基于多项式的曲线拟合方法与分析

赵文浩
厦门大学计算机科学系
23020201153860@xmu.edu.cn

摘　要：多项式拟合可以看作是机器学习的中线性回归的一个案例。本文通过基于多项式的曲线拟合实验，并对模型复杂度、拟合误差损失、过拟合现象、模型正则化进行分析。该案例涉及到了机器学习的五大要素，即损失函数、求解算法、模型选择、大数据、模型应用，可有效加深对 ML＝Lambda 的理解。通过分析回归模型的选择、过拟合现象和正则化操作，进一步讨论了过拟合与模型复杂度之间的关系以及机器学习中正则化的必要性，同时提出了几点防止模型过拟合的有效策略。

关键词：机器学习；线性回归；多项式拟合；过拟合；正则化

Curve Fitting Method and Analysis Based on Polynomial

Zhao Wenhao, Department of Computer Science, School of Information

Abstract: Polynomial fitting is an example of linear regression in machine learning. In this paper, the model complexity, fitting error loss, overfitting phenomenon and model regularization are analyzed through the polynomial curve fitting experiment. This case involves five elements of machine learning, namely Loss function, solving Algorithm, Model selection, Big Data, and the Application of model, which can effectively deepen the understanding of ML = LAMBDA. Through the analysis of regression model selection, overfitting and regularization, the relationship between overfitting and model complexity was further discussed, and the necessity of regularization in machine learning was also proposed. At the same time, several effective strategies to prevent model overfitting were proposed.

Key words: Machine Learning; Linear Regression; Polynomial Curve Fitting; Overfitting; Regularization

1. 引言

曲线拟合（Curve Fitting）是数据处理中广泛使用的数值方法，其本质是构造一个模型（函数），用连续且光滑的曲线或曲面近似地刻画给定的离散数据点对所表示的坐标之间的函数关系，通过数值方法求解出对应模型的参数，从而得到离散点所表示的自变量与因变量之间的内在联系，同时能够基于求解得到的模型，对所考量的变量进行分析和预测，起到一定的决策作用。曲线拟合的应用场景丰富，如图形图像处理领域对各种线型的拟合操作，金融领域基于用户信用数据和贷款的决策问题，生物领域对药物浓度和服用时间的关系求解等。

2. 多项式模型与算法求解

2.1 泰勒公式

泰勒公式是将一个在 $x=x_0$ 处具有 n 阶导数的函数 $f(x)$ 利用关于 $(x-x_0)$ 的 n 次多项式来逼近函数的方法[1]。泰勒公式指出，若函数 $f(x)$ 在包含 x_0 的某个闭区间 $[a,b]$ 上具有 n 阶导数，且在开区间 (a,b) 上具有 $(n+1)$ 阶导数，则对闭区间 $[a,b]$ 上任意一点 x，成立下式：

$$f(x)=\frac{f(x_0)}{0!}+\frac{f'(x_0)}{1!}(x-x_0)+\frac{f''(x_0)}{2!}(x-x_0)^2+\cdots+\frac{f^{(n)}(x_0)}{n!}(x-x_0)^n+R_n(x)$$

其中 $f^{(n)}(x)$ 为 $f(x)$ 的 n 阶导数，等式右侧多项式称为函数 $f(x)$ 在 x_0 处的泰勒展开。$R_n(x)$ 是泰勒公式余项，是 $(x-x_0)^n$ 的高阶无穷小，有以下不同形式：

① Peano 余项

$R_n(x)=o\left[(x-x_0)^n\right]$，这里要求 $f(x)$ 的 n 阶导数存在。

② Lagrange 余项

$R_n(x)=f^{(n+1)}\left[x_0+\theta(x-x_0)\right]\frac{(x-x_0)^{n+1}}{(n+1)!}$，其中 $\theta\in(0,1)$。

2.2 多项式模型

2.1 节的泰勒公式指出：如果函数满足一定的条件，可以用函数在某一点的各阶导数值作为系数构建一个多项式来近似表达这个函数。因此，常选用多项式函数对给定的离散数据点进行拟合，多项式模型的基本形式如下：

$$y(x,w)=w_0+w_1x+w_2x^2+\cdots+w_Mx^M=\sum_{j=0}^{M}w_jx^j$$

因此，问题转化为如何根据已有的离散数据点求解上述多项式模型的参数 w。针对该问题，本文给出两种模型参数求解的方法，分别为基于 Lagrange 插值多项式的启发式算法和基于损失函数的机器学习算法。

