Quora

Similarity in Questions

Table of contents

Introduction

Overview of Problem and Datasets

Given 2 questions, do they have the same meaning?

Why are so many Quora users posting questions that are readily answered on Google?

VS

Why do people ask Quora questions which can be answered easily by Google?

Overview of datasets

train.csv, train.tsv, dev.tsv

id	qid1	qid2	question1	question2	is_duplicate
…					

808566 rows

Overview of datasets

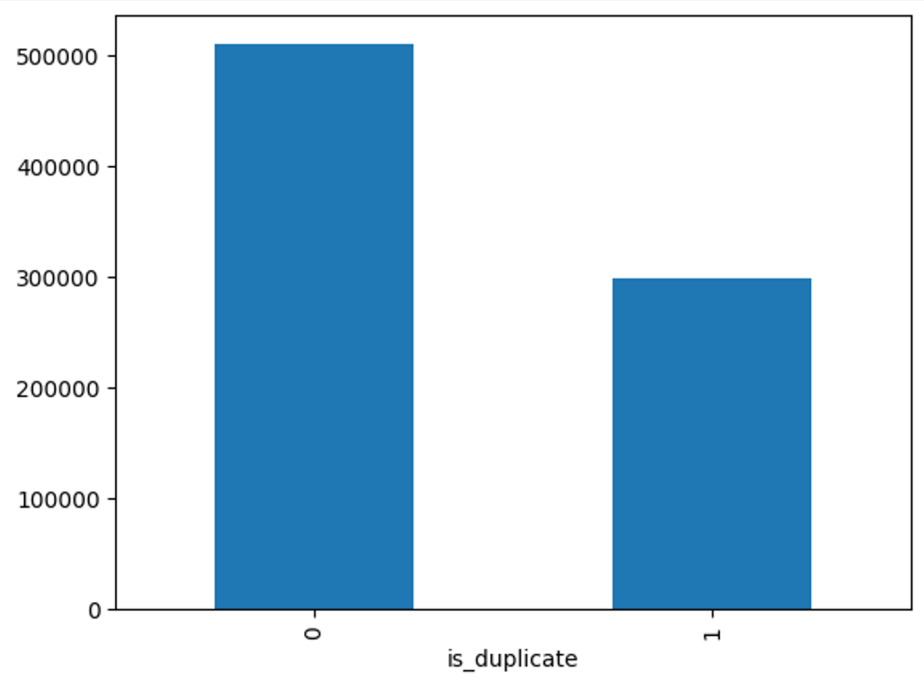
test.csv, test.tsv

id	question1	question2
…		

2736761 rows

Explain rationale behind choice of combined train dataset

Data Preprocessing



510040 (**is_duplicate=0**) vs 298526 (**is_duplicate=1**)

Explain rationale behind choice for changes made to dataset (E.g. removal of null values/empty rows)

```
[ ] df[df.isnull().any(axis=1)]
```

	id	qid1	qid2	question1	question2	is_duplicate
105780	105780	174363	174364	How can I develop android app?	NaN	0
201841	201841	303951	174364	How can I create an Android app?	NaN	0
363362	363362	493340	493341	NaN	My Chinese name is Haichao Yu. What English na...	0

Since there are very few entries with null values, we have decided to remove them

Data Cleaning

- Converting to lowercase
- Expanding Contractions
  - Contractions are shortened versions of words or syllables.
  - *E.g I'll → I will, don't → do not*
- Removing Hyperlinks
  - URLs contain a combination of letters, numbers, and special characters that may not contribute much to the semantic meaning of the text. Reduce the noise in the text data

Data Cleaning

- Removing HTML Tags
  - HTML tags are noise and don't add much value to understanding and analyzing text
  - *E.g <head>, <title>,
...*
- Standardizing Accent Characters
  - Convert characters into standard ASCII characters (*E.g è→ e, ã → a)*
  - Remove Non English characters
- Removing Mentions and Hashtags #,@
  - Don't generalize across samples and are noise in most NLP tasks

Data Preparation

Feature Engineering

Feature Engineering

1. Question length ratio
   - The questions will be split by words when vectorised so it is beneficial in enhancing similarity comparisons in text analysis

1. Number of words ratio
   - Ratio of the number of words in the shorter question over the length of the longer question

1. Same First and Last words
   - Duplicate question pairs tend to have a larger proportion of pairs with same first and last words, vice versa.

