R Notebook

1. DEFINING THE QUESTION

a) Specifying the Question

Determining which individuals are most likely to click on the ads.

b) Defining the Metrics of Success

Performing the Exploratory Data Analysis.

c) Understanding the context

Determining the audience the entrepreneur can target.

d) Recording the Experimental Design

1. Defining the question, the metric for success, the context and experimental design.
2. Loading and exploring the dataset.
3. Finding and dealing with outliers, anomalies, and missing data within the dataset.
4. Perform univariate and bivariate analysis.
5. Giving a conclusion and recommendation.

e) Relevance of the data

The data used in this project is for determining which audience should be targeted by the entrepreneur. The dataset link: ('http://bit.ly/IPAdvertisingData')

2. DATA ANALYSIS

a) Checking the Data

```
library(data.table)
```

```
library(ggplot2)
```

```
library(magrittr)
```

```
library(dplyr)
```

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

## The following objects are masked from 'package:data.table':
##
##     between, first, last

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

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

```
# Reading the data
df <- fread('http://bit.ly/IPAdvertisingData')
df
```

```
##       Daily Time Spent on Site   Age Area Income Daily Internet Usage
##                          <num> <int>       <num>                <num>
##    1:                    68.95    35    61833.90               256.09
##    2:                    80.23    31    68441.85               193.77
##    3:                    69.47    26    59785.94               236.50
##    4:                    74.15    29    54806.18               245.89
##    5:                    68.37    35    73889.99               225.58
##   ---
##  996:                    72.97    30    71384.57               208.58
##  997:                    51.30    45    67782.17               134.42
##  998:                    51.63    51    42415.72               120.37
##  999:                    55.55    19    41920.79               187.95
## 1000:                    45.01    26    29875.80               178.35
##                              Ad Topic Line           City  Male
##                                     <char>         <char> <int>
##    1:    Cloned 5thgeneration orchestration    Wrightburgh     0
##    2:    Monitored national standardization      West Jodi     1
##    3:      Organic bottom-line service-desk       Davidton     0
##    4: Triple-buffered reciprocal time-frame West Terrifurt     1
##    5:         Robust logistical utilization   South Manuel     0
##   ---
##  996:         Fundamental modular algorithm      Duffystad     1
##  997:       Grass-roots cohesive monitoring    New Darlene     1
##  998:          Expanded intangible solution  South Jessica     1
##  999:  Proactive bandwidth-monitored policy    West Steven     0
## 1000:       Virtual 5thgeneration emulation    Ronniemouth     0
##                       Country           Timestamp Clicked on Ad
##                        <char>              <POSc>         <int>
##    1:                Tunisia 2016-03-27 00:53:11             0
##    2:                 Nauru 2016-04-04 01:39:02             0
##    3:            San Marino 2016-03-13 20:35:42             0
```

```
##    4:                  Italy 2016-01-10 02:31:19            0
##    5:                Iceland 2016-06-03 03:36:18            0
##   ---
##  996:                Lebanon 2016-02-11 21:49:00            1
##  997: Bosnia and Herzegovina 2016-04-22 02:07:01            1
##  998:               Mongolia 2016-02-01 17:24:57            1
##  999:              Guatemala 2016-03-24 02:35:54            0
## 1000:                 Brazil 2016-06-03 21:43:21            1
```

```
# Viewing the dataset
View(df)
```

```
# Viewing the column names
colnames(df)
```

```
##  [1] "Daily Time Spent on Site" "Age"
##  [3] "Area Income"              "Daily Internet Usage"
##  [5] "Ad Topic Line"            "City"
##  [7] "Male"                     "Country"
##  [9] "Timestamp"                "Clicked on Ad"
```

```
# Previewing the dataset
class(df)
```

```
## [1] "data.table" "data.frame"
```

```
# Previewing the top of the dataset
head(df)
```

```
##    Daily Time Spent on Site   Age Area Income Daily Internet Usage
##                       <num> <int>       <num>                <num>
## 1:                    68.95    35    61833.90               256.09
## 2:                    80.23    31    68441.85               193.77
## 3:                    69.47    26    59785.94               236.50
## 4:                    74.15    29    54806.18               245.89
## 5:                    68.37    35    73889.99               225.58
## 6:                    59.99    23    59761.56               226.74
##                            Ad Topic Line           City  Male    Country
##                                   <char>         <char> <int>     <char>
## 1:    Cloned 5thgeneration orchestration    Wrightburgh     0    Tunisia
## 2:    Monitored national standardization      West Jodi     1      Nauru
## 3:      Organic bottom-line service-desk       Davidton     0 San Marino
## 4: Triple-buffered reciprocal time-frame West Terrifurt     1      Italy
## 5:         Robust logistical utilization   South Manuel     0    Iceland
## 6:       Sharable client-driven software      Jamieberg     1     Norway
##              Timestamp Clicked on Ad
##                  <POSc>         <int>
## 1: 2016-03-27 00:53:11             0
## 2: 2016-04-04 01:39:02             0
## 3: 2016-03-13 20:35:42             0
## 4: 2016-01-10 02:31:19             0
## 5: 2016-06-03 03:36:18             0
## 6: 2016-05-19 14:30:17             0
```

```
# Previewing the bottom of the dataset
tail(df)
```

```
##    Daily Time Spent on Site   Age Area Income Daily Internet Usage
##                       <num> <int>       <num>                <num>
## 1:                    43.70    28    63126.96               173.01
## 2:                    72.97    30    71384.57               208.58
## 3:                    51.30    45    67782.17               134.42
## 4:                    51.63    51    42415.72               120.37
## 5:                    55.55    19    41920.79               187.95
## 6:                    45.01    26    29875.80               178.35
##                           Ad Topic Line          City  Male
##                                  <char>        <char> <int>
## 1:        Front-line bifurcated ability  Nicholasland     0
## 2:        Fundamental modular algorithm     Duffystad     1
## 3:      Grass-roots cohesive monitoring   New Darlene     1
## 4:         Expanded intangible solution South Jessica     1
## 5: Proactive bandwidth-monitored policy   West Steven     0
## 6:      Virtual 5thgeneration emulation   Ronniemouth     0
##                   Country           Timestamp Clicked on Ad
##                    <char>              <POSc>         <int>
## 1:                Mayotte 2016-04-04 03:57:48             1
## 2:                Lebanon 2016-02-11 21:49:00             1
## 3: Bosnia and Herzegovina 2016-04-22 02:07:01             1
## 4:               Mongolia 2016-02-01 17:24:57             1
## 5:              Guatemala 2016-03-24 02:35:54             0
## 6:                 Brazil 2016-06-03 21:43:21             1
```

```
# Checking the shape of the dataset
dim(df)
```

```
## [1] 1000   10
```

1000 rows and 10 columns

b) Data Cleaning

Missing Values

```
# Checking for missing values
sum(is.na(df))
```

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

There are no missing values.

```
# Removing all rows with na
na.omit(df)
```

```
##       Daily Time Spent on Site   Age Area Income Daily Internet Usage
##                          <num> <int>       <num>                <num>
##    1:                    68.95    35    61833.90               256.09
##    2:                    80.23    31    68441.85               193.77
##    3:                    69.47    26    59785.94               236.50
##    4:                    74.15    29    54806.18               245.89
##    5:                    68.37    35    73889.99               225.58
##   ---
##  996:                    72.97    30    71384.57               208.58
##  997:                    51.30    45    67782.17               134.42
##  998:                    51.63    51    42415.72               120.37
##  999:                    55.55    19    41920.79               187.95
## 1000:                    45.01    26    29875.80               178.35
##                               Ad Topic Line           City  Male
##                                      <char>         <char> <int>
##    1:    Cloned 5thgeneration orchestration    Wrightburgh     0
##    2:    Monitored national standardization      West Jodi     1
##    3:      Organic bottom-line service-desk       Davidton     0
##    4: Triple-buffered reciprocal time-frame West Terrifurt     1
##    5:         Robust logistical utilization   South Manuel     0
##   ---
##  996:         Fundamental modular algorithm      Duffystad     1
##  997:       Grass-roots cohesive monitoring    New Darlene     1
##  998:          Expanded intangible solution  South Jessica     1
##  999:  Proactive bandwidth-monitored policy    West Steven     0
## 1000:       Virtual 5thgeneration emulation    Ronniemouth     0
##                       Country           Timestamp Clicked on Ad
##                        <char>              <POSc>         <int>
##    1:                Tunisia 2016-03-27 00:53:11             0
##    2:                  Nauru 2016-04-04 01:39:02             0
##    3:             San Marino 2016-03-13 20:35:42             0
##    4:                  Italy 2016-01-10 02:31:19             0
##    5:                Iceland 2016-06-03 03:36:18             0
##   ---
##  996:                Lebanon 2016-02-11 21:49:00             1
##  997: Bosnia and Herzegovina 2016-04-22 02:07:01             1
##  998:               Mongolia 2016-02-01 17:24:57             1
##  999:              Guatemala 2016-03-24 02:35:54             0
## 1000:                 Brazil 2016-06-03 21:43:21             1
```

Duplicates

```
# Checking for duplicates
duplicated_rows <- df[duplicated(df),]
duplicated_rows
```

```
## Empty data.table (0 rows and 10 cols): Daily Time Spent on Site,Age,Area Income,Daily Internet Usage
```

There are no duplicates

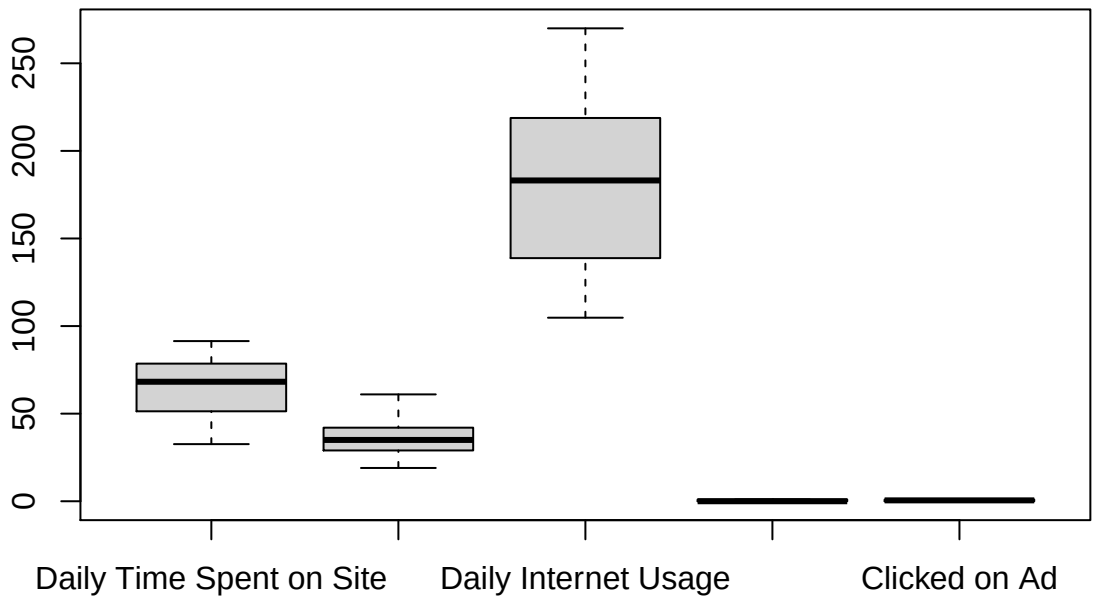
```r
# Displaying the unique items and assigning unique_items variable
unique_items <- df[!duplicated(df), ]
unique_items
```

```
##       Daily Time Spent on Site   Age Area Income Daily Internet Usage
##                          <num> <int>       <num>                <num>
##    1:                    68.95    35    61833.90               256.09
##    2:                    80.23    31    68441.85               193.77
##    3:                    69.47    26    59785.94               236.50
##    4:                    74.15    29    54806.18               245.89
##    5:                    68.37    35    73889.99               225.58
##   ---
##  996:                    72.97    30    71384.57               208.58
##  997:                    51.30    45    67782.17               134.42
##  998:                    51.63    51    42415.72               120.37
##  999:                    55.55    19    41920.79               187.95
## 1000:                    45.01    26    29875.80               178.35
##                               Ad Topic Line           City  Male
##                                      <char>         <char> <int>
##    1:    Cloned 5thgeneration orchestration    Wrightburgh     0
##    2:    Monitored national standardization      West Jodi     1
##    3:      Organic bottom-line service-desk       Davidton     0
##    4: Triple-buffered reciprocal time-frame West Terrifurt     1
##    5:         Robust logistical utilization   South Manuel     0
##   ---
##  996:         Fundamental modular algorithm      Duffystad     1
##  997:       Grass-roots cohesive monitoring    New Darlene     1
##  998:          Expanded intangible solution  South Jessica     1
##  999:  Proactive bandwidth-monitored policy    West Steven     0
## 1000:       Virtual 5thgeneration emulation    Ronniemouth     0
##                       Country           Timestamp Clicked on Ad
##                        <char>              <POSc>         <int>
##    1:                Tunisia 2016-03-27 00:53:11             0
##    2:                  Nauru 2016-04-04 01:39:02             0
##    3:             San Marino 2016-03-13 20:35:42             0
##    4:                  Italy 2016-01-10 02:31:19             0
##    5:                Iceland 2016-06-03 03:36:18             0
##   ---
##  996:                Lebanon 2016-02-11 21:49:00             1
##  997: Bosnia and Herzegovina 2016-04-22 02:07:01             1
##  998:               Mongolia 2016-02-01 17:24:57             1
##  999:              Guatemala 2016-03-24 02:35:54             0
## 1000:                 Brazil 2016-06-03 21:43:21             1
```

Outliers

```r
# Visualizing outliers using boxplot
df1 <- subset(df, select = c("Daily Time Spent on Site", "Age", "Daily Internet Usage", "Male", "Clicke

boxplot(df1)
```




```
# Renaming columns
df1 <- df1 %>% rename(Daily_Time_Spent_on_Site = "Daily Time Spent on Site")
df1 <- df1 %>% rename(Daily_Internet_Usage = "Daily Internet Usage")
df1 <- df1 %>% rename(Clicked_on_Ad = "Clicked on Ad")

df1
```

```
##       Daily_Time_Spent_on_Site   Age Daily_Internet_Usage  Male Clicked_on_Ad
##                          <num> <int>                <num> <int>         <int>
##    1:                    68.95    35               256.09     0             0
##    2:                    80.23    31               193.77     1             0
##    3:                    69.47    26               236.50     0             0
##    4:                    74.15    29               245.89     1             0
##    5:                    68.37    35               225.58     0             0
##   ---
##  996:                    72.97    30               208.58     1             1
##  997:                    51.30    45               134.42     1             1
##  998:                    51.63    51               120.37     1             1
##  999:                    55.55    19               187.95     0             0
## 1000:                    45.01    26               178.35     0             1
```

