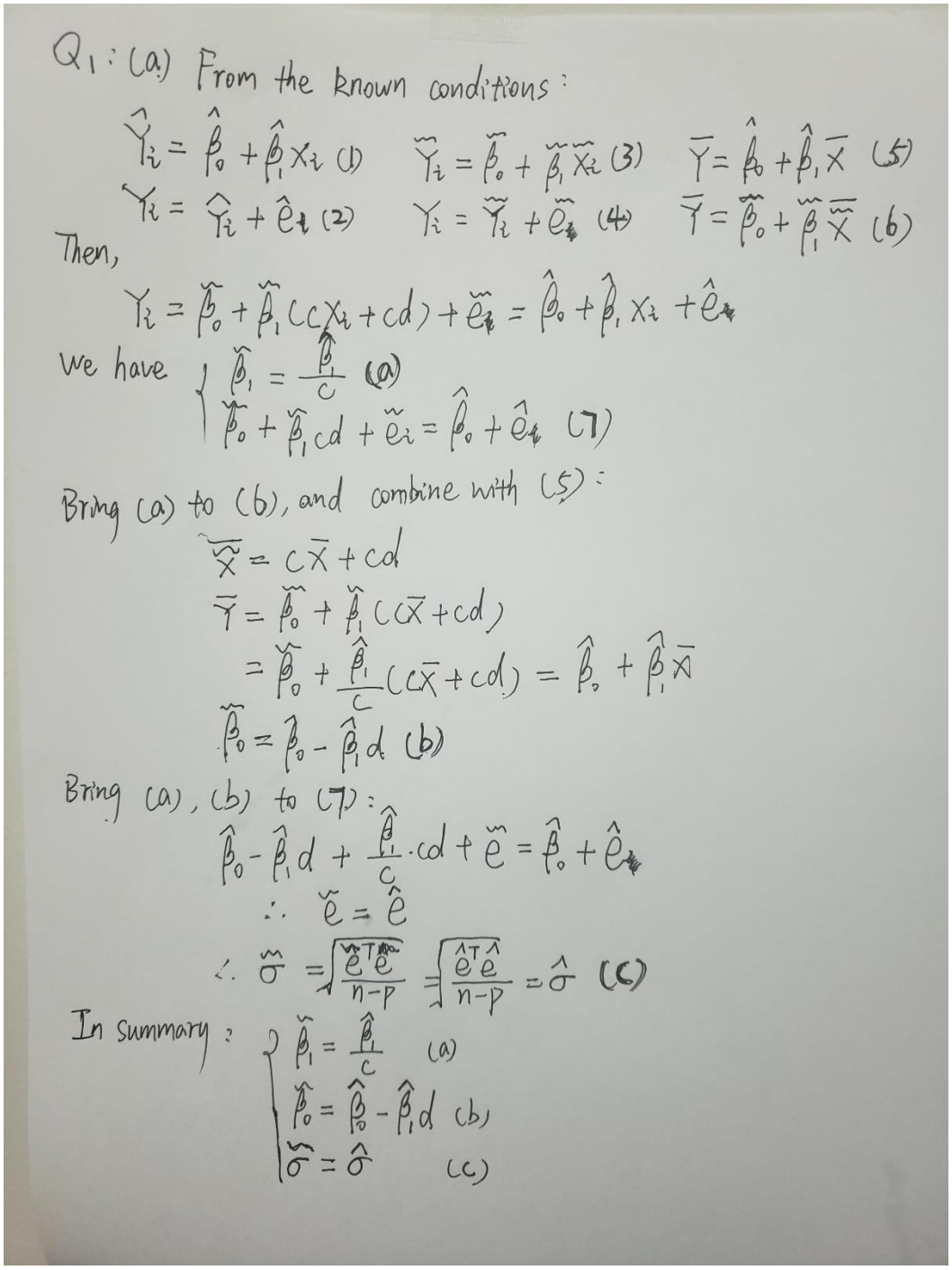
COMP9417 Homework1

Zid: z5239803

Name: Zhengyang Jia

Q1:



Q_1: (a) From the known conditions:

