CS 5350/6350: Machine Learining Fall 2015

Homework 5
Christopher Mertin

Handed out: Nov 20, 2015
Due date: Dec 8, 2015

General Instructions

- You are welcome to talk to other members of the class about the homework. I am more concerned that you understand the underlying concepts. However, you should write down your own solution. Please keep the class collaboration policy in mind.

- Feel free ask the instructor or the TAs questions about the homework.

- Your written solutions should be brief and clear. You need to show your work, not just the final answer, but you do *not* need to write it in glory detail. Your assignment should be **no more than 10 pages**. Every extra page will cost a point.

- Handwritten solutions will not be accepted.

- The homework is due by midnight of the due date. Please submit the homework on Canvas.

Notation

It is important that I define two different notations that are used in this homework assignment. I used this notation to make the math work easier, though my final results I remove this notation

Einstein Summation Notation

This simply removes the sum as the sum is <u>understood</u> for the context of the problem. Example:

$$\sum_{i=1}^{n} \zeta_i = \zeta_i$$

Mertin Product Notation

This removes the product symbol infront of some terms as it is <u>understood</u> from the context of the problem. I made a variation to make it different when compared to *Einstein Summation Notation* so you can tell the difference (note the tilde)

$$\prod_{i=1}^{n} \zeta_i = \underset{\sim}{\zeta_i}$$

1

1   Naïve Bayes

Consider the Boolean function $f_{TH(3,7)}$. This is a threshold function defined on the 7 dimensional Boolean cube as follows: given an instance x, $f_{TH(3,7)}(x) = 1$ if and only if 3 or more of x's components are 1.

1. [4 points] Show that $f_{TH(3,7)}$ has a linear decision surface over the 7 dimensional Boolean cube.

   - The weight vector can be defined as such $\mathbf{w} = (1, 1, 1, 1, 1, 1, 1)^T$, with the bias, θ, can be defined such that $\theta = -3$. This will give the result that if $\mathbf{w}^T \mathbf{x}_i + \theta \geq 0$, then $y = 1$, otherwise $y = -1$. This will result in the bounds such that if at least 3 of the components are 1, then the result would be $y = 1$.

2. [7 points] Assume that you are given data sampled according to the uniform distribution over the Boolean cube $\{0,1\}^7$ and labeled according to $f_{TH(3,7)}$. Use naïve Bayes to learn a hypothesis that predicts these labels. What is the hypothesis generated by the naïve Bayes algorithm? (You do not have to implement the algorithm here. You may assume that you have seen all the data required to get accurate estimates of the probabilities).

   - Let x_i be the ith input of the data in the cube for $i = \{1, 2, \ldots, n\}$. We can find the most probable hypothesis by

$$p(h|\mathcal{D}) = \arg \max_{h \in H} p\left(y_i | x_i\right) \tag{1}$$

   The question states that we are sampling from a normal distribution over the cube, we can represent all the data in the following way

$$\Rightarrow \arg \max_{y \in \{0,1\}} p(y) \prod_{i=1}^{7} p\left(x_i | y\right) \tag{2}$$

   which expands into the form

$$\max \left[p(0) \prod_{i=1}^{7} p\left(x_i | 0\right), p(1) \prod_{i=1}^{7} p\left(x_i | 1\right) \right] \tag{3}$$

$$p(0) = \text{probability that at least 5 elements are 0} \tag{4}$$

$$= \left(\frac{1}{2}\right)^7 \sum_{i=5}^{7} \binom{7}{i} = \frac{29}{128} \tag{5}$$

$$p(1) = 1 - p(0) = \frac{99}{128} \tag{6}$$

We now need to define $p(x_i = 0|y)$ and $p(x_i = 1|y)$, for $y = \{0, 1\}$. These would be the same for all values of i since they come from the same distribution

$$p(x_i|0) = \frac{p(x_i)p(0|x_i)}{p(0)} \tag{7}$$

For $x_i = 1$, we need to calculate the probability of $p(0|x_i = 1)$, which is the probability of *at least* 5 of the remaining 6 x_i's being 0

$$p(0|x_i = 1) = \left(\frac{1}{2}\right)^6 \left[\binom{6}{5} + \binom{6}{6}\right] = \frac{7}{64} \tag{8}$$

Now we can calculate $p(x_i = 1|0)$ from Equation (7)

$$p(x_i = 1|0) = \frac{\frac{1}{2}\frac{7}{64}}{\frac{29}{128}} = \frac{7}{29} \tag{9}$$

