Homework Assignment # 2

Due: Wednesday, October 14, 2015, 11:59 p.m.
Total marks: 110
Student name: Srijita Das
Email id:sridas@indiana.edu

Question 1. [15 MARKS]

Suppose that the number of accidents occurring daily in a certain plant has a Poisson distribution with an unknown mean λ. Based on previous experience in similar industrial plants, suppose that our initial feelings about the possible value of λ can be expressed by an exponential distribution with parameter $\theta = \frac{1}{2}$. That is, the prior density is

$$f(\lambda) = \theta \mathrm{e}^{-\theta\lambda}$$

where $\lambda \in (0, \infty)$. Assume there are 79 accidents over the next 9 days.

(a) [5 MARKS] Determine the maximum likelihood estimate of λ.

Ans: We know the pdf of a poisson distribution is

$$p(x|\lambda) = \lambda^x e^{-\lambda}/x! \; [where \; \lambda \in \mathbb{R}^+] \tag{1}$$

Now we need the MLE estimate of λ parameter, which can me estimated as

$$\lambda_{ML} = \underset{\lambda \in (0,\infty)}{argmax}\{P(D|\lambda)\} \tag{2}$$

Now,

$$\begin{aligned} P(D|\lambda) &= P(\{x_i\}_{i=1}^n|\lambda) \\ &= \prod_{i=1}^{n} P(x_i|\lambda) \; [\because D \to iid] \\ &= \frac{\lambda^{\sum_{i=1}^n x_i}.e^{-n\lambda}}{\sum_{i=1}^n x!} \\ \therefore \ell\ell P(D|\lambda) &= \sum_{i=1}^{n} x_i \ln\lambda - n\lambda \ln e - \sum_{i=1}^{n} \ln(x!) \end{aligned}$$

To maximize $\ell\ell P(D|\lambda)$ we take the derivative.

$$\frac{\partial}{\partial\lambda} P(\ell\ell(D|\lambda)) = \sum_{i=1}^{n} x_i.\frac{1}{\lambda} - n \tag{3}$$

Now we set derivative obtained in Equation 3 to 0 and we get

$$\begin{aligned} \frac{\partial}{\partial\lambda}(\ell\ell(D|\lambda)) &= 0 \\ \therefore \sum_{i=1}^{n} x_i.\frac{1}{\lambda} &= n \end{aligned}$$

$$\lambda = \frac{\sum_{i=1}^{n} x_i}{n} = \frac{79}{9} = 8.77 \; [from \; given]$$

(b) [5 MARKS] Determine the maximum a posteriori estimate of λ.

Ans: We know the pdf of a poisson distribution is

$$p(x|\lambda) = \lambda^x e^{-\lambda}/x! \; [where \; \lambda \in \mathbb{R}^+] \tag{4}$$

Now we need the MAP estimate of λ parameter, which can me estimated as

$$\lambda_{MAP} = \underset{\lambda \in (0,\infty)}{argmax}\{P(D|\lambda).P(\lambda\} \tag{5}$$

Now,

$$\begin{aligned} P(D|\lambda) &= P(\{x_i\}_{i=1}^n|\lambda) \\ &= \prod_{i=1}^{n} P(x_i|\lambda) \; [\because D \to iid] \\ &= \frac{\lambda^{\sum_{i=1}^n x_i}.e^{-n\lambda}}{\sum_{i=1}^n x!} \\ \therefore \ell\ell P(D|\lambda) &= \sum_{i=1}^{n} x_i \ln\lambda - n\lambda \ln e - \sum_{i=1}^{n} \ln(x!) \end{aligned}$$

We now have to find the prior $P(\lambda)$ which is given by the exponential distribution $\theta e^{-\theta\lambda}$ where $\theta = \frac{1}{2}$.

