

# Linear Algebra

Lecture slides for Chapter 2 of *Deep Learning*  
Ian Goodfellow  
2016-06-24

今天我们来看本书的第二章，Linear Algebra，也就是线性代数。

## About this chapter

- Not a comprehensive survey of all of linear algebra
- Focused on the subset most relevant to deep learning
- Larger subset: e.g., *Linear Algebra* by Georgi Shilov

作者在这里特意强点了下，我们这张的线性代数，只关注与深度学习有关的部分，别的内容不关注。如果想要系统学习的同学，作者也推荐了参考书目，就是 PPT 上的那本。

# Scalars

- A scalar is a single number
- Integers, real numbers, rational numbers, etc.
- We denote it with italic font:

$$a, n, x$$

(Goodfellow 2016)

首先先明确几个概念，标量。标量就是一个数字，整数、实数、有理数这些都属于标量。然后本书中用斜体小写字母表示标量。

# Vectors

- A vector is a 1-D array of numbers:

$$\boldsymbol{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}. \quad (2.1)$$

- Can be real, binary, integer, etc.
- Example notation for type and size:

$$\mathbb{R}^n$$

(Goodfellow 2016)

向量。向量是一组数的有序排列的集合 ( 数组 ), 一个向量里面可以有 N 个标量。物理学中也管他叫做矢量。

举个栗子：



雀巢有一款雪糕，就是图片中这个。里面的每一块雪糕单元都是一个标量，然后把他们有序地放到这个盒子里，就构成了一盒雪糕，这一盒雪糕就是一个向量。本书中用斜体加粗小写字母来表示向量。

# Matrices

- A matrix is a 2-D array of numbers:

$$\begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \end{bmatrix} \quad (2.2)$$

Row  
Column

- Example notation for type and shape:

$$\mathbf{A} \in \mathbb{R}^{m \times n}$$

(Goodfellow 2016)

矩阵，矩阵是 2 维的标量构成的数组，仍然用栗子来理解它：



每一个雪糕个体都是一个标量，然后把他们一盒一盒摆出来，构成的平面就是矩阵。矩阵用行和列来标定内容，比如，二行三列，就是对应的白色雪糕块代表的标量。大家可能注意到了，如果把一盒雪糕（向量）当作基本组成单位，矩阵是向量的一维阵列。如第三行就是蓝

色盒子雪糕代表的向量。矩阵用斜体加粗大写字母来表示。

# Tensors

- A tensor is an array of numbers, that may have
  - zero dimensions, and be a scalar
  - one dimension, and be a vector
  - two dimensions, and be a matrix
  - or more dimensions.

(Goodfellow 2016)

张量。张量其实是对上述所有量的一个推广形式，零阶张量，就是标量；一阶张量是向量；二阶张量是矩阵，以此类推。不过由于低阶张量都有对应名字，一般只有 3 阶以上我们称之为张量（高等流体力学中张量特指我们这里的三阶张量）。

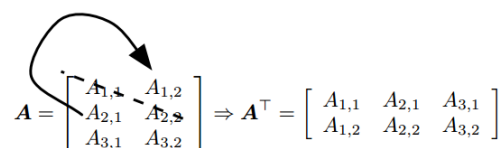
三阶张量举例：



其实就是从一排雪糕变成一箱雪糕，除了行列还引入了层（第三维度）的概念。更高阶的雪糕栗子大家自己想去吧，再放图估计都饿了一会儿。

# Matrix Transpose

$$(\mathbf{A}^\top)_{i,j} = A_{j,i}. \quad (2.3)$$



$$\mathbf{A} = \begin{bmatrix} A_{1,1} & A_{1,2} \\ A_{2,1} & A_{2,2} \\ A_{3,1} & A_{3,2} \end{bmatrix} \Rightarrow \mathbf{A}^\top = \begin{bmatrix} A_{1,1} & A_{2,1} & A_{3,1} \\ A_{1,2} & A_{2,2} & A_{3,2} \end{bmatrix}$$

Figure 2.1: The transpose of the matrix can be thought of as a mirror image across the main diagonal.

$$(\mathbf{AB})^\top = \mathbf{B}^\top \mathbf{A}^\top. \quad (2.9)$$

(Goodfellow 2016)

