# Week 12 Independent Project

Wilkister Mbaka

2022-07-15

## Advertising - Supervised Learning

### Defining the Question

#### Specifying the Question

Perform exploratory data analysis on the provided dataset and identify which individuals are most likely to click on her ads.

#### Defining the metrics of success

To perform univariate and bivariate data analysis and based on that, provide recommendations and on which individuals are most likely to click on her ads.

#### Understanding the context

A Kenyan entrepreneur has created an online cryptography course and would want to advertise it on her blog. She currently targets audiences originating from various countries. In the past, she ran ads to advertise a related course on the same blog and collected data in the process. She would now like to employ your services as a Data Science Consultant to help her identify which individuals are most likely to click on her ads.

#### Experimental design taken

1) Load the dataset
2) Clean the dataset.
3) perform Exploratory data analysis

### Reading The Data

```
#installing packages
library(data.table)
library(ggplot2)

#
#Loading the dataset
advert <- fread("http://bit.ly/IPAdvertisingData")
```

### Checking The Data

```
#Previewing the first 6 rows
head(advert)
```

```
##    Daily Time Spent on Site Age Area Income Daily Internet Usage
## 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
## 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
## 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 last 6 rows
tail(advert)
```

```
##    Daily Time Spent on Site Age Area Income Daily Internet Usage
## 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
## 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
## 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
```

```
# Dimensionanity of the data
dim(advert)
```

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

There are 1000 rows and 10 columns

```
#Checking the datatypes
str(advert)
```

```
## Classes 'data.table' and 'data.frame':   1000 obs. of  10 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 ...
##  $ Area Income             : num  61834 68442 59786 54806 73890 ...
##  $ Daily Internet Usage    : num  256 194 236 246 226 ...
##  $ Ad Topic Line           : chr  "Cloned 5thgeneration orchestration" "Monitored national standardi
##  $ City                    : chr  "Wrightburgh" "West Jodi" "Davidton" "West Terrifurt" ...
##  $ Male                    : int  0 1 0 1 0 1 0 1 1 1 ...
##  $ Country                 : chr  "Tunisia" "Nauru" "San Marino" "Italy" ...
##  $ Timestamp               : POSIXct, format: "2016-03-27 00:53:11" "2016-04-04 01:39:02" ...
##  $ Clicked on Ad           : int  0 0 0 0 0 0 0 1 0 0 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
# The dataset has integer, number, character and datetime datatype
```

## Tidying The Dataset

```
#Finding the total number of missing values in each column
colSums(is.na(advert))
```

```
## Daily Time Spent on Site                      Age              Area Income
##                        0                        0                        0
##     Daily Internet Usage            Ad Topic Line                     City
##                        0                        0                        0
##                     Male                  Country                Timestamp
##                        0                        0                        0
##            Clicked on Ad
##                        0
```

There are no missing values in the dataset

```
#Finding duplicated entries within the dataset
duplicated_rows <- advert[duplicated(advert),]
#Printing out the duplicated entries
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 in the datset

**Checking for outliers**

```
#Checking for outliers using boxplot
library(dplyr)
```

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

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

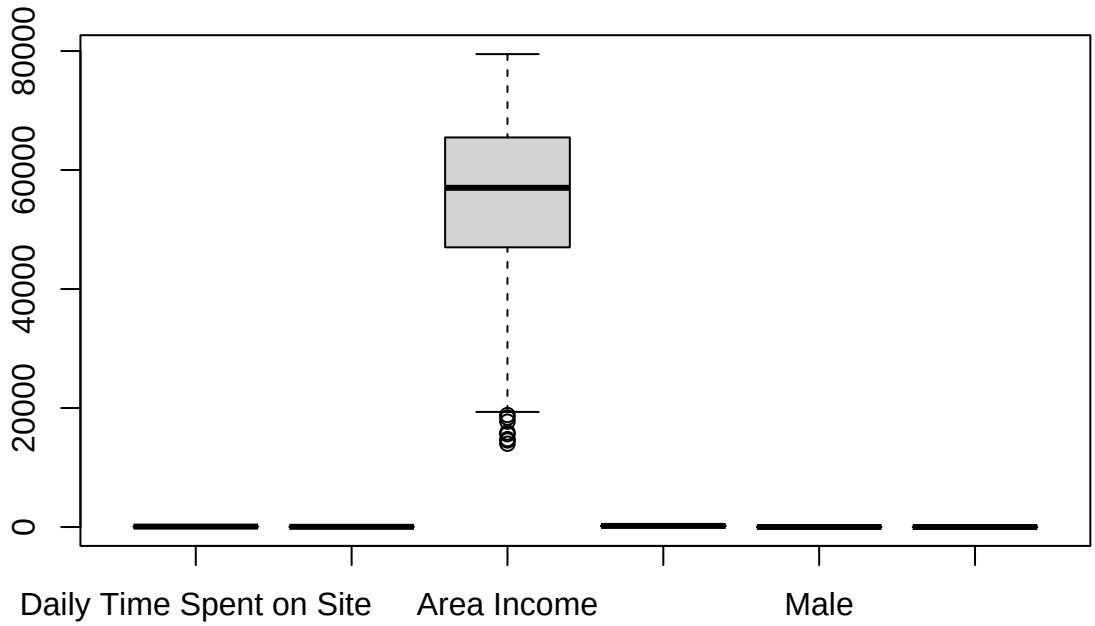
```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

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

```
data_num2 <- select_if(advert, is.numeric)
data_num2
```

```
##       Daily Time Spent on Site Age Area Income Daily Internet Usage Male
##    1:                    68.95  35    61833.90                256.09    0
##    2:                    80.23  31    68441.85                193.77    1
##    3:                    69.47  26    59785.94                236.50    0
##    4:                    74.15  29    54806.18                245.89    1
##    5:                    68.37  35    73889.99                225.58    0
##   ---
##  996:                    72.97  30    71384.57                208.58    1
##  997:                    51.30  45    67782.17                134.42    1
##  998:                    51.63  51    42415.72                120.37    1
##  999:                    55.55  19    41920.79                187.95    0
## 1000:                    45.01  26    29875.80                178.35    0
##       Clicked on Ad
##    1:             0
##    2:             0
##    3:             0
##    4:             0
##    5:             0
##   ---
##  996:             1
##  997:             1
##  998:             1
##  999:             0
## 1000:             1
```

```
boxplot(data_num2)
```




```
#We notice that outliers are only available in the Area income column, but since they represent
#income different areas, we fail to drop them
```

## Exploratory Data Analysis

### Univariate Analysis

### Numerical variables Measures of Central Tendency

### Mean

```
#Finding the mean of the numerical columns
advert_mean1 <- mean(advert$`Daily Time Spent on Site`)
advert_mean2 <- mean(advert$Age)
advert_mean3 <- mean(advert$`Area Income`)
advert_mean4 <- mean(advert$`Daily Internet Usage`)
#Printing out the results for daily time spent on site
advert_mean1
```

```
## [1] 65.0002
```

```
#Printing results for Age
advert_mean2
```

