Quora Question Pair Similarity

Nissar Ahmed Pinjari
MS in Computer Science
Khoury College of Computer Sciences, Northeastern University
pinjari.n@northeastern.edu

Sai Krishna Gadde
MS in Electrical and Computer Engineering
College of Engineering, Northeastern University
gadde.sa@northeastern.edu

Bhavana Pemmasani
MS in Electrical and Computer Engineering
College of Engineering, Northeastern University
pemmasani.b@northeastern.edu

Srinidi Somasundaram Subbulakshmi
MS in Electrical and Computer Engineering
College of Engineering, Northeastern University
somasundaramsubbul.s@northeastern.edu

***Abstract*—Machine Learning has become the pioneer of every industrial sector where businesses are able to obtain useful insights by efficiently dealing with data. Quora is a social question and answer platform where users can collaborate on sharing their knowledge about topics they know. To make such a platform more user-friendly it would be great if we can identify patterns of questions that have already been asked and provide users with relevant search results. Our paper introduces algorithms to classify if a question asked is a duplicate or not.**

***Index Terms*—Machine Learning, Quora, duplicate, NLP, classifier**

I. Introduction

It's possible to learn and share knowledge on Quora about anything. It serves as a forum for questions and connections with contributors of original insights and excellent responses. As a result, people are able to grasp the world better and benefit from each other's knowledge.

It's hardly surprising that many questions on Quora are similar in wording given that over 100 million people visit the site each month. Multiple questions with the same objective can make readers feel as though they must respond to various variations of the same question, while also making seekers spend more time looking for the best solution to their problem. Canonical questions are highly valued on Quora because they provide active writers and seekers a better experience and more long-term value.

II. Business Objectives

To improve the number of users accessing quora each day, it is very crucial to give them a satisfying user experience. This happens when the mis-classification rate is low or menial. We evaluate the probability of a pair of questions to be duplicates so that you can choose any threshold of choice.

III. Data

The data set consists of two csv files : train.csv and test.csv. Both of them consists of 6 columns in it indicating the id, question id's (qid1, qid2),the questions (q1 and q2), and a column indicating if a question is duplicate (is_duplicate).There are 404,290 rows in train.csv.

Data set source: https://www.kaggle.com/c/quora-question-pairs
Below are the examples of data point:
"id","qid1","qid2","question1","question2","is_duplicate"

1. "0","1","2","What is the step by step guide to invest in share market in india?","What is the step by step guide to invest in share market?","0"
2. "1","3","4","What is the story of Kohinoor (Koh_i_Noor) Diamond?","What would happen if the Indian government stole the Kohinoor (Koh_i_Noor) diamond back?","0"
3. "7","15","16","How can I be a good geologist?","What should I do to be a great geologist?","1"
4. "11","23","24","How do I read and find my YouTube comments?","How can I see all my Youtube comments?","1"

IV. Exploratory Data Analysis: Basic Feature Extraction

Exploratory Data Analysis (EDA) is a technique used to analyze trends and patterns in data using visual techniques like graphs and statistical representations. The python porter-stemmer module is used to retrieve the stem of each word. For example : the words 'driving', 'drove', 'driven' reduces to the root word 'drive'. First ,we read the total number of data points in the train.csv file. We then try to calculate the number of duplicates using 'is_duplicate' id. The plot in Fig .1. shows the classification based on the is_duplicate id.

Later, the data set is analyzed to figure out the number of unique and duplicate questions . The plot in Fig.2. shows the number of unique and repeated questions.

Finally, the number of occurrences of each question is found and the results are plotted .

A. Basic Feature Extraction :

We now construct some additional features to mark the frequency of qid1's (freq_qid1), frequency of qid2's (freq_qid2) , length of q1 (q1len), length of q2 (q2len), number of words in question 1 (q1_n_words), number of words in question 2

