1. c

1.

$$\min_{w} \frac{1}{2}\int_0^2 (e^x - wx)^2 dx$$

$$= \min_{w} \frac{1}{2}e^{2x} + \frac{1}{3}w^2x^3 - 2we^x(x-1)\Big|_0^2$$

$$= \min_{w} \frac{8}{3}w^2 - 2(e^2+1)w + \frac{1}{2}e^4 - \frac{1}{2}$$

$$= \min_{w} \left(w - \frac{3}{8}(e^2+1)\right)^2 + \frac{1}{2}e^4 - \frac{1}{2} - \frac{9}{64}(e^2+1)^2$$

when $w = \frac{3+3e^2}{8}$, there will be squared error with min value

2. b

2.

$$A(D) = \arg\min_{h\in H} E_{in}(h)$$

$$A^*(D) = \arg\min_{h\in H} E_{in+out}(h)$$

$$E_D[E_{in}(A(D))] \le E_D[E_{in}(A^*(D))]$$

$$E_D[E_{out}(A^*(D))] \le E_D[E_{out}(A(D))]$$

since $A^*(D)$ is generated by considering both Data in and out-samples

$$E_D[E_{in}(A^*(D))] = E_D[E_{out}(A^*(D))]$$

3. d

3.

$$X_h^T = \begin{bmatrix} x_1 & x_2 & \cdots & x_1+\varepsilon & x_2+\varepsilon & \cdots & x_N+\varepsilon \end{bmatrix}_{d+1 \times 2N} \qquad X_h = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_1+\varepsilon \\ \vdots \\ x_N+\varepsilon \end{bmatrix}_{2N \times d+1}$$

$$X_{ij} \text{ for } X_h^T X_h = 2\sum_{k=1}^{2N} X_h^T[i][k] \cdot X_h[k][j] + N \cdot \varepsilon^2 + \varepsilon \sum_{k=1}^{N} (X_{ik} + X_{kj})$$

∵ take expected value, the last term with ε is equal to zero due to normal distribution of noise.

$$E(X_h^T X_h) = 2X^T X + N \cdot \sigma^2 I_{d+1} \quad ※$$

4. e

4.

$$X_h^T = \begin{bmatrix} x_1 & x_2 & \cdots & x_1+\varepsilon & \cdots & x_N+\varepsilon \end{bmatrix}_{d+1 \times 2N} \qquad y_h = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \\ y_1 \\ \vdots \\ y_N \end{bmatrix}_{2N \times 1}$$

$$X_h^T y_h(i) = \sum_{j=1}^{2N} X_h^T(i)(j)\, y_h(j) = 2\sum_{j=1}^{2N} X_h^T(i)(j)\, y_h(j) + \varepsilon \sum_{j=1}^{2N} y_h(j)$$

same as (3), after taking expected value, the later term $= 0$.

$$E(X_h^T y_h) = 2X^T y \quad ※$$

5. d

5. $w = (Z^TZ + \lambda I)^{-1} Z^T y$, $Z = XQ$, $X^TX = Q\Gamma Q^T$, $Q^T = Q^{-1}$

$\lambda = 0$, $(Z^TZ)^{-1} = ((XQ)^T XQ)^{-1} = (Q^T X^T X Q)^{-1} = Q^T (X^TX)^{-1} Q$

$= Q^T Q \Gamma^{-1} Q^T Q = \Gamma^{-1}$ $\left((X^TX)^{-1} = Q\Gamma^{-1}Q^T\right)$

$\lambda > 0$, $(Z^TZ + \lambda I)^{-1} = (\Gamma + \lambda I)^{-1}$

$\Gamma^{-1} = \begin{bmatrix} \frac{1}{r_0} & & & \\ & \frac{1}{r_1} & & \\ & & \ddots & \\ & & & \frac{1}{r_d} \end{bmatrix}$, $(\Gamma + \lambda I)^{-1} = \begin{bmatrix} \frac{1}{r_0+\lambda} & & & \\ & \frac{1}{r_1+\lambda} & & \\ & & \ddots & \\ & & & \frac{1}{r_d+\lambda} \end{bmatrix}$

$\therefore \frac{u_i}{v_i} = \frac{\frac{1}{r_i+\lambda}}{\frac{1}{r_i}} = \frac{r_i}{r_i+\lambda}$ #

6. a

6.