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

```
#Printing the results for Area income
advert_mean3
```

```
## [1] 55000
```

```
#Printing results for daily internet usage
advert_mean4
```

```
## [1] 180.0001
```

```
#
```

### Median

```
#Finding the Median of the numerical columns
advert_median1 <- median(advert$`Daily Time Spent on Site`)
advert_median2 <- median(advert$Age)
advert_median3 <- median(advert$`Area Income`)
advert_median4 <- median(advert$`Daily Internet Usage`)
#Printing out the results for daily time spent on site
advert_median1
```

```
## [1] 68.215
```

```
#Printing results for Age
advert_median2
```

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

```
#Printing the results for Area income
advert_median3
```

```
## [1] 57012.3
```

```
#Printing results for daily internet usage
advert_median4
```

```
## [1] 183.13
```

### Mode

```r
#Creating a function for finding mode
getmode <- function(v) {
  uniqv <- unique(v)
  uniqv[which.max(tabulate(match(v, uniqv)))]
}
#Calculating the mode of each column
advert_mode1 <- getmode(advert$`Daily Time Spent on Site`)
advert_mode2 <- getmode(advert$Age)
advert_mode3 <- getmode(advert$`Area Income`)
advert_mode4 <- getmode(advert$`Daily Internet Usage`)

#Printing out the results for daily time spent on site
advert_mode1
```

```
## [1] 62.26
```

```r
#Printing results for Age
advert_mode2
```

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

```r
#Printing the results for Area income
advert_mode3
```

```
## [1] 61833.9
```

```r
#Printing results for daily internet usage
advert_mode4
```

```
## [1] 167.22
```

### Measures of Dispersion

### Maximum

```r
#Finding the maximum values in each cloumn
advert_max1 <- max(advert$`Daily Time Spent on Site`)
advert_max2 <- max(advert$Age)
advert_max3 <- max(advert$`Area Income`)
advert_max4 <- max(advert$`Daily Internet Usage`)

#Printing out the results for daily time spent on site
advert_max1
```

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

```r
#Printing results for Age
advert_max2
```

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

```
#Printing the results for Area income
advert_max3
```

```
## [1] 79484.8
```

```
#Printing results for daily internet usage
advert_max4
```

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

**Minimum**

```
#Finding the minimum values in each column
advert_min1 <- min(advert$`Daily Time Spent on Site`)
advert_min2 <- min(advert$Age)
advert_min3 <- min(advert$`Area Income`)
advert_min4 <- min(advert$`Daily Internet Usage`)

#Printing out the results for daily time spent on site
advert_min1
```

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

```
#Printing results for Age
advert_min2
```

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

```
#Printing the results for Area income
advert_min3
```

```
## [1] 13996.5
```

```
#Printing results for daily internet usage
advert_min4
```

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

**Quantiles**

```
#Finding the quantiles in each cloumn
advert_quan1 <- quantile(advert$`Daily Time Spent on Site`)
advert_quan2 <- quantile(advert$Age)
advert_quan3 <- quantile(advert$`Area Income`)
advert_quan4 <- quantile(advert$`Daily Internet Usage`)
#Printing out the results for daily time spent on site
advert_quan1
```

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

```
#Printing results for Age
advert_quan2
```

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

```
#Printing the results for Area income
advert_quan3
```

```
##       0%      25%      50%      75%     100%
## 13996.50 47031.80 57012.30 65470.64 79484.80
```

```
#Printing results for daily internet usage
advert_quan4
```

```
##       0%      25%      50%      75%     100%
## 104.7800 138.8300 183.1300 218.7925 269.9600
```

**Variance**

```
#Finding the variance in each cloumn
advert_var1 <- var(advert$`Daily Time Spent on Site`)
advert_var2 <- var(advert$Age)
advert_var3 <- var(advert$`Area Income`)
advert_var4 <- var(advert$`Daily Internet Usage`)
#Printing out the results for daily time spent on site
advert_var1
```

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

```
#Printing results for Age
advert_var2
```

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

```
#Printing the results for Area income
advert_var3
```

```
## [1] 179952406
```

```
#Printing results for daily internet usage
advert_var4
```

```
## [1] 1927.415
```

**Standard Deviation**

```r
#Finding the standard deviation in each cloumn
advert_sd1 <- sd(advert$`Daily Time Spent on Site`)
advert_sd2 <- sd(advert$Age)
advert_sd3 <- sd(advert$`Area Income`)
advert_sd4 <- sd(advert$`Daily Internet Usage`)
#Printing out the results for daily time spent on site
advert_sd1
```

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

```r
#Printing results for Age
advert_sd2
```

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

```r
#Printing the results for Area income
advert_sd3
```

```
## [1] 13414.63
```

```r
#Printing results for daily internet usage
advert_sd4
```

```
## [1] 43.90234
```

**Skeweness**

```r
#importing the necessary packages
library(moments)
#Finding the skewness in each cloumn
advert_sk1 <- skewness(advert$`Daily Time Spent on Site`)
advert_sk2 <- skewness(advert$Age)
advert_sk3 <- skewness(advert$`Area Income`)
advert_sk4 <- skewness(advert$`Daily Internet Usage`)

#Printing out the results for daily time spent on site
#negative value which interprets that majority of the data are greater than the mean which
#can also be interpreted that most data are concetrated on the right side of the tail.
advert_sk1
```

```
## [1] -0.3712026
```

```r
#Printing results for Age
#From the results below we can note that Age column has a positive skewness meaning majority of
#the data are less than the mean
advert_sk2
```

```
## [1] 0.4784227
```

```
#Printing the results for Area income
#The negative value which interprets that majority of the data are greater than the
#mean which can also be interpreted that most data are concetrated on the right side of the tail.
advert_sk3
```

```
## [1] -0.6493967
```

```
#Printing results for daily internet usage
#Daily internet usage column has a value close to 0 meaning its data is normally distributed.
advert_sk4
```

```
## [1] -0.03348703
```

**Kurtosis**

```
#Finding the skewness in each cloumn
advert_kr1 <- kurtosis(advert$`Daily Time Spent on Site`)
advert_kr2 <- kurtosis(advert$Age)
advert_kr3 <- kurtosis(advert$`Area Income`)
advert_kr4 <- kurtosis(advert$`Daily Internet Usage`)
#Printing out the results for daily time spent on site
advert_kr1
```

```
## [1] 1.903942
```

```
#Printing results for Age
advert_kr2
```

```
## [1] 2.595482
```

```
#Printing the results for Area income
advert_kr3
```

```
## [1] 2.894694
```

