San Jose State University

Computer Engineering Department

SJSU SAN JOSÉ STATE UNIVERSITY

CMPE-256: Large Scale Analytics

Patent Recommendation System

Project-Team 3

Instructor: Shih Yu Chang

Authors:

Ruchika Hazariwal

Prachi Chouksey

Pavani Somarouthu

Rachana Bumb

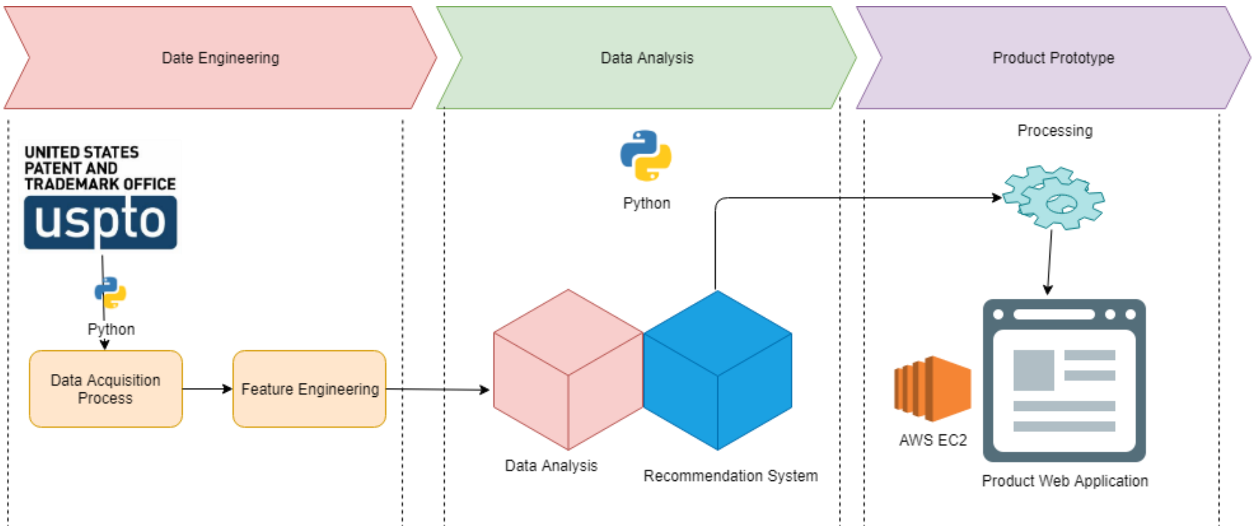
INDEX

Patent Recommendation System

1.1 Abstract: Recommender systems are assembled to help us to easily find the most proper data on the internet. Unlike the search engines recommender systems usher the information to the user without any blue-collar search effort. This is attained by using the similarities between users and/or items. There are innumerable methods to build a recommender system and these methods can be applied to many specific domains like shopping, movies and music. Since each application domain has its own determined needs, the method used for recommendations vary. As a specific application domain, patent recommender systems aim to give the most relevant patent recommendations to users according to their personal interests and preferences. Patent recommendation have specific challenges when compared to the other domains. From the technical point of view there are many different methods to build a recommender system. Thus, while general methods are used in patent recommendation, researchers also need some new methods to make proper patent recommendations. The proposed framework for building automatic recommendations in patents is composed of hybrid module: Content model and item-based collaborative model, and predict a recommendation list. The recommended objects are obtained by using a range of recommendation strategies based mainly on content based filtering and collaborative filtering approaches, each applied separately or in combination.

Index terms- Recommender System, News Recommendation, Content-based, Collaborative Filtering, Hybrid System.

