A Content-Based Filtering Recommendation System for Matching Student Questions to Interested Professionals

Destiny Adams Hannah Roe Harrison Ratcliffe

Olivia Lyons Sebastian Cortes

I. Abstract

CareerVillage.org is a forum based website created to help students without guidance get answers to career questions from real professionals. In an effort to improve their system for recommending questions to volunteering professionals, CareerVillage partnered with Kaggle and Data Science for Good to challenge programmers around the globe with preparing a solution.

This paper describes the process our team followed in an attempt to apply classroom-based knowledge to real world problems that have potentially great societal impact. As students ourselves, we understood the pressure and confusion of selecting a "final" career path, and so felt compelled to apply ourselves to this challenge.

Volunteering professionals are to receive email subscriptions on a weekly to monthly basis containing new questions, ideally related to their industry. Using the account, question, and answer data provided by CareerVillage, our team has developed a simple, content based recommendation system that would provide ten new questions based on the professionals previous activity. This output could be used to ensure professionals remain active by continuing to recommend relevant questions, which helps students navigate their path to being future professionals.

II. Problem Framing

Choosing a career path is often one of the biggest questions on every student's mind, and making the right decision can be extremely daunting for youths still trying to figure out who they are and what they are interested in. When faced with this choice, students are often encouraged to approach their guidance counselors for advice. Unfortunately, this is not always an option, as in the U.S. alone there are about 500 students for every guidance counselor (NACAC and ASCA 6). On top of this, other factors like lower school funding, family misfortunes, lower socioeconomic status, and limited access to role models in the workforce can also prevent students from receiving the career advice they need (Nakkula et al 117). This lack of access to career advice and resources greatly hinders a students career decision when compared their counterparts who do have access to said career resources (Nakkula et al 116-117). These students without access to a guidance counselor or other sources for career questions and advice must find different avenues to get their questions answered. Many of these students turn to the internet to find career advice and answers for their questions, specifically a platform called CareerVillage.

CareerVillage.org is a website where students who may be lacking in support can get their career questions answered by professionals. CareerVillage's main goal is to provide a welcoming platform for students to receive reliable career guidance from experienced professionals, particularly students who do not have easy access to career advice within their communities. Students can create an account and immediately start asking questions pertaining to specific careers or goals. Professionals can set up a volunteer account and specify preferences for the types of questions they are willing to answer. When a student asks a question, CareerVillage determines which professionals are the most relevant and qualified to answer, then emails the question to these professionals. Both students and professionals also have the option to read through previously asked and answered questions.

In order for students to receive reliable career advice, their questions must be accurately matched to relevant professionals. This accurate matching benefits both the student asking the question and the professional. Students who receive reliable answers are more likely to continue seeking advice, and professionals who receive relevant questions are more likely to answer them and thus are more likely to continue answering questions. Thus, CareerVillage is constantly trying to improve their solution to the problem of how accurately questions can be matched to relevant professionals who are most likely to answer them. CareerVillage recently turned to Kaggle, a data science platform where users can solve data science problems, distribute data, and take part in data science competitions, to help them find more efficient ways to match questions with relevant professionals. Using Kaggle, CareerVillage set up a competition which challenged Kaggle users to develop a solution which would recommend relevant questions to the professionals who are most likely to answer them, based on data from CareerVillage.org. This paper aims to provide a solution to the following problem proposed in CareerVillage's Kaggle competition: Using data from CareerVillage.org, can we develop a method to accurately recommend a student's questions to qualified professionals who are the most likely to answer them. We propose a recommender system built using Python as a solution this problem.

Effectively matching questions with professionals is important for providing students with the best user experience on CareerVillage.org and building its reputation as a reliable online resource for career advice. Since CareerVillage provides a platform for students to receive answers to their questions over the internet, it is important for it to be as reliable as possible. There are already many different platforms where people can ask career questions, but CareerVillage tries to get the most qualified professionals to answer rather than another random user. Thus, improving the accuracy of how CareerVillage recommends questions to professionals further increases CareerVillage's reliability and quality as a platform. This encourages students to keep asking questions and retain volunteer professionals by only giving them questions which are relevant to their career or interest. Furthermore, having an online resource such as CareerVillage where individuals can ask any career questions is a huge advantage for students, especially those who do not have access to more traditional methods of receiving career advice such as guidance counselors. This makes CareerVillage an important resource for underprivileged youth who often do not have access to career advice resources within their own community or cannot receive the advice they need from guidance counselors. By improving how CareerVillage recommends students questions to professionals, underprivileged youths are more likely to get the advice and support they need for deciding their careers.

