Homework 2 Report

系級:資工碩二 學號:R06922134 姓名:葉沛陽

Problem 1. (1%) 請簡單描述你實作之 logistic regression 以及 generative model 於此 task 的表現，並試著討論可能原因。

	public	private
Logistic regression	0.67580	
Generative model	0.21940	

此兩個 model 比較基準為資料未經過 one-hot encoding 且 normalization 等等錢處理，直接將原始值進行訓練的結果

由表格可看出 Logistic regression 比 generative model 有比較高的準確率，可能是因為 generative model 是在假設資料是在高斯分布的前提下去運算的，再加上資料未經過 one-hot encoding 且 normalization 所以資料的分佈可能跟高斯分布差異很大。而 logistic regression 則是直接使用 regression 方式去切割兩個 class，因此至少會有一定的準確率。

Problem 2. (1%) 請試著將 input feature 中的 gender, education, martial status 等改為 one-hot encoding 進行 training process，比較其模型準確率及其可能影響原因。

	public	private
Without one-hot	0.67580	
With one-hot	0.78060	

此表格為用同樣參數的 logistic regression model，除了有無使用 one-hot encoding，其他資料前處理都一致所得出的結果。

由表格可以看出有使用 one-hot encoding 會比沒有使用 one-hot encoding 來得準確，原因是因為原始資料將性別:男設定為 1、女設定為 2，教育…等等 feature 也是直接使用 1234 數字來表示類別，這會造成某些類別比較接近或是數值比較大，但也許跟他們事實上的關係並不符合，因此，使用 one-hot encoding 會讓每個類別各自獨立出來當成各個 feature，這樣會比較符合事實，也可得出比較高的準確率。

Problem 3. (1%) 請試著討論哪些 input features 的影響較大（實驗方法沒有特別限制，但請簡單闡述實驗方法）。

我採用 permutation test。簡單來說，要看某個 feature 影響力大不大，可以將那個 feature 整行隨機排列，然後觀察隨機排列前後對於結果的影響，取絕對值後，越大的表示這個 feature 影響結果越多越重要。

```
In [107]: feature_weight = []
          for i in range(valid_x_bias.shape[1]):
              c = np.copy(valid_x_bias)
              np.random.shuffle(c[:,i])
              feature_weight.append( abs((((sigmoid( np.dot(c, train_w) )>0.5) ==  valid_y).sum()/valid_y.shape[0]) -
                                         (((sigmoid( np.dot(valid_x_bias, train_w) )>0.5) ==  valid_y).sum()/valid_y.shape[0])) )

In [108]: feature_weight

Out[108]: [0.0,
           0.0232,
           0.017199999999999993,
           0.024400000000000033,
           0.019799999999999984,
           0.006400000000000017,
           0.028400000000000036,
           0.0,
           0.0,
           0.0014000000000000123,
           0.0,
           0.021600000000000008,
           0.01880000000000004,
           0.0014000000000000123,
           0.0,
           0.00019999999999997797,
           0.040600000000000025,
           0.019600000000000006,
           0.015199999999999991,
```

例如這樣
值為 0.0 表示有無隨機排列對於結果毫無影響，很不重要
而值越大表示那個 feature 對結果影響越大。

Problem 4. (1%) 請實作特徵標準化 (feature normalization)，並討論其對於模型準確率的影響與可能原因。

	public	private
Without normalization	0.78060	
With normalization	0.81880	

此表格為用同樣參數的 logistic regression model，除了有無使用 mean normalization，其他資料前處理都一致所得出的結果。
由表格可以看出有使用 normalization 會比沒有使用 normalization 來得準確，原因應該是原本的資料各個 feature 的上下限差異非常大，也許有些 feature 值就算差一點點就很重要，有些 feature 值差很多還是沒有什麼影響，若沒有先做 normalization 這會造成在用梯度下降訓練 weights 的時候，weights 會比較難 match 到較正確的位置，造成各 feature 的 weight 收斂速度不一，所以比較可能會得出較差的結果。

Problem 5. (1%) The Normal (or Gaussian) Distribution is a very common continuous probability distribution. Given the PDF of such distribution

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}, -\infty < x < \infty$$

please show that such integral over $(-\infty, \infty)$ is equal to 1.

ANS:

$$let \qquad I = \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx = \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(y-\mu)^2}{2\sigma^2}} dy$$

$$I^2 = \int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(y-\mu)^2}{2\sigma^2}} dxdy$$

$$= \frac{1}{\sqrt{2\pi}\sigma^2} \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} e^{-\frac{(x-\mu)^2+(y-\mu)^2}{2\sigma^2}} dxdy$$

$$let \qquad x-\mu = rcos\theta, \qquad y-\mu = rsin\theta$$

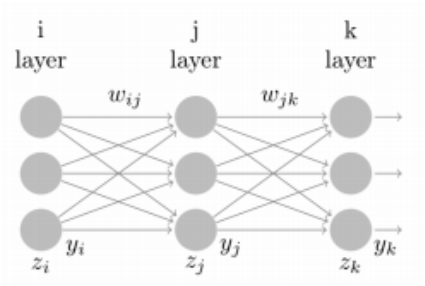
$$dxdy = rdrd\theta$$

$$I^2 = \frac{1}{2\pi\sigma^2} \int_0^{2\pi}\int_0^{\infty} e^{-\frac{r^2}{2\sigma^2}} rdrd\theta = \frac{1}{2\pi\sigma^2} 2\pi \int_0^{\infty} e^{-\frac{r^2}{2\sigma^2}} rdr$$

$$= \frac{1}{2\sigma^2} \int_0^{\infty} e^{-\frac{r^2}{2\sigma^2}} 2rdr = \frac{1}{2\sigma^2} \int_0^{\infty} e^{-\frac{r^2}{2\sigma^2}} dr^2$$

$$= \int_0^{\infty} e^{-r^2} dr^2 \quad = \quad 1$$

Problem 6. (1%) Given a three layers neural network, each layer labeled by its respective index variable. I.e. the letter of the index indicates which layer the symbol corresponds to.



For convenience, we may consider only one training example and ignore the bias term.Forward propagation of the input z_i is done as follows. Where $g(z)$ is some differentiable function (e.g. the logistic function).

$$
\begin{aligned}
y_i &= g(z_i) \\
z_j &= \sum_i w_{ij} y_i \\
y_j &= g(z_j) \\
z_k &= \sum_j w_{jk} y_j \\
y_k &= g(z_k)
\end{aligned}
$$

Derive the general expressions for the following partial derivatives of an error function E, also sime differentiable function, in the feed-forward neural network depicted. In other words, you should derive these partial derivatives into "computable derivative" (e.g. $\frac{\partial E}{\partial y_k}$ or $\frac{\partial z_k}{\partial w_{jk}}$).

$$(a)\frac{\partial E}{\partial z_k} \quad (b)\frac{\partial E}{\partial z_j} \quad (c)\frac{\partial E}{\partial w_{ij}}$$

ANS:

$$(a) \quad \frac{\partial E}{\partial z_k} = \frac{\partial E}{\partial y_k} \frac{\partial y_k}{\partial z_k}$$

$$(b) \quad \frac{\partial E}{\partial z_j} = \frac{\partial E}{\partial y_k} \frac{\partial y_k}{\partial z_k} \frac{\partial z_k}{\partial y_j} \frac{\partial y_j}{\partial z_j}$$

$$(c) \quad \frac{\partial E}{\partial w_{ij}} = \frac{\partial E}{\partial y_k} \frac{\partial y_k}{\partial z_k} \frac{\partial z_k}{\partial y_j} \frac{\partial y_j}{\partial z_j} \frac{\partial z_j}{\partial w_{ij}}$$