

Mathematics Notes
for
Computer Science
Information Technology

Hazer-BJTU

2024 / 2 / 16

目录

0.1	深度学习中的线性代数/概率论	4
0.1.1	多元函数微分	4
0.1.2	线性回归模型的解析解	5
0.1.3	SVD奇异值分解	6
0.1.4	极大似然估计与最小化交叉熵损失	7
0.2	算法/基础数学	7
0.2.1	离散傅里叶变换DFT与快速傅里叶变换FFT	7
0.3	算法竞赛中的数论与组合数学	10

0.1 深度学习中的线性代数/概率论

0.1.1 多元函数微分

考虑定义在 \mathbb{R}^n 上的函数 f ，其输出为一个向量 $\mathbf{y} \in \mathbb{R}^m$ ，如果存在线性函数 L ，使得：

$$f(\mathbf{x} + \mathbf{h}) = f(\mathbf{x}) + L(\mathbf{h}) + O(\|\mathbf{h}\|_2)$$

其中线性函数 L 满足：

$$L(\mathbf{x} + \mathbf{y}) = L(\mathbf{x}) + L(\mathbf{y})$$

$$L(\lambda \cdot \mathbf{x}) = \lambda \cdot L(\mathbf{x}), \lambda \in \mathbb{R}$$

那么我们就认为该函数 f 是可微的，一般来说，我们可以将线性函数 L 简单理解为线性变换，如果我们限制函数 f 的输出为一个实数 $y \in \mathbb{R}$ ，则微分也可以被表示为如下形式：

$$f(\mathbf{x} + \mathbf{h}) = f(\mathbf{x}) + \mathbf{w}^\top \mathbf{h} + O(\|\mathbf{h}\|_2), \mathbf{w} \in \mathbb{R}^n$$

一个基本的事实是可微 \Rightarrow 偏导数存在，因为：

$$\begin{aligned} \frac{f(\mathbf{x} + \mathbf{h}_i) - f(\mathbf{x})}{\Delta \mathbf{x}_i} &= \frac{\mathbf{w}_i \cdot \Delta \mathbf{x}_i}{\Delta \mathbf{x}_i} + \frac{O(\Delta \mathbf{x}_i)}{\Delta \mathbf{x}_i} = \mathbf{w}_i + \frac{O(\Delta \mathbf{x}_i)}{\Delta \mathbf{x}_i} \\ \Rightarrow \lim_{\Delta \mathbf{x}_i \rightarrow 0} \frac{f(\mathbf{x} + \mathbf{h}_i) - f(\mathbf{x})}{\Delta \mathbf{x}_i} &= \mathbf{w}_i + \lim_{\Delta \mathbf{x}_i \rightarrow 0} \frac{O(\Delta \mathbf{x}_i)}{\Delta \mathbf{x}_i} = \mathbf{w}_i \\ \Rightarrow \frac{\partial f}{\partial \mathbf{x}_i} &= \mathbf{w}_i \end{aligned}$$

由此可见，实际上向量 \mathbf{w} 就是由函数 f 关于各分量的偏导数构成的：

$$\mathbf{w} = \left(\frac{\partial f}{\partial \mathbf{x}_1}, \frac{\partial f}{\partial \mathbf{x}_2}, \frac{\partial f}{\partial \mathbf{x}_3}, \dots, \frac{\partial f}{\partial \mathbf{x}_n} \right)^\top$$

定义对于向量 $\mathbf{x} \in \mathbb{R}^n$ ： $d\mathbf{x} = (dx_1, dx_2, dx_3, \dots, dx_n)$ ，则根据全微分公式可以得出如下关系：

$$d\mathbf{x}^\top \mathbf{x} = 2\mathbf{x}^\top d\mathbf{x}$$

$$d(\mathbf{x} + \mathbf{y}) = d\mathbf{x} + d\mathbf{y}$$

$$d\mathbf{A}\mathbf{x} = \mathbf{A}d\mathbf{x}$$

$$d\mathbf{x}^\top \mathbf{A}\mathbf{x} = 2\mathbf{x}^\top \mathbf{A}d\mathbf{x}$$

