Machine Learning Techniques, Spring 2020, HW2

B07902064 資工二 蔡銘軒

July 22, 2020

Problem 1

Let $s_n = -y_n(A \cdot z_n + B)$, then we have $F(A,B) = \frac{1}{N}\sum_{n=1}^{N} ln(1+\exp(s_n))$.
Using the chain rule, we can calculate the partial derivatives as follows:

$$\frac{\partial F(A,B)}{\partial A} = \frac{1}{N}\sum_{n=1}^{N}\frac{\exp(s_n)}{1+\exp(s_n)}\cdot\frac{\partial s_n}{\partial A} = \frac{1}{N}\sum_{n=1}^{N}\frac{\exp(s_n)}{1+\exp(s_n)}\cdot -y_n z_n = \frac{1}{N}\sum_{n=1}^{N} -p_n y_n z_n$$

$$\frac{\partial F(A,B)}{\partial B} = \frac{1}{N}\sum_{n=1}^{N}\frac{\exp(s_n)}{1+\exp(s_n)}\cdot\frac{\partial s_n}{\partial B} = \frac{1}{N}\sum_{n=1}^{N}\frac{\exp(s_n)}{1+\exp(s_n)}\cdot -y_n = \frac{1}{N}\sum_{n=1}^{N} -p_n y_n$$

Then we have

$$\nabla F(A,B) = -\frac{1}{N}\sum_{n=1}^{N}[\ p_n y_n z_n,\ p_n y_n]^T$$

Problem 2

The matrix is given by

$$H(F) = \begin{bmatrix} \frac{\partial^2 F(A,B)}{\partial A^2} & \frac{\partial^2 F(A,B)}{\partial A \partial B} \\ \frac{\partial^2 F(A,B)}{\partial B \partial A} & \frac{\partial^2 F(A,B)}{\partial B^2} \end{bmatrix}$$

For future reference, the derivative of the logistic funtion is given by

$$\frac{d\theta(x)}{dx} = \frac{e^x}{(1+e^x)^2} = \frac{e^x}{1+e^x}\frac{1}{1+e^x} = \theta(x)\cdot(1-\theta(x))$$

Using the result and notation from the previous problem, we calculate the second derivatives as follows

$$\frac{\partial^2 F(A,B)}{\partial A^2} = -\sum_{n=1}^{N} y_n z_n \frac{\partial p_n}{\partial A} = -\sum_{n=1}^{N} y_n z_n p_n(1-p_n)\cdot\frac{\partial s_n}{\partial A} = \sum_{n=1}^{N}(y_n z_n)^2 p_n(1-p_n)$$

$$\frac{\partial^2 F(A,B)}{\partial A \partial B} = -\sum_{n=1}^{N} y_n z_n \frac{\partial p_n}{\partial B} = -\sum_{n=1}^{N} y_n z_n p_n(1-p_n)(-y_n) = \sum_{n=1}^{N} y_n^2 z_n p_n(1-p_n)$$

$$\frac{\partial^2 F(A,B)}{\partial B \partial A} = -\sum_{n=1}^{N} y_n \frac{\partial p_n}{\partial A} = -\sum_{n=1}^{N} y_n p_n(1-p_n)(-y_n z_n) = \sum_{n=1}^{N} y_n^2 z_n p_n(1-p_n)$$

$$\frac{\partial^2 F(A,B)}{\partial B^2} = -\sum_{n=1}^{N} y_n \frac{\partial p_n}{\partial B} = -\sum_{n=1}^{N} y_n p_n(1-p_n)(-y_n) = \sum_{n=1}^{N} y_n^2 p_n(1-p_n)$$

We have

$$H(F) = \sum_{n=1}^{N} \begin{bmatrix} (y_n z_n)^2 p_n(1-p_n) & y_n^2 z_n p_n(1-p_n) \\ y_n^2 z_n p_n(1-p_n) & y_n^2 p_n(1-p_n) \end{bmatrix}$$

Problem 3

We note that $y_n \in \{1, -1\}$ for $n = 1, 2, \ldots, N$, so $y_n^2 = 1$, and thus the Hessian matrix can be written as

$$H(F) = \begin{bmatrix} \sum_{n=1}^{N} z_n^2 p_n(1-p_n) & \sum_{n=1}^{N} z_n p_n(1-p_n) \\ \sum_{n=1}^{N} z_n p_n(1-p_n) & \sum_{n=1}^{N} p_n(1-p_n) \end{bmatrix}$$

