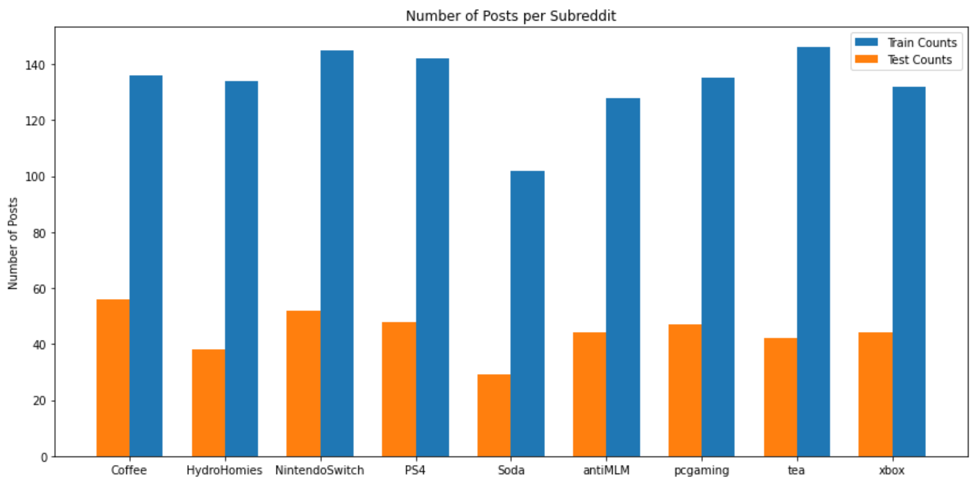
Lewis Ponsonby 2464980P

Q1a)

The below graphs display the distribution of labels across the training dataset and the validation dataset. The distributions show a similarity between labels across the two datasets with no obvious outliers, despite the difference in sample size. From this we can deduce the training dataset is representative of the validation and test data that will be used to assess the future classifiers.

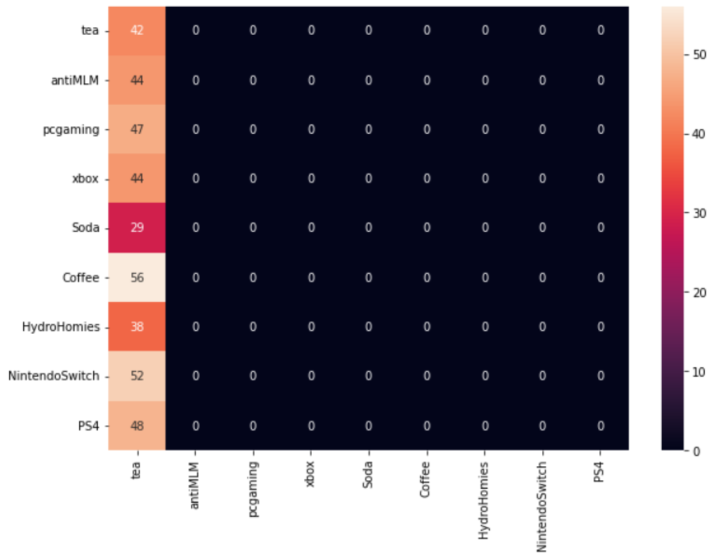


Q1b)

DummyClassifier – Strategy="most_frequent"

The below DummyClassifier model shows very poor fit to the test set. The method of the model is to classify all data inputs as the most frequent label in the dataset used for training. This results in all predictions to be assigned to the 1st subreddit - r/Tea.

The model scores 0.105 for accuracy score however this number is solely down to the fact a small fraction of the predictions will end up being assigned correctly to the only subreddit picked.



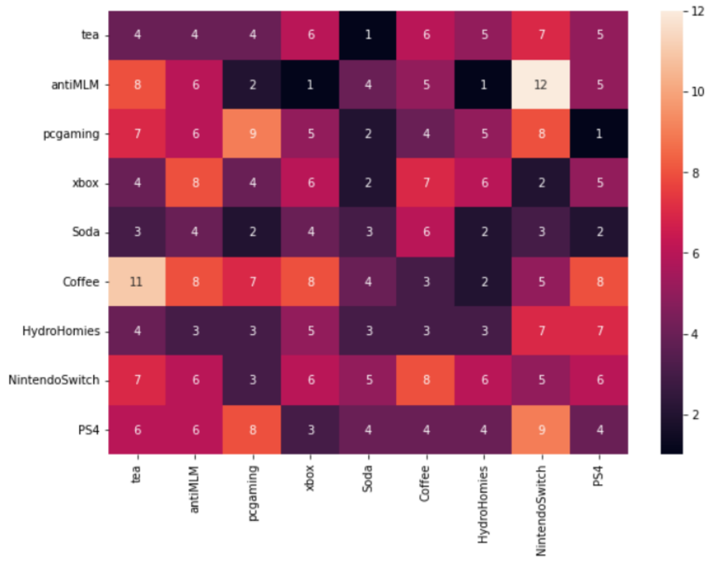
Accuracy Score	0.105
Macro Score	0.012
Weighted Score	0.011
Recall Score	0.105
F1 Score	0.020