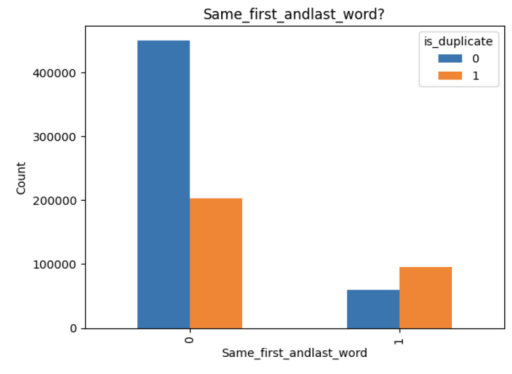


Fig.1

Feature Engineering

4. Cosine Similarity
   - Focuses on the direction of the vector
   - The cosine similarity ranges from -1 to 1.
     - 1 implies the documents are identical.
     - 0 implies the documents are orthogonal (no similarity).
     - −1 implies the documents are diametrically opposed.
   - Done by converting questions into bag of words representation and transforming into vector representation.

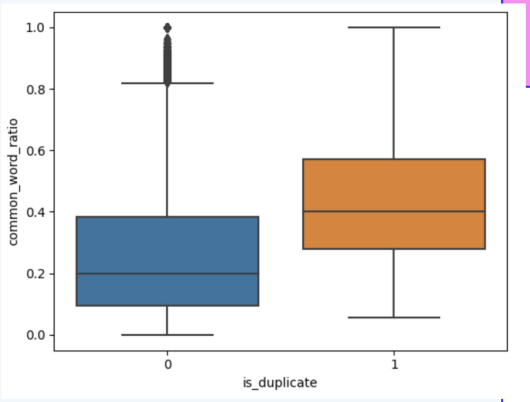
$$k(x, y) = \frac{xy^\top}{\|x\|\|y\|}$$

5. Jaccard Similarity
   - Quantifies how similar the sets are by comparing their shared elements with their total combined elements.
   - Done by tokenizing and lemmatizing the questions and filtering the stop words

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|}$$

Feature Engineering

6. Common words to unique words ratio
   - Duplicate question pairs tend to have more common words between both the question



7. Common stop words to unique words ratio (csc_ratio)
   - To see the influence of stop words on similarity of questions

Feature Engineering

Purpose: It enhances the accuracy of the data insights by overcoming complexities in data such as spelling variations, spelling mistakes, duplicate entries, abbreviations

8. Fuzzy Simple Ratio
   - Measures the similarity between two strings by calculating the minimum number of single-character edits needed to transform one string into the other.

