Detecting Duplicate Question Pairs Using Machine Learning

Shweta Shaw

05/11/2024

Abstract

This project leverages machine learning to identify duplicate questions within a dataset of paired questions, focusing on detecting pairs that express the same meaning despite slight wording variations. By creating a web application capable of recognizing such duplicates, this project showcases the use of machine learning for efficient content organization. This application is designed to improve content management processes, optimize search accuracy, and enhance user experience by grouping similar questions. It is particularly valuable for platforms that rely on effective question categorization to streamline user interactions and ensure relevant search results.

1. Problem Statement:

The project utilizes machine learning to address the issue of redundant questions on digital Q&A platforms. Users often submit questions that are worded differently but convey the same intent, leading to fragmented answers and a cluttered user experience. This project aims to develop a machine learning model capable of identifying pairs of questions that share similar meanings despite minor wording differences.

The model will be trained on a dataset of paired questions, learning to recognize subtle linguistic variations and determine whether questions are essentially duplicates. By detecting these duplicates, it can group similar questions, thereby reducing redundancy and enhancing information retrieval efficiency. This approach is particularly beneficial for content-heavy platforms, where organized content is vital for user engagement. Ultimately, this project demonstrates the way machine learning can effectively streamline content organization, improve search relevance, and create a more intuitive user experience by consolidating related questions into cohesive groups.

2. Market and Customer Needs Assessment

2.1 Market Analysis

Industry Overview

The digital question-and-answer (Q&A) platform market has experienced significant growth due to the increasing reliance on online information and user-generated content. Platforms like Quora, Stack Overflow, and Reddit facilitate user interactions by allowing individuals to ask and answer questions. However, these platforms often face challenges related to content redundancy, where multiple questions convey similar meanings, leading to inefficient information retrieval and a fragmented user experience.

Target Market

The primary target market for the *Duplicate Question Pairs* project includes digital Q&A platforms, forums, and knowledge-sharing websites that handle large volumes of user inquiries. This market encompasses various sectors, such as education, technology, healthcare, and customer service, where users frequently seek information and answers to common queries.

Competitive Landscape

The market currently features several players focused on improving search functionality and user experience on Q&A platforms. However, many existing solutions may not effectively address the nuances of language, leading to missed opportunities for content consolidation. By leveraging advanced machine learning techniques to identify duplicate questions, this project offers a distinct advantage over conventional keyword-based search methods.

Value Proposition

This project provides significant value by enhancing content organization and improving search accuracy. By efficiently grouping similar questions, the model can streamline content management processes, reduce user frustration caused by duplicate inquiries, and foster a more cohesive knowledge-sharing environment. This application of machine learning not only enhances user experience but also supports platforms in optimizing their content workflows and improving engagement metrics.

Market Potential

With the increasing demand for high-quality, organized content on digital platforms, the market potential for the *Duplicate Question Pairs* project is substantial. As organizations prioritize user experience and efficient content management, the implementation of advanced machine learning models to identify and group duplicate questions presents a timely and relevant solution. The project aligns with broader trends in AI and machine learning, positioning it favourably for adoption by various stakeholders in the digital content space.

2.2 Customer Segmentation

Educational Institutions and E-Learning Platforms
Online learning platforms like Coursera, Khan Academy, and educational institutions offering distance learning can benefit from this project by organizing frequently asked questions, enhancing course-related discussions, and streamlining student interactions. This application can help in consolidating queries related to course materials, thus providing clearer and more organized information.

Customer Support and Helpdesk Services
Businesses that operate customer support systems and helpdesk platforms can utilize this project to manage user inquiries effectively. By grouping similar questions, these organizations can reduce response times and improve customer satisfaction by directing users to existing answers rather than handling duplicate queries.

Community Forums and User Groups
Online forums and communities, such as Reddit and niche interest groups, often have users posting similar questions. This project can assist in better content organization, ensuring that related questions are grouped together, fostering a more cohesive discussion and enhancing user engagement.

3. Target Specification

3.1. Model Accuracy

Achieve at least 90% accuracy in identifying duplicate question pairs to ensure reliability in grouping similar questions

3.2. Precision and Recall Optimization

Attain a balance between high precision (minimizing false positives) and high recall (minimizing false negatives) with a target F1-score of 0.85 or higher. It is essential for maintaining content relevance and search accuracy.

3.3. Scalability

Support datasets with more than thousands of question pairs with efficient processing time.

3.4. Response Time

Provides real-time duplicate detection with a maximum response time of 2 seconds per question pair.

3.5. User Experience Enhancement

Improves search relevance by at least 20%, measured through user engagement metrics like reduced bounce rates and increased user retention.

4. External Search

To ensure the effectiveness and relevance of this project, examining the landscape of similar solutions and technologies can highlight existing standards, potential gaps, and areas for innovation. Here’s an overview of related tools, technologies, and industry practices that influence and shape the development of this project.

Existing Duplicate Detection Solutions

Q&A Platforms: Major Q&A platforms like Quora, Stack Overflow, and Reddit already utilize some form of duplicate question detection, but the accuracy and response times vary significantly across platforms. These platforms typically use rule-based filtering, keyword matching, or basic NLP algorithms to detect redundancies. However, the accuracy of such approaches is limited when it comes to questions with nuanced phrasing.

