Quora Question Pair Similarity Problem

Mahak Sharma
Mtech (CSE)
IIITD
mahak21047@iiitd

Palani Vigneshwar
Mtech (CSE)
IIITD
palani21062@iiitd

Giridhar S
Mtech (CSE)
IIITD
giridhar21026@iiitd

Abstract

This paper focuses on Natural Language Processing by detecting duplicated Quora questions based on Quora dataset. We examined the dataset and used machine learning models like decision tree, logistic regression, Random Forest, Linear SVM, RBF-SVM ,Multi-Layer Perceptron and XGBoost. We finally found that XGBoost has the best performance.

1 Introduction

Quora is a platform to ask questions which receives millions of questions which may not be unique. A few questions may have already been answered. If duplicates are allowed, quality of the answers would be corrupted thereby affecting the experience of the user asking the question. Hence the problem statement is to find whether two question are duplicate or not by using machine learning models and natural language processing.

2 Dataset and its Analysis/Preprocessing

Data is used from the Kaggle competition "Can you identify question pairs that have the same intent." Data Set is available in two parts training set and test set. We have predicted labels available on the training set, but test data doesn't have any predicted labels. Training Data Set consists of the following columns:

1. Id – It represented question pair set in the training set.

2. qid1, qid2 – Representation of unique ids of each question(Available only in the training set).

3. question1, question2 – Represents full text of each question.

4. isduplicate – It is the target variable. It has a value of 1 when question1 and question2 have the same meaning Otherwise, 0.

Test Data Set consists of the following columns:

Testid: Represents unique id for question pair

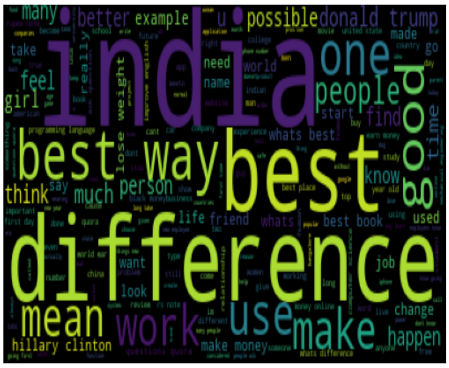
Training DataSet:

1. Data Set Size: 404290 rows * 6 columns 2. No. Of instances with 0 label: 255027 3. No. of instances with 1 label: 149263 4. Avg Length per question: 85.46078

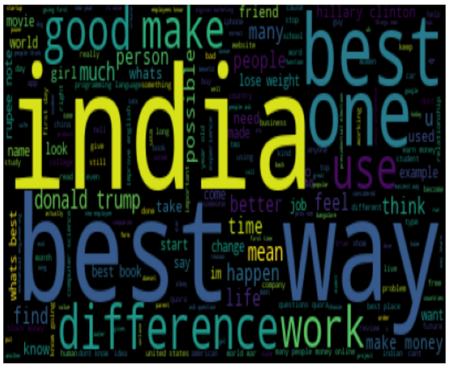
Test DataSet:

1. Data Set Size: 2345796 rows * 3 columns 2. Avg Length per question: 60.07010

The necessary preprocessing such as removing punctuation's,removing stopword's,removing tags and numbers,lower-casing the letters and converting the words to vectors using Word2vec and glove was done. One dataset used Word2vec to convert to vector and another dataset used glove to convert words to vector.



The wordcloud for q1 feature.



The wordcloud for q2 feature

3 Literature Review

1. Research paper [8] aimed at comparing machine learning models with hyper parameters (like SVM, Logistic Regression) with neural networks based models like LSTM, Continous Bag of Words.

2. Research paper [2] aimed at comparing Rule based method(Jaccad method), machine learning models (SVM) and neural network methods.

3. Paper [9] aimed at vectorization of text data and use of Siamese Deep Learning Network

4. Research paper [3] aims at comparing various ML models like KNN, Decision Tree, Random Forest, Extra Trees, Ada Boost, Xgboost

5. This research [1] aims at preprocessing, vectorization and comparing various models like Random forest, Decision Trees, SVM, Logistic Regression. It also focuses on log loss as a parameter for consideration.

