Purchase Intention

2022-12-20

Libraries

```
library(ggplot2)
library(ggcorrplot) #Correlation Matrix
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.3.2 --

## v tibble  3.1.8     v purrr   1.0.0
## v tidyr   1.2.1     v stringr 1.5.0
## v readr   2.1.3     v forcats 0.5.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x gridExtra::combine() masks dplyr::combine()
## x dplyr::filter()      masks stats::filter()
## x dplyr::lag()         masks stats::lag()
```

```
library(gridExtra)
library(RColorBrewer)
library(cowplot)
library(lares)
#install.packages("caret")
library(caret) #data partition
```

```
## Loading required package: lattice
##
## Attaching package: 'caret'
##
## The following object is masked from 'package:purrr':
##
##     lift
```

```
library(rpart)
library(rpart.plot)
```

```
online_shoppers_intention = read.csv('/Users/varunthallapelly/Desktop/MSc Business Analytics/Dissertati
onl=online_shoppers_intention
onll = onl
```

```
onl1 = onll
```

Finding Missing Values

```
colSums(is.na(onl1))
```

```
##          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
```

Attributes and Observation

```
dim(onl1)
```

```
## [1] 12330    18
```

There is no missing values in the following data set and there are 12,330 Observation and 18 Attributes # Structure of Dataset as per my requirement.

Identifying Duplicate Data

```
sum(duplicated(onl1))
```

```
## [1] 125
```

Remove the Duplicate and retrieve only the Unique variables.

```
onl1 = unique(onl1)
sum(duplicated(onl1))
```

```
## [1] 0
```

```
onll = onl1
```

Changing the Revenue and Weekend Observation

TRUE - 1 FALSE - 0

```
onl1$Revenue = gsub(FALSE, 0, onl1$Revenue)
onl1$Revenue = gsub(TRUE, 1, onl1$Revenue)

onl1$Weekend = gsub(FALSE, 0, onl1$Weekend)
onl1$Weekend = gsub(TRUE, 1, onl1$Weekend)
```

Changing Weekend, Revenue, Month, Visitor Type to factorial

```
onl1$Weekend = as.factor(onl1$Weekend)
onl1$Revenue = as.factor(onl1$Revenue)
onl1$Month = as.factor(onl1$Month)
onl1$VisitorType = as.factor(onl1$VisitorType)
```

Operating System

The Operating System is in numerical and needs to be renamed based on Top 8 Operating system in 2022.

```
onl1$OperatingSystems[onl1$OperatingSystems == 1] = "MS-Windows"
onl1$OperatingSystems[onl1$OperatingSystems == 2] = "Ubuntu"
onl1$OperatingSystems[onl1$OperatingSystems == 3] = "Mac OS"
onl1$OperatingSystems[onl1$OperatingSystems == 4] = "Fedora"
onl1$OperatingSystems[onl1$OperatingSystems == 5] = "Solaris"
onl1$OperatingSystems[onl1$OperatingSystems == 6] = "Free BSD"
onl1$OperatingSystems[onl1$OperatingSystems == 7] = "Chrome OS"
onl1$OperatingSystems[onl1$OperatingSystems == 8] = "CentOS"
```

Browser

The Browser is in numerical and needs to be renamed based on Top 15 browser in 2022.

```
onl1$Browser[onl1$Browser == 1] = "Safari"
onl1$Browser[onl1$Browser == 2] = "Microsoft Edge"
onl1$Browser[onl1$Browser == 3] = "Google Chrome"
onl1$Browser[onl1$Browser == 4] = "Opera"
onl1$Browser[onl1$Browser == 5] = "Firefox"
onl1$Browser[onl1$Browser == 6] = "Brave"
onl1$Browser[onl1$Browser == 7] = "Vivaldi"
onl1$Browser[onl1$Browser == 8] = "Torch"
onl1$Browser[onl1$Browser == 9] = "Avast Secure"
onl1$Browser[onl1$Browser == 10] = "UR Browser"
onl1$Browser[onl1$Browser == 11] = "Aloha Browser"
onl1$Browser[onl1$Browser == 12] = "Epic Privacy Browser"
onl1$Browser[onl1$Browser == 13] = "Slim Browser"
```

Region

The region is in numerical and considering the considering the highest buying online region in 2022.

```
onl1$Region[onl1$Region == 1] = "China"
onl1$Region[onl1$Region == 2] = "United Kingdom"
onl1$Region[onl1$Region == 3] = "South Korea"
onl1$Region[onl1$Region == 4] = "Denmark"
onl1$Region[onl1$Region == 5] = "Indonesia"
onl1$Region[onl1$Region == 6] = "Norway"
onl1$Region[onl1$Region == 7] = "United States"
onl1$Region[onl1$Region == 8] = "Finland"
onl1$Region[onl1$Region == 9] = "Sweden"
```

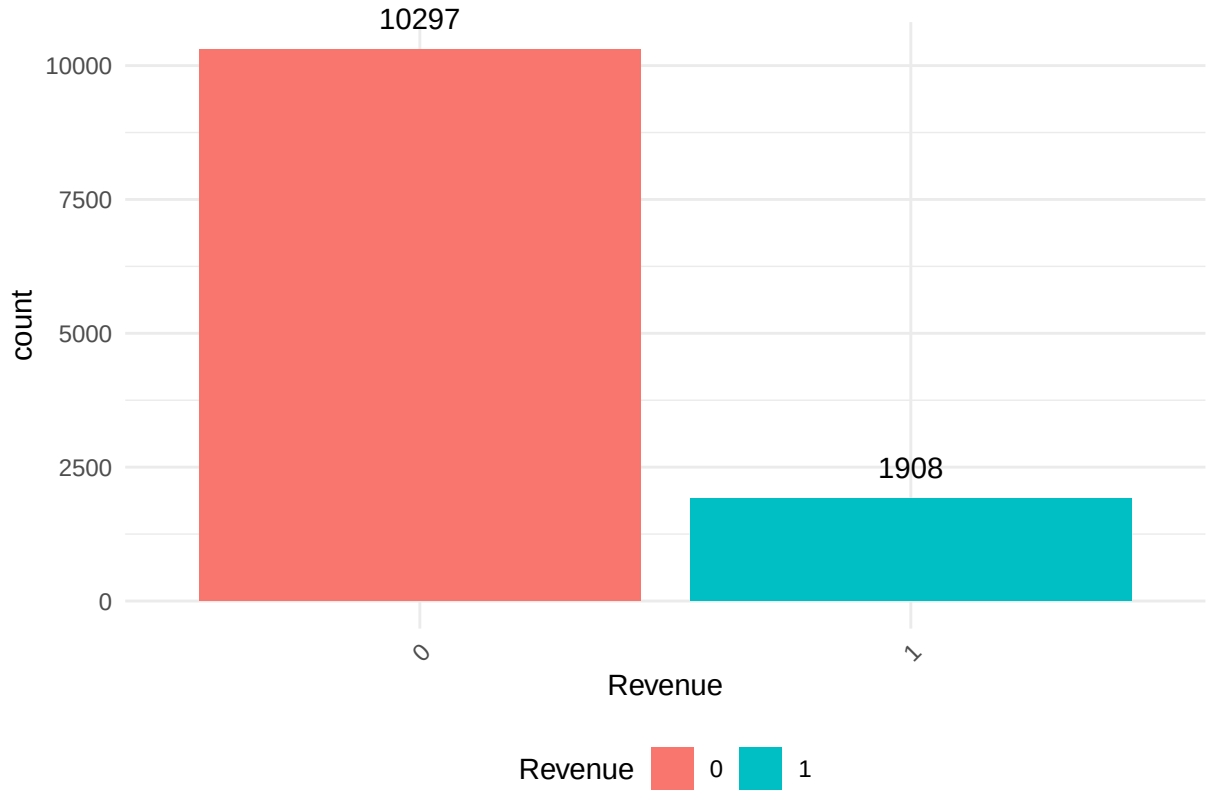
Created new table with Revenue == 1

```
#Remove rows where gender not equal to 'm'
db = subset(onl1,Revenue == 1 )
#db
```

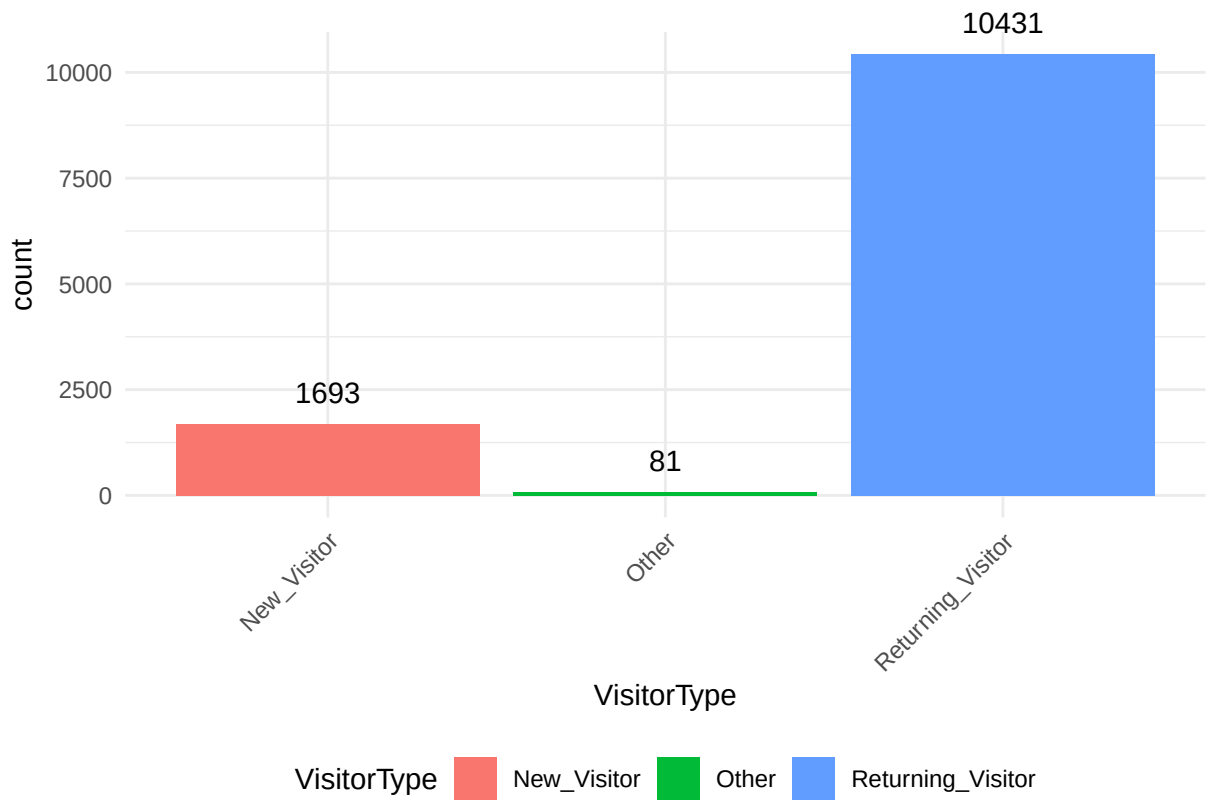
Descprtive Analysis

```
Revenue = ggplot(data=onl1, aes(x=Revenue, fill = Revenue)) +
                    geom_bar() +
                    geom_text(stat='count', aes(label=..count..), vjust=-1)+
                    coord_cartesian(clip = "off")+
                    theme_minimal()+
                    theme(axis.text.x = element_text(angle=45, hjust=1, vjust = 1),
                          legend.position = "bottom",
                          plot.margin = margin(t = 20, r = 10, b = 10, l = 10))


Revenue
```

```
## Warning: The dot-dot notation (`..count..`) was deprecated in ggplot2 3.4.0.
## i Please use `after_stat(count)` instead.
```