3. BIVARIATE AND UNIVARIATE ANALYSIS

a) Univariate Analysis

Measures of Central Tendency

```
# Summary statistics of the dataset
summary(df1)
```

```
##  Daily_Time_Spent_on_Site      Age        Daily_Internet_Usage      Male
##  Min.   :32.60            Min.   :19.00   Min.   :104.8        Min.   :0.000
##  1st Qu.:51.36            1st Qu.:29.00   1st Qu.:138.8        1st Qu.:0.000
##  Median :68.22            Median :35.00   Median :183.1        Median :0.000
##  Mean   :65.00            Mean   :36.01   Mean   :180.0        Mean   :0.481
##  3rd Qu.:78.55            3rd Qu.:42.00   3rd Qu.:218.8        3rd Qu.:1.000
##  Max.   :91.43            Max.   :61.00   Max.   :270.0        Max.   :1.000
##  Clicked_on_Ad
##  Min.   :0.0
##  1st Qu.:0.0
##  Median :0.5
##  Mean   :0.5
##  3rd Qu.:1.0
##  Max.   :1.0
```

```
# Median of age
df1.Age.median <- median(df$Age)
df1.Age.median
```

```
## [1] 35
```

```
# Mean of age
df1.Age.mean <- mean(df$Age)
df1.Age.mean
```

```
## [1] 36.009
```

```
# Mode of age
getmode <- function(v) {
   uniqv <- unique(v)
   uniqv[which.max(tabulate(match(v, uniqv)))]
}

df1.Age.mode <- getmode(df$Age)
df1.Age.mode
```

```
## [1] 31
```

Measures of Dispersion

```r
# Displaying the column names
colnames(df1)
```

```
## [1] "Daily_Time_Spent_on_Site" "Age"
## [3] "Daily_Internet_Usage"     "Male"
## [5] "Clicked_on_Ad"
```

```r
# Minimum code of Daily Time Spent on Site
df1.Daily_Time_Spent_on_Site.min <- min(df1$Daily_Time_Spent_on_Site)
df1.Daily_Time_Spent_on_Site.min
```

```
## [1] 32.6
```

```r
# Minimum code of Daily Internet Usage
df1.Daily_Internet_Usage.min <- min(df1$Daily_Internet_Usage)
df1.Daily_Internet_Usage.min
```

```
## [1] 104.78
```

```r
# Minimum code of Age
df1.Age.min <- min(df1$Age)
df1.Age.min
```

```
## [1] 19
```

```r
# Maximum code of age
df1.Age.max <- max(df1$Age)
df1.Age.max
```

```
## [1] 61
```

```r
# Maximum code of Daily Internet Usage
df1.Daily_Internet_Usage.max <- max(df1$Daily_Internet_Usage)
df1.Daily_Internet_Usage.max
```

```
## [1] 269.96
```

```r
# Maximum code of Daily Time Spent on Site
df1.Daily_Time_Spent_on_Site.max <- max(df1$Daily_Time_Spent_on_Site)
df1.Daily_Time_Spent_on_Site.max
```

```
## [1] 91.43
```

```r
# Range code of age
df1.Age.range <- range(df1$Age)
df1.Age.range
```

```
## [1] 19 61
```

```r
# Range code of Daily Time Spent on Site
df1.Daily_Time_Spent_on_Site.range <- range(df1$Daily_Time_Spent_on_Site)
df1.Daily_Time_Spent_on_Site.range
```

```
## [1] 32.60 91.43
```

```r
# Quantile code of Age
df1.Age.quantile <- quantile(df1$Age)
df1.Age.quantile
```

```
##   0%  25%  50%  75% 100%
##   19   29   35   42   61
```

```r
# Quantile code of Daily Time Spent on Site
df1.Daily_Time_Spent_on_Site.quantile <- quantile(df1$Daily_Time_Spent_on_Site)
df1.Daily_Time_Spent_on_Site.quantile
```

```
##      0%     25%     50%     75%    100%
## 32.6000 51.3600 68.2150 78.5475 91.4300
```

```r
# Variance code of Age
df1.Age.variance <- var(df1$Age)
df1.Age.variance
```

```
## [1] 77.18611
```

```r
# Variance code of Daily Time Spent on Site
df1.Daily_Time_Spent_on_Site.variance <- var(df1$Daily_Time_Spent_on_Site)
df1.Daily_Time_Spent_on_Site.variance
```

```
## [1] 251.3371
```

```r
# Standard deviation code of age
df1.Age.sd <- sd(df1$Age)
df1.Age.sd
```

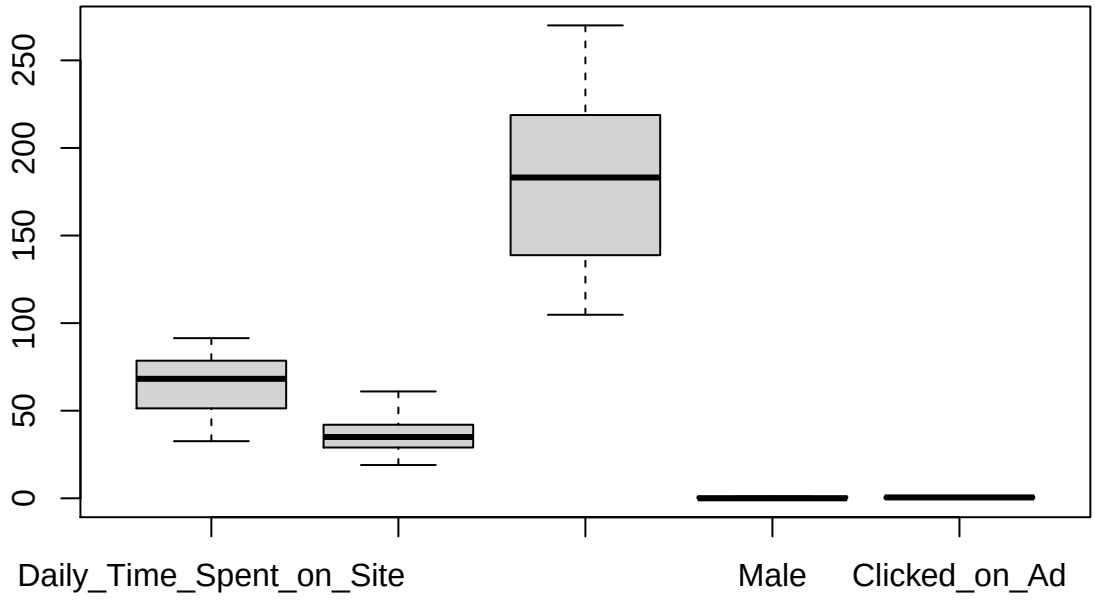
```
## [1] 8.785562
```

```r
# Standard deviation code Daily Time Spent on Site
df1.Daily_Time_Spent_on_Site.sd <- sd(df1$Daily_Time_Spent_on_Site)
df1.Daily_Time_Spent_on_Site.sd
```

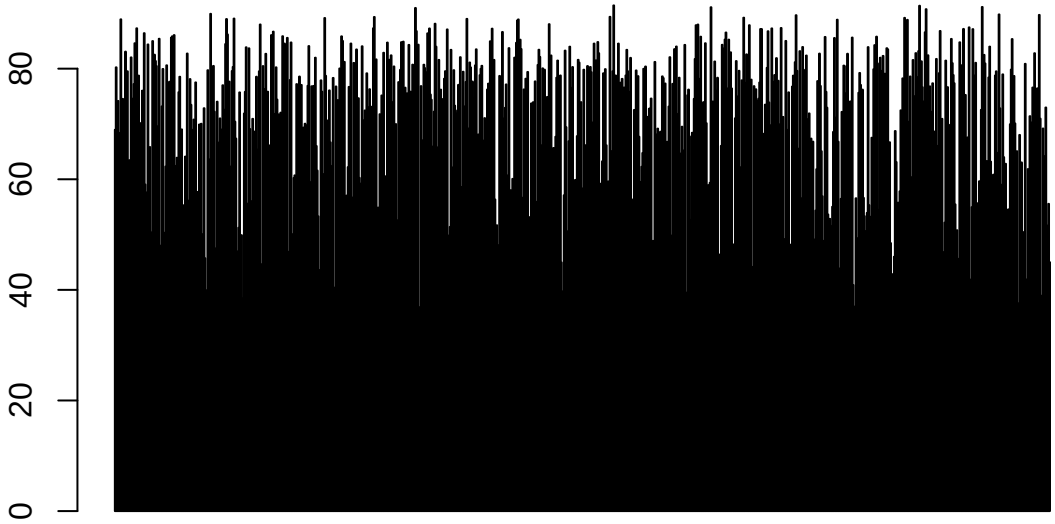
```
## [1] 15.85361
```

Univariate Graphical

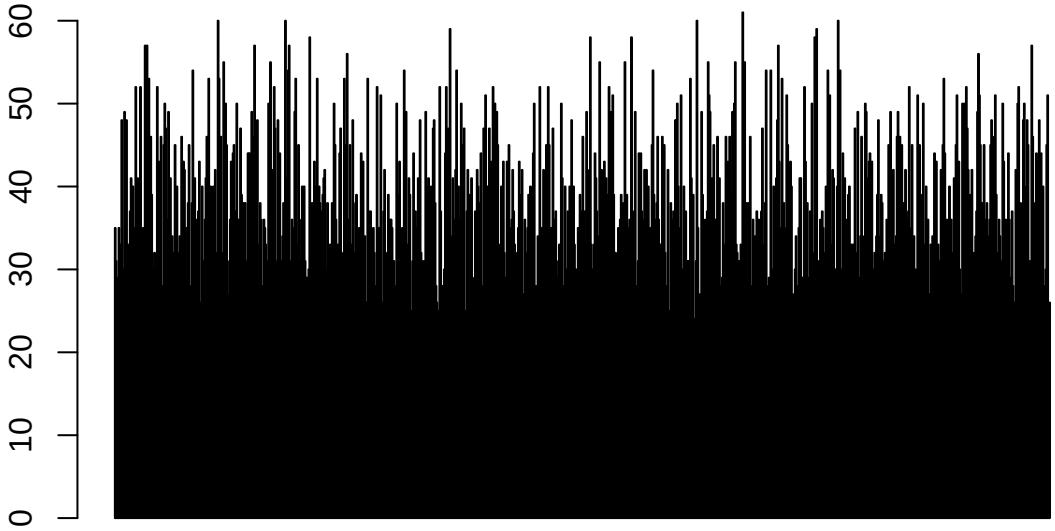
```r
# Visualizing a boxplot of numerical values
boxplot(df1)
```




```r
# Assigning the Daily Time Spent on Site column to the variable Daily Time Spent on Site
Daily_Time_Spent_on_Site <- df1$Daily_Time_Spent_on_Site
# Frequency Distribution
Daily_Time_Spent_on_Site_frequency <- table(Daily_Time_Spent_on_Site)
# Bar plot
barplot(Daily_Time_Spent_on_Site)
```



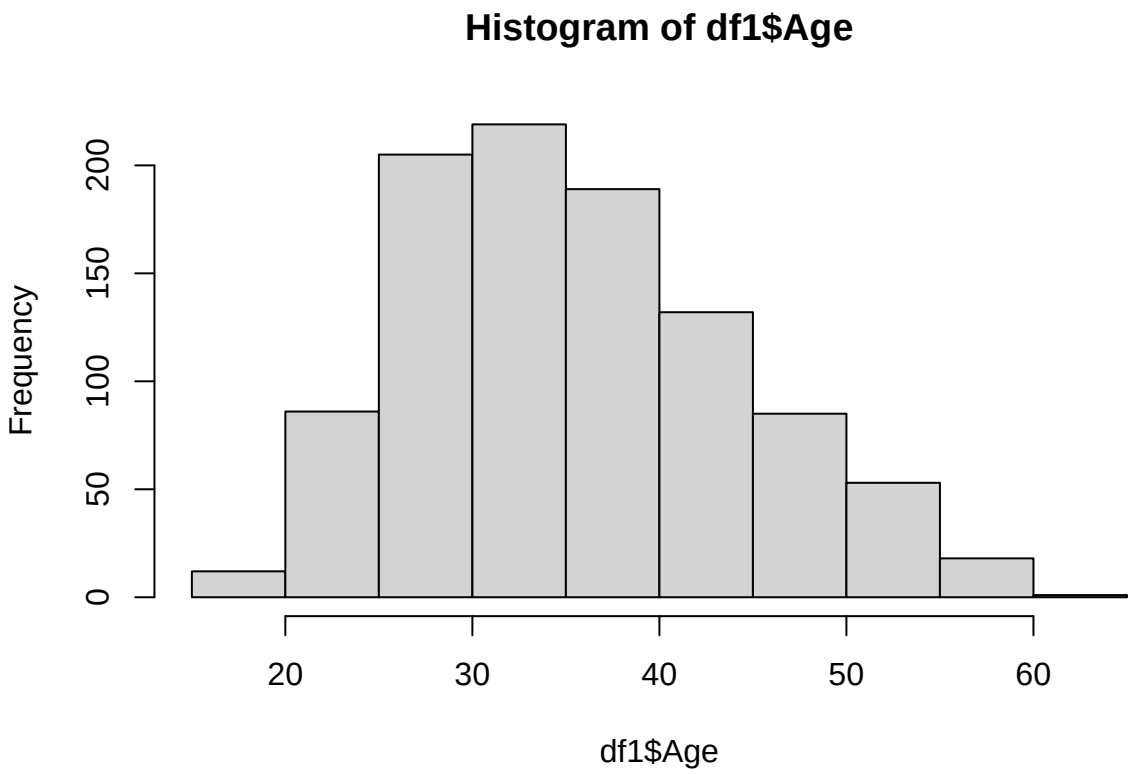

```
# Assigning the age column to the variable age
Age <- df1$Age
# Frequency Distribuution
Age_frequency <- table(Age)
# Bar plot
barplot(Age)
```



