期中報告__工業生產指數迴歸分析

姓名: 王富貴　　學號: D1146202

一、問題定義

問題一、分析不同工業對總指數的影響

- 分析方式 :
  - 簡單線性迴歸 → 其他工業和總指數個別作迴歸分析
  - 多元線性迴歸 → 其他工業都當作特徵並和總指數作迴歸分析
- 分析動機 :
  觀察台灣各種工業的發展情形對整體工業發展的影響
- 預期結果 :
  得到迴歸模型，並可從各種工業發展情況來預測台灣的總體工業發展情形

問題二、分析「電力及然氣供應業」和「用水供應業」對其他工業發展的影響

- 分析方式 :
  - 多元線性回歸 → 電力及然氣供應業、用水供應業(兩個特稱)和總指數及其他工業個別做迴歸分析
- 分析動機 :
  因為每項工業的發展幾乎都需要消耗電以及水，所以觀察「電力及然氣供應業」和「用水供應業」這兩種產業的發展對台灣其他工業發展的影響
- 預期結果 :
  得到迴歸模型，並可從「電力及然氣供應業」和「用水供應業」的發展情形，預測出其他工業的發展情形

二、資料來源

- 資料來源

  中華民國統計資訊網-總體統計資料庫 : https://statdb.dgbas.gov.tw/pxweb/Dialog/statfile9L.asp
- 資料說明
  - 資料名稱: 工業生產指數-月

  ▼ 工業生產指數介紹
  - 工業生產指數是經濟學家、政策制定者和投資者經常關注的經濟指標之一，因為它能夠提供有關經濟走勢的重要信息。
  - 反映出國家或地區的工業生產總體規模和增長速度。
  - 工業生產指數屬於經濟指標的領先指標，因為工業生產通常是經濟活動中的重要組成部分，其變化可能會影響到其他領域的經濟活動。
  - 工業生產指數可以反映出一個國家或地區的整體經濟狀況，因為工業生產通常是一個國家或地區經濟的重要組成部分。

  ▼ 資料欄位
  - 縱軸 : 西元年月
  - 橫軸(原始值) : Total_Index(總指數)、Manufacturing(製造業)、MetalMachinery_Electrical(金屬機電工業)、Information_Electronic(資訊電子工業)、Chemical(化學工業)、Consumer_Goods(民生工業)、Electricity_GasSupply(電力及燃氣供應業)、Water_Supply(用水供應業)

三、迴歸數學模型推導過程

▼ 符號定義

i : 第i個特徵

m : 第m筆資料

k : 迭代運算的第k次運算

$w^{(k)}$: 權重向量經過k次運算的結果

$w_i^{(k)}$: 權重向量中的第i個元素，經過k次迭代運算的結果

$x^{(m)}$: 第m筆輸入向量x(自變量)

$x_i^{(m)}$: 第m筆資料的第i個資料特徵

$y_t^{(m)}$: 資料樣本中的第m筆資料(因變量)

$y_p^{(k)(m)}$: 第m筆資料輸入樣本向量，經過迭代的預測值， $y_p^{(k)(m)} = w^{(k)} \cdot x^{(m)}$

$y_d^{(k)(m)}$: 預測值與因變量之間的差異， $y_d^{(k)(m)} = y_p^{(k)(m)} - y_t^{(m)}$

α : 學習率

▼ 推導過程

假設現在有一筆資料，有多個特徵值 x_i ， $i = \{0, 1, \ldots, n\}$，分析這些特徵值對因變量 y_t的影響 :

- **預測函式**
  令預測函式 $y_p^{(m)} = (w_0, w_1, \ldots, w_n)(x_0^{(m)}, x_1^{(m)}, \ldots, x_n^{(m)}) = w_0 x_0^{(m)} + w_1 x_1^{(m)} + \ldots + w_n x_n^{(m)}$
- **損失函式**
  - 則損失函式 L 為 : $L = (y_p^{(0)} - y_t^{(0)})^2 + (y_p^{(1)} - y_t^{(1)})^2 + \ldots + (y_p^{(M-1)} - y_t^{(M-1)})^2$，化簡後，可得Sum of Squared Errors，即 $SSE = \sum_{m=0}^{M-1} (y_p^{(m)} - y_t^{(m)})^2$
  - SSE會因資料筆數的數量而影響數值大小，所以進一步求Mean Squared Error，即 $MSE = \frac{1}{M} \sum_{m=0}^{M-1} (y_p^{(m)} - y_t^{(m)})^2$
  - 損失值為MSE二次曲線上的切點，透過損失函式的微分，可求得損失值，即
    $loss = \frac{1}{2M} \sum_{m=0}^{M-1} (y_p^{(m)} - y_t^{(m)})$
- 為了取得可以代表這筆料的超平面，所以要調整損失函示的權重，並透過梯度下降法的迭代，得到最小的損失值(最小均差平方和)
- **調整參數**
  - 對每個特徵值調整權重，所以損失函式針對每個特徵值的權重進行偏微分
  - 令 $L = \frac{1}{2M} \sum u^2$， $u = y_p^{(m)} - y_t^{(m)}$， $y_p^{(m)} = w_0 x_0^{(m)} + w_1 x_1^{(m)} + \ldots + w_n x_n^{(m)}$
  - 對其中一個特徵值的權重進行偏微分 : $\frac{\partial L}{\partial w_i} = \frac{1}{M} \sum_{m=0}^{M-1} y_d^{(k)(m)} \cdot x_i^{(m)}$