$$\begin{aligned} P(\lambda) &= \frac{1}{2} e^{\frac{-\lambda}{2}} \\ \therefore \ell\ell P(\lambda) &= \frac{-\lambda}{4} \ln e \end{aligned} \tag{6}$$

$$\begin{aligned} &\ln P(\lambda|D) \propto \ln P(D|\lambda) + \ln P(\lambda) \\ &\implies \ln P(\lambda|D) = \sum_{i=1}^{n} x_i \ln\lambda - n\lambda \ln e - \sum_{i=1}^{n} \ln(x!) - \frac{\lambda}{4} \ln e \end{aligned} \tag{7}$$

To maximize $\ell\ell P(\lambda|D)$ we take the derivative.

$$\frac{\partial}{\partial\lambda} P(\ell\ell(\lambda|D)) = \sum_{i=1}^{n} x_i.\frac{1}{\lambda} - n - \frac{1}{4} \tag{8}$$

Now we set derivative obtained in Equation 8 to 0 and we get

$$\frac{\partial}{\partial\lambda} P(\ell\ell(D|\lambda)) = 0$$

$$\therefore \frac{1}{\lambda}.79 - 9 - \frac{1}{4} = 0$$
$$\lambda = \frac{79.4}{37} = 8.54 \; [from \; given]$$

(c) [5 MARKS] Look at the plots of some exponential distributions to better understand the prior chosen on λ. Imagine that now new safety measures have been put in place and you believe that the number of accidents per day should sharply decrease. For example, maybe you now believe that 4 accidents per day would be a pretty high estimate. How might you change θ to better reflect this new belief about the number of accidents?

Ans: The pdf of an exponential distribution is given by $\frac{1}{\mu}e^{\frac{-x}{\mu}}$ where $x >= 0.\mu$ here is the mean of the distribution. Since the prior given in our case is $\theta e^{\frac{-\theta}{\lambda}}$, so, $\mu = \frac{1}{\theta}$ and $x = \frac{1}{\lambda}$ by comparing the equations. In the given problem it is said that we now believe that our estimate of the prior in too high that is $f(\lambda)$ is now too high. Since, safety measures have been now put into place, so, $f(\lambda)$ should decrease. From the function we see that to reduce $f(\lambda)$, we also have to reduce the value of θ to reflect this belief about the number of accidents.

Question 2. [25 MARKS]

Let $X_1, \ldots, X_n$ be i.i.d. Gaussian random variables, each having an unknown mean θ and known variance σ_0^2.

(a) [5 MARKS] Assume θ is itself selected from a normal distribution $\mathcal{N}(\mu, \sigma^2)$ having a known mean μ and a known variance σ^2. What is the maximum a posteriori (MAP) estimate of θ?

Ans: For the above question, the assumption is that we are calculating MAP estimate of θ for a single random variable X. We know the pdf of a normal distribution is

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-1}{2\sigma^2}(x-\mu)^2} \tag{9}$$

Now we need the MAP estimate of θ parameter, which can be estimated as

$$\theta_{MAP} = \underset{\theta \in (-\infty, \infty)}{argmax} \{P(D|\theta).P(\theta)\} \tag{10}$$

Now,

$$\begin{aligned} \ell\ell P(D|\theta) &= \ln \prod_{i=1}^{n} P(x_i|\theta) \; [\because D \to iid] \\ &= \ln((\frac{1}{\sqrt{2\pi\sigma_0}})^n e^{\frac{-1}{2\sigma_0^2}\sum_{i=1}^{n}(x_i-\theta)^2}) \\ &= n \ln \frac{1}{\sqrt{2\pi}} + n \ln \frac{1}{\sigma_0} - \frac{\sum_{i=1}^{n}(x_i-\theta)^2}{2\sigma_0^2} \end{aligned}$$

