Determining Quora Question Pairs similarity using Machine Learning

Sahithi Sallaram

April 27, 2022

Abstract

In this paper, Quora Question Pairs dataset is collected from Kaggle for determining if a question pair is duplicate or not. The model used for this purpose implements Stochastic Gradient Descent Classifiers with logistic regression and linear support vector machine methods using token features extracted from vectorization of words. The metric compared here is log loss for both logistic regression and linear SVM. Both classifiers had little difference in training log-loss but almost same values in test log-loss.

Keywords: Duplicate question pairs ; Quora question pairs ; SGDClassifier ; Duplicate detection.

1 Introduction

Quora is a question-and-answer website where users may ask questions and get answers from other users. Quora receives millions of questions not all of which are new and unique. The best answers get more votes, and these answers are a great learning resource for a variety of subjects. Duplicate questions are common on this site as quite a few of them have already been asked on Quora and have rich answers. If duplicates are allowed, it would corrupt the quality of answers thereby negatively effecting the experience of the person asking the questions, person answering the questions and the person searching the web for an answer (imagine searching Google for a question and finding 3 results from Quora instead of 1). This problem however is not unique to Quora, and many organizations have similar issues. Repeated questions, if handled individually, may prevent a user from seeing an existing high-quality response, and responders are unlikely to answer the same question repeatedly.

To resolve these concerns, duplicate questions are identified. It relieves responders from answering repeated queries and directs users to the most appropriate responses, thereby improving the overall user experience. In general, when a question is asked, Quora would use some methodology to find a subset of its existing question data base such that this subset contains questions which are “similar” to or about the same topic as the new question being asked. Once this subset has been identified, Quora would employ a machine learning technique to then determine if a duplicate question exists in this selected subset. If yes, it would notify the questioner and point them to it, else it creates the question.

This project aims to apply machine learning techniques to determine if any of the question pairs is a duplicate.

The problem of finding if two question pairs have the same meaning or not requires a method to capture the semantic information and words vector features of questions rather than just a group of words.

This work uses a model that implements a machine learning concept called SGDClassifier and analyze its performance on the dataset.

2 Related Work

In a similar research paper, Anishaa et al [3] reported the use of machine learning techniques to find similar questions. In this work the researchers have incorporated SQLite to preprocess the dataset and used machine learning algorithms like random forest , Logistic regression, SVM. They compared the performance of the model using error log loss function and response time.

From this paper, I found that the features extracted from the questions are limited and they did not use word vectorization which would have contributed to improving feature space.

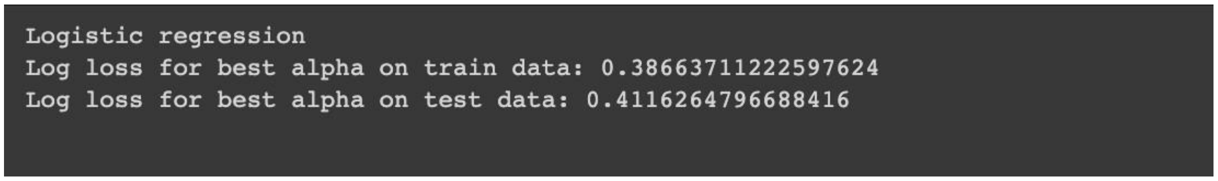
In my work I made use of python libraries for preprocessing , TF-IDF word vectors , other token features and used all the features with SGDClassifiers with hyperparameter tuning. As SGDClassifiers works well with large datasets.

3 Method

SGDClassifier implements regularized linear models with Stochastic Gradient Descent. The gradient of the loss is estimated for each sample at a time and the model is updated along the way with a decreasing strength schedule which is nothing but learning rate. Stochastic gradient descent is much faster than gradient descent when dealing with large data sets. Also, the training time in SGDClassifiers is much lesser when compared to the model implementing linear classifiers without SGDClassifier.

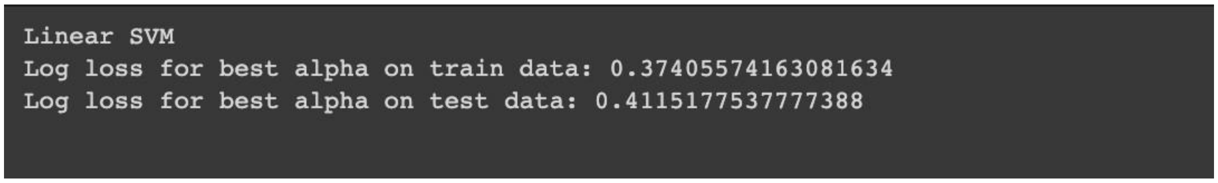
This model uses SGDClassifier to implement both the linear classifiers logistic regression and Support vector machine. The model behaviour can be controlled with the loss parameter that means to make the SGDClassifier perform as logistic regression loss parameter is set as ‘log’ and to make the SGDClassifier perform as SVM loss parameter is set as ‘hinge’. Each of these methods were also calibrated using the Sklearn Calibration Classifier.