  ▼ 推導過程

$$
\begin{aligned}
\frac{\partial L}{\partial w_i} &= \frac{\partial L}{\partial u} \frac{\partial u}{\partial w_i} \\
&= \frac{1}{M} \sum_{m=0}^{M-1} (w_0 x_0^{(m)} + w_1 x_1^{(m)} + \ldots + w_n x_n^{(m)} - y_t^{(m)}) \cdot x_i^{(m)} \\
&= \frac{1}{M} \sum_{m=0}^{M-1} (y_p^{(m)} - y_t^{(m)}) \cdot x_i^{(m)} \\
&= \frac{1}{M} \sum_{m=0}^{M-1} y_d^{(k)(m)} \cdot x_i^{(m)}
\end{aligned}
$$

- **梯度下降法**:
  - 因為MSE會形成一個二次函數的圖形，為了求得最小的損失值，所以要找到該二次函式的低谷
  - 透過梯度下降法，每一回合都會更新一次參數(權重)來形成新的預測函式，並用新的預測函數進入下一回合
  - 每一回合進行的流程: $\begin{bmatrix} w_0^{(k+1)} \\ w_1^{(k+1)} \\ \vdots \\ w_n^{(k+1)} \end{bmatrix} = \begin{bmatrix} w_0^{(k)} \\ w_1^{(k)} \\ \vdots \\ w_n^{(k)} \end{bmatrix} - \alpha \cdot \frac{1}{M} \cdot y_d^{(k)(m)} \begin{bmatrix} x_0^{(k)} \\ x_1^{(k)} \\ \vdots \\ x_n^{(k)} \end{bmatrix}$

  ▼ 推導過程:

$$\begin{bmatrix} w_0^{(k+1)} \\ w_1^{(k+1)} \\ \vdots \\ w_n^{(k+1)} \end{bmatrix} = \begin{bmatrix} w_0^{(k)} \\ w_1^{(k)} \\ \vdots \\ w_n^{(k)} \end{bmatrix} - \alpha \begin{bmatrix} L_{w_0}\ (w_0^{(k)}, w_1^{(k)}, ..., w_n^{(k)}) \\ L_{w_1}\ (w_0^{(k)}, w_1^{(k)}, ..., w_n^{(k)}) \\ \vdots \\ L_{w_n}\ (w_0^{(k)}, w_1^{(k)}, ..., w_n^{(k)}) \end{bmatrix} = \begin{bmatrix} w_0^{(k)} \\ w_1^{(k)} \\ \vdots \\ w_n^{(k)} \end{bmatrix} - \alpha \begin{bmatrix} \frac{\partial L_{w_1}\ (w_0^{(k)}, w_1^{(k)}, ..., w_n^{(k)})}{\partial w_0} \\ \frac{\partial L_{w_0}\ (w_0^{(k)}, w_1^{(k)}, ..., w_n^{(k)})}{\partial w_1} \\ \vdots \\ \frac{\partial L_{w_n}\ (w_0^{(k)}, w_1^{(k)}, ..., w_n^{(k)})}{\partial w_n} \end{bmatrix} = \begin{bmatrix} w_0^{(k)} \\ w_1^{(k)} \\ \vdots \\ w_n^{(k)} \end{bmatrix} - \alpha$$

$$= \begin{bmatrix} w_0^{(k)} \\ w_1^{(k)} \\ \vdots \\ w_n^{(k)} \end{bmatrix} - \alpha \cdot \frac{1}{M} \cdot y_d^{(k)(m)} \begin{bmatrix} x_0^{(k)} \\ x_1^{(k)} \\ \vdots \\ x_n^{(k)} \end{bmatrix} \rightarrow \text{形成下一回合的參數}$$

- 經過上述流程，最終可以得到一個代表該筆資料的一條直線(二維)或是平面(三維)或是超平面(三維以上)，並可根據該迴歸模型最終得到的預測函式進行分析預測

四、使用Python工具自動進行迴歸函式尋找

問題一(簡單線性迴歸)

