Prediction of Popular Questions on CareerVillage

Group 16: Liu Ye (A0262703L), Nandhini Selvaganapathy (A0262710N), Shen Xiaohan (A0262717B), Wang Haiting (A0262764X), Zuo Jiaxin (A0262833A)
GitHub link: https://github.com/leia-liu/BT5153_Group16_2023

Abstract

By connecting students with professionals, CareerVillage.org offers an online community where users can ask and receive answers to career-related questions. However, as the volume of questions on the platform increases, maintaining question quality has become a challenge. In this study, we aimed to improve question quality and revitalize user engagement by employing machine learning techniques to predict question popularity on CareerVillage.org. We conduct exploratory data analysis (EDA) to identify trends and associations between question scores and various features. Based on these insights, we perform feature engineering and construct predictive models using logistic regression, random forest, XGBoost, and neural networks, with three different feature sets: numerical features only, numerical and TF-IDF text representation, and RoBERTa embeddings. We find that adding text representations from TF-IDF or RoBERTa slightly improves the performance of models compared to using only numerical features. However, dimensionality reduction using PCA negatively impacts model performance. Our study then offers valuable insights for CareerVillage.org to adjust its traffic distribution strategy, expand its user base, and provide tailored suggestions to users for improving question popularity. The report concludes by discussing the limitations of the project and suggesting directions for future work.

1. Introduction

1.1 Background

CareerVillage.org is a non-profit organization that aims to bridge the gap between underserved youth and career guidance by providing them with free career advice from experts in various industries. These young people lack knowledge and social capital, which makes it difficult for them to make informed decisions about their educational and career choices. Through the platform, students can ask career-related questions, and voluntary professionals will answer based on their interests and expertise. To date, CareerVillage.org has provided over 3.5 million online learners with career advice from more than 25,000 guidance counselors.

This increase in volume has led to a new challenge of maintaining question quality, with potential duplicates and irrelevant questions appearing on the platform. This not only wastes scarce professional resources but also makes it difficult for users to identify the most relevant and useful information, ultimately diminishing the value of the platform.

1.2 Problem Statement

Therefore, the primary objective of our project is to improve the quality of questions asked by students on CareerVillage.org (measured by the scores of questions). To reach this objective, we build a machine learning model to predict whether a question will be popular. Our model should maximize the classification precision in predicting the popularity of questions. The direct impact of this model is:

1) The questions that are predicted popular are the ones that have the potential of being popular. They should be recommended to more users to increase content exposure.

2) The questions that are predicted not popular could receive suggestions on how to improve the question quality.

This enables CareerVillage.org to identify relevant and popular contents more efficiently. Achieving this goal helps increase user satisfaction, measured by net promotion scores (NPS) of students and professionals, and user engagement, measured by daily and monthly active users (DAU & MAU), which ultimately makes CareerVillage.org a more impact community in career development.

2. Dataset Description

The study's primary data source is Kaggle on "Data Science for Good: CareerVillage.org" (Feb 27, 2019). The company CareerVillage.org provided eight years of anonymized data from 2011 to 2019 to develop methods for question recommendation. The dataset comprises fifteen files, but only nine were used for relevance, as listed in Table 1 below.

Table 1. Description of used files

File Name	File Shape	Description	Features
questions	(23931,5)	questions are posted by students and are relevant to their future professional success	(questions_) id, author_id, date_added, title, body
question_scores	(23928,2)	'hearts' score for each question	id, score
tag_questions	(76553,2)	hashtag-to-question pairings	(tag_questions_) tag_id, question_id
tags	(16269,2)	the name of each tag	(tags_tag_) id, name
tag_users	(136663,2)	shows which hashtags each user follows	(tag_users_) tag_id, user_id
answers	(511235,5)	answers get posted in response to questions. answers can only be posted by users who are registered as professionals	(answers_) id, author_id, question_id, date_added, body
answer_scores	(51138, 2)	'hearts' score for each answer	id,score
comments	(14966, 5)	comments that are made on answers or questions	(comments_) id, author_id, parent_content_id, date_added, body
professionals	(28152, 5)	grown ups who volunteer their time to answer questions on the site	(professionals_) id, location, industry, headline, date_joined

The two primary files used are "questions" and "question_scores". Some numerical, categorical and text features can be extracted from the "question" file, and our target variable is generated from "score" in the "question_scores" file. Some other numerical and text features can be generated from the remaining files. The feature extraction methods will be elaborated on in sections 4.2 and 4.3.

