PRIFYSGOL
BANGOR
UNIVERSITY

Student Name	**Gelareh Kabiri**
Module Supervisor	**Heather He**
Module title	**Data Science**
Date of Submission	**18/01/2024**

Contents

Data Science

500681622

2024-01-05

Task_1

```
options(repos = list(CRAN = "https://cloud.r-project.org"))

# installing required packages
install.packages("rpart")

## Installing package into 'C:/Users/sohai/AppData/Local/R/win-library/4.3'
## (as 'lib' is unspecified)

## package 'rpart' successfully unpacked and MD5 sums checked

## Warning: cannot remove prior installation of package 'rpart'

## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying
## C:\Users\sohai\AppData\Local\R\win-
library\4.3\00LOCK\rpart\libs\x64\rpart.dll
## to C:\Users\sohai\AppData\Local\R\win-
library\4.3\rpart\libs\x64\rpart.dll:
## Permission denied

## Warning: restored 'rpart'

##
## The downloaded binary packages are in
##  C:\Users\sohai\AppData\Local\Temp\RtmpUHnVcS\downloaded_packages

install.packages("DBI")

## Installing package into 'C:/Users/sohai/AppData/Local/R/win-library/4.3'
## (as 'lib' is unspecified)

##
##   There is a binary version available but the source version is later:
##     binary source needs_compilation
## DBI  1.2.0  1.2.1             FALSE

## installing the source package 'DBI'
```

```r
install.packages("RMySQL")
```

```
## Installing package into 'C:/Users/sohai/AppData/Local/R/win-library/4.3'
## (as 'lib' is unspecified)

## package 'RMySQL' successfully unpacked and MD5 sums checked

## Warning: cannot remove prior installation of package 'RMySQL'

## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying
## C:\Users\sohai\AppData\Local\R\win-
library\4.3\00LOCK\RMySQL\libs\x64\RMySQL.dll
## to C:\Users\sohai\AppData\Local\R\win-
library\4.3\RMySQL\libs\x64\RMySQL.dll:
## Permission denied

## Warning: restored 'RMySQL'

##
## The downloaded binary packages are in
##  C:\Users\sohai\AppData\Local\Temp\RtmpUHnVcS\downloaded_packages
```

```r
library(rpart)
library(DBI)
library(RMySQL)
USER <- 'root'
PASSWORD <- '@G123K321MyA'
HOST <- 'localhost'
DBNAME <- 'world'

db <- dbConnect(MySQL(), user= USER, password= PASSWORD, host= HOST, dbname=
DBNAME, port=3306)
result <- dbGetQuery(db, statement="select * from world.customerchurn")
dbDisconnect(db)
```

```
## [1] TRUE
```

```r
head(result)
```

```
##    CUSTOMERID COLLEGE INCOME OVERAGE LEFTOVER  HOUSE HANDSET_PRICE
## 1 BTLC-007761    zero  89318       0        0 162233           266
## 2 BTLC-007682     one 142814     187       17 346690           716
## 3 BTLC-002228    zero  55675       0       32 792662           257
## 4 BTLC-011752     one  39559       0        0 416439           165
## 5 BTLC-015958    zero 145081       0        0 341108           583
## 6 BTLC-013969     one 120631      66       17 467811           884
##   OVER_15MINS_CALLS_PER_MONTH AVERAGE_CALL_DURATION REPORTED_SATISFACTION
## 1                           1                    12                 unsat
## 2                          24                     4                 unsat
## 3                           1                     1            very_unsat
## 4                           0                    15              very_sat
## 5                           0                     9                   avg
```

```
## 6                            4                      6                    sat
##   REPORTED_USAGE_LEVEL CONSIDERING_CHANGE_OF_PLAN LEAVE
## 1          very_little               considering  STAY
## 2                 high               considering LEAVE
## 3          very_little             never_thought  STAY
## 4                 high               considering  STAY
## 5                  avg                        no LEAVE
## 6            very_high               considering LEAVE
```

```
#data_cleaning
```

```
install.packages("dplyr")
```

```
## Installing package into 'C:/Users/sohai/AppData/Local/R/win-library/4.3'
## (as 'lib' is unspecified)
```

```
## package 'dplyr' successfully unpacked and MD5 sums checked
```

```
## Warning: cannot remove prior installation of package 'dplyr'
```

```
## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying
## C:\Users\sohai\AppData\Local\R\win-
library\4.3\00LOCK\dplyr\libs\x64\dplyr.dll
## to C:\Users\sohai\AppData\Local\R\win-
library\4.3\dplyr\libs\x64\dplyr.dll:
## Permission denied
```

```
## Warning: restored 'dplyr'
```

```
##
## The downloaded binary packages are in
##  C:\Users\sohai\AppData\Local\Temp\RtmpUHnVcS\downloaded_packages
```

```
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
```

```
# Remove CUSTOMERID column
result <- select(result, -CUSTOMERID)

# Convert 'COLLEGE' to a binary format (0 = No, 1 = Yes)
result$COLLEGE <- ifelse(result$COLLEGE == "zero", 0, 1)
head(result)
```

```
##   COLLEGE INCOME OVERAGE LEFTOVER  HOUSE HANDSET_PRICE
## 1       0  89318       0        0 162233           266
## 2       1 142814     187       17 346690           716
## 3       0  55675       0       32 792662           257
## 4       1  39559       0        0 416439           165
## 5       0 145081       0        0 341108           583
## 6       1 120631      66       17 467811           884
##   OVER_15MINS_CALLS_PER_MONTH AVERAGE_CALL_DURATION REPORTED_SATISFACTION
## 1                           1                    12                 unsat
## 2                          24                     4                 unsat
## 3                           1                     1            very_unsat
## 4                           0                    15              very_sat
## 5                           0                     9                   avg
## 6                           4                     6                   sat
##   REPORTED_USAGE_LEVEL CONSIDERING_CHANGE_OF_PLAN LEAVE
## 1          very_little                considering  STAY
## 2                 high                considering LEAVE
## 3          very_little              never_thought  STAY
## 4                 high                considering  STAY
## 5                  avg                         no LEAVE
## 6            very_high                considering LEAVE
```

```
#making_Decision_Tree

# Convert categorical variables to factors
categorical_columns <- c('COLLEGE', 'REPORTED_SATISFACTION',
'REPORTED_USAGE_LEVEL', 'CONSIDERING_CHANGE_OF_PLAN', 'LEAVE')
result[categorical_columns] <- lapply(result[categorical_columns], as.factor)
set.seed(123)  # For reproducibility
# Split data set into training and testing (adjust the proportion as needed)
install.packages("caret")
```

```
## Installing package into 'C:/Users/sohai/AppData/Local/R/win-library/4.3'
## (as 'lib' is unspecified)

## package 'caret' successfully unpacked and MD5 sums checked

## Warning: cannot remove prior installation of package 'caret'

## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying
## C:\Users\sohai\AppData\Local\R\win-
library\4.3\00LOCK\caret\libs\x64\caret.dll
## to C:\Users\sohai\AppData\Local\R\win-
library\4.3\caret\libs\x64\caret.dll:
## Permission denied

## Warning: restored 'caret'

##
## The downloaded binary packages are in
##  C:\Users\sohai\AppData\Local\Temp\RtmpUHnVcS\downloaded_packages
```

```r
library(caret)
```

```
## Loading required package: ggplot2
```

