Article:
Roy, Pradeep Kumar, Singh, Jyoti Prakash and Banerjee, Snehasish orcid.org/0000-0001-6355-0470 (2024) Is this question going to be closed?:Answering question closibility on Stack Exchange. Journal of Information Science. pp. 1291-1307. ISSN: 0165-5515

https://doi.org/10.1177/01655515221118665

Is this question going to be closed? Answering question closibility on Stack Exchange

Journal:	*Journal of Information Science*
Manuscript ID	JIS-21-0009.R4
Manuscript Type:	Research Paper
Keywords:	Closed Question, Community Question Answering, Question Quality, Machine Learning, Stack Exchange, Unanswered Question
Abstract:	Community question answering sites (CQAs) are often flooded with questions that are never answered. To cope with the problem, experienced users of Stack Exchange are now allowed to mark newly-posted questions as closed if they are of poor quality. Once closed, a question is no longer eligible to receive answers. However, identifying and closing sub-par questions takes time. Therefore, the purpose of this paper is to develop a supervised machine learning system that predicts question closibility, the possibility of a newly posted question to be eventually closed. Building on extant research on CQA question quality, the supervised machine learning system uses 17 features which were grouped into four categories, namely, asker features, community features, question content features, and textual features. The performance of the developed system was tested on questions posted on Stack Exchange from 11 randomly chosen topics. The classification performance was generally promising, and outperformed the baseline. Most of the measures of precision, recall, F1-score and AUC were above 0.90 irrespective of the topic of questions. By conceptualizing question closibility, the paper extends previous CQA research on question quality. While previous works were mostly restricted to programming-related questions drawn from Stack Overflow, it empirically tests question closibility on questions from 11 randomly chosen topics. The set of features used for classification offers a framework of question closibility that is not only more comprehensive but also more parsimonious compared with prior works.

Is this question going to be closed? Answering question closibility on Stack Exchange

Journal Title
XX(X):1–12

DOI: 10.1177/ToBeAssigned
www.sagepub.com/
SAGE

Abstract

Community question answering sites (CQAs) are often flooded with questions that are never answered. To cope with the problem, experienced users of Stack Exchange are now allowed to mark newly-posted questions as closed if they are of poor quality. Once closed, a question is no longer eligible to receive answers. However, identifying and closing subpar questions takes time. Therefore, the purpose of this paper is to develop a supervised machine learning system that predicts question closibility, the possibility of a newly posted question to be eventually closed. Building on extant research on CQA question quality, the supervised machine learning system uses 17 features that were grouped into four categories, namely, asker features, community features, question content features, and textual features. The performance of the developed system was tested on questions posted on Stack Exchange from 11 randomly chosen topics. The classification performance was generally promising and outperformed the baseline. Most of the measures of precision, recall, F1-score, and AUC were above 0.90 irrespective of the topic of questions. By conceptualizing question closibility, the paper extends previous CQA research on question quality. Unlike previous studies, which were mostly limited to programming-related questions from Stack Overflow, this one empirically tests question closibility on questions from 11 randomly selected topics. The set of features used for classification offers a framework of question closibility that is not only more comprehensive but also more parsimonious compared with prior works.

Keywords

Closed Question, Community Question Answering, Machine Learning, Question Quality, Stack Exchange, Unanswered Question

Introduction

Over the years, community question answering sites (CQAs) such as Baidu Zhidao, Stack Exchange, and Yahoo! Answers have cemented themselves as key avenues to search for information. Whenever individuals with Internet access face an information need, they have an easy option to ask questions on CQAs that can be answered by other online users. If the asker chooses an incoming answer as satisfactory, the question is said to be resolved (1; 2; 3; 4; 5; 6).

Despite the undoubted benefit of CQAs, a downside is that these sites have long been flooded with questions that are never answered. For example, by 2010, 42.8% of questions posted on Baidu Zhidao were reported to remain unanswered (7). By 2012, the volume of unanswered questions on Stack Overflow, a CQA site within Stack Exchange that is dedicated to computer programming, mounted to approximately 300,000 (8). About one-fifth of unresolved questions on Yahoo! Answers are known to remain completely ignored (9).

To cope with the problem of unanswered questions, experienced users of Stack Exchange are now allowed to mark newly-posted questions as closed. Specifically, a question can be closed if it is deemed to be duplicate, off-topic, opinion-seeking, unclear or vague. A question that is closed is not possible to be answered, but can be updated for reopening (Stack Exchange, 2018). Clearly, this functionality helps nurture the quality of questions on the platform.

Even with this development, under-cooked questions continue to serve as a thorn in the flesh of Stack Exchange's question-answering cycle. Identifying and closing them manually takes time, which experienced users would have rather spent on more meaningful questioning and answering activities. Furthermore, the queue of inappropriate questions on CQAs seems to be continually growing (7; 9; 10; 11; 2; 12). A review of questions on various topics such as *Golf* and *Mathematics* available on Stack Exchange confirmed that closed questions are indeed ubiquitous (see Table 1).

A potential remedy is to automatically identify questions that are likely to be closed before they are actually closed in reality. If the possibility for questions to be closed (henceforth referred as closibility) can be conveyed to askers automatically soon after they write their questions without human intervention, the volume of likely-to-be-closed questions on Stack Exchange will be reduced. This in turn will minimize the time that experienced users, who are valuable information sources in the CQA community, would spend in closing subpar questions. Instead, they could focus on asking and answering, thereby resulting in a more efficient use of the CQA platform for all and sundry.

Therefore, the purpose of this paper is to develop a supervised machine learning system that predicts the closibility of questions on Stack Exchange. Existing research on question closibility has used a large number of features (13) but achieved a 0.71 F1-score (14). Roy et al. (15) used deep learning frameworks and achieved very low prediction accuracy. Deep learning-based models

Prepared using sagej.cls *[Version: 2017/01/17 v1.20]*

automatically extracted features from the input, and hence, interpreting the reasons for question closibility was not feasible. The model proposed in this research uses a fewer number of manually extracted features and outperforms the baseline models. Moreover, the proposed model also helps to identify the reason for question closibility through a feature analysis.

The paper particularly builds on prior research that has been shedding light on reasons due to which several questions remain unanswered on CQAs (16; 9). In doing so, the paper is significant in two ways. First, it represents one of the earliest works to conceptualize what is referred as the closibility of questions. Using the body of CQA research on questions' likelihood to remain unanswered as a stepping stone, the paper deepens the scholarly understanding of factors that predict questions' likelihood to be closed. Second, the developed system to predict closibility is meant to be generic so that it can be applied to questions on a variety of topics. Its performance was evaluated by testing it on data from a range of 11 topics available on Stack Exchange. This extends previous research that has been mostly restricted to programming-related questions drawn from Stack Overflow (9; 14). The classification performance was generally promising.

