HarvardX PH125.9x: Capstone Project 2 Online Shopper's Purchasing Intention Model Comparison

Dr Luis Satch

2024-09-28

Table of Contents

Introduction

In the context of this capstone project, predicting online shoppers' purchasing intentions presents an opportunity to apply machine learning techniques to a practical and widely relevant problem. Being a significant part of the modern economy, e-commerce offers an interesting case study due to the availability of detailed user behaviour data

This project aims to predict whether an online shopping session will result in a purchase by comparing two machine learning models: a Multilayer Perceptron (MLP) and a Random Forest classifier. By evaluating the performance of these models, I aim to determine which algorithm better identifies patterns related to purchasing behaviour. Additionally, feature importance analysis will be conducted to highlight key factors driving purchase decisions.

The dataset used, the "Online Shoppers Purchasing Intention Dataset" from the UCI Machine Learning Repository, was selected because of its rich combination of numerical and categorical features—such as page views, bounce rates, and visitor types—that are commonly observed in e-commerce contexts. This dataset also poses interesting challenges like class imbalance and feature selection, which are addressed through careful preprocessing.

Install Libraries

In this section, I ensure that all necessary libraries for data analysis, visualisation, and modelling are installed. These libraries include tools for handling data, building machine learning models, and evaluating model performance.

```
# Install necessary libraries if they are not installed
if (!require(httr)) install.packages("httr", dependencies = TRUE)
if (!require(readr)) install.packages("readr", dependencies = TRUE)
if (!require(ggplot2)) install.packages("ggplot2", dependencies = TRUE)
if (!require(reshape2)) install.packages("reshape2", dependencies = TRUE)
if (!require(psych)) install.packages("psych", dependencies = TRUE)
if (!require(caret)) install.packages("caret", dependencies = TRUE)
if (!require(caTools)) install.packages("caTools", dependencies = TRUE)
if (!require(scales)) install.packages("scales", dependencies = TRUE)
if (!require(nnet)) install.packages("nnet", dependencies = TRUE)
if (!require(randomForest)) install.packages("randomForest", dependencies = TRUE)
```

Load Libraries

In this section, I load the libraries that I will use throughout the analysis, enabling access to their functions for tasks like data downloading, visualisation, and modelling.

```
# Load necessary libraries
library(httr)       # For making HTTP requests to access online data
library(readr)      # For reading and writing CSV
library(ggplot2)    # For creating visualisations and data plots
library(reshape2)   # For creating heatmaps
library(psych)      # For calculating point-biserial correlations
library(caret)      # For one-hot encoding and model evaluation (confusion matrix)
library(caTools)    # For splitting data into training and testing sets
library(scales)     # For data scaling/normalisation
library(nnet)       # For creating neural network models (Multilayer Perceptron)
library(randomForest) # For building random forest models (ensemble learning)
```

Data Download and Preparation

This section loads the data from an online source. Then, I prepare data for further analysis into a data frame.

Note: that there's a small difference between the code in this section and the R file for the same section. The R file clears the environment and resets the console before loading data.

```
# Set the URL and file path
url <- "https://archive.ics.uci.edu/ml/machine-learning-databases/00468/online_shoppers_intention.csv"
file_path <- "online_shoppers_intention.csv"

# Download the dataset if it doesn't exist in the working directory
if(!file.exists(file_path)) {
  GET(url, write_disk(file_path, overwrite = TRUE))
  message("Dataset downloaded successfully.")
} else {
```

```
  message("Dataset already exists in the working directory.")
}
```

```
## Dataset already exists in the working directory.
```

```
# Load the dataset into R as a data frame
online_shoppers_purchasing_intention_dataset <- read_csv(file_path)
```

```
## Rows: 12330 Columns: 18
```

```
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (2): Month, VisitorType
## dbl (14): Administrative, Administrative_Duration, Informational, Informatio...
## lgl  (2): Weekend, Revenue
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

- The dataset is loaded from a CSV file that contains 12,330 rows and 18 columns.
- The dataset includes both numerical and categorical variables related to online shopper behaviours, such as Revenue, BounceRates, PageValues, and VisitorType.

Key Information:

The dataset comprises variables like:

- Revenue: The target variable indicating whether a purchase was made (TRUE/FALSE).
- BounceRates, ExitRates: Metrics on how users leave the site.
- ProductRelated and ProductRelated_Duration: Time and engagement on product-related pages.
- Month and VisitorType: Categorical data specifying the month of visit and type of visitor (returning, new).

Data Exploration

In this section, I inspect the structure of the dataset, check for missing values, and review basic statistics for the numerical features. This step is crucial for understanding the data's content and quality.

```
# View the first few rows of the dataset
head(online_shoppers_purchasing_intention_dataset)
```

```
## # A tibble: 6 x 18
##   Administrative Administrative_Duration Informational Informational_Duration
##            <dbl>                   <dbl>         <dbl>                  <dbl>
## 1              0                       0             0                      0
## 2              0                       0             0                      0
## 3              0                       0             0                      0
## 4              0                       0             0                      0
```

```
## 5              0                      0           0                      0
## 6              0                      0           0                      0
## # i 14 more variables: ProductRelated <dbl>, ProductRelated_Duration <dbl>,
## #   BounceRates <dbl>, ExitRates <dbl>, PageValues <dbl>, SpecialDay <dbl>,
## #   Month <chr>, OperatingSystems <dbl>, Browser <dbl>, Region <dbl>,
## #   TrafficType <dbl>, VisitorType <chr>, Weekend <lgl>, Revenue <lgl>
```

```
# Check for missing values in the dataset
missing_values <- colSums(is.na(online_shoppers_purchasing_intention_dataset))
print(missing_values)
```

```
##          Administrative Administrative_Duration          Informational
##                       0                       0                       0
##  Informational_Duration          ProductRelated ProductRelated_Duration
##                       0                       0                       0
##            BounceRates              ExitRates             PageValues
##                       0                       0                       0
##              SpecialDay                   Month        OperatingSystems
##                       0                       0                       0
##                 Browser                  Region             TrafficType
##                       0                       0                       0
##             VisitorType                 Weekend                 Revenue
##                       0                       0                       0
```

```
# Check the structure of the dataset
str(online_shoppers_purchasing_intention_dataset)
```

```
## spc_tbl_ [12,330 x 18] (S3: spec_tbl_df/tbl_df/tbl/data.frame)
##  $ Administrative         : num [1:12330] 0 0 0 0 0 0 0 1 0 0 ...
##  $ Administrative_Duration: num [1:12330] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Informational          : num [1:12330] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Informational_Duration : num [1:12330] 0 0 0 0 0 0 0 0 0 0 ...
##  $ ProductRelated         : num [1:12330] 1 2 1 2 10 19 1 0 2 3 ...
##  $ ProductRelated_Duration: num [1:12330] 0 64 0 2.67 627.5 ...
##  $ BounceRates            : num [1:12330] 0.2 0 0.2 0.05 0.02 ...
##  $ ExitRates              : num [1:12330] 0.2 0.1 0.2 0.14 0.05 ...
##  $ PageValues             : num [1:12330] 0 0 0 0 0 0 0 0 0 0 ...
##  $ SpecialDay             : num [1:12330] 0 0 0 0 0 0 0.4 0 0.8 0.4 ...
##  $ Month                  : chr [1:12330] "Feb" "Feb" "Feb" "Feb" ...
##  $ OperatingSystems       : num [1:12330] 1 2 4 3 3 2 2 1 2 2 ...
##  $ Browser                : num [1:12330] 1 2 1 2 3 2 4 2 2 4 ...
##  $ Region                 : num [1:12330] 1 1 9 2 1 1 3 1 2 1 ...
##  $ TrafficType            : num [1:12330] 1 2 3 4 4 3 3 5 3 2 ...
##  $ VisitorType            : chr [1:12330] "Returning_Visitor" "Returning_Visitor" "Returning_Visitor" "Returning_Visitor" ...
##  $ Weekend                : logi [1:12330] FALSE FALSE FALSE FALSE TRUE FALSE ...
##  $ Revenue                : logi [1:12330] FALSE FALSE FALSE FALSE FALSE FALSE ...
##  - attr(*, "spec")=
##   .. cols(
##   ..   Administrative = col_double(),
##   ..   Administrative_Duration = col_double(),
##   ..   Informational = col_double(),
##   ..   Informational_Duration = col_double(),
##   ..   ProductRelated = col_double(),
```

```
##   ..   ProductRelated_Duration = col_double(),
##   ..   BounceRates = col_double(),
##   ..   ExitRates = col_double(),
##   ..   PageValues = col_double(),
##   ..   SpecialDay = col_double(),
##   ..   Month = col_character(),
##   ..   OperatingSystems = col_double(),
##   ..   Browser = col_double(),
##   ..   Region = col_double(),
##   ..   TrafficType = col_double(),
##   ..   VisitorType = col_character(),
##   ..   Weekend = col_logical(),
##   ..   Revenue = col_logical()
##   .. )
##  - attr(*, "problems")=<externalptr>
```

```
# Summary statistics for numerical features
summary(online_shoppers_purchasing_intention_dataset)
```

