Analysis of Online Shoppers' Purchase Intention

Kaarthik Sundaramoorthy, Sahil Shah and Vidhi Shah

6/23/2020

Milestone 1: Project Proposal

Problem Definition

In the era of online shopping, known as e-shopping, people use online transactions to buy the items they need while exploring it online. This helps the buyers as well as sellers to understand the patterns, intentions, and behavior of various online customers. Thereby, helping businesses improve their revenue by focusing on customer experiences and marketing. Hence, the analysis of online shoppers' purchase intention has become an emerging field in data mining. Click-stream analysis refers to the online shoppers' behavior analysis as they invoke a sequence of web pages in a particular session. Therefore, analyzing this data is a primary goal for successful online businesses as they extract the clicks and behavior through web page requests. Our proposed solution is to provide a decisive and feasible recommendation algorithm that will allow us to predict the behavior of the shoppers'.

System Funtionality

Firstly, we will build the model and analyze the performance of various classification algorithms such as Decision Tree, Random Forest, Support Vector Machine (SVM), and Naive Bayes. Then, the values of different evaluation metrics like Accuracy, Precision, Recall, F-score will be calculated to compare the performance of each of the algorithms. Lastly, we also plan on using these classification models to predict the shopper's intentions.

Requirements and Benefits

In today's economy e-commerce is becoming more extensive and businesses within this sector need to understand, the factors which come into play when a shopper ventures into a website to make a purchase. The benefit this holds is that it will enable the websites to better target ads or other factors which may lead to an increase in sales. These findings support the feasibility of accurate and scalable purchasing intention prediction for virtual shopping environments. This also helps in knowing the market capabilities of the brand when released in the new market while finding out the problems in the existing market and helps in relevant marketing strategies that can help in conquering the market.

Dataset Details and Core Algorithm

The dataset which is used is based on the "Online Shoppers Purchasing Intention" UCI dataset. It consists of numerical as well as categorical data. There are a total of 12,330 records where each row corresponds to the session data of the particular user. The total no. of records for which the session ended without any purchase is 10,442 which contributes to 84.5%. The core algorithm which we will be implementing is a

Decision Tree. Being a classification algorithm, it creates a tree-like structure by creating rules for breaking the dataset into small subsets in each step. We create a training model that is used to predict the class of the variable by simply learning decision rules deduced from the training set. At each step, a decision is taken to classify the data in the beneath classes. The leaf node holds the final results.

Milestone 2: Data Summary/visualization

The dataset used in the project is based on *"online shoppers purchasing intention"* available on UCI Machine Learning dataset.

Importing Libraries

This are the important libraries that are to be installed for the execution of the file.

```
library(ggplot2)
library(tidyverse)
library(gmodels)
library(dplyr)
library(ggmosaic)
library(corrplot)
library(caret)
library(rpart)
library(rpart.plot)
library(cluster)
library(fpc)
library(data.table)
library(knitr)
library(kableExtra)
library(plyr)
library(caTools)
```

Importing the Dataset

The `read.csv()` command is used to import the dataset.

```
dataset <- read.csv("online_shoppers_intention.csv", header = TRUE)
attach(dataset)
```

Checking the number of columns and rows of the dataset.

```
ncol(dataset)
```

```
## [1] 18
```

```
nrow(dataset)
```

```
## [1] 12330
```

Looking at the dataset data structure.

```r
str(dataset)
```

```
## 'data.frame':    12330 obs. of  18 variables:
##  $ Administrative         : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ Administrative_Duration: num  0 0 0 0 0 0 0 0 0 0 ...
##  $ Informational          : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ Informational_Duration : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ ProductRelated         : int  1 2 1 2 10 19 1 0 2 3 ...
##  $ ProductRelated_Duration: num  0 64 0 2.67 627.5 ...
##  $ BounceRates            : num  0.2 0 0.2 0.05 0.02 ...
##  $ ExitRates              : num  0.2 0.1 0.2 0.14 0.05 ...
##  $ PageValues             : num  0 0 0 0 0 0 0 0 0 0 ...
##  $ SpecialDay             : num  0 0 0 0 0 0 0.4 0 0.8 0.4 ...
##  $ Month                  : Factor w/ 10 levels "Aug","Dec","Feb",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ OperatingSystems       : int  1 2 4 3 3 2 2 1 2 2 ...
##  $ Browser                : int  1 2 1 2 3 2 4 2 2 4 ...
##  $ Region                 : int  1 1 9 2 1 1 3 1 2 1 ...
##  $ TrafficType            : int  1 2 3 4 4 3 3 5 3 2 ...
##  $ VisitorType            : Factor w/ 3 levels "New_Visitor",..: 3 3 3 3 3 3 3 3 3 3 ...
##  $ Weekend                : logi  FALSE FALSE FALSE FALSE TRUE FALSE ...
##  $ Revenue                : logi  FALSE FALSE FALSE FALSE FALSE FALSE ...
```

```r
summary(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
##  May    :3364   Min.   :1.000    Min.   : 1.000   Min.   :1.000
##  Nov    :2998   1st Qu.:2.000    1st Qu.: 2.000   1st Qu.:1.000
##  Mar    :1907   Median :2.000    Median : 2.000   Median :3.000
##  Dec    :1727   Mean   :2.124    Mean   : 2.357   Mean   :3.147
```

```
##  Oct     : 549   3rd Qu.:3.000    3rd Qu.: 2.000   3rd Qu.:4.000
##  Sep     : 448   Max.   :8.000    Max.   :13.000   Max.   :9.000
##  (Other):1337
##   TrafficType               VisitorType     Weekend         Revenue
##  Min.   : 1.00   New_Visitor      : 1694   Mode :logical   Mode :logical
##  1st Qu.: 2.00   Other            :   85   FALSE:9462      FALSE:10422
##  Median : 2.00   Returning_Visitor:10551   TRUE :2868      TRUE :1908
##  Mean   : 4.07
##  3rd Qu.: 4.00
##  Max.   :20.00
##
```

