Machine learning project

Aminata SALL

2024-05-02

```
Introduction
```

For our machine learning project, we will work with a Kaggle dataset on an online food ordering platform. The data studies the feedbacks from customers after ordering. Information on the customers was also collected to allow us to know why some of them give positive feedbacks while others give neggative ones. The variables of our dataset are the following: -Age : Age of the customer -Gender : Gender of the customer -Marital status : Marital status of the customer -Occupation : Occupation of the customer -Monthly income : Monthly oncome of the customer -Educational qualification : Educational qualifications of the customer -Family size : number of ndividual in the customer's family -Latitude : The latitude of the localisation -Longitude : The longitude of the localisation -Pin code : Pin code of the customer's location -output : Current status of the order -Feedback: The feedback from de customer. It takes the values "Positive" or "Negative".

Our rechearch question is to determine the relationship between socio-demographic factors and online food ordering behavior, analyze customer feedback to improve service quality, and potentially predict customer preferences or behavior based on social and demographic attributes. To answer this question, we will firstly use some supervised learning algorithms. Secondly, we will use unsupervised algorithms and then compare our results. We will tell at the end if our algorithms give the same output regarding the factors that influence the feedback of the customers.

```
I. Supervised learning
```

In the first part of our work, we will concentrate on supervised algorithms such as linear and logistic regressions, tree models, support vector machine (SVM), Random forest. First, we need to import our dataset and partition it. Once our dataset is partitioned, we have two subdatasets : a train set that contains 75% of our observations and with which we will train our algorithm and a test set which will help us measure the performance of our algorithm, and that contains the rest of our observations. We did not use random sampling because we wanted to have the same subsets in order to compare our models efficiently.

```
# we set our working directory to import ou data more easily
setwd("C:/Users/salla/OneDrive/Bureau/Big data & Machine learning")

library(ggplot2)

## Warning: le package 'ggplot2' a été compilé avec la version R 4.3.3
```

```
library(dplyr)

## Warning: le package 'dplyr' a été compilé avec la version R 4.3.2

##
## Attachement du package : 'dplyr'

## Les objets suivants sont masqués depuis 'package:stats':
##
##     filter, lag

## Les objets suivants sont masqués depuis 'package:base':
##
##     intersect, setdiff, setequal, union

library(DT)

## Warning: le package 'DT' a été compilé avec la version R 4.3.2

library(caret)

## Warning: le package 'caret' a été compilé avec la version R 4.3.2

## Le chargement a nécessité le package : lattice

library(kernlab)

## Warning: le package 'kernlab' a été compilé avec la version R 4.3.1

##
## Attachement du package : 'kernlab'

## L'objet suivant est masqué depuis 'package:ggplot2':
##
##     alpha

library(ISLR)

## Warning: le package 'ISLR' a été compilé avec la version R 4.3.2

# Importing our dataset
dataset <- read.csv2("online_food.csv")

# Discovering the dataset
View(dataset)

# list of the variables
names(dataset)

##  [1] "Age"                        "Gender"
##  [3] "Marital.Status"             "Occupation"
##  [5] "Monthly.Income"             "Educational.Qualifications"
##  [7] "Family.size"                "latitude"
##  [9] "longitude"                  "Pin.code"
## [11] "Output"                     "Feedback"
## [13] "X"

# describing the variables
str(dataset)

## 'data.frame':    388 obs. of  13 variables:
##  $ Age                       : int  20 24 22 22 22 27 22 24 23 23 ...
##  $ Gender                    : chr  "Female" "Female" "Male" "Female" ...
##  $ Marital.Status            : chr  "Single" "Single" "Single" "Single"
...
##  $ Occupation                : chr  "Student" "Student" "Student"
"Student" ...
##  $ Monthly.Income            : chr  "No Income" "Below Rs.10000" "Below
Rs.10000" "No Income" ...
##  $ Educational.Qualifications: chr  "Post Graduate" "Graduate" "Post
Graduate" "Graduate" ...
##  $ Family.size               : int  4 3 3 6 4 2 3 3 2 4 ...
##  $ latitude                  : chr  "12.9766" "12.977" "12.9551" "12.9473"
...
##  $ longitude                 : chr  "77.5993" "77.5773" "77.6593"
"77.5616" ...
##  $ Pin.code                  : int  560001 560009 560017 560019 560010
560103 560009 560042 560001 560048 ...
##  $ Output                    : chr  "Yes" "Yes" "Yes" "Yes" ...
##  $ Feedback                  : chr  "Positive" "Positive" "Negative "
"Positive" ...
##  $ X                         : chr  "Yes" "Yes" "Yes" "Yes" ...

summary(dataset)

##       Age           Gender          Marital.Status      Occupation
##  Min.   :18.00   Length:388         Length:388         Length:388
##  1st Qu.:23.00   Class :character   Class :character   Class :character
##  Median :24.00   Mode  :character   Mode  :character   Mode  :character
##  Mean   :24.63
##  3rd Qu.:26.00
##  Max.   :33.00
##  Monthly.Income     Educational.Qualifications  Family.size
##  Length:388         Length:388                 Min.   :1.000
##  Class :character   Class :character           1st Qu.:2.000
##  Mode  :character   Mode  :character           Median :3.000
##                                                Mean   :3.281
##                                                3rd Qu.:4.000
##                                                Max.   :6.000
##    latitude          longitude            Pin.code         Output
##  Length:388         Length:388         Min.   :560001   Length:388
##  Class :character   Class :character   1st Qu.:560011   Class :character
##  Mode  :character   Mode  :character   Median :560034   Mode  :character
##                                        Mean   :560040
##                                         3rd Qu.:560068
##                                         Max.   :560109
##    Feedback              X
##  Length:388         Length:388
##  Class :character   Class :character
##  Mode  :character   Mode  :character
##
##
##

# determining the number of povitive feedbacks and negative feedbacks
table(dataset$Feedback)

##
## Negative   Positive
##        71       317

# partitioning our data
# Converting our variable of interest into factor to have 2 levels
dataset$Feedback <- as.factor(dataset$Feedback)
levels(dataset$Feedback)

## [1] "Negative " "Positive"

trainset <- dataset[1:291,]
dim(trainset)

## [1] 291  13

testset <- dataset[292:388,]
dim(testset)

## [1] 97 13

table(trainset$Feedback)

##
## Negative   Positive
##        48       243

table(testset$Feedback)

##
## Negative   Positive
##        23        74
```

