机器学习导论 习题四

211300063, 张运吉, 211300063@smail.nju.edu.cn

2023 年 5 月 16 日

作业提交注意事项

1. 请在 LaTeX 模板中第一页填写个人的学号、姓名、邮箱;
2. 本次作业需提交作答后的该 pdf 文件、编程题 .ipynb 文件; **请将二者打包为 .zip 文件上传**. 注意命名规则, 三个文件均命名为“学号_姓名”+“. 后缀”(例如 211300001_张三”+“.pdf”、“.ipynb”、“.zip”);
3. 若多次提交作业, 则在命名 .zip 文件时加上版本号, 例如 211300001_张三_v1.zip”(批改时以版本号最高的文件为准);
4. 本次作业提交截止时间为 **5 月 24 日 23:59:59**. 未按照要求提交作业, 提交作业格式不正确, **作业命名不规范**, 将会被扣除部分作业分数; 除特殊原因 (如因病缓交, 需出示医院假条) 逾期未交作业, 本次作业记 0 分; **如发现抄袭, 抄袭和被抄袭双方成绩全部取消**;
5. 本次作业提交地址为 here, 请大家预留时间提前上交, 以防在临近截止日期时, 因网络等原因无法按时提交作业.

1 [15pts] Vanishing Gradient Problem

在使用梯度下降与反向传播训练深度神经网络时, 可能会出现梯度消失的问题, 即网络参数的梯度非常小, 导致网络更新非常缓慢, 甚至停止更新. 该问题的成因较为复杂, 有很多因素可能导致该问题的出现. 本题将主要讨论激活函数与该问题之间的联系.

(1) **[5pts]** 当在深度神经网络中采用 Sigmoid 激活函数时, 网络训练容易出现梯度消失问题. 为分析此现象, 请先求解 Sigmoid 导函数的值域, 并根据该范围, 进一步对该现象进行分析.

(2) **[5pts]** 当前深度神经网络大多采用 ReLU 激活函数, 试分析相较于 Sigmoid, ReLU 对梯度消失问题的缓解作用, 同时思考其可能带来的一些问题.

(3) **[5pts]** 请从激活函数之外的角度, 列举三项缓解梯度消失问题的措施.

Solution. 此处用于写解答 (中英文均可)

(1) $\because \text{Sigmoid}'(x) = \text{Sigmoid}(x)(1 - \text{Sigmoid}(x)), \text{Sigmoid}(x) \in (0, 1).$
$\therefore \text{Sigmoid}'(x) \in (0, 0.25)$
当神经网络的层数较多时，在反向传播过程中，每层都需要乘以导数来计算梯度。如果某些神经元的输出接近于饱和区间，即非常接近 0 或 1，那么它们对应的导数将会非常小，很容易就会出现梯度消失的现象.

(2) ReLU 的缓解作用:
ReLU 激活函数在正区间上导数始终为常数 1，因此不会出现梯度消失的情况，这是 ReLU 相较于 Sigmoid 的一个主要优势.
潜在问题:
a. 当神经元的输入小于零时，ReLU 函数的输出恒为零. 如果输入一直为负数，那么后面所有的权重更新将全都失效. 这会导致该神经元无法再次被激活，从而影响网络的表达能力.
b. ReLU 的输出并没有做归一化处理，因此其输出分布不受限制，可能会出现某些神经元输出特别大或特别小的情况。如果训练过程中发生这种情况，可能会影响模型的泛化能力，并引起过拟合等问题.

(3) 除了使用特定的激活函数之外，还可以从以下角度尝试缓解梯度消失问题：

- a. 批标准化:
  批标准化指的是对神经网络每一层的输入进行标准化。在进行批标准化时，将每个输入样本的特征都归一化到均值为 0，方差为 1 的分布上。这有助于避免某些激活函数（如 sigmoid、tanh 等）中超过饱和区域产生的梯度消失。同时，批标准化还可以作为一种正则化手段，帮助防止过拟合.
- b. 权重初始化:
  权重的过大或过小都容易导致梯度消失问题. 因此，在初始化时通常采用一些较为中性的方式来初始化权重，使得输出分布在合适的范围内. 如 xavier initialization、He initialization 等方法，会根据输入和输出的维度计算初始化权重的标准差，从而减少量纲的影响，使得输出数据的标准差更加平衡，加速模型收敛.

