Quora Duplicate Question Classifier

Semester	Winter2021
Course Code	AIML 2304
Group Name	Python Thinkers
Student names/Student IDs	Anto Francis (C0825095) Omer Volkan Guney (C0856373) Rupesh Chandran (C0826779) Sachin Sreekumar (C0825096)
Faculty Supervisor	Harriet Huang

Submission date: ***12/04/2022***

Contents

Abstract

Quora is a question-answering platform that lets users ask questions and get answers on them. It connects users having the same problems and allows them to share knowledge with the public. This allows people to get knowledge quickly which makes their life easy. The most common issue which has been faced by Quora users is question duplication. Sometimes users ask similar questions which have been answered before which results in question duplication. This makes the writers feel like they have to answer the same questions multiple times which reduces Quora's experience. Our aim is to resolve this problem by applying advanced NLP techniques to classify whether questions are repeated or not. This allows users to find good-quality answers easily.

Introduction

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.

Problem Statement

- Identify which questions asked on Quora are duplicates of questions that have already been asked.
- This could be useful to instantly provide answers to questions that have already been answered.

- We are tasked with predicting whether a pair of questions are duplicates or not.

Source of Data

Data Source: https://www.kaggle.com/c/quora-question-pairs

- Train.csv contains 5 columns: qid1, qid2, question1, question2, is_duplicate
- Size of Train.csv - 60MB
- Number of rows in Train.csv = 404,290

Data fields

- id - the id of a training set question pair
- qid1, qid2 - unique ids of each question (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.

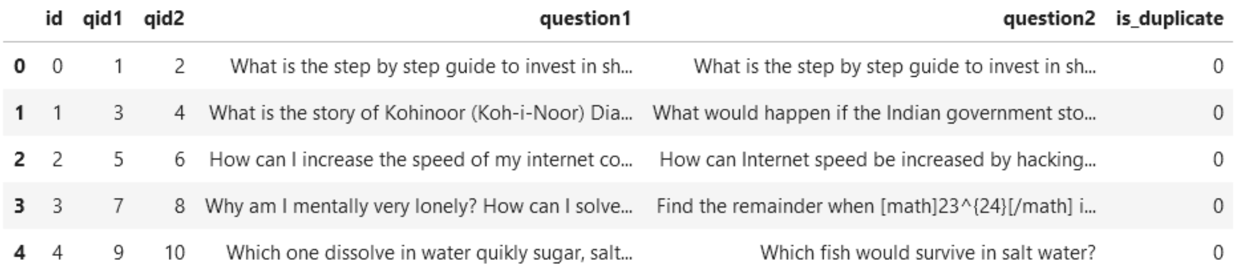
It is a binary classification problem, for a given pair of questions we need to predict if they are duplicate or not. This could be useful to instantly provide answers to questions that have already been answered.

Methods

We will be using different mechanisms to identify whether questions are similar or not. We will be analyzing common words, first and last words, and fuzz ratios for our analysis. We will also be using cosine similarity and Euclidean distance to see the similarity. Finally, we will be using SVC and Random Forest Classifiers to classify duplicate questions.

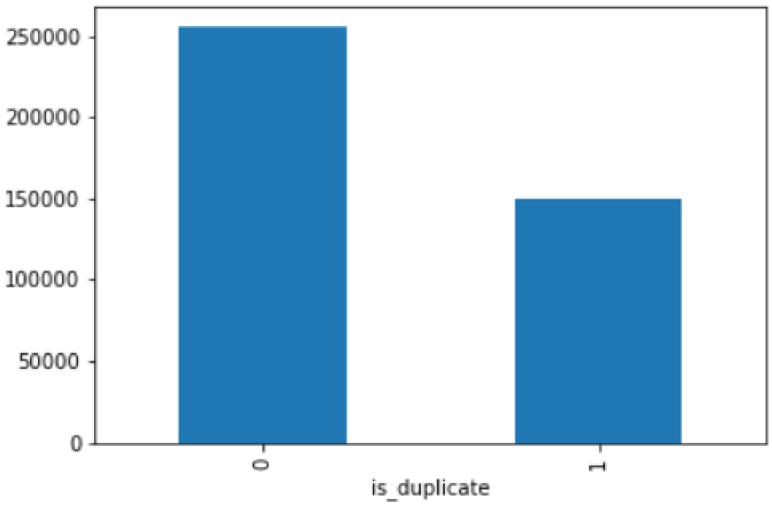
Exploratory Data Analysis

- First 5 rows of the dataset:

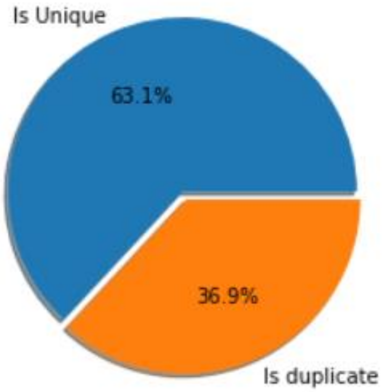


	id	qid1	qid2	question1	question2	is_duplicate
0	0	1	2	What is the step by step guide to invest in sh...	What is the step by step guide to invest in sh...	0
1	1	3	4	What is the story of Kohinoor (Koh-i-Noor) Dia...	What would happen if the Indian government sto...	0
2	2	5	6	How can I increase the speed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

- Distribution of data points among output classes

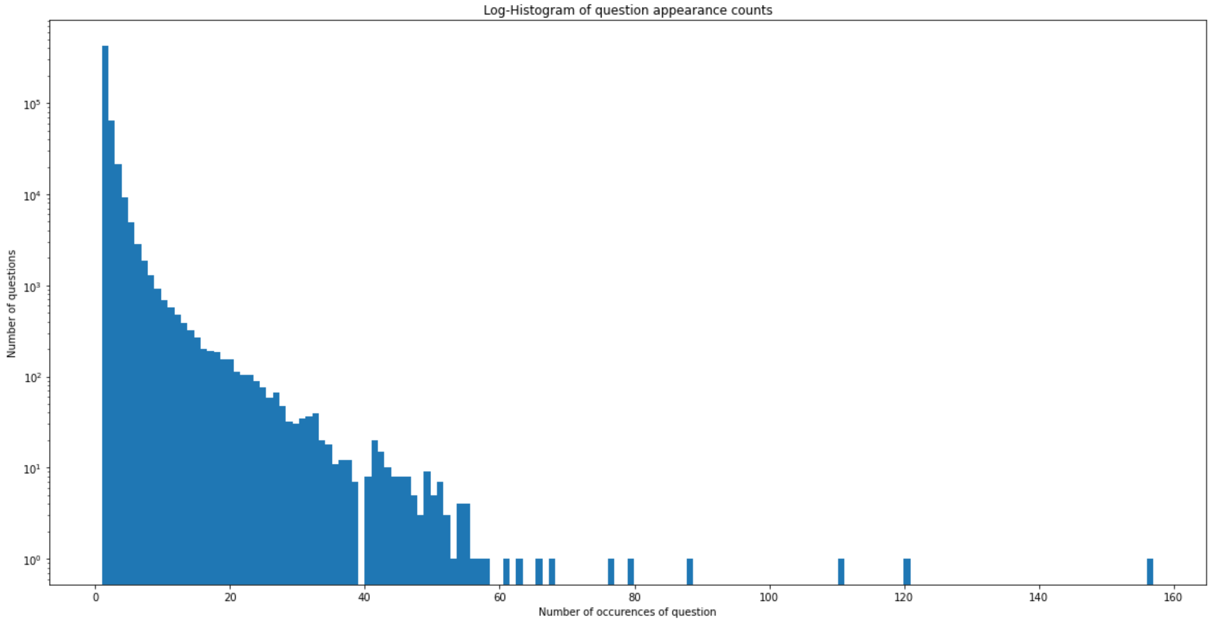


- Number of unique questions:
  - Total number of unique questions: 537933
  - Number of unique questions that appear more than one time: 111780 (20.78%)
  - Max number of times a single question is repeated: 157



Plot representing unique and repeated question

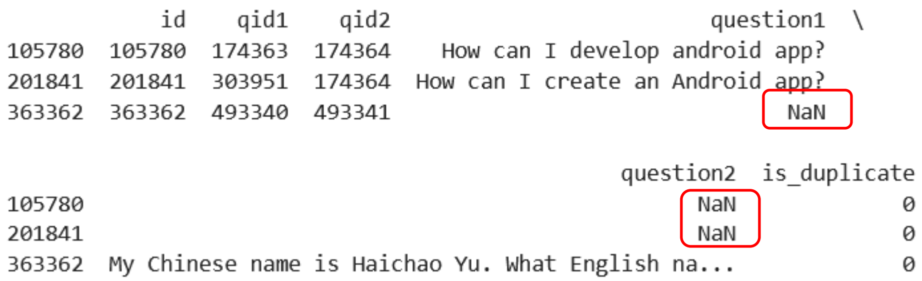
- ➢ Number of occurrences of each question



