Data Science Report

Rufus Kolawole Asake

2025-04-30

Task 1

Job Details

Job Title: Business Analyst
Company: Decathlon UK
Location: London SE16 (Hybrid)

Job Description:
Decathlon UK is seeking a proactive Business Analyst to join our team. The successful candidate will handle ad-hoc and recurring data requests from different business teams, work with technical teams to integrate new data sources for business value, and support various departments in making data-driven decisions. The ideal candidate will have experience in data analysis, strong communication skills, and the ability to work collaboratively in a hybrid work environment.

Cover Letter

Rufus Kolawole Asake
37 Caellepa
Bangor

Phone: 03457 125 563

10-Mar-25

Hiring Manager
Decathlon UK
London SE16

Dear Hiring Manager,

I am writing to express my interest in the Business Analyst position at Decathlon UK, as advertised on Indeed. With a strong background in data analysis and a passion for leveraging data to drive business solutions, I am confident in my ability to contribute effectively to your team.

In my previous role at Selected Intervention Twickenham, I was responsible for handling both ad-hoc and recurring data requests from various business units. By employing data visualization tools such as Power BI and Tableau, I translated complex datasets into actionable insights, facilitating informed decision-making across departments. My ability to work closely with technical teams to integrate new data sources aligns with the core responsibilities outlined in the job description.

I have a proven track record of successful project support, having collaborated with cross-functional teams to implement data-driven solutions that enhance operational efficiency. My experience in supporting the development, testing, and deployment of data integration projects has equipped me with a comprehensive understanding of the data lifecycle, which I am eager to bring to Decathlon UK.

What excites me about this opportunity is the chance to work in a hybrid environment at Decathlon UK, which I believe fosters a collaborative and flexible setting, essential for innovative problem-solving.

I am enthusiastic about the prospect of joining Decathlon UK and contributing to the success of your business initiatives. Thank you for considering my application. I look forward to the opportunity to discuss how my skills and experiences align with the needs of your team.

Sincerely,
Rufus Kolawole Asake

Task 2: Decision Tree Model

```
# Load the libraries
library(tidyverse)
library(rpart)
library(rpart.plot)
library(DBI)
library(RMySQL)
library(class)
library(caret)
```

```
# Define database connection credentials
USER <- 'root'
PASSWORD <- 'Bangor@123'
HOST <- 'localhost'
DBNAME <- 'world'
PORT <- 3306

# Connect to MySQL
db <- dbConnect(RMySQL::MySQL(),
                dbname = DBNAME,
                host = HOST,
                user = USER,
                password = PASSWORD,
                port = PORT)

# Fetch the dataset from MySQL
df <- dbGetQuery(db, "SELECT * FROM world.customerchurn")

# Close the database connection
dbDisconnect(db)
```

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

```
# View basic information about the dataset
str(df)
```

```
## 'data.frame':    22141 obs. of  10 variables:
##  $ ID               : int  11000 11001 11002 11003 11004 11005 11006 11007 11008 11009 ...
##  $ Year_Birth       : int  1969 1963 1951 1979 1969 1981 1955 1989 1983 1981 ...
##  $ Education        : chr  "Graduation" "PhD" "Master" "Graduation" ...
##  $ MaritalStatus    : chr  "Together" "Single" "Married" "Single" ...
##  $ Income           : int  23228 48918 67381 61825 44078 41967 75261 28691 24072 19414 ...
##  $ Recency          : int  71 21 67 56 17 66 17 56 79 32 ...
##  $ NumWebPurchases  : int  2 1 2 4 2 1 5 1 1 1 ...
##  $ NumStorePurchases: int  3 4 9 8 3 3 5 3 2 3 ...
##  $ NumWebVisitsMonth: int  8 4 7 4 5 4 2 8 8 8 ...
##  $ Response         : int  0 0 0 0 0 0 1 0 0 0 ...
```

```
summary(df)
```

```
##        ID          Year_Birth    Education          MaritalStatus
##  Min.   :11000   Min.   :1893   Length:22141       Length:22141
##  1st Qu.:16592   1st Qu.:1959   Class :character   Class :character
##  Median :22197   Median :1970   Mode  :character   Mode  :character
##  Mean   :22198   Mean   :1969
##  3rd Qu.:27799   3rd Qu.:1978
##  Max.   :33399   Max.   :1996
##      Income          Recency      NumWebPurchases  NumStorePurchases
##  Min.   :  1730   Min.   : 0.00   Min.   : 0.000   Min.   : 0.000
##  1st Qu.: 35441   1st Qu.:24.00   1st Qu.: 2.000   1st Qu.: 3.000
##  Median : 51529   Median :49.00   Median : 4.000   Median : 5.000
##  Mean   : 52514   Mean   :48.78   Mean   : 4.103   Mean   : 5.801
##  3rd Qu.: 68682   3rd Qu.:73.00   3rd Qu.: 6.000   3rd Qu.: 8.000
##  Max.   :666666   Max.   :99.00   Max.   :27.000   Max.   :13.000
##  NumWebVisitsMonth    Response
##  Min.   : 0.000    Min.   :0.0000
##  1st Qu.: 3.000    1st Qu.:0.0000
##  Median : 6.000    Median :0.0000
##  Mean   : 5.317    Mean   :0.1532
##  3rd Qu.: 7.000    3rd Qu.:0.0000
##  Max.   :20.000    Max.   :1.0000
```

```r
head(df)
```

