Coursework **Fall 2018**
Machine Learning (B555) **Roni Khardon**
Department of Computer Science, IU Bloomington

Programming Project 3

This assignment is due by Monday 11/5, 11:59pm via Canvas.

You can write your code in any programming language so long as we are able to test it on SICE servers. While we will not test all submissions we will select a subset of assignments to test.

Overview: Experiments with Classification Algorithms

In this programming project, we will be training linear classification models and assessing their performance on artificial and real datasets. Your goals in this assignment are to (1) compare the performance of the generative and discriminative models, and (2) compare Newton's method to gradient ascent.

Data

Data for this assignment is provided in a zip file `pp3data.zip` on Canvas. Each dataset is given in two files with the data in one and the labels in the other file. The datasets are as follows.

The first dataset (marked as A) has uniformly distributed data. Its labels were generated using some weight vector separating positive from negative examples. Therefore data does not conform to the Gaussian generative model but the data is linearly separable.

In the second dataset (marked as B) each class is generated from multiple Gaussians with differing covariance structure. This diverges even further from the generative model yet we designed it to be somewhat linearly separable.

The examples of the third dataset, USPS[1], represent 16×16 bitmaps of the characters 3 and 5. We use this dataset for the binary classification problem of distinguishing between these two characters.

Task 1: Generative vs. Discriminative

In this portion, you will implement and evaluate two algorithms for the binary classification problem. In particular, we will look at the following algorithms discussed in class:

- 2 class generative model with shared covariance matrix as developed in Section 4.2 of the textbook.

[1] http://www.gaussianprocess.org/gpml/data/

- Bayesian logistic regression where we use the simple prior $w \sim \mathcal{N}(0, \frac{1}{\alpha}I)$. In this case the update formula for w is: $w_{n+1} \leftarrow w_n - (\alpha I + \Phi^T R \Phi)^{-1}[\Phi^T(y-t) + \alpha w_n]$ where R is as in Eq (4.98) in the textbook. The predictive distribution is described in Section 4.5.2 of the textbook.

  Recall that, unlike the generative model, logistic regression relies on a free parameter (w_0) to capture an appropriate separating hyperplane. Therefore, you will need to add a feature fixed at one to the data for this algorithm. For the implementation, initialize w to the zero vector and set $\alpha = 0.1$ (unlike in previous projects, we won't be performing model selection here). Use the following test as a stopping criterion: $\frac{\|w_{n+1}-w_n\|_2}{\|w_n\|_2} < 10^{-3}$ or $n \geq 100$.

  To help you test your implementation of this algorithm we provide an additional dataset, *irlstest*, and solution weight vector in *irlsw*. The first entry in *irlsw* corresponds to w_0.

For both algorithms we classify a test example as positive iff $p(c = 1) \geq 1/2$.

Your task is to implement the 2 algorithms and generate learning curves as follows. For each dataset (A, B, USPS) evaluate each algorithm as follows: Step 1) Set aside 1/3 of the total data (randomly selected) to use as a test set. Step 2) Record the test set error rate as a function of increasing training set size (with each training set randomly selected from the other 2/3 of the total data). Repeat Steps 1 & 2 a total of 30 times to generate learning curves with error bars (i.e., $\pm 1\sigma$).

In your submission plot these results and discuss them: how do the algorithms perform on these datasets? are there systematic differences? and how do the differences depend on training set size?

Task 2: Newton's method vs. gradient ascent

We have introduced Newton's method in order to improve convergence of gradient based optimization. However, the updates in Newton's method can be expensive when the computation of the inverse Hessian is demanding. In general gradient ascent can perform many updates for the same time cost of one update of Newton's method. In this task we compare the two methods for the quality of the weight vector they produce as a function of time.

Recall that the update of gradient ascent is given by $w_{n+1} \leftarrow w_n - \eta[\Phi^T(y-t) + \alpha w_n]$. Gradient ascent is sensitive to the choice of learning rate η. For this part you should use $\eta = 10^{-3}$ (which we have verified to work reasonably well).

Our goal is to compare the two methods for the quality of the weight vector they produce as a function of time. To achieve this we need intermediate values of w as well as time stamps for when that value is produced. In your implementation you should store the value of w as well as the wall clock time after each update in Newton's method and every 10 iterations for gradient ascent. Once training is done you can evaluate the test set error of each of the w vectors produced. In this way the evaluation time (which can be costly) does not affect the time stamp of the learned vectors. Finally you can plot the test set error rate of the algorithm as a function of run time.

Please perform this evaluation on datasets A and USPS. To make sure we have stable results across submissions - please use the first 1/3 of the dataset for testing and the rest for training the algorithms. Since estimates of run time from your computing environment can be noisy please repeat the above 3 times (on the same 1/3, 2/3 partition of the data) and average the times across these runs. Note that the w vectors will be identical because the data is the same and their

evaluation does not need to be repeated.

The above descsription does not require a stopping criterion. However, to control run time in your experiments run them with the same stopping criterion for w, that is, $\frac{\|w_{n+1}-w_n\|_2}{\|w_n\|_2} < 10^{-3}$ and with a bound on the number of iteration stopping if $n \geq 100$ iterations for Newton's method and $n \geq 6000$ for gradient ascent.

In your submission plot these results and discuss them: how do the algorithms perform on these datasets? are there systematic differences? and how do the differences depend on data set characteristics?

Extra Credit (up to 20 points)

Write code and runs experiments with *one* of the following 3 variants: (1) Implement *line search* and evaluate its effect on the success and convergence speed of gradient ascent. (2) Implement stochastic gradient ascent (above we implement the batch version that uses the entire dataset in each update) and evaluate its performance. (3) Implement the generative model with non-shared covariance matrix and evaluate its performance. In any of these, consider variants of the algorithm, its parameters or setting, and possibly generate new data to help explore the performance. Write a short report explaining what you did, the results and your observations about them.

Additional Notes

- Equation 4.143 of Bishop (pg. 218) is incorrect as documented in the textbook errata. The LHS should be S_N^{-1}, **not** S_N.
- If you have problems with singular matrices, you can simply "regularize" them by adding a small constant to the diagonal, e.g., $+10^{-9}I$.

Submission

Please write clear code with sufficient documentation so that we can read it. In addition write a README file that explains how the code is organized (if in multiple files) and how to compile and run the code. If this is non-trivial please write a script that runs the code and explain how to use it in the README file. When run in this manner your code should produce all the results and plots as requested above.

Please submit two items via Canvas: (1) Please write a report on the experiments, their results, and your conclusions as requested above. Prepare a PDF file with this report. (2) Collect all your code for the assignment (including the README file) in a zip file named `pp3code.zip`. You do not need to include the data that we provided. Your code should assume that the data files will have names as specified above and will reside in sub-directory pp3data/ of the directory where the code is executed.

Grading

Your assignment will be graded based on (1) the clarity of the code, (2) its correctness, (3) the presentation and discussion of the results.