News Recommender for JhakaasNewsVala

Team:
Achintya Tripathi(achintyatripathi14@gmail.com),
Akshay Kumar(akshaymanhas20699@gmail.com),
Neetu Singla (neetu.singla12@gmail.com),
Sarbjit Kaur(Kaursarbjit95108@gmail.com)

Objective of Project:

To recommend the news to users of an app created by JhakaasNewsWala. Objective is to build a Hybrid System:-
(i) *The article recommender (Bot 0)*: This bot selects 10 articles to serve a user in such a way that the articles that get served often and ranked higher, have a higher likelihood of being consumed (Inputs to the bot is the corpus of new articles.)
(ii) *The user profiler (Bot 1+ Bot 2)*: Once the user starts consuming news stories, (s)he leaves behind a clickstream and the objective is to increase clickthrough and the frequency with which the user opens the app to consume stories.

Literature Review:

Finding suitable news articles for people based on their preferences and/or profiles is a key challenge for news websites and apps. News recommendation systems traditionally make use of user and content information, but this has become a redundant practice due to the short "shelf life" of news articles. If users do not find articles interesting on the first page, they are less likely to scroll past the home page. Lei et al. (2011) recommended a 'two-level' system that takes into account not only exclusive characteristics of news such as content and popularity, but also user interest, maintaining a good balance between diversity and originality. Other research such as that by Saranya and G. Sudha (2012) presents the use of a hadoop framework to tackle the problem of scalability within news recommender systems. Prior research also makes use of supervised learning methods and Naive Bayes classifiers to determine user ratings for articles, such as in research conducted by Chaturvedi. Liu et al. (2010) make use of a Bayesian framework to make recommendations by predicting users' current interests, which is something we could implement as well.

Brief Introduction:

Various recommendation approaches had been proposed based upon the domains they have been used in or applied to. Approaches are further classified by the type of data used for recommendation (**implicit or explicit**) and lastly by the type of algorithm used for recommendation (**collaborative, content based filtering or hybrid approach**).

The main difference between Content Based Filtering and Collaborative Filtering is that Collaborative Filtering works on preferences of other users (users with similar preferences for some items) to recommend new items whereas Content Based Filtering is not at all concerned with preferences of the other users.

Hybrid methods : In these methods, a combination of two or more recommendation algorithms are used to take or maximize advantage of some techniques and avoid or minimize the drawbacks of another. For example, by combining collaborative filtering methods, where the model fails when new items don't have ratings, with content-based systems, where feature information about the items is available, new items can be recommended more accurately and efficiently

Approach for Data Collection and Analysis (Strategy Used):

Recommender systems use various types of feedbacks from the users to narrow down the users' options to a small number:

- *implicit feedback:* user is not asked to provide them.
- *explicit feedback:* the total number of explicit feedbacks used by the recommender system should be minimum, as users usually don't have much time or don't like to provide feedback. Secondly, feedback asked to the user should be easy for an average user to understand so that he can provide the feedback easily and correctly.

For Bot 0:
As soon as the user comes to the site, provide him with the latest news from various topics -- done using scraping news and clustering these news into various categories and then recommend 2-3 from each category e.g sports, politics,covid etc..

For Bot 1:
Once the articles have been given then the user may search or select a news item from the recommendation. This is taken into account and then we use content based recommendation i.e. LSH using Jaccard similarity where the news selected by the user is given more preference. And the keyworks in that selected document is given more emphisis.

For Bot 2:
We will generate user click data by assuming some statistical distributions for different variables encountered and then to increase the frequency of clicks, we will use item-to-item collaborative filtering based upon co-visitation matrix (users who read this, also read that).

Implementation:

For Bot 0:

After scraping news articles from various news sources such as Reuters, Hindustan Times and FoxNews, we got a corpus of around 2916 news. We applied some preprocessing steps to clean our input text such as tokenization, removal of stop-words and Lemmatization. As the news data extracted is in the form of text and text cannot be used as input in machine learning models, So we need some way that can transform input text into numeric features in a meaningful way. We are using TF-IDF for this purpose. TF-IDF stands for **term frequency-inverse document frequency**. TF-IDF assigns more weight to less frequently occurring words rather than frequently occurring ones. It is based on the assumption that less frequently occurring words are more important.

