大家好，这篇是有关台大机器学习课程作业八的详解。

我的github地址：
https://github.com/Doraemonzzz

个人主页：
http://doraemonzzz.com/

作业地址:
https://www.csie.ntu.edu.tw/~htlin/course/ml15fall/

参考资料:
https://blog.csdn.net/a1015553840/article/details/51085129
http://www.vynguyen.net/category/study/machine-learning/page/6/
http://book.caltech.edu/bookforum/index.php
http://beader.me/mlnotebook/
https://blog.csdn.net/qian1122221/article/details/50130093
https://acecoooool.github.io/blog/

Problem 1

考虑一般情形，假设一共有L层，第l层有$d^{(\ell)}$个节点，激活函数为$\theta(x)$，损失函数为$J(x,y)$。

首先要进行前向传播，那么第$\ell-1$层到第l层一共需要的计算次数为

$$(d^{(\ell-1)}+1)\times d^{(\ell)}$$

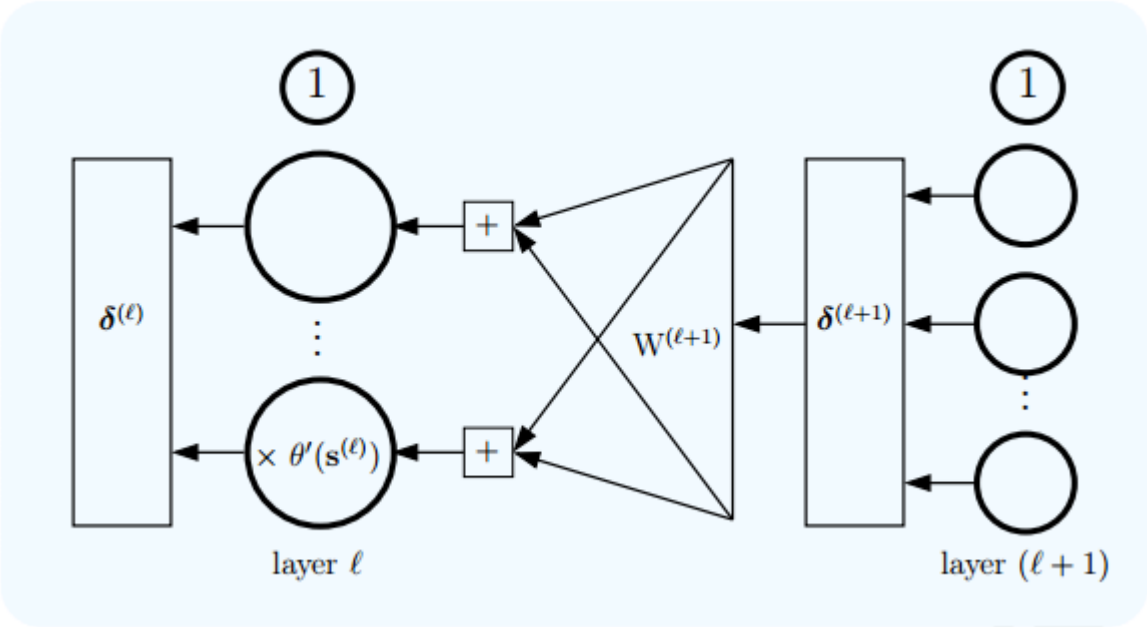
所以前向传播需要的计算次数为

$$\sum_{\ell=1}^{L}(d^{(\ell-1)}+1)\times d^{(\ell)}$$

对于此题来说，前向传播需要的计算次数为

$$(A+1)\times B+(B+1)\times 1$$

接着进行反向传播，先看计算图



接着回顾公式

$$\frac{\partial e}{\partial w_{ij}^{(\ell)}} = x_i^{(\ell-1)} \delta_j^{(\ell)}$$

$$\delta_j^{(\ell)} = \theta'(s_j^{(\ell)}) \sum_{k=1}^{d^{(\ell+1)}} w_{jk}^{(\ell+1)} \delta_k^{(\ell+1)}$$

从上述递推式可以看到，计算$\delta_j^{(\ell)}$需要的总计算次数和前向传播是一样的，计算$\frac{\partial e}{\partial w_{ij}^{(\ell)}}$总共需要的次数为第1层到第$L-1$层的节点总数（注意$\delta_j^{(L)}$是直接利用求导计算出来的，在这题中计算次数不计算在内），所以总共需要计算的次数为

$$\sum_{\ell=0}^{L}(d^{(\ell-1)}+1)\times d^{(\ell)} + \sum_{\ell=1}^{L-1} d^{(\ell)}$$

