Code ▾

R Notebook - Advertising Data - Bryan Honeck

The given dataset has 1,000 records of users with 10 attributes. The classification variable is whether or not they clicked on an advertisement, denoted 0 or 1 in the last column of the data frame. We are trying to predict which users are more likely to click on the advertisements.

Check the head of the data to see what the data looks like.

Hide

```
#original.data <- read.csv("C:/Users/Bryan/Downloads/train.csv")

head(original.data)
```

	id	Timestamp	Daily.Time.Spent.on.Site	...	Area.Income	Daily.Internet.U
	<int>	<fctr>	<dbl>	<int>	<dbl>	
1	1200	2016-01-01 02:52:10	80.67	34	58909.36	2
2	1201	2016-01-01 03:35:35	68.01	25	68357.96	1
3	1202	2016-01-01 05:31:22	80.94	36	60803.00	2
4	1203	2016-01-01 08:27:06	78.77	28	63497.62	2
5	1204	2016-01-01 15:14:24	36.56	29	42838.29	1
6	1205	2016-01-01 20:17:49	55.79	36	NA	1

6 rows | 1-7 of 10 columns

Hide

```
str(original.data)
```

```
'data.frame':	1000 obs. of  10 variables:
 $ id                      : int  1200 1201 1202 1203 1204 1205 1206 1207 1208 1209 ...
 $ Timestamp               : Factor w/ 775 levels "2016-01-01 02:52:10",..: 1 2 3 4 5 6 6 7 8 9
...
 $ Daily.Time.Spent.on.Site: num  80.7 68 80.9 78.8 36.6 ...
 $ Age                     : int  34 25 36 28 29 36 42 28 33 59 ...
 $ Area.Income             : num  58909 68358 60803 63498 42838 ...
 $ Daily.Internet.Usage    : num  240 188 240 212 196 ...
 $ Ad.Topic.Line           : Factor w/ 825 levels "Adaptive 24hour Graphic Interface",..: 646 15
662 566 724 388 558 618 107 199 ...
 $ gender                  : int  0 1 0 0 0 0 0 1 1 1 ...
 $ Country                 : Factor w/ 233 levels "Afghanistan",..: 167 1 24 83 217 138 64 132 7
4 88 ...
 $ Clicked                 : int  0 0 0 0 1 1 1 0 0 1 ...
```

Let’s check to see if there are any N/A values in the data set.

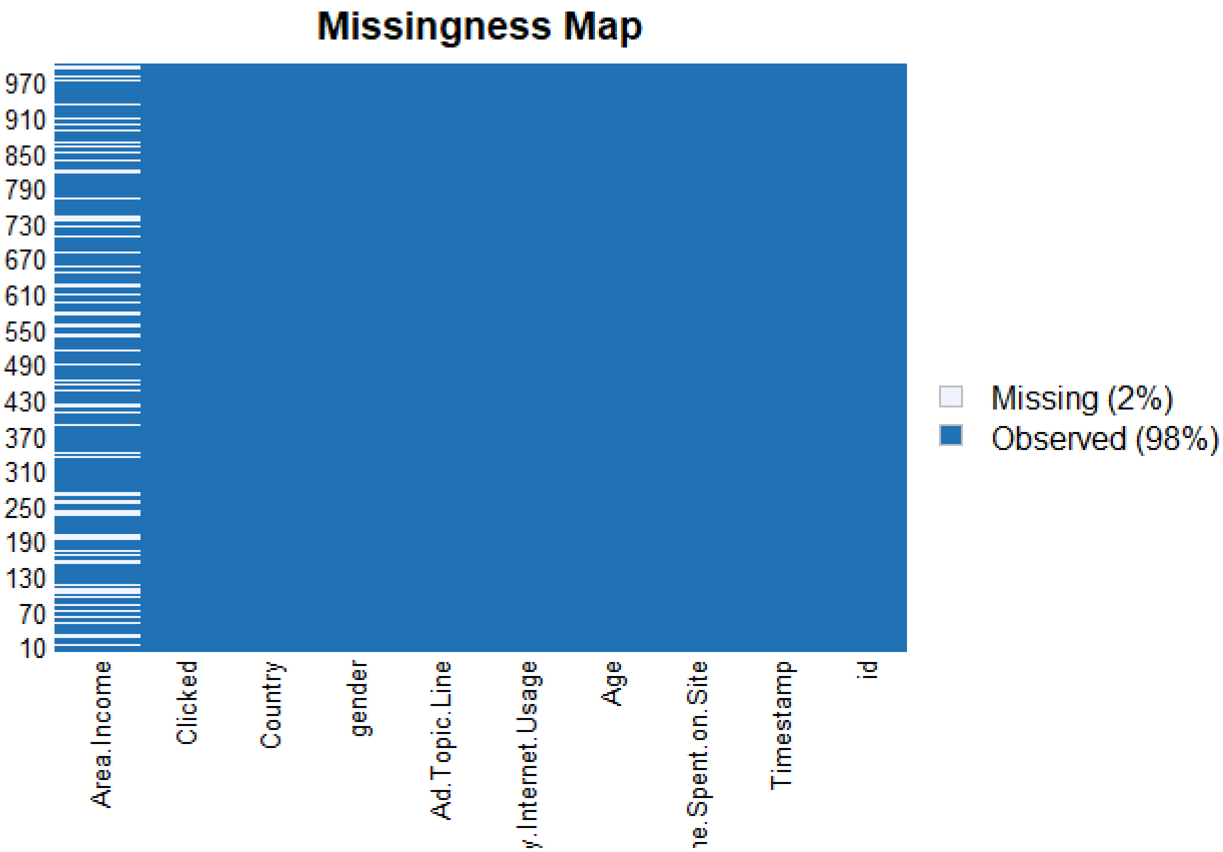
Hide

```
any(is.na(original.data))
```

```
[1] TRUE
```

