COMP9417 - Machine Learning
Homework 1: Linear Regression & Friends

Introduction In this homework, we will explore Linear Regression and its regularized counterparts, LASSO and Ridge Regression, in more depth.

Points Allocation There are a total of 25 marks. The available marks are:

- Question 1 a): 4 marks
- Question 1 b): 2 marks
- Question 2 a): 1 mark
- Question 2 b): 1 mark
- Question 2 c): 3 marks
- Question 2 d): 3 marks
- Question 2 e): 2 marks
- Question 2 f): 2 marks
- Question 2 g): 1 mark
- Question 3 a): 2 marks
- Question 3 b): 2 marks
- Question 3 c): 2 marks

What to Submit

- A single PDF file which contains solutions to each question. For each question, provide your solution in the form of text and requested plots. For some questions you will be requested to provide screen shots of code used to generate your answer — only include these when they are explicitly asked for.
- **.py file(s) containing all code you used for the project, which should be provided in a separate .zip file.** This code must match the code provided in the report.
- You may be deducted points for not following these instructions.
- You may be deducted points for poorly presented/formatted work. Please be neat and make your solutions clear. Start each question on a new page if necessary.

- You **cannot** submit a Jupyter notebook; this will receive a mark of zero. This does not stop you from developing your code in a notebook and then copying it into a .py file though, or using a tool such as **nbconvert** or similar.
- We will set up a Moodle forum for questions on this homework. Please read the existing questions before posting new questions. Please do some basic research online before posting questions. Please only post clarification questions. Any questions deemed to be *fishing* for answers will be ignored and/or deleted.
- Please check the Moodle forum for updates to this spec. It is your responsibility to check for announcements about the spec.
- Please complete your homework on your own, do not discuss your solution with other people in the course. General discussion of the problems is fine, but you must write out your own solution and acknowledge if you discussed any of the problems in your submission (including their name and zID).
- As usual, we monitor all online forums such as Chegg, StackExchange, etc. Posting homework questions on these site is equivalent to plagiarism and will result in a case of academic misconduct.

When and Where to Submit

- Due date: Week 3, Sunday **June 20th**, 2021 by **11:55pm**.
- Late submissions will incur a penalty of 20% per day (from the ceiling, i.e., total marks available for the homework) for the first 5 days. For example, if you submit 2 days late, the maximum possible mark is 60% of the available 25 marks.
- Submission must be done through Moodle, no exceptions.

Question 1. Simple Linear Regression

(a) Consider a data set consisting of X values (features) $X_1, \ldots, X_n$ and Y values (responses) $Y_1, \ldots, Y_n$. Let $\hat{\beta}_0, \hat{\beta}_1, \hat{\sigma}$ be the output of running ordinary least squares (OLS) regression on the data. Now define the transformation:

$$\widetilde{X}_i = c(X_i + d),$$

for each $i = 1, \ldots, n$, where $c \neq 0$ and d are arbitrary real constants. Let $\widetilde{\beta}_0, \widetilde{\beta}_1, \widetilde{\sigma}$ be the output of OLS on the data $\widetilde{X}_1, \ldots, \widetilde{X}_n$ and $Y_1, \ldots, Y_n$. Write equations for $\widetilde{\beta}_0, \widetilde{\beta}_1, \widetilde{\sigma}$ in terms of $\hat{\beta}_0, \hat{\beta}_1, \hat{\sigma}$ (and in terms of c, d), and be sure to justify your answers. Note that the estimate of error in OLS is taken to be:

$$\hat{\sigma} = \sqrt{\frac{\hat{e}^T \hat{e}}{n-p}},$$

where $\hat{e}$ is the vector of residuals, i.e. with i-the element $\hat{e}_i = Y_i - \hat{Y}_i$, where $\hat{Y}_i$ is the i-th prediction made by the model, and p is the number of features (so in this case $p = 2$).

Solution:
In tutorials we showed that the least squares estimates for $\beta_0, \beta_1, \sigma^2$ were

$$\hat{\beta}_1 = \frac{\overline{xy} - \overline{x}\,\overline{y}}{\overline{x^2} - \overline{x}^2}, \qquad \hat{\beta}_0 = \overline{y} - \beta_1 \overline{x}, \qquad \hat{\sigma}^2 = \frac{\hat{e}^T \hat{e}}{N-p}.$$

Letting $\widetilde{x}$ be the vector with i-th element $c(x_i + d)$ for some constants c, d, the least squares estimates are simply

$$\widetilde{\beta}_1 = \frac{\overline{\widetilde{x}y} - \overline{\widetilde{x}}\,\overline{y}}{\overline{\widetilde{x}^2} - \overline{\widetilde{x}}^2}, \qquad \widetilde{\beta}_0 = \overline{y} - \beta_1 \overline{\widetilde{x}}, \qquad \widetilde{\sigma}^2 = \frac{\widetilde{e}^T \widetilde{e}}{N-p}.$$

To write the new estimates in terms of the original estimates, we note the following results:

- $\overline{\widetilde{x}} = \frac{1}{n}\sum_{i=1}^n c(x_i + d) = c(\overline{x} + d)$,
- $\overline{\widetilde{x}y} = \frac{1}{n}\sum_{i=1}^n c(x_i + d)y_i = c(\overline{xy} + d\overline{y})$,
- $\overline{\widetilde{x}^2} = \frac{1}{n}\sum_{i=1}^n c^2(x_i + d)^2 = c^2(\overline{x^2} + 2d\overline{x} + d^2)$,
- $\overline{\widetilde{x}}^2 = c^2(\overline{x} + d)^2 = c^2(\overline{x}^2 + 2d\overline{x} + d^2)$.

