RESEARCH ARTICLE OPEN ACCESS

Quora Duplicate Questions Detection Computer Engineering

Asst. Prof. Omprakash Yadav [1], Saikumar Kandakatla [2], Shantanu Sawant [3], Chandan Soni [4], Murari Indra Bahadur [5]

Department of Computer Engineering
Xavier Institute of Engineering
Mumbai - India

ABSTRACT

Quora is a spot to pick up and share information—about anything. It's a stage to pose inquiries and associate with individuals who actually contribute extraordinary bits of information and quality answers. This enables individuals to gain from one another and to all the more likely comprehend the world. More than 100 million individuals visit Quora consistently, so it's nothing unexpected that numerous individuals pose correspondingly worded inquiries. Different inquiries with a similar purpose can make searchers invest more energy finding the best solution to their inquiry, and cause scholars to feel they have to answer numerous variants of a similar inquiry. Quora values sanctioned inquiries since they give a superior encounter to dynamic searchers and essayists, and offer more an incentive to both of these gatherings in the long haul. As of now, Quora utilizes a Random Forest model to distinguish copy questions. In any case, we have attempted to utilize advance natural language processing techniques to characterize whether question sets are copies or not. Doing so will make it simpler to discover top notch answers to questions bringing about an improved encounter for Quora scholars, searchers, and perusers.

Keywords :— Quora, Natural language processing, XGBoost model, Duplicate Questions Detection.

I. INTRODUCTION

As we realize that Quora is a Q&A site where anyone can represent a request and discover solutions.But we have attempted to utilize some of the advance natural language processing techniques to understand whether question sets are duplicate or not. Doing so will make it extremely simple to discover great responses to questions bringing about an improved encounter for Quora readers,writers and searchers.So for that what we have done is initially we have understood the test and train dataset and than applying some of the text analysis techniques on it by using the libraries like matplotlib and seaborn in order to get insights from the dataset with the help of visual graphs and also we have used some of the other libraries like sklearn, numpy and pandas to understand the dataset in a much better way.After all the preprocessing and analysis of the test and train dataset in the end we have build a XGBoost model in order to predict whether the questions in the dataset are duplicate or not.

II. DATASET

The dataset of our project consists of training and testing data.Training dataset basically consists of 404290 number of question pairs for training and that of Testing data consists 2345796 number of question pairs for training.

A. Training Data

Training data basically consists if these four colums:

- id**:** Looks like a simple rowID
- qid{1, 2}**:** The unique ID of each question in the pair.
- question{1, 2}**:** The actual textual contents of the questions.
- is_duplicate**:** The label that we are trying to predict whether the two questions are duplicates of each other.

B. Testing Data

- id**:** It is a simple rowID
- qid{1, 2}**:** The unique ID of each question in the pair.
- is_duplicate column is not given in the testing data as we have to predict the values for the is_duplicate column.

III. DATA ANALYSIS

A. Text Analysis

Text Analysis is tied in with parsing writings so as to remove machine-lucid realities from them. The reason for Text Analysis is to make organized information out of free content substance. The procedure can be thought of as cutting and dicing stores of unstructured, heterogeneous records into simple to-oversee and decipher information pieces[3].

Text Analysis techniques is been only applied on the training data so as to avoid auto generated questions in the testing dataset. So we have plotted the graph of probability against number of characters in order to better understand the distribution of the train and test data for our duplicate question detection process.

