MACHINE LEARNING ENGINEER NANODEGREE RITU YADAV
CAPSTONE PROJECT REPORT DATE: 23-06-2018

QUORA QUESTION PAIRS: IDENTIFYING QUESTIONS WITH SAME INTENT

Contents

1. Introduction

1.1 Domain Background

Quora is a place to gain and share knowledge—about anything. It's a platform to ask questions and connect with people who contribute unique insights and quality answers. This empowers people to learn from each other and to better understand the world.

Over 100 million people visit Quora every month, so it's no surprise that many people ask similarly worded questions. Multiple questions with the same intent can cause seekers to spend more time finding the best answer to their question, and make writers feel they need to answer multiple versions of the same question. Quora values canonical questions because they provide a better experience to active seekers and writers, and offer more value to both of these groups in the long term.

The most naïve implementation for this problem is to build a machine learning model which can predict whether a pair of question is duplicate or not. For example, in this case, a supervised learning model can be trained based on certain features to predict whether a question pair is duplicate or not. Beforehand prediction can help Quora to provide the users a better experience.

1.2 Problem Statement

The goal is to build a binary classification model using a simulated dataset containing a pair of questions and a binary class label stating whether a pair is duplicate or not. In this project, I will be handling this

problem by applying advanced techniques (Random Forest, K-Means, SVM, XGBoost etc.) to classify whether question pairs are duplicates or not. After applying several models, I'll be comparing the accuracy obtained with each model.

Doing so will make it easier to find high quality answers to questions resulting in an improved experience for Quora writers, seekers, and readers.

1.3 Metrics

Our goal is to predict the probability that the questions are duplicates (a number between 0 and 1 for each ID in the test set. The results obtained will be evaluated on the log loss between the predicted values and the ground truth.
One of the best ways to evaluate the performance of the benchmark model and the solution model is to measure the accuracy using **cross validation**.

Cross-validation: It is a technique to evaluate predictive models by partitioning the original sample into a training set to train the model, and a test set to evaluate it.
In k-fold cross-validation, the original sample is randomly partitioned into k equal size subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k-1 subsamples are used as training data. The cross-validation process is then repeated k times (the folds), with each of the k subsamples used exactly once as the validation data. The k results from the folds can then be averaged (or otherwise combined) to produce a single estimation. *The advantage of this method is that all observations are used for both training and validation, and each observation is used for validation exactly once.*

For handling the stated problem, I will try to minimize the log loss obtained for each model implemented. The model having the **least log loss value** will be evaluated as the best model in order to tackle this problem.

Log Loss: It takes into account the uncertainty of your prediction based on how much it varies from the actual label. This gives us a more nuanced view into the performance of our model. It is calculated as:

$-\log P(y_t \mid y_p) = -(y_t \log(y) + (1 - y_t) \log(1 - y_p))$; wherein,

y_t indicates the true label in {0,1} for a single sample and y_p indicates the corresponding probability that $y_t = 1$

2. Analysis

2.1 Data Exploration

Dataset used: Quora dataset provided on the Kaggle Competition (~ 4,00,000 records) : https://www.kaggle.com/quora/question-pairs-dataset
The dataset contains 404351 rows and 6 columns (id, qid1, qid2, question1, question1, is_duplicate)
Features:

- id - the id of a training set question pair – (Numeric)
- qid1, qid2 - unique ids of each question (only available in train.csv) – (Numeric)
- question1, question2 - the full text of each question – (String)

Target Variable:

- is_duplicate - the target variable, set to 1 if question1 and question2 have essentially the same meaning, and 0 otherwise. – (Numeric)

First 5 rows of the dataset are shown below:

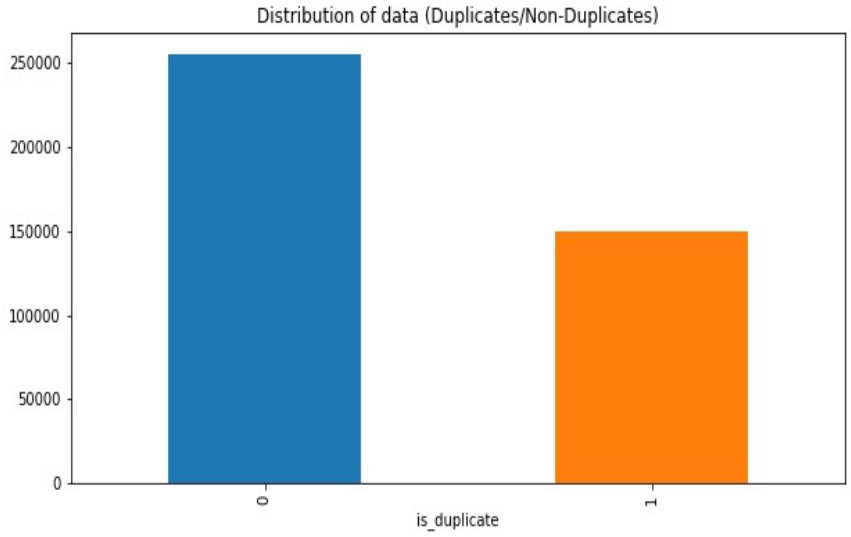
	id	qid1	qid2	question1	question2	is_duplicate
0	0	1	2	What is the step by step guide to invest in sh...	What is the step by step guide to invest in sh...	0
1	1	3	4	What is the story of Kohinoor (Koh-I Noor) Dia...	What would happen if the Indian government sto...	0
2	2	5	6	How can I increase the speed 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

