Multi-label Classification of Stack Overflow Questions using OneVsRest Classification Paradigm

Submitted By
Pramod Kumar Gudipati and Keerthi Korivi

Problem Definition

To Q&A forums like Stack Overflow and Quora, tagging questions is an important aspect for a good user experience as it helps in easy retrieval of the information. Also, organizing these posts will be lot easier.

We implemented a multi-label classification system that predicts tags to questions to enhance user experience using one-vs-rest paradigm. Generally, Classification in machine learning is multi-class. However, if we see stack overflow questions, there is a possibility that a question could belong to multiple domains i.e. multiple tags can be assigned to a question. For instance, a question in java domain could also belong to other domains like object oriented domain, multithreading, exception handling etc.

Background

Classification is the process of identifying a correct class label for a given input. Some examples of classification tasks are:

i) Decided whether an email is spam or not is a binary classification, since it has only two possible outcomes i.e. either spam or not spam
ii) Deciding what topic a newspaper article belongs to is a multi-class classification, since it has more than two possible outcomes such as sports, education, health, politics etc.

Multi-label classification is another kind of classification which involves assigning of multiple labels to a particular input. Our problem domain falls under multi-label classification.

The two major approaches of multi-label classification are:

i) Problem Transformation - transforms the multi-label classification problem into several single-label classification problems. A common approach, called OneVsRest or Binary Relevance divides the multi-label classification problem into 'n' independent binary classification problems.
ii) Adaptive Algorithms – use single-label classifiers like SVM, Random Forest, decision tree etc., to handle multi-label data.

Our project focuses on Binary Relevance approach using different classifiers like Stochastic Gradient Descent (SGD), Sequential Minimal Optimization (SMO) and Naïve Bayes.

Dataset

We took dataset provided for one of the competitions on Kaggle. The dataset contains about 150000 records of which we have subsampled the dataset to 50000 records. Each record contains the following columns: id, title, body, tags. Each record can have multiple tags. In total, there are 16827 unique tags. However, since not all tags have good frequency in the dataset. So, we have limited our experiments to 15 tags that are most common in the dataset. List of tags we

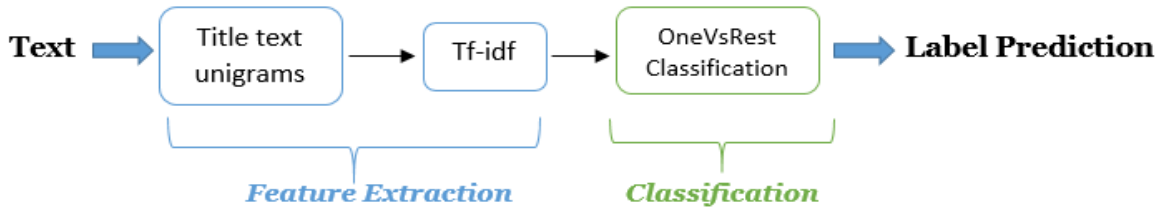
considered: java, android, JavaScript, multithreading, mysql, swing, exception, eclipse, python, Django, dictionary, matplotlib, numpy, scala and functional-programming.

We started our experiments with dataset of 10000 questions out of which 9000 are training instances and 1000 are test instances. Later, we extended our experiments up to 50000 questions.

Methodology

For each question in the dataset, we took title into consideration for classification with its associated tags for training and evaluation. Then, we have constructed unigrams for title of each question and calculated the Tf-Idf for those unigrams as part of feature extraction. The extracted features are then fed to oneVsRest classification paradigm to predict labels.

As stated in the background section, OneVsRest paradigm decomposes the classification problem into k independent binary classification problems, training a separate classifier for each of the k possible output labels.



Feature Extraction and Classification Flow

Implementation

We have implemented OneVsRest approach using following:

- Sklearn library
- Mulan API using MEKA GUI interface.

Tools Used: Pycharm, Scikit-learn, numpy, matplotlib, MEKA, Mulan API.

Data Pre-Processing is common to both of the above mentioned implementations. It includes removing code fragments, links, line breaks, punctuations, stop words from the data, since these information is not useful in classifying the text.

