CareerVillage Questions Matching Recommendation System

Feb 2024

Minh Duong

Springboard DSC - Capstone Project 3

Project mentor: Bernard Chan

Table of content

Executive Summary

CareerVillage.org, a non-profit dedicated to career guidance, seeks to elevate its platform by implementing a data-driven recommendation system. This system will match student queries with the most appropriate volunteers, fostering connections between aspiring minds and experienced professionals. Leveraging powerful data science techniques, the project aims to optimize engagement and ensure students receive relevant, valuable advice from motivated mentors. This initiative holds the potential to significantly enhance CareerVillage's impact, empowering underserved youth with essential career role models.

The CareerVillage Question Recommendation Model is designed to efficiently connect students with professionals by assigning a given career-related question to a set of professionals that are most likely answering. This report outlines the methodology, data exploration, model development, and key findings.

Methodology

When it comes to recommending new content, there are three main types of systems at play:

Content-based recommendations: Like finding a good friend with similar tastes, this method recommends items that share features with things you've enjoyed before. Imagine it like browsing a store where items are grouped by genre or style. This approach works well when we have detailed information about both users and items (like product descriptions or movie reviews).

Collaborative filtering: Think of it like borrowing recommendations from your friends – this method suggests items based on what similar users have liked. So, if people with similar tastes to you loved a particular product, it might be recommended to you too. However, this requires a large number of users with explicitly expressed preferences, and might not work well for new users or niche interests.

Hybrid approach: Combining the strengths of both worlds, this method starts with content-based recommendations for new users, then as they interact with the system, switches to collaborative filtering to personalize suggestions based on their specific preferences. This offers a smooth experience for newcomers while leveraging the power of user-based recommendations over time.

To address privacy concerns, users' information, both personal and usage information, is neither stored nor utilized for training purposes. Consequently, a **content-based approach** will be employed for this project.

Two recommendation models were developed. Each model built a profile for professionals based on their past answered questions or associated tags. These profiles were then used to recommend questions in the test set to professionals in the training set by considering the similarity in content or tags. The effectiveness of these new models was assessed by comparing their recommendation success rate (measured by response rates) to the existing approaches.