Formula to calculate tf-idf is: tfidf(t, d, D) = tf(t, d) * idf(t, D), where t is a term (word),d is a document that this term is in and D is a collection of all documents. Clustering is the process of grouping similar items together. Each group, also called as a cluster, contains items that are similar to each other. We clustered news articles into different categories. The dataset consists of 2916 documents. For each article in our dataset, we computed TF-IDF values. Then to find the number of clusters (the value of k) for k-means clustering, we used the elbow method and found that 6 clusters are optimal. Thus our corpus is now grouped into 6 clusters. Now when a new user with no information will enter our app, we will recommend him articles from each cluster sorted by date within each cluster based upon the proportion of articles in each cluster so as to give him/her diverse articles to understand his /her choice and with this approach we can cover the whole corpus.

Summary --

- tokenizing and stemming each news article.
- transforming the corpus into vector space using tf-idf
- clustering the documents using the k-means algorithm
- Recommend articles from each cluster by taking the proportion of articles according to size of articles in each cluster and sorting according to date.

For Bot 1: Content Based Approach(LSH + Jaccard Similarity) -

- LSH is used to perform Nearest Neighbor Searches based on a simple concept of "similarity".
- We say two items are similar if the intersection of their sets is sufficiently large.
- This is the exact same notion of Jaccard Similarity of Sets.
- First we create no. of planes cutting the data points into respective compartments. ** We perform this using no.of Permutations.
- We then map them into a hash table using a minhash function.
- Each new candidate is inserted into the hash table on by one and the table keeps on updating.

- Inside the table for each key we find similarities using Jaccard Similarity.
- What we are doing is that when we hash them in the table and map them in the respective sections we then use HashLSH forest to find the top-k similarities in those respective sections.
- The good part is that we can even find recommendations based on a few words or the whatever text we enter.
- Drawback - if the partitions were not able to group the similar documents into the same categories then they may never be recommended. Also if the hashing is sometimes irregular that is some hash sections may be allocated to some not to any one else thus recommendation may not even be anywhere close to what is expected or asked for this one needs to tune hyperparameter no_of_permutations to optimize for all the docs and the type of corpus.

LSH works better and faster as compared to conventional techniques that used O(n) time to process as compared to LSH which takes only O(log(n)).

Bot 2 Collaborative Filtering: --

User profiler - clickstream data has been generated by following steps:

User_id: generate a list of unique user_ids
Session_id: # of visits of users has been generated from poisson distribution
Article_id : random sample with replacement from total number of articles in corpus
Clicked_or_not: binomial distribution with n=10(total no. at articles recommended) and p=0.5 (prob of success)
Time_spent: three component Mixture gaussian distribution with mean vector (40, 76, 115), S.D. (6, 9, 11) and prior probability for three components as (0.2, 0.5, 0.3)
rating: had been taken as random from 1 to 5 for clicked articles.

final_user_click_data - DataFrame

Index	user_id	session	article_id	clicked_or_not	time_spent	rating
0	1	1	2725	0	0	0
1	1	1	2080	0	0	0
2	1	1	2155	1	126.383	4
3	1	1	2822	0	0	0
4	1	1	966	0	0	0
5	1	1	355	0	0	0
6	1	1	476	1	95.4723	5
7	1	1	2710	0	0	0

And Transforming it into the below table and implementing the rest. --

article_id	0	1	2	3	4	5	6	7	8	9	...	2906	2907	2908	2909	2910	2911	2912	2913	2914
user_id																				
1	0	0	4	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
3	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
5	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0

After this transformation we multiply it with cosine similarity matrix(item-item based) and recommend the suitable articles.

Now that we have a user click stream data we implemented matrix factorization on the data where user rating and articles visited was given more preference. We performed a Item-Item Based recommendation. This helped us to overcome the sparsity problem and thus it depended on the individual user.

Here one matrix is User-Article X Article-Article. We have used cosine similarity to establish a relation between similar articles and then user rating acts as weights when multiplied with the Article-Article matrix.

Example -

```
3 User News Rating Row-User and Col.-Doc Rating
array([[5, 2, 0, 0, 0],
       [5, 0, 0, 1, 0],
       [0, 4, 0, 4, 0]])
```

X

```
Similarity between documents
array([[1. , 0.1, 0.3, 0.3, 0.9],
       [0.2, 1. , 0.8, 0.9, 0.2],
       [0.2, 0.8, 1. , 0.9, 0.2],
       [0.1, 0.9, 0.8, 1. , 0.2],
       [0.8, 0.1, 0.3, 0.3, 1. ]])
```

