INFO 159/259

Homework 1: Classification

Due January 30th, 2023 (11:59 PM)

In this homework, you will write code that generates features for the positive/negative sentiment classification of movie reviews, and train a model on those features to predict the sentiment of new data. Example of selected movie reviews with labels:

Label	Review
pos	What can I say, it's a damn good movie. See it if you still haven't. Great camera works and lighting techniques. Awesome, just awesome. Orson Welles is incredible 'The Lady From Shanghai' can certainly take the place of 'Citizen Kane'.
neg	Can I give this a minus rating? No? Well, let me say that this is the most atrocious film I have ever tried to watch. It was Painful. Boringus Maximus. The plot(?) is well hidden in several sub-levels of nebulosity. I rented this film with a friend and, after about thirty minutes of hoping it would get better, we decided to "fast forward" a little to see if things would get any better. It never gets better. This film about some dude getting kidnapped by these two girls, sounds interesting, but, in reality, it is just a bore. Nothing even remotely interesting ever happens. If you ever get the chance to watch this, do yourself a favor, try "PLAN NINE FROM OUTER SPACE" instead "

The notebook for this assignment can be found here:

https://github.com/dbamman/nlp23/blob/main/HW1/HW1.ipynb

The notebook contains code which performs all the data parsing, model training, and evaluation for you. Once you hit the "Open in Colab" button at the top, **save a copy** in your Google Drive or Github before working on the file.

During training, a basic logistic regression model (from the scikit-learn package) is trained with your created features to predict each review's sentiment. Your ultimate goal for this assignment is to featurize the text creatively and optimize the accuracy of the model on the test data.

Your main tasks are to implement four features:

1. A bag of words feature. Fill out the **`bag_of_words`** function in the notebook to represent a document through its bag of words. Represent each **lowercase** word tokenized with the NLTK **`word_tokenize`** function through its binary value. Remember that the feature value for a word that shows up in a review should be 1, no matter how many times it is mentioned. Please keep your code between the **`#BEGIN/#END SOLUTION`** flags.

2. Create **three** different classes of features. Implement them in the `feature1`, `feature2` and `feature3` functions. Create features that you think would perform better than bag of words, and assess their independent performance on the development data. Describe your reasoning in your Colab notebook and include accuracy scores on the dev.txt data (printed in the Colab output). **You must report your reasoning and development scores in the table provided.** It is not required that your features actually perform well, but your justification and reasoning should be defensible. We are looking for thoughtful insights as to why your original features *should* outperform the simple bag of words implementation. Extra credit points will be awarded for the most creative features (features that few other students use and that are reasonably well-performing).

For more feature ideas consult SLP Ch. 5 and lecture slides, and please do keep the following constraints in mind:

- You should only change the code for `bag of words`, `feature1`, `feature2`, `feature3` and `combiner_function` along with any code you need to support those functions. You must use the classifier provided.
- You are free to use external dictionaries (such as LIWC or AFINN); if you use these, **upload them to Gradescope as well with your HW1 .ipynb.**
- Do not use pre-trained word vectors (including static embeddings like word2vec or contextual embeddings like BERT); we will cover these later.
- Do not use the predictions of any other supervised model on this data as a feature for logistic regression (e.g., do not train a separate CNN on this data, make predictions on the dev data using the CNN, and then treat your CNN prediction as a feature).
- Do not alter {train,dev,test}.txt or find additional sources of training data. This homework is focusing on how you *represent* your data, not other aspects of the training regime.
- Do not import any other additional libraries.

At the end of the notebook we've also included several other methods that you can use to interrogate your model; please feel free to use these to assess what kind of mistakes your current model is making in order to help brainstorm new features.

Once you've defined your features, described them, and evaluated their performance on the development data, it's time to make predictions on the test data. Adapt the `combiner_function` cell to include whichever features you like and execute it to make predictions on the test data. This will create the file `combiner_function_predictions.csv`; download this from Colab (using e.g. the file manager on the left panel) and submit it to Gradescope, along with your HW1 notebook. The five highest performing systems (revealed after the submission deadline) will receive extra credit for the assignment!

1 How to Submit

There are two different deliverables, each to be submitted on Gradescope.

- Submit to GradeScope "HW 1 Test Set Predictions":
  - **combiner_function_predictions.csv**. You can find files associated with the notebook if you click on the folder icon on the left-side panel of your Colab notebook. Please don't alter the file name.
- Submit to GradeScope "HW 1 Code":
  - Download your Colab notebook as an .ipynb file (File → Download .ipynb)
  - Submit **HW1.ipynb**. The file must be named in this way for the Gradescope autograder.
  - If you used any external dictionaries, please attach them as well for this submission.