模型选择：偏差、方差权衡与正则化

Liam Huang*

2017 年 3 月 30 日

*liamhuang0205@gmail.com

1 偏差和误差

1.1 模型选择的一般做法

- 选定一个机器学习算法;
- 根据拟合情况，调整算法的超参数;
- 以某种指标，选择误差最小的超参数组合。

核心问题：误差。

- 最终指标 $\leftarrow$ 误差
- 拟合情况 $\leftarrow$ 误差

1.2 误差的组成

机器学习任务中，误差可以认为由 3 部分组成

- 随机误差（Random Error）;
- 偏差（Error due to Bias）;
- 方差（Error due to Variance）。

随机误差 来源于训练数据本身的噪音，不可避免；服从高斯分布，记作 $\epsilon \sim \mathcal{N}(0, \sigma_\epsilon)$。

偏差 模型预测值与真实规律之间的差距，记作 $\text{Bias}(X) = E[\hat{f}(X)] - f(X)$。

方差 描述模型自身的不稳定性，记作 $\text{Var}(X) = E\left[\left(\hat{f}(X) - E[\hat{f}(X)]\right)^2\right]$。

提示. 这里的方差对应的随机变量，可以认为是相同的算法在不同训练集上得到的模型最后在验证集上的预测值。

考虑均方误差，

$$\begin{aligned}
\text{Err}(X) &= E\left[\left(y - \hat{f}(X)\right)^2\right] \\
&= E\left[\left(f(X) + \epsilon - \hat{f}(X)\right)^2\right] \\
&= \left(E[\hat{f}(X)] - f(X)\right)^2 + E\left[\left(\hat{f}(X) - E[\hat{f}(X)]\right)^2\right] + \sigma_\epsilon^2 \\
&= \text{Bias}^2 + \text{Variance} + \text{Random Error}.
\end{aligned} \tag{1}$$

1.3 图示

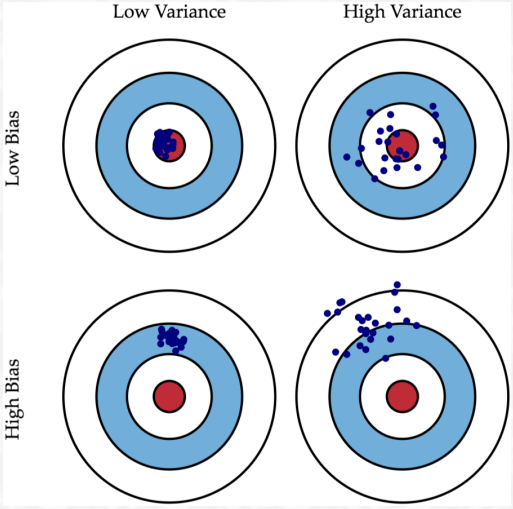

图 1：靶心图

左上 理想状况，在无穷的数据支持下，完美的算法可以达到。

右上 偏差小而方差大。

左下 偏差大而方差小。

右下 偏差大且方差大。

2 举个例子

我们假设有这样一个目标任务：训练模型，拟合一组接近线性相关的数据。

首先，我们用 Python 生成两组数据，分别作为训练集和验证集。

代码片段 1：生成数据集

```python
import numpy as np

np.random.seed(42) # the answer to life, the universe and everything
real = lambda x:x + x ** 0.1

x_train = np.linspace(0, 15, 100)
y_train = map(real, x_train)
y_noise = 2 * np.random.normal(size = x_train.size)
y_train = y_train + y_noise

x_valid = np.linspace(0, 15, 50)
y_valid = map(real, x_valid)
y_noise = 2 * np.random.normal(size = x_valid.size)
y_valid = y_valid + y_noise
```

现在，我们选用最小平方误差作为损失函数，尝试用多项式函数去拟合这些数据，得到两个模型 `prop` 和 `overf`。

代码片段 2：训练模型

```
prop   = np.polyfit(x_train, y_train, 1)
prop_  = np.poly1d(prop)
overf  = np.polyfit(x_train, y_train, 15)
overf_ = np.poly1d(overf)
```

而后，我们可以把模型的效果绘制出来。

代码片段 3：绘制拟合效果图像

```
import matplotlib.pyplot as plt

_ = plt.figure(figsize = (14, 6))
plt.subplot(1, 2, 1)
prop_e = np.mean((y_train - np.polyval(prop, x_train)) ** 2)
overf_e = np.mean((y_train - np.polyval(overf, x_train)) ** 2)
xp     = np.linspace(-2, 17, 200)
plt.plot(x_train, y_train, '.')
plt.plot(xp, prop_(xp), '-', label = 'proper, err: %.3f' % (prop_e))
plt.plot(xp, overf_(xp), '--', label = 'overfit, err: %.3f' % (overf_e))
plt.ylim(-5, 20)
plt.legend()
plt.title('train set')
plt.subplot(1, 2, 2)
# ...
plt.title('validation set')
```