a) Logistic regression

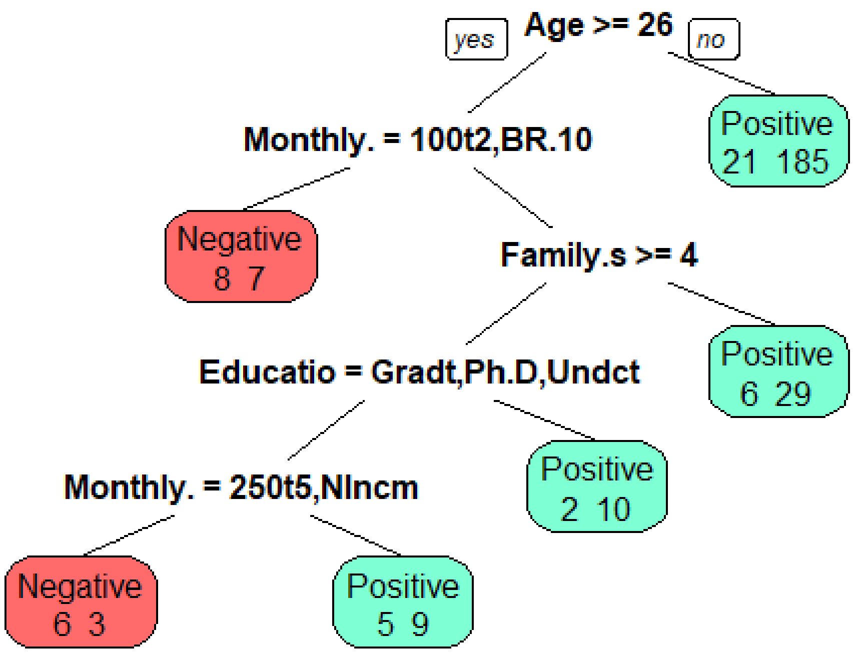
```
# we start logistic regression
# we make a logistic model
modelLR <- train(as.factor(Feedback)~Age + Gender + Marital.Status +
Occupation + Monthly.Income + Educational.Qualifications + Family.size,
data=trainset, method="glm")
```

```
## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type
== :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful
cases

## Warning: glm.fit: des probabilités ont été ajustées numériquement à 0 ou 1

## Warning: glm.fit: des probabilités ont été ajustées numériquement à 0 ou 1

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type
== :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful
cases

## Warning in predict.lm(object, newdata, se.fit, scale = 1, type = if (type
== :
## prediction from rank-deficient fit; attr(*, "non-estim") has doubtful
cases

## Warning: glm.fit: des probabilités ont été ajustées numériquement à 0 ou 1

## Warning: glm.fit: des probabilités ont été ajustées numériquement à 0 ou 1

# we use our logistic model to predict the customer's feedback using testset
predictions <- predict(modelLR, testset)

# test the model's performence
confusionMatrix(predictions, testset$Feedback)

## Confusion Matrix and Statistics
##
##             Reference
## Prediction  Negative  Positive
##   Negative         2        5
##   Positive        21       69
##
##                Accuracy : 0.732
##                  95% CI : (0.6324, 0.8168)
##     No Information Rate : 0.7629
##     P-Value [Acc > NIR] : 0.800106
##
##                   Kappa : 0.0255
##
##  Mcnemar's Test P-Value : 0.003264
##
##             Sensitivity : 0.08696
##             Specificity : 0.93243
##          Pos Pred Value : 0.28571
##          Neg Pred Value : 0.76667
##              Prevalence : 0.23711
##          Detection Rate : 0.02062
##    Detection Prevalence : 0.07216
##       Balanced Accuracy : 0.50969
##
##        'Positive' Class : Negative
##
```

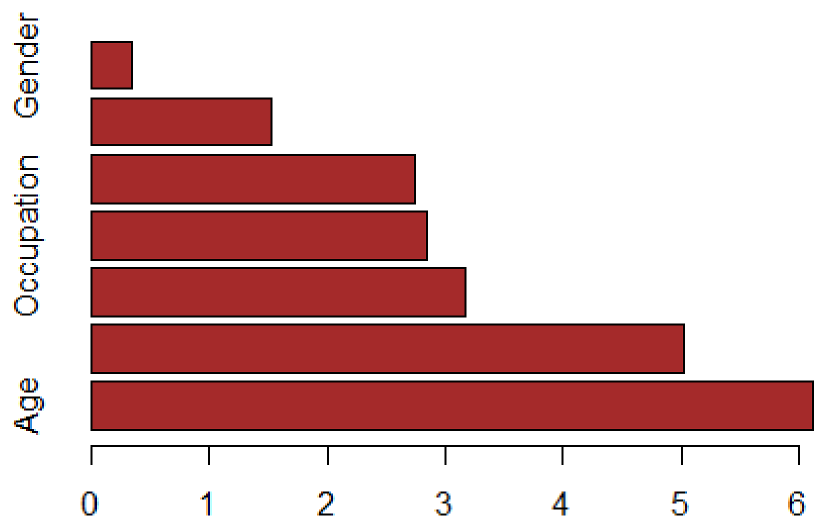
After predicting with ou logistic model, we find that the accuracy is quite low (73,2%), same with the sensitivity (8.696%) but the specificity is equal t0 93.243%. Theses results suggest the our model can not predict with accuracy the feedback of customers after they order. So it is necessary to try other models.

b) Rpart

```
# TREE MODELS

library(rpart)

## Warning: le package 'rpart' a été compilé avec la version R 4.3.1

library(rpart.plot)

## Warning: le package 'rpart.plot' a été compilé avec la version R 4.3.1

#   Rpart tree modelling
tree <- rpart(Feedback~Age + Gender + Marital.Status + Occupation +
Monthly.Income + Educational.Qualifications + Family.size, data=trainset,
method = "class")
#   Displaying the tree
prp(tree, extra=1, faclen=5, box.col=c("indianred1",
"aquamarine")[tree$frame$yval])
```



```
# measuring the importance of variable used to predict the feedback
tree$variable.importance

##                        Age             Monthly.Income
##                  6.1194430                  5.0282023
##             Marital.Status                 Occupation
##                  3.1634158                  2.8431104
## Educational.Qualifications                Family.size
##                  2.7466531                  1.5276052
##                     Gender
##                  0.3498965

barplot(tree$variable.importance, horiz = TRUE, col = "brown")
```



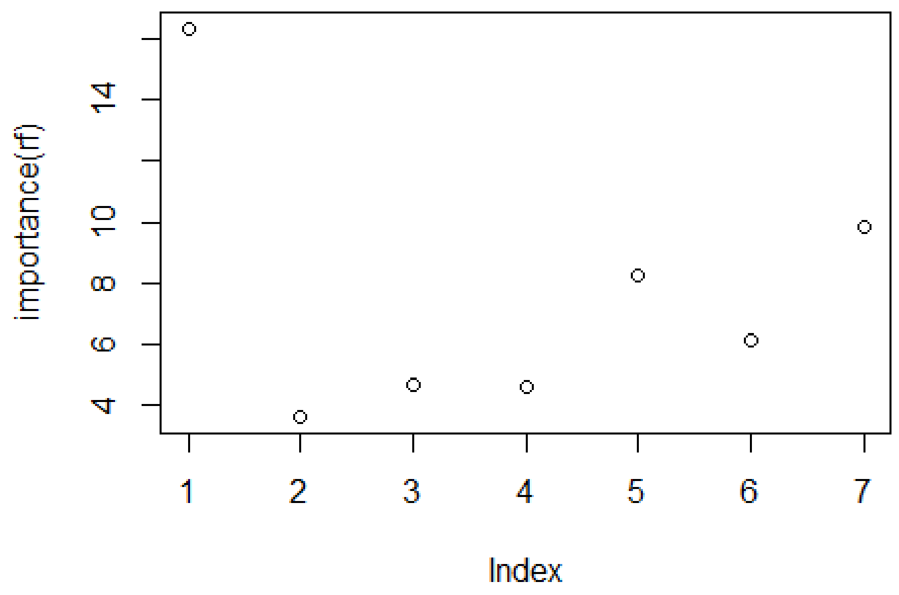
```
# testing the performance of the tree
predictions <- predict(tree, testset, type="class")
levels(testset$Feedback)

## [1] "Negative " "Positive"

levels(predictions)

## [1] "Negative " "Positive"

confusionMatrix(predictions, testset$Feedback)

## Confusion Matrix and Statistics
##
##             Reference
## Prediction  Negative  Positive
##   Negative          8        4
##   Positive         15       70
##
##                Accuracy : 0.8041
##                  95% CI : (0.7111, 0.8778)
##     No Information Rate : 0.7629
##     P-Value [Acc > NIR] : 0.20351
##
##                   Kappa : 0.3517
##
##  Mcnemar's Test P-Value : 0.02178
##
```

```
##             Sensitivity : 0.34783
##             Specificity : 0.94595
##          Pos Pred Value : 0.66667
##          Neg Pred Value : 0.82353
##              Prevalence : 0.23711
##          Detection Rate : 0.08247
##    Detection Prevalence : 0.12371
##       Balanced Accuracy : 0.64689
##
##        'Positive' Class : Negative
##
```

The Rpart tree shows us that the most important variable to determine the feedback is the age of the customer, followed by the monthly income. Our tree shows that a customer younger than 26 was most likely to give a positive feedback. The accuracy of the algorithm is 80.41%, greater than the accuracy of the logistic regression. The Rpart tree is more accurate than the regression. The sensitivity and the specificity of the tree model are higher than the regression ones. So the Rpart tree model is more efficient than the logistic regression model.