Parameters which define the model architecture are referred to as hyperparameters and thus the process of searching for the ideal model is called as hyperparameter tuning. For logistic regression and linear SVM, the hyper parameters tuned were alpha and the penalty. The regularizer is a penalty added to the loss function that shrinks model parameters towards the zero-vector using either the squared euclidean norm L2 or the absolute norm L1 or a combination of both (Elastic Net). The penalty parameter is set to values ‘l1’,’l2’,and ‘elasticnet’ for each of the loss functions. The log loss is calculated for different alpha values for each of the penalties. The alpha at which the log loss was less was considered as best alpha. Then the SGDClassifier is setup with best alpha and type of penalty at which best alpha is observed. The final training model with all the extracted features(explained in data section) is then setup with this SGDClassifier which is used to predict the question pair duplicity from test data.



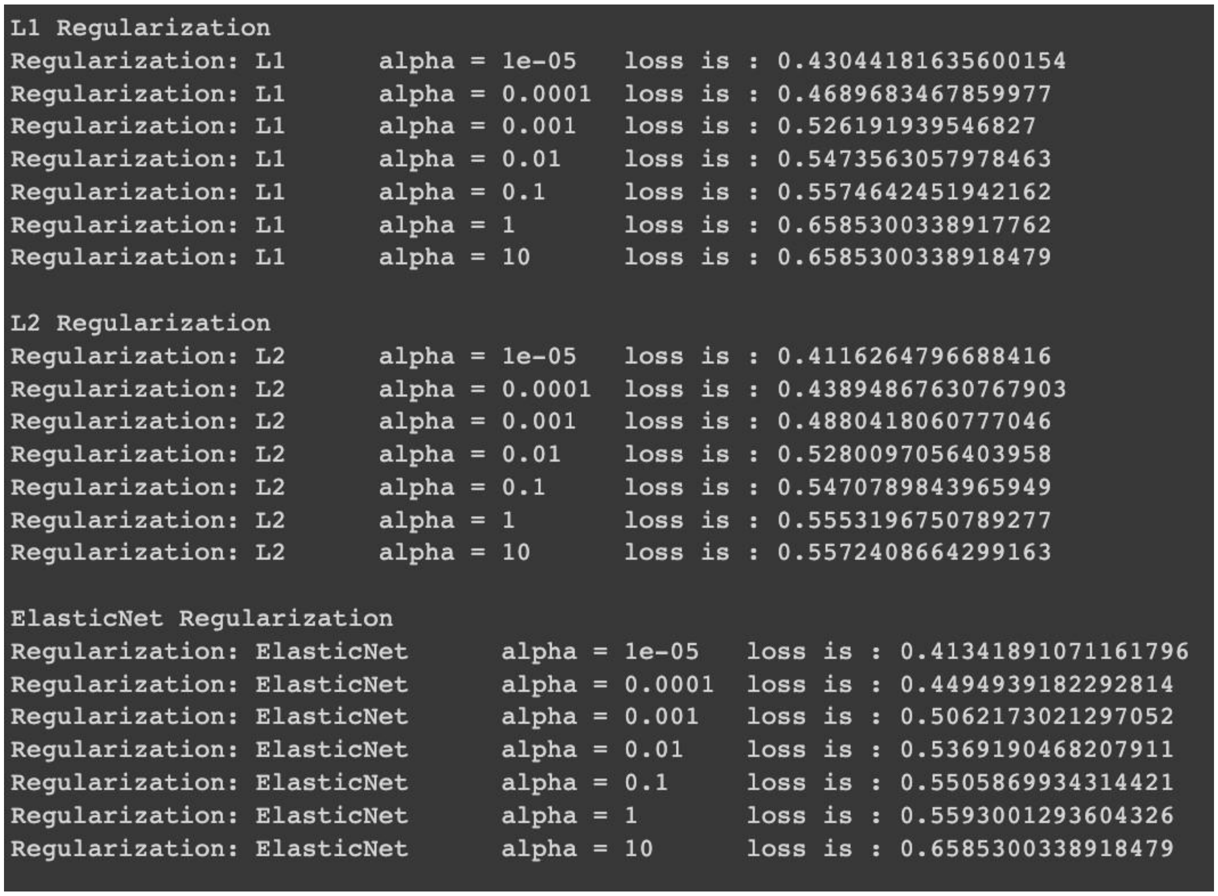
```
Logistic regression
Log loss for best alpha on train data: 0.38663711222597624
Log loss for best alpha on test data: 0.4116264796688416
```

