MLDS 401: Homework 5
Due: Nov 6, 3:00pm
Professor Malthouse

1. Use the following data:

```
dat = data.frame(
  x1=c(2.23,2.57,2.87,3.1,3.39,2.83,3.02,2.14,3.04,3.26,3.39,2.35,
    2.76,3.9,3.15),
  x2=c(9.66,8.94,4.4,6.64,4.91,8.52,8.04,9.05,7.71,5.11,5.05,8.51,
    6.59,4.9,6.96),
  y=c(12.37,12.66,12,11.93,11.06,13.03,13.13,11.44,12.86,10.84,
    11.2,11.56,10.83,12.63,12.46))
```

   (a) Generate a scatterplot matrix and comment.

   (b) Regress y on x_1. Is the overall model significant? Examine the residuals and comment.

   (c) Regress y on x_2. Is the overall model significant? Examine the residuals and comment.

   (d) Regress y on both x_1 and x_2. Is the overall model significant? Examine the residuals and comment.

   (e) Discuss the implications of this example on forward and backward selection. Assume that you will use a significance level for entry into the model of 0.05.

   (f) For you to think about but not turn in: how could you generate more data sets like this?

2. ACT problem 5.2 (Ridge and lasso regression)

3. This is a variation of JWHT problem 10 on page 263. You will learn how to simulate data where you know the true parameters as a way of evaluating different methods. You will also learn some tricks for generating random data. Assume the following model:

$$y_i = \beta_0 + \sum_{j=1}^{15} x_{ij}\beta_j + e_i,$$

   where $\beta_0 = 3$, $\beta_1 = 1$, $\beta_2 = -1$, $\beta_3 = 1.5$, $\beta_4 = 0.5$, $\beta_5 = -0.5$, and $\beta_j = 0$ for $j > 5$. Notice that $x_6, \ldots, x_{15}$ have no effect on y. We will generate a training set from this model of size 100, a very large test set of size 10,000, estimate various models using only the training data, and compare their accuracy on the test set. Note that you know the "truth," i.e., $\beta = (3, 1, -1, 1.5, 0.5, -0.5, 0, \ldots, 0)^\mathsf{T}$).

(a) We would like for the column vectors of x to be correlated (since if they are uncorrelated the problem is fairly trivial). Let $\mathbf{x} = (x_1, \ldots, x_p)^\mathsf{T}$ (in our case $p = 15$) be a multivariate normal random vector with mean $E(\mathbf{x}) = 0$ and covariance matrix Σ. There are no functions in R (that I know of) to generate $\mathbf{x}$ directly, but it is easy to generate uncorrelated random variables and multiply them by a certain matrix. Let $\mathbf{z} = (z_1, \ldots, z_p)^\mathsf{T}$ be a vector of uncorrelated standard normal random variables, i.e., $E(z_j) = 0$, $V(\mathbf{z}) = \mathbf{I}$, the identity matrix (so that the correlation between any two columns is 0 and the variance of each column is 1). We want to find $p \times p$ matrix $\mathbf{A}$ so that if we let $\mathbf{x} = \mathbf{A}^\mathsf{T}\mathbf{z}$ then

$$V(\mathbf{x}) = V(\mathbf{A}^\mathsf{T}\mathbf{z}) = \mathbf{A}^\mathsf{T}V(\mathbf{z})\mathbf{A} = \mathbf{A}^\mathsf{T}\mathbf{A} = \Sigma.$$

We can find this with a *Cholesky* decomposition. For this part, suppose $p = 4$ and we want

$$\Sigma = \begin{pmatrix} 1 & 0.9 & 0.9 & 0.9 \\ 0.9 & 1 & 0.9 & 0.9 \\ 0.9 & 0.9 & 1 & 0.9 \\ 0.9 & 0.9 & 0.9 & 1 \end{pmatrix}$$

Find the Cholesky decomposition (turn this in) and confirm that $\mathbf{A}^\mathsf{T}\mathbf{A} = \Sigma$. Hint: the `chol` function gives the decomposition, `t` does transposes, and `%*%` does matrix multiplication:

```
sigma = matrix(0.9, nrow=4, ncol=4) + .1*diag(4)
A = chol(sigma)
t(A) %*% A
```

(b) Generate 1000 random vectors with the matrix Σ from the previous part. Submit the variance matrix of $\mathbf{x}$ and answer whether it approximately equals Σ. Hint:

```
Z = matrix(rnorm(4000), nrow=1000)
X = Z %*% A
```

(c) Now generate a $10,100 \times 15$ matrix of x variables so that each x_j has variance 1 and $\text{cov}(x_j, x_k) = 0.9$ for $j \neq k$. Generate y values so that $\sigma_e^2 = V(e_i) = 9$. Hint: regenerate the $\mathbf{X}$ matrix using the commands given above so that it is 10100×15 and then compute y as follows (you have to modify the code above to generate $\mathbf{X}$):

```
set.seed(12345)
# generate a new Z, A and X
beta = c(1,-1,1.5,0.5,-0.5,rep(0,10))
e = rnorm(10100)*3
y = 3 + X %*% beta + e
```

(d) Estimate the true model (i.e., include only $x_1, \ldots, x_5$) using OLS. Submit your estimate of σ_e^2, the slopes and R^2. Do the 7 estimates roughly equal the true parameter values (e.g., within two standard errors)? Do the slopes have the correct signs? Are they significant? Do 95% confidence intervals cover the true values? Note: this is the case where you have a strong theory telling you which variables "cause" y. This is the ideal case, but you often do not have such a theory. Hint: make data frames first:

```
dat = data.frame(X)
dat$y <- y
train <- c(rep(T,100), rep(F, 10000))
```

(e) Apply the estimated model in the previous part to the `test` data set (with 10,000 observations) and report the value of MSE (Note that you generated data so that $\sigma_e^2 = 9$). Hint:

```
mean((test$y-predict(fit, test))^2)
```

(f) Now estimate an OLS model with all 15 predictors. Submit your parameter estimates. Are the coefficients approximately equal to their true values (e.g., within two standard errors)? Are $x_1, \ldots, x_5$ significant? Are their signs right?