Some descriptive statistics of the numerical features are as follows:

	id	qid1	qid2	duplicate
count	404351.000000	404351.000000	404351.000000	404351.000000
mean	202175.000000	391840.987691	390195.973765	0.369248
Std	116726.223686	228430.857607	228803.645742	0.482602
min	0.000000	1.000000	2.000000	0.000000
25%	101087.500000	193381.000000	191012.000000	0.000000
50%	202175.000000	390630.000000	388364.000000	0.000000
75%	303262.500000	589514.000000	588071.000000	1.000000
max	404350.000000	789800.000000	789801.000000	1.000000

Some other observations about the dataset:

- Total number of questions in the dataset: 404351
- Total number of duplicate questions in the dataset: 149306
- Ratio of duplicate questions in the dataset: 36.92%

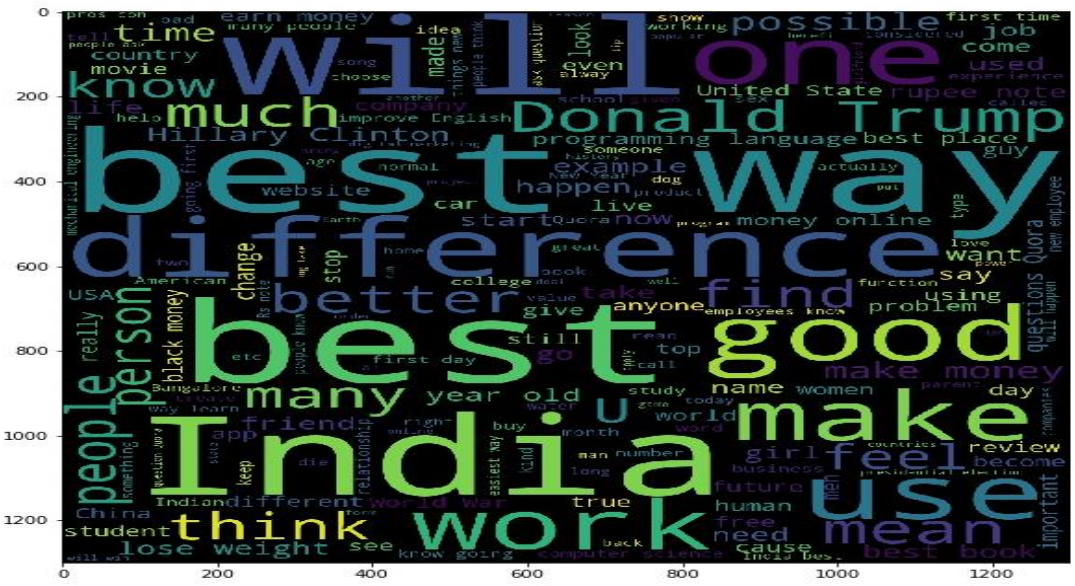


Word Cloud is a data visualization technique used for representing text data in which the size of each word indicates its frequency or importance. Significant textual data points can be highlighted using a word cloud. Here I have separated question1 and question2 into two corpuses and generated plots using wordcloud

library.

Question 1 Plot:

Question 2 Plot:



Observation: The above two plots clearly shows that there are some frequently occuring common words in the two questions (question1 and question2). For example: 'best', difference', ' Donald Trump' etc.

