First Assignment

Chiara Solito, VR487795

1 Prompts

The assignment consists in the development, in NLTK, OpenNLP, SketchEngine or GATE/Annie a **Naïve Bayes Classifier** able to detect a single class in one of the corpora available as attachments to the chosen package, by *distinguishing ENGLISH against NON-ENGLISH.* In particular the classifier has to be:

1. Trained on a split subset of the chosen corpus, by either using an existing partition between sample documents for training and for test or by using a random splitter among the available ones;
2. Devised as a pipeline of any chosen format, including the simplest version based on word2vec on a list of words obtained by one of the available lexical resources.

The test of the classifier shall give out the measures of **accuracy, precision, recall on the obtained confusion matrix** and WILL NOT BE EVALUATED ON THE LEVEL OF THE PERFORMANCES. In other terms, when the confusion is produced, then the value of the assignment will be good, independently of the percentage of false positive and negative results.

Deliver a short set of comments on the experience (do not deliver the entire code, but link it on a public repository like GitHub or the GATE Repo). Discuss: size of the corpus, size of the split training and test sets, performance indicators employed and their nature, employability of the classifier as a Probabilistic Language Model.

2 Delivering of the code

The code is in a public repository on GitHub:
ChiaraSolito/NLPAssignments/Assignment1. In the repository there are two versions of the code:[1]

- A notebook version, with all the comments from this document directly embedded.
- A python script version, that prints on terminal a formatted version of results.

3 Comments on the experience

3.1 Introduction

The classifier was developed with the nltk package, using its own classifier (`nltk.NaiveBayesClassifier()`) with a simple pipeline, that resembles word2vec:

1. Acquisition of data
2. Cleaning and pre-processing
   (a) Removal of non-alphanumeric characters and words
   (b) Removal of stop-words of all the languages
3. Tokenization

[1]https://github.com/ChiaraSolito/NLPAssignments/tree/main/Assignment1

4. Creation of the Bag of Words
5. Splitting dataset in train and test sets
6. Training the model
7. Testing and Querying the model

To run the code you can create the environment "nlp-lab", with the necessary requirments that you can find in the `.yaml` file, and you also have to install the genesis, udhr and gutenberg corpus from nltk.corpus.

3.2 Size of the corpus

The corpus was made mixing 11 pre-existing nltk corpus, from the **Genesis** corpus, the **Universal declaration of human rights** corpus and the **Gutenberg** corpus:

- 5 of the corpus are in english (two from Genesis, one from Udhr, two from Gutenberg)
- The other languages used are: Finnish, French (two corpus), Portuguese, German and Spanish

Informations about the dataset:

Corpus	**Size**	**Lexical Diversity**
English genesis	44764	0.06230453042623537
English web genesis	44054	0.06033504335588142
Finnish genesis	32520	0.2088560885608856
French genesis	46116	0.0803842484170353
Portuguese genesis	45094	0.08457887967357076
English-latin1 udhr	1781	0.29927007299270075
German udhr	1521	0.3806706114398422
French udhr	1935	0.2930232558139535
Spanish udhr	1763	0.3074305161656268
Emma by Jane Austen	192427	0.04059201671283136
Macbeth by Shakespeare	23140	0.17359550561797754

Total size of the corpus stands at 435115 words, of which 306166 are english.
After the preprocessing we have a total of: 15771 english words and 14317 non english words.

3.3 Size of the split training and test sets

Common split percentages include:

- Train: 80%, Test: 20%
- Train: 67%, Test: 33%
- Train: 50%, Test: 50%

Given the size of the corpus and some experiments during the developing of the code, it's been chosen to use the 67% and 33% split of the dataset.
In the end, after the preprocessing and cleaning of the data, the size of the train set is 19574 words and the size of the test set is 10514 words.

3.4 Performance indicators

Standard Metric measurements

The performance of the classifier on the constructed test set is analyzed with confusion matrix by measuring the standard metrics that are commonly used for measuring the classification performance of other classification models.
The experiments are evaluated using the standard metrics of accuracy, precision, recall and F-measure for classification. These were calculated using the predictive classification table, known as Confusion Matrix, where:

- TN (True Negative) : Number of correct predictions that an instance is irrelevant
- FP (False Positive) : Number of incorrect predictions that an instance is relevant
- FN (False Negative) : Number of incorrect predictions that an instance is irrelevant
- TP (True Positive) : Number of correct predictions that an instance is relevant
- Accuracy(ACC) - The proportion of the total number of predictions that were correct:
  Accuracy (%) = (TN + TP)/(TN+FN+FP+TP)
- Precision(PREC) - The proportion of the predicted relevant materials data sets that were correct:
  Precision (%) = TP / (FP + TP)
- Recall(REC) - The proportion of the relevant materials data sets that were correctly identified
  Recall (%) = TP / (FN + TP)
- F-Measure(FM) - Derives from precision and recall values:
  F-Measure (%) = (2 x REC x PREC)/(REC + PREC)

The F-Measure only produces a high result when Precision and Recall are both balanced, thus this is very significant.
The measures for the classifier are:

Tag	**Precision**	**Recall**	**F-measure**
english	0.5209	0.9890	0.6824
non-english	0.5200	0.0129	0.0252

Each row as to be intended as the one in which the label (Tag) is the True statement (so we'll look at the first one).

Confusion Matrix

The resultant confusion matrix was: (example from one of the experiments)

	english	non-english
english	<5427>	79
non-english	4948	<60>

(row = reference; col = test)

The classifier is good at identifying english words as such, but not as good as classifying non-english words.

3.5 Employability

This model can compute the probability of a given word being english or non-english, so it can actually be seen as a unigram model, it doesnt consider the sequences or sentences. Given the F-measure though I don't think it is actually employable as classifier.