Homework Set 1, CPSC 8420, Spring 2022

Sherrer, William

Due 03/03/2022, Thursday, 11:59PM EST

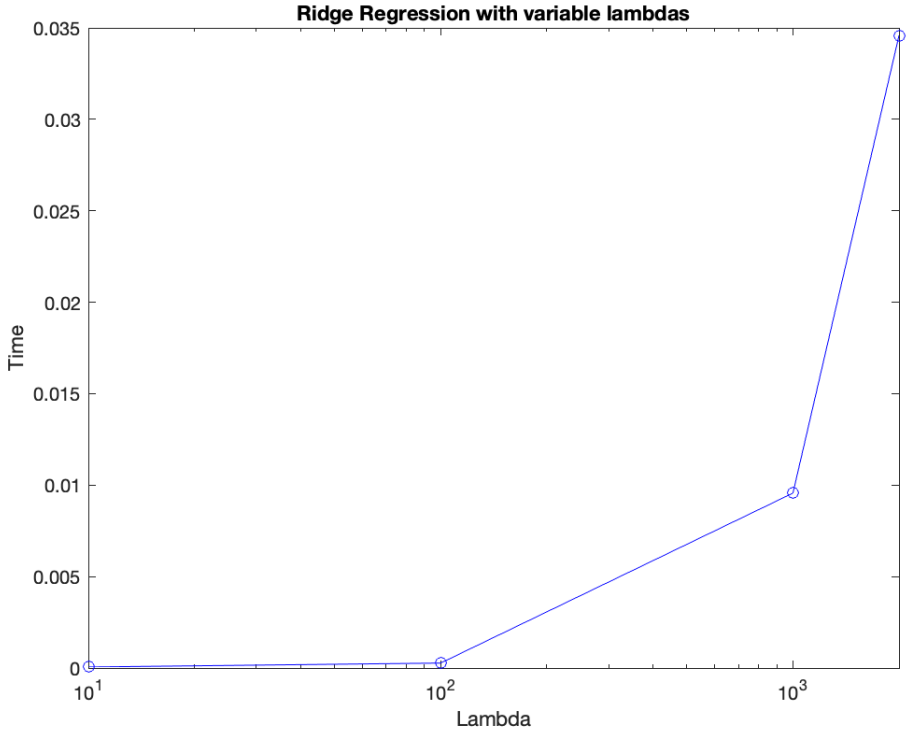
Ridge Regression

Please show that for arbitrary $\mathbf{A} \in \mathbb{R}^{n\times p}$, $(\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I}_p)^{-1}\mathbf{A}^T = \mathbf{A}^T(\mathbf{A}\mathbf{A}^T + \lambda\mathbf{I}_n)^{-1}$, where $\lambda > 0$. Now assume $n = 100$, please compare the time consumption when $p = [10, 100, 1000, 2000]$ and plot the results appropriately (*e.g.* in one figure where X-axis denotes p while Y-axis the time consumption).

Let $\mathbf{A} \in \mathbb{R}^{n\times p}$, we will use Singular Value Decomposition to compute U, S, V such that $U * S * V^T = A$ and $V * S * U^T = A^T$.Therefore, $AA^T = USV^TVSU^T = US^2U^T$ and $A^TA = VSU^TUSV^T = VS^2V^T$ where $UU^T = I_n$ and $VV^T = I_p$
Using SVD, we can show that:

$$
\begin{aligned}
(\mathbf{A}^T\mathbf{A} + \lambda\mathbf{I}_p)^{-1}\mathbf{A}^T &= (VS^2V^T + \lambda VV^T)^{-1}VSU^T \\
&= (V(S^2 + \lambda)V^T)^{-1}VSU^T \\
&= V(S^2 + \lambda)^{-1}V^TVSU^T \\
&= V(S^2 + \lambda)^{-1}SU^T
\end{aligned}
$$

$$
\begin{aligned}
\mathbf{A}^T(\mathbf{A}\mathbf{A}^T + \lambda\mathbf{I}_n)^{-1} &= VSU^T(US^2U^T + \lambda UU^T)^{-1} \\
&= VSU^T(U(S^2 + \lambda)U^T)^{-1} \\
&= VSU^TU(S^2 + \lambda)^{-1}U^T \\
&= VS(S^2 + \lambda)^{-1}U^T
\end{aligned}
$$