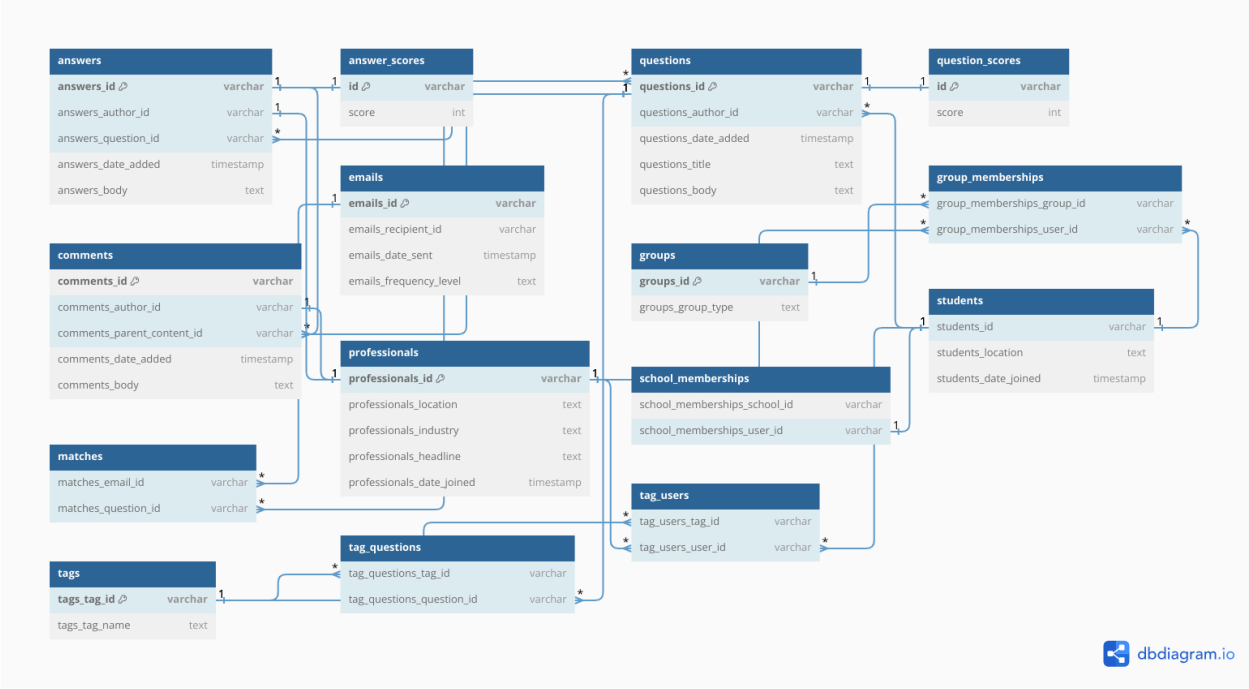
Data

Data Source & Description

This project utilizes data provided by CareerVillage, retrieved from Kaggle. The dataset encompasses valuable data tables such as:

- **questions**: Capturing the concerns and inquiries posed by young individuals seeking career guidance, which has 23931 questions
- **professionals**: Delving into basic information of the volunteers, which includes 28152 professionals
- **answers**: Unveiling the insights and advice shared by professionals in response to student questions, which contains 23110 records
- Professional Matching Inquiry involves tracking the associations established between questions and professionals through email interactions. This data is stored in two tables: **emails** and **matches**. The **matches** table reveals the assignment of questions to an email ID, while the **emails** table specifies which emails were dispatched to professionals. The amalgamation of data from these tables unveils details about the questions assigned to specific professionals. **matches** and **emails** are in long data format
- **tags, tag_questions and tag_users**: Categorizing individuals and inquiries based on relevant keywords and topics. Those tables are in long data format.

Below is the comprehensive visualization of the entire dataset. All of the tables were stored in csv format.



Data Cleaning & Preprocessing

The recommendation strategy was to compare the text similarity between the combined content of previous answered questions and their tags with the content and tags of the new questions, and then assigned each new question to 10 professionals that used to answer most similar questions.
The tables that were kept for this recommendation strategy: **questions, answers, professionals, tags, tag_users, tag_questions, emails, matches**. It was noticeable that tables **tags, tag_users, tag_questions, emails, matches** were in long format
The data preprocessing was the most time-consuming step in the project because data was stored across multiple tables. It required a lot of merging and examining to understand the relation between the tables and create data for modeling.

Determine the train/test split point

Creating a train/test set for this recommendation problem was a little different from the usual train/test split. I decided to use the questions before a certain cutoff day, which was the midpoint of the answers table by the answers date, for training purposes, and used the other half for model evaluation.
The goal was to build a train set that consisted of: professionals ids, the combined content of all questions that each professional had answered before, and the list of all tags associated with each professional.

Step 1: Create questions and professionals profiles in the train set:

- Questions profiles
    - I merged the **questions** and **answers** tables to create a set of **questions** that receive at least one answer.
    - Create a **question_tag_list** table by pivoting the **tag_questions** table, using question_id as the key and combining all the tags into a single list. The new table consisted of question ids and a list of tags associated with each of them.

- Professionals profiles:
    - I merged the **professionals** table with **tag_users** table to create a tags feature for the professionals.
    - I merged the **questions** and **answers** on the question ids, and then grouped the resulting dataframe by the professional ids, aggregating all the questions content into a single column. This created a dataframe with professional ids and the content of previous answered questions.
    - Combining two resulting data frames from the previous two steps produced the data frame that consisted of following features: professionals ids, list of tags and the content of previously answered questions.

Step 2: Create questions profiles for the questions in the test set

Using the same merging technique in creating questions and professionals profiles in the train set, I created profiles for questions and professionals in the test set.

Step 3: Text cleaning

To prepare text features for modeling, I created a text cleaning function that

- Remove special characters, stopwords, digits and leading/trailing white space.
- Lemmatized text
- Converted text to lowercase.