$\hat{Y}_i = \hat{\beta}_0 + \hat{\beta}_1 X_i$ (1) $\quad \tilde{Y}_i = \tilde{\beta}_0 + \tilde{\beta}_1 \tilde{X}_i$ (3) $\quad \bar{Y} = \hat{\beta}_0 + \hat{\beta}_1 \bar{X}$ (5)

$Y_i = \hat{Y}_i + \hat{e}_i$ (2) $\quad Y_i = \tilde{Y}_i + \tilde{e}_i$ (4) $\quad \bar{Y} = \tilde{\beta}_0 + \tilde{\beta}_1 \bar{\tilde{X}}$ (6)

Then,

$$Y_i = \tilde{\beta}_0 + \tilde{\beta}_1 (cX_i + cd) + \tilde{e}_i = \hat{\beta}_0 + \hat{\beta}_1 X_i + \hat{e}_i$$

We have $\begin{cases} \tilde{\beta}_1 = \dfrac{\hat{\beta}_1}{c} \quad (a) \\ \tilde{\beta}_0 + \tilde{\beta}_1 cd + \tilde{e}_i = \hat{\beta}_0 + \hat{e}_i \quad (7) \end{cases}$

Bring (a) to (b), and combine with (5):

$$\bar{\tilde{X}} = c\bar{X} + cd$$

$$\bar{\tilde{Y}} = \tilde{\beta}_0 + \tilde{\beta}_1 (c\bar{X} + cd)$$

$$= \tilde{\beta}_0 + \frac{\hat{\beta}_1}{c}(c\bar{X} + cd) = \hat{\beta}_0 + \hat{\beta}_1 \bar{X}$$

$$\tilde{\beta}_0 = \hat{\beta}_0 - \hat{\beta}_1 d \quad (b)$$

Bring (a), (b) to (7):

$$\hat{\beta}_0 - \hat{\beta}_1 d + \frac{\hat{\beta}_1}{c} \cdot cd + \tilde{e} = \hat{\beta}_0 + \hat{e}$$

$$\therefore \tilde{e} = \hat{e}$$

$$\therefore \tilde{\sigma} = \sqrt{\frac{\tilde{e}^T \tilde{e}}{n-p}} = \sqrt{\frac{\hat{e}^T \hat{e}}{n-p}} = \hat{\sigma} \quad (c)$$

In summary:

$$\begin{cases} \tilde{\beta}_1 = \dfrac{\hat{\beta}_1}{c} \quad (a) \\ \tilde{\beta}_0 = \hat{\beta}_0 - \hat{\beta}_1 d \quad (b) \\ \tilde{\sigma} = \hat{\sigma} \quad (c) \end{cases}$$

Q1: (b) Suppose there are $(n+m)$ patients in total, n of them receive treatment, while n of them don't.

Due to the presumed condition in the question,

We have $\bar{Y} = \hat{\beta}_0 + \hat{\beta}_1 \bar{X}$ (1) and

$$\begin{cases} \bar{X} = \frac{n}{n+m} \cdot 1 + \frac{m}{n+m} \cdot 0 = \frac{n}{n+m} & (2) \\ \bar{Y} = \frac{n\bar{Y}_T + m\bar{Y}_P}{n+m} & (3) \end{cases}$$

Combine (1), (2) and (3)

We have $\hat{\beta}_0 + \frac{n}{n+m}\hat{\beta}_1 = \frac{n}{n+m}\bar{Y}_T + \frac{m}{n+m}\bar{Y}_P$

Because the equation satisfy when n, m are non-negative int. (not both 0)

