UNIVERSITY OF ATHENS
DEPARTMENT OF INFORMATICS AND TELECOMMUNICATIONS

Deep Learning for NLP

Student name: *Apostolos Koukouvinis*
sdi: sdi2000098

Course: *Artificial Intelligence II (M138, M226, M262, M325)*
Semester: *Fall Semester 2023*

Contents

1 Abstract **2**

2 Data processing and analysis **2**
2.1 Pre-processing . . . 2
2.2 Analysis . . . 2
2.3 Data partitioning for train, test and validation . . . 2
2.4 Vectorization . . . 3

3 Algorithms and Experiments **3**
3.1 Experiments . . . 3
3.1.1 Table of trials . . . 3
3.1.2 More Experiments . . . 3
3.2 Hyper-parameter tuning . . . 4
3.3 Optimization techniques . . . 4
3.4 Evaluation . . . 4

4 Results and Overall Analysis **7**
4.1 Results Analysis . . . 7
4.1.1 Best trial . . . 8

1. Abstract

Given a set of tweets in greek and their labels (NEUTRAL, POSITIVE, NEGATIVE) we have to create and test different models the perform sentiment analysis using logistic regression. The main idea on how to tackle the problem is to clean the data, vectorize with different approaches, train and test different models, tune their hyperparameters and finally get and evaluate the results. I have to note from the beggining, that since I use optuna which is a randomized algorithm to tune the hyperparameters, each time the result will be slightly different in terms of C and max iter in Logistic Regression. For some reason, when I try to save and run the program, it executes again and changes the hyperaparameters. So the hyperparameters in the notebook may differ a little bit by these in the report. However this will not affect our final deductions.

2. Data processing and analysis

2.1. Pre-processing

The first think we will do is to lowercase all letters, since we want, capitalized or not, words with same letters to be the "same". I tried to use nltk to remove some stopwords, but for greek language, the stopwords was not that good, it was mainly ancient greek, os I used a manually made list of stopwords of the most famous words in greek. After trying sentiment analysis, with the urls included, I saw no change in the accuracy, so there is no need for them to be there, we remove them. We also remove acute tones, #, @ and generally all the characters that dont belong to greek or english alphabet. This is done because experimentally, there were no deductions we could make from those characters, they seem to be evenly distributed among all the sentiments. After analysis on the most frequent words, we can see there is no relationship with those words and sentiments, they seem to be uniformed distributed. We also remove urls. Basically, we keep only greek and english characters, and words that have more than one character. We use a stemmer, after trying lemmatization, I saw that the results were almost the same, while it took a lot more time, so stemming is the best choice.

2.2. Analysis

After analysis on the words frequency, we could not find any relationship between the 20 most frequent words and any sentiment, so we can drop these words or just dont mind them. However, after analysis in the relationship between politcal parties and sentiments, we deducted that some parties have the the tendancy towards a specific sentiment, we will use this fact later. It is also interesting, that after the data cleaning, the top 3 most frequent words per sentiment are common.

2.3. Data partitioning for train, test and validation

The first troa; for data partitioning was using cross validation or Kfold with K = 5 to split the combined dataset (combnined dataset means valid and train set together), and tune the best hyperparameters with optuna and hold those that maximize the mean accuracy of Kfold. However this took several hours only for one of the vectorization methods, so I gave up on this method. So the partition I finally used is to use

the inital split into train set and validation set in order to tune the hyperparameters and find the best vectorization using optuna. After tuning the hyperparameters, we procceed on some experiments. In the experiments with fixed vectorization and hyperparameters I use the KFold method to check if the accuracy we got is close to other partitions of the dataset. When I find the best technique, I try some scaling, dimension reduction etc.

2.4. Vectorization

The 4 Vevtorization techniques I tried are TFidf, CountVectorizer, Doc2Vec and HashingVectorizer. For each of them I used optuna for different hyperparmaeters and solvers fot logistic regression. After experiments I found out that the best choice is count vectorizer with 'C': 0.0016049784356762025, 'max iter': 724, 'solver': 'sag'. This vectorizer is the slower one, but since we dont care about the time we need for creating the model, it is a viable choice.

3. Algorithms and Experiments

3.1. Experiments

3.1.1. Table of trials. I will display, for all the optuna studies I have done the best hyperparameters and their respective scores. Keep in mind that the score we want to optimize is F1-score.

