Fundamentals of AI – CS5100 – Fall 2025

Assignment 4

Submitted by: Megha Manoj

Objective

The objective of this report is to summarize the findings from conducting document classification on the provided 20 newsgroups dataset using Logistic Regression, Naive Bayes and Convolutional Neural Network classifiers and summarize their results.

Dataset description

The 20 news groups dataset consists of 18774 documents belonging to 20 news group categories. The provided dataset has already been split into a train-test ratio of 6:4 with the following files provided for the train dataset:

- Train.data: A .data file consisting of records where each record represents
  - Document ID: A unique ID for one particular document
  - Word ID: A unique index representing a unique word in the entire dataset
  - Count: The number of times a particular word appears in a document.
- Train.map: A .map file where each record consists of a label name and its corresponding label ID.
- Train.label: A .label file where the encoded label ID for each document is given. The line number of a particular label ID represents the document ID of that particular label.

Similar files have been provided for the test dataset and a vocabulary.txt file is also present in which all unique words present in the dataset are provided. These words have their respective word ID represented by the line number on which they reside.

Methodology

The methodology for performing text classification requires several steps from exploring the dataset to determine how to preprocess the data to building the model and training it. This section describes in-depth each step performed for document classification.

A. **Exploratory Data Analysis**
   In order to gain a better understanding of the dataset before preprocessing it, a simple exploratory data analysis was conducted on the train dataset which gave the following insights. The average number of documents per category is 563 documents while maximum number of documents is 599 for the category 'soc.religion.christian' and the minimum number of documents is 376 for category 'talk.religion.misc'. A few categories in the dataset consist of a total number of documents below 500 indicating an imbalance in the data.

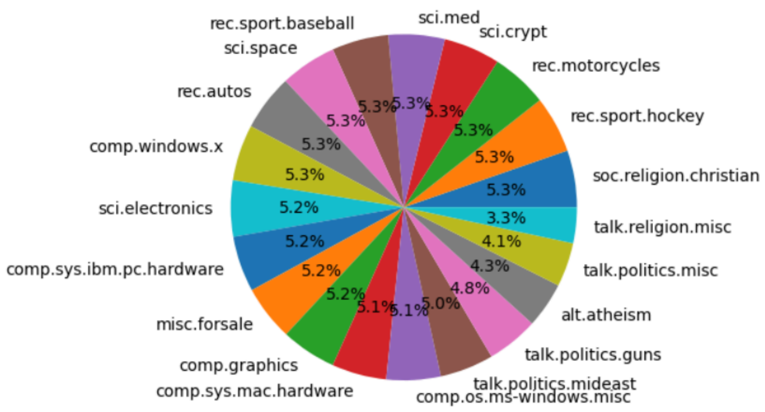
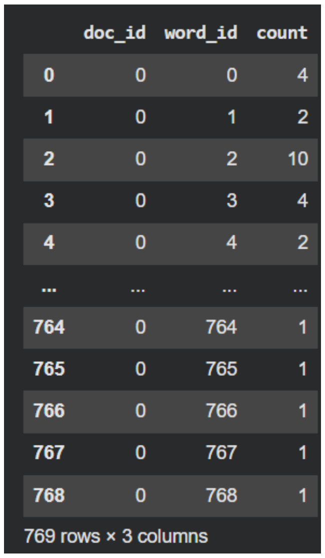


Figure 1. Distribution of documents under each label

Looking closer at a few documents in the dataset, the number of words in a document can vary from 2 to 11233. As the words for each document are specified with their count, it can be deduced that the ordering of words in the document is not provided. Therefore, conducting document classification on the following dataset would be dependent on the occurrence of key words which can be related to a particular category.



