Machine Learning Homework 2

1

當三點共線時，3D Perceptron 無法分類出中間的點不同於外面的兩個點，故無法被 shattered.

當四點共平面時，3D Perceptron 無法分類出兩對角點不同於另外兩對角點，故無法被 shattered.

For [a], $(2, 3, 4), (4, 3, 2), (3, 3, 3)$ 三點共線 $(2 + t, 3, 4 - t)$

For [b], $(1, 1, 1), (2, 3, 4).(4, 3, 2)$ 三點共平面 $x - 2y + z = 0$ 上, 而且 $(4, 2, 3)$ 不在上面，因此他可被 shattered.

For [c], $(1, 1, 1), (2, 3, 4).(4, 3, 2), (2, 2, 2)$ 四點共平面 $x - 2y + z = 0$.

For [d], $(2, 3, 4), (4, 3, 2), (4, 2, 3), (3, 2, 4)$ 四點共平面 $x + y + z = 9$.

答案為 [b].

2

在 origin-passing perceptrons 上，我們可以將其視為一條 $y - ax = 0$ 的直線以圓心為中心旋轉，而直線的左右兩端為(1, -1)或(-1, 1)。所以對任意點(x', y')，當 $y - ax = 0$ 旋轉後經過該點，則會將其值的判定從1轉為-1或從-1轉為1，再轉180度後再將其值恢復為原值。也就是說，隨著線的旋轉所有資料點會依角度的固定順序切換(1,-1)兩個狀態，切換前後便是兩種分類方法，而N個資料點與原點的直線不重複的情況下，最多有2N種分類方法。

答案為 [c].

3

對於算式 $\sum_{i=1}^{d} x_i^2$ 可以視為在d為空間中與原點的距離，因此若將所有點以與原點的距離對應到一維的新座標，便可轉化為 positive intervals 的形式。其 growth function 為 $C_2^{N+1} + 1 = \frac{1}{2}N^2 + \frac{1}{2}N + 1$。

答案為 [a].

4

Positive intervals 的 minimum break point 為 3 (三點依序為 +1, -1, +1 無法被正確分類)，故 VC dimension 為 2。 答案為 [d].

5

For [a], 當$N = 4$，4點不重複時16種組合皆可被正確分類。當$N = 5$，5點若有任兩點重疊，則該兩點不可能被分類為不同值。若五點不重複且由小到大為x_0, x_1, x_2, x_3, x_4，則不存在對應值為$+1, -1, +1, -1, +1$的分類法，故 $N = 5$ 為 break point，VC dimension 為 4。

For [b], 當$N = 4$，4點為$(1,0),(0,1),(-1,0),(0,-1)$ 時16種組合皆可被正確分類。當$N = 5$，設原先可被正確分類的組合為

$$(x_{min}, y_1), (x_{max}, y_2), (x_1, y_{min}), (x_2, y_{max}) \ \ with$$
$$x_{min} \le x_1, x_2, \le x_{max} \ \ , \ \ y_{min} \le y_1, y_2, \le y_{max}$$

若任三點組成的最小長方形包含其他任意點，則不存在三點為1且該任意點為-1的正確分類方式。設新加入的第五點為(x', y')，由此可得

$$(x', y') \ not \ in \ minimum \ rectangle \ of \ (x_{min}, y_1), (x_1, y_{min}), (x_2, y_{max})$$
$$\rightarrow (y' > y_{max} \lor y' < y_{min}) \lor (x' > max(x_1, x_2) \lor x' < x_{min}) --- (1)$$
$$(x', y') \ not \ in \ minimum \ rectangle \ of \ (x_{max}, y_1), (x_1, y_{min}), (x_2, y_{max})$$
$$\rightarrow (y' > y_{max} \lor y' < y_{min}) \lor (x' > x_{max} \lor x' < min(x_1, x_2)) --- (2)$$

因為滿足 $(x' > max(x_1, x_2) \land x' < min(x_1, x_2)$ 的 x' 不存在,

$$(1) \land (2) \rightarrow (y' > y_{max}) \lor (y' < y_{min}) \lor (x' > x_{max}) \lor (x' < x_{min})$$

