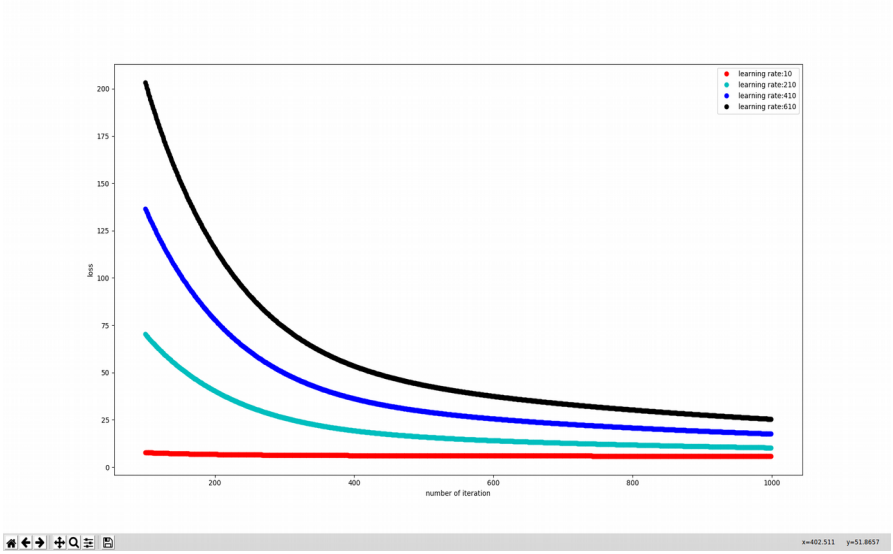
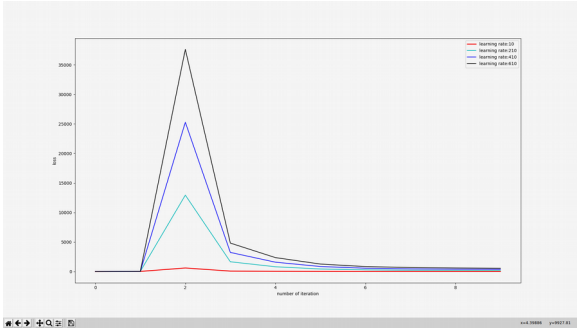
學號:B06901053
系級:電機三
姓名:謝承延

1. (2%) 使用四種不同的 learning rate 進行 training (其他參數需一致),作圖並討論其收斂過程(橫軸為 iteration 次數,縱軸為 loss 的大小,四種 learning rate 的收斂線請以不同顏色呈現在一張圖裡做比較)。



紅色：learning rate:10
淺藍色：learning rate:210
藍色：learning rate:410
黑色：learning rate:610
LOSS 以 root mean square 去計算
由圖可見 learning rate :10 的 loss 一直維持在比較低的位置，可知 weight 的值應不會太大，因為 model 的實做方法是先假設 weight 全為 0
learning rate:210 在 iteration 1000 次時達到 loss:10.294331441780066
learning rate:410 則是達到 loss:17.585774447350378
learning rate:610 達到 loss:25.38259841295957
learning rate:10 則是從 100 次 iteration 後就一直維持 7.5 以下
最終達到 5.8610160952504895
除了 10 以外，大部分的 learning rate 都是在 iteration 超過 100 之後開始有極劇的 loss 下降,然而
在 iteration 100 次之前，卻曾經歷 loss 非常高升的狀況
如圖



2. (1%) 比較取前 5 hrs 和前 9 hrs 的資料(5*18 + 1 v.s 9*18 + 1)在 validation set 上預測的結果,並說明造成的可能原因

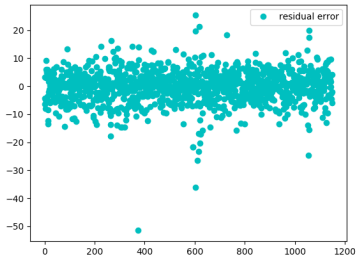
trainset validation set 區分說明：我利用隨機選擇的方式，抽出了 80%的資料給 train_set 20%的資料給 validation set,也就是說因為 train set 每次都不一樣 結果可能會有所不同，因此 5hr 9hr 將各分別做兩次結果預測

```
msk = np.random.rand(len(x)) < 0.8
train_x = x[msk]
train_y = y[msk]
validation_x = x[np.invert(msk)]
validation_y = y[np.invert(msk)]
```

learning rate 都取 10 iteration time:1000

取前 9hr：

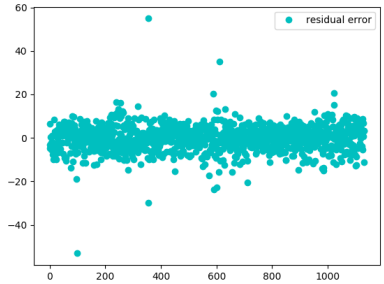
result1: residual error:



root mean square and r2_score:

```
root mean square:5.968015629372857
r2 score:0.8780499509095988
```