This problem was chosen because the team, as college students, understands how difficult it can be to choose a career and receive useful career advice. Access to resources for career advice

is something that every student should have, and CareerVillage provides just that. By answering this question of how to improve the way CareerVillage recommends students questions to professionals, the quality of advice students receive could greatly increase, in turn helping students find the best career path. This problem was also chosen because it was hosted on Kaggle.com as a data science competition. All the data, problem statements, and documentation were easily accessible from Kaggle.com, as well as advice from CareerVillage on where to start and possible approaches to the problem. This made the problem even more appealing as the data could be gathered and cleaned fairly quickly, making it possible to start working on a solution as soon as possible.

III. Data

As our project idea was that of Kaggle's "Data Science for Good: CareerVillage.org", we were provided our dataset by Kaggle. Teams competing were allowed to manipulate the datasets in the manner they preferred as long as their ending result can be validated by CareerVillage.org to be well performing, easy to implement, and extensive enough to add future data features.

Within our provided dataset were 15 .CSV files. Each spreadsheet file held information using variables that often varied from other files. Though often different in what data was stored, all of the CSV files could be cross-referenced using specific variables such as the IDs of a student, a professional, an email, etc. Below is a list of all the CSV files given to us through the dataset.

- *Answers*.csv – Responses from professionals to students in regards to questions regarding a field that the professional is accustomed with. Columns within this spreadsheet include IDs for the answer itself, the author, the question being responded to, the date of response, and the answer body.
- *Answers_scores*.csv – Scores in "Hearts" for any given answer. Columns include the answer's ID and its respective "Hearts" score.
- *Comments*.csv – Comments made to either answers or questions. Columns include ID of comment, the author, the parent content (the piece of content being commented on), the date added and the comment body.
- *Emails*.csv – All emails sent between individuals and the frequency of receiving a digest. Columns include ID of email, ID of email recipient, date sent, and frequency of digest.
- *Group_memberships*.csv – Groups that users can join. Columns include the group ID, and the groups user's ID
- *Groups*.csv – The types of groups. Columns include the group ID and their group type,
- *Matches*.csv – A collection of emails that contain questions. Columns include the emails' ID, and the ID's of all questions that may have been included within the email.
- *Professionals*.csv – A collection of professionals that have opted-in to respond to students' questions. Columns include the professionals' ID, location, industry, headline and date joined.
- *Questions*.csv – Contains all of the questions every student has submitted. Columns include the questions' ID, the ID of the author, the date added, the question title and question body.
- *Questions_scores*.csv – Scores in "Hearts" for any given question. Columns include the questions' ID and its respective "Hearts" score.

- *School_memberships.*csv – Schools that students can join, similar to a group. Columns include an ID for the school and the ID of student within that school grouping.
- *Students*.csv – A collection of students who've submitted questions to receive answers from professionals that are knowledgeable in the topic at hand. Columns include a student's ID, location, and date joined.
- *Tag_questions.*csv – Tags that can be used to mark a question belong to a specific grouping. Columns include the question's ID and the tag ID that the question has.
- *Tag_users.*csv – Tags that a user can follow. Columns include the user's ID and the ID of the tag they're following.
- *Tags.*csv – A collection of all tags that the system currently handles. Columns include the ID of each specific tag and the name of the tag.

Of the different spreadsheets available to use, we decided to focus only on a select few for this project. Those spreadsheets are:

- *Answers*.csv
- *Professionals*.csv
- *Questions*.csv

With our selection of those spreadsheets, we determined we were able to cross-reference between them all to get the information required for our model. Through using a questions' ID, we're able to generate keywords from the body, give the keywords value in relation to their importance and create a matrix using dot-product with answers to give a recommendation of an old answer to a new question. Then, the old answer's author ID could be given back to signify they're knowledgeable in the questions' topic, thus they should be able to respond to future questions from students.

The remaining spreadsheets were deemed insignificant to our project due to having information that lacked coherence with other data from the four spreadsheets we had chosen. Though added to the Kaggle dataset after we had downloaded the files, both *answers_scores and* questions_scores would not have been used for this project. This is due to our usage of finding our own keywords from the questions' or answers' body text. Other spreadsheets, such as *groups* and *school_memberships* had similar reasons for being unused. Though they can be implemented in, potentially, a future created, we did not find a concrete reason for grouping students or employees further.