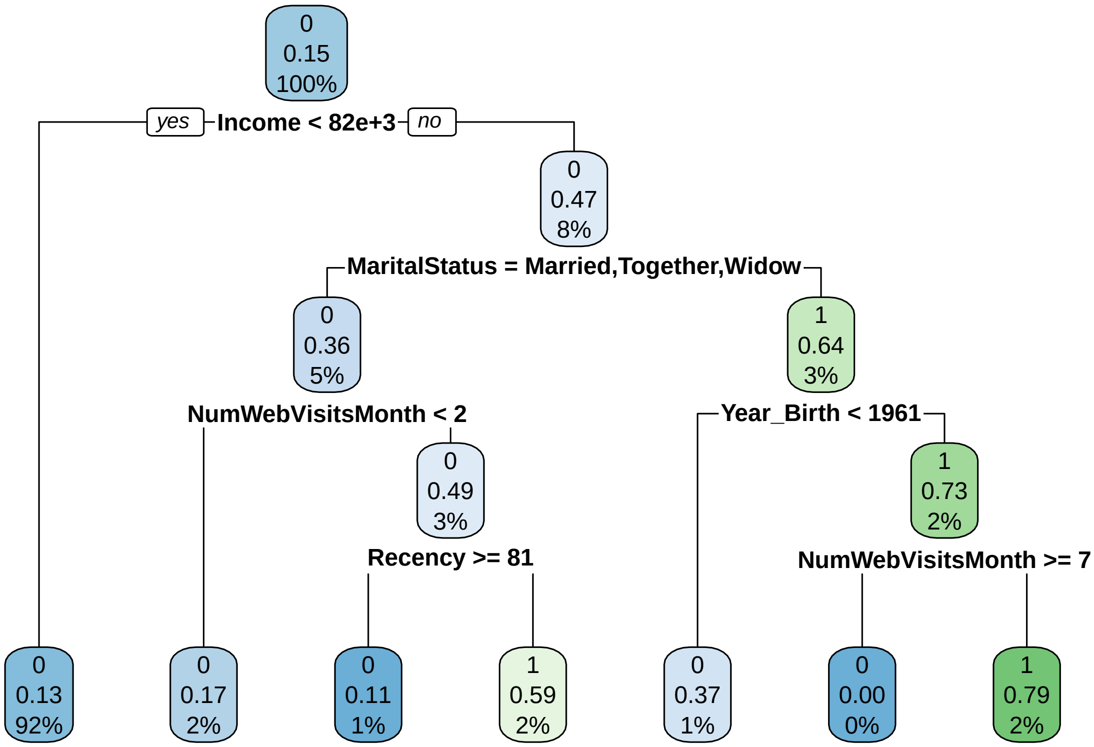
```
##      ID Year_Birth  Education MaritalStatus Income Recency NumWebPurchases
## 1 11000       1969 Graduation      Together  23228      71               2
## 2 11001       1963        PhD        Single  48918      21               1
## 3 11002       1951     Master       Married  67381      67               2
## 4 11003       1979 Graduation        Single  61825      56               4
## 5 11004       1969 Graduation       Married  44078      17               2
## 6 11005       1981 Graduation        Single  41967      66               1
##   NumStorePurchases NumWebVisitsMonth Response
## 1                 3                 8        0
## 2                 4                 4        0
## 3                 9                 7        0
## 4                 8                 4        0
## 5                 3                 5        0
## 6                 3                 4        0
```

```r
# Remove invalid birth years (e.g., before 1900)
df <- df %>% filter(Year_Birth >= 1900)

# Handle missing values in Income by replacing with the median value
df$Income[is.na(df$Income)] <- median(df$Income, na.rm = TRUE)

# Convert categorical variables to factors
df$Education <- as.factor(df$Education)
df$MaritalStatus <- as.factor(df$MaritalStatus)

# View cleaned dataset summary
summary(df)
```

```
##        ID          Year_Birth        Education       MaritalStatus
```

```
##  Min.   :11000   Min.   :1900   2n Cycle  : 1974   Married :8547
##  1st Qu.:16590   1st Qu.:1959   Basic     :  518   Together:5735
##  Median :22195   Median :1970   Graduation:11150   Single  :4755
##  Mean   :22196   Mean   :1969   Master    : 3717   Divorced:2229
##  3rd Qu.:27797   3rd Qu.:1978   PhD       : 4760   Widow   : 771
##  Max.   :33399   Max.   :1996                      Alone   :  38
##                                                    (Other) :  44
##      Income          Recency      NumWebPurchases  NumStorePurchases
##  Min.   :  1730   Min.   : 0.00   Min.   : 0.000   Min.   : 0.000
##  1st Qu.: 35441   1st Qu.:24.00   1st Qu.: 2.000   1st Qu.: 3.000
##  Median : 51518   Median :49.00   Median : 4.000   Median : 5.000
##  Mean   : 52492   Mean   :48.79   Mean   : 4.104   Mean   : 5.803
##  3rd Qu.: 68657   3rd Qu.:73.00   3rd Qu.: 6.000   3rd Qu.: 8.000
##  Max.   :666666   Max.   :99.00   Max.   :27.000   Max.   :13.000
##
##  NumWebVisitsMonth    Response
##  Min.   : 0.00     Min.   :0.0000
##  1st Qu.: 3.00     1st Qu.:0.0000
##  Median : 6.00     Median :0.0000
##  Mean   : 5.32     Mean   :0.1534
##  3rd Qu.: 7.00     3rd Qu.:0.0000
##  Max.   :20.00     Max.   :1.0000
##
```

```r
# Split dataset into training (80%) and testing (20%)
set.seed(123)  # For reproducibility
train_index <- sample(seq_len(nrow(df)), size = 0.8 * nrow(df))
train_data <- df[train_index, ]
test_data <- df[-train_index, ]

# Train the Decision Tree model
tree_model <- rpart(Response ~ ., data = train_data, method = "class")

# Visualize the Decision Tree
rpart.plot(tree_model)
```




```
# Make predictions on test set
predictions <- predict(tree_model, test_data, type = "class")

# Confusion Matrix
conf_matrix <- table(test_data$Response, predictions)
print(conf_matrix)
```

```
##    predictions
##        0    1
##   0 3689   68
##   1  528  139
```

```
# Calculate Accuracy
accuracy <- sum(diag(conf_matrix)) / sum(conf_matrix)
print(paste("Model Accuracy:", round(accuracy * 100, 2), "%"))
```

```
## [1] "Model Accuracy: 86.53 %"
```

```
# Save the trained model for Power BI
saveRDS(tree_model, "tree_model.rds")

# Make predictions on the entire dataset
df$Tree_Prediction <- predict(tree_model, df, type = "class")
```

```r
# Save predictions as CSV for Power BI
write.csv(df, "decision_tree_predictions.csv", row.names = FALSE)
```

