Project update

Original Timeline

Nov 10th – completion of linear regression and ridge regression codes

Nov 17th – completion of least squares codes

Nov 24th – completion of neural network code

Dec 1st – completion of rough first draft of final project report (assuming no progress of generative model)

Dec 8th – completion of generative neural network code

Dec 10th – completion of final draft of final project report

Project Status

So far, my project is running on schedule, if not slightly ahead. My dataset is intractable for simple linear regression since a unique solution cannot be found. Therefore, I have abandoned non-regularized linear regression and replaced it with LASSO and I now have two functioning codes that solve a least-squares problem with LASSO regularization[1] and ridge regression[2].

For both LASSO and ridge regression, I employ a binary classifier for each class. An image is classified according to the class of the binary classifier amongst the ensemble that signals the best fit. I have investigated the impact of changing the tuning parameter for each method and I have also solved for both 0th-, 1st-, and 2nd- degree polynomial fitting.

I have conducted two trials. For the first trial, I used the first 10000 images of the MNIST's "training" set as my training data and then the next 10000 in MNIST's training set as testing. Then, I moved on to using the entire 60000 images from MNIST's "training" set as my training data, and using the entire 10000 images from MNIST "testing" as my testing data. The results of my scaled and extended trials are provided below, with ridge regression results provided in Fig. 1 and LASSO results provided in Fig. 2. Error rate is defined as it has been in previous assignments

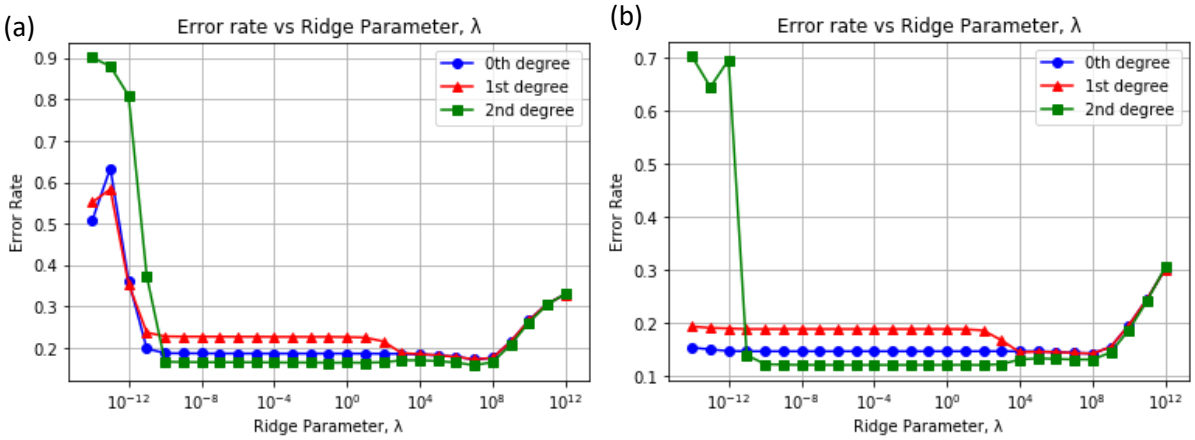

Fig 1. Error rate of least-squares ridge regression for 0^{th}-, 1^{st}-, and 2^{nd}- degree polynomial fitting of MNIST testing data for (a) scaled trails and (b) extended trials

