Author: Terry Migwi

12th September 2020

Week 13 IP - Supervised Learning with in R

1. Defining the question

Specifying the data analytic question

Use supervised machine learning algorithms to classify people who are most likely to click on ads on an online cryptography course advertised in a blog post.

Defining the metrics for success

This study will be considered successful if:

- Insights from the EDA are utilized
- We obtain an sccuracy score of 90% and above from the classifications

Understanding the context

Targetted advertising is customizing advertisements to reach a specific group of individuals who would be interested in a specific topic. Through targeted advertisement, advertisers can find and engage the most relevant audience by filtering internet users on the basis of their online behavior. Behavior targeting, or interest-based advertising, is the practice of analyzing and leveraging users' internet surfing patterns. In this way, marketers can deliver ads to only those internet users who have interests in similar entities.

Why is targetted Advertising important?

Targetted advertising is an advantage to both the advertiser and the audience. To the advertisers, it allows them to focus their time and resources to communicate the right information to the right people using the right platform and to the audience,it allows them to get the kind of information they would be interested in which would trigger them to click on the ads. Releasing ads that are not interesting to the audience is annoying and could lead the to blocking the ads or even leaving the blog site

What is cryptography?

Cryptography is associated with the process of converting ordinary plain text into unintelligible text and vice-versa. It is a method of storing and transmitting data in a particular form so that only those for whom it is intended can read and process it. Cryptography not only protects data from theft or alteration, but can also be used for user authentication. Earlier cryptography was effectively synonymous with encryption but nowadays cryptography is mainly based on mathematical theory and computer science practice.

What are advantages of cryptography?

Confidentiality - Information cannot be understood by anyone

Integrity - Information cannot be altered.

Non-repudiation - Sender cannot deny his/her intentions in the transmission of the information at a later stage

Authentication - Sender and receiver can confirm each

Where is cryptography applied

Cryptography is used in many applications like banking transactions cards, computer passwords, and e-commerce transactions.

It is therefore important to analyze the data to understand which individuals are more interested in this and similar topics hence leverage information obtained from this data to target like minded individuals. Targetted advertisement allows the advertiser to focus on sending the right information and the audience to focus on the right content.

Data Relevance

The data provided contains variables with personal details of the individuals e.g genderand age of the individuals, how often the individuals spend on the site and their daily internet usage and details of the advertisement e.g the ad line, whether the individuals clicked on the ads or not, when the individuals clicked on the ads e.t.c. I consider this data relevant for this study as it gives details of the individuals (our target audience), how much they are interested in the topic and geographic locations we should target to get more clicks.

Importing and loading the data

```
#importing required packages

#install.packages(data.table)
#install.packages("dplyr")
#install.packages("hablar")
```

```
#loading the packages required
library(data.table)
library(propagate)
```

```
## Loading required package: MASS

## Loading required package: tmvtnorm

## Loading required package: mvtnorm

## Loading required package: Matrix

## Loading required package: stats4

## Loading required package: gmm

## Loading required package: sandwich

## Loading required package: Rcpp

## Loading required package: ff

## Loading required package: bit
```

```
##
## Attaching package: 'bit'

## The following object is masked from 'package:data.table':
##
##     setattr

## The following object is masked from 'package:base':
##
##     xor

## Attaching package ff

## - getOption("fftempdir")=="C:/Users/USER/AppData/Local/Temp/RtmpKEbeJa/ff"

## - getOption("ffextension")=="ff"

## - getOption("ffdrop")==TRUE

## - getOption("fffinonexit")==TRUE

## - getOption("ffpagesize")==65536

## - getOption("ffcaching")=="mmnoflush"  -- consider "ffeachflush" if your system stalls on large write

## - getOption("ffbatchbytes")==84662026.24 -- consider a different value for tuning your system

## - getOption("ffmaxbytes")==4233101312 -- consider a different value for tuning your system

##
## Attaching package: 'ff'

## The following objects are masked from 'package:utils':
##
##     write.csv, write.csv2

## The following objects are masked from 'package:base':
##
##     is.factor, is.ordered

## Loading required package: minpack.lm
```

```
library(tidyr)
```

```
##
## Attaching package: 'tidyr'

## The following objects are masked from 'package:Matrix':
##
##     expand, pack, unpack
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following object is masked from 'package:propagate':
##
##     interval
```

```
## The following objects are masked from 'package:data.table':
##
##     hour, isoweek, mday, minute, month, quarter, second, wday, week,
##     yday, year
```

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

```
library(tidyverse)
```

```
## -- Attaching packages ---------------------------------------------------------------------------- tidy
```

```
## v ggplot2 3.3.2     v dplyr   1.0.2
## v tibble  3.0.3     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0
## v purrr   0.3.4
```

```
## -- Conflicts ------------------------------------------------------------------------------- tidyverse_
## x lubridate::as.difftime() masks base::as.difftime()
## x dplyr::between()         masks data.table::between()
## x lubridate::date()        masks base::date()
## x tidyr::expand()          masks Matrix::expand()
## x dplyr::filter()          masks stats::filter()
## x dplyr::first()           masks data.table::first()
## x lubridate::hour()        masks data.table::hour()
## x lubridate::intersect()   masks base::intersect()
## x lubridate::interval()    masks propagate::interval()
## x lubridate::isoweek()     masks data.table::isoweek()
## x dplyr::lag()             masks stats::lag()
## x dplyr::last()            masks data.table::last()
## x lubridate::mday()        masks data.table::mday()
## x lubridate::minute()      masks data.table::minute()
## x lubridate::month()       masks data.table::month()
## x tidyr::pack()            masks Matrix::pack()
## x lubridate::quarter()     masks data.table::quarter()
## x lubridate::second()      masks data.table::second()
## x dplyr::select()          masks MASS::select()
## x lubridate::setdiff()     masks base::setdiff()
## x purrr::transpose()       masks data.table::transpose()
## x lubridate::union()       masks base::union()
## x tidyr::unpack()          masks Matrix::unpack()
```

```
## x lubridate::wday()        masks data.table::wday()
## x lubridate::week()        masks data.table::week()
## x lubridate::yday()        masks data.table::yday()
## x lubridate::year()        masks data.table::year()
```

```
library(corrplot)
```

```
## corrplot 0.84 loaded
```

```
library(ggcorrplot)
library(psych)
```

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

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

## The following object is masked from 'package:propagate':
##
##     cor2cov

## The following object is masked from 'package:bit':
##
##     keysort
```

```
#library(data.table)
#loading the dataset

#advertising <- fread("C:\\Users\\USER\\Downloads\\IP's datasets\\Module 3 Core - R Programming\\advert

#previewing the top of the dataset
#head(advertising)


advertising <- read.csv("C:\\Users\\USER\\Downloads\\IP's datasets\\Module 3 Core - R Programming\\adve
head(advertising)
```

```
##   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 bottom of the dataset
tail(advertising)
```

```
##      Daily.Time.Spent.on.Site Age Area.Income Daily.Internet.Usage
## 995                     43.70  28    63126.96               173.01
## 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
## 995         Front-line bifurcated ability  Nicholasland    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
## 995                  Mayotte 2016-04-04 03:57:48             1
## 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
```

Checking the data

```
#checking the size of the dataset
dim(advertising)
```

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

The data has 1000 observations and 10 variables

```
# checking the structure of the dataset
str(advertising)
```

```
## '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               : chr  "2016-03-27 00:53:11" "2016-04-04 01:39:02" "2016-03-13 20:35:42"
##  $ Clicked.on.Ad           : int  0 0 0 0 0 0 0 1 0 0 ...
```

```
glimpse(advertising)
```

```
## Rows: 1,000
## Columns: 10
## $ Daily.Time.Spent.on.Site <dbl> 68.95, 80.23, 69.47, 74.15, 68.37, 59.99, ...
## $ Age                      <int> 35, 31, 26, 29, 35, 23, 33, 48, 30, 20, 49...
## $ Area.Income              <dbl> 61833.90, 68441.85, 59785.94, 54806.18, 73...
## $ Daily.Internet.Usage     <dbl> 256.09, 193.77, 236.50, 245.89, 225.58, 22...
## $ Ad.Topic.Line            <chr> "Cloned 5thgeneration orchestration", "Mon...
## $ City                     <chr> "Wrightburgh", "West Jodi", "Davidton", "W...
## $ Male                     <int> 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, ...
## $ Country                  <chr> "Tunisia", "Nauru", "San Marino", "Italy",...
## $ Timestamp                <chr> "2016-03-27 00:53:11", "2016-04-04 01:39:0...
## $ Clicked.on.Ad            <int> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, ...
```

Some of the variables e.g Timestamp have inappropriate datatypes

2.Data Preparation

Ensuring unformity

The variable names will be renamed to ensure the names are uniform

```
# get column names

colnames(advertising)
```

```
##  [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"
```

```
# changing column names

names(advertising)[1] <- "Daily_time_spent_on_site"
names(advertising)[3] <- "Area_income"
names(advertising)[4] <- "Daily_internet_usage"
names(advertising)[5] <- "Ad_topic_line"
names(advertising)[10] <- "Clicked_on_ad"

#confirming the variable names have been changed
colnames(advertising)
```

```
##  [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"
```

The naming of our variables is now uniform

```
#checking for unique values in the variables
lapply(advertising, function (x) {length(unique(x))})
```

```
## $Daily_time_spent_on_site
## [1] 900
##
## $Age
## [1] 43
##
## $Area_income
## [1] 1000
##
## $Daily_internet_usage
## [1] 966
##
## $Ad_topic_line
## [1] 1000
##
## $City
## [1] 969
##
## $Male
## [1] 2
##
## $Country
## [1] 237
##
## $Timestamp
## [1] 1000
##
## $Clicked_on_ad
## [1] 2
```

Male and Clicked_on_ads variables have 2 unique values hence they are factor variables.

Assigning Apropriate DataTypes

Variables Male and Clicked on ad are factors hence we will convert them. Timestamp variable is a datat time variable,

```
#changing variables to apprpriate data types
#converting male and clicked on ad variables to factor datatypes

#install.packages("dplyr")
#install.packages("hablar")
```

```
#library(dplyr)
#if (!require("pacman")) install.packages("pacman")
#pacman::p_load(tidyverse, hablar)

#advertising %>%
  #convert(fct(Male,
              #Clicked_on_ad))
```

```
#converting timestamp variable to datetime datatype

advertising[["Timestamp"]] <- as.POSIXct(advertising$Timestamp, tz=Sys.timezone())
str(advertising)
```

```
## '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 ...
```

```
glimpse(advertising)
```

```
## Rows: 1,000
## Columns: 10
## $ Daily_time_spent_on_site <dbl> 68.95, 80.23, 69.47, 74.15, 68.37, 59.99, ...
## $ Age                      <int> 35, 31, 26, 29, 35, 23, 33, 48, 30, 20, 49...
## $ Area_income              <dbl> 61833.90, 68441.85, 59785.94, 54806.18, 73...
## $ Daily_internet_usage     <dbl> 256.09, 193.77, 236.50, 245.89, 225.58, 22...
## $ Ad_topic_line            <chr> "Cloned 5thgeneration orchestration", "Mon...
## $ City                     <chr> "Wrightburgh", "West Jodi", "Davidton", "W...
## $ Male                     <int> 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, ...
## $ Country                  <chr> "Tunisia", "Nauru", "San Marino", "Italy",...
## $ Timestamp                <dttm> 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, 1, 0, 1, 0, ...
```

The datatype for timestamp has been changed to POSIXct. We now need to split the variable to two varibales of date and time

```
#splitting datetime into date and time

Time <- format(as.POSIXct(strptime(advertising$Timestamp,"%Y-%m-%d %H:%M:%S",tz="")) ,format = "%H:%M:%S
head(Time)
```

```
## [1] "00:53:11" "01:39:02" "20:35:42" "02:31:19" "03:36:18" "14:30:17"
```

```
Dates <- format(as.POSIXct(strptime(advertising$Timestamp,"%Y-%m-%d %H:%M:%S",tz="")) ,format = "%Y-%m-%
head(Dates)
```

```
## [1] "2016-03-27" "2016-04-04" "2016-03-13" "2016-01-10" "2016-06-03"
## [6] "2016-05-19"
```

```
advertising$Dates <- Dates
    advertising$Time <- Time

str(advertising)
```

```
## 'data.frame':    1000 obs. of  12 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 ...
##  $ Dates                   : chr  "2016-03-27" "2016-04-04" "2016-03-13" "2016-01-10" ...
##  $ Time                    : chr  "00:53:11" "01:39:02" "20:35:42" "02:31:19" ...
```

We now have the two variables of date and time split. We now separate the two variables date and time to individual variables of year, month, day, hour, minutes and seconds

```
#separating dates to hours minutes and days
#advertising$Dates = as.Date(advertising$Dates)

#library(tidyr)
#library(lubridate)

advertising <- separate(advertising, "Dates", c("Year", "Month", "Day"), sep = "-")

advertising <- separate(advertising, "Time", c("Hour", "Minutes", "Seconds"), sep = ":")

colnames(advertising)
```

```
##  [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"
## [11] "Year"                     "Month"
## [13] "Day"                      "Hour"
## [15] "Minutes"                  "Seconds"
```

```
#previewing our new dataset
head(advertising)
```

```
##   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 Year Month Day Hour Minutes Seconds
## 1 2016-03-27 00:53:11             0 2016    03  27   00      53      11
## 2 2016-04-04 01:39:02             0 2016    04  04   01      39      02
## 3 2016-03-13 20:35:42             0 2016    03  13   20      35      42
## 4 2016-01-10 02:31:19             0 2016    01  10   02      31      19
## 5 2016-06-03 03:36:18             0 2016    06  03   03      36      18
## 6 2016-05-19 14:30:17             0 2016    05  19   14      30      17
```

Our dataset now contains separate dates and time values we can analyze differently. We will now assign appropriate data types to our newly derived columns

```
#converting the new derived columns to factor variables

#advertising %>%
  #convert(fct(Year, Month, Day,
              #Hour, Minutes, Seconds))

advertising$Male = factor(advertising$Male)
advertising$Year = factor(advertising$Year)
advertising$Month = factor(advertising$Month)
advertising$Day = factor(advertising$Day)
advertising$Hour = factor(advertising$Hour)
advertising$Minutes = factor(advertising$Minutes)
advertising$Seconds = factor(advertising$Seconds)

#viewing the structure of our new dataframe

str(advertising)
```

```
## 'data.frame':    1000 obs. of  16 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                    : Factor w/ 2 levels "0","1": 1 2 1 2 1 2 1 2 2 2 ...
##  $ 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 ...
##  $ Year                    : Factor w/ 1 level "2016": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Month                   : Factor w/ 7 levels "01","02","03",..: 3 4 3 1 6 5 1 3 4 7 ...
##  $ Day                     : Factor w/ 31 levels "01","02","03",..: 27 4 13 10 3 19 28 7 18 11 ...
##  $ Hour                    : Factor w/ 24 levels "00","01","02",..: 1 2 21 3 4 15 21 2 10 2 ...
##  $ Minutes                 : Factor w/ 60 levels "00","01","02",..: 54 40 36 32 37 31 60 41 34 43 ..
##  $ Seconds                 : Factor w/ 60 levels "00","01","02",..: 12 3 43 20 19 18 33 16 43 52 ...
```

We now have our dataset with appropriate data types ready for cleaning and analysis

Completeness

Checking for missing values

```
#checking for missing values

colSums(is.na(advertising))
```

```
## 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                     Year                    Month
##                        0                        0                        0
##                      Day                     Hour                  Minutes
##                        0                        0                        0
##                  Seconds
##                        0
```

There are no missing values in our dataset hence the data is complete

Consistency

Checking for duplicate records

```
#checking for duplicate values
duplicates = advertising[duplicated(advertising),]
duplicates
```

```
##  [1] Daily_time_spent_on_site Age                      Area_income
##  [4] Daily_internet_usage     Ad_topic_line            City
##  [7] Male                     Country                  Timestamp
## [10] Clicked_on_ad            Year                     Month
## [13] Day                      Hour                     Minutes
## [16] Seconds
## <0 rows> (or 0-length row.names)
```

There are no duplicate records in our dataset hence our data in consistent

Checking for anomalies

Checking for outliers in the numerical variables

```r
#checking the structure of the dataset to identify numerical variables
str(advertising)
```

```
## 'data.frame':    1000 obs. of  16 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                    : Factor w/ 2 levels "0","1": 1 2 1 2 1 2 1 2 2 2 ...
##  $ 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 ...
##  $ Year                    : Factor w/ 1 level "2016": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Month                   : Factor w/ 7 levels "01","02","03",..: 3 4 3 1 6 5 1 3 4 7 ...
##  $ Day                     : Factor w/ 31 levels "01","02","03",..: 27 4 13 10 3 19 28 7 18 11 ...
##  $ Hour                    : Factor w/ 24 levels "00","01","02",..: 1 2 21 3 4 15 21 2 10 2 ...
##  $ Minutes                 : Factor w/ 60 levels "00","01","02",..: 54 40 36 32 37 31 60 41 34 43 ..
##  $ Seconds                 : Factor w/ 60 levels "00","01","02",..: 12 3 43 20 19 18 33 16 43 52 ...
```

The numerical variables are: daily_time_spent_on_site, area income, age, and daily_internet usage

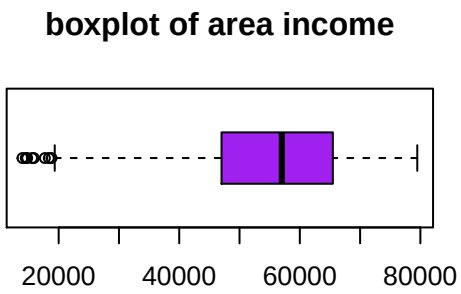
Boxplots

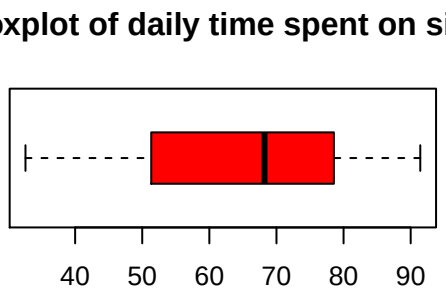
Checking for outlier values in the numerical variables

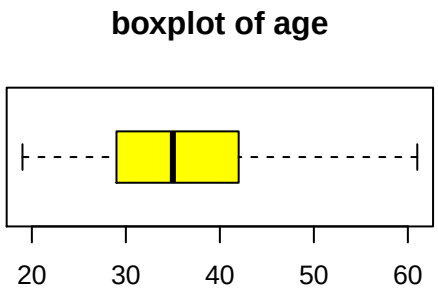
```r
#finding out the length of our dataset
length(advertising)
```

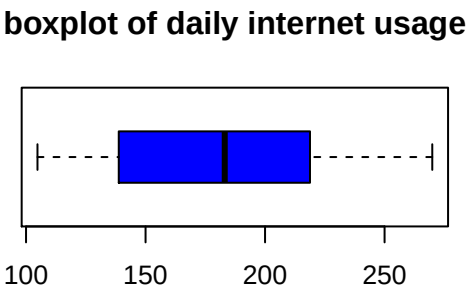
```
## [1] 16
```

