HTML HW3 r09922a21 郭蕙綺

1. $\mathbb{E}_D[E_{in}(W_{lin})] = 0.1^2 \cdot (1-\frac{19+1}{N}) \geq 0.005$

$\Rightarrow \quad 1-\frac{20}{N} \geq \frac{1}{2} \quad \Rightarrow \quad N \geq 40$ #

2. $E_{out}(g^*) = \underset{x\sim p}{\mathcal{E}} (\tilde{y}-y)^2 = \underset{x\sim p}{\mathcal{E}} (W_0+W_1X-X^2)^2$

Find minimum $E_{out}(g^*)$. Derive $\nabla_w \underset{x\sim p}{\mathcal{E}} (W_0+W_1X-X^2)^2 = 0$

$\frac{\partial}{\partial w} \mathcal{E}(W_0+W_1X-X^2)^2 = \mathcal{E}[2\cdot(W_0+W_1X-X^2)\cdot X] = 0$

$\Rightarrow W_0 + W_1\mathcal{E}[X] - \mathcal{E}[X^2] = 0$, For uniform distribution, $E[X] = \frac{1}{2}$, $Var[X] = \frac{1}{12} = E[X^2] - E^2[X]$

$E[X^2] = \frac{1}{12} + (\frac{1}{2})^2 = \frac{1}{3}$

$\Rightarrow W_0 + \frac{1}{2}W_1 - \frac{1}{3} = 0 \quad \Rightarrow \quad W_0 + \frac{W_1}{2} = \frac{1}{3} \quad \Rightarrow$ Choose (C) or (E).

when $X=0$, square error of (C) $= -\frac{1}{6}$, square error of (E) $= \frac{1}{3}$, Choose (C) #

3. $D = \{(X_1, X_1^2), (X_2, X_2^2)\}$, $h(x) = W_0 + W_1X$

代入, $\begin{cases} X_1^2 = W_0 + W_1X_1 \\ X_2^2 = W_0 + W_1X_2 \end{cases}$ $\quad W_1 = \frac{X_2^2 - X_1^2}{X_2 - X_1} = X_2 + X_1$, $W_0 = -X_1X_2$

$\mathbb{E}_D(|E_{in}(g) - E_{out}(g)|) = \mathbb{E}_D(|E_{out}(g)|) = \mathbb{E}_D \underset{(X,y)\sim p}{\mathcal{E}} [(W_1X+W_0) - X^2]^2$

$= \mathbb{E}_D \underset{(X,y)\sim p}{\mathcal{E}} [W_1^2X^2 + 2W_0W_1X + W_0^2 - 2(W_1X+W_0)\cdot X^2 + X^4]$, $E(X) = \frac{1}{2}$, $E(X^2) = \frac{1}{3}$, $E(X^3) = \frac{1}{4}$

$= \underset{(X,y)\sim p}{\mathcal{E}} (\frac{1}{3}W_1^2 + W_0W_1 + W_0^2 - \frac{1}{2}W_1 - \frac{2}{3}W_0 + \frac{1}{5})$ $\quad E(X^4) = \frac{1}{5}$

$\Rightarrow \int_0^1\int_0^1 (\frac{1}{3}X_1^2 + \frac{1}{3}X_2^2 + \frac{2}{3}X_1X_2 - X_1^2X_2 - X_1X_2^2 + X_1^2X_2^2 - \frac{1}{2}X_1 - \frac{1}{2}X_2 + \frac{2}{3}X_1X_2 + \frac{1}{5})\,dx_1dx_2$

$= \int_0^1 (\frac{1}{9}X_1^3 + \frac{1}{3}X_1X_2^2 + \frac{1}{3}X_1^2X_2 - \frac{1}{3}X_1^3X_2 - \frac{1}{2}X_1^2X_2^2 + \frac{1}{3}X_1^3X_2^2 - \frac{1}{4}X_1^2 - \frac{1}{2}X_1X_2 + \frac{1}{3}X_1^2X_2 + \frac{1}{5}X_1)\Big|_0^1 dx_2$

