Feature Extraction to Filter Out Low-Quality Answers from Social Question Answering Sites

ABSTRACT

Social Question Answering sites (SQAs) are online platforms that allow Internet users to ask questions, and obtain answers from others in the community. SQAs have been marred by the problem of low-quality answers. Worryingly, answer quality on SQAs have been reported to be following a downward trajectory in recent years. To this end, existing research has predominantly focused on finding the best answer, or identifying high-quality answers among the available responses. However, such scholarly efforts have not reduced the volume of low-quality answers on SQAs. Therefore, the goal of this research is to extract features in order to weed out low-quality answers as soon as they are posted on SQAs. Data from Stack Exchange was used to carry out the investigation. Informed by the literature, 26 features were extracted. Thereafter, machine learning algorithms were implemented that could correctly identify 85% to 96% of low-quality answers. The key contribution of this research is the development of a system to detect subpar answers on the fly at the time of posting. It is intended to be used as an early warning system that warns users about answer quality at the point of posting.

1. INTRODUCTION

Social Question Answering sites (SQAs) refer to online platforms that allow Internet users to ask questions on any topic, and obtain answers from others in the community [1–3]. These platforms leverage the wisdom of crowds [4], and facilitate prompt information seeking among the masses [5–9]. Examples of popular SQAs include Yahoo! Answers and Stack Exchange, which archive the questions asked and the answers submitted in response. These archives remain available for browsing among anyone with Internet access.

Despite their obvious advantages, SQAs are marred by the perennial problem of low-quality answers [10]. Existing research suggests that the problem primarily stems from the lack of adequate gate-keeping [11]. While some users post high quality answers out of altruism [12] and the motivation to gain recognition in the community [13,14], others could end up posting subpar answers due to either lack of domain knowledge or the deliberate motivation to abuse the functionality of SQAs [15,16]. Given the little editorial control, separating high quality answers from those that are subpar is challenging.

More worryingly, answer quality on SQAs have been reported to be following a downward trajectory in recent years [10]. For example, the proportion of low-quality content on Stack Overflow has grown from 4% in 2011 to 16% in 2014 [10]. More recently, [17] revealed that Chinese SQAs such as Baidu Zhidao also attract large volumes of low-quality answers.

Hence, the problem of low-quality answers on SQAs now warrants scholarly attention. The current mechanism that SQAs employ for answers' quality control requires human intervention. For any submitted question, the asker has the option to select an answer as the best answer [18]. Up-votes and down-votes from the online community is also considered. For example, on SQAs such as Stack Exchange, answers that are down-voted by many can be identified for deletion by reputed users. The deletion requests are then manually inspected by the site moderators.

This quality control mechanism is problematic for two reasons. One, it is not only slow but also inefficient, especially given the huge volume of answers posted on SQAs on a daily basis [13,17]. Two, this mechanism can be annoying to users who find their answers deleted suddenly from the SQA without any warnings. Most prior works in this field have focused on finding the best answer, or identifying high-quality answers among the available responses [19–22]. While such efforts are valuable, they have not helped to reduce the volume of low-quality answers on SQAs.

For these reasons, the goal of this research is to weed out low-quality answers as soon as they are posted on SQAs. Feature engineering is used to achieve this. The key contribution of this research is the development of a system to detect subpar answers on the fly at the time of posting. It is intended to be used as an early warning system that warns users about answer quality at the point of posting. This in turn offers them the opportunity to improve their answers before posting. In this way, the system supports users to post good answers, and prevents attempts to abuse SQAs. It also obviates the need for human intervention on the part of SQA moderators. The system can not only improve the quality of the SQA content proactively but also be used by the site moderators to assess the content quality of the site from time to time. The major contributions are as follows:

1 We proposed a machine learning based automated system to filter out low-quality answers.
2 A limited number of textual and non-textual features are used to build the model , and this in turn reduces the overall model complexity.
3 The data imbalance issues were addressed with SMOTE and ADASYN oversampling techniques. The outcomes on the balanced dataset were found better than that on the imbalanced dataset.

The rest of the paper is organised as follows. In Section 2, the literature review is presented. Section 3 describes the methodology. This is followed by the results in Section 4. In Section 5, the implications of the proposed work are discussed. Section 6 concludes the article with notes on its limitations.

2. LITERATURE REVIEW

The quality of answers on SQAs can range from excellent to abysmal [15,16,23–25]. A dominant strand of SQA research focuses on answer quality. For example, John *et al.* [26] proposed a model identifying factors that improve answer quality. Three groups of features were considered, namely, textual, social and content appraisal. Content appraisal features were found to play a major role in predicting high-quality answers.

Lee *et al.* [27] assigned a score to each voter to capture the level of agreement among those who up-voted or down-voted answers. This voting score was used as a feature to predict the best answer. These works typically use machine learning-

based classifiers. For example, in a study on Yahoo! Answers [28], features were extracted frim questions, answers and users. With a dataset of questions having at least five answers, a classification accuracy of nearly 80% was achieved in predicting the best answers.

Sahu *et al.* [29] identified a set of tag-based features to find the best answer. Their model achieved an accuracy of 69%. Tian *et al.* [21] found that answers posted earlier have a higher chance of being accepted as the best answer. They extracted 16 different features to find the best answer among the pool of answers and the result confirmed that contextual features played a major role. Their model achieved an accuracy of 72.27%. Yao *et al.* [22] proposed a system that detected high-quality answers to a posted question using users' voting behaviours.

Blooma *et al.* [30] used both social and textual features to predict high quality answers, and achieved an accuracy of 85%. In a similar study, the number of votes obtained by anthe answer was found to be a helpful feature to identify high-quality answers [31]. Blooma *et al.* [19] proposed a model to find the best answer to a given question. They extracted a number of textual and non-textual features and showed that non-textual features such as answerer/asker answerer/asker authority (defined as the number of best answers provided by the user) and user reputation were not relevant. Textual features including spelling errors and answer length also had little to do with the prediction of best answer. However, answer readability was a significant predictor of high-quality answers [32].

However, most of the features mentioned in these works such as [30] and [31] were evaluated manually. This does not help design an automated system to address the problem of low-quality answers, which remains a pressing problem. For instance, Srba & Bielikova [10] analyzed the content quality of Stack Overflow covering 2011–2014 and found a gradual decline. A possible reason for this decline is the presence of users known as " nNoobs" and " hHelp vampires" who are continuously posting low quality and duplicate questions on the website [33]. They also identified another group of users named " rReputation cCollectors" who produce answers to those low-quality questions.

Kucuktunc *et al.* [34] proposed a model for sentiment analysis on Yahoo! Answers. They found the answers posted on weekends have more positive sentiments as compared with those submitted onto weekdays. Li *et al.* [35] proposed a multi-criteria decision-making system to evaluate answer quality. They verified the model with data from five Chinese SQAs. Factors such as coverage, politeness, and readability were key determinants of high-quality answers. Elalfy *et al.* [36] proposed a hybrid model for best answer prediction on Stack Overflow. Their model used question-answer features, answer content features, and answer-answer features to predict the best answer and achieved a promising accuracy of 88.65%.