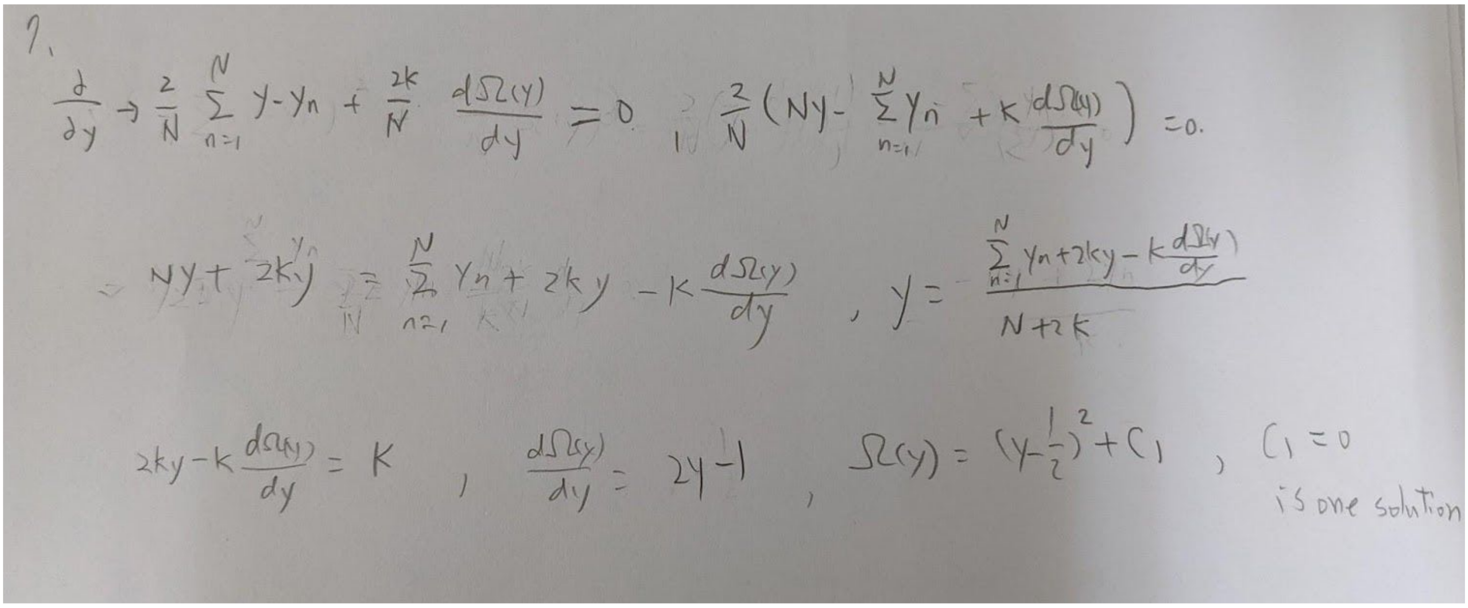
$\nabla\left(\frac{1}{N}\sum(wx_n - y_n)^2 + \frac{\lambda}{N}w^2\right) = \frac{2}{N}\sum x_n(wx_n - y_n) + \frac{2\lambda}{N}w = 0$

$w\left(\sum x_n^2 + \lambda\right) = \sum x_n y_n$, $w^* = \frac{\sum x_n y_n}{\sum x_n^2 + \lambda}$, $(w^*)^2 = \left(\frac{\sum_{n=1}^{N} x_n y_n}{\left(\sum_{n=1}^{N} x_n^2\right) + \lambda}\right)^2$ #

7. d*

Following photo shows the regular deduction, while there is another straightforward view.

Since we all know that the prior of the fair coin flip is half for two sides. To make prediction more close to prior probability, when y = 0.5, target function will receive the least penalty.



7.

