s

**S P MANDALI’S**

**RAMNARIAN RUIA AUTONOMOUS COLLEGE**

**L.N. ROAD, MATUNGA(EAST), MUMBAI - 400019**

**2019-20**

**A PROJECT REPORT ON**

**MOBILE BRAND**

**AND**

**CUSTOMER PURCHASE**

**ANALYSIS**

**BY**

**RAHUL MUKESH SINGH - 180130**

Under the Guidance of

**PROF. ABHIJEET GOLE**

**DECLARATION**

**Rahul Mukesh Singh (Roll No. : 180130)**, hereby declare that this project report entitled: **“MOBILE BRAND AND CUSTOMER PURCHASE ANALYSIS”** which is being submitted in fulfilment of the Masters of Science in Computer Science Examination conducted by Ramnarian Ruia College – [Autonomous, under Mumbai University] is the result of the work carried out by me under the supervision of **Prof. Abhijeet Gole** of Ramnarian Ruia College-Autonomous Mumbai.

The information submitted is true and original to the best of Knowledge.

**Signature**

**[Rahul Mukesh Singh]**

**Date:**

**ACKNOWLEDGEMENT**

We take this opportunity to express our profound gratitude and deep regards to our teachers for their exemplary guidance, monitoring and constant encouragement throughout the course of this project.

The blessings, help and guidance given by them, from time to time, shall carry me a long way in the journey of life on which we are about to embark.

We also take this opportunity to express a deep sense of gratitude to the Head Of Department, Mrs. Megha Sawant for her cordial support.

A large debt of gratitude is owned to my project guide Mr.Abhijeet Gole who has not only endured, but also assisted and inspired us for taking up the project on “MOBILE BRAND AND CUSTOMER PURCHASE ANALYSIS”.

We want to acknowledge and thank him for giving us the opportunity to do this under his guidance and for sharing his knowledge. His continuous guidance, time, valuable suggestions, inputs and helpful criticisms have helped me to accomplish such a task.

Lastly, We thank our parents, family and all those persons ‘Behind the Veil’ for their constant encouragement and support, which enabled me to complete the project through thick and thin.

**TABLE OF CONTENTS**

|  |
| --- |
| **Contents Page No.** |

|  |
| --- |
| 1. Title and Abstract………………………………………………….5 |
| 1. Background and literature review……………………………….7 |
| 1. Project Introduction and Scope…………………………………8 |
| 1. Results and Interpretations…………………………………… 12 |
| 1. Conclusions……………………………………………………...86 |
| 1. Future Scope and development…………………………….....87 |
| 1. References and Appendix………………………………………88 |
| 1. Index and acronyms……………………………………………..94 |

**TITLE AND ABSTRACT**

**MOBILE BRAND AND CUSTOMER PURCHASE ANALYSIS USING ADVANCED DATA ANALYTICS**

*The objective of this study is to study the dimensions of Mobile Phone Business with main focus on Mobile Brand and its customer purchasing habits. The targeted business is a firm which Wholesales or sells Mobile Phones in Huge Quantity. Currently, analysis is done based on dataset from Amazon. The objective is to analyze the current data available for the business and generate insights for future through data analytics.*

*The review and sales data available will help in understanding the top brands and features, which will in turn give the information to understand brand features and age group to target for improving business sales.* *Increase in demand of mobile phones, was most deciding factor to choose this domain for analysis. With huge demand and multiple Brands entering into this business, It becomes important to have edge from other competitors. So, Data Analytics helps to provide business intelligence to maximize sales.*

*The entire analysis was conducted with the help of Amazon.com mobile phone dataset from kaggle.com and R tool. Datasets was taken in CSV format. The current analysis is done with the sales and review data comprising of top 10 brands consisting of brands like Nokia, Samsung, Blackberry,etc.*

*In this it will also forecast the total sales for each brand to identify the top selling Brand in the future year. In this we are calculating the relationships between the Price and Mobile Camera by analyzing it via Age Group from Sales Data. This helps to understand the characteristic of a particular age group customer buying a certain product. This also helps to identify the customer class. I have also calculated the Top and Worst Features in each Brand using Reviews and Sentimental Analysis. Association is also considered in order to analyze => do customers move to purchase better camera phones?*

*Important factors i.e. price, age group, product features, brand name, age group were selected and analyzed through the use of clustering, classification and association analysis. From the analysis, it was clear that consumer’s value price followed by mobile phone features such as camera with respect to their age group as the most important variable amongst all and it also acted as a motivational force that influences them to go for a mobile phone purchase decision. It automatically identifies the important features of products from dataset’s consumer reviews.*

**BACKGROUND AND LITERATURE REVIEW**

Firms can know which group of customer’s to target and what is customer purchasing behavior using Data mining which in turn maximizes sales. Data Mining also helps firms to withstand the competition and also helps to improve customer and product service. To build Strong Brand Name, approach should be totally focused on customer. Database marketing can simplify the marketing strategies. Most firms built massive customers databases in order to track their purchase transactions. Data mining for knowledge-based marketing customers profiling, deviation analysis, and trend analysis are three major areas of application.

The customer’s point of view, higher price often result in higher expectations of the product or service performance. To attract customers, we need understand the strong features that are currently attracting the customers and features that are leading to customer loss. It’s also important to analyze current market scenario to be ahead of competitors. At the same time, in order to invest in business firms, there is also need to analyze the sales prediction for next few years. Different Age group customers follow different trends and liking. Today, Camera phone’s has become very important features in mobile phones in respect to sales.

So, in order to solve above mentioned problems, Data Analysis provides the solution through Aspect identification, classification, and analysis.

Also, correct predictive analytics is much more than just displaying the top products in a department for cross-selling. Oftentimes, the customer will already be purchasing those. Also the best models should also act as personal assistants for shopping in recommending products that have a positive but non-obvious association. Association also finds out are customers moving toward better features or are they happy with current features.

**PROJECT INTRODUCTION AND SCOPE**

This System contains the following: -

* Polarity Count in order to know positive and negative review.
* Emotion count in order to know emotion such as loyalty and anger behind each review.
* Confusion Matrix in order to see performance of the model
* Finding out top positive and negative features using reviews of customer.
* Trending topics among the Reviewers. (Using LDA)
* Clustering among customers based on [price and mobile camera megapixels] and analyzing it via age group of customers. (Using KMeans)
* Classification based on price and mobile camera pixels, in order to know which customer group to target for particular product. (Using Naïve Bayes)
* Association in order find out that whether customers are moving towards better features or they are satisfied with current feature. (Using Apriori)
* Forecasting Sales for particular Brand in order to find out on which to invest. (Using ARIMA Model)

Above features will help Sellers to analyze which Brand has best features and loyalty of customers towards that Brand, which will help to improve the sales.

Also forecasting of Sales for next few Years, helps sellers to take calculated risk in investing money on that Brand.

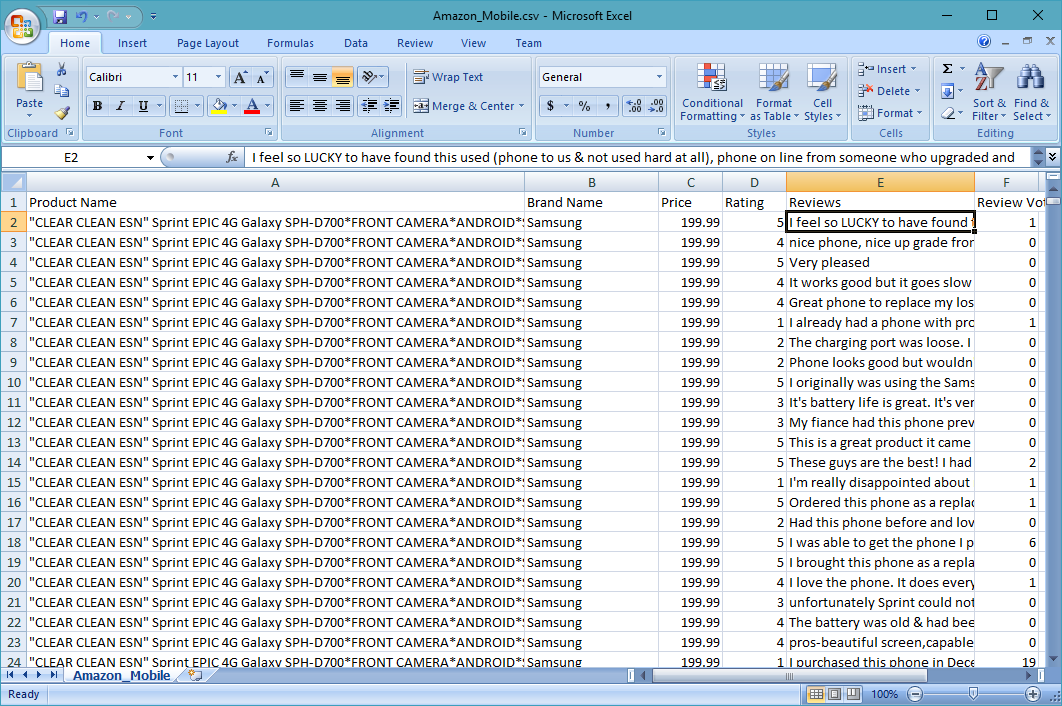
In this system, Language used is R Programming Language and the IDE used is RStudio.

Dataset

Dataset was taken from kaggle.com, which contained 400 thousand reviews of unlocked mobile phones sold on Amazon.com.

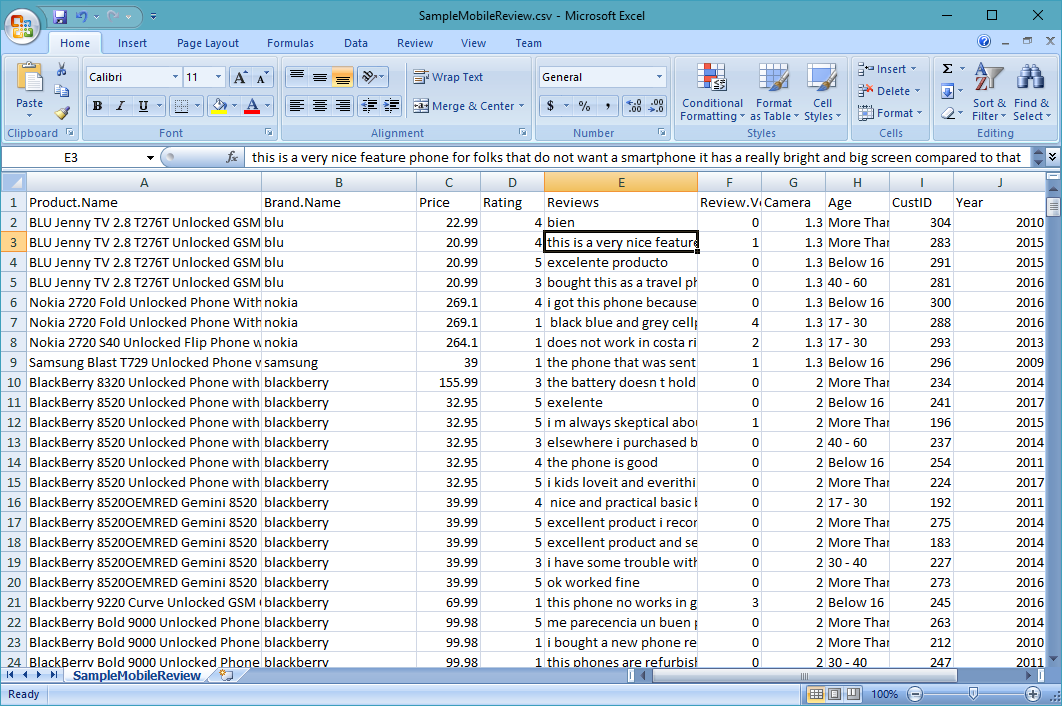
Fields Contains

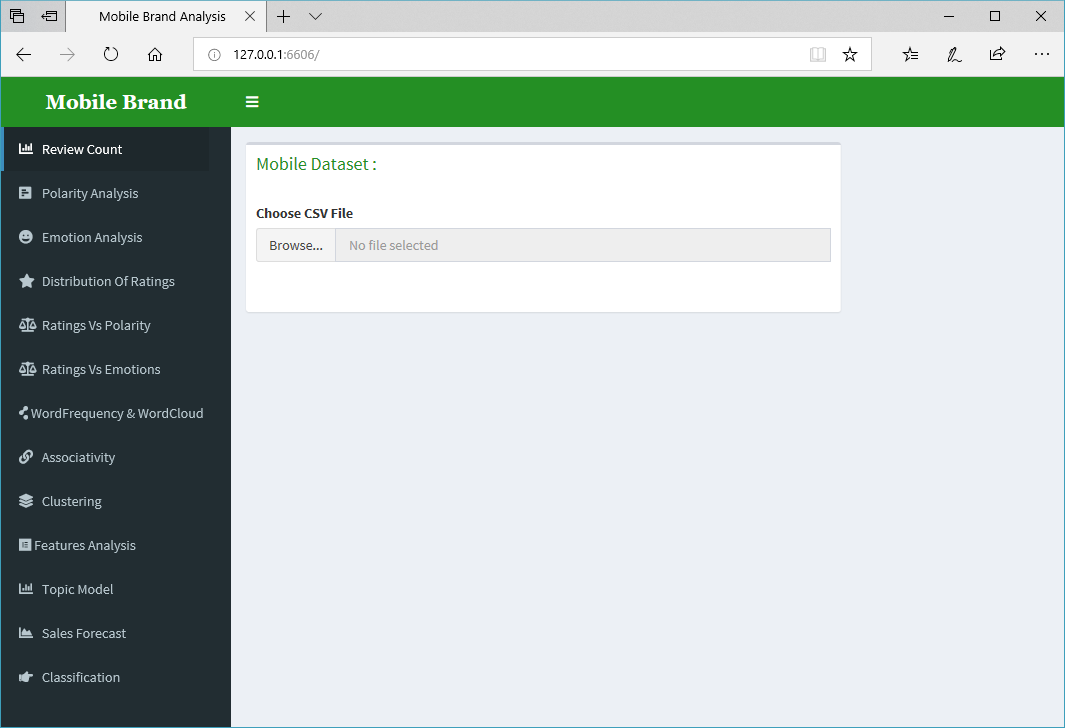
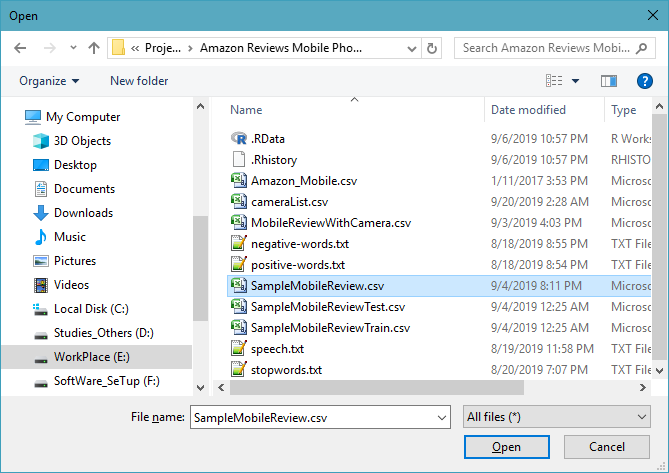
* Product Title (also contains features such as Camera, Operating Systems, etc)
* Brand
* Price
* Rating
* Review text
* Number of people who found the review helpful
* Customer ID
* Year of Purchase
* Age Group of Customer



Dataset was then Cleaned and Sampled based on No of Reviews Votes and Brands.

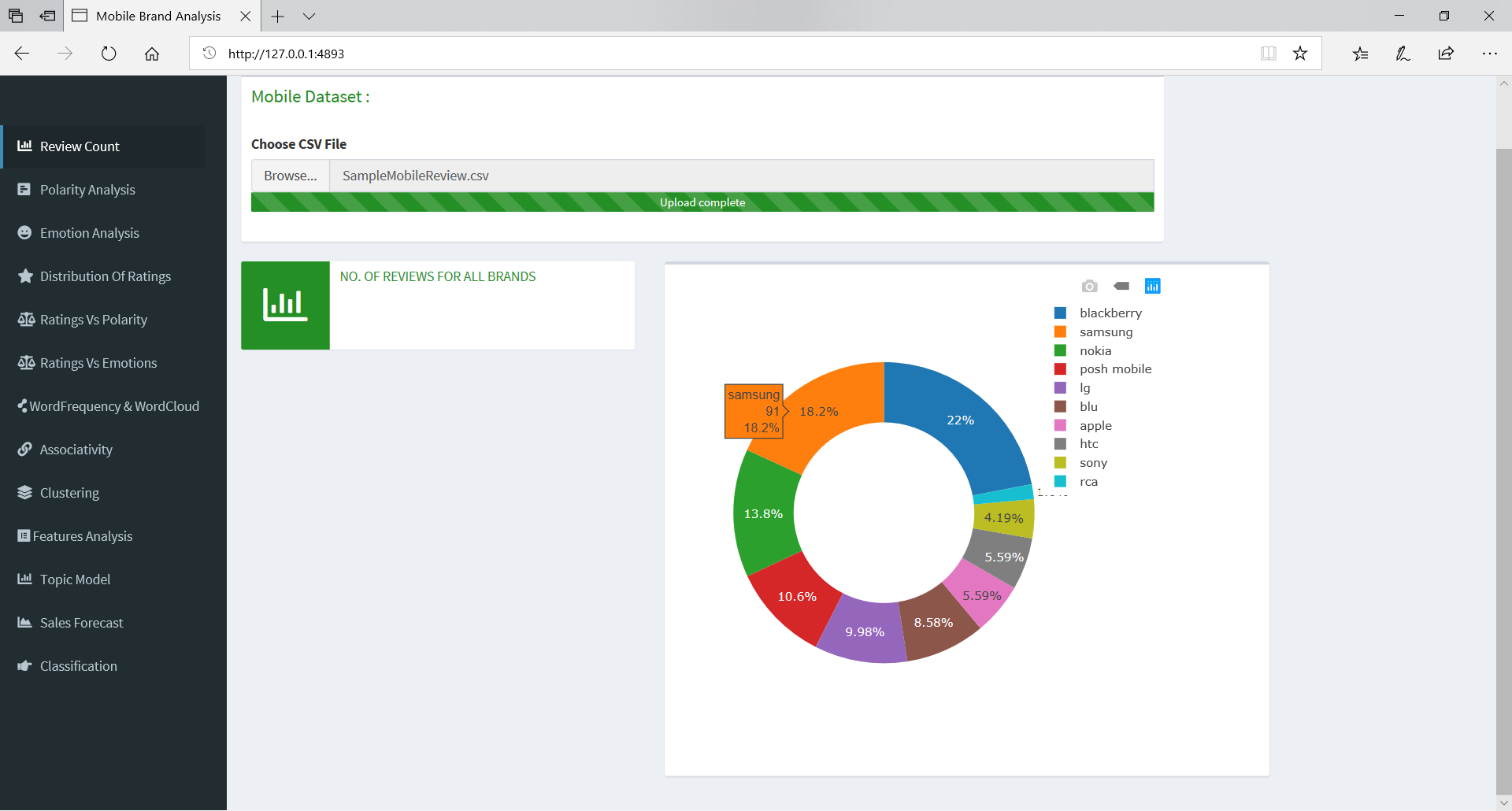
Also Camera Column was Extracted from Product Title to make it a new column (for Analysis) using regular expressions from above Original Dataset.





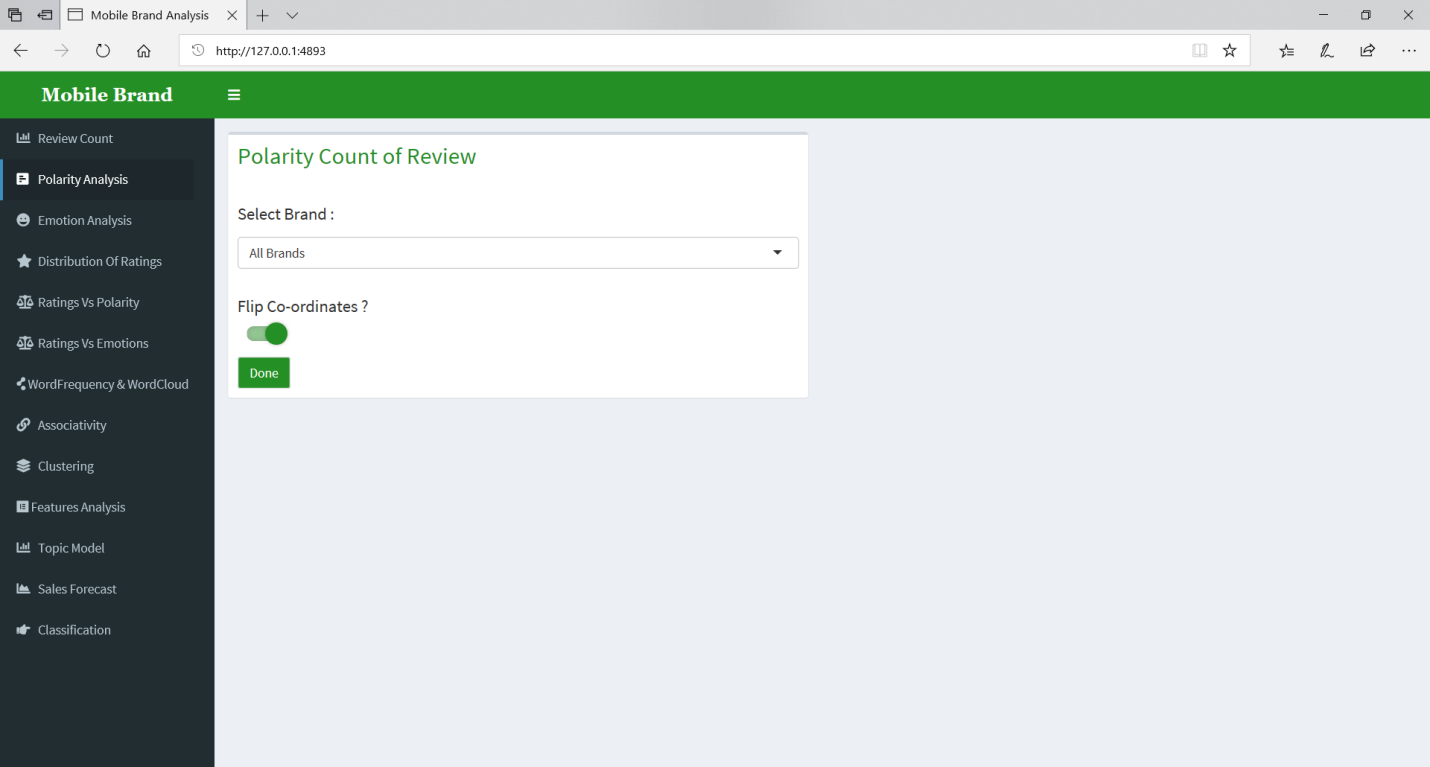
**Results and Interpretations**

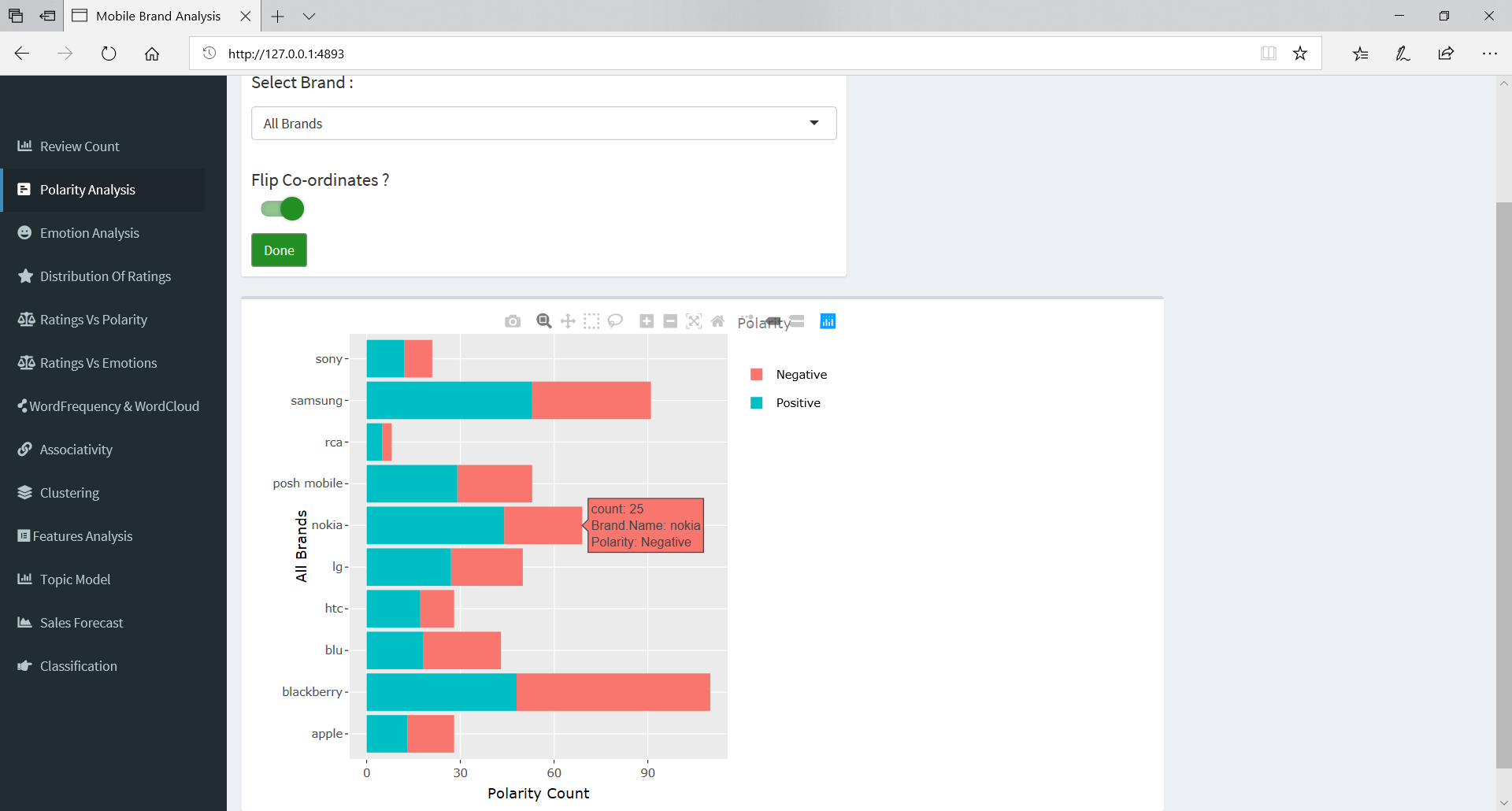
Q] Which Brand received more Reviews?