$= \int_0^1 (\frac{1}{9} + \frac{1}{3}X_2^2 + \frac{1}{3}X_2 - \frac{1}{3}X_2 - \frac{1}{2}X_2^2 + \frac{1}{3}X_2^2 - \frac{1}{4} - \frac{1}{2}X_2 + \frac{1}{3}X_2 + \frac{1}{5})\,dX_2$

$= \int_0^1 (\frac{1}{6}X_2^2 - \frac{1}{6}X_2 + \frac{11}{180})\,dX_2 = \frac{1}{18}X_2^3 - \frac{1}{12}X_2^2 + \frac{11}{180}X_2\Big|_0^1 = \frac{1}{18} - \frac{1}{12} + \frac{11}{180} = \frac{6}{180} = \frac{1}{30}$ #

4. $E_{in}(w) = \frac{1}{N}\sum_{n=1}^{N} -\ln\theta(y_n W^T X_n)$, $\begin{cases} \text{when } y_n = 1 \Rightarrow \theta(y_n W^T X_n) = \theta(W^T X_n) \\ \text{when } y_n = -1 \Rightarrow \theta(y_n W^T X_n) = \theta(-W^T X_n) \end{cases}$

When $y_n' = 1$ $(y_n = 1)$ need $\theta(W^T X_n)$

When $y_n' = 0$ $(y_n = -1)$ need $\theta(-W^T X_n)$.

So, the correct answer would be: $\frac{1}{N}\sum_{n=1}^{N} -(y_n' \ln\theta(W^T X_n) + (1-y_n')\ln(\theta(-W^T X_n)))$

minimize $\frac{1}{N}\sum_{n=1}^{N} -(y_n' \ln\theta(W^T X_n) + (1-y_n')\ln(\theta(-W^T X_n)))$

equivalent to maximize $\frac{1}{N}\sum_{n=1}^{N} (y_n' \ln\theta(W^T X_n) + (1-y_n')\ln(\theta(-W^T X_n)))$

Choose A.

5. ① $P(|M-V| > \epsilon) \le 2e^{-2\epsilon^2 N}$

$\Rightarrow P(|M-V| \le \epsilon) \ge 1 - 2e^{-2\epsilon^2 N}$

$\because$ prob more than $1-\delta$ $\Rightarrow 1 - 2e^{-2\epsilon^2 N} \le 1-\delta$

$\Rightarrow 2e^{-2\epsilon^2 N} \ge \delta \Rightarrow -2\epsilon^2 N \ge \ln\frac{\delta}{2} \Rightarrow \epsilon \le \sqrt{\frac{1}{2N}\ln\frac{2}{\delta}}$

$M - V \le \sqrt{\frac{1}{2N}\ln\frac{2}{\delta}} \Rightarrow M \le V + \sqrt{\frac{1}{2N}\ln\frac{2}{\delta}}$

② $\max_{\hat{M}} \text{likelihood}(\hat{M}) = \max_{\hat{M}} \prod_{n=1}^{N} P(y_n|\hat{M})$

$= \max_{\hat{M}} \prod_{n=1}^{N} (y_n \cdot \hat{M} + (1-y_n)\cdot(1-\hat{M}))$

$= \min_{\hat{M}} \sum_{n=1}^{N} -\ln[y_n \cdot \hat{M} + (1-y_n)\cdot(1-\hat{M})]$

To find minimum, derive $\nabla \sum_{n=1}^{N} -\ln[y_n \cdot \hat{M} + (1-y_n)\cdot(1-\hat{M})] = 0$

$\frac{\partial}{\partial \hat{M}} \sum_{n=1}^{N} -\ln[y_n \cdot \hat{M} + (1-y_n)(1-\hat{M})] = -\sum_{n=1}^{N} \frac{2y_n - 1}{y_n\hat{M} + (1-y_n)(1-\hat{M})} = -\left(\sum_{y_n=1}\frac{1}{\hat{M}} + \sum_{y_n=0}\frac{-1}{1-\hat{M}}\right) = 0$

$\Rightarrow \sum_{y_n=1}\frac{1}{\hat{M}} = \sum_{y_n=0}\frac{1}{1-\hat{M}}$, 又 $V = \frac{1}{N}\sum_{n=1}^{N} y_n$,

$\Rightarrow \sum_{y_n=1}\frac{1}{\hat{M}} = N\cdot V\cdot\frac{1}{\hat{M}}$, $\sum_{y_n=0}\frac{1}{1-\hat{M}} = N(1-V)\frac{1}{1-\hat{M}}$