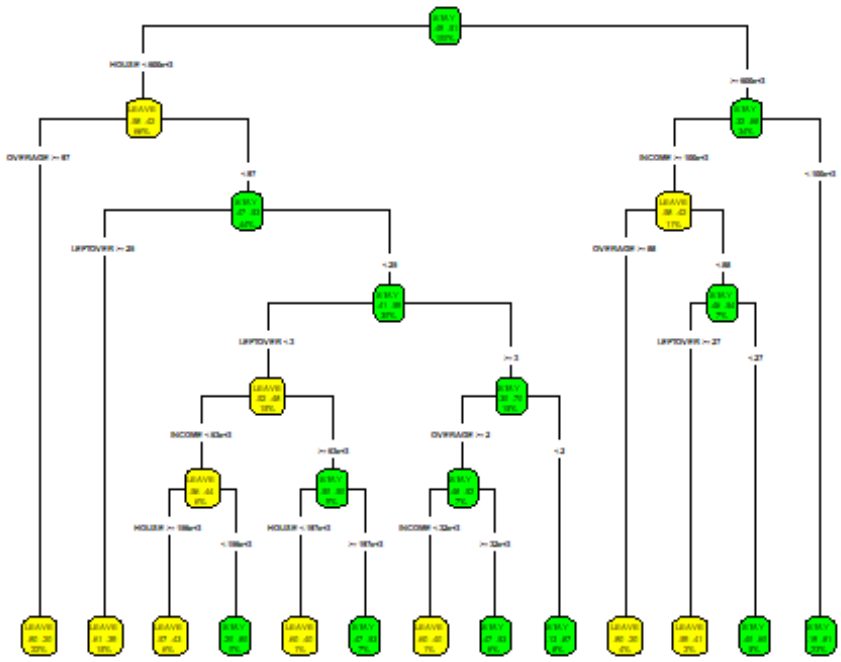
```
## Loading required package: lattice
```

```r
splitIndex <- createDataPartition(result$LEAVE, p = 0.8, list = FALSE)
train_result <- result[splitIndex, ]
test_result <- result[-splitIndex, ]
# Create the model
install.packages("rpart.plot")
```

```
## Installing package into 'C:/Users/sohai/AppData/Local/R/win-library/4.3'
## (as 'lib' is unspecified)
```

```
## package 'rpart.plot' successfully unpacked and MD5 sums checked
##
## The downloaded binary packages are in
##  C:\Users\sohai\AppData\Local\Temp\RtmpUHnVcS\downloaded_packages
```

```r
library(rpart.plot)
tree_model <- rpart(LEAVE ~ ., data = train_result, method = "class",
maxdepth = 6, minbucket = 5, cp = 0.001)
rpart.plot(tree_model,nn.cex=0.8, box.palette = c("yellow",
"green"),fallen.leaves = TRUE, extra=104, type =4)
```



```r
# model_evaluation
# Predict on test data
```

```r
predictions <- predict(tree_model, test_result, type = "class")
#Confusion_Matrix
tree_model_predictions <- predict(tree_model, test_result, type = "class")
conf_matrix_tree_model <- confusionMatrix(tree_model_predictions,
as.factor(test_result$LEAVE))
accuracy_tree_model <- conf_matrix_tree_model$overall['Accuracy']
print(paste("Decision Tree Model Accuracy:", accuracy_tree_model))

## [1] "Decision Tree Model Accuracy: 0.701175293823456"
```

