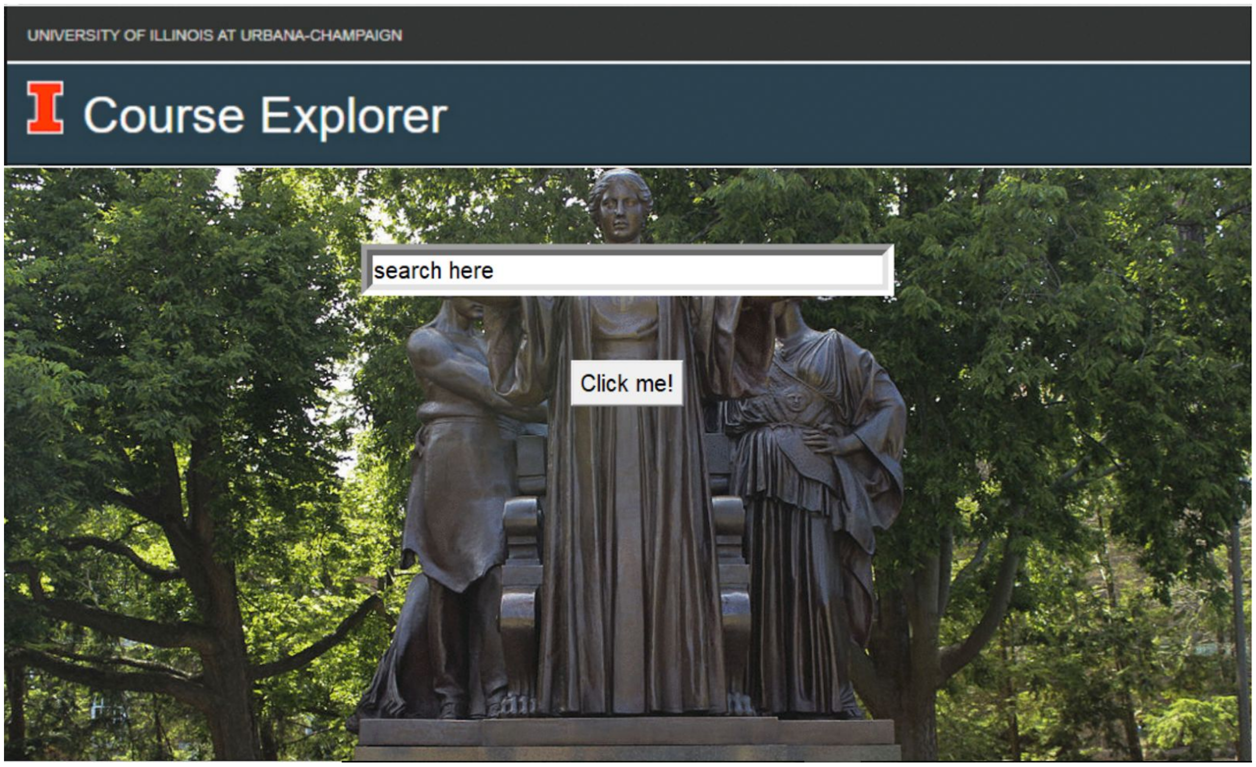
COURSE SEARCH ENGINE FOR U of I

CS410 - Text Information System

Team members:

1) Garima Garg (garimag2@illinois.edu)
2) Kumkum Yadav (kumkumy2@illinois.edu)
3) Paulami Ray (paulami2@illinois.edu)

Code is available at :
https://garimag2.github.io/Course_Search_Engine/



1. INTRODUCTION

Registering for the right courses forms a very integral part in the academic life of the student. Our project aims at solving these hard decisions of the students as per their query and help them in making an informed decision about the courses they can take based on the query. This documentation hence gives the overview of the model and discusses the advantage of this model. It compares the existing model with the proposed model and discusses future enhancements that can be made.