Using these, we can write

$$\begin{aligned}
\widetilde{\beta}_1 &= \frac{c(\overline{xy} + d\overline{y}) - c(\overline{x}\,\overline{y} + d\overline{y})}{c^2(\overline{x^2} + 2d\overline{x} + d^2) - c^2(\overline{x}^2 + 2d\overline{x} + d^2)} \\
&= \frac{1}{c}\frac{\overline{xy} - \overline{x}\,\overline{y}}{\overline{x^2} - \overline{x}^2} \\
&= \frac{\hat{\beta}_1}{c}.
\end{aligned}$$

Similarly, we have

$$\begin{aligned}\widetilde{\beta}_0 &= \overline{y} - \widetilde{\beta}_1 \overline{\widetilde{x}} \\ &= \overline{y} - \frac{\hat{\beta}_1}{c} c(\overline{x} + d) \\ &= \overline{y} - \hat{\beta}_1 \overline{x} - \hat{\beta}_1 d \\ &= \hat{\beta}_0 - \hat{\beta}_1 d.\end{aligned}$$

Finally, note that the i-th residual under the transformed data (and corresponding parameter estimates) is:

$$\begin{aligned}\widetilde{e}_i &= y_i - \widetilde{\beta}_0 - \widetilde{\beta}_1 \widetilde{x}_i \\ &= y_i - (\hat{\beta}_0 - \hat{\beta}_1 d) - \frac{\hat{\beta}_1}{c} c(x_i + d) \\ &= y_i - \hat{\beta}_0 + \hat{\beta}_1 d - \hat{\beta}_1 x_i - \hat{\beta}_1 d \\ &= y_i - \hat{\beta}_0 - \hat{\beta}_1 x_i \\ &= \hat{e}_i.\end{aligned}$$

Therefore, $\widetilde{e} = \hat{e}$, and so $\widetilde{\sigma} = \hat{\sigma}$. This last result tells us that the predictions are the same under the transformed data and varables as they are under the original data and variables:

$$y_i = \hat{\beta}_0 + \hat{\beta}_1 x_i = \widetilde{\beta}_0 + \widetilde{\beta}_1 \widetilde{x}_i.$$

The intuition here is that we are simply shifting and scaling each data point by the same amount. The horizontal shift by d units does not impact the relationship between points, so it has no impact on the estimate of the slope. However, since the slope captures the change in y for a corresponding increase in x, scaling x by c units must be compensated for by dividing the slope by c units. To understand the new estimate for the intercept, we see that for a fixed scale factor c, we simply need to vertically shift the intercept to compensate for the horizontal shift of the data. There is no change in the estimate of the standard error since the standard error is already standardised to be of the same unit measurement as the response, which in this case is unchanged when we transform x.

(b) Suppose you have a dataset where X takes only two values while Y can take arbitrary real values. To consider a concrete example, consider a clinical trial where $X_i = 1$ indicates that the i-th patient receives a dose of a particular drug (the treatment), and $X_i = 0$ indicates that they did not, and Y_i is the real-valued outcome for the i-th patient, e.g. blood pressure. Let $\overline{Y}_T$ and $\overline{Y}_P$ indicate the sample mean outcomes for the treatment group and non-treatment (placebo) group, respectively. What will be the value of the OLS coefficients $\hat{\beta}_0, \hat{\beta}_1$ in terms of the group means?

Solution:
Let N be the total number of observations, and N_P, N_T the number of observations corre-

sponding to placebo and treatment, respecively, so that $N = N_P + N_T$. Also let $\{Y_{Pi}\}_{i=1}^{N_P}$ and $\{Y_{Ti}\}_{i=1}^{N_T}$ denote the responses for placebo and treatment groups, respectively, and define $\{X_{Pi}\}_{i=1}^{N_P}$ and $\{X_{Ti}\}_{i=1}^{N_T}$ to be the feature values for placebo and treatment, respectively. Note that $X_{Pi} = 0$ for $i = 1, \ldots, N_p$ and $X_{Ti} = 0$ for $i = 1, \ldots, N_T$.