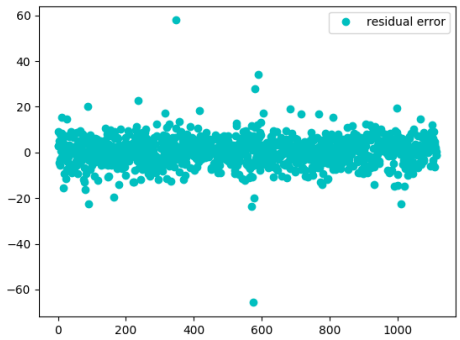
result2: residual error:



root mean square and r2_score:

```
root mean square:6.037953171612951
r2_score:0.8566388969771341
```

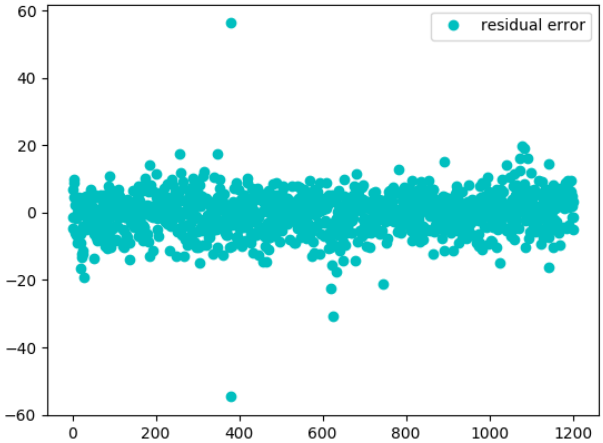
取前 5hr:



root mean square and r2_score:

```
root mean square:6.407319734941829
r2 score:0.8433131170665182
```

result2:



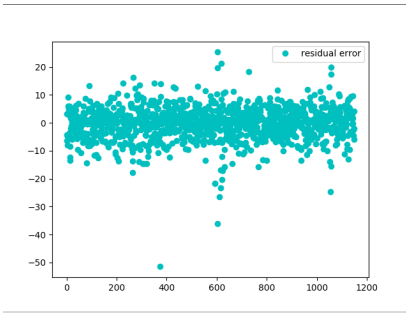
root mean square and r2_score:

```
root mean square:5.955598544406546
r2_score:0.8701791261793782
```

由以上結果發現,取前 9hr 的實驗結果稍微優於取前 5hr 的結果，然而變動卻不大，這可能的原因是此 model 在預測出下一個小時 pm2.5 的時候，6～9hr 的 data 權重是相對較輕的，因為權重較輕，故自然就不會對結果產生太大的影響，然而結果還是有差異，可以知道 9 個小時的 data 數量較多仍然對預測結果有正面的影響

3. (1%) 比較只取前 9 hrs 的 PM2.5 和取所有前 9 hrs 的 features(9*1 + 1 vs. 9*18 + 1) 在 validation set 上預測的結果,並說明造成的可能原因。

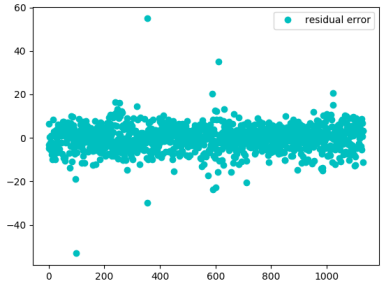
Iteration number 和 learning rate 同上:
取前 9hr 所有 feature:
result1: residual error:



root mean square and r2_score:

```
root mean square:5.968015629372857
r2 score:0.8780499509095988
```

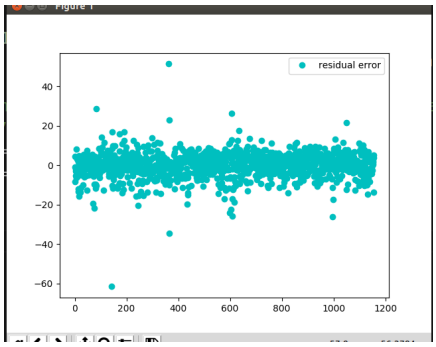
result2: residual error:



root mean square and r2_score:

```
root mean square:6.037953171612951
r2_score:0.8566388969771341
```

