Calculation of degree of difficulty of questions in Q & A communities

Dhal, Asit Kumar
Multimedia Communications Lab
Dept. of Electrical Engineering and Information Technology
TU Darmstadt
dhal.asitk@gmail.com

Abstract

Classification is a task in which different models are used to classify objects based on the characteristics of the objects. This is used by scientific community to model data in the domains like medicine, economy, weather forecasting, etc. In this lab, I am using different classification models to classify stackoverflow questions to determine the difficulty level. Many researchers have studied the classification of questions in different types of Q & A (Question and Answering Communities) systems by using numerical features. In this lab, I am using only textual attributes to classify questions using different well known models. I will prove textual features can be used to find difficulty level by using three well known text classification models and show the comparison of evaluation of different classification models.

1. Introduction

Stackoverflow is a Q & A system where users ask questions and other users, based on the understanding and skill, try to answer the question. Stackoverflow has a reward system which gives some reputation points based the quality of questions and quality of answers. Over the years, the system has become a large repository of valuable knowledge which is being used by many users.

The reward system is be used to enrich the user experience by giving points if the user answers a question and the asker accepts the answer, positive or negative points if the question/answer gets a vote up/down and the question/ answer is marked as a favorite question/answer by some user. Finding the difficulty level of a question can also be used find the skill set of the user who answers the question. This can be used to route the questions to the users of appropriate skill level. This can help the askers to get a quick answer and keep the easy questions away from a skilled user. The reward system can be modified to give more points if the user answers a difficult question. This can lead to better user experience. By finding users of higher skill level can be monetized for proper job advertisement(Stackoverflow careers).

In this lab, I am using the textual attributes along with only one numerical attribute for determining difficulty level of questions. I am using three well established classification models(Naive Bayes, Support Vector Machine and J48 Decision Tree) to classify questions as easy and difficult. I will compare the performance of these three algorithms and conclude with my argument to support the best among the three algorithms.

2. Existing Works

Researchers have done many experiments and built models to estimate the difficulty level of questions in Q & A systems[1]. The researchers have built competition based models for classification. In order to learn relative skill of different users, the researchers built True Skill model. True Skill is a Bayesian skill rating model[1]. Page rank is another model in which a user answers question x and fails to answer question y. So, a task graph is created between x and y and page-rank is applied on the task graph to build the model. Both of these models fail for a new user or an existing user asking a question in an unrelated field. So, textual features of the question, which is not associated with previous user activity, can give an acceptable result.

3. The System Design

In order to classify the questions, I am using WEKA machine learning tool. My application uses weka library in Java and publicly available stackoverflow dump. Figure 1 shows the overall system design.

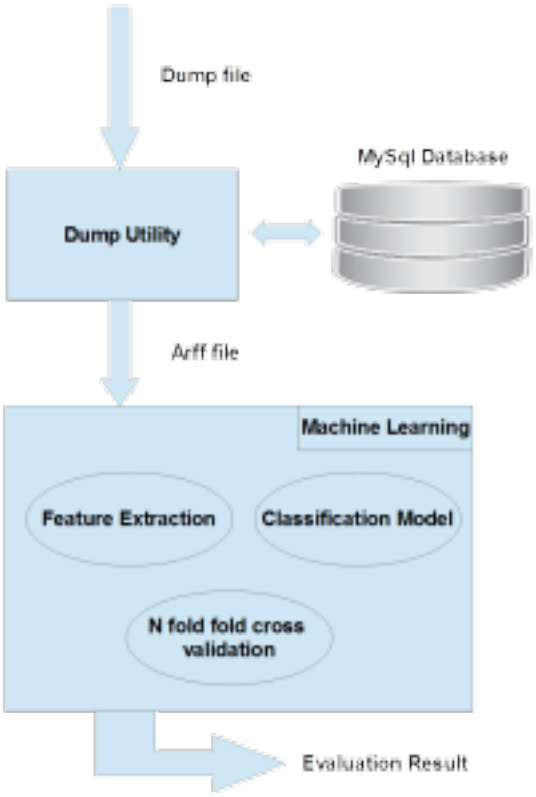

Figure 1. Overall System Design

The system has two parts.

1. The first part is loading XML dump file from stackoverflow to mysql database and generate arff file. The arff file will contain all textual attributes, percentage of lines of code, user reputation and difficulty level.

2. The second part is building classification models and evaluation of those models.

4. Data collection

Stackoverflow dumps are publicly available. I am considering only questions asked between 2013 and 2014 and all users in stackoverflow. Following two files are being considered for analysis.

1. Posts.xml : All questions and the corresponding answers.

2. Users.xml : All users registered in stackoverflow.

Following preprocessing is done for all questions.

1. The questions contained html tags for proper rendering. I removed those html tags. Those are not part of the question.