- c. 增加层数或者改变网络结构:
  增加网络层数可以扩展模型的表示能力，使得模型能够学习更加复杂的输入输出关系。如果网络层数不足，可能无法表达数据集中包含的潜在规律，因此也容易发生梯度消失问题. 然而，过多的层也会增加梯度消失的风险，而且还会出现训练变得更加困难、收敛速度减缓等问题. 因此，在实践中需要权衡深度和宽度，选择合适的网络结构来避免梯度消失问题。

2 [15pts] Derivation and Analysis of PCA

记中心化样本 $\mathbf{X} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n) \in \mathbb{R}^{d\times n}$ 满足 $\sum_i \boldsymbol{x}_i = \mathbf{0}$; 投影变换 $\mathbf{W} = (\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_d) \in \mathbb{R}^{d\times d}$, 每维是正交基向量, 满足 $\|\boldsymbol{w}_i\|_2 = 1, \boldsymbol{w}_i^\top \boldsymbol{w}_j = 0(\forall i \neq j)$.

(1) **[5pts]** 用拉格朗日乘子法求解 PCA 的优化问题.

$$\begin{aligned} \max_{\mathbf{W}} \quad & \operatorname{tr}\left(\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{W}\right) \\ \text{s.t.} \quad & \mathbf{W}^{\mathrm{T}}\mathbf{W} = \mathbf{I} \end{aligned}$$

(2) **[5pts]** 对于以下三个样本点: $\boldsymbol{x}_1 = (-1, 1)^{\mathrm{T}}, \boldsymbol{x}_2 = (0, -2)^{\mathrm{T}}, \boldsymbol{x}_3 = (1, 1)^{\mathrm{T}}$, 试用 (1) 中得到的结果求解最大主成分对应的 $\boldsymbol{w}_1$.

(3) **[5pts]** 设原样本 $\mathbf{X}$ 的协方差矩阵对应的 d 个特征值组成的投影变换为 $\mathbf{W}$. 考虑以下三种变换: 平移 (每个样本沿向量 $\boldsymbol{q}$ 方向移动距离 s)、放缩 (每个样本乘以放大率 α) 和旋转 (样本围绕点 $\boldsymbol{p}$ 顺时针旋转 θ). 试求解变换后的样本 $\hat{\mathbf{X}}$ 对应的 $\hat{\mathbf{W}}$.

Solution. 此处用于写解答 (中英文均可)

(1) 将原始问题写成优化问题的标准形式:

$$\begin{aligned} \min_{\mathbf{W}} \quad & -\operatorname{tr}\left(\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{W}\right) \\ \text{s.t.} \quad & \mathbf{W}^{\mathrm{T}}\mathbf{W} - \mathbf{I} = \mathbf{0}. \end{aligned}$$

此优化目标的拉格朗日函数为:

$$\begin{aligned} L(\mathbf{W}, \Theta) &= -\operatorname{tr}\left(\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{W}\right) + \left\langle \Theta, \mathbf{W}^{\mathrm{T}}\mathbf{W} - \mathbf{I}\right\rangle \\ &= -\operatorname{tr}\left(\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{W}\right) + \operatorname{tr}\left(\Theta^{\mathrm{T}}\left(\mathbf{W}^{\mathrm{T}}\mathbf{W} - \mathbf{I}\right)\right) \end{aligned}$$

其中 $\Theta \in \mathbb{R}^{d\times d}$, $\left\langle \Theta, \mathbf{W}^{\mathrm{T}}\mathbf{W} - \mathbf{I}\right\rangle = \operatorname{tr}\left(\Theta^{\mathrm{T}}\left(\mathbf{W}^{\mathrm{T}}\mathbf{W} - \mathbf{I}\right)\right)$ 为矩阵内积.
若此时仅考虑约束:$\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{w}_i = 1, i \in [d]$. 拉格朗日乘子矩阵 Θ 此时为对角矩阵, 令

$$\Theta = \Lambda = \operatorname{diag}\left(\lambda_1, \lambda_2, \ldots, \lambda_d\right)$$

