# Homework 2

## 1. Logistic Regression

We begin by investigating the effects of regularization ($L_1$ and $L_2$) on logistic regression classifiers. That is, we consider loss functions of the following form:

$$E_{LR}(w, w_0) \equiv NLL(w, w_0) + \lambda \|w\|_k^k$$

where $k = 1, 2$, and $NLL$ is the negative log-likelihood loss for logistic regression. We note that such classifiers can be trained efficiently via gradient descent, with the gradient term being given by (using $L_2$ as an example):

$$\frac{\partial E_{LR}}{\partial dw_j} = \frac{\partial NLL}{\partial dw_j} + \lambda \frac{\partial \|w\|_k^k}{\partial w_j} = \begin{cases} -\sum_i \frac{y^{(i)} e^{-y^{(i)}(wx^{(i)}+w_0)}}{1+e^{-y^{(i)}(wx^{(i)}+w_0)}} & j = 0 \\ -\sum_i \frac{y^{(i)} x_j^{(i)} e^{-y^{(i)}(wx^{(i)}+w_0)}}{1+e^{-y^{(i)}(wx^{(i)}+w_0)}} + 2\lambda w_j & j \neq 0 \end{cases}$$

To explore the effects of regularization, we first consider the non-regularized case, with $\lambda = 0$. **Figure 1(a)** shows that, as expected, the magnitude of the weight vector goes to infinity as the number of iterations of gradient descent grows large. (For the purposes of illustration, we used a deliberately small step size, $\eta = 10^{-4}$.) When we include regularization at $\lambda = 1$, however, the magnitude of the weight vector stabilizes, even as the number of iterations becomes large.

## 2. Support Vector Machine

## 3. Support Vector Machine with Pegasos

## 4. Handwritten Digit Recognition with MNIST

In this section, we investigate the performance of gradient descent-based algorithms for minimizing scalar functions of vector arguments. In particular, we consider the following functions: 1) the negative Gaussian function:

$$f(x) = -\frac{1}{\sqrt{(2\pi)^n |\Sigma|}} \exp\left[-\frac{1}{2}(x-u)^T \Sigma^{-1}(x-u)\right]$$

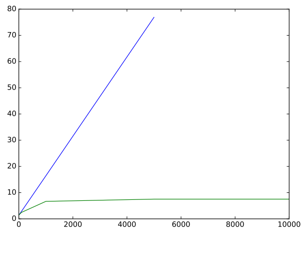
and 2) the quadratic bowl function $f(x) = \frac{1}{2}x^T Ax - x^T b$.

4.1. **Initialization and Hyperparameters.** We first explore the batch gradient descent algorithm. For loss function $f(x)$, the gradient descent algorithm performs $x = x - \eta\nabla f(x)$ where $\eta$ is the step size. These iterations continue until a convergence threshold is met, i.e. $|f(x) - f(\tilde{x})| < \epsilon$, where $x, \tilde{x}$ are successive iterations. We investigate the dependence of this algorithm on the initialization of the algorithm as well as its hyperparameters, namely the step size $\eta$ and the convergence threshold $\epsilon$. Throughout the experiments, we fix the following parameters of the functions $u = (10, 10), \Sigma = \begin{pmatrix} 1000 & 0 \\ 0 & 1000 \end{pmatrix}$ and $A = \begin{pmatrix} 10 & 5 \\ 5 & 10 \end{pmatrix}, b = (400, 400)$.

First, we note that the large variance in the negative Gaussian density yields small objective function values (scaled by the reciprocal of the determinant of $\Sigma$), as well as small gradients. Thus, we consider only large step sizes that will yield noticeable increments in the values of $x$; otherwise, no learning is achieved. Similarly, we only consider very small $\epsilon$ for the same reasons.

The results are shown in **Figure 1** for the negative Gaussian function, where the $z$-axis is the squared distance. We have used step sizes of $10^5, 10^6, \ldots, 10^10$ and convergence criteria of $10^{-10}, 10^{-11}, \ldots, 10^{-20}$. The plots reveal two major points. First, for any choice of initialization and step size used in the experiments, gradient descent does not converge to a suitable solution until $\epsilon$, the convergence threshold, is sufficient small. As the plots demonstrate, we require $\epsilon \approx 10^{-15}$ or $\log(\epsilon) \approx -35$ before the squared error falls to negligible values. Second, the step size must bee sufficiently large before convergence is achieved, in the same vein as the convergence threshold. Even for suitable $\epsilon$ values, a relatively small step size (i.e. $10^5$) does not yield approximate convergence to the correct solution. Indeed, for most cases, we require a step size of $\eta \approx 10^7$ before the suared loss becomes negligible. As discussed above, these issues are germane to the negative Gaussian function, due to the large determinant value effectively annihilating the step sizes unless sufficiently large.

On the other hand, the quadratic bowl function does not have such issues, and using large step sizes generally yields divergence. Indeed, even using $\eta = 1$ leads to diverging estimates for $\epsilon \approx 10^{-15}$. Thus, for this case we consider step sizes of $10^{-1}, 10^{-2}, \ldots, 10^{-5}$



(A) (1,1) (B) (5,5)

Figure 1. Squared distance to correct solution $u$ using the negative Gaussian function for various starting points, step sizes, and convergence criteria. The labels on the plots are the starting points.

and similarly convergence criteria of $10^{-10}, 10^{-11}, \ldots, 10^{-20}$. Results as shown in **Figure 2** demonstrate a very similar behavior for the quadratic bowl function; the convergence threshold $\epsilon$ must be suitably small for convergence to the correct values. In this case, however, if the step size $\eta$ is too small, then no $\epsilon$ used in our experiments can yield convergence, unlike the case of the negative Gaussian, in which a suitably small $\epsilon$ would yield convergence in all step sizes. We must have a large enough step size (in this case $\eta \approx 0.01$) for convergence in this example.