Natural Language Processing (NLP) Advances

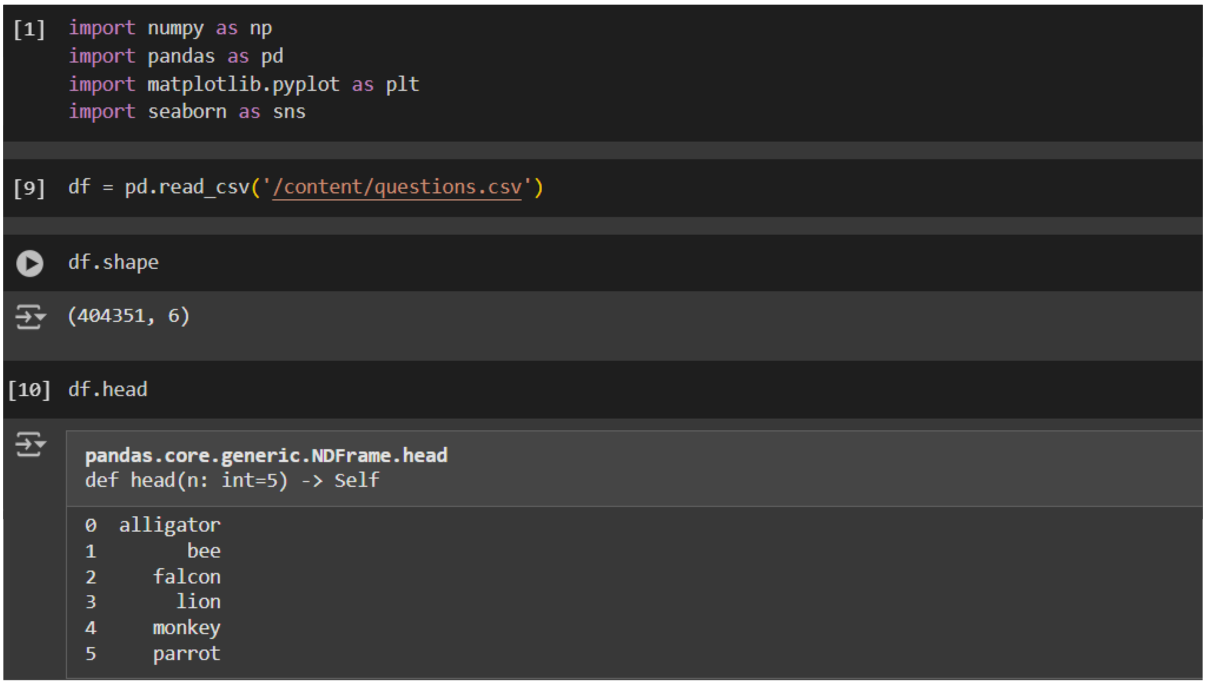
Text Embeddings: The use of word embeddings, sentence embeddings, and transformer-based embeddings has revolutionized the way machines understand textual data. Embeddings like Sentence-BERT are designed to capture semantic relationships between sentence pairs, making them ideal for projects focused on similarity detection, such as Duplicate Question Pairs.

Current Trends in Content Organization

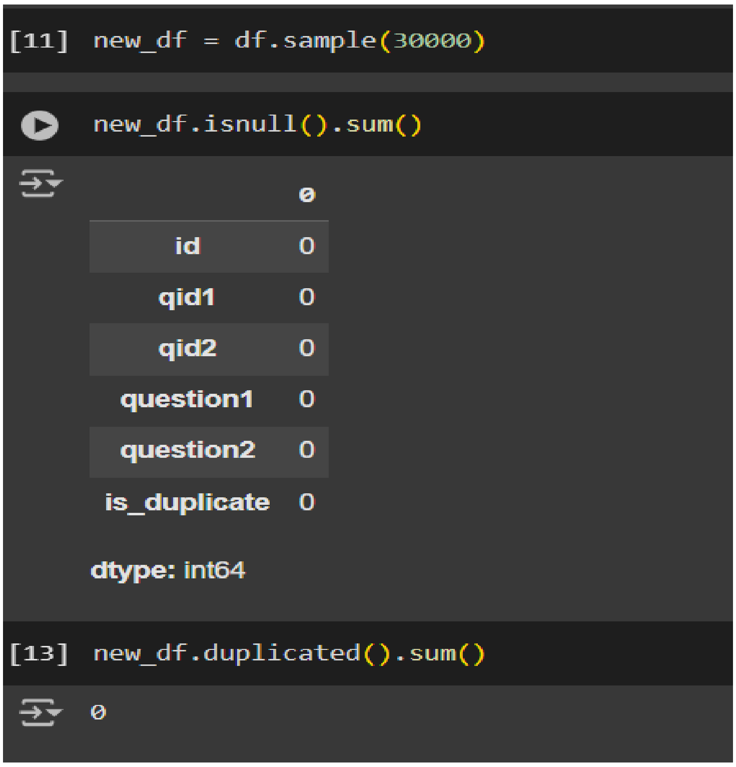
- **Personalized Search and Content Recommendations**: With platforms increasingly focusing on personalized user experiences, there is a growing demand for intelligent organization of content based on relevance, user behaviour, and preferences. Identifying duplicate questions is part of this effort, as it improves the relevance of search results and organizes content more intuitively.
- **Automated Moderation Tools**: Content-heavy platforms are looking to automate content moderation, and duplicate detection plays a crucial role in this. Flagging redundant questions allows moderators to focus on quality control, optimizing platform resources. Efficient duplicate detection tools can provide both automated solutions and augmented moderation for larger platforms.

