Instructions

To do this assignment, save a copy to your own Google drive by going to:

> File --> Save a copy in Drive

Before working on this notebook, please disable inline AI suggestions by going to:

> Settings (top right gear icon) --> AI assistance --> Show AI-powered inline completions (uncheck)

As stated in the syllabus, you're welcome to use AI as an aid, but your answers should be your own.

Submission instructions:

- **Due**: Tuesday 10/21 at 11:59PM AOE
- **Submission**: Turn in both a printed **PDF** and the **editable share link** on MyCourses.

Remember that to make the share link editable, you must **convert the notebook to "anyone with a link can edit"** before copying and pasting the link.

Problem 1

You will derive the formula used to compute the solution to *ridge regression*. The objective in ridge regression is:

$$f(\beta) = \|y - A\beta\|_2^2 + \lambda\|\beta\|_2^2$$

Here, β is the vector of coefficients that we want to optimize, A is the design matrix, y is the target, and λ is the regularization coefficient. The notation $\|\cdot\|_2$ represents the Euclidean (or L_2) norm.

Our goal is to find β that solves:

$$\min_\beta f(\beta)$$

Follow the next steps to compute it.

Problem 1.1

Express the ridge regression objective $f(\beta)$ in terms of linear and quadratic terms. Recall that $\|\beta\|_2^2 = \beta^T\beta$. The result should be similar to the objective function of linear regression.

$f(\beta) = \| y - A\beta \|_2^2 + \lambda \| \beta\|_2^2$

$= (y - A\beta)^T(y - A\beta) + \lambda \cdot \beta^T\beta$

$= y^Ty - 2y^TA\beta + \beta^T(A^TA + \lambda I)\beta$

Problem 1.2

Derive the gradient: $\nabla_\beta f(\beta)$ using the linear and quadratic terms above.

$$\nabla_\beta f(\beta) = \nabla_\beta(y^T y - 2y^T A\beta + \beta^T(A^T X + \lambda I)\beta)$$
$$= \nabla_\beta(-2y^T A\beta + \beta^T(A^T X + \lambda I)\beta)$$
$$= -2A^T y + (A^A X + \lambda I)\beta$$

Problem 1.3

Since f is convex, its minimal value is attained when

$$\nabla_\beta f(\beta) = 0$$

Derive the expression for the β that satisfies the inequality above. You can adapt the derivation of the similar formula for linear regression from the slides.

$$-2A^T y + (A^T X + \lambda I)\beta = 0$$
$$(A^T X + \lambda I)\beta = 2A^T y$$
$$\beta = (A^T X + \lambda I)^{-1} + 2A^T y$$

Problem 1.4

Implement the algorithm for computing β and use it on a small dataset of your choice. Do not forget about the intercept.

Hint: numpy.linalg.inv() will come in handy here

```
import numpy as np
X = np.array([[1.0, 2.0],
              [2.0, 0.0],
              [3.0, 1.0],
              [4.0, 3.0]])
X0 = np.hstack([np.ones((4, 1)), X]) #add a column of 1's for intercept
y = np.array([1.0, 2.5, 3.0, 5.0])
y0 = np.asarray(y, dtype=float).reshape(-1, 1)
n,p = X0.shape
lamb = 1.0
I = np.eye(p)
I[0, 0] = 0 #dont penalize the intercept
a = (X0.T @ X0) + (lamb*I)
b = X0.T @ y0
beta = np.linalg.solve(a, b)
print(beta)
```

```
[[0.1875]
 [1.    ]
 [0.125 ]]
```

Problem 1.5

Compare your solution with the Scikit-Learn implementation of ridge regression (or another standard implementation) using a small example. Are the results the same? Why yes, or no?

```
from sklearn.linear_model import Ridge

model = Ridge(alpha=1.0, fit_intercept=True)
model.fit(X,y)
print("Intercept:", model.intercept_)
print("Coefficients:", model.coef_)
```

```
Intercept: 0.18749999999999956
Coefficients: [1.    0.125]
```

Our results our very nearly identicle because I perform the same algorithm as scikit learn's implementation. At first I was getting different results because I was penalizing the intercept, but once I correctly cleared the (0,0) index of the identity matrix they lined up

Problem 2

You will now derive the Bayesian connection to the lasso as discussed in Section 6.2.2. of ISL.

Problem 2.1

Suppose that $y_i = \beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j + \epsilon_i$ where $\epsilon_1, \ldots, \epsilon_n$ are independent and identically distributed from a normal distribution $\mathcal{N}(0,1)$. Write out the likelihood for the data as a function of values β.

$$L(\beta) = \mathbb{P}(y|\beta) = \frac{1}{\sqrt{2\pi}} exp(-\frac{1}{2}(y_i - \beta x_i)^2)$$

Problem 2.2

Assume that the prior for $\beta : \beta_1, \ldots, \beta_p$ is that they are independent and identically distributed according to the *Laplace* distribution with mean zero and variance c. Write out the posterior for β in this setting using Bayes theorem.

By Bayes Theorem the Posterior is:

$$\mathbb{P}(\beta|y) = \frac{\mathbb{P}(y|\beta)\mathbb{P}(\beta)}{\mathbb{P}(y)} \propto \mathbb{P}(y|\beta)\mathbb{P}(y)$$

and the Laplace Prior for β values is:

$$\mathbb{P}(y) = g(\beta) = \frac{1}{2} exp(\frac{|\beta|}{1})$$

So:

$$\mathbb{P}(\beta|y) = \frac{1}{\sqrt{2\pi}} exp(-\frac{1}{2}(y_i - \beta x_i)^2)\frac{g(\beta)}{\mathbb{P}(y)} \propto \mathbb{P}(\beta|y)\mathbb{P}(y)$$

Problem 2.3

Argue that the lasso estimate is the value of β with maximal probability under this posterior distribution. Compute $\log$ of the probability in order to make this point. *Hint*: The denominator (= the probability of data) can be ignored when computing the maximum probability.

$$log\mathbb{P}(\beta|y) = log\mathbb{P}(y|\beta_) + log\mathbb{P}(y) + C$$

$$= \sum_{i=1}^{n} log(\frac{1}{\sqrt{2\pi}} exp(-\frac{1}{2}(y_i - \beta x_i)^2)) + \sum_{j=1}^{p} log(\frac{1}{2} exp(|\beta|)$$

$$= -\frac{1}{2}||y_i - X\beta||_2^2 + \sum_{j=1}^{p} |\beta_j| + C$$