▼ 設定

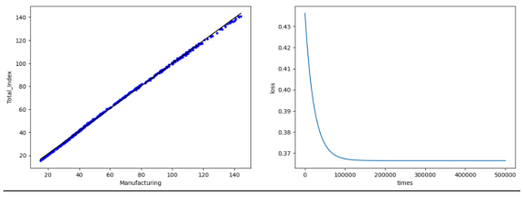
- 迭代次數 : 1000000
- 學習率 : 0.0001
- y_t : Total_Index
- y_p : 迴歸函式(經過迭代運算後求得的預測函式)

▼ 程式碼

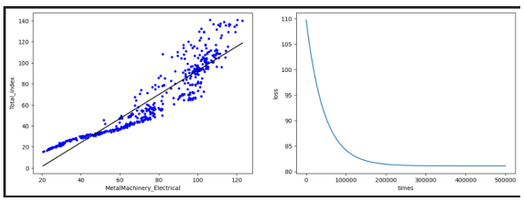
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

#預測函式
def pred(x, w):
    return np.matmul(x, w)

#讀取檔案
data = pd.read_csv('industry.csv')

#CSV的每項工業指數和總指數跑迴歸模型
for i in data:
    if i != "Total_Index" and i !=" ":
        x_data = np.array(data[[i]])
        yt = np.array(data[['Total_Index']])
        x = np.insert(x_data, 0, 1.0, axis=1)

        #取得x的行數(0)與列數(1)
        M = x.shape[0]
        D = x.shape[1]

        #設定迭代次數、學習率
        iters = 1000000
        alpha = 0.0001

        #設定初始權重和紀錄存儲
        w = np.ones(D)
        history = np.zeros((0, 2))  #第一個維度=0,第二個維度=2(有兩組數據:迭代次數、損失函數)
        for k in range(iters):
            #迭代運算，求得損失函式的低谷
            yp = pred(x, w)
            yd = yp - yt[:, 0]
            #梯度下降(權重計算)
            w = w - alpha * (x.T @ yd) / M
            if k % 10 == 0:
                # 損失值
                loss = np.mean(yd ** 2) / 2
                history = np.vstack((history, np.array([k, loss])))

        #印出損失值和迴歸函式
        print(i)
        print(loss)
        print("y={:.4f}x_0+{:.4f}x_1\n".format(w[0],w[1]))

        #劃出資料分布圖迴歸線、學習曲線
        fig_1,((ax1,ax2)) = plt.subplots(nrows = 1, ncols = 2, figsize = (15,5))
        xall = x[:, 1].ravel()
        x1 = np.array([[1, xall.min()], [1, xall.max()]])
        y1 = pred(x1, w)
        ax1.scatter(x[:, 1], yt, s=10, c='b')
        ax1.set(xlabel = i , ylabel = "Total_Index")
        ax1.plot(x1[:, 1], y1, c="k")
        ax2.plot(history[1:,0], history[1:,1])
        ax2.set(xlabel = "times" , ylabel = "loss")
```

模型成果(左圖為資料分布圖和迴歸直線，右圖為學習曲線，函式的參數為取4位小數點後的結果):

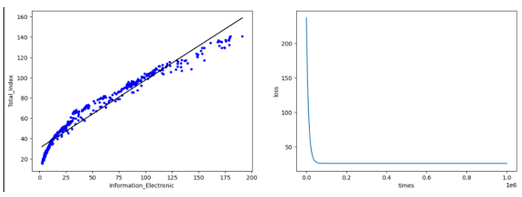
1. x_1 : **製造業**
   損失值 : 0.35245604396340396
   $y_p = 1.7102x_0 + 0.9839x_1$



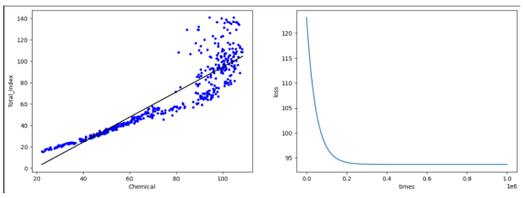
2. x_1 : **金屬機電工業**
   損失值 : 70.0198201635776
   $y_p = 29.8889x_0 + 0.6923x_1$



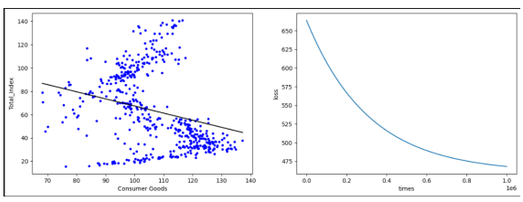
3. x_1 : **資訊電子工業**
   損失值 : 23.571850585185352
   $y_p = -5.4322x_0 + 0.9593x_1$



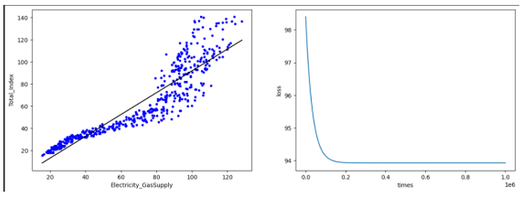
4. x_1 : **化學工業**
   損失值 : 74.53408301956193
   $y_p = -20.1785x_0 + 1.1130x_1$



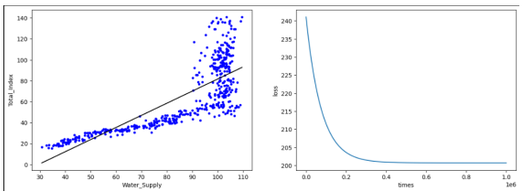
5. x_1 : **民生工業**
   損失值 : 431.7928294344629
   $y_p = 126.3862x_0 + -0.6044x_1$



6. x_1 : **電力及燃氣供應業**
   損失值 : 87.17789781652849
   $y_p = -31.9500x_0 + 1.1204x_1$



7. x_1 : **用水供應業**
   損失值 : 182.06039041384068
   $y_p = -31.9500x_0 + 1.1204x_1$



8. **函式結果:**