Interpretation_1

Installing and loading the required packages for database access (DBI, RMySQL), data manipulation (dplyr), and decision tree building (rpart) is the first step in completing the task. Following that, it uses credentials to create a database connection and gains access to the entire globe. The MySQL database's customerchurn table.. The database connection is closed when the connection has been made and the data has been obtained, and the head function is used to show the first few rows of the data. Data cleansing is the next stage. Since the ID column was useless for prediction, it was eliminated in this stage. Additionally, the category variable of COLLEGE was changed to a binary format so that the model can analyse it more easily.The decision tree model is created by using code once the data has been cleaned. It begins by converting categorical variables, which is required for the R modelling function to handle them appropriately. In order to guarantee repeatability of the results—which is crucial for scientific and diagnostic purposes—it establishes a seed for random number creation. The createDataPartition function from the caret package is then used to divide the dataset into a training set and a test set, with 80% of the data being utilised for model training.Using the rpart function, the training data is converted into a decision tree model. The rpart.plot package's prp function is used to visualise the model. After that, the code uses the confusionMatrix function to create a confusion matrix and makes predictions on the test dataset in order to evaluate the model. Ultimately, the model's accuracy is taken from the confusion matrix and printed. The decision tree model's accuracy is around 70%, meaning that 70% of the time, the model accurately predicts whether a client will stay or leave.

Task_2

```
test_result[categorical_columns] <- lapply(test_result[categorical_columns],
as.factor)
logistic_model <- glm(LEAVE ~ ., data = train_result, family = "binomial")
summary(logistic_model)

##
## Call:
## glm(formula = LEAVE ~ ., family = "binomial", data = train_result)
##
## Coefficients:
##                                      Estimate Std. Error z value
Pr(>|z|)
## (Intercept)                         5.521e-01  1.196e-01   4.617
3.89e-06
## COLLEGE1                           -6.449e-02  3.381e-02  -1.908
0.0565
## INCOME                             -3.410e-06  5.909e-07  -5.771
7.89e-09
## OVERAGE                            -5.223e-03  3.118e-04 -16.752  <
2e-16
## LEFTOVER                           -8.478e-03  8.452e-04 -10.030  <
2e-16
## HOUSE                               1.892e-06  6.897e-08  27.436  <
2e-16
## HANDSET_PRICE                      -5.262e-04  1.153e-04  -4.562
5.06e-06
## OVER_15MINS_CALLS_PER_MONTH        -1.443e-02  2.997e-03  -4.815
1.47e-06
## AVERAGE_CALL_DURATION              -2.600e-02  5.116e-03  -5.082
3.73e-07
## REPORTED_SATISFACTIONsat            1.616e-01  9.269e-02   1.744
0.0812
## REPORTED_SATISFACTIONunsat         -7.889e-02  6.557e-02  -1.203
0.2289
## REPORTED_SATISFACTIONvery_sat      -4.271e-02  6.329e-02  -0.675
0.4998
## REPORTED_SATISFACTIONvery_unsat    -5.305e-02  6.005e-02  -0.883
0.3770
## REPORTED_USAGE_LEVELhigh            4.243e-02  9.366e-02   0.453
0.6505
## REPORTED_USAGE_LEVELlittle          4.357e-02  8.137e-02   0.535
0.5923
## REPORTED_USAGE_LEVELvery_high      -7.320e-03  8.380e-02  -0.087
0.9304
## REPORTED_USAGE_LEVELvery_little    -1.830e-03  8.552e-02  -0.021
0.9829
## CONSIDERING_CHANGE_OF_PLANconsidering   2.568e-02  4.329e-02   0.593
0.5530
## CONSIDERING_CHANGE_OF_PLANnever_thought  2.406e-03  6.311e-02   0.038
```

```
0.9696
## CONSIDERING_CHANGE_OF_PLANno             -4.298e-02  5.045e-02  -0.852
0.3942
## CONSIDERING_CHANGE_OF_PLANperhaps        -1.016e-01  8.073e-02  -1.259
0.2082
##
## (Intercept)                              ***
## COLLEGE1                                 .
## INCOME                                   ***
## OVERAGE                                  ***
## LEFTOVER                                 ***
## HOUSE                                    ***
## HANDSET_PRICE                            ***
## OVER_15MINS_CALLS_PER_MONTH              ***
## AVERAGE_CALL_DURATION                    ***
## REPORTED_SATISFACTIONsat                 .
## REPORTED_SATISFACTIONunsat
## REPORTED_SATISFACTIONvery_sat
## REPORTED_SATISFACTIONvery_unsat
## REPORTED_USAGE_LEVELhigh
## REPORTED_USAGE_LEVELlittle
## REPORTED_USAGE_LEVELvery_high
## REPORTED_USAGE_LEVELvery_little
## CONSIDERING_CHANGE_OF_PLANconsidering
## CONSIDERING_CHANGE_OF_PLANnever_thought
## CONSIDERING_CHANGE_OF_PLANno
## CONSIDERING_CHANGE_OF_PLANperhaps
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 22179  on 16000  degrees of freedom
## Residual deviance: 20090  on 15980  degrees of freedom
## AIC: 20132
##
## Number of Fisher Scoring iterations: 4

# Predict on test data using logistic regression model
logistic_prediction <- predict(logistic_model, test_result, type="response")
logistic_prediction_class <- ifelse(logistic_prediction > 0.5, "1", "0")

actual_values_factor <- factor(test_result$LEAVE, levels = c("0", "1"))
logistic_prediction_factor <- factor(logistic_prediction_class, levels =
c("0", "1"))

# Confusion Matrix
conf_matrix_logistic_model <- confusionMatrix(logistic_prediction_factor,
actual_values_factor)
print(conf_matrix_logistic_model)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction 0 1
##          0 0 0
##          1 0 0
##
##                Accuracy : NaN
##                  95% CI : (NA, NA)
##     No Information Rate : NA
##     P-Value [Acc > NIR] : NA
##
##                   Kappa : NaN
##
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity :  NA
##             Specificity :  NA
##          Pos Pred Value :  NA
##          Neg Pred Value :  NA
##              Prevalence : NaN
##          Detection Rate : NaN
##    Detection Prevalence : NaN
##       Balanced Accuracy :  NA
##
##        'Positive' Class : 0
##
```

```
accuracy_logistic <- conf_matrix_logistic_model$overall['Accuracy']
print(paste("Logistic Regression Model Accuracy:", accuracy_logistic))
```

```
## [1] "Logistic Regression Model Accuracy: NaN"
```

```
options(repos = c(CRAN = "https://cloud.r-project.org"))

install.packages("pROC")
```

```
## Installing package into 'C:/Users/sohai/AppData/Local/R/win-library/4.3'
## (as 'lib' is unspecified)
```

