STAT 154: Homework 5

Release date: **Friday, March 22**

Due by: **11 PM, Friday, April 5**

The honor code

(a) Please state the names of people who you worked with for this homework. You can also provide your comments about the homework here.

(b) Please type/write the following sentences yourself and sign at the end. We want to make it *extra* clear that nobody cheats even unintentionally.

I hereby state that all of my solutions were entirely in my words and were written by me. I have not looked at another student's solutions and I have fairly credited all external sources in this write up.

Submission instructions

- It is a good idea to revisit your notes, slides and reading; and synthesize their main points BEFORE doing the homework.
- A .Rnw file corresponding to the homework is also uploaded for you. You may use that to write-up your solutions. Alternately, you can typeset your solutions in latex or submit neatly handwritten/scanned solutions.
- **For the parts that ask you to implement/run some R code, your answer should look something like this (code followed by result):**

```
myfun<- function(){
show('this is a dummy function')
}
myfun()

## [1] "this is a dummy function"
```

  Note that this is automatically generated if you use the R sweave environment.

- You need to submit the following:
  1. A pdf of your write-up to "HW5 write-up" that includes some of the code snippets and plots as asked in different parts.
  2. A Rmd or Rnw file, that has all your entire code, to "HW5 code".
- Ensure a proper submission to gradescope, otherwise it will not be graded. **This time we will not entertain any regrade requests for improper submission.**

Homework Overview

This homework revisits LASSO, kernel ridge regression and other regression methods. It also contains the regression analysis problem from HW4 Problem 6. Note that two more questions have been added in addition to the original HW4 Problem 6.

1 True or False (10 pts)

Examine whether the following statements are true or false and *provide one line justification.*

(a) (1 pt) ℓ_1 regularization can be used as a way to prevent bias.

(b) (1 pt) The LASSO estimate $\hat{\theta}_{\text{LASSO}}$ will have all its coordinates nonnegative.

(c) (1 pt)For regularization parameter $\lambda > 0$, unlike ridge regression, the LASSO solution may not be unique.

(d) (1 pt) Unlike in ridge regression, centering and scaling the data before running LASSO is not recommended.

(e) (1 pt) Given data matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ and response $\mathbf{y} \in \mathbb{R}^n$, with $n > d$. In kernel ridge regression computation, one has to invert a larger matrix than in original ridge regression.

(f) (1 pt) The Gram matrix in kernel ridge regression (see lecture note n8.pdf on Piazza) is a positive semi-definite matrix.

(g) (1 pt) For a valid kernel function K, the corresponding feature mapping ϕ can map a finite dimensional vector into an infinite dimensional vector.

(h) (2 pts) In kernel ridge regression with $d = 2$, the feature mapping $\phi(x) = \begin{bmatrix} x_1 \\ x_2 \\ 1 \end{bmatrix}$ corresponds to the following kernel function

$$k(\mathbf{x}, \mathbf{z}) = \left(\mathbf{x}^\top \mathbf{z}\right)^2 + 1.$$

(i) (1 pt) The R package **glmnet** will run faster if one inputs an array of lambdas as argument to **glmnet** rather than running a lambda at a time for the same array of lambdas with a for loop.

2 Coordinate descent for LASSO (19 pts)

For data design matrix $\mathbf{X} = \begin{bmatrix} x_1^\top \\ \vdots \\ x_n^\top \end{bmatrix} \in \mathbb{R}^{n \times d}$ and response $\mathbf{y} = \begin{bmatrix} y_1^\top \\ \vdots \\ y_n^\top \end{bmatrix} \in \mathbb{R}^n$, LASSO optimization problem can be formulated as follows,

$$\min_{\theta \in \mathbb{R}^d} f(\theta) := \|\mathbf{X}\theta - \mathbf{y}\|_2^2 + \lambda \|\theta\|_1 . \tag{1}$$

Observe that the objective in (1) can be equivalent written as follows without using the matrix notation.

$$\min_{\theta \in \mathbb{R}^d} \sum_{i=1}^{n} \left(x_i^\top \theta - y_i\right)^2 + \lambda \|\theta\|_1 .$$

Note that to align with our lecture notes and also for historical reasons, we are using the total square on the first term of equation (1), rather than the average square loss $\frac{1}{n}\|\mathbf{X}\theta - \mathbf{y}\|_2^2$ as follows:

$$\min_{\theta \in \mathbb{R}^d} \frac{1}{n} \|\mathbf{X}\theta - \mathbf{y}\|_2^2 + \omega \|\theta\|_1 . \tag{2}$$

1. (2 pts) **Show that** there exists a choice of ω (depends on λ) such that the LASSO formulation with the average square loss in equation (2) gives the same output as the LASSO formulation with the total square loss.