```
Manufacturing
0.35245604396340396
y=1.7102x_0+0.9839x_1

MetalMachinery_Electrical
70.0198201635776
y=-20.1785x_0+1.1130x_1

Information_Electronic
23.571850585185352
y=29.8889x_0+0.6923x_1

Chemical
74.53408301956193
y=-21.3826x_0+1.1438x_1

Consumer_Goods
431.7928294344629
y=126.3862x_0+-0.6044x_1

Electricity_GasSupply
87.17789781652849
y=-5.4322x_0+0.9593x_1

Water_Supply
182.06039041384068
y=-31.9500x_0+1.1204x_1
```

▼ **說明&推測:**

- 逐一將各種工業的發展情形和總指數作比對，可以觀察出台灣各工業發展對整體工業發展的影響
- 台灣的製造業指數和總指數所形成的預測函數幾乎和大部份資料點重合，且損失值只有0.3，可以推斷出台灣製造業和整體工業的發展趨勢一

  **也有可能是因為台灣有眾多製造業，所以在計算總指數時，製造業佔了很大的比重

- 資訊電子工業的損失值是第二低的，有可能是因為台灣的高科技產業也非常多，特別是在晶圓電子領域高度發展，所以對於台灣整體工業的生產有很大的影響
- 在所有的工業中，只有民生工業的發展和總指數呈現負相關，而且損失值也非常高，可能是因為民生工業並非台灣工業發展的主力，所以對總指數的影響較低

  **也有可能是因為，資源都投入到其他的工業發展上，所以導致民生工業的發展較落後

問題一(多元線性迴歸)

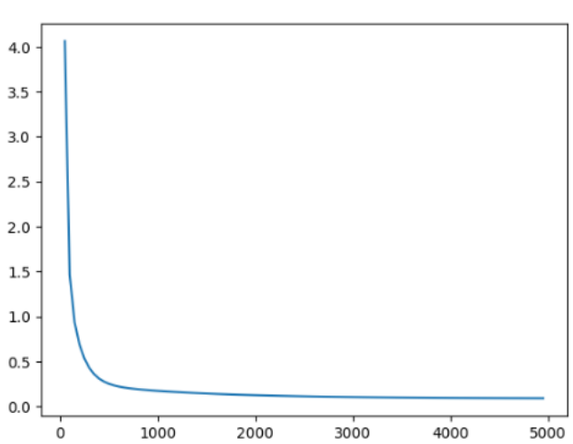
▼ 設定

- 迭代次數 : 5000
- 學習率 : 0.00001
- y_t : Total_Index
- y_p : 迴歸函式(經過迭代運算後求得的預測函式)
- x_1 : Manufacturing
- x_2 : MetalMachinery_Electrical
- x_3 : Information_Electronic
- x_4 : Chemical
- x_5 : Consumer_Goods
- x_6 : Electricity_GasSupply
- x_7 : Water_Supply

▼ 程式碼

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

def pred(x, w):
    return np.matmul(x, w)

data = pd.read_csv('industry.csv')

#將每項工業作為特徵值
x_data = np.array(data[['Manufacturing','MetalMachinery_Electrical','Information_Electronic','Chemical','Consumer_Goods','Electricity_G
yt = np.array(data[['Total_Index']])
x = np.insert(x_data, 0, 1.0, axis=1)
M = x.shape[0]
D = x.shape[1]
iters = 5000
alpha = 0.00001
w = np.ones(D)
history = np.zeros((0, 2))
for k in range(iters):
    yp = pred(x, w)
    yd = yp - yt[:, 0]
    w = w - alpha * (x.T @ yd) / M
    loss = np.mean(yd ** 2) / 2
    if k % 50 ==0:
        loss = np.mean(yd ** 2) / 2
        history = np.vstack((history, np.array([k, loss])))

print("loss:",loss)
print("y={:.4f}x_0+{:.4f}x_1+{:.4f}x_2+{:.4f}x_3+{:.4f}x_4+{:.4f}x_5+{:.4f}x_6+{:.4f}x_7".format(w[0],w[1],w[2],w[3],w[4],w[5],w[6],w[7

plt.plot(history[1:,0], history[1:,1])
plt.show()
```