Within the *professionals.csv* file, we concluded to cleaning up the data within that file. Data cleaning allows us to verify that our data used in our model fits to what we're trying to use it for or "correct these errors or at least to minimize their impact on study results" (Van den Broeck, 1). Throughout the file, we saw that there were professionals who lacked an industry, headline, or both. By having professionals that did select an industry or headline, we can group professionals by those parameters. However, if a professional did not have either an industry or a headline, we simply removed them from within this file and stored their ID in a text file. This text file would result in a list of all deleted professionals. With this, we're able to search through other useable spreadsheets and remove professionals that were initially deleted. This allows us to reduce the amount of data that our program has to run through by, and thus improving the run

time of our code. This process of cleaning the file was done through our *professionalsCleaner.py* file.

Through utilizing the *rake-nltk* python library, we were able to extract keywords from a question's body and header. As such, the *questions.csv* had to be cleaned in order for only the questions' ID, body keywords and header keywords to be the remaining data from that spreadsheet. To determine the keywords of text, we simply called the *rake()* function from *rake_nltk*. Scores are given to each keyword found through "word frequency, word degree and ratio of degree to frequency" (Rose, 7). All the found keywords were simple added to newly created columns for keywords from the question body and keywords from the question title. Then, every other column excluding ID and the two keywords columns were deleted from the spreadsheet. This was done using pandas' *drop()* function. A problem arose in that special characters such as "/?<" appeared in our keywords. We had to remove them using the *regex* substitution function. Then, all the keywords from both the title and body were combined as they're both parts of the same question.

Other spreadsheets were utilized to create graphs that would help us further understand the data. By looking at the *professionals.csv* file, we saw that looking further into statistics regarding their location could be helpful. A simple dictionary was created based on all the locations within the file. This allowed us to create a total tally of every professional that had a specific location attached to their name. As a result, we were able to find that there were similar locations that shared large amounts of users. This could be used for future model created and is talked about in a future section within the paper. We also had to deal with professionals that have been inactive since their account creation date. While we found that the length of the entire professionals file was about 30,000 accounts, it was unlikely that every single professional had answered a question. Upon overview of the *answers.csv*, there were 51,123 total answers. Upon checking the number of professionals that answered, there were only 10,169 different authors. This meant that about 2-in-3 professionals did not answer any questions. By looking at how many answers a professional responded too, we found that many answers very few questions. By creating a boxplot using seaborn, we found a large number of outliers.

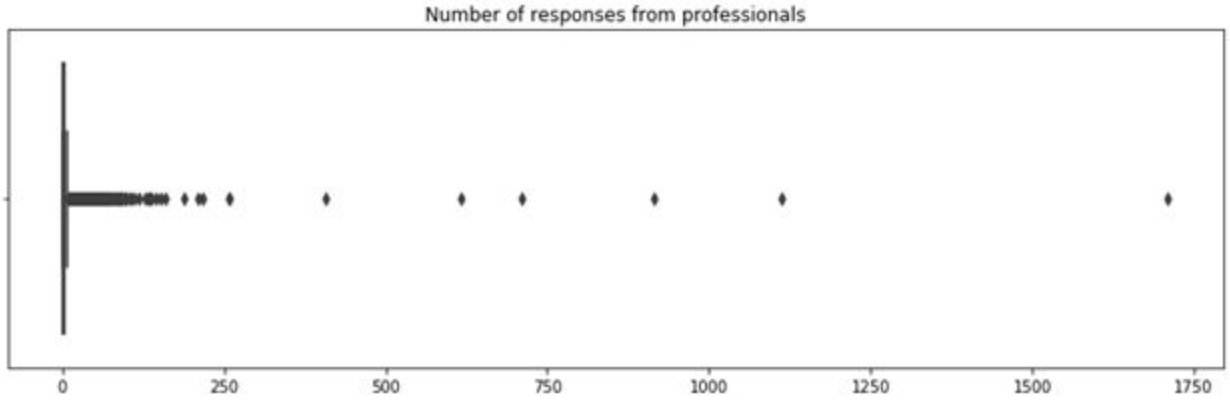

Figure 1. Boxplot of total responses a specific professional's ID is attached to. (n = 10,169)