Figure 1: Log-loss for logistic Regression



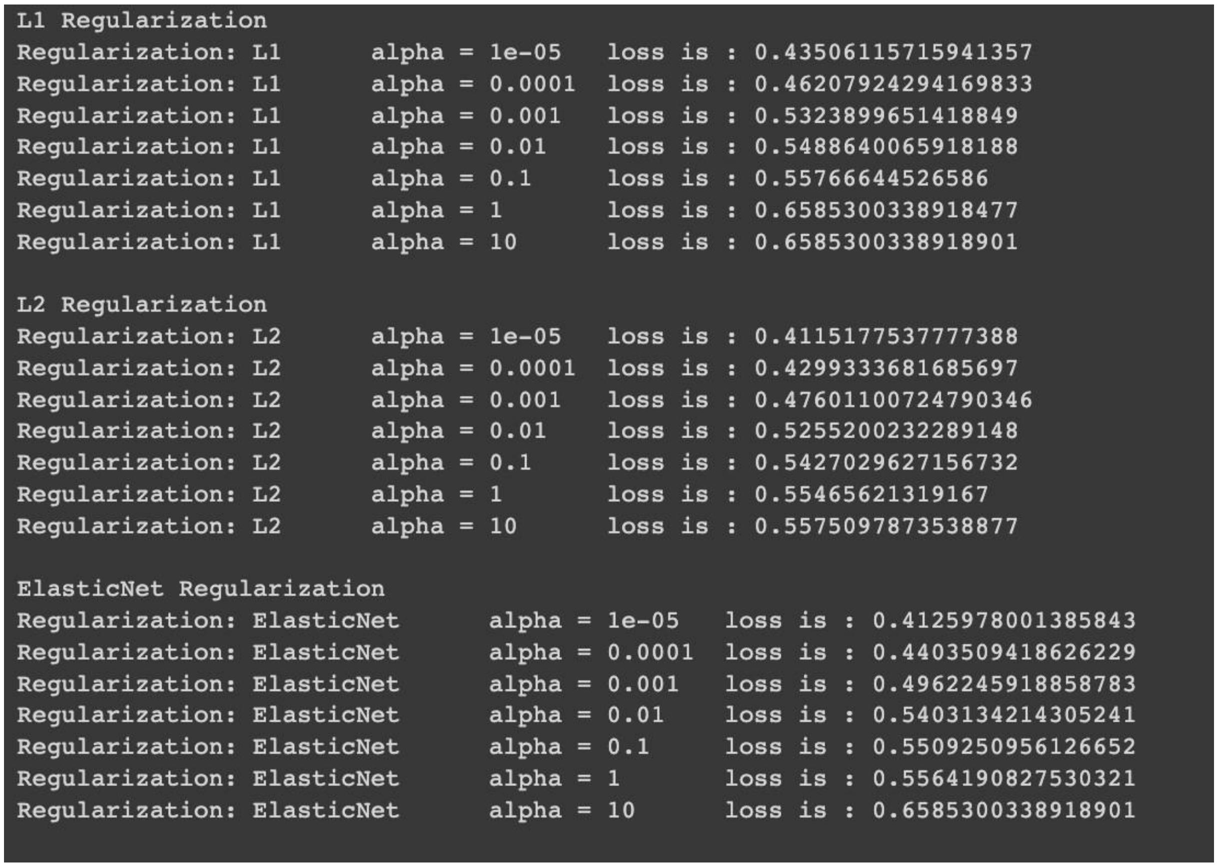
```
Linear SVM
Log loss for best alpha on train data: 0.37405574163081634
Log loss for best alpha on test data: 0.4115177537777388
```

Figure 2: Log-loss for SVM



```
L1 Regularization
Regularization: L1      alpha = 1e-05   loss is : 0.43044181635600154
Regularization: L1      alpha = 0.0001  loss is : 0.4689683467859977
Regularization: L1      alpha = 0.001   loss is : 0.526191939546827
Regularization: L1      alpha = 0.01    loss is : 0.5473563057978463
Regularization: L1      alpha = 0.1     loss is : 0.5574642451942162
Regularization: L1      alpha = 1       loss is : 0.6585300338917762
Regularization: L1      alpha = 10      loss is : 0.6585300338918479

L2 Regularization
Regularization: L2      alpha = 1e-05   loss is : 0.4116264796688416
Regularization: L2      alpha = 0.0001  loss is : 0.43894867630767903
Regularization: L2      alpha = 0.001   loss is : 0.4880418060777046
Regularization: L2      alpha = 0.01    loss is : 0.5280097056403958
Regularization: L2      alpha = 0.1     loss is : 0.5470789843965949
Regularization: L2      alpha = 1       loss is : 0.5553196750789277
Regularization: L2      alpha = 10      loss is : 0.5572408664299163

ElasticNet Regularization
Regularization: ElasticNet      alpha = 1e-05   loss is : 0.41341891071161796
Regularization: ElasticNet      alpha = 0.0001  loss is : 0.4494939182292814
Regularization: ElasticNet      alpha = 0.001   loss is : 0.5062173021297052
Regularization: ElasticNet      alpha = 0.01    loss is : 0.5369190468207911
Regularization: ElasticNet      alpha = 0.1     loss is : 0.5505869934314421
Regularization: ElasticNet      alpha = 1       loss is : 0.5593001293604326
Regularization: ElasticNet      alpha = 10      loss is : 0.6585300338918479
```

