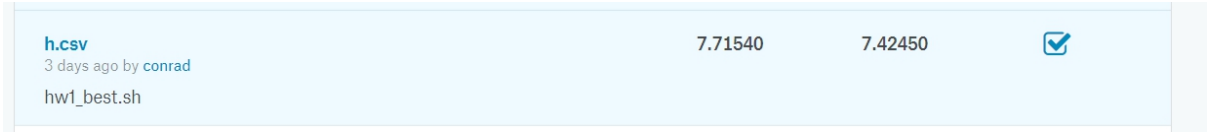
Homework 1 Report - PM2.5 Prediction

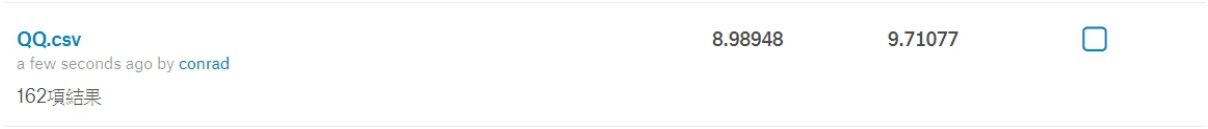
學號：R06942143 系級：電信丙組碩一 姓名：籃聖皓

1. (1%) 請分別使用每筆 data9 小時內所有 feature 的一次項（含 bias 項）以及每筆 data9 小時內 PM2.5 的一次項（含 bias 項）進行 training，比較並討論這兩種模型的 root mean-square error（根據 kaggle 上的 public/private score）。

只有一次項(private/public)



使用 162 項參數(private/public)



使用 162 項參數的結果會比較差的原因，應該有下面幾點：

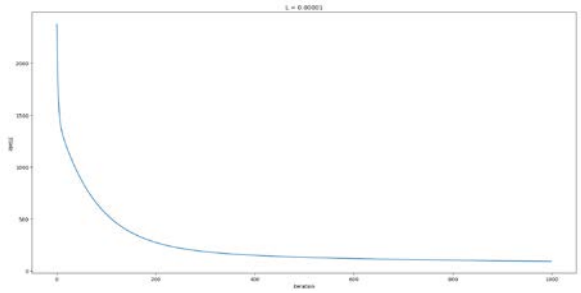
1. 太多的無效參數：
   使得模型會產生 overfitting，多了這些參數不會使模型變得更加的準確，反倒會使訓練時間變長，訓練結果變差。
2. 太多的 outlier：
   原本的 PM2.5 就有不少的 outlier，而新加入的參數也理當會有相當的 outlier，因此未經過處理前，效果是不會很好的。

可以知道並不是每份資料都是有相當的作用的，了解資料分布情形，從其中獲取適當的資訊，才可以讓 predict 的準確度提高。

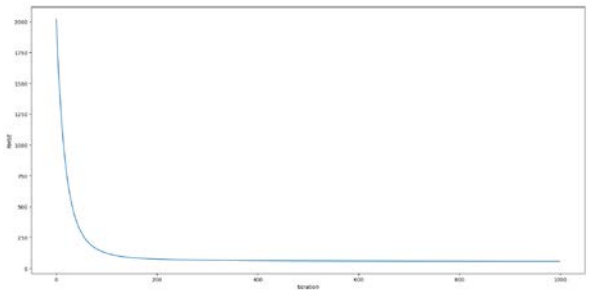
2. (2%) 請分別使用至少四種不同數值的 learning rate 進行 training（其他參數需一致），作圖並且討論其收斂過程。

```
iteration = 1000
datasize = 5652
feather = 9
```

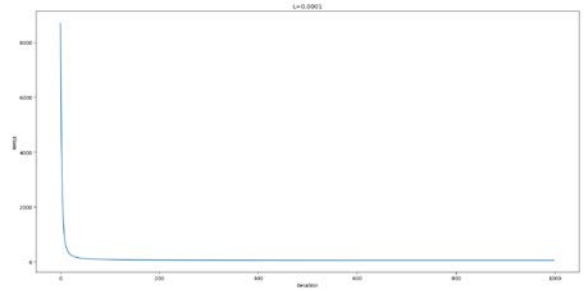
L = 0.00001



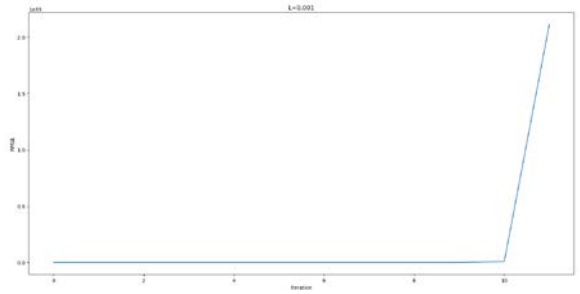
L= 0.00005



L = 0.0001



L = 0.001



可以從上面四張圖看出以下幾點結論：

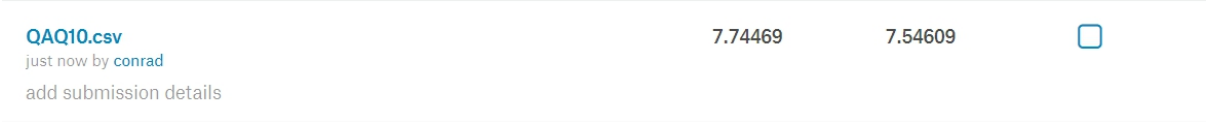
(1) L 在一定範圍內，其值越大收斂越快，不過若是超過一定的數值，則會發散無法收斂。

(2) Iteration 在一定次數之後就無法再降低 RMSE，要在降低 RMSE 可能需要對 Train.csv 再多做一些處理。

(3) 綜合以上結論，可以知道 L 是跟收斂速度有關係，而 RMSE 最後的數值還是依靠原先建立的模型以及對 data 的預處理。

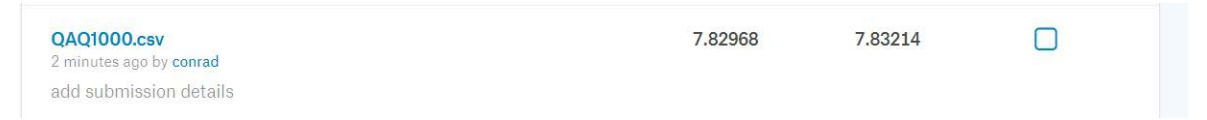
3. (1%) 請分別使用至少四種不同數值的 regulization parameter λ 進行 training（其他參數需一至），討論其 root mean-square error（根據 kaggle 上的 public/private score）。

λ = 10(private/public)

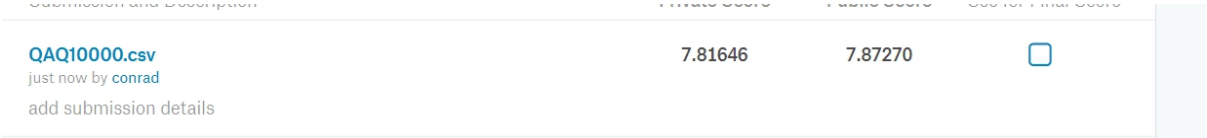


λ = 100(private/public)

λ = 1000(private/public)



λ = 10000(private/public)



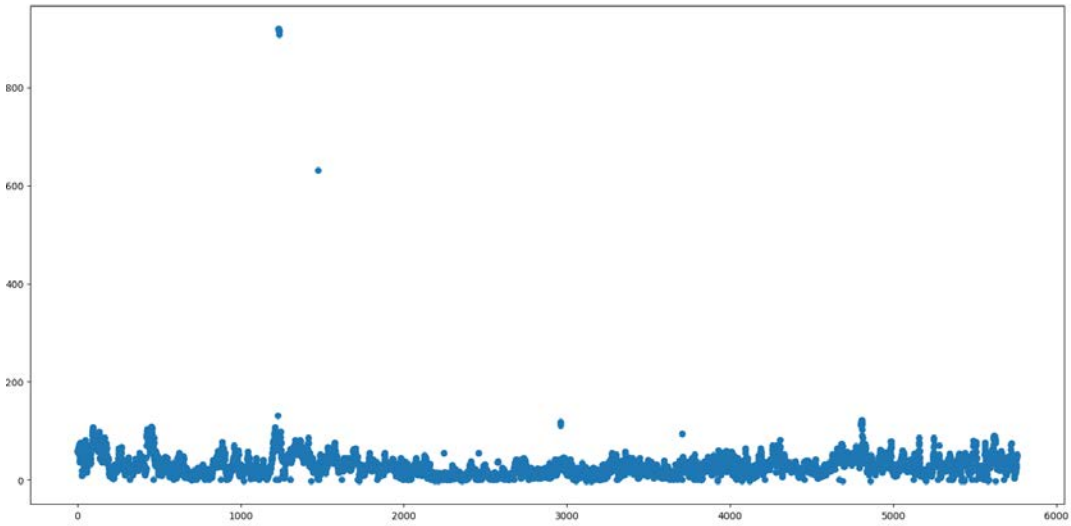
這邊可以看出來，在 hw_best.py 中加入 regularization 項之後，可以看出來其實準確度是降低的，可能的原因如下：

1. 本來的 function 並沒有 overfitting，加入 regularization 之後反倒會降低正常 weight 的效果，進而影響 predict 的結果。
2. 在 hw_best.py 中，所使用的參數只有 9 個變數，都是簡單的一次式，而 regularization 是希望可以將模型中高次項的 weight 降低，因此這個方式不適用在我所設計的模型之中。
3. 綜合以上結論，可以知道要使用 regularization 時應該要使用的時機是，
   (a) 當模型 overfitting
   (b) 模型很複雜，會降低準確度
   在這兩種情形發生時，再使用這個方式，才可以達到 regularization 的效果。

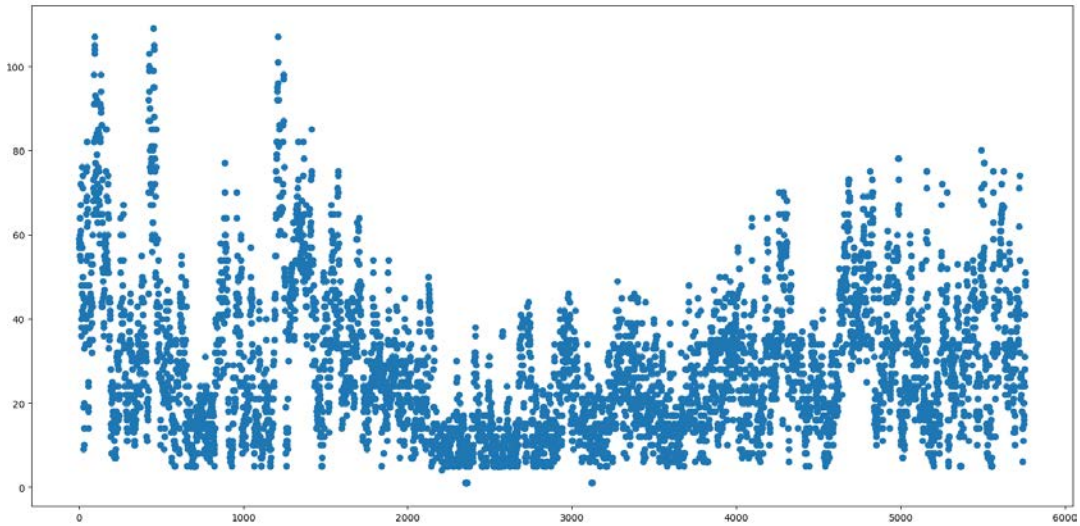
4. (1%) 請這次作業你的 best_hw1.sh 是如何實作的？（e.g. 有無對 Data 做任何 Preprocessing？Features 的選用有無任何考量？訓練相關參數的選用有無任何依據？）

