**COMP1804 – Applied Machine Learning**

**2023 – 2024: Term 2**

**Applied Machine Learning Coursework**

**Task Report**

**Nneka Ndubishi**

**MSc Data Science**

Contents

[Executive Summary 2](#_Toc162310073)

[1. Data exploration and assessment. 2](#_Toc162310074)

[2. Data splitting and cleaning. 3](#_Toc162310075)

[3. Data encoding 4](#_Toc162310076)

[3.1 Label Encoding 4](#_Toc162310077)

[Task 1: topic classification 5](#_Toc162310078)

[4a. Model building. 5](#_Toc162310079)

[4b. Model evaluation. 6](#_Toc162310080)

[4c. Task 1 Conclusions. 8](#_Toc162310081)

[Task 2: text clarity classification prototype 8](#_Toc162310082)

[5a. Ethical discussion 8](#_Toc162310083)

[5b. Data labelling 8](#_Toc162310084)

[5c. Model building and evaluation 9](#_Toc162310085)

[5d. Task 2 Conclusions 11](#_Toc162310086)

[6. Self-reflection 11](#_Toc162310087)

[References 12](#_Toc162310088)

Table of figures

[**Table 1: Categorical labels as numerical values** 4](#_Toc162310159)

[**Table 2: Hyperparameters with values and detailed descriptions** 5](#_Toc162310160)

[**Table 3: Confusion Matrix for Class model** 6](#_Toc162310161)

[**Table 4: Classification Report for Class model** 6](#_Toc162310162)

[**Table 5: Baseline Confusion Matrix** 7](#_Toc162310163)

[**Table 6: Baseline Classification Report** 7](#_Toc162310164)

[**Table 7: Hyperparameters with values and detailed descriptions** 10](#_Toc162310165)

[**Table 8: Classification report for text clarity model** 10](#_Toc162310166)

[**Table 9: Confusion Matrix for text clarity model** 10](#_Toc162310167)

[**Table 10: Baseline Classification report** 10](#_Toc162310168)

# Executive Summary

In this report, the tasks given by the client were successfully performed with visual results. Task 1 involved employing a word embedding model using spaCy and Logistic Regression for topic classification. Task 2 centred on developing a prototype for text clarity classification, TF-IDF vectorisation and Logistic Regression. There were ethical concerns addressed in Task 2 with a labelling process outlined. Despite meeting the clients requirements, further refinement could have taken place.

# Data exploration and assessment.

|  |  |
| --- | --- |
| Columns | Data Preprocessing Steps |
| par\_id | Removed duplicates to ensure each paragraph text has a unique identifier. |
| paragraph | Removed duplicates and integers from the text data to maintain data integrity. |
| has\_entity | Removed entries labelled as “data missing” to ensure only actual entities are present. |
| lexicon\_count | No specified preprocessing steps mentioned. |
| difficult\_words | No specified preprocessing steps mentioned. |
| last\_editor\_gender | No specified preprocessing steps mentioned. |
| Category | Converted all categories to lowercase to address inconsistencies in capitalisation. |
| text clarity | No specified preprocessing steps mentioned. |

Class Distribution was also performed to reveal any imbalances in the dataset. Figure 1 below reveals that ‘biographies’ had the most amount of entries within the dataset and ‘movies about artificial intelligence’ had the least. This disparity in the data can lead to biases during model training and may have impacted the model’s ability to generalise well to minority classes, such as ‘movies about artificial intelligence’.

A graph of blue rectangular bars with white text

Description automatically generated

**Figure 1: A bar chart illustrating the distribution of categories in the dataset.**

Figure 2 below illustrates the category distribution after the oversampling technique. Figure 1 shows a huge disparity in the minority class so an under-sampling technique wouldn’t have worked. There is now an increased the number of minority class samples.

