

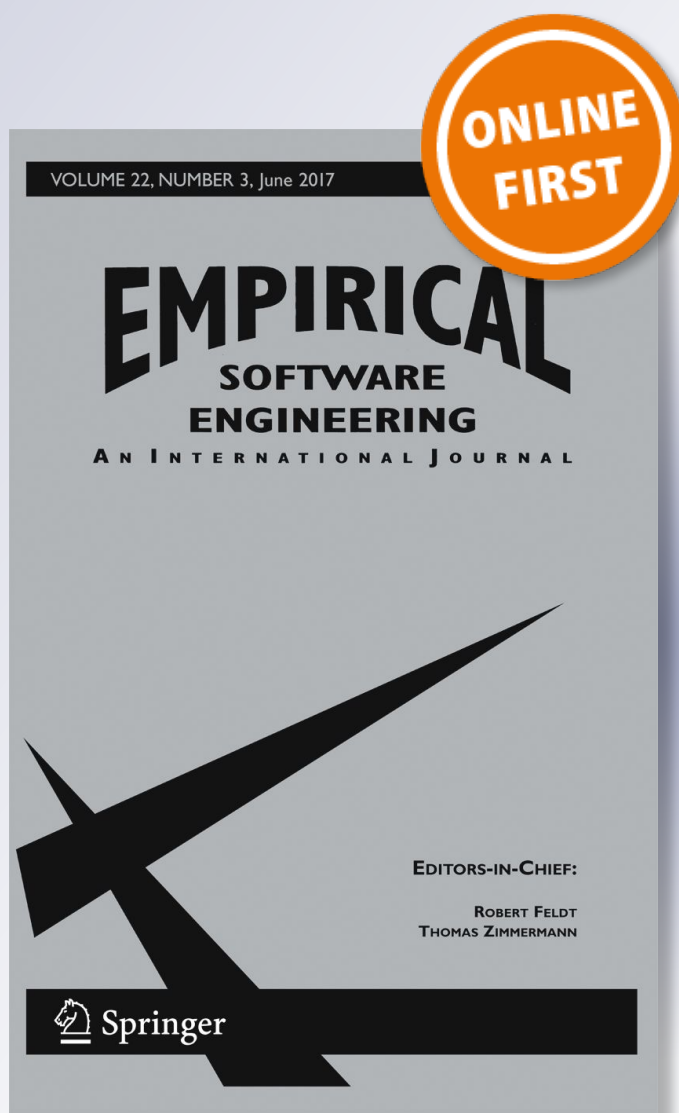
Understanding the factors for fast answers in technical Q&A websites

**Shaowei Wang, Tse-Hsun Chen &
Ahmed E. Hassan**

Empirical Software Engineering
An International Journal

ISSN 1382-3256

Empir Software Eng
DOI 10.1007/s10664-017-9558-5



Your article is protected by copyright and all rights are held exclusively by Springer Science+Business Media, LLC. This e-offprint is for personal use only and shall not be self-archived in electronic repositories. If you wish to self-archive your article, please use the accepted manuscript version for posting on your own website. You may further deposit the accepted manuscript version in any repository, provided it is only made publicly available 12 months after official publication or later and provided acknowledgement is given to the original source of publication and a link is inserted to the published article on Springer's website. The link must be accompanied by the following text: "The final publication is available at link.springer.com".



Understanding the factors for fast answers in technical Q&A websites

An empirical study of four stack exchange websites

Shaowei Wang¹ · Tse-Hsun Chen² ·
Ahmed E. Hassan¹

© Springer Science+Business Media, LLC 2017

Abstract Technical questions and answers (Q&A) websites accumulate a significant amount of knowledge from users. Developers are especially active on these Q&A websites, since developers are constantly facing new development challenges that require help from other experts. Over the years, Q&A website designers have derived several incentive systems (e.g., gamification) to encourage users to answer questions that are posted by others. However, the current incentive systems primarily focus on the quantity and quality of the answers instead of encouraging the rapid answering of questions. Improving the speed of getting an answer can significantly improve the user experience and increase user engagement on such Q&A websites. In this paper, we explore how one may improve the current incentive systems to motivate fast answering of questions. We use a logistic regression model to analyze 46 factors along four dimensions (i.e., question, asker, answer, and answerer dimension) in order to understand the relationship between the studied factors and the needed time to get an accepted answer. We conduct our study on the four most popular (i.e., with the most questions) Q&A Stack Exchange websites: Stack Overflow, Mathematics, Ask Ubuntu, and Superuser. We find that i) factors in the answerer dimension have the strongest effect on the needed time to get an accepted answer, after controlling for other factors; ii) the current incentive system does not recognize non-frequent answerers who often

Communicated by: Per Runeson

✉ Shaowei Wang
shaowei@cs.queensu.ca

Tse-Hsun Chen
peterc@encs.concordia.ca

Ahmed E. Hassan
ahmed@cs.queensu.ca

¹ Software Analysis and Intelligence Lab (SAIL), School of Computing, Queen's University, Kingston, ON, Canada

² Department of Computer Science and Engineering, Concordia University, Montreal, QC, Canada

answer questions which frequent answerers are not able to answer. Such questions that are answered by non-frequent answerers are as important (i.e., have similar range of scores) as those that are answered by frequent answerers; iii) the current incentive system motivates frequent answerers well, but such frequent answerers tend to answer short questions. Our findings suggest that Q&A website designers should improve their incentive systems to motivate non-frequent answerers to be more active and to answer questions fast, in order to shorten the waiting time to receive an answer (especially for questions that require specific knowledge that frequent answerers might not possess). In addition, the question answering incentive system needs to factor in the value and difficulty of answering the questions (e.g., providing more rewards to harder questions or questions that remain unanswered for a long period of time).

Keywords Logistic regression modeling · Factor importance analysis · Q&A websites · Response time

1 Introduction

The Internet is a great medium for accumulating knowledge from people across the world. Due to the large amount of knowledge that is accumulated, search engines, such as Google, provide people with the ability to search for the knowledge in which they are interested. Initially, these search engines only aim to give users the most relevant answers to their interest or question. However, over the past years, search engines have evolved to consider not only delivering correct answers but also the rapid delivery of such answers. For example, Google instant search (Cornea and Weininger 2014) provides instant suggestions to complete queries while also updating results as users type in their question. Moreover, today, most search engines (e.g., Google and Bing) pride themselves about their speed of delivering answers and list the response time of a user query at the top of their results page. A recent study (Colburn 2016) shows that increasing the speed of finding answers on an e-commerce website can increase sales by as much as 17%.

Similar to the Internet, technical questions and answers (Q&A) websites accumulate a significant amount of user-generated knowledge. Stack Exchange is a prominent example of such a website for asking and answering questions in various areas, ranging from technology to science, and even art. One of Stack Exchange's most popular technical Q&A websites is Stack Overflow (SO),¹ which has more than 12.3 million questions, 18.4 million answers, and 5.8 million users, according to the SO data dump of March 2016 (StackOverflow 2016).

Users that are involved in the Stack Exchange community are very active. Stack Overflow reports that there are more than 40 million monthly visitors (StackOverflow 2016), and many developers rely on Stack Overflow for solving the problems that they are currently facing. Hence, developers post their questions on such Q&A websites whenever they are blocked by a problem, in hope to receive responses as soon as possible to clear their roadblocks.

Over the years, Stack Exchange websites have enhanced their incentive systems to attract users to contribute to the knowledge base by asking and answering questions. For example, a developer can earn reputation scores by asking and answering questions on Stack Overflow, and a higher reputation score gives developers more privileges on the website (e.g., access to

¹<http://stackoverflow.com/>

site analytics data). However, the current incentive systems primarily focus on the quantity and quality of the answers with little to no focus on the speed of answering questions.² There are still tens of thousands of questions that took more than one week to receive an accepted answer (see Section 5). In other words, a large number of askers still must wait for a long time before getting an accepted answer.

Similar to implementing instant search on search engines, reducing the needed time to get an answer can significantly improve the user experience when asking questions on Q&A websites. Therefore, it is necessary to study the factors that may affect the needed time to get an accepted answer, and how the current incentive systems can better motivate fast answers to questions.

In this paper, we examine the relationship between different factors and the needed time for a question to get an accepted answer. We conduct our study on the four most popular (i.e., with most questions) technical Stack Exchange websites: Stack Overflow, Mathematics, Super User, and Ask Ubuntu. We study 55,853, 70,336, 7,134, and 10,776 questions on Stack Overflow, Mathematics, Super User, and Ask Ubuntu, respectively. We structure our inquiry using 46 factors along four dimensions:

- *Question*: Various textual and readability features of a question, as well as the popularity and difficulty of the question's tags.
- *Asker*: The reputation of an asker and his/her historical tendency to get answers.
- *Answer*: Textual features that are computed from the text of the accepted answer.
- *Answerer*: The historical activity level of the answerer who answered the question.

Since some factors are not changeable even if we modify the incentive systems (e.g., we cannot easily change how people write their questions and we cannot change the topic of questions), we control for these unchangeable factors in our logistic regression models. We then use the model to understand the relationship between the studied factors and the needed time to get an accepted answer, and how the current incentive systems can be improved to increase the speed of question answering.

Through case studies, we find that:

- 1) There exists a strong relationship between the factors in the answerer dimension and the needed time to get an accepted answer. After controlling for unchangeable factors such as the length of the answer, the speed of how fast an answerer answers questions in the past is the most important factor in our model.
- 2) The current incentive system does not recognize the non-frequent answerers who often answer questions which frequent answerers are not able to answer. In general, non-frequent answerers answer questions slower than frequent answerers. However, the questions that are answered by non-frequent answerers are as important (i.e., have similar range of score) as those that are answered by frequent answerers. Such slow-answered questions that are answered by non-frequent answerers may have remained unanswered if they were not answered by the non-frequent answerers, since such non-frequent answerers may have some unique expertise on certain topics.
- 3) The current incentive system motivates frequent answerers well, but frequent answerers tend to answer easy (in terms of size) questions. Frequent answerers tend to answer easier (in terms of the size of questions) questions than that of non-frequent answerers, which is acknowledged by Stack Overflow developers as well.

²<http://stackoverflow.com/help/whats-reputation>

Our findings highlight the need for Q&A website designers to improve the incentive system to attract the non-frequent answerers so they can become more active and answer questions faster (e.g., rewarding the non-frequent answerers more scores if they stay online for enough time) and improve the question answering incentive system to factor in the value and difficulty of answering the questions (e.g., providing additional rewards for answering hard questions or questions that remain unanswered for a long period of time).

Paper Organization Section 2 introduces the background information about Stack Exchange websites. Section 3 presents the definition of studied factors. Section 4 describes our data collection process. Section 5 presents the result of our preliminary study on the four studied websites. Section 6 presents the results of the case study and our discussion of the results. Section 7 describes the threats to validity of our observation. Section 8 describes the related work. Finally, Section 9 concludes the paper.

2 Background

In this section, we give a brief overview of how Stack Exchange Q&A websites work by using a real-life example from Stack Overflow.

2.1 Question Structure

A question on Stack Overflow (as well as other Stack Exchange Q&A websites) has a title and a body. Each question also contains additional information: tags, the developer who asked the question (asker), the date when the question was posted, question score, and favorite count. The details of each question are described in its body. Askers can attach code or URL(s) to provide more comprehensive information. Tags are added by askers manually to indicate the topics to which a question belongs. The favorite count indicates the number of developers who like this particular question. The question score indicates the total number of up and down votes that this question received. Whenever a developer wants to express that a question or an answer is useful, he/she could vote it up and the score for that particular answer will be increased by one. Similarly, a developer could vote a question/answer down and the score for this question/answer will be decreased. The complexity of a question varies in terms of topics, length, etc. Intuitively, the needed time to get an accepted answer for a question is associated with the complexity of the question. In this paper, we define a set of factors that are related to a question's title and body in an attempt to present the complexity of the question. We study the needed time to get an accepted answer, after controlling for factors that are related to a question's title and body (since these factors are not changeable).

2.2 Reputation Score System

Reputation score is the incentive system that is used on Stack Exchange websites. The current implementation of the incentive system (including all Stack Exchange websites) is designed to encourage users to perform desirable activities by awarding them points. A developer can earn reputation scores through several ways, such as asking good questions and providing useful answers.³ A developer could earn even more reputation scores if the

³<http://stackoverflow.com/help/whats-reputation>

answer is accepted by the asker. The developers could also gain reputation scores by helping improve questions or answers. However, the current reputation score system on Stack Exchange only considers the quality and quantity of the answers and questions that are posted by a user. The reputation score system misses a very important part – the speed of answering a question. In this paper, we would like to examine factors that impact the speed of answering a question and to provide some suggestions to improve the reputation score system based on our findings.

2.3 Tagging System

On Stack Overflow (as well as other Stack Exchange Q&A websites), each question can have at most five tags and must have at least one tag. Askers need to specify the tags of a question when they create the question. Tags can then be used for searching and browsing related questions. Askers with over 1,500 reputation scores are allowed to put any tags (even new tags) to questions; on the other hand, askers with less than 1,500 reputation scores are only allowed to use existing tags. Developers can also subscribe to receive updates on new questions that are associated with certain tags. Some tags are related to hard topics which may take more time to answer, while some tags are related to easy topics which may take less time to answer. Hence, intuitively, tags may have an impact on the needed time to get an accepted answer, and we are interested in studying such relationships in this paper.

2.4 User Profiles

Each registered developer has a profile, which contains information such as his/her reputation score, the tags in which he/she is interested, and the questions and answers that he/she posted. For example, we present the profile of a developer in Fig. 1. The developer's reputation score is 4,071. The questions that he asked in the past were mostly related to "java".

2.5 A Real-life Example

Figure 2 shows an example question that is posted on Stack Overflow.⁴ The title of this question is "How to split a string in Java". This question was posted by the developer "riyana" at 3:01 on August 10th, 2010. The question is basically to ask how to split a string into two strings in Java. Because the question is related to Java and string operation, the question is tagged with two tags, "java" and "string". The asker described the question by attaching a code snippet. The question was marked as a favorite 141 times and the question score count is 671.

Each question can receive many answers. Each answer has a body and contains the information about who answered the question and when the answer was posted. Each answer also has a score count that represents the up and down votes that the answer receives, and a check mark to indicate whether this answer is accepted. If an answer is accepted by the asker, a check mark will appear beneath the score counter. Note that each question in Stack Exchange can only have at most one accepted answer. In some cases, the asker may not accept any answers if he/she thinks that all the received answers are not good enough. In

⁴<http://stackoverflow.com/questions/3481828/how-to-split-a-string-in-java>, last accessed Sep 21th, 2017.

