作者	**生姜 DrGinger**
脚本	**生姜 DrGinger**
视频	**崔崔 CuiCui**
开源学习资源	**https://github.com/Visualize-ML**
平台	**https://www.youtube.com/@DrGinger_Jiang** **https://space.bilibili.com/3546865719052873** **https://space.bilibili.com/513194466**

10.3 多项式回归

本节你将掌握的核心技能：

- 通过人为构造高次幂扩展特征，从而建模非线性关系。
- 多项式回归的设计矩阵，扩展向量空间。
- 使用最小二乘法公式计算多项式回归系数。
- 理解欠拟合与过拟合。
- 二元甚至多元多项式回归，掌握交叉项与高次组合。

非线性特征

在本章前文的多元线性回归中，自变量 x_1、x_2 … x_D 都是可以通过实际测量得到的原始特征，比如身高、体重、年龄等。

但在实际建模中，我们并不总是局限于这些原始线性特征。特别是当单一特征不足以捕捉复杂非线性关系 (如图 1 所示)，我们可以通过特定方式“制造”新的特征。本节介绍的多项式回归就是在回归中增加“高次”非线性特征。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。
版权归清华大学出版社所有，请勿商用，引用请注明出处。
代码及 PDF 文件下载：https://github.com/Visualize-ML
本书配套微课视频均发布在 B 站——生姜 DrGinger：https://space.bilibili.com/513194466
欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

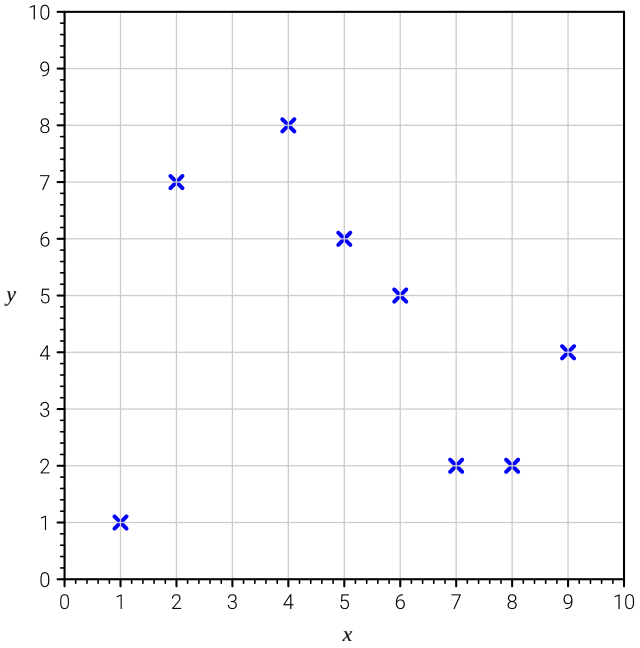


图 1. 呈现“非线性”关系的散点数据

多项式回归

如图 2 所示，虽然只有一个特征 x，但是我们可以人为构造 x^2、x^3 等非线性特征，进而扩展为 1、x、x^2、x^3 等一组特征。这种做法就引出了多项式回归 (polynomial regression)。

一元多项式回归是一种回归模型，它通过引入原始变量 x 的高次幂 (x^2、x^3 等等) 来拟合非线性的数据关系。m 阶多项式回归的解析式为

$$\hat{y} = b_0 + b_1 x + b_2 x^2 + \cdots + b_m x^m \tag{1}$$

其中，$b_2 x^2 + \cdots + b_m x^m$ 都是“人造”特征。

上式中，m 设为 1 时，我们便得到一元线性回归。也就是说，一元线性回归可以视作一元多项式回归的特例。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。
版权归清华大学出版社所有，请勿商用，引用请注明出处。
代码及 PDF 文件下载：https://github.com/Visualize-ML
本书配套微课视频均发布在 B 站——生姜 DrGinger：https://space.bilibili.com/513194466
欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

