SAN JOSÉ STATE UNIVERSITY

Subreddit recommender based on

User similarity

CMPE 256 - Large Scale Analytics

Summer 2019

Kunj Parikh <kunj.parikh@sjsu.edu> - 012532065

Table of Contents

Introduction

Recommender systems have gained popularity in a variety of areas like Amazon product recommenders, playlist generators for Youtube, Netflix, Spotify, content recommenders for Facebook, Twitter; recommenders for restaurants, online dating, and many others. Reddit 'The front page of the internet' has more than 100 million active users, more than 100K subreddits. Subreddit examples are like r/worldnews, r/gaming, etc. Each subreddit has millions of users who post content and comment on other posts. We use the comments as our dataset. We then build a profile of each user and recommend new subreddits for a given to explore.

Challenges

The biggest challenge was collecting enough data. To figure out a user that matches a given user perfectly, we need a large dataset to choose from. Reddit doesn't offer a method that returns random users, and I didnt want to crawl using specific subreddits, else the data would be biased towards them. So I depend on /all subreddit that contains highest rated posts from all subreddits.

1. Cold start problem: Some users haven't posted in any subreddits. We suggest most popular subreddits by default.
2. Feature selection: Once we get the dataset of users and their followed subreddits, we use tf-idf to scale down subreddits that are popular with everyone.
3. Size of the data set: It took 3-4 mins to collect 30 users data. So size was limited by time.
4. Synonymy: Two aliases of same user, end up having similar subreddit preference.
5. Random subreddit followers may affect accuracy because it would be part of dataset.
6. Sparse and high dimensional dataset - used PCA.

Data Acquisition and Preprocessing

Data collection

The data-set I started with can be found on Stanford's snap website[1]. The snapshot of dataset is below. It takes each post and tells us the source subreddit, destination subreddit, user who posted it. We can use this to calculate user model.

```
SOURCE_SUBREDDIT tab TARGET_SUBREDDIT tab POST_ID tab TIMESTAMP tab POST_LABEL tab POST_
leagueoflegends teamredditteams 1u4nrps 2013-12-31 16:39:58     1       345.0,298.0,0.75
theredlion      soccer  1u4qkd  2013-12-31 18:18:37     -1      101.0,98.0,0.74257425742
inlandempire    bikela  1u4qlzs 2014-01-01 14:54:35     1       85.0,85.0,0.752941176471
```

Figure 1: Sample available dataset

This dataset also contains sentiment analysis features from VADER (eg. VADER_positive), LIWC (eg. LIWC_Verbs). This could be used to calculate user similarity based on post content. I explored using this sentiment data but in the end didn't include it in the model.

Next I scrape data from live Reddit website. I use python's PRAW[2] module for this. The code for this is present on the github[3]. I also use mongodb[4] to collect maintain and keep the data accessible. The mongodb database name is Reddit, and I use 2 collections - subreddits and users to collect respective data. The sample dataset is dumped as bson files on github.

The strategy was to use /all subreddit and get top 250 posts across all subreddits, get the users of the posts, create a database of those users, and subreddits they posted in. This data collection script was then run at different times to collect different top 250 posts. At times even the number 250 was changed to collect more data.

Note : If you run dataset.py using python3, make sure to add your reddit OAuth details in praw.ini file.

Data processing

This phase can be summarized as the following:

1. First part was validating if user/subreddit information is not empty, in case the database/PRAW query was corrupted.
2. Adding the dataset to mongodb dataset - see mongo.py module on the project github.
3. I am collecting this dataset at different times, so if the user profile already exists in the mongodb dataset I skip processing the user. This way I prevent duplicates which will give wrong weights while computing similarity.
4. Create a pandas dataframe using mongodb dataset.

Validating data and making sure it is valid, updated, and without any errors representing the end of this phase of the project. Some more details of how the data is transformed is in the next section.

Here I show the snapshot of dataset. Rows are users, columns are subreddit names. Cells denote user post stats on corresponding subreddit.