Log-Histogram of question appearance counts

Maximum number of times a single question is repeated: 157

- Checking for NULL values



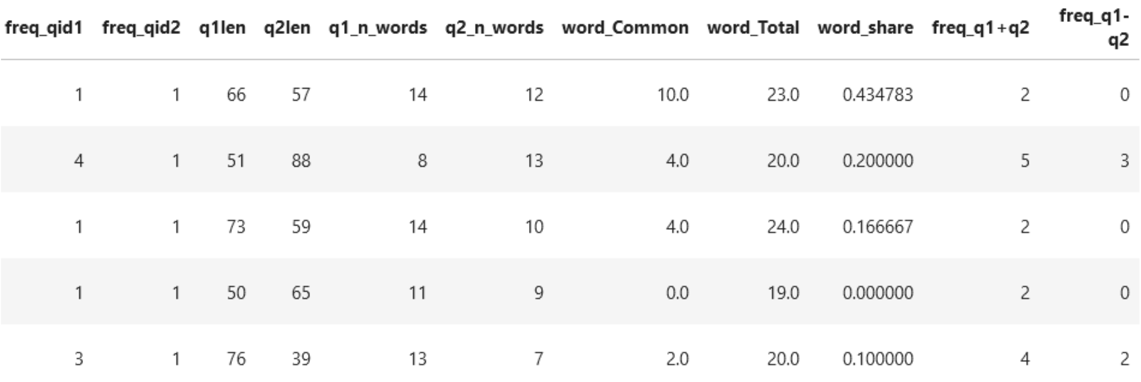
```
            id    qid1    qid2                         question1  \
105780  105780  174363  174364    How can I develop android app?
201841  201841  303951  174364  How can I create an Android app?
363362  363362  493340  493341                               NaN

                                               question2  is_duplicate
105780                                               NaN             0
201841                                               NaN             0
363362  My Chinese name is Haichao Yu. What English na...             0
```

We found 3 rows with null values. Since the value is very low, we removed these rows.

Basic Feature Extraction (Pre-cleaning)

We added the following new features:

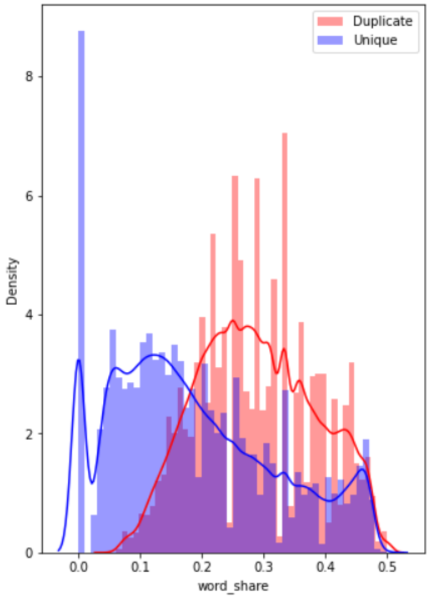
- freq_qid1 = Frequency of qid1's
- freq_qid2 = Frequency of qid2's
- q1len = Length of q1
- q2len = Length of q2
- q1_n_words = Number of words in Question 1
- q2_n_words = Number of words in Question 2
- word_Common = Number of common unique words in Question 1 and Question 2
- word_Total = Total num of words in Question 1 + Total num of words in Question 2
- word_share = word_common/word_Total
- freq_q1+freq_q2 = sum total of frequency of qid1 and qid2
- freq_q1-freq_q2 = absolute difference of frequency of qid1 and qid2



freq_qid1	freq_qid2	q1len	q2len	q1_n_words	q2_n_words	word_Common	word_Total	word_share	freq_q1+q2	freq_q1-q2
1	1	66	57	14	12	10.0	23.0	0.434783	2	0
4	1	51	88	8	13	4.0	20.0	0.200000	5	3
1	1	73	59	14	10	4.0	24.0	0.166667	2	0
1	1	50	65	11	9	0.0	19.0	0.000000	2	0
3	1	76	39	13	7	2.0	20.0	0.100000	4	2