1.2 Introduction: The expanding amount of data on the internet makes harder to find what we are really looking for. Even though the applications like search engines and RS readers help us, it is still difficult to find the useful data we really want to get. Rather, we are not invariably sure about what we want to get. We can only forage for what we know, and we try to find some associations to the new information. But this strategy of finding an item that the user will like highly resides on the serendipity, the heed of the user to inspect the search results and it requires plenty of effort. Still there is a huge possibility that the user could not search for the most satisfactory item for themselves at the end. As people are beginning to read patents online at a high rate, it is becoming a challenge to find the interesting patents. Most of the users spend plenty of time to find a patent based on interest on a single website or they just read the abstract which is not requisite. Patent recommender systems aim to give them the relevant patent recommendations to users according to their personal interests and preferences. The patent domain differs from other domains in many ways. For example; the popularity and recency of patent changes so fast over time. So, focusing on the recency issue becomes more challenging than it is in other domains. Also, some patents may be connected with each other that the user may want to read the previous patents related to the one he/she already reads, or he/she may want to keep informed about. Only user preferences can be an unsatisfactory solution to patent recommendation. This is because the user may want to read a patent when he/she is not really interested in the subject, but she thinks it is important. Recommender Systems or recommendation engines form or work from a specific type of information filtering system technique that attempts to recommend information items that are likely to be of interest to the user. These characteristics may be from the information item (the content based approach) or the user's social environment (the collaborative filtering approach. There are two kinds of collaboration in the system. One is the collaboration between the system and the user; another is the collaboration between the user and user. In this work, we want to examine how well content-based RSs work for recommending patents. In addition, we extend our RS by collaborative filtering.



1.2.1 System Architecture

1.2.2 Dataset:

To analyze, manipulate, clean and evaluate our results on large amount of datasets, we have web scraped patents data from official USPTO website from where legal data scraping is done. Complete Web scraping was done using python's Beautiful Soup module. Our dataset consists of approximately 13000 unique patent IDs with characteristics/features like patent number/ID, abstract, applicant city, applicant country, applicant name, applicant number, applicant state, applicant location, assignee name, cpc, family id, file date, inventors, patent date, title, URL. All these features/characteristics are being used to analyze the results for good recommendations by use of different keywords.

Link from where dataset is being web scraped: https://www.uspto.gov/

1.2.3 Data Preprocessing:

- Redundancy- We make sure redundant data is not present in our dataset while web scraping unique patents from the website.
- Missing Values- Few documents have missing abstracts, we fill in title in the abstract attribute to make sure data is complete.
- Attribute Selection- We have selected the following columns after preprocessing of data.
- Stemming- Generating variants of root/base words.

1.2.4 Data Acquisition Process: Our recommender system will follow prescriptive data analytics which involves high volume of data and advanced/complex analytical techniques to make correct recommendation.

1.3 Approaches:

1.3.1 User based Collaborative approach:

1.3.1.1 Exploratory Data Analysis: This part of the module answers questions like: What is the distribution of how many patents a user interacts with in the dataset? Provides a visual and descriptive statistics to assist with giving a look at the number of times each user interacts with a patent. Findings be like:

a. unique articles that have an interaction with a user.

b. unique articles in the dataset.

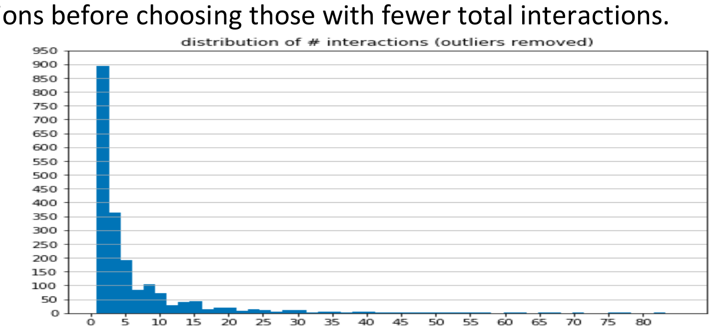
c. unique users in the dataset.

d.user-article interactions in the dataset.

1.3.1.2 User-User based Collaborative recommendation:

Find the most viewed **patent_id**, as well as how often it was viewed. The `email_mapper` function was considered a countable way to map users ID to Patent ids. There were a small number of null values, and it was found that all of these null values likely belonged to a single user. We don't actually have ratings for whether a user liked a patent or not. We only know that a user has interacted with the patent. In these cases, the popularity of a patent can really only be based on how often a patent was interacted with.

- Each **user** should only appear in each **row** once.
- Each patent should only show up in one **column**.
- **If a user has interacted with a patent, then place a 1 where the user-row meets for that patent-column**. It does not matter how many times a user has interacted with the patent, all entries where a user has interacted with the patent should be a 1.
- **If a user has not interacted with an item, then place a zero where the user-row meets for that patent-column**.
- Instead of arbitrarily choosing patents from the user choose patents with the most total interactions before choosing those with fewer total interactions.




```
In [6]: user_interaction_counts.describe()
Out[6]: count     1993.000000
        mean         6.464626
        std         11.040400
        min          1.000000
        25%          1.000000
        50%          3.000000
        75%          7.000000
        max        214.000000
        Name: title, dtype: float64
```

We could however instead recommend based solely on a random subset of the best ranked patents. This also makes a lot of sense for new users since they will likely want to explore the platform, and would probably like to start with some of the more popular patents. The downside of this is that it could potentially skew our recommendation algorithm later on as it would see users who have interacted with all the same patents and think they are similar, but this would only be because they were suggested the same patents to begin with.

