Maximizing Post Efficiency in Stack Overflow

Luís F. Correia Cleto
Delft University of Technology
l.f.correiacleto@student.tudelft.nl

Tiago Almeida Fernandes
Delft University of Technology
t.almeidafernandes@student.tudelft.nl

***Abstract*—Software developers frequently require access to new information for their work, either for being inexperienced or for requiring expert knowledge to handle a specific task outside their usual field of work. As such, Questions and Answers (Q&A) platforms have become valuable information venues for developers. One of the most prominent Q&A platforms that was built specifically for programmers is Stack Overflow, which counts with over 5 million users, who have created over 11 million discussions and 18 million answers. However, more than 1.2 million questions posted on this platform remain unanswered and over 2.9 million have no accepted answer. To efficiently retrieve the knowledge they seek, it is necessary for users of Stack Overflow to ask questions that receive good answers and, whenever possible, avoid creating posts with features that may harm the chances of the question being successfully answered.**

In this paper we present the findings of our study which aimed at discerning the most relevant traits of a question for determining the number of answers it will get and the quality of those answers. In addition we also analyzed similar traits for answers to determine how they might influence its quality or the chances of it being accepted by the author of a question. Finally, we built and evaluated prediction models for each of the traits we analyzed. Our findings show that there are indeed some characteristics of posts that are related to their rates of success and that they are not the same for when asking a question or replying to one.

I. Introduction

Software development is a knowledge-intensive activity [1] where knowledge is often distributed among several individuals with different areas of expertise. This implies that efficient knowledge management can be greatly important for software developers.

The widespread usage of social media and the Internet has originated in a rise of Q&A platforms such as Yahoo! Answers[1] or Answers[2] as a mean for exchanging knowledge. In these platforms users are encouraged to participate either by asking questions or answering them. The rewards for participating are usually based on some kind of point or karma system that rewards good answers and questions. While Q&A platforms enables quick and easy access to knowledge in areas unknown to the user, the quality of posts on these platforms can also vary widely from high quality posts made by experts to low quality or even misinforming posts, making quality detection for posts in Q&A websites an increasingly relevant area of research. For the domain of Software Development, Stack Overflow[3] has emerged as one of the dominant Q&A platforms where programmers can exchange knowledge between themselves. Since its foundation in August of 2008, over 11 million discussions have been started on Stack Overflow [2] along with around 18 million solutions.

The increase in usage of Q&A websites has brought a lot of new members as well as questions being asked. Similarly, the number of unanswered questions is also considerable. Stack Overflow alone had over 1.2 million unanswered questions on the 7th of January of 2016. If you take into account the number of questions with no accepted answers, this number grows to over 2.9 million. This could be due to several factors, such as the questions being uninteresting, poorly formulated, already answered, too difficult or other motives. The main purpose of this paper is to focus on identifying what relates with the number of answers that a certain Stack Overflow question will get and provide an analysis on how to create possibly high-quality posts on this platform. Therefore, we investigate the following research questions:

RQ1 *What is related to the number of answers to a question on Stack Overflow?* With the study described in this paper we intend to determine what characteristics of a question are related to the number of replies that it receives. These characteristics can be quantitative (code/natural text ratio, length of title, number of tags, presence of certain popular tags, ...) or qualitative (easiness of a question, type of question [3], topic, ...). We originally expected that at least for certain topics the amount of replies will be greater than for other less popular topics.

RQ2 *Which questions are more likely to receive high quality answers?* As the number of answers alone might simply indicate a controversial topic, it is also important to measure the quality of the answers that have been given since the intent of a user when they post a question on Stack Overflow is to receive a high quality answer rather than several useless answers. For this case, we expect the easiness of a question to be highly related.

RQ3 *What are the characteristics of an answer that relate to higher chances of it being accepted?* Out of all the answers that are provided, only one gets accepted as the solution. Knowing which features are related with higher chances of being accepted can contribute to increasing the effectiveness of answer posts on Stack Overflow. One possible factor for determining this is if the answer was

[1]http://answers.yahoo.com
[2]http://www.answers.com/
[3]http://stackoverflow.com/

the first one to be posted.

Our findings will enable us to make recommendations on how practitioners in the domain of software development can leverage the knowledge and use Q&A websites effectively to provide high quality topics and similarly receive appropriate answers.

The remainder of this paper will be structured as follows: Section II focuses on the background of our research, including related work. Section III describes our approach to the problem at hand as well as the dataset and API that were used during the study. In section IV we present and analyze our findings, section V covers some of the limitation of this study and section VI outlines our conclusions and contains recommendations for future work.

II. Background and Related work

Several studies have been undertaken to attempt to correlate features of a post in a Q&A platform with its quality. Many of these focus specifically on Stack Overflow and how to detect or improve the quality of a post on that platform.