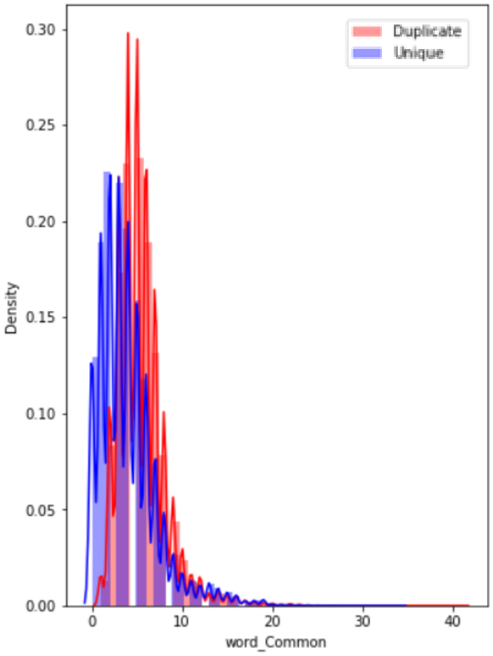
Features extracted from the dataset

- Analysis of the extracted features
  - Minimum length of the questions in question1: 1
  - Minimum length of the questions in question2: 1
  - Number of Questions with minimum length [question1]: 67
  - Number of Questions with minimum length [question2]: 24
- Feature: word_share



- ○ The distributions for normalized word_share have some overlap on the far right-hand side, i.e., there are quite a lot of questions with high word similarity
- ○ The average word share and Common no. of words of qid1 and qid2 is more when they are duplicate (Similar)

➢ Feature: word_common



- ○ The distributions of the word_Common feature in similar and non-similar questions are highly overlapping

Preprocessing of Text

- ○ Removing html tags
- ○ Removing Punctuations
- ○ Performing stemming
- ○ Removing Stopwords

- Expanding contractions etc.

Advanced Feature Extraction (NLP and Fuzzy Features)

Definition:

- Token: You get a token by splitting sentence a space
- Stop_Word : stop words as per NLTK.
- Word : A token that is not a stop_word

Features:

- cwc_min : Ratio of common_word_count to min lenghth of word count of Q1 and Q2
- cwc_min = common_word_count / (min(len(q1_words), len(q2_words))
- cwc_max : Ratio of common_word_count to max lenghth of word count of Q1 and Q2
- cwc_max = common_word_count / (max(len(q1_words), len(q2_words))
- csc_min : Ratio of common_stop_count to min lenghth of stop count of Q1 and Q2
- csc_min = common_stop_count / (min(len(q1_stops), len(q2_stops))
- csc_max : Ratio of common_stop_count to max lenghth of stop count of Q1 and Q2
- csc_max = common_stop_count / (max(len(q1_stops), len(q2_stops))
- ctc_min : Ratio of common_token_count to min lenghth of token count of Q1 and Q2
- ctc_min = common_token_count / (min(len(q1_tokens), len(q2_tokens))
- ctc_max : Ratio of common_token_count to max lenghth of token count of Q1 and Q2
- ctc_max = common_token_count / (max(len(q1_tokens), len(q2_tokens))
- last_word_eq : Check if Last word of both questions is equal or not
- last_word_eq = int(q1_tokens[-1] == q2_tokens[-1])

- first_word_eq : Check if First word of both questions is equal or not
- first_word_eq = int(q1_tokens[0] == q2_tokens[0])
- abs_len_diff : Abs. length difference
- abs_len_diff = abs(len(q1_tokens) - len(q2_tokens))
- mean_len : Average Token Length of both Questions
- mean_len = (len(q1_tokens) + len(q2_tokens))/2
- fuzz_ratio : https://github.com/seatgeek/fuzzywuzzy#usage

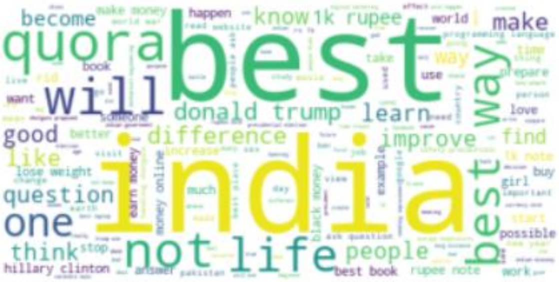
  http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/
- fuzz_partial_ratio : https://github.com/seatgeek/fuzzywuzzy#usage

  http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/
- token_sort_ratio : https://github.com/seatgeek/fuzzywuzzy#usage

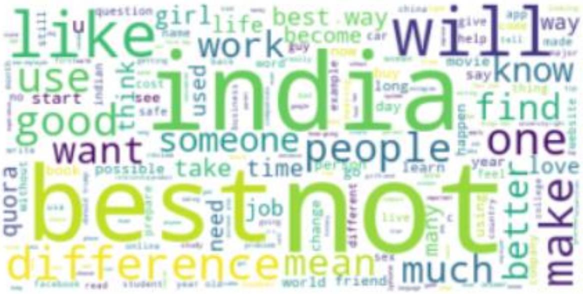
  http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/
- token_set_ratio : https://github.com/seatgeek/fuzzywuzzy#usage

  http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/
- longest_substr_ratio : Ratio of length longest common substring to min lenghth of token count of Q1 and Q2
- longest_substr_ratio = len(longest common substring) / (min(len(q1_tokens), len(q2_tokens))