The rest of the paper is organized as follows: The next section reviews the literature. The methodology is described thereafter. This is followed by the results and the discussion. The final section closes out the paper by highlighting its limitations and future scope.

Literature Review

Since their inception, CQAs have been garnering and archiving huge volumes of user-generated content from their community of users. Their proliferation has continued to attract sustained scholarly interest over the years (1; 17; 12; 18). In particular, CQAs have been widely investigated to resolve answer-related issues such as examining the quality of answers (9; 19; 20; 21; 22; 23), and finding the best answer among a pool of responses (24; 25). Much attention has also been devoted to identifying expert users who are capable of producing high quality answers (26; 27; 28; 29; 30; 31; 32; 4; 33; 34).

More recently, research has started to cast the spotlight on question-related issues such as clustering similar questions coupled with identification of hot topics (35; 36; 37), identification of questioning motivation (38), and detecting duplicate questions (39; 40; 41; 42; 43; 44; 45). Particularly relevant to this paper, scholars have also started to examine the quality of questions posted on CQAs (9; 16; 46; 12; 47).

For example, Shah et al. (48) used a dataset of 5,000 questions posted on Yahoo! Answers to classify question quality as either good or bad. With the help of support vector machine (SVM) classification, they achieved an accuracy of 93.08%. The excellent performance notwithstanding, a limitation of the work was that some of the features required human intervention. Hence, it does not offer a logistically viable strategy to predict question quality on the fly. Ponzanelli et al. (41) proposed a model to minimize low-quality content on CQAs using textual and non-textual features extracted from a Stack Overflow dataset. They achieved a precision of 41.9% for high quality questions and 64.91% for low quality questions. Correa and Sureka (49) found that around 8% of the total questions were subpar on Stack Overflow. They used 47 different textual and non-textual features for classification. The model achieved a modest accuracy of 66%.

Srba et al. (50) claimed that the quality of questions on Stack Overflow had been deteriorating over the years. They reported that the low-quality content of the site was 4.11% in 2011, which increased to 16.84% in 2016. They further showed that a particular group of users was continuously posting duplicate questions or definition-type questions on the site. Ahasanuzzaman et al. (39) proposed a system to find duplicate questions on Stack Overflow. They used cosine similarity, WordNet Similarity, Entity Overlap, and Entity type overlap on a dataset of some 1.3 million Stack Overflow questions to classify whether an enquiry was duplicate or not. Their system achieved a recall value of 66.10%. Yang et al. (46) analyzed unanswered questions on Yahoo! Answers, and proposed a supervised machine learning model to classify them. With a dataset containing 76,251 questions out of which 10,424 were unanswered, their model achieved a F1-score of 32.5%. Dror et al. (51) extended the work of (46) to predict the number of answers a question might receive. They achieved a F1-score of 40.3%.

Most closely related to the current paper is the work of (14). It proposed a model to predict closed questions on Stack Overflow. Several textual and non-textual features were extracted from the dataset to achieve a classification accuracy of 73%. This paper extends (14) in at least two ways. First, instead of confining the dataset to only programming-related questions, it draws data from 11 randomly chosen topics. This serves to enhance research generalizability. Second, informed by recent works such as (9), this paper incorporates several new features such as up votes, down votes, and interrogative words. Moreover, it drops variables such as number of short words, upper case and lower case characters. For one, the conceptualization of short words was unclear from (14). Besides, there is no reason to assume that the proportions of upper and lower case characters will determine question closibility. Therefore, this paper makes a modest attempt to present a more parsimonious set of features for predicting question closibility compared with previous works.

Methodology

Fig. 1 depicts the complete framework of the proposed model to predict question closibility. The steps are explained below:

Dataset: The dataset used in this paper was the data dump of Stack Exchange from January 2009 to March 2017*. The data dump is an anonymized dump of all posts, tags, votes, users, history, comments, badges and post history of the CQA in the form of eight different XML files. For the purpose of this paper, four XML files related to questions and users were relevant: User.xml, Post.xml, Votes.xml, and Badge.xml. These are represented within the dotted lines in Fig. 1. The dataset was downloaded from the archive in May 2017.

*https://archive.org/details/stackexchange/

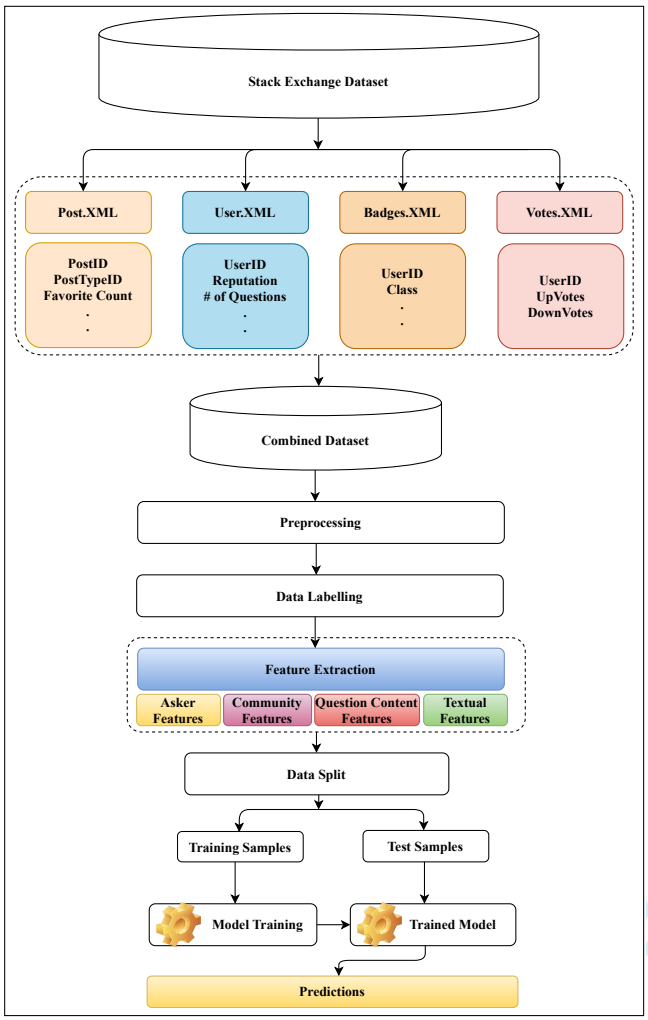

Figure 1. Proposed model to predict question closibility.

Combined dataset and preprocessing: From the XML files, all the relevant attributes were captured to form the combined dataset. These include post id, post type id, favorite count, view count, comment count, and answer count from Post.xml; user id, user reputation, and number of posts by the user from User.xml; user id and the class of the user from Badge.xml; and user id, up votes and down votes from Votes.xml. All of these attributes from the different XML files were combined together for further processing. The dataset was selected across 11 randomly selected topics. A statistics of the selected dataset is presented in Table 1, in which the topics are arranged in decreasing order of the proportion of closed questions.