对于此题来说，反向传播需要的计算次数为

$$(A+1)\times B + (B+1)\times 1 + B$$

所以进行一轮随机梯度下降法总共需要的次数为

$$(A+1)\times B + (B+1)\times 1 + (A+1)\times B + (B+1)\times 1 + B = 2AB + 5B + 2$$

Problem 2

假设一共有L层，第l层有$d^{(\ell)}$个节点（包括偏置项）。

注意此题每一层的节点数量算上了偏置项1，但是偏置项只有输出，没有输入，所以第ℓ层到第$\ell+1$层的权重数量为

$$d^{(\ell)} \times (d^{(\ell+1)} - 1)$$

此题$d^{(0)}=10, d^{(L)}=1$（注意这里我和题目中的$d^{(0)}$不一致，我的$d^{(0)}$是包括偏置项的），所以我们需要计算的量为

$$S=\sum_{\ell=0}^{L-2} d^{(\ell)} \times (d^{(\ell+1)}-1)+d^{(L-1)}$$

如果我们将$d^{(L)}$改为2，那么上述求和式就更整齐

$$S=\sum_{\ell=0}^{L-1} d^{(\ell)} \times (d^{(\ell+1)}-1)=\sum_{\ell=0}^{L-1} d^{(\ell)}d^{(\ell+1)}-\sum_{\ell=0}^{L-1} d^{(\ell)}$$

现在的约束条件为

$$d^{(0)}=10, d^{(L)}=2, \sum_{\ell=1}^{L-1} d^{(\ell)}=36, d^{(\ell)}\geq 2$$

所以我们只要考虑

$$S_1=\sum_{\ell=0}^{L-1} d^{(\ell)}d^{(\ell+1)}$$

这里要解释一下最后一个约束条件，这是因为L在这里也是变量，我假设有L层就要求每一层至少有一个神经元，加上偏置项每一层就至少有两个神经元。

这个问题很难，没法用穷举法，因为数量太多了，一定要注意，这里$(10,2,34,2)$和$(10,34,2,2)$是两种结构，之前网上参考的答案就没有考虑次序问题，所以答案是不对的。这里直接给出最小值的答案，来自Learning from data作业6。

最小值发生在一共19层，第1层到18层都是2个节点，所以权重的数量为

$$10+2\times 18=46$$

Problem 3

最大值发生在一共4层，设第1层有x个节点，第2层有y个节点，$x+y=36$，由上一题的讨论，我们要求下式的最大值

$$\begin{aligned} T&=10x+xy+2y \\ &=10x+x(36-x)+2(36-x) \\ &=10x+36x-x^2+72-2x \\ &=-x^2+44x+72 \end{aligned}$$

所以当$x=22$时，上式取最大值，所以神经网络的结构为$(10,22,14,1)$，权重的数量为

$$10\times(22-1)+22\times(14-1)+14=510$$

Problem 4

先对$\text{err}(w)$进行化简

$$\begin{aligned}\text{err}(w) &= ||x_n - ww^Tx_n||^2 \\ &= (x_n - ww^Tx_n)^T(x_n - ww^Tx_n) \\ &= x_n^Tx_n - 2(w^Tx_n)^Tw^Tx_n + x_n^Tww^Tww^Tx_n \\ &= x_n^Tx_n - 2(w^Tx_n)^2 + (x_n^Tw)(w^Tw)(w^Tx_n)(\text{注意}\ w^Tx_n = x_n^Tw,\ \text{并且都为一个数}) \\ &= x_n^Tx_n - 2(w^Tx_n)^2 + (x_n^Tw)^2(w^Tw)\end{aligned}$$

接着求梯度

$$\begin{aligned}\nabla_w\text{err}(w) &= \nabla_w\Big(x_n^Tx_n - 2(w^Tx_n)^2 + (x_n^Tw)^2(w^Tw)\Big) \\ &= -4w^Tx_n\nabla_ww^Tx_n + 2x_n^Tw(w^Tw)\nabla_w(x_n^Tw) + (x_n^Tw)^2\nabla w^Tw \\ &= -4w^Tx_nx_n + 2x_n^Tw(w^Tw)x_n + 2(x_n^Tw)^2w\end{aligned}$$