Hide

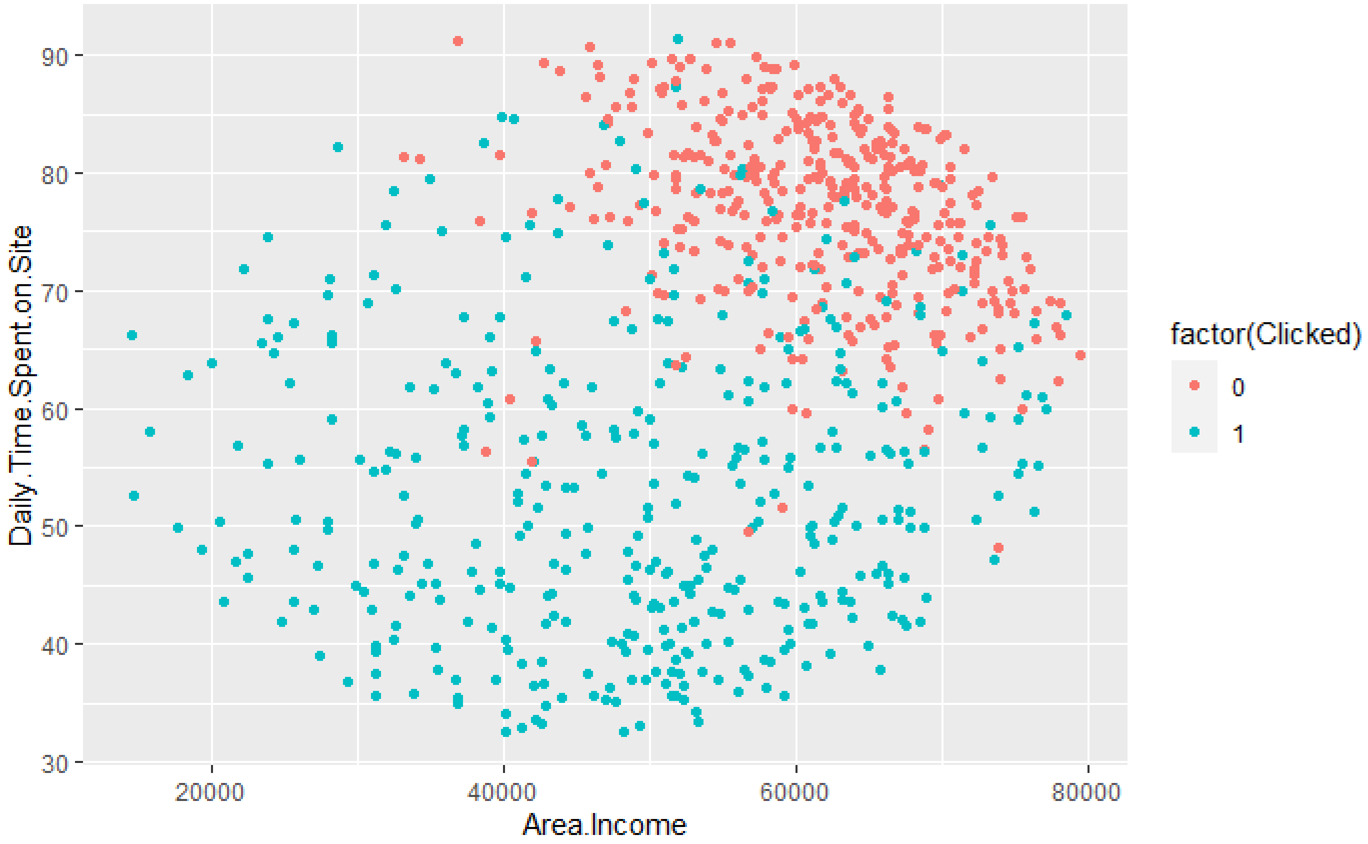
```
library(Amelia)

missmap(original.data)
```



According to the missmap from the Amelia library, Area Income appears to be the only one missing some data. Let’s start visualizing the data first to see if Area Income has a major impact on whether or not someone clicked on the ads. Suppose it does; we will write a funciton that will impute age as accurately as possible.

Hide

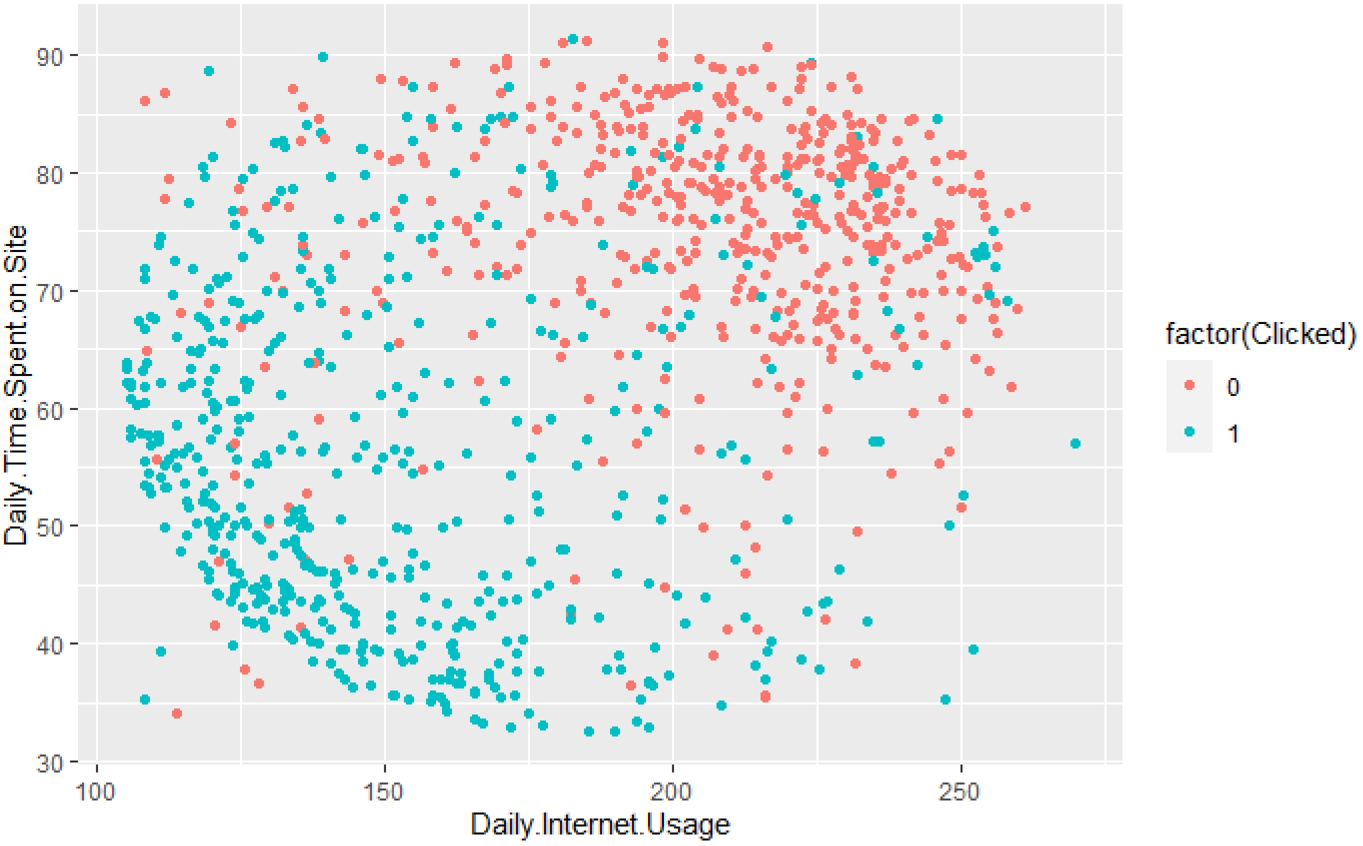
```
library(ggplot2)
pl <- ggplot(original.data, aes(Area.Income, Daily.Time.Spent.on.Site)) + geom_point(aes(color =
factor(Clicked)))
pl
```



