Homework #6 (BST 263, Spring 2019)

1. Shrinkage. In this exercise, you will explore the concept of shrinkage through an example called Stein's paradox. Suppose $Y_j \sim \mathcal{N}(\mu_j, \sigma^2)$ for $j = 1, \dots, p$. You observe $Y_1, \dots, Y_p$ and need to estimate $\mu_1, \dots, \mu_p$. For example, Y_j and μ_j could be measured blood glucose and mean blood glucose, respectively, for subject j.

   (a) The most obvious estimator of μ_j is $\hat{\mu}_j(1) = Y_j$. In fact, it's hard to imagine how one could improve upon this obvious estimate.

      i. What is the bias of $\hat{\mu}_j(1)$? (The bias is defined as $\mathrm{E}(\hat{\mu}_j(1)) - \mu_j$.)

      ii. Based on your answer to part 1(a)i and the bias-variance tradeoff, do you think it might be possible to improve upon $\hat{\mu}_j(1)$? Why?

   (b) A *shrinkage* estimator of μ_j is $\hat{\mu}_j(c) = cY_j$, where $c \in [0, 1]$ is a user-specified setting controlling the amount of shrinkage. Write R code to do the following:

      i. Set $p = 1000$. Randomly generate $\mu_j \sim \mathrm{Uniform}(0, 10)$ for $j = 1, \dots, p$.

      ii. Randomly generate $Y_j \sim \mathcal{N}(\mu_j, \sigma^2)$ for $j = 1, \dots, p$, where $\sigma = 4$.

      iii. For each $c \in \{0.0, 0.01, 0.02, \dots, 0.99, 1.0\}$:
      Compute the squared error, $\mathrm{SE}(c) = \sum_{j=1}^{p} (\hat{\mu}_j(c) - \mu_j)^2$.

      iv. Plot $\mathrm{SE}(c)$ versus c. For your particular data set, what value of c minimizes $\mathrm{SE}(c)$? So, does shrinkage help?

   (c) Mathematical verification:

      i. What is the bias of $\hat{\mu}_j(c)$, as a function of c, μ_j, and σ? Let's call it $\mathrm{bias}_j(c)$.

      ii. What is $\mathrm{Var}\big(\hat{\mu}_j(c)\big)$, as a function of c, μ_j, and σ?

      iii. Plot the following three curves on the same plot versus c:

         A. $\sum_{j=1}^{p} \mathrm{bias}_j(c)^2$

         B. $\sum_{j=1}^{p} \mathrm{Var}\big(\hat{\mu}_j(c)\big)$

         C. $\sum_{j=1}^{p} \big(\mathrm{bias}_j(c)^2 + \mathrm{Var}\big(\hat{\mu}_j(c)\big)\big)$

      Discuss what you see. Compare to your plot of $\mathrm{SE}(c)$.

2. Ridge. The ridge regression estimate $\hat{\beta}^{\text{ridge}} \in \mathbb{R}^p$ is the minimizer of

$$F(\beta) = \sum_{i=1}^{n} (y_i - x_i^{\mathsf{T}} \beta)^2 + \lambda \sum_{j=1}^{p} \beta_j^2.$$

   (a) Show that $\hat{\beta}^{\text{ridge}} = (A^{\mathsf{T}} A + \lambda I)^{-1} A^{\mathsf{T}} y$ where $A = \begin{bmatrix} x_1 & \cdots & x_n \end{bmatrix}^{\mathsf{T}} \in \mathbb{R}^{n \times p}$ and $y = (y_1, \dots, y_n)^{\mathsf{T}} \in \mathbb{R}^n$. Hint: The derivation is nearly identical to problem 1 from Homework 3.

   (b) Suppose $n = p$ and $A = I = I_{p \times p}$ (the $p \times p$ identity matrix).

      i. Show that $\hat{\beta}^{\text{ridge}} = cy$ for some $c \in \mathbb{R}$ in this case. Give the mathematical expression for c as a function of λ.

ii. Make the connection with the shrinkage example in problem 1 above, in terms of the assumed probabilistic model and the form of the estimator.

(c) Collinear predictors. Look over the following R code and add a comment to each line to explain. Try running the code with (i) `lambda=1`, (ii) `lambda=100`, and (iii) `lambda=10000`. Report what you get and discuss. What happens when you try to compute `beta_leastsquares`? Provide a mathematical explanation for why this issue happens with least-squares but not with ridge regression.

```
n = 100
x = runif(n)
y = 3*x + 0.25*rnorm(n)
A = cbind(rep(1,n), x, 2*x)
lambda = 1

beta_ridge = solve(t(A) %*% A + lambda*diag(3),  t(A) %*% y)
y_hat = A %*% beta_ridge
plot(x, y, col=4, pch=19)
points(x, y_hat, col=2, pch=19)

beta_leastsquares = solve(t(A) %*% A,  t(A) %*% y)
```

3. Lasso. In this exercise, you will gain intuition for how the lasso method works and why it yields sparsity. Run the following R code to generate data and define the lasso objective function, `F_lasso`. The lasso estimate is the minimizer of `F_lasso`.

```
set.seed(1)
n = 20  # number of samples
x1 = rnorm(n)  # predictor 1 values
x2 = rnorm(n)  # predictor 2 values
y = 2*x1 + 1*x2 + 0.25*rnorm(n)  # outcome values
F_lasso = function(b1,b2) {  # lasso objective function
    0.5*sum((y - b1*x1 - b2*x2)^2)/n + lambda*(abs(b1) + abs(b2))
}
```

(a) Run the following R code to view `F_lasso` for a range of `lambda` values.

```
# Perspective plot of F_lasso for a range of lambda values
betas = seq(-2,3,0.1)  # range of coefficient values for plots
for (lambda in seq(0,2,0.1)) {
    F_lasso_grid = outer(betas,betas,Vectorize(F_lasso))
    persp(betas,betas,F_lasso_grid,theta=120,phi=30)
    Sys.sleep(0.5)
}
```

Describe how the shape of `F_lasso` changes as `lambda` increases. Based on what you see, write a paragraph explaining in your own words why the lasso estimate sometimes contains exact zeros (i.e., why $\hat{\beta}_j$ may be exactly zero). (NOTE: Your explanation should be based on the shape of `F_lasso` — not the usual explanation of lasso's sparsity based on the constrained dual form.) The following code may also help you visualize what is happening.

```
# Contour plot of F_lasso for a range of lambda values
library("glmnet")
for (lambda in seq(0,2,0.02)) {
    F_lasso_grid = outer(betas,betas,Vectorize(F_lasso))
    contour(betas,betas,F_lasso_grid,nlevels=30)
    lasso_fit = glmnet(cbind(x1,x2),y,intercept=F,standardize=F,lambda=lambda)
    points(lasso_fit$beta[1],lasso_fit$beta[2],pch=19,cex=2,col=4)
    grid()
    Sys.sleep(0.1)
}
```

(b) Would you increase or decrease `lambda` if you wanted a sparser lasso estimate (i.e., more zeros)? Explain.

(c) For which value of `lambda` is the least-squares estimate equal to the minimizer of `F_lasso`? Based on the plots in part 3a, explain in your own words why the least-squares estimate doesn't usually contain exact zeros.