Exercises 5 · Quantifying uncertainty

Due Monday, February 23, 2015

(1) Uncertainty quantification: two views

We have now encountered two broad philosophies for quantifying uncertainty in statistical models. The first is bootstrapping. The second is making probabilistic assumptions (i.e. normality) about the random or unexplained component of the model. In no more than 1 typed page, compare these two philosophies. What assumptions do they require? What are their relative strengths and weaknesses?

(2) Case study: cheese sales and promotional displays

This question considers sales volume, price, and advertisting display activity for packages of Borden sliced cheese. The data, available as "cheese.csv", are taken from Rossi, Allenby, and McCulloch's textbook on *Bayesian Statistics and Marketing*. For each of 88 stores (store) in different US cities, we have repeated observations of the weekly sales volume (vol, in terms of packages sold), unit price (price), and whether the product was advertised with an in-store display during that week (disp = 1 for display). Altogether there are 5,555 observations in the data set.

Address the following questions thoroughly but concisely. Make sure to include the appropriate plots, statistical summaries, and measures of uncertainty to illustrate and support your conclusions. Present your solutions as though you are a consultant seeking to inform and convince a (statistics-savvy) client of your results.

(A) Ignoring price, do the in-store displays appear to have an effect on sales volume? Use an appropriate transformation. In light of your analysis, complete the following two sentences. "I estimate that in-store displays increase/decrease sales by —%. I am 95% confident that this quantity is between —% and —%."

(B) Is there reason to suspect that your result in (A) is confounded by pricing strategies? Show evidence either way. If the answer is yes, propose a model that allows you to adjust for price in assessing the marginal effect of in-store displays on sales volume. Remember back to our milk sales-versus-price data that a typical model for price elasticity of demand is of the form $\hat{y}_i = Kx_i^{\beta}$, where $\hat{y}$ is expected sales, x is price, K is a constant, and β is the elasticity—that is, the marginal effect of price on sales volume. You should recall

how to use linear least squares to fit such a model; now modify it to account for the effect of in-store displays.

As above, in light of your analysis, complete the following two sentences. "I estimate that in-store displays increase/decrease sales by —%, once the effect of price is accounted for. I am 95% confident that this quantity is between —% and —%."

(C) Does price elasticity for Borden cheese appear to be changed by the presence of in-store advertisement? (Hint: remember about interaction terms in models with numerical and categorical predictors.) As above, quote an appropriate confidence interval that addresses this question.

(D) Can you think of a possible economic explanation for your result in Part C?

(3) Data mining and cross validation

This week's final question introduces a very important concept from data mining called "cross validation." Return to the four polynomial regression models you fit to the "utilities.csv" data set, where you modeled daily gas bill versus temperature. You articulated an intuitive preference for one model over the other. Is there a way to formally validate these models? This turns out to be a crucial question in data mining, where the functional form of relationships in large data sets is rarely known in advance.

In statistics, we like models that predict well. But "predicting well" is not the same as "fitting the data well"—if it were, we'd just draw a very squiggly line through all the points, and call that our model. But this model would be horribly over-fit, and would generalize poorly to future data. That makes it a bad model.

If someone gave you a second, independent data set, you could actually check how well each model predicts the y's, given the x's. This is called *out-of-sample predictive validation*. With only one data set, of course you can't do this. (Plus, if you actually possessed that future data set, you'd want to use it for fitting the model.) But you can get some idea of how a model would generalize to future data by the process of *cross-validation*, which mimics out-of-sample predictive validation.

The idea is this. First, randomly split the rows (cases) of your data set into two chunks: a "training" set, and a "test" set. Use the training set alone to fit the model. Then use the fitted model to make predictions for the likely y values corresponding to x values in the test set. You can

then check the actual y values in the test set, to see how well you did at predicting them.

For example, here's how you would apply this idea to fit a quadratic model (once you've read in the data, of course):

```
# Define the daily_gas_bill variable
utilities$daily_gas_bill = utilities$gasbill/utilities$billingDays
# Choose how large the training set will be out of 117 total data points
n_train = 90
# Randomly pick points that will form the training set
training_points = sample(1:n_total, size=n_train, replace=FALSE)
# Split the data into the training and testing sets
training_data = utilities[training_points,]
testing_data = utilities[-training_points,]
# Fit the model to the training set
model2 = lm(daily_gas_bill ~ temp + I(temp^2), data=training_data)
# Predict on the testing set and calculate mean squared error
yhat_model2 = predict(model2, newdata=testing_data)
MSE_model2 = mean( (yhat_model2 - testing_data$daily_gas_bill)^2 )
MSE_model2
```

Try executing this code block a few times and see the different values of MSE (mean-squared error; see the notes below) that you get.

In fact, you should actually fit *all* your candidate models to the training set, and then generate different predictions from each model for the same test set. Just remember not to use the test set points in fitting the model—that would be cheating!

Some notes that will help you in thinking through this problem:

1) There is a lot of randomness in terms of which points get selected to be "training" versus "testing" points. Thus you'll want to use a Monte Carlo simulation to average the results of many different train/test splits.

2) There is also some arbitrariness in the ratio of training set size to the test set size. We've used a rough 80/20 split of the points in the data set. You may play around with this if you wish, but don't have to.

3) In the code block above, we're only fitting the quadratic model. In reality, you should actually fit *all four* of your candidate models to the training set, and then generate predictions from each model for the same test set. This is important to ensure that the four candidate models don't make predictions for different train/test splits (which would introduce an extra, unnecessary source of randomness.)

4) How to measure the quality of a model's out of sample prediction? Since you're fitting each model by least-squares, a natural measure of predictive ability is the sum of squared errors for the out-of-sample points. If $y_1, \ldots, y_m$ are the test-set points, and $\hat{y}_1, \ldots, \hat{y}_m$ are the predictions for these points generated by some model fit to the training data, then the mean of the squared out-of-sample errors is

$$\text{MSE} = (1/m) \sum_{k=1}^{m} (y_k - \hat{y}_k)^2 .$$

   Smaller values of MSE (mean squared error) are better!

5) MSE varies from one split to the next. Ideally you should characterize the between-model and within-model variation in MSE across your Monte Carlo simulation.

The overall question for you to answer is: which of the four models (linear through 4th-order polynomial) model predicts best under cross validation? Make sure to show appropriate visual and/or quantitative evidence from your cross-validation ouput. Also, please turn in your R script in a fixed-width font.