Deep Learning

Come from dive into deep learning

note For reading

我真的不懂忧郁

Delft University of Technology

TU Delft

Deep Learning

Come from dive into deep learning note For reading

by

我真的不懂忧郁

Student Name	Student Number
First Surname	1234567

Instructor:	I. Surname
Teaching Assistant:	I. Surname
Project Duration:	Month, Year - Month, Year
Faculty:	Faculty of Aerospace Engineering, Delft

Style: TU Delft Report Style, with modifications by Daan Zwaneveld

TU Delft

Preface

A preface...

我真的不懂忧郁

Delft, August 2024

Summary

A summary...

目录

Nomenclature

If a nomenclature is required, a simple template can be found below for convenience. Feel free to use, adapt or completely remove.

Abbreviations

Abbreviation	Definition
ISA	International Standard Atmosphere
...	

Symbols

Symbol	Definition	Unit
V	Velocity	[m/s]
...		
ρ	Density	[kg/m^3]
...		

Chapter 1

Linear Nerual Network

1.1. Practice 1: 线性回归

> **Question 1:** 假设有一些数据 $x_1, \cdots, x_n \in \mathbb{R}$。找使得 $\sum_i (x_i - b)^2$ 最小化的解析解，这个问题以及其解和正态分布有什么关系？

令 $\mathcal{L}(b) = \sum_i (x_i - b)^2$，则

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial b} &= -\sum_i 2(x_i - b) = 0 \\ \Rightarrow b &= \frac{x_1 + \cdots, x_n}{n} \end{aligned} \tag{1.1}$$

即令解析解最小化的 b 刚好是数据集 $x_1, \cdots, x_n$ 的均值。

> **Question 2:** 推导使用平方误差的线性回归优化问题的解析解。
>
> 1. 用向量表示法写出优化问题；
> 2. 计算损失对 ω 的梯度；
> 3. 通过将梯度设为 0、求解矩阵方程来找到解析解；
> 4. 什么时候可能比使用**随机梯度下降**更好？这种方法何时会失效？

假设数据维度为 d，共 n 组，因此数据矩阵为 $X \in \mathbb{R}^{n \times d}$

$$X = \begin{bmatrix} x_{11} & x_{21} & \cdots & x_{nd} \\ x_{12} & x_{22} & \cdots & x_{nd} \\ \vdots & \vdots & \ddots & \vdots \\ x_{1d} & x_{2d} & \cdots & x_{nd} \end{bmatrix} \tag{1.2}$$

预测值 $y = [y_1, y_2, \cdots, y_n], y_i \in \mathbb{R}, i \in 1, 2, \cdots, n$，标签值 $\hat{y} = [\hat{y}_1, \hat{y}_2, \cdots, \hat{y}_n], \hat{y}_i \in \mathbb{R}, i \in 1, 2, \cdots, n$

$$y = X^T \omega \tag{1.3}$$

其中 $\omega = (\omega^1, \omega_2, \cdots, \omega_d)^T$。优化问题写成

$$\mathcal{L}(\omega) = \sum_{i=1}^{n} (\omega^T x_i - \hat{y}_i)^2 = (X^T\omega - \hat{y})^T (X^T\omega - \hat{y}) \tag{1.4}$$

优化问题为

$$\omega = \arg\min_{\omega} \mathcal{L}(\omega) \tag{1.5}$$

计算损失函数的梯度：首先展开损失函数

$$\begin{aligned} \mathcal{L}(\omega) &= (\omega^T X - \hat{y}^T)(X^T\omega - \hat{y}) \\ &= \omega^T X X^T \omega - \omega^T X\hat{y} - \hat{y}^T X^T \omega + \hat{y}^T\hat{y} \end{aligned} \tag{1.6}$$

求梯度，当梯度为 0 是为驻点

$$\nabla_\omega \mathcal{L} = 0 \tag{1.7}$$

注意到 XX^T 是对称矩阵，所以 $XX^T = (XX^T)^T = (X^T)^T X^T = XX^T$

$$\nabla_\omega \mathcal{L} = 2XX^T\omega - 2X\hat{y} = 0 \tag{1.8}$$

所以解析解

$$\omega = (XX^T)^{-1}(X\hat{y}) \tag{1.9}$$

Question 3: 假定控制附加噪声 ε 的噪声模型是指数分布：$p(\varepsilon) = \frac{1}{2} exp(-|\varepsilon|)$，

