Auto-Tagging Piazza Posts

CS229 — Autumn 2013 — Prof. Andrew Ng

Ai Jiang *hajiang*
Kathy Sun *kathysun*

December 13, 2013

1 Problem Introduction

Our project is to make an automatic tag recommender for Piazza posts. Since each post could have multiple tags, the problem can be viewed as a multilabel classification problem, where each post and possible tag combination is a binary classification. This could be used to improve the user experience for students using Piazza as well as increase the accuracy and relevancy of tags. For the project, the data we used is the set of Piazza posts from the CS229 class, but the methods we describe generalize to Piazza data from any class.

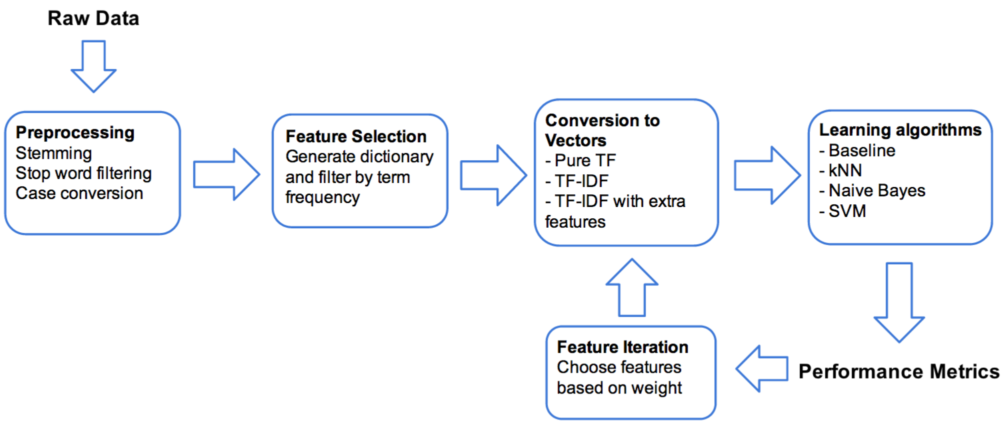
2 Method Approach

Each post on Piazza corresponds to a training example and the text content of the post is processed and converted into vector format, where each feature corresponds to some statistic about a word. We then run various classification learning algorithms to predict the tags of each post of a held out test set and calculate our desired metrics. From there we perform analysis of our initial results and implement further improvements to our algorithms.

3 Data Collection and Preprocessing

From the raw data, we applied stop word removal, stemming and case conversion. Then, in order to reduce the feature set, we computed the term frequency of each word and filtered out terms below a minimum cut-off value of 3, which gave us a dictionary of 2593 words. Lastly, we converted each post to vector format using the multinomial event model where each element corresponded to the term frequency of that word, and later the tf-idf score.

Figure 1: System Pipeline

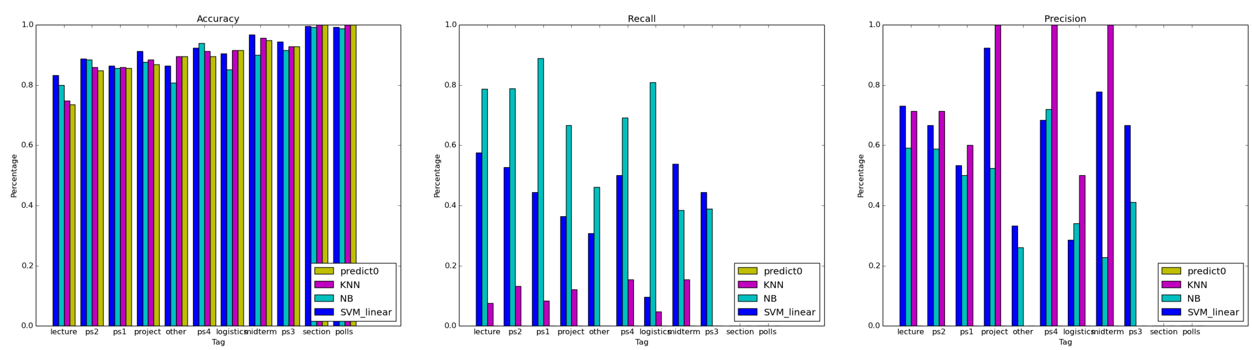


4 Learning Algorithms and Initial Results

We decided to use Naive Bayes and SVM to run supervised learning and for a baseline, we used a predictor that predicts 0 for everything. For a given tag, however, the large majority of posts has class 0, and thus our simple baseline predictor scored just as well as Naive Bayes and SVM in terms of accuracy. Because of this, we decided to use a kNN predictor as our baseline as well as to look at precision and recall in addition to accuracy. Naive Bayes seemed to do better for recall on most tags and kNN for precision.