Zhang *et al.* [37] developed a model for developing high-quality answers. With the help of 382 contributors, a survey report was created for identifying key factors that make high-quality answers. Some of the factors that emerged include social interaction, community knowledge quality, topic richness, and personalised recommendations. Palomera *et al.* [38] used a semi-supervised learning mechanism [39] to get informative content from SQAs, and achieved accuracies of 84.25% and 74.41% with regards to questions and answers respectively. Fu *et al.* [40] did a quality assessment of SQAs with 23 user-identified features and 24 data-centric features. Their analysis confirmed that the importance of content-based , user-based, and review-based features in predicting the quality of SQA posts, users and review features played key roles in the decision-making system for the quality of the CQA post.

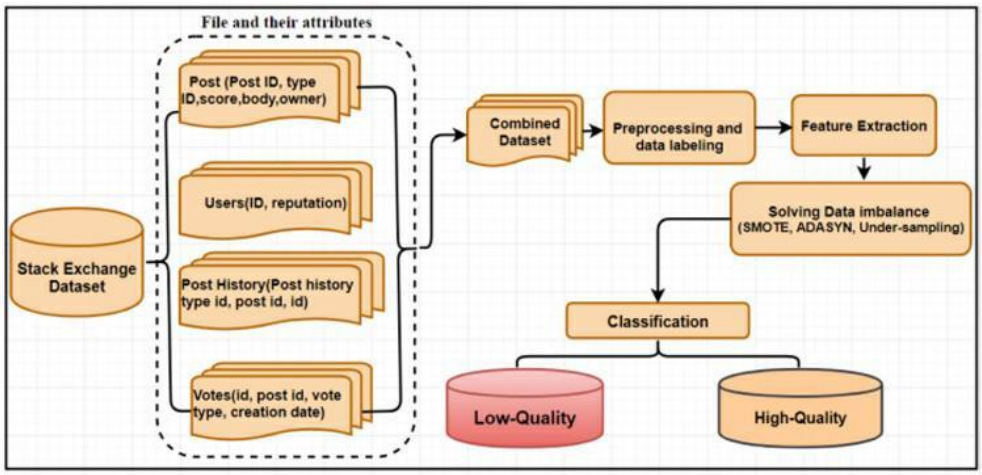
Tang *et al.* [41] predicted the response time for the posted query on the CQA platform with the help of the answer's interest, activeness on the platform, category of the questions and difficulty level of the question. Chergui *et al.* [42] developed a model to get the best suitable answer from the archive using similarity matching between a new question and the old questionsfor the new post. They used semantic inference with Bayesian inference to handle the semantic uncertainty. Their model highly correlated with human judgment. This motivated us to find an automated mechanism to further improve the quality of answers on SQAs. The idea is to come up with an early warning system that will prompt answerers to improve their answer on the fly if their entry is subpar.

3. RESEARCH METHODOLOGY

This research aims to weed out low-quality answers from SQAs. This is done based on answers' textual content [6,19,20,24,26,27,29,32] and votes [27,29,31,32,43] as these have been widely used in the SQA literature on answer quality. The dataset of Stack Exchange, a popular SQA, containing questions and answers posted from August 2008 to December 2016 was used for this work. It contains several XML files related to different attributes of the post (questions and answers)[1].

The dataset for this research is prepared using the information extracted from four files namely: Post.XML, Users.XML, PostHistory.XML, and Votes.XML. The steps followed are shown in Figure 1.

Figure 1: Flow diagram to weed out low-quality answers from SQAs **[Q14]**



3.1. Combined Dataset and Preprocessing

The dataset included Stack Exchange content on a variety of topics. This research analyzes answers corresponding to the top 20 topics in terms of the number of available data points. A classifier was trained for each topic. The variety of topics ensured the robustness of our classifier. The list of selected topics with their data dimensions is shown in Table 1.

Table 1: The datasets used for this research with their statistics

Sl	Dataset name	Total Answers	High-quality answer	Low quality answer	% low-quality answer
1	Astronomy	4,526	3,900	626	13.83
2	Biology	13,832	12,387	1,445	10.44
3	Board Games	12,960	11,283	1,677	12.94
4	Buddhism	7,767	6,365	1,402	18.05
5	Chemistry	15,985	14,098	1,887	11.80
6	Chess	5,946	4,989	957	16.09
7	Chinese	8,407	6,693	1,714	20.39
8	Christianity	19,175	14,868	4,307	22.46
9	Economics	3,435	2,874	561	16.33
10	Engineering	2,797	2,421	376	13.44
11	Fitness	12,087	9,597	2,490	20.60
12	German	14,983	13,107	1,876	12.52
13	Health	1,271	1,114	157	12.35
14	History	9,900	8,634	1,266	12.79
15	Islam	8,577	5,652	2,925	34.10
16	Judaism	30,012	26,023	3,989	13.29
17	Linguistic	6,192	5,110	1,082	17.47

18	Mechanics	12,688	10,615	2,073	16.34
19	Philosophy	14,622	10,868	3,754	25.67
20	Programmer	75,838	62,219	13,620	17.96

We filtered the desired fields and saved them in "CSV" files. The complete list of files with their attributes used is shown in Figure 1. The combined dataset was filtered first, and then it was labelled into two classes, answers with low quality (Class 0) and answers with high quality (Class 1). Answer quality was determined based on users' votes. We consider the answers that fail to obtain votes or obtain negative votes within an average time to get the votes to be of low quality.

NThe newly posted answers or answers which didwere not meet the require ment of attracting d time to get the user's votes are not considered for this study. However, as a question becomes old, it might not attract a lot of viewers. Hence answers posted after a certain amount of time may not receive votes despite being of good quality. Keeping such answers in the training data of the classifier may lead to misclassification. We therefore use d voting history data to identify such answers. Votes are considered to be a measure of interest in users. It is assumed that users are interested in the question as long as questions and their answers obtain votes. When the users stop voting the question or its answers, it indicates that the interest in the question has dropped. Answers posted after this drop in interest may not receive votes even if they are of high quality.

We check ed the voting history of all answers, and the answers posted after the last vote weare removed from the dataset. ThereafterAfter removing such answers from the dataset, the answer s that did not doesn't receive any votes (or receive d net negative votes) wereis classified as a low-quality answer s (Class 0). The remaining answers weare considered high-quality answers (Class 1).

3.2. Feature Extraction

We extracted a set of textual features that ranged from nouns, verbs and adjectives to readability and wrong (non-dictionary) words .such as "Noun, Adjective, Verbs, Readability, Wrong words, etc.", Derived features such as "N number of answers and, A answer-answer similarity as well as , etc.", and P personal features such as "overall reputation, reputation on the same domain, and reputation on a different domain were also extracted., etc." . The complete list of features is shown in Table 2 . These were are used to train the machine learning algorithms. After extracting features from the dataset, we appl iedy machine learning algorithms on the feature vector. In particular, We applied three classification algorithms were employed: (i) Naive Bayes [44] (ii) Gradient Boosting [45], and (iii) Random Forest [46]. The detailed results based on these classification techniques are explained in section 4.