```r
#plotting boxplots for all the numerical variables
par(mfrow=c(2,2))
boxplot((advertising$`Area_income`), horizontal = TRUE, col = 'purple', main = "boxplot of area income"
boxplot((advertising$`Daily_time_spent_on_site`), horizontal = TRUE, col = 'red', main = "boxplot of da
boxplot((advertising$'Age'), horizontal = TRUE, col = 'yellow', main = "boxplot of age")
boxplot((advertising$`Daily_internet_usage`), horizontal = TRUE, col = 'blue', main = "boxplot of daily
```









Observations 1. Area income variable has values ranging from below 0 to 80,000. We have a few values below 20,000 which are outliers, we will keep these outliers 2. Daily time spent on site has values from around 20 to 90 3. Age variable has observations from the age of 20 to 60 4. Daily internet usage has values from 100 to slightly above 250

3.Univariate Analysis

```
#printing out the summary of our dataset
summary(advertising)
```

```
##  Daily_time_spent_on_site      Age         Area_income    Daily_internet_usage
##  Min.   :32.60            Min.   :19.00   Min.   :13996   Min.   :104.8
##  1st Qu.:51.36            1st Qu.:29.00   1st Qu.:47032   1st Qu.:138.8
##  Median :68.22            Median :35.00   Median :57012   Median :183.1
##  Mean   :65.00            Mean   :36.01   Mean   :55000   Mean   :180.0
##  3rd Qu.:78.55            3rd Qu.:42.00   3rd Qu.:65471   3rd Qu.:218.8
##  Max.   :91.43            Max.   :61.00   Max.   :79485   Max.   :270.0
##
##  Ad_topic_line          City           Male      Country
##  Length:1000        Length:1000        0:519   Length:1000
##  Class :character   Class :character   1:481   Class :character
##  Mode  :character   Mode  :character           Mode  :character
##
##
```

```
##
##
##    Timestamp                     Clicked_on_ad   Year      Month         Day
##  Min.   :2016-01-01 02:52:10   Min.   :0.0   2016:1000   01:147   03     : 46
##  1st Qu.:2016-02-18 02:55:42   1st Qu.:0.0               02:160   17     : 42
##  Median :2016-04-07 17:27:29   Median :0.5               03:156   15     : 41
##  Mean   :2016-04-10 10:34:06   Mean   :0.5               04:147   10     : 37
##  3rd Qu.:2016-05-31 03:18:14   3rd Qu.:1.0               05:147   04     : 36
##  Max.   :2016-07-24 00:22:16   Max.   :1.0               06:142   26     : 36
##                                                          07:101   (Other):762
##       Hour        Minutes       Seconds
##  07     : 54   02     : 26   22     : 28
##  20     : 50   07     : 24   10     : 27
##  09     : 49   13     : 24   35     : 27
##  21     : 48   10     : 22   37     : 27
##  00     : 45   21     : 21   38     : 24
##  05     : 44   33     : 21   15     : 23
##  (Other):710   (Other):862   (Other):844
```

Observations

1. The minimum daily time spent on site is 32.60 while the maximum time spent on site is 91.43. The average time spent on site is 65. We are assuming this is in seconds
2. The minimum age recorded is 19 while the maximum age recorded is 61. The mean is 36 while the median is 35
3. The minimum area income is approximately 14, 000 while the maximum area income is approximately 80,000. The mean value is 55,000. This could mean that the data was mostly collected from people at entry and mid levels of employment.
4. The minimum daily internet usage is 104 while the maximum is 270.
5. We only have data of only one year, 2016.
6. We have more females than males

Measures of Central Tendency

Mode

```
# creating a mode function

getmode<- function(v){
  uniqv<-unique(v)
  uniqv[which.max(tabulate(match(v, uniqv)))]
}

#calling the function on all character variables
mode.city <- getmode(advertising$City) #mode of the city variable
print(mode.city)
```

```
## [1] "Lisamouth"
```

```
mode.country <- getmode(advertising$Country) #mode of the country variable
print(mode.country)
```

```
## [1] "Czech Republic"
```

```r
mode.adline<-getmode(advertising$Ad_topic_line) #mode of the advertisement topic line
print(mode.adline)
```

```
## [1] "Cloned 5thgeneration orchestration"
```

The most frequent city is Lisamouth, the most frequent country is Czech Republic, The most frequent ad line is Cloned 5thgeneration orchestration

```r
#calling the function on all the factor variables

mode.male <- getmode(advertising$Male)
print(mode.male)
```

```
## [1] 0
## Levels: 0 1
```

```r
mode.month <- getmode(advertising$Month) # mode value of the month
print(mode.month)
```

```
## [1] 02
## Levels: 01 02 03 04 05 06 07
```

```r
mode.day<-getmode(advertising$day)  #mode value of the day
print(mode.day)
```

```
## NULL
```

```r
mode.hour<- getmode(advertising$Hour) #mode value of hour
print(mode.hour)
```

```
## [1] 07
## 24 Levels: 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 ... 23
```

```r
mode.minutes<- getmode(advertising$Minutes) #mode value of minutes
print(mode.minutes)
```

```
## [1] 02
## 60 Levels: 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 ... 59
```

```r
mode.second<- getmode(advertising$Seconds) #mode value of seconds
print(mode.second)
```

```
## [1] 22
## 60 Levels: 00 01 02 03 04 05 06 07 08 09 10 11 12 13 14 15 16 17 18 19 ... 59
```

The factor variables are all equally represented

```r
#calling the mode function for all the numerical variables

mode.age <- getmode(advertising$Age) #mode of the age variable
print(mode.age)
```

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

```r
mode.time <- getmode(advertising$Daily_time_spent_on_site) #mode of the daily time spent on site
print(mode.time)
```

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

```r
mode.usage <- getmode(advertising$Daily_internet_usage)  #mode of the daily internet usage
print(mode.usage)
```

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

```r
mode.income <- getmode(advertising$Area_income)  #mode of the area income
print(mode.income)
```

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

The mode value of the age variable is 31

Mean and Median

Finding the mean and median values of the numerical variables

```r
#mean values of the numerical variables
mean.age<- mean(advertising$Age)
mean.time<- mean(advertising$Daily_time_spent_on_site)
mean.usage<- mean(advertising$Daily_internet_usage)
mean.income<- mean(advertising$Area_income)

#median values of the numerical variables
median.age<- median(advertising$Age)
median.time<- median(advertising$Daily_time_spent_on_site)
median.usage<- median(advertising$Daily_internet_usage)
median.income<- median(advertising$Area_income)

print(mean.age)
```

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

```r
print(median.age)
```

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

```
print("---------------")
```

```
## [1] "---------------"
```

```
print(mean.time)
```

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

```
print(median.time)
```

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

```
print("---------------")
```

```
## [1] "---------------"
```

```
print(mean.usage)
```

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

```
print(median.usage)
```

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

```
print("---------------")
```

```
## [1] "---------------"
```

```
print(mean.income)
```

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

```
print(median.income)
```

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

Measures of spread

```
#library(propagate)
```

```
#variance, std, skewness and kurtosis of age variable
print(sd(advertising$Age))
```

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

```r
print(var(advertising$Age))
```

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

```r
print(range(advertising$Age))
```

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

```r
print(skewness(advertising$Age))
```

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

```r
print(kurtosis(advertising$Age))
```

```
## [1] -0.4045182
```

```r
print("----------------------------")
```

```
## [1] "----------------------------"
```

```r
#variance, std, skewness and kurtosis of area income variable
print(sd(advertising$Area_income))
```

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

```r
print(var(advertising$Area_income))
```

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

```r
print(range(advertising$Area_income))
```

```
## [1] 13996.5 79484.8
```

```r
print(skewness(advertising$Area_income))
```

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

```r
print(kurtosis(advertising$Area_income))
```

```
## [1] -0.1053059
```

```r
print("-------------------------------")
```

```
## [1] "-------------------------------"
```

```r
#variance, std, range, skewness and kurtosis of daily internet usage variable
print(sd(advertising$Daily_internet_usage))
```

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

```r
print(var(advertising$Daily_internet_usage))
```

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

```r
print(range(advertising$Daily_internet_usage))
```

```
## [1] 104.78 269.96
```

```r
print(skewness(advertising$Daily_internet_usage))
```

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

```r
print(kurtosis(advertising$Daily_internet_usage))
```

```
## [1] -1.272299
```

```r
print("-----------------------------------------------")
```

```
## [1] "-----------------------------------------------"
```

```r
#variance, std, range, skewness and kurtosis of daily time spent on site  variable
print(sd(advertising$Daily_time_spent_on_site))
```

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

```r
print(var(advertising$Daily_time_spent_on_site))
```

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

```r
print(range(advertising$Daily_time_spent_on_site))
```

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

```r
print(skewness(advertising$Daily_time_spent_on_site))
```

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

```r
print(kurtosis(advertising$Daily_time_spent_on_site))
```

```
## [1] -1.096058
```

```
print("-------------------------------------------------")
```

```
## [1] "-------------------------------------------------"
```

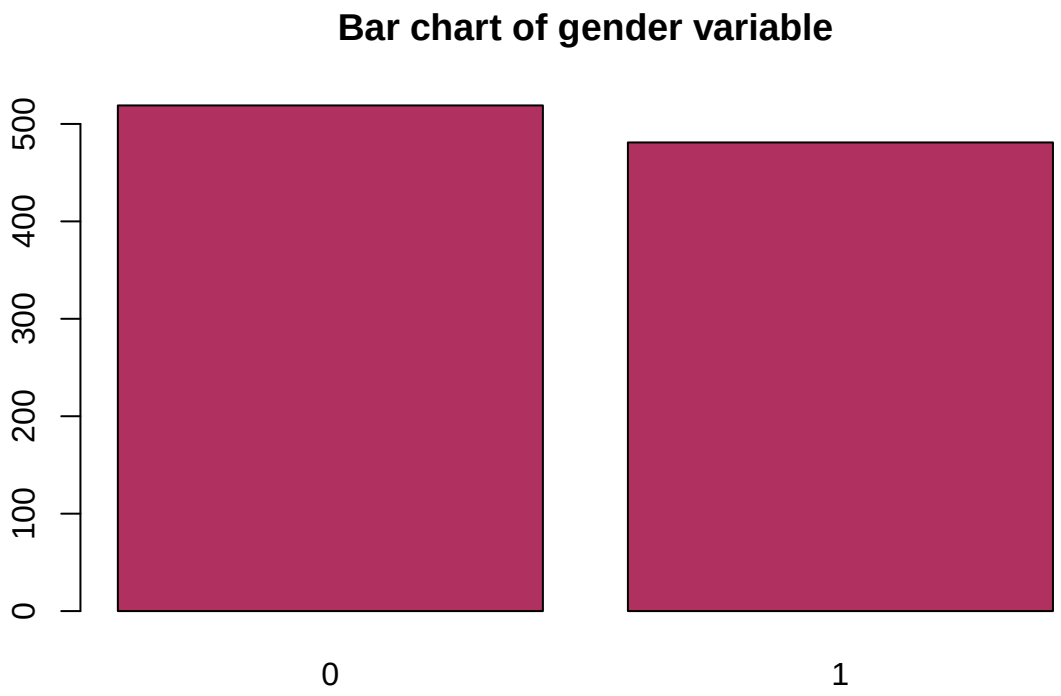
Observations

1. The Age variable has a negative kurtosis implying it is platykurtic i.e it has very thin tails, and it is also positively skewed.
2. The Area income variable has such a high variance with a negative kurtosis meaning it is platykurtic and negatively skewed.
3. The daily internet usage variable has an almost normal distribution with thin tails
4. The daily time spent on site variable is almost normally distributed with also thin tails

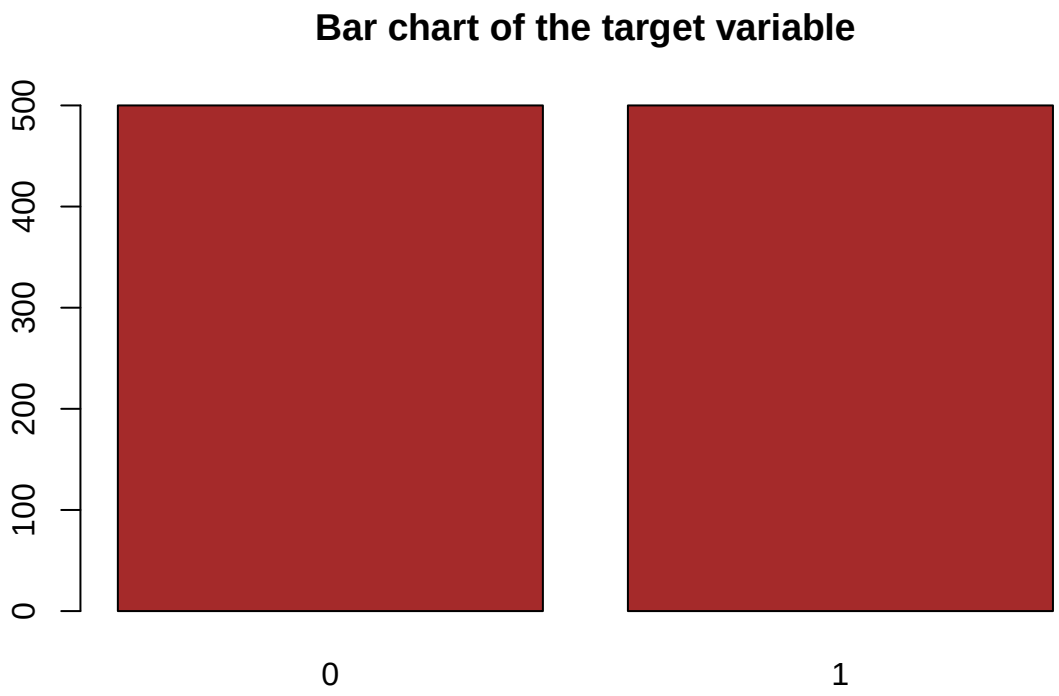
Bar plots

```
#loading the tidyverse library for using ggplots
#library(tidyverse)
```

```
#plotting barcharts of categorical variables

gender<-advertising$Male          #locating the male variable
gender_frequency<- table(gender)  #assigning a frequency table to the variable

target<-advertising$Clicked_on_ad #fetching the clicked on ad variable
target_frequency<- table(target)  #assigning a frequency table to the variable

#plotting bar charts using the frequency tables
par(mfrow=c(1,1))
barplot((gender_frequency), col = "maroon", main = "Bar chart of gender variable")
```




```
barplot((target_frequency), col = "brown", main = "Bar chart of the target variable")
```

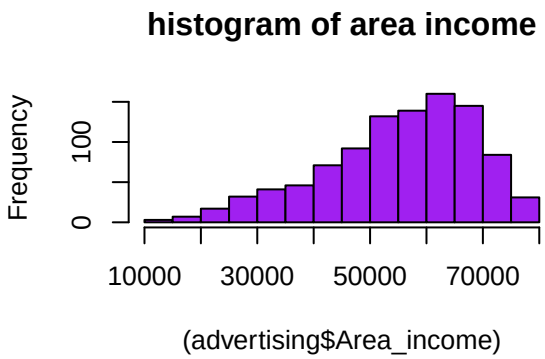


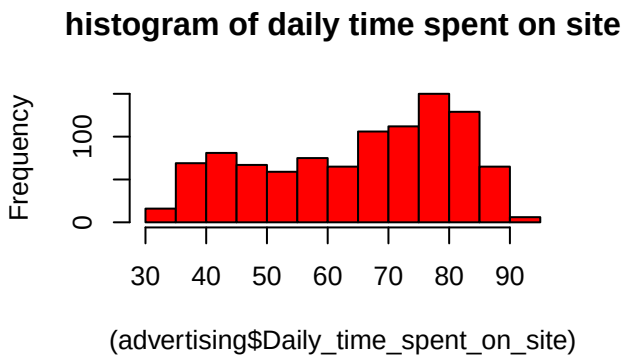
Observations

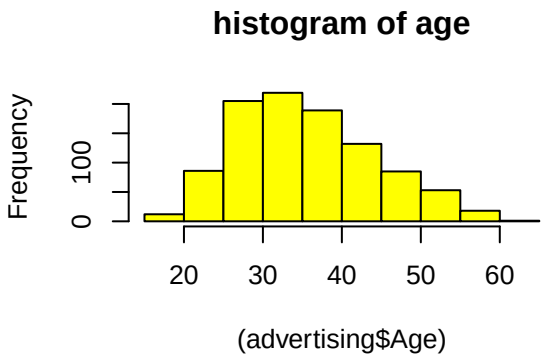
1. The number of females is slightly higher than the number of males
2. The number of people who clicked on the ads is not too far from those who didn't click the ads

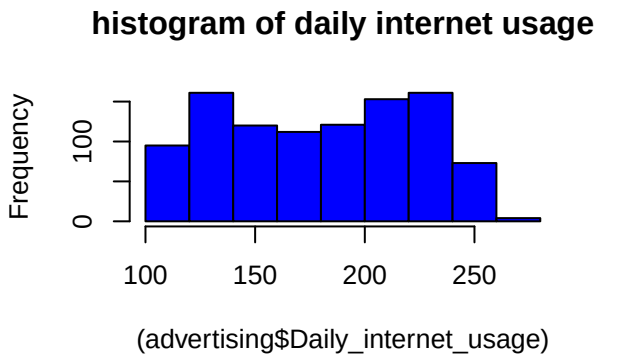
Hisograms of the numerical variables

```
#plotting multiple histograms
par(mfrow=c(2,2))
#histogram of area income
hist((advertising$`Area_income`), col = 'purple', main = "histogram of area income")