The purchasing intention model is designed as a classification problem which measures the purchasers' commitment to finalize purchase intent. Hence we have the session data of the users which has two categories : users who purchased the item and who didn't. The dataset consists of both numerical data and categorical data, and thus the target value is categorical. Table 1 refers to the numerical features and Table 2 refers to the categorical features used in the prediction model respectively. There are a total of 12,330 rows where each row represents session data of one particular user.

```r
tab1 <- read.csv("table1.csv", header = TRUE)
kable(tab1) %>%
  kable_styling(full_width = T)
```

ï..Feature.Name	Description	Min..value	Max..value	SD
Administrative	Number of pages visited by the visitor about account management	0	27.0	3.322e+00
Administrative duration	Total amount of time (in seconds) spent by the visitor on account management related pages	0	3399.0	1.768e+02
Informational	Number of pages visited by the visitor about Web site, communication and address information of the shopping site	0	24.0	1.270e+00
Informational duration	Total amount of time (in seconds) spent by the visitor on informational pages	0	2549.4	1.407e+02
Product related	Number of pages visited by visitor about product related pages	0	705.0	4.448e+01
Product related duration	Total amount of time (in seconds) spent by the visitor on product related pages	0	63974.0	1.914e+03
Bounce rates	Average bounce rate value of the pages visited by the visitor	0	0.2	4.849e-02
Exit rate	Average exit rate value of the pages visited by the visitor	0	0.2	4.860e-02
Page value	Average page value of the pages visited by the visitor	0	361.8	1.857e+01
Special day	Closeness of the site visiting time to a special day	0	1.0	1.989e-01

```r
tab2 <- read.csv("table2.csv", header = TRUE)
kable(tab2) %>%
  kable_styling(full_width = T)
```

ï..Name	Description	Values
OperatingSystems	Operating system of the visitor	8
Browser	Browser of the visitor	13
Region	Geographic region from which the session has been started by the visitor	9
TrafficType	Traffic source by which the visitor has arrived at the Web site (e.g., banner, SMS, direct)	20
VisitorType	Visitor type as New Visitor, Returning Visitor, and Other	3
Weekend	Boolean value indicating whether the date of the visit is weekend	2
Month	Month value of the visit date	10
Revenue	Class label indicating whether the visit has been finalized with a transaction	2

Taking the look at the **REVENUE** column which is the target column. The datatype of the REVENUE column is Logical which holds the value **TRUE** and **FALSE**.

```
library(gmodels)
summary(dataset$Revenue)
```

```
##    Mode   FALSE    TRUE
## logical   10422    1908
```

```
CrossTable(dataset$Revenue)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  12330
##
##
##           |     FALSE |      TRUE |
##           |-----------|-----------|
##           |     10422 |      1908 |
##           |     0.845 |     0.155 |
##           |-----------|-----------|
##
##
##
##
```

Adding the new *Revenue_binary* column by using Logical Data of Shopper's Revenue into binary dependent variable that will helpful for potential regression models. The data will be converted with values 0 and 1, i.e. If it is false the value is 0 and if true it will be 1.

```r
dataset <- dataset %>%
  mutate(Revenue_binary = ifelse(dataset$Revenue == "TRUE", 1, 0))
```

Checking the dataset if it has any missing values.