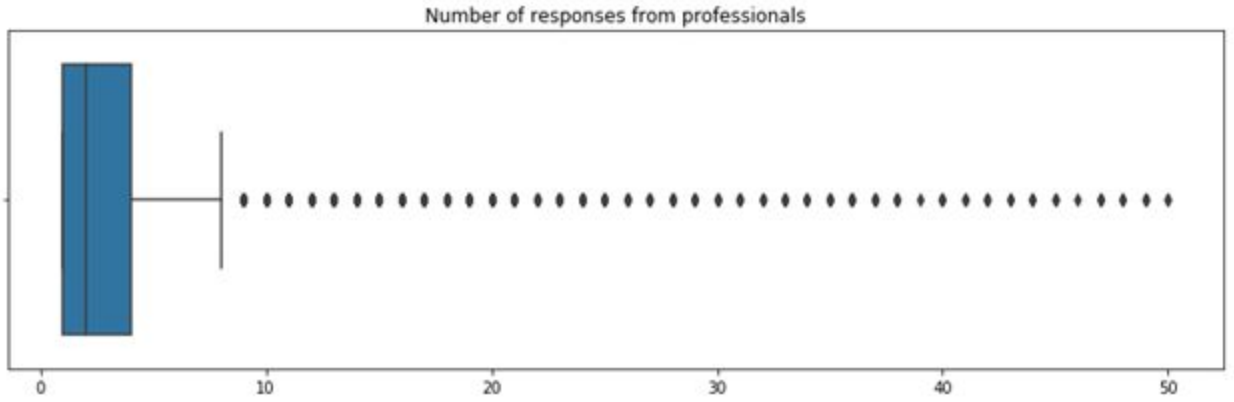

Figure 2. Boxplot, having removed outliers from Figure [1]. of total responses a specific professional's ID is attached to. (n = 10074)

Based on Figure 1, the outliers would heavily skew our data. The max value within the data of responses was 1,710. This indicates that one single professional answered over a thousand questions. The top ten number of responses from individuals ranged from 1,710 to 210. Instead of the process of dealing with outliers, we decided to "trim" our data instead of "winsorizing" or utilizing a "robust estimation method". (Kwak, 410) As a result, we decided to remove individuals who answered over 50 questions. This totaled to 95 individuals being removed. As shown by Figure 2, the boxplot was significantly improved by showing that most professionals often answered very few questions. Though there exist some outliers within this data as well, those that answered more than 9, we decided to leave them in as their answers could benefit our final result. Further data analysis is explained later in the paper.

IV. Process

What exactly is a Recommender system? To put it simply, it is a system that produces recommendations based on user activity and certain characteristics of the items involved (Ricci et al. 1). This is something that has become so common that most people encounter them going about their daily life. Any service that offers suggestions based on user activity is using a Recommender. Well-known examples include Netflix movie suggestions, Youtube's related videos, Spotify's weekly discover playlists, and countless other media services. Advertising firms also use Recommenders, which manifest in individuals being shown ads specifically tailored to their public interest and search engine queries.

There are two types of Recommender systems. One relies solely on the choices of one user and the content they choose to interact with. Their recommendations are based on similar characteristics between content they've interacted with and other content in the database. Items that closely match what they've already chosen to view are presented to the user as material they might also want to view. As an example, take a hypothetical movie streaming service user named Sally. Sally likes to watch horror movies, so the items that come up in her suggestions are also horror movies. This is because they share a genre tag and perhaps other characteristics with the movies she's already chosen to watch. She would not likely be recommended a fantasy genre film even if she would actually enjoy it, because the movies she's already watched are so

dissimilar to this one. The other type of Recommender involves the choices of other users who view similar content (Herlocker, 2004). If two users interact with similar content and one user also interacts with content the other has never seen, the first user may be recommended things not related to their own choices because of this. For example, take Sally again as the first user and add someone named Bob as another. Sally and Bob both frequently watch horror movies, perhaps even the same exact films. Bob also really loves fantasy films. Because a lot of his choices are similar to Sally's and the movie streaming service they use implements a collaboratively filtered Recommender system, Sally might find something like *The Labyrinth* or *The Lord of the Rings* among her horror recommendations. Despite the fact that she has never interacted with similar films, she is being shown these because of Bob's choices.

