Multiple Linear Regression on Customer Data

Insofe Lab Session

July 02 , 2017

# Go through the data and understand the attributes to get the info of the variables.

Write a code to clear the environment if needed

rm(list = ls(all = TRUE))

Write a code to set the working directory.

setwd("C:/Users/Thomas/Desktop/20170702\_Batch30\_CSE7302c\_CustomerData\_Assignment")

Think and load the libraries as and when required in this place only as best practice

library(vegan)

## Loading required package: permute

## Loading required package: lattice

## This is vegan 2.4-3

library(corrplot)

## Warning: package 'corrplot' was built under R version 3.4.1

library(car)

## Warning: package 'car' was built under R version 3.4.1

library(DMwR)

## Loading required package: grid

library(MASS)  
library(e1071)

## Warning: package 'e1071' was built under R version 3.4.1

Write a code to read the csv file as "data" as per required

customer.data = read.csv(file = "20170702\_Batch30\_CSE7302c\_CustomerData\_Assignment.csv", header = TRUE)

Write a code to know the names of the attributes

names(customer.data)

## [1] "CustomerID" "City"   
## [3] "NoOfChildren" "MinAgeOfChild"   
## [5] "MaxAgeOfChild" "Tenure"   
## [7] "FrquncyOfPurchase" "NoOfUnitsPurchased"   
## [9] "FrequencyOFPlay" "NoOfGamesPlayed"   
## [11] "NoOfGamesBought" "FavoriteChannelOfTransaction"  
## [13] "FavoriteGame" "TotalRevenueGenerated"

Write a code to find the dimensions of the data

dim(customer.data)

## [1] 3209 14

Write a code to see the head and tail of the dataset atleast 10rows

head(x = customer.data, n = 10L)

## CustomerID City NoOfChildren MinAgeOfChild MaxAgeOfChild Tenure  
## 1 1001 1 2 3 8 210  
## 2 1002 1 2 3 6 442  
## 3 1003 1 4 3 5 424  
## 4 1004 1 1 6 6 261  
## 5 1005 1 3 6 9 422  
## 6 1006 1 2 3 4 378  
## 7 1007 1 3 8 12 369  
## 8 1008 1 2 6 8 404  
## 9 1009 1 4 6 9 420  
## 10 1010 2 3 5 6 333  
## FrquncyOfPurchase NoOfUnitsPurchased FrequencyOFPlay NoOfGamesPlayed  
## 1 11 11 2344 108  
## 2 20 20 245 22  
## 3 18 18 1059 130  
## 4 11 9 365 34  
## 5 44 31 1066 102  
## 6 16 16 228 12  
## 7 25 15 75 2  
## 8 13 12 1488 118  
## 9 20 16 2743 163  
## 10 15 15 1967 56  
## NoOfGamesBought FavoriteChannelOfTransaction FavoriteGame  
## 1 10 Uniform Uniform  
## 2 7 Favorite Uniform  
## 3 18 Favorite Uniform  
## 4 11 Favorite Uniform  
## 5 44 Uniform Uniform  
## 6 16 Favorite Favorite  
## 7 25 Favorite Favorite  
## 8 13 Favorite Uniform  
## 9 16 Uniform Uniform  
## 10 15 Favorite Uniform  
## TotalRevenueGenerated  
## 1 107.51  
## 2 382.40  
## 3 135.01  
## 4 125.00  
## 5 335.05  
## 6 150.00  
## 7 127.50  
## 8 122.50  
## 9 164.96  
## 10 112.62

tail(x = customer.data, n = 10L)