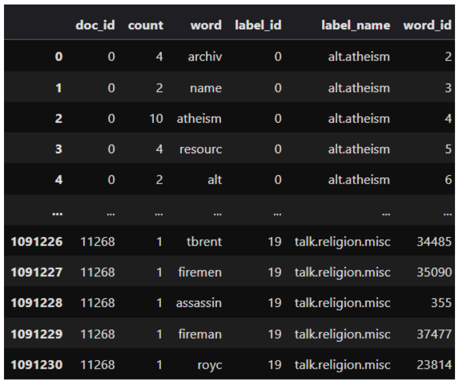
	doc_id	word_id	count
0	0	0	4
1	0	1	2
2	0	2	10
3	0	3	4
4	0	4	2
...	...	...	...
764	0	764	1
765	0	765	1
766	0	766	1
767	0	767	1
768	0	768	1

769 rows × 3 columns

Figure 2. Train.data file

All words within the dataset are in lower case with no special characters present in them. There is a significant number of stop words present within the document and certain words that are quite synonymous to each other such as 'run' and 'running' present throughout the documents. With respect to null records, there are none within the dataset.

B. Data preprocessing



	doc_id	count	word	label_id	label_name	word_id
0	0	4	archiv	0	alt.atheism	2
1	0	2	name	0	alt.atheism	3
2	0	10	atheism	0	alt.atheism	4
3	0	4	resourc	0	alt.atheism	5
4	0	2	alt	0	alt.atheism	6
...	...	...	...	...	...	...
1091226	11268	1	tbrent	19	talk.religion.misc	34485
1091227	11268	1	firemen	19	talk.religion.misc	35090
1091228	11268	1	assassin	19	talk.religion.misc	355
1091229	11268	1	fireman	19	talk.religion.misc	37477
1091230	11268	1	royc	19	talk.religion.misc	23814

In order to gain a better visualization of the words, their indexes as well as labels for each document, the provided 3 files are concatenated.The preprocessing steps for all 3 models are the same except at the final steps where the data is represented in numeric format for the classification task.

First, to address the imbalance present within the train dataset, data augmentation is performed by creating synthetic documents using random document lengths and vocabulary present in each category.

Figure 3. Preprocessed train dataset with labels

To remove noise information present in the form of common words such as ‘an’, ‘the’, etc. stop word removal is performed in the dataset. Since there are derived (words which vary grammatically but have the same meaning) present in the dataset, these words are reduced to their stem using stemming so that the size of the vocabulary is reduced to remove redundancy. The indexing of these words are then updated within the vocabulary because of redundant indexing caused by stemming in order to map the updated indexing to the train and test dataset. The total count of the stemmed words is also updated in each document in order to maintain consistency with the numbering of the data.

The data is then prepared for training and testing for the 3 models in the following formats:

A. Naive Bayes and Logistic Regression classifiers

Naive Bayes and Logistic Regression classifiers take into account the frequency of words within the corpus without considering the ordering of the words within a document. Additionally the provided data is in the form of Bag-of-Words approach with the frequency of each word present in every document being provided. Therefore, a count matrix is created using the document IDs as the rows and the words as the columns such that for a matrix C, C[0][2] would contain the count of a word ‘archiv’ (originally ‘archive’) with index 0 in document number 0. This matrix is then converted into Term Frequency-Inverse Document Frequency (TF-IDF) format which would assign scores for each word in each document based on its frequency of occurrence using the following formula:

$$TF - IDF(w, d, D) = \frac{\#\ Occurrence\ of\ word\ w\ in\ document\ d}{\#Total\ terms\ in\ document\ d} * log \frac{\#\ Total\ documents\ in\ the\ corpus\ D}{\#Documents\ with\ word\ w\ in\ them}$$

B. Convolutional Neural Networks (CNN)

Convolutional neural networks are used on data with sequential ordering to capture patterns and n-gram features within the documents. For this purpose, each document must be encoded into embeddings which consist of each word encoded by its corresponding word ID by setting a fixed length for the embedding of each document. For documents with count of words lower than the fixed length, the input is padded to match the fixed length size, else if it exceeds the fixed length the number of words within the document are trimmed to fit the length. The padding is done via a '<pad>' token initialized within the vocabulary. Furthermore to handle any unknown words present within the test data, a '<unk>' token is added which would be mapped to words which are out of vocabulary.

The test dataset is split into 50% test and 50% validation dataset to evaluate the model while training.