Bias–variance trade-off for k-NN

Assume $y = f(x) + \epsilon$ where $E(\epsilon) = 0, Var(\epsilon) = \sigma^2$. Please show that:

$$Err(x_0) = \sigma^2 + \frac{\sigma^2}{k} + [f(x_0) - \frac{1}{k}\sum_{l=1}^{k} f(x_l)]^2, \tag{1}$$

where x_l denotes the nearest neighbour data. Please justify Bias and Variance change when k increases and explain if necessary.

$E[(y - \hat{f})^2] = E[(f + \epsilon - \hat{f})^2] = E[(f + \epsilon - \hat{f} + E[\hat{f}] - E[\hat{f}])^2]$
$= E[(f - E[\hat{f}])^2] + E[\epsilon^2] + E[(E[\hat{f}] - \hat{f})^2] + 2E[(f - E[\hat{f})\epsilon] + 2E[\epsilon(E[\hat{f}] - \hat{f})] + 2E[(E[\hat{f}] - \hat{f}])(f - E[\hat{f})]$
$= (f - E[\hat{f}])^2 + E[\epsilon^2] + E[(E[\hat{f}] - \hat{f})^2] + 2(f - E[\hat{f}])E[\epsilon] + 2E[\epsilon]E[E[\hat{f}] - \hat{f}] + 2E[E[\hat{f}] - \hat{f}](f - E[\hat{f}]$
$= (f - E[\hat{f}])^2 + E[\epsilon^2] + E[(E[\hat{f}] - \hat{f})^2]$
$= (f - E[\hat{f}])^2 + Var[\epsilon] + Var[\hat{f}]$
$= Bias[\hat{f}]^2 + Var[\epsilon] + Var[\hat{f}]$
$= Bias[\hat{f}]^2 + \sigma^2 + Var[\hat{f}]$
In the case of k-NN it should be noted that all X_i are fixed in training set. i.e. $\tau = (x_i, Y_i)_{i=1}^N$
Below I will use the subscript τ for variables that depend on the training set. We say that the Error can be equated as:

$$Err(x_0) = E_\tau[(Y - \hat{f}_\tau(x_0))^2 | X = x_0] = (f(x_0) - E_\tau[\hat{f}_\tau(x_0)])^2 + E_\tau[(f(x_0) - E_\tau[\hat{f}_\tau(x_0)])^2] + \sigma^2 \tag{2}$$

Where $(f(x_0) - E_\tau[\hat{f}_\tau(x_0)])^2 = Bias^2$; $E_\tau[(f(x_0) - E_\tau[\hat{f}_\tau(x_0)])^2] = Variance$; and $\sigma^2 = noise$

We can simplify this expression first evaluating $E_\tau[\hat{f}_\tau(x_0)]$, accounting for $Y = f(x) + \epsilon$

$$E_\tau[\hat{f}_\tau(x_0)] = E_\tau[\frac{1}{k}\sum_{l=1}^{k} Y_{\tau,(l)}] = E_\tau[\frac{1}{k}\sum_{l=1}^{k}(f(x_l) + \epsilon_{\tau,l})] = \frac{1}{k}\sum_{l=1}^{k} f(x_l) + \frac{1}{k}\sum_{l=1}^{k} E_\tau[\epsilon_{\tau,l}] = \frac{1}{k}\sum_{l=1}^{k} f(x_l) \tag{3}$$

From that we can derive the $Bias^2(x_0)$ and $Variance(x_0)$:

$Bias^2(x_0) = (f(x_0) - E_\tau[\hat{f}_\tau(x_0)])^2 = (f(x_0) - \frac{1}{k}\sum_{l=1}^{k} f(x_l))^2$

$Variance(x_0) = E_\tau[(f(x_0) - E_\tau[\hat{f}_\tau(x_0)])^2]$

$= E_\tau[(\frac{1}{k}\sum_{l=1}^{k} Y_{\tau,l} - \frac{1}{k}\sum_{l=1}^{k} f(x_l))^2]$

$= E_\tau[(\frac{1}{k}\sum_{l=1}^{k}(f(x_l) + \epsilon_{\tau,l}) - \frac{1}{k}\sum_{l=1}^{k} f(x_l))^2]$

$= E_\tau[(\frac{1}{k}\sum_{l=1}^{k} \epsilon_{\tau,l})^2] = \frac{1}{k^2}E_\tau[(\sum_{l=1}^{k} \epsilon_{\tau,l})^2]$

$= \frac{1}{k^2}E_\tau[(\sum_{l=1}^{k} \epsilon_{\tau,l} - \underbrace{E\tau[\sum_{l=1}^{k} \epsilon_{\tau,l}]}_{=0})^2] = \frac{1}{k^2}Var_\tau(\sum_{l=1}^{k} \epsilon_{\tau,l})$

$= \frac{1}{k^2}\sum_{l=1}^{k} Var_\tau(\epsilon_{\tau,l}) = \frac{k\sigma^2}{k^2} = \frac{\sigma^2}{k}$

Therefore we put the components together to arrive at the final equation

$$Err(x_0) = \sigma^2 + \underbrace{\frac{\sigma^2}{k}}_{\text{Variance}} + \underbrace{[f(x_0) - \frac{1}{k}\sum_{l=1}^{k} f(x_l)]^2}_{\text{Bias}}$$

From simple observation of the equation, we can tell that as k increases, the variance will decrease as $(\frac{\sigma^2}{k})$ will tend towards 0, and the Bias will increase and tend towards $f(x_0)$ as the second term will decrease due to $\frac{1}{k}$ prefacing $\sum_{l=1}^{k} f(x_l)$