```
##  Administrative    Administrative_Duration Informational
##  Min.   : 0.000    Min.   :   0.00         Min.   : 0.0000
##  1st Qu.: 0.000    1st Qu.:   0.00         1st Qu.: 0.0000
##  Median : 1.000    Median :   7.50         Median : 0.0000
##  Mean   : 2.315    Mean   :  80.82         Mean   : 0.5036
##  3rd Qu.: 4.000    3rd Qu.:  93.26         3rd Qu.: 0.0000
##  Max.   :27.000    Max.   :3398.75         Max.   :24.0000
##  Informational_Duration ProductRelated   ProductRelated_Duration
##  Min.   :   0.00        Min.   :  0.00   Min.   :    0.0
##  1st Qu.:   0.00        1st Qu.:  7.00   1st Qu.:  184.1
##  Median :   0.00        Median : 18.00   Median :  598.9
##  Mean   :  34.47        Mean   : 31.73   Mean   : 1194.8
##  3rd Qu.:   0.00        3rd Qu.: 38.00   3rd Qu.: 1464.2
##  Max.   :2549.38        Max.   :705.00   Max.   :63973.5
##   BounceRates          ExitRates         PageValues        SpecialDay
##  Min.   :0.000000   Min.   :0.00000   Min.   :  0.000   Min.   :0.00000
##  1st Qu.:0.000000   1st Qu.:0.01429   1st Qu.:  0.000   1st Qu.:0.00000
##  Median :0.003112   Median :0.02516   Median :  0.000   Median :0.00000
##  Mean   :0.022191   Mean   :0.04307   Mean   :  5.889   Mean   :0.06143
##  3rd Qu.:0.016813   3rd Qu.:0.05000   3rd Qu.:  0.000   3rd Qu.:0.00000
##  Max.   :0.200000   Max.   :0.20000   Max.   :361.764   Max.   :1.00000
##     Month           OperatingSystems     Browser           Region
##  Length:12330       Min.   :1.000     Min.   : 1.000   Min.   :1.000
##  Class :character   1st Qu.:2.000     1st Qu.: 2.000   1st Qu.:1.000
##  Mode  :character   Median :2.000     Median : 2.000   Median :3.000
##                     Mean   :2.124     Mean   : 2.357   Mean   :3.147
##                     3rd Qu.:3.000     3rd Qu.: 2.000   3rd Qu.:4.000
##                     Max.   :8.000     Max.   :13.000   Max.   :9.000
##   TrafficType     VisitorType          Weekend          Revenue
##  Min.   : 1.00   Length:12330       Mode :logical   Mode :logical
##  1st Qu.: 2.00   Class :character   FALSE:9462      FALSE:10422
##  Median : 2.00   Mode  :character   TRUE :2868      TRUE :1908
##  Mean   : 4.07
##  3rd Qu.: 4.00
##  Max.   :20.00
```

Data Overview: The dataset contains 12,330 rows and 18 columns. It includes numerical, categorical, and logical variables, covering various aspects of online shopper behaviour, such as time spent on different types of web pages, bounce rates, exit rates, and whether a purchase (Revenue) was made.

Missing Values: There are no missing values across the dataset, which indicates that the data is complete and ready for analysis without requiring imputation or further preprocessing to handle gaps.

Revenue Distribution: Out of 12,330 instances, only 1,908 result in purchases (Revenue = TRUE), suggesting an imbalanced dataset with relatively few positive purchase outcomes. This may require special attention in model development, such as balancing techniques or specific performance metrics for minority classes.

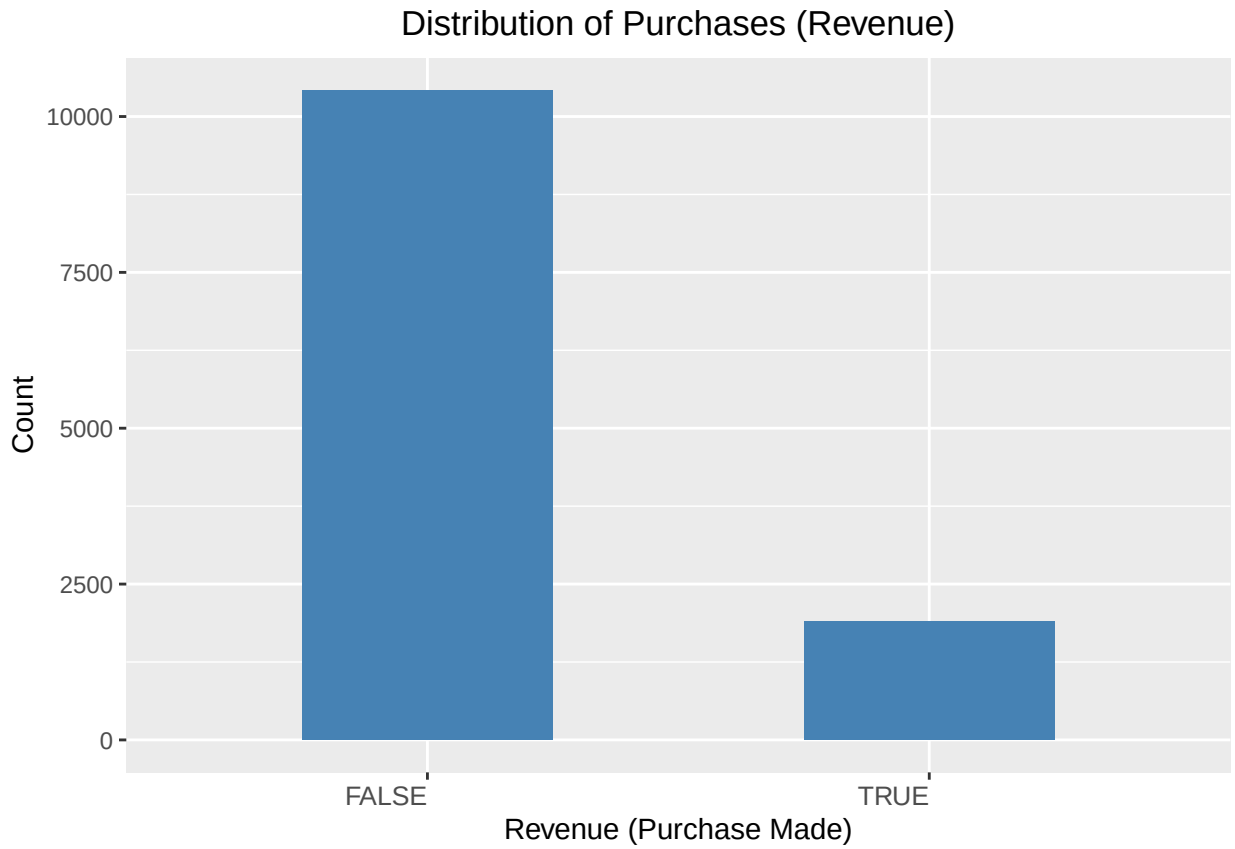
Time Spent on Pages: The mean values for the various time-based variables (e.g., ProductRelated_Duration, Administrative_Duration) show substantial variation. For example, users spend much more time on product-related pages (mean: 1194.8 seconds) compared to administrative or informational pages, highlighting that shoppers likely focus on product-related content during their sessions.

Bounce and Exit Rates: The average bounce and exit rates are relatively low (mean bounce rate: 0.022, exit rate: 0.043), indicating that most users explore multiple pages rather than leaving the site immediately after arriving.

Visualising the Distribution of the Target Variable

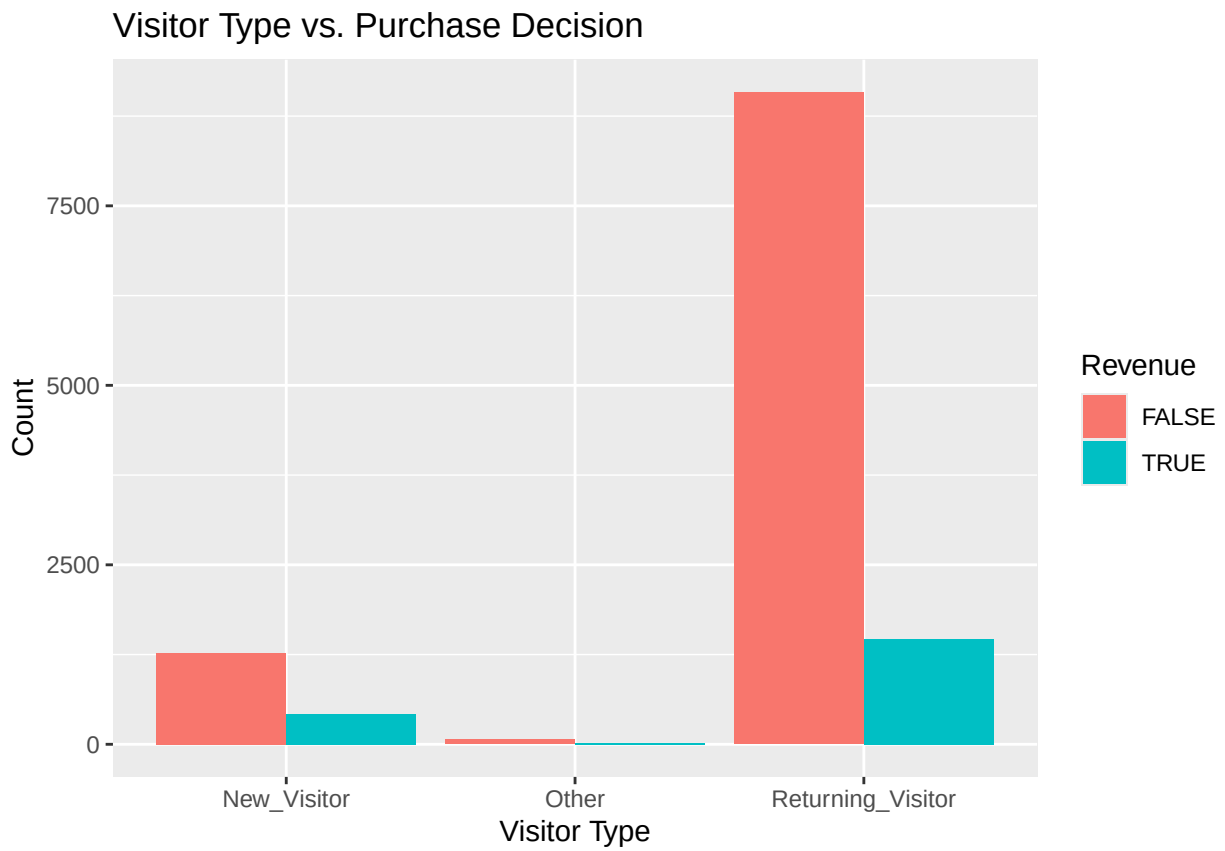
Here, I create visualisations to understand the distribution of the target variable (Revenue) and other important features. This helps us understand relationships in the dataset before modelling.