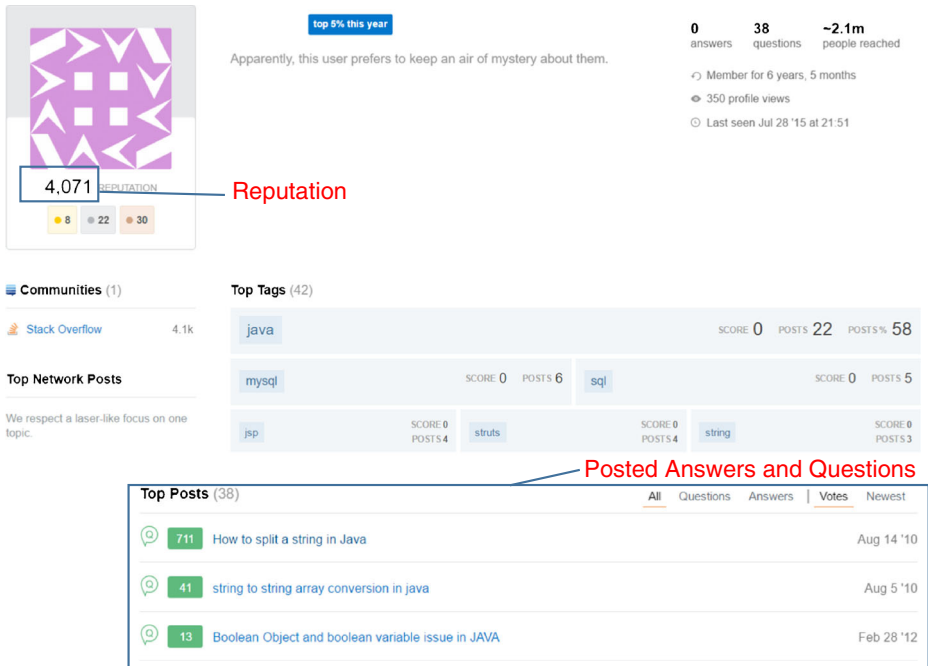


Fig. 1 An example of a user profile on Stack Overflow

Fig. 2, we could see that the answer, that was posted by “BalusC”, was accepted by the asker.

3 Studied Factors

Section 2 highlights the rich information available on Stack Exchange Q&A websites in terms of the content of a question, tags, and user profile. In this section, we discuss how we collect the studied factors that quantify the above-mentioned information. We consider factors along four dimensions: question, the developer who posts the question (asker), answer, and the developer who posts the answer (answerer), when studying the needed time to get an accepted answer for a question. We describe the studied factors in Table 1. The rationale and the calculation steps of each factor are discussed in the following subsections.

3.1 Question Factors

There are 16 question factors that are extracted from a question (e.g., content, title, and tags). In this study, we focus on the question-related factors that are available and are not changing over time (e.g., score and view count may change over time).

3.1.1 Text-Related Factors

Seven of the factors (i.e., Q_Body_Length, Q_Title_Length, Q_Code_Length, Q_Code_Ratio, Q_URL_Number, Q_Capital_Title, and Q_Title_Popularity) in this dimension are related to

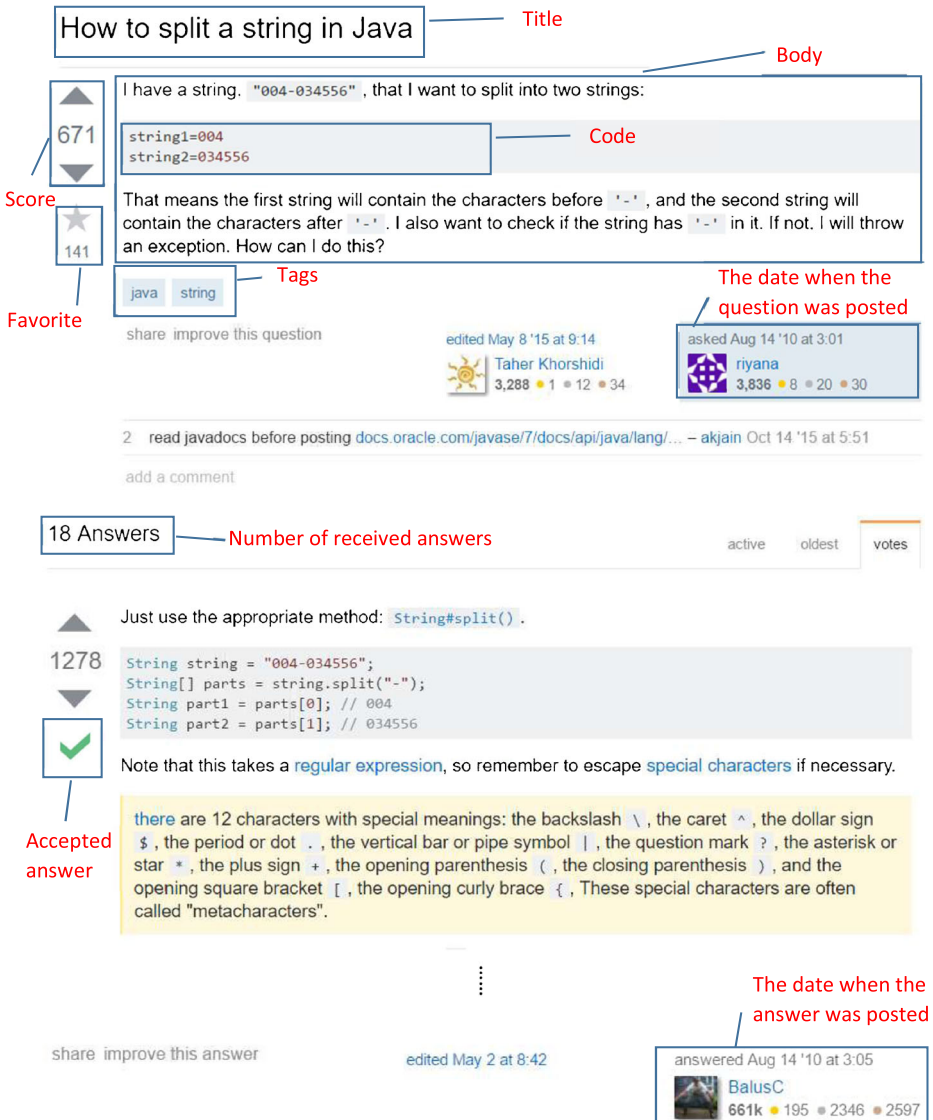


Fig. 2 An example of a question and its accepted answer on Stack Overflow

the content (i.e., title and body) of the question. These factors are used to measure the question complexity. Intuitively, the complexity of a question has a relationship with the speed of receiving accepted answers.

When calculating the factors that are related to code snippets in a question, we first need to identify the code snippets from a question's body. The code can be identified by detecting the tag "<code></code>" in our collected data. We use the hyperlink tag "" to identify URLs. After identifying the code and URL in the body, we are able to calculate the code-related factors in a question (see Table 1). In order to calculate the Q.Title.Popularity, we preprocess (i.e., stop word removal, punctuation removal,

Table 1 Factors potentially affecting the needed time to get an accepted answer for a question along four dimensions: question, asker, answer, and answerer

	Factor name	Explanation	Rationale
Question	Q_Body_Length	Length, in characters, of the question body, including source code and HTML tagging.	Questions that are too short may fail to describe the problem to community members, while overly long questions might discourage potential answerers (Asaduzzaman et al. 2013; Ponzanelli et al. 2014c).
	Q_Title_Length	Length, in characters, of the title of the question.	
	Q_Code_Length	Length, in characters, of the code that is contained in the question.	Attaching code snippets is considered a key factor that affects the quality of questions (Treude et al. 2011;
	Q_Code_Ratio	Ratio of code in the question body.	Ponzanelli et al. 2014c).
	Q_URL_Number	Number of URLs in the question body.	The presence of URLs provides a richer context, which helps answerers better understand a question (Ponzanelli et al. 2014c).
	Q_Capital_Title	1 if the title begins with a capital letter, 0 otherwise.	A capital title may attract more attention from potential answerers (Ponzanelli et al. 2014c).
	Q_Title_Popularity	Popularity of question titles.	A question title with popular/rare words may attract more/less attention from potential answerers.
	Tag_Number	Number of tags of the question.	Tags capture the topics of a question. The number of tags may indicate the complexity of a question.
	Min/Max/Mean/Sum_Tag_Speed	Minimum/maximum/mean/sum value of average time for the tags of the question to get an accepted answer in past one month.	The speed of getting accepted answers for questions with the same tag may be correlated.
Asker	Min/Max/Mean/Sum_Tag_Popularity	Minimum/maximum/mean/sum value of numbers of questions with an accepted answer that is associated with each tag of the question in past one month.	Questions with popular tags may attract more attention from the community, or may be buried by other questions with the same popular tags.
	Mean/Median/Sum_Favorite_Votes	Mean/total/Median number of favorite votes that the asker received in the past.	Recent studies show that there is a correlation between asker reputation and the quality of the post he/she writes (Ponzanelli et al. 2014a, c).
	Mean/Sum/Median_Up_Votes	Mean/total/Median number of up votes that the asker received in the past.	Hence, the questions written by certain askers (e.g., askers who always ask well-received questions) may be easier to understand.
	Mean/Sum/Median_Down_Votes	Mean/total/Median number of down votes that the asker received in the past.	

Table 1 (continued)

	Factor name	Explanation	Rationale
	Question_Accepted_Answer	Number of questions for which the asker received an accepted answer in the past.	Prior studies (Asaduzzaman et al. 2013; Yao et al. 2013; Anderson et al. 2012) find that the expertise level or presentation quality of a question does affect the chance of receiving answers. Thus, the number of accepted/first answers or the speed of receiving answers in the past may indicate the expertise level or presentation skill of the asker.
	Question_Answers	Number of questions for which the asker received answers in the past.	
	Total_Answers	Total number of answers received by the asker in the past.	
	Mean/Max/Min/Median.Speed_Accepted_Answer	Mean/maximum/minimum/median time to get an accepted answer for the asker in the past.	
	Mean/Max/Min/Median.Speed_First_Answer	Mean/maximum/minimum/median time to get the first answer for questions that the asker asked in the past.	
Answer	A_Body_Length	Length, in characters, of the answer body, including source code and HTML tagging.	A longer answer is usually more complex and takes more time to write.
	A_Code_Length	Length, in characters, of the code that is contained in the answer.	Answers with more code may require longer time. Hence, there may be a relationship between such code-related factors of answers and the time of getting accepted answers.
	A_Code_Ratio	Percentage of code in the answer body.	
	A_URL_Number	Number of URLs in the answer body.	The presence of URLs provides a richer context and thus may take more time for the answerer to find the additional contexts.
Answerer	A_Number_Answer	Number of answers that were posted by the answerer in the past.	Intuitively, there may be a relationship between the activity level of an answerer and the speed of the answerer answering a question. The activity level of an answerer could be estimated using the number of questions and answers that she/he posted.
	A_Number_Question	Number of questions that were posted by the answerer in the past.	
	A_Mean/Max/Min/Median.Speed_Answer	Mean/maximum/minimum/median time to answer questions in the past for the answerer.	The speed of answering questions in the past may be a good indicator of the speed of answering questions in the future for the same answerer.

number removal, and stemming) each title, and we calculate the information entropy of each title using the R package **tm**.⁵ The entropy metric is one form of inverse-document frequency (Aizawa 2003). For each question, we calculate the entropy of each unique word in the title, across the titles of all the questions. Hence, the entropy value for a word represents the rarity of a word across all the titles. Then, we take the average of the entropy value of each word in the title. Having too many rare words in the title will result in a small average entropy value. Hence, smaller entropy values indicate that the question is more specific (i.e., rare). This factor measures the popularity of the title of a given question across all question titles in our collected data.

Note that the text-related factors are not changeable even if we modify the incentive systems (e.g., we cannot easily change how people write their questions). Thus we control for these factors when we build our statistical models.

3.1.2 Tag-Related Factors

Tags naturally represent the topics of a question. Intuitively, some topics are easier to answer while some topics are harder to answer. The questions that are associated with the hard topics (tags) are more likely to take a longer period of time to receive an accepted answer than those that are associated with the easy topics (tags). Thus, we extract tag-related factors in various ways. In total, there are 9 factors in this dimension.

Tag_Number is used to measure the complexity of the questions (i.e., more complex questions tend to have more topics). A question that is related to a large number of topics may require more expertises to answer, and may, therefore, affect the time to get an accepted answer.

Tag popularity factors (i.e., Min\Max\Mean\Sum.Tag.Popularity) measure the popularity of tags. In most cases, a question is labeled with multiple tags. Thus, we take the minimum, maximum, mean, and sum of the popularity values of the tags for a question. The popularity of a tag could be an important factor that affects the needed time for getting accepted answers. Questions with popular tags may attract more attention from the community, or may lack answerers to answer the question (e.g., too many questions with the same tag).

Next, we explain how we calculate the tag popularity factors. The popularity of a tag changes over time. Thus, we measure the popularity of a tag using only recent history (i.e., 1 month prior to the posting of a question). We count the number of associated questions as a measure of the popularity of a tag. Given a tag t and a creation date d , the popularity of t on d is denoted by $Popularity(t, d)$, where $Popularity(t, d)$ is the number of questions which were created for the past month before d . Suppose that given a question q with a set of tags $Tags = t_1, t_2, \dots, t_n$. The factors Min\Max\Sum\Mean.Tag.Popularity could be calculated by taking the minimum, maximum, sum, and average of the popularity values ($Popularity(t, d)$) of all the tags ($Tags$) that are associated with the question.

Tag speed factors (i.e., Min\Max\Mean\Sum.Tag.Speed) consider the speed of getting an accepted answer in all the questions that are associated with a tag in a recent past period of time (i.e., past one month prior to the posting of the question). Namely, we compute the average time to get an accepted answer across all the questions that are associated with the same tag. We then take the minimum, maximum, mean, and sum of their average time, since a question usually has multiple tags. The intuition here is that questions that are labeled

⁵<https://cran.r-project.org/web/packages/tm/index.html>

with different tags may have different speed of getting accepted answers. We calculate the speed of a tag t , which is denoted by $Tag_Speed(t)$, using (1), in which Q_t is the set of questions that associated with tag t . $Tag_Speed(t)$ presents the average time for a tag to get an accepted answer.

$$Tag_Speed(t) = \frac{\sum_{q_i \in Q_t} \text{time to get an accepted answer for } q_i}{|Q_t|} \quad (1)$$

We can then calculate $Min\backslash Max\backslash Sum\backslash Mean_Tag_Speed$ by taking the minimum, maximum, sum, and average of the tag speed values ($Tag_Speed(t)$) of all tags of a question.

When calculating the tag speed and popularity factors, we consider the recent past period of time (i.e., one month) instead of the entire history since the speed and popularity of a tag vary over time (see Fig. 3).

Note that the tag-related factors are changeable if the incentive system is changed properly. For example, if we give more rewards to questions that are associated with some specific tags (e.g., rare or hard-to-answer tags), we may probably improve the speed of receiving answers for such questions.

3.2 Asker Factors

Asker factors consider 20 factors that are related to the asker (e.g., the developer who posts the question) in two ways: asker reputation and historical information. We consider the asker reputation because recent studies show that there is a correlation between a user's reputation and the quality of their posts (Ponzanelli et al. 2014a, c). The quality of a question may be associated with the speed of receiving accepted answers for the question. In this study, we do not use the reputation score directly to measure the reputation of the asker for the following two reasons: 1) we require a snapshot of the status of askers when they created the question, but the official data dump is released periodically every few months and it only reports the latest reputation score of a developer; 2) we find many cases where the askers in Mathematics, Ask Ubuntu and Super User gain 101 reputation scores by simply linking their Mathematics/Ask Ubuntu/Super User account to other Stack Exchange accounts. Hence, the reputation score may not measure the contribution of an asker accurately.

