PROJECT 2: TOPIC CATEGORIZATION - REPORT

MARCH 20, 2023

UNM Spring 2023

Course:

CS429 - Introduction to Machine Learning

Instructor:

Trilce Estrada

Teacher Assistant:

Adnan Bashir

Group Members:

Libra Vento

Sarathi TS

CONTENTS

I INTRODUCTION

Topic categorization is a common task in the field of Natural Language Processing (NLP). It refers to building a model which can determine to which category of texts or documents, among many in a set of them, a word or phrase is most likely to belong. Due to this nature, it is considered as a classification problem, thus requiring a classifier to solve it.

A first approach to this problem would be the use of a **Naïve Bayes classifier**, which bases itself on the Bayes theorem. A different approach requires the use of **logistic regression**. As an improvement, the importance of the features is considered to get more accurate results of the model.

The objective of this project is to implement these models from scratch to understand how they work. To do this, the classic newsgroups dataset is used to find out to which category a document belongs to, based on the words it contains.

II DATA REPRESENTATION

A **bag of words** representation is considered for the model. This means that the position of the words in the text it belongs to is ignored, in favor of solely representing its number of occurrences. In other words, all positions are considered as having the same distribution, but different frequencies. In this format, it is easier to apply for both Naïve Bayes and Logistic Regression.

In order to regularize the distribution, standarization is applied over the bag of words in a way which allows centering the feature values around the specific document the model is working with instead of considering their value for all the documents in the dataset. Each text is represented as a vector of the size of the number of unique words in the collection of texts, where each item is a number representing the frequency of a word in the specific text.

III NAÏVE BAYES

A Naïve Bayes classifier uses the **Bayesian Theorem** to model classification. This theorem stands the following:

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Naïve Bayes assumes **conditional independence**, which means that features are independent from each other given the class they belong to. In formal terms, that *X is conditionally independent of Y given Z if the probability distribution for X is independent of the probability distribution of Y given the value for Z* [3], or, mathematically expressed:

$$P(X, Y|Z) = P(X|Z)P(Y|Z)$$

Generalizing then, we get that:

$$P(X_1, X_2, ..., X_n|Y_i) = P(X_1|Y_i)P(X_2|Y_i)...P(X_n|Y_i)$$

For which the model aims to compute (considering that

$$\hat{y} = \underset{y \in Y}{\operatorname{argmax}}\, P_\theta(y|x)$$

where θ is estimated from the dataset `D`, using MLE or MAP. The latter is used on this case, as priors are used to regularize and avoid overfitting.

Applying Bayes' rule, we find that the decision rule this model considers is:

$$h_{NB}(X) = \underset{y \in Y}{\operatorname{argmax}}\, P(y) \prod_{i=1}^{n} P(X_i|y)$$

It considers `X` to be the features (in this case, the sequence of [X_0, .., X_n] words in an article, document or text), `y` as the classes (or each of the documents) and `n` as the number of features.

.

Q1: Explain why it would be difficult to accurately estimate the parameters of this model on a reasonable set of documents.

It would be difficult to accurately estimate the parameters of this model on a reasonable set of documents due to the possibility of having many repeated words, specially if the set of documents are all on similar topics, since the lexicon used would be the same, not to mention that within the same document words could be repeated. There is a fair chance that the rule of $P(X_i|Y) \neq P(X_j|Y) \forall i, j$ does not follow. An important measure against this would be to consider a stop list or a word ranking to filter out some words that do not add too much value.

III-A Implementation

The full implementation can be found in the appendix A.

A multinomial Naïve Bayes classifier was implmented. Maximum Likelihood Estimation (MLE) is considered to calculate the probability for `Y`, while Maximum a Posteriori (MAP) was used to obtain the probabilities for `P(A|B)`. They follow:

MLE:

$$P(Y_k) = \frac{NumberOfDocsLabeledY_k}{TotalNumberOfDocs}$$

MAP:

$$P(Y_k = \frac{(CountOfX_iInY_k) + (\alpha - 1)}{(TotalWordsInY_k) + ((\alpha - 1) * (LengthOfVocabList))}$$

with $\alpha = 1 + \beta$. The value of β was originally intended to be $\beta = \frac{1}{V}$, with `V` being the vocabulary size, however, this value was modified to find out if its value affected the final accuracy of the model.

