贝叶斯线性回归

Bayes Linear Regression

learning note For reading translation

我真的不懂忧郁

Delft University of Technology

TU Delft

贝叶斯线性回归

Bayes Linear Regression
learning note For reading translation

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

TU Delft

Preface

A preface...

我真的不懂忧郁

Delft, June 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

Bayes Linear Regression Background

贝叶斯线性回归和基于最小二乘法的线性回归本质上都是线性回归，只不过是从不同的角度去看。贝叶斯方法在线性回归中的主要任务是

下面的推导中，我们会展示如何从概率分布来描述线性回归。

1.1. 从概率密度函数的角度认识最小二乘估计

关于线性回归问题求解模型参数 $\mathcal{W}$ 时，采用的方法是最小二乘估计

$$\mathcal{L}(\omega) = \sum_{i=1}^{N} \|\mathcal{W}^T x^{(i)} - y^{(i)}\|^2 \tag{1.1}$$

并通过最小二乘估计，求解模型参数 ω 的矩阵形式表达

$$\mathcal{W} = (\mathcal{X}^T\mathcal{X})^{-1}\mathcal{X}^T\mathcal{Y} \tag{1.2}$$

矩阵表达有以下弊端

1. $\mathcal{X}^T\mathcal{X}$ 是一个 $p \times p$ 的对称矩阵，它至少时半正定矩阵，但一定不是正定矩阵，从而导致 $(\mathcal{X}^T\mathcal{X})^{-1}$ 可能是不可求的。
2. 由于 $\mathcal{X}$ 是样本集合，如果 X 的样本量很大，会导致 $\mathcal{X}^T\mathcal{X}$ 的计算成本很大；

从概率密度函数的角度观察，**最小二乘估计的本质是极大似然估计**：给定样本 $x^{(i)}$ 和对应标签 $y^{(i)}$ 之间的关联关系，可以得到 $P(y^{(i)}|x)(i)$ 的概率分布，也就是说实际上就是 $y^{(i)}$ 和 $\mathcal{W}^T x^{(i)}$ 的距离服从一个随机分布，建立数学模型如下

$$y^{(i)} = \mathcal{W}^T x^{(i)} + \epsilon \tag{1.3}$$

其中 $\epsilon \sim \mathcal{N}(\mu, \sigma^2)$，通过这样构造我们可以估计

$$P(y^{(i)}|x^{(i)}, \mathcal{W}) \sim \mathcal{N}(\mathcal{W}^T x^{(i)} + \mu, \sigma^2) \tag{1.4}$$

构建极大似然函数

$$\begin{aligned}\mathcal{L}(\mathcal{W}) &= \log\prod_{i=1}^{N} P(y^{(i)}|x^{(i)},\mathcal{W}) \\ &= \sum_{i=1}^{N}\log\left[\frac{1}{\sigma\sqrt{2\pi}}exp\left(\frac{[\,y^{(i)}-(\mathcal{W}^T x^{(i)}+\mu)\,]^2}{2\sigma^2}\right)\right]\end{aligned} \tag{1.5}$$

所以转成了一个优化问题

$$\mathcal{W} = \arg\max_{\mathcal{W}}\ \mathcal{L}(\mathcal{W}) \tag{1.6}$$

$$\mathcal{W} \propto \arg\max_{\mathcal{W}}\ \sum_{i=1}^{N}\left[y^{(i)}-(\mathcal{W}^T x^{(i)}+\mu)\right]^2 \tag{1.7}$$

注意到如果 $\mu=0$ 那就是最小二乘法的形式。

1.2. 从概率密度函数的角度认识过拟合和正则化

针对最小二乘估计的过拟合问题，引入正则化，常见的正则化有以下方式

1. Lasso 回归 ($\mathcal{L}_1$ 正则化);
2. 岭回归 ($\mathcal{L}_2$ 正则化)

从概率密度函数的角度考虑**基于正则化的最小二乘估计，可以将其视作关于 $\mathcal{W}$ 的最大后验估计**

$$\mathcal{W}_{MAP} = arg\max_{\mathcal{W}}\ \frac{P(\mathcal{Y}|\mathcal{W})\cdot P(\mathcal{W})}{P(\mathcal{Y})} \propto arg\max_{\mathcal{W}}\ P(\mathcal{Y}|\mathcal{W})\cdot P(\mathcal{W}) \tag{1.8}$$

