Quora Question Pairs

Can we identify question pairs that share the same intent?

Project Overview

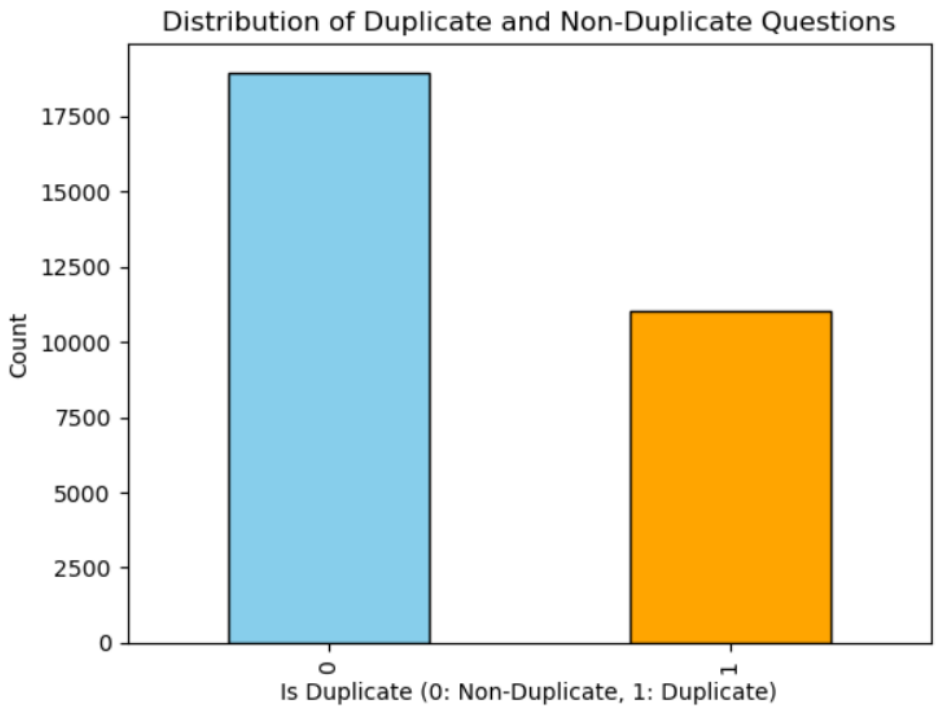
This project focuses on a Kaggle competition hosted by Quora. The objective is to identify pairs of questions that have the same underlying intent, even if they are phrased differently due to variations in wording or grammar. These subtle differences can lead to fragmented answers across multiple questions, negatively affecting user experience. Our goal is to build a model that, when given two questions, can predict whether they are duplicates.

Exploratory-Data-Analysis

```
Data columns (total 6 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   id            30000 non-null  int64
 1   qid1          30000 non-null  int64
 2   qid2          30000 non-null  int64
 3   question1     30000 non-null  object
 4   question2     30000 non-null  object
 5   is_duplicate  30000 non-null  int64
dtypes: int64(4), object(2)
```

id	qid1	qid2	question1	question2	is_duplicate
35471	10592	41171	How can plan a trip for 3 and half days to Goa...	How do I plan a trip to Goa?	1
325510	451766	451767	What are the steps to becoming a professional ...	How do I become a web designer step by step?	0
1618	3222	3223	What makes great movies incredibly engaging?	What are the best movies with a plot twist?	0
277490	396604	344829	Why does some content go viral on the internet?	How do I make a video go viral on WhatsApp?	0
114001	186168	125285	How should I prepare for a manual tester's job...	How can I prepare for manual testing interview...	0

1. We observe that each row contains two questions, their corresponding IDs, and a label indicating whether they are duplicates. Thus, this is a `BINARY CLASSIFICATION PROBLEM`.
2. As part of basic preprocessing, we drop any rows that are duplicates or contain null values.
3. Next, we check for **class imbalance**, an important consideration in classification tasks. To address class imbalance, we assign a higher penalty when the model misclassifies a minority class during training. This approach encourages the model to pay attention to minority classes, even if the overall accuracy is high.



