Machine Learning HW1

土木五 B08501011 何子勤

Programming part:

1. 解釋什麼樣的 data preprocessing 可以 improve 你 training/testing accuracy。請提供數據(例如 kaggle public score RMSE)以佐證你的想法。

   挑選可能與 PM2.5 相關的數據，從數值組成的長條圖挑選和 PM2.5 相似的數據組，再挑出粒子類的項目，最後由繪製折線圖觀察與未來 P M2.5 相似的項目。

my_sol.csv Complete · 21s ago · with 2, 3, 6, 14	3.49722	☑
my_sol.csv Complete · 15m ago · all factor in second order	5.00581	☐

2. 請實作 2nd-order polynomial regression model (不用考慮交互項)。(1%)

   貼上 polynomial regression 版本的 Gradient descent code 內容

```
for num in range(epoch):
      for b in range(int(x.shape[0]/batch_size)):
         t+=1
         x_batch = x[b*batch_size:(b+1)*batch_size]
         y_batch = y[b*batch_size:(b+1)*batch_size].reshape(-1,1)
         # Prediction of linear regression
         pred = np.dot(np.square(x_batch), w2) + np.dot(x_batch, w1) + bias
         # loss
         loss = y_batch - pred
         # Compute gradient
         g_t1 = np.dot(x_batch.transpose(),loss) * (-2)
         g_t2 = np.dot(np.square(x_batch).transpose(),loss) * (-2)
         g_t_b = loss.sum(axis=0) * (-2)
         m_t1 = beta_1*m_t1 + (1-beta_1)*g_t1
         m_t2 = beta_1*m_t2 + (1-beta_1)*g_t2
         v_t1 = beta_2*v_t1 + (1-beta_2)*np.multiply(g_t1, g_t1)
         v_t2 = beta_2*v_t2 + (1-beta_2)*np.multiply(g_t2, g_t2)
         m_cap1 = m_t1/(1-(beta_1**t))
         m_cap2 = m_t2/(1-(beta_1**t))
         v_cap1 = v_t1/(1-(beta_2**t))
```

```
        v_cap2 = v_t2/(1-(beta_2**t))
        m_t_b = 0.9*m_t_b + (1-0.9)*g_t_b
        v_t_b = 0.99*v_t_b + (1-0.99)*(g_t_b*g_t_b)
        m_cap_b = m_t_b/(1-(0.9**t))
        v_cap_b = v_t_b/(1-(0.99**t))

        # Update weight & bias
        w1 -= ((lr*m_cap1)/(np.sqrt(v_cap1)+epsilon)).reshape(-1, 1)
        w2 -= ((lr*m_cap2)/(np.sqrt(v_cap2)+epsilon)).reshape(-1, 1)
        bias -= (lr*m_cap_b)/(math.sqrt(v_cap_b)+epsilon)
```

(b) 在只使用 NO 數值作為 feature 的情況下，紀錄該 model 所訓練出的 parameter 數值(w2, w1, b)以及 kaggle public score.

my_sol.csv 　　　**5.42307**

Complete · 8d ago · polynomial with only NO (corrected)

```
w2:
[[0.00932009]
 [0.00618544]
 [0.00431545]
 [0.00440361]
 [0.00486678]
 [0.00545708]
 [0.00825261]
 [0.0139166 ]]
w1:
[[0.13713366]
 [0.13115987]
 [0.12713296]
 [0.12231566]
 [0.12191603]
 [0.12310089]
 [0.12555388]
 [0.13238816]]
bias:
[0.36294667]
```

Math part：

Machine Learning HW1

B08501011
何子勤

1. (a)

$$\nabla_W W^TAW = \frac{(W+\Delta W)^TA(W+\Delta W)-W^TAW}{(\Delta W)^T} = \frac{\Delta W^TAW + W^TA\Delta W + (\Delta W)^TA\Delta W}{(\Delta W)^T}$$

$$\because f(W+\Delta W) - f(W) \approx (\Delta W)^T \nabla f(W)$$

$$\Rightarrow (\Delta W)^TAW + (\Delta W)^T(W^TA)^T \Rightarrow \nabla_W W^TAW = AW + (W^TA)^T = AW + A^TW$$

