New York University Tandon School of Engineering
Computer Science and Engineering

CS-UY 4563: Written Homework 2.
Due Tuesday, February 18th, 2020, 11:59pm.

Collaboration is allowed on this problem set, but solutions must be written-up individually. Please list the names of any collaborators at the top of your solution set, or write "No Collaborators" if you worked alone.

Problem 1: Practice Framing a Multiple Regression Problem (10pts)

An online retailer like Amazon wants to determine which products to promote based on reviews. They only want to promote products that are likely to sell. For each product, they have past sales as well as reviews. The reviews have both a numeric score (from 1 to 5) and text.

(a) To formulate this as a machine learning problem, suggest a target variable that the online retailer could use.

(b) For the predictors of the target variable, a data scientist suggests to combine each review's numeric score with the frequency of occurrence of words that convey judgement like "bad", "good", "broke", or "bargain". The data scientist would like to use these features in a linear model.

Describe one logical way to build feature vectors for each product. Think about normalization here!

(c) Suppose that some reviews have a numeric score from 1 to 5 and others have a score from 1 to 10. How would change your features?

(d) Now suppose the reviews have either: a score from 1 to 5; a rating that is simply good or bad; or no numeric rating at all. How would you change your features?

Problem 2: Thinking About Data Transformations (10pts)

You are trying to fit a multiple linear regression model for a given data set. You have already transformed your data by appending a column of all ones, which resulted in a final data matrix:

$$X = \begin{bmatrix} 1 & x_{1,1} & x_{1,2} & \dots & x_{1,d} \\ 1 & x_{2,1} & x_{2,2} & \dots & x_{2,d} \\ \vdots & \vdots & & \vdots & \\ 1 & x_{n,1} & x_{n,2} & \dots & x_{n,d} \end{bmatrix}$$

However, your model does not seem to be working well. It obtains poor loss in both training and test.

(a) A friend suggests that you should try mean centering your data columns. In other words, for each i, compute the column mean $\bar{x}_i = \frac{1}{n}\sum_{j=1}^{n} x_{i,j}$ and subtract $\bar{x}_i$ from every entry in column i. Note that we won't mean center the first column, as doing so would set the 1s to 0s. Using Python broadcasting you might mean center by running:

```
T = X[:,1:]
X[:,1:] = T - np.mean(T,axis=0)
```

Do you expect your friend's suggestion to improve the performance of the linear model. Will it help in all cases? Some cases? No cases?

(b) Another friend suggests normalizing your data columns to have unit standard deviation. In other words for each i, compute the column standard deviation $\sigma_i = \sqrt{\frac{1}{n}\sum_{j=1}^{n}(x_{i,j} - \bar{x}_i)^2}$ and *divide* every column by σ_i. Using Python broadcasting you might run:

```
T = X[:,1:]
X[:,1:] = T/np.std(T,axis=0)
```

Do you expect your friends suggestion to improve the performance of the linear model. Will it help in all cases? Some cases? No cases?

(c) Would your answers to either of the two questions above change if you were fitting the model with ℓ_2 regularization? In other words, instead of minimizing the squared loss $L(\boldsymbol{\beta}) = \|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2$ alone, you were minimizing $L(\boldsymbol{\beta}) + \lambda\|\boldsymbol{\beta}\|_2^2$.

Problem 3: Practice With Gradients (10pts)

For $\mathbf{X} \in \mathbb{R}^{n\times d}$ and target vector $\mathbf{y} \in \mathbb{R}^n$, consider fitting a linear model of the form:

$$f_{\boldsymbol{\beta}}(\mathbf{x}) = \mathbf{X}\boldsymbol{\beta}$$

under the so-called ℓ_p loss: $L_p(\boldsymbol{\beta}) = \|\mathbf{y} - f_{\boldsymbol{\beta}}(\mathbf{x})\|_p^p$. Here $\|\cdot\|_p^p$ denotes the ℓ_p norm raised to the p power. I.e. for any even integer $p = 2, 4, 6, \ldots$

$$\|\mathbf{z}\|_p^p = \sum_{i=1}^{n} z_i^p$$

(a) Derive an expression for $\nabla g(\mathbf{z})$ where $g(\mathbf{z}) = \|\mathbf{z}\|_p^p$.

(b) Derive an expression for $\nabla L_p(\boldsymbol{\beta})$. **Hint:** Use chain rule.

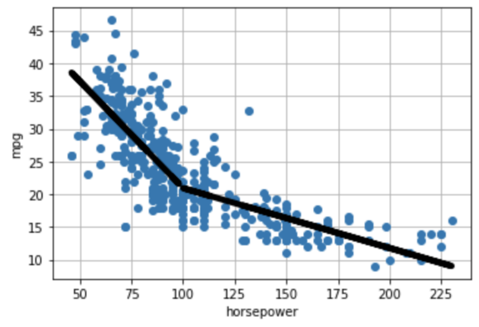
Problem 4: Piecewise Linear Regression via Feature Transformations (15pts)

Your goal is to fit a *piecewise* linear model to a single variate dataset of the form $(x_1, y_1), \ldots, (x_n, y_n)$ where all values are scalars. We will only use two pieces. In other words, for some known value λ,

$$f(x_i) = \begin{cases} a_1 + s_1 x_i & \text{for } x_i < \lambda \\ a_2 + s_2 x_i & \text{for } x_i \geq \lambda \end{cases}$$

with the additional **constraint** that $a_1 + s_1\lambda = a_2 + s_2\lambda$. This constraint ensures that our two linear models actually "meet" at $x = \lambda$, which means we get a continuous prediction function.

For example, when $\lambda = 100$, a piecewise linear fit for our MPG data might look like:



(a) Show that this model is equivalent to the following **unconstrained** model:

$$f(x_i) = \begin{cases} a_1 + s_1 x_i & \text{for } x_i < \lambda \\ a_1 + s_1\lambda - s_2\lambda + s_2 x_i & \text{for } x_i \geq \lambda \end{cases}$$

(b) Show how to fit an optimal f under the squared loss using an algorithm for multiple linear regression. In particular, your approach should:

- Transform the input data to form a data matrix $\mathbf{X}$ with multiple columns.
- Use a multiple regression algorithm to find the $\boldsymbol{\beta}$ which minimizes $\|\mathbf{y} - \mathbf{X}\boldsymbol{\beta}\|_2^2$.

- Extract from the optimal $\boldsymbol{\beta}$ optimal values for a_1, s_1, s_2.

You need to describe 1) a correct data transformation and 2) a correct mapping from $\boldsymbol{\beta}$ to a_1, s_1, s_2. **Note that in our model λ is known. It is not a model parameter which needs to be optimized.**

(c) Implement your algorithm in Python and apply it to the dataset from `demo_auto_mpg.ipynb`. Produce a piecewise linear fit for MPG as a function of Horsepower using the value $\lambda = 100$. Plot the result. You can attach a Jupyter notebook to your submission, or simply include the printed code and plot.

(d) (**5pts bonus**) Modify your approach to handle the case when λ is unknown. Again obtain a fit for MPG vs. horsepower. What value of λ gives the optimal fit? Include any modified code and a plot of your result.