#histogram of daily time spent on site
hist((advertising$`Daily_time_spent_on_site`), col = 'red', main = "histogram of daily time spent on si

#histogram of age
hist((advertising$'Age'), col = 'yellow', main = "histogram of age")

#histogram of daily internet usage
hist((advertising$`Daily_internet_usage`), col = 'blue', main = "histogram of daily internet usage")
```









Observations

1. Area income variable is negatively skewed, most of the observations recorded being lower as compared to the high area income
2. Age variable is positively skewed, most of the ages recorded are older
3. Daily internet usage and daily time spent on site are bimodal

3.Bivariate Analysis

Correlation Co-efficient and Covariance

```
#finding out how the age of an individual relates to how much time is spent on the site

print(cov(advertising$Daily_time_spent_on_site, advertising$Age))   #printing covariance
```

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

```
print(cor(advertising$Daily_time_spent_on_site, advertising$Age))   #printing correlation
```

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

There is a weak negative correlation between the age of an individual and how much time the individual spends on the site

```
#finding out the relationship between daily internet usage and time spent on site

print(cov(advertising$Daily_time_spent_on_site, advertising$Daily_internet_usage))  #printing covarianc
```

```
## [1] 360.9919
```

```
print(cor(advertising$Daily_time_spent_on_site, advertising$Daily_internet_usage))  #printing correlati
```

```
## [1] 0.5186585
```

There is a moderate positive relationship between an individual's daily internet usage and how much time the individual spends on site

```
#find out the relationship between area income and daily internet usage

print(cov(advertising$Area_income, advertising$Daily_internet_usage))   #printing covariance
```

```
## [1] 198762.5
```

```
print(cor(advertising$Area_income, advertising$Daily_internet_usage))   #printing correlation
```

```
## [1] 0.3374955
```

There is a weak positive relationship between area income and daily internet usage

```
#find out the relationship between area income and daily internet usage

print(cov(advertising$Area_income, advertising$Age))
```

```
## [1] -21520.93
```

```
print(cor(advertising$Area_income, advertising$Age))
```

```
## [1] -0.182605
```

There is a weak negative relationship between area income and age

Correlation Matrix with ggplots

```
#library(ggcorrplot)
```

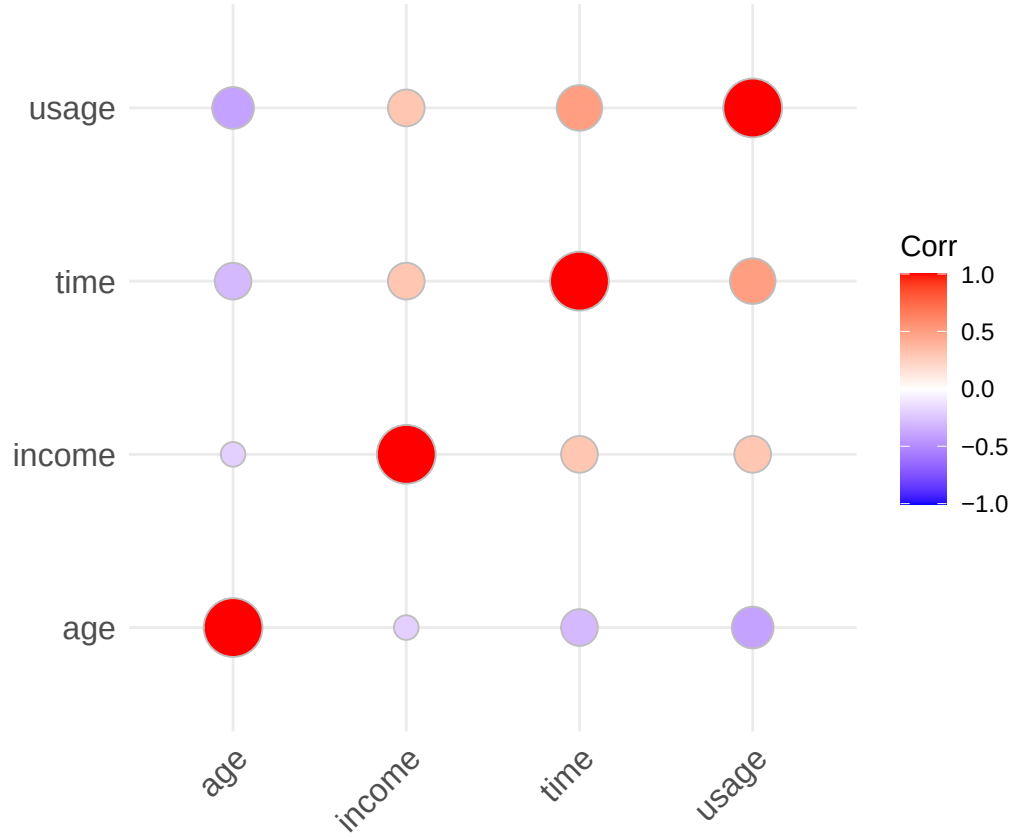
```
#fetching all the numerical variables from the advertising dataset

age<- advertising$Age   #fetching the age variable
income<-advertising$Area_income #fetching the income variable
time<-advertising$Daily_time_spent_on_site  #fetching the daily time spent on site
usage<-advertising$Daily_internet_usage    #fetching daily internet usage

#creating a new dataframe num with numerical variables
num<- data.frame(age, income, time, usage)
head(num)    #previewing the dataframe
```

```
##   age   income  time  usage
## 1  35 61833.90 68.95 256.09
## 2  31 68441.85 80.23 193.77
## 3  26 59785.94 69.47 236.50
## 4  29 54806.18 74.15 245.89
## 5  35 73889.99 68.37 225.58
## 6  23 59761.56 59.99 226.74
```

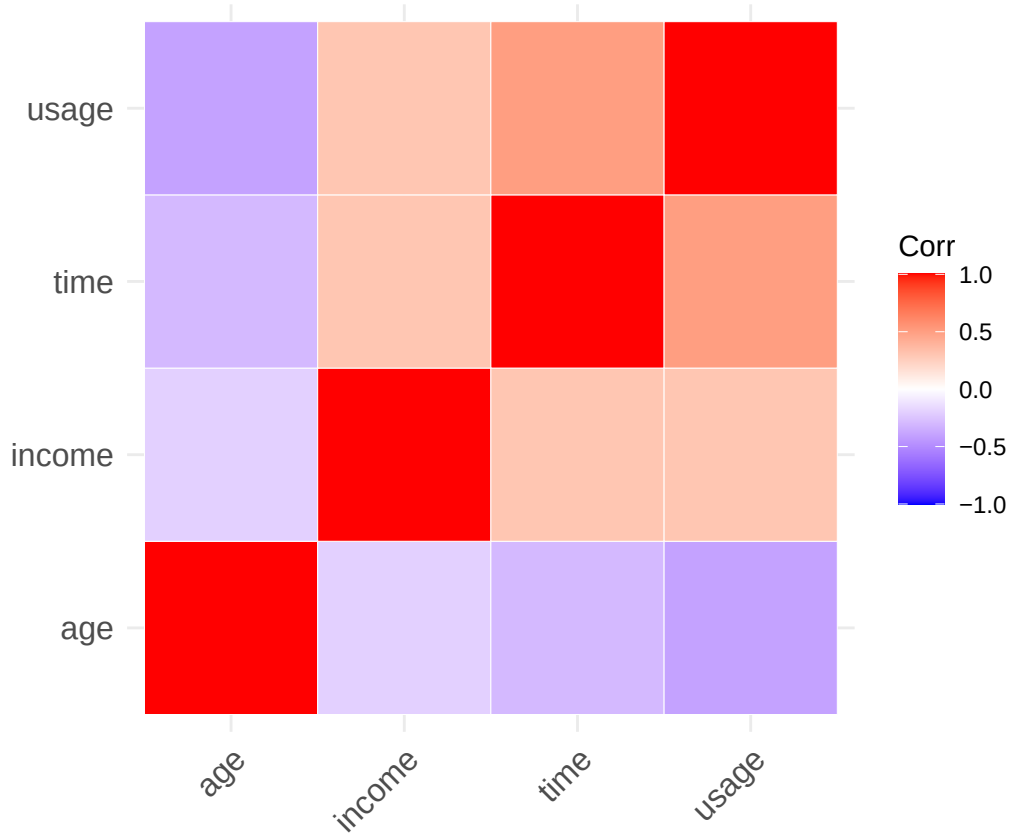
```
#calculating a correlation matrix of the dataframe created

corr <- round(cor(num), 1)
head(corr[, 1:4])   #previewing the matrix
```

```
##         age income time usage
## age     1.0   -0.2 -0.3  -0.4
## income -0.2    1.0  0.3   0.3
## time   -0.3    0.3  1.0   0.5
## usage  -0.4    0.3  0.5   1.0
```

```
#visualizing correlation matrix

ggcorrplot(corr, method = "circle")
```



```r
# Reordering the correlation matrix
# --------------------------------
# using hierarchical clustering
ggcorrplot(corr, hc.order = TRUE, outline.col = "white")
```

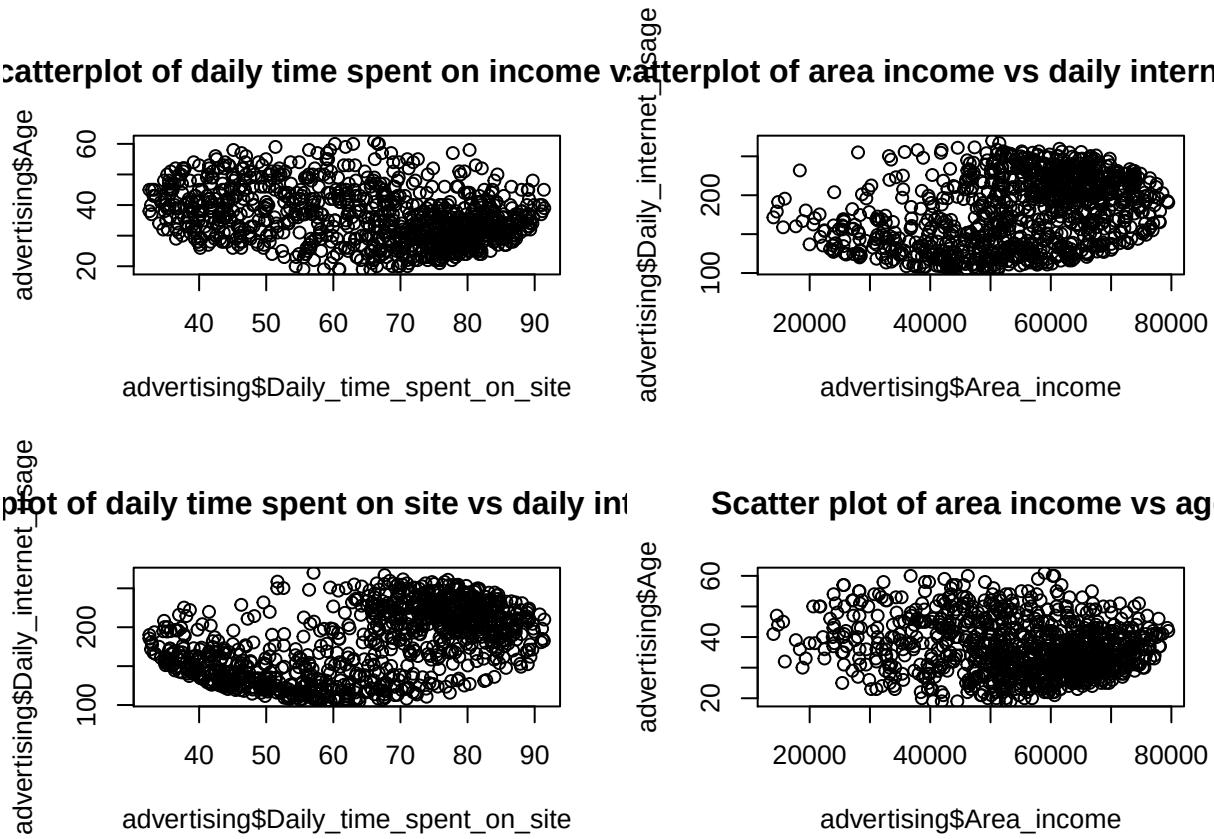


Observations

1. Variables usage(daily_internet_usage) and time(daily_time_spent_on_site) seem to have a moderate positive correlation
2. Variables usage and age seem to have a moderate negative correlation
3. Variables income and are weakly correlated

Scatter plots

```r
#scatter plots to show the relationship above

par(mfrow=c(2,2))
plot(advertising$Daily_time_spent_on_site,advertising$Age, main="Scatterplot of daily time spent on inc
plot(advertising$Area_income,advertising$Daily_internet_usage, main="Scatterplot of area income vs daily
plot(advertising$Daily_time_spent_on_site, advertising$Daily_internet_usage, main="Scatter plot of daily
plot(advertising$Area_income, advertising$Age,main="Scatter plot of area income vs age")
```



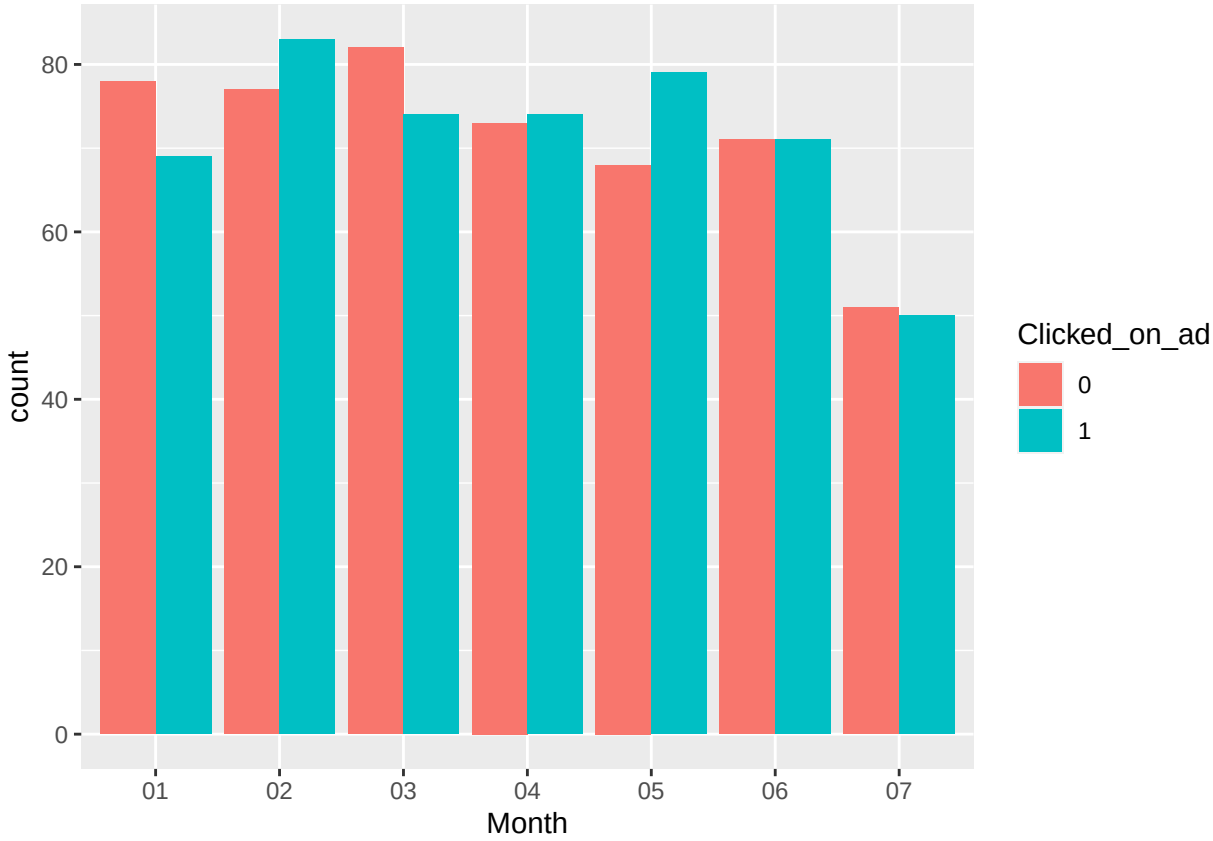
Observations

From the plots above, we do NOT observe any linear relationship between the variables

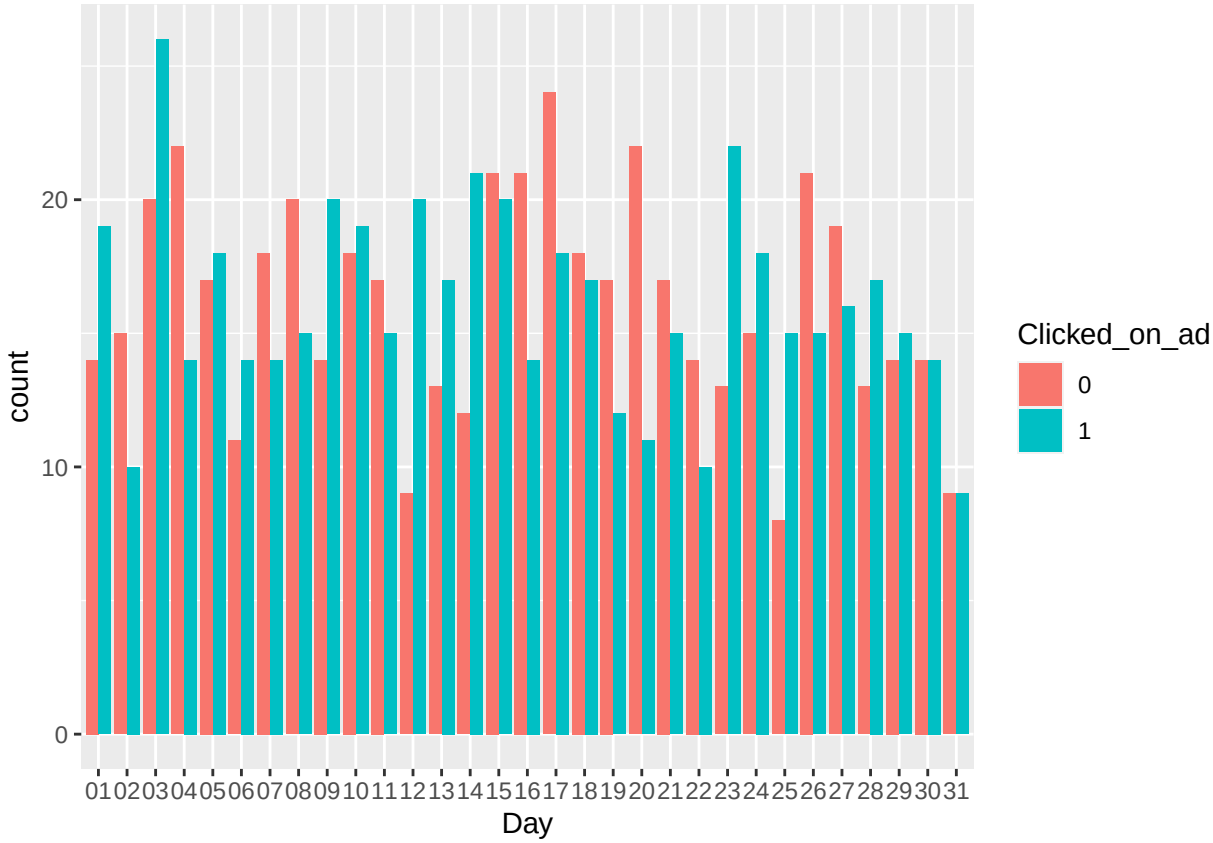
Using ggplots - barplots

We would like to find out when is the most popular time ads are clicked on i.e in month, day and hours

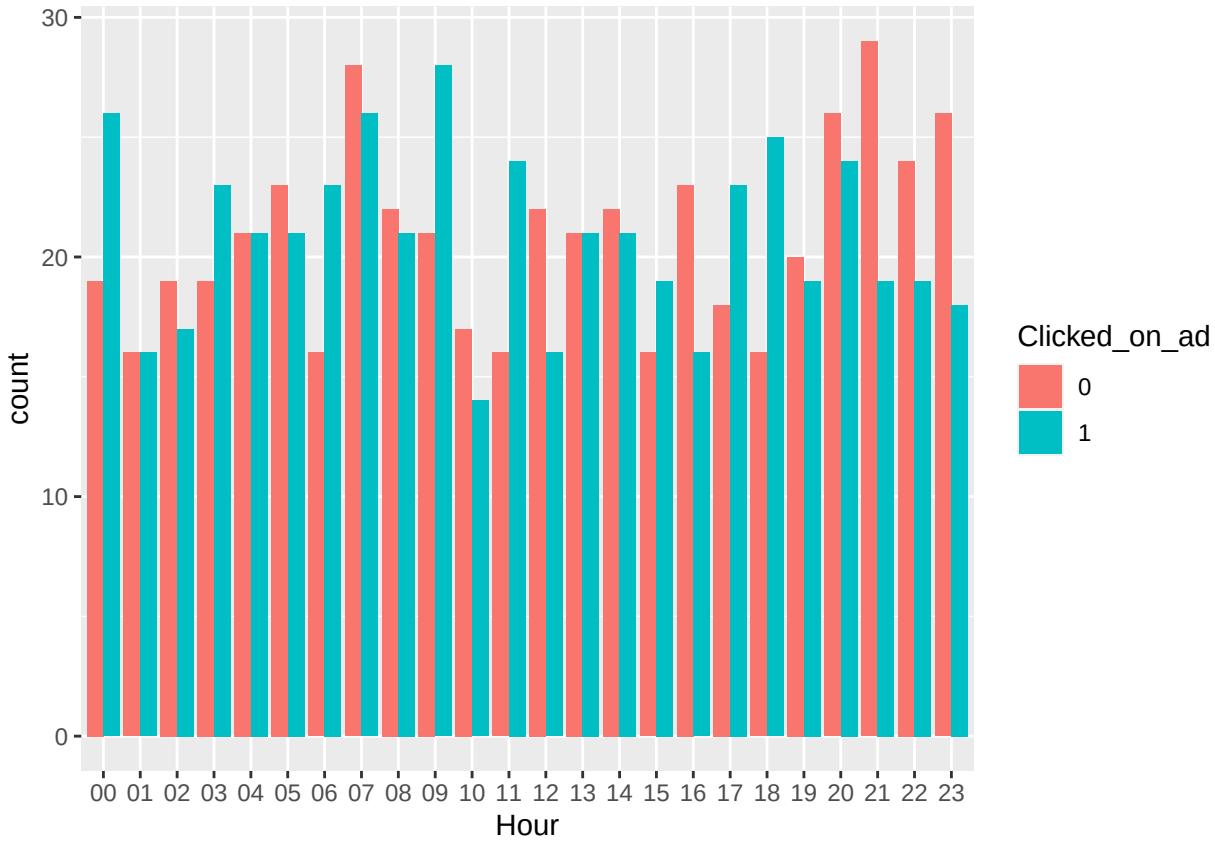
```
#converting the target variable, clicked on ads, to a factor variable
advertising$Clicked_on_ad = factor(advertising$Clicked_on_ad)