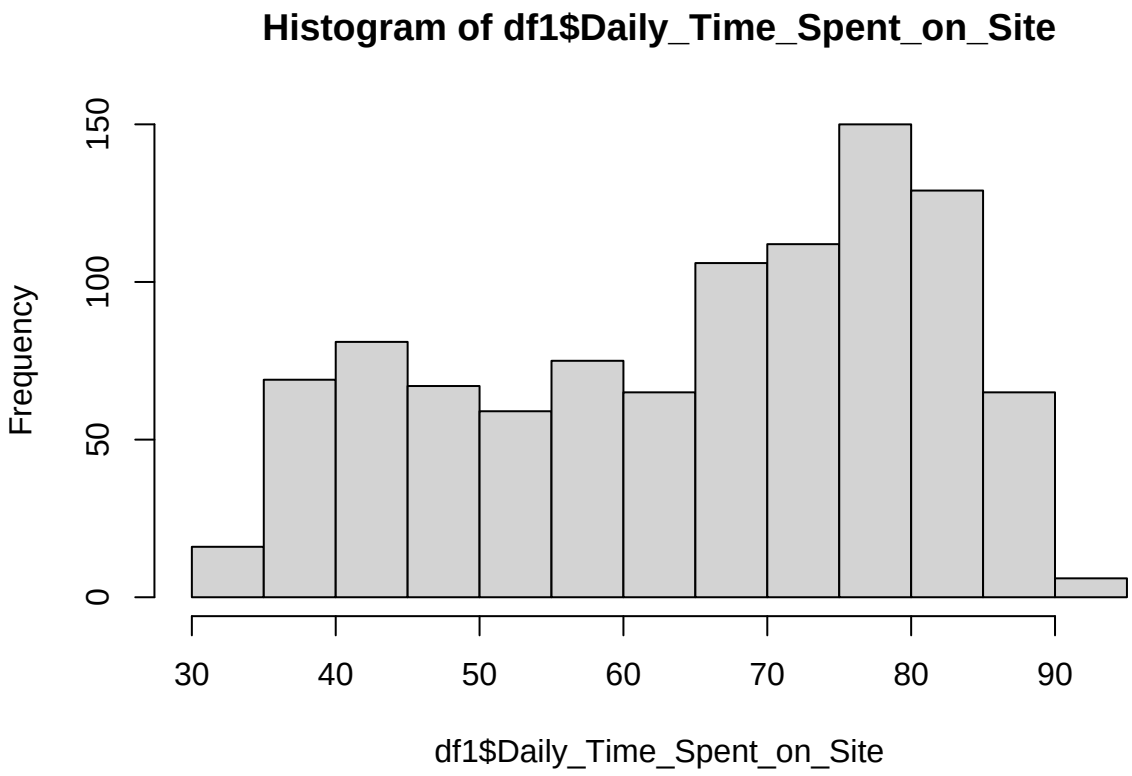

```
# Displaying the column names
colnames(df1)
```

```
## [1] "Daily_Time_Spent_on_Site" "Age"
## [3] "Daily_Internet_Usage"     "Male"
## [5] "Clicked_on_Ad"
```

```
# Histogram of age
hist(df1$Age)
```




```
# Histogram of Daily Time Spent on Site
hist(df1$Daily_Time_Spent_on_Site)
```



Bivariate analysis

```
# Assigning the age column to the variable age
Age<- df1$Age
# Covariance
cov(Daily_Time_Spent_on_Site, Age)
```

```
## [1] -46.17415
```

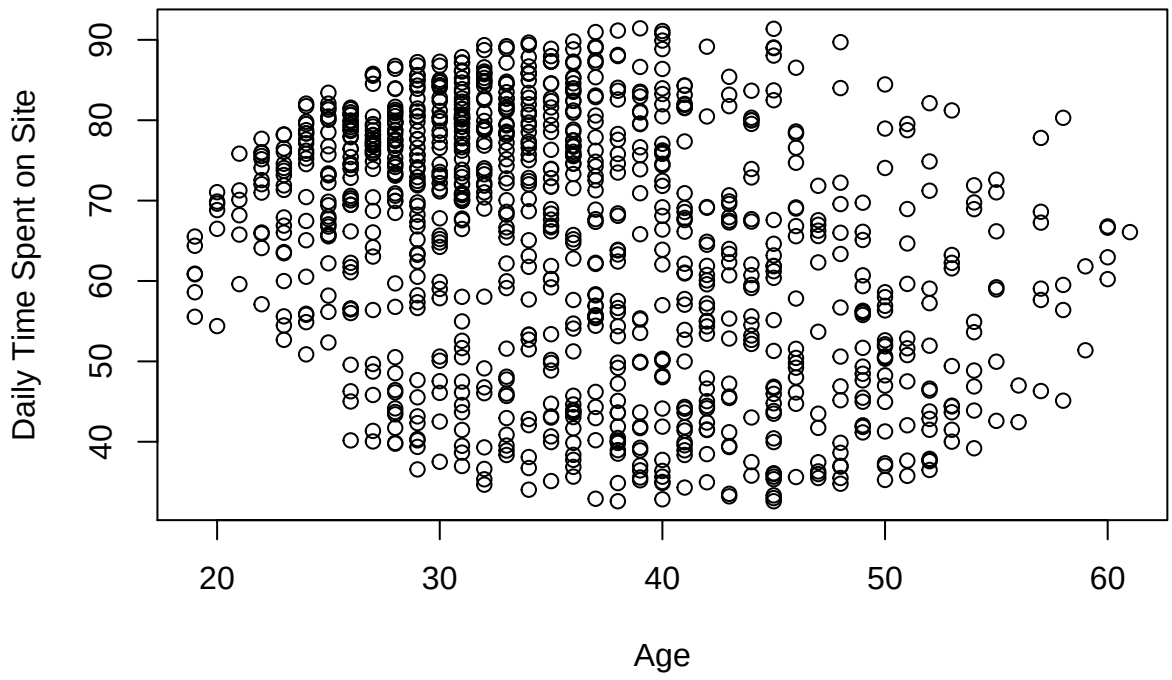
```
# Correlation
cor(Age,Daily_Time_Spent_on_Site)
```

```
## [1] -0.3315133
```

There is a negative correlation.

Graphical Techniques

```
# creating a scatterplot
plot(Age, Daily_Time_Spent_on_Site, xlab="Age", ylab="Daily Time Spent on Site")
```




```
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
# Rounding the correlation to two decimal places
res <- cor(df1)
round(res, 2)
```

```
##                           Daily_Time_Spent_on_Site   Age Daily_Internet_Usage
## Daily_Time_Spent_on_Site                     1.00 -0.33                 0.52
## Age                                         -0.33  1.00                -0.37
## Daily_Internet_Usage                         0.52 -0.37                 1.00
## Male                                        -0.02 -0.02                 0.03
## Clicked_on_Ad                               -0.75  0.49                -0.79
##                            Male Clicked_on_Ad
## Daily_Time_Spent_on_Site -0.02         -0.75
## Age                      -0.02          0.49
## Daily_Internet_Usage      0.03         -0.79
## Male                      1.00         -0.04
## Clicked_on_Ad            -0.04          1.00
```

4. IMPLEMENTING THE SOLUTION

SUPERVISED LEARNING

```
# Loading Libraries
library(ggplot2)
library(stringr)
```

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

SVM

```
library(caret)
```

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

```
str(df1)
```

```
## Classes 'data.table' and 'data.frame':   1000 obs. of  5 variables:
##  $ Daily_Time_Spent_on_Site: num  69 80.2 69.5 74.2 68.4 ...
##  $ Age                     : int  35 31 26 29 35 23 33 48 30 20 ...
##  $ Daily_Internet_Usage    : num  256 194 236 246 226 ...
##  $ Male                    : int  0 1 0 1 0 1 0 1 1 1 ...
##  $ Clicked_on_Ad           : int  0 0 0 0 0 0 0 1 0 0 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

Summary of df1

```
# Training the dataset
intrain <- createDataPartition(y = df1$Age, p= 0.8, list = FALSE)
training <- df1[intrain,]
head(training)
```

```
##    Daily_Time_Spent_on_Site   Age Daily_Internet_Usage  Male Clicked_on_Ad
##                       <num> <int>                <num> <int>         <int>
## 1:                    68.95    35               256.09     0             0
## 2:                    69.47    26               236.50     0             0
## 3:                    74.15    29               245.89     1             0
## 4:                    68.37    35               225.58     0             0
## 5:                    88.91    33               208.36     0             0
## 6:                    66.00    48               131.76     1             1
```

Training the dataset with the age column

```
testing <- df1[-intrain,]
head(testing)
```

```
##    Daily_Time_Spent_on_Site   Age Daily_Internet_Usage  Male Clicked_on_Ad
##                       <num> <int>                <num> <int>         <int>
## 1:                    80.23    31               193.77     1             0
## 2:                    59.99    23               226.74     1             0
## 3:                    42.95    33               143.56     0             1
## 4:                    74.58    40               135.51     1             1
## 5:                    77.22    30               224.44     1             0
## 6:                    84.59    35               226.54     1             0
```

Testing dataset values.

```
# Checking the dimensions of the training dataframe
dim(training);
```

```
## [1] 802   5
```

There are 802 rows and 5 columns

```
# Checking the dimensions of the testing dataframe
dim(testing);
```

```
## [1] 198   5
```

There are 198 rows and 5 columns

```
# Cleaning the data using the anyNA() method which checks for any null values.
anyNA(df1)
```

```
## [1] FALSE
```

There are no null values

```
# Displaying the summary of the data with summary() function
summary(df1)
```

```
##  Daily_Time_Spent_on_Site      Age        Daily_Internet_Usage      Male
##  Min.   :32.60            Min.   :19.00   Min.   :104.8        Min.   :0.000
##  1st Qu.:51.36            1st Qu.:29.00   1st Qu.:138.8        1st Qu.:0.000
##  Median :68.22            Median :35.00   Median :183.1        Median :0.000
##  Mean   :65.00            Mean   :36.01   Mean   :180.0        Mean   :0.481
##  3rd Qu.:78.55            3rd Qu.:42.00   3rd Qu.:218.8        3rd Qu.:1.000
##  Max.   :91.43            Max.   :61.00   Max.   :270.0        Max.   :1.000
##  Clicked_on_Ad
##  Min.   :0.0
##  1st Qu.:0.0
##  Median :0.5
##  Mean   :0.5
##  3rd Qu.:1.0
##  Max.   :1.0
```

Summary of df1 values.

```r
# Converting the categorical variables by factorizing them.

training[["Age"]] = factor(training[["Age"]])

# Before training the model you will need to control all the computational overheads.
# Implementing through the trainControl() method.
# We are using setting number =10 and repeats =3

trctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)

svm_Linear <- train(Age ~., data = training, method = "svmLinear",
trControl=trctrl,
preProcess = c("center", "scale"),
tuneLength = 10)

# Checking the result of our train() model
svm_Linear
```

```
## Support Vector Machines with Linear Kernel
##
## 802 samples
##   4 predictor
##  43 classes: '19', '20', '21', '22', '23', '24', '25', '26', '27', '28', '29', '30', '31', '32', '33
##
## Pre-processing: centered (4), scaled (4)
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 722, 723, 719, 720, 723, 722, ...
## Resampling results:
##
##   Accuracy  Kappa
##   0.065696  0.01935521
##
## Tuning parameter 'C' was held constant at a value of 1
```

There are 802 samples.

```r
# Using the predict() method for predicting results
# Passing 2 arguements, our trained model and our testing data frame.
test_pred <- predict(svm_Linear, newdata = testing)
test_pred
```

```
##   [1] 31 31 45 36 31 31 42 45 42 31 31 31 45 42 45 31 31 45 31 45 42 45 45 31 36
##  [26] 36 36 45 50 42 31 31 45 42 31 31 31 31 45 31 31 31 36 31 42 50 36 45 31 45
##  [51] 31 45 31 45 31 31 45 50 31 45 31 45 31 31 42 42 36 42 31 45 36 36 31 45 31
##  [76] 42 31 31 45 31 31 45 45 42 36 31 31 45 45 31 36 31 31 45 42 31 31 31 42 45
## [101] 36 45 31 31 31 42 50 31 42 42 31 31 31 45 31 31 31 31 45 31 50 31 50 31 31
## [126] 36 31 42 31 31 31 31 45 31 45 42 45 31 45 31 31 31 31 31 36 45 31 31 31 45
## [151] 31 42 42 45 42 31 31 31 31 42 42 42 45 45 31 31 31 31 31 45 45 50 31 45 45
## [176] 31 31 31 45 31 45 42 45 50 31 31 31 31 45 42 42 31 36 45 36 31 36 42
## 43 Levels: 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 ... 61
```

Predicted results of the testing results.

Linear Regression Model

```
# Previewing the dataset
head(df1)
```

```
##    Daily_Time_Spent_on_Site   Age Daily_Internet_Usage  Male Clicked_on_Ad
##                       <num> <int>                <num> <int>         <int>
## 1:                    68.95    35               256.09     0             0
## 2:                    80.23    31               193.77     1             0
## 3:                    69.47    26               236.50     0             0
## 4:                    74.15    29               245.89     1             0
## 5:                    68.37    35               225.58     0             0
## 6:                    59.99    23               226.74     1             0
```

```
# predicting the daily time spent on the site by the users
# Applying the lm() function.
multiple_lm <- lm(Daily_Time_Spent_on_Site ~ ., df1)

# Generating the anova table
anova(multiple_lm)
```

```
## Analysis of Variance Table
##
## Response: Daily_Time_Spent_on_Site
##                       Df Sum Sq Mean Sq  F value  Pr(>F)
## Age                    1  27595   27595 258.4518 < 2e-16 ***
## Daily_Internet_Usage   1  45724   45724 428.2494 < 2e-16 ***
## Male                   1    312     312   2.9221 0.08769 .
## Clicked_on_Ad          1  71220   71220 667.0528 < 2e-16 ***
## Residuals            995 106235     107
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
# Performing the prediction
pred2 <- predict(multiple_lm, df1)

