CSC - 501

REPORT (Final Project)

Heterogenous Dataset

SUBMITTED BY - GROUP B'''

Kapoor, Nikita	V00949246
Malik, Mona	V00935224
Sannath Reddy, Vemula	V00949217

Submitted To - Prof. Sean Chester

schester@uvic.ca

SECTION – 1

(a) Introduction:

We are given the data of stack exchange which tells about posts, users, comments, votes, related posts, changes made to a post and tags. For insights from the given data, we have worked mostly on posts, post links, users, tags and votes XML files. We have formulated 3 insights to work on with the given data.

- **First insight - Finding the relevant post to a post based on its title**.
  The files used are post links and posts to get the results. To fetch the most relevant post of a selected post, the post with most similarity measure was considered which was done using adjacent nodes concept from graph theory and then by applying TF-IDF similarity measure. The final output is represented as knowledge graph and also as a web page.

- **Second insight - Recommend posts to a user**
  Based on the posts he/she previously has answered or questioned, post is recommended.
  The files used are users and posts to get the results. Initially, we fetch the similar users based on 'AboutMe' column from users file and then compare the posts of similar users on which they were active to look for the similar posts that can be recommended to the user who did not know about it. This insight is in nature of the **suggestion/recommendation feature** that web-sites like YouTube offers.

- **Third insight - Cluster the strongly connected users** and analyze how the cluster is connected to another cluster through a weak link. The files used are tags.xml and users.xml. Initially, we fetched all tags used by users in different posts. Based on the tags used, we found similarity of users using **cosine similarity**. We create a heatmap of users showing how each user is interlinked to another. The same is represented in neo4j as graph with clusters of strongly connected users.

The report is about these three insights and visualizations supporting them. We worked mostly on how posts are linked (**graph analysis**) and how similar users/posts are linked (**text analysis**).

Tasks like retrieving the User Display Name for a post required us to use the concepts of relational database for mapping ids with actual values.

Data Pre-processing:

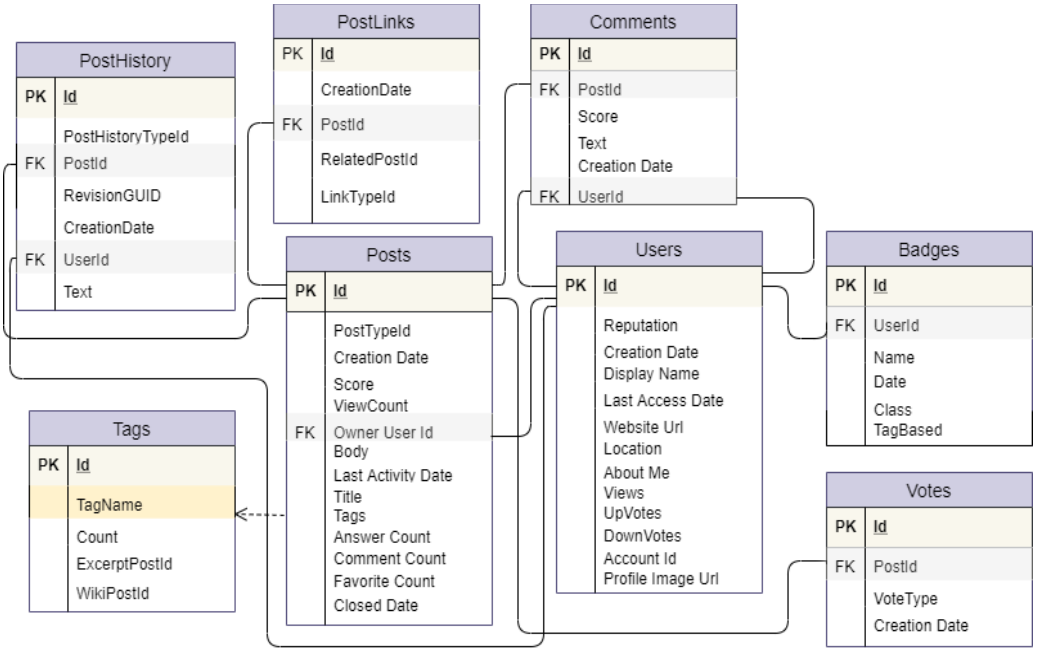
Text Data Pre-Processing - Dealing with columns with text data like 'AboutMe' , 'Tags' , we cleaned and pre-processed the data by removing hyperlinks, stop-words, punctuations and everything which had could not affect the text analysis.

Also, as the files of given dataset were in XML which are heavy-weighted, considering XMLs in every python file would not be efficient approach in terms of memory and time. So, before starting our work on insights, we have converted XMLs to CSVs which is much comfortable format to read the data and work on.