Table 2: List of selected features

Feature Types	Name of the feature (*Represented as*)		Explanation
Textual	Number of Nouns (*Noun*)	Total number of nouns is counted	
	Number of Verbs (*Verbs*)	Total number of Verb is counted	
	Number of Adjective (*Adjective*)	Total number of Adjective is counted	
	Entropy (*Entropy*)	The average entropy of each word in the answer text, indicating the amount of information being produced in the answer is calculated. The formula for entropy is given as follow $H(X\|Y) = \sum_{i,j} p(x_i, y_j) \log \frac{p(x_i)}{p(x_i, y_j)}$ where $p(x_i, y_j)$ is the probability of occurrence of a word in the answer text	

	Number of Difficult word (*Difficult word*)	If a word is not present in the ~~a~~ predefined 3 , 000 familiar English word's dictionary, it is termed as a difficult word. All such words are counted.
	Lex diversity (*Lex diversity*)	It is defined as the ratio of total unique words to the total number of words present in the answer.
	Number of single letter word (*One_letter*)	Total number of one letter word is counted
	Number of words having only two letters	Total number of two letter word is counted.
(Two letter)	~~Total number of two letter word is counted~~	
	Flesh reading ease score (*Flesh_RE*)	For each answer, Flesch reading ease score is calculated. It is a popular measure of ~~popular~~ reading ease ~~determiner~~. It gives a score between 0–100, where 0 indicate s ~~d the~~ poor readability.
	Dale-Chale score (*Dale_RS*)	By using a set of 3,000 words that American fourth-grade students are familiar with, the Dale Chale score is calculated for every answer.
	Total number of words (*total_words*)	After removing the stop words, total number of words is calculated.
	Set length (*Set_length*)	The number of unique words present in the answer is termed as set length.
	Number of Stop words (*Stop_words*)	Total number of stop words is counted.
	Number of Wrong word (*Wrong_words*)	W ~~The w~~ ords that are ~~is~~ not present in E ~~e~~ nchant dictionary is treated as wrong words and it is counted for every answer.
	Number of Points (*li_tags*)	The number of *li* tags is counted to count the number of bullet point present in the answer.
	Number of Code Snippet (*Code_tags*)	The number of code tag is counted to check the number of code snippet present in the answer.
	Number of words having more than two letters (*Longer_letter_word*)	Total number of longer letter word is counted
	Number of Modifications done on answers (*Modified*)	The number of times the answer has been modified is counted

Derived	Number of answers posted over a question (*N_Ans*)	Total number of answers present on the given question is counted
	Similarity between new and earlier posted top scorer answers (*A_A_Sim*)	The average cosine similarity of the answer with the top 3 answers of the question is calculated using: $Similarity = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2} \quad \sqrt{\sum_{i=1}^{n} B_i^2}}$ where A_i and B_i are vectors of answer, n is the number of unique words used in A and B.
	Question-answers similarity (*Q_A_sim*)	The average cosine similarity of the question with the answer is calculated.
	Answerer's domain and question topic similarity (*TopicSimilarity*)	The similarity between the topic of the question and the topics of previous answers given by the answerer is calculated. To obtain the topic of the question, we apply Latent Dirichlet Allocation (LDA) on the question and by setting the number of topics as one and the number of words as seven. We append all answers of the answerer and apply LDA to it. After obtaining the topic words of question and answer, we apply cosine similarity on them.
Personal	Answerer's reputation on same domain (*TopicRepSD*)	Is it calculated using the formula: $TopicSimilarity * R$ where *R* is the user's reputation.
	Answerer's reputation on different domain (*TopicRepDD*)	It is calculated a $(1 - TopicSimilarity) * R$
	Number of accepted answers on different domain of the answerer (*AcceptedAnsDD*)	Total number of answers accepted on different domain is counted.
	User reputation (*User_Rep*)	The reputation of the answerer is treated as one of the features.

3.3. Data Imbalance

For all topics in the dataset, the number of instances in Class 0 was less than Class 1, as shown in Table 1. We applied three techniques to resolve the issue of data imbalance. They are (i) Random Under-Sampling[46], (ii) SMOTE (Synthetic Minority Oversampling) [47], and (iii) ADASYN (Adaptive Synthetic Sampling)[48]. In Random Under-Sampling, we randomly removed samples from the majority class until the majority class and minority class samples became equal in number. SMOTE is an oversampling technique where synthetic samples are generated for each minority class sample between the selected sample and its nearest neighbour. In ADASYN, synthetic samples are generated between the selected sample and its nearest neighbour, but a density distribution function gives the number of samples generated for each minority class sample. This density distribution function gives more weight to the samples near the majority and minority class boundaries.

The three machine learning-based classifiers were then used to achieve our objective. The complete results obtained using the different classifiers are presented and discusses nextin the result section 4.

4. RESULT ANALYSIS

We use precision (PRE), recall (REC) and F1-Score as performance measurement parameters for our system [49].

4.1. Results with Imbalance Dataset [1]

Datasets across 20 different topics of Stack Exchange were collected and labelled as low- and high-quality answers by the techniques presented in section 3.1. Further, we divided the dataset into train and test sets in the ratio of 70:30. Firstly, we present the results of the "Programmer" dataset (Table 3, Table 4, and Table 5) and then with the best-identified model, datasets with the other topics are evaluated (Table 6, Table 7, and Table 8) . The detailed results of all datasets are presented later in Table 9. The system was trained and tested with three classification algorithms, namely (i) Naïve Bayes (NB) [44], (ii) Gradient Boosting (GB) [45], and (iii) Random Forest (RF) [46]. The detailed result of the Naive Bayes classifier is presented in Table 3 for the "Programmer" topic dataset.

Table 3: Results on the "Programmer" dataset with NB Classifier

Class	*PRE*	*REC*	*F1*
low quality (0)	0.21	0.84	0.33
high quality (1)	0.90	0.30	0.45

As can be seen from Table 3, PRE for Class 1 is good (0.90), but for Class 0 is poor (0.21) . REC for Class 1 is low (0.30) compared to Class 0 (0.84). And the F1 both classes (0.33 for Class 0 and 0.45 for Class 1) are not satisfactory. Similar results are obtained with the Gradient Boosting classifier too, as seen from Table 4; the Gradient Boosting classifier is performed slightly better than the Naive Bayes classifier. REC of Class 0 (our target class) continues to be abysmal (0.08). [2]

Table 4: Results on the "Programmer" dataset with GB Classifier

Class	*PRE*	*REC*	*F1*
low quality (0)	0.67	0.08	0.14
high quality (1)	0.83	0.99	0.90

Table 5: Results on the "Programmer" dataset with RF Classifier

Class	PRE	*REC*	*F1*
low quality (0)	0.67	0.11	0.19
high quality (1)	0.84	0.99	0.91

With the Random Forest classifier, the performance of the model for Class 0 improved, but still remained far from satisfactory , as evident from Table 5. Even the Receiver Operating Characteristics (ROC) curve for the Random Forest classifier, shown in Figure 2 , was very low. The confusion matrix (Figure 3) confirmed that 100 high-quality answers and 1,601 low-quality answers were misclassified. The test sample consisted of 8,282 high quality and 1,800 low-quality answers, indicating that the dataset was imbalanced. The imbalanced nature of the dataset made the classifier very good at detecting high-quality answers compared to low-quality answers. This class imbalance problem is consistent across all the 20 topics of the dataset, as shown in Table 1. Across all topics, total high-quality answers were 232,817 while total low-quality answers were 48,184, with 17.14% of answers are low quality.

