Module Code: CS3TM20
Assignment Report Title: Individual Report
Student Number: 26011251
Date when work completed: 11/03/21
Actual hours spent for the assignment: 35

Assignment evaluation:

1. I am not sure whether I have included enough processes for the first part of the coursework as it was not made clear in the coursework brief what processes are specifically required.
2. Though I was unsure on the processes, I am confident with the results for each method, having created confusion matrices and bar charts from the predictions.
3. A clearer coursework brief, specifying each process to use, would be very beneficial for future versions of the coursework.

Task 1: NLP Analysis

The first task requires the exploration of different NLP analysis methods, in use with the logistic regression classifier, in order to predict the sentiment of the development set tweets. Sentiment is defined as 'a thought, opinion, or idea based on a feeling about a situation, or a way of thinking about something' [1]. In the context of the coursework, the sentiment of a tweet is positive, negative, or neutral in tone.

Four distinct methods for the application of feature weighting have been explored. Each method has been conducted with, and without, preprocessing of the text, allowing to see its effect upon the accuracy of the sentiment prediction.

Preprocessing

Preprocessing is the preparation of text before it is passed to the analysis functions; by removing unnecessary items, the analysis model is able to predict sentiment with greater accuracy.

Original tweet: "@USAirways Being put back on hold for what has now been an HOUR is completely unacceptable"
Clean tweet: "put back hold hour complete unaccept"

The first feature of the tweets to be removed are any @mentions. The account name someone is tweeting to does not confer opinion or feeling, and as such will not assist in predicting the tweets' sentiment. In this vain, the retweets and any hyperlinks may also be removed as they also are lack a greater meaning.

The next target for removal is the punctuation, including any hashtags. Although punctuation can help split the text up, as well as confer whether a sentence is a question or an exclamation, the sentiment is determined by the words within the sentence, not the punctuation. Furthermore, the tweets have already been separated within the dataset, there is therefore no need to partition the text further.

Another way to clean the data is by converting all the text to lowercase. Words in lowercase are considered different from words in uppercase. For example, 'Canada' and 'canada' are spelt the same but are considered as two distinct words by the classifier [2]. Similarly, a person may unnecessarily capitalise letters, such as with 'DOG' or 'bICYcle'. Through converting all text to lowercase, each word is reduced to one consistent form (misspellings will still produce other distinct forms however).

The final process is to implement stemming. Stemming removes the suffix of a word, reducing it to its root form [3]. For example, both 'waiting' and 'waited' are reduced to the root 'wait'. This helps the classifier discern the sentiment of a word as each use within the text is reduced to the root form. Two problems exist with stemming which may reduce the accuracy of the classification. The first is false positives, where words with different stems are stemmed to the same root, such as: universal, university, and universe. The second is false negatives, where words that should be stemmed to the same root are not, such as: alumnus, alumni, and alumnae.

1.1 Single Models

Method 1: tf-idf bag of words vectorisation, logistic regression classification

To determine the relevancy of each word within the text data, the clean text must be vectorised. The TF-IDF vectoriser is 'a statistical measure that evaluates how relevant a word is to a document in a collection of documents' [4]. The TF-IDF for an individual word is calculated from how many times it appears in a document (term frequency) and how common it is across multiple documents (inverse document frequency). The higher the TF-IDF score, the more relevant the word is within the document.

TF = (Number of times the word occurs in the text) / (Total number of words in text)
IDF = (Total number of documents / Number of documents with word in it)
*TF-IDF = TF * IDF*

For example:

Sentence1: "The angry man shouted at the windmill"

Sentence2: "The protester ran from the angry policeman"

TF-IDF for 'angry' in Sentence1 = (1/7)(2/2) = 0.143*

TF-IDF for 'shouted' in Sentence1 = (1/7)(2/1) = 0.286*

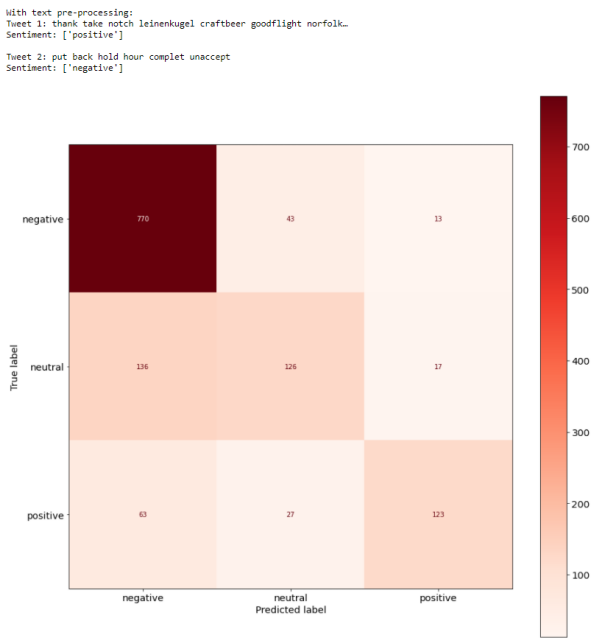
'Shouted' has a higher TF-IDF score than 'Angry' in Sentence1 so is of greater importance for sentiment analysis.

Logistic regression is a discriminative model based upon the sigmoid curve that directly models the conditional probability p(y|x) [5]. The model is trained using the training set vectors and the training set sentiments. The success of a model's prediction is conferred by 4 metrics [6]:

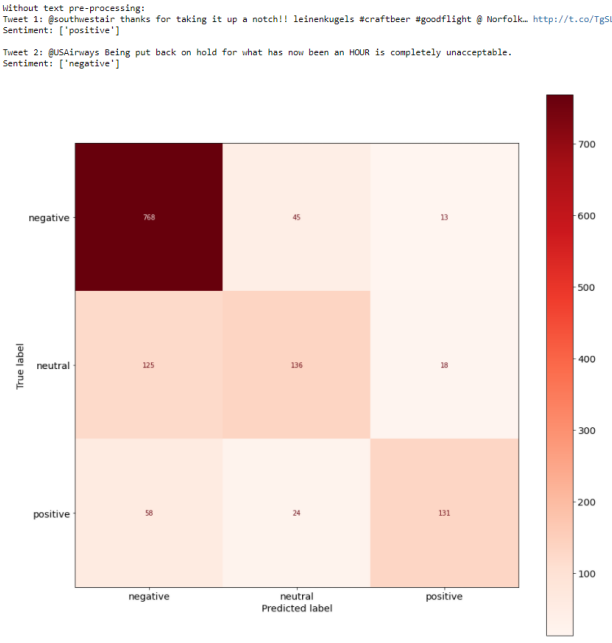
Accuracy - ratio of correctly predicted observations to the total observations
Precision - the ratio of correctly predicted positive observations to all the observations in the actual class
Recall - the ratio of correctly predicted positive observations to all the observations in the actual class
F1 Score - weighted average of Precision and Recall
(A score of 1 is the optimal for each of the metrics)