Although the problem is convex, since the ℓ_1-regularization term in the objective function is non-differentiable, it's not immediately clear how gradient descent can be used to solve this optimization problem directly. There is an alternative method, sub-gradient descent which is usually slow. In practice, for Lasso, another algorithm namely, coordinate descent is used and is well known to converge fast in practice.

In this algorithm, we optimize over one component of the unknown parameter vector, fixing all other components. It turns out that for the LASSO optimization problem, we can find a closed form solution for optimization over a single component fixing all other components (we will discuss this in lab). Using this fact, we obtain the following coordinate descent algorithm 1 for LASSO. The soft thresholding function is defined as

Algorithm 1 Coordinate descent for LASSO (or shooting algorithm)

Initialize $\theta = \theta_0$.
$r \leftarrow a \bmod b$
while Not converged **do**
  for $j = 1, \ldots, d$ **do**
    $a_j = 2\sum_{i=1}^{n} x_{ij}^2$;
    $c_j = 2\sum_{i=1}^{n} x_{ij}\left(y_i - \sum_{k=1,k\neq j}^{d} \theta_k x_{ik}\right)$;
    $\theta_j = \mathbf{soft}(\frac{c_j}{a_j}, \frac{\lambda}{a_j})$;
return θ

$$\mathbf{soft}(a, \delta) = \text{sign}(a)\,(|a| - \delta)_+$$

2. (2 pts) **Manually draw** the soft-thresholding function $a \mapsto \mathbf{soft}(a, 1)$. Is it a non-decreasing function if we restrict the function domain to $a > 0$?

3. (1 pt) Suppose we fix all but j-th coordinate of θ and optimize the objective (1) over θ_j. **Show that** the one dimensional objective obtained by fixing all other coordinates of θ except the j-th coordinate of the LASSO objective (1) in the coordinate descent algorithm is given by

$$g : \mathbb{R} \mapsto \mathbb{R}$$
$$g(\alpha) = \sum_{i=1}^{n} \left[\alpha x_{ij} + \sum_{k=1,k\neq j}^{d} \theta_k x_{ik} - y_i\right]^2 + \lambda\,|\alpha| + \lambda \sum_{k=1,k\neq j}^{d} |\theta_k|\,, \tag{3}$$

where we have introduced α to denote the role of θ_j and simplify our discussion later on.

For the following parts, we remark that the function g is convex. The only thing keeping g from being differentiable is the term with $|\alpha|$. Moreover, the function g is differentiable at

every point except $\alpha = 0$. So we will break the problem into 3 cases $\alpha > 0$, $\alpha < 0$ and $\alpha = 0$. It will be convenient to use the following expression as shorthand notations for the discussion to follow:

$$\text{sign}(\alpha) := \begin{cases} 1 & \alpha > 0 \\ 0 & \alpha = 0 \\ -1 & \alpha < 0 \end{cases}$$

$$a_j := 2\sum_{i=1}^{n} x_{ij}^2$$

$$c_j := 2\sum_{i=1}^{n} x_{ij}\left(y_i - \sum_{k=1,k\neq j}^{d} \theta_k x_{ik}\right).$$

4. (1 pt) **Write down** the expression for the derivative $g'(\alpha)$ for $\alpha \neq 0$. You can assume that $\sum_{i=1}^{n} x_{ij}^2 > 0$.

5. (1 pt) Suppose α^* is the point that minimizes g and assume that it is strictly positive, i.e., $\alpha^* > 0$. Then **show that** $\alpha^* = \frac{1}{a_j}(c_j - \lambda)$.

6. (1 pt) Now consider the other case, when α^* minimizes g and is strictly negative, i.e., $\alpha^* < 0$. Then, **show that** $\alpha^* = \frac{1}{a_j}(c_j + \lambda)$.

7. (3 pts) Now **give conditions** on c_j that imply that a minimizer α^* is positive and **conditions** for which a minimizer α^* is negative.