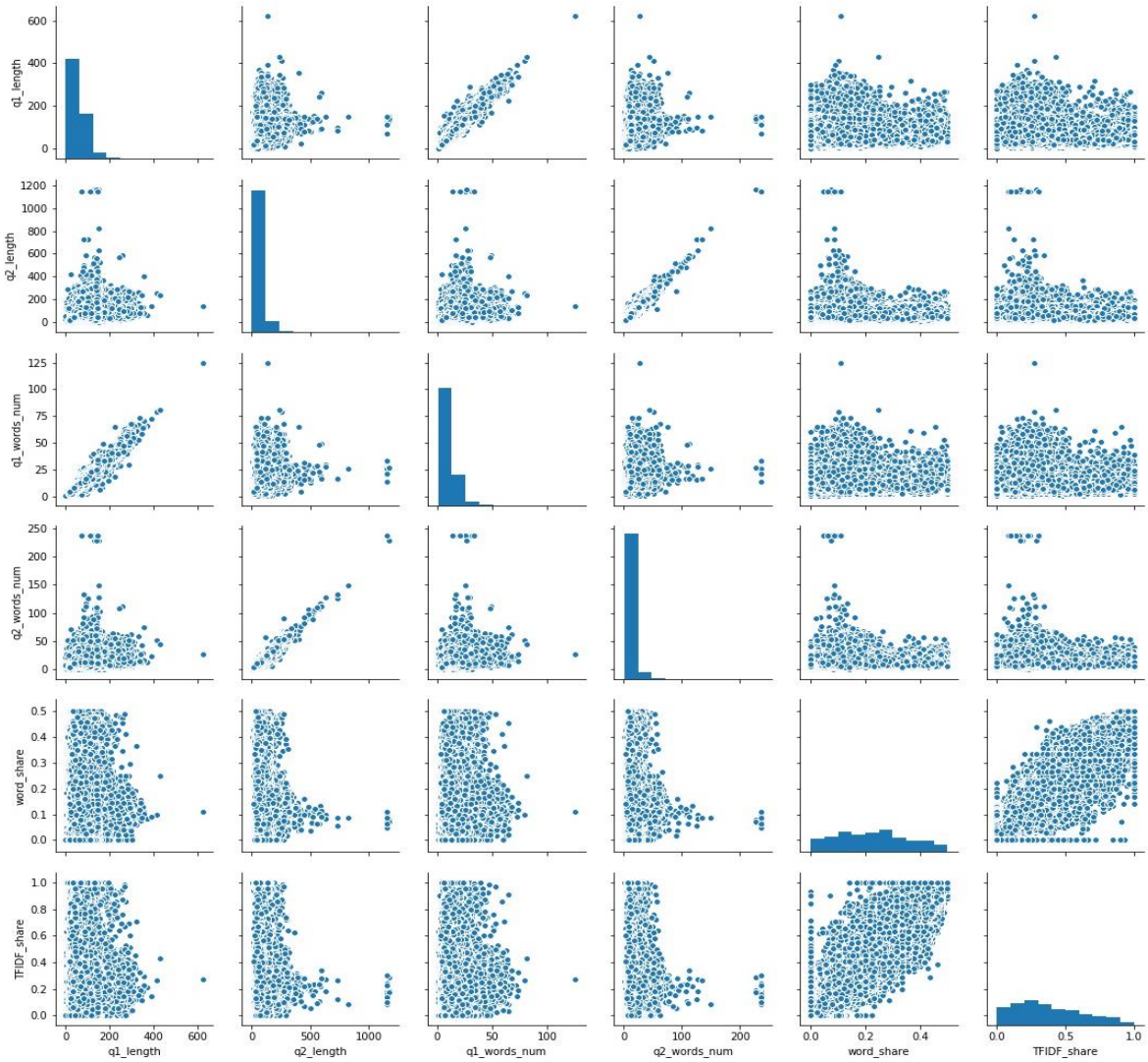
2.2 Exploratory Visualization

I have extracted features from the dataset listed below:

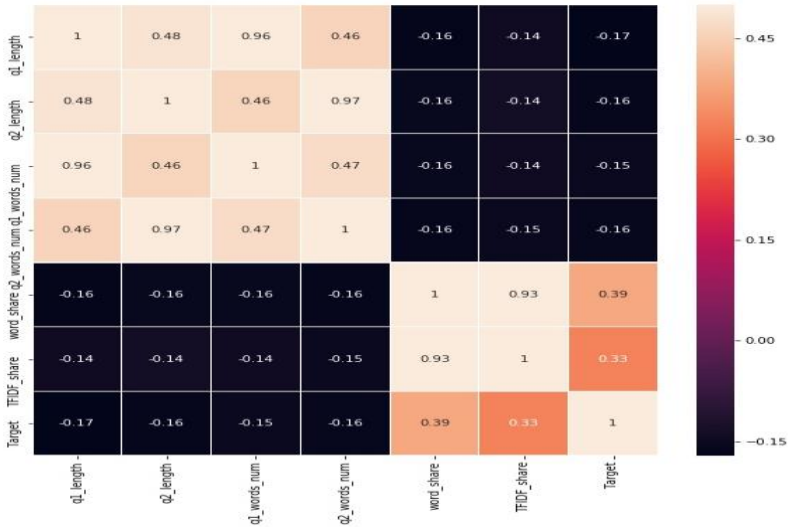
1. length of the questions
2. number of words
3. word share
4. tf-idf word share