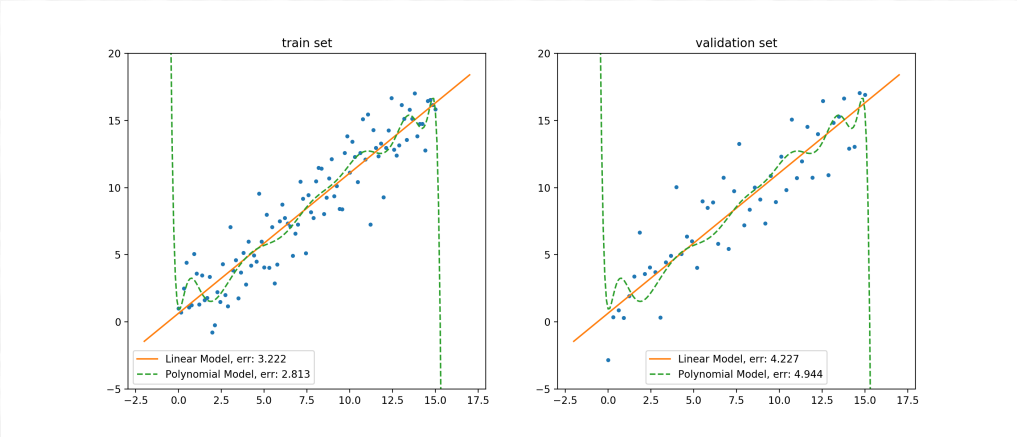

图 2: 拟合效果

在训练集上,

- 多项式模型的误差低于线性模型的误差;
- 线性模型在训练集上欠拟合;
- 线性模型在训练集上的偏差高于多项式模型。

在验证集上,

- 多项式模型的误差高于线性模型的误差;
- 多项式模型的误差在两个集合上变化明显;
- 多项式模型在训练集上过拟合;
- 多项式模型的方差高于线性模型;
- 线性模型的泛化能力更好。

3 权衡之术

3.1 克服强迫症

训练误差越低越好?

- 训练样本中的随机噪声不可避免;
- 训练样本的抽样不均匀;
- 模型本身学习能力有上限。

3.2 最佳平衡点的数学表述

通常的做法是：固定数据集，改变模型复杂度，寻找最优模型。

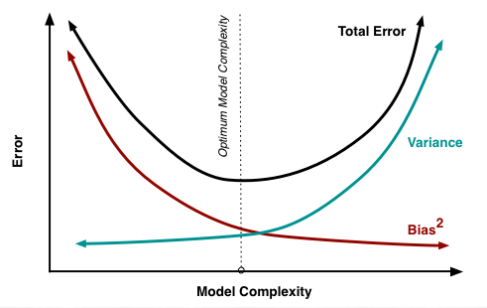

图 3: 偏差和方差的变化趋势

- 模型复杂度增加 $\rightarrow$ 描述能力增加 $\rightarrow$ 偏差减小、方差增大;
- 模型复杂度降低 $\rightarrow$ 描述能力降低 $\rightarrow$ 偏差增大、方差减小。

最佳平衡点是误差曲线的拐点，一阶导数有：

$$\frac{\mathrm{dBias}}{\mathrm{dComplexity}} = -\frac{\mathrm{dVariance}}{\mathrm{dComplexity}} \tag{2}$$

