Social Media Analysis

Data Clan

Daniel Chen, Ghazal Erfani, Shiven Shashidhar

95851: Making Products Count: Data Science for Product Managers

DSPM Project

Introduction

Today, almost all smartphones run on one of two operating systems; Google's Android or Apple's iOS. These two platforms accounted for more 99.7 percent of all new smartphones shipped in 2018. It's no surprise that these mobile operating systems dominate the market: they are both excellent. They have much in common with one another, as well as some key differences.

Given their market domination, it comes to no surprise that the battle of the two operating systems has raged for more than a decade now. Their quality, price, and value have been compared widely. This battle was most exemplified during the release of the iPhone 8/X and Samsung Galaxy S8 in 2017. These smartphones have been regarded by some as 'the most beautiful phones' ever made. They have also been consistently pitted against one another. Their designs, displays, special features, and values have been analyzed closely by many new sources and professional reviewers.

Ultimately, it is not the opinion of news sources or professional reviewers that determines the success of these smartphones. It's the opinion of the consumers. Thanks to the pervasiveness of social media, consumers' opinions about these phones are now widely available. One can analyze their social media posts, comments, and reviews from sources such as blogs, Facebook, Instagram, and Twitter to better understand the sentiments around these smartphones.

In this analysis, we are focused on answering two primary questions. Firstly, we would like to get a grasp of consumers' sentiments about the Galaxy S8 and iPhone 8/X prior and post their releases. Secondly, we would like to understand what attributes of the Galaxy S8 and iPhone 8/X are most important to consumers and how consumers feel about those attributes.

Answers to such questions can be of significant value to the creators of these devices: Apple and Samsung. They can use such sentiment analysis to gauge consumers' interests and questions, make better product and marketing decisions, and forecast sales. Additionally, such sentiment analysis can greatly benefit new consumers to the market. They can learn from the experiences and opinions of consumers similar to themselves and make more informed decisions about the devices they choose to purchase.

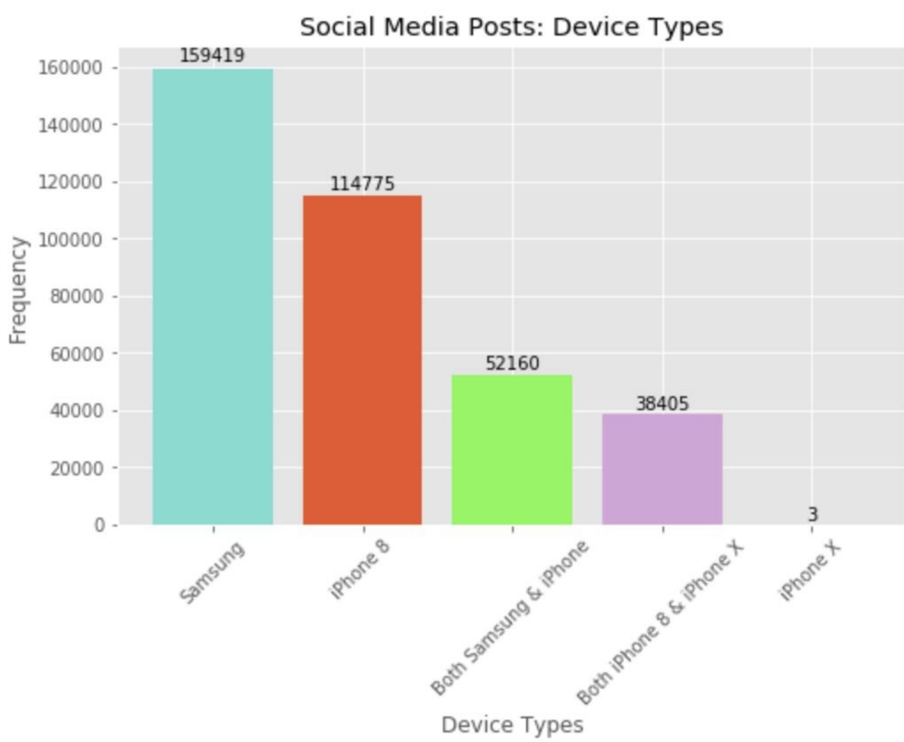
Data Preparation

To prepare this data for modeling, we performed a series of data cleansing and extraction steps. Firstly, we removed all non-original posts, such as replies or comments, because these posts can be hard to contextualize since they are detached from their original post. Secondly, we eliminated any posts from authors with more than 100 unique followers to ensure we are only capturing end consumers, not news sources or professional reviewers. We then grouped the posts into ones that contained the device names, such as 'Galaxy S8.' We created the following five groups of device types: Galaxy S8, Galaxy S8 & iPhone, iPhone 8, iPhone X, iPhone 8 and iPhone X.

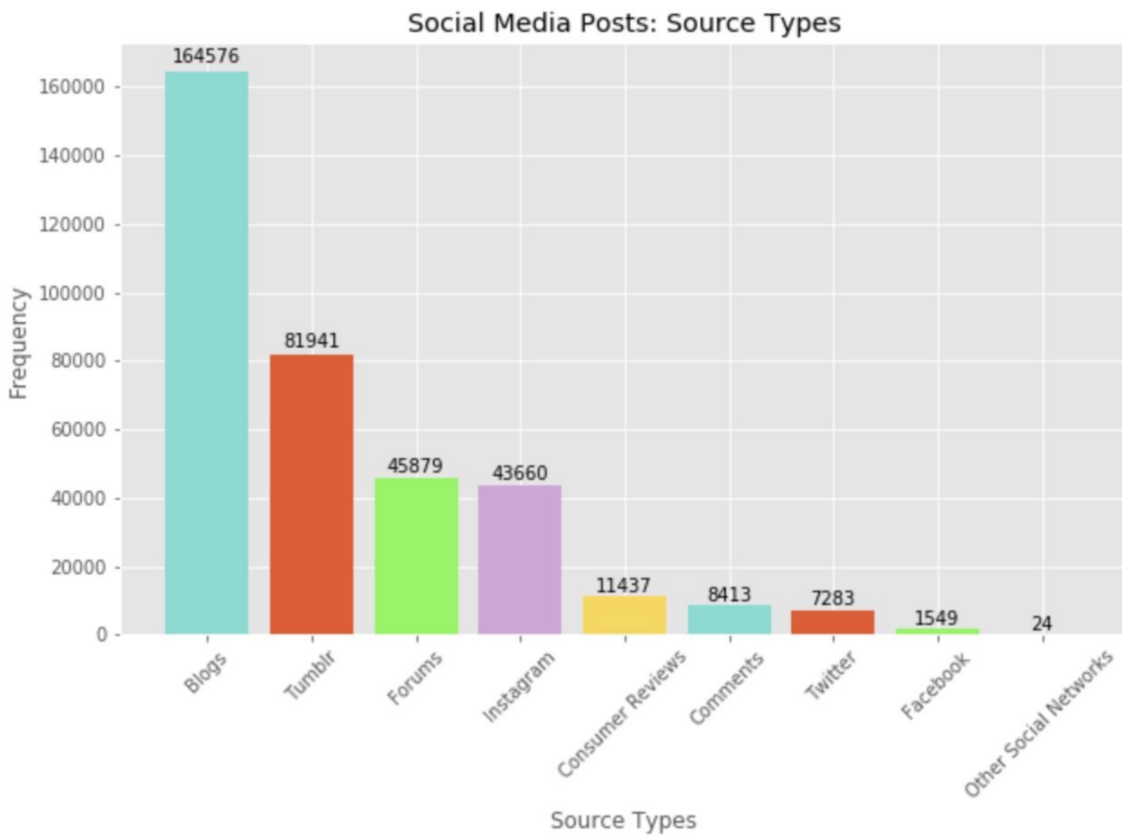
Next, we focused on preparing the posts for analysis. We removed punctuation and stopwords, converted to lowercase, tokenized and stemmed the posts, and did speech tagging. This allowed us to move forward with text analysis of the posts.

Exploratory Data Analysis

To better understand the data, we performed exploratory data analysis. Firstly, we looked at the distribution of posts across device types.



It seemed that Samsung had the greatest number of posts while iPhone X had very few. Moving forward, we chose to not focus on the analysis of iPhone X only. We also looked at the spread of posts across different sources.



We learned that a large majority of the posts were from blogs. We were surprised to find that very few posts originated from social media sites such as Twitter and Facebook. We considered that a larger number of blog posts likely implied better text analysis, given that such text is often more structured than that in social media posts.

In addition, we performed topic modeling using Latent Dirichlet Allocation model to further explore the data. This was based on the following hypothesis: each user's post contains latent 'topics' (themes) and each topic is a distribution over different words/tokens. In our model, we considered each text post to be a probability distribution over 10 different 'topics' and each topic to be a distribution over 3324 words (in case of Apple iPhone) and 5893 words (in case of Samsung Galaxy). Here each 'topic' of a user review can be thought of as the main focus of the review. For example, the user might be commenting about the design of iPhone or the price of Samsung Galaxy. Then, the topic for the first case would be 'design' and for the second case would be ‘price.’ Topic modeling enabled us to better understand the themes that are most pertinent to consumers.

Most significantly, we found that iPhone 8/X posts revealed four relevant topics: The first topic was primarily focused on the color, warranty and memory space. The second topic was focused on price/cost of the iPhone. The third topic mainly dealt with display, design, looks and screen quality. The fourth topic was based on wireless charging and power. Similarly, for Samsung Galaxy, we found four relevant topics: color, accessories, display and screen, and bixby and iris scanner. This provided us with some initial insights into consumers’ primary concerns.

Modeling Approach

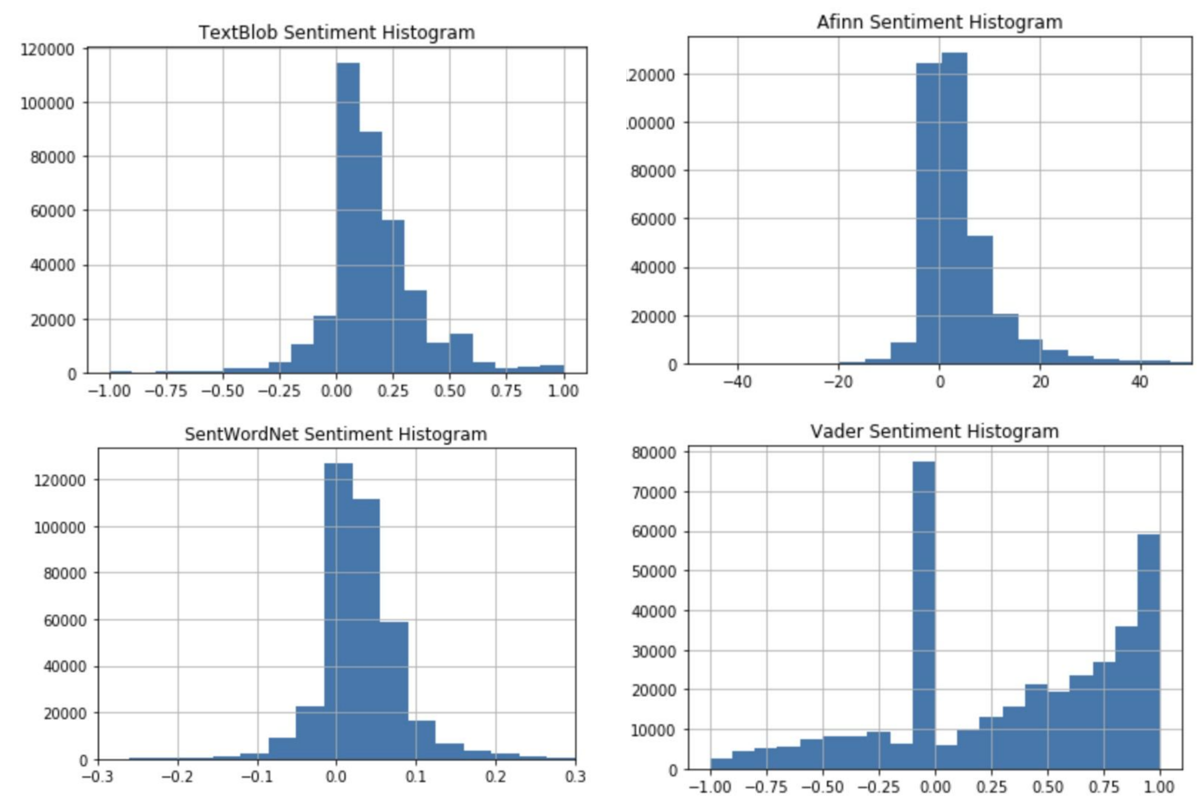
In aiming to better understand consumers’ sentiments about the Galaxy S8 and iPhone 8/X prior and post their releases, we applied several sentiment analysis models. These robust, pre-trained models are tuned to analyze the sentiments of textual content and return a sentiment rating value that indicates negative, neutral, or positive. Given the quintillion bytes of data that are generated every day, sentiment analysis tools such as these are used ubiquitously to make sense of unstructured information and understand opinions about a given subject. We chose to use four models for our analysis: TextBlob, Afinn, SentiWordNet 3.0, and VADER (Valence Aware Dictionary and Sentiment Reasoner). Some of these models, such as TextBlob and SentiWordNet 3.0 are well-suited for any text analysis while others, such as Afinn and VADER are specifically attuned to sentiments expressed in social media, such as punctuation and emoticons. Given that we do not have labels to evaluate our models, we chose to apply all four models in our analysis to have a framework for comparison of sentiments across models.

In aiming to understand what attributes of the Galaxy S8 and iPhone 8/X are most important to consumers, we first used bigram analysis. As mentioned in the book ‘Mining the Social Web’, bigram analysis is a simple and powerful way for clustering commonly co-occurring words from social media and blog text data. A quick review of different mobile phone features revealed an interesting pattern: many product features occur as bigrams. For example, consider iPhone X product features as outlined on the ‘Boost Mobile’ website for iPhone X: TrueDepth Camera, Intuitive Gestures, OLED Display, Portrait Lighting to name a few. Hence, we primarily utilized bigram analysis to determine the most important product attributes for Apple iPhone and

Samsung Galaxy. We then fed these most important features to our sentiment analysis models to understand how consumers of each device feel about these features.

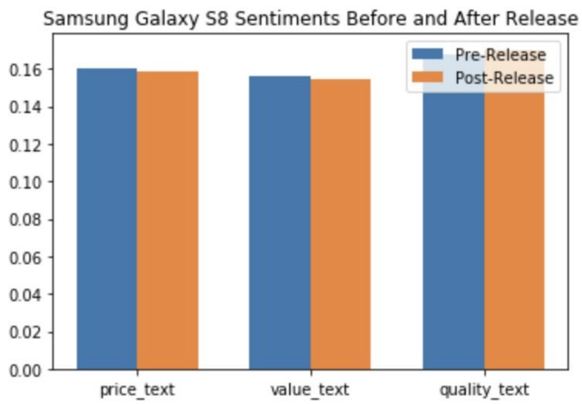
Results and Evaluation

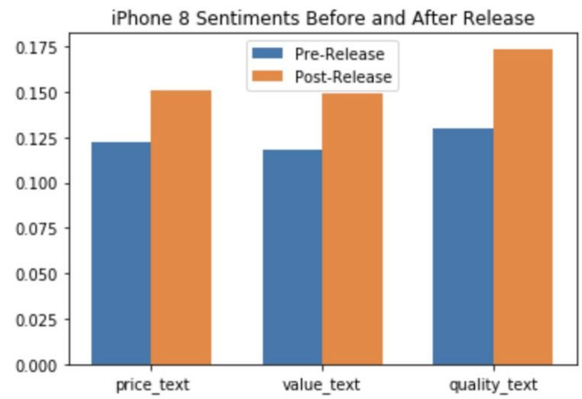
Firstly, we examined and compared the results of the four sentiment analysis models. As displayed by the graphs below, TextBlob, Afinn, and SentiWordNet models resulted in similar distributions of sentiments across all posts. The Vader model resulted in sentiments skewed towards the positive side.



In deciding which model to move forward with, we looked at several specific Twitter posts--due to their short length--and manually compared the sentiments of each model to find which model provided the most accurate one. With this manual analysis, we found that TextBlob seemed to provide the most accurate results and chose to move forward with this model.

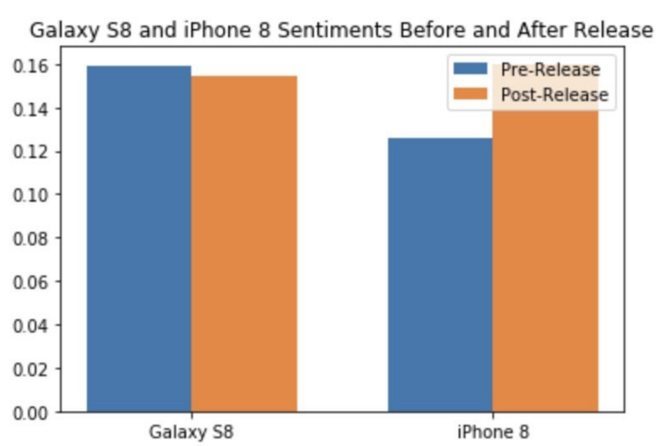
Next, we used the results of the sentiment analysis to understand consumers' sentiments around quality, price, and value of the devices. To do this, we identified posts related to quality, price, and value based on the words contained in each post and observed the sentiments of each post prior to and post release of each device. For Galaxy S8, we found that sentiments around quality, price, and value did not change much before and after its release. However, for iPhone 8, we found that sentiments around these three topics improved dramatically, most significantly around





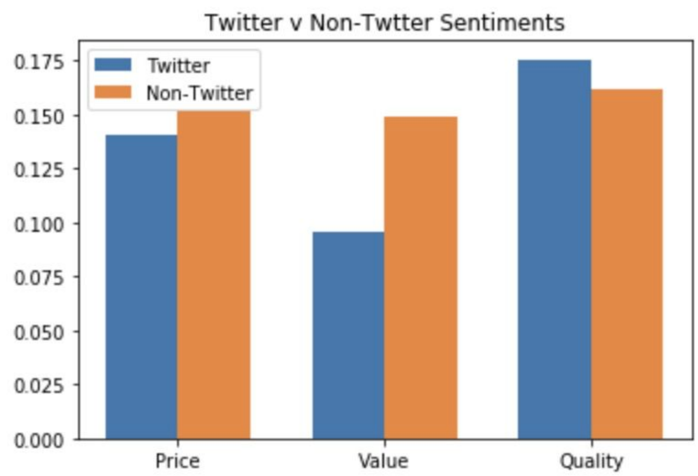
quality.

