Measuring Quora Question Pairs Similarity

Akhil Gandhi
Akhil.Gandhi@colorado.edu
CSCI 5502-001
University of Colorado Boulder

Janushi Shastri
Janush.Shastri@colorado.edu
CSCI 5502-001
University of Colorado Boulder

Mohit Snehal
mosn3654@colorado.edu
CSCI 5502-001
University of Colorado Boulder

Shaily Goyal
shgo4037@colorado.edu
CSCI 5502-001
University of Colorado Boulder

1 UNDERSTANDING THE PROBLEM

1.1 Problem Context

Quora, with over 100 million monthly visitors, is a critical platform for knowledge sharing and problem-solving. However, it faces a significant challenge: question duplication. This issue arises when users ask questions that are similar in intent but differ in their wording, leading to redundancy and inefficiencies. This duplication is problematic for several reasons:

- **Impact on Question Askers:** Individuals seeking information on Quora are often confronted with multiple similar questions, complicating their search for precise and quick answers. This redundancy can overwhelm users and hinder their ability to find relevant information efficiently.
- **Burden on Answer Providers:** Experts and enthusiasts contributing answers to Quora are frequently faced with repetitive queries. This repetition dilutes the quality of discourse on the platform and can lead to reduced engagement from knowledgeable contributors.
- **Challenges for Platform Owners (Quora):** DThe presence of duplicate content can negatively impact the platform's search engine optimization (SEO) rankings and complicate the management of content. This directly affects the usability and success of Quora as a knowledge-sharing platform.

1.2 Broader Implications

The problem of question duplication extends beyond Quora, posing wider implications for information retrieval and knowledge management systems. Addressing this issue is not only about enhancing the Quora platform but also about gaining insights into the capabilities and applications of natural language processing (NLP) techniques in real-world scenarios.

1.3 Objective

The primary objective of this study is to develop a predictive model that can identify whether a pair of questions on Quora share similar intent and meaning. This model aims to assist in content consolidation and improve the overall user experience on the platform. The project will demonstrate the model's effectiveness through a dashboard, providing a practical illustration of its utility.

1.4 Significance

Understanding and addressing the problem of question duplication is crucial for streamlining content, maximizing the utility of knowledge-sharing platforms, and enhancing user experience. This study aims to provide a solution that not only improves Quora's functionality but also contributes to the broader field of text mining and NLP.

1.5 Existing Solutions and Gaps

1.5.1 "Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks" (Severyn and Moschitti, 2015)[7]. Several research papers and commercial applications focus on question similarity like this paper employs CNNs for sentence matching tasks but doesn't focus on question duplication.

1.5.2 "Siamese Recurrent Architectures for Learning Sentence Similarity" (Mueller and Thyagarajan, 2016)[5][4]. The authors of the research paper uses LSTM-based Siamese networks for sentence similarity but does not apply to large-scale platforms like Quora.

1.5.3 "Natural Language Understanding with the Quora Question Pairs Dataset" (Lakshay Sharma et al, 2019) [8]. The authors of this research paper have focused on conducted extensive exploration of the data set and used various machine learning models but lacked in advanced feature engineering like utilisation fuzzy features.

1.6 Challenges in Question Similarity

Detecting question similarity is particularly challenging due to the variability in how users phrase their questions. Two questions might have the same intent but can be phrased entirely differently. Thus, it's essential to move beyond mere syntactic matching to semantic understanding.[2]. The existing literature often either lacks focus on the specific problem of question duplication in a large-scale, real-world setting or does not address the issue of model interpretability.

This project aims to fill the gaps by focusing on question duplication on Quora, using advanced feature engineering, and maintaining a balance between model performance and interpretability.

2 DATA PREPROCESSING AND UNDERSTANDING

The quality of data preprocessing and understanding in this project is critical, as it lays the foundation for accurate and meaningful analysis. This involves a series of steps to ensure data integrity and relevance.