3.3 外在表现

上述两个导数在宏观上难以计算，因此，需要有宏观的外在表现，辅助判断模型的拟合状态。在有限训练集上，增加模型复杂度，误差会一直下降。

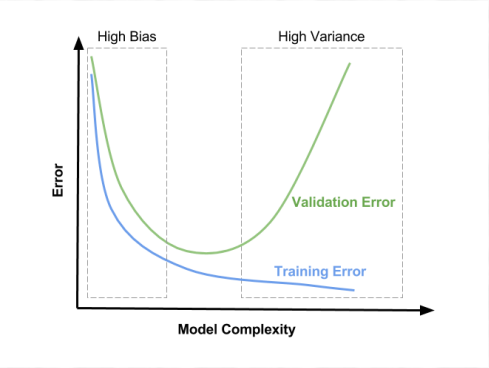

图 4：误差变化曲线

- 欠拟合：训练集和验证集上的误差都很大；
- 过拟合：训练集的误差小，而验证集的误差非常大。

3.4 处理过拟合与欠拟合

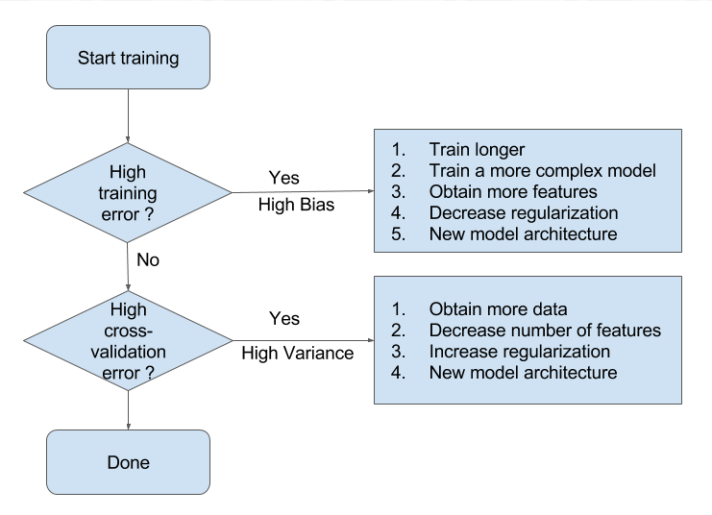

图 5：模型调优的工作流

当模型处于欠拟合状态时，根本的办法是增加模型复杂度。

- 增加模型的迭代次数；
- 更换描述能力更强的模型；
- 生成更多特征供训练使用；
- 降低正则化水平。

当模型处于过拟合状态时，根本的办法是降低模型复杂度。

- 扩增训练集；
- 减少训练使用的特征的数量；
- 提高正则化水平。

4 正则化

4.1 损失函数与目标函数

机器学习的目标，实际上就是找到一个足够好的函数 F^* 用以预测。

因此，我们要定义「好」

$$l(y, \hat{y}) = l\big(y, F(\vec{x})\big).$$

于是，在全局上有

$$F^* = \arg\min_F E_{y,\vec{x}}\Big[l\big(y, F(\vec{x})\big)\Big] = \arg\min_F L(F).$$

因此，机器学习的目标，就是一个最优化问题；而我们的目标就是使得全局损失函数 $L(F)$ 最小。

问题

- 只考虑了对数据的拟合；
- 没有考虑模型本身的复杂度；
- 过拟合怎么办？

引入正则项（regularizer）$\gamma\Omega(F)$，$\gamma > 0$，用来描述模型的复杂度。

修改最优化目标

$$F^* = \arg\min_F \mathrm{Obj}(F) = \arg\min_F L(F) + \gamma\Omega(F).$$

4.2 范数与正则项

范数是向量的某种抽象长度，它满足通常意义上的长度的三个基本性质：

- 非负性：$\|\vec{x}\| \geqslant 0$；
- 齐次性：$\|c \cdot \vec{x}\| = |c| \cdot \|\vec{x}\|$；
- 三角不等式：$\|\vec{x} + \vec{y}\| \leqslant \|\vec{x}\| + \|\vec{y}\|$。

常用的范数：

- L_0-范数：$\|\vec{x}\|_0 = \#(i)$，with $i \neq 0$；
- L_1-范数：$\|\vec{x}\|_1 = \sum_{i=1}^{d} |x_i|$；
- L_2-范数：$\|\vec{x}\|_2 = \left(\sum_{i=1}^{d} x_i^2\right)^{1/2}$；
- L_p-范数：$\|\vec{x}\|_p = \left(\sum_{i=1}^{d} x_i^p\right)^{1/p}$；
- L_∞-范数：$\|\vec{x}\|_\infty = \lim_{p \to +\infty} \left(\sum_{i=1}^{d} x_i^p\right)^{1/p}$。

范数的非负性，使得它有可能作为正则项使用。

若使用 $\|\vec{w}\|_p$ 作为正则项，则我们说使用了 L_p-正则项（the L_p-regularizer）。

4.3 L_0-正则项与 L_1-正则项（LASSO regularizer）

若引入 L_0-正则项，令

$$\Omega\big(F(\vec{x};\vec{w})\big) \stackrel{\text{def}}{=} \ell_0 \frac{\|\vec{w}\|_0}{n},\ \ell_0 > 0,$$

这意味着，我们希望绝大多数 $\vec{w}$ 的分量为零，从而实现了

- 稀疏化；
- 特征选择。

L_0-正则项的问题在于

- 非连续；
- 非凸；
- 不可求导。

因此，L_0-正则项虽好，但是难以求解（NP-Hard）。

因此我们考虑 L_1-正则项，令

$$\Omega\big(F(\vec{x};\vec{w})\big) \stackrel{\text{def}}{=} \ell_1 \frac{\|\vec{w}\|_1}{n},\ \ell_1 > 0,$$