5. Dataset Description:

5.1: Importing the libraries:

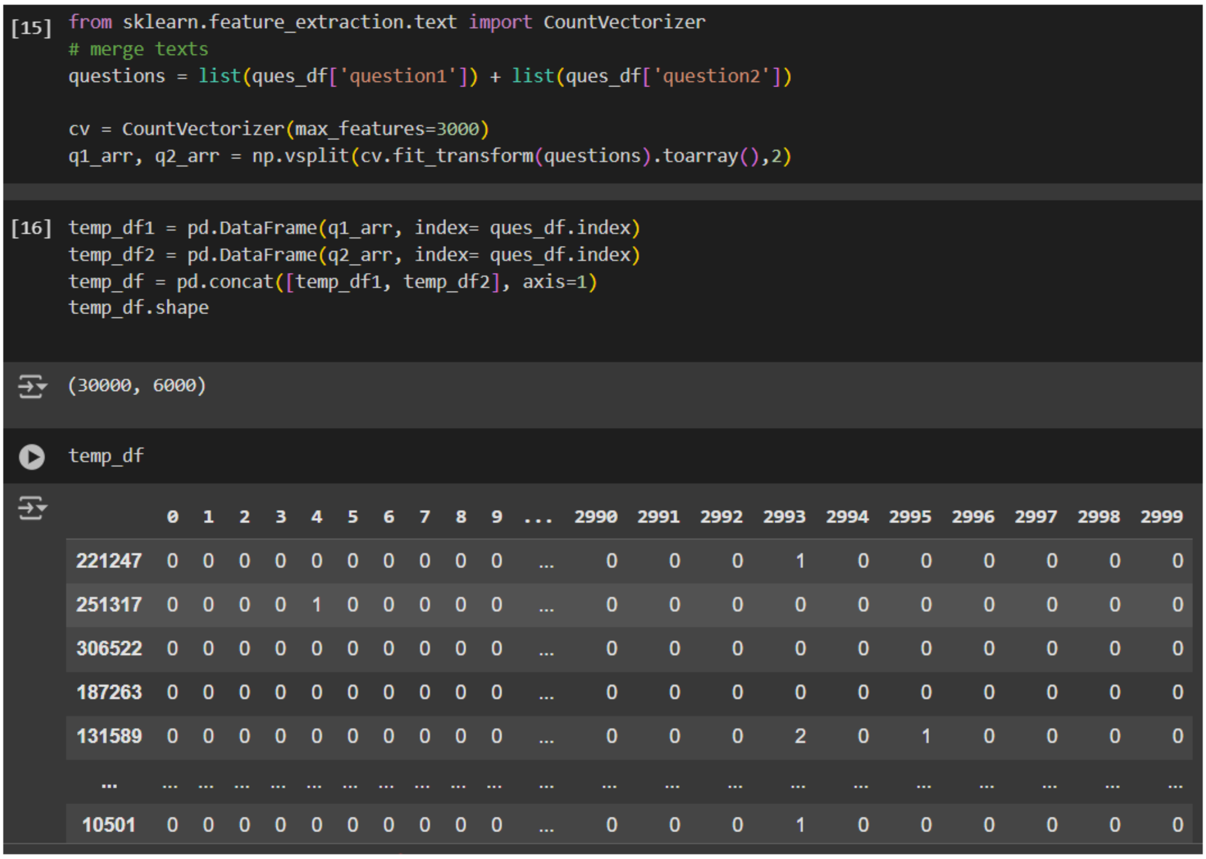


```
[1]  import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
     import seaborn as sns

[9]  df = pd.read_csv('/content/questions.csv')

     df.shape

     (404351, 6)

[10] df.head

     pandas.core.generic.NDFrame.head
     def head(n: int=5) -> Self

     0  alligator
     1        bee
     2     falcon
     3       lion
     4     monkey
     5     parrot
```



```
[11] new_df = df.sample(30000)

     new_df.isnull().sum()

                      0
             id       0
           qid1       0
           qid2       0
      question1       0
      question2       0
     is_duplicate     0

     dtype: int64

[13] new_df.duplicated().sum()

     0
```

6. Data Preprocessing:



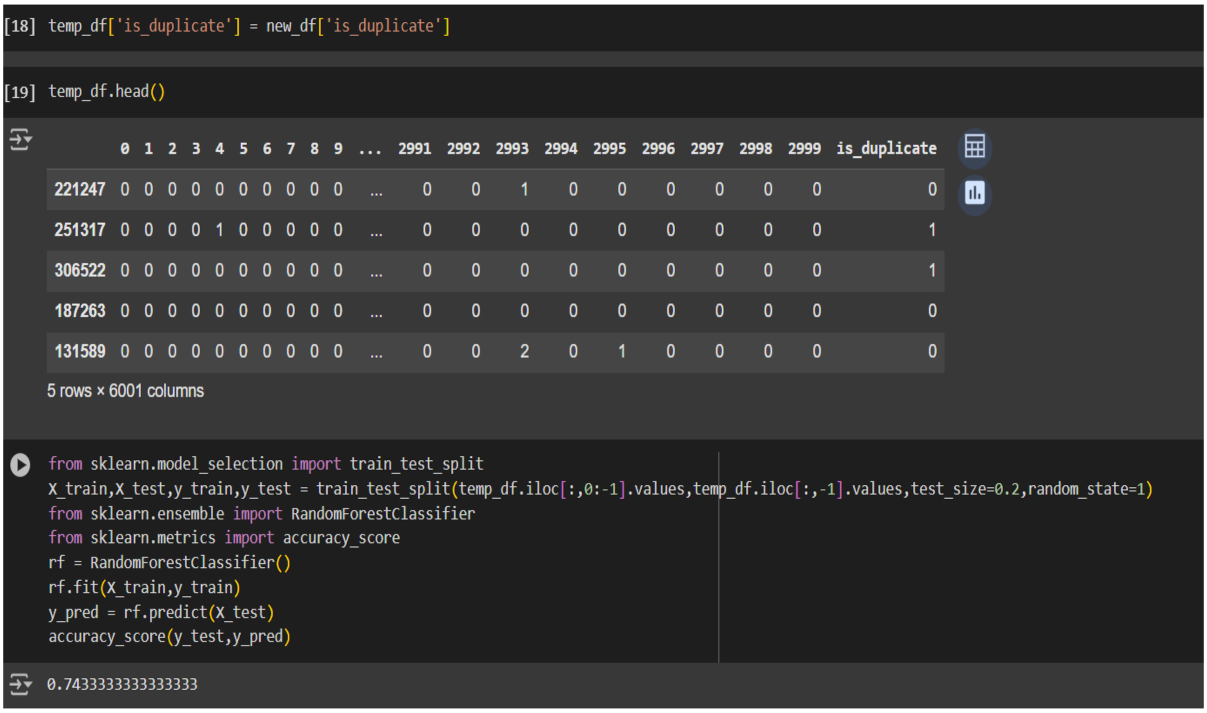
```
[15] from sklearn.feature_extraction.text import CountVectorizer
     # merge texts
     questions = list(ques_df['question1']) + list(ques_df['question2'])

     cv = CountVectorizer(max_features=3000)
     q1_arr, q2_arr = np.vsplit(cv.fit_transform(questions).toarray(),2)
```

```
[16] temp_df1 = pd.DataFrame(q1_arr, index= ques_df.index)
     temp_df2 = pd.DataFrame(q2_arr, index= ques_df.index)
     temp_df = pd.concat([temp_df1, temp_df2], axis=1)
     temp_df.shape
```

```
(30000, 6000)
```

```
temp_df
```