The success of the model can also be judged by the construction of a confusion matrix. The confusion matrix shows what the model predicted vs what the true value is. A higher number of correct predictions indicates a greater success in classification. The correct predictions are found down the diagonal of the matrix.

Curiously, the accuracy with preprocessing, 0.77, is less than the accuracy without preprocessing, 0.79. This disparity of 0.02 is relatively inconsequential for a small dataset but for a large data set is cause for concern. The errors with stemming may have caused this loss in accuracy however 0.77 is still a good score considering the wild variation in spelling and writing styles found in the tweets.



	precision	recall	f1-score	support
negative	0.79	0.93	0.86	826
neutral	0.64	0.45	0.53	279
positive	0.80	0.58	0.67	213
accuracy			0.77	1318
macro avg	0.75	0.65	0.69	1318
weighted avg	0.76	0.77	0.76	1318



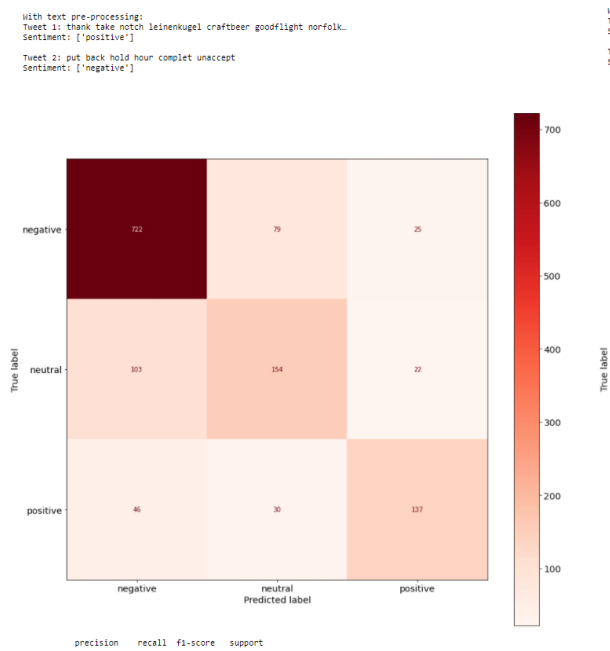
	precision	recall	f1-score	support
negative	0.81	0.93	0.86	826
neutral	0.66	0.49	0.56	279
positive	0.81	0.62	0.70	213
accuracy			0.79	1318
macro avg	0.76	0.68	0.71	1318
weighted avg	0.78	0.79	0.77	1318

Method 2: binary representation, logistic regression classification

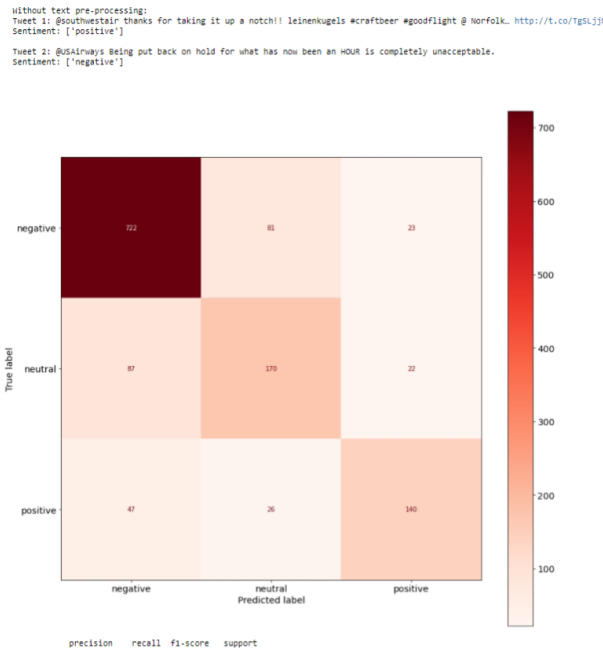
The second method utilises binary representation for the feature weighting. The vectors have a dimensionality equal to the size of the vocabulary, if the text data features the vocab word, a 1 for true will be placed in that dimension [7].

Vocab = [the, cat, was, hungry, dog, very, sad]

Sentence1: "The cat was hungry" = [1, 1, 1, 1, 0, 0, 0]



	precision	recall	f1-score	support
negative	0.83	0.87	0.85	826
neutral	0.59	0.55	0.57	279
positive	0.74	0.64	0.69	213
accuracy			0.77	1318
macro avg	0.72	0.69	0.70	1318
weighted avg	0.76	0.77	0.77	1318



	precision	recall	f1-score	support
negative	0.84	0.87	0.86	826
neutral	0.61	0.61	0.61	279
positive	0.76	0.66	0.70	213
accuracy			0.78	1318
macro avg	0.74	0.71	0.72	1318
weighted avg	0.78	0.78	0.78	1318

Sentence2: “The dog was very sad” = [1, 0, 1, 0, 1, 1, 1]

Similar to the the results for TF-IDF vectorisation, the binary count method produced accuracies of 0.77 with preprocessing and 0.78 without.

