Automatically Generate Tags for Questions Posted on StackExchange.com

Zhe Yu[*]
Department of Computer Science
North Carolina State University
Raleigh, NC 27606
zyu9@ncsu.edu

Abstract

In this project, the problem of how to automatically generate tags for questions posted on StackExchange websites is discussed. There are certain needs for this task since not all the users can tag their questions accurately due to the lack of understanding of the question or the website. An evaluation of $Fscore_M$ is utilized to meet the specific requirement of the task. After carefully studied the current multi-classification approaches in text mining, term frequency is selected for featurization, three different methods, SVM, Naive Bayes, and Decision Tree, are chosen to compare with each other in fulfilling the requirements. By analysing the results, imbalance is found to be the major problem for this task. (Not finished yet:)An over-sampling technique (SMOTE) is implemented to balance the data. The final result with SVM + SMOTE is satisfying.

1 Introduction

Now a days, more and more people post their questions and find their answers on StackExchange.com. With the help of tags, every user can find the questions they are most confident to solve. However, not all the users can tag their questions accurately due to the lack of understanding of the question or the website. This certain needs of automatically generate tags motivates the author to conduct this experiment. Without loss of generality, the experiment is conducted on one single site of StackExchange.com, which is http://anime.stackexchange.com/.

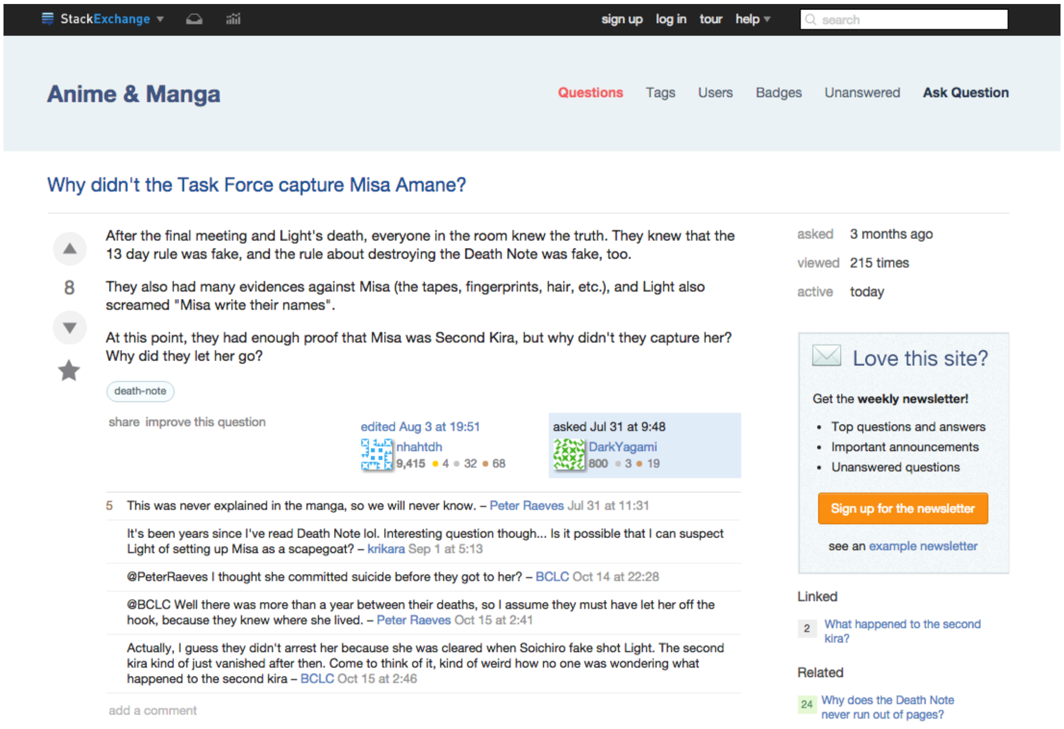
One significant characteristic of the data is imbalance. There are only 22 documents with the most minority tag ”resources” while 1609 documents with the most majority tag ”others”. To evaluate the performance, we use the $Fscore_M$ [1] across the tags to equally reflect the performance across each tag. The $Fscore_\mu$ [1] is also calculated as another view of the performance since it favors majority classes much more than the minority ones. Our target is to achieve highest $Fscore_M$.