Where we can do the same thing for $x_i = 0$, which would be the probability of at least 4 of the remaining 6 being 0

$$p(0|x_i = 0) = \left(\frac{1}{2}\right)^6 \left[\sum_{i=4}^{6}\binom{6}{i}\right] = \frac{22}{64} \tag{10}$$

where from here we can calculate $p(x_i = 0|0)$ from Equation (7) again as being

$$p(x_i = 0|0) = \frac{\frac{1}{2}\frac{22}{64}}{\frac{29}{128}} = \frac{22}{29} \tag{11}$$

Finally, we need to calculate it for $y = 1$, whcih can be done as follows

$$p(1|x_i = 1) = 1 - p(0|x_i = 1) = 1 - \frac{7}{64} = \frac{57}{64} \tag{12}$$

$$p(1|x_i = 0) = 1 - p(0|x_i = 0) = 1 - \frac{22}{64} = \frac{42}{64} \tag{13}$$

Leading to the final results of

$$p(x_i = 1|1) = \frac{\frac{1}{2}\frac{57}{64}}{\frac{99}{128}} = \frac{57}{99} \tag{14}$$

$$p(x_i = 0|1) = \frac{\frac{1}{2}\frac{42}{64}}{\frac{99}{128}} = \frac{42}{99} \tag{15}$$

From these we can build the hypothesis

$$h(\mathbf{x}) = \arg\max_{y \in \{0,1\}} \frac{29 + 70y}{128} \prod_{i=1}^{7} \frac{42 + 15\mathbf{x}_i}{99}y - \frac{22 - 15\mathbf{x}_i}{29}(y - 1) \tag{16}$$

3

3. [4 points] Show that the hypothesis produced in the previous question does not represent this function.

- This can be proven via *proof by contradiction*. Assume we have the vector $\mathbf{x} = (0, 0, 0, 1, 0, 0, 1)^T$ and that the hypothesis in the answer to the previous question correctly classifies the data. From the first question, we know that this should be classified as $y = 0$. We can test this to verify

$$h(\mathbf{x}) = \max_{y \in \{0,1\}} \begin{cases} \frac{29}{128} \left[\left(\frac{22}{29}\right)^5 \left(\frac{7}{29}\right)^2 \right] \approx 0.003317 & (y = 0) \\ \frac{99}{128} \left[\left(\frac{57}{99}\right)^2 \left(\frac{42}{99}\right)^5 \right] \approx 0.003524 & (y = 1) \end{cases} \tag{17}$$

which the hypothesis would choose the largest number which would be $y = 1$, which isn't true as it should be $y = 0$. This is a contradiction, therefore it doesn't describe this function.

4. [5 points] Are the naïve Bayes assumptions satisfied by $f_{TH(3,7)}$? Justify your answer.

- No, the naïve bayes assumptions are not satisfied by $f_{TH(3,7)}$. The assumption that is made is that $\mathbf{x}_i$ is independent for each value of i, which isn't actually the case. Any of the probability calculations should include the probability of the other features as well.

2 EM Algorithm

There are two grocery stores in the neighborhood of the U: Smith's and Trader Joe's. Each store has n checkout lanes. The number of customers for each lane per unit time, say one day, is distributed according to Poisson distribution with parameter λ. That is, for the i'th checkout lane,

$$P(\# \text{ of customers for lane } i = x_i | \lambda) = \frac{\lambda^{x_i} e^{-\lambda}}{x_i!}$$

with parameters λ_S and λ_T for Smith's and Trader Joe's, respectively.

1. [10 points] Given a record of customer counts $(x_1, \cdots, x_n)$, where x_i denotes the number of customers went through lane i, what is the most likely value of λ?

- The joint probability density function of $p(x_i | \lambda)$ is the product of each probabilty for $i = \{1, 2, \ldots, n\}$

$$p(x | \lambda) = \prod_{i=1}^{n} p(x_i | \lambda) = \prod_{i=1}^{n} \frac{\lambda^{x_i} e^{-\lambda}}{x_i!} \tag{18}$$

$$= \frac{\lambda^{\sum_{i=1}^{n} x_i} e^{-n\lambda}}{\prod_{i=1}^{n} x_i!} \tag{19}$$