A graph with blue and white bars

Description automatically generated

**Figure 2: A bar chart illustrating the Class Distribution after Oversampling the data**

# Data splitting and cleaning.

1. After the initial data preprocessing, the dataset was split into two dataframes: ‘X\_dataset’ containing the ‘has\_entity’ and ‘paragraph’ columns and ‘Y\_predict’ containing the ‘category’ column for classification. The reasoning for this is the ‘has\_entity’ feature indicates that the ‘paragraph’ data contains references to relevant entities.
2. Before splitting, ‘has\_entity’ was split into three columns, representing their entity using binary encoding for presence indication, aiding in machine interpretation.
3. Stratified split method was utilised due to imbalanced datasets. The oversampling technique applied to the only the training set to enhance generalisation.
4. Exploratory Data Analysis (EDAs) revealed the most frequent entity occurrence is when organisations and persons are simultaneously referenced, but not products.

# Data encoding

In the training dataset, columns labelled ‘has\_entity’, ‘paragraph’, ‘ORG’, ‘PRODUCT’ and ‘PERSON’ are present whilst the ‘has\_entity’ column is dropped because it duplicates information presented in the separate entity columns. The binary representation in the entity columns aids the machine by indicating which entities are associated with the textual data in the ‘paragraph’ column.

Data encoding consists of text encoding the textual data by tokenising, lowercasing, stop word and punctuation removal, lemmatisation, stop words, rare words and misspellings.

After text processing, ‘paragraph’ data is converted into word embeddings using spaCy pre-trained word vectors. This process involves utilising the semantic information encoded in the word vectors.

TF-IDF matrix was the alternative technique, it quantifies the importance of terms in a document based on their frequency and rarity. However, its inability to capture semantic relationships posed an issue. Therefore, using spaCy word vectors felt the most appropriate for this task.

## Label Encoding

The training dataset is prepared machine learning modelling, now the categorical labels are converted into numerical values. This process is done through label encoding, the mapping of the categorical labels to numerical values as follows:

**Table 1: Categorical labels as numerical values**

|  |  |
| --- | --- |
| Former categorical labels | Numerical values |
| artificial intelligence | 0 |
| biographies | 1 |
| movies about artificial intelligence | 2 |
| philosophy | 3 |
| programming | 4 |

The data is split into training, validation and test data using sklearn’s train\_test\_split. It is revealed that the Training set size is 7974, the Validation set size is 1709 and the Test set size is also 1709. This follows the 70:15:15 ratio for training, validation and test sets which ensures reliability as it is a larger training set.

# Task 1: topic classification

## 4a. Model building.

For feature selection, the approach used involved the binary, spaCy word vectors and label encoding that was implemented from the data preprocessing stage.

Logistic Regression is chosen for modelling because of its simplicity and effectiveness when dealing with the multi-class text classification and allows easy interpretation of feature to class relationships. Support Vector Machines (SVMs) is an alternative algorithm that was considered, their advantage of handling high dimensional data efficiently which makes it highly suitable. However, their computational complexity when dealing with larger datasets poses a problem, it can lead to longer training times compared to Logistic Regression.

A Grid Search technique was then implemented to identify what values for the hyperparameters are most appropriate for the model. The table below shows the results of the values along with their descriptions on why the choice was made.

**Table 2: Hyperparameters with values and detailed descriptions**

|  |  |  |
| --- | --- | --- |
| Hyperparameter | Value | Description |
| C | **0.01** | Controls the strength of regularisation in Logistic Regression. 0.1 indicates a stronger regularisation as it is a low value but at the cost of increased bias. |
| max\_iter | **100** | Maximum number of iterations for the solver to converge. The value of 100 is chosen to allow sufficient iterations for convergence without risking overly long training times. |
| penalty | **‘l2’** | Type of regularisation penalty. ‘l2’ refers to L2 regularisation. This regularisation helps to prevent overfitting by penalising large coefficients. |
| solver | **‘lbfgs’** | This is a optimisation algorithm. ‘lbfgs’ is a good choice for small datasets. |

Random Search technique is also effective for exploring large and complex search spaces. However, the hyperparameter search space is relatively small and straightforward so using Random Search might not be the most efficient approach.

In conclusion, the approach displays the significance of Logistic Regression in multi-class text classification due to its simplistic and effective nature. Through hyperparameter optimisation, a Grid Search was performed to fine tune the model’s performance and the chosen hyperparameters were selected to balance the model complexity.