If A is symmetric $A^T = A \Rightarrow \nabla_W W^TAW = 2AW$

(b)

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1m} \\ & & \ddots & \\ a_{m1} & & & a_{mm} \end{bmatrix} \quad B = \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1m} \\ & & \ddots & \\ b_{m1} & & & b_{mm} \end{bmatrix} \qquad tr(AB) = \sum_{j=1}^{m}\sum_{i=1}^{m} a_{ij}b_{ji} \qquad \frac{\partial tr(AB)}{\partial a_{ij}} = b_{ji}$$

$$AB = \begin{bmatrix} (a_{11}b_{11}+a_{12}b_{21}\cdots) & & \\ (a_{21}b_{11}+a_{22}b_{21}+\cdots) & (a_{21}b_{12}+a_{22}b_{22}\cdots) & \\ & & \ddots \end{bmatrix}$$

a_{ij} 係數即 b_{ji}

(c)

$$\det(A) = \sum_{j=1}^{n}(-1)^{i+j}a_{ij}\det(A_{ij}) \qquad \frac{\partial}{\partial a_{ij}}\det(A) = (-1)^{i+j}\det(A_{ij})$$

$$\Downarrow$$

又根據 Cramer rule

$$Ax = e_i \Rightarrow x_j = e_j^Tx = e_j^TA^{-1}e_i = \frac{(-1)^{i+j}\det(A_{ij})}{\det(A)} = \frac{\partial \log(\det(A))}{\partial a_{ij}}$$

$(x = A^{-1}e_i)$

2. (a)(i)

$$L(\theta) = \prod_{i=1}^{N} P(x_i|C_j) = \prod_{i=1}^{N}[\pi_1 f_{\mu_1,\Sigma_1}(x_i)]^{\hat{y}_i}[\pi_2 f_{\mu_2,\Sigma_2}(x_i)]^{1-\hat{y}_i} \qquad \text{Assume } \hat{y}_i = \begin{cases}1 & \text{if } y_i=1\\ 0 & \text{if } y_i=2\end{cases}$$

(ii)

$$\ln L(\theta) = \sum_{i=1}^{N}\ln p(x_i|\theta) = \sum_{i=1}^{N}\hat{y}_i\ln\pi_1 f_{\mu_1,\Sigma_1}(x_i) + (1-\hat{y}_i)\ln\pi_2 f_{\mu_2,\Sigma_2}(x_i)$$

$$= \sum_{i=1}^{N}\left\{\hat{y}_i\pi_1\left[-\frac{d}{2}\ln(2\pi)-\frac{1}{2}\ln|\Sigma_1|-\frac{1}{2}(x_i-\mu_1)^T\Sigma_1^{-1}(x_i-\mu_1)\right]+(1-\hat{y}_i)\pi_2\left[\frac{d}{2}\ln(2\pi)-\frac{1}{2}\ln|\Sigma_2|-\frac{1}{2}(x_i-\mu_2)^T\Sigma_2^{-1}(x_i-\mu_1)\right]\right\}$$

$\ln L(\theta)$ depend on μ_1

$$\Longrightarrow -\frac{1}{2}\sum_{i=1}^{N}\hat{y}_i(x_i-\mu_1)^T\Sigma_1^{-1}(x_i-\mu_1) \xRightarrow[\text{\& 取}\nabla\mu_1=0]{\Sigma^{-1}\text{ is symmetric}} \sum_{i=1}^{N}\hat{y}_i\Sigma^{-1}(x_i-\mu_1) = 0 \Rightarrow \sum_{i=1}^{N}\hat{y}_ix_i = N_1\mu_1 \Rightarrow \mu_1^* = \frac{1}{N_1}\sum_{i=1}^{N}\hat{y}_ix_i$$

