STAT 154: Homework 4

Release date: **Thursday, Mar 7**

Due by: **11 PM, Wednesday, Mar 20**

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 "HW4 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 "HW4 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 ordinary least squares and other regression methods. Some problems are from the **ESL book** 12th printing (The Elements of Statistical Learning) that is available at the following website:
https://web.stanford.edu/ hastie/ElemStatLearn/.)

1 True or False (8 pts)

Examine whether the following statements are true or false and *provide one line justification.* Linear model in the following statements refers to the linear model studied in class (see equation (1) for a concrete reference.)

(a) Under the linear model, the OLS estimator of the regression coefficients is unbiased.

(b) For the linear model, bias of the ridge regression increases and the variance decreases as we increase the regularization parameter λ.

(c) The LASSO, relative to least squares, is less flexible and hence will give improved prediction accuracy when its increase in variance is less than its decrease in bias.

(d) In LASSO, no matter how large you choose regularization λ, the estimator $\hat{\beta}(\lambda)$ will never be the vector 0.

(e) In LASSO, as λ increases, the ℓ_1-norm of the estimator $\hat{\beta}(\lambda)$ always decreases.

(f) Every eigenvalue of an idempotent matrix is always either zero or one. (Recall that A square matrix $M \in \mathbb{R}^{m \times m}$ is called *idempotent* if $M^2 = M$.)

(g) Let X be an $n \times d$ matrix of full rank. Let $H = X\left(X^\top X\right)^{-1} X^\top$. The matrix H is symmetrix, idempotent and PSD.

(h) Let $Q = \mathbb{I}_n - H$ where H is defined in the previous part. We have trace$(Q) = n$.

Fun fact: Note that for a feature matrix X with full column rank the matrix $H = X\left(X^\top X\right)^{-1} X^\top$ is called the *hat matrix* because in ordinary least squares, the predicted responses are given by $\hat{y} = Hy$, i.e., the matrix H adds a hat to y.

2 OLS theoretical properties (9 pts)

1. (2 pts) Consider a linear model where we observe the samples (x_i, y_i) for $i = 1, \ldots, n$, that are generated as follows

$$y_i = x_i^\top \beta^* + \epsilon, \tag{1}$$

where the error $\epsilon \overset{i.i.d.}{\sim} \mathcal{N}(0, \sigma^2)$. **Show that the OLS estimate $\hat{\beta}$ on data $\mathbf{X} \in \mathbb{R}^{n \times d}$ and $\mathbf{Y} \in \mathbb{R}^n$ satisfies**

$$\hat{\beta} \sim \mathcal{N}\left(\beta^*, \sigma^2 \left(\mathbf{X}^\top \mathbf{X}\right)^{-1}\right).$$

Also conclude that $\mathbb{E}\left[\mathbf{X}\hat{\beta}\right] = \mathbf{X}\beta^*$.

2. (2 pts) In the Gaussian linear model described above, **show that** $\mathbb{E}[RSS] = \sigma^2(n - d)$.

3. (2 pts) **ESL book Ex. 3.3 (a)** (part (b) is not needed). *Hint*: The notion unbiased linear estimator is explained in Section 3.2.2 around equation (3.19).

4. (3 pts) **ESL book Ex. 2.9**. We expect a solution that **does not** use the explicit data generation process. *Partial credit is given if you use the explicit data generation process.*

3 Theory of Ridge Regression (14 pts)

Given the feature matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ and the responses $\mathbf{y} \in \mathbb{R}^n$, ridge regression solves the following penalized least squares problem:

$$\min_{\theta} \|\mathbf{X}\theta - \mathbf{y}\|_2^2 + \lambda \|\theta\|_2^2 \tag{2}$$

Let $\mathbf{X}^\top \mathbf{X}$ have the following eigen-decomposition: $\mathbf{X}^\top \mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{U}^\top$ where $\mathbf{D}$ is a diagonal matrix with non-negative entries.

Let $\widehat{\theta}_\lambda$ denote the solution for the problem (2) above.