```
# Plot the distribution of Revenue
ggplot(online_shoppers_purchasing_intention_dataset, aes(x = Revenue)) +
  geom_bar(fill = 'steelblue', width = 0.5) +  # Adjust the width of the bars
  labs(title = "Distribution of Purchases (Revenue)",
       x = "Revenue (Purchase Made)", y = "Count") +
  theme(axis.text.x = element_text(size = 10, angle = 0, hjust = 1),  # Adjust x-axis text size
        plot.title = element_text(hjust = 0.5))  # Center the title
```



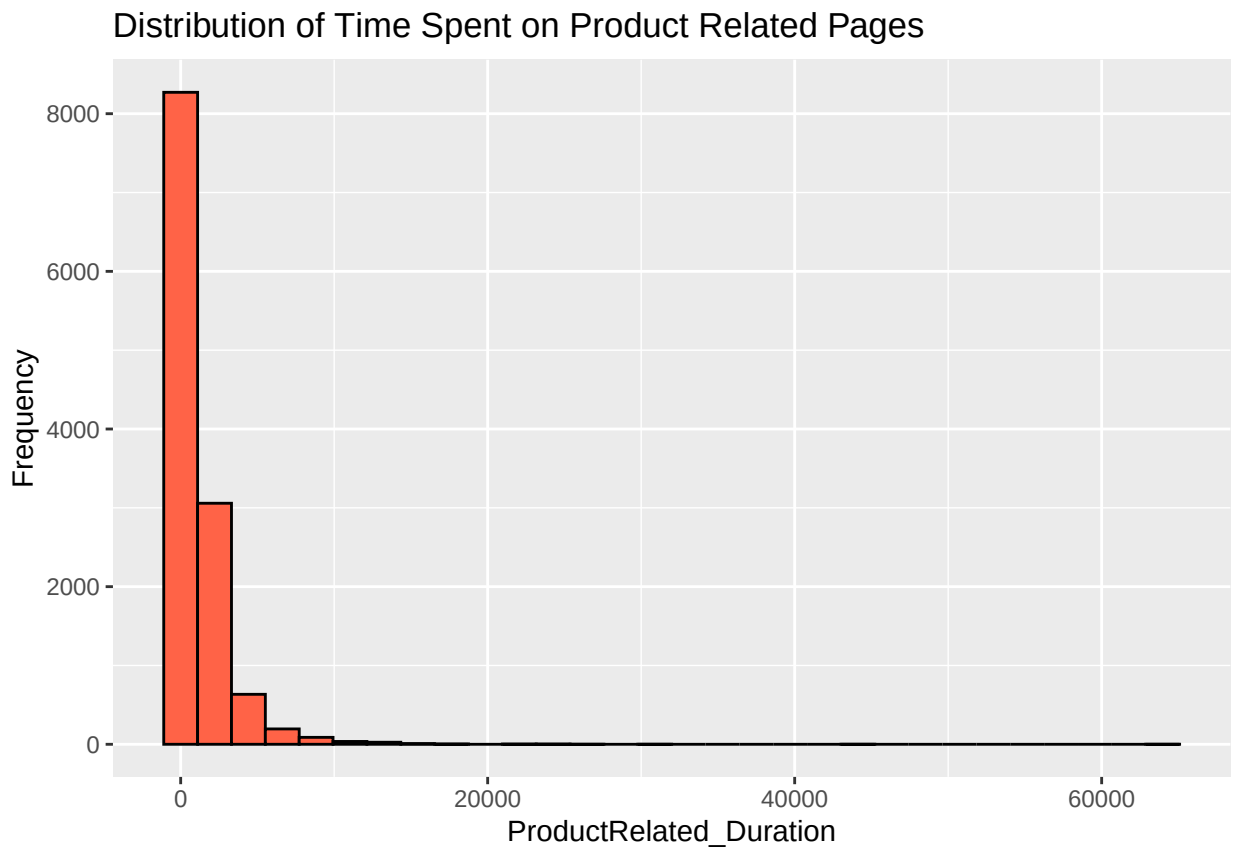
Distribution of Purchases (Revenue): - The dataset is highly imbalanced, with a much larger number of users who did not make a purchase (Revenue = FALSE) compared to those who did (Revenue = TRUE). This imbalance might influence model performance and may require handling techniques such as resampling.

```
# Bar plot for VisitorType and Revenue
ggplot(online_shoppers_purchasing_intention_dataset, aes(x = VisitorType, fill = Revenue)) +
  geom_bar(position = "dodge") +
  labs(title = "Visitor Type vs. Purchase Decision",
       x = "Visitor Type", y = "Count", fill = "Revenue")
```



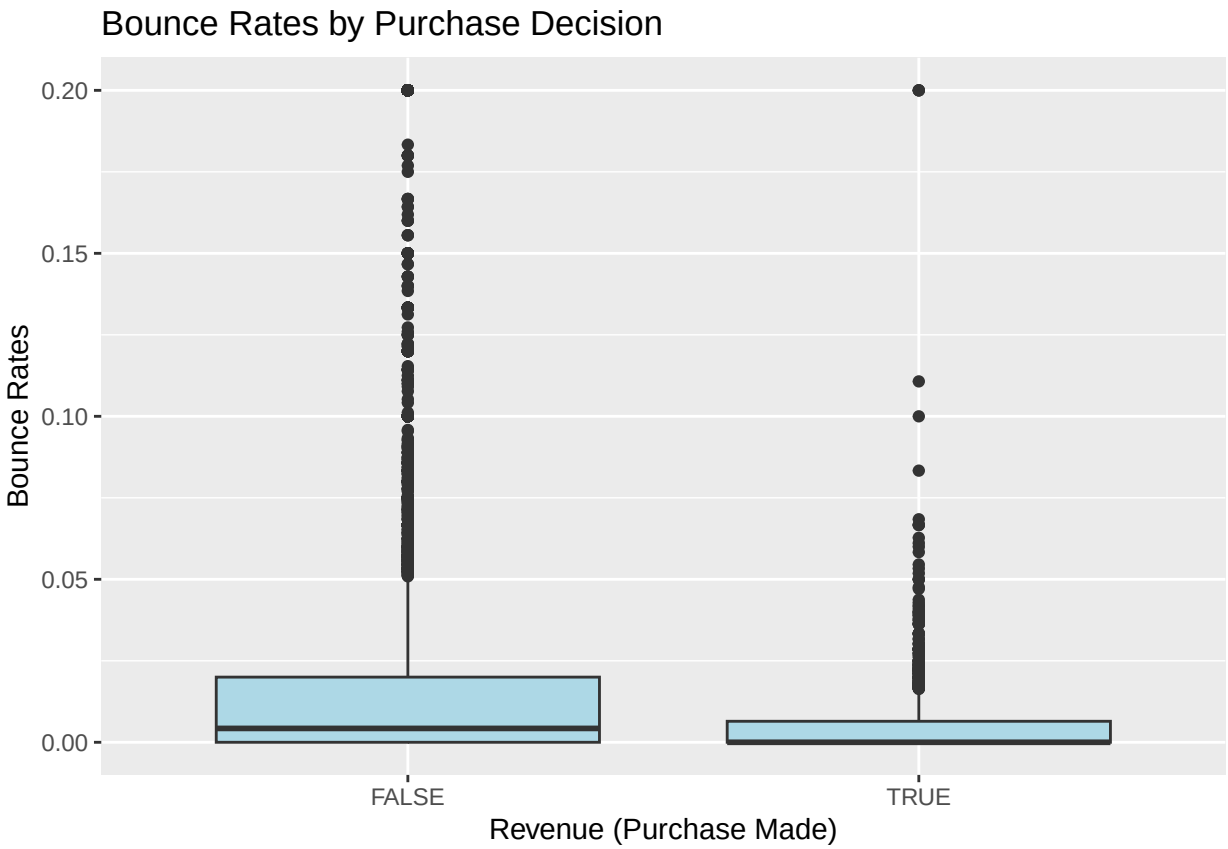
Visitor Type vs Purchase Decision: - Returning visitors are more likely to make a purchase compared to new or other visitor types. The majority of users are returning visitors who did not make a purchase, highlighting the importance of engagement strategies tailored to these users.