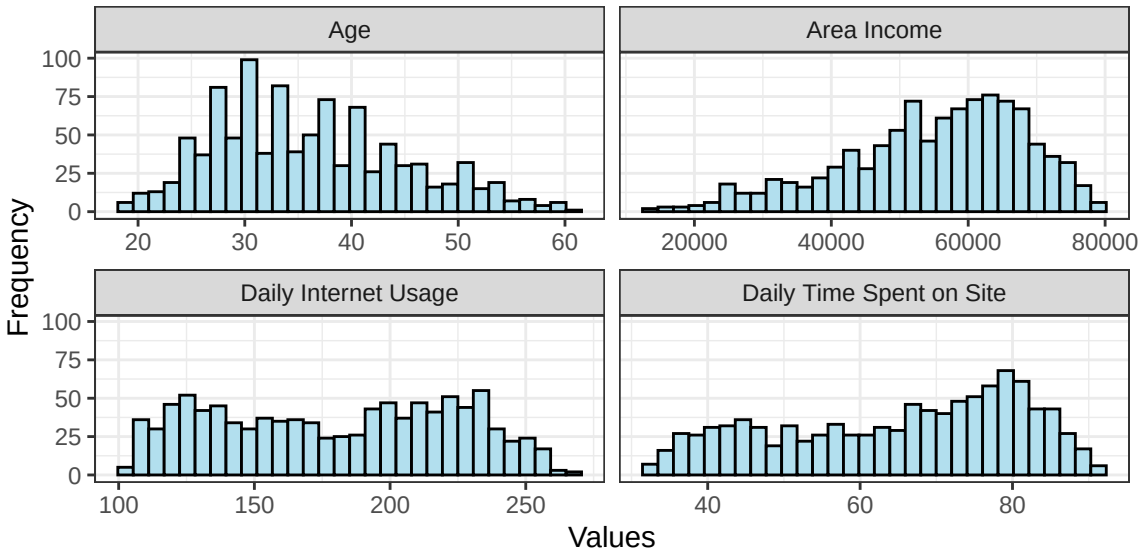
```
#Printing results for daily internet usage
advert_kr4
```

```
## [1] 1.727701
```

```
#All the kurtosis values are less than 3 which is called Platykurtic.
```

```
library (tidyr)
advert %>%
  gather(attributes, value, 1:4) %>%
  ggplot(aes(x = value)) +
  geom_histogram(fill = 'lightblue2', color = 'black') +
  facet_wrap(~attributes, scales = 'free_x') +
  labs(x="Values", y="Frequency") +
  theme_bw()
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```




1. Area Income is left skewed with the bracket between 60000 to 80000 having a high number of visitors to the site
2. Age is left skewed with the age bracket with a higher number of visitors is 30-31

**Categorical Values**

**Ad Topic Line**

```
#Unique values in the column

uniq_topic <- unique(advert$`Ad Topic Line`, )
length(uniq_topic)
```

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

There are 1000 unique topic lines meaning it would be impossible to get a good visualization.

**City**

```
#Unique values in the column

uniq_city <- unique(advert$City, )
length(uniq_city)
```

```
## [1] 969
```

There are 969 unique cities hence it would also be impossible to get a good visualization

**Country**

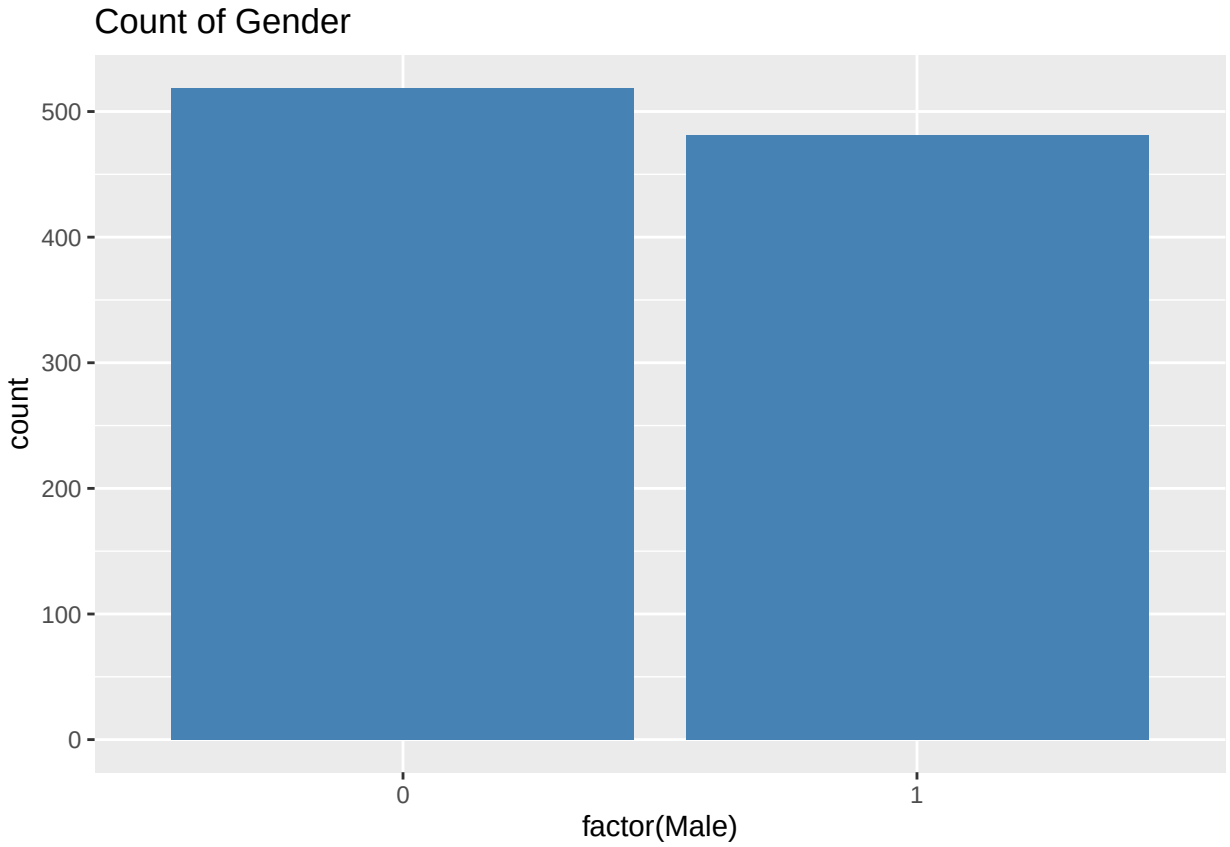
```r
#Unique values in the column
uniq_country <- unique(advert$Country)
length(uniq_country)
```