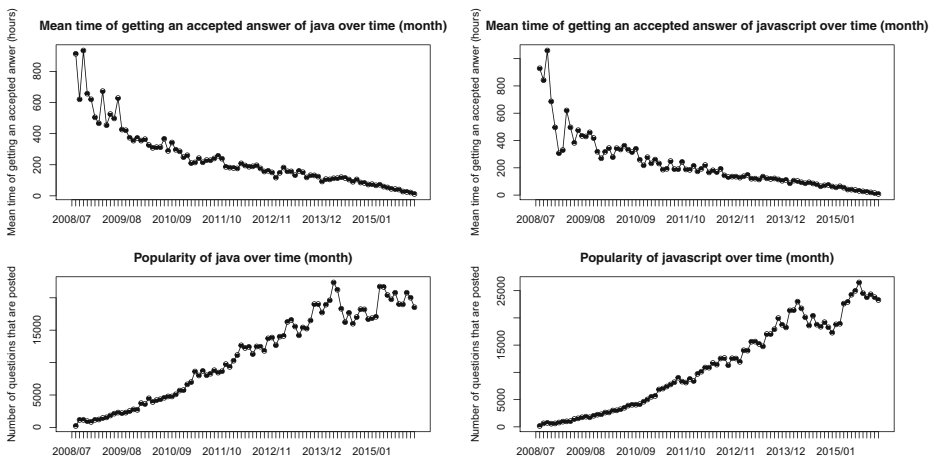


Fig. 3 The trends of speed and popularity of the popular tags “javascript” and “java” on Stack Overflow

Thus, we estimate the reputation of an asker by considering votes using the following proposed approach (Ponzanelli et al. 2014c). We estimate the reputation of an asker using the following factors: Mean\Sum\Median.Favorite.Votes, Mean\Sum\Median.Up.Votes, and Mean\Sum\Median.Down.Votes. As observed in prior studies (Asaduzzaman et al. 2013; Yao et al. 2013; Anderson et al. 2012), the expertise level of an asker or the presentation quality of a question may impact on the likelihood of receiving answers. Therefore, an asker who received fast or more answers (e.g., short time to get an accepted/first answer, or many questions with accepted answers) in the past may indicate that his/her questions may be easier to understand or answer. Thus, the tendency of receiving answers for a particular asker in the past may impact the speed of receiving accepted answers for future questions. Thus, we estimate the tendency of an asker to receive fast answers based on the following factors: Question_Accepted_Answer, Question_Answer, Total_Answer, Mean\Max\Min\Median.Speed_Accepted_Answer, and Mean\Max\Min\Median.Speed_First_Answer.

We present below our approaches to calculate these asker factors in detail. Since the vote data is readily available, we can calculate vote-related factors directly. With respect to factors such as: Question_Accepted_Answer, Question_Answer, Total_Answer, for each user, we count the number of questions he/she asked that have an accepted answer, the number of questions with at least one answer, and the total number of answers for all their questions in the past, respectively. Mean\Max\Min\Median.Speed_Accepted_Answer can be computed according to the description shown in Table 1 by taking the average, max, min, and median of the time to get an accepted answer across all questions that are posted by the developer. We do similar calculations on Mean\Max\Min\Median.Speed_First_Answer by considering the first answer of a question. Note that we consider the activities of the asker before the posting date of a question (i.e., we do not consider their future activity after the posting of an examined question).

Note that the asker factors are not changeable through the current incentive system. Thus, we control for these factors when we build our models.

3.3 Answer Factors

Answer factors consider four factors that are related to the answer of a question: A.Body.Length, A.Code.Length, A.Code.Ratio, and A.URL.Number. The definition and rationale of the factors are listed in Table 1. These factors estimate the complexity of an answer in various ways. Generally, an answerer is more likely to spend longer time to create a more complex and high quality answer.

Note that the answer factors could be controlled by the incentive system. For example, the incentive system could reward more scores to an answer with code than one without code, so answerers are driven to answer questions with attached code to get more scores.

3.4 Answerer Factors

The speed of getting an accepted answer is not only dependent on the question itself and the asker, but it is also related to the answerers who provide such answers. In this paper, we choose two ways to capture the activities of an answerer, which are posting answers and posting questions. We would like to use the quantity of questions and answers that are posted by an answerer and the tendency of an answerer to answer a question in the past to estimate the activeness of the answerer. We would like to study whether the needed time to get an accepted answer is impacted by the activeness of the answerers. Similar to the factors

Table 2 Basic descriptive information about the four websites

Name	Description	Period	#Questions
Stack Overflow	A Q&A community for programmers	2015.01.01–2015.12.31	55,853
Mathematics	A Q&A website for math-related questions	2010.7.20–2015.12.31	70,336
Ask Ubuntu	A Q&A website for Ubuntu users and developers	2009.01.08–2015.12.31	7,134
Super User	A Q&A website for computer enthusiasts and power users	2008.08.01–2015.12.31	10,776

in the asker dimension, we look at the activities of answerers in the past before the posting date of each examined question.

Answerer factors consider six factors that are related to answerers: the number of answers that have been previously posted by the answerer (i.e., *A_Number_Answer*), the number of questions that have been previously posted by the answerer (i.e., *A_Number_Question*), and the speed of responding to a question by the answerer in the past (i.e., *A_Max_Speed_Answer*, *A_Min_Speed_Answer*, *A_Mean_Speed_Answer*, and *A_Median_Speed_Answer*). The factors and their rationale are listed in Table 1.

Note that the answerer factors could be controlled by the incentive systems. For example, the current incentive system on Stack Exchange websites is designed to ensure high-quality answers, so answerers are driven to post high-quality answers. If the incentive system is designed to be more sensitive to the speed of answering a question, the answerer may be driven to answer questions faster.

4 Data Collection

In this section, we describe how we construct the datasets that we use for further analysis.

There are 154 Q&A websites under the Stack Exchange family⁶ as of May 1st, 2016. These websites cover a very wide range of topics, such as technology, culture, art, and business. We choose the top four most popular websites (i.e., most questions as of May 1st, 2016) which are related to software developers. The studied websites are Stack Overflow,⁷ Mathematics,⁸ Ask Ubuntu,⁹ and Super User.¹⁰ The basic description of each website is presented in Table 2.

We downloaded the data dump of these websites.¹¹ The data dump stores all the information for the questions, tags, votes, and user histories of the studied websites in XML files (e.g., *Posts.xml*, *Votes.xml*, *Users.xml*, *Tags.xml*). *Posts.xml* stores all the posted questions and answers. Each question contains a title, body, the ID of the developer who created the question (asker), the creation date, tags that are associated with this question, the ID of the accepted answer for the question (if any). Similarly, each answer contains a body, the creation date, and the ID of the developer who posted the answer. *Votes.xml* stores all the votes

⁶<https://stackexchange.com/sites>

⁷<http://stackoverflow.com/>

⁸<http://math.stackexchange.com/>

⁹<http://askubuntu.com/>

¹⁰<http://superuser.com/>

¹¹<https://archive.org/details/stackexchange>

made on all posts (i.e., both questions and answers). Each vote contains the following information: vote type (e.g., up, down, favorite, etc.), and the ID of the developers who voted and the voting date. In our study, we use Posts.xml and Votes.xml in the data dump.

We collect all the questions prior to Dec 31st, 2015 for Ask Ubuntu, Mathematics and Super User and all the questions for Stack Overflow of the entire year of 2015. We choose the questions that have a score that is larger than 1 and an accepted answer, since we want to make sure that all studied questions have attracted enough attention from the community (Ponzanelli et al. 2014c). We collect the questions for Stack Overflow in 2015 instead of all questions since we want to study the most recent questions. We select all questions before 2016 for the three other websites since the number of questions of the year 2015 is much smaller than Stack Overflow (i.e., 32,018, 7,969, and 7,607 for Mathematics, Ask Ubuntu, and Super User, respectively). Such small size of questions may not allow us to draw a reliable conclusion. We end up with 206,479, 120,159, 36,020, and 56,988 questions from Stack Overflow, Mathematics, Ask Ubuntu, and Super User, respectively.

We further filter out the questions that satisfy one of following criteria:

1. **Questions that are self-answered.** We find that some questions were answered by the same developer who posted the question. The purpose of posting a question for these developers is often not to seek an answer but instead for knowledge sharing (e.g., the askers found the answer themselves). Thus, we omit this type of questions from our study.
2. **Questions with missing data.** We find that some studied factors have missing values. For example, for the speed-related factors (e.g., Max_Tag_Speed and Max_Speed_Accepted_Answer), if the developers have never received any accepted answers in the past, the values of these factors would be missing (i.e., empty). In the following research questions, we build classification models using these factors in order to understand the relationship between the studied factors and the needed time to get an accepted answer. Such missing values affect the quality of the model. There are several common approaches for dealing with missing values (Mockus 2008): 1) analyzing only the available data (i.e., ignoring the missing data); 2) imputing the missing data with replacement values (e.g., median, mean); and 3) imputing the missing data according to a distribution (e.g., use the same values from similar posts). The first option is usually used when the values are missing at random (Briggs et al. 2003). Data is said to be “missing at random” if the reason that the data is missing is unrelated to actual values of the missing data. In our case, the values are missing because there is no historical data for the tags or the developers before the question is created (i.e., the reason that the data is missing is not related to the actual values of the data nor related to the needed time to receive an accepted answer). Thus, we choose the first option for removing missing data.

Based on the first criterion, we remove 25,672, 3344, 5225, and 7046 questions from Stack Overflow, Mathematics, Ask Ubuntu, and Super User, respectively. Based on the second criterion, we end up with 55,853, 70,336, 7134, and 10,776 questions on Stack Overflow, Mathematics, Ask Ubuntu, and Super User, respectively. Table 2 summarizes the data after the cleaning process. After collecting the data, we calculate the factors for each question as we describe in Section 3.

We make the datasets and results of our case study publicly available¹² and encourage others to replicate and verify our studies.

¹²<http://sail.cs.queensu.ca/replication/AnswerSpeedStackExchange/Index.html>

5 Preliminary Study

In this preliminary study, we first present some basic descriptive statistics about the needed time to get an accepted answer (TimeToGetAcceptedAnswer) for the four studied websites. We are also interested in studying the relationship between two simple yet intuitive factors (i.e., the length of the question body (Q_Body_Length) and whether the question contains code snippets) with TimeToGetAcceptedAnswer.

Q_Body_Length is the most intuitive factor that may affect the TimeToGetAcceptedAnswer. If a question has a longer body, the question may be more complex and may require more time to receive an accepted answer. In addition, attaching code snippets may help explain the questions better in technical Q&A websites. Thus, we would like to see if attaching code snippets to a question impacts the speed of receiving an answer.

Results More than half of the answered questions receive an accepted answer within one hour. We present the histograms of TimeToGetAcceptedAnswer across the four studied Q&A websites in Fig. 4. We find that more than half of the answered questions get accepted answers within one hour after the question is posted. There are 69.2%, 63.6%, 57.6%, and 63.7% answered questions that get an accepted answer within one hour on Stack Overflow, Mathematics, Ask Ubuntu, and Super User, respectively. Overall, at least 87.3% of the answered questions are answered within 24 hours across four websites. However, there are still 2144 (3.8%), 2401 (3.4%), 507 (7.1%), and 460 (4.3%) answered questions that received accepted answers beyond one week on Stack Overflow, Mathematics, Ask Ubuntu, and Super User, respectively.

In general, it takes more time to get an accepted answer if a question has a longer body, although the correlation is not strong. Figure 5 presents the boxplots of TimeToGetAcceptedAnswer against the length of question body (i.e., Q_Body_Length). For better visualization, we perform a logarithm-transformation on the value of the TimeToGetAcceptedAnswer and the Q_Body_Length. Across the four studied Q&A websites, we see that the mean value of TimeToGetAcceptedAnswer increases as the length of the question body increases. We also compute the Spearman correlation between TimeToGetAcceptedAnswer

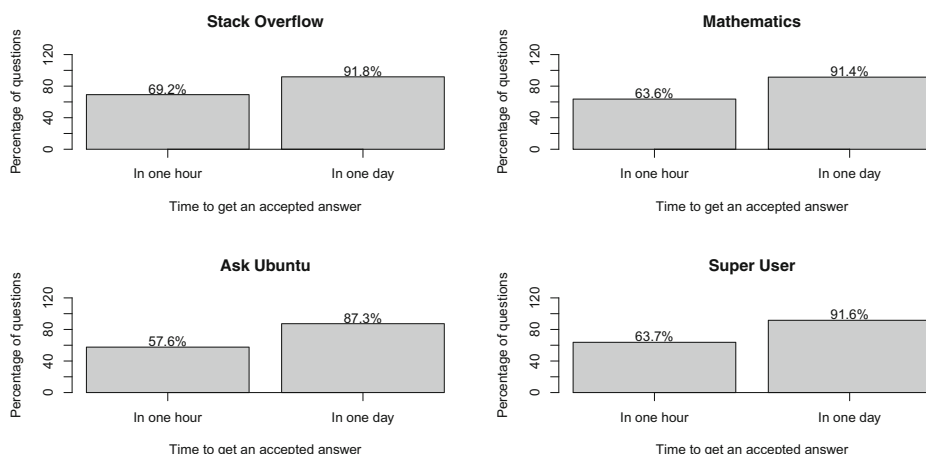


Fig. 4 The percentage of questions that receive an accepted answer in different time on the four studied Q&A websites

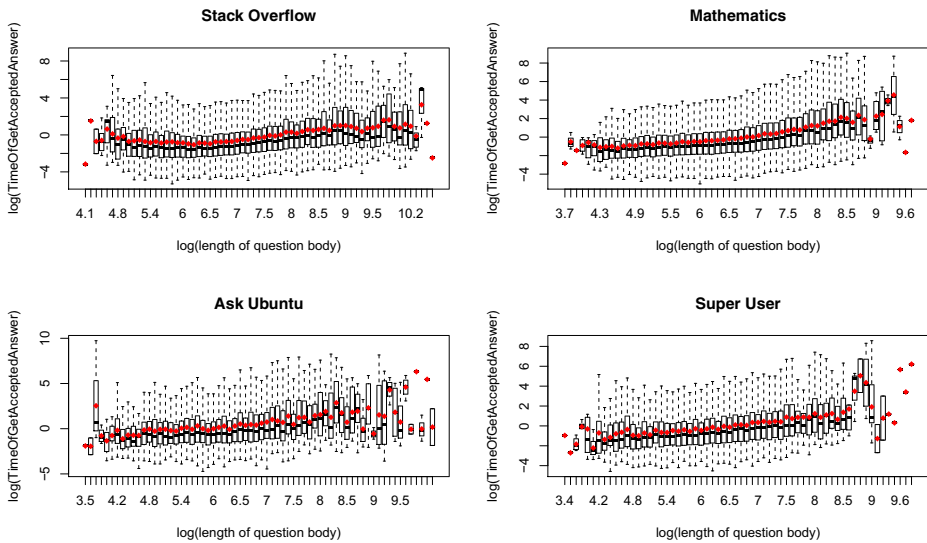


Fig. 5 Boxplots of the time (in hours) of getting an accepted answer against length of the question body. The mean value of each boxplot is also highlighted in red color in the plot

and `Q_Body_Length`. We choose Spearman correlation since it places no assumption on the distribution of the data (i.e., the data does not need to be normally distributed). The correlation values are 0.21, 0.21, 0.16, and 0.19 for Stack Overflow, Mathematics, Ask Ubuntu, and Super User, which implies that there is a correlation between the length of a question and the needed time to receive an accepted answer, even though the relation is weak. Such observation also implies that we must control for the length of a question in our statistical models in order to better understand the effect of other factors on the needed time to get an accepted answer.