(1)Data Preprocessing(remove outlier)：

有預先將 train.csv 做預處理，因為將 PM2.5 的數值放到圖上之後發現，會有一些特別高或是特別低的點(e.g.上一個是 80 幾下一個 400，或是最低竟然是負的)，這些 outlier 會嚴重影響預測結果，因此在預測前，要先將這些 outlier 去除，我應用的方式比較簡單，我是觀察每季(春夏秋冬)，這些季節大概 PM 的最大值為多少，設定一個 threshold 將太高和太低的值改成前一個正常的值，如下圖：



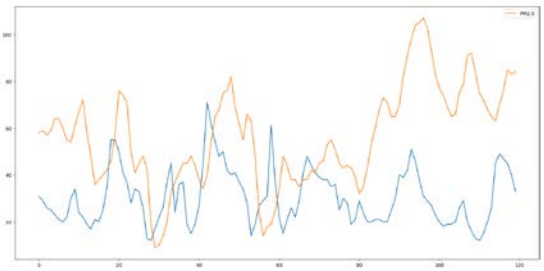
帶有 outlier 示意圖



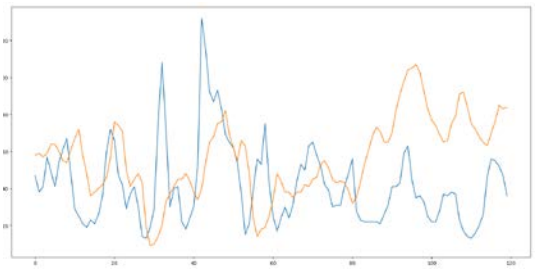
處理過後的樣本

(2)Feature selection：

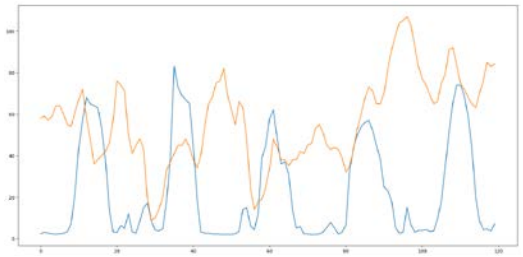
我是將每張圖的 feature 與 PM2.5 一起 PLOT 出來，觀察他們之間波動的相關性，可以發現比較有關聯的 feature 只有 PM2.5、PM10、NOX 等等，最後對於這五個 feature 個別比較，以及組合比較，可以得到最後的結果是只選用 PM2.5 效果最好，如下圖：



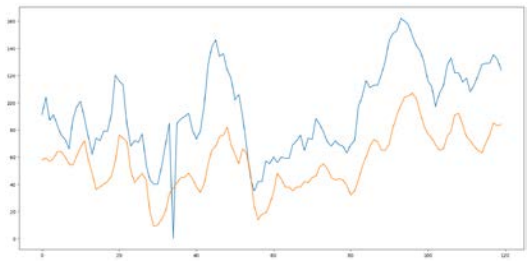
PM2.5vsNO



PM2.5 vs NOx



PM2.5vsO3



PM2.5 vs PM10

可以大致看出，若是這幾個 feature 有波動，PM2.5 也會受到些許影響

最後在比較 feature 選用時，發現只選擇 PM2.5 效果最好

(3)參數選用：

在訓練的時候，有將 train_error 在每一次 epoch 中 print 出來，可以發現在 epoch 數值為多少時，基本上已經飽和，便取用這樣的數值做運算(如第二題的圖)，Learning rate，也是經由觀察得來，L 如果太大 W 就不會收斂，因此在 epoch 時，觀察 W 的收斂狀況，可以知道選擇的 L 適合不適合，如第二題的圖。