1. 写出模型 $-log\ P(y|X)$ 下数据的负对数似然函数；

2. 试着写解析解；

3. 提出一种 *SGD* 算法来解决这个问题，那里可能出错？（当我们不断更新参数时，在驻点处会发生什么？）

1. ε 服从指数分布，带有噪声的线性回归问题为

$$y^{(i)} = \omega \cdot x^{(i)} + \varepsilon \tag{1.10}$$

其中 $\omega^{(i)} = (\omega_1^{(i)}, \omega_2^{(i)}, \cdots, \omega_n^{(i)}, b^{(i)})$，$x = (x_1^{(i)}, x_2^{(i)}, \cdots, x_n^{(i)}, 1)$，$y^{(i)} \in \mathbb{R}$，我们可以观测到给定 x 的 y 的条件概率

$$p(y^{(i)}|x^{(i)}) = \frac{1}{2} \exp\left(-|y^{(i)} - \omega \cdot x^{(i)}|\right) \tag{1.11}$$

所以构造似然函数

$$P(y|X) = \prod_{i=1}^{n} p(y^{(i)}|x^{(i)}) = \prod_{i=1}^{n} \frac{1}{2} \exp\left(-|y^{(i)} - \omega \cdot x^{(i)}|\right) \tag{1.12}$$

其中 $y^{(i)} = (y^{(1)}, y^{(2)}, \cdots, y^{(n)})$，$X = (x^{(1)}, x^{(2)}, \cdots, x^{(n)})$ 取对数不改变似然函数的单调性

$$\mathcal{L}(\omega) = -\log\ P(y|X) = -\sum_{i=1}^{n} \log\ \frac{1}{2} \exp\left(-|y^{(i)} - \omega \cdot x^{(i)}|\right) \tag{1.13}$$

2. 解析解即求 $\mathcal{L}(\omega)$ 负对数似然函数的最小值。展开 $\mathcal{L}(\omega)$ 中的其中一项

$$-\log \frac{1}{2}\exp\left(-|y^{(i)}-\omega\cdot x^{(i)}|\right)=\log\frac{1}{2}+|\,y^{(i)}-\omega\cdot x^{(i)}\,| \tag{1.14}$$

要上式子最小，只要绝对值项是最小，也就是说我需要去优化的函数变成了

$$\mathcal{L}'(\omega)=\sum_{i=1}^{n}|\,y^{(i)}-\omega\cdot x^{(i)}\,| \tag{1.15}$$

3. 由于该损失函数在具有突变，所以没有办法求导。所以需要重新定义 $\partial\mathcal{L}/\partial\omega$。

$$\frac{\partial\mathcal{L}}{\partial\omega}=\begin{cases}-x^{(i)} & y^{(i)}-\omega\cdot x^{(i)}>0\\ x^{(i)} & y^{(i)}-\omega\cdot x^{(i)}<0\\ 0 & y^{(i)}=\omega\cdot x^{(i)}\end{cases} \tag{1.16}$$

梯度下降

$$\omega\leftarrow\omega+\eta\cdot\frac{\partial\mathcal{L}}{\partial\omega} \tag{1.17}$$

记录：在高斯噪声的假设下，最小化均方误差等价于对线性模型的极大似然估计，对于指数分布噪声，则是考虑绝对值误差。

最小均方误差 (L_2 损失函数) 的鲁棒性比较差，如果损失函数在驻点处的比较平缓，则梯度下降会收到比较大的干扰，绝对误差损失函数 (L_1) 鲁棒性比较好，但是由于在 $x=0$ 出是突变的，无法求导。

1.2. Practice 2: 线性回归从零实现

Question 4: 如果我们将权重初始化为零，会发生什么？

Question 5: 假设试图为电压和电流关系建立一个模型，自动微分可以用来学习模型的参数吗？

Question 6: 能基于普朗克定律使用广谱能量密度来确定物体的温度么？

Question 7: 计算二阶导数时可能会遇到什么问题？

Question 8: 为什么在 *squared_loss* 函数中需要使用 *reshape* 函数

Question 9: 尝试使用不同的学习率，观察损失函数值下降的快慢

Question 10: 如果样本个数不能被批量整除，*data_iter* 函数的行为会有什么变化

References

[1] I. Surname, I. Surname, and I. Surname. “The Title of the Article”. In: *The Title of the Journal* 1.2 (2000), pp. 123–456.

