姓名：麻超
学号：201300066

一. (20 points) 神经网络基础

给定训练集 $D=\{(\boldsymbol{x}_1,\boldsymbol{y}_1),(\boldsymbol{x}_2,\boldsymbol{y}_2),...,(\boldsymbol{x}_m,\boldsymbol{y}_m)\}$. 其中 $\boldsymbol{x}_i\in\mathbb{R}^d$, $\boldsymbol{y}_i\in\mathbb{R}^l$ 表示输入示例由 d 个属性描述, 输出 l 维实值向量. 图 1给出了一个有 d 个输入神经元、l 个输出神经元、q 个隐层神经元的多层神经网络, 其中输出层第 j 个神经元的阈值用 θ_j 表示, 隐层第 h 个神经元的阈值用 γ_h 表示. 输入层第 i 个神经元与隐层第 h 个神经元之间的连接权为 v_{ih}, 隐层第 h 个神经元与输出层第 j 个神经元之间的连接权为 w_{hj}. 记隐层第 h 个神经元接收到的输入为 $\alpha_h=\sum_{i=1}^d v_{ih}x_i$, 输出层第 j 个神经元接收到的输入为 $\beta_j=\sum_{h=1}^q w_{hj}b_h$, 其中 b_h 为隐层第 h 个神经元的输出.

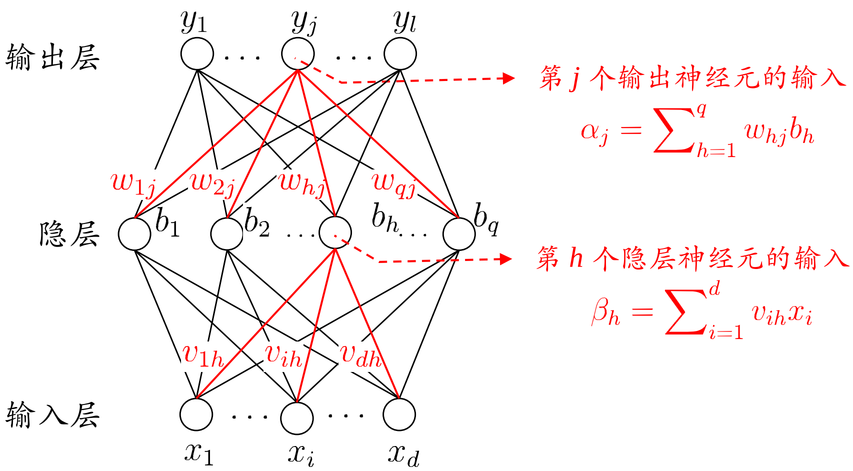

Figure 1: 多层神经网络（教材图 5.7）

不同任务中神经网络的输出层往往使用不同的激活函数和损失函数, 本题介绍几种常见的激活和损失函数, 并对其梯度进行推导.

1. 在二分类问题中（$l=1$）, 标记 $y\in\{0,1\}$, 一般使用 Sigmoid 函数作为激活函数, 使输出值在 $[0,1]$ 范围内, 使模型预测结果可直接作为概率输出. Sigmoid 函数的输出一般配合二元交叉熵 (Binary Cross-Entropy) 损失函数使用, 对于一个训练样本 $(\boldsymbol{x},y)$ 有

$$\ell(y,\hat{y}_1)=-\left[y\log(\hat{y}_1)+(1-y)\log(1-\hat{y}_1)\right] \tag{1}$$

记 $\hat{y}_1$ 为模型对样本属于正类的预测结果, 请计算 $\frac{\partial\ell(y,\hat{y}_1)}{\partial\beta_1}$,

2. 当 $l > 1$, 网络的预测结果为 $\hat{\boldsymbol{y}} \in \mathbb{R}^l$, 其中 $\hat{y}_i$ 表示输入被预测为第 i 类的概率. 对于第 i 类的样本, 其标记 $\boldsymbol{y} \in \{0,1\}^l$, 有 $y_i = 1$, $y_j = 0, j \neq i$. 对于一个训练样本 $(\boldsymbol{x}, \boldsymbol{y})$, 交叉熵损失函数 $\ell(\boldsymbol{y}, \hat{\boldsymbol{y}})$ 的定义如下

$$\ell(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\sum_{j=1}^{l} y_j \log \hat{y}_j \tag{2}$$

在多分类问题中, 一般使用 Softmax 层作为输出, Softmax 层的计算公式如下

$$\hat{y}_j = \frac{e^{\beta_j}}{\sum_{k=1}^{l} e^{\boldsymbol{\beta}_k}} \tag{3}$$

易见 Softmax 函数输出的 $\hat{\boldsymbol{y}}$ 符合 $\sum_{j=1}^{l} \hat{y}_j = 1$, 所以可以直接作为每个类别的概率. Softmax 输出一般配合交叉熵 (Cross Entropy) 损失函数使用, 请计算 $\frac{\partial \ell(\boldsymbol{y}, \hat{\boldsymbol{y}})}{\partial \beta_j}$,

3. 分析在二分类中使用 Softmax 和 Sigmoid 的联系与区别.