1.3.2 Content based:

Patent classification is a problem in today's, information and computer science. As a consequence of exponential growth, great importance has been put on the classification of patents into categories that describe the content of the abstract and body of patents. The function of classifier is to merge patent information into one or more predefined categories based on their content. Each patent can belong to several categories or may be its own category. Very rapid growth in the amount of text data leads to expansion of different automatic methods targeted to improve the speed and efficiency of automated patent classification and recommendation with textual content.

1.3.2.1 KNN and TF-IDF:

The module is designed to enable the classification and measurement of similarity of different and unique patents based on the required patent sample and keywords. It has four basic modules: GUI module, Patents module, Preprocessing module, KNN& TF_IDF module.

GUI module allows user to control the application and the user interaction with recommendation system. The patents module is designed for patent management and selecting data resources from official USPTO website. Patent resources are web-scrapped from USPTO official website. In this module a user can define patent categories, the feature which influences the final results of patent classification. The Preprocessing module checks the patents features, prepares and adjusts them into a format suitable for classification. The documents with a specific format module automatically remove the control characters that might have a negative impact on the result of classification. The main module in the framework is the KNN& TF_IDF module. The module contains the main methods for classification and determination of the patent weight value.

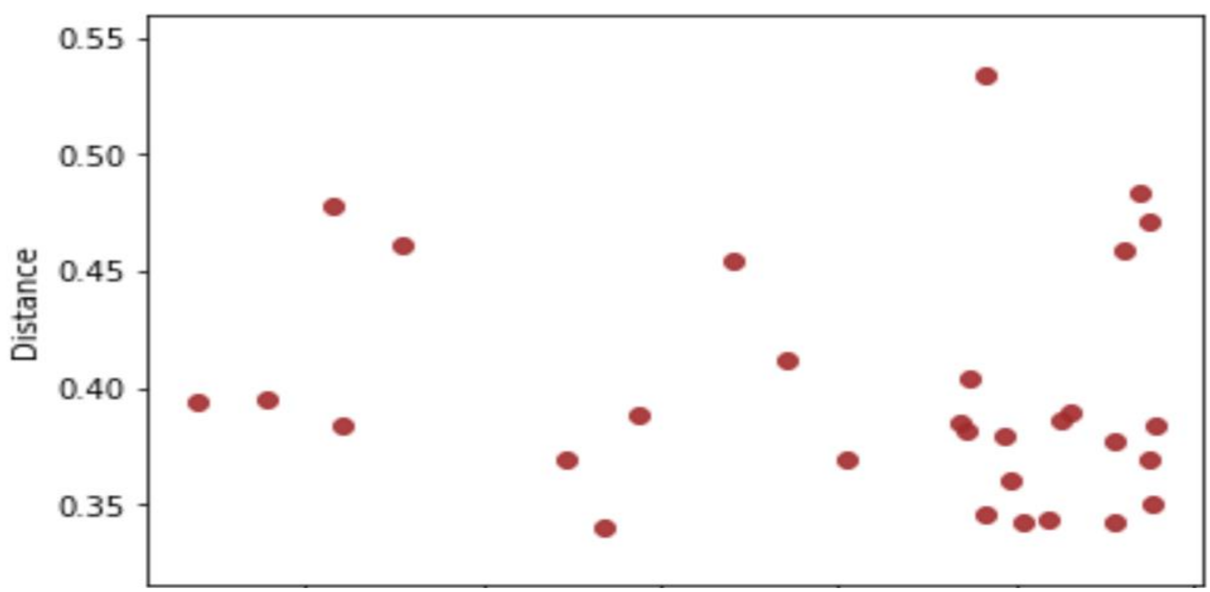
1.3.2.2 KNN Classifier:

Learning process: Learning process head-starts with parsing, stemming, removing stop words the basic abstract which searches the words in patent abstract and forms a vector. Parsing process removes all control characters, spaces between words, dots, commas, and similar characters. The formed vector represents a basic object that will be used for categorization and recommendation of patents.

