CSE 446: Machine Learning
Winter 2018

Assignment 3
w/o bonus questions

from
Lukas Nies
University of Washington

02/22/18

Contents

0 Policies

0.1 List of Collaborators

My collaborator was Edith Heiter (discussed parts of Problem 1 and 4). The development of the answers though was completely independent and individually.

0.2 List of Acknowledgments

None.

0.3 Policies

I have read and understood these policies.

0.4 Note: Bonus not included!

I will include some of the bonus questions in an extended version of this .pdf for Monday, 26th in the next upload.

1 Problem: Linear Regression on MNIST

1.1 Closed Form Estimator

1. If one runs the Closed Form Estimator with $\lambda = 0$ one encounters trying to invert a singular matrix $(X^T X)$ which is not possible per definition since the determinant is $\det(X^T X) = 0$. The matrix is therefore not invertible. To avoid this we introduce a regularization by adding the term $\lambda \mathbb{1}_d$. This is intuitively clear by considering the data itself: one digit consists of 28×28 pixels where most pixels (at the edges and in the corners) don't carry any information about the digit itself (matrix is sparse). When calculating $X^T X$ we get the same result: we have more "dimensions" than information for those "dimensions". In mathematical terms: $X^T X$ is under-determined.

2. For this part a grid search was implemented to search for different values of λ and the threshold to optimize the performance on the development set:

   (a) The best result was found with $\lambda = 0.02$ and a threshold of 0.5. The grid search ran for λ from 0.01 to 1 with steps of 0.01, the treshold ran from 0.1 to 1.0 in steps of 0.1.

   (b) The average squared error using the parameters stated above is as follows:
   - Training error = 0.013045
   - Development error = 0.01420
   - Test error = 0.01626

   (c) The misclassification error using the parameters stated above is as follows:
   - Training error = 0.93%
   - Development error = 1.08%
   - Test error = 1.76%

3. Samples with large values (far off the mean of the rest of the data points) have a strong influence on linear polynomial functions fitted through regression. This leads to large misclassification on most of the data points. A better model would be using a higher order polynomial to fit those samples more efficiently.

1.2 Linear regression using gradient descent

1. The proof is as follows:

$$\begin{aligned}
\frac{\partial \mathcal{L}_\lambda}{\partial w} &= \frac{\partial}{\partial w}\left(\frac{1}{N}\sum_{n=1}^{N}\frac{1}{2}\left(y_n - w^T x_n\right)^2 + \frac{\lambda}{2}\|w\|^2\right) \\
&= \frac{1}{N}\sum_{n=1}^{N}\left(-\frac{2x_n}{2}\right)\left(y_n - w^T x_n\right) + \left(\frac{2\lambda}{2}\mathbf{w}\right) \\
&= -\frac{1}{N}\sum_{n=1}^{N}\left(y_n - \hat{y}_n\right)x_n + \lambda\mathbf{w}
\end{aligned}$$

2. We can rewrite this as a matrix expression:

$$\frac{\partial \mathcal{L}_w}{\partial w} = -\frac{1}{N}\sum_{n=1}^{N}\left(y_n - \hat{y}_n\right)x_n + \lambda \mathbf{w} = -\frac{1}{N}X^T \cdot \left(Y - \hat{Y}\right) + \lambda \mathbf{w}$$

3. Stepsizes $-10^{-2} \leq \eta < -10^{-1}$ worked well for this problem. For the error rate see figure 1. For generating the plots, $\eta = \frac{1}{4} \times 10^{-1}$ and $\lambda = 10^{-2}$ were chosen. The lowest error I achieved is comparable to the closed form estimator, with 1.08% on the development set.