C. Model Architecture

This section describes the initialized parameters for each model and The models for logistic regression and naive bayes classifiers are initialized using the scikit-learn library in python while the CNN model has been built using PyTorch

1. Logistic Regression Classifier

A logistic regression model is initialized using the scikit-learn library with balanced weights and a maximum of 1000 iterations before the model finds the best possible solution.

2. Naive Bayes Classifier

A multinomial naive bayes classifier has been chosen due to the presence of discrete data within the dataset in the form of word count.

3. CNN

A CNN model trained for 25 epochs with an embedding size of 500 and 200 filters with skip gram sizes 2,3 and 4 to perform text classification was constructed using the following layers:

1. Embedding layer: Input layer which helps the model understand complex, categorical information by converting it into vector representation.

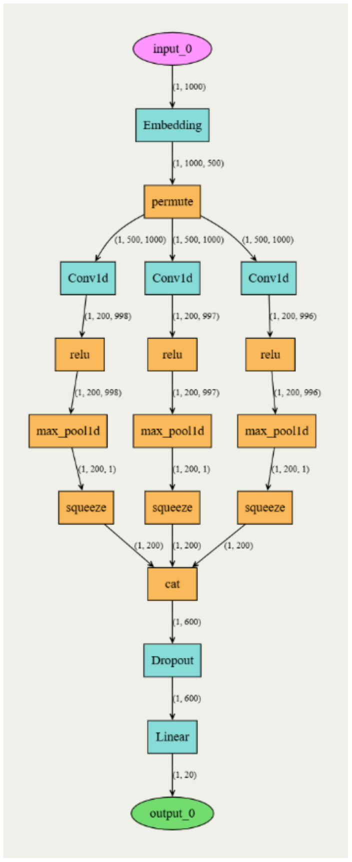


Figure 4. CNN Model Architecture for Text Classification

2. Conv1D: Convolutional layers which retrieve the features from the document. In this case a 100 filters are provided of region sizes 2, 3 and 4 which would output a feature map.
3. ReLU and Maxpooling1D: ReLu and Maxpooling is applied to the feature maps outputted by the convolution layer in order to reduce the it to a scalar vector
4. Dropout: 20% of the neurons are randomly disposed to prevent overfitting.
5. Linear: The linear layer would return the output of the classification by picking the label which has the highest value.

An AdamW optimizer with learning rate of 0.0001 and weight decay of 0.00001 with a loss function of categorical cross entropy loss to stabilize and improve model performance.

D. Metrics used for evaluating model performance :

For descriptive purposes, the definition of the parameters used in calculation of these metrics are as follows:

True Positive (TP): Number of documents which belong to a label were classified as belonging to their actual label.

True Negative (TN): Number of documents which don't belong to a label were classified as not belonging to that label.

False Positive: Number of documents which don't belong to a label but were classified as belonging to that label.

False Negative: Number of documents which belong to a label but were classified as not belonging to that label.

A total of 4 metrics have been used in evaluating the performance of the 3 models.

- Precision: A score between 0 and 1 representing the number of documents classified as a label actually belong to that label . A high value for precision indicates that the model has correctly predicted labels majority of the test data belonging to a particular label.

$$Precision \ = \ \frac{TP}{FP + TP}$$

- Recall: A score between 0 and 1 representing the number of documents correctly predicted as belonging to their label. A high value for precision indicates that the model has correctly predicted labels majority of the test data belonging to a particular label.

$$Recall \ = \ \frac{TP}{TN + TP}$$

- F1-Score: The F1-score combines the precision and recall score to give an overview of how well the model has performed for one particular label. A higher F1-score indicated good performance given that the precision and recall are also high.

