Analysis for ad target audience

Cindy Gachuhi

21 January 2022

R Week 1 IP

Defining the question

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

Loading the dataset

Dataset: http://bit.ly/IPAdvertisingData

```
# Let us import the library 'Data Table'
#
library("data.table")
ad_data <- fread('http://bit.ly/IPAdvertisingData')

# preview the first 6 values
head(ad_data)
```

```
##    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
```

Cleaning the dataset

Removing missing values

```
# using the function 'colSums', we can identify the total number of missing values in each column
#
colSums(is.na(ad_data))
```

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

We can see that no columns in our dataset contain missing values.

Removing duplicates

```
# To get rid of duplicates, we will identify the unique values from our dataset and assign them to vari
unique_ad <- unique(ad_data)

# print the variable and check out the unique values
unique_ad
```

```
##       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
##   ---
##  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
##    1:    Cloned 5thgeneration orchestration    Wrightburgh    0
##    2:    Monitored national standardization      West Jodi    1
##    3:      Organic bottom-line service-desk       Davidton    0
```

```
##    4: Triple-buffered reciprocal time-frame West Terrifurt    1
##    5:         Robust logistical utilization   South Manuel    0
##   ---
##  996:         Fundamental modular algorithm      Duffystad    1
##  997:       Grass-roots cohesive monitoring    New Darlene    1
##  998:          Expanded intangible solution  South Jessica    1
##  999:  Proactive bandwidth-monitored policy    West Steven    0
## 1000:       Virtual 5thgeneration emulation    Ronniemouth    0
##                       Country           Timestamp Clicked on Ad
##    1:                Tunisia 2016-03-27 00:53:11             0
##    2:                  Nauru 2016-04-04 01:39:02             0
##    3:             San Marino 2016-03-13 20:35:42             0
##    4:                  Italy 2016-01-10 02:31:19             0
##    5:                Iceland 2016-06-03 03:36:18             0
##   ---
##  996:                Lebanon 2016-02-11 21:49:00             1
##  997: Bosnia and Herzegovina 2016-04-22 02:07:01             1
##  998:               Mongolia 2016-02-01 17:24:57             1
##  999:              Guatemala 2016-03-24 02:35:54             0
## 1000:                 Brazil 2016-06-03 21:43:21             1
```

```r
# let us confirm whether all the duplicates have been removed
duplicated_ad <- ad_data[duplicated(ad_data),]
duplicated_ad
```

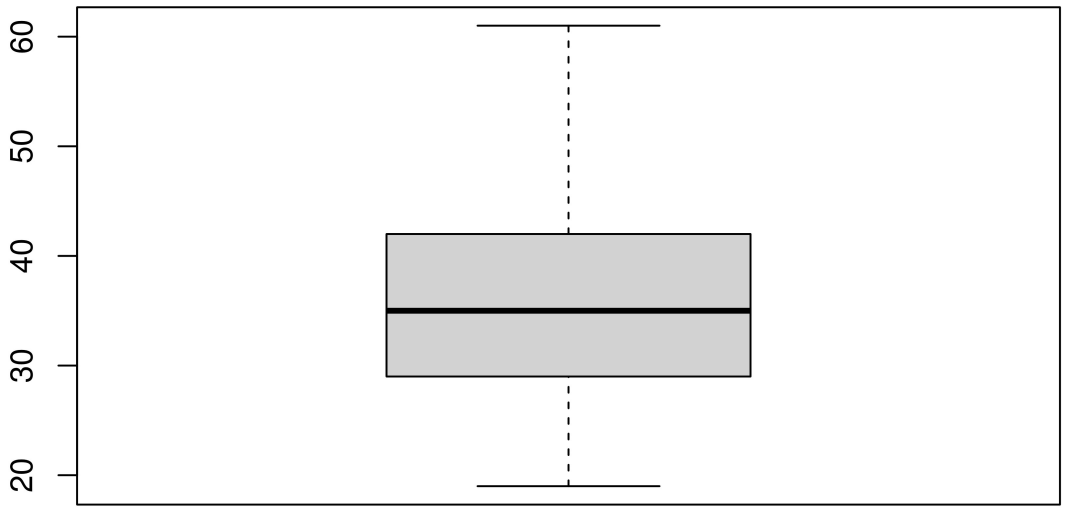
```
## Empty data.table (0 rows and 10 cols): Daily Time Spent on Site,Age,Area Income,Daily Internet Usage
```