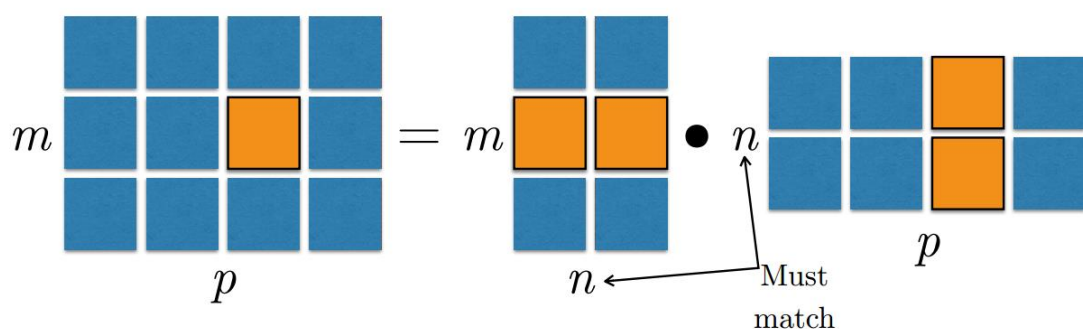
矩阵转置，说白了就是把矩阵沿对角线镜像一下，图中给出的是常规矩阵转置（非方阵，方阵）。然后矩阵转置有一个特性，即公式 2.9，两个矩阵相乘（矩阵相乘不具有交换律，即  $\mathbf{A} \times \mathbf{B} \neq \mathbf{B} \times \mathbf{A}$ ）然后转置，结果就等于两个矩阵各自转置后交换位置相乘：

$$(\mathbf{AB})^\top = \mathbf{B}^\top \mathbf{A}^\top.$$

# Matrix (Dot) Product

$$C = AB. \quad (2.4)$$

$$C_{i,j} = \sum_k A_{i,k} B_{k,j}. \quad (2.5)$$



(Goodfellow 2016)

矩阵点积，要求相乘的两个阵，前面阵的列数和后面阵的行数严格相等。矩阵点积是我们实现卷积的基本形式。点积公式 2.4，2.5：

$$C = AB.$$

$$C_{i,j} = \sum_k A_{i,k} B_{k,j}.$$

切记，矩阵点积并不是两矩阵对应元素相乘两矩阵对应元素相乘运算叫做 Hadamard 积：

$$C = A \odot B$$

$$C_{i,j} = A_{i,j} \times B_{i,j}$$



# Identity Matrix

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Figure 2.2: *Example identity matrix:* This is  $\boldsymbol{I}_3$ .

$$\forall \boldsymbol{x} \in \mathbb{R}^n, \boldsymbol{I}_n \boldsymbol{x} = \boldsymbol{x}. \tag{2.20}$$

(Goodfellow 2016)

单位矩阵，对角线元素为 1，其余元素均为 0 的方阵。任何矩阵（向量）与单位矩阵点积结果仍等于原矩阵（向量）。

# Systems of Equations

$$\mathbf{Ax} = \mathbf{b} \quad (2.11)$$

expands to

$$\mathbf{A}_{1,:}\mathbf{x} = b_1 \quad (2.12)$$

$$\mathbf{A}_{2,:}\mathbf{x} = b_2 \quad (2.13)$$

$$\dots \quad (2.14)$$

$$\mathbf{A}_{m,:}\mathbf{x} = b_m \quad (2.15)$$

(Goodfellow 2016)

矩阵点积可以代表传统的方程组 ,PPT 中不是很形象 ,这里我举个 3 元一次方程组的栗子 :

$$\mathbf{Ax} = \mathbf{b} \quad \mathbf{A} = \begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix} \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

等价于 :



$$\begin{cases} A_{11}x_1 + A_{12}x_2 + A_{13}x_3 = b_1 \\ A_{21}x_1 + A_{22}x_2 + A_{23}x_3 = b_2 \\ A_{31}x_1 + A_{32}x_2 + A_{33}x_3 = b_3 \end{cases}$$

# Solving Systems of Equations

- A linear system of equations can have:
  - No solution
  - Many solutions
  - Exactly one solution: this means multiplication by the matrix is an invertible function

(Goodfellow 2016)

通过上述的替换，我们就很容易把解方程组问题转化为矩阵运算问题，我们可以很容易地通过矩阵来判断方程组是否有解，有几个解地问题（如当系数矩阵为可逆矩阵，即满秩，同时结果阵不为零矩阵时，方程组有唯一非零解）。

