**CS 688: Machine Learning**
**George Mason University, Fall 22** **Homework 2** **Due at 11:59pm, Oct 4**

Problems marked with **E** are graded on effort, which means that they are graded subjectively on the perceived effort you put into them, rather than on correctness. We strongly encourage you to typeset your solutions in LaTeX.

1. (40 points) Recall that the logistic regression assumes the outcome is sampled from a Bernoulli distribution: $\Pr[y = 1|x, w] = sigm(w^\top x)$ where $sigm(\eta) := e^\eta/(1 + e^\eta)$ for all $\eta \in \mathbb{R}$. Finding maximum likelihood estimator on training data $(x_1, t_1), \ldots, (x_n, t_n)$ is equivalent to minimizing $E(w) := -\sum_{i=1}^n \ln \Pr[y = t_i|x_i, w]$. However, the above objective function is not very stable. In this problem, we applied the idea of regularization to logistic regression. Formally, we consider the following objective function

$$E_\lambda(w) := -\sum_{i=1}^n \ln \Pr[y = t_i|x_i, w] + \frac{\lambda}{2}\|w\|^2 \tag{1}$$

where $\lambda \geq 0$, $\|\cdot\|$ is 2-norm, and $x_i \in \mathbb{R}^d, t_i \in \{0, 1\}$ for all $i = 1, \ldots, n$.

   (a) Show that for all $w \in \mathbb{R}^d$

$$\nabla_w L_\lambda(w) = -\sum_{i=1}^n (t_i - \sigma_i)x_i + \lambda w \text{ and } \sum_i \sigma_i(1 - \sigma_i)x_i x_i^\top + \lambda\mathbb{I}$$

   where $\sigma_i := sigm(w^\top x)$ and $\mathbb{I}$ is the identity matrix of size $d$.

   (b) Again, in order to make our code fast, simplify this gradient expression with matrix algebra by expressing it in terms of $X \in \mathbb{R}^{n\times d}, t, \sigma \in \mathbb{R}^n$ and other relevant quantities.

   (c) Show the objective function $E_\lambda(w)$ in Eq. (1) is convex in $w$.

   (d) Implement gradient descent for logistic regression on 2 vs 9 dataset. Let us consider the classification problem of recognizing if a digit is a "2" or a "9"; y = 1 is the label corresponding to class "2" and y = 0 corresponds to the class "9". Download `mnist_2_vs_9.gz` dataset, and load the dataset in Python as follows.

```
import gzip, pickle
with gzip.open("mnist_2_vs_9.gz") as f:
  data = pickle.load(f, encoding="bytes")
  Xtrain, Ytrain, Xtest, Ytest, Xdev, Ydev = \
      data[b"Xtrain"], data[b"Ytrain"], data[b"Xtest"], \
      data[b"Ytest"], data[b"Xdev"], data[b"Ydev"]
```

   Make sure to include a constant in the model and do no regularize this term. Specifically, given $x \in \mathbb{R}^d$, $w = (w_1, \ldots, w_d)$ and $w_0 \in \mathbb{R}$, let $\Pr[y = 1|x, w] = sigm(w_0 + w^\top x)$ and $E(w, w_0) := -\sum_{i=1}^n \ln \Pr[y = t_i|x_i, w, w_0]$. Thus,

$$E_\lambda(w, w_0) := E(w, w_0) + \frac{\lambda}{2}\|w\|^2$$

   Then the gradient descent with step size $\eta > 0$ would be

$$\begin{aligned} w &\leftarrow w - \eta\nabla_w E_\lambda(w, w_0) = w - \eta(\nabla_w E(w, w_0) + \lambda w) \\ w_0 &\leftarrow w_0 - \eta\frac{\partial}{\partial w_0}E_\lambda(w, w_0) = w_0 - \eta\frac{\partial}{\partial w_0}E(w, w_0) \end{aligned}$$

Show your log loss $E(w)$ on the training data and development data (validation set) where the $x$-axis is the number of step of the above gradient descent and $w$ starts from 0. Specify your parameter choices $\eta$ and $\lambda$. What stepsize $\eta$ do you find works well and what value of $\lambda$ did you use based on the validation data?

(e) Make this plot again with both curves but use the classification error (zero one loss normalized by the number of data).

(f) Finally, report your lowest test error.

2. (bonus 20 points) Now we give a closer look at regularized logistic regression. We want to show the objective function is equivalent to placing a zero-mean Gaussian prior distribution on the coefficients. Formally, given $\lambda > 0$ and features $x = (x_1, \ldots, x_n)$ where $x_i \in \mathbb{R}^d$, if the prior of $w$ is sampled from a $d$-dimensional Gaussian $N(0, \frac{1}{\lambda}\mathbb{I})$ with zero mean and covariance $\lambda\mathbb{I}$ and the likelihood function is $\Pr[y = 1|x, w] = sigm(w^\top x)$,

   (a) Show the MAP estimator $w_{MAP} = \arg\max_w \Pr[w|x, t]$ given $t = (t_1, \ldots, t_n)$ satisfies $E_\lambda(w_{MAP}) = \min_w E_\lambda(w)$ defined in Eq. (1).

   (b) As we increase $\lambda > 0$, the prior of $w$ becomes narrow, and intuitively the MAP estimator gets closer to the zero. Show that $\|w_{MAP}\|$ decreases as $\lambda$ increases.

3. (Poisson regression, 20 points) Poisson regression considers the outcome is sampled from a Poisson distribution: $\Pr[y = t|x, w] = \frac{\mu^t e^{-\mu}}{t!}$ where $\mu = e^{w^\top x}$ for all $x$ and $w \in \mathbb{R}^d$.

   (a) For a training set $(x_i, t_i)_{i=1}^n$, compute the gradient and Hessian of the negative log likelihood function, $-\sum_i \ln \Pr[y = t_i|x_i, w]$.

   (b) Show the negative log likelihood function is convex.

4. (40 points) In this problem, we will use the naive Bayes algorithm to build a SPAM classifier. In a document-term matrix, the $i$-th raw and $j$-th colum is the number of occurance of the $j$-th token (e.g., vocabulary, email address...). You may use `numpy` and `pandas` to process files.

   (a) Implement a naive Bayes classifier for spam classification using the multinomial event model and Laplace smoothing. You can use the code outline provided in p4.py to train and test your model. Train your parameters using the document word matrix in `MATRIX.TRAIN` and then report the test error on `MATRIX.TEST`.

   (b) Find 5 tokens that has the largest likelihood ratio $\ln(\Pr[token = i|SPAM]/\Pr[token = i|NOTSPAM])$ from your estimated naive Bayes classifier.

   (c) Repeat part (a), but on training data of size 100, 200, 400, and 800, buy using the files `MATRIX.TRAIN.*`. Plot the test error on `MATRIX.TEST` to obtain a learning curve (test error vs training data size). Which training-set size gives the best test set error.