2.3 多项式模型求解

2.3.1 Lagrange 插值多项式

Lagrange 插值算法可以给出一个恰好穿过二维平面上 $(n+1)$ 个已知点 $(x_0,x_1,\cdots,x_n)$ 的多项式函数 $P_n(x)$，满足插值条件且次数不超过 n 的多项式存在且唯一。Lagrange 插值多项式的形式如下：

$P_n(x)=\sum_{k=0}^{n}\left[f(x_k)\prod_{i=0\wedge i\neq k}^{n}\frac{x-x_i}{x_k-x_i}\right]$，其中 $f(x_k)$ 为样本点 x_k 对应的函数值[2]

Lagrange 插值多项式的特点是多项式的每一项在对应点的函数值为 1，在其它点的函数值均为 0，因此可看作为 $(n+1)$ 段 n 次多项式的组合。对于 $(n+1)$ 个点，采用这种方式构造出的 n 次多项式函数是唯一的，且该函数穿过每一个样本点。以 3 个样本点 (x_0,x_1,x_2) 且 $x_0\neq x_1\neq x_2$ 为例构造出的二次多项式如下：

$$P_2(x)=f(x_0)\cdot\frac{(x-x_1)(x-x_2)}{(x_0-x_1)(x_0-x_2)}+f(x_1)\cdot\frac{(x-x_0)(x-x_2)}{(x_1-x_0)(x_1-x_2)}+f(x_2)\cdot\frac{(x-x_0)(x-x_1)}{(x_2-x_0)(x_2-x_1)}$$

在实际应用中，采用 Lagrange 插值构造出的多项式具有严重的过拟合现象，因此可以选取部分样本点构造 $m(m<n)$ 次多项式，但选取哪些 $(m+1)$ 个样本点仍是需要考虑的问题。Lagrange 插值是一种启发式算法，基于此，需要提出一种面向整体样本的机器学习算法。

2.3.2 基于损失函数的参数求解算法

针对样本点 $(\langle x_1,y_1\rangle,\langle x_2,y_2\rangle,\cdots,\langle x_N,y_N\rangle)$，需要构造多项式函数 $y(x,w)$ 使得整体误差最小。这里以预测值与真实值之间的平方误差为例构造损失函数：

$$E(w)=\frac{1}{2}\sum_{k=1}^{N}\{y(x_k,w)-y_k\}^2$$

为了使整体误差最小，要尽可能的降低误差损失函数$E(w)$。这里可以采用多元函数求极值的方式求得使得误差最小时的模型参数为：

$$w^* = \left(X^T X\right)^{-1} X^T y$$

其中 $X = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^n \\ 1 & x_2 & x_2^2 & \cdots & x_2^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n+1} & x_{n+1}^2 & \cdots & x_{n+1}^n \end{bmatrix}$，$w = \begin{bmatrix} w_0 \\ w_1 \\ \vdots \\ w_n \end{bmatrix}$，$y = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{n+1} \end{bmatrix}$

2.3.3 基于损失函数和正则化的参数求解算法

从后面的实验结果来看，2.3.2 节提出的基于损失函数的参数求解算法在多项式模型阶数较大时，容易产生过拟合现象，且此时对应的模型参数具有较高量级，使得多项式曲线更加陡峭。因此需要对模型参数$w = [w_0, w_1, \cdots, w_n]$进行大小限制，基于此提出了正则化的概念，对应的损失函数$\tilde{E}(w)$如下：

$$\tilde{E}(w) = \frac{1}{2}\sum_{k=1}^{N}\{y(x_k, w) - y_k\}^2 + \frac{\lambda}{2}\|w\|^2$$

其中$\|w\|^2 = w^T w = w_0^2 + w_1^2 + + w_M^2$，$M$ 为多项式阶数，系数λ代表正则项与平方误差的相对重要程度[3]。此时求解得到的模型参数为$w^* = \left(X^T X + \lambda I\right)^{-1} X^T y$。