由于样本之间独立同分布，因而有

$$\mathcal{W}_{MAP} \propto arg\max_{\mathcal{W}}\ \left[\log\prod_{i=1}^{N} P(y^{(i)}|\mathcal{W})\cdot P(\mathcal{W})\right] \tag{1.9}$$

令先验分布 $P(\mathcal{W})\sim\mathcal{N}(\mu_0,\sigma_0^2)$，将 $P(\mathcal{Y}|\mathcal{W})\sim\mathcal{N}(\mathcal{W}^T\mathcal{X},\sigma^2)$ 一同代入上式，有

$$\hat{\mathcal{W}}_{MAP} = arg\ \max_{\mathcal{W}}\ \sum_{i=1}^{N}\left[(y^{(i)}-\mathcal{W}^T x^{(i)})^2+\frac{\sigma^2}{\sigma_0^2}(\mathcal{W}-\mu_0)^2\right] \tag{1.10}$$

令 $\lambda=\frac{\sigma^2}{\sigma_0^2}$，$\mu_0=0$，上式转化为

$$\hat{\mathcal{W}}_{MAP} = arg\ \max_{\mathcal{W}}\ \sum_{i=1}^{N}\left[(y^{(i)}-\mathcal{W}^T x^{(i)})^2+\lambda\left\|\mathcal{W}\right\|_2^2\right] \tag{1.11}$$

这刚好是 $\mathcal{L}_2$ 正则化的线性回归，如果想要 Lasso 正则化，则用拉普拉斯分布替换高斯分布。

1.3. 小结

前面的推导主要说明了从概率分布角度的线性回归和一般的线性回归是一个问题，并且引出了我们的贝叶斯线性回归的模型

$$\begin{cases} f(x^{(i)}) = \mathcal{W}x^{(i)} \\ y^{(i)} = f(x^{(i)})+\epsilon \end{cases} \tag{1.12}$$

其中 $\epsilon\sim\mathcal{N}(\mu,\sigma^2)$，

1. 推断 (*Inference*)：求解后验概率 $P(\mathcal{W}|Data)$
2. 预测 (*Predict*)：基于后验概率对未知样本的情况进行预测求 $P(\hat{\mathcal{X}}|Data)$

接下来两个小章节主要是看如何在这个模型上作推断和预测。

Chapter 2

Inference 问题

推断问题就是求模型的参数 $\mathcal{W}$

2.1. Gaussian Distribution Proporty

theorem 2.1.1 (高斯分布的共轭先验):

proof.

2.2. 模型建立

首先我们需要对贝叶斯公式进行分解

$$P(\omega|\mathcal{X},\mathcal{Y}) = \frac{P(\omega,\mathcal{Y}|\mathcal{X})}{P(\mathcal{Y}|\mathcal{X})} = \frac{P(\omega|\mathcal{Y},\mathcal{X})P(\omega)}{\int_\omega P(\mathcal{Y}|\mathcal{X},\omega)P(\omega)d\omega} \tag{2.1}$$

其中 $P(\mathcal{Y}|\mathcal{X},\omega)$ 是似然函数，$P(\omega)$ 是先验函数，似然函数的求解过程为

$$P(\mathcal{Y}|\mathcal{X},\omega) = \prod_{i=1}^{N} P(y_1|x_i,\omega) \tag{2.2}$$

又因为 $\mathcal{Y} = \omega^T\mathcal{X} + \epsilon$，$\epsilon \sim \mathcal{N}(0,\sigma^2)$(这里假设期望为 0)，所以

$$P(y_i|x_i,\omega) \sim \mathcal{N}(\omega^T x_i, \sigma^2) \tag{2.3}$$

所以

$$P(\mathcal{Y}|\mathcal{X},\omega) = \prod_{i=1}^{N} P(y_i|x_i,\omega) = \prod_{i=1}^{N} \mathcal{N}(\omega^T x_i, \sigma^2) \tag{2.4}$$

我们假设 $P(\omega) \sim \mathcal{N}(0,\Sigma_P)$，又因为 $P(Y|X)$ 与参数 ω 无关，所以这是一个定值，所以我们将公式改写为

$$P(\omega|X,Y) \propto P(Y|\omega,X)P(\omega) \tag{2.5}$$

根据高斯分布的共轭性质：似然函数和先验函数都是高斯分布，所以后验也一定是高斯分布。

$$P(\omega|Data) \sim \mathcal{N}(\mu_\omega, \Sigma_\omega) \propto \prod_{i=1}^{N} \mathcal{N}(\omega^T x_i, \sigma^2)\mathcal{N}(0, \Sigma_p) \tag{2.6}$$