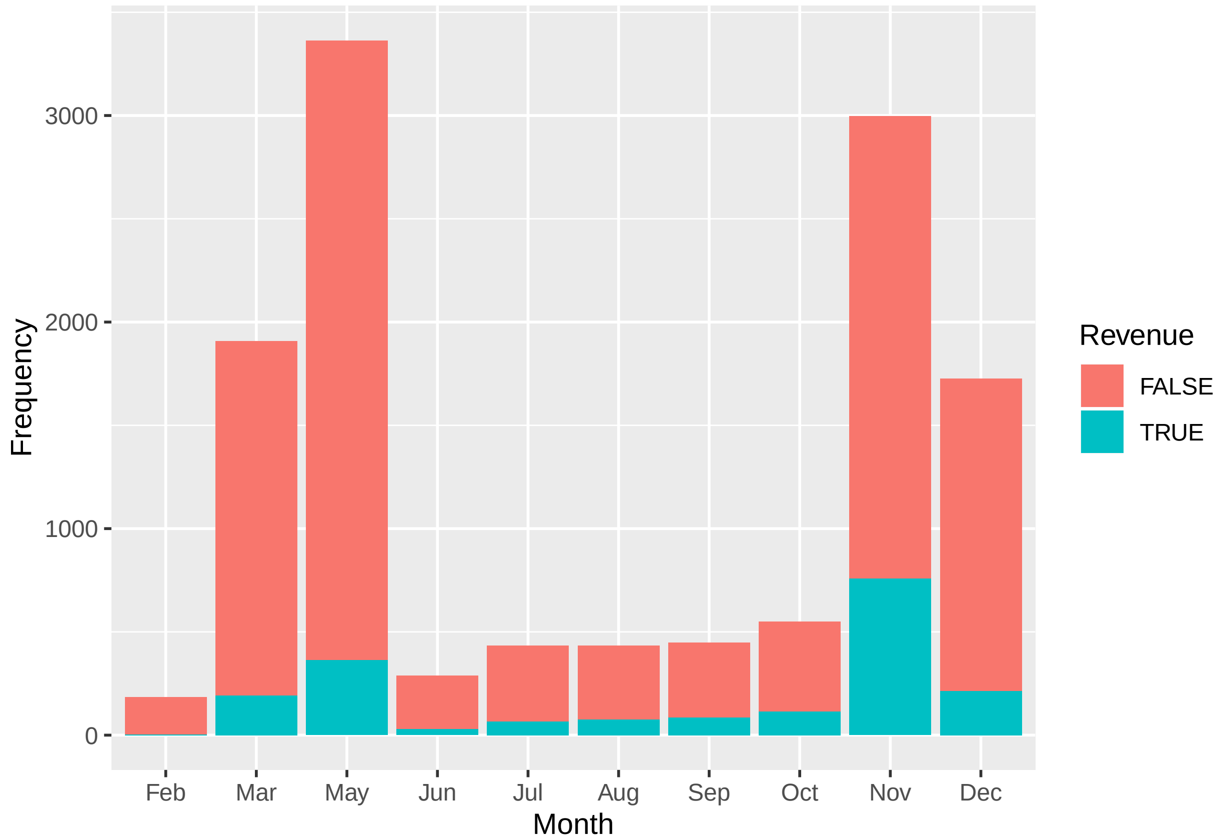
```r
colSums(is.na(dataset))
```

```
##          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
##          Revenue_binary
##                       0
```

Visualizations

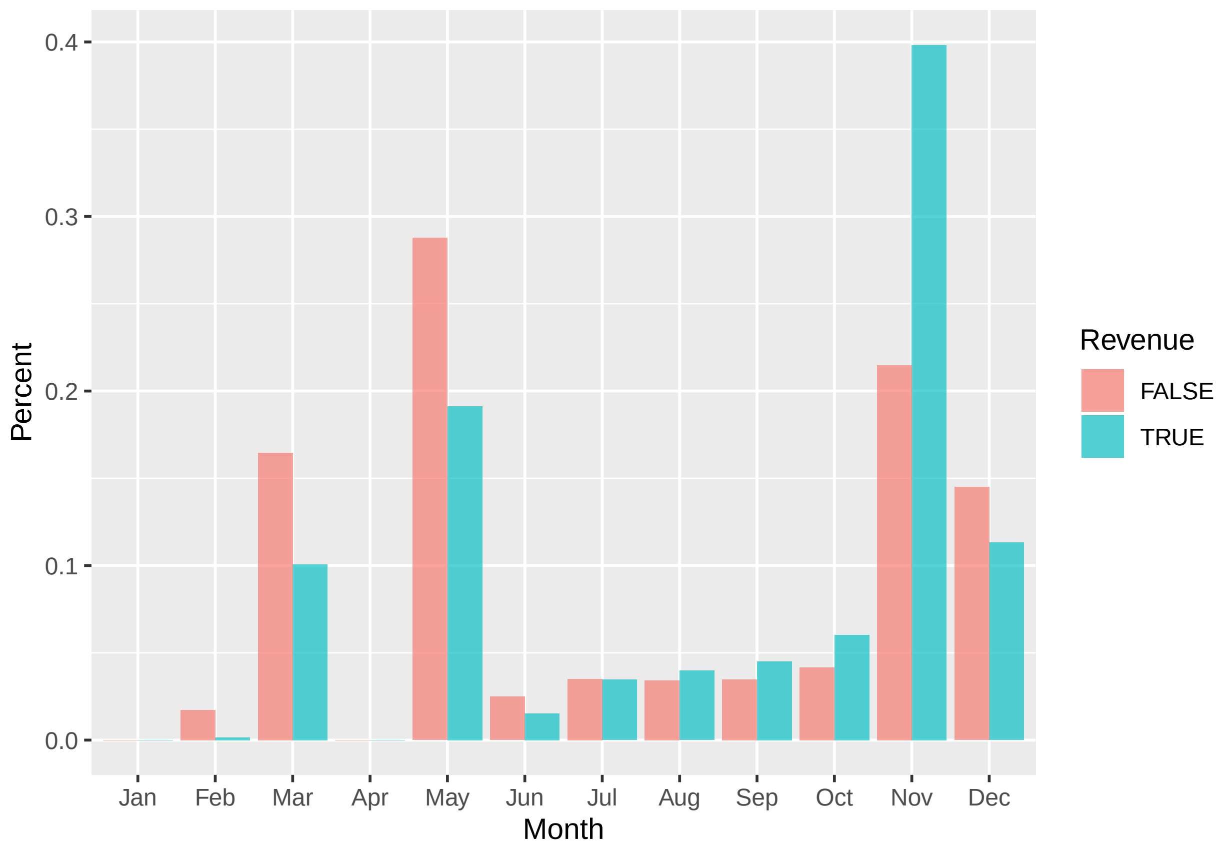
Month

```r
dataset$Month = factor(dataset$Month, levels = month.abb)
dataset %>%
  ggplot() +
  aes(x = Month, Revenue = ..count../nrow(dataset), fill = Revenue) +
  geom_bar() +
  ylab("Frequency")
```



