Homework #2: Machine Learning

Kyler Little

February 8, 2018

Problem #1

(a) More generally, if we are learning from ±1 data to predict a noisy target $P(y|\boldsymbol{x})$ with candidate hypothesis h, show that the maximum likelihood method reduces to the task of finding h that minimizes:

$$E_{\text{in}}(\boldsymbol{w}) = \sum_{n=1}^{N} [\![y_n = +1]\!] \ln(\frac{1}{h(\boldsymbol{x}_n)}) + [\![y_n = -1]\!] \ln \frac{1}{1 - h(\boldsymbol{x}_n)}$$

Formally, we are trying to learn the target function:

$$f(\boldsymbol{x}) = P[y = +1|\boldsymbol{x}]$$

In non-mathematical terms, this target function represents the probability of getting $y_n = +1$ from the data ($\boldsymbol{x}_n$). The function f is generated by a noisy target function, so the data doesn't give us the value of f explicitly. Instead, the data is generated by a noisy target function $P(y|\boldsymbol{x})$. Our goal is to minimize the error between the classifications of actual training data and the predicted classifications from our hypothesis model h. We can do this by talking about the notion of likelihood. The likelihood that the target distribution $P(y|\boldsymbol{x})$ is captured by our hypothesis $h(\boldsymbol{x})$ is:

$$P(y|\boldsymbol{x}) = \begin{cases} h(\boldsymbol{x}) & \text{for } y = +1; \\ 1 - h(\boldsymbol{x}) & \text{for } y = -1. \end{cases}$$

If we assume that the data points were independently generated (a very fair assumption to make), then we can express the probability of getting all of the y_n's in the data set from their corresponding x_n's as:

$$\prod_{n=1}^{N} P(y_n|\boldsymbol{x}_n)$$

The method of maximum likelihood method would select the hypothesis h which maximizes the probability above. We can actually express this maximization problem as a minimization problem by using the properties of 'ln'. The natural logarithm is a monotonically increasing function, and so taking the natural logarithm of the probability above will not affect the maximization problem. By the same token, '$-\ln$' is a monotonically decreasing function, so we can take the natural logarithm of the probability above and convert the maximization problem to a minimization problem without affecting global minima of the probability distribution. Using a simple logarithm rule ($-\ln x = \ln x^{-1}$), we find:

$$\prod_{n=1}^{N} P(y_n|\boldsymbol{x}_n) = \sum_{n=1}^{N} \ln \frac{1}{P(y_n|\boldsymbol{x}_n)}$$

Substituting in what $P(y|\boldsymbol{x})$ actually equals, we arrive at what we wanted. We now find h that minimizes:

$$E_{\text{in}}(\boldsymbol{w}) = \sum_{n=1}^{N} [\![y_n = +1]\!] \ln \frac{1}{h(\boldsymbol{x}_n)} + [\![y_n = -1]\!] \ln \frac{1}{1 - h(\boldsymbol{x}_n)}$$

(b) For the case $h(\boldsymbol{x}) = \theta(\boldsymbol{w}^T\boldsymbol{x})$, argue that minimizing the in-sample error in part (a) is equivalent to minimizing the one in (3.9).
Substitute in $h(\boldsymbol{x}) = \theta(\boldsymbol{w}^T\boldsymbol{x}) = \frac{e^{w^T\boldsymbol{x}}}{1+e^{w^T\boldsymbol{x}}}$ to the in-sample error from part (a). This yields:

$$E_{\text{in}}(\boldsymbol{w}) = \sum_{n=1}^{N} [\![y_n = +1]\!] \ln \frac{1 + e^{w^T\boldsymbol{x}}}{e^{w^T\boldsymbol{x}}} + [\![y_n = -1]\!] \ln(1 + e^{w^T\boldsymbol{x}})$$

Next, use the rule $b\log(x) = \log(x^b)$ to bring the y_n's inside the logarithms and $\log a + \log b = \log(ab)$ to reduce the two expression to a single expression:

$$E_{\text{in}}(\boldsymbol{w}) = \sum_{n=1}^{N} \ln(1 + e^{-y_n w^T\boldsymbol{x}})$$

Multiplying by a constant factor of $1/N$ will not change the minimization problem. Thus, the argument is complete.
Side Note (from LFD):
For two probabilistic distributions $p, 1-p$ and $q, 1-q$ with binary outcomes, the cross-entropy (from information theory) is:

$$p \log \frac{1}{q} + (1-p) \log \frac{1}{1-q}$$

The in-sample error in part (a) corresponds to a cross-entropy error measure on the data point $(\boldsymbol{x}_n, y_n)$, with $p = [\![y_n = +1]\!]$ and $q = h(\boldsymbol{x}_n)$.