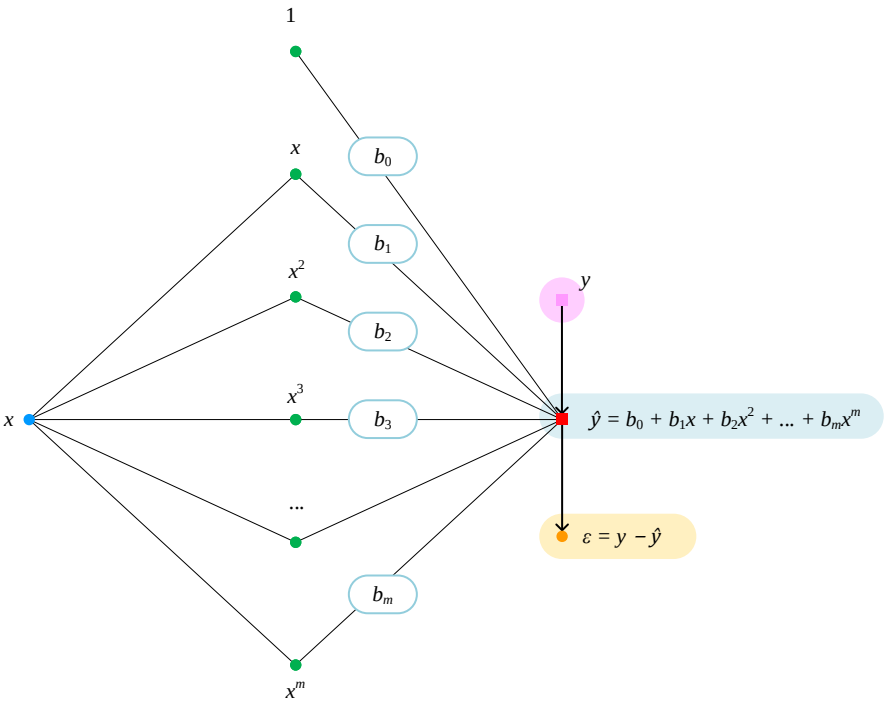


图 2. 一元多项式回归数据关系

虽然多项式回归的名字中含有“多项式”，但本质上仍是线性回归，因为它在线性地拟合这些非线性构造出来的新特征。

比如我们只有一个变量 x，如果我们想做一个二次多项式回归 (阶数为 2)，我们会使用三个特征：常数项 1、一次项 x、二次项 x^2，然后再去拟合。

增加多项式的阶数可以让模型更加灵活，拟合更复杂的数据，但阶数太高也容易导致过拟合——模型在训练数据上表现很好，但在新数据上却很差。本节最后会展开讨论这一点。

设计矩阵

从数据角度来看，如图 3 所示，在多项式回归中，我们不仅使用自变量的原始值，还将其不同阶数作为额外的特征，从而能够更好地拟合数据中非线性特征。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。
版权归清华大学出版社所有，请勿商用，引用请注明出处。
代码及 PDF 文件下载：https://github.com/Visualize-ML
本书配套微课视频均发布在 B 站——生姜 DrGinger：https://space.bilibili.com/513194466
欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