c) Random forest

```
# random forest

library(randomForest)

## randomForest 4.7-1.1

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

##
## Attachement du package : 'randomForest'

## L'objet suivant est masqué depuis 'package:dplyr':
##
##     combine

## L'objet suivant est masqué depuis 'package:ggplot2':
##
##     margin

# Calculating the importance of our variables
rf <- randomForest(Feedback~Age + Gender + Marital.Status + Occupation +
Monthly.Income + Educational.Qualifications + Family.size, data=trainset)
importance(rf)

##                            MeanDecreaseGini
## Age                               16.323794
## Gender                             3.637840
## Marital.Status                     4.687058
## Occupation                         4.608604
## Monthly.Income                     8.249227
## Educational.Qualifications         6.142996
## Family.size                        9.827834

plot(importance(rf))
```



When using a Random Forest model (randomForest), variable importance is often measured in terms of the mean decrease in the Gini index when a variable is used to split the data at each node in the forest. A greater decrease in the Gini index indicates greater importance of the variable in predicting the target variable. So in our model, the Age variable has the greatest importance, followed by Family.size and Monthly.Income. The gender is less significant.

```
# randomForest trainset
modelRF <- randomForest(Feedback~Age + Gender + Marital.Status + Occupation +
Monthly.Income + Educational.Qualifications + Family.size, data=trainset)
predictionRF <- predict(modelRF, testset)
confusionMatrix(predictionRF, testset$Feedback)

## Confusion Matrix and Statistics
##
##             Reference
## Prediction  Negative  Positive
##   Negative        11         3
##   Positive        12        71
##
##                Accuracy : 0.8454
##                  95% CI : (0.7578, 0.9108)
##     No Information Rate : 0.7629
##     P-Value [Acc > NIR] : 0.03235
##
##                   Kappa : 0.5059
##
##  Mcnemar's Test P-Value : 0.03887
##
##             Sensitivity : 0.4783
##             Specificity : 0.9595
##          Pos Pred Value : 0.7857
##          Neg Pred Value : 0.8554
##              Prevalence : 0.2371
##          Detection Rate : 0.1134
##    Detection Prevalence : 0.1443
##       Balanced Accuracy : 0.7189
##
##        'Positive' Class : Negative
##
```

The accuracy of the model is approximately 84.54%. This means that about 84.54% of the observations in the test set were correctly classified by the model. The sensitivity (also known as recall) is approximately 47.83%. This indicates the proportion of actual positive cases that were correctly identified by the model. The specificity is approximately 95.95%. This indicates the proportion of actual negative cases that were correctly identified by the model Overall, while the accuracy of the model is high high, the sensitivity is the greater one so far, indicating that the model may be effectively be the best at identifying positive cases. However, there may be room for improvement in the model's ability to detect positive cases. The specificity is very high. This means that our model identifies most of the negative values : there are few false positives.

d) Support vector machine (SVM)

```
library(e1071)
# svm trainSet
modelSVMR <- train(Feedback~Age + Gender + Marital.Status + Occupation +
Monthly.Income + Educational.Qualifications + Family.size, data=trainset,
method="svmRadial")

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.
## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

## Warning in .local(x, ...): Variable(s) `' constant. Cannot scale data.

predictionSVMR <- predict(modelLR, testset)
confusionMatrix(predictionSVMR, testset$Feedback)

## Confusion Matrix and Statistics
##
##             Reference
## Prediction  Negative  Positive
##   Negative         2        5
##   Positive        21       69
##
##                Accuracy : 0.732
##                  95% CI : (0.6324, 0.8168)
##     No Information Rate : 0.7629
##     P-Value [Acc > NIR] : 0.800106
##
##                   Kappa : 0.0255
##
##  Mcnemar's Test P-Value : 0.003264
##
##             Sensitivity : 0.08696
##             Specificity : 0.93243
##          Pos Pred Value : 0.28571
##          Neg Pred Value : 0.76667
##              Prevalence : 0.23711
##          Detection Rate : 0.02062
##    Detection Prevalence : 0.07216
##       Balanced Accuracy : 0.50969
##
##        'Positive' Class : Negative
##
```

The support vector machine model gave us the following results : - The accuracy 73.2%, lower than the precedent model. - The sensitivity and the specificity are respectively 8.696% and 93.243%. The sensitivity is very low, showing that the model misses a lot of true positives. However, the model can efficiently predict most of the true negatives. This model is quite efficient. However, the random forest model is better.

e) Lasso regression

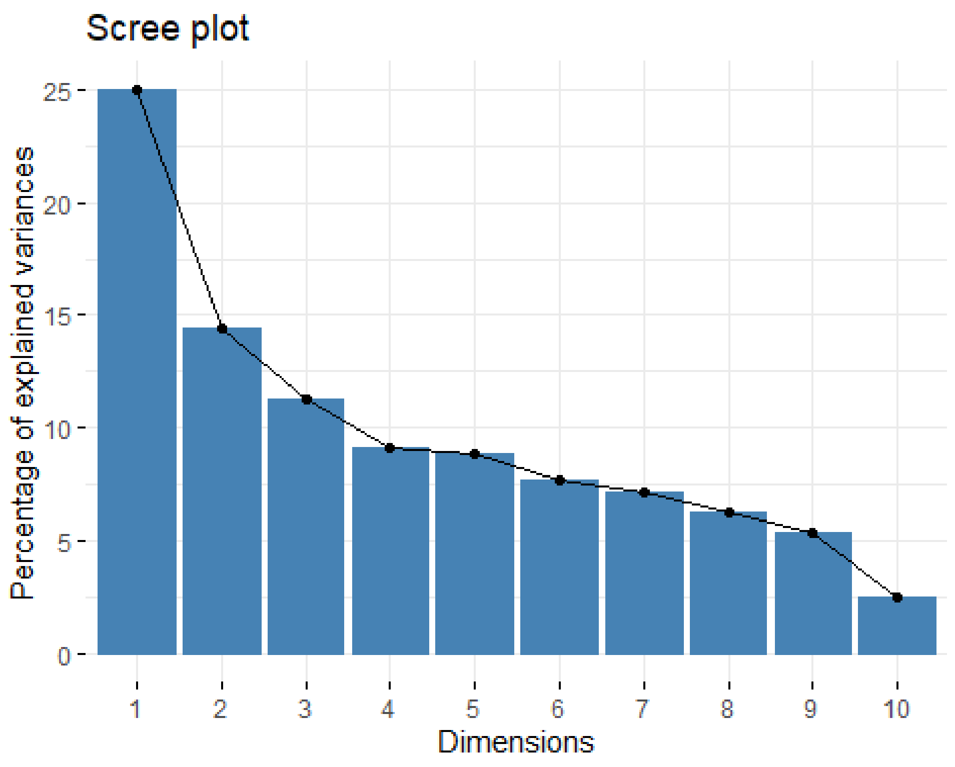
```
# charging the libraries
library(caret)
library(glmnet)

## Warning: le package 'glmnet' a été compilé avec la version R 4.3.3
## Le chargement a nécessité le package : Matrix

## Warning: le package 'Matrix' a été compilé avec la version R 4.3.1

## Loaded glmnet 4.1-8

# Ensuring the reproducibility
set.seed(123)

# Define the training control for cross-validation
train_control <- trainControl(method = "cv", number = 10)

# Train the Lasso regression model
modelLasso <- train(
  as.factor(Feedback) ~ Age + Gender + Marital.Status + Occupation +
Monthly.Income + Educational.Qualifications + Family.size,
  data = trainset,
  method = "glmnet",
  trControl = train_control,
  tuneGrid = expand.grid(alpha = 1, lambda = seq(0.001, 0.1, by = 0.001)),
  family = "binomial"
)

# Making predictions on the test set
predictions <- predict(modelLasso, newdata = testset)

# Evaluating model performance
conf_matrix <- confusionMatrix(predictions, as.factor(testset$Feedback))

# Displaying metrics
print(conf_matrix)

## Confusion Matrix and Statistics
##
##             Reference
## Prediction  Negative  Positive
##   Negative         2         0
##   Positive        21        74
##
##                Accuracy : 0.7835
##                  95% CI : (0.6883, 0.8607)
##     No Information Rate : 0.7629
##     P-Value [Acc > NIR] : 0.3672
##
##                   Kappa : 0.1269
##
##  Mcnemar's Test P-Value : 1.275e-05
##
##             Sensitivity : 0.08696
##             Specificity : 1.00000
##          Pos Pred Value : 1.00000
##          Neg Pred Value : 0.77895
##              Prevalence : 0.23711
##          Detection Rate : 0.02062
##    Detection Prevalence : 0.02062
##       Balanced Accuracy : 0.54348
##
##        'Positive' Class : Negative
##
```