可以如此处理的原因是此时 L 的第二项:

$$\operatorname{tr}\left(\Theta^{\mathrm{T}}\left(\mathbf{W}^{\mathrm{T}}\mathbf{W} - \mathbf{I}\right)\right) = \sum_{i=1}^{d} \lambda_i(\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{w}_i - 1)$$

拉格朗日函数变为:

$$L(\mathbf{W}, \Lambda) = -\operatorname{tr}\left(\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{W}\right) + \operatorname{tr}\left(\Lambda^{\mathrm{T}}\left(\mathbf{W}^{\mathrm{T}}\mathbf{W} - \mathbf{I}\right)\right)$$

对其求导:

$$\begin{aligned} \frac{\partial L(\mathbf{W}, \Lambda)}{\partial \mathbf{W}} &= \frac{\partial}{\partial \mathbf{W}}\left[-\operatorname{tr}\left(\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{W}\right) + \operatorname{tr}\left(\Lambda^{\mathrm{T}}\left(\mathbf{W}^{\mathrm{T}}\mathbf{W} - \mathbf{I}\right)\right)\right] \\ &= -\frac{\partial}{\partial \mathbf{W}}\operatorname{tr}\left(\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{W}\right) + \frac{\partial}{\partial \mathbf{W}}\operatorname{tr}\left(\Lambda^{\mathrm{T}}\left(\mathbf{W}^{\mathrm{T}}\mathbf{W} - \mathbf{I}\right)\right) \end{aligned}$$

由矩阵微分公式:

$$\frac{\partial}{\partial \mathbf{X}} \operatorname{tr}\left(\mathbf{X}^{\mathrm{T}} \mathbf{B} \mathbf{X}\right)=\mathbf{B} \mathbf{X}+\mathbf{B}^{\mathrm{T}} \mathbf{X}, \frac{\partial}{\partial \mathbf{X}} \operatorname{tr}\left(\mathbf{B} \mathbf{X}^{\mathrm{T}} \mathbf{X}\right)=\mathbf{X} \mathbf{B}^{\mathrm{T}}+\mathbf{X} \mathbf{B}$$

得到:

$$\begin{aligned}\frac{\partial L(\mathbf{W}, \Lambda)}{\partial \mathbf{W}} &=-2 \mathbf{X} \mathbf{X}^{\mathrm{T}} \mathbf{W}+\mathbf{W} \Lambda+\mathbf{W} \Lambda^{\mathrm{T}} \\ &=-2 \mathbf{X} \mathbf{X}^{\mathrm{T}} \mathbf{W}+\mathbf{W}\left(\Lambda+\Lambda^{\mathrm{T}}\right) \\ &=-2 \mathbf{X} \mathbf{X}^{\mathrm{T}} \mathbf{W}+2 \mathbf{W} \Lambda\end{aligned}$$

令 $\frac{\partial L(\mathbf{W}, \Lambda)}{\partial \mathbf{W}}=\mathbf{0}$, 可得:

$$\mathbf{X} \mathbf{X}^{\mathrm{T}} \mathbf{W}=\mathbf{W} \Lambda$$

展开可得:

$$\mathbf{X} \mathbf{X}^{\mathrm{T}} \boldsymbol{w}_i=\lambda_i \boldsymbol{w}_i, \quad i \in[d]$$

上式为矩阵特征值和特征向量的定义，$\lambda_i, \boldsymbol{w}_i$ 表示矩阵 $\mathbf{X}\mathbf{X}^{\mathrm{T}}$ 的特征值和对应的单位特征向量.

以上是仅考虑约束 $\boldsymbol{w}_i^{\mathrm{T}} \boldsymbol{w}_i=1, i \in[d]$ 的结果，还需要考虑约束 $\boldsymbol{w}_i^{\mathrm{T}} \boldsymbol{w}_j=0, i \neq j$. 因为 $\mathbf{X}\mathbf{X}^{\mathrm{T}}$ 是一个实对称矩阵，实对称矩阵的不同特征值所对应的特征向量之间相互正交，同一特征值的不同特征向量可以通过施密特正交化使其变得正交，所以通过上式求得的 $\boldsymbol{w}_i$ 可以同时满足约束 $\boldsymbol{w}_i^{\mathrm{T}} \boldsymbol{w}_i=1, \boldsymbol{w}_i^{\mathrm{T}} \boldsymbol{w}_j=0, i \neq j$.