3. Exploratory Data Analysis

To better understand the data, we performed exploratory data analysis on aforementioned datasets.

3.1 The Distribution of Scores (Figure 1)

The distribution of scores in the dataset appears to be right-skewed with a long tail. This means that the majority of the questions have low scores, while a small percentage of questions have relatively high scores. The tail of the distribution represents these high scoring questions, which are outliers in the dataset. This suggests that there are certain questions that are particularly well-received by the community and generate a lot of engagement, while most questions receive relatively little attention.

3.2 The Average Scores Decreasing Trend (Figure 2)

The trend of average scores per year in the dataset shows a general decrease over time. It suggests that the community's engagement with questions on the platform is declining. It is necessary to find out the factors contributing to popular and non-popular questions so that we can design strategies to revitalize user engagement.

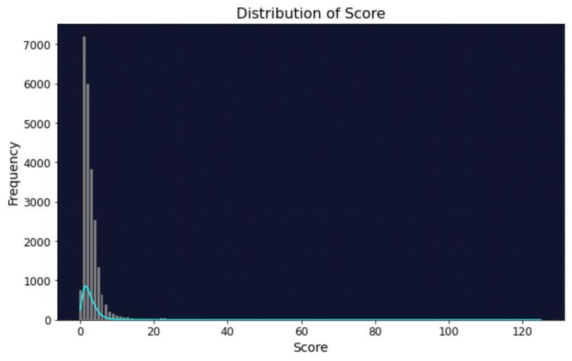

Figure 1. The distribution of scores

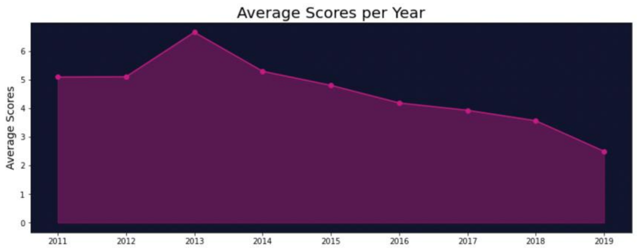

Figure 2. The average scores decreasing trend

3.3 Correlation Heatmap (Figure 3)

The correlation heatmap provides a graphical representation of the correlation matrix between different numerical features. It helps to identify the features that are most correlated with the score assigned to each question. Specifically, the heatmap shows that the score has a relatively strong correlation with the number of answers received by the question and the maximum score assigned to any answer for that question. This insight can inform feature selection later.

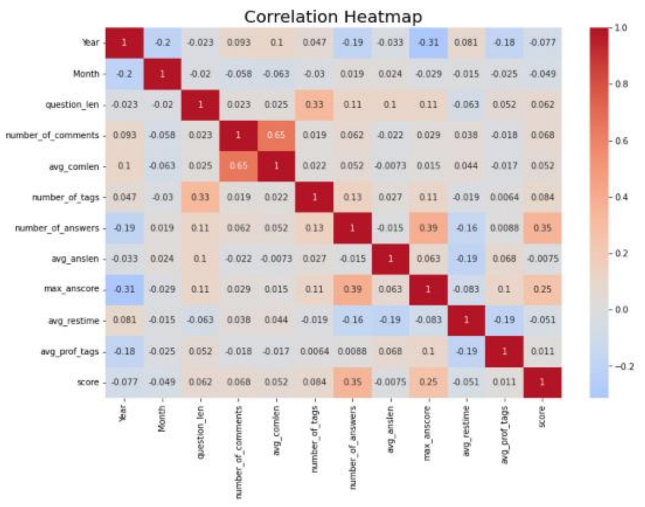

Figure 3. Correlation heatmap

3.4 Two Classes (Figure 4)

As our goal is to establish a classification model for predicting question popularity, it is essential to divide the dataset into two classes, popular and non-popular cases. With the third quartile score being 3 and considering the highly skewed distribution, we set the popularity threshold at 4. Questions scoring 4 or above are classified as popular, while those below 4 are deemed not popular. We then employ various methods to generate a balanced dataset, which will be detailed in section 4.4. The histogram plot of the label distribution shows the number of questions in each class after resampling. Popular questions are marked with 1, and not popular questions are marked with 0. Balancing the dataset helps enhance the model's accuracy in predicting both classes, and subsequent analysis will rely on the newly formed classes.