Blackberry, Samsung and Nokia Mobiles received more no. of reviews.

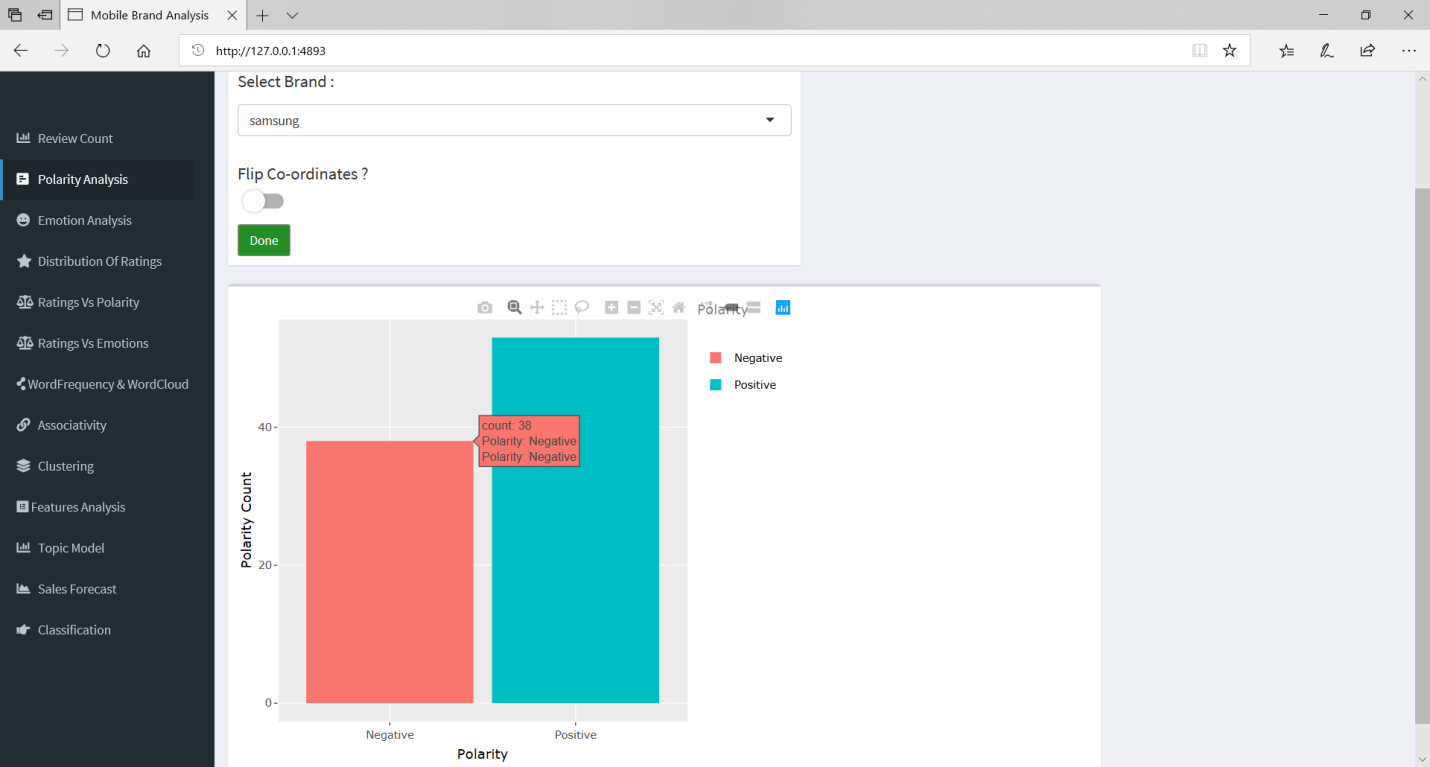
Q] Which Brand received more Positive or Negative Reviews?





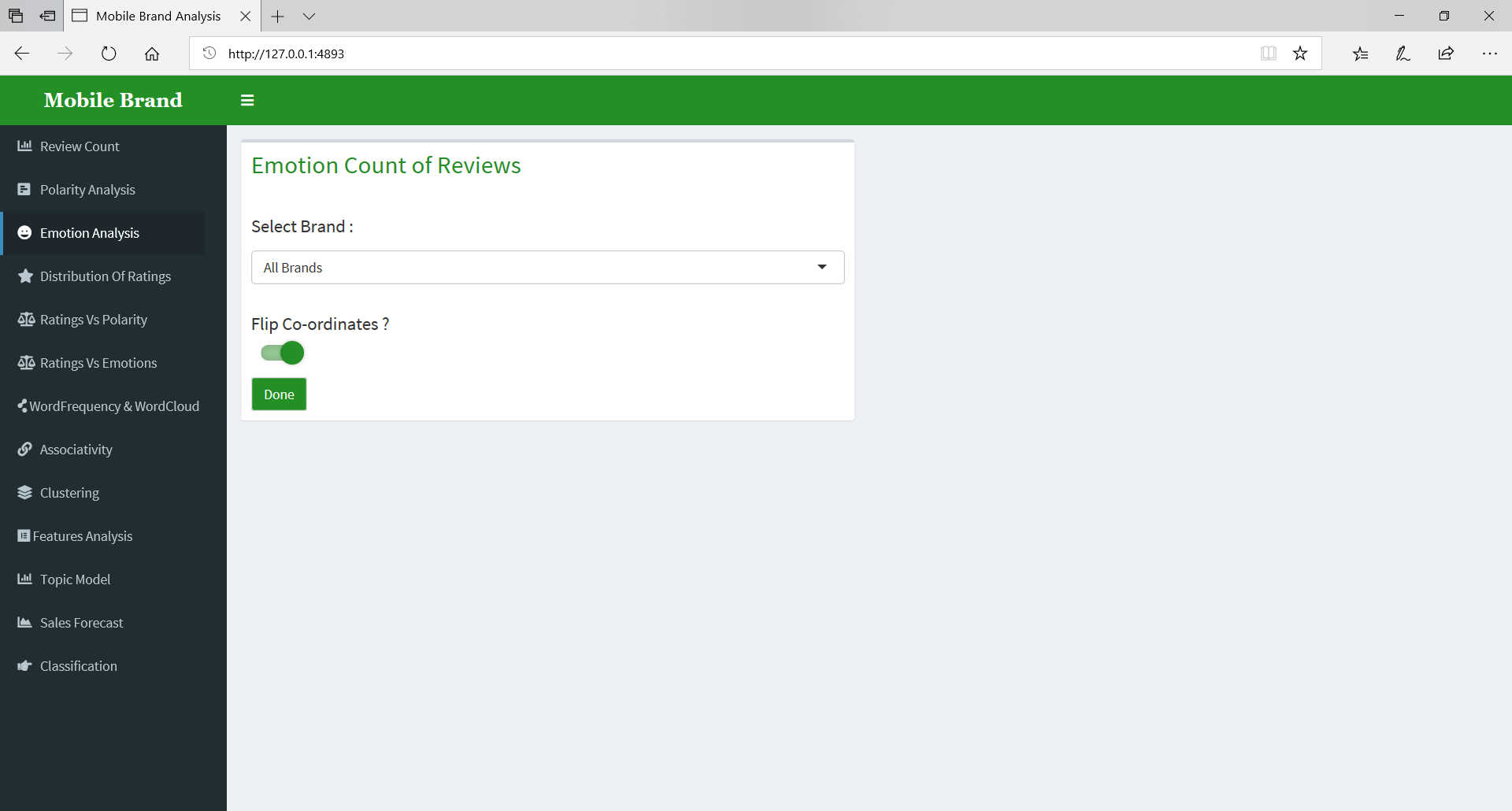
Samsung received most Positive Reviews and Blackberry received most negative Reviews.

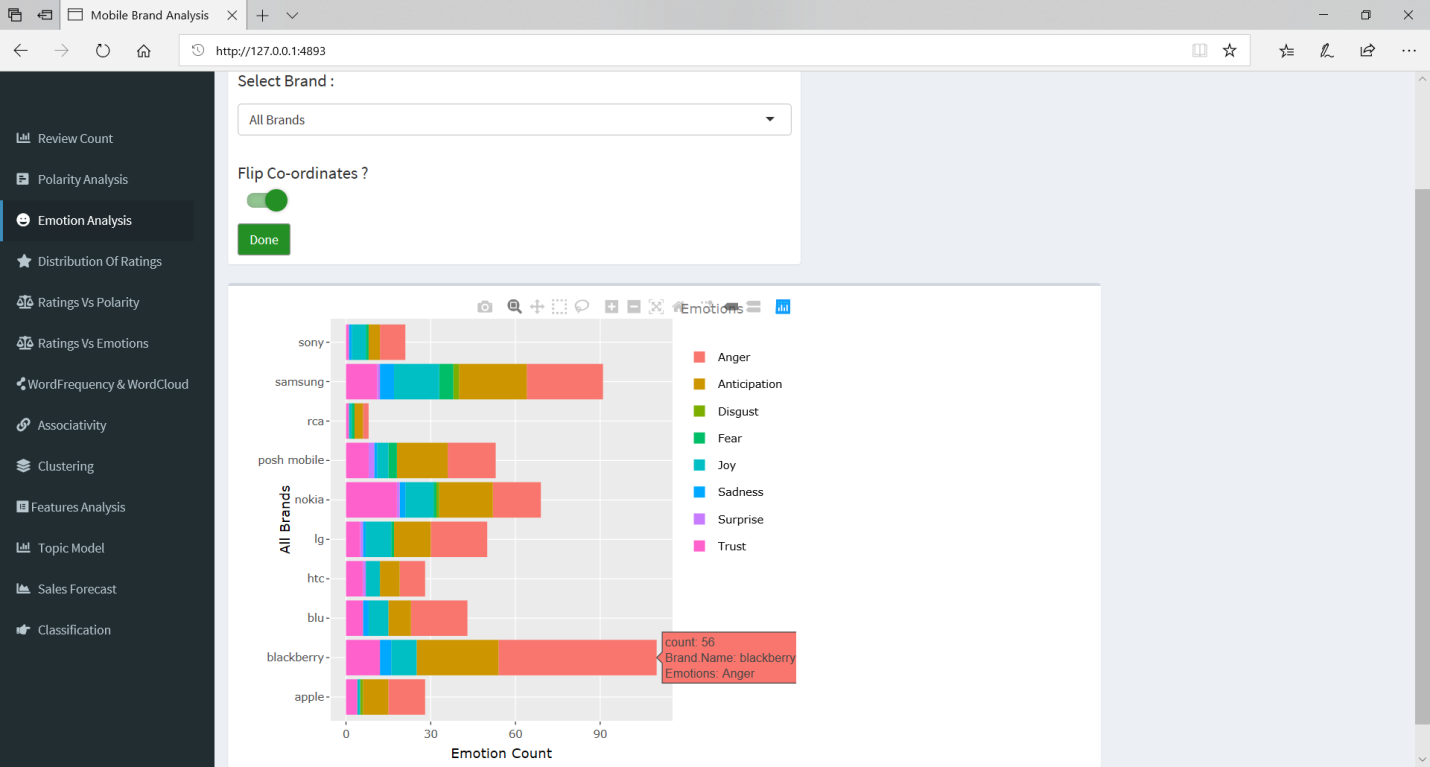
Q] Particular Brand (ex, Samsung) has more Positive or Negative Reviews?



Samsung has more positive reviews then Negative Reviews

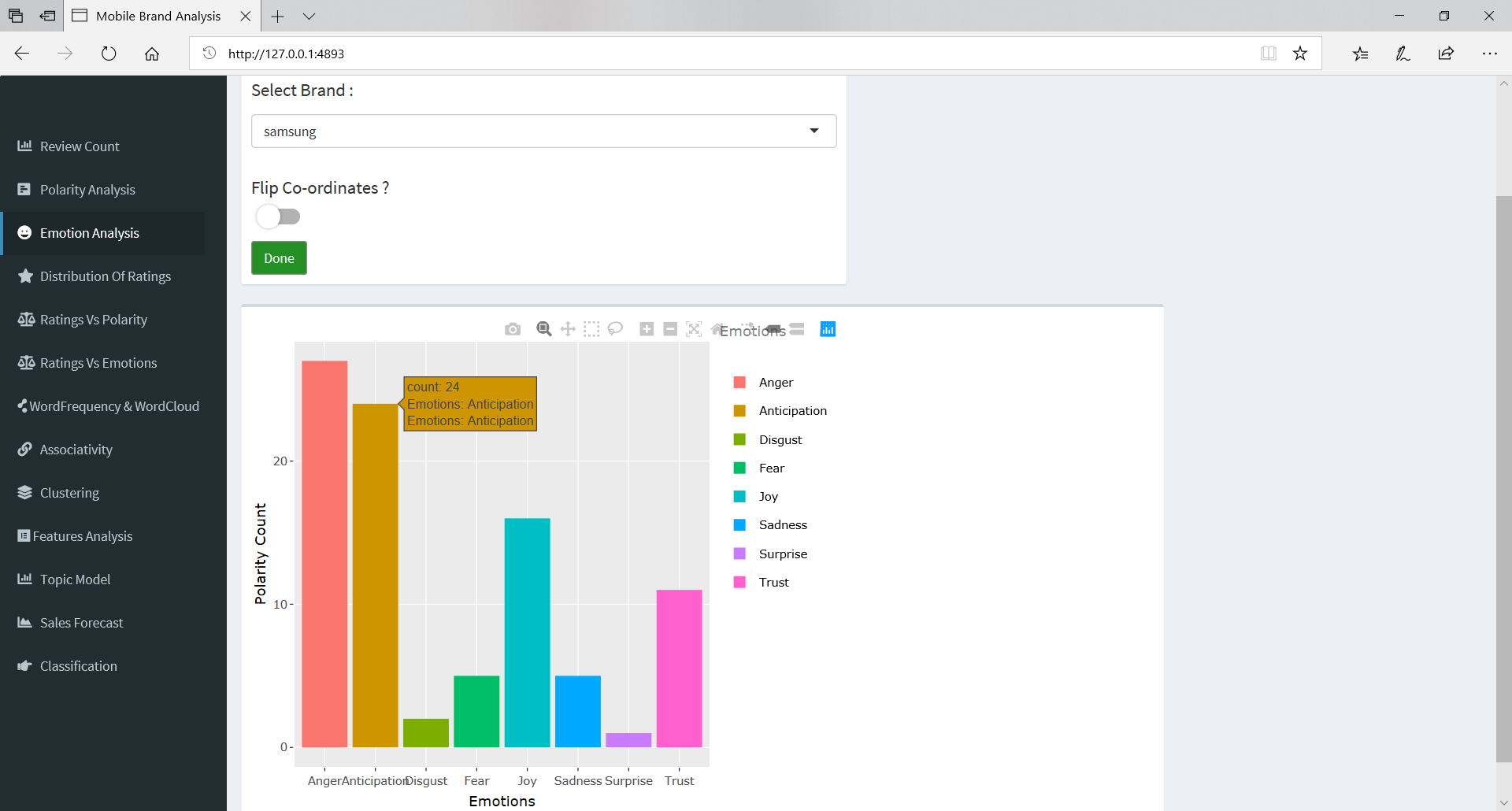
Q] Which Brand has more trust and anger in its Review?





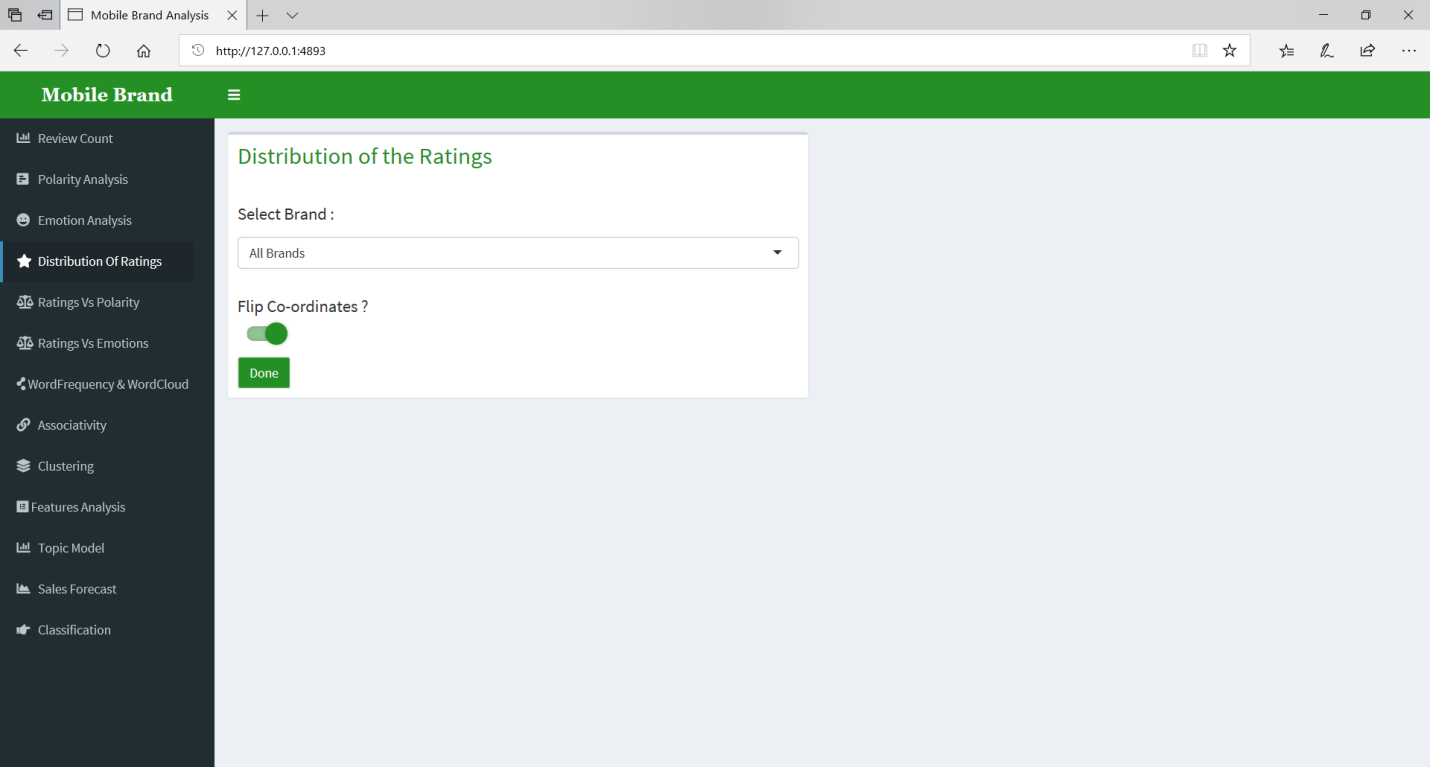
Nokia is most Trusted Brand while Blackberry has most number of anger reviews.

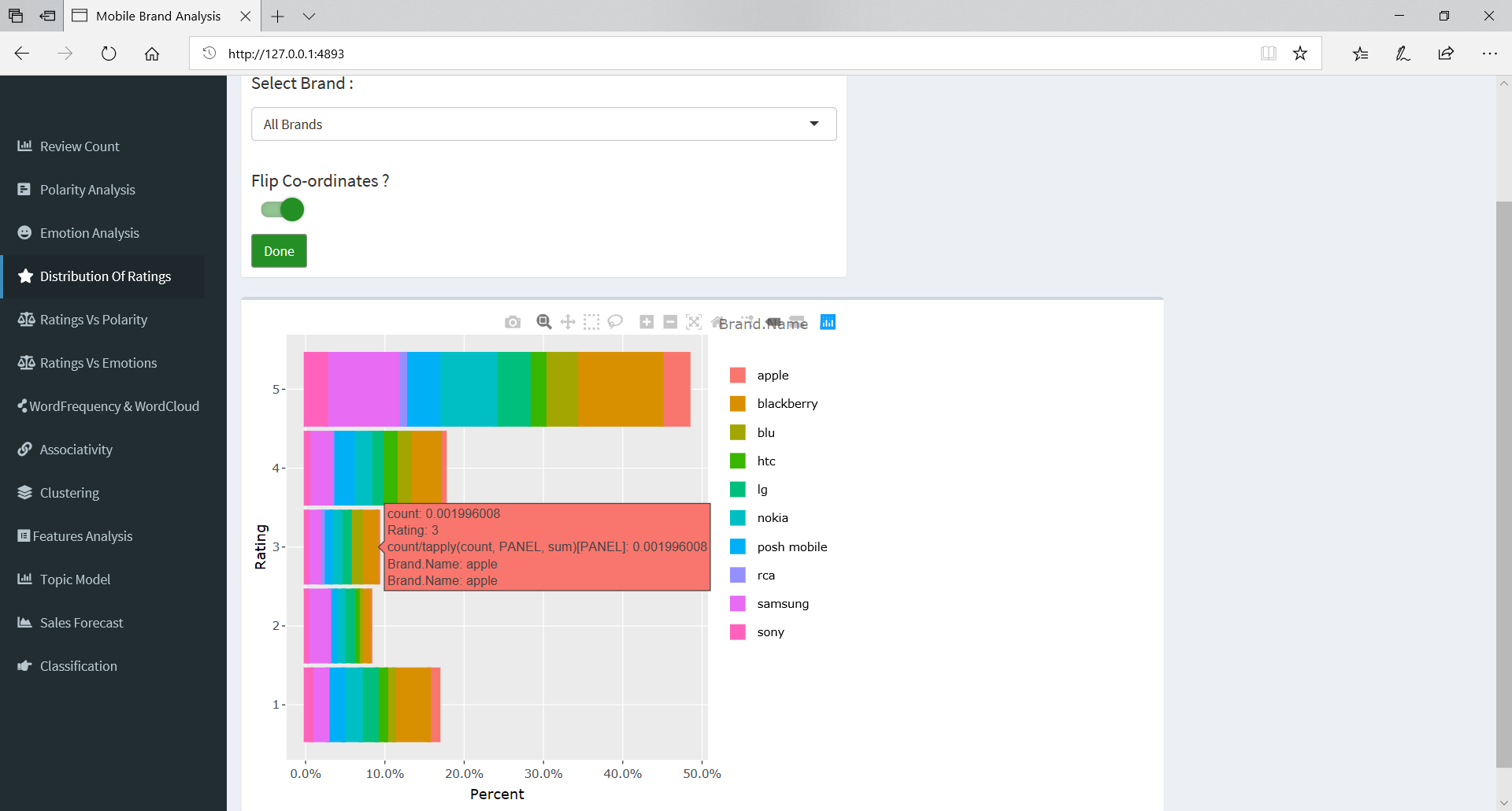
Q] Is Particular Brand (ex, Samsung) a trusted Brand among Reviewers?



Even though, Most Samsung Reviewer’s has anger in their review but it still it manages to receive good amount of joy and trust among the Positive Reviewers.

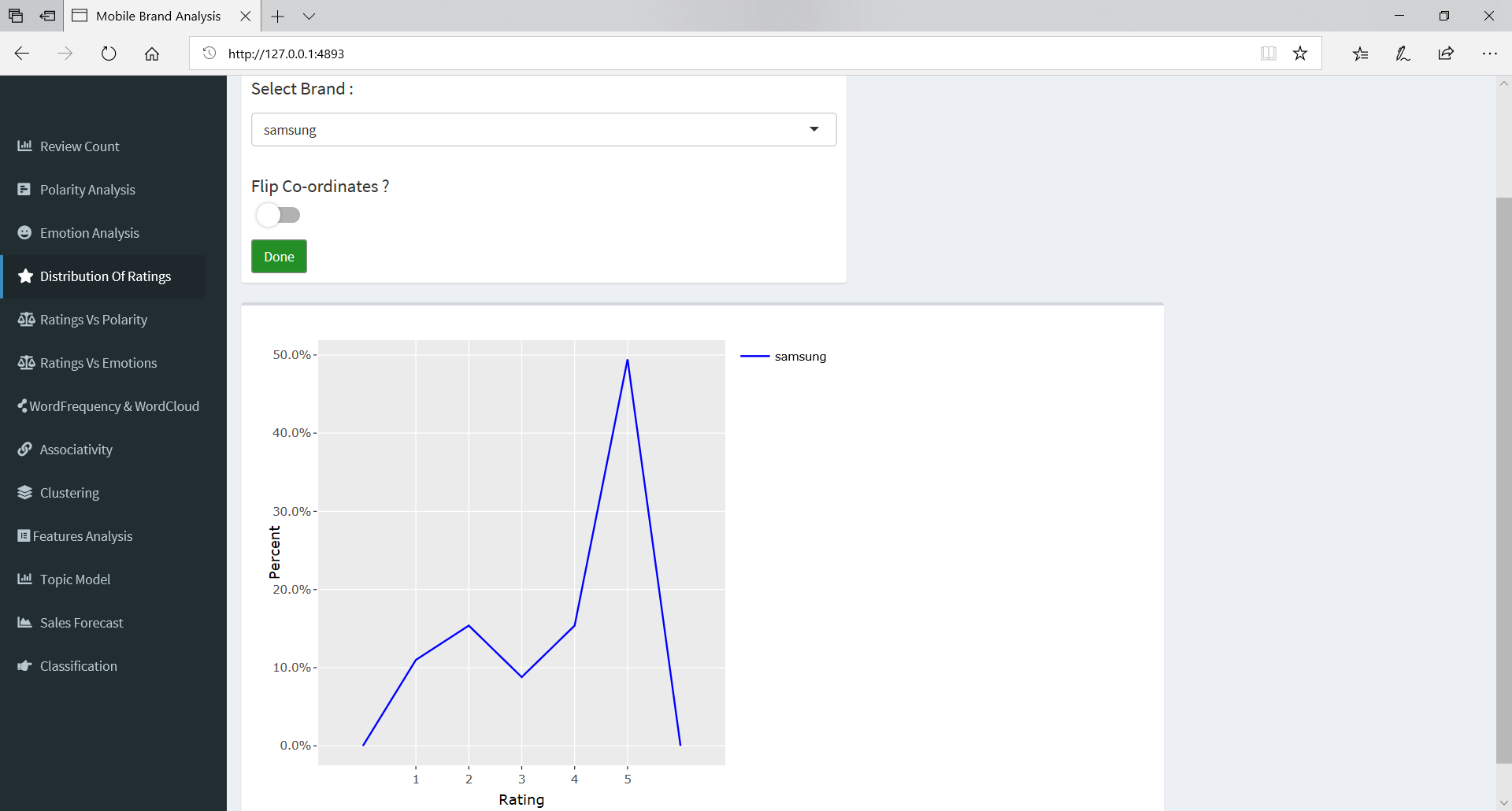
Q] Which Brand has received more no of 1 stars and 5 Stars?





