250B Problem Set 4

William Fedus

February 19, 2015

1 Properties of Σ

1.1 Invertibility from eigenvalues

If any of the Σ eigenvalues $\lambda_i = 0$, the matrix is not invertible. We can see that any $\lambda_i = 0$ implies that the matrix is not invertible because if $\lambda_i = 0$, then we assume there is a nontrivial solution with

$$\Sigma x_i = 0 x_i = 0 \tag{1}$$

however, by the Invertible Matrix Theorem, if Σ was invertible then there would only be a trivial solution, therefore, an eigenvalue of 0 implies non-invertibility.

1.2 Eigenvalues and Eigenvectors

Let $c > 0$ be any constant and assume we know that $\Sigma x = \lambda x$, then the eigenvalues of $\Sigma + cI$ are found in the normal procedure,

$$(\Sigma + cI)\, x = (\lambda + c)\, x \equiv \lambda' x \tag{2}$$

so we see that eigenvalues of this new matrix are simply $\lambda' = \lambda + c$. The eigenvectors $u_1, \ldots, u_n$ will remain the same since we have only rescaled the eigenvalues.

1.3 Inverse of Σ

Σ may be written in the spectral decomposition as

$$\Sigma = Q \Lambda Q^T \tag{3}$$

where Q is a matrix of the eigenvectors in column format and Λ is the diagonal matrix of eigenvalues. The inverse of Σ may be written as

$$\Sigma^{-1} = Q \Lambda^{-1} Q^T \tag{4}$$

which implies that the eigenvalues of Σ^{-1} are simply $1/\lambda_i \forall i$.

2 Regression

2.1 Convexity of $L(w)$ as a Function of w

For a set of data $\left((x^{(1)}, y^{(1)}), \ldots, (x^{(n)}, y^{(n)})\right) \in \mathbb{R}^p \times \mathbb{R}$, we define a loss function

$$L(w) = \sum_{i=1}^{n} \left(y^{(i)} - w \cdot x^{(i)}\right)^2 \tag{5}$$

Recall, to test convexity, we need to show that the Hessian matrix H, as defined in Equation 6, is positive semi-definite for all $z \in \mathbb{R}^p$.

$$H_{jk} = \frac{\partial^2 f}{\partial z_k \partial z_j} \tag{6}$$

so we calculate the Hessian for Equation 5,

$$H_{jk} = \frac{\partial^2 L}{\partial w_j \partial w_j} = 2 \sum_{i=1}^{n} x_j^{(i)} x_j^{(i)} \tag{7}$$

which indicates that H may be written in the form,

$$H = 2xx^T \tag{8}$$

to test if this is positive semi-definite, we check if the definition $z^T H z \geq 0$ holds,

$$\begin{aligned} z^T H z &= 2z^T x x^T z \\ &= 2\left(z^T x\right)\left(x^T z\right) \\ &= 2||x^T z||^2 \\ &= \geq 0 \end{aligned}$$

2.2 Gradient Descent Update

In order to find the w which minimizes Equation 5, we can use the iterative procedure of gradient descent,

$$w_{t+1} = w_t - \eta_t \nabla L(w_t) \tag{9}$$

which for $L(w)$ becomes

$$w_{t+1} = w_t + 2\eta_t \sum_{i=1}^{n} \left(y^{(i)} - w \cdot x^{(i)}\right) x^{(i)} \tag{10}$$

2.3 Newton-Raphson Update

The Newton-Raphson procedure relies on 2nd order Taylor approximation and has the following update form,

$$w_{t+1} = w_t - \eta_t H^{-1}(w_t)\nabla L(w_t) \tag{11}$$

so for $L(w)$ the update rule will become,

$$w_{t+1} = w_t + \eta_t (2xx^T)^{-1} \sum_{i=1}^{n} \left(y^{(i)} - w \cdot x^{(i)} \right) x^{(i)} \tag{12}$$

This result is very similar to the gradient descent update rule, however, now we compute the inverse of the Hessian matrix $H^{-1} = (2xx^T)^{-1}$ in order to create a second order model.

3 Convexity

We show that the following functions $f : \mathbb{R}^p \to \mathbb{R}$ are convex.