Overall, sentiments around iPhone 8 prior to its release were a bit lower than sentiments around Galaxy S8 prior to its release. However, sentiments for iPhone 8 became significantly more positive after its release whereas sentiments for Galaxy S8 became slightly more negative.



Given that Twitter is an important data source for this analysis, we chose to explore sentiments around quality, price, and value of Twitter posts compared to non-Twitter posts.

We found that sentiments on Twitter were more positive around value and price and more negative around quality compared to non-Twitter posts. This may be because Twitter posts are primarily opinions of direct consumers who may be more critical about price and the value they get out of the phone day-to-day. In contrast, non-Twitter posts, such as blogs, may be more critical of the long-term quality of phone features, not necessarily the day-to-day value to the consumer.



Next, to get a good picture of what features of the products are being discussed by the users the most, we looked at the 30 most frequently occurring bigrams in the reviews/post based on the social media analysis of Apple and Galaxy S8 posts. These features can be considered as the most important product attributes due to their frequent citations in posts. The following features were the most important attributes for Apple iPhone users in descending order:

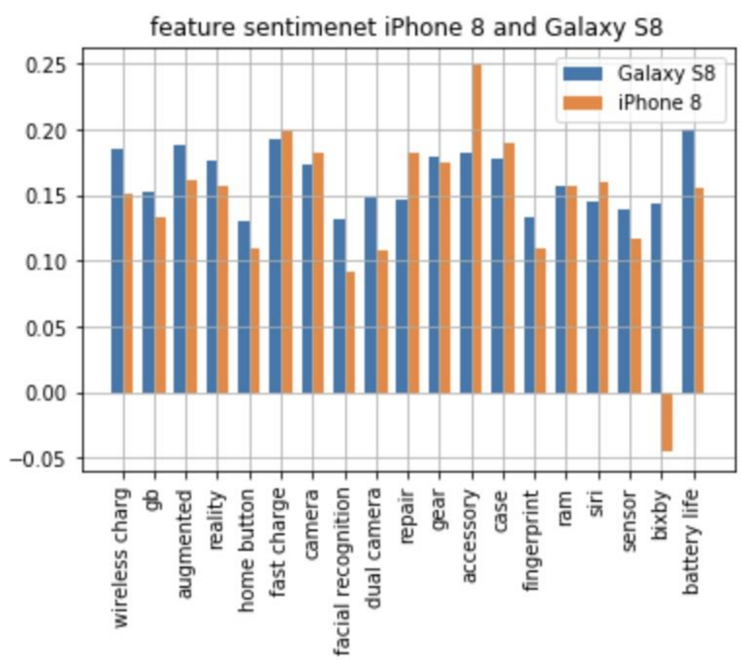
1. Wireless charging
2. Home button (Apple removed home button from iPhone X, so this generated a good amount of discussion among users)
3. Fast Charge
4. Portrait lighting (New feature introduced in iPhone)
5. Bionic Chip (Apple introduced this new feature iPhone 8 onwards)
6. Facial recognition (Apple introduced this new feature iPhone 8 onwards)
7. Charge Pad (Apple introduced this new feature iPhone 8 onwards)
8. Dual Camera

9. Oled Display (New OLED screen for iPhone X)

The following features were the most important attributes for Samsung Galaxy users in descending order:

1. Memory (as indicated by 'gb ram')
2. Infinity display
3. Iris Scanner
4. Bixby button
5. Fingerprint sensor/scanner
6. Bixby voice
7. Facial recognition
8. Battery life
9. Wireless charger
10. Red tint (an issue faced by some users with Samsung products)

To understand users' sentiments around these most important features, we applied the sentiment analysis to posts that contained content around these features.



We found that sentiments are generally more positive for iPhone 8 than those for Galaxy S8. In particular, there were more positive sentiments around iPhone 8 accessories, but more positive sentiments around Galaxy S8 battery life.

Conclusion and Recommendations

From a product manager's perspective, we can look at the data and get a strong idea about what features are most important to the users of our device. Once we have determined a feature to look at it is possible to analyze the distribution of how people feel about that specific feature. Using this knowledge, product managers can better allocate resources to develop or fix aspects of the product that is subpar. Also, knowing the strengths of a product from a user perspective is a valuable insight for a marketing team to help them determine what features of the product to promote to potential users.

This analysis shows that Samsung's battery life is a great part of the device and is driving much of what users are talking about. People also seem to really enjoy the dual camera. However, they are lacking in the repair and maintenance aspect of their business. This could be a good area of focus for a product manager.

Apple has been doing a great job when it comes to accessories and complementary aspects of their product. From the bi-gram analysis, we can see that apple watch, apple tv, and the apple store are all things that are very important to their users. Implementing more features that allow Apple devices to connect seamlessly to each other should be a priority. On the other hand, there are a few areas where Apple can improve so they can be more competitive in the market. Facial recognition and wireless charging are two features that rank highly among what consumers want and where iPhone 8 is falling short of their competitor.

Future Improvements

The methodology we used for analyzing tweets was identical to both blogs and longer articles. We assigned one sentiment score over for each review/tweet no matter how long the article was. An improvement would be to segment the longer articles so we can try and isolate parts of an article that are relevant to a specific feature and do analysis just on the relevant portion.

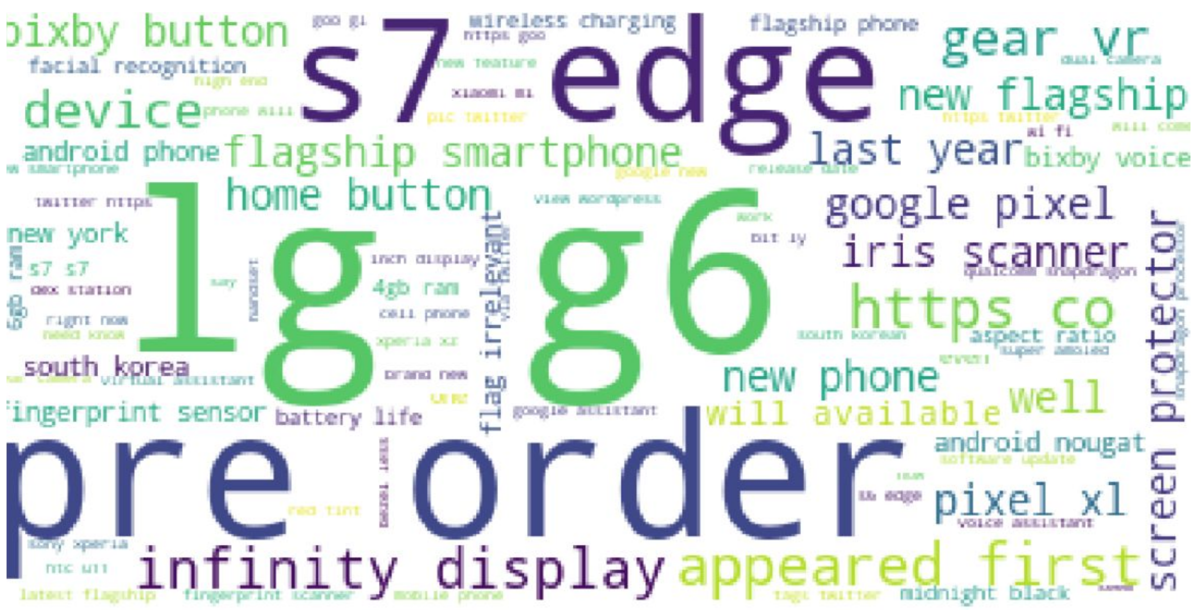
We also would like to implement our own sentiment analyzer by creating an LSTM RNN. A custom RNN would be useful to more accurately analyze review and text that are in this specific domain. While the out of the box sentiment analyzers performed fairly well, they were not designed specifically with our corpus in mind and has room for improvement. We would model this approach after SES, a self-supervised syntax based method of classification, that has been developed from Peking University.

References

1. https://www.digitaltrends.com/mobile/android-vs-ios/
2. https://www.gamingscan.com/ios-vs-android/
3. https://www.tomsguide.com/us/iphone-x-vs-galaxy-s8,review-4864.html
4. https://monkeylearn.com/sentiment-analysis/
5. https://www.webpages.uidaho.edu/~stevel/504/mining-the-social-web-2nd-edition.pdf
6. https://textblob.readthedocs.io/en/dev/
7. https://medium.com/analytics-vidhya/simplifying-social-media-sentiment-analysis-using-vader-in-python-f9e6ec6fc52f
8. http://corpustext.com/reference/sentiment_afinn.html
9. http://nmis.isti.cnr.it/sebastiani/Publications/LREC10.pdf
10. http://aclweb.org/anthology/Y09-2018

Additional Plots

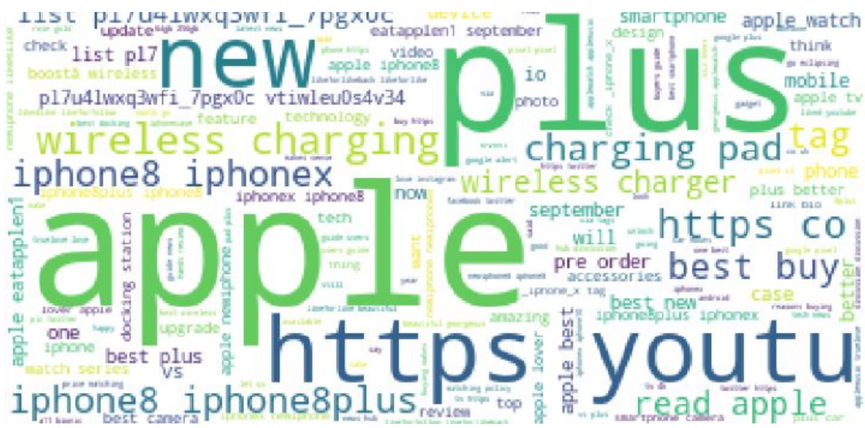
iPhone 8/X Word Cloud



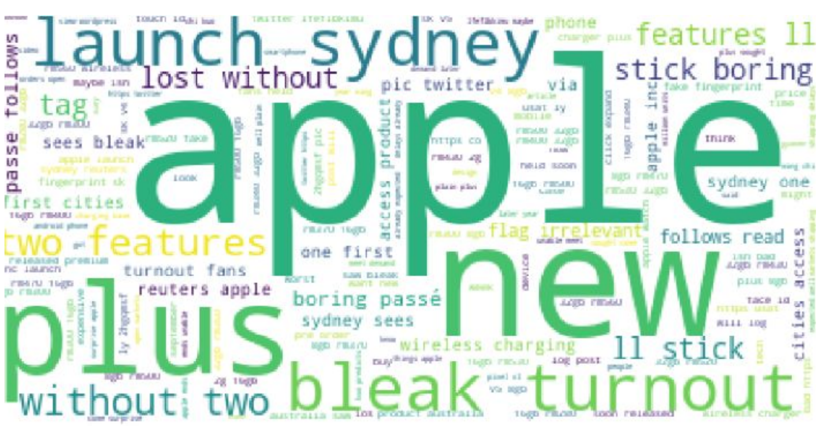
Galaxy S8 Word Cloud

```
Sound Bite Text: Report: Galaxy S8 Prototypes Impressed At MWC 2017 dlvr.it/NYHFyP @slideme pic.twitter.com/OW9WIvw2uX
TextBlob_sentiment: 1.0
Afn_Sent: 3.0
Vader_Sentiment: 0.4767
Sentwordnet_sent: -0.08

Sound Bite Text: Apple Leak Reveals iPhone 8 Nasty Surprises newssummedup.com/a/wps1fy fb.me/3gKqVChkT
TextBlob_sentiment: -1.0
Afn_Sent: -4.0
Vader_Sentiment: -0.6249
Sentwordnet_sent: 0.03

Sound Bite Text: Just received my galaxy S8 plus and one of my earphones don't work Sprint won't do anything. Service is terr
ible. @sprint #sprintservice
TextBlob_sentiment: -1.0
Afn_Sent: 0.0
Vader_Sentiment: -0.4767
Sentwordnet_sent: 0.01
```

Sentiment Analyzer comparison

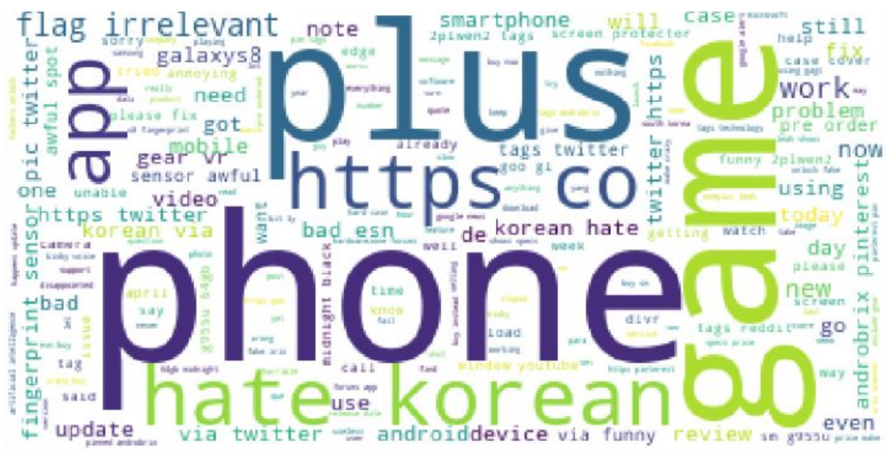


Positive iPhone 8 Word Cloud



Negative iPhone 8 Word Cloud

Posiitve Galaxy S8 Word Cloud



Negative Galaxy S8 Word Cloud

95-851: Social Media Analysis

Data Clan

Part 1: Sentiment Analysis

1. Load Data

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
plt.style.use('ggplot')
%matplotlib inline
```

In [2]:
```
Both_iPhones_data = pd.read_csv('Both_iPhones_data.csv')
both_samsung_and_iphone_data = pd.read_csv('both_samsung_and_iphone_dat
a.csv')
iphone8_data = pd.read_csv('iphone8_data.csv')
iphoneX_data = pd.read_csv('iphoneX_data.csv')
samsung_data = pd.read_csv('samsung_data.csv')
```

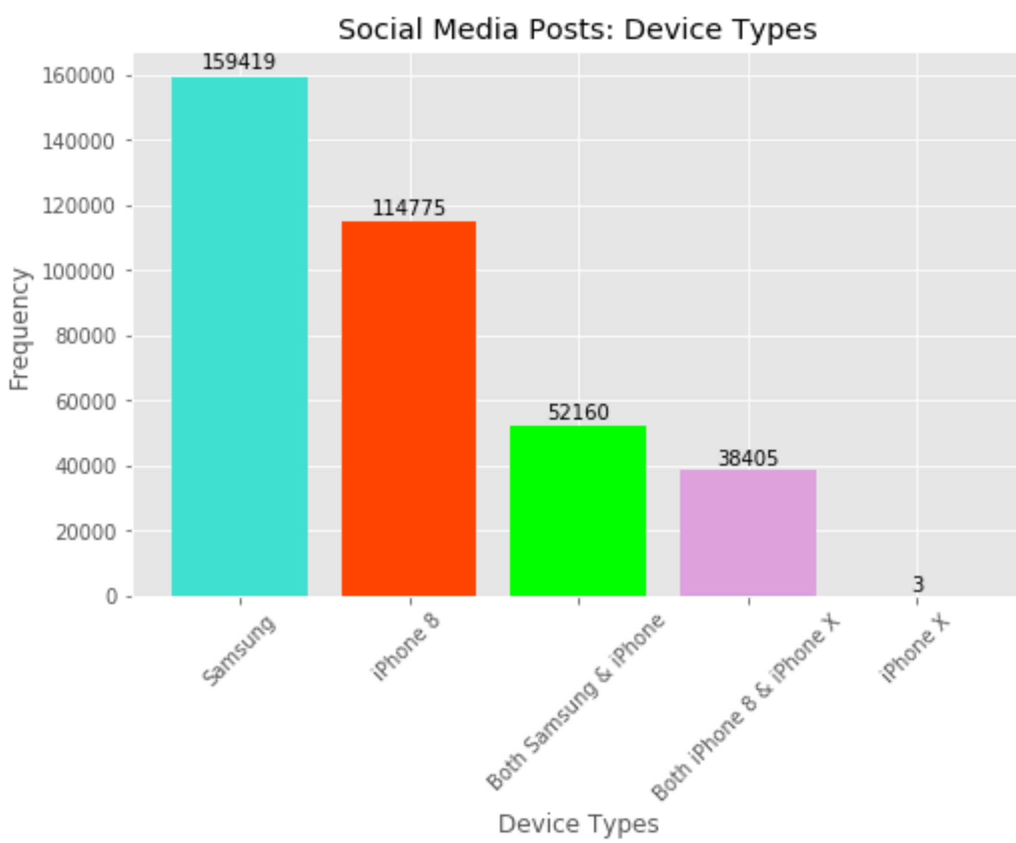
2. Exploratory Data Analysis

In [3]:
```
# add new column with device type
Both_iPhones_data['device_type'] = 'Both iPhone 8 & iPhone X'
both_samsung_and_iphone_data['device_type'] = 'Both Samsung & iPhone'
iphone8_data['device_type'] = 'iPhone 8'
iphoneX_data['device_type'] = 'iPhone X'
samsung_data['device_type'] = 'Samsung'
```

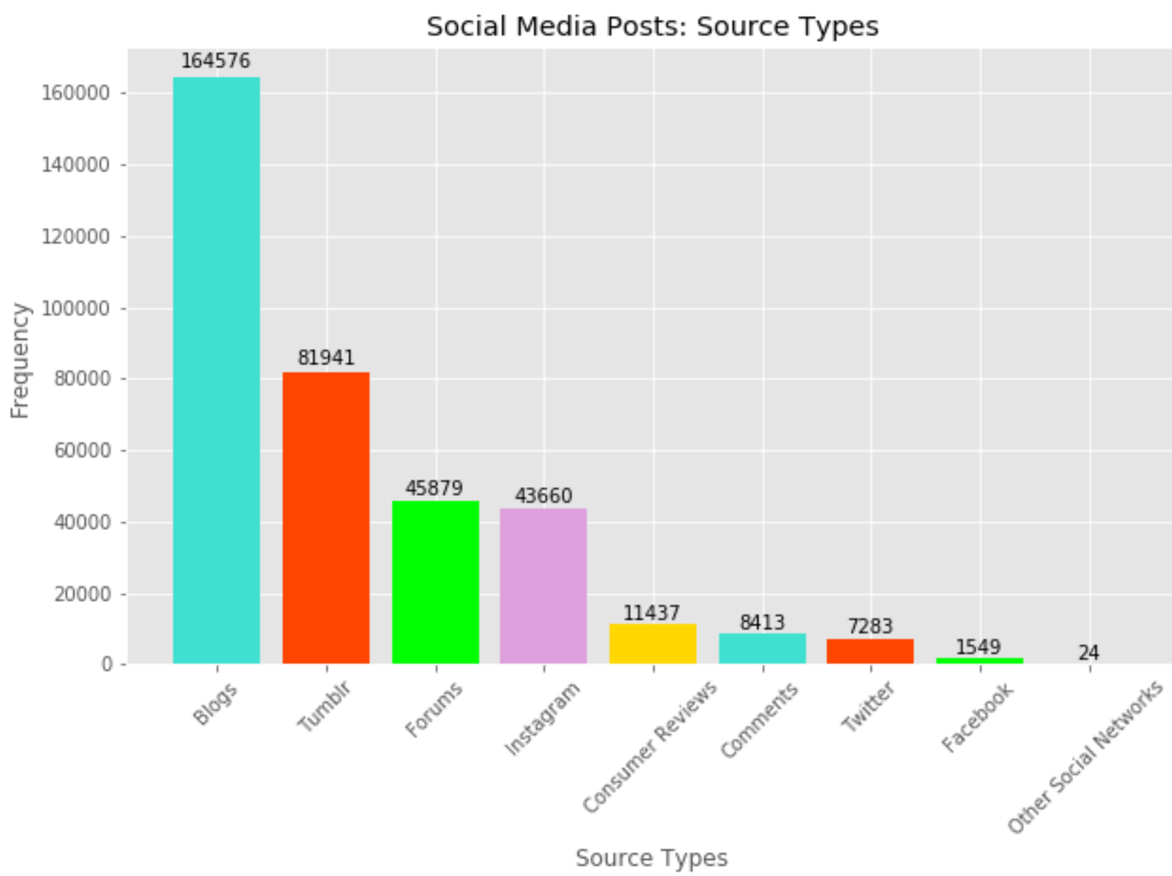
In [4]:
```
# concatenate all data into one df
all_data = pd.concat([Both_iPhones_data,both_samsung_and_iphone_data,iph
one8_data,iphoneX_data,samsung_data], sort=True)
```

In [5]:
```
def autolabel(rects):
    # attach some text labels
    for rect in rects:
        height = rect.get_height()
        plt.text(rect.get_x() + rect.get_width()/2., 1.01*height,
                '%d' % int(height),
                ha='center', va='bottom')
```

In [6]:
```
from collections import Counter
from operator import itemgetter
c=['turquoise', 'orangered', 'lime', 'plum', 'gold']

