CS540 Spring 2010: homework 3

1 Gradient and Hessian of log-likelihood for logistic regression

1. Let $\sigma(a) = \frac{1}{1+e^{-a}}$ be the sigmoid function. Show that
$$\frac{d\sigma(a)}{da} = \sigma(a)(1-\sigma(a)) \tag{1}$$
2. Using the previous result and the chain rule of calculus, show that
$$\frac{d}{d\mathbf{w}} - \sum_{i=1}^{N} \log[\mu_i^{y_i} \times (1-\mu_i)^{1-y_i}] = \sum_i (\mu_i - y_i)\mathbf{x}_i = \mathbf{X}^T(\boldsymbol{\mu} - \mathbf{y}) \tag{2}$$
3. The Hessian can be written as $\mathbf{H} = \mathbf{X}^T\mathbf{S}\mathbf{X}$, where $\mathbf{S} \stackrel{\text{def}}{=} \text{diag}(\mu_1(1-\mu_1), \ldots, \mu_n(1-\mu_n))$. Show that $\mathbf{H}$ is positive definite. (You may assume that $0 < \mu_i < 1$, so the elements of $\mathbf{S}$ will be strictly positive, and that $\mathbf{X}$ is full rank.)

2 Regularizing separate terms in 2d logistic regression

(Source: Jaakkola)

1. Consider the data in Figure 1, where we fit the model $p(y=1|\mathbf{x}, \mathbf{w}) = \sigma(w_0 + w_1x_1 + w_2x_2)$. Suppose we fit the model by maximum likelihood, i.e., we minimize
$$J(\mathbf{w}) = -\ell(\mathbf{w}, \mathcal{D}_{\text{train}}) \tag{3}$$
where $\ell(\mathbf{w}, \mathcal{D}_{\text{train}})$ is the log likelihood on the training set. Sketch a possible decision boundary corresponding to $\hat{\mathbf{w}}$. (Copy the figure first (a rough sketch is enough), and then superimpose your answer on your copy, since you will need multiple versions of this figure). Is your answer (decision boundary) unique? How many classification errors does your method make on the training set?
2. Now suppose we regularize only the w_0 parameter, i.e., we minimize
$$J_0(\mathbf{w}) = -\ell(\mathbf{w}, \mathcal{D}_{\text{train}}) + \lambda w_0^2 \tag{4}$$
Suppose λ is a very large number, so we regularize w_0 all the way to 0, but all other parameters are unregularized. Sketch a possible decision boundary. How many classification errors does your method make on the training set? Hint: consider the behavior of simple linear regression, $w_0 + w_1x_1 + w_2x_2$ when $x_1 = x_2 = 0$.
3. Now suppose we heavily regularize only the w_1 parameter, i.e., we minimize
$$J_1(\mathbf{w}) = -\ell(\mathbf{w}, \mathcal{D}_{\text{train}}) + \lambda w_1^2 \tag{5}$$
Sketch a possible decision boundary. How many classification errors does your method make on the training set?
4. Now suppose we heavily regularize only the w_2 parameter. Sketch a possible decision boundary. How many classification errors does your method make on the training set?

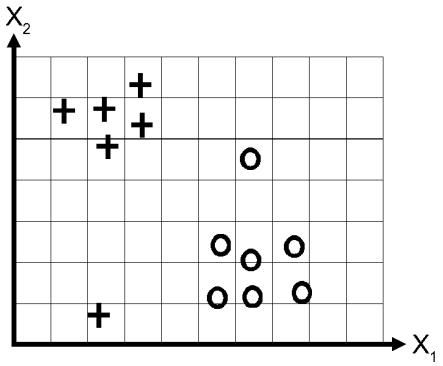

Figure 1: Data for logistic regression question.

3 Spam classification using logistic regression

Consider the email spam data set discussed on p300 of [HTF09]. This consists of 4601 email messages, from which 57 features have been extracted. These are as follows:

- 48 features giving the percentage (0 to 100) of words in a given message which match a given word on the list. The list contains words such as "business", "free", "george", etc. (The data was collected by George Forman, so his name occurs quite a lot.)
- 6 features giving the percentage (0 to 100) of characters in the email that match a given character on the list. The characters are ; ([! $ #
- Feature 55: The average length of an uninterrupted sequence of capital letters (max is 40.3, mean is 4.9)
- Feature 56: The length of the longest uninterrupted sequence of capital letters (max is 45.0, mean is 52.6)
- Feature 57: The sum of the lengts of uninterrupted sequence of capital letters (max is 25.6, mean is 282.2)

Load the data from `spamData.mat`, which contains a training set (of size 3065) and a test set (of size 1536). One can imagine performing several kinds of preprocessing to this data. Try each of the following separately:

1. Standardize the columns so they all have mean 0 and unit variance.
2. Transform the features using $\log(x_{ij} + 0.1)$.
3. Binarize the features using $\mathbb{I}(x_{ij} > 0)$.