```
## [1] 237
```

There are 237 unique countries.

**Gender**

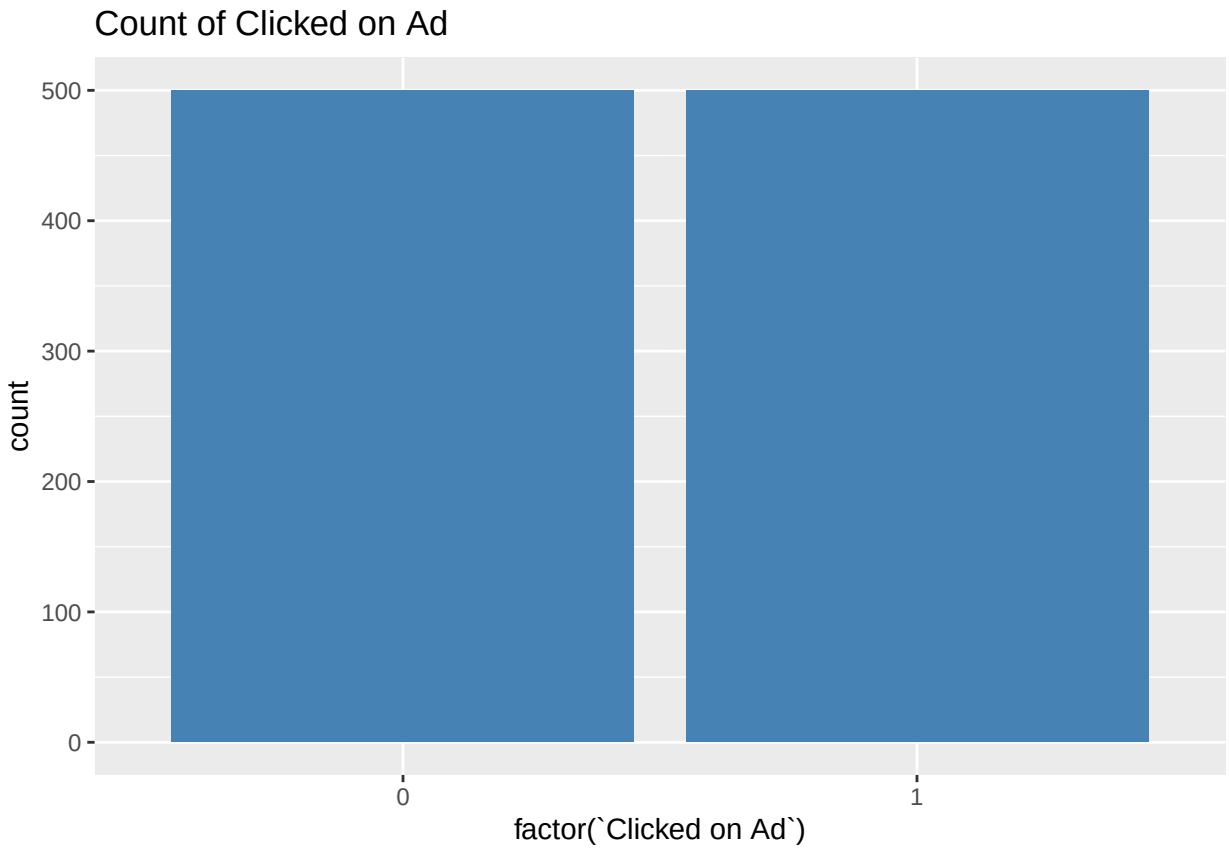
```r
### Gender
ggplot(advert, aes(x = factor(Male))) +
  geom_bar(fill="steelblue") +  ggtitle('Count of Gender')
```




Slightly more females than male(0 is female)

**Clicked on Ad**

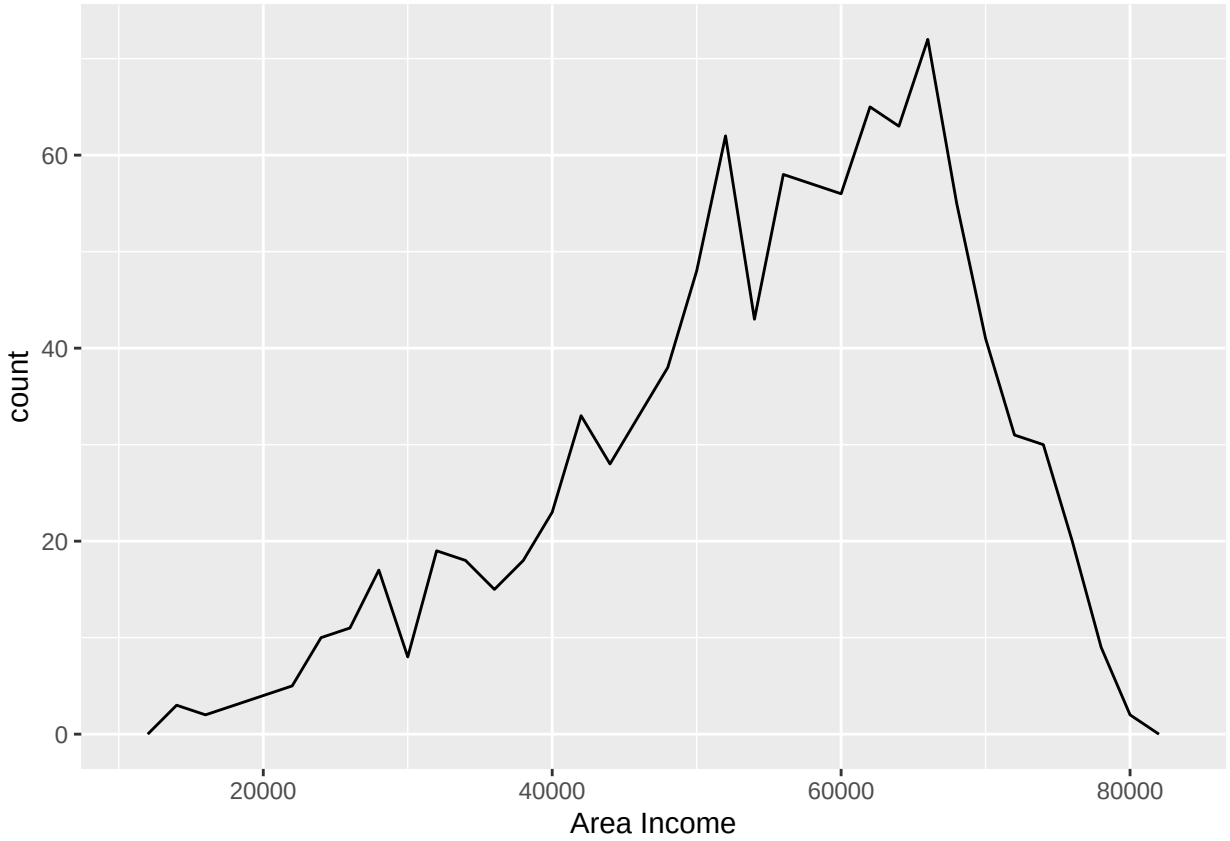
```r
# Countplot for clicked on ad column
# This is to see the number of people that clicked on the ad