Now, it is said that θ is itself chosen from a normal distribution $\mathcal{N}(\mu, \sigma^2)$. So,

$$\ell\ell P(\theta) = \ln(\frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-1}{2\sigma^2}(x_i-\mu)^2})$$

$$= \ln \frac{1}{\sqrt{2\pi}} + \ln \frac{1}{\sigma} - \frac{(\theta-\mu)^2}{2\sigma^2} \ln e \tag{11}$$

$$\ln P(\theta|D) \propto \ln P(D|\theta) + \ln P(\theta)$$
$$\implies \ln P(\theta|D) = n \ln \frac{1}{\sqrt{2\pi}} + n \ln \frac{1}{\sigma_0} - \frac{\sum_{i=1}^{n}(x_i-\theta)^2}{2\sigma_0^2} + \ln \frac{1}{\sqrt{2\pi}} + \ln \frac{1}{\sigma} - \frac{(\theta-\mu)^2}{2\sigma^2} \ln e \tag{12}$$

To maximize $\ell\ell P(\theta|D)$ we take the derivative.

$$\frac{\partial}{\partial\theta} P(\ell\ell(\theta|D)) = \frac{\sum_{i=1}^{n}(x_i-\theta)}{\sigma_o^2} - \frac{\theta-\mu}{\sigma^2} \tag{13}$$

Now we set derivative obtained in Equation 13 to 0 and we get

$$\begin{aligned} \frac{\partial}{\partial\theta} P(\ell\ell(\theta|D)) &= 0 \\ \therefore \frac{\sum_{i=1}^{n}(x_i-\theta)}{\sigma_o^2} - \frac{\theta-\mu}{\sigma^2} &= 0 \\ \theta_{MAP} &= \frac{\sigma^2 \sum_{i=1}^{n} x_i + \mu\sigma_0}{\sigma_0 + n\sigma^2} \end{aligned} \tag{14}$$

(b) [10 MARKS] Assume θ is itself selected from a Laplace distribution $\mathcal{L}(\mu, b)$ having a known mean (location) μ and a known scale (diversity) b. Recall that the pdf for a Laplace distribution is

$$p(x) = \frac{1}{2b} \exp\left(\frac{-|x-\mu|}{b}\right)$$

For simplicity, assume $\mu = 0$. What is the maximum a posteriori estimate of θ? If you cannot find a closed form solution, explain how you would use an iterative approach to obtain the solution.

Ans: For the above question, the assumption is that we are calculating MAP estimate of θ for a single random variable X.
We know that the pdf of normal distribution is as below:

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{-1}{2\sigma^2}(x-\mu)^2} \tag{15}$$

Also, the pdf of Laplace distribution is given from where θ is selected.
Now we need the MAP estimate of θ parameter, which can be estimated as

$$\theta_{MAP} = \underset{\theta\in(-\infty,\infty)}{argmax} \{P(D|\theta).P(\theta)\} \tag{16}$$

Now,

$$\ell\ell P(D|\theta) = \ln \prod_{i=1}^{n} P(x_i|\theta) \ [\because D \to iid]$$

$$
\begin{aligned}
&= \ln((\frac{1}{\sqrt{2\pi\sigma_0}})^n e^{\frac{-1}{2\sigma_0^2}\sum_{i=1}^n (x_i-\theta)^2}) \\
&= n\ln\frac{1}{\sqrt{2\pi}} + n\ln\frac{1}{\sigma_0} - \frac{\sum_{i=1}^n (x_i-\theta)^2}{2\sigma_0^2}
\end{aligned}
$$

Now, it is said that θ is itself chosen from a Laplace distribution $\mathcal{L}(\mu, b)$ It is also told to assume the mean $\mu = 0$. So,

$$
\begin{aligned}
\ell\ell P(\theta) &= \ln(\frac{1}{2b}\exp\left(\frac{-|\theta-0|}{b}\right)) \\
&= \ln\frac{1}{2b} + \frac{|\theta|}{b}\ln e
\end{aligned}
$$

$$\ln P(\theta|D) \propto \ln P(D|\theta) + \ln P(\theta)$$

$$\ln P(\theta|D) = n\ln\frac{1}{\sqrt{2\pi}} + n\ln\frac{1}{\sigma_0} - \frac{\sum_{i=1}^n (x_i-\theta)^2}{2\sigma_0^2} + \ln\frac{1}{2b} + \frac{|\theta|}{b}\ln e \quad (17)$$

(18)

To maximize $\ell\ell P(\theta|D)$ we take the derivative.

$$\frac{\partial}{\partial\theta} P(\ell\ell(\theta|D)) = \frac{\sum_{i=1}^n (x_i-\theta)}{\sigma_o^2} - \frac{1}{2b^2}\frac{\partial}{\partial\theta}|\theta| \quad (19)$$

From the above gradient expression, we see that θ is a piece wise function and has a piece wise derivative. It will have a derivative of 1 for $x > 0$ and a derivative of -1 for $x < 0$. Since, we do not get a single value of θ from the above expression, hence, the above expression does not have a closed form solution.