Data labelling: The dataset was labelled into two classes (Open and Closed) based on the attribute of closed date. Questions having some date values in the closed date field indicated that they had been closed. Hence, they were labeled as closed questions. All remaining questions were labeled as open questions.

Feature extraction: A total of 17 features were extracted from the combined dataset. Informed by (14), these were grouped them into four categories, namely, asker features, community features, question content features, and textual features.

Asker features include (1) account age, and (2) badge score. These were relevant because questions contributed by long-standing askers with high badge scores are likely to have less closibility than those posted by new and novice individuals (9; 52).

Community features include (3) post score, (4) reputation, (5) favorite count, (6) comment count, (7) view count, (8) answer count, (9) up votes, and (10) down votes. All of these reflect how well the CQA community accepts a user (53; 16). Obviously, the more whole-heartedly a user is accepted in the community, the less is the likelihood for the community to close a question submitted by the individual.

Question content features include (11) number of URLs, and (12) number of tags. These have been shown to play a part in determining question closibility (14).

Textual features include (13) question body length, (14) question title length, (15) number of special characters, (16) number of punctuation marks, and (17) number of interrogative words. These textual features are known to determine the clarity with which questions are articulated (16; 9; 14). Hence, they may also shape question closibility.

All the 17 features are described below, and listed in Table 2.

1. **Account age:** The age of users account is the time from the date of joining the site to their latest post.
2. **Badge score:** The number of badges a user (u) has earned. Say, a user earns badges ($b1$,$b2$,.....bn) then, Badge Score (Bs) of user u is calculated as:

$$Bs = \sum_{i=1}^{n} \frac{1}{No.\ of\ users\ having\ b_i} \quad (1)$$

3. **Post score** The score gained by the user (u) from the community users. The score is calculated as follows: Let the answers posted by the user u are $(ans_1, ans_2, ...ans_n)$ and question asked by user (u) are $(ques_1, ques_2, ...ques_m)$ then the score S(u) of the user u is:

$$S(u) = \sum_{i=1}^{n} S(ans_i) + \sum_{j=1}^{m} S(ques_j) \quad (2)$$

4. **Reputation:** Users receive reward points for good quality postings. Reward score reflects the standing of users in the CQA community
5. **Favourite count:** Every question or answer of user (u) can be marked as a favorite by a user (v). Suppose the number of questions of user (u) marked as favorite is n and the number of answers marked as favorite is m then favorite count of user (u) is:

$$Favourite\ Count(u) = n + m \quad (3)$$

6. **Comment count:** The number of comments received from peer users. Suppose the number of comments received on a user question is (C_qn) and Comments on their answers is C_an then the comment count of user (u) is:

$$Comment\ Count(u) = C_{qn} + C_{an} \quad (4)$$

7. **View count:** The number of community users who have seen the posts of user u.

Table 1. Statistics of closed questions for the selected topics of Stack Exchange.

Topics	Total Questions	Open Questions	Closed Questions	% of Closed Questions
Programmer	38,299	28,480	9,819	25.64%
Ask Ubuntu	251,769	218,136	33,633	13.36%
Golf	7,195	6,236	959	13.33%
Sci-fi	38,027	34,518	3,509	9.23%
Server Fault	238,765	219,584	19,181	8.03%
Dba	53,665	49,929	3,736	6.96%
Gaming	75,697	70,443	5,254	6.94%
Super User	343,034	320,168	22,866	6.67%
Apple	80,467	76,999	3,468	4.31%
Mathematics	222,288	215,372	6,916	3.11%
Code Review	42,451	41,484	967	2.28%

Table 2. List of selected features

Type	Features
Asker features	Account age
	Badge score
Community features	Post score
	Reputation
	Favorite count
	Comment count
	View count
	Answer count
	Up votes
	Down votes
Question content features	Number of URLs
	Number of tags
Textual features	Question body length
	Question title length
	Number of special characters
	Number of punctuation marks
	Number of interrogative words

8. **Answer count:** The number of answers received by the questions asked by a user u.
9. **Up votes:** The number of posts with positive score.
10. **Down votes:** The number of posts with negative score.
11. **Number of URLs:** The number of URLs present in the question body.
12. **Number of tags:** Tags are generally keywords given by users to represent the question domain precisely. A high number of tags could suggest greater question precision.
13. **Question body length:** The number of words in the body of the question after removing stop words.
14. **Question title length:** The number of words in the question title.
15. **Number of special characters:** The number of a special character such as @, >, <, etc., present in the question body.
16. **Number of punctuation marks:** The number of punctuation marks present in the question body.
17. **Number of interrogative words:** The number of interrogative words (start with 'wh', for example: 'what', 'when', 'where' etc.,) present in the question body.

Data Split: The complete dataset as a whole could not be used for training and testing purposes. Hence, it was split into training and testing components.

Training Samples: We randomly used 67% of the data points from the dataset to train the model.

Test Samples: To test the performance of the trained model, we used the remaining 33% unseen dataset, i.e., the data points which were not passed to the model during the training.

Model Training: We trained several machine learning models like Gradient Boosting, Logistic Regression, Naive Bayes, Random Forest, XGBoost and others with the extracted features.

Trained Model: When the trained models were ready for prediction with new data points, they were supplied with the test samples for predictions.

Predictions: The trained machine learning models predicted the category of the unseen data points. When the predicted category of an unseen data point matched with the actual category, it was deemed to be a correct prediction.

Model Setting and Evaluation Metrics

To evaluate the performance of the supervised machine learning system with the selected features, several classifiers were experimented (54; 55; 56). For the sake of brevity, the paper reports the results for Gradient Boosting, Logistic Regression, Naive Bayes, and Random Forest. XGBoost, for example, was omitted because it yielded very similar results to Gradient Boosting.

In the dataset, the proportions of closed and open questions were never comparable. Table 1 conveys that closed questions were consistently fewer than open questions for all the 11 topics. For example, out of 38,299 questions from the topic of *Programmer*, only 9,819 (25.64%) questions were closed.

Therefore, to study the effect of data imbalance, the classification was conducted in two phases: first with the imbalanced dataset, and next with its balanced version created by applying what is known as SMOTE–synthetic minority over-sampling technique (57). In SMOTE, the minority class data is oversampled by creating synthetic examples instead of over-sampling with replacement. This is known to result in realistic newly-created samples, and is superior to other techniques such as random oversampling, which increases the sample of minority classes by creating

multiple copies of the same data points. With such repeated instances in the dataset, classifiers tend to be over-fitted during the training process. Another oversampling technique known as ADASYN (ADAptive SYNthetic method) builds on SMOTE. Nonetheless, initial experiments on two datasets *Programmer* and Ask Ubuntu yielded similar results with both the SMOTE and ADASYN techniques on RF and GB classifiers, however, NB and LR classifiers performed better with SMOTE compared with ADASYN. Hence, we proceeded with SMOTE to balance the datasets. These were executed in Python on a machine having Intel Xeon(R) CPU 16 cores and 32GB RAM.