The plot describes the frequency of the revenue generated over the months.

```
table_month = table(dataset$Month, dataset$Revenue)
tab_mon =  as.data.frame(prop.table(table_month,2))
colnames(tab_mon) = c("Month", "Revenue", "perc")
ggplot(data = tab_mon, aes(x = Month, y = perc, fill = Revenue)) +
  geom_bar(stat = 'identity', position = 'dodge', alpha = 2/3) +
  xlab("Month")+
  ylab("Percent")
```



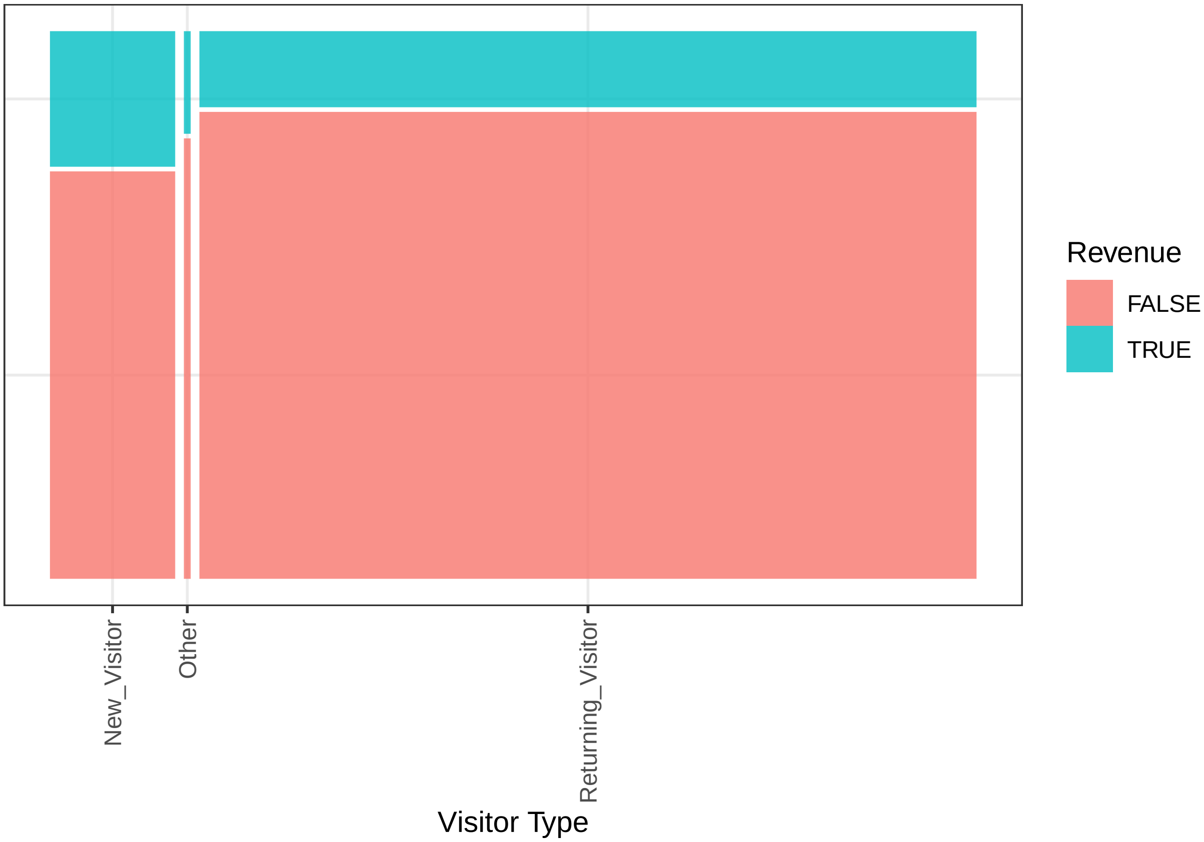
The plot portrays the high shopping rates in the months September, October and November with respect to the customers not buying the products. These months are comparatively considered as the *Holiday Season Months*. Also, there is high hits on the website with positive revenue in the month of may.