# Printing out the result
pred2
```

```
##        1        2        3        4        5        6        7        8
## 75.13900 77.42549 75.58430 73.73106 77.18840 74.43699 78.15162 53.99553
##        9       10       11       12       13       14       15       16
## 75.46542 77.02978 56.25921 75.51383 55.24760 76.88673 53.26461 50.98072
##       17       18       19       20       21       22       23       24
## 54.60202 80.32467 53.73281 52.96977 75.26861 75.61122 53.67380 76.82366
##       25       26       27       28       29       30       31       32
## 52.44920 76.39506 51.90100 56.06510 54.99764 76.98949 72.89279 76.21437
##       33       34       35       36       37       38       39       40
## 52.34921 47.66112 54.72842 78.31685 54.90304 77.81984 54.63767 52.60100
##       41       42       43       44       45       46       47       48
## 78.15093 74.90589 77.23070 77.69350 77.61286 51.62373 75.79182 79.76648
```

```
##       49       50       51       52       53       54       55       56
## 53.47152 56.10940 76.45414 76.53075 49.55280 54.09226 56.24845 80.02771
##       57       58       59       60       61       62       63       64
## 53.08125 55.76953 75.69179 53.17868 79.24581 77.77211 80.37497 78.51972
##       65       66       67       68       69       70       71       72
## 54.58259 76.91430 56.51709 49.36195 75.40225 54.15665 53.37678 78.45062
##       73       74       75       76       77       78       79       80
## 51.68679 51.14004 52.45995 76.57908 53.52390 74.58147 54.25273 56.56481
##       81       82       83       84       85       86       87       88
## 75.99875 75.18867 55.60556 55.62178 79.58773 56.61852 78.96643 53.52524
##       89       90       91       92       93       94       95       96
## 55.67808 54.44080 54.58190 51.98360 74.66136 53.61865 50.03164 75.44122
##       97       98       99      100      101      102      103      104
## 51.33950 53.18271 53.53741 76.51466 53.47834 73.86335 79.77788 74.81918
##      105      106      107      108      109      110      111      112
## 77.31056 75.92158 73.67058 53.63612 53.53731 77.90846 53.63153 47.49203
##      113      114      115      116      117      118      119      120
## 76.85179 54.34749 75.67904 74.02864 54.09028 54.72967 50.63563 53.82557
##      121      122      123      124      125      126      127      128
## 76.31783 75.89060 77.27366 48.06826 55.33691 76.44210 55.43633 79.40511
##      129      130      131      132      133      134      135      136
## 76.58583 76.84584 52.77235 54.29437 51.32674 77.28912 55.67548 51.22600
##      137      138      139      140      141      142      143      144
## 53.76304 52.65741 78.05954 81.29863 79.17463 50.15590 54.10432 75.96325
##      145      146      147      148      149      150      151      152
## 78.77502 52.92815 54.28101 54.90364 53.93909 55.13414 78.01992 80.78072
##      153      154      155      156      157      158      159      160
## 52.21817 75.53530 77.72975 76.19898 54.66649 47.33067 76.08540 52.50156
##      161      162      163      164      165      166      167      168
## 77.70898 75.67099 75.09398 80.36554 55.12200 49.72401 55.40214 75.68239
##      169      170      171      172      173      174      175      176
## 55.22741 78.90664 53.92636 78.56884 78.77295 80.51803 55.12736 77.05540
##      177      178      179      180      181      182      183      184
## 56.87245 77.68740 52.25636 78.25709 53.58706 51.01971 54.97093 74.23894
##      185      186      187      188      189      190      191      192
## 77.62363 55.76083 53.37357 74.74670 52.28927 53.40099 54.29433 55.92326
##      193      194      195      196      197      198      199      200
## 55.51562 53.67112 73.56987 54.98352 53.33723 74.84950 76.92238 78.72792
##      201      202      203      204      205      206      207      208
## 78.42170 73.26892 50.47973 78.09183 77.06410 54.39978 77.37845 77.75932
##      209      210      211      212      213      214      215      216
## 55.86631 52.81862 75.51781 52.41694 74.24558 51.89706 77.42756 51.73372
##      217      218      219      220      221      222      223      224
## 54.87013 55.48530 50.13514 51.93935 75.43457 78.09315 54.31319 53.27205
##      225      226      227      228      229      230      231      232
## 81.52704 53.61724 54.47098 52.99663 78.58151 77.67599 75.31767 52.13613
##      233      234      235      236      237      238      239      240
## 53.72275 46.33519 49.65560 53.56158 54.90159 81.24081 54.29838 74.76954
##      241      242      243      244      245      246      247      248
## 53.83967 55.00712 77.58402 78.11936 77.75727 77.58415 52.74008 44.03532
##      249      250      251      252      253      254      255      256
## 53.82966 51.30853 77.50881 54.56042 76.93045 50.04641 56.08791 78.15227
##      257      258      259      260      261      262      263      264
## 76.50592 52.56337 77.53973 54.58317 77.38584 55.72780 51.87418 55.56059
```

```
##      265      266      267      268      269      270      271      272
## 76.20436 54.23798 53.03216 76.78537 49.21078 77.81908 52.04490 76.12375
##      273      274      275      276      277      278      279      280
## 75.06309 80.47175 75.38347 52.00580 77.77346 78.59629 79.44664 76.93445
##      281      282      283      284      285      286      287      288
## 55.05484 52.05480 54.15532 76.39511 54.37640 75.27192 53.68237 78.20270
##      289      290      291      292      293      294      295      296
## 54.28298 50.80474 51.18632 77.92928 51.26429 76.78133 79.90951 76.21644
##      297      298      299      300      301      302      303      304
## 76.31517 76.74303 77.52961 74.31139 74.87095 52.69510 53.25390 53.09067
##      305      306      307      308      309      310      311      312
## 52.75417 48.51084 77.73718 76.81963 79.52460 54.11379 75.85101 77.34768
##      313      314      315      316      317      318      319      320
## 50.46484 73.81840 77.30129 56.05360 78.42695 73.99569 79.43526 56.91880
##      321      322      323      324      325      326      327      328
## 53.91551 80.39181 78.91669 75.97730 76.73637 50.03839 48.86028 76.49450
##      329      330      331      332      333      334      335      336
## 76.53409 47.95068 77.99709 77.57394 53.85850 74.22211 78.66553 52.58556
##      337      338      339      340      341      342      343      344
## 79.97937 74.45986 76.53082 76.88404 52.91541 56.61051 77.12058 76.34737
##      345      346      347      348      349      350      351      352
## 51.87278 74.78963 77.98029 50.56713 80.54362 49.13080 75.72134 75.46874
##      353      354      355      356      357      358      359      360
## 76.48777 77.07695 56.21691 76.01425 52.98661 55.99247 54.77210 78.28127
##      361      362      363      364      365      366      367      368
## 51.26559 54.71757 77.13328 52.56873 75.95652 54.01235 77.12522 75.29151
##      369      370      371      372      373      374      375      376
## 77.80840 76.06523 56.91480 52.38138 77.57995 54.54162 76.17539 76.50923
##      377      378      379      380      381      382      383      384
## 76.09218 55.46251 54.28762 74.54389 76.77191 53.89002 80.46565 74.10585
##      385      386      387      388      389      390      391      392
## 52.00377 77.47255 77.82181 53.35467 76.01495 54.19900 75.16386 78.70250
##      393      394      395      396      397      398      399      400
## 76.69801 77.84998 54.39049 75.65949 51.41146 47.66720 77.99045 76.53679
##      401      402      403      404      405      406      407      408
## 51.63109 76.77059 53.87123 78.18384 55.35039 76.25207 53.70133 52.78632
##      409      410      411      412      413      414      415      416
## 47.08361 53.80071 50.78868 77.78552 75.56410 56.32028 77.13334 53.68046
##      417      418      419      420      421      422      423      424
## 54.69334 75.66492 77.34551 78.66487 51.37177 76.98560 53.52390 54.02699
##      425      426      427      428      429      430      431      432
## 52.79779 54.15468 51.38787 76.63758 54.23991 74.86761 76.30642 76.13112
##      433      434      435      436      437      438      439      440
## 46.74167 76.62282 76.35819 54.41330 73.38780 77.16417 54.27959 79.91750
##      441      442      443      444      445      446      447      448
## 52.87501 76.43004 52.05825 54.55702 52.05020 73.31528 47.09157 78.55338
##      449      450      451      452      453      454      455      456
## 56.51513 78.15962 51.62762 52.29192 76.13918 77.54031 49.72144 76.13044
##      457      458      459      460      461      462      463      464
## 54.54558 75.08722 53.54678 74.95960 53.03090 52.14830 78.28794 50.02640
##      465      466      467      468      469      470      471      472
## 78.95439 51.24410 52.38065 50.40574 56.07985 74.75271 51.95210 77.98708
##      473      474      475      476      477      478      479      480
## 77.35425 76.20766 53.22027 75.89131 74.98780 54.12518 52.55466 47.69003
```

```
##      481      482      483      484      485      486      487      488
## 75.65480 78.69978 77.04056 52.38062 55.28112 53.56353 80.67059 77.49937
##      489      490      491      492      493      494      495      496
## 53.62676 73.99304 55.70566 54.54221 77.33816 54.34671 53.19540 77.35096
##      497      498      499      500      501      502      503      504
## 75.70589 54.96072 79.91349 54.38573 55.77891 75.52861 76.10823 56.17853
##      505      506      507      508      509      510      511      512
## 56.81135 77.05141 76.05856 55.54916 53.67653 74.86829 51.40932 74.41618
##      513      514      515      516      517      518      519      520
## 76.18082 55.33624 76.12646 53.02282 74.82462 54.13247 55.03336 51.91249
##      521      522      523      524      525      526      527      528
## 54.63093 52.91664 76.90895 55.26638 75.07651 81.81990 54.16004 76.14380
##      529      530      531      532      533      534      535      536
## 54.20042 79.84770 55.00982 54.99970 77.43770 77.11521 77.59140 76.56503
##      537      538      539      540      541      542      543      544
## 79.01471 76.79405 76.46900 75.21826 75.95178 74.56337 79.09135 54.57717
##      545      546      547      548      549      550      551      552
## 77.54437 53.73159 77.43488 76.94191 76.24665 75.37948 77.33081 76.41859
##      553      554      555      556      557      558      559      560
## 54.96621 53.02012 51.62294 77.49270 55.22812 76.42999 78.92472 74.63721
##      561      562      563      564      565      566      567      568
## 56.63533 52.44987 74.83343 81.37117 55.29455 77.50069 56.08519 75.86644
##      569      570      571      572      573      574      575      576
## 78.69509 74.44917 55.14478 74.47131 75.57898 49.79112 56.13156 55.90726
##      577      578      579      580      581      582      583      584
## 52.38269 76.78404 75.67966 77.10042 56.59370 55.12931 51.47379 53.83422
##      585      586      587      588      589      590      591      592
## 49.58828 74.79112 75.69923 55.40275 76.05722 51.59336 52.65398 56.34514
##      593      594      595      596      597      598      599      600
## 75.67967 78.24762 52.79444 51.14134 74.51225 74.46797 78.69707 49.43930
##      601      602      603      604      605      606      607      608
## 53.69661 55.09513 52.97179 76.54685 53.00265 53.84306 77.77811 74.21738
##      609      610      611      612      613      614      615      616
## 54.26215 50.33525 51.13531 51.24744 73.93325 75.06307 75.65889 53.97608
##      617      618      619      620      621      622      623      624
## 54.82438 75.18268 53.33316 76.51595 76.99622 76.31716 53.82767 80.80289
##      625      626      627      628      629      630      631      632
## 77.13931 50.56695 77.17223 52.81130 56.23160 74.09511 74.34500 74.68292
##      633      634      635      636      637      638      639      640
## 76.84177 54.09286 54.70423 51.54581 57.01018 78.00322 53.65896 76.00007
##      641      642      643      644      645      646      647      648
## 51.88902 76.90490 76.61476 78.81454 76.37838 51.64374 55.57871 51.69546
##      649      650      651      652      653      654      655      656
## 78.14890 76.58852 80.86672 75.69920 75.48693 77.72987 74.89780 55.65933
##      657      658      659      660      661      662      663      664
## 78.15967 74.55804 79.20692 74.96766 52.22288 45.35045 54.81436 54.90843
##      665      666      667      668      669      670      671      672
## 77.70292 48.16362 74.77423 76.10356 44.38455 54.41458 77.93260 51.72857
##      673      674      675      676      677      678      679      680
## 78.00586 53.31714 77.53160 75.98941 50.14045 51.81643 81.96629 55.37521
##      681      682      683      684      685      686      687      688
## 76.47566 51.37572 53.27538 78.71247 54.16807 74.82728 74.42700 75.62595
##      689      690      691      692      693      694      695      696
## 76.95331 75.01469 78.90462 75.44129 50.72293 54.64160 77.27705 76.22585
```