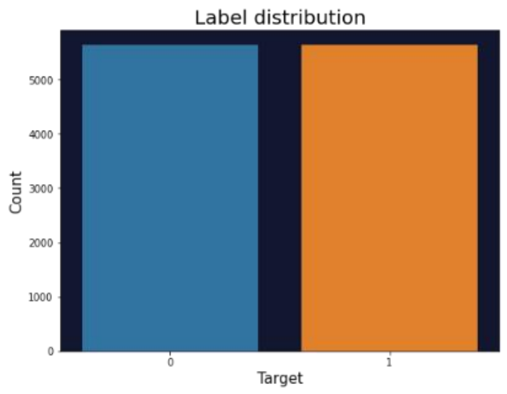

Figure 4. Label distribution

3.5 Three Box Plots (Figure 5 - 7)

The three boxplots display the distribution of three numerical features (Table 2), namely “number_of_tags”, “number_of_answers”, and “max_anscore”, for popular and non-popular questions. The boxplots demonstrate that high score questions tend to have more tags and receive more attention and better answers than low score questions. The most significant difference is observed in the number of answers received by the question, which is consistent with the correlation heatmap generated earlier.

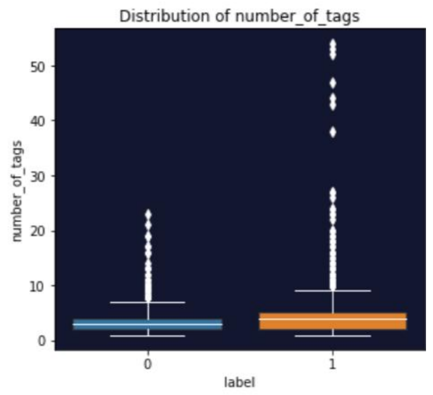

Figure 5. Distribution of “number_of_tags”

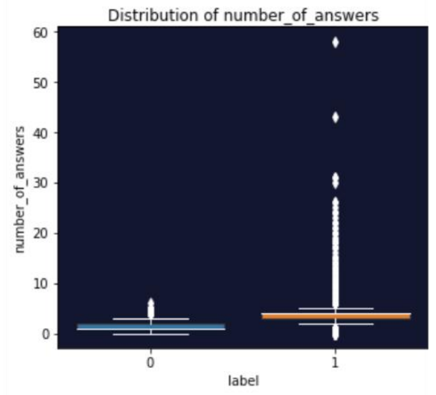

Figure 6. Distribution of “number_of_answers”

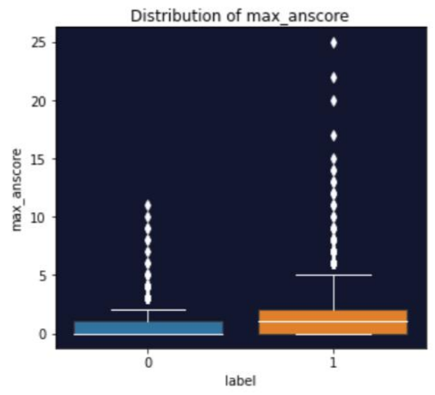

Figure 7. Distribution of “max_anscore”

3.6 Top Ten Tags Comparison (Figure 8 - 9)

The two graphs compare the top 10 mentioned tags in the high and low score groups to provide insight into the types of questions that tend to score higher. The results show that popular questions typically involve computer, technology, and business topics, while non-popular questions pertain to discipline of medicine. It suggests that users of the platform are more likely to engage with computer-related topics, which can help inform content creation and user engagement strategies.