3. 实验结果与分析

3.1 数据集的产生

本文基于三角正弦函数$\sin(2\pi x)$生成原始数据，并在此基础上添加高斯噪声$N(\mu = 0.25, \sigma = 1.0)$，得到的训练数据（保留三位小数）如表 3-1 和图 3-1 所示：

表 3-1 待拟合训练数据

x	0.000	0.111	0.222	0.333	0.444	0.555	0.666	0.777	0.888	1.000
$\sin(2\pi x)$	0.000	0.643	0.985	0.866	3.420	-3.420	-0.866	-0.985	-0.643	-0.245
y	-0.041	0.504	1.658	0.703	0.209	-0.591	-1.439	-0.947	-0.435	0.364

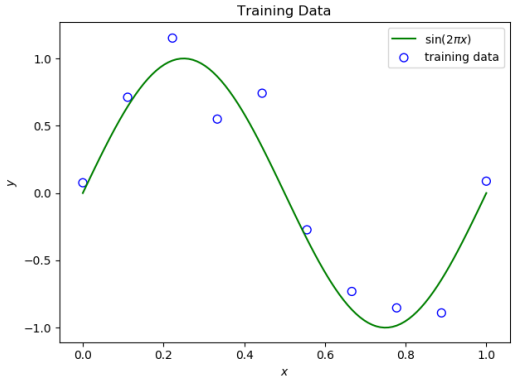


图 3-1 待拟合训练数据

3.2 拟合结果的定量分析

为了对拟合结果进行更加详细的衡量，本文引入均方根误差[3]指标定量分析拟合结果，均方误差根（RMSE）的计算方式如下：

$$E_{RMS} = \sqrt{2E\left(w^*\right)/N}$$

其中 w^* 为求解得到的模型参数，$E\left(w\right)$ 为 2.3.2 节提到的平方误差损失函数。

3.3 多项式阶数 M 的分析

本节主要讨论在给定样本点数量一定的条件下，多项式阶数 M 对拟合结果的影响。对于表 3-1 所示的待拟合数据，不同阶数对应的拟合结果如图 3-2 所示。

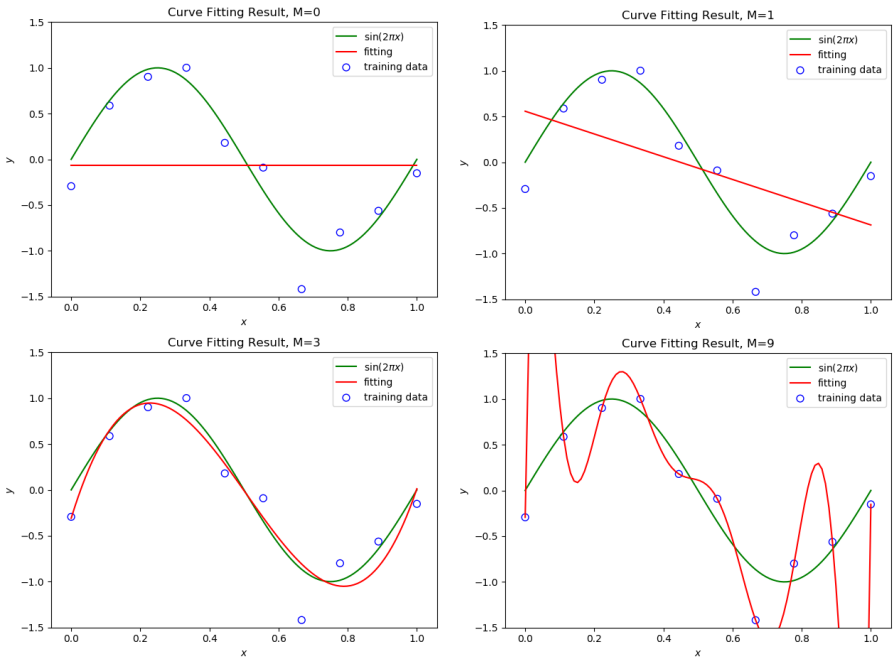


图 3-2 不同阶数下曲线拟合的结果

从拟合结果可以看出，随着多项式模型阶数的不断升高，模型复杂度增大，得到的拟合曲线由平缓到光滑再到陡峭，与待拟合训练数据的切合程度也越来越高。单从训练数据的角度来看，模型的阶数越高，拟合的效果越好。