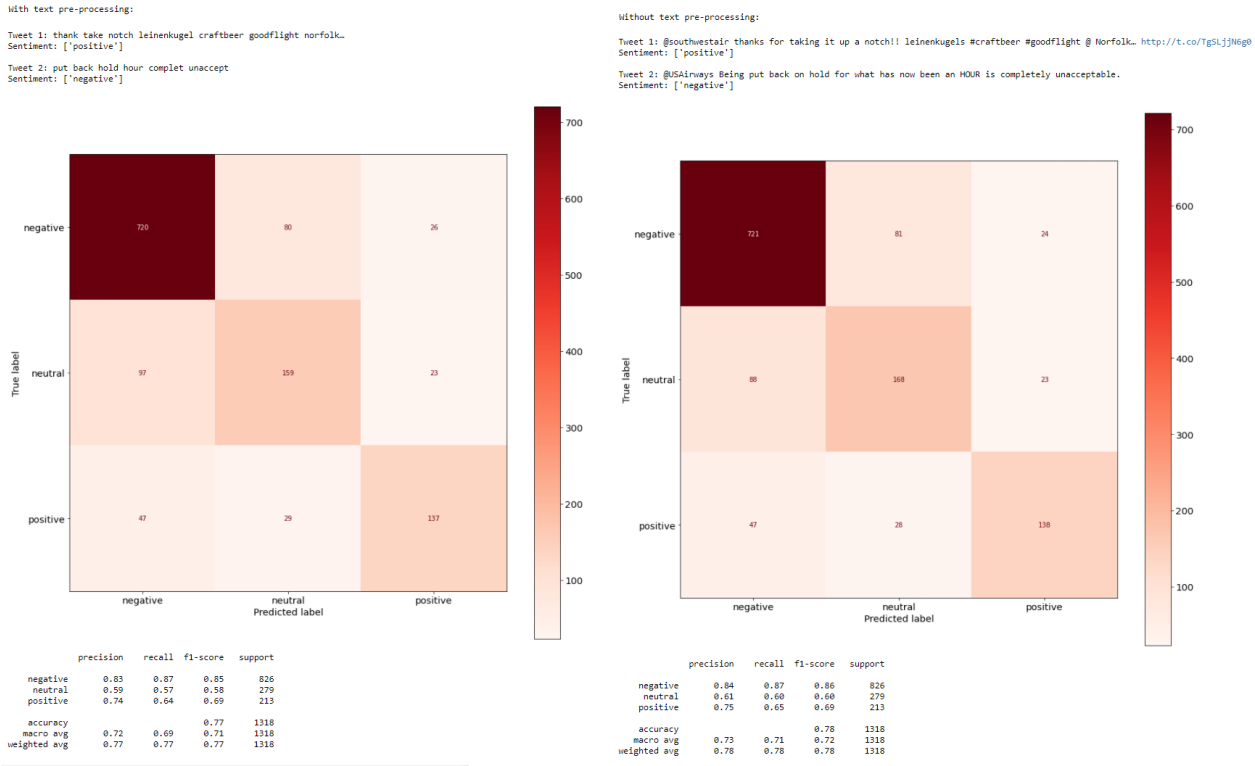
Method 3: count representation, logistic regression classification

Count representation vectorisation works in the same way as binary, though the number of occurrences of a word within the document is noted instead of simply whether it appears or not.

Vocab = [the, cat, was, very, hungry, and, tired, indeed, dog, sad]

Sentence1: “The cat was very hungry and very tired indeed” = [1, 1, 1, 2, 1, 1, 1, 1, 0, 0]

Sentence2: “The dog was very sad” = [1, 0, 1, 1, 0, 0, 0, 0, 1, 1]



Regular count vectorisation produces the same results as binary vectorisation, with accuracies of 0.77 with preprocessing and 0.78 without. All three of the basic vectorisation techniques produce accuracies of 0.77 with preprocessing, though TF-IDF has the greatest accuracy without preprocessing at 0.79. This suggests that while there is little difference in performance between the three, if given a choice, TF-IDF vectorisation should be used.

Method 4: word2vec, logistic regression classification

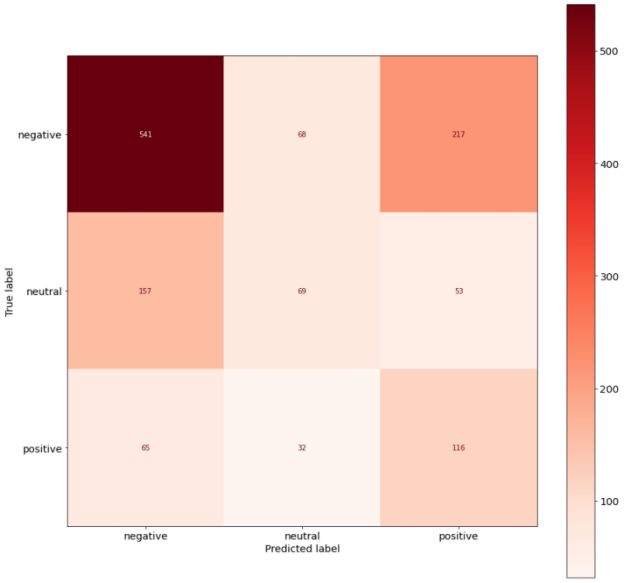
The word2vec model uses a neural network to learn word associations from a large corpus of text. Through assigning each word a vector, the cosine similarity between the vectors can be used to establish the semantic similarity between words.

‘Dog’ and ‘Cat’ would have greater semantic similarity than ‘Dog’ and ‘Car’ as they are both animals, whereas a car is not.

For classification of the tweets, the model is trained from all the text data. The average vector of each sentence is then calculated from the vectors of each word within the sentence. Sentences with a positive sentiment will have a similar vector to one another, likewise for negative and neutral tweets.