#barplot showing count of monthly clicks on the ads
ggplot(data = advertising) +
  geom_bar(mapping = aes(x = Month, fill = Clicked_on_ad), position = "dodge")
```




```
#barplot showing count of daily clicks on the ads
ggplot(data = advertising) +
  geom_bar(mapping = aes(x = Day, fill = Clicked_on_ad), position = "dodge")
```




```
#barplot showing count of hourly clicks
ggplot(data = advertising) +
  geom_bar(mapping = aes(x = Hour, fill = Clicked_on_ad), position = "dodge")
```

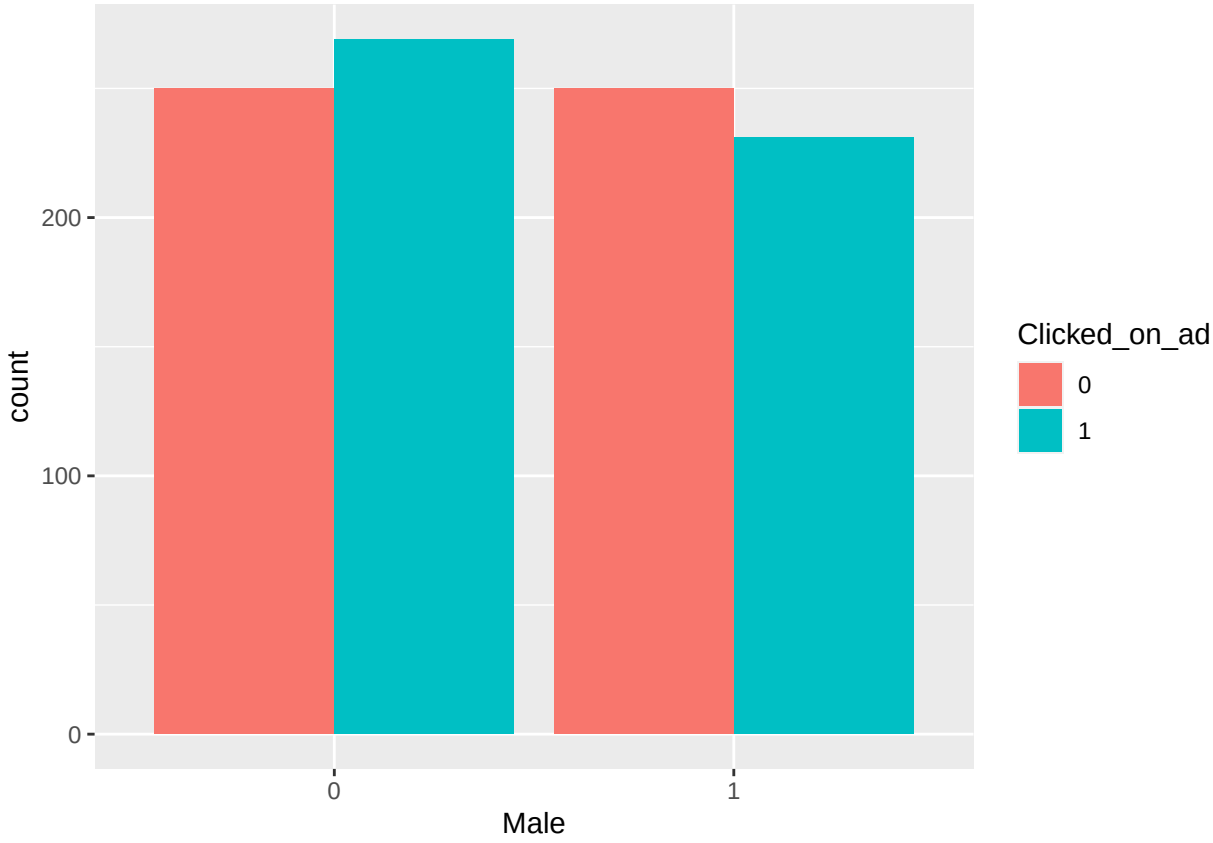



```
#barplot showing count of clicks per minute
#ggplot(data = advertising) +
  #geom_bar(mapping = aes(x = Minutes, fill = Clicked_on_ad), position = "dodge")+
  #coord_flip()
```

Observations

1 Months 2 and 3 had the most activity recorded, from both who clicked the ads and those who did NOt. 2. Months 1, 3 and 7 had more activity from those who did not click on the ads as compared to those who clicked on the ads. 3. Months 2,5 and 4 had more people who clicked on the ads as compared to those who did not clean on the ads 4. Month 6 had an equal number of people who clicked on the ads and those who did Not 5. We observe that at around mid month we had more people who were not clicking on the ads as compared to the beginning and the end of the month 6. From around 7pm to 11pm, we have more people not clicking on ads as compared to those who clicked on the ads 7. Midnight, 3am, 6am and 9am, 11am, 3pm, 5pm and 6pm are the hours with the most clicks on the ads.

```
#converting the target variable, clicked on ads, to a factor variable
advertising$Clicked_on_ad = factor(advertising$Clicked_on_ad)

#finding out which gender is more likely to click on the ads
ggplot(data = advertising) +
  geom_bar(mapping = aes(x = Male, fill = Clicked_on_ad), position = "dodge")
```



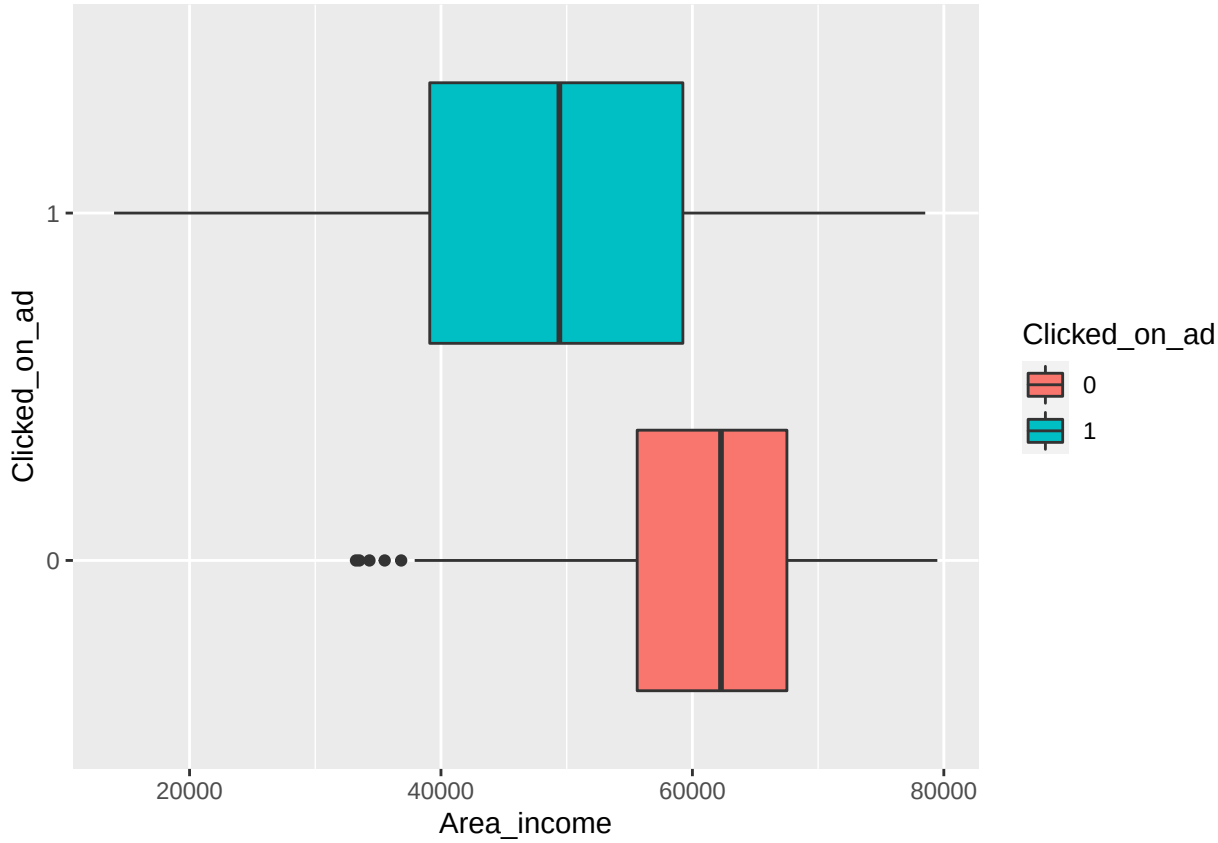
Observations

We have more number of females who clicked on the ads as compared to those who did not. Most males did not click on the ads.

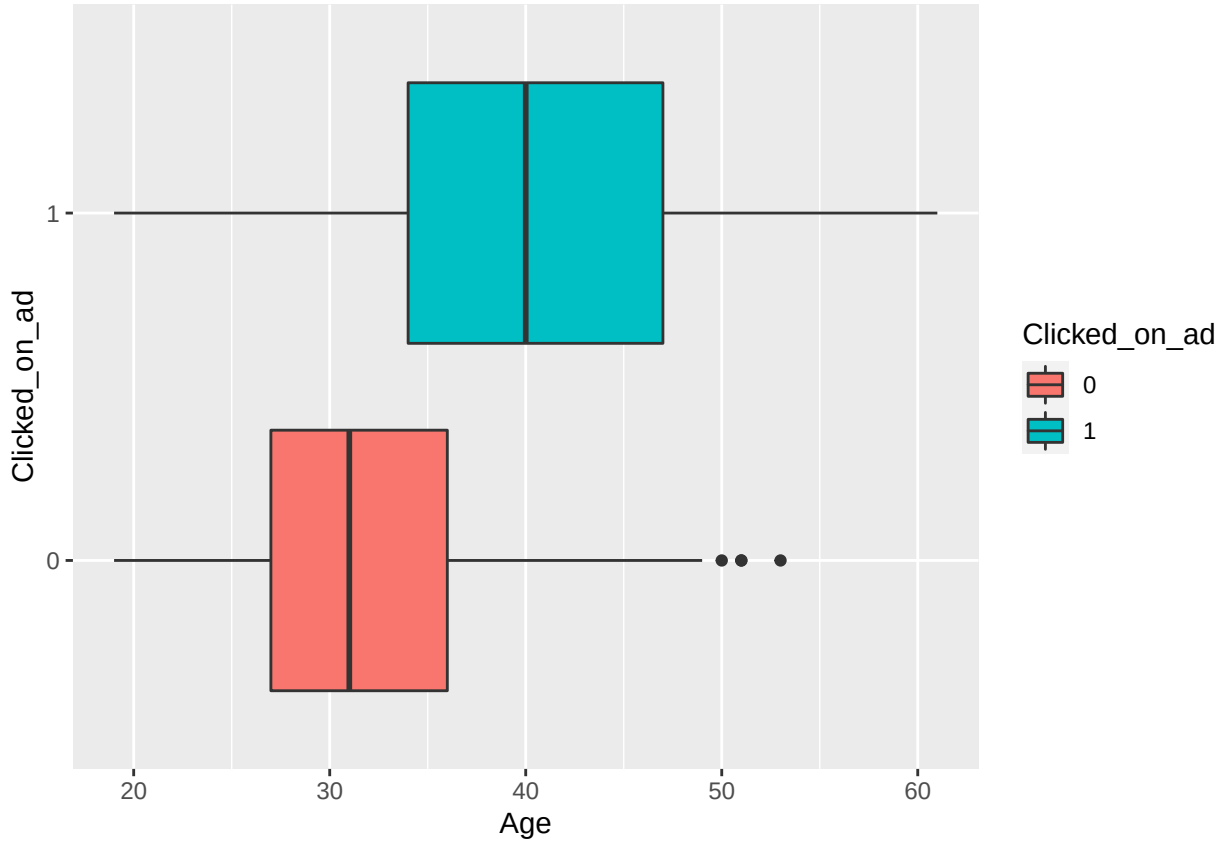
Using ggplots-boxplots

We would like to investigate how the numerical variables influence the target variable

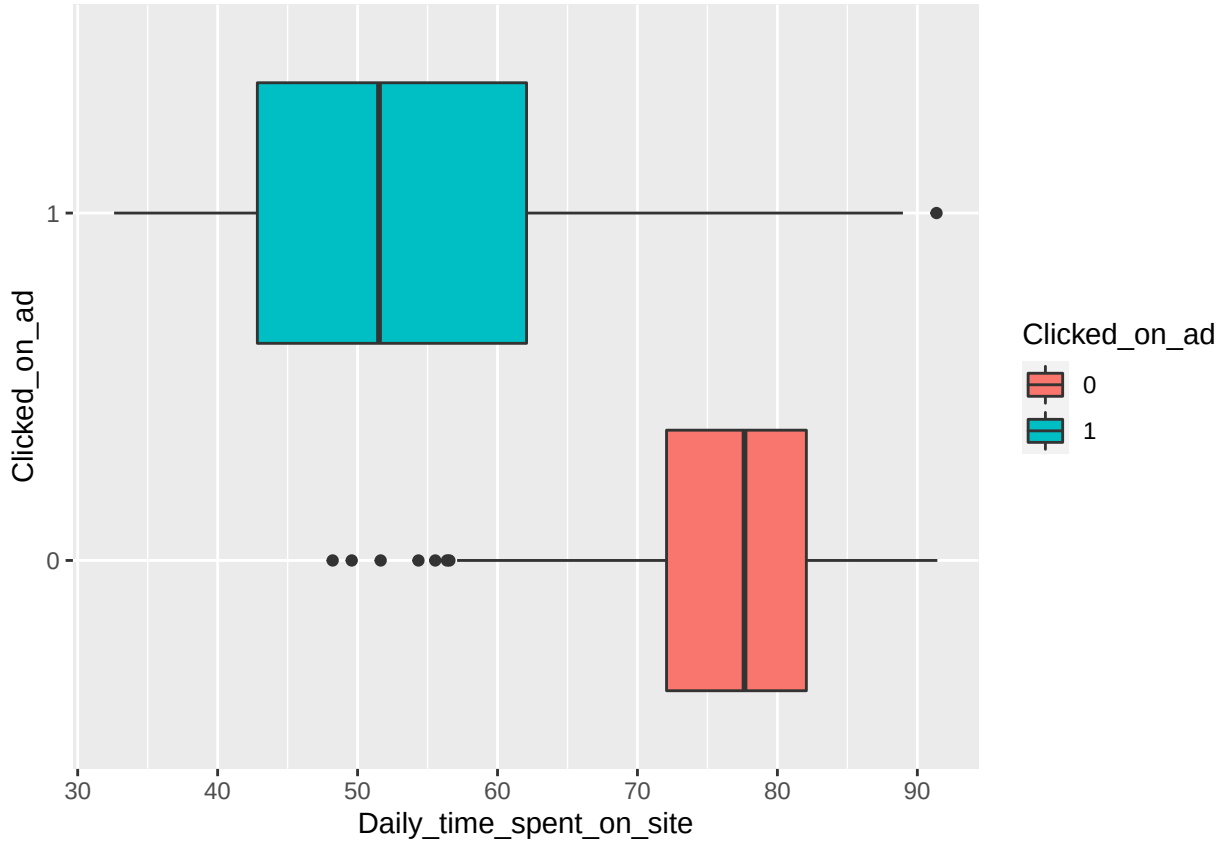
```
#plotting boxplots to show how the area income relates with the number of clicks
ggplot(data = advertising, mapping = aes(x = Clicked_on_ad, y = Area_income, fill = Clicked_on_ad)) +
  geom_boxplot() +
  coord_flip()
```



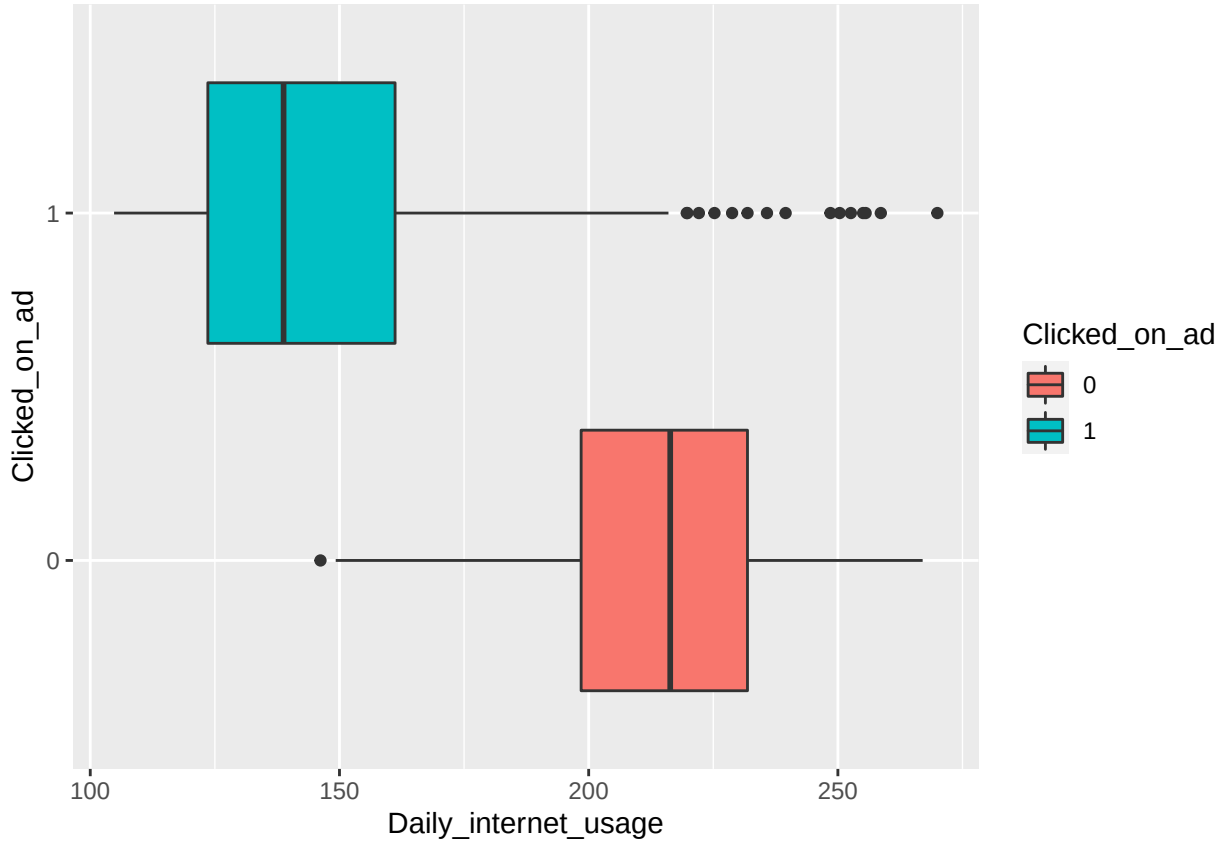