Determination of the weight matrix: To provide keyword based abstract classification and searching the patents it is necessary to establish the weight matrix. The matrix contains the values of relations between each unique words and patents. It is the initial object in the algorithm to calculate the individual importance (weight) of each searched patent. Each patent is represented as a vector in n-dimensional vector space. Dimension of the matrix is equal to product, the number of different unique keywords and the total number of patents.

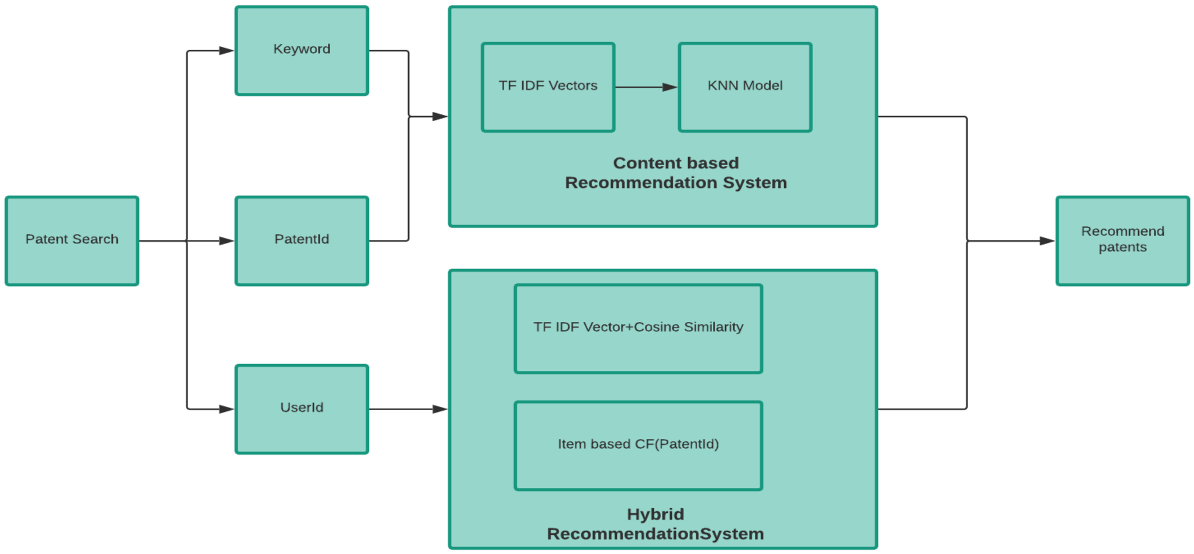
1.3.2.3 Term frequency –inverse document frequency (TF-IDF): Most popular research method used in the modulation of the hybrid algorithm described in this project. TF-IDF method determines the relative frequency of words in a particular patent through an inverse proportion of the word over the complete patent group of documents. In identifying the value, the strategy uses two functions: TF - term frequency of term i in document j and IDF - inverse document frequency of term i. In our research and testing the algorithm of framework this method showed quite good and relevant results.

1.3.2.4 KNN classification process: In the next step of proposed approach it is necessary to determine K value. K value of the KNN algorithm is a factor which shows a required number of patents from the collection which is closest to the selected wanted patent. The classification process determinates the distance of vectors from different patents.



1.3.2.5 KNN plot: Distance vs Patents

1.3.2.6 Framework Evaluation: this section presents the results of framework testing over the sets of 15000 patents from different categories in the learning phase and categorization. Calculation of TF-IDF values in weight matrix has shown as the most demanding part of the implemented algorithm. These TF-IDF vectors are dumped into pickle file.



1.3.2.7 Functional Flow of Content Based and Hybrid Recommendation System

1.3.3 Hybrid Approach: Content plus item based Collaborative recommendation system

In contrast to content-based filtering, a collaborative RS uses the User ID and Patent ID in collaboration to calculate the similarity between different patents.

