CMSC 478 — Spring 2017 — C. S. Marron
Lab 2: Logistic Regression

Data Description

For this lab, we've simulated 1000 observation of data from three sensors (s_1, s_2, s_3). The response variable, *threat* indicates whether the observation corresponds to a "threat," with $threat = 0$ indicating the observation is a threat, and $threat = 1$ indicating that it is not. You can think of the sensors as being biological or chemical sensors used to screen packages, or any other sort of sensor and threat scenario that you like. In addition, the variable *time* is a timestamp for the observation (really just an index from 1 to 1000). We have simulated a second data set of 1000 observations using the same method as the first; the second set is intended to be used for testing. The two data sets are in the files

- threat_data.csv
- threat_data_test.csv

Exercise 1: Load both data sets into R. Create a logistic regression model for *threat* using *time*, s_1, s_2, and s_3 as predictor variables. Select a threshold to assign each observation to a class based on the predicted probability of being a non-threat and determine the following misclassification rates for the training data:

- Overall percentage of misclassified observations
- Percentage of non-threat observations that are misclassified (false positives)
- Percentage of threat observations that are misclassified (false negatives)

Try several threshold values and choose the one that gives the best misclassification rates on the training data. Using this threshold, apply your model to the test data and determine the same three misclassification rates.

Some useful R functions:

- `attach(threat.data)` so you can access variables directly.
- `glm.fit = glm(..., family=binomial)` to fit the logistic regression model.
- `predict(glm.fit, type="response")` to get the predicted probability of being non-threat for each of the training observations.
- `predict(glm.fit, newdata=threat.data.test, type="response")` to get the predicted probability of being a non-threat for each of the test observations using the model fit on the training data.

- `predict(glm.fit, newdata=threat.data.test, type="response") > 0.8` to produce an assignment to classes with threshold 0.8. In the resulting vector, True indicates that the observation is assigned to the non-threat class, and False that it is assigned to the threat class.
- `threat.data[threat==0, ]` to extract the rows with $threat = 0$. Similarly, you can extract rows with $threat = 1$.

Exercise 2: Create a logistic regression model for $threat$ using s_1, s_2, and s_3 as predictor variables (do *not* include $time$). Try several threshold values; can you find any value of the threshold that gives reasonable misclassification rates on the training data?

Exercise 3: Create several *local* logistic regression model for $threat$ using s_1, s_2, and s_3 as predictor variables; that is, select contiguous subsets of rows and fit models to each subset; can you improve the misclassification rates locally?

For example, create a model using only `threat.data[1:100,]`, the first 100 observations ($time = 1, 2, \ldots, 100$); choose a threshold to assign observations to classes. Does this give better predictions for the first 100 observations?

Exercise 4: Examine the estimated coefficients from a number of local models. Do you see any patterns? Can you draw any conclusions about the generation of the data?

[Top]