- **模型成果:**
  - 損失值 : 0.09196718225902942
  - 預測函式: $y_p = 0.9902 + 0.2546x_1 + 0.1523x_2 + 0.3347x_3 + 0.1584x_4 + 0.0307x_5 + 0.0637x_6 - 0.0081x_7$



**三個以上的參數，所形成的迴歸模型會是一個超平面較難圖式呈現，所以在此僅有學習曲線

▼ **說明&推測:**

- 損失值非常低，推測應該是因為總指數是由各項工業指數計算求得
- 最後得到的預測函數，可以從權重看出台灣每項工業在台灣工業發展的占比
- 統計處經過計算得出各項工業生產指數的數據，而這裡所求的的預測函數，是從數據推出函式，推測應和統計處的計算公式相似

問題二

▼ 設定

- 迭代次數 : 5000
- 學習率 : 0.0001
- x_1 : Electricity_GasSupply
- x_2 : Water_Supply
- y_p : 迴歸函式(經過迭代運算後求得的預測函式)

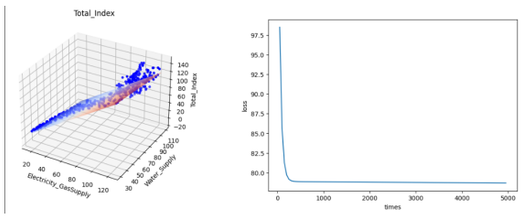
▼ 程式碼

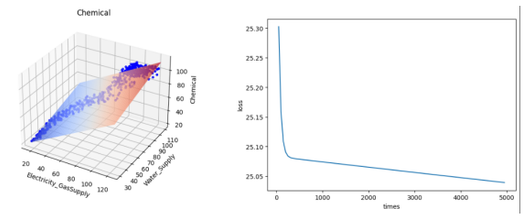
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from mpl_toolkits.mplot3d import Axes3D

def pred(x, w):
    return np.matmul(x, w)

data = pd.read_csv('industry.csv')

x_data = np.array(data[['Electricity_GasSupply', 'Water_Supply']])
for i in data:
    if i != " " and i != "Electricity_GasSupply" and i !="Water_Supply":
        yt = np.array(data[[i]])
        x = np.insert(x_data, 0, 1.0, axis=1)
        M = x.shape[0]
        D = x.shape[1]
        iters = 5000
        alpha = 0.0001
        w = np.ones(D)
        history = np.zeros((0, 2))
        for k in range(iters):
            yp = pred(x, w)
            yd = yp - yt[:, 0]
            w = w - alpha * (x.T @ yd) / M
            if k % 50 ==0:
                loss = np.mean(yd ** 2) / 2
                history = np.vstack((history, np.array([k, loss])))

        print(i)
        print(loss)
        print("y={:.4f}x_0+{:.4f}x_1+{:.4f}x_2\n".format(w[0],w[1],w[2]))

        #設定三維的資料分布及迴歸平面圖
        x1, x2 = np.meshgrid(np.linspace(x[:,1].min(), x[:,1].max(), 100),
                             np.linspace(x[:,2].min(), x[:,2].max(), 100))
        x_all = np.column_stack((np.ones(x1.size), x1.ravel(), x2.ravel()))
        y1 = pred(x_all, w)
        y1 = y1.reshape(x1.shape)
        fig = plt.figure(figsize=(16, 5))

        ax1 = fig.add_subplot(121, projection='3d')
        ax1.scatter(x[:,1], x[:,2], yt, s=10, c='b')
        ax1.set_xlabel('Electricity_GasSupply')
        ax1.set_ylabel('Water_Supply')
        ax1.set_zlabel(i)
        ax1.set_title(i)
        #設定回歸平面(這裡的alpha是透明度)
        ax1.plot_surface(x1, x2, y1, cmap='coolwarm', alpha=0.8)