Treude *et al.* undertook a qualitative approach to identifying and categorizing Stack Overflow questions. By manually analyzing a small sample of a few hundred discussions, they were able to identify 10 types of questions and determine that questions of type **review**, **conceptual**, **how-to** and **novice** were the types that got replied to the most often [3]. This might be due to the easiness of replying to **novice** questions, since they tend to be quite simple, as well as to *review* questions which mostly contain code fragments and are very concrete, often not requiring any external knowledge for the code to be understood. However, they also concluded that there are several other factors that influence the amount of replies a question can get, such as the length of the question, the presence of code snippets, the technology in question and even the day and time when it was posted.

On the topic of determining the quality of an answer, Hart and Sarma conducted a study to understand how novice users perceive the quality of answers on Stack Overflow [4]. By conducting a survey with 34 novices they found out that: *1*) social factors (like answerer reputation) have little impact on perceptions of answer quality; *2*) answer length is important insofar as longer answers tend to be more thorough; *3*) presentation is important - both code and prose are essential elements of a *high quality* answer. This contrasts with findings in other studies that give a higher relevance to popularity based metrics [5].

Ponzanelli *et al.* [6], by trying to decrease the size of Stack Overflow's review queue, added to the current metrics employed by Stack Overflow readability metrics and popularity metrics. They constructed a linear quality function for each metric and learned each one using genetic algorithms. The authors concluded that popularity metrics and readability metrics are the most useful metric sets to refine the low quality review queue. This suggests that these metrics might be the most adequate for distinguishing *low quality questions* from the rest.

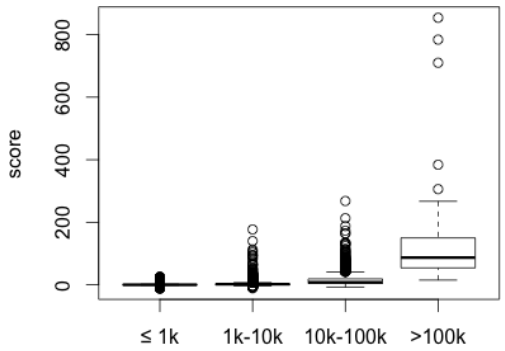

Fig. 1. Distribution of the score by the number of visualizations

In the domain of quality classification and prediction, Ponzanelli *et al.* [5] also devised and evaluated several metrics grouped into three different categories: Stack Overflow specific, readability, and popularity metrics. They then classified questions as 'very good', 'good', 'bad' or 'very bad'. They reported a precision of their methodology is approximately 80%. We have attempted to incorporate as many of the metrics described in their work as possible so as to further extend our analysis. Unfortunately, many of the popularity related metrics used for that study could not be retrieved from the StORMeD data set and were therefore not included in our analysis.

Daniele Romano and Martin Pinzger [7] came up with a methodology to improve Q&A Sites' voting systems by using weighted votes. The idea behind the weighted votes is to emphasize answers that receive the majority of votes when most of the answers were already posted. This solution tries to mitigate the factor that more recent answers tend to receive less votes even if their content may be better. The weighted votes are a measure based on the number of answers a question has at the point when the vote was done. Thus, a vote is more valuable when there are more answers. Unfortunately, we do not have the knowledge of when each vote was issued in our data set so it is not possible for us to also measure scores in that way.

III. Methodology

In order to conduct our research, we will resort to the StORMeD data set and API [8]. It contains approximately 900,000 questions relating to the Java language and their respective answers and comments. Each question and answer, besides the usual metadata, has its body represented in a *heterogeneous abstract syntax tree* (H-AST) that encapsulates the content in structured fragments such as Java constructs, stack traces, XML/HTML fragments, JSON fragments or text. In addition, it contains metainformation for common analysis like term frequency data, variable names, and mentioned reference types.

Posts on Stack Overflow (both questions and answers) have a user-determined score which serves as a direct measurement of quality as it is perceived by the community, mitigating the problem of subjectiveness when a single individual is to determine the quality of this type of content. Something

to take into account when using this quality measurement is that it can be affected by the popularity of the discussion: very popular questions will generally have higher scores than unpopular ones (see Figure 1), which may be high quality as well but not interesting for as many people. One approach we use to mitigate the popularity effect on quality is to stratify the discussions by popularity (e.g. compare questions with a similar number of views). When analyzing the quality of answers we also found that the score of an answer is highly correlated to the score of its question (with a coefficient of approximately 0.883). For this reason we chose to also analyze the quality of an answer as the ratio between its score and the score of the corresponding question. Since scores can also be 0 or even negative, before calculating the ratios we shifted all scores by the absolute value of the lowest negative score, incremented by 1 to avoid null values.