Notice that income does not appear to play a major role in whether or not people clicked on the ads. People of all incomes, according to this plot, click on the ads. The more important factor here appears to be the daily time spent on the site. Let’s examine another variable similar to this one that resides within the data frame.

Hide

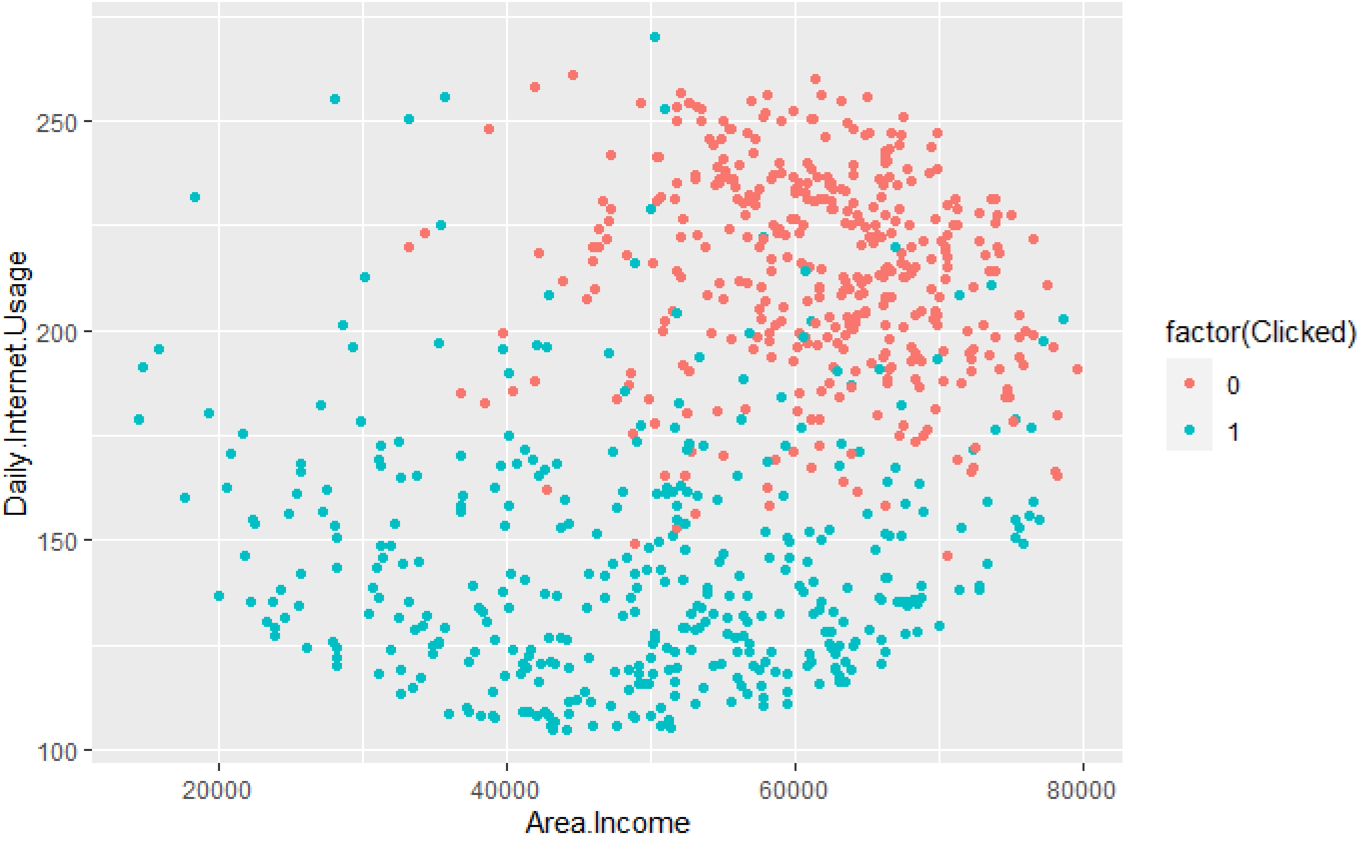
```
pl2 <- ggplot(original.data, aes(Daily.Internet.Usage, Daily.Time.Spent.on.Site)) + geom_point(a
es(color = factor(Clicked)))
pl2
```



There is still a significant amount of clustering here, but in my opinion, it appears to be a tighter cluster in terms of the Area Income. Let’s examine Daily Internet Usage mapped by Area Income.