與上面同理，任兩點組成的最小長方形也不應包含其他任意點，以 case $x' > x_{max}$ 討論,

$$(x_{max}, y_2) \ not \ in \ minimum \ rectangle \ of \ (x', y'), (x_1, y_{min})$$
$$\rightarrow y' < y_2 --- (3)$$
$$(x_{max}, y_2) \ not \ in \ minimum \ rectangle \ of \ (x', y'), (x_1, y_{max})$$
$$\rightarrow y' > y_2 --- (4)$$

滿足 $(3) \wedge (4)$ 的 y' 不存在，剩下的三個 case 也可以此類推出矛盾，故 $N = 5$ 為 break point，VC dimension 為 4。

For [c], 設被分類的資料X為

$$X = \begin{pmatrix} x_1^T \\ x_2^T \\ x_3^T \\ x_4^T \\ x_5^T \end{pmatrix} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \end{pmatrix} \qquad X^{-1} = \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ -1 & 1 & 0 & 0 & 0 \\ -1 & 0 & 1 & 0 & 0 \\ -1 & 0 & 0 & 1 & 0 \\ -1 & 0 & 0 & 0 & 1 \end{pmatrix}$$

對任意 y，我們皆可用 $w = X^{-1}y$ 來找出 w，$N = 4$ 可被 shattered。

若 $N = 6$，設 X 是大小為$6 * 5$的矩陣，由於矩陣基底只有5個，其中某 $x_i, i \in \{1, ..., 6\}$ 必定可被剩下所有 $x_j, j \neq i$ 所線性組成。設$x_6 = a_1x_1 + a_2x_2 + ... + a_5x_5$，若將$a_i < 0$ 的 x_i 對應到 $y_i = -1$、$a_i > 0$ 的 x_i 對應到 $y_i = 1$，則$y_6 = sign(x_6w) = sign(a_1x_1w + a_2x_2w + ... + a_5x_5w)$，而 $a_ix_iw > 0, i \in \{1, ..., 5\}$，不存在$y_6 = -1$ 的解。$N = 6$ 為 break point，VC dimension 為 5。

For [d], 我們可以將其視為 x_0 不一定要為 1，$w = [w_0, w_1, w_2, w_3]^T$，轉變為類似 [c] 的 $x \in R^3$ 的 perceptron 問題，因此可用同樣方法解出 $N = 5$ 為 break point，VC dimension 為 4。

只有 [c] 的 VC dimension 為 5，答案為 [c].

6

hypothesis 要足夠應付 2^N 種組合結果，$2^{10} < 1126 < 2^{11}$，也就是我可以將這 $N = 10$ 的1024種組合結果接對應到一個 Hypothesis。

答案為 [d].

7

由題目得知

$$g = argmin_{h \in H} E_{in}(h) \to E_{in}(g) \leq E_{in}(g*)$$
$$g* = argmin_{h \in H} E_{out}(h) \to E_{out} g* \leq E_{out}(g)$$
$$E_{in}(g) \leq E_{in}(g*) \leq Eout(g*) \leq E_{out}(g)$$

再由 Hoeffding bound 推出大於 $1-\delta$ 機率 $|E_{in} - E_{out} \leq \epsilon|$

$$\delta = 2M\exp(-2\epsilon^2 N)$$
$$\ln\frac{\delta}{2M} = -2\epsilon^2 N$$
$$\epsilon^2 = \frac{1}{-2N}\ln\frac{\delta}{2M} = \frac{1}{2N}\ln\frac{2M}{\delta}$$
$$\epsilon = \sqrt{\frac{1}{2N}\ln\frac{2M}{\delta}}$$

得到 $E_{out}(g) - E_{in}(g) \leq \sqrt{\frac{1}{2N}\ln\frac{2M}{\delta}}$，再加上 $E_{in}(g) \leq E_{out}(g*) \leq E_{out}(g)$，推得 $E_{out}(g) - E_{out}(g*)$ 的 upper bound 為 $\sqrt{\frac{1}{2N}\ln\frac{2M}{\delta}}$。答案為 [c].