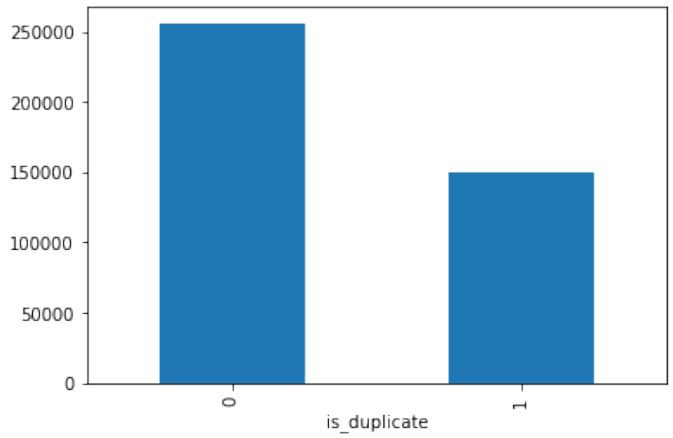


Fig. 1. Distribution of data points among output classes

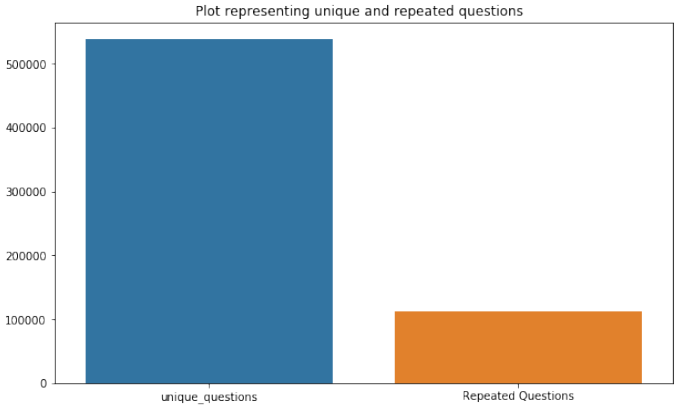


Fig. 2. Number of unique and repeated questions

(q2_n_words), number of common words in question 1 and 2 (word_common), total number of words in question 1 and 2 (word_total), shared word count (word_share), sum total of freq of q1 and q2 (freq_q1 + freq_q2), absolute difference of freq of q1 and q2 (freq_q1 - freq_q2). Plot in Fig.3. shows the number of occurrences