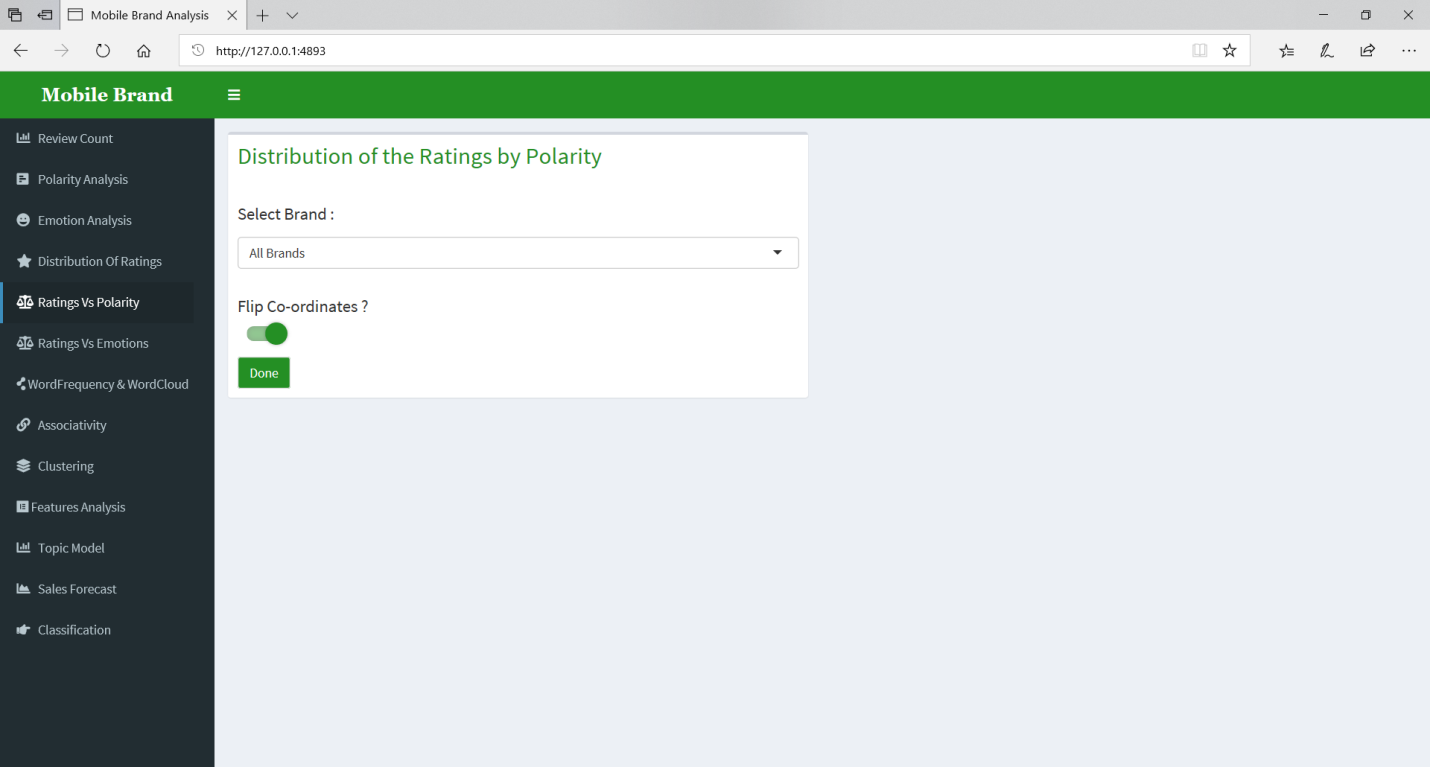
Blackberry and Nokia has received more no of 5 Stars but at the same time Blackberry has highest 1 star reviews.

Q] Is Particular Brand (ex, Samsung) has more 5 star reviews?

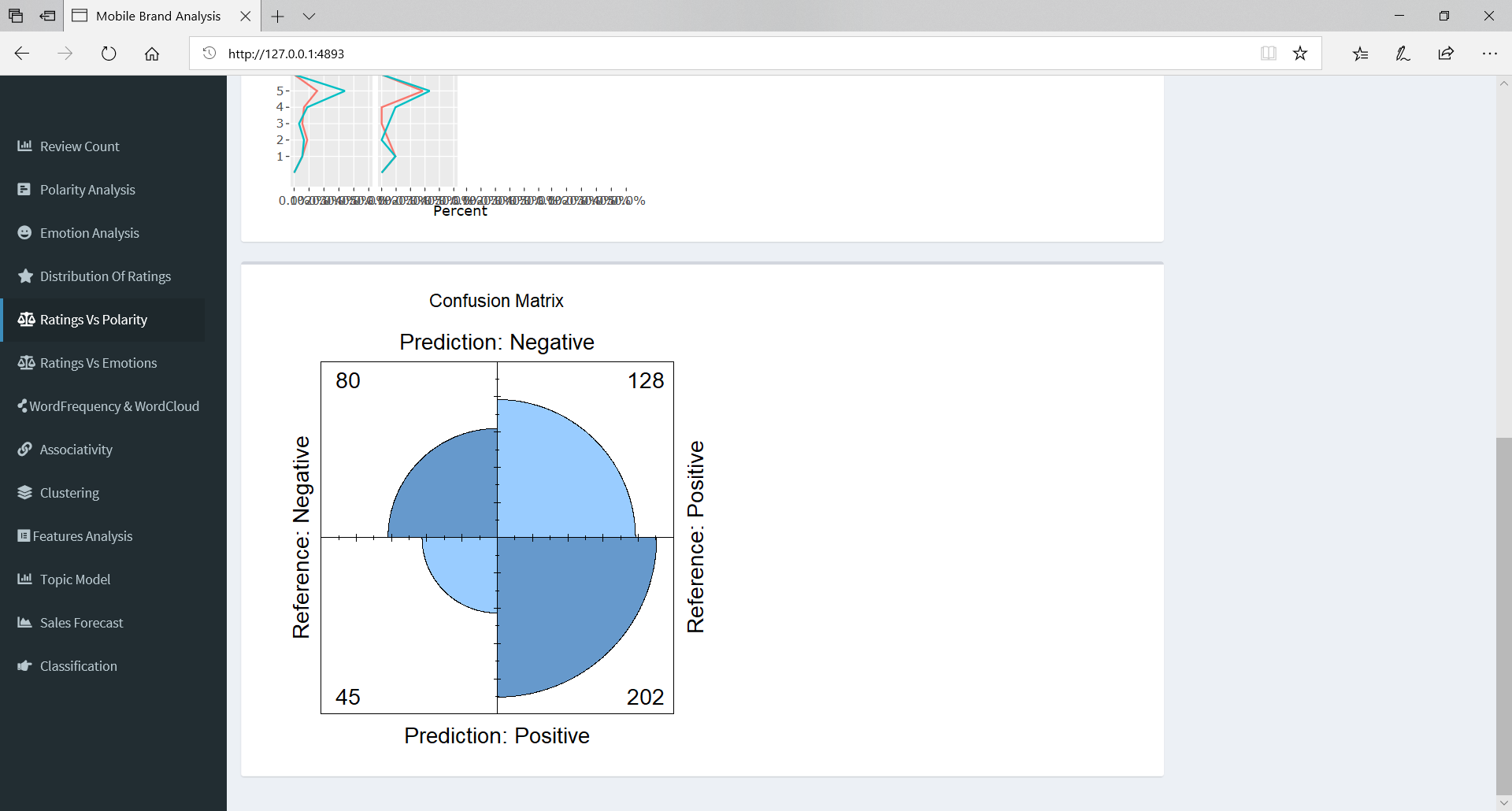


Yes, Samsung seems to receive approximate 5 star reviews >= other stars combined

Q] Was Polarity Count accurate?



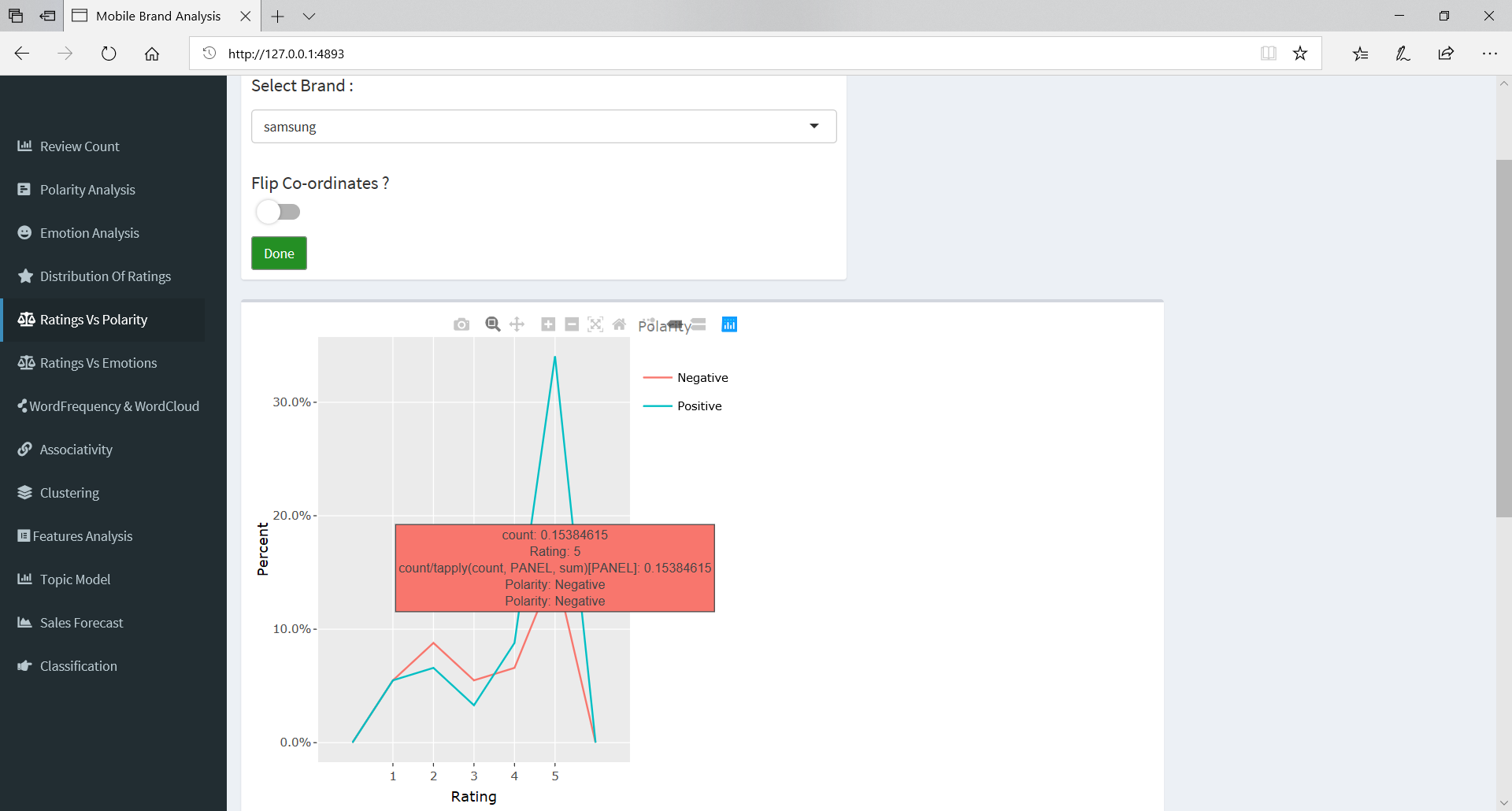


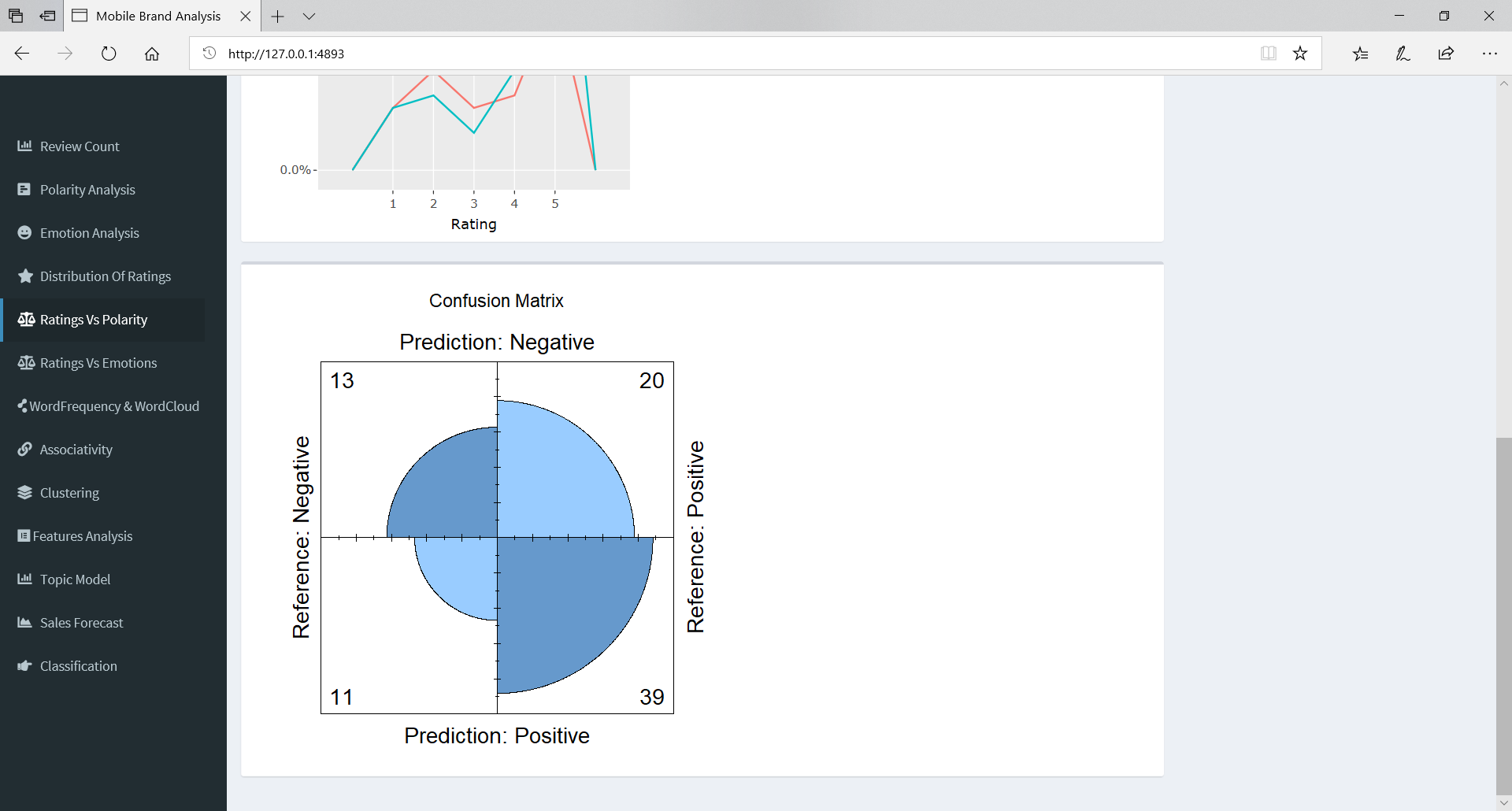


It turns out that Positive Polarity was more accurately calculated compared to Negative Polarity.

Total Accuracy was 61.98%.

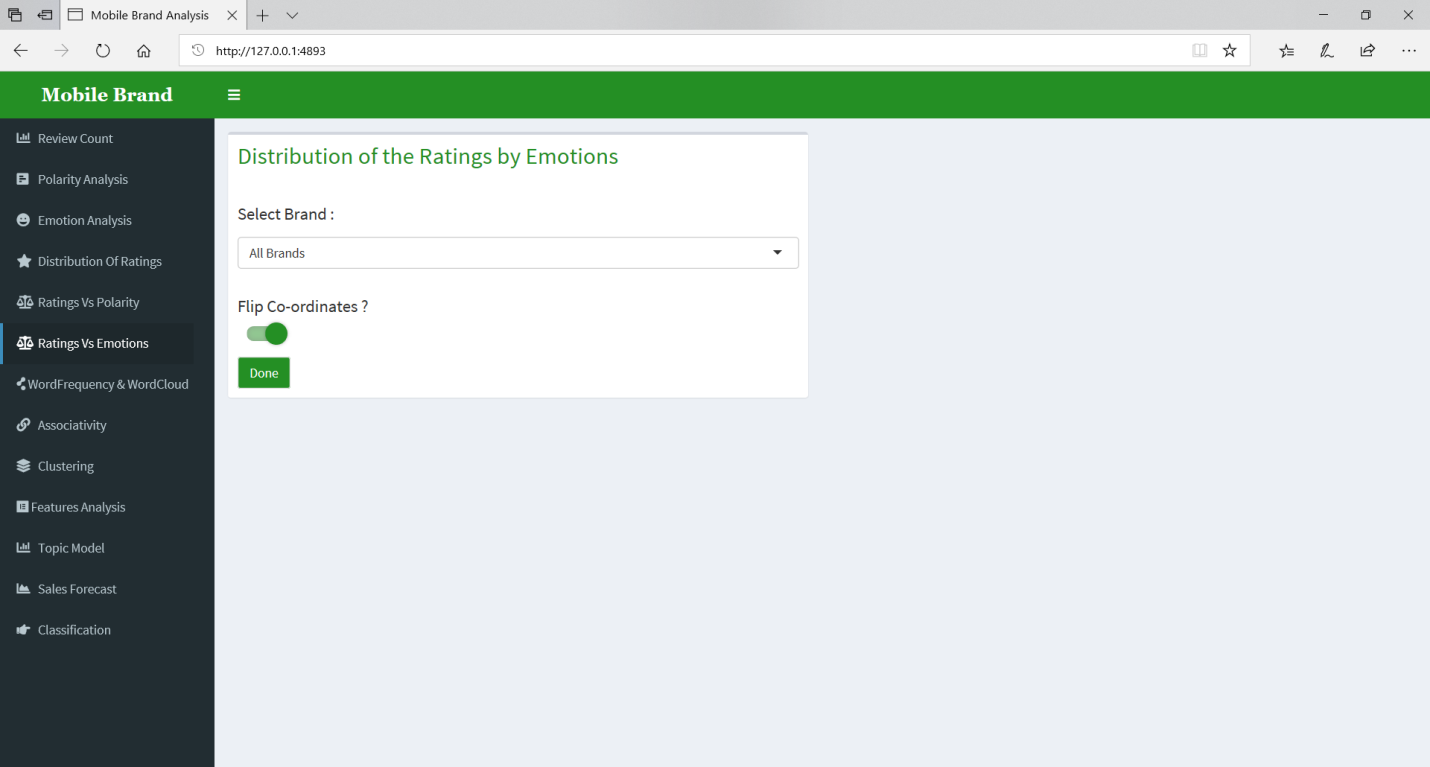
Q] Was Polarity Count accurate for particular brand (ex, Samsung)?

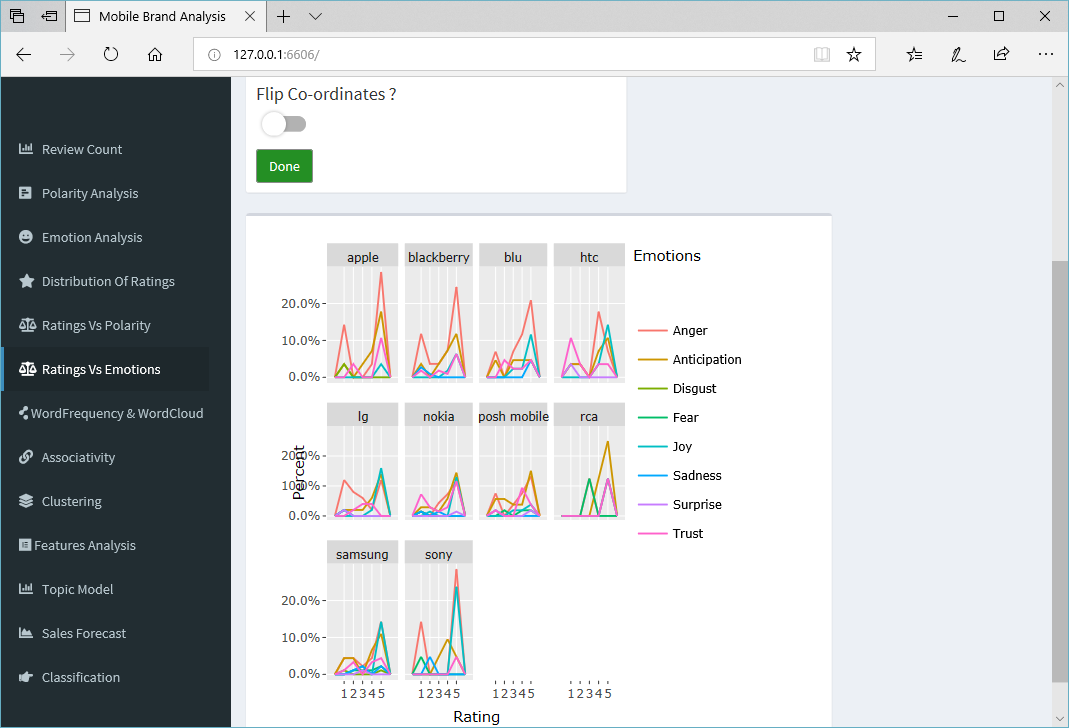




For Samsung, Polarity Count was 62.7% Accurate.

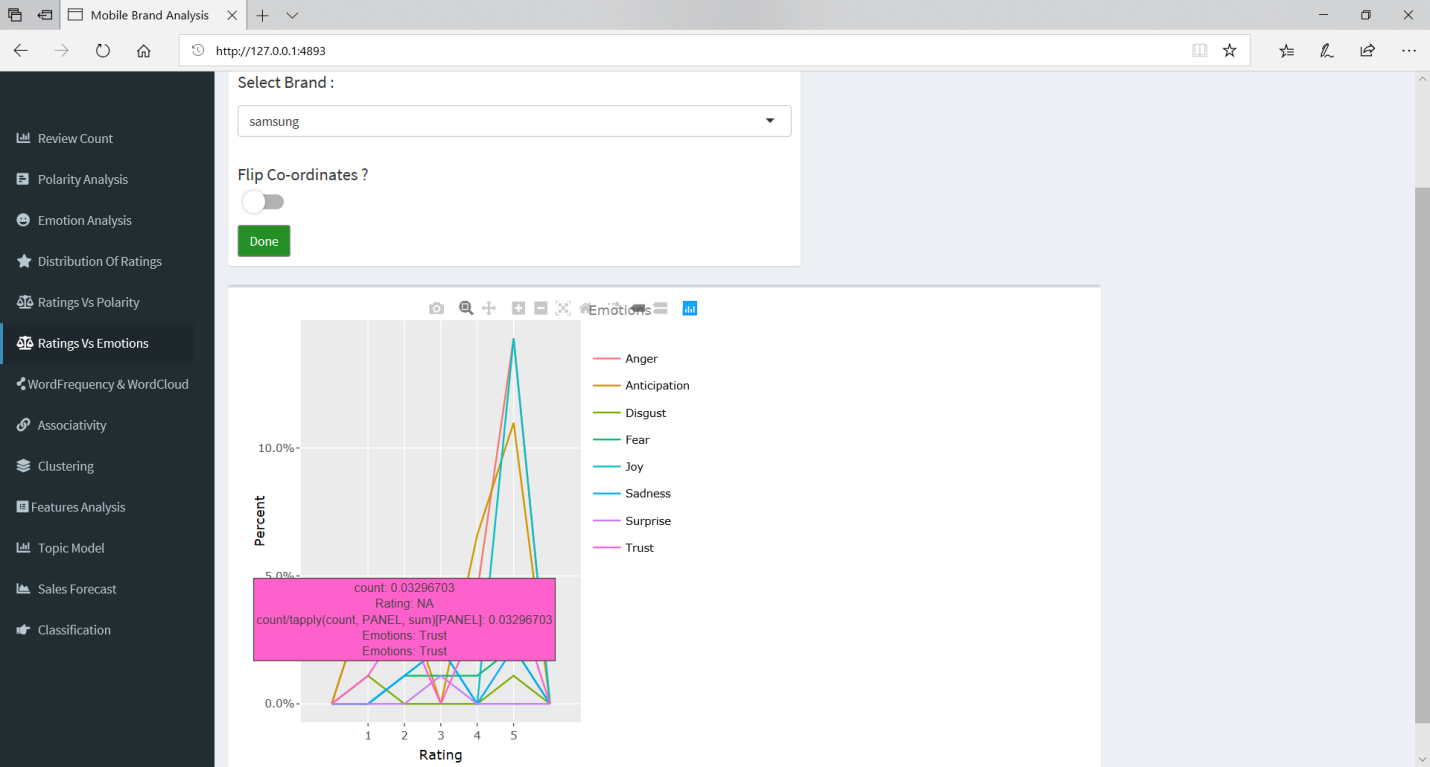
Q] 1 star rating contains which dominating Emotion?





