MULTILABEL TEXT CLASSIFICATION OF CODING QUESTIONS & MATCHING LABELS OF CODING QUESTIONS FROM DIFFERENT SOURCES USING SUPERVISED LEARNING

Yash Mehta (5901611149)
ymehta@usc.edu
(https://github.com/yashmehta894/MultiLabel-text-Classification-of-Coding-Questions)

Abstract

The data on the web can be helpful in a lot of meaningful ways to build models that predict the results based on systems that learn via a machine learning algorithm. This project focuses on one such integration of web resources with learning system and making sure that we learn that data which we think is not useful but might just turn out to be what we are looking for in order to train our Machine Learning systems. In this paper, a Multilabel Classification problem is solved by a variety of methods and the best results are extracted for different types of labels. In multi-label classification, each of the input data samples belongs to one or more than one class labels. The proposed paper describes integration of two coding websites in such a way that you learn how to classify coding questions from one website (i.e. Geeksforgeeks) using variety of multilabel classification methods and then test your classification methods on coding questions from another website (i.e. Leetcode). We need to match the tags given to a particular set of questions on one website with the tags given to the same type of questions on another website. This paper explains the method used to match the entities and resolve the entities simultaneously. The results involve the TF-IDF with n-gram range 1 to have an edge over all the other methods used in multilabel classification. Lastly, Random Forest Classifier seems to have a strikingly good F1 Score as well.

Keywords: Multi-label Learning, Classification, Machine Learning, Coding Questions.

I. INTRODUCTION

In general, assignment of a single target label for the input sample instances corresponds to classification in machine learning. This type of classification is called single label classification as only one label from a set of disjoint labels is assigned to the input data. However, there are several conditions where the input data falls under more than one class. This condition of classification, where the input data correspond to a set of class labels instead of one, is called Multi-label classification. Initially, text-categorization [1-5] and medical diagnosis [6] are the primarily focus of multi- label classification problems. But recent realization of the omnipresence of multi-label prediction tasks in real world problems drawn more and more research attention to this domain.[7]

In recent years, several techniques are developed and are available in the literature that are used to perform multi-label classification. Gjorgi et al. in their paper [14] categorizes these techniques into three major categories. No single method performs uniformly well on a wide range of datasets is the result of an extensive comparison of multi-label methods has been performed by Gjorgi et al. [14]. Each method outperforms other in only a few datasets and performs less efficiently in other datasets.

The motivation behind this project is the increasing need for classifying problems so that a solver can directly dive into the specifics of the approach without wasting too much time on the categorization of the problem. During real time coding contests, it becomes utmost important to do problems quickly and this multi-label classifier provides that swift categorization of questions without major hustle.

The rest of this article is organized as follows. Section II gives a brief overview of multi-label classification problem and the model of the system. Section III presents the proposed approaches for solving the problems and how to go about things. Section IV provides the experimentation results and the results comparison with existing methods and related discussions. Finally, in Section V, the contribution of the article is summarized and concluded.

II. PROBLEM STATEMENT AND SYSTEM MODEL

The essential problem to be solved is multi-label classification of questions into various tags accurately. This results in essentially two problems to solve; one is extraction of high-quality training and testing examples from various websites and matching the question tags from each website so they can be classified as a single entity. (Entity Resolution)

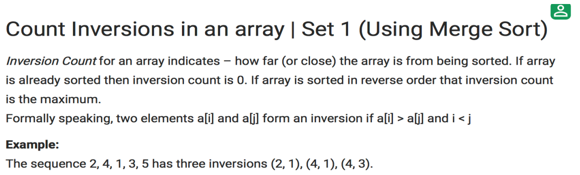
There are a few assumptions before we start moving ahead with the system. Any question which is to be classified falls within the desired categories for the classifier is trained. Any question with tag other than the pretrained classifier not be classified by the classifier.

As shown in Figure 2 it all starts with taking the input data and preprocessing it and doing analysis on that data as to how many samples are there per label, what is the distribution of the samples across specific labels, which are the most popular labels, etc. After that the data is split into 8:2 ration of training and testing set, the training data is put into a pipeline whose hyperparameters are tuned via GridSearchCV and the optimized encoder is used to evaluate the training data. Ultimately, the final F1- Score and other metrics are recorder. Finally, the entire dataset is retrained using the best encoder model selection and the final model is obtained.