```
##      697      698      699      700      701      702      703      704
## 53.31512 76.96600 78.90727 76.06862 75.98526 56.05436 47.38374 77.69147
##      705      706      707      708      709      710      711      712
## 78.07711 75.33710 50.56158 77.77619 50.97147 54.95613 55.86353 76.06258
##      713      714      715      716      717      718      719      720
## 80.19566 54.73175 74.96293 55.12394 52.28392 78.83003 79.89214 56.43181
##      721      722      723      724      725      726      727      728
## 75.71600 52.44044 51.38119 78.42774 74.35975 75.79858 74.64724 74.76684
##      729      730      731      732      733      734      735      736
## 79.22973 74.87833 76.50651 74.02194 79.82625 54.02504 53.36533 76.79004
##      737      738      739      740      741      742      743      744
## 74.70974 56.51112 48.19794 77.92322 52.79377 78.83741 77.26427 54.67120
##      745      746      747      748      749      750      751      752
## 54.54823 56.18398 45.82805 54.71752 52.18129 82.00059 53.70474 74.27717
##      753      754      755      756      757      758      759      760
## 78.89317 79.87523 78.15769 75.44798 55.78490 52.96571 49.36261 51.37373
##      761      762      763      764      765      766      767      768
## 77.42958 79.13500 54.94335 50.70601 54.97020 55.12190 48.44101 55.08098
##      769      770      771      772      773      774      775      776
## 50.64632 78.27048 76.75579 77.15807 78.45260 50.68674 51.61026 52.21888
##      777      778      779      780      781      782      783      784
## 52.04541 76.74235 51.12460 77.56926 48.55326 53.37546 76.25871 79.00266
##      785      786      787      788      789      790      791      792
## 51.63104 52.36389 78.45853 49.63065 78.61643 52.24764 49.92551 52.84018
##      793      794      795      796      797      798      799      800
## 76.22309 50.38699 56.40430 76.48978 76.70136 76.38637 78.71527 74.39468
##      801      802      803      804      805      806      807      808
## 52.72990 53.99342 54.99841 55.09714 53.74157 78.00113 55.30063 56.66019
##      809      810      811      812      813      814      815      816
## 51.10238 55.33086 51.39991 74.13073 79.05843 77.96956 76.60803 76.10159
##      817      818      819      820      821      822      823      824
## 53.63213 52.43431 77.94474 79.68115 56.29071 76.35612 44.13736 76.45084
##      825      826      827      828      829      830      831      832
## 76.44007 79.50246 75.48088 55.69427 55.82855 54.05061 51.38044 52.59962
##      833      834      835      836      837      838      839      840
## 53.60186 53.02481 76.38834 73.60682 56.40025 55.06154 53.44198 55.42760
##      841      842      843      844      845      846      847      848
## 55.95890 56.80193 77.79354 75.58302 75.97463 53.30362 53.42988 81.07897
##      849      850      851      852      853      854      855      856
## 72.97137 55.27651 73.83452 50.97341 55.28452 75.60310 45.83749 49.24704
##      857      858      859      860      861      862      863      864
## 75.35190 76.04307 52.30216 75.18540 50.28698 76.97745 80.24665 74.99327
##      865      866      867      868      869      870      871      872
## 75.73558 51.84259 77.37713 78.28198 74.69628 77.52023 49.31221 75.71864
##      873      874      875      876      877      878      879      880
## 78.48211 73.72369 81.00910 50.84978 56.77169 78.16842 76.35888 76.48235
##      881      882      883      884      885      886      887      888
## 50.98817 76.03706 76.83907 53.11084 74.43507 55.79167 53.86189 54.70953
##      889      890      891      892      893      894      895      896
## 76.61473 54.22786 75.93832 48.59075 50.75849 75.16111 77.85609 76.01755
##      897      898      899      900      901      902      903      904
## 74.73327 55.90382 56.03344 53.87729 56.45133 49.69516 55.72852 79.85440
##      905      906      907      908      909      910      911      912
## 76.58582 74.13743 50.04251 78.94961 54.71562 76.06321 50.99434 56.87448
```

```
##      913      914      915      916      917      918      919      920
## 51.48530 76.81091 53.33585 49.74484 54.62488 77.04731 76.07670 77.68608
##      921      922      923      924      925      926      927      928
## 75.75362 54.22384 54.26690 55.08845 52.85094 53.93976 77.49194 80.20575
##      929      930      931      932      933      934      935      936
## 79.15648 55.27982 76.32055 55.30058 55.58675 51.43017 73.96081 79.36476
##      937      938      939      940      941      942      943      944
## 56.81939 55.24559 50.62548 76.82289 54.33271 53.76380 53.57162 53.02212
##      945      946      947      948      949      950      951      952
## 54.82095 79.12160 77.84667 50.80546 49.09796 51.57805 54.57451 49.90198
##      953      954      955      956      957      958      959      960
## 46.11021 53.01068 75.16182 56.64743 51.60617 75.07179 74.51427 80.17888
##      961      962      963      964      965      966      967      968
## 54.07332 78.33429 76.80208 74.91066 75.56883 51.49139 51.36378 77.21118
##      969      970      971      972      973      974      975      976
## 54.97684 51.64373 52.31344 53.97470 53.64155 78.00785 53.60118 53.48637
##      977      978      979      980      981      982      983      984
## 55.46051 53.38416 80.68598 73.33610 52.40429 76.05730 53.93568 74.76551
##      985      986      987      988      989      990      991      992
## 73.21922 55.56731 74.52166 55.30668 78.35592 74.73125 51.33417 52.42970
##      993      994      995      996      997      998      999     1000
## 55.28114 74.76414 50.80275 47.09424 53.52665 55.05080 78.16833 50.25059
```

```
# Predicting the age of the individuals who click on the site
# Training the model using the training sets
linear <- lm(Age ~ ., data=df1)
summary(linear)
```

```
##
## Call:
## lm(formula = Age ~ ., data = df1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -22.2809  -5.0018  -0.4324   5.1276  20.9440
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)              24.581484   2.904269   8.464   <2e-16 ***
## Daily_Time_Spent_on_Site  0.052833   0.023369   2.261    0.024 *
## Daily_Internet_Usage      0.014085   0.009045   1.557    0.120
## Male                      0.041153   0.484964   0.085    0.932
## Clicked_on_Ad            10.876440   1.024908  10.612   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.636 on 995 degrees of freedom
## Multiple R-squared:  0.2475, Adjusted R-squared:  0.2445
## F-statistic: 81.83 on 4 and 995 DF,  p-value: < 2.2e-16
```

```
# Predicting the Output
predicted = predict(linear, df1)
predicted
```

```
##        1        2        3        4        5        6        7        8
## 31.83142 31.59072 31.58296 32.00363 31.37103 30.98579 32.21366 40.84192
##        9       10       11       12       13       14       15       16
## 31.68031 30.90376 39.69357 32.26334 40.76798 31.80025 39.74918 40.83228
##       17       18       19       20       21       22       23       24
## 40.20711 31.55678 40.05659 41.34804 31.86370 32.28265 39.97164 32.19134
##       25       26       27       28       29       30       31       32
## 40.00002 31.66722 39.95246 40.02283 40.84575 31.55322 31.95717 32.11034
##       33       34       35       36       37       38       39       40
## 41.00251 41.38969 40.42911 31.87830 40.55545 31.18504 39.84409 39.79766
##       41       42       43       44       45       46       47       48
## 31.92504 31.53004 31.77284 31.87133 31.71846 39.92419 31.14961 31.89244
##       49       50       51       52       53       54       55       56
## 39.58379 39.68368 31.25966 31.97822 39.98664 40.03432 40.43031 31.34065
##       57       58       59       60       61       62       63       64
## 41.02348 39.65572 32.05239 39.86977 31.68817 31.87528 30.77713 31.68957
##       65       66       67       68       69       70       71       72
## 39.87525 30.71640 40.31546 40.14468 31.65606 41.46220 39.85310 31.36329
##       73       74       75       76       77       78       79       80
## 40.58082 39.60689 39.58720 31.00951 39.82176 32.35975 40.06421 39.94406
##       81       82       83       84       85       86       87       88
## 31.69676 31.64124 40.08300 41.05399 30.91462 39.99061 31.50136 39.95388
##       89       90       91       92       93       94       95       96
## 40.04266 40.04412 41.09964 40.51362 31.53003 39.91758 40.14854 31.79101
##       97       98       99      100      101      102      103      104
## 40.08401 39.76960 39.56788 31.81510 40.03291 31.29680 32.06747 31.29777
##      105      106      107      108      109      110      111      112
## 30.94183 32.10238 31.83374 39.82425 39.87312 31.45769 41.47099 42.16126
##      113      114      115      116      117      118      119      120
## 31.31687 39.67776 31.38392 32.13184 41.58879 40.34860 42.83785 42.30632
##      121      122      123      124      125      126      127      128
## 32.13787 30.46789 31.68117 40.62495 40.54137 32.03811 41.49074 32.20162
##      129      130      131      132      133      134      135      136
## 31.44289 31.43007 40.41015 39.52648 40.65438 31.61547 39.55615 40.38898
##      137      138      139      140      141      142      143      144
## 39.46354 39.46783 30.85487 30.68649 31.69177 39.77675 40.18851 32.34278
##      145      146      147      148      149      150      151      152
## 31.19754 39.47729 40.27613 40.76944 40.23229 41.19966 30.17941 31.15505
##      153      154      155      156      157      158      159      160
## 39.68247 31.52019 31.40827 32.39936 39.60536 40.55156 31.80275 41.69239
##      161      162      163      164      165      166      167      168
## 31.40584 32.12864 31.92139 30.87581 39.61630 40.03336 40.97451 32.26425
##      169      170      171      172      173      174      175      176
## 40.12324 31.95834 39.76309 31.23225 31.58422 30.80938 40.74115 31.57697
##      177      178      179      180      181      182      183      184
## 40.15461 31.19659 39.72958 32.05069 39.61921 42.29500 40.99571 31.79343
##      185      186      187      188      189      190      191      192
## 32.06625 39.85934 40.11462 32.03199 42.10020 39.61819 39.84910 40.14520
##      193      194      195      196      197      198      199      200
## 40.26344 39.83161 32.27971 40.89585 39.79800 32.14461 31.33590 31.34452
##      201      202      203      204      205      206      207      208
## 31.10976 31.84694 39.81341 30.91263 31.34989 40.23676 31.63985 31.70384
##      209      210      211      212      213      214      215      216
## 40.58564 39.52014 31.74921 39.97235 31.99888 41.24816 32.14135 41.08568
```

```
##      217      218      219      220      221      222      223      224
## 40.52589 39.85062 40.12051 40.16284 32.28180 31.41317 40.02084 40.86036
##      225      226      227      228      229      230      231      232
## 31.70110 41.50315 40.17587 39.47938 31.11981 31.43166 31.35688 39.89130
##      233      234      235      236      237      238      239      240
## 40.78319 42.96951 40.27730 39.52060 41.25357 30.27614 41.18977 31.80023
##      241      242      243      244      245      246      247      248
## 41.81076 39.95861 31.87416 31.95458 30.86193 32.05601 39.66857 42.31567
##      249      250      251      252      253      254      255      256
## 40.07029 41.01411 31.76422 40.41007 31.96281 40.25248 40.18787 31.69579
##      257      258      259      260      261      262      263      264
## 32.11581 39.56891 32.17627 41.06412 31.51943 40.18724 40.66432 39.49455
##      265      266      267      268      269      270      271      272
## 32.46612 40.93586 41.11570 31.65029 40.03433 31.62576 40.21594 31.60553
##      273      274      275      276      277      278      279      280
## 32.19403 30.75005 31.85742 40.18085 32.24847 31.84680 31.12571 31.70892
##      281      282      283      284      285      286      287      288
## 39.93987 40.45061 39.48252 31.71648 39.84603 31.59136 40.20958 31.96498
##      289      290      291      292      293      294      295      296
## 39.82914 41.01124 40.54436 32.01811 40.66149 32.11533 30.71267 32.06032
##      297      298      299      300      301      302      303      304
## 31.84895 31.57877 31.85339 31.35965 31.79953 40.72433 39.67765 41.43957
##      305      306      307      308      309      310      311      312
## 39.56085 42.23482 32.08290 32.20545 31.69256 40.62949 31.78784 31.69399
##      313      314      315      316      317      318      319      320
## 41.83549 31.96290 32.31016 40.92070 31.22200 31.82632 31.47123 40.21568
##      321      322      323      324      325      326      327      328
## 40.08470 31.93388 31.60380 31.50929 32.43073 40.16533 40.26931 31.90318
##      329      330      331      332      333      334      335      336
## 31.23565 40.74472 31.67085 31.82825 40.37846 32.27734 31.83817 39.56594
##      337      338      339      340      341      342      343      344
## 31.78379 31.89234 31.65182 31.04066 39.75558 40.89877 32.48955 31.95354
##      345      346      347      348      349      350      351      352
## 40.58139 31.17664 31.70005 40.15579 31.12171 41.28091 31.86001 31.49430
##      353      354      355      356      357      358      359      360
## 31.93534 31.89105 41.20192 32.35050 39.80088 39.66289 40.40584 31.91565
##      361      362      363      364      365      366      367      368
## 39.67887 40.34459 31.56420 41.28338 31.53438 41.27181 31.20208 32.32986
##      369      370      371      372      373      374      375      376
## 31.59151 31.85147 40.15542 39.53879 31.69122 39.70092 31.35921 31.50192
##      377      378      379      380      381      382      383      384
## 32.45536 40.85771 40.79431 32.40374 31.98301 39.54019 31.05292 31.88955
##      385      386      387      388      389      390      391      392
## 40.24690 31.99763 31.72042 39.71658 31.42820 40.76046 32.09243 31.84895
##      393      394      395      396      397      398      399      400
## 31.69721 30.72694 40.79112 31.25291 40.17929 41.22145 31.44401 31.54061
##      401      402      403      404      405      406      407      408
## 39.73141 32.31432 39.63410 32.03701 40.17614 32.38700 40.43382 40.03315
##      409      410      411      412      413      414      415      416
## 41.83107 39.47812 40.53993 31.66882 31.15510 39.99219 32.15239 39.49526
##      417      418      419      420      421      422      423      424
## 40.91284 31.53313 31.30225 30.54242 39.57532 32.15080 40.01037 40.34546
##      425      426      427      428      429      430      431      432
## 39.66569 40.37354 40.55433 32.16555 40.39233 32.36273 32.29422 32.40081
```