For Sklearn implementation, we have created a pipeline using Tf-idf vectorizer and OneVsRest paradigm. We have used the Stochastic Gradient Descent (SGD) as classifier for each independent binary classification problem in the OneVsRest approach.

For Mulan API using MEKA, data needs to be in ARFF format. We have written a parser in python to convert it into ARFF format. The OneVsRest classifier in Mulan is specifically called as Binary Relevance approach. We have used Stochastic Gradient Descent (SGD), Sequential Minimal Optimization (SMO – A variant of Support Vector Machines) and Naïve Bayes classifiers for each independent binary classification problem.

Evaluation Criteria

Our Evaluation criteria is F1 score.

In statistical analysis of binary classification, the F1 score (also F-score or F-measure) is a measure of a test's accuracy. It considers both the precision p and the recall r of the test to compute the score: p is the number of correct positive results divided by the number of all positive results, and r is the number of correct positive results divided by the number of positive results that should have been returned. The F1 score can be interpreted as a weighted average of the precision and recall, where an F1 score reaches its best value at 1 and worst at 0.

The traditional F-measure or balanced F-score (F1 score) is the harmonic mean of precision and recall:

$$F_1 = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}$$

Experimental Results

MEKA using Mulan API:

Classifier	Precision	Recall	F1	Accuracy
SGD	0.179	0.339	0.758	70.4%
SMO	0.18	0.338	0.768	71.8%
Naïve Bayes	0.133	0.212	0.711	61.8%

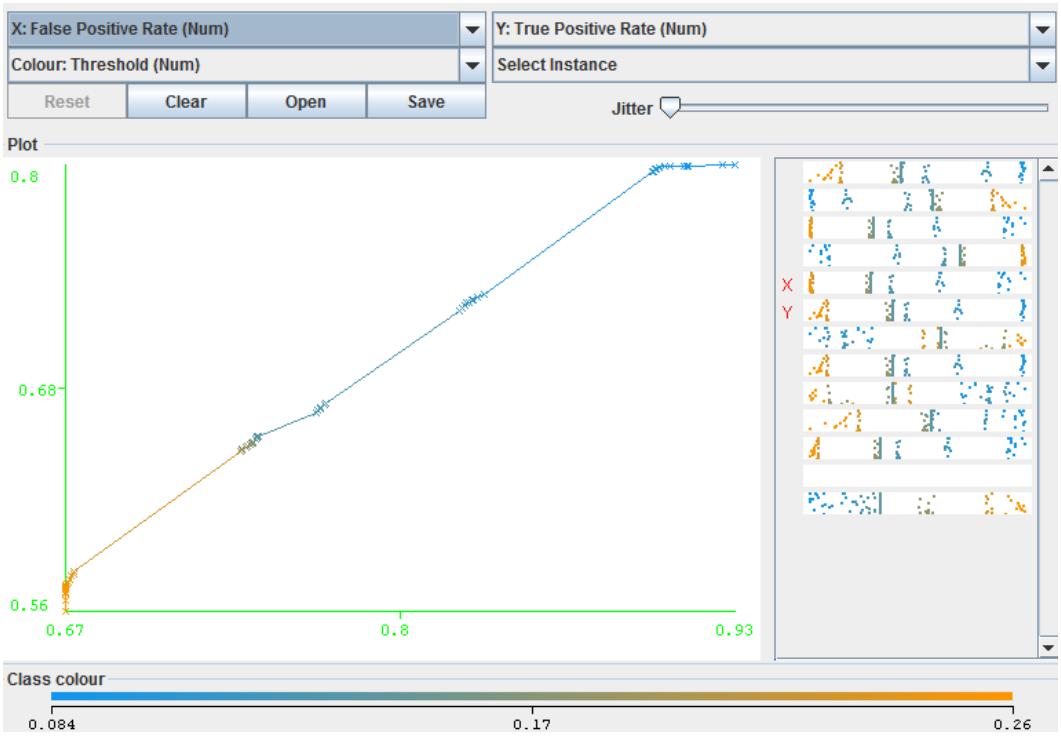
Performance measures of the classifiers used

