Analysing response time of StackOverflow question

Abhishek Vyas , MT19086
M.Tech CSE
IIIT Delhi
abhishek19086@iiitd

Chirag Chawla , MT19089
M.Tech CSE
IIIT Delhi
chirag19089@iiitd

Mansi Sharma , MT19092
M.Tech CSE
IIIT Delhi
mansi19092@iiitd

1 Problem Defination

The online educational platform has grown widely across the country. The Users of Questions and answers websites may include code fragment or unexplanatory words in the Questions which makes it sometimes harder to understand those questions and take time to answer them. In this Project we try to find the influence of the question in respect to answering time.

2 Problem Background

Stack Overflow is an open community for anyone that codes. They help us get answers to your toughest coding questions, share knowledge with your coworkers in private, and find your next dream job. Nowdays most of coders use stack overflow when stuck in some part of coding, to ask the question or to check the answers of questions that have already been asked. Now if someone asks a question on StackOverflow, he/she has to wait till the question get answered, and waiting time may vary from question to question depends on the type and difficulty level of the question. So here we are trying predict the time taken to get the answer of the question on the basis of certain feature,syntax and semantics of the question using various NLP,IR and Machine Learning Techniques so that the user get some estimate time to wait. StackOverflow has millions of questions that have been asked so far, so we are using only 10 percent of that data which is having approx 1.24 millions of question data.

3 Literature Survey

Various studies are made on the dataset of stackoverflow and to predict the answering time of the question. The factors which influence the answering of the question are difficulty level of question,tags present in the question.If the correct tag or the known keyword are used then the answering time will decrease. According to the study the classification can be based on either the non-tag(body of question) features or the tagged features. The classification with the help of tag features include the popularity of the tags,frequency of the tags and the co-occurrence of the tags.

4 Dataset

The Data set is consist of 10% of question and answers from Stack Overflow programming QA website.We have taken this data from kaggle. There are three tables in the dataset:

Questions It consist of question id,title, body, creation date, closed date (if answered), score, and owner ID for all non-deleted Stack Overflow questions.The question id is in multile of 10,title consist of question title,body contains the question body,creation date is date when question is created,score is the score of particular question,owner ID is Id of user who created the question.There are approx 1.24 Million enteries (Questions) in this table.

Answers It contains the body, creation date, score, and owner ID for each of the answers to above questions. The ParentId column links back to the Questions table.

Tags It contains the tags used for each question.

5 Simple Baseline Techniques

For baseline, we have implemented Jaccard coefficient similarity, tf-idf based similarity linear regression with multiple variables.

5.1 Preprocessing

In order to implement baseline techniques, first we have pre-processed data using following techniques :

Remove NaN values : Our data might

consist of NAN values, so first we handled Nan values. We have removed those rows for which value corresponding to "OwnerUserId" is Nan.

Deleting Columns : In Questions data, we have deleted "ClosedDate" because it consist many Nan values, and in Answers data, we have removed "Score" and "Body", because we are not using these attributes in our baselines models.

Merging Dataframes and time evaluation : After removing columns, now we wanted to evaluate the time taken by a question to be answered, so for this purpose we are merging our two dataframe : "Question" "Answer". After merging, now we subtract the question creation time and first answer creation time, and calculated the difference in hours. And we have added one more attribute "timeToFirstAnswer" which is having the information of that calculated difference.

Merge Qusetion Title and Question Body : Our question data having "title" which is having title of the question and "Body" which is having text, so we have merged these two columns to find similarity based on both "title" and "Body".

Remove HTML Tags : Our data consist of questions and answers, which might includes text and some HTML tags. These tags is not helpful in order to find similarity so we have removed HTML Tags from Qusetion-Data's Body+Title attribute.

Remove stop words : We are having large amount of data, and these stop words are not much useful in finding the similarity with other questions, so we have removed stop words from question text.

Tokenization : Now question is a string, but we want to find the similarity of questions with other questions which is done by using Bag of word type model, so we have tokenized our question text to create tokens.

Stemming : Stemming is helpful in order to match different words having same root word.

5.2 Jaccard Coefficient based similarity

Here our task is to predict the time of the questions posted on StackOverflow, one basic approach is to find the similar questions that have already been answered and predict the time based on that, and basic technique for this is Jaccard based Similarity .