	PewdiepieSubmissions	Minecraft	entitledparents	memes	GrandTheftAutoV	gtaonline	cursedcomments	Wolfenstein	ToiletPaperUSA	dankmem
-Yes-Sir-	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
1QUEEN12	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
1iopen	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
2003Slobra	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0
2510EA	0.0	0.0	0.0	1.0	1.0	1.0	0.0	0.0	0.0	1

5 rows × 4149 columns

Figure 2: Collected dataset

Implementation

The Subreddit recommendation engine focuses on recommending subreddits based on a given user. The idea is to compute user feature based similarity. In this section we will cover the application workflow, feature selection and the different model implementations.

Application workflow

The application assumes mongodb dataset is already created and mongod server is running on the machine. Run dataset.py if this is not done. The prediction engine in kNN.py[5] reads the mongodb collection and loads it in memory. I create a pandas dataframe out of it for easy manipulation. This dataframe will be instantiated in different models. Loading data may be an issue if we don't have enough RAM. Next I ask user for their username using python's input() function - for demo purpose the checked in code takes a random username instead of prompting for input. Given the user name the engine calls PRAW api (using getComments() method in dataset.py) to collect the given user's subreddit stats - I do this because the given username may not be in the scraped dataset. Once I have the data I pass it to the model and recommend new subreddit for the user. The application used github code from logicx24[8] as starting point.