Task 3: K-Nearest Neighbors (KNN)

```r
# Normalize numeric variables for KNN
df_norm <- df %>%
  mutate(across(c(Year_Birth, Income, Recency, NumWebPurchases, NumStorePurchases, NumWebVisitsMonth),
                ~ (.-min(.))/(max(.)-min(.))))

# Remove rows with missing values
df_norm <- na.omit(df_norm)

# Set seed for reproducibility
set.seed(123)

# Split dataset into training (80%) and testing (20%)
train_index <- sample(seq_len(nrow(df_norm)), size = 0.8 * nrow(df_norm))
train_data <- df_norm[train_index, ]
test_data <- df_norm[-train_index, ]

# Define predictor and target variables (remove Response from predictors)
train_x <- train_data %>% select(-Response) %>% select_if(is.numeric)
test_x <- test_data %>% select(-Response) %>% select_if(is.numeric)
train_y <- as.factor(train_data$Response)
test_y <- as.factor(test_data$Response)

# Convert predictor variables to matrices for KNN
train_x <- as.matrix(na.omit(train_x))
test_x <- as.matrix(na.omit(test_x))

# Train KNN model
knn_model <- knn(train = train_x, test = test_x, cl = train_y, k = 5)

# Save predictions in test_data (NOT df)
test_data$KNN_Prediction <- knn_model

# Generate Decision Tree Predictions
test_data$Tree_Prediction <- predict(tree_model, test_data, type = "class")


# Ensure ID column is present in test_data
test_data$Response <- as.factor(test_data$Response)
test_data$Tree_Prediction <- as.factor(test_data$Tree_Prediction)
test_data$KNN_Prediction <- as.factor(test_data$KNN_Prediction)

# Confusion Matrix
conf_matrix_knn <- table(test_y, test_data$KNN_Prediction)
conf_matrix_knn
```

```
##
```

```
## test_y    0    1
##      0 3660   97
##      1  648   19
```

```r
# Calculate Accuracy
accuracy_knn <- sum(diag(conf_matrix_knn)) / sum(conf_matrix_knn)
paste("KNN Model Accuracy:", round(accuracy_knn * 100, 2), "%")
```

```
## [1] "KNN Model Accuracy: 83.16 %"
```

