STATS 216v
Stanford University, Summer 2017

Problem Set 3

Due: August 4, 2017
Remember the university honor code. All work and answers must be your own.

---

1. Consider three curves, $\hat{g}_1$, $\hat{g}_2$ and $\hat{g}_3$, defined by

$$\hat{g}_1 = \arg\min_g \left(\sum_{i=1}^{n} (y_i - g(x_i))^2 + \lambda \int \left[g"(x) \right]^2 dx \right),$$

$$\hat{g}_2 = \arg\min_g \left(\sum_{i=1}^{n} (y_i - g(x_i))^2 + \lambda \int \left[g^{(3)}(x) \right]^2 dx \right),$$

$$\hat{g}_3 = \arg\min_g \left(\sum_{i=1}^{n} (y_i - g(x_i))^2 + \lambda \int \left[g^{(4)}(x) \right]^2 dx \right),$$

   where $g^{(m)}$ represents the mth derivative of g.

   (a) As $\lambda \to \infty$, which one among $\hat{g}_1$, $\hat{g}_2$ and $\hat{g}_3$ will have the smallest training RSS?

   (b) As $\lambda \to \infty$, which one among $\hat{g}_1$, $\hat{g}_2$ and $\hat{g}_3$ will have the smallest test RSS?

   (c) For $\lambda = 0$, will $\hat{g}_1$, $\hat{g}_2$ or $\hat{g}_3$ have the smaller training and test RSS?

2. Suppose that we carry out backward stepwise, forward stepwise, and best subset all on the same data set. Each approach will yield a sequence of models with $k = 0$ up through $k = p$ predictors.

   (a) Which approach with k predictors will have the smallest *test* residual sum of squares? Explain.

   (b) Which approach with k predictors will have the smallest *training* residual sum of squares? Explain.

   (c) True or False:

      i. The predictors in the k-variable model identified by forward stepwise are a subset of the predictors in the $(k+1)$-variable model identified by backward stepwise selection.

      ii. The predictors in the k-variable model identified by backward stepwise are a subset of the predictors in the $(k+1)$-variable model identified by forward stepwise selection.

      iii. The predictors in the k-variable model identified by best subset are a subset of the predictors in the $(k+1)$-variable model identified by best subset selection.

iv. The predictors in the k-variable model identified by backward stepwise are a subset of the predictors in the $(k+1)$-variable model identified by backward stepwise selection.

v. The predictors in the k-variable model identified by forward stepwise are a subset of the predictors in the $(k+1)$-variable model identified by forward stepwise selection.

Explain each answer.

3. This question uses the combined cycle power plant data set on Lichman, M. (2013). UCI Machine Learning Repository (http://archive.ics.uci.edu/ml). You can download the data set from the course website as "power_plants.csv". Here we aim to use the "Exhaust vacuum" (V) predictor to predict the "Net hourly electrical energy output" (EP) of the 9568 power plants. A description of the data set can be found at http://archive.ics.uci.edu/ml/datasets/Combined+Cycle+Power+Plant.

   (a) Denote x as the Exhaust vacuum (V) and y as the Net hourly electrical energy output (EP). Use the `poly()` function to fit a cubic polynomial regression to predict y using x. Report the regression output, and plot the resulting data and polynomial fits.

   (b) Plot the polynomial fits for a polynomials of degree 1,3,7, and 10, and report the associated residual sum of squares.

   (c) Perform cross-validation to select the optimal degree for the polynomial, and explain your results.

   (d) Use the `bs(x, df = 4)` function to fit a regression spline to predict EP using V. This will result in a spline with 5 of freedom when we include the intercept. Report the output for the fit. How did you choose the knots? Plot the resulting fit.

   (e) Now fit a regression spline for a range of degrees of freedom, and plot a few of the resulting fits and report the resulting RSS. Describe the results obtained.