```
#plotting boxplots to show how the age relates with the number of clicks
ggplot(data = advertising, mapping = aes(x = Clicked_on_ad, y = Age, fill = Clicked_on_ad)) +
  geom_boxplot() +
  coord_flip()
```




```
#plotting boxplots to show how the daily time spent on the site relates with the number of clicks
ggplot(data = advertising, mapping = aes(x = Clicked_on_ad, y = Daily_time_spent_on_site, fill = Clicke
  geom_boxplot() +
  coord_flip()
```




```
#plotting boxplots to show how the daily internet usage relates with the number of clicks
ggplot(data = advertising, mapping = aes(x = Clicked_on_ad, y = Daily_internet_usage, fill = Clicked_on_
  geom_boxplot() +
  coord_flip()
```



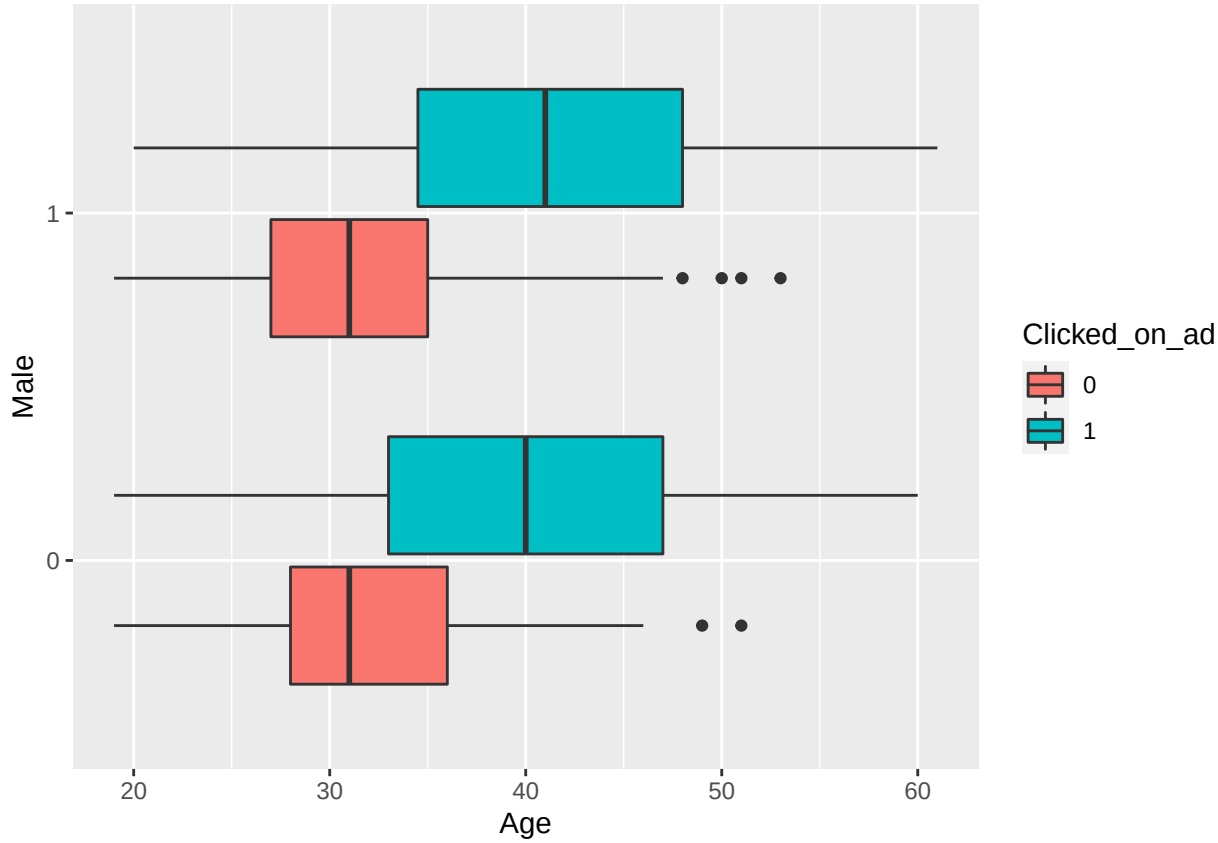
Observations

1. Most people who clicked on the ads have a lower income as compared to those who did Not click on the ads
2. Most people who clicked on the ads were older than those who did NOt click on the ads
3. Most people who clicked on the ads spent way less time on the site as compared to thos who did not click on the ads
4. The daily internet usage of most people who clicked on the ads is way less than those who did NOt click on the ads

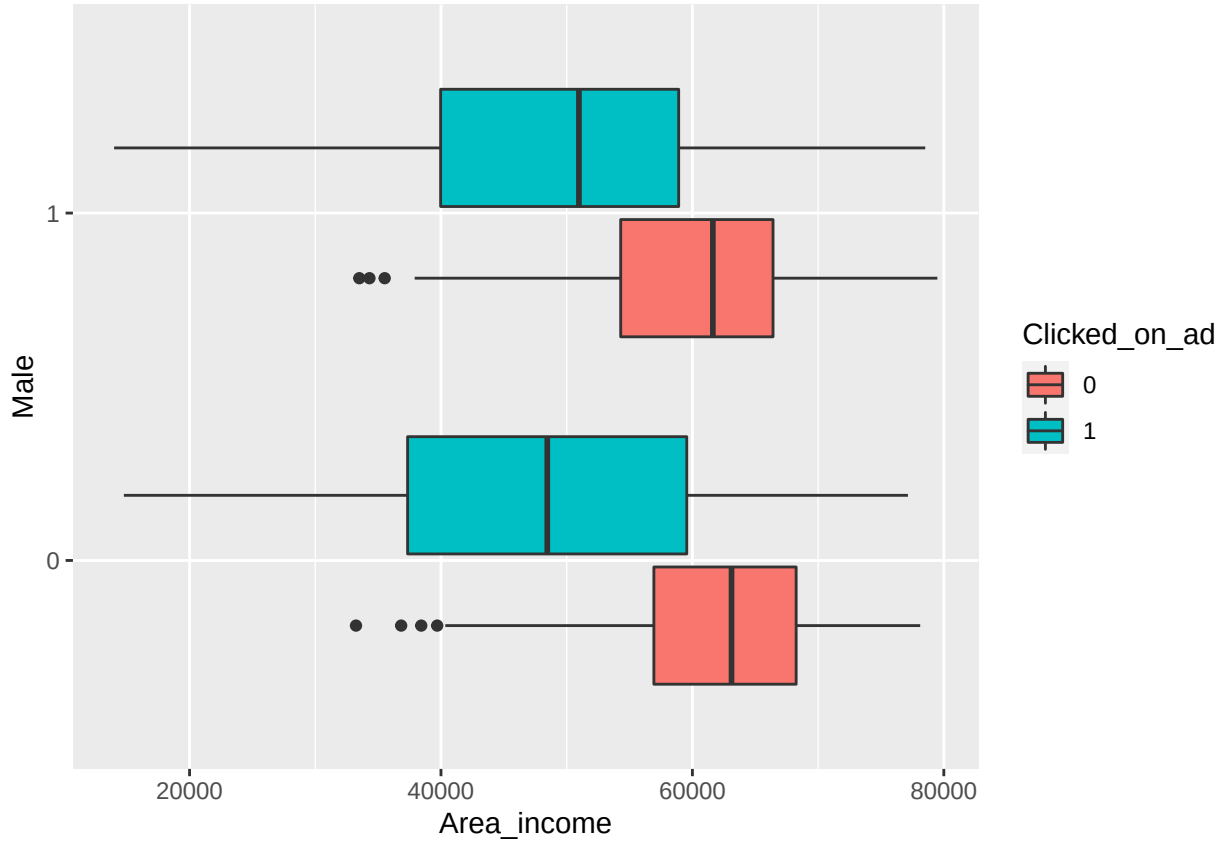
Conclusion

The entepreneur should target people with lower area income levels, older and those who spend less time on the site.

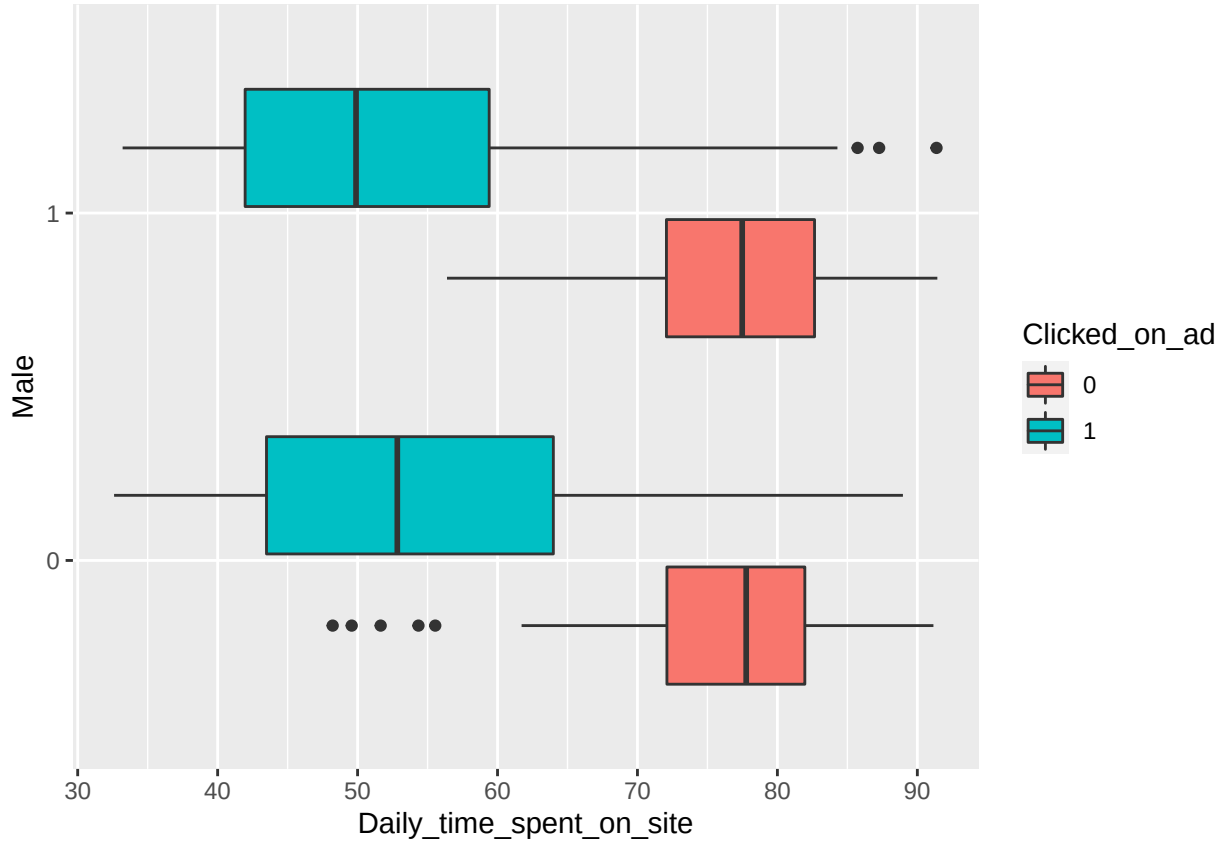
```
#boxplots to compare age and gender
ggplot(data = advertising, mapping = aes(x = Male, y = Age, fill = Clicked_on_ad)) +
  geom_boxplot() +
  coord_flip()
```



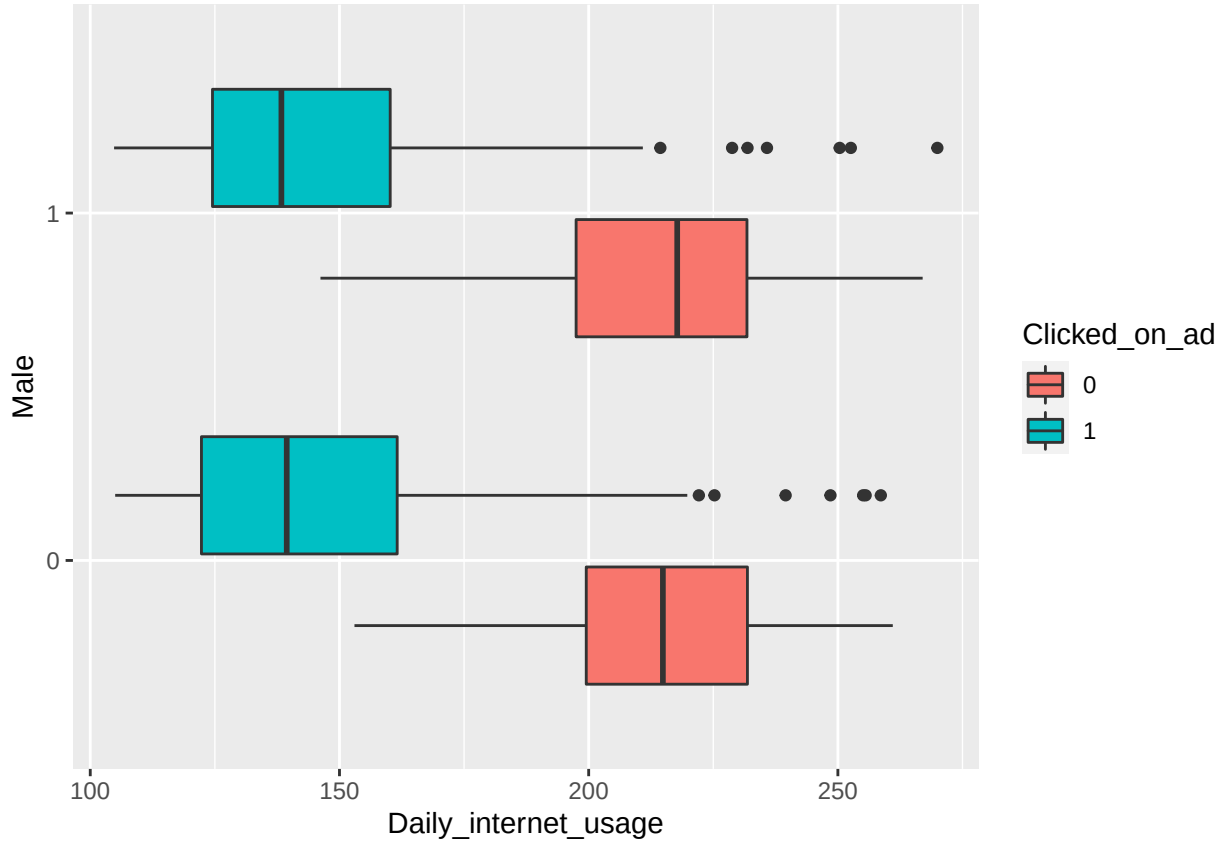