同理可得

$$\Rightarrow \mu_2^* = \frac{1}{N_2}\sum_{i=1}^{N}(1-\hat{y}_i)x_i$$

$\ln L(\theta)$ depend on Σ_1

取 $\nabla\Sigma_1$ & 利用 1.(b)(c) & Σ^{-1} is symmetric

$$\Longrightarrow -\frac{1}{2}\sum_{i=1}^{N}\hat{y}_i\ln|\Sigma_1| - \frac{1}{2}\sum_{i=1}^{N}\hat{y}_i(x_i-\mu_1)^T\Sigma_1^{-1}(x_i-\mu_1) \Rightarrow -\frac{N_1}{\cancel{2}}(\cancel{\Sigma_1^{-1}}) = -\frac{1}{\cancel{2}}\sum_{i=1}^{N}\hat{y}_i(x_i-\mu_1)(x_i-\mu_1)^T(\Sigma^{-1})(\cancel{\Sigma^{-1}})$$

$$\Rightarrow N_1(\Sigma_1) = \sum_{i=1}^{N}\hat{y}_i(x_i-\mu_1)(x_i-\mu_1)^T \Rightarrow \Sigma_1^* = \frac{1}{N_1}\sum_{i=1}^{N}\hat{y}_i(x_i-\mu_1^*)(x_i-\mu_1^*)^T$$

同理

$$\Rightarrow \Sigma_2^* = \frac{1}{N_2}\sum_{i=1}^{N}(1-\hat{y}_i)(x_i-\mu_2^*)(x_i-\mu_2^*)^T$$

$\ln L(\theta)$ depend on π_1

$$\Longrightarrow \begin{cases}\sum_{i=1}^{N}\hat{y}_i\ln\pi_1 + (1-\hat{y}_i)\ln\pi_2 \\ \pi_1+\pi_2 = 1\end{cases} \Rightarrow \nabla_{\pi_1}\sum_{i=1}^{N}\hat{y}_i\ln\pi_1 + (1-\hat{y}_i)\ln(1-\pi_1) = 0 \Rightarrow \frac{N_1}{\pi_1} - \frac{N_2}{1-\pi_1} = 0$$

$$\Rightarrow N_1 - N_1\pi_1 - N_2\pi_1 = 0 \Rightarrow \begin{cases}\pi_1^* = \frac{N_1}{N} \\ \pi_2^* = \frac{N_2}{N}\end{cases}$$

2 (a)(iii) $P_\theta(C_1|x) = \dfrac{P_\theta(x|C_1)P(C_1)}{P_\theta(x|C_1)P(C_1)+P_\theta(x|C_2)P(C_2)} = \dfrac{\pi_1 f_{\mu_1,\Sigma_1}(x)}{\pi_1 f_{\mu_1,\Sigma_1}(x)+\pi_2 f_{\mu_2,\Sigma_2}(x)}$ → 給定 x，求 x 為 Class 1 機率

$P_\theta(x|C_1) = f_{\mu_1,\Sigma_1}(x)$ ⇒ x 在 Class 1 的 gaussian distribution 中對應的機率

(iv) $P_\theta(C_1|x) = \dfrac{P_\theta(x|C_1)P(C_1)}{P_\theta(x|C_1)P(C_1)+P_\theta(x|C_2)P(C_2)} = \dfrac{1}{1+\frac{P_\theta(x|C_2)P(C_2)}{P_\theta(x|C_1)P(C_1)}} \xrightarrow{令 z = \ln\frac{P(x|C_1)P(C_1)}{P(x|C_2)P(C_2)}} \dfrac{1}{1+e^{(-z)}}$

$$z = \ln\frac{P(x|C_1)P(C_1)}{P(x|C_2)P(C_2)} = \ln\frac{\cancel{\frac{1}{(2\pi)^{\frac{D}{2}}}}\frac{1}{|\Sigma_1|^{\frac{1}{2}}}\exp\{-\frac{1}{2}(x-\mu_1)^T(\Sigma_1)^{-1}(x-\mu_1)\}P(C_1)}{\cancel{\frac{1}{(2\pi)^{\frac{D}{2}}}}\frac{1}{|\Sigma_2|^{\frac{1}{2}}}\exp\{-\frac{1}{2}(x-\mu_2)^T(\Sigma_2)^{-1}(x-\mu_2)\}P(C_2)} = \ln\frac{|\Sigma_2|^{\frac{1}{2}}}{|\Sigma_1|^{\frac{1}{2}}} - \frac{1}{2}[(x-\mu_1)^T(\Sigma_1)^{-1}(x-\mu_1)-(x-\mu_2)^T(\Sigma_2)^{-1}(x-\mu_2)] + \ln\frac{P(C_1)}{P(C_2)}$$