From the definition of p_n, we have $0 < p_n < 1$. So we can define $u_i = \sqrt{z_n^2 p_n(1-p_n)}$ and $v_i = \sqrt{p_n(1-p_n)}$. By Cauchy inequality, we have

$$\det(H(F)) = \left(\sum_{n=1}^{N} u_i^2\right)\left(\sum_{n=1}^{N} v_i^2\right) - \left(\sum_{n=1}^{N} u_i v_i\right)^2 \geq 0$$

Since the determinants of all upper left square matrices are all nonnegative, the matrix $H(F)$ is positive semi-definite.

Problem 4

Assume $x_0 = 1$ is the bias term added for the input. We let $w_0 = d - 1$, and $w_1 = w_2 = \cdots = w_d = 1$, then we have $\sum_{i=0}^{d} w_i x_i = d - 1 + t - (d - t) = 2t - 1$, where t is the number of $+1$ in $x_1, x_2, \ldots, x_d$. It is clear for $t \geq 1$, that is, there is at least one TRUE in the input, the output will be TRUE, and the output is FALSE only when $t = 0$.

Problem 5

Assume the loss function is the squared error as used in class, we use the notation and result derived in class. For the last layer, we have

$$\frac{\partial e_n}{\partial w_{i1}^L} = \frac{\partial e_n}{\partial s_1^L} \cdot \frac{\partial s_1^L}{\partial w_{i1}^L} = -2(y_n - s_1^L) \cdot x_i^{L-1}$$

Except for $x_0^{L-1} = 1$, which is fixed by default, x_i^{L-1} for $i = 1, 2, \ldots, d^{L-1}$ are all zero, as they are the output of the previous layer, and as all weights are zeros, all outputs are zero as well.
For other layers($1 \leq l < L$), we have

$$\frac{\partial e_n}{\partial w_{ij}^l} = \frac{\partial e_n}{\partial s_j^l} \cdot \frac{\partial s_j^l}{\partial w_{ij}^l} = \delta_j^l \cdot x_i^{l-1}$$

with

$$\begin{aligned} \delta_j^l = \frac{\partial e_n}{\partial s_j^l} &= \sum_{k=1}^{d^{l+1}} \frac{\partial e_n}{\partial s_k^{l+1}} \frac{\partial s_k^{l+1}}{\partial x_j^l} \frac{\partial x_j^l}{\partial s_j^l} \\ &= \sum_{k=1}^{d^{l+1}} \delta_k^{l+1} w_{jk}^{l+1} \tanh'(s_j^l) \end{aligned}$$

As all w_{ij} are zero, we can see that except for $\frac{\partial e_n}{\partial w_{01}^L} = -2y_n$, all other gradient components are zero.

Problem 6

We simply enumerate all possible structures of the network and find the maximum. The following snippet does the calculation, and the answer is 877.

```cpp
#include <bits/stdc++.h>
using namespace std;

int mem[50][50];

int solve(int prev, int rem) {
   // run out of neurons to build another layer. Connect to output layer and return.
   if (!rem) return prev;
   // if we have solved the same problem before, return the answer directly.
   if (mem[prev][rem]) return mem[prev][rem];
    int ans = 0;
   //enumerate the number of neurons in this layer.
    for (int i = 2; i <= rem; ++i)
        ans = max(ans, prev * (i - 1) + solve(i, rem - i));
    // save the answer for future use
   return mem[prev][rem] = ans;
}

int main() {
    cout << solve(12, 48) << endl;
    return 0;
}
```

It is assumed that in every layer, there are at least two neurons, one of which is the bias term. The bias term is not connected with the neurons in the previous layer. These are taken into account in the snippet above.