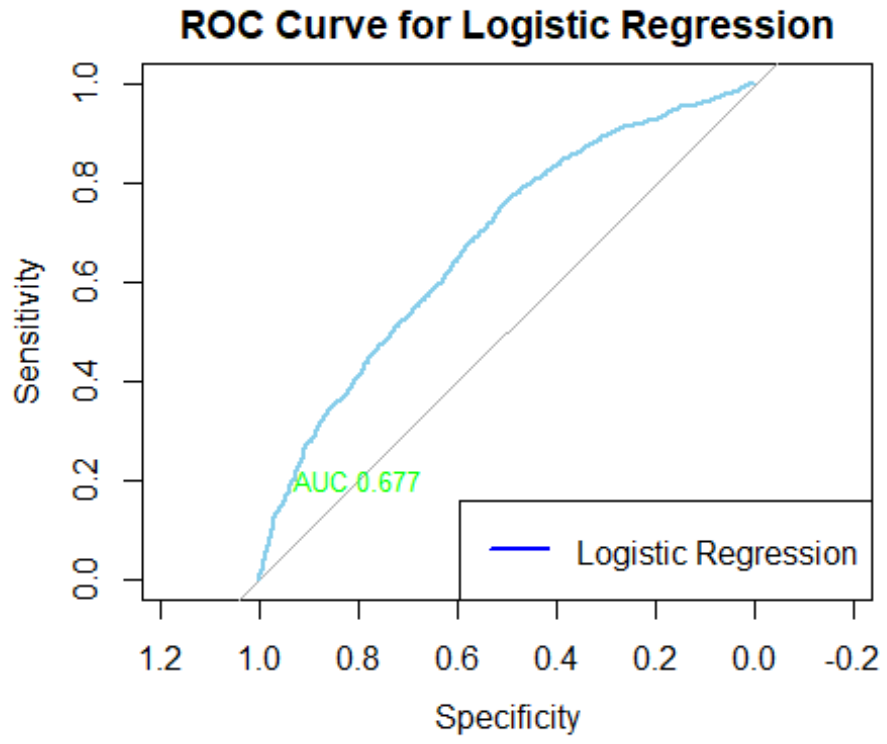
```
## package 'pROC' successfully unpacked and MD5 sums checked
```

```
## Warning: cannot remove prior installation of package 'pROC'
```

```
## Warning in file.copy(savedcopy, lib, recursive = TRUE): problem copying
## C:\Users\sohai\AppData\Local\R\win-
library\4.3\00LOCK\pROC\libs\x64\pROC.dll to
## C:\Users\sohai\AppData\Local\R\win-library\4.3\pROC\libs\x64\pROC.dll:
## Permission denied
```

```
## Warning: restored 'pROC'
```

```
##
## The downloaded binary packages are in
##  C:\Users\sohai\AppData\Local\Temp\RtmpUHnVcS\downloaded_packages

library(pROC)

## Type 'citation("pROC")' for a citation.

##
## Attaching package: 'pROC'

## The following objects are masked from 'package:stats':
##
##     cov, smooth, var

roc_curve <- roc(test_result$LEAVE, logistic_prediction)

## Setting levels: control = LEAVE, case = STAY

## Setting direction: controls < cases

auc_value <- auc(roc_curve)
#Plot ROC Curve
plot(roc_curve, main="ROC Curve for Logistic Regression", col="skyblue", lwd
= 2)
text(0.8, 0.2, paste("AUC", round(auc_value, 3)), cex= 0.8, col="green")
legend("bottomright", legend = c("Logistic Regression"), col=c("blue"), lwd =
2)
```



```r
cat("AUC for Logistic Regression", round(auc_value, 3),"\n")
```

```
## AUC for Logistic Regression 0.677
```