        ax2= fig.add_subplot(122)
        ax2.plot(history[1:,0], history[1:,1])
        ax2.set(xlabel="times",ylabel="loss")
```

模型成果(左圖為資料分布圖以及迴歸平面，右圖為學習曲線，函式的參數為取4位小數點後的結果):

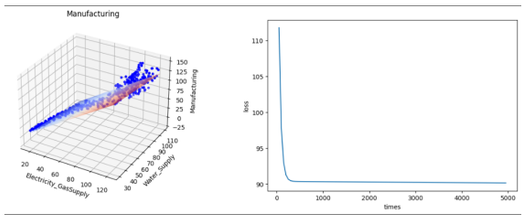
1. y_t : **總指數**
   損失值 : 78.66930964237585
   $y_p = 1.2871x_0 + 1.2325x_1 - 0.3167x_2$

5. y_t : **化學工業**
   損失值 : 25.038852002772888
   $y_p = 1.1399x_0 + 0.5878x_1 + 0.3594x_2$

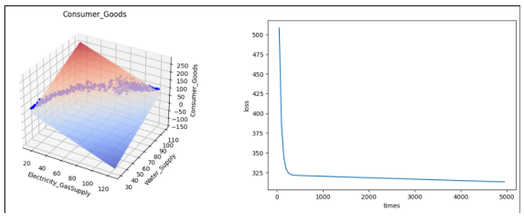




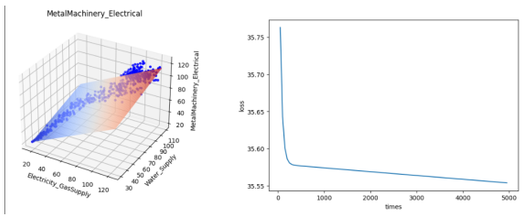
2. y_t：**製造業**
   損失值： 90.13794770690087
   $y_p = 1.3038x_0 + 1.2635x_1 - 0.3518x_2$



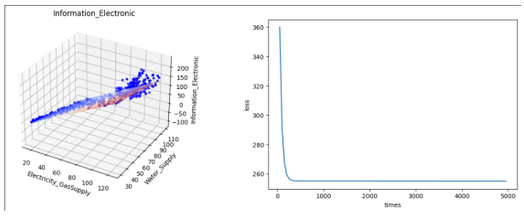
6. y_t：**民生工業**
   損失值 : 312.83830907186245
   $y_p = 3.2077x_0 - 1.7376x_1 + 2.6428x_2$



3. y_t：**金屬機電工業**
   損失值 : 35.55369558633736
   $y_p = 1.1046x_0 + 0.5863x_1 + 0.3731x_2$



4. y_t：**資訊電子工業**
   損失值 : 254.9419166449281
   $y_p = 1.2644x_0 + 2.1804x_1 - 1.2966x_2$



7. **函式結果:**

```
Total_Index
78.66930964237585
y=1.2871x_0+1.2325x_1+-0.3167x_2

Manufacturing
90.13794770690087
y=1.3038x_0+1.2635x_1+-0.3518x_2

MetalMachinery_Electrical
35.55369558633736
y=1.1046x_0+0.5863x_1+0.3731x_2

Information_Electronic
254.9419166449281
y=1.2644x_0+2.1804x_1+-1.2966x_2

Chemical
25.038852002772888
y=1.1399x_0+0.5878x_1+0.3594x_2

Consumer_Goods
312.83830907186245
y=3.2077x_0+-1.7376x_1+2.6428x_2
```

▼ **說明&推測:**

- 因為這個模型只有兩個參數，所以可以畫出三維的圖形，形成的平面是預測的結果
- 損失值越低平面的面積應該越小，不過可能是因為三維圖形顯示視角的關係，無法直接觀察出平面和損失值之間是否有關聯
- 化學工業和金屬機電工業的損失值較小，可推測和電力及然氣、用水供應業發展有較大關係

五、自製測試資料對找到的迴歸函式進行測試

資料來源 : 工業生產指數的末10筆資料(沒被訓練過的)，另存成CSV做為測試資料

問題一(簡單線性迴歸)

- 說明: 從每項工業和總指數求出的預測函數，去預測該年總工業生產指數

▼ 程式碼

```
import numpy as np
import pandas as pd

testdata = pd.read_csv('TD.csv')

#將各項工業建立成清單
xlab = []
for i in testdata:
```

```
    if i != "Total_Index" and i !=" ":
        xlab.append(i)

#將各測資輸入對應的預測函式，得出預測值
yp = []
for i in range(7):
    x_data = np.array(testdata[[xlab[i]]])
    y_col = []
    for j in x_data:
        x1 = int(j)
        y = [1.7102+0.9839*x1, -20.1785+1.1130*x1, 29.8889+0.6923*x1,
          -21.3826+1.1438*x1, 126.3862-0.6044*x1, -5.4322+0.9593*x1,
          -31.9500+1.1204*x1]
        #儲存預測出的資料
        y_col.append(round(y[i],2))
    yp.append(y_col)