Step 4: Handling missing values

There were many professionals and questions that did not have any associated tags. This resulted in missing values when creating train and test data. To address this problem, two approaches were considered:

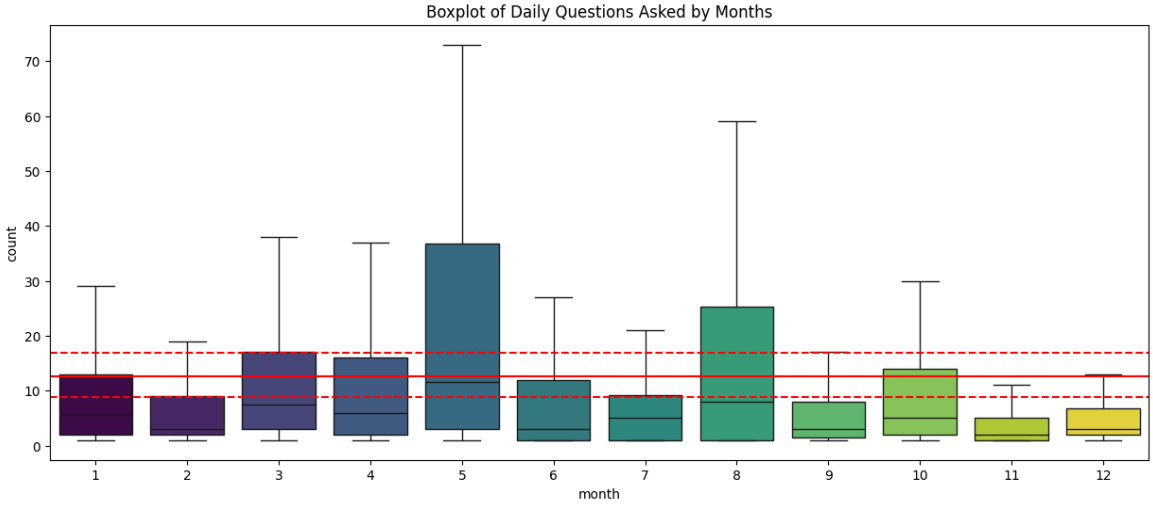
- Impute the missing tag values with “no_tag”
- Create an auto-tag-generated function to create tags for questions and professionals based on the question content.

In this project, I used the first approach for its simplicity. The second approach was considered for future improvement.

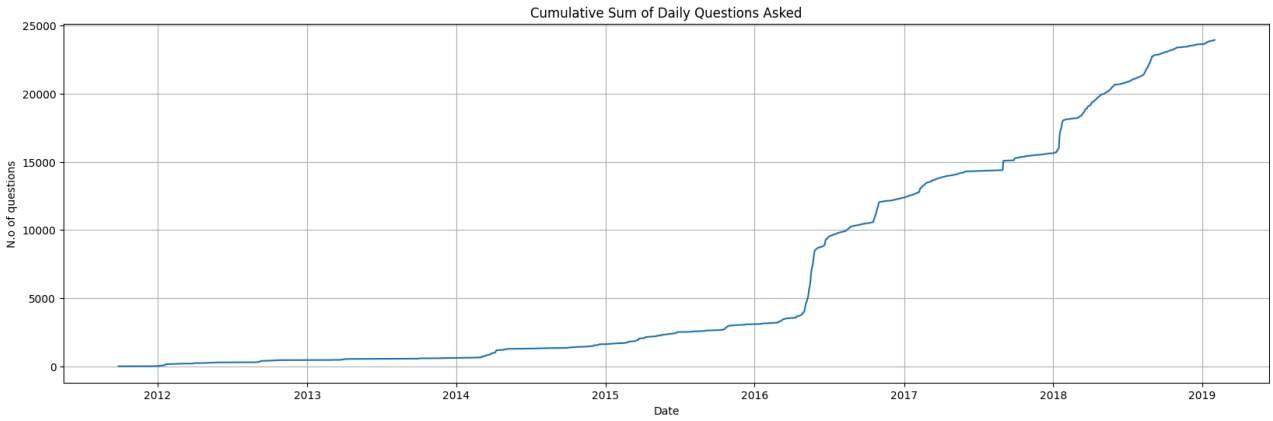
Exploratory Data Analysis (EDA)

While the recommendation system utilized natural language processing techniques, typically not demanding exhaustive exploratory data analysis, noteworthy discoveries emerged during the EDA process.