需要注意的是，曲线拟合的最终目的是得到泛化能力较强的模型，不仅要在训练数据上有着出色的拟合效果，同时对于模型未接触到的新样本数据也应当具有较好的拟合结果，否则将出现过拟合（Overfitting）现象。这里同样采用 3.1 节数据集的生成方式，建立样本容量为 100 的测试数据，用于模型的泛化能力测试。我们选用 3.2 节引入的均方误差根（RMSE）评价指标在多项式模型不同阶数的条件下，对训练集和测试集上的误差进行分析，如图 3-3 所示。

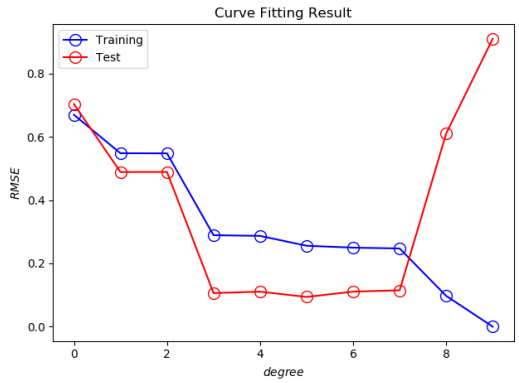


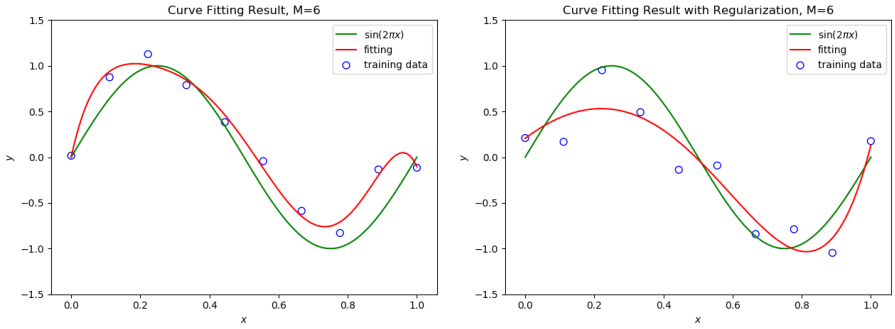
图 3-3 不同阶数下曲线拟合结果 RMSE 指标

从图 3-3 中可以看出，随着多项式模型阶数的不断升高，求解得到的模型在训练集上的误差逐步降低，但在测试集上的误差不断升高。这一现象说明高阶多项式模型很容易产生过拟合现象（Overfitting），无法应对新来的数据，因此需要采用 2.3.3 小节提出的正则化模型进行改进。

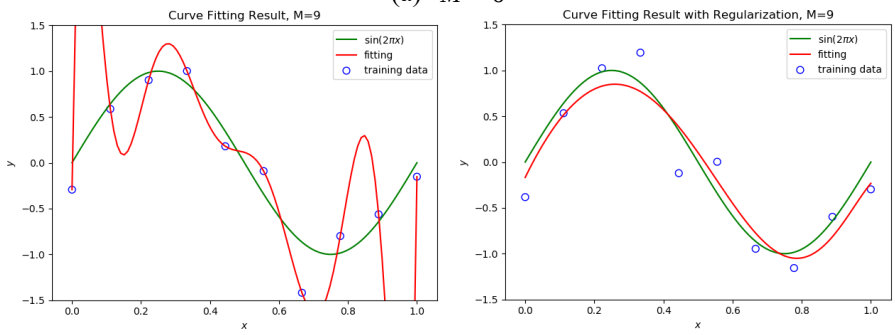
3.4 正则项系数的分析

从上一节中可以看到，随着多项式模型阶数的升高，过拟合现象越来越严重，最终导致求解得到的模型无法应对新数据。为了在一定程度上解决该问题，本文采用正则化的方式限制模型参数的量级，使得拟合曲线更加平滑。