In order to achieve a high $Fscore_M$, data needs to be balanced before the training of the classifier. Synthetic minority over-sampling technique (SMOTE) [2] is a widely accepted technique to tackle imbalance problems with great improvements reported in literature [3-5]. After the implementation of SMOTE in the experiment, an increase in $Fscore_M$ is reported without sacrificing the $Fscore_\mu$.

1.1 Data

All the data used in this project is in a single file called anime.txt. It is collected from http://anime.stackexchange.com/ and formated by RAISE Lab. It contains 4827 documents with tags. For each document, the first tag is treated as its label. All material of this project can be found

[*] Unit ID: zyu9, Student ID: 200109973.



Why didn't the Task Force capture Misa Amane?

8

After the final meeting and Light's death, everyone in the room knew the truth. They knew that the 13 day rule was fake, and the rule about destroying the Death Note was fake, too.

They also had many evidences against Misa (the tapes, fingerprints, hair, etc.), and Light also screamed "Misa write their names".

At this point, they had enough proof that Misa was Second Kira, but why didn't they capture her? Why did they let her go?

death-note

share improve this question

edited Aug 3 at 19:51
nhahtdh
9,415 ● 4 ● 32 ● 68

asked Jul 31 at 9:48
DarkYagami
800 ● 3 ● 19

5 This was never explained in the manga, so we will never know. – Peter Raeves Jul 31 at 11:31

It's been years since I've read Death Note lol. Interesting question though... Is it possible that I can suspect Light of setting up Misa as a scapegoat? – krikara Sep 1 at 5:13

@PeterRaeves I thought she committed suicide before they got to her? – BCLC Oct 14 at 22:28

@BCLC Well there was more than a year between their deaths, so I assume they must have let her off the hook, because they knew where she lived. – Peter Raeves Oct 15 at 2:41

Actually, I guess they didn't arrest her because she was cleared when Soichiro fake shot Light. The second kira kind of just vanished after then. Come to think of it, kind of weird how no one was wondering what happened to the second kira – BCLC Oct 15 at 2:46

add a comment

asked 3 months ago
viewed 215 times
active today

Love this site?