(Data Modelling) For the above insights, we have reused the code from assignments for data modelling (like adjacency list) which were generic and good enough for the dataset.

(b) ER Diagram:

Below is Entity-Relationship diagram that gives the relation between every entity in the dataset with other entities.



SECTION – 2

(a) Insights:

Insight – 1:

(a) About the Insight : Insight is to improvise/add feature that fetches the most relevant post for a particular post merging the concepts of graph and text analysis.

We are given posts and their related post details in posts.csv. The insight is to check that a post having more than one related post is selected, whether we can get better results for fetching a post which is most similar to the selected post using td-idf on title column.

Id	CreationD	PostId	RelatedPo	LinkTypeId
22791	2015-05-1	5826	334	1
22792	2015-05-1	5826	313	1
22793	2015-05-1	5826	14	1
22794	2015-05-1	5826	159	1
22795	2015-05-1	5826	234	1
22796	2015-05-1	5826	808	1
22797	2015-05-1	5826	739	1
22798	2015-05-1	5826	1216	1

Implementing the concepts of graph and text on posts like the one highlighted above to get the most relevant post among all the related posts.

(b) Synthesis: Implementation of Course Concepts, design choices

(Adjacency List , Triple Store Format, Knowledge Graph, TF-IDF)

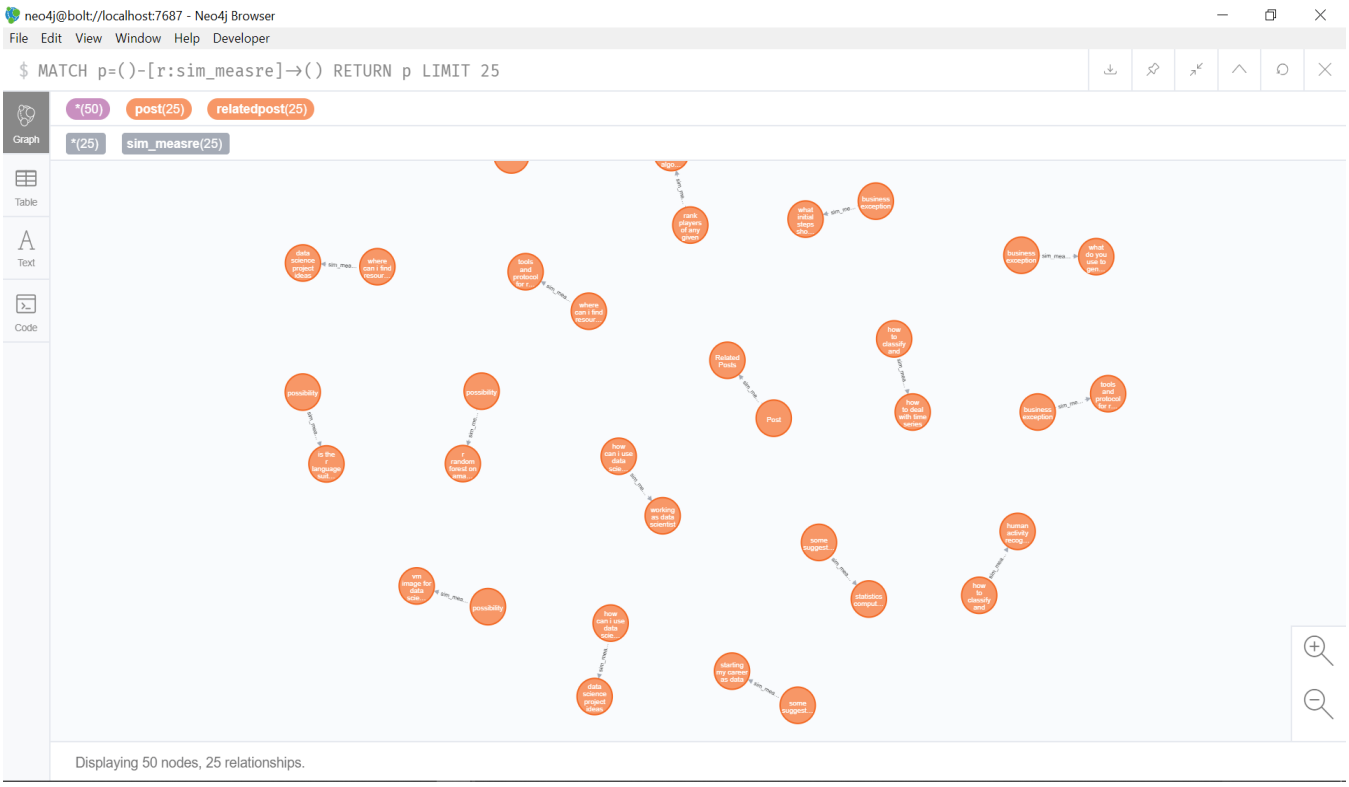
- Considering related posts as adjacent nodes of the posts, we have created an adjacency list out of post.xml.
- For a post, computing the similarity measure of it with every related post using TF-IDF vectorization.
- Below is the csv with relation : post - similarity measure -> related post.