As it was necessary to measure the popularity of topics (called tags in the Stack Overflow context), we resorted to the Stack Exchange Online API[4] where the usage information for each tag can be immediately retrieved in JSON[5] format. Additionally, we also wanted to measure the popularity on GitHub of APIs tagged in the question. For this we resorted to data provided by A. Sawant [9].

We developed a framework in Scala to conduct this study. It interacts with the StORMeD Devkit allowing the extraction of the specific variables present in the data set needed for our analysis. As there is a considerable amount of data present in the provided dataset, we retrieve and process sample data. This data collection can be divided in the following steps:

1) **Extract a sample** - randomly extract discussions from the original data set.
2) **Obtain data from Stack Exchange** - download extra data from Stack Exchange API.
3) **Compute features** - extract or compute the desired features from the data.
4) **Dump data to a CSV** - save the resulting data to a CSV file, ready to be analyzed.

Firstly, a random sample of 50.000 discussions, which represent about 5% of the total dataset, with a total of 91.591 answers was retrieved from the original StORMeD data set for performing the analysis.

For **RQ1** and **RQ2** the same CSV can provide the necessary data. To answer **RQ3** a different CSV file must be computed with different features as this research question focus on the answers' content rather than the questions' content.

The features for **RQ1** and **RQ2** are divided in two: questions' features (*QF*), which are features extracted from the question itself, and questions' classification (*QC*), which are features that allow to classify the question quality. Analogously, for **RQ3** there are: answers' features (*AF*) and answers' classification (*AC*). The extracted metrics are listed and described in Table I.

[4]https://api.stackexchange.com/

[5]JavaScript Object Notation

TABLE I
METRICS' LIST

	Metric	Details
QF	Title Length	number of characters in question's title
	Tags count	number of tags associated with question
	Max(tag popularity)	Highest popularity of a question's tags
	Avg(tag popularity)	Average popularity of the question's tags
	Min(tag popularity)	Minimum popularity of a question's tags
	Max(API popularity)	Highest (github) popularity of a tagged API
	Avg(API popularity)	Average (github) popularity of a tagged API
	Min(API popularity)	Minimum (github) popularity of a tagged API
AF	first posted	0 or 1 - 1 if was first answer
	same day	0 or 1 - 1 if posted in question's day
QF & AF	Code %	Total code lines percentage
	Java %	Java's lines percentage
	JSON %	JSON's lines percentage
	XML %	XML's lines percentage
	Stack traces %	Stack traces' lines percentage
	Length	number of characters
	Words Count	number of words (text only)
	text speak count	number of text speak ('afaik', 'wat', ...)
	urls count	number of urls in the text
	Coleman-Liau Index	$0.0588L - 0.296S - 15.8$ Where L is the average number of letters per 100 words and S is the average number of sentences per 100 words.
	Flesch Reading Ease Score	$206.835 - 1.015 * \frac{words}{sentences} - 84.6 * \frac{syllables}{words}$
	Flesch-Kincaid Grade level	$0.39 * \frac{words}{sentences} + 11.8 * \frac{syllables}{words} - 15.59$
	Automated Readability Index	$4.71 * \frac{characters}{words} + 0.5 * \frac{words}{sentences} - 21.43$
	Gunning Fog Index	$0.4 * [\frac{words}{sentences} + 100 * \frac{complex\ words}{words}]$
	SMOG Grade	$1.043 * \sqrt{polysyllables * \frac{30}{sentences}} + 3.1291$
	Day of Week	the day of the week (Monday-Sunday)
	Reputation	Author's reputation
	Intercalations	number of text and code intercalations
	Days since posted	days since the question was posted
QC	Score	question's score
	Answers count	number of answers in discussion
	Max(score)	max score among answers
	Avg(score)	average score among answers
	Min(score)	min score among answers
	Comments count	number of comments to question
	Max(length)	answers' max length
	Avg(length)	answers' average length
	Min(length)	answers' min length
	Has Accepted Answer	0 or 1 - 1 if there is an accepted answer
	View count	number of question's views
	Score Ratio	$\frac{Max(answer\ score)}{score}$
AC	Score	answer's score
	Comments count	number of comments to answer
	Accepted	0 or 1 - 1 if the accepted answer
	Max(length)	comments' max length
	Avg(length)	comments' average length
	Min(length)	comments' min length

After generating the CSV files with the desired information we began by running an analysis on both the questions' data and the answers' data to identify the relevant metrics to be analyzed further. For **RQ1** we focused on the relations with the number of answers a question receives, while for **RQ2** we focused on the quality of those answers instead and also took into account answer specific metrics that may influence its quality in order to allow a Stack Overflow user to better understand how to judge the quality of the answers they receive. To mitigate popularity bias when trying to answer both **RQ1** and **RQ2** we ran the same analysis again after filtering the sample for discussions with similar popularity values and compared the results. For **RQ2**, analyzing the aforementioned score ratio rather than simply the score also mitigates the popularity bias. **RQ3** seems less likely to be affected by popularity, but we did repeat the analysis for that question taking into account only posts with more than one answer since having only one answer would necessarily imply the first answer to be posted would be the accepted one. We also confirmed our results by ignoring posts with no accepted answers as these do not contain enough information to discern what traits lead an answer to be accepted.