4

which we can take the log of now to get the log-likelihood function

$$\ell(\theta) = \log\left[\frac{\lambda^{\sum_{i=1}^{n} x_i} e^{-n\lambda}}{\prod_{i=1}^{n} x_i!}\right] \tag{20}$$

$$= \log\left[\left(\sum_{i=1}^{n} x_i\right)\log(\lambda) + (-n\lambda)\log(e) - \prod_{i=1}^{n}\log(x_i!)\right] \tag{21}$$

where we want to maximize λ, so we take $\frac{\partial \ell(\theta)}{\partial \lambda}$ and set it equal to zero

$$0 = \frac{\partial \ell(\theta)}{\partial \lambda} = \frac{\sum_{i=1}^{n} x_i}{\lambda} - n \tag{22}$$

$$n\lambda = \sum_{i=1}^{n} x_i \tag{23}$$

$$\lambda = \frac{1}{n}\sum_{i=1}^{n} x_i \tag{24}$$

2. [10 points] Assume now that you are given a collection of m records $\{(x_{j1}, \cdots x_{jn})\}$, where $j = 1, \cdots, m$. You do not know which record is from Smith's and which is from Trader Joe's. Assume that the probability of a record is from the Smith's is η. In other words, it means that the probability that a record is from Trader Joe's is $1 - \eta$. Explain the generative model that governs the generation of this data collection. In doing so, name the parameters that are required in order to fully specify the model.

   • The data that is given to us is such that $i = \{1, 2, \ldots, n\}$ and $j = \{1, 2, \ldots, m\}$, where m is the number of records and n is the number of lanes. The generative model that can be used for this is

$$h(\mathbf{x}) = \arg\max_{y \in \{S, T\}} p(y) \prod_{j=1}^{m}\sum_{i=1}^{n} p(y|x_i) \tag{25}$$

$$= \arg\max_{y \in \{S, T\}} p(y) \prod_{j=1}^{m}\sum_{i=1}^{n} \frac{p(x_i|y)}{p(x_i)} \tag{26}$$

   We don't know the distribution of the m records, but we do know that there is a probability η that a record represents data from Smith's and $(1 - \eta)$ that it represents data from Trader Joe's. We can calculate the maximum likelyhood by Equation (26) by splitting the max for $y \in \{S, T\}$. The base equation used to represent the probability of *each record* is

$$h(\mathbf{x}) = \arg\max_{y \in \{S, T\}} \sum_{i=1}^{n} p(y)p(x_i|y) \tag{27}$$

5

which would be for a single data set. Therefore, to account for each record, the resulting equation would be

$$h(\mathbf{x}) = \arg\max_{y \in \{S,T\}} \begin{cases} \eta \prod_{j=i}^{m} \frac{\lambda_S^{\sum_{i=1}^{n} x_{j,i}} e^{-n\lambda_S}}{\prod_{i=1}^{n} x_{j,i}!} \\ (1-\eta) \prod_{j=i}^{m} \frac{\lambda_T^{\sum_{i=1}^{n} x_{j,i}} e^{-n\lambda_T}}{\prod_{i=1}^{n} x_{j,i}!} \end{cases} \tag{28}$$

To use $h(\mathbf{x})$ to classify the data, the parameters we would need would be η to tell the probability of which belongs to S or T, and we would need an initial value of λ which would be the value that we use in the first iteration for *both* λ_S and λ_T. One way to do this would be to either pick a random number or to take the average of the first record. The way that the EM algorithm can be used to cluster the records is described in the next part.

3. [10 points] Assume that you are given the parameters of the model described above. How would you use it to cluster records to two groups, the Smith's and the Trader Joe's?

- In order to cluster the data into two sets, we would take the first record and plug it into the two different cases of $h(\mathbf{x})$ and the largest result would correspond to that label.

  For example, if the kth record when plugged into $h(\mathbf{x})$ is largest for $y = S$, then that data set would be treated as belonging to Smith's. Following this, we would then need to update λ_S in the following way

$$\lambda_S = \frac{1}{k \cdot n} \sum_{j=1}^{k} \sum_{i=1}^{n} x_{j,i} \tag{29}$$

  where k represents the number of records that corresponded to S. We would iterate through all m records to get this result and continually update λ_S and λ_T until we got the best fit of the distributions.

4. [10 points] Given the collection of m records without labels of which store they came from, derive the update rule of the EM algorithm. Show all of your work.

- We can find the decision boundary in the following way. The boundary line is defined as:

$$p(\xi) = \frac{p(y = S|x_{j,i})}{p(y = T|x_{j,i})} = \frac{p(x_{j,i}|y = S)p(T)}{p(x_{j,i}|y = T)P(S)} > 1 \tag{30}$$

where $p(\xi)$ is being used to shorten the notation. We can plug in what we know for $p\left(x_{j,i}|y=\{S,T\}\right)$, $p(S)$, and $p(T)$, resulting in

$$p(\xi) = \frac{(1-\eta)}{\eta} \frac{\frac{\lambda_S^{\sum_{i=1}^n x_{j,i}} e^{-n\lambda_S}}{\prod_{i=1}^n x_{j,i}!}}{\frac{\lambda_T^{\sum_{i=1}^n x_{j,i}} e^{-n\lambda_T}}{\prod_{i=1}^n x_{j,i}!}} \tag{31}$$

