Walchand College Of Engineering, Sangli.

(An Autonomous Institute)

Department
Of
Computer Science and Engineering

A Project Report
On

Stack Overflow Tag Prediction: A Machine Learning based approach

Submitted by

Pranav Sunil Raut	(2016BTECS00077)
Sourabh Shashikant Pukale	(2016BTECS00076)
Dhanashree Shekhar Phulkar	(2016BTECS00019)

Under the Guidance
of

Mr. Kiran. P. Kamble
Guide
Computer Scienece & Engg. Dept,
WCE, Sangli.

November 2019

Walchand College of Engineering, Sangli
(An Autonomous Institute)

Department
Of
Computer Science and Engineering

CERTIFICATE

This is to certify that the Project Report entitled, **"STACK OVERFLOW TAG PREDICTION"** submitted by Mr. Pranav Sunil Raut, Mr. Sourabh Shashikant Pukale, Ms. Dhanashree Shekhar Phulkar, to Walchand College of Engineering ,Sangli, India, is a record of bonafide Project work of course 3CS491 PROJECT-I carried out by him/her under my/our supervision and guidance and is worthy of consideration for the award of the degree of Bachelor of Technology in Computer Science & Engineering of the Institute.

Mr. Kiran. P. Kamble
Guide
Computer Sci. & Engg. Dept,
WCE, Sangli.

Dr.B.F.Momin
Head Of Department
Computer Sci. & Engg.Dept,
WCE, Sangli

Acknowledgement

We would like to thank our Project Guide Mr. K. P. Kamble for his review on an earlier draft of this paper, pointers to several pieces of related work, and for his many insightful discussions on the link between explanation in Machine Learning and Association Mining. We would also like to thank several others panel Members for critical input of an earlier draft: Dr. B. F. Momin, Mr. Siddharaj D. Pujari, Mrs. S. B. Shinde, and all the reviewers.

Declaration

I hereby declare that work presented in this project report titled **"STACK OVERFLOW TAG PREDICTION: A MACHINE LEARNING BASED APPROACH"** submitted by me in the partial fulfillment of the requirement of the award of the degree of **Bachelor of Technology (B.Tech)** Submitted in the **Department of Computer Science & Engineering, Walchand College of Engineering, Sangli**, is an authentic record of my project work carried out under the guidance of Mr. K. P. Kamble(Asst. Prof., Computer Science & Engg. Dept, WCE, Sangli).

Date :	Pranav Raut	Sourabh Pukale	Dhanashree Phulkar
Place : Sangli	2016BTECS00077	2016BTECS00076	2016BTECS00019

Abstract

Stack overflow is popular platform for programming community to solve each other's programming or conceptual problems. Stack Overflow relies on users to properly tag the programming language and concept of a question and it simply assumes that the programming language of the snippets inside a question is the same as the tag of the question itself. Naïve programmers face problems of incorrect tagging to questions due to lack of knowledge about dependencies about a a particular package, language, framework, etc. This leads to down voting and the question being ignored even if the problem was a genuine one.

We proposed a Machine Learning (ML) and Natural Language Processing (NLP) based approach to solve this problem by auto suggestion of tags. Analysis of data suggests that there are many dependencies involved in the tags which leads to the introduction of association rule mining as a big contributor. We used association rules on top of the output of the classifier to get a f1-score of 0.42.

Keywords: tag prediction, natural language processing, one-vs-rest, machine learning, associative mining.

Table Of Contents

1 Introduction

Stack overflow has been a popular resource for software development from a decade. Programmers new to the community heavily rely on stack overflow to tackle with errors and other problems faced during learning a new tool or technology. Stack overflow is a platform where if a user posts a question along with the tags properly describing the domain of the question, the question alert is sent to the domain experts who have built their expertise by answering the questions correctly in the same domain.