Get the **weekly newsletter!**

- Top questions and answers
- Important announcements
- Unanswered questions

Sign up for the newsletter

see an example newsletter

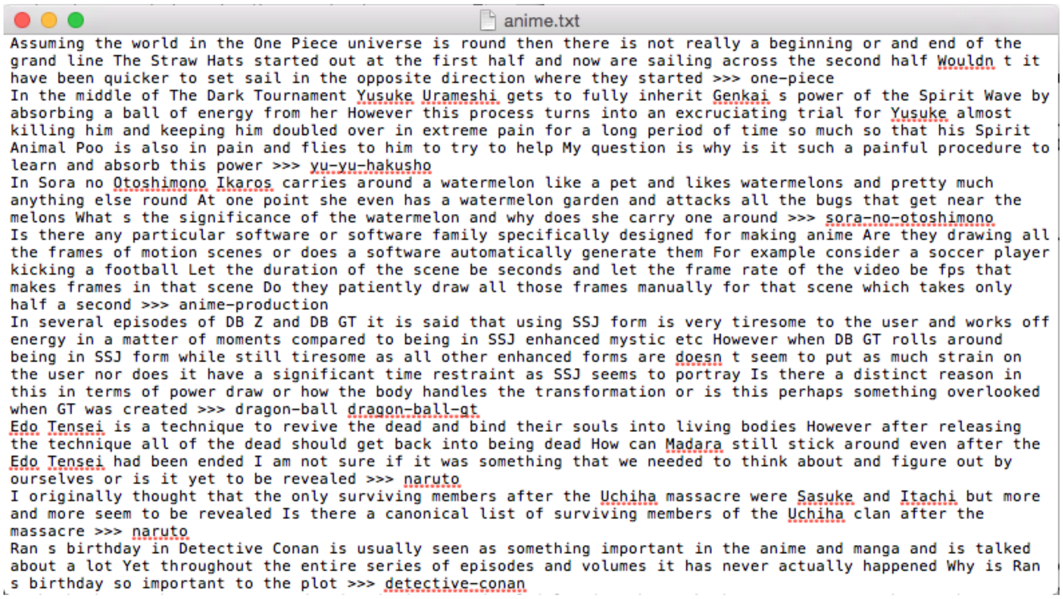
Linked

2 What happened to the second kira?

Related

24 Why does the Death Note never run out of pages?

Figure 1: Example of post



```
Assuming the world in the One Piece universe is round then there is not really a beginning or and end of the
grand line The Straw Hats started out at the first half and now are sailing across the second half Wouldn t it
have been quicker to set sail in the opposite direction where they started >>> one-piece
In the middle of The Dark Tournament Yusuke Urameshi gets to fully inherit Genkai s power of the Spirit Wave by
absorbing a ball of energy from her However this process turns into an excruciating trial for Yusuke almost
killing him and keeping him doubled over in extreme pain for a long period of time so much so that his Spirit
Animal Poo is also in pain and flies to him to try to help My question is why is it such a painful procedure to
learn and absorb this power >>> yu-yu-hakusho
In Sora no Otoshimono Ikaros carries around a watermelon like a pet and likes watermelons and pretty much
anything else round At one point she even has a watermelon garden and attacks all the bugs that get near the
melons What s the significance of the watermelon and why does she carry one around >>> sora-no-otoshimono
Is there any particular software or software family specifically designed for making anime Are they drawing all
the frames of motion scenes or does a software automatically generate them For example consider a soccer player
kicking a football Let the duration of the scene be seconds and let the frame rate of the video be fps that
makes frames in that scene Do they patiently draw all those frames manually for that scene which takes only
half a second >>> anime-production
In several episodes of DB Z and DB GT it is said that using SSJ form is very tiresome to the user and works off
energy in a matter of moments compared to being in SSJ enhanced mystic etc However when DB GT rolls around
being in SSJ form while still tiresome as all other enhanced forms are doesn t seem to put as much strain on
the user nor does it have a significant time restraint as SSJ seems to portray Is there a distinct reason in
this in terms of power draw or how the body handles the transformation or is this perhaps something overlooked
when GT was created >>> dragon-ball dragon-ball-gt
Edo Tensei is a technique to revive the dead and bind their souls into living bodies However after releasing
the technique all of the dead should get back into being dead How can Madara still stick around even after the
Edo Tensei had been ended I am not sure if it was something that we needed to think about and figure out by
ourselves or is it yet to be revealed >>> naruto
I originally thought that the only surviving members after the Uchiha massacre were Sasuke and Itachi but more
and more seem to be revealed Is there a canonical list of surviving members of the Uchiha clan after the
massacre >>> naruto
Ran s birthday in Detective Conan is usually seen as something important in the anime and manga and is talked
about a lot Yet throughout the entire series of episodes and volumes it has never actually happened Why is Ran
s birthday so important to the plot >>> detective-conan
```

Figure 2: Example of anime.txt

at https://github.com/azhe825/AGT.git. An example of document on anime.stackexchange.com is shown in Figure 1. An example of the processed data in anime.txt is shown in Figure 2.

The number of documents in each tag is: {'others': 1609, 'identification-request': 1099, 'naruto': 547, 'one-piece': 204, 'anime-production': 164, 'fullmetal-alchemist': 114, 'tropes': 110, 'bleach': 87, 'death-note': 86, 'fairy-tail': 78, 'dragon-ball': 66, 'code-geass': 51, 'japanese-language': 49, 'sword-art-online': 48, 'monogatari-series': 46, 'madoka-magica': 46, 'culture': 45, 'pokemon':

45, ’fate-stay-night’: 43, ’shingeki-no-kyojin’: 40, ’hunter-x-hunter’: 38, ’anime-history’: 35, ’manga-production’: 33, ’from-the-new-world’: 26, ’terminology’: 25, ’neon-genesis-evangelion’: 24, ’music’: 24, ’toaru-majutsu-no-index’: 23, ’resources’: 22}.

1.2 Related works

Clayton and Byrne, 2013 [6] have worked on StackOverflow tag prediction and developed an ACT-R inspired Bayesian probabilistic model. This approach achieves a 65% of accuracy by choosing the tag that has the highest log odds of being correct, given the tags prior log odds of occurrence and adjusting for the log likelihood ratio of the words in the post being associated with the tag.

Kuo, 2011 [7] has also worked on StackOverflow tag prediction. Kuo uses a co-occurrence model that predicts tags based on the relation (co-occurrence) between the words and tags. Initially built for next-word prediction in large documents, this model is adapted to the StackOverflow dataset by constraining the next word predicted to only tags. A 47% classification accuracy is achieved.