And with these the model should be able to identify which class a document belongs to by following the rule

$$Y^{new} = \underset{y \in Y}{\operatorname{argmax}} \, log_2(P(Y_k)) + \sum_i (NumberOfX_i^{new}) log_2(P(X_i|Y_k)$$

.

It is worth noting that no preprocessing techniques were necessary for this method due to the probability function assuming conditional independence.

III-B Experimentation

Q2a: Re-train your Naive Bayes classifier for values of β between 0.0001 and 1 and plot the accuracy over the test set for each value of β. Explain in a few sentences why accuracy drops for both small and large values of β

The results obtained for this testing are shown in Table 1. Some additional intermediate points where considered when plotting them into a graph. The results are shown in Figure 1.

β	Accuracy
10^{-4}	0.8671
10^{-3}	0.8746
10^{-2}	0.8775
10^{-1}	0.8775
10^{0}	0.8358

TABLE I: Values of β with their accuracies

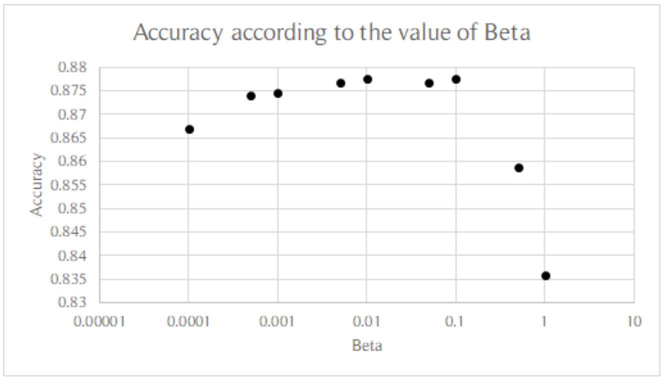


Fig. 1: Accuracies according to the value of β

The results show a peak of accuracy when working with $\beta = 10^{-2}$ and $\beta = 10^{-3}$. Even if the differences between accuracies are not too significant, as they range from 83% to 88%, it still shows that there is a difference depending on the value given to β.

With values too small for β, the prior distribution becomes flat, thus reducing the effect of the prior distribution over the weights of the priors. On the other hand, values that are too large decrease variance, as they increase prior information, and if too excessive, diminish prior uncertainty to an extent enough that makes the density too narrow to work with. [1]

IV LOGISTIC REGRESSION

Logistic regression is a type of learning model that uses the probability of an event to predict whether something belongs to a class by fitting a logistic function to the data. It follows:

$$\ln P(D_Y|D_X, w) = \sum_{j=1} y^j(w_0 + \sum_{i}^{n} w_i x^j{}_i) - \ln(1 + exp(w_0 + \sum_{i}^{n} w_i x^j{}_i))$$

With D: dataset, Y: class, y: instance of y, X: features, x: instance of x, w: weights matrix.

Gradient descent was included for this model in order to minimize the error:

$$W^{t+1} = W^t + \eta((\Delta - P(Y|W,X))X - \lambda W^t)$$

Where W: weights matrix, η: learning rate, Δ: Delta Equation matrix, λ: penalty term