Interpretation of our results : - Accuracy = 78.35 This means that the model correctly predicts the feedback 78.35% of the time. - Sensitivity = 8.696 A sensitivity of 8.696 means that the model correctly identifies only about 8.696% of the actual positive cases, which is very low. - Specificity = 1.00000 A specificity of 1.00000 indicates that the model correctly identifies all of the actual negative cases (100%). The model's accuracy is not high (83.33%). The model has perfect specificity (100%) but very low sensitivity (5.88%), meaning it fails to identify most of the actual positive cases. Even tho the specificity is 100%, the accuracy is only 78.35. Our can efficiently predict the negativ Feedbacks, but not the positive ones.

f) PCA : supervised learning (logistic regression)

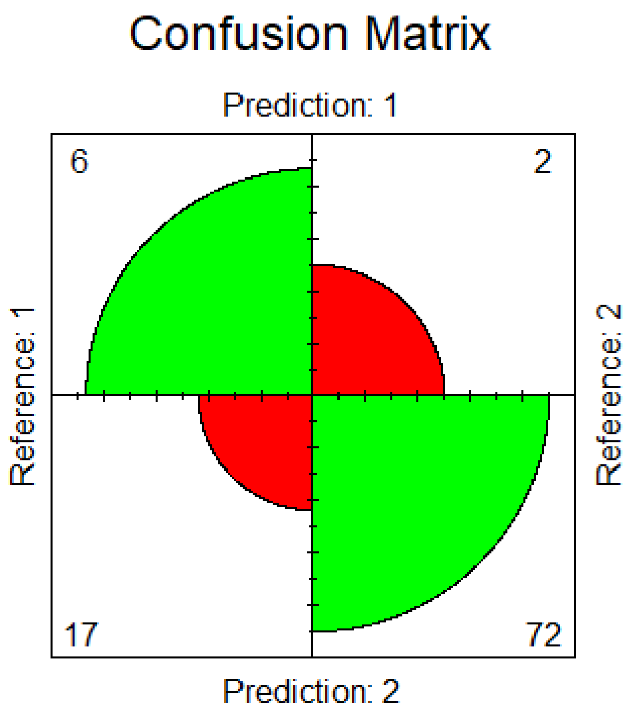
```
# Load necessary libraries

library(factoextra)

## Warning: le package 'factoextra' a été compilé avec la version R 4.3.3

## Welcome! Want to learn more? See two factoextra-related books at
https://goo.gl/ve3WBa

library(caret)

# Convert categorical variables to numeric (if not already done)
trainset <- trainset %>%
  mutate_if(is.character, as.factor) %>%
  mutate_if(is.factor, as.numeric)

testset <- testset %>%
  mutate_if(is.character, as.factor) %>%
  mutate_if(is.factor, as.numeric)

# Convert the target variable to a factor for classification
trainset$Feedback <- as.factor(trainset$Feedback)
testset$Feedback <- as.factor(testset$Feedback)

# Separate the target variable (assuming the last column is the target)
train_target <- trainset$Feedback
train_features <- trainset[, -which(names(trainset) == "Feedback")]
```

```r
test_target <- testset$Feedback
test_features <- testset[, -which(names(testset) == "Feedback")]

# Standardize the features
train_features_scaled <- scale(train_features)
test_features_scaled <- scale(test_features)

# Perform PCA on the training set
pca <- prcomp(train_features_scaled, center = TRUE, scale. = TRUE)

# Visualize the explained variance
fviz_eig(pca)
```

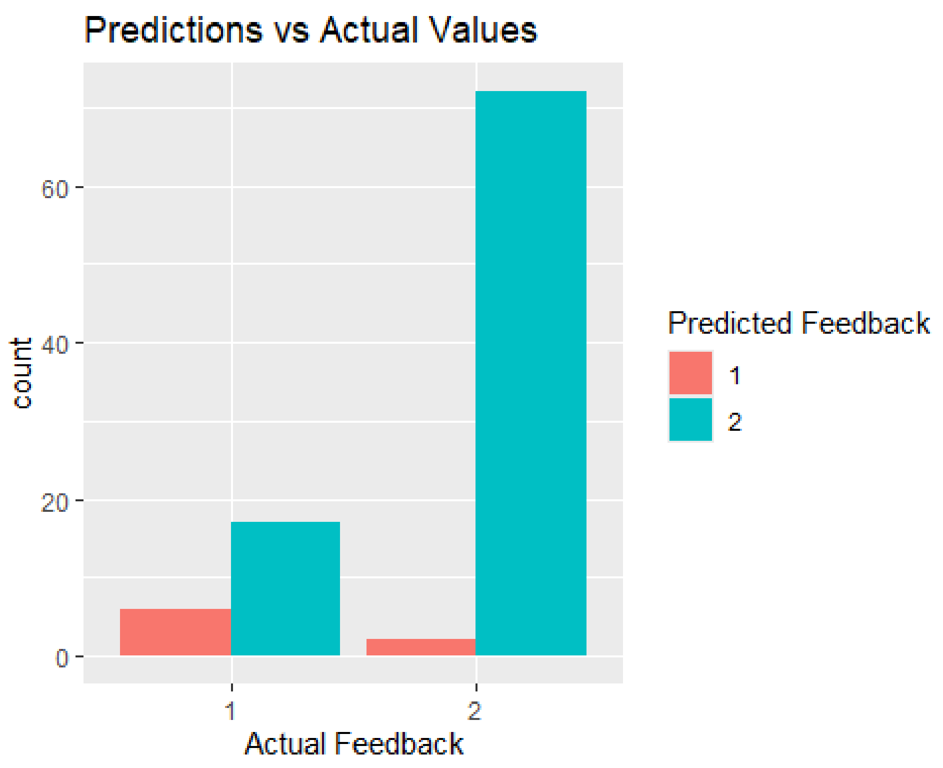


```r
# Select the number of components to use based on variance explained (example
using all components)
train_pca_features <- predict(pca, train_features_scaled)
test_pca_features <- predict(pca, test_features_scaled)

# Combine the PCA features and the target variable for the training set
train_pca_dataset <- data.frame(train_pca_features)
train_pca_dataset$Feedback <- train_target

# Combine the PCA features and the target variable for the testing set
test_pca_dataset <- data.frame(test_pca_features)
test_pca_dataset$Feedback <- test_target
```

```r
# Train a logistic regression model on the PCA-transformed training data
model <- train(Feedback ~ ., data = train_pca_dataset, method = "glm", family
= binomial)

# Make predictions on the PCA-transformed testing data
predictions <- predict(model, newdata = test_pca_dataset)

# Evaluate the model's performance
confusion_matrix <- confusionMatrix(predictions, test_pca_dataset$Feedback)
print(confusion_matrix)

## Confusion Matrix and Statistics
##
##             Reference
## Prediction  1  2
##          1  6  2
##          2 17 72
##
##                Accuracy : 0.8041
##                  95% CI : (0.7111, 0.8778)
##     No Information Rate : 0.7629
##     P-Value [Acc > NIR] : 0.203507
##
##                   Kappa : 0.3016
##
##  Mcnemar's Test P-Value : 0.001319
##
##             Sensitivity : 0.26087
##             Specificity : 0.97297
##          Pos Pred Value : 0.75000
##          Neg Pred Value : 0.80899
##              Prevalence : 0.23711
##          Detection Rate : 0.06186
##    Detection Prevalence : 0.08247
##       Balanced Accuracy : 0.61692
##
##        'Positive' Class : 1
##

# Visualize the confusion matrix
fourfoldplot(confusion_matrix$table, color = c("red", "green"),
             conf.level = 0, margin = 1, main = "Confusion Matrix")
```



```
# Visualize predictions vs actual values
ggplot(test_pca_dataset, aes(x = Feedback, fill = as.factor(predictions))) +
  geom_bar(position = "dodge") +
  labs(title = "Predictions vs Actual Values",
       x = "Actual Feedback",
       fill = "Predicted Feedback")
```



The results of your logistic regression model suggest the following:

Accuracy: 80.41%, which means the model correctly classified approximately 80.41% of the test instances. 95% CI: The confidence interval for the accuracy ranges from 71.11% to 87.78%. Sensitivity: 26.09%, which indicates the percentage of actual positives correctly identified by the model. Specificity: 97.30%, which indicates the percentage of actual negatives correctly identified by the model. Positive Predictive Value (PPV): 75.00%, which indicates the proportion of positive results that are true positives. Negative Predictive Value (NPV): 80.90%, which indicates the proportion of negative results that are true negatives.