It seems that the dataset has no duplicated values. We originally had 1000 rows and even after getting the unique values, the number of rows remained the same. After printing out variable 'duplicated_ad', we had no output; confirming that indeed our dataset has no duplicated values.

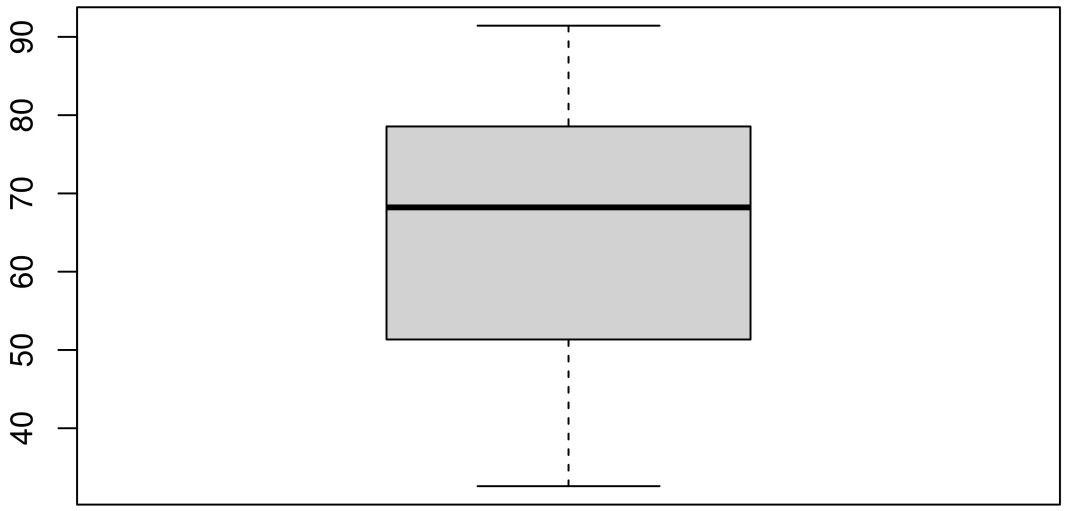
Checking for outliers

An outlier is an observation that is numerically distant from the rest of the data. When reviewing a boxplot, an outlier is defined as a data point that is located outside the fences ("whiskers") of the boxplot.

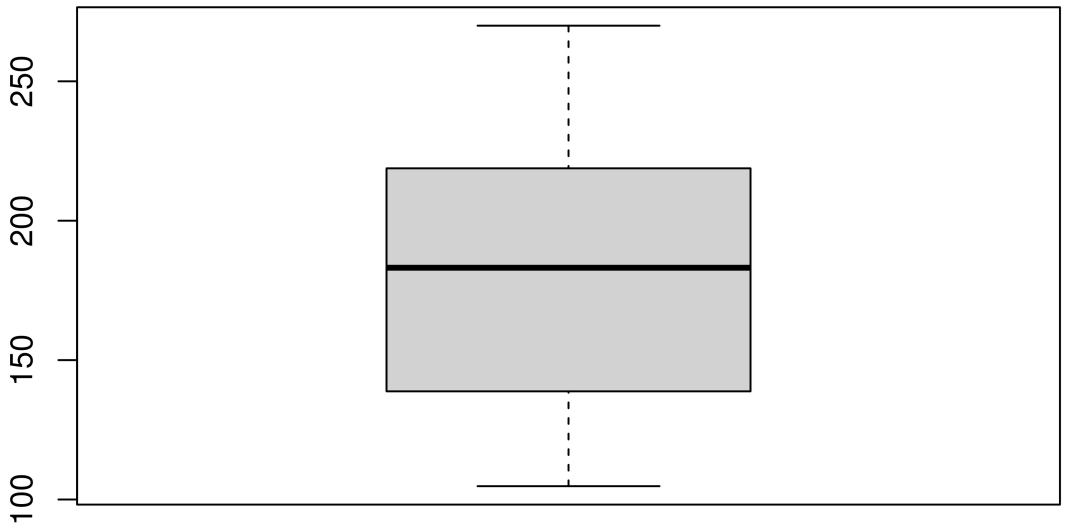
```r
# using boxplots, we will check for outliers in various columns
# column 'Age'

boxplot(ad_data$Age) # No outliers!
```