IV-A IMPLEMENTATION

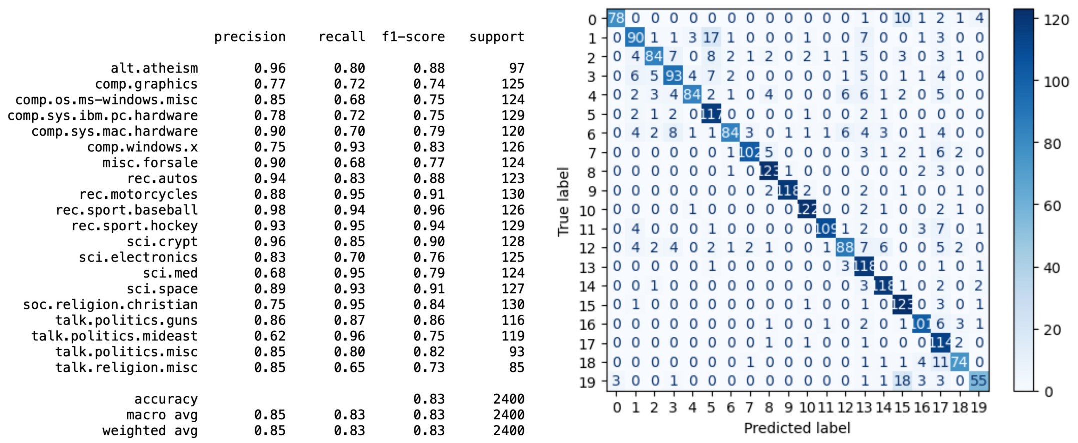
The full implementation can be found in the appendix A.

Due to the amount of data present in the dataset and the complexity of the operations needed to get this model to work, the model was implemented using sparse matrixes. Conversions of the data to this format were made accordingly because of this reason.

To make sure the model works accordingly and even if the classes are already encoded with an id, **One Hot Encoding** was used to minimize any idea of priority or order within them. As part of the initial preprocessing, **normalization** was applied to the frequencies of the features.

Q5: Are there any newsgroups that the algorithm(s) confuse more often than others? Why?

When initially training the model, class imbalance was noticeable in the dataset, as shown in the accuracy and confusion matrix in Figure 2.



```
                          precision    recall  f1-score   support

             alt.atheism       0.96      0.80      0.88        97
           comp.graphics       0.77      0.72      0.74       125
 comp.os.ms-windows.misc       0.85      0.68      0.75       124
comp.sys.ibm.pc.hardware       0.78      0.72      0.75       129
   comp.sys.mac.hardware       0.90      0.70      0.79       120
          comp.windows.x       0.75      0.93      0.83       126
            misc.forsale       0.90      0.68      0.77       124
               rec.autos       0.94      0.83      0.88       123
         rec.motorcycles       0.88      0.95      0.91       130
      rec.sport.baseball       0.98      0.94      0.96       126
        rec.sport.hockey       0.93      0.95      0.94       129
               sci.crypt       0.96      0.85      0.90       128
         sci.electronics       0.83      0.70      0.76       125
                 sci.med       0.68      0.95      0.79       124
               sci.space       0.89      0.93      0.91       127
  soc.religion.christian       0.75      0.95      0.84       130
      talk.politics.guns       0.86      0.87      0.86       116
   talk.politics.mideast       0.62      0.96      0.75       119
      talk.politics.misc       0.85      0.80      0.82        93
      talk.religion.misc       0.85      0.65      0.73        85

                accuracy                           0.83      2400
               macro avg       0.85      0.83      0.83      2400
            weighted avg       0.85      0.83      0.83      2400
```

Fig. 2: Classification Accuracy Report and Confusion Matrix

As seen from their low F1-scores compared to the other classes, news groups related to computing and sciences (notably *comp.graphics*, *comp.os.ms-windows.misc*, *comp.sys.ibm.pc.hardware*, *comp.sys.mac.hardware* and *sci.electronics*) share words in common, due to them being of similar fields and dealing with similar topics (such as computer hardware and software), so it is probable that the overlap between these classes was causing confusion. This was also reflected in the accuracy, as early testing resulted in values no higher than 78%. Because of this, some improvements were made during preprocessing to improve the model's performance, starting by the application **Synthetic Minority Oversampling Technique (SMOTE)** over the data to fix the imbalances. [6]

Q6: Propose a method for ranking the words in the dataset based on how much the classifier 'relies on' them when performing its classification.

