Subject: Stanford CS229 Machine Learning, Lecture 3, Weighted Least Squares, Logistic regression, Newton's Method

Date: from November 3, 2024 to November 22, 2024

Contents

CS229 Machine Learning, Weighted Least Squares, Logistic regression, Newton's Method, 2022, Lecture 3

YouTube: Stanford CS229 Machine Learning, Weighted Least Squares, Logistic regression, Newton's Method, 2022, Lecture 3

Introduction

Recall of Linear Regression

given: 给定训练数据集(training set) [a]

$$D = \{(\boldsymbol{x}^{(1)}, y^{(1)}), \cdots, (\boldsymbol{x}^{(n)}, y^{(n)})\}, \quad \text{其中} \boldsymbol{x}^{(i)} \in \mathbb{R}^{d+1}, y^{(i)} \in \mathbb{R}$$

do: 找到$\boldsymbol{\theta} \in \mathbb{R}^{d+1}$ 使得$\boldsymbol{\theta} = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{n} \left(y^{(i)} - h_{\boldsymbol{\theta}}(x^{(i)})\right)^2$, 其中$h(\boldsymbol{x}) = \boldsymbol{\theta}^T \boldsymbol{x}$

[a]特征$\boldsymbol{x}^{(i)} \in \mathbb{R}^{d+1}$是因为包含了常数项1，这只是一个为了方便的记号

Probabilistic View of Linear Regression

Noise / Error

使用概率的视角去看待线性回归是因为这样我们可以很自然地将其应用于更丰富的模型类别。

现在考虑一个有噪声的线性回归:

$$y^{(i)} = \boldsymbol{\theta}_*^T \boldsymbol{x}^{(i)} + \epsilon^{(i)} \tag{1}$$

其中对于每一个i来说，$\epsilon^{(i)}$是一个随机噪声(error/noise).[a]

由于噪声ϵ是随机的，那么对于所有的噪声$\{\epsilon^{(1)}, \epsilon^{(2)}, \cdots, \epsilon^{(n)}\}$，我们可以关注其统计性质，一般来说其统计性质的设定有:

1. $\mathbb{E}\left[\epsilon^{(i)}\right] = 0$，即无偏性[b]
2. $\mathbb{E}\left[\epsilon^{(i)}\epsilon^{(j)}\right] = \mathbb{E}\left[\epsilon^{(i)}\right]\mathbb{E}\left[\epsilon^{(j)}\right]$ for $i \neq j$，即误差是相互独立的(一个error不会提供另一个不同的error的任何信息)
3. $\mathbb{E}\left[(\epsilon^{(i)})^2\right] = \sigma^2$，即噪声的方差$= \mathbb{E}\left[(\epsilon^{(i)})^2\right] - \mathbb{E}\left[(\epsilon^{(i)})\right]^2 = \sigma^2$，这是描述噪声的另一个统计量

虽然真实情况可能并非如此，但是基本上我们会假设$\epsilon \sim \mathcal{N}(0, \sigma^2)$，关于高斯分布可以见附录A.

Note 1. 模型*(1)*实际上是所谓的前向模型*(Forward Model)*，就是说即使不知道$\boldsymbol{\theta}$是什么，但是知道标签y是通过何种方式生成的。或者说这是一种生成模型*(Generative Model)*，即给了特征$\boldsymbol{x}^{(i)}$，也给了相应的噪声$\epsilon^{(i)}$的构造方式，那么就可以表示$y^{(i)}$.

[a]模型可以解释的部分就是$\boldsymbol{\theta}$，但模型一般不能完全做到$y^{(i)} = \boldsymbol{\theta}^T \boldsymbol{x}^{(i)}$，这部分差异即不能解释的因素就可以归入噪声中；在物理中相似地可以理解为每一次进行测量时带来的测量误差(measurement error).

[b]无偏性说明了即使噪声在某些地方对y的预测产生了影响，但是总体上而言是并无影响的，即总体上"没有偏差"；此外如果噪声期望不为0，实际上可以对参数$\boldsymbol{\theta}$进行简单改变就能得到无偏.

