Assignment 1, Artificial Intelligence 2

Vyron-Georgios Anemogiannis 1115202000008

1 Introduction

In this assignment we are tasked with developing a sentiment classifier using Logistic regression.
The data set consists of IMDB reviews, (URL, Review, Score). One review is positive if it has an attached score ≥ 5, otherwise it is considered negative.
For this assignment we will use the SKLEARN and NTLK frameworks.

2 Data Processing

Immediately after getting the data from the tab separated file we start the regularisation process.
Regularisation is a very important step since it helps the model distinguish the basic characteristics of each class.
We start with simple changes to the text like:

- Removing all the apostrophes.
  By doing this step, sentences like ”I didn’t like” become ”I didnt like” helping the model distinguish positive sentences ”I did like the movie” from negative ones ”I didnt like the movie”.
  You may be asking why do this step since you are going to stem the words later. Hopefully, this step proves helpfull in distinguishing the negation inside the reviews.
- Making all the characters lowercase.
  Since the program distinguishes upper case from lower case characters, it is preferable to only keep one of them. So in the word vectors now, we have the ascii codes of only lowercase characters, decreasing the possible combinations.
- Removing

  Scrolling through the reviews, we often find the above mentioned sequence. We simply remove it since it has nothing to do with the actual review.
- Keeping only alphabetic characters
  Keeping only the alphabetic characters, gives us word vectors of 32 possible ascii codes.

Ideas for the cleaning of the text were taken from https://towardsdatascience.com/sentiment-analysis-with-text-mining

In addition to the above pre-processing technique we will use Stemming and lemmetization as described in https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html .
For this process, we will use the nltk library.
Stemming is the procedure in witch we cut the end of words. This helps us since words like ”walking” become ”walk” decreasing the possible amount of words.
Lemmatization also helps us by making all the words of the same family represented with one word. For example all the variations of the verb to be, simply become be.
The combination of these 2 techniques help make the words more uniform between reviews and by extension regularising the inputs.

Last we save the text to a json file as to not repeat the process every time we restart google colab, as well as easier porting to future assignments.

3 Data Vectorisation

In order to pass the data to the model, we first must make it a vector. In order to do that, we use SKLEARN’s TfidfVectorizer.
This method was chosen since it combines a count vectorization with the TFIDF transformer. This function is a measure of originality of a word in respect to all the other times it appears in other documents and is well documented in

https://scikit-learn.org/stable/modules/feature_extraction.html#tfidf-term-weighting.
In order to help with better regularisation we use the L_1 norm since it gives the best performance on text data according to the paper https://www.researchgate.net/publication/254051455_L1_vs_L2_Regularization_in_Text_Classification_when_Learning_from_Labeled_Features
For the parameters max df and min df all possible combinations were considered, but none beat the default.
Same applies for the ngram range. We were expecting based on https://towardsdatascience.com/leveraging-n-grams-to-ex (1,2) to yield better results than the default (1,1) but after testing we were proven wrong. As for higher values, google colab, can't handle them.

4 Logistic Regression Parameters Tuning

In this section, we test various hyper parameters for SKLEARN's Logistic Regression model https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html.
In order to decrease the randomness/variation between runs, we use everywhere the same random state.

- Tolerance:
  We tried various stopping tolerances but none made an actual difference.
- Inverse of regularisation strength: This parameter helps strengthen or lessen the penalty applied. We tried many values from 0.2 all the way to 10 and as we increased we got better and better results. It is important to note that even tho we could achieve even better results had we gone higher, the risk of overfitting became all the greater, thus we stopped there.
- Penalty: Between the options l2 and none, we chose l2, since with no penalty the risk of overfitting becomes too great.
- Solver Parameter: Between the ones that worked, lbfgs gave slightly better results, tho it must be mentioned that newton-cg game slightly worse but was much faster.
- Multi Class Parameter: We got the best results with the multinomial class.
- Max Iterations: We saw no change for iterations greater than 10, meaning the model converges quickly.

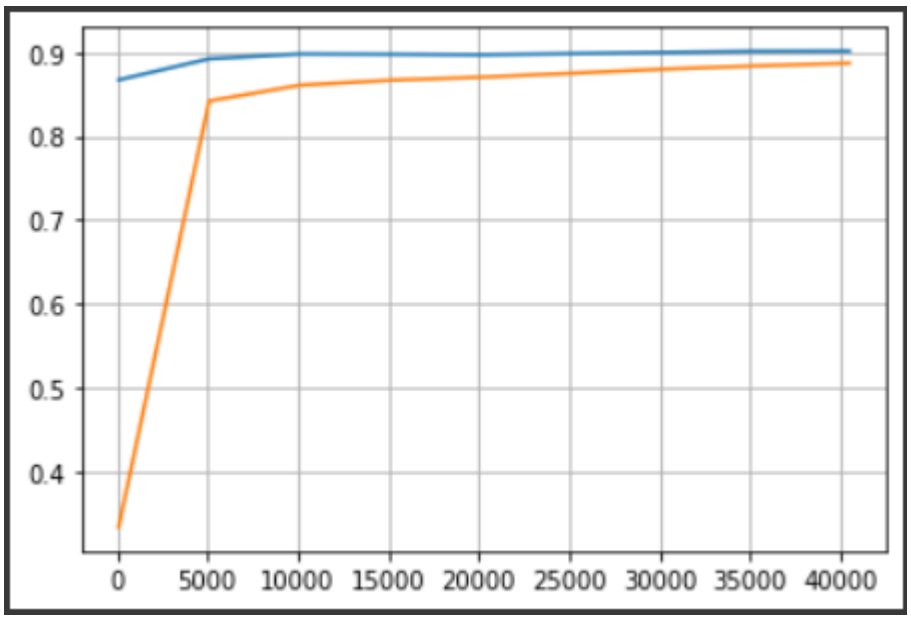
The testing was done by comparing the score cross validation returns. https://scikit-learn.org/stable/modules/cross_validation.html.
If the results using cross validation are good, it is our best chance when we are presented with a test set.

5 Testing Logistic Regression with a Validation set

Having found all the Hyper parameters we can now use our model on a train and validation set and get some metrics.
The convolution matrix along with the various scores requested are in the picture bellow.

	precision	recall	f1-score	support
0	0.89	0.88	0.88	2222
1	0.88	0.89	0.89	2279
accuracy			0.89	4501
macro avg	0.89	0.89	0.89	4501
weighted avg	0.89	0.89	0.89	4501

We can also see the learning curve (Blue:Train, Orange: Validation), counting F1 score



The results are fairly impressive using regression, and judging by the learning curve, we conclude that not much overfitting has taken place.

6 Cells to run

Not all cells are necessary to run the assignment. The cells, you need to run are the following:

- Cell 1, Importing the libraries
- Cell 2, To initialize the Paths, please, change path to reflect your system
- Cell 4, Vectorisation of the data
- You can skip all the Logistic Regression Parameter tuning
- Cell 13, Metrics on a random validation set for the model trained with the rest of the data (outside the validation set)
- Cell 14, To get the learning curve

7 Test Set

After running the above cells, we have created one more to help with the hidden test set.
In order to use it, first you have to add the path to the test set in the same format as the train set we were given.
In this cell, we do the same processing and regularisation as the Test Data. The model is trained with all the available data we had, using the parameters we tuned earlier.
Hopefully the last cell proves helpful, we would test it but we don't have a test set :(