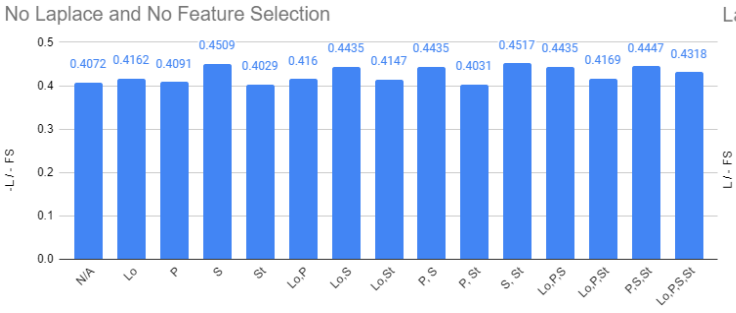
The implemented model is able to classify movie reviews using a Naive Bayes Classifier, (Laplace smoothing optional). 4 preprocessing methods are available in the `Preprocessor()` class: lowercasing, selective punctuation removal, NLTK stemming, and NLTK stop list filtering. Preprocessing + tokenising is performed on all datasets (training dataset + dev/test dataset). The model is then trained in the `Train()` class by calculating and storing prior probabilities and likelihoods.

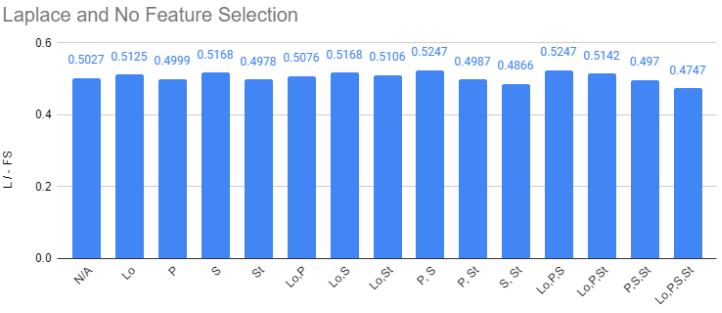
When classifying in the `Classify()` class, the model uses the priors to calculate the posterior probabilities of each sentence, and classifies the sentences' sentiment based on the largest posterior probability. It is also able to select features from a sentence if the features are in a list of ~6800 opinion words by Bing Liu and Minqing Hu.

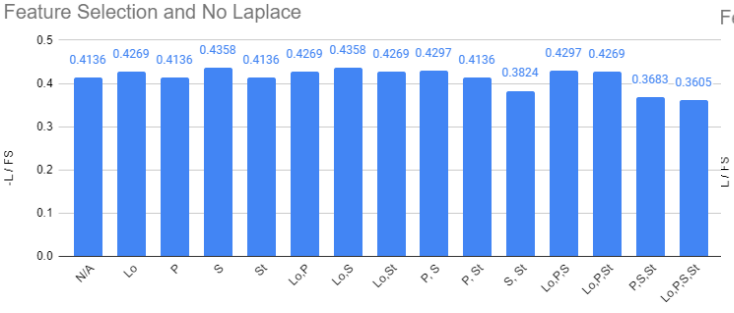
The model can also calculate confusion matrices and macro F1 scores and plot them in the `Evaluate()` class. It is also able to write its classification results into a .tsv file. Most of the methods above can be toggled by adjusting the parameters starting at line 24, and by command line arguments.

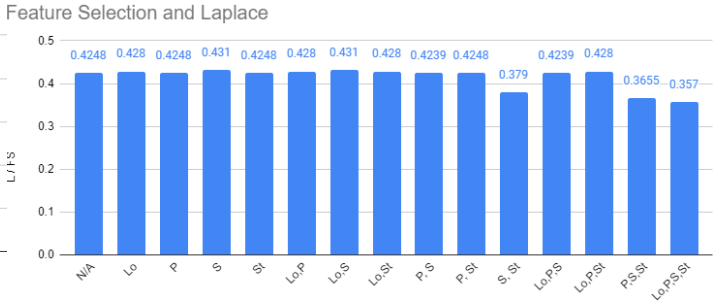
Legend: Lowercasing = Lo, Punctuation Removal = P, Stemming = S, Stoplisting = St, Laplace = L, FS = Feature Selection, N/A = no preprocessing