Then, we note the following results:

- $\overline{X} = \frac{1}{N}\sum_{i=1}^{N} X_i = \frac{1}{N}\left\{\sum_{i=1}^{N_P} X_{Pi} + \sum_{i=1}^{N_T} X_{Ti}\right\} = \frac{1}{N}\sum_{i=1}^{N_T} X_{Ti} = \frac{N_T}{N}$
- $\overline{X^2} = \frac{1}{N}\sum_{i=1}^{N} X_i^2 = \frac{1}{N}\left\{\sum_{i=1}^{N_P} X_{Pi}^2 + \sum_{i=1}^{N_T} X_{Ti}^2\right\} = \frac{1}{N}\sum_{i=1}^{N_T} X_{Ti}^2 = \frac{N_T}{N}$
- $\overline{XY} = \frac{1}{N}\sum_{i=1}^{N} X_i Y_i = \frac{1}{N}\left\{\sum_{i=1}^{N_P} X_{Pi}Y_{Pi} + \sum_{i=1}^{N_T} X_{Ti}Y_{Ti}\right\} = \frac{1}{N}\sum_{i=1}^{N_T} Y_{Ti} = \frac{N_T}{N}\overline{Y}_T.$
- $\overline{Y} = \frac{1}{N}\sum_{i=1}^{N} Y_i = \frac{1}{N}\left\{\sum_{i=1}^{N_P} Y_{Pi} + \sum_{i=1}^{N_T} Y_{Ti}\right\} = \frac{N_T}{N}\overline{Y}_T + \frac{N_P}{N}\overline{Y}_P$

Then, for our OLS estimate of $\hat{\beta}_1$, we have

$$\begin{aligned}
\hat{\beta}_1 &= \frac{\overline{XY} - \bar{X}\bar{Y}}{\overline{X^2} - \overline{X}^2} \\
&= \frac{\frac{N_T}{N}\overline{Y}_T - \frac{N_T}{N}\overline{Y}}{\frac{N_T}{N} - \frac{N_T^2}{N^2}} \\
&= \frac{\overline{Y}_T - \overline{Y}}{\frac{N_p}{N}} \\
&= \frac{\overline{Y}_T - \frac{N_T}{N}\overline{Y}_T - \frac{N_P}{N}\overline{Y}_P}{\frac{N_p}{N}} \\
&= \frac{\overline{Y}_T\left(1 - \frac{N_T}{N}\right) - \frac{N_P}{N}\overline{Y}_P}{\frac{N_p}{N}} \\
&= \overline{Y}_T - \overline{Y}_P.
\end{aligned}$$

The OLS estimate of $\hat{\beta}_0$ is then

$$\begin{aligned}
\hat{\beta}_0 &= \overline{Y} - \hat{\beta}_1\overline{X} \\
&= \left(\frac{N_T}{N}\overline{Y}_T + \frac{N_P}{N}\overline{Y}_P\right) - (\overline{Y}_T - \overline{Y}_P)\frac{N_T}{N} \\
&= \frac{N_P}{N}\overline{Y}_P + \frac{N_T}{N}\overline{Y}_P \\
&= \overline{Y}_P\left(\frac{N_P + N_T}{N}\right) \\
&= \overline{Y}_P.
\end{aligned}$$

Therefore, the model just predicts the group average for each of the two groups, since for a new observation with $X = 1$ (received treatment), the model is simply:

$$Y = \hat{\beta}_0 + \hat{\beta}_1 X = \overline{Y}_P + (\overline{Y}_T - \overline{Y}_P) \times 1 = \overline{Y}_T$$

and for $X = 0$ (no treatment):

$$Y = \hat{\beta}_0 + \hat{\beta}_1 X = \overline{Y}_P + (\overline{Y}_T - \overline{Y}_P) \times 0 = \overline{Y}_P.$$

What to submit: For both parts of the question, present your solution neatly - photos of handwritten work or using a tablet to write the answers is fine. Please include all working and circle your final answers.