2. Some questions contained sample code as a part of question. The sample codes were removed.

3. All questions are leveled with initial difficulty level of easy or difficult based on the following two conditions.

   (a) easy questions: questions which have an accepted answer within 3 days

   (b) difficult questions: questions which don't have an accepted answer and no downvotes.

4. Lines of code(LOC) is defined as number of lines of code present in the question. LOC percentage is calculated by using following formula[6].

$$M_{loc} = loc \times 100/(loc + num) \quad (1)$$

   where num = total number of sentences

5. I considered user reputation as the only numerical attribute for classification. This is based on the fact that if the user with higher reputation score asks a question, it's going to be a difficult question. If the user with less score asks a question, it's going to be an easy question.

To make the analysis quicker, I considered three questions posted for three different tags python, java and javascript. Total 10000 records for each tag were selected for analysis.

5. Arff File Generation

The dump tool generates an arff file which contains the following attributes.

Attribute Name	Attribute Type
Question Body	String
Question Title	String
LOC percentage	Numeric
User Reputation	Numeric
Difficulty Level	Nominal(Easy or Difficult)

6. Machine Learning

This part of the system basically does the following three things.

1. Feature Engineering

2. Classification

3. Evaluation

The detailed diagram of the system is shown in the figure 2.

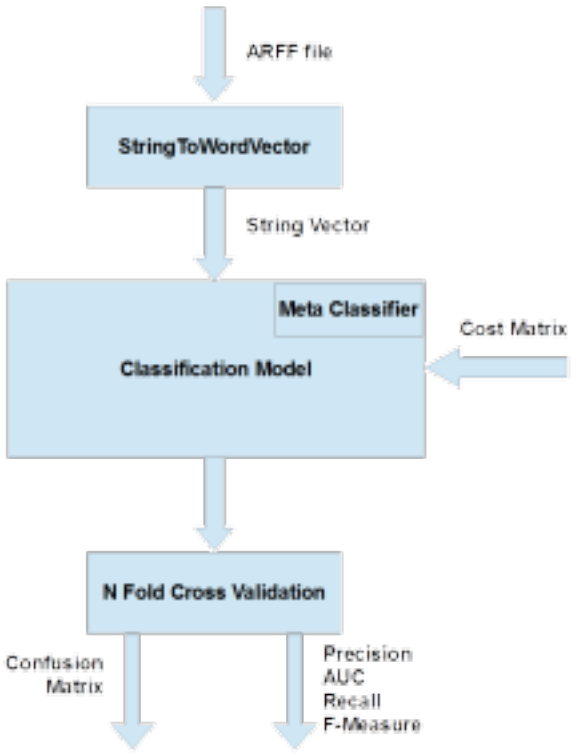

Figure 2. Detailed System Design

6.1 Feature Engineering

All the three classification models(Naive Bayes, SVM and J48) can work on numbers. So, I transformed both body and title of the questions into terms of vectors. Terms of vectors are basically represented by[2]

$$V_d = (w_1, w_2, ..., w_n) \tag{2}$$

where

$$t_i = weight\_of\_term_i[3]. \tag{3}$$

I used weka to transform all texts. For simplicity, I considered the following weka options.

1. lower case: all words are converted to lower case
2. alphabetic tokenizer: sentences are converted to words by considering space, comma, full stop, etc.
3. keep all words: all words are considered
4. word count: word count are considered, not the presence or absence of words

Some researchers have already done research about presence important words can be deciding factor in determining the difficulty level of questions[2]. I tried to find out how term frequency and inverse document frequency of different words. Term frequency is the frequency of words in a document. The importance of each word is assumed to b e inversely proportional to the number of texts that contain the word [3]. The inverse document frequency(IDF) factor of a word t is given by

$$IDF(t) = log(N/df(t)) \tag{4}$$

Where N is the total number of questions, df(t) is the number of questions that contains the word t. I found that if IDF of some word greater than some value, that word is an important word. For a very small set of questions, followings are the important words for python, java and javascript.

Python	modifier, pipeline, tcl, sublime, proxy, webcam, pyside, drive, mapping, south, sphinx, png, scrapy, entity, nodes, production, stream, distance, picture, selenium, zip, 15, bash
Java	abstractreflectionconverter, databind, deser, deserializercache, eventqueue, fasterxml, radicadoof, repaintmanager, synthtableui, thoughtworks, treeunmarshaller
Javascript	paypal, customer, lightbox, rectangles, grunt, tinymce, fancybox, dojo, tooltip, release, transition, xhr, accordion, horizontal, localhost, nodes, sign, contact, fb, media, meteor, blue

In order to force the classification model to give importance to these words, I added an weight attribute to each question. The weight is calculated as

$$Weight_{question} = 0.7 \times f \tag{5}$$

where f is frequency of important word in that question.

So, after string transformation and term weighting the arff file is converted to the following formats.