在此只证明最后一条，注意到：

$$\begin{aligned} \mathbf{x}^\top \mathbf{A}\mathbf{x} &= \sum_{i=1}^n \sum_{j=1}^n \mathbf{A}_{i,j} \mathbf{x}_i \mathbf{x}_j \\ \frac{\partial}{\partial \mathbf{x}_i} \mathbf{x}^\top \mathbf{A}\mathbf{x} &= 2\mathbf{A}_{i,i} \mathbf{x}_i + 2 \sum_{1 \leq j \leq n, j \neq i} \mathbf{A}_{i,j} \mathbf{x}_j = 2 \sum_{j=1}^n \mathbf{A}_{i,j} \mathbf{x}_j \\ \Rightarrow d\mathbf{x}^\top \mathbf{A}\mathbf{x} &= 2 \sum_{i=1}^n \sum_{j=1}^n \mathbf{A}_{i,j} \mathbf{x}_j d\mathbf{x}_i = 2\mathbf{x}^\top \mathbf{A}d\mathbf{x} \end{aligned}$$

与一元函数同理，如果上述函数 f 满足二阶偏导数连续的条件，则我们也可以利用Hessian矩阵做出更高阶的估计：

$$f(\mathbf{x} + \mathbf{h}) = f(\mathbf{x}) + \mathbf{w}^\top \mathbf{h} + \frac{1}{2} \mathbf{h}^\top \mathbf{H} \mathbf{h} + O(\|\mathbf{h}\|_2^2)$$

其中Hessian矩阵的形式为：

$$\mathbf{H}_{i,j} = \frac{\partial^2 f}{\partial \mathbf{x}_i \partial \mathbf{x}_j}$$

0.1.2 线性回归模型的解析解

一般的线性模型可以被描述为以下形式，其中 $\hat{y} \in \mathbb{R}, \mathbf{x} \in \mathbb{R}^d, \mathbf{w} \in \mathbb{R}^d$ ：

$$\hat{y} = \mathbf{w}^\top \mathbf{x} + \mathbf{b}$$

而对于批量的样本数据，使用 $\mathbf{X} \in \mathbb{R}^{n \times d}$ 表示 n 组样本， $\hat{\mathbf{Y}} \in \mathbb{R}^n$ 表示对于数据集上所有样本的预测结果向量，则可以进行如下矩阵表示：

$$\hat{\mathbf{Y}} = \mathbf{X}\mathbf{w} + \mathbf{B}$$

对于真实的数据 \mathbf{Y} ，线性回归要求我们最小化损失 $\|\hat{\mathbf{Y}} - \mathbf{Y}\|_2$ ，这是一个十分简单的优化问题，存在解析解，证明如下：

$$\begin{aligned} \|\hat{\mathbf{Y}} - \mathbf{Y}\|_2 &= \sqrt{(\hat{\mathbf{Y}} - \mathbf{Y})^\top (\hat{\mathbf{Y}} - \mathbf{Y})} \\ &= \sqrt{(\mathbf{X}\mathbf{w} + \mathbf{B} - \mathbf{Y})^\top (\mathbf{X}\mathbf{w} + \mathbf{B} - \mathbf{Y})} \end{aligned}$$

故问题转化为最小化 $(\mathbf{X}\mathbf{w} + \mathbf{B} - \mathbf{Y})^\top (\mathbf{X}\mathbf{w} + \mathbf{B} - \mathbf{Y})$ ，这是一个二次型，我们对于 \mathbf{w} 求导：

$$\begin{aligned} d(\mathbf{X}\mathbf{w} + \mathbf{B} - \mathbf{Y})^\top (\mathbf{X}\mathbf{w} + \mathbf{B} - \mathbf{Y}) &= 2(\mathbf{X}\mathbf{w} + \mathbf{B} - \mathbf{Y})^\top d(\mathbf{X}\mathbf{w} + \mathbf{B} - \mathbf{Y}) \\ &= 2(\mathbf{X}\mathbf{w} + \mathbf{B} - \mathbf{Y})^\top \mathbf{X} d\mathbf{w} \\ &= 0 \end{aligned}$$

故可以得到：

$$(\mathbf{X}\mathbf{w} + \mathbf{B} - \mathbf{Y})^\top \mathbf{X} = \mathbf{O}$$

等式两边同时取转置可知：

$$\begin{aligned} \mathbf{X}^\top (\mathbf{X}\mathbf{w} + \mathbf{B} - \mathbf{Y}) &= \mathbf{O} \\ \mathbf{X}^\top \mathbf{X}\mathbf{w} &= \mathbf{X}^\top (\mathbf{Y} - \mathbf{B}) \\ \mathbf{w} &= (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top (\mathbf{Y} - \mathbf{B}) \end{aligned}$$