Between these two options, we felt it was more suited to the project to implement a content-based Recommender rather than a collaborative one. This is partially because it is an overall simpler approach, but also because it seemed to make more sense to recommend questions to professionals based on ones they had previously chosen to answer, rather than giving them questions potentially unrelated to their area of expertise.

In order to begin creating the system, we had to clean up some of our data. Most of the work was done on the professionals.csv data set and involved removing those that had been inactive for over a year. We also extracted keywords from the question bodies by using the `Rake()` function and created a new column of these keywords. This served the purpose of matching questions already answered by a given professional to those that had yet to be answered. After our data was clean and the keywords extracted, we used the `CountVectorizer()` function to perform a keyword count on the new questions keyword column to get the frequency of each keyword. Using the same library, we created a cosine similarity matrix from the keyword count. Below is an example of a cosine similarity matrix as output by Jupyter Notebook. It is not the same one we used in this project, as attempting to print that one crashed one group member's computer. Instead, we have included one we used to test the function with a significantly smaller data set. Using the information generated by the cosine matrix, we are able to determine the similarity between questions based on how many keywords there are in common. Knowing this similarity allowed us to select ten questions that should be recommended to the professional.

```
array([[1.        , 0.15789474, 0.13764944, ..., 0.05263158, 0.05263158,
        0.05564149],
       [0.15789474, 1.        , 0.36706517, ..., 0.05263158, 0.05263158,
        0.05564149],
       [0.13764944, 0.36706517, 1.        , ..., 0.04588315, 0.04588315,
        0.04850713],
       ...,
       [0.05263158, 0.05263158, 0.04588315, ..., 1.        , 0.05263158,
        0.05564149],
       [0.05263158, 0.05263158, 0.04588315, ..., 0.05263158, 1.        ,
        0.05564149],
       [0.05564149, 0.05564149, 0.04850713, ..., 0.05564149, 0.05564149,
        1.        ]])
```

Figure 3. Cosine matrix example generated by a test program.

V. Reflection/Review

At the end of the day, our system does what we intended for it to do. Given a certain professional, it will use data from that professional's previously answered questions to find new questions similar to ones that have already been answered by that person. This works well because any given professional will tend to answer more questions on topics they're familiar with and/or enjoy talking about. Thus recommending new questions based on previously answered ones ensures that they will get recommendations that are within, or near to, their interests. This 'bag of words' approach to the problem is also good in its relative simplicity; the keywords are equally important in deciding what questions are related, whether it's a location or a field of research.

However, there is a flaw with this system that is unavoidable with this approach. If a certain professional has not answered any questions yet, they cannot be recommended any new questions because there is not enough data to compare them with. We address this problem in the data cleaning phase by removing any professionals from the pool who have no data (that is, hasn't answered anything yet). This extra preliminary step ensures that things will run smoothly down the line, but it leaves behind anyone who is brand-new to the platform and hasn't had a chance to answer anything yet, or anyone else who, for one reason or another, hasn't accrued any data. In the data provided by CareerVillage, 64.4% of professionals fall within this category of people who have not yet answered any questions.