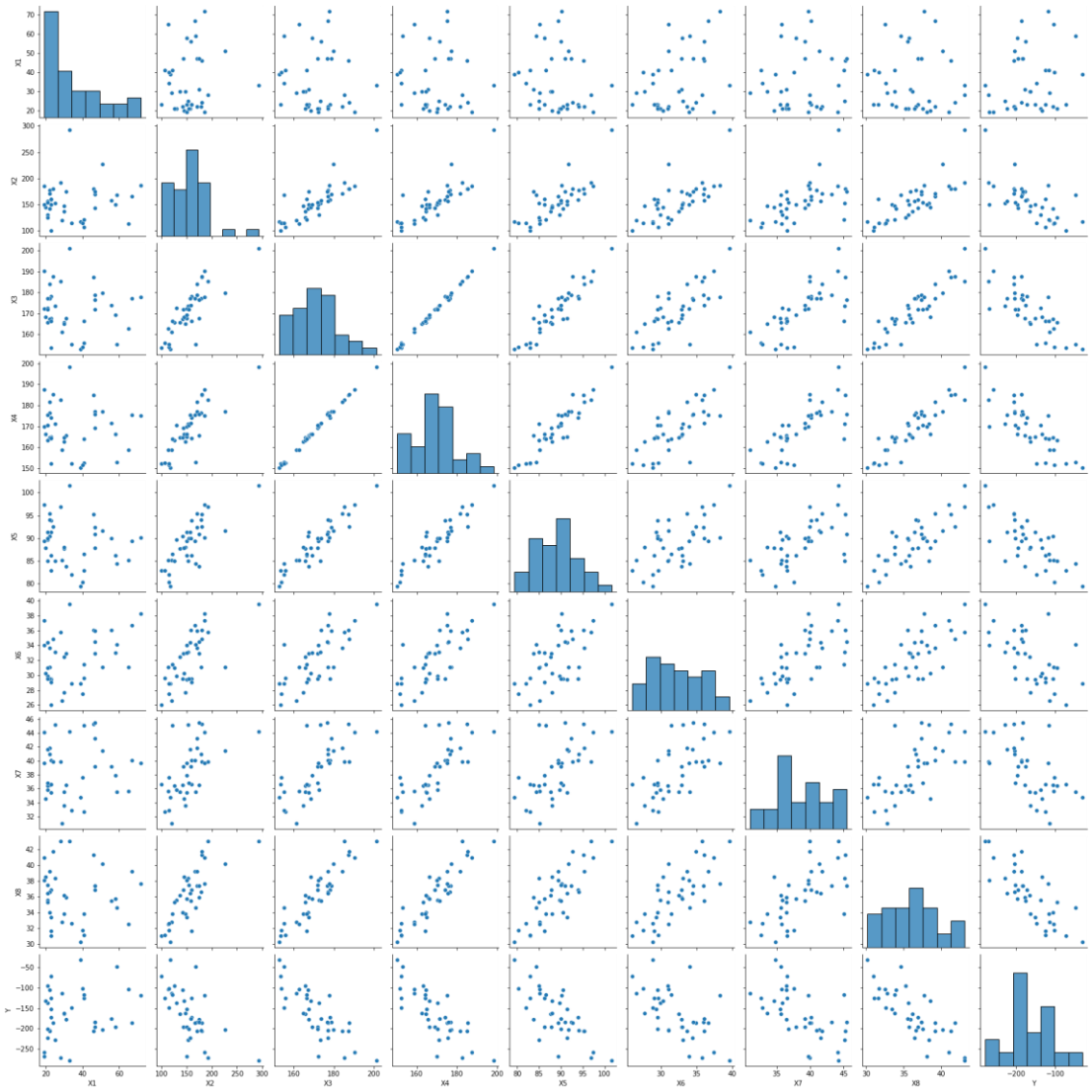
take $n=0$, the we got $\hat{\beta}_0 = \bar{Y}_P$

then take $m=0$, we got $\hat{\beta}_1 = \bar{Y}_T - \bar{Y}_P$

In conclusion $\begin{cases} \hat{\beta}_0 = \bar{Y}_P \\ \hat{\beta}_1 = \bar{Y}_T - \bar{Y}_P \end{cases}$

Q2:

(a)



The main diagonal is the histogram distribution of the corresponding univariate, while other non-diagonal sub-images are the scatter plot. The first 8 rows and columns are the 8 features. The last row and column represent Y. We notice that feature the correlation image between X3 and X4, X5, X8 can fit linear positive correlation. In particular, X3 and X4 are fit the positive correlation precisely.