Visualizing the above features:

1. Visualizing the pairwise relationships between the features constructed in a dataset :

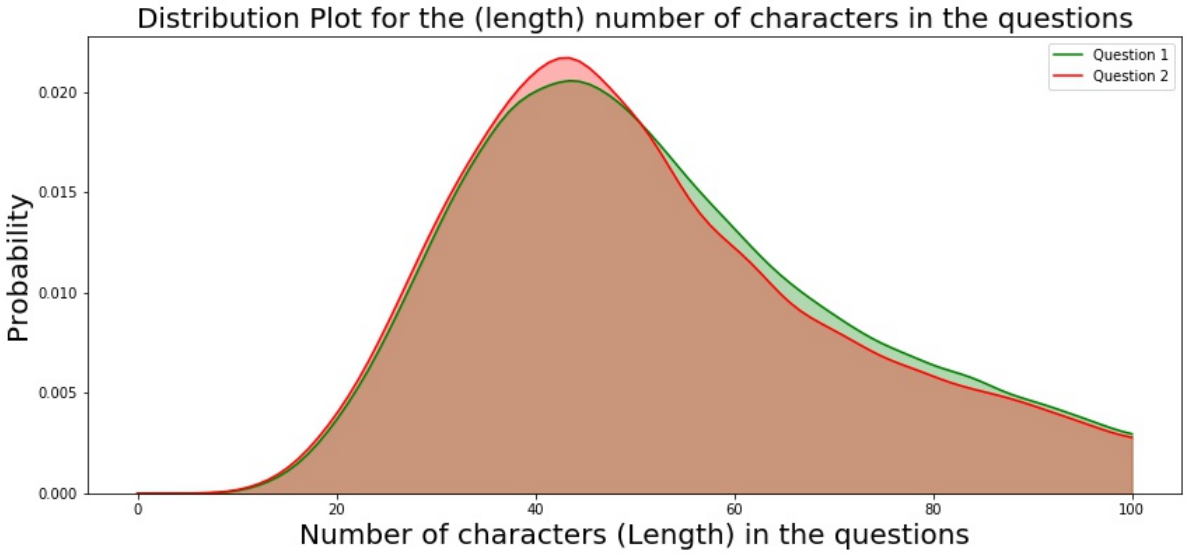


2. Correlation Matrix between the features constructed



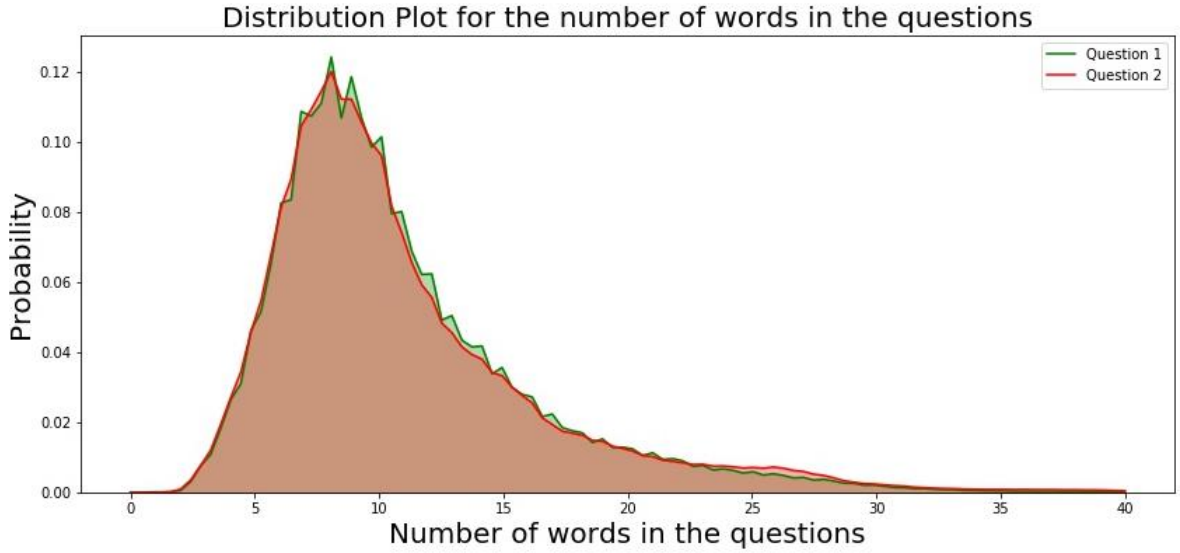
Observation: The above correlation matrix clearly shows that there is a high positive correlation between word_share, TF-IDF_Share and the target (indicating whether a question is duplicate or not) with correlation coefficient of 0.39, 0.32 respectively, which makes sense too.