	0	1	2	3	4	5	6	7	8	9	...	2990	2991	2992	2993	2994	2995	2996	2997	2998	2999
221247	0	0	0	0	0	0	0	0	0	0	...	0	0	0	1	0	0	0	0	0	0
251317	0	0	0	0	1	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
306522	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
187263	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
131589	0	0	0	0	0	0	0	0	0	0	...	0	0	0	2	0	1	0	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
10501	0	0	0	0	0	0	0	0	0	0	...	0	0	0	1	0	0	0	0	0	0



```
[18] temp_df['is_duplicate'] = new_df['is_duplicate']
```

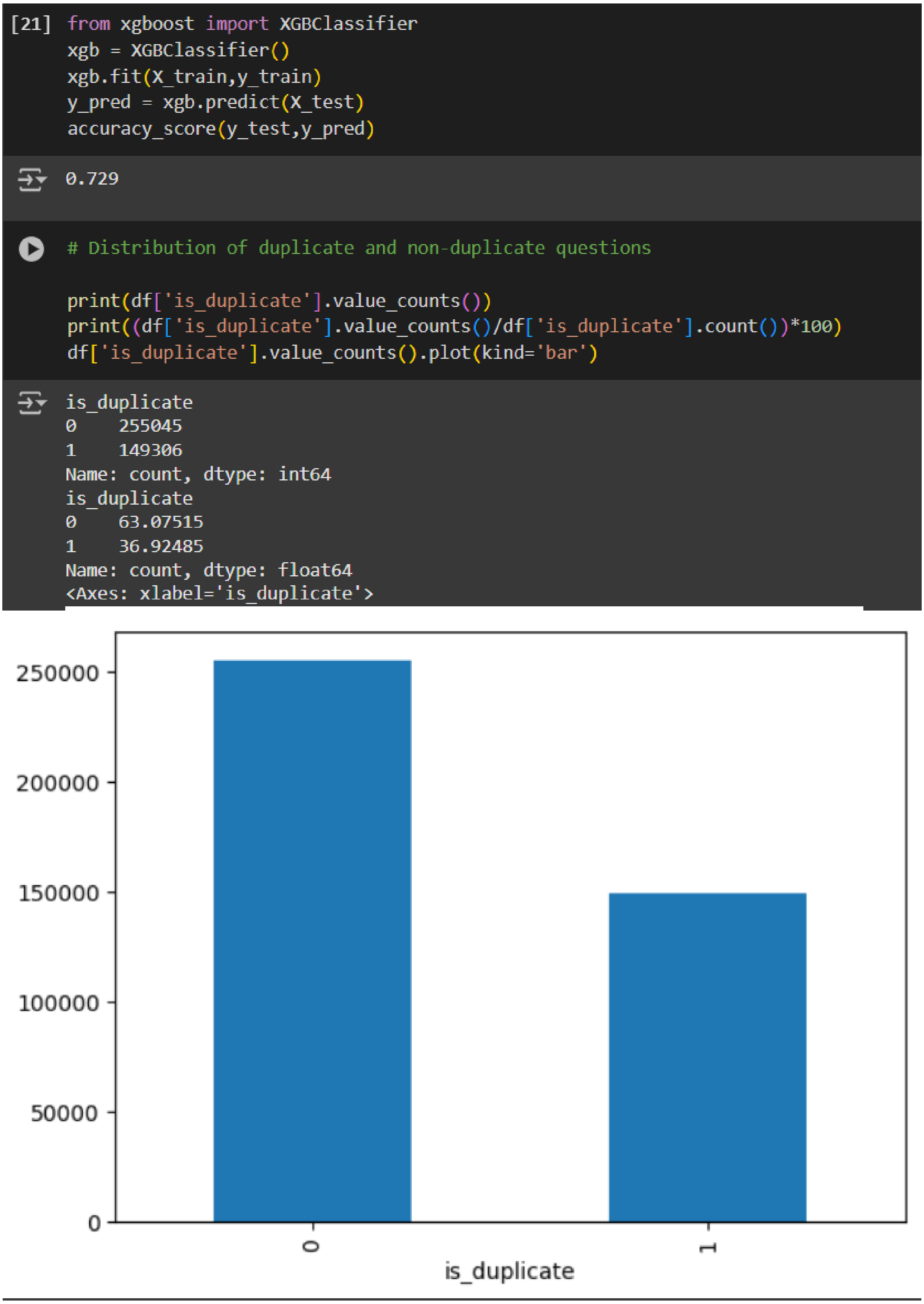
```
[19] temp_df.head()
```

	0	1	2	3	4	5	6	7	8	9	...	2991	2992	2993	2994	2995	2996	2997	2998	2999	is_duplicate
221247	0	0	0	0	0	0	0	0	0	0	...	0	0	1	0	0	0	0	0	0	0
251317	0	0	0	0	1	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	1
306522	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	1
187263	0	0	0	0	0	0	0	0	0	0	...	0	0	0	0	0	0	0	0	0	0