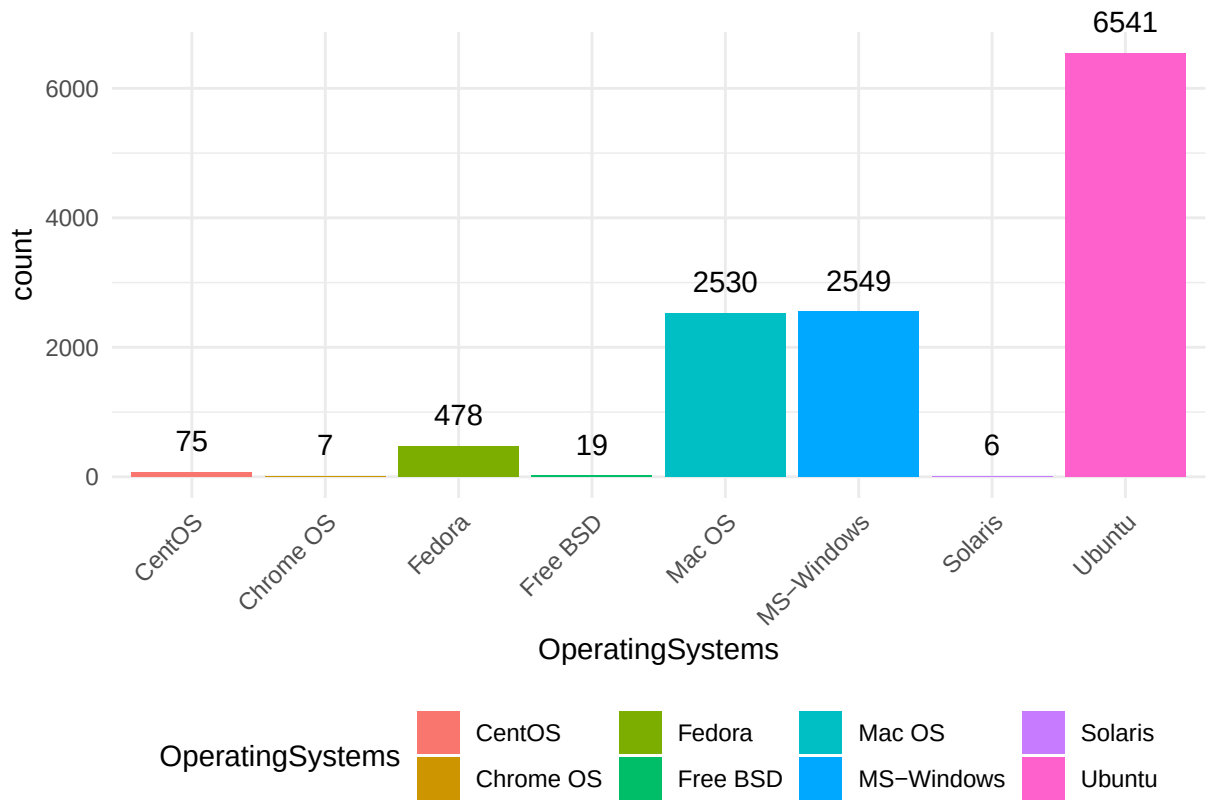

```
VisitorType = ggplot(data=onl1, aes(x=VisitorType, fill = VisitorType)) +
                    geom_bar() +
                    geom_text(stat='count', aes(label=..count..), vjust=-1)+
                    coord_cartesian(clip = "off")+
                    theme_minimal()+
                    theme(axis.text.x = element_text(angle=45, hjust=1, vjust = 1),
                          legend.position = "bottom",
                          plot.margin = margin(t = 20, r = 10, b = 10, l = 10))


VisitorType
```




```
Weekend = ggplot(data=onl1, aes(x=Weekend, fill = Weekend)) +
                  geom_bar() +
                  geom_text(stat='count', aes(label=..count..), vjust=-1)+
                  coord_cartesian(clip = "off")+
                  theme_minimal()+
                  theme(axis.text.x = element_text(angle=45, hjust=1, vjust = 1),
                        legend.position = "bottom",
                        plot.margin = margin(t = 20, r = 10, b = 10, l = 10))


Weekend
```

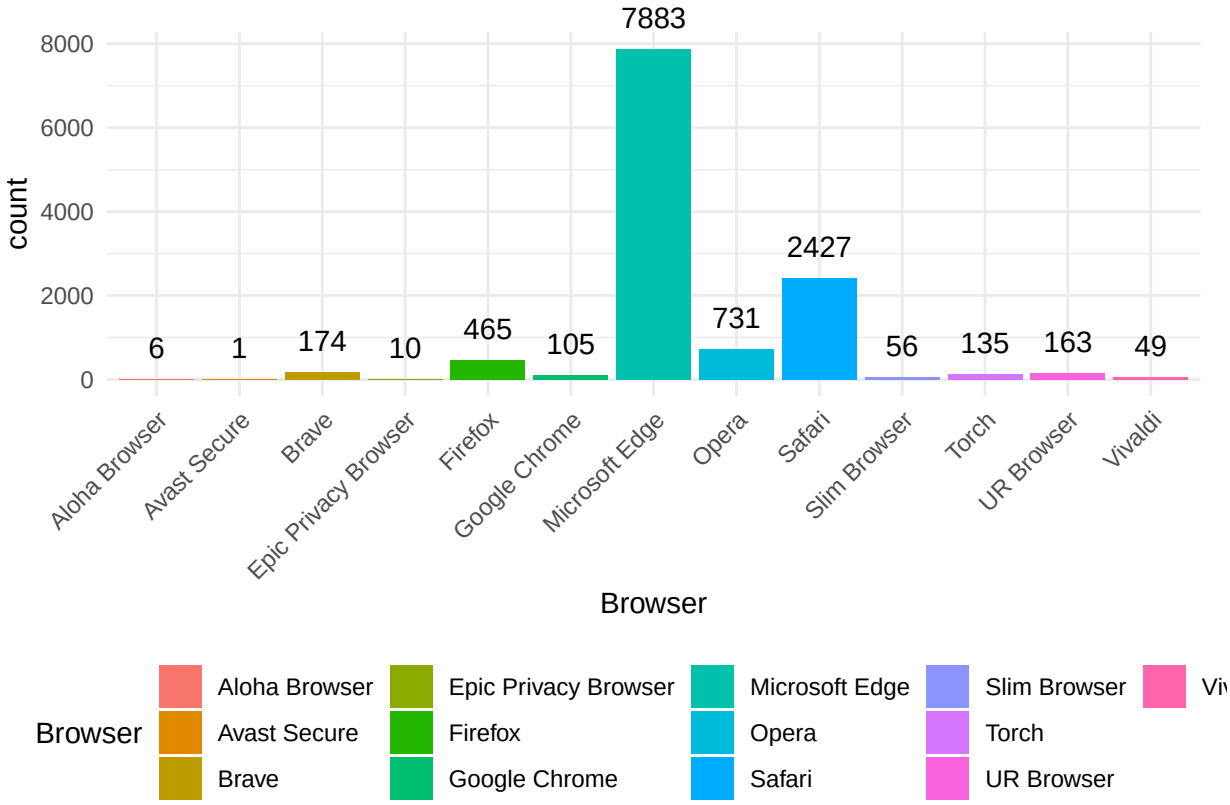
Operating System - Bar Graph

```
Operating_system = ggplot(data=onl1, aes(x = OperatingSystems, fill = OperatingSystems)) +
                   geom_bar() +
                   geom_text(stat='count', aes(label=..count..), vjust=-1)+
                   coord_cartesian(clip = "off")+
                   theme_minimal()+
                   theme(axis.text.x = element_text(angle=45, hjust=1, vjust = 1),
                         legend.position = "bottom",
                         plot.margin = margin(t = 20, r = 10, b = 10, l = 10))


Operating_system
```