In our project, the dataset comprises key fields essential for analyzing question similarities. The *id* field serves as the unique identifier for each question pair in the training set. Each question in a pair is distinctively identified by *qid1* and *qid2*, which are unique identifiers available in the training dataset. The main components of our analysis are *question1* and *question2*, containing the full texts of the questions. The crux of our objective is encapsulated in the *is_duplicate* field. This target variable is labeled as *1* when question1 and question2 have essentially the same meaning, and *0* otherwise. Effectively understanding and preprocessing these fields are crucial, as they are integral to the accuracy and success of our analysis in modeling and predicting question similarities.

2.1 Exploratory Data Analysis (EDA)

- **Data Source and Loading:** The dataset, obtained from Kaggle, consists of approximately 400,000 rows and 6 columns. It was loaded into memory using data manipulation libraries such as pandas, a standard practice ensuring efficient handling of large datasets.
- **Initial Data Assessment:** The preliminary analysis was conducted to understand the dataset's size, structure, and the data types present. This step is crucial for identifying any immediate data cleaning and preprocessing needs.
- **Statistical Analysis:** Basic statistical analyses were employed to gain insights into the dataset. This includes understanding data distribution, identifying any anomalies or outliers, and assessing the balance of the dataset.
- **Data Distribution and Imbalances:** Special attention was given to the distribution of data points among various output classes. Identifying imbalances is essential as it can significantly impact the performance and generalization ability of machine learning models.
- **Duplicate Entries Identification and Handling:** Duplicates can skew the analysis and lead to biased results. Detecting and addressing these ensured the uniqueness and quality of each data point.
- **Null Value Treatment:** Checking for and handling NULL or missing values is critical for maintaining data completeness and consistency. This step prevents errors in analysis and modeling, which can arise from incomplete data.

This completion of milestones in Exploratory Data Analysis signifies the foundational groundwork laid for the project. Each completed step has contributed significantly to data preprocessing and quality assurance, ensuring that the dataset is cleaned, standardized, and suitable for various natural language processing (NLP) tasks, including feature extraction, text classification, and similarity analysis.

2.2 Text Preprocessing

The project has effectively accomplished crucial steps in text preprocessing, a fundamental phase that prepares textual data for analysis and modeling within the field of natural language processing. The key tasks accomplished include:

- **Removal of HTML Tags and Punctuation Marks**: This step involved eliminating HTML tags to remove any artifacts that could interfere with textual analysis. Similarly, punctuation marks were removed to simplify the text and enhance its quality for NLP tasks.
- **Stemming**: A text normalization technique applied to reduce words to their base or root form, thereby simplifying word variations and reducing the dimensionality of the text data.
- **Stopwords Removal**: Commonly found words such as "and," "the," "in," and "is," which carry minimal semantic meaning, were removed to decrease noise and focus on more meaningful text content.
- **Expansion of Contractions**: Contractions like "don't" were expanded to their full forms (e.g., "do not") to maintain consistency in text data and ensure accuracy in subsequent analysis.

The successful completion of these preprocessing tasks represents a significant achievement in preparing the textual data. It lays a solid foundation for more accurate and insightful analysis and modeling in the field of natural language processing.[1]

3 METHODOLOGY

3.1 Basic Feature Engineering

The development of fundamental features within the dataset forms a key milestone in the project, focusing on revealing the intrinsic characteristics of the questions. This step is vital before advancing to more complex processing or data cleaning. The features crafted include:

- **"word_Common"**: This feature quantifies the count of shared unique words between two questions, such as Question 1 and Question 2, shedding light on the extent of vocabulary overlap between them.
- **"word_Total"**: Established to calculate the total word count across both questions, it provides a measure of the overall length of the question pair.
- **"word_share"**: This is a ratio derived from "word_Common" to "word_Total," offering an insight into the similarity between the questions based on their total word count.

The significance of these features lies in their pivotal role in deepening the understanding of the dataset's characteristics. They are instrumental in guiding subsequent analysis and in the development of machine learning models. An evaluation of these features' distributions for both similar and dissimilar questions has been undertaken to identify any potential issues such as overfitting or data overlap. This assessment is crucial for ensuring the relevance and balance of the features, which in turn facilitates the model-building process.

The completion of this phase represents a significant step in laying down the foundational groundwork for the project. It underscores the commitment to ensuring the quality and relevance of the features to be employed in further analyses and model development stages.