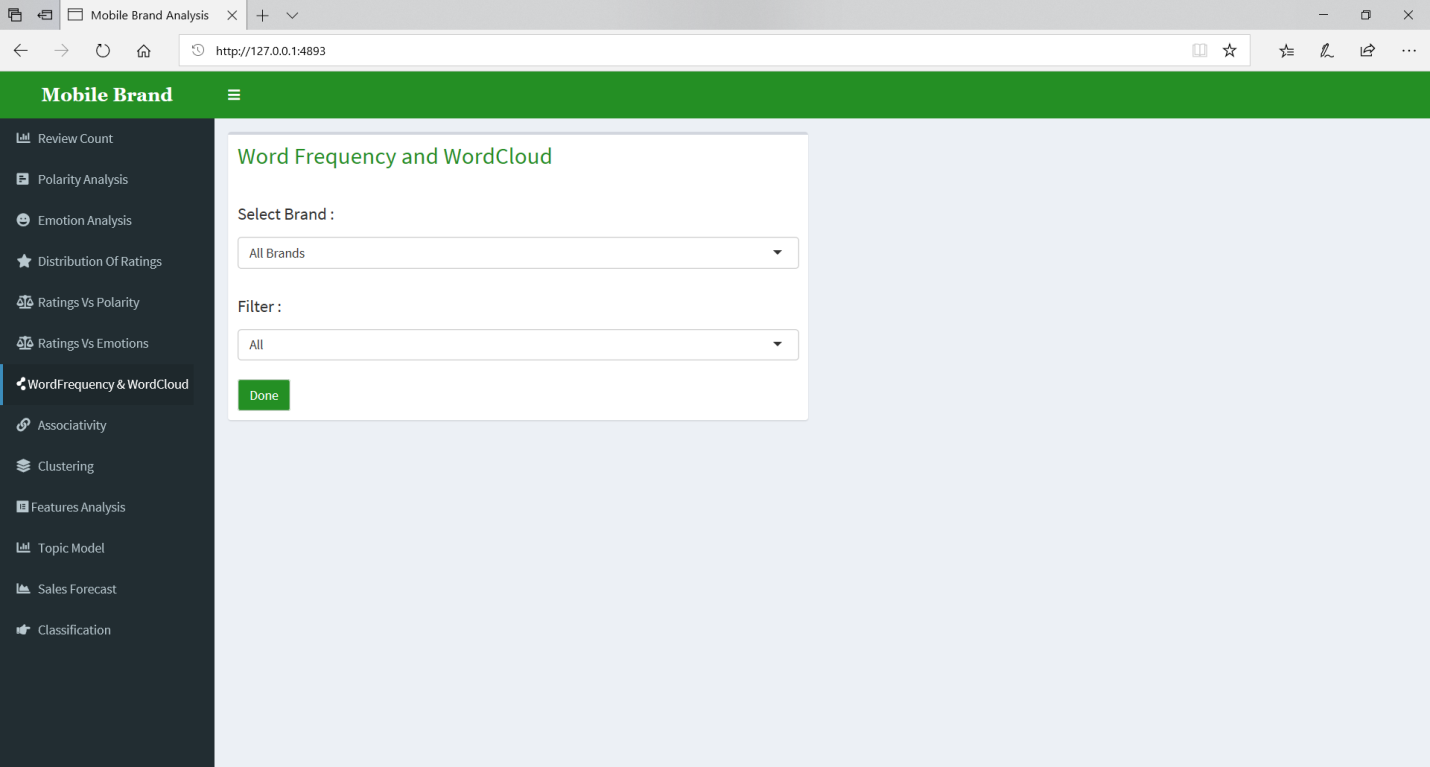
Anger dominated the 1 star rating.

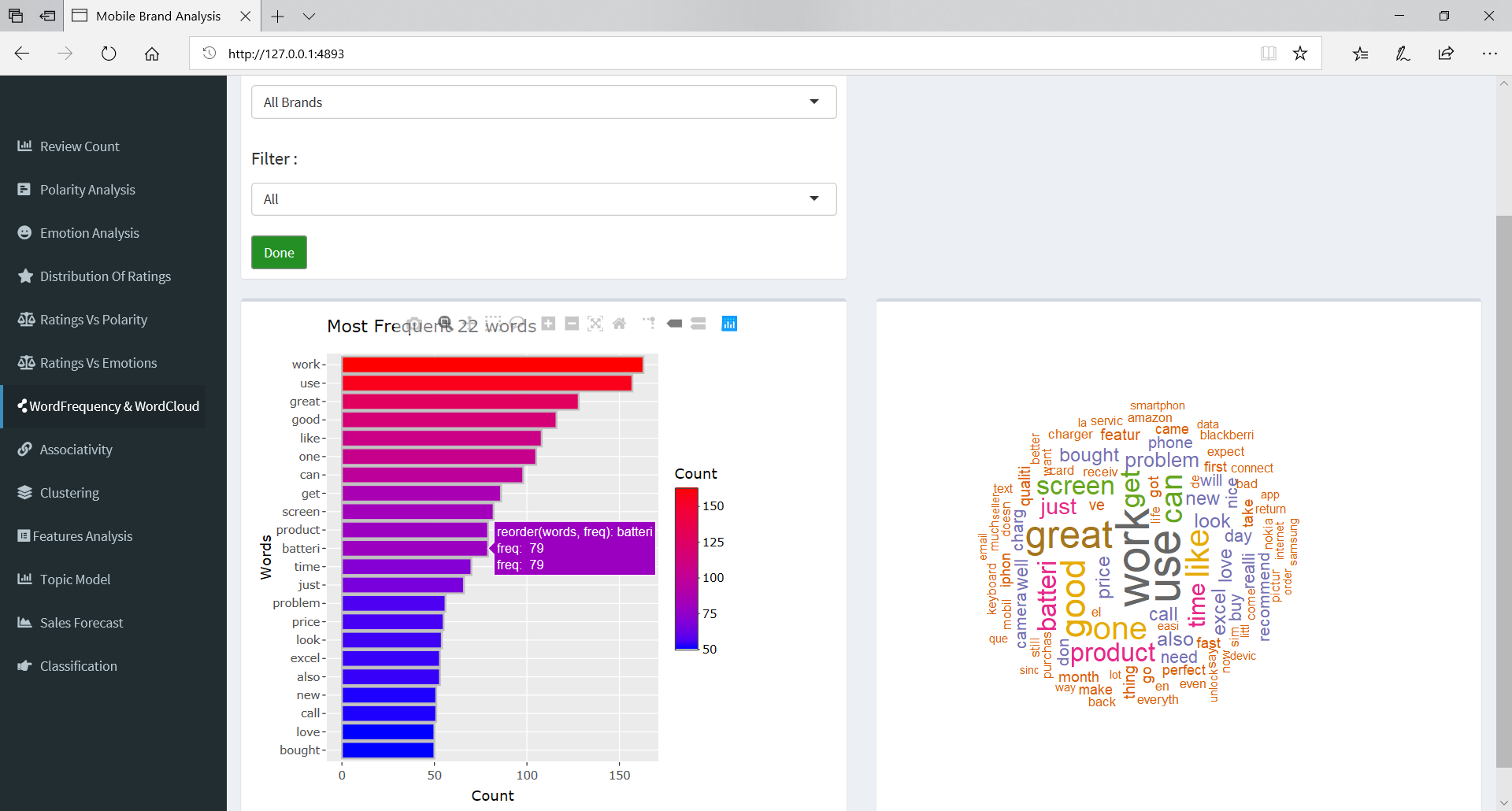
Q] Particular Brand’s (ex, Samsung’s) 5 star rating contains which dominating Emotion?



Joy dominated Samsung’s 5 Star Rating.

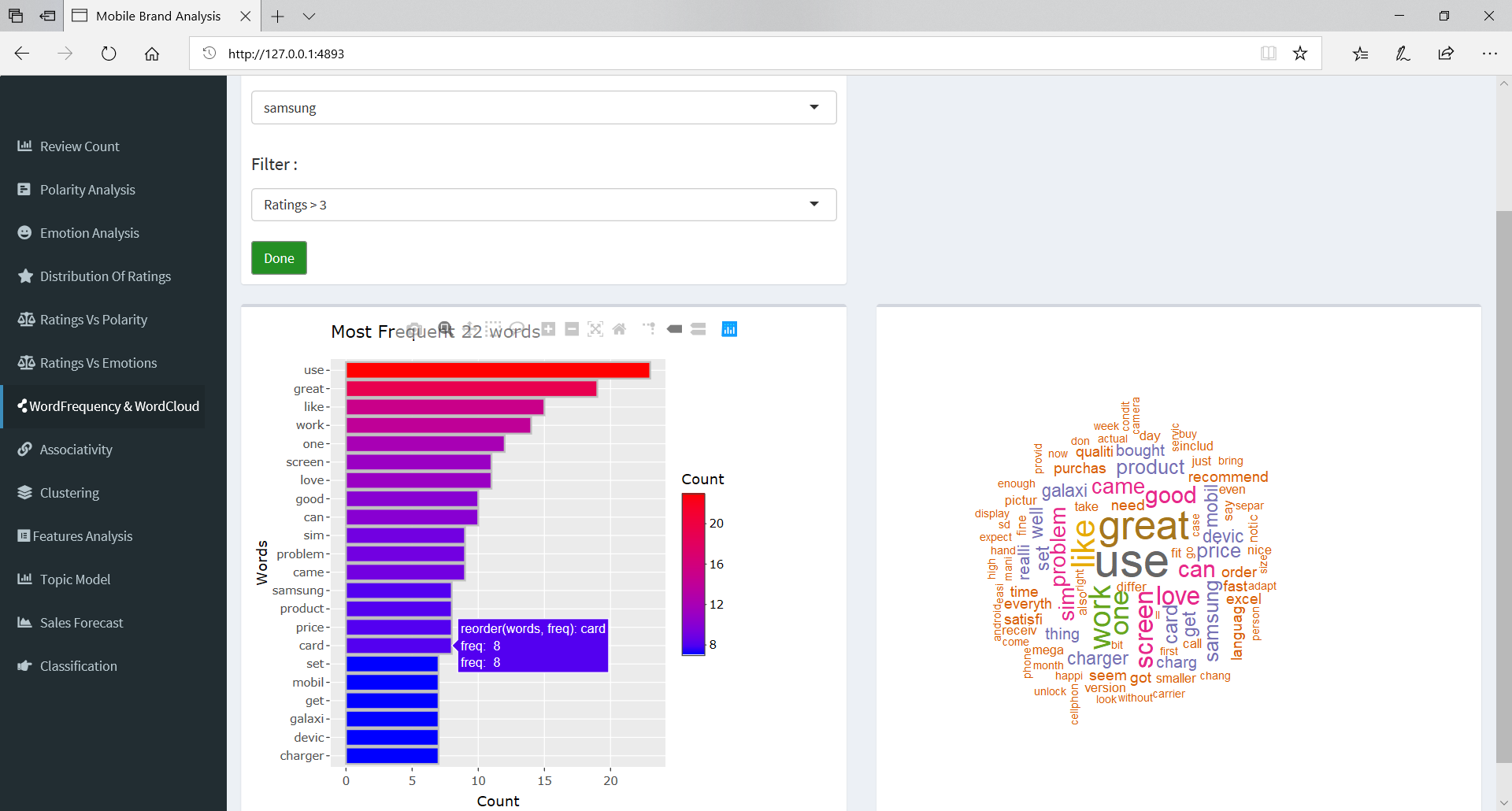
Q] Most Common Words Used in Reviews?





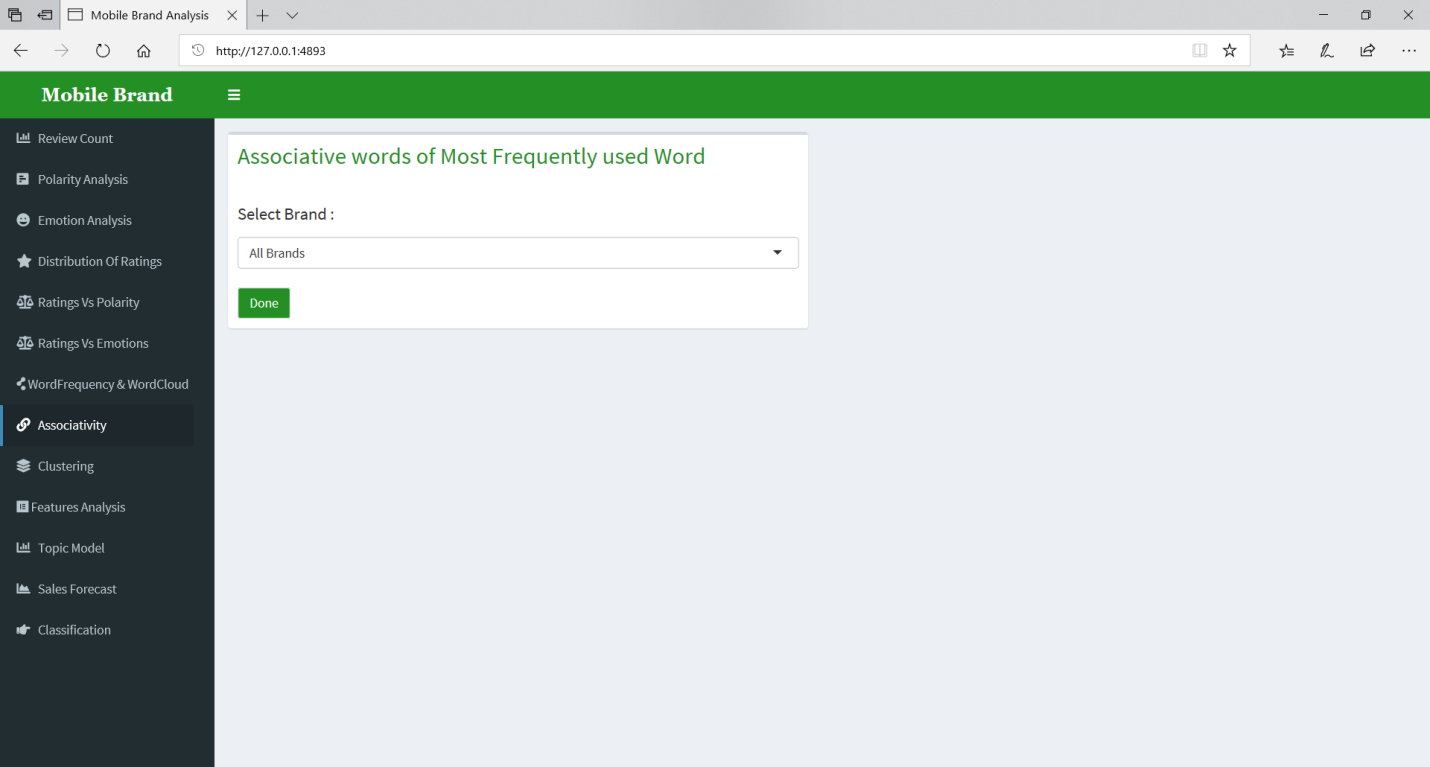
“Work”, “use”, “great” and “good” was commonly used words in reviews. But out of them, “great” and “good” word shows Mobile Brand’s are moving towards positive task.

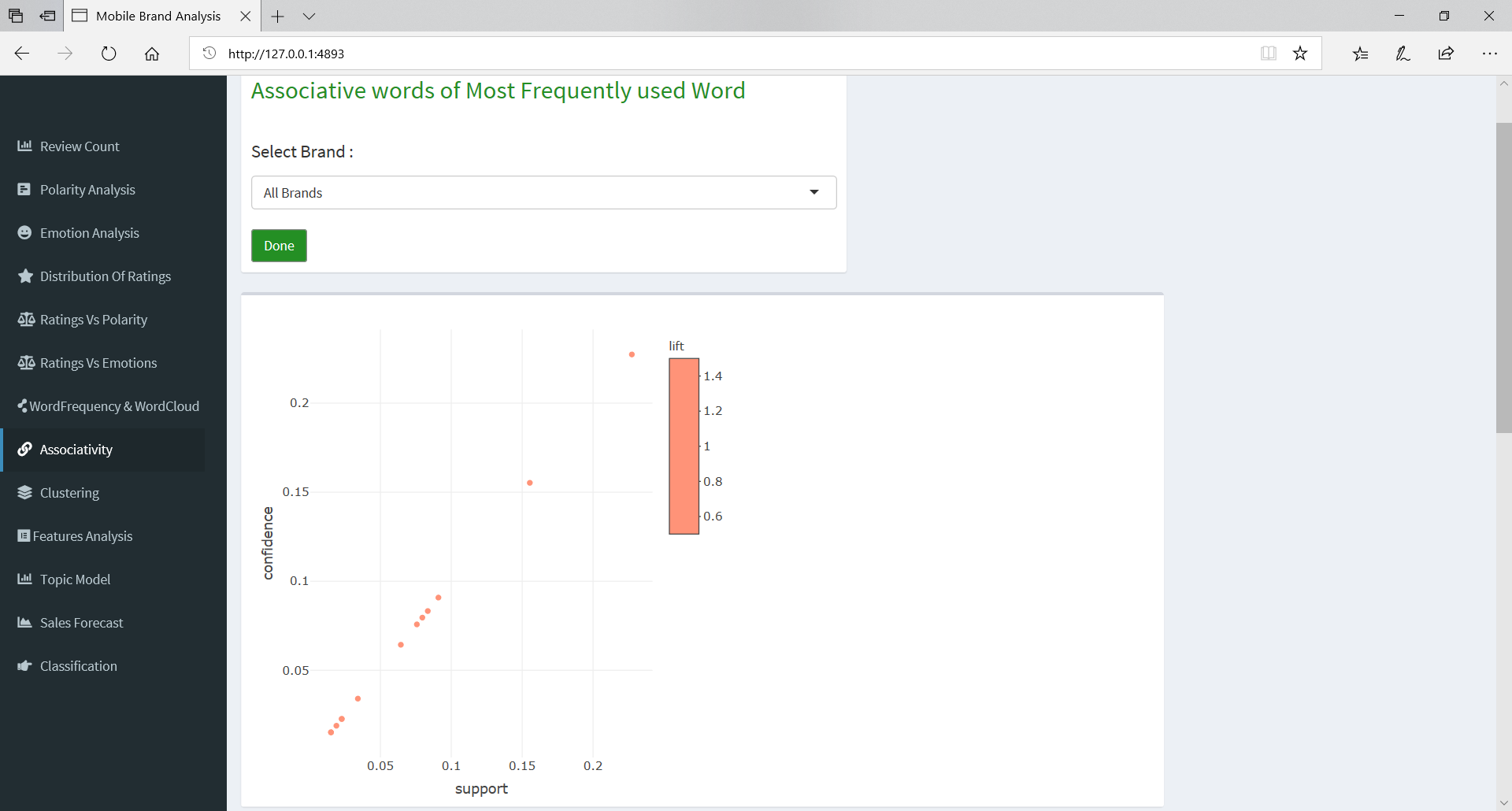
Q] Most Common Positive Words Used in Reviews for Samsung?

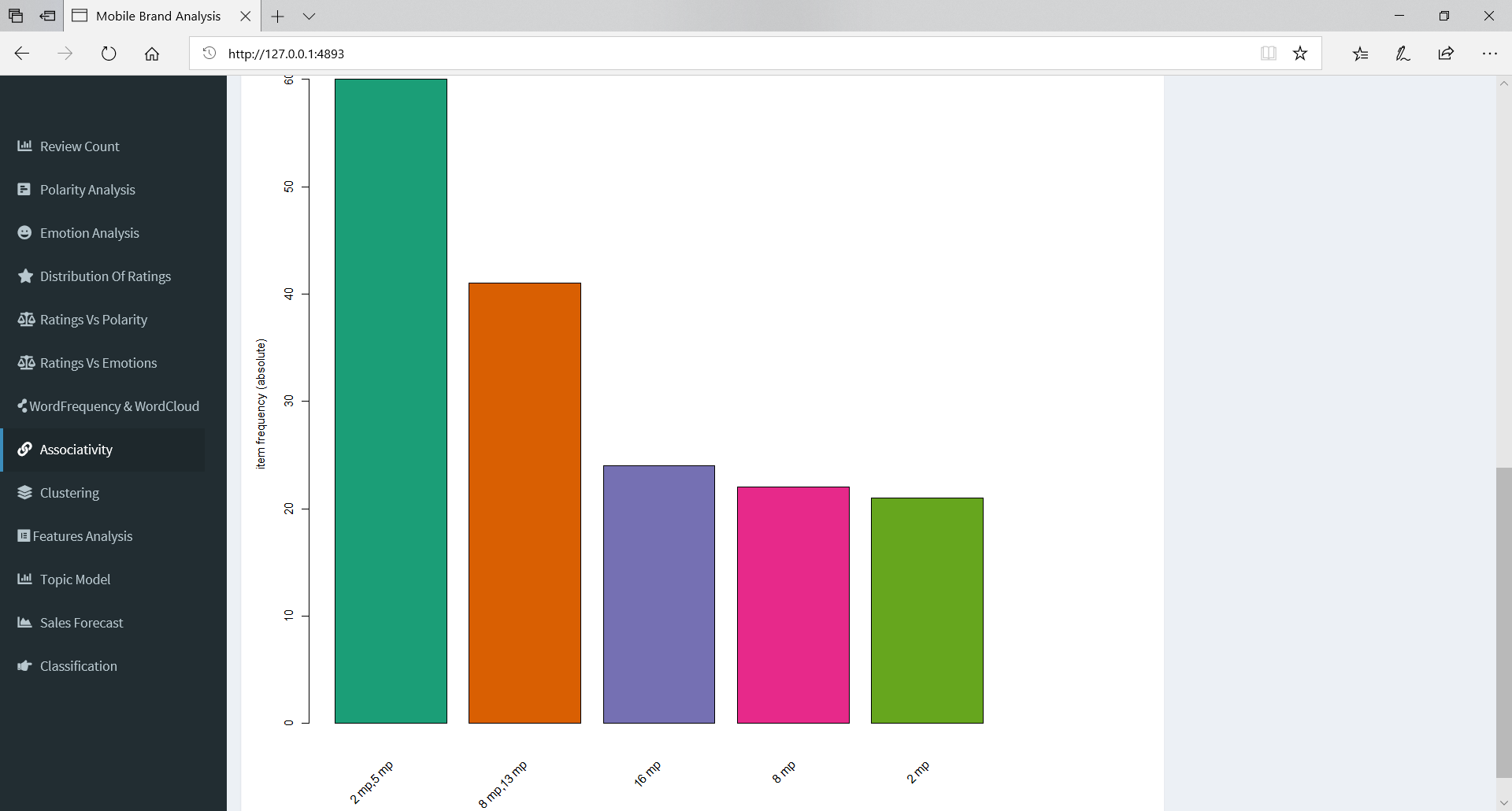


Out of all the word, “screen”, “price”, “galaxy” => (Samsung’s Mobile Series) and “charger” (included in first 22 words) were words that helps to know positive point of Samsung.

Q] Do Customer switch to better Camera Phones?