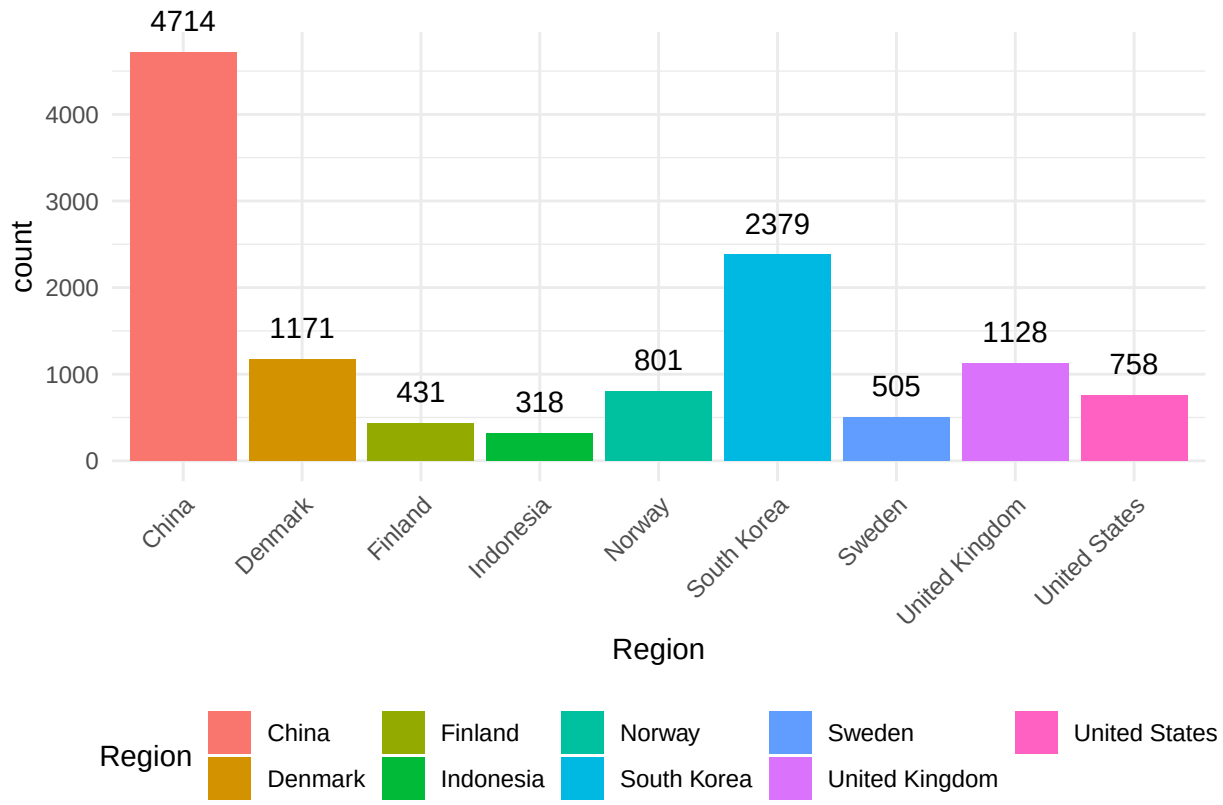
Browser - Bar Graph

```
Browser = ggplot(data=onl1, aes(x=Browser, fill = Browser)) +
         geom_bar() +
         geom_text(stat='count', aes(label=..count..), vjust=-1)+
         coord_cartesian(clip = "off")+
        theme_minimal()+
        theme(axis.text.x = element_text(angle=45, hjust=1, vjust = 1),
              legend.position = "bottom",
              plot.margin = margin(t = 20, r = 10, b = 10, l = 10))

Browser
```



Region - Bar Graph

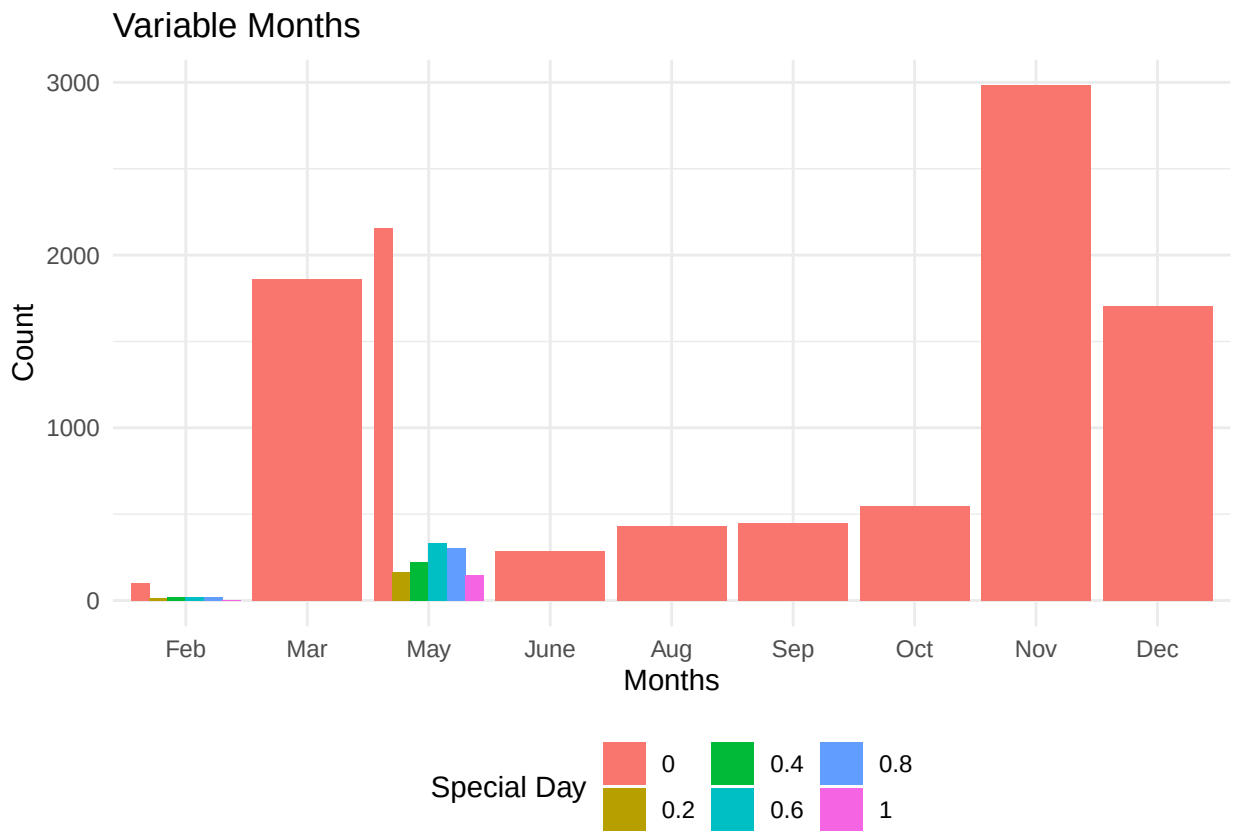
```
Region = ggplot(data=onl1, aes(x=Region, fill = Region)) +
         geom_bar() +
         geom_text(stat='count', aes(label=..count..), vjust=-1)+
         coord_cartesian(clip = "off")+
        theme_minimal()+
        theme(axis.text.x = element_text(angle=45, hjust=1, vjust = 1),
              legend.position = "bottom",
              plot.margin = margin(t = 20, r = 10, b = 10, l = 10))

Region
```



Months Vs Special Days

```
MvsSD = ggplot(onl1, aes(x=Month, fill = as.factor(SpecialDay))) +
  geom_bar(position="dodge") +
  labs(y = "Count",
       x = "Months",
       title = "Variable Months",
       fill = "Special Day") +
  scale_x_discrete(limits=c("Feb","Mar","May", "June","Aug","Sep","Oct","Nov","Dec"))+
  theme_minimal()+
  theme(legend.position = "bottom")

MvsSD
```

```
## Warning: Removed 432 rows containing non-finite values (‘stat_count()‘).
```



Summary of the data set

```
summary(onl1[,c(1:10)])
```

```
##  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 :   9.00         Median : 0.0000
##  Mean   : 2.339   Mean   :  81.65         Mean   : 0.5087
##  3rd Qu.: 4.000   3rd Qu.:  94.70         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.:  8.00   1st Qu.:  193.0
##  Median :   0.00        Median : 18.00   Median :  608.9
##  Mean   :  34.83        Mean   : 32.05   Mean   : 1207.0
##  3rd Qu.:   0.00        3rd Qu.: 38.00   3rd Qu.: 1477.2
##  Max.   :2549.38        Max.   :705.00   Max.   :63973.5
##   BounceRates         ExitRates         PageValues       SpecialDay
##  Min.   :0.000000   Min.   :0.00000   Min.   :  0.00   Min.   :0.00000
##  1st Qu.:0.000000   1st Qu.:0.01423   1st Qu.:  0.00   1st Qu.:0.00000
##  Median :0.002899   Median :0.02500   Median :  0.00   Median :0.00000
##  Mean   :0.020370   Mean   :0.04147   Mean   :  5.95   Mean   :0.06194
##  3rd Qu.:0.016667   3rd Qu.:0.04853   3rd Qu.:  0.00   3rd Qu.:0.00000
##  Max.   :0.200000   Max.   :0.20000   Max.   :361.76   Max.   :1.00000
```