Chapter A

Source Code Example

Adding source code to your report/thesis is supported with the package `listings`. *An example can be found below. Files can be added using* `\lstinputlisting[language=<language>]{<filename>}`.

```
"""
ISA Calculator: import the function, specify the height and it will return a
list in the following format: [Temperature,Density,Pressure,Speed of Sound].
Note that there is no check to see if the maximum altitude is reached.
"""

import math
g0 = 9.80665
R = 287.0
layer1 = [0, 288.15, 101325.0]
alt = [0,11000,20000,32000,47000,51000,71000,86000]
a = [-.0065,0,.0010,.0028,0,-.0028,-.0020]

def atmosphere(h):
    for i in range(0,len(alt)-1):
        if h >= alt[i]:
            layer0 = layer1[:]
            layer1[0] = min(h,alt[i+1])
            if a[i] != 0:
                layer1[1] = layer0[1] + a[i]*(layer1[0]-layer0[0])
                layer1[2] = layer0[2] * (layer1[1]/layer0[1])**(-g0/(a[i]*R))
            else:
                layer1[2] = layer0[2]*math.exp((-g0/(R*layer1[1]))*(layer1[0]-layer0[0]))
    return [layer1[1],layer1[2]/(R*layer1[1]),layer1[2],math.sqrt(1.4*R*layer1[1])]
```

Chapter B

Task Division Example

If a task division is required, a simple template can be found below for convenience. Feel free to use, adapt or completely remove.

表 B.1: Distribution of the workload

	Task	Student Name(s)
	Summary	
Chapter 1	Introduction	
Chapter 2		
Chapter 3		
Chapter *		
Chapter *	Conclusion	
	Editors	
	CAD and Figures	
	Document Design and Layout	

Chapter C

Derivative of Vector

C.1. 一元泰勒展开

我们知道函数的一阶导数表示函数在一点的斜率，这意味着函数在这一点的斜率行为可以用一条切线逼近

$$f(x) = f(x_0) + f'(x_0)(x - x_0) \tag{C.1}$$

这可以看作是一个一元多项式，因此能够想到如果想更多描述函数在某点处的行为（比如描述函数斜率的变化率还需要知道二阶导数）可以用多项式去逼近，这就是泰勒开展

$$f(x) = f(x_0) + f^{(1)}(x_0)(x - x_0) + \frac{f^{(2)}}{2!}(x - x_0)^2 + \cdots + \frac{f^{(n)}}{n!}(x - x_0)^n + o((x - x_0)^{n+1}) \tag{C.2}$$

C.2. 二元泰勒展开

定义点 (a_1, a_2) 在周边邻域的近似，泰勒定理需要研究的是在 (a_1, a_2) 周围邻域上的函数近似，设 $(x_1 = a_1 + tu, x_2 = a_2 + tv)$。构造辅助函数

$$\Phi(t) = f(a_1 + tu, a_2 + tv)(0 \leqslant t \leqslant 1) \tag{C.3}$$

并把 Φ 在 $t = 0$ 处展开去近似 $t = 1$ 的值

$$\begin{aligned} \Phi(t) &= \Phi(0) + \frac{\Phi'(0)}{1!}t + \frac{\Phi''(0)}{2!}t^2 + \cdots + \frac{\Phi^{(n)}(0)}{n!}t^n + \frac{\Phi^{(n+1)}(0)}{(n+1)!}t^{(n+1)} \\ \Phi(1) &= \Phi(0) + \frac{\Phi'(0)}{1!} + \frac{\Phi''(0)}{2!} + \cdots + \frac{\Phi^{(n)}(0)}{n!} + \frac{\Phi^{(n+1)}(0)}{(n+1)!} \end{aligned} \tag{C.4}$$

所以根据链式法则

$$\begin{aligned} \frac{df}{dt} &= \frac{\partial f}{\partial x_1}\frac{\partial x_1}{\partial t} + \frac{\partial f}{\partial x_1}\frac{\partial x_2}{\partial t} \\ \Phi'(0) &= u\frac{\partial f}{\partial x_1}(a_1, a_2) + v\frac{\partial f}{\partial x_2}(a_1, a_2) \\ \Phi''(0) &= \cdots \end{aligned} \tag{C.5}$$