3. Visualisation based on the feature: Length (Character Count)



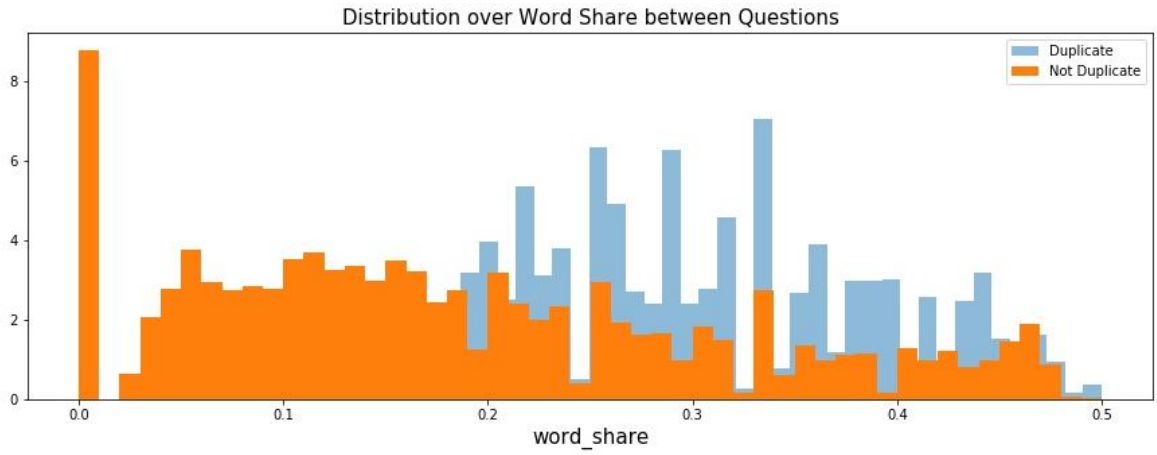
Observation: The above plot clearly shows that most of the questions are of length : 40-50

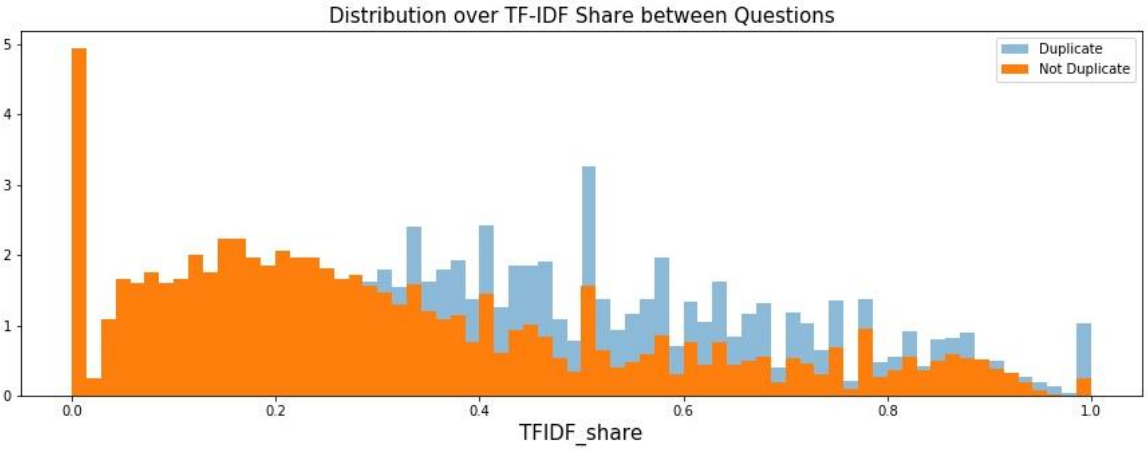
4. Visualisation based on the feature: Number of words in the question



Observation: The above plot clearly shows that most of the questions have 7-11 words.

5. Visualisation based on the features: Word Share and TF-IDF Word Share





Observation: The above two plots clearly shows that the ratio of both word share and tf-idf word share is higher for duplicate questions than for non-duplicate questions.

