Comp1804 Report

24th March, 2025

Word count: 2626

Executive summary.

Task 1: Topic Classification:

A model was developed so that it could help automatically predict the 7 classifications under petition_topic based on the usage of petition_text and/or has_entity as features. A pipeline was developed to fit the Gradient Descent classifier and it achieved ~89% accuracy on the test dataset, which meets the client's requirement. There was no overfitting on the model and most topics stayed within the misclassification threshold mentioned. Also, the model was able to identify the topic 'UK Government and Devolution' with less than 9% error as required, satisfying the required conditions.

Task 2: Predicting Petition Importance:

100 rows were manually labelled rows to train a supervised model being the same SGDClassifier from the previous task. The models were able to predict the labels at an accuracy of >65% which is better than the dummy classifier giving an accuracy of ~50% using random values, which helps meet the client's requirements of a scalable prototype. The model was able to label the unlabelled values as well and the exported CSV is available for viewing.

1. Data exploration and assessment.

The analysis began by checking the dataset's structure, consistency and its completeness. The key issues noticed were:

- **Missing Values**: Noticed that 'relevant_department' was missing some values for some of the petitions, 'deviation_across_regions' had missing numeric values as well and the 'petition_importance' which was the topic to predict in task 2 had over 8000 missing values.
- **Case Sensitivity Issues**: The 'petition_topic' field had some values which were incorrectly classified because of case sensitivity on the first letter, which was corrected.

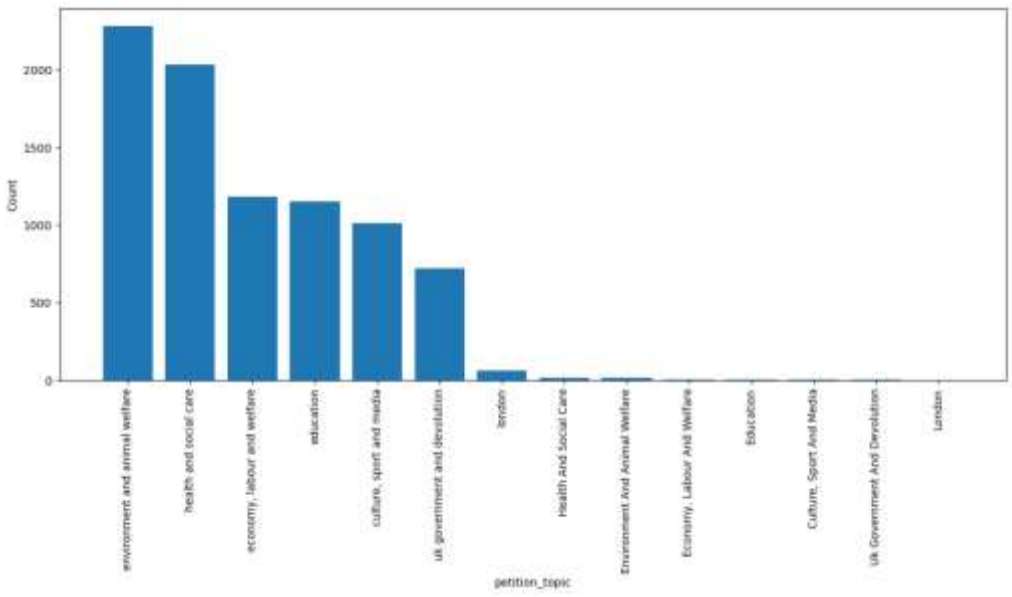

Figure 1 Petition_topic split

- **Duplicate Records**: Duplicated rows were minimal but they do exist in the dataset.

The other features like has_entity and petition_text helped providing insights on the patterns that were required to be developed to classify the target labels.

2. Data splitting and cleaning.

In Task 1, the dataset was focused primarily on the 'has_entity' and/or 'petition_text' as the features and 'petition_topic' as the target label. So, the train test split was performed in an 80:20 manner prior to the data being cleaned. This allowed the test data to not have any leakage that might cause a potential bias on the overall outcome. Allowing us to test the supervised model for accuracy.

