CS578	**Fall 2021**
HW 3	
Handed Out: November 19, 2021	*Due: Dec 5, 2021*

Instructions for submission

1. Conceptual Part: **Submit a single PDF on Gradescope with all your answers. Make sure you select the page corresponding to the beginning of each answer, else points might be deducted.** Your homework must be typed and must contain your name and Purdue ID. If you are using a late day, please indicate it at the top of the document. Do not send the instructors emails aboout Late Days.
2. Coding part: To submit your assignment, log into `data.cs.purdue.edu` (physically go to the lab or use ssh remotely) and follow these steps:
   (a) Place all of your code (`Execution.py`, `Logistic.py`, `Ensemble.py`, and any other files you used) in a folder titled `cs578-hw3`. **Make sure your folder is called this!**
   (b) Change directory to outside of `cs578-hw3/` folder (run `cd ..` from inside `cs578-hw3/` folder)
   (c) Execute the following command to turnin your code: `turnin -c cs578 -p hw2Fall2021 cs578-hw3`
   (d) To overwrite an old submission, simply execute this command again.
   (e) To verify the contents of your submission, execute this command: `turnin -v -c cs578 -p hw3Fall2021`.
   Do not forget the `-v` option, else your submission will be overwritten with an empty submission.

Questions

1. We define a concept space C that consists of the union of k disjoint intervals in a real line. A concept in C is represented therefore using $2k$ parameters: $a_1 < b_1 < a_2 < b_2 < ... < a_k < b_k$. An example (a real number) is classified as positive by such concept iff it lies in one of the intervals. Give the VC dimension of the hypothesis space H (and prove its correctness).
2. The Gradient Descent (GD) algorithm:
   (a) Write in one sentence: what are the hyper parameters of the GD algorithm.
   (b) Write in one sentence: what is the difference between $L1$ and $L2$ regularization.
   (c) Write down the gradient descent algorithm applied to hinge loss with $L2$ regularization.

(d) The same as above, but using the $l1$ regularization. **Note**: the $L1$ norm is convex but not differentiable.

Programming Assignment

This assignment is based on homework 2, the binary sentiment classification task. In this part of the assignment you are required to **improve the logistic regression model** that you implemented in HW2. Specifically, you will have to improve the model by using more expressive features. Your classifier will be defined over unigram, bigram, and trigram word features. This is likely to result in a model that overfits. In order to combat that you need to use one, or several methods such as ensemble and adding a regularizer to the objective function.

1. **Dataset/Task** The dataset you will use and the requirements for splitting the dataset are the same as homework 2. Your job will be predicting whether the input document is positive (1) or negative (0). You will still use the same csv file that contains all the data corresponding to this task, i.e., data.csv.

   As in HW2, you also need to split the data into a train and a test data sets. The splits are determined by you. Remember you should not use the **test set** in picking the optimal hyperparameter values. Take a look at the function `split_dataset` in `Execution.py`.

   More details can be found in the HW2 handout.

2. **N-gram Features** In HW2, you were required to implement the bag-of-words feature selection approach. However, you may know this approach has several drawbacks such that it does not preserve sentence structure and syntax that could be useful as features in a model.

   In this assignment, you are required to implement the n-gram model to extract word features. More details can be found in: `https://web.stanford.edu/~jurafsky/slp3/3.pdf`. You should implement cases when $n = 1, 2, 3$, i.e., **unigram**, **bigram**, and **trigram**.

   As an example of bigram, in the sentence "this laptop is awesome", the features are "this laptop", "laptop is" and "is awesome".

3. **SGD with Regularization** In HW2, you may have implemented the logistic regression algorithm using the stochastic gradient descent (SGD) optimizer. In order to reduce overfitting and improve the model's generalization performance, in this assignment, you need to include regularization in your logistic regression model. You are required to implement **both** L1 and L2 regularization methods and compare them in your report. The hyperparameter λ before the regularization term needs to be tuned. Simply report the performance of the best hyperparameter.