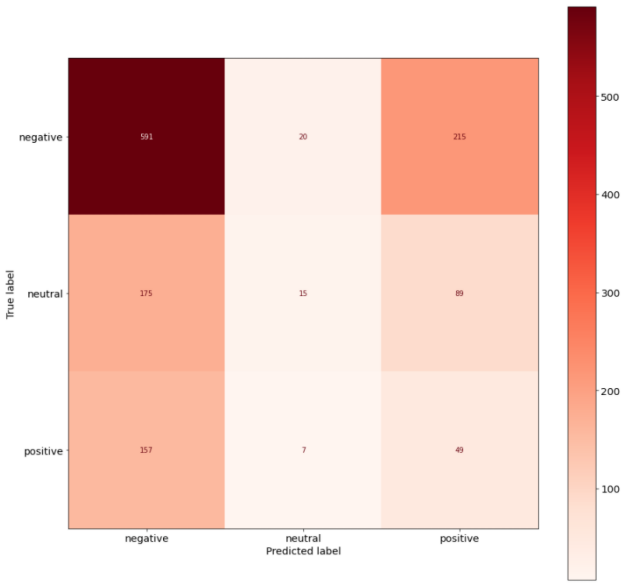
Compared to the other vectorisation methods, word2vec's accuracy in predicting the sentiments has been lacklustre, with an accuracy of 0.55 with preprocessing and 0.50 without. Predicting just over half of the sentiments correctly is poor given the other methods were able to do so with nearly 80% accuracy. Interestingly, the word2vec method is the only one in which preprocessing increasing the accuracy, in this case by 10% compared to without. The success of the word2vec method is dependent on the quality of the data so it is unsurprising that it has not performed well given the quality of written English displayed in some of the tweets.

With text pre-processing:



	precision	recall	f1-score	support
negative	0.71	0.65	0.68	826
neutral	0.41	0.25	0.31	279
positive	0.30	0.54	0.39	213
accuracy			0.55	1318
macro avg	0.47	0.48	0.46	1318
weighted avg	0.58	0.55	0.55	1318

Without text pre-processing:

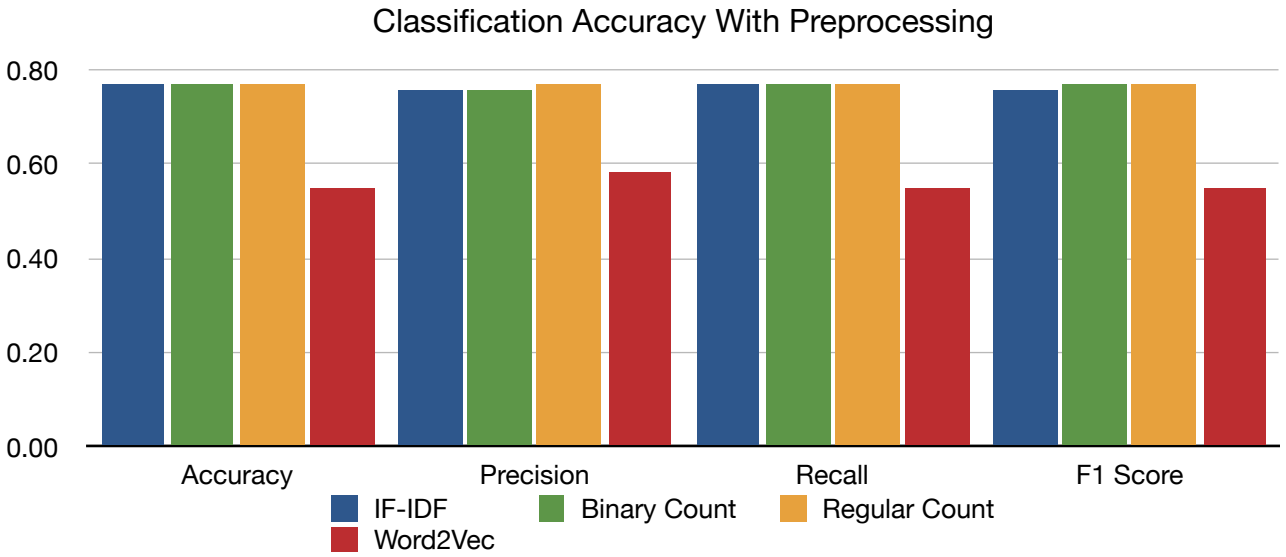


	precision	recall	f1-score	support
negative	0.64	0.72	0.68	826
neutral	0.36	0.05	0.09	279
positive	0.14	0.23	0.17	213
accuracy			0.50	1318
macro avg	0.38	0.33	0.31	1318
weighted avg	0.50	0.50	0.47	1318

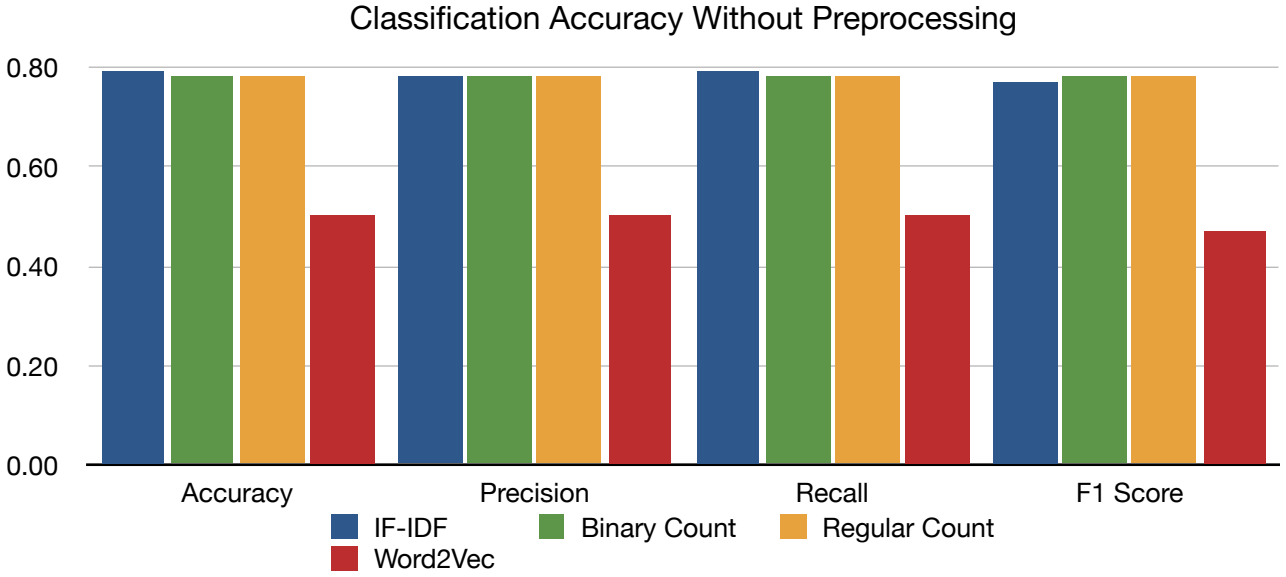
1.2 Comparison of all Single Models

From presenting the results in graph form it is clear that with preprocessing of the text, IF-IDF, binary count, and regular count vectorisation methods perform more or less the same, with values within 0.01 of one another. The best performing model is regular count vectorisation, though only by 0.01 more precision than binary count. The worst performing model is the word2vec vectorisation, with an accuracy of 0.55, 0.22 points behind the other three.