Iterative approach to the solution

We can use a gradient descent approach to solve this problem. We can start with some random value of $\theta > 1$ and take the derivative of θ in Equation 19 as 1.Then we will get the gradient of the MAP function. Our objective is to maximize this function. We will keep on decresing θ by small amounts and see if the value of the function in Equation 17 increases or not. Our objective is to see when the gradient of this function becomes 0 and the function reaches it's max value. We repeat this process till the time two consecutive θ values becomes almost equal and the gradient of the function becomes 0. Similarly, we can repeat the same process for a value of $\theta < 0$ and repeat the same process till the time the function converges. So, as per my intuition, there might be 2 values of θ for which the function converges.

(c) [10 MARKS] Now assume that we have **multivariate** i.i.d. Gaussian random variables, $\mathbf{X}_1, \ldots, \mathbf{X}_n$ with each $\mathbf{X}_i \sim \mathcal{N}(\boldsymbol{\theta}, \boldsymbol{\Sigma}_0)$ for some unknown mean $\boldsymbol{\theta} \in \mathbb{R}^d$ and known $\boldsymbol{\Sigma}_0 = \mathbf{I} \in \mathbb{R}^{d\times d}$, where $\mathbf{I}$ is the identity matrix. Assume $\boldsymbol{\theta} \in \mathbb{R}^d$ is selected from a zero-mean multivariate Gaussian $\mathcal{N}(\boldsymbol{\mu} = \mathbf{0}, \boldsymbol{\Sigma} = \sigma^2\mathbf{I})$ and a known variance parameter σ^2 on the diagonal. What is the MAP estimate of $\boldsymbol{\theta}$?

Ans: For the problem it is given that $\mathbf{X}_1, \ldots, \mathbf{X}_n$ are multivariate Gaussian iid's with each $\mathbf{X}_i \sim \mathcal{N}(\boldsymbol{\theta}, \boldsymbol{\Sigma}_0)$.
The pdf of multivariate gaussian variables is given by

$$p(x, \theta, \Sigma) = \frac{1}{\sqrt{2\pi|\Sigma|}} e^{\frac{-1}{2}(x-\theta)^T \Sigma^{-1}(x-\theta)} \tag{20}$$

Now we need the MAP estimate of θ parameter, which can be estimated as

$$\theta_{MAP} = \underset{\theta \in (-\infty, \infty)}{argmax} \{P(D|\theta).P(\theta)\} \tag{21}$$

Now,

$$\begin{aligned}
\ell\ell P(D|\theta) &= \ln \prod_{i=1}^{n} P(X_i|\theta) \ [\because D \to iid] \\
&= \ln \prod_{i=1}^{n} (\frac{1}{\sqrt{2\pi|\Sigma_0|}} e^{\frac{-1}{2}(X-\theta)^T \Sigma_0^{-1}(X-\theta)} \\
&= \ln((\frac{1}{\sqrt{2\pi|\Sigma_0|}})^n e^{\frac{-1}{2}\sum_{i=1}^{n}(X-\theta)^T \Sigma_0^{-1}(X-\theta)}) \\
&= n \ln(\frac{1}{\sqrt{2\pi}}) + n \ln \frac{1}{\sqrt{|\Sigma_0|}} - \frac{1}{2}\sum_{i=1}^{n}(X-\theta)^T \Sigma_0^{-1}(X-\theta) \ln e
\end{aligned}$$