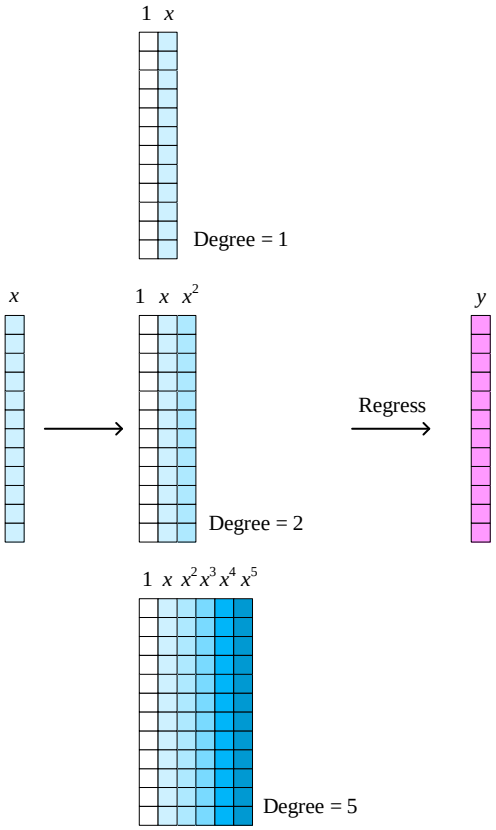


图 3．构造一元多项式回归中的设计矩阵

比如一元 5 阶多项式回归的设计矩阵 $\boldsymbol{X}$ 为

$$\boldsymbol{X}=\begin{bmatrix}\boldsymbol{1} & \boldsymbol{x} & \boldsymbol{x}\odot\boldsymbol{x} & \cdots & \boldsymbol{x}\odot\boldsymbol{x}\odot\boldsymbol{x}\odot\boldsymbol{x}\odot\boldsymbol{x}\end{bmatrix}=\begin{bmatrix}1 & x^{(1)} & \left(x^{(1)}\right)^2 & \cdots & \left(x^{(1)}\right)^5\\ 1 & x^{(2)} & \left(x^{(2)}\right)^2 & \cdots & \left(x^{(2)}\right)^5\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ 1 & x^{(n)} & \left(x^{(n)}\right)^2 & \cdots & \left(x^{(n)}\right)^5\end{bmatrix} \tag{2}$$

其中，$\boldsymbol{x}\odot\boldsymbol{x}$ 代表逐项积 (element-wise product)，也叫阿达玛乘积 (Hadamard product)。逐项积输入是两个相同形状的矩阵，输出是具有同样形状的、各个位置的元素等于两个输入矩阵相同位置元素的乘积的矩阵。

注意，向量是特殊的矩阵。

这样 $\boldsymbol{y}$ 可以写成

$$\underbrace{\begin{bmatrix}y^{(1)}\\ y^{(2)}\\ \vdots\\ y^{(n)}\end{bmatrix}}_{\boldsymbol{y}}=\underbrace{\begin{bmatrix}1 & x^{(1)} & \left(x^{(1)}\right)^2 & \cdots & \left(x^{(1)}\right)^5\\ 1 & x^{(2)} & \left(x^{(2)}\right)^2 & \cdots & \left(x^{(2)}\right)^5\\ \vdots & \vdots & \vdots & \ddots & \vdots\\ 1 & x^{(n)} & \left(x^{(n)}\right)^2 & \cdots & \left(x^{(n)}\right)^5\end{bmatrix}}_{\hat{\boldsymbol{X}}}\underbrace{\begin{bmatrix}b_0\\ b_1\\ b_2\\ \vdots\\ b_5\end{bmatrix}}_{\boldsymbol{b}}+\underbrace{\begin{bmatrix}\varepsilon^{(1)}\\ \varepsilon^{(2)}\\ \vdots\\ \varepsilon^{(n)}\end{bmatrix}}_{\boldsymbol{\varepsilon}} \tag{3}$$

从函数图像角度来讲，如图 4 所示，一元多项式回归模型是若干曲线叠加的结果。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。
版权归清华大学出版社所有，请勿商用，引用请注明出处。
代码及 PDF 文件下载：https://github.com/Visualize-ML
本书配套微课视频均发布在 B 站——生姜 DrGinger：https://space.bilibili.com/513194466
欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

