cs577 Assignment 2: Solution

Yuanxing Cheng, A20453410, CS577-f22
Department of Mathematics
Illinois Institute of Technology

October 5, 2022

Theoretical questions

Artificial neurons

1

Given a neuron with linear activation and weights $(0.1, 0.2., 0.2)$, compute the output for the input vector $1, 1$

Suppose the linear activation function be f, then the output would be

$$f(\theta^\top X + \theta_0) = f(0.1 + 0.2 \times 1 + 0.3 \times 1) = f(0.6)$$

2

Given a linear discriminant function $g(x)$ explain what should be its value on one side of the decision boundary, on the other side of the decision boundary, and on the decision boundary.

Since we only have one linear discriminant function, we assume we are doing a two class classification, then we say if $g(x) > 0$, then x is of class 1, and class 2 if $g(x) \leq 0$.

3

Given a linear discriminant function $g(x_1, x_2) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$ explain the meaning of the coefficients θs.

$\theta_{1,2}$ can be intepreted as a template that is used for the measuring of similarity between itself and xs. θ_0 is a bias term in this case, and can be used as the threshould for the strength of the similarity.

4

Let $g(x_1, x_2) = 1+2x_2+3x_3$. Find the normal to the decision boundary defined by g and the distance of the decision boundary from the origin.

Suppose the perpendicular point from the origin to the boundary from g is (x, y, z), then the vector (x, y, z) must be orthongonal to $0, 1/2, 1$ annd $1/3, 0, 1$. Thus,

$$\begin{cases} y/2 + z = 0 \\ x/3 + z = 0 \\ 1 + 2x + 3y = z \end{cases}$$

And this gives the point to be $(-3, -2, 1)$. And the distance is $\sqrt{9+4+1} = 4$. So the normal vector from the origin to the decision boundary would be $(-3/4, -1/2, 1/4)$.

5

When writing a discriminant function as $g(x) = \theta^\top x$ where x and θ are vectors, explain where the bias coefficient in θ is and what is the required change needed to the feature vector x to allow writing in this way.

In the common way we let θ_0 be the bias thus is the first element in θ, and accordingly, we add extra feature to the first column of X full of 1.

6

Write the expressions for the step and logistic activation functions. Explain the advantage of the sigmoid activation over step activation. What happens to the sigmoid when θ is small

Step activation function is $h_\theta(x) = \begin{cases} 1, & \theta^\top x > 0 \\ 0, & x \leq 0 \end{cases}$ and the sigmoid activation function is $h_\theta(x) = \frac{1}{1+\exp(-\theta^\top x)}$.

The advantage of sigmoid is that it's smooth and no information lost from input. When θ is small, the plot of sigmoid activation will be very close to the plot of step activation.

7

Explain how the sigmoid is obtained using a linear function to model the log-likelihood ratio.

The linear function is $\theta^\top x$ then we let it equals to the log-likelihood ratio obtaining the following:

$$\log\left(\frac{P(y=1\,|\,x)}{P(y=0\,|\,x)}\right) = \theta^\top x$$

Taking expoential on both sides and use the fact that $P(y=0\,|\,x) + P(y=1\,|\,x) = 1$ we obtain $P(y=1\,|\,x)\left(1+\exp(\theta^\top x)\right) = \exp(\theta^\top x)$ and thus $P(y=1\,|\,x) = h_\theta(x)$

8

Write the derivative of a sigmoid and use it to compute the derivative of log-sigmoid.

$$
\begin{aligned}
h_\theta(x) &= \frac{1}{1+\exp(-\theta^\top x)} \\
\frac{\mathrm{d}}{\mathrm{d}\theta} h_\theta(x) &= h_\theta(x)(1-h_\theta(x)) \\
\frac{\mathrm{d}}{\mathrm{d}\theta} \log h_\theta(x) &= (1-h_\theta(x))x
\end{aligned}
$$

9

Explain how to compute the direction used for updating parameters in gradient descent. How is the size of the update controlled?

The problem is to find $\theta^* = \arg\min_\theta J(\theta)$. Gradient descent is to update θ according to $\theta^{(i+1)} \leftarrow \theta^{(i)} - \eta \nabla J(\theta^{(i)})$. η can be used control the update size. With bigger η, the update is fast, but loss some accuracy.