Vector-based / Cosine-based Similarity: In this algorithm, patents are represented as two vectors that contain the user IDE and Patent ID. The similarity between user ID and Patent ID is calculated by the cosine of the angle between the two vectors. Database of approximately 20 users was created and stored as CSV file. Matrix of vectors is generated with rows and columns as User ID and Patent ID to know which user has read which patent of particular patent ID. Number represented in a row is matched to the Patent ID.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1		10349422	10349418	10349318	10349295	10349285	10349227	10349218	10349197	10348985	10348514	10348394	10348341	10348329	10348108	10347985	10347579	10347565	10347344	10347329	10347290
2	10349422	1	0.20611042	0.13420136	0.16896043	0.11372347	0.11297744	0.08209348	0.0303789	0.00422363	0.02082916	0.18004246	0.05114671	0.01348129	0.01054402	0.04733973	0.00675404	0.02520546	0.01404876	0.01172937	0.04270574
3	10349418	0.20611042	1	0.11721271	0.17904791	0.15421714	0.14600898	0.12422746	0.17871952	0.00762415	0.02299453	0.18209461	0.08667758	0.02958937	0.08096596	0.08115517	0.01681503	0.17961375	0.02313877	0.02639476	0.02433532
4	10349318	0.13420136	0.11721271	1	0.12835952	0.12243282	0.08640746	0.13524956	0.03788909	0.02231292	0.01085263	0.04396496	0.05770066	0.05258314	0.00230623	0.01967352	0.00583325	0.02356336	0.01741118	0.01147762	0.03159763
5	10349295	0.16896043	0.17904791	0.12835952	1	0.06939229	0.08706631	0.17525095	0.01156507	0.0095217	0.02472869	0.26652819	0.04818038	0.0331501	0.02254918	0.02957931	0.01437011	0	0.03370668	0.0209357	0.03964243
6	10349285	0.11372347	0.15421714	0.12243282	0.06939229	1	0.09025328	0.1762558	0.03391568	0.01582374	0.02867733	0.0939894	0.03991006	0.03187945	0.01621297	0.03312049	0.00830385	0.02942052	0.02902309	0.04104166	0.04494421
7	10349227	0.11297744	0.14600898	0.08640746	0.08706631	0.09025328	1	0.06254838	0.05991076	0.01405549	0.07853336	0.11317867	0.07538113	0	0.13690744	0.06403481	0.01286155	0.00159933	0.01495141	0.03565626	0.07605407
8	10349218	0.08209348	0.12422746	0.13524956	0.17525095	0.1762558	0.06254838	1	0.02341759	0.00599154	0.00918427	0.05958797	0.06008215	0.01145911	0.12163681	0.00556273	0.01779946	0.07576264	0.01438136	0.0292924	0.0209192
9	10349197	0.0303789	0.17871952	0.03788909	0.01156507	0.03391568	0.05991076	0.02341759	1	0.00124897	0.00661131	0.01508077	0.06931527	0.00925394	0.16854591	0.01892586	0.01667943	0.00783484	0.00748817	0.00530781	0.01630215
10	10348985	0.00422363	0.00762415	0.02231292	0.0095217	0.01582374	0.01405549	0.00599154	0.00124897	1	0.12078328	0.01365149	0.0045989	0.00866153	0.00574075	0.00204793	0.01747805	0.00854662	0.0082204	0	0.0013351
11	10348514	0.02082916	0.02299453	0.01085263	0.02472869	0.02867733	0.07853336	0.00918427	0.00661131	0.12078328	1	0.02468624	0.05387367	0	0.03049589	0.03223664	0.03706067	0.00350058	0.01639975	0.02128688	0.04943545
12	10348394	0.18004246	0.18209461	0.04396496	0.26652819	0.0939894	0.11317867	0.05958797	0.01508077	0.01365149	0.02468624	1	0.0225564	0.00466243	0.00599671	0.10293621	0.03386388	0.01836188	0.02169711	0.0234121	0.02859425
13	10348341	0.05114671	0.08667758	0.05770066	0.04818038	0.03991006	0.07538113	0.06008215	0.06931527	0.0045989	0.05387367	0.0225564	1	0.0086878	0.02206972	0.01803727	0.00910571	0.03241788	0.0298872	0.04409845	0.02811958
14	10348329	0.01348129	0.02958937	0.05258314	0.0331501	0.03187945	0	0.01145911	0.00925394	0.00866153	0	0.00466243	0.0086878	1	0.00953297	0	0.0117466	0	0.03037885	0	0.00242653
15	10348108	0.01054402	0.08096596	0.00230623	0.02254918	0.01621297	0.13690744	0.12163681	0.16854591	0.00574075	0.03049589	0.00599671	0.02206972	0.00953297	1	0.0657661	0.00867746	0.11211251	0.00587565	0.01683799	0.03290464
16	10347985	0.04733973	0.08115517	0.01967352	0.02957931	0.03312049	0.06403481	0.00556273	0.01892586	0.00204793	0.03223664	0.10293621	0.01803727	0	0.0657661	1	0.12793784	0.02600836	0.00871658	0.01255097	0.02512806