Problem 5

$$\begin{aligned}E_{\text{in}}(w) &= \frac{1}{N}\sum_{n=1}^N||x_n - ww^T(x_n+\epsilon_n)||^2 \\ &= \frac{1}{N}\sum_{n=1}^N(x_n - ww^T(x_n+\epsilon_n))^T(x_n - ww^T(x_n+\epsilon_n)) \\ &= \frac{1}{N}\sum_{n=1}^N\Big(x_n^Tx_n - 2(x_n+\epsilon_n)^Tww^Tx_n + (x_n+\epsilon_n)^Tww^Tww^T(x_n+\epsilon_n)\Big) \\ &= \frac{1}{N}\sum_{n=1}^N\Big(x_n^Tx_n - 2x_n^Tww^Tx_n - 2\epsilon_n^Tww^Tx_n + x_n^Tww^Tww^Tx_n + \epsilon_n^Tww^Tww^T\epsilon_n + 2\epsilon_n^Tww^Tww^Tx_n\Big) \\ &= \frac{1}{N}\sum_{n=1}^N\Big(x_n^Tx_n - 2x_n^Tww^Tx_n + x_n^Tww^Tww^Tx_n\Big) + \frac{1}{N}\sum_{n=1}^N\Big(-2\epsilon_n^Tww^Tx_n + \epsilon_n^Tww^Tww^T\epsilon_n + 2\epsilon_n^Tww^Tww^Tx_n\Big) \\ &= \frac{1}{N}\sum_{n=1}^N||x_n - ww^Tx_n||^2 + \frac{1}{N}\sum_{n=1}^N\Big(-2\epsilon_n^Tww^Tx_n + \epsilon_n^Tww^Tww^T\epsilon_n + 2\epsilon_n^Tww^Tww^Tx_n\Big)\end{aligned}$$

由于ϵ服从标准正态分布，所以对上式取期望可得

$$
\begin{aligned}
\epsilon(E_{\text{in}}(w)) &= \frac{1}{N}\sum_{n=1}^{N}||x_n - ww^Tx_n||^2 + \frac{1}{N}\sum_{n=1}^{N}\epsilon\Big(\epsilon_n^Tww^Tww^T\epsilon_n\Big) \\
&= \frac{1}{N}\sum_{n=1}^{N}||x_n - ww^Tx_n||^2 + \frac{1}{N}\sum_{n=1}^{N}\text{trace}\Big[\epsilon\Big(\epsilon_n^Tww^Tww^T\epsilon_n\Big)\Big] \\
&= \frac{1}{N}\sum_{n=1}^{N}||x_n - ww^Tx_n||^2 + \frac{1}{N}\sum_{n=1}^{N}\epsilon\Big[\text{trace}\Big(\epsilon_n^Tww^Tww^T\epsilon_n\Big)\Big] \\
&= \frac{1}{N}\sum_{n=1}^{N}||x_n - ww^Tx_n||^2 + \frac{1}{N}\sum_{n=1}^{N}\epsilon\Big[\text{trace}\Big(ww^T\epsilon_n\epsilon_n^Tww^T\Big)\Big] \\
&= \frac{1}{N}\sum_{n=1}^{N}||x_n - ww^Tx_n||^2 + \frac{1}{N}\sum_{n=1}^{N}\text{trace}\Big[\epsilon\Big(ww^T\epsilon_n\epsilon_n^Tww^T\Big)\Big] \\
&= \frac{1}{N}\sum_{n=1}^{N}||x_n - ww^Tx_n||^2 + \frac{1}{N}\sum_{n=1}^{N}\text{trace}\Big[\Big(ww^Tww^T\Big)\Big] \\
&= \frac{1}{N}\sum_{n=1}^{N}||x_n - ww^Tx_n||^2 + \frac{1}{N}\sum_{n=1}^{N}\text{trace}\Big[\Big(w^Tww^Tw\Big)\Big] \\
&= \frac{1}{N}\sum_{n=1}^{N}||x_n - ww^Tx_n||^2 + (w^Tw)^2
\end{aligned}
$$

所以

$$
\Omega(w) = (w^Tw)^2
$$

Problem 6

由几何意义可知，两个点的$1NN$算法推导出来的边界为过这两个点中点的中锤面，所以可得超平面方程为：

$$
\begin{aligned}
(x^+ - x^-)^T(x - \frac{x^+ + x^-}{2}) = 0 \\
(x^+ - x^-)^Tx - \frac{(x^+ - x^-)^T(x^+ + x^-)}{2} = 0 \\
(x^+ - x^-)^Tx - \frac{||x^+||^2 - ||x^-||^2}{2} = 0
\end{aligned}
$$

