Automatic Tag Prediction For Stack Overflow Questions

SMAI Project

Akhil Batra
Center for Exact Humanities
IIIT Hyderabad, India
akhil.batra@research.iiit.ac.in

SVK Rohit
Center for Exact Humanities
IIIT Hyderabad, India
rohit.sakala@research.iiit.ac.in

U Nithiya Shree
Center for Cognitive Science
IIIT Hyderabad, India
unithiya.shree@research.iiit.ac.in

***Abstract*—Nowadays, it is very clear that software engineers use a variety of online media to search and become informed of new and interesting technologies, and to learn from and help one another. So, stack overflow is one such platform available. Users increasingly use tags to describe the most important features of their posted contents or projects. Right now, it is done manually and we intend to make it automated by creating a model using the data in the stack overflow to predict the tag for the questions posted. This will be very helpful to the community of stack overflow because it can send notifications to the domain experts regarding the question and there are many more advantages to it.**

In this paper, we are using the stack overflow dataset as the training data and testing data. We use TfidfVectorizer to convert the collection of questions into features and then we reduce the dimensionality of the sparse matrix using chi2 which will give us the best n features and we then reduce the dimensionality of n to k by truncated svd. We finally get the dataset which has most of the information and apply various classification algorithms like SVM , Kernel SVM, Naive Bayes to predict. The accuracies are as Linear SVM @31.4%, kernel SVM (RBF) @ 39%, kernel SVM (Sigmoid) @ 42.1%, Gaussian Naive Bayes @ 34%.

Keywords—SVM; Kernel; Tfid; chi2; Stack overflow; tags;recommendation; Sigmoid;

I. INTRODUCTION (*Heading 1*)

In information systems, tagging is a popular way to categorize information and to search content. Therefore, almost all online newspapers, blogs, question-answer communities, and other similar sites make use of tags to categorize articles, posts, questions, answers, and so on. Similarly, Stack overflow uses tags to categorize programming questions so that their users can find similar questions on the same topic or find questions that they may be able to answer. On Stack Overflow, a questioner can add up to five tags to categorize a question using existing tags or using new tags they create. However, a user must have a certain level of reputation to create a new tag, which is determined by the site using a reputation score. Appropriate tagging of questions may be useful to get a quick answer because potential responders are able to be notified when a question is posted with a tag related to their interests. In order to investigate the current state of tagging questions on Stack Overflow, we first determined how fully questions are tagged. Our results (in Table I) show that almost 87% of the questions have less than five tags and 38.38% of questions have only one or two tags. We further investigated whether the number of tags and the number of times a question is viewed are related, since the chance of getting an answer may improve the more a question is viewed by the community. To this end, we calculate the average number of views per question for each group and plotted them on a graph, where the x-axis represents the number of tags and the y-axis represents the number of views per question (cf. Figure 1).

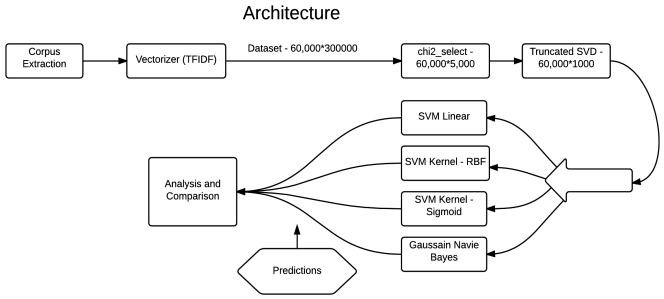
TABLE 1
Proportion of questions by number of tags

No of Tags	No of Questions	Percentage
1	438475	12.07
2	887037	25.68

3	997906	28.89
4	693234	20.07
5	437090	12.66

A questioner can mark one of the answers to their question as ‘accepted’ when they consider the answer to be the most helpful for them personally. 62% of questions with one tag had accepted answers whereas 68% of questions with five tags had accepted answers. The increased viewing by number of tags and the increased accepted answers by number of tags motivated us to build an automatic system for tagging Stack Overflow questions appropriately.

II. Our Approach



A. *Designing Dataset*

We have chosen to test on the top 1000 tags in the stack overflow dataset. Now, we have 60 questions for each tag and used these 60,000 questions as our training data and also we did cross validation on this.

B. *Vectorizing data*

We have used TFIDF Vectorizer to convert our data into a feature vector. TFIDF computes the frequency of a word occurring and gives it a value based on it. We remove stop words while doing performing this. Each dimension represents a word in the question which might turn into a tag in future. Now, using this we get a sparse matrix filled with zeros. We have convert into a matrix with lesser dimension having useful information without the loss of variance between features. By using a sparse matrix a classifier will not yield good results. As seen from the figure , this converts the data into a matrix of size 60,000 * 3,00,000 (feature vector).

C. Reducing dimensionality

Now, we try to reduce the dimension of the matrix that is convert the sparse matrix. We tried using FASTICA as it is the fastest technique to convert but due to memory constraints we could not. We have chi squared to select the best k features on decomposing the data into chi square statistical distribution. Initially we had got approx 300000 features , out of which we will be selecting 5000 best features using chi2. Later, we use truncatedSVD to decompose the 5000 best features into 1000 features as SVD doesn't make new projection matrices rather apply transformation on the given sparsed matrix. Thus, benefiting us to co-operate with constrained information.

D. Classifiers

1) Linear SVM

SVM is a linear discriminant function with support vectors which classifies the data point into two classes, but here we have 1000 classes, so multi class classifier is to be implemented. Now, we ran linear svm on the dataset with a lower bound of 0.5 ratio of support vector with L1 and L2 penalty. Accuracy is **31.4** percentage with a five fold cross validation.