$$= \ln\frac{|\Sigma_2|^{\frac{1}{2}}}{|\Sigma_1|^{\frac{1}{2}}} - \frac{1}{2}[(x-\mu_1)^T(\Sigma_1)^{-1}(x-\mu_1)-(x-\mu_2)^T(\Sigma_2)^{-1}(x-\mu_2)] + \ln\left(\frac{\pi_1}{\pi_2}\right)$$

3. (b)(i)

similar to (a)(i) $L(\theta) = \prod_{i=1}^{N}[\pi_1 f_{\mu_1,\Sigma}(x_i)]^{\hat{y}_i}[\pi_2 f_{\mu_2,\Sigma}(x_i)]^{1-\hat{y}_i}$

(ii)

μ_1^* and μ_2^* same with (a)(ii)

$\ln L(\theta)$ depend on Σ ⟹ $-\frac{1}{2}\sum_{i=1}^{N}\hat{y}_i\ln|\Sigma| - \frac{1}{2}\sum_{i=1}^{N}\hat{y}_i(x_i-\mu_1)^T\Sigma^{-1}(x_i-\mu_1) - \frac{1}{2}\sum_{i=1}^{N}(1-\hat{y}_i)\ln|\Sigma| - \frac{1}{2}\sum_{i=1}^{N}(1-\hat{y}_i)(x_i-\mu_2)^T\Sigma^{-1}(x_i-\mu_2)$

$\Rightarrow \Sigma = \frac{N_1}{N}\Sigma_1 + \frac{N_2}{N}\Sigma_2$ (Σ_1 and Σ_2 are calculated in (a)(ii))

π_1^* and π_2^* same with (a)(ii)

(iii)

$P_\theta(C_1|x) = \dfrac{\pi_1 f_{\mu_1,\Sigma}(x)}{\pi_1 f_{\mu_1,\Sigma}(x)+\pi_2 f_{\mu_2,\Sigma}(x)}$ ⇒ 給定 x，求 x 為 Class 1 的機率

$P_\theta(x|C_1) = f_{\mu_1,\Sigma}(x)$ ⇒ x 在 Class 1 的 gaussian distribution 中對應的機率

(iv)

$\Sigma_1 = \Sigma_2 = \Sigma$

$$z = -\frac{1}{2}\left[(x-\mu_1)^T(\Sigma^{-1})(x-\mu_1) - (x-\mu_2)^T(\Sigma)^{-1}(x-\mu_2)\right] + \ln\left(\frac{\pi_1}{\pi_2}\right)$$

$$= (\mu_1-\mu_2)^T\Sigma^{-1}x - \frac{1}{2}(\mu_1)^T\Sigma^{-1}\mu_1 + \frac{1}{2}(\mu_2)^T\Sigma^{-1}\mu_2 + \ln\left(\frac{\pi_1}{\pi_2}\right)$$

reference:
Pattern recognition and machine learning
Christopher M Bishop

3.