根据拉格朗日乘子法的原理可知，此时求得的结果仅是最优解的必要条件，$\mathbf{X}\mathbf{X}^{\mathrm{T}}$ 有 d 个相互正交的单位特征向量，所以还需要从这 d 个特征向量里找出 d' 个能使得目标函数达到最优值的特征向量作为最优解.

将 $\mathbf{X} \mathbf{X}^{\mathrm{T}} \boldsymbol{w}_i=\lambda_i \boldsymbol{w}_i$ 代入目标函数，因为需要降到的维数为 d', 得:

$$\begin{aligned}\min_{\mathbf{W}}-\operatorname{tr}\left(\mathbf{W}^{\mathrm{T}} \mathbf{X} \mathbf{X}^{\mathrm{T}} \mathbf{W}\right) &=\max_{\mathbf{W}} \operatorname{tr}\left(\mathbf{W}^{\mathrm{T}} \mathbf{X} \mathbf{X}^{\mathrm{T}} \mathbf{W}\right) \\ &=\max_{\mathbf{W}} \sum_{i=1}^{d^{\prime}} \boldsymbol{w}_i^{\mathrm{T}} \mathbf{X} \mathbf{X}^{\mathrm{T}} \boldsymbol{w}_i \\ &=\max_{\mathbf{W}} \sum_{i=1}^{d^{\prime}} \boldsymbol{w}_i^{\mathrm{T}} \cdot \lambda_i \boldsymbol{w}_i \\ &=\max_{\mathbf{W}} \sum_{i=1}^{d^{\prime}} \lambda_i \boldsymbol{w}_i^{\mathrm{T}} \boldsymbol{w}_i \\ &=\max_{\mathbf{W}} \sum_{i=1}^{d^{\prime}} \lambda_i\end{aligned}$$

因此，只需要令 $\lambda_1, \lambda_2, \ldots, \lambda_{d'}$ 和 $\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{d'}$ 分别为矩阵 $\mathbf{X}\mathbf{X}^{\mathrm{T}}$ 的前 d' 个最大的特征值和对应的单位特征向量就能得到最优解.

(2) 计算协方差矩阵:

$$\mathbf{X} \mathbf{X}^{\mathrm{T}}=\begin{bmatrix}-1 & 0 & 1 \\ 1 & -2 & 1\end{bmatrix} \cdot \begin{bmatrix}-1 & 1 \\ 0 & -2 \\ 1 & 1\end{bmatrix}=\begin{bmatrix}2 & 0 \\ 0 & 6\end{bmatrix}$$

对应的特征多项式:

$$f(\lambda) = |\lambda \mathbf{I} - \mathbf{X}\mathbf{X}^{\mathrm{T}}| = \begin{vmatrix} \lambda - 2 & 0 \\ 0 & \lambda - 6 \end{vmatrix} = 0$$

解得: $\lambda = 2$ 或 $\lambda = 6$.

对应的特征向量分别为 $\boldsymbol{y}_1 = (1, 0)^{\mathrm{T}}, \boldsymbol{y}_2 = (0, 1)^{\mathrm{T}}$. 由 (1) 中结论可知最大主成分对应的 $\boldsymbol{w}_1 = (0, 1)^{\mathrm{T}}$

(3) a. 平移：

每个样本沿 $\boldsymbol{q}$ 方向平移 s, 则有:

$$\hat{\mathbf{X}} = \mathbf{X} + s\mathbf{q}\mathbf{1}^T$$

其中 $\mathbf{1}$ 是维数为 n 的全 1 列向量.

将平移后的样本中心化后，求协方差矩阵:

$$\hat{\mathbf{H}} = (\hat{\mathbf{X}} - s\mathbf{q}\mathbf{1}^T)(\hat{\mathbf{X}} - s\mathbf{q}\mathbf{1}^T)^{\mathrm{T}} = \mathbf{X}\mathbf{X}^{\mathrm{T}}$$

所以投影矩阵不变:

$$\hat{\mathbf{W}} = \mathbf{W}$$

b. 放缩：

每个样本乘以缩放率 α，则有:

$$\hat{\mathbf{X}} = \alpha\mathbf{X}$$

由于是数乘变换，不需要对变化后的矩阵再进行中心化操作.