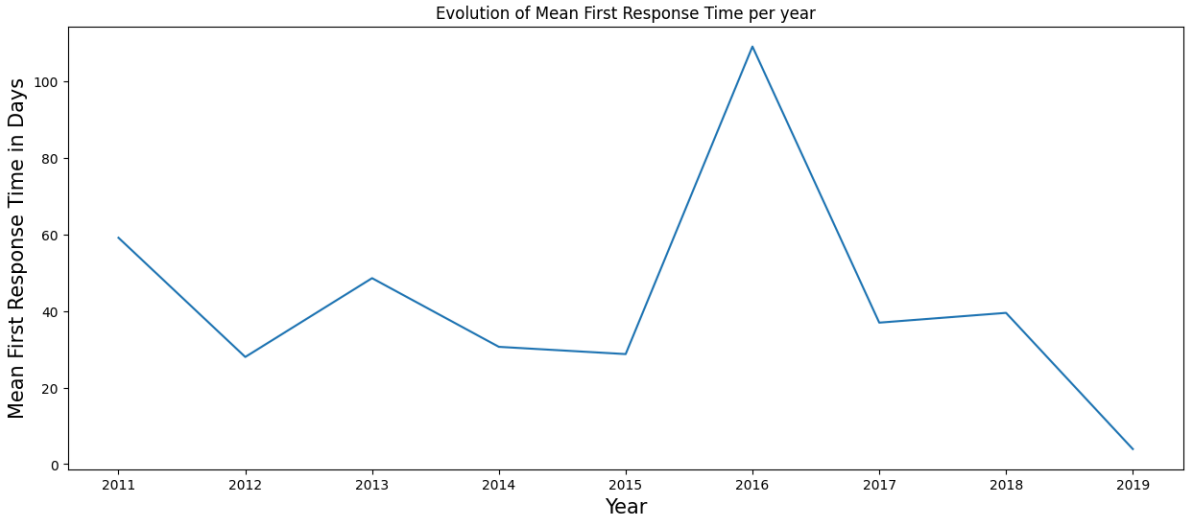
- The number of daily questions were peak in May, which was reasonable since May was the graduation month. January and August also had a higher number of daily questions asked compared to the other months.



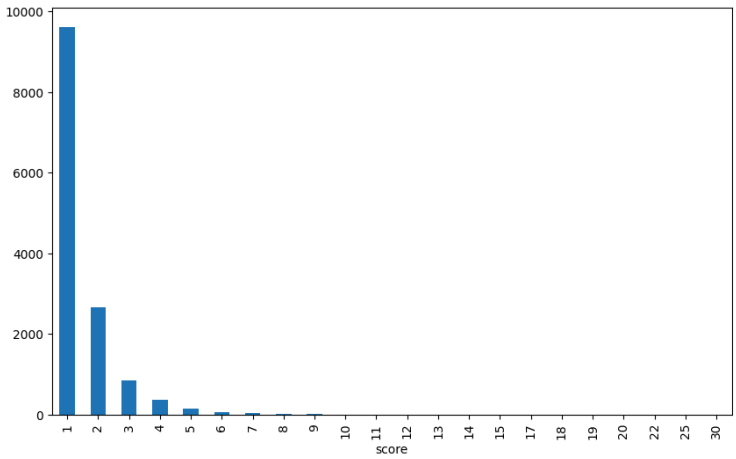
- There was a significant increase in the number of questions asked daily since the first half of 2016, which was due to the increase in the number of new user in 2016



- On average, a question is first answered within 65 days. About 25% of questions are answered within 24 hours after posting. Half of the questions are answered within 2 days.
- The mean response time for a questions changed overtime, and there was a big drop from 2018 to 2019



- The majority of responses garnered a score of 1, primarily due to the answer rating system prompting users to either upvote or downvote. This rating approach complicated the evaluation of answer quality, as lower scores could be attributed to fewer people reading a specific response, while higher scores might be a result of answers being present on the platform for an extended duration. Adopting a more effective answer rating scheme is advisable.