4. KL 散度 (Kullback-Leibler divergence) 定义了两个分布之间的距离, 对于两个离散分布 $Q(x)$ 和 $P(x)$, 其定义为

$$D_{\mathrm{KL}}(P \parallel Q) = \sum_{x \in \mathcal{X}} P(x) \log \left(\frac{P(x)}{Q(x)} \right) \tag{4}$$

其中 $\mathcal{X}$ 为 x 的取值空间. 试分析交叉熵损失函数和 KL 散度的关系.

解:

1. 根据已知条件, 有:

$$\begin{aligned}
f'(\beta_1, \theta_1) &= f(\beta_1, \theta_1) \cdot (1 - f(\beta_1, \theta_1)) \\
\frac{\partial \ell(y, \hat{y}_1)}{\partial \beta_1} &= \frac{\partial \ell(y, \hat{y}_1)}{\partial \hat{y}_1} \cdot \frac{\partial \hat{y}_1}{\partial \beta_1} \\
&= (-\frac{y}{\hat{y}_1} + \frac{1-y}{1-\hat{y}_1}) \cdot \hat{y}_1 \cdot (1 - \hat{y}_1) \\
&= -y + \hat{y}_1
\end{aligned}$$

2.

$$\frac{\partial \ell(\boldsymbol{y}, \hat{\boldsymbol{y}})}{\partial \beta_j} = \frac{\partial \ell(\boldsymbol{y}, \hat{\boldsymbol{y}})}{\partial \hat{\boldsymbol{y}}_j} \cdot \frac{\hat{\boldsymbol{y}}_j}{\partial \beta_j} + \sum_{k=1, j\neq k}^{l} \frac{\partial \ell(\boldsymbol{y}, \hat{\boldsymbol{y}})}{\partial \hat{\boldsymbol{y}}_k} \cdot \frac{\partial \ell(\hat{\boldsymbol{y}}_k)}{\partial \beta_j}$$
$$= -\boldsymbol{y}_j(1 - \hat{\boldsymbol{y}}_j) + \sum_{k=1, j\neq k}^{l} \boldsymbol{y}_k (1 - \frac{\sum_{m=1, m\neq j}^{l} e^{\beta} m}{e^{\beta} k} \cdot \hat{\boldsymbol{y}}_k)$$

3. 联系: 在二分类问题中,Sigmoid 与 Softmax 输出结构相同, 均为 $\frac{1}{1+e^{x_2-x_1}}$. 区别: 它们在二分类问题的输出原理不同. 其中 Sigmoid 针对于两点式分布问题, 结果中, 可以分为该类的概率为 P, 则不能分到该类的概率为 1-P. 与之对应,Softmax 方法针对多分类问题, 在化简至二分类时仍保留其思想, 其正例和反例概率分别计算.

4. 交叉熵是熵与 KL 散度的和.
在计算损失的过程中, 二者等价.

$$\begin{aligned} D_{KL}(P||Q) &= \sum_{x\in\mathcal{X}} P(x) \log(\frac{P(x)}{Q(x)}) \\ &= \sum_{x\in\mathcal{X}} P(x)(\log P(x) - \log Q(x)) \\ &= \sum_{x\in\mathcal{X}} P(x) \log P(x) - \sum_{x\in\mathcal{X}} P(x) \log Q(x) \\ &= -S(P) + \ell(P, Q) \end{aligned}$$

其中 S 为熵, 当 $S(A)$ 为常数时, 最小化 KL 散度等同于最小化交叉熵.