Visitor

```
theme_set(theme_bw())

## setting default parameters for mosaic plots
mosaic_theme = theme(axis.text.x = element_text(angle = 90,
                                                hjust = 1,
                                                vjust = 0.5),
                     axis.text.y = element_blank(),
                     axis.ticks.y = element_blank())
```

```
dataset %>%
  ggplot() +
  geom_mosaic(aes(x = product(Revenue, VisitorType), fill = Revenue)) +
  mosaic_theme +
  xlab("Visitor Type") +
  ylab(NULL)
```



The comparison of the VisitorType which are New_Visitors, Returning_Visitor and Others with Revenue generated. There are many returning visitors in the contrast to less new visitors. Although, the new visitors have high probablity of purchasing the product and help the revenue than the returning visitors.

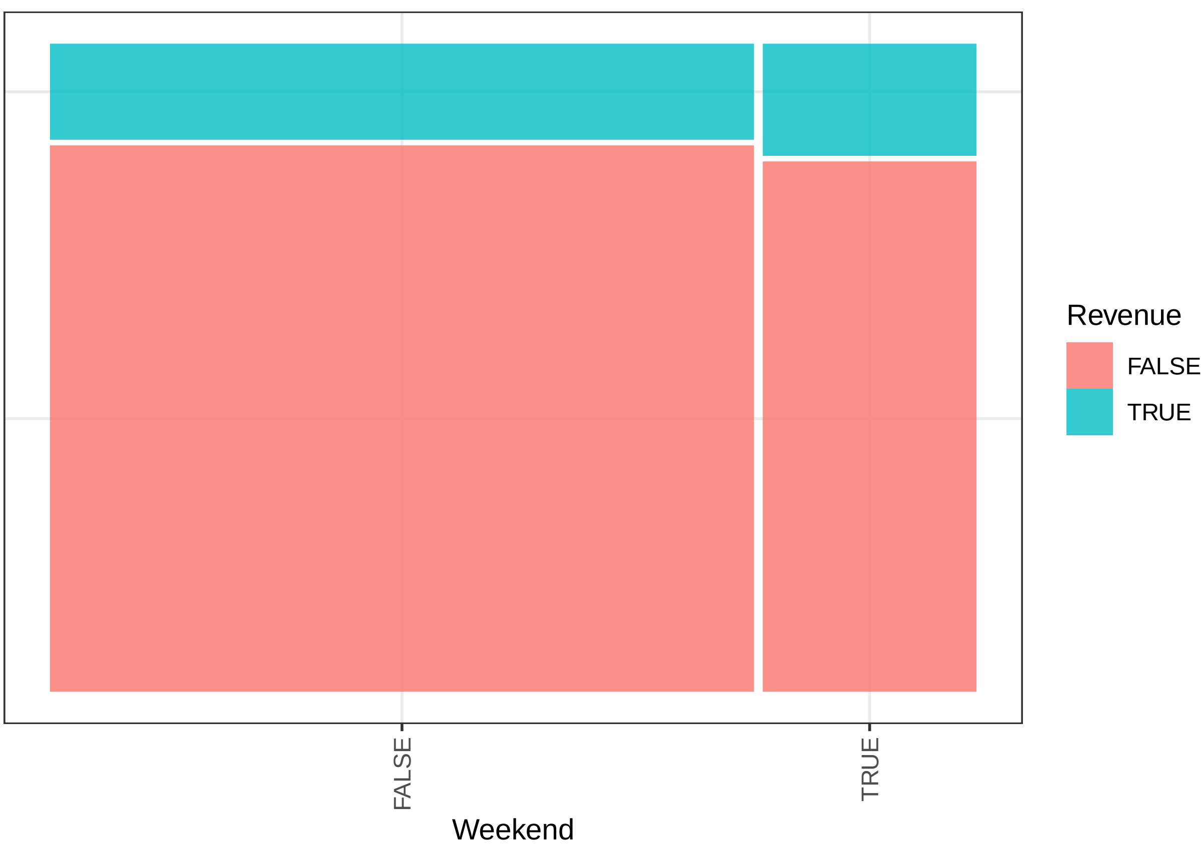
Weekend

```
CrossTable(dataset$Weekend, dataset$Revenue)
```

```
##
##
##    Cell Contents
## |-------------------------|
## |                       N |
## | Chi-square contribution |
## |           N / Row Total |
## |           N / Col Total |
## |         N / Table Total |
## |-------------------------|
##
##
## Total Observations in Table:  12330
##
##
##                 | dataset$Revenue
```

```
## dataset$Weekend |      FALSE |       TRUE | Row Total |
## -----------------|-----------|-----------|-----------|
##            FALSE |       8053 |       1409 |       9462 |
##                  |      0.381 |      2.080 |            |
##                  |      0.851 |      0.149 |      0.767 |
##                  |      0.773 |      0.738 |            |
##                  |      0.653 |      0.114 |            |
## -----------------|-----------|-----------|-----------|
##             TRUE |       2369 |        499 |       2868 |
##                  |      1.257 |      6.864 |            |
##                  |      0.826 |      0.174 |      0.233 |
##                  |      0.227 |      0.262 |            |
##                  |      0.192 |      0.040 |            |
## -----------------|-----------|-----------|-----------|
##     Column Total |      10422 |       1908 |      12330 |
##                  |      0.845 |      0.155 |            |
## -----------------|-----------|-----------|-----------|
##
##
```

```r
dataset %>%
  ggplot() +
  mosaic_theme +
  geom_mosaic(aes(x = product(Revenue,Weekend), fill = Revenue)) +
  xlab("Weekend") +
  ylab(NULL)
```