6.In paper [4] LSTM and biLSTM is used to find the semantic similarity between questions.

4 Baselines

Two baselines are used in the project –

1. Decision tree :Levenshtein distance calculated it is a string metric for measuring the difference between two sequences.[which was changed to cosine similarity for the upcoming models built on baselines]

2. Logistic Regression:Jaccard Similarity is calculated corresponding to every instance consisting of question pairs, and on this feature, logistic regression has been applied.

Model	**Train accuracy**	**Test accuracy**
Decision Tree	0.9971	0.74
Logistic Regression	0.65	0.65

Model	**Train Loss**	**Test Loss**
Decision Tree	0.0047	8.8197
Logistic Regression	0.585	0.6011

Model	**Train Precision**	**Train Recall**	**Train Fscore**
Decision Tree	1.0	1.0	1.0
Logistic Regression	0.61	0.59	0.59

Model	**Test Precision**	**Test Recall**	**Test Fscore**
Decision Tree	0.72	0.73	0.72
Logistic Regression	0.61	0.59	0.59

5 Final Models

The final models used in the project are –

1. Decision tree with cost complexity parameter ccpalpha = 0.000029

2. XGBoost classifier with depth = 10 and no. of estimators = 80

3. Logistic Regression with C=0.3 and max iterations = 600

4. Random Forest classifier with depth=40 and no. of estimators = 65

5. Linear SVM using SGD

6. SVM using rbf kernel by using RBF sampler

7. Multi layer perceptron with hidden layer sizes=300 and max iterations = 250

6 Results

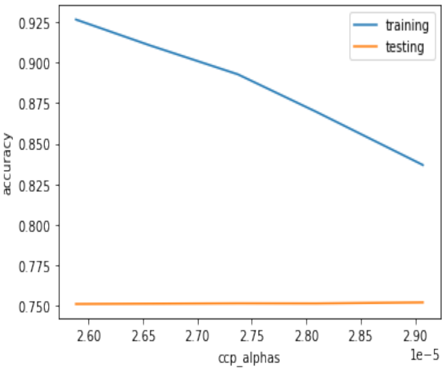
Here our some of our observations:

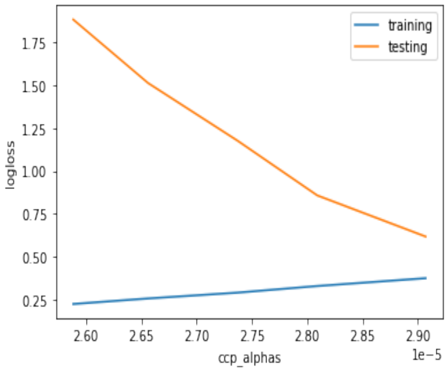
6.1 Decision tree

Decision Tree Alpha vs Accuracy and Alpha vs Log loss.

Alpha	**Train accuracy**	**Test accuracy**
0.000026	0.926605	0.751177
0.000027	0.910718	0.751367
0.000027	0.892810	0.751581
0.000028	0.869687	0.751523
0.000029	0.836860	0.752166

Alpha	Train Loss	Test Loss
0.000026	0.2238	1.8836
0.000027	0.2561	1.5129
0.000027	0.2901	1.1761
0.000028	0.3287	0.8572
0.000029	0.3743	0.6172





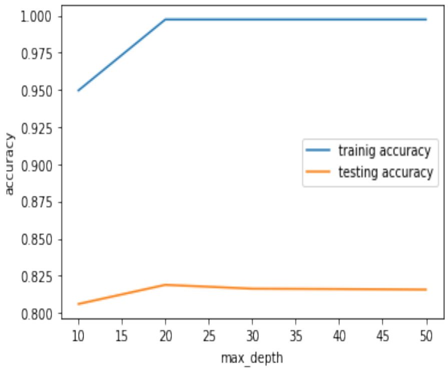
It can be seen from the plots and the table that ccpalpha = 0.000029 provides best peformance.

