STA 450/2102 — Assignment #1

Due at the beginning of class on February 4. Worth 18% of the final mark.

When many predictor variables are available to help in predicting a response variable, but many of these predictors may actually be irrelevant, it is common for people to try to find a subset of the predictor variables that are adequate to build a regression model. Such a simple model including few variables may be easier to interpret, and may lead to better predictions when the number of cases available to estimate the regression coefficients is small (at least if estimation is done by least squares, rather than by something more sophisticated).

This assignment concerns a simple variable selection method of this sort. Suppose you have p predictor variables and a response variable for each of n cases, and suppose that you think that at most one of the p predictor variables is actually helpful in predicting the response. A simple approach to deciding which of the p predictor variables is the relevant one is to just choose the predictor variable that has the biggest correlation (in absolute value) with the response variable.

This approach has some problems, however. One is that it won't work well if the assumption that at most one predictor is relevant is wrong. Another is that when the regression model based on this single predictor variable is fitted, the usual p-value that is produced for the test of the null hypothesis that there is no linear relationship between the predictor variable and the response variable will not be correct. Since the predictor variable was selected from amongst many on the basis of its high correlation with the response, what would otherwise be seen as a statistically significant relationship might just be due to chance.

In this assignment, you will do the following:

1. Write an S-Plus function that selects the predictor variable with the strongest correlation, fits a regression model with this single predictor variable, and finds the usual p-value for the test of the null hypothesis of no relationship with this predictor variable.

2. Write an S-Plus function to find the distribution of the p-value found by the above function when the null hypothesis is true — ie, when really none of the predictor variables are helpful in predicting the response variable. You should then investigate how far this distribution of p-values is from the desired Uniform(0,1) distribution.

3. Write an S-Plus function that implements this selection method along with a permutation test of the null hypothesis of no relationship. This should fix the problem of the p-values having the wrong distribution.

For the first task, you should write a function called `reg1.naive`, which takes as arguments a $n \times p$ matrix, X, of predictors and a vector of length n of responses and returns a list with an element named `coef`, which is the vector of regression coefficients (with zeros for all the predictors except the one selected), and an element named `pvalue`, which is the p-value for the test of the null hypothesis of no relationship that is found by the standard S-Plus `lm` function when a regression is done on the single predictor with highest correlation. This p-value is printed when you do commands such as

```
> m <- lm(y~x)
> summary(m)
```

For this function, however, you need to return the p-value for later use, not print it. As far as I can tell, the easiest way to do this is as follows:

```
> m <- lm(y~x)
> s <- summary(m)
> pvalue <- 1 - pf(s$fstatistic[1],s$fstatistic[2],s$fstatistic[3])
```

This finds the tail probability for the F statistic for the regression, which is how the regression p-value is computed.

For the second task, you should write an S-Plus function called `test.naive.pvalue`, which simulates datasets under the null hypothesis of no relationship, and finds p-values for each using `reg1.naive`. This function should take as arguments the number of cases in the simulated data sets, the number of predictor variables in the simulated data sets, the correlation between these predictor variables, and the number of simulated datasets to produce. The response variable should be independent of the predictors, as assumed under the null hypothesis. All the variables should marginally have standard normal distributions. You can generate correlated normal random variables by finding the Cholesky decomposition of the covariance matrix, as described in lecture. This test function should return a vector with the p-values found for these simulated datasets. You can then look at histograms or do other tests of how close the distribution of these p-values is to being uniform.

You should look at the distribution of p-values for datasets with 20 cases and 4 predictor variables per case, with the correlations among the predictor variables being 0, 0.9999, and -0.3333. Discuss the results.

For the third task, you should write a function called `reg1.perm` that behaves similarly to `reg1.naive` except that the p-value it returns is found with a permutation test. This is done by randomly shuffling the vector of responses N times and computing the naive p-value (with `reg1.naive`) for each shuffling. These p-values are then compared with the naive p-value found with the actual dataset, to produce a valid p-value. The parameters of `reg1.perm` should be X, y, and N.

Test this function on some randomly generated datasets such as those used for the second task, in order to confirm that it gives valid p-values even when the naive test does not.

You should hand in listings of your programs (which should be formatted for easy reading, with an appropriate amount of documentation), the output of your tests, and your conclusions.