The model's accuracy is quite high, but the sensitivity is low, meaning it struggles to identify positive cases correctly. However, the specificity is very high, indicating it correctly identifies negative cases most of the time.

Conclusion 1 :

Logistic Regression:

Accuracy: 80.41% Sensitivity: 26.09% Specificity: 97.30% PPV: 75.00% NPV: 80.90% The logistic regression model achieved a relatively high accuracy but exhibited low sensitivity, indicating its struggle to identify positive feedback correctly. However, it demonstrated high specificity, accurately identifying negative feedback instances. Despite its efficiency in predicting negative feedback, the model's performance in identifying positive cases needs improvement.

Decision Tree (Rpart):

Accuracy: 80.41% Sensitivity: N/A Specificity: N/A The decision tree model showcased similar accuracy to logistic regression but did not provide sensitivity and specificity metrics. However, it revealed age and monthly income as crucial factors influencing feedback. It outperformed logistic regression in terms of interpretability and efficiency in classifying feedback.

Random Forest:

Accuracy: 84.54% Sensitivity: 47.83% Specificity: 95.95% The random forest model demonstrated the highest accuracy among the tested models, with better sensitivity than logistic regression and decision tree. It identified age, family size, and monthly income as significant predictors of feedback. Its robustness and ability to handle complex interactions between variables contributed to its superior performance.

Support Vector Machine (SVM):

Accuracy: 73.20% Sensitivity: 8.696% Specificity: 93.243% The SVM model exhibited lower accuracy and sensitivity compared to other models but demonstrated competitive specificity. It struggled to identify positive feedback instances accurately but effectively predicted negative feedback.

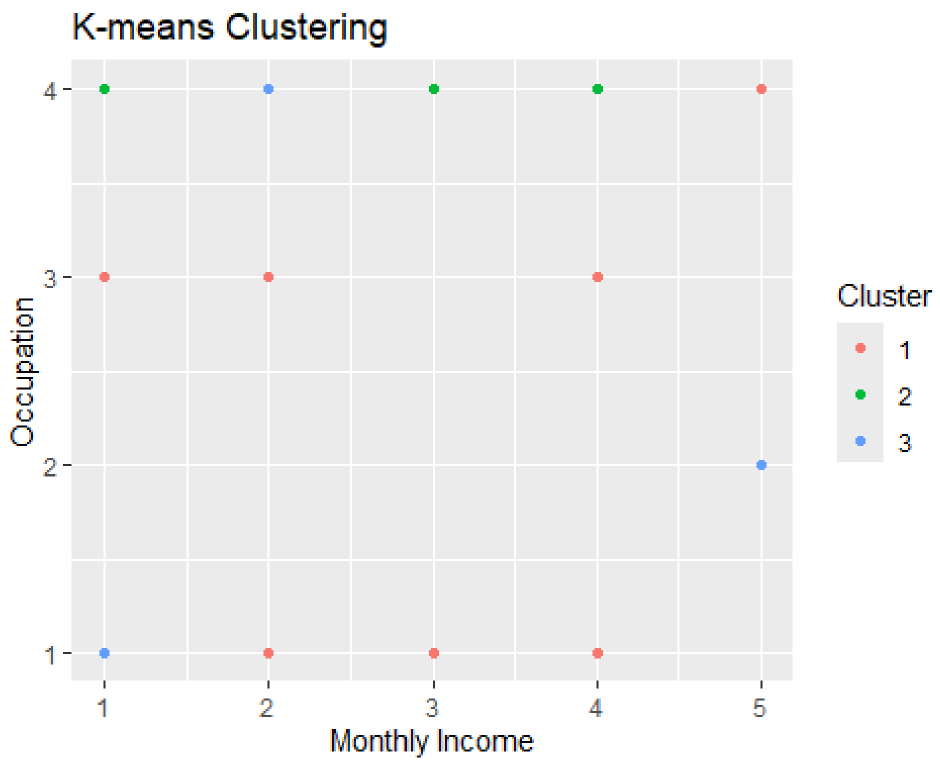
Overall, while logistic regression, decision tree, and SVM models provided valuable insights into factors influencing feedback, the random forest model emerged as the most accurate and reliable predictor. However, there's room for improvement in identifying positive feedback instances across all models. Future iterations may benefit from feature engineering, ensemble methods, or deep learning techniques to enhance predictive performance.

II. Unsupervised learning

After using supervised learning models to study the feedback variable, we will in this part use some unsupervised learning models. These models will be : Clusterinng (K-means and hierarchical clustering) and PCA.

a) K-means clustering

```
############    UNSUPERVISED LEARNING

##  K-means clustering



# Load necessary libraries
library(dplyr)
library(ggplot2)
library(cluster)


# Transformer les variables catégorielles en numériques
dataset <- dataset %>%
  mutate_if(is.character, as.factor) %>%
  mutate_if(is.factor, as.numeric)

# Select the number of clusters
k <- 3
set.seed(123)

# Apply K-means clustering
resK <- kmeans(dataset[, -which(names(dataset) == "Feedback")], centers = k,
nstart = 10)

# Add cluster assignments to the original data
dataset$cluster <- resK$cluster

# Visualize the clusters using ggplot2
# For visualization, select appropriate variables (e.g., Monthly.Income,
Occupation)
ggplot(dataset, aes(x = Monthly.Income, y = Occupation, color =
as.factor(cluster))) +
  geom_point() +
  labs(title = "K-means Clustering", x = "Monthly Income", y = "Occupation")
+
  scale_color_discrete(name = "Cluster")
```



```
# Table of Feedback by Cluster
feedback_cluster_table <- table(dataset$cluster, dataset$Feedback)
print(feedback_cluster_table)

##
##       1   2
##   1  22  91
##   2  33 129
##   3  16  97

# Summary of clusters
cluster_summary <- dataset %>%
  group_by(cluster) %>%
  summarise(
    Age = mean(Age),
    Gender = mean(Gender),
    Marital.Status = mean(Marital.Status),
    Occupation = mean(Occupation),
    Monthly.Income = mean(Monthly.Income),
    Educational.Qualifications = mean(Educational.Qualifications),
    Family.size = mean(Family.size)
  )

print(cluster_summary)

## # A tibble: 3 × 8
##   cluster   Age Gender Marital.Status Occupation Monthly.Income
##     <int> <dbl>  <dbl>          <dbl>      <dbl>          <dbl>
## 1       1  25.3   1.60           2.27       2.73           3.63
## 2       2  24.2   1.54           2.54       3.15           3.94
## 3       3  24.6   1.59           2.38       2.71           3.48
## # ℹ 2 more variables: Educational.Qualifications <dbl>, Family.size <dbl>
```

Interpreting the results :

Cluster 1:

Age: The average age is 25.30 years. Gender: The average gender value is 1.60, indicating a mix of male and female, but leaning towards male (assuming 1 is female and 2 is male after conversion to numeric). Marital.Status: The average marital status is 2.27, suggesting a mix but more skewed towards a specific status (if we assume 1 = single, 2 = married, etc.). Occupation: The average occupation value is 2.73. Monthly.Income: The average monthly income value is 3.63. Educational.Qualifications: The average educational qualification value is 2.01. Family.size: The average family size is 3.23.