本节主要讨论正则化特别是正则项系数对拟合结果的影响，这里分别以多项式模型阶数 $M=6$ 和 $M=9$ 为例，引入正则化操作前后的拟合结果如图 3-4 所示。



(a) $M=6$



(a) $M=9$

图 3-4 正则化前后拟合结果对比图

从图 3-4 中可以看出，引入正则化后，高阶多项式模型对应的拟合曲线由陡峭变得光滑，更符合数据的实际生成方式。为了进一步衡量正则化对过拟合现象的有效性，接下来仍然采用均方误差根对拟合结果进行定量分析。图 3-5 展示了引入正则化后，多项式模型的阶数对于拟合误差 RMSE 指标的影响；图 3-6 展示了多项式阶数 $M=9$ 的条件下，正则项系数与拟合误差 RMSE 的关系。

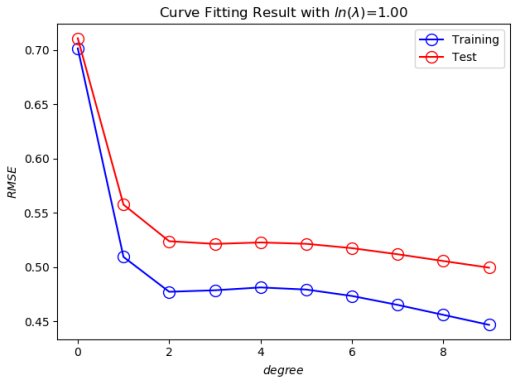


图 3-5 不同阶数 M 下拟合结果

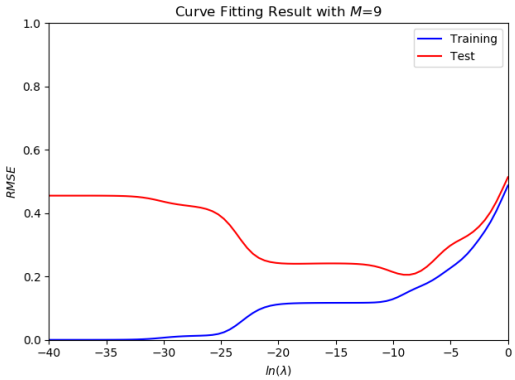


图 3-6 不同正则项系数 λ 下拟合结果

对比图 3-3 和图 3-5 容易发现，在引入正则化操作后，过拟合现象得到了很大程度的缓和。从图 3-6 中可以看出，正则项系数 λ 对于曲线拟合的结果有着较大的影响，需要根据实际进行调整。

4. 总结与展望

4.1 Strategies to Avoid Overfitting

过拟合是机器学习中经常出现的现象。从上文的描述来看，具有较高复杂度（对应于多项式模型阶数）的模型更容易造成模型过拟合，同时本文引入的正则化操作能够在一定程度上缓和过拟合的程度，使得多项式曲线更加平缓。通常来说，避免模型过拟合的方法主要有以下几种。

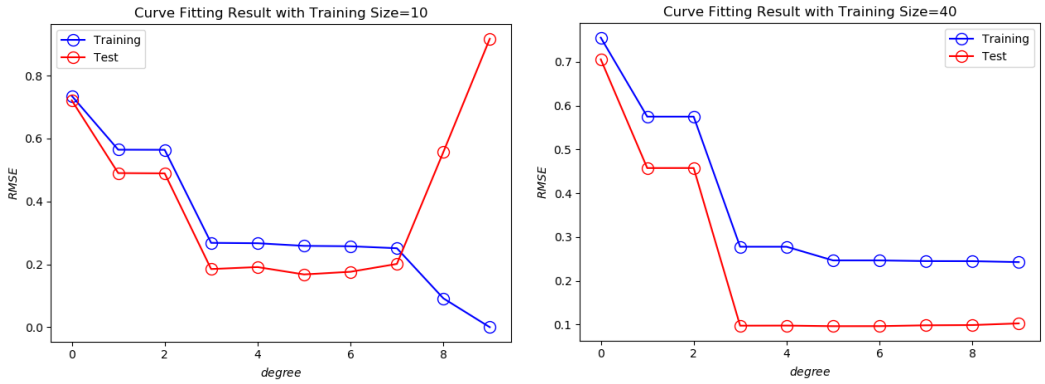
4.1.1 Model Simplification

从 3.3 小节对多项式阶数的分析中可以看出，随着模型阶数的不断增大，模型的复杂度逐步升高，得到的拟合曲线逐渐陡峭，在训练集上的表现不断增强，但无法应对新的测试数据，出现了较为严重的过拟合现象。因此，适当降低模型复杂度能够在一定程度上避免过拟合现象。