2. MOTIVATION

We are blessed with the college that provides us with enough flexibility to choose from a number of courses and learn from a wide range of topics, but this flexibility can also confuse a student and hence handicap him or her from taking an informed decision about the courses he or she wishes to take.
The current course explorer of University of Illinois, Urbana Champaign specializes in term matching. It is equipped to give courses per term, per semester and also of a specific department. But it lacks the ability to rank searches based on the user's input. It does not equip the user with the knowledge of which courses correspond to his or her query but gives all the courses that match the particular user's input.The users are still confused about which courses they should take.
Our model aims at solving this ambiguity by taking the input from the user and applying appropriate natural language processing techniques in the background to achieve the top 20 courses as per the user query thus giving the user a clear idea as to which course he or she should focus on and use that knowledge to register for the courses.
The model is hence motivated by the fact of increasing courses and job titles per year. Minute differences between a *data analyst* and a *data scientist* which is hard to capture by the user can be easily spotted by the model and hence help the user to make an clear informed decision.

3. SOFTWARE

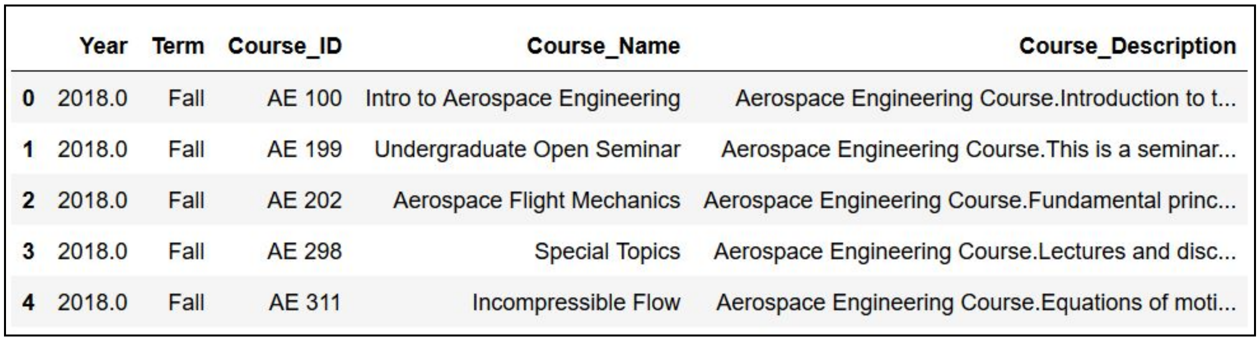
We used several Python libraries to build the search engine.The below table illustrates the library names and how we used it to achieve the results.

	Library Name	Used For
1	Nltk	Is used to obtain the stopwords list from the corpus.
2	tkinter	Is a GUI application wrapper in Python that forms the basis of our application layer.
3	Beautiful soup[4]	Is used to scrape data from websites and formed the basis of making the csv file of all course data.
4	Gensim[3]	This library is used to convert documents into a list of token, convert them into lowercase and remove punctuations. It is used to perform basic data cleaning and tokenizing process.
5	sklearn	Machine learning library that comprises of CountVectorizer, LatentDirichletAllocation and GridSearchCV. Sklearn is a strong python package that powers user with the machine learning techniques.
6	pylda	This library is used to make a visualization that better helps to understand the clusters made by LDA.
7	numpy	Is a basic python library used for data formatting.
8	pandas	Is a basic python library like numpy used for data formatting and data reading.

9	webbrowser	Is used to open a page in a webbrowser which corresponds to a particular url.
10	matplotlib	Is used to plot basic visualizations.

4. DATASET

The dataset after scraping and applying data processing techniques consists of 5 columns namely *Year*, *Term*, *Course_ID*, *Couse_Name* and *Course_Description*. The dataset is of 795 lines. The dataset contains courses from departments Engineering, Business, School of Information Sciences and Statistics. Figure 1 shows the dataset and its contents using data.head() in python :



	Year	Term	Course_ID	Course_Name	Course_Description
0	2018.0	Fall	AE 100	Intro to Aerospace Engineering	Aerospace Engineering Course.Introduction to t...
1	2018.0	Fall	AE 199	Undergraduate Open Seminar	Aerospace Engineering Course.This is a seminar...
2	2018.0	Fall	AE 202	Aerospace Flight Mechanics	Aerospace Engineering Course.Fundamental princ...
3	2018.0	Fall	AE 298	Special Topics	Aerospace Engineering Course.Lectures and disc...
4	2018.0	Fall	AE 311	Incompressible Flow	Aerospace Engineering Course.Equations of moti...