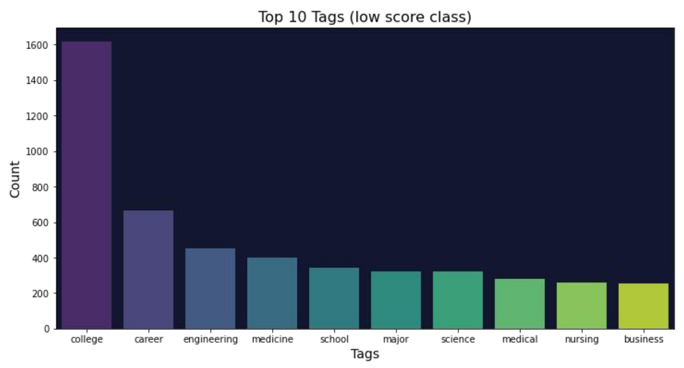

Figure 8. Top 10 tags for low sore class

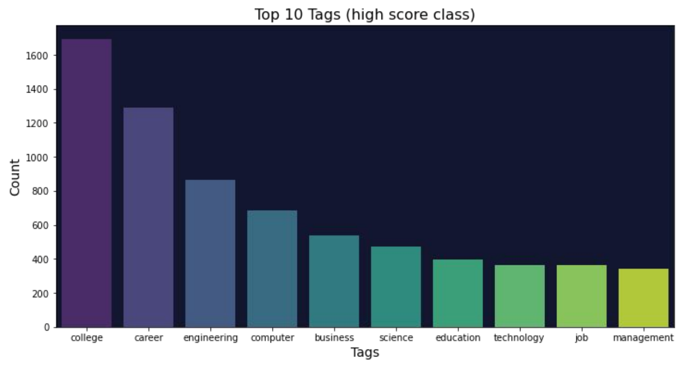

Figure 9. Top 10 tags for high score class

3.7 Word Cloud for Low & High Score (Figure 10-11)

Word clouds of low and high score questions reveal similarities, with both featuring common question-related words. This observation may be attributed to the similarity of question templates across both groups, which implies that differences might lie in the specific topics and content.

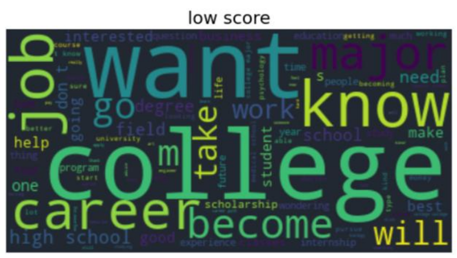

Figure 10. Word cloud for low score class

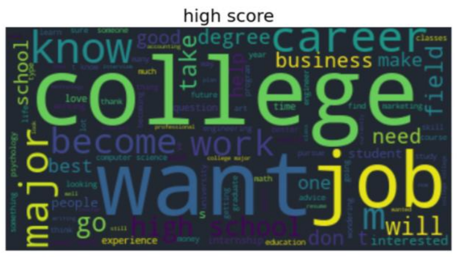

Figure 11. Word cloud for high score class

4. Methodology

4.1 Data Cleaning

At the beginning of our data cleaning process, we focused on time variables. Since we needed to extract the year and month from the “questions_date_added” in the “questions” file and also obtain the “answers_date_added” from the “answers” file to calculate the response time to each question, we transformed these two variables into an actual datetime format. This conversion was critical for us to perform temporal analyses and derive meaningful insights from the data.

Regarding text variables, which included “questions_title”, “questions_body”, “tags_tag_name”, “answers_body” and “comments_body”, we applied several cleaning steps to preprocess the text data. Firstly, we used the BeautifulSoup package to remove HTML tags, ensuring that only the actual text content remained. Then, we eliminated digits, punctuations, and other special symbols that do not carry significant meaning. Finally, we converted each letter to its lowercase, standardizing the text data and reducing redundancy.

These cleaning procedures were critical not only for vectorizing text into numerical features but also for calculating the length of the text. For example, we could determine how many words each answer or comment contained, which could provide valuable insights into the quality and depth of the response.

4.2 Numerical and Categorical Features Engineering

To improve the predictive power of our machine learning models, we aimed to generate relevant numerical and categorical features based on our prior exploratory data analysis. Our approach involved merging different data sources with our primary file, "questions", to extract features related to different aspects of questions, such as the question text, comments, answers, tags, and professional involvement.

