COMPSCI 617 Machine Learning

Homework 5

Yiteng Lu

1.Probabilistic Interpretation of Ridge Regression

1.a

Answer

$l(\beta) = ||\mathbf{y} - \mathbf{X}\beta||_2^2 + \lambda||\beta||_2^2$

$$\begin{aligned} l(\beta) &= (\mathbf{y} - \mathbf{X}\beta)^T(\mathbf{y} - \mathbf{X}\beta) + \lambda\beta^T\beta \\ &= \left(\mathbf{y}^T\mathbf{y} - \mathbf{y}^T\mathbf{X}\beta - \beta^T\mathbf{X}^T\mathbf{y} + \beta^T\mathbf{X}^T\mathbf{X}\beta\right) + \lambda\beta^T\beta \end{aligned}$$

$$\begin{aligned} \frac{\mathrm{d}l(\beta)}{\mathrm{d}\beta} &= 0 - \mathbf{y}^T\mathbf{X} - \left(\mathbf{X}^T\mathbf{y}\right)^T + 2\beta^T\mathbf{X}^T\mathbf{X} + 2\lambda\beta^T = 0 \\ &= -2\mathbf{y}^T\mathbf{X} + 2\beta^T\mathbf{X}^T\mathbf{X} + 2\lambda\beta^T = 0 \\ &\Rightarrow -\mathbf{y}^T\mathbf{X} + \beta^T\mathbf{X}^T\mathbf{X} + \lambda\beta^T = 0 \\ &\Rightarrow \beta^T\left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}\right) = \mathbf{y}^T\mathbf{X} \\ &\Rightarrow \left[\beta^T\left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}\right)\right]^T = \left(\mathbf{y}^T\mathbf{X}\right)^T \\ &\Rightarrow \left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}\right)^T\beta = \mathbf{X}^T\mathbf{y} \\ &\Rightarrow \left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}\right)\beta = \mathbf{X}^T\mathbf{y} \\ &\Rightarrow \beta = \left(\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I}\right)^{-1}\mathbf{X}^T\mathbf{y} \end{aligned}$$

1.b

Answer

$$
\begin{aligned}
Pr(\beta|\mathbf{y},\mathbf{X},\lambda) &= \frac{Pr(\mathbf{y}|\mathbf{X},\beta)Pr(\beta)}{Pr(\mathbf{y})} \\
&\propto Pr(\mathbf{y}|\mathbf{X},\beta)Pr(\beta) \\
&\propto \exp\left\{-\frac{1}{2}(\mathbf{y}-\mathbf{X}\beta)^T(\mathbf{I}_{n\times n})^{-1}(\mathbf{y}-\mathbf{X}\beta)\right\}\exp\left\{-\frac{1}{2}(\beta^T\lambda\mathbf{I}\beta)\right\} \\
&\propto \exp\left\{-\frac{1}{2}(\mathbf{y}-\mathbf{X}\beta)^T(\mathbf{y}-\mathbf{X}\beta)-\frac{1}{2}(\beta^T\lambda\mathbf{I}\beta)\right\} \\
&\propto \exp\left\{-\frac{1}{2}\left[(\mathbf{y}^T-\beta^T\mathbf{X}^T)(\mathbf{y}-\mathbf{X}\beta)+\beta^T\lambda\mathbf{I}\beta\right]\right\} \\
&\propto \exp\left\{-\frac{1}{2}\left[(\mathbf{y}^T\mathbf{y}-\mathbf{y}^T\mathbf{X}\beta-\beta^T\mathbf{X}^T\mathbf{y}+\beta^T\mathbf{X}^T\mathbf{X}\beta)+\beta^T\lambda\mathbf{I}\beta\right]\right\} \\
&\propto \exp\left\{-\frac{1}{2}\mathbf{y}^T\mathbf{y}\right\}\exp\left\{-\frac{1}{2}\left[-\mathbf{y}^T\mathbf{X}\beta-\beta^T\mathbf{X}^T\mathbf{y}+\beta^T\mathbf{X}^T\mathbf{X}\beta+\beta^T\lambda\mathbf{I}\beta\right]\right\} \\
&\propto \exp\left\{-\frac{1}{2}\left[-\mathbf{y}^T\mathbf{X}\beta-\beta^T\mathbf{X}^T\mathbf{y}+\beta^T\mathbf{X}^T\mathbf{X}\beta+\beta^T\lambda\mathbf{I}\beta\right]\right\} \qquad (1)
\end{aligned}
$$

Analysis:
$\mathbf{y}^T\mathbf{X}\beta = (\beta^T\mathbf{X}^T\mathbf{y})^T$, $\mathbf{y}$ is a $n\times 1$ vector, $\mathbf{X}$ is a $n\times p$ matrix, β is a $p\times 1$ vector;
$\mathbf{y}^T\mathbf{X}\beta : (1\times n)(n\times p)(p\times 1) = 1\times 1$ which implies $\mathbf{y}^T\mathbf{X}\beta$ is a scalar, and the transpose of a scalar is itself, therefore $\beta^T\mathbf{X}^T\mathbf{y} = \mathbf{y}^T\mathbf{X}\beta$

$$
\begin{aligned}
&\propto \exp\left\{-\frac{1}{2}\left[-2\beta^T\mathbf{X}^T\mathbf{y}+\beta^T\mathbf{X}^T\mathbf{X}\beta+\beta^T\lambda\mathbf{I}\beta\right]\right\} \\
&\propto \exp\left\{-\frac{1}{2}\left[-2\beta^T\mathbf{X}^T\mathbf{y}+\beta^T(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})\beta\right]\right\}
\end{aligned}
$$