## CustomerID City NoOfChildren MinAgeOfChild MaxAgeOfChild Tenure  
## 3200 4200 1 1 6 6 352  
## 3201 4201 1 2 7 8 313  
## 3202 4202 1 3 2 6 424  
## 3203 4203 1 2 5 7 424  
## 3204 4204 1 1 4 4 431  
## 3205 4205 1 2 4 6 365  
## 3206 4206 1 2 2 5 348  
## 3207 4207 1 2 5 7 341  
## 3208 4208 1 2 6 7 368  
## 3209 4209 1 3 2 6 389  
## FrquncyOfPurchase NoOfUnitsPurchased FrequencyOFPlay NoOfGamesPlayed  
## 3200 27 24 2042 194  
## 3201 15 15 952 113  
## 3202 20 12 2038 177  
## 3203 32 29 5933 382  
## 3204 26 26 1166 72  
## 3205 16 16 827 78  
## 3206 20 20 2933 294  
## 3207 16 13 1250 126  
## 3208 18 14 1364 122  
## 3209 25 17 475 111  
## NoOfGamesBought FavoriteChannelOfTransaction FavoriteGame  
## 3200 23 Favorite Uniform  
## 3201 10 Favorite Uniform  
## 3202 20 Uniform Uniform  
## 3203 29 Favorite Uniform  
## 3204 26 Favorite Uniform  
## 3205 16 Favorite Uniform  
## 3206 13 Favorite Uniform  
## 3207 16 Uniform Uniform  
## 3208 18 Favorite Uniform  
## 3209 25 Favorite Uniform  
## TotalRevenueGenerated  
## 3200 249.96  
## 3201 175.51  
## 3202 117.26  
## 3203 261.29  
## 3204 205.00  
## 3205 132.50  
## 3206 223.23  
## 3207 120.00  
## 3208 161.50  
## 3209 137.50

Write a code to see the data types of the attributes

str(customer.data)

## 'data.frame': 3209 obs. of 14 variables:  
## $ CustomerID : int 1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 ...  
## $ City : int 1 1 1 1 1 1 1 1 1 2 ...  
## $ NoOfChildren : int 2 2 4 1 3 2 3 2 4 3 ...  
## $ MinAgeOfChild : int 3 3 3 6 6 3 8 6 6 5 ...  
## $ MaxAgeOfChild : int 8 6 5 6 9 4 12 8 9 6 ...  
## $ Tenure : int 210 442 424 261 422 378 369 404 420 333 ...  
## $ FrquncyOfPurchase : int 11 20 18 11 44 16 25 13 20 15 ...  
## $ NoOfUnitsPurchased : int 11 20 18 9 31 16 15 12 16 15 ...  
## $ FrequencyOFPlay : int 2344 245 1059 365 1066 228 75 1488 2743 1967 ...  
## $ NoOfGamesPlayed : int 108 22 130 34 102 12 2 118 163 56 ...  
## $ NoOfGamesBought : int 10 7 18 11 44 16 25 13 16 15 ...  
## $ FavoriteChannelOfTransaction: Factor w/ 2 levels "Favorite","Uniform": 2 1 1 1 2 1 1 1 2 1 ...  
## $ FavoriteGame : Factor w/ 3 levels "Favorite","NONE",..: 3 3 3 3 3 1 1 3 3 3 ...  
## $ TotalRevenueGenerated : num 108 382 135 125 335 ...

Write a code to remove the unuseful variables and store the data

customer.data.withoutCustomerId = customer.data  
customer.data.withoutCustomerId$CustomerID = NULL

Write a code to observe the variables and convert them into the required formats

summary(customer.data.withoutCustomerId)

## City NoOfChildren MinAgeOfChild MaxAgeOfChild   
## Min. :1.000 Min. : 1.000 Min. : 0.000 Min. : 3.000   
## 1st Qu.:1.000 1st Qu.: 1.000 1st Qu.: 4.000 1st Qu.: 6.000   
## Median :1.000 Median : 2.000 Median : 5.000 Median : 7.000   
## Mean :1.114 Mean : 2.128 Mean : 4.961 Mean : 7.991   
## 3rd Qu.:1.000 3rd Qu.: 3.000 3rd Qu.: 6.000 3rd Qu.: 8.000   
## Max. :2.000 Max. :11.000 Max. :113.000 Max. :113.000   
## Tenure FrquncyOfPurchase NoOfUnitsPurchased FrequencyOFPlay  
## Min. :100.0 Min. : 1.00 Min. : 1.00 Min. : 0   
## 1st Qu.:301.0 1st Qu.: 11.00 1st Qu.: 10.00 1st Qu.: 446   
## Median :368.0 Median : 14.00 Median : 13.00 Median : 1029   
## Mean :347.5 Mean : 16.27 Mean : 14.68 Mean : 1568   
## 3rd Qu.:417.0 3rd Qu.: 19.00 3rd Qu.: 17.00 3rd Qu.: 2029   
## Max. :472.0 Max. :119.00 Max. :112.00 Max. :27829   
## NoOfGamesPlayed NoOfGamesBought FavoriteChannelOfTransaction  
## Min. : 0.00 Min. : 0.00 Favorite:2637   
## 1st Qu.: 37.00 1st Qu.: 10.00 Uniform : 572   
## Median : 70.00 Median : 14.00   
## Mean : 93.63 Mean : 14.76   
## 3rd Qu.: 119.00 3rd Qu.: 19.00   
## Max. :1166.00 Max. :115.00   
## FavoriteGame TotalRevenueGenerated  
## Favorite: 126 Min. :100.0   
## NONE : 30 1st Qu.:116.6   
## Uniform :3053 Median :142.4   
## Mean :168.5   
## 3rd Qu.:191.2   
## Max. :990.6