To analyze the case when $\alpha^* = 0$, we have to recall a few more mathematical concepts. For the function $g : \mathbb{R} \to \mathbb{R}$, its one-sided derivatives are defined as follows

$$D^+(g)(\alpha) = \lim_{\epsilon>0,\epsilon\to 0} \frac{g(\alpha+\epsilon) - g(\alpha)}{\epsilon}, \quad D^-(g)(\alpha) = \lim_{\epsilon>0,\epsilon\to 0} \frac{g(\alpha-\epsilon) - g(\alpha)}{\epsilon}.$$

Also **note the following useful fact**: If g is a convex function on $\mathbb{R}$, $\tilde{\alpha}$ is a minimizer of g if and only if

$$D^+(g)(\tilde{\alpha}) \geq 0 \text{ and } D^-(g)(\tilde{\alpha}) \geq 0. \tag{4}$$

8. (2 pts) **Derive the expressions** for the two one-sided derivatives of g at $\alpha = 0$.

9. (2 pts) **Show that** if $c_j \in [-\lambda, \lambda]$, then the minimizer of g satisfies $\alpha^* = 0$.
Hint: You may use the result (4).

10. (2 pts) Putting together the results from parts 5,6,7 and 8 we can conclude that the minimizer of g takes the following form

$$\alpha^* = \begin{cases} \frac{1}{a_j}(c_j - \lambda) & c_j > \lambda \\ 0 & c_j \in [-\lambda, \lambda] \\ \frac{1}{a_j}(c_j - \lambda) & c_j < -\lambda. \end{cases}$$

Show that this is exactly equivalent to the soft-thresholding applied in Algorithm 1.

11. (2 pts) **Discuss** qualitatively what the algorithm is doing and **why** it is a reasonable algorithm if we want a sparse solution.

3 Regression analysis on Ames Housing dataset (21 pts, Readable code snippet required in the write-up)

First download the Ames Housing dataset (AmesHousing.txt) from Piazza. You can find a complete description of the variables here (http://jse.amstat.org/v19n3/decock/DataDocumentation.txt). The dataset contains information from the Ames Assessor's Office used in computing assessed values for individual residential properties sold in Ames, IA from 2006 to 2010.

You may load the dataset as follows

```
# SET YOUR OWN WORKING DIRECTORY
# setwd("/Users/yuansichen/UCB/Teaching/2019_Spring/Problems/stat154copy/hws/hw4/")

# load the Ames txt data
Ames <- read.delim("AmesHousing.txt", header = TRUE, sep = "\t", dec = ".")
```

Motivation: A regression analysis can be used to answer typical questions as follows

- What is the predicted sale price of a 2B2B house with 1500 square feet living area?
- Which is the more important variable in predicting sale price, garage area or open porch area?

This type of analysis could be useful if you want to buy or sell a house in Ames area.

Data preprocessing: For the purpose of this homework, we introduce one particular data preprocessing pipeline (might not be optimal) for this dataset. We will only use a limited number of variables. **From now on, you will use the data.frame *AmesTiny* as the main data frame. Please split the training and test dataset exactly as we did in the following code (with the same seed).**

```
### DO NOT CHANGE THIS PART, BEGIN
# load the Ames txt data
Ames <- read.delim("AmesHousing.txt", header = TRUE, sep = "\t", dec = ".")

continuousVar <- colnames(Ames)[grep("Frontage|SF|Area|Porch",
                                     colnames(Ames))]

AmesTiny <- Ames[, c(continuousVar,
                     c("Overall.Qual",
                       "Overall.Cond","Neighborhood",
                       "SalePrice"))]
```

```r
# check NA
# colSums(is.na(AmesTiny))
# fill the Garage.Area NA with 0
AmesTiny$Garage.Area[is.na(AmesTiny$Garage.Area)] = 0
# change factor variable to actual factor in the data frame
AmesTiny$Overall.Qual <- factor(AmesTiny$Overall.Qual)
AmesTiny$Overall.Cond <- factor(AmesTiny$Overall.Cond)
# fill the continuous variable with column mean
for(i in 1:ncol(AmesTiny)){
  AmesTiny[is.na(AmesTiny[,i]), i] <- mean(AmesTiny[,i], na.rm = TRUE)
}

# divide the data into training and test datasets
set.seed(12345678)
testSize <- floor(nrow(AmesTiny)*0.1)
testIndex <- sample(seq_len(nrow(AmesTiny)), size = testSize)
AmesTinyTrain <- AmesTiny[-testIndex, ]
AmesTinyTest <- AmesTiny[testIndex, ]
### DO NOT CHANGE THIS PART, END
```

3.1 Fitting OLS (7 pts)

Use **lm()** to fit the following regression models to predict log(SalePrice + 1) using a handful of predictors. As you will notice, the complexity of the models will increase from one to the next.