DummyClassifier – Strategy=”stratified”

The below DummyClassifier models shows very poor fit to the test set. The method of the model is to randomly allocate labels across the test set based on the probability of each label in the train data set.

The model scores 0.115 for accuracy. However, this is once again due to the random probability that a Subreddit label is correctly allocated to a post.

Other than most_frequent and stratified, DummyClassifier can be passed with prior, uniform, or constant. Prior is similar to most_frequent whereas uniform generates predictions uniformly at random, where each class has an equal probability. Constant always predicts a constant label that is provided by the user.

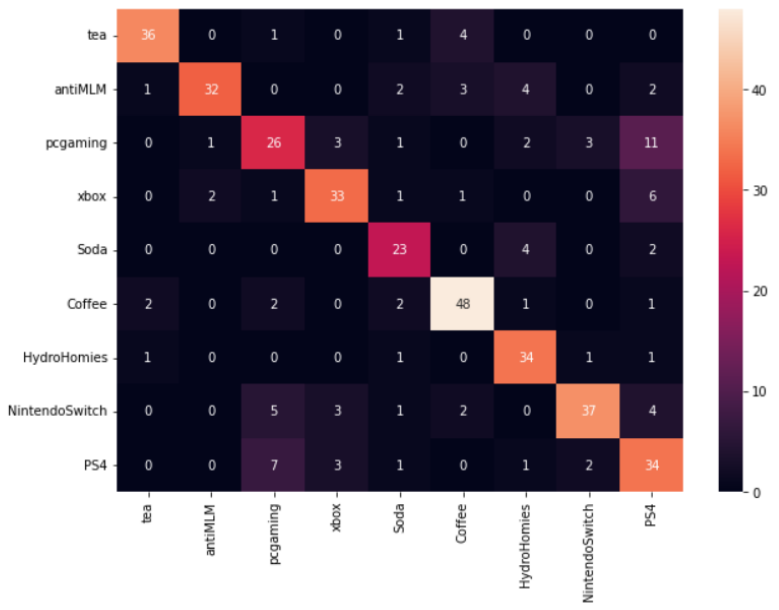


Accuracy Score	0.108
Macro Score	0.109
Weighted Score	0.109
Recall Score	0.108
F1 Score	0.108

LogisticRegression – One-Hot Vectorization

The below Logistic Regression model with use of one-hot vectorization shows a decent fit to the dataset. The confusion matrix shows a trend of correctly assigned labels, with ~25% labels incorrectly assigned.

The similarity of the F1 score (0.759) and the accuracy score (0.758) show the LogisticRegression model with one-hot vectorization is a well-rounded model, with similar levels of precision and recall.



The main parameter used in the Logistic Regression model is the solver, the default is set as lbfgs however there is also newton-cg, liblinear, sag and saga. For smaller datasets liblinear is a good choice whereas for larger datasets sag and saga are faster.

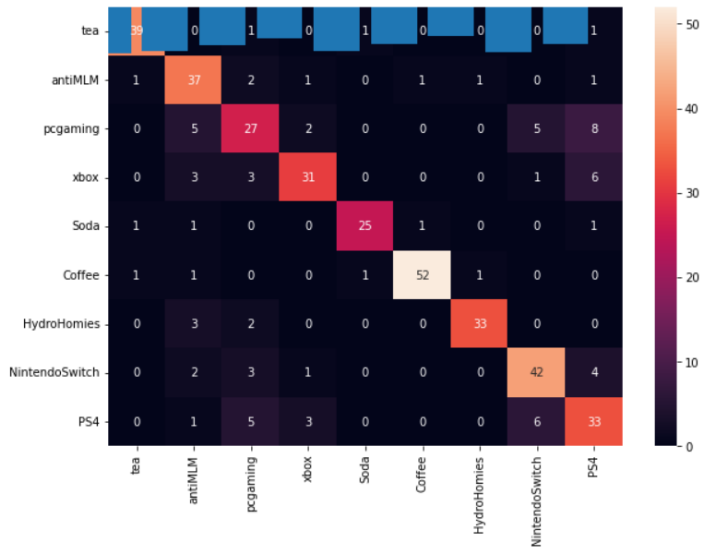
Accuracy Score	0.758
Macro Score	0.767
Weighted Score	0.770
Recall Score	0.758
F1 Score	0.759