As the *Day of the Week* is our single non-numeric feature, we separately analyzed how this variable might influence the quality of both questions and answers.

After completing the analysis we searched for potential pairwise correlations between features, deleting the ones with the largest mean absolute correlation. We then constructed prediction models [10]–[13] for each of the dependent variables targeted by our research questions. For **RQ1** and **RQ2** two techniques were used: Random Forest and Multiple Linear Regression. For **RQ3** we chose a different approach as we were dealing with a binary categorical variable: An answer is either accepted or it is not. As such, we resorted to the following techniques: Random Forest, Naive Bayes and Logistic Regression. In order to construct our prediction models we split data in training and testing sets and balanced the data set. The prediction models were trained and tested 10 times with different random training and testing sets. The results obtained are presented in Section IV.

IV. Results

In this section we present the results addressing each one of the research questions proposed.

A. ***RQ1**: What is related to the number of answers to a question on Stack Overflow?*

1) Analysis: Our initial hypothesis was that the higher the topic popularity was, the greater the amount of answers would be. However, this correlation did not prove to be as strong as expected. We found that the popularity of the tags did not influence the popularity of a question (i.e. the number of views the question gets), having a correlation coefficient of under 0.02. The results presented in Table II show that only

TABLE II
Highest answers' count correlation factors

	Answers Count	Answers Count (views 100-150)
View count	0.36	0.01
Reputation	0.17	0.08
Max(tag popularity)	-0.15	-0.09
Avg(tag popularity)	-0.07	0.20
Min(tag popularity)	0.12	0.21
Length	-0.15	-0.15
Score	0.24	0.06
Has Accepted Answer	0.31	0.33

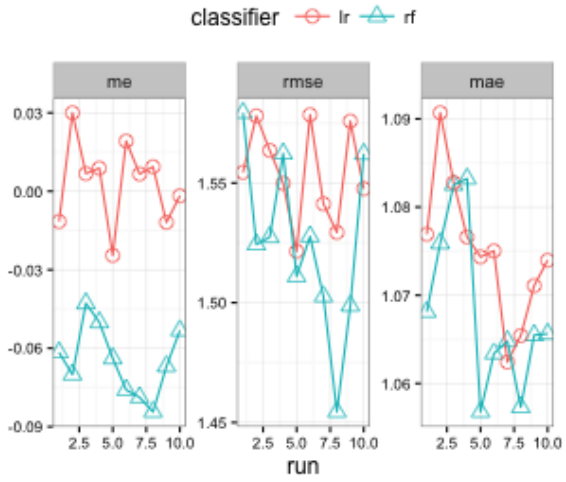

Fig. 2. Error of the prediction models (Multiple Linear Regression and Random Forest) for RQ1

by segmenting the questions with similar popularity (number of views) the tag popularity becomes relevant.

The lowest tag popularity value does influence the number of answers a question gets, even more so when the sample is filtered for questions with similar popularity. This is likely due to this value representing the specificity of a topic more accurately than the remaining tags. A low minimum value would indicate a highly specialized topic which might be difficult for an average user to reply to.

On the other hand, what is clearly correlated with the amount of answers a question gets is its number of views, as would be expected. If there are a lot of people viewing the discussion then it is more likely that some of them will post an answer.

The length of a question negatively correlates with the number of answers it receives. One possible explanation for this is that longer questions tend to be more complex and require greater understanding from a reader in order to enable them to write an answer. Both this and the specificity of the topic increase the difficulty of a question, strengthening the hypothesis that easier questions get the most answers.