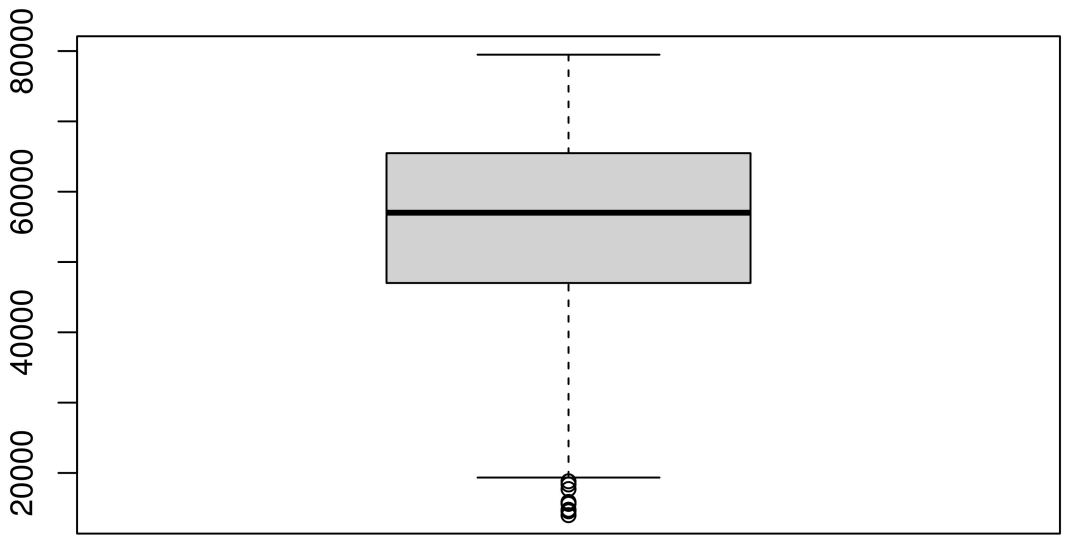

```r
#
boxplot(ad_data$`Daily Time Spent on Site`) # No outliers!
```




```
#
boxplot(ad_data$`Daily Internet Usage`) # No outliers!
```




```
#
boxplot(ad_data$`Area Income`)
```




```
# Since the outliers are not too far from our minimum value, we will keep them to avoid removing import
```

Univariate analysis

Measures and Central Tendency

```
# to get the descriptive statistics of all the numerical variables, we will use the function 'summary'
summary(ad_data)
```

```
##  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        Min.   :0.000   Length:1000
##  Class :character   Class :character   1st Qu.:0.000   Class :character
##  Mode  :character   Mode  :character   Median :0.000   Mode  :character
##                                        Mean   :0.481
##                                        3rd Qu.:1.000
##                                        Max.   :1.000
```

```
##    Timestamp                    Clicked on Ad
##  Min.   :2016-01-01 02:52:10   Min.   :0.0
##  1st Qu.:2016-02-18 02:55:42   1st Qu.:0.0
##  Median :2016-04-07 17:27:29   Median :0.5
##  Mean   :2016-04-10 10:34:06   Mean   :0.5
##  3rd Qu.:2016-05-31 03:18:14   3rd Qu.:1.0
##  Max.   :2016-07-24 00:22:16   Max.   :1.0
```

```r
#
# since mode doesn't have an inbuilt function, we will create a variable 'mode'
#
mode <- function(v) {
   uniqv <- unique(v)
   uniqv[which.max(tabulate(match(v, uniqv)))]
}

# let us get the mode in column 'Age'
age.mode <- mode(ad_data$Age)

#print out mode
age.mode
```