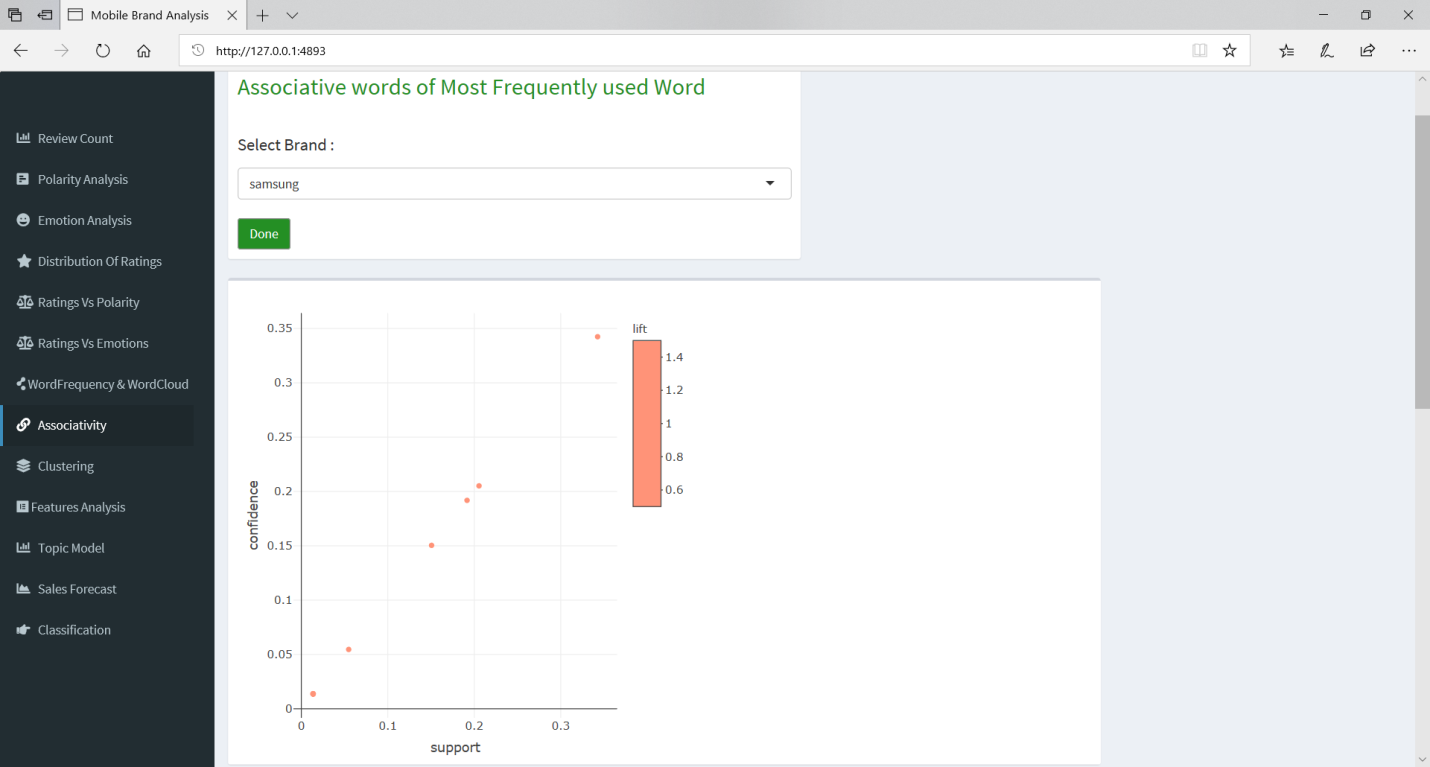


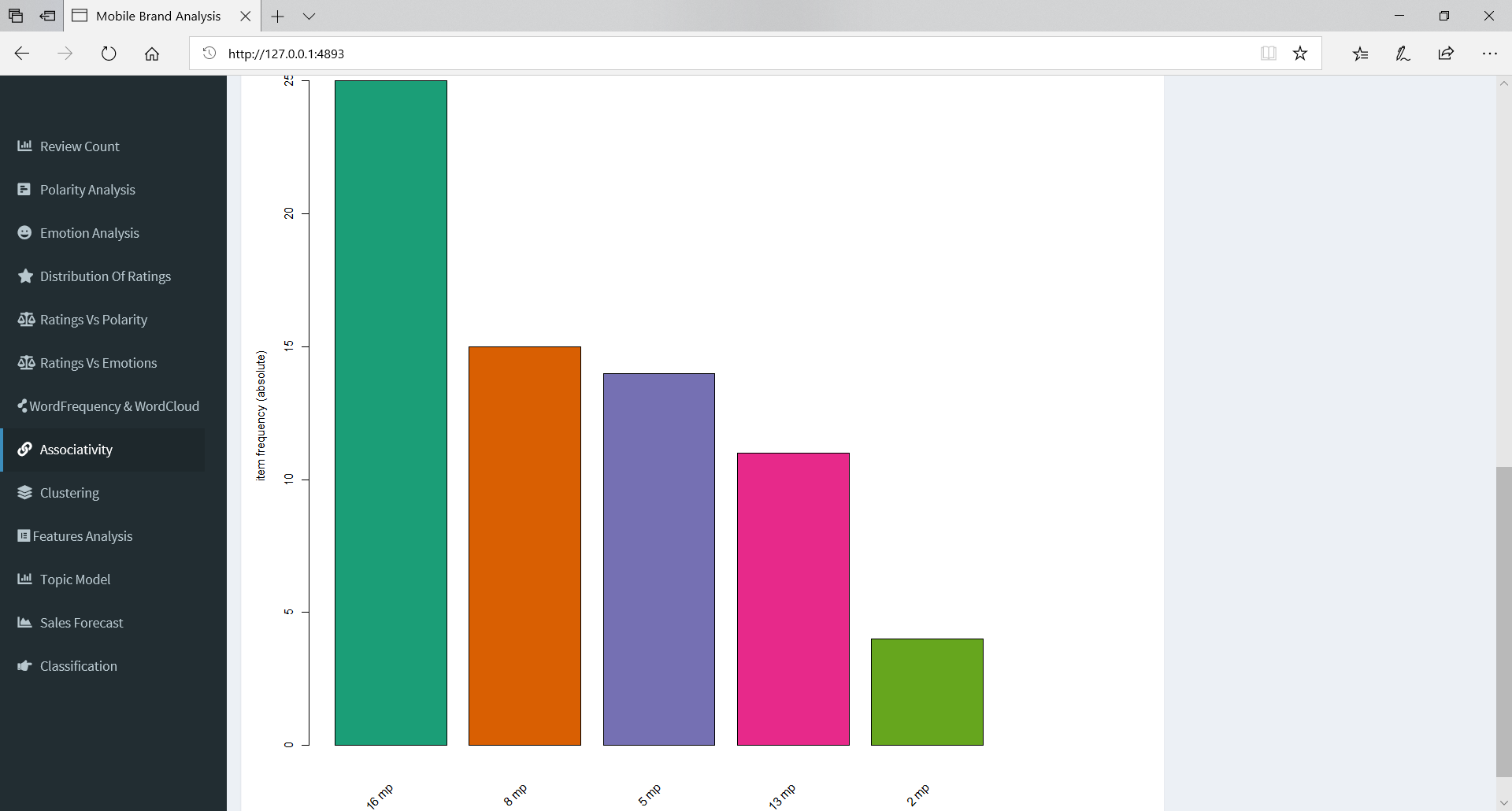


People moved from 2 mp to 5 mp at greater frequency.

But still saying 2mp Phone user will buy 5mp Phone will be wrong assumption, since there was no associativity found.

Q] Which MegaPixel Camera Samsung Phones was most Frequently purchased?





It turned out to be 16 mp camera phones were most purchased.

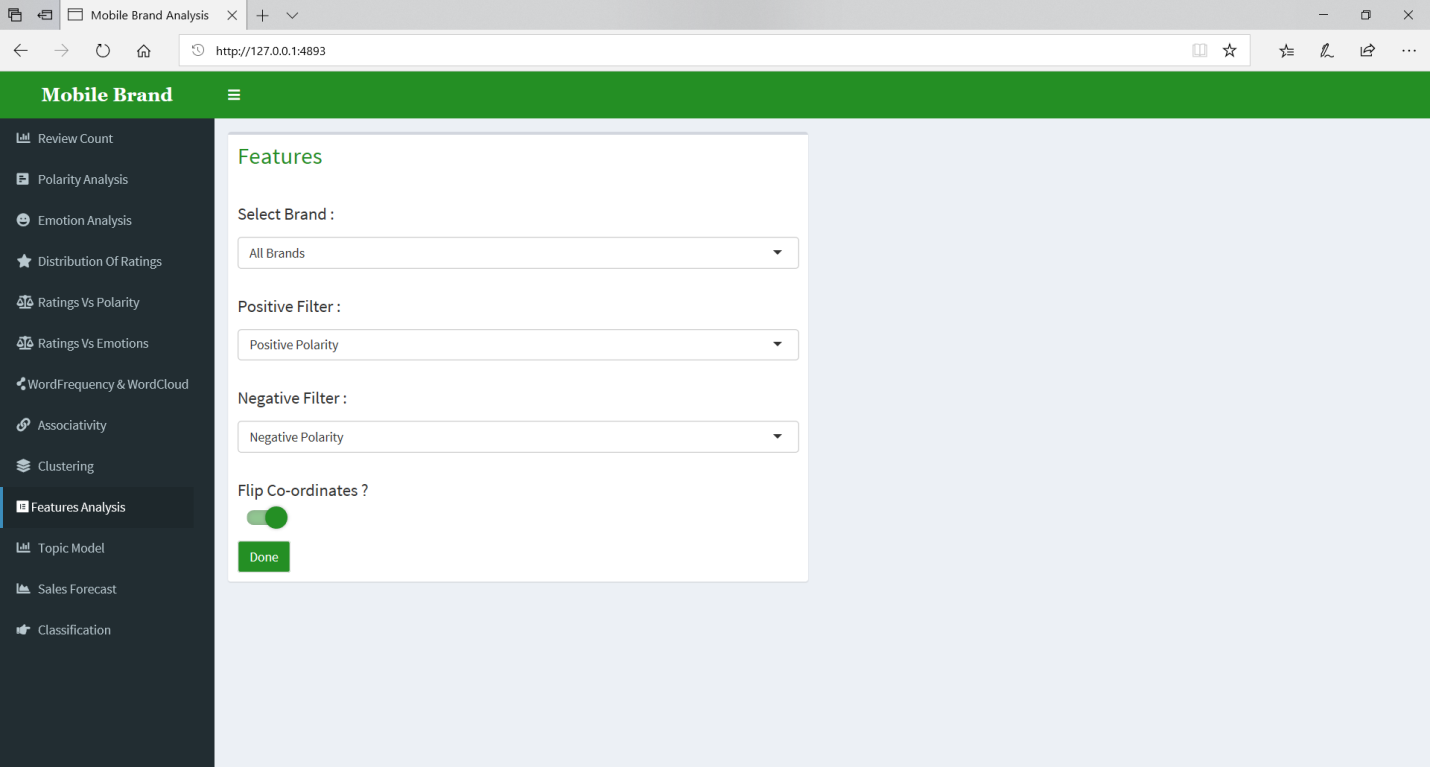
Q] Which Age Group Customer purchase more than 16 mp and price above $400 mobile phones?

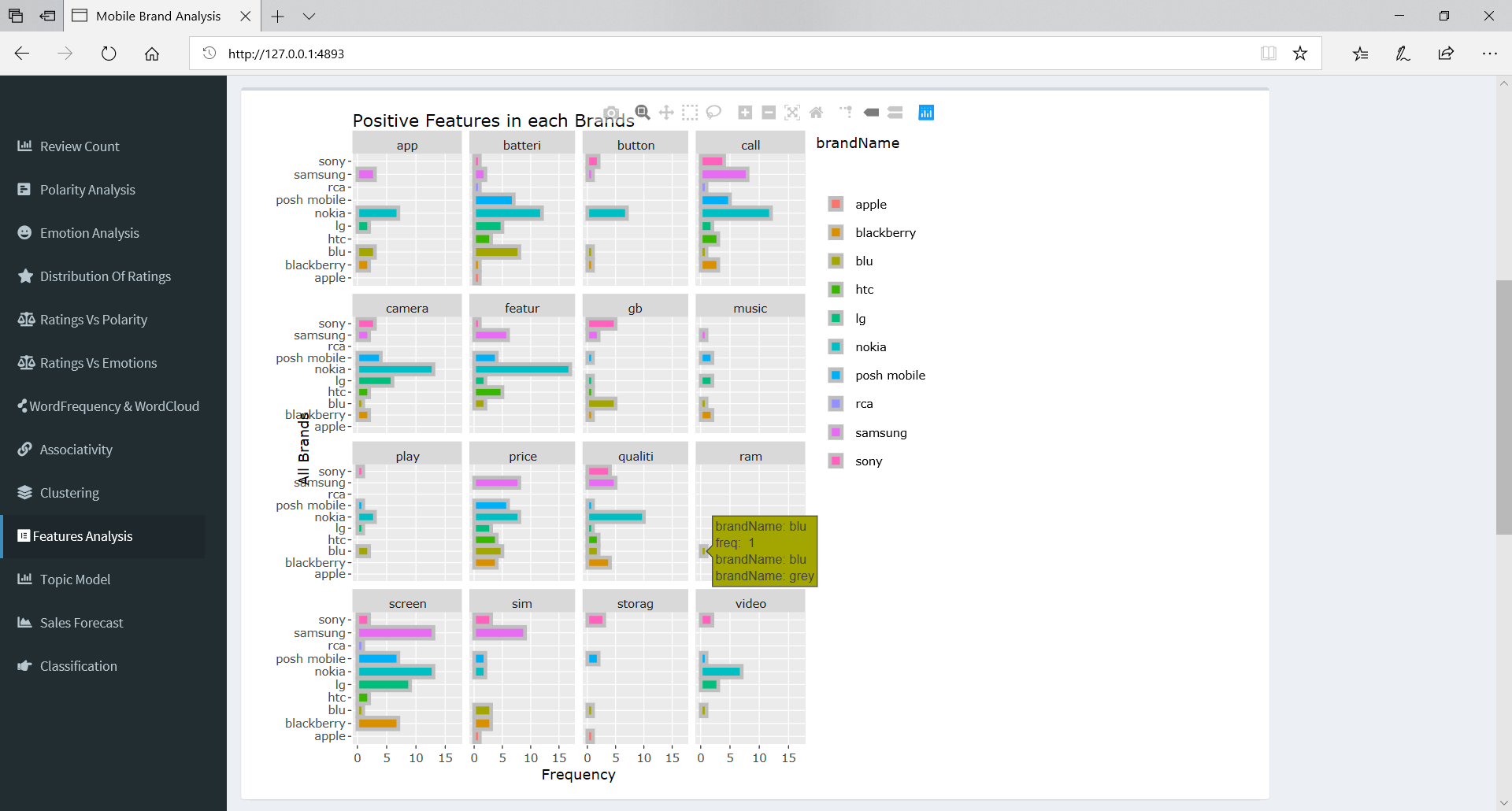




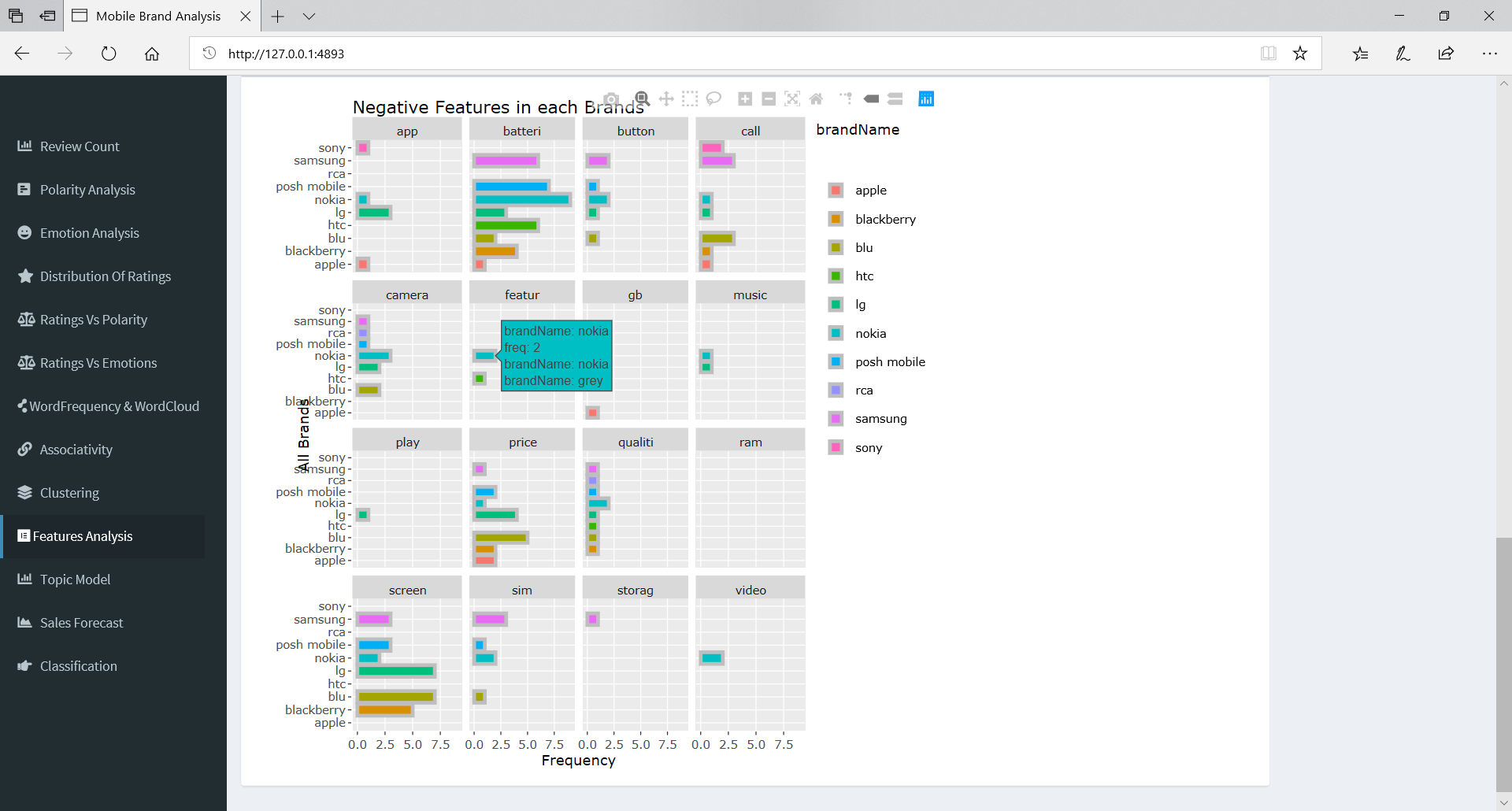
Mobile with camera of more than 16mp and more than $400 has attracted age group of 17 to 30 years of age.

Q] What are Positive and Negative Features in Mobile Brands?



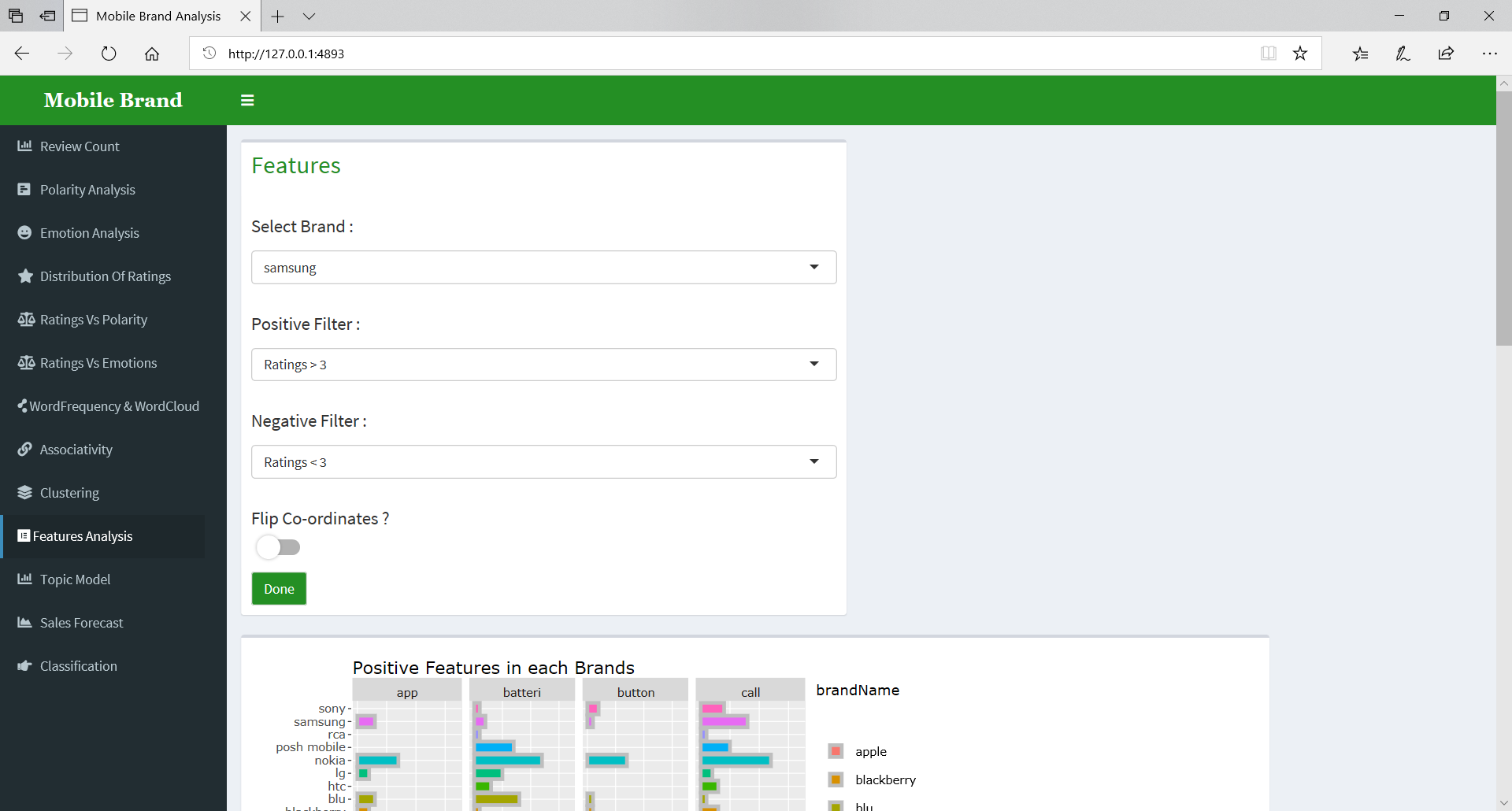


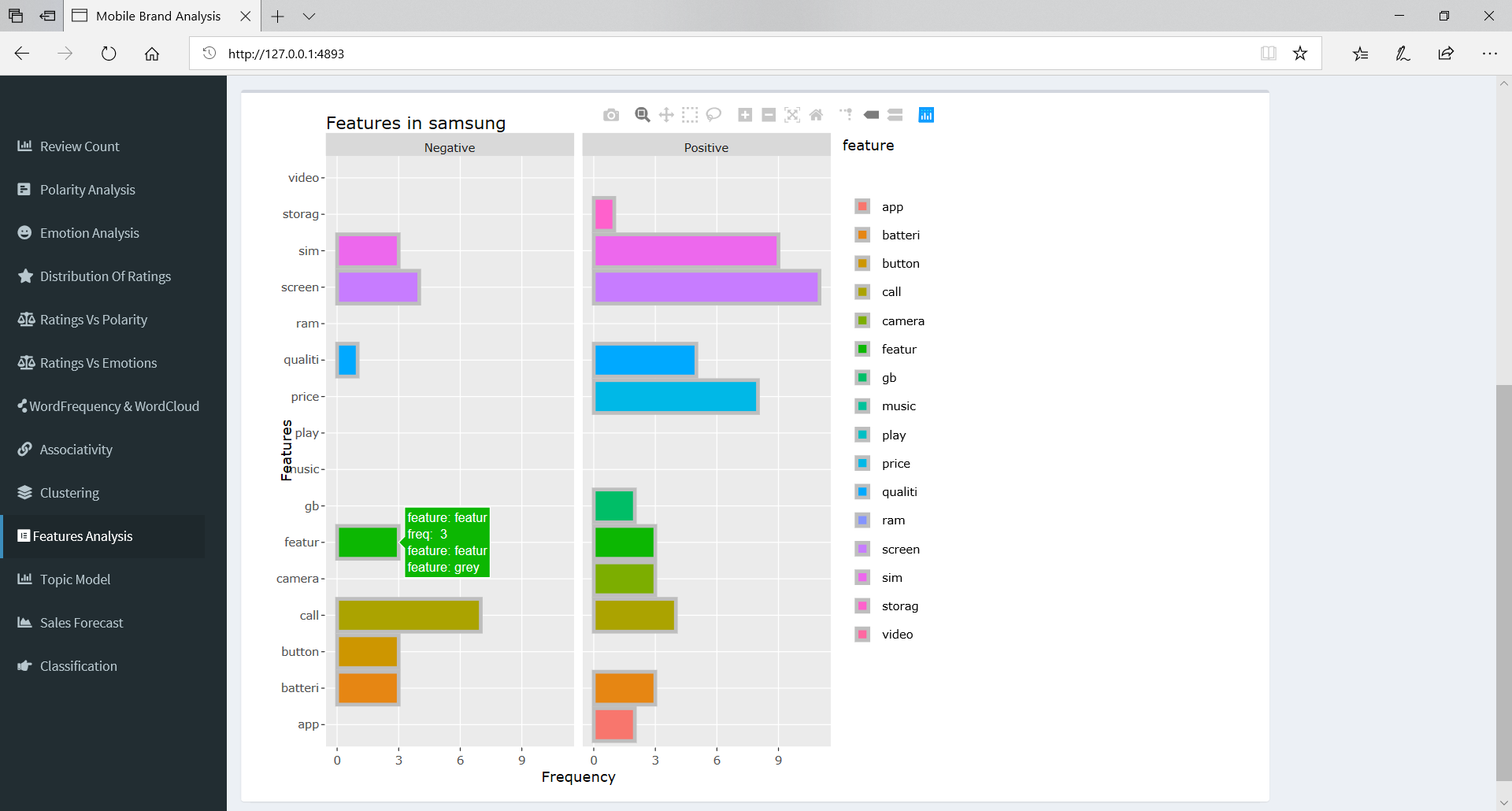
Above are positive feature of all Mobile Brand.



Above are negative feature of all Mobile Brand.

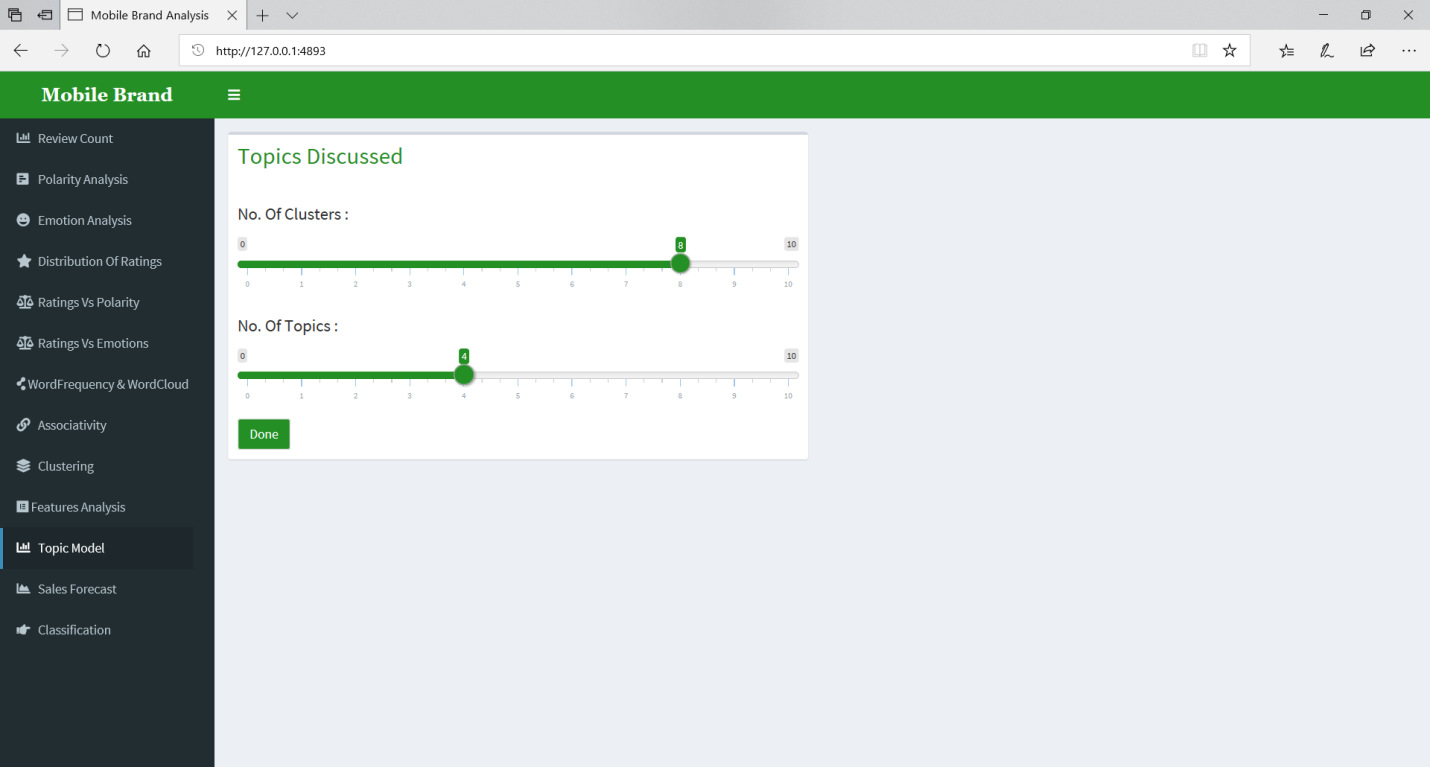
Q] What are Positive and Negative Features in Samsung?