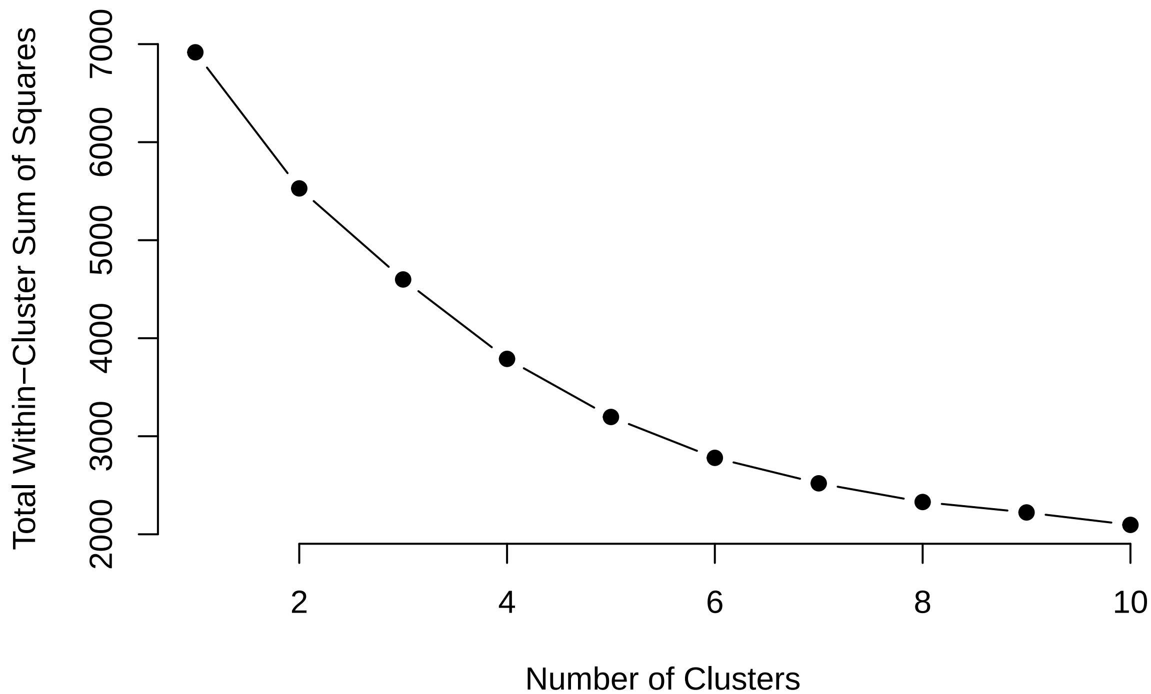
```r
# Save updated test_data with predictions for Power BI
write.csv(test_data, "knn_predictions.csv", row.names = FALSE)

# Save the trained model
saveRDS(knn_model, "knn_model.rds")
```