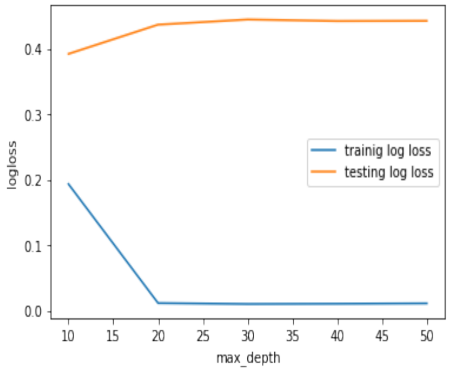
6.2 XGBoost

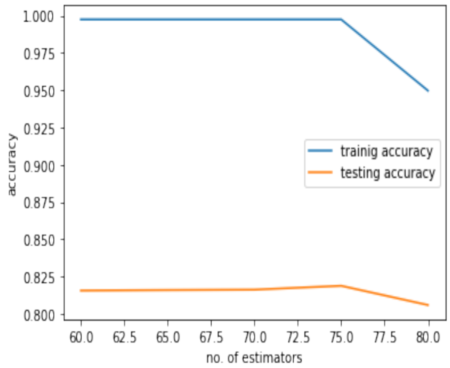
depth, no of estimators vs Accuracy, Log loss.

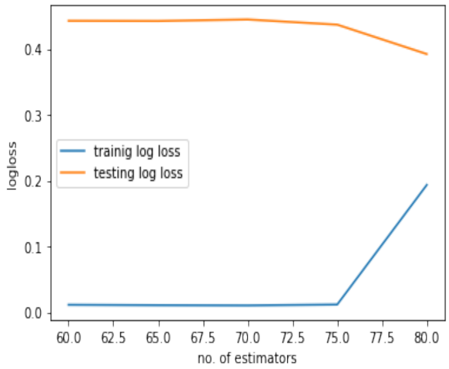
Depth	Estimator	Train accuracy	Test accuracy
10	80	0.9496	0.8059
20	75	0.9974	0.8188
30	70	0.9974	0.8162
40	65	0.9974	0.8159
50	60	0.9974	0.8156

Depth	Estimator	Train Loss	Test Loss
10	80	0.1934	0.3922
20	75	0.0116	0.4368
30	70	0.0102	0.4447
40	65	0.0105	0.4424
50	60	0.0112	0.4426









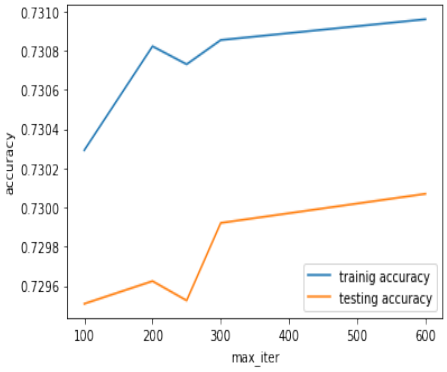
It can be seen from the plots and the table that depth = 10 and no.of estimators = 80 provides the best performance.

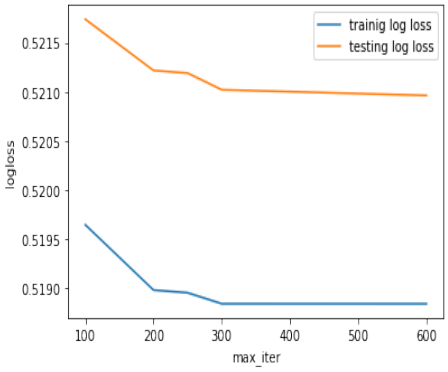
6.3 Logistic Regression

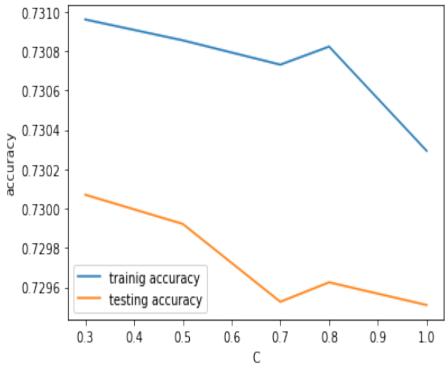
Max iterations,C vs Accuracy,Log loss.