Exploratory Analysis

Time Series for Revenue based on Region

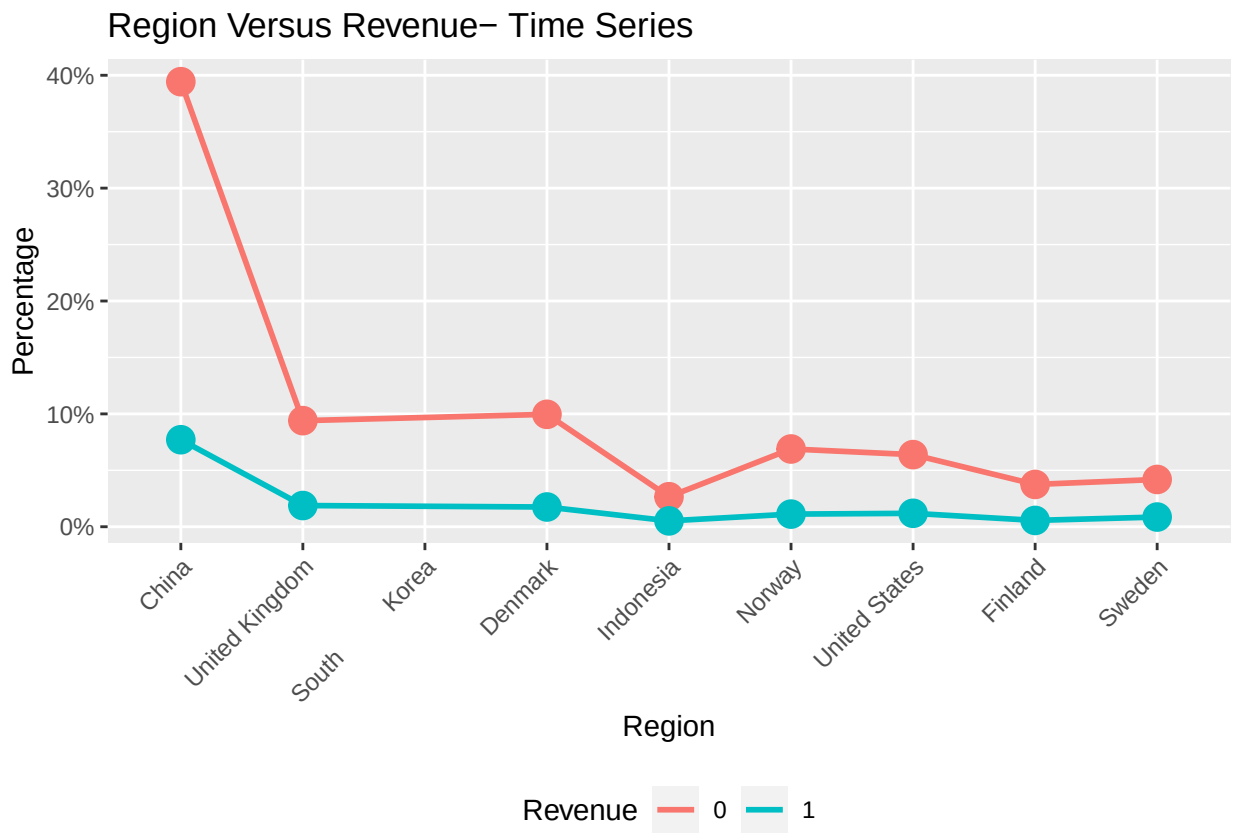
```
options(repr.plot.width = 7, repr.plot.height = 5)

trend <- data.frame(table(onl1$Region, onl1$Revenue))
names(trend) <- c("Region", "Revenue", "Frequency")

RvsR = ggplot(data = trend, mapping = aes(x = Region , y = Frequency)) +
       geom_line(mapping = aes(color = Revenue, group = Revenue), lwd = 1) +
       geom_point(mapping = aes(color = Revenue, group = Revenue, size = 0.1), show.legend = FALSE) +
       scale_y_continuous(labels = scales::percent_format(scale = 0.01)) +
       scale_x_discrete(limits=c("China","United Kingdom","South          Korea","Denmark","Indonesia",
       labs(x = "Region",
            fill = "Revenue",
            y = "Percentage",
            title = "Region Versus Revenue- Time Series" )+
       theme_grey()+
       theme(axis.text.x = element_text(angle=45, hjust=1, vjust = 1),
               legend.position = "bottom")
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
```

```
RvsR
```

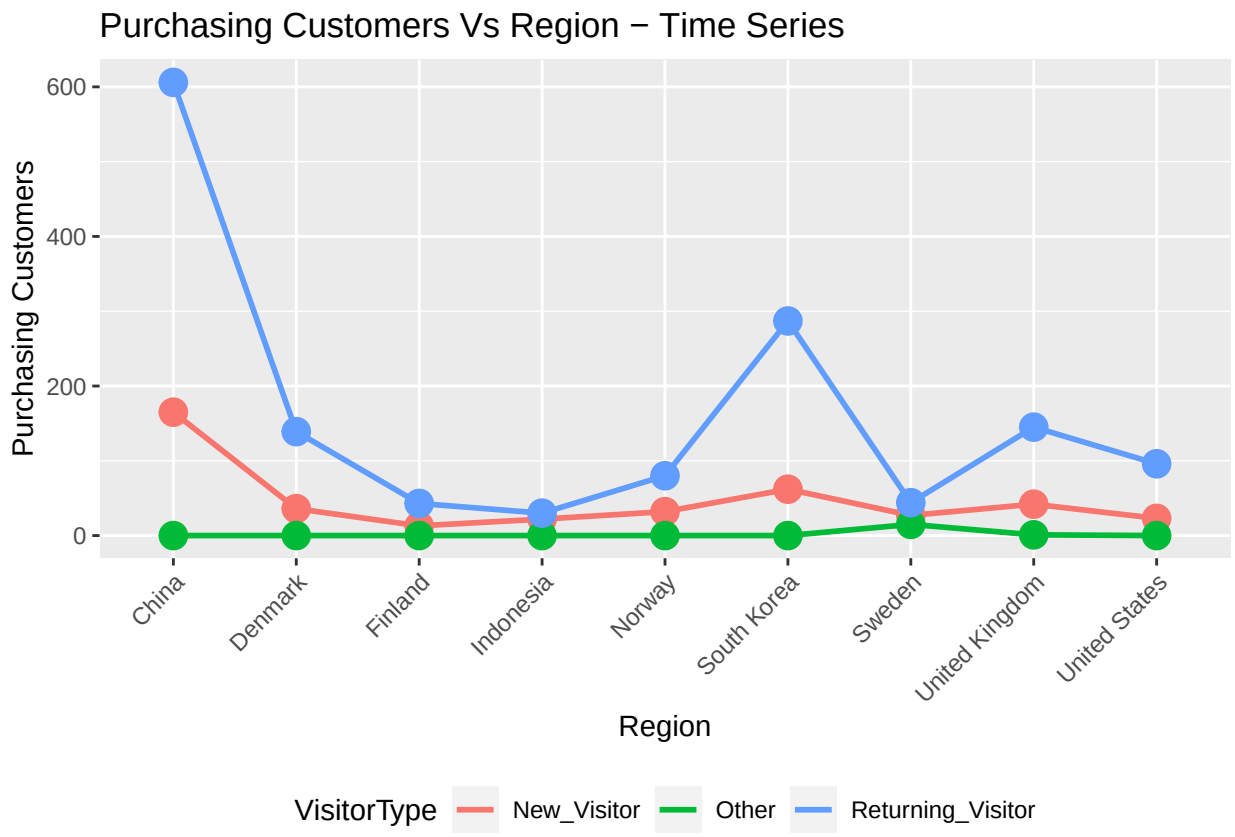
```
## Warning: Removed 2 rows containing missing values (‘geom_line()‘).
```

```
## Warning: Removed 2 rows containing missing values (‘geom_point()‘).
```



Time Series for Visitor Type based on Region

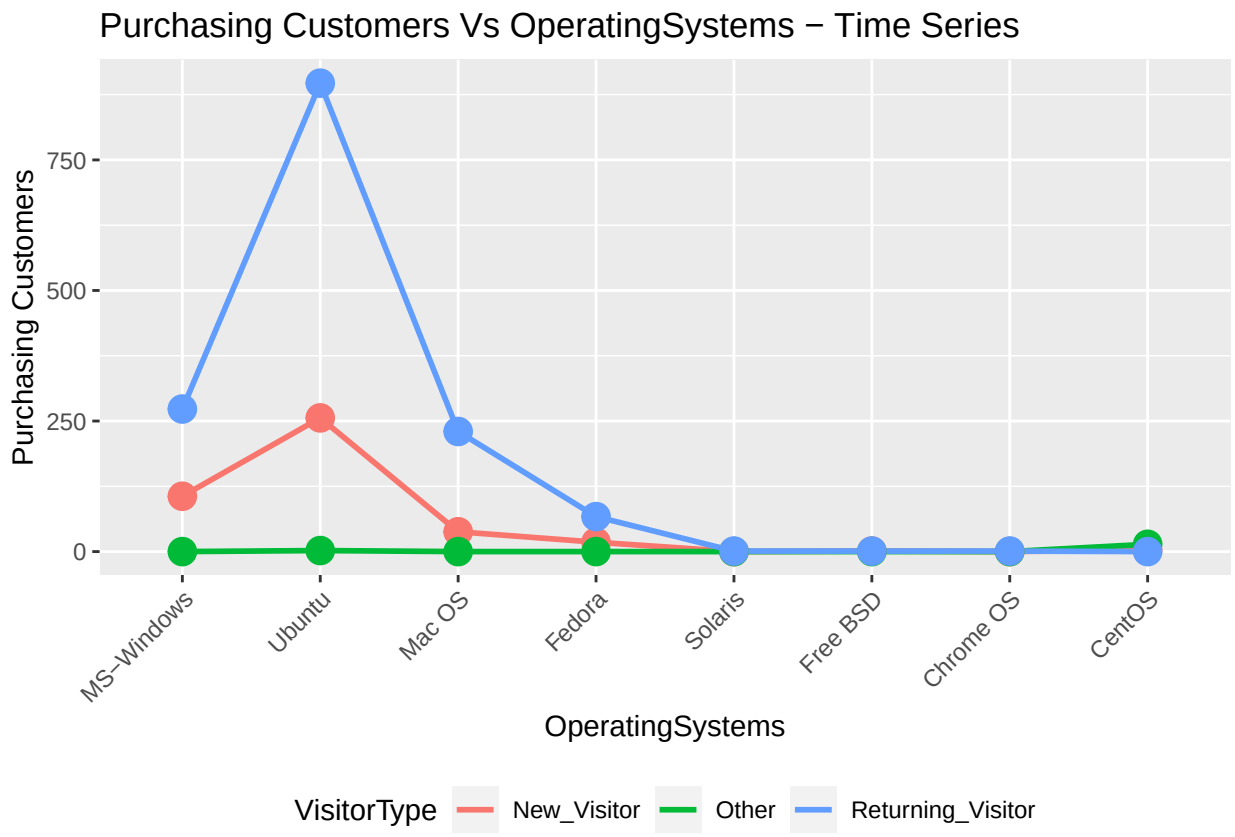
```
trendRegion <- data.frame(table(db$Region, db$VisitorType))
names(trendRegion) <- c("Region", "VisitorType", "Frequency")

PvsR = ggplot(data = trendRegion, mapping = aes(x = Region, y = Frequency)) +
       geom_line(mapping = aes(color = VisitorType, group = VisitorType), lwd = 1) +
       geom_point(mapping = aes(color = VisitorType, group = VisitorType, size = 0.1), show.legend = FA
       labs(x = "Region",
            fill = "Revenue",
            y = "Purchasing Customers",
           title = "Purchasing Customers Vs Region - Time Series" )+
       theme_grey()+
       theme(axis.text.x = element_text(angle=45, hjust=1, vjust = 1),
             legend.position = "bottom")

PvsR
```




```
#grid.arrange(RvsR, d, ncol = 2, nrow = 1)
```