Problem #2

Recall the objective function for linear regression can be expressed as

$$E(w) = \frac{1}{N}||Xw - y||^2$$

as in Equation (3.3) of LFD. Minimizing this function with respect to w leads to the optimal w as $(XTX)^{-1}X^Ty$. This solution holds only when X^TX is nonsingular. To overcome this problem, the following objective function is commonly minimized instead:

$$E_2(w) = ||Xw - y||^2 + \lambda||w||^2$$

where $\lambda > 0$ is a user-specified parameter. Please do the following:
(a) Derive the optimal w that minimize $E_2(w)$.
This is fairly straightforward. We differentiate $E_2(w)$, set the result to zero, and solve for w. The reason we can do this is because $E_2(w)$ is a convex function. This means its derivative is a monotonically increasing function, signifying that there is a unique global minimum. Before differentiating, it's necessary to reduce $E_2(w)$ to an expression that is more easily differentiated. To start, we recall that $||x|| = \sqrt{x^Tx}$. Applying this yields:

$$E_2(w) = \frac{1}{N}((Xw - y)^T(Xw - y) + \lambda w^Tw)$$

Expanding this out yields:

$$E_2(w) = \frac{1}{N}(w^TX^TXw - w^TX^Ty - y^TXw + y^Ty + \lambda w^Tw)$$

Note that $w^TX^Ty = (y^TXw)^T$, so we may combine the second and third terms, since they are identical real numbers after evaluation, resulting in $-w^TX^Ty - y^TXw = -2w^TX^Ty$. Now, we simply must take the derivative of $E_2(w)$, set it to 0, and solve for w. For the first term, we use the fact that $\frac{\partial(w^TAw)}{\partial w} = (A + A^T)w$. Since A is symmetric, we have that $(A + A^T)w = 2Aw$. In our situation, A is replaced by X^TX, so $\frac{\partial(w^TX^TXw)}{\partial w} = 2X^TXw$. For the second term, we know that $\frac{\partial(2w^TX^Ty)}{\partial w} = 2X^Ty$, as discussed in class. The third term disappears since it is constant with respect to w. The last term seems tricky at first, but an easy solution is illuminated by writing w^Tw as $w_1^2 + \cdots + w_n^2$. Thus, $\partial(\lambda w^Tw)/\partial w_i = 2\lambda w_i$. This holds $\forall i \in 1, \ldots, N$, so $\partial(\lambda w^Tw)/\partial w = 2\lambda w$. As a result, we have

$$\nabla E_2(w) = \frac{1}{N}(2X^TXw - 2X^Ty + 2\lambda w)$$

Now, we set $\nabla E_2(w)$ to zero and isolate w.

$$\begin{aligned}
0 &= \frac{1}{N}(2X^TXw - 2X^Ty + 2\lambda w)\\
0 &= X^TXw - X^Ty + \lambda w\\
X^Ty &= X^TXw + \lambda w\\
X^Ty &= (X^TX + \lambda I)w\\
(X^TX + \lambda I)^{-1}X^Ty &= (X^TX + \lambda I)^{-1}(X^TX + \lambda I)w\\
w &= (X^TX + \lambda I)^{-1}X^Ty
\end{aligned}$$

(b) Explain how this new objective function can overcome the singularity problem of X^TX.
We are not necessarily guaranteed that X^TX is invertible. However, we will show that

$(X^TX + \lambda I)$ is always invertible. We can do this by demonstrating that it's nonsingular. In this proof, we will show that $(X^TX + \lambda I)$ is a positive definite matrix, meaning that it's nonsingular.
First, write $(X^TX + \lambda I)$ as A^TA. It's clear that A^TA is still symmetric because X^TX is symmetric, and subtracting λI won't affect A's symmetry because it only affects the diagonal of X^TX. Now, in order for some matrix A to be positive definite, we must have that $x^TAx > 0$, $\forall x \neq 0$. Let's now prove this.

$$\begin{aligned} x^TA^TAx &> 0 \\ (Ax)^TAx &> 0 \\ ||Ax||^2 &> 0 \end{aligned}$$

Why is $||Ax||^2$ strictly greater than zero? It's obvious that $||Ax||^2$ is greater than or equal to zero because the Euclidean norm is nonnegative. However, we can guarantee that Ax cannot be equal to zero because subtracting the constant λ from the diagonal of X^TX gives us the liberty to always choose λ such that Ax is nonzero.