2.3 Algorithms and Techniques

Since the given problem is a binary classification problem so I have used several supervised learning algorithms and then afterwards have compared the accuracy of all the implemented algorithms.

Firstly, I have constructed the following features from the dataset:
1. length of the questions
2. number of words
3. word share
4. tf-idf word share: For finding the tf-idf word share, I have used CountVectorizer and TfidfTransformer from sklearn; which converts a collection of raw documents to a matrix of TF-IDF features.

Secondly, I have implemented the following supervised algorithms:

1. ***Random forest***: Random forest is an ensemble learning method. It is applied for both regression and classification problems. In classification problems, it is constructed as an ensemble of decision trees and outputs the class which is the mean of the classes all the trees. It reduces overfitting and thus, gives high accuracy. It automatically learns feature interactions and output feature importance. It is very expensive

and very difficult to interpret. It uses bagging approach which is an approach where you take random samples of data, build learning algorithms and take simple means to find bagging probabilities.

2. ***Decision trees:*** This algorithm uses a tree-like structure to do binary classification at each node and a target prediction is returned at the leaf of the tree. Like random forest, it is also capable of solving both classification and regression tasks and is also capable of learning feature interactions and outputting features importance. It is simple to interpret. This algorithm, however, does not work well when the problem is probabilistic.

3. ***Logistic Regression***: It is a regression model where the target variable nature is binary (0 or 1) . It computes a weighted sum of input features and outputs the logistic of the results. It is simple, fast and easy to interpret. It has low variance, so is less prone to overfitting.

4. ***Support Vector Machine***: It is an algorithm which analyses both classification and regression tasks. It takes input as a labelled training data and outputs an optimal hyperplane which separated the input labelled data and categorizes new examples.

5***. K Nearest neighbours***: It is also used for both regression and classification problems. It generally uses a lazy learning, where the function is only approximated locally and all the final computations are deferred until classification. This model calculates the average of k nearest data points to predict the testing data value. It provides good accuracy and is simple to understand too but is very sensitive to the local structure or noisy data.

6. ***Gradient boosting (XGBoost):*** It can also be applied to both regression and classification problems. This algorithm is similar to bagging only but here, the selection of sample is made more intelligently; subsequently more and more weight is given to classify the observations. It is mostly similar to random – forest algorithm but a prediction score is assigned instead of a binary value to each leaf, and during training one new tree is added to the ensemble and further optimization is done.

In order to evaluate each algorithm, I have used 20-fold cross validation approach to compare the above supervised algorithms and then selecting the best one.

2.4 Benchmark

Benchmarking here means, a standard solution which already performs well. Thus,

1. First, I will keep a very naïve model as my benchmarking model: x% change of getting a duplicate question pair, which is , nothing but the percentage of total duplicate pairs in the dataset i.e.,

   (Number of duplicate pairs in dataset)/ (Total number of records in dataset) * 100

   The accuracy obtained for the naïve classifier is: 36.92%

2. Second, keeping in mind that currently, Quora uses a Random Forest model to identify duplicate questions.

   So, first I will implement Random Forest Model to identify the duplicate questions and then I will use that model as a benchmark model and by applying several different models listed above, I will try to either cross or at-least match the benchmark model's accuracy.

   The accuracy obtained for Random Forest (20-fold Cross Validation): 69.93 %

3. Methodology

3.1 Data Preprocessing

Before the dataset is given to the model for training, pre-processing need to be performed. The following pre-processing of data I have performed:

1. Feature Extraction: Firstly, I have constructed features from the dataset. The following features have been constructed:
    a. Length of the question (Q1 and Q2) : Extracted using string functions in python
    b. Number of words (Q1 and Q2) : Extracted using string functions in python
    c. Word Share: The value is a normalized value which is calculated as the number of words in both question1 and question2 divided by the total number of words in question1 and question2

        word_share = (Number of words in Q1 and Q2) /
        (Number of words in Q1 + Number of words in Q2)

    d. TF-IDF Share: For finding the tf-idf word share, I have used CountVectorizer and TfidfTransformer from sklearn; which converts a collection of raw documents to a matrix of TF-IDF features.

