QTM 347 Machine Learning: Homework 3

Ruoxuan Xiong

Due 2024/04/07 11:59pm

Instructions The homework must be submitted in **one pdf file generated from the jupyter notebook**. Everyone needs to make a submission on Canvas. Please write your and your teammate's 7-digit SIS ID, and **do not** write your and your teammate's name in the homework submission. For the file names, please include IDs of all group members. For example, "QTM347 HW3 (xxxxxxx,xxxxxxx,xxxxxxx)". Please specify the random seed at an appropriate place in your code chunk. It may not be the at the start but specify it when you are executing codes that involve randomness. Please make sure everyone in the group submits the same file and everyone in the group will receive the same grade. There are 100 points in total in this homework.

Problem 1 (Classifying MNIST digits using principal component regression) [30 pts]

We will consider classifying the handwritten digits using principal component regression. Recall that principal component regression involves two steps. First, apply PCA to reduce the dimension of the dataset. Second, apply a regression model to the dimension-reduced dataset. To help you get started, we provided a handout notebook with instructions for loading the dataset into a numpy array.

(a) [10pts] Apply PCA to the training dataset with 20 principal components. Print the top 20 eigenvalues that correspond to the principal components. Also, print the explained variance ratios of the principal components.

(b) [10pts] Implement a principal component regression method by first applying PCA, then applying logistic regression to the dimension-reduced numpy matrix. Keep the number of principal components fixed at 20. Report the logistic regression model's training, validation, and test accuracy.

(c) [10pts] Select the number of principal components using the train-validation split in the handout. Report the number of principal components that achieve the highest validation accuracy. Then, report the training and test accuracy using this number of principal components in the principal component regression procedure. Comment on your findings.

Problem 2 [45 pts]

This question is based on the `College` data set, continuing from Homework 2. This data set has statistics for a large number of US Colleges from the 1995 issue of US News and World Report. The goal is to predict the acceptance rate using all variables other than `Accept` and `Apps`.

(a) [5pts] Split the data into a training set and a test set with 80% observations in the training set and 20% observations in the test set.

(b) [5pts] Fit a partial least squares model on the training set, with M chosen by cross-validation. Report the test error obtained, along with the value of M selected by cross-validation.

[Hint: You may find `sklearn.cross_decomposition.PLSRegression` useful.]

(c) [5pts] Fit a regression tree to the training set. Plot the tree and interpret the results. What test MSE do you obtain?

[Hint: You may find `DecisionTreeRegressor()` in the `sklearn.tree` library and `plot_tree()` in `sklearn.tree()` useful.]

(d) [5pts] Use cross-validation in order to determine the optimal depth of tree `max_depth`. Similarly, select the minimum number of samples for a split `min_samples_split` and for a leaf `min_samples_leaf` using cross-validation. Does selecting these parameters in the regression tree improve the test MSE? Plot the tree again and compare it to the plot from Step (c).

(e) [5pts] Use the bagging approach in order to analyze this data. What test MSE do you obtain? Use the `feature_importances_()` attribute in `DecisionTreeRegressor()` to identify which variables are most important.

[Hint: You may find `RandomForestRegressor()` in `sklearn.ensemble` library useful.]

(f) [5pts] Use random forests to analyze this data. What test MSE do you obtain? Use the `importance()` function to determine which variables are most important. Describe the effect of the number of trees m, as well as the number of variables considered at each split, on the error rate obtained.

[Hint: You may find `RandomForestRegressor()` in the `sklearn.ensemble` library useful. You may use the `n_estimators` parameter to change the number of trees. You may use the `max_features` parameter to change the number of features considered at each split.]

(g) [5pts] Fit a boosted regression tree model to the training set. Plot the tree and interpret the results. What test MSE do you obtain?

[Hint: You may find the function `GradientBoostingRegressor()` in the `sklearn.ensemble` library useful.]

(h) [10pts] Determine the optimal set of parameters for `n_estimators`, `max_depth` and `min_samples_split` for the boosted tree model using cross-validation.

[Hint: The range for the number of tree estimators is $[1, 400]$. The range for the maximum tree depth is $[1, 10]$. The range for the minimum samples per split is $[1, 10]$.]

Based on the selected parameters, make a horizontal bar plot for the feature importance scores and discuss your results.

[Hint: Keep the strongest predictive features at the top and the weakest predictive features at the bottom of the plot for better visualization.]

Problem 3 [25pts]

In this problem, you will implement the random forest algorithm from scratch. The data set for this assignment is the SPAMBASE dataset from the UCI repository available at `https://archive.ics.uci.edu/ml/datasets/spambase`. Split the original data into 80% for training and 20% for testing (chosen at random).

(a) [10pts] Begin by creating a bootstrap sample of size $1,000$ and then select a subset of p columns. Vary the value of p and report the p that results in the lowest cross-validation error. Now, train a decision tree classifier on the bootstrap sample by setting the depth to 6.

[Hint: You may want to use `DecisionTreeClassifier()` in `sklearn.tree()` to fit each tree. Refer to class notes and some suggested values to choose the value of p.]

(b) [10pts] Repeat the above step to generate T trees where T is in the range of $\{1, 50, 100, 150, 200, 300, 400\}$ and evaluate on the training set. Combine the predictions from all trees and assign the final class based on a majority vote of the predictions of every tree. In the case of ties, assign a class randomly among the ties. Then, report the training and test error, F1 score, and AUC by varying T in the range $\{1, 50, 100, 150, 200, 300, 400\}$.

(c) [5pts] Use an existing package to train a Random Forest algorithm with 10, 50, and 100 decision trees. Report similar metrics on both the training and testing sets. Report the top 10 features having the most influence on the model.

[Hint: You may find `RandomForestClassifier()` in the `sklearn.ensemble` library useful.]