```
#boxplots to compare area income and gender
ggplot(data = advertising, mapping = aes(x = Male, y = Area_income, fill = Clicked_on_ad)) +
  geom_boxplot() +
  coord_flip()
```




```
#boxplots to compare the daily time spent on the site and gender
ggplot(data = advertising, mapping = aes(x = Male, y = Daily_time_spent_on_site, fill = Clicked_on_ad))
  geom_boxplot() +
  coord_flip()
```




```
#plotting boxplots to compare the daily internet usage and gender
ggplot(data = advertising, mapping = aes(x = Male, y = Daily_internet_usage, fill = Clicked_on_ad)) +
  geom_boxplot() +
  coord_flip()
```



Observations

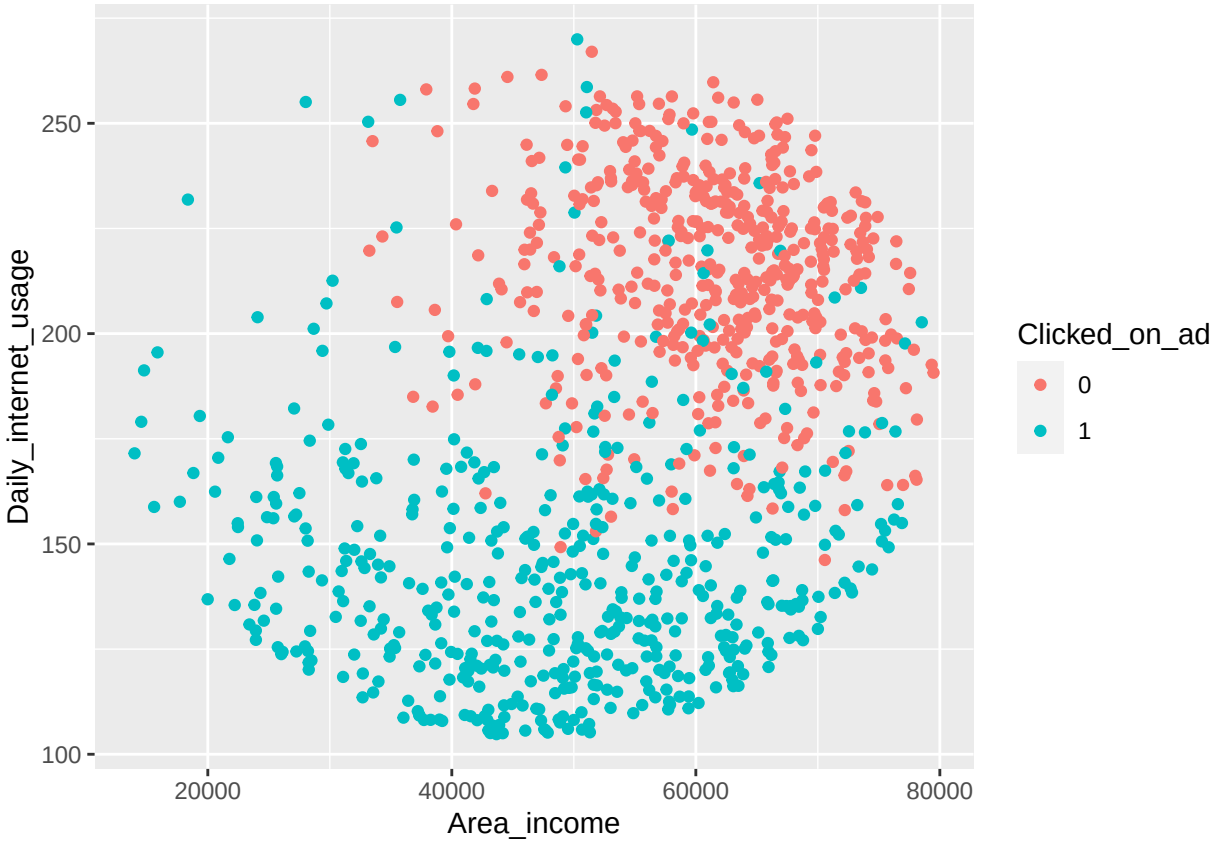
1. Generally, those who clicked on the ads were older, but the males were slightly older than the females
2. Individuals who clicked on the ads have a lower area income as compared to those who did not click on the ads.Of those who clicked on the ads, the males have an average area income that is slightly more than that of the females.
3. More of those who click on the ads spend less time on the site. Of those who click on the ads, the females generally spend more time on the site as compared to the males
4. In general, those who click on the ads have a lower daily internet usage, with a few observations as outlier values

ggplots - scatter plots

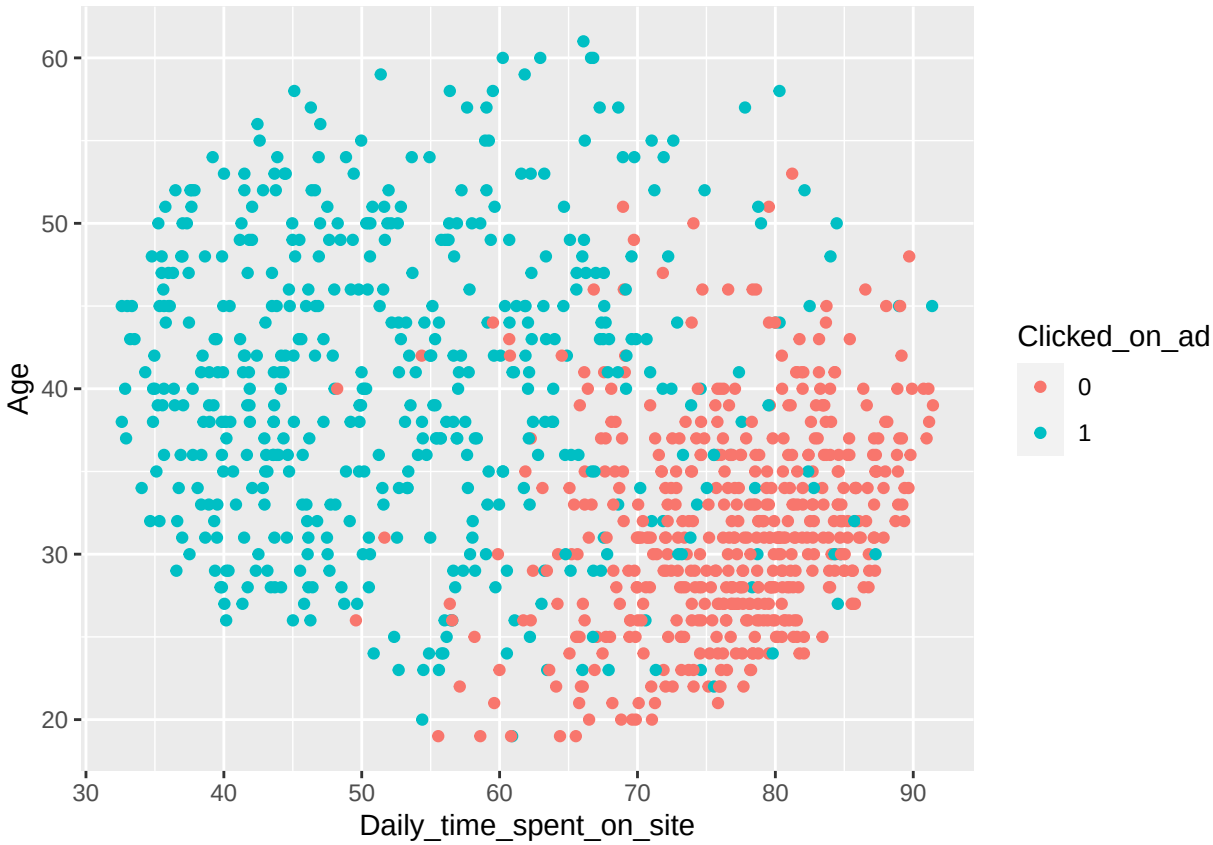
We would like to observe the relationship between the numerical varaibles and the number of clicks obtained from these variables

```
#scatter plot of the age and the area income of an individual
ggplot(data = advertising) +
  geom_point(mapping = aes(x = Area_income, y = Age, color = Clicked_on_ad))
```

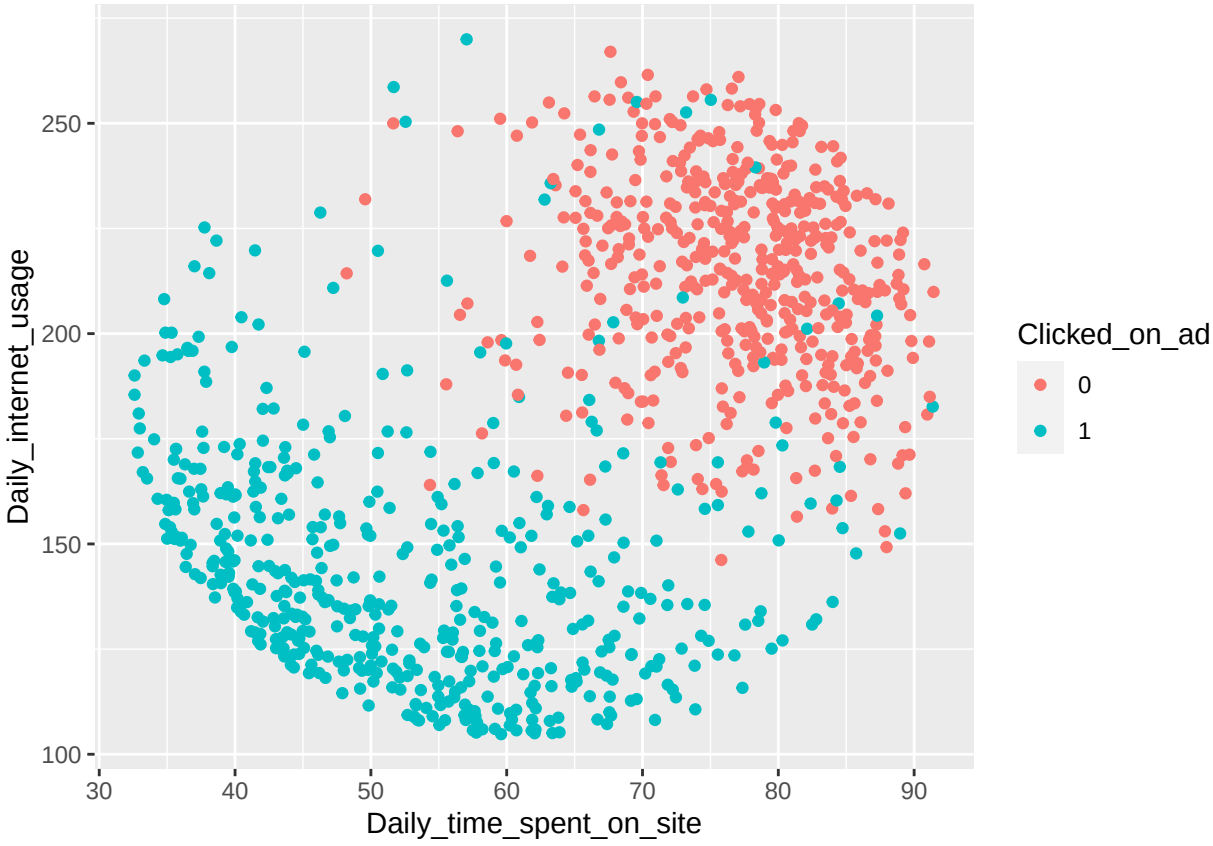

```
#scatter plot of daily internet usage and area income
ggplot(data = advertising) +
  geom_point(mapping = aes(x = Area_income, y = Daily_internet_usage, color = Clicked_on_ad))
```




```
#scatter plot of age and daily time spent on site
ggplot(data = advertising) +
  geom_point(mapping = aes(x = Daily_time_spent_on_site, y = Age, color = Clicked_on_ad))
```




```
#scatter plot of daily time spent on site and daily internet usage
ggplot(data = advertising) +
  geom_point(mapping = aes(x = Daily_time_spent_on_site, y = Daily_internet_usage, color = Clicked_on_ad
```



Observations

1. We observe that the number of people who clicked on the ads are more evenly distributed while most of the people who did not click on the ads have a higher area income and a bit younger
2. A great number of clicks comes from people who's daily internet usage is quite low and area income is also lower as compared to those who do Not click on the ads whose daily internet usage is significantly higher
3. A huge chuck of clicks come from people who spend significantly little time on the site as compared to those who spend more time on the site
4. Clearly, more clicks come from people who spend less time on the site and people whose daily internet usage is significantly lower as compared to those who spend more time on the site and have a high daily internet usage

Conclusions

The ads are getting more clicks from people who spend less time on the site and those whose daily internet usage is low.

```
# filter data with clicked ads only, and select columns for country, city and clicked on ad and hour
library(Lahman)
library(dplyr)
geo <- advertising %>%
  filter(Clicked_on_ad == 1) %>%
  select(Country, City, Clicked_on_ad)
head(geo)
```

```
##                 Country             City Clicked_on_ad
## 1             Australia Port Jefferybury             1
## 2                 Qatar  West Brandonton             1
## 3                 Egypt   West Katiefurt             1
## 4              Barbados     West William             1
## 5                 Spain   New Travistown             1
## 6 Palestinian Territory   West Dylanberg             1
```

```
#filter data without clicks on ads only, and select columns for country, city, aclicked on ad and hour
geo_no_clicks <- advertising %>%
  filter(Clicked_on_ad == 0) %>%
  select(Country, City, Clicked_on_ad, Hour)
head(geo_no_clicks)
```

```
##      Country           City Clicked_on_ad Hour
## 1    Tunisia    Wrightburgh             0   00
## 2      Nauru      West Jodi             0   01
## 3 San Marino       Davidton             0   20
## 4      Italy West Terrifurt             0   02
## 5    Iceland   South Manuel             0   03
## 6     Norway      Jamieberg             0   14
```

```
#what is the size of the two datasets

print(dim(geo)) #print the size of the dataset with clicked ads only
```

```
## [1] 500   3
```

```
print(dim(geo_no_clicks))  #print the dataset without clicked ads
```

```
## [1] 500   4
```

Both the datasets have an equal number of records hence clicks on ads were evenly distributed in the countries

```
#top 10 countries with the most clicks

countries_with_most_clicks <- geo %>%
  filter(Clicked_on_ad == 1) %>%  #creating a filter of the clicked on ads only
  arrange(Clicked_on_ad) %>%      #arranging in order of the frequency of clicked ads only
  head(10)                        #obtaining only the top 10
countries_with_most_clicks        #previweing the dataset
```

```
##                                          Country             City
## 1                                      Australia Port Jefferybury
## 2                                          Qatar  West Brandonton
## 3                                          Egypt   West Katiefurt
## 4                                       Barbados     West William
## 5                                          Spain   New Travistown
## 6                           Palestinian Territory   West Dylanberg
## 7  British Indian Ocean Territory (Chagos Archipelago)     Jessicastad
## 8                             Russian Federation      Millertown
```

```
## 9                                                 Burundi       South John
## 10                                                Tokelau    Harperborough
##    Clicked_on_ad
## 1              1
## 2              1
## 3              1
## 4              1
## 5              1
## 6              1
## 7              1
## 8              1
## 9              1
## 10             1
```

From the above, we observe that the top 10 countries with the most clicks are Australia, Qatar, Egypt, Barbados, Spain, Palestinian Territory, British Indian Ocean Territory, Russian Federation, Burundi and Tokelau. Hence the entrepreneur should probably consider focusing on this countries

```
#top 10 countries with the least clicks

countries_with_least_clicks <- geo_no_clicks %>%
  filter(Clicked_on_ad == 0) %>%
  arrange(Clicked_on_ad) %>%
  head(10)
countries_with_least_clicks
```

```
##       Country              City Clicked_on_ad Hour
## 1     Tunisia       Wrightburgh             0   00
## 2       Nauru         West Jodi             0   01
## 3  San Marino          Davidton             0   20
## 4       Italy    West Terrifurt             0   02
## 5     Iceland      South Manuel             0   03
## 6      Norway         Jamieberg             0   14
## 7     Myanmar       Brandonstad             0   20
## 8     Grenada         West Colin             0   09
## 9       Ghana         Ramirezton             0   01
## 10    Burundi East Theresashire             0   08
```

From the above, we can observe that the top countries with mo clicks are Tunisia, Nauru, San Marino, Italy, Iceland, Norway, Mynmar, Grenada, Ghana and Burundi.

Conclusions

From the analysis above, we have observed that the individuals who were mostly interested in the ads were females who were:

Older Had a lower area income *Spent less time on the ads* Had less daily internet usage

Hence the entrepreneur should target more individuals with these characters. Also, the entrepreneur has the potential of getting a wider market in the countries that did get more clicks.

In terms of time, the entrepreneur is better off increasing the ads towards the end and the beginning of the month as compared to the middle of the month.

5. Implementing the Solution

