CSCI4360/6360 - HW3

Instructor: Dr. Ninghao Liu (ninghao.liu@uga.edu)

March 17, 2024

– Upload two files to eLC (failing to do so will receive 20% penalty):
1) a zip file named "*YourID_HW3PQ.zip*" containing your programs for Question 2.
2) a pdf file "*YourID_HW4.pdf*" containing your report for Question 2.
– Due Date: April 2, 2024

1 Text Retrieval (35pt)

Given the following documents:

- Today the Dawgs won!
- Dawgs have won today
- Dawgs the champion!
- The Dawgs news today.

1) (5 pt) Preprocessing: Conduct punctuation removal, stop word removal (assume the stop word list is [the, a, of, to, for, have]), and case-folding on the documents. Output each document after preprocessing as a term sequence.

2) (5 pt) Construct the term-document incidence matrix based on the result of the question above. Report the resultant incidence matrix.

3) (10 pt) Implement TF-IDF on the term-document incidence matrix. Output the resultant TF-IDF matrix.

5) (10 pt) Graphically illustrate the inverted index you will create for the documents. Given the query "dawgs won", make use of the inverted index we just build to retrieve the matched documents. Illustrate the query processing procedure in details.

6) (5 pt) Explain the advantages of inverted index over term-document incidence matrices for organizing texts?

2 Model Implementation (65pt)

2.1 Introduction

In this homework, we will develop several machine learning models for a flower classification task. The models include *Decision Tree*, *Naïve Bayesian Classifier (NBC)*, *Logistic Regression*, and *Multilayers Perceptron (MLP)*.

2.1.1 Dataset

The Iris dataset is a fundamental database in pattern recognition, which was first proposed by R.A. Fisher (1950). The goal of this dataset is to predict the class of iris plants (*Setosa, Versicolour, and Virginica*) by giving four attributions (*sepal length, sepal width, petal length, and petal width*). The detailed information and the source data are available at UCI Machine Learning Repository: Iris Data Set.

2.1.2 Setup

In this homework, we focus on two of the three classes (*Versicolour and Virginica*), which are NOT well linearly separated from each other. Our dataset includes 100 samples (50 instances for each class) in total. This dataset could be found in *./iris.csv* stored in the comma-separated values (CSV) format. In prep-rocessing, we binarize each feature to support decision trees and NBC, while the normalization is applied to features for the other two models. Finally, we randomly split the dataset into training set and testing set, where the training set keeps 70% samples by default. All these procedures have been implemented as the function *prepare_dataset(filepath)* defined at the file *./base/utils.py*.

2.1.3 Metrics

Since we target a binary classification task, we choose *accuracy*, *F1-score*, and *ROC_AUC* scores as metrics to measure model performance. These metrics are developed as function *scoring(y_pred, y_prob)* at the file *./base/utils.py*.

2.1.4 Pipeline

We will first train the models using the training set. Then, we will let the models predict the labels of test instances. The metrics mentioned above will be used to evaluate the performance on the testing set. We have implemented this pipeline at *./run.py*. More details about training (e.g., optimizers, learning rates, batch sizes) have been abstracted as base classes *StatisticClassifier* and *GradientClassifier* in *./base/module.py*, and these base classes will only provide *clf.fit(X, Y)* and *clf.predict(X)* functions to the public.

2.1.5 Requirements

You need to make sure you have installed these packages before working on the codes: **NumPy**, **PyTorch**, **Scikit-Learn**.

2.2 Assignments (what you need to do)

We have implemented some parts of the four models in *./bayes.py*, *./linear.py*, *./nn.py*, and *./tree.py*. For each model, it inherits from either *GradientClassifier* or *StatisticClassifier*. Thus, you only need to implement the public function *clf.forward(x)* predicting the score of single instance x. Since the training procedure differs between the two statistic methods, the Naive Bayesian and Decision Tree classifiers should have one more private function *clf._fit(X, Y)*.
1) (15 pts) **Complete the codes in "./linear.py" to implement Logistc Regression (i.e., a linear model)**.
2) (15 pts) **Complete the codes in "./bayes.py" to implement NBC**.
3) (15 pts) **Complete the codes in "./tree.py" to implement decision trees**.
4) (20 pts) **Complete the codes in "./nn.py" to implement MLP model (i.e., a multi-layer neural network)**.

After completing the codes, please run "python run.py" and **report** your results.