Data cleaning

Check for null-values(Rafsaan), ensuring correct data type (integer for class, text for comments” -

Some important data cleaning procedures include checking for null values in a dataset, and ensuring that the datatype of each row is correct which can impact our classification task. Rows with null values either have to be filled in or deleted and after checking our data, there were 0 null values present in the “comments” and “class” column

Feature selection

Max-df = 0.7 (from grid search CV) but using this value did not make a difference with the test accuracy.

Max_df argument for the tf-idf vectorizer filters out terms that appear in more than the specified proportion of documents. Max_df = 0.7 was the best parameter found after performing gridsearch, which translates to “do not use words as features that appear in more than 50% of sentences”. However applying this parameter to our vectorizer did not reduce the amount of features already present after applying max_df = 2.

Vectorizing the features using TF-IDF vectorizer - Transformation of text data into vector data (explain the math behind tf-idf-vectorizer) (Rafsaan)

The Tf-IDF vectorizer take a sentence as an input and gives us a numeric, output vector which can then be fed into the different machine learning models we implemented. The calculation has 2 components:

1. **Term frequency (TF):** Measures the occurence of a term/feature in the sentence relative to the total number of terms in the sentence.

   Formula = # of times a word/feature appears in sentence i /# of words in sentence i

2. **Inverse domain frequency score:** Downweights the terms that appear frequently across multiple sentences, and gives more weight to rarer features/words.

   Formula = log(Total # of sentences / # of sentences the word / feature appears in)

TF-IDF score: Multiplying the 2 scores above gives us the TF-IDF score of each feature/word which is stored in the output vector. TF-IDF provides a meaningful representation of the text, emphasising words that are specific to certain comments while downplaying generic ones like "the" or "and."

Formula = Tf * IDF

5. Metrics

Accuracy as a Metric:

We picked accuracy score as a metric to measure model performance since it is easy to implement and interpret. It measures the proportion of correct classifications made by our model out of all the predictions.

Formula: Accuracy= # of correct predictions/ Total # of predictions

Or 1/n (i = 1 to n) summation 1{yi = f^(xi)} where n is the # of sentences provided to classify.

While accuracy works well for balanced datasets, it may be misleading in cases of class imbalance. For example, if spam comments are rare, the model might achieve high accuracy by simply predicting "non-spam" most of the time.

6. Results and Conclusion

Word Selection: The model identifies patterns in spam comments by emphasizing specific terms, and giving them more weight in the tf-idf vector output.

- **Common spam indicators**: Words like *"click," "check out," "subscribe,"* and *"link"* are often flagged as spam due to their promotional nature.
- **Impact of misclassification**: Mislabeling legitimate comments with similar phrases highlights the importance of nuanced vectorization and improved feature selection.

Conclusion:

Logistic Regression was the model selected for its simplicity and effectiveness. It demonstrated strong classification performance with the given dataset, achieving an initial 95.98 % accuracy . While it achieved the project's primary goal of distinguishing spam from non-spam, the approach can be further refined by experimenting with advanced vectorization techniques such as using BERT to vectorize the sentences, and then using a neural network approach.