Hide

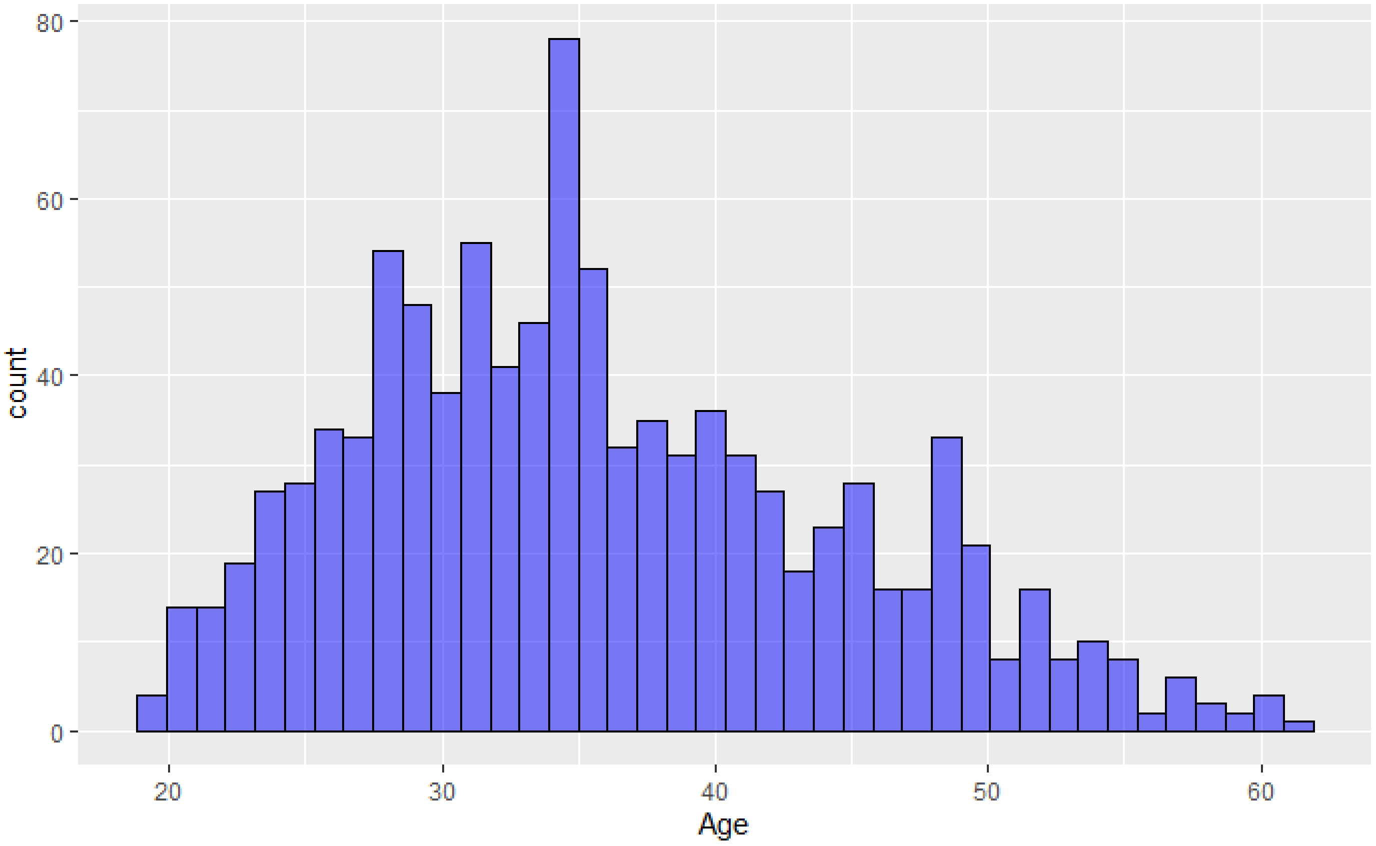
```
pl3 <- ggplot(original.data, aes(Area.Income, Daily.Internet.Usage)) + geom_point(aes(color = fa
ctor(Clicked)))
pl3
```



This plot looks pretty similar to the first one we examined. Given these two variables alone, we have a pretty good idea of whether or not a user clicked on the ads. Let's do more EDA then begin working on training our model.

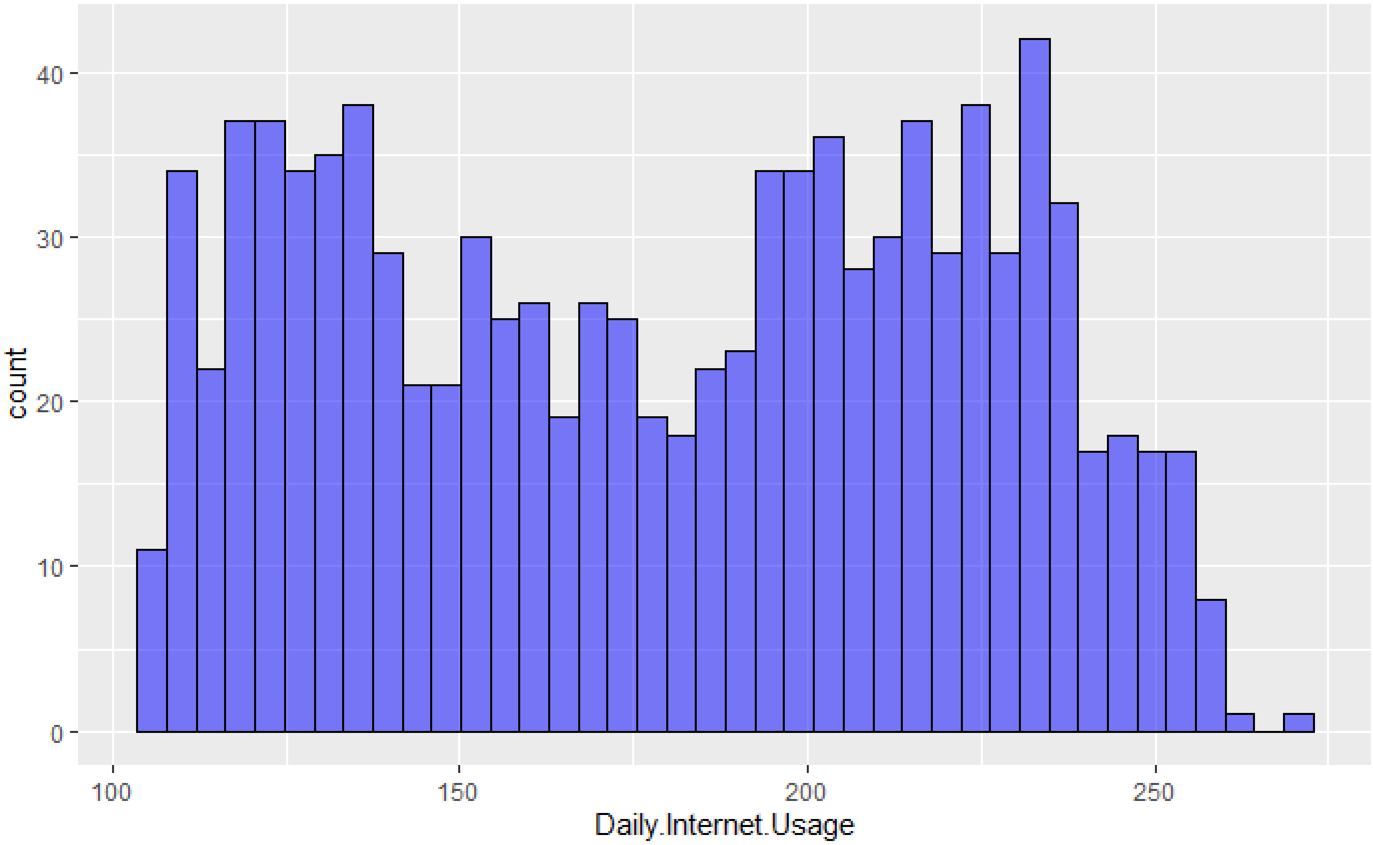
Hide

```
library(plotly)

pl4 <- ggplot(original.data, aes(Age)) + geom_histogram(fill = "blue", color = "black", alpha =
0.5, bins = 40)
pl4
```