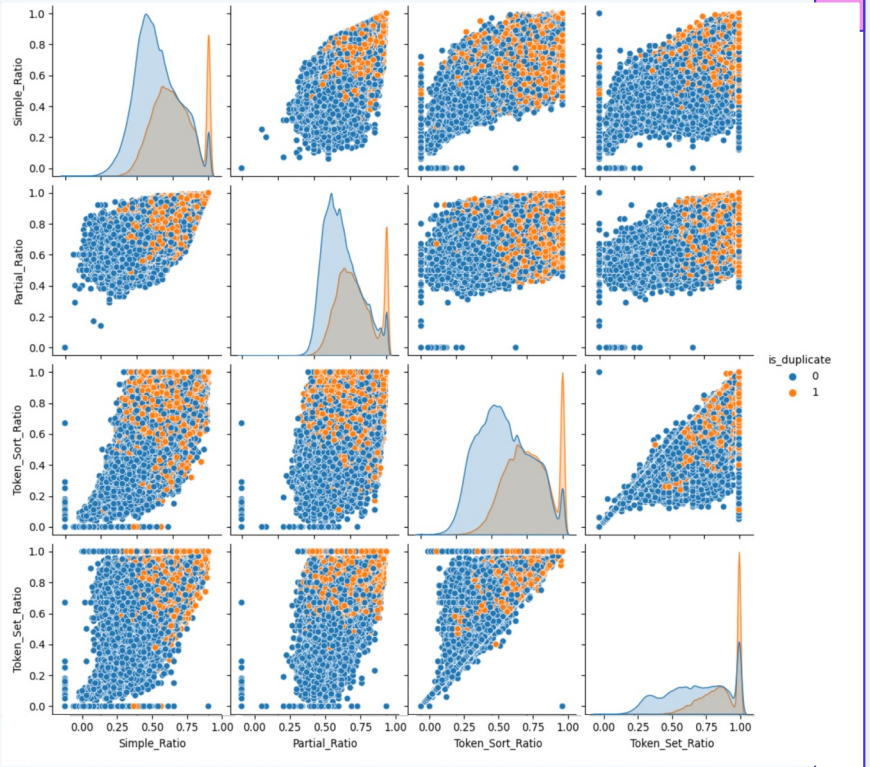
9. Fuzzy Partial Ratio
   - It is effective to identify matches when one string is a subset or prefix of the other.

10. Fuzzy Token Sort Ratio
    - It captures the essence of the strings' content rather than their specific order

Feature Engineering

11. Fuzzy Token Set Ratio
    - handles cases where word order differs but the same set of words exists in both strings

From the pair plot, we are able to differentiate between duplicate and non-duplicate where the blue coordinates are distinctly separated from the orange coordinates in the scatterplots. This implies that the plotted features are good discriminators for the categories.



Feature Engineering

TF-IDF Weighted Word-Vectors

- There is value in words present in the questions, so we use TF-IDF as a numerical way to vectorize text data
- Reflects how important a word is to a document in a collection or corpus of documents

Spacy GLOVE Model

- After finding the TF-IDF scores, we convert each question to a weighted average of word2vec vectors by these scores.

This created 300 features respectively for question 1 and question 2.

Feature Engineering

12. Semantic Cosine Similarity

13. Semantic Euclidean Distance
    - Created to relate the words vectors generated for question 1 and question 2 respectively together

14. Semantic Correlations
    - To see the correlation between the word vectors generated for question 1 and question 2 respectively

Feature Engineering Conclusion

Statistical & NLP Features
Dimension: 11

Vector Features
Dimension: 600

Semantic Relation Features
Dimension: 3

Total features = Statistical & NLP Features + Semantic Relation Features
Dimension: 14