Figure 2: ROC curve with RF classifier for "Programmer" dataset

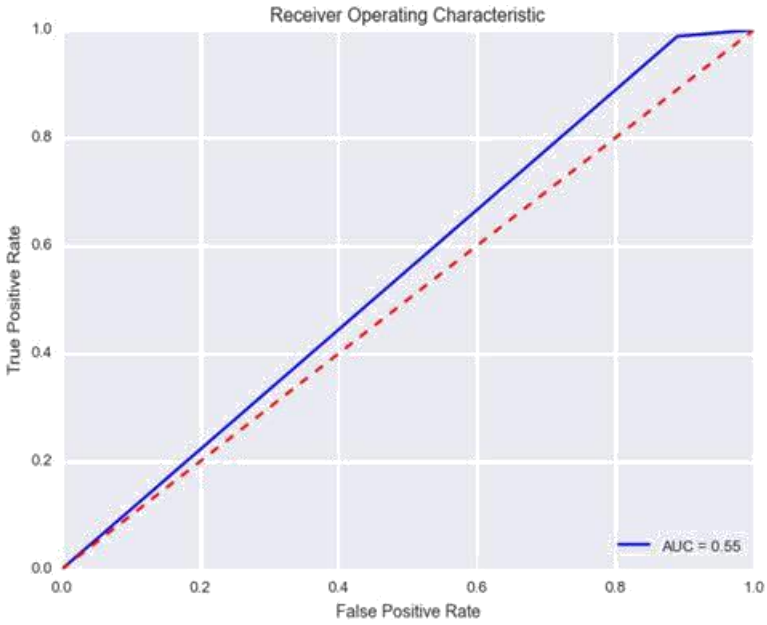
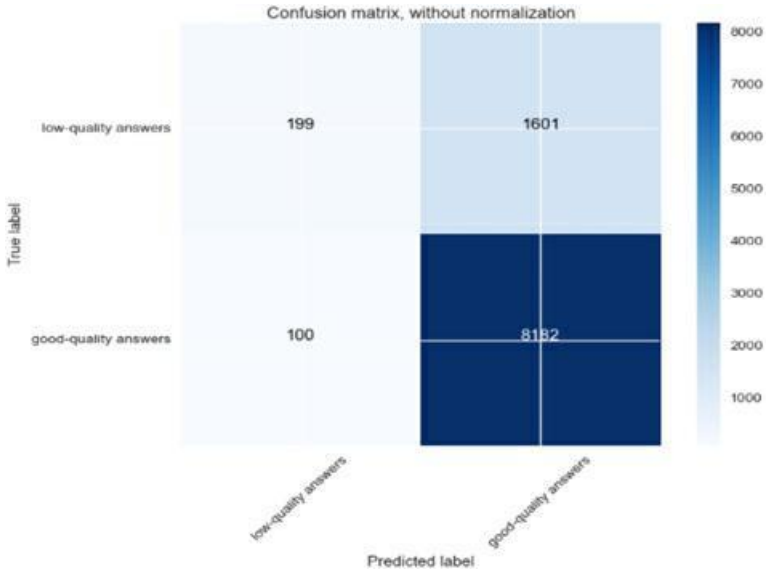


Figure 3: Confusion Matrix after applying RF Classifier on "Programmer" dataset



4.2. Results with Balanced Dataset

The earlier three classification algorithms were applied again on the dataset balanced using techniques described in the methodology section. The classification results on the dataset balanced using Random-Under sampling are reported in Table 6. The Random Forest classifier is found to outperform the other two classification techniques. The classification results with dataset balanced using SMOTE and ADASYN are shown in Tables7 and 8, respectively. As can be seen from Tables 6, 7 and 8, the result of Random Forest with data balanced with ADASYN turned out to be the best so far. The proposed system achieved PRE, REC and F1 value as 0.93, 0.85, and 0.89, respectively, for the "Programmer" topic as shown in Table 8, and the ROC value was 0.89 as shown in Figure 4.

Figure 4: ROC curve on RF with ADASYN on "Programmer" Topic

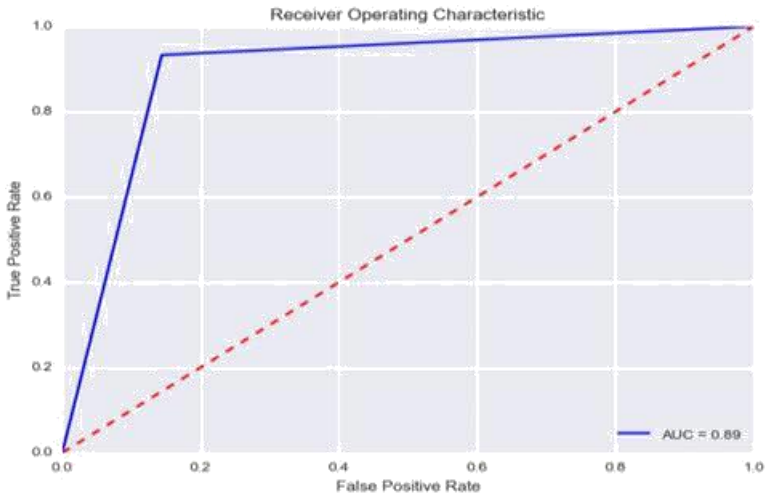


Table 6: Results on balanced (with Random Under-sampling) "Programmer" dataset with different classifiers

Classifier	Class	*PRE*	*REC*	*F1*
Random Forest	0	0.77	0.76	0.77
	1	0.74	0.75	0.74
Gradient Boosting	0	0.65	0.42	0.51
	1	0.75	0.89	0.81
Naive Bayes	0	0.37	0.85	0.52
	1	0.80	0.29	0.42

Table 7: Results on balanced (with SMOTE) "Programmer" dataset with different classifiers

	Class	*PRE*	*REC*	*F1*
Random Forest	0	0.91	0.74	0.81
	1	0.78	0.93	0.85
Gradient Boosting	0	0.81	0.81	0.81
	1	0.81	0.80	0.81
Naive Bayes	0	0.55	0.89	0.68
	1	0.71	0.26	0.39

Table 8: Results on balanced (with ADASYN) "Programmer" dataset with different classifiers

Classifier	Class	*PRE*	*REC*	*F1*
Random Forest	0	0.93	0.85	0.89
	1	0.85	0.93	0.89
Gradient Boosting	0	0.69	0.77	0.73
	1	0.74	0.65	0.69
Naive Bayes	0	0.54	0.88	0.67

	1	0.66	0.23	0.35

We repeated the experiment with the remaining 19 other datasets and sound similar results. To avoid repetition of the result, we have further reported only the results of the Random Forest classifier with different datasets. As shown in Table 9, the lowest ROC value is 0.86 whereas the best ROC value is 0.94. In general, with Science and Engineering topics, the results are better compared to "Linguistic ", "Fitness ", and "History ". The AUC-ROC curve for the remaining topics was also evaluated but are not included for brevity. Nonetheless, the obtained AUC value s for of all the topics are shown in Table 9.