$$F1 - Score \ = \ \frac{Precision * 2 * Recall}{Precision + Recall}$$

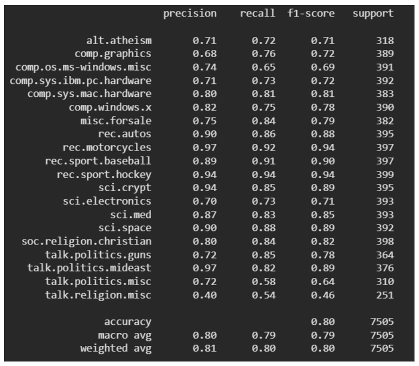
- Support: The number of documents in a test dataset belonging to a particular class.
- Accuracy: A score between 0 and 1 indicating the number of true positives within the classified output.

$$Accuracy \ = \ \frac{TP}{TP+FP+FN+TN}$$

The labels of the news groups are encoded into numbers from 0 to 20 with the mapping followed in the train dataset.

Discussion about model performance

A classification report was generated to analyze the results of the model testing after training each of the 3 classifiers which showed that the logistic and naive bayes classifiers both perform similarly with an accuracy of 80% while the CNN model has a low accuracy of 72%.

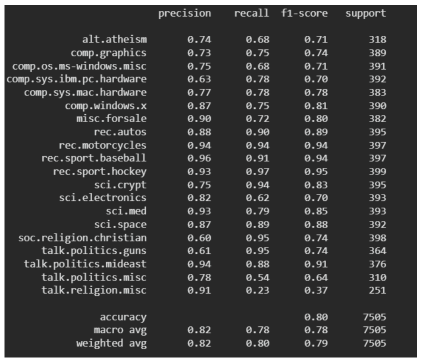


```
                          precision    recall  f1-score   support

             alt.atheism       0.71      0.72      0.71       318
           comp.graphics       0.68      0.76      0.72       389
 comp.os.ms-windows.misc       0.74      0.65      0.69       391
comp.sys.ibm.pc.hardware       0.71      0.73      0.72       392
   comp.sys.mac.hardware       0.80      0.81      0.81       383
          comp.windows.x       0.82      0.75      0.78       390
            misc.forsale       0.75      0.84      0.79       382
               rec.autos       0.90      0.86      0.88       395
         rec.motorcycles       0.97      0.92      0.94       397
      rec.sport.baseball       0.89      0.91      0.90       397
        rec.sport.hockey       0.94      0.94      0.94       399
               sci.crypt       0.94      0.85      0.89       395
         sci.electronics       0.70      0.73      0.71       393
                 sci.med       0.87      0.83      0.85       393
               sci.space       0.90      0.88      0.89       392
  soc.religion.christian       0.80      0.84      0.82       398
      talk.politics.guns       0.72      0.85      0.78       364
   talk.politics.mideast       0.97      0.82      0.89       376
      talk.politics.misc       0.72      0.58      0.64       310
      talk.religion.misc       0.40      0.54      0.46       251

                accuracy                           0.80      7505
               macro avg       0.80      0.79      0.79      7505
            weighted avg       0.81      0.80      0.80      7505
```

Logistic Regression Model

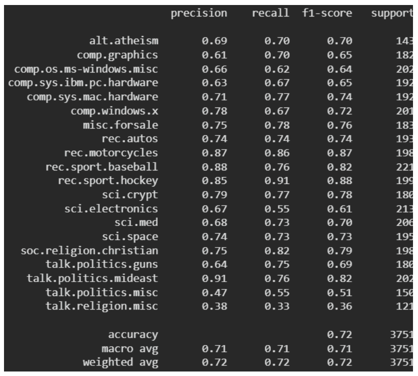


```
                          precision    recall  f1-score   support

             alt.atheism       0.74      0.68      0.71       318
           comp.graphics       0.73      0.75      0.74       389
 comp.os.ms-windows.misc       0.75      0.68      0.71       391
comp.sys.ibm.pc.hardware       0.63      0.78      0.70       392
   comp.sys.mac.hardware       0.77      0.78      0.78       383
          comp.windows.x       0.87      0.75      0.81       390
            misc.forsale       0.90      0.72      0.80       382
               rec.autos       0.88      0.90      0.89       395
         rec.motorcycles       0.94      0.94      0.94       397
      rec.sport.baseball       0.96      0.91      0.94       397
        rec.sport.hockey       0.93      0.97      0.95       399
               sci.crypt       0.75      0.94      0.83       395
         sci.electronics       0.82      0.62      0.70       393
                 sci.med       0.93      0.79      0.85       393
               sci.space       0.87      0.89      0.88       392
  soc.religion.christian       0.60      0.95      0.74       398
      talk.politics.guns       0.61      0.95      0.74       364
   talk.politics.mideast       0.94      0.88      0.91       376
      talk.politics.misc       0.78      0.54      0.64       310
      talk.religion.misc       0.91      0.23      0.37       251

                accuracy                           0.80      7505
               macro avg       0.82      0.78      0.78      7505
            weighted avg       0.82      0.80      0.79      7505
```