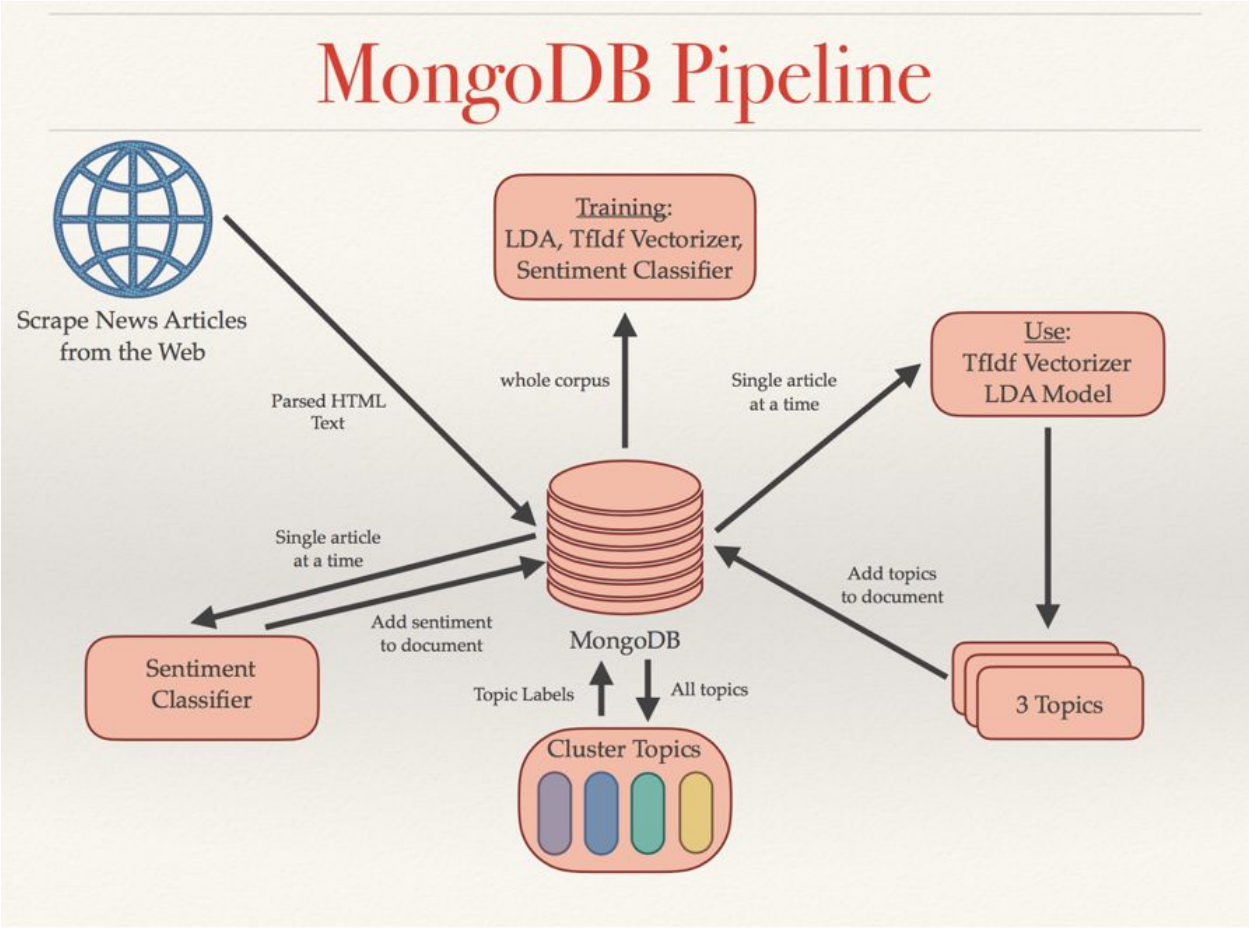

Figure 3: Application workflow with Mongodb[7]

Modeling Data and Feature Engineering

Principal component analysis (PCA)

The dataset I collected is very sparse as shown in Fig 2, And it is high dimensional ~4k columns. This is because there are a large number of subreddits. If I used the data as is I would suffer from “curse of dimensionality” so I use PCA dimensionality reduction technique. I tried with various number of n_components to preserve but finally chose 10 as a good value. Figure 5 shows the dataset after dimensionality reduction. I also wanted to reduce the impact of generally popular subreddits, so I used IDF on the dataset before PCA. This helps to reduce how much subreddits like “AskReddit” (which

are generally popular - akin to words present in all documents) contribute to similarity calculation.