A first idea was using **Term Frequency-Inverse Document Frequency (tf-idf)** over the data. It is a scoring technique which considers the word's frequency in the document such that the rarest terms have higher scores than those present in more documents. [4] **Mutual information** between a word and a class ($MI\left(X_i, Y_j\right) = \sum_{x_i,y_j} P\left(x_i, y_j\right) \log_2 \frac{P(x_i,y_j)}{P(x_i)P(y_j)}$) was used as well.

Q7: Implement your method, and print out the 100 words with the highest measure

After applying this method, the following 100 words were identified as the highest ranked: 'athos', 'bike', 'playoffs', 'ide', 'clipper', 'baseball', 'circuit', 'geb', 'sale', 'cica', 'apple', 'animation', 'henry', 'cars', 'cdt', 'arab', 'cramer', 'kendig', 'benedikt', 'window', 'sunos', 'motif', 'beauchaine', 'guns', 'car', 'sandvik', 'shuttle', 'israel', 'optilink', 'bobbe', 'zoo', 'israeli', 'buphy', 'polygon', 'sahak', 'melkonian', 'windows', 'graphics', 'psyrobtw', 'ceccarelli', 'quadra', 'tga', 'appressian', 'partial', 'ohanus', 'images', 'turks', 'gregg', 'newton', 'jaeger', 'turkish', 'royalroads', 'mlee', 'malcolm', 'clayton', 'armenians', 'christian', 'image', 'bible', 'ico', 'texture', 'offer', 'convert', 'ini', 'occupied', 'ksand', 'extermination', 'rutgers', 'vos', 'bskendig', 'apps', 'armenia', 'circuits', 'iff', 'jxp', 'geometric', 'centris', 'kent', 'weiss', 'tdawson', 'herringshaw', 'abortion', 'disease', 'shipping', 'arabs', 'dx', 'spencer', 'rendering', 'byuvm', 'bus', 'chastity', 'formats', 'medicine', 'med', 'burden', 'keith', 'alink', 'sky', 'olwm', 'skepticism'

The ranking made it possible to identify the point in which the ranking scores become seemingly the same, which allows for determining the number of best features *k* to select during feature extraction which is a way to find which were the most prevalent features in the dataset.

From the graph in Figure 3 (which displays the rankings from the first 10000 words and on) it was found that the value of *k* for this dataset should be around 20000 and 25000, as it is around that spot that the scores look more uniform.

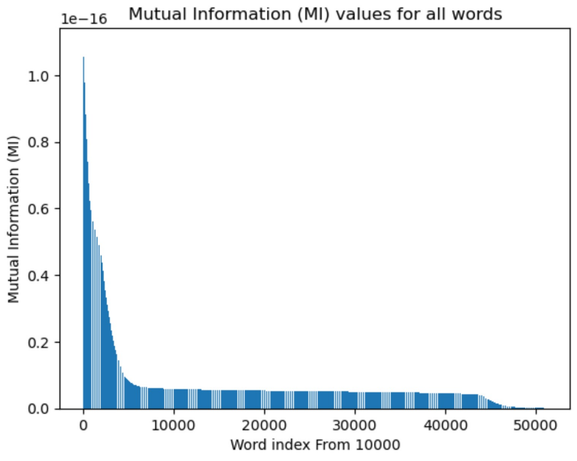


Fig. 3: Word Ranking values for all words

Chi-Squared was used as part of this process due to the fact that it allows testing the independence between two events, which is also why it is commonly used during feature selection. [5] Given that figuring out how to calculate the values for Chi-squared is outside the focus of this project, the function `SelectKBest`from the `sklearn` library with the Chi-Squared function from the same library was used instead. The k used for it was 25000, given the explanation in the previous paragraph.

IV-B Experimentation

Q3: Re-train your Logistic Regression classifier for values of η starting from 0.01 to 0.001, λ = 0.01 to 0.001 and vary your stopping criterion from number of total iterations and report the accuracy over the test set for each value. Explain in a few sentences your observations with respect to accuracies and sweet spots

In order to find the values for η and λ that gave the best accuracy for the model, a random search approach was initially tested. The summarized results for it are shown in Figures 4 and 5. For more detail refer to the appendix A.