协方差矩阵

$$\hat{\mathbf{H}} = \alpha^2\mathbf{X}\mathbf{X}^{\mathrm{T}}$$

数乘后矩阵的单位特征向量不变，因此放缩变换之后 $\hat{\mathbf{W}} = \mathbf{W}$.

c. 旋转：

每个样本围绕点 $\boldsymbol{p}$ 顺时针旋转 θ，可以看成把样本点平移到坐标原点进行旋转，最后把样本点再反向平移，则有:

$$\hat{\mathbf{X}} = \mathbf{R}(\theta)(\mathbf{X} - \boldsymbol{p} \cdot \mathbf{1}^{\mathrm{T}}) + \boldsymbol{p} \cdot \mathbf{1}^{\mathrm{T}}$$

由于平移变换不改变投影矩阵，所以我们不妨考虑围绕坐标原点的旋转：

$$\hat{\mathbf{X}} = \mathbf{R}(\theta)\mathbf{X}$$

计算协方差矩阵:

$$\hat{\mathbf{H}} = \mathbf{R}(\theta)\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{R}(\theta)^{\mathrm{T}}$$

由于旋转矩阵均为正交矩阵，即 $\mathbf{R}(\theta)^{-1} = \mathbf{R}(\theta)^{\mathrm{T}}$，令 $\mathbf{P} = \mathbf{R}(\theta)^{\mathrm{T}}$，得到

$$\hat{\mathbf{H}} = \mathbf{P}^{-1}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{P}$$

根据特征向量的性质，可以得到若 $\boldsymbol{w}$ 为原协方差矩阵 $\mathbf{X}\mathbf{X}^{\mathrm{T}}$ 的一个特征向量，$\hat{\boldsymbol{w}} = \mathbf{R}(\theta)\boldsymbol{w}$ 为新协方差矩阵 $\hat{\mathbf{H}}$ 的特征向量. 所以在旋转变换后

$$\hat{\mathbf{W}} = \mathbf{R}(\theta)\mathbf{W}$$

.

3 [35pts] Theoretical Analysis of k-means Algorithm

给定样本集 $\mathcal{D} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\}$, k-means 聚类算法希望获得簇划分 $\mathcal{C} = \{C_1, C_2, \cdots, C_k\}$, 使得最小化欧氏距离

$$J(\gamma, \mu_1, \ldots, \mu_k) = \sum_{i=1}^{n} \sum_{j=1}^{k} \gamma_{ij} \|\mathbf{x}_i - \mu_j\|^2, \tag{3.1}$$

其中 $\mu_1, \ldots, \mu_k$ 为 k 个簇的中心 (means), $\gamma \in \mathbb{R}^{n \times k}$ 为指示矩阵 (indicator matrix). γ 具体定义如下：若 $\mathbf{x}_i$ 属于第 j 个簇, 则 $\gamma_{ij} = 1$, 否则为 0. 算法 1 中所示为经典 k-means 聚类算法的具体流程 (与课本中描述稍有差别, 但实际上是等价的).

Algorithm 1: k-means Algorithm

1 Initialize $\mu_1, \ldots, \mu_k$.

2 **repeat**

3 **Step 1**: Decide the class memberships of $\{\mathbf{x}_i\}_{i=1}^n$ by assigning each of them to its nearest cluster center.

$$\gamma_{ij} = \begin{cases} 1, & \|\mathbf{x}_i - \mu_j\|^2 \le \|\mathbf{x}_i - \mu_{j'}\|^2, \forall j', \\ 0, & \text{otherwise.} \end{cases}$$

4 **Step 2**: For each $j \in \{1, \cdots, k\}$, recompute μ_j using the updated γ to be the center of mass of all points in C_j:

$$\mu_j = \frac{\sum_{i=1}^n \gamma_{ij} \mathbf{x}_i}{\sum_{i=1}^n \gamma_{ij}}.$$

5 **until** *the objective function J no longer changes*;

(1) **[5pts]** 试证明, 在算法 1 中, **Step 1** 和 **Step 2** 都会使目标函数 J 的值降低.

(2) **[5pts]** 试证明, 算法 1 会在有限步内停止.

(3) **[5pts]** 试证明, 目标函数 J 的最小值是关于 k 的非增函数, 其中 k 是聚类簇的数目.