3.2 Advanced Feature Extraction

The project has accomplished significant strides in advanced feature engineering for text data. This phase involved implementing a range of techniques to enhance the representation of text data, crucial for

in-depth natural language processing analysis. The key techniques employed include:

- **Tokenization**: This process involved breaking text into smaller units or tokens, allowing for a more detailed and granular analysis of the textual data.
- **Stop Word Removal**: By eliminating commonly used words that add little semantic value, we reduced noise in the dataset, thereby sharpening the focus on more meaningful text.
- **Word Selection**: Careful selection of words helped in refining the dataset, enabling the extraction of more informative features that capture the essence of the textual content.
- **Creation of Advanced Columns**: These columns were designed to encapsulate the subtleties and nuances within the text, aiding in tasks like capturing semantic nuances and quantifying similarities and differences in text.

A notable achievement in this stage was the utilization of TF-IDF (Term Frequency-Inverse Document Frequency) weighted word vectors. This technique is paramount in highlighting the importance of words within a document in relation to a broader corpus, effectively transforming documents into numerical vectors. Such a method proves invaluable in tasks like text classification, information retrieval, and recommendation systems. The integration of tokenization, stop word removal, and selective word usage, combined with the application of TF-IDF, has significantly augmented our ability to extract meaningful insights from text data, thus enhancing the overall capabilities in natural language processing and machine learning applications.[1]

3.3 Empowering Text Analysis with NLP and Word Embeddings

In our NLP approach, we leverage TF-IDF scores to transform individual questions into weighted averages of word2vec vectors. To accomplish this, we employ a pre-trained GLOVE model available through "Spacy." This model, trained on Wikipedia data, provides robust word semantics. Utilizing word embeddings like Word2Vec or GloVe serves to encapsulate the semantic essence of words within the context of the document.

By incorporating these techniques, we aim to capture the nuanced relationships between words, which is crucial for tasks involving natural language understanding, such as question similarity analysis or text classification. These methodologies help us in quantifying semantic similarity and extracting meaningful features from textual data, ultimately contributing to the effectiveness of machine learning models in comprehending and processing natural language.[6]

3.4 Machine Learning Models

The outlined process encapsulates the comprehensive journey of machine learning workflows. It begins with the fundamental step of reading data from files and storing it into SQL tables, a crucial stage in data acquisition and preparation. Following this, a 70:30 random train-test split is conducted, allowing the dataset to be divided for training and validation purposes.

Subsequently, various machine learning algorithms, including,

- Logistic Regression: Linear model for classification and probability estimation.
- Linear Support Vector Machines (SVM): Separates data with a hyperplane to classify or regress.
- Random Forest: Ensemble learning method using decision trees for robust predictions.
- XGBoost: Gradient boosting algorithm known for high performance in structured data tasks.

All of these models undergo hyperparameter tuning to optimize their performance. Additionally, the integration of Term Frequency-Inverse Document Frequency (Tf-Idf) features across multiple models aims to enhance their predictive capabilities.

The model evaluation phase constitutes a significant portion of the process. It involves training diverse models with distinct feature combinations and evaluating their performance. Log loss, a metric measuring the performance of classification models, serves as the benchmark for performance evaluation. From a basic random model that yielded a log loss of 0.4143914, various models were trained with different feature sets, with log loss values obtained for each configuration. Notably, the XGBoost model, employing a combination of basic features (BF), advanced features (AF), distance features (DF), and average Word2Vec (AVG-W2V), exhibited the most promising performance with a log loss of 0.313341, indicating its superior predictive accuracy in this context. Despite constraints limiting the use of the entire dataset due to PC resource limitations, the iterative process of feature engineering and hyperparameter tuning yielded valuable insights into model performance and capabilities.[6]

4 EVALUATION

In this section, we delve into the evaluation of our binary classification model's performance for Quora question similarity. Evaluating the model's effectiveness is crucial to understand how well it accomplishes its task of distinguishing between similar and dissimilar question pairs.