The important steps are explained as follows:

- **Data Extraction:**

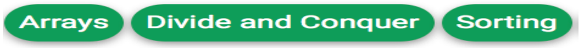
  The first source of data is Geeksforgeeks.com. This source of data was contained more than 2000+ questions. Each question had a title, question description and tags which were extracted. The following figure shows a question on Geeksforgeeks:



Count Inversions in an array | Set 1 (Using Merge Sort)

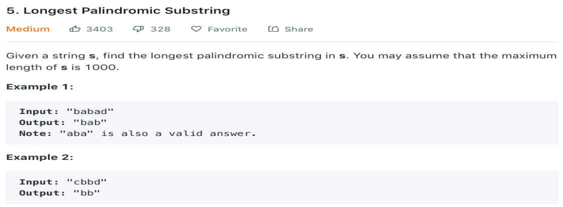
Inversion Count for an array indicates – how far (or close) the array is from being sorted. If array is already sorted then inversion count is 0. If array is sorted in reverse order that inversion count is the maximum.
Formally speaking, two elements a[i] and a[j] form an inversion if a[i] > a[j] and i < j

Example:
The sequence 2, 4, 1, 3, 5 has three inversions (2, 1), (4, 1), (4, 3).



Arrays Divide and Conquer Sorting

The second source of data is Leetcode.com. This source of data has a comprehensive list of questions along with tags as well were used as testing data. The following figure shows the detailed description of the questions as presented on Leetcode.



5. Longest Palindromic Substring

Medium 3403 328 Favorite Share

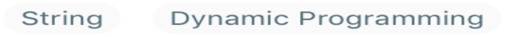
Given a string **s**, find the longest palindromic substring in **s**. You may assume that the maximum length of **s** is 1000.

Example 1:

```
Input: "babad"
Output: "bab"
Note: "aba" is also a valid answer.
```

Example 2:

```
Input: "cbbd"
Output: "bb"
```



String Dynamic Programming

Most categories don’t have enough training examples. So, the project picks the top labels that have sufficient training examples.

- **Matching question tags from both the websites**
  Some of the questions in both the websites were referred by the same category name but a lot of questions were referred by different category names. So, it became extremely important to match different category names to the same types of questions. The process which was followed to do that is described in the next section.
- **Training**
  The training data were question descriptions concatenated by question title with the labels given as 0 or 1 in the corresponding column.
  The methods used for classification are as follows:
  1. Linear SVC:
     The objective of a Linear SVC (Support Vector Classifier) in order to fit to the data, you provide, returning a "best fit" hyperplane that divides your data. some features are then. fed to the classifier to see what the "predicted" class is, after getting the hyperplane. [16] This makes this specific algorithm rather suitable for our use case, though you can use this for many situations.
  2. One-vs.-rest
     Training a single classifier per class, with the samples of that class as positive samples and all other samples as negatives. [16] This strategy requires the base classifiers to produce a real-valued confidence score for its decision, rather than just a class label; discrete class labels alone can lead to ambiguities, where multiple classes are predicted for a single sample.
  3. Random Forest Classifier
     An ensemble learning method for classification, regression and other tasks that operates by constructing a multiple decision trees at training

time and outputting the class that is the mode of the classes (classification) or mean prediction (regression) of the individual trees.[16]

4. K-Nearest Neighbor
   In k-NN classification, the output is a class membership. An object is classified by a plurality vote of its neighbors, with the object being assigned to the class most common among its k nearest neighbors.[16]

- Testing
  The testing of the classifier is done on sample of data taken from Leetcode. The data after testing comes as a series of predictions for the problem descriptions which is further evaluated.

III. SOLUTION APPROACH

The system model is shown in figure 2 which describes the entire flow in which the steps were followed. The first step was to extract data from two websites (i.e. Geeksforgeeks and Leetcode). This was done using JSoup library along with Selenium driver. After data extraction the most essential step was to match the categories coming from both the websites in a single entity. Figure 1 shows an example of two different tag names from different websites referring to same set. This is explained by following algorithm:

```
1.    For each tag on Geeksforgeeks:
      1.1.  Train a binary classifier with questions in that tag
            as positive samples and questions in all the other
            tags as negative samples.
      1.2.  Generate binary classifier for each tag.
2.    For each ambiguous tag on Leetcode for which entity
      resolution needs to be done:
      2.1.  For each of binary classifier pre-trained in step 1
            on Geeksforgeeks:
            2.1.1. Input the questions of that tag from
                   Leetcode in the binary classifier.
            2.1.2. Keep track of maximum number of positive
                   samples returned by every classifier.
3.    Return the classifier which gave maximum number of
      positive examples as the desired classifier for the am-
      biguous tag.
```