ggplot(advert, aes(x = factor(`Clicked on Ad`))) +
  geom_bar(fill="steelblue") +  ggtitle('Count of Clicked on Ad')
```




There are equal values of clicked and not clicked on ad

**Bivariate analysis**

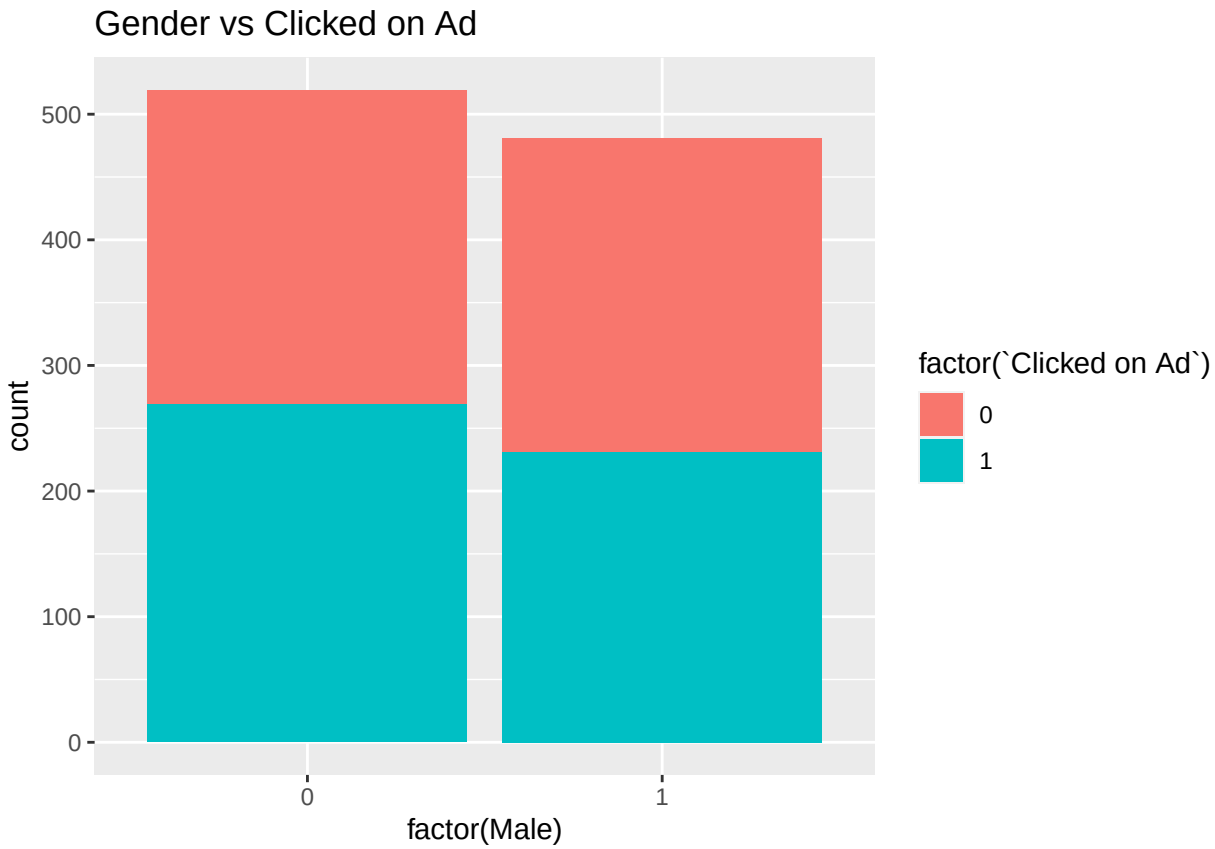
```
#Area income vs clicked on ad
ggplot(data = advert, mapping = aes(x = `Area Income`)) +
  geom_freqpoly(mapping = aes(colour = `Clicked on Ad`), binwidth = 2000)
```




As income increases, the rate of clicking on the Ad also increases

**Gender vs Clicked on Ad**

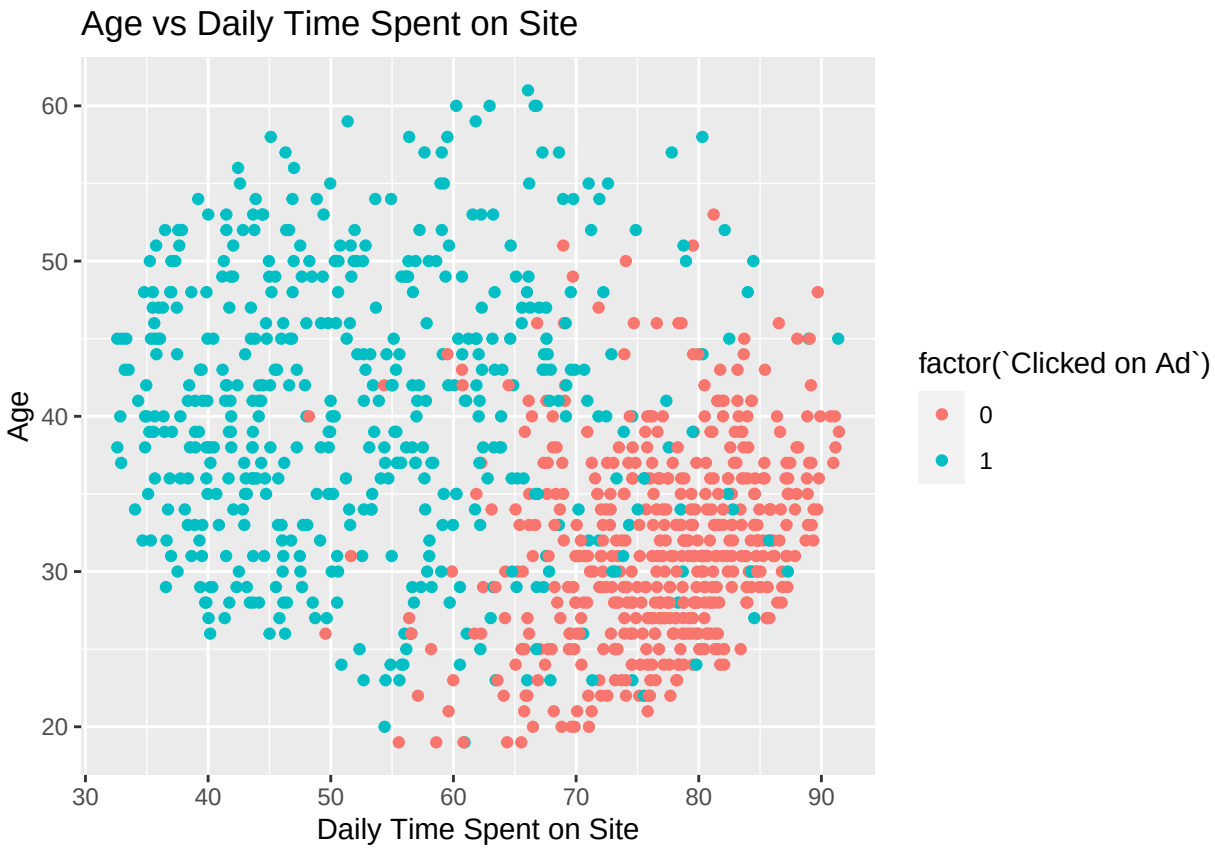
```
# How the genders click on the Ad
ggplot(advert, aes(x = factor(Male), fill = factor(`Clicked on Ad`))) +
  geom_bar() +  ggtitle('Gender vs Clicked on Ad')
```




From the data, We have slightly more women than men visiting the site and more women clicking on the Ad.