```
# Histogram for ProductRelated_Duration
ggplot(online_shoppers_purchasing_intention_dataset, aes(x = ProductRelated_Duration)) +
  geom_histogram(bins = 30, fill = 'tomato', color = 'black') +
  labs(title = "Distribution of Time Spent on Product Related Pages",
       x = "ProductRelated_Duration", y = "Frequency")
```



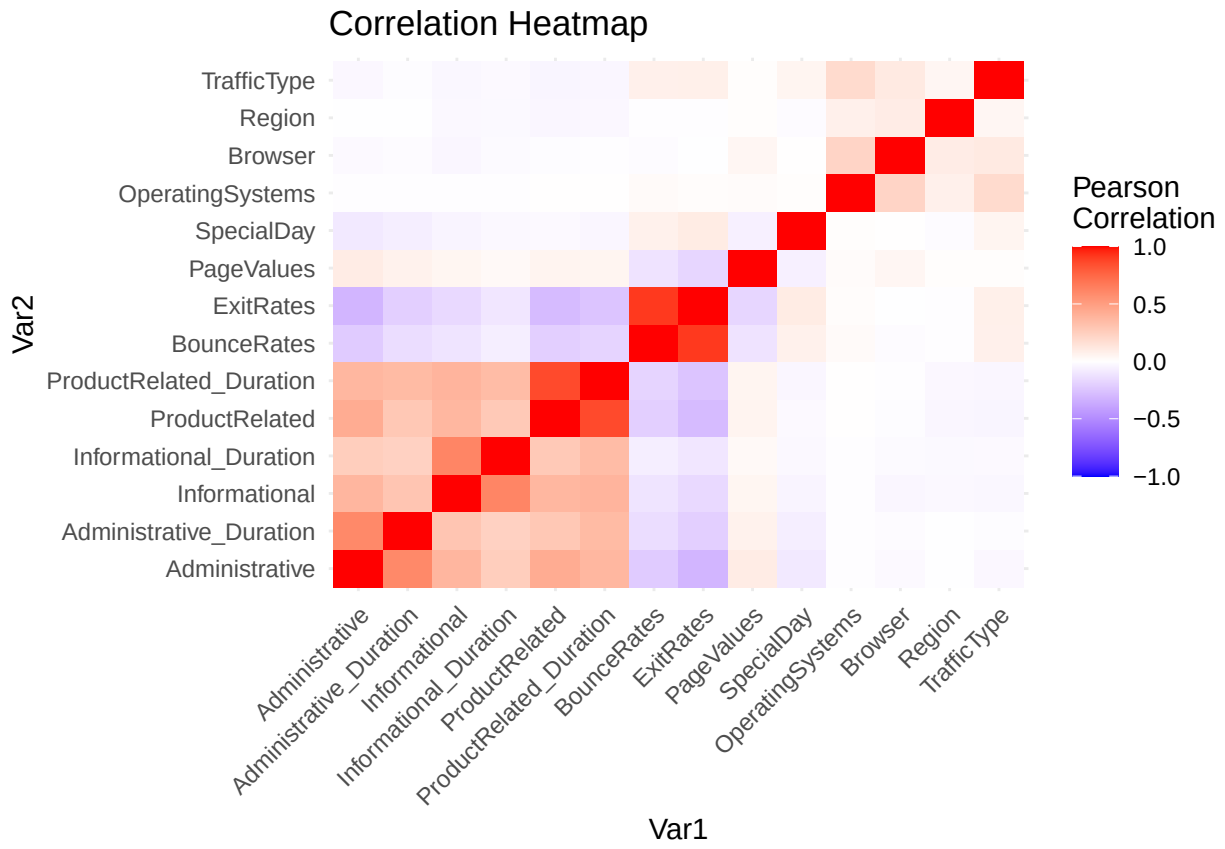
Distribution of Time Spent on Product-Related Pages: - Most users spend a relatively short amount of time on product-related pages, with the distribution being highly skewed. There are some outliers who spend significantly more time, but the bulk of users spend under 5,000 seconds.

```
# Boxplot for BounceRates grouped by Revenue
ggplot(online_shoppers_purchasing_intention_dataset, aes(x = Revenue, y = BounceRates)) +
  geom_boxplot(fill = 'lightblue') +
  labs(title = "Bounce Rates by Purchase Decision",
       x = "Revenue (Purchase Made)", y = "Bounce Rates")
```



Bounce Rates by Purchase Decision: - Users who did not make a purchase generally have higher bounce rates compared to those who did make a purchase. This suggests that a higher bounce rate is associated with a lower likelihood of conversion, and reducing bounce rates could lead to better sales outcomes.

```
# Calculate correlation matrix for numerical features
correlation_matrix <- cor(online_shoppers_purchasing_intention_dataset[, sapply(online_shoppers_purchasing_intention

# Melt the matrix for plotting
melted_corr <- melt(correlation_matrix)

# Plot the heatmap
ggplot(data = melted_corr, aes(x = Var1, y = Var2, fill = value)) +
  geom_tile() +
  scale_fill_gradient2(low = "blue", high = "red", mid = "white",
                       midpoint = 0, limit = c(-1, 1), space = "Lab",
                       name="Pearson\nCorrelation") +
  theme_minimal() +
  labs(title = "Correlation Heatmap") +
  theme(axis.text.x = element_text(angle = 45, vjust = 1, size = 9, hjust = 1))
```



Correlation Heatmap: - Strong positive correlations are observed between ProductRelated and ProductRelated_Duration, as expected, since these both measure aspects of product page engagement. There is also a notable correlation between BounceRates and ExitRates, indicating that these features may share similar information.

Calculating Correlation with Revenue

In this section, I calculate correlations between the numerical features and the target variable (Revenue). This allows me to explore relationships between these features and the likelihood of a purchase.

```
# Select numerical columns for correlation
numerical_columns <- online_shoppers_purchasing_intention_dataset[, sapply(online_shoppers_purchasing_intention_d

# Add Revenue as a binary numeric column (1 for TRUE, 0 for FALSE)
numerical_columns$Revenue <- as.numeric(online_shoppers_purchasing_intention_dataset$Revenue)

# Calculate point-biserial correlation
correlation_with_revenue <- corr.test(numerical_columns, use="pairwise", method="pearson")
print(correlation_with_revenue)
```

```
## Call:corr.test(x = numerical_columns, use = "pairwise", method = "pearson")
## Correlation matrix
##                          Administrative Administrative_Duration Informational
## Administrative                     1.00                   0.60          0.38
## Administrative_Duration            0.60                   1.00          0.30
```

```
## Informational                0.38                0.30          1.00
## Informational_Duration       0.26                0.24          0.62
## ProductRelated               0.43                0.29          0.37
## ProductRelated_Duration      0.37                0.36          0.39
## BounceRates                 -0.22               -0.14         -0.12
## ExitRates                   -0.32               -0.21         -0.16
## PageValues                   0.10                0.07          0.05
## SpecialDay                  -0.09               -0.07         -0.05
## OperatingSystems            -0.01               -0.01         -0.01
## Browser                     -0.03               -0.02         -0.04
## Region                      -0.01               -0.01         -0.03
## TrafficType                 -0.03               -0.01         -0.03
## Revenue                      0.14                0.09          0.10
##                      Informational_Duration ProductRelated
## Administrative                     0.26           0.43
## Administrative_Duration            0.24           0.29
## Informational                      0.62           0.37
## Informational_Duration             1.00           0.28
## ProductRelated                     0.28           1.00
## ProductRelated_Duration            0.35           0.86
## BounceRates                       -0.07          -0.20
## ExitRates                         -0.11          -0.29
## PageValues                         0.03           0.06
## SpecialDay                        -0.03          -0.02
## OperatingSystems                  -0.01           0.00
## Browser                           -0.02          -0.01
## Region                            -0.03          -0.04
## TrafficType                       -0.02          -0.04
## Revenue                            0.07           0.16
##                      ProductRelated_Duration BounceRates ExitRates
## Administrative                      0.37       -0.22      -0.32
## Administrative_Duration             0.36       -0.14      -0.21
## Informational                       0.39       -0.12      -0.16
## Informational_Duration              0.35       -0.07      -0.11
## ProductRelated                      0.86       -0.20      -0.29
## ProductRelated_Duration             1.00       -0.18      -0.25
## BounceRates                        -0.18        1.00       0.91
## ExitRates                          -0.25        0.91       1.00
## PageValues                          0.05       -0.12      -0.17
## SpecialDay                         -0.04        0.07       0.10
## OperatingSystems                    0.00        0.02       0.01
## Browser                            -0.01       -0.02       0.00
## Region                             -0.03       -0.01      -0.01
## TrafficType                        -0.04        0.08       0.08
## Revenue                             0.15       -0.15      -0.21
##                      PageValues SpecialDay OperatingSystems Browser Region
## Administrative             0.10      -0.09            -0.01   -0.03  -0.01
## Administrative_Duration    0.07      -0.07            -0.01   -0.02  -0.01
## Informational              0.05      -0.05            -0.01   -0.04  -0.03
## Informational_Duration     0.03      -0.03            -0.01   -0.02  -0.03
## ProductRelated             0.06      -0.02             0.00   -0.01  -0.04
## ProductRelated_Duration    0.05      -0.04             0.00   -0.01  -0.03
## BounceRates               -0.12       0.07             0.02   -0.02  -0.01
## ExitRates                 -0.17       0.10             0.01    0.00  -0.01
```