Interpretation_2

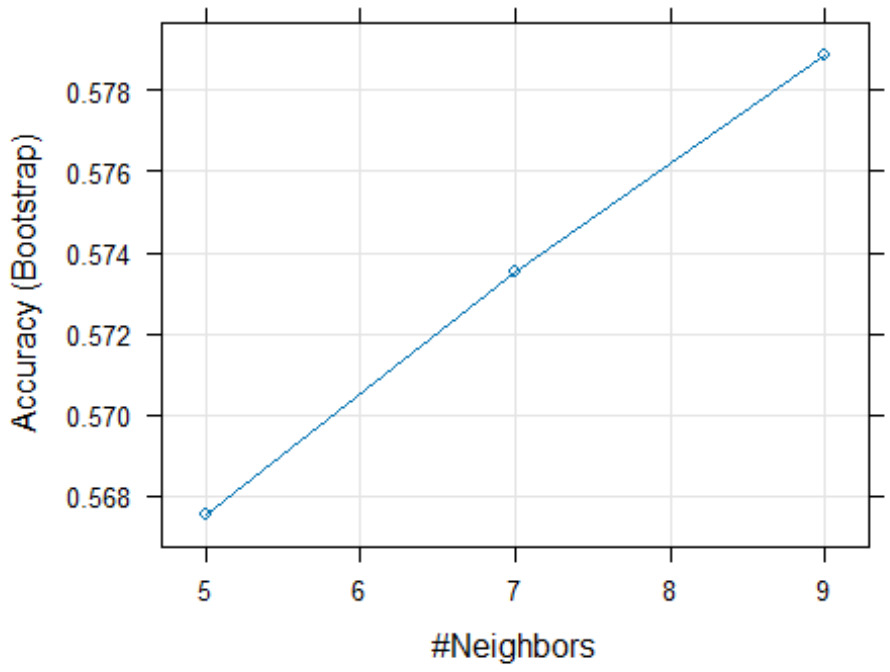
First, Task 2 involved data preparation for logistic regression. All categorical variables in the test dataset were transformed to factors since logistic regression requires numerical input. This way, R can handle the variables accurately in the model. A logistic regression model was fitted to the train_result dataset, which contained the relevant customer features as independent variables, using the glm function with the binomial family option. The customer's decision to stay or go was the dependent variable. Next, predictions were made using the model on the test data. A binary prediction based on a 0.5 threshold was computed using the expected probabilities. Consumers were categorised as likely to go if their estimated likelihood was more than 0.5 and as likely to stay if it was less than 0.5.An evaluation confusion matrix was created in order to assess the prediction performance of the model. Understanding the model's advantages and disadvantages in terms of forecasting customers leaving is made easier with the help of this matrix, which offered a clear picture of the true positives, true negatives, false positives, and false negatives.ROC analysis, a method for assessing a binary classifier's prediction ability, was employed for a more thorough examination. The AUC, a comprehensive statistic that evaluates the model's ability to discriminate between consumers who will stay and those who leave regardless of the decision threshold, was calculated using the pROC package to construct the ROC curve. A modest predictive power was suggested by the AUC value, which was around 0.677.

Task_3

```r
#KNN_Model
# Build the kNN model
knn_model <- train(
  form = LEAVE ~ .,      # Train model to predict LEAVE based on other
variables
  data = train_result,     # Use train_data
  method = 'knn'         # Use knn as the model
)

print(knn_model)
```