The **Weekend** analysis shows that more than 70% of visitors are visiting the site on weekdays, with 15% chance of actually buying the products. The rest 30% visit on the weekend and there is 17% speculation of buying.

Milestone 3: Algorithm Testing

Currently we are implemented two classification algorithms Decision Tree and K-Nearest Neighbors and achieved really good accuracies in both of them. In the next milestone we are planning to implement naive bayes and SVM.

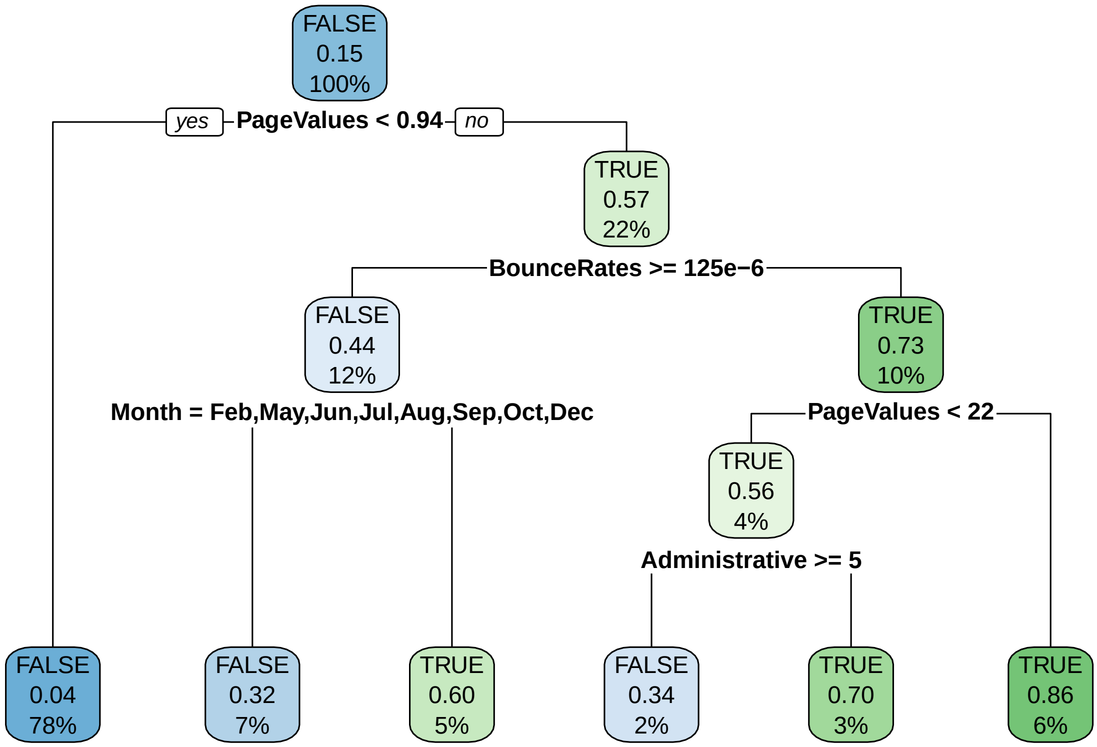
Decision Tree

Now preparing the dataset for the classification algorithms. Splitting the dataset into testing and training data for training the model. We will split the data in ratio 80:20.

```
dataset_classify = dataset[-c(19:22)]
set.seed(123)
split = sample.split(dataset_classify$Revenue, SplitRatio = 0.8)
training_data = subset(dataset_classify, split == TRUE)
test_data = subset(dataset_classify, split == FALSE)
```