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

Measures of Dispersion

```r
# we already got the quantiles, minimum and maximum values
# for the rest of the measures, we will focus on column 'daily time spent on site'
#
# range
site_time.range <- range(ad_data$`Daily Time Spent on Site`)
site_time.range
```

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

```r
# variance
# The variance is a numerical measure of how the data values is dispersed around the mean.
site_time.var <- var(ad_data$`Daily Time Spent on Site`)
site_time.var
```

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

```r
# standard deviation
site_time.sd <- sd(ad_data$`Daily Time Spent on Site`)
site_time.sd
```

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

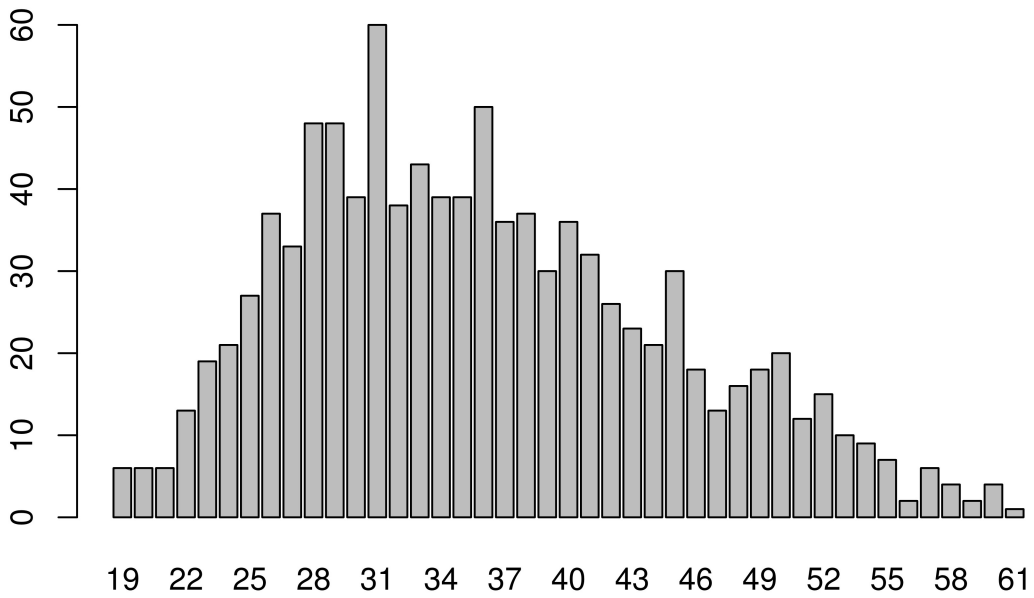
Univariate visualizations

```r
# fetching the 'Age' column
age <- ad_data$`Age`

# computing the frequency distribution using the table() function
age.fr <- table(age)
age.fr
```

Frequency distribution using a barplot

```
## age
## 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44
##  6  6  6 13 19 21 27 37 33 48 48 39 60 38 43 39 39 50 36 37 30 36 32 26 23 21
## 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61
## 30 18 13 16 18 20 12 15 10  9  7  2  6  4  2  4  1
```

```r
# using a barplot to visualize this
barplot(age.fr)
```




```r
# From ages 28-37(the more frequent ages), age 31 is the most frequent
# Age 61 is the least frequent
```