In Task 2 the column 'petition_importance' where 100 manually labelled rows were added was used. On which, a train-test split of 80:20 was performed, allowing the training and testing to be performed on the supervised classifier.

The cleaning process followed was:

- Removal of duplicates as mentioned in the exploration.
- Fixing the case sensitivity issues
- Imputing missing values in 'deviation_across_regions' column with its mean and the 'relevant_department' as 'Unknown'.
- The 'petition_text' field was then subjected to be lemmatised and then tokenized based on the functions which were discussed during class. In certain cases, it was embedded and rest it was tokenized to meet requirements.

3. Data encoding.

Text and numerical data were encoded to prepare the data for machine learning models:

- 'petition_text' was cleaned using spaCY was lemmatized, these tokenized results were passed through the tfidfVectorizer in some cases to help process non-neural network supervised models which were used, whereas when neural based classifiers like the MLPClassifier was used in task 1, the field was word-embedded with defined functions and then the MinMaxScaler was applied on it. For task 2, the tokenized list of words was retained for the models.
- 'has_entity' was subjected to various encoding practises in the coursework, some cases it was subjected to a direct LabelEncode, it was also passed through a function to split the entity values into 3 columns 'has_event', 'has_date' and 'has_person' via a python function to store the value as true or false based on the entity value as a whole.
- 'relevant_department' was subjected to OneHotEncoding practice in the 2nd task for both the models.
- 'text_length' this field was retrieved from the overall clean_text length was subject to standardization.
- 'petition_topic' was subject to LabelEncoding as well to retrieve target labels for the models in task 1.

4. Task 1: topic classification.

4a. Model building.

The topic of the first task which the client wanted to focus on being "whether it is possible to use machine learning to identify the topic of a petition, for some specific topics of interest" and based on trials and errors experienced while developing the model it is definitely possible to develop a supervised machine learning model to predict specific topics using feature encoding and model training.

The process to find the perfect model began by using a neural network approach with the MLPClassifier and its features where the 'petition_text' column was subjected to being word embedded for the model and to test the outcome, 'has_entity' was not used as a feature. The first set of hyperparameters which were used to grid search across 5 cross-validation folds are as follows:

Component	Parameter	Value
MLPClassifier	hidden_layer_sizes	(20,20), (10,10), (30,30)
MLPClassifier	alpha	0.0001, 0.001,0.01
MLPClassifier	max_iter	400, 600, 1000

The model ran for over 1806.24 seconds and it returned an accuracy of only **78%** with a lot of overfitting and the scores under the performance metric was not meeting the conditions either.

On the second attempt along with RandomOverSampler(*RandomOverSampler*, 2024) was utilized to stable the imbalance caused in the classes and the feature 'has_entity', which was label encoded, was included alongside the word embedded 'petition_text' to see if there would be significant impact on the neural network model's performance using gridsearch and the hyperparameters which were chosen for this were as follows:

Component	Parameter	Value
MLPClassifier	hidden_layer_sizes	(200,20), (200,)
MLPClassifier	alpha	0.00001, 0.0001, 0.001
MLPClassifier	max_iter	600

The model now ran its longest run yet of 9745.64 seconds, which is not good and unfortunately adding the has_entity field did not result in much of an improvement as the model still returned an accuracy of 81% without any overfitting, but unfortunately the client's conditions were not met.

On the third attempt, the has_entity column as a feature and the word embedding for neural network model's approach was dropped together and a simpler approach was picked up to try and use the LogisticRegression model to get a positive outcome. In this case, the tokenized version of the

‘petition_text’ was used alongside the gridsearch on the following complex set of hyperparameters:

Component	Parameter	Value
TfidfVectorizer	ngram_range	(1,1), (1,2)
TfidfVectorizer	max_features	2000, 25000
TfidfVectorizer	min_df	1, 2
TfidfVectorizer	max_df	0.5, 1.0
LogisticRegression	C	1
LogisticRegression	class_weight	'balanced',None
LogisticRegression	penalty	l2, l1
LogisticRegression	solver	saga

This model ran for over 8372.78 seconds and in overall returned the best accuracy score of **87%** with <1% overfitting. The reason for utilizing some of the above parameters are that solver ‘saga’ allows processing multinomial multilabel data with l2 and l1 regularization which is not possible with the default solver (*scikit-learn*, 2024).