Attribute Name	Attribute Type
Word 1	Numeric
Word 2	Numeric
.....	Numeric
Word n	Numeric
Weight	Numeric
LOC percentage	Numeric
User Reputation	Numeric
Difficulty Level	Nominal(Easy or Difficult)

6.2 Classification Model

In order to maximize classification accuracy, adjustments are made to take into account the costs of misclassification, adjustments must be made to the learning process. Weka has a cost sensitive classifier which wraps the main classification model. I am using cost matrix to penalize misclassification. Following is the cost matrix for this lab.

0.0	9.0
30.0	0.0

[0.0 9.0; 30.0 0.0]

I used the following three classification models.

6.2.1 Naive Bayes

Naive Bayes classification model is a probabilistic classification model based on Bayes' theorem with strong (naive) independence assumptions between the features[3].
For this lab, I used kernel density estimator(-K) option in weka to build classification model.

6.2.2 SVM

SVM or support vector machine classification model is a supervised learning models with associated learning algorithms that analyze data and recognize patterns, used for classification and regression analysis. It constructs a hyperplane or set of hyperplanes in a high- or infinite-dimensional space, which can be used for classification, regression, or other tasks. Intuitively, a good separation is achieved by the hyperplane that has the largest distance to the nearest training data point of any class (so-called functional margin), since in general the larger the margin the lower the generalization error of the classifier[4].
In weka, SMO is used to build svm models. For this lab, I used the following parameters.

-C 1.0 -L 0.0010 -P 1.0E-12 -N 0 -V -1 -W 1

6.2.3 J48

J48 builds decision trees from a set of training data in the same way as ID3, using the concept of information entropy. At each node of the tree, C4.5 chooses the attribute of the data that most effectively splits its set of samples into subsets enriched in one class or the other. The splitting criterion is the normalized information gain (difference in entropy). The attribute with the highest normalized information gain is chosen to make the decision. The C4.5 algorithm then recurs on the smaller sublists.[5]

6.2.4 Cross Validation

I used 10-fold cross validation for building the classification model.

7. Evaluation

The result for questions with python tags for 3 models are shown below.

	TP Rate	FP Rate	F-Measure	ROC Area
Easy-SVM	0.783	0.699	0.773	0.543
Easy-Naive Bayes	0.85	0.626	0.0.824	0.649
Easy-J48	0.741	0.756	0.742	0.49
Difficult-SVM	0.303	0.217	0.312	0.543
Difficult-Naive Bayes	0.374	0.15	0.412	0.65
Difficult-J48	0.244	0.63	0.615	0.49

The result for questions with javascript tags for 3 models are shown below.

	TP Rate	FP Rate	F-Measure	ROC Area
Easy-SVM	0.786	0.665	0.788	0.56
Easy-Naive Bayes	0.749	0.514	0.791	0.657
Easy-J48	0.717	0.791	0.73	0.471
Difficult-SVM	0.335	0.214	0.332	0.56
Difficult-Naive Bayes	0.486	0.251	0.409	0.657
Difficult-J48	0.209	0.283	0.198	0.471

8. Application

The application I developed is a java based command line application. In order to use, the user either has to pass command line parameters or modify the config section of build.xml.

The application uses MySql as the relational database and ant as the build script.

8.1 Advantages

The application has following advantages.

1. It can be used other xml dumps from stackoverflow with minimal code changes.
2. More classification models can be added with minimal code changes if weka supports those models.
3. Most classification models can be optimized by using options provided by weka.

8.2 Disadvantages

The application has following disadvantages.

1. It crashes for huge amount of data.
2. J48 and SVM are not properly optimized.

9. Conclusion

In this lab, I found out textual information of a question can be used to determine the difficulty level of a question. The features frequency of important words and percentage of lines of code along with user reputation can determine the difficulty level. I represented evaluation result of three classification models Naive Bayes, SVM and J48 Decision Tree. From the experiments, I found that Naive Bayes performs better than SVM and J48. SVM is very slow in while dealing with huge amount of data.

References

[1] Jing Liu, Quan Wang, Chin-Yew Lin, Hsiao-Wuen Honn: Question Difficulty Estimation in Community Question Answering Services. Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, Seattle, Washington, USA, 18-21 October 2013.

[2] Takenobu Tokunaga, Makoto1 Iwayama: Text categorization based on weighted inverse document frequency.Department of Computer Science Tokyo Institute of Technology, March 1994

[3] Naive Bayes : `http://en.wikipedia.org/wiki/Naive_Bayes_classifier`

[4] Support vector machine: `http://en.wikipedia.org/wiki/Support_vector_machine`

[5] J48: `http://en.wikipedia.org/wiki/C4.5_algorithm`

[6] Luca Ponzanelli, Andrea Mocci, Alberto Bacchelli, Michele Lanza: Understanding and Classifying the Quality of Technical Forum Questions. Faculty of Informatics Universit della Svizzera italiana Lugano, Switzerland, June 2014