Additionally, from the table one can also see that questions

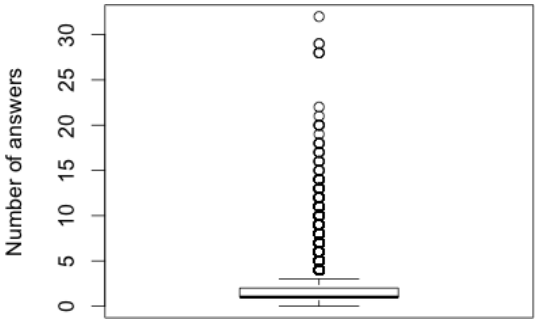

Fig. 3. Distribution of the number of answers per question

TABLE III
TOP 5 INCREASE IN MSE FOR NUMBER OF ANSWERS (RQ1)

Metric	Increase in MSE (%)
Gunning Fog Index	19.59
Java %	16.60
Reputation	16.37
Coleman Liau Index	14.77
Intercalations	14.37

TABLE IV
HIGH QUALITY ANSWERS' CORRELATION FACTORS

	Max(score)	Avg(score)	Min(score)	Ratio
View count	0.44	0.40	0.14	0.11
Reputation	0.32	0.32	0.18	-0.01
Length	-0.18	-0.16	-0.04	-0.08
Min(tag)	0.00	-0.01	-0.03	0.13
Answers Count	0.53	0.36	-0.14	0.14

TABLE V
HIGH QUALITY ANSWERS' CORRELATION FACTORS (100-150 VIEWS)

	Max(score)	Avg(score)	Min(score)
Tags count	-0.19	-0.15	-0.02
length	-0.19	-0.16	-0.06
Words Count	-0.14	-0.12	-0.04
Text speak Count	-0.11	-0.09	-0.03
Reputation	0.22	0.23	0.17
Score	0.23	0.23	0.14
Answers count	0.52	0.35	-0.08
Has accepted answer	0.41	0.40	0.26

with more answers tend to have a higher score. This correlation becomes a lot weaker once a popularity filter is used, which would be expected since the number of views of a question and its score have a high correlation factor of about 0.63. Finally, the results also show that the more the answers a question gets, the more likely it is that its author will find one to be satisfactory and mark it as an accepted answer.

2) Prediction Model: As we are attempting to predict a numerical variable, we built prediction models using multiple linear regression and random forest. In Figure 2 are presented the mean prediction errors for 10 evaluations of our models using random training and testing data. It can be seen that both models perform similarly, with multiple linear regression being less biased while random forest has a slightly lower error. While the prediction error does not appear to be very high, it is important to take into account that 95% of the questions have less than 4 answers (see Figure 3).

By looking at Table III we see that readability metrics (readability indexes and presence of code) have the highest impact in our prediction models along with the reputation of the author.

*B. **RQ2**: Which questions are more likely to receive high quality answers?*

1) Analysis: Quality is an abstract and highly subjective unit of measure. As such, we have decided to classify questions and answers' quality accordingly to the their community-decided score (points given by the users). Additionally, as we found that the scores of answers are highly correlated with the scores of the corresponding questions, we also calculated quality as the ratio between answers' scores and those of their questions. For this research question we attempted to find correlations between traits of a question and the quality of the answers it gets.

What seems to relate to the quality of answers the most is the popularity of the question (refer to Table IV), once again showing that a post's score is heavily biased by the popularity of the corresponding discussions. By analyzing the score ratio we can mitigate this bias and the relation between the specificity of a topic and the quality of answer becomes more apparent as the popularity of its most specific tag is shown to be correlated with the quality of the answers.

In Table V we present our results again after applying the same popularity filter as in our analysis for **RQ1**. By looking at the data now we see that the reputation of the question's author has the strongest correlation with the score of its answers. This makes sense as users with higher reputation are associated with higher quality content and high quality questions should get higher quality answers. Once again, the length of a question will negatively impact the quality of its answers. The same can be said for the number of tags. The increase in the number of tags can be justified by an increase in the specificity of the topic. As suggested in our analysis to **RQ1** both the length and specificity of a question might increase its difficulty, thus strengthening our original hypothesis that easier questions are more likely to receive high quality answers.

One more observation that can be made from this data is that the use of text speak vocabulary (e.g. 'wat', 'afaik', 'doesnt') in the question's body tends to have a negative impact on the quality of its answers. This shows that the way a question is written might influence how good the answers are (or perhaps who makes them).

In addition to the previous results, other interesting ob-

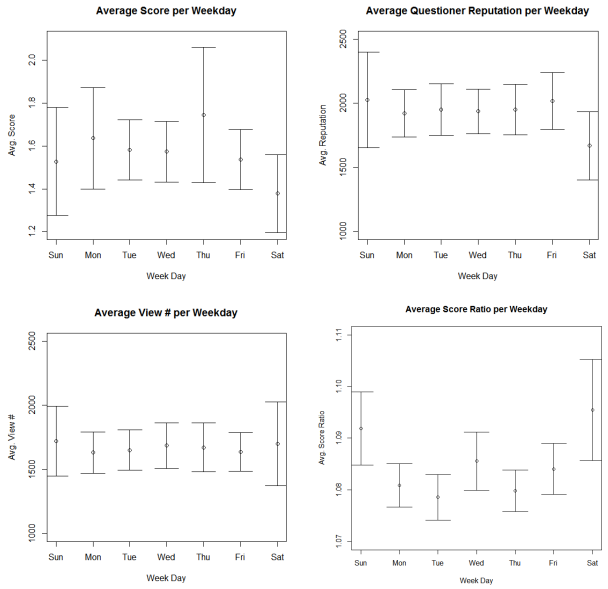

Fig. 4. Means and respective 95% confidence intervals for a question's score, author's reputation, views and answer/question score ratio per weekday when the question was posted.

