Predictive Analysis for Upvotes on a Reddit Post

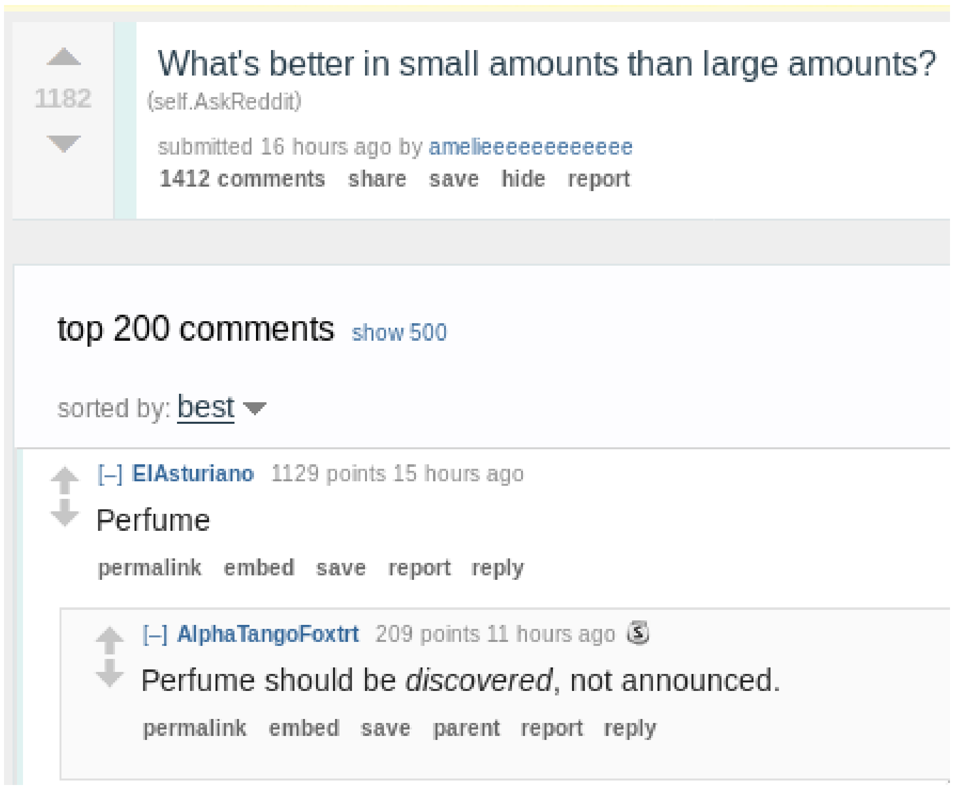
April 10, 2020

1 Concept Description:

Our team's goal is to perform data analysis on the comments sections of reddit. Our initial plan is to extract data from various Reddit comments sections. Given the data we have, we will be use *Naive Bayes* to make a predictive analysis on the amount of upvotes a post receives, comparing it with *k-Nearest Neighbor* and *Support Vector Machines*.

2 Data Collection:

Data was collected from https://github.com/linanqiu/reddit-dataset. In this repository, 50 csv files containing reddit comment section datasets are available with a free use policy granted by Linanqiu. The repo notes this was scraped by another of their tools, `omega-red`.


1182

What's better in small amounts than large amounts? (self.AskReddit)

submitted 16 hours ago by amelieeeeeeeeeeee

1412 comments share save hide report

top 200 comments show 500

sorted by: best

[–] ElAsturiano 1129 points 15 hours ago

Perfume

permalink embed save report reply

[–] AlphaTangoFoxtrt 209 points 11 hours ago

Perfume should be *discovered*, not announced.

permalink embed save parent report reply

Reddit post example

3 Example Description:

Most of the data here is pretty well laid out, the initial repo put it pretty well, so we will be using that for the most part(some adaptations for our project):

The `.csv`s are named `<metareddit>_<subreddit>.csv`. The headers are described here and in `headers.txt`.

- `text`: Text of the comment / thread
- `id`: Unique reddit id for the comment / thread
- `subreddit`: Subreddit that the comment / thread belongs to
- `meta`: Metareddit that the comment / thread belongs to. Subreddits belong to metareddits. A subreddit can be `leagueoflegends`. The metareddit for that subreddit would be `gaming`, which can also include the subreddit `dota2`. **This is an arbitrary distinction from omega-red, that we will not include in our analysis.**
- `time`: UNIX timestamp of the comment / thread
- `author`: Username of the author of the comment / thread
- `ups`: Number of upvotes the comment / thread received
- `downs`: Number of downvotes the comment / thread received
- `authorlinkkarma`: The author's link karma. What is Link Karma?
- `authorkarma`: The author's karma. Reddit FAQ explaining karma.
- `authorisgold`: Boolean indicator for the gold status of the user. `1` for gold users, `0` for non-gold (normal) users. Reddit FAQ explaining gold status.