Modeling

Question-content based recommendation

- TF-IDF Vectorization

The textual content of questions in the train set was transformed into numerical vectors using TF-IDF (Term Frequency-Inverse Document Frequency) vectorization. This process assigned weights to words based on their importance in individual questions relative to the entire dataset, creating a numerical representation of question content.

- Model Training

The model was trained using the TF-IDF vectors and professional response data. During training, the system learned the relationships between the content features of questions and the preferences exhibited by professionals. This training phase aimed to establish a model capable of making accurate and personalized question recommendations.

- Recommendation Generation

The recommendation generation process involves assessing the content similarity between questions the professional has answered and those in the question pool using cosine similarity. The system identifies questions with similar content attributes and recommends them to the professional based on their historical preferences. The model generated a recommendation of 10 professionals for each question.

- Code snippet

```python
#Create a function that takes a set of professionals and a set of questions and calculate the cosine similarity scores.
#Return 10 questions with the highest similarity scores to each professional
def content_recommendation(professional_df,question_df):
    """
    This function take two inputs in pandas dataframe format
    professional_df['profesisonal_id','question_content']
    question_df['question_id','question_content']
    It will return a recommendation dataframe that assigns each question to
    10 most relevant professionals
    """
    # Vectorize text data using TF-IDF (could also use other vectorization methods)
    vectorizer = TfidfVectorizer(min_df=2, max_df=0.7)
    professional_vector = vectorizer.fit_transform(professional_df['question_content'])
    question_vector = vectorizer.transform(question_df['question_content'])

    # Calculate cosine similarity matrix
    cosine_matrix = cosine_similarity(professional_vector, question_vector)
    cosine_similarity_df = pd.DataFrame(cosine_matrix,
                                columns=question_df.index)
    cosine_similarity_df.index = professional_df.index
    recommendation_df = cosine_similarity_df.apply(lambda x: x.nlargest(10).index.tolist(), axis=1).explode().reset_index()
    recommendation_df.columns = ['professional_id', 'question_id']
    return recommendation_df
```

```python
#Create a recommendation dataframe for the professionals in the train set with the questions in the test set.
content_based_recommendation_df = content_recommendation(train_professional_df, test_question_df)
```

Tag-based recommendation

A similar model was built using the same modeling techniques as the content-based model. However, for the second model, I employed the tags of the professionals and questions to calculate the cosine similarity. I then compared the performance of the two models.

- Code snippet

```python
#Create a function that takes a set of professionals and a set of questions and calculate the cosine similarity scores.
#Return 10 questions with the highest similarity scores to each professional
def tags_based_recommendation(professional_df, question_df):
    """
    This function take two inputs in pandas dataframe format
    professional_df['profesisonal_id','tag_name']
    question_df['question_id','tag_name']
    It will return a recommendation dataframe that assigns each question to
    10 most relevant professionals
    """
    # Vectorize text data using TF-IDF
    vectorizer = TfidfVectorizer(analyzer='word',stop_words= 'english', min_df=2, max_df=0.7)
    professional_vector = vectorizer.fit_transform(professional_df['tag_name'])
    question_vector = vectorizer.transform(question_df['tag_name'])

    # Calculate cosine similarity matrix
    cosine_matrix = cosine_similarity(professional_vector, question_vector)
    cosine_similarity_df = pd.DataFrame(cosine_matrix,
                                columns=question_df.index)
    cosine_similarity_df.index = professional_df.index
    recommendation_df = cosine_similarity_df.apply(lambda x: x.nlargest(10).index.tolist(), axis=1).explode().reset_index()
    recommendation_df.columns = ['professional_id', 'question_id']
    return recommendation_df
```

```python
tag_based_recommendation_df = tags_based_recommendation(train_professional_tag,test_question_tag)
```

Model Evaluation

To comprehensively gauge the effectiveness of the recommendation models, it is imperative to establish a nuanced matching score that aligns a group of professionals with a designated set of questions. The criteria constituting a successful metric encompass multifaceted dimensions:

- Answer Rating: The assigned rating to the answer of a question serves as a fundamental aspect of the evaluation.
- Response Time: The duration taken to respond to an assigned question plays a crucial role in assessing the efficiency of the recommendation.
- Response Rate: Calculated as the ratio of the number of responses to the total number of questions assigned, the response rate provides insights into the engagement level with the questions.