二. (20 points) **运算的向量化**
在编程实践中, 一般需要将运算写成向量或者矩阵运算的形式, 这叫做运算的向量化 (vectorization). 向量化可以充分利用计算机体系结构对矩阵运算的支持加速计算, 大部分数学运算库例如numpy也对矩阵计算有专门的优化. 另一方面, 如果一个运算可以写成向量计算的形式, 会更容易写出其导数形式并进行优化. 本题中举两个简单的例子

1. 给定示例矩阵 $\boldsymbol{X} \in \mathbb{R}^{m\times d}$, 表示 m 个示例（向量）, 每个示例有 d 维,

计算 m 个示例两两之间的距离矩阵 $\boldsymbol{D} \in \mathbb{R}^{m\times m}$, 两个向量之间的欧式距离定义为 $\|\boldsymbol{x}-\boldsymbol{y}\|_2 = \sqrt{\sum_{i=1}^{d}(x_i - y_i)^2}$. 求距离矩阵可以通过循环的方式, 即`plain_distance_function`中实现的方法;

```
import numpy as np

def plain_distance_function(X):
    # 直观的距离计算实现方法
    # 首先初始化一个空的距离矩阵D
    D = np.zeros((X.shape[0], X.shape[0]))
    # 循环遍历每一个样本对
    for i in range(X.shape[0]):
        for j in range(X.shape[0]):
            # 计算样本i和样本j的距离
            D[i, j] = np.sqrt(np.sum((X[i] - X[j])**2))
    return D
```

2. 输入一个矩阵 $\boldsymbol{X} \in \mathbb{R}^{m\times d}$, 表示 m 个向量, 每个向量有 d 维, 要求对输入矩阵的行按照一个给定的排列 $\boldsymbol{p} = \{p_1, p_2, ..., p_m\}$ 进行重新排列. 即输出一个新的矩阵 $\boldsymbol{X}'$, 其中第 i 行的内容为输入矩阵的第 p_i 行. 假设重排列为一个函数 perm 即 $\boldsymbol{X}' = \mathrm{perm}(\boldsymbol{X})$, 已知梯度 $\frac{\partial \ell}{\partial \boldsymbol{X}'}$, 需要计算 $\frac{\partial \ell}{\partial \boldsymbol{X}}$. 对矩阵的行进行排列可以采用简单的循环实现, 例如`plain_permutation_function`中的实现方法.

```
import numpy as np

def plain_permutation_function(X, p):
    # 初始化结果矩阵, 其中每一行对应一个样本
    permuted_X = np.zeros_like(X)
    for i in range(X.shape[0]):
        # 采用循环的方式对每一个样本进行重排列
        permuted_X[i] = X[p[i]]
    return permuted_X
```

请给出上述两种任务的向量化实现方案，并分析上述实现方法和向量化实现方法之间运行时间的差异。（提示：比如可以针对不同规模的矩阵大小来尝试分析主要操作的运行时间）

解：

三. (20 points) **支持向量机**

考虑标准的 SVM 优化问题如下 (即教材公式 (6.35)),

$$\begin{aligned}\min_{\boldsymbol{w},b,\xi_i} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m}\xi_i \\ \text{s.t.} \quad & y_i\left(\boldsymbol{w}^\top \boldsymbol{x}_i + b\right) \geq 1-\xi_i \\ & \xi_i \geq 0, i \in [m].\end{aligned} \tag{5}$$

注意到, 在 (2.1) 中, 对于正例和负例, 其在目标函数中分类错误的“惩罚”是相同的. 在实际场景中, 很多时候正例和负例错分的“惩罚”代价是不同的（参考教材 2.3.4 节）. 比如考虑癌症诊断问题, 将一个确实患有癌症的人误分类为健康人, 以及将健康人误分类为患有癌症, 产生的错误影响以及代价不应该认为是等同的. 所以对负例分类错误的样本 (即 false positive) 施加 $k>0$ 倍于正例中被分错的样本的“惩罚”. 对于此类场景下

1. 请给出相应的 SVM 优化问题.
2. 请给出相应的对偶问题, 要求详细的推导步骤, 如 KKT 条件等.

解:

1.

$$\begin{aligned}\min_{\boldsymbol{w},b,\xi_i} \quad & \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m}s_i\xi_i \\ \text{s.t.} \quad & y_i\left(\boldsymbol{w}^\top \boldsymbol{x}_i + b\right) \geq 1-\xi_i \\ & \xi_i \geq 0, i \in [m].\end{aligned}$$

其中,s_i 为第 i 个训练样本对 C 的加权系数, 负例分类错误时为 k, 正例分类错误时为 1.