1.c

Answer

Now we start from the equation (1) which was derived in the middle of the proof from problem 1.b

$$
\begin{aligned}
Pr(\beta|\mathbf{y},\mathbf{X},\lambda) &\propto \exp\left\{-\frac{1}{2}\left[-\mathbf{y}^T\mathbf{X}\beta-\beta^T\mathbf{X}^T\mathbf{y}+\beta^T\mathbf{X}^T\mathbf{X}\beta+\beta^T\lambda\mathbf{I}\beta\right]\right\}\\
&\propto \exp\left\{-\frac{1}{2}\left[-\mathbf{y}^T\mathbf{X}\beta-\beta^T\mathbf{X}^T\mathbf{y}+\beta^T(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})\beta\right]\right\}\\
&\propto \exp\left\{-\frac{1}{2}\left[-\mathbf{y}^T\mathbf{X}\beta-\beta^T\mathbf{X}^T\mathbf{y}+\beta^T(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})\beta\right]\right\}\exp\left\{-\frac{1}{2}\left[\mathbf{y}^T\mathbf{X}(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}\right]\right\}\\
&\propto \exp\left\{-\frac{1}{2}\left[-\mathbf{y}^T\mathbf{X}\beta-\beta^T\mathbf{X}^T\mathbf{y}+\beta^T(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})\beta+\mathbf{y}^T\mathbf{X}(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}\right]\right\}\\
&\propto \exp\left\{-\frac{1}{2}\left[\beta^T(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})\beta-\beta^T\mathbf{X}^T\mathbf{y}-\mathbf{y}^T\mathbf{X}\beta+\mathbf{y}^T\mathbf{X}(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}\right]\right\}\\
&\propto \exp\left\{-\frac{1}{2}\left[\beta^T(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})\left(\beta-(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}\right)-\mathbf{y}^T\mathbf{X}\left(\beta-(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}\right)\right]\right\}\\
&\propto \exp\left\{-\frac{1}{2}\left[\left(\beta^T(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})-\mathbf{y}^T\mathbf{X}\right)\left(\beta-(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}\right)\right]\right\}\\
&\propto \exp\left\{-\frac{1}{2}\left[\left(\beta^T-\mathbf{y}^T\mathbf{X}(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\right)\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)\left(\beta-(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}\right)\right]\right\}\\
&\propto \exp\left\{-\frac{1}{2}\left[\left(\beta-\left[(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\right]^T\mathbf{X}^T\mathbf{y}\right)^T\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)\left(\beta-(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}\right)\right]\right\}
\end{aligned}
$$

Analysis:
$\mathbf{X}^T\mathbf{X}$ is a symmetric matrix because $\left(\mathbf{X}^T\mathbf{X}\right)^T=\mathbf{X}^T\mathbf{X}$
$\lambda\mathbf{I}$ is also a symmetric matrix because $\mathbf{I}$ is an identity matrix
The sum of two symmetric matrices is also a symmetric matrix;
Thus we have the following:

$$\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)^{-1}=\left[\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)^{-1}\right]^T=\mathbf{I}$$

$$\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)^{-1}=\left[\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)^{-1}\right]^T\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)^T$$

Since we have $\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)^{-1}=\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)^{-1}\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)$ and $\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)^T=\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)$:

$$\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)^{-1}\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)=\left[\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)^{-1}\right]^T\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)$$

Multiply both sides by $\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)^{-1}$ we have:

$$\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)^{-1}=\left[\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)^{-1}\right]^T$$

$$\propto \exp\left\{-\frac{1}{2}\left[\left(\beta-(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}\right)^T\left(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I}\right)\left(\beta-(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}\right)\right]\right\}$$

1.d

Answer

The distribution is multivariate normal distribution with $\mu=(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$; $\mathbf{\Sigma}=(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}$;
The normal distribution reaches its maximum when β equal the mean of the distribution:

$$\beta^*=(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{y}$$

2. Statistical Learning Theory: Sub-Gaussian Bounds

2.a

Answer

$$\Pr_{X\sim\mathcal{D}}(X > t) = \Pr(e^{sX} > e^{st}) \leq \frac{\mathbb{E}[e^{sX}]}{e^{st}} = \frac{e^{\frac{\sigma^2 s^2}{2}}}{e^{st}} = e^{\frac{\sigma^2 s^2}{2} - st}$$

2.b

Answer

To make the tightest possible bound, we need make $e^{\frac{\sigma^2 s^2}{2} - st}$ smallest possible, which means we shoule make $\frac{\sigma^2 s^2}{2} - st$ smallest possible since e^x is the monotonic increaseing function.