Another model called SNIF-ACT (Fu & Pirolli, 2007 [8]) also uses co-occurrences and can predict link traversal for search queries. The model predicts the most likely link that a person will click on by a search query (goal state) and fetched results.

2 Method

As suggested by [9] and [10], term frequency is calculated as feature and term frequency inverse document frequency (tf-idf) is used to select the most effective features. Inspired by CSC 522, three of the most famous classification methods, SVM, Naive Bayes, and Decision Tree, are used to conduct the experiments. In the second part of the experiment, SMOTE is implemented to balance the data before training.

2.1 Preprocessing

Tokenization: stemming and stop words removal.

Term frequency: simple term frequency count is utilized to construct the feature matrix. The value in feature matrix $M_{i,j}$ represents the number of appearance of the j_{th} token in the i_{th} document.

Tf-idf selection: tf-idf score of each token is calculated as $tfidf(token_i) = \frac{WordCount(token_i)}{\Sigma WordCount(token_j)} log(\frac{TotalNumberofDocs}{NumberofDocs(token_i)})$. Tokens are then sorted by their tf-idf scores and tokens with the largest n tf-idf scores are selected to form the new feature matrix.

L2 Normalization: L2 Normalization is implemented on each row of the feature matrix to eliminate the difference between long documents and short documents.

SMOTE: SMOTE with five nearest neighbors is implemented to over-sample documents in each tag to reach the number of majority tag. That is, SMOTE is implemented on each training set and after SMOTE, each tag will have the same number of documents.

2.2 Classifiers

Support Vector Machine: linear kernel with primal problem solving is chosen since state-of-art accuracy on text classification is reported in [11].

Naive Bayes: multinomial model is chosen according to the performance report in [12].

Decision Tree: CART [13].

2.3 Performance Metrics

Cross validation: 5 by 5 cross validation, each time the classifier is trained on 80% of the data and tested on the rest 20%. For each classifier with each number of features, there are 25 results. The median and iqr of the 25 results represent the performance. A low iqr means that we can trust the result.

$Fscore_M$**:** F-score on each tag can be calculated after the trained classifier is tested on test set. The mean of F-score on each tag is then calculated to represent the overall performance of the classifier. Regardless of the population within each tag, the $Fscore_M$ represents performance on each tag equally [1].

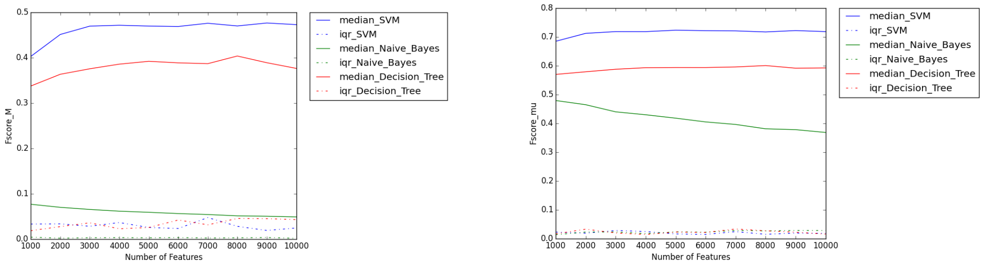
$Fscore_\mu$**:** the $Fscore_\mu$ on each tag is calculated by multiply F-score on each tag by its population. Classifiers are trained to achieve high in this score by default, which is not desired in this task and therefore over-sampling methods like SMOTE is introduced in the later experiment [1].

3 Experiments

In the first experiment, SMOTE is not implemented. The performances of SVM, Naive Bayes, and Decision Tree are compared against each other by their $Fscore_M$ and $Fscore_\mu$.

In the second experiment, SMOTE is implemented to over-sample documents in each tag to reach the number of majority tag. The performances of SVM, Naive Bayes, and Decision Tree, with or without SMOTE are compared against each other by their $Fscore_M$ and $Fscore_\mu$.

3.1 Without SMOTE



(a) $Fscore_M$

(b) $Fscore_\mu$

Figure 3: F-scores without SMOTE

SVM outperforms Naive Bayes and Decision Tree in both $Fscore_M$ and $Fscore_\mu$, according to Figure 3. However, even though SVM can achieve a 0.7 $Fscore_\mu$, its $Fscore_M$ is below 0.5, which is definitely not good enough.