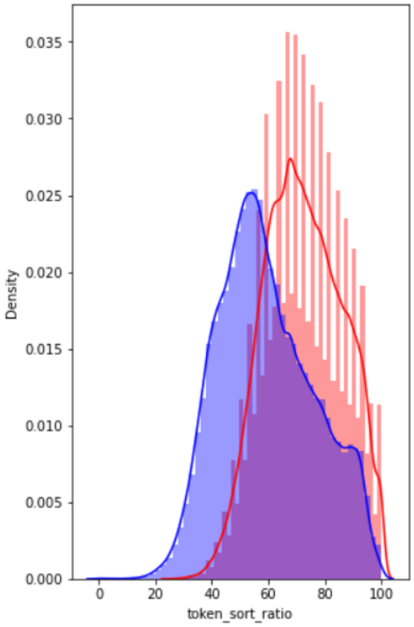
Word Cloud for duplicate question pairs



Word Cloud for unique question pairs



Distribution of the token_sort_ratio:



Featurization text data with tf-idf weighted word-vectors

TF or Term Frequency is the number of times the word *t* occurs in document *d* divided by the total number of the words in document *d*. In other words, it is the probability of finding a word in document *d*.

If a word occurs in more documents, then IDF decreases. The cell value is a multiplication of TF * IDF. More importance to rare words in documents and more important if a word is frequent in a document/review.

Specify the target value

Here, target is the column, is_duplicate, which denotes if the question pair is duplicate or note. The values of the columns are 0 (Is unique) or 1(is duplicate).

Splitting data into train and test with size=0.3

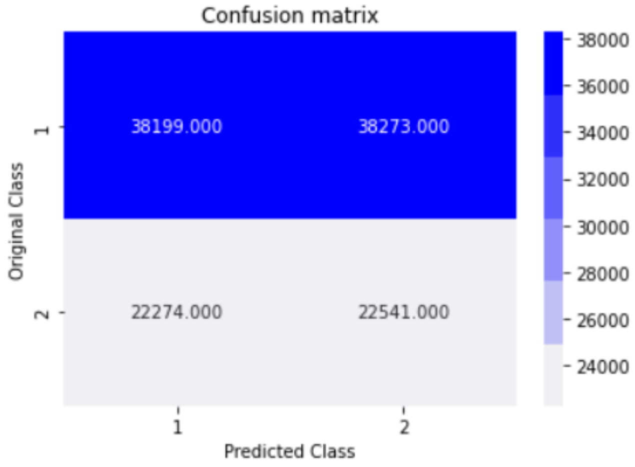
Number of data points in train data: (283000, 28)

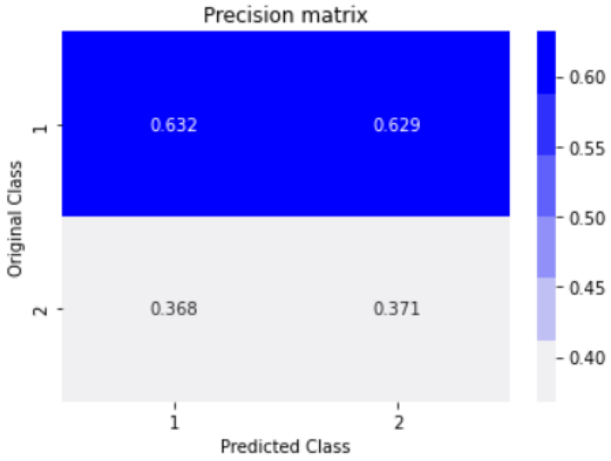
Number of data points in test data: (121287, 28)

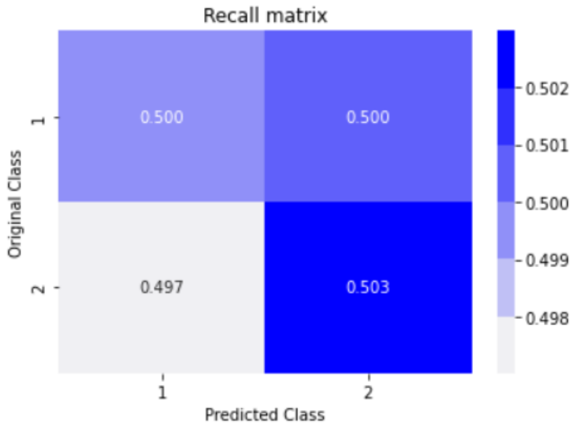
- Combining our question1 and question2 vectorized for train data
- Combining our question1 and question2 vectorized for test data
- Checking our class distribution in train and test data