To verify the results obtained, a grid search approach was also implemented. For it, values for η were first changed while keeping λ constant, and then values of λ were modified while η was kept constant. The experiment was repeated 5 times, each one of them with a different number of epochs. The plots comparing the accuracies and training losses with these variables are shown in Figures 6 and 7.

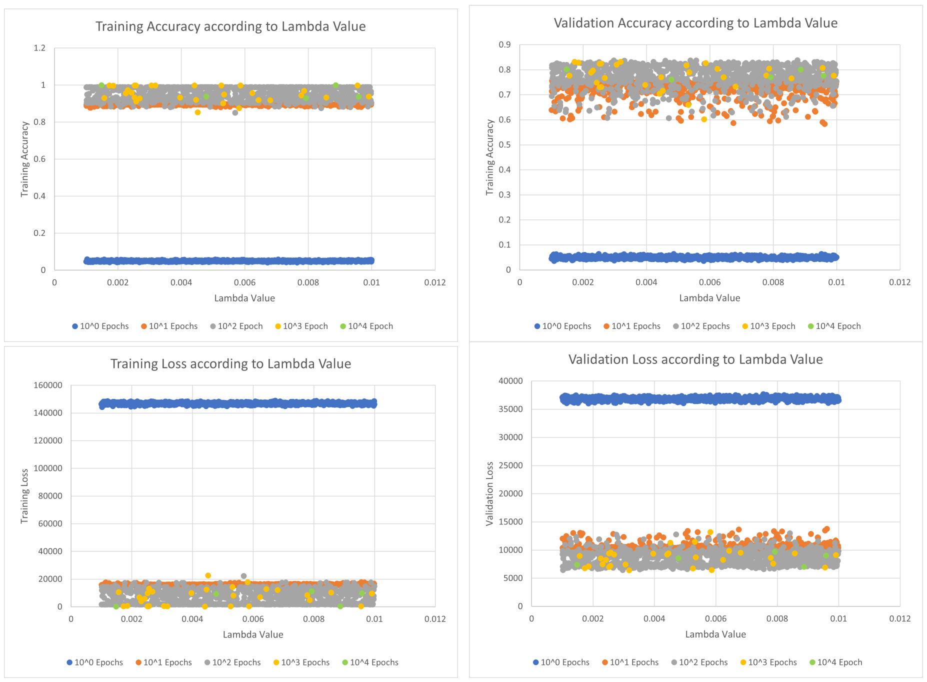


Fig. 4: Accuracies and Loss according to the values of λ using random search

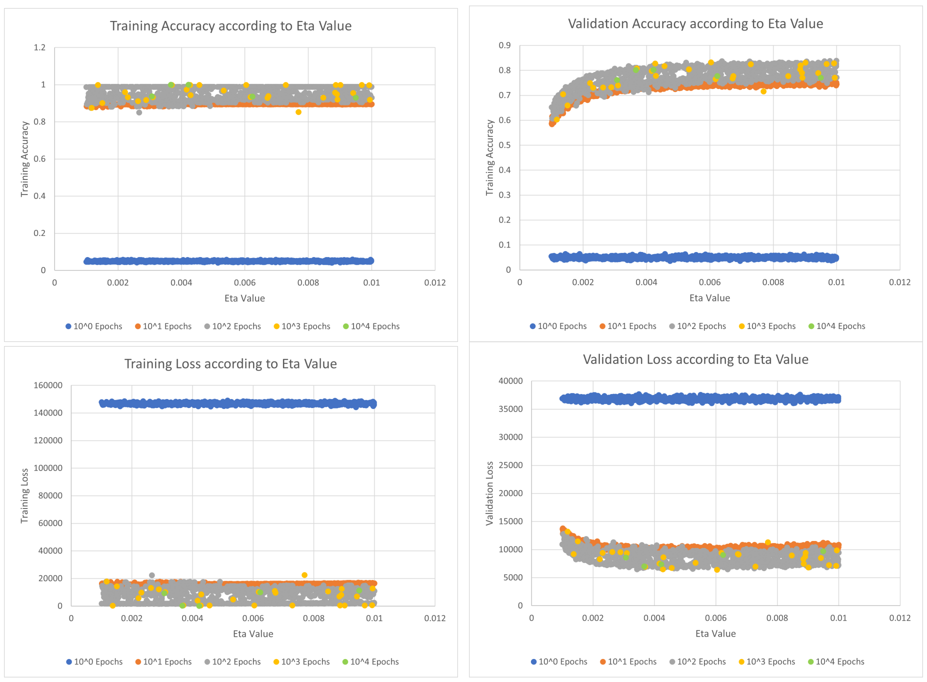


Fig. 5: Accuracies and Loss according to the values of η using random search

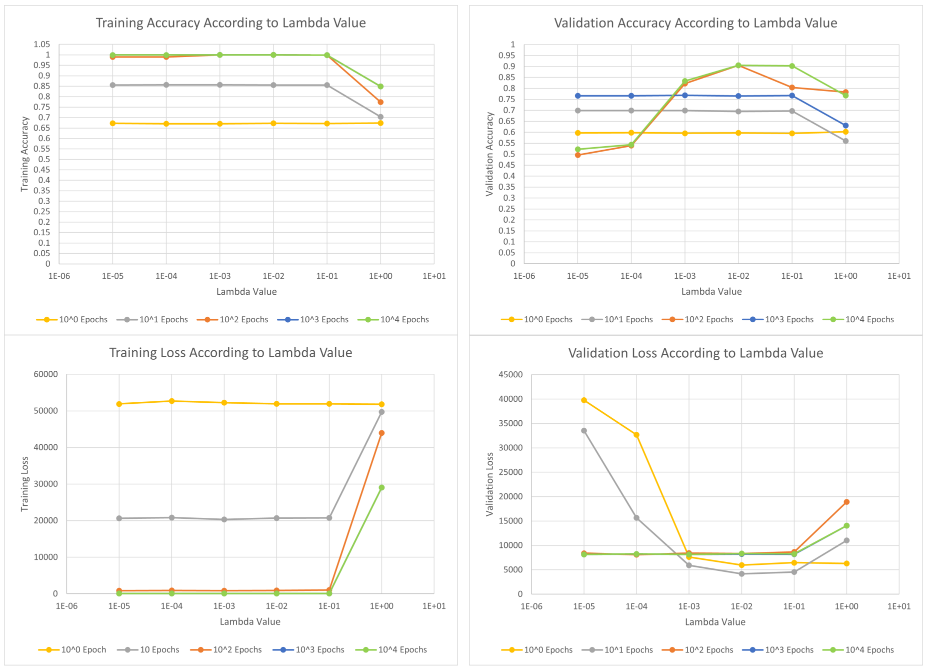


Fig. 6: Accuracies and Loss according to the values of λ using grid search

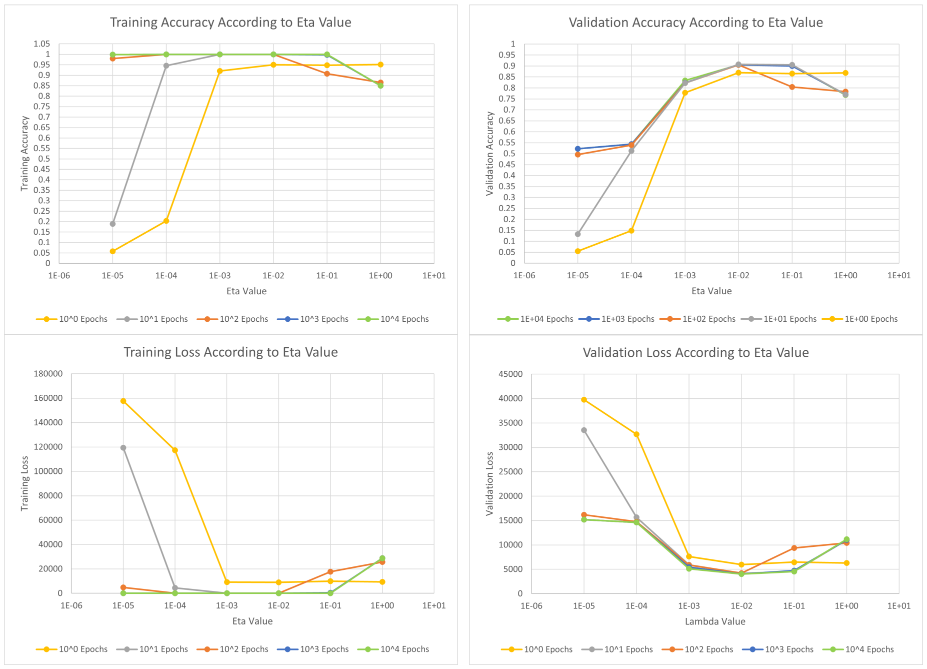


Fig. 7: Accuracies and Loss according to the values of η using grid search

