Solution Submission Template

CT/DT Number: DT20184372894
Contestant Name: Rushirajsinh Parmar
College Name: Charotar University of Science and Technology

1. Background

Stack Overflow is the largest, most trusted online community for developers to learn, share their programming knowledge, and build their careers.

Stack Overflow is something which every programmer use one way or another. Each month, over 50 million developers come to Stack Overflow to learn, share their knowledge, and build their careers. It features questions and answers on a wide range of topics in computer programming and other related topics. The website serves as a platform for users to ask and answer questions, and, through membership and active participation, to vote questions and answers up or down and edit questions and answers in a fashion similar to a wiki or Digg. As of August 2019 Stack Overflow has over 1,10,901,490 registered users, and it exceeded 18123431 questions in Aug 25,2019. Based on the type of tags assigned to questions, the top eight most discussed topics on the site are: Java, JavaScript, C#, PHP, Android, jQuery, Python and HTML

Reference : https://data.stackexchange.com/stackoverflow/query/227868/select-count-from-users
https://data.stackexchange.com/stackoverflow/query/edit/849225

2. Your Understanding

The dataset consists of 6M entries (61,20,000) which consists of 'Title' , 'Body' and 'Tags' of the questions posted on StackOverflow.The dataset contains content from disparate stack exchange sites, containing a mix of both technical and non-technical questions. Here, 'Title' and 'Body' are the independent attributes whereas 'Tag' is the dependent variable. The task is to predict the tags (a.k.a. keywords, topics, summaries), given only the question text and its title. The dataset is quite messy and contains a lot of redundancy which needs to be removed using Preprocessing techniques. The dataset is also highly unbalanced out of a total 42,000 tags, the top 15 % (5,500) tags cover almost 99.04 % questions (explained in detail in later sections). Hence, we also need to try to reduce the bias nature of dataset

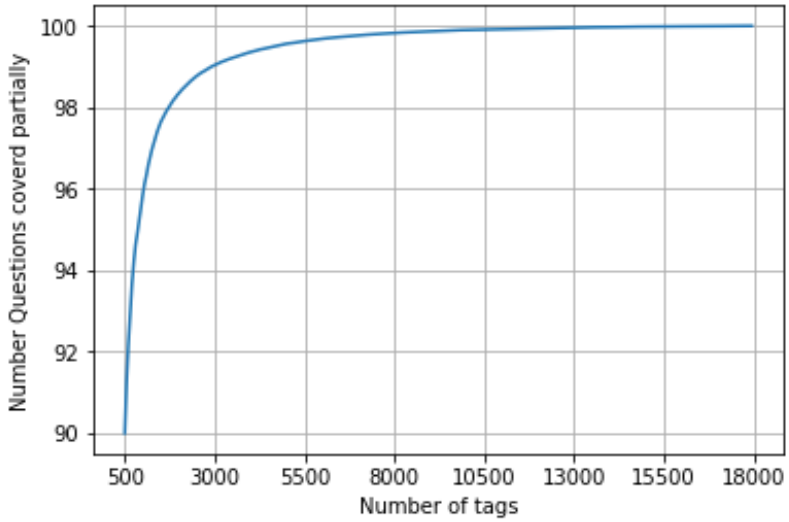
3. Scope

Due to the vast and robust amount of data entries in the csv file (total size : 8.6 GB) and considering the configurations of my laptop (8 GB RAM, 128 GB SSD) it isn't quite possible to load the csv file directly, hence as a more proper and effective alternative , I've chose to use 'Sqlite' database which is quite fast and more efficient to handle large databases.

The bottleneck of this challenge is the hardware configuration (computing power limitations). Hence I also tried to use Google Colab which offers a limited amount of free GPU usage but it wasn't quite sufficient as a result even though slow and encountering lagging issues I had to run the algorithm on local machine

After Preprocessing the data, the first step was to convert tags for multi-label problems, hence using the concepts of Binary Relevance. The next step is splitting the dataset into train and test and then featurizing the data. As per the evaluation criteria which takes " Mean F1 score" into account, the metric " Micro F1 score " should be used to evaluate our model.

We have a total of 42,000 unique tags considering all questions, but after analysis it turns out that by only using 5,500 tags wen can cover 99.04 % questions. This are the top 15% tags which cover 99% of the questions.




```
with  5500 tags we are covering  99.04 % of questions
```

As for choosing the classifier, I've used Logistic Regression with One Vs Rest classifier. One important thing to be noted here is that SVM (Support Vector Machine) cannot be used as it will create sparse matrix which eventually will lead us to memory error. Random Forest and Decision Trees also cannot be used as they do not generalise very well on large datasets.