$\Rightarrow \cancel{N}\cdot V\cdot\frac{1}{\hat{M}} = \cancel{N}\cdot(1-V)\cdot\frac{1}{1-\hat{M}} \Rightarrow \frac{1-\hat{M}}{\hat{M}} = \frac{1-V}{V} \Rightarrow V = \hat{M}$ #

HTML HW3

5. (3) To minimize, derive $\nabla E^{sqr}(\hat{y}) = 0$

$$\frac{\partial}{\partial \hat{y}} \frac{1}{N}\sum_{n=1}^{N}(\hat{y}-y_n)^2 = \frac{1}{N}\sum_{n=1}^{N} 2\cdot(\hat{y}-y_n)\cdot 1 = \frac{1}{N}\left[2\cdot N\cdot\hat{y} - 2\sum_{n=1}^{N} y_n\right] = 2\hat{y} - \frac{2}{N}\sum_{n=1}^{N} y_n = 0$$

$$\Rightarrow 2\hat{y} = \frac{2}{N}\sum_{n=1}^{N} y_n \Rightarrow \hat{y} = \frac{1}{N}\sum_{n=1}^{N} y_n \ \#$$

(4) $$\frac{\partial}{\partial \hat{y}} \frac{1}{N}\sum_{n=1}^{N}(y_n \ln\hat{y} + (1-y_n)\ln(1-\hat{y}))$$

$$= \frac{1}{N}\sum_{n=1}^{N} y_n \frac{1}{\hat{y}} + (1-y_n)\frac{-1}{1-\hat{y}} = \frac{1}{N}\left[\frac{1}{\hat{y}}\sum_{n=1}^{N} y_n + \frac{-1}{1-\hat{y}}\sum_{n=1}^{N}(1-y_n)\right]$$

$$= \frac{1}{N}\left[\frac{1}{\hat{y}}\sum_{n=1}^{N} y_n + \frac{-1}{1-\hat{y}}\left[N - \sum_{n=1}^{N} y_n\right]\right] = \frac{1}{N}\left[\frac{-N}{1-\hat{y}} + \frac{1}{\hat{y}}\sum_{n=1}^{N} y_n - \frac{-1}{1-\hat{y}}\sum_{n=1}^{N} y_n\right]$$

$$= \frac{1}{N}\left[\frac{-N}{1-\hat{y}} + \frac{1}{\hat{y}} N\cdot v - \frac{-1}{1-\hat{y}} N\cdot v\right] = 0$$

$$\Rightarrow \frac{1}{\hat{y}-1} + \frac{1}{\hat{y}} - \frac{v}{\hat{y}-1} = 0 \Rightarrow \frac{v}{\hat{y}} = \frac{v-1}{\hat{y}-1} \Rightarrow v\hat{y} - v = \hat{y}v - \hat{y} \Rightarrow v = \hat{y} \ \#$$

Choose (e)

ML HW3

6. In PLA, we update $W_{t+1}$ when $sign(W_t X_{n(t)}) \neq y_{n(t)}$, the error function is $(y_n X_n)$.

Look at stochastic gradient descent, the error function is $(-yW^T X)$. since we want to update $W_{t+1}$ when $sign(W_t X_{n(t)}) \neq y_{n(t)}$. which means that $y \cdot W^T X$ is negtive.

As a result, we need to add "−" in front of $yW^T X$.

We don't want to update $W_{t+1}$ when $sign(W_t X_{n(t)}) = y_{n(t)}$, which means that we want the error function equal to 0.