Figure 3: Penalty Regularization with different alphas in Logistic regression



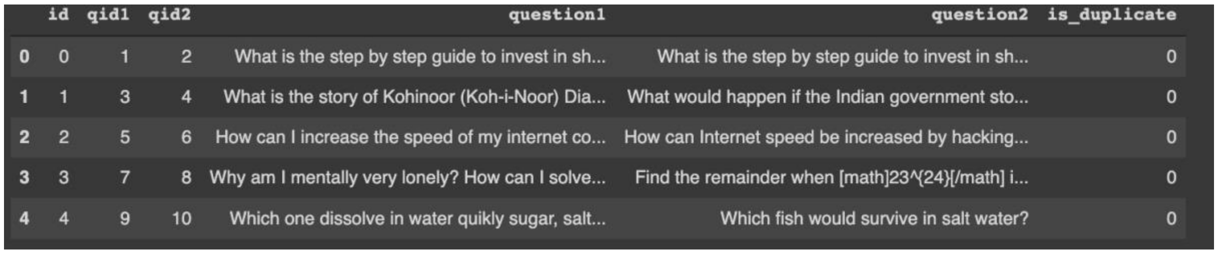
```
L1 Regularization
Regularization: L1      alpha = 1e-05   loss is : 0.43506115715941357
Regularization: L1      alpha = 0.0001  loss is : 0.46207924294169833
Regularization: L1      alpha = 0.001   loss is : 0.5323899651418849
Regularization: L1      alpha = 0.01    loss is : 0.5488640065918188
Regularization: L1      alpha = 0.1     loss is : 0.55766644526586
Regularization: L1      alpha = 1       loss is : 0.6585300338918477
Regularization: L1      alpha = 10      loss is : 0.6585300338918901

L2 Regularization
Regularization: L2      alpha = 1e-05   loss is : 0.4115177537777388
Regularization: L2      alpha = 0.0001  loss is : 0.4299333681685697
Regularization: L2      alpha = 0.001   loss is : 0.47601100724790346
Regularization: L2      alpha = 0.01    loss is : 0.5255200232289148
Regularization: L2      alpha = 0.1     loss is : 0.5427029627156732
Regularization: L2      alpha = 1       loss is : 0.55465621319167
Regularization: L2      alpha = 10      loss is : 0.5575097873538877

ElasticNet Regularization
Regularization: ElasticNet      alpha = 1e-05   loss is : 0.4125978001385843
Regularization: ElasticNet      alpha = 0.0001  loss is : 0.4403509418626229
Regularization: ElasticNet      alpha = 0.001   loss is : 0.4962245918858783
Regularization: ElasticNet      alpha = 0.01    loss is : 0.5403134214305241
Regularization: ElasticNet      alpha = 0.1     loss is : 0.5509250956126652
Regularization: ElasticNet      alpha = 1       loss is : 0.5564190827530321
Regularization: ElasticNet      alpha = 10      loss is : 0.6585300338918901
```

Figure 4: Penalty Regularization with different alphas in SVM

4 Data and Evaluation

The Dataset I have used is Quora question pairs taken from Kaggle. The data set consists of train and test .csv files. The train set consists of 404,290 question pairs and 2,345,795 question pairs in test file. The train file has 6 columns which are id - the unique id of a training set question pair, qid1, qid2 are unique ids of each question which are only available in train.csv, question1, question2 - the full text of each question, is duplicate - the target variable, set to 1 if question1 and question2 have essentially the same meaning, and 0 otherwise.

The preview of train data is as below



	id	qid1	qid2	question1	question2	is_duplicate
0	0	1	2	What is the step by step guide to invest in sh...	What is the step by step guide to invest in sh...	0
1	1	3	4	What is the story of Kohinoor (Koh-i-Noor) Dia...	What would happen if the Indian government sto...	0
2	2	5	6	How can I increase the speed of my internet co...	How can Internet speed be increased by hacking...	0
3	3	7	8	Why am I mentally very lonely? How can I solve...	Find the remainder when [math]23^{24}[/math] i...	0
4	4	9	10	Which one dissolve in water quikly sugar, salt...	Which fish would survive in salt water?	0

Figure 5: Train data preview

As the test data file has millions of questions, I have used only train dataset and divided the data into train and test sets in the split ratio of 80:20 . The train dataset has 63% non-duplicate pairs and 27% duplicate pairs based on is duplicate column. To maintain this proportion, I have used stratification with is duplicate column while train test split. Random state parameter has been used for reproducing the same sets of train and test data.

As part of Data cleaning and preprocessing the textual data from question1 and question2 columns has been subjected to the following activities - Text conversion to lowercase, few special characters and numbers are replaced with their string equivalents, expanded the contraction words with the help of contractions library (example : “I wasn’t aware of it “ is converted to “I was not aware of it”), removal of special characters, removal of HTML tags using beautiful soup python library, removal of stop words using NLTK python library. Featurization is important to get more information from the input data and can be extracted by analyzing the words or tokens of the questions. some of the extracted features are as follows q1 len, q2 len, q1 words, q2 words, words total, words common, words shared, num common adj , num common prn, num common n(number of common adjectives, proper nouns, non-proper nouns in question1 and question2 I have used PoS features from NLTK) . Also, extracted few like more token sort ratio (strings are sorted alphabetically and joined together distance similarity ratio is calculated between the strings), token set ratio by using methods from fuzzywuzzy which is a very helpful python library for string matching. Also made use of some features involving length of words from questions. The number of features after this process is 25.