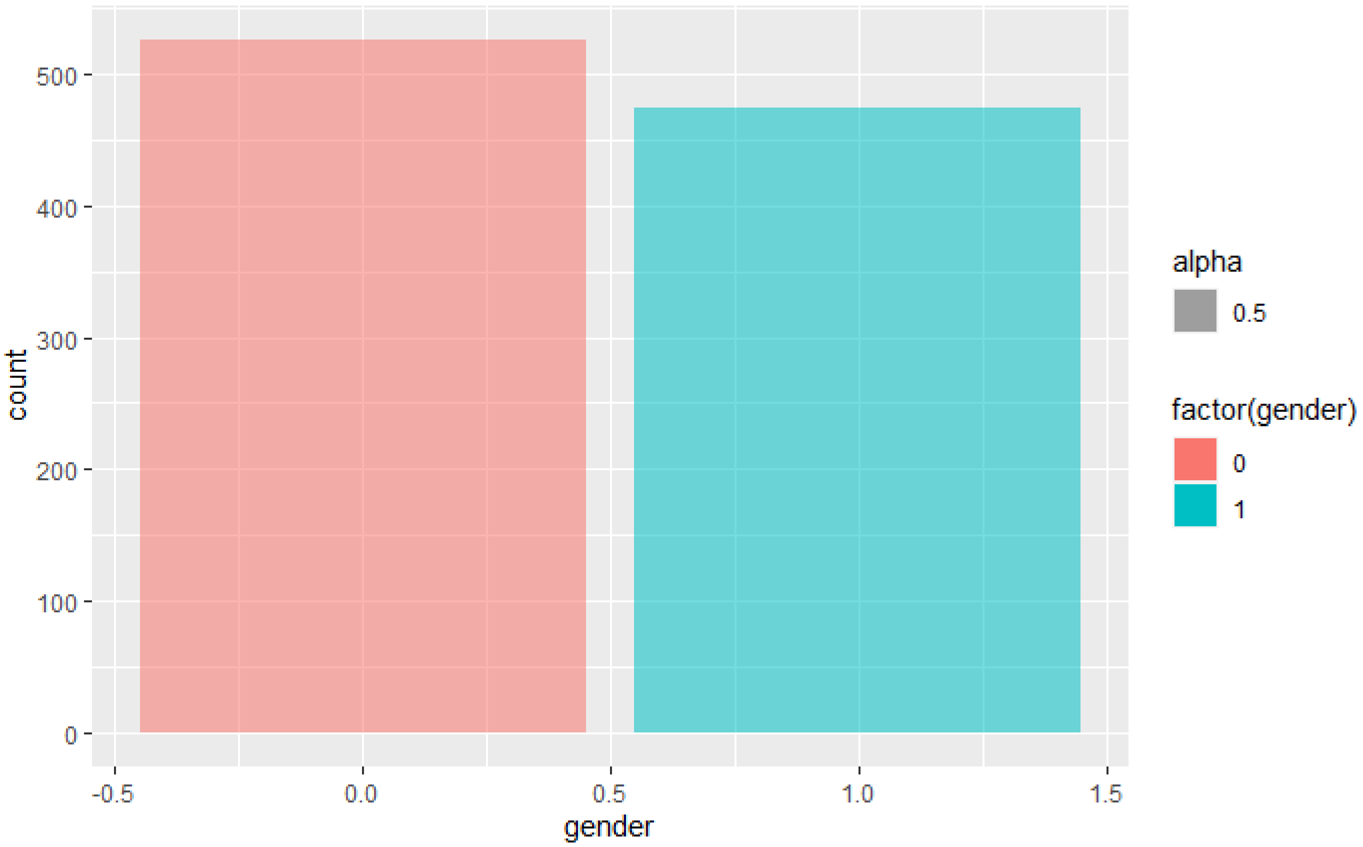
Hide

```
pl5 <- ggplot(original.data, aes(Daily.Internet.Usage)) + geom_histogram(fill = "blue", color =
"black", alpha = 0.5, bins = 40)
pl5
```



Hide

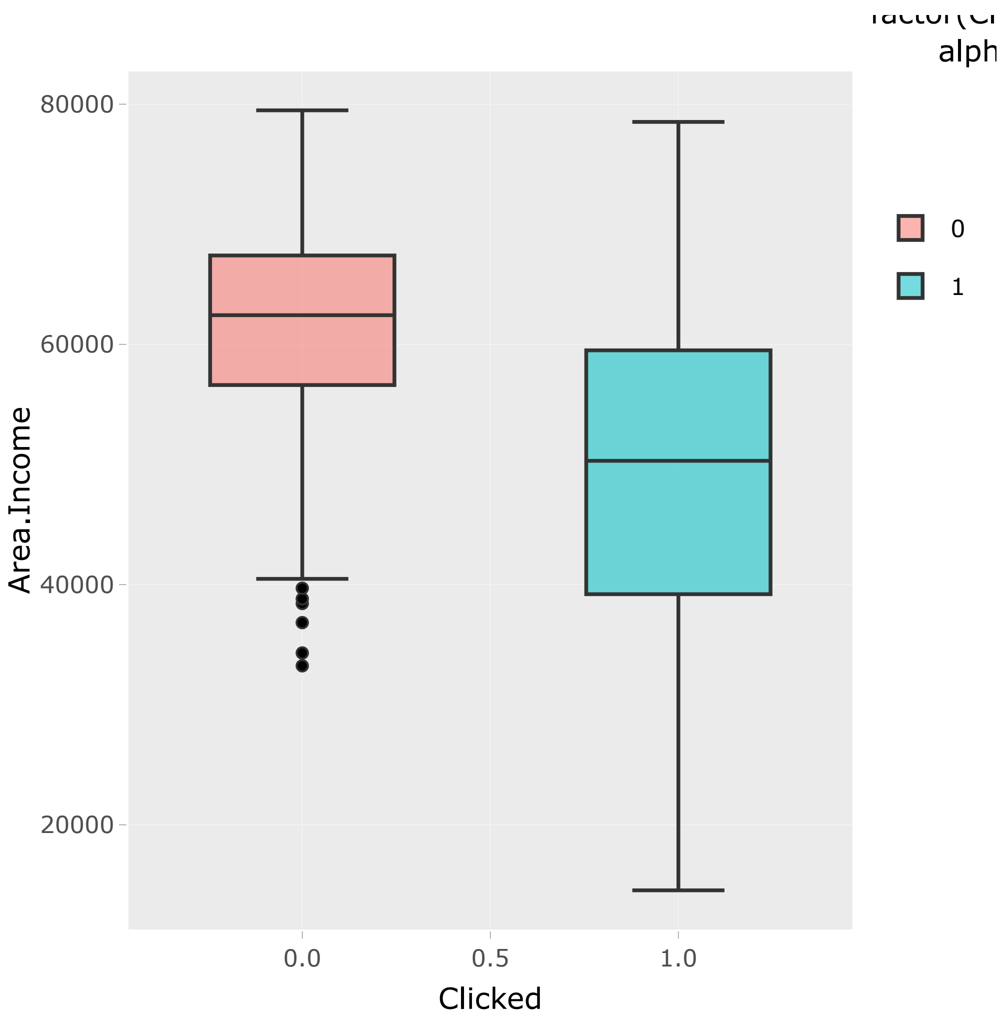
```
pl6 <- ggplot(original.data, aes(gender)) + geom_bar(aes(fill = factor(gender), alpha = 0.5))
pl6
```