这意味着，我们希望 $\vec{w}$ 中各分量绝对值的和尽可能小。

事实上，L_1-正则项是 L_0-正则项最紧的凸放松（tightest convex relaxation, Emmanuel Candes）。

参数更新的变化

考虑目标函数

$$\mathrm{Obj}(F) = L(F) + \gamma \cdot \ell_1 \frac{\|\vec{w}\|_1}{n},$$

有对参数 w_i 的偏导数

$$\frac{\partial \mathrm{Obj}}{\partial w_i} = \frac{\partial L}{\partial w_i} + \frac{\gamma \ell_1}{n} \mathrm{sgn}(w_i).$$

因此有参数更新

$$w_i \to w_i' \stackrel{\text{def}}{=} w_i - \eta \frac{\partial L}{\partial w_i} - \eta \frac{\gamma \ell_1}{n} \mathrm{sgn}(w_i).$$

多出的项 $\eta \frac{\gamma \ell_1}{n} \mathrm{sgn}(w_i)$ 使得 $w_i \to 0$，实现「稀疏化」。

4.4 L_2-正则项（Ridge Regularizer）

若引入 L_2-正则项，则令

$$\Omega\Big(F(\vec{x};\vec{w})\Big) \stackrel{\text{def}}{=} \ell_2 \frac{\|\vec{w}\|_2^2}{2n},\ \ell_2 > 0,$$

因此有目标函数

$$\text{Obj}(F) = L(F) + \gamma \cdot \ell_2 \frac{\|\vec{w}\|_2^2}{2n},$$

对参数 w_i 的偏导数

$$\frac{\partial \text{Obj}}{\partial w_i} = \frac{\partial L}{\partial w_i} + \frac{\gamma \ell_2}{n} w_i.$$

再有参数更新

$$\begin{aligned} w_i \to w_i' &\stackrel{\text{def}}{=} w_i - \eta \frac{\partial L}{\partial w_i} - \eta \frac{\gamma \ell_2}{n} w_i \\ &= \left(1 - \eta \frac{\gamma \ell_2}{n}\right) w_i - \eta \frac{\partial L}{\partial w_i}. \end{aligned}$$

考虑到 $\eta \frac{\gamma \ell_2}{n} > 0$，因此，引入 L_2-正则项之后，相当于衰减了（decay）参数的权重，使参数趋近于零。

4.5 为什么能避免过拟合

对于 L_1-正则项，

- 只更新主要参数的值；
- 避免噪音干扰。

对于 L_2-正则项，考虑多项式拟合任务，

- 为了拟合噪声，函数图像会「扭曲」；
- 扭曲的本质是在小范围内突变；
- 也就是参数过大。

L_2-正则项使参数的各分量尽可能稠密地贴近 0，避免了某个分量特别大的情况。

4.6 L_1-正则项与 L_2-正则项的区别

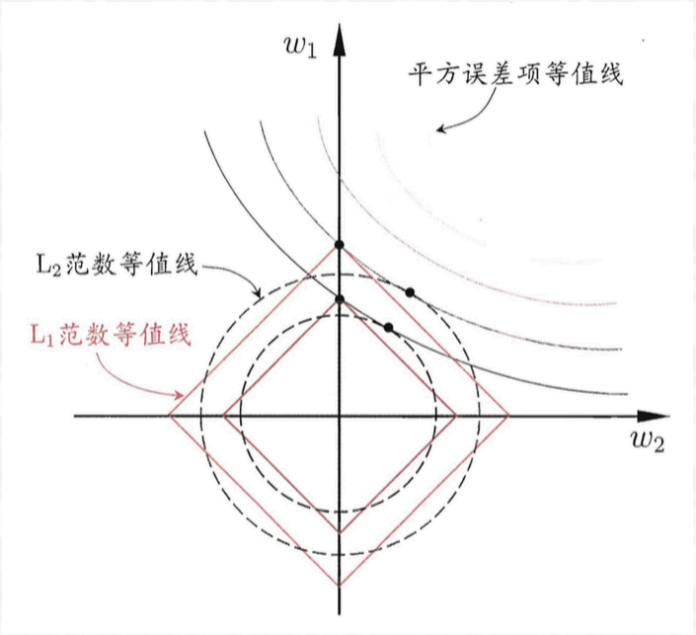

图 6: L_1-正则项更容易得到稀疏解

图 6 以两个参数分量为例, 绘制了三组等值线。

- 外层的等值线数值大;
- 在正则项和损失函数之间权衡。

L_1-正则项的等值线与损失函数的等值线, 容易相交在坐标轴上——有一项参数为 0, 产生稀疏性。

UGA