Whether a question has code snippets or not has a small impact on the needed time to get an accepted answer on Stack Overflow and a negligible impact on other three websites. We classify the questions into two groups: questions with code snippets (*group_{code}*) and questions without code snippets (*group_{noCode}*). Table 3 presents the results of the comparison between *group_{code}* and *group_{noCode}*. In general, we do not find much difference in terms of `TimeToGetAcceptedAnswer` between these two groups.

In terms of mean values, the `TimeToGetAcceptedAnswer` of *group_{code}* is smaller than that of *group_{noCode}* on Stack Overflow and Ask Ubuntu, while the `TimeToGetAcceptedAnswer` of *group_{code}* is longer than that of *group_{noCode}* on Super User. In

Table 3 The comparison between the questions with code (*group_{code}*) and those without code (*group_{noCode}*)

Websites	Mean of <i>group_{code}</i> (hours)	Mean of <i>group_{noCode}</i> (hours)	<i>p</i> -value	Cliff's <i>d</i>
Stack Overflow	43.0	119.5	1.8E-124	0.21 (small)
Mathematics	66.9	65.1	0.10	0.03 (negligible)
Ask Ubuntu	113.9	163.7	0.37	0.01 (negligible)
Super User	145.2	124.0	1.6E-08	0.07 (negligible)

Mathematics, the mean values of the TimeToGetAcceptedAnswer of two groups are almost the same. We perform a Mann-Whitney U test (Moore et al. 2009) and Bonferroni correction (Dunn 1961) to test whether or not the differences of TimeToGetAcceptedAnswer are statistically significant between the two groups. We choose Mann-Whitney U test since it is a non-parametric test (does not have any assumption on the underlying data distribution). We use Bonferroni correction to control the familywise error rate in multiple comparisons. We also compute Cliff's d (Cliff 1993), which measures the effect size of the differences between two groups (i.e., how large is the difference). The effect size is assessed using the thresholds provided by Cliff (1993), i.e. $|d| < 0.147$ means the effect size is negligible, $|d| < 0.33$ means small, $|d| < 0.474$ means medium, and large otherwise. From Table 3, we see that the differences between the two groups on Mathematics and Ask Ubuntu are not significant (adjusted p -value > 0.0125), while the differences on Stack Overflow and Super User are statistical significant (adjusted p -value < 0.0125). In terms of the effect size, the differences on Stack Overflow is small and negligible for the other three websites.

In short, although most answered questions receive an accepted answer within an hour, there are still hundreds of thousands of questions that take a long time to be answered, which may delay developers. In addition, the time to receive an accepted answer has a weak relationship with the length of a question and whether a question contains code snippet. There may be other factors that have a more important impact on the speed of getting an accepted answer. Hence, in the next section, we use a logistic regression to build a model using multiple factors to further understand which factors are related to the speed of getting an accepted answer.

6 Case Study Results

In this section, we first present the approach that we use to study the relationship between the studied factors and the needed time to get an accepted answer. Then, we discuss our case study results.

6.1 Using Regression Models to Study the Relationship Between the Studied Factors and the Speed of Getting an Accepted Answer

As we find in Section 5, factors that measure the complexity (body length) and richness (whether a question contains code snippets) of a question have a low correlation with the needed time to get an accepted answer. Hence, in this Section, we want to further investigate the relationship between each studied factor in the four dimensions (as presented in Section 3) with the needed time to get an accepted answer (referred to as TimeToGetAcceptedAnswer). We hope the result can help a Q&A website designers improve the incentive systems to aim to shorten the needed time to receive an accepted answer for a question.

Approach We are interested in studying the factors that may be used to distinguish between the questions that receive an accepted answer rapidly and the questions that take a long time to receive an accepted answer. To do so, we use classification models to understand the impact of each studied factor on the speed (i.e., fast and slow) of getting an accepted answer for a question. Similar to prior studies (McIntosh et al. 2016; Thongtanunam et al. 2016; Chen et al. 2012), our goal of building a classification model is not to predict the speed of getting an accepted answer for a question, but to understand the relationship between the factors (referred as the explanatory variables of the model) and the

speed of getting an accepted answer for a question (referred as the response variable of the model).

We sort the questions based on their needed time to get an accepted answer, and then label the top 20% of questions as the fast-answered questions and bottom 20% of questions as the slow-answered questions. Then, we remove correlated and redundant factors and build our models by using logistic regression model. Logistic regression model enables us to examine the effect of one or more variables on a response variable when controlling for other variables. Similar to previous work (McIntosh et al. 2016; Thongtanunam et al. 2016), we added non-linear terms in the model to capture more complex relationship in the data by employing restricted cubic splines (Harrell 2006). We use the R package **rms**¹³ as the implementation of our logistic regression model. We use AUC and bootstrapping to assess the explanatory power of the logistic regression model by following prior studies (McIntosh et al. 2016; Thongtanunam et al. 2016). High AUC means the model has high ability to capture the relationship between the explanatory variables and the response variable. For more details about the process of model construction, please see [Appendix](#).

To understand the impact of each factor to the TimeToGetAcceptedAnswer. We use the **anova** function in the R package **rms** to compute the Wald χ^2 value (i.e., impact) and the statistical significance (p -value) of each factor. We choose ANOVA since the studied factors are normalized (see Normality Adjustment in [Appendix](#)) and independent of each other. To understand the impact of each dimension of factors, we also jointly test the Wald χ^2 on each dimension. The larger the Wald χ^2 value, the larger the impact of a factor on the TimeToGetAcceptedAnswer. To ease the comparison across the four websites, we present the overall and non-linear (NL) Wald χ^2 of each factor as the **proportion** in relation to the total Wald χ^2 of the corresponding model for each website. Hence, the sum of Wald χ^2 of all factors is 1.

We use the **Predict** function in the **rms** R package to plot the estimated likelihood of whether the speed of getting an accepted answer for a question belongs to the fast or slow category against a factor. The analysis allows us to further understand how a factor affects the value of the response variable. We hold the other factors at their median values when exploring one factor.

Results Our resulting models are stable and have high explanatory power. Table 4 shows that when using the remaining factors after the variable selection process, our models achieve AUC values of 0.946, 0.942, 0.850, and 0.857 on Stack Overflow, Mathematics, Ask Ubuntu, and Super User, respectively. The high AUC values suggest that our models have a high explanatory power when studying the needed time to get an accepted answer (i.e., fast or slow). The AUC optimism is also small, ranging from 0.0005 to 0.008, which means that the resulting models are stable (i.e., not overfitted).

After controlling for unchangeable factors, such as the length of the answer, the speed of how fast an answerer answers questions in the past is the most important factor in our models. Table 4 shows the results of our models and the effect of each factor on TimeToGetAcceptedAnswer. Intuitively, longer answers require more time to write. As supported by the results shown in Table 4, A.Body.Length is ranked as the most important factor in Mathematics, and is ranked as the second most important factor in the rest of the studied websites. The relationship between A.Body.Length and TimeToGetAcceptedAnswer is

¹³<https://cran.r-project.org/web/packages/rms/index.html>

Table 4 An overview of the results of the regression models

Factor	Stack overflow		Mathematics		Ask ubuntu		Super user		
AUC	0.946		0.942		0.850		0.857		
AUC optimism	0.0008		0.0005		0.008		0.005		
Wald χ^2	5962.4		7613.6		700.1		1083.7		
Budget Degrees of Freedom (D.F.)	3723		4689		475		785		
Degrees of Freedom (D.F.) Spent	42		41		40		39		
	Overall	NL	Overall	NL	Overall	NL	Overall	NL	
Question									
Q_URL.Number	<i>D.F.</i>	1	–	1	–	1	–	1	–
	χ^2	0.2***		0		0.1		0	
Q_Body.Length	<i>D.F.</i>	1	–	2	1	3	2	3	2
	χ^2	6.0***		4.2***	0.4***	12.9***	1*	10.6***	0.6*
Q_Title.Length	<i>D.F.</i>	1	–	1	–	1	–	1	–
	χ^2	0.3***		0		0.7*		0.1	
Q_Code.Length	<i>D.F.</i>	1	–	1	–	1	–	1	–
	χ^2	0.7***		0		3.4***		0.2	
Q_Title.Popularity	<i>D.F.</i>	2	1	1	–	1	–	1	–
	χ^2	1.5***	0.1***	0.4***		1.9***		1.0***	
Q_Capital.Title	<i>D.F.</i>	1	–	1	–	1	–	1	–
	χ^2	0		0.1*		0		0	
Tag.Number	<i>D.F.</i>	1		1		1		1	
	χ^2	0.5***		0.6***		0.3		0	
Min.Tag.Speed	<i>D.F.</i>	2	1	1	–	1	–	1	–
	χ^2	0.2**	0.1***	0.1***		0.6***		0	
Mean.Tag.Speed	<i>D.F.</i>	3	2	3	2	3	2	2	1
	χ^2	6.4***	3.3***	3.0***	2.6***	2.4***	1.6***	2.6***	2.2***
Min.Tag.Popularity	<i>D.F.</i>	2	1	1	–	1	–	1	–
	χ^2	0.2***	0.2***	0		0		0.4***	
Mean.Tag.Popularity	<i>D.F.</i>	2	1	1	–	1	–	1	–
	χ^2	1.4***	0	0.2***		0		0.1	
Asker									
Mean.Favorite.Votes	<i>D.F.</i>	1	–	1	–	1	–	1	–
	χ^2	0		0.3***		0.6*	–	0.1	
Median.Favorite.Votes	<i>D.F.</i>	1	–	1	–	1	–	1	–
	χ^2	0		0		1.4***		0	
Sum.Up.Votes	<i>D.F.</i>	1	–	1	–	+		1	–
	χ^2	1.0***		0.5***				0.1	
Median.Up.Votes	<i>D.F.</i>	1	–	1	–	1	–	1	–
	χ^2	0		0.1**		0.5*		0	
Median.Down.Votes	<i>D.F.</i>	1	–	1		1	–	+	
	χ^2	0.1***		0		0.5***			

Table 4 (continued)

	Overall		NL		Overall		NL		Overall		NL	
Sum_Down_Votes	<i>D.F.</i>	1	–	1	–	1	–	1	–			
	χ^2	0.2***		0		1.8***		0.1				
Total_Answers	<i>D.F.</i>	1	–	1	–	1	–	1	–			
	χ^2	0.4***		1.0***		1.0**		0.4				
Min_Speed_Accepted_Answer	<i>D.F.</i>	1	–	1	–	1	–	1	–			
	χ^2	0		0		1.0**		0.5**				
Median_Speed_Accepted_Answer	<i>D.F.</i>	1	–	3	2	2	1	1	–			
	χ^2	0*	–	1.6***	1.4***	1.5**	0.8**	0.6**				
Mean_Speed_Accepted_Answer	<i>D.F.</i>	1	–	2	1	1	–	1	–			
	χ^2	0.2***		0.4***	0.1***	0.1		0.3*				
Answer												
A_URL_Number	<i>D.F.</i>	1	–	1	–	1	–	1	–			
	χ^2	0.1**		0		0		0.2				
A_Body_Length	<i>D.F.</i>	2	1	4	3	4	3	4	3			
	χ^2	7.9***	0.5***	41.7***	0.9***	9.0***	0.5	12.7***	0			
A_Code_Length	<i>D.F.</i>	2	1	1	–	3	2	2	1			
	χ^2	0.5***	0.3***	0		0.9***	0.8***	0.7**	0.2			
Answerer												
A_Number_Answer	<i>D.F.</i>	2	1	1	–	1	–	1	–			
	χ^2	0.4***	0	0.7***		0.1		0.1				
A_Number_Question	<i>D.F.</i>	1	–	1	–	1	–	1	–			
	χ^2	0.5***		0.3***		0		0.7***				
A_Median_Speed_Answer	<i>D.F.</i>	4	3	4	3	4	3	4	3			
	χ^2	20.2***	14.8***	32.3***	22.1***	39.6***	21.0***	47.7***	30.5***			
A_Mean_Speed_Answer	<i>D.F.</i>	2	1	2	1	2	1	3	2			
	χ^2	1.4***	1.2***	0.1*	0	0.1	0.1	0.3	0.1			
A_Max_Speed_Answer	<i>D.F.</i>	1	–	+		+		+				
	χ^2	0.7***										
Dimension												
Question	<i>D.F.</i>	17		14		14		14				
	χ^2	33.9***		12.4***		22.6***		17.9***				
Asker	<i>D.F.</i>	10		13		11		10				
	χ^2	3.0***		7.4***		6.8***		2.7***				
Answer	<i>D.F.</i>	5		6		7		6				
	χ^2	20.6***		39.1***		20.7***		21.5***				
Answerer	<i>D.F.</i>	10		8		8		9				
	χ^2	42.4***		41.0***		49.7***		57.9***				

The overall and non-linear (NL) Wald χ^2 of each factor is shown as the **proportion** in relation to the total Wald χ^2 of the model. The top five factors for each website are shown in bold and italic. (*) $p < 0.05$; (**) $p < 0.01$; (***) $p < 0.001$. (+) Discarded during factor selection; (–) Non-linear term not allocated

almost linear, since the non-linear (i.e., non-linear column in Table 4) term does not provide much explanatory power to the models.

After controlling for A.Body.Length, the median speed of answering questions in the past for an answerer (A.Median.Speed.Answer) contributes the most in the regression models across four websites (see the overall Wald χ^2 proportion values). We could also see that the relationship between A.Median.Speed.Answer and TimeToGetAcceptedAnswer is non-linear, since the non-linear term of A.Median.Speed.Answer provides a statistically significant and large explanatory power to the model. From Table 4, we observe that in most cases, non-linear terms do not provide much explanatory power to the models, except for some factors that are related to speed (i.e., A.Mean.Speed.Answer, Mean.Tag.Speed, and A.Median.Speed.Answer), which make significant contributions to the models.

Figure 6 shows the relationship between the most important factors and the TimeToGetAcceptedAnswer. The gray area shows the confidence interval. The larger the confidence interval, the wider the gray area (i.e., the relationship is less clear). We find that the probability of getting a fast answer increases as the length of answer body decreases across four websites. We can also see that the probability of getting a slow answer increases significantly when the value of A.Median.Speed.Answer increases up until an inflection point with a small confidence interval (i.e., the gray bands are narrow). After the inflection point, the curve goes down gradually but with a wide confidence interval. The analysis result indicates that a question is more likely to receive a fast accepted answer from answerers who previously answered questions fast in the past. After the inflection point, the probability goes down slowly with a larger uncertainty (i.e., the relationship is less clear due to the lack of data points in that data range).