Now, it is said that θ is itself chosen from zero mean multivariate Gaussian $\mathcal{N}(\boldsymbol{\mu} = \mathbf{0}, \boldsymbol{\Sigma} = \sigma^2\mathbf{I})$ and a known variance parameter σ^2 on the diagonal.It is given that θ has 0 mean. So,

$$\begin{aligned}
\ell\ell P(\theta) &= \ln(\frac{1}{\sqrt{2\pi|\Sigma}} e^{\frac{-1}{2}\theta^T \Sigma^{-1}\theta}) \\
&= \ln \frac{1}{\sqrt{2\pi}} + \ln \frac{1}{|\Sigma|} - \frac{1}{2}\theta^T \Sigma^{-1}\theta \ln e
\end{aligned} \tag{22}$$

$$\ln P(\theta|D) \propto \ln P(D|\theta) + \ln P(\theta)$$

$$\begin{aligned}
\implies \ln P(\theta|D) &= n \ln(\frac{1}{\sqrt{2\pi}}) + n \ln \frac{1}{\sqrt{|\Sigma_0|}} - \frac{1}{2}\sum_{i=1}^{n}(X-\theta)^T \Sigma_0^{-1}(X-\theta) \ln e \\
&\quad + \ln \frac{1}{\sqrt{2\pi}} + \ln \frac{1}{|\Sigma|} - \frac{1}{2}\theta^T \Sigma^{-1}\theta \ln e
\end{aligned}$$

To maximize $\ell\ell P(\theta|D)$ we take the derivative.

$$\frac{\partial}{\partial\theta} P(\ell\ell(\theta|D)) = \frac{-1}{2}\sum_{i=1}^{n} \frac{\partial}{\partial\theta}(X-\theta)^T \Sigma_0^{-1}(X-\theta) - \frac{1}{2}\frac{\partial}{\partial\theta}(\theta)^T \Sigma^{-1}(\theta) \tag{23}$$

$$= \frac{1}{2}\sum_{i=1}^{n} 2(X-\theta) - \frac{\sigma^2}{2} 2\theta \ [\because \Sigma_0^{-1} = I \ and \ \Sigma = \sigma^2 I] \tag{24}$$

(25)

Now we set derivative obtained in Equation 23 to 0 and we get

$$\begin{aligned} \frac{\partial}{\partial\theta}P(\ell\ell(\theta|D)) &= 0 \\ \therefore \sum_{i=1}^{n}(X-\theta)-\sigma^2\theta &= 0 \\ \implies \sum_{i=1}^{n}X-n\theta-\sigma^2\theta &= 0 \\ \theta_{MAP} &= \frac{\sum_{i=1}^{n}X_i}{n+\sigma^2} \end{aligned}$$

(26)

Question 3. [10 MARKS]

Program the two iterative algorithms, gradient descent and Newton's method, to find the minimum of a function of two-dimensional variable $\mathbf{x} = (x_1, x_2)$

$$f(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$$

Set the step length to $\eta = 1$ and try two different starting points: $x^{(0)} = (1.2, 1.2)$ and a more difficult $x^{(0)} = (-1.2, 1)$. Reduce the step size to some $\eta < 1$ and repeat the minimization. Show all your work and discuss the optimization processes you tested.

Ans: I implemented both the gradient descent and the Newton Raphson method for optimizing the function mentioned above and below are my observations. I am also attaching my code along with the submission.(2 codes).

Gradient Descent:

When I tried implementing the Gradient Descent method with $\eta = 1$, x_1 and x_2 reaches some really small value and the Python code throws "Overflow Error". So, I could not get the results with $\eta = 1$.I tried implementing Gradient descent with $\eta = 0.001$ and error threshold=0.000001. Also, since, the values becomes extremely small while the function approaches convergence, so increasing the precision in the code helped.