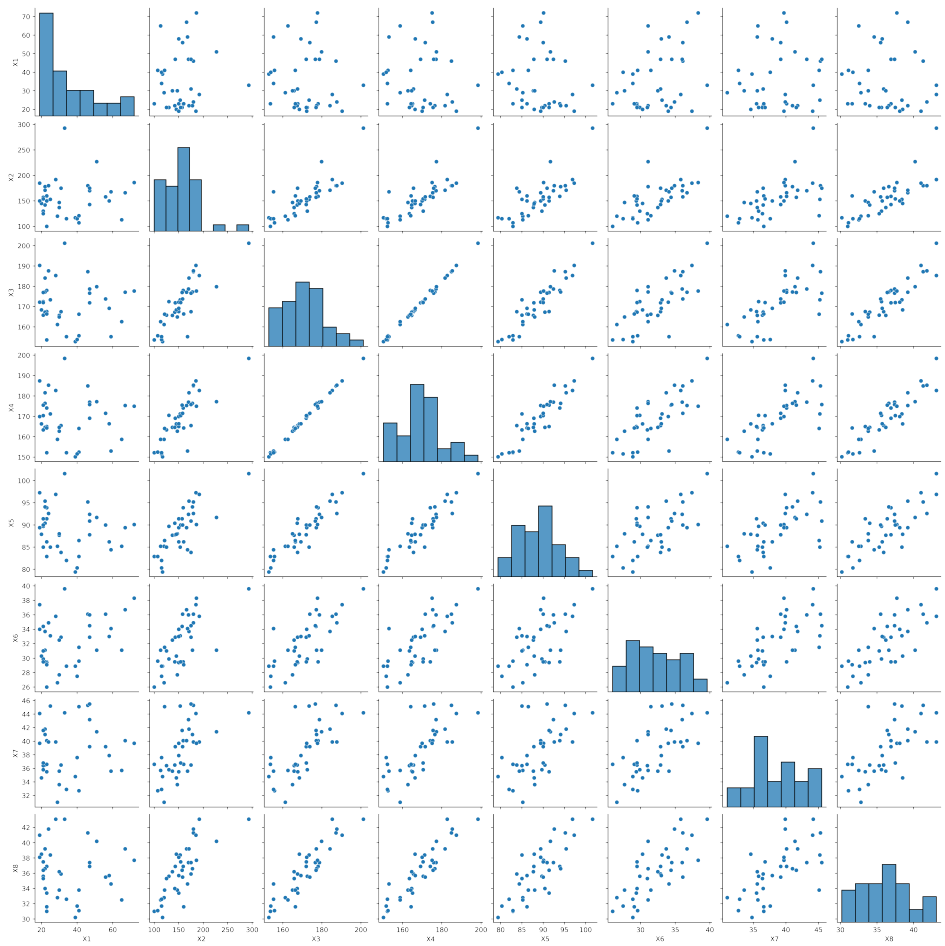
Question 2. LASSO vs. Ridge Regression

In this problem we will consider the dataset provided in `data.csv`, with response variable Y, and features $X_1, \ldots, X_8$.

(a) Use a pairs plot to study the correlations between the features. In 3-4 sentences, describe what you see and how this might affect a linear regression model. *What to submit: a single plot, some commentary.*

Solution:

From the pairs plot, we can see an almost perfect correlation between X3, X4 and X5. In general, there is a large amount of multicollinearity in the dataset. Including all of the variables into the model will introduce multicollinearity in to the regression, which may cause issues such as not being able to compute the MLE estimate of β since the data matrix may be rank deficient and so $X^T X$ may not be invertible. Further issues would be that interpretation of the regression coefficients for each of the predictors would not be as accurate as in the non-collinear case, since we cannot hold any one variable constant whilst changing the other. We might also expect to see larger standard error estimates for the regression coefficients.



Code for this section:

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
from sklearn.linear_model import Lasso, Ridge, LinearRegression
import seaborn as sns

data = pd.read_csv("data.csv")
X = data.iloc[:,:-1]
Y = data.iloc[:,-1]

sns.pairplot(data.iloc[:,:-1])
plt.savefig("figures/PairsPlot.png", dpi=400)
plt.show()
```

(b) In order for LASSO and Ridge to be run properly, we often rescale the features in the dataset. First, rescale each feature so that it has zero mean, and then rescale it so that $\sum_{i=1}^{n} X_{ij}^2 = n$ where n

denotes the total number of observations. *What to submit: print out the sum of squared observations of each of the 8 (transformed) features, i.e.* $\sum_i X_{ij}^2$ *for* $j = 1, \ldots, 8$

Solution:

The squared sums should all be equal to $n = 38$ if scaling is done properly.

```
        scaled_X = np.apply_along_axis(lambda x: x - x.mean(), 0, X)
        for i in range(scaled_X.shape[1]):
            xsqbar = np.sqrt(np.mean(scaled_X[:,i]**2))
            scaled_X[:, i] /= xsqbar