3 classes (error with the y-axis label; should be micro F1)**:**

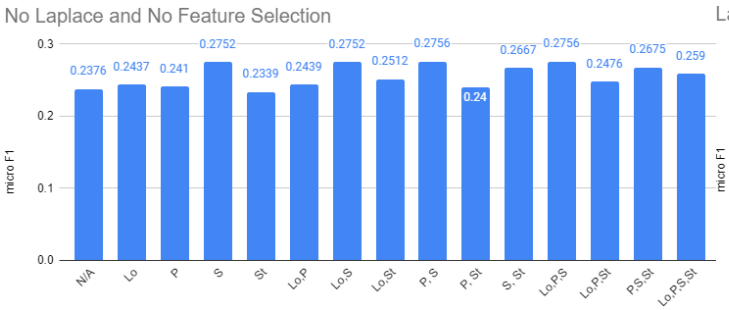


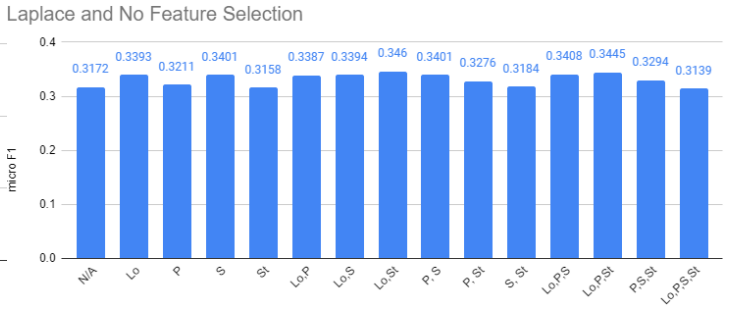


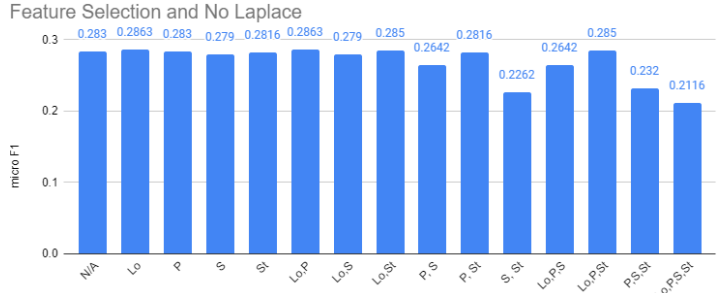


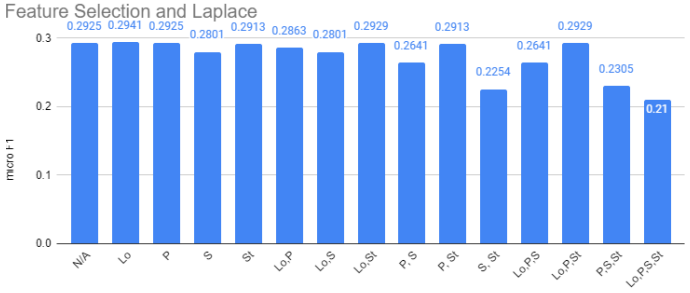


5 classes:









The results seem to indicate that the model is significantly better than the majority-class classifier, being able to outperform it in terms of their macro-F1 score by 262% in classifying 3 classes and 381% in classifying 5 classes. Preprocessing mostly seems to give a marginal performance increase. Lowercasing and stemming consistently increases the macro F1 score when used. Stemming was also used in all but 2 of the top performing configurations.

On the contrary, punctuation removal and stoplisting gave worse scores sometimes than outright not preprocessing. When used alone, stoplisting always yields equal or worse scores than not performing preprocessing, and has a tendency to reduce scores when paired with stemming. Punctuation removal does not work with the feature selection classifier, as it does not account for punctuation. Stoplisting also does not seem to work with the feature selection classifier, because the stoplist and the opinion word list do not have any common words, thus making the stoplist redundant. Curiously, when the classifier performs feature selection, punctuation removal and stoplisting will reduce the micro F1 score when paired with stemming, even though punctuation removal and stoplisting does nothing in feature selection.

The feature selection (FS) for this model can perform slightly better than the baseline classifier (no FS + no Laplace) at classifying 3 classes, and significantly better at classifying 5 classes. The FS was improved marginally by Laplace smoothing. The FS implicitly filters out less "informative" words, as opinion words are more often indicative of certain sentiments, which makes it perform better than the baseline classifier. The FS also narrows down the amount of words used in classification. This reduces the chances of the classifier running into a word that is absent in the training dataset, which can cause inaccurate classifications, as evident by the marginal improvement when paired with Laplace smoothing. The FS (relative to the all-words approach) also does better in 5 classes vs 3 classes because opinion words can be more nuanced than non-opinion words. Opinion words can express slight to powerful sentiments, while non-opinion words are more absolute in what they convey. This makes the FS perform comparatively better in the 5 sentiment scale, as the scale can reflect the nuance better. It should be noted that the performance dips significantly when going from 3 classes to 5 classes in all configurations due to the curse of dimensionality, making it easier to misclassify sentences. (e.g. if a document was correctly classified as 3, the same document would be classified as 4 or 5 in 5-class, and only one of those classes would be the correct classification.)

However, the FS does worse than the all-words classifier with Laplace smoothing. This might be because the FS' list does not contain a lot of words, and it is not updated to accommodate Internet slang. For example, the sentence "complete snoozefest with predictable plot" will get classified by the FS as a neutral sentence, as none of those words are in the list. As the data is drawn from Rotten Tomatoes, a publicly available online review aggregate service, users are more likely to use Internet slang to express their sentiment, which the FS can't accommodate. The FS will definitely do better when classifying professionally written reviews, though it is unknown if it will beat the all-words classifier w/ Laplace.

Laplace smoothing works really well in general because it slightly skews the posterior probability towards 0.5 instead of outright making it 0 when encountering a word that wasn't in the training set. This makes the posterior probabilities more accurate, as a sentence with only one unknown word shouldn't be considered as 0%.

The dev set will be classified with configurations that yielded the highest macro F1 scores, as the goal of this assignment is to maximise that. As such, the 3 classes classification model would be an all-words classifier with Laplace smoothing, with punctuation removal and stemming. The 5 classes model would also be an all-words classifier with Laplace smoothing, with lowercasing, punctuation removal, and stopwords. To play it safe with classifying the test set, a consistently high performing configuration would be chosen. The model would be an all-words classifier with Laplace smoothing. Preprocessing would involve punctuation removal and stemming, as the combination was able to perform very well, if not the highest scoring one, in the 3/5 classes all-words classifiers, with or without Laplace.