131589	0	0	0	0	0	0	0	0	0	0	...	0	0	2	0	1	0	0	0	0	0

5 rows × 6001 columns

```
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test = train_test_split(temp_df.iloc[:,0:-1].values,temp_df.iloc[:,-1].values,test_size=0.2,random_state=1)
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score
rf = RandomForestClassifier()
rf.fit(X_train,y_train)
y_pred = rf.predict(X_test)
accuracy_score(y_test,y_pred)
```

```
0.7433333333333333
```

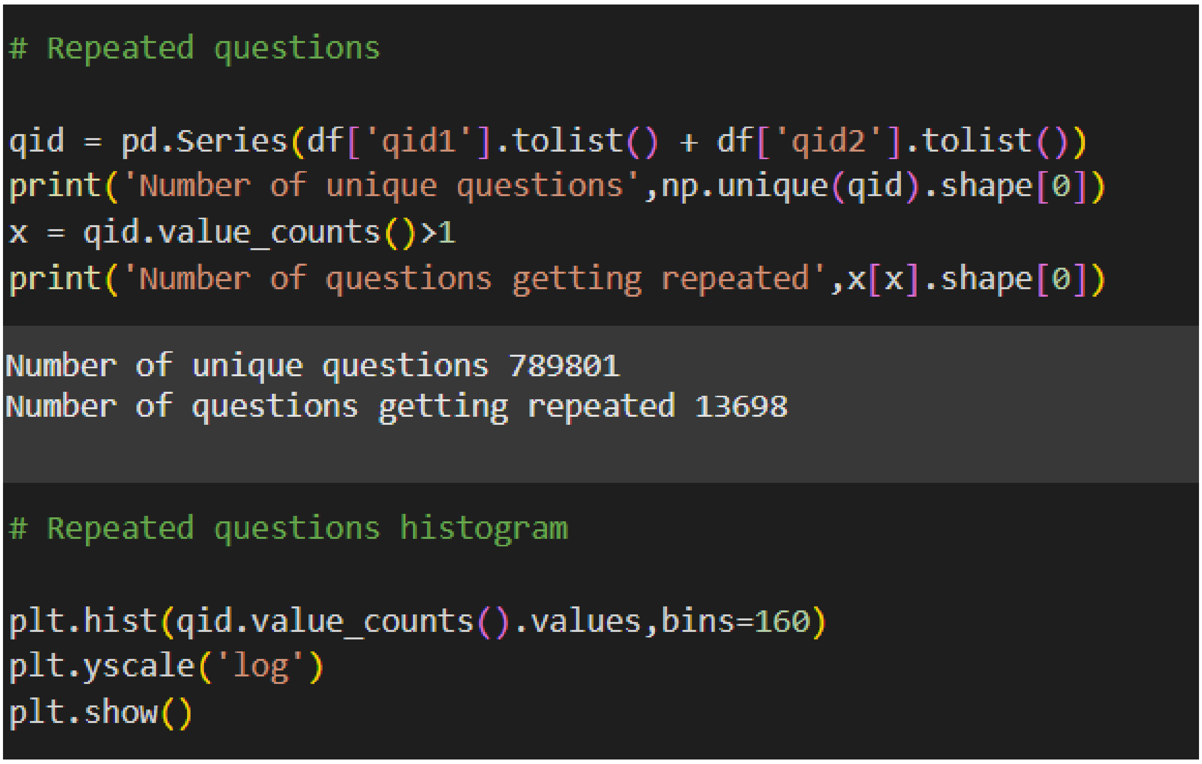


```
[21] from xgboost import XGBClassifier
     xgb = XGBClassifier()
     xgb.fit(X_train,y_train)
     y_pred = xgb.predict(X_test)
     accuracy_score(y_test,y_pred)
```

```
0.729
```

```
# Distribution of duplicate and non-duplicate questions

print(df['is_duplicate'].value_counts())
print((df['is_duplicate'].value_counts()/df['is_duplicate'].count())*100)
df['is_duplicate'].value_counts().plot(kind='bar')
```