Labels	Precision	Recall	F1	Accuracy
Java	1.000	1.000	1.000	100%
Android	0.185	0.018	0.03	70.6%
JavaScript	0.870	0.788	0.826	85.7%
Python	0.250	0.010	0.019	79.4%
Multi-threading	0.167	0.100	0.125	98.6%

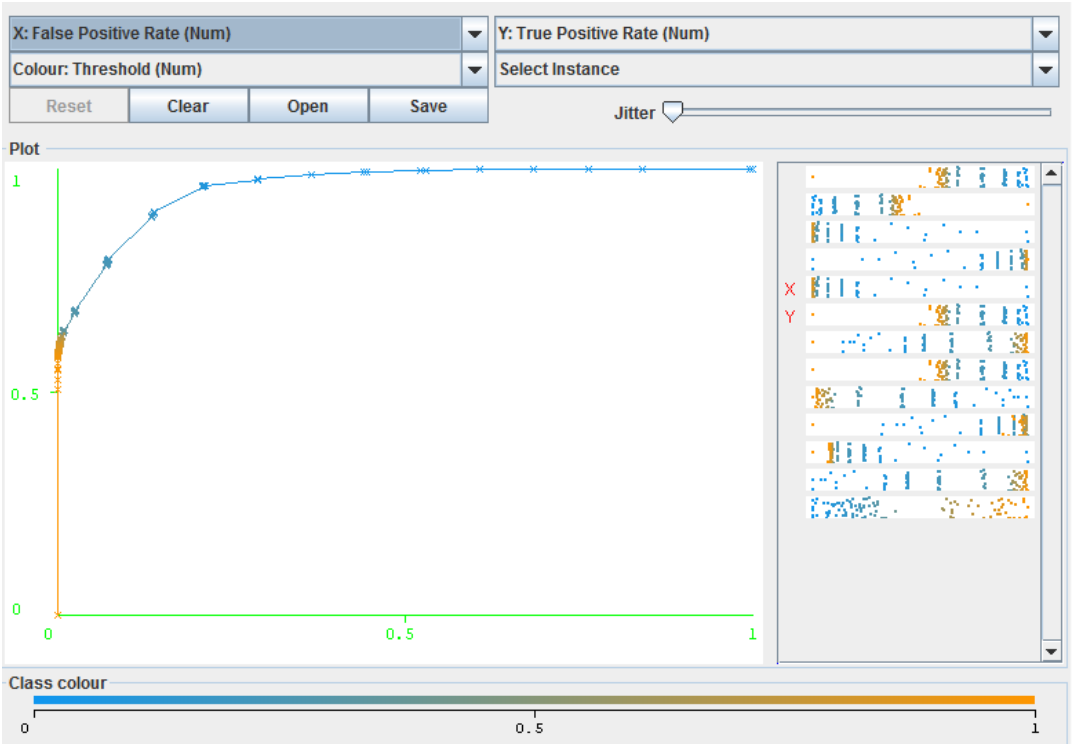
Performance measures of the most common labels

We see that for JAVA the accuracy is 100% due to the fact that the TP rate = 100%. Every Question in the test set has the label JAVA and they are correctly classified. We have experimented with MEKA to get the confusion matrix, unfortunately we MEKA does not support that.