The classification performance was evaluated in terms of precision, recall, F1-score, and area under the receiver operating characteristic curve (58). These are explained as follows:

- *Precision*: It is defined as the fraction of closed questions among the retrieved closed questions. It is computed as:

$$Precision = \frac{T_p}{T_p + F_p} \tag{5}$$

  where T_p is true positive (*closed* questions classified as *closed*), F_p is false positive (*open* questions classified as *closed*).
- *Recall*: It is the fraction of *closed* questions that have been retrieved over the total amount of *closed* questions in the system , recall is computed as:

$$Recall = \frac{T_p}{T_p + F_n} \tag{6}$$

  Where T_p is true positive (*closed* question classified as *closed*), F_n is false negative (*closed* question classified as *open*).
- *F1-score*: is the harmonic mean of *Precision* and *Recall*, *F1-score* is computed as:

$$F1 - score = 2 * \frac{precision * recall}{precision + recall} \tag{7}$$

- *AUC:* Receiver operating characteristic curve (ROC) is the plot between the true positive rate and the false positive rate of the classifier for different thresholds

$$True\ Positive\ Rate = \frac{T_p}{T_p + F_n} \tag{8}$$

$$False\ Positive\ Rate = \frac{F_p}{F_p + T_n} \tag{9}$$

  Where F_P is the number of *open* questions that are classified as *closed*. The greater the area under the receiver operating characteristic curve (AUC) value, the greater is the accuracy of classifiers.

Results

This section presents the experimental results obtained using the various machine learning algorithms on both the imbalanced and the balanced datasets. The datasets were divided into two parts. One part contained 67% of the data, and was used for model training. The other part contained the remaining 33% of the data, and was used to test the model performance.

Results with Imbalanced Datasets

The experiment commenced with questions from the topics of *Programmer* and *Ask Ubuntu*. This was because they contained the highest proportions of closed questions. The results are presented in Table 3.

Two interesting observations were made. First, Naive Bayes yielded antagonistic results for questions from the two topics. For the topic of *Programmer*, recall was higher for open questions (0.93) vis-a-vis closed ones (0.16). In other words, a better recall was achieved for the class having greater number of data instances. For the topic of *Ask Ubuntu* however, a better recall was achieved for closed questions (0.91) vis-a-vis open ones (0.16). Put differently, a better recall was obtained for the class having fewer data instances.

Second, none of the classifiers yielded very promising results consistently in terms of precision, recall, F1-score, and AUC across questions from both the topics. The imbalance in the dataset was identified as a possible reason. Therefore, the data were balanced to recheck the classification performance. As indicated earlier, SMOTE was applied for this purpose (57).

Results with Balanced Datasets

The performance of the proposed model was now checked using the balanced datasets from the topics of *Programmer* and *Ask Ubuntu*. The detailed results are presented in Table 4. There were two notable observations. First, even with balanced datasets, Naive Bayes continued to yield antagonistic results for questions from the two topics. For the topic of *Programmer*, recall was higher for open questions (0.88) vis-a-vis closed ones (0.26). For the topic of *Ask Ubuntu* however, a better recall was achieved for closed questions (0.94) vis-a-vis open ones (0.12). Given the poor performance, the data distribution was checked. The features were not normally distributed. As mentioned in literature (59), Naive Bayes classifier performs well with datasets that are normally distributed or categorical in nature. This was the reason for the anomaly.

Second, Gradient Boosting outperformed the other classifiers in predicting question closibility. This was true across both the topics in terms of all the four selected performance indicators: precision, recall, F1-score, and AUC (58). The confusion matrix detecting the true distribution of the data instances over the different classes and the ROC curves for the topics of *Programmer* (AUC = 0.82) and*Ask Ubuntu* (AUC = 0.94) are shown in Figures 2, 3, 4, and 5.

Given that Gradient Boosting emerged as the best performing classifier, it was therefore employed on the balanced datasets for all the topics to predict question closibility. The results are summarized in Table 5. The classification performance was generally promising.

As indicated earlier, the most pertinent prior work for this paper is (14). It proposed a model to predict closed questions from a dataset of programming-related questions drawn from Stack Overflow. Therefore, using questions from the topic of *Programmer*, this paper uses (14) as a baseline for comparison. The comparative outcomes are shown in Table 7. The classification performance using Gradient Boosting was traced in terms of precision, recall, F1-score, and AUC. To afford a granular analysis, the classification

Table 3. Results with the imbalanced dataset for the topics of *Programmer* and *Ask Ubuntu*

		Programmer				Ask Ubuntu			
Classifier	Class	Precision	Recall	F1-Score	AUC	Precision	Recall	F1-Score	AUC
Gradient Boosting	Open	0.76	0.96	0.85	0.55	0.91	0.99	0.95	0.71
	Closed	0.59	0.14	0.23		0.92	0.42	0.57	
	Average	0.68	0.55	0.54		0.92	0.71	0.71	
Naive Bayes	Open	0.76	0.93	0.83	0.54	0.91	0.16	0.27	0.54
	Closed	0.44	0.16	0.23		0.15	0.91	0.26	
	Average	0.60	0.55	0.53		0.53	0.54	0.27	
Logistic Regression	Open	0.74	0.99	0.85	0.51	0.86	1.00	0.92	0.50
	Closed	0.46	0.03	0.05		0.14	0.01	0.01	
	Average	0.60	0.51	0.45		0.50	0.51	0.46	
Random Forest	Open	0.76	0.94	0.84	0.56	0.91	0.99	0.95	0.70
	Closed	0.49	0.17	0.26		0.90	0.41	0.57	
	Average	0.63	0.56	0.55		0.91	0.70	0.76	

Table 4. Results with the balanced dataset for the topics of *Programmer* and *Ask Ubuntu*