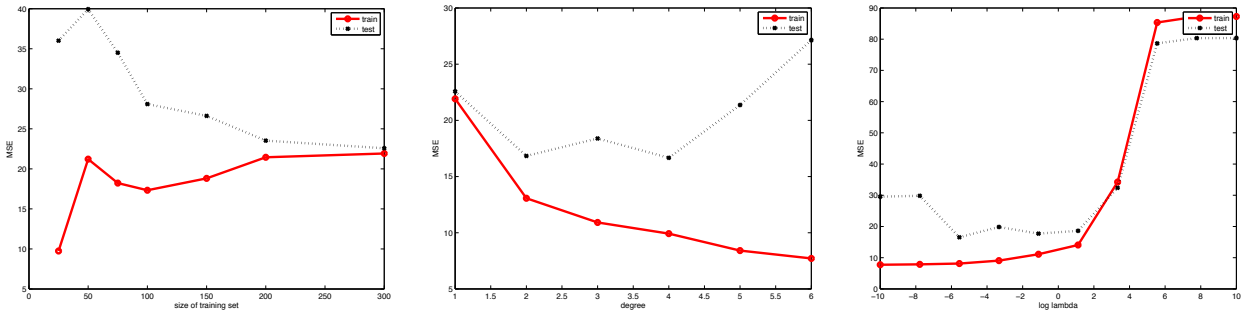
Shrinkage Methods

For vanilla linear regression model: $\min \|\mathbf{y} - \mathbf{A}\boldsymbol{\beta}\|_2^2$, we denote the solution as $\hat{\boldsymbol{\beta}}_{LS}$; for ridge regression model: $\min \|\mathbf{y} - \mathbf{A}\boldsymbol{\beta}\|_2^2 + \lambda * \|\boldsymbol{\beta}\|_2^2$, we denote the solution as $\hat{\boldsymbol{\beta}}_\lambda^{Ridge}$; for Lasso model: $\min \frac{1}{2}\|\mathbf{y} - \mathbf{A}\boldsymbol{\beta}\|_2^2 + \lambda * \|\boldsymbol{\beta}\|_1$, we denote the solution as $\hat{\boldsymbol{\beta}}_\lambda^{Lasso}$; for Subset Selection model: $\min \frac{1}{2}\|\mathbf{y} - \mathbf{A}\boldsymbol{\beta}\|_2^2 + \lambda * \|\boldsymbol{\beta}\|_0$, we denote the solution as $\hat{\boldsymbol{\beta}}_\lambda^{Subset}$, now please derive each $\hat{\boldsymbol{\beta}}$ given $\mathbf{y}, \mathbf{A}(s.t.\ \mathbf{A}^T\mathbf{A} = \mathbf{I}), \lambda$. Also, show the relationship of (each element in) $\hat{\boldsymbol{\beta}}_\lambda^{Ridge}, \hat{\boldsymbol{\beta}}_\lambda^{Lasso}, \hat{\boldsymbol{\beta}}_\lambda^{Subset}$ with (that in) $\hat{\boldsymbol{\beta}}_{LS}$ respectively. (you are encouraged to illustrate the relationship with figures appropriately.)

For $\hat{\boldsymbol{\beta}}_{LS}$ we can minimize the solution by solving for when $\frac{\partial}{\partial\beta} = 2A^T(y - A\boldsymbol{\beta}) = 0$; $A^T y = A^T A\boldsymbol{\beta}$;

$\boldsymbol{\beta} = (A^T A)^{-1} A^T y$; With $(A^T A = I)$; $\boldsymbol{\beta} = A^T y$
For Ridge Regression $\hat{\boldsymbol{\beta}}_\lambda^{Ridge}$ we can apply the same concept. $\frac{\partial}{\partial \beta} = (y - A\boldsymbol{\beta})^T (y - A\boldsymbol{\beta}) + \lambda \boldsymbol{\beta}^T \boldsymbol{\beta} = 0$;
$(A^T A + \lambda I)\boldsymbol{\beta} - A^T y = 0$; $(A^T A + \lambda I)\boldsymbol{\beta} = A^T y$; with $(A^T A = I)$; $\boldsymbol{\beta} = \lambda A^T y$

Linear Regression and its extension



(a) (b) (c)

Figure 1: MSE vs (a) training set size, (b) polynomial degree, (c) size of ridge penalty. Solid Red = training, dotted black = test.

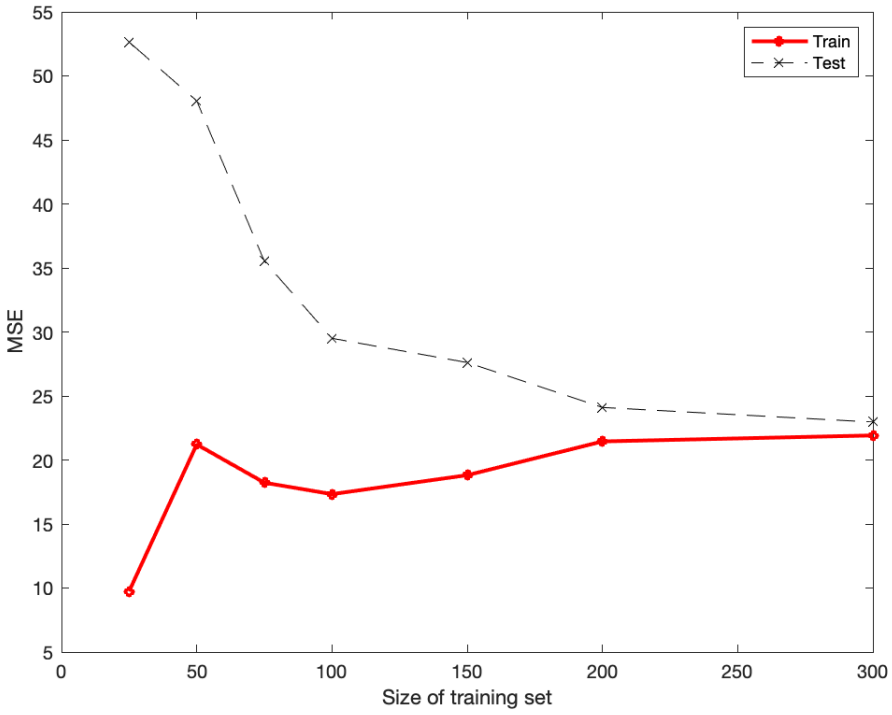
In the Boston housing dataset, there are 506 records. We will use first 13 features as inputs, x, and the 14th feature, median house price, as the output y. All features are continuous, except feature 4, which is binary. However, we will treat this like any other continuous variable.