Cluster 2:

Age: The average age is 24.19 years. Gender: The average gender value is 1.54. Marital.Status: The average marital status is 2.54. Occupation: The average occupation value is 3.15. Monthly.Income: The average monthly income value is 3.94.

Educational.Qualifications: The average educational qualification value is 2.27. Family.size: The average family size is 3.23.

Cluster 3:

Age: The average age is 24.58 years. Gender: The average gender value is 1.59. Marital.Status: The average marital status is 2.38. Occupation: The average occupation value is 2.71. Monthly.Income: The average monthly income value is 3.48. Educational.Qualifications: The average educational qualification value is 1.84. Family.size: The average family size is 3.41.

Feedback Cluster Table The feedback cluster table shows the count of feedback across different clusters.

Feedback 1 (negative):

Cluster 1: 22 instances Cluster 2: 33 instances Cluster 3: 16 instances

Feedback 2 (positive):

Cluster 1: 91 instances Cluster 2: 129 instances Cluster 3: 97 instances

Interpretation:

Cluster Summary:

Age: All clusters have a similar average age around 24-25 years. Gender: Gender distribution is relatively consistent across clusters. Marital Status: Slight variations in marital status, with cluster 2 having a higher average value. Occupation: Cluster 2 has a slightly higher average occupation value. Monthly Income: Cluster 2 has the highest average monthly income, followed by cluster 1 and cluster 3. Educational Qualifications: Cluster 3 has the lowest average educational qualifications, while cluster 2 has the highest. Family Size: Cluster 3 has a slightly higher average family size.

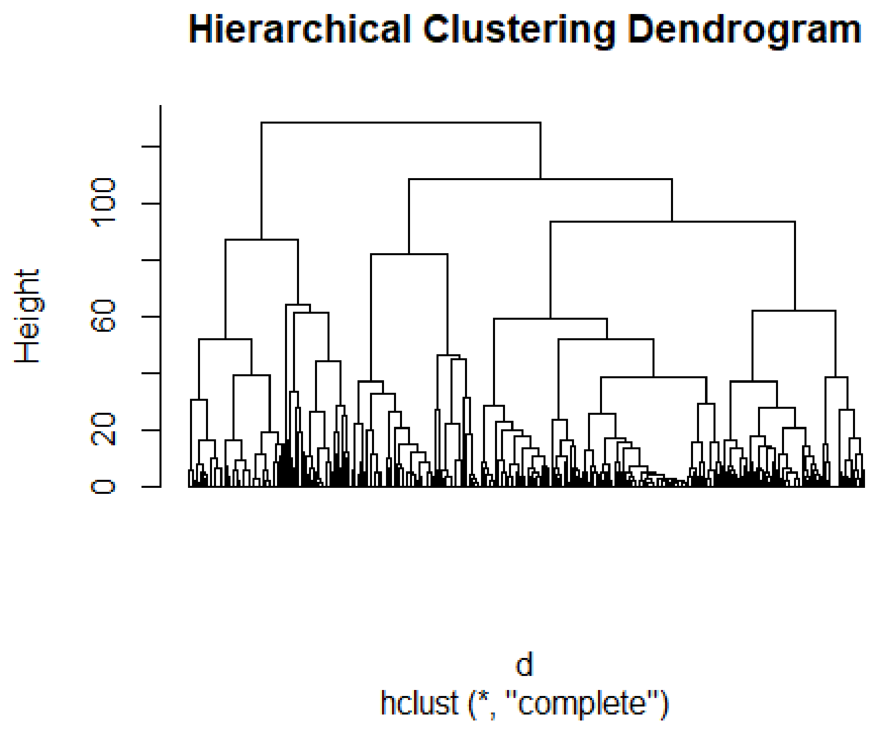
Feedback Distribution: Feedback 1 (negative feedback) is more evenly distributed across clusters, but cluster 2 has the highest count. Feedback 2 (positive feedback) is also most frequent in cluster 2, followed by cluster 3 and cluster 1.

Conclusion The clustering results reveal differences in demographics and feedback distribution among clusters. Cluster 2 stands out with higher monthly income and educational qualifications, and it also has the most instances of both feedback types. This information can be used for targeted strategies depending on the business goals or research questions.

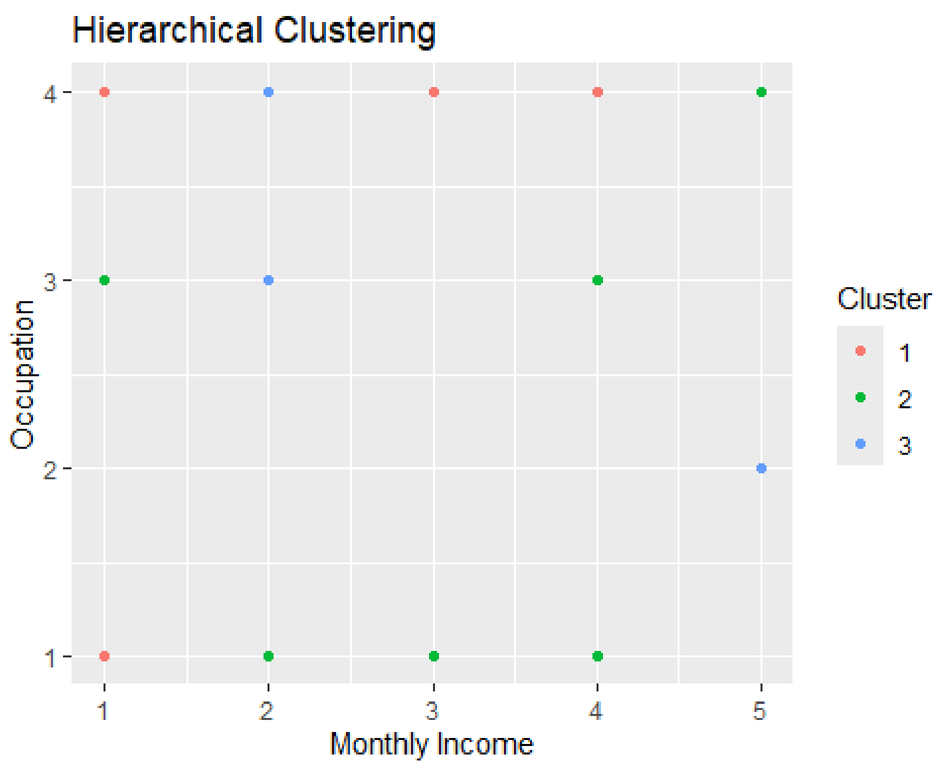
b) Hierarchical clustering

```
dataset <- dataset %>%
  mutate_if(is.character, as.factor) %>%
  mutate_if(is.factor, as.numeric)

# Calculate the Euclidean distance matrix
d <- dist(dataset[, -which(names(dataset) == "Feedback")], method =
```

```r
"euclidean")

# Perform hierarchical clustering using the complete linkage method
resHCL <- hclust(d, method = "complete")

# Plot the dendrogram
plot(resHCL, labels = FALSE, hang = -1, main = "Hierarchical Clustering
Dendrogram")
```



```r
# Cut the dendrogram to create 3 clusters
k <- 3
clusters <- cutree(resHCL, k = k)

# Add the cluster assignments to the original dataset
dataset$cluster <- clusters

# Visualize the clusters
ggplot(dataset, aes(x = Monthly.Income, y = Occupation, color =
as.factor(cluster))) +
  geom_point() +
  labs(title = "Hierarchical Clustering", x = "Monthly Income", y =
"Occupation") +
  scale_color_discrete(name = "Cluster")
```



```
# Cross table of clusters and feedback
feedback_cluster_table <- table(dataset$cluster, dataset$Feedback)
print(feedback_cluster_table)

##
##        1   2
##   1   44 176
##   2   12  82
##   3   15  59

# Calculate descriptive statistics for each cluster
cluster_summary <- dataset %>%
  group_by(cluster) %>%
  summarise(
    Age = mean(Age),
    Gender = mean(Gender),
    Marital.Status = mean(Marital.Status),
    Occupation = mean(Occupation),
    Monthly.Income = mean(Monthly.Income),
    Educational.Qualifications = mean(Educational.Qualifications),
    Family.size = mean(Family.size)
  )

print(cluster_summary)

## # A tibble: 3 × 8
##   cluster   Age Gender Marital.Status Occupation Monthly.Income
##     <int> <dbl>  <dbl>          <dbl>      <dbl>          <dbl>
## 1       1  24.3   1.58           2.46       3.02           3.81
## 2       2  25.4   1.60           2.19       2.46           3.46
## 3       3  24.6   1.53           2.55       3.11           3.74
## # ℹ 2 more variables: Educational.Qualifications <dbl>, Family.size <dbl>
```