Hide

```
pl7 <- ggplot(original.data, aes(Clicked, Area.Income)) + geom_boxplot(aes(group=Clicked, fill=f
actor(Clicked), alpha = 0.4))
ggplotly(pl7)
```

```
Removed 225 rows containing non-finite values (stat_boxplot).
```



We see from the plots above some very interesting information. The people that clicked on the ads spend less time on the internet in general and on the site each day. They also tend to have lower incomes, per the boxplot. Let’s write a function to impute Area.Income into the missing spots.

Using plotly, we can easily get the median Area Income to impute for each group: 62,430.55 where Clicked=0; 50,306.31 where Clicked=1.

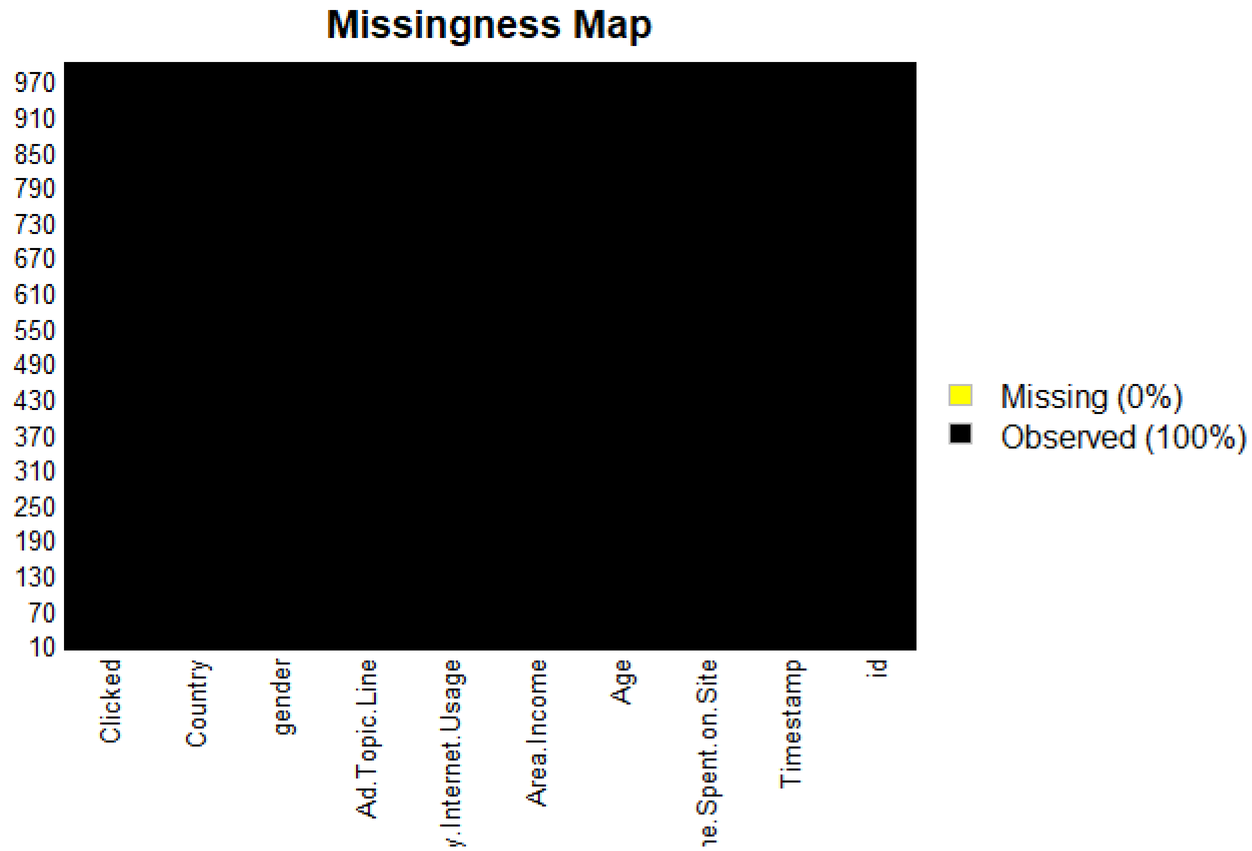
Hide

```
 impute_income <- function(income, clicked) {
   out <- income
   for (i in 1:length(income)){
     if(is.na(income[i]))
    {
       if(clicked[i] == 0)
         {
            out[i] <- 62430.55
         }
       else
         {
            out[i] <- 50306.31
         }
    }
     else
    {
       out[i] <- income[i]
    }
   }
   return(out)
}

original.incomes <- original.data$Area.Income

fixed.incomes <- impute_income(original.data$Area.Income, original.data$Clicked)

original.data$Area.Income <- fixed.incomes
```

Let’s check to see that it worked properly!