从而分类器为

$$
g_{\text{Lin}}(x) = \text{sign}((x^+ - x^-)^Tx - \frac{||x^+||^2 - ||x^-||^2}{2})
$$

Problem 7

由$\text{sign}(x) = \text{sign}(ax)(a > 0)$可得：

$$
\begin{aligned}
g_{\text{rbfnet}}(x) &= \text{sign}(\beta_+\exp(-||x - \mu_+||^2) + \beta_-\exp(-||x - \mu_-||^2)) \\
&= \text{sign}(\beta_+\exp(||x - \mu_-||^2 - ||x - \mu_+||^2) + \beta_-)
\end{aligned}
$$

我们来计算$||x - \mu_-||^2 - ||x - \mu_+||^2$

$$
\begin{aligned}
||x-\mu_-||^2-||x-\mu_+||^2 &= (x-\mu_-)^T(x-\mu_-)-(x-\mu_+)^T(x-\mu_+) \\
&= x^T-2\mu_-^Tx+\mu_-^T\mu_- -(x^T-2\mu_+^Tx+\mu_+^T\mu_+) \\
&= 2(\mu_+-\mu_-)^Tx+\mu_-^T\mu_- -\mu_+^T\mu_+
\end{aligned}
$$

所以

$$
g_{\text{rbfnet}}(x)=\text{sign}(\beta_+\exp(2(\mu_+-\mu_-)^Tx+\mu_-^T\mu_- -\mu_+^T\mu_+)+\beta_-)
$$

我们来求解$\beta_+\exp(2(\mu_+-\mu_-)^Tx+\mu_-^T\mu_- -\mu_+^T\mu_+)+\beta_->0$

$$
\begin{aligned}
\beta_+\exp(2(\mu_+-\mu_-)^Tx+\mu_-^T\mu_- -\mu_+^T\mu_+)+\beta_->0 \Leftrightarrow \\
\exp(2(\mu_+-\mu_-)^Tx+\mu_-^T\mu_- -\mu_+^T\mu_+)>-\frac{\beta_-}{\beta_+} \Leftrightarrow \\
2(\mu_+-\mu_-)^Tx+\mu_-^T\mu_- -\mu_+^T\mu_+>\ln(-\frac{\beta_-}{\beta_+}) \Leftrightarrow \\
2(\mu_+-\mu_-)^Tx+\mu_-^T\mu_- -\mu_+^T\mu_+-\ln(-\frac{\beta_-}{\beta_+})>0
\end{aligned}
$$

同理可得

$$
\begin{aligned}
\beta_+\exp(2(\mu_+-\mu_-)^Tx+\mu_-^T\mu_- -\mu_+^T\mu_+)+\beta_-<0 \Leftrightarrow \\
2(\mu_+-\mu_-)^Tx+\mu_-^T\mu_- -\mu_+^T\mu_+-\ln(-\frac{\beta_-}{\beta_+})<0
\end{aligned}
$$

从而

$$
g_{\text{rbfnet}}(x)=\text{sign}(2(\mu_+-\mu_-)^Tx+\mu_-^T\mu_- -\mu_+^T\mu_+-\ln(-\frac{\beta_-}{\beta_+}))
$$

Problem 8

回顾定义：

$$
z_n=[\text{RBF}(x_n,x_1),\text{RBF}(x_n,x_2),\dots,\text{RBF}(x_n,x_N)]
$$

由于

$$
\text{RBF}(x;\mu)=[[x=\mu]]
$$

所以z_n为第n个元素为1，其余元素均为0的向量，从而$Z=I$，回顾β的计算公式

$$
\beta=(Z^TZ)^{-1}Z^Ty=y
$$

Problem 9

回顾我们的计算式：

$$\min_{W,V} E_{in}(w_m, v_n) \propto \sum_{\text{user n rated movie m}} \Big(r_{nm} - w_m^T v_n\Big)^2$$
$$= \sum_{m=1}^{M} \Big(\sum_{(x_n, r_{nm}) \in \mathcal{D}_m} \Big(r_{nm} - w_m^T v_n\Big)^2 \Big)$$