8

已知 positive ray model 的 growth function 為 $m_H(N) = N+1$，對 P 代入$\epsilon = 0.1, \delta = 0.1$ 得到

$$4(2N+1)exp(-\frac{1}{800}N) \leq 0.1$$
$$N > 10496.26$$

最小滿足的選項為[b].

9

將 w 用 $u+v$ 代入後對 v 做偏微

$$\frac{\partial(E(u) + b_E(u)^T)v + \frac{1}{2}v^T A_E(u)v)}{\partial v} = 0$$
$$b_E(u) + A_E(u)v = 0 \to v = -(A_E^{-1}(u)b_E(u))$$

答案為 [b].

10

先求出 $\nabla E_{in}(w)$

$$\begin{aligned}\nabla E_{in}(w) &= \frac{1}{N}\sum_{n=1}^{N}\frac{1}{1+\exp(-y_n w^T x_n)}\exp(-y_n w^T x_n)(-y_n x_n)\\ &= \frac{1}{N}\sum_{n=1}^{N} h(y_n x_n)(-y_n x_n)\end{aligned}$$

再求出 $\nabla^2 E_{in}(w)$

$$\begin{aligned}\nabla^2 E_{in}(w) &= \frac{1}{N}\sum_{n=1}^{N}\frac{-\exp(y_n w^T x_n)(y_n x_n)}{(1+\exp(y_n w^T x_n))^2}(-y_n x_n)\\ &= \frac{1}{N}\sum_{n=1}^{N} h(y_n x_n)h(-y_n x_n)x_n x_n^T\end{aligned}$$

答案為 [d].

11

已知經過X經過 singular value decomposition 產生的 $U\Sigma V^T$ 具有以下性質: $UU^T = I_N$, $V^T = V^{-1}$，設 $Rank(\Sigma) = r$。

For [c],

$$XX^\dagger = U\Sigma V^T V\Sigma^\dagger U^T = U\Sigma\Sigma^\dagger U^T$$

$$\Sigma\Sigma^\dagger = \begin{pmatrix} \Sigma_{1,1}\frac{1}{\Sigma_{1,1}} & 0 & \cdots & \cdots & \cdots & 0\\ 0 & \ddots & & & & \vdots\\ \vdots & & \Sigma_{r,r}\frac{1}{\Sigma_{r,r}} & & & \vdots\\ \vdots & & & 0 & & \vdots\\ \vdots & & & & \ddots & \vdots\\ 0 & \cdots & \cdots & \cdots & \cdots & 0\end{pmatrix} = \begin{pmatrix} 1 & 0 & \cdots & \cdots & \cdots & 0\\ 0 & \ddots & & & & \vdots\\ \vdots & & 1 & & & \vdots\\ \vdots & & & 0 & & \vdots\\ \vdots & & & & \ddots & \vdots\\ 0 & \cdots & \cdots & \cdots & \cdots & 0\end{pmatrix}$$

$$\neq I_N$$

若 $XX^\dagger = U\Sigma\Sigma^\dagger U^T = I_N$，則 $\Sigma\Sigma^\dagger = U^T I_N U = I_N$，但已知 $XX^\dagger \neq I_N$，因此 [c] 敘述錯誤，答案為 [c].

12

將 logistic regression 中的機率函式 θ 替換後得到 E_{in}

$$E_{in} = \frac{1}{N}\sum_{n=1}^{N} -\ln(\frac{1}{a\sqrt{2\pi}} \exp\frac{(y_n - w^T x_n)^2}{a^2})$$

對 w 微分得到

$$\frac{\partial E_i n}{\partial \theta}\frac{\partial \theta}{\partial w} = \frac{1}{N}\sum_{n=1}^{N} -\frac{1}{\theta} * \theta * \frac{-2x_n(y_n - w^T x_n)}{a^2} = 0$$

可觀察出當 $\frac{\partial(y-w^T X)}{\partial w} = 0$ 時上述E_{in} 函式 gradient 也會是0，因此等同於 linear regression's w 的求法，也就是 $w = (X^T X)^{-1} X^T y$。

答案為 [a].

