Machine Learning HW1　　電機四 黃馨平 b02901039

1.Linear regression function by Gradient Descent.

```
for(int k=0;k<time;k++)                          ←training 次數
    for(int n=0;n<datanum;n++)                   ←data 數目
        for(int m=0;m<parnum;m++)                ←W 和 b 的總數
            Y[n]=Y[n]+W[m]*Data[m][n];           ←計算估計的 Y 值
        for(int m=0;m<parnum;m++)                ←對每個 W
            D[m]=D[m]-2*(R[n]-Y[n])*Data[m][n];  ←算微分值
            G[m]=G[m]+D[m]*D[m];                 ←Adagrad 的 G
    for(int m=0;m<parnum;m++)                    ←對每個 W
        W[m]=W[m]-rate*D[m]/sqrt(G[m]);          ←更新 W
        D[m]=0;                                  ←將微分值歸 0
```

2.Describe your method.

$$y = b + \sum w_i x_i$$

$$L = \sum_n \left(\hat{y}^n - \left(b + \sum w_i x_i \right) \right)^2$$

將前九個小時中，每個小時的十八個參數當作 feature，將所有 NR 當成 0，總共有 9*18=162 個 feature，再加上 1 個 bias。

也就是用前九個小時的 162 個數字做線性組合後加上 bias 來估計第十個小時的 PM2.5。

3.Discussion on regularization.

可以發現當 λ 越大的時候 training set 的 RMSE 就越大，λ 為 0 的時候 RMSE 為最小，這是當然的，因為原本就是在 fit training set，如果加入 regularizer，自然 RMSE 就會變大。

而當 λ 在某個特定大小時，testing set 的 RMSE 為最小，λ 變大或變小都會讓 RMSE 變大。雖然看不出太大的變化，但是可以得知有一定大小的 λ，會使 testing set 的結果有進步，表示加入 regularizer 是有用的。

Regularizer 的主要功能是讓 W 的值變小，以避免 overfitting。覺得看不出太大變化的原因可能是因為參數的數目很多，所以原本每個 W 的值就已經很小了，加入 Regularizer 也只是讓他再變小一點點，所以看不出太大的差異。(還是說，其實本來就已經沒有 fit 的很好了。)

λ	Training set RMSE	Testing set RMSE
0.01	5.70596	5.72658
0.001	5.7059	5.72589

0.0001	5.70585	5.72583
0.00007	5.70585	5.72582
0.00004	5.70585	5.72582 (Kaggle 說進步 0.00000，應該有進步)
0.00001	5.7058	5.72582
0	5.7058	5.72582

可以發現當λ為 0.00001 左右的時候，public testing set 的 RMSE 有變小一些，推測設定 λ=0.0001~0.00001 之間，可以達到加入 regularizer 最好的效果。

4.Discussion on learning rate.

Learning rate 如果太大的話，training set 的 RMSE 會越來越大，train 出的參數以及微分值都會變很大，而且無法收斂到最低點。Learning rate 如果太小的話，training set 的 RMSE 會變小得越來越慢，train 出的參數以及微分值也會變化的很小，雖然可以收斂，但是要 train 很長很長的時間。

適當的 learning 可以讓 training 的圈數不用太多，就可以讓 RMSE 及微分值都變得很小，取得適當的參數。

Learning rate	Training set RMSE	附註
0.001	329.113	因為 training 速度太慢，所以在次數 500000 時，誤差還非常大，還要 train 更久才行
0.01	13.0646	
0.1	6.99508	
1	5.70522	
4	5.68857	約在 90000 次左右 5.70346
7	5.68597	約在 60000 次左右 5.70344
10	5.68616	約在 60000 次左右 5.7034

比較 training 次數=500000 左右的 RMSE，發現在 learning rate 越大的時候，就愈快可以達到 5.7 左右的 RMSE。Learning rate 很小的情況下，即便 train 500000 次，結果還是很差。

(因為實在 train 太久了，所以 code 中將已經 train 到 500000 次左右的參數當作初始值。)

5.Other discussion-adagrad

$$w^{t+1} \leftarrow w^t - \frac{\eta}{\sqrt{\sum_{i=0}^{t}(g^i)^2}} g^t$$

在沒有使用 adagrad 之前，RMSE、微分值降低的速度非常慢，幾乎是等速然後越來越慢，參數變化的速度也很慢，在使用 adagrad 之後，RMSE 降低的速度變快很多，一開始參數變化的速度以及微分值降低的速度可以很快很快，而當越接近目標的時候，參數變化的速度及微分值降低的速度就越慢，最後 RMSE 變很小。可以發現在有使用 adagrad 的情況下，當 learning=7 左右的時候，train 的速度比較快，較容易達到小的 RMSE。