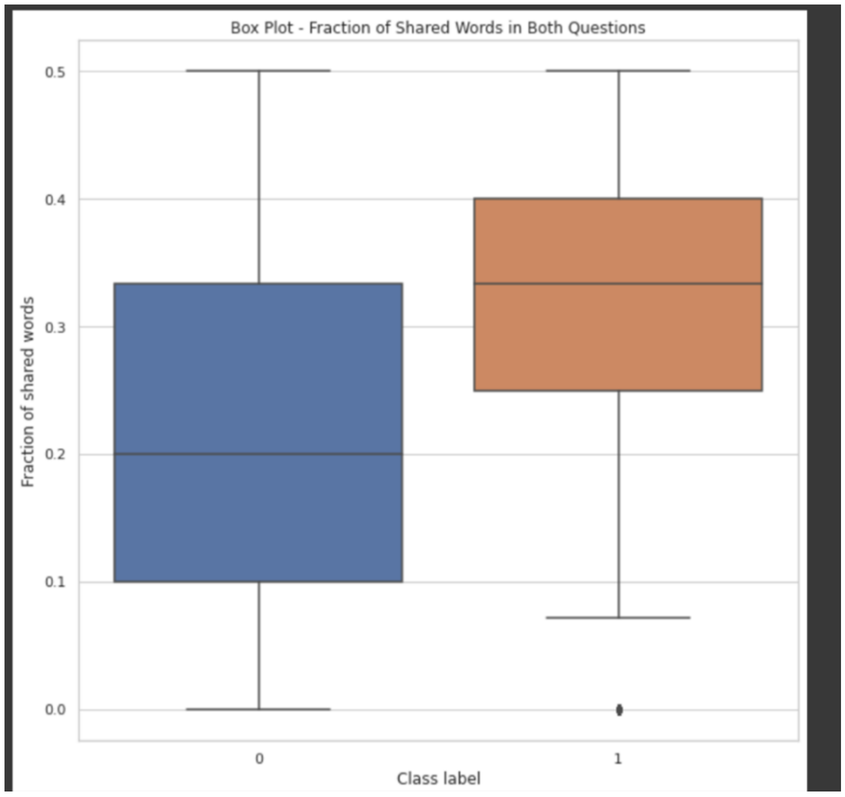


Figure 6: Box plot showing words shared feature

Word vectorization is one of the aspects I found very intriguing as it increases the number of meaningful features .It is a great technique in natural language processing to map words to a corresponding vector of real numbers. These are useful in capturing the semantic meaning of

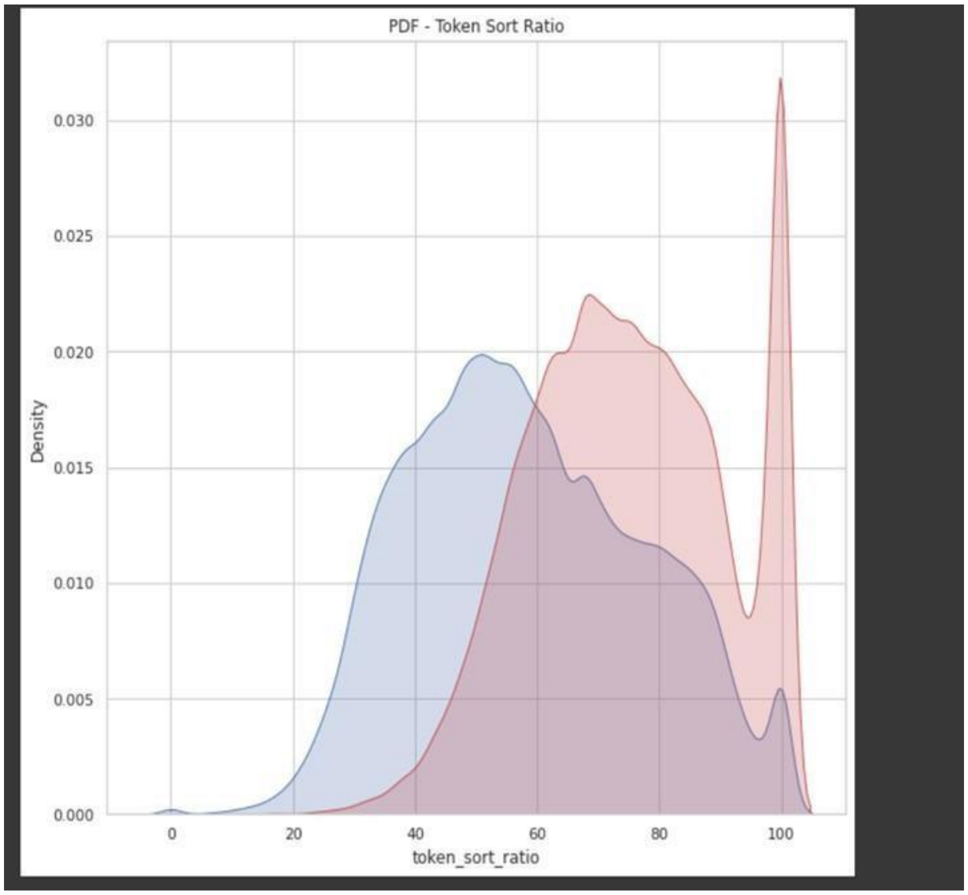


Figure 7: PDF plot showing token sort ratio feature

a sentence. The type of vectorization used here is TF-IDF. Term Frequency Inverse Document Frequency is the is the ratio of the number of times a word (term) occurs in a particular question, to the number of times the word occurs in all the questions (our entire corpus). A higher TF-IDF value indicates a more important word.TF-IDF word vectors of the tokens can be generated by using TfidfVectorizer package from sklearn.feature extraction.text python library. Each question has about 77969 dimensions after TF-IDF vectorization.All these extracted features are used in the model.

5 Results and Insights

This model achieved log loss of 0.386 with best alpha on train data and log loss of 0.4116 with best alpha on test data with logistic regression . The Log loss for best alpha on train data is 0.374 and log loss for best alpha on test data is 0.4115 with SVM. Other performance metrics like precision and recall were calculated and confusion matrices were plotted.