counts_per_device_type = Counter()
for device_type in all_data.device_type:
    counts_per_device_type[device_type] +=1

counts_per_device_type_sorted = sorted(counts_per_device_type.items(),
                                  reverse=True,
                                  key=itemgetter(1))

device_types = [device_type for device_type, count in counts_per_device_
type_sorted]
counts = [count for device_type, count in counts_per_device_type_sorted]

plt.figure(figsize=(8, 5))
bar1 = plt.bar(range(len(device_types)), counts, color = c)
plt.xlabel('Device Types')
plt.xticks(range(len(device_types)), device_types, rotation=45)
plt.ylabel('Frequency')
plt.title("Social Media Posts: Device Types")
autolabel(bar1)
plt.show()
```



In [7]:
```
from collections import Counter
from operator import itemgetter
c=['turquoise', 'orangered', 'lime', 'plum', 'gold']
counts_per_source_type = Counter()
for source_type in all_data["Source Type"]:
    counts_per_source_type[source_type] +=1

counts_per_source_type_sorted = sorted(counts_per_source_type.items(),
                                       reverse=True,
                                       key=itemgetter(1))

source_types = [source_type for source_type, count in counts_per_source_
type_sorted]
counts = [count for source_type, count in counts_per_source_type_sorted]

plt.figure(figsize=(10, 6))
bar2 = plt.bar(range(len(source_types)), counts, color = c)
plt.xlabel('Source Types')
plt.xticks(range(len(source_types)), source_types, rotation=45)
plt.ylabel('Frequency')
plt.title("Social Media Posts: Source Types")
autolabel(bar2)
plt.show()
```



3. Data Preprocessing

In [9]:
```
%%time
# remove punctuation
import string
all_data.LowerText = all_data.LowerText.apply(lambda x: x.translate(stri
ng.punctuation))
```

```
Wall time: 16.4 s
```

In [10]:
```
%%time
# remove numbers
all_data.LowerText = all_data.LowerText.apply(lambda x: x.translate(stri
ng.digits))
```

```
Wall time: 15 s
```

In [11]:
```
%%time
#tokenize
all_data["Tokenized_Text"] = all_data.LowerText.apply(lambda x: x.split(
' '))
```

```
Wall time: 8.55 s
```

In [12]:
```
%%time
# stem
import nltk
from nltk.stem.porter import PorterStemmer
all_data["Stemmed_text"] = all_data.Tokenized_Text.apply(lambda x: [Port
erStemmer().stem(y) for y in x])
```

```
Wall time: 13min
```

In [15]:
```
%%time
# remove stopwords
from nltk.corpus import stopwords
# import nltk
# nltk.download('stopwords')
sw = stopwords.words('english')
all_data.Stemmed_text = all_data.Stemmed_text.apply(lambda x: [item for
item in x if item not in sw])
```

```
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\dpc50\AppData\Roaming\nltk_data...
[nltk_data]   Unzipping corpora\stopwords.zip.

Wall time: 1min 12s
```

4. Create Sentiment Analysis Models

In [16]:
```
%%time
from textblob import TextBlob
all_data['TextBlob_sentiment'] = np.array([TextBlob(text).sentiment.pola
rity for text in all_data['Sound Bite Text']])
```

```
Wall time: 6min 30s
```

In [17]:
```
%%time
from afinn import Afinn
afn = Afinn(emoticons=True)
all_data["Afn_Sent"] = all_data["Sound Bite Text"].apply(lambda x: afn.s
core(x))
```

```
Wall time: 21min 18s
```

In [50]:
```
import spacy
nlp = spacy.load('en')
from nltk.corpus import sentiwordnet as swn

def analyze_sentiment_sentiwordnet_lexicon(review,
                                           verbose=False):

    # tokenize and POS tag text tokens
    tagged_text = [(token.text, token.tag_) for token in nlp(review)]
    pos_score = neg_score = token_count = obj_score = 0
    # get wordnet synsets based on POS tags
    # get sentiment scores if synsets are found
    for word, tag in tagged_text:
        ss_set = None
        if 'NN' in tag and list(swn.senti_synsets(word, 'n')):
            ss_set = list(swn.senti_synsets(word, 'n'))[0]
        elif 'VB' in tag and list(swn.senti_synsets(word, 'v')):
            ss_set = list(swn.senti_synsets(word, 'v'))[0]
        elif 'JJ' in tag and list(swn.senti_synsets(word, 'a')):
            ss_set = list(swn.senti_synsets(word, 'a'))[0]
        elif 'RB' in tag and list(swn.senti_synsets(word, 'r')):
            ss_set = list(swn.senti_synsets(word, 'r'))[0]
        # if senti-synset is found
        if ss_set:
            # add scores for all found synsets
            pos_score += ss_set.pos_score()
            neg_score += ss_set.neg_score()
            obj_score += ss_set.obj_score()
            token_count += 1

    # aggregate final scores
    final_score = pos_score - neg_score
    norm_final_score = 0
    if(token_count != 0):
        norm_final_score = round(float(final_score) / token_count, 2)

    final_sentiment = 'positive' if norm_final_score >= 0 else 'negativ
e'
    if verbose:
        norm_obj_score = round(float(obj_score) / token_count, 2)
        norm_pos_score = round(float(pos_score) / token_count, 2)
        norm_neg_score = round(float(neg_score) / token_count, 2)
        # to display results in a nice table
        sentiment_frame = pd.DataFrame([[final_sentiment, norm_obj_score
, norm_pos_score,
                                         norm_neg_score, norm_final_scor
e]],
                                       columns=pd.MultiIndex(levels=[['S
ENTIMENT STATS:'],
                                                              ['Predicted
Sentiment', 'Objectivity',
                                                               'Positive'
, 'Negative', 'Overall']],
                                                              labels=[[0,
0,0,0,0],[0,1,2,3,4]]))
        print(sentiment_frame)
```

```
    return norm_final_score
```

In [51]:
```
%%time
all_data["Sentwordnet_sent"] = all_data["Sound Bite Text"].apply(lambda
x: analyze_sentiment_sentiwordnet_lexicon(x))
```

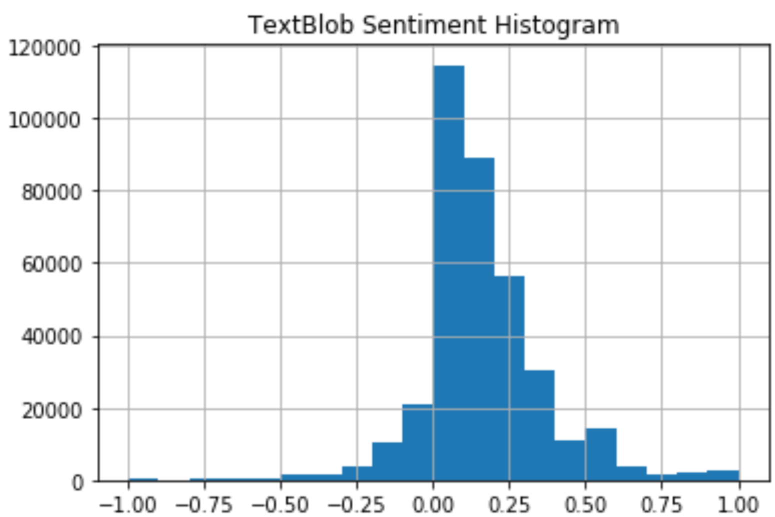
Wall time: 4h 15min 40s

In [21]:
```
%%time
# vader model
from vaderSentiment.vaderSentiment import SentimentIntensityAnalyzer
sia = SentimentIntensityAnalyzer()
all_data['Vader_Sentiment'] =  all_data['Sound Bite Text'].apply(lambda
x: float(sia.polarity_scores(x)['compound']))
```

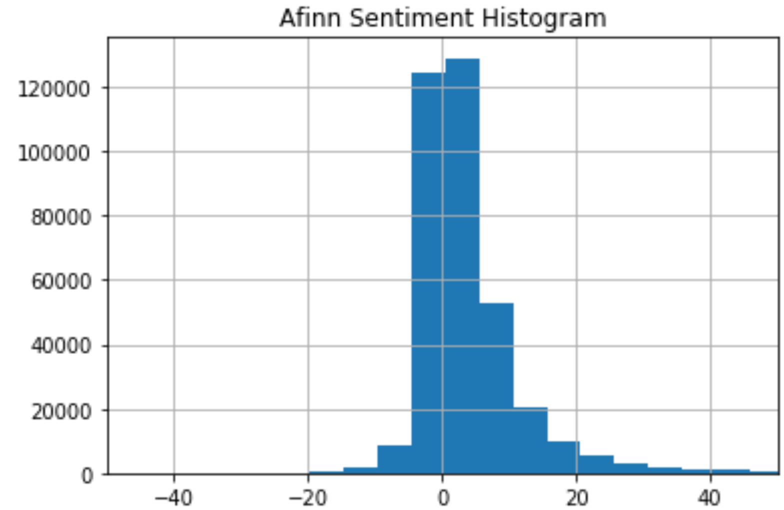
Wall time: 15min 3s

In [52]:
```
all_data.to_csv("all_data_sent.csv",index = False)
```

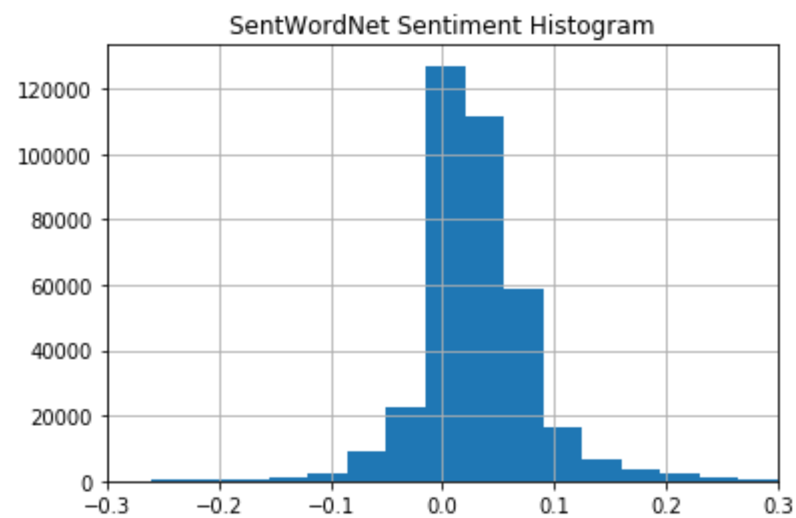
In [139]:
```
all_data.TextBlob_sentiment.hist(bins = 20)
plt.title("TextBlob Sentiment Histogram")
plt.show()
```



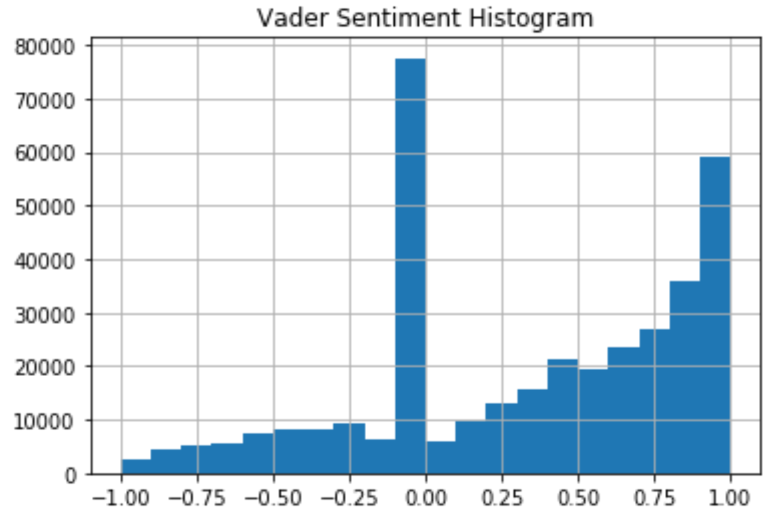
In [140]:
```
all_data.Afn_Sent.hist(bins = 100)
plt.title("Afinn Sentiment Histogram")
plt.xlim(-50,50)
plt.show()
```



In [141]:
```
all_data.Sentwordnet_sent.hist(bins = 50)
plt.title("SentWordNet Sentiment Histogram")
plt.xlim(-.3,.3)
plt.show()
```



In [142]:
```
all_data.Vader_Sentiment.hist(bins = 20)
plt.title("Vader Sentiment Histogram")

plt.show()
```



In [117]:
```
twitter_data = all_data[all_data["Source Type"] == "Twitter"]
twitter_data.shape
```

Out[117]: (7283, 22)

In [121]:
```
twitter_df = pd.DataFrame(twitter_data)
twitter_df.columns = all_data.columns
twitter_df["sent_diff"] = abs(twitter_df.TextBlob_sentiment - twitter_df
.Sentwordnet_sent)
```

In [124]:
```
sorted_twitter = twitter_df.sort_values(by = "sent_diff",ascending = Fal
se)
```

In [129]:
```
for i in range(5):
    text = sorted_twitter.iloc[i,5]
    print(text)

    for y in range(14,18):
        print(sorted_twitter.columns[y],sorted_twitter.iloc[i,y])
    print()
```

```
I added a video to a @YouTube playlist youtu.be/0MxC9sVp6mg?a Apple Iph
one 8 Ringtone Awesome (Never Ever)
TextBlob_sentiment 1.0
Afn_Sent 4.0
Vader_Sentiment 0.6249
Sentwordnet_sent -0.1

Report: Galaxy S8 Prototypes Impressed At MWC 2017 dlvr.it/NYHFyP @slid
eme pic.twitter.com/OW9WIvw2uX
TextBlob_sentiment 1.0
Afn_Sent 3.0
Vader_Sentiment 0.4767
Sentwordnet_sent -0.08

Apple Leak Reveals iPhone 8 Nasty Surprises newssummedup.com/a/wps1fy f
b.me/3gKqVChkT
TextBlob_sentiment -1.0
Afn_Sent -4.0
Vader_Sentiment -0.6249
Sentwordnet_sent 0.03

Just received my galaxy S8 plus and one of my earphones don't work Spri
nt won't do anything. Service is terrible. @sprint #sprintservice
TextBlob_sentiment -1.0
Afn_Sent 0.0
Vader_Sentiment -0.4767
Sentwordnet_sent 0.01