```
## PageValues                1.00       -0.06             0.02     0.05    0.01
## SpecialDay               -0.06        1.00             0.01     0.00   -0.02
## OperatingSystems          0.02        0.01             1.00     0.22    0.08
## Browser                   0.05        0.00             0.22     1.00    0.10
## Region                    0.01       -0.02             0.08     0.10    1.00
## TrafficType               0.01        0.05             0.19     0.11    0.05
## Revenue                   0.49       -0.08            -0.01     0.02   -0.01
##                      TrafficType Revenue
## Administrative             -0.03    0.14
## Administrative_Duration    -0.01    0.09
## Informational              -0.03    0.10
## Informational_Duration     -0.02    0.07
## ProductRelated             -0.04    0.16
## ProductRelated_Duration    -0.04    0.15
## BounceRates                 0.08   -0.15
## ExitRates                   0.08   -0.21
## PageValues                  0.01    0.49
## SpecialDay                  0.05   -0.08
## OperatingSystems            0.19   -0.01
## Browser                     0.11    0.02
## Region                      0.05   -0.01
## TrafficType                 1.00   -0.01
## Revenue                    -0.01    1.00
## Sample Size
## [1] 12330
## Probability values (Entries above the diagonal are adjusted for multiple tests.)
##                      Administrative Administrative_Duration Informational
## Administrative                 0.00                   0.00          0.00
## Administrative_Duration        0.00                   0.00          0.00
## Informational                  0.00                   0.00          0.00
## Informational_Duration         0.00                   0.00          0.00
## ProductRelated                 0.00                   0.00          0.00
## ProductRelated_Duration        0.00                   0.00          0.00
## BounceRates                    0.00                   0.00          0.00
## ExitRates                      0.00                   0.00          0.00
## PageValues                     0.00                   0.00          0.00
## SpecialDay                     0.00                   0.00          0.00
## OperatingSystems               0.48                   0.41          0.29
## Browser                        0.01                   0.09          0.00
## Region                         0.54                   0.54          0.00
## TrafficType                    0.00                   0.11          0.00
## Revenue                        0.00                   0.00          0.00
##                      Informational_Duration ProductRelated
## Administrative                         0.00           0.00
## Administrative_Duration                0.00           0.00
## Informational                          0.00           0.00
## Informational_Duration                 0.00           0.00
## ProductRelated                         0.00           0.00
## ProductRelated_Duration                0.00           0.00
## BounceRates                            0.00           0.00
## ExitRates                              0.00           0.00
## PageValues                             0.00           0.00
## SpecialDay                             0.00           0.01
## OperatingSystems                       0.29           0.63
```

```
## Browser                          0.03           0.14
## Region                           0.00           0.00
## TrafficType                      0.01           0.00
## Revenue                          0.00           0.00
##                    ProductRelated_Duration BounceRates ExitRates
## Administrative                        0.00       0.00      0.00
## Administrative_Duration               0.00       0.00      0.00
## Informational                         0.00       0.00      0.00
## Informational_Duration                0.00       0.00      0.00
## ProductRelated                        0.00       0.00      0.00
## ProductRelated_Duration               0.00       0.00      0.00
## BounceRates                           0.00       0.00      0.00
## ExitRates                             0.00       0.00      0.00
## PageValues                            0.00       0.00      0.00
## SpecialDay                            0.00       0.00      0.00
## OperatingSystems                      0.74       0.01      0.11
## Browser                               0.41       0.08      0.62
## Region                                0.00       0.47      0.32
## TrafficType                           0.00       0.00      0.00
## Revenue                               0.00       0.00      0.00
##                    PageValues SpecialDay OperatingSystems Browser Region
## Administrative           0.00       0.00             1.00    0.17   1.00
## Administrative_Duration  0.00       0.00             1.00    1.00   1.00
## Informational            0.00       0.00             1.00    0.00   0.04
## Informational_Duration   0.02       0.02             1.00    0.87   0.08
## ProductRelated           0.00       0.23             1.00    1.00   0.00
## ProductRelated_Duration  0.00       0.00             1.00    1.00   0.01
## BounceRates              0.00       0.00             0.23    1.00   1.00
## ExitRates                0.00       0.00             1.00    1.00   1.00
## PageValues               0.00       0.00             1.00    0.00   1.00
## SpecialDay               0.00       0.00             1.00    1.00   1.00
## OperatingSystems         0.04       0.16             0.00    0.00   0.00
## Browser                  0.00       0.70             0.00    0.00   0.00
## Region                   0.21       0.07             0.00    0.00   0.00
## TrafficType              0.16       0.00             0.00    0.00   0.00
## Revenue                  0.00       0.00             0.10    0.01   0.20
##                    TrafficType Revenue
## Administrative            0.01    0.00
## Administrative_Duration   1.00    0.00
## Informational             0.00    0.00
## Informational_Duration    0.19    0.00
## ProductRelated            0.00    0.00
## ProductRelated_Duration   0.00    0.00
## BounceRates               0.00    0.00
## ExitRates                 0.00    0.00
## PageValues                1.00    0.00
## SpecialDay                0.00    0.00
## OperatingSystems          0.00    1.00
## Browser                   0.00    0.23
## Region                    0.00    1.00
## TrafficType               0.00    1.00
## Revenue                   0.57    0.00
##
##  To see confidence intervals of the correlations, print with the short=FALSE option
```

Positive Correlation with Revenue: - Features such as ProductRelated, ProductRelated_Duration, and PageValues exhibit the strongest positive correlations with Revenue, with correlation values of 0.16, 0.15, and 0.49, respectively. This suggests that more interactions with product-related pages and higher page values increase the likelihood of a purchase.

Negative Correlation with Revenue: - BounceRates and ExitRates show moderate negative correlations with Revenue, at -0.15 and -0.21, respectively. This indicates that users who leave the website quickly (i.e., higher bounce or exit rates) are less likely to make a purchase.

Weak or No Correlation: - Other features such as OperatingSystems, Browser, Region, and TrafficType have very weak or near-zero correlations with Revenue, suggesting these variables have minimal impact on purchase decisions.

Feature Selection

I perform feature selection by focusing on key features that have moderate to strong correlations with Revenue. I also perform one-hot encoding to convert categorical variables into numerical ones for model building.

```
# Feature selection: focusing on key features with moderate to strong correlations with Revenue
# Convert VisitorType to dummy variables (one-hot encoding)
online_shoppers_purchasing_intention_dataset$VisitorType <- as.factor(online_shoppers_purchasing_intention_dataset$
dummies <- dummyVars(" ~ VisitorType + Month", data = online_shoppers_purchasing_intention_dataset)
online_shoppers_purchasing_intention_dataset_transformed <- data.frame(predict(dummies, newdata = online_shoppers

# Combine with the original dataset, keeping selected features
final_data <- cbind(online_shoppers_purchasing_intention_dataset[, c("PageValues", "ProductRelated", "ProductRelated
                    online_shoppers_purchasing_intention_dataset_transformed)
```

In the feature selection step, the following key features were selected:

1. **PageValues**
2. **ProductRelated**
3. **ProductRelated_Duration**
4. **BounceRates**
5. **ExitRates**
6. **Revenue** (target variable)

Additionally, categorical variables VisitorType and Month were transformed using one-hot encoding (dummy variables) and combined with the selected features to create the final dataset used for modelling.

Data Splitting and Scaling

In this section, I split the data into training and testing sets (80%-20%) and scale the numerical features to ensure that they are on the same scale before modelling.

```
# Set seed for reproducibility
set.seed(3350)

# Split the data
split <- sample.split(final_data$Revenue, SplitRatio = 0.8)
```

```r
train_data <- subset(final_data, split == TRUE)
test_data <- subset(final_data, split == FALSE)

# Standardise the numerical features in the training and testing data
train_data_scaled <- as.data.frame(lapply(train_data[, c("PageValues", "ProductRelated", "ProductRelated_Duration", "
test_data_scaled <- as.data.frame(lapply(test_data[, c("PageValues", "ProductRelated", "ProductRelated_Duration", "Bo

# Add the categorical variables and target (Revenue) back into the datasets
train_data <- cbind(train_data_scaled, train_data[, -c(1:5)])
test_data <- cbind(test_data_scaled, test_data[, -c(1:5)])
```

Model 1: MLP Model

I train a Multilayer Perceptron (MLP) model to classify whether or not a shopper will make a purchase. The model is trained on the training data and evaluated on the testing data.

```r
# Ensure that Revenue is a factor for classification
train_data$Revenue <- as.factor(train_data$Revenue)
test_data$Revenue <- as.factor(test_data$Revenue)

# Train the MLP model for classification
mlp_model <- nnet(Revenue ~ ., data = train_data, size = 10, maxit = 200)
```

```
## # weights:  201
## initial  value 7382.876325
## iter  10 value 2493.569422
## iter  20 value 2302.411503
## iter  30 value 2255.666463
## iter  40 value 2208.919051
## iter  50 value 2172.254904
## iter  60 value 2147.197916
## iter  70 value 2126.069882
## iter  80 value 2108.237129
## iter  90 value 2095.413158
## iter 100 value 2082.088866
## iter 110 value 2069.182940
## iter 120 value 2057.835869
## iter 130 value 2050.735758
## iter 140 value 2043.085164
## iter 150 value 2033.211766
## iter 160 value 2023.112572
## iter 170 value 2017.575283
## iter 180 value 2010.291158
## iter 190 value 1999.048367
## iter 200 value 1991.068552
## final  value 1991.068552
## stopped after 200 iterations
```