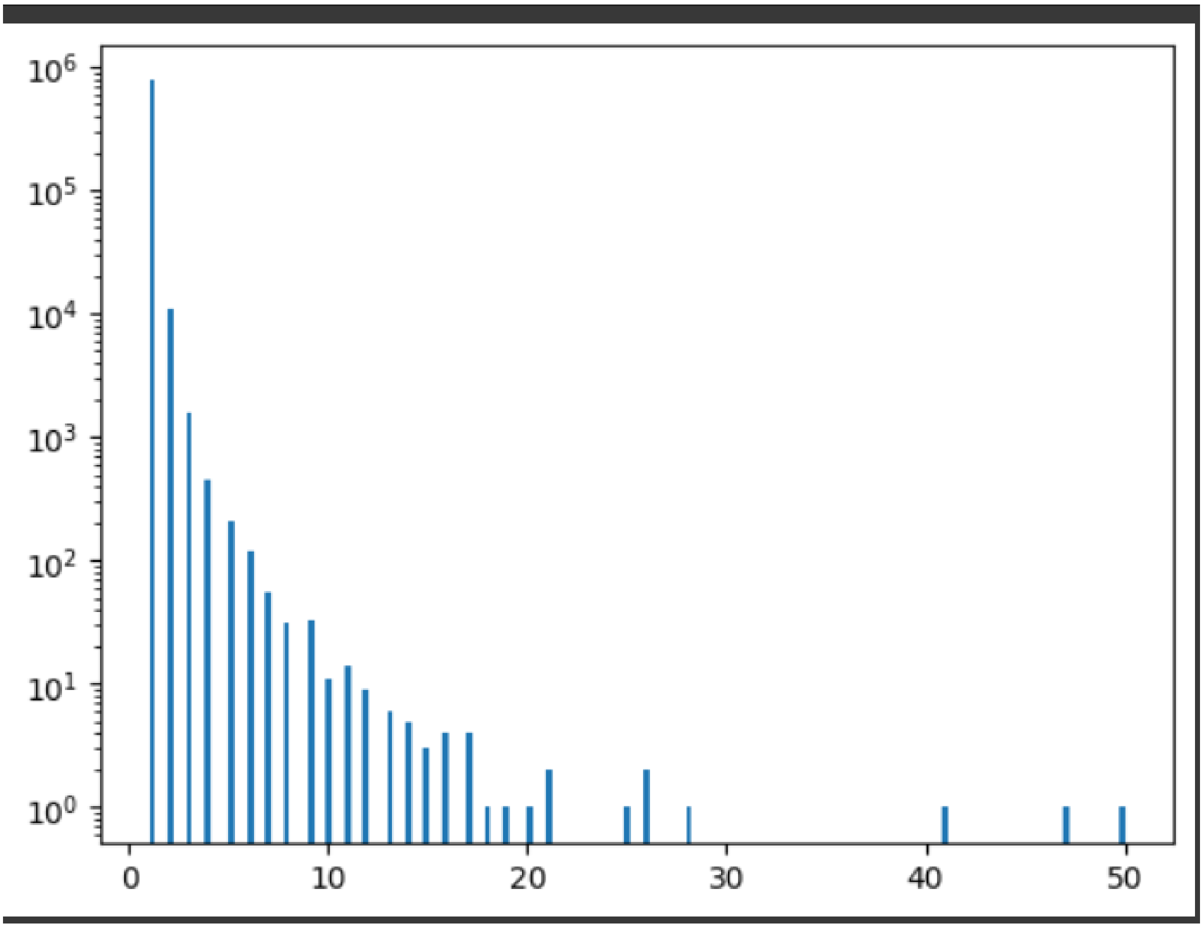
```
is_duplicate
0    255045
1    149306
Name: count, dtype: int64
is_duplicate
0    63.07515
1    36.92485
Name: count, dtype: float64
<Axes: xlabel='is_duplicate'>
```

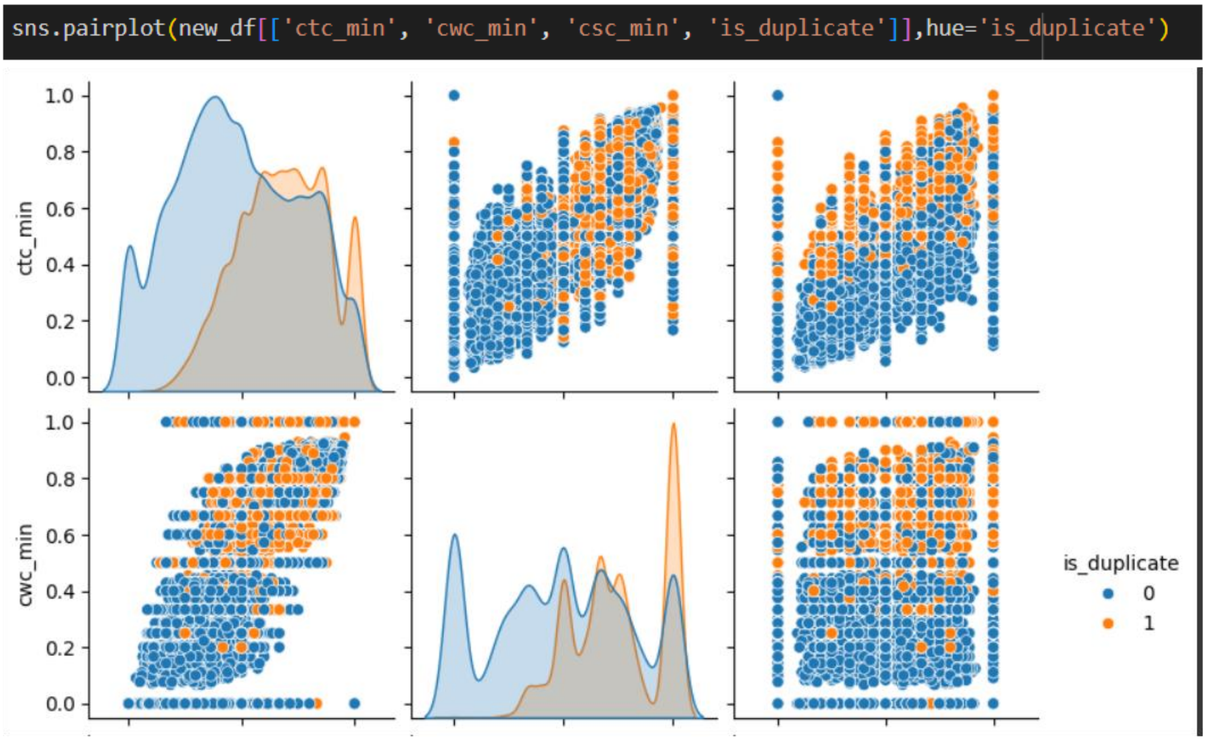


```
# Repeated questions

qid = pd.Series(df['qid1'].tolist() + df['qid2'].tolist())
print('Number of unique questions',np.unique(qid).shape[0])
x = qid.value_counts()>1
print('Number of questions getting repeated',x[x].shape[0])
```

```
Number of unique questions 789801
Number of questions getting repeated 13698
```

```
# Repeated questions histogram

plt.hist(qid.value_counts().values,bins=160)
plt.yscale('log')
plt.show()
```





```
sns.pairplot(new_df[['ctc_min', 'cwc_min', 'csc_min', 'is_duplicate']],hue='is_duplicate')
```