What’s the Best #Samsung Galaxy S8 Case? 5 Affordable Ways to Protect Y
our Investment muo.co/2s3YDH3 pic.twitter.com/8y49tQEIPJ
TextBlob_sentiment 1.0
Afn_Sent 6.0
Vader_Sentiment 0.7783
Sentwordnet_sent 0.0
```

In [118]:
```
random_data = np.random.permutation(twitter_data)[:10]
random_tweets_df = pd.DataFrame(random_data)
random_tweets_df.columns = all_data.columns
```

In [119]:
```

```

In [120]:

Out[120]:

	LowerText	Media Type	Post ID	Post Type	Published Date (GMT-04:00) New York	Sound Bite Te
3	i liked a @youtube video youtu.be/n2wjlnnjrhi?...	Link	911756621767958530	Original	Sep 23, 2017 8:58:12 PM	I liked a @YouTube vid youtu.be/n2wJLnNJRHI'
7	i liked a @youtube video youtu.be/xkjblk52d1w?...	Link	854526508865646592	Original	Apr 18, 2017 10:46:10 PM	I liked a @YouTube vid youtu.be/xKjbLK52D1w'
4	galaxy s8 screen resolution confirmed in lates...	Link	844516071294074880	Original	Mar 22, 2017 7:48:16 AM	Galaxy S8 scre resolution confirmec late:
5	мне понравилось видео "iphone 8 plus vs. galax...	No Media	912175638601195521	Original	Sep 25, 2017 12:43:14 AM	Мне понравилось вид "iPhone 8 Plus vs. Gala:
2	android circuit: new galaxy s8 issues, microso...	Image; Link	860637802731511808	Original	May 5, 2017 7:30:16 PM	Android Circuit: N Galaxy S8 Issu Micros
1	lito 3 in 1 electroplating hard pc phone case ...	Link	904644944434008066	Original	Sep 4, 2017 5:58:56 AM	LITO 3 in 1 Electroplati Hard PC Phone Case
8	i added a video to a @youtube playlist youtu.b...	Link	859054448336875520	Original	May 1, 2017 10:38:35 AM	I added a video t @YouTube play youtu.l
0	leak confirms iphone 8 will be larger than iph...	Link	870450876913156096	Original	Jun 1, 2017 9:23:55 PM	Leak Confirms iPhon Will Be Larger Than iPl
6	the new samsung galaxy s8 has finally arrived....	Image; Link	855416909285572609	Original	Apr 21, 2017 9:44:18 AM	The new Samsung Gala S8 has finally arrived
9	no root adblocker & package disabler - works o...	Link	915643244226269184	Original	Oct 4, 2017 2:22:15 PM	NO ROOT AdBlocke Package Disabler - wo

10 rows × 23 columns

In [81]:
```
for i in range(10):
    text = random_tweets_df.iloc[i,5]
    print(text)

    for y in range(14,18):
        print(random_tweets_df.columns[y],random_tweets_df.iloc[i,y])
    print()
```

(10 iPhone HACKS and TRICKS 2017) has been published on My Iphone 8 - myiphone8.co.uk/2017/10/15/10-… pic.twitter.com/xGGH3M88JX
TextBlob_sentiment 0.0
Afn_Sent 0.0
Vader_Sentiment -0.3034
Sentwordnet_sent 0.0

My phone is beat to fuck...where this iPhone 8 at?
TextBlob_sentiment 0.0
Afn_Sent -4.0
Vader_Sentiment 0.0
Sentwordnet_sent 0.06

California-Based Company Debuts New, Stylish Case Just in Time for the iPhone 8, iPhone 8 Plus and iPhone X Release ift.tt/2wZOIDF
TextBlob_sentiment 0.3181818181818182
Afn_Sent 0.0
Vader_Sentiment 0.0
Sentwordnet_sent 0.07

Samsung Galaxy S8 SM-G950U - 64GB - Midnight Black (AT&T) Smartphone pic.twitter.com/gvpDJKFzHC
TextBlob_sentiment -0.16666666666666666
Afn_Sent 0.0
Vader_Sentiment 0.0
Sentwordnet_sent -0.07

Surprise: Galaxy S8 has the 'best smartphone display' buff.ly/2o2gJGl
TextBlob_sentiment 1.0
Afn_Sent 3.0
Vader_Sentiment 0.743
Sentwordnet_sent 0.23

I liked a @YouTube video youtu.be/frbB28ofS_c?a iPhone 8 Triple Bad News Leak
TextBlob_sentiment -0.04999999999999993
Afn_Sent -2.0
Vader_Sentiment -0.4767
Sentwordnet_sent -0.12

The LG V30 could get curves like the Galaxy S8 fb.me/8v8zRhmer
TextBlob_sentiment 0.0
Afn_Sent 2.0
Vader_Sentiment 0.3612
Sentwordnet_sent 0.04

Galaxy S8 vs 7 Plus vs LG G6 vs Pixel vs 3T SPEED Test! youtu.be/OX4JucpvbJM via @YouTube Really great test of all the top phones.
TextBlob_sentiment 0.65
Afn_Sent 5.0
Vader_Sentiment 0.7569
Sentwordnet_sent 0.09

Samsung's Galaxy S8 looks great but it still won't convince me to replace my iPhone. Maybe it's customer loyalty, maybe it's just right.
TextBlob_sentiment 0.5428571428571429
Afn_Sent 10.0

```
Vader_Sentiment 0.7343
Sentwordnet_sent -0.02

Apple unlikely to switch to USB-C on the iPhone 8 because you can’t hav
e nice things dlvr.it/NWjGCG
TextBlob_sentiment 0.049999999999999999
Afn_Sent 2.0
Vader_Sentiment 0.4215
Sentwordnet_sent 0.19
```

5. Apply Sentiment Analysis to Quality, Price, and Value of Devices

In [1]:
```
import pandas as pd
all_data = pd.read_csv("all_data_sent.csv")
all_data.head()
```

```
C:\Users\dpc50\Anaconda3\lib\site-packages\IPython\core\interactiveshel
l.py:3020: DtypeWarning: Columns (7,9) have mixed types. Specify dtype
option on import or set low_memory=False.
  interactivity=interactivity, compiler=compiler, result=result)
```

Out[1]:

	LowerText	Media Type	Post ID	Post Type	Published Date (GMT-04:00) New York	Sound Bite Text	Source Type
0	following the naming system of the past severa...	Link	718bbba167877e763cfe851413849ed8	Original	May 8, 2017 7:39:37 AM	Following the naming system of the past severa...	Blogs
1	the processing cost for the oled-based 3d touc...	No Media	17194239754920322211	Original	May 19, 2017 7:42:09 AM	The processing cost for the OLED-based 3D Touc...	Blogs
2	the processing cost for the oled-based 3d touc...	No Media	5279493144373937172	Original	May 19, 2017 7:42:09 AM	The processing cost for the OLED-based 3D Touc...	Blogs
3	the processing cost for the oled-based 3d touc...	No Media	10937716975525293181	Original	May 19, 2017 7:46:00 AM	The processing cost for the OLED-based 3D Touc...	Blogs
4	we have 9 exciting iphone 8 rumors for the ult...	No Media	http://learnbonds.com/133761/iphone-8-rumors-f...	Original	May 20, 2017 10:20:00 AM	We Have 9 Exciting iPhone 8 Rumors for the Ult...	Blogs

In [2]:
```
value = ["value","worth","use","appreciate","advantage","benefit","purpo
se"]
price = ["\$","price","buy","sell","cost","demand","expensive","cheap",
"affordable","money"]
quality = ["quality","design","desplay","look","screen","battery","wate
r","proof","performance","lens","speaker"]
```

In [4]:
```
samsung_data = all_data[all_data.device_type == "Samsung"]
```

In [6]:
```
%%time
temp = None
for i, word in enumerate(price):
    if i == 0:
        temp = all_data.LowerText.str.contains(word)
    else:
        temp = temp | all_data.LowerText.str.contains(word)

all_data['price_text'] =temp
print(all_data['price_text'].sum())
```

```
91537
Wall time: 4.09 s
```

In [7]:
```
%%time
temp = None
for i, word in enumerate(value):
    if i == 0:
        temp = all_data.LowerText.str.contains(word)
    else:
        temp = temp | all_data.LowerText.str.contains(word)

all_data['value_text'] =temp
print(all_data['value_text'].sum())
```

```
99398
Wall time: 2.73 s
```

In [8]:
```
%%time
temp = None
for i, word in enumerate(quality):
    if i == 0:
        temp = all_data.LowerText.str.contains(word)
    else:
        temp = temp | all_data.LowerText.str.contains(word)

all_data['quality_text'] =temp
print(all_data['quality_text'].sum())
```

```
134186
Wall time: 4.31 s
```

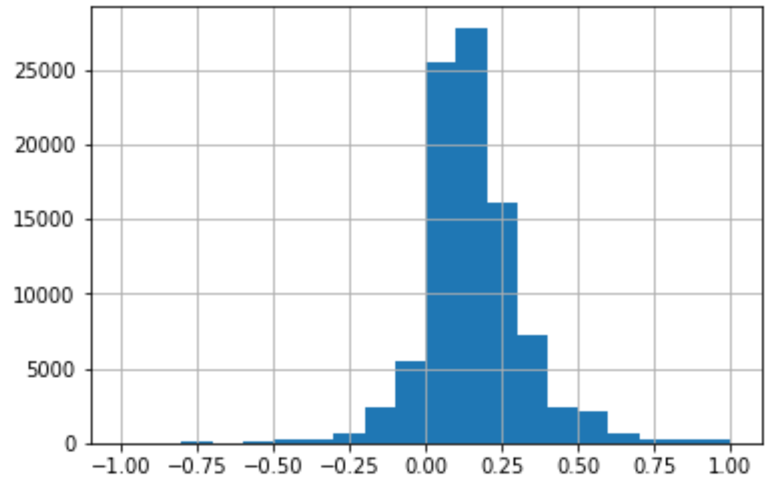
In [9]:
```
all_data.device_type.unique()
```

Out[9]:
```
array(['Both iPhone 8 & iPhone X', 'Both Samsung & iPhone', 'iPhone 8',
       'iPhone X', 'Samsung'], dtype=object)
```

In [10]:
```
import matplotlib.pyplot as plt
```

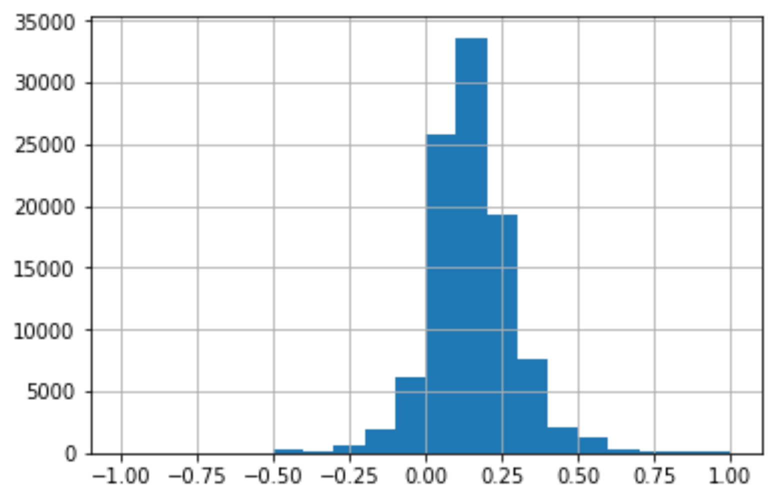
In [82]:
```
all_data[all_data['price_text'] == True].TextBlob_sentiment.hist(bins = 20)
```

Out[82]: <matplotlib.axes._subplots.AxesSubplot at 0x249efc56860>



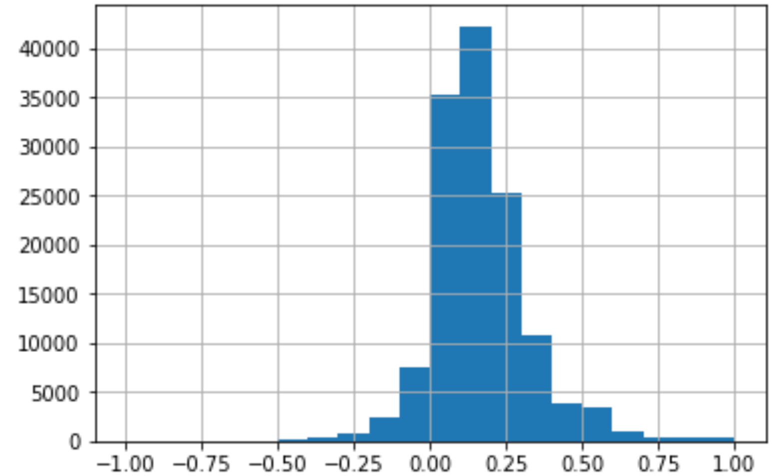
In [83]:
```
all_data[all_data['value_text'] == True].TextBlob_sentiment.hist(bins = 20)
```

Out[83]: <matplotlib.axes._subplots.AxesSubplot at 0x249e6feecc0>



In [84]:
```
all_data[all_data['quality_text'] == True].TextBlob_sentiment.hist(bins
= 20)
```

Out[84]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x249f2abff28>
```



In [15]:
```
all_data.columns
```

Out[15]:
```
Index(['LowerText', 'Media Type', 'Post ID', 'Post Type',
       'Published Date (GMT-04:00) New York', 'Sound Bite Text', 'Sourc
e Type',
       'contain8', 'containSamsung', 'containX', 'containiPhone',
       'device_type', 'Tokenized_Text', 'Stemmed_text', 'TextBlob_senti
ment',
       'Afn_Sent', 'Vader_Sentiment', 'Sentwordnet_sent', 'price_text',
       'value_text', 'quality_text', 'Date'],
      dtype='object')
```

6. Sentiment Analysis Before and After Releases

In [14]:
```
%%time
all_data['Date'] = pd.to_datetime(all_data["Published Date (GMT-04:00) N
ew York"])
```

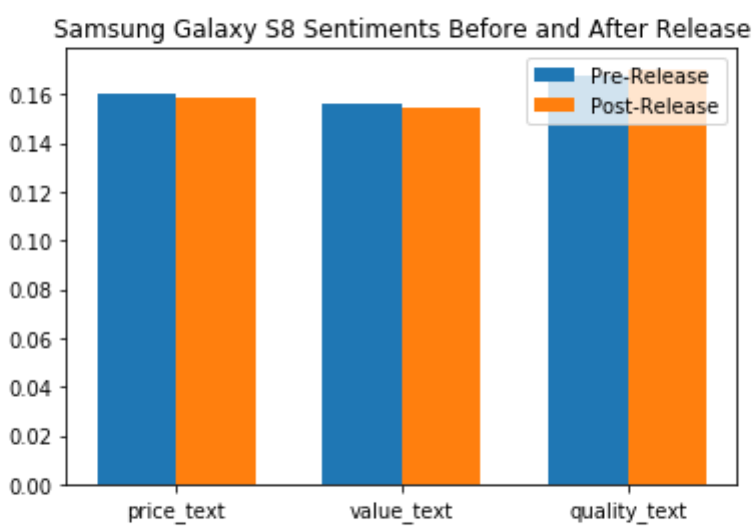
```
Wall time: 56.2 s
```

In [16]:
```
samsung_release = "2017-04-21"
iphone_release = "2017-09-12"
```

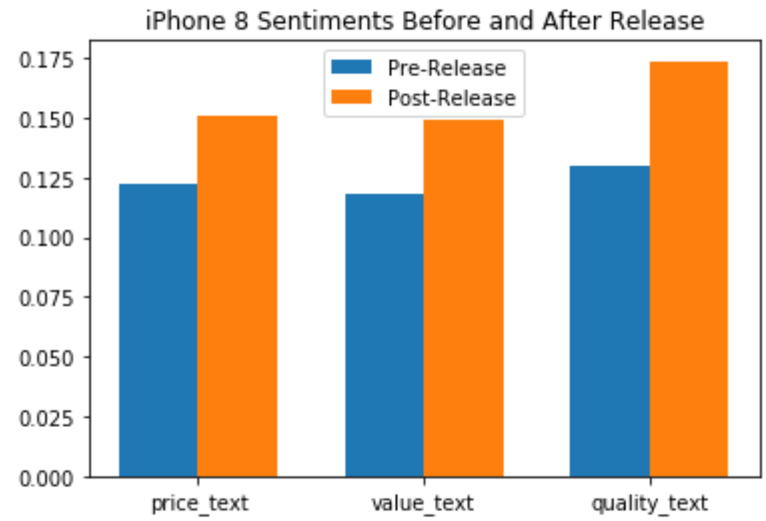
In [92]:
```
import numpy as np
samsung_data = []
features = ["price_text","value_text","quality_text"]
for feat in features:
    sub_data = all_data[all_data.device_type == "Samsung"]
    sub_data = sub_data[sub_data[feat] == True]
    samsung_data.append([sub_data[sub_data.Date < samsung_release].TextB
lob_sentiment.mean(),sub_data[sub_data.Date > samsung_release].TextBlob_
sentiment.mean()])

samsung_data = np.array(samsung_data)
```

In [93]:
```
import numpy as np
iphone_data = []

for feat in features:
    sub_data = all_data[all_data.device_type == "iPhone 8"]
    sub_data = sub_data[sub_data[feat] == True]
    iphone_data.append([sub_data[sub_data.Date < iphone_release].TextBlo
b_sentiment.mean(),sub_data[sub_data.Date > iphone_release].TextBlob_sen
timent.mean()])

iphone_data = np.array(iphone_data)
```

In [94]:
```
n = np.array(list(range(3)))
plt.bar(n,samsung_data[:,0],.35,label = "Pre-Release")
plt.bar(n+.35,samsung_data[:,1],.35,label = "Post-Release")
plt.xticks(n+.35/2,features)
plt.legend()
plt.title("Samsung Galaxy S8 Sentiments Before and After Release")
plt.show()
```



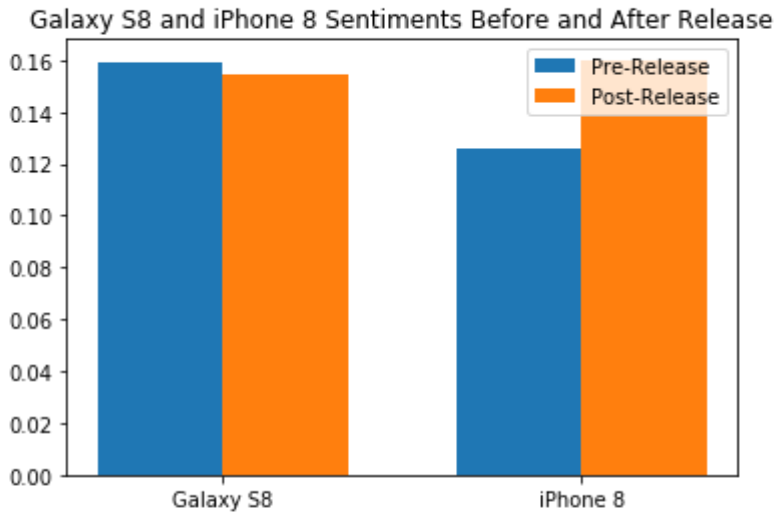
In [95]:
```
n = np.array(list(range(3)))
plt.bar(n,iphone_data[:,0],.35,label = "Pre-Release")
plt.bar(n+.35,iphone_data[:,1],.35,label = "Post-Release")
plt.xticks(n+.35/2,features)
plt.legend()
plt.title("iPhone 8 Sentiments Before and After Release")
plt.show()
```



In [96]:
```
samsung_sent_pre_post  = [all_data[all_data.device_type == "Samsung"][all_data[all_data.device_type == "Samsung"].Date < samsung_release].TextBlob_sentiment.mean(),all_data[all_data.device_type == "Samsung"][all_data[all_data.device_type == "Samsung"].Date > samsung_release].TextBlob_sentiment.mean()]
```

In [97]:
```
iphone_8_sent_pre_post  = [all_data[all_data.device_type == "iPhone 8"][all_data[all_data.device_type == "iPhone 8"].Date < iphone_release].TextBlob_sentiment.mean(),all_data[all_data.device_type == "iPhone 8"][all_data[all_data.device_type == "iPhone 8"].Date > iphone_release].TextBlob_sentiment.mean()]
```