Probabilistic View of Linear Regression – Why Least Square Method

由式(1)可知$\epsilon^{(i)} = y^{(i)} - \boldsymbol{\theta}_*^T \boldsymbol{x}^{(i)}$，因此若$\epsilon \sim \mathcal{N}(0, \sigma^2)$，那么$y - \boldsymbol{\theta}_*^T \boldsymbol{x} \sim \mathcal{N}(0, \sigma^2)$，$y \sim \mathcal{N}(\boldsymbol{\theta}_*^T \boldsymbol{x}, \sigma^2)$，因此有：

$$P(y^{(i)}|\boldsymbol{x}^{(i)}; \boldsymbol{\theta}_*) = \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{(y - \boldsymbol{\theta}_*^T)^2}{2\sigma^2}) \tag{2}$$

这也就是说，确定模型的参数$\boldsymbol{\theta}$就意味着确定相应分布，即特征$\boldsymbol{x}^{(i)}$对应的标签$y^{(i)}$满足概率分布(2)，那么实际上模型的Loss就可以理解为在此概率分布下的期望$\mathbb{E}[y^{(i)} \neq \boldsymbol{\theta}\boldsymbol{x}]$。

Likelihood[a]: 似然性是指在已知特征$\boldsymbol{x}^{(i)}$时$y^{(i)}$取值的可能性，我们当然是要选取"最可能"的$y^{(i)}$作为预测标签(最大化对应的似然)。我们需要估计的是模型的似然，即参数$\boldsymbol{\theta}$的似然:

$$\mathcal{L}(\boldsymbol{\theta}) = P(\boldsymbol{y}|\boldsymbol{x}; \boldsymbol{\theta}) \overset{i.i.d.}{=} \prod_{i=1}^{n} P(y^{(i)}|\boldsymbol{x^{(i)}}; \boldsymbol{\theta}) = \prod_{i=1}^{n} \frac{1}{\sigma\sqrt{2\pi}} \exp(-\frac{(y^{(i)} - \boldsymbol{\theta}_*^T \boldsymbol{x}^{(i)})^2}{2\sigma^2}) \tag{3}$$

可以看到当前式(3)中的结果是将所有样本的似然性相乘，这是因为我们假设样本是独立同分布的(i.i.d.)。但是为方便进行梯度下降，我们更倾向于使用一种累和而非累加的形式，而取对数就可以不改变单调性地将累乘转化为累加:

$$l(\boldsymbol{\theta}) = \log \mathcal{L}(\boldsymbol{\theta}) = \sum_{i=1}^{n} \log P(y^{(i)}|\boldsymbol{x^{(i)}}; \boldsymbol{\theta}) = \sum_{i=1}^{n} \log \frac{1}{\sigma\sqrt{2\pi}} - \sum_{i=1}^{n} \frac{(y^{(i)} - \boldsymbol{\theta}_*^T \boldsymbol{x}^{(i)})^2}{2\sigma^2} \tag{4}$$

我们的目的是使得$\boldsymbol{x}^{(i)}$对应的标签$y^{(i)}$的概率最大，即最大化似然$\mathcal{L}(\boldsymbol{\theta})$，而对数不改变单调性，因此最终目的就是：

$$\max_{\boldsymbol{\theta}} l(\boldsymbol{\theta}) = \min_{\boldsymbol{\theta}} \sum_{i=1}^{n} \frac{(y^{(i)} - \boldsymbol{\theta}_*^T \boldsymbol{x}^{(i)})^2}{2} \triangleq J(\boldsymbol{\theta}) \tag{5}$$

显然，式(5)就是我们熟知的最小二乘法。

对$J(\boldsymbol{\theta})$求导可得：

$$\frac{\partial}{\partial \boldsymbol{\theta}} J(\boldsymbol{\theta}) = \sum_{i=1}^{n} \left(h_\theta(x^{(i)}) - y^{(i)}\right) x^{(i)} \tag{6}$$