Evaluation

How we evaluate models used

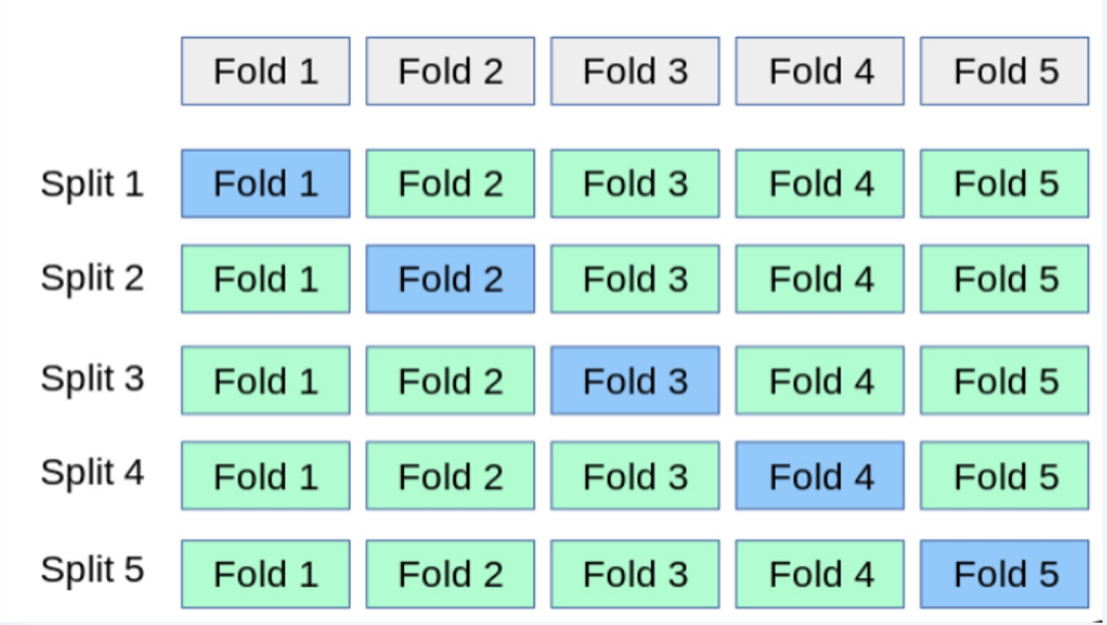
Real world distribution of data

Non-duplicate pair of questions

Duplicate pair of questions

How to evaluate the performance of our model?

Use K-fold cross validation to get evaluation scores of different metrics



Evaluation Metric - F1 Score

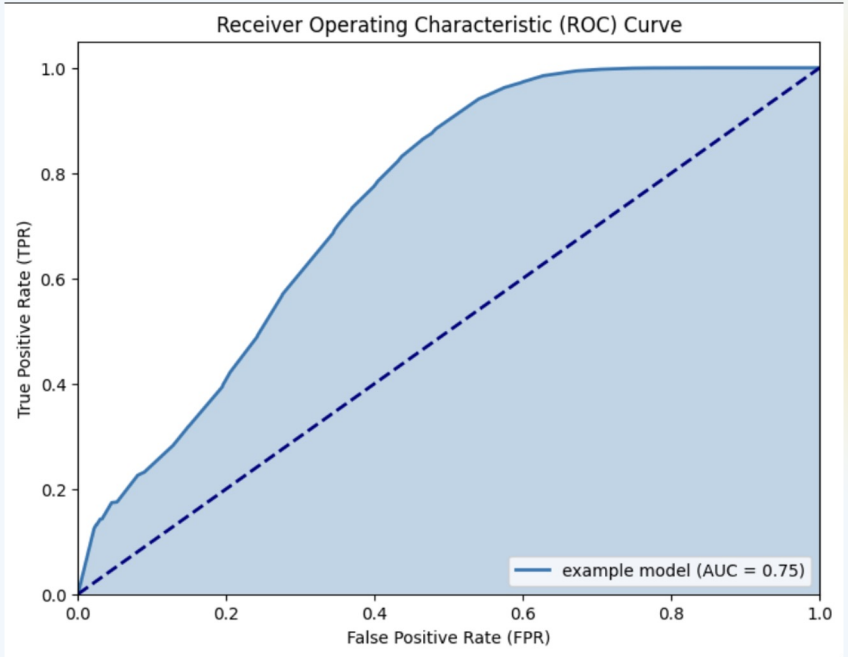
- Accounts for imbalanced dataset
- Balances between precision and recall
  - Precision ensures that the model do not excessively mark pairs of question as similar so as to prevent user from getting a notified of two unrelated questions being similar.
  - Recall ensures that the model does not miss out on potential similar questions that Quora can mark as being similar.