Task 4: Clustering

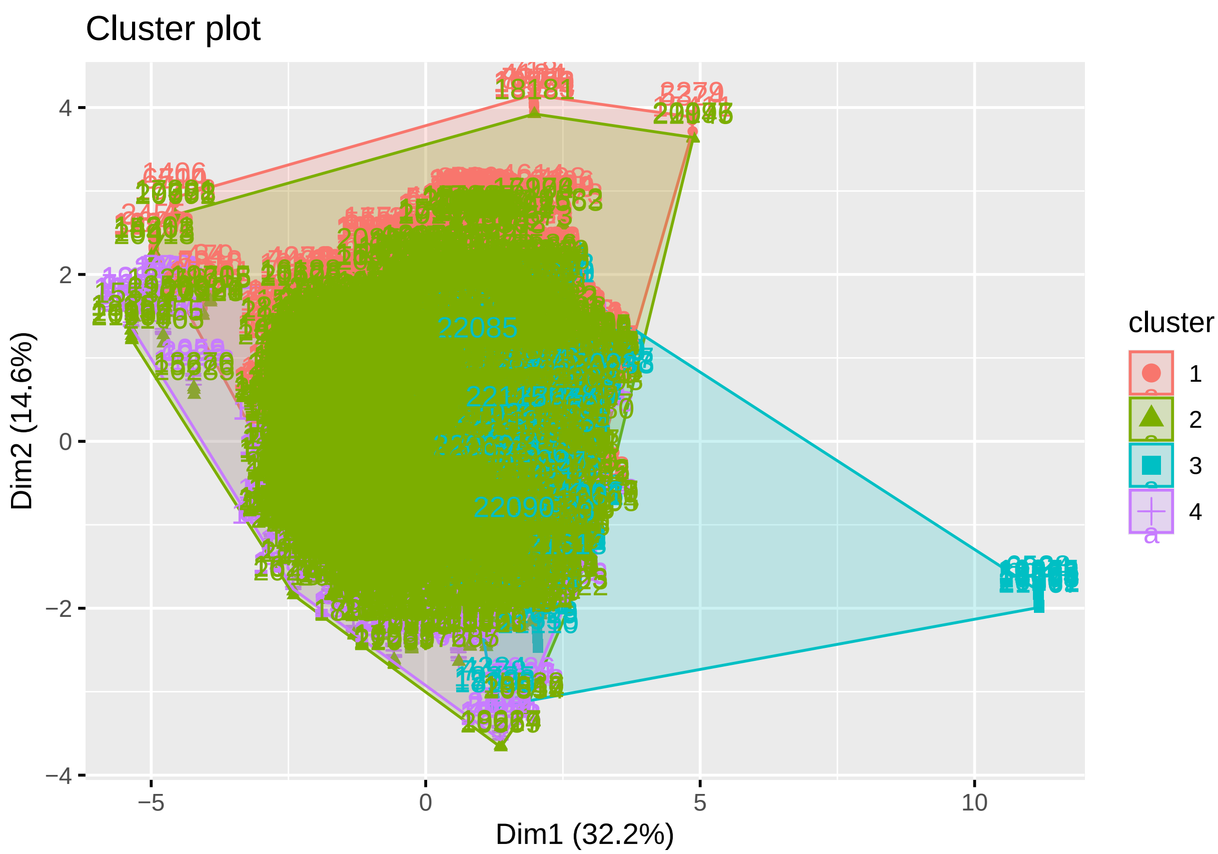
```r
# Load clustering libraries
library(factoextra)
```

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

```r
library(cluster)

# Prepare dataset for clustering
df_cluster <- df %>%
  select(-Education, -MaritalStatus, -Response) %>%
  mutate(across(where(is.numeric), ~ (.-min(.))/(max(.)-min(.))))

df_cluster <- na.omit(df_cluster)

# Determine the optimal number of clusters using the Elbow Method
set.seed(123)
wss <- function(k) {
  kmeans(df_cluster, k, nstart = 10)$tot.withinss
}
k_values <- 1:10
wss_values <- map_dbl(k_values, wss)

# Plot the Elbow Method graph
plot(k_values, wss_values, type = "b", pch = 19, frame = FALSE,
     xlab = "Number of Clusters", ylab = "Total Within-Cluster Sum of Squares")
```




```r
# Train K-Means clustering model
optimal_k <- 4  # Adjust this based on the Elbow plot
set.seed(123)
kmeans_model <- kmeans(df_cluster, centers = optimal_k, nstart = 10)

df_cluster$Cluster <- as.factor(kmeans_model$cluster)

# Visualize clusters
fviz_cluster(kmeans_model, data = df_cluster %>% select_if(is.numeric))
```




```r
# Save clustered dataset
write.csv(df_cluster, "clustered_customers.csv", row.names = FALSE)
```

Summary and Findings

- Decision Tree achieved an accuracy of 86.53%.
- KNN model trained with k=5 and achieved an accuracy of 83.16%..
- Cluster analysis identified 4 clusters.