(b)

```
[14] file=pd.read_csv('/content/drive/MyDrive/data.csv')
     file_rescale = preprocessing.scale(file)
```

```
[15] def Calculate_SquareSum(df,num):
         s_square = 0
         for i in file_rescale:
             item = i[num-1]
             s_square = s_square + item*item
         print("The Sum of squared observations of feature X%d is " % num, end="");
         print("{:.2f}".format(s_square))
         return
```

```
[16] Calculate_SquareSum(file_rescale,1)
     Calculate_SquareSum(file_rescale,2)
     Calculate_SquareSum(file_rescale,3)
     Calculate_SquareSum(file_rescale,4)
     Calculate_SquareSum(file_rescale,5)
     Calculate_SquareSum(file_rescale,6)
     Calculate_SquareSum(file_rescale,7)
     Calculate_SquareSum(file_rescale,8)
```

```
The Sum of squared observations of feature X1 is 38.00
The Sum of squared observations of feature X2 is 38.00
The Sum of squared observations of feature X3 is 38.00
The Sum of squared observations of feature X4 is 38.00
The Sum of squared observations of feature X5 is 38.00
The Sum of squared observations of feature X6 is 38.00
The Sum of squared observations of feature X7 is 38.00
The Sum of squared observations of feature X8 is 38.00
```

After rescaling, the sum of squares of each feature is 38.

(c)

```
[47] data = file_rescale
     X, y = data[:, :-1], data[:, -1]
```

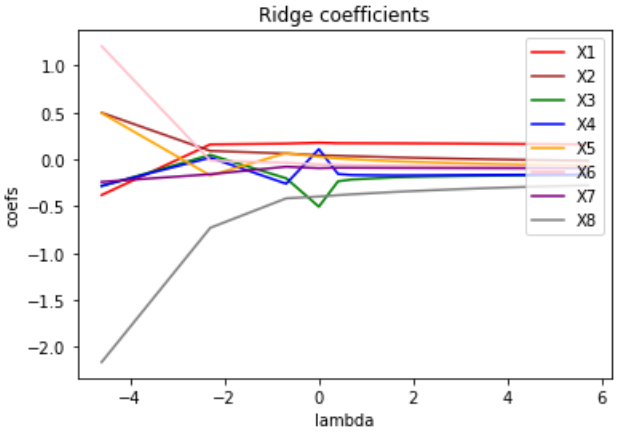
```
[48] Xt = X[1:]
     yt = y[1:]
```

```
[49] lam = ([np.log(0.01), np.log(0.1), np.log(0.5), np.log(1), np.log(1.5), np.log(2), np.log(5),
             np.log(10), np.log(20), np.log(30), np.log(50), np.log(100), np.log(200), np.log(300)])
```

```
[50] coefs = []
     for item in lam:
         ridge = linear_model.Ridge(alpha=item, fit_intercept=True)
         ridge.fit(Xt, yt)
         coefs.append(ridge.coef_)
```

```
[51] Coefs = np.array(coefs)
     y1=Coefs[:,0]
     y2=Coefs[:,1]
     y3=Coefs[:,2]
     y4=Coefs[:,3]
     y5=Coefs[:,4]
     y6=Coefs[:,5]
     y7=Coefs[:,6]
     y8=Coefs[:,7]
```

```
[52] ax = plt.gca()
     Colr = ['red', 'brown', 'green', 'blue', 'orange', 'pink', 'purple', 'grey']

     ax.plot(lam,y1,color= Colr[0])
     ax.plot(lam,y2,color= Colr[1])
     ax.plot(lam,y3,color= Colr[2])
     ax.plot(lam,y4,color= Colr[3])
     ax.plot(lam,y5,color= Colr[4])
     ax.plot(lam,y6,color= Colr[5])
     ax.plot(lam,y7,color= Colr[6])
     ax.plot(lam,y8,color= Colr[7])


     plt.xlabel('lambda')
     plt.ylabel('coefs')
     plt.title('Ridge coefficients')
     plt.axis('tight')
     plt.legend(('X1', 'X2', 'X3', 'X4', 'X5', 'X6', 'X7', 'X8'),loc='upper right')
     plt.show()
```



The coefficient of X3 and X4 first share the same figure. Then X3 and X4 splits up, but they end up approaching zero. X5 figure is a fluctuant wave, and if we draw a line Xm as the mean of X3 and X4, we can notice X5 is symmetry to Xm.