# Matrix Inversion

- Matrix inverse:

$$\mathbf{A}^{-1}\mathbf{A} = \mathbf{I}_n. \quad (2.21)$$

- Solving a system using an inverse:

$$\mathbf{A}\mathbf{x} = \mathbf{b} \quad (2.22)$$

$$\mathbf{A}^{-1}\mathbf{A}\mathbf{x} = \mathbf{A}^{-1}\mathbf{b} \quad (2.23)$$

$$\mathbf{I}_n\mathbf{x} = \mathbf{A}^{-1}\mathbf{b} \quad (2.24)$$

- Numerically unstable, but useful for abstract analysis

(Goodfellow 2016)

矩阵求逆，一个矩阵与他的逆矩阵点积结果是一个单位矩阵（求逆矩阵地方法我们这里不作说明）。通过逆矩阵我们很容易就可以求得方程组的解（同理这个方法可以求逆卷积）。

# Invertibility

- Matrix can't be inverted if...
  - More rows than columns
  - More columns than rows
  - Redundant rows/columns (“linearly dependent”, “low rank”)

(Goodfellow 2016)

如何判断一个矩阵是否可逆：

- 1、该矩阵必须为方阵；
- 2、没有线性相关的行/列（行列线性无关/满秩）

$$z = \alpha x + (1 - \alpha)y$$

何时线性无关呢，很简单，就是不存在上图的结果（上图被称作线性相关）。

# Norms

- Functions that measure how “large” a vector is
- Similar to a distance between zero and the point represented by the vector
  - $f(\mathbf{x}) = 0 \Rightarrow \mathbf{x} = \mathbf{0}$
  - $f(\mathbf{x} + \mathbf{y}) \leq f(\mathbf{x}) + f(\mathbf{y})$  (the *triangle inequality*)
  - $\forall \alpha \in \mathbb{R}, f(\alpha \mathbf{x}) = |\alpha|f(\mathbf{x})$

(Goodfellow 2016)

正则。作者在 ppt 中将正则定义为用来测量一个向量“大小”的方法。规则类似于用向量来表示的点举例零点的距离（该页 PPT 中的正则特指 L2 正则，推广就是两点举例公式，矩阵论中我们将 L2 正则称之为欧几里得范数，也叫做二范数）。

# Norms

- $L^p$  norm

$$\|\mathbf{x}\|_p = \left( \sum_i |x_i|^p \right)^{\frac{1}{p}}$$

- Most popular norm: L2 norm,  $p=2$

- L1 norm,  $p=1$ :  $\|\mathbf{x}\|_1 = \sum_i |x_i|$ . (2.31)

- Max norm, infinite  $p$ :  $\|\mathbf{x}\|_\infty = \max_i |x_i|$ . (2.32)

(Goodfellow 2016)

推广后的正则 ( N 维空间中两点距离公式 )。公式就不做说明了 , L1 正则就是数轴 ( 1 维空间 ) 上两点距离 , L2 正则就是笛卡尔坐标系 ( 2 维空间 ) 下的两点距离。这里还提出了无穷范数 , 即寻找向量中的极值。

# Special Matrices and Vectors

- Unit vector:

$$\|\boldsymbol{x}\|_2 = 1. \quad (2.36)$$

- Symmetric Matrix:

$$\boldsymbol{A} = \boldsymbol{A}^\top. \quad (2.35)$$

- Orthogonal matrix:

$$\begin{aligned} \boldsymbol{A}^\top \boldsymbol{A} &= \boldsymbol{A} \boldsymbol{A}^\top = \boldsymbol{I}. \\ \boldsymbol{A}^{-1} &= \boldsymbol{A}^\top \end{aligned} \quad (2.37)$$

(Goodfellow 2016)

然后是一些特殊的矩阵和向量：

- 1、单位向量：即模长为 1 的向量；
- 2、对称矩阵：即逆矩阵等于原矩阵的矩阵；
- 3、正交矩阵：逆矩阵与转置矩阵相等的矩阵。



# Eigendecomposition

- Eigenvector and eigenvalue:

$$\mathbf{A}\mathbf{v} = \lambda\mathbf{v}. \quad (2.39)$$

- Eigendecomposition of a diagonalizable matrix:

$$\mathbf{A} = \mathbf{V}\text{diag}(\boldsymbol{\lambda})\mathbf{V}^{-1}. \quad (2.40)$$