The likelihood of receiving a fast accepted answer relies mostly on the answerers rather than on factors in the question, asker, and answer dimensions. As shown in Table 4, we can see that the answerer dimension has the largest explanatory power in the model across the four websites. The finding suggests that the speed of receiving an accepted answer

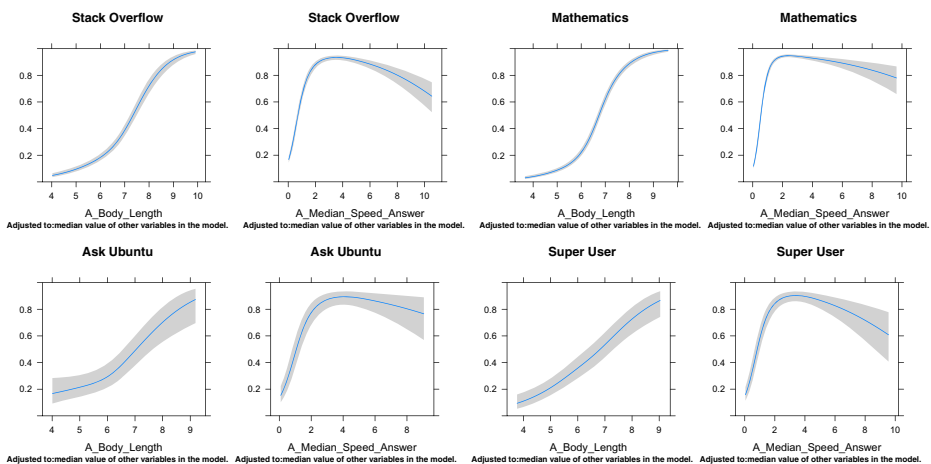


Fig. 6 The estimated probability when the values of A.Body.Length, A.Median.Speed.Answer change. Y axis is the probability of getting a slow answer. X axis is the value after taking the logarithm of factors. The gray area shows the 95% confidence interval

Table 5 The results of cross-website model validation in terms of the ratio in relation to the AUC values that are obtained from the models that are trained from the same website

Training → Testing	Ratio	Training → Testing	Ratio
Super User → Ask Ubuntu	0.99	Stack Overflow → Super User	0.97
Super User → Stack Overflow	0.98	Stack Overflow → Ask Ubuntu	0.98
Super User → Mathematics	0.98	Stack Overflow → Mathematics	0.98
Ask Ubuntu → Super User	0.98	Mathematics → Ask Ubuntu	0.98
Ask Ubuntu → Stack Overflow	0.98	Mathematics → Super User	0.99
Ask Ubuntu → Mathematics	0.97	Mathematics → Stack Overflow	0.99

across the websites relies primarily on the community, i.e., the answerers. In other words, when an asker posts a new question on the Q&A website, the biggest factor that may affect the needed time to receive an answer is the person who answers the actual question (after controlling for unchangeable factors).

Our findings suggest that it is very important to find the most suitable answerers to answer a question and motivate them to answer questions faster. Thus, in order to shorten the waiting time for an asker to get an accepted answer, we suggest that Q&A website designers should make the incentive system more sensitive to the speed of answering a question for an answerer. In addition, Q&A website designers should spend more efforts on delivering the questions to the most suitable answerers as soon as possible.

Discussion We are also interested in investigating whether the relationships between the studied factors and the needed time to get an accepted answer are consistent across Stack Exchange Q&A websites. By knowing this, we can further verify the importance of the factors when studying the needed time to get an accepted answer.

Thus, we perform a cross-website model validation. In each iteration, we pick one Q&A website as the training data to train the model and test the learned model on the rest of the Q&A websites. If the model built from one website works well on other websites, it indicates that the studied factors share a similar relationship with the needed time to get an accepted answer across different websites.

The relationships between the studied factors and the needed time to get an accepted answer are consistent across websites. Table 5 presents the results of cross-website validation in terms of the ratio in relation to the AUC value obtained from the models trained on the same website. The ratios range from 0.97 to 0.99, which indicates that the model trained from other websites performs as good as the one trained from itself, which further indicates that the relationships between the studied factors and the needed time to get an accepted answer are consistent across websites. Our finding indicates that regardless of the types of the questions that are asked (e.g., computer science theory, programming, math, or system administration), finding the right answerers who can answer your question can significantly help reduce the needed time to get an accepted answer.

After controlling for unchangeable factors, the speed of how fast an answerer answers questions in the past is the most important factor in our model. Thus, in order to shorten the waiting time for an asker to get an accepted answer, Q&A website designers should make the incentive system more sensitive to the speed of answering a question and spend more efforts on delivering the questions to the most appropriate answerers as soon as possible.

6.2 Understanding the Relationship Between the Answerer Community and the Speed of Getting an Accepted Answer

In the previous subsection, we find that the speed of receiving an accepted answer across the websites relies primarily on the communities - answerers. In addition, we find that the effect of the studied factors on the needed time to get an accepted answer is consistent across the studied websites. Thus, in this section, we would like to investigate more about the answerer community. We would like to know who actually answers the questions. We also would like to investigate the potential reason that drives answerers to answer questions. By better understanding the answerer community, we can provide some suggestions to the Q&A website designers on how to improve the incentive system to attract more answerers to answer questions faster.

Approach To understand who answers questions, we classify the answerers into different groups based on the number of questions that they answered in the past. Then, we present some basic descriptive (i.e., the number of questions that are answered by different groups of answerers and the percentage of different groups of answerers) statistics about the answerers who belong to the different groups. We use plots to visualize our results relative to the different groups of answerers. We also compare the TimeToGetAcceptedAnswer of the questions that were answered by non-frequent answerers (i.e., those who answered no more than 5 questions in the past) and frequent answerers (i.e., those who answered more than 5 questions in the past). We perform a Mann-Whitney U test (Moore et al. 2009) and Bonferroni correction (Dunn 1961) to determine whether the differences between two groups are statistically significant. Finally, we use Cliff's d (Cliff 1993) to determine the effect size of the differences between the two groups. More specifically, to understand who actually answers slowly, we analyze the relationship between the slow-answered questions (i.e., bottom 20% of the questions) and the different answerer groups. To understand the potential reasons that drive answerers to answer the questions, we analyze how the reputation scores change over time and to examine whether the current reputation score system motivates the answerers well.

Results 86–96% of the accepted answers are written by frequent answerers. Figure 7 presents the percentage of the questions that were answered by the answerers who have different levels of contribution (i.e., answered a different number of questions in the past). We could see that 94.6%, 96.1%, 88.9% and 86.2% of the questions in Stack Overflow, Mathematics, Ask Ubuntu, and Super User were answered by frequent answerers (developers who answered more than 5 questions in the past). We find that more answers were answered by frequent answerers on Stack Overflow and Mathematics than on Ask Ubuntu and Super User. On Ask Ubuntu and Super User, there is a higher ratio of answerers who only answered one or two questions in the past.

In general, on Stack Overflow and Mathematics, the questions were almost answered by the same group of answerers - frequent answerers. Such differences among the answerer community on the four studied websites may also explain why the answerer dimension has a smaller, but still very significant, effect on the TimeToGetAnAcceptedAnswer on Stack Overflow and Mathematics than on the other two websites, where almost all questions are answered by frequent answerers.

In general, non-frequent answerers answer questions slower than frequent answerers. However, the questions that are answered by non-frequent answerers are as important (i.e., have similar range of scores) as those that are answered by frequent answerers. To further

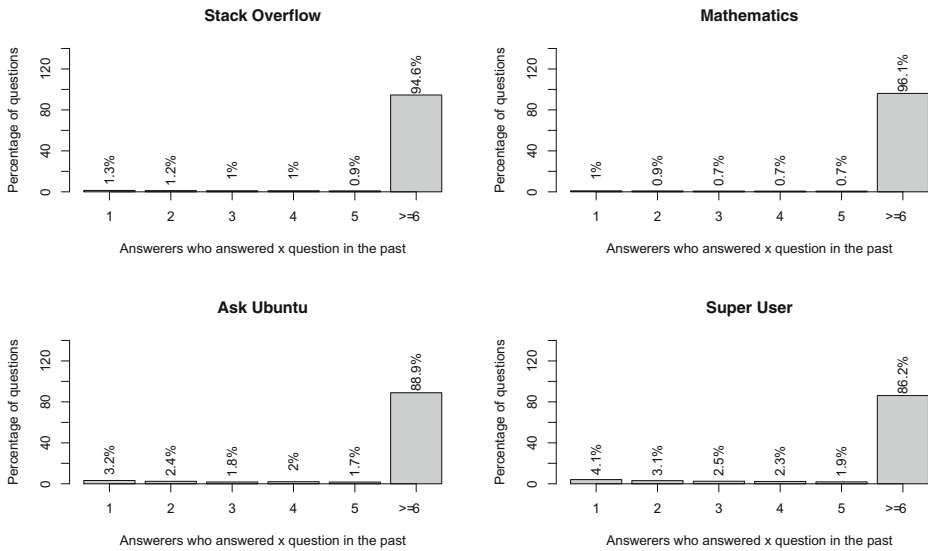


Fig. 7 The percentage of questions that are answered by answerers who answered X question in the past

understand the relationship between different answerer groups and the TimeToGetAnAcceptedAnswer, we further study how fast do answerers in different groups answer questions. Table 6 shows the difference of TimeToGetAcceptedAnswer between frequent answerers and non-frequent answerers. We see a consistent pattern that frequent answerers answer questions faster than non-frequent answerers across the four websites. The mean values of TimeToGetAnAcceptedAnswer of frequent answerers are 43.4, 61.9, 117.2, and 103.4 hours on Stack Overflow, Mathematics, Ask Ubuntu, and Super User, respectively; however, the mean values of TimeToGetAnAcceptedAnswer of non-frequent answerers are 267.7, 225.7, 425.0, and 347.2 hours on these websites, respectively.

On average, the frequent answerers answer questions within 81 hours (i.e., 3.4 days) across the four websites, while non-frequent answerers take 316.4 hours (i.e., 13.1 days) to answer questions. The results of Mann-Whitney U test and Bonferroni correction show that the differences are statistically significant (adjusted p -value < 0.0125). The results of Cliff's d suggests that the effect sizes of the differences are medium, small, small, and negligible on Stack Overflow, Mathematics, Ask Ubuntu, and Super User, respectively. In general, non-frequent answerers answer questions much slower than frequent answerers.

Table 6 The comparison of TimeToGetAnAcceptedAnswer between frequent answerers (FA) and non-frequent answerers (NFA)

Websites	Mean of FA (hours)	Median of FA (hours)	Mean of NFA (hours)	Median of NFA (hours)	p -value	Cliff's d
Stack Overflow	43.4	0.4	267.7	1.5	2.2E-16	0.35 (medium)
Mathematics	61.9	0.5	225.7	0.9	2.2E-16	0.18 (small)
Ask Ubuntu	117.2	0.7	425.0	1.1	2.5E-13	0.16 (small)
Super User	103.4	0.5	347.2	0.8	1.2E-13	0.12 (negligible)

Table 7 The comparison of score between the questions that are answered by non-frequent answerers (NFA) and those that are answered by frequent answerers (FA)

Websites	Mean of FA	Mean of NFA	<i>p</i> -value	Cliff's <i>d</i>
Stack Overflow	3.5	3.5	0.34	0.0004 (negligible)
Mathematics	4.1	4.2	0.96	0.01 (negligible)
Ask Ubuntu	9.1	10.4	0.94	0.05 (negligible)
Super User	7.3	7.5	0.51	0.02 (negligible)

We also look at the scores of the questions that are answered by non-frequent answerers and frequent answerers (see Table 7). The results show that there are no significant differences between the questions that are answered by non-frequent answerers and frequent answerers. In other words, the questions that are answered by non-frequent answerers are as important as those that are answered by frequent answerers. For instance, Fig. 8 presents an example of a highly-scored question that waited for more than one week to receive an accepted answer from a non-frequent answerer.¹⁴ The question received 82 scores. Its corresponding accepted answer that was provided by the non-frequent answerer received 98 scores and an additional 100 bounties from the community, which indicates that the answer is not only helpful to the asker, but also very useful to other users on Super User.

At least 61.3% of the questions that are answered by non-frequent answerers are slow-answered questions. Such slow-answered questions are likely to remain unanswered if they were not answered by the non-frequent answerers.

86.9%, 71.0%, 65.5%, and 61.3% of the questions that are answered by non-frequent answerers are slow-answered questions on Stack Overflow, Mathematics, Ask Ubuntu, and Super User. Moreover, we also look at the number of slow-answered questions that are answered by non-frequent answerers. We observe that 1245 (11.4%), 799 (5.7%), 232 (16.2%), and 409 (19.0%) slow-answered questions are answered by non-frequent answerers on Stack Overflow, Mathematics, Ask Ubuntu, and Super User, respectively. Such slow-answered questions probably would have remained unanswered if they were not answered by non-frequent answerers. Anderson et al. (2012) observed that the answerers on Stack Overflow are organized like a latent “pyramid” with active answerers at the top. Once questions are created, frequent answerers, who are active on Q&A websites, would try to answer the questions according to their expertise. However, the remaining questions would need to wait for another set of answerers (i.e., non-frequent answerers) to answer.




In other words, such slow-answered questions, which probably require unique expertise, could only be answered by such non-frequent answerers, which is compatible with the prior observation that some answerers may have unique expertise on specific topics (Adamic et al. 2008).

For example, Fig. 9 presents the profile of the user that provided the accepted answer for the question that is shown in Fig. 8. This non-frequent answerer only answered four questions in total. However, two of them received 56 and 98 scores and both of these two questions are related to “search on Windows 10”, which indicates that the non-frequent answerer likely has high expertise on “search on Windows 10”.

The current incentive system only motivates frequent answerers well, but not non-frequent answerers. Frequent answerers tend to answer easier questions than non-frequent answerers.

¹⁴<https://superuser.com/questions/950009/cortana-search-is-not-finding-applications-on-windows-10/>

Cortana Search is not finding applications on Windows 10


 82

 43





For the past several versions (since Windows Vista, I think), if you hit the `Windows key` and start typing, Windows will search for applications.

Since upgrading to Windows 10 with **Cortana**, she is only hit or miss at finding applications. And she doesn't do partial searches, either.

Some examples:

- `WinKey` + type "Paint" does not find "mspaint". Instead it suggests some applications from the store.
- `WinKey` + type "Excel" does not find Excel. Nor does typing "Word" find Word. However, typing OneNote finds OneNote.

What is the reason for this behavior and how I can get Cortana to find the applications installed on my machine again? Is there a setting I am missing?


 98



+100

Found a solution here: [Cortana not finding Desktop apps when searching for them](#)

Here is the relevant part:

I reinstalled **Cortana** using the following procedure:

1. Open an elevated Command Prompt window (press win + X, and then press A)
2. Type `start powershell` and press enter
3. Run the command (in one line):

```
Get-AppXPackage -Name Microsoft.Windows.Cortana | Foreach {Add-AppxPackage -DisableDevelopmentMode -Register "$($_.InstallLocation)\AppXManifest.xml"}
```

After 30 seconds the problem was solved on my machine. Incredible.