现在固定了v_n，我们要对每个m，求$\sum_{(x_n, r_{nm}) \in \mathcal{D}_m} \Big(r_{nm} - w_m^T v_n\Big)^2$的最小值，来求其梯度：

$$\nabla_{w_m} \sum_{(x_n, r_{nm}) \in \mathcal{D}_m} \Big(r_{nm} - w_m^T v_n\Big)^2 = \sum_{(x_n, r_{nm}) \in \mathcal{D}_m} 2\Big(r_{nm} - w_m^T v_n\Big) v_n = 0$$
$$\sum_{(x_n, r_{nm}) \in \mathcal{D}_m} \Big(r_{nm} - w_m^T v_n\Big) v_n = 0$$
$$\sum_{(x_n, r_{nm}) \in \mathcal{D}_m} r_{nm} v_n = \sum_{(x_n, r_{nm}) \in \mathcal{D}_m} (w_m^T v_n) v_n$$

注意此处$\tilde{d} = 1$，从而w_m, v_n都为实数，并且由题意，$v_n = 1$，从而

$$\sum_{(x_n, r_{nm}) \in \mathcal{D}_m} r_{nm} = \sum_{(x_n, r_{nm}) \in \mathcal{D}_m} w_m$$
$$w_m = \frac{\sum_{(x_n, r_{nm}) \in \mathcal{D}_m} r_{nm}}{\sum_{(x_n, r_{nm}) \in \mathcal{D}_m} 1}$$

这个量即为电影的平均分数。

Problem 10

根据定义计算v_{N+1}^T

$$v_{N+1}^T w_m = \frac{1}{N} \sum_{n=1}^{N} v_n^T w_m = \frac{1}{N} \sum_{n=1}^{N} r_{nm}$$

所以我们要找使得$\frac{1}{N} \sum_{n=1}^{N} r_{nm}$最大的电影，即评分最高的电影。

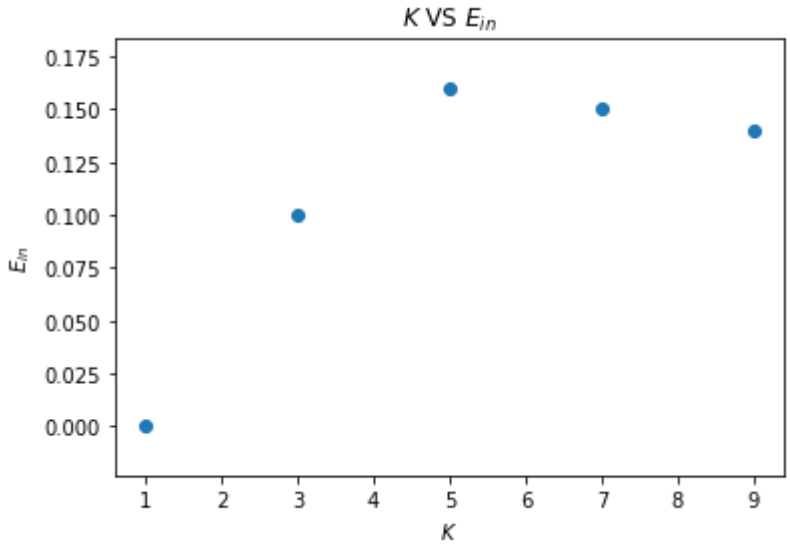
Problem 11-12

```
import numpy as np
import matplotlib.pyplot as plt
train = np.loadtxt("hw8_train.dat")
test = np.loadtxt("hw8_test.dat")

def Knn(x, data, k):
    distance = np.sum((x[:-1] - data[:, :-1])**2, axis = 1)
    index = np.argsort(distance)[:k]
    label = data[:, -1][index]
    return np.sign(np.sum(label))

def Error(data1, data2 ,k):
    error = 0
```

```
    for x in data1:
        #判断label是否相等
        error += Knn(x, data2, k) != x[-1]
    return error/data1.shape[0]
```