(4) **[10pts]** 记 $\hat{\mathbf{x}}$ 为 n 个样本的中心点, 定义如下变量:

total deviation	$T(X) = \sum_{i=1}^n \|\mathbf{x}_i - \hat{\mathbf{x}}\|^2 / n$
intra-cluster deviation	$W_j(X) = \sum_{i=1}^n \gamma_{ij} \|\mathbf{x}_i - \mu_j\|^2 / \sum_{i=1}^n \gamma_{ij}$
inter-cluster deviation	$B(X) = \sum_{j=1}^k \frac{\sum_{i=1}^n \gamma_{ij}}{n} \|\mu_j - \hat{\mathbf{x}}\|^2$

试探究以上三个变量之间有什么样的等式关系? 基于此, 请证明, k-means 聚类算法可以认为是在最小化 intra-cluster deviation 的加权平均, 同时近似最大化 inter-cluster deviation.

(5) **[10pts]** 在公式 3.1 中, 我们使用 ℓ_2-范数来度量距离 (即欧氏距离), 下面我们考虑使用 ℓ_1-范数来度量距离:

$$J'(\gamma, \mu_1, \ldots, \mu_k) = \sum_{i=1}^{n} \sum_{j=1}^{k} \gamma_{ij} \|\mathbf{x}_i - \mu_j\|_1. \tag{3.2}$$

- **[5pts]** 请仿效算法 1 (k-means-ℓ_2 算法), 给出新的算法 (命名为 k-means-ℓ_1 算法) 以优化公式 3.2 中的目标函数 J'.
- **[5pts]** 当样本集中存在少量异常点 (outliers) 时, 对于上述的 k-means-ℓ_2 和 k-means-ℓ_1 算法, 我们应该如何选择? 请从算法鲁棒性的角度分析, 说明哪个算法具有更好的鲁棒性?

Solution. 此处用于写解答 (中英文均可)

(1)
- Step1

  假设 step1 更新前样本 x_l 属于簇 C_j, 更新后属于簇 $C_{j'}$，那么由更新条件有:

  $$\|\mathbf{x}_l - \mu_{j'}\|^2 \leq \|\mathbf{x}_p - \mu_j\|^2$$

  记经过 step1 所有发生簇变换的样本的下标集合 $\mathcal{L}$.
  则:

  $$J = \sum_{l \in \mathcal{L}} \|\mathbf{x}_l - \mu_{j_l}\|^2 + \sum_{i \notin \mathcal{L}} \sum_{j=1}^{k} \gamma_{ij} \|\mathbf{x}_i - \mu_j\|^2$$

  $$J' = \sum_{l \in \mathcal{L}} \|\mathbf{x}_l - \mu_{j'_l}\|^2 + \sum_{i \notin \mathcal{L}} \sum_{j=1}^{k} \gamma_{ij} \|\mathbf{x}_i - \mu_j\|^2$$

  由于 $\sum\limits_{l \in \mathcal{L}} \|\mathbf{x}_l - \mu_{j'_l}\|^2 \leq \sum\limits_{l \in \mathcal{L}} \|\mathbf{x}_l - \mu_{j_l}\|^2$, 所以 $J' \leq J$.
- Step2

  Step2 步骤是在更新 Step1 得到的簇的中心点，欲证明在更新了中心点之后,$J' \leq J$, 只需证明新的中心点是使得新簇中所有点的中心距最小的点，即:

  $$\mu_j = \arg\min_{\mu} \sum_{i=1}^{n} \gamma_{ij} \|\mathbf{x}_i - \mu\|^2, \quad \text{其中} \mu_j \text{ 是更新后的中心点.}$$

  目标函数对 μ 求导并令其等于 0，可得：

  $$\frac{\partial}{\partial \mu} \sum_{i=1}^{n} \gamma_{ij} \|\mathbf{x}_i - \mu_j\|^2 = -2 \sum_{i=1}^{n} \gamma_{ij} (\mathbf{x}_i - \mu_j) = 0$$

  解之得：

  $$\mu_j = \frac{\sum_{i=1}^{n} \gamma_{ij} \mathbf{x}_i}{\sum_{i=1}^{n} \gamma_{ij}}$$

  正好是 Step2 对应得更新规则. 由此可得 $J' \leq J$.