$$\begin{aligned}
\phi(s) &= \frac{\sigma^2 s^2}{2} - st \\
\phi'(s) &= \sigma^2 s - t = 0 \\
s^* &= \frac{t}{\sigma^2} \\
\phi(s^*) &= \frac{t^2}{2\sigma^2} - \frac{t^2}{\sigma^2} = -\frac{t^2}{2\sigma^2} \\
\Pr_{X\sim\mathcal{D}}(X > t) &\leq e^{-\frac{t^2}{2\sigma^2}}
\end{aligned}$$

2.c

Answer

Analysis:

Given $X_1, \dots, X_n$ are n independent random variables and the functions of r.v (e^{sX_i}) are also independent, therefore:

$$\mathbb{E}[e^{sX_1}e^{sX_2}\dots e^{sX_n}] = \mathbb{E}[e^{X_1}]\mathbb{E}[e^{X_2}]\dots\mathbb{E}[e^{X_n}]$$

$$\mathbb{E}[e^{sX_i}] \le e^{\frac{\sigma^2 s^2}{2}}$$

$$\mathbb{E}[e^{X_1}]\mathbb{E}[e^{X_2}]\dots\mathbb{E}[e^{X_n}] \le e^{\frac{\sigma^2 s^2}{2}} e^{\frac{\sigma^2 s^2}{2}} \dots e^{\frac{\sigma^2 s^2}{2}}$$

$$\mathbb{E}[e^{s\sum_i^n X_i}] = \mathbb{E}[e^{sX_1}e^{sX_2}\dots e^{sX_n}] = \mathbb{E}[e^{X_1}]\mathbb{E}[e^{X_2}]\dots\mathbb{E}[e^{X_n}] \le e^{\frac{\sigma^2 s^2}{2}} e^{\frac{\sigma^2 s^2}{2}} \dots e^{\frac{\sigma^2 s^2}{2}} \le e^{\frac{n\sigma^2 s^2}{2}}$$

$$\mathbb{E}[e^{s\sum_i^n X_i}] \le e^{\frac{n\sigma^2 s^2}{2}}$$

$$\Pr(\frac{1}{n}\sum_{i=1}^n X_i > t) = \Pr(\sum_{i=1}^n X_i > nt) = \Pr(e^{s\sum_i X_i} > e^{snt}) \le \frac{\mathbb{E}[e^{s\sum_i^n X_i}]}{e^{nst}} \le \frac{e^{\frac{n\sigma^2 s^2}{2}}}{e^{nst}} = e^{\frac{n\sigma^2 s^2}{2} - nst}$$

$$\phi(s) = \frac{\sigma^2 n s^2}{2} - snt$$
$$\phi'(s) = \sigma^2 ns - nt = 0$$
$$s^* = \frac{t}{\sigma^2}$$
$$\phi(s^*) = \frac{nt^2}{2\sigma^2} - \frac{nt^2}{\sigma^2} = -\frac{nt^2}{2\sigma^2}$$

$$\Pr(\frac{1}{n}\sum_{i=1}^n X_i > t) \le e^{\phi(s^*)} \le e^{-\frac{nt^2}{2\sigma^2}}$$

3 K-Means as an Optimization Problem

3.a

Answer

Objective function: $\sum_i \min_k ||x_i - c_k||_2^2$

3.b

Answer

$L(\tilde{\mathbf{x}}^{(t)}) = \sum_i (x_i - \tilde{x}_k^{(t)})^2$ for x_i belongs to C_k

for the first step, we assign the the x to the cloest centroid $\tilde{x}^{(t)}$ which makes the objective:

$$L(\tilde{\mathbf{x}}^{(t)}) = \sum_i \min_k (x_i - \tilde{x}_k^{(t)})^2$$

thus we have:

$$\sum_i \min_k (x_i - \tilde{x}_k^{(t)})^2 \le \sum_i (x_i - \tilde{x}_k^{(t)})^2$$

Assign the new centroid based on the mean of the new clusters, which is $\tilde{\mathbf{x}}_k^{(t+1)}$ for each cluster we have x_i for $x_i \in C_k$, we want find:

$$\arg\min \sum_i ||x_i - z||^2$$

Take the derivative and set it to 0, we can get:

$$2\sum_i^n x_i - z = 0$$

$$z = \frac{\sum_i x_i}{n} = \bar{x}$$

thus we can conclude the distance between the mean of a cluster of data points and every single data point in that cluster is smallest, since $\tilde{\mathbf{x}}_k^{(t+1)}$ is the center of the new clusters formed in t+1 iteration, we have:

$$\sum_i (x_i - \tilde{x}_k^{(t+1)})^2 \le \sum_i \min_k (x_i - \tilde{x}_k^{(t)})^2 \le \sum_i (x_i - \tilde{x}_k^{(t)})^2$$

Given the proof shown above, it is clear now our objective function is monotonic decreasing, and since the function is also non-negative,which means it is bounded, therefore, K-means will converge

3.c

Answer

Data set: suppose we have four one-dimensional data points: $x_1 = 1, x_2 = 2, x_3 = 5, x_4 = 7$. and k = 3 is asked.
(dataset credited to Discussion8 question 2.2)