Hide

```
missmap(original.data, col = c("yellow", "black"))
```



Great! Having no missing data is a good feeling. Now, let’s start running some models. Begin with K-means and the relevant columns.

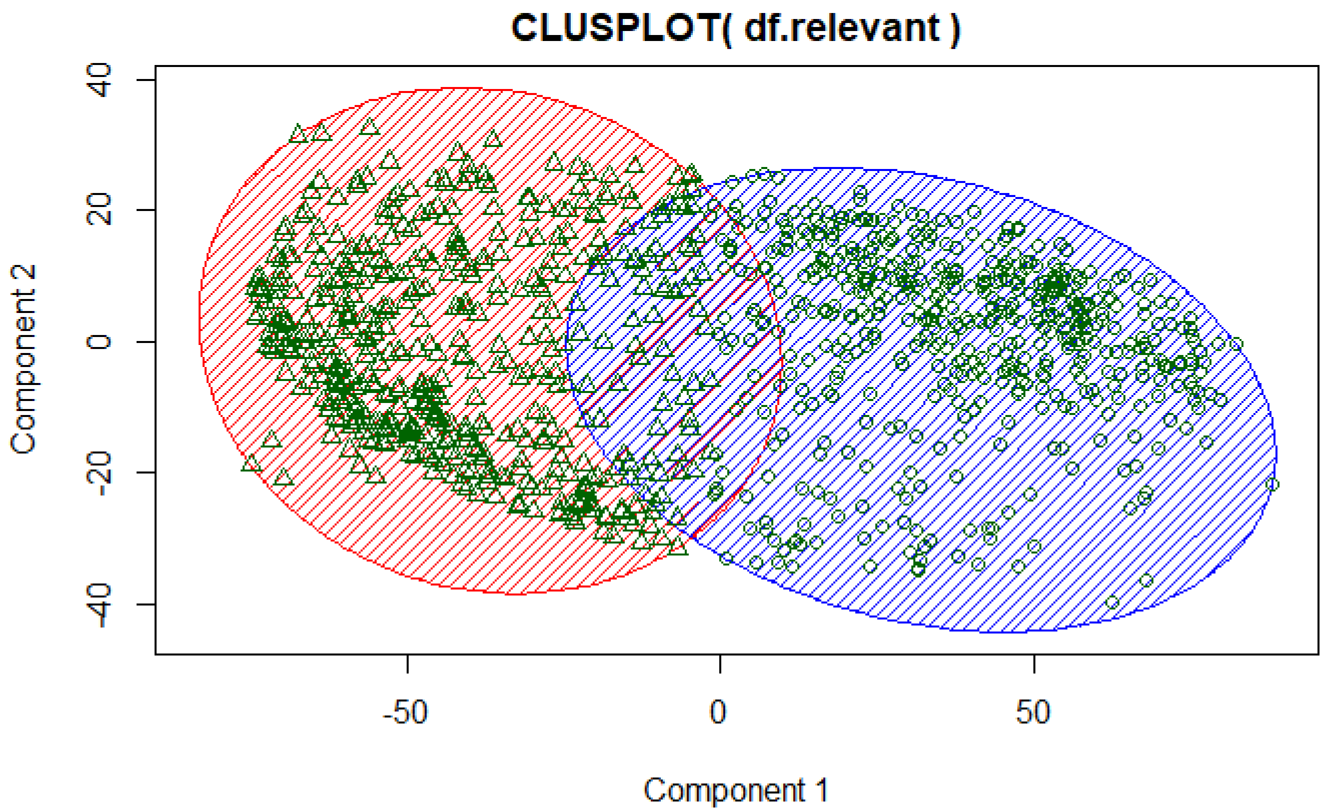
Hide

```
library(cluster)

df.relevant <- data.frame(original.data$Daily.Time.Spent.on.Site, original.data$Daily.Internet.Usage)

cluster.click <- kmeans(df.relevant, 2, nstart = 10)
```

Let’s look at the clusplot!

Hide

```
library(cluster)

clusplot(df.relevant, cluster.click$cluster, color = TRUE, shade = TRUE, labels = 0, lines = 0)
```



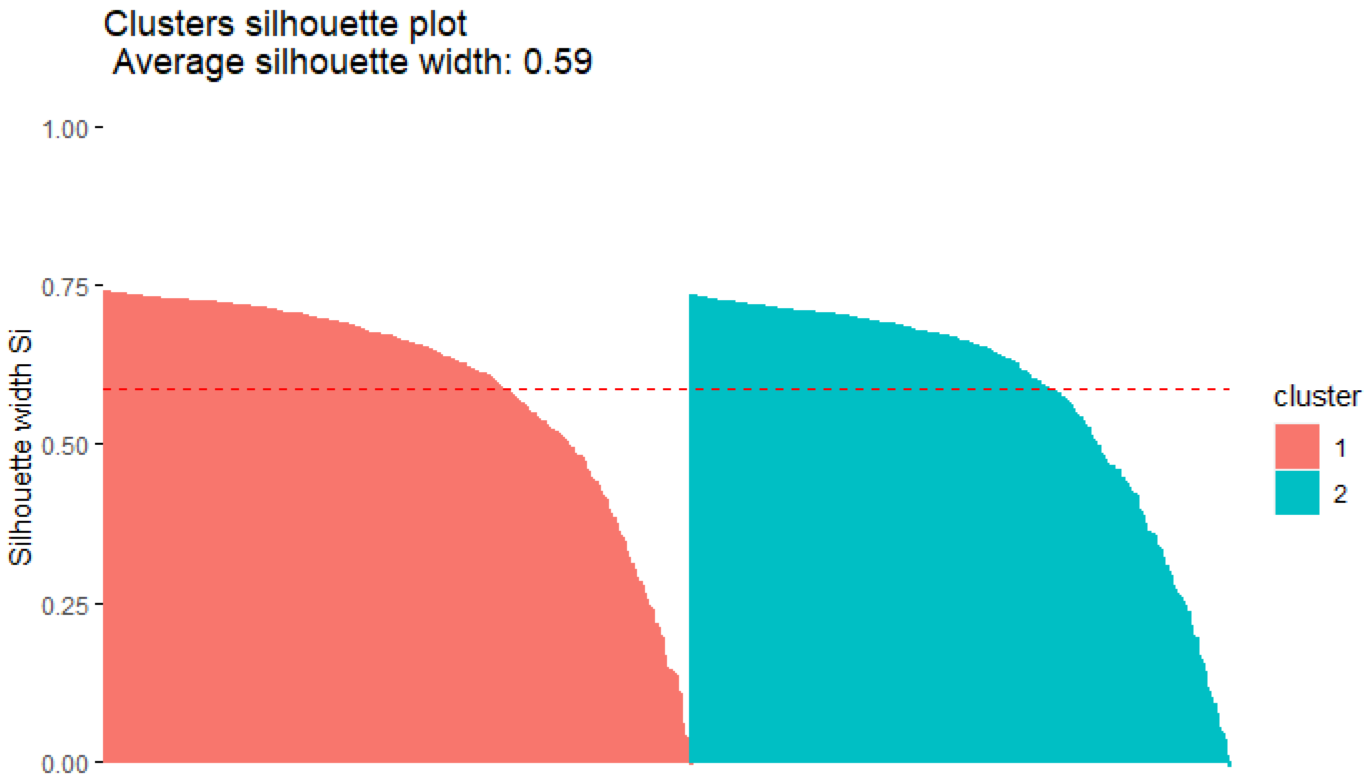
These two components explain 100 % of the point variability.