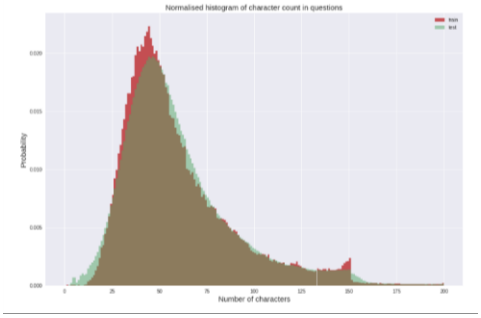

Fig. 1 Probability Vs Number of Characters

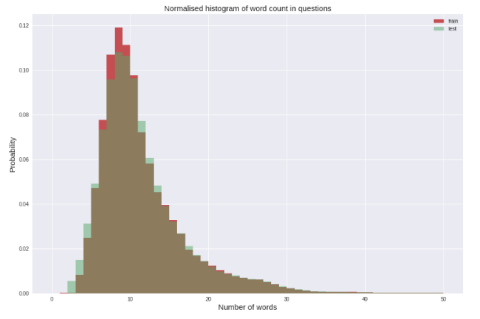

Fig. 2 Probability Vs Number of Words

B. Semantic Analysis

Semantic analysis depicts the way toward understanding normal language–the way that people convey dependent on significance and setting. It examines setting in the encompassing content and it dissects the content structure to precisely disambiguate the best possible significance of words that have more than one definition.In our project we have used semantic analysis just to understand how many questions are there with question marks, full stops, numbers so that we can get a better idea about the structure of our dataset and which will help us to ultimately do a good prediction for the duplicate questions[1].

C. Initial Feature Analysis

In this section we will be analyzing some of the important features of our dataset with the help of word share graph which will try to visually differentiate the duplicate and non.duplicate questions present in our dataset.Feature analysis is very important step in any model building so that we would be able to get better accuracy or validation score when we will execute our model thus providing we invest our efforts on working only on the most impacting features of available in our dataset and neglecting the rest ones which also results in saving our valuable time while building the model.

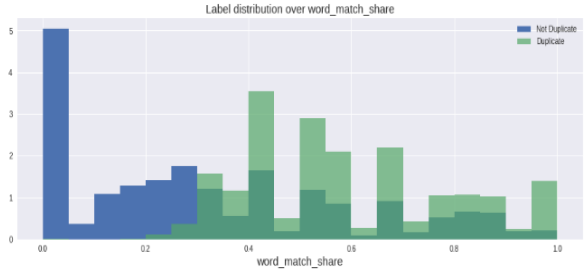

Fig. 3 Word Match Share

IV. DATA PREPROCESSING

A. Term frequency – Inverse Document Frequency

We are presently going to attempt to improve this features, by utilizing something many refer to as TF-IDF (Term Frequency-Inverse Document Frequency). This implies we gauge the terms by how phenomenal they are, implying that we care increasingly about uncommon words existing in the two inquiries than basic one. This bodes well, as we care progressively about whether "work out" shows up in both than "and" - as unprecedented words will be increasingly demonstrative of the substance. Tf-idf represents term recurrence opposite record recurrence, and the tf-idf weight is a weight frequently utilized in data recovery and content mining. This weight is a factual measure used to assess how significant a word is to a report in an assortment or corpus. The significance builds relatively to the occasions a word shows up in the record yet is counterbalanced by the recurrence of the word in the corpus. Varieties of the tf-idf weighting plan are regularly utilized via web indexes as a focal instrument in scoring and positioning a record's significance given a client inquiry. Tf-idf can be effectively utilized for stop-words separating in different subject fields including content synopsis and order. Ordinarily, the tf-idf weight is created by two terms: the first processes the standardized Term Frequency (TF), otherwise known as. the occasions a word shows up in an archive, isolated by the complete number of words in that record; the subsequent term is the Inverse Document Frequency (IDF), registered as the logarithm of the quantity of the reports in the corpus partitioned by the quantity of archives where the particular term shows up. TF: Term Frequency, which gauges how every now and again a term happens in a report. Since each report is distinctive long, it is conceivable that a term would show up significantly more occasions in long records than shorter ones. TF(t) = (Number of times term t shows up in a record)/(Total number of terms in the report)[1].