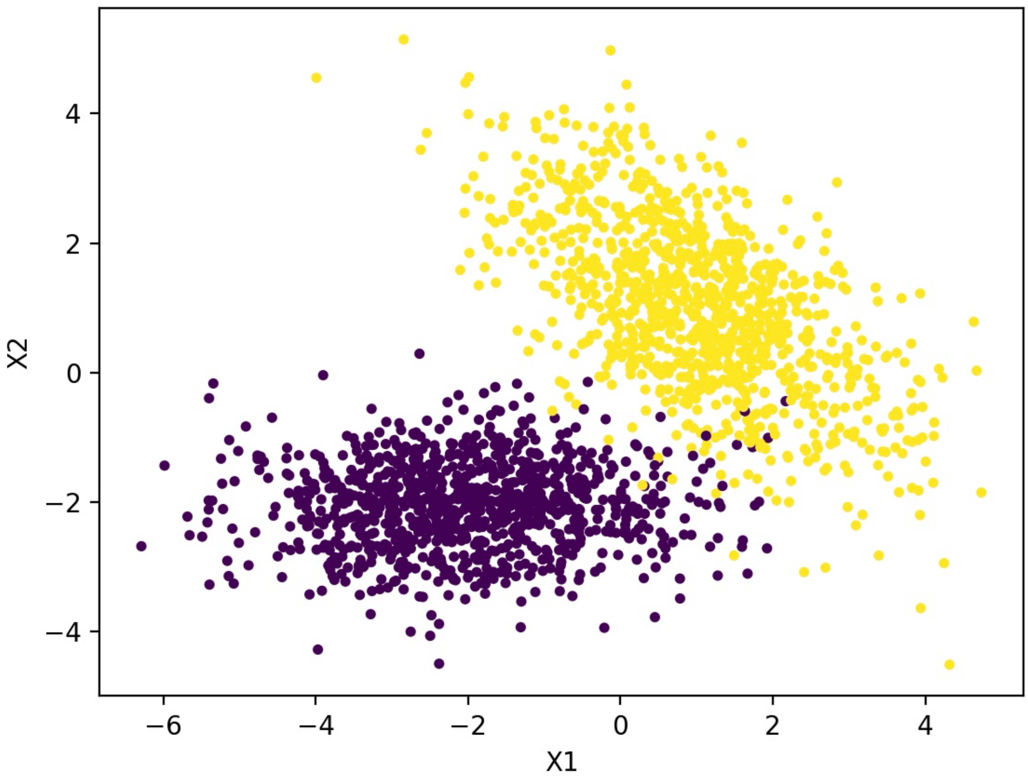
(a)

```
N = 2000 N1 = 1000 N2 = 1000
pi1 = 0.5 pi2 = 0.5
C1avg =
[[-2.02571697]
 [-2.04619501]]
C2avg =
[[1.01143637]
 [1.00493194]]
Cvar =
[[ 1.85889712 -0.51610136]
 [-0.51610136  1.14373928]]
```

(b)

```
N = 2000 N1 = 1000 N2 = 1000
pi1 = 0.5 pi2 = 0.5
C1avg =
[[-2.02571697]
 [-2.04619501]]
C2avg =
[[1.01143637]
 [1.00493194]]
C1Var =
[[2.01130388 0.03386452]
 [0.03386452 0.45977354]]
C2Var =
[[ 1.70649036 -1.06606724]
 [-1.06606724  1.82770502]]
```

(C)



我認為(b)的表現會比較好，因為 Class 1 和 Class2 的分佈走向顯示兩個種類的 covariance matrix 並不相同，Class 1 的兩變數 covariance 應接近於零，但 Class 兩變數應有一定程度相關，故兩者的 covariance matrix 不應假設相同。

Code:

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

# datashape = (2000, 3)
data = np.load("./data.npy")

# class count
N = 0
N1 = 0
N2 = 0
for i in range(2000):
    N += 1
    if data[i, 2] == 0:
        N1 += 1
    else:
        N2 += 1
print("N = " + str(N), "N1 = " + str(N1), "N2 = " + str(N2))
print("pi1 = " + str(N1 / N), "pi2 = " + str(N2 / N))

# calculate average
X1 = np.array([data[:, 0] * (1 - data[:, 2]), data[:, 1] * (1 - data[:,
2])])
X2 = np.array([data[:, 0] * data[:, 2], data[:, 1] * data[:, 2]])
C1avg = (np.sum(X1, axis = 1) / N1).reshape(2, 1)
C2avg = (np.sum(X2, axis = 1) / N2).reshape(2, 1)
print("C1avg = ")
print(C1avg)
print("C2avg = ")
print(C2avg)