Combine the conditions, the error function is $\max(0, -yW^T X)$ #

7. $V \approx -\nabla E_{in}(W_t)$

$$\nabla E_{in}(W_t) = \frac{\partial\, err(W, x, y)}{\partial W_{i=y}} = \frac{\partial}{\partial W_{i=y}} - \ln \frac{\exp(W_y^T x)}{\sum_{i=1}^{k} \exp(W_i^T x)} = \frac{\partial}{\partial W_{i=y}} \left( \ln \sum_{i=1}^{k} \exp(W_i^T x) - \ln \exp(W_y^T x) \right)$$

$$= \frac{1}{\sum_{i=1}^{k} \exp(W_i^T x)} \cdot \frac{\partial}{\partial W_{i=y}} \sum_{i=1}^{k} \exp(W_i^T x) - \frac{\partial}{\partial W_{i=y}} W_y^T X$$

$$= \frac{1}{\sum_{i=1}^{k} \exp(W_i^T x)} \cdot \exp(W_y^T x) \cdot X - [\![ y = k ]\!] \cdot X$$

$$= \left( h_y(x) - [\![ y = k ]\!] \right) \cdot X$$

Example: $(X_n, y_n)$, ∴ $y_n$-th column of $V$ = $y_n$-th column of $-(h_y(x) - [\![ y = k ]\!]) \cdot X$ equal to $(1 - h_y(x_n)) \cdot X_n$ #

8. $\Phi_2(X_1) = (1, 0, 1, 0, 0, 1)$

$\Phi_2(X_2) = (1, 0, -1, 1, 0, 1)$

$\Phi_2(X_3) = (1, -1, 0, 1, 0, 0)$

$\Phi_2(X_4) = (1, 1, 0, 1, 0, 0)$

For (a), $\tilde{W} \cdot \Phi_2(X_1) = 0$ ⇒ label $= 1 \neq y_1$

(b), $\tilde{w} \cdot \Phi_2(X_1) = -1$

$\tilde{w} \cdot \Phi_2(X_2) = 1$ ⇒ label $= 1 \neq y_2$

(c) $\tilde{w} \cdot \Phi_2(X_1) = 0$ ⇒ label $= 1 \neq y_1$

(d) $\tilde{w} \cdot \Phi_2(X_1) = 0$ ⇒ label $= 1 \neq y_1$

(e) $\tilde{w} \cdot \Phi_2(X_1) = -1$ $\tilde{w} \cdot \Phi_2(X_3) = 0$

$\tilde{w} \cdot \Phi_2(X_2) = -1$ $\tilde{w} \cdot \Phi_2(X_4) = 0$ ⇒ all correct #

9. $W_{lin} = (X^T X)^{-1} X^T y$

After transform : $Transform\_X = (\Gamma \cdot X^T)^T = X \cdot \Gamma^T$

$\tilde{w} = [(X\Gamma^T)^T \cdot X\Gamma^T]^{-1} \cdot (X \cdot \Gamma^T)^T \cdot y$

$= (\Gamma \cdot X^T \cdot X \cdot \Gamma^T)^{-1} \cdot \Gamma \cdot X^T \cdot y$

$= (\Gamma^T)^{-1} \cdot (X^T X)^{-1} \cdot \Gamma^{-1} \cdot \Gamma \cdot X^T \cdot y$

$= (\Gamma^T)^{-1} \cdot (X^T X)^{-1} X^T y$

$W_{lin} = \Gamma^T \tilde{w} = (X^T X)^{-1} X^T y$ #

10. $\Phi(X_1) = [1, 0, 0, \ldots 0)$

$\Phi(X_2) = [0, 1, \cdots, 0)$ $\Rightarrow X = I_n$ ,

$\Phi(X_n) = [0, 0, \ldots, 1]$

$\tilde{w} = (X^T X)^{-1} X^T \cdot y$

$= (I_n^T \cdot I_n)^{-1} \cdot I_n^T \cdot y$

$= I_n \cdot I_n \cdot y = y$ #

11. $E_{in}^{0/1}(g) = \frac{1}{N} \sum_{n=1}^{N} [\![ \underset{k \in y}{\operatorname{argmax}} (W_{[k]}^T X_n) \neq y_n ]\!]$

$e_k = (W_{[k]}^T X_n y_n - 1)^2$

$E_{in}^{sqr} = \frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{K} e_k = \frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{K} [(W_{[k]}^T X_n)^2 \underbrace{- 2 W_{[k]}^T X_n y_n}_{\because \text{ error}, \therefore \downarrow = +2|W_{[k]}^T X_n|} + 1] = \frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{K} [(W_{[k]}^T X_n)^2 + 2|W_{[k]}^T X_n| + 1]$