For categorical variables, we extracted the year and month from the date when each question was asked and used one-hot encoding to convert them.

For numerical variables, we employed various methods to extract information from the different aspects of a question.

1) **Question text.** We combined the "question_title" and "question_body" columns to create a new feature, "question_whole", and calculated the number of words in this feature, which we named "question_len";

2) **Comments.** We hypothesized that questions with a higher number of comments and more high-quality comments are more likely to become popular on the platform. Therefore, we generated two new features: "number_of_comments" and "avg_comlen". The former represents the total number of comments received by a question, while the latter represents the average length of comments;

3) **Answers.** Section 3.3 and 3.5 proves that questions with more answers, particularly those with popular and lengthy responses, are more likely to become popular on the platform. As a result, we calculated the number of answers per question, "number_of_answer" and their average length, "avg_anslen". Additionally, we considered the highest score among the answers to each question, "max_anscore", and the average response time in days, "avg_restime";

4) **Tags.** We also recognized the potential value of tags in providing insights into popular topics. Therefore, we extracted the number of tags used per question and generated a new feature called "number_of_tags". We also combined all the tags together as "tags_tag_name" for future usage in our modelling;

5) **Professionals.** Our hypothesis was that if a professional who answered a question used a substantial number of tags, it could increase the likelihood of students agreeing with the answer and giving it a higher "hearts" score. To test this idea, we generated a new feature called "avg_prof_tags", which represents the average number of tags all professionals used per question.

To ensure that all the newly-generated numerical features were given similar weight in the modeling process, we used StandardScaler to scale them to a mean of 0 and a standard deviation of 1. This preprocessing step allowed us to avoid placing more importance on variables with higher values and ensured that all features were given equal consideration during the model training.

All the features mentioned above are listed in Table 2.

Table 2. Generated features

	Variable	**Descriptions**
Categorical	Year	year when the question was asked
	Month	month when the question was asked
Numerical	question_len	length of question_whole
	number_of_comments	total number of comments per question
	avg_comlen	average length of comments
	number_of_answers	total number of answers per question
	avg_anslen	average length of answers
	max_anscore	the highest score among the answers per question
	avg_restime	average response time in days
	number_of_tags	number of tags used per question
	avg_prof_tags	average number of tags all professionals used per question
Text	question_whole	question title and body
	tags_tag_name	all the tags used per question

4.3 Text Features Engineering

We explored two different methods for converting the two text variables in Table 2 into numerical features. The first method involved TF-IDF Vectorization, which assigns weights to words based on their importance. We applied this method separately to "tags_tag_name" and "question_whole", resulting in 11330-dimensions and 3102-dimensions vectors, respectively. However, we faced the challenge of high dimensionality, so we performed Principal Component Analysis (PCA) on the TF-IDF matrix to reduce the dimensions.

To overcome the high dimensionality issue, we also explored obtaining embeddings from a pre-trained RoBERTa model (roberta-base). We set the maximum number of words to tokenize to 256 for "question_whole" and 64 for "tags_tag_name". Then, we extracted the embeddings for the [CLS] token as the final embeddings for each feature. The embeddings obtained from the roberta-base model were much lower in dimensionality (768 dimensions) compared to TF-IDF Vectorization. We also applied PCA to the tags embeddings and reduced the dimensions to 10. Ultimately, the two text features added 778 dimensions to the input features of our models.

4.4 Balancing the Data by Sampling

During the data exploration phase, we observed that the dataset was highly imbalanced, with only 25% of the questions labeled as popular. This class imbalance could result in a classifier that is biased towards the majority class, leading to poor classification performance on the minority class (popular questions), which is what we were primarily interested in predicting.

To address this issue, we first attempted to use Synthetic Minority Over-sampling Technique (SMOTE) to oversample the minority class. However, the performance of the model on the minority class was still unsatisfactory, indicating that a different approach was needed.

Given that there were sufficient data points, we then opted for a simpler approach and randomly sampled from the majority class to make it have the same number of data points as the minority class. This approach resulted in a balanced dataset with 5635 records in each class, totaling 11,270 records.