Figure 2: Initial Tag Metrics of Training Set Split 70/30

(a) Accuracies (b) Recalls (c) Precisions



5 Analysis and Improvements

From our initial results, we plotted the bias variance graph and saw that our test error generally decreased as we added more training examples, suggesting we had high variance. As

Figure 3: Bias and Variance Diagnosis

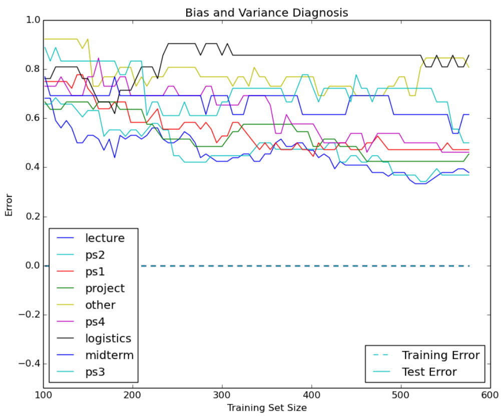
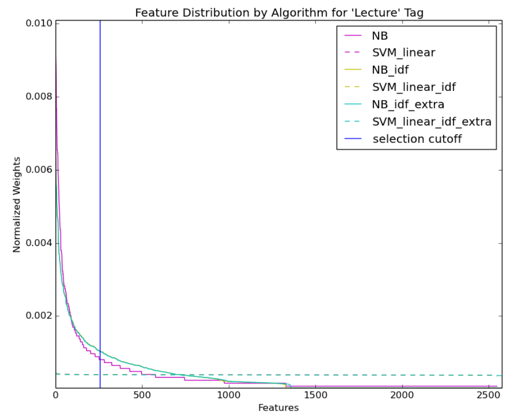


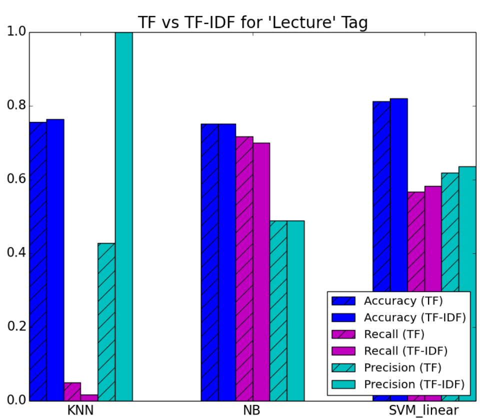
Figure 4: Select 10% Best Features



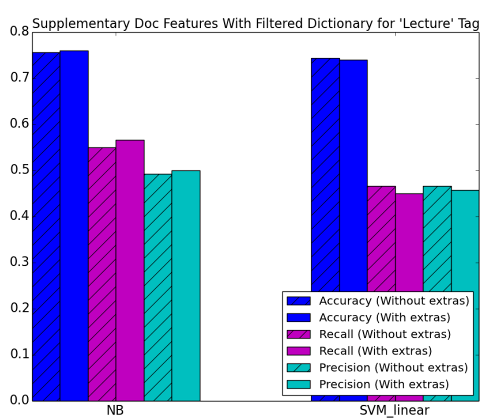
a result, we decided to decrease the amount of features by selecting features based on the weights of the initial learning to reduce our feature set down to 259 (10%). The downward trend also suggests more training examples may help. Another optimization we did was use tf-idf scores instead of pure term frequencies. The last improvement that we made was to use course handouts from the class and convert them into extra features. We read in the documents from the class including homeworks, lecture notes, section notes and logistics documents and then converted them to the same vector representation as our training examples and used the cosine distance between each post and each document as a new feature. This improved all of our metrics for Naive Bayes classifier but didn't have a large effect on the SVM classifier.