LogisticRegression – TF-IDF Vectorization

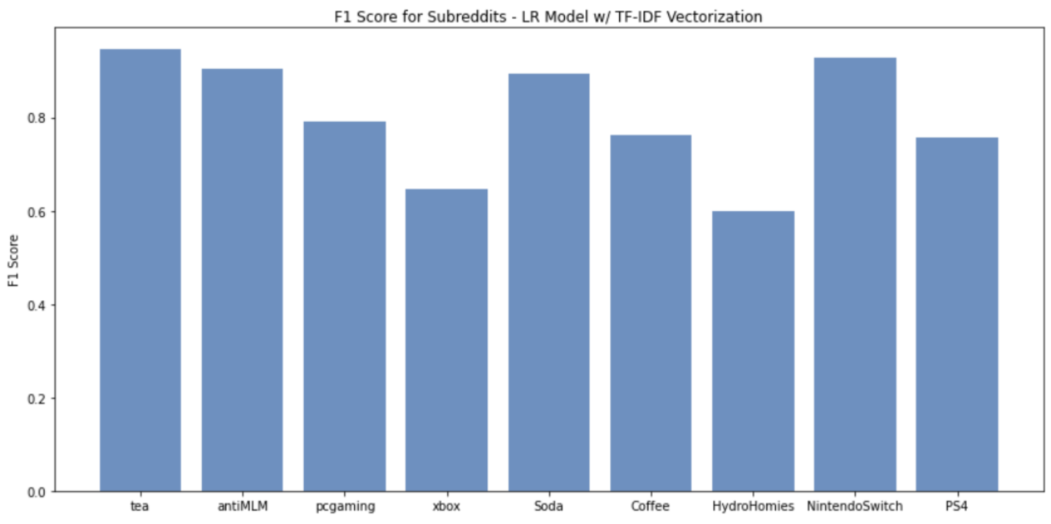
A LogisticRegression model with a TF-IDF vectorization implementation provides the best F1 score with 0.799. This is a ~0.04 improvement in score from the one-hot encoding.

As the TF-IDF model implements the same LogisticRegression classifier, the parameters passed are made up of the same options.

As shown in the confusion matrix below the model performs well with the test dataset, with very few incorrect labels assigned. The graph shows the labels that the model performed best on were r/tea, r/Soda, r/NintendoSwitch and r/antiMLM. The model performed worse with r/xbox and r/HydroHomies posts.



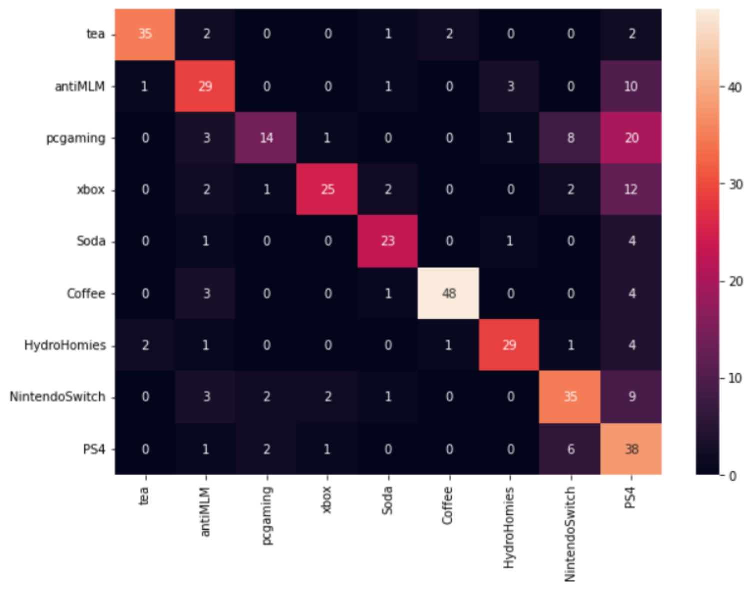
Accuracy Score	0.798
Macro Score	0.810
Weighted Score	0.804
Recall Score	0.798
F1 Score	0.799



SVC Classifier – One-Hot Vectorization

The SVC classifier performed slightly worse than the LogisticRegression model with one-hot vectorization. The classifier typically performs well with larger number of samples, so the accuracy score may be improved had a larger training and test set been used.