3.1 $f(x) = x^T Mx$, where $M \in \mathbb{R}^{p\times p}$

We may write $f(x)$ as,

$$f(x) = x^T Mx = \sum_{i,j} x_i M_{ij} x_j \tag{13}$$

and then

$$H_{kl} = \frac{\partial^2 f}{\partial x_k \partial x_l} = M_{kl} \tag{14}$$

but since M is positive semidefinite, then H is also positive semidefinite and the function $f(x)$ is convex.

3.2 $f(x) = e^{u\cdot x}$, for some $u \in \mathbb{R}^p$

We may write the Hessian matrix H as,

$$H_{ij} = \frac{\partial^2 f}{\partial x_i \partial x_j} = e^{u\cdot x} u_i u_j \tag{15}$$

so then $H = e^{u\cdot x} uu^T$ which can be shown to be positive semidefinite from the same argument in Section 2.1.

3.3 $f(x) = \max(g(x), h(x))$, where g and h are convex.

Here, both $g(x)$ and $h(x)$ are assumed to be convex, and we will take the maximum of both of these to find $f(x)$. Since both functions are convex, both of their Hessian matrices, H_g and H_h, respectively, will be positive semidefinite. The Hessian matrix of $f(x)$, H_f will be either H_g and H_h and therefore, since both are positive semidefinite, H_f will be positive semidefinite implying that $f(x)$ is convex.

4 Logistic Regression using Gradient Descent

4.1 Algorithm

Our function will return a classification vector $w \in \mathbb{R}^p$ obtained by gradient descent on the logistic regression loss function, Equation 16

$$L(w) = \sum_{i=1}^{n} \ln\left(1 + e^{-y^{(i)}(w \cdot x^{(i)})}\right) \tag{16}$$

In order to find the w which minimizes Equation 16, we use gradient descent for logistic regression as described in Algorithm 1.

Algorithm 1 Gradient Descent

```
1: procedure GRADIENT DESCENT FOR LOGISTIC REGRESSION
2:     $w \leftarrow 0$
3:     for t = 0, ..., m do
4:         $w_{t+1} = w_t + \eta_t \sum_{i=1}^{n} y^{(i)} x^{(i)} P_{w_t}\left(-y^{(i)} | x^{(i)}\right)$
5:     return $w$
```

where n is the number of training examples and the step size is η_t. In a simplest format, η_t may be chosen to be a constant value, that is, the algorithm will take a constant step size at each iteration. However, for step sizes too small, convergence may be excessively slow and for step sizes too large, convergence may be an issue since the algorithm may overstep the equilibrium. To choose η_t, we employ the backtracking line search to minimize $L(w_{t+1})$ algorithm.

4.2 Performance on Toy Data Set

We run this algorithm on the toy data set of 6 points in $\mathbb{R}^2$ provided, each with label ± 1. Convergence of the algorithm is defined by sequential changes of the loss function $L(w)$ of $\epsilon < 0.001$ which is achieved after 2181 iterations of the algorithm. The resulting decision boundary is seen in Figure 4.2.

This trivial data set is linearly separable with a considerable margin, but the number of iterations before convergence to our established tolerance is slow. In order to see the operation of the algorithm, we show the state of the classification

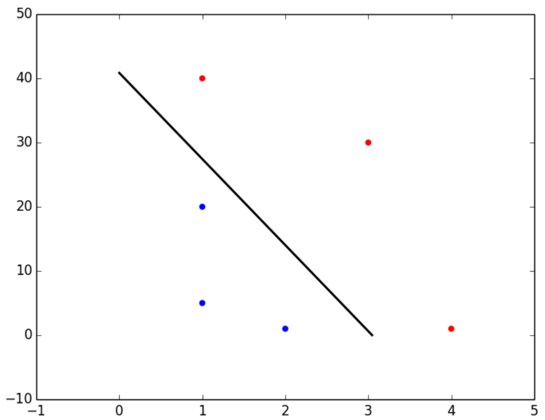

Figure 1: Resulting classification boundary in black after 2181 iterations of the gradient descent algorithm with backtracking line search for η_t selection. Toy data set are plotted in red (+1 label) and blue (−1 label).

boundary after 10, 100 and 1000 iterations of the algorithm (all pre-converged) in Figure 4.2