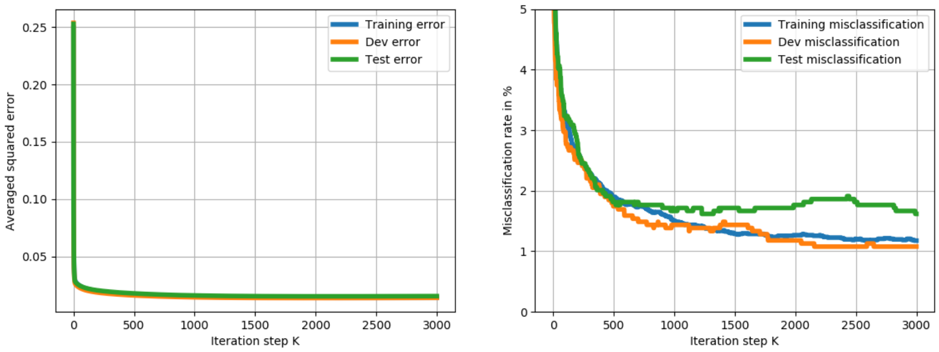

Figure 1: Plot of averaged squared errors (left) and misclassification loss in percent (right) for the gradient descent algorithm. For generating the plots, $\eta = \frac{1}{4} \times 10^{-1}$ and $\eta = 10^{-2}$ were chosen.

1.3 Linear Regression Using Stochastic Gradient Descent

1. The stochastic gradient descent diverges in this case with a learning rate for about $\eta = -0.1$ at $\lambda = 0.005$. For too large values of η the algorithm might never find the global minimum and therefore the gradient gets larger and larger which leads to divergence.

2. Stepsizes $\eta \leq -10^{-1}$ worked well for this problem. For the error rate see figure 2. For generating the plots, constant $\eta = \frac{1}{4} \times 10^{-1}$ and $\eta = 0.005$ were chosen. The lowest error I achieved is comparable to the closed form estimator, with 1.03% on the development set.

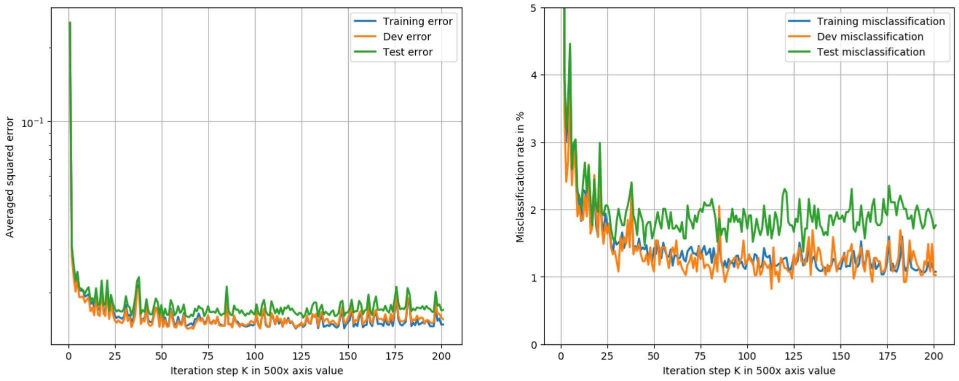

Figure 2: Plot of averaged squared errors (left, note the logarithmic vertical axis) and misclassification loss in percent (right) for the stochastic gradient descent algorithm. The horizontal axis shows the iteration steps for every 500th step. For generating the plots, $\eta = \frac{1}{4} \times 10^{-1}$ and $\eta = 0.005$ were chosen.

