Assignment 2: Scalability of Support Vector Machines

Purposes

- Understanding the most popular Support Vector Machines (SVMs).
- Designing a basic algorithm to generate linearly separable datasets.
- Understanding and implementing the basic linear SVM with soft margin.
- Understanding and using the built-in SVMs in scikit-learn.
- Evaluating the scalability of SVMs on linear separable datasets.

Implementation Tasks

1. (4 points) Implement a Python class named `LinearSVC` which learns a linear Support Vector Classifier (SVC) from a set of training data. The class is required to have the following components:
   - (1 point) A constructor `__init__` which initialize an SVC using the given learning rate, number of epochs and a random seed. (Similar to the perceptron class in our textbook.)
   - (1 point) A training function `fit` which trains the SVC based on a given labeled dataset. We consider the soft-margin SVC using a hinge loss. You are required to use L2-regularization and expose the regularization parameter as a hyperparameter.
   - (1 point) A function `net_input` which computes the preactivation value for a given input sample.
   - (1 point) A function `predict` which generates the prediction for a given input sample.

   The structure is very similar to the perceptron and Adaline classes in our textbook.
2. (2 points) Write a Python function `make_classification` which generates a set of linearly separable data based on a random separation hyperplane. We learned that an $(d-1)$-dimensional hyperplane can be defined as the set of points in $\mathbb{R}^d$ satisfying an equation $\bar{a}^T\bar{x} = b$, i.e., $\{\bar{x} \in \mathbb{R}^d \mid \bar{a}^T\bar{x} = b\}$. For simplicity, we assume that $b = 0$, then the hyperplane can be determined by a random vector $\bar{a}$. We use this idea to design the following algorithm to generate random data which are linearly separable regarding to any number and dimension:
   - *Step 1.* Randomly generate a d-dimensional vector $\bar{a}$.
   - *Step 2.* Randomly select n samples $\bar{x}_1, \ldots, \bar{x}_n$ in the range of $[-u, u]$ in each dimension. You may use a uniform or Gaussian distribution to do so.
   - *Step 3.* Give each $\bar{x}_i$ a label y_i such that if $\bar{a}^T\bar{x} < 0$ then $y_i = -1$, otherwise $y_i = 1$.

   Therefore, your function should have the following parameters that should given by the user: d, n, u, and a random seed for reproducing the data. You need to additionally subdivide the dataset to a training dataset (70%) and a test dataset (30%). You may use the scikit-learn function to do so, but make sure that you specify the random seed such that the subdivision is reproducible.
3. (2 points) Investigate the scalability of the LinearSVC class you have implemented. Using the dataset generator developed in the previous task, you may produce random datasets regarding to the 9 combinations of the following scales: $d = 10$, 50, 100 and $n = 500$, 5000, 50000. You may assign a large constant such as 100 to u. (Please feel free to slightly adjust the scales according to your computer's hardware.) Evaluate the time cost and loss convergence of your linear SVC on the 9 datasets. The comparison should be given by tables along with explanations.
4. (2 points) Read the scikit-learn documentation for SVMs. We are going to investigate the performance of solving primal and dual problems for linear classification. You may use the 9 datasets in the previous task. The easiest way to reuse a dataset is to keep all data in a file. In this task, you are required to use the built-in class `LinearSVC` in `sklearn.svm`. You may use the hinge loss and the default value for the regularization parameter. For each dataset, compare the time costs and loss convergences of training a linear SVC by solving the primal and dual problems.

Report

You are required to write a report for this assignment. It should include the following parts:

- For Task 1, explain the data structures and algorithms in your implementation.
- For Task 2, you should present a demo for $d = 2$, $n = 100$. A figure should be presented to show the distribution of the linearly separable data that are randomly generated. You may decide the value of u.
- For Task 3, tables and figures should be presented for the time costs and loss convergences. Please also explain your observations.
- For Task 4, tables and figures should be presented for the time costs and loss convergences. Please also explain your observations.