CSE 842: Natural Language Processing
Homework 1 Assignment

Due Date: Sept. 26, 2019 at noon

Dataset

For this assignment you will use *Sentiment Polarity Dataset Version 2.0* (Pang, Lee, Vaithyanathan). The original version of this dataset was released with Thumbs Up? Sentiment Classification using Machine Learning Techniques. It can be downloaded from the author's website as polarity dataset v2.0 or you can download it or automatically load it into NLTK as movie_reviews (corpus #23). Please read their README file to understand the data format and how to divide the data into folds for evaluation.

Problem 1: Naïve Bayes Classifier

Description: For this problem you will need to implement and evaluate a Naïve Bayes classifier.
Use *unigrams* as the features for training this classifier. Complete the following steps:

1. Create a configuration for three-fold cross validation. In each run, assign two folds out of the three as the training data and assign the remaining fold as the testing data.
2. In each run, complete the following:
    a. Use the training data to build a vocabulary set. Treat each punctuation mark as a word.
    b. Train the NB model, i.e., calculate *P(w|c)* and *P(c)*. If needed, use Laplace (add-1) smoothing (where k=1) or add-k smoothing (where k=0.1).
    c. Apply the NB model to each document in the testing set to predict *positive* or *negative*.
    d. Compare your model's predicted class with the ground truth of the testing data to calculate F1, precision, recall, and accuracy.

Code Requirements:

1. For the training step please use the following command line format: *train fold1 fold2*
   This means to train the NB model based on fold1 and fold2. Running this command will create a file containing all model parameters. You can use any format of your choice to save the learned model parameters, but please name the file *yournetid_params*. Remember that calculations should be based in log space.
2. For the testing step please use the following command line format: *test fold3*
   This should print out a statement summarizing the evaluation results, i.e., "Precision: ##, Recall: ##, F1: ##, Accuracy: ##". You will need to read in the parameter file before applying the model to the testing data. (Optional: if you want to skip the intermediate step of writing/reading a separate parameter file, please note this in your written report.)

What to submit:

1. Your Python code implementing the NB model, including training, testing, and evaluation; name the file *yournetid_hw1p1.py*.
2. In the written report, include any information I may need to run your code (e.g., what version of Python you used, how the dataset is accessed, format of your parameter file). Also include any extra steps you took with a brief description justifying your actions (e.g., used k=0.1 smoothing for X reason or removed stop words for X reason). Report the *average* of your evaluation metrics across all runs.

PROBLEM 2: NLTK & SKLEARN

Description: For this problem, I want you to gain familiarity working with NLTK and scikit-learn. We're going to use this Movie Reviews Sentiment Analysis Tutorial and make modifications. Note that this tutorial uses a Jupyter notebook setup, but that is not required for this assignment. Complete the following steps:

1. Using the tutorial, implement a NB model for sentiment analysis and calculate the precision, recall, F_1, and accuracy for the NB model. The number of folds/runs and any other modifications you make here is up to you, but must be justified in the written report.
2. Implement an SVM model for sentiment analysis using the same dataset and similar setup. For the SVM test 2 different types of features: *unigrams as bag-of-words* and the *TF-IDF* (frequency scores that reflect how important a word is in a document). Calculate (average) precision, recall, F_1, and accuracy for both models. Again, you can choose the number of folds and other parameters, but your choices must be justified in the written report.

Code Requirements: Please name your Python files as *yournetid_hw1p2_nb, yournetid_hw1p2_svmbow,* and *yournetid_hw1p2_svmtf*. If you choose a setup with all three models within one file, please name the file *yournetid_hw1p2_all* and note what I need to do to run the code in the written report.

What to Submit:

1. Your Python files implementing NB, SVM with bag-of-words features, and SVM with TF-IDF features.
2. In the written report, include any information I will need to run your code. Also, include a discussion of the parameters or modeling setup you used and justify why you chose your approach. Report the average results for all 3 models.

BONUS PROBLEM (OPTIONAL)

This problem must be completed on your own. Choose the strings based on your last name. Calculate the minimum edit distance between the following two strings.

Last names A – M: "implements" & "merriment"
Last names N – Z: "merriment" & "increments"

Use the following distances: 0 for a match, +1 for insertions and deletions, +3 for replacement. Report your minimum edit distance and optimal backtrace path (which characters are inserted/deleted/replaced). You can either type up your table or take a picture if you calculate it by hand and attach that to the PDF.

SUBMISSION SUMMARY

Please submit the Python files that solve problems 1 & 2 and *one* written report in PDF format named *yournetid_hw1_report.pdf*.