17	10347579	0.00675404	0.01681503	0.00583325	0.01437011	0.00830385	0.01286155	0.01779946	0.01667943	0.01747805	0.03706067	0.03386388	0.00910571	0.0117466	0.00867746	0.12793784	1	0.01447656	0.0019529	0.0176369	0.00581798
18	10347565	0.02520546	0.17961375	0.02356336	0	0.02942052	0.00159933	0.07576264	0.00783484	0.00854662	0.00350058	0.01836188	0.03241788	0	0.11211251	0.02600836	0.01447656	1	0.01577895	0.00141341	0
19	10347344	0.01404876	0.02313877	0.01741118	0.03370668	0.02902309	0.01495141	0.01438136	0.00748817	0.0082204	0.01639975	0.02169711	0.0298872	0.03037885	0.00587565	0.00871658	0.0019529	0.01577895	1	0.14788842	0.01781601
20	10347329	0.01172937	0.02639476	0.01147762	0.0209357	0.04104166	0.03565626	0.0292924	0.00530781	0	0.02128688	0.0234121	0.04409845	0	0.01683799	0.01255097	0.0176369	0.00141341	0.14788842	1	0.02457856
21	10347290	0.04270574	0.02433532	0.03159763	0.03964243	0.04494421	0.07605407	0.0209192	0.01630215	0.0013351	0.04943545	0.02859425	0.02811958	0.00242653	0.03290464	0.02512806	0.00581798	0	0.01781601	0.02457856	1
22	10347265	0.0205777	0.14824748	0.0136764	0.01023096	0.02158576	0.01152105	0.0187648	0.30496721	0.01007927	0.01673953	0.0153761	0.18862547	0.00464859	0.0096976	0.01278912	0.00446068	0.23663903	0.00683468	0.00320046	0.00627896
23	10347139	0.01307438	0.01289892	0.01240582	0.01928937	0.07044482	0.00724179	0	0.00728541	0.00939737	0.00687649	0.02200794	0.01328375	0.02683578	0.00194143	0.00188258	0.02786016	0	0.0330902	0.00663044	0.00477143
24	10347063	0.02070449	0.01570261	0.02397164	0.00546015	0.01423842	0.02764528	0.0266742	0.01340768	0.00317599	0.01208507	0.00126278	0.03849544	0.00392151	0.01192603	0.01898595	0.00625053	0.0004633	0.01519412	0.02856505	0.01249318
25	10347026	0	0.00974688	0.02566673	0	0.00971276	0.0338277	0	0.01873854	0.08474821	0.06632967	0.01164441	0	0.02319027	0	0.00229109	0.0022795	0.00234502	0.00798496	0.00302422	0.00385917
26	10347007	0.0293877	0.01477173	0.02661293	0.02808504	0.02149615	0.07431017	0.04472225	0.10124495	0.10588735	0.03457298	0.03589472	0.02432972	0.01816431	0.1107952	0.03270913	0.03069069	0.04364835	0.01478142	0.00529376	0.04852209
27	10346958	0.00926839	0.00283108	0.0107438	0.01232694	0.05215606	0.01327857	0.01653624	0.0037172	0.11529602	0.07071882	0.03585555	0.00696977	0	0.01310642	0.00152149	0.00823911	0.0015573	0.02093107	0.01445755	0.00135534
28	10346888	0	0.02233747	0.07329897	0.05145757	0.18637096	0.11565764	0.03204313	0.00546966	0.0049403	0.00435218	0.00413139	0.01200126	0.02346539	0.00257633	0.00249822	0.00290154	0	0.00824797	0.00439938	0.04645631
29	10346794	0.0042526	0.00774589	0.01382488	0.01658444	0.02204425	0.01294138	0.02680625	0.01096104	0.00155049	0.01619688	0.0171686	0.00622959	0.01398237	0.01569446	0.00355816	0.0083198	0	0.03050059	0.02732239	0.0517482
30	10346765	0.01759332	0.03007469	0.12965967	0.03192231	0.24170313	0.24870097	0.04474047	0.0089593	0.01009073	0.01334518	0.04579451	0.03342406	0.02983103	0.01611628	0.00588383	0.00524279	0.00293065	0.03577206	0.05704874	0.00542334
31	10346720	0.00609044	0.00894573	0.00701435	0.02514617	0.00383552	0.02105171	0.01910766	0.00153	0.03418965	0.02443818	0.006589	0.00293693	0.0161962	0	0.00958632	0.0428697	0	0.0170348	0.00489972	0.00791528
32	10346649	0.02729346	0.06770838	0.03188811	0.03480981	0.0374226	0.03865551	0.07814734	0.14329488	0.00622301	0.01014506	0.04561806	0.04120666	0.05034419	0.06505252	0.01034635	0.00158687	0	0.09254542	0.07795427	0.07833113
33	10346624	0.00570329	0.01875116	0.02832185	0.02603392	0.01367073	0.05106401	0.01327453	0.02999517	0.055806	0.08068889	0.01336585	0.06611113	0.01299922	0.01860145	0.02297618	0.01561781	0.00285389	0.00817343	0.0215921	0.0099481
34	10346617	0.02453976	0.01377475	0.0352609	0.00783193	0.03643179	0.02125904	0.01656674	0.01145822	0.00644435	0.0096211	0.05286468	0.01877346	0.01793129	0.01281816	0.01245345	0.06043835	0	0.018258	0.04344426	0.02136027

