Assignment 3
CSE 446: Machine Learning

University of Washington

Due: November 5, 2017

November 3, 2017: updates in red.

You will submit the following files or documents for this homework, compressed into a gzipped tarball named `A3.tgz`.

- Your **report** as a **pdf file** named `A3.pdf` containing answers to written questions. You must include the plots and explanation for programming questions (if required) in this document only. We *strongly* recommend typesetting your scientific writing using LaTeX. Some free tools that might help: ShareLaTeX (www.sharelatex.com), TexStudio (Windows), MacTex (Mac), TexMaker (cross-platform), and Detexify[2] (online). If you want to type, but don't know (and don't want to learn) LaTeX, consider using a markdown editor with real-time preview and equation editing (e.g., stackedit.io, marxi.co). Writing solutions by hand is fine, as long as they are neat; you will need to scan them into a single pdf.
  Part of the training we aim to give you in this advanced class includes practice with technical writing. Organize your report as neatly as possible, and articulate your thoughts as clearly as possible. We prefer quality over quantity. Do not flood the report with tangential information such as low-level documentation of your code that belongs in code comments or the README. Similarly, when discussing the experimental results, do not copy and paste the entire system output directly to the report. Instead, create tables and figures to organize the experimental results.
- **Python source code** for the programming assignment. Please note that we will not accept Jupyter notebooks. You will submit your code together with a neatly written README file to instruct how to run your code with different settings (if applicable). We assume that you always follow good practice of coding (commenting, structuring), and these factors are not central to your grade.

You are welcome to use any Python libraries for data munging, visualization, and numerical linear algebra. Examples includes Numpy, Pandas, and Matplotlib. You may **not**, however, use Python machine learning libraries such as Scikit-Learn or TensorFlow. If in doubt, email the instructors.

1 L_2-Regularized Logistic Regression [10 points]

Let the training data be $\langle\langle \mathbf{x}_1, y_1\rangle, \ldots, \langle \mathbf{x}_N, y_N\rangle\rangle$, where $y_n \in \{-1, +1\}$. Consider minimizing

$$F(\mathbf{w}, b) = \underbrace{\frac{1}{N}\sum_{n=1}^{N} \text{logLoss}(\mathbf{w}, b, \mathbf{x}_n, y_n)}_{\text{logLoss}_{\text{train}}(\mathbf{w},b)} + \lambda\|\mathbf{w}\|_2^2$$

Note that $\text{logLoss}_{\text{test}}(\mathbf{w}, b)$ is defined similarly to $\text{logLoss}_{\text{train}}(\mathbf{w}, b)$, but on the test data. State whether each of the following statements is true or false, and briefly explain.

(a) Let $(\hat{\mathbf{w}}, \hat{b}) = \arg\min_{\mathbf{w},b} F(\mathbf{w}, b)$ be a global optimum. $\hat{\mathbf{w}}$ is typically sparse (has many zero entries). [2 points]

(b) If the training data is linearly separable, then some weights $\mathbf{w}[j]$ might become infinite if $\lambda = 0$. [1 point]

(c) $\text{logLoss}_{\text{train}}(\hat{\mathbf{w}}, \hat{b})$ never decreases as we increase λ. [1 point]

(d) $\text{logLoss}_{\text{test}}(\hat{\mathbf{w}}, \hat{b})$ always increases as we increase λ. [1 point]

Now answer the following questions:

(e) Suppose we have only one feature. Now we create an alternative training dataset $\{\langle \mathbf{x'}_n, y'_n\rangle\}_{n=1}^N$, now with *two* features, so that:

$$\mathbf{x'}_n[1] = \mathbf{x'}_n[2] = \mathbf{x}_n[1]$$

That is, the new dataset is the same as the old one, but with two copies of the single feature. If we find $\hat{\mathbf{w}}[1]$ and $\hat{b}$ on the original data and $\hat{\mathbf{w}}'[1]$, $\hat{\mathbf{w}}'[2]$, and $\hat{b}'$ on the new dataset by solving L_2-regularized logistic regression (two separate times).

i. Write down the **unregularized** log-likelihood we want to maximize for each of the two logistic regressions. [2 points]

ii. Given the log-likelihood functions, what is the relationship between $\langle \hat{\mathbf{w}}[1], \hat{b}\rangle$ and $\langle \hat{\mathbf{w}}'[1], \hat{\mathbf{w}}'[2], \hat{b}'\rangle$? [2 points]

iii. Would your answer for the previous question change if we were using L_2 regularization? Argue why or why not. (Remember that we don't regularize the bias.) [1 point]

2 Implementation: Stochastic Gradient Descent [60 points]

In this problem, you will implement stochastic gradient descent (SGD) for both **linear regression** and **logistic regression** on the same dataset. The goal is to predict whether a patient has diabetes (label 1) or not (label –1). In clinical informatics, machine learning approaches have been widely adopted to predict clinically adverse events based on patient data. If you are interested in applications of machine learning to biomedicine and healthcare, take a look at the MIMIC II clinical database demo set at https://physionet.org/mimic2/demo/.