Figure 1: Dataset

5. IMPLEMENTATION

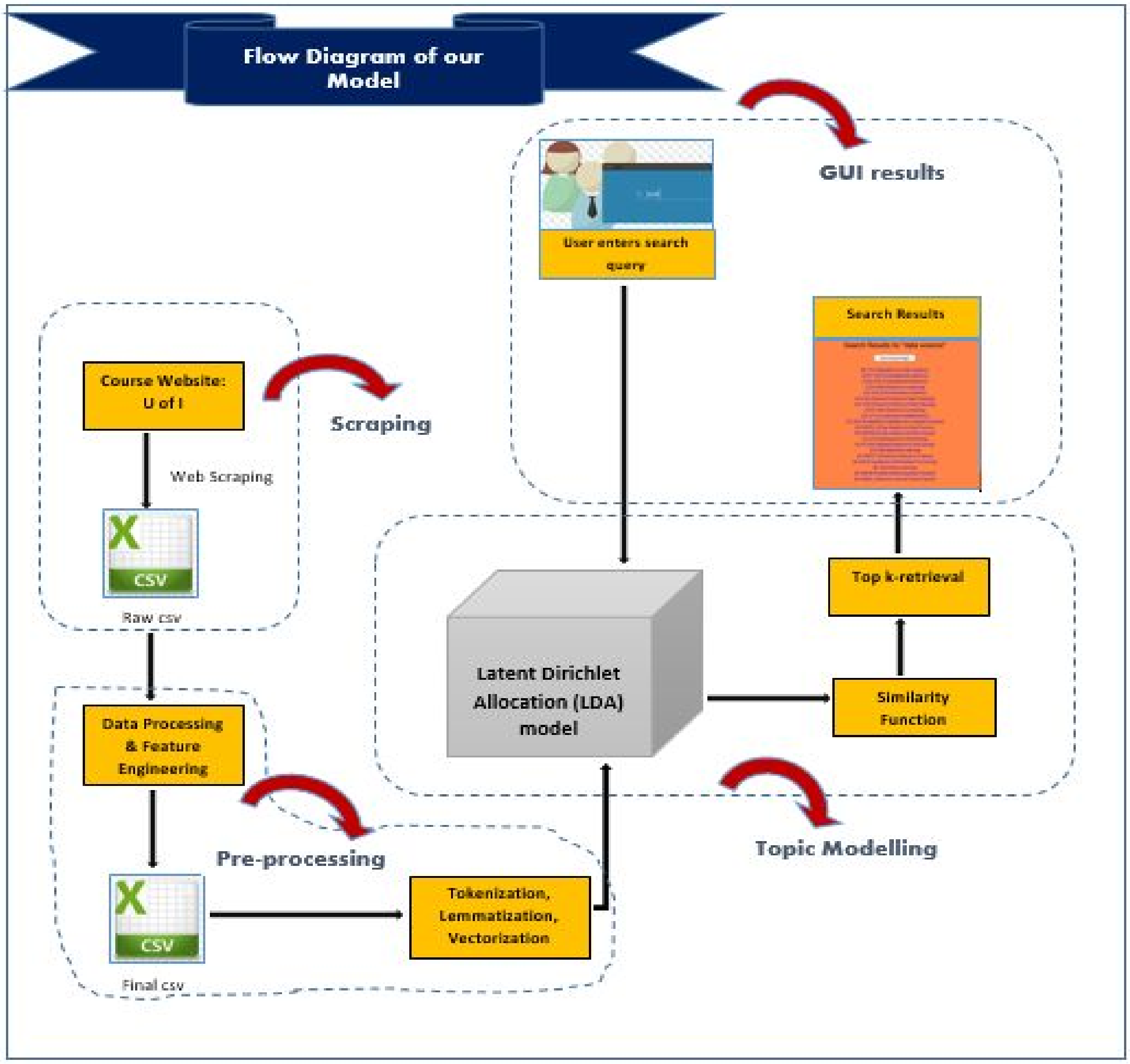


Figure 2 : Process flow diagram

Steps followed :

1) **Scraping:** This formed one of the most integral part in the project. HTML as a language works even if the guidelines are not accurately followed, but this poses a problem for writing code to scrape that data. BeautifulSoup is a

library in Python that comes to the rescue. It allows us to easily read the HTML file and search contents in it with the help of the find_all functionality. Using this we were able to create a csv file for each department and then merge those csv files to form the dataset. For more information on the dataset please refer the dataset section.