1.3.3.1 Similarity Matrix

As combination strategy, we use the weighted hybrid strategy as explained in. For our first version, we highly decided to weight both components: content plus collaborative equally. The content-based component is important for recommending new patents even if the user has not read any patent before. Additionally, the content-based system is able to provide content to users with special interests as well. But it comes up that the collaborative filtering part is as important as the content-based component.

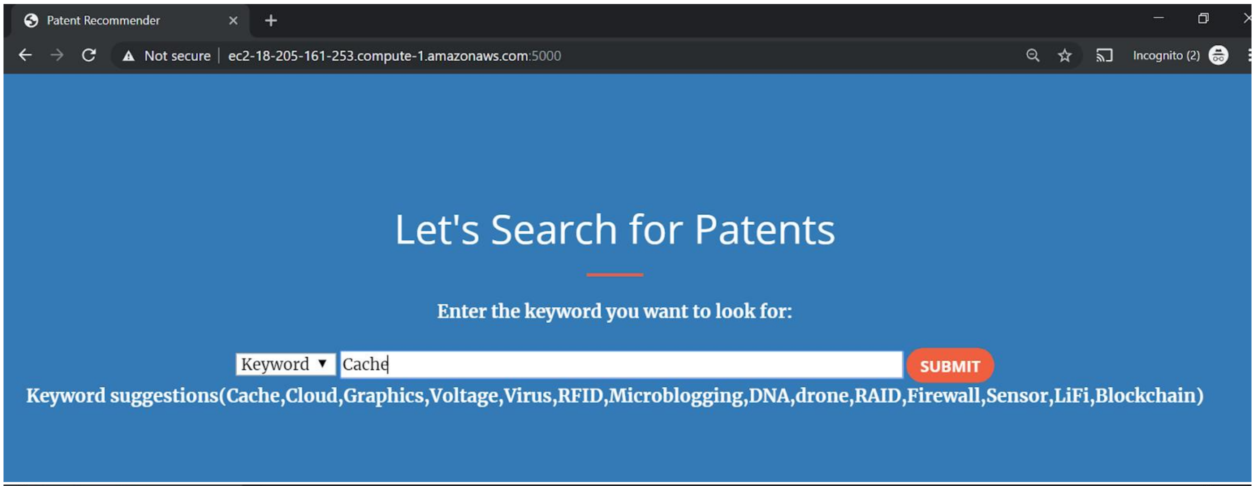
We tested the hybrid recommendations with a selected user group. We paid attention to choose users with unique user ID with distinct interest of category of patents to ensure and protect comparability to our users.