[a] 可见https://en.wikipedia.org/wiki/Likelihood_function

Classification

Introduction

分类问题也是机器学习中的一大类重要问题，且在生活中非常常见。一个最简单的分类问题是二分类：

given: 给定训练数据集(training set)

$$D = \{(\boldsymbol{x}^{(1)}, y^{(1)}), \cdots, (\boldsymbol{x}^{(n)}, y^{(n)})\}, \quad \text{其中} \boldsymbol{x}^{(i)} \in \mathbb{R}^{d+1}, y^{(i)} \in \{-1, 1\}$$

do: 找到合适的方式将$y = 1$的正类(positive class)与$y = -1$的负类(negative class)尽可能区分开[a]。

当数据非常简单时，例如只有一维、数据量不大，并且类别相同的数据很好地聚在一

起时，我们可以使用简单的线性回归就将数据很好的分开，如图1左图所示。但是当同类数据不是很集中时，就很难使用简单的线性回归进行分类，如图1右图所示，直线会逐渐变平导致系数逐渐趋于零使的分类器几乎与特征无关（不能提取特征）。

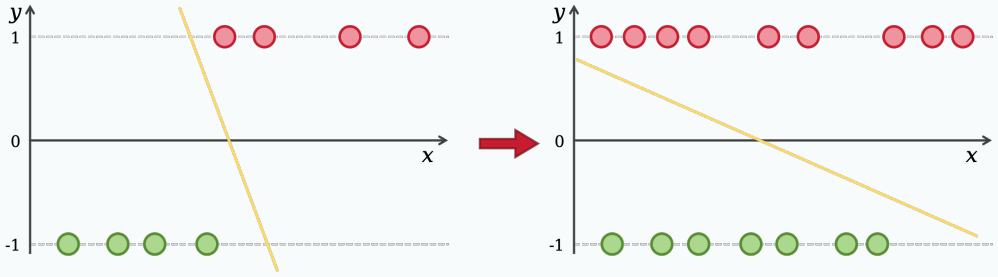

Figure 1: example of classification

[a]正类与负类只是名称，并无特定含义，同时取任何能代表类别不同的指示如{0,1}都可以

Logistic Regression–Introduction

可以看到我们使用直线或超平面并不足以对数据点进行很好的分类，如果我们的分类器是一个更复杂的曲线(如图2使用所谓的logistic回归对$\{0,1\}$标签进行回归)，那么会对提升分类成功率也许会有帮助。

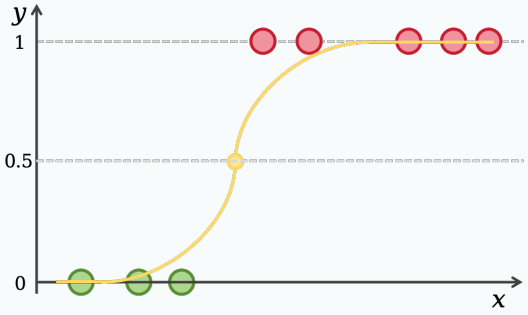

Figure 2: logistic regression

让我们的线性模型变成曲线/曲面的自然做法是添加非线性，即对模型$h(\boldsymbol{x}) = \boldsymbol{\theta}^T\boldsymbol{x}$中的$\boldsymbol{\theta}^T\boldsymbol{x}$添加非线性作用$g(\cdot)$，这种非线性作用被称为链接函数(link function)[a]。一个常用的link function是sigmoid函数：

$$g(x) = \frac{1}{1+e^{-x}} \tag{7}$$

显然有$0 < g(x) < 1$，因此自然可以作为对$\{0,1\}$标签的分类器，并且由于此函数非常"光滑"，因此计算导数时性质较好，也便于进行梯度下降。

这样就得到了logistic regression模型：

$$h(\boldsymbol{x}) = g(\boldsymbol{\theta}^T\boldsymbol{x}) = (1 + exp(-\boldsymbol{\theta}^T\boldsymbol{x}))^{-1} \tag{8}$$