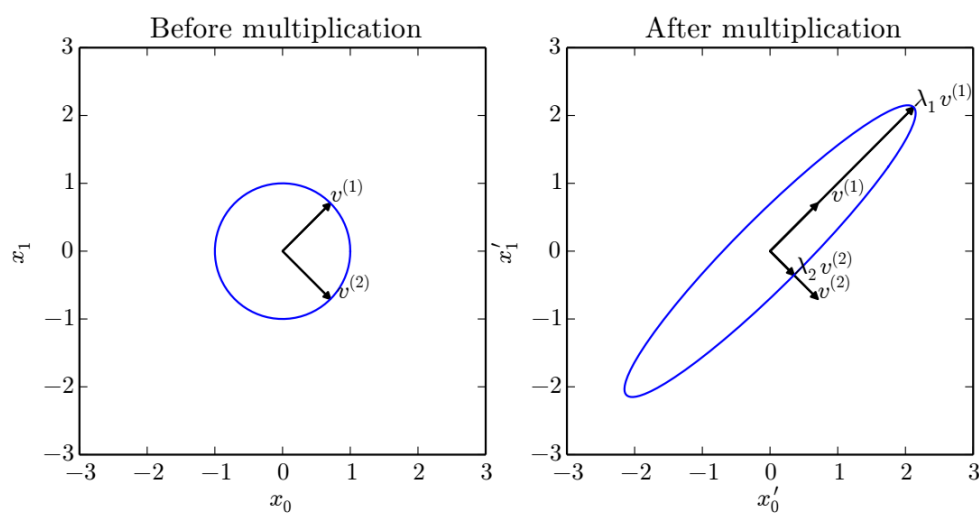
- Every real symmetric matrix has a real, orthogonal eigendecomposition:

$$\mathbf{A} = \mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^{\top} \quad (2.41)$$

(Goodfellow 2016)

特征分解。只有可对角化矩阵才能够进行特征值分解。一个矩阵与其对应的特征向量的点积等于其特征值与该特征向量的乘积（式 2.39）。我们可以根据这个特性将矩阵进行分解，分解为其特征向量与特征值的矩阵点积（式 2.40）。实对称矩阵的特征分解结果是正交的（式 2.41）。

# Effect of Eigenvalues



(Goodfellow 2016)

上图是特征分解的一个几何效果。我们定义一个特征向量为  $v_1, v_2$  的矩阵  $A$ ，我们画一个由单位向量组成的单位圆（单位矩阵  $U$ ），如图左，然后我们画出两矩阵的点积（ $AU$ ），我们会发现单位圆被扭曲了，扭曲的方向即两个特征向量的方向，缩放大小为特征值的大小。

# Singular Value Decomposition

- Similar to eigendecomposition
- More general; matrix need not be square

$$\mathbf{A} = \mathbf{U} \mathbf{D} \mathbf{V}^{\top}. \quad (2.43)$$

(Goodfellow 2016)

奇异值分解 ( SVD 分解 )。这里不对分解过程进行描述。奇异值分解与特征值分解类似，也是拆分矩阵的一种方法，但是适用范围更广，非方阵也可以进行 SVD 分解。

Perhaps the most useful feature of the SVD is that we can use it to partially generalize matrix inversion to non-square matrices.

也许 SVD 的最有用的用处是我们可以使用它来将矩阵求逆部分推广到非正方形矩阵。

# Moore-Penrose Pseudoinverse

$$\boldsymbol{x} = \boldsymbol{A}^+ \boldsymbol{y}$$

- If the equation has:
  - Exactly one solution: this is the same as the inverse.
  - No solution: this gives us the solution with the smallest error  $\|\boldsymbol{Ax} - \boldsymbol{y}\|_2$ .
  - Many solutions: this gives us the solution with the smallest norm of  $\boldsymbol{x}$ .