The key parameter used in the SVC classifier is the kernel type used in the algorithm. If no type is defined then rbf will be used, however there is also linear, poly, sigmoid and precomputed kernel types.



	tea	antiMLM	pcgaming	xbox	Soda	Coffee	HydroHomies	NintendoSwitch	PS4
tea	35	2	0	0	1	2	0	0	2
antiMLM	1	29	0	0	1	0	3	0	10
pcgaming	0	3	14	1	0	0	1	8	20
xbox	0	2	1	25	2	0	0	2	12
Soda	0	1	0	0	23	0	1	0	4
Coffee	0	3	0	0	1	48	0	0	4
HydroHomies	2	1	0	0	0	1	29	1	4
NintendoSwitch	0	3	2	2	1	0	0	35	9
PS4	0	1	2	1	0	0	0	6	38

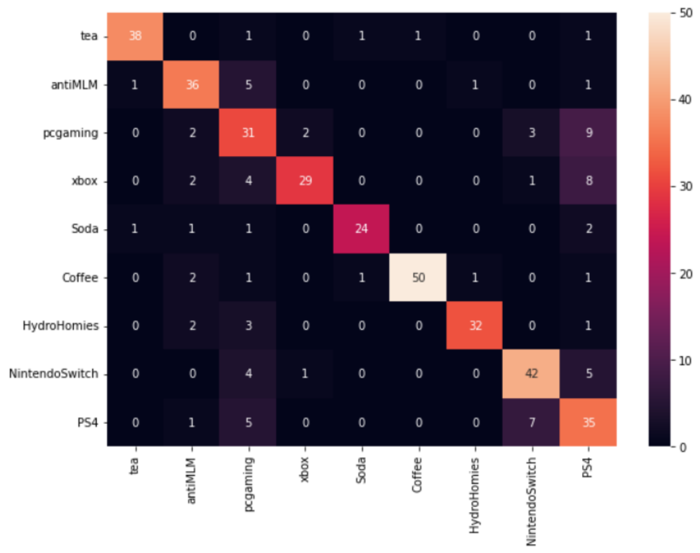
Accuracy Score	0.690
Macro Score	0.755
Weighted Score	0.751
Recall Score	0.690
F1 Score	0.696

Q1c)

SVC Classifier – TF-IDF Vectorization

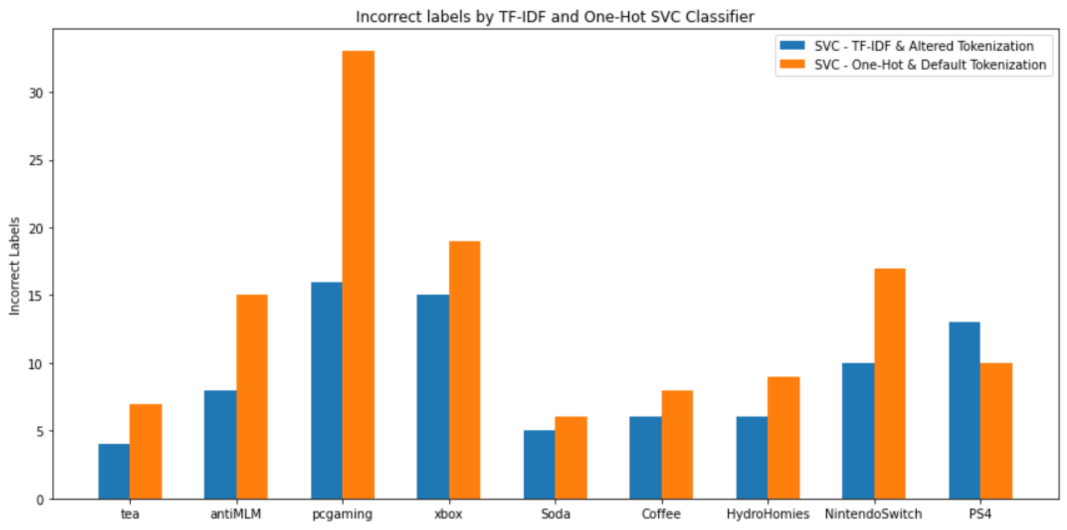
I selected the SVC classifier to base my work off as it is suitable for high dimension classification problems such as this one. As linear regression is more suited for binary classification problems e.g. classifying something as good or bad, I felt SVC was more appropriate. Given that TF-IDF vectorization improved the performance of the LogisticRegression model I felt that it could do the same for the SVC classifier and was proven correct.