2 Binary Classification with Logistic Regression

1. For proofing this, we look at the cases $y_n = 1$ and $y_n = 0$ separately.
   Case $y_n = 1$:

$$\begin{aligned}
\frac{\partial \mathcal{L}_\lambda}{\partial w} &= \frac{\partial}{\partial w}\left(-\frac{1}{N}\sum_{n=1}^{N} \log p_w(y_n = 1|x_n) + \frac{\lambda}{2}\|\mathbf{w}\|^2\right) \\
&= \frac{\partial}{\partial w}\left(-\frac{1}{N}\sum_{n=1}^{N} \frac{1}{1+\exp(-wx_n)} + \frac{\lambda}{2}\|\mathbf{w}\|^2\right) \\
&= \frac{\partial}{\partial w}\left(-\frac{1}{N}\sum_{n=1}^{N} [\log(1) - \log(1+\exp(-wx_n)] + \frac{\lambda}{2}\|\mathbf{w}\|^2\right) \\
&= -\frac{1}{N}\sum_{n=1}^{N}\left[\frac{x_n \exp(-wx_n)}{1+\exp(-wx_n)}\right] + \lambda\mathbf{w} \\
&= -\frac{1}{N}\sum_{n=1}^{N}\left[x_n(\hat{y}_n^{-1} - 1)\hat{y}_n\right] + \lambda\mathbf{w} = -\frac{1}{N}\sum_{n=1}^{N}(1-\hat{y}_n)x_n + \lambda\mathbf{w}
\end{aligned}$$

Case $y_n = 0$:

$$\begin{aligned}
\frac{\partial \mathcal{L}_\lambda}{\partial w} &= \frac{\partial}{\partial w}\left(-\frac{1}{N}\sum_{n=1}^{N}\log p_w(y_n = 0|x_n) + \frac{\lambda}{2}\|\mathbf{w}\|^2\right)\\
&= \frac{\partial}{\partial w}\left(-\frac{1}{N}\sum_{n=1}^{N}\frac{1}{1+\exp(+wx_n)} + \frac{\lambda}{2}\|\mathbf{w}\|^2\right)\\
&= \frac{\partial}{\partial w}\left(-\frac{1}{N}\sum_{n=1}^{N}[\log(1) - \log(1+\exp(+wx_n)] + \frac{\lambda}{2}\|\mathbf{w}\|^2\right)\\
&= -\frac{1}{N}\sum_{n=1}^{N}\left[\frac{-x_n\exp(+wx_n)}{1+\exp(+wx_n)}\right] + \lambda\mathbf{w} = -\frac{1}{N}\sum_{n=1}^{N}\left[\frac{-x_n}{\exp(-wx_n)+1}\right] + \lambda\mathbf{w}\\
&= -\frac{1}{N}\sum_{n=1}^{N}-x_n\hat{y}_n + \lambda\mathbf{w} = -\frac{1}{N}\sum_{n=1}^{N}(0-\hat{y}_n)x_n + \lambda\mathbf{w}
\end{aligned}$$

2. We can rewrite this as a matrix expression:

$$\frac{\partial \mathcal{L}_\lambda}{\partial w} = -\frac{1}{N}\sum_{n=1}^{N}(y_n - \hat{y}_n)\,x_n + \lambda\mathbf{w} = -\frac{1}{N}X^T \cdot \left(Y - \hat{Y}\right) + \lambda\mathbf{w}$$

3. Properties of logistic regression

   (a) Suppose the data is linear separable and $\lambda = 0$. In order to fit the data best, one would like to optimize the sigmoid function $\frac{1}{1+\exp(-wx)}$. Since the data is linear separable all, data points with $y_n = 0$ are left of $x = 0$ and all points with $y_n = 1$ are to the right. In this case, the optimal fit would be the Heaviside function (step function, 0 for $x < 0$, 1 for $x > 0$). In order to optimize the sigmoid function to approach the Heaviside function, $w \to \inf$. Hence, our weight vector would diverge.

   (b) If we suppose that $d > n$ ($\lambda = 0$) then the data matrix is sparse and several features will carry no information (equal 0). To fit a larger accumulation of features with value 0 the logistic function must approach, similar to previous question, the Heaviside function. Therefore the weight vector will diverge.

   (c) To avoid the divergence of the weight vector one can introduce regularization which avoids sparsity and linear separability such that the algorithm stops early enough to give a good estimation without diverging too fast. If one does not consider this the algorithm might overfit which influences the true error.

