Project 6: Training Models

Owen O'Reilly | 12/1/2024 | AMS-595

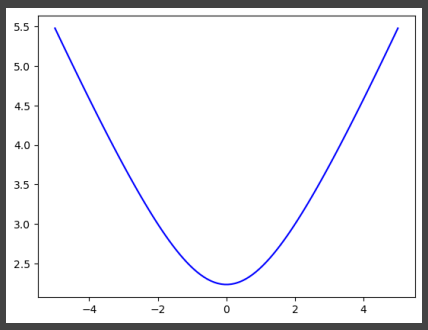
Summary

This project was divided into four sections: gradient descent, linear regression, logistic regression, and binary classification of images

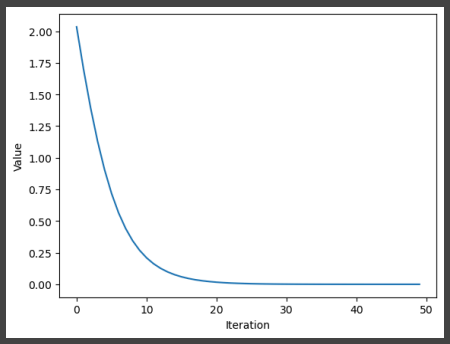
Script 1: Gradient Descent

The goal of this was to iteratively improve the "guess" for the minimum of a defined 2D function ($\sqrt{x^2 + 5}$) based on its derivative for a set number of iterations. This was accomplished in three steps:

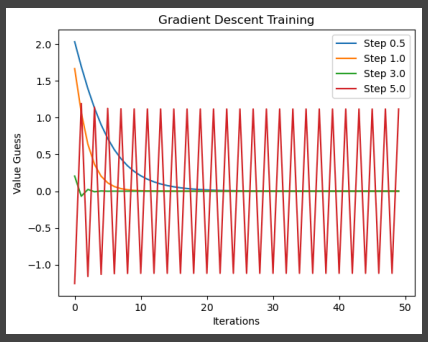
First, the function itself was defined as a lambda function and plotted.



Second, the derivative of the function was defined, and a `line_descent` method created that would conduct gradient descent with a defined step size and number of iterations. A test run was conducted using a step size of 0.5, and the error at each step was plotted.



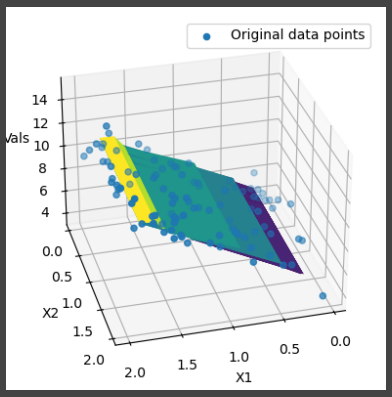
Finally, the performance of the method with step sizes of 0.5, 1, 3, and 5 were compared, and the error plotted.



The results are explored in the **Discussion**.

Script 2: Linear Regression of 3D Data

The goal of this script is to generate a "plane of best fit" based on a set of 3D data points using a linear regression algorithm. The script itself generates some random data points, divided into (X1,X2) coordinates and associated Y "heights." A best-fit plane "Θ" is then found in one step using the normal equation. A set of prediction points are generated using this new plane, and plotted alongside the original data. The parameters of Θ are also returned.



Script 3: Logistic Regression

This script was actually provided in its entirety; the goal was to find the learning rate and # epochs that produced the most accurate logistic regression. It was also requested that we comment the code meaningfully. As such, results will be presented in the **Discussion** section.

Script 4: Binary Image Classification: Cat or Not Cat

The goal of this section was to write a neural network that could be trained on 64x64 pixel labeled images that do or do not contain a cat. Once trained, the network should be able to predict whether an image contains a cat with greater-than-random accuracy. The instructions indicated that this should be written from scratch based on in-class activities; however, this problem was not covered in class, and as such I took the liberty of exploring the Keras library for NNs to solve it. I will accept any deductions as a result of this. Two models were created; one was trained on a larger dataset than the other in order to investigate the effect this would have on model prediction accuracy.

Discussion

Gradient Descent

The step sizes of 0.5, 1.0, and 3.0 converged appropriately, but the step size of 5.0 showed oscillation that indicated each step was too drastic of a correction, prohibiting good convergence. 0.5 converged most smoothly, as might be anticipated, but 1.0 appeared to have the best balance of smoothness and rapidity of convergence. 3.0 overshot slightly but quickly corrected to almost 0 error well before any other method.

Linear Regression

The regression was successful in finding a plane that appears to be close to if not absolute best fit for the given data points. The intercept value is ~3.84, and coefficients are ~[0.4115, 4.213]. Comments were added to the file to clarify what occurs in different segments of the code.

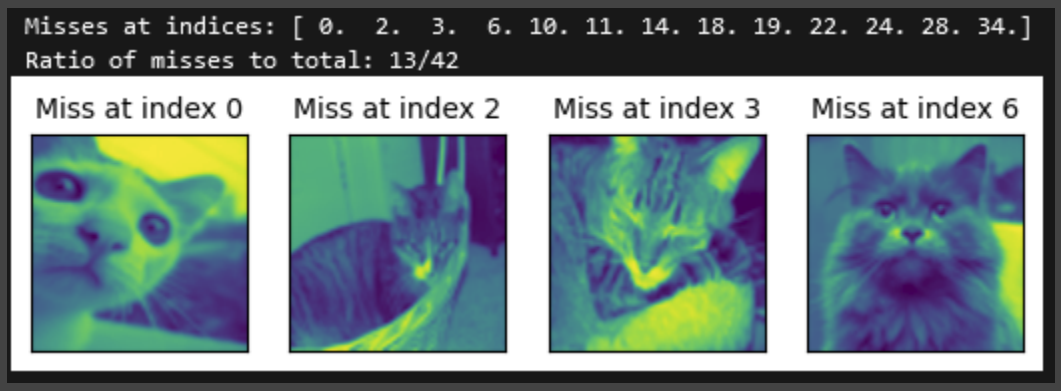
Logistic Regression

Using a learning rate (LR) of 20 and a # epochs (NE) of 100000 produced an accuracy of 91.25% on the training set and 100% on the test set; this appeared to be a minima, as any increase/decrease to the NE or LR produced lowered accuracy on one or both training sets. The large number of epochs makes sense, as it allows for many iterations of refinement to occur, but the learning rate seems exceptionally high compared to what we experimented with in our lectures. A LR of 0.001 with the same number of epochs did provide a test set accuracy of 100% as well, but the training set accuracy decreased to 87.5%. This may be an artifact of the data being analyzed; a different combination of epochs and a lower learning rate could yield another local minima.

Binary Classification

The models produced by the Keras-based neural network trained herein perform adequately at the cat recognition task. The best model trained on the full dataset had an accuracy of 74%, and seemed to miss mainly images not containing cats. Images of the first 4 misses indicate that there may be issues in recognition when the cat fills the frame, either with its entire face or body as well. This may indicate a reliance on silhouette by the model.

The model trained on the subdivided training set had an accuracy of 69%, but in this case appeared to mainly have issues in recognizing cats.



The lowered accuracy rate is expected, as the binary classifier relies on using as much data as possible to find optimal coefficients for pattern recognition. Providing the model with more pictures in both the "cat" and "not cat" categories would allow for even greater model accuracy.

Overall, this set of exercises was a good introduction to how the theory we have covered in lecture can be applied in actual analysis, and demonstrated the predictive power of regression-based analysis.