4. **Ensemble** An ensemble of classifiers is a set of classifiers whose individual decisions are combined in some way, typically by weighted or unweighted voting, to classify new

examples with the goal of improving accuracy and reliability. As combining diverse, independent opinions in human decision-making to pursue a more protective mechanism, the ensemble model provides an efficient way to improve accuracy and robustness over single model methods. Ensemble methods provide a huge practical usefulness in that it has the promise of reducing and perhaps even eliminating some key shortcomings of standard learning algorithms.

In this assignment, you are required to implement an ensemble model to predict the binary sentiment label. You have learned boosting and bagging during the lecture. Here you you **only need to implement bagging model**. Please note that the **basic classifier** of the ensemble model should be **logistic regression**.

5. **Implementation Details** You are given a starter code `Logistic.py` which is similar as in HW2. The differences are: 1) you need to implement the unigram, bigram, and trigram models for the feature extraction; 2) you need to implement the L1 and L2 regularizers. Please note that 1) and 2) should be implemented **together**. The hyperparameters that you need to tune are: learning rate, maximum number of epochs, feature extraction method (unigram, bigram, or trigram), regularizer (L1 or L2), and λ (the coefficient of the regularizer). In order to save your time, for determining which feature extraction method should be used, you can fix other hyperparameters first and compare unigram, bigram and trigram. Similarly, for determining which regularizer should be used, you can fix other hyperparameters and compare L1 and L2 regularizers. After the feature extraction method and the regularizer are chosen, you are required to use **5-fold cross validation** for tuning other parameters. Finally, remember to use your optimal hyperparameters to update the default values for them in the `train` function arguments. We will only check the performance of your best model during grading.

   You are also given a starter code `Ensemble.py`, in which you should implement the bagging model that uses the logistic regression as the basic classifier. 5-fold cross validation is not required here. The feature selection method should be the same as what you choose in the implementation of the logistic regression model. The hyperparameter you need to tune is **the number of weak classifiers**. Remember to use the best value to update the `num_clf` argument in the `train` function.

   Please note that although you have the freedom to modify most of the starter code and implement it how you feel comfortable, in order to streamline the process of evaluation, the `train()` and `predict()` functions may not change.

6. **Evaluation** Similar as HW2, you are also given the evaluation program called `Execution.py`. In this program, there is one function which you must program which is `split_dataset`. This function takes in the full `.csv` dataset file and you must specify how to split it into train and test sets.

   The primary way you will be evaluating the performance of both your models will be through classification accuracy which is already implemented in Execution.py. When you run Execution.py, it will produce these accuracies on the train and test sets. Please look at the code for further information.

You can simply run Execution.py from the command line in the following way:

```
python3 Execution.py
```

It is important to note that the python command assumes that a python3 interpreter is being used. Additionally, it is important that `Execution.py, Logistic.py`, and the `.csv` files stay in the same folder because `Execution.py` will only load the data files when they are in the same directory.

It is important to not alter the code for specific sections in this program because these exact functions will be used to evaluate your code and model. More detailed information is contained in the comments of the starter code.

7. **Report** You will need to submit a report detailing the results of your models. Please have the following sections labeled in the order specified. Below are the following requirements for the report:

   (a) Explain how you split the dataset and why you selected those splits for the training and test sets.

   (b) For unigram, bigram, and trigram cases, answer two questions:

   - What is the number of features that were actually generated during training?
   - What is the size of the feature space? (You should count the features that were not included)

   (c) For the logistic regression model, what feature extraction and regularizer did you use? After answering this, you need to state what are the hyperparameters you tuned in your model. For each hyperparameter, plot learning curves based on training and validation performance as a function of the hyperparameter. There is no requirement that you should put training and validation together or in separate figures.

   (d) Describe how you implement the ensemble model. Similarly, state what the hyperparameters are and plot learning curves as a function of hyper-parameter assignments.

   (e) Report the optimal values for the hyperparameters and explain how you arrived at those values. Report the accuracy of your logistic regression model and ensemble model on the test set with those selected hyperparameter values. Compare them with your results in HW2 and explain how the feature selection methods, regularizers and the ensemble method impact your model.