3 Multi-Class classification using Least Squares

3.1 ”One vs. all Classification” with Linear Regression

1. We can rewrite this as a matrix expression:

$$\frac{\partial \mathcal{L}_W}{\partial W} = -\frac{1}{N}\sum_{n=1}^{N} x_n \left(y_n - \hat{y}_n\right)^T + \lambda \mathbf{W} = -\frac{1}{N} X^T \cdot \left(Y - \hat{Y}\right) + \lambda \mathbf{W}$$

2. Stepsizes $-10^{-2} \leq \eta \leq -10^{-1}$ worked well for this problem using the stochastic gradient descent. For the error rate see figure 3. For generating the plots, $\eta = \frac{2}{4} \times 10^{-1}$ and $\eta = 10^{-2}$ were chosen. The lowest error I achieved is roughly 13% on the development set.

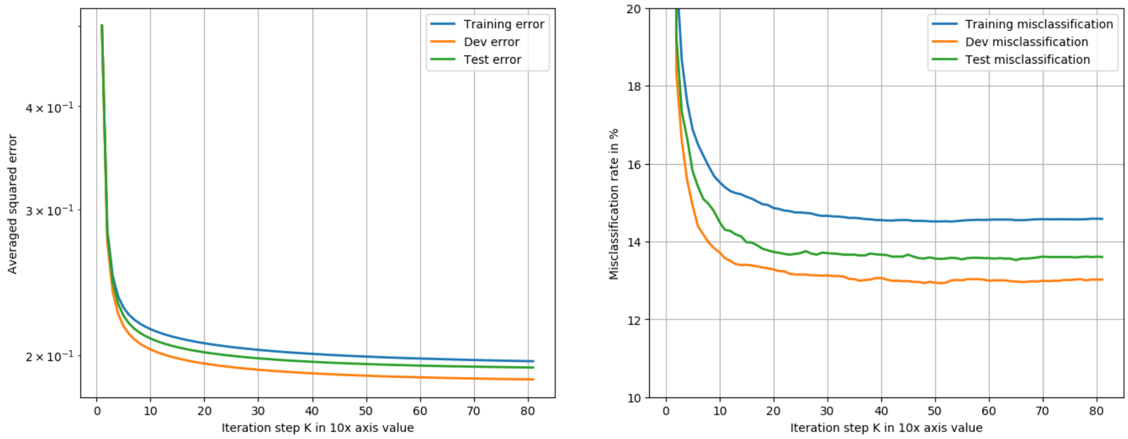

Figure 3: Plot of averaged squared errors (left) and misclassification loss in percent (right) for the gradient descent algorithm. For generating the plots, $\eta = \frac{1}{4} \times 10^{-1}$ and $\eta = 10^{-2}$ were chosen.

4 Probability and Maximum Likelihood Estimation

4.1 Probability Review

1. (a) Since the disease is quite rare it is likely that one does not have the disease even if one is tested positive. The amount of healthy persons in a group is larger than the number of sick persons and even if the test is highly accurate it's much more likely that one is healthy given the test is incorrect.

   (b) The probability of being tested positive (P) given having the disease ($\bar{H}$) is given by:

$$P(P|\bar{H}) = \frac{P(\bar{H}|P)P(P)}{P(\bar{H})} = 0.99. \tag{1}$$

   By reversing this Bayesian theorem we can yield the probability having the disease given being tested positive:

$$P(P|\bar{H}) = \frac{P(\bar{H}|P)P(\bar{H})}{P(P)}, \tag{2}$$

   where $P(P)$ is the total probability being tested positive, which is the sum of the probabilities of being tested positive and being sick, or being tested positive and being healthy:

$$P(\bar{H}|P) = \frac{P(P|\bar{H})P(\bar{H})}{P(P)} = \frac{P(P|\bar{H})P(\bar{H})}{P(P|\bar{H})P(\bar{H}) + P(P|H)P(H)} \tag{3}$$

$$= \frac{0.99 \times 10^{-4}}{0.99 \times 10^{-4} + 0.01 \times (1 - 10^{-4})} = \frac{1}{102} = 0.98\% \tag{4}$$