Stack overflow depends on the tags of the question to take it forward. But a new programmer in the community may lack knowledge of giving proper tags, this may lead to the question being down voted of the question which is failing to be relevant to the community to which it is forwarded. This may lead to demotivation for the new users. But proper relevant tagging to the question posted may add to the knowledge of the community.

Stack overflow even relies on the user to tag the code snippet posted in the question. Stack overflow relies on the tag of a question to determine how to typeset any snippet in it. If a question is not tagged with any programming language, then the code is not typeset; however, the moment a tag is added, the code is rendered with different colors for the different syntactic constructs of the language. These causes demand an auto tagging or auto tag suggesting system to replace the traditional system of manual tagging. We built a solution to tackle with this problem with the help of Machine learning and Natural language processing by treating problem of programming language tag detection and other conceptual tag detection separately.

1.1 Overview and Problem Statement

Stack overflow is a programming community platform to resolve problems and queries faced by programmers around the globe. Its objective is to only resolve bugs and doubts regarding any code snippets provided by the user and does not go with generic theoretical based questions. This introduces a template of input to be given by the users. Template consists of title, body and code snippet. To make sure that the bugs and doubts are solved by experts in that particular domain with in a quick succession, tags are assigned to the question. Tags describe the domain of the problem, programming language, etc. Tags assigned help the question reach to appropriate person with sufficient knowledge on that problem. This knowledge of the expert is analyzed by previous

questions answered, topic of interest chosen, etc. of that person.

The platform takes tags as input from the user. This can be a problem in the below cases:

- Naive programmer incorrectly tagging questions due to lack of knowledge regarding the dependencies, packages being used.
- Incorrect tags causing spamming of irrelevant questions to a particular domain expert.

Our Problem Statement is to make an automatic tag suggestion system to predict tags in the real time by analyzing the content of the question to help new programmers and prevent spamming.

1.2 Motivation

Stack overflow is used by millions of students to get answers for their technical doubts and questions. The need of assigning most relevant tags to the questions gave an idea to solve this problem with the existing advanced technologies and algorithms. The project intend to assign the tags so that the questions would be sent to appropriate experts from the domain to get the most accurate answers. Current system uses the method of self assigning tags to the questions. The users who are unaware of the domain and expect knowledge/answers from experts cannot assign tags by themselves. This is where our project helps to assign most relevant tags to the questions to get answers from persons who are experts in that domain.

1.3 Contributions of this work

Our approach constitutes of supervised machine learning and association rule mining. By analyzing the data we can conclude that we need to take more efforts for data processing. We first did basic preprocessing steps that include stemming, data in lower case, removing stop words using TF-IDF, removal of tags, etc. This gave us meaningful but still large data. In order to exploit maximum information from the data we focused of trimming data content to get maximum information. To do this we took full title of the question, first 400 chars and last 100 chars from the body of the question and entire code snippet to use as input to the supervised learning model. We trained a One vs Rest classifier to predict tags on the input data. We used the 6 GB dataset to generate association rules between the tags being predicted with a confidence of 70% and support of 1000. The proposed approach is definitely better than the traditional approach as it gives an insight about tags that should be included for the question being asked.

We used f1-score as a metric to analyze the results and with the trained model and generated rules as it maintains a balance between precision and recall. Our approach just with classifier gave a f1-score of 0.35 which was increased to 0.42 after the introduction of association rule mining.

2 Background and Related Work

We surveyed the recent literature and research in tag recommendation systems to get an idea of what improvements could be made. Tag Prediction comes under a broad domain of multi-label classification. The Multi-label classicification can be grouped into two categories: Problem transformation methods and Algorithm adaption methods.

- Nareshpalsingh Et al. (Dec 2017). "Multi-label Classification Methods: A Comparative Study" [**1**]
  1. Nareshpalsingh introduced various methods to tackle with the problem of multi-label classification in paper.
  2. Problem Transformation and Algorithm adaptions methods are explained in depth.

  Below listed are some of the prominent methods to solve multi-label classification problems.