2) **Data Processing and Feature Engineering:** The dataset obtained from the site had a lot of issues. Firstly, many courses were routed to their parent courses with a comment in the course description that this course is similar to the parent course. This hence would pose a problem in data processing for the specific course. Secondly, course descriptions for most courses were not very accurate and informative. Hence, these descriptions were taken from the course websites to help NLP models to group them under the correct topic. These changes were made manually to make the dataset better and obtain the final csv file to be used in NLP.

3) **Tokenization, Lemmatization and Vectorization:** These three processes work on the course descriptions of the dataset. Tokenization process cleans the data, converts all the words into lowercase, tokenizes the data and removes punctuations. Lemmatization helps to convert the data to its root word. For example *mathematics* and *math* are mapped to the root word *math*.

4) **Latent Dirichlet Allocation (LDA):** LDA is a generative statistical model[1] which allows to group the vectorized data. This grouping helps us to obtain variety of topics based on the number of components entered in the model. The model first randomly assigns the word into n topics. Then the model computes probability of the topic per document that is the proportion of words in the document and also the proportion of assigning a word to a specific n topic. This then is done over and over again till the model achieves maximum likelihood. The results are the values that achieve this maximum likelihood.[2] To choose the best models we use GridSearchCV algorithm which helps to train multiple LDA models over multiple topics

and helps to choose the best LDA model for the process. Here we use an LDA model with number of topics as 10 and learning decay 0.9.

5) **Similarity Function :** Similarity Function is what defines which grouping will be chosen with respect to a specific query in a vector space. In our model we use the euclidean distance as the similarity function.

6) **Top k-retrieval** : Using the query from the user and applying the similarity function gives a list of documents from closest to farthest similarity. We can then use the top k specification to return the closest topics to the query. Here we use k=20 and return 20 top courses with respect to the query.

7) **User enters search query :** This is achieved using Tkinter library in Python where a search box takes the input from the user and sends it to the backend process from where it goes to the LDA model, similarity function and finally top k-retrieval of documents.

8) **Search Results:** This is an extension to the GUI where the results are displayed to the user. The same is achieved through the Tkinter library. There is also a button "Go to Course Page!" that takes the user to the official course page of UIUC.

6. README FOR CODE IMPLEMENTATION

For executing this model one would need Jupyter 5.4.1 installed on their systems. The following steps should be followed to run the project from start to end:

1) Download the course_explorer.zip file to your desktop.
2) Unzip this file and save it on your systems.
3) This folder would consist of files final.ipynb, Scraping and Cleaning.ipynb, images folder, final_worked_new.csv, numbers.txt, README.txt.
4) Run the Scraping and Cleaning.ipynb to generate the scraped dataset.
5) Run the final.ipynb (Note : The final.ipynb uses final_worked_new.csv as manual cleaning was required in the dataset.)

6) Wait for the .ipynb to execute and for the GUI to appear.
7) Enter your query and press “Click Me!”.
8) See the results and plan your coursework!

7. RESULTS

The model that is proposed takes the query from the user in the Tkinter GUI. The user can input the query in the search box and then press the button “Click Me!”. The figure demonstrating the same can be seen in Figure 2. Once the “Click me” is clicked, the input goes to the backend process of LDA, similarity function and top k-retrieval documents. The output is then printed out in the results page from closest to farthest as shown in Figure 2.
The test is performed for the word “Analytics”. This search produces top 20 course names as shown in Figure 2. If we concentrate on just the top 5 documents we see that the courses are very closely related to the topic of analytics. This hence helps to identify that our model is working well.
Performing the searches on other words like “Programming” gives us very good results for the top 5 documents.
This gives us a good overview of the model. Our test case consisted of queries like *“Programming”, “computer science”, “aerospace”, ”finance”, ”mechanical”, “seminar”, “business”* and *“ data structures”.*