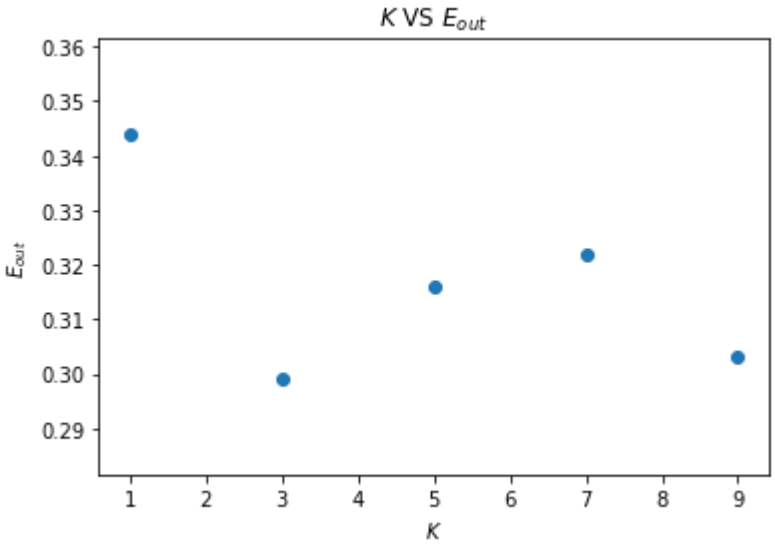
输出结果

```
K = [1, 3, 5, 7, 9]
Ein = []
for k in K:
    Ein.append(Error(train, train, k))

plt.scatter(K, Ein)
plt.title("$K$ VS $E_{in}$")
plt.xlabel("$K$")
plt.ylabel("$E_{in}$")
plt.show()
```



Problem 13-14

```
K = [1, 3, 5, 7, 9]
Eout = []
for k in K:
    Eout.append(Error(test, train, k))

plt.scatter(K, Eout)
plt.title("$K$ VS $E_{out}$")
plt.xlabel("$K$")
plt.ylabel("$E_{out}$")
plt.show()
```

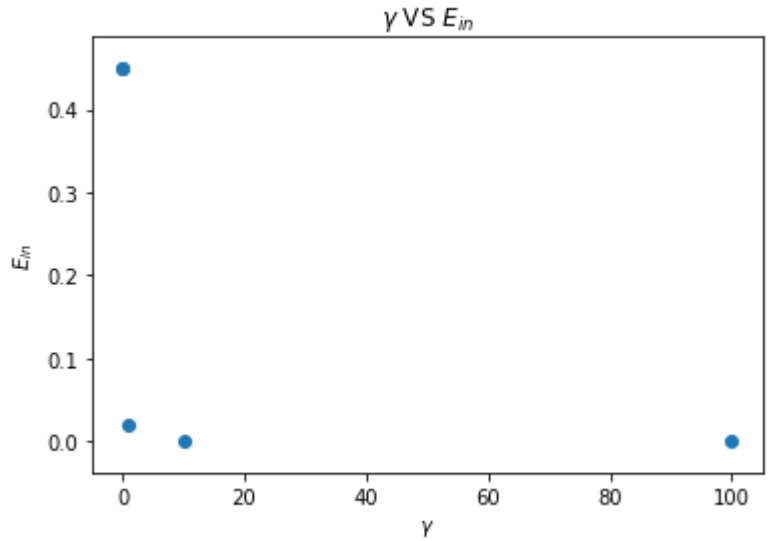


Problem 15-16

```
def Knn_new(x, data, gamma):
    distance = np.sum((x[:-1] - data[:, :-1])**2, axis = 1)
    Exp = np.exp(-gamma * distance)
    S = data[:, -1] * Exp
    return np.sign(np.sum(S))

def Error_new(data1, data2, gamma):
    error = 0
    for x in data1:
        #判断label是否相等
        error += Knn_new(x, data2, gamma) != x[-1]
    return error/data1.shape[0]
```