4 Data Import and Wrangling:

The data from Linanqiu's data set was pretty thoroughly cleaned - there is no punctuation in the text and all letters are lowercase. However, the column names weren't quite helpfully named and some had inconsistent indexing - some start from 1, others started from 0. We then ran the following code to accomplish this:

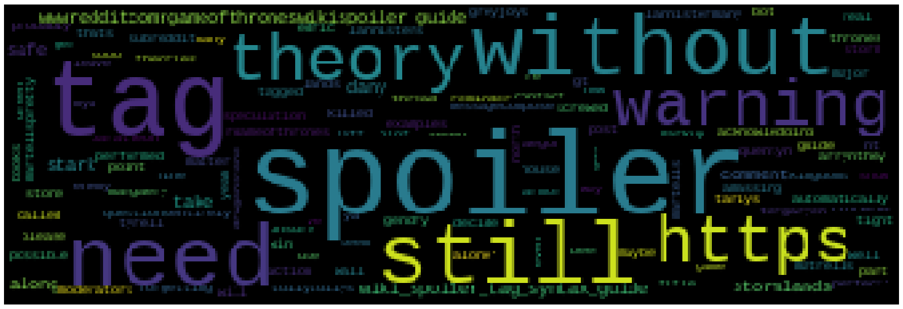
```
In [33]: from os import listdir, path, makedirs
         import pandas as pd
         import numpy as np

         #Pandas setup
         data_files = []
         dataframes = []

         #Clean directory dirs
         data_dir = "../src-data"
         out_dir = "processed-data"

         if not path.exists(out_dir):
             makedirs(out_dir)


         #Go through each file and process them columns
         for filename in listdir(data_dir):
             has_idx = True
             file_list = filename.split('.')
             if(len(file_list) > 1):
                 if filename.split('.')[1] == 'csv':
                     data_files.append(filename)
                     df_temp = (pd.read_csv(data_dir + '/' + filename))
                     df_temp.drop([0], inplace=True)
                     df_temp.drop(columns="Unnamed: 0", inplace=True)
                     #There may be a more elegant way to do this check but this will suffice.
                     for idx,row in enumerate(df_temp['0']):
                         if(row != idx):
                             has_idx = False
                     if(has_idx == True):
                         #There may also be a more elegant way to do this rename.
                         df_temp.rename(columns={
                             "1":"text",
                             "2":"id",
                             "3":"subreddit",
                             "4":"meta",
                             "5":"time",
                             "6":"author",
                             "7":"ups",
                             "8":"downs",
                             "9":"authorlinkkarma",
                             "10":"authorkarma",
                             "11":"authorisgold"
                         }, inplace=True)
                         df_temp.drop(columns="0")
                     df_temp.rename(columns={
                         "0":"text",
                         "1":"id",
                         "2":"subreddit",
                         "3":"meta",
                         "4":"time",
                         "5":"author",
                         "6":"ups",
                         "7":"downs",
                         "8":"authorlinkkarma",
                         "9":"authorkarma",
                         "10":"authorisgold"
                     }, inplace=True)
                     df_temp.to_csv(out_dir + '/' + filename, index=False)
                     dataframes.append(df_temp)
```

5 Exploratory Data Analysis:

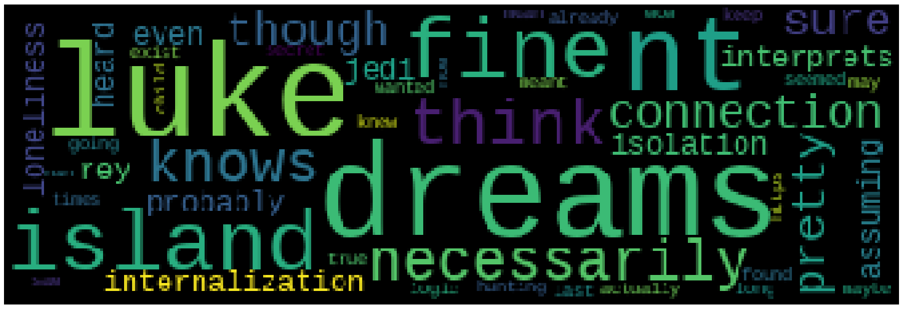
For an initial visualization, we'll write some code to go through the texts and generate some word-clouds:

```
In [37]: from wordcloud import WordCloud
         import matplotlib.pyplot as plt
         for dataframe in dataframes[:5]:
             text = (dataframe['text'].values)
             wordcloud = WordCloud(font_path="mono.ttf").generate(str(text))
             print(dataframe['subreddit'].values[0])
             plt.imshow(wordcloud)
             plt.axis("off")
             plt.show()
```