Figure 5: Changing Feature Representation and Size

(a) TF vs TF-IDF Representation



(b) Course Documents as Extra Features with Filtered Dictionary



6 Results

Table 1: Top Five Most Useful Words Used as Features

lecture	Naive Bayes	note	lectur	function	1	0
	SVM w/ Linear Kernel	lectur	read	page	typo	2
ps2	Naive Bayes	set	point	word	label	1
	SVM w/ Linear Kernel	ps2	empti	liblinear	wo	shown
ps1	Naive Bayes	question	problem	function	theta	1
	SVM w/ Linear Kernel	5a	sheet	quick	anyth	assumpt
project	Naive Bayes	project	featur	data	learn	propos
	SVM w/ Linear Kernel	project	propos	multiclass	svms	normal
other	Naive Bayes	learn	featur	question	data	answer
	SVM w/ Linear Kernel	scipi	associ	collabor	unobserv	link
ps4	Naive Bayes	state	valu	1	iter	problem
	SVM w/ Linear Kernel	norm	ps4	4	definit	assum
logistics	Naive Bayes	scpd	student	exam	correct	submit
	SVM w/ Linear Kernel	mandatori	calendar	find	huang	writeup
midterm	Naive Bayes	midterm	exam	scpd	1	time
	SVM w/ Linear Kernel	midterm	abov	cover	doe	practic
ps3	Naive Bayes	problem	centroid	imag	bound	ps3
	SVM w/ Linear Kernel	ps3	werent	distinct	markov	q1b
section	Naive Bayes	featur	section	hour	offic	class
	SVM w/ Linear Kernel	section	friday	attach	quick	code
polls	Naive Bayes	hard	easi	latex	bayesian	midterm
	SVM w/ Linear Kernel	bayesian	frequentist	item	vote	submityou

Table 2: Best Algorithm per Tag

Tag	Accuracy	Recall	Precision
lecture	SVM:82%	NB:66%	SVM:63%
ps1	SVM:88%	NB:72%	SVM:71%
ps2	SVM:87%	NB:81%	SVM:75%
ps3	SVM:93%	NB:53%	SVM:61%
ps4	SVM:93%	NB:69%	NB:86%
midterm	SVM:96%	NB:80%	SVM:60%
logistics	SVM:91%	NB:95%	NB:40%
project	SVM:93%	NB:79%	SVM:76%
other	SVM:89%	NB:54%	NB:70%

7 Conclusion

We were able to build a successful system to automatically tag Piazza posts with high accuracy. Depending on the application requirements and what metric should be prioritized, different learning algorithms seem to work better than others (i.e. Naive Bayes for recall, kNN for precision). Since we treated each tag as an independent binary classification problem, our training set was heavily skewed with more untagged than tagged labels for each tag. This made it difficult to learn for some tags. Trying various ideas such as tf-idf and supplementary document features had different effects on different classifiers but generally improved performance metrics. In the future, it would be interesting to try to treat each field of the post separately, similar to the BM25-F algorithm from information retrieval. The techniques and ideas presented here would also be useful in the application of determining post similarity, which could be used to detect whether a similar post has been made in order to reduce duplicate posts.

References

[1] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Mach. Learn.*, 20(3):273–297, September 1995.

[2] J. L. Dawson. Suffix removal for word conflation. In *Bulletin of the Association for Literary and Linguistic Computing*, pages 33–46, 1974.

[3] C. D. Manning, P. Raghavan, and H. Schutze. *Scoring, term weighting, and the vector space model*, page 100. 2008.

[4] Andrew McCallum and Kamal Nigam. A comparison of event models for naive bayes text classification. In *AAAI-98 Workshop on Learning for Text Categorization*, volume 752, pages 41–48. AAAI Press, 1998.

[5] José R. Pérez-Agüera, Javier Arroyo, Jane Greenberg, Joaquin Perez Iglesias, and Victor Fresno. Using bm25f for semantic search. In *Proceedings of the 3rd International Semantic Search Workshop*, SEMSEARCH '10, pages 2:1–2:8, New York, NY, USA, 2010. ACM.

[6] David M. W. Powers. Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness & Correlation. Technical Report SIE-07-001, School of Informatics and Engineering, Flinders University, Adelaide, Australia, 2007.

[7] A. Rajaraman and J. D. Ullman. *Data Mining*, pages 1–17. Cambridge University Press, 2011.