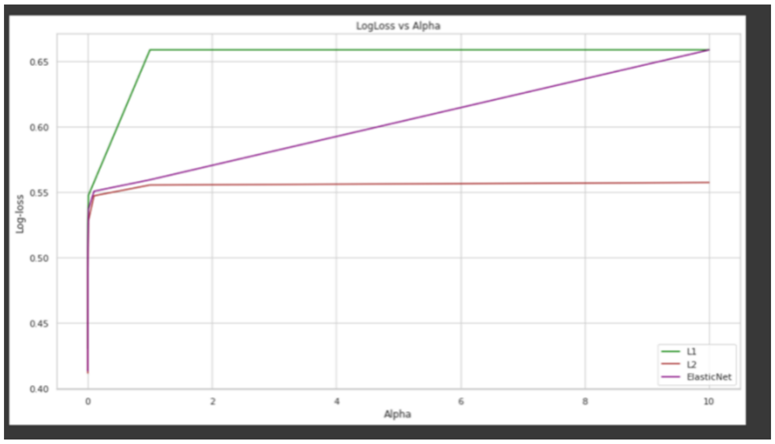


Figure 8: Log loss vs Alpha in Logistic Regression
Figure 8 shows the plot between Log-loss vs Alpha for three penalties L1, L2 and Elasticnet in Logistic Regression.

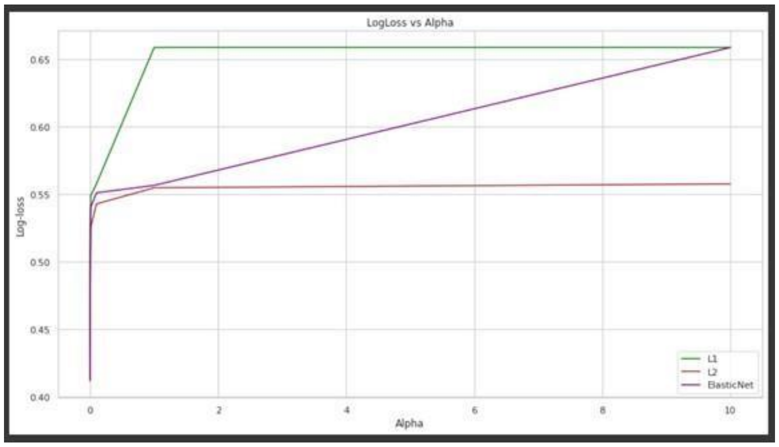


Figure 9: Log loss vs Alpha in SVM
Figure 9 shows the plot between Log-loss vs Alpha for three penalties L1, L2 and Elasticnet in Linear SVM.