In Jaccard Similarity method, we consider documents as set of tokens. Now for each test document we converted it to the set and for each train document we calculate jaccard score using this formula :

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|}$$

To predict the time to answer given question we take average of time taken by top five similar questions.The error that we get in our prediction is as follows:

Mean Absolute Error: 593.1482 Hrs
Mean Squared Error: 2623.6718 Hrs

5.3 Feature Engineering

The indicative features are created which are based on the tag and the body of the question which are helpful in predicting the time taken to answer a particular question. The extracted features can be grouped as tag and non-tag based features.

Non-tag based features are as follows:

- **Start word of Question Title :**It store starting word of title, having information weather staring word is interrogative or not may be useful to predict hardness (time taken to solve question).
- **Title end with question mark :**It is true if title of question is end with question mark otherwise it is false.Question mark at end of title may helpful the complication of question.
- **Question Created on Week days :**It store the day of week on which question is asked, question asked on weekend may have different result as compare to question asked in normal day.
- **Question Creation Hours of day :** It store the hour information at which question asked in 24 Hrs format.
- **Body Length :** It store length of body of question.
- **Title Length :** It store length of Title of question.

- **First Tag of question :** A question may have many tags here we consider the first tag of question from tag table for our analysis.
- **Time duration to answer first time :** It store the time difference between the question creation date the time when it is answered first time.

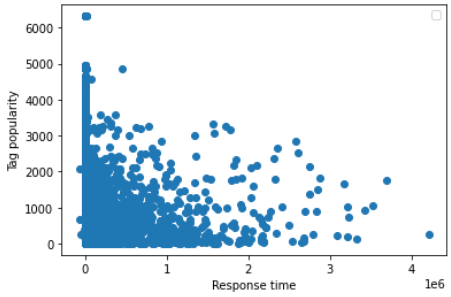
Tag based features are as follows:

- **Tag popularity:** For each question, there are specific tags used to identify the question. Popularity of a tag indicates how frequently the tags is used i.e

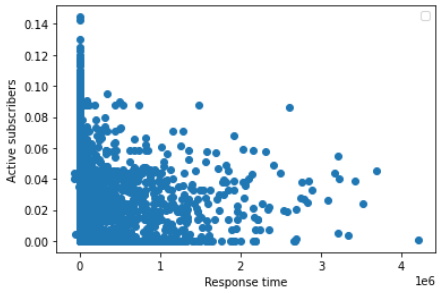
  Tag popularity(tag)=(number of questions in which it appeared)/(Total number of question)

  for each question,average tag popularity is computed.

  The relation between the popularity of tags and response time is shown in below figure . With the increase in the average tag popularity in a question , the response time decreases .

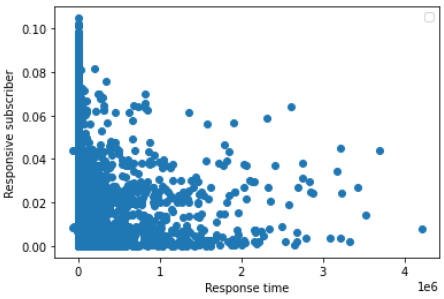


- **Number of active subscriber:** A user is called as subscriber of a tag if he had posted at least one answer within the past history. The past can indicate year,month,day.We considered past as 3 months.And the user is called as active subscriber of a tag if he is a subscriber and posted sufficient answer(10,20,30). After computing number of active subscriber of a particular tag we compute the average number of active subscriber in a question.
- **Percent of active subscriber:** We also compute the ratio of active subscriber to the subscriber for each tag.For each question, we compute the percentage of active subscriber of individual tags. The relation between the percent of active subscribers and response time is shown in below figure . With the increase in the average active subscriber in a question , the response time decreases . the question which has active subscriber in higher bits lie with response time of 0 to (0.2 * e**6) i.e 0 to 80 minutes



- **Number of Responsive subscriber:** A user is called as responsive subscriber of a tag if he answered within the 1 hour of question posted . for a particular question, average number of responsive subscriber are computed.
- **Percent of Responsive subscriber:** similarly for each tag , ratio of number of responsive subscriber to number of subscribers are computed.