Model 1: $(d = 2)$

$$\textbf{log(SalePrice + 1)} = \beta_0 + \beta_1 \textbf{Gr.Liv.Area} + \epsilon$$

Model 2: $(d = 3)$

$$\textbf{log(SalePrice + 1)} = \beta_0 + \beta_1 \textbf{Gr.Liv.Area} + \beta_2 \textbf{Garage.Area} + \epsilon$$

Model 3: $(d = 4)$

$$\textbf{log(SalePrice + 1)} = \beta_0 + \beta_1 \textbf{Gr.Liv.Area} + \beta_2 \textbf{Garage.Area} + \beta_3 \textbf{Open.Porch.SF} + \epsilon$$

Model 4: $(d = 5)$

$$\textbf{log(SalePrice + 1)} = \beta_0 + \beta_1 \textbf{Gr.Liv.Area} + \beta_2 \textbf{Garage.Area} + \beta_3 \textbf{Open.Porch.SF} + \beta_4 \textbf{Lot.Area} + \epsilon$$

Model 5: $(d = 14)$

$$\textbf{log(SalePrice + 1)} = \beta_0 + \beta_1 \textbf{Gr.Liv.Area} + \beta_2 \textbf{Garage.Area} + \beta_3 \textbf{Open.Porch.SF} + \beta_4 \textbf{Lot.Area} + (\beta_5, \ldots, \beta_{5+8})^\top \textbf{Dummified.Overall.Qual} + \epsilon$$

Model 6: $(d = 22)$

$$\textbf{log(SalePrice + 1)} = \beta_0 + \beta_1\textbf{Gr.Liv.Area} + \beta_2\textbf{Garage.Area} + \beta_3\textbf{Open.Porch.SF} + \beta_4\textbf{Lot.Area} + (\beta_5, \ldots, \beta_{5+8})^\top\textbf{Dummified.Overall.Qual} + (\beta_{14}, \ldots, \beta_{14+7})^\top\textbf{Dummified.Overall.Cond} + \epsilon$$

Model 7: $(d = 23)$

$$\textbf{log(SalePrice + 1)} = \beta_0 + \beta_1\textbf{Gr.Liv.Area} + \beta_2\textbf{Garage.Area} + \beta_3\textbf{Open.Porch.SF} + \beta_4\textbf{Lot.Area} + (\beta_5, \ldots, \beta_{5+8})^\top\textbf{Dummified.Overall.Qual} + (\beta_{14}, \ldots, \beta_{14+7})^\top\textbf{Dummified.Overall.Cond} + \beta_{22}\textbf{log(Gr.Liv.Area+1)} + \epsilon$$

Model 8: $(d = 24)$

$$\textbf{log(SalePrice + 1)} = \beta_0 + \beta_1\textbf{Gr.Liv.Area} + \beta_2\textbf{Garage.Area} + \beta_3\textbf{Open.Porch.SF} + \beta_4\textbf{Lot.Area} + (\beta_5, \ldots, \beta_{5+8})^\top\textbf{Dummified.Overall.Qual} + (\beta_{14}, \ldots, \beta_{14+7})^\top\textbf{Dummified.Overall.Cond} + \beta_{22}\textbf{log(Gr.Liv.Area+1)} + \beta_{23}\textbf{log(Gr.Liv.Area+1)}^2 + \epsilon$$

Model 9: $(d = 25)$

$$\textbf{log(SalePrice + 1)} = \beta_0 + \beta_1\textbf{Gr.Liv.Area} + \beta_2\textbf{Garage.Area} + \beta_3\textbf{Open.Porch.SF} + \beta_4\textbf{Lot.Area} + (\beta_5, \ldots, \beta_{5+8})^\top\textbf{Dummified.Overall.Qual} + (\beta_{14}, \ldots, \beta_{14+7})^\top\textbf{Dummified.Overall.Cond} + \beta_{22}\textbf{log(Gr.Liv.Area+1)} + \beta_{23}\textbf{log(Gr.Liv.Area+1)}^2 + \beta_{24}\textbf{log(Gr.Liv.Area+1)}^3 + \epsilon$$