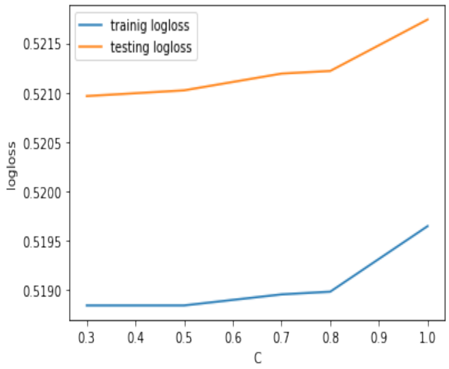
maxiter	C	Train accuracy	Test accuracy
100	1.0	0.7302	0.7295
200	0.8	0.7308	0.7296
250	0.7	0.7307	0.7295
300	0.5	0.7308	0.7299
600	0.3	0.7309	0.7300

maxiter	C	Train Loss	Test Loss
100	1.0	0.5196	0.5217
200	0.8	0.5189	0.5212
250	0.7	0.5189	0.5211
300	0.5	0.5188	0.5210
600	0.3	0.5188	0.5209









It can be seen from the plots and the table that max iterations = 600 and C = 0.3 provides the best performance.

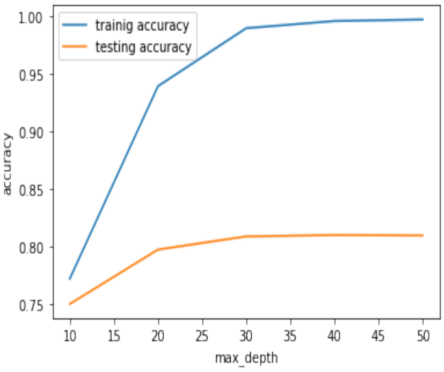
6.4 Random Forest

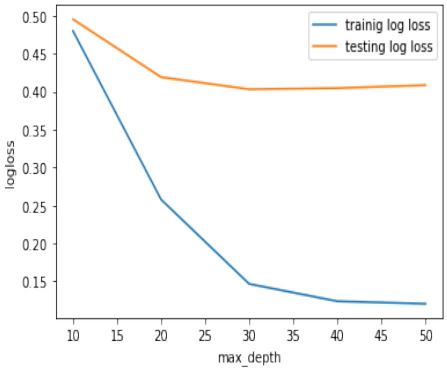
depth, no of estimators vs Accuracy.

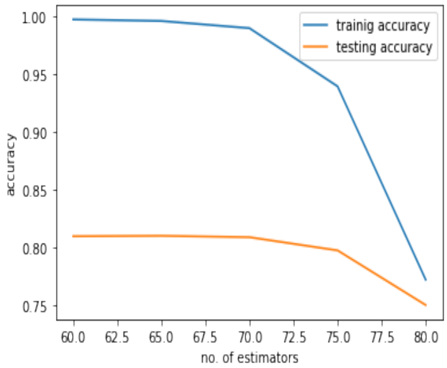
Depth	Estimator	Train accuracy	Test accuracy
10	80	0.7722	0.7503
20	75	0.9391	0.7975
30	70	0.9893	0.8089
40	65	0.9955	0.8101
50	60	0.9969	0.8097

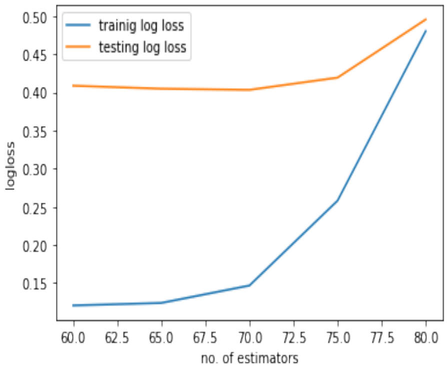
depth, no of estimators vs Log loss.