Global optimal solution: cluster 1 $\{x_1, x_2\}$ with center $z_1 = 1.5$; cluser 2 $\{x_3\}$ with center $z_2 = 5$; cluster 3 $\{x_4\}$ with center $z_3 = 7$
Initialize centers using $z_1 = x_1, z_2 = x_2, z_3 = x_3$, therefore we have clusters:

$$\{x_1\}; \{x_2\}; \{x_3, x_4\}$$

then we find out the new centers for each cluster:

$$z_1^{(1)} = x_1 = 1; z_2^{(1)} = x_2 = 2; z_3^{(1)} = \frac{x_3 + x_4}{2} = 6$$

if we assign each point to the new centers we would still get the same clusters because x_3 and x_4 still sit cloest to the new cluster center $z_3 = 6$, so it has no further update, thus we can never reach the optimal solution by this initialization.

3.d

Answer

K means initialize the starting centorids randomly and then in the first step, and secondly gather the points to form clusters based on their distance between those centorids, lastly, k means will find out their new centers. Step 2 and 3 are repeated until convergence. In the very first step, we locally choose centarin amount of points to be the centers and then we choose x_i which $\min \sum_i (x_i - C_k)^2$, so what we are doing is to locally optimize the cost function, and in the later step, we are going to find the centers in the new clusters, to be specifically, $\min \sum_i ||x - z||^2$, which we are assume the other points are fixed and we only consider the minimization in the one cluster in those steps. Therefore, the algorithm cannot gauarantee global optimality.

For Gussian Mixture, the objective is to maximize the observed data log-likelihood, which is $\sum_i \log \sum_K p(X_i = x_i | z_i = k, \theta) p(z_i = k|\theta)$, however EM tends to maximizes the auxiliary function which is $\sum_i \sum_k p(z_i = k|x_i, \theta_t) \log \frac{p(X_i = x_i, z_i = k|\theta)}{p(z_i = k|x_i, \theta_t)}$. The auxiliary function is obtained by the Jensen's Inequality which indicates log-likelihood(θ) $\geq$ A(θ, θ_t), therefore, EM cannot guarantee Gaussian Mixture Model always find the globally solution.

4 EM Algorithm for Topic Modeling

4.a

Answer

$$
\begin{aligned}
\log[Pr(D=d_1, D=d_2, ..D=d_i, W=w_1, W=w_2, \dots W=w_i|\alpha,\beta)] &= \log\left[\prod_i^N \prod_j^W \Big(\sum_k Pr(d_i, z_k, w_j)\Big)^{n(w_j,d_i)}\right] \\
&= \log\left[\prod_i^N \prod_j^W \Big(\sum_k Pr(d_i)Pr(z_k|d_i)Pr(w_j|z_k)\Big)^{n(w_j,d_i)}\right] \\
&= \log\left[\prod_i^N \prod_j^W \Big(\sum_k \frac{1}{M}\alpha_{ik}\beta_{kj}\Big)^{n(w_j,d_i)}\right] \\
&= \sum_i^M \sum_j^W n(w_j, d_i)\log\left[\frac{1}{M}\sum_k \alpha_{ik}\beta_{kj}\right]
\end{aligned}
$$

4.b

Answer

$$
Pr(z_k|w_j, d_i, \alpha^{old}, \beta^{old}) = \frac{Pr(d_i, z_k, w_j|\alpha^{old}\beta^{old})}{Pr(d_i, w_j|\alpha^{old}\beta^{old})} = \frac{\frac{1}{M}\alpha^{old}\beta^{old}}{\sum_k \frac{1}{M}\alpha_{ik}^{old}\beta_{kj}^{old}} = \frac{\alpha_{ik}^{old}\beta_{kj}^{old}}{\sum\limits_k \alpha_{ik}^{old}\beta_{kj}^{old}}
$$

Let:

$$
\gamma_{ijk} = \frac{\alpha_{ik}^{old}\beta_{kj}^{old}}{\sum\limits_k \alpha_{ik}^{old}\beta_{kj}^{old}}
$$

4.c

Answer

$\theta = (\alpha, \beta)$
$\theta_t = (\alpha_t, \beta_t)$

$A(\theta, \theta_t) = \sum_i \sum_j n(d_i, w_j) \sum_k \gamma_{ijk} \log \dfrac{Pr(d_i, z_k, w_j)}{\gamma_{ijk}}$

$L(\alpha_{ik}, \beta_{kj}) = A(\theta, \theta_t) + \lambda_1(1 - \Sigma_k \alpha_{ik}) + \lambda_2 \sum_j (1 - \beta_{kj})$

$$\begin{aligned}\frac{\partial L(\theta, \theta_t)}{\partial \alpha_{i'k'}} &= \frac{\partial A(\theta, \theta_t)}{\partial \alpha_{i'k'}} - \lambda_1 \\ &= n(w_j, d_i) \sum_j \frac{\gamma_{i'jk'}}{\alpha_{i'k'}} - \lambda_1 \\ &= 0\end{aligned}$$