# calculate variance
X1 = np.array([(data[:, 0] - C1avg[0])* (1 - data[:, 2]), (data[:, 1] -
C1avg[1]) * (1 - data[:, 2])])
X2 = np.array([(data[:, 0] - C2avg[0])* data[:, 2], (data[:, 1] - C2avg[1])
* data[:, 2]])
C1Var = np.matmul(X1, np.transpose(X1)) / N1
C2Var = np.matmul(X2, np.transpose(X2)) / N2
```

```
'''
print("C1Var = ")
print(C1Var)
print("C2Var = ")
print(C2Var)
'''
CVar = N1 / N * C1Var + N2 /  N * C2Var
print("Cvar = ")
print(CVar)
# plot data
plt.scatter(data[:, 0], data[:, 1], marker = '.', c = data[:, 2])
plt.xlabel("X1")
plt.ylabel("X2")
plt.show()
```

4. (a) $\sum_i k_i (y_i - x_i\theta)^2 + \lambda \sum_j w_j^2 = (y-x\theta)^T K (y-x\theta) + \lambda\theta^T\theta$

$= y^TKy - y^TKx\theta - \theta^Tx^TKy + \theta^Tx^TKx\theta + \lambda\theta^T\theta$

∇_θ, $\theta = \theta+\Delta\theta$ 找 $(\Delta\theta)^T$ 係數 $\Rightarrow -(y^TKx)^T - x^TKy + 2x^TKx\theta + \lambda 2\theta = 0$

$K^T=K \Rightarrow -\cancel{2}x^TKy + \cancel{2}x^TKx\theta + \lambda\cancel{2}\theta = 0 \Rightarrow (x^TKx+\lambda I)\theta = x^TKy$

$\Rightarrow \theta^* = (x^TKx+\lambda I)^{-1}x^TKy$

5.

$\tilde{L}_{ss}(w,b) = E\left[\frac{1}{2N}\sum_{i=1}^{N}(f_{w,b}(x_i+\eta_i)-y_i)^2\right]$

$= \frac{1}{2N}E\left[\sum_{i=1}^{N}(W^T(x_i+\eta_i)+b-y_i)^2\right] = \frac{1}{2N}E\left[\sum_{i=1}^{N}(W^Tx_i + W^T\eta_i + b - y_i)^T(W^Tx_i+W^T\eta_i+b-y_i)\right]$

$= \frac{1}{2N}E\left[\sum_{i=1}^{N}(f_{w,b}(x_i)-y_i)^T(f_{w,b}(x_i)-y_i) + (W^T\eta_i)^T(f_{w,b}(x_i)-y_i) + (f_{w,b}(x_i)-y_i)^T W^T\eta_i + (W^T\eta_i)^T W^T\eta_i\right]$

$E(\eta_i)=0$，中間兩項為0

$\Rightarrow = \frac{1}{2N}\sum_{i=1}^{N}E[(f_{w,b}(x_i)-y_i)^2] + \frac{1}{2N}\sum_{i=1}^{N}E(\eta_i^T W W^T\eta_i)$

$\eta_i^TW, W^T\eta_i$ 皆為 1x1 matrix

$\Rightarrow = \frac{1}{2N}\sum_{i=1}^{N}(f_{w,b}(x_i)-y_i)^2 + \frac{1}{2N}\sum_{i=1}^{N}E(W^T\eta_i)(\eta_i^TW) = \frac{1}{2N}\sum_{i=1}^{N}(f_{w,b}(x_i)-y_i)^2 + \frac{1}{2N}\sum_{i=1}^{N}E(W^T\eta_i\eta_i^TW)$

$\sum_{i=1}^{N}E(\eta_i\eta_i^T) = N\sigma^2 I$ $W^TW = \|W\|^2$

$\Rightarrow = \frac{1}{2N}\sum_{i=1}^{N}(f_{w,b}(x_i)-y_i)^2 + \frac{\cancel{N}\sigma^2}{2\cancel{N}}\|W\|^2 = \frac{1}{2N}\sum_{i=1}^{N}(f_{w,b}(x_i)-y_i)^2 + \frac{\sigma^2}{2}\|W\|^2$