The exploratory data analysis (EDA) phase accentuates the inadequacy of the current answer rating system in effectively evaluating the quality of answers. To fortify the evaluation's robustness, there is a pressing need to adopt a more sophisticated rating system before integrating answer ratings as a reliable measure of match. The optimal metric for success crystallizes as a synergistic combination of both response rate and response time.

In the initial modeling phase, simplicity steers the approach, with a primary focus on utilizing response rate as the predominant metric of success. This streamlined strategy facilitates a clearer understanding of the models' performance during the preliminary stages.

Determine the original response rates of the questions in the test set

- Merge the **emails** and **matches** on the email id to determine what questions were assigned to what professionals.
- Filter the result data frame with professionals in the train set and questions in the test set. The result data frame, **test_question_w_answer_set**, consisted of pairs of questions with each of its answers.
- Create a set of questions in the test set that received an answer. Using this set to filter the question,answer pair in the previous data frame.
- With the result data frame, it could be determined whether a question that was assigned to a professional through email received a response.
- ***Average response rate of the original matching system: 0.0038050571864699247***

Determine the response rates of the questions in the test set with question content based recommendation

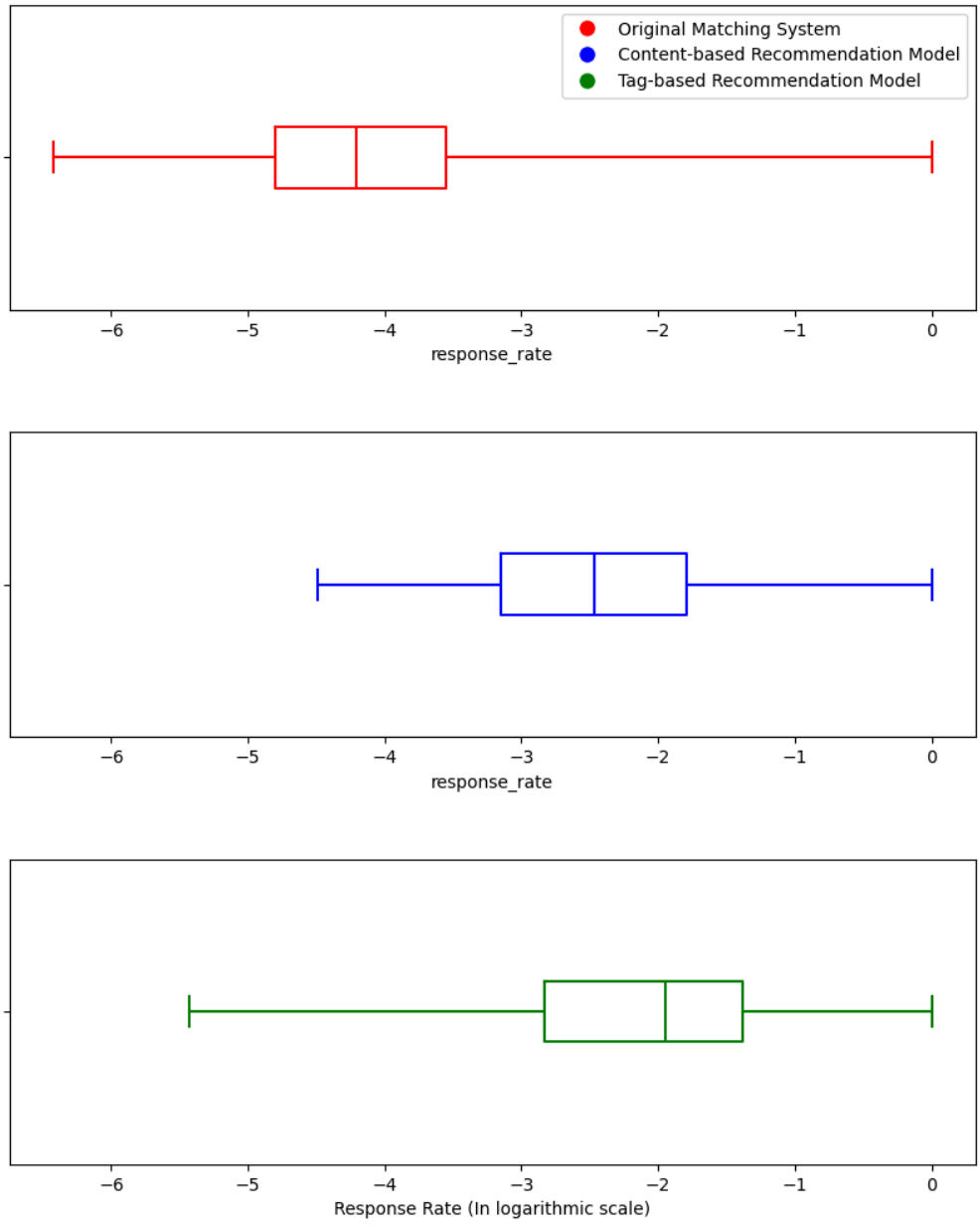
- The model recommended a question to 10 most relevant professionals.
- Comparing the professional, question pairs in the recommendation result with the set of questions that
- ***The the average response rate of the question content-based recommendation model: 0.0042015005359056804***