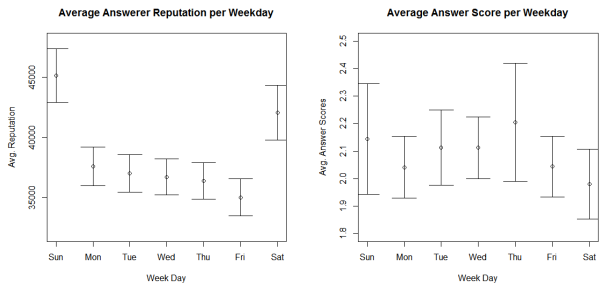

Fig. 5. Means and respective 95% confidence intervals for an answer's author's reputation and score per weekday when the answer was posted.

servations can be made from analyzing the weekdays when questions and answers are created. Figure 4 shows that questions asked on a Saturday have a lower average score than the global average. This decrease of nearly 15% is accompanied by similar drop in the average reputation of questions' authors. Taking into account that weekends see an almost 50% decrease in the number of questions asked. One possible explanation for this is that less questions are being asked by professional programmers (who work mostly on weekdays), making low quality questions asked by students or hobbyists more noticeable. However, the popularity and the quality of the answers received are not noticeably affected.

Conversely, when looking at the reputation of the answers' authors (see Figure 5), weekends see a significant increase: 12% above the global average for Saturdays and 20% for Sundays. Following a similar line of though as before, this could be because professional programmers are more available for answering questions on weekends than on weekdays. It is worth noting that while the reputation of

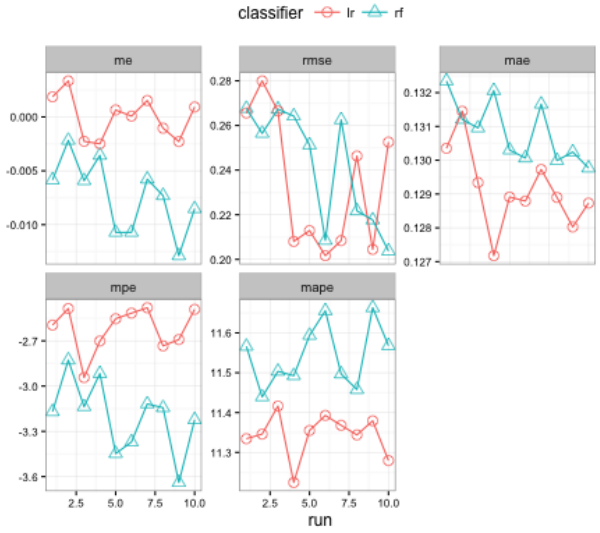

Fig. 6. Error of the prediction models (Linear Regression and Random Forest) for RQ2

TABLE VI
TOP 5 INCREASE IN MSE FOR ANSWERS' QUALITY (RQ2)

Metric	Increase in MSE (%)
Text Speak Count	10.54
length	9.35
Min(tag popularity)	8.81
Coleman Liau Index	8.78
Avg(tag popularity)	8.55

authors' answers sees a significant increase on weekends, the same is not verified for the answers' scores. This could be due to the decrease on the quality of the questions posted on those days. While the answers posted on weekends could be for questions created on weekdays, we've concluded that replying quickly to a question is important for both an answer's score and its chances of being marked as accepted (see Section IV-C).

2) Prediction Model: We built prediction models for the answer/question score ratios using multiple linear regression and random forest. In Figure 6 are presented the mean prediction errors for 10 evaluations of our models using random training and testing data. Once again, both models perform similarly, with multiple linear regression being less biased.

Table VI shows the importance of each variable extracted from the random forest model. The results agree with those of the correlation analysis, showing that the presence of text speak is an important factor to take into account when predicting the quality of the answers a question will get, along with a question's length and topic specificity.