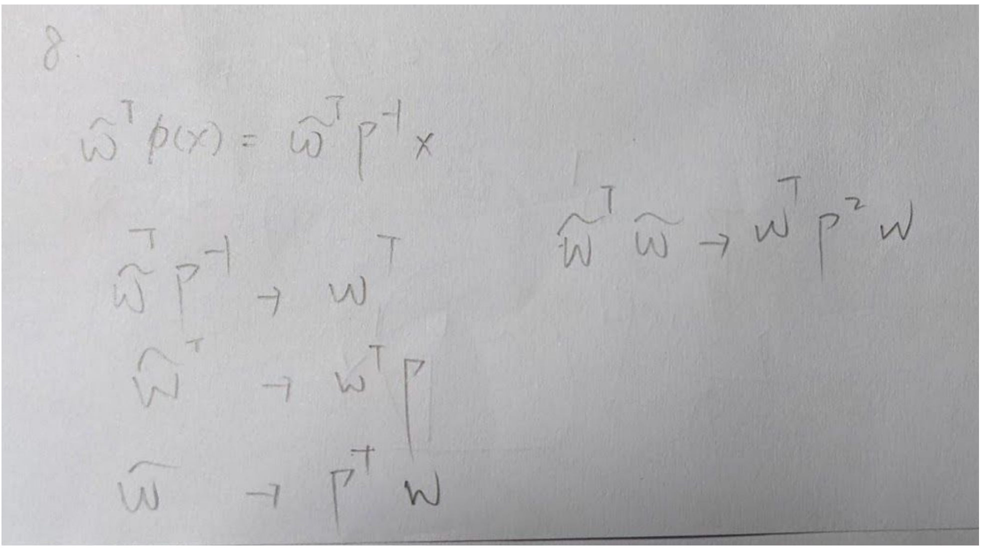
$$\frac{d}{dy} \to \frac{2}{N}\sum_{n=1}^{N} y-y_n + \frac{2k}{N}\frac{d\Omega(y)}{dy} = 0, \quad \frac{2}{N}\left(Ny - \sum_{n=1}^{N} y_n + k\frac{d\Omega(y)}{dy}\right) = 0.$$

$$Ny + 2ky = \sum_{n=1}^{N} y_n + 2ky - k\frac{d\Omega(y)}{dy}, \quad y = \frac{\sum_{n=1}^{N} y_n + 2ky - k\frac{d\Omega(y)}{dy}}{N+2k}$$

$$2ky - k\frac{d\Omega(y)}{dy} = k, \quad \frac{d\Omega(y)}{dy} = 2y-1, \quad \Omega(y) = \left(y-\frac{1}{2}\right)^2 + C_1, \quad C_1 = 0$$

is one solution

8. b



8.

$$\tilde{w}^T \phi(x) = \tilde{w}^T \Gamma^{-1} x$$

$$\tilde{w}^T \Gamma^{-1} \to w^T$$

$$\tilde{w}^T \to w^T \Gamma$$

$$\tilde{w} \to \Gamma^T w$$

$$\tilde{w}^T \tilde{w} \to w^T \Gamma^2 w$$

9. b

9. $\lambda \sum_{i=0}^{d} B_i w_i^2 = w^T \lambda B w$

$$\sum_{k=1}^{K} (w^T \hat{x}_n - \hat{y}_n)^2 = \sum (w^T \hat{x}_n)^2 - 2\sum w^T \hat{x}_n \hat{y}_n + \sum \hat{y}_n^2$$

$$= w^T \hat{X}^T \hat{X} w + \hat{y}\hat{y}^T - 2(\hat{X} w)^T \hat{y}$$

when $\hat{X} = \sqrt{\lambda} \cdot \sqrt{B}$, where $\sqrt{B}$ is the diagonal matrix with $\sqrt{B_0}, \sqrt{B_1} \ldots$ in the diagonals, and $\hat{y} = 0$. The result is same as previous. ※

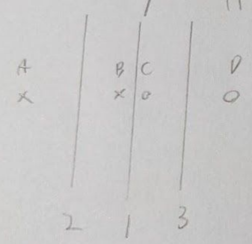
10. e

When selecting each point in data, the class for validation is always different from the majority of sub training set. For example, choosing +1 as validation point, and the A_majority will return -1 since there are N-1 positive and N negative points in training set.

11. c

11.

Error only happens at the neighbor of positive and negative point.