In [138]:
```
n = np.array(list(range(2)))
plt.bar(n,[samsung_sent_pre_post[0],iphone_8_sent_pre_post[0]],.35,label
= "Pre-Release")
plt.bar(n+.35,[samsung_sent_pre_post[1],iphone_8_sent_pre_post[1]],.35,l
abel = "Post-Release")
plt.xticks(n+.35/2,["Galaxy S8","iPhone 8"])
plt.legend()
plt.title("Galaxy S8 and iPhone 8 Sentiments Before and After Release")
plt.show()
```

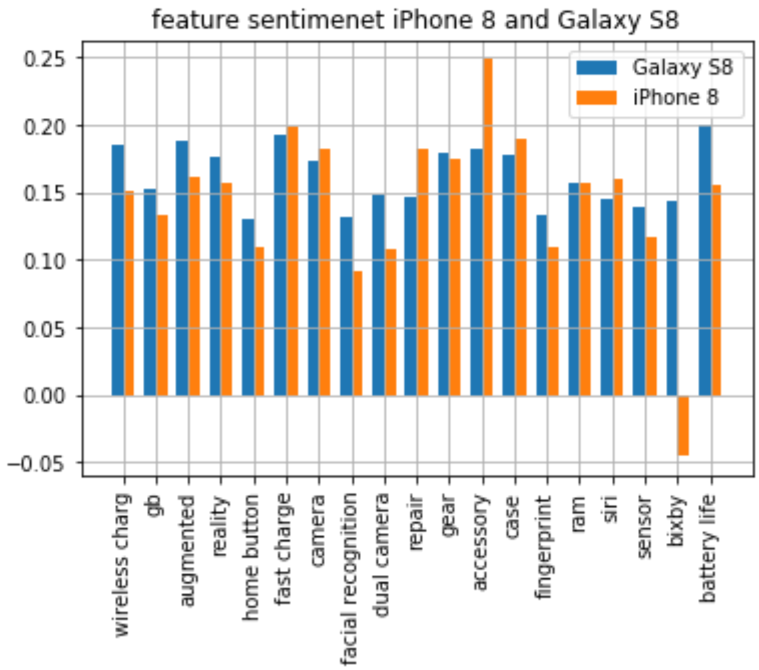


7. Sentiment of Features

In [101]:
```
features = ["wireless charg","gb","augmented", "reality","home button",
"fast charge","camera","facial recognition","dual camera","repair","gea
r","accessory","case","fingerprint","ram","siri","sensor","bixby","batte
ry life"]
```

In [102]:
```
feature_data =[]
for f in features:
    row = []
    for dev in ["Samsung","iPhone 8"]:
        temp_df = all_data[all_data.device_type == dev]
        dev_val = temp_df[temp_df.LowerText.str.contains(f)].TextBlob_se
ntiment.mean()
        row.append(dev_val)
    feature_data.append(row)
feature_data = np.array(feature_data)
```

In [103]:
```
width = .35
ind = np.arange(len(features))
plt.bar(ind,feature_data[:,0],width,label = "Galaxy S8")
plt.bar(ind+width,feature_data[:,1],width,label = "iPhone 8")
plt.xticks(ind+width/2,features,rotation = 90)
plt.title("feature sentimenet iPhone 8 and Galaxy S8")
plt.legend()
plt.grid()
plt.show()
```

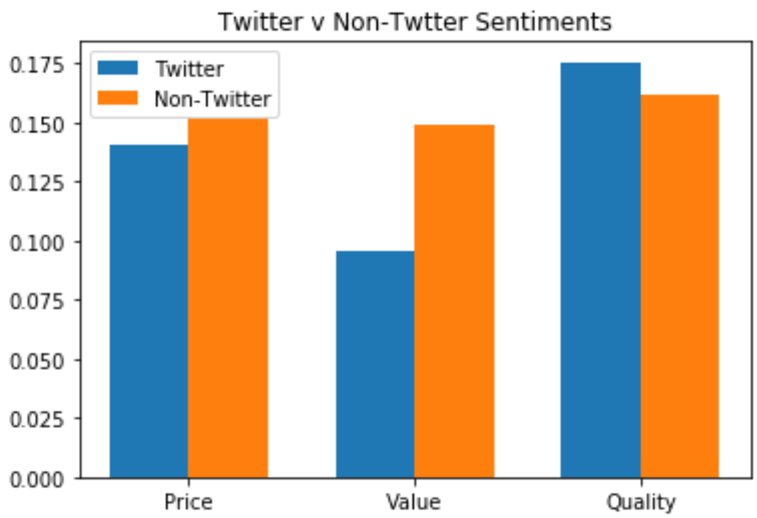


8. Twitter v. Non-Twitter data sources

In [104]:
```
twitter_data = []
features = ["price_text","value_text","quality_text"]
for feat in features:
    sub_data = all_data[all_data["Source Type"] == "Twitter"]
    sub_data = sub_data[sub_data[feat] == True]
    twitter_data.append(sub_data.TextBlob_sentiment.mean())

twitter_data = np.array(twitter_data)
```

In [105]:
```
non_twiter_data = []
features = ["price_text","value_text","quality_text"]
for feat in features:
    sub_data = all_data[all_data["Source Type"] != "Twitter"]
    sub_data = sub_data[sub_data[feat] == True]
    non_twiter_data.append(sub_data.TextBlob_sentiment.mean())
non_twiter_data = np.array(non_twiter_data)
```

In [106]:
```
n = np.array(list(range(3)))
plt.bar(n,twitter_data,.35,label = "Twitter")
plt.bar(n+.35,non_twiter_data,.35,label = "Non-Twitter")
plt.xticks(n+.35/2,["Price","Value","Quality"])
plt.legend()
plt.title("Twitter v Non-Twtter Sentiments")
plt.show()
```

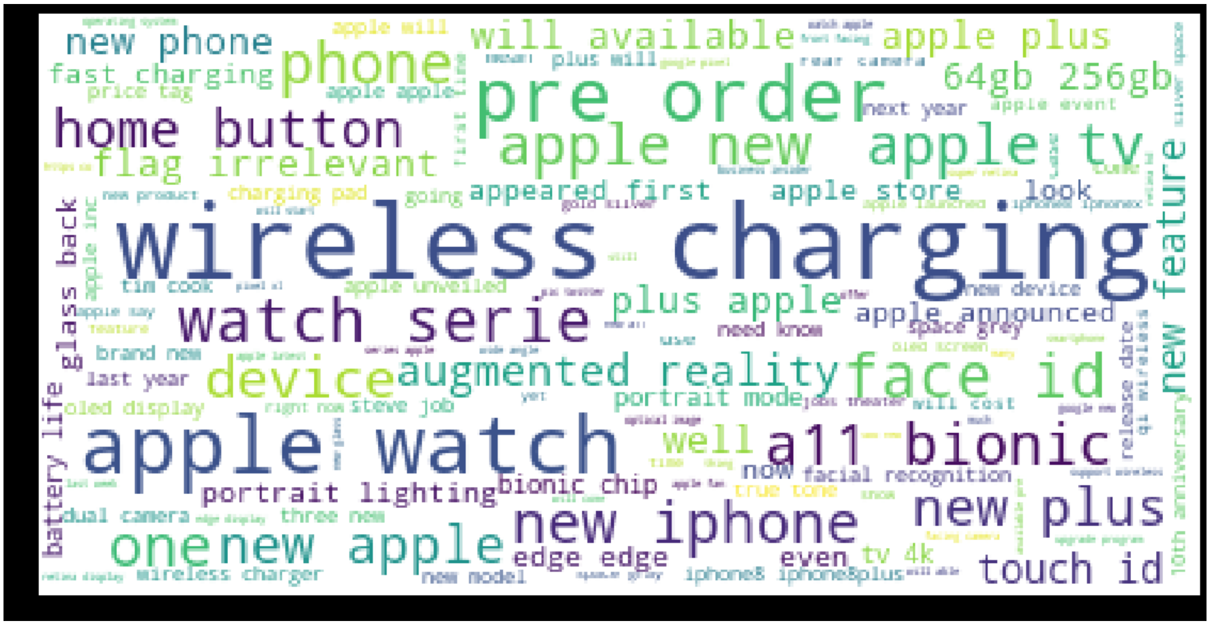


9. Word Cloud Data Exploration

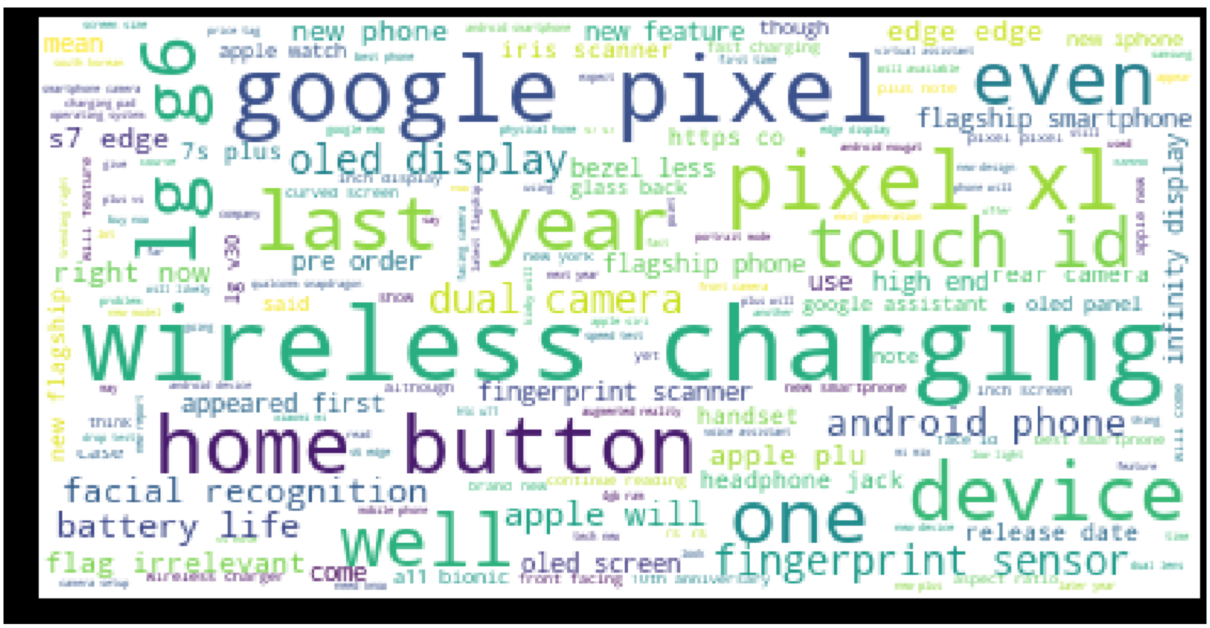
In [107]:
```
from wordcloud import WordCloud, STOPWORDS
stopwords = list(STOPWORDS) +["iphone","galaxy","s8","ifttt","http","if
t","tt","samsung"]
def wordcloud(text,col,stopwords):
    wordcloud = WordCloud(background_color="white",stopwords=stopwords).
generate(" ".join([i for i in text[col]]))
    plt.figure( figsize=(20,10), facecolor='k')
    plt.imshow(wordcloud)
    plt.axis("off")
```

In [108]:
```
for device in all_data.device_type.unique():
    temp = all_data[all_data.device_type == device]
    print(device)
    wordcloud(temp,"LowerText",stopwords)
    plt.show()
```

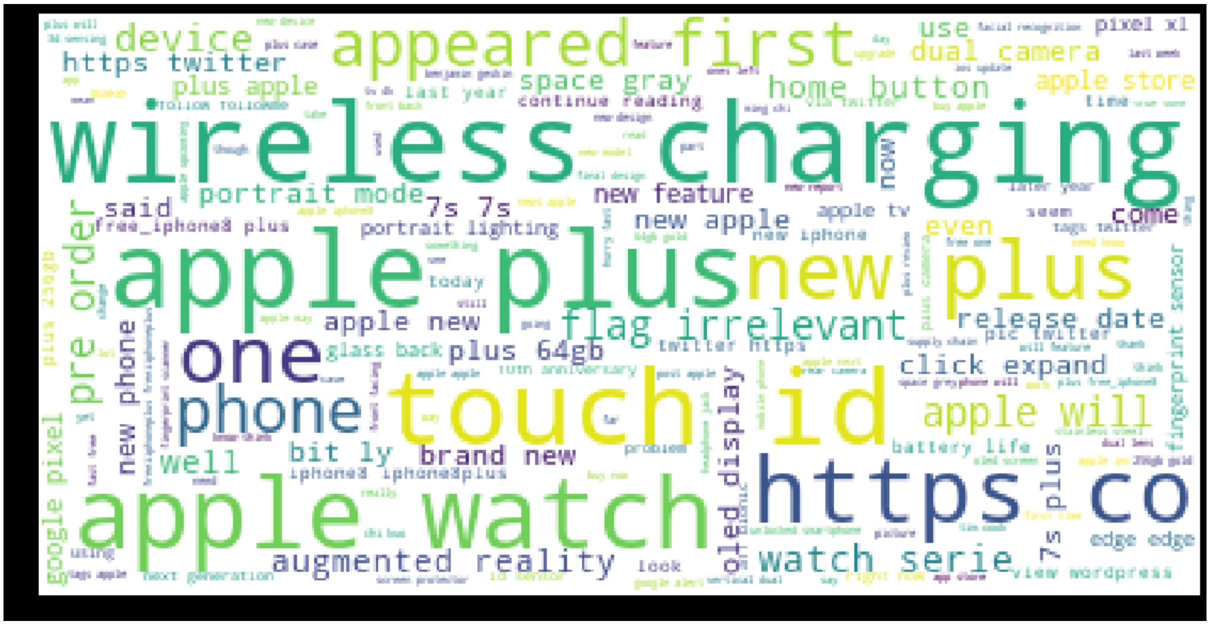
Both iPhone 8 & iPhone X



Both Samsung & iPhone

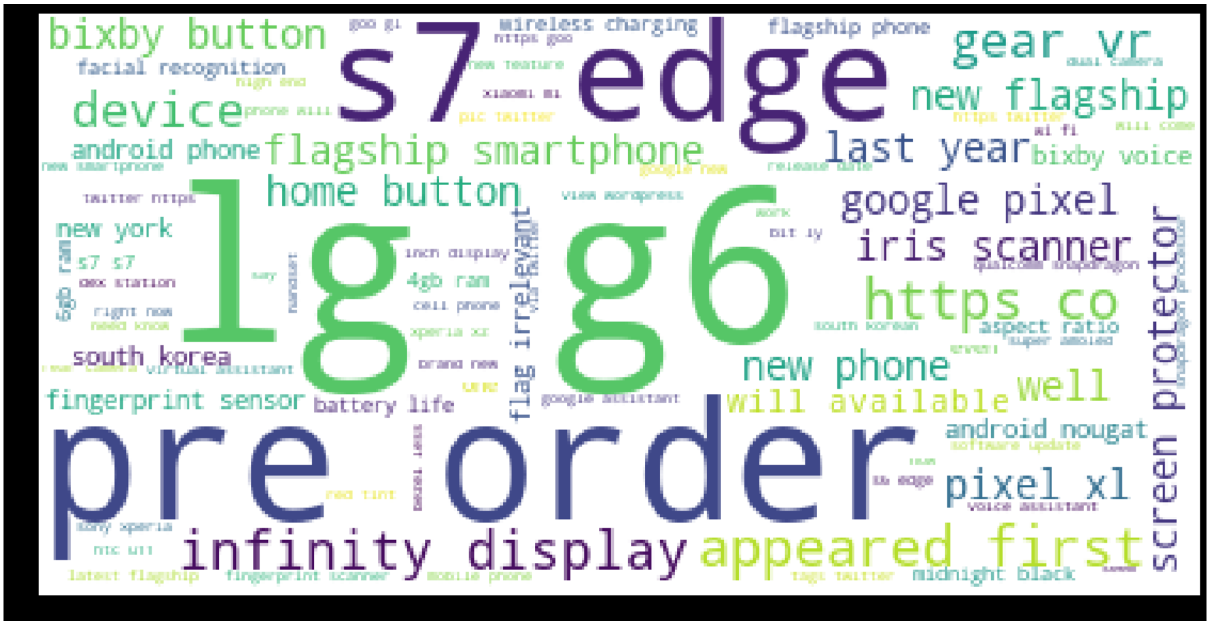


iPhone 8



iPhone X

Samsung

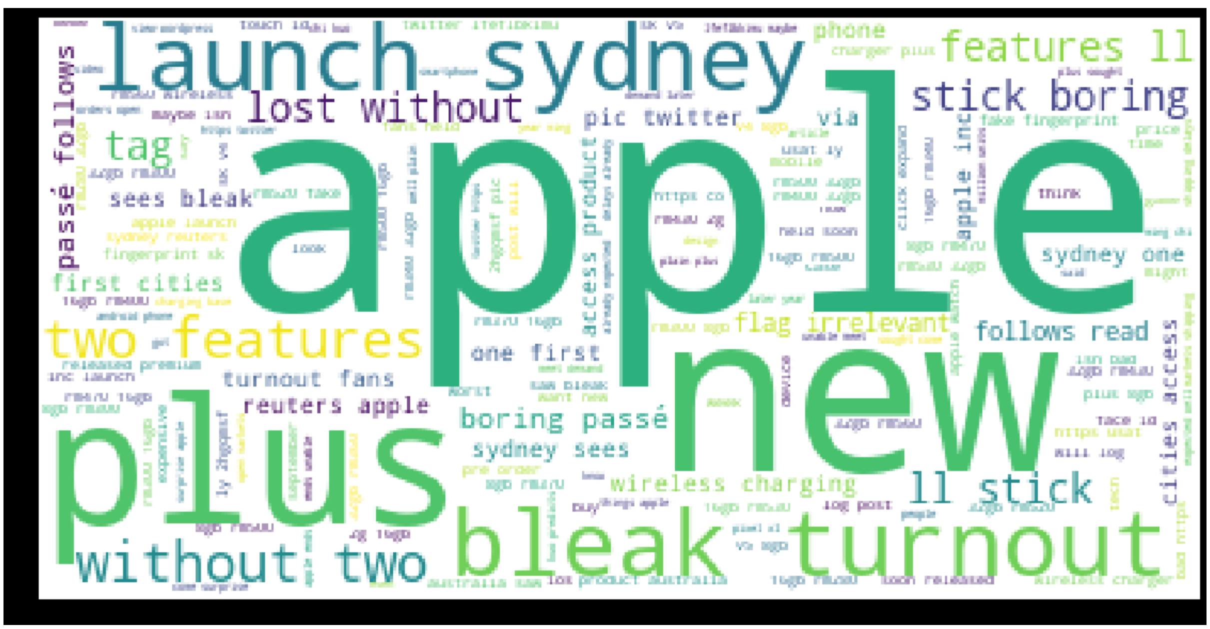


lg g6
pre order
s7 edge
infinity display
appeared first
https co
bixby button
device
flagship smartphone
home button
gear vr
new flagship
last year
google pixel
iris scanner
new phone
will available
well
pixel xl
screen protector
android nougat
fingerprint sensor
south korea
android phone
facial recognition
new york
bixby voice
4gb ram
battery life
aspect ratio
midnight black
wireless charging
flagship phone
flag irrelevant
s7 s7