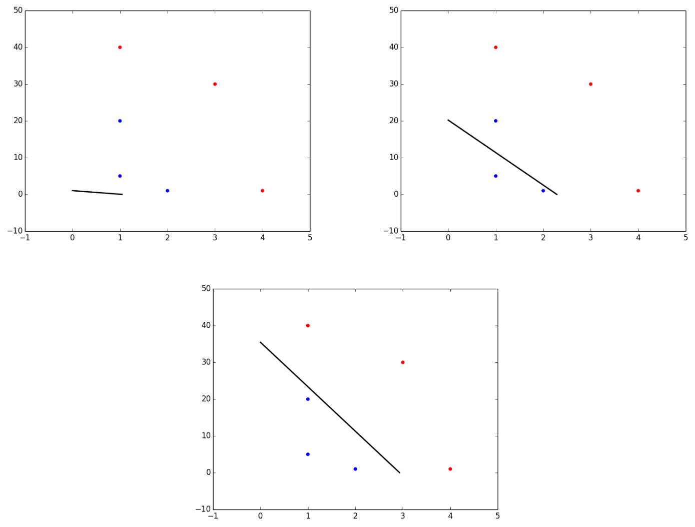

Figure 2: Resulting classification boundary in black after 10,100 and 1000 iterations of the gradient descent algorithm with backtracking line search for η_t selection. Toy data set are plotted in red (+1 label) and blue (−1 label).

Notice that the algorithm continues to move towards the optimal solution, however, the time to convergence simply takes a long time with this type of data.

4.3 Performance on the Scaled Toy Data Set

The toy data set has considerably higher variance in the x_2 coordinate, where points range $[1, 40]$. To normalize this to be inline with the range associated with the x_1 cooridiante, we scale the second axis down by a factor of 10. Now when we perform gradient descent, we achieve very rapid convergence after only 55 iterations (with convergence defined identically to Section 4.2), a several order of magnitude improvement. The resulting classification boundary is seen in Figure 4.3.

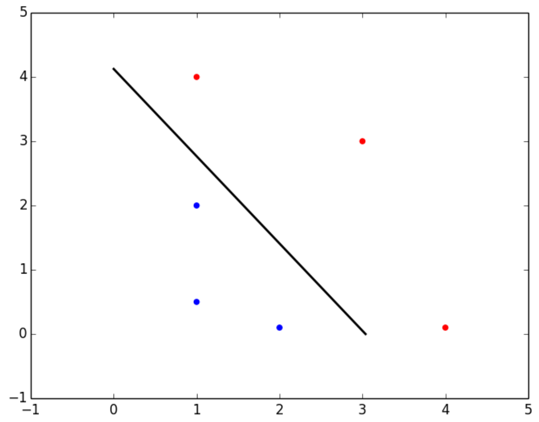

Figure 3: Resulting classification boundary in black after only 55 iterations of the gradient descent algorithm with backtracking line search for η_t selection. The rescaled toy data set are plotted in red (+1 label) and blue (-1 label).

This indicates the importance of scaling each feature to comparable ranges before using this methodology.

4.4 Performance on General 2D Data

As a final test of the gradient descent algorithm, we consider the performance on 100 data points drawn from two multivariate Gaussian distributions, one defined as generating points with label $+1$ and one with label -1. Here the resulting points overlap and thus are not perfectly linearly separable. In Figure 4.4 we see the resulting decision boundary in black as well as the labeled points.

As we can see, the logistic regression gradient descent, with η_t chosen via backtracking line search has performed well and finds a very suitable linear

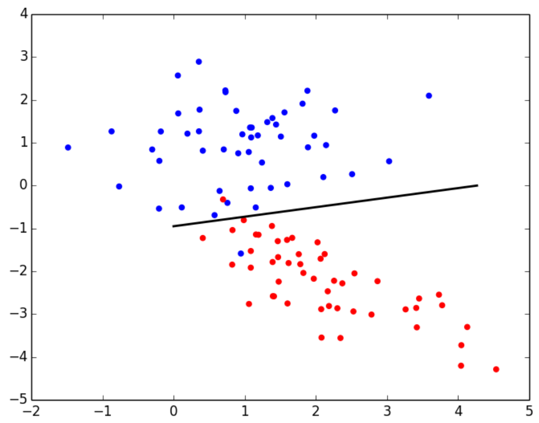

Figure 4: Resulting classification boundary in black after only 20 iterations of the gradient descent algorithm with backtracking line search for η_t selection. The rescaled toy data set are plotted in red (+1 label) and blue (-1 label).

classifier in these data sets.