        np.power(scaled_X,2).sum(axis=0)

```

(c) Now we will apply ridge regression to this dataset, recall that ridge regression is defined as the solution to the optimisation:

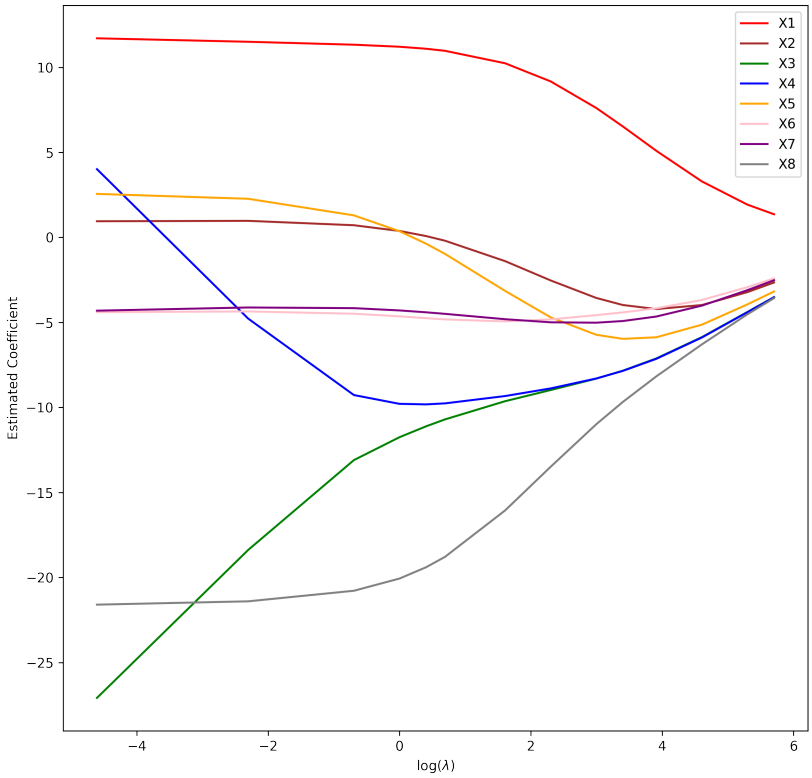
$$\hat{\beta} = \arg\min \left\{ \frac{1}{2} \|Y - X\beta\|_2^2 + \lambda \|\beta\|_2^2 \right\}.$$

Run ridge regression with $\lambda = \{0.01, 0.1, 0.5, 1, 1.5, 2, 5, 10, 20, 30, 50, 100, 200, 300\}$. Create a plot with x-axis representing $\log(\lambda)$, and y-axis representing the value of the coefficient for each feature in each of the fitted ridge models. In other words, the plot should describe what happens to each of the coefficients in your model for the different choices of λ. For this problem you are permitted to use the `sklearn` implementation of Ridge regressiom to run the models and extract the coefficients, and base matplotlib/numpy to create the plots but no other packages are to be used to generate this plot. In a few lines, comment on what you see, in particular what do you observe for features $3, 4, 5$?

What to submit: a single plot, some commentary, a screen shot of the code used for this section. Your plot must have a legend, and you must use the following colors: ['red', 'brown', 'green', 'blue', 'orange', 'pink', 'purple', 'grey'] for the features X1,..,X8 in the plot.

Solution:

Here I run the model for the various values of λ and plot the Ridge path for each of the covariates in the model excluding the intercept. We can see that as λ is made larger, the coefficients are shrunk towards zero as the optimisation method tries harder to satistfy the norm constraint. In particular, we can see that for very small values of λ, X3 and X4 had almost opposite coefficients, and as λ is increased, they are given equal weight in the model. This is due to the nature of the 2 norm, which prefers to give uniform weight to variables that are interchangeable in the model, rather than remove one and keep the other as is the case with the 1-norm. These two variables are perfectly correlated so we see this effect much earlier. For the X5 variable, which is also strongly correlated, we see the model give all three variables uniform weight eventually.



Code used for this section:

```
            lambdas = [0.01,0.1,0.5,1,1.5, 2,5,10, 20, 30,50, 100, 200, 300]
            N = len(lambdas)
            coefs_mat = np.zeros((scaled_X.shape[1], N))
            for i in range(N):
                L = lambdas[i]
                ridge_lm = Ridge(alpha=L).fit(scaled_X,Y)
                coefs_mat[:,i] = ridge_lm.coef_

            colors = ['red', 'brown', 'green', 'blue', 'orange', 'pink', 'purple', '
    grey']
            plt.figure(figsize=(10,10))
            for i in range(X.shape[1]):
                lab = "X" + str(i + 1)
                plt.plot(np.log(lambdas), coefs_mat[i], label=lab, color=colors[i])
                plt.legend()
            plt.xlabel(r"log($\lambda$)")
            plt.ylabel("Estimated Coefficient")
            plt.savefig("figures/RidgePath.png", dpi=300)
```

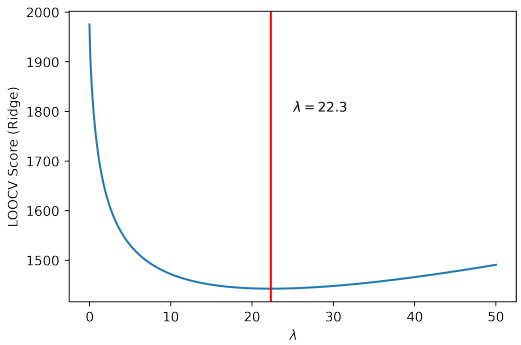
```
            plt.show()

```

(d) In this part, we will use Leave-One-Out Cross Validation (LOOCV) to find a good value of λ for the ridge problem. Create a fine grid of λ values running from 0 to 50 in increments of 0.1, so the grid would be: $0, 0.1, 0.2, \ldots, 50$. For each data point $i = 1, \ldots, n$, run ridge with each λ value on the dataset with point i removed, find $\hat{\beta}$, then get the leave-one-out error for predicting Y_i. Average the squared error over all n choics of i. Plot the leave-one-out error against λ and find the best λ value. Compare your results to standard Ordinary Least Squares (OLS), does the ridge seem to give better prediction error based on your analysis? Note that for this question you are **not** permitted to use any existing packages that implement cross validation, you must write the code yourself from scratch. You must create the plot yourself from scratch using basic matplotlib functionality.

What to submit: a single plot, some commentary, a screen shot of any code used for this section.

Solution:

We see the optimal value in terms of lowest LOOCV score occuring for a choice of $\lambda = 22.3$. Obviously this is doing much better than OLS since OLS corresponds to the $\lambda = 0$ case.



Code used for this section:

```
        lambdas = np.arange(0, 50.1, step=0.1)
        n = scaled_X.shape[0]
        N = lambdas.shape[0]
        CV_score = np.zeros(N)
        curIdx = 0
        for L in lambdas:
            sq_errs = 0.
            for i in range(n):
                x_i = scaled_X[i]
                x_removed_i = np.delete(scaled_X, i, axis=0)      # dataset without i-
    th observation
                y_i = Y[i]
                y_removed_i = np.delete(Y, i, axis=0)

                mod = Ridge(alpha=L).fit(x_removed_i, y_removed_i)
                sq_errs += (mod.predict(x_i.reshape(1,-1))-y_i)**2

```

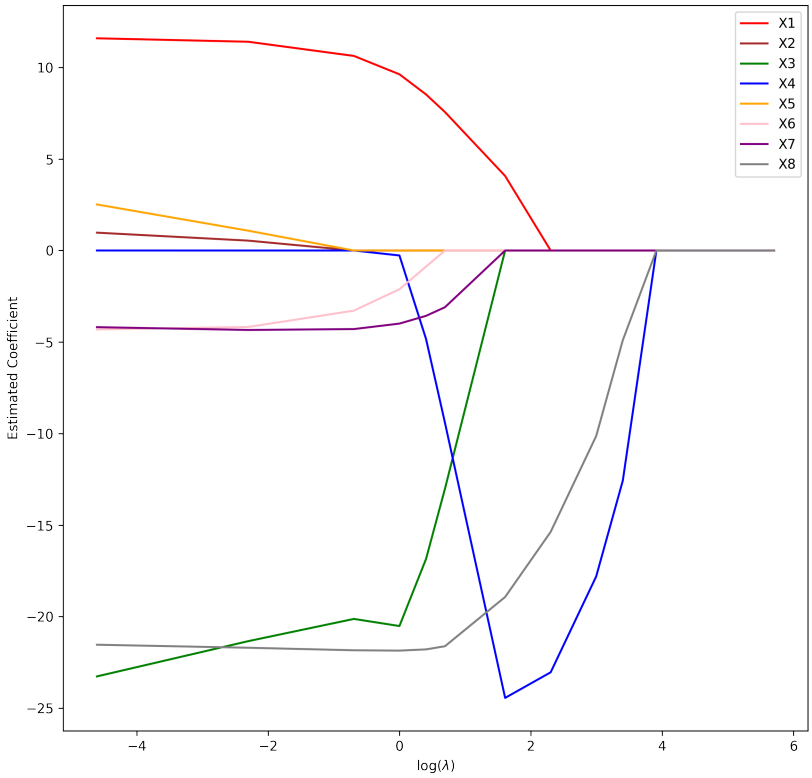
```
            CV_score[curIdx] = sq_errs/n
            curIdx += 1

        min_idx = np.argmin(CV_score)
        plt.plot(lambdas, CV_score)
        plt.xlabel(r"$\lambda$")
        plt.ylabel("LOOCV Score (Ridge)")
        plt.axvline(x=lambdas[min_idx], color="red")
        plt.annotate(f"$\lambda = {lambdas[min_idx]}$", xy=(25,1800))
        plt.savefig("figures/RidgeCV.png", dpi=400)
        plt.show()

```

(e) Recall the LASSO problem:

$$\hat{\beta} = \arg\min \left\{ \frac{1}{2} \|Y - X\beta\|_2^2 + \lambda \|\beta\|_1 \right\}.$$

Repeat part (c) for the LASSO. *What to submit: a single plot, some commentary, a screen shot of the code used for this section. You must use the same color scheme as in part (c).*

Solution:
We see a similar pattern to the ridge case except that now the 1-norm penalty of the LASSO forces X4 to be zero and attached all weight to X3 since these two variables carry the same signal. In the ridge case the signal was spread uniformly across, but the LASSO prefers sparse solutions and so picks only one of them. The LASSO ends up setting all variables to zero as λ is made to be increasingly large, something that does not happen in the Ridge case.



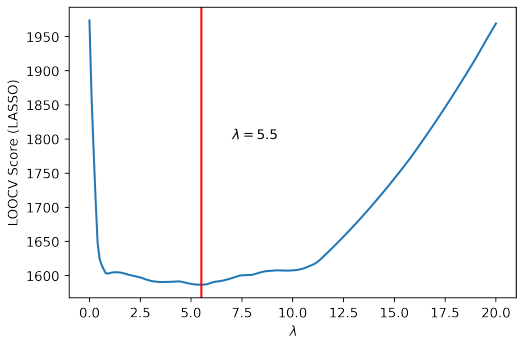
Code used for this section:

```
        lambdas = [0.01,0.1,0.5,1,1.5, 2,5,10, 20, 30,50, 100, 200, 300]
        N = len(lambdas)
        coefs_mat = np.zeros((scaled_X.shape[1], N))
        for i in range(N):
            L = lambdas[i]
            lasso_lm = Lasso(alpha=L).fit(scaled_X,Y)
            coefs_mat[:,i] = lasso_lm.coef_

        colors = ['red', 'brown', 'green', 'blue', 'orange', 'pink', 'purple', 'grey']
        plt.figure(figsize=(10,10))
        for i in range(X.shape[1]):
            lab = "X" + str(i + 1)
            plt.plot(np.log(lambdas), coefs_mat[i], label=lab, color=colors[i])
            plt.legend()
        plt.xlabel(r"log($\lambda$)")
        plt.ylabel("Estimated Coefficient")
        plt.savefig("figures/LASSOPath.png", dpi=300)
        plt.show()
```

```

```

(f) Repeat the leave-one-out analysis of part (d) for the LASSO and for a grid of λ values $0, 0.1, \ldots, 20$. Note that `sklearn` will throw some warnings for the $\lambda = 0$ case which can be safely ignored for our purposes. *What to submit: a single plot, some commentary, a screen shot of the code used for this section.*

Solution:



Code used for this section:

```
            lambdas = np.arange(0, 20.1, step=0.1)
            n = scaled_X.shape[0]
            N = lambdas.shape[0]
            CV_score = np.zeros(N)
            curIdx = 0
            for L in lambdas:
                sq_errs = 0.
                for i in range(n):
                    x_i = scaled_X[i]
                    x_removed_i = np.delete(scaled_X, i, axis=0)      # dataset
    without i-th observation
                    y_i = Y[i]
                    y_removed_i = np.delete(Y, i, axis=0)

                    mod = Lasso(alpha=L).fit(x_removed_i, y_removed_i)
                    sq_errs += (mod.predict(x_i.reshape(1,-1))-y_i)**2