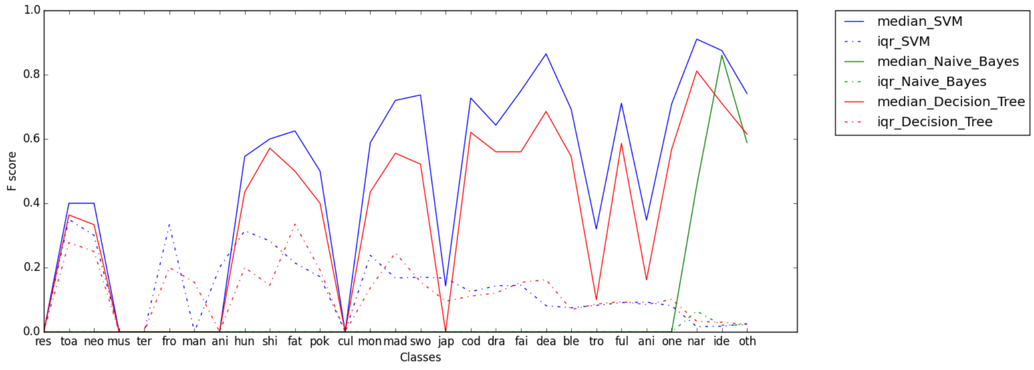
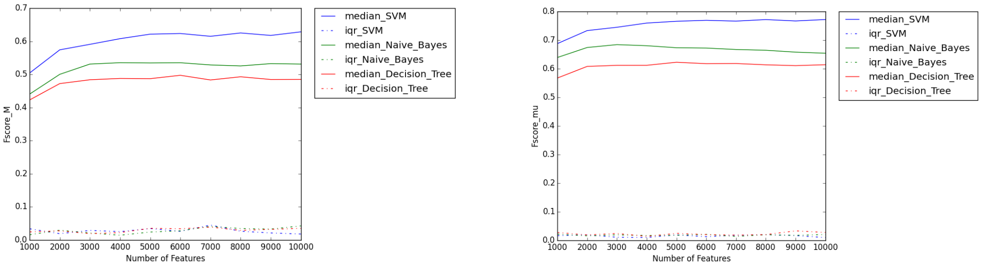


Figure 4: F-scores of each tag without SMOTE, sorted by population

As we can see from Figure 4, most of the tags are actually ignored (low median or high iqr) by all the classifiers due to their small population. This implies that a) $Fscore_\mu$ cannot correctly represent the true performance; b) over-sampling is essential and promising in achieving a better $Fscore_M$. Another fact is that SVM wins on every tag.

3.2 With SMOTE



(a) $Fscore_M$

(b) $Fscore_\mu$

Figure 5: F-scores with SMOTE

SVM still outperforms Naive Bayes and Decision Tree in both $Fscore_M$ and $Fscore_\mu$, according to Figure 5. The major difference is that with SMOTE, Naive Bayes outperforms Decision Tree while it is the opposite without SMOTE.

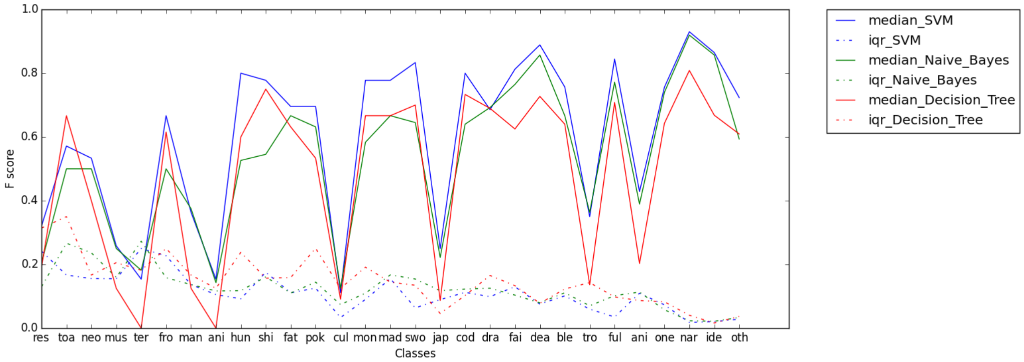
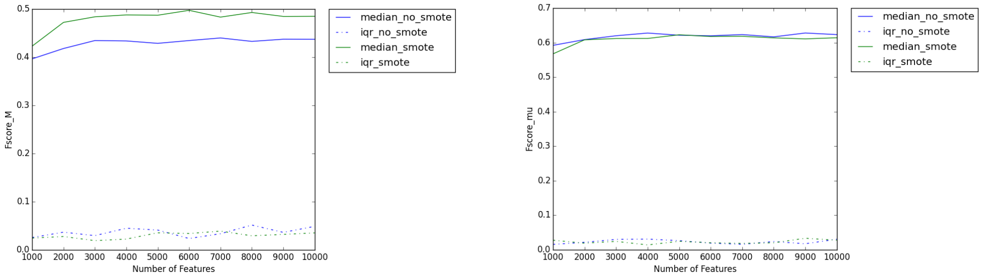