I adopted an altered approach to the pre-processing & tokenizing, choosing to restrict output POS tagged as nouns, proper nouns, adjectives, and verbs. I felt this would improve the conciseness of the data being used to fit the models and filter out certain stop words that may have been missed by the existing list. Using the same training and test data, the altered approach increased the SVC classifiers F1 score from 0.696 to 0.799.



Accuracy Score	0.792
Macro Score	0.822
Weighted Score	0.815
Recall Score	0.792
F1 Score	0.799

As shown by the below figure, the addition of TF-IDF vectorization and altered tokenization decreased incorrect labelling for a majority of the subreddits across the test set - dramatically reducing the number for r/pcgaming. The model performed better than the one-hot vectorization approach with a LogisticRegression model, and similarly to the LogisticRegression model with TF-IDF vectorization.



Q2a)

The parameters I tuned were C=1.08, sublinear_tf=True, max_features=2670 and ngram_range=(1,3). Altering these parameters resulted in a F1 score increase from 0.763 to 0.785 across the validation data set.

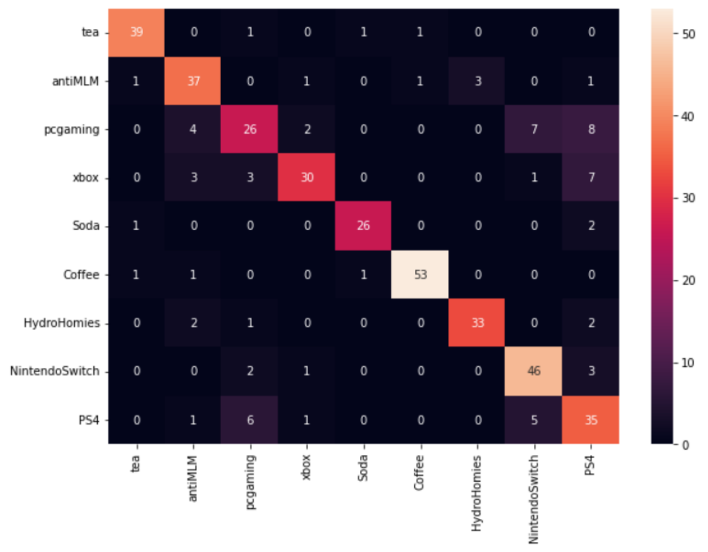
sublinear_tf was an easy parameter to tune as it was a boolean, true or false. Analysing the various scores below showed they all increased when sublinear_tf was enabled. Sub linear tf-scaling scales down the weight of terms that excessively in documents. For example, a word that appears 100x more than another likely does not require 100x the significance across a document. I believe this improved the performance of the model as there may have been words omitted from the stop words list that appeared in the data set and skewed results.

By slowly incrementing C, I concluded that 1.08 was the optimal parameter, as changing values either side resulted in negligible or negative changes in the F1 score. C represents the inverse of regularisation strength - the amount the model reduces the effect of increasing parameter values, where not enough data is available for them to have an effect. I believe this parameter improved performance as the size of the data being classified is relatively small.

With tuning max_features I adopted a similar approach of slowly incrementing the parameter and assessing based on F1 score, resulting in a value of 2670. This results in selecting the top n features ordered by term frequency across the corpus. max_features=10 would result in the 10 most frequent words being selected, whereas a number such as 50000 would result in the data being unchanged, as the vocabulary is not that large.

The parameter I selected to tune was ngram_range. From lecture content I was aware introducing bigrams and trigrams etc. typically increases the performance of models. I found (1,3) - unigrams, bigrams & trigrams - gave the best increase in F1 score. This was the case because n-grams allow for commonly occurring phrases to be paired and predicted when occurring in other documents.

I tried experimenting with other parameters such as the solver used in the model, however any change in solver resulted in no positive changes in F1 score. I found increasing max_iters had no effect on results as the model typically converged before the default 100 iterations regardless. The below figures show the performance of the tuned model on the test features, as well as comparison to the performance of the untuned model.