Bivariate analysis

```
# Correlation Coefficient

# It is a normalized measurement of how the two are linearly related. If the correlation coefficient is

# get the columns
daily_usage <- ad_data$`Daily Internet Usage`
clicked <- ad_data$`Clicked on Ad`
site_time <- ad_data$`Daily Time Spent on Site`

# get the correlation coefficients between the variables using cor()
cor(daily_usage , site_time)
```

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

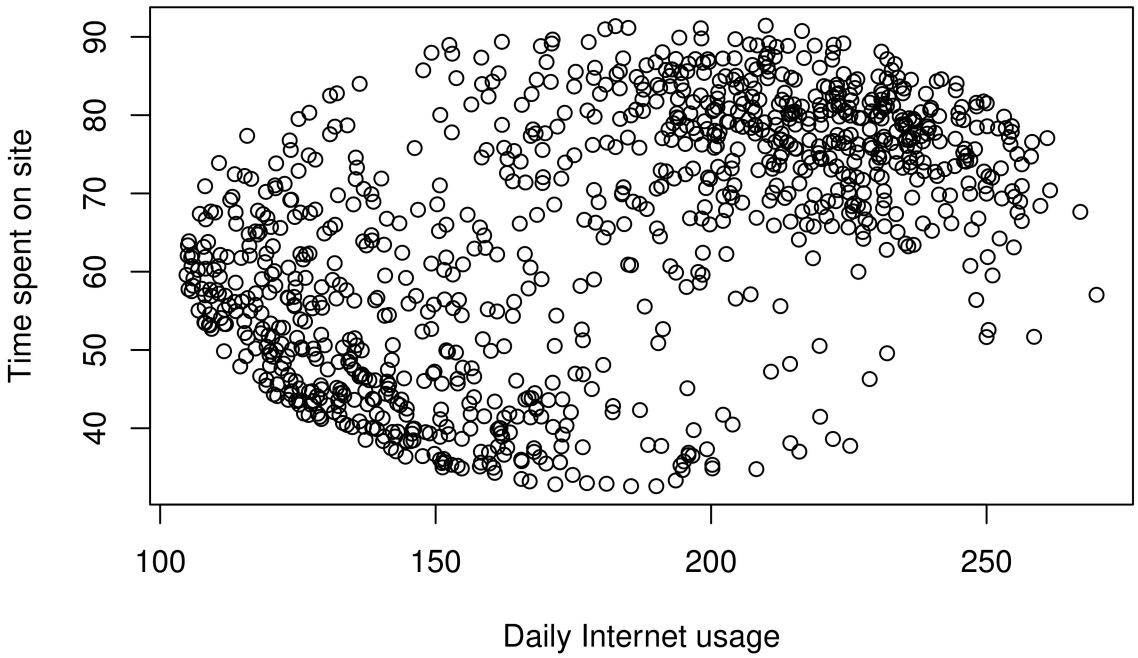
```
# 0.5; indicates a positive correlation although it is a bit weak.
# However, we can see that people who have a higher daily internet usage spend more time on the site

cor(clicked, daily_usage)
```

```
## [1] -0.7865392
```