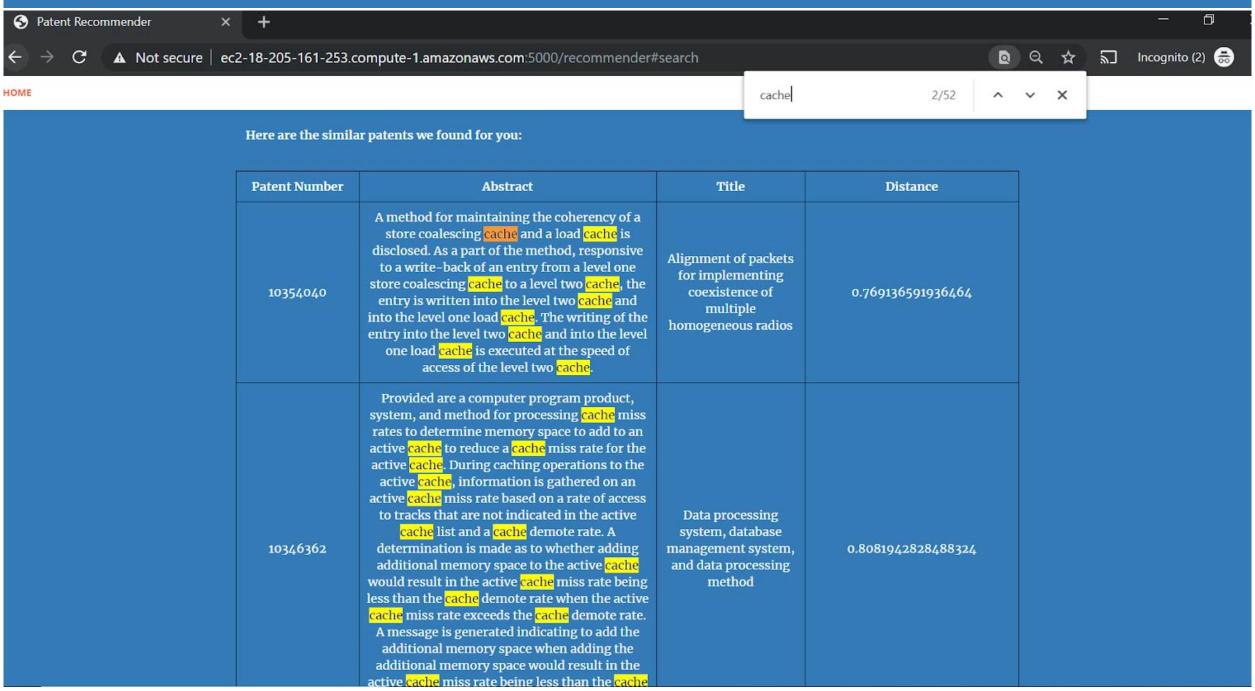
1.4 UI Design/ Prototype:

Micro web framework called Flask is used to integrate Python code with UI. It does not require an exceptional database layer, hence all and everything is added to flask as an extension and can be used in an integrated way. This UI/ prototype where user can interact with the system to get patent recommendations consists of 2 search boxes:

- First search box is a dropdown consisting of 3 different type of selections: keyword, User ID and Patent ID
- Second search box is to type the respective keyword, user id or patent id as per the selection in first search box.

After filling in the 2 search boxes and hitting the “search” button, top 10 relevant patent recommendations get displayed on same web page.





1.5 Conclusion: In this project, while trying 3 different approaches to recommend patents to users: Content Based recommendation system, hybrid: Content based plus item based collaborative recommendation system and user based collaborative recommendation system, the performance and execution time of hybrid approach was the best as compared to other two recommendation system because of use of KNN, TF-IDF and relating items to users for better recommendation with highest efficiency.

1.6 Future Research Directions:

As this proposed strategy focuses on putting up explanations against patent hybrid recommendations, there are distinct areas of explanations in which there is place for improvement.

- Explanations have proved valid results in Hybrid Content-based recommender systems. It provides transparency. It has been keenly noticed that users like and feel more confident in recommendations looked upon as efficient and transparent. But, a lot of effort is needed to investigate different mechanisms to attain recommendation system transparency. Specifically, critical challenges are to obtain more meaningful explanations from distinct computational functions and modules that would be useful to pop-up the trust in recommender systems. A patent's features have been used by researchers to provide explanations against recommendations. Therefore, in order to have a sufficient set of features, the content must either be in a form whose features can be extracted automatically by a computer (e.g., text).
- Research and development on automated feature/characteristics extraction methods are required. Researchers have been working on creating accurate and exact recommendations. However, featuring these accurate recommendations with explanations iwhich attracts the user more effectively is a crucial problem.

1.7 References:

- https://www.uspto.gov/patents-application-process/search-patents
- https://pandas.pydata.org
- https://en.wikipedia.org/wiki/Tf%E2%80%93idf
- https://en.wikipedia.org/wiki/Recommender_system