#輸出成csv檔(output1.csv)
np.savetxt("output1.csv", np.array(yp).T, delimiter=",", fmt='%.2f')

#在output1.csv加上原始的總指數資料
df = pd.read_csv('output1.csv', header=None)
init = testdata["Total_Index"].tolist()
df.insert(loc=0, column='', value=init)

#在output1.csv加上表頭
header = ['raw_data'] + xlab
df.columns = header

#將csv修改結果輸出
df.to_csv('output1.csv', index=False)

#顯示前10行
df.head(10)
```

▼ **測試結果**(各項工業的預測函數推算出來的預測總指數)

raw_data : 測試集原始的總指數資料

	raw_data	Manufacturing	MetalMachinery_Electrical	Information_Electronic	Chemical	Consumer_Goods	Electricity_GasSupply	Water_Supply
0	131.57	132.57	101.14	146.89	93.00	66.55	96.25	75.61
1	134.96	136.50	103.36	148.27	99.86	65.34	102.97	80.09
2	136.01	136.50	102.25	151.04	95.28	65.34	108.72	80.09
3	136.76	136.50	100.03	153.12	89.57	60.51	117.36	90.17
4	140.81	141.42	97.80	162.12	86.13	58.09	112.56	90.17
5	132.34	133.55	92.23	153.12	80.42	64.13	102.97	85.69
6	129.71	130.60	91.12	150.35	80.42	64.74	100.09	85.69
7	129.49	130.60	94.46	150.35	74.70	62.32	93.38	78.97
8	129.49	130.60	94.46	148.27	80.42	61.11	92.42	86.81
9	108.41	108.96	69.97	128.89	70.12	75.62	84.74	84.57

▼ **結果說明&推測**

- 損失值最低的製造業所訓練出的模型和原始資料的總指數數值最接近，預測最準確
- 損失值越大的模型，預測出的數值就和實際數值相差越遠

問題一(多元線性迴歸)

- 說明: 從多元線性回歸求得的預測函數，去預測該年總工業生產指數

▼ 程式碼

```
import numpy as np
import pandas as pd

testdata = pd.read_csv('TD.csv')

#定義預測函式
def fun(x1,x2,x3,x4,x5,x6,x7):
    return 0.9902+0.2546*x1+0.1523*x2+0.3347*x3+0.1584*x4+0.0307*x5+0.0637*x6-0.0081*x7

yp = []
for i in range(10):
    rd = testdata.iloc[i].values  #讀取指定的row，並將row的資料轉成矩陣
    x1,x2,x3,x4,x5,x6,x7 = rd[2], rd[3], rd[4], rd[5], rd[6], rd[7], rd[8]
    yp.append(fun(x1,x2,x3,x4,x5,x6,x7))

np.savetxt("output2.csv", np.array(yp).T, delimiter=",", fmt='%.2f')
df = pd.read_csv('output2.csv', header=None)

init = testdata["Total_Index"].tolist()
df.insert(loc=0, column='', value=init)

header = ['raw_data','predict']
df.columns = header

df.to_csv('output2.csv', index=False)

df.head(10)
```

▼ **測試結果**

raw_data : 測試集原始的總指數資料

predict : 預測函式預測總指數的結果

	raw_data	predict
0	131.57	133.41
1	134.96	136.75
2	136.01	137.95
3	136.76	138.29
4	140.81	142.79
5	132.34	134.30
6	129.71	131.57
7	129.49	131.19
8	129.49	130.93
9	108.41	109.93

▼ **結果說明&推測**

- 將所有工業都當作特徵值訓練的模型，預測出的數值和實際數值非常接近
- 可以進一步驗證，推測該迴歸函式和統計處計算總指數的公式應該非常相似

問題二

- 說明: 從每項工業和電力及然氣供應業、用水供應業計算出來的迴歸模型，預測該項工業的發展狀況

▼ 程式碼
預測值:

```
import numpy as np
import pandas as pd

testdata = pd.read_csv('TD.csv')

#將各項工業建立成清單
xlab = []
for i in testdata:
    if i != " " and i != "Electricity_GasSupply" and i !="Water_Supply":
        xlab.append(i)

#將各測資入對應的預測函式
yp = []
for i in range(6):
    xEG = np.array(testdata[['Electricity_GasSupply']])
    xWS = np.array(testdata[['Water_Supply']])
    y_col = []
    for j in range(10):
        x1 = int(xEG[j])
        x2 = int(xWS[j])
        y = [1.2871+1.2325*x1-0.3167*x2, 1.3038+1.2635*x1-0.3518*x2,
             1.1046+0.5863*x1+0.3731*x2, 1.2644+2.1804*x1-1.2966*x2,
             1.1399+0.5878*x1+0.3594*x2, 3.2077-1.7376*x1+2.6428*x2]
        y_col.append(round(y[i],2))
    yp.append(y_col)