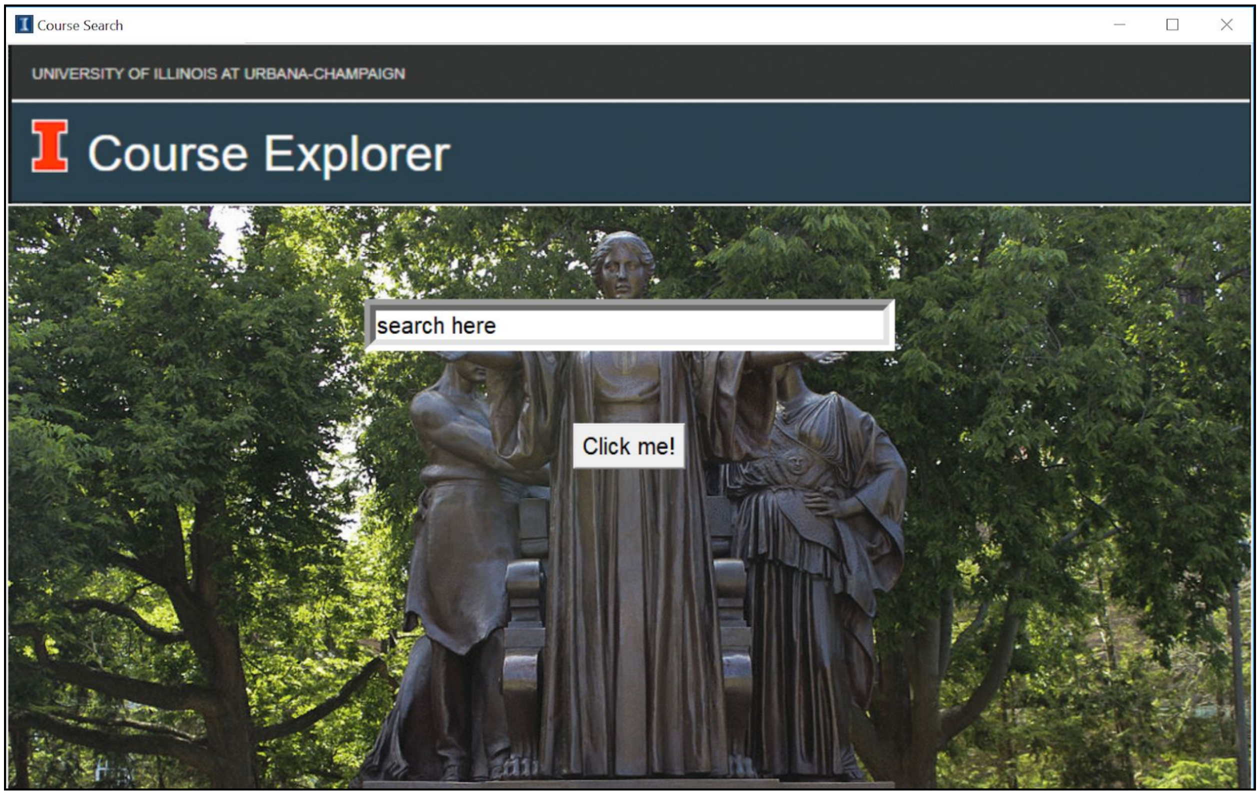
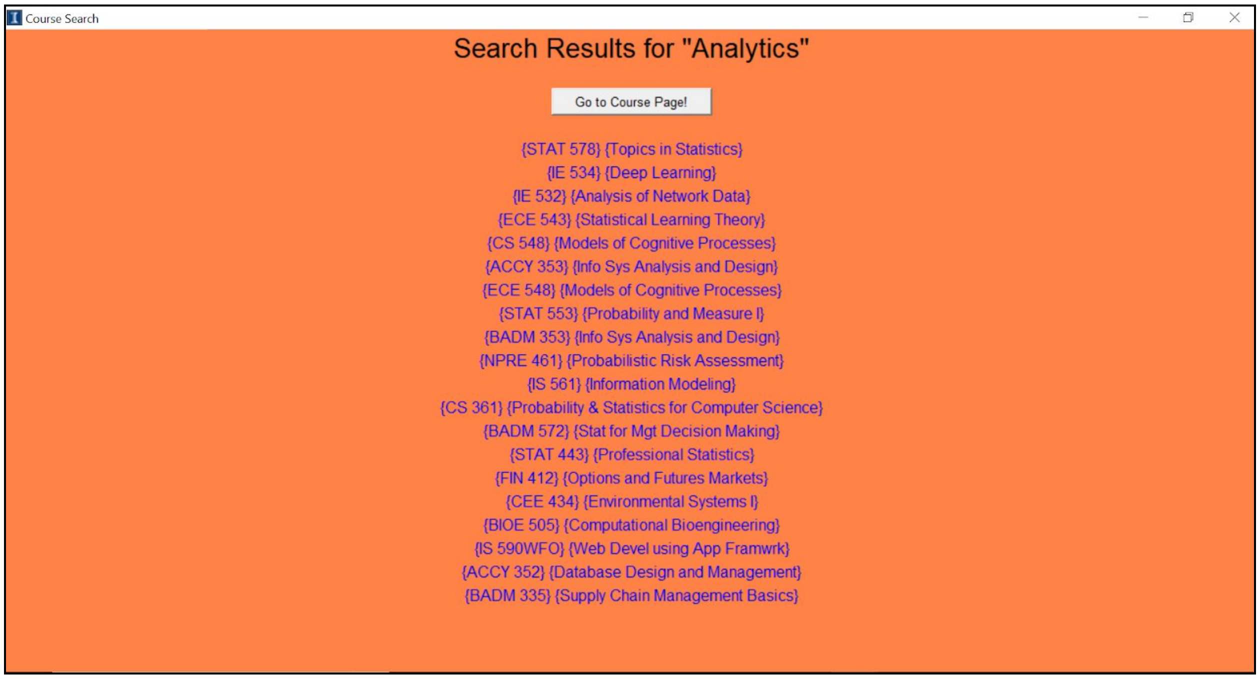


