Machine Learning - Problem Set 4

PPHA 30546 - Professor Clapp
Winter 2023

This assignment must be handed in via Gradescope on Canvas by **11:45pm Central Time on Wednesday, March 1st.** You are welcome (and encouraged!) to form study groups (of no more than 2 students) to work on the problem sets and mini-projects together. But you must write your own code and your own solutions. Please be sure to include the names of those in your group on your submission.

You should submit your answers in one of two ways:

1. As a single PDF containing BOTH a write-up of your solutions that directly integrates any relevant supporting output from your code (e.g., estimates, tables, figures) AND your code appended to the end of your write up. You may type your answers or write them out by hand and scan them (as long as they are legible). Your original code may be a Python (*.py) or Jupyter Notebook (*.ipynb) file converted to PDF format. OR

2. As a single PDF of a Jupyter Notebook (*.ipynb) file with your your solutions and explanations written in Markdown.[1]

Regardless of how you submit your answers, be sure to make it clear what question you are answering by labeling the sections of your write up well and assigning your answers to the appropriate question in Gradescope. Also, be sure that it is immediately obvious what supporting output from your code (e.g., estimates, tables, figures) you are referring to in your answers. In addition, your answers should be direct and concise. Points will be taken off for including extraneous information, even if part of your answer is correct. You may use bullet points if they are beneficial. Finally, for your code, please also be sure to practice the good coding practices you learned in PPHA 30535/6 and comment your code, cite any sources you consult, etc.

You are allowed to consult the textbook authors' websites, Python documentation, and websites like StackOverflow for general coding questions. You are not allowed to consult material from other classes (e.g., old problem sets, exams, answer keys) or websites that post solutions to the textbook questions.

1. Do the following questions from Chapter 6 of the *Introduction to Statistical Learning* textbook:

   (a) Question 9, parts (a), (b), (e)-(g)

   - For (a), please use a 50/50 training/test split. To avoid confusion among partners and facilitate grading, please also set `random_state=37` when you split the data.

[1] Converting a Jupyter Notebook to PDF is not always straightforward (e.g., some methods don't wrap text properly). Please ensure that your PDF is legible! We will deduct points if we cannot read your PDF (even if you have the correct answers in your Notebook).

- For (e) and (f), be sure to standardize the data before performing PCR and PLS using scikit-learn's `StandardScaler` command. Use 10-fold cross-validation (10FCV) on the training set, shuffle the data randomly for splitting, and set `random_state=1`.
- Python does not have a PCR command, so you should use scikit-learn's `PCA` command, then run an OLS regression using the resulting principal components.
- Scikit-learn does have a PLS regression command (`PLSRegression`).

2. Do the following questions from Chapter 8 of the *Introduction to Statistical Learning* textbook:

   (a) Question 4

   (b) This question is a modified version of Question 9. It involves the *OJ* data set which is available on Canvas.[2]

      i. Create a training set containing a random sample of 800 observations, and a test set containing the remaining observations.

      ii. Fit a full, unpruned tree to the training data, with *Purchase* as the response and the other variables as predictors. What is the training error rate?

      iii. Create a plot of the tree.[3] The plot is a mess, isn't it? For the purposes of this question, fit another tree with the `max_depth` parameter set to 3 in order to get an interpretable plot. How many terminal nodes does the tree have? Pick one of the terminal nodes, and interpret the information displayed.

      iv. Predict the response on the test data, and produce a confusion matrix comparing the test labels to the predicted test labels. What is the test error rate?

      v. Use cost complexity pruning to determine the optimal subtree for prediction by tuning the the α hyperparameter.[4] Produce a plot with the values of α (`ccp_alpha`) on the x-axis and the cross-validated classification error rate on the y-axis calculated using 5-fold cross-validation (5FCV).[5] Which α corresponds to the lowest cross-validated classification error rate?

      vi. Now produce a second plot showing the values of α (`ccp_alpha`) on the x-axis and the tree size on the y-axis calculated using the method in the previous question. Note that by the nature of how pruning works, tree size is a function of the α hyperparameter, but there can be multiple values of alpha that produce the same tree size.[6] Briefly explain how the value of α affects tree size and model complexity more general.

      vii. Produce a plot of the optimal pruned subtree obtained using cross-validation. If cross-validation does not lead to selection of a pruned tree, then create a pruned tree with five terminal nodes.

[2] For variable definitions, see https://rdrr.io/cran/ISLR/man/OJ.html.

[3] There are two ways to plot a tree in Python: scikit-learn's `plot_tree()` function and `Graphviz`. The latter is a little difficult to work with, so use the former.

[4] This is the `ccp_alpha` argument in scikit-learn's `DecisionTreeClassifier`. You can use `GridSearchCV` to tune α.

[5] Use the training dataset for 5FCV, shuffle the data randomly for splitting, and set `random_state=13`.

[6] Also, note that tree size is the number of terminal nodes or leaves and you can find this using the `.get_n_leaves()` command after fitting the model.

viii. Compare the training error rates between the pruned and unpruned trees. Which is higher? Briefly explain.

ix. Compare the test error rates between the pruned and unpruned trees. Which is higher? Briefly explain.