		Programmer				Ask Ubuntu			
Classifier	Class	Precision	Recall	F1-Score	AUC	Precision	Recall	F1-Score	AUC
Gradient Boosting	Open	0.76	0.93	0.84	0.82	0.90	0.99	0.95	0.94
	Closed	0.92	0.71	0.80		0.99	0.89	0.94	
	Average	0.84	0.82	0.82		0.95	0.94	0.95	
Naive Bayes	Open	0.54	0.88	0.67	0.57	0.65	0.12	0.20	0.53
	Closed	0.68	0.26	0.38		0.52	0.94	0.66	
	Average	0.61	0.57	0.53		0.59	0.53	0.43	
Logistic Regression	Open	0.56	0.51	0.54	0.56	0.66	0.40	0.50	0.60
	Closed	0.55	0.60	0.58		0.57	0.80	0.66	
	Average	0.56	0.56	0.56		0.62	0.60	0.58	
Random Forest	Open	0.74	0.9	0.81	0.74	0.89	0.99	0.94	0.93
	Closed	0.88	0.68	0.76		0.99	0.88	0.93	
	Average	0.81	0.79	0.79		0.94	0.94	0.94	

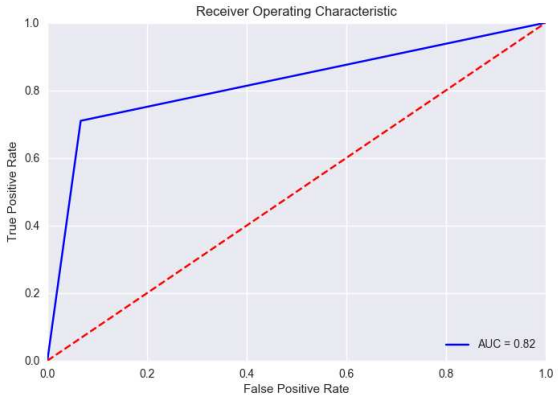

Figure 2. ROC curve for Programmer topic

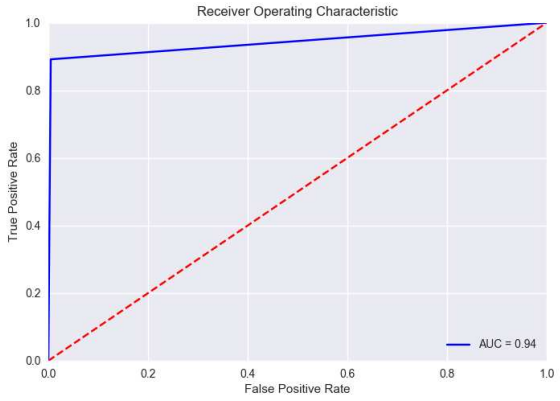

Figure 3. ROC curve for Ask Ubuntu topic

was performed in four steps that involved including asker features, community features, question content features, and textual features one by one. The proposed model mostly outperformed the baseline with the selected feature set as shown in Table 6. The performance of the proposed work was also compared with Roy et al. (15).

Results with Deep Learning

Of late, deep learning-based models such as Convolutional Neural Network (CNN), Long-Short Term Memory (LSTM), and the transformer-based BERT models are widely being used in natural language processing. These models are often preferred to traditional machine learning as they are capable of finding hidden contextual patterns from the dataset using their complex architecture. Hence, we decided to explore deep learning.

Table 5. Results with datasets after balancing with Gradient Boosting.

Topics	Class	Precision	Recall	F1-Score	AUC
Programmer	Open	0.76	0.93	0.84	0.82
	Closed	0.92	0.71	0.80	
Ask Ubuntu	Open	0.90	0.99	0.95	0.94
	Closed	0.99	0.89	0.94	
Golf	Open	0.91	0.95	0.93	0.93
	Closed	0.95	0.90	0.92	
Sci-fi	Open	0.92	0.97	0.94	0.95
	Closed	0.97	0.91	0.94	
Server Fault	Open	0.92	0.99	0.96	0.96
	Closed	0.99	0.92	0.95	
Dba	Open	0.93	0.99	0.96	0.96
	Closed	0.99	0.92	0.96	
Gaming	Open	0.93	0.98	0.96	0.96
	Closed	0.98	0.93	0.96	
Super User	Open	0.93	0.99	0.96	0.96
	Closed	0.99	0.92	0.96	
Apple	Open	0.94	1.00	0.96	0.96
	Closed	1.00	0.93	0.96	
Mathematics	Open	0.95	0.99	0.97	0.97
	Closed	0.99	0.95	0.97	
Code Review	Open	0.97	0.99	0.98	0.98
	Closed	0.99	0.97	0.98	

Table 6. Comparison with baseline.

Asker Features	Community Features	Question Content Features	Textual Features	Approach	Precision	Recall	F1-Score	AUC
✓				(14)	0.63	0.63	0.63	0.62
✓				Proposed	0.59	0.55	0.57	0.60
✓	✓			(14)	0.70	0.65	0.67	0.68
✓	✓			Proposed	0.91	0.70	0.79	0.82
✓	✓	✓		(14)	0.70	0.65	0.67	0.68
✓	✓	✓		Proposed	0.91	0.71	0.80	0.82
✓	✓	✓	✓	(14)	0.69	0.65	0.67	0.68
✓	✓	✓	✓	**Proposed**	**0.92**	**0.71**	**0.80**	**0.82**

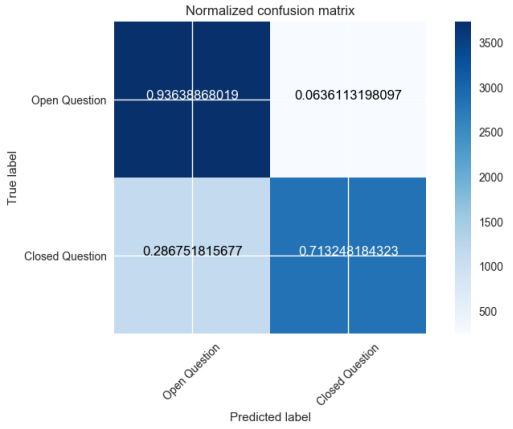

Figure 4. Confusion matrix for Programmer topic

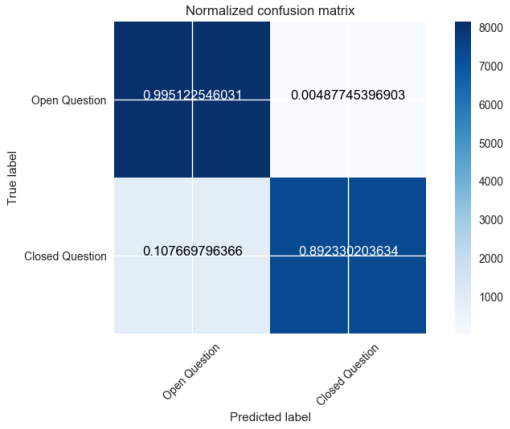

Figure 5. Confusion matrix for Ask Ubuntu topic

The experiments were conducted on *Programmer* topic using the Google Collabotary platform, where all required libraries are pre-installed. Firstly, we experimented with the CNN model with two variations (i) without using

Table 7. Comparison with Deep Learning.