Time Series for Visitor Type based on Operating Systems

```
trendOS <- data.frame(table(db$OperatingSystems, db$VisitorType))
names(trendOS) <- c("OperatingSystems", "VisitorType", "Frequency")

PvsOS = ggplot(data = trendOS, mapping = aes(x = OperatingSystems, y = Frequency)) +
        geom_line(mapping = aes(color = VisitorType, group = VisitorType), lwd = 1) +
        geom_point(mapping = aes(color = VisitorType, group = VisitorType, size = 0.1), show.legend = F
        scale_x_discrete(limits = c("MS-Windows","Ubuntu","Mac OS","Fedora","Solaris","Free BSD","Chrom
        labs(x = "OperatingSystems",
             fill = "Revenue",
             y = "Purchasing Customers",
             title = "Purchasing Customers Vs OperatingSystems - Time Series" )+
        theme_grey()+
        theme(axis.text.x = element_text(angle=45, hjust=1, vjust = 1),
                legend.position = "bottom")

PvsOS
```

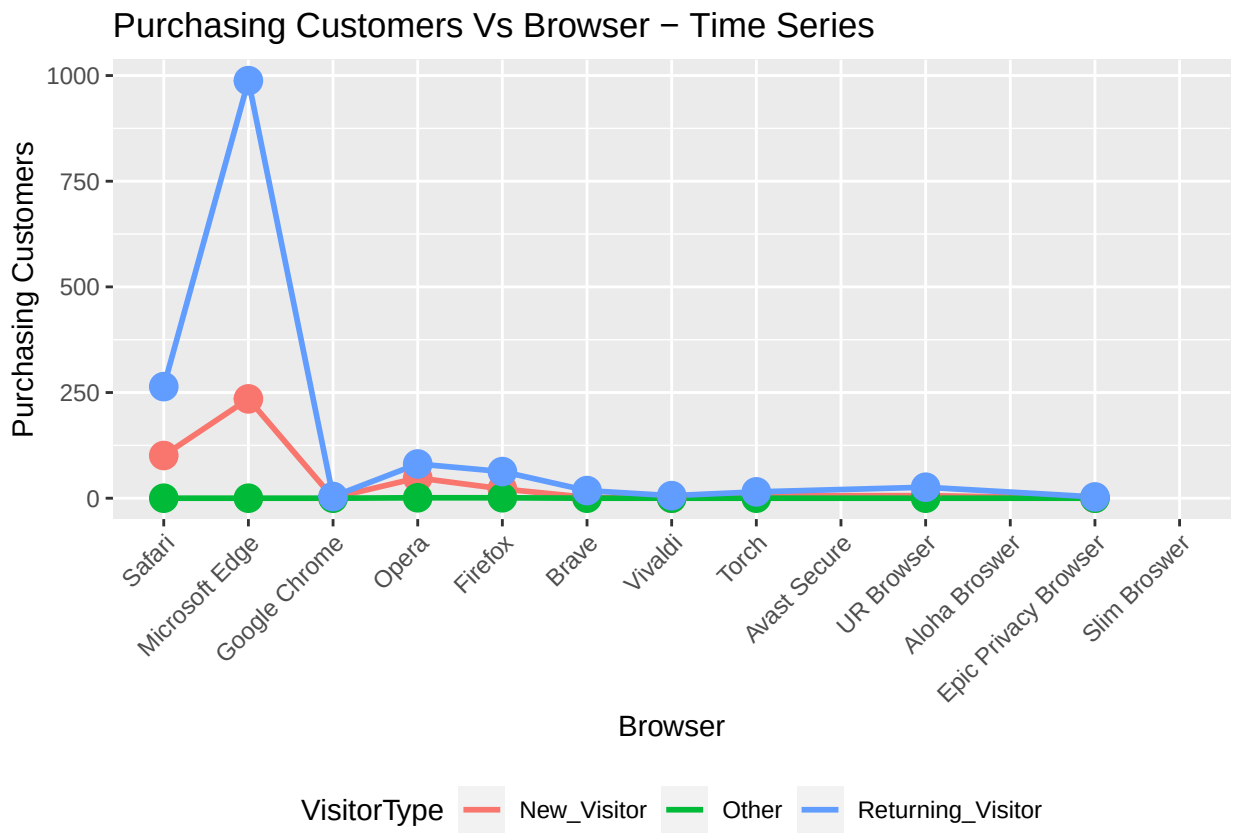


Time Series for Visitor Type based on Browser

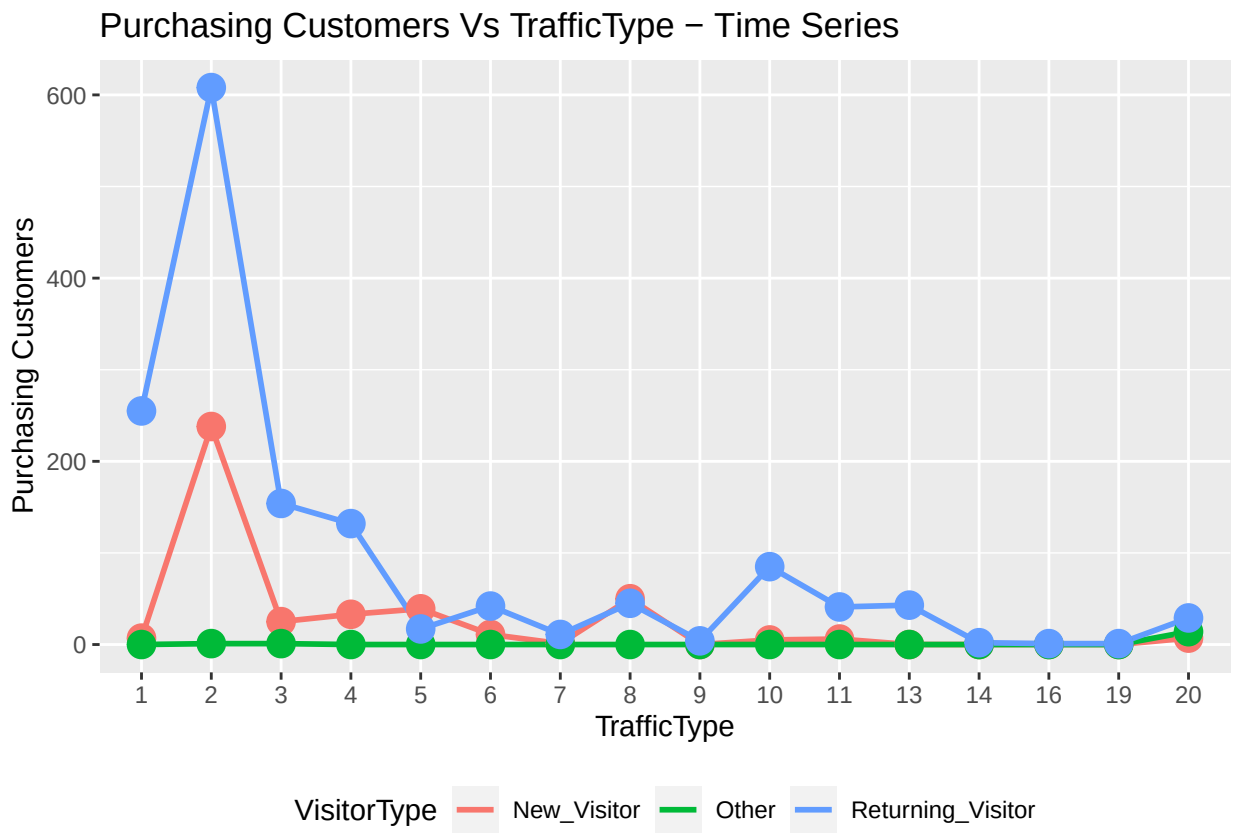
```
trendBrowser <- data.frame(table(db$Browser, db$VisitorType))
names(trendBrowser) <- c("Browser", "VisitorType", "Frequency")

PvsB = ggplot(data = trendBrowser, mapping = aes(x = Browser, y = Frequency)) +
       geom_line(mapping = aes(color = VisitorType, group = VisitorType), lwd = 1) +
       geom_point(mapping = aes(color = VisitorType, group = VisitorType, size = 0.1), show.legend = FA
       scale_x_discrete(limits = c("Safari","Microsoft Edge","Google Chrome","Opera","Firefox","Brave",
       labs(x = "Browser",
            fill = "Revenue",
            y = "Purchasing Customers",
            title = "Purchasing Customers Vs Browser - Time Series" ) +
            theme_grey()+
            theme(axis.text.x = element_text(angle=45, hjust=1, vjust = 1),
                  legend.position = "bottom")
PvsB
```

```
## Warning: Removed 6 rows containing missing values (‘geom_line()‘).
```

```
## Warning: Removed 6 rows containing missing values (‘geom_point()‘).
```