  The relation between the percent of Responsive subscribers and response time is shown in below figure . With the increase in the average Responsive subscriber in a question , the response time decreases . the question which has Responsive subscriber in higher bits lie with response time of 0 to (0.2 * e**6) i.e 0 to 80 minutes



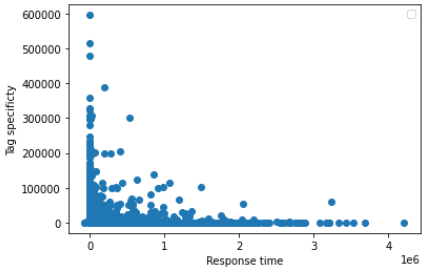
- **Tag Specificity:** It represent the co-occurence of the tags appearing in the question and we calculate the mutual information by:

  MI(tag1,tag2)=P(tag1,tag2)/(P(tag1)*P(tag2))

  where P(tag1,tag2) represent the probability of tags appearing together P(tag1),P(tag2)

represents the probability of tag1,tag2 appearing in the question.

The relation between the tag specificity and response time is shown in below figure . With the increase in the together of tags in a question , the response time decreases . the question which has specificity in higher bits lie with response time of 0 to (0.2 * e**6) i.e 0 to 80 minutes



5.4 Linear Regression with multiple Variable

As the result of jaccard similarity is not much better so try to predict the result using extracted features.

After extracting the features we apply Linear Regression with multiple Variable to predict time taken to answer a given question.First we use only on non-tagged based features for prediction later we include tagged with feature with them and try to show the impact of tagged based features.

First for non-tagged based features we split the data in 7:3 ratio as training and testing data.We use Label Encoder to encode training data before applying Linear Regression.The Mean Absolute Error and Root Mean Squared Error on predicting time taken to answer using non-tagged based features are as follows:-

Mean Absolute Error: 515.27 Hrs

Mean Squared Error: 2293.54 Hrs

Later we include tag-based features along with non-tagged based feature and predict time to answer question after splliting data in ratio 7:3 as training and testing data.The Mean Absolute Error and Root Mean Squared Error on predicting time taken to answer using both tagged and non-tagged based features are as follows :-

Mean Absolute Error: 518.36 Hrs

Mean Squared Error: 2291.41 Hrs

5.5 TFIDF

Here our task is to predict the time of the questions posted on StackOverflow, one basic approach is to find the similar questions that have already been answered and predict the time based on that, and most common technique which is used in order to find the similarity between documents. So we are also using TF-IDF to find the similarity between questions.

We are implementing this model in 100,000 records of merged Dataframe. First we have splitted the data into train and test. Train data is having 80,000 records and test data is having 20,000 records. Now we have implemented TF-IDF on the train data using TFIDF Vectorizer.

Now our task is to find similar questions from train data with particular question from test data, but in order to do it, we are adding tf-idf for each word in test question to all train questions, so it takes lot of time. So in order to save processing during query time, we are using champion list with top 100 questions having high tf-ids. Next we are taking test data row wise, split the question based on space sums the td-idf values for these splitted tokens for each train question find top 5 questions for which tf-idf score for particular test question is high and evaluate the avg time taken of these 5 question which is our predicted time .

At last, we have evaluated the mean error, i.e the mean of difference of actual and predicted time i.e

Mean Absolute Error : 392.99 hrs

6 Classification

As we have done feature engineering to extract features based on tags, and question body related information. Now we will use these features to do the classification.We have created three classes based on the response time of the question, if the response time of a question is less than or equals to 16 minute then this question's class is labelled '0', if response time is greater than 16 minute but less than or equals to 60 minutes then it is labelled as in class '1', and if response time of the question is greater than 60 minutes then it is labelled as in class '2'. Now we had splitted our data into train and test data. So we trained classification models such as knn,gaussian naive bayes and support vector machine based on training data, and predict the classes of test data.