Depth	Estimator	Train Loss	Test Loss
10	80	0.4803	0.4956
20	75	0.2579	0.4193
30	70	0.1466	0.4033
40	65	0.1237	0.4047
50	60	0.1204	0.4087



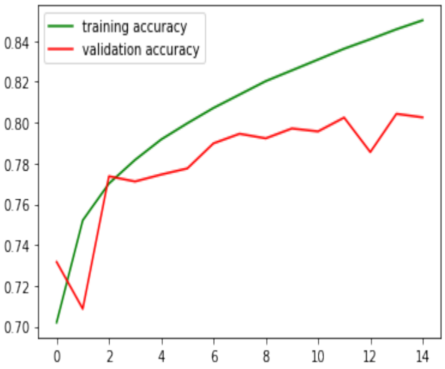


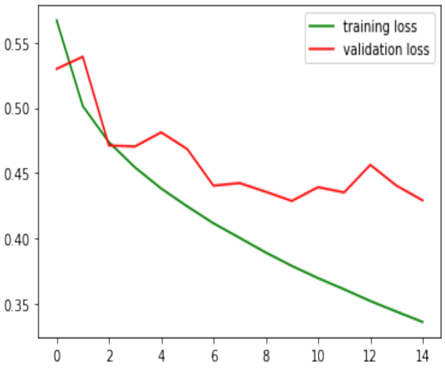




It can be seen from the plots and the table that depth = 40 and no. of estimators = 65 provides the best performance.

6.5 LSTM





Epoch vs Accuracy

Epoch	Train accuracy	Test accuracy
6	0.7997	0.7775
7	0.8072	0.7898
8	0.8138	0.7945
9	0.8203	0.7923
10	0.8256	0.7971

Epoch vs Loss

Epoch	Train Loss	Test loss
6	0.4245	0.4684
7	0.4116	0.4404
8	0.4005	0.4424
9	0.3893	0.4357
10	0.3790	0.4287

Epoch vs Train Precision,Train Recall

Epoch	Train Precision	Train Recall
6	0.7415	0.7029
7	0.7497	0.7177
8	0.7558	0.7326
9	0.7617	0.7477
10	0.7675	0.7575

Epoch vs Test Precision,Test Recall

Epoch	Test Precision	Test Recall
6	0.7605	0.5775
7	0.6994	0.7525
8	0.76748	0.6341
9	0.6937	0.7802
10	0.7114	0.7554

From the above plots and the tables, it can be seen that LSTM gives best performance when epoch = 10. The Kaggle loss achieved through this model is 0.41

6.6 Models

Training and test accuracy of all models

Model	Train accuracy	Test accuracy
Decision Tree	0.8043	0.7519
XGBoost	0.9496	0.8059
Logistic Regression	0.7309	0.7300
Random Forest	0.9968	0.8088
Linear SVM	0.7359	0.7347
RBF SVM	0.7622	0.7592
Multi Layer Perceptron	0.7598	0.7587
LSTM	0.8209	0.7946

Training and test log loss of all models

Model	Train Loss	Test Loss
Decision Tree	0.4133	0.5193
XGBoost	0.1934	0.3922
Logistic Regression	0.5188	0.5209
Random Forest	0.1210	0.4096
Linear SVM	0.5212	0.5232
RBF SVM	0.4756	0.4798
Multi Layer Perceptron	0.4712	0.4743
LSTM	0.3870	0.4347

Training Precision, Recall and Fscore of all models

Model	Train Precision	Train Recall	Train Fscore
Decision Tree	0.79	0.79	0.79
XGBoost	1.00	1.00	1.00
Logistic Regression	0.71	0.69	0.70
Random Forest	1.0	1.0	1.0
Linear SVM	0.72	0.69	0.70
RBF SVM	0.75	0.73	0.74
Multi Layer Perceptron	0.75	0.72	0.73
LSTM	0.76	0.74	0.75