IDF: Inverse Document Frequency, which gauges how significant a term is. While figuring TF, all terms are considered similarly significant. Anyway it is realized that specific terms, for example, "is", "of", and "that", may seem a great deal of times yet have little significance.Typically, the tf-idf weight is composed by two terms: the first computes the normalized Term Frequency (TF), aka. the number of times a word appears in a document, divided by the total number of words in that document; the second term is the Inverse Document Frequency (IDF), computed as the logarithm of the number of the documents in the corpus divided by the number of documents where the specific term appears.

TF: Term Frequency, which measures how frequently a term occurs in a document. Since every document is different in length, it is possible that a term would appear much more times in long documents than shorter ones.

TF(t) = (Number of times term t appears in a document) / (Total number of terms in the document).

IDF: Inverse Document Frequency, which measures how important a term is. While computing TF, all terms are considered equally important. However it is known that certain terms, such as "is", "of", and "that", may appear a lot of times but have little importance.

IDF(t) = log_e(Total number of documents / Number of documents with term t in it)[1].

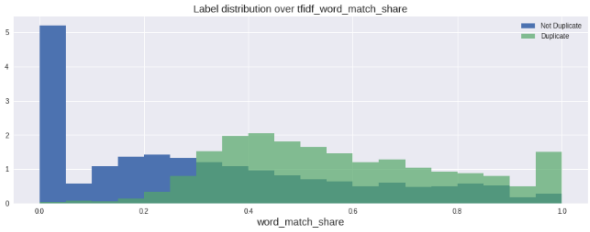

Fig. 4 TF-IDF Word Match Share

B. Area Under Curve

AUC, or Area Under Curve, is basically a metric for binary classification.Exactness manages ones and zeros, which means you either got the class mark right or you didn't. Yet, numerous classifiers can measure their vulnerability about the appropriate response by yielding a likelihood esteem. To register exactness from probabilities you need a limit to choose when zero transforms into one[2]. The most regular edge is obviously 0.5. How about we guess you have an eccentric classifier. It can find all the solutions right, however it yields 0.7 for negative models and 0.9 for positive models. Unmistakably, a limit of 0.5 won't get you far here. Be that as it may, 0.8 would be simply great. That is the general purpose of utilizing AUC - it thinks about every conceivable edge. Different limits bring about various genuine positive/bogus positive rates. As you decline the limit, you get all the more evident positives, yet in addition all the more bogus positives. From an arbitrary classifier you can expect the same number of genuine positives as bogus positives. That is the run line on the plot. AUC score for the case is 0.5. A score for an ideal classifier would be 1. Frequently you get something in the middle.

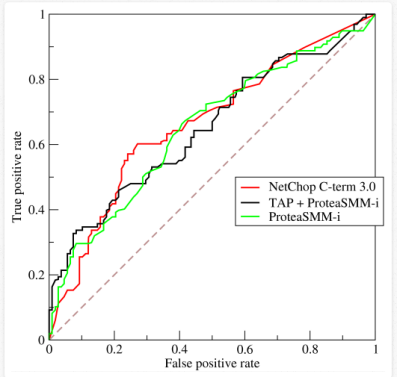

Fig. 5 Area Under Curve

C. Rebalancing the data

However, before we do this, we might want to rebalance the information that XGBoost gets, since we have 37% positive class in our training set, and just 17% in the test information. By re-adjusting the information so our training set has 17% positives. So by oversampling the negative class we have tried to rebalance our dataset. Oversampling and undersampling in data analysis are basically the techniques used to adjust the class distribution of a data set[3].

D. Logloss function

Logarithmic loss basically tries to measure the performance of a classification model where the prediction input is a probability value between 0 and 1[2].The objective of our model is to limit this worth. An ideal model would have a log loss of 0. Log loss increments as the anticipated likelihood separates from the genuine mark. So anticipating a likelihood of .012 when the real perception name is 1 would be terrible and bring about a high log loss. Log Loss considers the vulnerability of your expectation dependent on the amount it differs from the genuine name. This gives us a more nuanced see into the exhibition of our model. The diagram underneath shows the scope of conceivable log loss esteems given a genuine perception (isDog = 1). As the anticipated likelihood approachelog loss gradually diminishes. As the anticipated likelihood diminishes, be that as it may, the log loss increments quickly. Log loss punishes the two sorts of mistakes, yet particularly those predications that are sure and wrong!