我们的目的就是求解 μ_ω 和 Σ_ω。

2.3. 模型求解

似然函数的矩阵形式写成

$$\begin{aligned} P(Y|X,\omega) &= \frac{1}{(2\pi)^{N/2}\sigma^N} exp\left\{-\frac{1}{2\sigma^2}\sum_{i=1}^{N}(Y - W^T X^T)(Y - XW)\right\} \\ &= \frac{1}{(2\pi)^{N/2}\sigma^N} exp\left\{-\frac{1}{2}\sum_{i=1}^{N}(Y - W^T X^T)\sigma^{-2}(Y - XW)\right\} \end{aligned} \tag{2.7}$$

其中 $W = [\omega, \omega, \cdots, \omega]^T$，所以我们有

$$P(Y|X,\omega) \sim \mathcal{N}(WX, \sigma^2 I) \tag{2.8}$$

代入模型

$$P(\omega|Data) \sim \mathcal{N}(\mu_\omega, \Sigma_\omega) \propto \mathcal{N}(WX, \sigma^2 I)\mathcal{N}(0, \Sigma_p) \tag{2.9}$$

$$\begin{aligned} \mathcal{N}(WX, \sigma^2 I)\mathcal{N}(0, \Sigma_p) \propto exp&\left\{-\frac{1}{2}\sum_{i=1}^{N}(Y - W^T X^T)\sigma^{-2}(Y - XW) - \frac{1}{2}\omega^T \Sigma_p^{-1}\omega\right\} \\ exp&\left\{-\frac{1}{2\sigma^2}(Y^T Y - 2Y^T XW + W^T X^T XW) - \frac{1}{2}W^T \Sigma_p^{-1} W\right\} \end{aligned} \tag{2.10}$$

采用待定系数法类比来确定参数即可。对于一个分布 $p(x) \sim \mathcal{N}(\mu, \Sigma)$，它的指数部分为

$$exp\left\{-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right\} = exp\left\{-\frac{1}{2}(x^T \Sigma^{-1} x - 2\mu^T \Sigma^{-1} x + \Delta)\right\} \tag{2.11}$$

其中 Δ 是常数部分。类比发现

$$x^T \Sigma^{-1} x = W^T \sigma^{-2} X^T XW + W^T \Sigma_p^{-1} W \tag{2.12}$$

可以得到

$$\Sigma_\omega^{-1} = \sigma^{-2} X^T X + \Sigma_p^{-1} \tag{2.13}$$

令 $A = \Sigma_\omega^{-1}$，我们希望可以从一次项中得到 μ_ω，将一次项提取出来进行观察可以得到

$$\begin{aligned} &\mu^T A = \sigma^{-2} Y^T X \\ &\Rightarrow (\mu^T A)^T = (\sigma^{-2} Y^T X)^T \\ &\Rightarrow A^T \mu = \sigma^{-2} X^T Y \end{aligned} \tag{2.14}$$

因为 Σ_ω 是协方差矩阵，所以一定是对称的，所以 $A^T = A$，所以

$$\mu_\omega = \sigma^{-2} A^{-1} X^T Y \tag{2.15}$$

2.4. 小结

利用贝叶斯推断的方法来确定参数之间的分布，也就是确定 $P(W|X,Y)$

Chapter 3

Predict 问题

经过 inference 后，我们已经成功导出了 $P(\omega|Data)$ 的分布

$$P(W|X,Y) \sim \mathcal{N}(\mu_\omega, \Sigma_\omega) \tag{3.1}$$

下面一步就是根据这个参数分布去预测给定 x^* 对应的 y^*。

3.1. Predict

模型预测第一步

$$f(x^*) = x^{*T}\omega \tag{3.2}$$

而在 inference 的时候，我们得到了 $P(W|X,Y) \sim \mathcal{N}(\mu_\omega, \Sigma_\omega)$，所以

$$f(x^*) = x^{*T}\omega \sim \mathcal{N}(x^{*T}\mu_\omega, x^{*T}\Sigma_\omega x^*) \tag{3.3}$$

在根据 $\epsilon \sim \mathcal{N}(0, \sigma^2)$，所以

$$P(y^*|Data, x^*) \sim \mathcal{N}(x^{*T}\mu_\omega, x^{*T}\Sigma_\omega x^* + \sigma^2) \tag{3.4}$$

这里我们就给出了预测分布。

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	