Which can be simplified in the following way. Since the exponential terms are independent of j, they can be pulled outside of the project. And since the denominators are the same in each of the probabilities, we can cancel them. The following is the much more simplified result using *Einstein Summation Notation* and *Mertin Product Notation*

$$p(\xi) = \underbrace{\frac{(1-\eta)}{\eta}}_{\alpha} \underbrace{\frac{e^{-nm\lambda_S}}{e^{-nm\lambda_T}}}_{\beta} \underbrace{\frac{\lambda_S^{x_{j,i}}}{\lambda_T^{x_{j,i}}}}_{\gamma_{j,i}} > 1 \tag{32}$$

which the substitutions in the underbraces can be made to make the simplification process easier, resulting in

$$p(\xi) = \alpha\beta\underbrace{\gamma_{j,i}} > 1 \tag{33}$$

Following this, the log of both sides can be taken, resulting in

$$p(\xi) = \log(\alpha) + \log(\beta) + \log\left(\underbrace{\gamma_{j,i}}\right) > \log(1) \tag{34}$$

$$p(\xi) = \log(\alpha) + \log(\beta) + \log\left(\underbrace{\gamma_{j,i}}\right) > 0 \tag{35}$$

Where we can plug in back the values of α, β, and $\gamma_{j,i}$

$$p(\xi) = \log\left(\underbrace{\lambda_S^{x_{j,i}}}\right) - \log\left(\underbrace{\lambda_T^{x_{j,i}}}\right) + \log\left(\frac{(1-\eta)}{\eta}\right) + \log\left(\frac{e^{-nm\lambda_S}}{e^{-nm\lambda_T}}\right) > 0 \tag{36}$$

$$p(\xi) = x_{j,i}\log\left(\underbrace{\frac{\lambda_S}{\lambda_T}}\right) + \log\left(\frac{(1-\eta)}{\eta}\right) + nm\left(\lambda_T - \lambda_S\right) > 0 \tag{37}$$

where going back to the normal notation gives us the definition of the update

$$p(\xi) = \underbrace{\prod_{j=1}^m \sum_{i=1}^n x_{j,i}\log\left(\frac{\lambda_S}{\lambda_T}\right)}_{\mathbf{w}^T\mathbf{x}} + \underbrace{\log\left(\frac{(1-\eta)}{\eta}\right) + nm\left(\lambda_T - \lambda_S\right)}_{b} > 0 \tag{38}$$

7

3 Experiment

We looked maximum a posteriori learning of the logistic regression classifier in class. In particular, we showed that learning the classifier is equivalent to the following optimization problem:

$$\min_{\mathbf{w}} \left\{ \sum_{i=1}^{m} \log\left(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)\right) + \frac{1}{\sigma^2} \mathbf{w}^T \mathbf{w} \right\}$$

In this question, you will derive the stochastic gradient descent algorithm for the logistic regression classifier, and also implement it with cross-validation. Detailed instructions on cross-validation procedure can be found in homework 1, and instructions on SGD can be found in homework 4.

1. [5 points] What is the derivative of the function $\log\left(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)\right)$ with respect to the weight vector?

   - We can simply treat $\mathbf{w}$ as a variable and it becomes a normal differentiation problem

   $$f(\mathbf{x}_i) = \log\left(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)\right) \tag{39}$$

   where we can perform a u-substition such as

   $$u = 1 + \exp\left(-y_i \mathbf{w}^T \mathbf{x}_i\right) \tag{40}$$

   $$\frac{\mathrm{d}u}{\mathrm{d}\mathbf{w}^T} = -y_i \mathbf{x}_i \exp\left(-y_i \mathbf{w}^T \mathbf{x}_i\right) \tag{41}$$

   such that we can use these to differentiate $f(\mathbf{x}_i)$ to be

   $$\frac{\partial f(\mathbf{x}_i)}{\partial \mathbf{w}^T} = \frac{\partial f(u)}{\partial \mathbf{w}^T} \frac{\partial u}{\partial \mathbf{w}^T} = \frac{-y_i \mathbf{x}_i}{\exp\left(y_i \mathbf{w}^T \mathbf{x}_i\right) + 1} \tag{42}$$

   To overcome instances of *overflow errors* in the computations, this equation can be rewritten in the following (equivalent) format from when Equation (42) overflows

   $$\frac{\partial f(\mathbf{x}_i)}{\partial \mathbf{w}^T} = \frac{(-y_i \mathbf{x}_i) \cdot \exp(-y_i \mathbf{w}^T \mathbf{x}_i)}{1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)} \tag{43}$$

2. [5 points] The inner most step in the SGD algorithm is the gradient update where we use a single example instead of the entire dataset to compute the gradient. Write down the objective where the entire dataset is composed of a single example, say $(\mathbf{x}_i, y_i)$. Derive the gradient of this objective with respect to the weight vector.