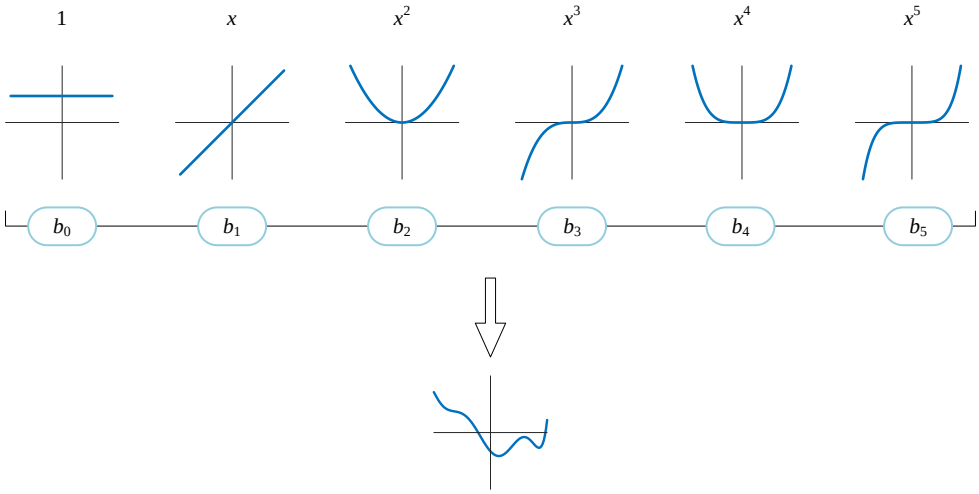


图 4．一元五次函数可以看作是 6 个图像叠加的结果，图片来自《机器学习》

求解多项式回归

求解一元 (x) 多项式回归中参数 $\boldsymbol{b}$ 和本章前文多元线性回归用的方法一致，如图 5 所示。一元多项式回归中的设计矩阵 $\boldsymbol{X}$ 列向量也张成一个多维向量空间。

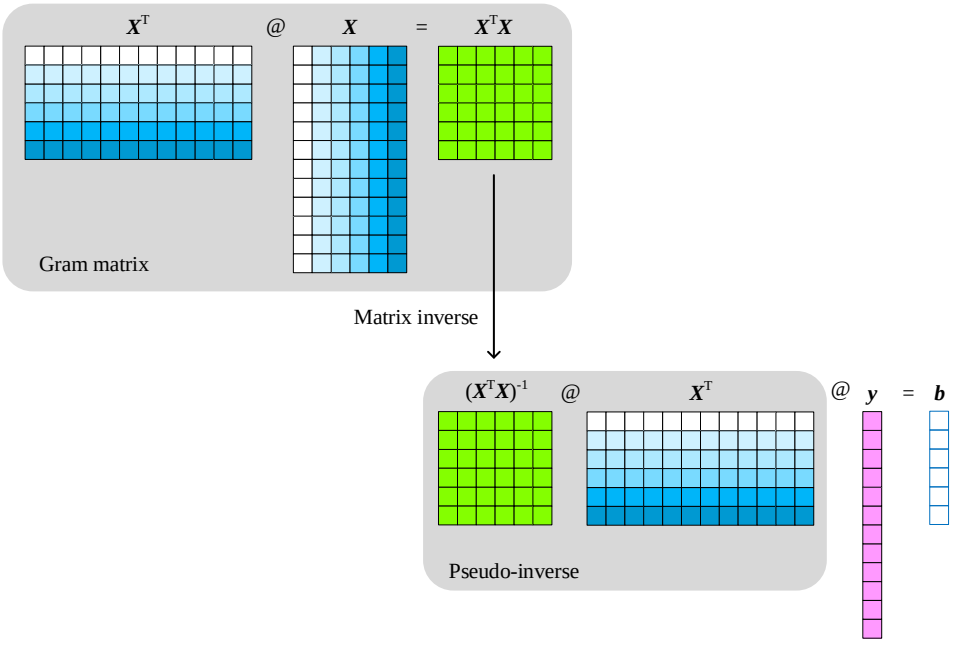


图 5. 计算 $\boldsymbol{b}$，一元多项式回归

下面看一个具体的示例。

一元二次多项式回归

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。
版权归清华大学出版社所有，请勿商用，引用请注明出处。
代码及 PDF 文件下载：https://github.com/Visualize-ML
本书配套微课视频均发布在 B 站——生姜 DrGinger：https://space.bilibili.com/513194466
欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