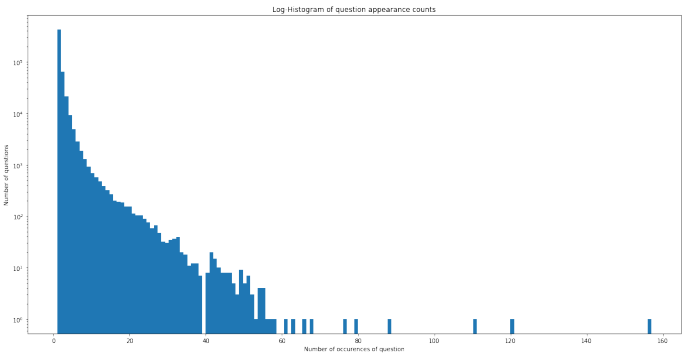


Fig. 3. Number of occurrences of each question

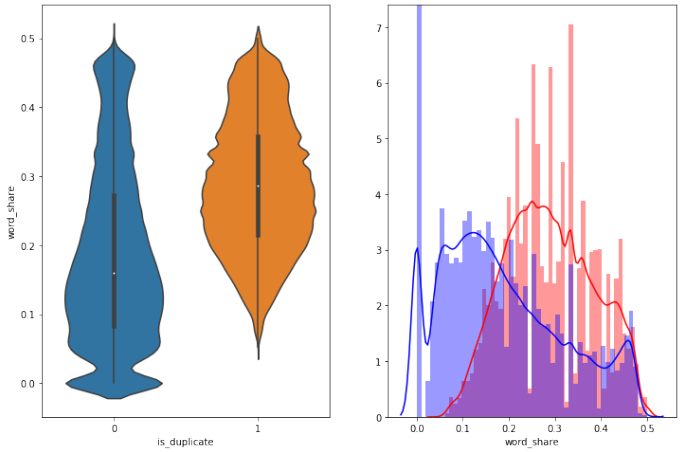


Fig. 4. Distribution for Average word-share

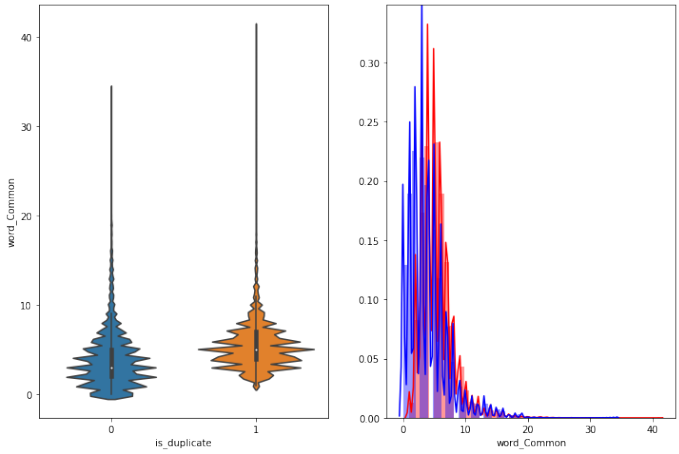


Fig. 5. Distribution for word-common feature in similar and non-similar questions

B. Feature : Word Share

Word share is given by : $wordshare = \frac{wordcommon}{wordtotal}$ The normalized word share distributions shown in Fig.4. display considerable overlap on the far right-hand side which implies high word similarity.

C. Feature: Word Common

The violin and the normalized plots shown in Fig.5. display that the word_common feature in similar and non_similar questions are highly overlapping.

D. Splitting:

We build train and test by randomly splitting in the ratio of 70:30.

E. Cleaning:

Everything is converted to lower-case. Contractions were removed. Currency symbols are replaced with currency names. Hyperlinks are removed. Alpha-numeric characters are removed. Inflections are removed with word-lemmatizer. HTML tags are removed.

V. EDA : Advanced Feature Extraction

A. Text preprocessing

Preprocessing of text involves removing of html tags, punctuations , stemming the word , removing stop words, and expanding contractions. The training data set is evaluated for few other advanced features. These include :

$$cwc\_min = \frac{common\_word\_count}{min(len(q1\_words),len(q2\_words))}$$

$$cwc\_max = \frac{common\_word\_count}{max(len(q1\_words),len(q2\_words))}$$

$$csc\_min = \frac{common\_stop\_count}{min(len(q1\_stops),len(q2\_stops))}$$

$$csc\_max = \frac{common\_stop\_count}{max(len(q1\_stops),len(q2\_stops))}$$

$$ctc\_min = \frac{common\_token\_count}{min(len(q1\_tokens),len(q2\_tokens))}$$

$$ctc\_max = \frac{common\_token\_count}{max(len(q1\_tokens),len(q2\_tokens))}$$

$$last\_word\_eq = int(q1\_tokens[-1] == q2\_tokens[-1])$$

$$first\_word\_eq = int(q1\_tokens[0] == q2\_tokens[0])$$

$$abs\_len\_diff = abs(len(q1\_tokens) - len(q2\_tokens))$$

$$mean\_len = \frac{(len(q1\_tokens)+len(q2\_tokens))}{2}$$

These features are finally merged with nlp_features_train.csv. Fuzzywuzzy is a python library that is used for string matching or pattern identification. The fuzz module contains methods like token_sort_ratio which sorts a given string alphabetically and compare them with a simple ratio . Similarly token_set_ratio tokenizes both the strings but instead of directly sorting and comparing, the tokens are split into two groups : the insertion and remainder which are used to build the comparison string. fuzz_ratio and fuzz_partial_ratio gives the percentage of accuracy between strings, either entirely or as n-length sub strings.

VI. Analysis of Extracted Features

A. Plotting word clouds

Word cloud is a technique to show which words are frequent among a given group of words. To infer the frequency of words, we create word cloud of duplicate and non-duplicate question pairs . The word cloud for a duplicate question pair indicates the words that occur frequently and makes a question sound duplicate while the wordcloud for a non-duplicate question pair shows words that recur and still does not make a question duplicate.

B. Pair plot of features

We then obtain pair for both the classes (0= not a duplicate, 1= duplicate based on the is_label feature). These representations gave a clear idea of the similarity between the two classes based on the taken metrics.//

Below are the visualizations obtained for the above mentioned features, along with token sort ratio, and fuzz ratio.