2.1 Dataset

For this problem, we will use the Pima Indians Diabetes Data Set: https://archive.ics.uci.edu/ml/datasets/pima+indians+diabetes. Please read the web page to gain an understanding of the data.

2.2 Model Training

Among all 768 patients, we will separate 500 patients as training data (`A3.train.csv`) and 268 patients as test data (`A3.test.csv`). All features are min-max normalized[1] based on the training dataset. Recall that SGD performs gradient descent using a noisy estimate of the full gradient based on just the current example.

(a) Write down the equation for the weight update steps for both linear regression and logistic regression. Include the bias update as well. Let $\mathbf{x}^{(k)}$ be the feature vector for the example randomly chosen on iteration k, and $y^{(k)}$ be the correct class. [15 points]

(b) Implement SGD for both linear regression and logistic regression. Reserve a random subset of the training data as a development set. Initialize the weight vector $\mathbf{w}$ and the bias b to zero. For each step size $\eta \in \{0.8, 0.001, 0.00001\}$, for both linear regression and logistic regression, and without regularization:

(i) Plot the average squared error as a function of the iteration k, where

$$\text{averageSquaredError}(k) = \frac{1}{k}\sum_{t=1}^{k}(\hat{y}^{(t)} - y^{(t)})^2$$

where $\hat{y}^{(t)}$ is the prediction of your model on the example chosen on SGD iteration t, with the weights and bias from that iteration. Note that the prediction function for linear regression is different from the prediction function for logistic regression. [10 points]

(ii) Plot the L_2 norm of the weights (ignore the bias) across iterations. (In general, L_2 regularization might be useful if the norm of these weights grows very large over the course of model training. In this assignment, however, you are *not* expected to perform L_2 regularization.) [10 points]

(iii) Use the learned model to predict whether each patient in the *test* set has diabetes. Your prediction should be in $\{-1, +1\}$, just like the data—this means you'll need to decide on a prediction rule for the linear regression model that maps continuous outputs to $\{-1, +1\}$. You should measure test-set accuracy. Plot test-set accuracy across iterations. [10 points]

For the plots above, you don't need to plot every single step k; do preliminary checks to select the plotting interval. That is, select a value I and plot the average squared error (or the L_2 norm of the weights or the test set accuracy) at every Ith iteration. You might need a different I value for (i), (ii), and (iii) and for linear vs. logistic regression to see interesting differences. We

[1] For more details about min-max normalization, refer to: http://sebastianraschka.com/Articles/2014_about_feature_scaling.html

recommend that you start with $I = 100$, but do some checks to see if the picture looks different for larger or smaller I. For each plot above, just choose one I.

Note that we are asking for $2 \times 3 \times 3 = 18$ separate plots (two loss functions, three η values, and three y-axis values). Please label each one clearly so that we aren't confused about which is which!

(c) After 100,000 iterations:

(i) Select the best linear regression model based on your development data, and report the weights of the following features: BMI, 2-hour serum insulin level, plasma glucose concentration, and the bias. Do the same for the best logistic regression model. [10 points]

(ii) Provide an interpretation of the effect of the features above for diabetes classification based on these inferred weights. Do the two models give the same finding? Discuss. [5 points]

3 Implementation: Gradient Descent [30 points]

For either linear regression or logistic regression, repeat the exercise in problem 2.2, but this time using gradient descent with a fixed step size. This time, we aren't telling you what step sizes to try. Using plots like the ones you made before, try to achieve performance that's as good as what you got using SGD. Use development data for hyperparameter tuning, like you did earlier.

For full credit, your discussion must include: (i) which model you focused on (linear or logistic regression); (ii) your final test-set accuracy; (iii) the step size you selected; (iv) how many iterations (i.e., passes over the training data) it took to get to the final model; (v) a comparison of the final model to the SGD-trained model. You may use up to two plots to help elucidate your findings.

4 Bonus [up to 10 extra points]

Many software libraries offer implementations of batch algorithms that, like gradient descent, only require calculation of your objective function's gradient (and perhaps the function's value), in order to find its minimum. Examples of such algorithms are L-BFGS and conjugate gradient. Repeat problem 3 (including the writeup) using such an algorithm, as implemented in an available library.