```
##      433      434      435      436      437      438      439      440
## 42.72678 32.12462 32.15221 40.41520 31.09673 31.62963 40.09726 31.07364
##      441      442      443      444      445      446      447      448
## 39.97385 32.11280 39.50386 39.84191 39.61169 32.12895 40.53162 31.56111
##      449      450      451      452      453      454      455      456
## 39.88367 31.12003 40.01533 40.45583 32.22060 31.61348 42.67097 31.43608
##      457      458      459      460      461      462      463      464
## 39.52431 31.94983 40.75514 31.98472 39.93252 39.70046 31.04608 40.19627
##      465      466      467      468      469      470      471      472
## 31.96672 40.04211 42.05830 40.38384 40.63896 31.25918 40.13228 31.20820
##      473      474      475      476      477      478      479      480
## 31.64145 32.14656 39.95724 31.68461 31.88224 39.74370 39.60518 40.37213
##      481      482      483      484      485      486      487      488
## 30.55798 31.66615 31.25466 40.00182 39.57918 40.71530 31.29040 31.63710
##      489      490      491      492      493      494      495      496
## 39.58337 32.31366 39.76181 40.33558 30.51361 40.30580 40.73968 32.03539
##      497      498      499      500      501      502      503      504
## 31.95671 39.44918 30.93160 40.40885 39.94053 32.36405 31.29875 40.02552
##      505      506      507      508      509      510      511      512
## 40.55413 31.90497 31.98540 39.80434 42.14321 32.12156 40.86511 31.59518
##      513      514      515      516      517      518      519      520
## 32.11947 40.11668 32.40964 39.60444 32.39837 41.69233 40.40808 39.47863
##      521      522      523      524      525      526      527      528
## 39.83860 40.53799 31.66279 40.85256 32.08667 30.40373 39.72010 30.56723
##      529      530      531      532      533      534      535      536
## 40.13172 31.62654 40.60196 39.84607 32.23291 32.40979 31.45983 31.71378
##      537      538      539      540      541      542      543      544
## 31.17366 30.91983 32.37566 31.68119 31.85535 32.11119 31.44813 40.40088
##      545      546      547      548      549      550      551      552
## 31.60897 40.07902 31.72682 32.29793 32.11755 31.76594 31.75400 31.89274
##      553      554      555      556      557      558      559      560
## 40.60954 39.65901 39.57435 31.56757 40.08598 31.98508 30.92080 32.09140
##      561      562      563      564      565      566      567      568
## 40.57257 39.56322 31.30379 30.21169 40.53612 31.45980 39.60181 32.18134
##      569      570      571      572      573      574      575      576
## 30.71709 31.86369 39.88629 31.85610 31.94257 41.67028 39.58742 39.93325
##      577      578      579      580      581      582      583      584
## 39.46806 32.15448 31.08975 31.43660 40.69900 40.62202 39.84926 40.98490
##      585      586      587      588      589      590      591      592
## 40.77968 32.36128 32.26825 39.59073 32.01970 39.83496 40.97343 40.03864
##      593      594      595      596      597      598      599      600
## 31.46105 31.64960 40.03716 40.03311 31.96303 32.04783 31.87590 40.62735
##      601      602      603      604      605      606      607      608
## 41.85559 39.91912 41.20233 31.53418 39.95061 40.86448 31.12452 31.45014
##      609      610      611      612      613      614      615      616
## 40.41558 42.21065 39.74638 39.67328 32.05675 31.87372 31.57840 39.85035
##      617      618      619      620      621      622      623      624
## 39.82345 31.85458 39.46555 31.61489 31.61905 32.22572 40.81568 31.37257
##      625      626      627      628      629      630      631      632
## 31.46076 41.15467 31.85539 39.79817 39.80823 31.85534 31.30852 32.49759
##      633      634      635      636      637      638      639      640
## 31.45557 39.51082 40.46283 40.09485 40.15529 32.22643 40.78451 31.67742
##      641      642      643      644      645      646      647      648
## 41.15041 31.97109 31.91841 31.16523 32.29128 39.88013 39.49911 39.88461
```

```
##      649      650      651      652      653      654      655      656
## 30.88425 32.21727 31.46838 32.04717 32.28326 32.44006 31.81911 39.97238
##      657      658      659      660      661      662      663      664
## 31.70146 32.20464 31.79920 31.80340 40.24410 41.16662 39.77353 41.33428
##      665      666      667      668      669      670      671      672
## 31.90940 41.32027 31.82979 32.11122 41.80238 40.52401 31.53539 41.58521
##      673      674      675      676      677      678      679      680
## 32.26008 40.27873 31.63627 32.15801 39.85744 40.10435 31.37803 40.76537
##      681      682      683      684      685      686      687      688
## 31.57119 39.83120 39.56448 31.35642 39.53300 31.84520 32.02631 31.53951
##      689      690      691      692      693      694      695      696
## 31.76870 31.89117 31.96331 32.49463 41.52161 40.55218 31.02948 31.90423
##      697      698      699      700      701      702      703      704
## 39.96591 31.83719 31.84020 31.77524 30.50655 41.00713 42.98700 31.61384
##      705      706      707      708      709      710      711      712
## 31.69027 31.85379 41.61254 32.08431 40.46641 41.72252 39.53860 31.33392
##      713      714      715      716      717      718      719      720
## 31.46714 39.91299 31.43088 39.79039 39.56138 31.78422 30.88322 40.63025
##      721      722      723      724      725      726      727      728
## 32.04729 39.71315 39.88706 30.14799 31.92855 32.04807 31.35717 32.17760
##      729      730      731      732      733      734      735      736
## 31.77145 32.03071 30.76338 31.41555 31.72099 39.53724 40.30565 32.15388
##      737      738      739      740      741      742      743      744
## 32.07663 40.94792 40.96455 31.85518 39.70921 31.01692 31.14387 40.27212
##      745      746      747      748      749      750      751      752
## 40.68800 40.27644 42.48750 39.75074 40.59757 29.76322 40.90518 32.11988
##      753      754      755      756      757      758      759      760
## 30.98991 31.48463 31.63714 31.26530 40.04043 39.88862 40.14867 42.10953
##      761      762      763      764      765      766      767      768
## 31.81495 31.12315 40.11133 40.34401 40.00311 39.81567 40.37019 39.91356
##      769      770      771      772      773      774      775      776
## 41.53854 31.61294 31.50667 30.56599 32.28937 41.82090 39.79268 40.16752
##      777      778      779      780      781      782      783      784
## 40.57662 31.56444 40.93459 31.93467 41.93888 41.23308 32.08302 31.34940
##      785      786      787      788      789      790      791      792
## 39.52846 39.45636 30.86609 41.40988 31.86124 39.57702 39.81734 39.72803
##      793      794      795      796      797      798      799      800
## 30.49089 39.72184 39.75349 31.50386 31.60807 31.97751 30.91657 31.87136
##      801      802      803      804      805      806      807      808
## 40.22740 39.88899 40.13620 39.63082 40.11496 31.60253 40.17862 39.80287
##      809      810      811      812      813      814      815      816
## 39.97820 40.81632 39.79411 32.24337 31.48098 31.37220 32.39226 32.06918
##      817      818      819      820      821      822      823      824
## 39.65727 40.32908 31.91120 31.52276 39.98529 32.20038 42.92492 31.75516
##      825      826      827      828      829      830      831      832
## 31.39124 31.71547 32.17020 39.51164 40.50636 39.92397 39.78413 39.62652
##      833      834      835      836      837      838      839      840
## 39.50625 39.68403 31.39502 31.54783 39.68609 40.09819 39.64027 40.06888
##      841      842      843      844      845      846      847      848
## 40.88529 40.25096 31.52218 31.82215 32.48952 39.71144 40.19534 31.65592
##      849      850      851      852      853      854      855      856
## 32.02513 40.28726 32.29712 39.57994 40.21438 31.94171 43.02176 40.82752
##      857      858      859      860      861      862      863      864
## 32.14104 31.66803 39.76006 32.45919 42.89909 31.28722 31.21108 32.07335
```

```
##      865      866      867      868      869      870      871      872
## 32.00451 39.59063 31.95664 32.42553 31.37985 32.06350 40.33185 31.96181
##      873      874      875      876      877      878      879      880
## 31.14794 31.96248 31.02401 40.03653 40.52813 30.88557 32.41415 31.13176
##      881      882      883      884      885      886      887      888
## 40.20355 32.10244 32.02076 39.88896 31.88820 39.66820 39.83018 39.88039
##      889      890      891      892      893      894      895      896
## 31.82519 39.92002 31.98948 40.79392 40.14143 31.67202 31.87341 31.82469
##      897      898      899      900      901      902      903      904
## 32.10211 41.17600 39.91065 39.47897 39.85974 40.46807 39.57642 31.57649
##      905      906      907      908      909      910      911      912
## 31.96309 31.97242 40.45584 31.91436 40.52729 31.74681 40.32878 40.57641
##      913      914      915      916      917      918      919      920
## 39.60667 32.22328 39.52231 39.76075 40.00497 32.03510 31.81776 30.63932
##      921      922      923      924      925      926      927      928
## 32.32153 40.02727 39.72117 40.04724 40.80677 39.48278 31.26453 32.00269
##      929      930      931      932      933      934      935      936
## 31.47908 41.65902 32.23740 40.33632 40.43881 39.54947 32.26166 31.75967
##      937      938      939      940      941      942      943      944
## 40.31598 40.04222 39.85719 30.44872 39.60637 41.90156 41.46202 39.58691
##      945      946      947      948      949      950      951      952
## 40.51584 32.11814 31.17177 39.72105 41.13449 42.35018 40.78895 39.97202
##      953      954      955      956      957      958      959      960
## 42.08241 39.85760 30.83149 39.87581 39.77265 31.75636 32.13948 31.40556
##      961      962      963      964      965      966      967      968
## 40.20886 31.49236 31.25893 31.31201 31.31317 40.05904 39.97341 30.86834
##      969      970      971      972      973      974      975      976
## 39.60409 41.02877 39.72115 39.74578 40.41280 31.71762 39.48801 39.56817
##      977      978      979      980      981      982      983      984
## 40.25409 40.04254 30.69627 31.92308 41.42472 32.26510 40.31323 32.30404
##      985      986      987      988      989      990      991      992
## 31.74577 41.07073 31.93855 40.32661 32.24130 31.94198 39.72278 39.53883
##      993      994      995      996      997      998      999     1000
## 40.85359 31.22075 40.20362 42.29220 40.10275 39.92228 30.16368 40.34805
```

Prediction results.

Decision Tree

```
# Loading  the party package
library(party)
```

```
## Loading required package: grid
```

```
## Loading required package: mvtnorm
```

```
## Loading required package: modeltools
```

```
## Loading required package: stats4
```

```
## Loading required package: strucchange

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:data.table':
##
##     yearmon, yearqtr

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: sandwich

##
## Attaching package: 'strucchange'

## The following object is masked from 'package:stringr':
##
##     boundary
```

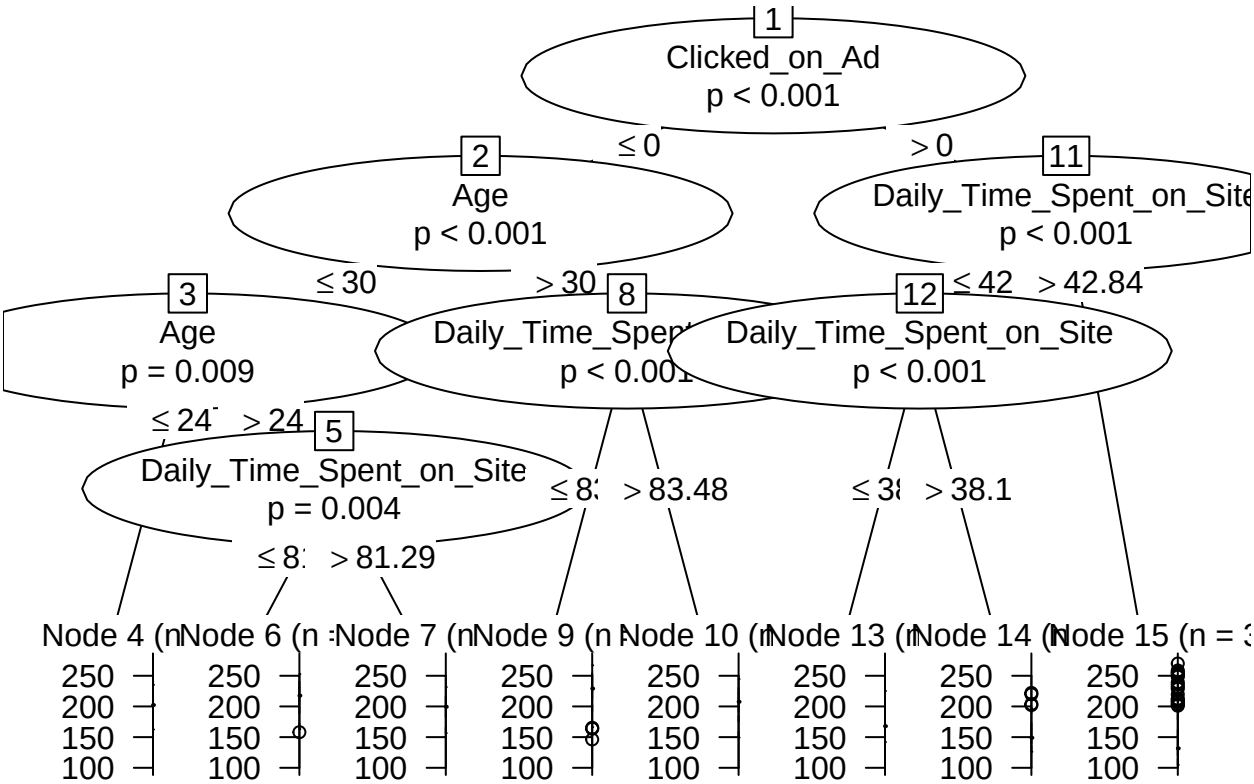
```
# Displaying the top of dataset df1.
head(df1)
```

```
##    Daily_Time_Spent_on_Site   Age Daily_Internet_Usage  Male Clicked_on_Ad
##                       <num> <int>                <num> <int>         <int>
## 1:                    68.95    35               256.09     0             0
## 2:                    80.23    31               193.77     1             0
## 3:                    69.47    26               236.50     0             0
## 4:                    74.15    29               245.89     1             0
## 5:                    68.37    35               225.58     0             0
## 6:                    59.99    23               226.74     1             0
```

```
# Creating the input data frame.
input.dat <- df1

# Creating the tree.
  output.tree <- ctree( Daily_Internet_Usage ~ Age + Clicked_on_Ad +
Daily_Time_Spent_on_Site,
  data = input.dat)

# Plotting the tree.
plot(output.tree)
```



Naives Bayes

```
library(ggplot2)
library(caret)#confusionMatrix
library(caretEnsemble)
```

```
##
## Attaching package: 'caretEnsemble'
```

```
## The following object is masked from 'package:ggplot2':
##
##     autoplot
```

```
library(psych)
```

```
##
## Attaching package: 'psych'
```

```
## The following objects are masked from 'package:ggplot2':
##
##     %+%, alpha
```

```
library(Amelia)#missmap
```

```
## Loading required package: Rcpp
```

```
## ##
## ## Amelia II: Multiple Imputation
## ## (Version 1.8.0, built: 2021-05-26)
## ## Copyright (C) 2005-2022 James Honaker, Gary King and Matthew Blackwell
## ## Refer to http://gking.harvard.edu/amelia/ for more information
## ##
```

```
library(mice) #mice
```

```
##
## Attaching package: 'mice'
```

```
## The following object is masked from 'package:stats':
##
##     filter
```

```
## The following objects are masked from 'package:base':
##
##     cbind, rbind
```

```
library(GGally) #ggpairs
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(rpart)
```