Table 9: Model performance on different topics of Stack Exchange

Topic	Class	*PRE*	*REC*	*F1*	AUC
Linguistics	0	0.91	0.86	0.89	
	1	0.88	0.92	0.90	0.90
Philosophy	0	0.89	0.81	0.85	
	1	0.83	0.90	0.86	0.86
Economics	0	0.93	0.89	0.91	
	1	0.90	0.94	0.92	0.92
Engineering	0	0.93	0.91	0.92	
	1	0.92	0.93	0.93	0.93
Mechanics	0	0.93	0.91	0.92	
	1	0.91	0.93	0.92	0.92
Board Games	0	0.95	0.92	0.93	
	1	0.92	0.95	0.93	0.94
Chess	0	0.93	0.89	0.91	
	1	0.90	0.93	0.92	0.92
Fitness	0	0.90	0.84	0.87	
	1	0.87	0.92	0.89	0.89
Health	0	0.90	0.86	0.88	
	1	0.87	0.91	0.89	0.90
History	0	0.94	0.93	0.94	
	1	0.93	0.94	0.94	0.93
Chinese	0	0.90	0.83	0.86	
	1	0.85	0.91	0.88	0.87
German	0	0.96	0.91	0.93	
	1	0.91	0.96	0.97	0.93

Buddhism	0	0.89	0.87	0.88	
	1	0.87	0.90	0.88	0.89
Christianity	0	0.90	0.89	0.89	
	1	0.89	0.90	0.91	0.89
Islam	0	0.93	0.91	0.92	
	1	0.91	0.93	0.92	0.87
Judaism	0	0.96	0.89	0.92	
	1	0.90	0.96	0.93	0.92
Astronomy	0 1	0.93	0.91	0.92	
		0.91	0.93	0.92	0.91
Biology	0	0.96	0.92	0.94	
	1	0.92	0.96	0.94	0.94
Chemistry	0	0.95	0.92	0.94	
	1	0.93	0.95	0.94	0.94

4.3. Feature Importance

We have used a total of 26 different features to achieve our objective. To achieve our objective of "detecting low-quality answers" on Stack Exchange, we have extracted several textual and non-textual features from the dataset. We checked the importance of these features on different topics of Stack Exchange. It might be possible that some features have more impact than others for specific topics. [3]

The feature importance graph on the "Programmer" topic is shown in Figure 5. On the "Programmer" topic, the features answer-answer similarity (A_A_Sim) was the most important. In contrast, the readability-based features such as Dale Chale score (Dale RS), Flesh reading ease (Flesh RE), and User reputation (User Rep) are less influential. On the "Linguistic" topic, except answer-answer similarity (A_A_Sim), Flesh reading ease (Flesh RE), and Difficult words were found to be the most effective features as shown in Figure 6. Since answers posted on the "Linguistic " topic are expected to be more readable and linguistically correct without any special characters or formulas (which are likely to appear in the "Programmer" topic dataset), all other features have average contributions over the different topics. Overall, this analysis confirmed that all 26 features contributed to classification performance over the different topics.

Figure 5: Feature importance graph on the "Programmer" topic

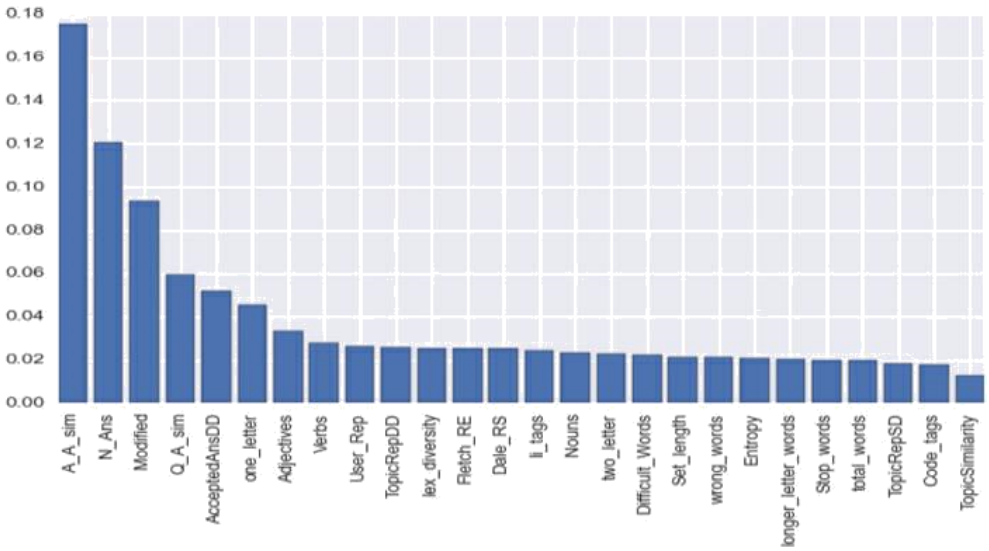
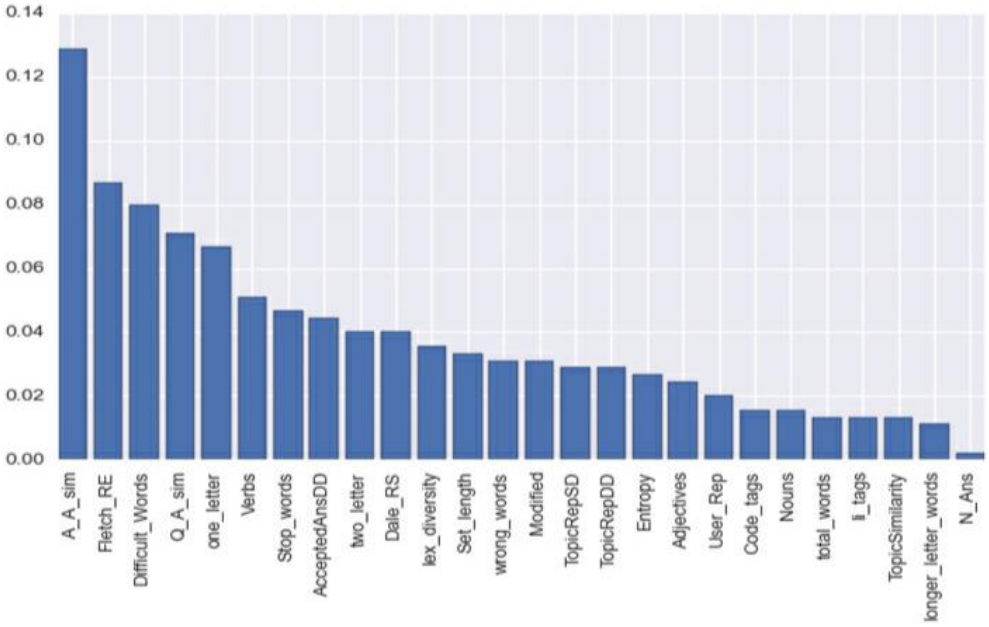


Figure 6: Feature importance graph on the "Linguistic" topic



5. DISCUSSION

In this article, we proposed an early warning system for filtering out low-quality answers f rom SQAs~~or SQA sites~~. The system wa~~i~~s trained to detect low-quality answers using a Random Forest classifier. The PRE value (meaning what fraction of system detected low-quality answers wea~~r~~e really low quality) ranged from~~, is in the range of~~ 85% to 96%. So, our system can correctly identify 85% to 96% of low-quality answers. The proposed system can also find 81% to 96% of low-quality answers from posted answers as indicated by the REC column of Table 9. The system wa~~i~~s trained and tested across 20 different topics ~~belonging to linguistic, Chinese, fitness, engineering, etc~~. The results are largely consistent across the different ~~se varying topics,~~ highlighting the generalizability of ~~indicating that~~ our system is general enough to be applied ~~to any topic~~.