With a starting point of $x^{(0)} = (1.2, 1.2)$, my solution with precision of 25 from gradient descent was

$x_1 = 1.00000000124883825029588, x_2 = 1.00000000124883825029588$

With a starting point of $x^{(0)} = (-1.2, 1)$, my solution with precision of 25 from gradient descent was

$x_1 = 0.99999999875101133818189 99, x_2 = 0.99999999749702472728732 44$

However, it took a little time to converge to the solution due to small learning rate. From the first starting point, it took me 46288 iterations to reach the solution and from the second starting point, it took me 49092 iterations to reach the solution.

Newton-Raphson method With the Newton Raphson method, it was a lot easier to converge to the solution. It also took lesser time since there was no step size involved in Newton-Raphson.

With a starting point of $x^{(0)} = (1.2, 1.2)$, my solution with the Newton-Raphson method was $x_1 = 1.0000, x_2 = 1.0000$ With a starting point of $x^{(0)} = (-1.2, 1)$, my solution with the Newton-Raphson method was
$x_1 = 1.0000, x_2 = 1.0000$
From the first starting point, convergence was reached in 6 iterations and from the second starting point, convergence was reached in 8 iterations. This method also gives better result if I increase the precision in the Python code.

Question 4. [60 MARKS]

In this question, you will implement linear regression and Poisson regression. An initial script in python has been given to you, called `script_classify.py`, and associated python files. You will be running on a blog dataset, with 230 features and 60,000 samples. Note that the first 50 features are constants for each sample, giving information about the features, and so are removed when the data is loaded. Baseline algorithms, including mean and random predictions, are used to serve as sanity checks. We should be able to outperform random predictions, and the mean value of the target in the training set. We will be examining some of the practical aspects of implementing regression. As a suggestion, you should start or complete Question 4 before doing this question.

(a) [5 MARKS] The main linear regression class is `FSLinearRegression`. The FS stands for FeatureSelect. The provided implementation has subselected features and then simply explicitly solved for $\mathbf{w} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$. Increase the number of selected features (including all the way to including all the feature). What do you find? How can this be remedied?

Ans: You need to enter the parameter "split" with the code to run it on a training set of the first 20000 samples and testset of the next 5000 samples.

No. of features	Error(LR)	Error(Random)	Error(Mean)
3	431.633661535	51109.9404082	797.047211111
10	918.196531642	45166.3452591	237.828044444

For feature size of 100, 200 and 230, I tried running the code but the code did not compute the optimal w because for such a large number of features, $(X^T X)$ becomes a singular matrix and hence, the inverse of this matrix doesn't exist and hence, the coefficients are not calculated.Generally, as the number of features increases, the accuracy of Linear regression decreases as we can see that l2 squared error with feature size=3 is 431.633661 whereas l2 squared error with feature size=10 is 918.19653.Note that the results were produced when I ran the code with training size=300 and test size=100

This can be remedied by adding regularisation to the regression. If the features are linearly dependent, then $(X^T X)$ cannot be inverted but after adding the regularisation parameter to this matrix, the matrix can be inverted.

(b) [10 MARKS] Now implement Ridge Regression, where a ridge regularizer $\lambda \|\mathbf{w}\|_2^2$ is added to the optimization. Run this algorithm on all the features. How does the result differ from (a)? Discuss results for different regularization parameter λ values. Modify the code to report error average over multiple splits of the data (at least 10 splits).

Ans: I implemented Ridge regression in the code in the module in the module algorithms.py. I added an extra λI facttor to the $X^T X$ matrix and the code now works for all the 230 features.I

ran the code for various values of λ starting from 1 and incrementing it by 1 every time to see the accuracy. When I run Ridge regression on a training set of 300 and test set of 100 randomly chosen from the main data, with λ=1, I get squared error of 702.616350094. When I run the same training and test set for mean prediction, I get a squared error of 1854.48394444 and I get a squared error of 11478.9053253 for random prediction. FS Linear regression on the other hand gives a squared error of 12334.6214524 with 10 features selected. So, in either case, we were able to reduce the squared error or the average error with Ridge regression than Mean, random or Linear regression.