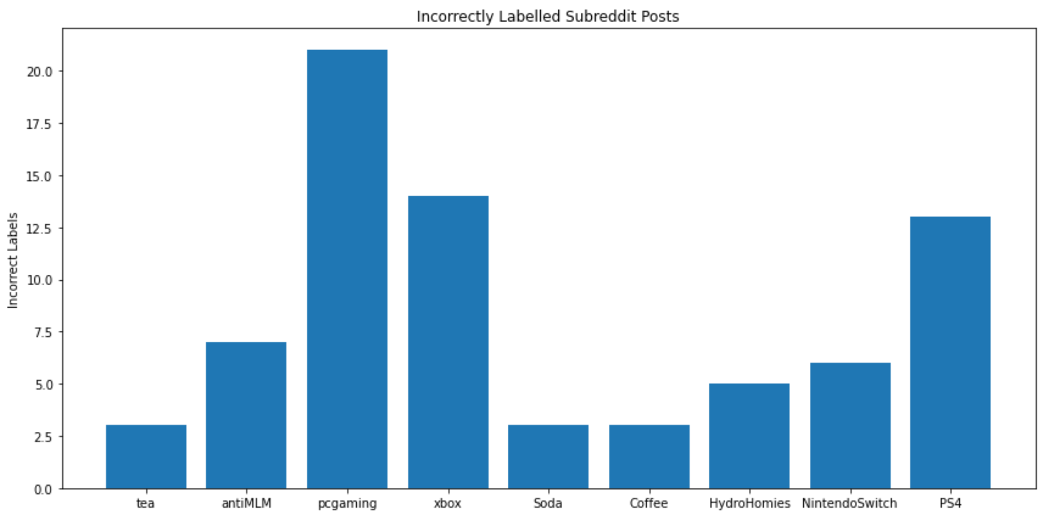


Accuracy Score	0.812
Macro Score	0.824
Weighted Score	0.818
Recall Score	0.812
F1 Score	0.812

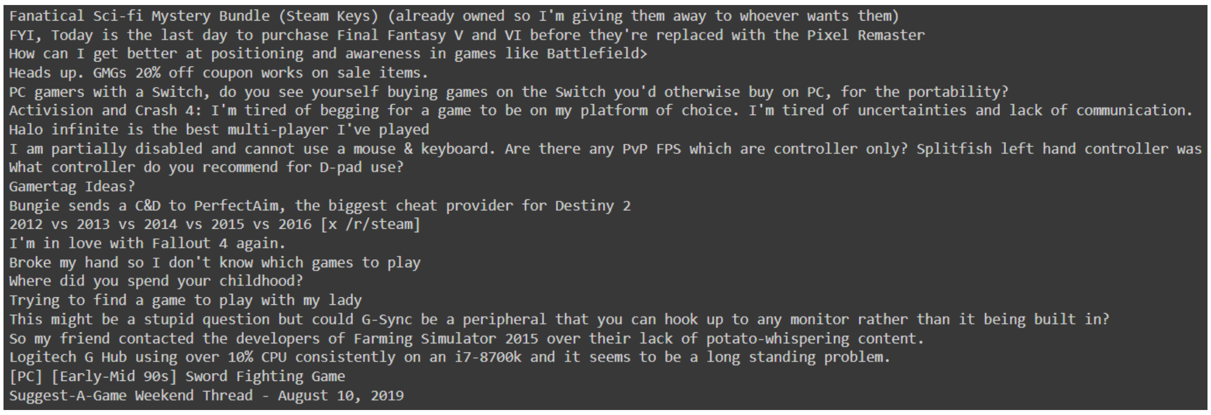
Test Data	**Accuracy**	**Macro**	**Weighted**	**Recall**	**F1**
Untuned	0.798	0.810	0.804	0.798	0.799
Tuned	0.812	0.824	0.818	0.812	0.812

Q2b)

From the confusion matrix, and the bar chart below, it is clear the pcgaming subreddit is occurring with the most false classifications, followed by r/PS4 and r/xbox.



The below posts from r/pcgaming were all incorrectly labelled by the tuned model:



```
Fanatical Sci-fi Mystery Bundle (Steam Keys) (already owned so I'm giving them away to whoever wants them)
FYI, Today is the last day to purchase Final Fantasy V and VI before they're replaced with the Pixel Remaster
How can I get better at positioning and awareness in games like Battlefield>
Heads up. GMGs 20% off coupon works on sale items.
PC gamers with a Switch, do you see yourself buying games on the Switch you'd otherwise buy on PC, for the portability?
Activision and Crash 4: I'm tired of begging for a game to be on my platform of choice. I'm tired of uncertainties and lack of communication.
Halo infinite is the best multi-player I've played
I am partially disabled and cannot use a mouse & keyboard. Are there any PvP FPS which are controller only? Splitfish left hand controller was
What controller do you recommend for D-pad use?
Gamertag Ideas?
Bungie sends a C&D to PerfectAim, the biggest cheat provider for Destiny 2
2012 vs 2013 vs 2014 vs 2015 vs 2016 [x /r/steam]
I'm in love with Fallout 4 again.
Broke my hand so I don't know which games to play
Where did you spend your childhood?
Trying to find a game to play with my lady
This might be a stupid question but could G-Sync be a peripheral that you can hook up to any monitor rather than it being built in?
So my friend contacted the developers of Farming Simulator 2015 over their lack of potato-whispering content.
Logitech G Hub using over 10% CPU consistently on an i7-8700k and it seems to be a long standing problem.
[PC] [Early-Mid 90s] Sword Fighting Game
Suggest-A-Game Weekend Thread - August 10, 2019
```