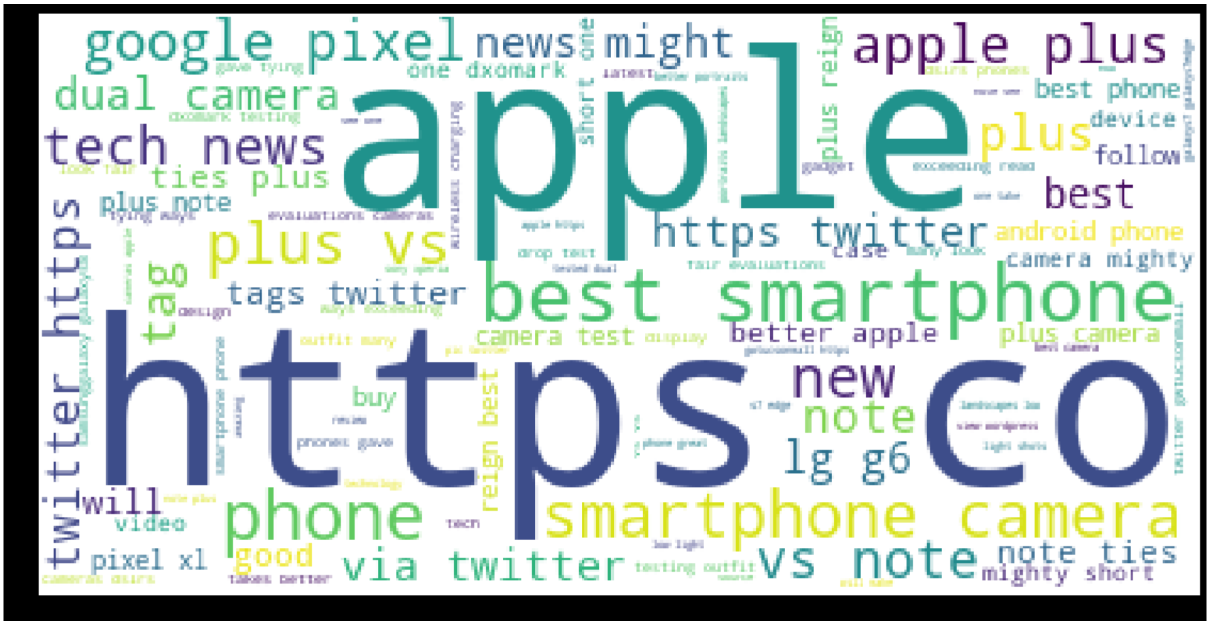
In [110]:
```
#positive and negative for all devices
for device in all_data.device_type.unique():
    temp = all_data[all_data.device_type == device]
    if device != "iPhone X":
        print(device)
        print("Positive")
        positive_temp = temp[temp.TextBlob_sentiment > temp.TextBlob_sen
timent.mean()+ 2*temp.TextBlob_sentiment.std()]
        wordcloud(positive_temp,"LowerText",stopwords)

        plt.show()
        print("Negative")
        neg_temp = temp[temp.TextBlob_sentiment < temp.TextBlob_sentimen
t.mean()- 2*temp.TextBlob_sentiment.std()]
        wordcloud(neg_temp,"LowerText",stopwords)

        plt.show()
```

```
Both iPhone 8 & iPhone X
Positive
```

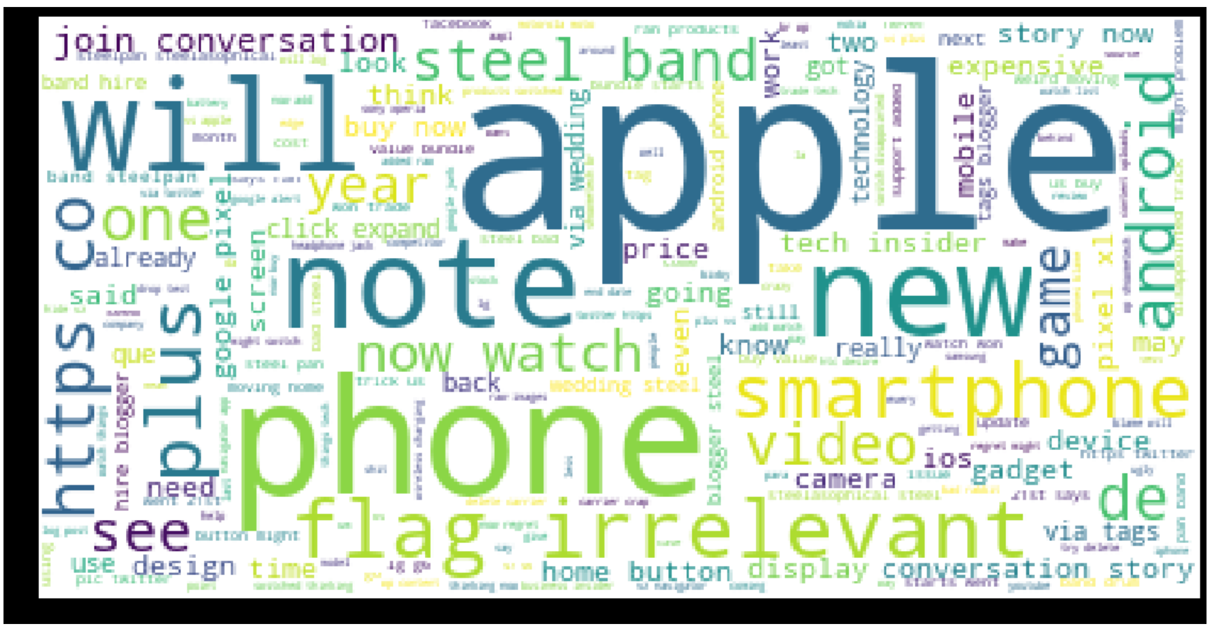

```
Negative
```



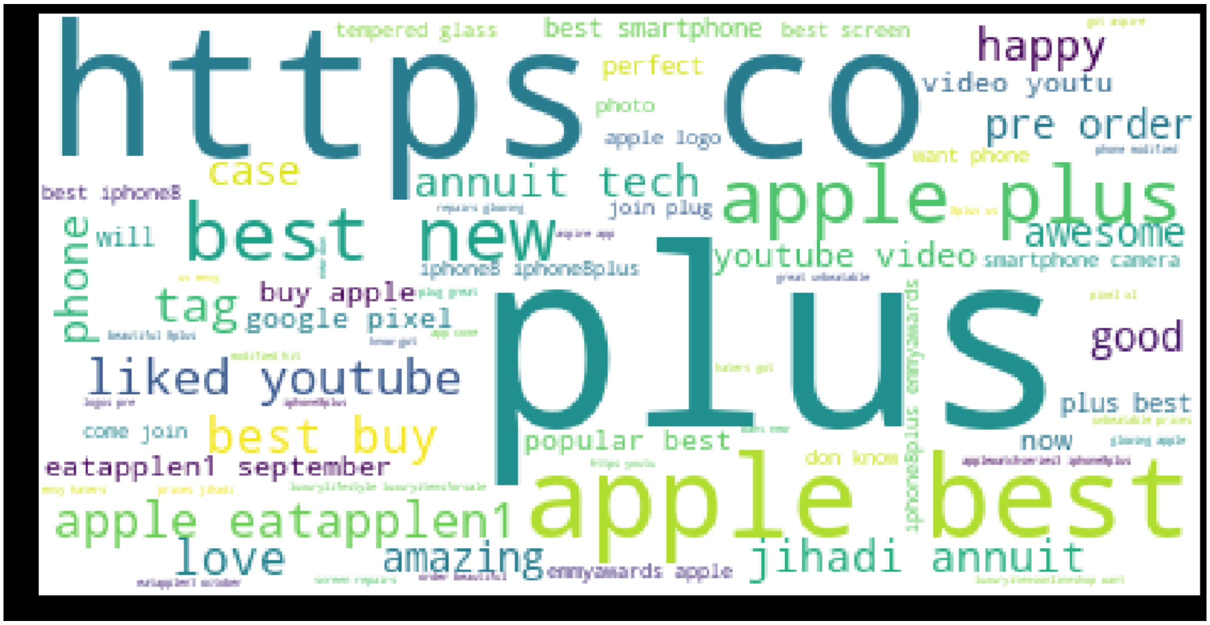

```
Both Samsung & iPhone
Positive
```



Negative



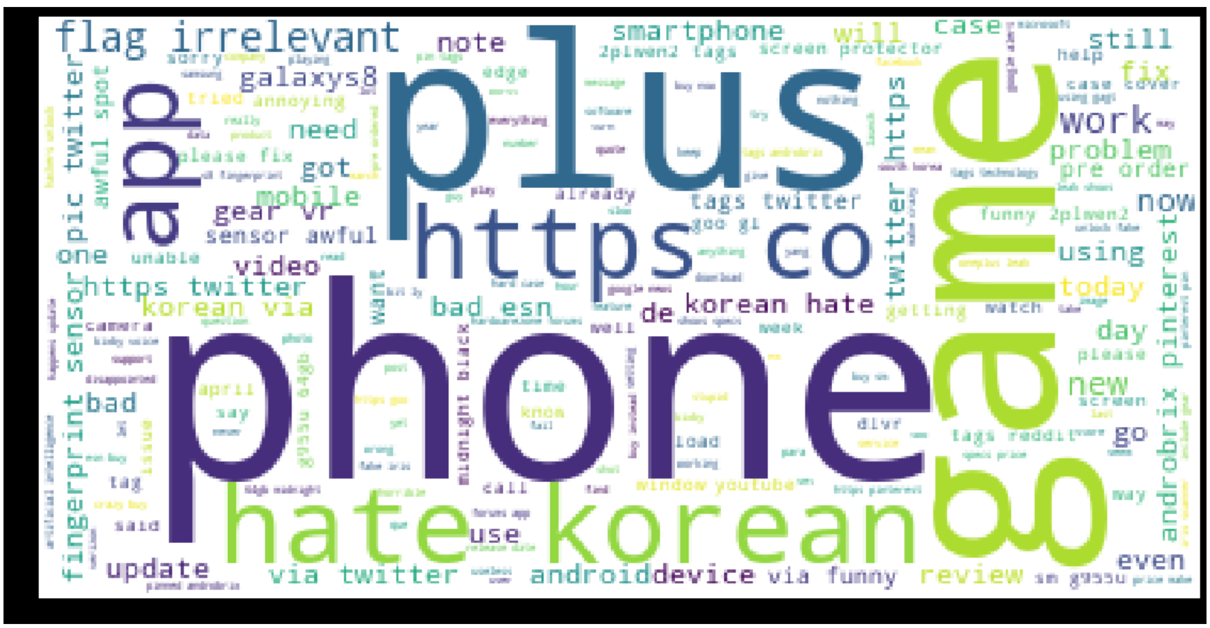
iPhone 8
Positive



Negative

Samsung
Positive

Negative



In [ ]:

In [ ]:

95-851: Social Media Analysis

Data Clan

Part 2: Important Attribute Analysis

1. Load Data

In [0]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import spacy
import seaborn as sns
```

In [0]:
```
import re
import string
import codecs

def makeWordList(path):

    with codecs.open(path, "r", encoding='utf-8', errors='ignore') as f:
        corpus_text = f.read()

    for c in string.punctuation:
        corpus_text = corpus_text.replace(c, "")  # -- (1)

    text = re.sub(r'\S*\d\S*', '', corpus_text) # -- (2)
    text = re.sub(r'[^\w\s]', '', text)         # -- (3)

    text = text.lower().split()           # -- (4)

    li = []
    for token in text:
        li.append(token)

    return " ".join(li)
```

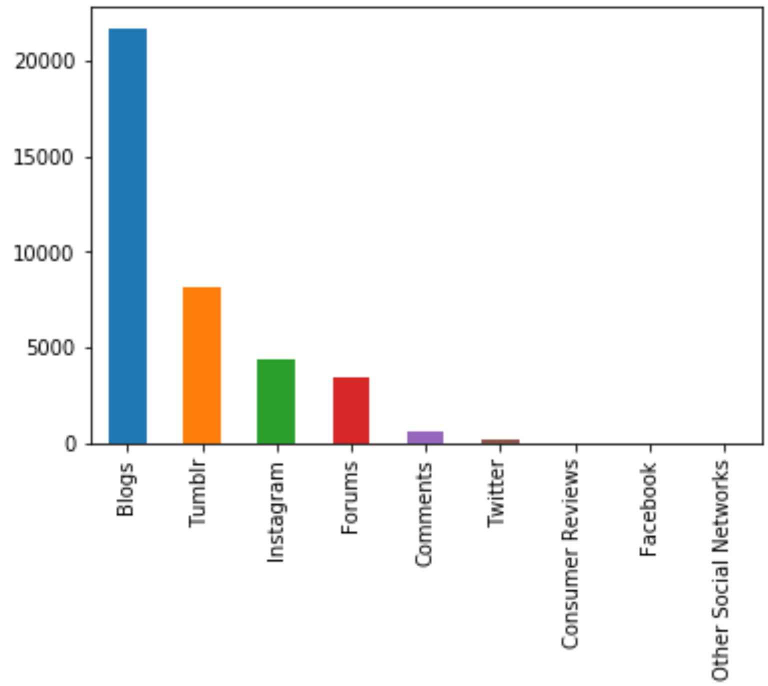
In [0]:
```
# load text corpus
#Both_IPhones_data
df = pd.read_csv('./Cleaned/Both_iPhones_data.csv', encoding='utf-8')
```

2. iPhone Analysis

Sources of iPhone posts:

In [0]:
```
df['Source Type'].value_counts().plot('bar')
```

Out[0]: <matplotlib.axes._subplots.AxesSubplot at 0x7f8af6057128>



As we can see majority of the reviews/posts come from blogs.

In [0]:
```
corpus = list(df['Sound Bite Text'])
```

In [0]:
```
corpus[0]
```

Out[0]:
```
'Following the naming system of the past several years of the iPhone, t
his year's release would be named the iPhone 7s. Talk suggests there wi
ll still be the release of the iPhone 7s in the addition to the newer m
odel. What this will be named – iPhone 8, iPhone 10, iPhone X, somethin
g completely different – we'll just have to wait and see! Lesson of the
Week: Julian Robertson "Our mandate is to find the 200 best companies i
n the world and invest in them, and find the 200 worst companies in the
world and go short on them.'
```

In [0]:
```
len(corpus)
```

Out[0]: 38405

There 38405 reviews/posts on iPhone 8 and iPhone X combined

In [0]:
```
import spacy
nlp = spacy.load('en', disable=['ner', 'parser', 'tagger']) #disabling t
o optimize loading time

def word_tokenizer(doc):
    parsed_doc = nlp(doc)
    return([token.lemma_.lower() for token in parsed_doc if re.match('[a
-zA-Z]+$', token.orth_) and token.lemma_ != '-PRON-'])
```

In [0]:
```
from sklearn.feature_extraction import text
my_stop_words = text.ENGLISH_STOP_WORDS.union(['iphone', 'apple', 'mac',
'ipad', 'plus', 'iphonex', 'applewatch', 'macbook'])
```

In [0]:
```
#constructing a document matrix
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer(min_df=50, stop_words=my_stop_words, max_df
=0.8, tokenizer=word_tokenizer)
doc_matrix = vectorizer.fit_transform(corpus)
print('Number of features in tf-idf:', len(vectorizer.vocabulary_)) # a
 mapping of terms to feature indices
```

```
Number of features in tf-idf: 3324
```

In [0]:
```
doc_matrix = doc_matrix.toarray()
```

In [0]:
```
doc_matrix.shape
```

Out[0]:
```
(38405, 3324)
```

After feature engineering, we have extract 3324 feature values corresponding to 38405 posts/reviews.

In [0]:
```
#constructing feature set
feature_set = pd.DataFrame(doc_matrix)
```

In [0]:
```
feature_set.columns = vectorizer.get_feature_names()
```

In [0]:
```
feature_set.head()
```

Out[0]:

	10	2	3	4	5	a.m.	aapl	abandon	ability	able	...	youtuber	yuan	zaharov
0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
1	0.085010	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
2	0.091318	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
3	0.090216	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0
4	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0

5 rows × 3324 columns

In [0]:
```
num_topics = 10
from sklearn.decomposition import LatentDirichletAllocation
lda = LatentDirichletAllocation(n_components=num_topics, learning_method
='online', max_iter=10, n_jobs=-1 , random_state=0)
lda.fit(doc_matrix)
```

Out[0]:
```
LatentDirichletAllocation(batch_size=128, doc_topic_prior=None,
             evaluate_every=-1, learning_decay=0.7,
             learning_method='online', learning_offset=10.0,
             max_doc_update_iter=100, max_iter=10, mean_change_tol=0.00
1,
             n_components=10, n_jobs=-1, n_topics=None, perp_tol=0.1,
             random_state=0, topic_word_prior=None,
             total_samples=1000000.0, verbose=0)
```

In [0]:
```
print("Log Likelihood: ", lda.score(doc_matrix))
```

```
Log Likelihood:  -1577041.7641789156
```

In [0]:
```
print("Perplexity: ", lda.perplexity(doc_matrix))
```

```
Perplexity:  3110.62261221589
```

In [0]:
```
best_topics = 10
best_lda_model = lda
```

In [0]:
```
# Topic - Word matrix
topic_keywords_matrix = pd.DataFrame(best_lda_model.components_)
topic_keywords_matrix.columns = vectorizer.get_feature_names()
```

In [0]:
```
topic_word_distributions = np.array([topic_word_pseudocounts / np.sum(to
pic_word_pseudocounts)
                                     for topic_word_pseudocounts in best
_lda_model.components_])
```

In [0]:
```
word_columns = vectorizer.get_feature_names()
```

Classify posts based on common themes

In [0]:
```
# Top 20 words per topic
print('Displaying the top %d words per topic and their probabilities wit
hin the topic...' % 20)
print()

for topic_idx in range(best_topics):
    print('[Topic ', topic_idx, ']', sep='')
    sort_indices = np.argsort(topic_word_distributions[topic_idx])[::-1]
    for rank in range(20):
        word_idx = sort_indices[rank]
        print(word_columns[word_idx], ':', topic_word_distributions[topi
c_idx, word_idx])
    print()
```

```
Displaying the top 20 words per topic and their probabilities within th
e topic...