Testing Precision, Recall and Fscore of all models

Model	Test Precision	Test Recall	Test Fscore
Decision Tree	0.73	0.73	0.73
XGBoost	0.8	0.78	0.79
Logistic Regression	0.71	0.69	0.70
Random Forest	0.8	0.78	0.79
Linear SVM	0.72	0.69	0.70
RBF SVM	0.74	0.73	0.73
Multi Layer Perceptron	0.75	0.72	0.73
LSTM	0.727	0.706	0.716

6.7 Comparison

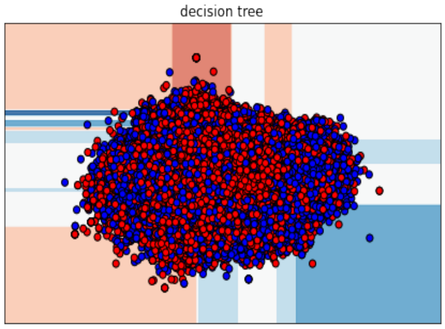
The baseline models used were decision tree and logistic regression. The baseline decision tree model had an training accuracy of 0.99 and testing accuracy of 0.74. This clearly showed that it was overfittig. To improve the test accuracy a series of analysis were performed and results are shown above.

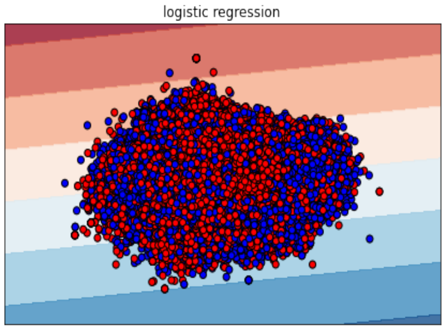
Decision tree after cost complexity pruning gave testing accuracy of 0.75 which was not much of an improvement but the overfitting was reduced. Then all the above models were run as shown above and it was found that XGBoost gave the best testing accuracy and log loss combined among all of the classifiers - 0.8059 and 0.3922

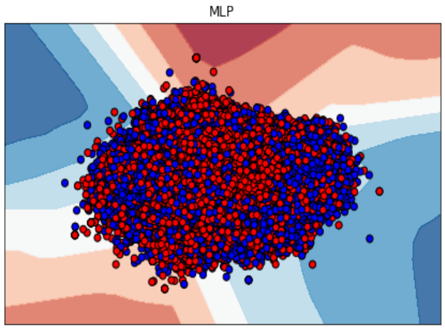
Random forest too gave a good testing accuracy of 0.8088 but had a high log loss than Xgboost. LSTM gives a accuracy of 0.79 which is better than baseline models but still falls short to XGBoost. Multiple layer perceptron gave almost the same accuracy for different parameters and gave an log loss of 0.47 which was again better than the baseline models.

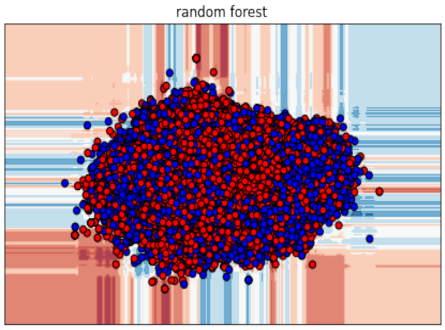
XGBoost had the fscore of 0.79 which was better than the baseline models where decision tree and logistic regression had 0.72 and 0.59 respectively. SVM using SGD is done which gives better performance basline logistic regression. RBFSampler from sklearn is used to map the data to higher dimensions and that data is used in SVM using the SGD. This gives a better performance than linear svm and also than the baseline models. Random Forest also gives a good accuracy comparable to XGBoost. But it has a little high log loss than XGBoost.