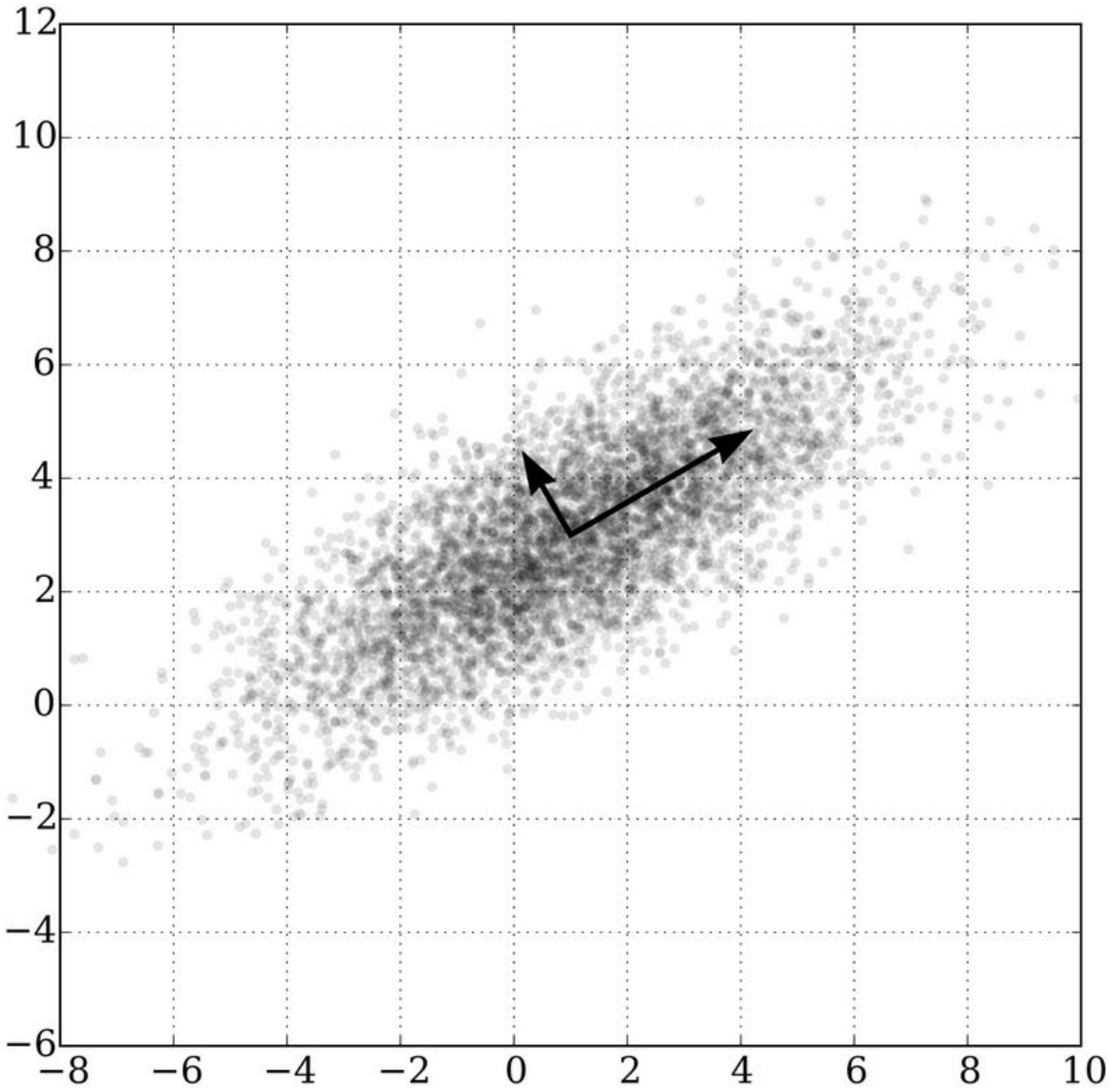

Figure 4: Concept of PCA[6]

	0	1	2	3	4	5	6	7	8	9
-Yes-Sir-	-0.590303	-0.075641	-0.096686	0.023442	-0.154037	-0.027941	-0.031285	0.223193	0.295544	0.377013
1QUEEN12	1.330965	-1.121789	0.308707	0.110620	-0.067700	0.177984	0.125906	-0.353450	0.797143	-0.308894
1iopen	1.877512	-1.136717	0.192306	-0.006869	-0.353502	-0.303375	-0.004259	0.457290	0.150513	0.152242
2003Slobra	-0.903909	-0.165171	0.032738	-0.078973	0.058808	-0.022944	-0.131464	-0.059234	-0.033310	-0.134361
2510EA	-0.296304	1.203761	-0.387954	0.132807	0.049805	1.105133	-0.250573	-0.737542	-0.854322	0.084089

Figure 5: Dataset after dimensionality reduction

User based similarity

I use sklearn's Nearest Neighbors[10] on the transformed dataframe I got after applying PCA to compute user similarity. I tried with different similarity measures like Minkowski, L1, L2.. Finally settled on cosine distance. This is because my dataset can be imagined as documents (users) with word frequency (subreddits), and cosine is the most popular similarity method for text based similarity.