$$\frac{2}{\left(\frac{1}{\text{Precision}} + \frac{1}{\text{Recall}}\right)}$$

Evaluation Metric - ROC AUC

Area under curve of the Receiver Operating Characteristic

- Accounts for imbalanced dataset
- Provides a comprehensive view of the different model's ability to differentiate between similar and non-similar pairs of questions for multiple thresholds



Evaluation Metric - Log Loss

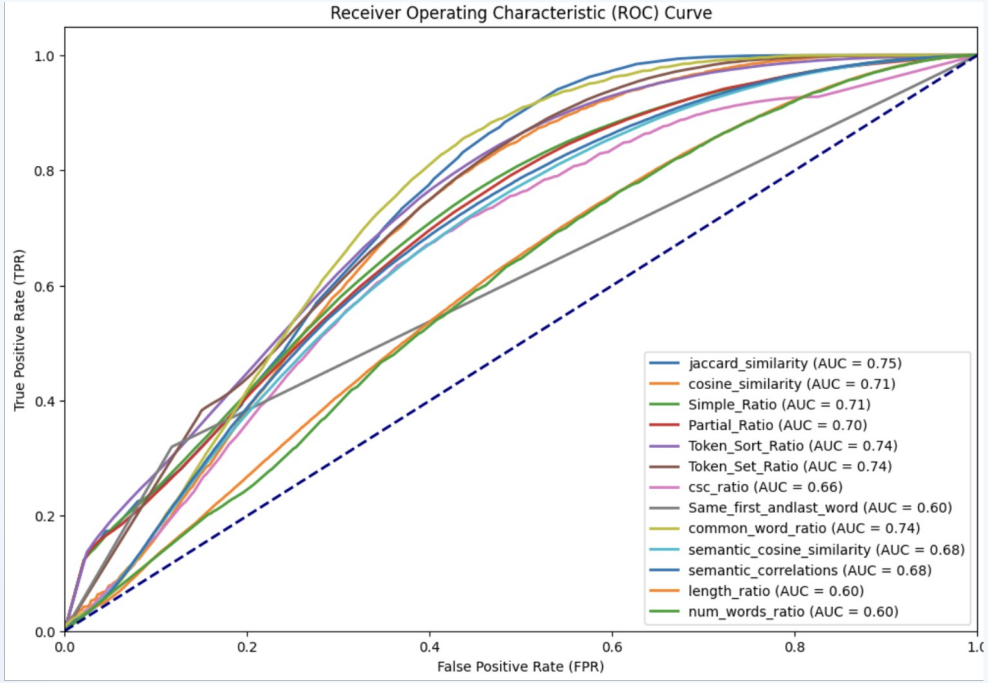
Logarithmic Loss

- Quantifies the difference between predicted probabilities and actual outcome
- Penalizes confident misclassification
- Can prevent over reliance on certain features leading to suboptimal generalization

$$-\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1 - y_i) \cdot log(1 - p(y_i))$$

Feature evaluation

ROC_AUC of some similarity features if we use the individual feature for classification



ML Algorithms

Baselines and Tuning

Our Approach to ML

Preliminary models

- Random model
- 5 ML classification models:
  - Logistic Regression
  - SVM
  - Decision Tree
  - Random Forest
  - XGBoost

Tuning for best models

- Hyperparameter tuning
- Feature selection (RFE)

ML Models

1. Baseline: Random model

Model

- Serves as a baseline to compare evaluation metrics of of ML algorithms
- Used sklearn's Dummy Classifier with the 'stratified strategy'

Evaluation

F1 score: 0.37 | ROC AUC: 0.50 | Log-Loss: **16.75**

- Poor performance expected as it is random
- Log-loss value serves as a benchmark

ML Models

1. Baseline: Random model

Model

- Serves as a baseline to compare evaluation metrics of of ML algorithms
- Used sklearn's Dummy Classifier with the 'stratified strategy'

Evaluation

F1 score: 0.37 | ROC AUC: 0.50 | Log-Loss: **16.75**

- Poor performance expected as it is random
- Log-loss value serves as a benchmark

