Assignment 4

Approach:
The first improvement I made to increase accuracy was replacing the manual text cleansing and fixed keyword dictionaries from Assignment 3 with scikit-learn's TfidfVectorizer. This tool converts raw documents into numerical features by calculating term frequency inverse document frequency (TF-IDF) scores, which automatically downweighs common words and up weighs more relevant words. Many of the manual cleansing done in the previous assignment could have remove certain tokens that may have been relevant.

The second improvement I made was to split the original training data into a smaller training set and a validation set. The validation set was used for hyperparameter tuning. I tuned the regularization factor (C) for the same classifiers used in Assignment 3 (logistic regression OvR, logistic regression OvO, and SVM) and tried different TF-IDF settings (unigrams, unigrams and bigrams, and different min_df or max_df values). Training with the new TF-IDF settings with all the classification models resulted in better accuracy score. These accuracy scores were calculated after the tuning stage utilizing the test data.

Observations regarding accuracy:
This new method achieved a much higher accuracy than 45% seen in Assignment 3 where SVM achieved an accracy of 67.5%, logistic regression with OvR achieved 67.5% and logistic regression OvO achieved an accuracy of 66.46%. I observed tuning the TF-IDF settings with both unigrams and bigrams and filtering out extremely rare or common terms help increase the accuracy score. Among the classifiers, SVM performed better than logistic regression (OvR and OvO) where a smaller C value (except less that 1) gave the best results.

Why I got the accuracy:
I believe the main reason the accuracy improves is because there is better representation of the text to train the model. TF-IDF features capture more descriptive and relevant vocabulary than the fixed keyword lists from Assignment 3. Including bigrams helps the model learn short phrases which can help distinguish different articles. Using a validation split to tune C lets the linear models find a good bias-variance trade-off.

Credit: ChatGPT for helping with code writing