4.1.2 Introduce More Training Data

在指定模型复杂度的条件下，过拟合发生的主要原因是给定的训练数据不足，难以发现数据整体趋势，因此适当增加训练数据能够提高模型的泛化能力，在一定程度上避免过拟合现象。图 4-1 (a)和图 4-1 (b)分别展示了训练数据大小为 10 和 40 的条件下，拟合结果随模型复杂度的变化情况（测试数据大小均为 100）。

从图中可以明显看出，扩充训练数据对于避免过拟合现象非常有效。



(a) Training Size =10　　(b) Training Size =40

图 4-1 不同阶数 M 下拟合结果

4.1.3 Regularization

从 3.4 小节对正则项的分析中可以看出，对于同样复杂度的模型，引入正则化操作能够缓和过拟合现象，使拟合曲线更加平滑，模型泛化能力更强。同时，正则项系数的选取对于模型训练至关重要，需要根据实际进行更新调整。

4.2 Conclusion

本文主要介绍了多项式曲线拟合，首先通过引入泰勒展开式说明选用多项式模型拟合数据的必要性，同时通过观察模型复杂度与拟合结果之间的关系，了解了过拟合现象，并剖析内部原因。针对该问题，提出了三种避免过拟合现象的方法，分别为简化模型、扩充训练集和正则化，其中本文详细介绍了正则化的数学表达以及正则项系数的选取。通过实验发现，本文提出的三种方法确实能够在一定程度上缓和过拟合现象，提高模型的泛化能力。

5. 关键代码分析

5.1 数据生成代码分析

根据 3.1 节数据集的产生方式，即 $y=\sin\left(2\pi x\right)+N\left(\mu=0.25,\sigma=1.0\right)$，对应的数据生成代码如表 5-1 所示。

表 5-1 数据生成

- 数据生成

```
def create_toy_data(func, sample_size, std):
    x = np.linspace(0, 1, sample_size)
    # 在 func(x)的基础上添加高斯噪声
    t = func(x) + np.random.normal(scale=std, size=x.shape)
    return x, t
```

5.2 普通曲线拟合代码分析

根据 2.3.2 节模型参数 $w^{*}=\left(X^{T}X\right)^{-1}X^{T}y$ 的求解过程，对应的数据变换和参数求解的代码如表 5-2 和 5-3 所示。

表 5-2 数据矩阵变换

- 数据矩阵变换

```
def transform(self, x):
    if x.ndim == 1:
        x = x[:, None]
    x_t = x.transpose()
    features = [np.ones(len(x))]
    for degree in range(1, self.degree + 1):
        for items in itertools.combinations_with_replacement(x_t, degree):
            features.append(functools.reduce(lambda x, y: x * y, items))
    return np.asarray(features).transpose()
```

表 5-3 模型参数求解

- 模型参数求解

```
def fit(self, X: np.ndarray, t: np.ndarray):
    self.w = np.linalg.pinv(X) @ t
    self.var = np.mean(np.square(X @ self.w - t))
```

5.3 正则化代码分析

根据 2.3.3 节引入正则项后模型参数 $w^* = \left(X^T X + \lambda I\right)^{-1} X^T y$ 的求解过程，对应的求解代码如表 5-4 所示。

表 5-4 模型参数求解（正则化）

- 模型参数求解（正则化）

```
def fit(self, X: np.ndarray, t: np.ndarray):
    eye = np.eye(np.size(X, 1))
    self.w = np.linalg.solve(self.alpha * eye + X.T @ X, X.T @ t)
```

参考文献

[1] 安世全. 泰勒公式及其应用[J]. 高等数学研究, 2001, 004(003):26-28.

[2] 唐仁献. Lagrange 基本插值多项式的性质及应用[J]. 零陵师专学报, 1995(S1):49-51.

[3] Bishop C M . Pattern Recognition and Machine Learning (Information Science and Statistics)[M]. Springer-Verlag New York, Inc. 2007.