The same trend is shown by the results from classification without preprocessing of the text data. word2vec vectorisation trails behind the other three models, with only a 50% chance of correctly classifying a given tweet. The best performing model this time is IF-IDF, though again by a very small margin. The problem with using word2vec without cleaning the text data is that the number of misspelt words or oddities, combined with the sheer size of the corpus makes it hard to accurately generate vectors for each word, and in turn each sentence. The model may be improved by increasing the number of times a word must appear before it is included, though this would likely remove many correctly spelt and used words, not just those misspelt or containing random capitalisation.



	IF-IDF	Binary Count	Regular Count	Word2Vec
Accuracy	0.77	0.77	0.77	0.55
Precision	0.76	0.76	0.77	0.58
Recall	0.77	0.77	0.77	0.55
F1 Score	0.76	0.77	0.77	0.55



	IF-IDF	Binary Count	Regular Count	Word2Vec
Accuracy	0.79	0.78	0.78	0.50
Precision	0.78	0.78	0.78	0.50
Recall	0.79	0.78	0.78	0.50
F1 Score	0.77	0.78	0.78	0.47

Task 2: Prediction Challenge

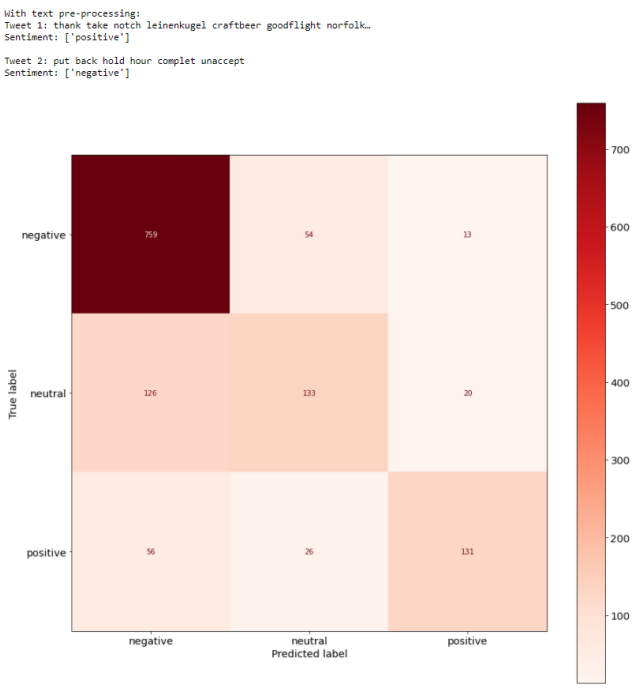
2.1 Combined Models

The second task requires the combined use of classification models to accurately predict the sentiment for the test set tweets. From exploration of the single models, all using the logistic regression classifier, the TF-IDF vectorisation method was shown to perform the best without preprocessing, and was just as accurate as the binary and regular count vectorisation methods; consequently TF-IDF vectorisation will be used in combination with the chosen classifier(s).

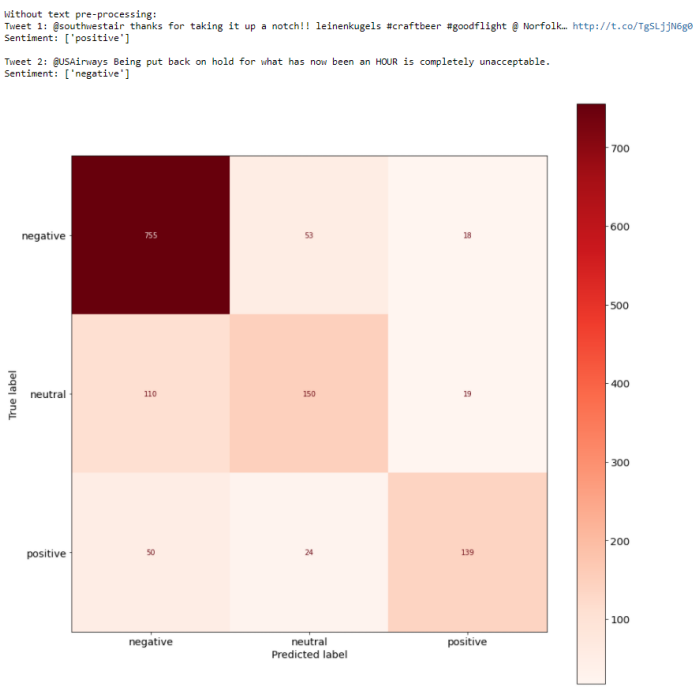
Method 1: tf-idf bag of words vectorisation, SVM classification

The first new classifier is the support vector machine. The objective of the SVM classifier is to find a hyperplane in an N-dimensional space that distinctly classifies the data points [8]. The optimal hyperplane is the one with the maximum margin, or in other words, the maximum distance between data points of both classes. In regards to the sentiment classification, the classes are positive, negative, and neutral; the hyperplane will consequently be 2-dimensional.

The results for SVM classification are very similar to those for logistic regression. Classification is conducted at an accuracy of 0.78 with preprocessing and 0.79 without.