Model	Precision	Recall	F1-Score	AUC
Roy et al. (15)	0.47	0.49	0.48	0.81
Proposed	**0.92**	**0.71**	**0.80**	**0.82**

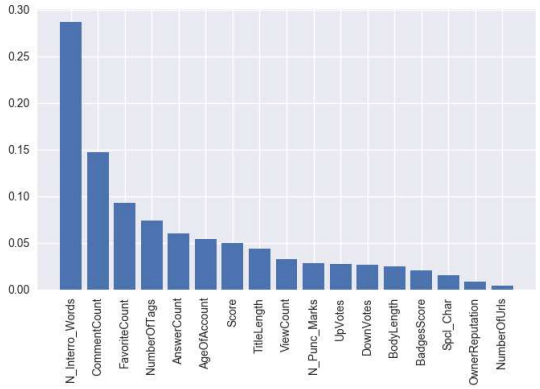

Figure 6. Feature importance graph for the topic of *Programmer*.

the Dropout layer and (ii) Using the Dropout layer. The outcomes are presented in Table 8. The CNN model with a single layer of convolution worked well for the open category question prediction but performed poorly for the closed category. Even adding a dropout layer did not substantially improve model performance. The F1-score was only 0.33 for closed questions. This shows that most closed questions remained untraceable by the CNN model.

Next, we increased the convolution layer from one to two and repeated the experiments. The outcomes of the 2-layered CNN model improved slightly but were still lower than what was achieved using traditional machine learning. The LSTM and BERT model also exhibited similar performances. The AUC value using the LSTM model was 0.77, and using the BERT model was 0.61, indicating that the LSTM model performed better among all the deep learning and transformer-based models.

Feature Importance

To dig deeper, the relative strength of the features was assessed for the topics of *Programmer* and *Ask Ubuntu*. Using Gradient Boosting, the feature importance graphs are shown in Fig. 6 and Fig. 7 respectively.

For the topic of *Programmer*, the top three contributing features were number of interrogative words, comment count, and favorite count. For the topic of *Ask Ubuntu*, the top three features were answer count, number of tags, and number of interrogative words. Across the two topics, the top three features were mostly dominated by non-textual features. The only exception was number of interrogative words.

The effect of the selected features can also be seen from Table 6, in which the model performance is shown as a function of the feature subsets. Each subset of features, namely, asker features, community features, question content features, and textual features, was useful as the values of the performance measures continued to rise progressively.

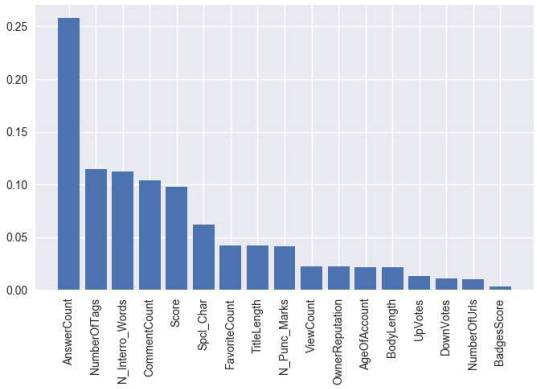

Figure 7. Feature importance graph for the topic of *Ask Ubuntu*.

Discussion

This section deals with the experimental outcomes and the major finding of the research, including theoretical contributions and implications for practice (60). CQAs are known to be flooded with questions that are never answered (7; 9; 10). To cope with the problem, experienced users of Stack Exchange are now allowed to mark newly-posted questions as closed if it is duplicate, off-topic, opinion-seeking, unclear or vague. However, identifying and closing subpar questions manually takes time.

To this end, this paper developed a supervised machine learning system that predicts question closibility–the possibility of a newly posted question to be eventually closed. It leveraged the body of literature that has been shedding light on reasons due to which several questions remain unanswered on CQAs (16; 9). The system was tested on questions posted on Stack Exchange from 11 randomly chosen topics. Gradient Boosting emerged as the best performing classifier. As shown in Table 5, the classification performance was generally promising. Most of the measures of precision, recall, F1-score and AUC were above 0.9 with minimum values of 0.76, 0.71, 0.80 and 0.82 respectively.

The system was found to outperform the baseline-an earlier model to predict closed questions proposed by (14). As shown in Table 6, the best F1-score obtained by the baseline (14) was 0.67 whereas our model achieved 0.80 in a similar setting. The paper also supplements previous related works. (46), for example, only analyzed unanswered questions, and achieved a F1-score of 32.5%. Again, (39) achieved a recall of 66.10% in identifying duplicate questions. These factors such as being unanswered and duplication can all lead to a question being closed. This research combines all such factors of question closibility, and achieves an average F1-score of 0.92 with a minimum of 0.82 for the topic of *Programmer*, and a maximum 0.97 for the topic of Mathematics. Moreover, unlike works such as (48) that used features requiring human annotation, the system developed in this paper employs features that could be readily obtained. Therefore, it offers a viable strategy to predict question closibility on CQAs in real time.

Table 8. Results of Gradient Boosting (traditional machine learning) in comparison with deep learning

Topics	Class	Precision	Recall	F1-Score	AUC
1CNN	Open	0.79	0.89	0.84	0.59
	Closed	0.47	0.28	0.35	
1CNN+Dropout	Open	0.78	0.83	0.80	0.56
	Closed	0.37	0.30	0.33	
2CNN	Open	0.77	0.97	0.86	0.55
	Closed	0.61	0.13	0.22	
2CNN+Dropout	Open	0.80	0.89	0.84	0.55
	Closed	0.50	0.33	0.40	
LSTM	Open	0.78	0.97	0.86	0.77
	Closed	0.69	0.17	0.27	
BERT	Open	0.79	0.95	0.86	0.61
	Closed	0.68	0.28	0.40	
Gradient Boosting	Open	0.76	0.93	0.84	0.82
	Closed	0.92	0.71	0.80	

Theoretical Contributions

The theoretical contributions of this paper are four-fold. First, it conceptualizes what is referred as the closibility of questions. It also develops a supervised machine learning system that predicts the possibility of questions on Stack Exchange to be closed. This serves to deepen the scholarly understanding of factors that predict questions' likelihood to be closed on CQAs. It serves as a call for scholars to explore the theme of question closibility more granularly in the future.

Second, the paper builds on the literature on question quality (9; 16) to identify new features for predicting question closibility. Features such as the use of interrogative words were not used in earlier works to distinguish between closed and open questions (14). However, number of interrogative words was among the top three features for the topics of both *Programmer* and *Ask Ubuntu* (Fig. 6 and Fig. 7). It should be incorporated in related future research. In sum, the 17 features identified in this paper seem more comprehensive than prior works.