By creating a balanced dataset, we increased the chances of the classifier learning patterns in the minority class, resulting in better performance on predicting popular questions.

4.5 Train Test Split

To assess and contrast the effectiveness of diverse prediction models, the train-test split approach is implemented on the finalized and balanced dataset. Specifically, the dataset is split into two subsets, the training and testing sets, where the training set constitutes 80% of the dataset, and the remaining 20% is used as the testing set. The training set is used to build the models, while the testing set is utilized to evaluate the models' performance on new data.

5. Modeling

5.1 Models

To train our models, we utilized four classification methods, namely logistic regression, random forest, XGBoost, and neural network. While the first three models do not have any exceptional characteristics, further elaboration will be provided on the neural network model.

The neural network model employed in this study comprises three hidden layers with 64, 128, and 64 nodes, respectively, enabling the model to learn complex patterns in the data. The use of batch normalization and dropout regularization techniques enhances the model's generality. The output layer of the model is comprised of two nodes, utilizing a softmax activation function for effective classification. To optimize the model, the Adam optimizer was employed with sparse categorical cross-entropy used as the loss function, while accuracy was chosen as the evaluation metric. The model was trained on preprocessed training data with a batch size of 100. Furthermore, early stopping was employed to prevent overfitting. Training stopped if the validation accuracy failed to improve for three consecutive epochs, with a maximum training limit of 15 epochs.

To assess the performance of different features as inputs to our models, we employed a three-step approach:

1) Firstly, we excluded text features and included categorical features such as “Year” and “Month”, along with numerical features such as “question_len” and “number_of_comments”. We applied all four models to determine the predictive power of these variables.

2) Secondly, we incorporated text features into our models through two approaches: TF-IDF and RoBERTa embeddings. We utilized logistic regression and random forest for TF-IDF to analyze the influence of words. After applying PCA to the TF-IDF approach, we only used logistic regression to evaluate whether dimensionality reduction affected model performance. For the RoBERTa embeddings approach, we employed four models to obtain optimal result.

3) Lastly, we applied PCA to reduce the dimension of our second method to match that of the first method. We applied logistic regression to examine the effect of dimension reduction on the model performance.

5.2 Models Performance

We assessed the performance of our models using precision, recall, f1 score, and accuracy metrics. It should be noted that the balanced nature of our dataset resulted in equal metric scores. Our analysis indicated that the model excluding text features showed the best performance, with a score of 0.87 achieved by random forest. For the model utilizing TF-IDF vectorization, logistic regression performed the best with a score of 0.86. However, we observed a decline in performance by 0.26 after implementing dimension reduction via PCA. Among models using RoBERTa embeddings, logistic regression, XGBoost, and neural network showed equal performance with a score of 0.86. However, we observed a decrease in performance to 0.71 after applying PCA.

5.3 Tables

The following tables summarize the performance for each model.

Table 3. Evaluation metrics for models excluding text features

METRICS	LOGISTIC REGRESSION	RANDOM FOREST	XGBOOST	NEURAL NETWORK
PRECISION	0.86	**0.87**	0.86	0.86
RECALL	0.86	**0.87**	0.86	0.86
F1 SCORE	0.86	**0.87**	0.86	0.86
ACCURACY	0.86	**0.87**	0.86	0.86

Table 4. Evaluation metrics for models including text features (TF-IDF)

METRICS	LOGISTIC REGRESSION	RANDOM FOREST	LOGISTIC REGRESSION (AFTER PCA)
PRECISION	**0.86**	0.85	0.6
RECALL	**0.86**	0.85	0.6
F1 SCORE	**0.86**	0.85	0.6
ACCURACY	**0.86**	0.85	0.6

Table 5. Evaluation metrics for models including text features (RoBERTa)

METRICS	LR	RF	XGB	NN	LR (AFTER PCA)
PRECISION	**0.86**	0.84	**0.86**	**0.86**	0.71
RECALL	**0.86**	0.84	**0.86**	**0.86**	0.71
F1 SCORE	**0.86**	0.84	**0.86**	**0.86**	0.71
ACCURACY	**0.86**	0.84	**0.86**	**0.86**	0.71

6. Insights

6.1 Results Interpretation

For the models that does not include text features, i.e., TF-IDF or RoBERTa sentence embeddingss, random forest has the best performance. Random forest is followed by neural network, XGBoost and logistic regression, where the three models have comparable performance.