## 4b. Model evaluation.

The tables below provide an evaluation of the model’s performance using a confusion matrix and a classification report.

For clarification, Class 0 is ‘artificial intelligence’, Class 1 is ‘biographies’, Class 2 is ‘movies about artificial intelligence’, Class 3 is ‘philosophy’ and Class 4 is ‘programming’.

**Table 3: Confusion Matrix for Class model**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Predicted Class 0 | Predicted Class 1 | Predicted Class 2 | Predicted Class 3 | Predicted Class 4 |
| **Actual Class 0** | 172 | 35 | 38 | 27 | 79 |
| **Actual Class 1** | 49 | 144 | 64 | 46 | 22 |
| **Actual Class 2** | 32 | 89 | 182 | 30 | 14 |
| **Actual Class 3** | 108 | 52 | 33 | 63 | 75 |
| **Actual Class 4** | 131 | 10 | 58 | 33 | 123 |

The Confusion Matrix summarises the model’s performance by showing the number of correct and incorrect predictions with count values broken down by each class (Brownlee, 2020). The correct predictions appear as diagonal elements from the top left to the bottom right. The bullet points below highlight the model’s evaluations:

* 108 paragraphs labelled as ‘philosophy’ **(Class 3)** were mistakenly classified as ‘artificial intelligence’.
* 131 paragraphs labelled as ‘programming’ **(Class 4)** were misclassified as ‘artificial intelligence’ **(Class 0).**

**Table 4: Classification Report for Class model**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | Precision | Recall | F1-Score | Support |
| **0** | 0.35 | 0.49 | 0.41 | 351 |
| **1** | 0.44 | 0.44 | 0.44 | 325 |
| **2** | 0.49 | 0.52 | 0.50 | 347 |
| **3** | 0.32 | 0.19 | 0.24 | 331 |
| **4** | 0.39 | 0.35 | 0.37 | 355 |
| **Accuracy** |  |  | 0.40 | 1709 |
| **Macro Avg** | 0.40 | 0.40 | 0.39 | 1709 |
| **Weighted Avg** | 0.40 | 0.40 | 0.39 | 1709 |

A Classification Report provides a summary of the model’s performance metrics for each class, it includes precision, recall, F1-score and support. Precision measures the accuracy of positive predictions made by the model, whereas recall assess the model’s ability to identify all positive predictions and F1-score balanced the precision and recall. Lastly, support is the number of actual occurrences of the class in the dataset. (Kharwal, 2021).

The macro average and weighted average provides an overall assessment of the performance metrics.

To summarise the Classification report, the model’s ability to correctly identify instances of Class 3 (philosophy) from all actual instances is relatively low as it is only 19%. Furthermore, Class 2 (movies about artificial intelligence) has the highest scores for each metric, this indicates that the model is able to identify this specific class more accurately and effectively more than the other classes.

**Table 5: Baseline Confusion Matrix**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Predicted Class 0 | Predicted Class 1 | Predicted Class 2 | Predicted Class 3 | Predicted Class 4 |
| **Actual Class 0** | 0 | 351 | 0 | 0 | 0 |
| **Actual Class 1** | 0 | 325 | 0 | 0 | 0 |
| **Actual Class 2** | 0 | 347 | 0 | 0 | 0 |
| **Actual Class 3** | 0 | 331 | 0 | 0 | 0 |
| **Actual Class 4** | 0 | 355 | 0 | 0 | 0 |

**Table 6: Baseline Classification Report**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Class | Precision | Recall | F1-Score | Support |
| **0** | 0.00 | 0.00 | 0.00 | 351 |
| **1** | 0.19 | 1.00 | 0.32 | 325 |
| **2** | 0.00 | 0.00 | 0.00 | 347 |
| **3** | 0.00 | 0.00 | 0.00 | 331 |
| **4** | 0.00 | 0.00 | 0.00 | 355 |
| **Accuracy** |  |  | 0.19 | 1709 |
| **Macro Avg** | 0.04 | 0.20 | 0.06 | 1709 |
| **Weighted Avg** | 0.04 | 0.19 | 0.06 | 1709 |

To assess the model’s effectiveness, a baseline report is using the majority class concept is used. This acts as an extra reassurance to see if the model created significantly outperforms a trivial approach and the client reiterated that our model shouldn’t overfit the training data and that misclassifications into unrelated classes are minimised.

