

FA-582 – Assignment 4

Problem 1

This problem uses the OJ data set (OJ.csv).

- a) Create a training set containing a random sample of 800 observations, and a test set containing the remaining observations.
- b) Fit a tree to the training data, with Purchase as the response and the other variables as predictors. Use the summary() function to produce summary statistics about the tree, and describe the results obtained. What is the training error rate? How many terminal nodes does the tree have?
- c) Type in the name of the tree object in order to get a detailed text output. Pick one of the terminal nodes, and interpret the information displayed.
- d) Create a plot of the tree, and interpret the results.
- e) 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?
- f) Apply the cv.tree() function to the training set in order to determine the optimal tree size.
- g) Produce a plot with tree size on the x-axis and cross-validated classification error rate on the y-axis.

Problem 2

This problem uses the Caravan data set (CARAVAN.csv).

- a) Create a training set consisting of the first 1,000 observations, and a test set consisting of the remaining observations.
- b) Fit a boosting model to the training set with Purchase as the response and the other variables as predictors. Use 1,000 trees, and a shrinkage value of 0.01. Which predictors appear to be the most important?
- c) Use the boosting model to predict the response on the test data. Predict that a person will make a purchase if the estimated probability of purchase is greater than 20 %. Form a confusion matrix. What fraction of the people predicted to make a purchase do in fact make one? How does this compare with the results obtained from applying random forest model, KNN or logistic regression to this data set?

Problem 3

In this problem, you will generate simulated data, and then perform PCA and K-means clustering on the data.

- a) Generate a simulated data set with 20 observations in each of three classes (i.e. 60 observations total), and 50 variables. Hint: There are a number of functions in R that you can use to generate data. One example is the `rnorm()` function; `runif()` is another option. Be sure to add a mean shift to the observations in each class so that there are three distinct classes.
- b) Perform PCA on the 60 observations and plot the first two principal component score vectors. Use a different color to indicate the observations in each of the three classes. If the three classes appear separated in this plot, then continue on to part (c). If not, then return to part (a) and modify the simulation so that there is greater separation between the three classes. Do not continue to part (c) until the three classes show at least some separation in the first two principal component score vectors.
- c) Perform K-means clustering of the observations with $K = 3$. How well do the clusters that you obtained in K-means clustering compare to the true class labels? Hint: You can use the `table()` function in R to compare the true class labels to the class labels obtained by clustering. Be careful how you interpret the results: K-means clustering will arbitrarily number the clusters, so you cannot simply check whether the true class labels and clustering labels are the same.
- d) Perform K-means clustering with $K = 2$. Describe your results.
- e) Now perform K-means clustering with $K = 4$, and describe your results.
- f) Now perform K-means clustering with $K = 3$ on the first two principal component score vectors, rather than on the raw data. That is, perform K-means clustering on the 60×2 matrix of which the first column is the first principal component score vector, and the second column is the second principal component score vector. Comment on the results.
- g) Using the `scale()` function, perform K-means clustering with $K = 3$ on the data after scaling each variable to have standard deviation one. How do these results compare to those obtained in (b)? Explain.