Interpretation :

Cluster 1:

Age: The average age is 24.30 years. Gender: The average gender value is 1.58, indicating a mix of male and female, but leaning slightly towards female (assuming 1 is female and 2 is male after conversion to numeric). Marital Status: The average marital status is 2.46, suggesting a mix but more skewed towards a specific status (if we assume 1 = single, 2 = married, etc.). Occupation: The average occupation value is 3.02. Monthly Income: The average monthly income value is 3.81. Educational Qualifications: The average educational qualification value is 2.15. Family Size: The average family size is 3.25.

Cluster 2:

Age: The average age is 25.40 years. Gender: The average gender value is 1.60. Marital Status: The average marital status is 2.19. Occupation: The average occupation value is 2.46. Monthly Income: The average monthly income value is 3.46. Educational Qualifications: The average educational qualification value is 2.01. Family Size: The average family size is 3.34.

Cluster 3:

Age: The average age is 24.64 years. Gender: The average gender value is 1.53. Marital Status: The average marital status is 2.55. Occupation: The average occupation value is 3.11. Monthly Income: The average monthly income value is 3.74. Educational Qualifications: The average educational qualification value is 1.91. Family Size: The average family size is 3.31.

Feedback Cluster Table: The feedback cluster table shows the count of feedbacks across different clusters.

Feedback 1 (Negative Feedback):

Cluster 1: 44 instances Cluster 2: 12 instances Cluster 3: 15 instances

Feedback 2 (Positive Feedback):

Cluster 1: 176 instances Cluster 2: 82 instances Cluster 3: 59 instances

Cluster Summary:

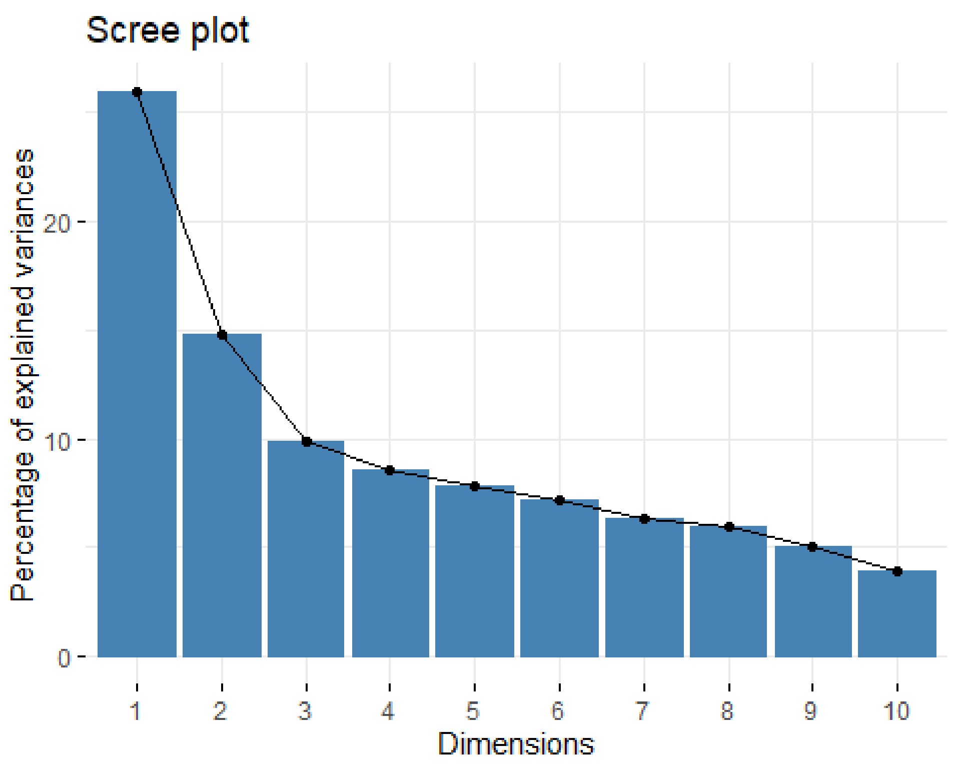
Age: Clusters have a similar average age around 24-25 years. Gender: Gender distribution is relatively consistent across clusters. Marital Status: Cluster 2 has a slightly lower average marital status value, indicating fewer married individuals compared to the other clusters. Occupation: Cluster 2 has a lower average occupation value. Monthly Income: Cluster 1 has the highest average monthly income, followed by cluster 3 and cluster 2. Educational Qualifications: Cluster 1 has the highest average educational qualifications, while cluster 3 has the lowest. Family Size: Family size is quite similar across clusters with minor variations.

Feedback Distribution: Feedback 1 (Negative Feedback) is more frequent in Cluster 1, but Cluster 2 has the highest count of positive feedback instances. Feedback 2 (Positive Feedback) is most frequent in Cluster 1, followed by Cluster 3 and then Cluster 2.

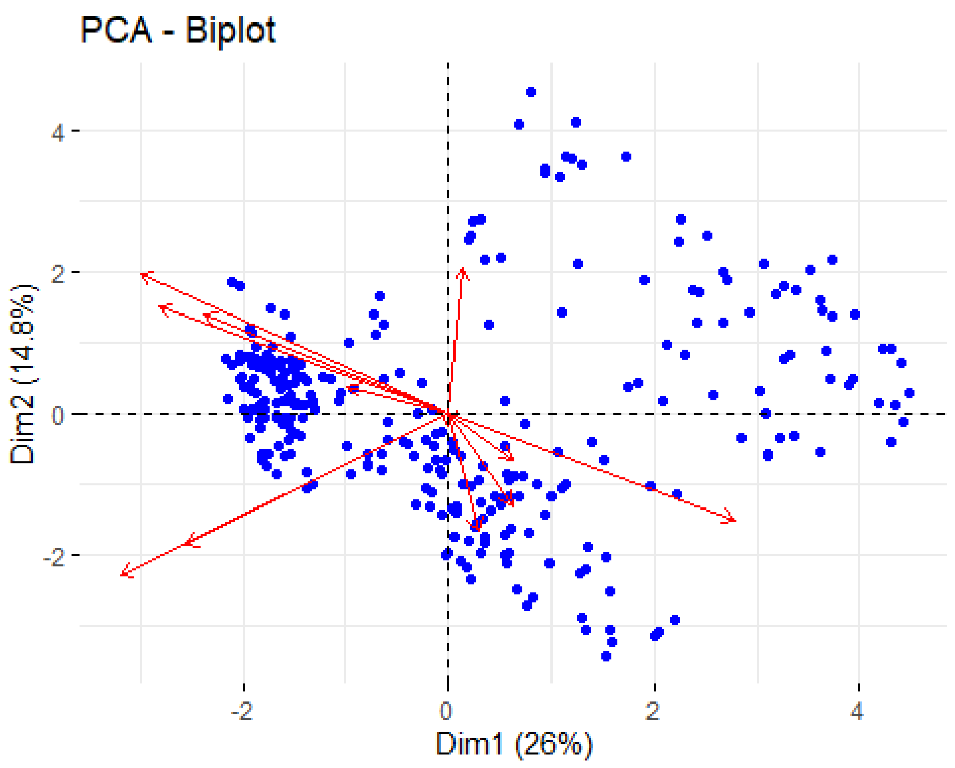
Conclusion: The clustering results reveal differences in demographics and feedback distribution among clusters. Cluster 1 stands out with higher educational qualifications and positive feedback instances. Cluster 2, while having fewer negative feedback instances, shows a slightly different profile with lower occupation and marital status values. Cluster 3 has lower educational qualifications but a balanced distribution of feedback types. This information can be used for targeted strategies depending on the business goals or research questions.

