Programming Assignment 3: Candy Store of Regressions and Gradients

Due November 10, 2023

The purpose of this assignment is to instantiate gradient descent as a permanent and ineradicable element of your consciousness and memory. For this assignment you will develop code to reproduce figures below; the first depicts a linear regression model and the second logistic regression.

This assignment will give you insight into the general structure of machine learning pipelines in code. Modify code by filling in methods; you should not need to define new methods or functions. The starter code has comments to indicate which methods should be modified, though no hints are provided on the order for completing the code.

Notice there three classes: 'MLModel,' 'LinReg,' and 'LogReg,' the latter two of which *inherit* from MLModel. MLModel is a base class which implements the fit() method. You may see various other methods in this base class, but they are not defined here; you will write their functionality in LinReg and LogReg.

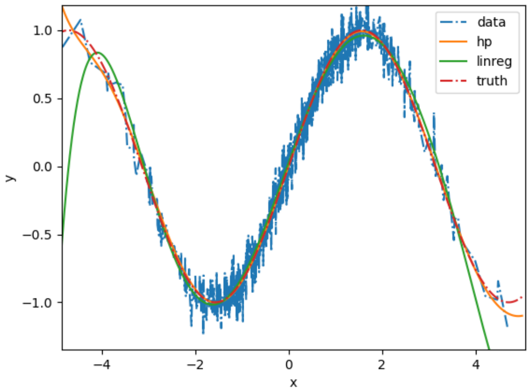
As the name suggests, LogReg implements logistic regression for classification. You will implement two versions of this, one with the negative log likelihood loss function, and the other with the mean squared error. Your metric in metrics() method should be accuracy, i.e. $\mathbb{P}_{\mathcal{X}\times\mathcal{Y}}(\tilde{y}(x) = y)$. Recall from class that classification has two steps: build a "score" function, call it $\bar{y} : \mathcal{X} \to (0, 1)$, and then threshold $\tilde{y}(x) = \mathbb{1}_{\bar{y}(x)\geq t}$ for some (arbitrarily?) chosen t. Let $t = 1/2$ for this assignment.

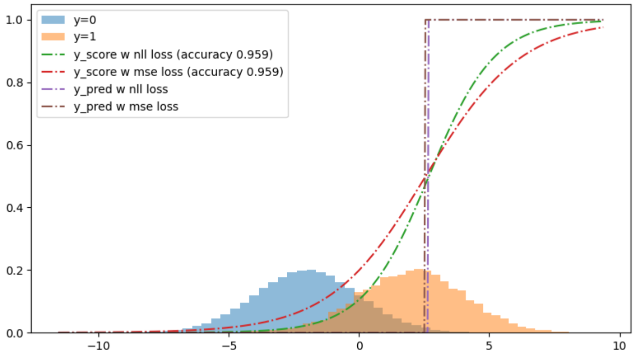
LinReg implements linear regression. You already know how to implement linear regression with Hilbert Progression, and this code includes its implementation. You are to additionally implement linear regression using gradient descent. You may start with a low degree model, but your code should work for a higher degree one (you may need to use a computational detail from class). You may use your loss function for metrics() in this class.

Implementation of gradients for logistic regression has been completed in the last worksheet. You should triple check your computation of the gradient, and also triple check its implementation in code. As the computational pipeline for gradient descent is fairly light, problems may likely concentrate here. Also be careful with signs. And most importantly, have fun!

Deliverables:

1. code as .py file; you do not need to delete the main function,
2. plots replicating the figures below (will not be exact replicate),
3. your coefficients from linear regression for model of degree at least 9 (in canvas submission comments),
4. an explanation for why the decision boundary in the second figure is not at zero (in canvas submission comments).


data
hp
linreg
truth
y
x


y=0
y=1
y_score w nll loss (accuracy 0.959)
y_score w mse loss (accuracy 0.959)
y_pred w nll loss
y_pred w mse loss