It takes nearly 6-7 hours to train the model using Logistic Regression with One vs Rest Classifier. Hence, to improve the training time and improve performance it would be better to model less data points (0.5 M data points) ,add more weight to title and consider 500 tags only (covers 91 % questions)

After creating a new dataset and uploading to sqlite database, we need to featurize the data with Tf-Idf vectoriser and then apply Logistic Regression with One vs Rest classifier.

4. Out of Scope

Due to limitation in Computational power, I have used only 1M data points which covers 99% of tags, the rest amount of data (5 M) consists of hardly 1% tags which increases the complexity and takes more running time. If provided higher computational power, the whole dataset can be processed easily and for the same reason, few of the methods which can further enhance the efficiency of model could not be used as it cannot be run on local machine. As a part of further/ future enhancement I have listed down a few points

(1) Performing HyperParameter tuning on Alpha or Lambda for Logistic Regression to improve the performance using Grid Search

(2) Using One vs Rest classifier with Linear SVM

(3) Using SGD Classifier with Hinge Loss

(4) Using Smooth Non-Linear Optimizer

We have observed here that some tags appear more often than others leading to an ***'Imbalanced Dataset Problem'.*** Hence to overcome such problems, we can use methods like 'Random Oversampler', ' SMOTE' etc which try to reduce the gap between majority class and minority class and bring a balance between the dependent variables in turn decreasing the bias nature of dataset. Particularly for this problem statement, **Oversampling** (technique to increase the data points in minority class) won't help much because the gap between the amount of **majority tags (5,500)** and **Minority tags (36,500)** is much larger so we can use **Undersampling** (technique to decrease the data points in majority class) which might prove more beneficial.

5. Assumptions

General Assumptions :

(1) Corpora Language - English

(2) Phrasal Independence

Technical Assumptions

The whole dataset cannot be used to load and train the model due to limitation in processing power so we need to use 'sqlite' database from which data can be fetched easily and pre-processed. We have a total of 42,000 tags but out of that 5,500 tags cover 99.04 % of questions, hence only using 5,500 tags we can reduce the complexity and improve our model training time

6. Solution Approach

High Level Solution Approach

All the steps are included in detail below :

1.0 : Exploratory Data Analysis

1.1 : Data Loading and Cleaning

1.1.2 : Using Pandas with Sqlite to load the data

1.1.3: Counting number of rows

1.1.4 : Checking for duplicates

Creating a new database with no duplicates

1.2 : Analysis of Tags

1.2.1 : Total number of Unique tags

1.2.2 : Number of time a tag appeared (Plotting charts)

1.2.3 : Tags per question

1.2.4 : Most Frequent Tags

1.2.5 : Plotting Word Cloud, Top 20 Tags

1.3 : Cleaning and Pre-processing of questions

1.3.1 : Preprocessing

(1) Sample 1M data points
(due to limitation in computational power)

(2) Separate out Code snippets from body

(3) Remove special characters from question title and description (not in code)

(4) Remove Stopwords (Except " C ")

(5) Remove HTML Tags

(6) Convert all characters to small letters

(7) Use snowball stemmer to stem words

2.0 : Machine Learning Models

2.1 : Converting tags for multilevel problems (Countvectoriser)

2.2 : Splitting the dataset into train and test

2.3 : Featurizing the dataset

2.4 : Applying Logistic Regression with One vs Rest Classifier

2.5 : Modelling with less data points and giving more weight to title and 500 tags only

2.5.1 : Preprocessing of questions (same as before) just giving More weight to title

2.5.2 : Featurizing the data with Tf-Idf vectoriser

2.5.3 : Applying Logistic Regression with One vs Rest classifier using loss = Log, Hinge etc

Models/ Algorithms proposed

Logistic Regression with One vs Rest Classifier

7. Implementation Framework

General Implementation Approach

Sqlite , Python3 , Sklearn are the frameworks used to train the model

Includes Hardware and Software Details

Hardware configuration: 8 GB RAM , 128 GB SSD

Software : Jupyter Notebook

8. Solution Submission

https://github.com/rushirajsherlocked/TCS-HumAIn---Taxonomy-Classification

9. Appendix

Training time usually may take around 6-7 hours but can be made faster if you are using GPU or have access to 16/32 GB RAM

10. References

References :

https://stackoverflow.com/questions/15115765/how-to-access-sparse-matrix-elements

http://www.sqlitetutorial.net/sqlite-python/create-tables/

http://www.sqlitetutorial.net/sqlite-delete/

https://stackoverflow.com/questions/2279706/select-random-row-from-a-sqlite-table

http://www.bernzilla.com/2008/05/13/selecting-a-random-row-from-an-sqlite-table/

https://www.analyticsvidhya.com/blog/2017/08/introduction-to-multi-label-classification/

https://stats.stackexchange.com/questions/117796/scikit-multi-label-classification