(2) 算法 1 是一个迭代的过程，每次迭代目标函数 J 都会单调递减，且 J 有下界 0. 因此我们只需要证明，在有限次迭代之后，算法会终止.
假设在第 t 次迭代后，经过 Step1 和 Step2 步骤得到的新的簇中心与之前的相同，

即 $\mu_i^{(t)} = \mu_i^{(t-1)}, i \in [k]$. 根据算法的定义, 对于任意数据点 x_i, 它属于原先的某个簇 $C_j^{(t-1)}$ 并且 $\|x_i - \mu_j^{(t-1)}\| \leq \|x_i - \mu_{j'}^{(t-1)}\|$ 对所有 $j \neq j'$ 都成立. 而由于 $\mu_i^{(t)} = \mu_i^{(t-1)}$, 所以上述不等式仍然成立. 因此，在第 t 次迭代后，所有的数据点都属于原先的簇，并没有发生改变，所以此时目标函数 J 也没有发生改变，算法会终止.
综上所述, 算法 1 会在有限步内停止.

(3) 假设当前得到一个最优簇划分 $\mathcal{C} = \{C_1, C_2, \cdots, C_k\}$, 此时目标函数 $J(k)$ 达到最小 $J(k)_{min}$. 若此时新增加一个簇 C_{k+1}, 初始状态下 C_{k+1} 中仅含一个样本点 (这个样本点可以任取一个)，则 $\sum_{i=1}^{n} \gamma_{i,k+1} \|\mathbf{x}_i - \mu_{k+1}\|^2 = 0$, 记此时的目标函数为 $J(k+1)$, $J(k+1) = \sum_{i=1}^{n} \sum_{j=1}^{k+1} \gamma_{ij} \|\mathbf{x}_i - \mu_j\|^2 = J(k) + \sum_{i=1}^{n} \gamma_{i,k+1} \|\mathbf{x}_i - \mu_{k+1}\|^2$, 有 $J(k+1) = J(k)_{min}$. 由 (1) 中结论，每一次迭代都会使 $J(k+1)$ 的值变小，因此最终有 $J(k+1)_{min} \leq J(k)_{min}$. 即目标函数 J 的最小值是关于 k 的非增函数.

(4)

$$
\begin{aligned}
T(X) &= \frac{\sum_{i=1}^{n} \|\mathbf{x}_i - \hat{\mathbf{x}}\|^2}{n} \\
&= \frac{\sum_{i=1}^{n} \|\mathbf{x}_i - \mu_{x_i} + \mu_{x_i} - \hat{\mathbf{x}}\|^2}{n} \quad (\mu_{x_i} \text{ 表示 } x_i \text{ 所在簇的中心}) \\
&= \frac{\sum_{i=1}^{n} \sum_{j=1}^{k} \gamma_{ij} \|\mathbf{x}_i - \mu_j + \mu_j - \hat{\mathbf{x}}\|^2}{n} \\
&= \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{k} \gamma_{ij} \|\mathbf{x}_i - \mu_j\|^2 + \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{k} \gamma_{ij} \|\mu_j - \hat{\mathbf{x}}\|^2 + \frac{2}{n} \sum_{i=1}^{n} \sum_{j=1}^{k} \gamma_{ij} (\mathbf{x}_i - \mu_j)^{\mathrm{T}} (\mu_j - \hat{\mathbf{x}}) \\
W(X) &= \frac{1}{n} \sum_{j=1}^{k} \left(W_j(X) \cdot \sum_{i=1}^{n} \gamma_{ij} \right) = \frac{1}{n} \sum_{j=1}^{k} \sum_{i=1}^{n} \gamma_{ij} \|\mathbf{x}_i - \mu_j\|^2 \\
B(X) &= \sum_{j=1}^{k} \frac{\sum_{i=1}^{n} \gamma_{ij}}{n} \|\mu_j - \hat{\mathbf{x}}\|^2 = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{k} \gamma_{ij} \|\mu_j - \hat{\mathbf{x}}\|^2
\end{aligned}
$$