Validation point	number of error	hypothesis
B	1	2
C	1	3

$E_{LOOCV} = \frac{2}{N}$ ※

12. e

12. constant hypothesis

valid	$h(x)$	error
$(e,2)$	$h(x)=0$	$e_1 = 2^2 = 4$
$(3,0)$	$h(x)=1$	$e_2 = 1^2 = 1$
$(-3,0)$	$h(x)=1$	$e_3 = 1$

linear hypothesis

valid	$h(x)$	error
$(e,2)$	$h(x)=0$	$e_1 = 2^2 = 4$
$(-3,0)$	$h(x) = \frac{2}{e-3}(x-3)$	$e_2 = \left(\frac{-12}{e-3}\right)^2$
$(3,0)$	$h(x) = \frac{2}{e+3}(x+3)$	$e_3 = \left(\frac{12}{e+3}\right)^2$

$$4+1+1 = 4 + \left(\frac{-12}{e-3}\right)^2 + \left(\frac{12}{e+3}\right)^2, \quad \left(\frac{1}{e-3}\right)^2 + \left(\frac{1}{e+3}\right)^2 = \frac{1}{72}$$

$$e^4 - 18e^2 - 1215 = 0, \quad e^2 = 81 \pm 36\sqrt{6}, \text{ (負不合)}, \quad e = \sqrt{81+36\sqrt{6}} \text{ ※}$$

13. d

13.

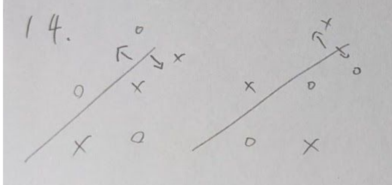
$$\operatorname{Var}_{D_{val}\sim P^K}[E_{val}(h)] = \operatorname{Var}_{D_{val}\sim P^K}\left[\frac{1}{K}\sum err(h(x),y)\right] = \frac{1}{K^2}\operatorname{Var}_{D_{val}\sim P^K}\left[\sum err(h(x),y)\right]$$

$$= \frac{1}{K^2}\sum \operatorname{Var}_{(x,y)\sim P}(err(h(x),y)) = \frac{1}{K^2}\cdot K \operatorname{Var}_{(x,y)\sim P}(err(h(x),y))$$

$$= \frac{1}{K}\cdot \operatorname{Var}_{(x,y)\sim P}[err(h(x),y)] \text{ ※}$$

14. c

14.



$$\left(\frac{1}{4}+\frac{1}{4}\right)\frac{1}{16} = \frac{1}{32} \text{ ※}$$

$E_{in} = \frac{1}{4}$ $\quad$ $E_{in} = \frac{1}{4}$

15. a

15.

$$p \cdot \varepsilon_+ + (1-p) \cdot \varepsilon_- = 1-p$$

$$p(\varepsilon_+ - \varepsilon_- + 1) = 1 - \varepsilon_-$$

$$p = \frac{1-\varepsilon_-}{\varepsilon_+ - \varepsilon_- + 1} \quad ※$$

16. b

a. Shell script

```
./train -s 0 -c 0.00005 -e 0.000001 -q hw4_train_convert.txt
./predict -b 1 hw4_test_convert.txt hw4_train_convert.txt.model result16

./train -s 0 -c 0.005 -e 0.000001 -q hw4_test_convert.txt
./predict -b 1 hw4_test_convert.txt hw4_train_convert.txt.model result16

./train -s 0 -c 0.5 -e 0.000001 -q hw4_train_convert.txt
./predict -b 1 hw4_test_convert.txt hw4_train_convert.txt.model result16

./train -s 0 -c 50 -e 0.000001 -q hw4_train_convert.txt
./predict -b 1 hw4_test_convert.txt hw4_train_convert.txt.model result16

./train -s 0 -c 5000 -e 0.000001 -q hw4_train_convert.txt
./predict -b 1 hw4_test_convert.txt hw4_train_convert.txt.model result16
```

b. Termianl message

```
yiwenlai@YiWens-MBP  ~/Desktop/liblinear-2.42  INSERT  bash hw4_q16.sh
Accuracy = 51.6667% (155/300)
Accuracy = 51.6667% (155/300)
Accuracy = 80.6667% (242/300)
Accuracy = 87% (261/300)
Accuracy = 86.6667% (260/300)
```