In conclusion, it is important to recognise that there are limitations when interpreting the figures as these could include things like the uneven distribution of classes.

## 4c. Task 1 Conclusions.

* Model’s success aligns with the clients definition as it surpasses the trivial baseline, demonstrates minimal overfitting and less than 10% of the paragraph texts are misclassifying “artificial intelligence” as “programming”.
* Matthews Correlation Coefficient (MCC) should serve as an additional scalar performance metric because it takes into account true positives, true negatives, false positives and false negatives and it simplifies the binary classification.

# Task 2: text clarity classification prototype

## 5a. Ethical discussion

There are several important considerations that arise when discussing the ethical implications of using a clarity algorithm to assess a user’s work, such as the bias and fairness of it. The model may discriminate against certain types of content and this could stem from biased training data. To rectify this, employing techniques such as bias detection could help promote fairness and mitigate discrimination.

The potential harm to users is also another consideration, it could be a case of unfair rejections or feeling discouraged from participating in the community. Therefore, it is essential to consider the psychological impact on the users who receive a rejection notification. Implementing user testing and gathering feedback would mitigate the ethical harm.

Due to its nature, there will be no clear explanation as to why the text has been rejected. This may not benefit the users as they aren’t receiving feedback on their work. Providing a clarity assessment criteria can promote fairness and transparency for users.

## 5b. Data labelling

A Clarity Criteria should be first implemented, the Clarity Criteria would consist of the two labels: ‘Clear Enough’ and ‘Not Clear Enough’. ‘Clear Enough’ indicates to the machine that the paragraph is logically organised and free from overly technical language, making it easy to understand to the wider audience. Whereas, ‘Not Clear Enough’ implies that the paragraph is disorganised and contains ambiguous language, full of jargon and requires more knowledge to understand.

#### The labelling process:

1. Generate a random sample of 100 paragraphs for insight into the clarity levels.
2. Manually label the paragraphs based on Clarity Criteria as “clear\_enough” or “not\_clear\_enough”.
3. Review of subset for consistency in applied Clarity Criteria.
4. Evaluate certainty levels for each labelled paragraph as high, medium or low.

##### Final Label Statistics:

***Total Paragraphs Labelled:*** *100*

***Number of ‘Clear Enough’ Paragraphs:***

***Number of ‘Not Clear Enough' Paragraphs:***

##### Examples of correctly labelled paragraphs:

1. *Clear Enough:*

***Paragraph:***

***Level of certainty:*** *High*

1. *Not Clear Enough:*

***Paragraph:***

***Level of certainty:*** *Medium*

## 5c. Model building and evaluation

##### Data preprocessing

1. Address class imbalance by implementing mode imputation to fill missing values in the text clarity column.
2. Perform oversampling technique to rectify the data imbalance.
3. Drop duplications and missing values from dataset before splitting into two dataframes.
4. One dataframe contains ‘par\_id’ and ‘paragraph’ columns, the other includes ‘par\_id’ and ‘text clarity’ labels.

##### Text encoding and model building’

1. Pre-process the data: extract text from ‘paragraph’ and convert into a list.
2. Implement TF-IDF vectorisation to transform text into a format suitable the model (Simha, 2021).
3. Extract labels from ‘text clarity’ dataframe and split the dataset into training and testing sets using scikit-learn, with an 80-20 split.
4. Train using a Logistic Regression model on the training data to predict the text clarity labels based on TF-IDF features.

A Grid Search technique was then implemented to identify what values for the hyperparameters are most appropriate for the model. The table below shows the results of the values along with their descriptions on why the choice was made.