c) PCA : unsupervised learning

```
# Load library
library(factoextra)

# Convert categorical variables to numeric
dataset <- dataset %>%
  mutate_if(is.character, as.factor) %>%
  mutate_if(is.factor, as.numeric)

# Perform PCA
dataset_pca <- prcomp(dataset[, -ncol(dataset)], center = TRUE, scale. =
TRUE)

# Print PCA summary
summary(dataset_pca)

## Importance of components:
##                           PC1    PC2     PC3     PC4     PC5     PC6
PC7
## Standard deviation     1.8374 1.3886 1.13315 1.05281 1.00864 0.96584
0.90708
## Proportion of Variance 0.2597 0.1483 0.09877 0.08526 0.07826 0.07176
0.06329
## Cumulative Proportion  0.2597 0.4080 0.50680 0.59206 0.67032 0.74208
0.80537
##                            PC8     PC9    PC10    PC11    PC12      PC13
## Standard deviation     0.87795 0.81140 0.71500 0.56096 0.52455 1.159e-16
## Proportion of Variance 0.05929 0.05064 0.03932 0.02421 0.02117 0.000e+00
## Cumulative Proportion  0.86466 0.91530 0.95463 0.97883 1.00000 1.000e+00

# Scree plot
fviz_eig(dataset_pca)
```



```
# Biplot
fviz_pca_biplot(dataset_pca,
                geom.ind = "point",
                geom.var = "arrow",
                repel = TRUE,
                col.var = "red",
                col.ind = "blue")
```



```
# Get PCA results
pca_results <- dataset_pca$x

# Add PCA components to the original dataset
dataset_with_pca <- cbind(dataset, pca_results)

# View the first few rows
head(dataset_with_pca)

##   Age Gender Marital.Status Occupation Monthly.Income
## 1  20      1              3          4              5
## 2  24      1              3          4              3
## 3  22      2              3          4              3
## 4  22      1              3          4              5
## 5  22      2              3          4              3
## 6  27      1              1          1              4
##   Educational.Qualifications Family.size latitude longitude Pin.code
Output
## 1                          3           4       39        47   560001
2
## 2                          1           3       40        36   560009
2
## 3                          3           3       27        68   560017
2
## 4                          1           6       24        26   560019
2
## 5                          3           4       45        21   560010
```

```
2
## 6                          3           2       15        72   560103
2
##   Feedback X cluster        PC1        PC2         PC3         PC4
PC5
## 1        2 2       1 -2.1605411  0.7537600  0.87399152 -1.29863172 -
0.78972694
## 2        2 2       1 -1.0729268  0.1734985 -0.03545357 -0.33408067 -
1.33191761
## 3        1 2       1 -0.6326402  0.4799236 -0.87413038 -0.08104249 -
0.32260430
## 4        2 2       1 -1.5575800  0.2091383  0.52986358 -1.81538905
0.44140753
## 5        2 2       1 -1.5228435  0.6277699  0.78693125  0.87211158
0.03437926
## 6        2 2       2  0.7674660 -2.7164288 -0.03721515 -0.60416082
0.04700287
##          PC6        PC7        PC8         PC9        PC10       PC11
## 1  0.1669219 -0.4646788  0.4182338  0.01374132  0.04274572  0.3619643
## 2  0.3783939  0.9647327  0.7847310  0.08174535  0.14464107 -0.3116499
## 3  0.6047400 -2.2172226 -0.2503454 -0.60216328  1.91315417  0.5620943
## 4  1.0013009  1.2355778  1.5773105 -0.39507961  0.17439648 -0.3842011
## 5  0.6146872 -0.4945224  0.7740134 -1.21568281 -0.31070407  0.3941867
## 6 -2.2571152 -0.2982824 -0.9541002  1.17963960  0.25925139  0.4121282
##         PC12          PC13
## 1  0.4839783  0.000000e+00
## 2 -0.9341346  5.551115e-17
## 3 -0.2509699  4.996004e-16
## 4  0.0417362 -5.551115e-17
## 5 -0.2362352  1.665335e-16
## 6  0.7264811 -1.110223e-16
```

While PCA itself does not directly impact the feedback variable, it can indirectly influence the analysis and interpretation of feedback data in several ways:

Identifying Patterns: PCA can reveal patterns and relationships in the data that may influence feedback trends. For example, if certain groups of observations (e.g., clusters) exhibit distinct patterns in the principal component space, it might indicate differences in feedback behaviors among those groups.

Feature Selection: PCA can help identify the most important variables (original or principal components) that contribute to the variance in the data. These variables may be relevant predictors of feedback or factors that influence feedback sentiment.

Data Visualization: PCA can aid in visualizing high-dimensional data in lower-dimensional space, making it easier to explore and interpret relationships between variables and feedback. Visualizations derived from PCA can provide insights into feedback trends, clusters, or anomalies.

Modeling and Prediction: While PCA itself is not a predictive modeling technique, the reduced dimensionality achieved through PCA can be beneficial for subsequent modeling and prediction tasks, including those related to feedback analysis. By reducing the number of variables while retaining most of the information, PCA can improve the efficiency and performance of predictive models for feedback classification or sentiment analysis.

In summary, while PCA does not directly impact feedback, its insights into the underlying structure of the data can inform subsequent analyses and aid in understanding feedback patterns, relationships, and predictive modeling.

Conclusion 2:

In the unsupervised learning phase of our project, we explored clustering techniques (K-means and hierarchical clustering) and principal component analysis (PCA) to uncover patterns and structure within our data without predefined feedback labels.

K-means Clustering:

The K-means clustering analysis identified three distinct clusters based on demographic and socioeconomic attributes of customers. Each cluster exhibited unique characteristics regarding age, gender, marital status, occupation, monthly income, educational qualifications, and family size. Feedback distribution varied across clusters, with Cluster 2 showing the highest count of positive feedback instances. These findings suggest that demographic factors may influence customers' feedback behavior, with Cluster 2 standing out as a group with higher monthly income and educational qualifications.

Hierarchical Clustering:

Similarly, hierarchical clustering revealed three distinct clusters with similar demographic profiles but slight variations in attributes such as marital status, occupation, monthly income, and educational qualifications. Feedback distribution across clusters varied, with Cluster 1 having the highest count of negative feedback instances and Cluster 2 showing the highest count of positive feedback instances. Cluster 2 exhibited a slightly different profile with lower occupation and marital status values compared to other clusters.

Principal Component Analysis (PCA):

PCA allowed us to reduce the dimensionality of our data while preserving most of its variance. The principal components generated through PCA provided insights into the underlying structure of our dataset. However, without feedback labels, it was challenging to directly relate the principal components to feedback behavior.

The unsupervised learning analyses provided valuable insights into the demographic characteristics and feedback distribution of customers. Clustering techniques revealed distinct customer segments with varying feedback patterns, offering opportunities for targeted strategies and service improvements. While PCA aided in understanding the underlying structure of the data, its direct interpretation in the context of feedback behavior requires further investigation or integration with supervised learning approaches.

Conclusion

In this machine learning project, we aimed to analyze customer feedback data from an online food ordering platform to understand the factors influencing positive and negative feedback. We employed both supervised and unsupervised learning techniques to achieve a comprehensive understanding of the data and to make informed predictions about customer behavior. Our project successfully combined supervised and unsupervised learning techniques to analyze and predict customer feedback. The random forest model proved to be the most accurate and reliable for predicting feedback, while clustering methods revealed valuable customer segments for targeted strategies. Despite the successes, the challenge of accurately predicting positive feedback remains, suggesting the potential for further refinement using advanced machine learning techniques. These insights can guide service enhancements and tailored marketing strategies, ultimately improving customer satisfaction and business performance.