## 'Positive' Class : 1   
##

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

## Accuracy: 0.7558737

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

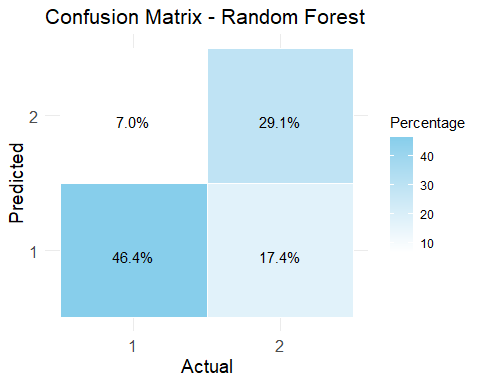
## Precision: 0.7271899

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

## Recall: 0.8692216

#Visualize Random Forest Confusion Matrix

# Convert the confusion matrix to a data frame for plotting  
conf\_matrix\_rf\_df <- as.data.frame(as.table(confusion\_rf))  
  
# Calculate percentages for a more informative plot  
conf\_matrix\_rf\_df$Percentage <- conf\_matrix\_rf\_df$Freq / sum(conf\_matrix\_rf\_df$Freq) \* 100  
  
# Plot the confusion matrix using ggplot2  
conf\_plot\_rf <- ggplot(conf\_matrix\_rf\_df, aes(x = Reference, y = Prediction, fill = Percentage)) +  
 geom\_tile(color = "white") +  
 geom\_text(aes(label = sprintf("%.1f%%", Percentage)), vjust = 1) +  
 scale\_fill\_gradient(low = "white", high = "skyblue") + # Adjust the color scale  
 theme\_minimal() +  
 labs(title = "Confusion Matrix - Random Forest",  
 x = "Actual",  
 y = "Predicted") +  
 theme(axis.text = element\_text(size = 12), # Adjust text size  
 axis.title = element\_text(size = 14), # Adjust axis title size  
 plot.title = element\_text(size = 16)) # Adjust plot title size  
  
# Display the plot  
print(conf\_plot\_rf)



## Algorithm 3: Naive Bayes

LFS\_Selected$LFSSTAT <- as.factor(LFS\_Selected$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)) # Use 10% of the data  
train\_index\_nb <- sample(1:nrow(LFS\_Selected), sample\_size)  
smaller\_training\_data\_nb <- LFS\_Selected[train\_index\_nb, ]  
testing\_data\_nb <- LFS\_Selected[-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 383891 151115  
## 2 59993 235366  
##   
## Accuracy : 0.7458   
## 95% CI : (0.7448, 0.7467)  
## No Information Rate : 0.5346   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.4812   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.8648   
## Specificity : 0.6090   
## Pos Pred Value : 0.7175   
## Neg Pred Value : 0.7969   
## Prevalence : 0.5346   
## Detection Rate : 0.4623   
## Detection Prevalence : 0.6443   
## Balanced Accuracy : 0.7369   
##   
## 'Positive' Class : 1   
##

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

## Accuracy: 0.7457648

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

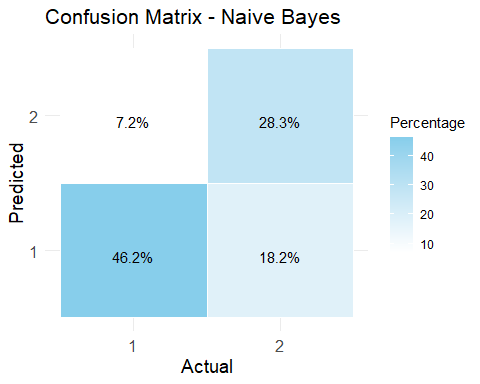
## Precision: 0.7175452

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

## Recall: 0.8648453

#Visualize Naive Bayes Confusion Matrix

# Convert the confusion matrix to a data frame for plotting  
conf\_matrix\_nb\_df <- as.data.frame(as.table(confusion\_nb))  
  
# Calculate percentages for a more informative plot  
conf\_matrix\_nb\_df$Percentage <- conf\_matrix\_nb\_df$Freq / sum(conf\_matrix\_nb\_df$Freq) \* 100  
  