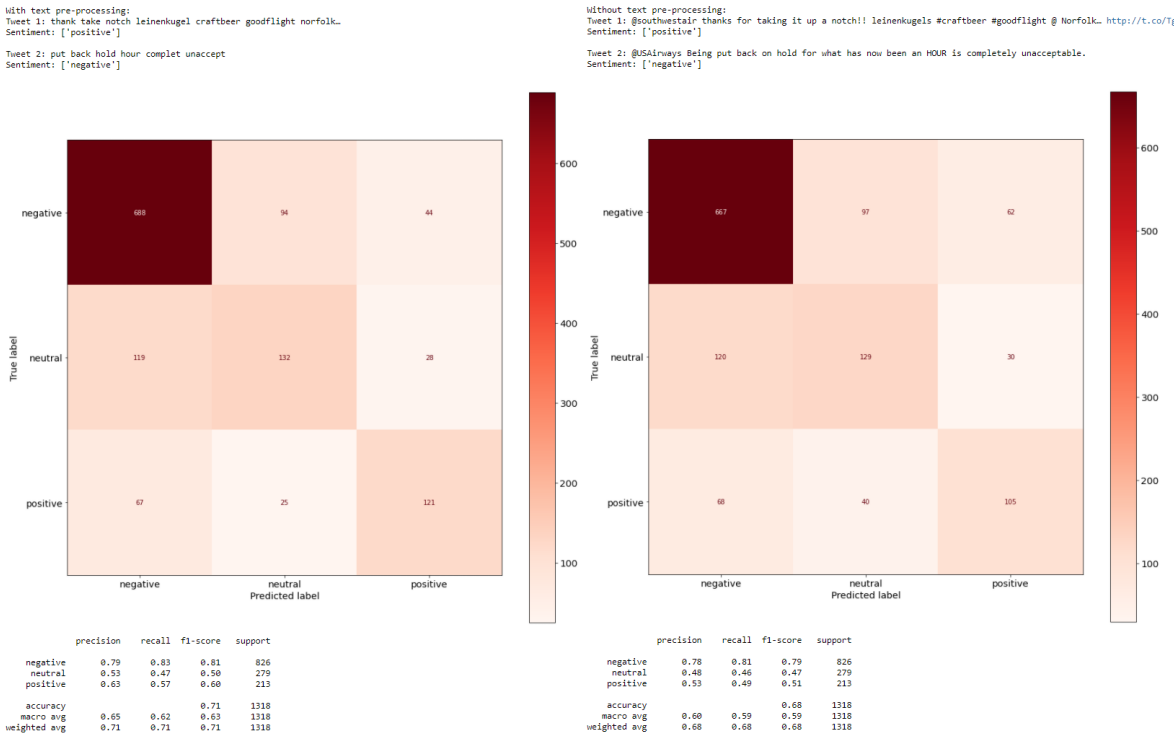
	precision	recall	f1-score	support
negative	0.81	0.92	0.86	826
neutral	0.62	0.48	0.54	279
positive	0.80	0.62	0.69	213
accuracy			0.78	1318
macro avg	0.74	0.67	0.70	1318
weighted avg	0.77	0.78	0.77	1318



	precision	recall	f1-score	support
negative	0.83	0.91	0.87	826
neutral	0.66	0.54	0.59	279
positive	0.79	0.65	0.71	213
accuracy			0.79	1318
macro avg	0.76	0.70	0.72	1318
weighted avg	0.78	0.79	0.78	1318

Method 2: tf-idf bag of words vectorisation, decision tree classification

The second new classifier is the decision tree. Classification decision trees use recursive partitioning to sort the vectors into clusters, in this case positive, negative, and neutral sentiments. The data is split into partitions, and then split up further along on each of the branches. While classification is extremely fast compared to other methods, and unimportant features (such as misspelt words) are excluded, decision trees are prone to overfitting and counter intuitive decision making; for a large data set this can be a detriment to the accuracy [9].



Indeed the accuracy of the decision tree classification is not among the best, predicting the tweet sentiments with an accuracy of 0.71 with preprocessing and 0.68 without. Like with the word2vec logistic classification, preprocessing has helped increase accuracy, though not to a score comparable to the first three logistic regression methods.

Method 3: tf-idf bag of words vectorisation, naive bayes classification

The third method uses the naive bayes classifier. The classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature. The Bayes Theorem forms the basis of the classifier:

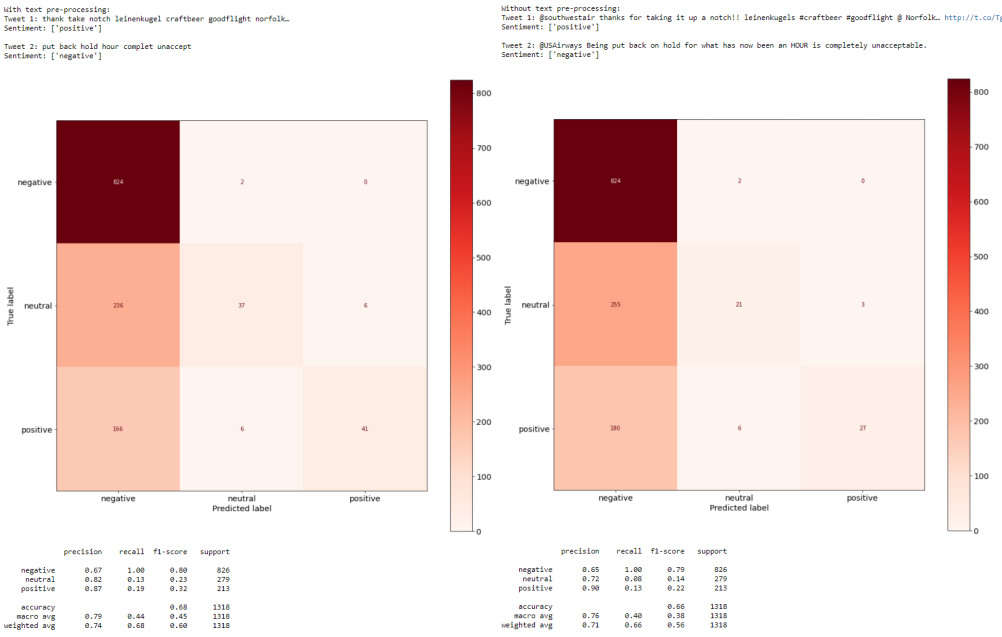
$$P(c|x) = \frac{P(x|c)\,P(c)}{P(x)}$$

P(c/x) is the posterior probability of class (c, target) given predictor(x, attributes)