As for example, consider the tag Bit Magic from Geeksforgeeks. There will be a binary classifier pretrained for Bit Magic along with all the classifiers of Geeksforgeeks. When this Bit Magic binary classifier is applied to the set of examples from Leetcode under the tag Bit Manipulation, it gives maximum number of positive examples as compared to any other classifier. So, the questions in tag Bit Manipulation in Leetcode has to refer to tag Bit Magic in Geeksforgeeks.

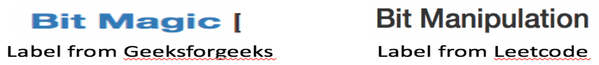


Fig 1

After the tags are successfully matched, we will train a number of classifiers on the data extracted from geeksforgeeks and then subsequently test it on the data extracted from leetcode. Now whenever a new question arrives on Leetcode or any other website we can classify quickly.

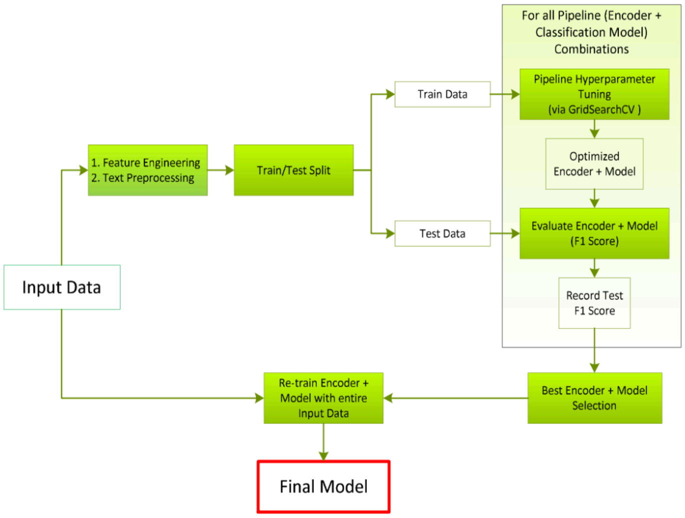

For all Pipeline (Encoder + Classification Model) Combinations
Pipeline Hyperparameter Tuning (via GridSearchCV)
Train Data
1. Feature Engineering
2. Text Preprocessing
Train/Test Split
Optimized Encoder + Model
Test Data
Evaluate Encoder + Model (F1 Score)
Input Data
Record Test F1 Score
Re-train Encoder + Model with entire Input Data
Best Encoder + Model Selection
Final Model

Fig. 2 Adapted from [15]

IV. EXPERIMENTAL RESULTS

- Precision Score

Precision is the proportion of the predicted correct labels to the total number of actual labels averaged over all instances.[16] The two notable algorithms to look at in Figure 3 are Random Forest Classifier and TF-IDF with ngram range of 1 and maximum df as 0.5. The reason for TF-IDF to perform well in all the tags is quite simple that in TF-IDF calculates the document and inverse document frequency for individual word thus taking into account uniqueness of the keyword. In coding questions, the type is heavily determined on a couple of same or similar words so it becomes deterministic in concluding the possible tags. Other than that, Random Forest Classifier performs relatively better than others except the tags like two pointer and sorting. In two pointer method and sorting method, the keywords involved does not define that level of uniqueness on which a random forest can discover a separate classifier for it so the scores might be low.