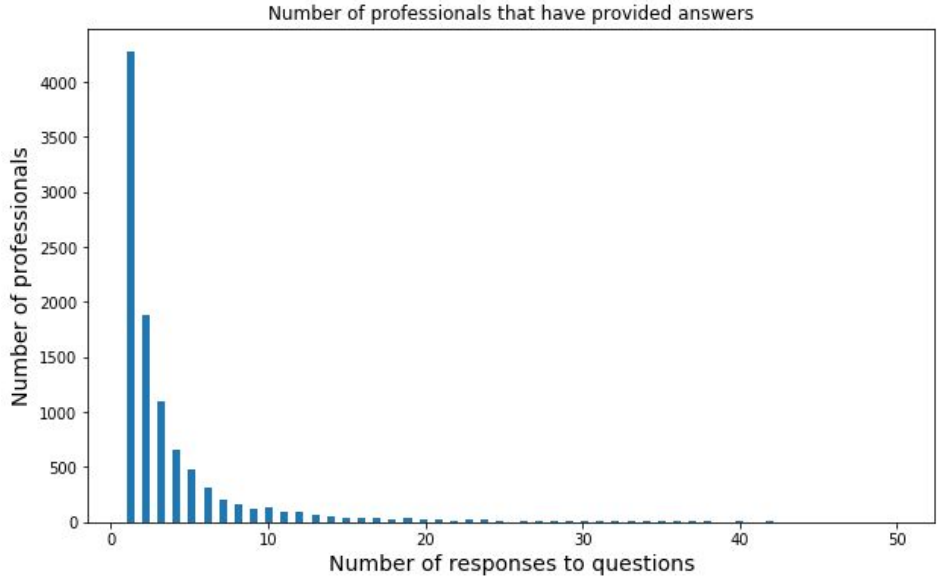

Figure 4. Histogram representing distribution of how many questions professionals answer. Professionals who have answered less than 10 questions are in the majority. Out of the 28,152 professionals provided, about 100 of them have answered 50 or more questions and have not been included on this graph. There are 18,133 professionals who have answered exactly zero questions, over four times the number of professionals who have answered exactly one.

There are a few ways that we could remedy this issue. One method that wouldn't completely change our approach is to create a bag-of-words for these professionals based on their industry,

headline, and location data that they may have provided when they signed up for CareerVillage. This data is very generic and might not lead to very good recommendations, as two professionals from New York within the Engineering industry might have vastly different amounts of interest in answering questions about college applications to engineering schools. However even a basic recommendation system like this would be preferable to random assignment or nothing at all.

Another approach to including the professionals who have not yet answered anything is to account for them in the recommendation system via a collaborative system (Gong, 745). In this approach, a professional with very limited data would instead be given recommendations based on the interest of their peers. This approach requires a metric for deducing the similarity of any two professionals. A simple way would be to once again compare bags-of-words that are made from the professionals' user information, such as their industry. However there is a feature of CareerVillage that can be leveraged to serve this purpose: any user joining the website, as a student or a professional, can join a group. This can be a school, an honor society, or any other binding feature between a number of people. For the sake of privacy, CareerVillage has provided IDs for the groups as well as user IDs for each person belonging to it, but the actual name of the group is hidden. We can leverage this data to make connections between professionals who are in a group together but might not share any other traits, such as two professors of different departments in the same university. For professionals who don't have any other useful data, we can instead make their bag-of-words an amalgamation of the various bags-of-words from the other professionals within their group. This might result in some odd results, such as a biology professor being recommended questions that are better suited for someone in a financial aid office of the same institution, but this supplemental data can help produce recommendations that are more accurate for location, which is something everyone in this example group would share. This collaborative recommendation system can be implemented through industry instead of group, as well. With further research into an implementation of this method, we might be able to determine which approach would be more appropriate for the situation.

As it is, the bag-of-words approach weighs each keyword based on how often it appears. For example, if a certain professional mentions 'GPA' multiple times in every question they have answered, then that keyword will have a high weight. This comes into account when matching to a question; if a given question has the same word mentioned many times, then a match between that question and that professional is much more likely, since the professional 'prefers' to talk about that keyword. Something we noticed in our initial analysis of the data was that a number of questions asked were pertaining to a specific location, such as, 'What's the best engineering school in New York?' While an engineer in any part of the world might have the answer to something like this, it would be better suited for a professional who is a native of the location in question. In our current implementation, location isn't treated any differently from any other keyword. A good subject of further study in this problem would be to explore how location data might influence a match. This could be done by increasing the weight of locations within the bag of words, so that the location is naturally made a higher priority in the later filtering.

If we were to take a machine learning approach to this issue, we might be able to 'teach' a machine the difference between questions that are directly concerning a location and questions that simply have a location mentioned within them. Once this distinction is drawn, the former category can be filtered with location at a high priority, so that it only gets suggested to professionals who share that location in their user information, and questions with the latter category can be processed with the normal amount of concern for location data.

There are a lot of variables in this situation that could all be leveraged in the recommendation system. We chose to make use of keywords derived from question text and tags, but there are many factors that that overlooks which could be factored into the system (McDonald, 231). For example, CareerVillage provided records of what email frequency each professional has signed up for, if any, such as ‘daily’ or ‘monthly’. In calibrating the system, it might be prudent to tweak the number of suggestions based on this preference, so that (for example) ‘daily’ subscribers would only get two or three recommendations and ‘monthly’ or more would have ten or more to choose from. It might be a good idea to also give the ‘daily’ subscribers the simpler questions, since they are likely to answer at a higher frequency which is easier with simpler questions rather than in-depth ones.