Macro-Average Curve

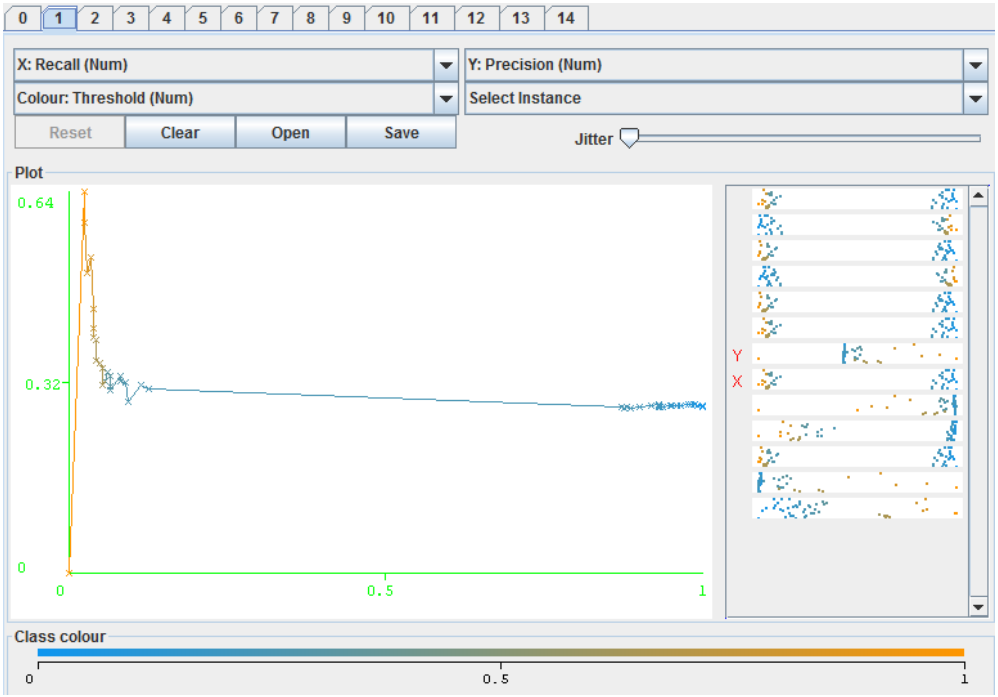


Micro-Average Curve

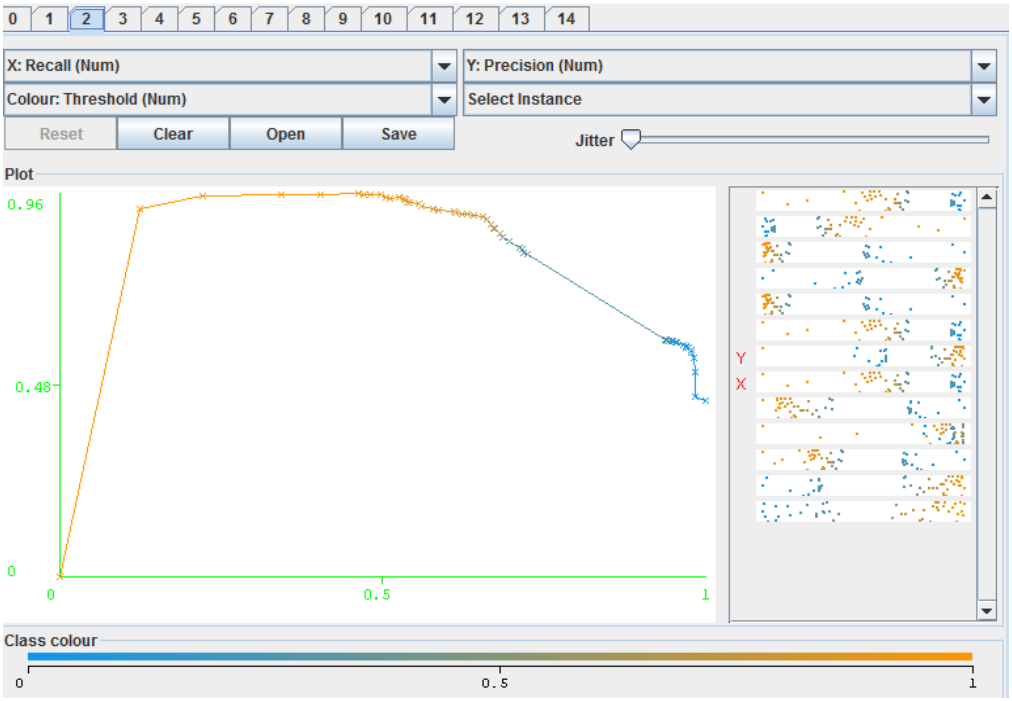


Precision-Recall curves

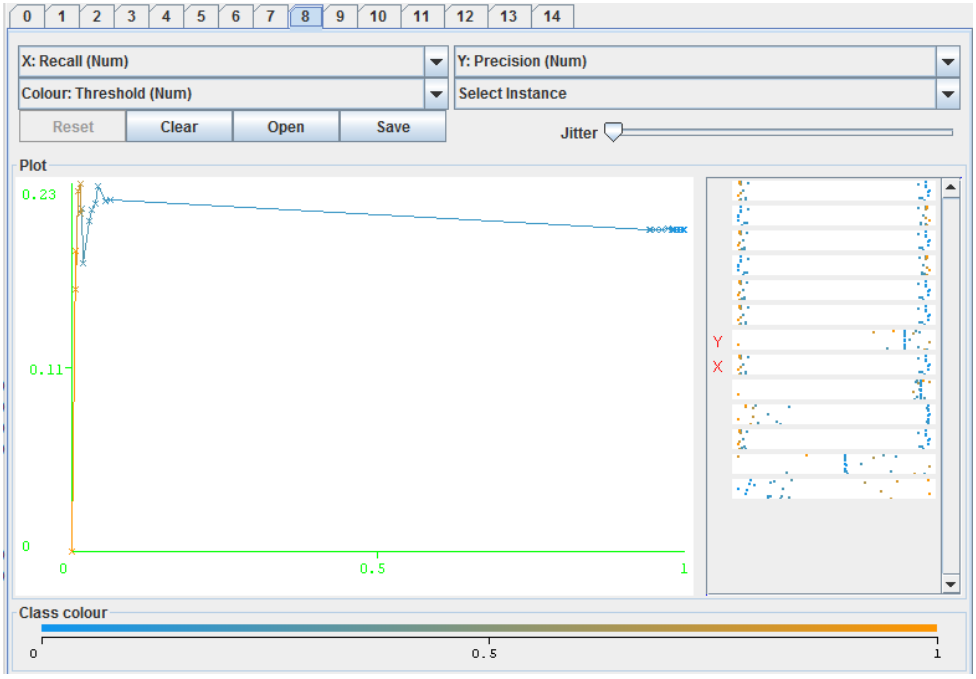
Android



JavaScript

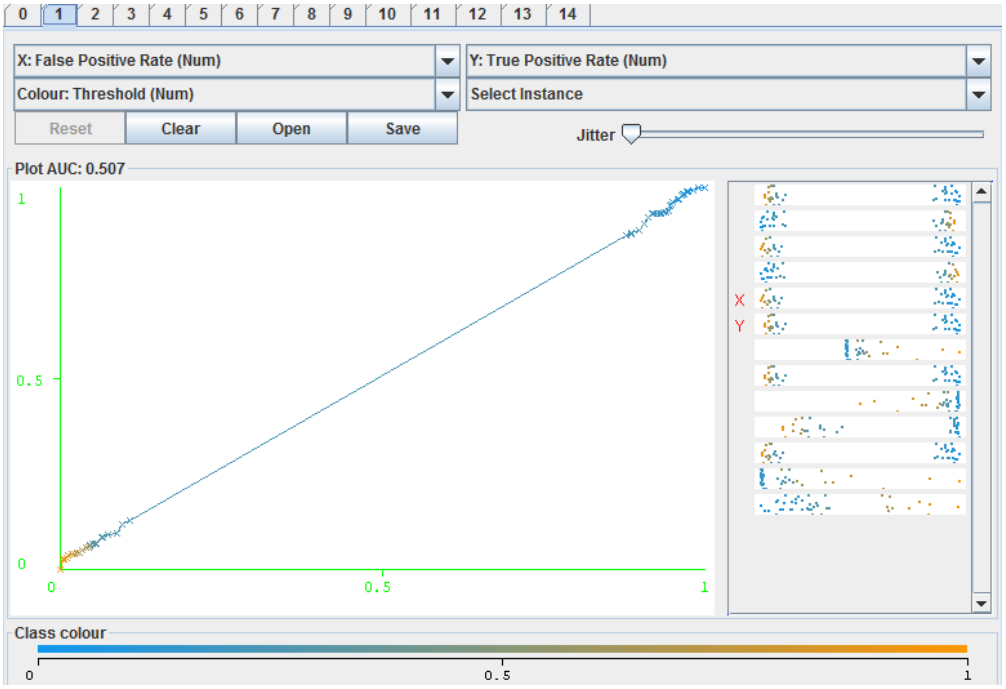


Python

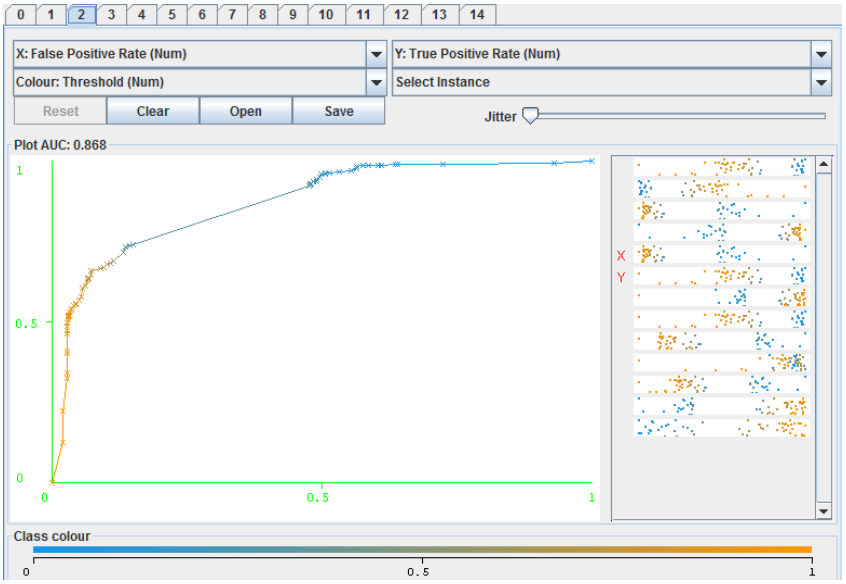


ROC Curves

Android



JavaScript