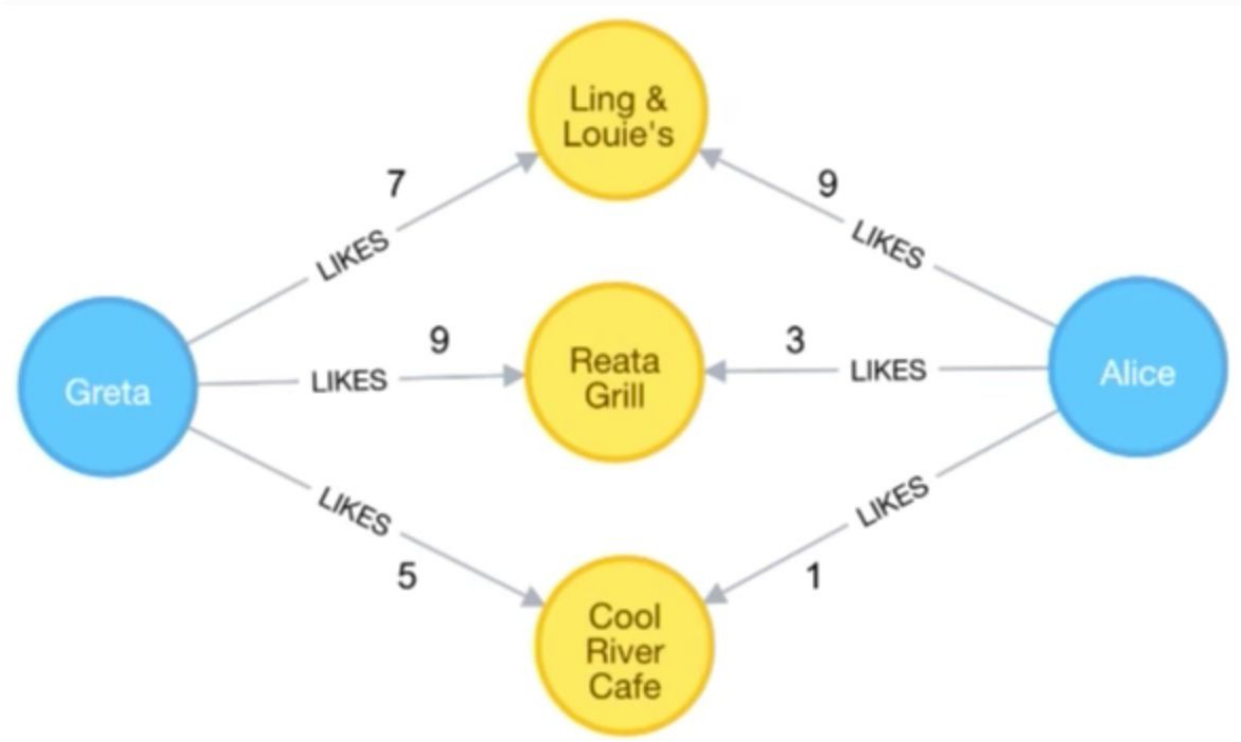

Figure 3: Application workflow[9]

Subreddit recommender

Now that I have similarity of given user with all others, I take first N users and weigh their subreddit stats by their similarity value to given user. I add up all the weighted vectors to get a vector for given user. I discard all the subreddits the given user has already followed/posted in,

the remaining subreddits are sorted in descending order. This is my ranking of recommendation. In the code I print out the rank one recommendation.

Results

Here I show the result of my recommendation for a user ‘Alice_In_Wonderland1”. Based on their subreddit posts history and user based similarity the Subreddit Recommendation engine thinks “/science” is the most likely subreddit they might be interested in, and in which they have not participated yet.

```
names, sim = findNeighbors(df, username)
rec = getRecommendedSubreddit(df, names, sim, username)
print("Recommended subreddit for user {} is {}".format(username, rec[0]))
```

```
Recommended subreddit for user Alice_In_Wonderland1 is science
```

Conclusions

This work presents user based similarity methods to recommend subreddits to Reddit users. We used their past comment history, created user profile, used dimensionality reduction along with cosine based user similarity kNN model. We scrapped data using REST APIs, and stored data using mongodb. The trained model gave us a ranked list of subreddits that would be interesting for the user.

References

1. http://snap.stanford.edu/data/soc-RedditHyperlinks.html
2. https://praw.readthedocs.io/en/latest/
3. https://github.com/KunjParikh/cmpe256/blob/master/iproject/cosine/dataset.py
4. https://github.com/KunjParikh/cmpe256/blob/master/iproject/cosine/mongo.py
5. https://github.com/KunjParikh/cmpe256/blob/master/iproject/cosine/kNN.py
6. https://en.wikipedia.org/wiki/Principal_component_analysis
7. https://www.mongodb.com/blog/post/training-machine-learning-models-with-mongodb
8. https://github.com/logicx24/SubredditRecommendationEngine
9. https://neo4j.com/blog/real-time-recommendation-engine-data-science/
10. https://scikit-learn.org/stable/modules/generated/sklearn.neighbors.NearestNeighbors.html
11. Das, A. K., Bhat, N., Guha, S., & Palan, J. (2019). A Personalized Subreddit Recommendation Engine. arXiv preprint arXiv:1905.01263.