As for the models that includes TF-IDF, logistic regression has a slight advantage over random forest. Given that the training f1 score of random forest model in this case is 1.00 and that of logistic regression is 0.90, the superior performance of logistic regression can be attributed to random forest being overfitting.

Applying PCA in the models including TF-IDF, we can observe a sharp decrease in the metrics of logsitc regression. This is because terms (words) in the document-term matrix are relatively independent. Reducing the dimension of the input data eliminates a large amount of variation in the data, causing a loss of information.

For the models using RoBERTa embeddings, logistic regression, XGBoost, and neural network has the same performance. This may be attributed to the fact that the sentence embeddings generated by pre-trained RoBERTa already provide good representations for the input sentences. The classification task is therefore easy and models with low complexity, e.g., logistic regression, can achieve similar performance as the models with high complexity, e.g., neural network.

After performing PCA on the text features, our logistic regression model witnesses a steep drop in model metrics. This is due to the fact that RoBERTa embeddings provide a succinct representation for each sentence. Further reducing the dimension of the embeddings lead to significant information loss, which explains the worsening performance of the logistic regression model. Also, PCA assumes that a linear relation between the original features, with which it constructs principal components to represent them. However, RoBERTa embeddings are the weights of the last hidden state of the transformer model, which does not necessarily have linear relationships between dimensions. Therefore, PCA may not be a proper way to reduce the input dimension in our case.

We also calculated the feature importance for the random forest model in the TF-IDF approach to gain insights into the impact of words on model prediction. The top 20 features were plotted, and it was found that the top 8 features were non-text features, with some text features appearing among the top 20. This aligns with our model results that the model without text features already performed well with a score of 0.86, and including text features did not lead to a significant improvement in model performance. Furthermore, most of the text features were general words like “career”, “want”, and “college”, with only “computer” being related to a specific discipline. This finding is consistent with our EDA results.

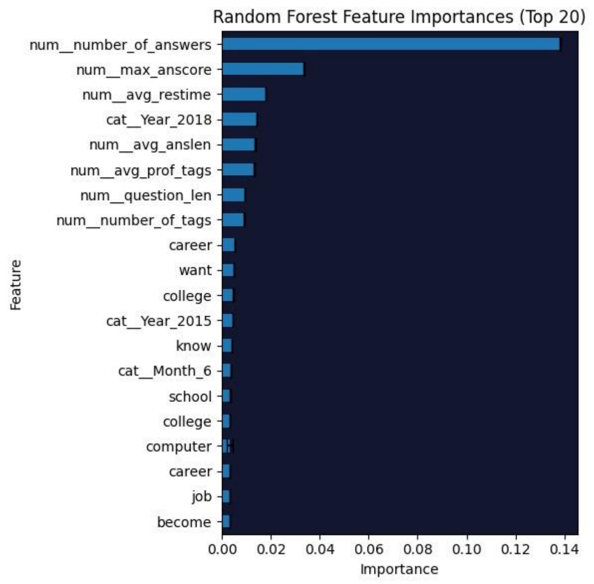

Figure 12. Feature Importance of Random Fores

6.2 Business Application

Our project highlights a few measures CareerVillage can do.

First, the right skewness of the score distribution and the decreasing trend of the average score revealed in our EDA suggest that most questions are not attracting much attention from the user community. At the meantime, EDA also shows that the questions with high scores use highly similar words with the questions with low scores. This show that the content of two types of questions may be similar, while the disparity in scores may be a result of the traffic distribution strategy of CareerVillage. In other words, the questions with low scores do not necessarily have low quality. It is just that when some questions attract attention in the first place, they get pushed to the top of the website by the recommendation system of CareerVillage, magnifying the difference in traffic and scores. This indicates that CareerVillage may adjust the distribution of its traffic to offer more exposure to the questions that are not receiving high scores at the moments, as they are potentially good questions. This also enriches the websites with questions of different categories.