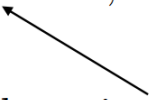
(Goodfellow 2016)

Moore-Penrose 逆，国内的教材上管这个方法叫做广义逆。值得注意的一点，这个方法的逆用得是单词 Pseudoinverse，这个单词意思是伪逆，也就是说他是逆的推广，即使是上面我们说不能求逆的矩阵（不满秩），我们也可以求伪逆，这也是为什么我们把这个方法叫做“广义”逆的原因。有了广义逆，我们就可以用它来解出任何的方程组了。

# Computing the Pseudoinverse

The SVD allows the computation of the pseudoinverse:

$$\mathbf{A}^+ = \mathbf{V} \mathbf{D}^+ \mathbf{U}^\top, \quad (2.47)$$



Take reciprocal of non-zero entries

(Goodfellow 2016)

求广义逆的 SVD 方法，只不过上面我们用 SVD 来分解矩阵，这里我们用 SVD 来“合成”广义逆矩阵。

# Trace

$$\text{Tr}(\mathbf{A}) = \sum_i \mathbf{A}_{i,i}. \quad (2.48)$$

$$\text{Tr}(\mathbf{ABC}) = \text{Tr}(\mathbf{CAB}) = \text{Tr}(\mathbf{BCA}) \quad (2.51)$$

(Goodfellow 2016)

然后就是矩阵的迹 ( Trace ), 英文叫做跟踪操作, 不知道为啥中文就叫迹了, 莫非是足迹的意思? 迹操作就是把一个矩阵对角线求和, 比较有意思的是迹运算中的矩阵点积神奇的具有了交换律。使用迹运算可以快速地求出矩阵的 Frobenius 范数 ( 即矩阵元素绝对值的平方和再开平方, 相当于对每一行/列取二范数 ):

$$\|\mathbf{A}\|_F = \sqrt{\text{Tr}(\mathbf{AA}^\top)}.$$

## 2.12 Example: Principal Components Analysis

书中最后给出了一个运用线性代数的算法栗子, PPT 中没有, 这里就截取书的图片来讲一下。示例算法叫做主成分分析 ( Principal Components Analysis ), 应该群里好多大神都接

触过，是一种用来提取数据主要特征（成分）的方法，在保留主要成分的基础上对数据进行压缩的过程（像不像上一次讲的用网络参数来存储关于数据“记忆”的过程？）。

$$\begin{array}{c} \mathbf{x}^{(i)} \in \mathbb{R}^n \\ \Downarrow \\ \mathbf{c}^{(i)} \in \mathbb{R}^l \\ l < n \end{array} \quad \left\{ \begin{array}{l} \mathbf{c} = f(\mathbf{x}) \\ \mathbf{x} \approx g(\mathbf{c}) = g(f(\mathbf{x})) \end{array} \right.$$

先来看我们的目标，将  $N$  维的数据无损压缩到  $L$  维，构造出编码器  $f$  及解码器  $g$ 。为了方便运算，这里我们直接用矩阵点积来作为我们的解码器：

$$g(\mathbf{c}) = \mathbf{D}\mathbf{c}, \text{ where } \mathbf{D} \in \mathbb{R}^{n \times l}$$

然后构造出我们的优化目标（解出  $f$ ），这里我们用平方二范数来计算我们的误差（反向传播思想，是不是和神经网络一样）：

$$\mathbf{c}^* = \arg \min_{\mathbf{c}} \|\mathbf{x} - g(\mathbf{c})\|_2^2.$$

将上式展开得到：

$$\begin{aligned} & (\mathbf{x} - g(\mathbf{c}))^\top (\mathbf{x} - g(\mathbf{c})) \\ &= \mathbf{x}^\top \mathbf{x} - \mathbf{x}^\top g(\mathbf{c}) - g(\mathbf{c})^\top \mathbf{x} + g(\mathbf{c})^\top g(\mathbf{c}) \\ &= \mathbf{x}^\top \mathbf{x} - 2\mathbf{x}^\top g(\mathbf{c}) + g(\mathbf{c})^\top g(\mathbf{c}) \end{aligned}$$

$\mathbf{x}^\top \mathbf{x}$  一项与  $\mathbf{c}$  无关，将该项删除，得到：

$$\mathbf{c}^* = \arg \min_{\mathbf{c}} -2\mathbf{x}^\top g(\mathbf{c}) + g(\mathbf{c})^\top g(\mathbf{c}).$$