Training on random model

Log loss on test data using Random Model: 0.887


Confusion matrix
38199.000
38273.000
22274.000
22541.000
Original Class
1
2
Predicted Class
1
2
38000
36000
34000
32000
30000
28000
26000
24000

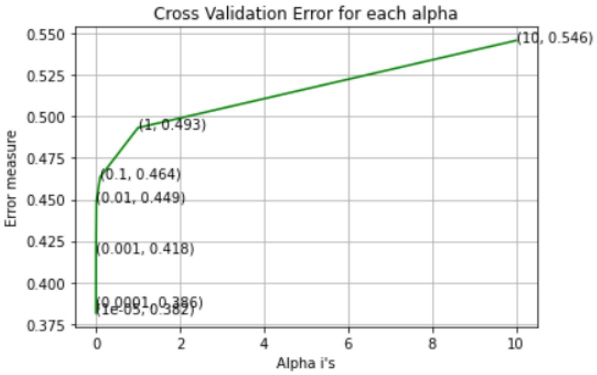

Precision matrix
0.632
0.629
0.368
0.371
Original Class
1
2
Predicted Class
1
2
0.60
0.55
0.50
0.45
0.40

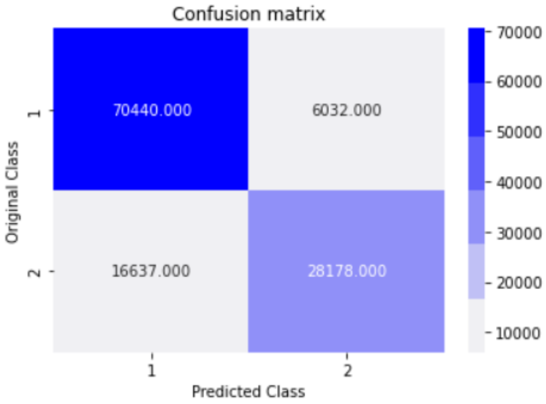

Recall matrix
0.500
0.500
0.497
0.503
Original Class
1
2
Predicted Class
1
2
0.502
0.501
0.500
0.499
0.498