$\alpha_{i'k'} = \dfrac{n(w_j, d_i) \sum_j \gamma_{i'jk'}}{\lambda_1}$

Given: $\sum_k \alpha_{ik} = 1$

$$\sum_k \frac{n(w_j, d_i) \sum_j \gamma_{ijk}}{\lambda_1} = 1$$

$$\sum_k \sum_j n(w_j, d_i) \gamma_{ijk} = \lambda_1$$

$$\sum_j n(w_j, d_i) \sum_k Pr(z_k | w_j, d_i, \alpha^{old}, \beta^{old}) = \lambda_1$$

$$\sum_j n(w_j, d_i) \times 1 = \lambda_1$$

$$\lambda_1 = n(d_i)$$

$$\alpha_{i'k'} = \frac{\sum_j^N n(w_j, d_i)\gamma_{i'jk'}}{n(d_i)}$$

$$\begin{aligned}\frac{\partial L(\theta, \theta_t)}{\partial \beta_{k'j'}} &= \frac{\partial A(\theta, \theta_t)}{\partial \alpha_{k'j'}} - \lambda_2 \\ &= \sum_i n(w_j, d_i) \frac{\gamma_{ij'k'}}{\beta_{k'j'}} - \lambda_2 \\ &= 0\end{aligned}$$

$\beta_{k'j'} = \sum_i n(w_j, d_i) \dfrac{\gamma_{ij'k'}}{\lambda_2}$

Given: $\sum_j \beta_{kj} = 1$

$$\sum_j \frac{\sum_i n(w_j, d_i)\gamma_{ijk}}{\lambda_2} = 1$$

$$\sum_j \sum_i n(w_j, d_i)\gamma_{ijk} = \lambda_2$$

$$\beta_{k'j'} = \frac{\sum_i n(w_j, d_i)\gamma_{ij'k'}}{\sum_j \sum_i n(w_j, d_i)\gamma_{ijk}}$$

5. What Happens to My Gradient

5.a

define: $\mathbf{z}_l = \mathbf{W}_l^T \mathbf{h}_{l-1} + \mathbf{b_l}$

$1.\nabla_{\mathbf{w}_3} \mathcal{L}(\theta) = \nabla_{\mathbf{z}_3} \mathcal{L}(\theta) \dfrac{\partial z_3}{\partial \mathbf{w}_3}$

where we have:

$$\frac{\partial \mathcal{L}(\theta)}{\partial f(\mathbf{x_i})} = -\sum_{i=1}^{N} \frac{y_i}{f(\mathbf{x}_i)} + \frac{y_i - 1}{1 - f(\mathbf{x}_i)}$$

$$\frac{\partial f(\mathbf{x_i})}{\partial z_3} = \sigma(z_3)(1 - \sigma(z_3))$$

$$\nabla_{\mathbf{z}_3} \mathcal{L}(\theta) = \frac{\partial \mathcal{L}(\theta)}{\partial f(\mathbf{x_i})} \frac{\partial f(\mathbf{x_i})}{\partial z_3} = -\sum_{i=1}^{N} \Big[\frac{y_i}{f(\mathbf{x}_i)} + \frac{y_i - 1}{1 - f(\mathbf{x}_i)}\Big]\sigma(z_3)(1 - \sigma(z_3))$$

$$\frac{\partial z_3}{\partial \mathbf{w}_3} = \mathbf{h}_2^{\mathrm{T}}$$

$2.\nabla_{b_3} \mathcal{L}(\theta) = \nabla_{\mathbf{z}_3} \mathcal{L}(\theta) \dfrac{\partial z_3}{\partial b_3}$

where we have:

$$\frac{\partial \mathcal{L}(\theta)}{\partial f(\mathbf{x_i})} = -\sum_{i=1}^{N} \frac{y_i}{f(\mathbf{x}_i)} + \frac{y_i - 1}{1 - f(\mathbf{x}_i)}$$

$$\frac{\partial z_3}{\partial \mathbf{b}_3} = 1$$

$3.\nabla_{\mathbf{W}_2} \mathcal{L}(\theta) = \nabla_{\mathbf{z}_2} \mathcal{L}(\theta) \dfrac{\partial \mathbf{z}_2}{\partial \mathbf{W}_2}$

where we have:

$$\nabla_{\mathbf{z}_2} \mathcal{L}(\theta) = \nabla_{\mathbf{z}_3} \mathcal{L}(\theta) \frac{\partial \mathbf{z}_3}{\partial \mathbf{z}_2}$$

$$\frac{\partial \mathbf{z}_3}{\partial \mathbf{z}_2} = \mathbf{w}_3 \frac{d\sigma(\mathbf{z}_2)}{d\mathbf{z}_2} = \mathbf{w}_3 \circ \sigma(\mathbf{z}_2) \circ (\mathbf{1} - \sigma(\mathbf{z_2}))$$

Note: let $\circ$ to be the element-wise product of two vectors.

$$\nabla_{\mathbf{z}_2} \mathcal{L}(\theta) = \nabla_{\mathbf{z}_3} \mathcal{L}(\theta) \circ \mathbf{w}_3 \circ \sigma(\mathbf{z}_2) \circ (\mathbf{1} - \sigma(\mathbf{z_2}))$$