The system's performance wa~~i~~s better for Science and Engineering topics such as ~~Engineering, Science,~~ "Programmer " compared to the likes of "Chinese ", "Linguistic ", and "Fitness ~~, etc~~. F~~The f~~eatures such as code snippets (code tags) and bullet points (li tags) play ed a crucial role in segregating low- and high-quality answers in these topics~~subject categories~~. Such~~The above~~ features wea~~r~~e mostly present in Science and Engineerinf topics ~~"Science" topics~~, but not in topics such as "Linguistic" and "Chinese"~~linguistic topics such as Chinese, language, etc~~.

In the present research, we have also raised the issue of data imbalance and adopted appropriate techniques to resolve it~~that~~. Our system has reported better results compared to similar work [50]. The average *PRE* of low-quality question detection obtained by [50] was 0.68 whereas, it is on average around 0.90 in our case. The proposed system uses primarily textual features of answers, with 20 out of 26 features being textual. Textual features make this system easily adaptable to

other SQA sites such as Quora and Yahoo! Answers. YA, etc.

5.1. Theoretical Contributions

The main contribution of this research is the extraction of effective features to weed out low-quality answers from SQAs, particularly Stack Exchange. Twenty-six features were extracted from the answer text as well as from user reputation and post history. Twenty out of those 26 features were extracted from the answer text only. The other six features were extracted from the user's reputation and answering behaviour. In an earlier research, Toba *et al.* [21] used 40 different features on the YA dataset to report *PRE* and *REC* of 0.58 and 0.69, respectively. We experimentally reduced the number of features from 40 to 26 to improve the performance of our system. All features used in our system can be extracted in an automated manner, which gives an edge over the system proposed by Blooma *et al.*, 2012 [30] where they used manually -tagged features. The proposed model is therefore easily implementable on other SQAs as the features can be easily computed regardless of the platform. Our analysis of over 20 different topics reveals that, on average, 17.14% of answers are of low quality, which is in line with prior research [10].

5.2. Implications for Practice

The proposed system can be best put to use as an early warning system for answers posted on SQA sites. SQA users can be warned about their answer being low in quality as soon as they finish typing the answer. The users will also get some suggestions regarding their answers, such as (i) the formatting of the answer, (ii) the readability score of the answer text, (iii) using bullets to improve the clarity of the answer. This urge to improve the quality of the answer will make the user more engaged with the SQA site. When put in practice, this system will flag low-quality answers being posted on the site. However, the system will not disturb the user if their answer does not belong to the low-quality class.

There are users called "R reputation cCollectors" [13] on Stack Exchange who posting a huge number of low-quality answers to gain reputation points [10]. Our early warning system can warn themsuch 'Reputation Collectors about the subpar quality of their answer s while posting. These 'R reputation cCollectors can then modify their answers to meet the desired quality standard because doing so will also increase their chances of obtaining votes. Since 'Reputation C they usuallyollectors post a huge volume of answers, this strategy can potentially turn r"Reputation cCollectors" from athe weakness of SQAs to a strength.

From the site moderators' point of view, the current system can be used as a tool to detect low-quality answers from time to time. This system can also be used to find users who continuously post low-quality answers so that action can be taken against them. The proposed system could be adapted to flag sends a notification to the site's moderators when if a user goes ahead with posting low-quality answers posts the same answer even after being warned. This makes the job of site moderators a' bit easier because they will be having the list of low-quality answers (along with the corresponding answerers) as soon as they are posted.

6. CONCLUSION

The current research highlights the issue of low-quality content on SQA sites such as Stack Exchange and presents a machine learning based solution for that. We have also raised the issue of data imbalance prevailing in SQA sites. Various data balancing techniques were used to balance our dataset. The ADASYN data balancing technique was found to be the best. Twenty-six features were extracted to train our classifier. The system showed good results across a variety of topics on Stack Exchange, highlighting its generalizability. Though our proposed system can be used to detect low-quality answers, it is unable to exactly specify what factors make the quality of the answer low.

Future work can be focused on building a dynamic suggestion system that can find the weaknesses in an answer and pinpoint the user about their mistakes. This will warn users that their answer is of low quality and inform them what is making their answer subpar. The other limitation of the proposed system is that it was tested with different topical data from Stack Exchange only. There are other SQAs with distinctive functionalities that have not been explored. Moreover, the current system uses only syntactic textual features. The use of semantic textual features along with some expert answers as a baseline may be explored to improve it further. Some deep learning-based systems with advanced word representation schemes such as Word2Vec and, Glove may also be exploited. By utilising a combination of

rule-based techniques, statistical approaches, and error analysis approaches, an advanced model can hopefully be developed in the future to better weed out the low-quality answers from SQA s.

Note

1 https://archive.org/details/stackexchae[accessed online in January, 2017.

References

1 **G. Blanco, R. Prez-Lpez, F. Fdez-Riverola, and A. M. G. Loureno**, "Understanding the social evolution of the java community in stack overflow: *A 10-year study of developer interactions*," *Future Gener. Comput. Syst.*, Vol. 105, pp. 446–454, 2020. doi:10.1016/j.future.2019.12.021

2 **J. Herrera, D. Parra, and B. Poblete**, "Social QA in non-CQA platforms," *Future Gener. Comput. Syst.*, Vol. 105, pp. 631–649, 2020. doi:10.1016/j.future.2019.12.023

3 **J. Yin, W. X. Zhao, and X. M. Li**, "Type-aware question answering over knowledge base with attention-based tree-structured neural networks," *J. Comput. Sci. Technol.*, Vol. 32, pp. 805–813, 2017. doi:10.1007/s11390-017-1761-8

4 **J. Surowiecki, M. P. Silverman, *et al.***, "The wisdom of crowds," *Am. J. Phys.*, Vol. 75, pp. 190–192, 2007. doi:10.1119/1.2423042

5 **A. Y. Chua, and S. Banerjee**, "So fast so good: An analysis of answer quality and answer speed in community question-answering sites," *J. Am. Soc. Inf. Sci. Technol.*, Vol. 64, pp. 2058–2068, 2013. doi:10.1002/asi.22902

6 **A. Y. Chua, and S. Banerjee**, "Measuring the effectiveness of answers in yahoo! answers," *Online Inf. Rev.*, Vol. 39, pp. 104–118, 2015. doi:10.1108/OIR-10-2014-0232

7 **V. Kitzie, and C. Shah**, "Faster, better, or both? looking at both sides of online question-answering coin," *Proceedings of the American Society for Information Science and Technology*, Vol. 48, pp. 1–4, 2011. doi:10.1002/meet.2011.14504801180