Problem #3

In logistic regression, the objective function can be written as:

$$E(w) = \frac{1}{N}\sum_{n=1}^{N} \ln(1 + e^{-y_n w^T \boldsymbol{x}_n})$$

(a) (10 points) Compute the first-order derivative $\nabla E(w)$. You will need to provide the intermediate steps of derivation.
Using the chain rule, we first note that $\frac{d(\ln \boldsymbol{x})}{d\boldsymbol{x}} = \frac{1}{\boldsymbol{x}}$. Thus, we automatically have part of the derivative as

$$\frac{1}{N}\sum_{n=1}^{N} \frac{1}{1 + e^{-y_n w^T x_n}}$$

Next, we differentiate the argument of the natural logarithm.

$$\frac{d(1 + e^{-y_n w^T x_n})}{dw} = -y_n \boldsymbol{x}_n e^{-y_n w^T x_n}$$

This is easy to derive. If we write $w^T\boldsymbol{x}$ as $w_1x_1 + \cdots + w_nx_n$, then it becomes obvious why $\partial(w^T\boldsymbol{x})/\partial w = \boldsymbol{x}$. We are left with:

$$\nabla E(w) = -\frac{1}{N}\sum_{n=1}^{N} \frac{y_n \boldsymbol{x}_n e^{-y_n w^T \boldsymbol{x}_n}}{1 + e^{-y_n w^T \boldsymbol{x}_n}}$$

Now, we'd like to make the exponential in the denominator positive, so we multiply by a convenient version of the number one.

$$\nabla E(w) = -\frac{1}{N}\sum_{n=1}^{N} \frac{y_n \boldsymbol{x}_n e^{-y_n w^T \boldsymbol{x}_n}}{1+e^{-y_n w^T \boldsymbol{x}_n}} \left(\frac{1+e^{y_n w^T \boldsymbol{x}_n}/1+e^{-y_n w^T \boldsymbol{x}_n}}{1+e^{y_n w^T \boldsymbol{x}_n}/1+e^{-y_n w^T \boldsymbol{x}_n}}\right)$$

$$\nabla E(w) = -\frac{1}{N}\sum_{n=1}^{N} \frac{y_n \boldsymbol{x}_n e^{-y_n w^T \boldsymbol{x}_n}}{1+e^{y_n w^T \boldsymbol{x}_n}} \left(\frac{1+e^{y_n w^T \boldsymbol{x}_n}}{1+e^{-y_n w^T \boldsymbol{x}_n}}\right)$$

$$\nabla E(w) = -\frac{1}{N}\sum_{n=1}^{N} \frac{y_n \boldsymbol{x}_n e^{-y_n w^T \boldsymbol{x}_n} + y_n \boldsymbol{x}_n e^{0}}{1+e^{y_n w^T \boldsymbol{x}_n}} \left(\frac{1}{1+e^{-y_n w^T \boldsymbol{x}_n}}\right)$$

$$\nabla E(w) = -\frac{1}{N}\sum_{n=1}^{N} \frac{y_n \boldsymbol{x}_n (e^{-y_n w^T \boldsymbol{x}_n} + 1)}{1+e^{y_n w^T \boldsymbol{x}_n}} \left(\frac{1}{1+e^{-y_n w^T \boldsymbol{x}_n}}\right)$$

$$\nabla E(w) = -\frac{1}{N}\sum_{n=1}^{N} \frac{y_n \boldsymbol{x}_n}{1+e^{y_n w^T \boldsymbol{x}_n}}$$

This is the result obtained in the book, so we are done.
(b) (10 points) Once the optimal w is obtained, it will be used to make predictions as follows:

$$\text{Predicted class of x} = \begin{cases} 1 & \text{if } \theta(w^T x) \geq 0.5 \\ -1 & \text{if } \theta(w^T x) < 0.5 \end{cases}$$

where the function $\theta(z) = \frac{1}{1+e^{-z}}$ looks like

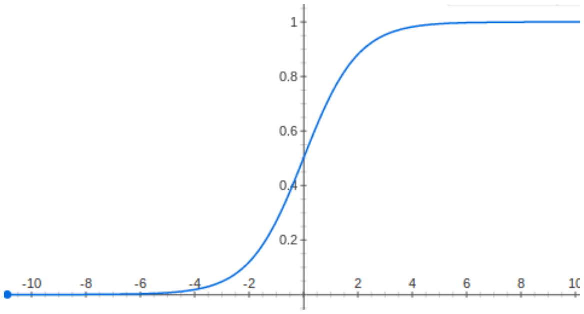


Figure 1: A basic sigmoid curve

Explain why the decision boundary of logistic regression is still linear, though the linear signal $w^T x$ is passed through a nonlinear function θ to compute the outcome of prediction. Is the decision boundary still linear if the prediction rule is changed to the following? Justify briefly.