2. Logistic regression

Model

- Regularized logistic regression model with stochastic gradient descent (SGD) learning

Evaluation

F1 score: 0.54 | ROC AUC: 0.65 | Log Loss: 0.54

- Improved log-loss, but F1 score remains low
- Reasons for poor performance:
  - correlated features
  - lack of expressivity

ML Models

3. SVM

Model

- Radial kernel approximation
- Linear Support Vector Classification

Evaluation

F1 score: 0.64 | ROC AUC: 0.71 | Log Loss: 0.48

- Improvement over logistic regression
- Data not linearly separable, so SVM is less preferred compared to tree methods

ML Models

3. SVM

Model

- Radial kernel approximation
- Linear Support Vector Classification

Evaluation

F1 score: 0.64 | ROC AUC: 0.71 | Log Loss: 0.48

- Improvement over logistic regression
- Data not linearly separable, so SVM is less preferred compared to tree methods

4. Decision Tree

Model

- A tree-like classification models by making decisions based on features' values to classify instances.

Evaluation

F1 score: 0.65 | ROC AUC: 0.72 | Log Loss: 0.47

- Improvement from logistic regression and SVM

ML Models

5. Random Forest

Model

- Trains multiple decision trees in parallel and decides final class by majority vote
- A bagging technique to decrease variance and control over-fitting

Evaluation

F1 score: 0.70 | ROC AUC: 0.76 | Log Loss: 0.43

- Improvement over a single decision tree as it uses bagging

ML Models

5. Random Forest

Model

- Trains multiple decision trees in parallel and decides final class by majority vote
- A bagging technique to decrease variance and control over-fitting

Evaluation

F1 score: 0.70 | ROC AUC: 0.76 | Log Loss: 0.43

- Improvement over a single decision tree as it uses bagging

6. XGBoost

Model

- Sequentially creates decision trees and improves on mistakes of previous trees
- A boosting technique to reduce bias and improve performance

Evaluation

F1 score: 0.71 | ROC AUC: 0.77 | Log Loss: 0.43

- Similar performance to random forest

Hyperparameter Tuning

Rationale

- Tree-based methods worked better than linear models and margin-based classifiers
- Concluded that the 2 best models were XGBoost (F1 score: 0.71, AUC_ROC: 0.77), RandomForest (F1 score: 0.70, AUC_ROC: 0.76)
- Improvement in the performance of the model using hyperparameter tuning, searching for the parameters using Halving GridSearch
- First set the gridsearch to a large range and then narrowed the range down via trial and error

Hyperparameter Tuning

Outcome

Random Forest

Evaluation

F1 score: 0.71 | ROC AUC: 0.77 | Log Loss: 0.43

- Slight Improvement in F1 score and ROC-AUC, but log loss remains the same

XGBoost

Evaluation

F1 score: 0.71 | ROC AUC: 0.77 | Log Loss: 0.43

- Slight improvement in ROC-AUC, but F1 score and Log loss remains the same

Limitations for Hyperparameter Tuning:

- Param grid used may not be large enough during GridSearch

Recursive Feature Selection

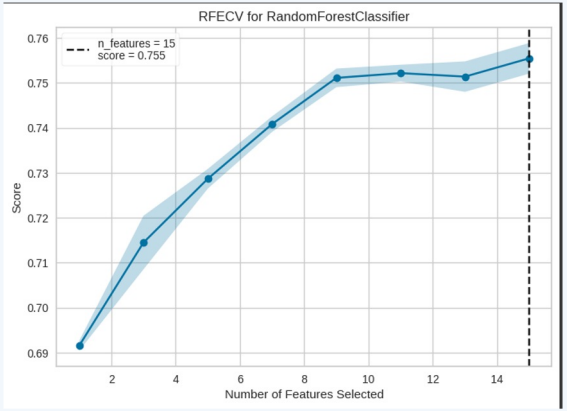
Rationale

- To find out if we can improve model performance through the dropping of less important features for each individual model
- Plotting of graph with the Number of Features Selected against Score for the model
- Function to also output a table which tells us which features are selected based on the importance value they are assigned
- Outputs a model with selected features that are deemed important for that particular model