1. Classifier Chains
   - In this, the first classifier is trained just on the input data and then each next classifier is trained on the input space and all the previous classifiers in the chain.
   - The classifiers are linked along a chain where each classifier deals with the single label classification problem associated with the label.
   - Each link in the chain is expressed with the 0/1 label associations of all previous links.
2. Label Powerset
   - In this, the problem is transformed into a multi-class problem with single multi-class classifier is trained on all unique label combinations found in the training data.
   - This approach take possible correlations between class labels into account. More commonly this approach is called the label-powerset method, because it considers each member of the power set of labels in the training set as a single label.
   - This method needs worst case $2^{|C|}$ classifiers, and has a high computational complexity.
3. One vs One
   - One vs One considers each binary pair of classes and trains classifier on subset of data containing those classes.

- It trains total $n*(n-1)/2$ classes. During the classification phases each classifier predicts one class.
- Single classifier in one vs one uses subset of data, so single classifier is faster for one vs one.

The above mentioned approaches are good to solve a multi-label multi-class problems but fail to analyse relations between tags. Association rule mining analyse the relations and generate rules used to predict all the relevent tags for the question. Our approach is the integration of one-vs-rest algorithm and FPGrowth algorithm.

3 Proposed Method/Algorithm

Machine Learning algorithm: one vs rest, Association Rule Mining algorithm: FPGrowth.

3.1 Problem Definition

To predict tags by analysing the title and body content of the question posted on stack-overflow.

3.2 Proposed Idea/System

Proposed methodology as shown in Figure 1:

1. Data Pre-Processing

   Data needs to be processed and converted to set of words which classifies the labels which belongs to the question. Dataset consists of 3 GB compressed file. It can be downloaded from kaggle website. Below mentioned are 5 steps:

   (a) Converting questions to lowercase

   - The title and body contains words with both cases they are converted to lowercase for standard format.

   (b) Separating code-snippets from Body

   - Content written inside ¡code¿ tag is separated and passed to programming language prediction.
   - Stack overflow data analysis says majority of questions contain code snippet which can be used to predict tag.

   (c) Removal html-tags, punctuation and non-alphabetic characters

   - Html tags, punctuation and non-alphabetic characters are irrelevant and do not help to predict labels.

   (d) Removing stop words using TF-IDF

- Removing all the stop-words present in the comments. Stop words are basically a set of commonly used words in any language. We can focus on the important words instead on common words.
- TF-IDF (term frequency–inverse document frequency), is a numerical statistic that is intended to reflect how important a word is to a document in a collection or corpus.
- Words in the document with a high tfidf score occur frequently in the document and provide the most information about that specific document.

(e) Use SnowballStemmer to stem the words

- Stemming is process which basically transform words with roughly the same semantics to one standard form. For example, for amusing, amusement, and amused, the stem would be amus.

text	tags
get valu public void event anoth im make applic android im use acceleromet sensor phone ly give null valu	java android accelerometer void
nan suppos two random variabl gt b gt joint pdf fabab sts let alon assist suggest would great appreci	probability reference-request probability-distributions
use piccard iter find solut ode let mntime n mathbbr use picard iter dot x ax x x f n obvious pattern integr g	functional-analysis differential-equations
hard search symbol googl ask instead lt look like comment doesnt work like html one line comment like pu	javascript symbols
first step python jython import start program python jython winxp later win tri import sever project edit a	python jython modularity
nan sinc googl releas googl app busi setup larg number account wit nister central locat anyon know solut	email user-management google-apps management gmail
mount error permiss deni lxc contain cifssamba share samba share mount fine ubuntu host root differ cor	ubuntu samba mount cifs lxc
flash game server suggest nodej red etc quick summari complet flash game similar tetri readi link ead nobl	flash node js multiplayer
nan observ strang bug code suspect tie way close form systeminvalidoperationexcept op gt exchandlerop	c# winforms backgroundworker
find date time document time date timed repres multipl way ex today best way	parsing time web-crawler information-retrieval
continu sha hash php possibl continu hash php say exampl start hash larg chuck data like ctx hashinitsha h	php hash sha
settingsdesignerc vs webconfig join project connect string defin settingsdesign connect string entriesshoul	web-config appsettings
open multipl instanc program linux say exampl open multipl instanc gedit editor wrote shell script l edit ge	linux bash shell command-line script
window batch add hostentri want use batch script add new entri host file automat use windo tranetd gtgt	windows- batch hosts
nan look function similar hoverint scroll function move sidebar stay put scroll thing movingcalcul take pla	jquery scrolling hoverintent