Scikit-learn:

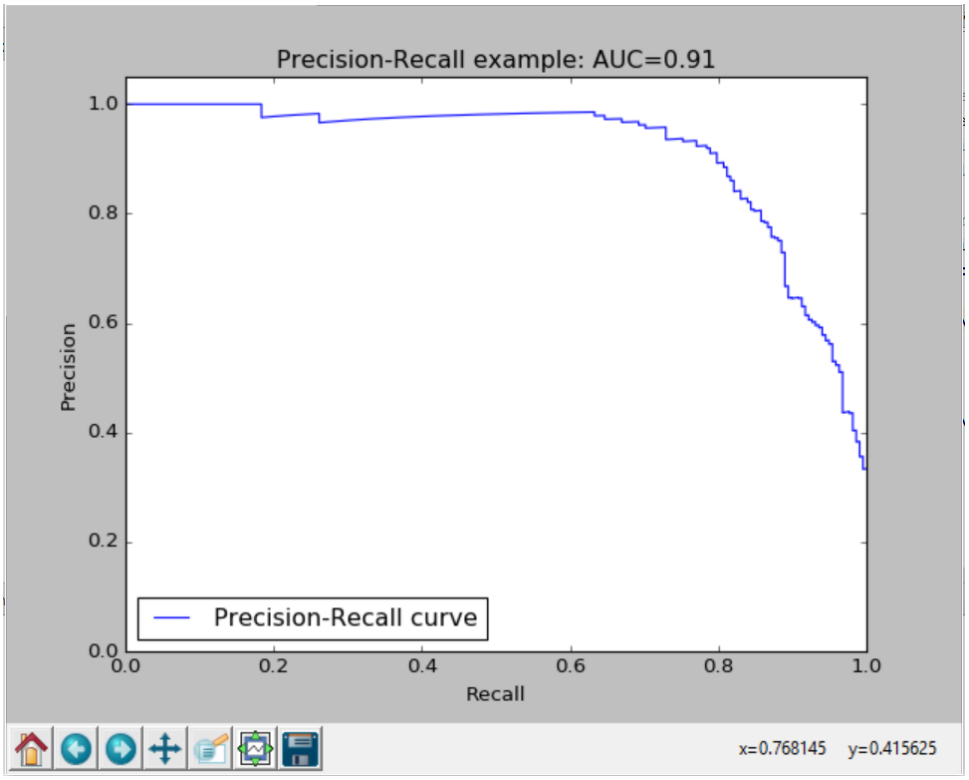
Feature	Coefficient score	Feature	Coefficient score
javascript	-1.6913	android	12.0921
django	-1.545	activity	4.9559
python	-1.4093	listview	4.5251
jquery	-1.4083	textview	3.6899
mysql	-1.2015	intent	3.5279
engine	-1.1508	edittext	3.4479
blackberry	-1.1438	webview	3.4079
swing	-1.1307	layout	3.3395
js	-1.101	apk	3.2841
java	-1.0909	asynctask	3.221
form	-0.9577	spinner	2.9194
web	-0.8941	bitmap	2.8836
ajax	-0.8288	emulator	2.6715
j2me	-0.8133	imageview	2.6403
div	-0.766	drawable	2.6042
model	-0.7543	mapview	2.4907
javafx	-0.7534	fragment	2.2645
pyqt	-0.7459	actionbar	2.1486
backbone	-0.7436	activities	2.0438
spring	-0.6992	alertdialog	2.0358