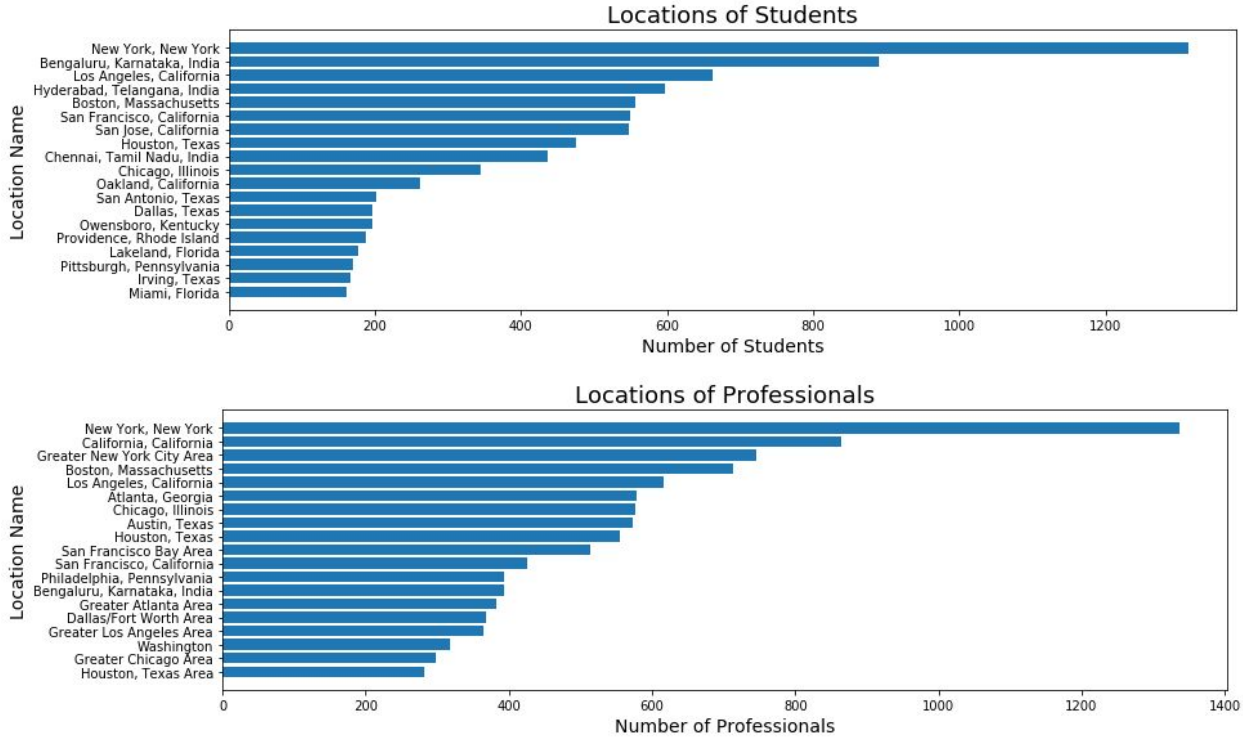

Figure 5. The most popular locations for students and professionals on CareerVillage. Note that the most popular location for both groups is New York, New York. Beyond that, many of the students are located in Bengaluru while much fewer professionals are also located in Bengaluru. Matching students from India with the more numerous professionals from California might not result in a follow-through from the professional if the question pertains to Bengaluru specifically.

We experienced a number of challenges in creating our system. To begin, technical issues have been ongoing resulting from the sheer size of the data. Transferring datasets became a challenge because GitHub has a size limit on its uploads, and various steps in our process were slowed by long runtimes caused by the large datasets. In addition some tools, such as the Rake function from nltk, didn’t work quite the way we anticipated. In particular Rake failed to clean our data in the way we expected, and so we had setbacks and had to put in extra time to clean the

data of things it missed such as HTML tags and unusual punctuation. Using a regex expression, we were able to expunge the rest of the non-keyword characters from the bag-of-words.