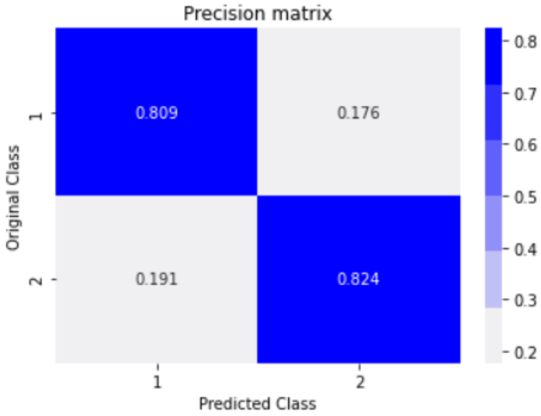
Training on Logistic Regression

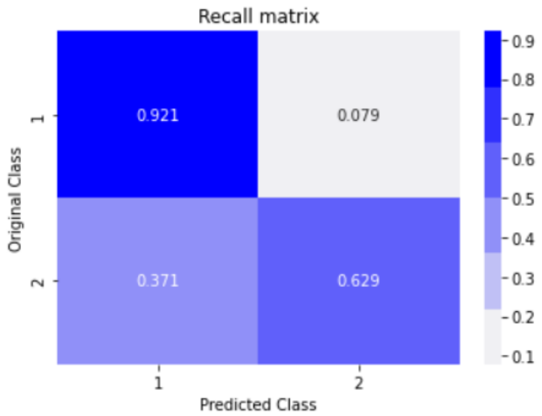
Log loss on train data: 0.373

Log loss on test data: 0.381





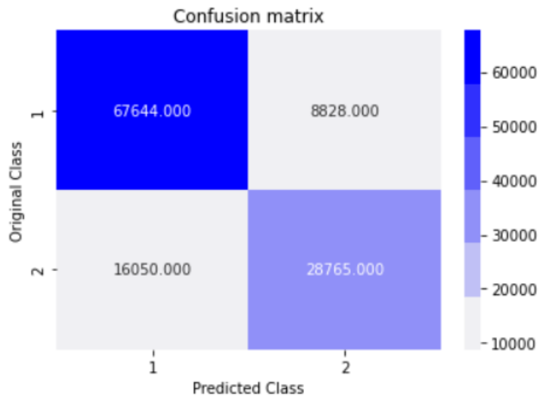


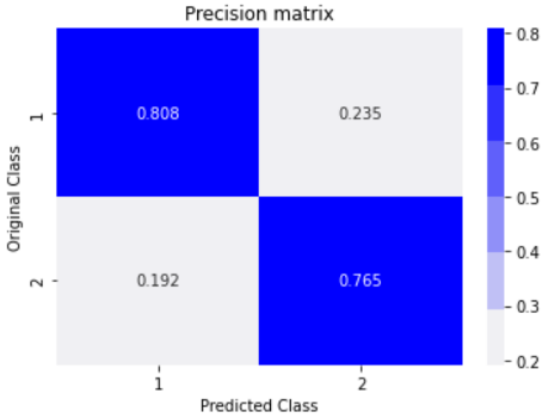


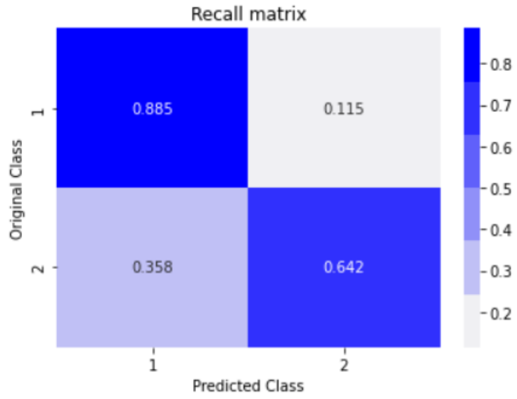
Training on Linear SVM

Log loss on train data: 0.421

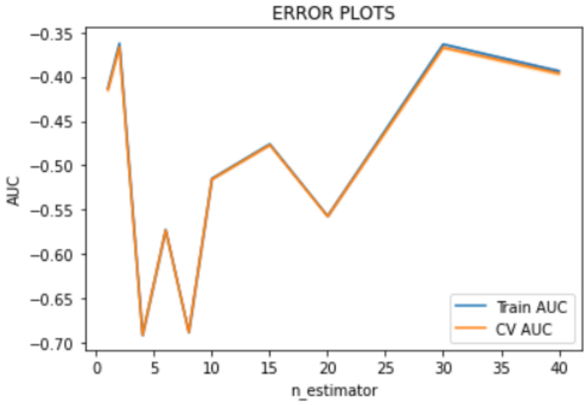
Log loss on test data: 0.431





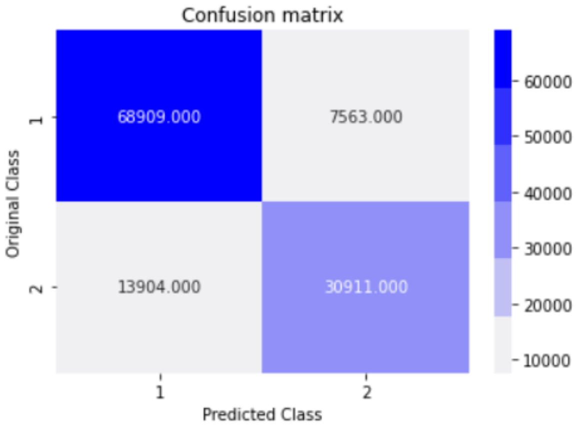


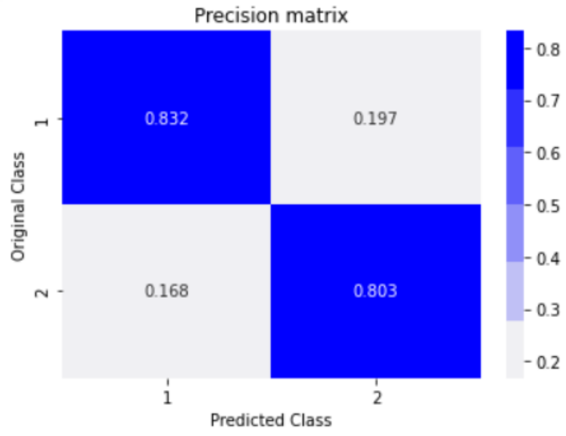
Training on Xgboost

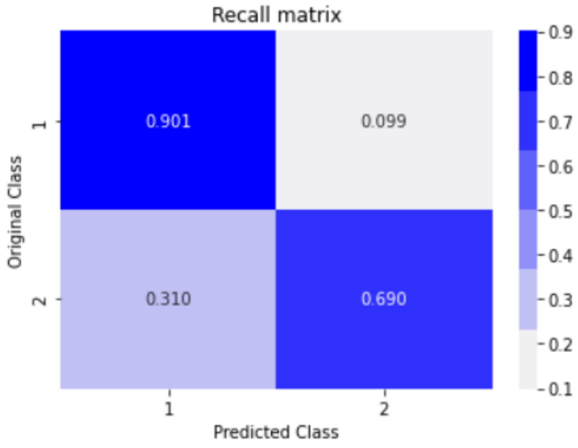


Log loss on train data: 0.363

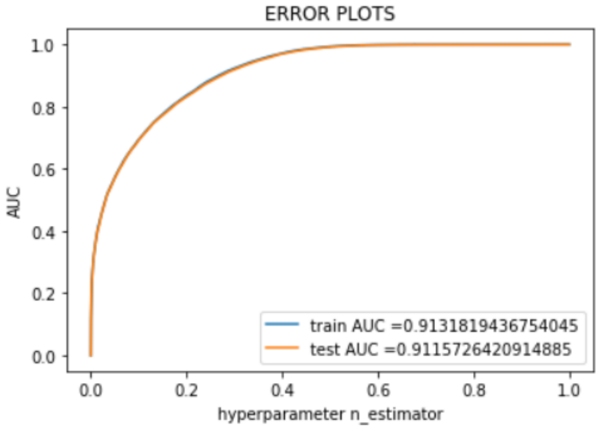
Log loss on test data: 0.366





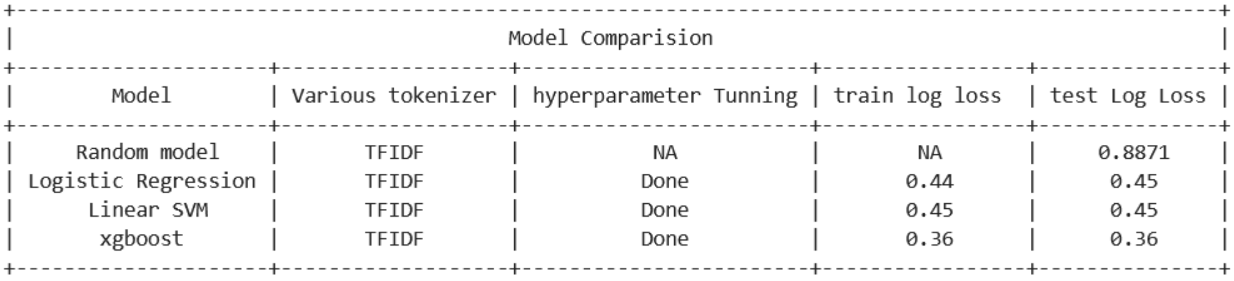


Plotting roc curve



Conclusions

Model Comparison



```
+---------------------------------------------------------------------------------------------------+
|                                          Model Comparision                                        |
+---------------------+-------------------+-----------------------+------------------+---------------+
|        Model        | Various tokenizer | hyperparameter Tunning | train log loss  | test Log Loss |
+---------------------+-------------------+-----------------------+------------------+---------------+
