# Machine Learning HW2 Report

電機三 郭笛萱

B03901009

Discussed with b03901009 施順耀 b03901016 陳昊

## Problem 1 : Logistic Regression Function

```
#training start#
for i in range (0, iteration, 1):    #run for n iteration
    likelihood = 0
    for j in range(len(train[0])):
        sig_train[j] = 0
    sig_b = 0

    for j in range(pos, pos + len(train) / batch): #run every data in each mini-batch
        f = 0.0
        for k in range(0, len(train[0])):
            f += w_train[k] * train[j][k]
        f += b
        f = (1 / (1.0 + math.exp(-f)))            #calculate f_w,b with sigmoid function
        for k in range(0, len(train[0])):
            sig_train[k] += -(spam_train[j] - f) * train[j][k]
        sig_b += -(spam_train[j] - f)
        #calculate loss function
        likelihood += -((spam_train[j] * math.log(f + epsilon))
                        + ((1.0-spam_train[j]) * math.log(1.0-f+epsilon)))
    #decide the starting point of each mini-batch
    if mini_batch == 1 :
        if pos != len(train) / batch * (batch - 1) :
            pos += len(train) / batch
        else : pos = 0
    for j in range(0, len(train[0])):                #update weight
        w_train[j] -= n_train[j] * sig_train[j]
    b -= n_b * sig_b
```

## Problem 2: Describe Method2

1. I use generative model to solve the problem. First calculate the sigma & mean values of class 1 & 2, and to make sigma1 & sigma2 be the same, I use

   $\sum = \frac{N1}{N1+N2}\sum 1 + \frac{N2}{N1+N2}\sum 2$ .

   Next, calculate w & b.

   $$\mathrm{w} = (\mu 1 - \mu 2)^T \sum^{-1}$$

   $$\mathrm{b} = -\frac{1}{2}(\mu 1)^T \sum^{-1} \mu 1 + \frac{1}{2}(\mu 2)^T \sum^{-1} \mu 2 + ln\frac{N1}{N2}$$

   finally, we can get $\mathrm{P}(\mathrm{C}_1|\mathrm{x}) = \sigma(w \cdot x + b)$

   If $\mathrm{P}(\mathrm{C}_1|\mathrm{x}) > 0.5 => ans = 1,\ else\ ans = 0$

2. The result of generative model on training data is only 0.85, on validation set is only 0.61, and 0.8600 on public score. Compare with discriminative model, the result on training data is 0.925, and on validation set is 0.920.

   I think the reason why generative model performs not so well is because this method uses too many estimation in it, the real model may not be Gaussion distribution and although using Naïve Bayes Classifier is simple, in the real model, different features may not be independent.

## Problem 3 : Descibe Method 1

1. Mini-batch: After trying different combination of batch size and number of iteration, batch size = 2, iteration = 300000 performs best on training set and validation set, however, overfitting occurs. This model got only 0.92 on public score, so I choose batch size = 5 and iteration = 300000, which got 0.93667 on public score, as the final model.

2. Choosing features: There are 57 features given in this homework, and I try to delete different feature each time and calculate the training score and validation score with one feature misses. It shows that deleting features 2, 18, 26, 54, 57 gives the best results on validation set. However, the public score doesn't give the best

result while ignoring these features. The best public score I get is when deleting the 54th & 57th features.

The reason for deleting the 57th feature is not so surprising, since the 57th feature is the total capital letters of the email, intuitively, the real email can have a long paragraph, so does spam email. So no matter how many capital letters there are, it doesn't give us the information to see whether it is a spam email or not.

3. Loss function

   Choosing cross entropy as loss function.

$$L(f) = \sum -\left[\left(y - \ln\left(f(x^n + \varepsilon)\right)\right) + (1 - y) * (1 - \ln\left(f(x^n + \varepsilon)\right))\right]$$

   Notice that when f approaches zero, ln(f) will be -∞, so we should add an epsilon in the equation, where epsilon = 1e-20 in my model.

   The best model I got had a loss function = 170.9

## Problem 4 : Other Discussion on Validation Set

I cut 1000 data from the training data to be my validation set, however, validation set sometimes cannot correctly reflect the trend of testing data. For example, the highest public score I got is 0.93667, and the validation score I got is 0.9270, however, if I try another model, I got 0.9300 on the validation score, but the public score is only 0.92333. I think next time, I should try to randomly produce my validation set, so maybe it can estimate the result on testing data better.