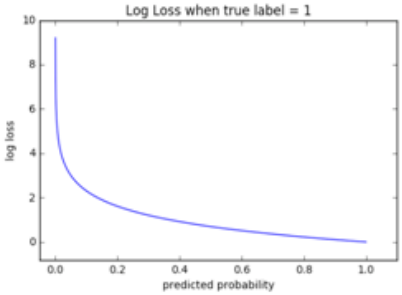

Fig. 6 Working of Logloss

IV. MODEL EXECUTION

Boosting is a group approach(meaning it includes a few trees) that begins from a more fragile choice and continues assembling the models with the end goal that the last forecast is the weighted entirety of all the more vulnerable chiefs. The loads are alloted dependent on the presentation of an individual tree. Gathering parameters are determined in stagewise way which implies that while computing the consequent weight, the gaining from the past tree is considered as well.XGBoost improves the inclination boosting technique considerably further[2]. XGBoost (extreme gradient boosting) regularizes information better than typical slope helped Trees.

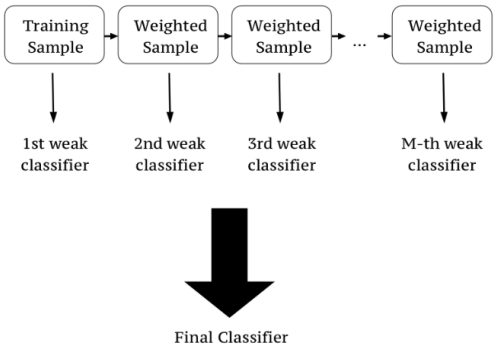

Fig. 7 Working of XGBoost Classifier

It was created by Tianqi Chen in C++ however now has interfaces for Python, R, Julia. XGBoost's target work is the entirety of misfortune work assessed over all the forecasts and a regularization work for all indicators (j trees). In the recipe fj implies an expectation originating from the jth tree[2].

$$obj(\theta) = \sum_{i}^{n} l(y_i - \hat{y}_i) + \sum_{j=1}^{j} \Omega(f_j)$$

Loss function relies upon the undertaking being performed (grouping, relapse, and so on.) and a regularization term is portrayed by the accompanying condition:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

Initial segment (γT) is liable for controlling the general number of made leaves, and the subsequent term ($12\lambda\sum Tj=1w2j$) looks out for the scores. Science Involved Unlike the other tree-building calculations, XGBoost doesn't utilize entropy or Gini records. Rather, it uses slope (the blunder term) and hessian for making the trees[2].

V. OUTPUT

Finally after building the XGBoost model we have executed the algorithm for around 50 epochs after which we are able to get a stable train logloss and valid logloss value with the help of the log loss function which we have used while building the model.After which we are simply saving the output of our prediction model in a csv file named simple_xgb.csv.

	test_id	is_duplicate
2	0	0.031617
3	1	0.360195
4	2	0.364868
5	3	0.000796
6	4	0.197357
7	5	0.017268

Fig. 8 simple_xgb.csv

VI. CONCLUSION

Quora is a spot to pick up and share information—about anything. It's a stage to pose inquiries and interface with individuals who contribute interesting bits of knowledge and quality answers. This enables individuals to gain from one another and to all the more likely comprehend the world.Thus we have finally built a model using some of the advance natural language processing techniques in order to help reader,writers and seekers find quality answers of the questions.

VII. REFERENCES

[1] Natural Language Processing By Steven Bird,Ewan Klein and Edward Looper.
[2] XGBoost With Python, Gradient Boosted Trees With XGBoost and scikit-learn By Jason Brownlee.
[3] Data preprocessing techniques for classification without discrimination by Faisal Kamiran and Toon Calders.