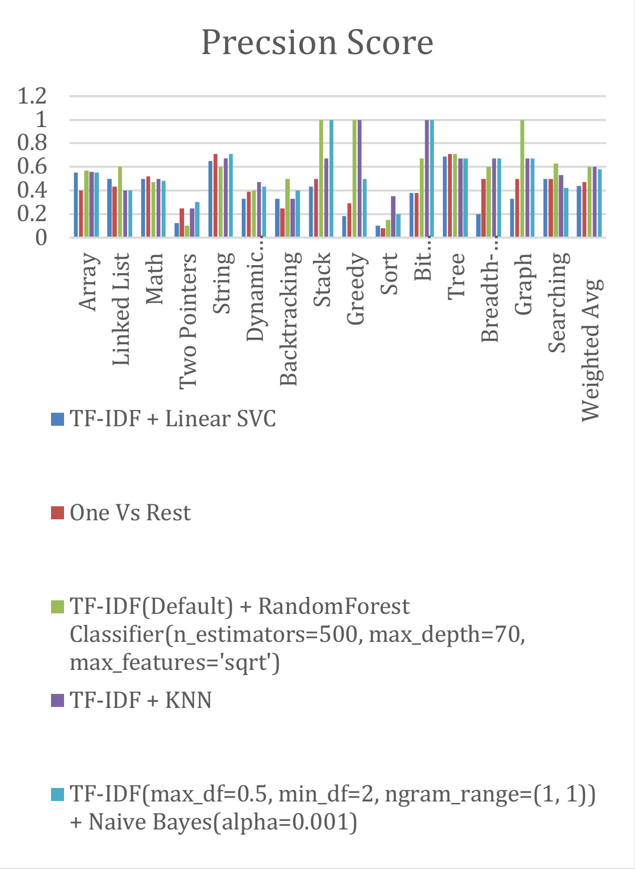


Figure 3

- Recall Score

Recall is the proportion of the predicted correct labels to the total number of predicted labels averaged over all instances. [16]The two notable algorithms to look at in Figure 4 are KNN and TF-IDF with ngram range 1. Both of them are doing exceedingly well with a good recall score throughout. Except the scores Random Forest Classifier has gone down in a couple of types like BFS and Two Pointer.

- F1-Score

F1 measure is given by the harmonic mean of Precision and Recall.[16]The two most trusted algorithm for classifying coding questions has to be Random Forest Classifier and TF-IDF with Ngram range of 1 as it is very evident from the Figure 5.

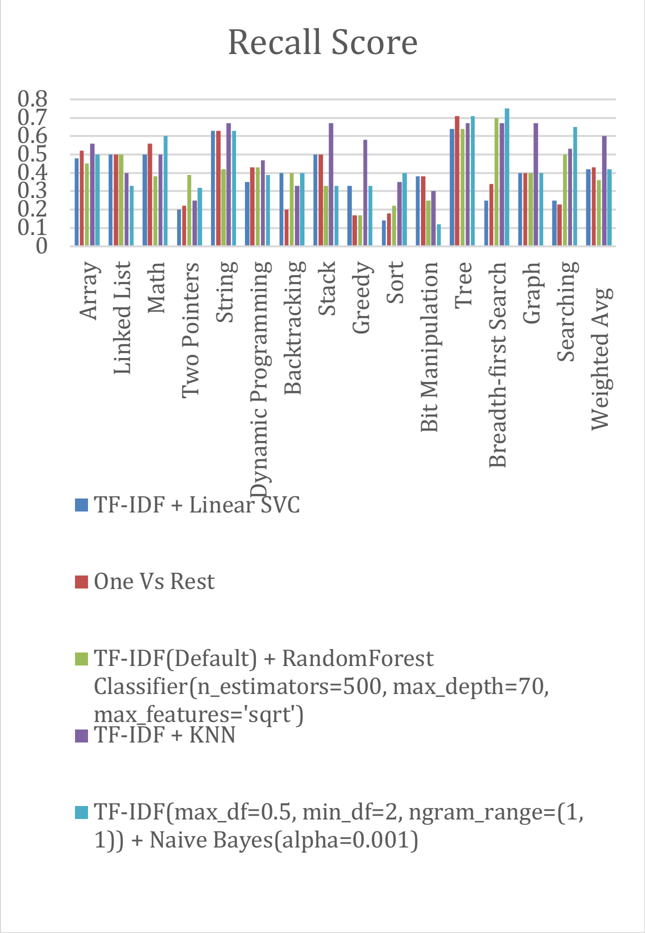


Figure 4

V. CONCLUSION

Out of 5 models used for prediction, the best one uses TF-IDF vectorizer and KNN Vectorizer to achieve an overall F1 Score of 0.71. Also, String and Math are the two most popular tags in coding questions. On an average a question is classified into 2 tags. One of the few strongly corelated tags are Dynamic Programming and String.

Hence, from this project, we conclude that the data on the web can be used to train Machine Learning systems and automate the human process of classifying questions in a Machine oriented way. The idea is to make use of the rich corpus of data available freely on the web and bring that into a format which can be learnt by the system and further help in predicting future results.

