Homework 2

Felipe Toledo
ft8bn

March 16, 2021

Clarifications

- I discussed the homework with Dane Williamson (dw3zn).

Problem 1

Even though the softmax function is a nonlinear function, it can still be considered as a linear classifier. A linear classifier is one where a hyperplane is formed by taking a linear combination of the features, such that one side of the hyperplane predicts one class and the other side predicts the other. For multi-class classification we use the softmax function which calculates the probability of an input of being in a particular output class.

We can convert this multi-class classification problem into a binary classification problem if we separate one class from the rest of the classes, also known as "one versus all classification". In general, to solve a multi-class classification problem with n classes, n logistic regression models are trained each considering only a single class. Whenever a new data point has to be classified, the n models predict the probability of the point falling into its respective class. Finally, the point is assigned to the class with the highest probability as predicted by one of the n models.

In that case we can define two probabilities: $P(y|x)$ and $P(\neg y|x)$. For binary classification, we know that those probabilities are:

$$P(y|x) = \frac{1}{1 + exp(\boldsymbol{\omega}^T \boldsymbol{x} + b)} \tag{1}$$

$$P(\neg y|x) = 1 - P(y|x) = \frac{exp(\boldsymbol{\omega}_1^T \boldsymbol{x} + b)}{1 + exp(\boldsymbol{\omega}^T \boldsymbol{x} + b)} \tag{2}$$

The output would be of class "y" if the following is satisfied:

$$P(y|x) > P(\neg y|x)$$

which is equivalent to:

$$\frac{P(y|x)}{P(\neg y|x)} > 1$$

If we apply the logarithm to both sides, we will get:

$$\log(\frac{P(y|x)}{P(\neg y|x)}) > \log(1)$$

$$\log(\frac{P(y|x)}{P(\neg y|x)}) > 0 \tag{3}$$

If we replace (1) and (2) intro (3) we will get:

$$\log(exp(\boldsymbol{\omega}_1^T \boldsymbol{x} + b)) - \cancel{log(1 + exp(\boldsymbol{\omega}^T \boldsymbol{x} + b))} - log(1) + \cancel{log(1 + exp(\boldsymbol{\omega}^T \boldsymbol{x} + b))} > 0$$

$$\log(exp(\boldsymbol{\omega}_1^T \boldsymbol{x} + b)) - \emptyset > 0$$

$$\boldsymbol{\omega}_1^T \boldsymbol{x} + b > 0 \tag{4}$$

From (4) we see that the decision boundary is given by the plane $\boldsymbol{\omega}_1^T\boldsymbol{x} + b$, consequently we can conclude that the softmax function is still a linear classifier.

Problem 2

Lets start from the definition of the softmax function:

$$P(y|x) = \frac{exp(\boldsymbol{\omega}_y^T\boldsymbol{x})}{\sum_{y' \in Y} exp(\boldsymbol{\omega}_{y'}^T\boldsymbol{x})}$$

Now lets find out what is the $P(y = 1|x)$

$$\begin{aligned}
P(y = 1|x) &= \frac{exp(\boldsymbol{\omega}_1^T\boldsymbol{x})}{exp(\boldsymbol{\omega}_0^T\boldsymbol{x}) + exp(\boldsymbol{\omega}_1^T\boldsymbol{x})} \\
&= \frac{exp(\boldsymbol{\omega}_1^T\boldsymbol{x})}{exp(\boldsymbol{\omega}_0^T\boldsymbol{x}) + exp(\boldsymbol{\omega}_1^T\boldsymbol{x})} \cdot \frac{exp(-\boldsymbol{\omega}_1^T\boldsymbol{x})}{exp(-\boldsymbol{\omega}_1^T\boldsymbol{x})} \\
&= \frac{exp(\boldsymbol{\omega}_1^T\boldsymbol{x} - \boldsymbol{\omega}_1^T\boldsymbol{x})}{exp(\boldsymbol{\omega}_0^T\boldsymbol{x} - \boldsymbol{\omega}_1^T\boldsymbol{x}) + exp(\boldsymbol{\omega}_1^T\boldsymbol{x} - \boldsymbol{\omega}_1^T\boldsymbol{x})} \\
&= \frac{exp(0)}{exp((\boldsymbol{\omega}_0^T - \boldsymbol{\omega}_1^T)\boldsymbol{x}) + exp(0)} \\
&= \frac{1}{exp((\boldsymbol{\omega}_0^T - \boldsymbol{\omega}_1^T)\boldsymbol{x}) + 1} \qquad (1)
\end{aligned}$$

Softmax function is used for multi-class classification problems, in the above equation we end up having $\boldsymbol{\omega}_0^T$ and $\boldsymbol{\omega}_1^T$ that represent the classification weight associated with the two possible labels. To simplify the equation, we can express the subtraction of the weights in the following way:

$$\begin{aligned}
\boldsymbol{\omega}^T &= \boldsymbol{\omega}_0^T - \boldsymbol{\omega}_1^T \\
-\boldsymbol{\omega}^T &= \boldsymbol{\omega}_1^T - \boldsymbol{\omega}_0^T \qquad (2)
\end{aligned}$$

If we replace (2) in (1), we get:

$$P(y = 1|x) = \frac{1}{exp(-\boldsymbol{\omega}_1^T\boldsymbol{x}) + 1}$$

Problem 3

Lets start by the definition of cross-entropy loss function:

$$H(Q(Y|x), P(Y|x)) = -\sum_{y' \in Y} Q(Y = y'|x) \log P(Y = y'|x) \qquad (3)$$

With the empirical distribution:

$$Q(Y = y'|x^{(i)}) = \begin{cases} 1 & y' = y \\ 0 & y' \neq y \end{cases}$$

Based on the empirical distribution, we can see in (1) that we are only going to sum the terms in which the predicted label $y^{'}$ is equal to the ground truth y. Moreover, if we use the collection of training examples denoted by $\{(x^{(i)}, y^{(i)})\}_{i=1}^{m}$ instead of the predicted labels $y^{'} \in Y$. We can replace $Q(Y = y^{'}|x^{(i)})$ in (1) by the ground truth probability of $y^{(i)}$ which will always be 1 according to the empirical distribution, and $P(Y = y^{'}|x)$ by the probability of the predicted classes $P(Y = y^{(i)}|x^{(i)})$. By doing that, we will need to modify the bounds of the summation to iterate over all the training data. Obtaining:

$$\sum_{i=1}^{m} H(Q(Y|x^{(i)}), P(Y|x^{(i)})) = -\sum_{i=1}^{m} y^{(i)} \log P(Y = y^{(i)}|x^{(i)})$$
$$\sum_{i=1}^{m} H(Q(Y|x^{(i)}), P(Y|x^{(i)})) = L(\theta)$$

Problem 4

The code for this problem is in a python notebook called "ft8bn-q4.ipynb". The results of the hypertunning of the parameters was saved in a file called "results.pkl", this file is loaded in the same notebook. As indicated in point (e) the test headlines predictions were saved in a file called "news-tst.pred".

Problem 5

The code for this problem is in a python notebook called "ft8bn-cnn.ipynb". The validation accuracy that I got from the cnn model is 0.6352.