It is clear that the posts are about gaming when read with human perception but to a classifying model it is not as simple. However, there may be a recurring pattern as to why they are incorrectly classified. From the confusion matrix we can see that pcgaming posts are often incorrectly classified as PS4 or Xbox and vice versa. Posts about different games or games genres can occur in any of three subreddits, as games and genres are available on all three platforms. 'Fanatical', 'Sci-Fi' and 'Mystery' genre games can appear on the PS4, Xbox or PC, just as Dawn of War 3 or Battlefield can be played on all three platforms. The model is making predictions from training data where the above terms likely occur more frequently in other subreddits, hence the post being incorrectly classified.

Subreddits such as soda, tea or coffee are less likely to have this problem, as there are not many cross-over brands that make soda and tea or coffee. Likewise, it is unlikely pc game names will appear in r/antiMLM posts. r/antiMLM posts may have been classified incorrectly as the subreddit has no clear distinguishable topic. Subreddits such as tea and coffee, however, are much more defined.

Lewis Ponsonby 2464980P

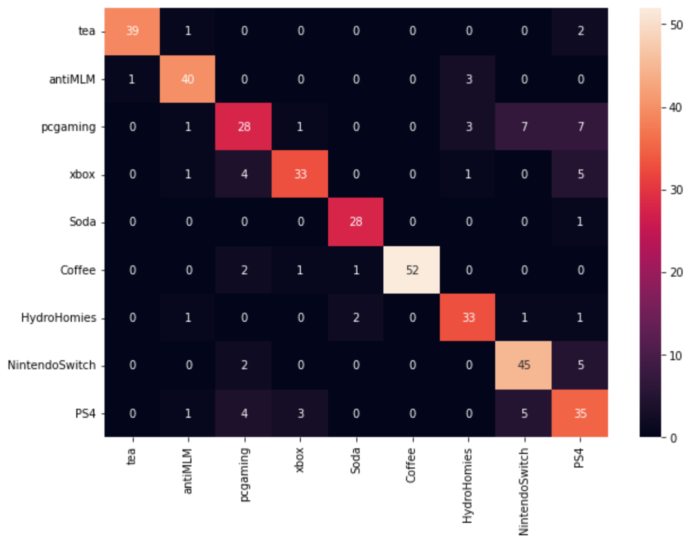
Q3a)

The first feature I would propose would be the implementation of a FeatureUnion. This would allow for the concatenation of posts' title and body text. Many of the posts have small bodies of text, making it hard for the classification model to assign them correct labels on that text alone. Combining the title and the body would allow for longer documents and improved modelling. Implementing this with scikit-learn's FeatureUnion would allow for weighting to be used e.g. assigning the title of a post more relevance than the body of text.

The second feature I would propose is the use of a voting classifier with the original logistic regression model as one of the estimators. Different types of models perform differently with the same dataset and introducing a voting classifier would allow for cross referencing and selecting the classification that the majority of the models select.

Q3b)

The model with the two features implemented performed better than all previous models, with an F1 score of 0.833.



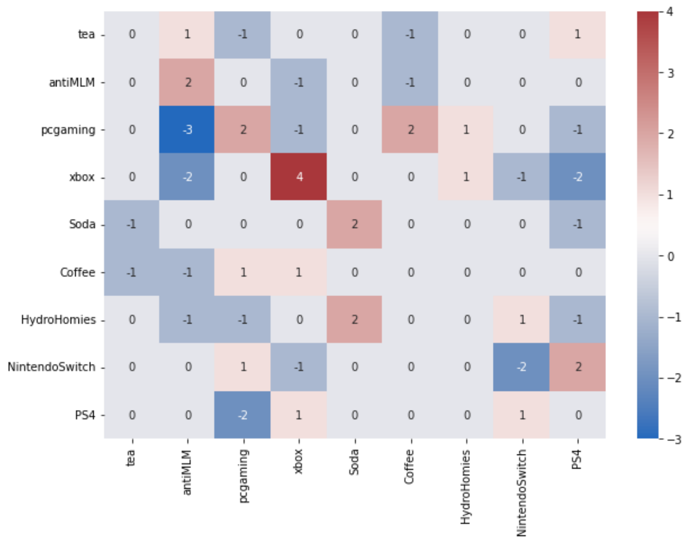
Accuracy Score	0.832
Macro Score	0.840
Weighted Score	0.838
Recall Score	0.832
F1 Score	0.833

Q3c)

Comparing data in the table below, the model using the engineered features performed better in all five evaluation metrics used.