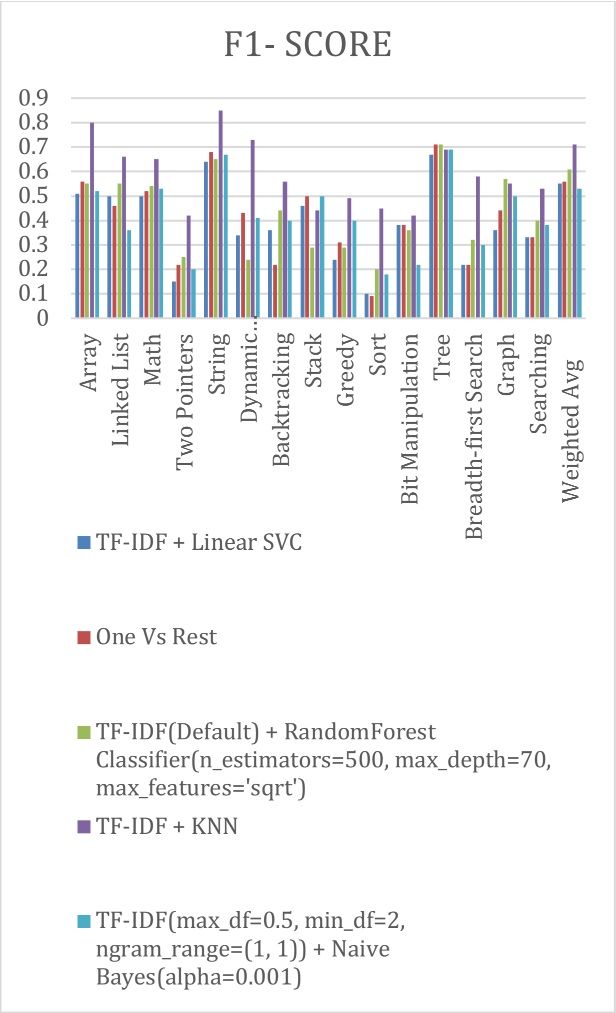


Fig 5

References

[1] Gonclaves T, Quaresma P, "A Preliminary approach to the multi-label classification problem of Portuguese juridical documents", Piers F.M., Abreu S.P. (Ed.), EPIA 2003, LNCS (LNAI), vol. 2902, pp. 435-444, Springer, Heidelberg, 2003.

[2] Joachims T, "Text categorization with support vector machines: Learning with many relevant features", Nedellec C, Rouveirol C (Ed.), ECML, LNCS, vol. 1938, pp. 137-142, Springer, Heidelberg, 1998.

[3] Luo X, Zincir Heywood A.N., "Evaluation of Two Systems on Multi- class Multi-label document classification", Hacid M.S., Murray N.V., Ras Z.W., Tsumoto S (Ed.), ISMIS 2005, LNCS (LNAI), vol. 3488, pp. 161-169, Springer, Heidelberg, 2005.

[4] Tikk D, Biro G, "Experiments with multi-label text classifier on the Reuters collection", Proc. Of the International Conference on Computational Cybernetics (ICCC 2003), Hungary, pp. 33-38, 2003.

[5] Yu K, Yu S, Tresp V, "Multi-label informed latent semantic indexing", Proceedings of the 28th annual international ACM SIGIR conference on Research and Development in information retrieval, pp. 258-265, 2005.

[6] Karalic A, Pirnat V, "Significance level based multiple tree classification", Informatica, 15(5), 12 pages, 1991.

[7] Grigorios Tsoumakas, Ioannis Katakis, and Ioannis Vlahavas, "Mining Multi-label Data", O. Maimon, L. Rokach (Ed.), Springer, 2nd Edition, 2010.

[14] Gjorgji Madjarov, Dragi Kocev, Dejan Gjorgjevikj, Saso Dzeroski, "An extensive experimental comparison of methods for multi-label learning", Pattern Recognition, Vol. 45, pp. 3084-3104, 2012.

[15] Multi Label Text Classification of movie genres using its plot(n.d.). Retrieved March 15, 2019, from https://github.com/shashankvmaiya/Movie-Genre-Multi-Label-Text-Classification

[16] Types of classification algorithms in Machine Learning Retrieved March 15, 2019, from https://medium.com/@Mandysidana/machine-learning-types-of-classification-9497bd4f2e14