17. a

a. Shell script

```
./train -s 0 -c 0.00005 -e 0.000001 -q hw4_train_convert.txt
./predict -b 1 hw4_train_convert.txt hw4_train_convert.txt.model result17
./train -s 0 -c 0.005 -e 0.000001 -q hw4_train_convert.txt
./predict -b 1 hw4_train_convert.txt hw4_train_convert.txt.model result17
./train -s 0 -c 0.5 -e 0.000001 -q hw4_train_convert.txt
./predict -b 1 hw4_train_convert.txt hw4_train_convert.txt.model result17
./train -s 0 -c 50 -e 0.000001 -q hw4_train_convert.txt
./predict -b 1 hw4_train_convert.txt hw4_train_convert.txt.model result17
./train -s 0 -c 5000 -e 0.000001 -q hw4_train_convert.txt
./predict -b 1 hw4_train_convert.txt hw4_train_convert.txt.model result17
```

b. Terminal messge:

```
yiwenlai@YiWens-MBP  ~/Desktop/liblinear-2.42  INSERT  bash hw4_q17.sh
Accuracy = 46.5% (93/200)
Accuracy = 80.5% (161/200)
Accuracy = 87% (174/200)
Accuracy = 90% (180/200)
Accuracy = 91% (182/200)
```

18. e,

a. Run the shell script first

```
python3 hw4_q18.py
./train -s 0 -c 0.00005 -e 0.000001 -q hw4_sub_train.txt
./predict -b 1 hw4_val.txt hw4_sub_train.txt.model result18
./train -s 0 -c 0.005 -e 0.000001 -q hw4_sub_train.txt
./predict -b 1 hw4_val.txt hw4_sub_train.txt.model result18
./train -s 0 -c 0.5 -e 0.000001 -q hw4_sub_train.txt
./predict -b 1 hw4_val.txt hw4_sub_train.txt.model result18
./train -s 0 -c 50 -e 0.000001 -q hw4_sub_train.txt
./predict -b 1 hw4_val.txt hw4_sub_train.txt.model result18
./train -s 0 -c 5000 -e 0.000001 -q hw4_sub_train.txt
./predict -b 1 hw4_val.txt hw4_sub_train.txt.model result18
```

b. Code(hw4_q18.py) to process data in q18:

```python
import numpy as np
sub_train = []
val = []
with open("hw4_train_convert.txt") as f:
    lines = f.readlines()
    for i in range(len(lines)):
        if i >= 120:
            val.append(lines[i])
        else:
```

```
            sub_train.append(lines[i])
with open("hw4_sub_train.txt",'w') as f1:
    for row in sub_train:
        f1.write(row)
with open("hw4_val.txt",'w') as f2:
    for row in val:
        f2.write(row)
```

c. After running the shell script, we choose the \lambda with highest accuracy, and train with it again and test on the whole test set. 1 minus the accuracy is the answer (1 - 85.667% := 0.143).

```
yiwenlai@YiWens-MBP  ~/Desktop/liblinear-2.42  INSERT  ./train -s 0 -c 50 -e 0.000001 -q hw4_sub_train.txt
yiwenlai@YiWens-MBP  ~/Desktop/liblinear-2.42  INSERT  ./predict -b 1 hw4_test_convert.txt  hw4_sub_train.txt.model result18
Accuracy = 85.6667% (257/300)
yiwenlai@YiWens-MBP  ~/Desktop/liblinear-2.42  INSERT
```

19. d

a. Shell Script

```
./train -s 0 -c 50 -e 0.000001 -q hw4_train_convert.txt
./predict -b 1 hw4_test_convert.txt hw4_train_convert.txt.model result19
```

b. Terminal message: 1 minus the highest accuracy is the answer (1 - 87% “= 0.13).

```
yiwenlai@YiWens-MBP  ~/Desktop/liblinear-2.42  INSERT  bash hw4_q19.sh
Accuracy = 87% (261/300)
```