To gauge the quality of our predictions and the alignment between our model's outputs and the ground truth, we will employ a set of well-established performance metrics. Two key metrics we will be focusing on are Log Loss and Binary Classification Metrics.
Through these performance metrics, we aim to assess the strengths and weaknesses of our binary classification model in the context of Quora question similarity, ultimately providing valuable insights into its real-world applicability and performance.

4.1 Log Loss

Logarithmic Loss, commonly referred to as Log Loss or Cross-Entropy Loss, serves as a fundamental metric for assessing the performance of probabilistic models, especially in classification tasks.
In our study of Quora question similarity, Log Loss will provide a quantitative measure of the dissimilarity between predicted probabilities and actual binary labels (similar or dissimilar) for question pairs. Lower Log Loss values indicate better model performance, with zero being the ideal score, representing perfect alignment between predicted probabilities and true labels. It is particularly

useful in scenarios where the model outputs probability scores, such as when utilizing machine learning models to predict question pair similarity. In our experiments, we will report the Log Loss to evaluate the calibration and accuracy of our probabilistic models.

$$\text{Log Loss} = -\frac{1}{N}\sum_{i=1}^{N}\left(y_i \log(p_i) + (1-y_i)\log(1-p_i)\right) \quad (1)$$

The aforementioned formula calculates the loss or error in a classification problem where N represents the total number of data points or instances. Short summary of the variables of the above equation are as follows:

- y_i is the true binary label for the i-th data point, where y_i is 1 if the instance belongs to the positive class (e.g., similar questions) and 0 if it belongs to the negative class (e.g., dissimilar questions).
- p_i represents the predicted probability that the i-th instance belongs to the positive class. This probability is typically generated by a machine learning model.
- The formula calculates the Log Loss for each data point (i) and then takes the average over all data points.

4.2 Binary Confusion Matrix

Binary Confusion Matrix is a valuable tool for understanding how our binary classification model makes predictions. It categorizes the model's predictions into four categories:

- **True Positives (TP)**: The model correctly identifies similar question pairs.
- **True Negatives (TN)**: The model correctly identifies dissimilar question pairs.
- **False Positives (FP)**: The model incorrectly predicts similarity when the questions are dissimilar.
- **False Negatives (FN)**: The model incorrectly predicts dissimilarity when the questions are similar.

From the Confusion Matrix, we derive the following essential metrics:

4.2.1 Precision. Precision measures the model's ability to correctly identify similar question pairs without falsely labeling dissimilar ones. It is calculated as:

$$Precision = \frac{TP}{TP + FP}$$

4.2.2 Recall. Recall, also known as Sensitivity, measures the model's ability to identify all similar question pairs. It is calculated as:

$$Recall = \frac{TP}{TP + FN}$$

These metrics will help us in assessing different aspects of our model's performance. Precision will tell us how many of the predicted similar pairs are actually similar, while Recall will inform us about the model's capability to find all the similar pairs within the dataset.

4.3 Visualization & Interpretations

(1) **Word Share Violin Plot and Distribution:**

Description: Two subplots showing the distribution of "word_share" for duplicate and non-duplicate samples using a violin plot and Kernel Density Estimation.

Interpretation: Observes the variation in "word_share" between duplicate and non-duplicate samples.

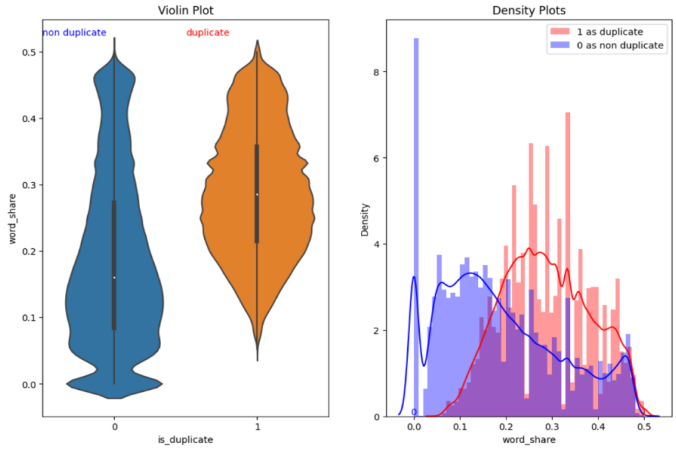

Figure 1: Violin Plot Density Plot

(2) **2D Scatter Plot with t-SNE Embedding:**

Description: A 2D scatter plot using t-SNE to visualize data in a reduced dimension.