2. We can rewrite the table as follows: It follows:

	S=0	S=1	
C=1	P(S=0 ∩ C=1)	P(S=1 ∩ C=1)	P(C=1)
C=0	P(S=0 ∩ C=0)	P(S=1 ∩ C=0)	P(C=0)
	P(S=0)	P(S=1)	1

	S=0	S=1	
C=1	$\frac{23}{151}$	$\frac{34}{151}$	$\frac{57}{151}$
C=0	$\frac{41}{151}$	$\frac{53}{151}$	$\frac{94}{151}$
	$\frac{64}{151}$	$\frac{87}{151}$	1

(a) $\hat{p}(C=1, S=1) = P(C=1 \cap S=1) = \frac{34}{151} = 22.52\%$

(b) $\hat{p}(C=1|S=1) = \frac{P(C=1 \cap S=1)}{P(S=1)} = \frac{34}{87} = 39.08\%$

(c) $\hat{p}(C=0|S=0) = \frac{P(C=0 \cap S=0)}{P(S=0)} = \frac{41}{64} = 64.06\%$

3. A hat over a parameter denotes an estimator of the parameter. The estimator estimates a probability value of an event based on a limited amount of samples whereas the actual probability of an event is given by looking at all samples.

4.2 Maximum Likelihood Estimation

1. If the observations are independent then the likelihood function is given by the product of the probability mass function which is the Poisson distribution in this case:

$$l(\lambda, G_1, \ldots, G_N) = \prod_{i=1}^{N} \frac{\lambda^{G_i}}{G_i!} \exp(-\lambda) \tag{5}$$

Therefore the log-likelihood function is given applying the (natural) logarithm:

$$\begin{aligned} L(\lambda, G_1, \ldots, G_N) &= \ln\left(\prod_{i=1}^{N} \frac{\lambda^{G_i}}{G_i!} \exp(-\lambda)\right) = \sum_{i=1}^{N} \left[\ln\left(\frac{\lambda^{G_i}}{G_i!} \exp(-\lambda)\right)\right] \\ &= \sum_{i=1}^{N} \left[\ln(\lambda^{G_i}) - \ln(G_i!) + \ln(\exp(-\lambda))\right] \\ &= \sum_{i=1}^{N} \left[G_i \ln(\lambda) - \ln(G_i!) - \lambda\right] \end{aligned} \tag{6}$$

2. In order to find the MLE for λ we need to solve

$$\hat{\lambda} = \arg\max_{\lambda}(L(\lambda, G_1, \ldots, G_N)) \tag{7}$$

by taking the derivative in respect to λ:

$$\frac{\partial}{\partial \lambda}\left(L(\lambda, G_1, \ldots, G_N)\right) = 0. \tag{8}$$

$$\begin{aligned} &\Leftrightarrow 0 = \frac{\partial}{\partial \lambda}\left(\sum_{i=1}^{N} \left[G_i \ln(\lambda) - \ln(G_i!) - \lambda\right]\right) \\ &\Leftrightarrow 0 = \sum_{i=1}^{N} \left[\frac{G_i}{\hat{\lambda}} - 1\right] = -N + \frac{1}{\lambda} \sum_{i=1}^{N} G_i \\ &\Leftrightarrow \hat{\lambda} = \frac{1}{N} \sum_{i=1}^{N} G_i. \end{aligned} \tag{9}$$

3. The MLE for λ using the observed G is therefore:

$$\hat{\lambda} = \frac{1}{N} \sum_{i=1}^{N} G_i. = \frac{1}{8}(6+4+2+7+5+1+2+5) = 4 \tag{10}$$

References

/