Running the decision tree from the “rpart” library:

```
dt_model<- rpart(Revenue ~ . , data = training_data, method="class")
rpart.plot(dt_model)
```



The predictive model suggests that Page Values greater than 0.94 lead to a TRUE 57% of the time. On top of this, an effective Bounce Rate above 0 improves our TRUE to 73% and Administrative type '5' or below result in a TRUE 83% of the time. Also, we see that October and November are good months for shoppers' conversions.

```
dt.pred <- predict(dt_model,test_data,type = "class")
mean(dt.pred==test_data$Revenue)
```

```
## [1] 0.8965937
```

Confusion Matrix for Decision Tree

```
cfdt<-table(dt.pred,test_data$Revenue)
cfdt
```

```
##
## dt.pred FALSE TRUE
##   FALSE  1989  160
##   TRUE     95  222
```

Accuracy measures for Decision Tree:

```
dt_precision<- cfdt[1,1]/(sum(cfdt[1,]))
dt_recall<- cfdt[1,1]/(sum(cfdt[,1]))
dt_fscore <- 2*dt_precision*dt_recall/(dt_precision+dt_recall)
dt_precision
```

```
## [1] 0.9255468
```

```r
dt_recall
```

```
## [1] 0.9544146
```

```r
dt_fscore
```

```
## [1] 0.939759
```

KNN

Firstly, we have to prepare the data for analysis. The data preprocessing has to be done before we run any machine learning algorithm. The data preprocessing includes the preparation of the data converting from categorical data to ordinal factors. Creating Binary variables for the *Weekend* column for False as '0' and True as '1'.

Running the KNN algorithm from the "class" package:

```r
dataset_knn <- dataset[c(1:17,19)]
dataset_knn$OperatingSystems <- factor(dataset$OperatingSystems,
                                   order = TRUE,
                                   levels = c(6,3,7,1,5,2,4,8))
dataset_knn$Browser <- factor(dataset$Browser,
                          order = TRUE,
                          levels = c(9,3,6,7,1,2,8,11,4,5,10,13,12))
dataset_knn$Region <- factor(dataset$Region,
                         order = TRUE,
                         levels = c(8,6,3,4,7,1,5,2,9))
dataset_knn$TrafficType <- factor(dataset$TrafficType,
                              order = TRUE,
                              levels = c(12,15,17,18,13,19,3,9,1,6,4,14,11,10,5,2,20,8,7,16))
dataset_knn$Month <- factor(dataset$Month,
                        levels = c('Feb', 'Mar', 'May',
                                   'Jun','Jul','Aug','Sep','Oct','Nov','Dec'),
                        labels = c(2,3,5,6,7,8,9,10,11,12))
dataset_knn$VisitorType<- factor(dataset$VisitorType,
                                 levels = c('Returning_Visitor', 'Other', 'New_Visitor'),
                                 labels = c(1,2,3))
dataset_knn$Weekend <- factor(dataset$Weekend,
                              levels = c('TRUE','FALSE'),
                              labels = c(1,0))
```

Splitting the data for the testing and training data in the ratio 80:20 for the KNN.

```r
set.seed(1233)
splitknn = sample.split(dataset_knn$Revenue_binary, SplitRatio = 0.8)
training_data_knn = subset(dataset_knn, split == TRUE)
test_data_knn = subset(dataset_knn, split == FALSE)
```

Bulding the KNN classifier as the y__pred

```r
library(class)
y_pred = knn(train = training_data_knn[, -18],
             test = test_data_knn[, -18],
             cl = training_data_knn[, 18],
             k = 5,
             prob = TRUE)
```

Building the Confusion Matrix for the K-Nearest Neighbor

```r
cm = table(test_data_knn[, 18], y_pred)
cm
```

```
##    y_pred
##        0    1
##   0 2021   63
##   1  270  112
```

Precision, Recall and F-score for KNN

```r
knn_precision<- cm[1,1]/(sum(cm[1,]))
knn_recall<- cm[1,1]/(sum(cm[,1]))
knn_fscore <- 2*knn_precision*knn_recall/(knn_precision+knn_recall)
knn_precision
```

```
## [1] 0.9697697
```

```r
knn_recall
```

```
## [1] 0.8821475
```

```r
knn_fscore
```

```
## [1] 0.9238857
```