Interpretation: Reveals the spatial arrangement of data points in two dimensions, emphasizing local relationships.

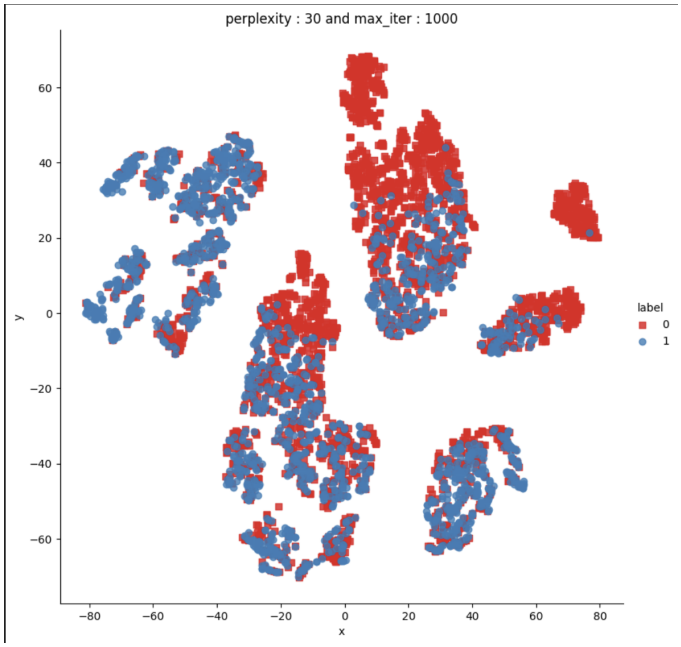

Figure 2: 2D Scatter Plot with t-SNE (0- non duplicate, 1-duplicate)

(3) **3D Scatter Plot using Matplotlib:**
Description: A 3D scatter plot with data points colored by their labels.
Interpretation: Extends the understanding of data distribution into the third dimension, aiding in identifying patterns and clusters.

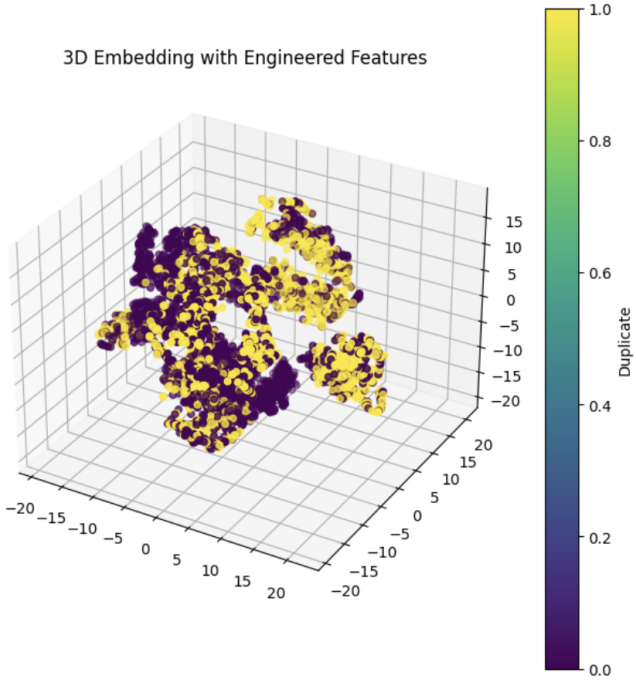

Figure 3: 3D Scatter Plot

(4) **3D Plot using Seaborn Pairplot:**
Description: A Seaborn pair plot representing scatter plots for three dimensions of t-SNE embedding.
Interpretation: Explores relationships between different dimensions in the 3D space, providing insights into potential clusters.