[a]link function的作用就是增加非线性，在深度学习中，实际上就是激活函数(activation function)的作用

Logistic Regression–Loss Function

与式(3)中一样，我们同样以概率和统计的视角考虑logistic regression的极大似然。首先令：

$$P(y=1|x;\theta)=h_\theta(x),\quad P(y=0|x;\theta)=1-h_\theta(x) \tag{9}$$

此时将式(10)合二为一即为：

$$P(y|x;\theta)=h_\theta(x)^y\times(1-h_\theta(x))^{1-y},\quad y\in\{0,1\} \tag{10}$$

因此可以将似然函数定义如下：

$$\mathcal{L}(\boldsymbol{\theta})=P(\boldsymbol{y}|\boldsymbol{x};\boldsymbol{\theta})\overset{i.i.d.}{=}\prod_{i=1}^{n}P(y^{(i)}|\boldsymbol{x^{(i)}};\boldsymbol{\theta})=\prod_{i=1}^{n}h_\theta(x^{(i)})^{y^{(i)}}(1-h_\theta(x^{(i)}))^{1-y^{(i)}} \tag{11}$$

分析式(11)的定义形式可以看出，如果标签$y^{(i)}=1$，那么只剩下$h_\theta(x^{(i)})^{y^{(i)}}$，此时我们期望的当然是$h_\theta(x^{(i)})^{y^{(i)}}$更加接近1，即使得$\mathcal{L}(\boldsymbol{\theta})$更大；如果标签$y^{(i)}=0$，那么只剩下$1-h_\theta(x^{(i)})^{y^{(i)}}$，此时我们期望的是$h_\theta(x^{(i)})^{y^{(i)}}$更加接近0，同样使得$\mathcal{L}(\boldsymbol{\theta})$更大。

与式(4)类似，对式(11)取对数，得到：

$$l(\boldsymbol{\theta})=\log\mathcal{L}(\boldsymbol{\theta})=\sum_{i=1}^{n}\log P(y^{(i)}|\boldsymbol{x^{(i)}};\boldsymbol{\theta})=\sum_{i=1}^{n}\Big(y^{(i)}\log h_\theta(x^{(i)})+(1-y^{(i)})(1-\log h_\theta(x^{(i)}))\Big) \tag{12}$$

因此我们的损失函数定义为：

$$J(\boldsymbol{\theta})\triangleq-\sum_{i=1}^{n}\Big(y^{(i)}\log h_\theta(x^{(i)})+(1-y^{(i)})\log(1-h_\theta(x^{(i)}))\Big) \tag{13}$$

此时再进行梯度下降：

$$\boldsymbol{\theta}^{i+1}=\boldsymbol{\theta}^{i}-\alpha\frac{\partial}{\partial\boldsymbol{\theta}}J(\boldsymbol{\theta}) \tag{14}$$

其中

$$\frac{\partial}{\partial\boldsymbol{\theta}}J(\boldsymbol{\theta})=\sum_{i=1}^{n}\Big(h_\theta(x^{(i)})-y^{(i)}\Big)\,x^{(i)} \tag{15}$$

巧合的是，这里损失函数的导数竟然与线性回归中的式(6)一模一样，事实上这也四一个普遍的规律，将会在后续课程中介绍。此处式(15)的详细推导详见附录B。

Newton's Methods

Newton's Methods

给定函数$f:\mathbb{R}^d\to\mathbb{R}$，我们想要找到$f$的零点，即$f(\boldsymbol{\theta})=0$,这实际上对应于我们在寻找最值过程中的目标$l'(\boldsymbol{\theta})=0$。我们的迭代过程如图3所示：

1. 初始点为猜测点（随机/一定方式框定的）记为$\boldsymbol{\theta}^{(0)}$
2. 沿该点切线方向寻找其与x轴交点，记为$\boldsymbol{\theta}^{(1)}$
3. 依照2的模式迭代直至满足停机准则