From these results the following optimal values for η and λ were obtained:

Number of epochs	Random Search		Grid Search	
	η	λ	η	λ
10^0	0.00189	0.00951	0.00100	0.0010
10^1	0.00876	0.00374	0.01000	0.0010
10^2	0.00999	0.00877	0.01000	0.00010
10^3	0.00603	0.00318	0.01000	0.00001
10^4	0.00422	0.00148	0.01000	0.00001

TABLE II: Optimal values for η and λ according to the number of epochs

For more details on the results of these experiment refer to the file in the appendix (A).

During random search, 1000 samples were used for 10^0, 10^1 and 10^2, while for 10^3 30 were used and 10^4 only 5 were used. This due to the computational cost of running the operations with more epochs taking too much time to be completed.

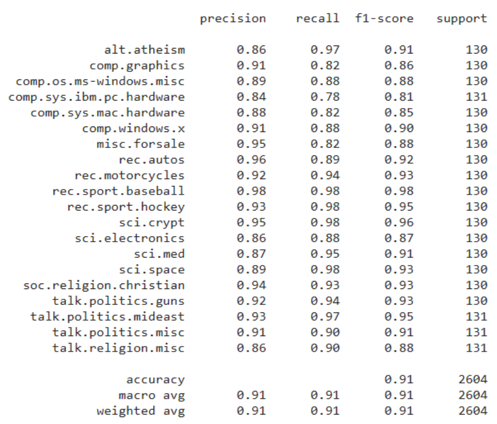
It is interesting to note that the results obtained when running only one epoch are drastically different from those obtained when running more than one, with accuracies being extremely low and loss being extremely high, despite the values η and λ have.

There is a visible tendency that high values for λ values produce worse results. Smaller values do as well, but to a lesser degree. $\lambda = 0.01$ produces the best results overall. As for η, lower values produce worse results than higher ones do for the most part. Only in some of the repetitions it was visible that too high of a value also caused a worse performance. Its sweetspot varies according to the number of epochs, but it is never greater than the value of η.

After obtaining these results, the rest of experiments were done with $\eta = 0.1$, $\lambda = 0.00001$, 1000 epochs (minimum 5 before converging) and a threshold of 1.