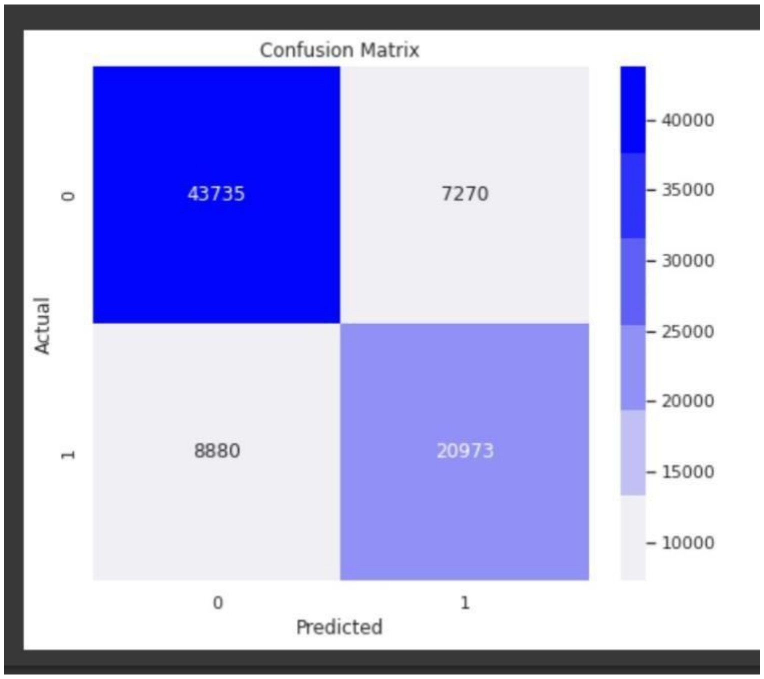


Figure 10: Confusion Matrix for Logistics regression

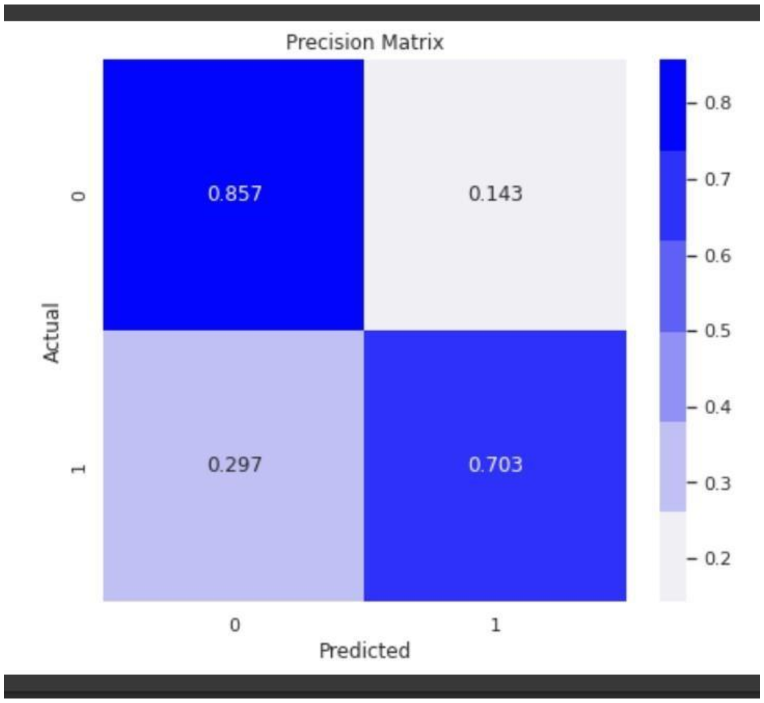


Figure 11: Precision Matrix for Logistics regression

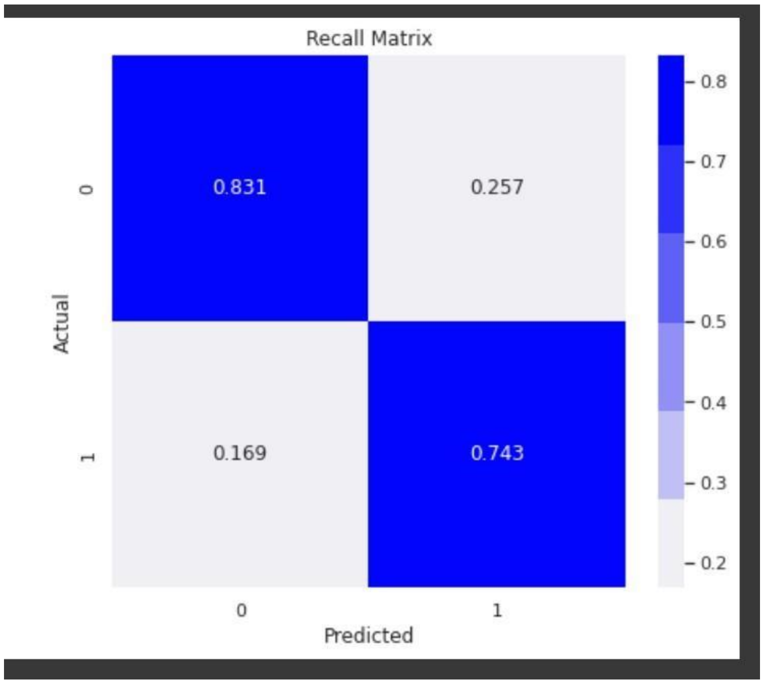


Figure 12: Recall Matrix for Logistics regression

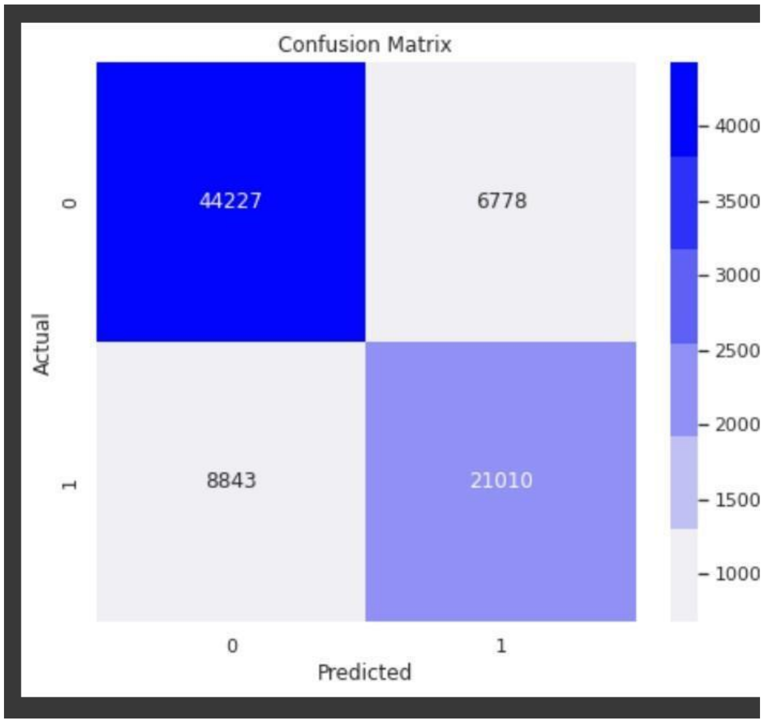


Figure 13: Confusion Matrix for SVM

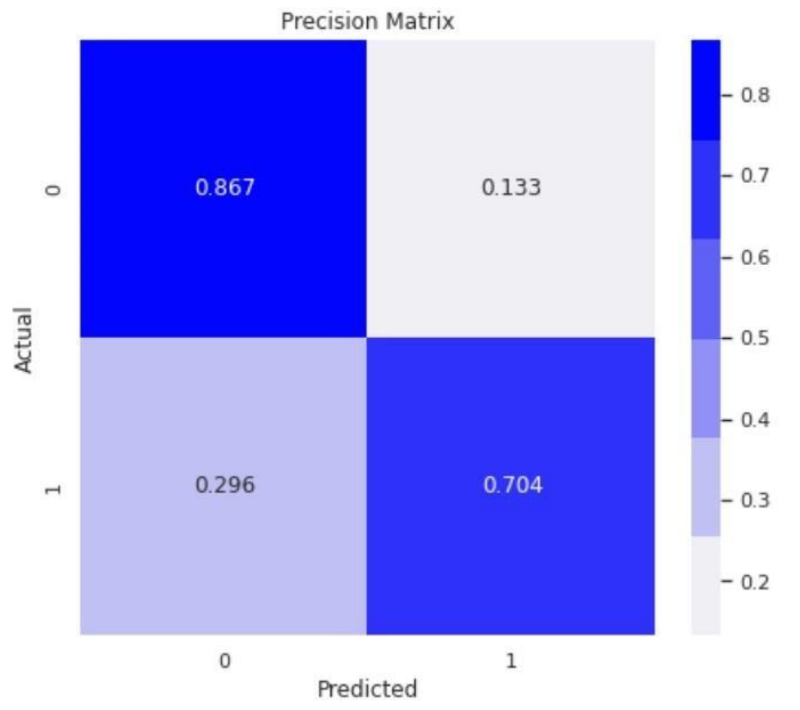


Figure 14: Precision Matrix for SVM

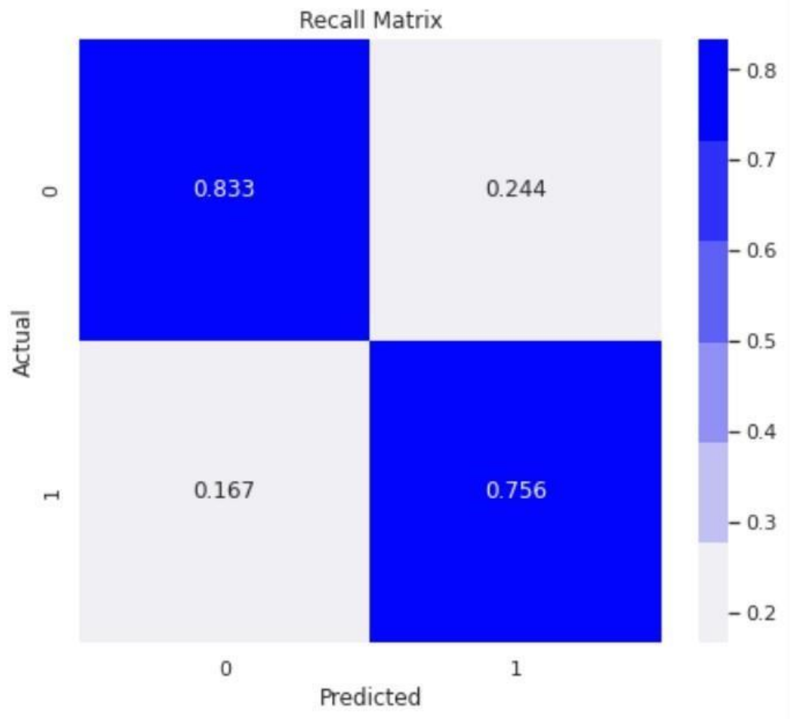


Figure 15: Recall Matrix for SVM

6 Conclusion and Future work

Logistic Regression and SVM with SGDClassifier hyperparameter tuning have been used to solve the duplicate question problem posed by the Quora dataset. The log loss on train data is 0.386 and Log loss on test data is 0.4116 with Logistic Regression. With Linear SVM log loss on train data is 0.374 and Log loss on test data is 0.4115. Linear SVM performed little better with train data than logistic regression. But the log loss values on test data are same for both. Hence it can be concluded that both methods performed almost similarly . As Future work , this problem can be handled by using more than one vectorization techniques in order to increase dimensions of the data. Also , machine learning concepts like ensemble methods can be used to club different algorithms together to boost the performance of overall model.

References

[1] Samir AbdelRahman and Catherine Blake. Sbdlrhmn: A rule-based human interpretation system for semantic textual similarity task. In ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 536–542, 2012.