Representing the relations as Triple store format to check for similarity measures and number of adjacent nodes of each node.

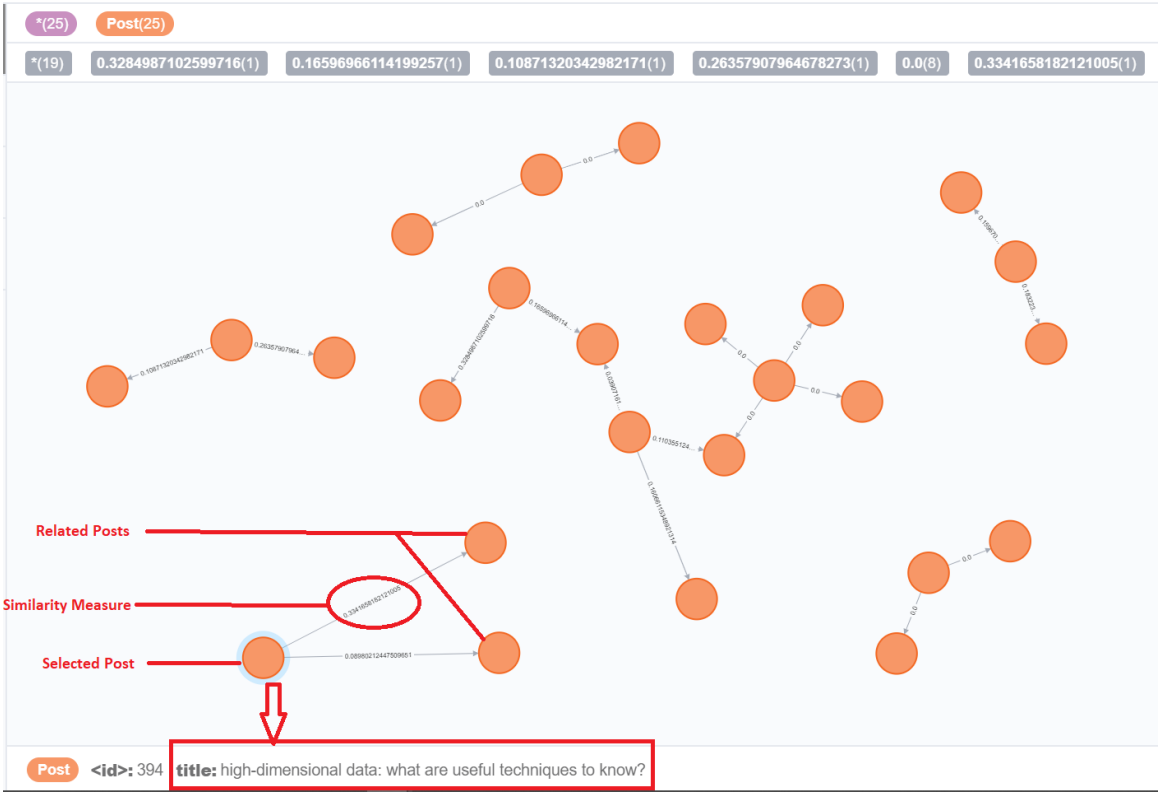
	A	B	C
	SUBJECT	PREDICATE	OBJECT
	Post	Similarity Measur	Related Posts
	some suggestion for career in data science or predictive modeling	0.165969661	starting my career as data scientist is software engineering experience required
	some suggestion for career in data science or predictive modeling	0.32849871	statistics computer science data science
	how to classify and cluster this time series data	0.108713203	human activity recognition using smartphone data set problem
	how to classify and cluster this time series data	0.26357908	how to deal with time series which change in seasonality or other patterns
	high-dimensional data: what are useful techniques to know?	0.089802124	solving a system of equations with sparse data
	high-dimensional data: what are useful techniques to know?	0.334165818	nearest neighbors search for very high dimensional data
	graph layout for a network of molecules	0.183223408	visualizing a graph with a million vertices
	graph layout for a network of molecules	0.159670582	visualizing deep neural network training
	where can i find resources and papers regarding data science in the area of public health	0.039071613	starting my career as data scientist is software engineering experience required
	where can i find resources and papers regarding data science in the area of public health	0.110355125	tools and protocol for reproducible data science using python
	where can i find resources and papers regarding data science in the area of public health	0.160661153	data science project ideas
	possibility of working on kddcup data in local system	0.105130347	is the r language suitable for big data
	possibility of working on kddcup data in local system	0	r random forest on amazon ec2 error cannot allocate vector of size 5 4 gb
	possibility of working on kddcup data in local system	0.092437581	vm image for data science projects