然后把我们刚才构造的 D 带入得到：

$$\begin{aligned}\mathbf{c}^* &= \arg \min_{\mathbf{c}} -2\mathbf{x}^\top \mathbf{D}\mathbf{c} + \mathbf{c}^\top \mathbf{D}^\top \mathbf{D}\mathbf{c} \\ &= \arg \min_{\mathbf{c}} -2\mathbf{x}^\top \mathbf{D}\mathbf{c} + \mathbf{c}^\top \mathbf{I}_l \mathbf{c} \\ &= \arg \min_{\mathbf{c}} -2\mathbf{x}^\top \mathbf{D}\mathbf{c} + \mathbf{c}^\top \mathbf{c}\end{aligned}$$

高中数学老师告诉我们，求一个函数的极值，要求他的导数( 梯度 )，极值位置导数( 梯度 )

为 0，所以我们对 c 求导解出极值点：

$$\begin{aligned}\nabla_{\mathbf{c}}(-2\mathbf{x}^\top \mathbf{D}\mathbf{c} + \mathbf{c}^\top \mathbf{c}) &= \mathbf{0} \\ -2\mathbf{D}^\top \mathbf{x} + 2\mathbf{c} &= \mathbf{0} \\ \mathbf{c} &= \mathbf{D}^\top \mathbf{x}. \\ f(\mathbf{x}) &= \mathbf{D}^\top \mathbf{x}.\end{aligned}$$

然后我们需要做的就是求出 D 的具体内容了，根据上面计算的我们可以给出我们计算出的

重建 x 公式：



$$r(\boldsymbol{x}) = g(f(\boldsymbol{x})) = \boldsymbol{D}\boldsymbol{D}^\top \boldsymbol{x}.$$

同样用上面反向传播的思路求得最优  $\boldsymbol{D}$  :

$$\boldsymbol{D}^* = \arg \min_{\boldsymbol{D}} \sqrt{\sum_{i,j} \left( x_j^{(i)} - r(\boldsymbol{x}^{(i)})_j \right)^2} \text{ subject to } \boldsymbol{D}^\top \boldsymbol{D} = \boldsymbol{I}_l$$

后面同样将二范数替换为平方二范数。为了推导我们的结果，我们从  $l=1$  开始推到，即  $\boldsymbol{D}=\boldsymbol{d}$  是一个一维向量。

$$\boldsymbol{d}^* = \arg \min_{\boldsymbol{d}} \sum_i \|\boldsymbol{x}^{(i)} - \boldsymbol{d}\boldsymbol{d}^\top \boldsymbol{x}^{(i)}\|_2^2 \text{ subject to } \|\boldsymbol{d}\|_2 = 1.$$

公式变形一下 ( 汽车人变身 ):

$$\boldsymbol{d}^* = \arg \min_{\boldsymbol{d}} \sum_i \|\boldsymbol{x}^{(i)} - \boldsymbol{d}^\top \boldsymbol{x}^{(i)} \boldsymbol{d}\|_2^2 \text{ subject to } \|\boldsymbol{d}\|_2 = 1,$$

$$\boldsymbol{d}^* = \arg \min_{\boldsymbol{d}} \sum_i \|\boldsymbol{x}^{(i)} - \boldsymbol{x}^{(i)\top} \boldsymbol{d} \boldsymbol{d}\|_2^2 \text{ subject to } \|\boldsymbol{d}\|_2 = 1.$$

大大的求和号看的我们好难受，我们这里定义一个矩阵，通过矩阵运算来遍历所有的  $\boldsymbol{x}$ ，以便我们可以取消掉求和号，方便我们的运算：

$$\boldsymbol{X}_{i,:} = \boldsymbol{x}^{(i)\top}$$

我们将矩阵的每一行都装入一个  $\boldsymbol{x}$  ( 排排坐吃果果 )，大家类比刚才的雪糕栗子。

这样我们的式子就变为：

$$\boldsymbol{d}^* = \arg \min_{\boldsymbol{d}} \|\boldsymbol{X} - \boldsymbol{X} \boldsymbol{d} \boldsymbol{d}^\top\|_F^2 \text{ subject to } \boldsymbol{d}^\top \boldsymbol{d} = 1.$$