TABLE VII
ANSWER MORE LIKELY TO BE ACCEPTED

	Accepted	Accepted (>1 answers)
First posted	0.24	0.24
Length	0.14	0.13
Score	0.30	0.30
Comments count	0.23	0.19

TABLE VIII
TOP 5 DECREASE IN ACCURACY (RQ3)

Metric	Increase in MSE (%)
First posted	54.52
Reputation	24.02
Length	11.70
Same day	10.71
Flesch-Kincaid Grade level	9.51

C. ***RQ3**: What are the characteristics of an answer that relate to higher chances of it being accepted?*

1) Analysis: Our original prediction for this question was that being the first answer to the question may make it more likely to be the accepted one. Our results corroborate this hypothesis. We initially calculated the correlation for the whole sample and then again for only questions with more than 1 answer (since for questions with only one answer, the accepted answer will obviously be the first) and the correlation coefficient remains steady at 0.24 for both samples (refer to Table IV-C). This correlation is also maintained if we filter the sample for answers to questions that have an accepted answer. Opposite to the trend seen for questions, users tend to prefer longer answers, attributing these with a higher score. This might be because longer answers tend to be more thorough [4] and usually may contain some code to support the text.

Being an accepted answer is also positively correlated with the score and number of comments it receives. The number of comments is also correlated with whether an answer was posted on the same day as the question or not with a coefficient of 0.15. A possible explanation for these values is that the author of the question discusses the answers he gets until he finally accepts an answer. This suggests that discussion may lead to improvements on answers, making them more likely to be accepted.

Once an answer has been accepted there is little interest left in discussing more answers. As the accepted answer is the one that better answers the question, and the one users immediately see when they are dealing with a similar question, it is no surprise it should have a higher score than remaining answers. However, this is not always the case as some users may post a more complete answer after another has already been accepted, achieving a higher score, or an author might select a worse answer as the accepted one if it gives him an easy solution he can directly use to solve his problem.

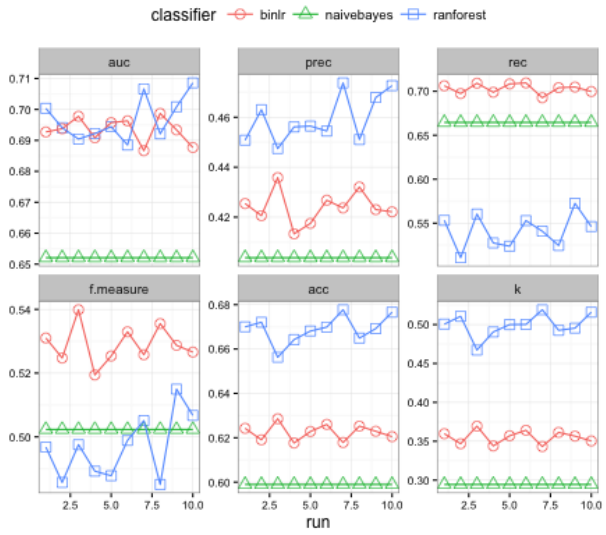

Fig. 7. Performance of the prediction models (Logistic Regression, Naive Bayes and Random Forest) for RQ3

2) Prediction Model: For this scenario we built three prediction models using Logistic Regression, Random Forest and Naive Bayes. Figure 7 details the performance results of 10 evaluations of our prediction models with random raining and test data. The accuracy obtained with the models varies between 60 and 68%, with Random Forest outperforming the other two techniques. Despite Random Forest reporting the highest accuracy and precision it has a much lower recall than the other two, resulting in a lower F-measure. As a result, Logistic Regression presents itself as the most balanced technique to predict whether an answer will be accepted or not.

Table VIII shows the importance of each variable extracted from the random forest model. The results agree with those of the correlation analysis, showing that being the first answer posted indeed relates to higher chance of it being accepted. The reputation of the author, the length and quickness of the answer are also factors worth taking into account.

V. LIMITATIONS

While we worked with a relatively large data set and ensured that all correlations we present are statistically significant (we measured all respective p-values to be considerably lower than 0.05), it is worth taking into account that we worked exclusively with Stack Overflow questions. For this reason, results may vary when applied to discussions from other platforms. Additionally, the StORMeD data set only contains discussions tagged with ”java”, as such the results may vary even within Stack Overflow if analyzing discussions for other topics.

Another limitation of this study is that very few popularity metrics were available in the data set, leaving our analysis for popularity based metrics (such as answer acceptance rate, approved edit suggestion, badges, ...) rather small. The reputation of posts’ authors at the time of the posts’ creation is

also not available, limiting the conclusions that can be drawn from correlations between authors' reputation and quality determining metrics.

VI. Conclusion

Understanding how to manage knowledge efficiently is crucial for developers. Knowing how to maximize this efficiency when using Q&A websites, such as Stack Overflow, can be a great benefit as it may allow programmers to quickly receive answers to their questions as well as improve the quality of the answers they may receive. On the other hand it may also help those providing the answers in creating higher quality material.