Cypher Query:

```
load csv from 'file:///SimilarityMeasure_TripleStore.csv' as tsf
create ( p : post  { title : tsf[0] } ) - [ s : sim_measre { similarity : tsf [1] } ] -> ( rp : relatedpost { title : tsf[2] } )
```



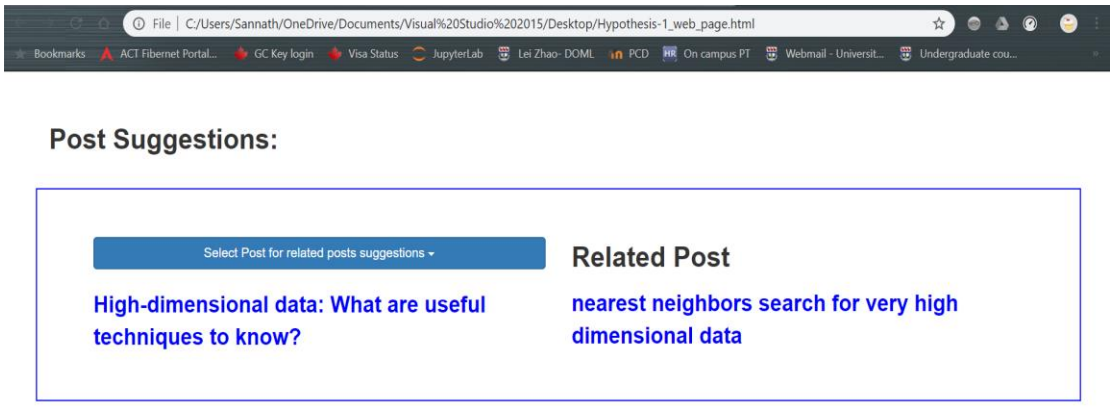
Knowledge graph from the above csv in neo4j:



(c) Transforming Raw Data to Insights: Visualizations

Posts	RelatedPost
Some suggestion for career in data science or predictive modeling	statistics computer science data science
How to classify and cluster this time series data	how to deal with time series which change in seasonality or other patterns
Business exception reporting	tools and protocol for reproducible data science using python
Web Framework Built for Recommendations	organized processes to clean data
High-dimensional data: What are useful techniques to know?	nearest neighbors search for very high dimensional data
Rank players of any given sport	can machine learning algorithms predict sports scores or plays

Representing the csv as a web page with posts populated in dropdown and displaying the related post next to it, we have the following output:



Insight – 2:

(a) About the Insight : Trying to recommend posts to similar users (similar on the basis of their about me), improvising suggestion/recommendation feature that fetches the most relevant posts to a particular user merging concept of text and relational data. We are given about me description of a user in users.xml. We found similarity of users on the basis of their about me using tf-idf concept. Further we use these similarities to confirm that similar profile user is active on same concepts using tags and finally suggest posts to similar nature users on the basis of their active posts.

These filtrations for finding similar users based on 'AboutMe' of user and 'Tags' of posts in which user is active, confirm that appropriate posts are recommended to the correct user.

(b) Synthesis: Implementation of Course Concepts, design choices

(TF-IDF, Mapping UserID's, PostID's– Relational data)

Systematic Synthesis Steps are as below:

Step – 1: For user, computing the similarity measure of it with another user using TF-IDF vectorization (using about me column of user.xml) and selecting most similar user for a user.

Below is the data fetched that represents a user with his/her about me and most similar user.

	index	_Id	_AboutMe	AboutMe_C	most_sim_users
0	0	-1	Hi, I'm not really a person. \r\n\r\n ...	hi i am not really a person i am a backgro...	35592
1	1	1	\r\n\r\n Developer at Stack Overflow focusin...	developer at stack overflow focusing on publi...	13425
2	2	2	Developer on the Stack Overflow team. Find...	developer on the stack overflow team find me...	13425
3	3	3	I'm a student interested in technology. ...	i am a student interested in technology also...	18781
4	4	4	I work with <a href="https://olo.com" rel="...	i work with olo solving problems for people wh...	30

Step – 2: In the second step, we used our results from users.csv of most similar user pairs to mapped with post.csv to find the tags of similar users and then finding their active posts.

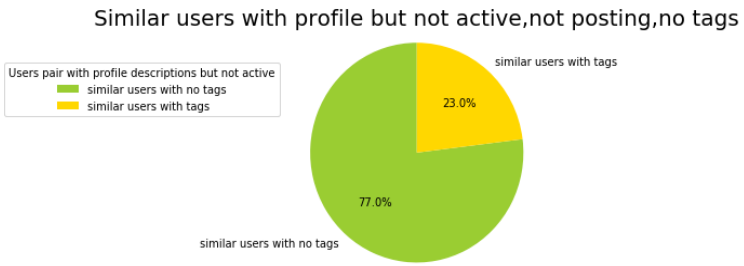
We considered Tags to group users , as its important to analyze whether similar users with same profile(about me) are viewing stack exchange for the similar posts/content.

