CSCI 5521: Introduction to Machine Learning (Spring 2024)[1]

Homework 4

Due date: Apr 24, 2024 11:59pm

1. (**30 points**) Suppose we use a linear SVM classifier for a binary classification problem with a set of data points shown in Figure 1, where the samples closest to the boundary are illustrated: samples with positive labels are (-2, -1), (0, 1), (-1, 1), (-1, 2), and samples with negative labels are (1, 0), (2, 0), (0, -1), (1, -1).

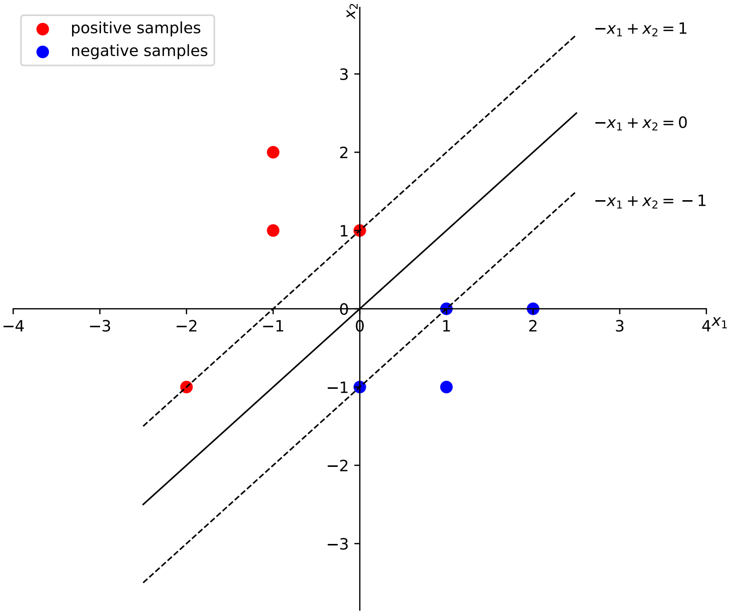

Figure 1: For a binary classification problem where positive samples are shown red and negative ones shown blue. The solid line is the boundary and dashed ones define the margins.

   (a) List the support vectors.

   (b) Pick three samples and calculate their distances to the hyperplane $-x_1 + x_2 = 0$.

   (c) If the sample (-1, 1) is removed, will the decision boundary change? What if we remove both (1, 0) and (0, -1)?

   (d) If a new sample (0.5, -0.5) comes as a positive sample, will the decision boundary change? If so, what method should you use in this case?

[1] Instructor: Catherine Qi Zhao. TA: Xianyu Chen, Amarachi Nzeukwu, James Yang, Yifeng Zhang. Email: csci5521.s2024@gmail.com

(e) In the soft margin SVM method, C is a hyperparameter (see Eqs. 14.10 or 14.11 in Chapter 14.3 of the textbook). What would happen when you use a very large value of C? How about using a very small one?

(f) In real-world applications, how would you decide which SVM methods to use (hard margin vs. soft margin, linear vs. kernel)?

2. (**30 points**) Table 2 shows data collected on a person's decision to attend an academic workshop at their university depending on various factors, including their interest in the workshop topic, the time of the day of the workshop, as well as their past attendance at similar workshops. Answer the following questions:

Interest in Topic	**Time of Day**	**Past Attendance**	**Attend?**
High	Afternoon	≥ 2	Yes
Low	Afternoon	1	No
Medium	Evening	0	No
Medium	Afternoon	1	Yes
Low	Evening	0	No
Medium	Morning	0	No
Low	Morning	≥ 2	Yes
Medium	Morning	≥ 2	Yes
High	Evening	1	No
Low	Evening	1	Yes
High	Evening	0	No
Low	Afternoon	0	No
High	Morning	0	No

(a) We wish to build a decision tree to help decide if the person will attend the workshop. Draw the decision tree that fits this data and show how to calculate each node split using entropy as the impurity measure.
Note: If the entropy is the same for two or more features, you can select any of the features to split.

(b) Based on the decision tree, will the person attend the workshop if they have a **Low** level of interest, the time of day is **Evening**, and the person has attended **more than 2** similar workshops in the past?

3. (**40 points**) In this programming exercise you will implement a Decision Tree for optical-digit classification. You will train your Decision Tree using the `optdigits_train.txt` data, tune the minimum node entropy using `optdigits_valid.txt` data, and test the prediction performance using the `optdigits_test.txt` data. For each file, the first 64 columns correspond to the binary

features for different samples while the last one stores the labels. The minimum node entropy is used as a threshold to stop splitting the tree, and set the node as a leaf. You are going to implement the following helper functions:

functions	descriptions
metric(label)	compute the entropy or gini based on a list of labels.
combined_metric(left, right)	weighted combine two metrics from two lists.
generate_tree(data, label)	recursively calls itself to build up the decision tree.

Table 1: List of functions to be implemented. Please refer to the code comments for detailed implementations.

(a) Implement a Decision Tree with the minimum node entropy θ=0.01, 0.05, 0.1, 0.2, 0.5, 1.0, 2.0 and 4.0. The minimum node entropy is used to determine the leaf nodes, *i.e.,* a node is a leaf node if its node entropy is lower than the selected minimum node entropy.
Report the training and validation accuracy by different θ.
Which θ performs best? Report the accuracy on the test set using this θ.
What can you say about the model complexity of the Decision Tree, given the training and validation accuracy? Briefly explain.

(b) Apart from the entropy, another category of Decision Tree algorithm leverages a different metric: Gini Index. The following is the formula of Gini Index:

$$Gini = 1 - \sum_i p_i^2, \tag{1}$$

where p_i is the sample as the p_i in entropy that denotes probability of class i. You are going to implement `gini_index()`, and run the same set of experiments again. **What can you say about the performance? Are they more or less the same, or which one is better. Briefly explain.**

We have provided the skeleton code `MyDecisionTree.py` for implementing the algorithm. `MyDecisionTree.py` is written in a *scikit-learn* convention, where you have a *fit* function for model training and a *predict* function for generating predictions on given samples. To verify your implementation, call the main function `hw4.py`.

More details can be found in comments of the source files.

Submission

- **Things to submit:**

1. hw4_sol.pdf: a document containing all your answers for the written questions (including answers for problem 3).
2. `MyDecisionTree.py`: a Python source file containing your implementation of the decision tree algorithm. Use the skeleton file `MyDecisionTree.py` found with the data on the class web site, and fill in the missing parts (see comments for details about each function) . **The *fit* function should recursively construct the decision tree with the auxiliary function *generate_tree*.** The *predict* function should take features as inputs and return the predicted class labels.

- **Submit**: All material must be submitted electronically via Gradescope. **Note that There are two entries for the assignment, *i.e.,* Hw4-Written (for hw4_sol.pdf) and Hw4-Programming (for a zipped file containing the Python code), please submit your files accordingly.** We will grade the assignment with vanilla Python, and code submitted as iPython notebooks will not be graded.