而:

$$
\frac{2}{n} \sum_{i=1}^{n} \sum_{j=1}^{k} \gamma_{ij} (\mathbf{x}_i - \mu_j)^{\mathrm{T}} (\mu_j - \hat{\mathbf{x}}) = \frac{2}{n} \sum_{j=1}^{k} \left[\sum_{i=1}^{n} \gamma_{ij} (\mathbf{x}_i - \mu_j)^{\mathrm{T}} \right] (\mu_j - \hat{\mathbf{x}})
$$

对于某个确定的簇 C_j, $\sum_{i=1}^{n} \gamma_{ij} (\mathbf{x}_i - \mu_j)^{\mathrm{T}} = 0$, 因为该簇的所有点到簇中心的向量之和是零向量. 所以我们有:

$$
T(X) = W(X) + B(X)
$$

即 $T(X)$ 等于 intra-cluster deviation 的加权平均加上 inter-cluster deviation.
对于给定的样本 $T(X)$ 是定值，而 $W(X) = \frac{J}{n}$, J 是目标函数.
因此最小化 J 的过程相当于最小化 $W(X)$, 同时近似最大化 $B(X)$, 也即 k-means 聚类算法可以认为是在最小化 intra-cluster deviation 的加权平均, 同时近似最大化 inter-cluster deviation.

(5)
- k-means-ℓ_1 算法

Algorithm 2: k-means-ℓ_1 Algorithm

1 Initialize $\mu_1, \ldots, \mu_k$.

2 **repeat**

3 **Step 1**: Decide the class memberships of $\{\mathbf{x}_i\}_{i=1}^n$ by assigning each of them to its nearest cluster center.

$$\gamma_{ij} = \begin{cases} 1, & \|\mathbf{x}_i - \mu_j\|_1 \leq \|\mathbf{x}_i - \mu_{j'}\|_1, \forall j', \\ 0, & \text{otherwise.} \end{cases}$$

4 **Step 2**: For each $j \in \{1, \cdots, k\}$, recompute μ_j using the updated γ to be the median of all points in C_j:

$$\mu_j = \text{Median}\{\mathbf{x}_i \mid \gamma ij = 1\}.$$

5 **until** *the objective function J no longer changes*;

- 从算法鲁棒性的角度考虑，在存在异常点的情况下，k-means-ℓ_1 算法比 k-means-ℓ_2 算法算法具有更好的鲁棒性.
  首先，因为 k-means-ℓ_2 算法的距离度量是 ℓ_2 范数的平方，如果有异常点，那么计算出来的 ℓ_2 范数距离会比较大, k-means-ℓ_1 使用曼哈顿距离，尽管存在异常点，但据此计算出来的距离不会比 ℓ_2 范数距离大.
  其次, k-means-ℓ_2 算法簇中心点是求所有在该簇中的点的均值，如果有异常点，异常点对中心点的计算影响比较大，使得计算出来的中心点偏离实际中心点较远，而 k-means-ℓ_1 算法则是取中位数，少量异常点对中位数的影响比较小，从而计算出来的中心点相对实际中心点的偏离不会太大.

4 [35pts] Neural Network in Practice

本题需编程实现多层前馈神经网络, 且不使用现有神经网络库, 具体内容见 lab.ipynb 文件.

4.1 任务要求

在不使用现有神经网络库的前提下, 复现 lab.ipynb 中利用 PyTorch 编写的一段多层前馈神经网络代码, 需注意:

- 保持网络结构一致;
- 保持损失函数一致;
- 保持 batch_size, learning_rate, epochs 超参一致.

编写时可参考 lab.ipynb 文件中给出的代码框架, 也可以将其删除, 编写你自己的代码. 任务完成后, 需提交对应的 .ipynb 文件.

4.2 评分标准

我们会重新运行你所提交的 .ipynb 文件, 具体评分标准如下:

- 5/35: 复现的代码可正常运行;
- 10/35: 复现的代码保持了网络结构、损失函数以及各超参一致;
- 15/35: 复现的代码可在 5min 内迭代完规定的 epochs 轮数;
- 25/35: 运行结果中 running_loss 整体为下降趋势;
- 35/35: 运行结果中最后一轮的 running_acc 超过 90%.

4.3 测试环境

以下为测试环境采用的版本:

- Python: 3.8.16
- PyTorch: 2.0.0
- Numpy: 1.24.2

你可以使用你喜欢的任意版本, 只需确保你的代码能顺利运行.