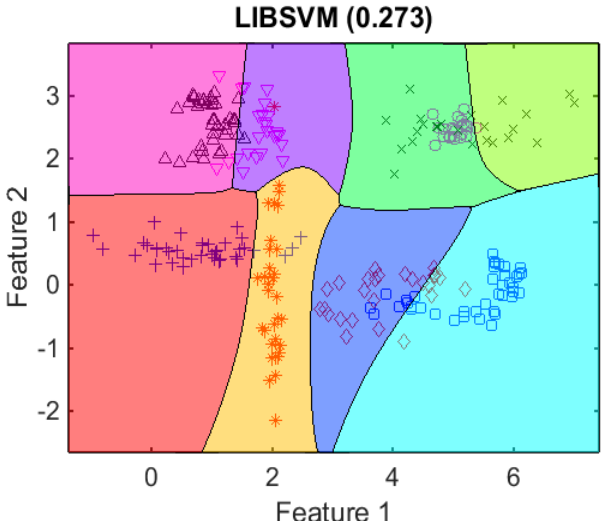
2) Kernel SVM

Kernel SVM has a slight variation from the Linear SVM. Kernel SVM uses kernel Trick to make to transform the data into higher dimension. Kernel trick is using a function between two data points and calculating a matrix for each dot product as an element of the matrix. The function used for RBF kernel is accuracies with different functions are listed below. RBF is **Radial Basis Function kernel**, is a popular kernel function used in various kernelised learning algorithms. The RBF gave a better accuracy on 5 fold cross validation than the linear SVM. The accuracy for RBF kernel function is **39 percent.**

$$K(\mathbf{x}, \mathbf{x}') = \exp\left(-\frac{||\mathbf{x} - \mathbf{x}'||^2}{2\sigma^2}\right)$$

The another function used is sigmoid and the accuracy for that **42.1 percent.**

$$\text{Sigmoid} = \tanh(\gamma\langle x, x'\rangle + r)$$

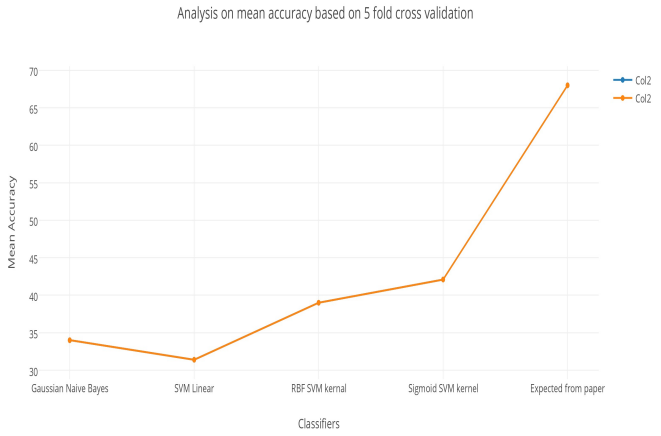


3) Gaussian Naive Bayes

Gaussian Naive Bayes classifier assumes that the continuous values are associated with each class are distributed according to a Gaussian distribution. The accuracy we got from this is **34 percent**.

III. Analysis

Frequentistic Approach - Now, we have got very less accuracies in our current paper than the chosen one. The reason for that is we have the reduced the dimensionality of the matrix from nearly 3 lakhs to 5000 features. In the due process, we have lost some of the important features which were important discriminators between tags. The reason behind reducing the dimensions is that we had a sparse matrix and we wanted remove the zeros from the matrix and gain a reduced dimensioned matrix with only the important features. But, through this paper we can conclude that we have failed to collect the information at one place and lost information instead that too important features. The figure below shows us the difference :-



Now, let us discuss the accuracies that we have got using different classifiers.

The best classifier that we have got is the kernel Sigmoid SVM and the worst performing was linear SVM. As data being highly sparsed, there is no gurantee for data being linearly classifiable. That could be the reason linear SVM is worst performing in this case. So, this gives a clear hint of nee d of projection of data into some higher dimension using kernel technique. So, we have chosen two different kernel function on the dataset Gaussian and sigmoid.

Their performance is relatively comparable, but sigmoid kernel function performed better.

Bayesian Approach :-

It is always a good approach to compare between a frequentest and Bayesian approach. Since the data was based on discrete bag of model, we have tried multinomial naïve bayes as it could have been a better choice but, what we have observed is that when the data is being decomposed into lower dimension, the data samples take the negative values of some feature as it works on the frequency of the feature variable. So, this lead us to choose a guassian naïve bayes classifier which can perform on real valued data and it gave a decent accuracy.

Acknowledgment

We would like to thank our Professor Avinash Sharma (Our Course Instructor) and our mentor TA Irshad Bhatt and Syyed for helping us around in pointing out the mistakes we have committed and giving us the right direction.

References

[1] Avigit K. Saha, Ripon K. Saha and Kevin A. Schneider. 2013. **A Discriminative Model Approach for Suggesting Tags Automatically for Stack Overflow Questions.** In*10th Working Conference on Mining Software Repositories. Mining Challenge*. IEEE, pages 73-76

[2] Shaowei Wang, David Lo, Bogdan Vasilescu, and Alexander Serebrenik, 2014. **EnTagRec: An Enhanced Tag Recommendation System for Software Information Sites**. In *30th International Conference on Software Maintenance and Evolution (ICSME 2014)*. IEEE

[3] Hansen, P. C. (1987). "The truncated SVD as a method for regularization". *BIT* **27**: 534–553. doi:10.1007/BF01937276.

[4] Hazewinkel, Michiel, ed. (2001),"Chi-squared distribution",*Encyclopedia of Mathematics*,Springer,ISBN978-1-55608-010-4