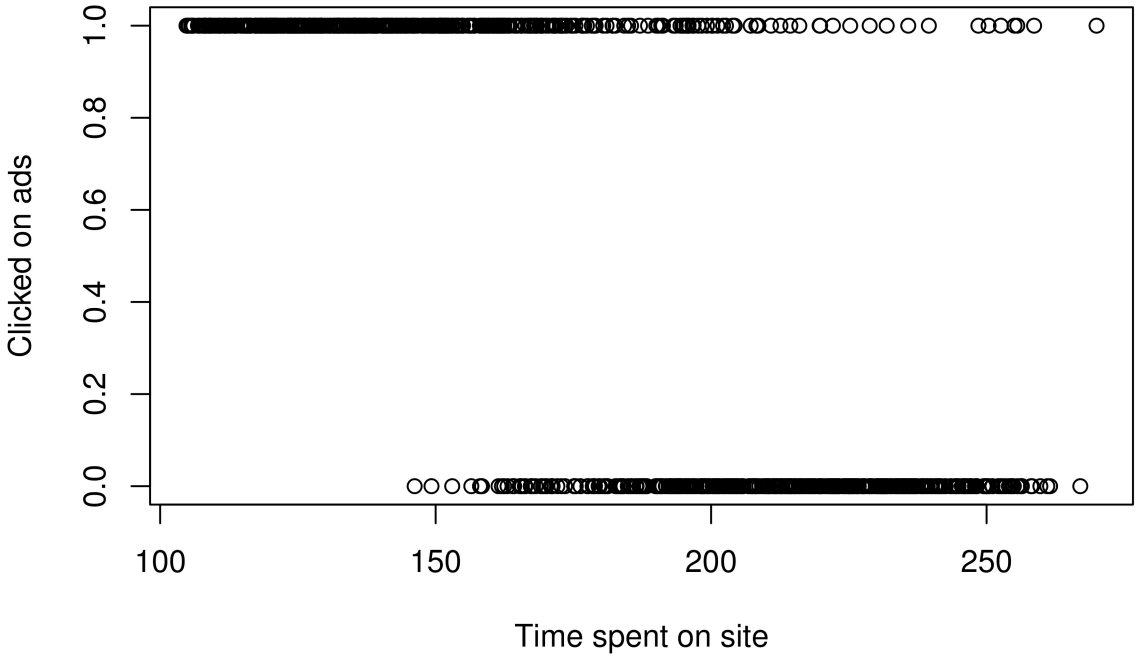
```
# -0.78; a negative correlation which is relatively strong.
# We see that people who spend more time on the site and on the Internet tend to click less on the ads.
```

Visualizations

```
# scatterplots
plot(daily_usage, site_time, xlab="Daily Internet usage", ylab="Time spent on site")
```




```
# The points are quite scattered but if you look closely, you see the points show a weak positive linea
# between the two variables

plot(daily_usage, clicked, xlab="Time spent on site", ylab="Clicked on ads")
```




```
# With how the points are distributed, we realize a rather strong negative correlation between the vari
```

From our bivariate analysis, we see that the users who have higher daily internet usage tend to spend more time on the site. Also, we realised that those users who spend more time on the site, actually tend to click on the ads less. This might be because the ads are not in their interest hence they tend to ignore them. A very brief survey should be taken by the user before using the site so as to tailor the ads to the user's interests which might make them click on them more frequently.

Supervised learning - KNN method

```
# Confirming the dataset
head(ad_data)
```

```
##    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
```

```
str(ad_data)
```

```
## Classes 'data.table' and 'data.frame':   1000 obs. of  10 variables:
##  $ Daily Time Spent on Site: num  69 80.2 69.5 74.2 68.4 ...
##  $ Age                     : int  35 31 26 29 35 23 33 48 30 20 ...
##  $ Area Income             : num  61834 68442 59786 54806 73890 ...
##  $ Daily Internet Usage    : num  256 194 236 246 226 ...
##  $ Ad Topic Line           : chr  "Cloned 5thgeneration orchestration" "Monitored national standardi
##  $ City                    : chr  "Wrightburgh" "West Jodi" "Davidton" "West Terrifurt" ...
##  $ Male                    : int  0 1 0 1 0 1 0 1 1 1 ...
##  $ Country                 : chr  "Tunisia" "Nauru" "San Marino" "Italy" ...
##  $ Timestamp               : POSIXct, format: "2016-03-27 00:53:11" "2016-04-04 01:39:02" ...
##  $ Clicked on Ad           : int  0 0 0 0 0 0 0 1 0 0 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