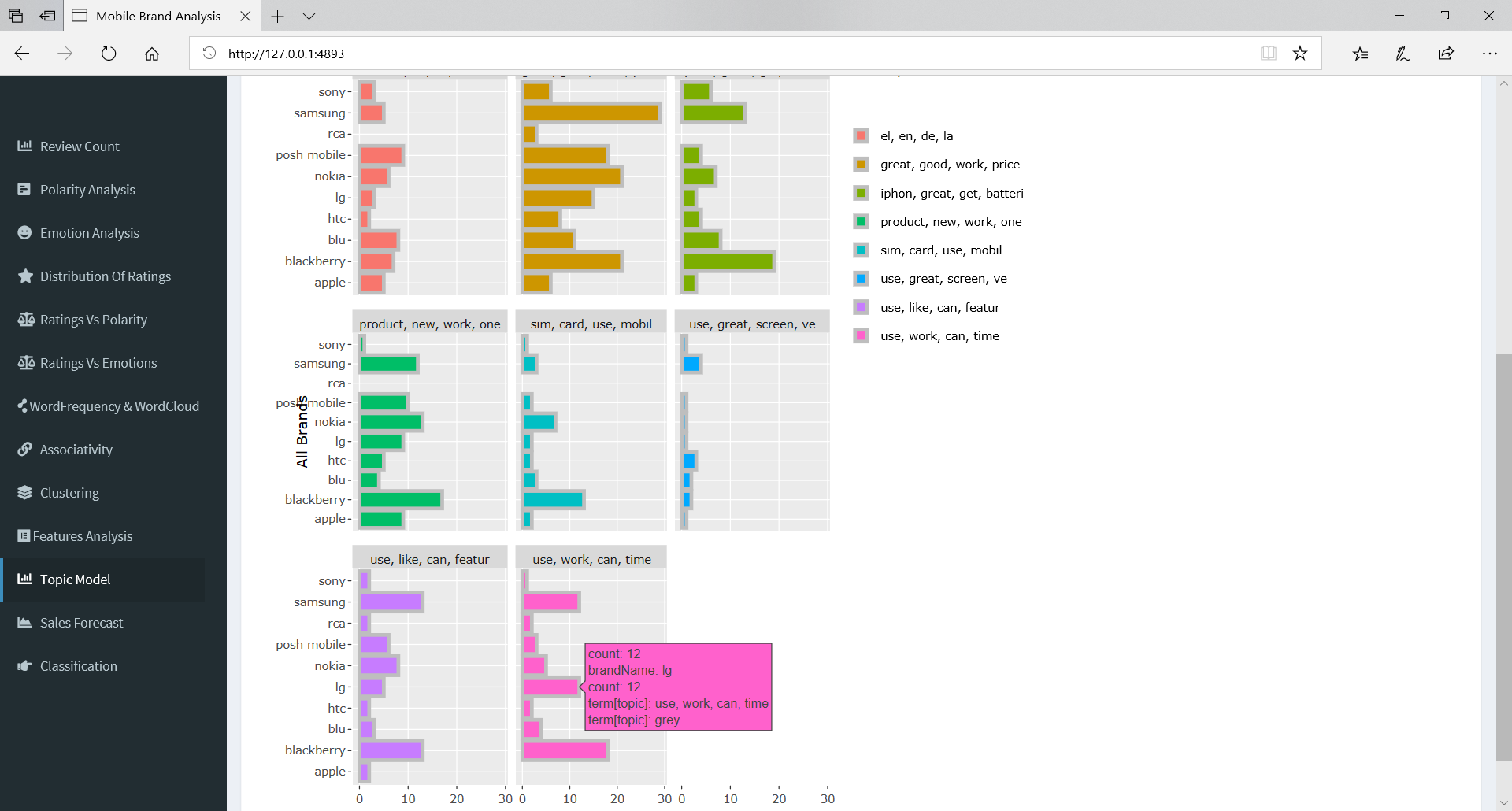




Samsung has best Positive Feature as screen and most Negative Feature as calling experience i.e. maybe sound quality or support for sim.

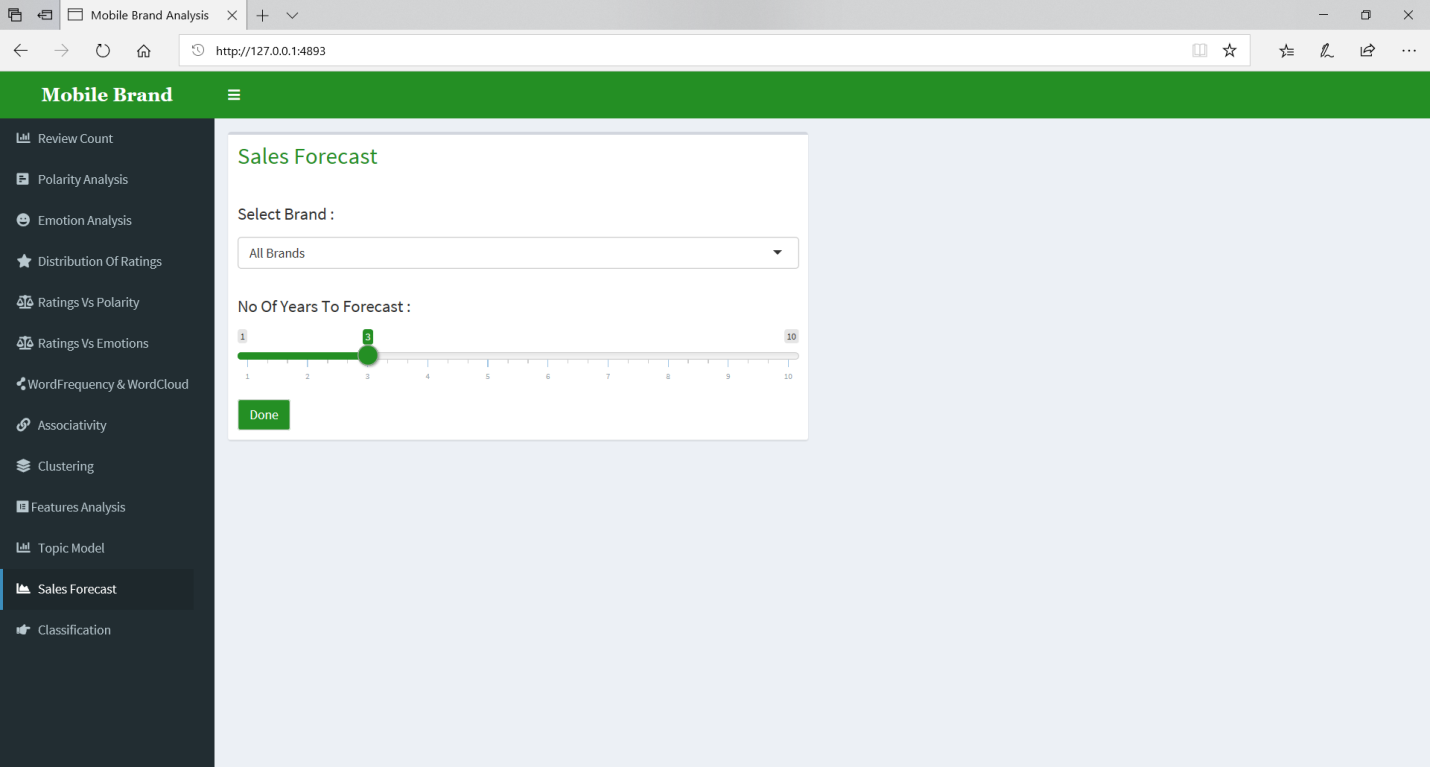
Q] Most common topics in reviews?

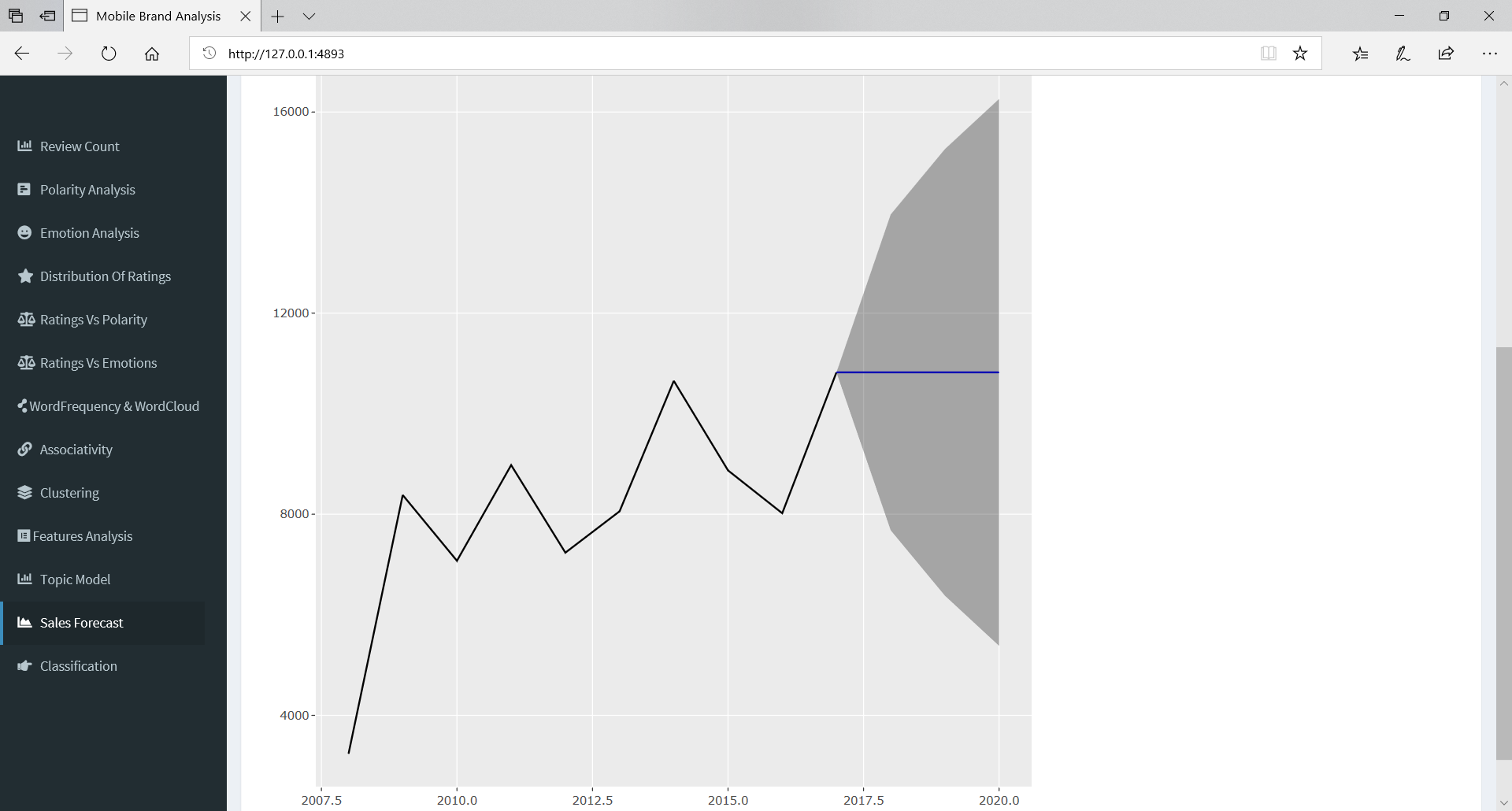




Great, good, work, price is used in single review for second most time. This gives us Idea that Price of Phones has became great maybe cheaper.

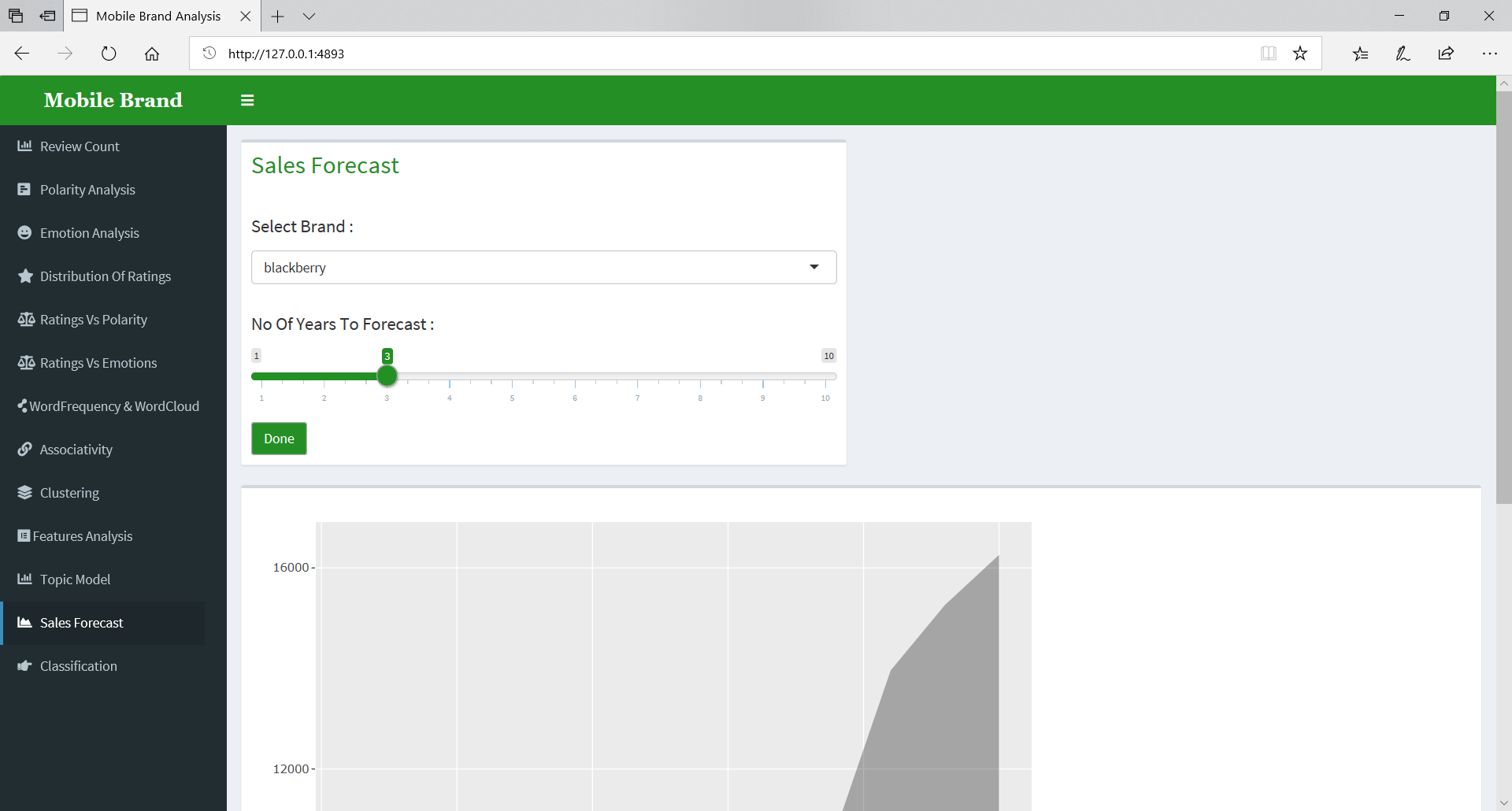
Q] Will Sale increase or decrease in next 3 years?

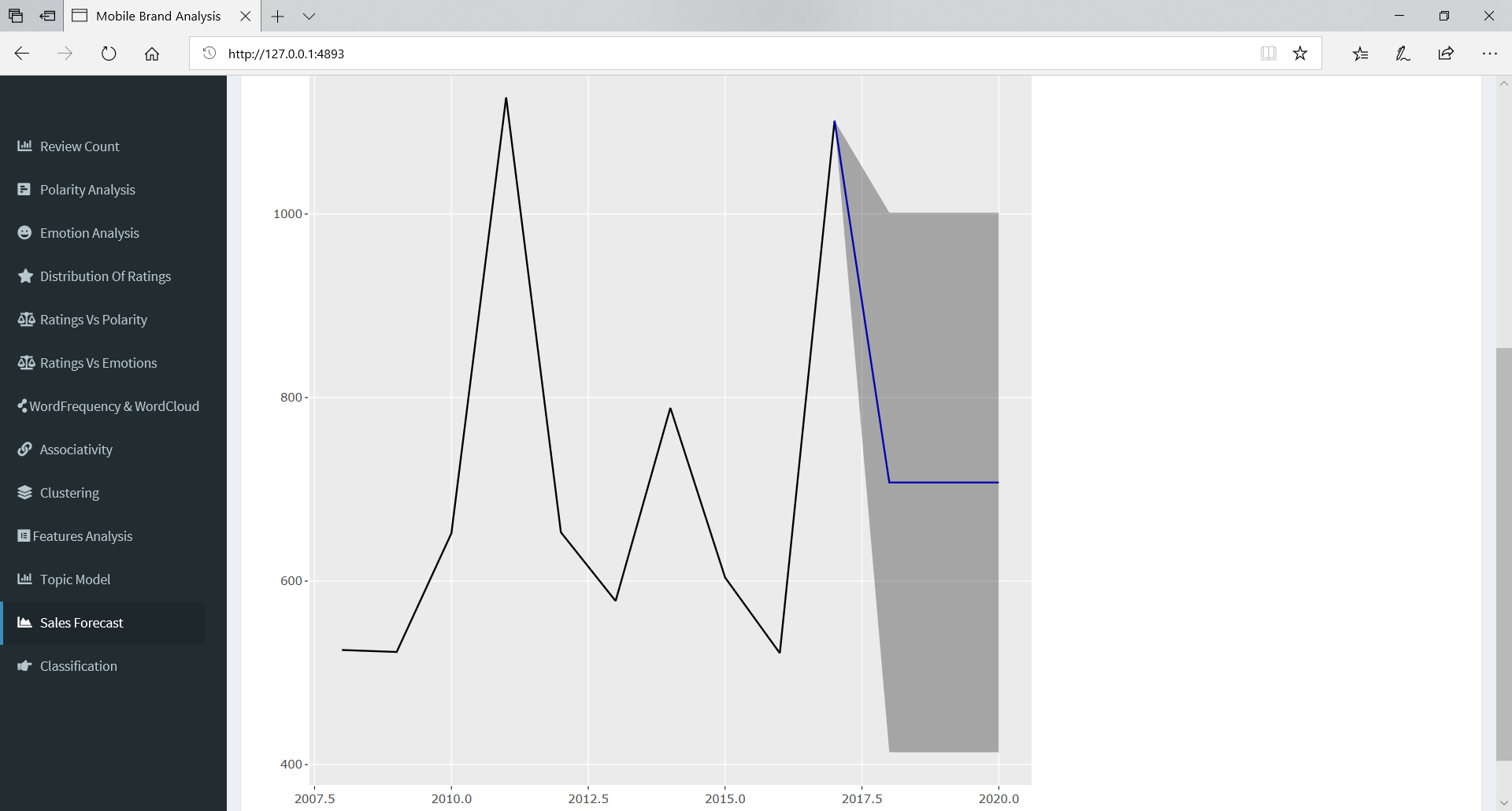




Sales of overall brands may remain constant in next 3 years.

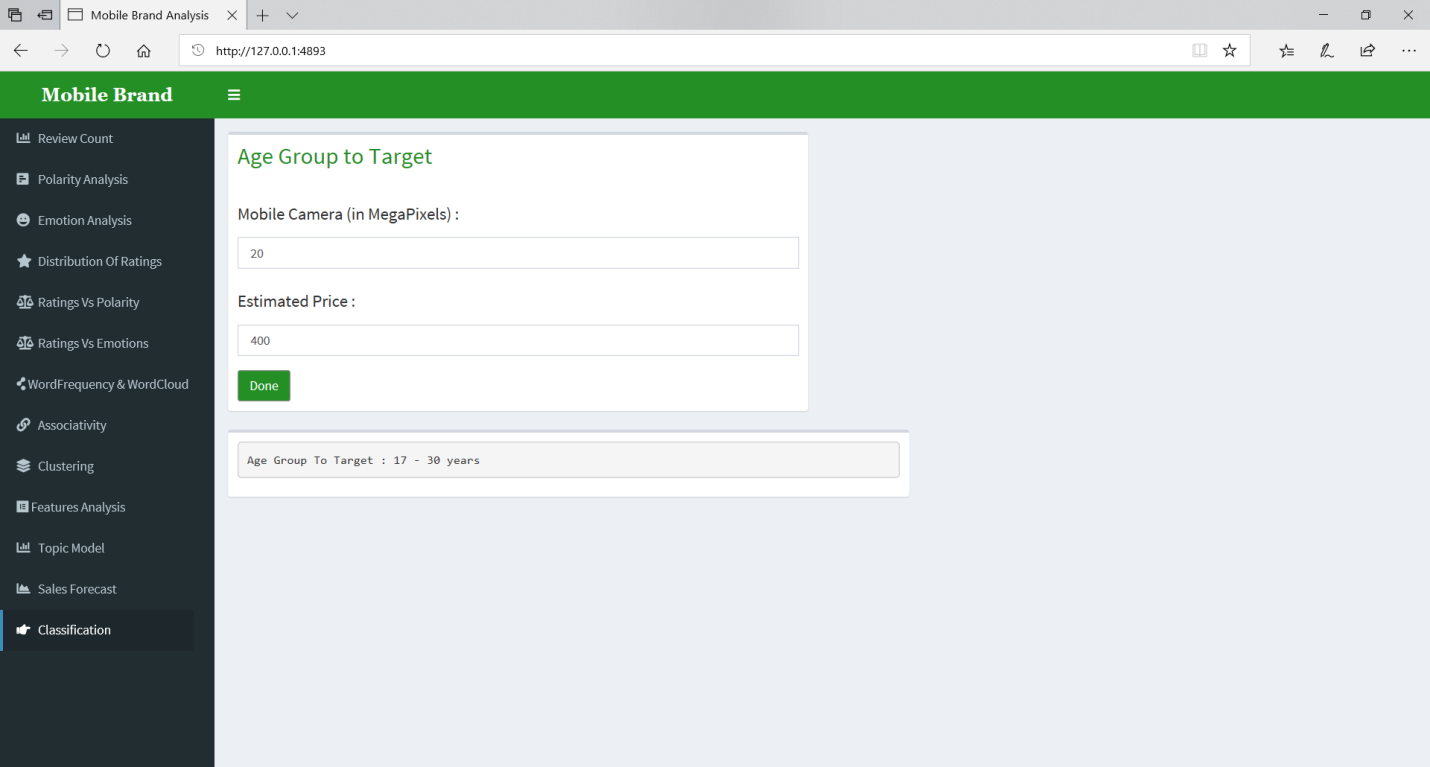
Q] Will Sale increase or decrease in next 3 years for Blackberry?



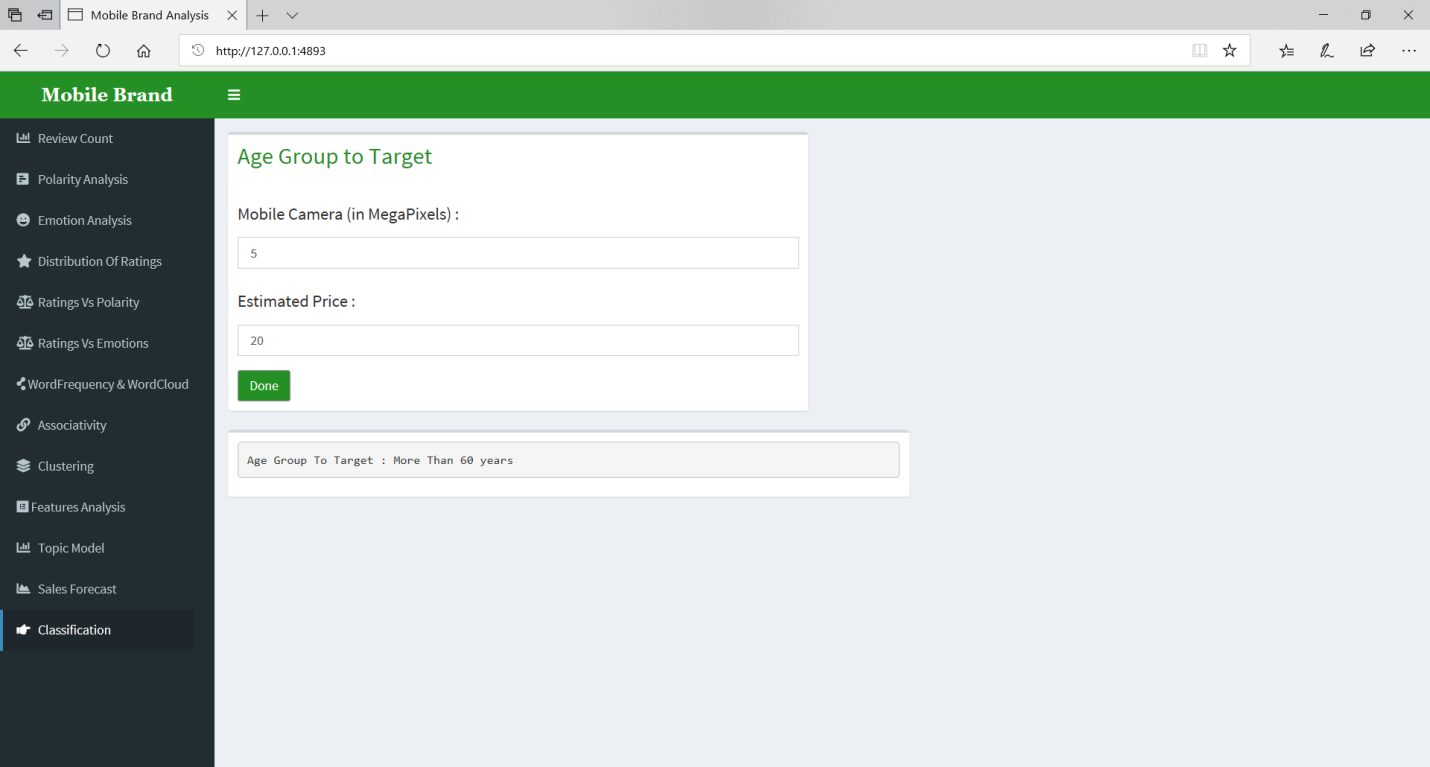


Current trend shows that Blackberry Sales will decline at huge rate.