[2] Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. Semeval-2012 task 6: A pilot on semantic textual similarity. In ** SEM 2012: The First Joint Conference on Lexical and Computational Semantics–Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393, 2012.

[3] VKR Anishaa, P Sathvika, and S Rawat. Identifying similar question pairs using machine learning techniques. *Indian Journal of Science and Technology*, 14(20):1635–1641, 2021.

[4] Sumit Bhatia, Prakhar Biyani, and Prasenjit Mitra. Summarizing online forum discussions– can dialog acts of individual messages help? In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 2127–2131, 2014.

[5] Thanh Ngoc Dao and Troy Simpson. Measuring similarity between sentences. *The Code Project*, 2005.

[6] Jonathan L Elsas and Jaime G Carbonell. It pays to be picky: an evaluation of thread retrieval in online forums. In *Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval*, pages 714–715, 2009.

[7] Zainab Imtiaz, Muhammad Umer, Muhammad Ahmad, Saleem Ullah, Gyu Sang Choi, and Arif Mehmood. Duplicate questions pair detection using siamese malstm. *IEEE Access*, 8:21932–21942, 2020.

[8] Jonas Mueller and Aditya Thyagarajan. Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

[9] Badri N Patro, Vinod K Kurmi, Sandeep Kumar, and Vinay P Namboodiri. Learning semantic sentence embeddings using sequential pair-wise discriminator. *arXiv preprint arXiv:1806.00807*, 2018.

[10] Razvan Rughinis¸, Alina Petra Marinescu-Nenciu, S¸tefania Matei, and Cosima Rughis¸.˘ Computersupported collaborative questioning. regimes of online sociality on quora. In *2014 9th Iberian Conference on Information Systems and Technologies (CISTI)*, pages 1–6. IEEE, 2014.