   - We want the weight vector that minimizes our function, $J(\mathbf{w})$, so we need to take the gradient of it, but first we can *simplify* $J(\mathbf{w})$ to make the computation easier, which results in

$$J(\mathbf{w}) = \sum_{i=1}^{n} \log\left(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)\right) + \frac{\mathbf{w}^T \mathbf{w}}{\sigma^2} \tag{44}$$

$$= \sum_{i=1}^{n} \log\left(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)\right) + \frac{||\mathbf{w}||^2}{\sigma^2} \tag{45}$$

where we can take the derivative of this with respect to $\mathbf{w}$ and drop the summation since we're treating each record as the *only* data point. The first term is the same as the result in Equation (42) so only the second term needs to be differentiated with respect to $\mathbf{w}$.

$$\frac{\partial J(\mathbf{w})}{\partial \mathbf{w}} = \frac{-y_1 \mathbf{x}_1}{\exp(y_1 \mathbf{w}^T \mathbf{x}_1) + 1} + 2\frac{||\mathbf{w}||}{\sigma^2} \tag{46}$$

3. [10 points] Write down the pseudo code for the stochastic gradient algorithm using the gradient from previous part.

   - The pseudo code is represented in Algorithm 1, where γ is the learning rate where $C = \sigma^2$, and there's a vector $\mathbf{x}_i \in \mathbb{R}^{n \times 1}$ in a set S with m different "records." t ranges from 1 to $(T \cdot m)$, where T is the maximum number of epochs, since $\mathbf{w}$ isn't initialized to $\vec{0}$ at the start of each epoch.

Algorithm 1 Stochastic Gradient Descent$(S = \{(\mathbf{x}_i, y_i)\}_m)$

Initialize $\mathbf{w}_{(0)} \in \mathbb{R}^{n \times 1}$
Initialize list $\mathbf{w}_{list} \in \mathbb{R}^{T \times n}$
for $epoch = 1, 2, \ldots$ **to** T **do**
    for all $(\mathbf{x}_i, y_i) \in S$ **do**
        Pick random example $(\mathbf{x}_i, y_i)$
        Treat $(\mathbf{x}_i, y_i)$ as a full dataset and compute $\vec{\nabla} J\left(\mathbf{w}_{(t)}\right)$ from Equation (46)
        $\gamma_{(t)} = \frac{\gamma_0}{1 + (\gamma_0 \cdot t)/C}$
        $\mathbf{w}_{(t+1)} = \mathbf{w}_{(t)} - \gamma_{(t)} \vec{\nabla} J\left(\mathbf{w}_{(t)}\right)$
    end for
    Append $\mathbf{w}$ from current *epoch* to $\mathbf{w}_{list}$
end for
return $\min_{\mathbf{w} \in \mathbf{w}_{list}} \left\{ \sum_{i=1}^{m} \log\left(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)\right) + \frac{||\mathbf{w}||^2}{\sigma^2} \right\}$

---

4. [20 points] Implement your pseduo code as a training algorithm with cross-validation on the σ parameter. This parameter basically helps trade off between generalizability and model fit.

   Use the two `astro` data sets (original and scaled) from homework 4 to train and evaluate the learner. In your writeup, please report on the accuracy of your system, what value of sigma you chose based on cross validation, how many epochs you chose to run SGD, and a plot of the NEGATIVE log likelihood after each epoch of SGD.

- In order to implement Stochastic Gradient Descent, Algorithm 1 was used. In doing so, there was a hyper parameter σ which needed to be chosen. However, the "learning rate" $\gamma_{(t)}$ was dependent upon an initial value, γ_0, and some parameter C where $C = \sigma^2$. This learning rate changed over time and was defined as $\frac{\gamma_0}{1+(\gamma_0 \cdot t)/C}$ which was important as the learning rate would decrease over time as the function presumably as it approached the maxima. These hyper parameters were *also* used to get the general range of what they should be. Table 1 indicates the ranges that were initially trained over to see what the best values to use were. In the table, it only lists the minimum and maximum values, though the intermediary values were powers of 10 between the min and max.