Figure 6: F-scores of each tag with SMOTE, sorted by population

As we can see from Figure 6, tags with small population are not totally ignored by classifiers any more. This implies that the data is well balanced by SMOTE. Still, SVM wins on most of the tags.

3.3 Comparison between SMOTE and No-SMOTE



(a) $Fscore_M$

(b) $Fscore_\mu$

Figure 7: F-scores of Decision Tree

Decision Tree: from Figure 7 we can see that SMOTE improves the $Fscore_M$ without sacrificing $Fscore_\mu$. It is also suggested by Figure 8 that performance of Decision Tree on most of the tags with

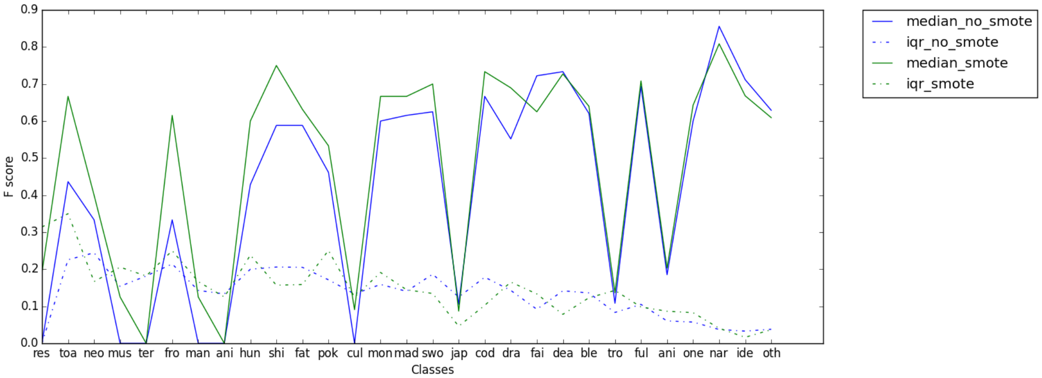
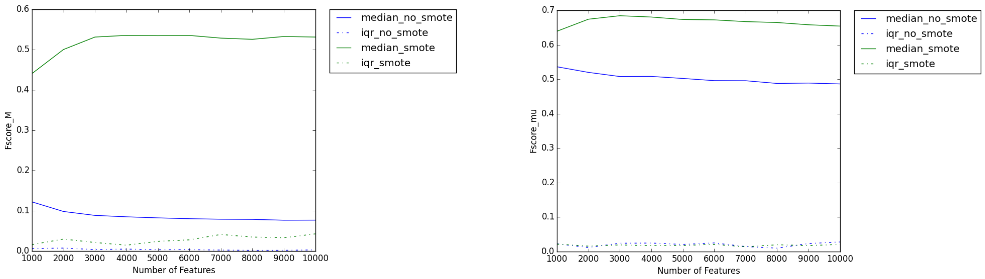


Figure 8: F-scores of each tag with Decision Tree, sorted by population

small population is improved by SMOTE (higher median or lower iqr of F score). The performance of Decision Tree (CART) with SMOTE, 3000 features can achieve 0.49 in $Fscore_M$ and 0.61 in $Fscore_\mu$.



