CSE 5523 Homework 4: Linear Models and SGD on Brain Image Data

Alan Ritter

In this assignment you will train linear models using stochastic gradient descent to predict whether a human subject is viewing a picture or reading a sentence from their fMRI brain image data.

NOTE: It may take some time to run the experiments for this assignment, so we recommend that you start early. We also recommend doing as much computation as possible using Numpy vector and matrix operations, this will be much faster than, for example writing loops to compute dot products in Python. You can find more details about Numpy online.[1]

Data

The data comes from the starplus dataset[2] (only the first subject). We provide starter code to read the provided matlab file into Python/Numpy arrays. The data for each trial consists of a sequence of brain images that we have flattened into a feature vector X. Your task is to train a linear classifier that outputs, $y \in \{+1, -1\}$, which correspond to whether the subject was first shown a picture or a sentence during the trial. More details on the dataset are included in the comments of the starter code and in the README file.[3]

[1] `https://docs.scipy.org/doc/numpy-dev/user/quickstart.html`

[2] `http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-81/www/`

[3] `http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-81/www/README-data-documentation.txt`

Stochastic Gradient Descent (6 points)

Recall the regularized logistic loss that we discussed in class (taken from Logistic Regression):

$$\mathcal{L}_{\log}(w) = \sum_l \log(1 + \exp(-y_l w \cdot x_l)) + \lambda \sum_i w_i^2$$

And also the Hinge-Loss from linear soft-margin SVMs:

$$\mathcal{L}_{\text{hinge}}(w) = \sum_l \max(0, 1 - y_l w \cdot x_l) + \lambda \sum_i w_i^2$$

Your task will be to implement stochastic gradient descent for each of these loss functions by completing the associated sections in the starter code. You can run the provided code like so:

```
> python linear_brain.py
```

(this assumes the data file is in the same directory)

First, for each loss, implement the associated functions in the starter code (e.g. `HingeLoss` and `LogisticLoss`). These functions should return the total loss for a dataset given a specific set of weights. This is useful for monitoring convergence during gradient descent.

Next, fill in the missing part of the associated gradient descent routines (`SgdHinge` and `SgdLogistic`).

Parameter Tuning (3 points)

After implementing the loss and gradients you are now ready to train your models. Because there are a relatively small number of trials available for each subject, we have provided you with code for hold-one-out cross validation.[4]

Your next task is to tune the learning rate, η, and regularization parameters λ for each loss on this dataset using held out development data. Use the first 20 trials, after (seeded) random permutation, as development data to tune η and λ. Try out several values including at least $\eta \in \{0.1, 0.01, 0.001, 0.0001\}$

[4]We use a seperate linear model to classify each datapoint that is trained on all of the data except that point.

and $\lambda \in \{1, 0.3, 0.1\}$ for each loss function and report acccuracy on the development data for each parameter combination that you try. Stop updating the gradient when the loss changes by less than 0.0001 or some number of maximum iterations (e.g. 100). Print out the loss at each iteration and turn in the output of your program in a separate file with your report.

Results on Test Data (2 points)

After tuning parameters of your algorithm on the development data (first 20 trials), evaluate the performance of your model on the remaining trials. Report the final accuracy for both loss functions.

Inspecting the Model's Parameters (2 Extra Credit Points)

The provided code reads in a data structure `rois` which defines "regions of interest" in the brain. XYZ coordinates (and also the coresponding column indices in the data) for each voxel that is the region of interest are provided.

Based on the parameters learned by your model, what brain regions seem to be most important for predicting whether the subject is viewing a picture or a sentence? Provide some justification for your answer. (hint: The learned weight vector corresponds to a sequence of brain images over time, so you will need to collapse this...)

What to Turn In

Turn in your code and a brief writeup presents the results requested above. You do not need to turn in the provided dataset.