Assume that we have "e" errors, $E_{in}^{0/1}(g) = \frac{e}{N}$

$E_{in}^{sqr} = \frac{1}{N} \sum_{n=1}^{N} \sum_{k=1}^{K} (W_{[k]}^T X_n y_n - 1)^2$, since we want to find the tighrest upper bound, we

assume that $\begin{cases} W_{[k]}^T X_n = 1 & \text{if } y_n = k \\ W_{[k]}^T X_n = -1 & \text{if } y_n \neq k \end{cases}$ $\Rightarrow E_{in}^{sqr} \geq \frac{1}{N} \cdot \sum_{n=1, n \in set(e)}^{N} \sum_{k=1}^{K} [(W_{[k]}^T X_n)^2 + 2|W_{[k]}^T X_n| + 1]$

$\geq \frac{1}{N} \cdot e \cdot K$ ( $(W_k^T X_n)^2$, $2|W_k^T X_n|$ are $> 0$)

$E_{in}^{0/1}(g) = \frac{e}{N} \leq \frac{1}{K} \sum_{k=1}^{K} e_k$ #

```python
#Q12

import numpy as np
import random
f = open('hw3_train.dat.txt')
lines = f.readlines()
train_x = []
train_y = []
for line in lines:
    train_x.append(list(map(float, line.split('\t')[0:10])))
    train_y.append(float(line.split('\t')[-1]))

new_train_x = []
for x in train_x:
    new_x = [1] + x + [xi**2 for xi in x ]
    new_train_x.append(new_x)
X = np.array(new_train_x)
y = np.array(train_y)
W = np.dot(np.dot(np.linalg.inv(np.dot(X.transpose(), X)), X.transpose()), y)
f = open('hw3_test.dat.txt')
lines = f.readlines()
test_x = []
test_y = []
for line in lines:
    test_x.append(list(map(float, line.split('\t')[0:10])))
    test_y.append(float(line.split('\t')[-1]))
new_test_x = []
for x in test_x:
    new_x = [1] + x + [xi**2 for xi in x ]
    new_test_x.append(new_x)
pred_y = []
for x in new_test_x:
    if np.dot(W, np.array(x))>0:
        pred_y.append(1.0)
    else:
        pred_y.append(-1.0)

pred_y_train = []
for x in new_train_x:
    if np.dot(W, np.array(x))>0:
        pred_y_train.append(1.0)
    else:
        pred_y_train.append(-1.0)

acc_out = 0
acc_in = 0

for i in range(len(pred_y)):
    if pred_y[i] == test_y[i]:
        acc_out += 1

for i in range(len(train_y)):
    if pred_y_train[i] == train_y[i]:
        acc_in += 1
Eout = acc_out/len(pred_y)
Ein = acc_in/len(train_y)

print(Ein-Eout)
```

```python
#Q13
new_train_x = []
for x in train_x:
    new_x = [1] + x
    for i in range(2,9):
        new_x += [xi**i for xi in x ]
    new_train_x.append(new_x)


X = np.array(new_train_x)
y = np.array(train_y)

W = np.dot(np.dot(np.linalg.inv(np.dot(X.transpose(), X)), X.transpose()), y)

new_test_x = []
for x in test_x:
    new_x = [1] + x
    for i in range(2,9):
        new_x += [xi**i for xi in x ]
    new_test_x.append(new_x)

pred_y = []
for x in new_test_x:
    ans=1.0 if np.dot(W, np.array(x))>0 else -1.0
    pred_y.append(ans)

pred_y_train = []
for x in new_train_x:
    ans=1.0 if np.dot(W, np.array(x))>0 else -1.0
    pred_y_train.append(ans)

acc_out = 0
acc_in = 0

for i in range(len(pred_y)):
    if pred_y[i] == test_y[i]:
        acc_out += 1

for i in range(len(train_y)):
    if pred_y_train[i] == train_y[i]:
        acc_in += 1
Eout = acc_out/len(pred_y)
Ein = acc_in/len(train_y)

print(Ein-Eout)
```

```python
#Q14
new_train_x = []
for x in train_x:
    new_x = [1] + x
    for i in range(10):
        for j in range(i+1, 10):
            new_x += [x[i]*x[j]]
    new_x += [xi**2 for xi in x ]
    new_train_x.append(new_x)


X = np.array(new_train_x)
y = np.array(train_y)
print(X.shape)

W = np.dot(np.dot(np.linalg.inv(np.dot(X.transpose(), X)), X.transpose()), y)