# Code to remove the outliers (rows)   
customer.data.withoutCustomerId = customer.data.withoutCustomerId[(customer.data.withoutCustomerId$FrequencyOFPlay < 10000),]  
customer.data.withoutCustomerId = customer.data.withoutCustomerId[(customer.data.withoutCustomerId$NoOfChildren <= 10),]  
customer.data.withoutCustomerId = customer.data.withoutCustomerId[(customer.data.withoutCustomerId$MinAgeOfChild < 15),]  
customer.data.withoutCustomerId = customer.data.withoutCustomerId[customer.data.withoutCustomerId$NoOfGamesBought < 100,]  
customer.data.withoutCustomerId = customer.data.withoutCustomerId[customer.data.withoutCustomerId$NoOfGamesPlayed < 500,]  
customer.data.withoutCustomerId = customer.data.withoutCustomerId[customer.data.withoutCustomerId$NoOfUnitsPurchased < 50,]  
customer.data.withoutCustomerId = customer.data.withoutCustomerId[customer.data.withoutCustomerId$FrquncyOfPurchase < 50,]  
  
#Removing some wrong values in the data and applying KNNImputation to these values  
# Age of children can never be greater than 100  
customer.data.withoutCustomerId[customer.data.withoutCustomerId$MinAgeOfChild > 100,"MinAgeOfChild"] = NA  
customer.data.withoutCustomerId[customer.data.withoutCustomerId$MaxAgeOfChild > 100,"MaxAgeOfChild"] = NA  
# Library DMwR  
customer.data.withoutCustomerId= knnImputation(data = customer.data.withoutCustomerId, k = 5)  
  
summary(customer.data.withoutCustomerId)

## City NoOfChildren MinAgeOfChild MaxAgeOfChild   
## Min. :1.000 Min. :1.000 Min. : 0.00 Min. : 3.000   
## 1st Qu.:1.000 1st Qu.:1.000 1st Qu.: 4.00 1st Qu.: 6.000   
## Median :1.000 Median :2.000 Median : 5.00 Median : 7.000   
## Mean :1.114 Mean :2.117 Mean : 4.85 Mean : 7.324   
## 3rd Qu.:1.000 3rd Qu.:3.000 3rd Qu.: 6.00 3rd Qu.: 8.000   
## Max. :2.000 Max. :9.000 Max. :12.00 Max. :23.000   
## Tenure FrquncyOfPurchase NoOfUnitsPurchased FrequencyOFPlay  
## Min. :100.0 Min. : 1.00 Min. : 1.0 Min. : 0   
## 1st Qu.:299.5 1st Qu.:11.00 1st Qu.:10.0 1st Qu.: 444   
## Median :367.0 Median :14.00 Median :13.0 Median :1013   
## Mean :346.6 Mean :15.85 Mean :14.3 Mean :1468   
## 3rd Qu.:416.0 3rd Qu.:19.00 3rd Qu.:17.0 3rd Qu.:1942   
## Max. :472.0 Max. :49.00 Max. :47.0 Max. :9697   
## NoOfGamesPlayed NoOfGamesBought FavoriteChannelOfTransaction  
## Min. : 0.00 Min. : 0.00 Favorite:2584   
## 1st Qu.: 37.00 1st Qu.:10.00 Uniform : 567   
## Median : 69.00 Median :13.00   
## Mean : 88.78 Mean :14.36   
## 3rd Qu.:117.00 3rd Qu.:18.00   
## Max. :486.00 Max. :49.00   
## FavoriteGame TotalRevenueGenerated  
## Favorite: 126 Min. :100.0   
## NONE : 30 1st Qu.:115.9   
## Uniform :2995 Median :140.5   
## Mean :164.5   
## 3rd Qu.:189.3   
## Max. :771.0