(d)

```
[87] from sklearn.metrics import accuracy_score, explained_variance_score, mean_squared_error,mean_absolute_error
     X = Xt
     Y = yt
     lamdda = np.arange(0, stop=50.1, step=0.1)
```

```
[88] for train_ix, test_ix in cv.split(X):
            # split data
            X_train, X_test = X[train_ix, :], X[test_ix, :]
            y_train, y_test = y[train_ix], y[test_ix]
```

```
[89] coefs = []
     MSE =list()
     for item in lamdda:
          Y_true, Y_predict = list(), list()
          ERR = list(); MERR = 0;
          for train_ix, test_ix in method.split(X):
              # split data
              X_train, X_test = X[train_ix, :], X[test_ix, :]
              y_train, y_test = Y[train_ix], Y[test_ix]
              ridge = linear_model.Ridge(alpha=item, fit_intercept=True)
              ridge.fit(X_train, y_train)

              yhat = ridge.predict(X_test)
              Y_true.append(y_test[0])
              Y_predict.append(yhat[0])
              er = abs(y_test[0]-yhat[0])
              ERR.append(er)
          for i in ERR:
              MERR = MERR + i*i

          MSE.append(MERR)
          coefs.append(ridge.coef_)
     print(len(MSE))
     print(MSE.index(min(MSE)))
     #print('Error: %.3f' % MERR)
```

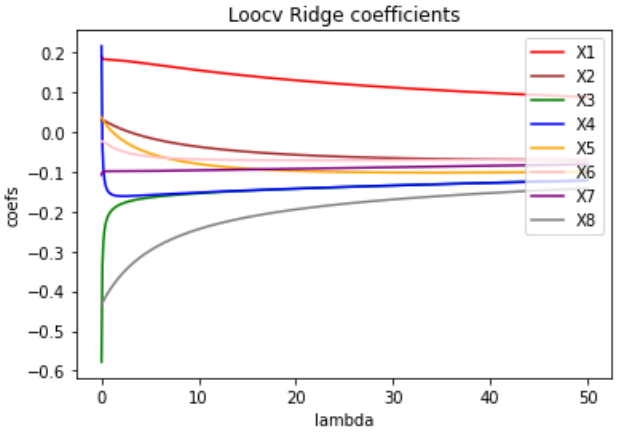
```
501
217
```

We can notice that the error would be minimum when lambda is 21.6,

Which is computed by (217*0.1 – 0.1)=21.6

The variation is more obvious than standard LOOCV.

```
[110] Coefs = np.array(coefs)
      y1=Coefs[:,0]
      y2=Coefs[:,1]
      y3=Coefs[:,2]
      y4=Coefs[:,3]
      y5=Coefs[:,4]
      y6=Coefs[:,5]
      y7=Coefs[:,6]
      y8=Coefs[:,7]
      ax = plt.gca()
      Colr = ['red', 'brown', 'green', 'blue', 'orange', 'pink', 'purple', 'grey']

      ax.plot(lamdda,y1,color= Colr[0])
      ax.plot(lamdda,y2,color= Colr[1])
      ax.plot(lamdda,y3,color= Colr[2])
      ax.plot(lamdda,y4,color= Colr[3])
      ax.plot(lamdda,y5,color= Colr[4])
      ax.plot(lamdda,y6,color= Colr[5])
      ax.plot(lamdda,y7,color= Colr[6])
      ax.plot(lamdda,y8,color= Colr[7])

      plt.xlabel('lambda')
      plt.ylabel('coefs')
      plt.title('Loocv Ridge coefficients')
      plt.axis('tight')
      plt.legend(('X1', 'X2', 'X3', 'X4', 'X5', 'X6', 'X7', 'X8'),loc='upper right')
      plt.show()
```