new_test_x = []
for x in test_x:
    new_x = [1] + x
    for i in range(10):
        for j in range(i+1, 10):
            new_x += [x[i]*x[j]]
    new_x += [xi**2 for xi in x ]
    new_test_x.append(new_x)

pred_y = []
for x in new_test_x:
    ans=1.0 if np.dot(W, np.array(x))>0 else -1.0
    pred_y.append(ans)

pred_y_train = []
for x in new_train_x:
    ans=1.0 if np.dot(W, np.array(x))>0 else -1.0
    pred_y_train.append(ans)

acc_out = 0
acc_in = 0

for i in range(len(pred_y)):
    if pred_y[i] == test_y[i]:
        acc_out += 1

for i in range(len(train_y)):
    if pred_y_train[i] == train_y[i]:
        acc_in += 1
Eout = acc_out/len(pred_y)
Ein = acc_in/len(train_y)

print(Ein-Eout)
```

```python
#Q15
new_train_xs = []
train_x0 = []
for i in range(len(train_x)):
    train_x0.append([1]+train_x[i])
for i in range(10):
    new_train_xs.append(np.array(train_x0)[:,:i+2])

y = np.array(train_y)

W = []
for x in new_train_xs:
    W.append(np.dot(np.dot(np.linalg.inv(np.dot(np.array(x).transpose(), np.array(x))),
np.array(x).transpose()), y))

new_test_xs = []
test_x0 = []
for i in range(len(test_x)):
    test_x0.append([1]+test_x[i])
for i in range(10):
    new_test_xs.append(np.array(test_x0)[:,:i+2])

pred_ys = []
for i, new_test_x in enumerate(new_test_xs):
    pred_y = []
    for x in new_test_x:
        ans=1.0 if np.dot(W[i], np.array(x))>0 else -1.0
        pred_y.append(ans)
    pred_ys.append(pred_y)

pred_ys_train = []
for i, new_train_x in enumerate(new_train_xs):
    pred_y_train = []
    for x in new_train_x:
        ans=1.0 if np.dot(W[i], np.array(x))>0 else -1.0
        pred_y_train.append(ans)
    pred_ys_train.append(pred_y_train)

acc_outs = []
E = []
for i in range(10):
    acc_out = 0
    for j in range(len(pred_ys[i])):
        if pred_ys[i][j] == test_y[j]:
            acc_out += 1
    acc_outs.append(acc_out/len(test_y))

for i in range(10):
    acc_in = 0
    for j in range(len(pred_ys_train[i])):
        if pred_ys_train[i][j] == train_y[j]:
            acc_in += 1
    E.append(abs(acc_in/len(train_y) - acc_outs[i]))
print(np.argmin(E))
```

```python
#Q16
import random
E = []
for i in range(200):
    randomlist = random.sample(range(0, 10), 5)
    new_train_x = []
    for x in train_x:
        new_x = [1]
        for ran in randomlist:
            new_x += [x[ran]]
        new_train_x.append(new_x)

    X = np.array(new_train_x)
    y = np.array(train_y)

    W = np.dot(np.dot(np.linalg.inv(np.dot(X.transpose(), X)), X.transpose()), y)

    new_test_x = []
    for x in test_x:
        new_x = [1]
        for ran in randomlist:
            new_x += [x[ran]]
        new_test_x.append(new_x)

    pred_y = []
    for x in new_test_x:
        ans=1.0 if np.dot(W, np.array(x))>0 else -1.0
        pred_y.append(ans)

    pred_y_train = []
    for x in new_train_x:
        ans=1.0 if np.dot(W, np.array(x))>0 else -1.0
        pred_y_train.append(ans)

    acc_out = 0
    acc_in = 0

    for i in range(len(pred_y)):
        if pred_y[i] == test_y[i]:
            acc_out += 1

    for i in range(len(train_y)):
        if pred_y_train[i] == train_y[i]:
            acc_in += 1

    Eout = acc_out/len(pred_y)
    Ein = acc_in/len(train_y)
    E.append(abs(Ein-Eout))

print(sum(E)/200)
```