Pre-Processing

As part of the preprocessing steps, I have performed the following transformations:

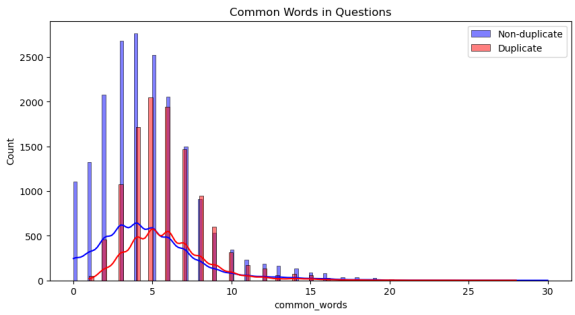
1. Converted all text to lowercase.
2. Replaced emojis with their meanings.
3. Expanded contractions (e.g., `'ve` to `have`).
4. Replaced special characters with their names.
5. Shortened large numbers with suffixes (`b` for billion, `m` for million, and `k` for thousand).
6. Removed HTML tags.
7. Abbreviated common terms (e.g., `GM` for `Good Morning`).
8. Applied stemming to reduce words to their root forms.

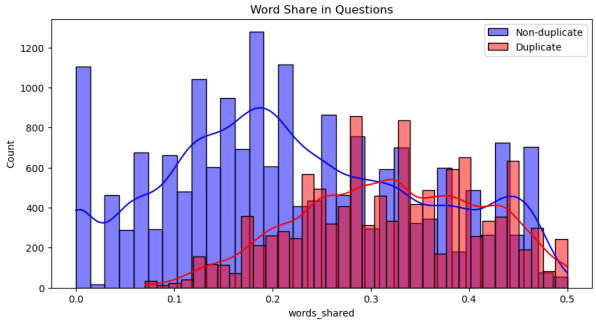
- **Why not remove stop words?** They will be used to create new features that will assist in classification later.
- **Why not perform spelling correction?** Simply because the data is too large, and it would consume a significant amount of time.

Feature-Engeneering

As part of creating my own features, I have calculated the following:
Batch 1 of Features

1. Length of question 1 (`q1`).
2. Length of question 2 (`q2`).
3. Number of words in question 1 (`q1`).
4. Number of words in question 2 (`q2`).
5. Common words between `q1` and `q2`.
6. Total words in `q1` + `q2`.
7. Ratio of common words to total words.





- We observe that when the number of common words is less than 5 (< 5), the questions tend to be classified as non-duplicates.
- Similarly, when the ratio of common words to total words is less than 3 (< 3), the questions are likely to be classified as non-duplicates.

Batch 2 of Features

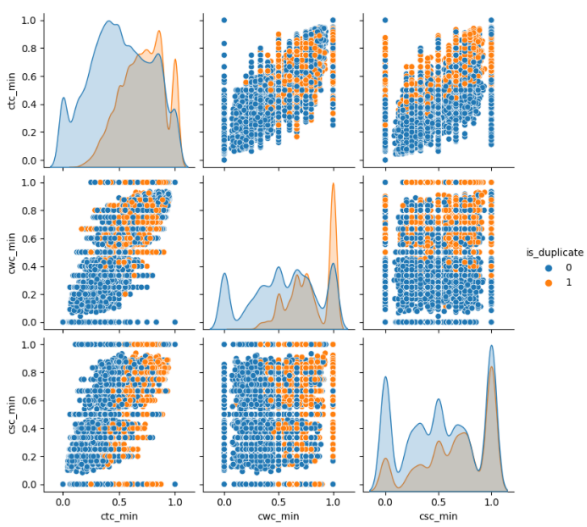
1. Ratio of common words minimum length of both questions.
2. Ratio of common words maximum length of both questions.
3. Ratio of common stop words to minimum length of both questions.
4. Ratio of common stop words to maximum length of both questions.
5. Ratio of common tokens to the minimum length of both questions.
6. Ratio of common tokens to the maximum length of both questions.
7. Last word of both question is same or not.
8. First word of both question is same or not.