2. Removing NaN values: Checking if any NaN values are there in the dataset; if exists replace them with zeros.

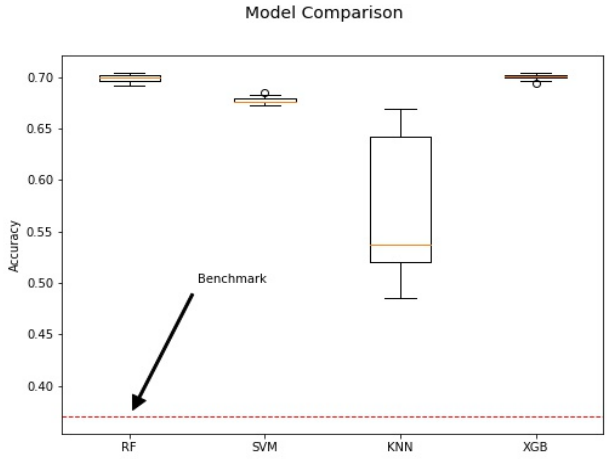
3. Normalization of numerical features: Normalization is done to standardize the range of numerical features. It standardizes features by removing the mean and scaling to unit variance. I have used StandardScaler() from sklearn for feature scaling.

3.2 Implementation

After performing the necessary data preprocessing, I have trained my dataset on six models and I have evaluated each of the model using k-fold cross validation (k=20). The details are given below:

Random Forest, Decision Tree are trained and are evaluated using k-fold cross validation. The accuracy obtained is 69.93% and 67.75%. Then I trained Logistic Regression model and the accuracy obtained is 66.36%. I trained SVM default parameters (max_iter = 500 to avoid memory issue) without performing grid search, the accuracy obtained is 67.85%. Then, I again trained SVM but this time used exhaustive grid search to find the best parameters. The best parameters obtained are {‘kernel’:’’sigmoid’, ‘C’: ‘0.5’} and the accuracy obtained is 56.82%. Then I trained KNN and performed exhaustive grid search to find the best parameters. The best parameters obtained are : {'n_neighbors': 2, 'weights': 'uniform'} and the accuracy obtained is 70.05%. For XGboost, with default parameters and k-fold cross validation, the accuracy obtained is 70.86% which is better than the benchmark models (naïve classifier: 36.92% and random-forest: 69.93%).

The models are compared by creating the boxplots with all cross validation scores, as shown below:



3.3 Refinement

The XGboost model trained with default parameters and evaluated with k-fold cross validation gave better accuracy than the benchmark models. So, I tried tuning the parameters of XGboost to obtain more accuracy. I performed exhaustive grid search to obtain the best parameters and k-fold cross validation for evaluation. I also played with different parameters 'eta', 'max_depth' etc. The final accuracy obtained is 72.4%.

Accuracy can be improved if more number of features are added to the dataset.

4 Results

4.1 Model Evaluation and Validation

The final model I selected to solve the problem is XGBoost (Gradient Boosting). The model is pretty robust because a boosted tree uses many trees to do the training, and unlike random forest, it uses scores in the leaves instead of binary classification.

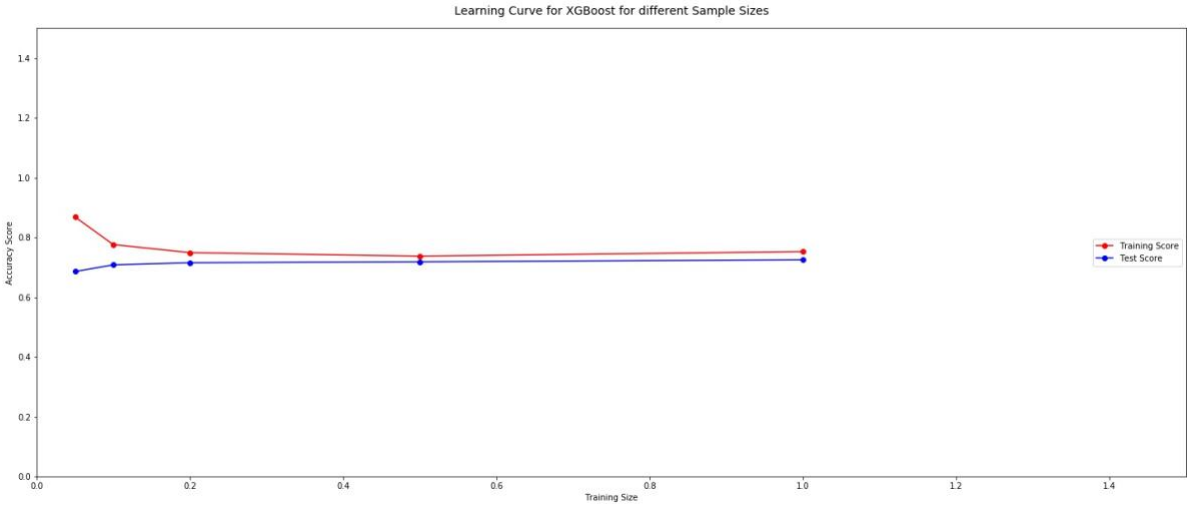
The final result obtained is better than both the benchmark models (naïve classifier and random forest classifier). The log-loss obtained is pretty less and the accuracy obtained is 72.4%

To determine the robustness of the final model selected, the discrepancy between the training and test accuracy scores is evaluated by first randomly splitting the original dataset in the ratio 80:20 (80% training set and 20% testing set) and afterwards, I have trained the same model with different training sizes.