P(c) is the prior probability of class

P(x/c) is the likelihood which is the probability of predictor given class

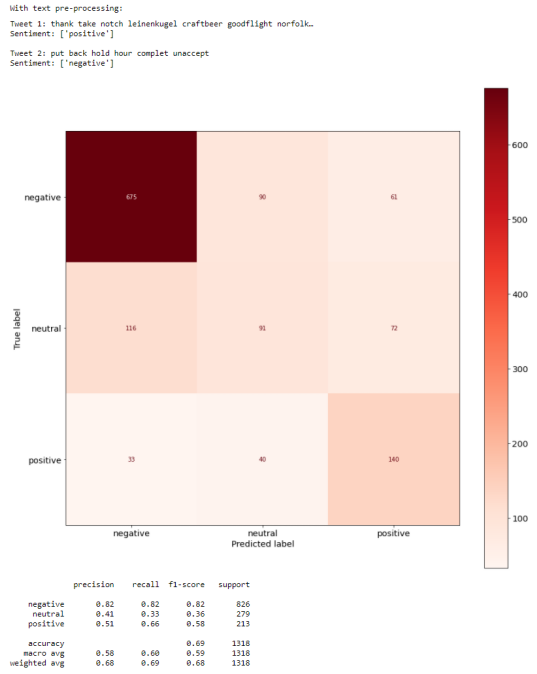
P(x) is the prior probability of predictor



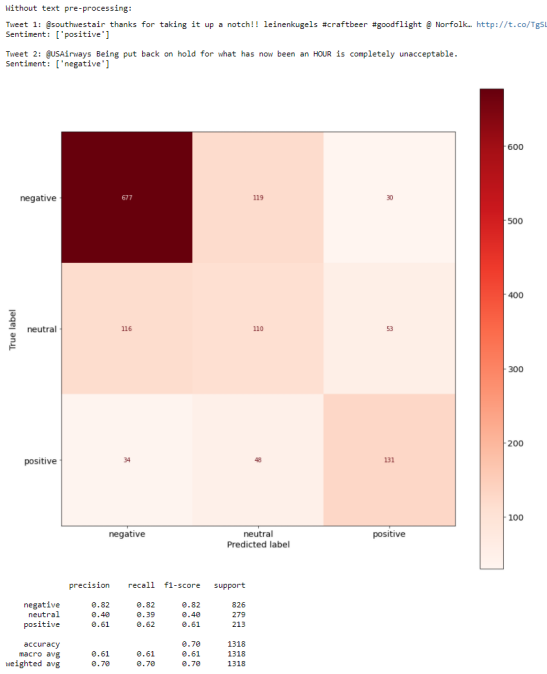
Naive Bayes also underperforms with accuracies of 0.66 and 0.68 with/without preprocessing. While the classifier is fast in multi-class prediction, it is know to be a bad estimator, so the outputs should ‘not be taken seriously’ [10].

Method 4: tf-idf bag of words vectorisation, multi-layer perceptron

Classification can also be conducted with a neural network that trains using back-propagation. The inputs to the network are the text data and the class labels (positive, negative, neutral). The classifier has faired well with the tweets, predicting with accuracies of 0.69 and 0.70 with/without preprocessing.



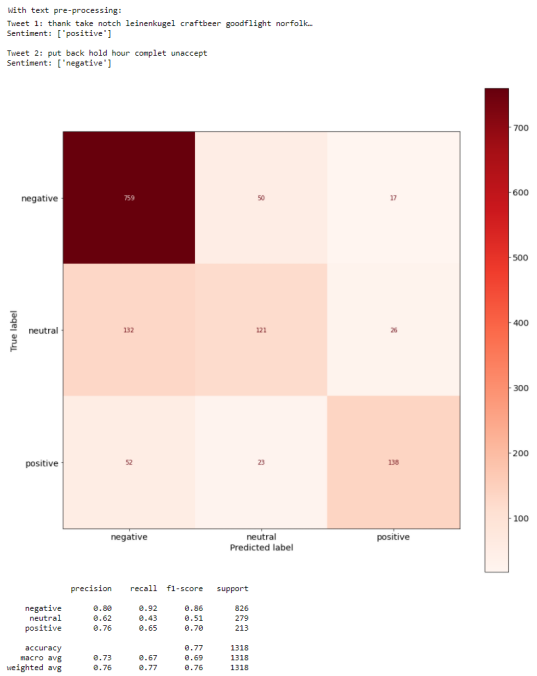
	precision	recall	f1-score	support
negative	0.82	0.82	0.82	826
neutral	0.41	0.33	0.36	279
positive	0.51	0.66	0.58	213
accuracy			0.69	1318
macro avg	0.58	0.60	0.59	1318
weighted avg	0.68	0.69	0.68	1318



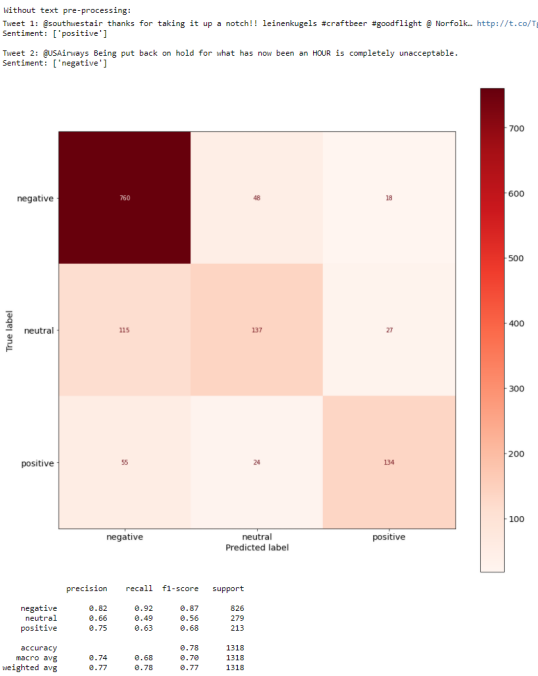
	precision	recall	f1-score	support
negative	0.82	0.82	0.82	826
neutral	0.40	0.39	0.40	279
positive	0.61	0.62	0.61	213
accuracy			0.70	1318
macro avg	0.61	0.61	0.61	1318
weighted avg	0.70	0.70	0.70	1318

Method 5: tf-idf bag of words vectorisation, stochastic gradient descent

The SGD classifier implements a plain stochastic gradient descent learning routine which supports different loss functions and penalties for classification [11]. The classifier has performed very well, achieving accuracies of 0.77 and 0.78 with/without preprocessing.