```r
# Summary of the model
summary(mlp_model)
```

```
## a 18-10-1 network with 201 weights
## options were - entropy fitting
##   b->h1  i1->h1  i2->h1  i3->h1  i4->h1  i5->h1  i6->h1  i7->h1  i8->h1  i9->h1
## -150.47   -5.67   94.06  -75.68   30.95 -332.52   20.47   -9.15 -160.67   -4.17
## i10->h1 i11->h1 i12->h1 i13->h1 i14->h1 i15->h1 i16->h1 i17->h1 i18->h1
## -116.07   28.70  -52.20   56.56  -30.02   19.54  -81.11   18.67    8.99
##   b->h2  i1->h2  i2->h2  i3->h2  i4->h2  i5->h2  i6->h2  i7->h2  i8->h2  i9->h2
##  -35.59   -4.84   -1.00    0.51    0.32   -2.32  -11.84  -11.44  -11.83   40.92
## i10->h2 i11->h2 i12->h2 i13->h2 i14->h2 i15->h2 i16->h2 i17->h2 i18->h2
##   41.06  -70.99 -171.04   41.31  -35.28   41.87   41.56   -7.21   41.88
##   b->h3  i1->h3  i2->h3  i3->h3  i4->h3  i5->h3  i6->h3  i7->h3  i8->h3  i9->h3
## -163.38  -19.91 -628.60  149.75 -153.19  -59.85  -24.70  116.74 -254.34  -59.48
## i10->h3 i11->h3 i12->h3 i13->h3 i14->h3 i15->h3 i16->h3 i17->h3 i18->h3
##  190.81    4.73 -238.46  143.75  165.11  124.21 -185.10  -97.27 -212.28
##   b->h4  i1->h4  i2->h4  i3->h4  i4->h4  i5->h4  i6->h4  i7->h4  i8->h4  i9->h4
##  -97.01   -0.58   -2.10   13.12  -59.79 -131.76  -52.94   -7.48  -36.57   40.04
## i10->h4 i11->h4 i12->h4 i13->h4 i14->h4 i15->h4 i16->h4 i17->h4 i18->h4
##   37.99   29.12  -21.22 -269.59   10.34    2.44   51.40   -9.95   33.28
##   b->h5  i1->h5  i2->h5  i3->h5  i4->h5  i5->h5  i6->h5  i7->h5  i8->h5  i9->h5
##  -91.16 -414.29    0.00   -0.09   -0.40    0.46  -30.45  -30.56  -30.90  -13.51
## i10->h5 i11->h5 i12->h5 i13->h5 i14->h5 i15->h5 i16->h5 i17->h5 i18->h5
##  -13.59   44.04  -13.48  -13.18  -11.80   31.85  -14.10  -13.46  -75.14
##   b->h6  i1->h6  i2->h6  i3->h6  i4->h6  i5->h6  i6->h6  i7->h6  i8->h6  i9->h6
##  152.56 -197.28  -49.32  -79.35  156.35 -116.36  162.86  116.92 -128.04  184.60
## i10->h6 i11->h6 i12->h6 i13->h6 i14->h6 i15->h6 i16->h6 i17->h6 i18->h6
##  -47.11 -106.82  107.80   58.81 -118.80   66.09  -69.87  -61.87  138.52
##   b->h7  i1->h7  i2->h7  i3->h7  i4->h7  i5->h7  i6->h7  i7->h7  i8->h7  i9->h7
##  -62.67   -5.48    4.78   -5.94  -15.20   20.03  -21.36  -15.09  -27.33    9.16
## i10->h7 i11->h7 i12->h7 i13->h7 i14->h7 i15->h7 i16->h7 i17->h7 i18->h7
## -190.12   33.45 -147.81   82.58  100.96   87.29   82.13 -192.38   72.17
##   b->h8  i1->h8  i2->h8  i3->h8  i4->h8  i5->h8  i6->h8  i7->h8  i8->h8  i9->h8
##  -46.77  -86.71   -5.03   28.78   -6.48   87.19   79.19 -109.40  -18.09  -13.32
## i10->h8 i11->h8 i12->h8 i13->h8 i14->h8 i15->h8 i16->h8 i17->h8 i18->h8
##  166.18  -10.51  -10.28 -148.24   45.67   83.26 -247.51   33.77   51.45
##   b->h9  i1->h9  i2->h9  i3->h9  i4->h9  i5->h9  i6->h9  i7->h9  i8->h9  i9->h9
##  -16.12    0.38   -1.18   -3.08   -1.14   -0.02  118.40 -145.54   12.50    1.51
## i10->h9 i11->h9 i12->h9 i13->h9 i14->h9 i15->h9 i16->h9 i17->h9 i18->h9
##   -5.67  -21.98    0.66    3.38    2.78    1.73    0.52    1.43    0.49
##   b->h10  i1->h10  i2->h10  i3->h10  i4->h10  i5->h10  i6->h10  i7->h10
##   159.28  -313.03  -100.70    27.33    17.66    45.56   147.70   141.74
##  i8->h10  i9->h10 i10->h10 i11->h10 i12->h10 i13->h10 i14->h10 i15->h10
##  -130.38    44.42    34.67   136.94   -59.18  -149.22   194.03   115.01
## i16->h10 i17->h10 i18->h10
##   -31.73  -148.67    26.57
##    b->o   h1->o   h2->o   h3->o   h4->o   h5->o   h6->o   h7->o   h8->o   h9->o
##   -0.52   -1.35  -61.28    2.21    1.44 -142.21   -1.45   -1.96   -2.00    2.96
##  h10->o
##    1.63
```

```r
# Generate predictions using the MLP model
predictions <- predict(mlp_model, test_data, type = "class")

# Convert predictions and test_data$Revenue to factors with the same levels
predictions <- factor(predictions, levels = levels(test_data$Revenue))
```

```r
test_data$Revenue <- factor(test_data$Revenue)

# Confusion matrix to evaluate the model
confusionMatrix(predictions, test_data$Revenue)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
##      FALSE  1861  143
##      TRUE    223  239
##
##                Accuracy : 0.8516
##                  95% CI : (0.8369, 0.8654)
##     No Information Rate : 0.8451
##     P-Value [Acc > NIR] : 0.1946
##
##                   Kappa : 0.4778
##
##  Mcnemar's Test P-Value : 3.637e-05
##
##             Sensitivity : 0.8930
##             Specificity : 0.6257
##          Pos Pred Value : 0.9286
##          Neg Pred Value : 0.5173
##              Prevalence : 0.8451
##          Detection Rate : 0.7547
##    Detection Prevalence : 0.8127
##       Balanced Accuracy : 0.7593
##
##        'Positive' Class : FALSE
##
```

The results for the MLP (Multilayer Perceptron) model in this section show:

1. **Training Process**:
   - The MLP model, configured with 10 neurons in the hidden layer and a maximum of 200 iterations, successfully trained, reaching a final cost function value of **1991.07**. This suggests the model converged after 200 iterations, improving its ability to classify the target variable (Revenue).
2. **Model Structure**:
   - The summary reveals that the model has **18 input features**, **10 hidden neurons**, and **1 output neuron**, using a total of **201 weights**. These weights represent the connections between neurons and are essential for determining the strength and direction of the relationships between inputs and outputs.
3. **Confusion Matrix**:
   - The confusion matrix shows the performance of the MLP model on the test data:
     - **True negatives (correctly classified as FALSE)**: 1861
     - **False negatives (incorrectly classified as FALSE)**: 143
     - **True positives (correctly classified as TRUE)**: 239
     - **False positives (incorrectly classified as TRUE)**: 223

4. **Key Metrics**:
    - **Accuracy**: The model has an accuracy of **0.8516**, meaning it correctly classifies about 85% of instances. The 95% confidence interval is (0.8369, 0.8654), indicating a reliable classification performance.
    - **Sensitivity** (True Negative Rate): **0.8930**, indicating the model is highly effective at correctly identifying users who did not make a purchase.
    - **Specificity** (True Positive Rate): **0.6257**, meaning the model has moderate success in identifying users who did make a purchase.
    - **Kappa**: **0.4778**, which measures agreement between predicted and actual classifications, suggests moderate classification performance.
    - **Balanced Accuracy**: **0.7593**, indicating that the model performs reasonably well across both classes (purchases and no purchases).

Insights:

- The model performs well, with high sensitivity (good at predicting users who don't purchase). However, its specificity is lower, indicating that the model struggles somewhat to accurately predict users who make a purchase.
- The imbalance in the dataset (many more FALSE than TRUE values for Revenue) is reflected in the confusion matrix and accuracy metrics, as the model does well in predicting the majority class (FALSE) but is less effective with the minority class (TRUE).
- The **McNemar's Test P-Value** being quite low (3.637e-05) suggests that there is a significant difference between the model's predictions of TRUE and FALSE, reinforcing that the model may need further optimisation to handle the imbalanced data.

Hyperparameter Tuning for MLP Model

In this section, I will use grid search and cross-validation to fine-tune the number of neurons and regularisation because the initial MLP model showed promising accuracy but could benefit from better specificity in predicting purchases.