即可得到参数的最优解，前提是矩阵 $\mathbf{X}^\top \mathbf{X}$ 可逆。

0.1.3 SVD奇异值分解

一般来说，任何实矩阵 $\mathbf{A} \in \mathbb{R}^{n \times m}$ 都可以被无条件地分解为如下三个矩阵的乘积：

$$\mathbf{A}_{n \times m} = \mathbf{U}_{n \times n} \mathbf{\Sigma}_{n \times m} \mathbf{V}_{m \times m}^\top$$

其中 \mathbf{U}, \mathbf{V} 均为正交矩阵，并且 $\mathbf{\Sigma}$ 满足：

$$\Sigma_{i,j} = \begin{cases} \sqrt{\lambda_i} & i = j \\ 0 & i \neq j \end{cases}$$

考虑 $\mathbf{A}^\top \mathbf{A}$ ，这是一个实对称矩阵，故其一定可以被正交对角化，也即存在正交矩阵 \mathbf{V} ，使得：

$$\mathbf{A}^\top \mathbf{A} = \mathbf{V} \mathbf{\Lambda} \mathbf{V}^\top$$

其中：

$$\mathbf{\Lambda}_{m \times m} = \begin{bmatrix} \lambda_1 & & & \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \lambda_m \end{bmatrix}$$

考虑如下一组向量，我们断言它们之间是互相正交的：

$$\frac{\mathbf{A}\mathbf{v}_1}{\sqrt{\lambda_1}}, \frac{\mathbf{A}\mathbf{v}_2}{\sqrt{\lambda_2}}, \frac{\mathbf{A}\mathbf{v}_3}{\sqrt{\lambda_3}}, \dots, \frac{\mathbf{A}\mathbf{v}_m}{\sqrt{\lambda_m}}$$

证明如下：

$$\frac{\mathbf{A}\mathbf{v}_i}{\sqrt{\lambda_i}} \cdot \frac{\mathbf{A}\mathbf{v}_j}{\sqrt{\lambda_j}} = \frac{\mathbf{v}_i^\top \mathbf{A}^\top \mathbf{A} \mathbf{v}_j}{\sqrt{\lambda_i \lambda_j}} = \frac{\lambda_j \mathbf{v}_i^\top \mathbf{v}_j}{\sqrt{\lambda_i \lambda_j}} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases}$$

若 $m \geq n$ ，考虑如下矩阵：

$$\mathbf{U}_{n \times n} = \left(\frac{\mathbf{A}\mathbf{v}_1}{\sqrt{\lambda_1}}, \frac{\mathbf{A}\mathbf{v}_2}{\sqrt{\lambda_2}}, \frac{\mathbf{A}\mathbf{v}_3}{\sqrt{\lambda_3}}, \dots, \frac{\mathbf{A}\mathbf{v}_n}{\sqrt{\lambda_n}} \right)$$

根据上述证明， \mathbf{U} 是正交矩阵，并且满足：

$$\mathbf{U}\mathbf{\Sigma} = \mathbf{A}\mathbf{V}$$

$$\mathbf{A} = \mathbf{A}\mathbf{V}\mathbf{V}^\top = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$$

若 $m < n$ ，我们可以反过来对 \mathbf{A}^\top 做奇异值分解，也可以得到相同的结果，奇异值分解告诉我们：任何线性变换都可以被分解为一次旋转(旋转、反射或其复合)，一次维度变换及拉伸，一次旋转的复合。除此之外，其还可以被用于求一般矩阵的“逆”：

$$\mathbf{A} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\top$$

$$\mathbf{A}^+ = \mathbf{V}\mathbf{\Sigma}^+\mathbf{U}^\top$$

其中 $\mathbf{\Sigma}^+$ 由将 $\mathbf{\Sigma}$ 中非零元素取倒数后再转置得到。

0.1.4 极大似然估计与最小化交叉熵损失

0.2 算法/基础数学

0.2.1 离散傅里叶变换DFT与快速傅里叶变换FFT

对于数列 $\{a_n\}, \{b_n\}, 0 \leq n < N$, 我们可以如下定义其离散卷积:

$$(a * b)_k = \sum_{i=0}^k a_i b_{k-i} \quad 0 \leq k < N$$

我们记单位根 $e^{\frac{2k\pi i}{n}} = \omega_n^k$, 则可以如下定义其离散傅里叶变换及其逆变换:

$$\begin{aligned} DFT(a)_k &= \sum_{t=0}^{N-1} a_t \cdot \omega_N^{-kt} \\ a_k &= \frac{1}{N} \sum_{t=0}^{N-1} DFT(a)_t \cdot \omega_N^{kt} \end{aligned}$$