(g) Apply the estimated model in the previous part to the `test` data set (with 10,000 observations) and report the value of MSE.

(h) Now estimate a stepwise regression model on all 15 predictors. Report the final model. Did the "right" variables $(x_1, \ldots, x_5)$ come into the model?

(i) Apply the estimated model in the previous part to the `test` data set (with 10,000 observations) and report the value of MSE. How does this value compare with the other two models?

(j) Now estimate a ridge regression models on all 15 predictors and use `cv.glmnet` to pick λ. Generate and submit a ridge trace.

(k) Apply the estimated ridge model in the previous part to the `test` data set and report the value of MSE.

(l) Now estimate a lasso regression models on all 15 predictors and use `cv.glmnet` to pick λ. Generate and submit a ridge trace.

(m) Apply the estimated ridge model in the previous part to the `test` data set and report the value of MSE.

(n) The file `hw5.R` has some R code that I have written to run all of these models, apply them to the test set, and print their respective values of MSE. You can read it in with the following command

```
> source("hw5.R")
```

This reads in a function called `hw5`, which has three arguments: `beta` = true slope vector, `rho` = correlation between the x vectors and `sigmae` = standard deviation of the errors. For example, if you type `hw5()` you will run the function with the default values, `rho`=0.9 and `sigmae`=3 (so that the error variance is 9). You can enter different values with, e.g., `hw5(rho=0.5, sigmae=2)`. Call this function 9 times under the following conditions and note which model(s) perform best in each case. Briefly summarize your findings, i.e., under what circumstances do you recommend ridge regression? Stepwise? No selection or shrinkage?

Condition	`rho`	`sigmae`
High noise, High multicollinearity	0.9	5
Moderate noise, High multicollinearity	0.9	3
Low noise, High multicollinearity	0.9	1
High noise, Moderate multicollinearity	0.5	5
Moderate noise, Moderate multicollinearity	0.5	3
Low noise, Moderate multicollinearity	0.5	1
High noise, Low multicollinearity	0.1	5
Moderate noise, Low multicollinearity	0.1	3
Low noise, Low multicollinearity	0.1	1

(o) For you to think about, but not turn in: how do your conclusions change if you increase the size of your training sample? Values of `beta`?

4. This problem gives you a taste of feature engineering. You have data from a German book company. The company would like to have a machine learning model to predict how much each customer will spend if sent an offer based on previous purchase history. You will build a model based on a previous offer sent on 2014/11/25.

   (a) The file `customer2.csv` has three variables:

   - `id`: customer id, primary key.
   - `train`: indicates if the customer is in the training (1) or test (0) test.
   - `target`: the amount spent in response to the offer sent on 2014/11/25.

   Read the data into software of your choice, compute the natural log of target+1 = `logtarg` and print out basic descriptive statistics.

   (b) The file `orders.csv` is the transaction file, with one record for each unique item purchased. Here is information about the variables:

   - `id`: customer id, matches the customer table.
   - `orddate`: date of order.

- `ordnum`: order number, which uniquely identifies an order
- `category`: gives metadata on the category of an item, e.g., history, cooking, how-to, etc. We will not use this variable for now.
- `qty`: number of the item purchased.
- `price`: unit price. Note that `price*qty` gives the total amount spent on the item.

Read in the file and create a new variable, time (years) since the transaction as `2014/11/25 - orddate`. Compute basic descriptives. Hint: in R/dplyr use

```
ord=read.csv("orders.csv") %>% mutate(t=as.numeric(as.Date("2014/11/25")
  - as.Date(orddate, "%d%b%Y"))/365.25)
```

(c) Roll up/aggregate the transaction file so that you have one record per customer and the following variables. I will call this the "RFM" table. Submit basic summary statistics.

- `id`: customer id.
- `tof`: time on file—years since the first order, hint: maximum value of `t`.
- `r`: recency—years since the most recent order.
- `f`: frequency of orders. Hint: in dplyr see the `n_distinct` function.
- `m`: monetary—total amount spent in the past.

(d) Join the customer and RFM tables. Regress `logtarg` on `log(tof)`, `log(r)`, `log(f)`, and `log(m+1)` using only the training data. Show a summary of the fitted model.

(e) Apply the model from the previous part to the test set and compute the mean squared error on the test set.

5. Submit your final bike model.

(a) Give a rational for why you included each variable, i.e., why would the variable cause bike demand? Document what your model, e.g., what variables did you combine to form composites, etc. Give a correlation matrix and VIFs for the variables in the final model.

(b) Apply ridge regression to your final model (generate a ridge trace) to evaluate how robust your conclusions are. For example, if some of your variables flip signs in the ridge trace, then your conclusions about the variable are not stable. Also show a lasso trace.