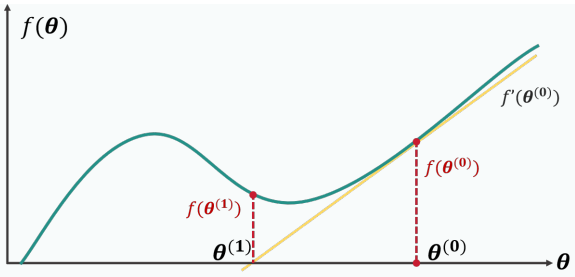

Figure 3: Newton method

这种迭代格式的数学形式就是：

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \Delta$$

其中根据三角形$\triangle_{\boldsymbol{f(\theta^{(t)})\theta^{(t)}\theta^{(t+1)}}}$的结构就可以知道

$$\Delta = f(\boldsymbol{\theta}^{(t)})/f^{'}(\boldsymbol{\theta}^{(t)})$$

自然地，当函数值是高维情形$\boldsymbol{\theta} \in \mathbb{R}^{d+1}$时：

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - H^{-1} \cdot \nabla_{\boldsymbol{\theta}} l(\boldsymbol{\theta})$$

其中$H \in \mathbb{R}^{(d+1)\times(d+1)}$是Hessian矩阵，$H_{ij} = \frac{\partial^2}{\partial\theta_i \partial\theta_j} l(\boldsymbol{\theta})$。

最后值得一提的是，牛顿法在收敛速度方面一般来说是非常快的，但是可以看到计算Hessian阵所需要的计算耗时和存储都非常巨大。

Comparison of different optimization algorithms

Comparison of different optimization algorithms			
Methods	**Per iteration**	**Compute**[1]	**Steps to error** ϵ
SGD	1 data point	$\Theta(d)$	ϵ^{-2}
Batch GD	N data points	$\Theta(nd)$	$\approx \epsilon^{-1}$
Netwon Method	N data points	$\Omega(nd^2)$	$\approx \log(\epsilon^{-1})$

[1] d为数据维度，n为训练数据量（并不等价于数据集大小，因为可能没训练完）
[2] $\Theta(x)$表示渐近紧确界，即在x的常数范围$(C_1 x, C_2 x)$内变动；$\Omega(x)$表示渐近下界，表示至少是x

Table 1: Representative Algorithms by Category

A Gaussian Distribution

Gaussian Distribution

Gauss分布(Gaussian Distribution)是最常用的分布，其概率密度函数为(如图4):

$$P(x;\mu,\sigma^2)=\frac{1}{\sigma\sqrt{2\pi}}\exp(-\frac{(x-\mu)^2}{2\sigma^2})$$

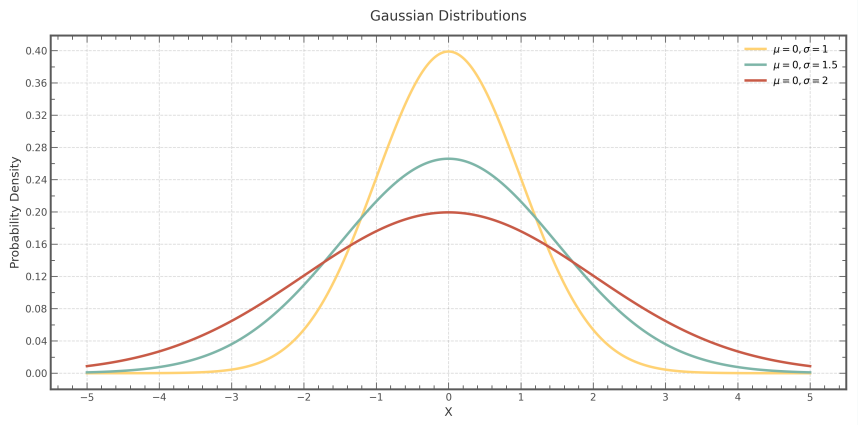

Figure 4: Gaussian distribution

B Gradient of loss function

Gaussian Distribution

$$J(\boldsymbol{\theta})\triangleq-\sum_{i=1}^{n}\left(y^{(i)}\log h_\theta(x^{(i)})+(1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right)$$

$$\frac{\partial}{\partial\boldsymbol{\theta}}J(\boldsymbol{\theta})=\sum_{i=1}^{n}\left(h_\theta(x^{(i)})-y^{(i)}\right)x^{(i)}$$