# To convert the vaiables into Factors  
categorical.attributes = c("City", "FavoriteChannelOfTransaction", "FavoriteGame")  
customer.data.withoutCustomerId[,categorical.attributes] = data.frame(apply(customer.data.withoutCustomerId[,categorical.attributes],2,function(x) as.factor(as.character(x))))

Check the changes again using str() command

str(customer.data.withoutCustomerId)

## 'data.frame': 3151 obs. of 13 variables:  
## $ City : Factor w/ 2 levels "1","2": 1 1 1 1 1 1 1 1 1 2 ...  
## $ NoOfChildren : int 2 2 4 1 3 2 3 2 4 3 ...  
## $ MinAgeOfChild : int 3 3 3 6 6 3 8 6 6 5 ...  
## $ MaxAgeOfChild : num 8 6 5 6 9 4 12 8 9 6 ...  
## $ Tenure : int 210 442 424 261 422 378 369 404 420 333 ...  
## $ FrquncyOfPurchase : int 11 20 18 11 44 16 25 13 20 15 ...  
## $ NoOfUnitsPurchased : int 11 20 18 9 31 16 15 12 16 15 ...  
## $ FrequencyOFPlay : int 2344 245 1059 365 1066 228 75 1488 2743 1967 ...  
## $ NoOfGamesPlayed : int 108 22 130 34 102 12 2 118 163 56 ...  
## $ NoOfGamesBought : int 10 7 18 11 44 16 25 13 16 15 ...  
## $ FavoriteChannelOfTransaction: Factor w/ 2 levels "Favorite","Uniform": 2 1 1 1 2 1 1 1 2 1 ...  
## $ FavoriteGame : Factor w/ 3 levels "Favorite","NONE",..: 3 3 3 3 3 1 1 3 3 3 ...  
## $ TotalRevenueGenerated : num 108 382 135 125 335 ...

Write a Code to seperate the target

TotalRevenueGenerated = customer.data.withoutCustomerId$TotalRevenueGenerated

Write a code to get the numerical attributes out and store them as data\_num

numerical.attributes = c("NoOfChildren", "MinAgeOfChild", "MaxAgeOfChild", "Tenure", "NoOfUnitsPurchased","FrquncyOfPurchase", "FrequencyOFPlay","NoOfGamesPlayed", "NoOfGamesBought")  
data\_num = customer.data.withoutCustomerId[, numerical.attributes]

Write a code for Normalizing the data\_num

# Library vegan  
normalized.data\_num = decostand(data\_num, "normalize", MARGIN = 2)

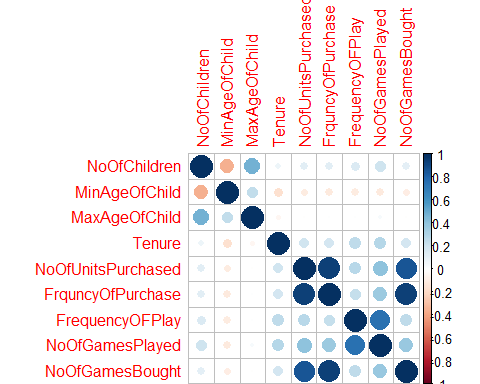
Write a code to observe the correlation between the attributes

r = cor(normalized.data\_num)  
r

## NoOfChildren MinAgeOfChild MaxAgeOfChild Tenure  
## NoOfChildren 1.00000000 -0.35209070 0.470088335 0.07835039  
## MinAgeOfChild -0.35209070 1.00000000 0.245618640 -0.16482138  
## MaxAgeOfChild 0.47008834 0.24561864 1.000000000 -0.04353032  
## Tenure 0.07835039 -0.16482138 -0.043530321 1.00000000  
## NoOfUnitsPurchased 0.12486566 -0.10529212 -0.006674528 0.19250739  
## FrquncyOfPurchase 0.12385047 -0.11134932 0.004613855 0.18763815  
## FrequencyOFPlay 0.15296633 -0.10882065 0.001221403 0.25176963  
## NoOfGamesPlayed 0.20803313 -0.11844208 0.037558498 0.28148710  
## NoOfGamesBought 0.11414624 -0.09336059 -0.006211917 0.18123283  
## NoOfUnitsPurchased FrquncyOfPurchase FrequencyOFPlay  
## NoOfChildren 0.124865657 0.123850472 0.152966332  
## MinAgeOfChild -0.105292123 -0.111349317 -0.108820654  
## MaxAgeOfChild -0.006674528 0.004613855 0.001221403  
## Tenure 0.192507388 0.187638146 0.251769629  
## NoOfUnitsPurchased 1.000000000 0.930601180 0.273227042  
## FrquncyOfPurchase 0.930601180 1.000000000 0.235297288  
## FrequencyOFPlay 0.273227042 0.235297288 1.000000000  
## NoOfGamesPlayed 0.405926095 0.365736690 0.749421207  
## NoOfGamesBought 0.853619169 0.936990990 0.252180001  
## NoOfGamesPlayed NoOfGamesBought  
## NoOfChildren 0.2080331 0.114146237  
## MinAgeOfChild -0.1184421 -0.093360593  
## MaxAgeOfChild 0.0375585 -0.006211917  
## Tenure 0.2814871 0.181232834  
## NoOfUnitsPurchased 0.4059261 0.853619169  
## FrquncyOfPurchase 0.3657367 0.936990990  
## FrequencyOFPlay 0.7494212 0.252180001  
## NoOfGamesPlayed 1.0000000 0.374659305  
## NoOfGamesBought 0.3746593 1.000000000