Problem 7

$$\begin{aligned} err_n(\mathbf{w}) &= ||\mathbf{x}_n - \mathbf{w}\mathbf{w}^T\mathbf{x}_n||^2 \\ &= (\mathbf{x}_n - \mathbf{w}\mathbf{w}^T\mathbf{x}_n)^T(\mathbf{x}_n - \mathbf{w}\mathbf{w}^T\mathbf{x}_n) \\ &= \mathbf{x}_n^T\mathbf{x}_n - 2(\mathbf{w}^T\mathbf{x}_n)^T\mathbf{w}^T\mathbf{x}_n + \mathbf{x}_n^T\mathbf{w}\mathbf{w}^T\mathbf{w}\mathbf{w}^T\mathbf{x}_n \\ &= \mathbf{x}_n^T\mathbf{x}_n - 2(\mathbf{w}^T\mathbf{x}_n)^2 + (\mathbf{x}_n^T\mathbf{w})(\mathbf{w}^T\mathbf{w})(\mathbf{w}^T\mathbf{x}_n) \\ &= \mathbf{x}_n^T\mathbf{x}_n - 2(\mathbf{w}^T\mathbf{x}_n)^2 + (\mathbf{x}_n^T\mathbf{w})^2(\mathbf{w}^T\mathbf{w}) \end{aligned}$$

In the last step, we note that $\mathbf{x}_n^T\mathbf{w}$ is a scalar, and thus $\mathbf{x}_n^T\mathbf{w} = \mathbf{w}^T\mathbf{x}_n$.

$$\nabla_{\mathbf{w}} err_n(\mathbf{w}) = -4(\mathbf{w}^T\mathbf{x}_n)\mathbf{x}_n + 2(\mathbf{x}_n^T\mathbf{w})\mathbf{x}_n(\mathbf{w}^T\mathbf{w}) + 2(\mathbf{x}_n^T\mathbf{w})^2\mathbf{w}$$

Problem 8

$$
\begin{aligned}
E_{in}(\mathbf{w}) &= \frac{1}{N}\sum_{n=1}^{N} ||\mathbf{x}_n - \mathbf{w}\mathbf{w}^T(\mathbf{x}_n + \epsilon_n)||^2 \\
&= \frac{1}{N}\sum_{n=1}^{N} (\mathbf{x}_n - \mathbf{w}\mathbf{w}^T(\mathbf{x}_n + \epsilon_n))^T(\mathbf{x}_n - \mathbf{w}\mathbf{w}^T(\mathbf{x}_n + \epsilon_n)) \\
&= \frac{1}{N}\sum_{n=1}^{N} \mathbf{x}^T\mathbf{x} - 2(\mathbf{x}_n + \epsilon_n)^T\mathbf{w}\mathbf{w}^T\mathbf{x}_n + (\mathbf{x}_n + \epsilon_n)^T\mathbf{w}\mathbf{w}^T\mathbf{w}\mathbf{w}^T(\mathbf{x}_n + \epsilon_n) \\
&= \frac{1}{N}\sum_{n=1}^{N} \mathbf{x}^T\mathbf{x} - 2\mathbf{x}_n^T\mathbf{w}\mathbf{w}^T\mathbf{x}_n - 2\epsilon_n^T\mathbf{w}\mathbf{w}^T\mathbf{x}_n + \mathbf{x}_n^T\mathbf{w}\mathbf{w}^T\mathbf{w}\mathbf{w}^T\mathbf{x}_n + 2\epsilon_n^T\mathbf{w}\mathbf{w}^T\mathbf{w}\mathbf{w}^T\mathbf{x}_n + \epsilon_n^T\mathbf{w}\mathbf{w}^T\mathbf{w}\mathbf{w}^T\epsilon_n \\
&= \frac{1}{N}\sum_{n=1}^{N} \mathbf{x}^T\mathbf{x} - 2\mathbf{x}_n^T\mathbf{w}\mathbf{w}^T\mathbf{x}_n + \mathbf{x}_n^T\mathbf{w}\mathbf{w}^T\mathbf{w}\mathbf{w}^T\mathbf{x}_n + \frac{1}{N}\sum_{n=1}^{N} -2\epsilon_n^T\mathbf{w}\mathbf{w}^T\mathbf{x}_n + 2\epsilon_n^T\mathbf{w}\mathbf{w}^T\mathbf{w}\mathbf{w}^T\mathbf{x}_n + \epsilon_n^T\mathbf{w}\mathbf{w}^T\mathbf{w}\mathbf{w}^T\epsilon_n \\
&= \frac{1}{N}\sum_{n=1}^{N} ||\mathbf{x}_n - \mathbf{w}\mathbf{w}^T\mathbf{x}_n||^2 + \frac{1}{N}\sum_{n=1}^{N} -2\epsilon_n^T\mathbf{w}\mathbf{w}^T\mathbf{x}_n + 2\epsilon_n^T\mathbf{w}\mathbf{w}^T\mathbf{w}\mathbf{w}^T\mathbf{x}_n + \epsilon_n^T\mathbf{w}\mathbf{w}^T\mathbf{w}\mathbf{w}^T\epsilon_n
\end{aligned}
$$