```
# Describing the data
describe(df1)
```

```
##                           vars    n   mean    sd median trimmed   mad    min
## Daily_Time_Spent_on_Site    1 1000  65.00 15.85  68.22   65.74 17.92  32.60
## Age                         2 1000  36.01  8.79  35.00   35.51  8.90  19.00
## Daily_Internet_Usage        3 1000 180.00 43.90 183.13  179.99 58.61 104.78
## Male                        4 1000   0.48  0.50   0.00    0.48  0.00   0.00
## Clicked_on_Ad               5 1000   0.50  0.50   0.50    0.50  0.74   0.00
##                             max  range  skew kurtosis   se
## Daily_Time_Spent_on_Site  91.43  58.83 -0.37    -1.10 0.50
## Age                       61.00  42.00  0.48    -0.41 0.28
## Daily_Internet_Usage     269.96 165.18 -0.03    -1.28 1.39
## Male                       1.00   1.00  0.08    -2.00 0.02
## Clicked_on_Ad              1.00   1.00  0.00    -2.00 0.02
```

```
# Displaying the top of the dataset
head(df1)
```

```
##    Daily_Time_Spent_on_Site   Age Daily_Internet_Usage  Male Clicked_on_Ad
##                       <num> <int>                <num> <int>         <int>
## 1:                    68.95    35               256.09     0             0
## 2:                    80.23    31               193.77     1             0
## 3:                    69.47    26               236.50     0             0
## 4:                    74.15    29               245.89     1             0
## 5:                    68.37    35               225.58     0             0
## 6:                    59.99    23               226.74     1             0
```

```
# Converting the output variable into a categorical variable
df1$Clicked_on_Ad <- factor(df1$Clicked_on_Ad)
df1$Clicked_on_Ad
```

```
##    [1] 0 0 0 0 0 0 0 1 0 0 1 0 1 0 1 1 1 0 1 1 0 0 1 0 1 0 1 1 1 0 0 0 1 1 1 0 1
##   [38] 0 1 1 0 0 0 0 0 1 0 0 1 1 0 0 1 1 1 0 1 1 0 1 0 0 0 0 1 0 1 1 0 1 1 0 1 1
##   [75] 1 0 1 0 1 1 0 0 1 1 0 1 0 1 1 1 1 1 0 1 1 0 1 1 1 0 1 0 0 0 0 0 0 1 1 0 1
##  [112] 1 0 1 0 0 1 1 1 1 0 0 0 1 1 0 1 0 0 0 1 1 1 0 1 1 1 1 0 0 0 1 1 0 0 1 1 1
##  [149] 1 1 0 0 1 0 0 0 1 1 0 1 0 0 0 0 1 1 1 0 1 0 1 0 0 0 1 0 1 0 1 0 1 1 1 0 0
##  [186] 1 1 0 1 1 1 1 1 1 0 1 1 0 0 0 0 0 1 0 0 1 0 0 1 1 0 1 0 1 0 1 1 1 1 1 0 0
##  [223] 1 1 0 1 1 1 0 0 0 1 1 1 1 1 1 0 1 0 1 1 0 0 0 0 1 1 1 1 0 1 0 1 1 0 0 1 0
##  [260] 1 0 1 1 1 0 1 1 0 1 0 1 0 0 0 0 1 0 0 0 0 1 1 1 0 1 0 1 0 1 1 1 0 1 0 0 0
##  [297] 0 0 0 0 0 1 1 1 1 1 0 0 0 1 0 0 1 0 0 1 0 0 0 1 1 0 0 0 0 1 1 0 0 1 0 0 1
##  [334] 0 0 1 0 0 0 0 1 1 0 0 1 0 0 1 0 1 0 0 0 0 1 0 1 1 1 0 1 1 0 1 0 1 0 0 0 0
##  [371] 1 1 0 1 0 0 0 1 1 0 0 1 0 0 1 0 0 1 0 1 0 0 0 0 1 0 1 1 0 0 1 0 1 0 1 0 1
##  [408] 1 1 1 1 0 0 1 0 1 1 0 0 0 1 0 1 1 1 1 1 0 1 0 0 0 1 0 0 1 0 0 1 0 1 0 1 1
##  [445] 1 0 1 0 1 0 1 1 0 0 1 0 1 0 1 0 1 1 0 1 0 1 1 1 1 1 0 1 0 0 0 1 0 0 1 1 1 0
##  [482] 0 0 1 1 1 0 0 1 0 1 1 0 1 1 0 0 1 0 1 1 0 0 1 1 0 0 1 1 0 1 0 0 1 0 1 0 1
##  [519] 1 1 1 1 0 1 0 0 1 0 1 0 1 1 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 1 1 1
##  [556] 0 1 0 0 0 1 1 0 0 1 0 1 0 0 0 1 0 0 1 1 1 1 0 0 0 1 1 1 1 1 0 0 1 0 1 1 1
##  [593] 0 0 1 1 0 0 0 1 1 1 1 0 1 1 0 0 1 1 1 1 0 0 0 1 1 0 1 0 0 0 1 0 0 1 0 1 1
##  [630] 0 0 0 0 1 1 1 1 0 1 0 1 0 0 0 0 1 1 1 0 0 0 0 0 0 0 1 0 0 0 0 1 1 1 1 0 1
##  [667] 0 0 1 1 0 1 0 1 0 0 1 1 0 1 0 1 1 0 1 0 0 0 0 0 0 0 1 1 0 0 1 0 0 0 0 1 1
##  [704] 0 0 0 1 0 1 1 1 0 0 1 0 1 1 0 0 1 0 1 1 0 0 0 0 0 0 0 0 0 0 1 1 0 0 1 1 0
##  [741] 1 0 0 1 1 1 1 1 1 1 0 1 0 0 0 0 0 1 1 1 1 0 0 1 1 1 1 1 1 1 0 0 0 0 1 1 1 1
##  [778] 0 1 0 1 1 0 0 1 1 0 1 0 1 1 1 0 1 1 0 0 0 0 0 1 1 1 1 1 0 1 1 1 1 1 0 0 0
##  [815] 0 0 1 1 0 0 1 0 1 0 0 0 0 1 1 1 1 1 1 1 1 0 0 1 1 1 1 1 1 1 0 0 0 1 1 0 0 1 0
##  [852] 1 1 0 1 1 0 0 1 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 0 1 1 0 0 0 1 0 0 1 0 1 1 1
##  [889] 0 1 0 1 1 0 0 0 0 1 1 1 1 1 1 0 0 0 1 0 1 0 1 1 1 0 1 1 1 0 0 0 0 1 1 1 1
##  [926] 1 0 0 0 1 0 1 1 1 0 0 1 1 1 0 1 1 1 1 1 0 0 1 1 1 1 1 1 1 0 1 1 0 0 0 1 0
##  [963] 0 0 0 1 1 0 1 1 1 1 1 0 1 1 1 1 0 0 1 0 1 0 0 1 0 1 0 0 1 1 1 0 1 1 1 1 0
## [1000] 1
## Levels: 0 1
```

```
# Displaying the top of the dataset
head(df1)
```

```
##    Daily_Time_Spent_on_Site   Age Daily_Internet_Usage  Male Clicked_on_Ad
##                       <num> <int>                <num> <int>        <fctr>
## 1:                    68.95    35               256.09     0             0
```

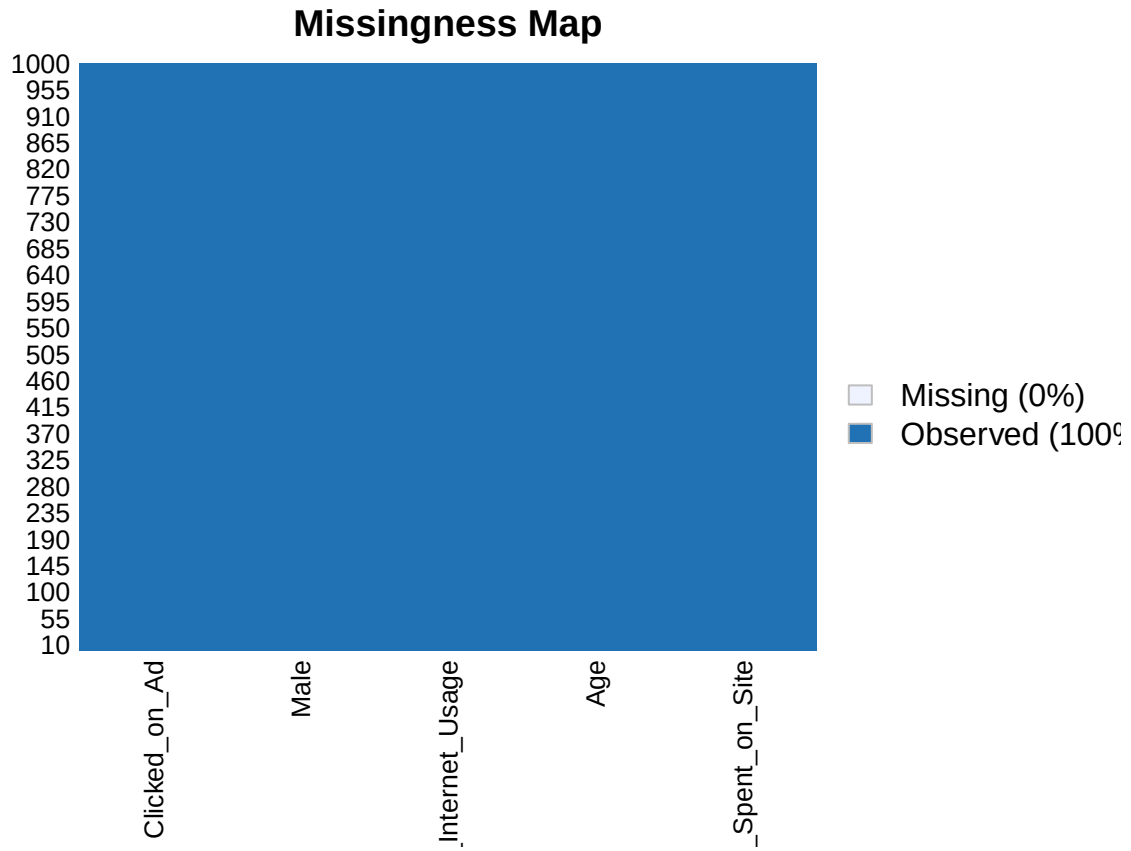
```
## 2:                    80.23    31              193.77     1             0
## 3:                    69.47    26              236.50     0             0
## 4:                    74.15    29              245.89     1             0
## 5:                    68.37    35              225.58     0             0
## 6:                    59.99    23              226.74     1             0
```

```
# Cleaning the dataset by setting zero values to NA's

df1[, 1:3][df[, 1:3] == 0] <- NA
```

Replacing the zero's to NA.

```
# Visualizing the dataset by checking for missing values
missmap(df1)
```

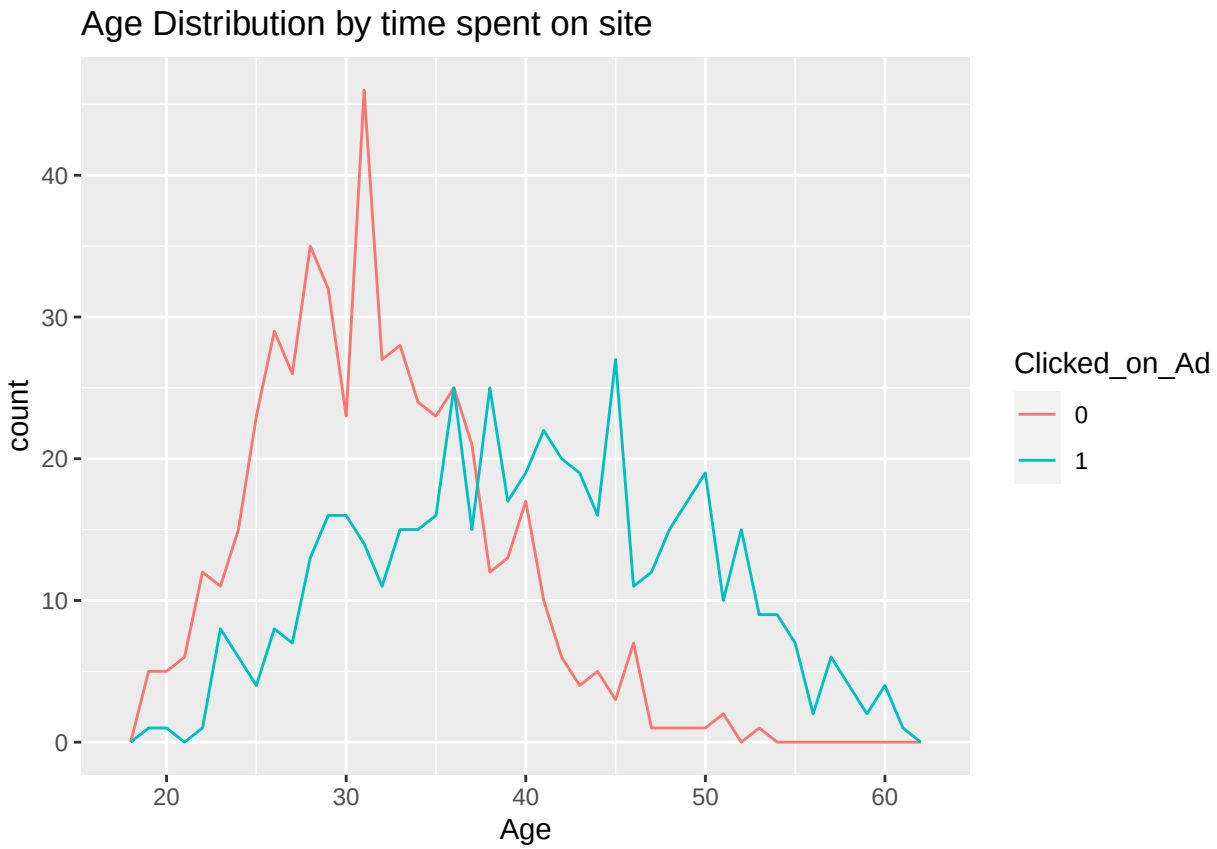


There are many missing values therefore removing them willreduce the dataset.