**Age vs Daily Time Spent on Site**

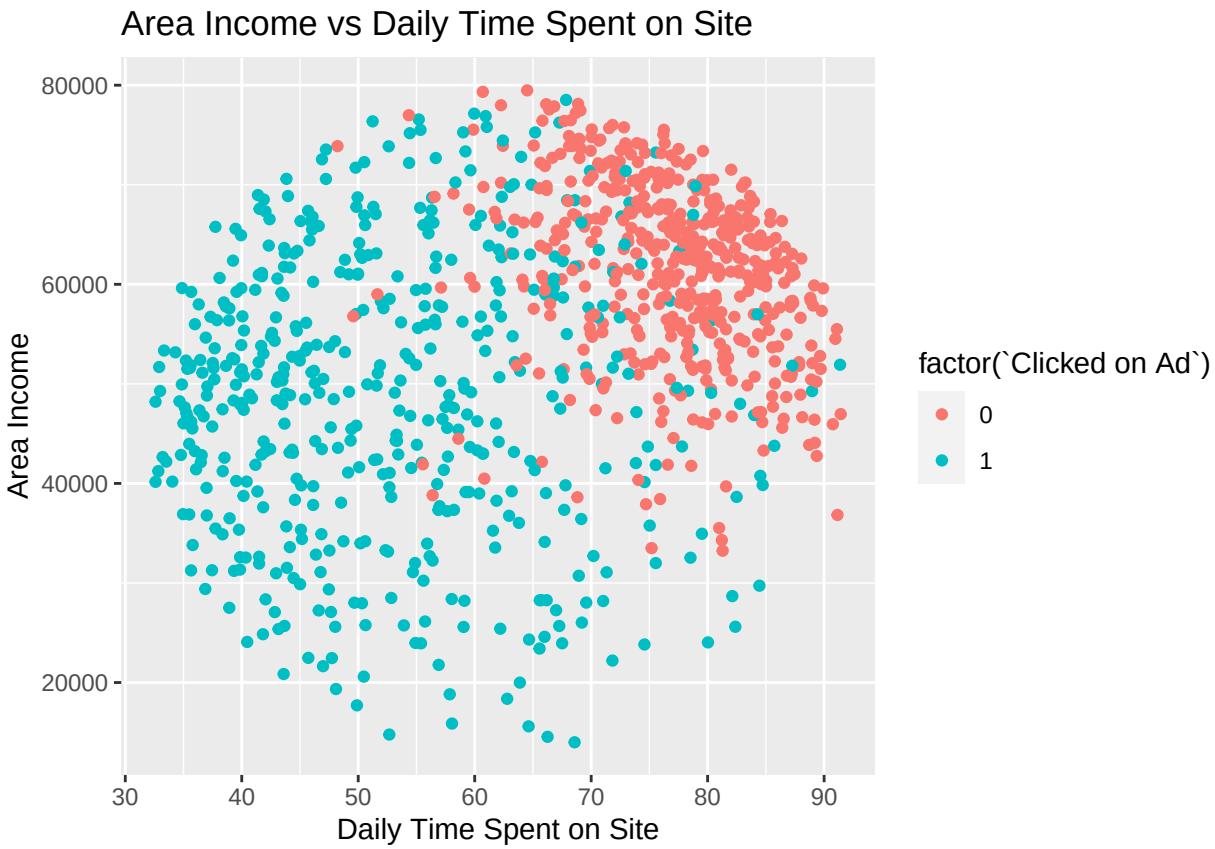
```
# Which Age group visits the site regularly
ggplot(advert, aes(x = `Daily Time Spent on Site`, y = Age, col = factor(`Clicked on Ad`))) +
  geom_point() +  ggtitle('Age vs Daily Time Spent on Site')
```




Visitors to the site who spent less time on the site were likely to click on the Ad

**Area Income vs Daily Time Spent on Site**

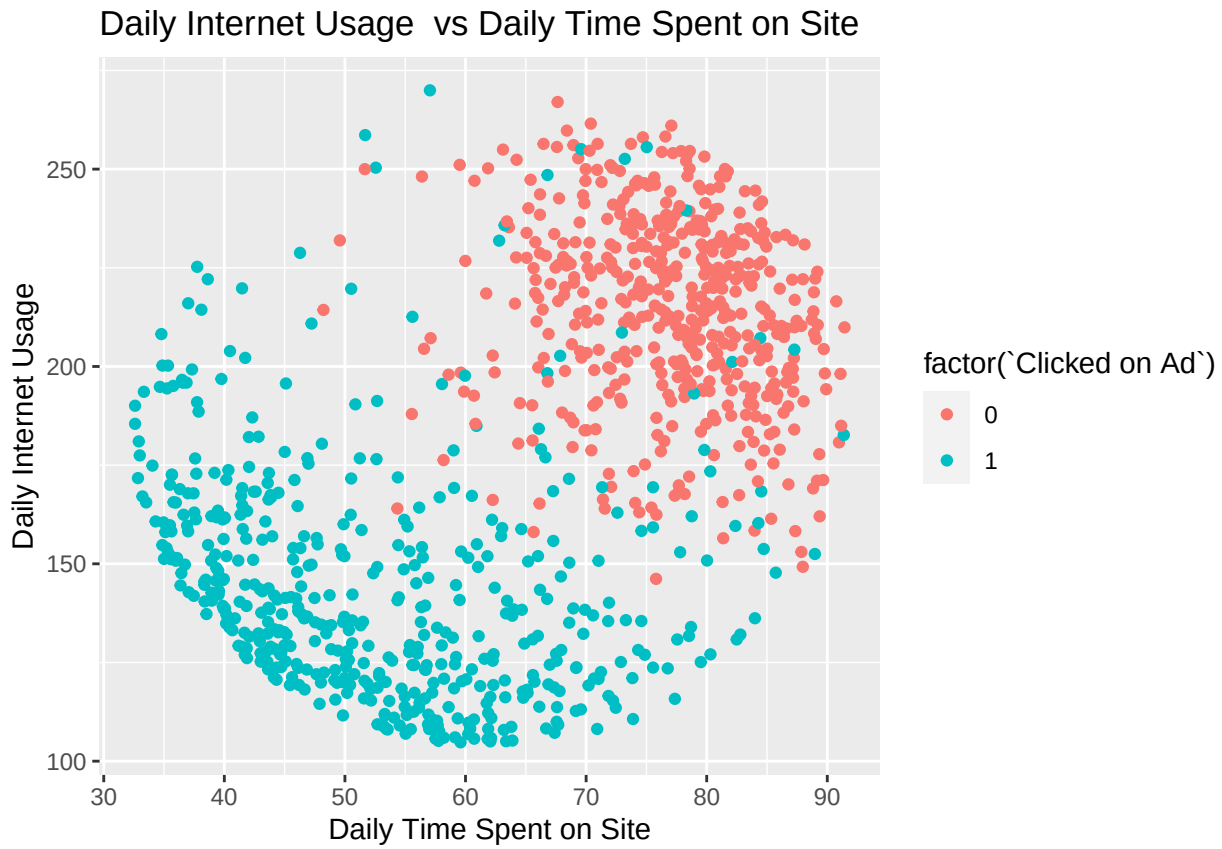
```
# How the area income relates to the daily time spent on site
ggplot(advert, aes(x = `Daily Time Spent on Site`, y = `Area Income`, col = factor(`Clicked on Ad`))) +
  geom_point() +  ggtitle('Area Income vs Daily Time Spent on Site')
```




People who live in high income areas and spend more time on the sight are highly unlikely to click on the Ad

**Daily Internet Usage vs Daily Time Spent on Site**