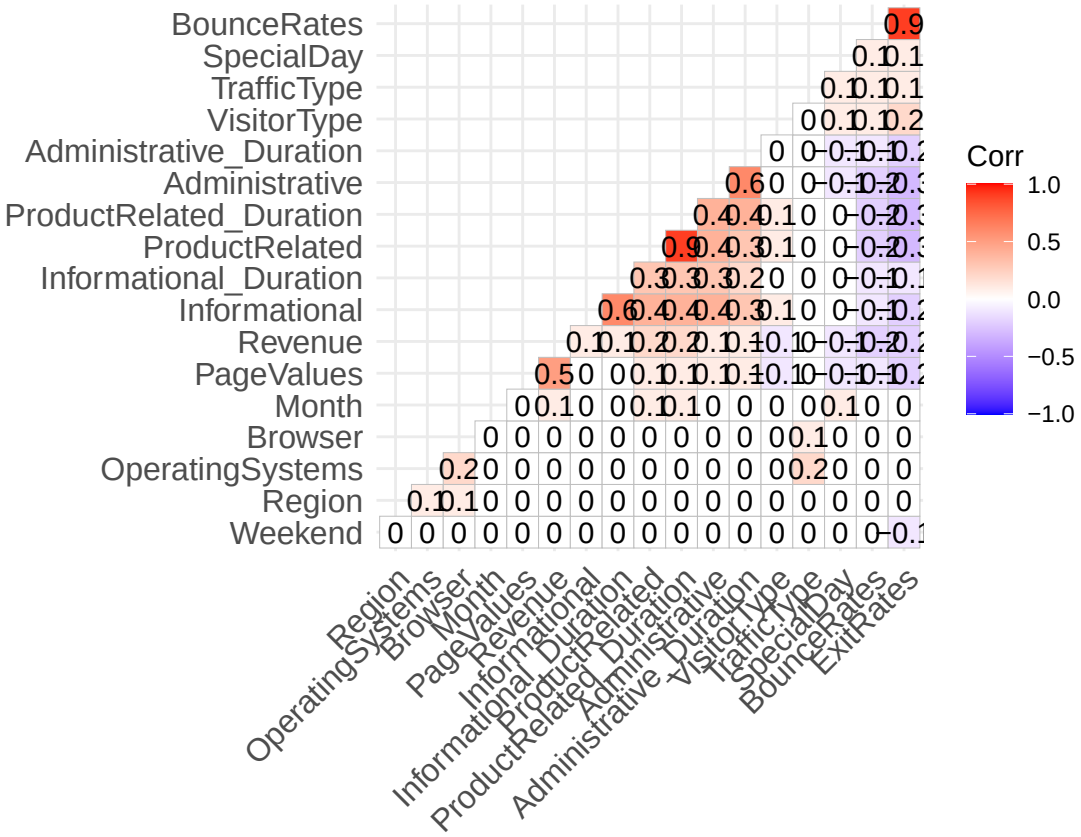

```
trendTrafficType <- data.frame(table(db$TrafficType, db$VisitorType))
names(trendTrafficType) <- c("TrafficType", "VisitorType", "Frequency")
PvsT = ggplot(data = trendTrafficType, mapping = aes(x = TrafficType, y = Frequency)) +
  geom_line(mapping = aes(color = VisitorType, group = VisitorType), lwd = 1) +
  geom_point(mapping = aes(color = VisitorType, group = VisitorType, size = 0.1), show.legend = FALSE) +
  labs(x = "TrafficType",
       fill = "Revenue",
       y = "Purchasing Customers",
       title = "Purchasing Customers Vs TrafficType - Time Series" )+
    theme_grey()+
  theme(legend.position = "bottom")

PvsT
```



grid.arrange(RvsR, PvsT, PvsB, PvsOS, ncol = 2, nrow = 2)

Correlation Matrix

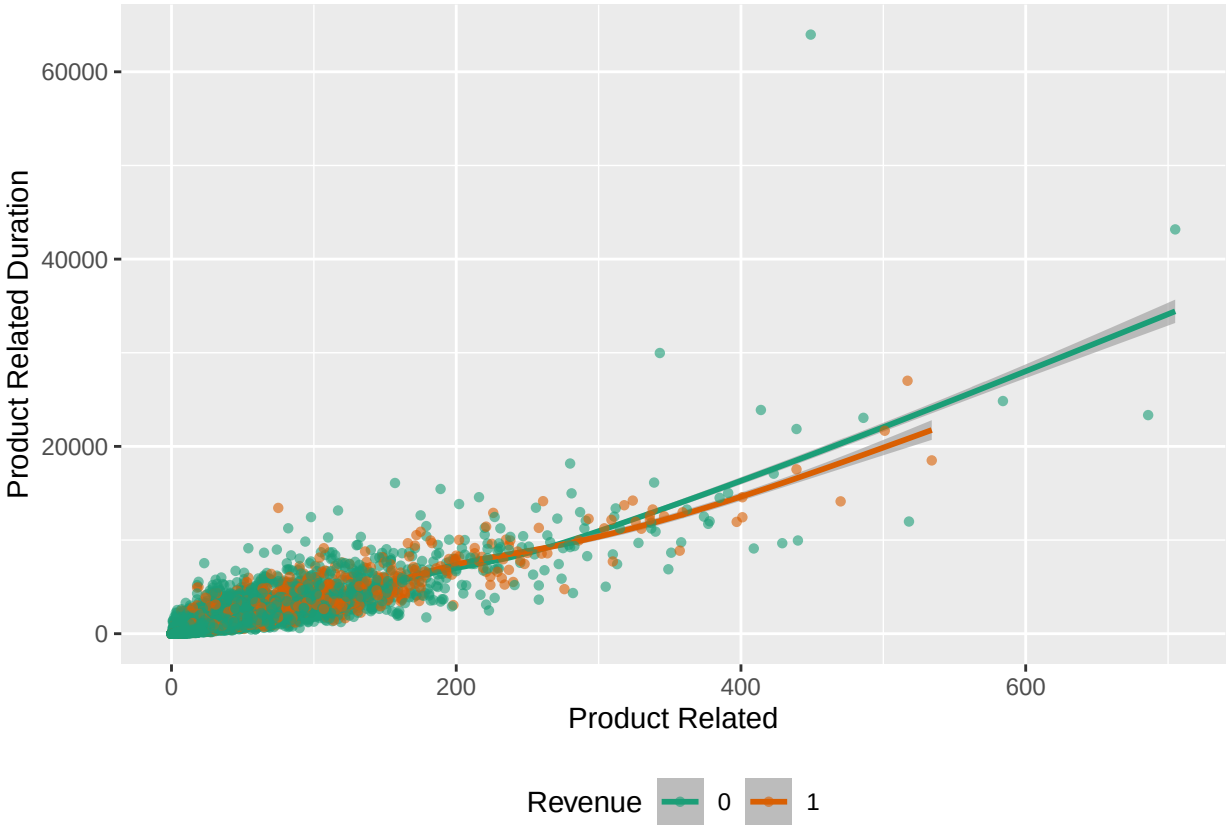
```
library(ggcorrplot)
cor1 = onl
cor1$Month = as.numeric(as.factor(cor1$Month))
cor1$VisitorType = as.numeric(as.factor(cor1$VisitorType))
cor1$Weekend = as.numeric(as.factor(cor1$Weekend))
cor1$Revenue = as.numeric(cor1$Revenue)

# Computing correlation matrix
correlation_matrix <- round(cor(cor1),1)

# Computing correlation matrix with p-values
corrp.mat <- cor_pmat(cor1)

# Adding correlation significance level
ggcorrplot(correlation_matrix, hc.order =TRUE,
           type ="lower", lab =TRUE)
```



Product Related Vs Product Related Duration

```
PRDvsPR = ggplot(onl1) +
  geom_smooth(aes(x = ProductRelated, y = ProductRelated_Duration, color = Revenue), alpha = 0.6) +
  geom_point(aes(x = ProductRelated, y = ProductRelated_Duration, color = Revenue), alpha = 0.6,shape =
  scale_color_brewer(palette = "Dark2") +
  theme_replace() +
  theme(legend.position = "bottom") +
  labs(x = "Product Related", y = "Product Related Duration")
PRDvsPR
```

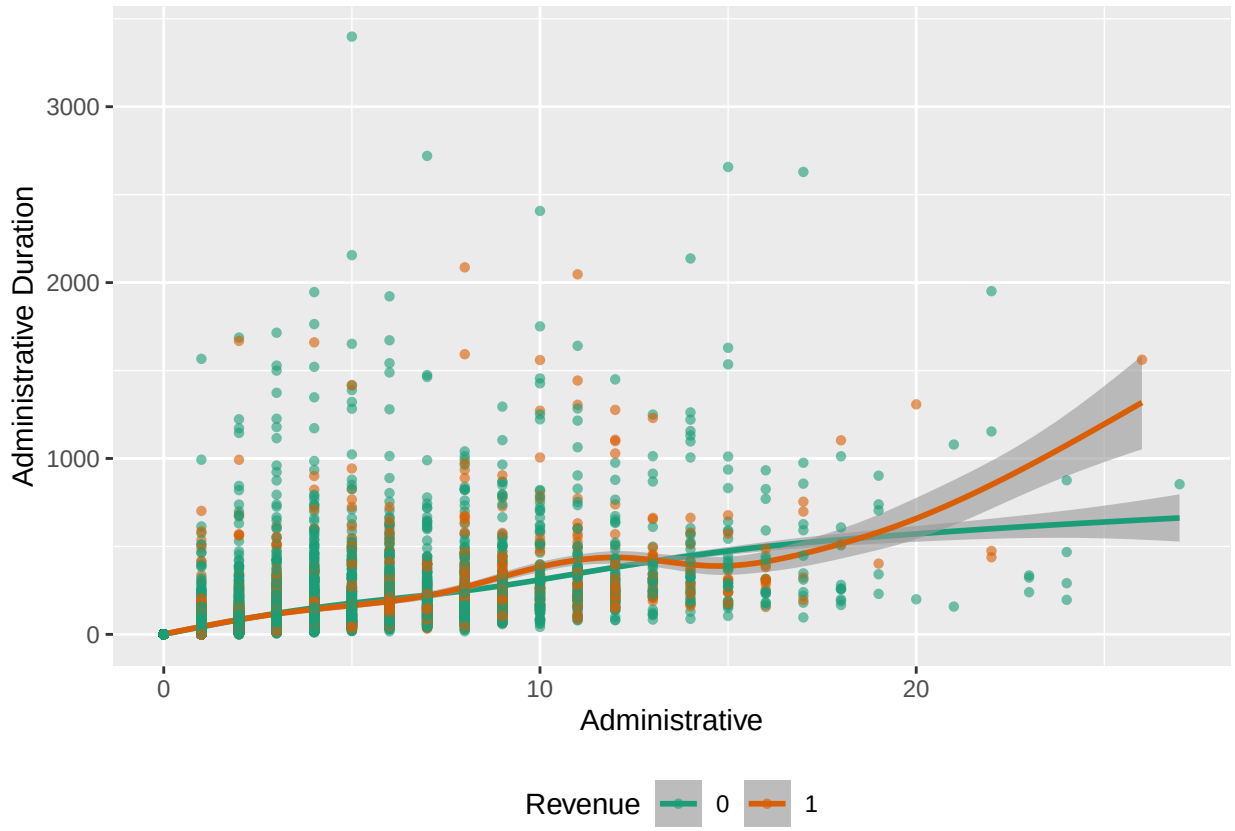
```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```



Administrative vs Administrative Duration