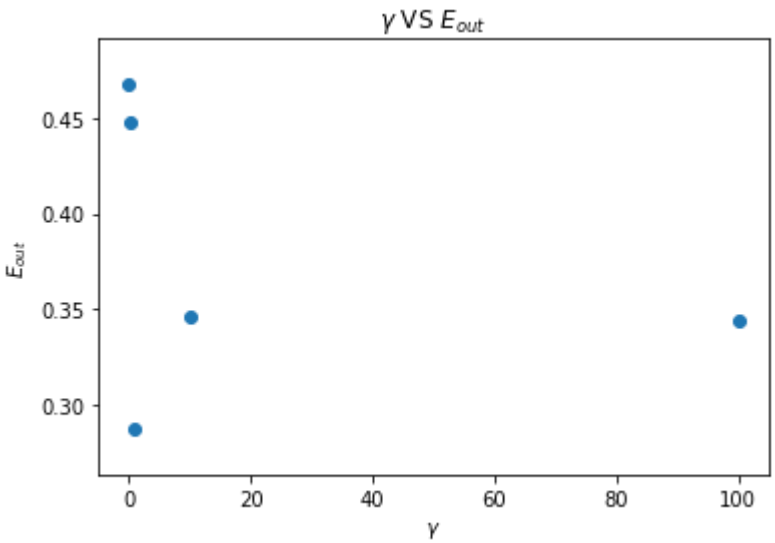
输出结果

```
Gamma = [0.001, 0.1, 1, 10, 100]
Ein_new = []
for gamma in Gamma:
    Ein_new.append(Error_new(train, train, gamma))

plt.scatter(Gamma, Ein_new)
plt.title("$\gamma$ VS $E_{in}$")
plt.xlabel("$\gamma$")
plt.ylabel("$E_{in}$")
plt.show()
```



Problem 17-18

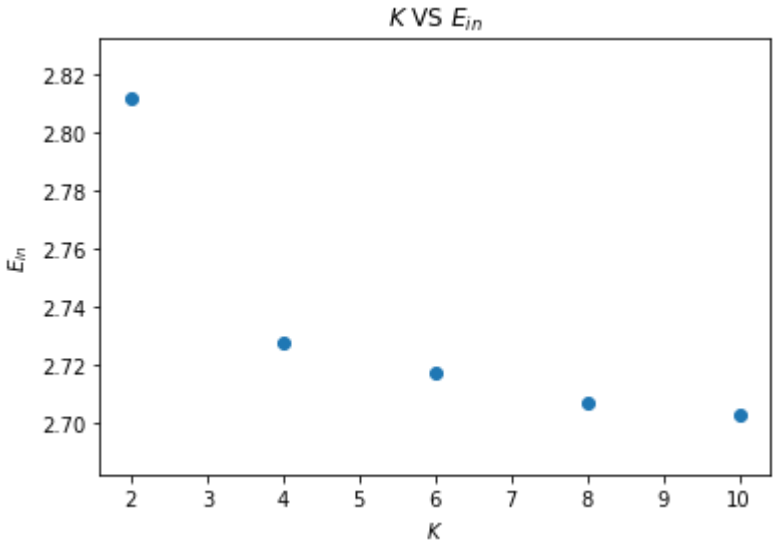
```
Gamma = [0.001, 0.1, 1, 10, 100]
Eout_new = []
for gamma in Gamma:
    Eout_new.append(Error_new(test, train, gamma))

plt.scatter(Gamma, Eout_new)
plt.title("$\gamma$ VS $E_{out}$")
plt.xlabel("$\gamma$")
plt.ylabel("$E_{out}$")
plt.show()
```



Problem 19-20

```
import numpy as np
```

```
import matplotlib.pyplot as plt
train = np.loadtxt("hw8_nolabel_train.dat")

def K_mean(data, k):
    data = train
    N = data.shape[0]
    #初始化中心
    index = np.random.choice(range(N), size=k, replace=False )
    center = data[index]
    newcenter = np.copy(center)
    label = np.zeros(N)
    while True:
        #Step 1:计算距离最近的中心
        for i in range(N):
            distance = np.sum((data[i] - center)**2, axis = 1)
            label[i] = np.argmin(distance)
        #Step 2:计算新的中心
        for i in range(k):
            newcenter[i] = np.mean(data[label == i])
        #判断是否收敛
        if np.sum((newcenter - center)**2) < 1e-5:
            break
        center = np.copy(newcenter)

    #计算误差
    Error = 0
    for i in range(k):
        Error += np.sum((data[label == i] - center[i])**2)
    return Error/N

K = [2, 4, 6, 8, 10]
Ein = []
for k in K:
    Ein.append(K_mean(train, k))

plt.scatter(K, Ein)
plt.title("$K$ VS $E_{in}$")
plt.xlabel("$K$")
plt.ylabel("$E_{in}$")
plt.show()
```



Problem 21

这题一开始没思路，做了下面一题反推了一些思路，但是我证明不出来原命题，只能证明如下更弱的命题：

$$\Delta \geq 2, \text{如果 } N \geq 3\Delta \log_2 \Delta, \text{那么 } N^{\Delta} + 1 \leq 2^N,$$
$$\text{其中 } \Delta, N \text{均为整数}$$