[share](#) [improve this answer](#)

[edited Aug 14 '15 at 16:50](#)


[answered Aug 12 '15 at 3:29](#)
 **Augusto Barreto**
 1,792 ● 8 ● 9

Fig. 8 An example of a high-scored question that waited for more than one week to receive an accepted answer from a non-frequent answerer

One possible reason that drives frequent answerers to be more active may be that they have a much stronger interest in increasing their reputation scores than non-frequent answerers. Figure 10 shows the increase in reputation score from March 2016 to June 2016 for different groups of developers that have different reputation scores (i.e., $\log(x)$). The finding gives an initial evidence that the developers who have more reputation scores are more active on increasing their reputation scores and the developers that have low reputation scores are not motivated by the incentive systems. The one with high reputation scores are usually frequent answerers and the one with low reputation scores are usually non-frequent answerers (i.e., the reputation score of a developer has a high correlation with the number of questions that are answered by this developer with a correlation value at least of 0.87 across four web-sites). In other words, the current incentive system only attracts some of the answerers (i.e.,

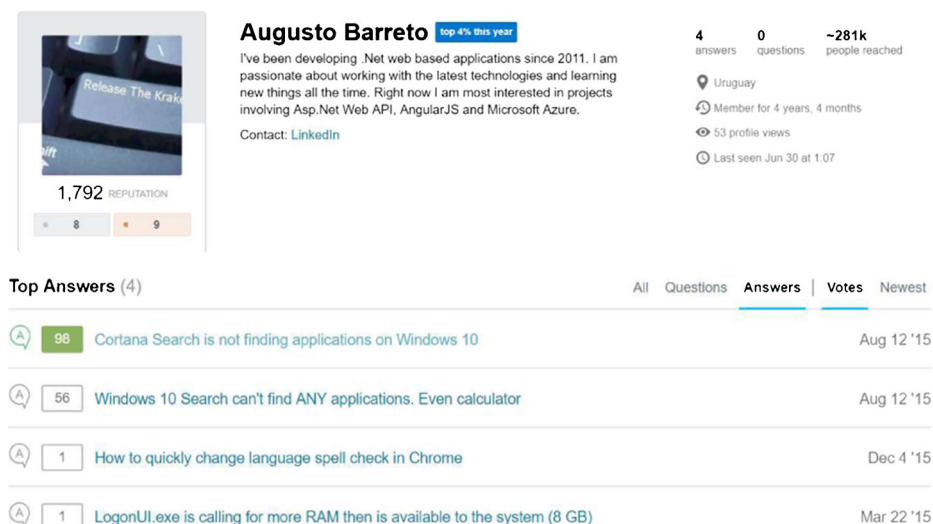


Fig. 9 An example of profile of a non-frequent answerer whose expertise in “search on windows 10”

the frequent answerers who have high reputation scores) while there is still a considerable number of non-frequent answerers who are not attracted by the incentive system.

These non-frequent answerers could have answered questions fast, but due to some reasons the answers were delayed. One possible reason is that the non-frequent answerers are not active enough to spend time on the Q&A websites in search of questions to answer. However, such non-frequent answerers are important for the community since they have their own expertise to answer some very specific questions (i.e., slow-answered questions). It would significantly improve the askers' satisfaction if these slow-answered questions could be answered fast. For instance, Fig. 11 presents a question that waited for about *one year* to get its accepted answer from a non-frequent answerer on Stack Overflow.¹⁵ The question is about the error that comes from the Julia program language when using a Python library called PyPloy and is tagged with “matplotlib” and “julia-lang” (the tag for julia program language). We notice that there is less than 3000 questions on Stack Overflow that are tagged with “julia-lang”, which implies that questions that are tagged with “julia-lang” are rare. Only the developers that are knowledgeable of “julia” are able to answer it. If the answerer were more active, this question probably would be answered much faster.

Based on what we observed in this example, we are also interested to see whether the tags that are associated with slow-answered questions are more specific and rarer than those tags that are associated with fast-answered questions. To do so, we compare the popularity (i.e., Mean_Tag_Popularity) of tags between slow-answered and fast-answered questions. Compared with the median Mean_Tag_Popularity values of slow-answered questions (i.e., 3288, 191, 34, and 89 on Stack Overflow, Mathematics, Ask Ubuntu, and Super User, respectively), we see that the median Mean_Tag_Popularity values of fast-answered questions (i.e., 6540, 279, 38, and 89.5 on these four websites) are significantly higher. The results show that slow-answered questions are usually associated with rarer tags than fast-answered questions across the four studied websites.

¹⁵<http://stackoverflow.com/questions/28553722/pyplot-error-in-julia-type-pyobject-has-no-field-set-yscale>

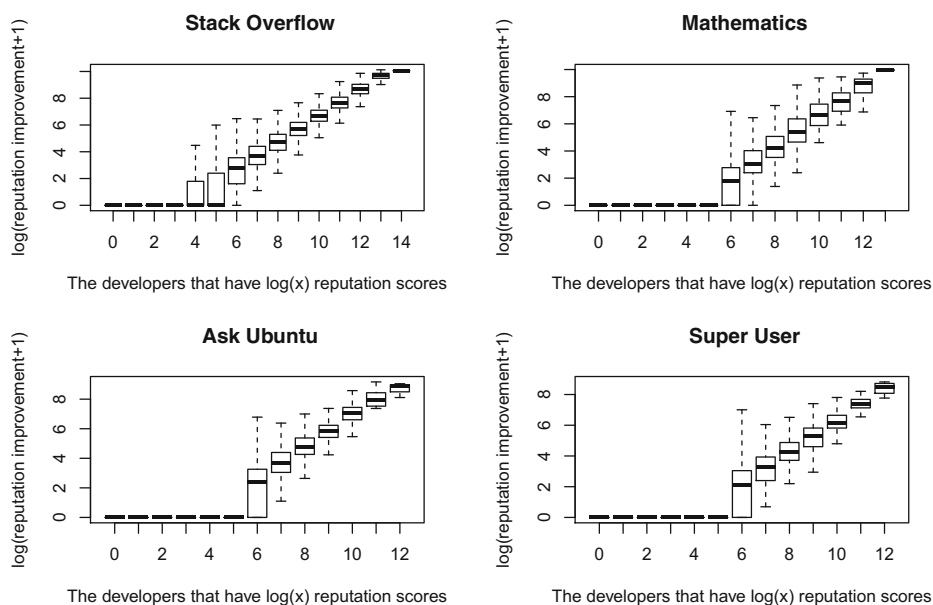


Fig. 10 The boxplots of increase in reputation scores (from March 2016 to June 2016) against developers that have $\log(x)$ reputation scores in March 2016

To further understand whether the questions that were answered by non-frequent answerers are more complex, we compare the average body length of the questions between frequent and non-frequent answerers. The results show that *the questions that were answered by non-frequent answerers are significantly longer than those that were answered by frequent answerers*, which probably implies that questions that were answered by non-frequent answerers are more complicated than the ones that were answered by frequent answerers. In other words, frequent answerers tend to answer short questions. One possible reason is that some frequent answerers prefer answering easier questions since such questions have the same reward as hard questions. This observation is also acknowledged by the Stack Overflow developers.¹⁶ Developers are complaining that some answerers may game the incentive system by always answering easy questions, and other more difficult and perhaps valuable questions would be ignored and buried by new questions. However, future studies should conduct a survey with developers in order to further understand what drives developers to answer particular questions over other questions.

As we observed, non-frequent answerers are usually the bottleneck and they are essential to the Q&A websites, since they may have unique knowledge on certain topics (Adamic et al. 2008). Thus, to help reduce the needed time to get an answer, Q&A website designers should improve the incentive system to attract non-frequent answerers to be more active and answer questions fast (e.g., rewarding the non-frequent answerers more scores if they stay online for enough time). Another suggestion is to consider improving the question answering incentive system to factor in the value and difficulty of answering questions

¹⁶ <https://meta.stackoverflow.com/questions/295688/how-to-highlight-difficult-or-old-questions-and-then-reward-the-answers>; the post received 56 up votes, 10 favorite votes, and 1,189 views

PyPlot Error in Julia: type PyObject has no field set_yscale

▲ 3 ▼ ★

I am programming in Julia but using PyPloy library. I want to plot an histogram with log y-axis. But when I use the following code:

```
using PyPlot
List = [rand() for i = 1:100]
plt.hist(List)
plt.gca().set_yscale("log")
```

I get the following error:

...

Thanks in advance.

matplotlib julia-lang

share edit edited Feb 17 '15 at 13:47 asked Feb 17 '15 at 2:29

RM- 126 • 3 • 12

1 Answer active oldest votes

▲ 2 ▼ ✓

I feel like this should be more prominently explained in the documentation, but if you scroll down to the bottom of the [Readme for PyCall](#) (which PyPlot uses) it says:

Important: The biggest difference from Python is that object attributes/members are accessed with `o[:attribute]` rather than `o.attribute`, so that `o.method(...)` in Python is replaced by `o[:method]` (...)

So, as @jverzani mentioned, after you call any module-level function from PyPlot that returns an object, that object is a PyObject and all of the attributes and methods have to be called using the bracket notation with a symbol.

share edit answered Feb 25 '16 at 13:38

Bill Gross 372 • 2 • 9

Fig. 11 An example of a domain-specific question that waited for one year to receive an accepted answer from a non-frequent answerer

(e.g., providing additional rewards to harder questions or questions that remain for long time), since frequent answerers tend to answer short questions.

In general, non-frequent answerers answer questions slower than frequent answerers and the questions that are answered by non-frequent answerers are as important (i.e., have similar range of scores) as those that are answered by frequent answerers. Such slow-answered questions would have remained unanswered if they were not answered by the non-frequent answerers. Hence Q&A website designers should improve the incentive system to attract the non-frequent answerers to be more active and improve the incentive system to factor in the value and difficulty of questions.

7 Threats to Validity

7.1 Internal Validity

One threat to internal validation relates to the categorization on our datasets, in which we consider top 20% and bottom 20% as the fast-answered questions and slow-answered questions. To address this threat, we build the regression model using different percentages of data. We consider using the top and bottom 30% and 40% of the data to build a regression model. The mean values of the TimeToGetAcceptedAnswer of slow-answered questions are 7–19.1 days and 5.1–14.3 days for 30% and 40% of the data across the four websites. We want to see if the findings are consistent when the threshold changes. Table 8 presents the results of the model built using the top and bottom 30% and 40% of the data. We highlight the top five most important factors for each website in bold font. We see that the top five most important factors are consistent in the models that are built using 20%, 30%, and 40% of data across the four studied websites. In terms of AUC, as the percentage of the data that is used to build the model increases, the AUC value decreases. This is expected, because as the data gets closer to the median boundary, the difference between the fast-answered and slow-answered questions becomes smaller. However, even with the top and bottom 40% of data, the resulting models are still reasonably good: the models achieve AUC of 0.88, 0.87, 0.76, and 0.77 on Stack Overflow, Mathematic, Ask Ubuntu, and Super User, respectively. Based on the above-mentioned results, we can conclude that our observations are not particularly sensitive to the threshold that we choose.

7.2 External Validity

It is unclear whether our findings hold for other Q&A websites under Stack Exchange or other Q&A websites. To alleviate this issue, we do the experiment on the four most popular Q&A websites under Stack Exchange. Regarding the factors that we considered, there might be additional factors that could be more relevant to the needed time to get an accepted answer for a question. However, our results show that the explanatory power of our models is very high when using the studied factors. Future studies should investigate more Q&A websites and consider more factors.

Another threat is that we did not validate our findings with the stakeholder of Stack Exchange (e.g., Stack Exchange designers and users). To alleviate this issue, we shared our findings with Stack Overflow developers and they agree with our finding that frequent answerers tend to answer easier questions and a better way is needed to motivate users to contribute to hard questions. They mention that there is no solution yet. Hence, future studies are needed. This paper is the first work that identifies the problem while providing initial insights for possible solutions for future studies to explore.

8 Related Work

In this section, we discuss related work to our paper. We focus on three closely related areas: understanding and improving question quality, connecting askers and answerers, and understanding incentive systems.

Table 8 The results of the model built based on top and bottom 30% and 40% of data

Factor	Stack overflow		Mathematics		Ask ubuntu		Super User	
The model built on top and bottom 30% of the data								
AUC	0.917		0.912		0.802		0.817	
AUC optimism	0.0006		0.0003		0.003		0.001	
	Overall NL		Overall NL		Overall NL		Overall NL	
A_Median_Speed_Answer	<i>D.F.</i> 4	3	4	3	4	3	4	3
	χ^2	21.7***	16.5***	33.4***	23.0***	38.8***	21.1***	49.3***
A_Body_Length	<i>D.F.</i> 2	1	4	3	4	3	4	3
	χ^2	6.9***	0.5***	40.0***	1.1***	8.0***	0.4	12.4***
Q_Body_Length	<i>D.F.</i> 1	–	2	1	3	2	3	2
	χ^2	5.3***		2.5***	0.2***	10.7***	1*	8.3***
Mean_Tag_Speed	<i>D.F.</i> 3	2	3	2	3	2	2	1
	χ^2	5.5***	2.6***	2.6***	2.3***	2.6	1.7	3.4***
Q_Title_Popularity	<i>D.F.</i> 2	1	1	–	1	–	1	–
	χ^2	1.1***	0.1***	0.3***		2.1***		0.5***
Median_Speed_Accepted_Answer	<i>D.F.</i> 1	–	3	2	2	1	1	–
	χ^2	0.1***	–	1.4***	1.1***	1.7***	1.2**	0.9***
Dimension								
Question	<i>D.F.</i> 17		14		14		14	
	χ^2	31.9***		9.3***		20.9***		15.1***
Asker	<i>D.F.</i> 10		13		11		10	
	χ^2	2.3***		7.3***		6.6**		3.0
Answer	<i>D.F.</i> 5		6		7		6	
	χ^2	22.1***		39.7***		20.8***		20.3***
Answerer	<i>D.F.</i> 10		8		8		9	
	χ^2	43.6***		43.6***		51.5***		61.6***
The model built on top and bottom 40% of the data								
AUC	0.876		0.872		0.756		0.769	
AUC optimism	0.0006		0.0002		0.003		0.001	
	Overall NL		Overall NL		Overall NL		Overall NL	
A_Median_Speed_Answer	<i>D.F.</i> 4	3	4	3	4	3	4	3
	χ^2	22.0***	16.7***	30.9***	20.1***	38.6***	22.7***	48.6***
A_Body_Length	<i>D.F.</i> 2	1	4	3	4	3	4	3
	χ^2	5.9***	0.6***	38.3***	0.9***	7.4***	0.6	10.4***
Q_Body_Length	<i>D.F.</i> 1	–	2	1	3	2	3	2
	χ^2	4.6***		1.6***	0.2***	9.2***	1.1**	7.4***
Mean_Tag_Speed	<i>D.F.</i> 3	2	3	2	3	2	2	1
	χ^2	4.7***	2.2***	2.1***	1.8***	2.9***	2.4***	3.0***
Q_Title_Popularity	<i>D.F.</i> 2	1	1	–	1	–	1	–
	χ^2	0.9***	0.1***	0.2***		2.6***		1.0***
Median_Speed_Accepted_Answer	<i>D.F.</i> 1	–	3	2	2	1	1	–
	χ^2	0*	–	1.2***	1.0***	2.0***	1.7***	0.7**

Table 8 (continued)

Factor		Stack overflow	Mathematics	Ask ubuntu	Super User
Dimension					
Question	<i>D.F.</i>	17	14	15	14
	χ^2	29.2***	7.6***	21.4***	14.3***
Asker	<i>D.F.</i>	10	13	11	10
	χ^2	2.4***	7.0***	5.5***	2.7*
Answer	<i>D.F.</i>	5	6	7	6
	χ^2	22.7***	40.8***	21.0***	18.7***
Answerer	<i>D.F.</i>	10	8	7	9
	χ^2	45.5***	44.3***	51.9***	64.1***

The top five most important factors and the most important dimension of each website are in bold

8.1 Understanding and Improving Question Quality

A considerable amount of work has been done on understanding and improving question quality on Q&A websites. Asaduzzaman et al. (2013) performed a study on the unanswered questions on Stack Overflow and revealed the reasons that why some questions did not receive any answer. For example, the question is too short, not clear, too hard, or not even a question for the Stack Overflow community. Asaduzzaman et al. (2013) also proposed a set of metrics that are related to the question and asker dimension to determine the time remained for an unanswered question to be answered. Rahman and Roy (2015) proposed a prediction model by employing metrics related to user behavior, topics, and popularity of a question to determine unresolved questions. Different from their work, we study the factors that affect the speed of getting an accepted answer instead of the remained time for a question to be answered. We also consider more factors and more dimensions, and we focus on studying the relationship between the answerer community and the needed time to get an accepted answer.