其中逆变换的证明如下:

$$\begin{aligned} \frac{1}{N} \sum_{t=0}^{N-1} DFT(a)_t \cdot \omega_N^{kt} &= \frac{1}{N} \sum_{t=0}^{N-1} \left(\sum_{u=0}^{N-1} a_u \cdot \omega_N^{-tu} \right) \cdot \omega_N^{kt} \\ &= \frac{1}{N} \sum_{t=0}^{N-1} \sum_{u=0}^{N-1} a_u \cdot \omega_N^{t(k-u)} \\ &= \frac{1}{N} \sum_{u=0}^{N-1} \sum_{t=0}^{N-1} a_u \cdot \omega_N^{t(k-u)} \end{aligned}$$

首先考虑如果 $u = k$, 则有以下式成立:

$$\begin{aligned} \omega_N^{t(k-u)} &= \omega_N^0 = 1 \\ \sum_{t=0}^{N-1} a_u \cdot \omega_N^{t(k-u)} &= Na_u = Na_k \end{aligned}$$

然后考虑如果 $u \neq k$, 注意到 $\omega_N^N = 1$, 则有以下式成立:

$$\begin{aligned} \sum_{t=0}^{N-1} a_u \cdot \omega_N^{t(k-u)} &= a_u \sum_{t=0}^{N-1} \omega_N^{t(k-u)} \\ &= a_u \cdot \frac{1 - \omega_N^{N(k-u)}}{1 - \omega_N^{k-u}} \\ &= 0 \end{aligned}$$

故综上所述, 逆变换得证:

$$\frac{1}{N} \sum_{t=0}^{N-1} DFT(a)_t \cdot \omega_N^{kt} = \frac{1}{N} \cdot Na_k = a_k$$

接着我们引入卷积定理的离散形式：

$$a * b = DFT^{-1}(DFT(a) \odot DFT(b))$$

为了证明上式，我们只需要证明：

$$(a * b)_k = DFT^{-1}(DFT(a) \odot DFT(b))_k \quad 0 \leq k < N$$

利用定义展开右式，同理可证：

$$\begin{aligned} & DFT^{-1}(DFT(a) \odot DFT(b))_k \\ &= \frac{1}{N} \sum_{t=0}^{N-1} (DFT(a) \odot DFT(b))_t \cdot \omega_N^{kt} \\ &= \frac{1}{N} \sum_{t=0}^{N-1} DFT(a)_t \cdot DFT(b)_t \cdot \omega_N^{kt} \\ &= \frac{1}{N} \sum_{t=0}^{N-1} \left(\sum_{n=0}^{N-1} a_n \cdot \omega_N^{-tn} \right) \cdot \left(\sum_{m=0}^{N-1} b_m \cdot \omega_N^{-tm} \right) \cdot \omega_N^{kt} \\ &= \frac{1}{N} \sum_{t=0}^{N-1} \left(\sum_{n=0}^{N-1} \sum_{m=0}^{N-1} a_n b_m \cdot \omega_N^{-t(n+m)} \right) \cdot \omega_N^{kt} \\ &= \frac{1}{N} \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} \sum_{t=0}^{N-1} a_n b_m \cdot \omega_N^{t(k-n-m)} \\ &= \frac{1}{N} \sum_{n=0}^{N-1} \sum_{m=0}^{N-1} [n+m=k] N a_n b_m \\ &= \sum_{n=0}^k a_n b_{k-n} \\ &= (a * b)_k \end{aligned}$$

快速傅里叶变换算法可以帮助我们高效地计算离散傅里叶变换：

$$\{a_n\} \xrightarrow{FFT} \{DFT(a)_n\}$$

结合卷积定理，我们得以加速多项式乘法至 $O(n \log n)$ 时间复杂度：

$$\begin{array}{ccc} \{a\}, \{b\} & \xrightarrow{O(n^2)} & \{a * b\} \\ \downarrow O(n \log n) & & \uparrow O(n \log n) \\ \{DFT(a)\}, \{DFT(b)\} & \xrightarrow{O(n)} & \{DFT(a) \odot DFT(b)\} \end{array}$$