$\dfrac{\partial \mathbf{z}_2}{\partial \mathbf{W}_2} = \begin{bmatrix} \\ \mathbf{h}_1^T \\ \\ \end{bmatrix}$: this is a matrix constructed by H rows of $\mathbf{h}_1^{\mathrm{T}}$

$$\nabla_{\mathbf{W}_2} \mathcal{L}(\theta) = \nabla_{\mathbf{z}_3} \mathcal{L}(\theta_3) \circ \mathbf{w}_3 \circ \sigma(\mathbf{z}_2) \circ (\mathbf{1} - \sigma(\mathbf{z_2})) \circ \begin{bmatrix} \\ \mathbf{h}_1^T \\ \\ \end{bmatrix}$$

4.$\nabla_{\mathbf{b}_2}\mathcal{L}(\theta) = \nabla_{\mathbf{z}_2}\mathcal{L}(\theta)\frac{\partial \mathbf{z}_2}{\partial \mathbf{b}_2}$

where we have:

$\frac{\partial \mathbf{z}_2}{\partial \mathbf{b}_2} = \mathbf{I}$

Note: $\mathbf{I}$ is a $H \times H$ identity matrix

5.$\nabla_{\mathbf{W}_1}\mathcal{L}(\theta) = \nabla_{\mathbf{z}_1}\mathcal{L}(\theta)\frac{\partial \mathbf{z}_1}{\partial \mathbf{W}_1}$

where we have:

$\nabla_{\mathbf{z}_1}\mathcal{L}(\theta) = \nabla_{\mathbf{z}_2}\mathcal{L}(\theta)\frac{\partial \mathbf{z}_2}{\partial \mathbf{z}_1}$

$\frac{\partial \mathbf{z}_2}{\partial \mathbf{z}_1} = \mathbf{W}_2\frac{d\sigma(\mathbf{z}_1)}{d\mathbf{z}_1} = \mathbf{W}_2 \circ \sigma(\mathbf{z}_1) \circ (\mathbf{1} - \sigma(\mathbf{z}_1))$

$\nabla_{\mathbf{z}_1}\mathcal{L}(\theta) = \nabla_{\mathbf{z}_2}\mathcal{L}(\theta)\mathbf{W}_2 \circ \sigma(\mathbf{z}_1) \circ (\mathbf{1} - \sigma(\mathbf{z}_1))$

$\frac{\partial \mathbf{z}_1}{\partial \mathbf{W}_1} = \begin{bmatrix} \\ \mathbf{x}_i^T \\ \\ \end{bmatrix}$:this is a matrix constructed by H rows of $\mathbf{x}_i$ $\nabla_{\mathbf{W}_1}\mathcal{L}(\theta) = \nabla_{\mathbf{z}_2}\mathcal{L}(\theta_3)\mathbf{w}_2 \circ \sigma(\mathbf{z}_1) \circ (\mathbf{1} - \sigma(\mathbf{z}_1)) \circ \begin{bmatrix} \\ \mathbf{x}_i^T \\ \\ \end{bmatrix}$

6.$\nabla_{\mathbf{b}_1}\mathcal{L}(\theta) = \nabla_{\mathbf{z}_1}\mathcal{L}(\theta)\frac{\partial \mathbf{z}_1}{\partial \mathbf{b}_1}$

where we have:

$\frac{\partial \mathbf{z}_1}{\partial \mathbf{b}_1} = \mathbf{I}$

Note: $\mathbf{I}$ is a $H \times H$ identity matrix

5.b

In [48]:
```
%reset -f
from scipy.io import loadmat
All_data = loadmat('mnist_all.mat')
train0  =All_data['train0']
test0 = All_data['test0']
train1 = All_data['train1']
test1 = All_data['test1']
```