Another implication is that, as questions tagged with computer, technology and business are likely to attract more attention, CareerVillage can suggest students who want to ask these questions to include these tags in their questions. This allows them to attract more traffic to their questions. The opposite side of the above finding also worth notice. As questions related to medicine are receiving low scores, CareerVillage should consider expanding its user base among medical students. This can be done by 1) targeted promotion and ads in medical school communities; 2) invite more medical related professionals to CareerVillage to provide high quality answers.

The importance of our model lies in the questions that are predicted popular but are in fact not popular. Such questions indeed have the potential of being popular, e.g., good topic, well-defined questions, popular disciplines, but are currently not attracting enough attention as they deserve. For these questions, CareerVillage can actively recommend them to professionals in the related field so that these questions can receive enough attention. By having professionals to answer them, CareerVillage can inject answers of high quality to these questions and make other users enjoy these questions. This will help increase the score of such questions.

Our model can also benefit students who ask questions by offering suggestions on improving its popularity. Namely, for the questions that are predicted not popular, CareerVillage can offer suggestions based on the feature importance of the model and the LIME prediction explanations. For example, one of most common reasons for questions to be predicted not popular is the short length of the question. If LIME suggest that a question is not popular due to its short length, CareerVillage can suggest the student to enrich the question with more details. Another example is that if LIME indicates that a question is not popular because of its few tags, CareerVillage can simply suggest the student to include more tags in the question so that the question can be recommended to more professionals in the related field.

7. Limitations & Future work

While developing our predictive model, we focused on simpler models such as logistic regression and random forest, rather than exploring more novel methods such as Text CNN and GCN. However, we might benefit from trying these approaches to see if they could help us improve our predictions further.

Another limitation of our study is that we determined the popularity threshold by considering the low frequency of the "heart" function's usage. We based this decision on the fact that, on CareerVillage, even the most popular question only had a "hearts" score of 125, which is relatively low compared to other Q&A platforms. In the future, if CareerVillage sees a significant increase in "heart" usage, we may need to reevaluate our threshold and re-examine our findings

It's worth noting that our analysis is more applicable to questions that have been posted for some time, as we could only collect features from answers and comments after a question was published. To address the need for predicting popularity for newly asked questions, we could consider using only features from the question aspect, and exploring other datasets like "students" to build a predictive model based on limited features. This would enable us to identify popular questions as soon as they are posted.

8. Conclusion

The primary goal of our project is to build a machine learning model to predict whether a question on CareerVillage is a popular one. To achieve so, we first conduct EDA to analyse the trend and characteristics of the data, which reveal the association between question scores and numerical features, e.g., the question length and the number of tags. Based on the insights gained from EDA, we leverage feature engineering to extract features that are relevant and meaningful for modeling. We then construct the model in three methods: 1) using only numerical features; 2) using numerical features and TF-IDF text representation; 3) using RoBERTa features. We find that while using only numerical features already allows us to highly accurately predict whether a question will be popular, adding text representations from TF-IDF or RoBERTa indeed improves the performance to some extent. However, conducting dimension reduction with PCA has a strong negative impact on the model performance. Our project contributes to CareerVillage both qualitatively and quantitatively. Through EDA, we suggest CareerVillage to adjust its traffic distribution strategy and expand its user base. Through modeling, we build a model that can predict whether a question will be popular or not. For the questions predicted popular but are not yet popular, CareerVillage can offer more exposure. For the questions predicted not popular, our model help CareerVillage make suggestions to students to make the questions more popular.

References

Data Science for Good: CareerVillage.org.

Kaggle. (2019). Retrieved April17, 2023, from

https://www.kaggle.com/competitions/data-science-for-good-careervillage/overview

Scikit-learn TfidfVectorizer documentation.

Retrieved April 18, 2023, from

https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

Huggingface RoBERTa introduction.

Retrieved April 18, 2023, from

https://huggingface.co/docs/transformers/model_doc/roberta

Yoon Kim. (2014). Convolutional Neural Networks for Sentence Classification

Retrieved April 21, 2023, from

https://arxiv.org/abs/1408.5882

Liang Yao, Chengsheng Mao, Yuan Luo. (2018). Graph Convolutional Networks for Text Classification.

Retrieved April 21, 2023, from

https://arxiv.org/abs/1809.05679