Determine the response rates of the questions in the test set with tags based recommendation

- Using the same method of determining the response rate in th question content based recommendation model, ***the average response rate of the tags-based recommendation model: 0.0059592711682743835***

Furthermore, the comparison of average response rates for each question involves creating boxplots for the original system and two recommendation models, namely, the question content-based recommendation model and the tags-based recommendation model. The rates are transformed to a logarithmic scale due to their proximity to zero.

Boxplots of response rates of the original matching and the new recommendation models



Future Steps

- Integrate the response time into the successful metric.
- Considers professional activeness (the amount of time between each activity of a user) as a feature for the model.
- Comprising professionals previous response into the training phase.
- Dynamic tagging: Implement a dynamic automatic tagging system to adapt to evolving professional expertise.
- User feedback loop: Develop a mechanism to collect and incorporate user feedback for continuous improvement.
- Improve user rating mechanism.
- Enhanced collaborative filtering: Explore advanced collaborative filtering techniques to improve personalized recommendations.

Conclusion

In conclusion, the CareerVillage Questions Matching Recommendation System project has successfully laid the foundation for a data-driven recommendation system aimed at connecting students with professionals on the CareerVillage.org platform. The project employed content-based recommendation techniques, utilizing natural language processing and machine learning to match career-related questions with professionals likely to provide relevant and valuable insights.

The methodology involved data preprocessing, creating profiles for questions and professionals in the train set, and leveraging TF-IDF vectorization for question-content-based recommendations. Additionally, a tag-based recommendation model was developed, considering the similarities in tags associated with questions and professionals.

The exploratory data analysis (EDA) phase revealed insights into question trends, response times, and the limitations of the existing answer rating system. The evaluation criteria included answer rating, response time, and response rate, emphasizing the need for a nuanced matching score.

The models were tested against the original matching system, and their average response rates were compared using logarithmic scale boxplots. The question content-based recommendation model demonstrated a response rate of 0.0042, while the tags-based recommendation model achieved a response rate of 0.00596.

Future steps involve integrating response time into the success metric, considering professional activeness, incorporating previous responses into training, implementing dynamic tagging, establishing a user feedback loop, enhancing the user rating mechanism, and exploring advanced collaborative filtering techniques.

Overall, the project marks a significant stride toward enhancing the CareerVillage platform, empowering students with valuable career guidance and fostering meaningful connections between students and professionals. The recommendations and insights derived from this project provide a solid foundation for future enhancements and improvements to ensure a more personalized and impactful user experience on CareerVillage.org.