Hide

```
library(factoextra)

sil <- silhouette(cluster.click$cluster, dist(df.relevant))
fviz_silhouette(sil)
```

	cluster <fctr>	size <int>	ave.sil.width <dbl>
1	1	522	0.59
2	2	478	0.58

2 rows



Above 0.5; looks pretty good!

Let’s start the KNN model:

Hide

```
var(train2[, 1])
```

```
[1] 83416.67
```

Hide

```
var(train2[, 3])
```

```
[1] 249.0543
```

Hide

```
clicked <- train2[, 10]

knn.df <- data.frame(train2$Daily.Time.Spent.on.Site, train2$Daily.Internet.Usage)

#CBIND the classification column
knn.df <- cbind(knn.df, clicked)

knn.standardized <- scale(knn.df[, -3])


var(knn.standardized[, 1])
```

```
[1] 1
```

Hide

```
var(knn.standardized[, 2])
```

```
[1] 1
```

Hide

```
#Test - first 300 rows for test set
test.index <- 1:300
test.data <- knn.standardized[test.index, ]
test.clicked <- clicked[test.index]

#Train
train.data <- knn.standardized[-test.index, ]
train.clicked <- clicked[-test.index]
```

Hide

```
#Run the model
library(class)

predictions.clicked <- knn(train.data, test.data, train.clicked, k=3)

mean(test.clicked != predictions.clicked)
```

```
[1] 0.1633333
```

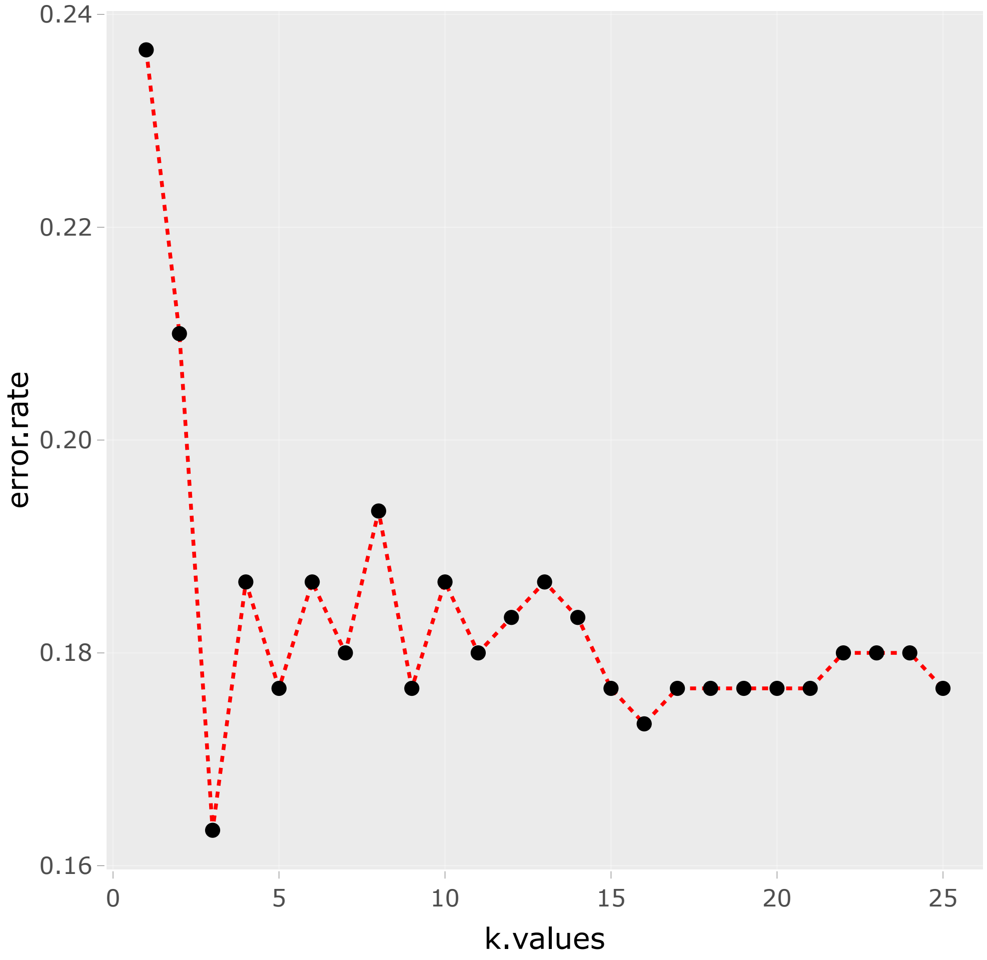
Let’s observe the model with other k-values

Hide

```
predictions.clicked <- NULL
error.rate <- NULL

for (i in 1:25) {
  predictions.clicked <- knn(train.data, test.data, train.clicked, k=i)
  error.rate[i] <- mean(test.clicked != predictions.clicked)
}
```

Hide

```
k.values <- 1:25

error.df <- data.frame(error.rate, k.values)

pl8 <- ggplot(error.df, aes(k.values, error.rate)) + geom_point() + geom_line(lty="dotted", colo
r = "red")
ggplotly(pl8)
```



So, the model runs most accurately with k=3; so that is what we will choose.

Hide

```
final.predictions.clicked <- knn(train.data, test.data, train.clicked, k=3)

mean(test.clicked != final.predictions.clicked)
```

```
[1] 0.1633333
```

The model is approximately 83.67% accurate in it’s predicitons.