Below data is fetched by making relations between tags of similar users and their active posts.

	Similar_Users	User1_id	User1_post	user1_Tags	user2_id	User2_post	user2_Tags
0	(5, 41808)	5	[5]	machine-learning	41808	none	none
1	(11, 26237)	11	[33025]	machine-learning time-series data-science-model	26237	none	none
2	(34, 31386)	34	[26631]	deep-learning reinforcement-learning correlation	31386	none	none
3	(36, 10841)	36	[7, 1113]	education open-source bigdata clustering text-...	10841	none	none
4	(38, 10088)	38	none	none	10088	[56129]	scikit-learn learning-to-rank
5	(43, 32164)	43	none	none	32164	[18956]	python pandas categorical-data

Also, we analyzed that many similar users have active account and description but no posts.

This was done by checking the tags column, if a user has any tags implies the user has some active posts else the user doesn't have any posts.



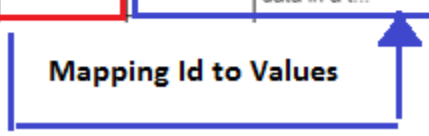
Users with no posts have no tags and that is what the pie chart above says, 23% of similar users have posts which they posted/viewed/answered and 77% of the users with no active posts. We have considered 23% of simiar users for next step to achieve the task of recommending posts to similar user.

Next, we further filter the similar user by implementing TF-IDF concept on their tags so that now we have similar user based on about me and tags(that is users with same profile and same interest in posts.)

	index	Similar_Users	User1_id	User1_post	user1_Tags	user2_id	User2_post	user2_Tags	new_col
2	10	(75, 33176)	75	[5933]	machine-learning data-mining dataset	33176	[19546]	dataset statistics	similar users based on aboutme and tags
8	136	(1264, 9966)	1264	[10165]	random-forest feature-extraction categorical-d...	9966	[6004]	data-mining text-mining feature-selection feat...	similar users based on aboutme and tags

Step – 3: Further we used the above results and use active posts related to a user to suggest post to similar user. Additionally, we use user_id to get user display name and their respective post_id to get the title of the posts.

	User1_id	User1_post	user2_id	User2_post	U1_Name	PostTitles_user1	U2_Name	PostTitles_user2
0	75	[5933]	33176	[19546]	Shagun Sodhani	[What possible data products can be built usin...	tejal567	[How to reduce an unknown size data into a fix...
1	1264	[10165]	9966	[6004]	prooffreader	[What's the best way to use binned data in a t...	untitledprogrammer	[How to select features from text data?]

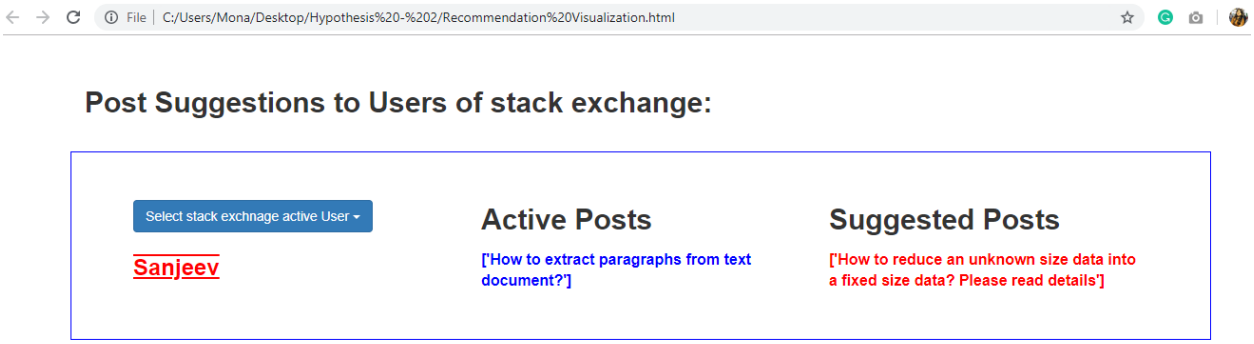


Final Result of the above processing , used for post suggestions to users.

	U1_Name	PostTitles_user1	U2_Name	PostTitles_user2
0	Shagun Sodhani	[What possible data products can be built usin...	tejal567	[How to reduce an unknown size data into a fix...
1	prooffreader	[What's the best way to use binned data in a t...	untitledprogrammer	[How to select features from text data?]
2	Minu	[Visualization using D3, Simple Excel Question...	Pranav Pandey	[How to know for sure if we can learn from a g...