                CV_score[curIdx] = sq_errs/n
                curIdx += 1

            min_idx = np.argmin(CV_score)
            plt.plot(lambdas, CV_score)
            plt.xlabel(r"$\lambda$")
            plt.ylabel("LOOCV Score (LASSO)")
            plt.axvline(x=lambdas[min_idx], color="red")
            plt.annotate(f"$\lambda = {lambdas[min_idx]}$", xy=(7,1800))
            plt.savefig("figures/LASSOCV.png", dpi=400)
            plt.show()

```

(g) Briefly comment on the differences you observed between the LASSO and Ridge. Which model do you prefer and why? Provide reasonable justification here for full marks. *What to submit: some commentary and potentially plots if your discussion requires it.*

Solution:
Roughly, students should note something along the lines of: If the goal is prediction, then ridge is the preferred choice given the LOO analysis. If the goal is inference (understanding the relationship between response and covariates), then the LASSO gives us a much simpler model.

Question 3. Sparse Solutions with LASSO

In this question, we will try to understand why LASSO regression yields sparse solutions. Sparse means that the solution of the LASSO optimisation problem:

$$\hat{\beta} = \arg\min \left\{ \frac{1}{2}\|Y - X\beta\|_2^2 + \lambda\|\beta\|_1 \right\}$$

has most of its entries $\hat{\beta}_j = 0$, which you may have observed empirically in the previous question. To study this from a theoretical perspective, we will consider a somewhat extreme case in which we take the penalty term λ to be very large, and show that the optimal LASSO solution is $\hat{\beta} = 0_p$, the vector of all zeroes. Assume that $X \in \mathbb{R}^{n\times p}$, $Y \in \mathbb{R}^n$ and the optimisation is over $\beta \in \mathbb{R}^p$.

(a) Consider the quantity $|\langle Y, X\beta\rangle|$. Show that $|\langle Y, X\beta\rangle| \leq \max_j |X_j^T Y| \sum_j |\beta_j|$, where X_j denotes the j-th column of X.

Solution:

$$\begin{aligned}
|\langle Y, X\beta\rangle| &= |\langle Y, X_1\beta_1 + X_2\beta_2 + \dots X_p\beta_p\rangle| \\
&= \left| \left\langle Y, \sum_{j=1}^{p} X_j\beta_j \right\rangle \right| \\
&= \left| \sum_{j=1}^{p} \langle Y, X_j\beta_j\rangle \right| \\
&= \left| \sum_{j=1}^{p} \beta_j \langle Y, X_j\rangle \right| \\
&\leq \sum_{j=1}^{p} |\beta_j| \, |\langle Y, X_j\rangle| \\
&\leq \sum_{j=1}^{p} |\beta_j| \, |X_j^T Y| \\
&\leq \max_j |X_j^T Y| \sum_{j=1}^{p} |\beta_j|,
\end{aligned}$$

where the first inequality follows by the triangle inequality. Alternatively, note that the result actually follows immediately by applying Hölder's inequality. Many students here made the same error along the lines of stating that,

$$\left|\sum_{j=1}^{n}(X_j^T Y)\beta_j\right| \leq \left|\sum_{j=1}^{n}\max_j(X_j^T Y)\beta_j\right|.$$

This is not true, for example, assume that $n = 3$ and that

$$X_1^T Y = 2, \quad X_2^T Y = -5 \quad X_3^T Y = 1,$$

so that $\max_j X_j^T Y = 2$, and further assume that $\beta_1 = 1, \beta_2 = -2, \beta_3 = 1$, the

$$\left|\sum_{j=1}^{n}(X_j^T Y)\beta_j\right| = |2 \times 1 - 5 \times (-2) + 1 \times 1| = |2 + 10 + 1| = 13,$$

whereas

$$\left|\sum_{j=1}^{n}\max_j(X_j^T Y)\beta_j\right| = |2 \times (\beta_1 + \beta_2 + \beta_3)| = 0$$

(b) We will now assume that λ is very large, such that it satisfies:

$$\lambda \geq \max_j |X_j^T Y|.$$

Using the result of part (a), and the assumption on λ, prove that $\hat{\beta} = 0_p$ is a solution of the LASSO problem.

Solution:
To show that $\hat{\beta} = 0_p$, we need to show that 0_P achieves the smallest loss of all possible choices of $\beta \in \mathbb{R}^p$, i.e. for any $\beta \in \mathbb{R}^p$, it holds that

$$\ell(0_p) \leq \ell(\beta),$$

where

$$\ell(\beta) = \frac{1}{2}\|Y - X\beta\|_2^2 + \lambda\|\beta\|_1.$$

First, note that

$$\ell(0_p) = \frac{1}{2}\|Y\|_2^2.$$

Then, let $\beta \in \mathbb{R}^p$ such that $\beta \neq 0_p$, we have

$$\begin{aligned}
\ell(\beta) &= \frac{1}{2}\|Y - X\beta\|_2^2 + \lambda\|\beta\|_1 \\
&= \frac{1}{2}\|Y\|_2^2 + \frac{1}{2}\|X\beta\|_2^2 - \langle Y, X\beta\rangle + \lambda\|\beta\|_1 \\
&\geq \frac{1}{2}\|Y\|_2^2 + \frac{1}{2}\|X\beta\|_2^2 - \langle Y, X\beta\rangle + \max_j |X_j^T Y|\|\beta\|_1 \\
&\geq \frac{1}{2}\|Y\|_2^2 + \frac{1}{2}\|X\beta\|_2^2 - \langle Y, X\beta\rangle + |\langle Y, X\beta\rangle| \\
&\geq \frac{1}{2}\|Y\|_2^2 + \frac{1}{2}\|X\beta\|_2^2 \\
&\geq \frac{1}{2}\|Y\|_2^2 \\
&= \ell(0_p).
\end{aligned}$$

The first inequality follows by the assumption on λ. The second uses the result in (a). The third inequality uses the fact that $|x| \geq -x$ for any $x \in \mathbb{R}$, and the last inequality follows by the fact that norms are non-negative.

(c) In the previous part, we showed that $\hat{\beta} = 0_p$ is a minimizer of $\ell(\beta)$. Prove that $\hat{\beta}$ is the unique minimizer of $\ell(\beta)$, i.e. if $\beta \neq 0_p$, then $\ell(\beta) > \ell(0_p)$. **hint: consider the two cases:** $\|X\beta\|_2 = 0$ **and** $\|X\beta\|_2 > 0$

Solution:
Let $\beta \neq 0_p$. We first consider the case $\|X\beta\|_2 > 0$. we have

$$\begin{aligned}
\ell(\beta) &= \frac{1}{2}\|Y\|^2 + \frac{1}{2}\|X\beta\|_2^2 - \langle Y, X\beta\rangle + \lambda\|\beta\|_1 \\
&\geq \frac{1}{2}\|Y\|^2 + \frac{1}{2}\|X\beta\|_2^2 \\
&> \frac{1}{2}\|Y\|^2 \\
&= \ell(0_p).
\end{aligned}$$

In case 2, $\|X\beta\|_2 = 0 \implies X\beta = 0$, and so

$$\ell(\beta) = \frac{1}{2}\|Y\|^2 + \lambda\|\beta\|_1 > \frac{1}{2}\|Y\|^2$$

where the strict inequality holds since $\beta \neq 0_p$ by assumption.

What to submit: For all parts of the question, present your solution neatly - photos of handwritten work or using a tablet to write the answers is fine. Please include all working and circle your final answers. Note that if you cannot do part (a), you are still free to use the result of part (a) to complete parts (b) and (c).