DSC 140A - Homework 02

Due: Wednesday, April 17

Instructions: Write your solutions to the following problems either by typing them or handwriting them on another piece of paper. Show your work or provide justification unless otherwise noted. If you write code to solve a problem, include the code by copy/pasting or as a screenshot. You may use `numpy`, `matplotlib` (or another plotting library), and any standard library module, but no other third-party libraries unless specified. Submit homeworks via Gradescope by 11:59 PM.

Problem 1.

Let $(\vec{x}^{(1)}, y_1), \ldots, (\vec{x}^{(n)}, y_n)$ be a set of n training examples, where $\vec{x}^{(i)} \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$

Recall that the mean squared error of a linear predictor is defined to be

$$R(\vec{w}) = \frac{1}{n} \sum_{i=1}^{n} (\text{Aug}(\vec{x}^{(i)}) \cdot \vec{w} - y_i)^2.$$

In lecture, we saw two equivalent expressions for the gradient of $R(\vec{w})$:

$$\frac{dR}{d\vec{w}} = \frac{2}{n} \sum_{i=1}^{n} (\text{Aug}(\vec{x}^{(i)}) \cdot \vec{w} - y_i) \, \text{Aug}(\vec{x}^{(i)}),$$

and, in matrix-vector form:

$$\frac{dR}{d\vec{w}} = \frac{2}{n} X^T (X\vec{w} - \vec{y}),$$

where X is the $n \times (d+1)$ *design matrix* whose ith row is $\text{Aug}(\vec{x}^{(i)})$, and $\vec{y}$ is the $n \times 1$ vector whose ith entry is y_i.

Show that these two expressions are equivalent.

Problem 2.

Let $f(\vec{z}) = e^{z_1} + e^{z_2} + e^{z_3} + (z_1 - 1)^2 + \|\vec{z}\|^2$ be a function of $\vec{z} = (z_1, z_2, z_3)^T$.

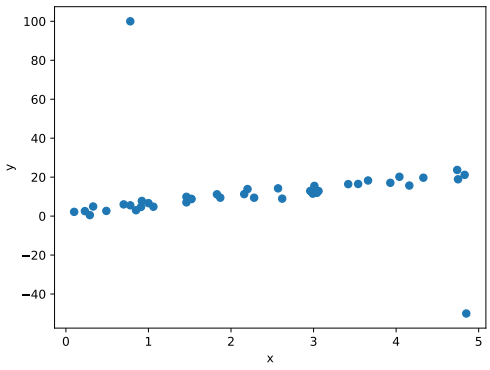
Find a minimizer of this function using gradient descent (implemented with code; don't run GD by hand!). Report:

- The initial point, $\vec{z}^{(0)}$;
- your choice of step size;
- the stopping criterion you chose and number of iterations taken to reach the criterion;
- the minimizer you found;
- the minimum value of the function.

Include your code.

Problem 3.

In a previous homework, you performed least squares regression on the data set shown below:



This data set can be downloaded again at the following link:

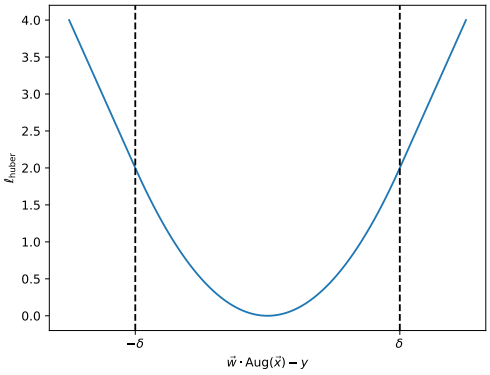
`https://f000.backblazeb2.com/file/jeldridge-data/002-regression_outlier/data.csv`

As you can see, the data contains outliers which may affect our regression. In the earlier homework, you should have found that least squares regression is not robust to these outliers.

The *Huber loss* is a loss function used in regression that is less sensitive to outliers than the square loss. It can be thought of as a "mix" between the square loss and the absolute loss. For linear prediction functions $H(\vec{x}) = \vec{w} \cdot \text{Aug}(\vec{x})$, the Huber loss is defined as:

$$\ell_{\text{huber}}(\text{Aug}(x) \cdot \vec{w}, y) = \begin{cases} \frac{1}{2}(\text{Aug}(x) \cdot \vec{w} - y)^2, & \text{if } |\,\text{Aug}(x) \cdot \vec{w} - y| \leq \delta, \\ \delta(\text{Aug}(x) \cdot \vec{w} - y) - \frac{1}{2}\delta^2, & \text{if } \text{Aug}(x) \cdot \vec{w} - y > \delta, \\ -\delta(\text{Aug}(x) \cdot \vec{w} - y) - \frac{1}{2}\delta^2, & \text{if } \text{Aug}(x) \cdot \vec{w} - y < -\delta, \end{cases}$$

A plot of the Huber loss is shown below.



Note that, despite being a piecewise function, the Huber loss is differentiable (unlike the absolute loss). However, there is no closed-form solution for the minimizer of the empirical risk with respect to the Huber loss, so if we want to train a regression model using this loss we need to use an iterative optimization algorithm, like gradient descent.

a) Write a function to compute the risk of a linear prediction function $w_0 + w_1 x$ with respect to the Huber loss (with $\delta = 1$) for the data given above. Use your function to compute the risk for $\vec{w} = (20, -1)$. Report the risk, and include your code.

By running `huber_risk([20, -1])`, we find that the risk of the linear prediction function $20 - x$ with respect to the Huber loss is approximately 11.085.

b) The gradient of the Huber loss with respect to $\vec{w}$ is also a piecewise function. Compute this gradient.

Note that since the Huber loss is differentiable, the gradient can be computed by separately computing the gradient within each piece of the piecewise function. You may use any of the matrix-vector calculus rules we've seen in class to help you compute the gradient. For example, you may use the fact that $\frac{d}{d\vec{w}}\vec{w} \cdot \text{Aug}(\vec{x}) = \text{Aug}(\vec{x})$.

c) Write a function that computes the gradient of the empirical risk with respect to the Huber loss with $\delta = 1$ for a linear prediction function $w_0 + w_1 x$. Use your function to compute the gradient for $\vec{w} = (20, -1)$. Show your code.

d) Implement and run gradient descent to minimize the empirical risk with respect to the Huber loss (using $\delta = 1$) on the data set above. In addition to your code, include:

- The optimal solution found by gradient descent;
- A plot of the empirical risk of $\vec{w}^{(t)}$ at each iteration. In other words, include a plot whose horizontal axis is iteration number, t, and whose vertical axis measures the empirical risk of $\vec{w}^{(t)}$.
- A plot of the data and the linear prediction function corresponding to the optimal solution.

e) Implement and run *stochastic* gradient descent with a batch size of 8 to minimize the empirical risk with respect to the Huber loss (using $\delta = 1$) on the data set above.

Include:

- The optimal solution found by SGD;
- A plot of the empirical risk of $\vec{w}^{(t)}$ at each iteration. In other words, include a plot whose horizontal axis is iteration number, t, and whose vertical axis measures the empirical risk of $\vec{w}^{(t)}$.

Note that you should plot the risk with respect to the whole data set, and not the risk with respect to the random batch (the latter will be very noisy, and it's the former that we're trying to optimize).