F 范数登场了，用我们刚才讲地迹方法来把它拆开：

$$\begin{aligned}
& \arg \min_{\mathbf{d}} \|\mathbf{X} - \mathbf{X} \mathbf{d} \mathbf{d}^\top\|_F^2 \\
&= \arg \min_{\mathbf{d}} \text{Tr} \left( \left( \mathbf{X} - \mathbf{X} \mathbf{d} \mathbf{d}^\top \right)^\top \left( \mathbf{X} - \mathbf{X} \mathbf{d} \mathbf{d}^\top \right) \right) \\
&= \arg \min_{\mathbf{d}} \text{Tr}(\mathbf{X}^\top \mathbf{X} - \mathbf{X}^\top \mathbf{X} \mathbf{d} \mathbf{d}^\top - \mathbf{d} \mathbf{d}^\top \mathbf{X}^\top \mathbf{X} + \mathbf{d} \mathbf{d}^\top \mathbf{X}^\top \mathbf{X} \mathbf{d} \mathbf{d}^\top) \\
&= \arg \min_{\mathbf{d}} \text{Tr}(\mathbf{X}^\top \mathbf{X}) - \text{Tr}(\mathbf{X}^\top \mathbf{X} \mathbf{d} \mathbf{d}^\top) - \text{Tr}(\mathbf{d} \mathbf{d}^\top \mathbf{X}^\top \mathbf{X}) + \text{Tr}(\mathbf{d} \mathbf{d}^\top \mathbf{X}^\top \mathbf{X} \mathbf{d} \mathbf{d}^\top) \\
&= \arg \min_{\mathbf{d}} -\text{Tr}(\mathbf{X}^\top \mathbf{X} \mathbf{d} \mathbf{d}^\top) - \text{Tr}(\mathbf{d} \mathbf{d}^\top \mathbf{X}^\top \mathbf{X}) + \text{Tr}(\mathbf{d} \mathbf{d}^\top \mathbf{X}^\top \mathbf{X} \mathbf{d} \mathbf{d}^\top) \\
&= \arg \min_{\mathbf{d}} -2 \text{Tr}(\mathbf{X}^\top \mathbf{X} \mathbf{d} \mathbf{d}^\top) + \text{Tr}(\mathbf{d} \mathbf{d}^\top \mathbf{X}^\top \mathbf{X} \mathbf{d} \mathbf{d}^\top) \\
&= \arg \min_{\mathbf{d}} -2 \text{Tr}(\mathbf{X}^\top \mathbf{X} \mathbf{d} \mathbf{d}^\top) + \text{Tr}(\mathbf{X}^\top \mathbf{X} \mathbf{d} \mathbf{d}^\top \mathbf{d} \mathbf{d}^\top) \\
&= \arg \min_{\mathbf{d}} -2 \text{Tr}(\mathbf{X}^\top \mathbf{X} \mathbf{d} \mathbf{d}^\top) + \text{Tr}(\mathbf{X}^\top \mathbf{X} \mathbf{d} \mathbf{d}^\top) \text{ subject to } \mathbf{d}^\top \mathbf{d} = 1 \\
&= \arg \min_{\mathbf{d}} -\text{Tr}(\mathbf{X}^\top \mathbf{X} \mathbf{d} \mathbf{d}^\top) \text{ subject to } \mathbf{d}^\top \mathbf{d} = 1 \\
&= \arg \max_{\mathbf{d}} \text{Tr}(\mathbf{X}^\top \mathbf{X} \mathbf{d} \mathbf{d}^\top) \text{ subject to } \mathbf{d}^\top \mathbf{d} = 1 \\
&= \arg \max_{\mathbf{d}} \text{Tr}(\mathbf{d}^\top \mathbf{X}^\top \mathbf{X} \mathbf{d}) \text{ subject to } \mathbf{d}^\top \mathbf{d} = 1
\end{aligned}$$

然后终于化到了一个比较好看地式子。是不是长的很像特征值分解( 把  $\mathbf{X}^\top \mathbf{X}$  看成特征值阵 ), 我们可以解得  $\mathbf{d}$  的所有解即为  $\mathbf{X}^\top \mathbf{X}$  的特征向量, 这里我们需要的最大  $\mathbf{d}$  就是  $\mathbf{X}^\top \mathbf{X}$  对应最大特征值的特征向量。将结论进行推广,  $\mathbf{D}$  即为对应  $\mathbf{X}^\top \mathbf{X}$  对应最大的  $l$  个特征值的特征向量组成的编码阵,

今天的内容就这么多, 可能有点难, 希望各位读者多多思考, 可能第一遍真的不太容易懂, 希望大家多多思考

关注极市平台获取更多干货及在线分享信息



关注 AG Group 获取好玩的科技信息