```
ggplot(advert, aes(x = `Daily Time Spent on Site`, y = `Daily Internet Usage`, col = factor(`Clicked on
  geom_point() +  ggtitle('Daily Internet Usage  vs Daily Time Spent on Site')
```




People whose daily internet usage is less and also the daily time spent on site is less are highly likey to click on the Ad

**Correlation**

```
#Finding the correlation of the numerical columns
# Identify numeric columns
library("dplyr")
# Subset numeric columns with dplyr
data_num3 <- select_if(advert, is.numeric)
data_num3
```

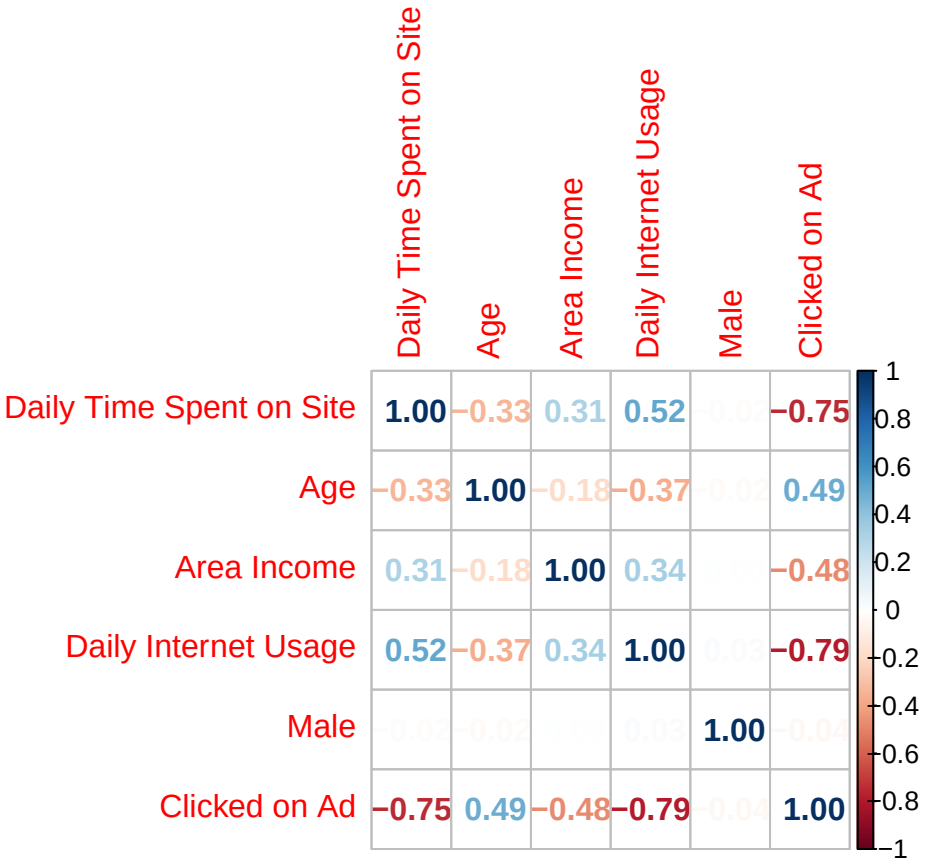
```
##       Daily Time Spent on Site Age Area Income Daily Internet Usage Male
##    1:                    68.95  35    61833.90                256.09    0
##    2:                    80.23  31    68441.85                193.77    1
##    3:                    69.47  26    59785.94                236.50    0
##    4:                    74.15  29    54806.18                245.89    1
##    5:                    68.37  35    73889.99                225.58    0
##   ---
##  996:                    72.97  30    71384.57                208.58    1
##  997:                    51.30  45    67782.17                134.42    1
##  998:                    51.63  51    42415.72                120.37    1
##  999:                    55.55  19    41920.79                187.95    0
## 1000:                    45.01  26    29875.80                178.35    0
##       Clicked on Ad
##    1:             0
##    2:             0
```

```
##    3:             0
##    4:             0
##    5:             0
##   ---
##  996:             1
##  997:             1
##  998:             1
##  999:             0
## 1000:             1
```

```
# computing correlation matrix
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
#Assigning m to the correlation
# correlation matrix
M<-cor(data_num2)
corrplot(M, method="number")
```




From the correlation plot 1. There is a moderate positive correlation of 0.52 between Daily internet usage and Daily time spent on site.

2. There is also a weak positive correlation of 0.31 between area income and daily time spent on site
3. There is a high negative correlation of -0.75 between clicked on ad and daily time spent on site
4. There is a high negative correlation of -0.79 between clicked on ad and daily internet usage

## Supervised Learning

### SVM

```
#Create a copy of our dataframe
advert_model <- data.frame(advert)            # Create copy of data

#Drop unnecessary columns. We will drop Ad topic line, City, Country, Timestamp
#because they are categorical almost all or all rows have different values.
advert_model = subset(advert_model, select = -c(Timestamp,`Ad.Topic.Line`,City,Country) )
head(advert_model)
```

```
##   Daily.Time.Spent.on.Site Age Area.Income Daily.Internet.Usage Male
## 1                    68.95  35    61833.90               256.09    0
## 2                    80.23  31    68441.85               193.77    1
## 3                    69.47  26    59785.94               236.50    0
## 4                    74.15  29    54806.18               245.89    1
## 5                    68.37  35    73889.99               225.58    0
## 6                    59.99  23    59761.56               226.74    1
##   Clicked.on.Ad
## 1             0
## 2             0
## 3             0
## 4             0
## 5             0
## 6             0
```

```
# Convert gender_vector to a factor
advert_model[["Clicked.on.Ad"]] = factor(advert_model[["Clicked.on.Ad"]])
```

```
# Next we split the data into training set and testing set.
# NB: The training set will be used for model building while the testing set for model evaluation.
# ---
# - The "y" parameter takes the value of variable according to which data needs to be partitioned.
# In our case, target variable is at V14, so we are passing heart$V14
# - The "p" parameter holds a decimal value in the range of 0-1. It's to show the percentage of the spl
# We are using p=0.7. It means that data split should be done in 70:30 ratio.
# So, 70% of the data is used for training and the remaining 30% is for testing the model.
# - The "list" parameter is for whether to return a list or matrix.
# We are passing FALSE for not returning a list
# ---
#
library(caret)
```

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