Figure 3: Course Explorer Entry Page



Search Results for "Analytics"

{STAT 578} {Topics in Statistics}
{IE 534} {Deep Learning}
{IE 532} {Analysis of Network Data}
{ECE 543} {Statistical Learning Theory}
{CS 548} {Models of Cognitive Processes}
{ACCY 353} {Info Sys Analysis and Design}
{ECE 548} {Models of Cognitive Processes}
{STAT 553} {Probability and Measure I}
{BADM 353} {Info Sys Analysis and Design}
{NPRE 461} {Probabilistic Risk Assessment}
{IS 561} {Information Modeling}
{CS 361} {Probability & Statistics for Computer Science}
{BADM 572} {Stat for Mgt Decision Making}
{STAT 443} {Professional Statistics}
{FIN 412} {Options and Futures Markets}
{CEE 434} {Environmental Systems I}
{BIOE 505} {Computational Bioengineering}
{IS 590WFO} {Web Devel using App Framwrk}
{ACCY 352} {Database Design and Management}
{BADM 335} {Supply Chain Management Basics}

Figure 4: Search Results

The visualization tool pyLDAvis helped us a lot to analyze the topics in the group and provided deeper understanding on the results of the model. The visualization can be seen in Figure 4.

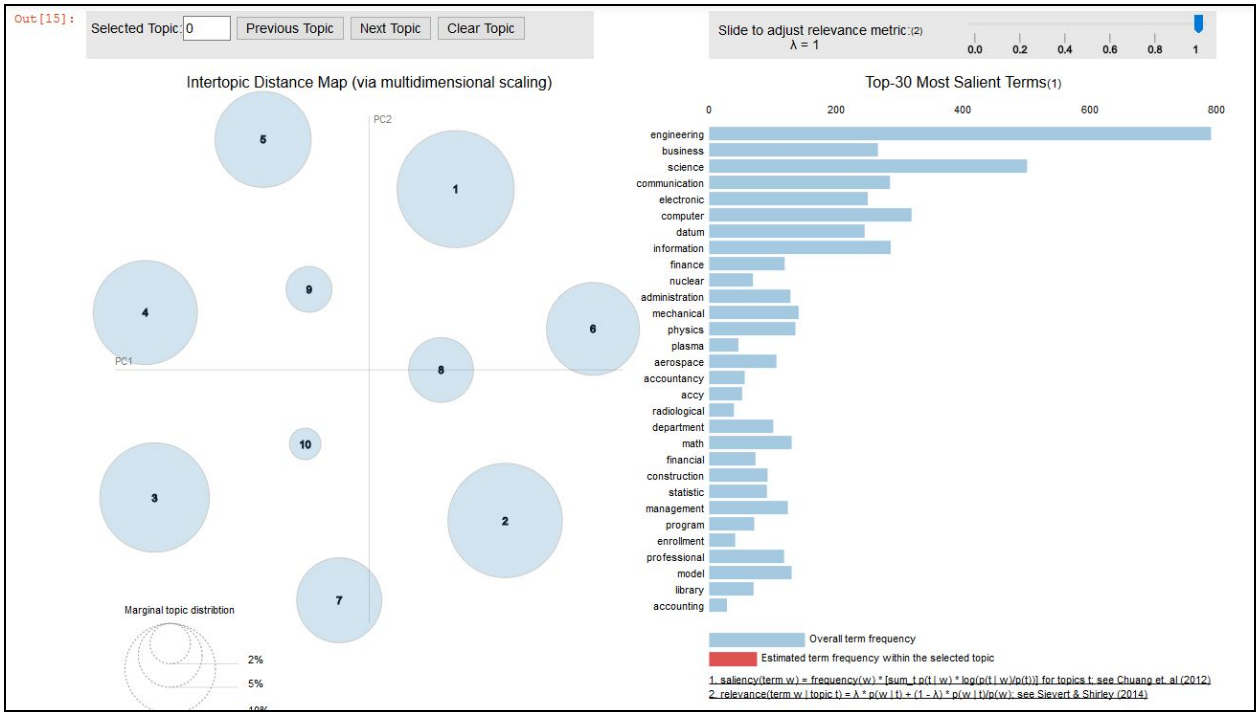

Out[15]:
Selected Topic: 0
Previous Topic
Next Topic
Clear Topic
Slide to adjust relevance metric:(2)
λ = 1
0.0
0.2
0.4
0.6
0.8
1
Intertopic Distance Map (via multidimensional scaling)
PC2
PC1
1
2
3
4
5
6
7
8
9
10
Marginal topic distribtion
2%
5%
10%
Top-30 Most Salient Terms(1)
0
200
400
600
800
engineering
business
science
communication
electronic
computer
datum
information
finance
nuclear
administration
mechanical
physics
plasma
aerospace
accountancy
accy
radiological
department
math
financial
construction
statistic
management
program
enrollment
professional
model
library
accounting
Overall term frequency
Estimated term frequency within the selected topic
1. saliency(term w) = frequency(w) * [sum_t p(t | w) * log(p(t | w)/p(t))] for topics t; see Chuang et. al (2012)
2. relevance(term w | topic t) = λ * p(w | t) + (1 - λ) * p(w | t)/p(w); see Sievert & Shirley (2014)

Figure 5 : Per topic visualization

8. CONTRIBUTION OF EACH TEAMMATE