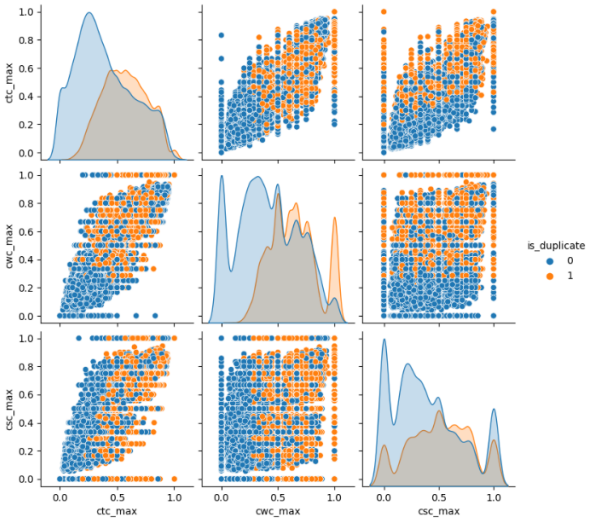
Batch 3 of Features

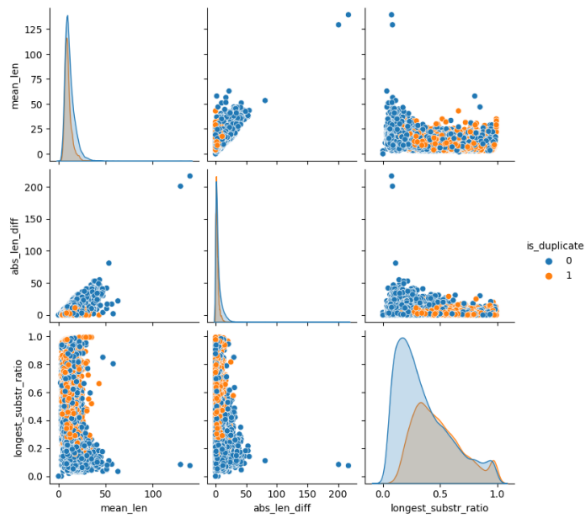
1. Absolute difference in the number of tokens in both questions.
2. Average number of tokens in both questions.
3. Ratio of the longest common substring to the minimum length of both questions.

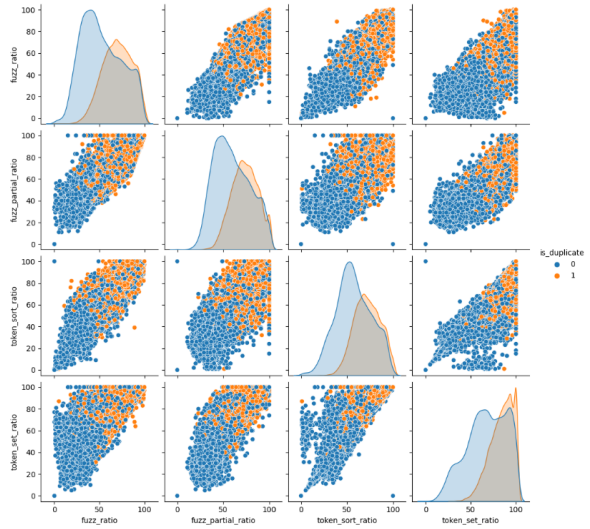
Batch 4 of Features

1. Fuzzy Ratio
2. Fuzzy Partial Ratio
3. Token Sort Ratio
4. Token Set Ratio









Now, we create a corpus of words using both columns of $q1$ and $q2$. Then, we select the top 3000 most frequent words as the dimensions of vectors to represent each sentence.

Although this method may not be ideal for several reasons, including:

- Sparse matrix representation,
- Lack of semantic meaning capturing,
- High-dimensional data,
- Out of vocabulary (OOV) issues,

For the time being, let's focus on the Bag of Words method.

```
We will now combine these 6000 columns (3000 each from q1 and q2) with the 23 features we created
earlier making a total of 6023 final features
```

Model Training & Evaluation

We split the data with 80% for training and the remaining 20% for testing. We will use a Random Forest classifier. The result we obtain after training is approximately 0.7863 on the test data. The corresponding confusion matrix is shown below:

```
[3228,  533]
[ 749, 1490]
```