The performance metrics on this model was great for the misclassification requirement but unfortunately failed on the <9% error on precision requirement when it comes to the requirement on the topic – ‘uk government and devolution’.

On the fourth attempt, which was the best out of all of them, the SGDClassifier model was utilized with hinge loss (so a Linear Support Vector Machine (SVM) model) to process the data using gridsearch with the following hyperparameters:

Component	Parameter	Value
TfidfVectorizer	ngram_range	(1,1), (1,2)
TfidfVectorizer	max_features	2000, 25000
SGDClassifier	penalty	l2
SGDClassifier	alpha	0.0001

The justification behind choosing this model is because “Text implementation works with data represented as dense or sparse arrays of floating-point values for the features. The model it fits can be controlled with the loss parameter; by default, it fits a linear support vector machine”(*scikit-learn*, 2024). It is effective with the sparse matrix output from the tfidf vectorizer. The inclusion of unigrams and bigrams helped improve the contextual understanding and capturing phrases. Increase in the max features allows the model to understand the data better, whilst preventing a possible underfit form happening. Thus, having fewer hyperparameters from before allowed the model to complete in under 294.13 seconds and return an accuracy of **~89%** which is above the threshold set by the client while the misclassification and the error on precision for a topic was met under the threshold as well

4b. Model evaluation.

The client's evaluation focused on the following points:

- Test accuracy >= 86%
- No Significant overfitting (between test and training accuracy)
- No more than 13% misclassified topics in at least 5 out of 7 classes
- Less than 9% incorrect predictions for "UK Government and Devolution" class

The overall test accuracy was found to be ~89% and the training accuracy was found to be ~90% which does not suggest overfitting, suggesting the model meets client's first and second requirement.

The classification report seen below shows the performance metric score for each of the classes:

Topic	Precision	Recall	F1-score	Support
Environment and Animal Welfare	0.90	0.88	0.89	203
Health and Social Care	0.85	0.91	0.88	238
Economy, Labour and Welfare	0.91	0.86	0.88	232
Education	0.95	0.93	0.94	460
Culture, Sport and Media	0.87	0.91	0.89	409
UK Government and Devolution	1.00	0.42	0.59	12
London	0.83	0.82	0.83	144

Metric	Score
Accuracy	0.90
Macro Avg	0.84
Weighted Avg	0.9

The Recall for condition for misclassification for 5 out of 7 classes is also met, along with the precision value for "UK Government and Devolution", meaning all 4 conditions from the client are met with this model.

The confusion matrix below also shows how majority of the predictions fall along the diagonal, showcasing the model's accuracy.