```
# Define the grid of hyperparameters (size and decay only)
tune_grid <- expand.grid(size = c(5, 10, 15),        # Number of neurons in the hidden layer
                         decay = c(0.01, 0.1, 0.5))  # Regularisation parameter to prevent overfitting

# Set up cross-validation control
train_control <- trainControl(method = "cv", number = 5)  # 5-fold cross-validation

# Train the MLP model with grid search
set.seed(3350)
mlp_tuned <- train(Revenue ~ .,
                   data = train_data,
                   method = "nnet",
                   tuneGrid = tune_grid,
                   trControl = train_control,
                   trace = FALSE,  # Suppress training output
                   maxit = 200)    # Set maxit outside of tuneGrid

# Print the best model found by grid search
print(mlp_tuned$bestTune)
```

```
##   size decay
## 1    5  0.01
```

```r
# Predict on the test data using the best tuned model
best_predictions <- predict(mlp_tuned, newdata = test_data)

# Confusion matrix to evaluate the tuned model
confusionMatrix(best_predictions, test_data$Revenue)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
##      FALSE  1987  136
##      TRUE     97  246
##
##                Accuracy : 0.9055
##                  95% CI : (0.8933, 0.9168)
##     No Information Rate : 0.8451
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.6234
##
##  Mcnemar's Test P-Value : 0.01279
##
##             Sensitivity : 0.9535
##             Specificity : 0.6440
##          Pos Pred Value : 0.9359
##          Neg Pred Value : 0.7172
##              Prevalence : 0.8451
##          Detection Rate : 0.8058
##    Detection Prevalence : 0.8609
##       Balanced Accuracy : 0.7987
##
##        'Positive' Class : FALSE
##
```

1. **Optimal Hyperparameters**:
    - The grid search identified that the optimal configuration is **5 neurons** in the hidden layer with a **decay (regularisation) of 0.01**. This relatively small network with light regularisation suggests that the model can generalise well without overfitting.
2. **Improved Model Performance**:
    - The tuned MLP model shows an accuracy of **90.55%**, a notable improvement from the initial model's accuracy of 85.16%. The 95% confidence interval (0.8933, 0.9168) suggests a reliable performance.
3. **Sensitivity and Specificity**:
    - **Sensitivity (True Negative Rate)** improved to **95.35%**, meaning the model is very good at identifying non-purchasers.
    - **Specificity (True Positive Rate)** slightly improved to **64.40%**, indicating better but still moderate performance in identifying purchasers compared to the initial model.

4. **Kappa Score**:
    - The **Kappa score of 0.6234** shows a significant improvement in agreement between predicted and actual outcomes, demonstrating a more balanced model that better handles the class imbalance.
5. **Balanced Accuracy**:
    - **Balanced Accuracy** increased to **79.87%**, indicating that the model's ability to classify both classes (purchasers and non-purchasers) has improved, with a better balance between sensitivity and specificity.

Model 2: Random Forest Model

In this section, I train a Random Forest model to classify purchases, using an ensemble approach to improve prediction performance.

```
# Train the Random Forest model
set.seed(3350)
rf_model <- randomForest(Revenue ~ ., data = train_data, ntree = 100, mtry = 3, importance = TRUE)

# Print the summary of the model
print(rf_model)
```

```
##
## Call:
##  randomForest(formula = Revenue ~ ., data = train_data, ntree = 100,      mtry = 3, importance = TRUE)
##                Type of random forest: classification
##                      Number of trees: 100
## No. of variables tried at each split: 3
##
##         OOB estimate of  error rate: 10.03%
## Confusion matrix:
##       FALSE TRUE class.error
## FALSE  8097  241  0.02890381
## TRUE    748  778  0.49017038
```

1. **Model Configuration**:
    - The model was trained using **100 trees** with **3 variables** randomly selected at each split. Random Forest typically benefits from bagging, which leads to reduced variance and improved accuracy compared to single models.
2. **Out-of-Bag (OOB) Error**:
    - The OOB error estimate is **10.03%**, indicating that the Random Forest model performs well, with an error rate of only 10% on the out-of-bag samples (data not used in the training process). This low error rate shows that the model generalises well to unseen data.
3. **Confusion Matrix**:
    - For **non-purchasers (FALSE)**:
        - The model correctly classified **8097** instances and misclassified **241** instances, resulting in a class error rate of **2.89%**.
    - For **purchasers (TRUE)**:

  - The model correctly classified **778** instances but misclassified **748** instances, leading to a much higher class error rate of **49.02%**.

4. **Imbalance in Class Error**:
   - The model performs exceptionally well for the majority class (non-purchasers) with a very low error rate. However, it struggles with the minority class (purchasers), with nearly half of the true purchasers misclassified. This suggests the model is biased towards the majority class, which is common in imbalanced datasets.

Model Comparison and Feature Importance

Finally, I compare the MLP and Random Forest models using various performance metrics and visualise the importance of features in each model.

```
# Evaluate the Random Forest Model
rf_predictions <- predict(rf_model, newdata = test_data)
confusionMatrix(rf_predictions, test_data$Revenue)
```

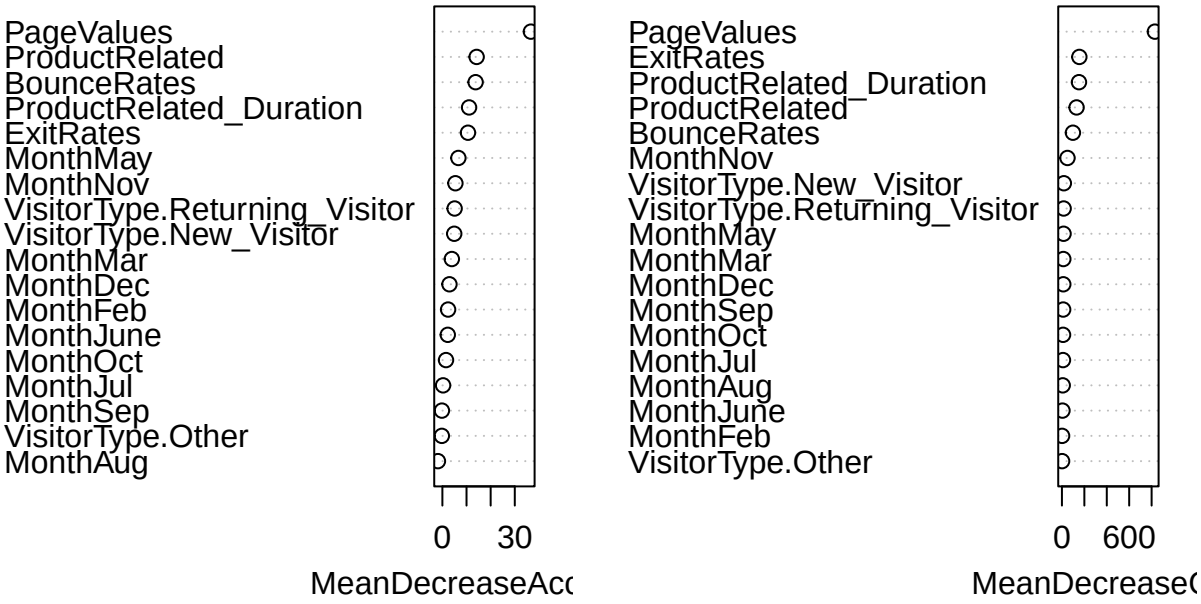
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
##      FALSE  2027  167
##      TRUE     57  215
##
##                Accuracy : 0.9092
##                  95% CI : (0.8971, 0.9202)
##     No Information Rate : 0.8451
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6068
##
##  Mcnemar's Test P-Value : 3.268e-13
##
##             Sensitivity : 0.9726
##             Specificity : 0.5628
##          Pos Pred Value : 0.9239
##          Neg Pred Value : 0.7904
##              Prevalence : 0.8451
##          Detection Rate : 0.8220
##    Detection Prevalence : 0.8897
##       Balanced Accuracy : 0.7677
##
##        'Positive' Class : FALSE
##
```

The Random Forest model achieved 90.92% accuracy with a high sensitivity of 97.26%. However, the model struggles with specificity at 56.28%, meaning it correctly identifies non-purchasers but has difficulty distinguishing purchasers from non-purchasers.

```r
# View feature importance from Random Forest
importance(rf_model)
```

```
##                                 FALSE       TRUE MeanDecreaseAccuracy
## PageValues                  28.2889364 40.9109415           36.6668470
## ProductRelated              12.8362494  0.1438200           14.1837668
## ProductRelated_Duration      9.3831958 -0.1661320           11.0615487
## BounceRates                  8.9590156  6.9303263           13.7621411
## ExitRates                    5.5401370 11.0354249           10.6067611
## VisitorType.New_Visitor      4.7753549  0.7500962            4.9055130
## VisitorType.Other            0.6827938 -1.7508098           -0.2324661
## VisitorType.Returning_Visitor 5.0799519 -0.5961472           5.0645502
## MonthAug                    -1.5797013 -0.7940597           -1.8504409
## MonthDec                     3.2655021  0.5801575            2.9224963
## MonthFeb                     0.3062228  2.9529815            2.3566405
## MonthJul                     0.5609782 -0.3186923            0.2841445
## MonthJune                    1.3113044  1.9661948            2.2217883
## MonthMar                     3.9212098  1.1191906            3.9116843
## MonthMay                     3.5223260  4.1750291            6.6225076
## MonthNov                    -2.5655395  7.6606705            5.3855561
## MonthOct                     2.2249266 -0.3422983            1.5011966
## MonthSep                     0.5032577 -1.0266789           -0.2010504
##                             MeanDecreaseGini
## PageValues                        829.126343
## ProductRelated                    130.526187
## ProductRelated_Duration           152.013071
## BounceRates                        97.918714
## ExitRates                         155.360904
## VisitorType.New_Visitor            17.148976
## VisitorType.Other                   1.397439
## VisitorType.Returning_Visitor      15.224444
## MonthAug                            7.033604
## MonthDec                           12.204020
## MonthFeb                            1.467940
## MonthJul                            9.169708
## MonthJune                           4.697517
## MonthMar                           12.445064
## MonthMay                           14.593375
## MonthNov                           49.431984
## MonthOct                            9.934678
## MonthSep                           10.952675
```

```r
varImpPlot(rf_model)
```



Feature importance from the Random Forest model highlights **PageValues**, **ExitRates**, and **ProductRelated_Duration** as the most important predictors. These features have the largest impact on model performance, indicating that user interaction with product pages and navigation behaviour (exits and bounces) are key factors in determining purchase behaviour.

```
# Calculate performance metrics for the MLP model
mlp_conf_matrix <- confusionMatrix(predictions, test_data$Revenue)
mlp_accuracy <- mlp_conf_matrix$overall['Accuracy']
mlp_precision <- mlp_conf_matrix$byClass['Pos Pred Value']
mlp_recall <- mlp_conf_matrix$byClass['Sensitivity']
mlp_f1 <- 2 * (mlp_precision * mlp_recall) / (mlp_precision + mlp_recall)
```

The MLP model has slightly lower accuracy (85.16%) compared to Random Forest but shows better specificity (62.57%) in predicting purchasers, indicating a stronger balance between sensitivity and specificity for detecting the minority class.