Figure 1: Data Preprocessing Result

2. Programming Language Prediction

- Programming language tag detection is to be one separately because if we analyze the past data, we understand than there is at least one programming language associated with the question.
- Programming languages being independent from the concepts and other programming languages, have to be treated separately for better results. We will be using Gradient Boosting Classifier to predict the programming language.

3. Using Association Rule mining to find co-relations among classes.

Association rule mining is a procedure which is meant to find frequent patterns, correlations, associations, or causal structures from data sets found in various kinds of databases such as relational databases, transactional databases, and other forms of data repositories.

Association rules are simple If/Then statements that help discover relationships between seemingly independent relational databases or other data repositories.

An association rule has two parts:

- an antecedent (if)
- an consequent (then)

An antecedent is something that is found in data, and a consequent is an item that is found in combination with the antecedent. For example:

"If the question asked has a tag of css, then html tag appears on 66% of css questions"

In the above association rule, css is the antecedent and html is the consequent. Association rules are created by thoroughly analyzing data and looking for frequent if/then patterns. Depending on the following two parameters, the important relationships are observed:

- Support: Support indicates how frequently the if/then relationship appears in the database.
- Confidence: Confidence tells about the number of times these relationships have been found to be true.

Some of the algorithms that use association rules are AIS, SETM, Apriori, FP-Growth.

4. One-vs-Rest Model

   - An intuitive approach to solving multi-label problem is to decompose it into multiple independent binary classification problems (one per category).
   - In an "one-to-rest" strategy, one could build multiple independent classifiers and, for an unseen instance, choose the class for which the confidence is maximized.
   - Here, N classifiers are used where N = number of classes, each class predict the probability of belonging of particular class.

3.3 System Architecture

Used One vs Rest Model to solve the Multilabel Multiclass Problem. Used Association Rule Mining to analyse the relations between tags and generated rules to assign relevant tags to the question.

3.4 System Analysis and Design

3.4.1 Requirement Specification

Tools and Technologies: Anaconda, Data Mining, Machine Learning, Natural language processing.
Software Requirements: Python 3.7.0, Anaconda 5.3.0, Google Colab
Hardware requirements: Machine with minimum 64GB RAM, 50GB free memory storage

3.4.2 Flowcharts / Activity Diagram

3.4.3 Design and Test Steps

Training:

1. Data Preprocessing.
    - lowercase conversion, removal of unwanted characters
    - Separate code snippets from body
    - Remove html tags
    - Remove stopwords
2. Generate Association rules among tags using FPGrowth algorithm
3. Generate Model using One-vs-Rest

Testing:

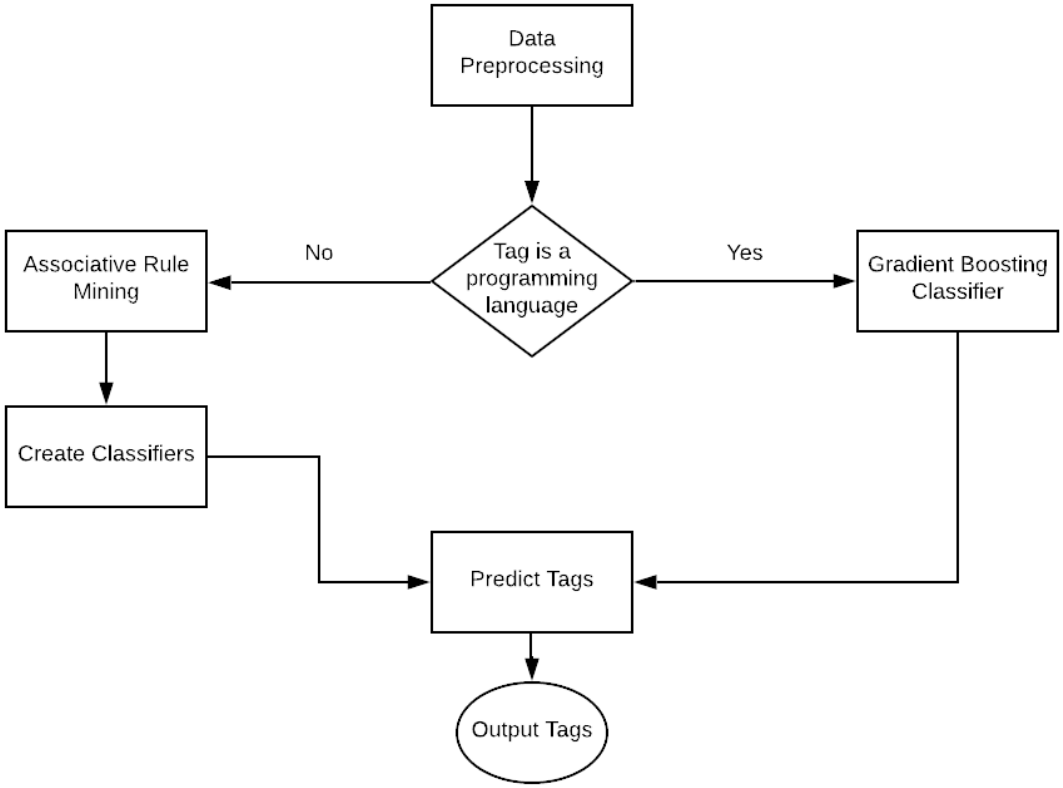

Figure 2: Activity Diagram of the proposed method

1. Load the model
2. Clean and stem the title and body of question
3. Predict the tags.

3.4.4 Algorithms and Pseudo Code

1. One-vs-Rest
2. Logistic Regression:

   Repeat

   1. Calculate gradient average
   2. Multiply by learning rate
   3. Subtract from weights

3. FPGrowth:

   1) The first step is to scan the database to find the occurrences of the itemsets in the database. This step is the same as the first step of Apriori. The count of 1-itemsets in the database is called support count or frequency of 1-itemset.

   2) The second step is to construct the FP tree. For this, create the root of the tree. The root is represented by null.

   3) The next step is to scan the database again and examine the transactions. Examine the first transaction and find out the itemset in it. The itemset with the max count is taken at the top, the next itemset with lower count and so on. It means that the branch of the tree is constructed with transaction itemsets in descending order of count.

   4) The next transaction in the database is examined. The itemsets are ordered in descending order of count. If any itemset of this transaction is already present in another branch (for example in the 1st transaction), then this transaction branch would share a common prefix to the root.

   This means that the common itemset is linked to the new node of another itemset in this transaction.

   5) Also, the count of the itemset is incremented as it occurs in the transactions. Both the common node and new node count is increased by 1 as they are created and linked according to transactions.

   6) The next step is to mine the created FP Tree. For this, the lowest node is examined first along with the links of the lowest nodes. The lowest node represents the frequency pattern length 1. From this, traverse the path in the FP Tree. This path or paths are called a conditional pattern base.

   Conditional pattern base is a sub-database consisting of prefix paths in the FP tree occurring with the lowest node (suffix).