Vectorization	$\approx C$	Max iter	Solver	Accuracy	F1-Score	Precision	Recall
TfidfVectorizer	0.0045	551	saga	40.79	40.27	41.05	40.79
Doc2Vec	0.0064	781	newton-cg	37.15	37.18	37.80	37.75
CountVectorizer	0.0016	724	sag	41.42	41.27	41.53	41.42
HashingVectorizer	0.0019	103	lbfgs	38.53	38.30	38.63	38.53

Table 1: Trials

As we can see the most competitive are count vectorizer and tfidf. We will use the count vectorizer, since we really dont care about complexity of Vectorization. We have to note, that count vectorizer is way too slow, it is about 50 times slower than hashing vectorization

3.1.2. More Experiments. For the rest of the experimetns, we have fixed vectorization method and hyperparameters of logistic regression and we will check different approaches. If we find a better approach we may try to alterate again the vectorization methodd or the hyperparameters. We check if some of the most common words are most likely to belong to one sentiment. It seems like that common words are uniformed distributed between the sentiments. Then we try to concat the text with the political party, perhaps there is a linkage between parties and the sentiments of their texts. It seems like it got worse, the accuracy droped about 0.2. Next we try some scaling with robust scaler provided by skleanr, there is no benefit from this. Next,

we try reduce dimension using sparse random projection, gaussian random projection and TruncatedSVD , no benefit at all. Since we didnt gain anything from projection, scaling and dimesnion reduction, our final model is count vectorizer with 'C': 0.0016049784356762025, 'max iter': 724, 'solver': 'sag'.

3.2. Hyper-parameter tuning

I tried to use optuna for each vectorizer and for all types of gradient based logistic regression algorithms with different hyperparameters. The optuna range is C : [0.001,1], Max Iter : [100,1000], solver : ['lbfgs', 'newton-cg', 'newton-cholesky', 'sag', 'saga']. C and Max Iter values, were decided after some experiments, from which I deducted that the best C is always in that range and max iter is normallyh between 200 and 700 with some outliers, so the range for max iter is a bit exapnded.

3.3. Optimization techniques

As said before, mainly optuna abd K-Fold.

3.4. Evaluation

As it was requested from the project, the metrics I used are f1, accuracy and precision metrics plus recall just for dipalying reasons. Based on what we can tolerate most, false negatives, false neutrals or false positives, we would try to optimize different metrics. Since nothing of this is specified on project, I tried to optimize F1 metric.