```
PRDvsPR = ggplot(onl1) +
  geom_smooth(aes(x = Administrative, y = Administrative_Duration, color = Revenue), alpha = 0.6) +
  geom_point(aes(x = Administrative, y = Administrative_Duration, color = Revenue), alpha = 0.6, shape =
  scale_color_brewer(palette = "Dark2") +
  theme_replace() +
  theme(legend.position = "bottom") +
  labs(x = "Administrative", y = "Administrative Duration")
PRDvsPR
```

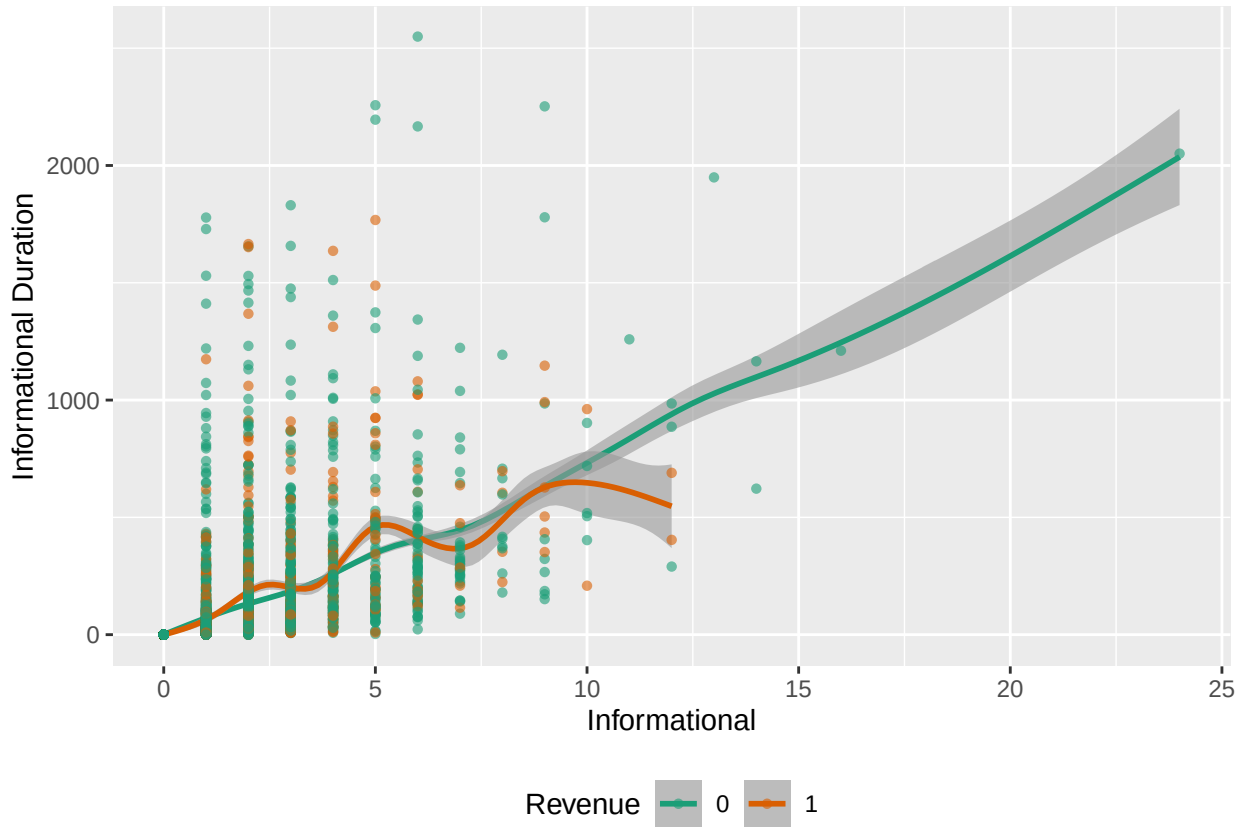
```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```



Informational vs Informational Duration

```
PRDvsPR = ggplot(onl1) +
  geom_smooth(aes(x = Informational, y = Informational_Duration, color = Revenue), alpha = 0.6) +
  geom_point(aes(x = Informational, y = Informational_Duration, color = Revenue), alpha = 0.6, shape =
  scale_color_brewer(palette = "Dark2") +
  theme_replace() +
  theme(legend.position = "bottom") +
  labs(x = "Informational", y = "Informational Duration")
PRDvsPR
```

```
## `geom_smooth()` using method = 'gam' and formula = 'y ~ s(x, bs = "cs")'
```



MODELLING

```
d = na.omit(onl)
onl = unique(onl)
sum(duplicated(onl))
```

```
## [1] 0
```

```
#Analysis of variance
onl2 = onl
lg_mod = glm(Revenue ~. ,data=onl2,family = binomial(link = "logit"))

anova(lg_mod,test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: Revenue
##
## Terms added sequentially (first to last)
##
##
##                          Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
```

```
## NULL                                    12204    10582.5
## Administrative             1   199.93    12203    10382.6 < 2.2e-16 ***
## Administrative_Duration    1     1.64    12202    10381.0 0.2000474
## Informational              1    18.98    12201    10362.0 1.319e-05 ***
## Informational_Duration     1     0.88    12200    10361.1 0.3473824
## ProductRelated             1    88.69    12199    10272.4 < 2.2e-16 ***
## ProductRelated_Duration    1     6.98    12198    10265.4 0.0082211 **
## BounceRates                1   386.15    12197     9879.3 < 2.2e-16 ***
## ExitRates                  1   305.80    12196     9573.5 < 2.2e-16 ***
## PageValues                 1  2132.59    12195     7440.9 < 2.2e-16 ***
## SpecialDay                 1    21.80    12194     7419.1 3.022e-06 ***
## Month                      9   239.85    12185     7179.2 < 2.2e-16 ***
## OperatingSystems           1     4.14    12184     7175.1 0.0418434 *
## Browser                    1     4.63    12183     7170.5 0.0314997 *
## Region                     1     0.80    12182     7169.7 0.3712997
## TrafficType                1     0.15    12181     7169.5 0.6961835
## VisitorType                2    14.87    12179     7154.6 0.0005915 ***
## Weekend                    1     2.06    12178     7152.6 0.1511590
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

from the above table we see the significance of variance

```
#creating new data with respect to feature selection
fs = onl2
fs = subset(fs, select = -c(Administrative_Duration,Informational_Duration,Region,TrafficType,Weekend) )
```

```
#NAIVE BAYES - model1
fs$Revenue<- as.factor(fs$Revenue)

splitting = createDataPartition(fs$Revenue, p=0.75, list=FALSE)
train1 = fs[ splitting,]
test1 = fs[-splitting,]

#over sampling the dataset
#install.packages("ROSE")
library(ROSE)
```

```
## Loaded ROSE 0.0-4
```

```
oversampling_onl = ovun.sample( Revenue ~ ., data = train1, method = "over", N = 10108)$data

x=train1
y=train1$Revenue
library(e1071)
```

```
##
## Attaching package: 'e1071'
```

```
## The following object is masked from 'package:lares':
##
##     impute
```

```
model1 = naiveBayes(x,y)
p<- predict(model1,test1,type="class")

confusionMatrix(p,test1$Revenue)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
##      FALSE  2488   14
##      TRUE     86  463
##
##                Accuracy : 0.9672
##                  95% CI : (0.9603, 0.9733)
##     No Information Rate : 0.8437
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.883
##
##  Mcnemar's Test P-Value : 1.248e-12
##
##             Sensitivity : 0.9666
##             Specificity : 0.9706
##          Pos Pred Value : 0.9944
##          Neg Pred Value : 0.8434
##              Prevalence : 0.8437
##          Detection Rate : 0.8155
##    Detection Prevalence : 0.8201
##       Balanced Accuracy : 0.9686
##
##        'Positive' Class : FALSE
##
```

```
#support vector machine (SVM) - model2

# Split the data into a training set and a test set
index2 = createDataPartition(fs$Revenue, p = 0.75, list = FALSE)
train2 = fs[index2, ]
test2 = fs[-index2, ]

oversampling_onl = ovun.sample( Revenue ~ ., data = train2, method = "over", N = 10108)$data

# Training the model
model2 <- train(
  Revenue ~ .,
  data = train2,
  method = "svmLinear",
  trControl = trainControl(method = "cv", number = 5),
  tuneLength = 5
)