   (f) Perform cross-validation in order to select the best degrees of freedom for a regression spline. Plot the CV estimate of error versus the degrees of freedom. Describe your results.

4. This problem works with the `body` dataset, which you can download from the homework folder on the class website.. The goal of this problem is to perform and compare Principal Components Regression and Partial Least Squares on the problem of trying to predict someone's weight. While you can use any R tools at your disposal to complete the problem, `library(pls)` and `Lab 3` from chapter 6 of `ISLR` will probably be very helpful, and the problem set was written with these approaches in mind. More information about this dataset can be found at

   `http://www.amstat.org/publications/jse/v11n2/datasets.heinz.html`.

   (a) Read the `body` dataset into R using the `load()` function. This dataset contains:

- `X`: A dataframe containing 21 different types of measurements on the human body.
- `Y`: A dataframe that contains the age, weight (kg), height (cm), and the biological sex of each person in the sample.

Let's say we forgot how the binary variable sex is coded in this dataset. Using a simple visualization, explain how you can tell which sex is which.

(b) Here we run set.seed(2017) in R. Reserve 200 observations from your dataset to act as a test set and use the remaining 307 as a training set. On the training set, use both `pcr` and `plsr` to fit models to predict a person's weight based on the variables in `X`. Use the options `scale = TRUE` and `validation='CV'`. Why does it make sense to scale our variables in this case?

(c) Run `summary()` on each of the objects calculated above, and compare the training % variance explained from the `pcr` output to the `plsr` output. Do you notice any consistent patterns (in comparing the two)? Is that pattern surprising? Explain why or why not.

(d) For each of the models, pick a number of components that you would use to predict future values of weight from `X`. Please include any further analysis you use to decide on the number of components.

(e) Practically speaking, it might be nice if we could guess a person's weight without measuring 21 different quantities. Do either of the methods performed above allow us to do that? If not, pick another method that will, and fit it on the training data.

(f) Compare all 3 methods in terms of performance on the test set. Keep in mind that you should only run one version of each model on the test set. Any necessary selection of parameters should be done only with the training set.

5. This question uses the `ALS` dataset. In Problem Set 1, we saw that plain linear regression did not perform too well on this problem, and in Problem Set 2 we were able to lower the RMSE by using a lasso fit. Still, we were working in a linear regression context. In this exercise, we tackle the problem via non-linear regression trees. *Note:* because of the size of this dataset, some of the commands may take a couple of seconds to run.

(a) Using the `tree` package, fit a decision tree to our training data. Call your object `full.tree.model` and print its summary.

(b) Plot your tree.

(c) Now let us prune this tree. Set `seed(2017)`, and evaluate the CV error of the pruned trees by using the `cv.tree` command. Store the result of `cv.tree` on an object called `tree.cv`.

(d) Plot it by typing `plot(tree.cv$size, tree.cv$dev, type='b')`. Here, 'dev' is short for *deviance*, which can be roughly interpreted as the sum of squared error.

(e) Identify the pruned tree with smallest CV error. Plot it.

(f) Compute the RMSE of the full tree and the pruned tree. Which fares better? How does the Lasso (from Problem Set 2) compare to these methods?

(g) We know that in general we can improve the performance of decision trees via boosting. Set `set.seed(2017)` and fit a boosted tree to the training data via the `gbm` command in the `gbm` package, and call it `gbm.model`. Use the options `distribution="gaussian", n.trees=1700, shrinkage=0.01, cv.folds=5`, and print your model.

(h) Find the number of trees that minimizes the CV error. Plot the CV error vs. the number of trees, and indicate the minimizing number of trees in your plot.

(i) What is the test RMSE of this model?

(j) Finally, let us try using random forests. Set `set.seed(2017)` use the `randomForest` package to fit a random forest to the training data with parameters `mtry=80, importance=TRUE`. Plot the out-of-bag error versus the number of trees.

(k) Compute the test RMSE of this model. How does it compare to boosting and decision trees? Comment on these results.