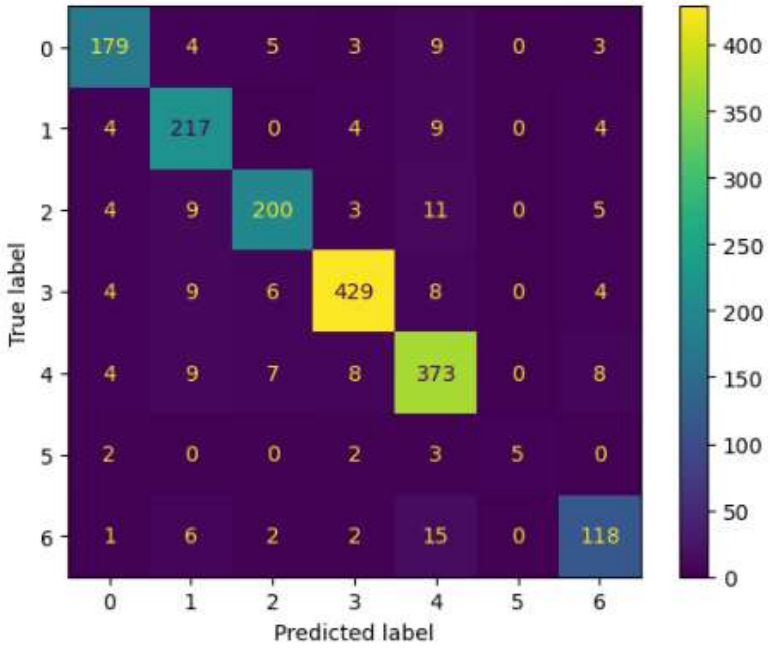

Figure 2 Task 1 Final Model Confusion Matrix

4c. Task 1 Conclusions.

This brings an end to Task 1 and the conclusion is as follows:

- As per the evaluation above, all 4 criteria mentioned have been met by the model, with more fine tuning and data influx, we can improve the model's accuracy.
- As the classes are imbalanced, the best metric to consider is the **macro average score** as it provides equal weightage to all classes irrespective of the sample size, so the scores are reflected equally.

5. Task 2: petition importance classification prototype.

5a. Ethical discussion.

The task raises several ethical concerns because of its potential influence on a political level and at a public level as well. The following 'Data Hazard Labels' (*Data Hazard labels*, 2025)are relevant to the task:

- **Ranks or Classifies People:** The model inherently decides if a petition is important or not irrespective of the impact it might have on the people that are at the receiving end of the decision, the people who might have fewer representation could potentially be on the de-prioritization of their petitions which is not right.
- **Lacks Community Involvement:** Petitions in fact are raised by and for a common group of people who are facing an issue and want an immediate outcome for it, but if the model is not designed to accept this group as a data source before making a decision on its importance, then it fails to reflect a broad understanding of the issue at hand.
- **Automates Decision Making:** The model is being designed only as a support tool for the client to help decide an outcome for the one's in power. However, the decision the model makes should never be set as the final fact, as it might have an impact on the petition which might not be prevalent at the time but is likely to be a major issue in the future. This leads to potentially unfair outcomes.

While the model is developed to help the client attain a system that helps make a decision, it should only be used as a tool to meet a requirement and not be the final say in a democratic situation

5b. Data labelling.

The labelling process that was implemented was purely based on intuition and on the relevancy of the matter at the current date. Petitions were marked as "important" for petitions having an impact on large groups or political reasons. The text was the factor that was personally considered so its relevancy in the current scenario played a major role in the matter being marked as important or not. Hence, petitions which showed lower urgency in comparison were marked as "not important".

While the process was judged as fairly as possible, it does raise a point that the final outcome was not judged by an authority or a group of people, hence leading to bias in situations. This introduces an element of "Rank or Classifying People" without regarding for the impact this might have on the people on the receiving end of the petition. Regardless, it helps provide a dataset to train a model to predict rest of the data on.

5c. Model building and evaluation.

The goal was to build a prototype model that would be able to classify models as “important” or “not important” based on the text and the features. So, a supervised learning model, using the SGDClassifier model was developed

The features that were used for both the models involves the following columns:

- **Petition_text**, as it holds context rich information that can be used by tokenizing and transforming the data using the TfidfVectorizer.
- **Relevant_department,** although it holds some missing data, the remaining department names were One-Hot Encoded and used as a feature, after the missing values were filled as ‘UNKNOWN’
- **Has_entity,** this column was split into 3 types being has_event, has_date, has_person which was then used as binary data source, i.e., holding 1’s and 0’s to be used in the classifier.
- **Deviation_across_region,** this numeric data source was used as is to train the model, after imputing the missing values with the mean values across the table.