We elaborated a list of metrics for both questions and answers for analyzing and classifying their quality. Using the StORMeD scala API and their data set, as well as the Stack Exchange API, we extracted a data set with the metrics we wanted to study for each question and its answers. We then analyzed the data for potential correlations between features and quality, taking into account the bias caused by popularity on the community-perceived quality. We were also able to build prediction models for the relevant quality metrics.

By analyzing our results we found that length has an opposite effect on questions and on answers. Users tend to value shorter questions more than longer ones, while the opposite is true for answers. As the specificity of a question's topic also negatively impacts its answer rate and answer quality, questions should be kept as short and general as possible while, on the other hand, answers should be thorough. Additionally, being the first answer to be posted is highly related to being the accepted answer.

We found that on weekends the reputation of answers' authors tends to be higher while that of the questions' authors decreases significantly (along with the quality of the questions). Suggesting professional programmers and experts ask more questions during week days but are more available to answer them on weekends. We also concluded that the quickness of a reply is highly valued as the first answers to be posted tend to be the accepted ones and have higher scores.

Additionally, we confirmed that popularity and reputation go a long way in contributing to both creating high quality questions and receiving a high amount of good quality answers. Lastly, while it does not seem like readability metrics have a large impact on the number of answers or the popularity of a question, the use of text speak in a question is negatively correlated with the quality of the answers it receives.

A. Future Work

A valuable extension to our research would be to incorporate more popularity metrics from Stack Overflow (e.g. answer acceptance rate, approved edit suggestions, badges, ...). It would also be interesting to analyze other topics besides ones tagged with "java" or on other platforms besides Stack Overflow in order to strengthen or improve the hypotheses presented. It would also be interesting for the popularity metrics to be recorded at the time a post was created as these are likely to change over time. It might also be interesting to repeat the study using a different measurement for post quality that ignores the bias induced by popularity altogether.

Incorporating the concept of weighted votes proposed by Daniele Romano and Martin Pinzger [7] might also lead to new insights when analyzing answer quality as it mitigates the effect of late answers getting less votes even when they are high quality.

References

[1] P. N. Robillard, "The role of knowledge in software development," *Commun. ACM*, vol. 42, no. 1, pp. 87–92, Jan. 1999.

[2] StackOverflow, "Stackoverflow," http://stackoverflow.com/10m, online December 2015.

[3] C. Treude, O. Barzilay, and M.-A. Storey, "How do programmers ask and answer questions on the web? (nier track)," in *Proceedings of the 33rd International Conference on Software Engineering*, ser. ICSE '11. New York, NY, USA: ACM, 2011, pp. 804–807.

[4] K. Hart and A. Sarma, "Perceptions of answer quality in an online technical question and answer forum," in *Proceedings of the 7th International Workshop on Cooperative and Human Aspects of Software Engineering*, ser. CHASE 2014. New York, NY, USA: ACM, 2014, pp. 103–106.

[5] L. Ponzanelli, A. Mocci, A. Bacchelli, and M. Lanza, "Understanding and classifying the quality of technical forum questions," in *Proceedings of the 2014 14th International Conference on Quality Software*, ser. QSIC '14. Washington, DC, USA: IEEE Computer Society, 2014, pp. 343–352.

[6] 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*, Sept 2014, pp. 541–544.

[7] D. Romano and M. Pinzger, "Towards a weighted voting system for q&a sites," in *Proceedings of the 2013 IEEE International Conference on Software Maintenance*, ser. ICSM '13. Washington, DC, USA: IEEE Computer Society, 2013, pp. 368–371.

[8] L. Ponzanelli, A. Mocci, and M. Lanza, "Stormed: Stack overflow ready made data," in *Proceedings of the 12th Working Conference on Mining Software Repositories*, ser. MSR '15, 2015, pp. 474–477.

[9] A. A. Sawant and A. Bacchelli, "A dataset for api usage," in *Proceedings of the 12th Working Conference on Mining Software Repositories*, ser. MSR '15. Piscataway, NJ, USA: IEEE Press, 2015, pp. 506–509.

[10] L. Breiman, "Random forests," *Machine learning*, vol. 45, no. 1, pp. 5–32, 2001.

[11] D. W. Hosmer Jr, S. Lemeshow, and R. X. Sturdivant, *Applied logistic regression*. John Wiley & Sons, 2013, vol. 398.

[12] G. A. Seber and A. J. Lee, *Linear regression analysis*. John Wiley & Sons, 2012, vol. 936.

[13] I. Rish, "An empirical study of the naive bayes classifier," in *IJCAI 2001 workshop on empirical methods in artificial intelligence*, vol. 3, no. 22. IBM New York, 2001, pp. 41–46.