SML

Assignment 3

Mar 2024

1 Instructions

- You can use inbuilt libraries for Math, plotting, and handling the data (eg. NumPy, Pandas, Matplotlib).
- Usage instructions for other libraries can be found in the question.
- Only (*.py) files should be submitted for code.
- Create a (*.pdf) report explaining your assumptions, approach, results, and any further detail asked in the question.
- You should be able to replicate your results during demo.
- Note you are not allowed to use libraries which can take data, fit the model, predict the labels and give final evaluation metrics.

---

2 Question-1

Use https://storage.googleapis.com/tensorflow/tf-keras-datasets/mnist.npz MNIST dataset for this question and select two digits - 0 and 1. Label them as -1 and 1. In this exercise you will be implementing AdaBoost.M1. Perform following tasks.

- Divide the train set into train and val set. Keep 1000 samples from each class for val. Note val should be used to evaluate the performance of the classifier. Must not be used in obtaining PCA matrix.
- Apply PCA and reduce the dimension to $p = 5$. You can use the train set of the two classes to obtain PCA matrix. For the remaining parts, use the reduced dimension dataset.
- Now learn a decision tree using the train set. You need to grow a decision stump. For each dimension, find the unique values and sort them in ascending order. The splits to be evaluated will be midpoint of two consecutive unique values. Find the best split by minimizing weighted

miss-classification error. Denote this as $h_1(x)$. Note as we are dealing with real numbers, each value may be unique. So just sorting them and taking midpoint of consecutive values may also result in similar tree. [2]

- Compute α_1 and update weights.
- Now build another tree $h_2(x)$ using the train set but with updated weights. Compute α_2 and update weights. Similarly grow 300 such stumps.
- After every iteration find the accuracy on val set and report. You should show a plot of accuracy on val set vs. number of trees. Use the tree that gives highest accuracy and evaluate that tree on test set. Report test accuracy. [2]

Q2. Consider the above as a regression problem. Apply gradient boosting using absolute loss and report the MSE between predicted and actual values of test set.

- Divide the train set into train and val set. Keep 1000 samples from each class for val. Note val should be used to evaluate the performance of the classifier. Must not be used in obtaining PCA matrix.
- Apply PCA and reduce the dimension to $p = 5$. You can use the train set of the two classes to obtain PCA matrix. For the remaining parts, use the reduced dimension dataset.
- Now learn a decision tree using the train set. You need to grow a decision stump. For each dimension, find the unique values and sort them in ascending order. The splits to be evaluated will be midpoint of two consecutive unique values. Find the best split by minimizing SSR. Denote this as $h_1(x)$. [1]
- Compute residue using $y - .01h_1(x)$.
- Now build another tree $h_2(x)$ using the train set but with updated labels. Note, now you have to update labels based on the way we update labels for absolute loss. That is the labels will be obtained as negative gradients. Compute residue using $y - .01h_1(x) - .01h_2(x)$. [1]
- Similarly grow 300 such stumps. Note, the labels are updated every iteration based on negative gradients.
- After every iteration find the MSE on val set and report. You should show a plot of MSE on val set vs. number of trees. Use the tree that gives lowest MSE and evaluate that tree on test set. Report test MSE. [1]