Next, I ran the ridge regression for various values of λ starting from 1 to 100 on the training set of nearly 45000 data points and test set of 5000 data points which is nearly the same size of training and test dataset we will be using for 10- fold cross validation. I wanted to see what id the value od λ that works best and then use that value of λ to report the accuracy of 10-fold cross validation with Ridge regression.In the below table I have jotted down some values of λ and their accuracy.

λ	Error(Ridge regression
1	873.621010124
10	1082.24899523
20	1082.20972816
30	1082.21337103
40	1082.22904281
50	1082.24849891
60	1082.26886307
70	1082.28901101
80	1082.28901101
90	1082.3271571
100	1082.34495971

What I see from the above data is that accuracy is least for λ=1. As, I increase λ by 10, the accuracy start increasing though by minute amounts. I also decreased λ by 0.5 from 1 and saw that the squared error was again increasing. Hence, I think that λ=1 might be a good choice to consider and go forward with the 10-fold cross validation with λ=1.

10-Fold Cross Validation on the Blog dataset:
When I do a 10-Fold cross validation on the dataset with Ridge regression, the average error over 10- folds comes out to be 1168.1908625389824 with the parameter λ=1. This estimate of error is worse than when I chose a training set of 45000 data points and test set of 5000 data points. The one observation that I see from the 10 fold cross validation is that it gives pretty descent squared error for all the folds except one. When this one fold is used as test and the other as training, it gives a huge error of 7681.90672002 which causes the average error of 10-fold cross validation of Ridge regression to go up and perform badly than the Ridge regression without 10-Fold cross validation.

(c) [10 MARKS] Imagine that the dataset size continues to grow, which causes the matrices to become large $\boldsymbol{X} \in \mathbb{R}^{n \times d}$ for n samples and d features. One option is to go back to subselecting features. In `FSLinearRegression`, add an approach to select 10 features and explain your choice. How does accuracy change compared to the original `FSLinearRegression` and `RidgeRegression`? What happens to the accuracy of the solution?

Ans: I tried subselecting features in Python by calculating the covariance matrix of all possible pairs of features using the numpy.cov function. But somehow it did not work and the Python code was running out of memory. So, I imported the dataset in R and wrote the below code to calculate the correlation matrix and remove all those features whose correlation between themselves is more than 0.75.

```
This R code was used to remove correlated features
library(mlbench)
library(caret)
data<-read.csv("dataBlog_train_norm.csv",header=FALSE)
data1<-data[,51:277]
data2<-data[,279:280]
dataf<-cbind(data1,data2)
cmat <- cor(data)
hcor <- findCorrelation(cmat, cutoff=0.75)
print hcor
```

I normalized the dataset before importing it into R. I also did normalization of the dataset in R. I have attached the normalized dataset with the zip.For normalization, I did (value-min)/(max-min) for each set of features. The column 278 of the dataset had all 0 values and hence was adding no information to the predictor. Hence, I deleted this column.R has this ready made package caret which calculates the correlation matrix as well as gives all those features that are highly correlated.I got about 25 features that are highly correlated and removed them from the feature set. For the remaining features, I now chose 10 random features from the rest over features. The various results of running Linear and Ridge Regression with first 10 features as well as the feature set selected by me is jotted down in the below table.The below results were run on the first 45000 rows of the dataset as training set and next 5000 rows of the dataset as the test set.

Type	Accuracy	Feature size	procedure	λ
LR	367.121820943	10(my selection)	split	
LR	682.483567663	10(first)	split	
RR	322.599345603	230	split	0.1
RR	367.115218961	10(my selection)	split	0.1
RR	1458.37136939346	10(my selection)	cross-val	1
RR	300.554654373	10(first)	split	0.1
RR	1168.78913566584	230	cross-val	0.1
Random	668123.13692	230	split	
Mean	343.470500383	230	split	

In the above table, cross-val stands for the fact that the results were 10-fold cross validated and split means that the results were just reported over one split of data. This may not be the true estimate of error. After removing the collinear features and subselecting the features, Linear Regression does a lot better with my selected feature set. With sub selected features according to the procedure followed by me, Ridge regression does almost the same performance as that with the first 10 features. However, both the Linear and Ridge regression perform almost close to the Mean regressor and much better than the random regressor.