Most important features that are used in determining the labels

Accuracy	0.622
f-measure	0.765100671141
Average Precision	0.84

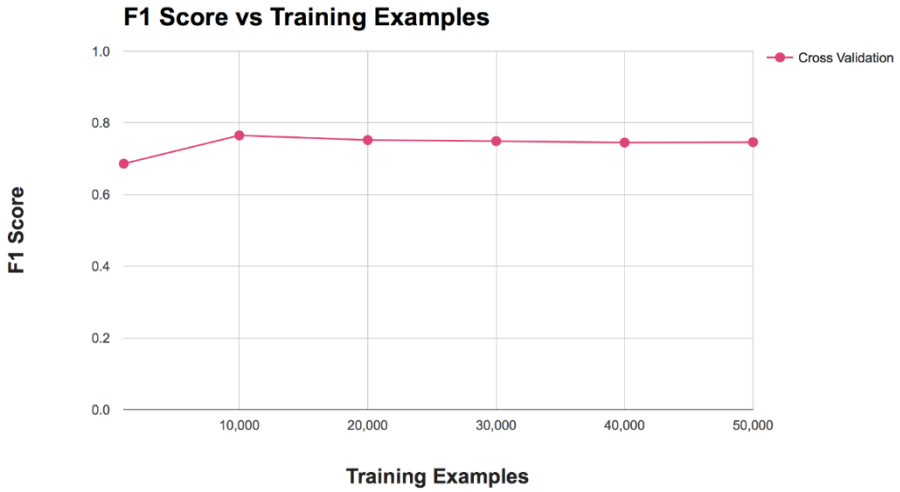
Performance measure of 10000 records using SGD Classifier

Precision-Recall curve



Precision-Recall curve for 10000 records using the SGD Classifier

F1 Score Vs Number of training instances



Future Work

- Ensuring similar counts of questions for each label in the training data can improve the overall accuracy of the classifier
- Analyzing the effect of using n-grams on overall accuracy of classifier

- Considering, Title as a baseline approach and verifying the accuracy by augmenting body text with the title text as an incremental approach.
- Considering a programming dataset, focusing on keywords in code fragments can improve tag prediction accuracy.

References

https://en.wikipedia.org/wiki/F1_score

http://www.jatit.org/volumes/Vol84No3/13Vol84No3.pdf

http://mulan.sourceforge.net/

http://meka.sourceforge.net/

https://www.kaggle.com/c/facebook-recruiting-iii-keyword-extraction

http://jmlr.csail.mit.edu/proceedings/papers/v28/bi13.pdf

http://www.jmlr.org/proceedings/papers/v9/dekel10a/dekel10a.pdf

http://lpis.csd.auth.gr/publications/tsoumakas-ijdwm.pdf

http://www.clei.org/cleiej/papers/v14i1p4.pdf

http://people.oregonstate.edu/~sorowerm/pdf/Qual-Multilabel-Shahed-CompleteVersion.pdf

http://scikit-learn.org/stable/auto_examples/plot_multilabel.html

https://www.youtube.com/watch?v=nNDqbUhtIRg

http://stanford.edu/~meric/files/cs229.pdf