Third, the supervised machine learning system developed to predict question closibility was intended to cater to questions on a variety of topics. For this reason, features such as code snippet–commonly used in prior works that focus solely on programming-related questions—-were dropped. This serves to enhance the parsimony while widening the applicability of the classifier. The classification performance was evaluated by testing the system on questions from a range of 11 topics available on Stack Exchange. As evident from Table 5, majority of the measures of precision, recall, F1-score and AUC were above 0.90. This extends previous research that was mostly restricted to programming-related questions drawn from Stack Overflow (9; 14).

Fourth, the paper has implications for the usage of supervised machine learning on CQAs. For one, it demonstrates the utility of SMOTE to predict question closibility (57). Previous CQA studies focusing on question quality did not mitigate the data imbalance problem before feeding datasets to classifiers. This paper confirms that classification performance would be inadequate if the dataset is overly imbalanced. It also echoes previous research (59) that Naive Bayes classifier does not work well if the features are not normally distributed. Gradient Boosting emerged as the best classifier regardless of questions' topic.

Implications for practice

On the practical front, this paper demonstrates the possibility of screening new questions while they are being submitted on CQAs (61). Moderators and administrators of CQAs could use the system developed for the purpose of this paper as a preliminary filter to weed out questions with high closibility. The current system may be placed just beneath the user interface to work silently and screen all incoming questions. Besides, this automated system could be employed on the archives of previously-posted questions on CQAs to evaluate question quality at regular intervals.

Overall, the system is able to free users from the tedious task of manually closing subpar questions on CQAs. It will reduce not only the community participation time in marking questions as closed but also the moderation job time. These in turn will pave the way for a more efficient use of the CQA platform as an information-seeking avenue.

Conclusion, Limitations and Future scope

On CQAs, the number of closed questions has been rising. This paper presents a machine learning method to predict question closibility, which is defined as the possibility of a question to be closed, as a first step toward solving the problem. A supervised machine learning system was developed and tested on data from Stack Exchange on as many as 11 topics. The feature set was more parsimonious compared with those used in prior works. Even then, the system fared better than earlier studies. Gradient Boosting emerged as the best performing classifier regardless of the question topics. Therefore, the key takeaway message from the paper is this: Using supervised learning, it is possible to automatically identify questions that are likely to be closed before they are actually closed, with reasonably high accuracy. This holds immense promise for the future of CQAs.

Nonetheless, a limitation of the proposed system is that the features were selected manually. It was difficult to find the optimal number of features when the system was tested on 11 different topics. The current system could be enhanced

by replacing the manual feature selection method with an automated approach. For this purpose, future research may consider neural network-based frameworks such as CNN, LSTM and BERT. Also, the same can be tested by drawing data from other CQAs such as Quora and Yahoo! Answers.

Another limitation of the paper lies in the use of only English questions. As many CQAs allow question-answering in non-English languages, future research could develop a language-independent model. In addition, there can be several grammatical errors in questions, making it tough for the model to identify the exact context. A model that is able to auto-correct grammatical mistakes could be developed in the future to address these issues. Hopefully, such research efforts will further enhance the value of CQAs to Internet users in the long run.

References

[1] Ahmad A, Feng C, Ge S et al. A survey on mining stack overflow: question and answering (q&a) community. *Data Technologies and Applications* 2018; 52(2): 190–247.

[2] Kafle S, de Silva N and Dou D. An overview of utilizing knowledge bases in neural networks for question answering. *Information Systems Frontiers* 2020; 22(5): 1095–1111.

[3] Li M, Chen L and Xu Y. Extracting core questions in community question answering based on particle swarm optimization. *Data Technologies and Applications* 2019; 53(4): 456–483.

[4] Loginova E, Varanasi S and Neumann G. Towards end-to-end multilingual question answering. *Information Systems Frontiers* ; 23.

[5] Roy PK, Ahmad Z, Singh JP et al. Finding and ranking high-quality answers in community question answering sites. *Global Journal of Flexible Systems Management* 2018; 19(1): 53–68.

[6] Gupta S, Kar AK, Baabdullah A et al. Big data with cognitive computing: A review for the future. *International Journal of Information Management* 2018; 42: 78–89.

[7] Li B and King I. Routing questions to appropriate answerers in community question answering services. In *Proceedings of the 19th ACM international conference on Information and knowledge management*. ACM, pp. 1585–1588.

[8] Saha RK, Saha AK and Perry DE. Toward understanding the causes of unanswered questions in software information sites: a case study of stack overflow. In *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering*. ACM, pp. 663–666.

[9] Chua AY and Banerjee S. Answers or no answers: Studying question answerability in stack overflow. *Journal of Information Science* 2015; 41(5): 720–731.

[10] Shen H, Liu G, Wang H et al. Socialq&a: An online social network based question and answer system. *IEEE Transactions on Big Data* 2017; 3(1): 91–106.

[11] Huang TC, Hsieh CH and Wang HC. Automatic meeting summarization and topic detection system. *Data Technologies and Applications* 2018; 52(3): 351–365.

[12] Paredes JN, Simari GI, Martinez MV et al. Netder: An architecture for reasoning about malicious behavior. *Information Systems Frontiers* 2021; 23: 185–201.

[13] Chua AY and Banerjee S. So fast so good: An analysis of answer quality and answer speed in community question-answering sites. *Journal of the American Society for Information Science and Technology* 2013; 64(10): 2058–2068.

[14] Correa D and Sureka A. Fit or unfit: analysis and prediction of'closed questions' on stack overflow. In *Proceedings of the first ACM conference on Online social networks*. ACM, pp. 201–212.

[15] Roy PK and Singh JP. Predicting closed questions on community question answering sites using convolutional neural network. *Neural Computing and Applications* 2020; 32(14): 10555–10572.

[16] Asaduzzaman M, Mashiyat AS, Roy CK et al. Answering questions about unanswered questions of stack overflow. In *Mining Software Repositories (MSR), 2013 10th IEEE Working Conference on*. IEEE, pp. 97–100.

[17] Zhou T. Understanding online knowledge community user continuance. *Data Technologies and Applications* 2018; 52(3): 45–58.

[18] Choi E and Shah C. Asking for more than an answer: What do askers expect in online q&a services? *Journal of Information Science* 2017; 43(3): 424–435.

[19] Harper FM, Raban D, Rafaeli S et al. Predictors of answer quality in online q&a sites. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. ACM, pp. 865–874.

[20] Toba H, Ming ZY, Adriani M et al. Discovering high quality answers in community question answering archives using a hierarchy of classifiers. *Information Sciences* 2014; 261: 101–115.

[21] Yao Y, Tong H, Xie T et al. Detecting high-quality posts in community question answering sites. *Information Sciences* 2015; 302: 70–82.

[22] Liu Y, Lin Z, Zheng X et al. Incorporating social information to perform diverse replier recommendation in question and answer communities. *Journal of Information Science* 2016; 42(4): 449–464.