(d) [15 MARKS] Now imagine that your dataset has gotten even larger, to the point where dropping features is not enough. Instead of removing features, implement a stochastic optimization approach to obtaining the linear regression solution (see Section 4.5.3). Explain your implementation choices.

Ans: Before implementing Stochastic optimization, I normalized the data by importing it in R and using the same technique as mentioned in the earlier answer. Without normalizing, when I tried to implement Stochastic Optimization, it was giving Singular matrix error as the determinant of the matrix was becoming zero. I normalized the data to convert the features into a scale from 0 to 1. Then ran the stochastic gradient descent on a train size of the first 45000 data points and testsize of 5000. I tried running it with various learning rates and error thresholds. The best accuracy I got out of this was 1710290.28554. This was with a learning rate of 0.00001 and convergence criteria of error threshold as 0.00000001. May be reducing learning rates with times will give better result but I did not try that out. Also, I noticed that my Loss function was pretty unstable in this case and was fluctuating a lot, sometimes increasing its value and sometimes decreasing its value.I am not very sure why this was happening. The initial weight vector has been set to all ones randomly.

(e) [20 MARKS] Next you notice that the target is always positive, with many zeros and small numbers and a few large values. These target values look a little like they could come from a Poisson distribution. You recall that generalized linear models allow non-linear transfers on the linear solution, and that conveniently the Poisson distribution happens to have a nice link function. Implement Poisson regression on this data. Hint: using an exponential transfer can cause numerical instabilities. For training, you might consider scaling down the target so that there are not such large values. Moreover, the approach can be somewhat sensitive to features being collinear. You could consider subselecting some number of features once again. Explain the choices you use. You should be able to obtain better performance than linear regression.

Ans:I tried implementing Poisson regression but somehow it is not converging. I normalized the data in the dataset before running Poisson Regression.It did not work on all the 230 features in the dataset and throws an error message that the inverse matrix (X^TCX) is not invertible. I tried implementing Poisson regression with a training size of first 20000 and a test-size of 5000 on the first 10 features of the dataset and it worked one time, it return a huge error of 1262625.51007. When I try running Poisson Regression on a train set of 45000 and testset of 5000, it throws me memory error when evaluating the 45000 x 45000 diagonal Cost Matrix. Next, I tried running it on a testsize of 20000 and testsize of 5000 again and it throws me the error that the determinant of the matrix whose inverse is to be found becomes zero. My objective function(Error function) is also unstable here and is either increasing or decreasing.Next, for the same training set, I tried to add a regularisation parameter $\lambda = 1$ to it and with 10 features taken. This also did not work and the matrix becomes singular whose inverse is to be taken. I also tried running the code with the output values normalized within the range 0 and 1 but that doesn't seem to work either.

References

[1] http://machinelearningmastery.com/feature-selection-with-the-caret-r-package/

Homework policies:

Your assignment must be typed; for example, in Latex, Microsoft Word, Lyx, etc. Images may be scanned and inserted into the document if it is too complicated to draw them properly. Submit a

single pdf document or, if you are attaching your code, submit your code together with the typed (single) document as one .zip file. All code (if applicable) should be turned in when you submit your assignment. Use Matlab, Python, R, Java or C. Policy for late submission assignments: Unless there are legitimate circumstances, late assignments will be accepted up to 5 days after the due date and graded using the following rule:

on time: your score 1

1 day late: your score 0.9

2 days late: your score 0.7

3 days late: your score 0.5

4 days late: your score 0.3

5 days late: your score 0.1

For example, this means that if you submit 3 days late and get 80 points for your answers, your total number of points will be $80 \times 0.5 = 40$ points. All assignments are individual, except when collaboration is explicitly allowed. All the sources used for problem solution must be acknowledged, e.g. web sites, books, research papers, personal communication with people, etc. Academic honesty is taken seriously; for detailed information see Indiana University Code of Student Rights, Responsibilities, and Conduct.

Good luck!