为了简化问题，此处我们只讨论 $N = 2^K, K \in \mathbb{N}$ 的简单情形，为了计算离散傅里叶变换，我们的目标是计算下列数值：

$$\mathbf{F} = [F(w_n^0), F(w_n^{-1}), F(w_n^{-2}), \dots, F(w_n^{-(N-1)})]$$

$$F(x) = \sum_{t=0}^{N-1} a_t x^t \Rightarrow DFT(a)_k = F(\omega_N^{-k})$$

我们将函数 $F(x)$ 拆分为如下两个部分，注意到当 $N \neq 1$ 时其为偶数：

$$\begin{aligned} F(x) &= a_0 + a_1 x + a_2 x^2 + \dots + a_{N-1} x^{N-1} \\ &= (a_0 + a_2 x^2 + \dots + a_{N-2} x^{N-2}) + (a_1 x + a_3 x^3 + \dots + a_{N-1} x^{N-1}) \\ &= (a_0 + a_2 x^2 + \dots + a_{N-2} x^{N-2}) + x(a_1 + a_3 x^2 + \dots + a_{N-1} x^{N-2}) \\ &= A_e(x^2) + x A_o(x^2) \end{aligned}$$

故问题转化为计算如下数值：

$$A_e(\omega_N^{-2k}), A_o(\omega_N^{-2k}) \quad 0 \leq k < N$$

注意到：

$$\begin{aligned} \omega_N^{-2k} &= \omega_{\frac{N}{2}}^{-k} \\ \omega_{\frac{N}{2}}^{-(k+\frac{N}{2})} &= \omega_{\frac{N}{2}}^{-k} \\ \omega_N^{-(k+\frac{N}{2})} &= -\omega_N^{-k} \end{aligned}$$

所以我们实际上只需要计算如下数值：

$$\begin{aligned} \mathbf{A}_e &= [A_e(\omega_{\frac{N}{2}}^0), A_e(\omega_{\frac{N}{2}}^{-1}), A_e(\omega_{\frac{N}{2}}^{-2}), \dots, A_e(\omega_{\frac{N}{2}}^{-(\frac{N}{2}-1)})] \\ \mathbf{A}_o &= [A_o(\omega_{\frac{N}{2}}^0), A_o(\omega_{\frac{N}{2}}^{-1}), A_o(\omega_{\frac{N}{2}}^{-2}), \dots, A_o(\omega_{\frac{N}{2}}^{-(\frac{N}{2}-1)})] \end{aligned}$$

便可以计算出所需要的数值：

$$\begin{aligned} \mathbf{F}[k] &= \mathbf{A}_e[k] + \omega_N^{-k} \mathbf{A}_o[k] \\ \mathbf{F}\left[k + \frac{N}{2}\right] &= \mathbf{A}_e[k] - \omega_N^{-k} \mathbf{A}_o[k] \\ 0 &\leq k < \frac{N}{2} \end{aligned}$$

在此我们只证明第二个算式：

$$\begin{aligned}
 \mathbf{F} \left[k + \frac{N}{2} \right] &= F(\omega_N^{-(k+\frac{N}{2})}) \\
 &= A_e(\omega_N^{-2(k+\frac{N}{2})}) + \omega_N^{-(k+\frac{N}{2})} A_o(\omega_N^{-2(k+\frac{N}{2})}) \\
 &= A_e(\omega_{\frac{N}{2}}^{-k+\frac{N}{2}}) - \omega_N^{-k} A_o(\omega_{\frac{N}{2}}^{-k+\frac{N}{2}}) \\
 &= A_e(\omega_{\frac{N}{2}}^{-k}) - \omega_N^{-k} A_o(\omega_{\frac{N}{2}}^{-k}) \\
 &= \mathbf{A}_e[k] - \omega_N^{-k} \mathbf{A}_o[k]
 \end{aligned}$$

而 $\mathbf{A}_e, \mathbf{A}_o$ 的计算又可以递归地使用上述方法，并且问题的规模在指数级地缩减，故我们可以利用FFT算法高效地实现离散傅里叶变换地计算。

0.3 算法竞赛中的数论与组合数学