Write a code to plot the corrplot of correlation between the attributes

# Library corrplot  
corrplot(r)

 Write a code to seperate the categorical attributes

data\_cat = customer.data.withoutCustomerId[, categorical.attributes]

Write a code to combine the numerical,categorical data along with the target

customer.data.withoutCustomerId = cbind(normalized.data\_num,data\_cat, TotalRevenueGenerated)

Write a code to set the seed and comment why it is used.

# Setting the seed to make sure that the same set of rows are taken each time when we randomly select the training and test data.  
set.seed(1234)

Write a code to get the train rows using sample

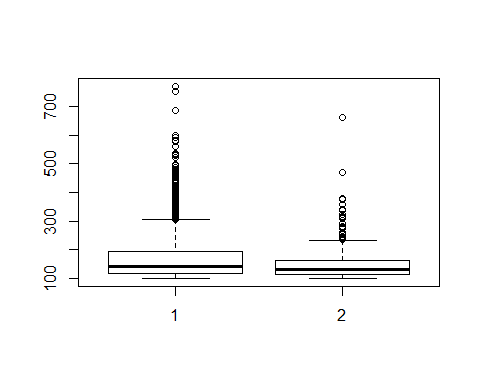
train.rows = sample(x = 1:nrow(customer.data.withoutCustomerId), size = 0.7\*nrow(customer.data.withoutCustomerId))

Write a code to get the train and test

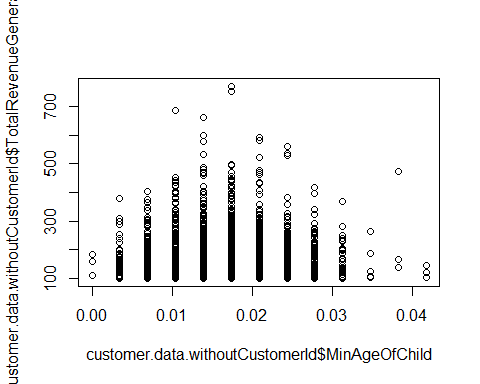
train.data = customer.data.withoutCustomerId[train.rows, ]  
test.data = customer.data.withoutCustomerId[-train.rows, ]

Write a code to just plot the grahs between attributes and targets

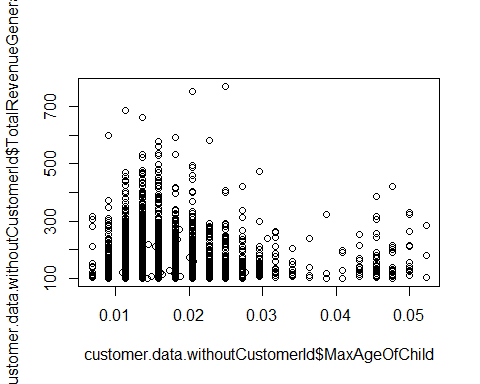
plot(customer.data.withoutCustomerId$City, customer.data.withoutCustomerId$TotalRevenueGenerated)