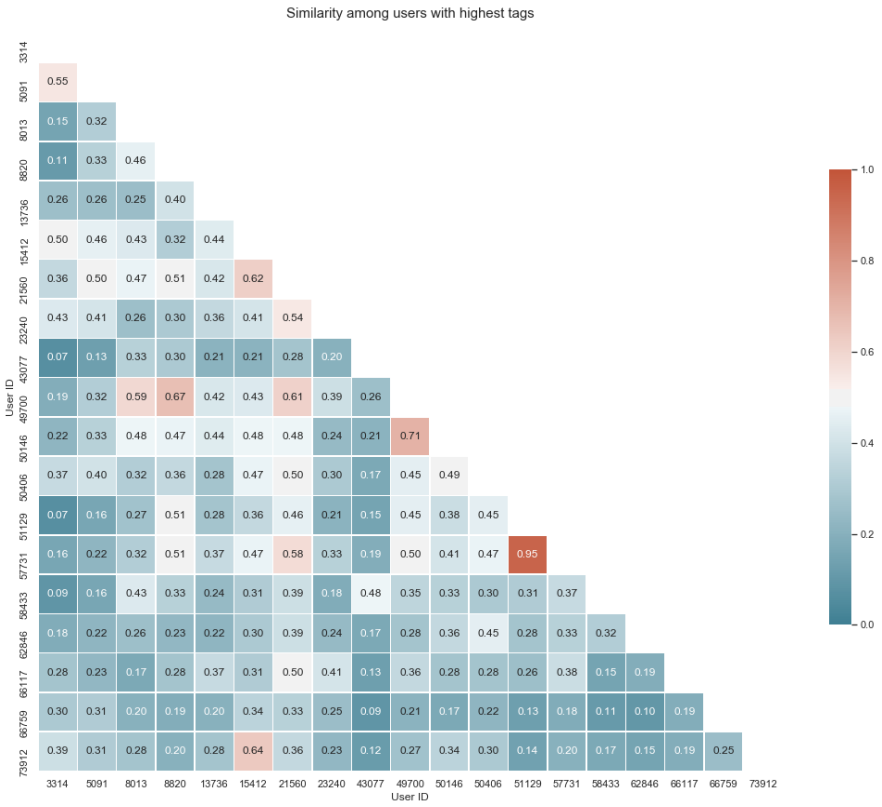
(c) Transforming Raw Data to Insights: Visualizations:

Representing the final output csv as a web page with user populated in dropdown and displaying the respective user active posts and post suggested to him.



Insight – 3 :

(a) About the Insight : The insight is about how users are connected to each other representing the strongly connected users as clusters.



Similar users for this insight are fetched based on the tags of the posts which a user has viewed/posted/answered. This graph shows similarity among users having highest number of tags in the posts with score 1.0 being highest similarity score and score 0 being the lowest score.

(b) Synthesis: Implementation of Course Concepts, design choices

(Cosine Similarity using TF-IDF, Clustering)

Document similarity: We have appended all the tags used by a user into a string and calculated cosine similarity of every user with the other based on the “tags” string. The tags string is converted into TF-IDF vector using TfidfVectorizer since two documents are similar if their vectors are similar.

There was no need of normalizing the data such as removing stop-words, emoticons, punctuations as the string contains tags e.g. <machine learning> <data science>. Only a regex is applied to fetch words from the <> tags.

Why cosine similarity?

Cosine similarity measures the cosine of angle between 2 vectors i.e., Cosine gives the similarity measure between 2 vectors ignoring the scaling factor. For instance, taking movies and ratings data, if a user1 has rated (1.0, 2.0, 3.0, 4.0) for 4 different movies and user2 has rated same movies as (0.11,0.21,0.31,0.41). Here both the user’s style of rating the movies was same even though the actual ratings are different. Such vectors can only be treated similar if cosine similarity is considered for calculating the similarity.

Working on the data, we have done the below steps:

1. All possible pairs of users have been created based on tags
2. The cosine similarity is calculated for each of these pairs (To compute cosine similarity, we take dot product of feature vector of both users).
3. Grouped the users who have more similarity measure as strongly connected users and users with low similarity as weakly connected users.
4. Represented the strongly connected grouped users as clusters in neo4j and also the relation between one cluster and another through a weak link.