In [106]:
```
import numpy as np
from sklearn.utils import resample
class NN:
    def __init__(self,d):
        self.W = []
        self.b = []
        self.d = d
        for i in range(d):
            if i == 0:
                w = 0.001*np.random.randn(4,784)
            else:
                w = 0.01*np.random.randn(4,4)
            b = 0.01*np.random.randn(4)
            self.W.append(w)
            self.b.append(b)
        self.W.append(0.01*np.random.randn(4))
        self.b.append(0.01*np.random.randn())

    def activation(self,z):
        return 1/(1+np.exp(-z))
    def forward(self,x):
        h = x
        All_h = []
        for i in range(len(self.W)):
            h =  self.W[i]@h + self.b[i]
            if i !=len(self.W)-1:
                for k, p in enumerate(h):
                    h[k]= self.activation(p)
            else:
                h =self.activation(h)
            All_h.append(h)
        return All_h

    def loss(self,X):
        loss = 0
        for i in X:
            training_features = i[:-1]
            y = i[-1]
            predict = self.forward(training_features)[-1]
            loss = loss - (y*np.log(predict) + (1-y)*np.log(1-predict))
        print("loss is ", loss)
        return loss
    def backpropagate(self,mini_batch):
        D_w = []
        D_b = []
        for i in range(len(self.W)):
            d_w = np.zeros_like(self.W[i])
            D_w.append(d_w)
            d_b = np.zeros_like(self.b[i])
            D_b.append(d_b)
        for training_sample in mini_batch:
            layer_output = self.forward(training_sample[:-1])
            n =  len(layer_output)
            y = training_sample[-1]
            gradient =[]
            gradient_zd =-(y/layer_output[n-1] +(y-1)/(1-layer_output[n-1]))*layer_output[n-1]*(1-layer_output[n-1])
            gradient.append(gradient_zd)
            for i in range(self.d,1,-1):
                if i == self.d:
                    gradient_z = gradient_zd*self.W[i] * layer_output[i-1]*(1-layer_output[i-1])
                else:
                    gradient_z = np.dot(gradient_zd,self.W[i]) * layer_output[i-1]*(1-layer_output[i-1])
                gradient.append(gradient_z)
                gradient_zd = gradient_z
            gradient.append(np.dot(gradient_zd,self.W[1])* layer_output[0]*(1-layer_output[0]))

            d_w_l = []
            d_b_l = []
            d_wd = gradient[0] * layer_output[self.d-1]
            d_w_l.append(d_wd)

            for i in range(self.d+1):
                d_b_l.append(gradient[i])       ##[b_d+1, b_d .... b_1]
            for i in range(1,len(gradient)):
                if i < len(gradient) - 1:
                    d_w_i = np.array([layer_output[i-1],]*4)
                    for j in range(gradient[i].shape[0]):
                        d_w_i[j] = gradient[i][j] * d_w_i[j]
                else:
                    d_w_i = np.array([training_sample[:-1],]*4)
                    for k in range(gradient[i].shape[0]):
                        d_w_i[k] =  gradient[i][k]*d_w_i[k]
#                           print("gradient z1 element", gradient[i][k])
                d_w_l.append(d_w_i)

            for i in range(len(d_w_l)):
                D_w[i] = D_w[i]+d_w_l[-(i+1)]
                D_b[i] = D_b[i]+d_b_l[-(i+1)]
        return D_w,D_b


    def train(self,X):
            i= 0
            loss = 1000
            l = 0
            while loss > 300:
                print("train iteration:",l+1)
                mini_batch = resample(X,n_samples = 200)
                D_w, D_b= self.backpropagate(mini_batch)
                for i in range(len(self.W)):
                    self.W[i] = self.W[i] - 0.1 *D_w[i]/200
                    self.b[i] = self.b[i] -  0.1 *D_b[i]/200
                loss = self.loss(X)
                l = l+1

    def prediction(self,X):
        y = []
        for x in X:
            all_h = self.forward(x)
            if all_h[2] > 0.5:
                y.append(1)
            else:
                y.append(0)
        return y
```

```
In [103]: c=NN(2)
          c.train(mixed_train)
```

```
train iteration: 1
loss is  8752.43442903357
train iteration: 2
loss is  8751.675910214275
train iteration: 3
loss is  8779.547305636424
train iteration: 4
loss is  8754.644557241958
train iteration: 5
loss is  8748.674221975412
train iteration: 6
loss is  8750.0424323202
train iteration: 7
loss is  8748.512447547539
train iteration: 8
loss is  8742.51841954908
train iteration: 9
loss is  8740.128056191892
train iteration: 10
loss is  8742.222995757389
```

```
In [116]: P0 = c.prediction(test0)
          P1 = c.prediction(test1)
          p_train0 = c.prediction(train0)
          p_train1 =c.prediction(train1)
          accuracy_0 = sum(P0 - np.zeros(test0.shape[0]))/test0.shape[0]
          print("The test accuracy on digit 0 is", 1-accuracy_0)
          accuracy_0 = sum(p_train0 - np.zeros(train0.shape[0]))/train0.shape[0]
          print("The training accuracy on digit 0 is", 1-accuracy_0)
          accuracy_1 = sum(np.ones(test1.shape[0]) - P1)/test1.shape[0]
          print("The test accuracy on digit 1 is ", 1- accuracy_1)
          accuracy_1 = sum(np.ones(train1.shape[0]) - p_train1)/train1.shape[0]
          print("The training accuracy on digit 1 is ", 1- accuracy_1)
```

```
/Users/Yiteng/anaconda3/lib/python3.7/site-packages/ipykernel_launcher.py:20: RuntimeWarning: overflow encountered i
n exp
```

```
The test accuracy on digit 0 is 0.996938775510204
The training accuracy on digit 0 is 0.9969609994934999
The test accuracy on digit 1 is  1.0
The training accuracy on digit 1 is  0.9970335212103233
```

5.c

```
In [128]: ### d =  1
          mini_batch = resample(mixed_train,n_samples=200)
          c1 = NN(1)
          g_w1,_ = c1.backpropagate(mini_batch)
          c2 = NN(2)
          g_w2,_=c2.backpropagate(mini_batch)
          c3 = NN(3)
          g_w3,_=c3.backpropagate(mini_batch)
          c4 = NN(4)
          g_w4,_=c4.backpropagate(mini_batch)
```

```
In [129]: print("1-layer w_1 gradient:",np.linalg.norm(g_w1[0]))
          print("2-layer w_1 gradient:",np.linalg.norm(g_w2[0]))
          print("3-layer w_1 gradient:",np.linalg.norm(g_w3[0]))
          print("4-layer w_1 gradient:",np.linalg.norm(g_w4[0]))
```

```
1-layer w_1 gradient: 148.64171167643602
2-layer w_1 gradient: 0.904329701884857
3-layer w_1 gradient: 0.005497313162780188
4-layer w_1 gradient: 0.00011249048769731869
```