令 $t=1$，则 $u=x_1-a_1, v=x_2-a_2$，因此得到二元函数的泰勒公式

$$\begin{aligned}\Phi(1)=f(x_1,x_2)=&f(a_1,a_2)+(x_1-a_1)\frac{\partial f}{\partial x_1}(a_1,a_2)+(x_2-a_2)\frac{\partial f}{\partial x_2}(a_1,a_2)\\&+\frac{1}{2!}[(x_1-a_1)^2\frac{\partial^2 f}{\partial x_1^2}(a_1,a_2)+(x_1-a_1)(x_2-a_2)\frac{\partial^2 f}{\partial x_1\partial x_2}(a_1,a_2)\\&+(x_1-a_1)(x_2-a_2)\frac{\partial^2 f}{\partial x_2\partial x_1}(a_1,a_2)+(x_2-a_2)^2\frac{\partial^2 f}{\partial x_2^2}(a_1,a_2)]+\cdots\end{aligned}\tag{C.6}$$

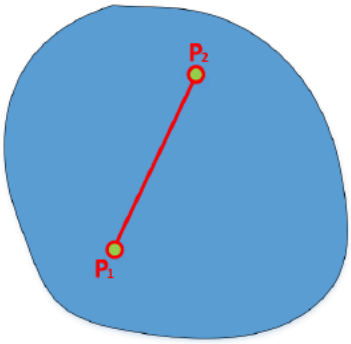
写成矩阵的形式，令 $x=[x_1,x_2], a=[a_1,a_2]$

$$f(x)=f(a)+\nabla f(a)\cdot(x-a)+\frac{1}{2}(x-a)^T H(a)(x-a)\tag{C.7}$$

其中 $H(a)$ 为二阶 *Hessian* 矩阵。

二元函数的辅助函数

函数 $f(x_1,x_2)$ 在开区域 R 中有二阶连续偏导，其中 $P_1(a_1,a_2)$ 是该区域的一个点，我们在开区域中任选另一个点 $P_2(a_1+u,a_2+v)$，并且我们设 u 和 v 足够小，来保证从 P_1 沿直线运动到 P_2 的路径仍然在开区域中：



则描述从 P_1 到 P_2 的运动轨迹的参数方程为 $(a_1+t\cdot u, a_2+t\cdot v)$，

因此定义参数方程 $F(t)=f(a_1+t\cdot u,a_2+t\cdot v)$，我们知道当 u 和 v 变化时，P_2 可以表示开区域附近邻域上的任意一个点。因为 $f(a_1+t\cdot u,a_2+t\cdot v)$ 中，$x_1=a_1+t\cdot u$，$x_2=a_2+t\cdot v$，因此其实对 t 求导就可以应用链式法则。

同理，因为 t 在 $[0,1]$ 之间是连续的，所以我们可以对 F 在 $t=0$ 进行泰勒展开（其实就是在 (a_1,a_2) 点展开），令 $t=1$ 就能得到：

$$F(1)=F(0)+\frac{F'(0)}{1!}+\frac{F''(0)}{2!}+\ldots\tag{四.15}$$

换句话说，$f(a_1+u,a_2+v)$ 就等于 $F(1)$，也就是说可以用上面的公式来进行近似。因此，只要当 u 和 v 任意取值时，我们就能得到 R 上 $P_1(a_1,a_2)$ 点附近函数值所有的近似值了！

现在，我们重新令变量 $x_1=u+a_1$，$x_2=v+a_2$，这样代入到上式，我们就能得到 $f(x_1,x_2)$ 在 (a_1,a_2) 处的泰勒展开。

图 C.1: 二元函数的辅助函数

C.3. 小结

函数在一点 a 展开，关注的是以 a 点附近邻域的函数的行为，肯定要满足在一点处的展开的值等于 a 处的函数值，所以泰勒级数中常数项等于 $f(a)$，后面 n 阶导项等于 $(x-a)^n$ 次方，当 $x=a$ 时满足 $f(x)=f(a)$。然后一阶导数项刚好用一条直线逼近。