Navya Ajay

Homework 3

1) In my analysis of pandemic-related communications on social media, the textual content of tweets is transformed into meaningful features for modeling using several techniques. Primarily, the text is processed through a TfidfVectorizer to capture both unigrams (single words) and bigrams (pairs of consecutive words), which are then weighted by their Term Frequency-Inverse Document Frequency (TF-IDF) scores. This method helps in emphasizing words that are important in a particular tweet but not commonly used across all tweets, thus providing a nuanced measure of textual relevance. The choice of these features is driven by the need to capture the context and specific language nuances that might indicate influential communication. Additionally, sentiment scores derived from the text provide a direct measure of the tone (positive, negative, or neutral) of each message, which is crucial in understanding how leaders' messages might affect public perception and behavior during a crisis. These features are chosen to provide a comprehensive understanding of the communicative impact, capturing both the explicit content and the underlying sentiment of the tweets.
2) The chosen classifier for this project is the Support Vector Machine (SVM), particularly effective for binary classification tasks like distinguishing between high and low engagement tweets. SVMs are well-suited for this project due to their ability to handle high-dimensional data, such as the features derived from text processing (unigrams, bigrams, TF-IDF scores). Moreover, SVMs work well with sparse data matrices, which are typical when text data is transformed into TF-IDF vectors. Compared to classifiers like k-Nearest Neighbors (kNN), SVM is more efficient at handling the large volume and high dimensionality of data without a significant drop in performance due to the curse of dimensionality. Additionally, SVMs provide a clear margin of separation between classes, which is beneficial in ensuring robustness and accuracy in classification outcomes, especially when the classes may be imbalanced as is often the case with engagement metrics in social media data. This makes SVM a more appropriate choice over simpler models like Naïve Bayes, which might not perform as well with the text features' skewed distributions and interdependencies.
3) In evaluating the performance of the SVM classifier, the technique of k-fold cross-validation was utilized, specifically employing a 5-fold setup. This method is particularly useful as it ensures that every data point gets to be in a test set exactly once and in a training set four times, thus providing a comprehensive assessment of the model’s performance across different subsets of data. This approach helps mitigate any bias that could arise from non-random splits of the data, making the evaluation more robust and reliable. For performance metrics, accuracy, precision, and recall were employed. Accuracy measures the overall effectiveness of the classifier in correctly identifying both high and low engagement tweets. Precision and recall are crucial in contexts where the costs of false positives and false negatives differ; precision measures

the correctness achieved in the positive (high engagement) predictions, while recall addresses the classifier's ability to capture all actual positive cases. These metrics together provide a holistic view of the classifier's performance, considering both the balance of predictions and their correctness.

4) Implementation
    a) For data preprocessing, the Python library Pandas was used to handle basic operations like renaming columns. TextBlob was employed for sentiment analysis, extracting the polarity of each tweet's text. This measure gauges the sentiment's positivity or negativity. The preprocessing did not include stopword removal, stemming, or tokenization explicitly stated in the code. However, these steps could enhance model performance by reducing noise and focusing on meaningful content
    b) The TF-IDF vectorization of tweet texts was performed using Scikit-learn's TfidfVectorizer, which processes the text to compute the Term Frequency-Inverse Document Frequency values for unigrams and bigrams. This technique emphasizes words that are important but not frequent across all documents, thereby capturing key thematic elements in the text data. No explicit feature selection or thresholding for term frequency was mentioned, but such methods could be integrated to focus on more relevant features. Normalization is inherently part of the TF-IDF process, which scales the features to have a uniform impact on the results.
    c) The Support Vector Machine (SVM) classifier was implemented using Scikit-learn's SVC function, chosen for its efficacy in handling high-dimensional data. The kernel used was 'rbf' (Radial Basis Function), which is effective for non-linear data separation. Parameter tuning was conducted through GridSearchCV, optimizing 'C' and 'gamma' to find the best settings for model accuracy. This method systematically works through multiple combinations of parameters, cross-validating the results to ensure robustness.
    d) For k-fold cross-validation, the dataset was partitioned using Scikit-learn's StratifiedKFold, with k set to 5. This method ensures each fold is a good representative of the whole by maintaining approximately the same percentage of samples of each target class as the complete set. In each of the 5 iterations, the dataset was divided into distinct training and test sets, with one fold used for testing and the others for training, rotating systematically through all folds.
    e) Performance metrics such as accuracy, precision, and recall were calculated using Scikit-learn's classification_report, which provides these statistics for each class. This function evaluates the classifier's predictions against the actual labels from the test sets generated during each fold of the cross-validation, summarizing the classifier's ability to identify high and low engagement tweets effectively. The

input to this function includes the true labels and predicted labels, with the output being a detailed breakdown of each metric for assessing model performance.

5) Summary of Performance Metrics by Fold

Fold	Accuracy	Precision (Class 1)	Recall (Class 1)	F1-Score (Class 1)	Precision (Class 0)	Recall (Class 0)	F1-Score (Class 0)
1	25%	0.25	1.00	0.40	1.00	0.01	0.01
2	25%	0.25	1.00	0.40	1.00	0.00	0.01
3	25%	0.25	1.00	0.40	1.00	0.00	0.01
4	25%	0.25	1.00	0.40	1.00	0.00	0.01
5	25%	0.25	1.00	0.40	1.00	0.00	0.01

Average Performance Across All Folds

Metric	Value
Average Accuracy	25%
Avg Precision (1)	0.25
Avg Recall (1)	1.00
Avg F1-Score (1)	0.40
Avg Precision (0)	1.00
Avg Recall (0)	0.00
Avg F1-Score (0)	0.01