(a) $Fscore_M$

(b) $Fscore_\mu$

Figure 9: F-scores of Naive Bayes

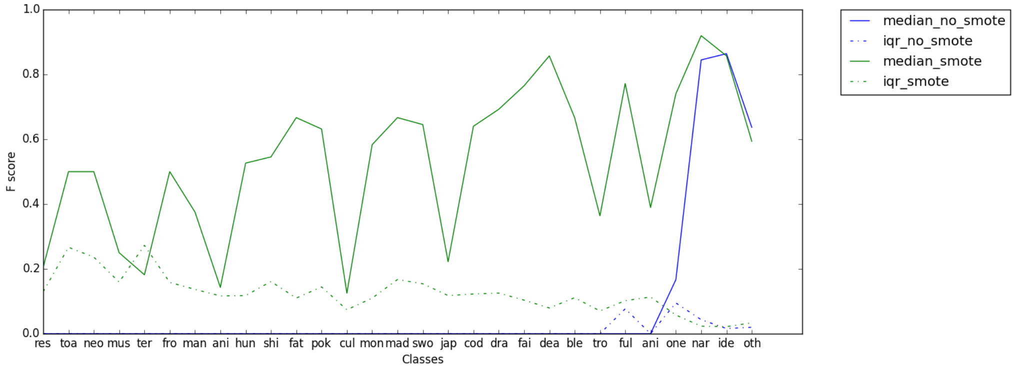
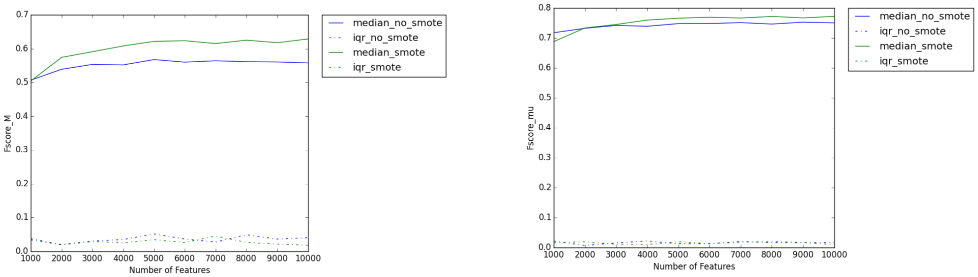


Figure 10: F-scores of each tag with Naive Bayes, sorted by population

Naive Bayes: SMOTE impacts a lot on the performance of Naive Bayes. As we can see from Figure 9, without SMOTE, the $Fscore_M$ is about 0.1 while it reaches above 0.5 with SMOTE. Even the $Fscore_\mu$ is improved by about 0.2 after SMOTE is implemented. It is shown in Figure 10 that Naive Bayes totally ignores the tags with low population (0 in median of F score) while all the tags are considered after SMOTE. This explains the huge improvement of $Fscore_M$ in Figure 9(a).

The performance of Naive Bayes (multinomial) with SMOTE, 3000 features can achieve 0.53 in $Fscore_M$ and 0.69 in $Fscore_\mu$.



(a) $Fscore_M$

(b) $Fscore_\mu$

Figure 11: F-scores of Support Vector Machine

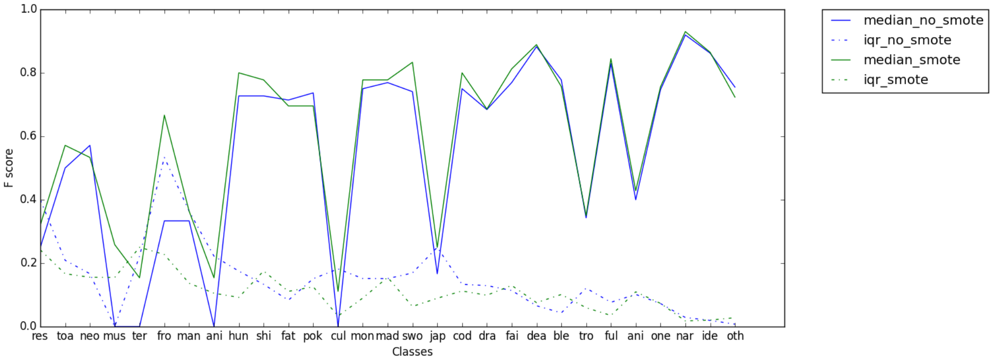


Figure 12: F-scores of each tag with Support Vector Machine, sorted by population

Support Vector Machine: SVM is still the best classifier for this task. As shown in Figure 11, SMOTE improves the $Fscore_M$ by approximate 0.1 without sacrificing $Fscore_\mu$. Although 0.6 in $Fscore_M$ and 0.7 in $Fscore_\mu$ do not seem enough, this result is actually comparable to the state-of-art result [6]. Similar as the result of Decision Tree, Figure 12 shows that the F scores on minority tags improves after SMOTE, which leads to the improvement of $Fscore_M$. The performance of linear SVM with SMOTE, 5000 features can achieve 0.62 in $Fscore_M$ and 0.76 in $Fscore_\mu$.