Appendix—Code

```r
knitr::opts_chunk$set(echo= TRUE, warning=FALSE, message=FALSE)
library(ggplot2)
library(tidyverse)
library(gmodels)
library(dplyr)
library(ggmosaic)
library(corrplot)
library(caret)
library(rpart)
library(rpart.plot)
library(cluster)
library(fpc)
```

```r
library(data.table)
library(knitr)
library(kableExtra)
library(plyr)
library(caTools)
dataset <- read.csv("online_shoppers_intention.csv", header = TRUE)
attach(dataset)
ncol(dataset)
nrow(dataset)
str(dataset)
summary(dataset)
tab1 <- read.csv("table1.csv", header = TRUE)
kable(tab1) %>%
  kable_styling(full_width = T)
tab2 <- read.csv("table2.csv", header = TRUE)
kable(tab2) %>%
  kable_styling(full_width = T)
library(gmodels)
summary(dataset$Revenue)
CrossTable(dataset$Revenue)
dataset <- dataset %>%
  mutate(Revenue_binary = ifelse(dataset$Revenue == "TRUE", 1, 0))
colSums(is.na(dataset))
dataset$Month = factor(dataset$Month, levels = month.abb)
dataset %>%
  ggplot() +
  aes(x = Month, Revenue = ..count../nrow(dataset), fill = Revenue) +
  geom_bar() +
  ylab("Frequency")
table_month = table(dataset$Month, dataset$Revenue)
tab_mon =  as.data.frame(prop.table(table_month,2))
colnames(tab_mon) = c("Month", "Revenue", "perc")
ggplot(data = tab_mon, aes(x = Month, y = perc, fill = Revenue)) +
  geom_bar(stat = 'identity', position = 'dodge', alpha = 2/3) +
  xlab("Month")+
  ylab("Percent")
theme_set(theme_bw())

## setting default parameters for mosaic plots
mosaic_theme = theme(axis.text.x = element_text(angle = 90,
                                                hjust = 1,
                                                vjust = 0.5),
                     axis.text.y = element_blank(),
                     axis.ticks.y = element_blank())
dataset %>%
  ggplot() +
  geom_mosaic(aes(x = product(Revenue, VisitorType), fill = Revenue)) +
  mosaic_theme +
  xlab("Visitor Type") +
  ylab(NULL)
CrossTable(dataset$Weekend, dataset$Revenue)
dataset %>%
  ggplot() +
```

```r
  mosaic_theme +
  geom_mosaic(aes(x = product(Revenue,Weekend), fill = Revenue)) +
  xlab("Weekend") +
  ylab(NULL)
dataset_classify = dataset[-c(19:22)]
set.seed(123)
split = sample.split(dataset_classify$Revenue, SplitRatio = 0.8)
training_data = subset(dataset_classify, split == TRUE)
test_data = subset(dataset_classify, split == FALSE)
dt_model<- rpart(Revenue ~ . , data = training_data, method="class")
rpart.plot(dt_model)
dt.pred <- predict(dt_model,test_data,type = "class")
mean(dt.pred==test_data$Revenue)
cfdt<-table(dt.pred,test_data$Revenue)
cfdt
dt_precision<- cfdt[1,1]/(sum(cfdt[1,]))
dt_recall<- cfdt[1,1]/(sum(cfdt[,1]))
dt_fscore <- 2*dt_precision*dt_recall/(dt_precision+dt_recall)
dt_precision
dt_recall
dt_fscore
dataset_knn <- dataset[c(1:17,19)]
dataset_knn$OperatingSystems <- factor(dataset$OperatingSystems,
                                   order = TRUE,
                                   levels = c(6,3,7,1,5,2,4,8))
dataset_knn$Browser <- factor(dataset$Browser,
                          order = TRUE,
                          levels = c(9,3,6,7,1,2,8,11,4,5,10,13,12))
dataset_knn$Region <- factor(dataset$Region,
                         order = TRUE,
                         levels = c(8,6,3,4,7,1,5,2,9))
dataset_knn$TrafficType <- factor(dataset$TrafficType,
                              order = TRUE,
                              levels = c(12,15,17,18,13,19,3,9,1,6,4,14,11,10,5,2,20,8,7,16))
dataset_knn$Month <- factor(dataset$Month,
                        levels = c('Feb', 'Mar', 'May',
                                   'Jun','Jul','Aug','Sep','Oct','Nov','Dec'),
                        labels = c(2,3,5,6,7,8,9,10,11,12))
dataset_knn$VisitorType<- factor(dataset$VisitorType,
                                  levels = c('Returning_Visitor', 'Other', 'New_Visitor'),
                                  labels = c(1,2,3))
dataset_knn$Weekend <- factor(dataset$Weekend,
                              levels = c('TRUE','FALSE'),
                              labels = c(1,0))
set.seed(1233)
splitknn = sample.split(dataset_knn$Revenue_binary, SplitRatio = 0.8)
training_data_knn = subset(dataset_knn, split == TRUE)
test_data_knn = subset(dataset_knn, split == FALSE)
library(class)
y_pred = knn(train = training_data_knn[, -18],
             test = test_data_knn[, -18],
             cl = training_data_knn[, 18],
             k = 5,
```

```
             prob = TRUE)

cm = table(test_data_knn[, 18], y_pred)
cm
knn_precision<- cm[1,1]/(sum(cm[1,]))
knn_recall<- cm[1,1]/(sum(cm[,1]))
knn_fscore <- 2*knn_precision*knn_recall/(knn_precision+knn_recall)
knn_precision
knn_recall
knn_fscore
```