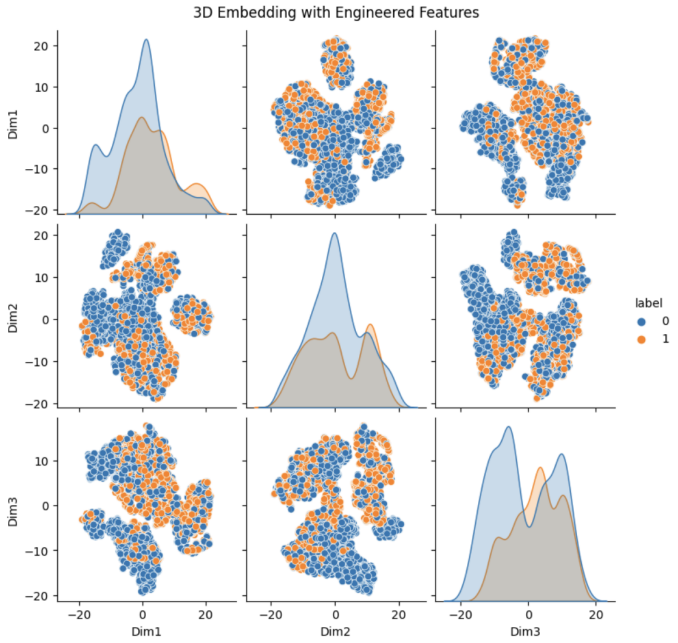

Figure 4: Seaborn Pairplot(0- non duplicate, 1-duplicate)

5 CHALLENGES & RESOLUTIONS

Throughout the project, we faced several challenges, each requiring strategic solutions to ensure the success of our efforts. The following are key challenges we encountered, along with detailed explanations of our approaches:

5.1 Data Preprocessing and Cleaning

- **Challenge:** The initial dataset exhibited variations in formatting, language, and quality due to questions from diverse sources.
- **Resolution:** We designed a comprehensive preprocessing pipeline, incorporating techniques such as tokenization, stemming, and stop-word removal. Regular expressions were employed to address formatting inconsistencies, and spell-check algorithms were utilized to rectify common spelling errors. This rigorous preprocessing ensured the uniformity and quality of the dataset.

5.2 Choosing the Right Embeddings

- **Challenge:** With a myriad of embedding techniques available, selecting the most suitable one for our dataset and the problem posed a significant challenge.
- **Resolution:** We conducted an extensive evaluation of simpler embeddings like Word2Vec and GloVe. Through iterative experimentation on a subset of our data, we identified the embeddings that most effectively captured semantic similarities for our specific use case. This meticulous approach ensured the adoption of embeddings aligned with our project goals.

5.3 Handling Imbalanced Data

- **Challenge:** The dataset displayed a notable imbalance, particularly with dissimilar questions, impacting the training of our models.
- **Resolution:** To counter this, we implemented data augmentation techniques for the minority class and adopted a stratified sampling approach. Additionally, we explored and experimented with loss functions tailored to address challenges associated with imbalanced datasets. These measures improved the robustness and generalizability of our models.

5.4 Feature Selection for Similarity Prediction

- **Challenge:** Selecting the right features for predicting question similarity proved challenging, especially with overlapping distributions and high word similarity between questions.
- **Resolution:** Through meticulous analysis, we identified 'word_common' and 'word_share' as clearer predictors, while other features exhibited poor separation and substantial overlap. This emphasized the critical role of selecting appropriate features for accurate models in this context, guiding our subsequent feature engineering endeavors.

5.5 Challenges in Advanced Feature Extraction

- **Challenge:** Fuzzy algorithms lacked complete semantic representation, and token-based features faced issues with varied sentence structures.
- **Resolution:** We enhanced fuzzy algorithms by integrating advanced techniques, such as machine learning, to improve semantic understanding. Additionally, we incorporated advanced text normalization and feature engineering to enhance the accuracy of feature extraction, contributing to improved question similarity prediction.

These challenges, though formidable, provided valuable learning experiences and insights that influenced the refinement of our methodologies and approaches for effective question similarity prediction. [3]

6 RESULT AND ANALYSIS

6.1 Initial Model Assessment

The initial step involved training a random model to establish a baseline log loss, resulting in a value of 0.887699. This worst-case log loss serves as a reference point for evaluating the subsequent models.