7. Constraints and Regulations

7.1. Data Privacy and Security

- Anonymization and Data Masking: All user data should be anonymized, removing any personally identifiable information (PII) to protect user privacy in accordance with data protection laws.
- Compliance with Privacy Laws: Adherence to privacy regulations such as GDPR (General Data Protection Regulation), DPDPA (Digital Personal Data Protection Act 2023) and other relevant laws must be maintained. User data handling, storage, and processing should follow these standards.

7.2. Ethical AI Practices

- Bias Prevention: Proactively address and mitigate any biases in the model by using diverse datasets. Regular reviews should be conducted to ensure fairness and minimize cultural, linguistic, or demographic biases.
- Transparency and Explainability: Document the model's duplicate detection process, so users understand the reasoning behind detected duplicates. This is crucial for maintaining user trust, especially on public platforms.

7.3. Dataset Constraints

- Data Licensing Compliance: Ensure that all datasets used are publicly available or legally licensed for commercial purposes, respecting copyright and intellectual property rights.
- Data Quality and Diversity: The training dataset should cover a range of question types, topics, and languages to ensure accurate duplicate detection across diverse content. Update the dataset regularly to reflect changing language and user behaviour.

8. Monetization Strategies for the website application

8.1. Subscription Model

Premium Access for Enhanced Features: Offer basic duplicate question detection for free, but include advanced features (e.g., detailed similarity scoring, bulk question processing) in a premium subscription.

8.2. Content Organization as a Service

B2B Service for Content Management: Partner with organizations with large Q&A databases (e.g., customer support sites, knowledge bases) to offer content organization services. Charge a monthly or yearly fee based on the amount of content processed or the complexity of the organization required.

8.3. Advertising Revenue

Targeted Ads Based on Search Patterns: Incorporate ads related to popular or frequently duplicated questions. For example, display ads for specific services or products based on commonly asked questions, ensuring relevance.