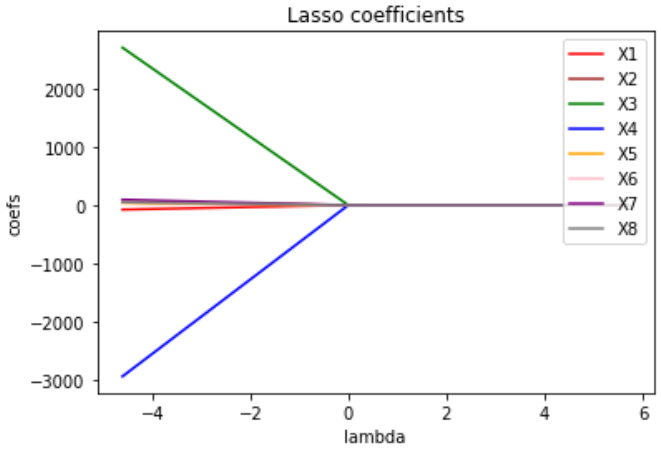


(e)

```
[53] coefs_Lasso = []
     for item in lam:
                ridge = linear_model.Lasso(alpha=item, fit_intercept=True)
                ridge.fit(Xt, yt)
                coefs_Lasso.append(ridge.coef_)
```

```
[54] Coefs_Lasso = np.array(coefs_Lasso)
     y1=Coefs_Lasso[:,0]
     y2=Coefs_Lasso[:,1]
     y3=Coefs_Lasso[:,2]
     y4=Coefs_Lasso[:,3]
     y5=Coefs_Lasso[:,4]
     y6=Coefs_Lasso[:,5]
     y7=Coefs_Lasso[:,6]
     y8=Coefs_Lasso[:,7]
```

```
[55] ax = plt.gca()
     Colr = ['red', 'brown', 'green', 'blue', 'orange', 'pink', 'purple', 'grey']

     ax.plot(lam,y1,color= Colr[0])
     ax.plot(lam,y2,color= Colr[1])
     ax.plot(lam,y3,color= Colr[2])
     ax.plot(lam,y4,color= Colr[3])
     ax.plot(lam,y5,color= Colr[4])
     ax.plot(lam,y6,color= Colr[5])
     ax.plot(lam,y7,color= Colr[6])
     ax.plot(lam,y8,color= Colr[7])


     plt.xlabel('lambda')
     plt.ylabel('coefs')
     plt.title('Lasso coefficients')
     plt.axis('tight')
     plt.legend(('X1', 'X2', 'X3', 'X4', 'X5', 'X6', 'X7', 'X8'),loc='upper right')
     plt.show()
```

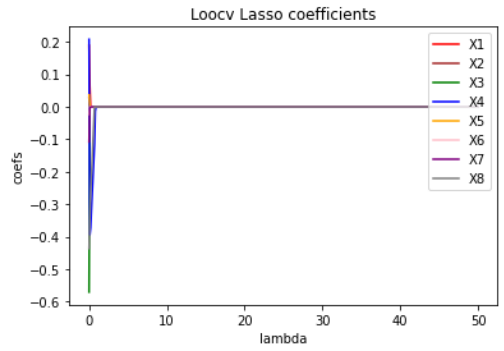


The coefficient of X3 decreases to zero while coefficient of X4 increases to zero. The X3 and X4 figures are in symmetry. Other features' coefficients keep stable at around zero.

(f)

```
[116] coefs_Lasso = []
      MSE =list()
      for item in lamdda:
            Y_true, Y_predict = list(), list()
            ERR = list(); MERR = 0;
            for train_ix, test_ix in method.split(X):
                # split data
                X_train, X_test = X[train_ix, :], X[test_ix, :]
                y_train, y_test = Y[train_ix], Y[test_ix]
                lasso = linear_model.Lasso(alpha=item, fit_intercept=True)
                lasso.fit(X_train, y_train)

                yhat = lasso.predict(X_test)
                Y_true.append(y_test[0])
                Y_predict.append(yhat[0])
                er = abs(y_test[0]-yhat[0])
                ERR.append(er)
            for i in ERR:
                MERR = MERR + i*i

            MSE.append(MERR)
            coefs_Lasso.append(lasso.coef_)
      print(len(MSE))
      print(MSE.index(min(MSE)))
```

```
[117] Coefs_Lasso = np.array(coefs_Lasso)
      y1=Coefs_Lasso[:,0]
      y2=Coefs_Lasso[:,1]
      y3=Coefs_Lasso[:,2]
      y4=Coefs_Lasso[:,3]
      y5=Coefs_Lasso[:,4]
      y6=Coefs_Lasso[:,5]
      y7=Coefs_Lasso[:,6]
      y8=Coefs_Lasso[:,7]
      ax = plt.gca()
      Colr = ['red', 'brown', 'green', 'blue', 'orange', 'pink', 'purple', 'grey']

      ax.plot(lamdda,y1,color= Colr[0])
      ax.plot(lamdda,y2,color= Colr[1])
      ax.plot(lamdda,y3,color= Colr[2])
      ax.plot(lamdda,y4,color= Colr[3])
      ax.plot(lamdda,y5,color= Colr[4])
      ax.plot(lamdda,y6,color= Colr[5])
      ax.plot(lamdda,y7,color= Colr[6])
      ax.plot(lamdda,y8,color= Colr[7])

      plt.xlabel('lambda')
      plt.ylabel('coefs')
      plt.title('Loocv Lasso coefficients')
      plt.axis('tight')
      plt.legend(('X1', 'X2', 'X3', 'X4', 'X5', 'X6', 'X7', 'X8'),loc='upper right')
      plt.show()
```



