CSCI 5521: Machine Learning Fundamentals (Fall 2022)[1]

Homework 4 (Tuesday, Nov. 29)

Due on Gradescope at 11:59 PM, Friday, Dec. 9

Instructions:

- This homework has 3 questions, 100 points.
- For each question, especially programming, please provide a detailed explanation/calculation to avoid point reduction.

1. (**20 points**) Below table shows data collected on a person's decision to go for a run or not, depending on the weather conditions that day. Answer the following questions:

Outlook	Temperature	Humidity	Run?
Rainy	Mild	High	No
Overcast	Cool	Normal	No
Rainy	Cool	High	No
Overcast	Mild	Normal	No
Rainy	Cool	Normal	No
Sunny	Hot	High	Yes
Sunny	Mild	Normal	Yes
Rainy	Cool	High	No
Sunny	Cool	Normal	Yes
Rainy	Hot	High	No
Sunny	Mild	Normal	Yes
Rainy	Mild	Normal	Yes
Sunny	Mild	Normal	Yes
Sunny	Hot	High	Yes
Sunny	Cool	Normal	Yes

   (a) For the above table, we wish to build a decision tree classifier to help decide if the runner will go for a run tomorrow. 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 features to split.

   (b) Based on the decision tree, if tomorrow's outlook is Overcast, the temperature is Cool, and the Humidity level is High, will the runner go for a run?

Programming assignments: The following two problems involve programming. We gave you skeleton files; you don't need to modify Q2.py, Q3.py and simulate_data.py.

2. (**30 points**) In this programming exercise, you will implement a Univariate 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.

[1]Instructor: Kshitij Tayal (tayal@umn.edu). TA: Xianyu Chen, and Yoshitaka Inoue (csci5521@umn.edu).

(a) Implement a Decision Tree with the minimum node entropy $\theta = 0.01, 0.05, 0.1, 0.2, 0.4, 0.8, 1.0,$ and 2.0. Report the training and validation error rates by different θ. What θ should you use? Report the error rate on the test set using this θ.

(b) What can you say about the model complexity of the Decision Tree, given the training and validation error rates? 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 in q2.py. More details can be found in comments of the source files.

3. (**50 points**) In this programming exercise, you will implement a Kernel Perceptron using Radial Basis Function (RBF) kernel. Similar to the perceptron algorithm, which classifies data into binary classes with $y = \text{sign}\left(w^T x\right)$, the Kernel Perceptron makes the decision with the function $y = \text{sign}\left(\sum_t \alpha^t r^t K\left(x^t, x\right)\right)$, where x^t and r^t are training samples and their labels, x is the sample to be classified, α^t is a counter that records the time x^t is misclassified during training, and $K(x^t, x) = exp\left(\frac{-\|x^t - x\|^2}{2\sigma^2}\right)$ is the RBF kernel (σ is a parameter). During training, the counter α^t is initialized as 0 and will be updated as $\alpha^t = \alpha^t + 1$ when $y^t \neq r^t$ for the current iteration.

   (a) Test your implementation of the Kernel Perceptron algorithm using the simulated data generated by the generat_data function (You do not need to modify the function). Report the accuracy and plot the decision boundary. The decision boundary is a path that separates a region your model classifies as $+1$ from the region it classifies as -1.

      i. Hint: One way to plot decision boundaries (we accept other ways) is using the meshgrid and contourf functions from matplotlib.

      ii. Hint 2: When using meshgrid, make sure the number of points in your grid is not extremely large. Otherwise, it will be very slow to make predictions for each grid position. On the other side, using too few points will make the boundary look jagged.

      iii. Hint 3: To select a good parameter value (σ) for the RBF kernel, look at Figure 14.5 in the textbook (page 409, 4th edition), where $\sigma = s^2$. Experiment with different σ values (not limited to those in the textbook) and choose the one that provides the best accuracy. For simplicity, we do not have a separate validation split, and the model selection is conducted on the test split. Note that you need to specify your choice of σ at the beginning of MyKernelPerceptron.py

   (b) What happens as you increase σ ? Report the results for different σ values and briefly explain.

   (c) Once you have tested that your Kernel Perceptron works, train and evaluate your implementation using two subsets of the optdigits dataset. The first, contains only the digits 4 and 9, in the files **digits49_train.txt** and **digits49_test.txt**. And the second, containing only the digits 7 and 9, in the files **digits79_train.txt** and **digits79_test.txt**. Report the test accuracy on both datasets. Note that you may need to specify different σ for different datasets.

Submission

- Additional Instructions: You must submit the code in Python. Other programming languages are not accepted and would cause credit loss. We provide the template, including the function's corresponding inputs and outputs; you need to fill in the function and generate the expected results.

- Things to submit:
  - `hw4_sol.pdf`: the document including all of your answers to the problem **including the summary or figure of the programming problems. You need to attach all figures and answers which you generated**. The PDF format is accepted. If you want to use a hand-written document, you can scan it and ensure that the scanned copy is clearly readable. If it is difficult to read, credit would be lost.
  - `MyDecisionTree.py`: a text file containing the python function for Problem 2. Use the skeleton file.
  - `MyKernelPerceptron.py`: a text file containing the python function for Problem 3. Use the skeleton file.
  - `visualization.py`: a text file containing the python function for Problem 3. Use the skeleton file.