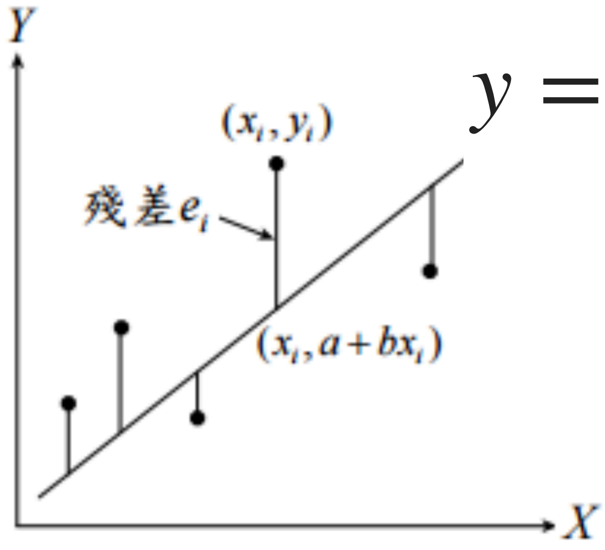
HW5

LINEAR REGRESSION FOR PM2.5 PREDICTION

OUTLINE

- Linear regression
- Problem explanation
- Assignments & Grading
- Submission & Rules

LINEAR REGRESSION



$$y = a + b * x$$

- 回歸直線（高中年代）
  - 給定一堆 $(y^1, x_1^1)\ \ (y^2, x_1^2)\ldots.$

    希望找到一組w使得預測值跟真實的值很接近

$$y = w_0 + x_1 * w_1$$

- 線性迴歸（linear regression)
  - 給定一堆 $(y^1, (x_1^1, x_2^1, \ldots, x_N^1))\ \ (y^2, (x_1^2, x_2^2, \ldots, x_N^2))\ldots.$

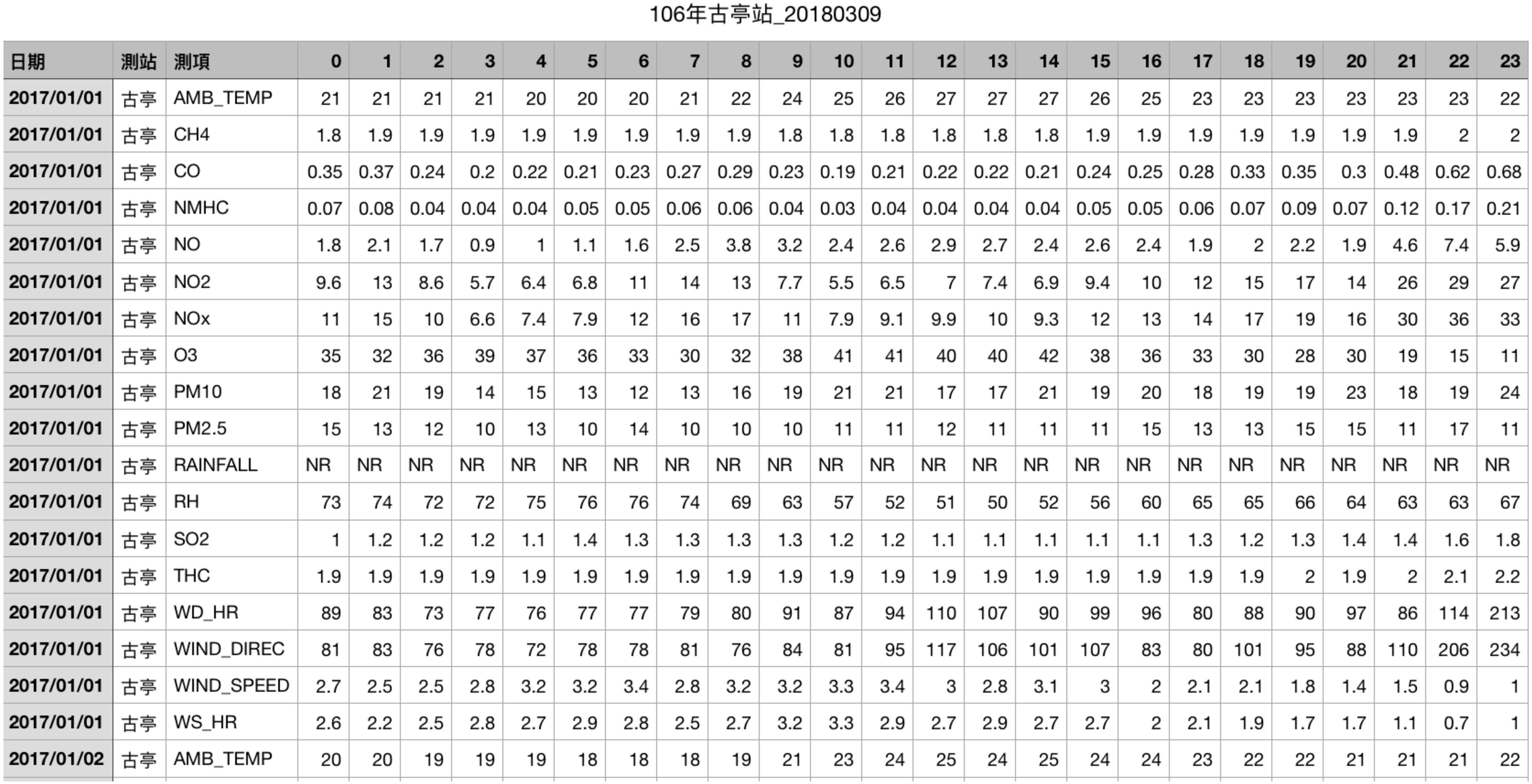
    希望找到一組w 使得預測值跟真實的值很接近

$$y = w_0 + \sum^{N} x_i * w_i$$

ON PM2.5 PREDICTION ?

- 以小時為單位，利用前N小時的資料來預測下一個小時的 ”PM2.5”
    - if N = 3: Jan/1 data[00:00, 01:00, 02:00] –> Jan/1 pm2.5[03:00]

DATA



106年古亭站_20180309

日期	測站	測項	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23
2017/01/01	古亭	AMB_TEMP	21	21	21	21	20	20	20	21	22	24	25	26	27	27	27	26	25	23	23	23	23	23	23	22
2017/01/01	古亭	CH4	1.8	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.8	1.8	1.8	1.8	1.8	1.8	1.9	1.9	1.9	1.9	1.9	1.9	1.9	2	2
2017/01/01	古亭	CO	0.35	0.37	0.24	0.2	0.22	0.21	0.23	0.27	0.29	0.23	0.19	0.21	0.22	0.22	0.21	0.24	0.25	0.28	0.33	0.35	0.3	0.48	0.62	0.68
2017/01/01	古亭	NMHC	0.07	0.08	0.04	0.04	0.04	0.05	0.05	0.06	0.06	0.04	0.03	0.04	0.04	0.04	0.04	0.05	0.05	0.06	0.07	0.09	0.07	0.12	0.17	0.21
2017/01/01	古亭	NO	1.8	2.1	1.7	0.9	1	1.1	1.6	2.5	3.8	3.2	2.4	2.6	2.9	2.7	2.4	2.6	2.4	1.9	2	2.2	1.9	4.6	7.4	5.9
2017/01/01	古亭	NO2	9.6	13	8.6	5.7	6.4	6.8	11	14	13	7.7	5.5	6.5	7	7.4	6.9	9.4	10	12	15	17	14	26	29	27
2017/01/01	古亭	NOx	11	15	10	6.6	7.4	7.9	12	16	17	11	7.9	9.1	9.9	10	9.3	12	13	14	17	19	16	30	36	33
2017/01/01	古亭	O3	35	32	36	39	37	36	33	30	32	38	41	41	40	40	42	38	36	33	30	28	30	19	15	11
2017/01/01	古亭	PM10	18	21	19	14	15	13	12	13	16	19	21	21	17	17	21	19	20	18	19	19	23	18	19	24
2017/01/01	古亭	PM2.5	15	13	12	10	13	10	14	10	10	10	11	11	12	11	11	11	15	13	13	15	15	11	17	11
2017/01/01	古亭	RAINFALL	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR	NR
2017/01/01	古亭	RH	73	74	72	72	75	76	76	74	69	63	57	52	51	50	52	56	60	65	65	66	64	63	63	67
2017/01/01	古亭	SO2	1	1.2	1.2	1.2	1.1	1.4	1.3	1.3	1.3	1.3	1.2	1.2	1.1	1.1	1.1	1.1	1.1	1.3	1.2	1.3	1.4	1.4	1.6	1.8
2017/01/01	古亭	THC	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9	1.9	2	1.9	2	2.1	2.2
2017/01/01	古亭	WD_HR	89	83	73	77	76	77	77	79	80	91	87	94	110	107	90	99	96	80	88	90	97	86	114	213
2017/01/01	古亭	WIND_DIREC	81	83	76	78	72	78	78	81	76	84	81	95	117	106	101	107	83	80	101	95	88	110	206	234
2017/01/01	古亭	WIND_SPEED	2.7	2.5	2.5	2.8	3.2	3.2	3.4	2.8	3.2	3.2	3.3	3.4	3	2.8	3.1	3	2	2.1	2.1	1.8	1.4	1.5	0.9	1
2017/01/01	古亭	WS_HR	2.6	2.2	2.5	2.8	2.7	2.9	2.8	2.5	2.7	3.2	3.3	2.9	2.7	2.9	2.7	2.7	2	2.1	1.9	1.7	1.7	1.1	0.7	1
2017/01/02	古亭	AMB_TEMP	20	20	19	19	19	18	18	18	19	21	23	24	25	24	25	24	24	23	22	22	21	21	21	22

DATA

- **18** features for each hour
  - {CH4, CO, NO, NO2, PM2.5…..}
- training data(train.csv)
  - 12個月的前20天
- testing data(test.csv
  - 12個月的21日之後

PROBLEM EXPLANATION

$$(y^1, (x_1^1, x_2^1, \ldots, x_N^1))$$

- 將前N小時的feature串起來成為x，target(y)則是這一小時的PM2.5數值，這樣就形成了一筆data。

$$(y, [x_1, x_2, \ldots, x_?])$$

?

$$(y, [x_1, x_2, \ldots, x_?, 1])$$

$$y = w_0 + \boxed{\sum x_i * w_i}$$

$$[x_1, x_2, \ldots, x_?, 1] * [w_1, w_2, \ldots, w_?, w_0]^T$$

- 對每一小時都做出data，拼起來得到X,Y_real

$$\begin{bmatrix} y^1 \\ y^2 \\ \cdot\ \cdot \\ y^M \end{bmatrix} \quad <=> \quad \begin{bmatrix} x_1^1, x_2^1, \ldots, x_?^1, 1 \\ x_1^2, x_2^2, \ldots, x_?^2, 1 \\ \cdot\ \cdot \\ x_1^M, x_2^M, \ldots, x_?^M, 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ \cdot\ \cdot \\ w_? \\ w_0 \end{bmatrix}$$

$$Y_{real}$$

$$Y_{predict} = X * W$$

PROBLEM EXPLANATION

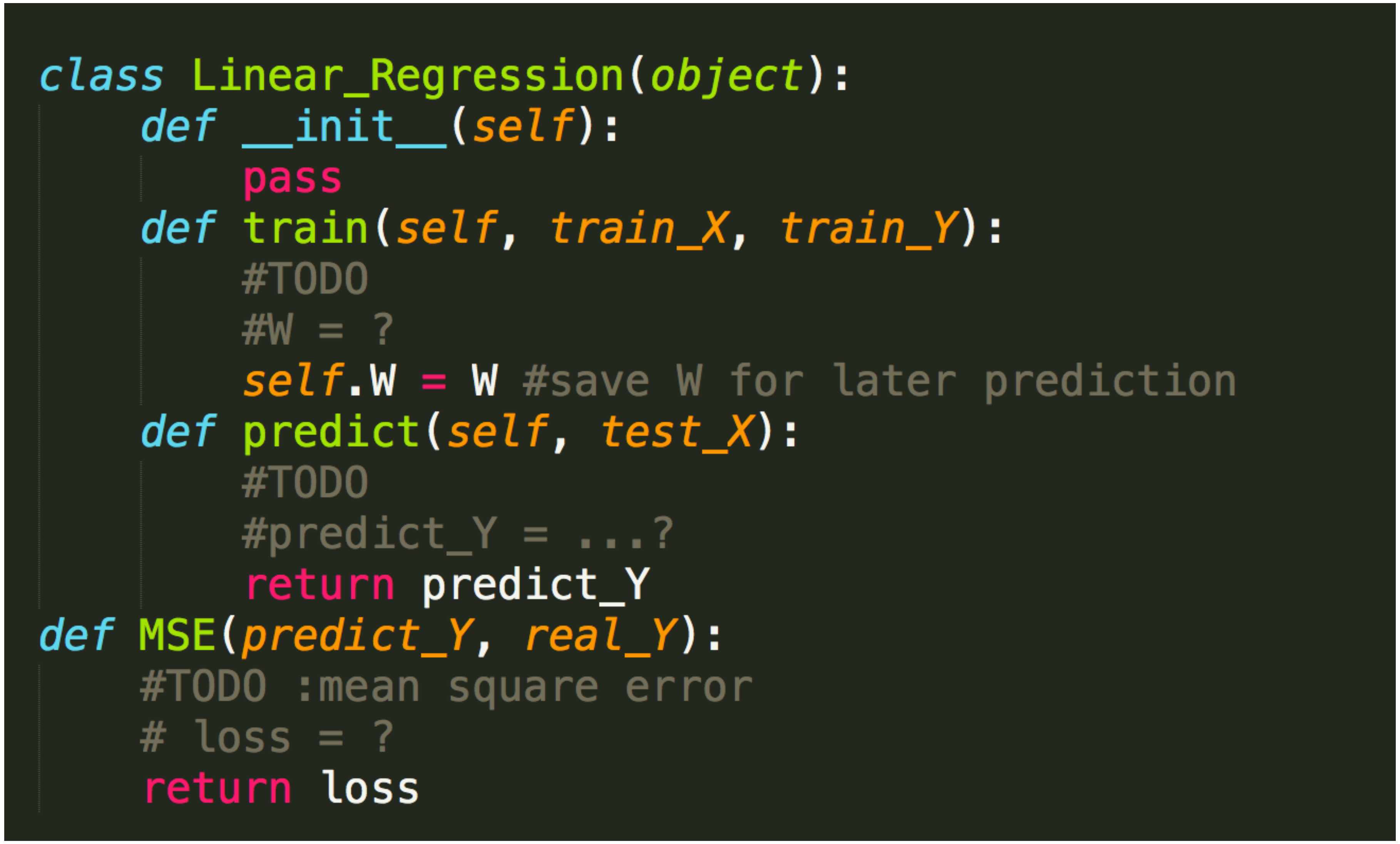
- 希望Y_predict 和Y_real 很接近(error小)
  - 衡量error : 對Y_predict和Y_real算Mean Square Error (MSE)，可以得到error的程度。
- 利用上頁方法得到X_train, Y_train, X_test, Y_test：
  - Train : 利用X_train, Y_train找到一個最好的W使得 training error 最小

$$L(W) = 1/M * ((Y_{train} - X_{train} * W)^T * (Y_{train} - X_{train} * W))$$

    - 提示：對W微分，極值在一階導函數=0的地方
  - Test: 用上述的W對X_test做預測，看看是否與Y_test很接近(testing error)

ASSIGNMENTS & GRADING

- Q1. (3%) 完成main.py中TODO 的部分

- 

```python
class Linear_Regression(object):
    def __init__(self):
        pass
    def train(self, train_X, train_Y):
        #TODO
        #W = ?
        self.W = W #save W for later prediction
    def predict(self, test_X):
        #TODO
        #predict_Y = ...?
        return predict_Y
def MSE(predict_Y, real_Y):
    #TODO :mean square error
    # loss = ?
    return loss
```

ASSIGNMENTS & GRADING

report.pdf

- Q2. (1%) 將training error 和testing error 對N=1~48作圖，並畫在同一張圖上，並解釋兩者變化的趨勢。(x軸是N, y軸是loss)

- Q3. (2%) 改進目前的方法，並將方法及結果'詳細'寫在報告中。(會根據你誠意給分)

- note1: 其實load train & test set 的部分我幫你們寫好了（佛吧！

- note2: Q1 main.py 裡面的TODO完成後，其他地方不用/不要動，執行就會output一個ans.txt，我會根據這個評分

- note3 : Q2, Q3 有圖附圖，解釋務必清楚完整

ASSIGNMENTS & GRADING

- something you can do for Q3(can only choose one from below)
    - ex1: $y = w_0 + \sum x_i * w_i$ 如果少了w0這項？(1%)
    - ex2: 18 features有些可能不需要？拿掉試試看 ？(1%)
    - ex3: 作業會用到numpy 的matrix inverse ，解釋它是怎麼做的？如果matrix non-invertible 也可以嗎？ (2%)
    - or anything you think worth trying …

SUBMISSION

- complete main.py and write your answers and results of Q2, Q3 in the report.pdf
- put your files above in a folder

```
|– ./b01234567_hw5
    |– main.py
    |– report.pdf
```

- compress it into a zip file and upload your b01234567_hw5.zip to CEIBA

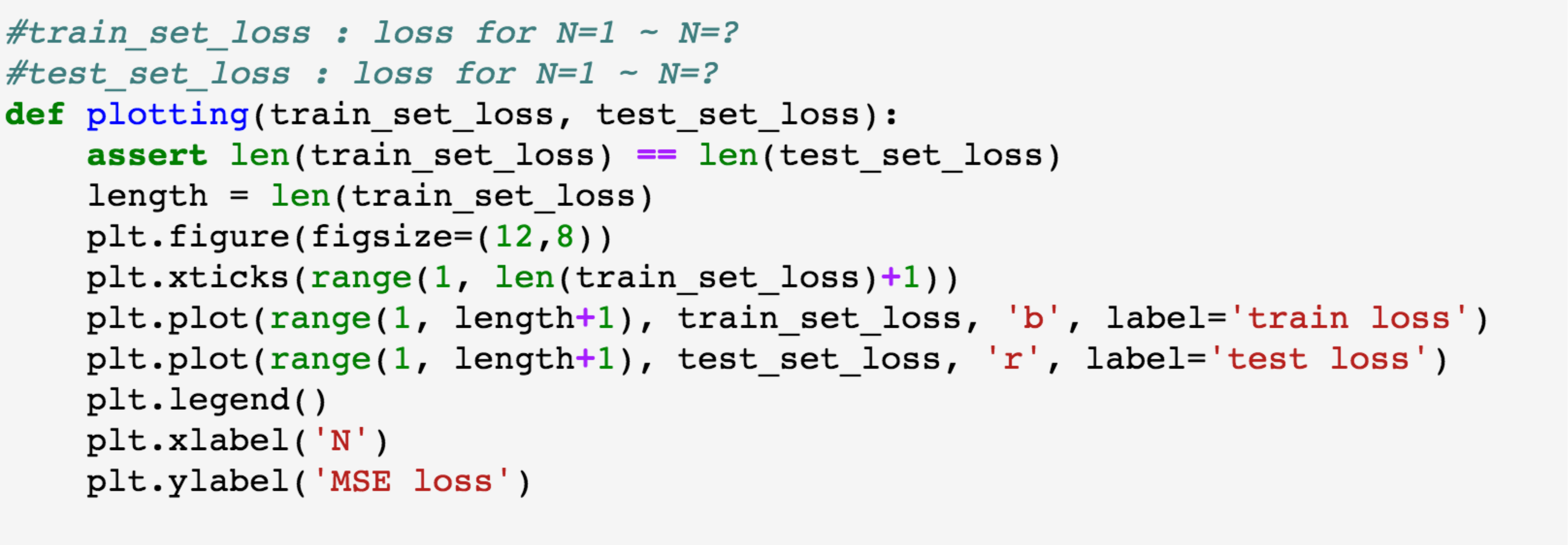
RULES

- Plagiarism = 0 point
- Deadline : 12/20 23:59:59
- Late submission: total score * 0.8 (per day)
- Any other error : total score * 0.8
- if questions for me, please send me email or post them on FB group. FB msg won’t be replied.
- 陳元瑞 r07922070@ntu.edu.tw

some tips for HW5

陳元瑞 r07922070@ntu.edu.tw

Plotting function



```python
#train_set_loss : loss for N=1 ~ N=?
#test_set_loss : loss for N=1 ~ N=?
def plotting(train_set_loss, test_set_loss):
    assert len(train_set_loss) == len(test_set_loss)
    length = len(train_set_loss)
    plt.figure(figsize=(12,8))
    plt.xticks(range(1, len(train_set_loss)+1))
    plt.plot(range(1, length+1), train_set_loss, 'b', label='train loss')
    plt.plot(range(1, length+1), test_set_loss, 'r', label='test loss')
    plt.legend()
    plt.xlabel('N')
    plt.ylabel('MSE loss')
```