1. b

$0.01\left(1-\frac{12}{N}\right) \geq 0.006$

$0.4 \geq \frac{12}{N}$

$N \geq 30$

2. C

由講義第9章第10頁得知.

invertible X^TX. → unique solution → $\nabla E_{in}(w)=0$

3. C

(a) rank不变

(b) rank不变

(c) rank会改变.

(d) rank不变

(e) rank不变.

4. e

根据 hoeffding's inequality $P[|\nu-u|>\varepsilon] \leq 2\exp(-2\varepsilon^2 N)$

$\frac{1}{N}\Sigma(\hat{y}-y_n)^2$

$\frac{\partial}{\partial \hat{y}} E_{in}(\hat{y}) = \frac{1}{N}\Sigma 2(\hat{y}-y_n)$

$\hat{y}=0 \qquad \nu=\frac{1}{N}\Sigma y_n$

$\nabla E_{in}(\hat{y}) = \underbrace{\frac{1}{N}\Sigma(-2)\cdot y_n}_{-2\nu}$

$-\nabla E_{in}(\hat{y}) = 2\nu$

掃描全能王 創建

5. c

likelihood $(h) = P(x_1)h(x_1) \times P(x_2)h(x_2) \cdots P(x_N)h(x_N)$

∵ uniform distribution

∴ $P(x) = \frac{1}{\theta}$

likelihood $= \frac{1}{\theta} \cdot y_1 \times \frac{1}{\theta} \cdot y_2 \cdots$

$= \prod_{n=1}^{N} \frac{y_n}{\theta}$

6. b

$-\nabla E_{in} = \frac{1}{N} \Sigma y x$

$E_{in} = \frac{1}{N} \Sigma (-y w^T x)$

$\because y_n = -1, \ w^T x = 1.$

$w^T x \neq y_n.$

$err(w, x, y) = \max(0, -y w^T x)$

$= \max(0, 1)$

$= 1$

7. a

$\frac{\partial}{\partial w} err_{exp}(w, x, y) = \frac{\partial}{\partial w} \exp(-y w^T x)$

$\nabla err_{exp}(w, x, y) = -yx \exp(-y w^T x)$

$-\nabla err_{exp}(w, x, y) = yx \exp(-y w^T x)$

掃描全能王 創建

8. b.

$$E(w) \approx E(u) + bE(u)^T(v) + \frac{1}{2}(v)^T A_E(w)(v) \qquad w \leftarrow u+v.$$

$$E'(v) \approx bE(u)^T + A_E(w)\cdot v = 0.$$

$$v = -bE(u)^T \cdot A_E(u)^{-1}$$

$$= -A_E(u)^{-1} b_E(u)$$

9. b.

$$E_{in}(w) = \frac{1}{N}|Xw - y|^2$$

$$= \frac{1}{N}(w^T X^T X w - 2w^T X^T y + y^T y)$$

$$\nabla E_{in} = \frac{1}{N}(2X^T X w - 2X^T y)$$

$$\nabla^2 E_{in} = \frac{2}{N}(X^T X)$$

10. b.

$$err(w, x, y) = -\sum [y=k] \ln h_k(x)$$

$$= -\left([y=k]\frac{1}{h_k(x)} - (1-[y=k])\frac{1}{1-h_k(x)}\right)\cdot \frac{\partial h_k(x)}{\partial w_i}$$

$$= -h_k(x)(1-h_k(x))\cdot x_i\left([y=k]\frac{1}{h_k(x)} - (1-[y=k])\frac{1}{1-h_k(x)}\right)$$

$$= -x_i\left([y=k](1-h_k(x)) - (1-[y=k])h_k(x)\right)$$

$$= -x_i\left([y=k] - [y=k]h_k(x) - h_k(x) + [y=k]h_k(x)\right)$$

$$= (h_k(x) - [y=k])\cdot x_i$$

掃描全能王 創建

Code

14~20

d、c、c、b、a、b、d

讀檔案

```
def load_data(file):
    f = open(file, "r")
    data = f.read()
    f.close()

    all_data = []
    all_label = []

    for line in data.split("\n"):
        sub_data = []
        i = 0
        for value in line.split("\t"):
            if value != "" and i == 10:
                value = float(value)
                all_label.append(value)
                i += 1
            elif value != "":
                value = float(value)
                sub_data.append(value)
                i += 1
        if sub_data != []:
            all_data.append(sub_data)

    for i in range(len(all_data)):
        all_data[i].insert(0, 1)

    all_data = np.array(all_data)
    all_label = np.array(all_label)
    return all_data, all_label


train_data, train_label = load_data("./hw3_train.dat.tx
t")
test_data, test_label = load_data("./hw3_test.dat.txt")
```

14

```
def linear_regression(x, y):
    x_pinv = np.linalg.pinv(x)
    w = np.dot(x_pinv, y)
    return w

w = linear_regression(train_data, train_label)
se_arr = []
for i in range(len(train_data)):
    pred = np.dot(train_data[i], w)
    se = (pred - train_label[i])**2
    se_arr.append(se)
Ein = np.mean(se_arr)
print(E_in)
```

15

```
def SGD_LR(x, y, lr, Ein):
    epoch = 0
    Ein_SGD = 100
    w = np.zeros(11)
    while Ein_SGD > Ein*1.01:
        num = random.randint(0, 999)
        w = w + lr * 2 * (y[num] - np.dot(w, x[num].T))
 * x[num]
        Ein_SGD = np.square(np.dot(w, x.T) - y)
        Ein_SGD = np.mean(Ein_SGD)
        epoch += 1
    return epoch

epoch_arr = []
for i in range(1000):
    print(i)
    epoch = SGD_LR(train_data, train_label, 0.001, Ein)
    epoch_arr.append(epoch)
ans = np.mean(epoch_arr)
print(ans)
```

16

```
def theta(s):
    return 1 / (1 + np.exp(-s))

def SGD_Logistic(x, y, lr):
    w = np.zeros(11)
    for i in range(500):
        num = random.randint(0, 999)
        grad = theta(-
y[num]*np.dot(w, x[num].T))*(y[num]*x[num])
        w = w + lr*grad
    ce_arr = []
    for j in range(len(x)):
        ce = np.log(1 + np.exp(-
y[j]*np.dot(w, x[j].T)))
        ce_arr.append(ce)
    Ein = np.mean(ce_arr)
    return Ein

Ein_arr = []
for i in range(1000):
    Ein = SGD_Logistic(train_data, train_label, 0.001)
    Ein_arr.append(Ein)
ans = np.mean(Ein_arr)
print(ans)
```

17

```
def SGD_Logistic_17(x, y, lr):
    w = linear_regression(x, y)
    for i in range(500):
        num = int(np.random.uniform(0, len(x)))
        grad = theta(-
y[num]*np.dot(w, x[num].T))*(y[num]*x[num])
        w = w + lr*grad
    ce_arr = []
    for j in range(len(x)):
        ce = np.log(1 + np.exp(-
y[j]*np.dot(w, x[j].T)))
```

```
        ce_arr.append(ce)
    Ein = np.mean(ce_arr)
    return Ein

Ein_arr = []
for i in range(1000):
    Ein = SGD_Logistic_17(train_data, train_label, 0.00
1)
    Ein_arr.append(Ein)
ans = np.mean(Ein_arr)
print(ans)
```

18

```
def sign(number):
    if number > 0:
        return 1
    else:
        return -1

def Ein_Eout_diff(train_x, train_y, test_x, test_y):
    w = linear_regression(train_x, train_y)
    error = 0
    for i in range(len(train_x)):
        pred = sign(np.dot(w, train_x[i].T))
        if pred != train_y[i]:
            error += 1
    Ein = error / len(train_x)
    error = 0
    for i in range(len(test_x)):
        pred = sign(np.dot(w, test_x[i].T))
        if pred != test_y[i]:
            error += 1
    Eout = error / len(test_x)
    return np.abs(Ein - Eout)

ans = Ein_Eout_diff(train_data, train_label, test_data,
 test_label)
print(ans)
```

19

```
def trasform(x):
    new_x = []
    for i in range(len(x)):
        tmp = []
        tmp.append(1)
        for j in range(len(x[i])):
            if x[i][j] != 1:
                tmp.append(x[i][j])
                tmp.append(x[i][j]**2)
                tmp.append(x[i][j]**3)
        new_x.append(tmp)
    new_x = np.array(new_x)
    return new_x

new_train_data = trasform(train_data)
new_test_data = trasform(test_data)

ans = Ein_Eout_diff(new_train_data, train_label, new_te
st_data, test_label)
print(ans)
```

20

```
def trasform(x):
    new_x = []
    for i in range(len(x)):
        tmp = []
        tmp.append(1)
        for j in range(len(x[i])):
            if x[i][j] != 1:
                tmp.append(x[i][j])
                tmp.append(x[i][j]**2)
                tmp.append(x[i][j]**3)
                tmp.append(x[i][j]**4)
                tmp.append(x[i][j]**5)
                tmp.append(x[i][j]**6)
                tmp.append(x[i][j]**7)
                tmp.append(x[i][j]**8)
```

```
                tmp.append(x[i][j]**9)
                tmp.append(x[i][j]**10)
        new_x.append(tmp)
    new_x = np.array(new_x)
    return new_x

new_train_data = trasform(train_data)
new_test_data = trasform(test_data)

ans = Ein_Eout_diff(new_train_data, train_label, new_te
st_data, test_label)
print(ans)
```