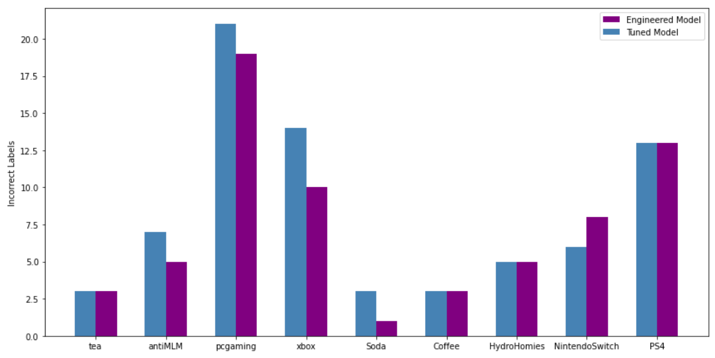
Test Data	**Accuracy**	**Macro**	**Weighted**	**Recall**	**F1**
Tuned	0.812	0.824	0.818	0.812	0.812
Engineered	0.832	0.840	0.838	0.832	0.833

From the confusion matrix the improvement of classification from the tuned model to the engineered model can be seen. There is an increase in correctly assigned labels for antiMLM, pcgaming, xbox and soda. Only r/NintendoSwitch was assigned worse with the engineered model, however overall, there was 8 extra correctly assigned labels.

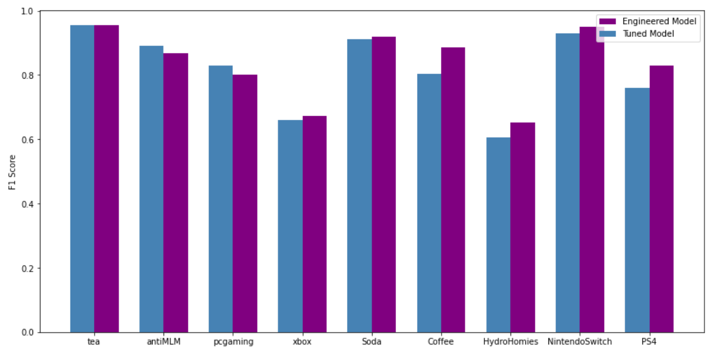


	tea	antiMLM	pcgaming	xbox	Soda	Coffee	HydroHomies	NintendoSwitch	PS4
tea	0	1	-1	0	0	-1	0	0	1
antiMLM	0	2	0	-1	0	-1	0	0	0
pcgaming	0	-3	2	-1	0	2	1	0	-1
xbox	0	-2	0	4	0	0	1	-1	-2
Soda	-1	0	0	0	2	0	0	0	-1
Coffee	-1	-1	1	1	0	0	0	0	0
HydroHomies	0	-1	-1	0	2	0	0	1	-1
NintendoSwitch	0	0	1	-1	0	0	0	-2	2
PS4	0	0	-2	1	0	0	0	1	0

As shown in the figure below, the engineered model reduced the number of incorrectly assigned labels for all but one of the subreddits. Given the majority of incorrectly assigned labels belonged to pcgaming, PS4 and xbox, improving the accuracy of the three was key in getting a model with higher performance. The chart shows that pcgaming and xbox both reduced in incorrect labels however they still ranked with the lowest accuracy alongside PS4. I believe this problem is difficult to solve with any implemented features, as certain words such as games and genres, will always belong to multiple classes. I think increasing the size of the training data set would be the most effective method to improve the classification of these subreddits.



From analysing the F1 scores in the figure below, it is clear the model struggled classifying r/HydroHomies correctly. Posts often being tagged with HydroHomies when they are not, or not being tagged with HydroHomies when they are. I believe the incorrect classifications derive from the lack of a distinct topic throughout the training and test data. A selection of the most successful subreddits in terms of F1 score include r/tea, r/soda and r/coffee. These subreddits all have distinguishable topics, with common keywords that the model can use to classify in future instances. Words such as cappuccino, espresso and latte will appear in coffee posts, brands such as Coke, Sprite and Fanta will appear in soda posts. Although this problem cannot be directly fixed, it can be improved with increasing the size of the training dataset as logistic regression models can tend to overfit on very sparse data.


Engineered Model
Tuned Model
F1 Score
tea
antiMLM
pcgaming
xbox
Soda
Coffee
HydroHomies
NintendoSwitch
PS4