1. Load the housing.data file. We will use the first 300 cases for training and the remaining 206 cases for testing. However, the records seem to be sorted in some kind of order. To eliminate this, we will shuffle the data before splitting into a training/test set. So we can all compare results, let use the following convention:

```
data = load('housing.data');
x = data(:, 1:13);
y = data(:,14);
[n,d] = size(x);
seed = 2; rand('state',seed); randn('state', seed);
perm = randperm(n); % remove any possible ordering fx
x = x(perm,:); y = y(perm);
Ntrain = 300;
Xtrain = x(1:Ntrain,:); ytrain = y(1:Ntrain);
Xtest = x(Ntrain+1:end,:); ytest = y(Ntrain+1:end);
```

2. Now extract the first n records of the training data, for $n \in \{25, 50, 75, 100, 150, 200, 300\}$. For each such training subset, standardize it (you may use *zscore* function in Matlab), and fit a linear regression model using least squares. (Remember to include an offset term.) Then standardize the whole test set in the same way. Compute the mean squared error on the training subset and on the whole test set. Plot MSE versus training set size. You should get a plot like Figure 1(a). Turn in your plot and code. Explain why the test error decreases as n increases, and why the train error increases as n increases. Why do the curves eventually meet? As a debugging aid, here are the regression weights I get when I train on the first 25 cases (the first term is the offset, w0): $[26.11, -0.58, 3.02, \ldots, -0.21, -0.27, -1.16]$.

   The curves eventually meet because the the projection matrix β gets more accurate with the more training data you supply. The MSE is smaller for a small sample because there is less variability. For example, if you made a projection matrix with 1 data point then it will match perfectly to that one data point. If you make a projection matrix for 306 data points then they won't all fit perfectly however the model can more accurately predict the testing data.