For each version of the data, fit a logistic regression model. Use cross validation to choose the strength of the ℓ_2 regularizer. Report the mean error rate on the training and test sets. You should get numbers similar to this:

```
method   train   test
stnd     0.082   0.079
log      0.052   0.059
binary   0.065   0.072
```

(The precise values will depend on what regularization value you choose.) Turn in your code and numerical results.

4 Symmetric version of L2 regularized multinomial logistic regression

(Source: Ex 18.3 of [HTF09])
Consider optimizing

$$\sum_{i=1}^{N} \log p(y_i|\mathbf{x}_i, \mathbf{w}) - \lambda \sum_{c=1}^{C} ||\mathbf{w}_c||_2^2 \tag{6}$$

where

$$p(y = c|\mathbf{x}, \mathbf{w}) = \frac{\exp(w_{c0} + \mathbf{w}_c^T \mathbf{x})}{\sum_{k=1}^{C} \exp(w_{k0} + \mathbf{w}_k^T \mathbf{x})} \tag{7}$$

Show that at the optimum we have $\sum_{c=1}^{C} \hat{w}_{cj} = 0$ for $j = 1 : D$. What about the $\hat{w}_{c0}$ terms? Discuss issues with these constant parameters, and how they can be resolved.

5 Sufficient statistics for online linear regression, part II

Consider exercise 11.4 from the book. Last time you showed that the sufficient statistics to update w_0 and w_1 online are $C_{xy}, C_{xx}, \overline{x}, \overline{y}$, and you derived a recursive update for $\overline{x}$ and C_{xy}.

1. Derive a recursive update for $\overline{y}$ and C_{xx}.
2. Implement the online learning algorithm, i.e., write a function of the form `[w,ss] = linregUpdateSS(ss, x, y)`, where x and y are scalars and ss is a structure containing the sufficient statistics.
3. Plot the coefficients over "time", using the dataset in `linregDemo1`. Check that they converge to the solution given by the batch (offline) learner (i.e, ordinary least squares).

Turn in your derivation, code and plot.

6 Demo of robustness of unconditional Laplace

Figure 4.2 in the book (p109) illustrates that the Student distribution is more robust to outliers than the Gaussian. This figure was created by `gaussVsStudentOutlierDemo`. Make a copy of this file and modify it to fit both datasets (the "pure" one and the one with outliers) with a Laplace distribution. Turn in your code and plot.

7 Robust linear regression using Laplace noise

The function `linregRobustLaplaceLinprog1dFit` fits a simple linear regression model with Laplacian noise, and was used to create Figure 11.7a in the book (p325). However, it assumes the input is scalar, $x_i \in \mathbb{R}$. Modify this function to handle any size of input, so $\mathbf{x} \in \mathbb{R}^D$. Check that it gives the same result as the scalar code. Turn in your code.

8 Robust linear regression using Student noise

One way to define a robust linear regression model is to use

$$p(y|\mathbf{x}, \mathbf{w}, \sigma^2, \nu) = \mathcal{T}(y|\mathbf{w}^T \mathbf{x}, \sigma^2, \nu) \tag{8}$$

1. Derive the negative log likelihood (NLL) for this model. Simplify as much as possible.
2. Derive its gradient (a vector) wrt $\mathbf{w}$. (You may assume ν and σ^2 are fixed.) Hint: use the chain rule of calculus.

3. Implement a function `StudentLoss` which evaluates the NLL and its gradient. Then implement a function `linregRobustStudentFit` that optimizes the Student NLL using `minfunc`. Turn in your code.

4. Apply your function to the data in `linregRobustDemo` and plot the result for $\nu \in \{2, 4, 6, 8, 10\}$. Turn in your code and plot.

5. Now derive the gradient wrt ν and σ^2 and create a modified fitting function. Implement this. What is the MLE for ν and σ^2 for the data in `linregRobustDemo`? Plot the resulting prediction. Turn in your algebra, code and plot. Hint: appendix B of [LLT89] contains some potentially relevant formulas.

References

[HTF09] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2009. 2nd edition.

[LLT89] K. Lange, R. Little, and J. Taylor. Robust statistical modeling using the t disribution. *J. of the Am. Stat. Assoc.*, 84(408):881–896, 1989.