For the first term, we have

$$
\mathbb{E}(\frac{1}{N}\sum_{n=1}^{N} ||\mathbf{x}_n - \mathbf{w}\mathbf{w}^T\mathbf{x}_n||^2) = \frac{1}{N}\sum_{n=1}^{N} ||\mathbf{x}_n - \mathbf{w}\mathbf{w}^T\mathbf{x}_n||^2
$$

With the calculation above and the knowledge that $\mathbb{E}(\epsilon_n) = 0$ and $\mathbb{E}(\epsilon_n\epsilon_n^T) = I_n$, we have

$$
\begin{aligned}
\Omega(\mathbf{w}) &= \mathbb{E}\left(\frac{1}{N}\sum_{n=1}^{N} -2\epsilon_n^T\mathbf{w}\mathbf{w}^T\mathbf{x}_n + 2\epsilon_n^T\mathbf{w}\mathbf{w}^T\mathbf{w}\mathbf{w}^T\mathbf{x}_n + \epsilon_n^T\mathbf{w}\mathbf{w}^T\mathbf{w}\mathbf{w}^T\epsilon_n\right) \\
&= \frac{1}{N}\sum_{n=1}^{N} \mathbb{E}\left(-2\mathbf{w}^T\epsilon_n\mathbf{w}^T\mathbf{x}_n + 2\mathbf{w}^T\epsilon_n\mathbf{w}^T\mathbf{w}\mathbf{w}^T\mathbf{x}_n + \mathbf{w}^T\mathbf{w}\mathbf{w}^T\epsilon_n\epsilon_n^T\mathbf{w}\right) \\
&= \frac{1}{N}\sum_{n=1}^{N} \mathbb{E}\left(\mathbf{w}^T\mathbf{w}\mathbf{w}^T\epsilon_n\epsilon_n^T\mathbf{w}\right) \\
&= \frac{1}{N}\sum_{n=1}^{N} (\mathbf{w}^T\mathbf{w})^2 \\
&= (\mathbf{w}^T\mathbf{w})^2
\end{aligned}
$$

Problem 9

Let

$$U = \begin{bmatrix} u_{11} & u_{12} & \dots & u_{1\tilde{d}} \\ u_{21} & u_{22} & \dots & u_{2\tilde{d}} \\ \vdots & \vdots & \ddots & \vdots \\ u_{d1} & u_{d2} & \dots & u_{d\tilde{d}} \end{bmatrix}$$

And the error function is given by $E = \sum_{i=1}^{d} (g_i(\mathbf{x}) - x_i)^2$. We expand $g(\mathbf{x})$ as follows:

$$\begin{aligned} U \cdot tanh(U^T \mathbf{x}) &= \begin{bmatrix} u_{11} & u_{12} & \dots & u_{1\tilde{d}} \\ u_{21} & u_{22} & \dots & u_{2\tilde{d}} \\ \vdots & \vdots & \ddots & \vdots \\ u_{d1} & u_{d2} & \dots & u_{d\tilde{d}} \end{bmatrix} tanh \left(\begin{bmatrix} u_{11} & u_{21} & \dots & u_{d1} \\ u_{12} & u_{22} & \dots & u_{d2} \\ \vdots & \vdots & \ddots & \vdots \\ u_{1\tilde{d}} & u_{2\tilde{d}} & \dots & u_{d\tilde{d}} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_d \end{bmatrix} \right) \\ &= \begin{bmatrix} u_{11} & u_{12} & \dots & u_{1\tilde{d}} \\ u_{21} & u_{22} & \dots & u_{2\tilde{d}} \\ \vdots & \vdots & \ddots & \vdots \\ u_{d1} & u_{d2} & \dots & u_{d\tilde{d}} \end{bmatrix} \begin{bmatrix} tanh(\sum_{i=1}^{d} u_{i1} x_i) \\ tanh(\sum_{i=1}^{d} u_{i2} x_i) \\ \vdots \\ tanh(\sum_{i=1}^{d} u_{i\tilde{d}} x_i) \end{bmatrix} \\ &= \begin{bmatrix} \sum_{j=1}^{\tilde{d}} (u_{1j} \cdot tanh(\sum_{i=1}^{d} u_{ij} x_i)) \\ \sum_{j=1}^{\tilde{d}} (u_{2j} \cdot tanh(\sum_{i=1}^{d} u_{ij} x_i)) \\ \vdots \\ \sum_{j=1}^{\tilde{d}} (u_{dj} \cdot tanh(\sum_{i=1}^{d} u_{ij} x_i)) \end{bmatrix} \end{aligned}$$

Now we have

$$E = \sum_{k=1}^{d} \left(\sum_{j=1}^{\tilde{d}} \left(u_{kj} \cdot tanh \left(\sum_{i=1}^{d} u_{ij} x_i \right) \right) - x_k \right)^2$$

Problem 10

We first formulate $E_{10}(\mathbf{w})$:

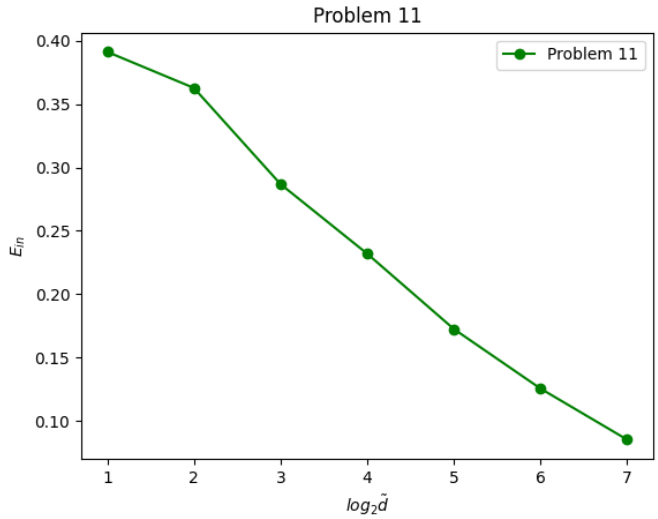
$$E_{10}(\mathbf{w}) = \sum_{k=1}^{d} \left(\sum_{j=1}^{\tilde{d}} \left(w_{jk}^{(2)} \cdot tanh \left(\sum_{i=1}^{d} w_{ij}^{(1)} x_i \right) \right) - x_k \right)^2$$

For $1 \le a \le d$, $1 \le b \le \tilde{d}$, the derivatives are given by

$$\begin{aligned} \frac{\partial E_9(\mathbf{u})}{\partial u_{ab}} &= \left(\sum_{k=1}^{d} 2 \left(\sum_{j=1}^{\tilde{d}} \left(u_{kj} \cdot tanh \left(\sum_{i=1}^{d} u_{ij} x_i \right) \right) - x_k \right) \cdot u_{kb} \cdot sech^2 \left(\sum_{i=1}^{d} u_{ib} x_i \right) x_a \right) \\ &\quad + 2 \left(\sum_{j=1}^{\tilde{d}} \left(u_{aj} \cdot tanh \left(\sum_{i=1}^{d} u_{ij} x_i \right) \right) - x_a \right) \cdot tanh \left(\sum_{i=1}^{d} u_{ib} x_i \right) \\ \frac{\partial E_{10}(\mathbf{w})}{\partial w_{ab}^{(1)}} &= \left(\sum_{k=1}^{d} 2 \left(\sum_{j=1}^{\tilde{d}} \left(w_{jk}^{(2)} \cdot tanh \left(\sum_{i=1}^{d} w_{ij}^{(1)} x_i \right) \right) - x_k \right) \cdot \left(w_{bk}^{(2)} \cdot sech^2 \left(\sum_{i=1}^{d} w_{ib}^{(1)} x_i \right) x_a \right) \right) \\ \frac{\partial E_{10}(\mathbf{w})}{\partial w_{ba}^{(2)}} &= \left(2 \sum_{j=1}^{\tilde{d}} \left(w_{ja}^{(2)} \cdot tanh \left(\sum_{i=1}^{d} w_{ij}^{(1)} x_i \right) \right) - x_a \right) \cdot tanh \left(\sum_{i=1}^{d} w_{ib}^{(1)} x_i \right) \end{aligned}$$