2. 令 α, μ 表示拉格朗日乘子, 有:

$$\begin{aligned} L(\boldsymbol{w}, b, \xi, \alpha, \mu) = & \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^m s_i\xi_i \\ & + \sum_{i=1}^m \alpha_i(1-\xi_i - y_i(\boldsymbol{w}\boldsymbol{x}_i + b)) - \sum_{i=1}^m \mu_i\xi_i \end{aligned}$$

$$\because \Delta_{\boldsymbol{w}} L = \Delta_b L = \Delta_{\xi_i} L = 0$$

$$\therefore w - \sum_{i=1}^m \alpha_i y_i \boldsymbol{x}_i = \sum_{i=1}^m \alpha_i y_i = Cs_i - (\alpha_i + \mu_i) = 0$$

故对偶问题为:

$$\begin{aligned} \max_{\alpha} \quad & \sum_{i=1}^m \alpha_i - \frac{1}{2}\sum_{i=1}^m\sum_{j=1}^m (\alpha_i\alpha_j y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j) \\ \text{s.t.} \quad & \sum_{i=1}^m y_i\alpha_i = 0 \\ & 0 \le \alpha_i \le s_i C, i \in [m]. \end{aligned}$$

KKT 条件为:

$$\begin{cases} y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) - 1 + \xi_i \ge 0 \\ \alpha_i(y_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) - 1 + \xi_i \ge 0) = 0, \mu_i\xi_i = 0 \\ \alpha_i, \mu_i, \xi_i \ge 0 \\ \boldsymbol{w} = \sum_{i=1}^m \alpha_i y_i \boldsymbol{x}_i, \sum_{i=1}^m \alpha_i, y_i = 0 \end{cases}$$

四. (20 points) **核函数**

教材 6.3 节介绍了 Mercer 定理, 说明对于一个二元函数 $k(\cdot,\cdot)$, 当且仅当对任意 m 和 $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$, 它对应的 Gram 矩阵 (核矩阵) 是半正定的时, 它是一个有效的核函数. 核矩阵 $\boldsymbol{K}$ 中的元素为 $K_{ij} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$. 请根据 Mercer 定理证明以下核函数是有效的.

1. $\kappa_3 = a_1\kappa_1 + a_2\kappa_2$, 其中 $a_1, a_2 \ge 0$.
2. $f(\cdot)$ 是任意实值函数, 由 $\kappa_4(\boldsymbol{x}, \boldsymbol{x}') = f(\boldsymbol{x})f(\boldsymbol{x}')$ 定义的 κ_4.
3. 由 $\kappa_5(\boldsymbol{x}, \boldsymbol{x}') = \kappa_1(\boldsymbol{x}, \boldsymbol{x}')\kappa_2(\boldsymbol{x}, \boldsymbol{x}')$ 定义的 κ_5.

4. 由 $\kappa_6(\boldsymbol{x}, \boldsymbol{x}') = f(\boldsymbol{x})\kappa_1(\boldsymbol{x}, \boldsymbol{x}') f(\boldsymbol{x}')$ 定义的 κ_6

解:

1. 对于任意非零向量 x, 有:

$$\begin{aligned} x^T K_3 x &= x^T(a_1K_1 + a_2K_2)x \\ &= a_1x^TK_1x + a_2x^TK_2x \end{aligned}$$

$$\because a_1, a_2 > 0, x^TK_1x, x^TK_2x \geq 0$$
$$\therefore x^TK_3x \geq 0$$

因此 K_3 半正定, 故核函数有效.

2. 定义对称矩阵 M 如下:

$$M = diag(f(x), f(x), ..., f(x))$$

其对应的核函数可以看作两个矩阵的乘积, 即 $K_4 = MM^T$. 故 K_4 为半正定矩阵,k_4 为核函数.

3. 只需证明两个半正定矩阵的直积仍然是半正定矩阵, 可以定义 $K_5 = K_1 \circ K_2$.

$$\begin{aligned} y^TK_5y = y^T&(K_1 \circ K_2)y \\ &= \sum_{i=1}^{m}\sum_{j=1}^{m} k_1(x_i, x_j)k_2(x_i, x_j)y_iy_j \\ &= tr(diag(y_1, y_2, ..., y_m)K_1diag(y_1, y_2, ..., y_m)K_2^T) \\ &= tr(diag(y_1, y_2, ..., y_m)AA^Tdiag(y_1, y_2, ..., y_m)BB^T) \\ &= tr(B \quad diag(y_1, y_2, ..., y_m)A(B \quad diag(y_1, y_2, ..., y_m)A)^T) \\ &= tr(PP^T) \geq 0 \end{aligned}$$

所以 k_5 是一个核函数.