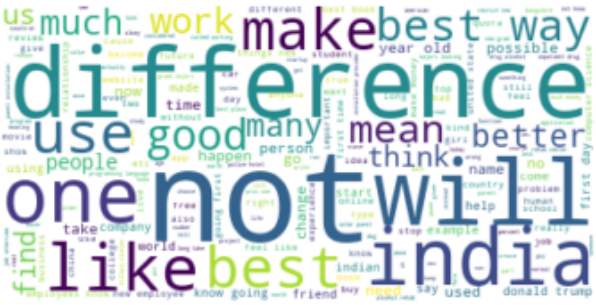


Fig. 6. Visualization of word cloud for non-duplicate pairs

Fig. 7. Visualization of word cloud for duplicate pairs

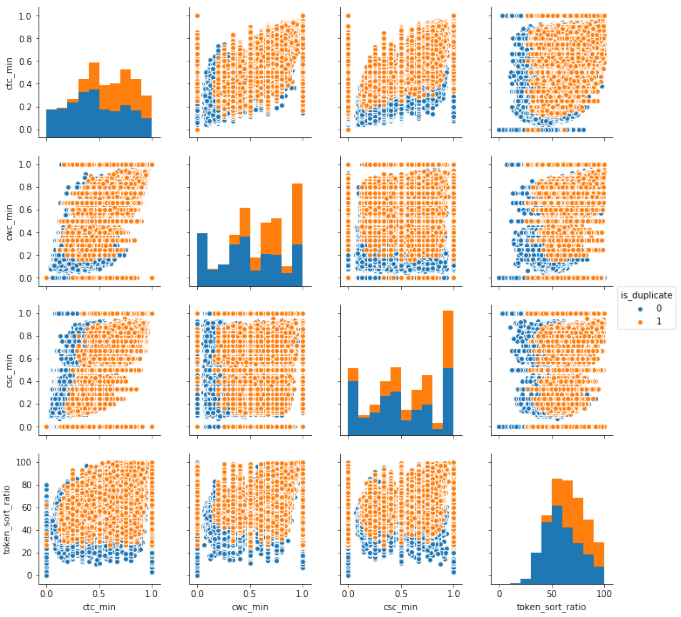


Fig. 8. Pair plot of features

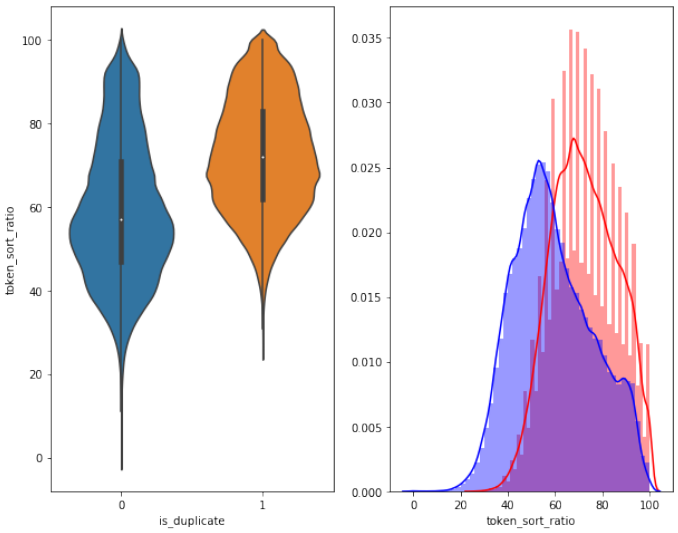


Fig. 9. Visualization of Token sort ratio

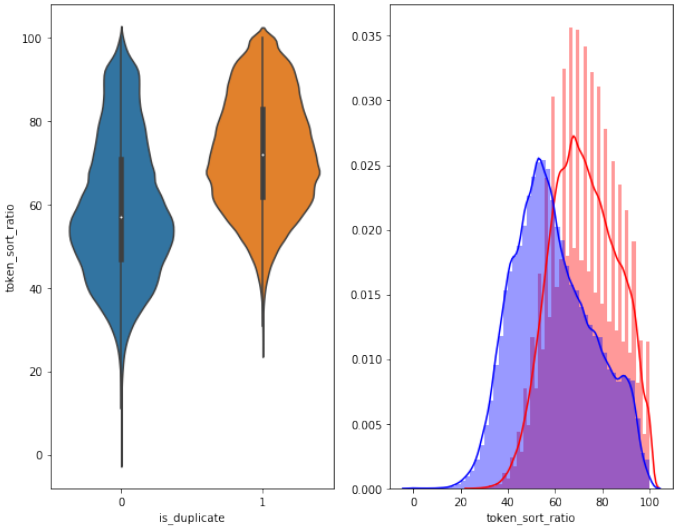


Fig. 10. Visualization of Token sort ratio vs is_duplicate

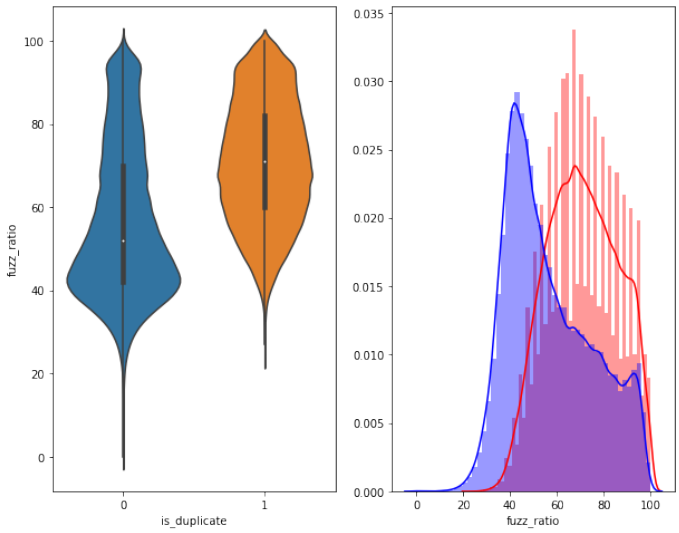
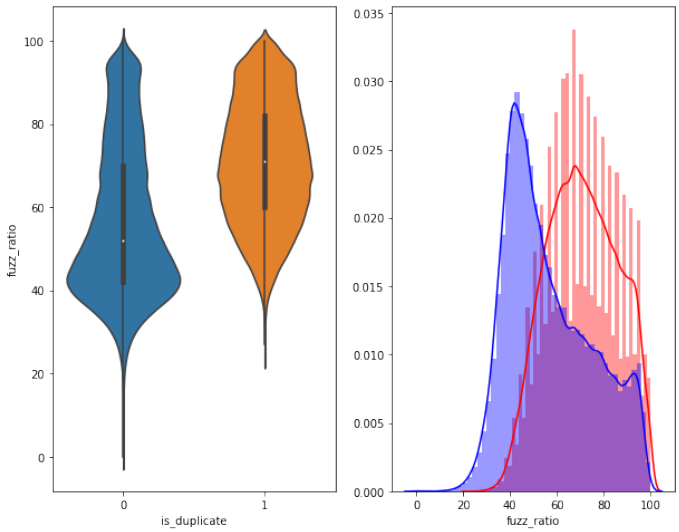


Fig. 11. Visualization of fuzz ratio



C. *Visualization using t-SNE*

t-SNE is a very useful technique to measure the similarity between words . It takes a group of high-dimensional vocabulary word feature vectors and later compresses them into 2 dimension x,y pairs. The idea behind this is to keep similar words closer on the plane , while maximizing distance between words that are not similar. Dimensionality reduction help observe similarities in data more closely in addition to giving a better visualization

VII. Featurizing text data with TFIDF weighted word vectors

TF-IDF is a measure of how relevant a word is to a document in a collection of documents. The main difference between term frequency and IDF is that term frequency alone doesn't take the importance of a term into account. IDF focuses on the importance of words in a document/page based on the uniqueness when compared to other documents/pages. Both of these methods have been used in information retrieval, but are mostly used in combination for more effective information retrieval. The higher the TF-IDF score the more important or relevant the term is and as a term gets less relevant, its TF-IDF score approaches to 0. After evaluating the TF-IDF score, we convert each question to a weighted average of word2vec vectors by these scores. spaCy is a python library that is used to compare two objects and make a prediction of how similar they are.

A. *Dimensionality of data for modelling*

1. Number of features in nlp dataframe : 17
2. Number of features in preprocessed dataframe : 12
3. Number of features in question1 w2v dataframe : 384

4. Number of features in question2 w2v dataframe : 384
5. Number of features in final dataframe : 794

VIII. Evaluation Metrics

The evaluation metrics used here include : log loss and confusion matrix (with 0-1 loss values). These metrics are chosen over accuracy due to their ability to work with probability estimates.