(g)

From previous screenshots we can find that Ridge Regression are more sensitive to small variation and can depict the fluctuation explicitly. Lasso Regression focus on the overall variation more, which means it could smooth the curves and remove some features.

So if our model has many variates, and almost all of them have limited impacts on model, we should use Ridge Regression to capture the tiny variation.

If only few variates have significant impacts on our model, we should use Lasso Regression to amplify their significance.

Q3:

Q3 : (a)

$$X = \begin{pmatrix} X_{11} & X_{12} & \cdots & X_{1p} \\ X_{21} & & & \\ \vdots & & \ddots & \\ X_{n1} & & \cdots & X_{np} \end{pmatrix}, \quad \beta = \begin{pmatrix} \beta_1 \\ \beta_2 \\ \vdots \\ \beta_p \end{pmatrix} \quad Y = \begin{pmatrix} Y_1 \\ Y_2 \\ \vdots \\ Y_n \end{pmatrix}$$

first consider left part of the inequation

$$\begin{aligned}
|\langle Y, X\beta \rangle| &= |Y^T \cdot (X\beta)| \\
&= |Y^T \cdot \hat{Y}| \\
&= |Y_1\hat{Y}_1 + Y_2\hat{Y}_2 + \cdots Y_n\hat{Y}_n| \\
&= |Y_1(X_{11}\beta_1 + X_{12}\beta_2 + \cdots X_{1p}\beta_p) + \cdots + Y_n(X_{n1}\beta_1 + X_{n2}\beta_2 + \cdots X_{np}\beta_p)| \\
&= |\beta_1(X_{11}Y_1 + X_{21}Y_2 + \cdots X_{n1}Y_n) + \cdots + \beta_p(X_{1p}Y_1 + X_{2p}Y_2 + \cdots X_{np}Y_n)| \\
&\le |\beta_1(X_{11}Y_1 + X_{21}Y_2 + \cdots X_{n1}Y_n)| + \cdots + |\beta_p(X_{1p}Y_1 + X_{2p}Y_2 + \cdots X_{np}Y_n)| \\
&\le |\beta_1(X_{1j}Y_1 + X_{2j}Y_2 + \cdots X_{nj}Y_n)| + \cdots + |\beta_p(X_{1j}Y_1 + X_{2j}Y_2 + \cdots X_{nj}Y_n)| \\
&\le |\beta_1| \cdot |(X_{1j}Y_1 + X_{2j}Y_2 + \cdots X_{nj}Y_n)| + \cdots + |\beta_p| \cdot |(X_{1j}Y_1 + X_{2j}Y_2 + \cdots X_{nj}Y_n)| \\
&= |(X_{1j}Y_1 + X_{2j}Y_2 + \cdots X_{nj}Y_n)| \cdot (|\beta_1| + \cdots + |\beta_p|) \\
&= \max_j |X_j^T Y| \cdot \textstyle\sum_j |\beta_j|
\end{aligned}$$

thus, $|\langle Y, X\beta \rangle| \le \max_j |X_j^T Y| \cdot \sum_j |\beta_j|$, when X_j denotes jth colum of X.