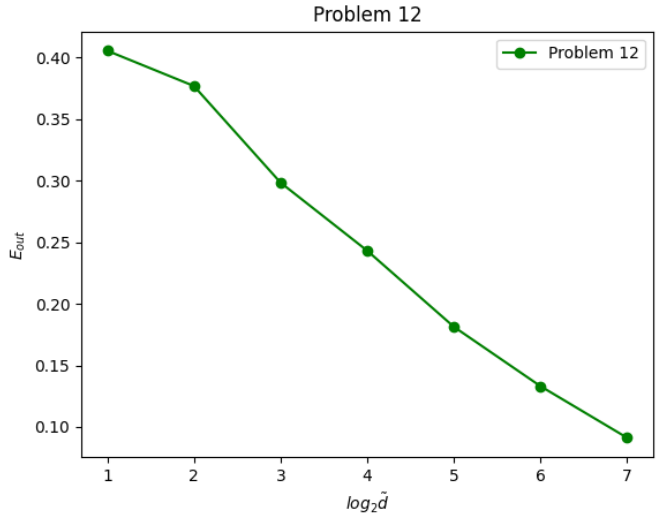
Under the condition $u_{ij} = w_{ij}^{(1)} = w_{ji}^{(2)}$, we can see $\frac{\partial E_9(\mathbf{u})}{\partial u_{ab}} = \frac{\partial E_{10}(\mathbf{w})}{\partial w_{ab}^{(1)}} + \frac{\partial E_{10}(\mathbf{w})}{\partial w_{ba}^{(2)}}$

Problem 11



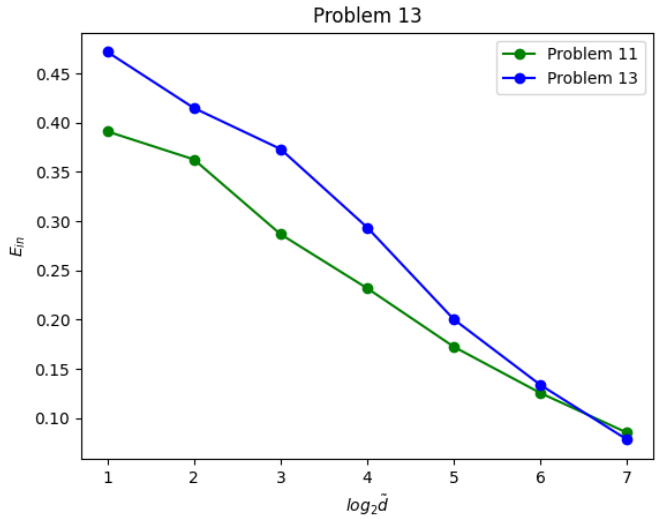
We can see that as the number of hidden layer increases, E_{in} drops. The result is reasonable as adding more hidden layers gives the model more "power" to fit the data.

Problem 12



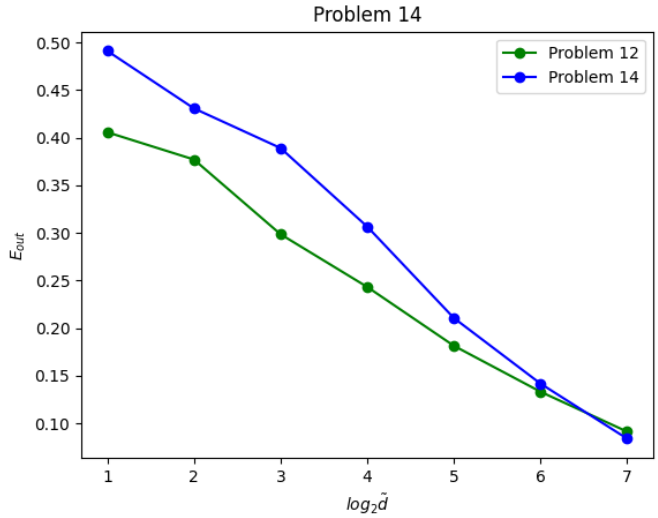
We can see that E_{out} also drops as more hidden layers are added. We can also observe that although E_{out} is higher than E_{in}, they are very close to each other. So it's safe to assume that we are not experiencing overfitting. This is also reasonable as the model has a simple structure and not so many parameters.