```
# Displaying the column names
colnames(df1)
```

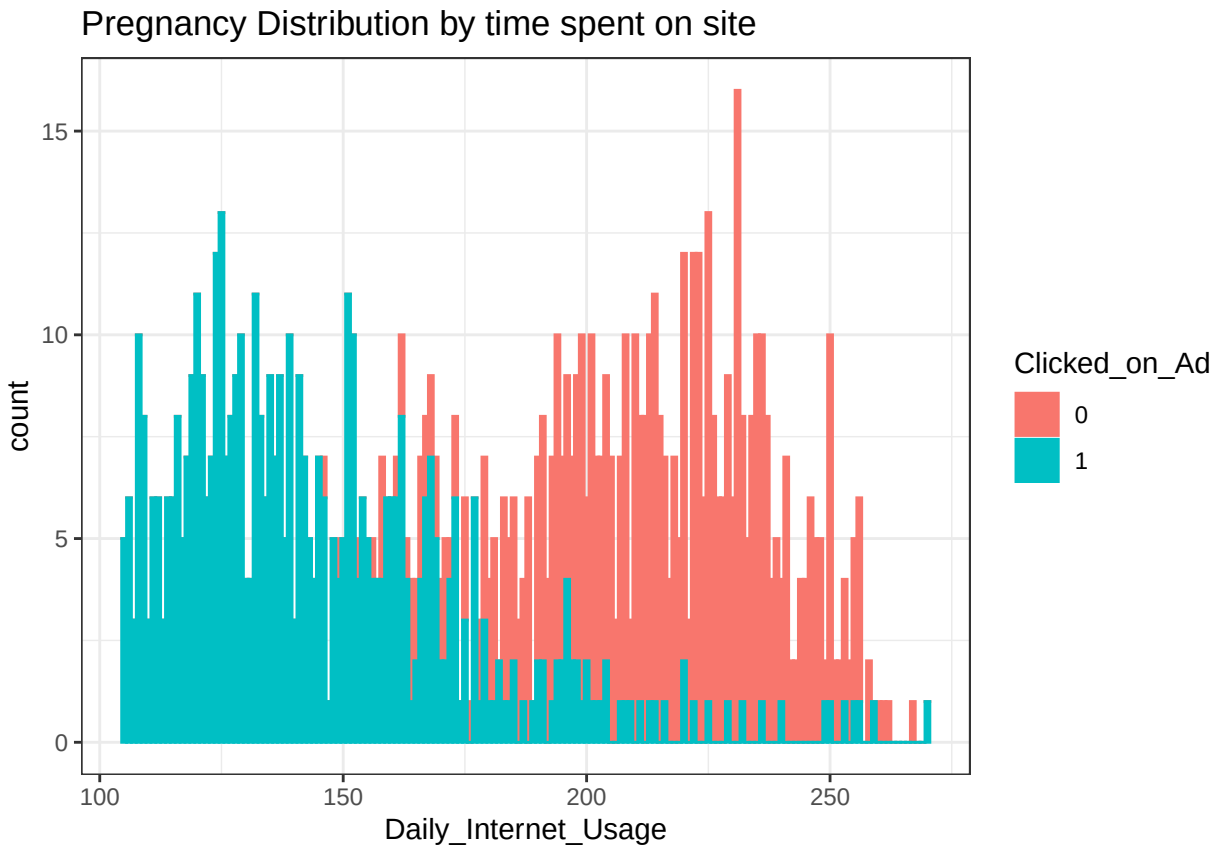
```
## [1] "Daily_Time_Spent_on_Site" "Age"
## [3] "Daily_Internet_Usage"     "Male"
## [5] "Clicked_on_Ad"
```

```
# Creating some visualisations to check each variable
# Visualisation 1
ggplot(df1, aes(Age, colour = Clicked_on_Ad)) +
geom_freqpoly(binwidth = 1) + labs(title="Age Distribution by time spent on site")
```



As the age increases the count reduces.

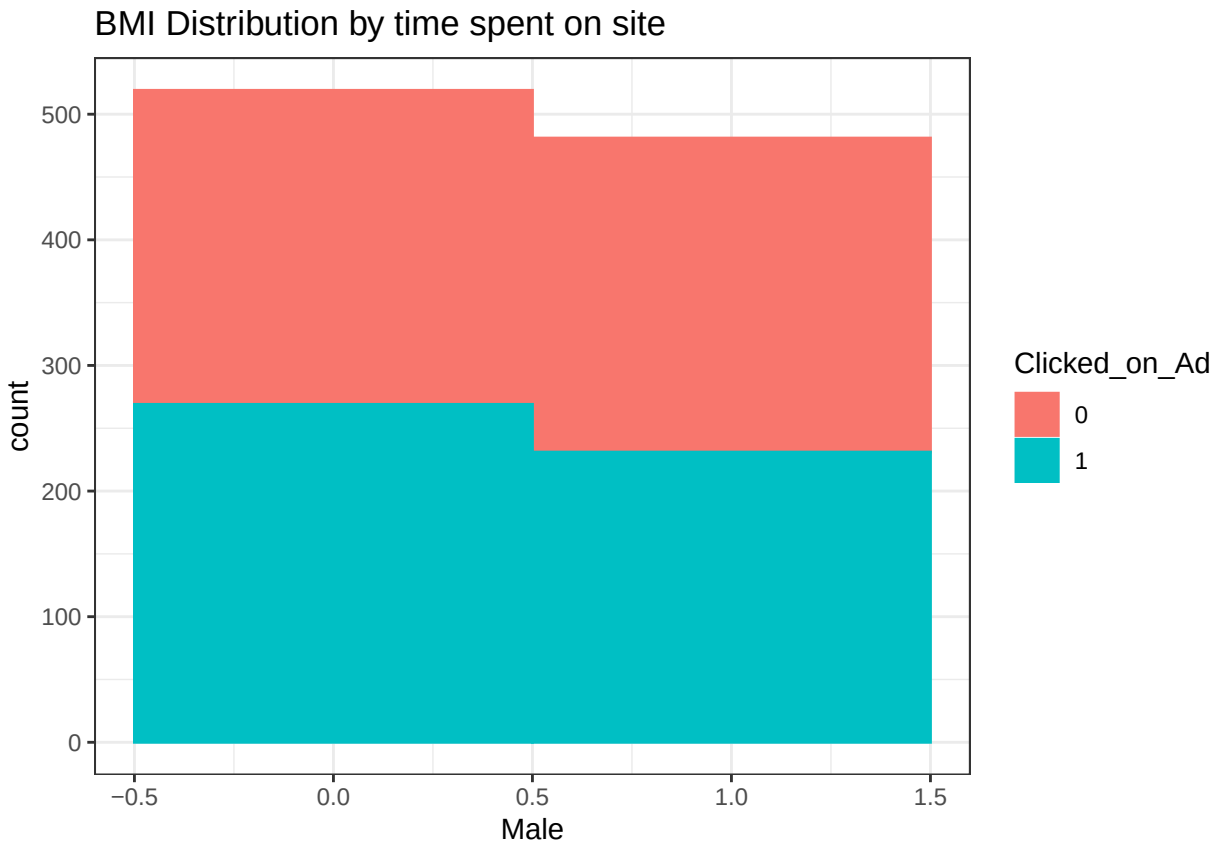
```
# Visualisation 2
c <- ggplot(df1, aes(x=Daily_Internet_Usage, fill=Clicked_on_Ad, color=Clicked_on_Ad)) +
geom_histogram(binwidth = 1) + labs(title="Pregnancy Distribution by time spent on site")
c + theme_bw()
```



The daily internet usage increases per couunt.

```
# Visualisation 3

P <- ggplot(df1, aes(x=Male, fill=Clicked_on_Ad, color=Clicked_on_Ad)) +
geom_histogram(binwidth = 1) + labs(title="BMI Distribution by time spent on site")
P + theme_bw()
```




```
# Splitting data into training and test data sets

indxTrain <- createDataPartition(y = df1$Clicked_on_Ad,p = 0.75,list = FALSE)
training <- df1[indxTrain,]
training
```

```
##      Daily_Time_Spent_on_Site   Age Daily_Internet_Usage  Male Clicked_on_Ad
##                         <num> <int>                <num> <int>        <fctr>
##   1:                    68.95    35               256.09     0             0
##   2:                    74.15    29               245.89     1             0
##   3:                    68.37    35               225.58     0             0
##   4:                    59.99    23               226.74     1             0
##   5:                    88.91    33               208.36     0             0
##  ---
## 746:                    43.70    28               173.01     0             1
## 747:                    72.97    30               208.58     1             1
## 748:                    51.30    45               134.42     1             1
## 749:                    51.63    51               120.37     1             1
## 750:                    55.55    19               187.95     0             0
```

```
testing <- df1[-indxTrain,]
testing
```

```
##      Daily_Time_Spent_on_Site   Age Daily_Internet_Usage  Male Clicked_on_Ad
##                         <num> <int>                <num> <int>        <fctr>
```

```
##   1:                    80.23    31              193.77     1             0
##   2:                    69.47    26              236.50     0             0
##   3:                    66.00    48              131.76     1             1
##   4:                    74.53    30              221.51     1             0
##   5:                    82.03    41              187.53     0             0
##  ---
## 246:                    40.18    29              151.96     0             1
## 247:                    50.48    50              162.43     0             1
## 248:                    54.97    31              116.38     1             1
## 249:                    67.26    57              168.41     1             1
## 250:                    45.01    26              178.35     0             1
```

Displaying the test dataset,

```
# Checking dimensions of the split

prop.table(table(df1$Clicked_on_Ad)) * 100
```

```
##
##  0  1
## 50 50
```

The dimensions are 50/50 split.

```
prop.table(table(testing$Clicked_on_Ad)) * 100
```

```
##
##  0  1
## 50 50
```

The dimensions for the testing clicked ad are 50/50.

```
prop.table(table(training$Clicked_on_Ad)) * 100
```

```
##
##  0  1
## 50 50
```

The dimensions for the training clicked ad are 50/50.

```
# Creating objects x which holds the predictor variables and y which holds the response variables

x = training[,-5]
y = training$Clicked_on_Ad
```

x and y predictor variables.

```
# Loading our inbuilt e1071 package that holds the Naive Bayes function.

library(e1071)
```

```r
# Building the model

model = train(x,y,'nb',trControl=trainControl(method='cv',number=10))
```

```
## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 18

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 8

## Warning in FUN(X[[i]], ...): Numerical 0 probability for all classes with
## observation 30
```

```r
# Model Evalution
# Predicting our testing set
Predict <- predict(model, newdata = testing)

# Confusion matrix to see accuracy value and other parameter values

confusionMatrix(Predict, testing$Clicked_on_Ad )
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 123   9
##          1   2 116
##
##                Accuracy : 0.956
##                  95% CI : (0.9226, 0.9778)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.912
##
##  Mcnemar's Test P-Value : 0.07044
##
##             Sensitivity : 0.9840
##             Specificity : 0.9280
##          Pos Pred Value : 0.9318
##          Neg Pred Value : 0.9831
##              Prevalence : 0.5000
##          Detection Rate : 0.4920
##    Detection Prevalence : 0.5280
##       Balanced Accuracy : 0.9560
##
##        'Positive' Class : 0
##
```

The confusion matrix of the model.

KNN

```
set.seed(1234)

# Randomizing the rows, creates a uniform distribution of 150
random <- runif(150)
df1_random <- df1[order(random),]
df1_random
```

```
##      Daily_Time_Spent_on_Site   Age Daily_Internet_Usage  Male Clicked_on_Ad
##                         <num> <int>                <num> <int>        <fctr>
##   1:                    88.91    33               208.36     0             0
##   2:                    86.06    32               178.92     1             0
##   3:                    55.35    39               153.17     1             1
##   4:                    39.94    41               156.30     0             1
##   5:                    41.49    53               169.18     0             1
##  ---
## 146:                    46.98    50               175.37     0             1
## 147:                    49.78    46               152.24     0             1
## 148:                    37.51    30               163.00     1             1
## 149:                    50.43    46               119.32     1             1
## 150:                    77.65    27               212.79     0             0
```

```
# Selecting the first 6 rows from iris_random
head(df1_random)
```

```
##    Daily_Time_Spent_on_Site   Age Daily_Internet_Usage  Male Clicked_on_Ad
##                       <num> <int>                <num> <int>        <fctr>
## 1:                    88.91    33               208.36     0             0
## 2:                    86.06    32               178.92     1             0
## 3:                    55.35    39               153.17     1             1
## 4:                    39.94    41               156.30     0             1
## 5:                    41.49    53               169.18     0             1
## 6:                    74.02    32               210.54     0             0
```

```
normal <- function(x) (
  return( ((x - min(x)) /(max(x)-min(x))) )
)
normal(1:5)
```

```
## [1] 0.00 0.25 0.50 0.75 1.00
```

```
df1_new <- as.data.frame(lapply(df1_random[,-5], normal))
summary(df1_new)
```

```
##  Daily_Time_Spent_on_Site      Age          Daily_Internet_Usage
##  Min.   :0.0000           Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.2986           1st Qu.:0.2500   1st Qu.:0.1903
##  Median :0.5959           Median :0.4000   Median :0.4430
##  Mean   :0.5425           Mean   :0.4138   Mean   :0.4427
```

```
##  3rd Qu.:0.7752           3rd Qu.:0.5750   3rd Qu.:0.6633
##  Max.   :1.0000           Max.   :1.0000   Max.   :1.0000
##       Male
##  Min.   :0.0000
##  1st Qu.:0.0000
##  Median :0.0000
##  Mean   :0.4533
##  3rd Qu.:1.0000
##  Max.   :1.0000
```

Normalization of the dataset.

```
# Lets now create test and train data sets

train <- df1_new[1:130,]
test <- df1_new[131:150,]
train_sp <- df1_random[1:130,5]
test_sp <- df1_random[131:150,5]
```

Testing and training the dataset.

```
df1_train_labels <- df1[1:65, 1]
df1_test_labels <- df1[66:100, 1]
```

5. CONCLUSION

The people that clicked on the ads on the blog were aged between 19 yrs and 61 years old.

The internet usage ranged between 104.8 to 269 units with the time spent on the blog was between 32 to 91 minutes.

There was a negative correlation between age and daily time spent on Site of the individuals.

The ads were mostly viewed by the young and middle aged audience.

6. RECOMMENDATIONS

The ads that should be placed on the blog should be relevant to the ages so that the individuals can click on the ads.

For the older people they can minimize the ads and for the younger people they can maximize the ads so that each can relate to ads accordingly.