Q] Which age group to target for Mobile Phone based on camera and price?



17 to 30 years Age Group is likely to buy Mobile Phones with 20 mp camera and $400 estimated price.



More than 60 years Age Group is likely to buy Mobile Phones with 5 mp camera and $20 estimated price.

**CODE**

**MobileMethods.R**

library(tm)

library(SnowballC)

library(wordcloud)

library(RColorBrewer)

library(plyr)

library(arules)

library(arulesViz)

library(RColorBrewer)

library(ggplot2)

library(graph)

library(Rgraphviz)

library(topicmodels)

library(ggConvexHull)

library(caTools)

library(naivebayes)

library(forecast)

library(ggfortify)

library(caret)

############################################################################

polarityOfReviewEach <- function(eachBrandData, reverse){

plot <- ggplot(data = eachBrandData, aes(x=Polarity,fill=Polarity)) +

geom\_bar(stat = "count",size = 1) +

xlab("Polarity") + ylab("Polarity Count")

if(reverse){

plot <- plot + coord\_flip() + theme(legend.position = "top")

}

plot

}

polarityOfReviewAll <- function(allBrandData, reverse){

plot <- ggplot(data = allBrandData, aes(x=Brand.Name,fill=Polarity)) +

geom\_bar(stat = "count",size = 1) +

xlab("All Brands") + ylab("Polarity Count")

if(reverse){

plot <- plot + coord\_flip() + theme(legend.position = "top")

}

plot

}

emotionOfReviewEach <- function(eachBrandData, reverse){

plot <- ggplot(data = eachBrandData, aes(x=Emotions,fill=Emotions)) +

geom\_bar(stat = "count",size = 1) +

xlab("Emotions") + ylab("Polarity Count")

if(reverse){

plot <- plot + coord\_flip() + theme(legend.position = "top")

}

plot

}

emotionOfReviewAll <- function(allBrandData, reverse){

plot <- ggplot(data = allBrandData, aes(x=Brand.Name,fill=Emotions)) +

geom\_bar(stat = "count",size = 1) +

xlab("All Brands") + ylab("Emotion Count")

if(reverse){

plot <- plot + coord\_flip() + theme(legend.position = "top")

}

plot

}

distributionOfRatingsEach <- function(eachBrandData, reverse){

plot <- ggplot(eachBrandData, aes(x = Rating, y = ..count../tapply(..count..,..PANEL..,sum)[..PANEL..],

fill = Brand.Name )) + geom\_freqpoly(binwidth = 1, col = "blue") +

xlab("Rating") + ylab("Percent") +

scale\_y\_continuous(labels = scales::percent) +

scale\_x\_continuous(breaks = c(1,2,3,4,5))

if(reverse){

plot <- plot + coord\_flip() + theme(legend.position = "top")

}

plot

}

distributionOfRatingsAll <- function(allBrandData, reverse){

plot <- ggplot(allBrandData, aes(x = Rating, y = ..count../tapply(..count..,..PANEL..,sum)[..PANEL..],

fill = Brand.Name, color = Brand.Name)) +

geom\_bar(stat = "count", size = 1)+

xlab("Rating") + ylab("Percent") +

scale\_y\_continuous(labels = scales::percent) +

scale\_x\_continuous(breaks = c(1,2,3,4,5))

if(reverse){

plot <- plot + coord\_flip() + theme(legend.position = "top")

}

plot

}

distributionOfRatingsbyPolarityEach <- function(eachBrandData, reverse){

plot <- ggplot(eachBrandData, aes(x = Rating, y = ..count../tapply(..count..,..PANEL..,sum)[..PANEL..],

fill = Polarity, color = Polarity )) +

geom\_freqpoly(binwidth = 1) +

xlab("Rating") + ylab("Percent") +

scale\_y\_continuous(labels = scales::percent) +

scale\_x\_continuous(breaks = c(1,2,3,4,5))

if(reverse){

plot <- plot + coord\_flip() + theme(legend.position = "top")

}

plot

}

distributionOfRatingsbyPolarityAll <- function(allBrandData, reverse){

plot <- ggplot(allBrandData, aes(x = Rating, y = ..count../tapply(..count..,..PANEL..,sum)[..PANEL..],

fill = Polarity, color = Polarity )) +

geom\_freqpoly(binwidth = 1) +

xlab("Rating") + ylab("Percent") +

scale\_y\_continuous(labels = scales::percent) +

scale\_x\_continuous(breaks = c(1,2,3,4,5)) +

facet\_wrap(~Brand.Name)

if(reverse){

plot <- plot + coord\_flip() + theme(legend.position = "top")

}

plot

}

distributionOfRatingsbyEmotionEach <- function(eachBrandData, reverse){

plot <- ggplot(eachBrandData, aes(x = Rating, y = ..count../tapply(..count..,..PANEL..,sum)[..PANEL..],

fill = Emotions, color = Emotions )) +

geom\_freqpoly(binwidth = 1) +

xlab("Rating") + ylab("Percent") +

scale\_y\_continuous(labels = scales::percent) +

scale\_x\_continuous(breaks = c(1,2,3,4,5))

if(reverse){

plot <- plot + coord\_flip() + theme(legend.position = "top")

}

plot

}

distributionOfRatingsbyEmotionAll <- function(allBrandData, reverse){

plot <- ggplot(allBrandData, aes(x = Rating, y = ..count../tapply(..count..,..PANEL..,sum)[..PANEL..],

fill = Emotions, color = Emotions )) +

geom\_freqpoly(binwidth = 1) +

xlab("Rating") + ylab("Percent") +

scale\_y\_continuous(labels = scales::percent) +

scale\_x\_continuous(breaks = c(1,2,3,4,5)) +

facet\_wrap(~Brand.Name)

if(reverse){

plot <- plot + coord\_flip() + theme(legend.position = "top")

}

plot

}

############################################################################

getData <- function(brandData, brand\_Name){

if(brand\_Name == "All Brands"){

brandData

}else{

brandData[brandData$Brand.Name == brand\_Name,]

}

}

getFilteredData <- function(brandData, filterCategory, brand\_Name){

dataset <- getData(brandData, brand\_Name)

if(filterCategory == "All"){

dataset

}else if(filterCategory == "Positive Polarity"){

dataset[dataset$Polarity == "Positive", ]

}else if(filterCategory == "Negative Polarity"){

dataset[dataset$Polarity == "Negative", ]

}else if(filterCategory == "Ratings > 3"){

dataset[dataset$Rating > 3,]

}else if(filterCategory == "Ratings < 3"){

dataset[dataset$Rating < 3,]

}

}

noOfReview <- function(allBrandData){

brandReviewData <- allBrandData %>%

group\_by(Brand.Name)%>%

summarise(Count = n())

plot\_ly(brandReviewData, labels = ~Brand.Name, values = ~Count) %>% add\_pie(hole = 0.6)

}

polarityOfReview <- function(brandData, reverse, brand\_Name){

brandData <- getData(brandData, brand\_Name)

if(brand\_Name == "All Brands"){

plot <- polarityOfReviewAll(brandData, reverse)

}else{

plot <- polarityOfReviewEach(brandData, reverse)

}

plot

}

emotionOfReview <- function(brandData, reverse, brand\_Name){

brandData <- getData(brandData, brand\_Name)

if(brand\_Name == "All Brands"){

plot <- emotionOfReviewAll(brandData, reverse)

}else{

plot <- emotionOfReviewEach(brandData, reverse)

}

plot

}

distributionOfRatings <- function(brandData, reverse, brand\_Name){

brandData <- getData(brandData, brand\_Name)

if(brand\_Name == "All Brands"){

plot <- distributionOfRatingsAll(brandData, reverse)

}else{

plot <- distributionOfRatingsEach(brandData, reverse)

}

plot

}

distributionOfRatingsbyPolarity <- function(brandData, reverse, brand\_Name){

brandData <- getData(brandData, brand\_Name)

if(brand\_Name == "All Brands"){

plot <- distributionOfRatingsbyPolarityAll(brandData, reverse)

}else{

plot <- distributionOfRatingsbyPolarityEach(brandData, reverse)

}

plot

}

confusionMatrixPlot <- function(dataSample, brand\_Name){

dataSample <- getData(dataSample, brand\_Name)

polaritySample <- dataSample[dataSample$Rating > 3 | dataSample$Rating < 3,]

polaritySample$ActualPolarity <- ""

polaritySample[polaritySample$Rating > 3,]$ActualPolarity <- "Positive"

polaritySample[polaritySample$Rating < 3,]$ActualPolarity <- "Negative"

confusionMat <- confusionMatrix(as.factor(polaritySample$Polarity), as.factor(polaritySample$ActualPolarity))

fourfoldplot(confusionMat$table,

conf.level = 0, margin = 1, main = "Confusion Matrix")

}

distributionOfRatingsbyEmotion <- function(brandData, reverse, brand\_Name){

brandData <- getData(brandData, brand\_Name)

if(brand\_Name == "All Brands"){

plot <- distributionOfRatingsbyEmotionAll(brandData, reverse)

}else{

plot <- distributionOfRatingsbyEmotionEach(brandData, reverse)

}

plot

}

############################################################################

removeNonASCII <- function(reviewList){

removeRule <- grep("reviewList", iconv(reviewList, "latin1", "ASCII", sub="reviewList"))

reviewList[-removeRule]

}

getTermDocMatrix <- function(dataSet){

sortedData <- dataSet[order(dataSet$Review.Votes,decreasing = TRUE),]

sampleReview <- sortedData$Reviews

reviewsDF <- data.frame(sampleReview)

reviewForCorpus <- reviewsDF$sampleReview

corpus <- Corpus(VectorSource(reviewForCorpus))

corpus <- tm\_map(corpus, removeWords, c("phone", stopwords("english")))

trans <- content\_transformer(function(x,pattern)gsub(pattern," ",x))

corpus <- tm\_map(corpus, stemDocument)

corpus <- tm\_map(corpus,trans,"\\s+\\w\\s+")

termMatrix <- TermDocumentMatrix(corpus, control = list(wordLengths = c(1, Inf)))

termMatrix

}

getDataFrameMatrix <- function(termMatrix){

matrix <- as.matrix(termMatrix)

sortMatrix <- sort(rowSums(matrix),decreasing=TRUE)

dataFrameMatrix <- data.frame(words = names(sortMatrix),freq = sortMatrix)

dataFrameMatrix

}

############################################################################

wordFreqGraph <- function (dataFrameMatrix){

ggplot(dataFrameMatrix[1:22,], aes(x = reorder(words, freq), y = freq, fill = freq)) +

geom\_bar(stat = "identity", col = "grey") +

xlab("Words") + ylab("Count") +

ggtitle("Most Frequent 22 words") +

coord\_flip() +

scale\_fill\_gradient("Count", low = "blue", high = "red")

}

wordCloudGraph <- function(dataFrameMatrix){

wordcloud(dataFrameMatrix$words,dataFrameMatrix$freq, max.words=97,random.order=FALSE,

rot.per=0.35,colors=brewer.pal(8,"Dark2"))

}

topicModelGraph <- function(dataSet, termMatrix, noOfCluster, noOfTopics){

docTermMatrix <- as.DocumentTermMatrix(termMatrix)

wordSumDoc <- apply(docTermMatrix , 1, sum)

docTermMatrix <- docTermMatrix[wordSumDoc > 0, ]

docs <- as.integer(docTermMatrix[["dimnames"]][["Docs"]])

lda <- LDA(docTermMatrix, k = noOfCluster)

term <- terms(lda, noOfTopics)

term <- apply(term, MARGIN = 2, paste, collapse = ", ")

topic <- topics(lda, 1)

topics <- data.frame(brandName = dataSet[docs,]$Brand.Name, topic)

plot <- ggplot(data = topics, aes(x=brandName, y = ..count.., fill=term[topic], color = term[topic])) +

geom\_bar(stat = "count", size = 1, col = "grey") +

coord\_flip() + theme(legend.position = "top") +

xlab("All Brands") + ylab("Count") +

facet\_wrap(~term[topic])

plot

}

############################################################################

getTransaction <- function(dataSample, brand\_name){

dataSample <- getData(dataSample, brand\_name)

dataSampleCamera <- dataSample[order(dataSample$Camera),]

aprioriDataSample <- unique(dataSampleCamera[,c("CustID","Camera")])

set.seed(200)

sample <- sample.split(aprioriDataSample$Camera, SplitRatio = 0.8)

aprioriDataSample <- subset(aprioriDataSample, sample == TRUE)

aprioriDataSample$Camera <- paste(aprioriDataSample$Camera, "mp")

prepareAprioriData <- ddply(aprioriDataSample,c("CustID"), function(subset) paste(subset$Camera,

collapse = ","))

prepareAprioriData$CustID <- NULL

write.csv(prepareAprioriData,"cameraList.csv", row.names = TRUE)

transaction = read.transactions(file="cameraList.csv", rm.duplicates= TRUE, format="basket",sep=",",cols=1)

transaction@itemInfo$labels <- gsub("\"","",transaction@itemInfo$labels)

transaction

}

getAssociationPlot <- function(transaction){

cameraRules <- apriori(transaction,

parameter = list(sup = 0.01, conf = 0.01, target="rules"));

plot(cameraRules, method = "scatter", engine = "htmlwidget")

}

getItemFreqPlot <- function(transaction){

itemFrequencyPlot(transaction, topN = 5,type="absolute",

col = brewer.pal(5, "Dark2"),

main="Frequency according to mobile's camera")

}

############################################################################

getClassification <- function(dataSample, cameraMP, priceMobile){

cameraMP <- as.double(cameraMP)

priceMobile <- as.double(priceMobile)

trainNaive <- naive\_bayes(Age ~ Camera + Price, dataSample)

testNaive <- data.frame(Camera = c(cameraMP), Price = c(priceMobile))

predict(trainNaive, testNaive)

}

############################################################################

getSalesForecast <- function(dataSample, forecastYear, brand\_name){

dataSample <- getData(dataSample, brand\_name)

revenueYear <- aggregate(list(Revenue = dataSample$Price), by=list(Year = dataSample$Year), FUN=sum)

revenueYear <- revenueYear[order(revenueYear$Year),]

tsRevenue <- ts(revenueYear$Revenue, start = revenueYear$Year[1])

arimaModel <- auto.arima(tsRevenue)

forecastRevenue <- forecast(arimaModel, h = forecastYear)

autoplot(forecastRevenue, xlab = "Year", ylab = "Revenue")

}

############################################################################

getCluster <- function(dataSample, noOfClusters, isFrame, pointSize){

set.seed(400)

sample <- sample.split(dataSample$Age, SplitRatio = 0.20)

dataSampleAge <- subset(dataSample, sample == TRUE)

set.seed(80)

custCluster <- kmeans(dataSampleAge[, c("Price","Camera")], noOfClusters, nstart = 80)

custCluster$cluster <- as.factor(custCluster$cluster)

plot <-ggplot(dataSampleAge, aes(Price, Camera, col = custCluster$cluster)) +

geom\_point(aes(shape = dataSampleAge$Age), size = pointSize) +

guides(shape = guide\_legend("Age"), fill = guide\_legend("Clusters"), col = FALSE) + ylim(0, 50)

if(isFrame){

plot <- plot + geom\_convexhull(alpha = 0.1, aes(fill = custCluster$cluster))

}

plot

}

############################################################################

getFeatureData <- function(topBrands, PositiveData, NegativeData){

categories = c("screen","batteri", "app", "camera", "sim",

"call", "button" , "gb", "play", "qualiti", "price", "music", "featur",

"video","storag", "ram")

featureData <- data.frame(brandName = character(), feature = character(), freq = numeric(),

polarity = character(), stringsAsFactors = FALSE)

for(brand in topBrands){

eachBrandDataPos <- PositiveData[PositiveData$Brand.Name == brand,]

eachBrandDataNeg <- NegativeData[NegativeData$Brand.Name == brand,]

posTDM <- getTermDocMatrix(eachBrandDataPos)

posDFM <- getDataFrameMatrix(posTDM)

negTDM <- getTermDocMatrix(eachBrandDataNeg)

negDFM <- getDataFrameMatrix(negTDM)

for (category in categories) {

if(category %in% posDFM$words){

posCategoryFreq <- posDFM[posDFM$words == category,]$freq

} else {

posCategoryFreq <- 0

}

if(category %in% negDFM$words){

negCategoryFreq <- negDFM[negDFM$words == category,]$freq

} else {

negCategoryFreq <- 0

}

featureData[nrow(featureData) + 1,] <- list(brand, category, posCategoryFreq, 'Positive')

featureData[nrow(featureData) + 1,] <- list(brand, category, negCategoryFreq, 'Negative')

}

}

featureData

}

############################################################################

featureByPolarity <- function(featureData, polarityValue, reverse){

plot <- ggplot(data = featureData[featureData$polarity == polarityValue,], aes(x = brandName, y = freq, fill = brandName, color = brandName)) +

geom\_bar(stat = "identity", size = 1, col = "grey") +

xlab("All Brands") + ylab("Frequency") +

ggtitle(paste(polarityValue,"Features in each Brands")) +

facet\_wrap(~feature)

if(reverse){

plot <- plot + coord\_flip() + theme(legend.position = "top")

}

plot

}

featureByBrand <- function(featureData, brand\_Name, reverse){

plot <- ggplot(data = featureData[featureData$brandName == brand\_Name,], aes(x = feature, y = freq, fill = feature, color = feature)) +

geom\_bar(stat = "identity", size = 1, col = "grey") +

xlab("Features") + ylab("Frequency") +

ggtitle(paste("Features in", brand\_Name)) +

facet\_wrap(~polarity)

if(reverse){

plot <- plot + coord\_flip() + theme(legend.position = "top")

}

plot}

**CleaningandTrainingData.R**

library(stringr)

library(syuzhet)

library(Hmisc)

library(caTools)

setwd("E:/MScCS Final Project 2/MScCS Final Project/")

############################################################################

data <- read.csv("Amazon\_Mobile.csv")

data <- na.omit(data)

data <- data[!(data$Brand.Name==''),]

data <- unique(data)

data$Reviews <- as.character(data$Reviews)

data$Brand.Name <- tolower(data$Brand.Name)

############################################################################

data$Camera <- 0

decimalPoint <- "\\d+(\\.\\d{1,2})?"

cameraDetailRegex <- regex(paste(decimalPoint,"[ \t]\*mp", sep = ""),ignore\_case = TRUE)

for(i in 1:nrow(data)){

megaPixel <- str\_extract\_all(data$Product.Name[i],cameraDetailRegex)[[1]]

if(length(megaPixel) == 0){

data$Camera[i] <- 0

}else{

megaPixelValue <- max(str\_extract(megaPixel, decimalPoint))

data$Camera[i] <- megaPixelValue

}

}

data$Camera <- as.double(data$Camera)

data <- data[!(data$Camera == 0),]

data[str\_detect(data$Brand.Name ,"lg"),]$Brand.Name = "lg"

data[str\_detect(data$Brand.Name ,"sony"),]$Brand.Name = "sony"

data[str\_detect(data$Brand.Name ,"htc"),]$Brand.Name = "htc"

############################################################################

brandReviewCount <- aggregate(data$Review.Votes, by=list(Brand.Name=data$Brand.Name), FUN=sum)

sortBrandReviewCount <- brandReviewCount[order(brandReviewCount$x, decreasing = TRUE),]

topTenBrand <- sortBrandReviewCount[1:10,]$Brand.Name

topTenBrand

rm(brandReviewCount)

rm(sortBrandReviewCount)

data <- data[data$Brand.Name %in% topTenBrand,]

write.csv(data,"MobileReviewWithCamera.csv", row.names=FALSE)

############################################################################

funcCleanReviews <- function (reviews) {

clean\_reviews <- gsub('[[:punct:]]', ' ', reviews)

clean\_reviews <- gsub('[[:digit:]]', ' ', clean\_reviews)

clean\_reviews <- gsub('http\\w+', ' ', clean\_reviews)

clean\_reviews <- gsub('[ \t]{2,}', ' ', clean\_reviews)

clean\_reviews <- gsub('^\\s+|\\s+$', ' ', clean\_reviews)

clean\_reviews <- gsub('<.\*>', '', enc2native(clean\_reviews))

clean\_reviews <- gsub('[^\x01-\x7F]', '', clean\_reviews)

clean\_reviews <- tolower(clean\_reviews)

clean\_reviews

}

############################################################################

set.seed(500)

sample <- sample.split(data$Brand.Name, SplitRatio = (500 / nrow(data)))

dataSample <- subset(data, sample == TRUE)

dataSample$Reviews <- funcCleanReviews(dataSample$Reviews)

dataSample <- na.omit(dataSample)

dataSample <- dataSample[!(dataSample$Brand.Name == ''),]

dataSample <- unique(dataSample)

age <- c("Below 16", "17 - 30", "30 - 40", "40 - 60", "More Than 60")

ageData <- list()

i <- 1

for (megaPixel in dataSample$Camera) {

if(megaPixel >= 20){

indexProb <- sample(5, 1, prob = c(0.05, 0.55, 0.2, 0.1, 0.1))

}else if(megaPixel >= 13){

indexProb <- sample(5, 1, prob = c(0.05, 0.45, 0.3, 0.1, 0.1))

}else if(megaPixel >= 8){

indexProb <- sample(5, 1, prob = c(0.1, 0.3, 0.2, 0.3, 0.1))

}else if(megaPixel >= 5){

indexProb <- sample(5, 1, prob = c(0.15, 0.25, 0.3, 0.2, 0.1))

}else{

indexProb <- sample(5, 1, prob = c(0.2, 0.1, 0.2, 0.2, 0.3))

}

ageData[i] <- age[indexProb]

i <- i + 1

}

dataSample$Age <- as.character(ageData)

############################################################################

dataSample <- read.csv("SampleMobileReview.csv")

dataSample <- dataSample[order(dataSample$Camera),]

dataSample$CustID <- 0

cameraTable <- aggregate(Reviews ~ Camera, dataSample, FUN = length)

for(cam in c(23,41)){

sizeCameraList = nrow(dataSample[dataSample$Camera == cam,])

custIDList = seq(1, sizeCameraList + 10)

dataSample[dataSample$Camera == cam,]$CustID <- sample(custIDList, sizeCameraList)

}

startCustID <- max(dataSample$CustID) + 1

for(cam in c(12,16)){

sizeCameraList = nrow(dataSample[dataSample$Camera == cam,])

custIDList = seq(startCustID, startCustID + sizeCameraList + 10)

dataSample[dataSample$Camera == cam,]$CustID <- sample(custIDList, sizeCameraList)

}

startCustID <- max(dataSample$CustID) + 1

for(cam in c(8,13)){

sizeCameraList = nrow(dataSample[dataSample$Camera == cam,])

custIDList = seq(startCustID, startCustID + sizeCameraList + 10)

dataSample[dataSample$Camera == cam,]$CustID <- sample(custIDList, sizeCameraList)

}

startCustID <- max(dataSample$CustID) + 1

for(cam in c(3,3.20)){

sizeCameraList = nrow(dataSample[dataSample$Camera == cam,])

custIDList = seq(1, sizeCameraList + 10)

dataSample[dataSample$Camera == cam,]$CustID <- sample(custIDList, sizeCameraList)

}

startCustID <- max(dataSample$CustID) + 1

for(cam in c(2,5)){

sizeCameraList = nrow(dataSample[dataSample$Camera == cam,])

custIDList = seq(startCustID, startCustID + sizeCameraList + 10)

dataSample[dataSample$Camera == cam,]$CustID <- sample(custIDList, sizeCameraList)

}

startCustID <- max(dataSample$CustID) + 1

sizeCameraList <- nrow(dataSample[dataSample$CustID == 0,])

custIDList = seq(startCustID, startCustID + sizeCameraList)

dataSample[dataSample$CustID == 0,]$CustID <- sample(custIDList, sizeCameraList, replace = TRUE)

x <- aggregate(CustID ~ Camera, dataSample, FUN = length)

############################################################################

dataSample$Year <- sample(seq(2008,2017), nrow(dataSample),

prob = c(0.05, 0.08, 0.07, 0.09, 0.09, 0.1, 0.12, 0.13, 0.12 ,0.15), replace = TRUE)

############################################################################

emotions <- get\_nrc\_sentiment(as.character(dataSample$Reviews))

emotionTable <- emotions[,1:8]

polarityTable <- emotions[,9:10]

dataSample$Polarity <- capitalize(colnames(polarityTable)[apply(polarityTable,1,which.max)])

dataSample$Emotions <- capitalize(colnames(emotionTable)[apply(emotionTable,1,which.max)])

write.csv(dataSample,"SampleMobileReview.csv", row.names=FALSE)

############################################################################

set.seed(500)

sample <- sample.split(dataSample$Brand.Name, SplitRatio = 0.75)

trainData <- subset(dataSample, sample == TRUE)

write.csv(trainData,"SampleMobileReviewTrain.csv", row.names=FALSE)

testData <- subset(dataSample, sample == FALSE)

write.csv(testData,"SampleMobileReviewTest.csv", row.names=FALSE)

rm(emotions)

rm(emotionTable)

rm(polarityTable)

############################################################################

rm(list=ls())

gc()

**MobileAnalysis.R**

library(shiny)

library(shinydashboard)

library(shinyWidgets)

library(shinyjs)

library(plotly)

library(plyr)

library(dplyr)

library(shinycssloaders)

source("E:/MScCS Final Project 2/MScCS Final Project/MobileMethods.R")

setwd("E:/MScCS Final Project 2/MScCS Final Project/")

options(shiny.maxRequestSize = 30\*1024^2)