#輸出成csv檔(output1.csv)
np.savetxt("output3.csv", np.array(yp).T, delimiter=",", fmt='%.2f')

#在output1.csv加上原始的總指數資料
df = pd.read_csv('output3.csv', header=None)

#在output1.csv加上表頭
header = xlab
df.columns = header

#將csv修改結果輸出
df.to_csv('output3.csv', index=False)

#顯示前10行
df.head(10)
```

實際值:

```
import numpy as np
import pandas as pd

testdata = pd.read_csv('TD.csv')

#資料範圍選取
a = testdata.loc[:,'Total_Index':'Consumer_Goods']
a.head(10)
```

▼ **測試結果**

- 預測函式輸出資料:

	Total_Index	Manufacturing	MetalMachinery_Electrical	Information_Electronic	Chemical	Consumer_Goods
0	101.53	101.46	99.07	107.91	97.95	72.73
1	108.89	108.90	104.67	117.99	103.50	71.14
2	116.28	116.48	108.18	131.07	107.03	60.71
3	124.53	124.69	116.82	139.03	115.55	68.86
4	118.36	118.37	113.89	128.12	112.61	77.55
5	107.31	107.14	106.53	111.51	105.30	84.35
6	103.61	103.35	104.77	104.97	103.53	89.57
7	96.88	96.62	98.43	97.48	97.26	85.87
8	93.43	92.89	100.46	86.23	99.19	106.11
9	84.21	83.49	95.02	71.38	93.77	114.72

- 實際原始資料:

	Total_Index	Manufacturing	MetalMachinery_Electrical	Information_Electronic	Chemical	Consumer_Goods
0	131.57	133.98	109.69	169.22	100.59	99.48
1	134.96	137.03	111.49	171.67	106.56	101.52
2	136.01	137.75	110.65	175.78	102.91	101.23
3	136.76	137.84	108.01	178.31	97.54	109.01
4	140.81	142.57	106.22	191.04	94.17	113.10
5	132.34	134.18	101.85	178.90	89.63	103.69
6	129.71	131.59	100.68	174.01	89.57	102.03
7	129.49	131.89	103.39	174.49	84.52	106.83
8	129.49	131.89	103.08	171.79	89.25	108.59
9	108.41	109.72	81.97	143.18	80.99	84.94

▼ **結果說明&推測(預測函式資料和實際原始資料做比對)**

- 以電力及然氣供應業、用水供應業當作特徵值，去訓練模型預測其他工業發展的情況，所預測出來的數值和實際數值差距較大
- 資訊電子工業的損失比民生工業小，但預測出來的結果和實際數值的差異卻明顯比民生工業還要大
- 該模型預測結果和實際有所差距，可以推斷除了電力及然氣供應業、用水供應業，應該還有其他更多影響工業發展的因素

六、對於期中報告製作過程之具體收穫與心得

製作過程心得

整個製作過程最繁瑣的是查找資料集，網路上有很多開放資料，但適合分析的卻不多，實際自己操作後，才明白收集有分析價值的資料是件不容易的事。在分析的過程中，我會一直發現自己在概念上有很多的盲區，像是損失值對預測函數的影響、因變量和自變量之間的關係、每個步驟計算的意義等，從實際自己設計問題以及設計推倒模型的過程中，我對於整個迴歸分析的脈絡越來清晰，也更加理解實際的應用。

反思

- **程式設計問題**
  問題一(簡單線性迴歸)每個模型的訓練都是由同樣的程式訓練出來，訓練次數和學習率都是一樣的，我有針對模型跑出來的結果調整學習率以及迭代次數，盡可能讓所有模型的學習曲線在沒有超過低谷的前提下趨於平緩，不過民生工業的學習曲線仍無法判斷損失值是否已達最低點，問題二的模型設計也有同樣的狀況。相同的學習率和迭代次數可以讓同樣模型設計的程式自動跑完，但卻有可能會導致部分模型沒有訓練完成，是不是有可能讓模型依據不同的數據自動調整學習率以及學習次數，我覺得是我值得再研究學習的方向。
- **問題二延伸思考**
  雖然問題二的模型跑出來的預測結果和實際有所落差，但從三維的點狀圖分布，我覺得電力及然氣供應業、用水供應業這兩種產業和台灣的其餘工業發展仍有一定程度的關係，也許只從迴歸分析比較難從這些資料去推算出他們之間的關係或影響，不過我認為應該會有更好的分析方式或模型設計可以更清楚的分析出這些資料之間的關聯，設計出更好的預測模型，期待之後可以學到更多的演算法，進行進一步的分析。