4. K_1 为半正定矩阵, 可以将其写成 $K_1 = NN^T$. 故 k_6 的核矩阵为:

$$K_6 = MNN^TM^T = MN(MN^T).$$

这是一个半正定矩阵, 故 k_6 是一个核函数.

五. (20 points) **主成成分分析**

$\boldsymbol{x} \in \mathbb{R}^d$ 是一个随机向量, 其均值和协方差分别是 $\boldsymbol{\mu} = \mathbb{E}(\boldsymbol{x}) \in \mathbb{R}^d$, $\boldsymbol{\Sigma} = \mathbb{E}(\boldsymbol{x} - \boldsymbol{\mu_x})(\boldsymbol{x} - \boldsymbol{\mu_x})^\top \in \mathbb{R}^{d \times d}$. 定义随机变量 $\{y_i = \boldsymbol{u}_i^\top \boldsymbol{x} + a_i \in \mathbb{R}, i = 1, \ldots, d' \leq d\}$ 为 $\boldsymbol{x}$ 的主成分, 其中 $\boldsymbol{u}_i \in \mathbb{R}^d$ 是单位向量 ($\boldsymbol{u}_i^\top \boldsymbol{u}_i = 1$), $a_i \in \mathbb{R}$, $\{y_i\}_{i=1}^{d'}$ 是互不相关的零均值随机变量, 它们的方差满足 $\mathrm{var}(y_1) \geq \mathrm{var}(y_2) \geq \cdots \geq \mathrm{var}(y_{d'})$. 假设 $\boldsymbol{\Sigma}$ 没有重复的特征值.

1. 请证明 $\{a_i = -\boldsymbol{u}_i^\top \boldsymbol{\mu}\}_{i=1}^{d'}$.
2. 请证明 $\boldsymbol{u}_1$ 是 $\boldsymbol{\Sigma}$ 最大的特征值对应的特征向量. (提示: 写出要最大化的目标函数, 写出约束条件, 使用拉格朗日乘子法)
3. 请证明 $\boldsymbol{u}_2^\top \boldsymbol{u}_1 = 0$, 且 $\boldsymbol{u}_2$ 是 $\boldsymbol{\Sigma}$ 第二大特征值对应的特征向量. (提示: 由 $\{y_i\}_{i=1}^{d}$ 是互不相关的零均值随机变量可推出 $\boldsymbol{u}_2^\top \boldsymbol{u}_1 = 0$, 可作为第二小问的约束条件之一)
4. 通过 PCA 进行降维, 得到的随机变量满足 $\mathrm{var}(y_1) \geq \mathrm{var}(y_2) \geq \cdots \geq \mathrm{var}(y_d)$, 也就是降维后的数据在不同维度上有不同的方差, 从而导致不同维度的数值范围差异很大, 如果想要降维后的样本在不同维度具有大致相同的数值范围, 应该怎么做?

解:

1. 由于 $\{y_i\}_{i=1}^{d'}$ 为互不相关的零均值随机变量, 所以有:

$$\mathbb{E}(y_i) = \mathbb{E}(\boldsymbol{u}_i^T) = 0$$

所以可以得到:

$$\boldsymbol{u}_i^T \mathbb{E}(x) + a_i = 0$$

$$a_i = -\boldsymbol{u}$$

$$\boldsymbol{u}_i^T \mathbb{E}(x)$$

2. 由于需要最大化主成分的方差, 所以可以得到如下的优化目标:

$$\max \quad \sum_{i=1}^{d'} var(y_i)$$

又

$$\begin{aligned} var(y_i) &= \mathbb{E}(y_i^2) - \mathbb{E}(y_i)^2 \\ &= \mathbb{E}(y_i^2) \\ &= \mathbb{E}(\boldsymbol{u}_i^T\boldsymbol{x}\cdot\boldsymbol{u}_i^T\boldsymbol{x} + 2(-\boldsymbol{u}_i^T\mu\cdot\boldsymbol{u}_i^T\boldsymbol{x}) + \boldsymbol{u}_i^T\mu\cdot\mu\boldsymbol{u}_i^T) \\ &= \mathbb{E}(\boldsymbol{u}_i^T(x-\mu)(x-\mu)^T\boldsymbol{u}_i) \\ &= \boldsymbol{u}_i^T\sum\boldsymbol{u}_i \end{aligned}$$

所以原目标可以写为如下形式:

$$\begin{aligned} \min \quad & -\sum_{i=1}^{d'}\boldsymbol{u}_i^T\sum\boldsymbol{u}_i \\ \text{s.t.} \quad & \boldsymbol{u}_i^T\boldsymbol{u}_i = 1, i\in[m] \end{aligned}$$

