STATS216v Introduction to Statistical Learning
Stanford University, Summer 2017

**Problem Set 2**

**Due: Friday, July 21**

Remember the university honor code. All work and answers must be your own.

---

1. Suppose you have collected a dataset of information about 40 patients, ages 55 to 60 years old. You have variables $X_1$ = weight, $X_2$ = years they have smoked, and $Y$ = whether or not they have had a heart attack. You fit a logistic regression model, which yields the following coefficients: $\hat{\beta}_0 = -3.8, \hat{\beta}_1 = 0.007, \hat{\beta}_2 = 0.03$.

   (a) Estimate the probability that a new patient who weighs 143 lbs and has smoked for 17 years will have a heart attack.

   (b) Suppose a patient does not smoke: $X_2 = 0$. At what weight would we predict the chance of having a heart attack is 10%?

2. Recall that the Lasso estimate for $\beta$ is the minimizer of the following expression:

$$\sum_{i=1}^{n} \left( y_i - \beta_0 - \sum_{j=1}^{p} \beta_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} |\beta_j|$$

   where $\lambda$ is a tuning parameter.

   (a) We call $\lambda \sum_{j=1}^{p} |\beta_j|$ the Lasso penalty term. Why do we call it a penalty?

   (b) In terms of other techniques from Stats 216v, what value of $\beta$ gives the minimum of the above expression when $\lambda = 0$?

   (c) What value of $\beta$ gives the minimum when $\lambda = \infty$?

   (d) Briefly explain that what will happen to each of the following quantities as we increase the parameter $\lambda$ from 0 to $\infty$:

      i. The training RSS.
      ii. The test RSS.
      iii. The variance of the $\beta_i$'s.
      iv. The bias.

3. The `Default` data set in the `ISLR` library contains data about 10000 simulated credit card customers. Our goal is to model the probability that a customer will default on their debt. We will use a logistic regression model the using `default` as the response and `income` and `balance` as features. For this problem, we are interested in the standard errors of our estimates of the logistic regression coefficients. We will compute standard errors in two ways: (1) using the bootstrap and (2) using the standard formula for computing standard errors, which is implemented in the `glm()` function.

(a) In two sentences or less, explain what the "standard error" for a coefficient in the model means.

(b) Load the data into your workspace using the following commands:

```
> library("ISLR")
> data("Default")
```

Next, using the `summary()` and `glm()` functions, determine the estimated standard errors for the coefficients associated with `income` and `balance` in the multiple logistic regression model.

(c) Write a function `boot_fn()` that takes as input the `Default` data set as well as an index of the observations, and that outputs the coefficient estimates for `income` and `balance` in the multiple logistic regression model.

(d) Use the `boot()` function together with your `boot_fn()` to estimate the standard errors of the logistic regression coefficients for `income` and `balance` in the multiple logistic regression model. Do 100 bootstrap replications, using `set.seed(2017)` to set the seed.

(e) Comment on the estimated standard errors from the two approaches. Using the standard errors from `glm()`, which of the two predictors are statistically significant?

4. This question should be answered using the `Weekly` dataset, which is part of the `ISLR` package. This data is similar in nature to the `SMarket` data used in section 4.6 of out textbook, except that it contains 1,089 weekly returns for 21 years, from the beginning of 1990 to the end of 2010.

(a) Use the full dataset to perform a logistic regression with `Direction` as the response and the five lag variables plus `Volume` as predictors. Call your model `glm.fit`. Use the summary function to print the results. Do any of the predictors appear to be statistically significant? If so, which ones?

(b) Use the following code to produce the confusion matrix for this problem.

```
> glm.probs = predict(glm.fit, type = "response")
> glm.pred = rep("Down", length(glm.probs))
> glm.pred[glm.probs > 0.5] = "Up"
> table(glm.pred, Weekly$Direction)
```

Explain what the confusion matrix is telling you about the types of mistakes made by logistic regression.

(c) Now fit the logistic regression model using a training data period from 1990 to 2007, with `Lag1`, `Lag2`, and `Lag3` as the only predictors. Compute the confusion matrix and the overall fraction of correct predictions for the held out data (that is, the data from 2008 and 2010).

(d) Repeat (c) using LDA. Use `library(MASS)` to work with the `lda()` command.

(e) Repeat (c) using KNN with $K = 1$. Invoke `library(class)` to work with the `knn()` command.

(f) Which of the models from parts (c), (d), and (e) appears to provide the best results on this data?

(g) What is one scenario in which you might expect an LDA model to outperform a logistic regression model?

(h) What is one scenario in which you might expect a KNN model to outperform a logistic regression model?

5. This question uses the `ALS` dataset from Homework 1. We saw that applying basic linear regression did not yield very good results, mainly because only a few of the many input predictors are actually strongly associated with the output. One alternative to remedy this problem is to use the Lasso, since it automatically does variable selection for us.

   First, load the `als.RData` file.

   (a) Fit a regression model to the training data using the Lasso. Select the regularization parameter via cross-validation. To do so, you'll need to install (and then load) the package `glmnet`. Use the function `cv.glmnet()` within this package to fit the model with cross-validation. Before you do so, use `set.seed(2017)` right before running `cv.glmnet()` to ensure you get the same results as in the solutions. Store the result of `cv.glmnet()` in a variable called `lasso.cv`.

   (b) Produce a plot via `plot(lasso.cv)` to visualize the cross-validated error for different values of your parameter.

   (c) Let's use the 1-standard-error rule to pick the tuning parameter $\lambda$. Set

   ```
   > my.lambda = lasso.cv$lambda.1se
   ```

   Print the value of the `my.lambda`.

   (d) Display the predictors with non-zero coefficients via

   ```
   > nonzero = predict(lasso.cv, s = 'lambda.1se', type = 'nonzero')
   > colnames(train.X)[nonzero]
   ```

   Out of the 323 original predictors, how many have non-zero coefficients?

   (e) Compute the test RMSE for the model fit in part (b), using the regularization parameter chosen in part (c).

   Note that it performs significantly better than the linear regression from Homework 1 (which had an RMSE of 0.7527). In the ALS Prediction Prize4Life Challenge mentioned in Homework 1, the test error from the Lasso model alone already yielded a very competitive score for the challenge!

   (f) Repeat parts (a,d,e) using ridge regression instead of the Lasso. Set again the seed to the value `2017` before calling `cv.glmnet()`. Store the result of `cv.glmnet()` in a variable called `ridge.cv`. Comment on your findings.