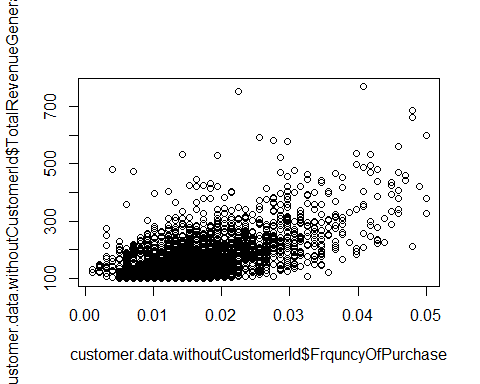
plot(customer.data.withoutCustomerId$MinAgeOfChild, customer.data.withoutCustomerId$TotalRevenueGenerated)



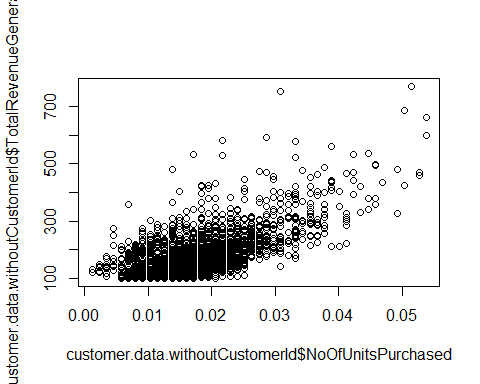
plot(customer.data.withoutCustomerId$MaxAgeOfChild, customer.data.withoutCustomerId$TotalRevenueGenerated)



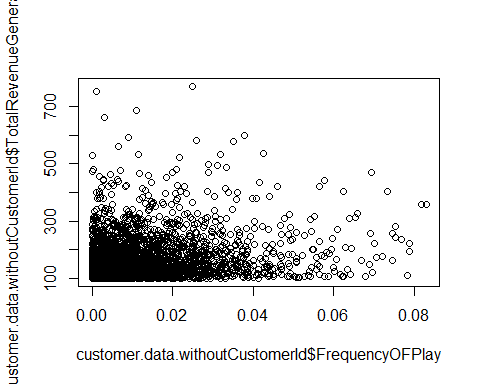
plot(customer.data.withoutCustomerId$FrquncyOfPurchase, customer.data.withoutCustomerId$TotalRevenueGenerated)



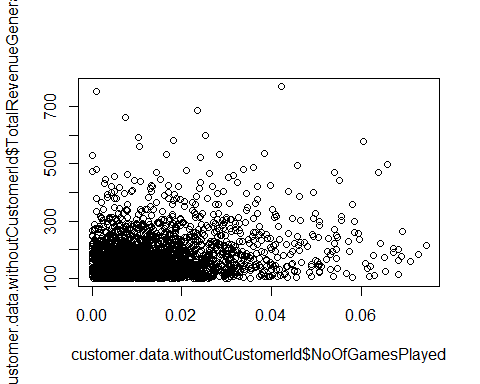
plot(customer.data.withoutCustomerId$NoOfUnitsPurchased, customer.data.withoutCustomerId$TotalRevenueGenerated)



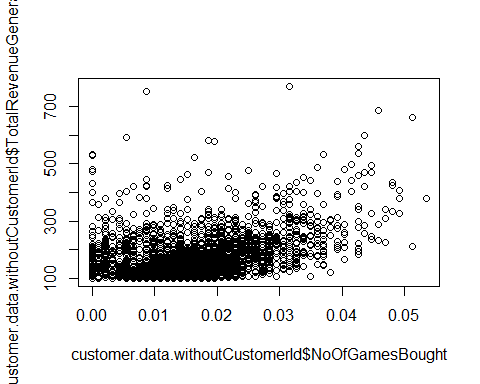
plot(customer.data.withoutCustomerId$FrequencyOFPlay, customer.data.withoutCustomerId$TotalRevenueGenerated)



plot(customer.data.withoutCustomerId$NoOfGamesPlayed, customer.data.withoutCustomerId$TotalRevenueGenerated)



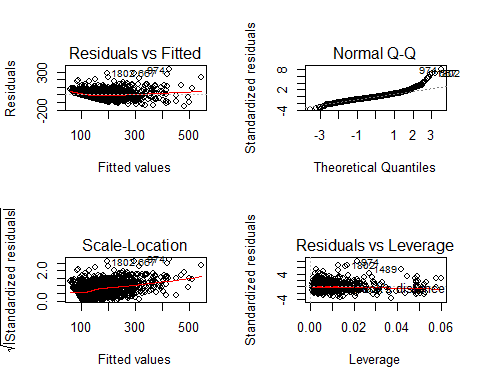
plot(customer.data.withoutCustomerId$NoOfGamesBought, customer.data.withoutCustomerId$TotalRevenueGenerated)