For the Supervised Learning model, after the cleaning was performed, the data is split 80:20 using the train and test split function, a pipeline was configured with the Tfidf vectorizer for the petition_text and the SGD Classifier. The hyperparameter for the model and vectorizer are as follows:

Component	Parameter	Value
TfidfVectorizer	ngram_range	(1,1), (1,2)
TfidfVectorizer	max_features	2000, 25000
SGDClassifier	penalty	L2
SGDClassifier	alpha	0.0001

For this model, 100 labels were manually added into the column and then the used to train the model. The results from testing the model showed a 65% accuracy with minimum overfitting. The classification report is as follows:

Topic	precision	recall	f1-score	support
Important	0.64	0.7	0.67	10
Non-Important	0.67	0.6	0.63	10

Metric	Score
Accuracy	0.65
Macro Avg F1	0.65
Weighted Avg F1	0.65

The confusion matrix also aligns saying the model has the majority accuracy on the true values:

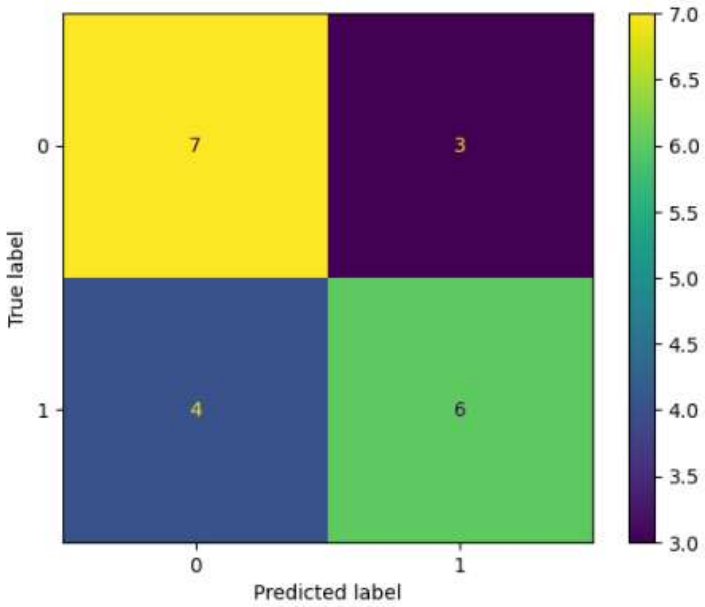

Figure 3 Task 2 Final Model Confusion Matrix

Upon checking with a dummy classifier, the model generated only 45% accuracy, showing that the model developed has been accurate comparatively.

5d. Task 2 Conclusions.

You should provide three short bullet points with the following:

- The model is successful as it shows ~65% accuracy to a supervised model. However, there were no signs of overfitting and the model seems to performing well as a prototype.
- The task depends on the value of importance or non-importance being classified correctly; without any bias it is difficult to perform. As the labels were manually filled first and then a model was developed on to predict. This individual judgement is difficult to confirm, so it is a difficult and unstable target to automate.
- Improving the feature set can help in incorporating real world signals like petition signatures, number of people who have signed it, i.e., engagement, etc. Additionally, having more labelled values can help improve the performance of the model.

6. Self-reflection.

The hardest part of this coursework was while attempting the first task as, that required a lot of patience to choose which classifier the work had to be based on. More learning is required to understand which classifier will excel on which so as to not waste a lot of time.

7. References.

Data Hazard labels (2025) https://datahazards.com/labels.html. (Accessed: 16th March, 2025)

RandomOverSampler (2024) https://imbalancedlearn.org/stable/references/generated/imblearn.over_sampling.RandomOverSampler.html. (Accessed: 20th March, 2025)

scikit-learn (2024) https://scikit-learn.org/stable/index.html. (Accessed: 20th March, 2025)

Appendix A: Declaration of AI Use

I have used AI while undertaking my assignment in the following ways:

- To develop research questions on the topic – YES
- To explain concepts – YES
- To support my use of language – YES

In other ways, as described below:

To validate if the points or ideas I am conveying the essay are portrayed in the right tone and format.