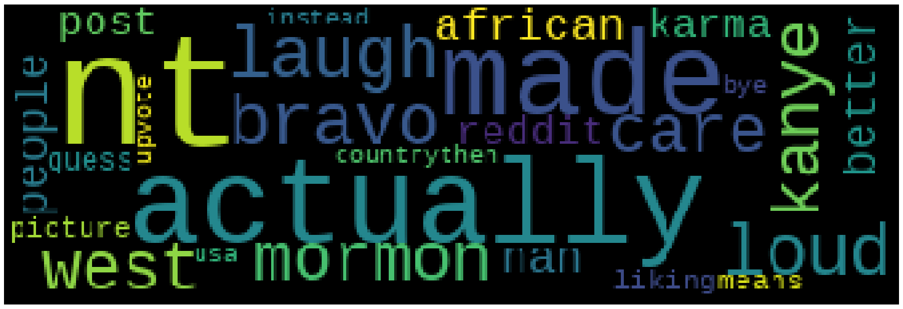
```
gameofthrones
```



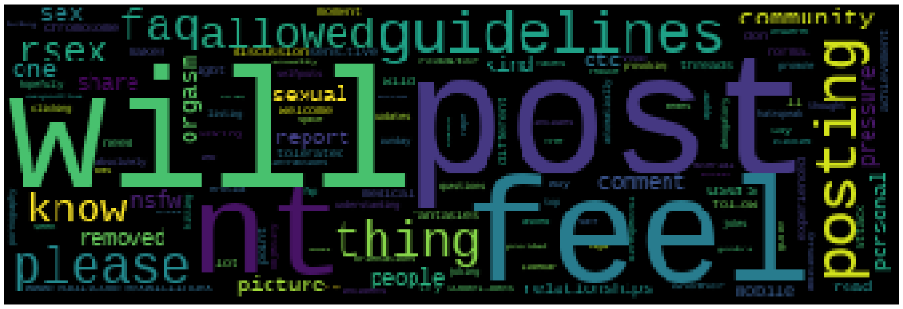
```
starwars
```



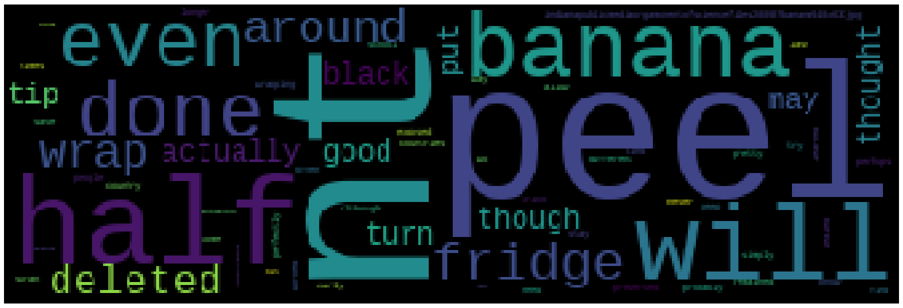
facepalm



sex



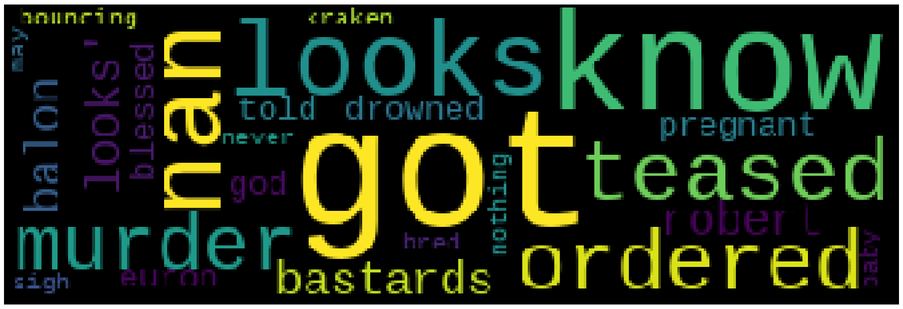
youshouldknow



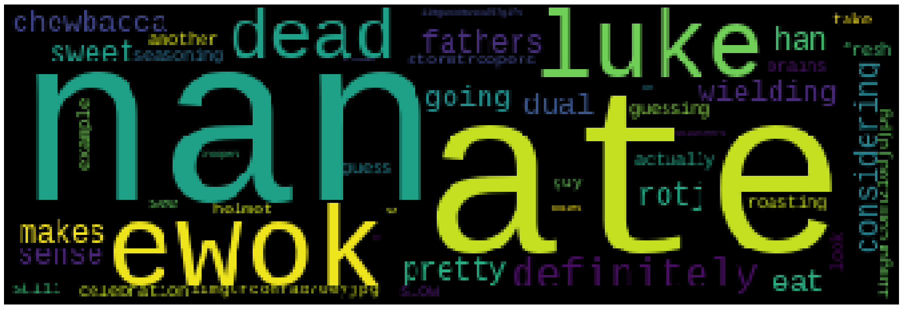
Then, we filtered based off of those with an upvote count higher than 50.
This process revealed some markedly different results.

```
In [38]: for dataframe in dataframes[:5]:
             text = (dataframe.loc[dataframe['ups'] > 50]['text'].values)
             wordcloud = WordCloud(font_path="mono.ttf").generate(str(text))
             print(dataframe['subreddit'].values[0])
             plt.imshow(wordcloud)
             plt.axis("off")
             plt.show()
```