A. Log-loss matrix

In log-loss matrix case, the classification is done based on the probabilities. How closely the prediction probability matches the associated actual or true value is indicated by log-loss (0 or 1 in case of binary classification). The higher the log-loss number, the more the predicted probability deviates from the actual value. The formula for log-loss is as follows:

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} log(p(y_i)) + (1-y_i).log(1-p(y_i))$$

B. Binary Confusion Matrix

A confusion matrix is a table-based method of displaying the effectiveness of your prediction model. The number of predictions the model made when it rightly or erroneously categorised the classes is indicated by each entry in a confusion matrix. A binary classification problem has only two classes to classify, preferably a positive and a negative class. Now let's look at the metrics of the Confusion Matrix.
True Positive (TP): It refers to the number of predictions where the classifier correctly predicts the positive class as positive.
True Negative (TN): It refers to the number of predictions where the classifier correctly predicts the negative class as negative.
False Positive (FP): It refers to the number of predictions where the classifier incorrectly predicts the negative class as positive.
False Negative (FN): It refers to the number of predictions where the classifier incorrectly predicts the positive class as negative.
Accuracy: It provides you with the model's overall accuracy, or the percentage of all samples that the classifier correctly classified. Formula for confusion matrix is: $\frac{TP+TN}{TP+TN+FP+FN}$
Misclassification rate: It reveals what percentage of predictions were incorrect. It is also known as Classification Error. It can be calculated using (1-Accuracy).

IX. Machine Learning Models

A. Random Model

First, we generated a random model (using the rand function) and tested on it to figure out the worst case log-loss metrics. The distribution of class predictions were equal between correct and incorrect labels and this is evident from the confusion matrix of Fig.13. The model outputted a log loss value of 0.887242646958, which is very bad and as expected from a random classifier.

Then we make use of sci-kit learn to model better Linear classifiers (SVM, logistic regression, etc.). The

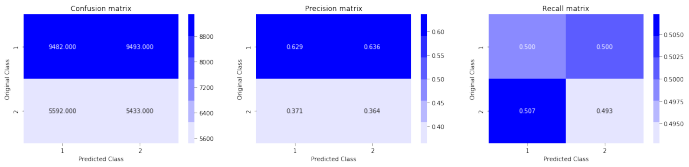


Fig. 13. Random Model Performance

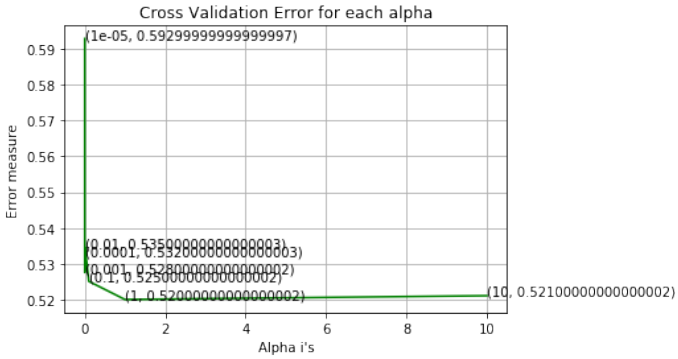


Fig. 14. Cross Validation on Logistic Regression Model

sklearn.linear_model.SGDClassifier is utilised to model Support Vector Machine (SVM) and Logistic Regression with Stochastic Gradient descent (SGD) learning: the gradient of the loss is estimated each sample at a time and the model is updated along the way with a decreasing strength schedule (aka learning rate). In this SGD classifier, the choice of the model is done based on the loss function opted within the parameters. Also, this classifier has parameters for regularization.

B. Logistic Regression Model

The SGD classifier is tuned to logistic regression model by choosing loss function as "log" within the parameters. l2 regularization is selected for the model with alpha being the regularization parameter. The prediction is implemented as a probability estimate through (predict_proba) while fitting the model. Probability estimates are used for further inference and enhancement of algorithm rather than just considering absolute predictions without having an idea of the influence factors. Based on the predictions, the model is then cross validated over a range of alpha to find the optimal alpha value with minimum log loss error measure.