   7) Construct a Conditional FP Tree, which is formed by a count of itemsets in the path. The itemsets meeting the threshold support are considered in the Conditional FP Tree.

   8) Frequent Patterns are generated from the Conditional FP Tree.

4 Performance Study

4.1 Implementation/Simulation Environment

Deep learning Server:

- Server comes with 128GB RAM, i7 6th Gen Processor.
- The dataset consisted of more than 20 Lakh samples with more than 500 unique tags, and each sample is associated with atleast 1 tag.
- To deal with the high computing power and RAM, deep learning server provided a great platform to build the moidel.

Libraries used:

- numpy
- pandas
- nltk
- re
- dill
- sklearn

Metrics Measured:

Definition of the Terms:

- Positive (P) : Observation is positive (for example: is an apple).
- Negative (N) : Observation is not positive (for example: is not an apple).
- True Positive (TP) : Observation is positive, and is predicted to be positive.
- False Negative (FN) : Observation is positive, but is predicted negative.
- True Negative (TN) : Observation is negative, and is predicted to be negative.
- False Positive (FP) : Observation is negative, but is predicted positive.

Recall:

Recall can be defined as the ratio of the total number of correctly classified positive examples divide to the total number of positive examples. High Recall indicates the class is correctly recognized (small number of FN).

Precision:

To get the value of precision we divide the total number of correctly classified positive examples by the total number of predicted positive examples. High Precision indicates an example labeled as positive is indeed positive (small number of FP).

$$\text{Recall} = \frac{\text{True positives}}{\text{Number of positives}} = \frac{\text{TP}}{\text{TP} + \text{FN}}$$

$$\text{Precision} = \frac{\text{True positives}}{\text{Number of predicted positive}} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

Figure 3: Precision and Recall Formula

High recall, low precision:This means that most of the positive examples are correctly recognized (low FN) but there are a lot of false positives.

Low recall, high precision: This shows that we miss a lot of positive examples (high FN) but those we predict as positive are indeed positive (low FP)

F-measure:

Since we have two measures (Precision and Recall) it helps to have a measurement that represents both of them. We calculate an F-measure which uses Harmonic Mean

in place of Arithmetic Mean as it punishes the extreme values more. The F-Measure will always be nearer to the smaller value of Precision or Recall.

$$F\text{ - measure} = \frac{2*\text{Recall}*\text{Precision}}{\text{Recall} + \text{Precision}}$$

Figure 4: F-measure Formula

4.2 Results and Analysis

1. One vs Rest Classifier The One vs Rest Classifier was trained on randomly sampled 2 million data points with 500 iterations. Observations concluded that top 500 tags had more than 85% of data samples which we used to train the classifier. An unknown class tag was created to avoid data loss. We trained on 0.5 million data points with complete text in the title, body and code snippet of the questions, this resulted in an f1-score of 0.35 without association rules. We experimented the data by taking full title text, first 400 characters and last 100 characters of the body and entire code snippet to train the classifier using 2 million data points. This resulted in f1-score to be 0.358

2. Association Rules by FP-Growth Association Rules were generated from all 6 million data samples with support = 1500 and confidence = 70%. FP-Growth on 6 million data points with above mentioned support and confidence generated 2130 rules.

Combining association rules on top of the results of One vs Rest resulted in an f1-score of 0.42.

linq sorting	c#
js svg	javascript
c# net-mvc visual-studio-	asp
dataset vb	net
j java logging	log
c# g net	oracle
jqgrid jquery-ui	jquery
net-mvc nhibernate	asp
d ipad iphone	cocos
javascript jquery-ajax json	jquery
asp entity-framework mvc	net-mvc
fancybox html	jquery
c# net- vb	net
mysqli query	mysql
class templates	c++
animation cocoa-touch	iphone
drop-down-menu net-mvc-	asp