1. (1 pt) **Show that for any $\lambda > 0$, the solution $\theta^{\mathrm{RR}}(\lambda)$ is unique and derive its expression.**

2. (2 pts) For *all the following parts*, we assume the linear regression model: That is the data is generated as follows:

   $$\mathbf{y} = \mathbf{X}\theta^* + \boldsymbol{\epsilon}$$

   where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \sigma^2 I_n)$. Using the previous part, **show that the distribution of the ridge-estimate is given as follows:**

   $$\widehat{\theta}_\lambda \sim \mathcal{N}(\mathbf{W}_\lambda \theta^*, \sigma^2 \mathbf{W}_\lambda (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I}_d)^{-1})$$
   $$\text{where} \quad \mathbf{W}_\lambda = (\mathbf{X}^\top \mathbf{X} + \lambda \mathbf{I}_d)^{-1} \mathbf{X}^\top \mathbf{X}.$$

3. (4 pts) Let $\mathbf{X}^\top \mathbf{X} = \mathbf{U}\mathbf{D}\mathbf{U}^\top$ with $\mathbf{D}_{ii} = d_i$. **Show the following:**

   (a) The squared bias is given by

   $$\textbf{squared bias} = \left\| \mathbb{E}[\widehat{\theta}_\lambda] - \theta^* \right\|_2^2 = \sum_{i=1}^{d} \frac{\lambda^2}{(d_i + \lambda)^2} (v_i)^2,$$

   where $v = \mathbf{U}^\top \theta^*$.

   (b) The (scalar) variance is given by

   $$\textbf{scalar-variance} = \mathbb{E} \left\| \widehat{\theta}_\lambda - \mathbb{E}[\widehat{\theta}_\lambda] \right\|_2^2 = \sigma^2 \sum_{i=1}^{d} \frac{d_i}{(d_i + \lambda)^2}.$$

   Note that here we are considering the scalar versions of the bias and variance defined in class by taking the norms of the corresponding quantities.

4. (2 pts) **What is the value of the squared-bias and variance at $\lambda = 0$ and as $\lambda \to \infty$. Do you see a trade-off in the squared-bias and variance change as λ increases?**

5. (2 pts) Recall the definition of the moment matrix $\mathbf{M}$ from class:

$$\mathbf{M}(\lambda) = \mathbb{E}[(\widehat{\theta}_\lambda - \theta^*)(\widehat{\theta}_\lambda - \theta^*)^\top].$$

Recall the mean-squared error

$$\mathrm{MSE}(\widehat{\theta}_\lambda) := \mathbb{E}[\left\|\widehat{\theta}_\lambda - \theta^*\right\|_2^2].$$

Show that $\mathbb{E}[\left\|\widehat{\theta}_\lambda - \theta^*\right\|_2^2] = \mathrm{trace}(\mathbf{M}(\lambda))$**. Moreover, show that**

$$\mathbf{MSE}(\widehat{\theta}_\lambda) = \mathbf{squared\text{-}bias} + \mathbf{scalar\text{-}variance}$$

6. (3 pts) Recall that when $\mathbf{X}^\top\mathbf{X}$ is invertible, the OLS-estimator is unbiased. For this case, **show that its mean squared error satisfies** $\mathbf{MSE}(\theta^{\mathrm{OLS}}) = \mathrm{trace}(\mathbf{M}(0))$**. Furthermore, show that there exists a range of** $\lambda > 0$ **for which**

$$\mathrm{MSE}(\widehat{\theta}_\lambda) < \mathrm{MSE}(\theta^{\mathrm{OLS}}).$$

Conclude that there always exists a range of $\lambda > 0$**, for which the MSE is smaller for ridge regression when compared to OLS in the Gaussian linear model.**

4 Gradient descent for simple functions (8 pts)

A function $f : \mathbb{R}^d \to \mathbb{R}$ is called *convex*, if

$$\forall x \in \mathbb{R}^d, y \in \mathbb{R}^d, \lambda \in [0,1], f(\lambda x + (1-\lambda)y) \leq \lambda f(x) + (1-\lambda)f(y).$$

Recall Taylor's theorem for twice differentiable functions of vectors, which holds for all $x, y \in \mathbb{R}^d$:

$$f(y) = f(x) + \nabla f(x)^\top (y-x) + \frac{1}{2}(y-x)^\top \nabla^2 f(\widetilde{x})(y-x),$$

for some $\widetilde{x}$. One can show that a *twice differentiable function* f is convex if and only if $\nabla^2 f(x)$ is positive semidefinite for all $x \in \mathbb{R}^d$. You can take this fact for granted.[1]