证明：

我们先证明如下结论：

$$\text{当 } N = 3\Delta \log_2 \Delta \text{时，} N^{\Delta} < 2^N$$

$$N^{\Delta} < 2^N \Leftrightarrow$$
$$N^{\Delta} < 2^{3\Delta \log_2 \Delta} \Leftrightarrow$$
$$N^{\Delta} < \Delta^{3\Delta} \Leftrightarrow$$
$$N < \Delta^3 \Leftrightarrow$$
$$3\Delta \log_2 \Delta < \Delta^3 \Leftrightarrow$$
$$3 \log_2 \Delta < \Delta^2$$

当$\Delta \geq 2$时，上式成立。

接着证明

$$N \geq 3\Delta \log_2 \Delta \text{时，} N^{\Delta} < 2^N$$

上式等价于取对数之后的情形

$$\Delta \ln N < N \ln 2$$

设$f(x) = \Delta \ln x - x \ln 2$，那么

$$f'(x) = \frac{\Delta}{x} - \ln 2$$
$$\text{所以当 } x \geq \frac{\Delta}{\ln 2}, f'(x) \leq 0; \text{ 当 } x < \frac{\Delta}{\ln 2}, f'(x) > 0$$

注意

$$3\Delta \log_2 \Delta = \frac{3\Delta \ln \Delta}{\ln 2}, \Delta \geq 2$$

所以

$$3\Delta \log_2 \Delta = \frac{3\Delta \ln \Delta}{\ln 2} \geq \frac{\Delta}{\ln 2}$$

从而当$x \geq 3\Delta \log_2 \Delta$时，$f'(x) \leq 0$，又因为$f(3\Delta \log_2 \Delta) < 0$，所以

$$\text{当} x \geq 3\Delta \log_2 \Delta \text{时}，\Delta \ln x < x \ln 2$$

所以

$$N \geq 3\Delta \log_2 \Delta \text{时}，N^{\Delta} < 2^N$$

又因为这里假定Δ, N都为整数，所以

$$N^{\Delta} + 1 \leq 2^N$$

Problem 22

因为第1层到第2层的变换是固定的，所以至少考虑第1层能产生多少个$0, 1$的组合就行。

回顾机器学习基石第6讲的一个函数$B(N, k)$

$$B(N, k)\text{表示break point为}k\text{时}，N\text{个点能表示的最多组合数量}。$$

我们知道d维感知机（不包括偏置项）的break point为$d + 1$，所以如果有N组数据，那么最多能够表示的组合数量小于等于

$$B(N, d + 1)$$

对于$d - 3 - 1$架构的神经网络，第0层到第1层的每个节点相当于感知机，表示的组合数量最多为$B(N, d + 1)$，所以3个节点能够表示的组合数量小于等于

$$B(N, d + 1)^3$$

回顾机器学习基石第6讲，有如下不等式估计

$$B(N, k) \leq \sum_{i=0}^{k-1} \binom{N}{i}$$

Learning from data第二章Problem 2.5（可以参考我的解答，传送门）给出如下结论

$$\sum_{i=0}^{D} \binom{N}{i} \leq N^D + 1$$

所以

$$B(N,d+1) \leq \sum_{i=0}^{d} \binom{N}{i} \leq N^d + 1 \leq N^{d+1} + 1$$

从而3个节点能够表示的组合数量小于等于

$$(N^{d+1}+1)^3 = N^{3(d+1)} + 3N^{2(d+1)} + 3N^{d+1} + 1$$

令$\Delta = 3(d+1)+1 \geq 4$，取$N \geq \Delta \geq 4$，那么

$$\begin{aligned}(N^{d+1}+1)^3 &= N^{3(d+1)} + 3N^{2(d+1)} + 3N^{d+1} + 1 \\ &< N^{3(d+1)} + N^{3(d+1)} + N^{3(d+1)} + 1 \\ &= 3N^{3(d+1)} + 1 \\ &< N^{3(d+1)+1} + 1\end{aligned}$$

如果$N \geq 3\Delta \log_2 \Delta$，那么满足Problem 21的条件，所以

$$(N^{d+1}+1)^3 < N^{3(d+1)+1} + 1 \leq 2^N$$

从而无法shatter大于等于$3\Delta \log_2 \Delta$个点，所以

$$VCD < 3\Delta \log_2 \Delta = 3(3(d+1)+1)\log_2(3(d+1)+1)$$

所以命题成立。