Out of the whole data, we found the most and least similar users having score 0.948778 and 0.0 respectively

```
[294]: # for most and least similar users but user id have to be mapped with "flag"
       print("Most Similar Users")
       df_similarity.loc[[df_similarity.similarity.values.argmax()]]
```

```
Most Similar Users
```

[294]:

	pair	similarity
5931	(79, 86)	0.948778

```
[295]: # for most and least similar users but user id have to be mapped with "flag"
       print("Least Similar Users")
       df_similarity.loc[[df_similarity.similarity.values.argmin()]]
```

```
Least Similar Users
```

[295]:

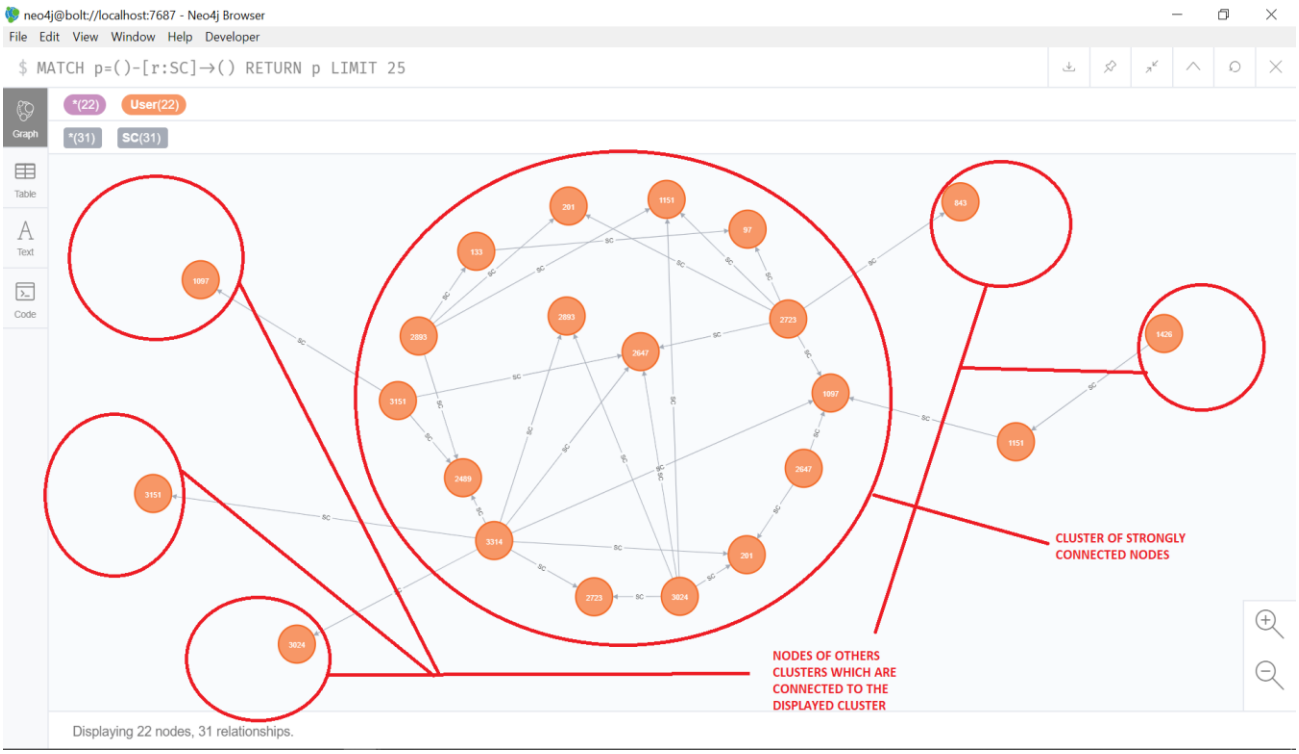
	pair	similarity
51	(0, 52)	0.0

Considering the similarity value between users as relation between them, we have shown the users who are strongly connected and weakly connected. The threshold for dividing users based on similarity was taken as 0.2. Any pair of users with less than 0.2 cosine similarity is considered as weakly connected pair and the pair of users with cosine similarity above the threshold are strongly connected pair. We have the below CSV after segregating strongly-connected nodes and weakly-connected nodes of user.