Q3 (b) from part (a) we know $\max_j |X_j^T Y| \sum_j |\beta_j| \geq |\langle Y, X\beta \rangle|$

$$L(\beta) = \frac{1}{2}\|Y - X\beta\|_2^2 + \lambda\|\beta\|_1$$
$$= \frac{1}{2}(Y - X\beta)^T(Y - X\beta) + \lambda\|\beta\|_1$$
$$\geq \frac{1}{2}(Y - X\beta)^T(Y - X\beta) + \max_j |X_j^T Y| \cdot \sum_j |\beta_j|$$
$$\geq \frac{1}{2}(Y - X\beta)^T(Y - X\beta) + |\langle Y, X\beta \rangle|$$
$$= \frac{1}{2}(YY^T - 2Y^T X\beta + \beta^T X^T X\beta) + |Y^T X\beta|$$

Let $L_b(\beta) = \frac{1}{2}(YY^T - 2Y^T X\beta + \beta^T X^T X\beta) + |Y^T X\beta|$, then $L(\beta) \geq L_b(\beta)$

we need to split the case by $Y^T X\beta \geq 0$, and $Y^T X\beta < 0$

if $Y^T X\beta \geq 0$, $L_b(\beta) = \frac{1}{2}(YY^T - 2Y^T X\beta + \beta^T X^T X\beta) + Y^T X\beta$
$$= \frac{1}{2}(YY^T + \beta^T X^T X\beta)$$
$$= \frac{1}{2}(YY^T + \hat{Y}^T\hat{Y})$$

for $\hat{Y}^T\hat{Y}$, it's non-negative because its the sum of each element square

YY^T is settled, so to make $L_b(\beta)$ minimal, $\hat{Y}^T\hat{Y}$ should be lease

only when $\beta = 0_p$, $\hat{Y}^T\hat{Y} = 0$, which is its minimum

if $Y^T X\beta \leq 0$, $L_b(\beta) = \frac{1}{2}(YY^T - 2Y^T X\beta + \beta^T X^T X\beta) - Y^T X\beta$
$$= \frac{1}{2}(YY^T - 4Y^T X\beta + \beta^T X^T X\beta)$$

To make $L_b(\beta)$ least, we should make $\frac{1}{2}(\beta^T X^T X\beta - 4Y^T X\beta)$ least

we notice that $(-4Y^T X\beta) \geq 0$ due to the assumption.

from previous demonstration $\beta^T X^T X\beta \geq 0$

only when $\beta = 0$, $\frac{1}{2}(\beta^T X^T X\beta - 4Y^T X\beta)$ has its minimum. 0.

Combine these steps, we find $\hat{\beta} = 0_p$ is a solution of Lasso

Q3 (c) from the previous demonstration in (b)

We know $\beta = 0_p$ is a solution to this Lasso problem.

We now need to prove $\beta = 0_p$ is the unique solution.

suppose there is at least another solution $\beta_1 \neq 0_p$, besides $\beta_0 = 0_p$

设 $Y^T X\beta_1 > 0$, $L_b(\beta_1) = \frac{1}{2}(YY^T - 2Y^T X\beta_1 + \beta_1^T X^T X\beta_1) + Y^T X\beta_1$

$= \frac{1}{2}YY^T + \frac{1}{2}\beta_1^T X^T X\beta_1$

$L_b(\beta_0) = \frac{1}{2}YY^T$

$\therefore L_b(\beta_1) = L_b(\beta_0) + \frac{1}{2}\beta_1^T X^T X\beta_1 \quad (\beta_1^T X^T X\beta_1 > 0)$

$\therefore L_b(\beta_1) > L_b(\beta_0)$

设 $Y^T X\beta_1 < 0$, $L_b(\beta_1) = \frac{1}{2}(YY^T - 2Y^T X\beta_1 + \beta^T X^T X\beta_1) - Y^T X\beta_1$

$= \frac{1}{2}YY^T + \frac{1}{2}(\beta_1^T X^T X\beta_1 - 4Y^T X\beta_1)$

$= L_b(\beta_0) + \frac{1}{2}(\beta_1^T X^T X\beta_1 - 4Y^T X\beta_1)$

Because $\beta_1^T X^T X\beta_1 > 0$, $Y^T X\beta < 0$, $\therefore -4Y^T X\beta_1 > 0$

$\therefore \frac{1}{2}(\beta_1^T X^T X\beta_1 - 4Y^T X\beta_1) > 0$

$\therefore L_b(\beta_1) > L_b(\beta_0)$

Now we know, as long as $\beta_1 \neq 0$, $L_b(\beta_1) > L_b(\beta_0)$

In other words, only when $\beta = 0$, $L(\beta)$ has minimal value.

Thus $\beta = 0_p$, is the unique solution to this Lasso problem.