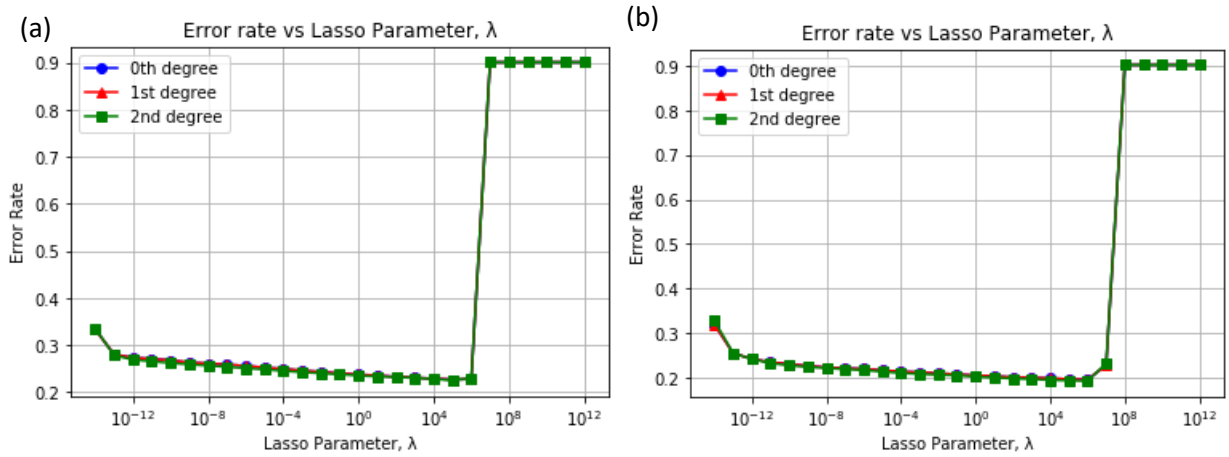

Fig 2. Error rate of least-squares LASSO for 0^{th}-, 1^{st}-, and 2^{nd}- degree polynomial fitting of MNIST testing data for (a) scaled trails and (b) extended trials

Preliminary Observations

So far there have been a number of noteworthy observations.

The dataset seems sufficiently large enough such that broad insights can be gained from either the full data set (60000 training samples) or from a fraction of the data set (10000 training samples). Given the consistency between both trials, it is unlikely that significant insights will be gained from detailed cross validation efforts for this particular data set. To get the most out of this project, I should focus my efforts elsewhere.

Of the two methods, ridge regression exhibits the most variability. Strangely, the 0^{th} degree fit outperforms the 1^{st} degree fit, which suggests over-fitting yet performance improves for the 2^{nd} degree fit. This may be the result of a bug. The minimum λ for which ridge regression works appears to be

dependent on the size of the training set. Meanwhile, at very large λ's, error rates between the different polynomial fits converge with each other before increasing in unison.

Unlike ridge regression, The LASSO method appears to be extremely consistent. Across both trials and across multiple degrees of fitting, LASSO method performance remains largely unchanged. A zoom-in of the differences in polynomial fits is provided in Fig. 3. Pessimistically, this means that adding polynomial weights and increasing training sample size are neither particularly effective means of improving performance. But optimistically, it means that a simple and quick approach is nearly as effective as a more-expansive, higher-polynomial approach.

Fig 3. Error rate of LASSO-regularized least-squares classification for 0th-, 1st-, and 2nd- degree polynomial fitting of testing data from extended trials. 0th and 1st degree fits are nearly identical while modest improvements are made for the 2nd-degree fit.

Preprocessing

Initially, I considered preprocessing my data by making the number of training images from each class equal, but in retrospect this seemed counterproductive since additional data should only be beneficial. Additionally, the initial performance of my codes suggests adequate performance can be achieved without preprocessing. Further work may involve repeating my process after applying a centering algorithm to the images. Beyond that, additional work may involve rotating the images by a random multiple of 90° or introducing slight, random shifts in pixel location before repeating the fitting process. This exercise could provide great insight into the importance of preprocessing.

Updated Timeline

I am interested in replacing squared error loss with a different loss function, such as hinge loss, and repeating this evaluation. I've made a slight adjustment with the updated timeline to include hinge loss. The only reason for this addition is that I think it'd be interesting to try hinge loss out. Similarly, I would like to experiment with the effects of preprocessing but for now I have omitted a full commitment of that step.

Nov 10th – completion of ridge regression code w/ least squares loss

Nov 17th – completion of lasso code w/ least squares loss

Nov 22nd – completion of lasso and ridge regression code w/ hinge loss

Nov 27th – completion of neural network code

Dec 1st – completion of rough first draft of final project report (assuming no progress of generative model)

Dec 8th – completion of generative neural network code

Dec 10th – completion of final draft of final project report

Github links

[1] https://github.com/jlking2/Neural_net_numbers/blob/main/binary_lasso_combo_4.py

[2] https://github.com/jlking2/Neural_net_numbers/blob/main/binary_Tik_combo_3.py