Problem 13



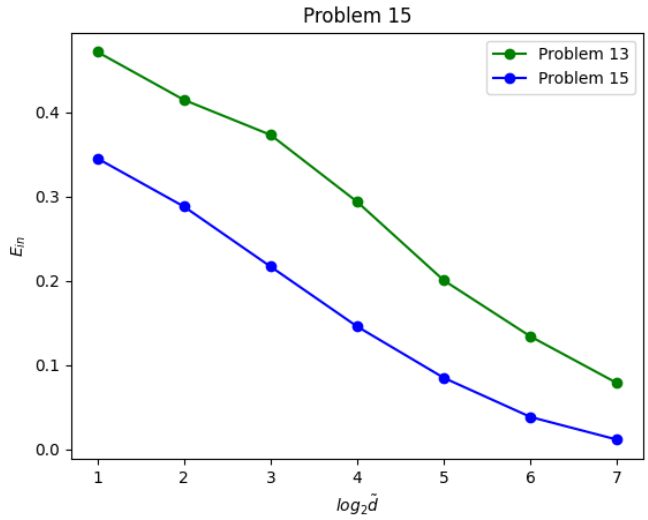
We can see the same pattern here. As more hidden layers are added, the model has more power to fit the data, leading to lower E_{in}. Compared with the figure in **Problem 11**, we observer that E_{in} is slightly higher when $log_2\tilde{d}$ is small. This is likely due to the regularization we impose on the model, as regularization usually contrains the model's ability to fit the training data, leading to higher E_{in}. But as $\tilde{d}$ gets bigger, the model has more ability to fit the data, so the gap decreases.

Problem 14



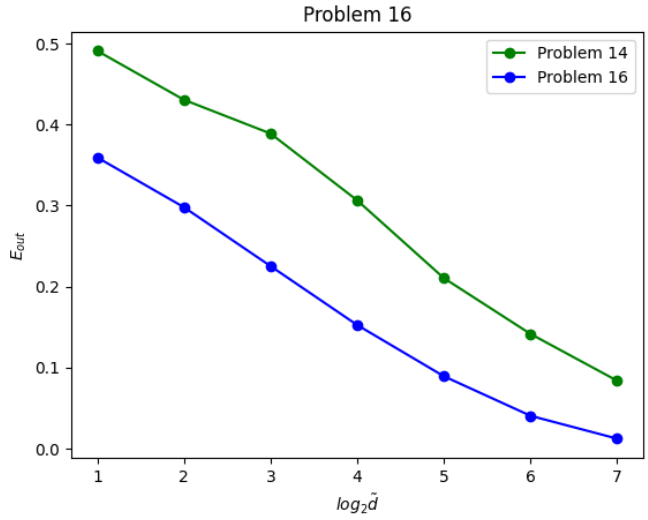
Similar to the figure in **Problem 12**, E_{out} also drops as $log_2\tilde{d}$ increases. And E_{out} is also close to E_{in}, so we don't have to be too worried about overfitting in this case. Compared with the figure in **Problem 12**, we can see that E_{out} is slightly higher than that in **Problem 12** when $\tilde{d}$ is small. A possible explanation is that, regularization is often used to combat overfitting, but the result in **Problem 11** and **Problem 12** does not show signs of overfitting. Adding constraints to models that are already simple can hurt performance.

Problem 15



PCA shows similar pattern to the figure in the previous problems. E_{in} decreases as $\tilde{d}$ increases. Compared with the result in **Problem 13**, we can see that PCA gives significant less error than the autoencoder with constraints.