	A	B	C
1	USER	WEAKLY CONNECTED USERS	STRONGLY CONNECTED USERS
2	0	[]	[97, 122, 133, 201, 843, 1097, 1151, 1426, 2489, 2647, 2723, 2893, 3024, 3151, 3314, 3436, 5043, 5091, 6514, 7812, 7966, 8013, 8432, 8517, 8820, 9793, 10522
3	84	[]	[84, 122, 133, 201, 843, 1097, 1151, 1426, 2489, 2647, 2723, 2893, 3024, 3151, 3314, 3436, 5043, 5091, 6514, 7812, 7966, 8013, 8432, 8517, 8820, 9793, 10522
4	97	[]	[84, 97, 133, 201, 843, 1097, 1151, 1426, 2489, 2647, 2723, 2893, 3024, 3151, 3314, 3436, 5043, 5091, 6514, 7812, 7966, 8013, 8432, 8517, 8820, 9793, 10522,
5	122	[]	[84, 97, 122, 133, 201, 843, 1097, 1151, 1426, 2489, 2647, 2723, 2893, 3024, 3151, 3314, 3436, 5043, 5091, 6514, 7812, 7966, 8013, 8432, 8517, 8820, 9793, 10522,
6	133	[97]	[84, 122, 201, 843, 1097, 1151, 1426, 2489, 2647, 2723, 2893, 3024, 3151, 3314, 3436, 5043, 5091, 6514, 7812, 7966, 8013, 8432, 8517, 8820, 9793, 10522, 110
7	201	[]	[84, 97, 122, 133, 843, 1097, 1151, 1426, 2489, 2647, 2723, 2893, 3024, 3151, 3314, 3436, 5043, 5091, 6514, 7812, 7966, 8013, 8432, 8517, 8820, 9793, 10522,
8	843	[]	[84, 97, 122, 133, 201, 1097, 1151, 1426, 2489, 2647, 2723, 2893, 3024, 3151, 3314, 3436, 5043, 5091, 6514, 7812, 7966, 8013, 8432, 8517, 8820, 9793, 10522,
9	1097	[122]	[84, 97, 133, 201, 843, 1151, 1426, 2489, 2647, 2723, 2893, 3024, 3151, 3314, 3436, 5043, 5091, 6514, 7812, 7966, 8013, 8432, 8517, 8820, 9793, 10522, 11044
10	1151	[843, 1097]	[84, 97, 122, 133, 201, 1426, 2489, 2647, 2723, 2893, 3024, 3151, 3314, 3436, 5043, 5091, 6514, 7812, 7966, 8013, 8432, 8517, 8820, 9793, 10522, 11044, 1109
11	1426	[1151]	[84, 97, 122, 133, 201, 843, 1097, 2489, 2647, 2723, 2893, 3024, 3151, 3314, 3436, 5043, 5091, 6514, 7812, 7966, 8013, 8432, 8517, 8820, 9793, 10522, 11044,
12	2489	[]	[84, 97, 122, 133, 201, 843, 1097, 1151, 1426, 2647, 2723, 2893, 3024, 3151, 3314, 3436, 5043, 5091, 6514, 7812, 7966, 8013, 8432, 8517, 8820, 9793, 10522, 1
13	2647	[201, 1097]	[84, 97, 122, 133, 843, 1151, 1426, 2489, 2723, 2893, 3024, 3151, 3314, 3436, 5043, 5091, 6514, 7812, 7966, 8013, 8432, 8517, 8820, 9793, 10522, 11044, 1109

(c) Transforming Raw Data to Insights: Visualizations

Now we have created clusters of strongly connected nodes and their connection with other clusters as graph, and below is the sample of the graph which displays one of the clusters and the nodes connecting to it from different clusters.



Few of the applications of such graph:

- The graph tells us how a group of users (similar users) are connected to another group of users and how one post which is being circulated in a cluster can reach the other cluster. The nodes/users connecting 2 clusters are the medium for passing on the post to other group of users and they are also considered as weak-links.

- The clusters can be even used to suggest a post of a user to another user in the same cluster as the users have been categorized based on the tags.

- Or even to find the user where a post originated from.

(b) Challenges and Solutions:

- Representing outputs of insights-1,2 as web page :

As python is language being used, it is difficult to integrate python with HTML until we have a web framework/server that interprets python files. The installations and setup procedures for web server are a huge task which was not required for a simple web page which has 3-4 input tags. To make it simple and accessible even without web servers we used ajax for the web page which could take the csv file contents and display the results.

- Similarity measure :

Implementing the basic formula of TF-IDF without using any library ignored many documents which were similar. As mentioned in the insights, TF-IDF without normalizing the values were even not much of use and ignored few documents which were similar vectors as well.

It also was not time and memory efficient when dealing with 67k users dataset as the resultant matrix would be very sparse. Cosine similarity / Pearson correlation coefficient was the solution to this challenge which returned the value of similar documents correctly. Even the return value is of type Compressed Sparse Row Matrix which was memory and time efficient.

SECTION – 3

Extension of course concept:

The insights mentioned and explained above covered topics like recommender systems, clustering which actually are not a part of the course, but we have implemented these concepts on the basis of course concepts. For instance, Recommender system is one of the concepts to predict user preference and is more related to data mining. Making use of TF-IDF similarity, we have further explored about the topic which lead to the cosine similarity called Pearson correlation coefficient. TF-IDF implemented with cosine function gets the appropriate similarity measure. Similarly, the other insights were formulated on the basic ideas like page ranking and clustering.

Conclusion:

Summarizing the project, we have implemented the modules of the course for the dataset with the intention of blending different data models to compose/improve features for StackExchange out of results which we got after analysing the data.