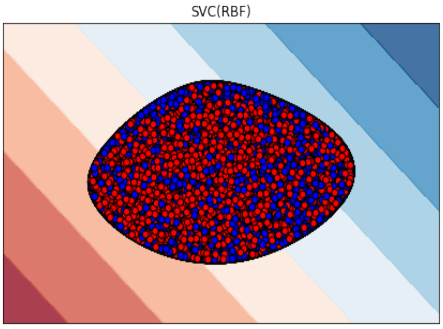
6.8 Decision boundaries

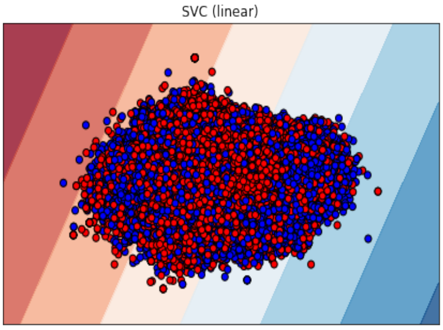


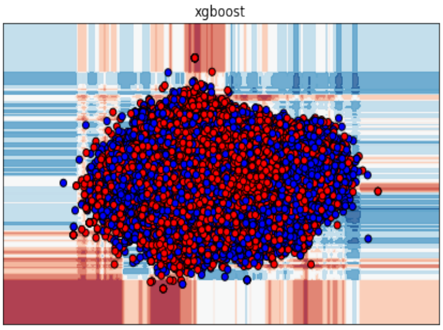












7 Conclusion

We tested 8 different models(Random Forest, XGBoost, Logistic Regression, Decision Tree, Multi Layer Perceptron, Linear, RBF SVM and LSTM). From our observations we found out that XGBoost is the best model in terms of log loss efficiency, precision, recall and fscore. It prrovided an test accuracy of 0.805, test log loss of 0.39 and fscore of 0.79.

8 Contribution of each member

Palani Vigneshwar

Implemented part of code and also the blog.

Mahak Sharma

Implemented part of code and also the presentation.

Giridhar S

Implemented part of code and also the report.

9 References

[1] V. Bhalerao, S. Ar, and S. Panda. A machine learning model to identify duplicate questions in social media forums. International Journal of Innovative Technology and Exploring Engineering, 9:370–373, 04 2021.https://www.ijitee.org/wp-content/uploads/papers/v9i4/D1362029420.pdf

[2] C. Saedi, J. Rodrigues, J. Silva, A. Branco, and V. Maraev. Learning profiles in duplicate question detection. In 2017 IEEE international conference on information reuse and integration (IRI), pages 544– 550. IEEE, 2017.

[3] A. S. Shashank Pathak and S. Shukla. Semantic string similarityfor quora question pairs, 2018.http://www.iraj.in/journal/journal$_{file}/journal_{pdf}/6-489-154893342177-80.pdf$

[4]Yllias Chali and Rafat Islam, Question-Question Similarity in Online Forums, FIRE'18: Proceedings of the 10th annual meeting of the Forum for Information Retrieval EvaluationDecember 2018 Pages 21–28https://doi.org/10.1145/3293339.3293345

[5] https://scikit-learn.org/stable/auto_examples/classification/plot_classifier_comparison.html#sphx-glr-auto-examples-classification-plot-classifier

[6] https://scikit-learn.org/stable/auto_examples/svm/plot_separating_hyperplane.html

[7] https://en.wikipedia.org/wiki/Jaccard_index

[8] L. Sharma, L. Graesser, N. Nangia, and U. Evci. Natural language understanding with the quora question pairs dataset, 2019. https://arxiv.org/ftp/arxiv/papers/1907/1907.01041.pdf

[9] M. Kashif and C. Arora. Question answering system based on sentence similarity. PhD thesis, 2017.

[10] https://drive.google.com/file/d/1AfUqjl5of1Dx8rYIqUtTPaxZLfkkxbXZ/view?usp=sharing - The code and data in the google drive