Q4: Report your overall testing accuracy, and print out the confusion matrix

As seen in the report for accuracies per class after this training, overall accuracy was about 0.91, and as visible from its respective confusion matrix (Figure 8), class confusion was effectively reduced in comparison to how it looked before. This indicates that the modifications applied during preprocessing were able to improve the model. However, the testing accuracy obtained was only 0.85, which is less than what was obtained while using Naïve Bayes.



```
                          precision    recall  f1-score   support

             alt.atheism       0.86      0.97      0.91       130
           comp.graphics       0.91      0.82      0.86       130
 comp.os.ms-windows.misc       0.89      0.88      0.88       130
comp.sys.ibm.pc.hardware       0.84      0.78      0.81       131
   comp.sys.mac.hardware       0.88      0.82      0.85       130
          comp.windows.x       0.91      0.88      0.90       130
            misc.forsale       0.95      0.82      0.88       130
               rec.autos       0.96      0.89      0.92       130
         rec.motorcycles       0.92      0.94      0.93       130
      rec.sport.baseball       0.98      0.98      0.98       130
        rec.sport.hockey       0.93      0.98      0.95       130
               sci.crypt       0.95      0.98      0.96       130
         sci.electronics       0.86      0.88      0.87       130
                 sci.med       0.87      0.95      0.91       130
               sci.space       0.89      0.98      0.93       130
  soc.religion.christian       0.94      0.93      0.93       130
      talk.politics.guns       0.92      0.94      0.93       130
   talk.politics.mideast       0.93      0.97      0.95       131
      talk.politics.misc       0.91      0.90      0.91       131
      talk.religion.misc       0.86      0.90      0.88       131

                accuracy                           0.91      2604
               macro avg       0.91      0.91      0.91      2604
            weighted avg       0.91      0.91      0.91      2604
```

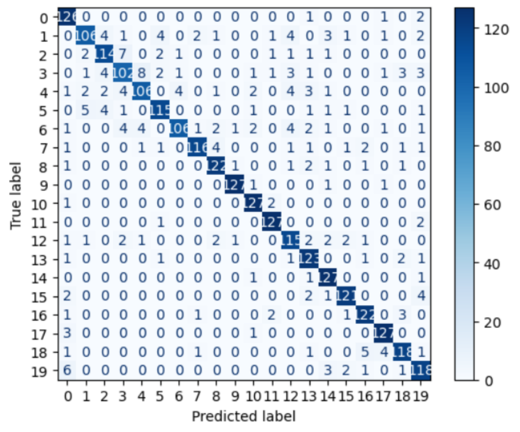


Fig. 8: Classification Accuracy Report and Confusion Matrix with optimal η and λ

Q8: If the points in the training dataset were not sampled independently at random from the same distribution of data we plan to classify in the future, we might call that training set biased. Dataset bias is a problem because the performance of a classifier on a biased dataset will not accurately reflect its future performance in the real world. Look again at the words your classifier is 'relying on'. Do you see any signs of dataset bias?

There are signs of dataset bias due to there being highly ranked words that are exclusive to very specific situations. Some of them include *"sunos"* -an old operating system for Sun Microsystems workstations-, *"quadra"* -an Apple Macintosh model from the mid-ninties- and *"rutgers"* -short for Rutgers University, which offers a vast catalogue of academic degrees, not limited to a certain field.

V CONCLUSIONS

Naïve Bayes and Logistic Regression are two classification approaches that operate under similar principles to find relationships between classes and features, yet produce different results. Both models were successfully implemented as intended and produced satisfactory results, indicating they are both viable for classification problems. However, the former, while being more simple and making the assumption of conditional independence, was able to give us a higher accuracy than the latter, even after modifying the way the data was preprocessed to improve its performance.

VI APPENDIX & REFERENCES

APPENDIX

(A) Link to Naïve Bayes notebook

(B) Link to Logistic Regression notebook

(C) Link to random search experiment sheet

(D) Link to optimal values experiment sheet

REFERENCES

[1] B. Junker (2016). Basics of Bayesian Statistics. Retrieved from https://www.stat.cmu.edu/~brian/463-663/week09/Chapter%2003.pdf

[2] T. M. Mitchell. Machine Learning. McGraw-Hill Science/Engineering/Math, 1997, pp. 154 - 200.

[3] T. Estrada. 2023. CS429/529 Naïve Bayes [PowerPoint slides].

[4] H. Sánchez. 2021. Base de Datos II (CS2702) Recuperación de la Información: Text Document Retrieval [PowerPoint slides].

[5] S.K. Gajawada. 2014. Chi-Square Test for Feature Selection in Machine learning. Retrieved from https://towardsdatascience.com/chi-square-test-for-feature-selection-in-machine-learning-206b1f0b8223

[6] S. Satpathy. 2021. Overcoming Class Imbalance using SMOTE Techniques https://www.analyticsvidhya.com/blog/2020/10/overcoming-class-imbalance-using-smote-techniques/