Machine Learning
Homework 4 : Regression - II
(due Midnight Feb 18)

Instructions

1. In case you are unfamiliar with the Python data ecosystem (NumPy, Pandas), you are recommended to study the first four chapters of the Python data science handbook. A doubt clearing session would be organised in case you have any difficulties in the data science ecosystem.
2. The deadline for full score is Midnight Feb 18. You can get 50% credit for late submission (Midnight Feb 20).
3. Total marks = 20
4. You have to type the assignment using a word processing engine, create a pdf and upload on the form. Please note that only pdf files will be accepted.
5. All code/Jupyter notebooks must be put up as **secret gists** and linked in the created pdf submission. Again, only secret gists. Not public ones.
6. Any instances of cheating/plagiarism will not be tolerated at all.
7. Cite all the pertinent references in IEEE format.
8. The least count of grading would be 0.5 marks.

1. Learn $y = \theta_0 + \theta_1 \times x$ on the following small dataset on pen and paper. You may scan or click picture of your answers and attach to the pdf.

$$X = \begin{bmatrix} 1 \\ 3 \\ 6 \end{bmatrix}$$

and

$$Y = \begin{bmatrix} 6 \\ 10 \\ 16 \end{bmatrix}$$

using:

   (a) Coordinate descent where initial values of (θ_0, θ_1) is (0, 0). Show the calculations for first 3 iterations. **[1 mark]**

   (b) Stochastic Gradient desent where initial values of $\theta_0, \theta_1)$ is (0, 0) and step size (or learning rate) is $\alpha = 0.01$. Show the calculations for initial 1 epochs (or 1*3 iterations). **[1 mark]**

   (c) Normal equation for ridge regression with penalizing coeffient $\lambda = 1$ **[1 mark]**

2. In this question, you will be writing your custom linear regression implementations.

   (a) Write a function `normalEquationRidgeRegression(X, y, λ)` where X is our feature matrix containing N samples (rows) and d features (columns) and y is our output vector containing N samples. λ is the penalty coefficient. This function returns a vector θ containing $d + 1$ rows. You are free to use numpy's matrix inverse, determinant and multiplication routines. **[1 mark]**

(b) Write a function `coodrdinateDescentRegression(X, y)` where X is our feature matrix containing N samples (rows) and d features (columns) and y is our output vector containing N samples. This function returns a vector θ containing $d+1$ rows. Please note this is the unregularised linear regression. **[2 marks]**

(c) Write a function `coodrdinateDescentLasso(X, y, λ)` where X is our feature matrix containing N samples (rows) and d features (columns) and y is our output vector containing N samples. λ is the penalty coefficient for the ℓ_1 regularisation. This function returns a vector θ containing $d+1$ rows. **[2 marks]**

(d) Write a function `sgdRegression(X, y, alpha = 0.1)` to learn the regression coefficients using stochastic gradient descent. You have to write the formulae for gradient wrt the different $\theta_j \forall j \in (1, ..d)$ **[1 mark]**

(e) Write a function `gradientDescentAutogradLasso(X, y, alpha = 0.1, λ)` to learn the regression coefficients for LASSO using gradient descent. Instead of writing the formulae for computing gradients by yourself, you will use Autograd to automatically do that for you. Gradients for $|\theta|$ are not defined, do you still get the correct solution? **[1 mark]**.

3. The following question is to aid our understanding of gradient descent variants. We will be reusing the data from Q1.

(a) Create a Matplotlib animation where the plot contains two columns: the first one being the contour plot and the second one being the linear regression fit on the data. The different frames in the animation correspond to different iterations of stochastic gradient descent applied on the dataset to learn θ_0 and θ_1. For each iteration, draw the current value of θ_0 and θ_1 on the contour plot and also an arrow to the next θ_0 and θ_1 as learnt by gradient update rule. Correspondingly draw the $y = \theta_0 + \theta_1 \times x$ line on the other subplot showing the scatter plot. The overall title of the plot show be the iteration number and the residual sum of squares. **[1 mark]**

(b) Do the same as part a, but using coordinate descent **[1 mark]**

4. (a) For the following X and y, use scikit-learn to learn a ℓ_2 regularised linear model. **[1 mark]**

(b) In the previous assignment, when you tried solvign the problem using normal equations, you find that one of the matrix in the normal equation was non-invertible. If you use the normal equations for Ridge regression, with say $\lambda = 1$, can you now learn the coefficients? If yes, calculate the coefficients. **[1 mark]**

$$X = \begin{bmatrix} 1 & 2 \\ 2 & 4 \\ 3 & 6 \\ 4 & 8 \end{bmatrix}$$

and

$$Y = \begin{bmatrix} 2 \\ 3 \\ 4 \\ 5 \end{bmatrix}$$

5. Show the usage of scikit learn's LASSO and Ridge module for the real estate price prediction regression problem. First, you may want to normalise the features between 0 and 1. You may also want to ensure that when you're predicting, you normalise the input features of the test set using the same function used to transform the train input features.

(a) Using 5-fold cross-validation report the optimal penalty coefficient for each fold, alongside, Train, Validation and Test RMSE error for Ridge regression. **[1 mark]**

(b) Using 5-fold cross-validation report the optimal penalty coefficient for each fold, alongside, Train, Validation and Test RMSE error for LASSO. **[1 mark]**

(c) Draw the regularisation path for LASSO and Ridge regression for the different variables. What does this tell you about sparsity of the solution? **[2 marks]**

(d) Let us pick up a single fold (train on first 80% data and last 20% as test data) and plot train, test error as a function of λ for Rdige and LASSO.**[2 marks]**

Questions below will not contribute to your score. They are presented here for the interested reader

6. Create a Jupyter widget/Matplolitb animation to draw the contour plot for Ridge and LASSO as a function of λ. Show the changing contour as λ is varied.

7. Plot the different ℓ_p norms for p less than 1. What value of p will give highest sparsity? Whay don't we use this norm for sparse solutions? Can you use Autograd with this norm and still obtain a good sparse solution?
8. Use statsmodel to create a linear fit. What other numbers does the fit show you beyond what you have seen thus far. What is the meaning of those numbers?
9. Study about the assumptions in linear regression. We studied one such condition through the previous assignment that the residuals should be normally distributed. What other conditions must hold?
10. Read about Poisson Regression. What you have studied in the lectures lays the foundations for most advanced techniques.
11. What is the other connotation/meaning of regularisation beyond prevent overfitting? What are we trying to make more regular?