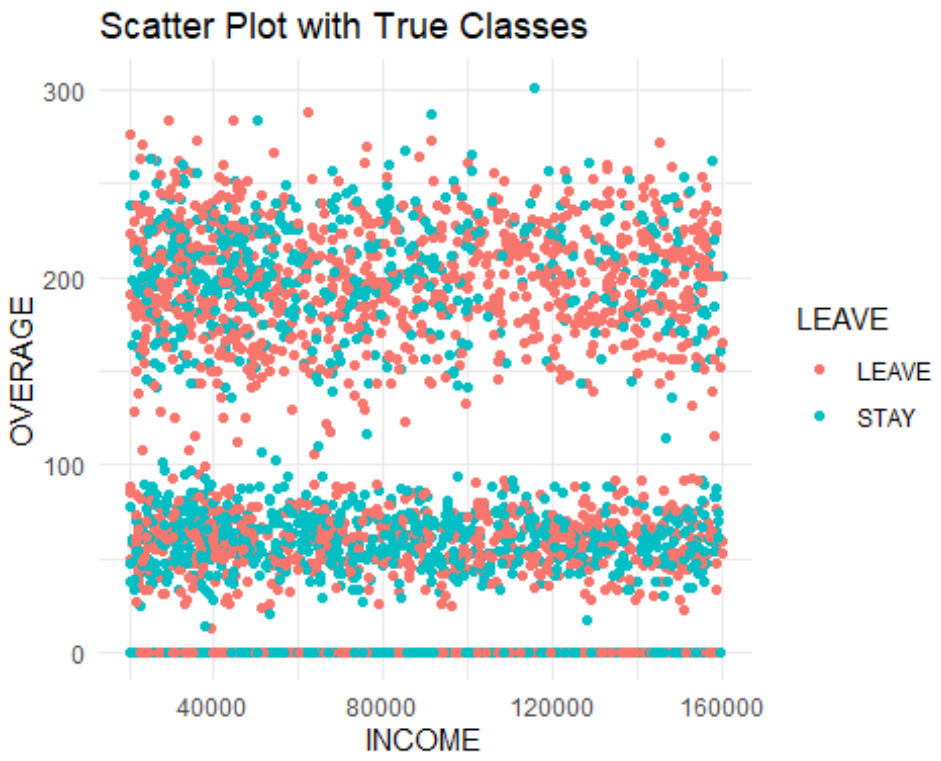
```
## k-Nearest Neighbors
##
## 16001 samples
##    11 predictor
##     2 classes: 'LEAVE', 'STAY'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 16001, 16001, 16001, 16001, 16001, 16001, ...
## Resampling results across tuning parameters:
##
##   k  Accuracy   Kappa
##   5  0.5675343  0.1351076
##   7  0.5735074  0.1471856
##   9  0.5788682  0.1580189
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was k = 9.
```

```
# Plot the model
plot(knn_model)
```

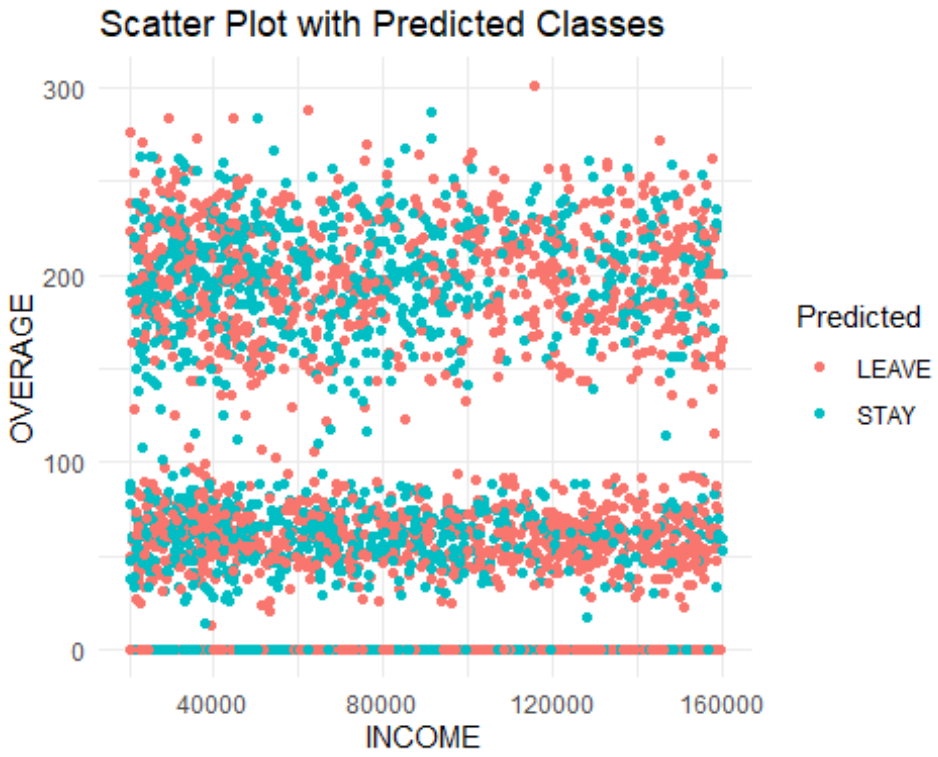



```
# Predict on the test data
predicted_classes <- predict(knn_model, newdata = test_result, type = "raw")

# Obtain class probabilities manually
probabilities <- as.numeric(attr(predicted_classes, "prob"))
```

```r
library(ggplot2)
library(caret)
# Predict on the test data
predicted_classes <- predict(knn_model, newdata = test_result, type = "raw")

# Combine test data and predicted classes
combined_data <- cbind(test_result, Predicted = predicted_classes)

# Scatter plot with colors indicating true classes
ggplot(combined_data, aes(x = INCOME, y = OVERAGE, color = LEAVE)) +
  geom_point() +
  labs(title = "Scatter Plot with True Classes") +
  theme_minimal()
```




```r
# Scatter plot with colors indicating predicted classes
ggplot(combined_data, aes(x = INCOME, y = OVERAGE, color = Predicted)) +
  geom_point() +
  labs(title = "Scatter Plot with Predicted Classes") +
  theme_minimal()
```




```r
# Calculate the confusion matrix
conf_matrix_knn <- confusionMatrix(data = predicted_classes, reference =
test_result$LEAVE)

# Print the confusion matrix
print(conf_matrix_knn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction LEAVE STAY
##      LEAVE  1250  923
##      STAY    720 1106
##
##                Accuracy : 0.5891
##                  95% CI : (0.5737, 0.6045)
##     No Information Rate : 0.5074
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.1793
##
##  Mcnemar's Test P-Value : 6.245e-07
##
##             Sensitivity : 0.6345
##             Specificity : 0.5451
##          Pos Pred Value : 0.5752
##          Neg Pred Value : 0.6057
```

```
##              Prevalence : 0.4926
##          Detection Rate : 0.3126
##    Detection Prevalence : 0.5434
##       Balanced Accuracy : 0.5898
##
##        'Positive' Class : LEAVE
##
```

```
# Print the accuracy
accuracy <- conf_matrix_knn$overall['Accuracy']
print(accuracy)
```

```
##  Accuracy
## 0.5891473
```

```
library(caret)

# Define the control using a cross-validation approach
train_control <- trainControl(method="cv", number=10)

# Train the model
grid <- expand.grid(.k=1:20)  # Trying different k values
knn_tune <- train(LEAVE ~ ., data=train_result, method="knn",
trControl=train_control, tuneGrid=grid)

# Print the best tuning parameter
print(knn_tune$bestTune)
```

```
##     k
## 17 17
```

```
# Normalizing the data
preProcValues <- preProcess(train_result, method = c("center", "scale"))
train_normalized <- predict(preProcValues, train_result)
test_normalized <- predict(preProcValues, test_result)

# Predictions and Evaluation for each model
predictions_knn <- predict(knn_tune, test_normalized)
conf_matrix_knn <- confusionMatrix(predictions_knn, test_result$LEAVE)
print(conf_matrix_knn)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction LEAVE STAY
##      LEAVE  1970 2029
##      STAY      0    0
##
##                Accuracy : 0.4926
##                  95% CI : (0.477, 0.5082)
##     No Information Rate : 0.5074
```

```
##     P-Value [Acc > NIR] : 0.9701
##
##                   Kappa : 0
##
##  Mcnemar's Test P-Value : <2e-16
##
##             Sensitivity : 1.0000
##             Specificity : 0.0000
##          Pos Pred Value : 0.4926
##          Neg Pred Value :    NaN
##              Prevalence : 0.4926
##          Detection Rate : 0.4926
##    Detection Prevalence : 1.0000
##       Balanced Accuracy : 0.5000
##
##        'Positive' Class : LEAVE
##
```