Model 10: $(d = 26)$

$$\textbf{log(SalePrice + 1)} = \beta_0 + \beta_1\textbf{Gr.Liv.Area} + \beta_2\textbf{Garage.Area} + \beta_3\textbf{Open.Porch.SF} + \beta_4\textbf{Lot.Area} + (\beta_5, \ldots, \beta_{5+8})^\top\textbf{Dummified.Overall.Qual} + (\beta_{14}, \ldots, \beta_{14+7})^\top\textbf{Dummified.Overall.Cond} + \beta_{22}\textbf{log(Gr.Liv.Area+1)} + \beta_{23}\textbf{log(Gr.Liv.Area+1)}^2 + \beta_{24}\textbf{log(Gr.Liv.Area+1)}^3 + \beta_{25}\textbf{log(Gr.Liv.Area+1)}^4 + \epsilon$$

Model 11: $(d = 27)$

$$\textbf{log(SalePrice + 1)} = \beta_0 + \beta_1\textbf{Gr.Liv.Area} + \beta_2\textbf{Garage.Area} + \beta_3\textbf{Open.Porch.SF} + \beta_4\textbf{Lot.Area} + (\beta_5, \ldots, \beta_{5+8})^\top\textbf{Dummified.Overall.Qual} + (\beta_{14}, \ldots, \beta_{14+7})^\top\textbf{Dummified.Overall.Cond} + \beta_{22}\textbf{log(Gr.Liv.Area+1)} + \beta_{23}\textbf{log(Gr.Liv.Area+1)}^2 + \beta_{24}\textbf{log(Gr.Liv.Area+1)}^3 + \beta_{25}\textbf{log(Gr.Liv.Area+1)}^4 + \beta_{26}\textbf{log(Gr.Liv.Area+1)}^5 + \epsilon$$

1. (1 pt) **Implement your R function *MSE()* (show your code in the write-up)** which takes regression coefficient $\beta \in \mathbb{R}^d$, new data matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ and response $\mathbf{y} \in \mathbb{R}^n$ as input, and output the mean squared error.

$$MSE = \frac{1}{n} \|\mathbf{y} - \mathbf{X}\beta\|_2^2 .$$

2. (1 pt) **Implement your R function *R2()* (show your code in the write-up)** which takes regression coefficient $\beta \in \mathbb{R}^d$, a data matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ and response $\mathbf{y} \in \mathbb{R}^n$ as input, and outputs coefficient of determination:

$$R^2 = \mathbf{cor}\left(\mathbf{y}, \mathbf{X}\beta\right)^2 ,$$

   where **cor** denotes correlation.

3. (3 pt) Use your functions **MSE()**, **R2()** to create a data frame **modelQuality** for for these measures (MSE, R^2) on the training data set **AmesTinyTrain** for all 11 models above. The data frame **modelQuality** should have 2 columns and 11 rows. Check your results of MSE and R2 with R's implementation (Create another data frame **modelQualityRImp** using **summary()** on the output of **lm()**). **Show code and print modelQuality and modelQualityRImp side by side**. Which model has smallest training MSE?
   Hint: you can get the data design matrix from the **lm()** output using **model.matrix()**.

```
# columns involved in 11 models, you might want to use a for loop to loop over models
Xnames <- c("Gr.Liv.Area", "Garage.Area","Open.Porch.SF",
                      "Lot.Area",
                      "factor(Overall.Qual)",
                       "factor(Overall.Cond)",
                      "I(log(Gr.Liv.Area+1))",
                       "I(log(Gr.Liv.Area+1)^2)",
                       "I(log(Gr.Liv.Area+1)^3)",
                      "I(log(Gr.Liv.Area+1)^4)","I(log(Gr.Liv.Area+1)^5)")
```

4. (1 pt) **Plot the training MSEs against the number of predictors in the model and describe in one sentence any trend in the graph (show only plot no code required)**. The number of predictors is treated as a rough measure of model complexity here.

5. (1 pt) *Normally, it is not allowed to check the test data set until the last step of the analysis.* Here, for the purpose of this homework, **plot both the training MSEs and test MSEs against the number of predictors in the model. Is test MSE always decreasing? Which model has the lowest test MSE? (show only the plot, no code required).** Hint: Use **predict()**

3.2 Single validation set (or holdout method) for model selection (2 pts)

As we mentioned in class, the training MSEs are not an appropriate way to assess the predictive power of the models, since the training set, which is used for calibrating models, is also used for model testing.