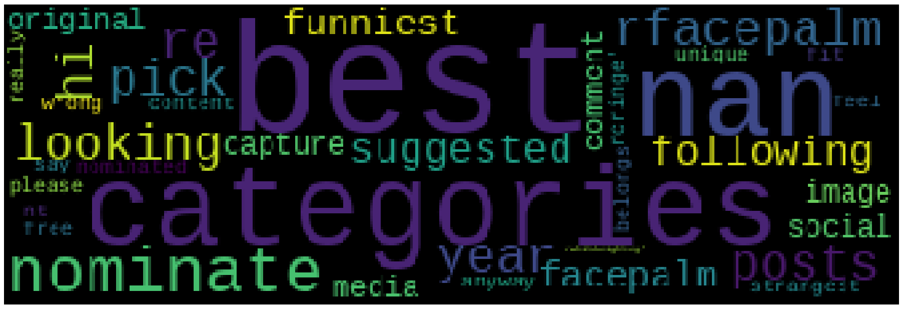
```
gameofthrones
```



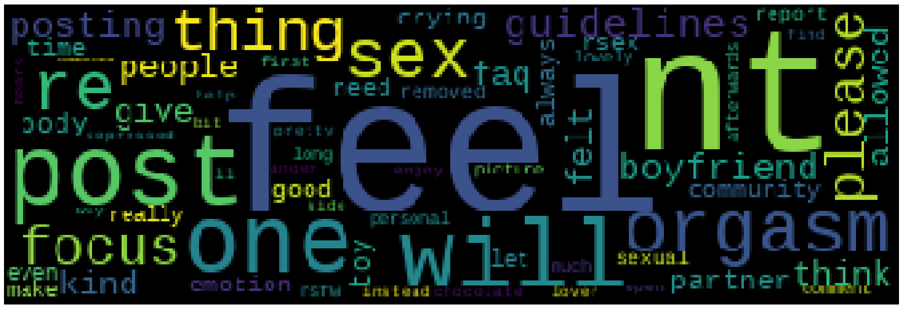
```
starwars
```



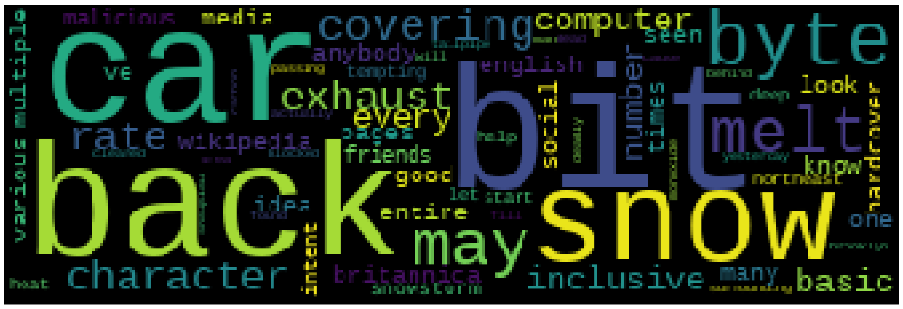
facepalm



sex



youshouldknow



This shows that there may be some further data considerations - what about the `[deleted]` posts? What is going on with 'nan' / 'spoiler'? These are going to yield some interesting ramifications for our analysis stage.

Additionally, there are remnants of contractions, such as "nt", that we probably need to clean up, as well as "https" from links.

6 Mining or Analytics:

Record any mining or analytics techniques you perform. This may include visualizations.

7 Evaluation:

Record the evaluation of your mining or analytics techniques.

8 Results:

Record the results of your findings.

9 References

- https://www.pewresearch.org/wp-content/uploads/sites/8/2016/02/PJ_2016.02.25_Reddit_FINAL.pdf
- https://ieeexplore.ieee.org/document/8622535
- https://github.com/linanqiu/reddit-dataset
- https://stackoverflow.com/questions/39870642/matplotlib-how-to-plot-a-high-resolution-graph
- https://stackoverflow.com/questions/43606339/generate-word-cloud-from-single-column-pandas-dataframe
- https://pandas.pydata.org/pandas-docs/stable/getting_started/10min.html