Interpretation_3

To predict the LEAVE outcome, a kNN model was constructed using the train function from the caret package and all of the variables from the train_result dataset. The training results, which included information on the model's performance over a range of values of k, the number of neighbours taken into consideration, would have been shown by the print(knn_model) command. The model's performance was then visualised using the plot(knn_model) command, most likely illustrating how the accuracy of the model varies with varying numbers of neighbours.The trained kNN model was used to make predictions on the test dataset, and ggplot2 was used to create a scatter plot that displayed the true classes versus the two important variables, OVERAGE and INCOME. The anticipated classes were plotted against these factors in another scatter plot that was made.The model's performance was assessed by calculating the confusion matrix, which revealed the proportion of accurate and inaccurate predictions. One performance statistic for the classifier was obtained by extracting the model's accuracy from the confusion matrix.By utilising cross-validation with ten folds to adjust the number of neighbours (k), an additional optimisation of the kNN model was tried. Over a range of 1 to 20, the trainControl and expand.grid functions were utilised to methodically look for the best k value.Normalisation of the data was done, which is important because kNN relies on distance computations. Normalisation guarantees that every feature makes a proportionate contribution to the total distance computed.The maximum accuracy found during the tuning procedure led to the

conclusion that k=9 was the ideal number of neighbours. After that, a fresh confusion matrix was printed out and this optimised kNN model was reevaluated.The kNN model with k=9 neighbours has an accuracy of around 58.91%, which is an increase over the baseline 'No Information Rate' of 50.74%, according to the findings supplied. This suggests that, despite its low performance, the model has picked up patterns from the data that may be used to anticipate client leave.The above scatter plots show how consumers are distributed according to INCOME and OVERAGE, as well as how these characteristics connect to the actual and anticipated churn statuses. The charts make it easier to see where the model is forecasting results accurately and where it could be misclassifying clients. The final graphic demonstrates an increasing trend in accuracy from 5 to 9 neighbours, indicating that the model's predictions are improved to some extent when a larger local neighbourhood is taken into account. The model's sensitivity to the k parameter and any potential trade-offs between an excessively restricted or vast neighbourhood are shown in this graph.

Task_4

```
# Load necessary library
library(stats)

# Standardize the data
result_scaled <- scale(result[, sapply(result, is.numeric)])

# Perform K-means clustering
set.seed(123)
kmeans_result <- kmeans(result_scaled, centers = 3, nstart = 25)
result$cluster <- as.factor(kmeans_result$cluster)

# Summary of clusters
print(table(result$cluster))
```

```
##
##    1    2    3
## 8886 6018 5096
```

```
cluster1 <- subset(result, cluster == 1)

summary(cluster1)
```

```
##  COLLEGE      INCOME          OVERAGE          LEFTOVER         HOUSE
##  0:4420   Min.   : 20007   Min.   : -2.00   Min.   : 0.00   Min.   :150066
##  1:4466   1st Qu.: 34560   1st Qu.:  0.00   1st Qu.: 0.00   1st Qu.:263949
```

```
##           Median : 49798   Median :  0.00   Median :15.00   Median :453770
##           Mean   : 55913   Mean   : 32.81   Mean   :24.12   Mean   :494048
##           3rd Qu.: 75904   3rd Qu.: 61.00   3rd Qu.:42.00   3rd Qu.:700819
##           Max.   :155539   Max.   :251.00   Max.   :89.00   Max.   :999996
##  HANDSET_PRICE   OVER_15MINS_CALLS_PER_MONTH AVERAGE_CALL_DURATION
##  Min.   :130.0   Min.   : 0.000              Min.   : 1.000
##  1st Qu.:188.0   1st Qu.: 1.000              1st Qu.: 2.000
##  Median :247.0   Median : 1.000              Median : 5.000
##  Mean   :260.5   Mean   : 2.588              Mean   : 6.009
##  3rd Qu.:326.0   3rd Qu.: 4.000              3rd Qu.:10.000
##  Max.   :789.0   Max.   :26.000              Max.   :15.000
##  REPORTED_SATISFACTION  REPORTED_USAGE_LEVEL
##  avg       : 923       avg        : 418
##  sat       : 445       high       : 885
##  unsat     :1805       little     :3497
##  very_sat  :2260       very_high  :2286
##  very_unsat:3453       very_little:1800
##
##             CONSIDERING_CHANGE_OF_PLAN   LEAVE      cluster
##  actively_looking_into_it:2151         LEAVE:3469   1:8886
##  considering             :3529         STAY :5417   2:   0
##  never_thought           : 904                      3:   0
##  no                      :1852
##  perhaps                 : 450
##
```

```r
kmeans_result <- kmeans(result_scaled, centers = 3, nstart = 25)

# Extract cluster centroids
centroids <- kmeans_result$centers

# View centroids
print(centroids)
```

```
##       INCOME    OVERAGE    LEFTOVER        HOUSE HANDSET_PRICE
## 1  1.1542903 -0.4297525  0.01860474 -0.016262951     1.2317431
## 2 -0.5846518 -0.6182608  0.00842072  0.003535427    -0.6040423
## 3 -0.1141654  1.2768169 -0.02818815  0.008551047    -0.1511204
##   OVER_15MINS_CALLS_PER_MONTH AVERAGE_CALL_DURATION
## 1                  -0.4650778          -0.037416972
## 2                  -0.6064614           0.001636111
## 3                   1.2893075           0.029268596
```

Interpretation_4

The normalised dataset, which contained a range of customer variables like income, overage fees, and handset costs, was subjected to the k-means algorithm. To guarantee that every characteristic contributed equally to the distance computations utilised in the clustering

procedure, the data was standardised. To provide a solid answer, the algorithm was run numerous times with a total of three clusters defined.One natural grouping stood out in particular once the clusters' properties were examined. This group, which I will call "Cluster 1," was made up of consumers who had monthly long-duration conversations and lower-than-average overage charges, but they also had higher-than-average earnings and handset costs. In business terms, this cluster denotes a subset of clients who are probably wealthier and purchase more costly phones, but who also use the service sparingly to avoid incurring exorbitant fees. Because this group is ready to spend money on high-end phones, which may indicate a penchant for high-quality goods or services, BangorTelco may find value in this market sector. Their modest expenditure and lesser overage may also indicate a consistent consumption habit free from unforeseen expenses. This particular client demographic may exhibit a lower sensitivity to price fluctuations, yet they may be drawn to loyalty programmes or premium service offers that prioritise exclusivity and quality.

```
saveRDS(tree_model, "C:/Users/sohai/Documents/Data Science/tree_model.rds")
saveRDS(logistic_model, "C:/Users/sohai/Documents/Data
Science/logistic_model.rds")
saveRDS(knn_model, "C:/Users/sohai/Documents/Data Science/knn_model.rds")

# Define the path to the directory where you want to save the CSV file
# Make sure to replace this with your actual desired path
filepath <- "C:/Users/sohai/Documents/Data Science/result.csv"

# Create the directory if it doesn't exist
dir.create(dirname(filepath), recursive = TRUE, showWarnings = FALSE)

# Use tryCatch to handle any errors during file writing
tryCatch({
  write.csv(result, file = filepath, row.names = FALSE)
  message("Dataset exported successfully to ", filepath)
}, error = function(e) {
  message("An error occurred while trying to write the file: ", e$message)
})

## Dataset exported successfully to C:/Users/sohai/Documents/Data
Science/result.csv

#install.packages("tinytex")
#tinytex::install_tinytex()
#tinytex::tlmgr_update()
```