ML hw1 b02901014 王崇勳

1. 實作 linear regression

Code:

```
lr = 0.0000000003
for i in range(iteration):
    y = np.dot(xArray,weight)
    grd = 2*np.dot(np.transpose(xArray),(y-yHat)) + 2*lamda*weight
    loss = (np.dot(np.transpose(y-yHat),(y- yHat)) +
            lamda*np.dot(np.transpose(weight), weight)
    weight = weight - grd*lr
```

其中:

xArray 及取出來的 feature、grd 為計算出來的 gradient、lr 為 learning rate。詳細的作法會在後面的部分說明。

2. 方法簡介

feature 的部份我將每一個小時的 18 個因素都考慮進去。先將 training data 排成一個 18*5760 的陣列，用一個 18*9 的 window 掃過整個陣列，再將 window 掃到的資料轉成一個 1*162 的 row 方便進行運算。因爲每一個月的資料都只有 20 天，爲了解決 1/20 到 2/1 的斷層，因此當 window 掃到每一個月的 20 號的最後十個小時（要留最後一小時當做 Y），就會直接跨越到下一個月的 1 號，也就是說每一個月總共可以得到 471 個 row。按照這個步驟做，可以得到一個 5652*162 的陣列。另外，爲了運算方便，我將 b 合併到第 163 個 column，b 的初始值預設是 w。最後可以得到 X 是一個 5652*163 的陣列。

另一方面要處理的是矩陣運算的問題。根據上課的投影片可以知道 loss 與 gradient 的算法：

$$\sum_{n=1}^{10}\left(\hat{y}^n-\left(b+w\cdot x_{cp}^n\right)\right)^2$$

loss

$$\frac{\partial L}{\partial w}=?\sum_{n=1}^{10}2\left(\hat{y}^n-\left(b+w\cdot x_{cp}^n\right)\right)\left(-x_{cp}^n\right)$$

$$\frac{\partial L}{\partial b}=?\sum_{n=1}^{10}2\left(\hat{y}^n-\left(b+w\cdot x_{cp}^n\right)\right)$$

gradient

為了方便計算，我將 X、w、Y 直接帶入方程式中。其中遇到需要平方的部分，就將其中一個矩陣就由轉至矩陣取代，例如 X^2 就變成 X^TX。另外一方面因為已經將 b 合併到 w 裡面，因此最後可以得到：

$$Loss = (Y - X*w)^T(Y - X*w)$$

$$Gradient = 2*X^T(Y - X*w)$$

3. Regularization

從改變不同 λ 可以發現加上 regularization 以後影響不大，當 λ 大於 1 以後，甚至比沒有加上 regularization 時的結果差。(迭代 20 萬次，learning rate = 3*10^-3)

λ	0.01	0.1	0	1
Error (upload to kaggle)	5.66587	5.66588	5.66587	5.66589
λ	10	100	1000	10000
Error (upload to kaggle)	5.66598	5.66697	5.67692	5.78115

4. Learning rate

Learning rate 代表了 gradient descent 每次往 global minima 或 local minima 一次可以走的 step。從不同 learning rate 的比較結果可以發現大於 10^-9 數量級，loss 很容易跑到無限大的地方。當 learning rate 越來越小以後，會發現最終的 loss 會比較高，而且在 kaggo 上面測出的 error 也會比較高。不過從 Figure.1 和 Figure.2 可以看出來雖然 learning rate 不同，但是兩者一開始 loss 的下降速度是差不多的。(迭代 20 萬次，λ = 0)

Learning rate	10^-8	10^-9	10^-10	10^-11	10^-12
loss	nan	nan	33.24859887	41.36314205	71.26669305
Error (upload to kaggle)	nan	nan	5.66587	6.81598	9.96737

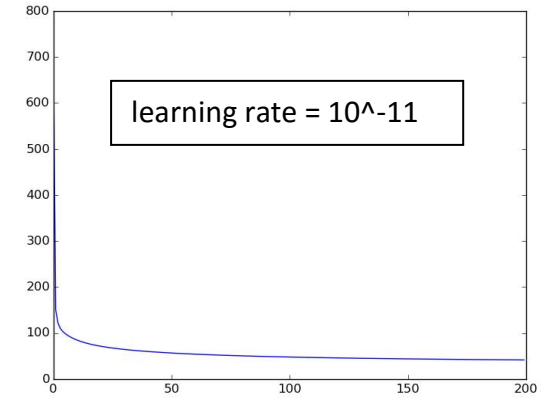

Figure.1

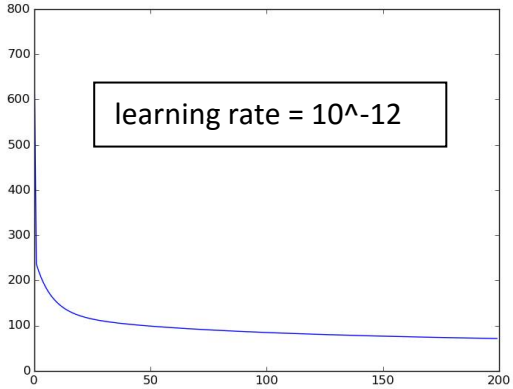

Figure.2

5. 本次作業最好的結果是迭代 50 萬次，learning rate = 3*10^3，lamda = 0，在 kaggle 上面得到的分數是 5.66461。