The training (or in-sample) MSEs tend to have an optimistic bias towards complicated models. There are in-sample metrics that take model complexity into account, such as Akaike's information criterion (AIC) and Bayesian Information Criterion (BIC), but these are not applicable when an explicit likelihood is not present in the model.

The most straightforward approach is to split the training dataset into two parts: one for actual training and one for validation. This approach is commonly known as the validation method. A common split is 80-20: use 80% of the data to train the model and 20% to validate the model.

1. (0pts) Select 20% of your dataset as holdout. You should simply copy the code below.

```
### DO NOT CHANGE THIS PART, BEGIN
set.seed(123456)
valSize <- floor(nrow(AmesTinyTrain)*0.2)
valIndex <- sample(seq_len(nrow(AmesTinyTrain)), size = valSize)
# actual training data
AmesTinyActTrain <- AmesTinyTrain[-valIndex, ]
AmesTinyActVal <- AmesTinyTrain[valIndex, ]
### DO NOT CHANGE THIS PART, END
```

2. (2pts) For each of the 11 regression models, train on the remaining 80% of the data, predict the validation data, and compute the validation MSE. For this, **create a new data frame *modelQualitySingleVal* for these measures (trainMSE, valMSE) and plots these values. Identify which model gives the lowest validation MSE. Show the plots and the code for this part in the write-up.**

3.3 Cross validation for model selection (3 pts)

Cross-validation is an alternative to the single validation method. A useful package for such a task is **caret**. To generate folds, we can use **createFolds()**.

```
library(caret)
# create 10 folds
folds <- createFolds(AmesTinyTrain$SalePrice, k = 10)
```

Note that **folds** contains a list of vectors of indices. Since randomness is involved, you might get a different list. This is why you should also specify a random seed with **set.seed()** so you can replicate your analyses.

1. (2 pts) Use **createFolds()** to create a list folds to do a 5-fold cross validation to estimate the prediction error for log(SalePrice+1). Specifically, for each fold, for each model, train the model based on all observations except the ones in the fold and compute the validation MSE on data in that fold. You shold end up having a 11×5 matrix with rows corresponding the models and columns corresponding to the folds. **Show the code for this part in the write-up.**

2. (1 pts) The CV-MSE is the average MSE over folds.

$$\text{MSE}_{CV} = \frac{1}{\#\text{folds}} \sum_{i=1}^{\#\text{folds}} \text{MSE on i-th fold.}$$

Calculte CV-MSE for each model. **Plot trainMSE, testMSE, singleValMSE, CV-MSEs again the number of features in each model in the same plot. Which model gives the lowest CV-MSE? Show the plots and the code for this part in the write-up.**

3.4 Fitting ridge regression (3 pts)

1. (1 pts) Use ridge regression to predict log(SalePrice+1) with all predictors in **AmesTiny** and regularization parameter $\lambda = 1.0$. Make sure you dummify the factor variables and then scale the design matrix before fitting your model. **Show the code for this part in the write-up.**

2. (2 pts) Varying the parameter λ with ridge regression (Your λ should contain 0.1, 1000 and at least 10 numbers in between), plot trainMSE and CV-MSEs of ridge against λ in the same plot. **Which λ achieves the minimum training MSE and CV-MSEs? Show the plots and the code for this part in the write-up.**

3.5 Fitting LASSO (2 pts)

1. (2 pts) Run LASSO to predict log(SalePrice+1) with all predictors in **AmesTiny**. Varying the parameter λ with LASSO, plot trainMSE and CV-MSEs of LASSO against λ in the same plot. **Which λ achieves the minimum training MSE and CV-MSEs? Show the plots and the code for this part in the write-up.**

3.6 Variable Importance (4 pts)

Recall from lecture notes that a feature importance plot is a bar plot which has feature names as x-axis and the absolute values of a subset of coordinates of β as y-axis. You might want to use the **geom_bar** function in **ggplot2**.

1. (2 pts) For the ridge estimate $\hat{\beta}_{\text{ridge}}$ which achieves the best CV-MSE in the subsection 3.4, **show the feature importance plot for the top 10 coordinates of $\hat{\beta}_{\text{ridge}}$ with the largest absolute values.**

2. (2 pts) For the LASSO estimate $\hat{\beta}_{\text{LASSO}}$ which achieves the best CV-MSE in the subsection 3.5, **show the feature importance plot for the top 10 coordinates of $\hat{\beta}_{\text{LASSO}}$ with the largest absolute values.**