Ponzanelli et al. (2014c) performed an empirical study on the correlation between a set of proposed factors and the quality of a question on Stack Overflow. The authors also built a classification model to identify high-quality and low-quality questions when the questions are created. Based on the same factors, the authors also proposed an approach to detect low-quality questions on Stack Overflow (Ponzanelli et al. 2014a). Table 9 compares the most important question-related and asker-related factors between ours and that of Ponzanelli *et al.* In terms of question-related factors, the factors that are related to the length of questions are both important to the quality and the speed of getting an accepted answer for a question. Code snippets are important to the quality of a question, while they are not that important to the speed of getting an accepted answer. Regarding the asker-related factors, factors that are related to answers that an asker got previously are important to both quality and speed, which may indicate that the presentation quality of a question is important (e.g., some askers tend to ask clear or simpler questions). Based on the comparison, we observe that some factors are shared between the quality and speed of getting an accepted answer, which implies that the quality of a question is associated with the speed of getting an accepted answer.

Yao et al. (2013) found that the quality of an answer is highly correlated with that of its question. They proposed a family of algorithms to identify the quality of questions and answers based on this finding. Anderson et al. (2012) investigated the dynamics of the

Table 9 Comparison of the most important factors between the speed of getting an accepted answer and the quality of a question

Factor	Our study	Ponzanelli et al. (2014c)
Question-related factors	1) answering speed for questions that belong to different tags (i.e., Mean_Tag_Speed); 2) body length (i.e., Q_Body_Length)	1) factors about body length (i.e., word count and sentence count); 2) code ratio
Asker-related factors	1) number of previous answers (i.e., Total_Answers); 2) speed of getting accepted answers in the past (i.e., Speed_Accepted_Answer)	1) number of down votes; 2) whether previous questions received an accepted answer

community activity (e.g., answering and voting) over time and determined whether a question and its answers would continue to draw attention in the future, as well as whether a question has been sufficiently answered. Adamic et al. (2008) found that users who focus on narrow and specific topics are more likely to receive high-quality answers in Yahoo! Answers. They proposed an approach to determine the best answer by leveraging the user interests and answer characteristics. Yang et al. (2011) applied classification models (i.e., Naive Bayes, C4.5, AdaBoost, SVM) to determine whether a question on Yahoo! Answers will be answered.

Different from the above-mentioned studies, we focus on investigating the relationship between various factors (i.e., question factors, asker factors, answer factors, and answerer factors) and the speed of getting an accepted answer for a question. Our observations provide insights to Q&A website designers on how to improve user experience by shortening the waiting time for a developer to receive an accepted answer.

8.2 Connecting Askers and Answerers

A number of studies focus on user behavior and the content of discussions. Treude et al. (2011) studied how programmers ask and answer questions on Stack Overflow. Wang et al. (2013) investigated how developers help each other, and what are the most popular topics that are discussed on Stack Overflow. The most related work is from Squire (2015). She performed an empirical study to investigate whether the user support of projects should be moved to Stack Overflow from mailing lists and forums. She found that moving to Stack Overflow could speed up the answering of questions and improve the user participation. However, she did not study which factors are related to the speed of answering of questions. San Pedro and Karatzoglou (2014) presented a supervised Bayesian approach, named RankSLDA, to recommend questions for experts. Ponzanelli et al. (2014b) proposed an approach to recommend the related discussion on Stack Overflow based on the context in an integrated development environment (IDE). To help find the right answerer for a question, Wang et al. (2014) proposed an approach that combines a Bayesian inference model and a frequentist inference model for tag (i.e., topic) recommendation on three Stack Exchange websites. Xia et al. (2013) proposed an approach called TagCombine to recommend tags for a question. Linares-Vásquez et al. (2014) investigated how developers react to Android API modification on Stack Overflow and they find that the change-proneness of Android

API impacts the volume of the discussions among the developers in the Stack Overflow community.

Our paper is different from the above-mentioned studies, which mostly focus on finding the right answerer for a question. Our paper focuses on studying the relationship between the studied of factors and the time to get an accepted answer. We also analyze the answerer community and provide our suggestions to Q&A website designers.

8.3 Understanding Incentive Systems

A number of studies focus on studying the incentive system of Q&A website. Anderson et al. (2013) studied how user behavior is steered by rewards that people get when participating on Stack Overflow. They found that a badge can increase the overall level of user participation on the site and the extent of steering depends on how close the user is to the badge boundary. Cavusoglu et al. (2015) also performed an empirical study on the incentive system on Stack Overflow and provided evidence to confirm the value of the incentive system to simulate voluntary participation. Vasilescu et al. (2014) performed an empirical study to compare the user behavior in mailing lists and Stack Overflow. They found that the participants on Stack Overflow provide faster answers than on mailing lists because of the incentive system (i.e., reputation score system). Antin and Churchill (2011) analyzed the badge in social media from a psychological perspective and present five functions: goal setting, instruction, reputation, status/affirmation, and group identification.

Our paper is different from above-mentioned studies, which mostly focus on understanding the incentive system. Our study focuses on investigating the weakness of the incentive system and on providing suggestions to improve it.

9 Conclusion

Developers nowadays rely heavily on technical Q&A websites for solving problems that they face on a daily basis. Hence, any delay in getting an answer may delay the development cycle and reduce user experience when asking questions on such Q&A websites. In this paper, we study the factors that may affect the needed time for a question to get an accepted answer and provide suggestions to improve the incentive system of Q&A websites. We consider 46 factors along four dimensions (answerer, question, asker, and answer) that are potentially correlated to the needed time to get an accepted answer. We analyze four most popular technical Stack Exchange Q&A websites, including Stack Overflow, Mathematics, Super User, and Ask Ubuntu. We find that: 1) after controlling for other factors, the answerer of a question has the strongest relationship with the needed time to receive an accepted answer; 2) 61.3–86.9% of the questions that are answered by non-frequent answerers are slow-answered questions. Such slow-answered questions are usually more complex (in terms of the size of the question) than fast-answered questions but are as important as fast-answered questions. Such slow-answered questions may have remained unanswered if they were not answered by the non-frequent answerers; 3) the current incentive system does not recognize the non-frequent answerers who often answer questions which frequent answerers fail to answer; 4) the current incentive system motivate frequent answerers well, but frequent answerers tend to answer easy (in terms of size) questions. Our findings suggest that the Q&A website designers should improve their incentive system to motivate non-frequent answerers to be more active on answering questions and to answer questions fast, in order to shorten the waiting time to receive an answer. The Q&A website designers

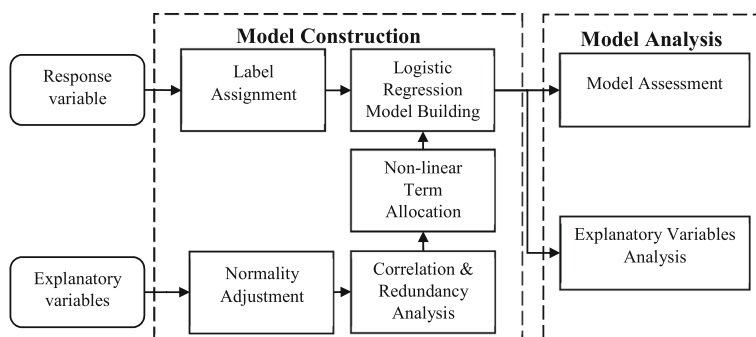


Fig. 12 An overview of our model construction and analysis approaches

should also improve the question answering incentive system to factor in the value and difficulty of answering the questions (e.g., providing additional rewards to harder questions or questions that remain unanswered for long time).

Future studies should validate our findings through user surveys among answerers. For example, whether answerers are motivated by the current incentive system on Stack Exchange, and which factors cause delayed answers. Future studies should also perform user surveys to understand the acceptable waiting time for getting an accepted answer. As noted by Zhou and Mockus (2011), developers' participation in a project is correlated with social factors (e.g., communication network). Future studies should also study how social factors could potentially encourage such non-frequent answerers to become more active.

Appendix A: Model Building and Analysis Process

In this appendix, we present the detail of our model building process.

Figure 12 shows an overview of our model building process. We use the R package **rms**¹⁷ as the implementation of our logistic regression model. Below, we describe the detailed steps of our model building process.

1. Label Assignment Since we use a classification model to understand the impact of the studied factors on the speed. We first need to select the questions that are used to build the model and assign the label (i.e., fast-answered question or slow-answered question) to these questions.

As the results shown in Section 5, more than half of the questions were answered within one hour. Thus, the needed time to answer a question is very close (i.e., within minutes) for most questions. Such skewness in the data will have a negative impact on the resulting model (i.e., increase bias).

Figure 13 presents the percentage of the questions that are received in the time window that are around the median cut-off point (i.e., median of TimeToGetAcceptedAnswer). We see that the number of questions that is around the median cut-off point is notably large. For example, 10.8% (6027) of the questions receive an accepted answer within a time window of 5 minutes less or larger than the median value of TimeToGetAcceptedAnswer on Stack

¹⁷<https://cran.r-project.org/web/packages/rms/index.html>

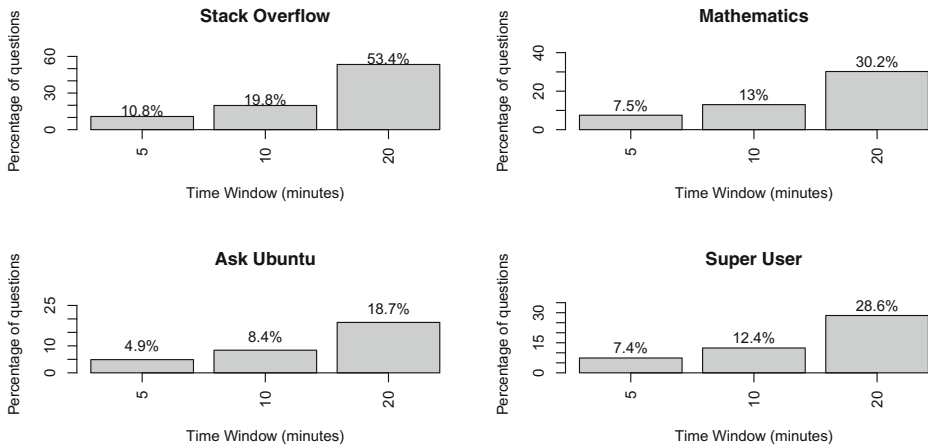


Fig. 13 The percentage of questions that received answers within time window of median ($\text{TimeToGetAcceptedAnswer} \pm x$ minutes)

Overflow. If we loosen the time window to 20 minutes, 53.4% (29,834) of the questions receive an accepted answer in 20 minutes less or larger than the median value of $\text{TimeToGetAcceptedAnswer}$. In other word, more than half of the questions on Stack Overflow land on the boundary, which probably could result in having a large amount of noise in our built model.

To reduce such noise, we sort the questions based on their needed time to get an accepted answer, and then label the top 20% of questions as the fast-answered questions and bottom 20% of questions as the slow-answered question. This approach intuitively fits with goals of our study (studying the speed of answering where a few minutes difference should not be used to distinguish between a fast-answered question and a slow-answered question). The mean values of $\text{TimeToGetAcceptedAnswer}$ of two groups are shown in Table 10. We could observe that the fast-answered questions were answered within 0.1 hours on average, while slow-answered questions needed at least 10 days to be answered.

2. Normality Adjustment When building a logistic regression model, the model prefers the explanatory variables to be normally distributed in order to produce a more stable and robust model (Freedman 2005). In our case, most of the studied factors are skewed. All studied factors are considered as highly skewed (i.e., the skewness is larger than 1) (Bulmer 1979) except for `Tag_Level_Difference`, `Tag_Number`, `Q_Title_Popularity`,

Table 10 The comparison of mean values of $\text{TimeToGetAcceptedAnswer}$ between fast-answered and slow-answered questions

Website	Fast-answered questions (hours)	Slow-answered questions (hours)
Stack Overflow	0.06	245.5
Mathematics	0.09	320.6
Ask Ubuntu	0.1	687.9
Super User	0.08	623.8

Mean_Down_Votes, Median_Down_Votes, and Sum_Down_Votes. Therefore, we apply a logarithm transformation $[\ln(x + 1)]$ to all the studied factors to reduce skewness.

3. Correlation & Redundancy Analysis We remove correlated and redundant factors using the following steps: i) removing factors with zero variance; ii) removing highly correlated factors; iii) and removing redundant factors.

We first remove factors with zero variance, since these factors do not have any contribution to the model. For example, the variance of Median_Down_Votes of Super User is 0, which indicates the value of Median_Down_Votes of the studied Super User data (top 20% and bottom 20%) is unique (i.e., 0 in this case).

Highly correlated factors can cause multicollinearity problems in our model. Thus, we perform a correlation analysis to remove highly correlated factors using a variable clustering analysis technique by following prior studies (Thongtanunam et al. 2016; McIntosh et al. 2016). We construct a hierarchical overview of the correlation among the factors and select one factor from each cluster of highly-correlated variables, i.e., $|\rho| > 0.7$ (Thongtanunam et al. 2016).

After this step, there remains 28, 28, 26, and 27 factors in the Stack Overflow, Mathematics, Ask Ubuntu, and Super User data, respectively (see the remained factors at Table 4).

Correlation analysis reduces multicollinearity among the factors, but it may not detect all of the redundant factors (i.e., factors that do not have a unique signal relative to the other factors). We remove redundant factors by using the *redun* function in the R package **rms**¹⁸ with the default R^2 threshold of 0.9. However, no factors were removed in this step. The final factors are presented in Table 4.