Problem 16



E_{out} is slightly higher than E_{in}, but it also decreases as $\tilde{d}$ increases, and the gap between them is not very large. Compared with the result in **Problem 14**, we can see similar pattern to the figure in **Problem 15**. PCA has less error than autoencoder with constraints.

Problem 17

We show that for any $\Delta \geq 2$, $N^\Delta + 1 < 2^N$ for $N \in [3\Delta log_2\Delta, \infty)$.
We first discuss the case where $N = 3\Delta log_2\Delta$.

$$
\begin{aligned}
&LHS = N^\Delta + 1 = (3\Delta log_2\Delta)^\Delta + 1 \\
&RHS = 2^{3\Delta log_2\Delta} = (\Delta^3)^\Delta \\
&\because \Delta \geq 2 \\
&\therefore \Delta^3 - 3\Delta log_2\Delta \geq 2 \\
&\implies (3\Delta log_2\Delta)^\Delta + 1 < (\Delta^3)^\Delta \\
&\implies N^\Delta + 1 < 2^N
\end{aligned}
$$

To complete the proof, we derive the following result for later use.

$3\Delta log_2\Delta < \Delta^3$ for $\Delta \geq 2$
$\implies \Delta < \dfrac{3\Delta log_2\Delta}{log_2(3\Delta log_2\Delta)}$ take log_2 on both sides, divide both sides by LHS, multiply both sides by Δ
$\implies \Delta < \dfrac{3\Delta log_2\Delta}{log_2(3\Delta log_2\Delta)} < \dfrac{N}{log_2 N}$ for $N > 3\Delta log_2\Delta$, since $\dfrac{x}{log_2 x}$ is monotonically increasing in N's domain
$\implies \Delta log_2 N < N$
$\implies N^\Delta < 2^N$

We then consider the derivatives

$$
\begin{aligned}
&\frac{d(N^\Delta + 1)}{dN} = \Delta N^{\Delta - 1} \\
&\frac{d(2^N)}{dN} = ln2 \cdot 2^N
\end{aligned}
$$

As $N \geq 3\Delta log_2\Delta$, we have

$$\Delta N^{\Delta-1} < N^\Delta < 2^N < ln2 \cdot 2^N$$

Since $N^\Delta + 1 < 2^N$ when $N = 3\Delta log_2\Delta$, and the derivative of 2^N is larger than that of $N^\Delta + 1$ for $N > 3\Delta log_2\Delta$, we conclude with the **racetrack principle** that the original proposition is correct.

Problem 18

We first introduce some notations and conclusions from **Machine Learning Foundation**, which we later refer to as *rules*:

1. $m_\mathcal{H}(N)$ is the growth function.
2. $B(N, k)$ is the bounding function.
3. A d-dimensional perceptron has VC dimension $d_{vc} = d + 1$.
4. $m_\mathcal{H}(N) \leq B(N, k) = \sum_{i=0}^{k-1} \binom{N}{i} \leq N^{k-1}$, for $N \geq 2, k \geq 3$ where k is the break point.

Now let $\Delta = 3(d+1)+1$, where $d \geq 1$, and assume $N \geq 3\Delta log_2\Delta$. For a $d-3-1$ neural network, consider the perceptrons in the 1-*st* layer, the *rules* give

$$m_{\mathcal{H}_{3A}}(N) \leq B(N, d+2)^3 \leq \left(\sum_{i=0}^{d+1} \binom{N}{i}\right)^3 \leq (N^{d+1})^3 \leq (N^{d+1} + 1)^3$$

which leads to the following inequality

$$(N^{d+1} + 1)^3 = N^{3(d+1)} + 3N^{2(d+1)} + 3N^{(d+1)} + 1 < 3N^{3(d+1)} + 1 < N \cdot N^{3(d+1)} + 1 = N^{3(d+1)+1} + 1 = N^\Delta + 1$$

By the conclusion in the previous problem, we have

$$m_{\mathcal{H}_{3A}}(N) \leq (N^{d+1}+1)^3 < N^{\Delta}+1 < 2^N$$

This shows that the neural network cannot shatter any N points, for $N \geq 3\Delta log_2\Delta$, and thus the VC dimension is less than $3\Delta log_2\Delta = 3 \cdot (3(d+1)+1)\log_2(3(d+1)+1)$