[Topic 0]
gb : 0.08124314795349012
gold : 0.04410283249794682
silver : 0.03711345405706172
iphonex : 0.0285691050166316
grey : 0.02727555025725522
ready : 0.026801837573345554
warranty : 0.021902380389274053
idr : 0.02064325040860555
space : 0.018827639424979174
black : 0.01584397212368294
color : 0.01509612515748844
appletv : 0.013902717245887595
po : 0.013067984098703432
plus : 0.011254466475556044
instagood : 0.010192217994243068
order : 0.010189852782789836
eta : 0.0098416074187699
available : 0.009334004021601942
open : 0.009299183846883606
jualiphone : 0.008682850861908217

[Topic 1]
apple : 0.01815531268165605
new : 0.0086841931537251
plus : 0.008435350915833199
price : 0.008331457723834815
launch : 0.008097981188887608
order : 0.00793602440560132
pre : 0.0067097997776872885
buy : 0.00602034729244769
release : 0.005694689001243799
say : 0.005668815577593309
store : 0.005218897977990837
year : 0.0051913922843324556
model : 0.0050264809302259204
report : 0.004999077140273415
phone : 0.004960685616460343
start : 0.0046675662288371556
watch : 0.004647171762996064
demand : 0.004630651588159105
month : 0.0046294769896978736
cost : 0.00461872773267132

[Topic 2]
apple : 0.011828223378695959
new : 0.010006526162587823
plus : 0.009257686725840368
phone : 0.00669340601564742
camera : 0.006646041024878505
feature : 0.005944818032704595
screen : 0.005280006789421025
make : 0.0046961159768002724
use : 0.004618115398280779
```

```
device : 0.004578096433708312
ios : 0.004558068432185897
display : 0.0043804375700023535
like : 0.004282907722111291
design : 0.00420636343799351
face : 0.004029503978618222
look : 0.0037913617336497673
come : 0.003764873922151073
pixel : 0.003695638816508181
just : 0.00366095659509803
charge : 0.0036340945592757924

[Topic 3]
charge : 0.09471580298408803
wireless : 0.08809183515390136
qi : 0.03385878645396637
pad : 0.03366664704402715
charger : 0.02851052747205663
apple : 0.020380056723358932
powerbyproxi : 0.019116451497411972
support : 0.018801019730501457
ikea : 0.01604313863402815
case : 0.01563248848028313
standard : 0.015162903208354289
include : 0.01332835582450516
new : 0.01116593672738357З
accessory : 0.011103829175289681
belkin : 0.011011322946472989
airpower : 0.01099039025909077
plug : 0.010038895446879493
power : 0.009287162741726637
work : 0.008985304164221561
starbucks : 0.008754326118411045

[Topic 4]
iphonex : 0.08960288972084668
plus : 0.04746456270083673
tag : 0.038212148016760615
apple : 0.033074284759226996
vs : 0.028523997028196503
ifttt : 0.02745385169595188
twitter : 0.019510303088625374
tech : 0.019205887195247066
ios : 0.01815838823625497
macbook : 0.017664221001397994
unboxing : 0.01727397601811946
beat : 0.01686242554196894
applewatch : 0.015469643578818509
buy : 0.015277808763483421
ipad : 0.014220181532959767
video : 0.014144355018765547
technology : 0.01341462433820431
red : 0.013404207616320099
fee : 0.013206740931377792
photo : 0.012973578849661068

[Topic 5]
```

```
charge : 0.044324826135869765
usb : 0.04097267300820534
fast : 0.039901188917655005
c : 0.03329894061261873
charger : 0.02686114602812956
adapter : 0.02356626362105339
cable : 0.01936804869551776
power : 0.01798376715485782
keybanc : 0.016901642051168744
lightning : 0.016680226841080698
support : 0.014710136949003862
minute : 0.014507291653038294
post : 0.014431883955452184
plus : 0.014370831190036158
review : 0.014322574861014894
1 : 0.013507167036357503
apple : 0.01321933361759763
box : 0.012828378503101917
need : 0.012468429764443343
overload : 0.012327821276818402

[Topic 6]
irrelevant : 0.043299476684149324
flag : 0.04311701274629856
google : 0.03838700904319528
alert : 0.030528258504849335
news : 0.026788452297191675
cnet : 0.022953801970168143
ifttt : 0.020759165189906236
business : 0.018242310135475937
mobile : 0.01608912860332735
coverage : 0.015969745398043706
forbes : 0.015771801605714935
tag : 0.015555109796640867
t : 0.015299204936198288
insider : 0.014314094546552858
network : 0.012905734364261397
apple : 0.01275583378990395
lte : 0.01270380479182214
blog : 0.010880641166205143
app : 0.010836862845147132
support : 0.010454659010314707

[Topic 7]
survey : 0.0488375331585272
bio : 0.030879927921308923
link : 0.023593061094276523
decline : 0.02272482765930906
poll : 0.016621392092961748
percent : 0.016245034958426538
life : 0.01618450352169856
pick : 0.016011803060931615
profit : 0.015032429903392586
euro : 0.014788897081045402
guy : 0.014353902788908118
bernstein : 0.013664133065030348
engadget : 0.01352622911061442
```

```
fm : 0.013343328720601282
battery : 0.012951714084298236
reaction : 0.012914364971852582
quick : 0.01284860636083979
want : 0.012751589472656784
leader : 0.012216763718522117
capacity : 0.012203182161546963

[Topic 8]
stock : 0.03585405365791643
finance : 0.0345413882921526
white : 0.027059329899661453
dan : 0.025603750948926175
q : 0.02313041247333707
complicate : 0.020661015560073832
trust : 0.020142604216971095
ini : 0.019370501565060195
quote : 0.01898964986396941
wife : 0.01716157180700678
crackle : 0.016350318296938463
murata : 0.01607455274165567
supplier : 0.01571600155834047
arab : 0.014150862974402172
limit : 0.013656971443163024
anda : 0.012738443695236875
hari : 0.012577921305126055
kali : 0.012394704461044157
jackpot : 0.01219542364602757
doom : 0.011781556661991209

[Topic 9]
lol : 0.03301548070797941
nintendo : 0.023786806892505313
love : 0.0235321469320083
wait : 0.01920062996033466
upsell : 0.018962421686790593
okami : 0.0187734724978438
patent : 0.018483969696744904
pink : 0.01834346037205404
hardwick : 0.018191343816844657
rating : 0.017932316790226475
lesson : 0.01735694558175868
silicon : 0.016374992164142396
story : 0.016045993431963248
valley : 0.01579357132723258
syndicate : 0.015691816928065185
tim : 0.014261446137632526
promo : 0.013799462239500314
viral : 0.013332275540478089
original : 0.012487001474789717
xs : 0.012091667016089382
```

Topic 0 gives us an idea that users are talking about color of the phone here. Top keywords are 'black' 'color', 'gold' color, 'silver' color, 'warranty' and 'space'.

Topic 1 tells us that here users are talking about price, cost etc.

Topic 2 tells us that here users are talking about display, design, look, pixel, screen etc.

Topic 3 tells us that users here are talking about charge/charging, plug, power, wireless

Topic 4 seems irrelevant to our analysis

Topic 5 users are talking about charging/power similar to topic 3.

Topic 6, 7, 8, 9 seems irrelevant.

Experimentation with bigrams (instead of unigrams as per above)

In [0]:
```
from sklearn.feature_extraction.text import CountVectorizer
countvectorizer = CountVectorizer(min_df=50, stop_words=my_stop_words, m
ax_df=0.8, tokenizer=word_tokenizer, ngram_range=(2,2))
count_matrix = countvectorizer.fit_transform(corpus)
print('Number of features in tf-idf:', len(countvectorizer.vocabulary_))
# a mapping of terms to feature indices
```

```
Number of features in tf-idf: 4668
```

In [0]:
```
count_matrix = count_matrix.toarray()
sum_bigrams = count_matrix.sum(axis=0)
words_freq = [(word, sum_bigrams[idx]) for word, idx in countvectorizer.
vocabulary_.items()]
words_freq =sorted(words_freq, key = lambda x: x[1], reverse=True)
```

In [0]:
```
words_freq[:30]
```

Out[0]:
```
[('wireless charge', 10643),
 ('pre order', 6228),
 ('new x', 3829),
 ('watch series', 3683),
 ('new iphones', 3570),
 ('gb gb', 3536),
 ('home button', 2675),
 ('augment reality', 2482),
 ('new phone', 2421),
 ('new feature', 2406),
 ('fast charge', 2285),
 ('flag irrelevant', 2185),
 ('portrait light', 1944),
 ('edge edge', 1775),
 ('appear \ufeff1', 1742),
 ('bionic chip', 1730),
 ('portrait mode', 1699),
 ('facial recognition', 1689),
 ('battery life', 1681),
 ('charge pad', 1678),
 ('steve job', 1613),
 ('new model', 1507),
 ('dual camera', 1455),
 ('x come', 1422),
 ('wireless charger', 1371),
 ('usb c', 1365),
 ('x x', 1335),
 ('tim cook', 1290),
 ('oled display', 1278),
 ('brand new', 1278)]
```

Most important attributes for iPhone:

The 30 most frequently occuring bigrams in the reviews/post based on the above analysis. This gives us a hint on what features/characteristics of the phone are being discussed by the users the most. Some of the most talked about attributes based on our analysis above were:

1. Wireless charging
2. Home button (Apple removed home button from iPhone X, so this generated a good amount of discussion among users)
3. Fast Charge
4. Portrait lighting (New feature introduced in iPhone)
5. Bionic Chip (Apple introduced this new feature iPhone 8 onwards)
6. Facial recognition (Apple introduced this new feature iPhone 8 onwards)
7. Charge Pad (Apple introduced this new feature iPhone 8 onwards)
8. Dual Camera
9. Oled Display (New OLED screen for iPhone X)

Perform bigram topic modeling

In [0]:
```
#constructing a document matrix
vectorizer2 = TfidfVectorizer(min_df=50, stop_words=my_stop_words, max_d
f=0.8, tokenizer=word_tokenizer, ngram_range=(2,2))
doc_matrix2 = vectorizer2.fit_transform(corpus)
print('Number of features in tf-idf:', len(vectorizer2.vocabulary_)) # a
mapping of terms to feature indices
```

```
Number of features in tf-idf: 4668
```

In [0]:
```
doc_matrix2 = doc_matrix2.toarray()
```

In [0]:
```
doc_matrix2.shape
```

Out[0]:
```
(38405, 4668)
```

After feature engineering, we have extract 3324 feature values corresponding to 38405 posts/reviews.

In [0]:
```
#constructing feature set
feature_set2 = pd.DataFrame(doc_matrix2)
```

In [0]:
```
feature_set2.columns = vectorizer2.get_feature_names()
```

In [0]:
```
feature_set2.head()
```

Out[0]:

	10 anniversary	2 generation	2 new	3 generation	3 party	3 quarter	4 quarter	aapl stock	able charge	able make	...	r
0	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	
1	0.133166	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	
2	0.136158	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	
3	0.136158	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	
4	0.000000	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	

5 rows × 4668 columns

In [0]:
```
num_topics = 10
from sklearn.decomposition import LatentDirichletAllocation
lda2 = LatentDirichletAllocation(n_components=num_topics, learning_metho
d='online', max_iter=10, n_jobs=-1 , random_state=0)
lda2.fit(doc_matrix2)
```

Out[0]:
```
LatentDirichletAllocation(batch_size=128, doc_topic_prior=None,
             evaluate_every=-1, learning_decay=0.7,
             learning_method='online', learning_offset=10.0,
             max_doc_update_iter=100, max_iter=10, mean_change_tol=0.00
1,
             n_components=10, n_jobs=-1, n_topics=None, perp_tol=0.1,
             random_state=0, topic_word_prior=None,
             total_samples=1000000.0, verbose=0)
```

In [0]:
```
print("Log Likelihood: ", lda2.score(doc_matrix2))
```

```
Log Likelihood:  -1118189.9989448085
```

In [0]:
```
print("Perplexity: ", lda2.perplexity(doc_matrix2))
```

```
Perplexity:  8249.049629477855
```

In [0]:
```
# Select the best hyperparameters
best_topics2 = 10
best_lda_model2 = lda2
```

In [0]:
```
# Topic - Word matrix
topic_keywords_matrix2 = pd.DataFrame(best_lda_model2.components_)
topic_keywords_matrix2.columns = vectorizer2.get_feature_names()
```

In [0]:
```
topic_word_distributions2 = np.array([topic_word_pseudocounts / np.sum(t
opic_word_pseudocounts)
                                      for topic_word_pseudocounts in best
_lda_model2.components_])
```

In [0]:
```
word_columns2 = vectorizer2.get_feature_names()
```

In [0]:
```
# Top 20 words per topic
print('Displaying the top %d bigrams per topic and their probabilities w
ithin the topic...' % 20)
print()

for topic_idx in range(best_topics):
    print('[Topic ', topic_idx, ']', sep='')
    sort_indices2 = np.argsort(topic_word_distributions2[topic_idx])[::-
1]
    for rank in range(20):
        word_idx = sort_indices2[rank]
        print(word_columns2[word_idx], ':', topic_word_distributions2[to
pic_idx, word_idx])
    print()
```

```
Displaying the top 20 bigrams per topic and their probabilities within
the topic...

[Topic 0]
wireless charge : 0.01777715387585753
fast charge : 0.005707561620969423
home button : 0.005619200957530918
new x : 0.005398372501192109
new iphones : 0.00480258993425143
new feature : 0.004261652655317761
watch series : 0.004260063519744209
battery life : 0.004204906541827839
charge pad : 0.004000343630079443
bionic chip : 0.0037819184737335715
oled display : 0.0037052513743764666
facial recognition : 0.0033390255074162135
edge edge : 0.003289685871626061
dual camera : 0.003283278056528878
usb c : 0.003269607419867699
qi wireless : 0.0031986219364717925
new phone : 0.0031846874940162266
x screen : 0.0031141966471887146
x new : 0.0030215366906442506
augment reality : 0.002967439346028165

[Topic 1]
gb gb : 0.03762385944149371
t mobile : 0.01874271496916777
space grey : 0.01840246092550391
click expand : 0.01592886398264929
reason buy : 0.015287086148785026
press release : 0.012144052130992962
x tag : 0.01186034899489786
make sense : 0.011428131604306014
gold silver : 0.011185669310730267
gb x : 0.009612922557250797
buy instead : 0.009541162214916148
gb idr : 0.009488791872972623
order x : 0.009434609995557806
x gb : 0.00930293733367522
grey gb : 0.008380105860010175
early month : 0.008249423145602989
silver black : 0.008136187753191946
release blog : 0.00795947806393744
support new : 0.007949153367450409
x compare : 0.007947430379486054

[Topic 2]
view wordpress : 0.01516915329326385
wordpress tag : 0.011142057294053076
x cost : 0.011084407682524864
repair cost : 0.010461507821954702
steve job : 0.010063961765172498
story watch : 0.009969386694481205
conversation story : 0.009621811321003268
join conversation : 0.009603545108181538
x important : 0.009017556228525967
```

```
screen repair : 0.00891184053788994
x device : 0.007833158801703564
watch announce : 0.0077626973491623135
plan buy : 0.007459852475847686
cost repair : 0.007398801985339411
tech insider : 0.007206758638818512
like yes : 0.006364975277894009
job theater : 0.006302653246049779
social medium : 0.006039151765313626
double wireless : 0.005870391854396006
tech gadget : 0.005470332624200753

[Topic 3]
pre order : 0.0367942741361525
x vs : 0.018229173819462798
x pre : 0.013428087963948553
buy x : 0.012697034576084428
x launch : 0.010380085337237341
launch day : 0.008403785174313002
vs x : 0.008359200815876662
release date : 0.007577479837023744
include x : 0.007470432795633398
ios device : 0.007412353098381237
x available : 0.006939876196166203
brand new : 0.006884312128899353
appear 1 : 0.0067936578660800365
red picture : 0.0064154740272507815
ahead x : 0.006104483037642014
people wait : 0.006067422151315750
new x : 0.006065973307130645
available pre : 0.005905824716179976
x production : 0.005744568440631799
new ios : 0.005712308997467998

[Topic 4]
watch series : 0.009976390732677577
pre order : 0.008424344894718007
pixel xl : 0.00796612751089979
launch x : 0.006743084886250254
x watch : 0.006360760923148467
x price : 0.006017138045293169
announce new : 0.005973609805081 61
release x : 0.005720383445273442
new x : 0.005476768675273 89
phone x : 0.005243327627414728
x release : 0.005082340676331869
gb model : 0.005066371289838914
x x : 0.004870949980700288
wait x : 0.004864324170263605
wireless charger : 0.004431993943319008
tag ifttt : 0.004392754196807779
new model : 0.004354409017870886
announce x : 0.004344413471617086
unveil new : 0.004324292015832824
upcoming x : 0.004256955761630575

[Topic 5]
```

```
wait x : 0.022631219614314073
google news : 0.022206278821022774
tag ifttt : 0.020434267608786565
technology google : 0.02042808135488185
google pixel : 0.017994727620403272
ifttt tag : 0.01598691155204227
fee ifttt : 0.012294987609380881
news article : 0.012094114338428674
x look : 0.010901292967946811
coverage technology : 0.010813969147249313
tear x : 0.010505070051334308
upgrade x : 0.009685056333560317
force restart : 0.009138265139555348
compare x : 0.008793762135525773
x follow : 0.008716006885858196
article technology : 0.00823009142113509
wi fi : 0.008119325045843367
late iphones : 0.00793082474590451
article forum : 0.007159186203082063
x vs : 0.007005076857336505

[Topic 6]
new x : 0.01845351207275172
x early : 0.014661865895045128
x tech : 0.012515804356585222
case x : 0.010324525609612646
tech news : 0.01027652703416699
read original : 0.009713569900372494
x read : 0.009384985023621178
x begin : 0.009056008483964591
original post : 0.008344906105505649
want new : 0.008298202891983481
await x : 0.008207057398361929
x actually : 0.008069475796140722
x announcement : 0.008034323178426266
long await : 0.007749218447140597
launch month : 0.007376135311276976
thing x : 0.007237513006818029
make sure : 0.007152039470720113
x ready : 0.007126510222274577
announcement x : 0.006864795729980973
swell battery : 0.006629691124221259