6.2 Model Training and Hyperparameter Tuning

Due to RAM constraints, model training was performed on a sampled subset of the total data. Utilizing both Random and Grid search techniques, hyperparameters were tuned to enhance the performance of each model.

6.3 Model Performance Overview

The log loss scores for different machine learning models, considering various feature combinations, are as follows:

Model	Features Used	Log Loss
Logistic Regression	BF + AF	0.4003415
Linear SVM	BF + AF	0.4036245
Random Forest	BF + AF	0.4143914
XGBoost	BF + AF	0.362546
Logistic Regression	BF + AF + Tf-Idf	0.358445
Linear SVM	BF + AF + Tf-Idf	0.362049
Logistic Regression	BF + AF + DF + AVG-W2V	0.3882535
Linear SVM	BF + AF + DF + AVG-W2V	0.394458
XGBoost	BF + AF + DF + AVG-W2V	0.313341

Fig: Machine Learning Models

6.3.1 Logistic Regression and Linear SVM (BF + AF).

- Logistic Regression: 0.4003415
- Linear SVM: 0.4036245

6.3.2 Random Forest and XGBoost (BF + AF).

- Random Forest: 0.4143914
- XGBoost: 0.362546

6.3.3 Logistic Regression and Linear SVM (BF + AF + Tf-Idf).

- Logistic Regression: 0.358445
- Linear SVM: 0.362049

6.3.4 Logistic Regression, Linear SVM, and XGBoost (BF + AF + DF + AVG-W2V).

- Logistic Regression: 0.3882535
- Linear SVM: 0.394458
- XGBoost: 0.313341

6.4 Observations and Considerations

- The log loss scores serve as key indicators of predictive performance, where lower values suggest better accuracy.
- Models incorporating advanced features (AF) consistently outperform the baseline model, emphasizing the significance of feature engineering.
- The inclusion of Tf-Idf features further refines the performance of Logistic Regression and Linear SVM, indicating the importance of text representation.
- Integrating distance features (DF) and average Word2Vec (AVG-W2V) in XGBoost leads to the lowest log loss, demonstrating superior predictive accuracy and the effectiveness of ensemble methods.

6.5 Future Considerations

Despite the constraints, the iterative process of feature engineering and hyperparameter tuning provided valuable insights into model performance and capabilities. Future considerations may involve:

- Leveraging advanced methodologies, such as BERT embeddings, to capture intricate semantic nuances, contingent upon the scale and nature of the available data.
- Exploring larger datasets to potentially overcome RAM limitations and improve the scalability of the models.
- Conducting more extensive hyperparameter tuning for further optimization.

REFERENCES

[1] Ansari, N., and Sharma, R. Identifying semantically duplicate questions using data science approach: A quora case study, 2020.

[2] Duan, N., Tang, D., Chen, P., and Zhou, M. Mining similar questions with paraphrasing from community-based qa archives. In *Proceedings of the AAAI Conference on Artificial Intelligence* (2012), vol. 26.

[3] Gontumukkala, S. S. T., Godavarthi, Y. S. V., Gonugunta, B. R. R. T., Gupta, D., and Palaniswamy, S. Quora question pairs identification and insincere questions classification. 1–6.

[4] Mueller, J., and Thyagarajan, A. Siamese recurrent architectures for learning sentence similarity. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence* (2016), AAAI'16, AAAI Press, p. 2786–2792.

[5] Palangi, H., Deng, L., Shen, Y., Gao, J., He, X., Chen, J., Song, X., and Ward, R. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech, and Language Processing 24*, 4 (2016), 694–707.

[6] Sakata, W., Shibata, T., Tanaka, R., and Kurohashi, S. Faq retrieval using query-question similarity and bert-based query-answer relevance, 2019.

[7] Severyn, A., and Moschitti, A. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval* (New York, NY, USA, 2015), SIGIR '15, Association for Computing Machinery, p. 373–382.

[8] Sharma, L., Graesser, L., Nangia, N., and Evci, U. Natural language understanding with the quora question pairs dataset, 2019.