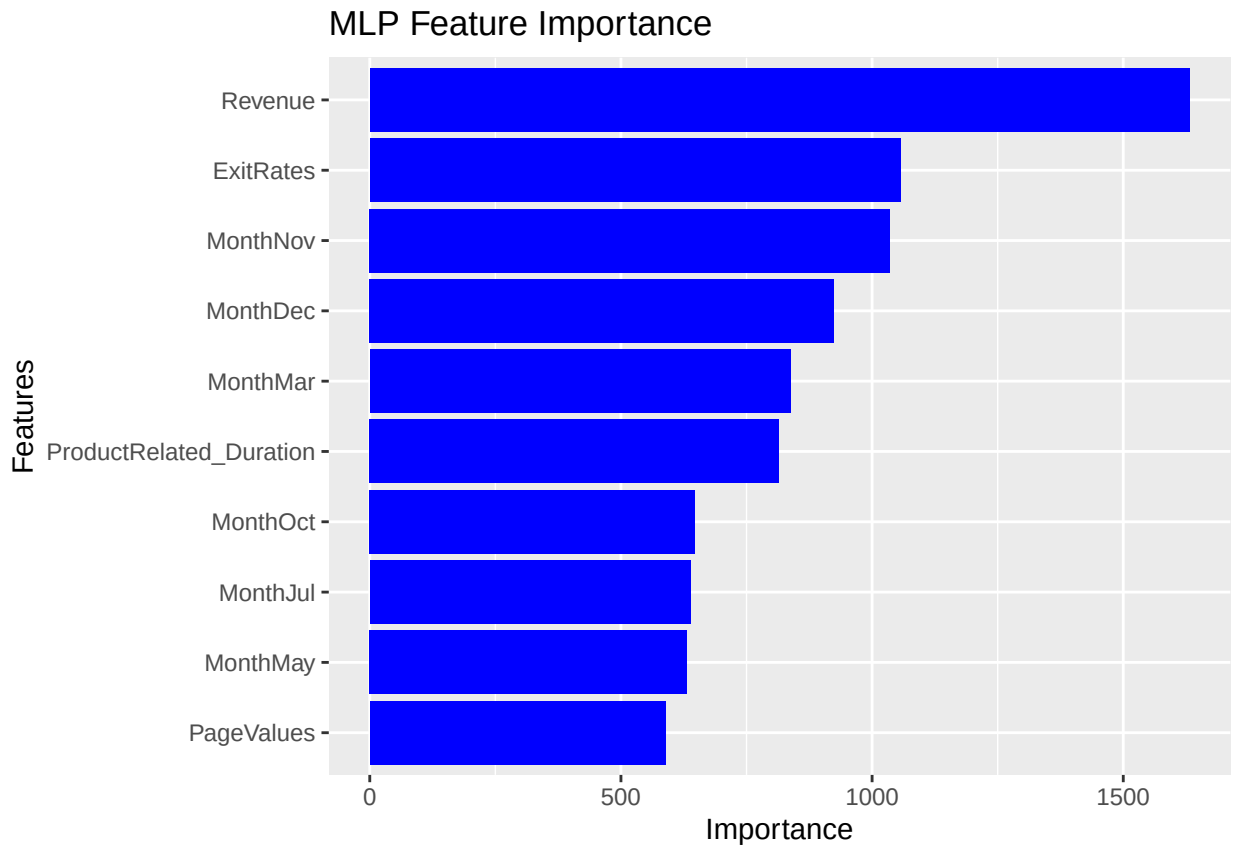
```
# Calculate performance metrics for the Random Forest model
rf_conf_matrix <- confusionMatrix(rf_predictions, test_data$Revenue)
rf_accuracy <- rf_conf_matrix$overall['Accuracy']
rf_precision <- rf_conf_matrix$byClass['Pos Pred Value']
rf_recall <- rf_conf_matrix$byClass['Sensitivity']
rf_f1 <- 2 * (rf_precision * rf_recall) / (rf_precision + rf_recall)

# Print model comparison results
model_comparison <- data.frame(
  Model = c("MLP", "Random Forest"),
```

```
  Accuracy = c(mlp_accuracy, rf_accuracy),
  Precision = c(mlp_precision, rf_precision),
  Recall = c(mlp_recall, rf_recall),
  F1_Score = c(mlp_f1, rf_f1)
)

print(model_comparison)
```

```
##           Model  Accuracy Precision    Recall  F1_Score
## 1           MLP 0.8515815 0.9286427 0.8929942 0.9104697
## 2 Random Forest 0.9091646 0.9238833 0.9726488 0.9476391
```

Comparing the two models, the Random Forest outperforms the MLP model in terms of overall accuracy and sensitivity. However, the MLP model provides a slightly better balance for detecting the minority class (purchasers). The Random Forest's F1 score (94.76%) is slightly higher than that of the MLP model (91.05%).

```
# Now, extract and visualise the importance of features based on the MLP model's weights
weights <- mlp_model$wts

# Find the number of input features (including bias)
n_input <- ncol(train_data) - 1  # Subtract 1 for the target variable

# Extract the weights connecting the input layer to the hidden layer (skip the output layer for now)
input_weights <- weights[1:(n_input * 10)]  # 10 is the number of hidden neurons used

# Calculate the importance of each input feature as the sum of absolute values of their weights
importance_scores <- apply(matrix(input_weights, nrow = n_input), 1, function(x) sum(abs(x)))

# Create a data frame for feature importance
mlp_importance <- data.frame(Feature = colnames(train_data)[1:n_input], Importance = importance_scores)

# Sort by importance and visualise
mlp_importance_sorted <- mlp_importance[order(-mlp_importance$Importance), ]

# Visualise the importance of the top 10 features
top_features <- head(mlp_importance_sorted, 10)
ggplot(top_features, aes(x = reorder(Feature, Importance), y = Importance)) +
  geom_bar(stat = "identity", fill = 'blue') +
  labs(title = "MLP Feature Importance", x = "Features", y = "Importance") +
  coord_flip()
```



The MLP model identifies **Revenue**, **ExitRates**, and several months (e.g., **MonthNov**, **MonthDec**) as the most significant features. These are consistent with the top features from the Random Forest model, further indicating the importance of user engagement with product pages and seasonal trends in predicting purchase behaviour.

Conclusion

In this project, I set out to predict whether an online shopping session would culminate in a purchase by developing and comparing two machine learning models: a Multilayer Perceptron (MLP) and a Random Forest classifier. Through comprehensive data exploration and preprocessing, I prepared the dataset to address challenges such as class imbalance and the presence of both numerical and categorical variables.

My analysis demonstrated that both models achieved high accuracy in predicting purchase intentions. The Random Forest model slightly outperformed the tuned MLP model, achieving an accuracy of **90.92%** compared to **90.55%** for the MLP. The Random Forest's superior performance is attributed to its ensemble nature, which enhances predictive accuracy by aggregating the results of multiple decision trees.

However, the MLP model exhibited better balance in detecting the minority class (purchasers). Specifically, the MLP achieved a higher specificity, indicating it was more effective in correctly identifying actual purchasers despite the dataset's imbalance. This suggests that while Random Forests may offer higher overall accuracy, **MLPs might be more suitable when the correct classification of the minority class is a priority.**

Feature importance analysis revealed that variables such as **PageValues**, **ExitRates**, and **ProductRelated_Duration** are significant predictors of purchasing **behaviour**. These features indicate that users who spend more time on product-related pages, view pages with higher value, and have lower exit rates are more likely to make a purchase. Additionally, categorical variables like **Month** and **VisitorType** also played

a crucial role, highlighting the impact of seasonal trends and user engagement levels on purchasing decisions.

Despite the strong performance of both models, the project faced limitations due to the imbalanced nature of the dataset, with a significantly higher number of non-purchase instances. Future work could involve implementing techniques like Synthetic Minority Over-sampling Technique (SMOTE) or cost-sensitive learning to further improve the models' ability to detect purchasers. Additionally, exploring other algorithms such as Gradient Boosting Machines or Deep Learning models might yield even better predictive performance.

Overall, this project underscores the potential of machine learning models in predicting online shoppers' purchasing intentions. The insights gained from feature importance analysis can inform e-commerce businesses in refining their marketing strategies, **personalising** user experiences, and ultimately increasing conversion rates. By focusing on the key factors that influence purchasing **behaviour**, businesses can make data-driven decisions to enhance customer satisfaction and drive growth.

Sources:

1. **Irizarry, R. (2022)** *PH125.8x: Data Science: Machine Learning*. HarvardX, edX. Available at: https://www.edx.org/course/harvardx-ph125-8x-data-science-machine-learning (Accessed: 29 September 2024).
2. **Irizarry, R. A. (2022)** *Introducción a la Ciencia de Datos: Análisis de datos y algoritmos de predicción con R*. HarvardX. Available at: https://leanpub.com/datasciencebook (Accessed: 29 September 2024).
3. **Nelson, H. (2023)** *Essential Math for AI: Next-Level Mathematics for Efficient and Successful AI Systems*. 1st edn. Sebastopol: O'Reilly Media.
4. **Sakar, C. and Kastro, Y. (2018)** *Online Shoppers Purchasing Intention Dataset*. UCI Machine Learning Repository. Available at: https://doi.org/10.24432/C5F88Q.