```
# Getting rid of columns with values that cannot be scaled to fit the model and lack the right features
#ads_new = subset(ad_data, select = -c('Ad Topic Line' ,'Timestamp','Country','City'))
#ads_new
#ads_new <- ad_data[c("Ad Topic Line","Timestamp","Country", "City")]

ads_new = ad_data[,-c(5,6,8,9)]

#Checking the number of remaining columns
ncol(ads_new)
```

```
## [1] 6
```

```
nrow(ads_new)
```

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

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

```
## Daily Time Spent on Site                      Age              Area Income
##                        0                        0                        0
##     Daily Internet Usage                     Male            Clicked on Ad
##                        0                        0                        0
```

```r
ads <- ads_new[1:1000,1:6]
ads
```

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

```r
# Changing the format of the table into factors for it fit the classification modelling appropriately
ads$Clicked.on.Ad <- as.factor(ads$`Clicked on Ad`)
```

```r
# Checking to confirm the data structures on the columns if the conversion underwent successfully
str(ads)
```

```
## Classes 'data.table' and 'data.frame':   1000 obs. of  7 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 ...
##  $ Male                    : int  0 1 0 1 0 1 0 1 1 1 ...
##  $ Clicked on Ad           : int  0 0 0 0 0 0 0 1 0 0 ...
##  $ Clicked.on.Ad           : Factor w/ 2 levels "0","1": 1 1 1 1 1 1 1 2 1 1 ...
##  - attr(*, ".internal.selfref")=<externalptr>
```

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

```
## Daily Time Spent on Site                      Age              Area Income
##                        0                        0                        0
##     Daily Internet Usage                     Male            Clicked on Ad
##                        0                        0                        0
##            Clicked.on.Ad
##                        0
```

```r
ads
```

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

```r
# set seed
set.seed(400)

# The normalization function is created
normalize <- function(x){ (x - min(x))/ (max(x)- min(x))}

# Normalizing the columns by standardizing the values using the normalizing function
ads$Daily.Time.Spent.on.Site<- normalize(ads$Daily.Time.Spent.on.Site)
```

```
## Warning in min(x): no non-missing arguments to min; returning Inf

## Warning in max(x): no non-missing arguments to max; returning -Inf

## Warning in min(x): no non-missing arguments to min; returning Inf
```

```r
ads$Age<- normalize(ads$Age)
ads$Area.Income<- normalize(ads$Area.Income)
```

```
## Warning in min(x): no non-missing arguments to min; returning Inf

## Warning in max(x): no non-missing arguments to max; returning -Inf

## Warning in min(x): no non-missing arguments to min; returning Inf
```

```r
ads$Male<- normalize(ads$Male)
ads$Daily.Internet.Usage<- normalize(ads$Daily.Internet.Usage)
```

```
## Warning in min(x): no non-missing arguments to min; returning Inf

## Warning in max(x): no non-missing arguments to max; returning -Inf

## Warning in min(x): no non-missing arguments to min; returning Inf
```

```r
# Splitting the dataset to train and test samples
tr_ads <- ads[1:750,1:6]
te_ads <- ads[751:1000,6]
tr_ad <- ads[1:750,1:6]
te_ad <- ads[751:1000,6]
```

```r
tr_ad
```

```
##      Daily Time Spent on Site       Age Area Income Daily Internet Usage Male
##   1:                    68.95 0.3809524    61833.90               256.09    0
##   2:                    80.23 0.2857143    68441.85               193.77    1
##   3:                    69.47 0.1666667    59785.94               236.50    0
##   4:                    74.15 0.2380952    54806.18               245.89    1
##   5:                    68.37 0.3809524    73889.99               225.58    0
##  ---
## 746:                    57.99 0.7380952    62466.10               124.58    0
## 747:                    66.80 0.2380952    59683.16               248.51    0
## 748:                    49.13 0.3095238    41097.17               120.49    0
## 749:                    45.11 0.9285714    39799.73               195.69    0
## 750:                    54.35 0.5476190    76984.21               164.02    0
##      Clicked on Ad
##   1:             0
##   2:             0
##   3:             0
##   4:             0
##   5:             0
##  ---
## 746:             1
## 747:             1
## 748:             1
## 749:             1
## 750:             0
```