Parameter	Min	Max
σ	10^{-7}	10^3
γ_0	10^{-10}	10^{-3}

Table 1: Hyper-parameter Cross Validation Ranges

Using these ranges, 5-fold cross validation was implemented on the training data to get the best parameters for the model. Table 2 shows the top 5 parameters for the file `astro/original/train` while Table 3 shows the top 5 parameters for the file `astro/scaled/train`. The *best* value was chosen for each file, which was then used to train over the *entire* training set and then used to classify the test data. During Cross Validation, only 5 epochs were chosen and it was only 5-fold cross validation

Accuracy	γ_0	σ
0.91229	10^{-7}	100
0.91196	10^{-7}	10
0.91196	10^{-7}	0.1
0.91164	10^{-7}	1000
0.91099	10^{-7}	1.00

Table 2: Top 5 hyper-parameter combinations for `astro/original/train`

Accuracy	γ_0	σ
0.54298	10^{-8}	0.001
0.54298	10^{-8}	0.001
0.54298	10^{-8}	0.001
0.54298	10^{-7}	0.001
0.54298	10^{-6}	0.001

Table 3: Top 5 hyper-parameter combinations for `astro/scaled/train`

As can be seen in the tables, in the original data σ varied while the other parameters stayed the same. The opposite was true for the scaled data, where σ remained

constant and the other parameters varied. From using the best hyper-parameters, where 10 epochs for each data set was chosen. The test set `astro/original/test` scored a classification accuracy of 0.90625, while the scaled data did much worse and `astro/scaled/test` had a classification accuracy of 0.50000. These results agree pretty well with the last homework assignment which is reassuring on the implemented algorithm working correctly. Figure 4 shows the negative log likelihood function of each of the test files at the end of each epoch.
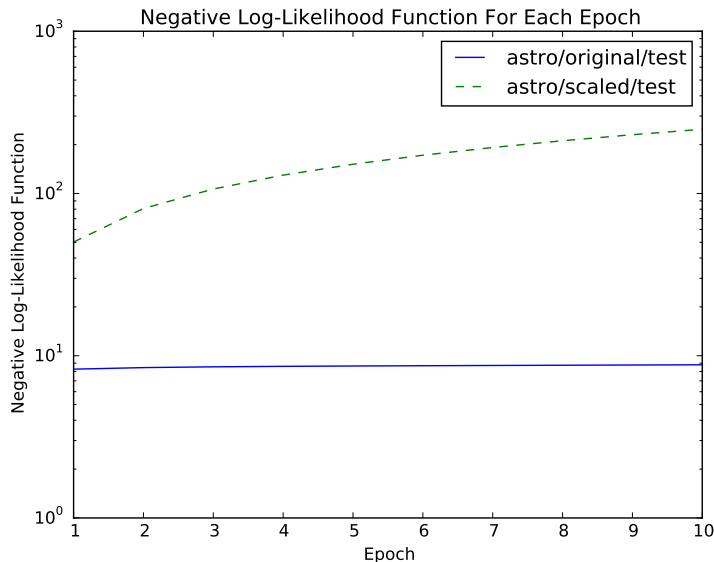

Figure 1: Negative Log Likelihood Function of $J(\mathbf{w})$

As mentioned in previous homeworks, you may use any programming language for your implementation. Upload your code along with a script so the TAs can run your solution in the CADE environment.

4   HAPPY HOLIDAYS Extra Credit

25pts You've seen stochastic gradient decent applied to logistic regression. Now we ask why this is a viable strategy for optimizing this objective. Prove that SGD will find the optimal value for this function. This can be done by demonstrating that the objective is convex. There are many ways to prove this. One of the most straightforward ways to show this is demonstrate that the Hessian is positive-semidefninite.

1. [5 points] Find the Gradient of:

$$\sum_{i=1}^{m} \log\left(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)\right) + \frac{1}{\sigma^2} \mathbf{w}^T \mathbf{w}$$

11

- The gradient of any function is defined as

$$\vec{\nabla} f(\mathbf{x}) = \sum_{i=1}^{\mathcal{D}} \frac{\partial f(\mathbf{x})}{\partial x_i} \hat{e}_i \tag{47}$$

where $\mathcal{D}$ is the dimensionality/number of dependent variables in $f(\mathbf{x})$ and $\hat{e}_i$ is the normal vector for that coordinate – a value of 1 for all dimensions in cartesian space. This requires a derivative over $\mathbf{x}$ and $\mathbf{w}$ which is

$$\vec{\nabla} f(\mathbf{x}_i) = \sum_{i=1}^{\mathcal{D}} \frac{\partial f(\mathbf{x}_i)}{\partial \mathbf{w}} + \frac{\partial f(\mathbf{x}_i)}{\partial \mathbf{x}_i} \tag{48}$$