该问题的 Lagrange 函数为:

$$L(\boldsymbol{u}_i, v_i) = -\sum_{i=1}^{d'}u_i^T\sum\boldsymbol{u}_i + \sum_{i=1}^{d'}v_i(\boldsymbol{u}_i^T\boldsymbol{u}_i - 1)$$

其 Lagrange 乘子为 $v_i \geq 0$. 分别求偏导得到:

$$\frac{\partial L}{\partial\boldsymbol{u}_i} = -2\sum\boldsymbol{u}_i + 2v_i\boldsymbol{u}_i = 0$$

$$\frac{\partial L}{\partial v_i} = \boldsymbol{u}_i^T\boldsymbol{u}_i - 1 = 0$$

所以有 $\sum\boldsymbol{u}_i = v_i\boldsymbol{u}_i$. 故 v_i 是协方差矩阵 $\sum$ 对应的特征值,$\boldsymbol{u}_i$ 是协方差矩阵 $\sum$ 的特征向量, 且:

$$var(y_i) = \boldsymbol{u}_i^T\sum\boldsymbol{u}_i = v_i\boldsymbol{u}_i^T\boldsymbol{u}_i = v_i$$

故 $var(y_i)$ 对应的向量 $\boldsymbol{u}_i$ 就是最大特征值 v_i 对应的特征向量.

3. 由题意可以得到新的优化问题, 即:

$$\begin{aligned} \max_{\boldsymbol{u}_i\in\{\boldsymbol{u}_i\}_{i=1}^{d'}} \quad & \boldsymbol{u}_i^T(\boldsymbol{x}-\mu)(x-\mu)^T\boldsymbol{u}_i \\ \text{s.t.} \quad & \boldsymbol{u}_i^T\boldsymbol{u}_i = 1, \boldsymbol{u}_i\in\{\boldsymbol{u}_i\}_{i=1}^{d'} \\ & \boldsymbol{u}_i^T\boldsymbol{u}_i = 0, \boldsymbol{u}_i\in\{\boldsymbol{u}_i\}_{i=2}^{d'}. \end{aligned}$$

其 Lagrange 函数为:

$$L(\boldsymbol{u}_i,\lambda_i,\beta_i)=\boldsymbol{u}_i^T(\boldsymbol{x}-\mu)(\boldsymbol{x}-\mu)^T\boldsymbol{u}_i-\lambda_i(\boldsymbol{u}_i^T\boldsymbol{u}_i-1)-\beta_i\boldsymbol{u}_i^T\boldsymbol{u}_1$$

分别求偏导可得:

$$\begin{aligned}\frac{\partial L}{\partial \boldsymbol{u}_i}&=2(\boldsymbol{x}-\mu)(\boldsymbol{x}-\mu)^T\boldsymbol{u}_i-2\lambda_i\boldsymbol{u}_i-\beta_i\boldsymbol{u}_1=0\\ \frac{\partial L}{\partial \lambda_i}&=\boldsymbol{u}_i^T\boldsymbol{u}_i-1=0\\ \frac{\partial L}{\partial \beta_i}&=\boldsymbol{u}_i^T\boldsymbol{u}_1=0\end{aligned}$$

于是有:

$$\boldsymbol{u}_i^T(\boldsymbol{x}-\mu)(\boldsymbol{x}-\mu)^T\boldsymbol{u}_i-\boldsymbol{u}_i^T\frac{1}{2}\beta_i\boldsymbol{u}_1=\boldsymbol{u}_1^T\lambda_i\boldsymbol{u}_i=\lambda_i\boldsymbol{u}_i^T\boldsymbol{u}_i=0$$

$$\boldsymbol{u}_i^T\sum_x\boldsymbol{u}_i=\frac{1}{2}\beta_i$$

由于 $\sum_x$ 为对称矩阵, 可以得到

$$\frac{1}{2}\beta_i=\boldsymbol{u}_i^T\sum_x\boldsymbol{u}_1=\boldsymbol{u}_i^T\lambda_1\boldsymbol{u}_1=0$$

所以可以得到 $\beta=0$.
此时就与上一问几乎一直, 即最大化 λ_i, 且 λ_2 为最大, 故 $\boldsymbol{u}_2$ 是 $\sum$ 第二大的特征值对应的特征向量.

4. 可以在 PCA 降维前对数据进行标准化处理, 也就是对数据先减去均值再除以标准差, 此时就可以使得均值为 0, 标准差为 1.