UWEO StatR301 – Spring 2013: Homework 1

Due: Thursday April 18, before Class (grace period 1 additional week)

Assaf Oron, *assaf@uw.edu*

Reading related to this assignment: Lectures 1-2; Lab 1; Rizzo Ch. 6-8 (scanned copy on class website); Hastie et al. Section 7.11; Dobson and Barnett Sections 12.1-12.3.

Instructions:

- Please submit online in the class dropbox. Please submit either ***.pdf is accepted as the main submission (with code pasted verbatim), or *.rmd.** (tip: to save time when using knitr, use `'cache=TRUE'` in the chunk header for any code chunks that run big simulations – this will prevent them from re-running each time you recompile the code).
- **Starred (*) questions and question-parts are not required.** You may submit them if you choose, or do any part of them without submitting.
- **Grading is determined chiefly by effort, not by correctness.** If your submission shows evidence of independent, honest effort commensurate with the amount of homework assigned – you will receive full credit.

1. Randomization ("permutation") Inference: Challenger Dataset.

Recall the Challenger accident o-ring dataset used in Winter. In case you lost it, it is available next to this document. The temperature effect there seemed nonlinear, and was concentrated near the edge of the observed range (tragically, the actual temperature on the morning of the fatal accident was far below that range). We will examine the numerical p-value for the effect.

a. Obtain and record the standard regression p-value for the temperature, using logistic regression with only one term: a linear one for the temperature (i.e., don't put pressure into the model). Recall that since there are multiple observations per record, the formula syntax is

```
cbind(Eroded,Intact) ~ tempF
```

b. Find **a nonparametric numerical p-value, as shown in Lecture 1,** by running 10,000 instances of this model with the temperature values randomly permuted. One way to do it, would be by creating a new variable on-the-fly each time:

```
challenger$permT=sample(challenger$tempF)
```

And then running the regression on the permuted variable. Also recall that the coefficient can be retrieved via, e.g., `coef(glm()).`

Compare the numerical p-value to the standard one, and comment. Final note: if the true coefficient is more extreme than all your permuted-label ones, it is more prudent to express it as p<1/M (with M being the ensemble size), rather than a zero p-value.

Try to use some 'Xapply' function (or 'ddply', 'foreach', etc.) to wrap the simulation, but if you get stuck just use a loop. I ran a stupid loop and it took less than 2 minutes on my machine.

c*. Add a quadratic effect of temperature to the simulation, and compare the standard and numerical p-values. Since the effect is calculated from the same variable, the exact same order-permutation can (and should!) be used for both the linear and quadratic effects in each instance (just to clarify, this doesn't make the simulation mechanics more complicated).

2. Estimator-Performance Simulation: Bayesian vs. Frequentist.

This question is reminiscent of StatR201 HW2, question 3, but uses the coin-toss Beta-Binomial example from Lecture 2 rather than some contrived bus-stop scenario.

We compare the performance of the ordinary frequentist Binomial MLE for p, with a Bayesian one having a Beta prior symmetric around p=0.5, whose weight is equivalent to $\alpha+\beta=6$ observations. Use the posterior mean for the Bayesian estimator.

a. Compare the performance on bias, variance and RMSE (root-mean-square-error), using sample sizes of n=5, 25 and 125, and assuming that the true p is indeed 0.5. Run 1000 instances of each condition. **Also show comparative boxplots or qqplots of the two estimators, separately for each sample size, with a horizonal line marking the true value p=0.5.**

Recall, `rmse=function(x,ref) sqrt(mean((x-ref)^2))`

b. Repeat the drill, but now the coin is really biased with p=0.7.

Comment on the results. Important hint: you do not have to run any sort of Bayesian simulation, or even invoke the Beta distribution. Recall that in class, we found a simple closed-form algebraic formula for the posterior mean in this case.

c*. Add to the comparison, the Bayesian estimators of posterior median and mode. For the former, you will probably need to invoke 'dbeta'. For the latter, there is a closed-form algebraic formula (look up Wikipedia if you can't calculate it). Do they do better or worse than the posterior mean in a? In b?