We can rewrite our equation $f(\mathbf{x}_i)$ to make it easier to differentiate

$$f(\mathbf{x}_i) = \sum_{i=1}^{n} \log\left(1 + \exp(-y_i \mathbf{w}^T \mathbf{x}_i)\right) + \frac{||\mathbf{w}||^2}{\sigma^2} \tag{49}$$

where the derivative of a sum of a function is the sum of the derivative of that function, so the differentials can be brought inside the sum and the respected differentials can be taken and the partial derivative outside the braces denotes that segment representing that differential of $f(\mathbf{x}_i)$

$$\vec{\nabla} f(\mathbf{x}_i) = \left\{ \sum_{i=1}^{n} \frac{-y_i \mathbf{x}_i}{\exp(y_i \mathbf{w}^T \mathbf{x}_i) + 1} + \frac{2\,||\mathbf{w}||}{\sigma^2} \right\}_{\frac{\partial f}{\partial \mathbf{w}}} + \left\{ \sum_{i=1}^{n} \frac{-y_i \mathbf{w}^T}{\exp(y_i \mathbf{w}^T \mathbf{x}_i) + 1} \right\}_{\frac{\partial f}{\partial \mathbf{x}}} \tag{50}$$

2. [5 points] Find the Hessian of (a).

- The Hessian Matrix is defined as

$$\mathcal{H}\left(f(\mathbf{x}_i)\right) = \begin{bmatrix} \frac{\partial^2 f}{\partial \mathbf{x}_i^2} & \frac{\partial^2 f}{\partial \mathbf{x}_i \partial \mathbf{w}} \\ \frac{\partial^2 f}{\partial \mathbf{w} \partial \mathbf{x}_i} & \frac{\partial^2 f}{\partial \mathbf{w}^2} \end{bmatrix} \tag{51}$$

$$\frac{\partial^2 f(\mathbf{x}_i)}{\partial \mathbf{x}_i^2} = \sum_{i=1}^{n} \frac{||\mathbf{w}||^2 y_i^2 e^{\varsigma_i}}{\left(e^{\varsigma_i} + 1\right)^2} \tag{52}$$

$$\frac{\partial^2 f(\mathbf{x}_i)}{\partial \mathbf{x}_i \partial \mathbf{w}} = \sum_{i=1}^{n} \frac{y_i \left[e^{\varsigma_i} \left(\varsigma_i - 1\right) - 1\right]}{\left(e^{\varsigma_i} + 1\right)^2} \tag{53}$$

$$\frac{\partial^2 f(\mathbf{x}_i)}{\partial \mathbf{w} \partial \mathbf{x}_i} = \frac{\partial^2 f(\mathbf{x}_i)}{\partial \mathbf{x}_i \partial \mathbf{w}} = \sum_{i=1}^{n} \frac{y_i \left[e^{\varsigma_i} \left(\varsigma_i - 1\right) - 1\right]}{\left(e^{\varsigma_i} + 1\right)^2} \tag{54}$$

$$\frac{\partial^2 f(\mathbf{x}_i)}{\partial \mathbf{w}^2} = \sum_{i=1}^{n} \frac{||\mathbf{x}_i||^2 y_i^2 e^{\varsigma_i}}{\left(e^{\varsigma_i} + 1\right)^2} + \frac{2}{\sigma^2} \tag{55}$$

where $\varsigma_i = \mathbf{w}^T \mathbf{x}_i y_i$

12

3. [10 points] prove that the Hessian is positive semidefinite.

- A matrix is *positive semidefinite* if and only if $\mathbf{b}^T \mathcal{H} \mathbf{b} > \vec{0} \ \forall \ \mathbf{b} \neq \vec{0}$, where $\mathbf{b}$ is a vector. $\mathcal{H}(f(\mathbf{x}_i))$ can be re-written as, by using Einstein Summation Notation

$$\mathcal{H} = \begin{bmatrix} \frac{||\mathbf{w}||^2 y_i^2 e^{\varsigma_i}}{(e^{\varsigma_i}+1)^2} & \frac{y_i[e^{\varsigma_i}(\varsigma_i-1)-1]}{(e^{\varsigma_i}+1)^2} \\ \frac{y_i[e^{\varsigma_i}(\varsigma_i-1)-1]}{(e^{\varsigma_i}+1)^2} & \frac{||\mathbf{x}_i||^2 y_i^2 e^{\varsigma_i}}{(e^{\varsigma_i}+1)^2} + \frac{2}{\sigma^2} \end{bmatrix} \tag{56}$$

where $\varsigma_i = \mathbf{w}^T \mathbf{x}_i y_i$

We only care about if it is going to be negative, so without loss of generality we can remove all the norms, squares, and denominators since they are *always* positive. We can also generalize our vector $\mathbf{b}$ as $\mathbf{b} = (\alpha_1, \alpha_2)^T$, where α_1 and α_2 can be anything