Recursive Feature Selection

Outcome for RandomForest

- Score: 0.755
- All the features were selected for Random Forest

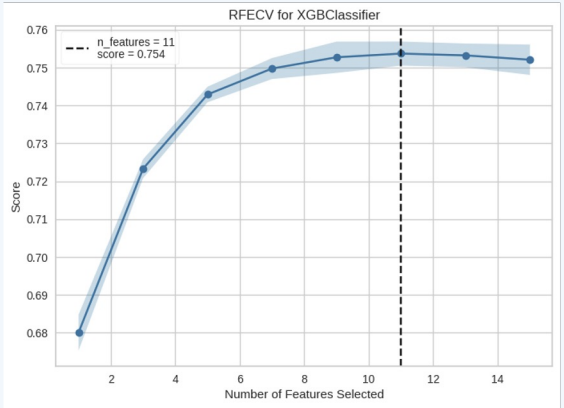



```
Feature Rankings with Names and Support:
Feature: jaccard_similarity, Rank: 1, Importance: 0.11 | Selected
Feature: cosine_similarity, Rank: 1, Importance: 0.08 | Selected
Feature: Simple_Ratio, Rank: 1, Importance: 0.05 | Selected
Feature: Partial_Ratio, Rank: 1, Importance: 0.05 | Selected
Feature: Token_Sort_Ratio, Rank: 1, Importance: 0.08 | Selected
Feature: Token_Set_Ratio, Rank: 1, Importance: 0.08 | Selected
Feature: csc_ratio, Rank: 1, Importance: 0.06 | Selected
Feature: Same_first_andlast_word, Rank: 1, Importance: 0.02 | Selected
Feature: common_word_ratio, Rank: 1, Importance: 0.17 | Selected
Feature: semantic_cosine_similarity, Rank: 1, Importance: 0.05 | Selected
Feature: semantic_euclidean_distance, Rank: 1, Importance: 0.08 | Selected
Feature: semantic_correlations, Rank: 1, Importance: 0.07 | Selected
Feature: length_ratio, Rank: 1, Importance: 0.05 | Selected
Feature: num_words_ratio, Rank: 1, Importance: 0.04 | Selected
```

Recursive Feature Selection

Outcome for XGBoost

- Score: 0.754
- 3 of the features were not selected for XGBoost after the function detected a drop or no change in the model performance
- Could be due to the fact that the 3 excluded features were highly correlated with some of the other features which led to noise in the data




```
Feature Rankings with Names and Support:
Feature: jaccard_similarity, Rank: 1, Importance: 0.25999999046325684 | Selected
Feature: cosine_similarity, Rank: 1, Importance: 0.03999999910593033 | Selected
Feature: Simple_Ratio, Rank: 2, Importance: 0.09000000357627869 | Not Selected
Feature: Partial_Ratio, Rank: 3, Importance: 0.07000000029802322 | Not Selected
Feature: Token_Sort_Ratio, Rank: 1, Importance: 0.07000000029802322 | Selected
Feature: Token_Set_Ratio, Rank: 1, Importance: 0.10000000149011612 | Selected
Feature: csc_ratio, Rank: 1, Importance: 0.27000001072883606 | Selected
Feature: Same_first_andlast_word, Rank: 1, Importance: 0.029999999329447746 | Selected
Feature: common_word_ratio, Rank: 1, Importance: 0.03999999910593033 | Selected
Feature: semantic_cosine_similarity, Rank: 3, Importance: 0.019999999552965164 | Not Selected
Feature: semantic_euclidean_distance, Rank: 1, Importance: 0.019999999552965164 | Selected
```

Conclusion