|     Random model    |       TFIDF       |           NA           |       NA        |     0.8871    |
| Logistic Regression |       TFIDF       |          Done          |      0.44       |      0.45     |
|      Linear SVM     |       TFIDF       |          Done          |      0.45       |      0.45     |
|       xgboost       |       TFIDF       |          Done          |      0.36       |      0.36     |
+---------------------+-------------------+-----------------------+------------------+---------------+
```

References

Sklearn.linear_model.Sgdclassifier. scikit. (n.d.). Retrieved April 12, 2022, from http://scikit-learn.org/stable/modules/generated/sklearn.linear_model.SGDClassifier.html

Cohen, P. by A. (n.d.). Chairnerd. Retrieved April 12, 2022, from http://chairnerd.seatgeek.com/fuzzywuzzy-fuzzy-string-matching-in-python/

When to use which fuzz function to compare 2 strings. Stack Overflow. Retrieved April 12, 2022, from https://stackoverflow.com/questions/31806695/when-to-use-which-fuzz-function-to-compare-2-strings

How to visualize dependence of Model Performance & Alpha with matplotlib? Stack Overflow. Retrieved April 12, 2022, from https://stackoverflow.com/a/48803361/4084039

Seatgeek. (n.d.). *Seatgeek/fuzzywuzzy: Fuzzy string matching in Python*. GitHub. Retrieved April 12, 2022, from https://github.com/seatgeek/fuzzywuzzy