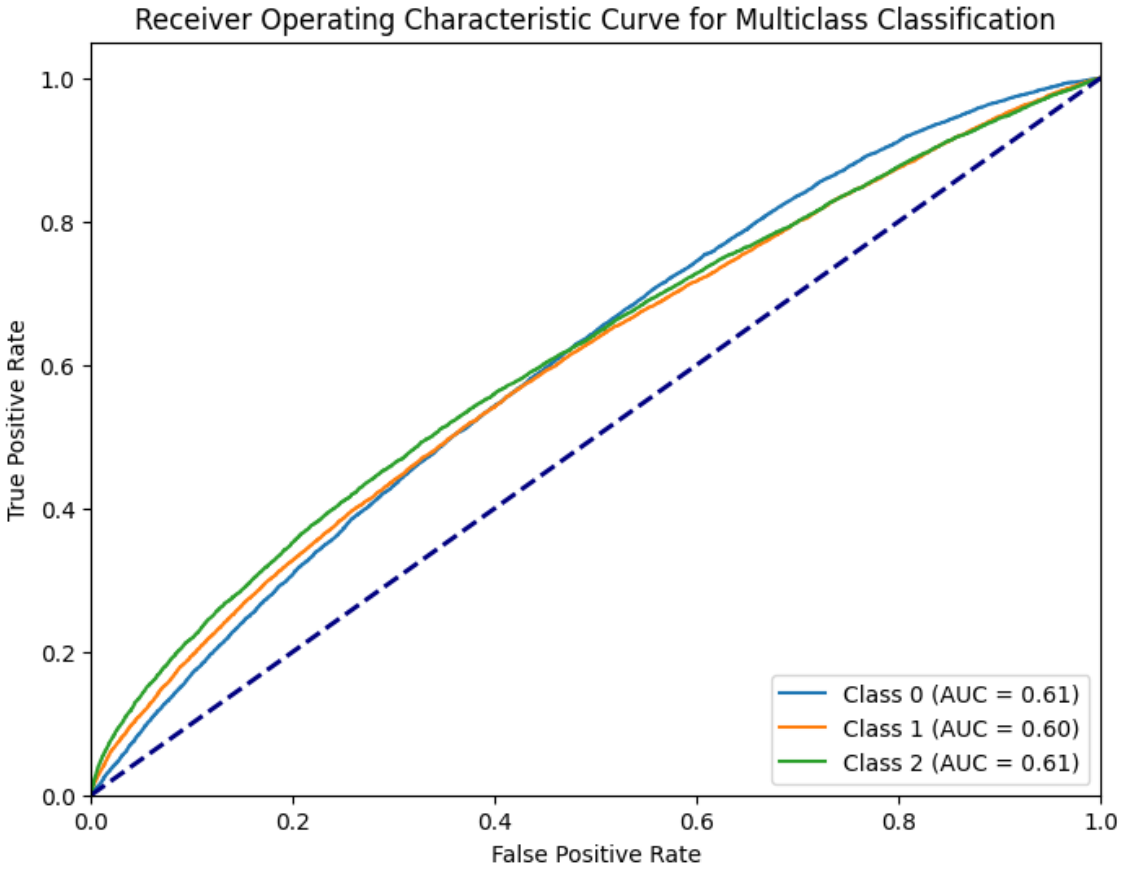

Figure 1: ROC Curve

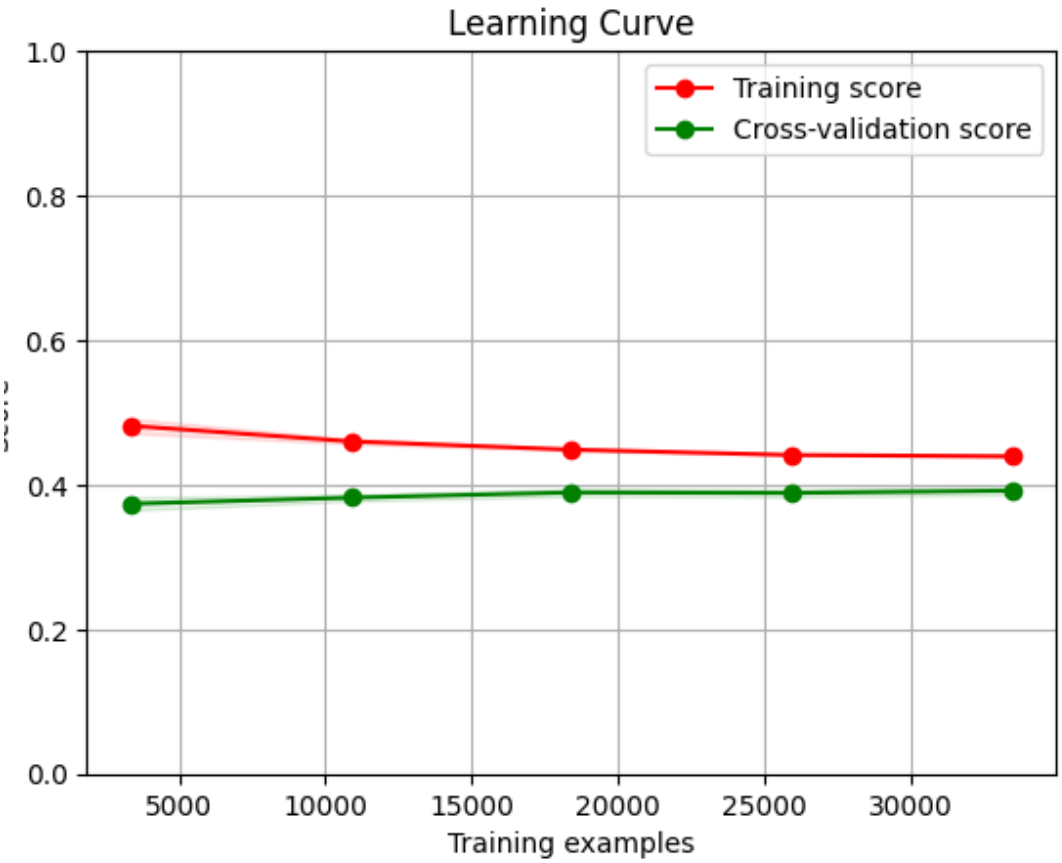

Figure 2: Learning Curve

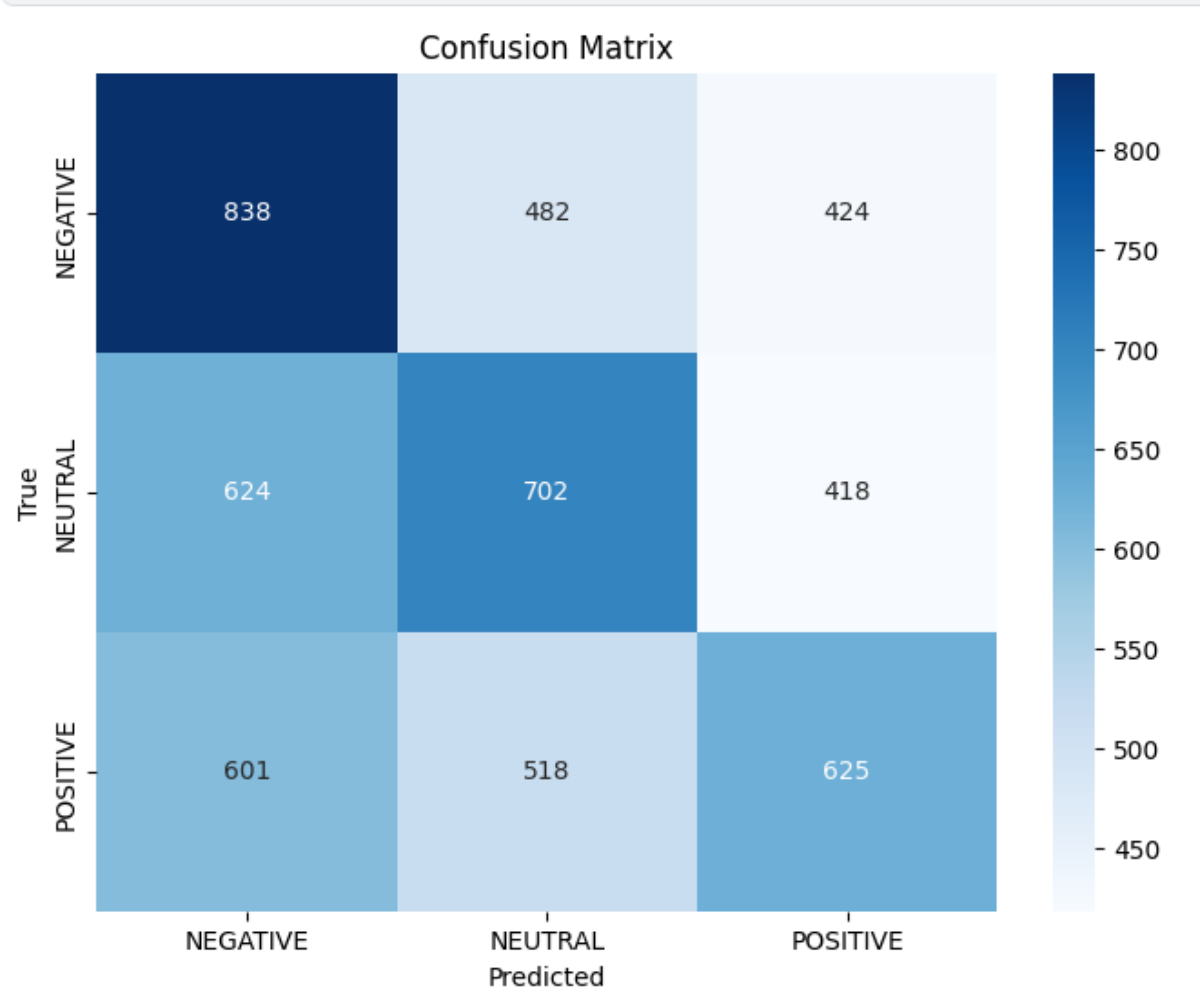

Figure 3: Confusion Matrix

I am not sure whether the model is underfitting or not. The training and validation performance, as shown in the Learning curve, are poor, but I dont think that we should blame the model for this. If 0.4 is an acceptable score, which I think it is, the model is not underfitting. For sure the model is not Overfitting since the perfromance of training and validation are very close. It is a balanced model, the curves converge to a simillar performance level, indicating that the model is learning from the data without overfitting or underfitting, the curves reach a plateau, indicating that additional data or training does not significantly improve performance.

4. Results and Overall Analysis

4.1. Results Analysis

The results are not that good, for sentiment analysis with 3 categories, approximately 40 % score is almost somthing like flipping the coin. My guess is that the dataset has a lot of misleading labels, many labels are wrong, as a colleague mentioned in piazza. Plus, Sentiment analysis has on its kernel a level of ambiguity, since not even linguists always agree on the label of a sentence. Furthermore, the text is in greek, so the li-

braries we can use are limited, for example stemming and lemmatization in greek, are very poor. I cant think, right now, of something else I could try to improve my results.

4.1.1. Best trial. The best trial is count vectorizer with 'C': 0.0016049784356762025, 'max iter': 724, 'solver': 'sag' technizque, with an f1-score close to 39.5 % for K-Folds, however other trials with train and validaion set only, were up to 41.5%. If we wanted to make some assumptions based on the hyperparameter values there would be : the small value of C indicate that the regulization is strong, and the model is preventing overfitting with such small C "steps". The numbe rof iterations is relatively small, so it converges pretty fast. So with such small C and iterations, we may deduct that our model finds a local optima relatively fast.