取前 9hr pm2.5:
residual error:



root mean square and r2_score:

```
root mean square:6.4592273754846525
r2_score:0.8396053910160273
```

feature 變成只取 PM2.5 以後,誤差明顯變大 r2_score 也下降了不少,這代表在取所有 feature 的 model 當中,其他的 feature 在預測 PM2.5 的任務中扮演了蠻重要的角色（權重）,當把其他 feature 的資料拿掉以後,自然預測就無法更加精準,順代一提 train 的過程中 gradient 的數值最後都相當接近 0，這可以排除 underfitting 造成此現象的可能

```
gradient: [-6.78136871e-08]
```

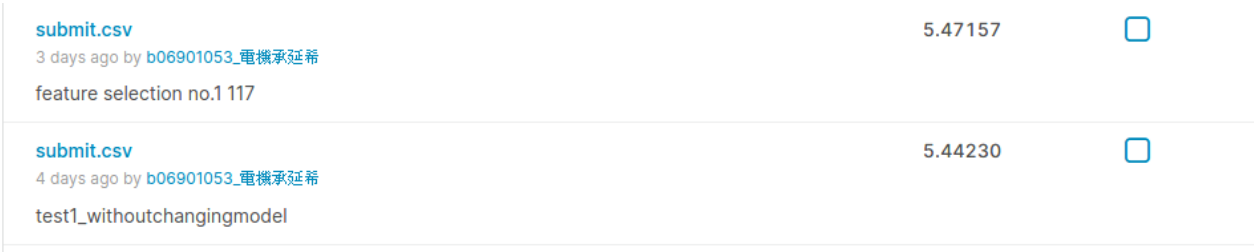
4. (2%) 請說明你超越 baseline 的 model 是如何實作的(例如:怎麼進行 feature selection, 有沒有做 pre-processing、learning rate 的調整、advanced gradient descent 技術、不同的 model 等等)。

我跟 colab 使用的 gradient descent 技術相同都是 adagram，發現 learning rate 只要不要調整的太大 performance 影響並不大，最終我將 learning rate 定在 10
接著我開始將重點放在 feature selection
我先使用 sklearn 中的 selectKBest 來找出 f_regression 得分前幾名高的 feature（81～89 都是 pm2.5 72～80 則是 pm10）,目的只是要了解哪些 feature 在這個 model 的預測結果上可能很重要

	scores	features
89	23547.317717	90
88	9115.591333	89
80	5874.824452	81
79	4579.226522	80
87	4497.531805	88
78	3369.723640	79
86	3255.950750	87
77	2515.329276	78
85	2437.165734	86
76	1920.399590	77

```
bestfeatures = SelectKBest(f_regression, k=90)
```

如上面兩張圖所示
然而我也發現 如果直接用 selectKbest 的 fit method 去選出前幾名 feature 拿來使用，performance 並不會有多少改善，甚至在 kaggle 上表現更差了（score 比用全部的 feature 的 model 還要高）



因此我決定改採用 backward selection 的方式一個一個刪除 feature

我決定要不要刪除此 feature 的標準為以下幾點
1.validation set 的 score 要下降
2.查看 test.csv 的前三項數據
3.先從 selectKbest 中較不重要的 feature（score 比較小）開始嘗試刪除

在此要特別說明第二點，我查看了自己上傳的預測值 其中發現 score 越高的，他的第一項跟第三項數據都越高，而第二項數據則越低，因此我預測第一項跟第三項要變小,第二項要變大 預測結果才算是變好,倘若這三者都符合就滿足第二項

最後通過樸實無華又枯燥的不斷測試 並且刪除 feature 後 我使用的 feature 有:

AMB_TEMP	前 9 個小時
CO	前 9 個小時
NMHC	前 9 個小時
NO	前 9 個小時
NO2	前 9 個小時
NOx	前 9 個小時
O3	前 9 個小時
PM10	前 9 個小時
PM2.5	前 9 個小時
RAINFALL	前 9 個小時
RH	前 9 個小時
WD_HR	前 9 個小時
WIND_SPEED	後 2 個小時

Normalization:

除了 pm2.5,我把每一項都剪掉平均數 再除以標準差

因為我想讓 pm2.5 對整個 model 的影響力最大