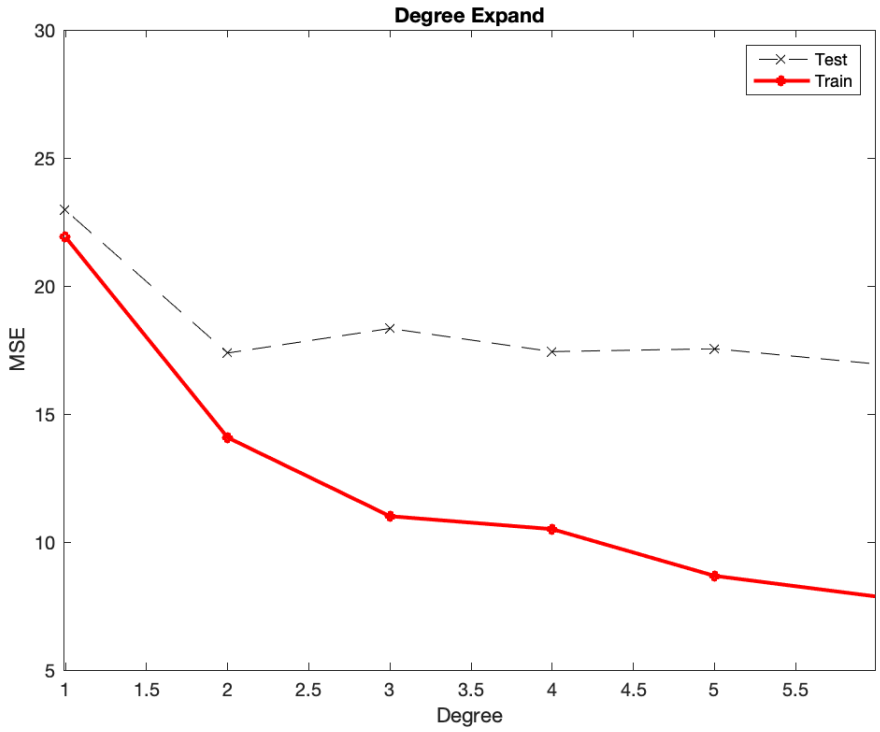
3. We will now replace the original features with an expanded set of features based on higher order terms. (We will ignore interaction terms.) For example, a quadratic expansion gives:

$$\begin{pmatrix} x_{11} & x_{12} & \ldots & x_{1d} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \ldots & x_{nd} \end{pmatrix} \rightarrow \begin{pmatrix} x_{11} & x_{12} & \ldots & x_{1d} & x_{11}^2 & x_{12}^2 & \ldots & x_{1d}^2 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \ldots & x_{nd} & x_{n1}^2 & x_{n2}^2 & \ldots & x_{nd}^2 \end{pmatrix} \tag{4}$$

   The provided function degexpand(X,deg,addOnes) will replace each row of X with all pow-

ers up to degree deg. Use this function to train (by least squares) models with degrees 1 to 6. Use all the the training data. Plot the MSE on the training and test sets vs degree. You should get a plot like Figure 1(b). Turn in your plot and code. Explain why the test error decreases and then increases with degree, and why the train error decreases with degree.

As we expand more and more features we find that we are fitting to more features therefore we are subject to overfitting. This is illustrated in the graph as the MSE of the training set continues to go down however the testing MSE stays and even rises in some occasions.

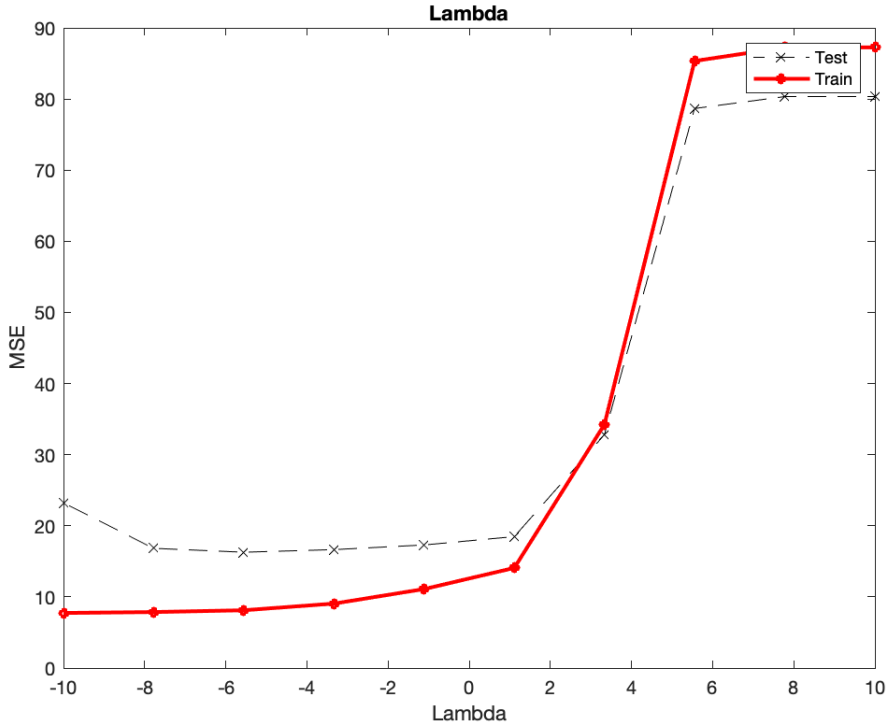


4. Now we will use ridge regression to regularize the degree 6 polynomial. Fit models using ridge regression with the following values for λ:

$$lambdas = [0\ logspace(-10, 10, 10)]$$

Use all the training data. Plot the MSE on the training and test sets vs $log_{10}(\lambda)$. You should get a plot like Figure 1(c). Turn in your plot and code. Explain why the test error goes down and then up with increasing λ, and why the train error goes up with increasing λ.

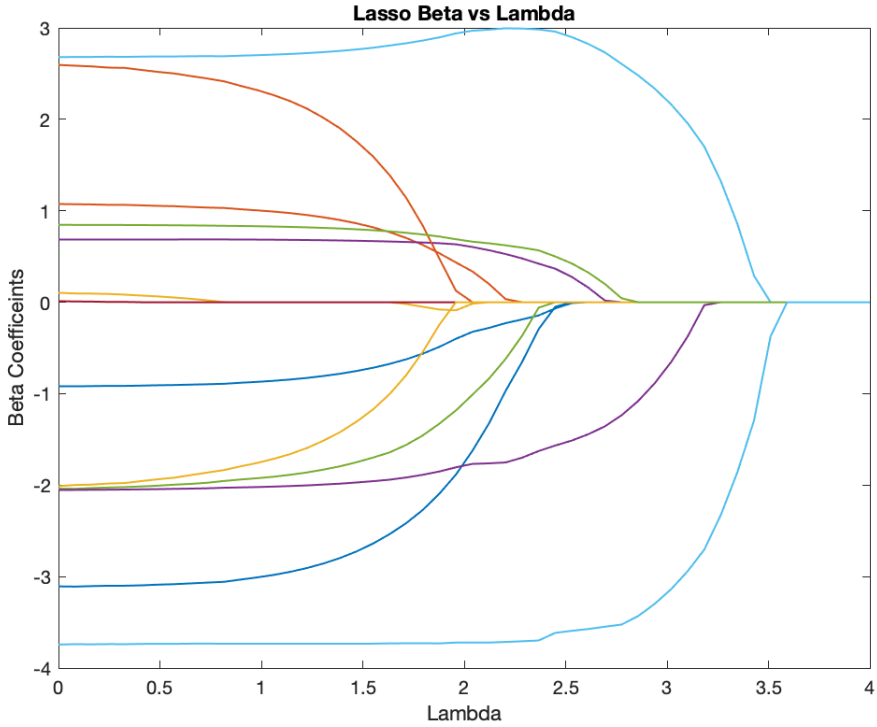
The MSE increases with lambda because we are introducing more variability. Since we can calculate ridge regression as $\hat{\beta}_{\lambda}^{Ridge} = (A^T A + \lambda I)^{-1} A^T y$, we can see that we are changing what is added to $A^T A$ by scaling λ. When $\lambda > 0$ we are linearly adding a larger variable into the equation and therefore increasing the MSE since the projection doesn't match the expected output.



5. We turn to Lasso method with objective $\frac{1}{2}\|\mathbf{X}\beta - y\|^2 + \lambda\|\beta\|_1$ where λ varies in:

$$lambdas = [logspace(-10, 10, 10)]$$

and we make use of all training samples with no feature expansion. Please plot the changes of β with λ changes.


Lasso Beta vs Lambda
Beta Coefficeints
Lambda
3
2
1
0
-1
-2
-3
-4
0
0.5
1
1.5
2
2.5
3
3.5
4