# Make predictions on the test set
predictions = predict(model2, test2)
```

```r
# Evaluate the model
confusionMatrix(predictions, test2$Revenue)
```

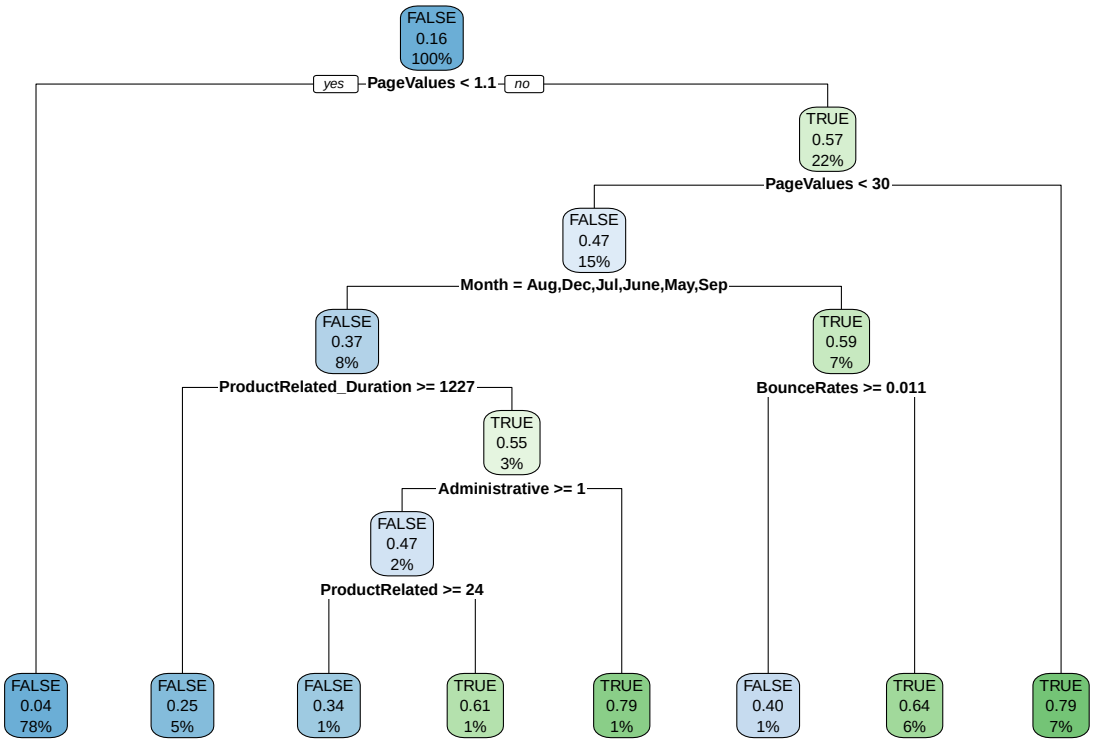
```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
##      FALSE  2512  301
##      TRUE     62  176
##
##                Accuracy : 0.881
##                  95% CI : (0.869, 0.8923)
##     No Information Rate : 0.8437
##     P-Value [Acc > NIR] : 2.352e-09
##
##                   Kappa : 0.4333
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9759
##             Specificity : 0.3690
##          Pos Pred Value : 0.8930
##          Neg Pred Value : 0.7395
##              Prevalence : 0.8437
##          Detection Rate : 0.8233
##    Detection Prevalence : 0.9220
##       Balanced Accuracy : 0.6724
##
##        'Positive' Class : FALSE
##
```

```r
#DECISION TREE - model3
# Split the data into a training set and a test set
index3 = createDataPartition(fs$Revenue, p = 0.75, list = FALSE)
train3 = fs[index3, ]
test3 = fs[-index3, ]

oversampling_onl = ovun.sample( Revenue ~ ., data = train3, method = "over", N = 10108)$data

# Train the model
model3 = train(Revenue ~ ., data = train3, method = "rpart",trControl = trainControl(method = "cv",numbe

# Make predictions on the test set
predictions = predict(model3, test3)

# Evaluate the model
confusionMatrix(predictions, test3$Revenue)
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction FALSE TRUE
```

```
##      FALSE  2474  239
##      TRUE    100  238
##
##                Accuracy : 0.8889
##                  95% CI : (0.8772, 0.8998)
##     No Information Rate : 0.8437
##     P-Value [Acc > NIR] : 4.086e-13
##
##                   Kappa : 0.5221
##
##  Mcnemar's Test P-Value : 6.623e-14
##
##             Sensitivity : 0.9611
##             Specificity : 0.4990
##          Pos Pred Value : 0.9119
##          Neg Pred Value : 0.7041
##              Prevalence : 0.8437
##          Detection Rate : 0.8109
##    Detection Prevalence : 0.8892
##       Balanced Accuracy : 0.7301
##
##        'Positive' Class : FALSE
##
```

```r
#decision tree plot
tree<- rpart(Revenue~., data = train3, method = 'class')
rpart.plot(tree)
```




```
#KNN - model4
library(class)
library(caret)

fs$Revenue=as.numeric(onl$Revenue)

str(fs)
```

```
## 'data.frame':    12205 obs. of  13 variables:
##  $ Administrative         : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ Informational          : int  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                  : chr  "Feb" "Feb" "Feb" "Feb" ...
##  $ OperatingSystems       : int  1 2 4 3 3 2 2 1 2 2 ...
##  $ Browser                : int  1 2 1 2 3 2 4 2 2 4 ...
##  $ VisitorType            : chr  "Returning_Visitor" "Returning_Visitor" "Returning_Visitor" "Returni
##  $ Revenue                : num  0 0 0 0 0 0 0 0 0 0 ...
```

```
onl_norm = fs
e = na.omit(onl_norm)
```

```
onl_norm = subset(onl_norm, select = -c(Month,VisitorType) )

str(onl_norm)
```

```
## 'data.frame':    12205 obs. of  11 variables:
##  $ Administrative         : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ Informational          : int  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 ...
##  $ OperatingSystems       : int  1 2 4 3 3 2 2 1 2 2 ...
##  $ Browser                : int  1 2 1 2 3 2 4 2 2 4 ...
##  $ Revenue                : num  0 0 0 0 0 0 0 0 0 0 ...
```

```
##Generate a random number that is 90% of the total number of rows in dataset.
 ran <- sample(1:nrow(onl_norm), 0.9 * nrow(onl_norm))

 ##the normalization function is created
 nor <-function(x) { (x -min(x))/(max(x)-min(x))   }

norm <- as.data.frame(lapply(onl_norm[,colnames(onl_norm)[colnames(onl_norm) != 'Revenue']], nor))

train_knn <- norm[ran,]
test_knn <- norm[-ran,]

target_category <- onl_norm[ran,11]
test_category <- onl_norm[-ran,11]
 ##run knn function
 model4 <- knn(train_knn,test_knn,cl=target_category,k=13)

 #source code - https://towardsdatascience.com/k-nearest-neighbors-algorithm-with-examples-in-r-simply-

confusionMatrix(table(model4,test_category))
```

```
## Confusion Matrix and Statistics
##
##       test_category
## model4    0    1
##      0 1020  127
##      1   19   55
##
##                Accuracy : 0.8804
##                  95% CI : (0.8609, 0.8981)
##     No Information Rate : 0.8509
##     P-Value [Acc > NIR] : 0.001713
##
##                   Kappa : 0.3759
##
##  Mcnemar's Test P-Value : < 2.2e-16
```

```
##
##             Sensitivity : 0.9817
##             Specificity : 0.3022
##          Pos Pred Value : 0.8893
##          Neg Pred Value : 0.7432
##              Prevalence : 0.8509
##          Detection Rate : 0.8354
##    Detection Prevalence : 0.9394
##       Balanced Accuracy : 0.6420
##
##        'Positive' Class : 0
##
```

```
#Analysis of variance
lg_mod<-glm(Revenue ~. ,data=onl,family = binomial(link = "logit"))

anova(lg_mod,test = "Chisq")
```

```
## Analysis of Deviance Table
##
## Model: binomial, link: logit
##
## Response: Revenue
##
## Terms added sequentially (first to last)
##
##
##                           Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
## NULL                                      12204    10582.5
## Administrative             1   199.93     12203    10382.6 < 2.2e-16 ***
## Administrative_Duration    1     1.64     12202    10381.0 0.2000474
## Informational              1    18.98     12201    10362.0 1.319e-05 ***
## Informational_Duration     1     0.88     12200    10361.1 0.3473824
## ProductRelated             1    88.69     12199    10272.4 < 2.2e-16 ***
## ProductRelated_Duration    1     6.98     12198    10265.4 0.0082211 **
## BounceRates                1   386.15     12197     9879.3 < 2.2e-16 ***
## ExitRates                  1   305.80     12196     9573.5 < 2.2e-16 ***
## PageValues                 1  2132.59     12195     7440.9 < 2.2e-16 ***
## SpecialDay                 1    21.80     12194     7419.1 3.022e-06 ***
## Month                      9   239.85     12185     7179.2 < 2.2e-16 ***
## OperatingSystems           1     4.14     12184     7175.1 0.0418434 *
## Browser                    1     4.63     12183     7170.5 0.0314997 *
## Region                     1     0.80     12182     7169.7 0.3712997
## TrafficType                1     0.15     12181     7169.5 0.6961835
## VisitorType                2    14.87     12179     7154.6 0.0005915 ***
## Weekend                    1     2.06     12178     7152.6 0.1511590
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
#LOGISTICS REGRESSION - model5
index5 = createDataPartition(fs$Revenue, p=0.75, list=FALSE)
train5 = fs[ index2,]
test5 =  fs[-index2,]
```

```r
model5 = glm(Revenue~., data=train2,family = binomial(link = "logit"))

pred =  predict(model5,newdata = test2,type = "response")

pred = as.numeric(pred)

pred =as.factor(round(pred,0))

test5$Revenue=as.numeric(test5$Revenue)
confusionMatrix(table(pred,test5$Revenue))
```

```
## Confusion Matrix and Statistics
##
##
## pred    0    1
##    0 2512  304
##    1   62  173
##
##                Accuracy : 0.88
##                  95% CI : (0.868, 0.8914)
##     No Information Rate : 0.8437
##     P-Value [Acc > NIR] : 6.117e-09
##
##                   Kappa : 0.4268
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9759
##             Specificity : 0.3627
##          Pos Pred Value : 0.8920
##          Neg Pred Value : 0.7362
##              Prevalence : 0.8437
##          Detection Rate : 0.8233
##    Detection Prevalence : 0.9230
##       Balanced Accuracy : 0.6693
##
##        'Positive' Class : 0
##
```