10

Explain the stop condition for gradient descent. Should the condition use the loss change or parameter change? Explain your answer.

Stop condition: $\left|J(\theta^{(i+1)}) - J(\theta^{(i)})\right| < \epsilon$. And this is using the loss change. We don't use parameter change because parameter may change very slow when the loss hasn't reach minimum and thus prevent earlier exit.

11

Explain what happens when using a learning rate that is too large or too small.

Too large learning rate might cause gradient descent fail to converge and too small learning rate might cause gradient descent wasting times.

12

Explain how the empirical error loss is computed. What is the problem in using the empirical error loss in gradient descent.

Empirical error loss is computed by counting the incorrect predictions. Thus is always a step function. Thus the gradient will always be 0. If using gradient descent, the parameters cannot be updated.

13

Write the log-likelihood for a binary classifier and show how to develop it into binary cross-entropy.

Assuming total m samples, then the likelihood for binary classifier is

$$L(\theta) = \prod_{i=1}^{m} P(y = 1 \,|\, x^{(i)})^{y^{(i)}} P(y = 0 \,|\, x^{(i)})^{1-y^{(i)}}$$

The binary cross-entropy uses the negative log-likelihood and is written as follows

$$l(\theta) = -\log l(\theta) = -\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)}))$$

14

Write the gradient of the binary cross-entropy loss function with respect to the parameter vector when using a sigmoid activation function. Write the gradient descent update rule when using this loss function. Explain the meaning of the update equation you wrote.

Following last several questions, we obtain

$$\begin{aligned}\frac{\mathrm{d}}{\mathrm{d}\theta} l(\theta) &= -\frac{\mathrm{d}}{\mathrm{d}\theta} \sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) \\ &= -\sum_{i=1}^{m} y^{(i)} (1 - h_\theta(x^{(i)})) x^{(i)} + (1 - y^{(i)})(-h_\theta(x^{(i)})) x^{(i)} \\ &= -\sum_{i=1}^{m} (y^{(i)} - h_\theta(x^{(i)})) x^{(i)}\end{aligned}$$

Thus the update rule would be

$$\theta^{(i+1)} \leftarrow \theta^{(i)} + \eta \sum_{i=1}^{m} (y^{(i)} - h_\theta(x^{(i)})) x^{(i)}$$

Explain: using $\theta^{(i)}$ we can calculate $h_\theta(x^{(i)})$ and then with the given true label, we can calculate $(y^{(i)} - h_\theta(x^{(i)})) x^{(i)}$. Sum them up and times a learning rate, we get the update size to next θ.

15

Explain the two strategies for multi-class classification. In which strategy is it easier to discriminate the data.

One against all others strategy means each discriminant function gives the exact answer for one of the classes. And thus need at least k functions for k-class classification. One against each other strategy meaning that for any two different classes, there's a discriminant function. and thus need more discriminant functions generally. For k-class classification it requires at least $k(k+1)/2$ number of functions to discriminate. And with these extra discriminant functions, this strategy is easier.

16

Explain the template matching interpretation of linear discriminats. Using this interpretation what is the meaning of using multiple linear discriminant functions.

When parameters are treated as template, the linear discriminant function measures the similarity of the data, and the parameters. Higher values means great match.
Using multiple linear discriminant functions is when we want data are overall close to the different parameters.

17

Explain the need for using softmax in the output layer of a multi-class classifier.

We want the sum of output values equal to 1 so that it can be intepreted as probability.

18

Write the derivative of softmax and use it to compute the derivative of log-softmax.

Let the softmax function be $s(z_i) = \frac{\exp(z_i)}{\sum_i \exp(z_i)}$. Then the derivative is

$$\frac{\mathrm{d}}{\mathrm{d}z_j} s(z_i) = s(z_i)(\delta_{ij} - s(z_j))$$

$$\frac{\mathrm{d}}{\mathrm{d}\theta_j} s(\theta_i^\top x) = s(\theta_i^\top x)(\delta_{ij} - s(\theta_j^\top x))x$$

And the log-softmax has derivative

$$\frac{\mathrm{d}}{\mathrm{d}\theta_j} \log s(\theta_i^\top x) = (\delta_{ij} - s(\theta_j^\top x))x$$

19

Write the log-likelihood for a multi-class classifier and show how to develop it into categorical cross-entropy.

We use negative log-likelihood.

$$\begin{aligned} l(\theta) &= -\log\left(\prod_{i=1}^{m}\prod_{j=1}^{k} P(y^{(i)} = j \mid X^{(i)})^{\mathbb{1}_{y^{(i)}=j}}\right) \\ &= -\sum_{i=1}^{m}\sum_{j=1}^{k} \mathbb{1}_{y^{(i)}=j} \log P(y^{(i)} = j \mid X^{(i)}) \\ &= -\sum_{i=1}^{m}\sum_{j=1}^{k} \mathbb{1}_{y^{(i)}=j} \log h_{\theta_j}(x^{(i)}) \end{aligned}$$

20

Write the gradient of the categorical cross-entropy loss function wrt the parameter vector when using a softmax activation function. Write the gradient descent update rule when using this loss function. Explain the meaning of the update equation you wrote.

$$\frac{\mathrm{d}}{\mathrm{d}\theta_j} l(\theta) = \sum_{i=1}^{m}\left(h_{\theta_j}(x^{(i)}) - \mathbb{1}_{y^{(i)}=j}\right)x^{(i)}$$

And the update rule will be

$$\theta_j^{(k+1)} \leftarrow \theta_j^{(k)} - \eta \sum_{i=1}^{m} \left(h_{\theta_j}(x^{(i)}) - \mathbb{1}_{y^{(i)}=j} \right) x^{(i)}$$

Explain: using $\theta^{(i)}$ we can calculate $h_\theta(x^{(i)})$ and then with the given true label, we can calculate $\left(h_{\theta_j}(x^{(i)}) - \mathbb{1}_{y^{(i)}=j} \right) x^{(i)}$. Sum them up and times a learning rate, we get the update size to next θ.

Neural networks

1

Given a two layer network where the number of hidden units is larger than the number of inputs, explain the dimensionality increase interpretation of the hidden layer. Explain what is the possible benefit of such dimensionality increase.

Suppose input is of n dimension and we have $h > n$ hidden units. Then $Z = WX$ and z is of h dimension. Possible benefit would be easier to find a correct discriminant function like SVM.

2

Given a two layer network where the number of hidden units is smaller than the number of inputs, explain the dimensionality decrease interpretation of the hidden layer. Explain what is the possible benefit of such dimensionality decrease.

The explain is similar. The possible benefit is to obtain a simpler and thus better neural network structure while maintaining the accuracy.

3

Given the update equations for the output layer in a 2 layer network, explain why we cannot use directly the same equations for the hidden layer. Assume that both layers have the same activation function.

According to the chain rule, we need to multiply another derivative for the hidden layer update.

4

Apply the chain rule to compute the gradients with respect to both hidden layer and output layer parameters of a 2 layer regression network with a single output where the hidden layer use sigmoid activation and the output layer uses linear activation.

We have the following.

$$\begin{aligned} \hat{y} &= V^\top Z = v_0 + v_1 z_1 + \cdots v_h z_h \\ z_i &= s(w_i^\top x) \\ l(\theta) &= \frac{1}{2} \sum_i \left(\hat{y}^{(i)} - y^{(i)} \right)^2 \end{aligned}$$

$$
\begin{aligned}
\frac{\partial l}{\partial v} &= \frac{\partial l}{\partial \hat{y}}\frac{\partial \hat{y}}{\partial v} = \frac{1}{2}2\sum_i \left(\hat{y}^{(i)} - y^{(i)}\right) z^{(i)} \\
&= \sum_i \left(\hat{y}^{(i)} - y^{(i)}\right) z^{(i)} \\
V^{k+1} &\leftarrow V^k - \eta_1 \sum_i \left(\hat{y}^{(i)} - y^{(i)}\right) z^{(i)} \\
\frac{\partial l}{\partial w_i} &= \frac{\partial l}{\partial \hat{y}}\frac{\partial \hat{y}}{\partial z_i}\frac{\partial z_i}{\partial w_i} \\
&= \sum_i \left(\hat{y}^{(i)} - y^{(i)}\right) v_i z_i^{(i)}(1 - z_i^{(i)})x^{(i)} \\
w_j^{k+1} &\leftarrow w_j^k - \eta_2 \sum_i \left(\hat{y}^{(i)} - y^{(i)}\right) v_i z_i^{(i)}(1 - z_i^{(i)})x^{(i)}
\end{aligned}
$$

5

Apply the chain rule to compute the gradients with respect to both hidden layer and output layer parameters of a 2 layer regression network with multiple output where the hidden layer use sigmoid activation and the output layer uses linear activation.

We have the following. Supposing output dimension is m,

$$
\begin{aligned}
\hat{y}_i &= V_i^\top Z \\
z_i &= s(w_i^\top x) \\
l(\theta) &= \frac{1}{2}\sum_i \left|\hat{y}^{(i)} - y^{(i)}\right|^2 \\
\frac{\partial l}{\partial v_j} &= \frac{\partial l}{\partial \hat{y}}\frac{\partial \hat{y}}{\partial v_j} = \frac{1}{2}2\sum_i \left(\hat{y}^{(i)} - y^{(i)}\right)^\top z^{(i)} = \sum_i \left(\hat{y}^{(i)} - y^{(i)}\right)^\top z^{(i)} \\
V^{k+1} &\leftarrow V^k - \eta_1 \sum_i \left(\hat{y}^{(i)} - y^{(i)}\right)^\top z^{(i)} \\
\frac{\partial l}{\partial w_i} &= \frac{\partial l}{\partial \hat{y}}\frac{\partial \hat{y}}{\partial z_i}\frac{\partial z_i}{\partial w_i} \\
&= \sum_i \left(\hat{y}^{(i)} - y^{(i)}\right)^\top v_i z_i^{(i)}(1 - z_i^{(i)})x^{(i)} \\
w_j^{k+1} &\leftarrow w_j^k - \eta_2 \sum_i \left(\hat{y}^{(i)} - y^{(i)}\right)^\top v_i z_i^{(i)}(1 - z_i^{(i)})x^{(i)}
\end{aligned}
$$

6

Apply the chain rule to compute the gradients with respect to both hidden layer and output layer parameters of a 2 layer binary classification network where the hidden layer use sigmoid activation and the output layer uses sigmoid activation.

We have the following

$$
\begin{aligned}
\hat{y}_i &= s(V_i^\top Z) \\
z_i &= s(w_i^\top x) \\
l(\theta) &= -\sum_{i=1}^{m} y^{(i)} \log h_\theta(x^{(i)}) + (1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) \\
\frac{\partial l}{\partial v_j} &= \frac{\partial l}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial v_j} = -\sum_i \left(\frac{y^{(i)}}{\hat{y}^{(i)}} - \frac{1 - y^{(i)}}{1 - \hat{y}^{(i)}} \right) \hat{y}^{(i)} \left(1 - \hat{y}_j^{(i)}\right) z^{(i)} \\
&= -\sum_i^{m} \left(\frac{y^{(i)}}{\hat{y}^{(i)}} - \frac{1 - y^{(i)}}{1 - \hat{y}^{(i)}} \right) \hat{y}^{(i)} \left(1 - \hat{y}_j^{(i)}\right) z^{(i)} \\
V^{k+1} &\leftarrow V^k - \eta_1 \sum_i^{m} \left(\frac{y^{(i)}}{\hat{y}^{(i)}} - \frac{1 - y^{(i)}}{1 - \hat{y}^{(i)}} \right) \hat{y}^{(i)} \left(1 - \hat{y}_j^{(i)}\right) z^{(i)} \\
\frac{\partial l}{\partial w_i} &= \frac{\partial l}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_i} \frac{\partial z_i}{\partial w_i} \\
&= -\sum_i \left(\frac{y^{(i)}}{\hat{y}^{(i)}} - \frac{1 - y^{(i)}}{1 - \hat{y}^{(i)}} \right) \hat{y}^{(i)} \left(1 - \hat{y}_j^{(i)}\right) z_i^{(i)} z_i^{(i)} (1 - z_i^{(i)}) x^{(i)} \\
w_j^{k+1} &\leftarrow w_j^k - \eta_2 \sum_i \left(\frac{y^{(i)}}{\hat{y}^{(i)}} - \frac{1 - y^{(i)}}{1 - \hat{y}^{(i)}} \right) \hat{y}^{(i)} \left(1 - \hat{y}_j^{(i)}\right) \left(z_i^{(i)}\right)^2 (1 - z_i^{(i)}) x^{(i)}
\end{aligned}
$$

7

Apply the chain rule to compute the gradients with respect to both hidden layer and output layer parameters of a 2 layer multi-class classification network where the hidden layer use sigmoid activation and the output layer uses softmax activation.