Proof. 由于$h(x)=\frac{1}{1+\exp(-x)}$，因此

$$h^{'}(x)=\frac{\exp(-x)}{(1+\exp(-x))^2}=h(x)(1-h(x))$$

从而对于$h_\theta(x^{(i)})=\frac{1}{1+\exp(-\boldsymbol{\theta}^\top x^{(i)})}$，其导数为

$$\frac{\partial h_\theta(x^{(i)})}{\partial\boldsymbol{\theta}}=h_\theta(x^{(i)})\left(1-h_\theta(x^{(i)})\right)x^{(i)}.$$

将$J(\boldsymbol{\theta})$ 关于$\boldsymbol{\theta}$ 的梯度表示为:

$$\frac{\partial J(\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}=-\sum_{i=1}^{n}\frac{\partial}{\partial\boldsymbol{\theta}}\left[y^{(i)}\log h_\theta(x^{(i)})+(1-y^{(i)})\log(1-h_\theta(x^{(i)}))\right].$$

对于第一项$y^{(i)} \log h_\theta(x^{(i)})$:

$$\frac{\partial}{\partial \boldsymbol{\theta}} \left[y^{(i)} \log h_\theta(x^{(i)}) \right] = y^{(i)} \frac{1}{h_\theta(x^{(i)})} \cdot \frac{\partial h_\theta(x^{(i)})}{\partial \boldsymbol{\theta}}.$$

对于第二项$(1 - y^{(i)}) \log(1 - h_\theta(x^{(i)}))$:

$$\frac{\partial}{\partial \boldsymbol{\theta}} \left[(1 - y^{(i)}) \log(1 - h_\theta(x^{(i)})) \right] = (1 - y^{(i)}) \frac{-1}{1 - h_\theta(x^{(i)})} \cdot \frac{\partial h_\theta(x^{(i)})}{\partial \boldsymbol{\theta}}.$$

将两部分合并:

$$\frac{\partial J(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = -\sum_{i=1}^{n} \left[y^{(i)} \frac{1}{h_\theta(x^{(i)})} - (1 - y^{(i)}) \frac{1}{1 - h_\theta(x^{(i)})} \right] \cdot \frac{\partial h_\theta(x^{(i)})}{\partial \boldsymbol{\theta}}.$$

注意到:

$$\frac{1}{h_\theta(x^{(i)})} - \frac{1}{1 - h_\theta(x^{(i)})} = \frac{h_\theta(x^{(i)}) - y^{(i)}}{h_\theta(x^{(i)})(1 - h_\theta(x^{(i)}))}.$$

再结合$\frac{\partial h_\theta(x^{(i)})}{\partial \boldsymbol{\theta}} = h_\theta(x^{(i)})(1 - h_\theta(x^{(i)}))x^{(i)}$, 最终梯度为:

$$\frac{\partial}{\partial \boldsymbol{\theta}} J(\boldsymbol{\theta}) = \sum_{i=1}^{n} \left(h_\theta(x^{(i)}) - y^{(i)} \right) x^{(i)}$$

□

C Codes

C.1 Python codes for plotting Gaussian distribution

```python
import numpy as np
import matplotlib.pyplot as plt

# generate data from gaussian distribution
x = np.linspace(-5, 5, 500)
mu_sigma_pairs = [
    (0, 1, colors["huang"]),
    (0, 1.5, colors["lv1"]),
    (0, 2, colors["hong"])
]

# create a figure
fig, ax = plt.subplots(figsize=(12, 6), dpi=400)

# plot several lines
for mu, sigma, color in mu_sigma_pairs:
    y = (1 / (np.sqrt(2*np.pi) * sigma)) * np.exp(-0.5 * ((x - mu) / sigma)**2)
    ax.plot(x, y, color=color, lw=2.5, label=f"$\mu={mu}, \sigma={sigma}$")

set_figure()  # a function for figure plot setting
plt.savefig("gaussian_distributions.png", dpi = 400)  # save the figure
plt.show()
```

First updated: November 22, 2024
Last updated: May 31, 2025, modified equation(8), thanks to Yifu Zheng!

References