13, 14, 15, 16

Code:

```
import numpy as np
import random
import time
import math
import tqdm


#=========================Basic Function================================


def sign(arr):
  for i in range(arr.shape[0]):
    if arr[i] < 0:
      arr[i] = -1
    else:
      arr[i] = 1

  return arr

def sigmoid(arr):
  for i in range(arr.shape[0]):
    arr[i] = 1 / (1 + math.exp(-arr[i]))
  return arr

def E_sqr(x, y, w):
  y_hat = np.dot(x, w)
  return np.sum((y - y_hat)**2) / y.shape[0]

def E_01(x, y, w):
  y_hat = sign(np.dot(x, w))
  return np.sum(y != y_hat) / y.shape[0]
```

```python
#=========================Linear Regression=============================


def LinearRegression(x, y):
  x_pi = np.linalg.pinv(x)
  w_lin = np.dot(x_pi, y)
  return w_lin


#=========================Logistic Regression===========================


def gradient(x, y, w):
  err = np.zeros(x.shape[1])
  for i in range(x.shape[0]):
    err += sigmoid(-y[i] * np.dot(w.T, x[i])) * (-y[i] * x[i])
  return err / x.shape[0]

def LogisticRegression(x, y):
   lr = 0.1
   T = 500
   w = np.zeros((x.shape[1], 1))
   for i in range(T):
      err = gradient(x, y, w)
      w = w - lr * np.reshape(err, w.shape)
   return w


#=========================Generate Data=================================


def flipCoin():
  return random.choice([-1, 1])

def generateData(num):
  x = np.zeros((num, 2))
  y = np.zeros(num)

  for i in range(num):
    y[i] = flipCoin()
    if y[i] == 1:
      x[i] = np.random.multivariate_normal([2, 3], [[0.6, 0], [0, 0.6]])
    elif y[i] == -1:
      x[i] = np.random.multivariate_normal([0, 4], [[0.4, 0], [0, 0.4]])

  x = np.hstack((np.ones((num, 1)), x))

  return x, y

def outlierData(num):
  x = np.zeros((num, 2))
  y = np.ones(num)
  for i in range(num):
    x[i] = np.random.multivariate_normal([6, 0], [[0.3, 0], [0, 0.1]])

  x = np.hstack((np.ones((num, 1)), x))

  return x, y

#=======================================================================
```

```python
def main():
  T = 100
  err_sqr = 0
  err_01_train_lin = 0
  err_01_test_lin = 0
  err_01_test_log = 0
  err_01_out_lin = 0
  err_01_out_log = 0

  p13, p14, p15, p16 = True, True, True, True

  for i in tqdm.trange(T):
    random.seed(time.time())
    x_Train, y_Train = generateData(200)
    x_Test, y_Test = generateData(5000)
    if p13 or p14 or p15:
      w_lin = LinearRegression(x_Train, y_Train)

      if p13:
        err_sqr += E_sqr(x_Test, y_Test, w_lin)

      if p14 or p15:
        err_01_train_lin += E_01(x_Train, y_Train, w_lin)
        err_01_test_lin  += E_01(x_Test, y_Test, w_lin)

    if p15:
      w_log = LogisticRegression(x_Train, y_Train)
      w_log = np.reshape(w_log, w_lin.shape)
      err_01_test_log  += E_01(x_Test, y_Test, w_log)

    if p16:
      x_out, y_out = outlierData(20)
      x_Train = np.vstack((x_Train, x_out))
      y_Train = np.concatenate((y_Train, y_out))

      w_lin = LinearRegression(x_Train, y_Train)
      w_log = LogisticRegression(x_Train, y_Train)
      w_log = np.reshape(w_log, w_lin.shape)
      err_01_out_lin += E_01(x_Test, y_Test, w_lin)
      err_01_out_log += E_01(x_Test, y_Test, w_log)

  if p13:
    print("Problem 13:", err_sqr / T)

  if p14:
    print("Problem 14:", abs(err_01_train_lin / T - err_01_test_lin / T))

  if p15:
    print("Problem 15:", err_01_test_lin / T, err_01_test_log / T)

  if p16:
    print("Problem 16:", err_01_out_lin / T, err_01_out_log / T)

  return

if __name__ == '__main__':
  main()
```

答案依序為 [d], [e], [b], [c].