The learning curves for the training and testing set scores is shown below:

The curves for the training and testing sets scores converges as the training size increases. Hence, the below curve clearly indicates that the model generalizes well to the unseen data as well; I am confident that my model is valid.

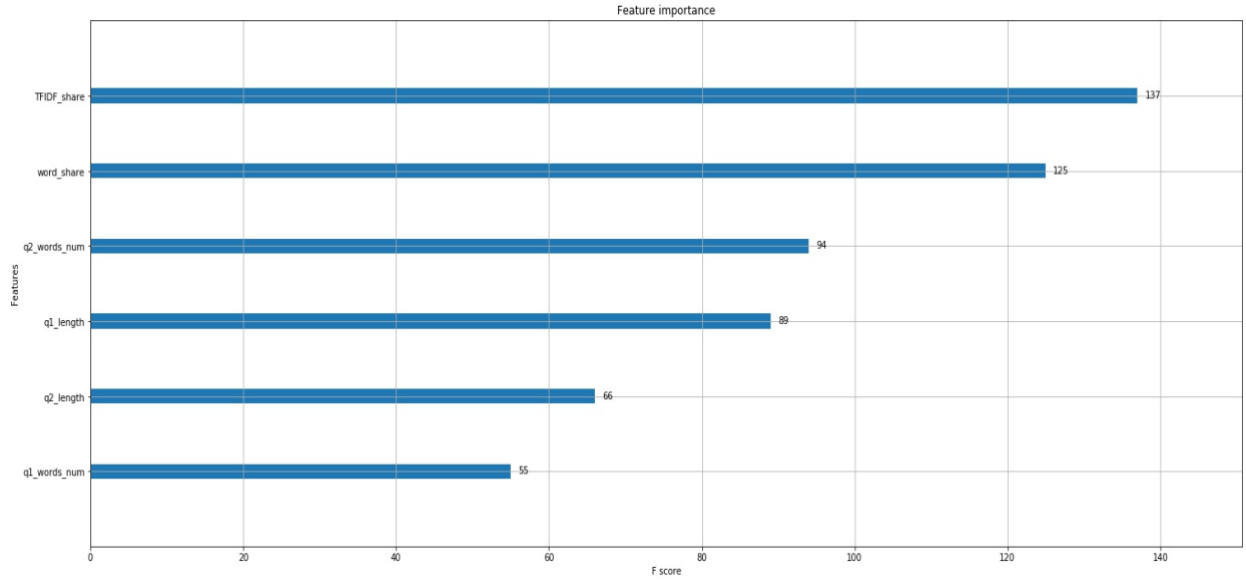
Also, identifying more than 70% of duplicate questions is pretty good enough for providing a good user experience.



5 Conclusion

5.1 Feature Importance

For XGboost , I have obtained the feature importance from the trained model. The below plot show the features along with their respective importance:



The above plot shows that TF-IDF is the most important feature followed by the ‘word share’ feature which is true logically too.

5.2 Reflection

To summarize, this project deals with a problem of identifying of pairs of duplicate questions which is basically a natural language processing problem. To tackle this problem, I have extracted some features from the given dataset like TF-IDF share, number of characters in the question (question length), number of words in the question and word share. After performing feature scaling, I have trained six machine

learning models to help identifying the duplicate questions. The models built are Random Forest, KNN, Logistic Regression, Decision Trees, SVM, XGboost and evaluation method used is k-fold Cross Validation. The models are also tuned using exhaustive grid search, GridSearchCV. Further, I have also shown the relative importance of each feature used in training the model. The final model shows an excellent accuracy (better than the benchmark models (naïve classifier and random forest classifier)) and appears to be generalizable to unseen data as well.

5.3 Improvement

The final model trained on the features extracted shows an excellent accuracy. However, more improvement can be done. I will try to add more number of features using sentiment analysis to the dataset, which will help in giving a better accuracy. In my opinion, since this is more like a natural language processing problem, so several natural language processing techniques can be applied to improve the model and a more detailed analysis (handling special characters, math symbols etc) is required to extract the features from the dataset.

References:

1. http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html
2. https://www.kaggle.com/quora/question-pairs-dataset
3. https://www.analyticsvidhya.com/blog/2016/03/complete-guide-parameter-tuning-xgboost-with-codes-python/
4. http://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html