The team has worked together at all steps in the development of the project. The team members have shown equal contribution in the making of the model. We have worked together at each step and have come up with interesting ideas to improve the performance of the model.

9. FUTURE SCOPE

The model is still a novice representation of the final product that we wish to implement. Many useful things can be done with this model. Some of them are stated below :

1) We can use the user's CV and extract words from them making a personalized dashboard of courses for the user best suited with his or her profile.
2) We could also add a recommender system with some initial interests taken from the user and map those interests to generate similar courses that could be liked or disliked by the user.

Overall the model can be applied to many industries and not only as the search engine for UIUC. This model can be very useful to the users to plan their roadmap and make a course structure more suitable to their needs.

10. CONCLUSION

Finally, we can see that our model performs well for the given dataset. The model successfully captures the top 20 documents related to a user's query. We start off with analysing the capabilities in the existing search model and then think of ways to implement a search engine based on user's query. The model uses various techniques like scraping, data preprocessing, topic modelling and GUI development.

REFERENCES

[1] Wikipedia, *Latent Dirichlet Allocation*
https://en.wikipedia.org/wiki/Latent_Dirichlet_allocation
[2] Edwin Chen, *Introduction to Latent Dirichlet Allocation*,
http://blog.echen.me/2011/08/22/introduction-to-latent-dirichlet-allocation/
[3] Radim Rehurek, gensim Documentation *Release 0.8.6*
https://media.readthedocs.org/pdf/gensim/stable/gensim.pdf
[4] PythonForBeginners, *Beautiful Soup 4 Python*, Mar. 09, 2016
http://www.pythonforbeginners.com/beautifulsoup/beautifulsoup-4-python
[5] pyLDAvis, *Welcome to pyLDAvis's documentation!* http://pyldavis.readthedocs.io/en/latest/
[6] Tkinter — Python interface to Tcl/Tk¶ https://docs.python.org/2/library/tkinter.html