Multinomial Naive Bayes Classifier



```
                          precision    recall  f1-score   support

             alt.atheism       0.69      0.70      0.70       143
           comp.graphics       0.61      0.70      0.65       182
 comp.os.ms-windows.misc       0.66      0.62      0.64       202
comp.sys.ibm.pc.hardware       0.63      0.67      0.65       192
   comp.sys.mac.hardware       0.71      0.77      0.74       192
          comp.windows.x       0.78      0.67      0.72       201
            misc.forsale       0.75      0.78      0.76       183
               rec.autos       0.74      0.74      0.74       193
         rec.motorcycles       0.87      0.86      0.87       198
      rec.sport.baseball       0.88      0.76      0.82       221
        rec.sport.hockey       0.85      0.91      0.88       199
               sci.crypt       0.79      0.77      0.78       180
         sci.electronics       0.67      0.55      0.61       213
                 sci.med       0.68      0.73      0.70       206
               sci.space       0.74      0.73      0.73       195
  soc.religion.christian       0.75      0.82      0.79       198
      talk.politics.guns       0.64      0.75      0.69       180
   talk.politics.mideast       0.91      0.76      0.82       202
      talk.politics.misc       0.47      0.55      0.51       150
      talk.religion.misc       0.38      0.33      0.36       121

                accuracy                           0.72      3751
               macro avg       0.71      0.71      0.71      3751
            weighted avg       0.72      0.72      0.72      3751
```

CNN

Classification report of each model

In terms of F1-score which provides a better analysis of the precision and recall values, the F1-scores of each label was lower by 0.2 before the data augmentation was added to the preprocessing steps. Due to the correction of the dataset imbalance, the scores have overall improved with the naive bayes model having better performance in comparison with the other 2 models. But the performance across all labels seem to vary drastically.

The Naive Bayes classifier has been able to give an overall good performance in terms of f1-scores for most of the labels except for religion and politics miscellaneous which has a high precision but very low recall score indicating that the model is capable of avoiding false positives for these labels. However because the support is low in comparison to the rest of the dataset, these scores are most likely to be unreliable measures towards evaluating the model performance. Label 11 on the other hand has a low precision score but high recall indicating that there are chances of documents being misclassified for that label.

The logistic regression classifier seems to have comparatively average performance with the naive bayes classifier as the f1-scores mostly revolve around 0.7 to 0.89. The precision and recall values seem to be lower in most cases which is likely the cause for low f1-scores. Increasing the maximum number of iterations did not improve the model performance.

The CNN classifier has the worst performance with respect to the classification report as the F1-scores of all labels vary more drastically than the former 2 classifiers. Initially with embedding dimensions of the performance for religion and politics miscellaneous groups seem to be low for CNN as well. The labels for hockey and motorcycles seem to have the highest F1-scores 0.88 and 0.87 respectively.

To conclude, the models perform fairly well In order to improve the model performance, the next potential step is to attempt filtering out low frequency vocabulary from the train dataset and storing it as temporary vocabulary for preprocessing test data as well as try more variations of parameter values such as embedding size, number of epochs, etc for CNN.

References

- Kim, Yoon. "Convolutional neural networks for sentence classification." *arXiv preprint arXiv:1408.5882* (2014).
- https://towardsai.net/p/l/stemming-porter-vs-snowball-vs-lancaster