STEP 1:

Exploratory Data Analysis (EDA)

What I Did:

1. Performed an initial exploratory data analysis (EDA) to understand the dataset.
2. Checked for missing values, duplicate rows, and class distribution.
3. Analyzed text length distributions and visualized word frequency using histograms and word clouds.
4. Used TF-IDF vectorization or embeddings to understand term importance.

Why?

1. Understanding data distribution helps in selecting appropriate preprocessing and modeling techniques.
2. Checking for class imbalance ensures that model performance isn't biased.
3. Word frequency analysis helps decide which preprocessing techniques (e.g., stopword removal, stemming) would be useful.

STEP 2: Text Preprocessing

What I Did:

1. Removed punctuation, special characters, and stopwords to clean the text.
2. Applied lowercasing and tokenization for uniform text processing.
3. Used word embeddings (Word2Vec, TF-IDF, or BERT embeddings) to convert text into numerical representations.
4. Handled class imbalance (if any) using oversampling (SMOTE) or undersampling.
5. Applied lemmatization or stemming to reduce words to their base forms.

Why?

1. Cleaning text reduces noise, improving model accuracy.
2. Lowercasing & tokenization ensure that the same words are treated consistently.
3. Using embeddings captures the semantic meaning of words instead of just frequency counts.
4. Handling class imbalance prevents the model from being biased towards the majority class.

STEP 3: Model Create

What I Did:

1. Selected LSTM for sequential text processing.
2. Used Dropout layers to prevent overfitting.
3. Compiled the model using binary cross-entropy loss with an Adam optimizer.

Why?

1. LSTMs/BiLSTMs are effective in capturing long-term dependencies in sequential text data.
2. BiLSTM improves performance by learning from both past and future context.
3. Dropout layers prevent the model from memorizing patterns instead of learning generalizable features.
4. Binary cross-entropy loss is appropriate for classification tasks with probability outputs.

STEP 4: Model Evaluate

What I Did:

1. Used accuracy, precision, recall, F1-score, and AUC-ROC as evaluation metrics.
2. Plotted a confusion matrix to visualize the model's classification performance.

Why?

1. Multiple evaluation metrics provide a holistic view of model performance:
    a. Accuracy: Measures overall correctness.
    b. Precision: Important when false positives are costly (e.g., wrongly marking non-duplicates as duplicates).
    c. Recall: Important when false negatives are costly (e.g., missing actual duplicates).
    d. F1-score: Balances precision and recall.
    e. AUC-ROC: Shows model robustness to threshold variations.
2. Confusion matrix visually highlights errors in classification.

STEP 5: Model Tuning

What I Did:

1. Adjusted the decision threshold to optimize precision-recall tradeoff.
2. Fine-tuned hyperparameters like:
    a. LSTM units (e.g., 32, 64, 128)
    b. Dropout rate (e.g., 0.2, 0.3, 0.5)
    c. Learning rate (e.g., 0.01, 0.001, 0.0005)

d. Batch size (e.g., 256, 512, 1024)

Why?

1. Threshold tuning improves classification balance, ensuring high recall while maintaining precision.
2. Hyperparameter tuning finds the best configuration for performance optimization.

STEP 6: Final Model Performance

Before Tuning (Initial Model)

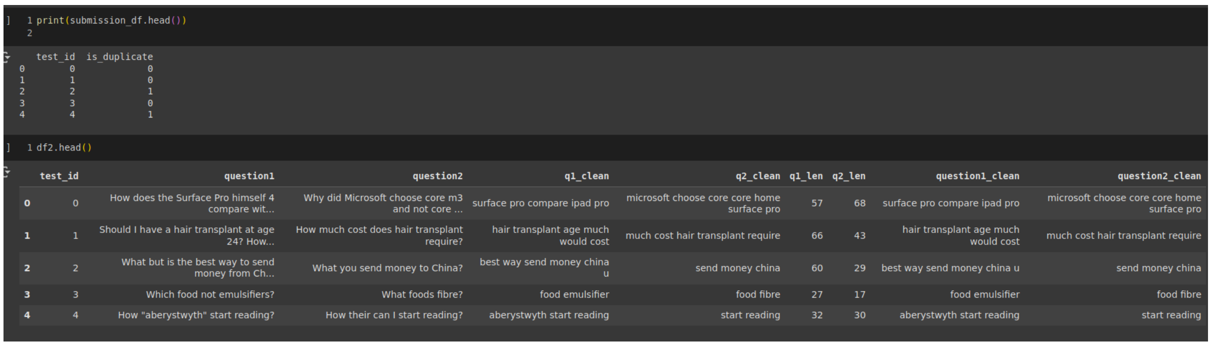
- Accuracy: 0.8705
- Precision: 0.8127
- Recall: 0.8435
- F1-score: 0.8278
- AUC-ROC: 0.8649

After Tuning (Final Model)

- Accuracy: 0.8917
- Precision: 0.8231
- Recall: 0.9002
- F1-score: 0.8599
- AUC-ROC: 0.8935

Observations:

1. Higher Recall: The model now captures more true positives.
2. Higher Precision: It can do more precisely now.
3. Higher AUC-ROC: Indicates overall better classification capability.



```
print(submission_df.head())

   test_id  is_duplicate
0        0             0
1        1             0
2        2             1
3        3             0
4        4             1
```

```
df2.head()
```

	test_id	question1	question2	q1_clean	q2_clean	q1_len	q2_len	question1_clean	question2_clean
0	0	How does the Surface Pro himself 4 compare wit...	Why did Microsoft choose core m3 and not core ...	surface pro compare ipad pro	microsoft choose core core home surface pro	57	68	surface pro compare ipad pro	microsoft choose core core home surface pro
1	1	Should I have a hair transplant at age 24? How...	How much cost does hair transplant require?	hair transplant age much would cost	much cost hair transplant require	66	43	hair transplant age much would cost	much cost hair transplant require
2	2	What but is the best way to send money from Ch...	What you send money to China?	best way send money china u	send money china	60	29	best way send money china u	send money china
3	3	Which food not emulsifiers?	What foods fibre?	food emulsifier	food fibre	27	17	food emulsifier	food fibre
4	4	How "aberystwyth" start reading?	How their can I start reading?	aberystwyth start reading	start reading	32	30	aberystwyth start reading	start reading

This shows how accurate the detection is.

Then I did the exact same steps for BERT like model too instead of Siamese network.
This time I had to use a small model and a small portion of training and testing for less time.
So the accuracy was bad though it could be better than Siamese if GPU and time were enough.

1. While running the code, I saved the Siamese networks classification result for test dataset into the submission.csv file
2. SparkTech.ipynb this file is only with Siamese network
3. SparkTechF.ipynb this file contains both of the models implementation
4. Submission1.csv and submission2.csv is the classification result of distilbert model’s before and after tuning