$$\mathcal{H}' = \begin{bmatrix} e^{\varsigma_i} & e^{\varsigma_i}(\varsigma_i - 1) - 1 \\ e^{\varsigma_i}(\varsigma_i - 1) - 1 & e^{\varsigma_i} + \frac{2}{\sigma^2} \end{bmatrix} \tag{57}$$

where $\mathbf{b}^T \mathcal{H}' \mathbf{b}$ can be generalized with the following result

$$\mathbf{b}^T \begin{bmatrix} x_1 & x_2 \\ x_3 & x_4 \end{bmatrix} \mathbf{b} = x_1 \alpha_1^2 + x_4 \alpha_2^2 + \alpha_1 \alpha_2 (x_2 + x_3) \tag{58}$$

when applied to $\mathcal{H}'$ results in

$$\mathbf{b}^T \mathcal{H}' \mathbf{b} = \alpha_1^2 e^{\varsigma_i} + \alpha_2^2 \left(e^{\varsigma_i} + \frac{2}{\sigma^2} \right) + 2\alpha_1 \alpha_2 \left(e^{\varsigma_i}(\varsigma_i - 1) - 1 \right) \tag{59}$$

Where there's two instances that we need to prove. (1) If $\alpha_1, \alpha_2 > 0$ then $\alpha_1 \alpha_2 (x_2 + x_3) > 0$ and (2) if $\alpha_1 > 0$ and $\alpha_2 < 0$, then $\alpha_1^2 x_1 + \alpha_2^2 x_2 > \alpha_1 \alpha_2 (x_2 + x_3)$.

$$\alpha_1, \alpha_2 > 0 \Rightarrow \alpha_1 \alpha_2 (x_2 + x_3) > 0 \tag{60}$$

$$\alpha_1 \alpha_2 (x_2 + x_3) = 2\alpha_1 \alpha_2 \varsigma_i e^{\varsigma_i} - 2e^{\varsigma_i} - 2 > 0 \tag{61}$$

$$\underbrace{\varsigma_i e^{\varsigma_i} - e^{\varsigma_i} - 1}_{> \ 0 \ \text{if} \ \varsigma_i e^{\varsigma_i} > e^{\varsigma_i} - 1}; \quad \varsigma_i \neq 0 \tag{62}$$

$$\Rightarrow \varsigma_i e^{\varsigma_i} > e^{\varsigma_i} - 1 \tag{63}$$

$$\Rightarrow \varsigma_i > 1 - e^{-\varsigma_i} \quad \square \tag{64}$$

where $\varsigma_i \geq 1 \ \forall \ i$. Now for the alternative case where if $\alpha_1 > 0$ and $\alpha_2 < 0$, that $\alpha_1^2 x_1 + \alpha_2^2 x_2 > \alpha_1 \alpha_2 (x_2 + x_3)$

$$\alpha_1^2 e^{\varsigma_i} + \alpha_2^2 \left(e^{\varsigma_i} + \frac{2}{\sigma^2} \right) > \alpha_1 \alpha_2 [e^{\varsigma_i}(\varsigma_i - 1) - 1] \tag{65}$$

$$\left[\alpha_1^2 + \alpha_2^2 + \frac{2}{\sigma^2 e^{\varsigma_i}} \right] > [\alpha_1 \alpha_2 (\varsigma_i - 1) - e^{-\varsigma_i}] \tag{66}$$

13

if $\varsigma_i \gg 1$

$$\underbrace{\alpha_1^2 + \alpha_2^2}_{> \ 0} > \underbrace{\alpha_1\alpha_2}_{< \ 0}\underbrace{(\varsigma_i - 1)}_{> \ 0} \qquad \square \tag{67}$$

for $\varsigma_i \geq 0$ (always)

$$\underbrace{\alpha_1^2 + \alpha_2^2}_{> \ 0} + \underbrace{e^{-\varsigma_i}}_{\geq \ 1}\underbrace{\left(\frac{2}{\sigma^2} + 1\right)}_{> \ 0} > \underbrace{\alpha_1\alpha_2}_{< \ 0}\underbrace{(\varsigma_i - 1)}_{> \ 0} \qquad \square \tag{68}$$

4. [5 points] Why does this prove that we are GUARANTEED to have within $\epsilon > 0$ of the right answer?

- Because the Hessian Matrix of the function is positive-semidefinite, the given function is convex. On top of which, since we are maximizing the function, we are required to step with the gradient and stop when the slope is less than some tolerance (ϵ). The gradient gives the slope/direction of the greatest change and we "walk" in teh direction which makes the gradient zero. We are guarenteed that one exists since the Hessian is positive-semidefinite.