1. (2 pts) Let $L \geq 0$. Consider the function of one variable $f(x) = \frac{L}{2}x^2$. **Show that it is convex. Derive the gradient descent update where we use a step-size of** γ **and start at some point** $x^{(0)} \neq 0$**.**

2. (3 pts) **What does the behavior of the updates look like for the above setting and the choices** $\gamma \in \{1/L, 2/L\}$**? What happens when we use a step size** $\gamma \in \left[0, \frac{2}{L}\right)$ **such that** $\gamma \neq 1/L$**? For this step size, how many steps does it take for gradient descent updates to converge to within** ϵ **of the optimum (as a function of the tuple** $(\gamma, L, |x^{(0)}|, \epsilon)$**)?**

[1] This problem draws inspiration from the class CS 189.

3. (1 pt) **How do your answers in the previous part change if $f(x) = \frac{L}{2}(x-c)^2$ for some constant c?**

4. (2 pts) Let $L \geq m \geq 0$. Now consider the function of two variables $f(x) = \frac{L}{2}x_1^2 + \frac{m}{2}x_2^2$. **Show that the function is convex by computing its Hessian $\nabla^2 f(x)$. Derive closed form expressions for the iterations if we start at the point (a, b), and run gradient descent with step-size γ.** Start by writing out the result of the first iteration as $A \begin{bmatrix} a \\ b \end{bmatrix}$ for some matrix A.

5 High dimensional regression (9 pts, readable code snippet required in the write-up)

Suppose we have a design matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ where $d > n$, i.e., we have many more predictor variables than observations. Moreover, we have quantitative response vector $\mathbf{y} \in \mathbb{R}^n$, and we plan to fit a linear regression model.

1. (2 pts) **Explain why the ordinary least squares solution for $(\mathbf{X}, \mathbf{y})$ is not unique. What can you say about the residuals of any OLS solution?**

2. (1 pt) **Is the ridge regression solution unique? Why or why not?**

3. (1 pt) Suppose you compute a series of ridge solutions $\widehat{\beta}(\lambda)$ for $\mathbf{X}$ and $\mathbf{y}$, letting λ get monotonically smaller. **What can you say about the limiting ridge solution as $\lambda \downarrow 0$?**

4. (1 pt) (Code) Fix $n = 1000, d = 5000$. Generate a design matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ with each entry drawn i.i.d. from $\mathcal{N}(0, 1)$, an error vector $\boldsymbol{\epsilon} \in \mathbb{R}^n$ with each entry drawn i.i.d. from $\mathcal{N}(0, 0.25)$ and a response vector $\mathbf{y} \in \mathbb{R}^n$, satisfying

$$\mathbf{y} = \mathbf{X}\beta^* + \boldsymbol{\epsilon},$$

where $\beta^* = (\underbrace{1, \ldots, 1}_{15}, 0, \ldots, 0)^\top$ having 15 non-zero entries. **Show your code for this part in the write-up. *Do not* display any data.**

5. (1 pt) (Code) Split the n samples into training (size $4n/5$) and test (size $n/5$) sets. **Show your code for this part in the write-up**.

6. (1 pt) (Code) Using the **glmnet** package, fit ridge regression on the training data. Plot the training MSE vs lambda and test MSE vs lambda in the same plot with two different colors. We expect you to choose a large range of lambdas so that the test MSE is not monotone. **Show the plots and the code for this part in the write-up.**

7. (1 pt) (Code) Using the **glmnet** package, fit LASSO regression on the training data. Plot the training MSE vs lambda and test MSE vs lambda in the same plot with two different colors. We expect you to choose a large range of lambdas so that the test MSE is not monotone. **Show the plots and the code for this part in the write-up.**

8. (1 pt) **Which model ridge or LASSO and what regularization parameter has the smallest test MSE?**

6 Regression analysis on Ames Housing dataset (15 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
```

6.1 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()**

6.2 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.**

6.3 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.**

6.4 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.**