############################################################################

# allBrandData <- read.csv("SampleMobileReview.csv", stringsAsFactors = FALSE)

# topBrands <- unique(allBrandData$Brand.Name)

# allBrandName <- c("All Brands", topBrands)

filterCategory <- c("All", "Positive Polarity", "Negative Polarity", "Ratings > 3", "Ratings < 3")

filterPos <- c("Positive Polarity", "Ratings > 3")

filterNeg <- c("Negative Polarity", "Ratings < 3")

allBrandData <- NULL

topBrands <- NULL

allBrandName <- NULL

############################################################################

ui <- dashboardPage(

####################

dashboardHeader(title = "Mobile Brand Analysis"),

####################

dashboardSidebar(sidebarMenu(

useShinyjs(),

style = "position: fixed; overflow: visible;",

menuItem("Review Count", tabName = "review", icon = icon("chart-bar")),

menuItem("Polarity Analysis", tabName = "polarityAnalysis", icon = icon("poll-h")),

menuItem("Emotion Analysis", tabName = "emotionAnalysis", icon = icon("grin")),

menuItem("Distribution Of Ratings", tabName = "distributionOfRatings", icon = icon("star")),

menuItem("Ratings Vs Polarity", tabName = "ratingsByPolarity", icon = icon("balance-scale")),

menuItem("Ratings Vs Emotions", tabName = "ratingsByEmotion", icon = icon("balance-scale")),

menuItem("WordFrequency & WordCloud", tabName = "wordFrequency", icon = icon("cloudsmith")),

menuItem("Associativity", tabName = "associativity", icon = icon("link")),

menuItem("Clustering", tabName = "clustering", icon = icon("layer-group")),

menuItem("Features Analysis", tabName = "features", icon = icon("elementor")),

menuItem("Topic Model", tabName = "topicModel", icon = icon("chart-bar")),

menuItem("Sales Forecast", tabName = "salesForecast", icon = icon("chart-area")),

menuItem("Classification", tabName = "classify", icon = icon("hand-point-right"))

)),

####################

dashboardBody(

tags$head(

includeCSS("E:/MScCS Final Project 2/MScCS Final Project/style.css")

),

tabItems(

tabItem(tabName = "review",

fluidRow(

box(title = p("Mobile Dataset : "), width = 9,

fileInput("mobileDataset", "Choose CSV File",

accept = c(

"text/csv",

"text/comma-separated-values",

".csv")

)),

hidden(

div(id = "datasetID",

infoBox(title = p("No. Of Reviews for all Brands")),

box(

width = 6,

plotlyOutput("reviewCount", height = 500, width = 500) %>% withSpinner(color="#248f24")

)

))

)

),

tabItem(tabName = "polarityAnalysis",

fluidRow(

box(title = p("Polarity Count of Review"),

selectInput("brandSelectPolarity",label = h4("Select Brand : "),choices = allBrandName),

materialSwitch("polarityReverse",label = h4("Flip Co-ordinates ?"),value = FALSE, status = "primary"),

actionButton("donePolarityCount", "Done")

),

hidden(

div(id = "polarityAnalysisID",

box(

width = 9,

plotlyOutput("polarityCount", height = 500, width = 600) %>% withSpinner(color="#248f24")

)))

)

),

tabItem(tabName = "emotionAnalysis",

fluidRow(

box(title = p("Emotion Count of Reviews"),

selectInput("brandSelectEmotion",label = h4("Select Brand : "),choices = allBrandName),

materialSwitch("emotionReverse",label = h4("Flip Co-ordinates ?"),value = FALSE, status = "primary"),

actionButton("doneEmotionCount", "Done")

),

hidden(

div(id = "emotionAnalysisID",

box(

width = 9,

plotlyOutput("emotionCount", height = 500, width = 600) %>% withSpinner(color="#248f24")

)))

)

),

tabItem(tabName = "distributionOfRatings",

fluidRow(

box(title = p("Distribution of the Ratings"),

selectInput("brandSelectRatings",label = h4("Select Brand : "),choices = allBrandName),

materialSwitch("distOfRatingsReverse",label = h4("Flip Co-ordinates ?"),value = FALSE, status = "primary"),

actionButton("doneDistOfRatings", "Done")

),

hidden(

div(id = "distributionOfRatingsID",

box(

width = 9,

plotlyOutput("distOfRatings", height = 500, width = 600) %>% withSpinner(color="#248f24")

)))

)

),

tabItem(tabName = "ratingsByPolarity",

fluidRow(

box(title = p("Distribution of the Ratings by Polarity"),

selectInput("brandSelectPolarityRatings",label = h4("Select Brand : "),choices = allBrandName),

materialSwitch("distOfPolarityRatingsReverse",label = h4("Flip Co-ordinates ?"),value = FALSE, status = "primary"),

actionButton("doneDistOfPolarityRatings", "Done")

),

hidden(

div(id = "ratingsByPolarityID",

box(

width = 9,

plotlyOutput("distOfPolarityRatings", height = 500, width = 500) %>% withSpinner(color="#248f24")

),

box(

width = 9,

plotOutput("confusionMatrixGraph", height = 500, width = 500) %>% withSpinner(color="#248f24")

))

))

),

tabItem(tabName = "ratingsByEmotion",

fluidRow(

box(title = p("Distribution of the Ratings by Emotions"),

selectInput("brandSelectEmotionRatings",label = h4("Select Brand : "),choices = allBrandName),

materialSwitch("distOfEmotionRatingsReverse",label = h4("Flip Co-ordinates ?"),value = FALSE, status = "primary"),

actionButton("doneDistOfEmotionRatings", "Done")

),

hidden(

div(id = "ratingsByEmotionID",

box(

width = 9,

plotlyOutput("distOfEmotionRatings", height = 500, width = 500) %>% withSpinner(color="#248f24")

)))

)

),

tabItem(tabName = "wordFrequency",

fluidRow(

box(title = p("Word Frequency and WordCloud"),

selectInput("brandSelectWordFrequency",label = h4("Select Brand : "),choices = allBrandName),

selectInput("filterSelectWordFrequency",label = h4("Filter : "), choices = filterCategory),

actionButton("doneWordFrequency", "Done")

)),

fluidRow(

hidden(

div(id = "wordFrequencyID",

box(

width = 6,

plotlyOutput("wordFrequencyGraph", height = 500, width = 500) %>% withSpinner(color="#248f24")

),

box(

width = 6,

plotOutput("wordCloudGraph", height = 500, width = 500) %>% withSpinner(color="#248f24")

))

))

),

tabItem(tabName = "associativity",

fluidRow(

box(title = p("Associative words of Most Frequently used Word"),

selectInput("brandSelectAssociativity",label = h4("Select Brand : "),choices = allBrandName),

actionButton("doneAssociativity", "Done")

),

hidden(

div(id = "associativityID",

box(

width = 9,

plotlyOutput("associativityGraph", height = 500, width = 500) %>% withSpinner(color="#248f24")

),

box(

width = 9,

plotOutput("itemFreqGraph", height = 800, width = 800) %>% withSpinner(color="#248f24")

)

))

)

),

tabItem(tabName = "clustering",

fluidRow(

box(title = p("Clustering"), width = 3,

sliderInput("pointSize",label = h4("Point Size : "), min = 0.1, max = 3.0, value = 2.5),

materialSwitch("frameConvexHull",label = h4("Frame (Convex Hull) ?"),value = FALSE, status = "primary"),

sliderInput("noOfClusters",label = h4("No. Of Clusters : "), min = 0, max = 10, value = 4)

),

hidden(

div(id = "clusteringID",

box(

width = 3,

plotOutput("clusterGraph", height = 600, width = 800) %>% withSpinner(color="#248f24")

))

))

),

tabItem(tabName = "features",

fluidRow(

box(title = p("Features"),

selectInput("brandSelectFeatures",label = h4("Select Brand : "),choices = allBrandName),

selectInput("posFilter",label = h4("Positive Filter : "),choices = filterPos),

selectInput("negFilter",label = h4("Negative Filter : "),choices = filterNeg),

materialSwitch("featureReverse",label = h4("Flip Co-ordinates ?"),value = TRUE, status = "primary"),

actionButton("doneFeature", "Done")

)),

fluidRow(

hidden(

div(id = "featureID1",

box(

width = 10,

plotlyOutput("featureGraph1", height = 700, width = 700) %>% withSpinner(color="#248f24")

))),

hidden(

div(

id = "featureID2",

box(

width = 10,

plotlyOutput("featureGraph2", height = 700, width = 700) %>% withSpinner(color="#248f24")

))

))

),

tabItem(tabName = "topicModel",

fluidRow(

box(title = p("Topics Discussed"),

sliderInput("noOfClustersTopics",label = h4("No. Of Clusters : "), min = 0, max = 10, value = 8),

sliderInput("noOfTopics",label = h4("No. Of Topics : "), min = 0, max = 10, value = 4),

actionButton("doneTopics", "Done")

),

hidden(

div(id = "topicsID",

box(

width = 12,

plotlyOutput("topicsGraph", height = 800, width = 800) %>% withSpinner(color="#248f24")

))

))

),

tabItem(tabName = "salesForecast",

fluidRow(

box(title = p("Sales Forecast"),

selectInput("brandSelectForecast",label = h4("Select Brand : "),choices = allBrandName),

sliderInput("forecastYear",label = h4("No Of Years To Forecast : "), min = 1, max = 10, value = 3),

actionButton("doneForecast", "Done")

),

hidden(

div(id = "forecastID",

box(

width = 12,

plotlyOutput("forecastGraph", height = 800, width = 800) %>% withSpinner(color="#248f24")

))

))

),

tabItem(tabName = "classify",

fluidRow(

box(title = p("Age Group to Target"),

textInput("cameraMP",label = h4("Mobile Camera (in MegaPixels) : "), placeholder = "16.5"),

textInput("priceMobile",label = h4("Estimated Price : "), placeholder = "400.5"),

actionButton("doneClassify", "Done")

),

hidden(

div(id = "classifyID",

box(

width = 7,

verbatimTextOutput("ageGroup")

))

))

)

))

####################

)

###########################################################################################################################

server <- function(input, output, session) {

observeEvent(input$mobileDataset,

{

mobileData <- input$mobileDataset

allBrandData <<- read.csv(mobileData$datapath, stringsAsFactors = FALSE)

topBrands <<- unique(allBrandData$Brand.Name)

allBrandName <<- c("All Brands", topBrands)

updateSelectInput(session, "brandSelectPolarity", choices = allBrandName)

updateSelectInput(session, "brandSelectEmotion", choices = allBrandName)

updateSelectInput(session, "brandSelectRatings", choices = allBrandName)

updateSelectInput(session, "brandSelectPolarityRatings", choices = allBrandName)

updateSelectInput(session, "brandSelectEmotionRatings", choices = allBrandName)

updateSelectInput(session, "brandSelectWordFrequency", choices = allBrandName)

updateSelectInput(session, "brandSelectAssociativity", choices = allBrandName)

updateSelectInput(session, "brandSelectFeatures", choices = allBrandName)

updateSelectInput(session, "brandSelectForecast", choices = allBrandName)

shinyjs::show("datasetID")

output$reviewCount <- renderPlotly({noOfReview(allBrandData)})

})

observeEvent(input$donePolarityCount,

{

shinyjs::show("polarityAnalysisID")

reverse <- input$polarityReverse

brand\_name <- input$brandSelectPolarity

output$polarityCount <- renderPlotly({polarityOfReview(allBrandData, reverse, brand\_name)})

})

observeEvent(input$doneEmotionCount,

{

shinyjs::show("emotionAnalysisID")

reverse <- input$emotionReverse

brand\_name <- input$brandSelectEmotion

output$emotionCount <- renderPlotly({emotionOfReview(allBrandData, reverse, brand\_name)})

})

observeEvent(input$doneDistOfRatings,

{

shinyjs::show("distributionOfRatingsID")

reverse <- input$distOfRatingsReverse

brand\_name <- input$brandSelectRatings

output$distOfRatings <- renderPlotly({distributionOfRatings(allBrandData, reverse, brand\_name)})

})

observeEvent(input$doneDistOfPolarityRatings,

{

shinyjs::show("ratingsByPolarityID")

reverse <- input$distOfPolarityRatingsReverse

brand\_name <- input$brandSelectPolarityRatings

output$distOfPolarityRatings <- renderPlotly({distributionOfRatingsbyPolarity(allBrandData, reverse, brand\_name)})

output$confusionMatrixGraph <- renderPlot({confusionMatrixPlot(allBrandData, brand\_name)})

})

observeEvent(input$doneDistOfEmotionRatings,

{

shinyjs::show("ratingsByEmotionID")

reverse <- input$distOfEmotionRatingsReverse

brand\_name <- input$brandSelectEmotionRatings

output$distOfEmotionRatings <- renderPlotly({distributionOfRatingsbyEmotion(allBrandData, reverse, brand\_name)})

})

observeEvent(input$doneWordFrequency,

{

shinyjs::show("wordFrequencyID")

brand\_name <- input$brandSelectWordFrequency

filterCategory <- input$filterSelectWordFrequency

dataSet <- getFilteredData(allBrandData,filterCategory,brand\_name)

termMatrix <- getTermDocMatrix(dataSet)

dataFrameMatrix <- getDataFrameMatrix(termMatrix)

output$wordFrequencyGraph <- renderPlotly({wordFreqGraph(dataFrameMatrix)})

output$wordCloudGraph <- renderPlot({wordCloudGraph(dataFrameMatrix)})

})

observeEvent(input$doneAssociativity,

{

shinyjs::show("associativityID")

brand\_name <- input$brandSelectAssociativity

transaction <- getTransaction(allBrandData, brand\_name)

output$associativityGraph <- renderPlotly({getAssociationPlot(transaction)})

output$itemFreqGraph <- renderPlot({getItemFreqPlot(transaction)})

})

observeEvent({

input$pointSize

input$frameConvexHull

input$noOfClusters

},

{

shinyjs::show("clusteringID")

pointSize <- input$pointSize

isFrame <- input$frameConvexHull

noOfClusters <- input$noOfClusters

output$clusterGraph <- renderPlot({getCluster(allBrandData, noOfClusters, isFrame, pointSize)})

})

observeEvent(input$doneFeature,

{

brand\_name <- input$brandSelectFeatures

if(brand\_name == "All Brands"){

shinyjs::show("featureID1")

shinyjs::show("featureID2")

}else{

shinyjs::show("featureID1")

shinyjs::hide("featureID2")

}

posFilter <- input$posFilter

posFilterData <- getFilteredData(allBrandData,posFilter,brand\_name)

negFilter <- input$negFilter

negFilterData <- getFilteredData(allBrandData,negFilter,brand\_name)

featureReverse <- input$featureReverse

featureData <- getFeatureData(topBrands, posFilterData, negFilterData)

if(brand\_name == "All Brands"){

output$featureGraph1 <- renderPlotly({featureByPolarity(featureData,'Positive', featureReverse)})

output$featureGraph2 <- renderPlotly({featureByPolarity(featureData,'Negative', featureReverse)})

}else{

output$featureGraph1 <- renderPlotly({featureByBrand(featureData, brand\_name, featureReverse)})

}

})

observeEvent(input$doneTopics,

{

shinyjs::show("topicsID")

noOfClusters <- input$noOfClustersTopics

noOfTopics <- input$noOfTopics

termMatrix <- getTermDocMatrix(allBrandData)

output$topicsGraph <- renderPlotly({topicModelGraph(allBrandData, termMatrix, noOfClusters, noOfTopics)})

})

observeEvent(input$doneForecast,

{

shinyjs::show("forecastID")

brand\_name <- input$brandSelectForecast

forecastYear <- input$forecastYear

output$forecastGraph <- renderPlotly({getSalesForecast(allBrandData,forecastYear,brand\_name)})

})

observeEvent(input$doneClassify,

{

shinyjs::show("classifyID")

cameraMP <- input$cameraMP

priceMobile <- input$priceMobile

ageGroupText <- paste("Age Group To Target :", getClassification(allBrandData, cameraMP, priceMobile),"years")

output$ageGroup <- renderText({ageGroupText})

})

}

shinyApp(ui, server)

**CONCLUSIONS**

1) The Polarity count has an Accuracy of 61.98%.

2) There was no association found for customers to proof that they are moving towards better Camera Phones.

3) Higher range of MegaPixels in Mobile Phones Camera attracts 17 to 30 years age group customers.

4) Sales of all Mobile Brands collectively in next few years are going to be constant but particular Brands such as Blackberry may face very low Sales in coming 3 years.

**FUTURE SCOPE AND DEVELOPMENT**

1) Analysis of whether customer’s moving towards buying higher range of RAM in Mobile Phone.

2) Maximize the Sales based on the Location.

3) Finding out the Relationship between Mobile Brand Features and Gender.

4) Predicting optimal price for new product according to its feature.

5) Including the Analysis of Associated Accessories such as EarPhones, Chargers,etc for that Mobile Brand.

**REFERENCES AND APPENDIX**

**References**

* https://www.kaggle.com (Dataset)
* https://www.r-bloggers.com
* <https://www.analyticsvidhya.com>
* <https://rpubs.com>
* <http://shiny.rstudio.com>
* https://towardsdatascience.com
* https://www.datascience.com
* <https://sites.google.com/site/dataclusteringalgorithms/k-means-clustering-algorithm>
* Zatar, T. (2014, October). *IJSER*. Retrieved August 30, 2018, from International Journal of Scientific & Engineering Research, Volume 5, Issue 10: <https://www.ijser.org/researchpaper/Data-mining-in-marketing.pdf>

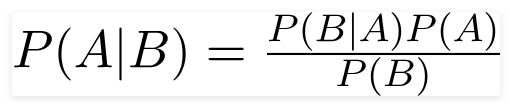
**Appendix**

* **Naïve Bayes Classification :-**

**Principle of Naive Bayes Classifier:**

A Naive Bayes classifier is a probabilistic machine learning model that’s used for classification task. The crux of the classifier is based on the Bayes theorem.

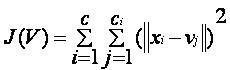
**Bayes Theorem:**



Using Bayes theorem, we can find the probability of **A** happening, given that **B** has occurred. Here, **B** is the evidence and **A** is the hypothesis. The assumption made here is that the predictors/features are independent. That is presence of one particular feature does not affect the other. Hence it is called naive.

* **KMeans Clustering :-**

k-means is  one of  the simplest unsupervised  learning  algorithms  that  solve  the well  known clustering problem. The procedure follows a simple and  easy  way  to classify a given data set  through a certain number of  clusters (assume k clusters) fixed apriori. The  main  idea  is to define k centers, one for each cluster. These centers  should  be placed in a cunning  way  because of  different  location  causes different  result. So, the better  choice  is  to place them  as  much as possible  far away from each other. The  next  step is to take each point belonging  to a  given data set and associate it to the nearest center. When no point  is  pending,  the first step is completed and an early group age  is done. At this point we need to re-calculate k new centroids as barycenter of  the clusters resulting from the previous step. After we have these k new centroids, a new binding has to be done  between  the same data set points  and  the nearest new center. A loop has been generated. As a result of  this loop we  may  notice that the k centers change their location step by step until no more changes  are done or  in  other words centers do not move any more. Finally, this  algorithm  aims at  minimizing  an objective function know as squared error function given by:

[](https://sites.google.com/site/dataclusteringalgorithms/k-means-clustering-algorithm/kmeans.JPG?attredirects=0)

where,  
                           *‘||xi- vj||’* is the Euclidean distance between *xi* and *vj.*

*‘ci’* is the number of data points in *ith* cluster.

*‘c’* is the number of cluster centers.

**Algorithmic steps for k-means clustering**

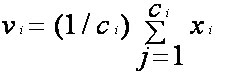
Let  X = {x1,x2,x3,……..,xn} be the set of data points and V = {v1,v2,…….,vc} be the set of centers.

1) Randomly select *‘c’* cluster centers.

2) Calculate the distance between each data point and cluster centers.

3) Assign the data point to the cluster center whose distance from the cluster center is minimum of all the cluster centers..

4) Recalculate the new cluster center using:

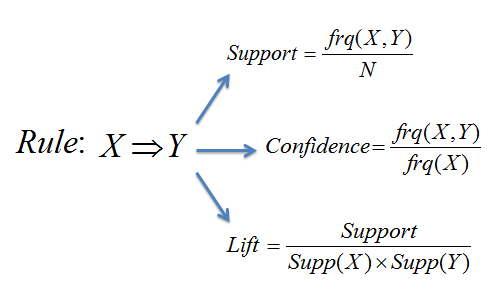


where,*‘ci’* represents the number of data points in *ith* cluster.

5) Recalculate the distance between each data point and new obtained cluster centers.

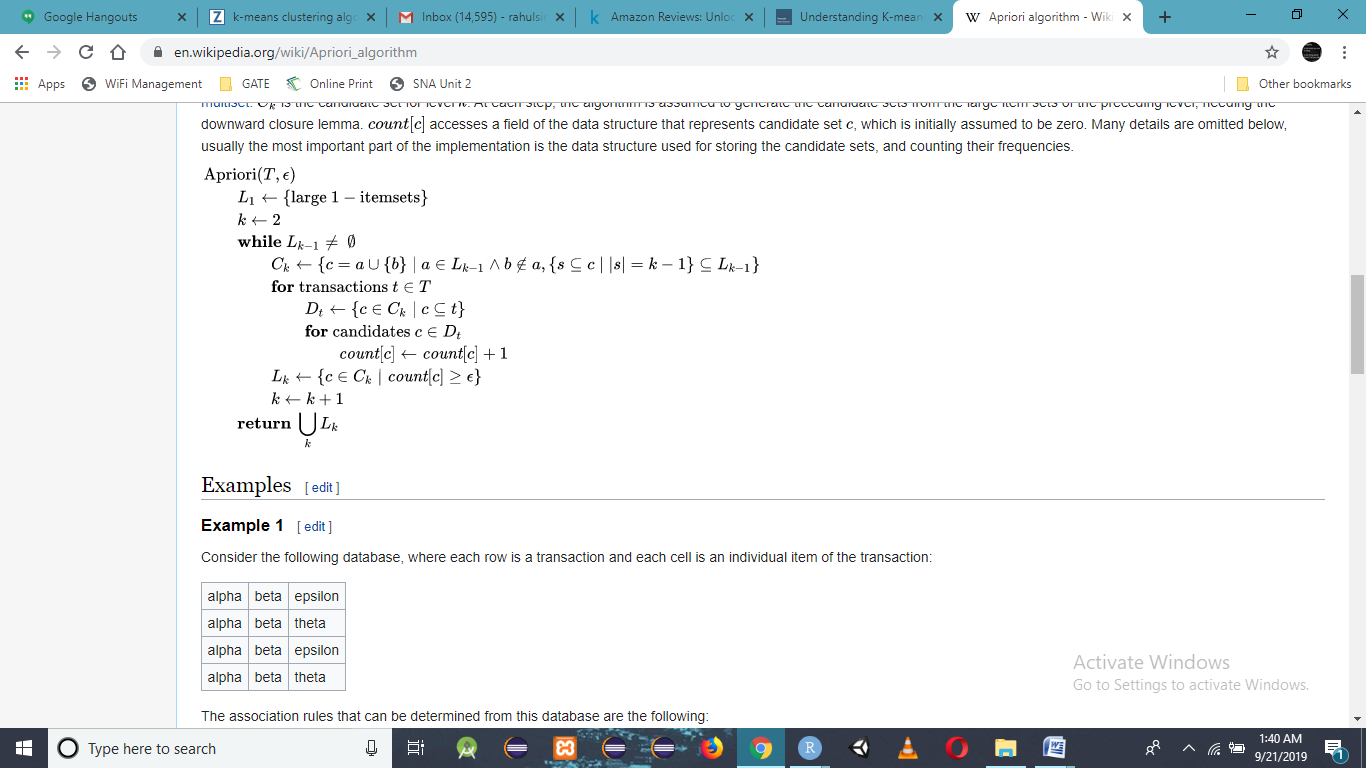
6) If no data point was reassigned then stop, otherwise repeat from step 3).

* **Apriori :-**



Apriori is an algorithm for frequent item set mining and association rule learning over transactional databases. It proceeds by identifying the frequent individual items in the database and extending them to larger and larger item sets as long as those item sets appear sufficiently often in the database. The frequent item sets determined by apriori can be used to determine association rules which highlight general trends in the database: this has applications in domains such as market basket.

The A-Priori Algorithm is designed to reduce the number of pairs that must be counted, at the expense of performing two passes over data, rather than one pass.



* **ARIMA Model :-**

An [ARIMA model](https://en.wikipedia.org/wiki/Autoregressive_integrated_moving_average) is a class of statistical models for analyzing and forecasting time series data.

It explicitly caters to a suite of standard structures in time series data, and as such provides a simple yet powerful method for making skillful time series forecasts.

ARIMA is an acronym that stands for AutoRegressive Integrated Moving Average. It is a generalization of the simpler AutoRegressive Moving Average and adds the notion of integration.

This acronym is descriptive, capturing the key aspects of the model itself. Briefly, they are:

**AR**: *Autoregression*. A model that uses the dependent relationship between an observation and some number of lagged observations.

**I**: *Integrated*. The use of differencing of raw observations (e.g. subtracting an observation from an observation at the previous time step) in order to make the time series stationary.

**MA**: *Moving Average*. A model that uses the dependency between an observation and a residual error from a moving average model applied to lagged observations.

* **Topic Model :-**

In [machine learning](https://en.wikipedia.org/wiki/Machine_learning) and [natural language processing](https://en.wikipedia.org/wiki/Natural_language_processing), a **topic model** is a type of [statistical model](https://en.wikipedia.org/wiki/Statistical_model) for discovering the abstract "topics" that occur in a collection of documents. Topic modeling is a frequently used text-mining tool for discovery of hidden semantic structures in a text body. Intuitively, given that a document is about a particular topic, one would expect particular words to appear in the document more or less frequently: "dog" and "bone" will appear more often in documents about dogs.

* **LDA :-**

Latent Dirichlet Allocation (LDA) is an example of topic model and is used to classify text in a document to a particular topic.

In more detail, LDA represents documents as mixtures of topics that spit out words with certain probabilities. It assumes that documents are produced in the following fashion: when writing each document, you

Decide on the number of words N the document will have (say, according to a Poisson distribution).

Choose a topic mixture for the document (according to a Dirichlet distribution over a fixed set of K topics).

**INDEX AND ACRONYMS**

ARIMA - AutoRegressive Integrated Moving Average

LDA - Latent Dirichlet Allocation

IDE - Integrated Development Environment

P(A) and P(B) – Probability of Event A and B respectively

P(A|B) - Probability of occurrence of event A given the event B is true

P(B|A) - Probability of the occurrence of event B given the event A is true

frq – Frequency

CSV – Comma Separated Values