Figure 5: Association Rules Generated

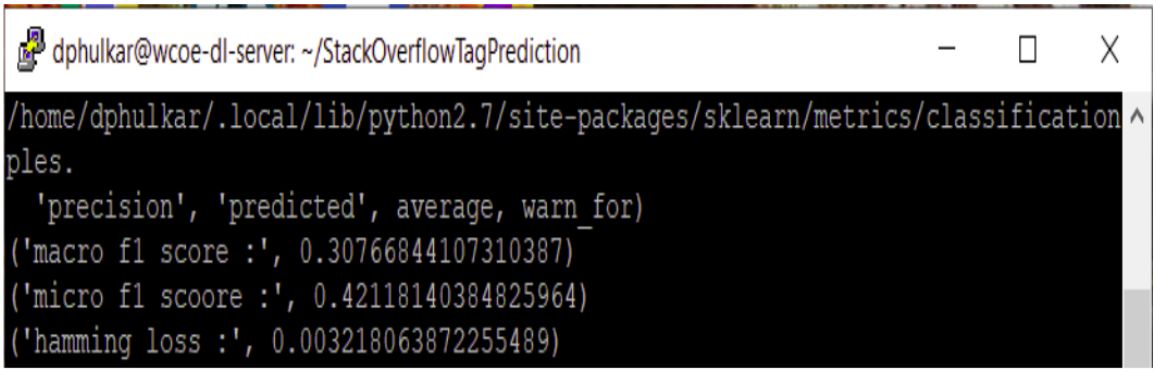

Figure 6: Final F1-score

4.3 Summary of performance study

Trained a multi label classifier to predict tags. Learned various techniques to solve a multi label classification problem. We used One vs Rest Method to deal with this problem, the One-Vs-The-Rest classifier strategy fits one binary classifier per class. We associate a set of positive examples for a given class and a set of negative examples which represent all the other classes. We used top 500 tags to deal with memory, resulting into 500 classifiers being built. We used logistic regression to deal with binary classification. For every logistic classifier we have a complexity of O((f+1)csE) where f=number of features, c= number of classes, s= number of

samples, E= number of epochs and let t be number of tags(500 in this case). Association mining rules were preprocessed contributing an complexity of O(n$\hat{2}$) where n= number of unique sets.

This resulted in overall complexity to be,

Training time Complexity: O(t*(f+1)csE + n$\hat{2}$)

Testing time Complexity: O(t*(f+1)cs + log(n))

Space complexity: Fp-Growth uses Space complexity of O(s). TF-IDF also makes large contribution in Space complexity.

5 Conclusions and Future Work

We started the project using stack overflow dataset and comparing the F1-score and time complexity with other proposed projects around the globe for this particular problem and the results are good enough. We achieved the F1 score of 0.42, which is good enough. We studied various algorithms like One-vs-Rest, Label Powerset, classifier chains, FPGrowth and analysed their time and space complexities. Future Work: One can try more algorithms and combinations to meet the need of time and space complexity along with the relations existing between tags to analyse more accurately to yeild better F1-score.

6 References

[1] Jadon Mayurisingh Nareshpalsingh, Prof. Hiteshri N. Modi (December 2017) "Multi-label Classification Methods: A Comparative Study" e-ISSN: 2395-0056, p-ISSN: 2395-0072.

[2] Kamel Alreshedy, Dhanush Dharmaretnam, Daniel M. German, Venkatesh Srinivasan and T. Aaron Gulliver(December 2018) "Predicting the Programming Language of Questions and Snippets of StackOverflow Using Natural Language Processing".

[3] "Solving Multi-Label Classification problems (Case studies included)", https://www.analyticsvidhya.com/blog/2017/08/introduction-to-multi-label-classification

[4] T. Saini and S. Tripathi, "Predicting tags for stack overflow questions using different classifiers," 2018 4th International Conference on Recent Advances in Information Technology (RAIT), Dhanbad, 2018, pp. 1-5. doi: 10.1109/RAIT.2018.8389059, URL: http://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=8389059&isnumber=8388962