VI. Conclusion

While the process was not without its hiccups, this project was successful in demonstrating a basic content based recommendation system. Our output goal was to display the top ten related questions for a given professional. As a team, we generally consider this recommendation system to be a successful learning experience, however it is not complete enough to be considered as a system for the CareerVillage website. This is partly due to the way that content based recommender systems work-- because the system makes recommendations from previous user activity, the user is not encouraged to try answering questions that deviate from their norm. We also had to exclude a large number of professionals from the model who had not answered any questions previously, meaning that newly registered professionals would not receive any recommendations until they answered at least one question.

```
Importing......done!
Performing data cleaning...(this might take a few minutes)......pruning extra professionals....done!
generating keywords......done!
Beginning count vectorizer......done!
Now creating count matrix......done!
Now creating cosine similarity matrix......done!
enter a professional id# (just press return for a default ID for testing): 01b3e033848b41f6b7a55cefc59ba61a

10 questions to recommend to this professional:
How many years of school do you have to go through to be an vet?
How many years of school to I have to go through in total if i want to become a Pediatrician
I want to work with animals when I grow up.What can I do?
What is the hardest or most difficult thing to learn while studying for a Veterinarian?
Do veterinarians work with all animals or can they work with a certain group/species?
how many years do i have to be in school to be a lawyer?
what can i do to work with animals
What are some requirements to become a veterinarian?
what type of vet would you recommend the most?
Do you have to work with all kinds of animals when becoming a veterinarian?
```

Figure 6. Output results for our recommendation system.

Another aspect of this recommender system that lacks industrial functionality is method of comparing keywords. In a non-default result, we noticed that recommendations that could be considered outliers were likely matched because of the similarity in the phrasing of questions. In the output results image (figure 6), we can see a discrepancy among the recommended questions. Eight out of the ten recommended questions are related to veterinarian industries, which is to be expected since the titles of previously answered questions include: “What careers should I explore if I want to work with animals?”, “how many years did you go to school just to pursue a career as a veterinarian?”, “What importance does a PhD hold?”, and “What would be the most difficult thing about becoming a veterinarian?” However, two of these questions relate to fields outside of the clear “vet” trend. This could be attributed to the keyword match of the phrase “how many years of school” instead of the desired veterinarian keyword.

Despite fallbacks, this project proved to be a great insight into how the recommender systems of today’s modern era operate.

References

Herlocker, Jonathan L., Joseph A. Konstan, Loren G. Terveen, and John T. Reidl. "Evaluating Collaborative Filtering Recommender Systems." *ACM Transactions on Information Systems, Volume 22 Issue 1*, January 2004, Pages 5-53.

Nakkula, M., Danylchuk, L., Miller, K., & Tamerler, K. "Promoting Career Development with Low-Income Students of Color." *Compelling counseling interventions: Celebrating VISTAS' fifth anniversary*, 2008, pp. 115-124, https://www.counseling.org/resources/library/vistas/2008-V-Print-complete-PDFs-for-ACA/Nakkula_Article_12.pdf. Accessed 21 April 2019.

National Association for College Admission Counseling (NACAC), and American School Counselor Association (ASCA). *NACAC and ASCA State-by-State Student-to-Counselor Ratio Report.* National Association for College Admission Counseling (NACAC) and American School Counselor Association (ASCA). https://www.schoolcounselor.org/asca/media/asca/Publications/ratioreport.pdf. Accessed 23 April 2019.

Van den Broeck, J., Cunningham, S., Eeckels, R., Herbst, K. "Data Cleaning: Detecting, Diagnosing, and Editing Data Abnormalities". *PLos Med*. https://journals.plos.org/plosmedicine/article?id=10.1371/journal.pmed.0020267. Accessed 23 April 2019.

Rose et al, "Automatic Keyword Extraction from Individual Documents". *Text Mining: Applications and Theory*. Wiley, 2010.

Kwak, S. K., Kim, J. H. "Statistical data preparation: management of missing values and outliers". *Korean Journal of Anesthesiology*. 2017 Aug., pp. 407-411.

Gong, SongJie. "A Collaborative Filtering Recommendation Algorithm Based on User Clustering and Item Clustering." *Jsoftware.us*, Journal of Software, July 2010, www.jsoftware.us/vol5/jsw0507-9.pdf.

McDonald, David W, and Mark S Ackerman. "Expertise Recommender: A Flexible Recommendation System and Architecture." *ACM Digital Library*, University of California, Irvine, 1 Dec. 2000.

Ricci, Francesco, Rokach, Lior, and Shapira, Bracha. "Introductions to Recommender Systems Handbook". *Recommender Systems Handbook*. Boston: *Springer*, 2011. pp 1-35.