4 Conclusions and Future Works

Conclusions of this paper are: a) $Fscore_M$ is a good and reliable performance metric regarding to multi-classification problems; b) linear SVM outperforms Naive Bayes and Decision Tree in this task; c) SMOTE improves the result of $Fscore_M$ significantly without sacrificing $Fscore_\mu$, which also implies that imbalance is indeed a problem in this tag generation task; d) 5000 features are enough for this tag generation task. With SMOTE, linear SVM can achieve an $Fscore_M$ of 0.62 and $Fscore_\mu$ of 0.76 with feature number 5000. The result is reliable since it is repeated 25 times and the iqr is lower than 0.04. This is the method we suggest for this certain tag generation task.

In our next step, different kernels of SVM can be tested to see if further improvement is available. There are other promising classifiers, artificial neural network with deep learning for example, as well. Another possible way to improve the performance can be: a) first divide the multi-classification problem to k (k is the number of different classes) binary classification problems (one class as positive and all the other classes as negative); b) implement SMOTE on each binary classification problem to over-sample the documents of positive class to match the population of negative documents; c) train a linear SVM classifier on each of the binary classification problem; d) aggregate the trained classifiers to predict unknown tags based on scores of each SVM output (distance from decision boundary).

The result of this project can be implemented to automatically generate tag for posts or simply providing users suggestions of tag options if the result is not considered reliable enough.

References

[1] Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information Processing & Management*, Vol. 45(4), pp. 427-437.

[2] Chawla, Nitesh V., Kevin W. Bowyer, Lawrence O. Hall & W. Philip Kegelmeyer. (2002) SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, pp. 321-357.

[3] Han, H., Wang, W. Y., & Mao, B. H. (2005). Borderline-SMOTE: a new over-sampling method in imbalanced data sets learning. *Advances in intelligent computing*, pp. 878-887.

[4] Bunkhumpornpat, C., Sinapiromsaran, K., & Lursinsap, C. (2009). Safe-level-smote: Safe-level-synthetic minority over-sampling technique for handling the class imbalanced problem. *Advances in Knowledge Discovery and Data Mining*, pp. 475-482.

[5] Luengo, J., Fernndez, A., Garca, S., & Herrera, F. (2011). Addressing data complexity for imbalanced data sets: analysis of SMOTE-based oversampling and evolutionary undersampling. *Soft Computing*, Vol. 15(10), pp. 1909-1936.

[6] Clayton, S., & Byrne, M.. (2013) Predicting tags for stackoverflow posts. *In Proceedings of ICCM*, Vol. 2013.

[7] Kuo, D. (2011). On word prediction methods. *Technical report*, EECS Department, University of California, Berkeley.

[8] Fu, W. T., & Pirolli, P. (2007). SNIF-ACT: A cognitive model of user navigation on the World Wide Web. *HumanComputer Interaction*, Vol. 22(4), pp. 355-412.

[9] Larsen, B., & Aone, C. (1999, August). Fast and effective text mining using linear-time document clustering. *In Proceedings of the fifth ACM SIGKDD international conference on Knowledge discovery and data mining* pp. 16-22.

[10] Menzies, T. (2006). Improving IV&V Techniques Through the Analysis of Project Anomalies: Bayes networks-preliminary report. Tech. rep., West Virginia University. (http://menzies.us/pdf/06anomalies-bayes0.pdf).

[11] Joachims, T. (2006, August). Training linear SVMs in linear time. *In Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining* pp. 217-226.

[12] McCallum, A., & Nigam, K. (1998, July). A comparison of event models for Naive Bayes text classification. *In AAAI-98 workshop on learning for text categorization* Vol. 752, pp. 41-48.

[13] Miotto, O., Tan, T. W., & Brusic, V. (2005). Supporting the curation of biological databases with reusable text mining. *Genome informatics*, Vol. 16(2), pp. 32-44.

Resources

All the materials of this project can be found at https://github.com/azhe825/AGT.git.