```
intrain <- createDataPartition(y = advert_model$Clicked.on.Ad, p= 0.7, list = FALSE)
training <- advert_model[intrain,]
testing <- advert_model[-intrain,]
```

```r
# Before we train our model we will need to control all the computational overheads.
# We will implement this through the trainControl() method.
# This will allow us to use the train() function provided by the caret package.
# ---
# The trainControl method will take three parameters:
# a) The "method" parameter defines the resampling method,
# in this demo we'll be using the repeatedcv or the repeated cross-validation method.
# b) The next parameter is the "number", this basically holds the number of resampling iterations.
# c) The "repeats " parameter contains the sets to compute for our repeated cross-validation.
# We are using setting number =10 and repeats =3
# ---
#
trctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)

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

```r
# We can then check the reult of our train() model as shown below
# ---
#
svm_Linear
```

```
## Support Vector Machines with Linear Kernel
##
## 700 samples
##   5 predictor
##   2 classes: '0', '1'
##
## Pre-processing: centered (5), scaled (5)
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 630, 630, 630, 630, 630, 630, ...
## Resampling results:
##
##   Accuracy  Kappa
##   0.972381  0.9447619
##
## Tuning parameter 'C' was held constant at a value of 1
```

```r
# We can use the predict() method for predicting results as shown below.
# We pass 2 arguements, our trained model and our testing data frame.
# ---
#
test_pred <- predict(svm_Linear, newdata = testing)
test_pred
```

```
##   [1] 0 0 0 1 1 1 1 1 0 0 1 1 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 0 1 0 0
##  [38] 1 1 0 0 0 1 1 0 1 0 1 1 0 1 0 1 1 0 0 1 1 0 1 0 1 1 1 1 1 0 0 0 0 1 0 0 1
##  [75] 0 1 0 1 0 1 0 0 0 1 0 0 1 1 0 0 0 0 0 0 0 1 0 1 0 1 0 0 0 1 0 0 0 1 1 1 0
## [112] 0 1 0 0 0 0 1 1 0 0 1 0 0 0 0 0 1 0 0 0 0 1 0 1 1 1 1 1 0 1 0 0 1 1 0 1 1
## [149] 0 1 0 1 1 0 1 1 1 1 1 0 1 1 0 0 0 0 0 0 1 0 1 1 1 0 0 1 0 1 0 0 1 1 0 1 1
```

```
## [186] 0 1 1 0 0 1 0 1 1 1 0 1 0 1 0 1 0 0 1 0 1 1 1 1 1 0 1 0 0 1 1 1 1 0 0 0 1
## [223] 0 0 0 0 1 0 1 1 0 1 1 1 1 1 1 0 1 0 0 0 1 0 0 0 1 1 1 0 0 1 0 0 0 1 1 0 1
## [260] 1 0 0 1 0 1 0 1 1 0 1 1 1 0 1 0 1 0 1 0 1 0 0 1 1 1 1 1 1 0 0 0 0 1 1 0 1
## [297] 1 0 1 1
## Levels: 0 1
```

```
# Now checking for our accuracy of our model by using a confusion matrix
# ---
#
confusionMatrix(table(test_pred, testing$Clicked.on.Ad))
```

```
## Confusion Matrix and Statistics
##
##
## test_pred   0   1
##         0 144   6
##         1   6 144
##
##                Accuracy : 0.96
##                  95% CI : (0.9312, 0.9792)
##     No Information Rate : 0.5
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.92
##
##  Mcnemar's Test P-Value : 1
##
##             Sensitivity : 0.96
##             Specificity : 0.96
##          Pos Pred Value : 0.96
##          Neg Pred Value : 0.96
##              Prevalence : 0.50
##          Detection Rate : 0.48
##    Detection Prevalence : 0.50
##       Balanced Accuracy : 0.96
##
##        'Positive' Class : 0
##
```

The accuracy of this model is 97.67% which is good.

# Challenging the Solution

## KNN

```
# Normalizing the numerical variables of the data set.
df_norm <- as.data.frame(apply(advert_model[1:5], 2, function(x) (x - min(x))/(max(x)-min(x))))
head(df_norm)
```

```
##   Daily.Time.Spent.on.Site       Age Area.Income Daily.Internet.Usage Male
```

```
## 1                0.6178820 0.3809524   0.7304725            0.9160310    0
## 2                0.8096209 0.2857143   0.8313752            0.5387456    1
## 3                0.6267211 0.1666667   0.6992003            0.7974331    0
## 4                0.7062723 0.2380952   0.6231599            0.8542802    1
## 5                0.6080231 0.3809524   0.9145678            0.7313234    0
## 6                0.4655788 0.0952381   0.6988280            0.7383460    1
```

```
# Add the y variable back to the dataframe
df_norm$Clicked.on.Ad = advert_model$Clicked.on.Ad
head(df_norm)
```

```
##   Daily.Time.Spent.on.Site       Age Area.Income Daily.Internet.Usage Male
## 1                0.6178820 0.3809524   0.7304725            0.9160310    0
## 2                0.8096209 0.2857143   0.8313752            0.5387456    1
## 3                0.6267211 0.1666667   0.6992003            0.7974331    0
## 4                0.7062723 0.2380952   0.6231599            0.8542802    1
## 5                0.6080231 0.3809524   0.9145678            0.7313234    0
## 6                0.4655788 0.0952381   0.6988280            0.7383460    1
##   Clicked.on.Ad
## 1             0
## 2             0
## 3             0
## 4             0
## 5             0
## 6             0
```

```
# Next we split the data into training set and testing set.
intrain <- createDataPartition(y = df_norm$Clicked.on.Ad, p= 0.7, list = FALSE)
training <- df_norm[intrain,]
testing <- df_norm[-intrain,]
```

```
# Now we can use the K-NN algorithm. Lets call the "class" package which contains the K-NN algorithm.
# We then have to provide 'k' value which is no of nearest neighbours(NN) to look for
# in order to classify the test data point.
# Lets build a model on it; cl is the class of the training data set and k is the no of neighbours to l
# in order to classify it accordingly.

library(class)
require(class)
model <- knn(train= training,test=testing, ,cl= training$`Clicked.on.Ad`,k=5)
table(factor(model))
```

```
##
##   0   1
## 150 150
```

```
table(testing$`Clicked.on.Ad`,model)
```

```
##    model
##       0   1
##   0 150   0
##   1   0 150
```

The KNN model has predicted all the values correctly.

## Conclusion

Below are some of the conclusions we have:

1. Most of the individuals in the site are of the average age of 36.
2. Individuals between the age of 30 - 50 spend the most time on the site.
3. Individuals at the age of 31 are the most in the site.
4. KNN performed better with 100% accuracy while SVM had 96% Accuracy

## Recommendations

Our recommendations are:

1. More advertisement should cater to individuals in their 30s but extend to the age bracket (30-50).
2. We would recommend to use KNN because it has better performance