[Topic 7]
tag x : 0.018665099637521994
tech technology : 0.014550602618738065
want x : 0.012767396766025646
airpods appletv : 0.010861281545403903
appletv red : 0.010556565149701215
beat photo : 0.010556565149680845
picture beat : 0.010556565149680642
ios airpods : 0.010556565149671031
publish 1 : 0.008542896231318486
x event : 0.008444574976048287
fan hold : 0.008221432928863242
information overload : 0.007783837074654198
overload news : 0.007783837074654198
```

```
x think : 0.006938942437503515
smartphone x : 0.006747806933323833
tag tech : 0.006542870096404709
rss fee : 0.006514280613442105
protective case : 0.006307453756539623
buy wait : 0.006206323485942185 5
receive email : 0.005926756019365995

[Topic 8]
flag irrelevant : 0.030978940699707518
augment reality : 0.011661082738129349
tim cook : 0.01136735869853837
link bio : 0.010229944460782863
ming chi : 0.009728043494038647
chi kuo : 0.00970624856372086
supply chain : 0.008969505134708233
demand x : 0.00863925299938879
post x : 0.007290697926333248 5
cut production : 0.007203029888510839
analyst ming : 0.0070870885803032
x announce : 0.006864669969482181 5
kgi security : 0.006765120904902963
million unit : 0.006365353063468234
video x : 0.006239646971532049
capital market : 0.006202412541195928
coverage flag : 0.006097327642347377
x edition : 0.005934916866748179 5
x unit : 0.005693431110674264
high price : 0.005580698483380989

[Topic 9]
portrait light : 0.02052770099916783
x case : 0.019758721601113926
gb ram : 0.013363843453209106
continue read : 0.009734371336432157
low light : 0.009350199449015556
x come : 0.008806243167385213
case available : 0.008392627566891964
like x : 0.00813573026850883 7
car support : 0.007884930348044307
light effect : 0.007086515472157515
phone like : 0.007041962666465463
new video : 0.006985850627826908
portrait mode : 0.006970989617306466
x include : 0.006427039202337871
new pixel : 0.006342096641668364 5
screen protector : 0.006308424161221779
allow user : 0.00605734480695968
temper glass : 0.005966556480412648
light mode : 0.005964998936192403
power button : 0.005724769985985333
```

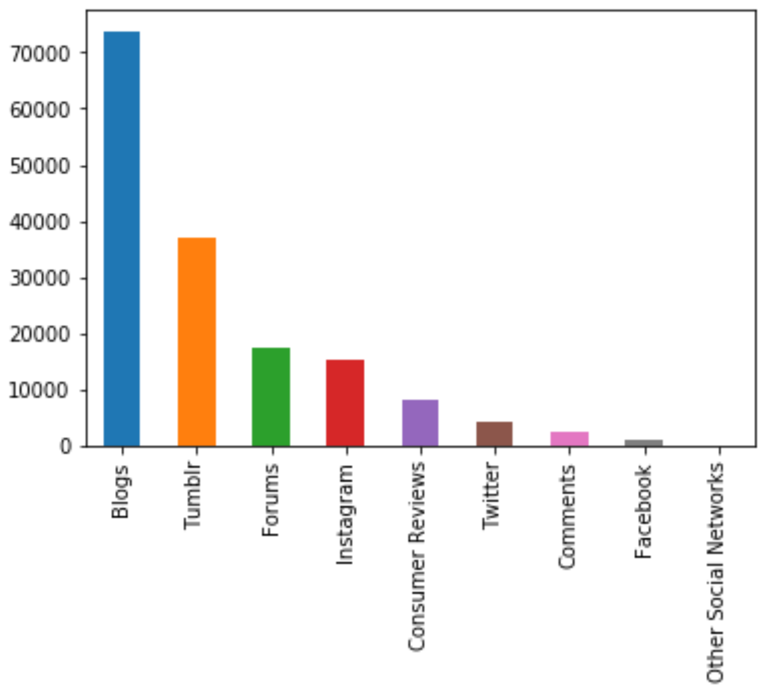
2. Galaxy S8 Analysis

In [0]:
```
# load text corpus
# samsung_data
df = pd.read_csv('./Cleaned/samsung_data.csv', encoding='utf-8')
```

Sources of Galaxy S8 posts:

In [0]:
```
df['Source Type'].value_counts().plot('bar')
```

Out[0]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x7f8aa1d57fd0>
```



As we can see majority of the reviews/posts come from blogs.

In [0]:
```
corpus = list(df['Sound Bite Text'])
```

In [0]:
```
corpus[0]
```

Out[0]:
```
'Samsung Galaxy S8 G950U G950P Sprint unlock by usb cable On Cell Phone
Forums Need to unlock a S8 Sprint ? Find here an exclusive Sprint Samsu
ng Galaxy S8 G950U G950P unlock method, available for this phone model
that cannot be unlocked by unlock codes. This Samsung Galaxy S8 unlock
service is the only option to unlock this phone model and make it work
both in the USA and abroad on any gsm network. The SM-G950U G950P unloc
k is done online by USB cable, not by codes calculated from IMEI. ... S
upported models - Sprint Samsung Galaxy S8 G950U G950P HOW THE SAMSUNG
GALAXY S8 UNLOCK IS DONE? Important: We are available according to our
Online program , there is shown if are available at a certain time. Aft
er buying this Sprint Samsung s8 unlock service, you will receive in a
few minutes the instructions about how to prepare your computer and pho
ne before the online unlock (some simple steps in our video tutorial, t
akes less than 5 min).'
```

In [0]:
```
len(corpus)
```

Out[0]: 159419

There 159419 reviews/posts on Samsung

In [0]:
```
import spacy
nlp = spacy.load('en', disable=['ner', 'parser', 'tagger']) #disabling t
o optimize loading time

def word_tokenizer(doc):
    parsed_doc = nlp(doc)
    return([token.lemma_.lower() for token in parsed_doc if re.match('[a
-zA-Z]+$', token.orth_) and token.lemma_ != '-PRON-'])
```

In [0]:
```
from sklearn.feature_extraction import text
my_stop_words = text.ENGLISH_STOP_WORDS.union(['android', 'samsung', 'ga
laxy'])
```

In [0]:
```
#constructing a document matrix
from sklearn.feature_extraction.text import TfidfVectorizer
vectorizer = TfidfVectorizer(min_df=50, stop_words=my_stop_words, max_df
=0.8, tokenizer=word_tokenizer, max_features=10000)
doc_matrix = vectorizer.fit_transform(corpus)
print('Number of features in tf-idf:', len(vectorizer.vocabulary_)) # a
 mapping of terms to feature indices
```

```
Number of features in tf-idf: 5893
```

In [0]:
```
doc_matrix = doc_matrix.toarray()
```

In [0]:
```
doc_matrix.shape
```

Out[0]: (159419, 5893)

After feature engineering, we have extract 3324 feature values corresponding to 38405 posts/reviews.

In [0]:
```
#constructing feature set
feature_set = pd.DataFrame(doc_matrix)
```

In [0]:
```
feature_set.columns = vectorizer.get_feature_names()
```

In [0]:
```
feature_set.head()
```

Out[0]:

	2	3	4	5	8	a.m.	abandon	abhijeet	ability	able	...	zenfone	zenpad	zero	zip
0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0
1	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0

5 rows × 5893 columns

In [0]:
```
num_topics = 10
from sklearn.decomposition import LatentDirichletAllocation
lda = LatentDirichletAllocation(n_components=num_topics, learning_method
='online', max_iter=10, n_jobs=-1 , random_state=0)
lda.fit(doc_matrix)
```

Out[0]:
```
LatentDirichletAllocation(batch_size=128, doc_topic_prior=None,
             evaluate_every=-1, learning_decay=0.7,
             learning_method='online', learning_offset=10.0,
             max_doc_update_iter=100, max_iter=10, mean_change_tol=0.00
1,
             n_components=10, n_jobs=-1, n_topics=None, perp_tol=0.1,
             random_state=0, topic_word_prior=None,
             total_samples=1000000.0, verbose=0)
```

In [0]:
```
print("Log Likelihood: ", lda.score(doc_matrix))
```

```
Log Likelihood:  -5798516.057844428
```

In [0]:
```
print("Perplexity: ", lda.perplexity(doc_matrix))
```

```
Perplexity:  4538.315328002736
```

In [0]:
```
# Select the best hyperparameters
best_topics = 10
best_lda_model = lda
```

In [0]:
```
# Topic - Word matrix
topic_keywords_matrix = pd.DataFrame(best_lda_model.components_)
topic_keywords_matrix.columns = vectorizer.get_feature_names()
```

In [0]:
```
topic_word_distributions = np.array([topic_word_pseudocounts / np.sum(to
pic_word_pseudocounts)
                                     for topic_word_pseudocounts in best
_lda_model.components_])
```

In [0]:
```
word_columns = vectorizer.get_feature_names()
```

Classify posts based on common themes

In [0]:
```
# Top 20 words per topic
print('Displaying the top %d words per topic and their probabilities wit
hin the topic...' % 20)
print()

for topic_idx in range(best_topics):
    print('[Topic ', topic_idx, ']', sep='')
    sort_indices = np.argsort(topic_word_distributions[topic_idx])[::-1]
    for rank in range(20):
        word_idx = sort_indices[rank]
        print(word_columns[word_idx], ':', topic_word_distributions[topi
c_idx, word_idx])
    print()
```

```
Displaying the top 20 words per topic and their probabilities within th
e topic...

[Topic 0]
order : 0.025629200779942435
pre : 0.023921682668846124
red : 0.0200222946339203
update : 0.019982828284417258
tint : 0.017059332137557472
mobile : 0.016332604114190775
fix : 0.014775292640924982
issue : 0.014586623602296258
t : 0.014016315097136106
april : 0.012957829245554362
unlock : 0.011438362430679635
carrier : 0.011053283721860265
available : 0.010391017561181994
verizon : 0.00990162520947982
preorder : 0.009302892980303549
start : 0.009047962760063819
plus : 0.008824077481106484
free : 0.007711189121191785
buy : 0.007628000092502139
customer : 0.007606230136410973

[Topic 1]
dex : 0.05082160971296733
desktop : 0.030507896922510797
dock : 0.028539222046489263
cnet : 0.026972042363368698
send : 0.024042807306729905
icon : 0.023636572949141302
launcher : 0.022831299760820677
use : 0.022076469307822748
station : 0.02028452643008502
gagt : 0.01979355103667749
pc : 0.012929465571382692
theme : 0.012714763966309647
pic : 0.011716710352179905
oneplus : 0.011575105289503715
blow : 0.01039817042052308
keyboard : 0.009945257780014235
linux : 0.009722364676153268
exo : 0.009395714416167291
monitor : 0.009296862889490732
snapchat : 0.008726528030189203

[Topic 2]
twitter : 0.09948925415844262
tag : 0.0735915278124991
ifttt : 0.07187079477700534
rt : 0.031373806155106945
tech : 0.02957588264737773
trend : 0.02553501737342278
canada : 0.023853412077789835
wallpaper : 0.023646846821076786
central : 0.01639424999864219
```

```
sky : 0.01453942024849919
authority : 0.013805131854674269
pocketnow : 0.013249517708734315
tweet : 0.011512189573575934
androidauth : 0.011374590301579997
phandroid : 0.010928041369318013
facebookpages : 0.010259588275808269
sunset : 0.008963168392543094
review : 0.008735166248242431
literally : 0.007919459969613046
fact : 0.007045931190551231

[Topic 3]
sm : 0.047780676897771344
gb : 0.04319419927614894
black : 0.04154046246260742
unlock : 0.03313379170419474
midnight : 0.030762633924182325
buy : 0.021442947560195318
new : 0.0185181812707691
hot : 0.01828007736677942
cell : 0.017397962091330457
smartphone : 0.016939770356931796
gray : 0.015954727478019994
plus : 0.015169571043869342
usa : 0.013381793164040492
gold : 0.012228758265259288
hardwarezone : 0.011923305911720295
silver : 0.011901509917271411
factory : 0.01106312166319571
verizon : 0.01065469502581019
blue : 0.01029271379056016
edition : 0.00951095237791331

[Topic 4]
gb : 0.03748659205415564
ram : 0.021726597125692092
protector : 0.017953590075589426
nokia : 0.01718515443263322
camera : 0.014914665052254007
snapdragon : 0.01461055814721028
dual : 0.01331194176359344l
leak : 0.012460963015819091
active : 0.012069960365180299
oneplus : 0.012042996803531601
storage : 0.011899592616774386
price : 0.009806549826336946
glass : 0.009771624394529214
processor : 0.009660444310400623
sim : 0.009451270598235676
screen : 0.00926240709046109
exynos : 0.008959732435901518
qualcomm : 0.008744810961050797
variant : 0.008638023488809774
core : 0.008193271483404041

[Topic 5]
```

```
vs : 0.03732839682019339
tag : 0.03724716307955657
plus : 0.03677043111377488
review : 0.027966582731792967
samsunggalaxy : 0.02790934906672396
video : 0.025052012817627683
tech : 0.024406775087481437
news : 0.02372713099100591
google : 0.02100027504386046
unboxing : 0.0206372677433747
unboxyourphone : 0.020268909665089384
flag : 0.019463355696413682
lg : 0.01865277569126849
irrelevant : 0.01865003478663751
youtube : 0.018225979640022347
gadget : 0.017918126872637528
smartphone : 0.016783183650173766
comparison : 0.01659543457594826
crash : 0.016574982580934348
phone : 0.01580066723909783

[Topic 6]
case : 0.06916029000863662
charge : 0.020825427868707277
plus : 0.01983616639631788
cover : 0.019186817704493882
work : 0.01864295342132865
charger : 0.015555918883688544
fit : 0.012587637551242429
cable : 0.010544185366986931
protection : 0.010452880336884114
wireless : 0.010160425267176277
phone : 0.009975415122745981
great : 0.009903877282459634
drop : 0.009891423120508697
leather : 0.009634579866624528
spigen : 0.009528764837933333
fast : 0.009238725967594189
protect : 0.008935015180624438
orchid : 0.008620234042130723
good : 0.008144529089332716
tpu : 0.007981544949822865

[Topic 7]
phone : 0.009515392409287065
new : 0.00859758154970041
display : 0.005731316122863418
launch : 0.005715067536826397
screen : 0.00550505426784757
device : 0.005393550226497029
bixby : 0.005326982797831303
note : 0.005305034546894121
feature : 0.0049290264482895465
1 : 0.004754393049320687
flagship : 0.0046087435141167555
make : 0.004385665366805055
smartphone : 0.004299481713947154
```

```
like : 0.0042276195988919015
come : 0.004179053585469187
look : 0.004160609016774414
year : 0.00408333136872767
camera : 0.0039859590708253515
just : 0.003921727729671118
good : 0.0038409114607552727

[Topic 8]
love : 0.02230886253618591
otterbox : 0.01935389154278483
amaze : 0.018622363568244083
photography : 0.018495260591904313
shoot : 0.018322635379162056
photo : 0.016716859196319964
instagood : 0.013504672947735526
spring : 0.012753208340096134
recovery : 0.01221458020534909
photooftheday : 0.011328562795161048
nature : 0.011325147818411752
tutorial : 0.01129770992154491
cute : 0.011131808759521886
beautiful : 0.010392023995140715
tag : 0.010240277712328112
samsunggalaxy : 0.009447937630440122
picture : 0.00929289084685844
lol : 0.009273849541039868
happy : 0.00878375647831126
nofilter : 0.008709405205897575

[Topic 9]
app : 0.014934176382772875
use : 0.009283640279277566
button : 0.009096391222315255
work : 0.007235155303009214
bixby : 0.006658602120881599
phone : 0.006391407875892996
download : 0.0055541712016843985
user : 0.005217064341355947
play : 0.00512353173012694
google : 0.004778909594207152
update : 0.004741245260947795
iris : 0.004574852281995002
new : 0.004500541659447313
try : 0.004368627323125851
music : 0.0043291356212816715
recognition : 0.004311893852065395
game : 0.004291029451961403
scanner : 0.004032503766164054
microsoft : 0.003947638253331034
message : 0.003914852082678205
```

Experimentation with bigrams (instead of unigrams as per above)

In [0]:
```
from sklearn.feature_extraction.text import CountVectorizer
countvectorizer = CountVectorizer(min_df=50, stop_words=my_stop_words, m
ax_df=0.8, tokenizer=word_tokenizer, ngram_range=(2,2), max_features=100
00)
count_matrix = countvectorizer.fit_transform(corpus)
print('Number of features in tf-idf:', len(countvectorizer.vocabulary_))
# a mapping of terms to feature indices
```

```
Number of features in tf-idf: 10000
```

In [0]:
```
count_matrix = count_matrix.toarray()
sum_bigrams = count_matrix.sum(axis=0)
words_freq = [(word, sum_bigrams[idx]) for word, idx in countvectorizer.
vocabulary_.items()]
words_freq =sorted(words_freq, key = lambda x: x[1], reverse=True)
```

In [0]:
```
words_freq[:30]
```

Out[0]:
```
[('pre order', 13278),
 ('gb ram', 8942),
 ('t mobile', 8295),
 ('appear \ufeff1', 7467),
 ('infinity display', 6357),
 ('new phone', 5603),
 ('new flagship', 5374),
 ('gear vr', 5336),
 ('screen protector', 4944),
 ('google pixel', 4895),
 ('home button', 4839),
 ('tag ifttt', 4769),
 ('iris scanner', 4667),
 ('pixel xl', 4478),
 ('bixby button', 4340),
 ('look like', 4275),
 ('fingerprint sensor', 4090),
 ('south korea', 3956),
 ('bixby voice', 3814),
 ('ram gb', 3799),
 ('flag irrelevant', 3799),
 ('new york', 3777),
 ('flagship phone', 3642),
 ('facial recognition', 3582),
 ('midnight black', 3522),
 ('aspect ratio', 3505),
 ('battery life', 3405),
 ('wireless charge', 3336),
 ('red tint', 3260),
 ('fingerprint scanner', 3254)]
```

Most important attributes for Galaxy S8:

The 30 most frequently occuring bigrams in the reviews/post based on the above analysis. This gives us a hint on what features/characteristics of the phone are being discussed by the users the most. Some of the most talked about attributes based on our analysis above were:

1. Memory (as indicated by 'gb ram')
2. Infinity display
3. Iris Scanner
4. Bixby button
5. Fingerprint sensor/scanner
6. Bixby voice
7. Facial recognition
8. Red tint (an issue faced by some users with Samsung products)
9. Battery life
10. Wireless charger

In [0]: