exploratory

October 29, 2021

1 Exploration

This notebook is dedicated to exploring the SXSW Twitter dataset with an eye towards extracting brand-related sentiments.

1.1 Bird's Eye View

I begin my exploratory analysis by trying to get an overall sense of what people were talking about regarding Apple and Google.

[1]:
```
import json
import re
from functools import partial
from pprint import pprint
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import nltk

# Set Seaborn theme and default palette
sns.set_theme(font_scale=1.25, style="darkgrid")
sns.set_palette("deep", desat=0.85, color_codes=True)

# Turn on inline plotting
%matplotlib inline

# Load Black auto-formatter
%load_ext nb_black

# Enable automatic reloading
%load_ext autoreload
%autoreload 2
```

```
<IPython.core.display.Javascript object>
```

[2]:
```
from ndg_tools import language as lang, plotting
from ndg_tools.sklearn.vectorizers import FreqVectorizer
```

```
# Set my default MPL settings
plt.rcParams.update(plotting.MPL_DEFAULTS)
```

```
[nltk_data] Downloading package averaged_perceptron_tagger to
[nltk_data]     C:\Users\ndgig\AppData\Roaming\nltk_data…
[nltk_data]   Package averaged_perceptron_tagger is already up-to-
[nltk_data]       date!
[nltk_data] Downloading package universal_tagset to
[nltk_data]     C:\Users\ndgig\AppData\Roaming\nltk_data…
[nltk_data]   Package universal_tagset is already up-to-date!
[nltk_data] Downloading package wordnet to
[nltk_data]     C:\Users\ndgig\AppData\Roaming\nltk_data…
[nltk_data]   Package wordnet is already up-to-date!
```

```
<IPython.core.display.Javascript object>
```

[3]:
```
df = pd.read_json("data/processed_tweets.json")
df.head()
```

[3]:

	text	object_of_emotion	emotion	target
0	.@wesley83 I have a 3G iPhone. After 3 hrs twe…	iPhone	Negative	0
1	@jessedee Know about @fludapp ? Awesome iPad/i…	iOS App	Positive	1
2	@swonderlin Can not wait for #iPad 2 also. The…	iPad	Positive	1
3	@sxsw I hope this year's festival isn't as cra…	iOS App	Negative	0
4	@sxtxstate great stuff on Fri #SXSW: Marissa M…	Google	Positive	1

```
<IPython.core.display.Javascript object>
```

[4]:
```
# Get indices for each major brand
apple_indices = df["object_of_emotion"] == "Apple"
google_indices = df["object_of_emotion"] == "Google"

# Slice out tweets for each category
apple_tweets = df.loc[apple_indices, "text"]
google_tweets = df.loc[google_indices, "text"]

apple_tweets.head(10)
```

[4]:
```
9      Counting down the days to #sxsw plus strong Ca…
40     @mention - Great weather to greet you for #sxs…
47     HOORAY RT UI@mention Apple Is Opening A Pop-Up…
49     wooooo!!! UI@mention Apple store downtown Aust…
```

```
62     #OMFG! RT @mention Heard about Apple's pop-up …
64     Again? RT @mention Line at the Apple store is …
83     Nice!! RT @mention Hey, Apple fans! Get a peek…
109    Kawasaki: "Not C.S. Lewis level reasoning, but…
111    Kawasaki: "pagemaker saved Apple." Oh those we…
116    At #SXSW, #Apple schools the marketing experts…
Name: text, dtype: object
```

```
<IPython.core.display.Javascript object>
```

I define some stopwords and make a large set of stopwords to use later.

[5]:
```
my_stop = {
    "rt",
    "#sxsw",
    "southbysouthwest",
    "sxswi",
    "austin",
    "#austin",
    "sxsw",
    "tweet",
    "tweet's",
    "america",
    "twitter",
    "#sxswi's",
    "mention",
    "link",
    "#sxswi",
}

brand_stop = {
    "google",
    "android",
    "andoid",
    "androidsxsw",
    "applesxsw",
    "ipad",
    "iphone",
    "app",
    "apple",
}

all_stop = my_stop | brand_stop | lang.fetch_stopwords("nltk_english |␣
 ↪sklearn_english")

pprint(all_stop, compact=True, width=120)
```

```
{'#austin', '#sxsw', '#sxswi', "#sxswi's", 'a', 'about', 'above', 'across',
'after', 'afterwards', 'again', 'against',
```

```
 'ain', 'all', 'almost', 'alone', 'along', 'already', 'also', 'although',
'always', 'am', 'america', 'among', 'amongst',
 'amoungst', 'amount', 'an', 'and', 'andoid', 'android', 'androidsxsw',
'another', 'any', 'anyhow', 'anyone',
 'anything', 'anyway', 'anywhere', 'app', 'apple', 'applesxsw', 'are', 'aren',
"aren't", 'around', 'as', 'at', 'austin',
 'back', 'be', 'became', 'because', 'become', 'becomes', 'becoming', 'been',
'before', 'beforehand', 'behind', 'being',
 'below', 'beside', 'besides', 'between', 'beyond', 'bill', 'both', 'bottom',
'but', 'by', 'call', 'can', 'cannot',
 'cant', 'co', 'con', 'could', 'couldn', "couldn't", 'couldnt', 'cry', 'd',
'de', 'describe', 'detail', 'did', 'didn',
 "didn't", 'do', 'does', 'doesn', "doesn't", 'doing', 'don', "don't", 'done',
'down', 'due', 'during', 'each', 'eg',
 'eight', 'either', 'eleven', 'else', 'elsewhere', 'empty', 'enough', 'etc',
'even', 'ever', 'every', 'everyone',
 'everything', 'everywhere', 'except', 'few', 'fifteen', 'fifty', 'fill',
'find', 'fire', 'first', 'five', 'for',
 'former', 'formerly', 'forty', 'found', 'four', 'from', 'front', 'full',
'further', 'get', 'give', 'go', 'google',
 'had', 'hadn', "hadn't", 'has', 'hasn', "hasn't", 'hasnt', 'have', 'haven',
"haven't", 'having', 'he', 'hence', 'her',
 'here', 'hereafter', 'hereby', 'herein', 'hereupon', 'hers', 'herself', 'him',
'himself', 'his', 'how', 'however',
 'hundred', 'i', 'ie', 'if', 'in', 'inc', 'indeed', 'interest', 'into', 'ipad',
'iphone', 'is', 'isn', "isn't", 'it',
 "it's", 'its', 'itself', 'just', 'keep', 'last', 'latter', 'latterly', 'least',
'less', 'link', 'll', 'ltd', 'm', 'ma',
 'made', 'many', 'may', 'me', 'meanwhile', 'mention', 'might', 'mightn',
"mightn't", 'mill', 'mine', 'more', 'moreover',
 'most', 'mostly', 'move', 'much', 'must', 'mustn', "mustn't", 'my', 'myself',
'name', 'namely', 'needn', "needn't",
 'neither', 'never', 'nevertheless', 'next', 'nine', 'no', 'nobody', 'none',
'noone', 'nor', 'not', 'nothing', 'now',
 'nowhere', 'o', 'of', 'off', 'often', 'on', 'once', 'one', 'only', 'onto',
'or', 'other', 'others', 'otherwise', 'our',
 'ours', 'ourselves', 'out', 'over', 'own', 'part', 'per', 'perhaps', 'please',
'put', 'rather', 're', 'rt', 's',
 'same', 'see', 'seem', 'seemed', 'seeming', 'seems', 'serious', 'several',
'shan', "shan't", 'she', "she's", 'should',
 "should've", 'shouldn', "shouldn't", 'show', 'side', 'since', 'sincere', 'six',
'sixty', 'so', 'some', 'somehow',
 'someone', 'something', 'sometime', 'sometimes', 'somewhere',
'southbysouthwest', 'still', 'such', 'sxsw', 'sxswi',
 'system', 't', 'take', 'ten', 'than', 'that', "that'll", 'the', 'their',
'theirs', 'them', 'themselves', 'then',
 'thence', 'there', 'thereafter', 'thereby', 'therefore', 'therein',
'thereupon', 'these', 'they', 'thick', 'thin',
```

```
 'third', 'this', 'those', 'though', 'three', 'through', 'throughout', 'thru',
'thus', 'to', 'together', 'too', 'top',
 'toward', 'towards', 'tweet', "tweet's", 'twelve', 'twenty', 'twitter', 'two',
'un', 'under', 'until', 'up', 'upon',
 'us', 've', 'very', 'via', 'was', 'wasn', "wasn't", 'we', 'well', 'were',
'weren', "weren't", 'what', 'whatever',
 'when', 'whence', 'whenever', 'where', 'whereafter', 'whereas', 'whereby',
'wherein', 'whereupon', 'wherever',
 'whether', 'which', 'while', 'whither', 'who', 'whoever', 'whole', 'whom',
'whose', 'why', 'will', 'with', 'within',
 'without', 'won', "won't", 'would', 'wouldn', "wouldn't", 'y', 'yet', 'you',
"you'd", "you'll", "you're", "you've",
 'your', 'yours', 'yourself', 'yourselves'}
```

```
<IPython.core.display.Javascript object>
```

1.1.1 Apple Quadgrams

[6]:
```
apple_quad = lang.scored_quadgrams(
    apple_tweets,
    preprocessor=lambda x: x.lower(),
    metric="likelihood_ratio",
    stopwords=my_stop,
    min_freq=5,
)

apple_quad.head(20)
```

```
HBox(children=(FloatProgress(value=0.0, max=666.0), HTML(value='')))
```

[6]:
```
quadgram
(pop, up, store, in)           1885.407154
(apple, pop, up, store)        1852.171196
(opening, pop, up, store)      1772.326695
(pop, up, store, at)           1674.016805
(open, pop, up, store)         1658.820686
(pop, up, apple, store)        1645.065413
(pop, up, store, for)          1613.412937
(up, pop, up, store)           1585.743096
(apple, pop, up, shop)         1531.373324
(to, open, pop, up)            1505.511433
(the, apple, pop, up)          1497.927955
(is, opening, pop, up)         1411.367442
(open, pop, up, shop)          1349.639500
(pop, up, shop, at)            1321.329124
(apple, opening, pop, up)      1294.518478
(before, it, even, begins)     1279.159460
```

```
(the, pop, up, apple)          1207.625959
(no, one, ever, heard)         1199.392611
(because, they, don, go)       1177.277746
(technology, no, one, ever)    1151.957780
Name: score, dtype: float64
```

```
<IPython.core.display.Javascript object>
```

Many of the quadgrams (according to the 'likelihood_ratio' metric) are about Apple's pop-up store where the iPad 2 is being launched. This article describes the crowd swarming for the launch. Also this sentence seems to be popular:

> "Apple comes up with cool technology no one's ever heard of because they don't go to conferences."

1.1.2 Google Quadgrams

[7]:
```
google_quad = lang.scored_quadgrams(
    google_tweets,
    preprocessor=lambda x: x.lower(),
    metric="likelihood_ratio",
    stopwords=my_stop,
    min_freq=5,
)
google_quad.head(20)
```

```
HBox(children=(FloatProgress(value=0.0, max=522.0), HTML(value='')))
```

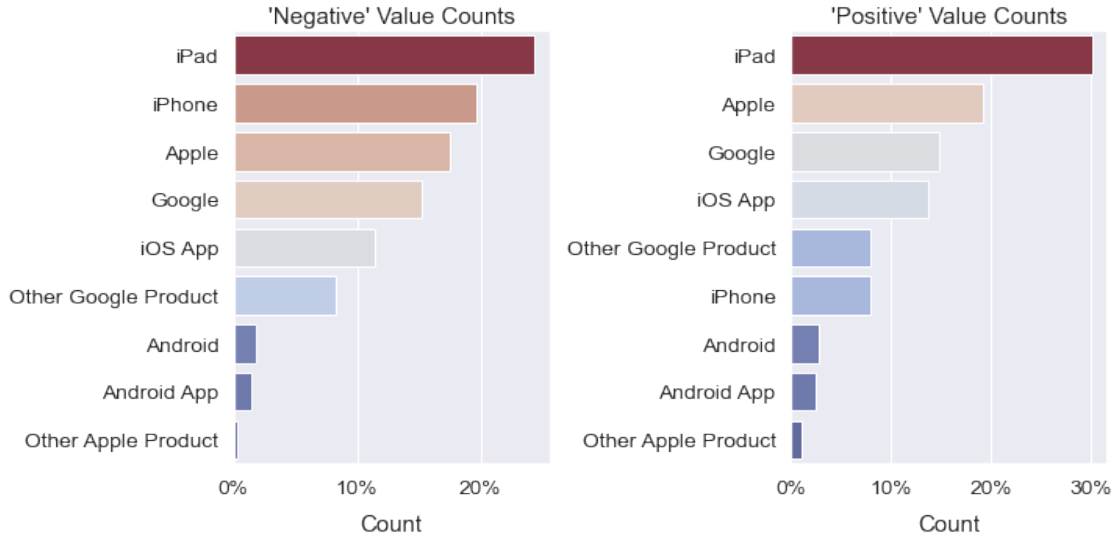
[7]:
```
quadgram
(new, social, network, called)          1007.270392
(major, new, social, network)            955.562655
(social, network, called, circles)       929.589424
(network, called, circles, possibly)     799.208825
(launch, major, new, social)             784.610473
(launch, new, social, network)           760.231698
(called, circles, possibly, today)       723.792797
(to, launch, major, new)                 684.003281
(google, to, launch, major)              619.854722
(marissa, mayer, google, will)           590.063949
(to, launch, new, social)                588.522054
(before, you, speak, mark)               510.882437
(think, before, you, speak)              503.996413
(new, think, before, you)                465.506819
(speak, mark, belinsky, 911tweets)       460.776865
(mark, belinsky, 911tweets, panel)       452.995125
(digital, physical, worlds, through)     440.644057
(you, speak, mark, belinsky)             411.662504
```

```
(google, to, launch, new)                411.445568
(physical, worlds, through, mobile)      388.900340
Name: score, dtype: float64
```

```
<IPython.core.display.Javascript object>
```

The top quadrams about Google all have to do with the anticipated Google Circles launch.

[8]:
```
fig = plotting.countplot(
    df.explode("object_of_emotion").groupby("emotion")["object_of_emotion"],
    normalize=True,
)
```




```
<IPython.core.display.Javascript object>
```

There is no glaringly obvious pattern in the counts of 'Negative' and 'Positive' tweets for each brand. Talk about the new iPad leads in both the 'Negative' and 'Positive' categories, whereas Google leads in the 'Neutral' category.

A color palette for the sentiment classes.

[9]:
```
emo_pal = dict(Negative="r", Neutral="gray", Positive="g")
emo_pal
```

[9]:
```
{'Negative': 'r', 'Neutral': 'gray', 'Positive': 'g'}
```

```
<IPython.core.display.Javascript object>
```

1.2 Keywords by Brand

I construct "superdocuments" by grouping by 'emotion' and 'object_of_emotion' and concatenating the raw tweets in each group. Every brand/product will have 2 superdocuments: positive and

negative.

[10]:
```
brand_docs = (
    # Get Series where each value is a list of row indices
    pd.Series(df.groupby(["emotion", "object_of_emotion"]).groups)
    # Replace lists of row indices with sliced out tweets
    .map(lambda x: df.loc[x, "text"])
    # Fuse the tweets together
    .map(lambda x: " ".join(x))
)
# Get rid of Neutral group and swap index levels
brand_docs = brand_docs.drop(index=np.nan, level=1).swaplevel(0, 1)
brand_docs
```

[10]:
```
Android                Negative    they took away the lego pit but replaced it
wi…
Android App            Negative    Beware, the android #sxsw app for schedules
is…
Apple                  Negative    Again? RT @mention Line at the Apple store is
…
Google                 Negative    @mention - False Alarm: Google Circles Not
Com…
Other Apple Product    Negative    @mention I meant iTunes doesn't work for me
(I…
Other Google Product   Negative    UI@mention Google to Launch Major New Social
N…
iOS App                Negative    @sxsw I hope this year's festival isn't as
cra…
iPad                   Negative    attending @mention iPad design headaches
#sxsw…
iPhone                 Negative    .@wesley83 I have a 3G iPhone. After 3 hrs
twe…
Android                Positive    #SXSW is just starting, #CTIA is around the
co…
Android App            Positive    Find & Start Impromptu Parties at #SXSW With
@…
Apple                  Positive    Counting down the days to #sxsw plus strong
Ca…
Google                 Positive    @sxtxstate great stuff on Fri #SXSW: Marissa
M…
Other Apple Product    Positive    Pedicab + iPhone charger would be epic win.
#S…
Other Google Product   Positive    Gotta love this #SXSW Google Calendar
featurin…
iOS App                Positive    @jessedee Know about @fludapp ? Awesome
iPad/i…
iPad                   Positive    @swonderlin Can not wait for #iPad 2 also.
```

```
The…
iPhone                Positive    I love my @mention iPhone case from #Sxsw but
…
dtype: object
```

```
<IPython.core.display.Javascript object>
```

Now I use my `FreqVectorizer` to extract tf-idf vectors for each superdocument. Each document is transformed into a vector of TF-IDF scores where the features are words. For each term in each superdocument, the score is (roughly) the term's local frequency times a measure of its rarity in the corpus as a whole. I set the 'max_df' to 0.3, meaning that terms which occur in more than 30% of the documents are excluded. This separates the wheat from the chaff.

See the main notebook for an overview of my `FreqVectorizer` class, which extends Scikit-Learns `TfidfVectorizer`.

[11]:
```
# Make vectorizer
tfidf = FreqVectorizer(
    stop_words=all_stop,
    ngram_range=(1, 2),
    norm="l2",
    use_idf=True,
    max_df=0.25,
)

# Make vectors
brand_vecs = tfidf.fit_transform(brand_docs.values)

# Place vectors in DataFrame
brand_vecs = lang.frame_doc_vecs(
    brand_vecs,
    tfidf.vocabulary_,
    brand_docs.index,
)

# Transpose so that vectors run along columns
brand_vecs = brand_vecs.T.sort_index(level=0, axis=1)
brand_vecs.index = brand_vecs.index.str.replace("_", " ")

# Sort for effect
brand_vecs.sort_values(("Apple", "Negative"), ascending=False)
```

[11]:

	Android		Android App		Apple	\
	Negative	Positive	Negative	Positive	Negative	Positive
fascist	0.0	0.0	0.0	0.0	0.311925	0.000000
fascist company	0.0	0.0	0.0	0.0	0.267364	0.000000
classiest	0.0	0.0	0.0	0.0	0.200523	0.000000
pop	0.0	0.0	0.0	0.0	0.192015	0.471025
swisher	0.0	0.0	0.0	0.0	0.178243	0.000000

```
…                  …       …          …       …       …       …
girl congrats        0.0      0.0         0.0      0.0  0.000000  0.000000
ginger man           0.0      0.0         0.0      0.0  0.000000  0.000000
ginger               0.0      0.0         0.0      0.0  0.000000  0.000000
gilt group           0.0      0.0         0.0      0.0  0.000000  0.000000
zzzs battery         0.0      0.0         0.0      0.0  0.000000  0.000000

                 Google          Other Apple Product           \
               Negative Positive            Negative  Positive
fascist             0.0      0.0                 0.0  0.000000
fascist company     0.0      0.0                 0.0  0.000000
classiest           0.0      0.0                 0.0  0.000000
pop                 0.0      0.0                 0.0  0.000000
swisher             0.0      0.0                 0.0  0.000000
…                   …        …                   …       …
girl congrats       0.0      0.0                 0.0  0.048621
ginger man          0.0      0.0                 0.0  0.000000
ginger              0.0      0.0                 0.0  0.000000
gilt group          0.0      0.0                 0.0  0.048621
zzzs battery        0.0      0.0                 0.0  0.000000

               Other Google Product           iOS App              iPad  \
                           Negative Positive Negative Positive  Negative
fascist                         0.0      0.0      0.0      0.0  0.000000
fascist company                 0.0      0.0      0.0      0.0  0.000000
classiest                       0.0      0.0      0.0      0.0  0.000000
pop                             0.0      0.0      0.0      0.0  0.068462
swisher                         0.0      0.0      0.0      0.0  0.000000
…                               …        …        …        …       …
girl congrats                   0.0      0.0      0.0      0.0  0.000000
ginger man                      0.0      0.0      0.0      0.0  0.000000
ginger                          0.0      0.0      0.0      0.0  0.000000
gilt group                      0.0      0.0      0.0      0.0  0.000000
zzzs battery                    0.0      0.0      0.0      0.0  0.000000

                          iPhone
                Positive  Negative Positive
fascist         0.000000  0.000000      0.0
fascist company 0.000000  0.000000      0.0
classiest       0.000000  0.000000      0.0
pop             0.380749  0.000000      0.0
swisher         0.000000  0.000000      0.0
…                  …         …        …
girl congrats   0.000000  0.000000      0.0
ginger man      0.006885  0.000000      0.0
ginger          0.006885  0.000000      0.0
gilt group      0.000000  0.000000      0.0
```

```
zzzs battery     0.000000  0.025761      0.0

[23953 rows x 18 columns]
```

```
<IPython.core.display.Javascript object>
```

[12]:
```
def plot_brand_clouds(
    column,
    dst_schema="images/{column}_eda_clouds.svg",
    cmap=("Reds", "Greens"),
    size=(10, 4),
    ncols=1,
    max_font_size=None,
    random_state=156,
    brand_vecs=brand_vecs,
    **kwargs,
):
    fig = plotting.wordcloud(
        brand_vecs.loc[:, column],
        cmap=list(cmap),
        size=size,
        ncols=ncols,
        repeat=True,
        max_font_size=max_font_size,
        random_state=random_state,
        **kwargs,
    )
    fig.savefig(dst_schema.format(column=column.lower().replace(" ", "_")))
    return fig
```

```
<IPython.core.display.Javascript object>
```

1.2.1 Apple

Here is one of the most striking Wordclouds in the notebook. It reveals that people were talking about Apple being a "fascist company". This began with tech journalist Kara Swisher, who provoked a flurry of tweets by saying that Apple was the "classiest fascist company in America".

On the positive side, a lot of people were talking about the pop-up store and circulating the following quote:

> apple comes up with cool technology no one's ever heard of because they don't go to conferences

[13]:
```
fig = plot_brand_clouds("Apple")
```

Negative

Positive


```
<IPython.core.display.Javascript object>
```

1.2.2 iPhone

Regarding the negative, there was a tweet bragging about T-Mobile, retweeted a few times:

> Looking forward to delicious T-Mobile 4G here in Austin while iPhone users struggle to do anything. #SXSW

There were similar remarks about AT&T's service making iPhone's useless as a brick: > Austin is getting full, and #SXSW is underway. I can tell because my iPhone is an intermittent brick. #crowded

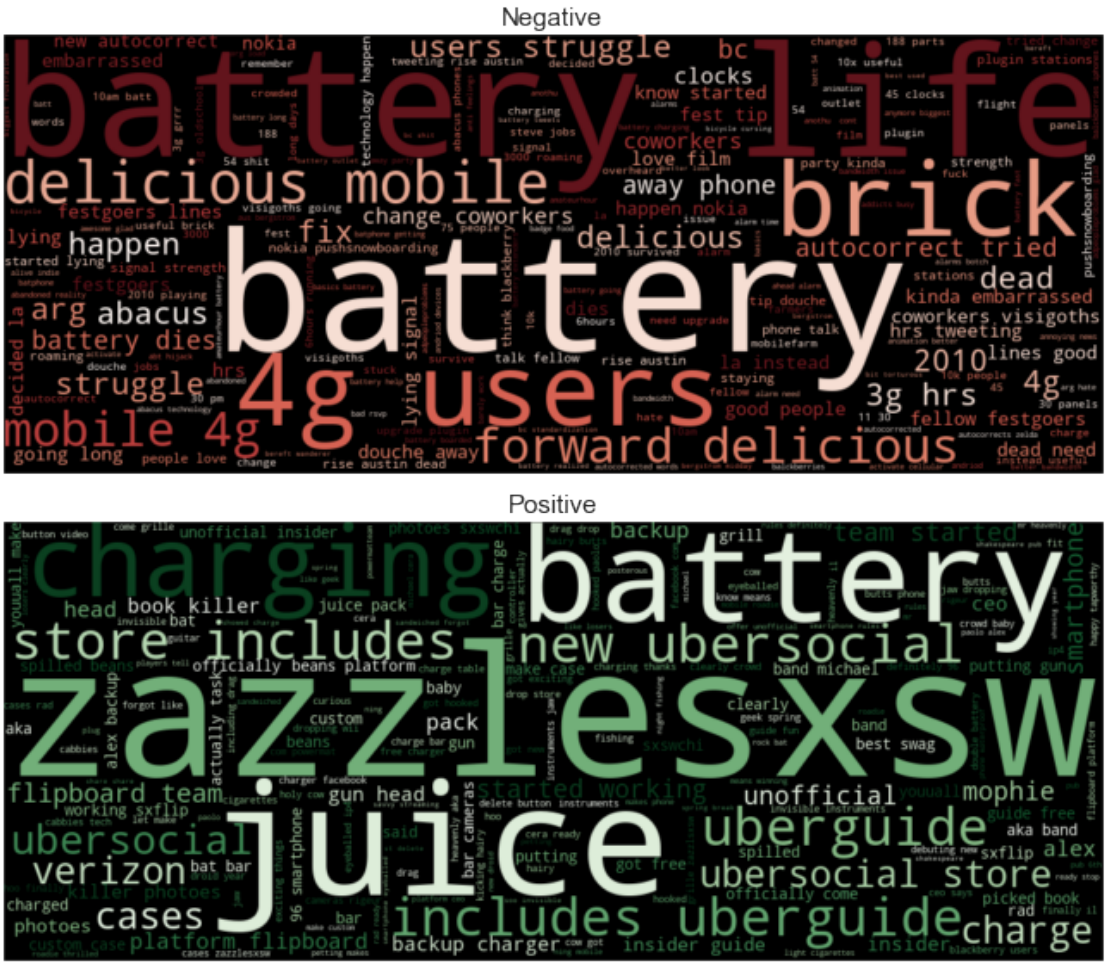
> Decided to go to LA instead of #SXSW, because my AT&T iPhone would be about as useful as a brick in Austin.

There was also talk about battery life problems.

> #sxsw is exposing my iphone's horrendous battery life.

> This #SXSW I am grateful for: my bicycle, having a back-up Twitter app. Cursing: losing an hour of zzzs, iPhone battery life.
>
> Disgusted with my iPhone's battery life. Already down to 11% at 3:30 pm while my blackberry is going strong. #Sxsw

[14]:
```
fig = plot_brand_clouds("iPhone")
```




```
<IPython.core.display.Javascript object>
```

Many positive tweets seem to be about how glad people are to have a charger.

The positive chatter about Flipboard was related to its well-designed iPad app.

> Epicurious, flipboard, CNN, wired, and MOMA as examples of good iPad design #SXSW {link}

The talk about Zazzle was related to designing custom iPhone cases, a service they offer.

> Zazzle is gearing up to hit #SXSW! Look out for our tweets on where you can come by to create your own iPhone case! #zazzlesxsw

There are some positive tweets about the newly-available **Verizon iPhones** and their superior service.

1.2.3 iPad

The talk about "design headaches" is related to a talk given by Josh Clark on the topic of iPad design challenges and failings. It seems like constructive criticism which is not intended to harm the brand.

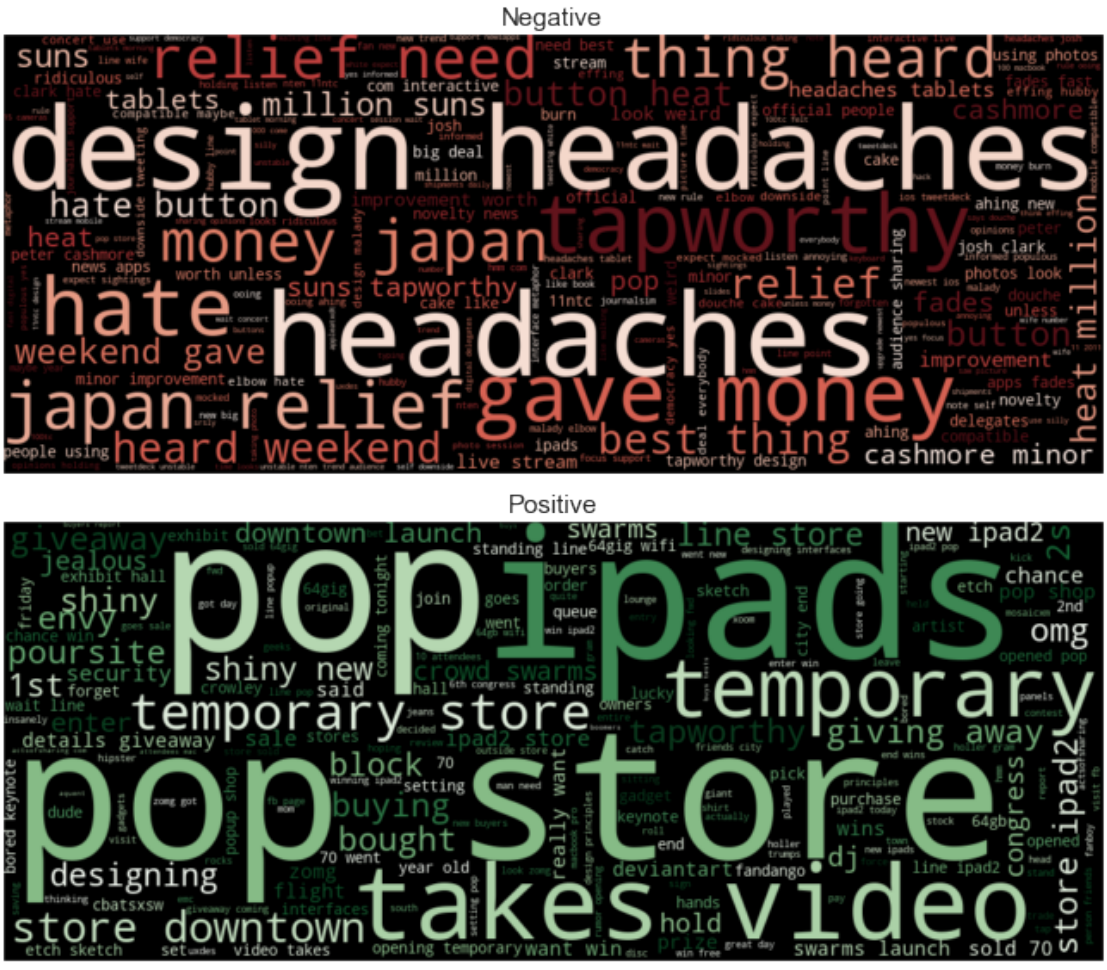
The talk about "japan relief" has to do with the following virally circulated quote:

> Best thing I've heard this weekend at #SXSW "I gave my iPad 2 money to #Japan relief. I don't need an iPad 2."

The quote expresses a definite negative attitude towards Apple and iPad, which it implies are associated with self-indulgence and excess.

The positive chatter is again focused on the pop-up store, with words like "shiny", "gadget", and "envy" showing up.

[15]:
```
fig = plot_brand_clouds("iPad")
```



```
<IPython.core.display.Javascript object>
```

1.2.4 iOS Apps

The negative chatter seems to focus on the short lifecycle of news apps, and is related to this article from the time period.

There are some complaints about apps using geolocation eating up battery life.

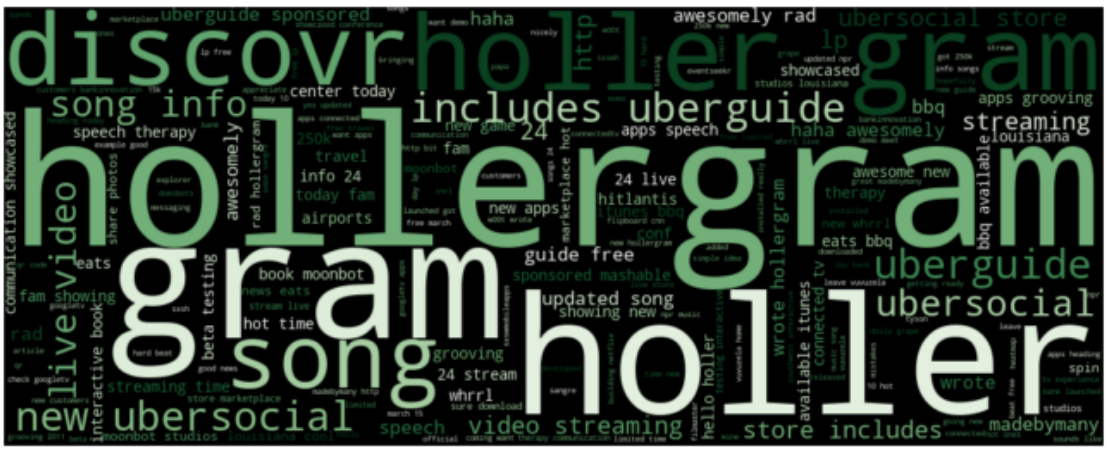
Holler Gram was a social media app which existed for use at South by Southwest, according this article.

These wordclouds don't seem to be as interesting as some of the others.

[16]:
```
fig = plot_brand_clouds("iOS App")
```

Negative

Positive




```
<IPython.core.display.Javascript object>
```

1.2.5 Google

There appears to have been a Guardian article going around titled "The #Google and #Bing smackdown in all its bloody banality".

People were saying things like:

> So true!!! RT @mention 'Google lost its way by caring too much for the business vs. the users' - @mention #psych #sxsw

People seemed to be excited about a talk given by Marissa Mayer. They were also anticipating the launch of Google Circles.

This mantra was being virally tweeted.

> RT @mention @mention "Google before you tweet" is the new "think before you speak." - Mark Belinsky, #911tweets panel at #SXSW.

[17]:
```
fig = plot_brand_clouds("Google")
```


```
<IPython.core.display.Javascript object>
```

1.2.6 Android

The most interesting phrase here is "apps like ipod", which appears to originate from the following tweet:

> @mention Android needs a way to group apps like you can now do with iPad/iPod. #SXSW #hhrs

There is also talk about bugginess, as in:

> Is it just me or has the @mention client for Android gotten really buggy lately? #SXSW to blame?

This is good news for Apple.

[18]:
```
fig = plot_brand_clouds("Android")
```

<IPython.core.display.Javascript object>

1.2.7 Android Apps

There were some complaints about the specific South by Southwest Android app.

> Beware, the android #sxsw app for schedules is completely innacurate. Just walked to the hyatt for no reason #sxswfail

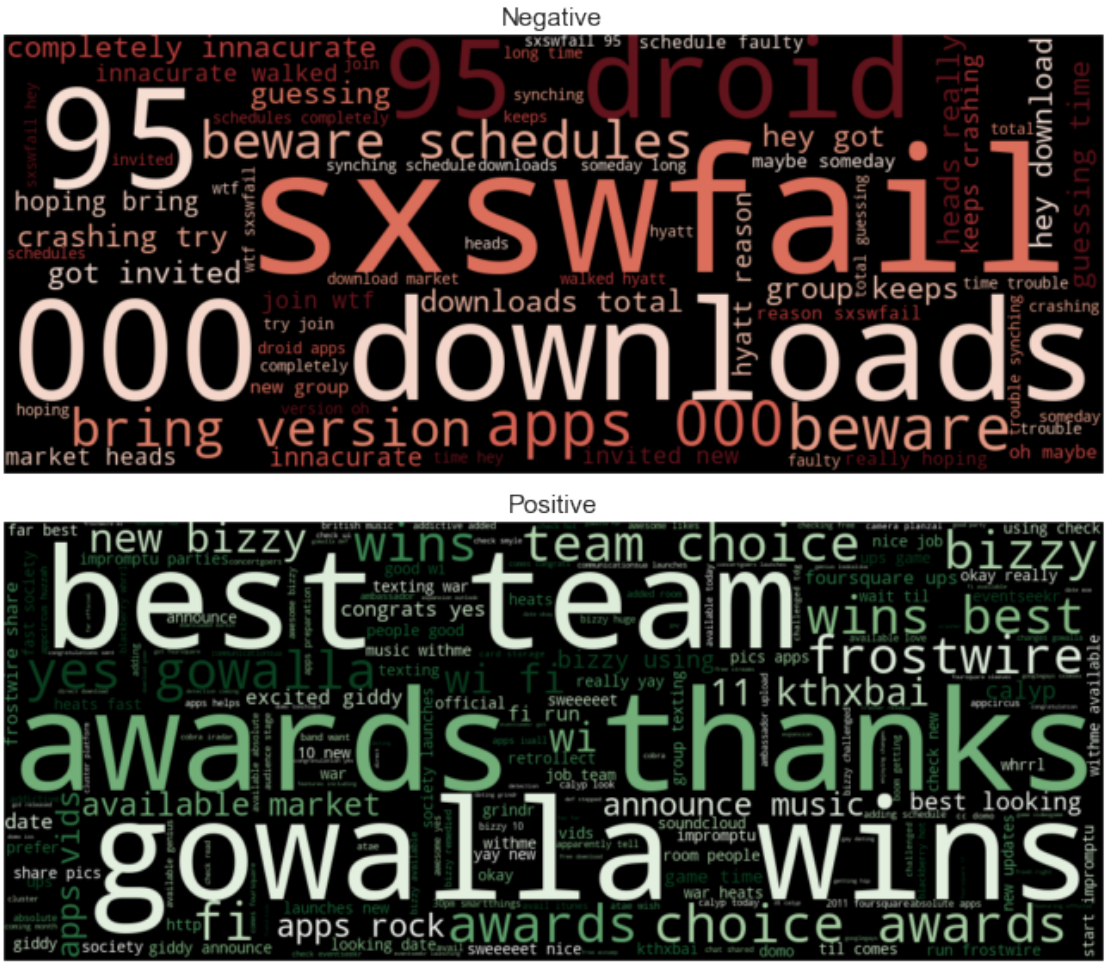
A few people tweeted about this, although it doesn't seem particularly juicy:

> 95% of iPhone and Droid apps have less than 1,000 downloads total. #SXSW

There was a lot of cheering for Gowalla's app winning the Team Android Choice Awards:

> Nice! RT @mention Yes! Gowalla wins best Andoid app at the Team Android Choice Awards. Thanks all! #sxsw

[19]:
```
fig = plot_brand_clouds("Android App")
```




```
<IPython.core.display.Javascript object>
```

2 Two Upshots

2.1 You're Viewed as a Tyrant

People like that Apple products just work out of the box, but they find your paternalistic approach to managing your products off-putting. Send the message that when you buy an Apple product, you are free to do what you want with it. Keep control over the most important things, but relinquish control over the less important things. Make people feel like they have the freedom to customize your products as they see fit. Make some concessions to placate the majority, while allowing the elite techno-snobs to continue complaining on the fringe.

2.2 Battery Life Needs Improvement

There were a lot complaints about the iPhone's battery life. One user suggested that their Blackberry was doing much better. There were also complaints about #batterykiller apps which use geolocation in the background. If you made a big publicized effort to increase the iPhone's battery life, that would get people excited.

[ ]: