

Work 1 ——线性回归

本次目标：由前 9 个小时的 18 个 features(包含 PM2.5)预测第 10 个小时的 PM2.5。

步骤：

加载'train.csv'：数据读取格式为'big5'，在 train.csv 中，为 12 个月，每个月取 20 天，每天 24 小时的资料（每个小时的资料有 18 个特征）

```
data = pd.read_csv('./train.csv', encoding = 'big5')
```

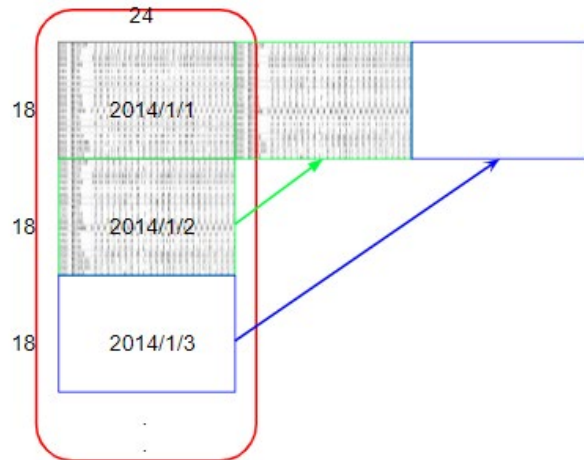
	日期	测站	测项	0	1	2	3	4	5	6	...	14	15	16	17	18	19	20	21	22	23
0	2014/1/1	豊原	AMB_TEMP	14	14	14	13	12	12	12	...	22	22	21	19	17	16	15	15	15	15
1	2014/1/1	豊原	CH4	1.8	1.8	1.8	1.8	1.8	1.8	1.8	...	1.8	1.8	1.8	1.8	1.8	1.8	1.8	1.8	1.8	1.8
2	2014/1/1	豊原	CO	0.51	0.41	0.39	0.37	0.35	0.3	0.37	...	0.37	0.37	0.47	0.69	0.56	0.45	0.38	0.35	0.36	0.32
3	2014/1/1	豊原	NMHC	0.2	0.15	0.13	0.12	0.11	0.06	0.1	...	0.1	0.13	0.14	0.23	0.18	0.12	0.1	0.09	0.1	0.08
4	2014/1/1	豊原	NO	0.9	0.6	0.5	1.7	1.8	1.5	1.9	...	2.5	2.2	2.5	2.3	2.1	1.9	1.5	1.6	1.8	1.5
...
4315	2014/12/20	豊原	THC	1.8	1.8	1.8	1.8	1.8	1.7	1.7	...	1.8	1.8	2	2.1	2	1.9	1.9	1.9	2	2
4316	2014/12/20	豊原	WD_HR	46	13	61	44	55	68	66	...	59	308	327	21	100	109	108	114	108	109
4317	2014/12/20	豊原	WIND_DIREC	36	55	72	327	74	52	59	...	18	311	52	54	121	97	107	118	100	105
4318	2014/12/20	豊原	WIND_SPEED	1.9	2.4	1.9	2.8	2.3	1.9	2.1	...	2.3	2.6	1.3	1	1.5	1	1.7	1.5	2	2
4319	2014/12/20	豊原	WS_HR	0.7	0.8	1.8	1	1.9	1.7	2.1	...	1.3	1.7	0.7	0.4	1.1	1.4	1.3	1.6	1.8	2

预处理：取需要的数值部分，将'Rainfall'栏位全部补 0

```
data = data.iloc[:, 3:]
data[data == 'NR'] = 0
raw_data = data.to_numpy()
```

特征提取： 取需要的数值部分， 将'Rainfall'栏位全部补 0。

Extract Features



Extract Features

Pseudo code

- 1 | Declare a 18-dim vector (Data)
- 2 | for i_th row in training data :
- 3 | Data[i_th row%18].append(every element in i_th row)

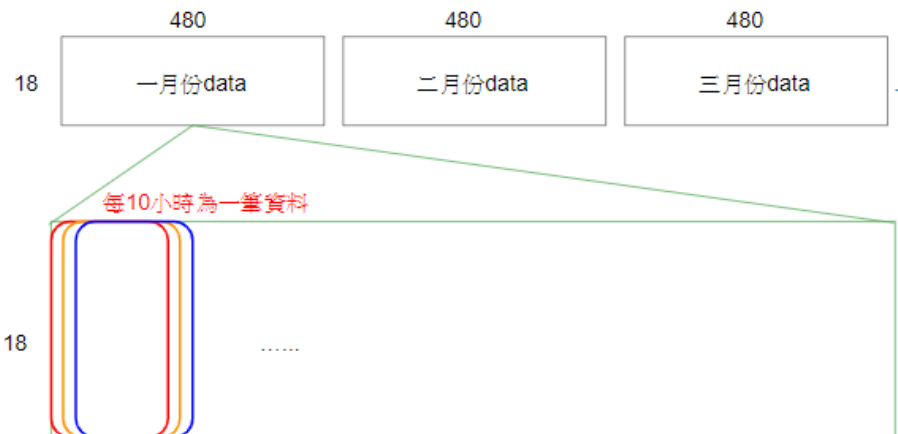
Data will become a vector like

2014/1/1	2014/1/2	2014/1/3
----------	----------	----------

将资料依照每个月份重组成 12 个 18 特征 480 小时 ($24 * 20$) 的资料。

```
month_data = {}
for month in range(12):
    sample = np.empty([18, 480])
    for day in range(20):
        sample[:, day * 24 : (day + 1) * 24] = raw_data[18 * (20 * month + day) : 18 * (20 * month + day + 1), :]
    month_data[month] = sample
```