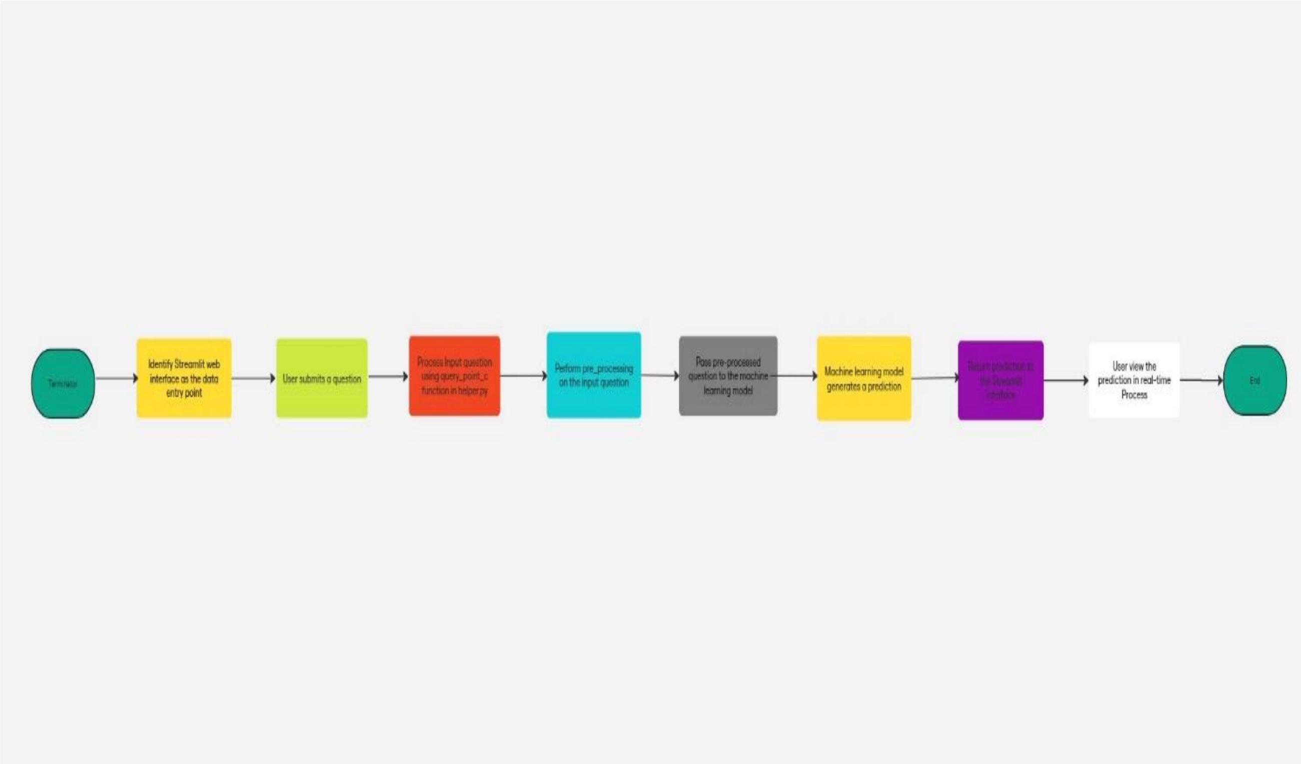
9. Final Product Prototype

9.1. Key Features:

- ➢ **User Interface for Query Entry**: The web application provides a simple interface where users can input question pairs to check for duplicates.
- ➢ **Real-Time Processing**: Once questions are submitted, the model processes them in real time, determining whether they are duplicates based on the binary classification model.
- ➢ **Result Display**: The application displays results in an intuitive format, indicating duplicate status and similarity scores to users, enhancing usability and transparency.
- ➢ **Application Workflow**: Users enter questions via a Streamlit interface, which are then processed by query_point_c in helper.py (cleaned, tokenized, encoded) before being analyzed by a machine learning model. The prediction

(answer or duplicate detection) is displayed in real-time, ensuring an interactive user experience.

- **Model Deployment on Heroku**: The entire application, including the model, API, and frontend, is hosted on Heroku, providing a scalable and cost-effective platform for deployment.


Terminator
Identify Streamlit web interface as the data entry point
User submits a question
Process Input question using query_point_c function in helper.py
Perform pre_processing on the input question
Pass pre-processed question to the machine learning model
Machine learning model generates a prediction
Return prediction to the Streamlit interface
User view the prediction in real-time Process
End

10.Various models used

10.1. Working:

- **Binary Classification Model**: The project uses a machine learning model designed for binary classification, which means it classifies inputs into two distinct classes—in this case, “duplicate” or “not duplicate.”
- **Duplicacy Check**: Its purpose is to check if two questions are duplicates, meaning they convey the same meaning despite possible wording differences.
- **is_duplicate () Function**: The function is_duplicate() is a part of the model that takes in a pair of questions and processes them to determine their duplicacy status.
- **Output as 0 or 1**: The model’s output is binary
  - 0 indicates that the questions are *not duplicates* (different questions).
  - 1 indicates that the questions are *duplicates* (same intent or meaning).

10.2. Approach Used:

Dataset Columns:

The dataset contains multiple columns:

- qid1: ID of the first question in a pair.
- qid2: ID of the second question in a pair.
- q1: The text of the first question.
- q2: The text of the second question.
- is_duplicate(): A label (binary) indicating if the questions are duplicates (1) or not (0).

Bag of Words (BoW) on q1 and q2:

- A Bag of Words model is applied to q1 and q2, which are columns 3 and 4.
- This technique converts the text data into numerical features based on the frequency of words, enabling the model to interpret the textual content of the questions.

Target Variable (Y):

- The is_duplicate column (column 5) is designated as the target variable, often referred to as Y.
- This column is what the model will attempt to predict (whether questions are duplicates or not).

Feature Columns (q1 and q2):

- Columns 3 (q1) and 4 (q2) represent the feature columns that provide the input data.
- These features are combined with Y to build a dataset on which machine learning algorithms will be applied.

Algorithm Application:

- A **Random Forest Classifier** is chosen as the algorithm to classify questions as duplicates or not.
- Random Forest is known for its robustness and accuracy in handling classification tasks with high-dimensional data.

Model Accuracy:

- The Random Forest Classifier is applied to the dataset to evaluate its effectiveness.

- The goal is to determine how accurately the model can predict duplicate questions by analysing q1, q2, and the target label is_duplicate.

11. Advanced Features

11.1. Token Features

- cwc_min: This is the ratio of the number of common words to the length of the smaller question
- cwc_max: This is the ratio of the number of common words to the length of the larger question
- csc_min: This is the ratio of the number of common stop words to the smaller stop word count among the two questions.
- csc_max: This is the ratio of the number of common stop words to the larger stop word count among the two questions.
- ctc_min: This is the ratio of the number of common tokens to the smaller token count among the two questions.
- ctc_max: This is the ratio of the number of common tokens to the larger token count among the two questions.
- last_word_eq: 1 if the last word in the two questions is same, 0 otherwise.
- first_word_eq: 1 if the first word in the two questions is same, 0 otherwise.

11.2 Length Based Features

- mean_len: Mean of the length of the two questions (number of words)
- abs_len_diff: Absolute difference between the length of the two questions (number of words)
- longest_substr_ratio: Ratio of the length of the longest substring among the two questions to the length of the smaller question

11.3 Fuzzy Features

- Fuzz_ratio: fuzz_ratio score from fuzzywuzzy
- Fuzz_partial_ratio: fuzz_partial_ratio from fuzzywuzzy
- Token_sort_ratio: token sort ratio from fuzzywuzzy
- Token_set_ratio: token set ratio from fuzzywuzzy

Conclusion:

The *Duplicate Question Pairs* project harnesses machine learning to identify and manage duplicate questions, improving content organization on Q&A platforms. By accurately detecting similar queries, it enhances search relevance, reduces redundancy, and enriches user experience by unifying related questions. This scalable solution exemplifies the practical impact of AI in optimizing content management and information retrieval, with potential applications across various content-heavy platforms.

Resources and References:

Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow by Aurélien Géron: Offers practical guidance on building machine learning models and implementing NLP techniques.

Speech and Language Processing by Daniel Jurafsky and James H. Martin: Comprehensive guide on NLP fundamentals, especially useful for text similarity models.