The magnitude of w1 is decreasing when we increase the number of the hidden layers

5.d

$$\nabla \mathbf{W}_1 \mathcal{L}(\theta) = \frac{\partial \mathcal{L}(\theta)}{\partial f(x)} \frac{\partial f(x)}{\partial z_l} \frac{\partial z_l}{\partial z_{l-1}} \frac{\partial z_{l-1}}{\partial z_{l-2}} \cdots \frac{\partial z_1}{\partial \mathbf{w}_1}$$

We have:

$$\frac{\partial z_l}{\partial z_{l-1}} = \mathbf{W}_l \circ \sigma(\mathbf{z}_l) \circ (\mathbf{1} - \sigma(\mathbf{z}_l))$$

Analysis:

Let $\Sigma = \sigma(\mathbf{z}_l) \circ (\mathbf{1} - \sigma(\mathbf{z}_l))$ is Hadamard product which the result is a vector in which element is $\sigma(z_{li})(1 - \sigma(z_{li}))$
Now, we take few steps to see the upper bound of the sigmoid derivative:

$$\sigma(z)(1 - \sigma(z)) = \frac{e^{-x}}{(1 + e^{-x})^2}$$

Take the derivative of $\sigma(z)(1 - \sigma(z))$ and set it to 0 , we can get:

$$\frac{e^x(e^x - 1)}{(1 + e^x)^3} = 0$$
$$e^x = 0 \ e^x = 1$$

$e^x = 1$ is the upper bound for the sigmoid derivative, therefore, we have $\sigma(z)(1 - \sigma(z)) \le \frac{1}{4}$, which means every element in that vector is bounded by $\frac{1}{4}$

Back to $\mathbf{W}_l \circ \Sigma$, this equation shows that every row of $\mathbf{W}_l$ would multiply by the element in Σ, which essential means $\mathbf{W}$ is multiplying a diagonal matrix in which every entry on the diagonal is an element in Σ

$$\frac{\partial z_l}{\partial z_{l-1}} = \mathbf{W}_l \cdot \mathrm{diag}(\Sigma)$$

Use one of the basic matrix norm inequality we have:

$$\left\| \frac{\partial z_l}{\partial z_{l-1}} \right\| = \left\| \mathbf{W}_l \cdot \mathrm{diag}(\Sigma) \right\| \le \left\| \mathbf{W}_l \right\| \left\| \mathrm{diag}(\Sigma) \right\|$$

Given the norm of diagonal matrix is bounded by the maximum entry on the diagonal, so it is bounded by $\frac{1}{4}$

Since all the W are diagonalizable, so they can all be written in the eigenvector decomposition form. $W = P\Lambda P^{-1}$ We can also have the $W^T = (P^{-1})^T \Lambda P^T$, here we define P as the matrix constructed by the eigenvectors with orthnormal basis. Therefore, according to the definition of L2 norm for the matrix:

$$||A||_2 = \sqrt{\lambda_{max}(A^T A)} = \sqrt{\lambda_{max}(P\Lambda^2 P^T)} = \sigma_{max}(A)$$

The eigenvalues are embedded in Λ,the maximum singular value is the absolute value of the max eigenvalue. Therefore $||A||_2$ is bounded by $4 - \epsilon$ so we have:

$$\left\| \frac{\partial z_l}{\partial z_{l-1}} \right\| \le \frac{1}{4} * (4 - \epsilon) \le 1$$

$$\begin{aligned} \left\| \nabla \mathbf{W}_1 \mathcal{L}(\theta) \right\|_{d->\infty} &= \left\| \frac{\partial \mathcal{L}(\theta)}{\partial f(x)} \frac{\partial f(x)}{\partial z_l} \frac{\partial z_l}{\partial z_{l-1}} \frac{\partial z_{l-1}}{\partial z_{l-2}} \cdots \frac{\partial z_1}{\partial \mathbf{w}_1} \right\| \\ &\le \left\| \frac{\partial \mathcal{L}(\theta)}{\partial f(x)} \right\| \left\| \frac{\partial f(x)}{\partial z_l} \right\| \left\| \frac{\partial z_l}{\partial z_{l-1}} \right\| \left\| \frac{\partial z_{l-1}}{\partial z_{l-2}} \right\| \cdots \left\| \frac{\partial z_1}{\partial \mathbf{w}_1} \right\| \end{aligned}$$

if d goes into infinity, then there are infinite numeber of layers, which means infinite numbers less than 1 would be multiplied together, then the overall result goes to 0; if the norm of $\nabla \mathbf{W}_1 \mathcal{L}(\theta)$ is zero, we can conclude that:

$$\lim_{d \to \infty} \nabla_{\mathbf{W}_1} \mathcal{L}(\theta) = \mathbf{0}$$