Extract Features



Extract Features

Pseudo code

- 1 | Declare train_x for previous 9-hr data, and train_y for 10th-hr pm2.5
- 2 | for i in all the given data:
- 3 | sample every 10 hrs :
- 4 | train_x.append(previous 9-hr data)
- 5 | train_y.append(the value of 10th-hr pm2.5)
- 6 | add a bias term to every data in train_x

每个月有 480 小时，每 9 个小时形成一个 data，每个月就有 471 个 data。12 个月则有 $12 * 471$ 个 data。每个 data 有 $18 * 9$ 个特征值， $12 * 471$ 个 target 值

```
x = np.empty([12 * 471, 18 * 9], dtype = float)
y = np.empty([12 * 471, 1], dtype = float)
for month in range(12):
    for day in range(20):
        for hour in range(24):
            if day == 19 and hour > 14:
                continue
            x[month * 471 + day * 24 + hour, :] = month_data[month]
            [:, day * 24 + hour : day * 24 + hour + 9].reshape(1, -1) #vector dim:18*9 (9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9 9)
            y[month * 471 + day * 24 + hour, 0] = month_data[month]
            [9, day * 24 + hour + 9] #value
```

标准化：待完成

训练数据：使用线性回归训练数据

Implement linear regression

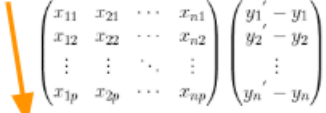
Pseudo code

- 1 | Declare weight vector, initial lr ,and # of iteration
- 2 | for i_th iteration :
- 3 | y' = the inner product of train_x and weight vector
- 4 | Loss = $y' - \text{train_y}$
- 5 | gradient = $2 * \text{np.dot}((\text{train_x})', L)$
- 6 | weight vector -= learning rate * gradient

3.
$$\begin{pmatrix} y_1' \\ y_2' \\ \vdots \\ y_n' \end{pmatrix} = \begin{pmatrix} x_{11} & \cdots & x_{1p} \\ x_{21} & \cdots & x_{2p} \\ \vdots & \ddots & \vdots \\ x_{n1} & \cdots & x_{np} \end{pmatrix} \cdot \begin{pmatrix} w_1 \\ w_2 \\ \vdots \\ w_p \end{pmatrix}$$

4.
$$L = \begin{pmatrix} y_1' \\ y_2' \\ \vdots \\ y_n' \end{pmatrix} - \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}$$

5. gradient = $2 \times$



$$\begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1p} \\ x_{21} & x_{22} & \cdots & x_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{np} \end{pmatrix} \begin{pmatrix} y_1' - y_1 \\ y_2' - y_2 \\ \vdots \\ y_n' - y_n \end{pmatrix}$$

p-dim vector

Adagrad

Pseudo code

- 1 | Declare weight vector, initial lr ,and # of iteration
 Declare prev_gra storing gradients in every previous iterations
- 2 | for i_th iteration :
- 3 | y' = the inner product of train_x and weight vector
- 4 | Loss = $y' - \text{train_y}$
- 5 | gradient = $2 * \text{np.dot}((\text{train_x})', L)$
 prev_gra += gra**2
 ada = np.sqrt(prev_gra)
- 6 | weight vector -= learning rate * gradient / ada

测试：载入 test.csv，根据和训练数据相同的预先处理和特征提取的方式处理，使得形成 240 个 $18 * 9 + 1$ 的资料。test.csv 如下图

	id_0	AMB_TEMP	21	21.1	20	20.1	19	19.1	19.2	18	17
0	id_0	CH4	1.7	1.7	1.7	1.7	1.7	1.7	1.7	1.7	1.8
1	id_0	CO	0.39	0.36	0.36	0.4	0.53	0.55	0.34	0.31	0.23
2	id_0	NMHC	0.16	0.24	0.22	0.27	0.27	0.26	0.27	0.29	0.1
3	id_0	NO	1.3	1.3	1.3	1.3	1.4	1.6	1.2	1.1	0.9
4	id_0	NO2	17	14	13	14	18	21	8.9	9.4	5
...
4314	id_239	THC	1.8	1.8	1.8	1.8	1.7	1.7	1.7	1.7	1.7
4315	id_239	WD_HR	80	92	95	95	96	97	96	96	84
4316	id_239	WIND_DIREC	76	99	93	97	93	94	98	97	65
4317	id_239	WIND_SPEED	2.2	3.2	2.5	3.6	5	4.2	5.7	4.9	3.6
4318	id_239	WS_HR	1.7	2.8	2.6	3.3	3.5	5	4.9	5.2	3.6

通过 test.csv 保存预测值至 csv 文件，并上交：

csv 文件名：学号.csv

完成方式：MatLab/Python

以上代码：仅供参考

文件上交至 2025227034@stu.suda.edu.cn，截止至期末