We have the following

$$
\begin{aligned}
\hat{y}_i &= S(V_i^\top Z) \\
z_i &= s(w_i^\top x) \\
l(\theta) &= -\sum_{i=1}^{m} \sum_{j=1}^{k} \mathbb{1}_{y^{(i)}=j} \log h_{\theta_j}(x^{(i)}) \\
\frac{\partial l}{\partial v_j} &= \frac{\partial l}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial v_j} = -\sum_{i=1}^{m} \sum_{j=1}^{k} \frac{\mathbb{1}_{y^{(i)}=j}}{\hat{y}^{(i)}} \hat{y}^{(i)} (\delta_{ij} - \hat{y}^{(j)}) z^{(i)} \\
V^{k+1} &\leftarrow V^k - \eta_1 \sum_{i=1}^{m} \sum_{j=1}^{k} \frac{\mathbb{1}_{y^{(i)}=j}}{\hat{y}^{(i)}} \hat{y}^{(i)} (\delta_{ij} - \hat{y}^{(j)}) z^{(i)}
\end{aligned}
$$

$$
\begin{aligned}
\frac{\partial l}{\partial w_i} &= \frac{\partial l}{\partial \hat{y}} \frac{\partial \hat{y}}{\partial z_i} \frac{\partial z_i}{\partial w_i} \\
&= -\sum_{i=1}^{m} \sum_{j=1}^{k} \frac{\mathbb{1}_{y^{(i)}=j}}{\hat{y}^{(i)}} \hat{y}^{(i)} (\delta_{ij} - \hat{y}^{(j)}) z_i^{(i)} z_i^{(i)} (1 - z_i^{(i)}) x^{(i)} \\
w_j^{k+1} &\leftarrow w_j^k - \eta_2 \sum_{i=1}^{m} \sum_{j=1}^{k} \frac{\mathbb{1}_{y^{(i)}=j}}{\hat{y}^{(i)}} \hat{y}^{(i)} (\delta_{ij} - \hat{y}^{(j)}) z_i^{(i)} z_i^{(i)} (1 - z_i^{(i)}) x^{(i)}
\end{aligned}
$$

8

Describe how weights in the network should be intialized and explain why they are initialized in this way.

Use random normal distributed samples, for simplicity. This way we can avoid too big weights or too small weights, which have negative impact on the convergence of gradient descent.

Computation graphs

1

Explain the advantage of using computation graphs. Describe what is computed during the forward pass and backward pass of the network during backpropagation. Explain what each node in the graph needs to be able to compute and what information it needs to store, and why.

Advantage is to avoid making mistakes in using chain rule. In forward pass, nodes ouput are computed, from the first to last layer. In backward pass during backpropagation, derivative of loss wrt parameters are computed, and the derivative is stored so that we can have the overall product in the chain rule.

2

Given a two layer network where the hidden layer output is given by $z = f_1(W_1, x)$ and the output of output layer is $\hat{y} = f_2(W_2, z)$. Express the output $\hat{y}$ as a function composition. Assuming the output layer has a single node, express the L_2 loss for a batch $\left\{x^{(i)}, y^{(i)}\right\}_{i=1}^{m}$ in terms of the composite function you wrote and use the chain rule to write its derivatives wrt W_i. Repeat the question but using cross-entropy loss.

Output is $\hat{y} = f_2(W_2, f_1(W_1, x))$. L_2 loss for a batch of data is $l = \sum_i^m \frac{1}{2}(\hat{y}^{(i)} - y^{(i)})^2 = \sum_i^m \frac{1}{2}(f_2(W_2, f_1(W_1, x^{(i)})) - y^{(i)})^2$