```
#packages required for modeling
library(psych)
library(countrycode)
library(class)
```

```
#viewing our variables and datatypes
library(tidyverse)
glimpse(advertising)
```

```
## Rows: 1,000
## Columns: 16
## $ Daily_time_spent_on_site <dbl> 68.95, 80.23, 69.47, 74.15, 68.37, 59.99, ...
## $ Age                      <int> 35, 31, 26, 29, 35, 23, 33, 48, 30, 20, 49...
## $ Area_income              <dbl> 61833.90, 68441.85, 59785.94, 54806.18, 73...
## $ Daily_internet_usage     <dbl> 256.09, 193.77, 236.50, 245.89, 225.58, 22...
## $ Ad_topic_line            <chr> "Cloned 5thgeneration orchestration", "Mon...
## $ City                     <chr> "Wrightburgh", "West Jodi", "Davidton", "W...
## $ Male                     <fct> 0, 1, 0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, ...
## $ Country                  <chr> "Tunisia", "Nauru", "San Marino", "Italy",...
## $ Timestamp                <dttm> 2016-03-27 00:53:11, 2016-04-04 01:39:02,...
## $ Clicked_on_ad            <fct> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, ...
## $ Year                     <fct> 2016, 2016, 2016, 2016, 2016, 2016, 2016, ...
## $ Month                    <fct> 03, 04, 03, 01, 06, 05, 01, 03, 04, 07, 03...
## $ Day                      <fct> 27, 04, 13, 10, 03, 19, 28, 07, 18, 11, 16...
## $ Hour                     <fct> 00, 01, 20, 02, 03, 14, 20, 01, 09, 01, 20...
## $ Minutes                  <fct> 53, 39, 35, 31, 36, 30, 59, 40, 33, 42, 19...
## $ Seconds                  <fct> 11, 02, 42, 19, 18, 17, 32, 15, 42, 51, 01...
```

- The cities and countries have high cardinality hence we will convert the countries to continents using the countrycode package and use the continents to perform the modelling

```
library(countrycode)

advertising$Continent <- countrycode(sourcevar = advertising[, "Country"],
                            origin = "country.name",
                            destination = "continent")
```

```
## Warning in countrycode(sourcevar = advertising[, "Country"], origin = "country.name", : Some values w
```

```
#previewing the columns
head(advertising)
```

```
##   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 Year Month Day Hour Minutes Seconds
## 1 2016-03-27 00:53:11             0 2016    03  27   00      53      11
## 2 2016-04-04 01:39:02             0 2016    04  04   01      39      02
## 3 2016-03-13 20:35:42             0 2016    03  13   20      35      42
## 4 2016-01-10 02:31:19             0 2016    01  10   02      31      19
## 5 2016-06-03 03:36:18             0 2016    06  03   03      36      18
## 6 2016-05-19 14:30:17             0 2016    05  19   14      30      17
##   Continent
## 1    Africa
## 2   Oceania
## 3    Europe
## 4    Europe
## 5    Europe
## 6    Europe
```

```
#checking the Continent column
table(advertising$Continent)
```

```
##
##   Africa Americas     Asia   Europe  Oceania
##      214      219      218      214      100
```

All the continents but Oceania seem to be equally represented

```
str(advertising)
```

```
## 'data.frame':    1000 obs. of  17 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                    : Factor w/ 2 levels "0","1": 1 2 1 2 1 2 1 2 2 2 ...
##  $ 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           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 2 1 1 ...
##  $ Year                    : Factor w/ 1 level "2016": 1 1 1 1 1 1 1 1 1 1 ...
##  $ Month                   : Factor w/ 7 levels "01","02","03",..: 3 4 3 1 6 5 1 3 4 7 ...
##  $ Day                     : Factor w/ 31 levels "01","02","03",..: 27 4 13 10 3 19 28 7 18 11 ...
##  $ Hour                    : Factor w/ 24 levels "00","01","02",..: 1 2 21 3 4 15 21 2 10 2 ...
##  $ Minutes                 : Factor w/ 60 levels "00","01","02",..: 54 40 36 32 37 31 60 41 34 43 ..
##  $ Seconds                 : Factor w/ 60 levels "00","01","02",..: 12 3 43 20 19 18 33 16 43 52 ...
##  $ Continent               : chr  "Africa" "Oceania" "Europe" "Europe" ...
```

```r
#Pre-processing: converting variables to appropriate data types for modeling
#converting factor variables to integers
advertising$Male = as.numeric(advertising$Male)
advertising$Month = as.numeric(advertising$Month)
advertising$Day = as.numeric(advertising$Day)
advertising$Hour = as.numeric(advertising$Hour)
advertising$Minutes = as.numeric(advertising$Minutes)
advertising$Seconds = as.numeric(advertising$Seconds)


#clicked on ad variable as a factor
advertising$Clicked_on_ad = as.factor(advertising$Clicked_on_ad)

library(plyr)
```

```
## ------------------------------------------------------------------------------
```

```
## You have loaded plyr after dplyr - this is likely to cause problems.
## If you need functions from both plyr and dplyr, please load plyr first, then dplyr:
## library(plyr); library(dplyr)
```

```
## ------------------------------------------------------------------------------
```

```
##
## Attaching package: 'plyr'
```

```
## The following objects are masked from 'package:dplyr':
##
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
## The following object is masked from 'package:purrr':
##
##     compact
```

```r
#encoding character variable continent
advertising$Continent <- factor(advertising$Continent, order = TRUE, levels =c('Africa', 'Americas', 'As

advertising$Continent_Numeric <-mapvalues(advertising$Continent, from = c('Africa', 'Americas', 'Asia',

glimpse(advertising)
```

```
## Rows: 1,000
## Columns: 18
## $ Daily_time_spent_on_site <dbl> 68.95, 80.23, 69.47, 74.15, 68.37, 59.99, ...
## $ Age                      <int> 35, 31, 26, 29, 35, 23, 33, 48, 30, 20, 49...
## $ Area_income              <dbl> 61833.90, 68441.85, 59785.94, 54806.18, 73...
## $ Daily_internet_usage     <dbl> 256.09, 193.77, 236.50, 245.89, 225.58, 22...
## $ Ad_topic_line            <chr> "Cloned 5thgeneration orchestration", "Mon...
## $ City                     <chr> "Wrightburgh", "West Jodi", "Davidton", "W...
## $ Male                     <dbl> 1, 2, 1, 2, 1, 2, 1, 2, 2, 2, 1, 2, 2, 1, ...
```

```
## $ Country                 <chr> "Tunisia", "Nauru", "San Marino", "Italy",...
## $ Timestamp               <dttm> 2016-03-27 00:53:11, 2016-04-04 01:39:02,...
## $ Clicked_on_ad           <fct> 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, ...
## $ Year                    <fct> 2016, 2016, 2016, 2016, 2016, 2016, 2016, ...
## $ Month                   <dbl> 3, 4, 3, 1, 6, 5, 1, 3, 4, 7, 3, 5, 6, 4, ...
## $ Day                     <dbl> 27, 4, 13, 10, 3, 19, 28, 7, 18, 11, 16, 8...
## $ Hour                    <dbl> 1, 2, 21, 3, 4, 15, 21, 2, 10, 2, 21, 9, 2...
## $ Minutes                 <dbl> 54, 40, 36, 32, 37, 31, 60, 41, 34, 43, 20...
## $ Seconds                 <dbl> 12, 3, 43, 20, 19, 18, 33, 16, 43, 52, 2, ...
## $ Continent               <ord> Africa, Oceania, Europe, Europe, Europe, E...
## $ Continent_Numeric       <ord> 1, 5, 4, 4, 4, 4, 3, 5, 2, 1, 3, 1, 1, 4, ...
```

Some countries could not be recognized by the function countrycode hence were not assigned a continent, this gave rise to some null values, let's look into this

```
#checking for null values
colSums(is.na(advertising))
```

```
## 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                     Year                   Month
##                        0                        0                       0
##                      Day                     Hour                 Minutes
##                        0                        0                       0
##                  Seconds                Continent       Continent_Numeric
##                        0                       35                      35
```

We have 35 missing values in the continent column. We will drop these values as we don't want to make any assumptions

```
#deleting null values

advertising <- na.omit(advertising)

colSums(is.na(advertising))
```

```
## 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                     Year                   Month
##                        0                        0                       0
##                      Day                     Hour                 Minutes
##                        0                        0                       0
##                  Seconds                Continent       Continent_Numeric
##                        0                        0                       0
```

```r
#deleting the timestamp, year and continent columns as they are irrelevant for modeling
#omitting the ad_line_topic, country, and city columns as they have high cardinality

advertising$Timestamp <- NULL
advertising$Year <- NULL
advertising$Ad_topic_line <- NULL
advertising$City <- NULL
advertising$Country <- NULL
advertising$Continent <- NULL

#checking the columns has been removed
colnames(advertising)
```

```
##  [1] "Daily_time_spent_on_site" "Age"
##  [3] "Area_income"              "Daily_internet_usage"
##  [5] "Male"                     "Clicked_on_ad"
##  [7] "Month"                    "Day"
##  [9] "Hour"                     "Minutes"
## [11] "Seconds"                  "Continent_Numeric"
```

We now have only relevant columns for our study

```r
#obtaining a descriptive summary of the variables

#library(psych)
describe(advertising)
```

```
##                          vars   n     mean       sd   median  trimmed      mad
## Daily_time_spent_on_site    1 965    65.15    15.76    68.25    65.91    17.76
## Age                         2 965    36.04     8.83    35.00    35.54     8.90
## Area_income                 3 965 54972.55 13433.69 56986.73 55990.94 13370.69
## Daily_internet_usage        4 965   179.86    43.96   182.65   179.85    58.73
## Male                        5 965     1.48     0.50     1.00     1.47     0.00
## Clicked_on_ad*              6 965     1.50     0.50     2.00     1.50     0.00
## Month                       7 965     3.81     1.92     4.00     3.76     2.97
## Day                         8 965    15.54     8.76    15.00    15.48    11.86
## Hour                        9 965    12.68     6.97    13.00    12.71     8.90
## Minutes                    10 965    30.13    17.22    31.00    30.09    22.24
## Seconds                    11 965    30.71    16.88    31.00    30.80    20.76
## Continent_Numeric*         12 965     2.76     1.30     3.00     2.70     1.48
##                             min      max    range  skew kurtosis     se
## Daily_time_spent_on_site  32.60    91.43    58.83 -0.38    -1.07   0.51
## Age                       19.00    61.00    42.00  0.47    -0.43   0.28
## Area_income             13996.50 79484.80 65488.30 -0.64    -0.13 432.45
## Daily_internet_usage      104.78   269.96   165.18 -0.03    -1.27   1.41
## Male                       1.00     2.00     1.00  0.09    -1.99   0.02
## Clicked_on_ad*             1.00     2.00     1.00  0.00    -2.00   0.02
## Month                      1.00     7.00     6.00  0.09    -1.18   0.06
## Day                        1.00    31.00    30.00  0.04    -1.18   0.28
## Hour                       1.00    24.00    23.00 -0.01    -1.23   0.22
## Minutes                    1.00    60.00    59.00  0.02    -1.18   0.55
## Seconds                    1.00    60.00    59.00 -0.03    -1.15   0.54
## Continent_Numeric*         1.00     5.00     4.00  0.13    -1.14   0.04
```

We observe that all the variables have varying range values both continous and factor variables. Hence we will normalize the continous variables

```
#normalizing the continous variables
normalize <- function(x) (
  return( ((x - min(x)) /(max(x)-min(x))) )
)

advertising$Daily_time_spent_on_site <- normalize(advertising$Daily_time_spent_on_site)
advertising$Daily_internet_usage <- normalize(advertising$Daily_internet_usage)
advertising$Area_income <- normalize(advertising$Area_income)
advertising$Age<- normalize(advertising$Age)

head(advertising)
```

```
##   Daily_time_spent_on_site       Age Area_income Daily_internet_usage Male
## 1                0.6178820 0.3809524   0.7304725            0.9160310    1
## 2                0.8096209 0.2857143   0.8313752            0.5387456    2
## 3                0.6267211 0.1666667   0.6992003            0.7974331    1
## 4                0.7062723 0.2380952   0.6231599            0.8542802    2
## 5                0.6080231 0.3809524   0.9145678            0.7313234    1
## 6                0.4655788 0.0952381   0.6988280            0.7383460    2
##   Clicked_on_ad Month Day Hour Minutes Seconds Continent_Numeric
## 1             0     3  27    1      54      12                 1
## 2             0     4   4    2      40       3                 5
## 3             0     3  13   21      36      43                 4
## 4             0     1  10    3      32      20                 4
## 5             0     6   3    4      37      19                 4
## 6             0     5  19   15      31      18                 4
```

```
#checking the structure of our dataframe
str(advertising)
```

```
## 'data.frame':    965 obs. of  12 variables:
##  $ Daily_time_spent_on_site: num  0.618 0.81 0.627 0.706 0.608 ...
##  $ Age                     : num  0.381 0.286 0.167 0.238 0.381 ...
##  $ Area_income             : num  0.73 0.831 0.699 0.623 0.915 ...
##  $ Daily_internet_usage    : num  0.916 0.539 0.797 0.854 0.731 ...
##  $ Male                    : num  1 2 1 2 1 2 1 2 2 2 ...
##  $ Clicked_on_ad           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 2 1 1 ...
##  $ Month                   : num  3 4 3 1 6 5 1 3 4 7 ...
##  $ Day                     : num  27 4 13 10 3 19 28 7 18 11 ...
##  $ Hour                    : num  1 2 21 3 4 15 21 2 10 2 ...
##  $ Minutes                 : num  54 40 36 32 37 31 60 41 34 43 ...
##  $ Seconds                 : num  12 3 43 20 19 18 33 16 43 52 ...
##  $ Continent_Numeric       : Ord.factor w/ 5 levels "1"<"2"<"3"<"4"<..: 1 5 4 4 4 4 3 5 2 1 ...
##  - attr(*, "na.action")= 'omit' Named int [1:35] 19 30 54 71 85 95 110 132 134 137 ...
##   ..- attr(*, "names")= chr [1:35] "19" "30" "54" "71" ...
```

Supervised Machine Learning Models

We will apply various classification models to our dataset to predict persons who are likely to click on the ads

```r
set.seed(123)
# Creating a random number equal 70% of total number of rows
ran <- sample(1:nrow(advertising),0.7 * nrow(advertising))

# The training dataset extracted
ad_train <- advertising[ran,]
#head(ad_train)

# The test dataset extracted
ad_test <- advertising[-ran,]
#ad_test

#extracting the target variable from the target variable
ad_target <- (advertising[ran,6])
#ad_target

#extracting the target variable from the test dataset
test_target <- (advertising[-ran,6])
#test_target

#calculating the square root of the length of the target variable to get an optimal k
print(sqrt(length(ad_test)))
```

Baseline Model - K-Nearest Neighbors

```
## [1] 3.464102
```

```r
#3

# Running the knn function, with k = 3 as from the above calculation
library(class)
k <- knn(ad_train,ad_test,cl=ad_target,k=3)

# Creating the confucion matrix
matrix <- table(k,test_target)
print(matrix)
```

```
##    test_target
## k    0  1
##   0 75 59
##   1 76 80
```

```r
# Checking the accuracy
accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
print(accuracy(matrix))
```

```
## [1] 53.44828
```

Our baseline model has an accuracy score of 53%, which is poor hence we will use random forests, svm and naive bayes to try and achieve a better accuracy

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

Naive Bayes Algorithm

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

## The following objects are masked from 'package:propagate':
##
##     kurtosis, skewness
```

```
library(rpart)
library(caret)
```

```
## Loading required package: lattice

##
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':
##
##     lift
```

```
#splitting data into 80% training and 20% test sets

#indxTrain <- createDataPartition(y = advertising$Clicked_on_ad,p = 0.8,list = FALSE)
#training <- advertising[indxTrain,]
#testing <- advertising[-indxTrain,]

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

#x = training[,c(1:5,7:12)]
#y = training$Clicked_on_ad
```

```
# Building Naive Bayes model on our data
# setting the metod as cross validation with 10 iterations
model = train(ad_train, ad_target,'nb',trControl=trainControl(method='cv',number=10))
```

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

# Getting the confusion matrix to see accuracy value and other parameter values
confusionMatrix(Predict, test_target )
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 151   0
##          1   0 139
##
##                Accuracy : 1
##                  95% CI : (0.9874, 1)
##     No Information Rate : 0.5207
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 1
##
##  Mcnemar's Test P-Value : NA
##
##             Sensitivity : 1.0000
##             Specificity : 1.0000
##          Pos Pred Value : 1.0000
##          Neg Pred Value : 1.0000
##              Prevalence : 0.5207
##          Detection Rate : 0.5207
##    Detection Prevalence : 0.5207
##       Balanced Accuracy : 1.0000
##
##        'Positive' Class : 0
##
```

From the Naive Bayes Model, we have an accuracy score of 100%, with all the observations classified correclty. This is an overfit which means that our model would not do well with new data

```
# Svm modeling packages
#library(caret)    # for classification and regression training
library(kernlab)  # for fitting SVMs
```

Svm Model

```
##
## Attaching package: 'kernlab'

## The following object is masked from 'package:psych':
##
##     alpha

## The following object is masked from 'package:purrr':
##
##     cross

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

```r
# Model interpretability packages
library(pdp)      # for partial dependence plots, etc.
```

```
##
## Attaching package: 'pdp'

## The following object is masked from 'package:purrr':
##
##     partial
```

```r
library(vip)      # for variable importance plots
```

```
##
## Attaching package: 'vip'

## The following object is masked from 'package:utils':
##
##     vi
```

```r
#setting the method to repeatedcv and 10 number of iterations
trctrl <- trainControl(method = "repeatedcv", number = 10, repeats = 3)

#fitting svm linear
svm_Linear <- train(Clicked_on_ad ~., data = ad_train, method = "svmLinear",
trControl=trctrl,
preProcess = c("center", "scale"),
tuneLength = 10)

#result of our train model
svm_Linear
```

```
## Support Vector Machines with Linear Kernel
##
## 675 samples
##  11 predictor
##   2 classes: '0', '1'
##
## Pre-processing: centered (14), scaled (14)
## Resampling: Cross-Validated (10 fold, repeated 3 times)
## Summary of sample sizes: 607, 607, 607, 608, 607, 608, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.9619179  0.923818
##
## Tuning parameter 'C' was held constant at a value of 1
```

```r
#model performance
#predictions
svm_pred <- predict(svm_Linear, newdata = ad_test)

#confusion matrix
confusionMatrix(table(svm_pred, ad_test$Clicked_on_ad))
```

```
## Confusion Matrix and Statistics
##
##
## svm_pred   0   1
##        0 149   6
##        1   2 133
##
##                Accuracy : 0.9724
##                  95% CI : (0.9464, 0.988)
##     No Information Rate : 0.5207
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9447
##
##  Mcnemar's Test P-Value : 0.2888
##
##             Sensitivity : 0.9868
##             Specificity : 0.9568
##          Pos Pred Value : 0.9613
##          Neg Pred Value : 0.9852
##              Prevalence : 0.5207
##          Detection Rate : 0.5138
##    Detection Prevalence : 0.5345
##       Balanced Accuracy : 0.9718
##
##        'Positive' Class : 0
##
```