	precision	recall	f1-score	support
negative	0.80	0.92	0.86	826
neutral	0.62	0.43	0.51	279
positive	0.76	0.65	0.70	213
accuracy			0.77	1318
macro avg	0.73	0.67	0.69	1318
weighted avg	0.76	0.77	0.76	1318

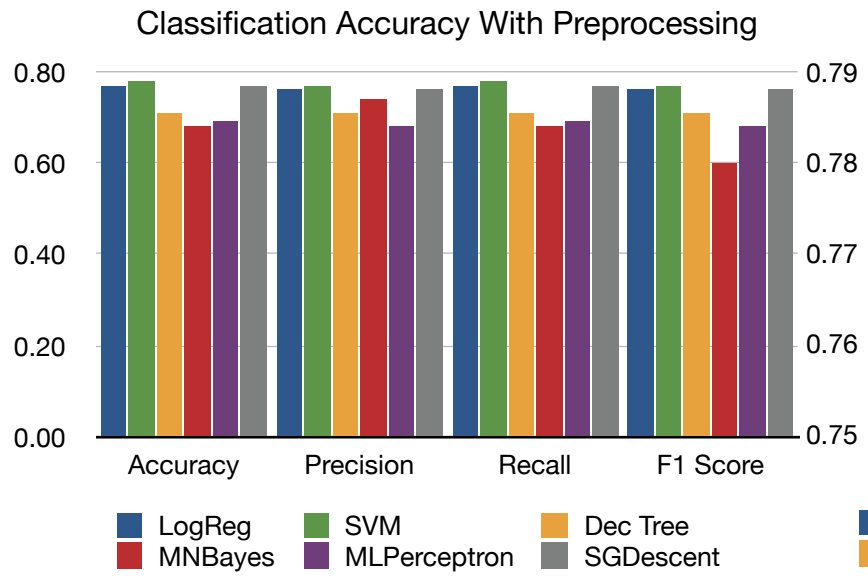


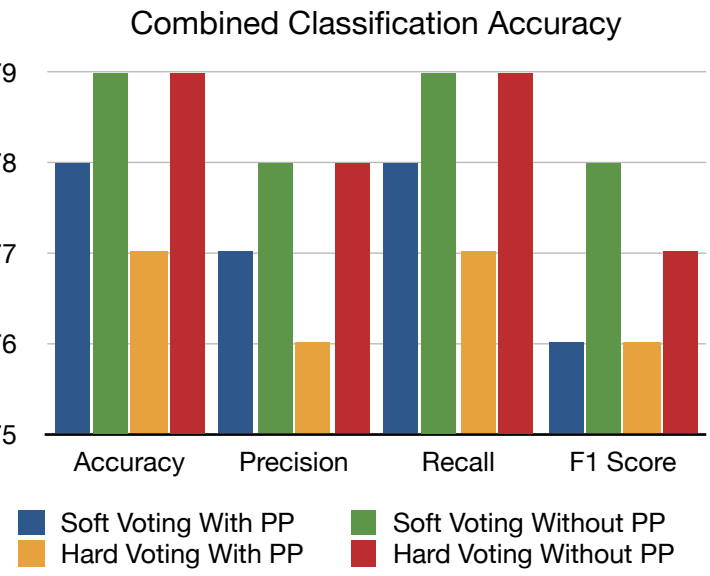
	precision	recall	f1-score	support
negative	0.82	0.92	0.87	826
neutral	0.66	0.49	0.56	279
positive	0.75	0.63	0.68	213
accuracy			0.78	1318
macro avg	0.74	0.68	0.70	1318
weighted avg	0.77	0.78	0.77	1318

2.2 Test Set Prediction

To predict the test set sentiments with the optimal accuracy, a combination of the best models is to be used. The ensemble voting classifier combines multiple classifiers, trained and evaluated in parallel in order to exploit the different aspects of each algorithm; doing so confers lower error and less over-fitting in the predictions [12]. With hard voting, the class receiving the highest number of votes will be chosen. With soft voting, the probability vector for each predicted class are summed up and averaged.

From testing of TF-IDF bag of words vectorisation with each classifier type, two of the best performing have been chosen: logistic regression and support vector machine classification. Testing has also occurred to determine whether soft or hard voting is optimal. The results show that soft voting without preprocessing will yield the greatest accuracy, and as such the test set sentiments will be predicted by this model.





	LogReg	SVM	Dec Tree	MNBayes	MLPerceptron	SGDescent
Accuracy	0.77	0.78	0.71	0.68	0.69	0.77
Precision	0.76	0.77	0.71	0.74	0.68	0.76
Recall	0.77	0.78	0.71	0.68	0.69	0.77
F1 Score	0.76	0.77	0.71	0.60	0.68	0.76

	Soft Voting With PP	Soft Voting Without PP	Hard Voting With PP	Hard Voting Without PP
Accuracy	0.78	0.79	0.77	0.79
Precision	0.77	0.78	0.76	0.78
Recall	0.78	0.79	0.77	0.79
F1 Score	0.76	0.78	0.76	0.77

Bibliography:

1. Sentiment Definition Cambridge Dictionary
2. 'When to convert to lowercase in preprocessing' Stack Overflow
3. Tushar Srivastava 'NLP: A quick guide to stemming' 06/08/19 Medium
4. Bruno Stecanella 'What is TF-IDF?' 10/05/19 MonkeyLearn
5. Dehao Zhang 'Sentiment Classification with Logistic Regression - Analysing Yelp Reviews' 26/07/20 Towards Data Science
6. 'Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures' 09/10/16 Exsilio Solutions
7. Hunter Heidenreich 'Natural Language Processing: Count Vectorization with scikit-learn' 24/08/18 Towards Data Science
8. Rohith Gandhi Support Vector Machine - Introduction to Machine Learning Algorithms 07/06/18 Towards Data Science
9. Afroz Chakure Decision Tree Classification 06/07/19 Medium
10. Sunil Ray 6 Easy Steps to Learn Naive Bayes Algorithm with codes in Python and R 11/09/17 Analytics Vidhya
11. Sklearn.linear_model.SGDClassifier skit-learn.org
12. Sanchita Mangale Voting Classifier 18/05/19 Medium