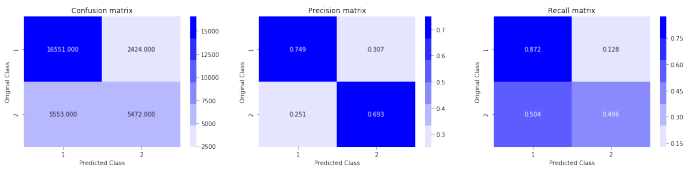


Fig. 15. Logistic Regression Model Performance

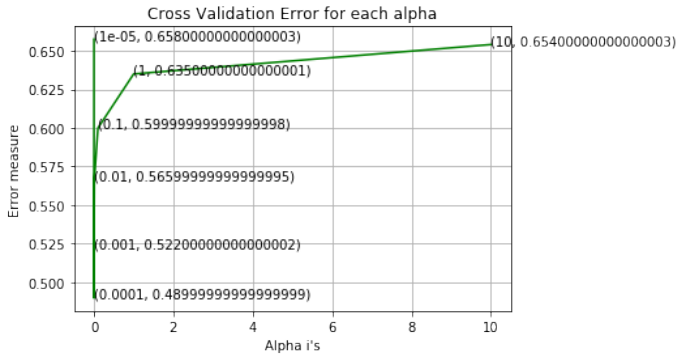


Fig. 16. Cross Validation on SVM Model

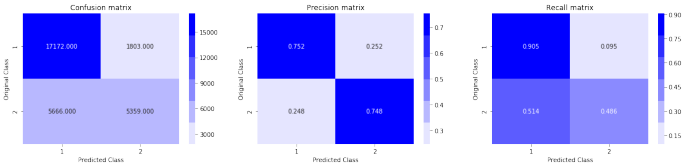


Fig. 17. SVM Model Performance

Fig.14. shows the Cross Validation across the taken range of alpha. The following are the outputs for optimal alpha:
For values of best alpha = 1: The train log loss is: 0.513842874233
For values of best alpha = 1: The test log loss is: 0.520035530431

Based on the optimal alpha, the model is generated and tested on the (70:30) split test data (3000 points). Fig.15. shows the performance visualisation of logistic regression model in terms of confusion matrix.

C. Support Vector Machine (SVM) Model

The SVM model is chosen by selecting ”hinge” loss as the parameter in SGD classifier. The model is l1 regularized and again it is fitted to compute probability estimates instead of absolute predictions, thus allowing us to analyse in terms of log loss error measure. Then it is cross validated across a range of alpha and visualised similarly as in logistic regression model.

Fig.16. shows the Cross Validation across the taken range of alpha. The following are the outputs for optimal alpha:
For values of best alpha = 0.0001: The train log loss is: 0.478054677285
For values of best alpha = 0.0001: The test log loss is: 0.489669093534

Based on the optimal alpha, the model is generated and tested on the (70:30) split test data (3000 points). Fig.17. shows the performance visualisation of SVM model in terms of confusion matrix.

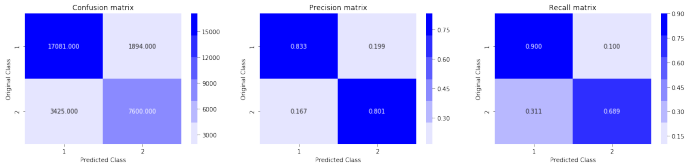


Fig. 18. XGboost Logistic Regression Model Performance

D. XGboost Logistic Regression Model

Based on the test log loss values, the logistic regression and SVM models are much better compared to the worst case of random model. To further enhance the performance, we implemented XGboost on logistic regression model and evaluated the log loss metric until improvement is seen.
The test log loss is: 0.357054433715; this is comparatively lower than the linear classifiers showing that there is scope of improvement to the model. Fig.18. shows the performance visualisation of xgboost logistic regression model in terms of confusion matrix.

X. Conclusion

For a given pair of questions, it was predicted if they are duplicates or not. The considered binary classification problem is modelled and visualized using logistic regression, SVM and XGboost techniques. The log-loss values approached to 0 as we moved across different models which reflected an improvement in prediction accuracy as also indicated by the confusion matrix representations.

Acknowledgment

This project and the research behind it would not have been possible without the support of our Professor, Mark Zolotas. His teaching and ideas have greatly influenced in successfully organising and completing our project.

References

[1] Data set Source:https://www.kaggle.com/c/quora-question-pairs
[2] SGD documentation:https://scikit-learn.org/stable/modules/generated/sklearn.linear_mo
[3] Evaluation Metrics: https://www.kaggle.com/competitions/quora-question-pairs/overview/evaluation
[4] Spacy word2vec:https://spacy.io/usage/linguistic-featuresvectors-similarity