[23] Liu Y, Li S, Cao Y et al. Understanding and summarizing answers in community-based question answering services. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING 2008)*. pp. 497–504.

[24] Liu M, Liu Y and Yang Q. Predicting werers for new questions in community question answering. In *International Conference on Web-Age Information Management*. Springer, pp. 127–138.

[25] Tian Q, Zhang P and Li B. Towards predicting the best answers in community-based question-answering services. In *ICWSM*. pp. 725–728.

[26] Oh HJ, Yoon YC and Kim HK. Finding more trustworthy answers: Various trustworthiness factors in question answering. *Journal of information science* 2013; 39(4): 509–522.

[27] Zheng X, Hu Z, Xu A et al. Algorithm for recommending answer providers in community-based question answering. *Journal of Information Science* 2012; 38(1): 3–14.

[28] Huna A, Srba I and Bielikova M. Exploiting content quality and question difficulty in cqa reputation systems. In *International Conference and School on Network Science*. Springer, pp. 68–81.

[29] Neshati M, Fallahnejad Z and Beigy H. On dynamicity of expert finding in community question answering. *Information Processing & Management* 2017; 53(5): 1026–1042.

[30] Srba I, Grznar M and Bielikova M. Utilizing non-qa data to improve questions routing for users with low qa activity in cqa. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*. ACM, pp. 129–136.

[31] Yan Z and Zhou J. Optimal answerer ranking for new questions in community question answering. *Information Processing & Management* 2015; 51(1): 163–178.

[32] Wang GA, Jiao J, Abrahams AS et al. Expertrank: A topic-aware expert finding algorithm for online knowledge communities. *Decision Support Systems* 2013; 54(3): 1442–1451.

[33] Fichman P. A comparative assessment of answer quality on four question answering sites. *Journal of Information Science* 2011; 37(5): 476–486.

[34] Liu X, Croft WB and Koll M. Finding experts in community-based question-answering services. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. pp. 315–316.

[35] Qiu X and Huang X. Convolutional neural tensor network architecture for community-based question answering. In *IJCAI*. pp. 1305–1311.

[36] Zhang Z and Li Q. Questionholic: Hot topic discovery and trend analysis in community question answering systems. *Expert Systems with Applications* 2011; 38(6): 6848–6855.

[37] Verma S, Sharma R, Deb S et al. Artificial intelligence in marketing: Systematic review and future research direction. *International Journal of Information Management Data Insights* 2021; : 100002.

[38] Espina A and Figueroa A. Why was this asked? automatically recognizing multiple motivations behind community question-answering questions. *Expert Systems with Applications* 2017; 80: 126–135.

[39] Ahasanuzzaman M, Asaduzzaman M, Roy CK et al. Mining duplicate questions of stack overflow. In *Mining Software Repositories (MSR), 2016 IEEE/ACM 13th Working Conference on*. IEEE, pp. 402–412.

[40] Figueroa A. Automatically generating effective search queries directly from community question-answering questions for finding related questions. *Expert Systems with Applications* 2017; 77: 11–19.

[41] Ponzanelli L, Mocci A, Bacchelli A et al. Improving low quality stack overflow post detection. In *Software Maintenance and Evolution (ICSME), 2014 IEEE International Conference on*. IEEE, pp. 541–544.

[42] Xia X, Lo D, Correa D et al. It takes two to tango: Deleted stack overflow question prediction with text and meta features. In *Computer Software and Applications Conference (COMPSAC), 2016 IEEE 40th Annual*, volume 1. IEEE, pp. 73–82.

[43] Zhang Y, Lo D, Xia X et al. Multi-factor duplicate question detection in stack overflow. *Journal of Computer Science and Technology* 2015; 30(5): 981–997.

[44] Zhang WE, Sheng QZ, Lau JH et al. Detecting duplicate posts in programming qa communities via latent semantics and association rules. In *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, pp. 1221–1229.

[45] Chintalapudi N, Battineni G, Di Canio M et al. Text mining with sentiment analysis on seafarers' medical documents. *International Journal of Information Management Data Insights* 2021; 1(1): 100005.

[46] Yang L, Bao S, Lin Q et al. Analyzing and predicting not-answered questions in community-based question answering services. In *AAAI*, volume 11. pp. 1273–1278.

[47] Grover P and Kar AK. Big data analytics: A review on theoretical contributions and tools used in literature. *Global Journal of Flexible Systems Management* 2017; 18(3): 203–229.

[48] Shah C, Kitzie V and Choi E. Questioning the question–addressing the answerability of questions in community question-answering. In *System Sciences (HICSS), 2014 47th Hawaii International Conference on*. IEEE, pp. 1386–1395.

[49] Correa D and Sureka A. Chaff from the wheat: characterization and modeling of deleted questions on stack overflow. In *Proceedings of the 23rd international conference on World wide web*. ACM, pp. 631–642.

[50] Srba I and Bielikova M. Why is stack overflow failing? preserving sustainability in community question answering. *IEEE Software* 2016; 33(4): 80–89.

[51] Dror G, Maarek Y and Szpektor I. Will my question be answered? predicting" question answerability" in community question-answering sites. In *ECML/PKDD (3)*. pp. 499–514.

[52] Jeon J, Croft WB and Lee JH. Finding similar questions in large question and answer archives. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. ACM, pp. 84–90.

[53] Agichtein E, Castillo C, Donato D et al. Finding high-quality content in social media. In *Proceedings of the 2008 international conference on web search and data mining*. ACM, pp. 183–194.

[54] Breiman L. Random forests. *Machine learning* 2001; 45(1): 5–32.

[55] Friedman JH. Greedy function approximation: a gradient boosting machine. *Annals of statistics* 2001; 29(5): 1189–1232.

[56] Rish I. An empirical study of the naive bayes classifier. In *IJCAI 2001 workshop on empirical methods in artificial intelligence*, volume 3. IBM, pp. 41–46.

[57] Chawla NV, Bowyer KW, Hall LO et al. Smote: synthetic minority over-sampling technique. *Journal of artificial intelligence research* 2002; 16: 321–357.

[58] Powers DM. Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. *Journal of Machine Learning Technologies* 2011; 2(1): 37–63.

[59] Lewis DD. Naive (bayes) at forty: The independence assumption in information retrieval. In *European conference on machine learning*. Springer, pp. 4–15.

[60] Kar AK and Dwivedi YK. Theory building with big data-driven research–moving away from the "what" towards the "why". *International Journal of Information Management* 2020; 54: 102205.

[61] Kar AK. What affects usage satisfaction in mobile payments? modelling user generated content to develop the "digital service usage satisfaction model". *Information Systems Frontiers* 2020; : 1–21DOI:https://doi.org/10.1007/

s10796-020-10045-0.