8 **L. Mamykina, B. Manoim, M. Mittal, G. Hripcsak, and B. Hartmann**, "Design lessons from the fastest Q&A site in the west," in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, 2011, pp. 2857–2866. ACM

9 **T.-L. Wong**, "Answering reachability queries on incrementally updated graphs by hierarchical labeling schema," *J. Comput. Sci. Technol.*, Vol. 31, pp. 381–399, 2016. doi:10.1007/s11390-016-1633-7

10 **I. Srba, and M. Bielikova**, "Why is stack overflow failing? preserving sustainability in community question answering," *IEEE Softw.*, Vol. 33, pp. 80–89, 2016b. doi:10.1109/MS.2016.34

11 **Y. Zhang, D. Lo, X. Xia, and J.-L. Sun**, "Multi-factor duplicate question detection in stack overflow," *J. Comput. Sci. Technol.*, Vol. 30, pp. 981–997, 2015. doi:10.1007/s11390-015-1576-4

12 **B. Yang, and S. Manandhar**, "Tag-based expert recommendation in community question answering," in Advances in Social Networks Analysis and Mining (ASONAM), 2014 IEEE/ACM International Conference on, 2014, pp. 960–963. IEEE.

13 **P. K. Roy, J. P. Singh, A. M. Baabdullah, H. Kizgin, and N. P. Rana**, "Identifying reputation collectors in community question answering (CQA) sites: exploring the dark side of social media," *Int. J. Inf. Manage.*, Vol. 42, pp. 25–35, 2018b. doi:10.1016/j.ijinfomgt.2018.05.003

14 **X. Wang, C. Huang, L. Yao, B. Benatallah, and M. Dong**, "A survey on expert recommendation in community question answering," *J. Comput. Sci. Technol.*, Vol. 33, pp. 625–653, 2018. doi:10.1007/s11390-018-1845-0

15 **R. Ren, H. Duan, W. Liu, and J. Liu**, "Aunet: An unsupervised method for answer reliability evaluation in community QA systems," in Asia-Pacific Web (APWeb) and Web-Age Information Management (WAIM) Joint International Conference on Web and Big Data, 2018, pp. 281–292. Springer.

16 **M. A. Suryanto, E. P. Lim, A. Sun, and R. H. Chiang**, "Quality-aware collaborative question answering: methods and evaluation," in Proceedings of the Second ACM International Conference on web Search and Data Mining, 2009, pp. 142–151. ACM.

17 **X. He, L. Wang, W. Zhang, and P. Zhang**, "Research on the quality prediction of online Chinese question answering community answers based on comments," in Proceedings of the 2nd International Conference on Big Data Technologies, 2019, pp. 114–120.

18 **B. Qu, G. Cong, C. Li, A. Sun, and H. Chen**, "An evaluation of classification models for question topic categorization," *J. Am. Soc. Inf. Sci. Technol.*, Vol. 63, pp. 889–903, 2012. doi:10.1002/asi.22611

19 **M. J. Blooma, A. Y. Chua, and D. H.-L. Goh**, "A predictive framework for retrieving the best answer," in Proceedings of the 2008 ACM Symposium on Applied Computing, 2008, pp. 1107–1111. ACM

20 **C. Chen, K. Wu, V. Srinivasan, and R. K. Bharadwaj**, "The best answers? think twice: identifying commercial campaigns in the CQA forums," *J. Comput. Sci. Technol.*, Vol. 30, pp. 810–828, 2015. doi:10.1007/s11390-015-1562-x

21 **H. Toba, Z.-Y. Ming, M. Adriani, and T.-S. Chua**, "Discovering high quality answers in community question answering archives using a hierarchy of classifiers," *Inf. Sci. (Ny)*, Vol. 261, pp. 101–115, 2014. doi:10.1016/j.ins.2013.10.030

22 **Y. Yao, H. Tong, T. Xie, L. Akoglu, F. Xu, and J. Lu**, "Detecting high quality posts in community question answering sites," *Inf. Sci. (Ny)*, Vol. 302, pp. 70–82, 2015. doi:10.1016/j.ins.2014.12.038

23 **E. Agichtein, C. Castillo, D. Donato, A. Gionis, and G. Mishne**, "Finding high quality content in social media," in Proceedings of the 2008 International Conference on web Search and Data Mining, 2008, pp. 183–194. ACM

24 **M. J. Blooma, A. Y.-K. Chua, and D. H.-L. Goh**, "Selection of the best answer in CQA services," in Information Technology: New Generations (ITNG), 2010 Seventh International Conference, 2010, pp. 534–539. IEEE

25 **L. Xu, J. Xiang, Y. Wang, and F. Ni**, "Data-driven approach for quality evaluation on knowledge sharing platform," *arXiv Preprint ArXiv:1903.00384*, 1–6, 2019.**[Q5]**

26 **B. M. John, A. Y.-K. Chua, and D. H.-L. Goh**, "What makes a high-quality user-generated answer?," *IEEE Internet Comput.*, Vol. 15, pp. 66–71, 2011. doi:10.1109/MIC.2011.23

27 **C. T. Lee, E. M. Rodrigues, G. Kazai, N. Milic-Frayling, and A. Ignjatovic**, "Model for voter scoring and best answer selection in community Q&A services. In Web Intelligence and intelligent agent technologies, 2009. WI-IAT'09," in IEEE/WIC/ACM International Joint Conferences on, 2009, pp. 116–123. IEEE volume 1.

28 **C. Shah, and J. Pomerantz**, "Evaluating and predicting answer quality in community QA," in Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2010, pp. 411–418. ACM.

29 **T. P. Sahu, N. K. Nagwani, and S. Verma**, "Selecting best answer: An empirical analysis on community question answering sites," *IEEE. Access.*, Vol. 4, pp. 4797–4808, 2016. doi:10.1109/ACCESS.2016.2600622

30 **M. J. Blooma, D. Hoe-LianGoh, and A. Yeow-Kuan Chua**, "Predictors of high-quality answers," *Online Inf. Rev.*, Vol. 36, pp. 383–400, 2012. doi:10.1108/14684521211241413

31 **J. Lou, Y. Fang, K. H. Lim, and J. Z. Peng**, "Contributing high quantity and quality knowledge to online CQA communities," *J. Am. Soc. Inf. Sci. Technol.*, Vol. 64, pp. 356–371, 2013. doi:10.1002/asi.22750

32 **P. K. Roy, Z. Ahmad, J. P. Singh, M. A. A. Alryalat, N. P. Rana, and Y. K. Dwivedi**, "Finding and ranking high quality answers in community question answering sites," *Global Journal of Flexible Systems Management*, Vol. 19, pp. 53–68, 2018a. doi:10.1007/s40171-017-0172-6

33 **D. Hoogeveen, A. Bennett, Y. Li, K. M. Verspoor, and T. Baldwin**, "Detecting mis-flagged duplicate questions in community question answering archives," in Tvwelfth International AAAI Conference on Web and Social Media, 2018, pp. 112–120.