```r
# tr_df <- as.data.frame(train_ads)
# tr2_df <- as.data.frame(train_ad)
# te_df <- as.data.frame(test_ads)
# te2_df <- as.data.frame(test_ad)
# ```
# ```{r}
# class(tr_df)
```

```
# library(class)
# #require(class)
#
# model_ads <- knn(train = tr_df, test = te_df,cl = tr2_df,k = 13)
#table(factor(model_ads))
#table(te2_df, model_ads)
```

```
# length(tr2_df)
# length(tr_df)
```

```
#class(test_ad)
```

```
# train_ads2 <- train_ads[1:750,1:6]
# test_ads2 <- test_ads[751:1000,1:6]
# train_ad2 <- train_ad[1:750, 1:6]
# test_ad2 <- test_ad[751:1000, 1:6]
```

```
# train_ads3 <- train_ads[,1:6]
# train_ads3
```

```
# train_ad3 <- train_ad[,1:6]
# train_ad3
```

```
# test_ad3 <- test_ad[,6]
# test_ad3
```

```
# test_ads3 <- test_ads[,6]
# test_ads3
```

```
# Instantiating the model and fitting it onto our dataset
#library(class)
#require(class)
# model_ads <- knn(train = train_ads3, test = test_ads3,cl = train_ad3,k = 13)
# table(factor(model_ads))
# table(test_ad3, model_ads)

#model_ads <- knn(train = train_ads3, test = prc_test,cl = prc_train_labels, k=10)
```

Findings: For the 0 values, 110 values were correctly predicted while 10 were incorrectly predicted For the 1 values, 129 values were correctly predicted while 1 was incorrectly predicted This indicates that our KNN model is quite accurate.

```
# Running the knn function
# pred_ad <- knn(train_ads,test_ads, cl = train_ad, k=13)
# pred_ad
# ```
# ```{r}
# pred_ads <- table(pred_ad, test_ad)
# pred_ads
```

Findings: For the 0 values, 110 were correctly predicted while 1 was incorrectly predicted For the 1 values, 129 were correctly predicted while 10 were incorrectly predicted

```
# Checking the percentage accuracy of our model

# accuracy <- function(x){sum(diag(x)/(sum(rowSums(x)))) * 100}
# accuracy(pred_ads)

# An accuracy score of 95.6%. Not bad. Let us compare this method to the SVM method.
```

SVM

```
# we will make use of the same columns used for the KNN model
# ads_svm = subset(ad_data, select = -c(5,6,8,9) )
# head(ads_svm)
#
# # Creating partitions to split the data set into training and test samples
# svm_train_ads <- createDataPartition(y = ads_svm$Clicked.on.Ad, p = 0.75, list = TRUE)
# training_ads <- ads_new[intrain,]
# testing_ads <- ads_new[-intrain,]
```

```
# # Creating the appropriate dimensional sizes for the model
# dim(training_ads);
# dim(testing_ads);
```

```
# Changing the format into factors to make the dataset convenient for classification
#training_ads$Clicked.on.Ad <- as.factor(training_ads$Clicked.on.Ad )
```

```
# Optimizing the SVM model
# svm_trctrl <- trainControl (method = "cv", number = 10)
# svm_Linear_ads <- train(Clicked.on.Ad ~ ., data = training_ads, method = "svmLinear", trControl = sum
#
# svm_Linear_ads
```

```
# Running the model
# test_pred_svm <- predict(svm_Linear_ads, newdata = testing_ads)
# test_pred_svm
```

```
# Using confusion matrix to get a detailed report of the outcome
# conf_svm <- confusionMatrix(table(test_pred_svm, testing_ads$Clicked.on.Ad))
# conf_svm
```