 Write a code to form a linear regression model

lm.model.ordinary = lm(TotalRevenueGenerated ~ City + NoOfChildren + MinAgeOfChild + MaxAgeOfChild + NoOfUnitsPurchased + FrquncyOfPurchase + FrequencyOFPlay + (NoOfGamesPlayed \* NoOfGamesBought) + FavoriteChannelOfTransaction + FavoriteGame , data = train.data)

Write a code to plot the model

par(mfrow = c(2,2))  
plot(lm.model.ordinary)

 Write a code to check the summary of the model

summary(lm.model.ordinary)

##   
## Call:  
## lm(formula = TotalRevenueGenerated ~ City + NoOfChildren + MinAgeOfChild +   
## MaxAgeOfChild + NoOfUnitsPurchased + FrquncyOfPurchase +   
## FrequencyOFPlay + (NoOfGamesPlayed \* NoOfGamesBought) + FavoriteChannelOfTransaction +   
## FavoriteGame, data = train.data)  
##   
## Residuals:  
## Min 1Q Median 3Q Max   
## -147.37 -26.31 -3.62 20.56 328.70   
##   
## Coefficients:  
## Estimate Std. Error t value Pr(>|t|)  
## (Intercept) 34.013 6.554 5.190 2.30e-07  
## City2 10.905 2.974 3.667 0.000251  
## NoOfChildren 565.054 162.598 3.475 0.000520  
## MinAgeOfChild 941.825 197.981 4.757 2.09e-06  
## MaxAgeOfChild -316.603 188.782 -1.677 0.093669  
## NoOfUnitsPurchased 7104.293 378.524 18.768 < 2e-16  
## FrquncyOfPurchase 10755.620 525.659 20.461 < 2e-16  
## FrequencyOFPlay 379.183 109.788 3.454 0.000563  
## NoOfGamesPlayed -1191.943 209.805 -5.681 1.52e-08  
## NoOfGamesBought -11098.599 329.423 -33.691 < 2e-16  
## FavoriteChannelOfTransactionUniform -12.976 2.524 -5.140 2.99e-07  
## FavoriteGameNONE -1.579 10.378 -0.152 0.879048  
## FavoriteGameUniform -2.871 4.600 -0.624 0.532576  
## NoOfGamesPlayed:NoOfGamesBought 41318.930 8357.898 4.944 8.25e-07  
##   
## (Intercept) \*\*\*  
## City2 \*\*\*  
## NoOfChildren \*\*\*  
## MinAgeOfChild \*\*\*  
## MaxAgeOfChild .   
## NoOfUnitsPurchased \*\*\*  
## FrquncyOfPurchase \*\*\*  
## FrequencyOFPlay \*\*\*  
## NoOfGamesPlayed \*\*\*  
## NoOfGamesBought \*\*\*  
## FavoriteChannelOfTransactionUniform \*\*\*  
## FavoriteGameNONE   
## FavoriteGameUniform   
## NoOfGamesPlayed:NoOfGamesBought \*\*\*  
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## Residual standard error: 42.63 on 2191 degrees of freedom  
## Multiple R-squared: 0.6604, Adjusted R-squared: 0.6584   
## F-statistic: 327.8 on 13 and 2191 DF, p-value: < 2.2e-16

Write a code to predict the value of the target on the linear model

data.predict = predict(lm.model.ordinary, test.data[, !(names(test.data) %in% c("TotalRevenueGenerated"))])

Write a code to use stepAIC

lm.model.stepAIC = stepAIC(lm.model.ordinary, direction = "both")