User Data

Similarity Score

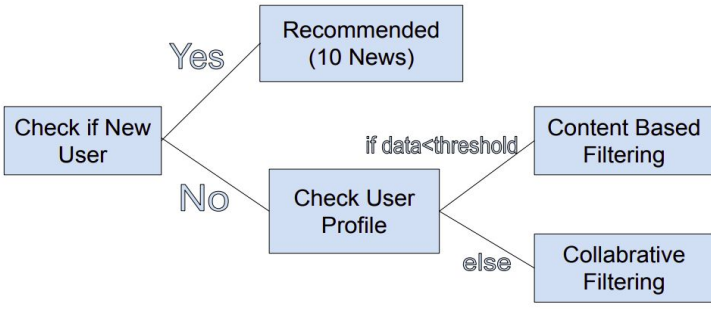
Output

```
User Recommendation -- Example
array([[5.4, 2.5, 3.1, 3.3, 4.9],
       [5.1, 1.4, 2.3, 2.5, 4.7],
       [1.2, 7.6, 6.4, 7.6, 1.6]])
```

So User1 likes article 1, thus article 5 which is similar to the article one, was recommended even when there was no data. Similar for the user users.

Hybrid Recommender System:- Bot 0 + Bot 1 Bot 2

This is how all the bots will be combined and work together in the system.



OutCome and ShortComings -

Bot 0: 10 Recommendation -

news_recommended - DataFrame

Index	article_id	link	text	title	date	cluster
2881	2882	https://in…	FILE PHOTO…	Spain's th…	2020-09-17…	0
2884	2885	https://in…	(Reuters) …	Factbox: T…	2020-09-17…	0
2890	2891	https://ww…	FILE PHOTO…	Oil indust…	2020-09-15…	1
2668	2669	https://ww…	NEW YORK (…	Wall Stree…	2020-09-15…	1
2658	2659	https://ma…	The city o…	Portland's…	2020-09-10…	2
2240	2241	https://ma…	All produc…	Amazon's '…	2020-09-04…	2
2666	2667	https://in…	Chinese Pr…	Chinese Pr…	2020-09-11…	3
2221	2222	http://tec…	If the mea…	OrCam Tech…	2020-09-03…	3
2915	2916	https://in…	MELBOURNE,…	METALS-LME…	2020-09-17…	4
2882	2883	https://ww…	FILE PHOTO…	Australia …	2020-09-17…	4
1842	1843	http://tec…	Few electr…	Hear from …	2020-09-09…	5
2553	2554	http://tec…	Lucas di G…	Hear E-Pri…	2020-09-02…	5

Bot 1 : New recommendation when the user has only selected a few articles or searched for them.

```
Article searched --  Facebook new companies
Recomendation 1 --- UK plans to drop 'Facebook tax', Mail on Sunday says

Recomendation 2 --- What Matters: Russian trolls are back. Here's what you need to know

Recomendation 3 --- Salesforce beats quarterly revenue estimates on online demand

Recomendation 4 --- After restricting a group critical of Thailand's monarchy, Facebook says it will take legal action against the government -

Recomendation 5 --- UK denies it plans to drop tax on digital giants
```

Bot 2: Collaborative filtering one the data we had about the user.

```
Sports events hit by the coronavirus epidemic
Italy's top sports body calls for all events to be canceled until April 3
UPDATE 1-Italy's top sports body calls for all events to be cancelled until April 3
Lambda School raises $74M for its virtual coding school where you pay tuition only after you get a job — TechCrunch
COVID SCIENCE-Modern nursing homes safer in pandemic; virus level in nose, throat may help guide treatment
In China, newly-listed ChiNext shares surge in historic reform
Facebook launches climate science info center amid fake news criticism
Another resignation from Brazil's economy ministry
Breakingviews - Corona Capital: Hockey, HBO, Minority businesses
German economy likely to shrink less this year than during 2009 crisis - IfW
```

So we were able to achieve the primary goals i.e. We were able to recommend news to our users..

Also now that we have enough data about the user, we will also implement User-User Based filtering.

Deployment -

We are working on a webpage where the user will be provided with 10 recommendations as soon as a new user comes and the subsequent parts will take places.Though the algorithms have been finalised and decided but the deployment hasn't been completed yet. We are using Heroku for deployment and almost done with the first part i.e recommendation of 10 articles.

Certain errors and problems are occurring due to less knowledge in this subject matter. However real-time suggestions to the user is currently in the development stage and will be deployed really soon.

Final Overview --

Firstly this project helped all the team members understand how a project is built and how much research work has to be done before starting any project. Secondly we also learned about the challenges faced during actual deployment and what should be the approach while undergoing such a project.

So what were we as a team able to accomplish -

1. We read several articles before finalizing on the algorithms to work on.
2. We were able to recommend new users articles and create a Hybrid System.
3. We were able to implement the algorithms we initially planned.

What we were not able to do or are currently on --

1. Finally deployment - We are currently on this stage as being a new area we are still understanding how to incorporate the website with our model.