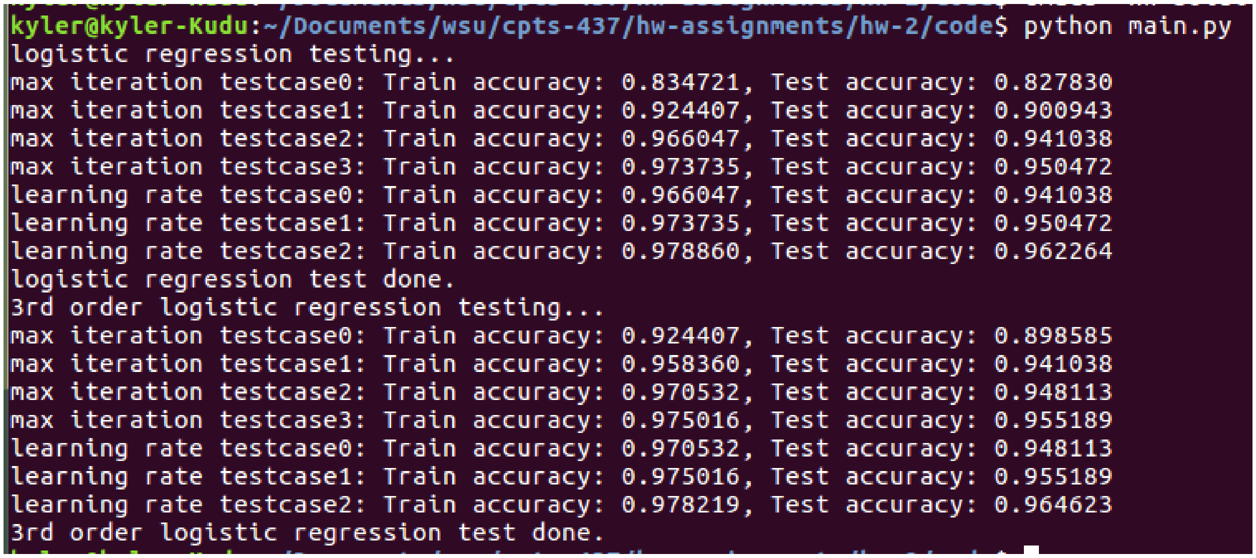
$$\text{Predicted class of x} = \begin{cases} 1 & \text{if } \theta(w^T x) \geq 0.9 \\ -1 & \text{if } \theta(w^T x) < 0.9 \end{cases}$$

The reason why the decision boundary of logistic regression is still linear is because the nonlinear function that processes the linear signal $w^T x$ is a monotonically increasing function. In other words, the output can be linearly separated because monotonically increasing/decreasing functions will never return to a value in their codomains after they have left it. Thus, even though the linear signal is passed through a nonlinear function θ, the output of this nonlinear function is still linearly separable.
The new decision boundary listed above is still linear because we are merely translating the boundary (which is already linear).

In light of your answers to the above two questions, what is the essential property of logistic regression that results in the linear decision boundary?
The functional transformation must be monotonically increasing or decreasing. That is what allows logistic regression to retain a linear decision boundary.

Problem #4

As your final deliverable to a customer, would you use the linear model with or without the 3rd order polynomial transform? Briefly explain your reasoning.
Based on my results, I would use the linear model without the 3rd order polynomial transform. The results I obtained with the 3rd order polynomial transform were not significantly better than those without the transform, as you can see in the figure below. Although the 3rd order polynomial transform seems to do better than the unmodified linear model when the maximum iterations are low, they achieve roughly equivalent results when the maximum number of iterations is higher. Furthermore, they achieve essentially the same accuracy ratings for the same learning rates. In my opinion, the very marginal improvement in the model



```
kyler@kyler-Kudu:~/Documents/wsu/cpts-437/hw-assignments/hw-2/code$ python main.py
logistic regression testing...
max iteration testcase0: Train accuracy: 0.834721, Test accuracy: 0.827830
max iteration testcase1: Train accuracy: 0.924407, Test accuracy: 0.900943
max iteration testcase2: Train accuracy: 0.966047, Test accuracy: 0.941038
max iteration testcase3: Train accuracy: 0.973735, Test accuracy: 0.950472
learning rate testcase0: Train accuracy: 0.966047, Test accuracy: 0.941038
learning rate testcase1: Train accuracy: 0.973735, Test accuracy: 0.950472
learning rate testcase2: Train accuracy: 0.978860, Test accuracy: 0.962264
logistic regression test done.
3rd order logistic regression testing...
max iteration testcase0: Train accuracy: 0.924407, Test accuracy: 0.898585
max iteration testcase1: Train accuracy: 0.958360, Test accuracy: 0.941038
max iteration testcase2: Train accuracy: 0.970532, Test accuracy: 0.948113
max iteration testcase3: Train accuracy: 0.975016, Test accuracy: 0.955189
learning rate testcase0: Train accuracy: 0.970532, Test accuracy: 0.948113
learning rate testcase1: Train accuracy: 0.975016, Test accuracy: 0.955189
learning rate testcase2: Train accuracy: 0.978219, Test accuracy: 0.964623
3rd order logistic regression test done.
```

Figure 2: Logistic Regression Console Output

is not worth it due to the sacrifice in performance. The 3rd order polynomial transform adds much unnecessary overhead, and it also causes poor generalization.