6.1 TFIDF with fast cosine and quality score on Classification Model

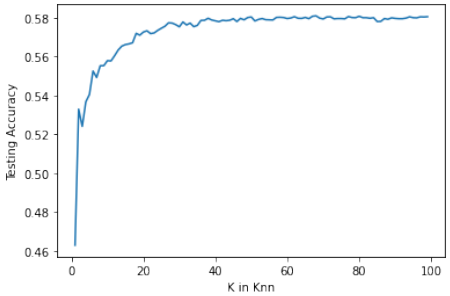
we have splitted question dataset into train and test into 80:20 ratio, now our task is to predict the response time for the test questions. This is be done in the following steps :

- For a particular test question, we have predicted the class label using the classification model.
- Now according to the predicted class, we applied the fast cosine based tfidf model to find the tfidf score of that particular test question with respect to all the questions on train dataset having label as of the corresponding test question.
- Do the same thing with the title of the test question by applying fast cosine with all the question's titles of train dataset having label as of test question to find the tfidf score based on title.
- Now we merged these two tfidf scores i.e body based score and title based score by multiplying 0.6 to title based score and 0.4 to the body based score, and then add them up.
- And after getting the tfidf score for the train questions with respect to test questions, we have added the score of the train questions to get the net score values.
  Net score(q,d) = score(d) + cosine(q, d)
- Now according to net score values, we extracted square root n questions having high net score values ('n' is the number of questions having some score with respect to the test question).
- Next we will get the response time of these square root n questions, and take the average to evaluate the response time of the corresponding test question.
- we have performed the same steps to predict the response time for all the test questions. Then we calculate the error rate, i.e the difference between actual response time and predicted response time. Then evaluate the average error rate for all test questions, which is : **237.85 hrs**

6.2 Classification using tagged and non-tagged based features

In this method we combine tagged and non-tagged based features and predict the class of questions using various classification methods. We first do label encoding of whole data,remove features that are not required then split the data into 7:3 ratio as training and testing data.After this we use following models for prediction and find accuracy in each case:-

- **KNN :** We use sklearn library for KNN and use default parameters as p=2 which is means it use euclidean distance as distance measure.Similarly other default values have different meanings.Here we make accuracy graph for different values of K form 1 to 100 and then use best value of K.Here we make Accuracy graph at different values of k from 1 to 100 and than use best value of k as per our data.



  Graph is almost constant for k more than 85 so we take K=94 is our case and find accuracy as:-
  Accuracy : 0.5804
  F1 Score : 0.5804
- **Gaussian Naive Bayes :** We use sklearn library for Gaussian Naive Bayes with default parameters for class prediction of different questions.For performance measurement, we find accuracy and f1_score which are as follows:-
  Accuracy : 0.5826
  F1 Score : 0.5826
- **Support Vector Machine :** We use sklearn library with default parameters which are as C=1,kernel="rbf",gamma="scale".We pridict the classes of different questions and find accuracy and f1_score for performance measurement which are as follows:-
  Accuracy : 0.5833
  F1 Score : 0.5833

Here we can see that is all classifier we get almost same accuracy and f1_score which is approx .59 .

7 Results

As now days most of the developers uses stack overflow to find the answers of coding related

doubts, but after putting a question, he/she has to wait till someone responds. This waiting time may be of short duration or may be of long duration, so the person will not know that whether he/she has to wait or can do some other approach. So that it will be very helpful if we can predict the response time before actually putting the question. We have applied several Information retreival, and machine learning techniques to find the response time, and here are the results of various methods :

Jaccard Coefficient method :
Mean Absolute Error : 593.14 hrs
Mean squared Error : 2623.67 hrs

Linear Regression on non tagged features only :
Mean Absolute Error : 515.27 hrs
Mean squared Error : 2293.54 hrs

Linear Regression on tagged and non tagged features :
Mean Absolute Error : 518.36 hrs
Mean squared Error : 2291.41 hrs

TF-IDF with champion list :
Mean Absolue Error : 392.99 hrs

TFIDF with quality score and fast cosine method on classification model :
Mean Absolue Error : 237.85 hrs

KNN :
Accuracy : 0.5804
F1 Score : 0.5804

Gaussian Naive Bayes :
Accuracy : 0.5826
F1 Score : 0.5826

Support Vector Machine :
Accuracy : 0.5833
F1 Score : 0.5833

8 Refrences

Bhat, V., Gokhale, A., Jadhav, R., Pudipeddi, J. and Akoglu, L., 2015. Effects of tag usage on question response time. Social Network Analysis and Mining, 5(1).