# Plot the confusion matrix using ggplot2  
conf\_plot\_nb <- ggplot(conf\_matrix\_nb\_df, aes(x = Reference, y = Prediction, fill = Percentage)) +  
 geom\_tile(color = "white") +  
 geom\_text(aes(label = sprintf("%.1f%%", Percentage)), vjust = 1) +  
 scale\_fill\_gradient(low = "white", high = "skyblue") + # Adjust the color scale  
 theme\_minimal() +  
 labs(title = "Confusion Matrix - Naive Bayes",  
 x = "Actual",  
 y = "Predicted") +  
 theme(axis.text = element\_text(size = 12), # Adjust text size  
 axis.title = element\_text(size = 14), # Adjust axis title size  
 plot.title = element\_text(size = 16)) # Adjust plot title size  
  
# Display the plot  
print(conf\_plot\_nb)



# LFS Oct: Read October 2023 Dataset

Oct\_dataset\_path <- "C:/820WF/LFS\_2017\_2023/lfs\_1023.csv"  
Oct\_data <- read.csv(Oct\_dataset\_path)

# LFS Oct: Select a smaller subset of data and Combine Categories

# Select only the relevant variables  
Oct\_Selected <- Oct\_data[c("SURVYEAR","LFSSTAT", "PROV", "AGE\_12", "SEX", "MARSTAT", "EDUC", "IMMIG")]  
  
# Combine the four LFSSTAT categories into two levels  
Oct\_Selected$LFSSTAT <- ifelse(Oct\_Selected$LFSSTAT == 1, 1, 2)

# LFS Oct: Predictions on Oct Data

# Make predictions using the model  
predictions\_rf\_Oct <- predict(model\_rf, Oct\_Selected)  
  
# Ensure both are factors with the same levels  
predictions\_rf\_Oct <- as.factor(predictions\_rf\_Oct)  
Oct\_Selected$LFSSTAT <- as.factor(Oct\_Selected$LFSSTAT)  
  
# Confusion matrix  
confusion\_rf\_Oct <- confusionMatrix(predictions\_rf\_Oct, Oct\_Selected$LFSSTAT)  
  
# Calculate accuracy, precision, and recall  
accuracy\_rf\_Oct <- confusion\_rf\_Oct$overall['Accuracy']  
precision\_rf\_Oct <- confusion\_rf\_Oct$byClass['Pos Pred Value']  
recall\_rf\_Oct <- confusion\_rf\_Oct$byClass['Sensitivity']  
  
# Print the confusion matrix and results  
print(confusion\_rf\_Oct)

## Confusion Matrix and Statistics  
##   
## Reference  
## Prediction 1 2  
## 1 51818 18161  
## 2 7588 31035  
##   
## Accuracy : 0.7629   
## 95% CI : (0.7604, 0.7654)  
## No Information Rate : 0.547   
## P-Value [Acc > NIR] : < 2.2e-16   
##   
## Kappa : 0.5126   
##   
## Mcnemar's Test P-Value : < 2.2e-16   
##   
## Sensitivity : 0.8723   
## Specificity : 0.6308   
## Pos Pred Value : 0.7405   
## Neg Pred Value : 0.8035   
## Prevalence : 0.5470   
## Detection Rate : 0.4771   
## Detection Prevalence : 0.6444   
## Balanced Accuracy : 0.7516   
##   
## 'Positive' Class : 1   
##

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

## Accuracy: 0.7629049

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

## Precision: 0.7404793

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

## Recall: 0.8722688

#Visualize Random Forest Confusion Matrix - Oct

# Convert the confusion matrix to a data frame for plotting  
conf\_matrix\_rf\_Oct\_df <- as.data.frame(as.table(confusion\_rf\_Oct))  
  
# Calculate percentages for a more informative plot  
conf\_matrix\_rf\_Oct\_df$Percentage <- conf\_matrix\_rf\_Oct\_df$Freq / sum(conf\_matrix\_rf\_Oct\_df$Freq) \* 100  
  
# Plot the confusion matrix using ggplot2  
conf\_plot\_rf\_Oct <- ggplot(conf\_matrix\_rf\_Oct\_df, aes(x = Reference, y = Prediction, fill = Percentage)) +  
 geom\_tile(color = "white") +  
 geom\_text(aes(label = sprintf("%.1f%%", Percentage)), vjust = 1) +  
 scale\_fill\_gradient(low = "white", high = "skyblue") + # Adjust the color scale  
 theme\_minimal() +  
 labs(title = "Confusion Matrix - Random Forest",  
 x = "Actual",  
 y = "Predicted") +  
 theme(axis.text = element\_text(size = 12), # Adjust text size  
 axis.title = element\_text(size = 14), # Adjust axis title size  
 plot.title = element\_text(size = 16)) # Adjust plot title size  
  
# Display the plot  
print(conf\_plot\_rf\_Oct)