The svm model has an accuracy of approximately 98% on the data, with a total of 5 incorrect classifications. This is so far the best model for the data as it has a reasonable accuracy score.

```
library(caret)
# Control params for SVM

ctrl <- trainControl(
  method = "cv",
  number = 10,
)

# Tune an SVM
set.seed(5628)            # for reproducibility
svm <- train(
  Clicked_on_ad ~ .,
  data = ad_train,
  method = "svmRadial",    # using RadialBasis
  preProcess = c("center", "scale"),
  trControl = ctrl,
  tuneLength = 10
)

# Print results
svm
```

Improving svm Model Performace

```
## Support Vector Machines with Radial Basis Function Kernel
##
## 675 samples
##  11 predictor
##   2 classes: '0', '1'
##
## Pre-processing: centered (14), scaled (14)
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 606, 607, 608, 607, 608, 608, ...
## Resampling results across tuning parameters:
##
##   C       Accuracy   Kappa
##     0.25  0.9600282  0.9201480
##     0.50  0.9629261  0.9259359
##     1.00  0.9600062  0.9200994
##     2.00  0.9599630  0.9199590
##     4.00  0.9584704  0.9169692
##     8.00  0.9569779  0.9140125
##    16.00  0.9510077  0.9020057
##    32.00  0.9420745  0.8841187
##    64.00  0.9316919  0.8633358
##   128.00  0.9243377  0.8485967
##
## Tuning parameter 'sigma' was held constant at a value of 0.0446545
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were sigma = 0.0446545 and C = 0.5.
```

```
#predictions
svm_pred <- predict(svm, newdata = ad_test)

#confusion matrix
confusionMatrix(table(svm_pred, ad_test$Clicked_on_ad))
```

```
## Confusion Matrix and Statistics
##
##
## svm_pred   0   1
##        0 150   8
##        1   1 131
##
##                Accuracy : 0.969
##                  95% CI : (0.9419, 0.9857)
##     No Information Rate : 0.5207
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9377
##
##  Mcnemar's Test P-Value : 0.0455
##
##             Sensitivity : 0.9934
##             Specificity : 0.9424
##          Pos Pred Value : 0.9494
```

```
##          Neg Pred Value : 0.9924
##              Prevalence : 0.5207
##          Detection Rate : 0.5172
##    Detection Prevalence : 0.5448
##       Balanced Accuracy : 0.9679
##
##        'Positive' Class : 0
##
```

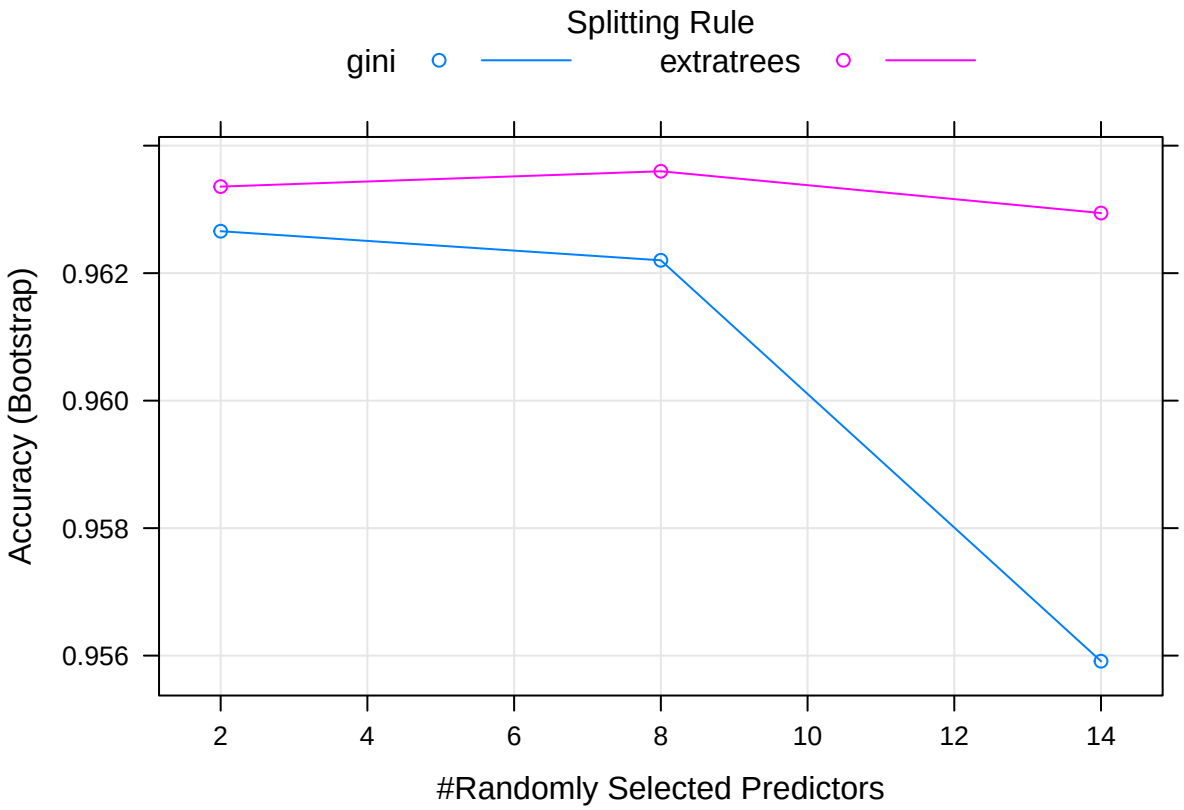
The svmRadial has actually performed worse than the svmLinear hence the svmLinear is the better svm model for this dataset

```
#fitting a single tree to the data
set.seed(12)
library(ranger)
library(caret)
rf_model <- train(Clicked_on_ad ~ .,
                  data = advertising,
                  method = "ranger")
rf_model
```

Random Forests Model

```
## Random Forest
##
## 965 samples
##  11 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 965, 965, 965, 965, 965, 965, ...
## Resampling results across tuning parameters:
##
##   mtry  splitrule   Accuracy   Kappa
##    2    gini        0.9626580  0.9252212
##    2    extratrees  0.9633578  0.9266471
##    8    gini        0.9622016  0.9242872
##    8    extratrees  0.9635985  0.9271041
##   14    gini        0.9559121  0.9117077
##   14    extratrees  0.9629428  0.9257960
##
## Tuning parameter 'min.node.size' was held constant at a value of 1
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were mtry = 8, splitrule = extratrees
##  and min.node.size = 1.
```

```
#plotting the model
plot(rf_model)
```

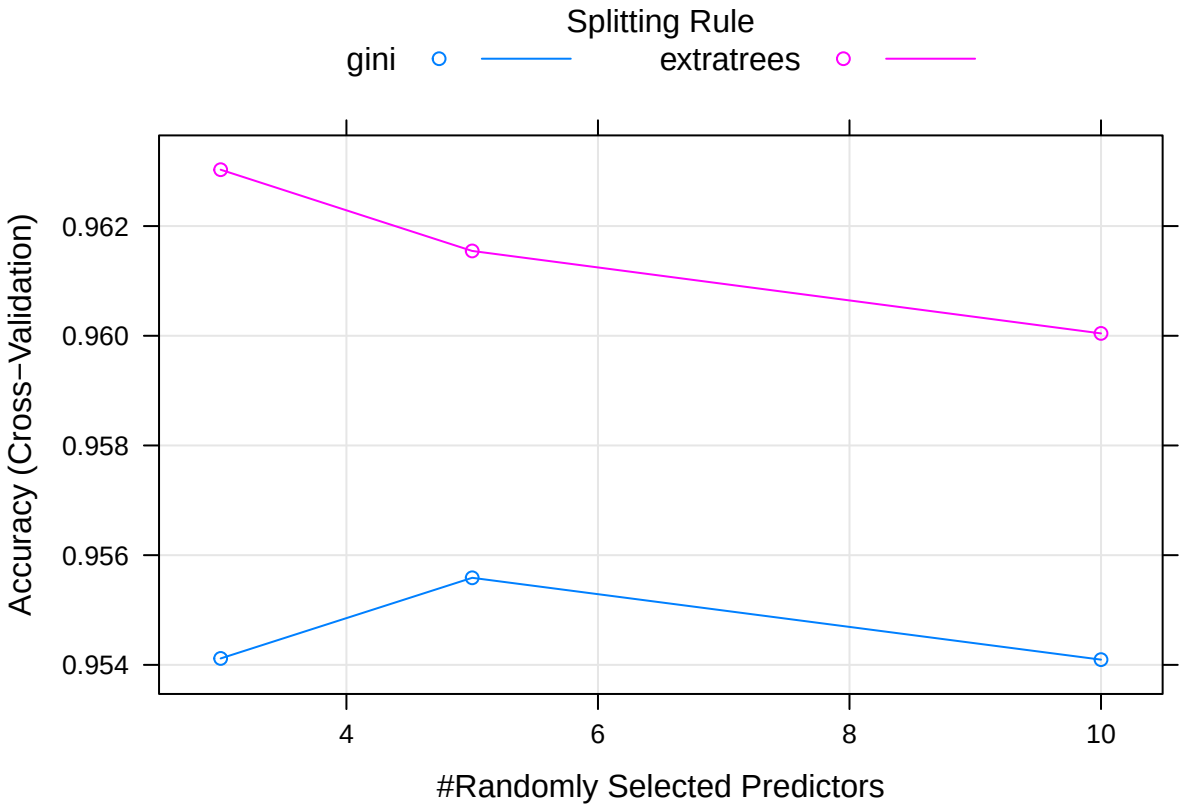


The Gini index reduces steadily from 2 to 8 and then gradually from 8 to 14, with an aim to reduce gini impurity. This means that the splitting rule is selecting the best gin but towards the end, it is overfitting

```
# Improving model performance
# Training the model
rf_model1 <- train(Clicked_on_ad ~ .,
                   data = ad_train,
                   method = "ranger",
                   tuneLength = 5)

#setting grid search
set.seed(42)
myGrid <- expand.grid(mtry = c(3,5,10),
                      splitrule = c("gini", "extratrees"),
                      min.node.size = 10)
rf_model1 <- train(Clicked_on_ad ~ .,
                   data = ad_train,
                   method = "ranger",
                   tuneGrid = myGrid,
                   trControl = trainControl(method = "cv",
                                            number = 5,
                                            verboseIter = FALSE))

# Printing the model
rf_model1
```

```
## Random Forest
##
## 675 samples
##  11 predictor
##   2 classes: '0', '1'
##
## No pre-processing
## Resampling: Cross-Validated (5 fold)
## Summary of sample sizes: 540, 541, 540, 540, 539
## Resampling results across tuning parameters:
##
##   mtry  splitrule   Accuracy   Kappa
##    3    gini        0.9541170  0.9082053
##    3    extratrees  0.9630278  0.9260783
##    5    gini        0.9555874  0.9111606
##    5    extratrees  0.9615464  0.9231109
##   10    gini        0.9540950  0.9081800
##   10    extratrees  0.9600429  0.9201043
##
## Tuning parameter 'min.node.size' was held constant at a value of 10
## Accuracy was used to select the optimal model using the largest value.
## The final values used for the model were mtry = 3, splitrule = extratrees
##  and min.node.size = 10.
```

```
plot(rf_model1)
```



The Gini is a straight line with a descending slope while the splitting rule increases steadily at first then stagnates after sometime after selecting the split that minimizes the gini impurity

```
#evaluating model performance
#predictions
rf_pred <- predict(rf_model1, newdata = ad_test)

#confusion matrix
confusionMatrix(table(rf_pred, ad_test$Clicked_on_ad))
```

```
## Confusion Matrix and Statistics
##
##
## rf_pred   0   1
##       0 150   7
##       1   1 132
##
##                Accuracy : 0.9724
##                  95% CI : (0.9464, 0.988)
##     No Information Rate : 0.5207
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.9446
##
##  Mcnemar's Test P-Value : 0.0771
##
##             Sensitivity : 0.9934
##             Specificity : 0.9496
##          Pos Pred Value : 0.9554
##          Neg Pred Value : 0.9925
##              Prevalence : 0.5207
##          Detection Rate : 0.5172
##    Detection Prevalence : 0.5414
##       Balanced Accuracy : 0.9715
##
##        'Positive' Class : 0
##
```

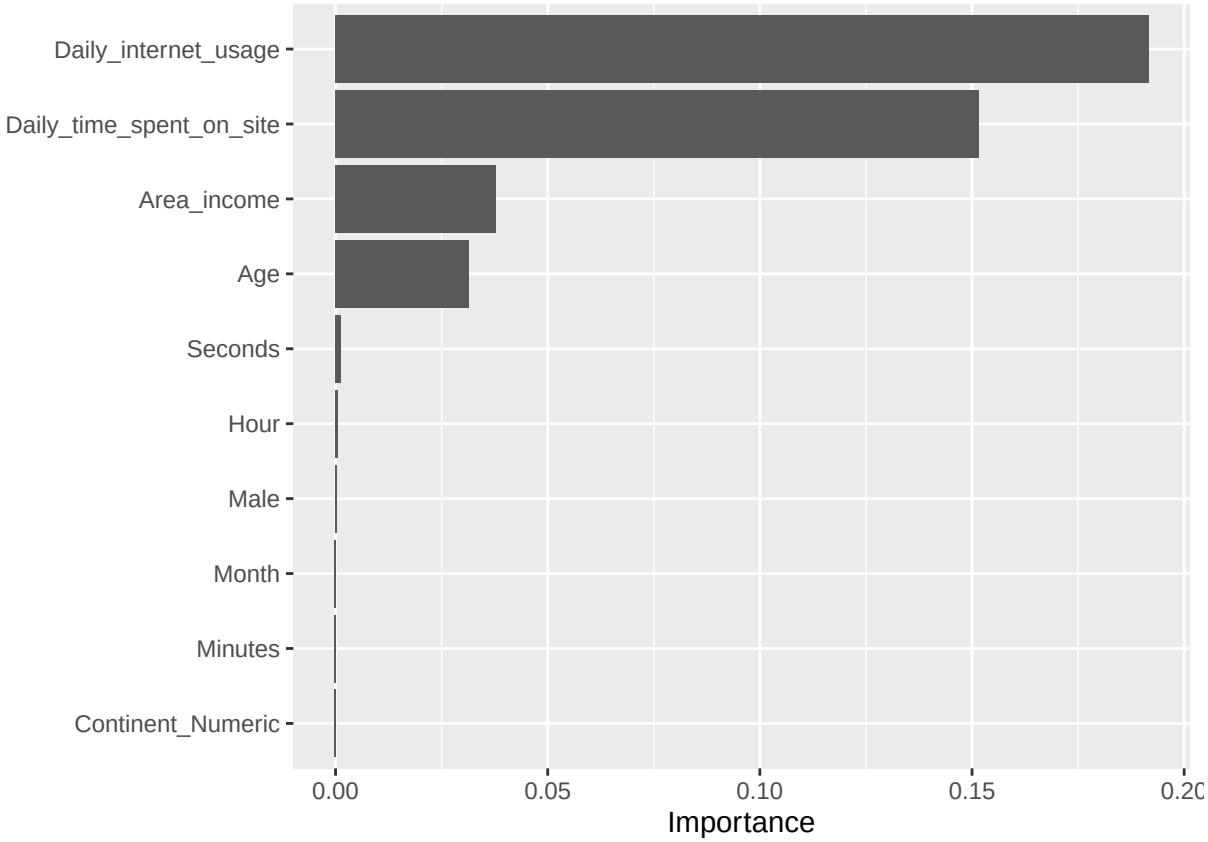
The Random Forests has obtained an accuracy of approximately 98% with 6 incorrect observations. This is the second best model we have so far.

```
#Feature Importance
library(ranger)
# re-run model with permutation-based variable importance
rf_permutation <- ranger(
  formula = Clicked_on_ad ~ .,
  data = ad_train,
  #num.trees = 2000,
  mtry = 3,
  min.node.size = 10,
  #sample.fraction = .80,
  replace = FALSE,
  importance = "permutation",
```

```
  respect.unordered.factors = "order",
  verbose = FALSE,
  seed  = 123
)
```

```
#plotting a graph of the feature importance
vip::vip(rf_permutation, num_features = 10, bar = TRUE)
```

```
## Warning in vip.default(rf_permutation, num_features = 10, bar = TRUE): The 'bar'
## argument has been deprecated in favor of the new 'geom' argument. It will be
## removed in version 0.3.0.
```



From the graph above, it is evident that the variables that are most important are;

- Daily internet usage
- Daily time spent on site
- Age
- Area income

while the minutes, month and seconds do not have any influence on predicting who is most likely to click on the ads.

```
#library(caretEnsemble)
#library(mlr)

#set.seed(100)

#control_stacking <- trainControl(method="repeatedcv", number=5, repeats=2, savePredictions=TRUE, class

#algorithms_to_use <- c('rpart', 'glm', 'knn', 'svmLinear')

#stacked_models <- caretList(Clicked_on_ad ~., data=advertising, trControl=control_stacking, #methodLis

#stacking_results <- resamples(stacked_models)

#summary(stacking_results)
```

Stacked Models

6. Challenging the solution

From the solutions above, the svm model has performed the best with a total of only 5 incorrect observations followed by the Decision Trees with a total of 6 incorrect observations. Both of the models have resulted to an accuracy of approximately 98% which has fulfilled our metric for succees, hence we can say that this study has been successful. However, we should have stacked the four models we have used to combine the prediction power of all to get better results.

6. Follow up questions

1. How would our models perform using different metrics of success?
2. Would having more observations improve our models?

The End!