4. Non-linear Term Allocation When building a logistic regression model, some factors potentially share non-linear relationships with the response variable. However, logistic regression models are mainly used for modeling linear relationships. Thus, we use restricted cubic splines (Harrell 2006) to add the non-linear terms of factors into the model by following prior studies (Thongtanunam et al. 2016; McIntosh et al. 2016). We measure the non-linear relationship by calculating the Spearman multiple ρ^2 between the dependent variable y and linear and quadratic forms of each factor (x_i, x_i^2). A large ρ^2 indicates that there is a high chance for a non-linear relationship between a factor and the response variable, which indicates that the factor should be assigned a larger degree of freedom. By observing the rough clustering of the factors according to their ρ^2 , we cluster the factors into four groups according to the Spearman multiple ρ^2 values across the four websites (see Figure 14). We give factors in the first, second, and third groups five, four, and three degrees, respectively.

5. Logistic Regression Model Building Finally, after selecting the factors and specifying the non-linear terms of the factors, we build our regression models using the preprocessed data. When building the model, we consider text-related question factors, asker factors, answer factors as control variables by including it in the model; an approach that is commonly used in regression models (Miller and Han 2001; Bird et al. 2011; Chen et al. 2012). We use the function **lrm** in the R package **rms** as the implementation of logistic regression model and use the **rccs** function in **rms** as the implementation of restricted cubic splines.

¹⁸<https://cran.r-project.org/web/packages/rms/index.html>

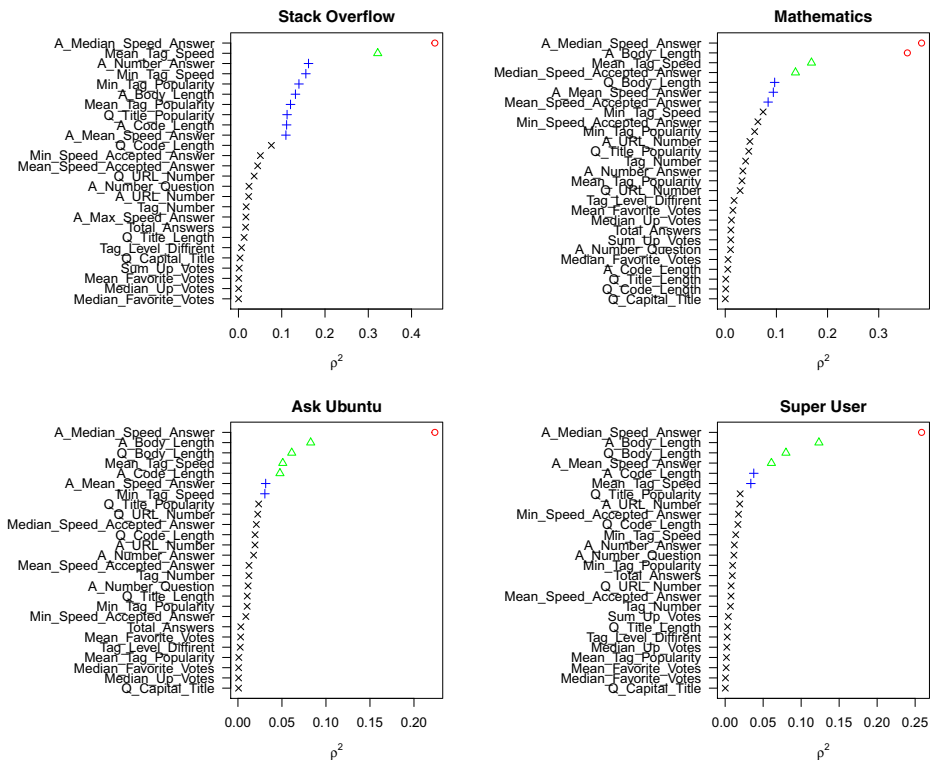


Fig. 14 Dotplot of the Spearman multiple ρ^2 of each factor in the four studied websites. The larger the ρ^2 value, the more likely the factor has a non-linear relationship with the response variable. The first, second, and third groups of factors (categorized by the ρ^2 value) are highlighted with red circle, green triangle, and blue plus, respectively

6. Model assessment We use AUC and bootstrapping to assess the explanatory power of the logistic regression model (i.e., ability of the model to capture the relationship between the explanatory variables and the response variable). AUC is the area under the Receiver Operating Characteristic (ROC) curve (Han 2005). The area under ROC curve is often used as a measure of the quality of classification models. A random classifier has an AUC of 0.5, while the AUC for a perfect classifier is equal to 1. In practice, most of the regression models have an AUC between 0.5 and 1.

Since AUC can be an overestimation (i.e., higher than it actually is) if the model is overfitted to the data, we further evaluate the stability of our model. Similar to prior work (McIntosh et al. 2016; Thongtanunam et al. 2016), we reduce such overestimation by using a bootstrap-derived approach (Efron 1986). The steps of the bootstrap-derived approach are listed below:

1. From the original dataset with n records (i.e., 55,853, 70,336, 7,134, and 10,776 for Stack Overflow, Mathematics, Ask Ubuntu, and Super User, respectively), select a bootstrap sample, i.e., a random sample of n records with replacement.
2. In the bootstrap sample, we build a model using the same allocation of knots as was used in the original dataset.

3. Apply the model that is built using the bootstrap sample on the bootstrapped and the original datasets. We calculate the AUC for each model.
4. The optimism is the difference in the AUC of the bootstrap sample and the original sample. Note that optimism is not an absolute value. A positive sign indicates that AUC of the original sample is larger than that of the bootstrap sample; a negative sign indicates that AUC of the bootstrap sample is larger than that of the original sample.

The above process is repeated 1,000 times and the average (mean) optimism is calculated. Small optimism values indicate that the model does not suffer from overfitting.

7. Explanatory Variables Analysis After our model assessment step, if the AUC value is high and the optimism value is low (i.e., our model can explain the TimeToGetAcceptedAnswer well with low bias), we can then use the model to study the impact of each factor on the TimeToGetAcceptedAnswer. We measure the impact of each factor on the TimeToGetAcceptedAnswer using the Wald χ^2 test (Chambers 1991). The Wald χ^2 test is commonly used in biostatistic (Harrell 2006) and software engineering (McIntosh et al. 2016; Thongtanunam et al. 2016) research to understand the impact of factors in a model.

References

- Adamic LA, Zhang J, Bakshy E, Ackerman MS (2008) Knowledge sharing and yahoo answers: everyone knows something. In: Proceedings of the 17th international conference on world wide web, WWW '08, pp 665–674
- Aizawa A (2003) An information-theoretic perspective of tf-idf measures. *Inf. Process. Manage.* 39(1):45–65
- Anderson A, Huttenlocher D, Kleinberg J, Leskovec J (2012) Discovering value from community activity on focused question answering sites: a case study of stack overflow. In: Proceedings of the 18th ACM SIGKDD international conference on knowledge discovery and data mining, KDD '12, pp 850–858
- Anderson A, Huttenlocher D, Kleinberg J, Leskovec J (2013) Steering user behavior with badges. In: Proceedings of the 22nd international conference on world wide web, WWW '13, pp 95–106
- Antin J, Churchill EF (2011) Badges in social media: a social psychological perspective. In: CHI 2011 Gamification workshop proceedings (Vancouver, BC, Canada, 2011)
- Asaduzzaman M, Mashiyat AS, Roy CK, Schneider KA (2013) Answering questions about unanswered questions of stack overflow. In: Proceedings of the 10th working conference on mining software repositories, MSR '13, pp 97–100
- Bird C, Nagappan N, Murphy B, Gall H, Devanbu P (2011) Don't touch my code!: Examining the effects of ownership on software quality. In: Proceedings of the 19th ACM SIGSOFT symposium and the 13th european conference on foundations of software engineering, pp 4–14
- Briggs A, Clark T, Wolstenholme J, Clark P (2003) Missing... presumed at random: cost analysis of incomplete data. *Health Econ* 12(5):377–393
- Bulmer M (1979) Principles of statistics. Dover books on mathematics series. Dover Publications, New York
- Cavusoglu H, Li Z, Huang K-W (2015) Can gamification motivate voluntary contributions?: the case of stackoverflow Q&A community. In: Proceedings of the 18th ACM conference companion on computer supported cooperative work & social computing, CSCW'15 companion, pp 171–174
- Chambers JM (1991) Statistical models in S. CRC Press, Inc., Boca Raton
- Chen T-H, Thomas SW, Nagappan M, Hassan AE (2012) Explaining software defects using topic models. In: Proceedings of the 9th IEEE working conference on mining software repositories, MSR '12, pp 189–198
- Cliff N (1993) Dominance statistics: ordinal analyses to answer ordinal questions. *Psychol Bull* 114:494–509
- Colburn L (2016) SLI systems granted auto complete patent. <https://www.sli-systems.com/sli-systems-granted-auto-complete-patent>
- Cornea R, Weininger N (2014) Providing autocomplete suggestions. US Patent 8,645,825
- Dunn OJ (1961) Multiple comparisons among means. *American Statistical Association* 56:52–64
- Efron B (1986) How biased is the apparent error rate of a prediction rule? *J Am Stat Assoc* 81(394):461–470
- Freedman D (2005) Statistical models: theory and practice. Cambridge University Press, Cambridge
- Han J (2005) Data mining: concepts and techniques. Morgan Kaufmann Publishers Inc, San Francisco

- Harrell FE Jr (2006) Regression modeling strategies. Springer-Verlag New York, Inc., Secaucus
- Linares-Vásquez M, Bavota G, Di Penta M, Oliveto R, Poshyanyk D (2014) How do api changes trigger stack overflow discussions? A study on the android sdk. In: Proceedings of the 22nd international conference on program comprehension, ICPC 2014, New York, NY, USA, pp 83–94
- McIntosh S, Kamei Y, Adams B, Hassan AE (2016) An empirical study of the impact of modern code review practices on software quality. *Empir Softw Eng* 21(5):2146–2189
- Miller HJ, Han J (2001) Geographic data mining and knowledge discovery. Taylor & Francis Inc, New York
- Mockus A (2008) Missing data in software engineering. Springer, London, pp 185–200
- Moore D, Maccabe G, Craig B (2009) Introduction to the practice of statistics. W.H. Freeman and Company, San Francisco
- Ponzanelli L, Mocci A, Bacchelli A, Lanza M, Fullerton D (2014a) Improving low quality stack overflow post detection. In: 30th IEEE international conference on software maintenance and evolution, Victoria, BC, Canada, September 29–October 3, 2014, pp 541–544
- Ponzanelli L, Bavota G, Di Penta M, Oliveto R, Lanza M (2014b) Mining stackoverflow to turn the IDE into a self-confident programming prompter. In: Proceedings of the 11th working conference on mining software repositories, MSR '13, pp 102–111
- Ponzanelli L, Mocci A, Bacchelli A, Lanza M (2014c) Understanding and classifying the quality of technical forum questions. In: 14th international conference on quality software, pp 343–352
- Rahman MM, Roy CK (2015) An insight into the unresolved questions at stack overflow. In: Proceedings of the 12th working conference on mining software repositories, MSR '15, pp 426–429
- San Pedro J, Karatzoglou A (2014) Question recommendation for collaborative question answering systems with RankSLDA. In: Proceedings of the 8th ACM conference on recommender systems. ACM, pp 193–200
- Squire M (2015) Should we move to stack overflow? Measuring the utility of social media for developer support. In: Proceedings of the 37th international conference on software engineering, pp 219–228
- StackOverflow (2016) Developer survey results 2016. <http://stackoverflow.com/research/user-survey-2016>
- Thongtanunam P, McIntosh S, Hassan AE, Iida H (2016) Revisiting code ownership and its relationship with software quality in the scope of modern code review. In: Proceedings of the 38th international conference on software engineering, pp 1039–1050
- Treude C, Barzilay O, Storey M-A (2011) How do programmers ask and answer questions on the web? (nier track). In: Proceedings of the 33rd international conference on software engineering, pp 804–807
- Vasilescu B, Serebrenik A, Devanbu P, Filkov V (2014) How social Q&A sites are changing knowledge sharing in open source software communities. In: Proceedings of the 17th ACM conference on computer supported cooperative work & social computing, CSCW '14, pp 342–354
- Wang S, Lo D, Jiang L (2013) An empirical study on developer interactions in stackoverflow. In: Proceedings of the 28th annual ACM symposium on applied computing, SAC '13, Coimbra, Portugal, March 18–22, 2013, pp 1019–1024
- Wang S, Lo D, Vasilescu B, Serebrenik A (2014) Entagrec: an enhanced tag recommendation system for software information sites. In: Proceedings of the international conference on software maintenance and evolution, pp 291–300
- Xia X, Lo D, Wang X, Zhou B (2013) Tag recommendation in software information sites. In: Proceedings of the 10th working conference on mining software repositories, MSR '13, pp 287–296
- Yang L, Bao S, Lin Q, Wu X, Han D, Su Z, Yu Y (2011) Analyzing and predicting not-answered questions in community-based question answering services. In: Proceedings of the twenty-fifth AAAI conference on artificial intelligence, AAAI'11. AAAI Press, pp 1273–1278
- Yao Y, Tong H, Xie T, Akoglu L, Xu F, Lu J (2013) Want a good answer? Ask a good question first! arXiv:1311.6876
- Zhou M, Mockus A (2011) Does the initial environment impact the future of developers? In: Proceedings of the 33rd international conference on software engineering, ICSE '11, pp 271–280



Shaowei Wang is a Postdoc in the Software Analysis and Intelligence (SAIL) Lab at Queen's University, Canada. He obtained his PhD from Singapore Management University, and BSc from Zhejiang University. His research interests include code mining and recommendation, software maintenance, developer forum analysis, and mining software repositories. He has served as a reviewer of a number of high-quality journals (e.g., IEEE Transaction on Software Engineer, Empirical Software Engineering). More information at: <https://sites.google.com/site/wswshaoweiwang/>.



Tse-Hsun Chen is an Assistant Professor in the Department of Computer Science and Software Engineering at Concordia University, Montreal, Canada. He obtained his BSc from the University of British Columbia, and MSc and PhD from Queen's University. Besides his academic career, Dr. Chen also worked as a software performance engineer at BlackBerry for over four years. His research interests include performance engineering, database performance, program analysis, log analysis, testing, and mining software repositories. Early tools that are developed by Dr. Chen are integrated into industrial practice for ensuring the quality of large-scale enterprise systems. More information at: <http://petertsehsun.github.io/>.



Ahmed E. Hassan is the Canada Research Chair (CRC) in Software Analytics, and the NSERC/BlackBerry Software Engineering Chair at the School of Computing at Queen's University, Canada. His research interests include mining software repositories, empirical software engineering, load testing, and log mining. Hassan received a PhD in Computer Science from the University of Waterloo. He spearheaded the creation of the Mining Software Repositories (MSR) conference and its research community. Hassan also serves on the editorial boards of IEEE Transactions on Software Engineering, Springer Journal of Empirical Software Engineering, Springer Journal of Computing, and PeerJ Computer Science. Contact ahmed@cs.queensu.ca. More information at: <http://sail.cs.queensu.ca/>.