## Start: AIC=16563.15  
## TotalRevenueGenerated ~ City + NoOfChildren + MinAgeOfChild +   
## MaxAgeOfChild + NoOfUnitsPurchased + FrquncyOfPurchase +   
## FrequencyOFPlay + (NoOfGamesPlayed \* NoOfGamesBought) + FavoriteChannelOfTransaction +   
## FavoriteGame  
##   
## Df Sum of Sq RSS AIC  
## - FavoriteGame 2 720 3983200 16560  
## <none> 3982480 16563  
## - MaxAgeOfChild 1 5112 3987593 16564  
## - FrequencyOFPlay 1 21682 4004163 16573  
## - NoOfChildren 1 21951 4004432 16573  
## - City 1 24439 4006920 16575  
## - MinAgeOfChild 1 41134 4023615 16584  
## - NoOfGamesPlayed:NoOfGamesBought 1 44424 4026904 16586  
## - FavoriteChannelOfTransaction 1 48027 4030508 16588  
## - NoOfUnitsPurchased 1 640273 4622754 16890  
## - FrquncyOfPurchase 1 760982 4743463 16947  
##   
## Step: AIC=16559.54  
## TotalRevenueGenerated ~ City + NoOfChildren + MinAgeOfChild +   
## MaxAgeOfChild + NoOfUnitsPurchased + FrquncyOfPurchase +   
## FrequencyOFPlay + NoOfGamesPlayed + NoOfGamesBought + FavoriteChannelOfTransaction +   
## NoOfGamesPlayed:NoOfGamesBought  
##   
## Df Sum of Sq RSS AIC  
## <none> 3983200 16560  
## - MaxAgeOfChild 1 5190 3988390 16560  
## + FavoriteGame 2 720 3982480 16563  
## - FrequencyOFPlay 1 21534 4004734 16569  
## - NoOfChildren 1 22052 4005252 16570  
## - City 1 24444 4007644 16571  
## - MinAgeOfChild 1 41586 4024786 16580  
## - FavoriteChannelOfTransaction 1 48180 4031380 16584  
## - NoOfGamesPlayed:NoOfGamesBought 1 48429 4031629 16584  
## - NoOfUnitsPurchased 1 640278 4623478 16886  
## - FrquncyOfPurchase 1 769554 4752754 16947

Write a code to predict it on stepAIC

data.predict.stepAIC = predict(lm.model.stepAIC, test.data[, !(names(test.data) %in% c("TotalRevenueGenerated"))])

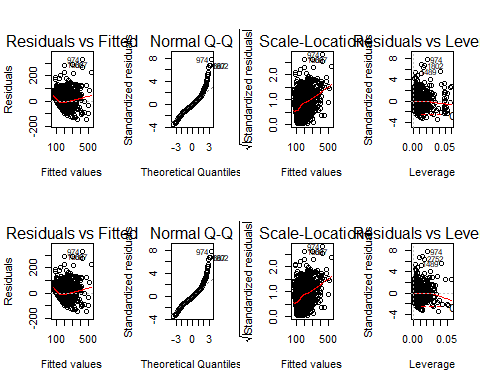
Write a code to check the multicollinearity in the lm model

vif(lm.model.ordinary)

## GVIF Df GVIF^(1/(2\*Df))  
## City 1.082325 1 1.040348  
## NoOfChildren 1.944582 1 1.394483  
## MinAgeOfChild 1.566044 1 1.251417  
## MaxAgeOfChild 1.755946 1 1.325121  
## NoOfUnitsPurchased 8.800694 1 2.966596  
## FrquncyOfPurchase 19.299691 1 4.393141  
## FrequencyOFPlay 2.277993 1 1.509302  
## NoOfGamesPlayed 7.043828 1 2.654021  
## NoOfGamesBought 9.772635 1 3.126121  
## FavoriteChannelOfTransaction 1.111848 1 1.054442  
## FavoriteGame 1.155144 2 1.036714  
## NoOfGamesPlayed:NoOfGamesBought 8.321342 1 2.884674

Write a code to check the plots of the models

par(mfrow = c(2,4))  
plot(lm.model.ordinary)  
plot(lm.model.stepAIC)



Write a code to check the multicollinearity problem

vif(lm.model.stepAIC)

## City NoOfChildren   
## 1.082097 1.943988   
## MinAgeOfChild MaxAgeOfChild   
## 1.562290 1.753976   
## NoOfUnitsPurchased FrquncyOfPurchase   
## 8.786162 19.139888   
## FrequencyOFPlay NoOfGamesPlayed   
## 2.277079 6.717575   
## NoOfGamesBought FavoriteChannelOfTransaction   
## 9.326273 1.111221   
## NoOfGamesPlayed:NoOfGamesBought   
## 7.952508

Write a code to evaluate the error in the prediction of AIC

regr.eval(test.data$TotalRevenueGenerated, data.predict.stepAIC)

## mae mse rmse mape   
## 29.0679067 1593.4304771 39.9177965 0.1817721

Write a code to evaluate the error on the prediction of Linear Regression

regr.eval(test.data$TotalRevenueGenerated, data.predict)

## mae mse rmse mape   
## 29.0527323 1590.8979757 39.8860624 0.1816509