20. c, 0.12

a. Table corresponding to each log10(\lamda), and last row is for the E_cv

4	2	0	-2	-4
17	31	32	34	35
26	37	36	32	31
19	34	36	38	38
16	30	32	34	31
18	32	33	38	36
19.2	32.8	33.8	35.2	34.2
0.52	0.18	0.155	0.12	0.145

b. Shell Script

```
python3 hw4_q20.py 0
./train -s 0 -c 0.00005 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 1
./train -s 0 -c 0.00005 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 2
./train -s 0 -c 0.00005 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 3
./train -s 0 -c 0.00005 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 4
./train -s 0 -c 0.00005 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20

python3 hw4_q20.py 0
./train -s 0 -c 0.005 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 1
./train -s 0 -c 0.005 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 2
./train -s 0 -c 0.005 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 3
./train -s 0 -c 0.005 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 4
./train -s 0 -c 0.005 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
```

```
python3 hw4_q20.py 0
./train -s 0 -c 0.5 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 1
./train -s 0 -c 0.5 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 2
./train -s 0 -c 0.5 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 3
./train -s 0 -c 0.5 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 4
./train -s 0 -c 0.5 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20

python3 hw4_q20.py 0
./train -s 0 -c 50 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 1
./train -s 0 -c 50 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 2
./train -s 0 -c 50 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 3
./train -s 0 -c 50 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 4
./train -s 0 -c 50 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
```

```
python3 hw4_q20.py 0
./train -s 0 -c 5000 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 1
./train -s 0 -c 5000 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 2
./train -s 0 -c 5000 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 3
./train -s 0 -c 5000 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
python3 hw4_q20.py 4
./train -s 0 -c 5000 -e 0.000001 -q hw4_sub_train_20.txt
./predict -b 1 hw4_val_20.txt hw4_sub_train_20.txt.model result20
```

c. Terminal Message:

```
yiwenlai@YiWens-MBP ~/Desktop/liblinear-2.42 INSERT bash hw4_q20.sh
start
Accuracy = 42.5% (17/40)
Accuracy = 65% (26/40)
Accuracy = 47.5% (19/40)
Accuracy = 40% (16/40)
Accuracy = 45% (18/40)
start
Accuracy = 77.5% (31/40)
Accuracy = 92.5% (37/40)
Accuracy = 85% (34/40)
Accuracy = 75% (30/40)
Accuracy = 80% (32/40)
start
Accuracy = 80% (32/40)
Accuracy = 90% (36/40)
Accuracy = 90% (36/40)
Accuracy = 80% (32/40)
Accuracy = 82.5% (33/40)
start
Accuracy = 85% (34/40)
Accuracy = 80% (32/40)
Accuracy = 95% (38/40)
Accuracy = 85% (34/40)
Accuracy = 95% (38/40)
start
Accuracy = 87.5% (35/40)
Accuracy = 77.5% (31/40)
Accuracy = 95% (38/40)
Accuracy = 77.5% (31/40)
Accuracy = 90% (36/40)
```

d. Code to process data in q20:

```
import sys
start_index = 0
end_index = 0
sub_train = []
```

```python
val = []
if len(sys.argv) < 2:
   print('no argument')
   sys.exit()
else:
   if sys.argv[1] == '0':
       print('start')
   start_index = 40 * int(sys.argv[1])
   end_index = start_index + 39
with open("hw4_train_convert.txt") as f:
   lines = f.readlines()
   for i in range(len(lines)):
       if start_index <= i <= end_index:
           val.append(lines[i])
       else:
           sub_train.append(lines[i])
with open("hw4_sub_train_20.txt",'w') as f1:
   for row in sub_train:
       f1.write(row)
with open("hw4_val_20.txt",'w') as f2:
   for row in val:
       f2.write(row)
```

Code to process all training and testing data in right format:

```python
inputfile = ['hw4_train.txt','hw4_test.txt']
outputfile = ['hw4_train_convert.txt','hw4_test_convert.txt']
# print(inputfile[1])

for x in range(2):
   text = []
   with open(inputfile[x]) as f:
       lines = f.readlines()
       for line in lines:
           transform = []
```

```
            nums = line.strip().split()
            data = [float(num) for num in nums]
            transform.append(data[len(data)-1])
            transform.append(1)
            for i in range(6):
                transform.append(data[i])
            for i in range(6):
                for j in range(6-i):
                    transform.append(data[i]*data[i+j])
            # print(len(transform))
            text.append(transform)
        f.close()
    with open(outputfile[x],'w')as f1:
        for t in text:
            row_text = ""
            for i in range(len(t)):
                if i == 0:
                    if t[i] == 1:
                        row_text += ("+1 ")
                    elif t[i] == -1:
                        row_text += ("-1 ")
                else:
                    row_text += (str(i)+":")
                    row_text += str(t[i])
                    row_text += " "
            f1.write(row_text+"\n")
        f1.close()
```