**Table 7: Hyperparameters with values and detailed descriptions**

|  |  |  |
| --- | --- | --- |
| Hyperparameter | Value | Description |
| C | **0.001** | Controls the strength of regularisation in Logistic Regression. 0.001 indicates a stronger regularisation as it is a very low value. |
| max\_iter | **100** | Maximum number of iterations for the solver to converge. The value of 100 is chosen to allow sufficient iterations for convergence without risking overly long training times. |
| penalty | **‘l2’** | Type of regularisation penalty. ‘l2’ refers to L2 regularisation. This regularisation helps to prevent overfitting by penalising large coefficients. |
| solver | **‘liblinear’** | This is a optimisation algorithm. ‘lbfgs’ is a good choice for small datasets. |

**Table 8: Classification report for text clarity model**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| text clarity | Precision | Recall | F1-Score | Support |
| **Clear enough** | 1.00 | 1.00 | 1.00 | 1735 |
| **Not clear enough** | 1.00 | 1.00 | 1.00 | 1804 |
| **Accuracy** |  |  | 1.00 | 3539 |
| **Macro Avg** | 1.00 | 1.00 | 1.00 | 3539 |
| **Weighted Avg** | 1.00 | 1.00 | 1.00 | 3539 |

**Table 9: Confusion Matrix for text clarity model**

|  |  |  |
| --- | --- | --- |
| Actual labels/Predicted labels | Predicted  clear\_enough | Predicted  not\_clear\_enough |
| **Actual clear\_enough** | 1735 | 0 |
| **Actual not\_clear\_enough** | 0 | 1804 |

Table 8 results indicate there are no misclassifications as there are 1735 instances of ‘clear\_enough’ and 1804 instances of ‘not\_clear\_enough’ indicating a perfect classification.

Table 9’s Classification Report presents precision, recall, F1-score and support for each label with all the metrics displaying a perfect score indicating a balanced model performance.

**Table 10: Baseline Classification report**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| text clarity | Precision | Recall | F1-Score | Support |
| **Clear enough** | 0.49 | 1.00 | 0.66 | 1735 |
| **Not clear enough** | 0.00 | 0.00 | 0.00 | 1804 |
| **Accuracy** |  |  | 0.49 | 3539 |
| **Macro Avg** | 0.25 | 0.50 | 0.33 | 3539 |
| **Weighted Avg** | 0.24 | 0.49 | 0.32 | 3539 |

Table 10 is a Baseline Classification Report that was performed to provide a reference point for evaluating the model’s performance. It achieves perfect for all metrics for the majority class (‘clear\_enough’) but it completely failed to predict the minority class (‘not\_clear\_enough’).

In contrast, the model achieves perfect scores for both classes indicating its balanced accuracy compared to the baseline report.

## 5d. Task 2 Conclusions

* The model is successful according to the client’s definition of success as it does not overfit on the training data and outperforms the baseline model, as seen on Table 8 and Table 10.
* It is recommended that the Matthews Correlation Coefficient (MCC) should serve as an additional scalar performance metric because it takes into account true positives, true negatives, false positives and false negatives. The MCC provides a single value that represents the simplicity of binary classifications, this makes it a suitable choice for the model’s performance evaluation.
* A suggestion that could heavily improve the model’s performance is to incorporate more diverse features into the model, such as adding semantic features. This will capture a broader range of linguistic characteristics that contribute to text clarity.

# Self-reflection

An area that could have been improved was the modelling in Task 1, the performance metric scores seem relatively low so there is definitely room for improvement with the programming. I would improve it by having a better understanding of the concept.

# References

Brownlee, J. (2020, August 15). *What is a Confusion Matrix in Machine Learning*. Retrieved from Machine Learning Mastery: https://machinelearningmastery.com/confusion-matrix-machine-learning/

Kharwal, A. (2021, July 7). *Classification Report in Machine Learning*. Retrieved from The Clever Programmer: https://thecleverprogrammer.com/2021/07/07/classification-report-in-machine-learning/

Simha, A. (2021, October 6). *Understanding TF-IDF for Machine Learning*. Retrieved from Capital One: https://www.capitalone.com/tech/machine-learning/understanding-tf-idf/