34 **O. Kucuktunc, B. B. Cambazoglu, I. Weber, and H. Ferhatosmanoglu**, "A large-scale sentiment analysis for yahoo! answers," in Proceedings of the Fifth ACM International Conference on Web Search and Data Mining, 2012, pp. 633–642.

35 **M. Li, Y. Li, Q. Peng, and J. Wang**, "A hybrid MCDM model combining DANP with TODIM to evaluate the information

quality of community question answering in a two-dimensional linguistic environment," *Expert Syst.*, Vol. 38, no. 2, pp. e12619, 2021.

36 D. Elalfy, W. Gad, and R. Ismail, "A hybrid model to predict best answers in question answering communities," *Egyptian Informatics Journal*, Vol. 19, no. 1, pp. 21–31, 2018. doi:10.1016/j.eij.2017.06.002

37 Y. Zhang, M. Zhang, N. Luo, Y. Wang, and T. Niu, "Understanding the formation mechanism of high-quality knowledge in social question and answer communities: A knowledge co-creation perspective," *Int. J. Inf. Manage.*, Vol. 48, pp. 72–84, 2019. doi:10.1016/j.ijinfomgt.2019.01.022

38 D. Palomera, and A. Figueroa, "Leveraging linguistic traits and semi-supervised learning to single out informational content across how-to community question-answering archives," *Inf. Sci. (Ny)*, Vol. 381, pp. 20–32, 2017. doi:10.1016/j.ins.2016.11.006

39 J. E. Van Engelen, and H. H. Hoos, "A survey on semi-supervised learning," *Mach. Learn.*, 1–68, 2019.**[Q6]**

40 H. Fu, and S. Oh, "Quality assessment of answers with user-identified criteria and data-driven features in social Q&A," *Inf. Process. Manag.*, Vol. 56, no. 1, pp. 14–28, 2019. doi:10.1016/j.ipm.2018.08.007

41 A. Tang, P. Ren, and Z. Sun, "Multi-feature based question–answerer model matching for predicting response time in CQA," *Knowl. Based. Syst.*, Vol. 182, pp. 104794, 2019. doi:10.1016/j.knosys.2019.06.002

42 O. Chergui, A. Begdouri, and D. Groux-Leclet, "Integrating a Bayesian semantic similarity approach into CBR for knowledge reuse in community question answering," *Knowl. Based. Syst.*, Vol. 185, pp. 104919, 2019. doi:10.1016/j.knosys.2019.104919

43 Q. Tian, P. Zhang, and B. Li, "Towards predicting the best answers in community-based question-answering services," in ICWSM, 2013, pp. 725–728.

44 I. Rish, "An empirical study of the naive Bayes classifier," in IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence, 2001, pp. 41–46. IBM volume 3.

45 J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Ann. Stat.*, 1189–1232, 2001.**[Q7]**

46 L. Breiman, "Random forests," *Mach. Learn.*, Vol. 45, pp. 5–32, 2001. doi:10.1023/A:1010933404324

47 N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "Smote: synthetic minority over-sampling technique," *J. Artif. Intell. Res.*, Vol. 16, pp. 321–357, 2002. doi:10.1613/jair.953

48 H. He, Y. Bai, E. A. Garcia, and S. Li, "ADASYN: Adaptive synthetic sampling approach for imbalanced learning," in 2008 IEEE International Joint Conference on Neural Networks (IEEE World Congress on Computational Intelligence), 2008, pp. 1322–1328. IEEE.

49 J. Davis, and M. Goadrich, "The relationship between precision-recall and roc curves," in Proceedings of the 23rd International Conference on Machine Learning ICML '06, 2006, pp. 233–240. ACM

50 L. Ponzanelli, A. Mocci, A. Bacchelli, M. Lanza, and D. Fullerton, "Improving low quality stack overflow post detection," in Software Maintenance and Evolution (ICSME), 2014 IEEE International Conference on, 2014, pp. 541–544. IEEE.

51 **X.-Q. Bao, and Y.-F. Wu**, "A tensor neural network with layer wise pre-training: Towards effective answer retrieval," *J. Comput. Sci. Technol.*, Vol. 31, pp. 1151–1160, 2016**[Q8]**. doi:10.1007/s11390-016-1689-4

52**M. Debruyne, S. Höppner, and S. Serneels**, "Outlyingness: which variables contribute most?," *Stat. Comput.*, Vol. 29, pp. 707–723, 2019**[Q9]**. doi:10.1007/s11222-018-9831-5

53**M. Khabsa, A. Elmagarmid, I. Ilyas, H. Hammady, and M. Ouzzani**, "Learning to identify relevant studies for systematic reviews using random forest and external information," *Mach. Learn.*, Vol. 102, no. 3, pp. 465–482, 2016**[Q10]**. doi:10.1007/s10994-015-5535-7

54**H. Langseth, and T. D. Nielsen**, "Classification using hierarchical naive Bayes models," *Mach. Learn.*, Vol. 63, no. 2, pp. 135–159, 2006**[Q11]**. doi:10.1007/s10994-006-6136-2

~~55 T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in Advances in Neural Information Processing Systems, 2013, pp. 3111–3119**[Q12]**.~~

55**C. R. Stephens, H. F. Huerta, and A. R. Linares**, "When is the Naive Bayes approximation not so naive?," *Mach. Learn.*, Vol. 107, no. 2, pp. 397–441, 2018**[Q13]**. doi:10.1007/s10994-017-5658-0

Pradeep Kumar Roy is currently an Assistant Professor with the Department of Computer Science and Engineering, Indian

Institute of Information Technology (IIIT), Surat. He received his Ph.D. in Computer Science and Engineering from the National Institute of Technology Patna, in 2018. His area of specialisation straddles across question answering, text mining and information retrieval, and wireless sensor networks. He has published articles in different Journals, including IJIM, Neural Processing Letters, Neural Computing and Applications, Future Generation Computer Systems, and others. He has also published the conference proceedings in various international conferences. **E-mail:** pkroynitp@gmail.com

Zishan Ahmad received his B.Tech. degree in Computer Science and Engineering in 2013. He received M.Tech. in Information Technology from National Institute of Technology Patna in 2017. Currently he is working as a PhD Research Scholar in the department of Computer Science and Engineering, Indian Institute of Technology (IIT), Patna. His current research area is in text mining and data analytics, social analytics. **E-mail:** zishan.itpg@nitp.ac.in

Jyoti Prakash Singh is an assistant professor in the department of Computer science and Engineering in National Institute of Technology Patna. He has co-authored seven books in the area of C programming, Data Structures, Operating systems and Ad Hoc Networks. Apart from this, he has around 25 international journal publications and more than 40 international conference proceedings. His research interests include text mining, deep learning, social network and information security. He is associate Editor of International Journal of Electronic Government Research (IJEGR). **Corresponding author. E-mail:** jyotip.singh@gmail.com

Snehasish Banerjee is a Lecturer at the York Management School in the University of York. He holds a PhD from Nanyang Technological University. His area of specialisation straddles across information science and digital marketing. His works have appeared in outlets such as Computers in Human Behaviour, Internet Research, Journal of the Association for Information Science and Technology, Online Information Review, and Tourism Management. **E-mail:** snehasish.banerjee@york.ac.uk