图 1 中散点数据的自变量列向量 $\boldsymbol{x}$、因变量列向量 $\boldsymbol{y}$ 分别为

$$\boldsymbol{x}=\begin{bmatrix}1\\2\\4\\5\\6\\7\\8\\9\end{bmatrix},\quad \boldsymbol{y}=\begin{bmatrix}1\\7\\8\\6\\5\\2\\2\\4\end{bmatrix} \tag{4}$$

举个例子，阶数为 2 的一元 (x) 多项式回归 (一元二次多项式回归) 对应的模型为

$$\hat{y}=b_0+b_1x+b_2x^2 \tag{5}$$

在多项式回归分析中，“二次”指的是变量的平方项参与了模型，也就是说除了常数项 b_0、b_1x 以外，模型中还包含 b_2x^2。(5) 中所有项的次数中最高的称为多项式的次数，即 2。

一元二次多项式回归设计矩阵 $\boldsymbol{X}$ 为

$$\boldsymbol{X}=\begin{bmatrix}\boldsymbol{1} & \boldsymbol{x} & \boldsymbol{x}\odot\boldsymbol{x}\end{bmatrix}=\begin{bmatrix}1&1&1\\1&2&4\\1&4&16\\1&5&25\\1&6&36\\1&7&49\\1&8&64\\1&9&81\end{bmatrix} \tag{6}$$

如图 6 所示，向量 $\boldsymbol{y}$ 朝 span($\boldsymbol{1}$, $\boldsymbol{x}$, $\boldsymbol{x}\odot\boldsymbol{x}$) 正交投影的结果就是 $\hat{\boldsymbol{y}}$。

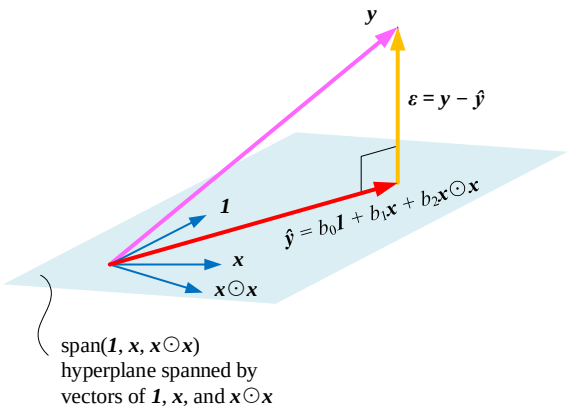


图 6. $\boldsymbol{y}$ 朝 span($\boldsymbol{1}$, $\boldsymbol{x}$, $\boldsymbol{x}\odot\boldsymbol{x}$) 投影，一元二次多项式线性回归

用和多元线性回归同样的方法计算 $\boldsymbol{b}$

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。
版权归清华大学出版社所有，请勿商用，引用请注明出处。
代码及 PDF 文件下载：https://github.com/Visualize-ML
本书配套微课视频均发布在 B 站——生姜 DrGinger：https://space.bilibili.com/513194466
欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

$$
\begin{aligned}
\boldsymbol{b} &= \left(\boldsymbol{X}^{\mathrm{T}}\boldsymbol{X}\right)^{-1}\boldsymbol{X}^{\mathrm{T}}\boldsymbol{y} \\
&= \left(\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 4 & 16 \\ 1 & 5 & 25 \\ 1 & 6 & 36 \\ 1 & 7 & 49 \\ 1 & 8 & 64 \\ 1 & 9 & 81 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 4 & 16 \\ 1 & 5 & 25 \\ 1 & 6 & 36 \\ 1 & 7 & 49 \\ 1 & 8 & 64 \\ 1 & 9 & 81 \end{bmatrix}\right)^{-1} \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 4 \\ 1 & 4 & 16 \\ 1 & 5 & 25 \\ 1 & 6 & 36 \\ 1 & 7 & 49 \\ 1 & 8 & 64 \\ 1 & 9 & 81 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} 1 \\ 7 \\ 8 \\ 6 \\ 5 \\ 2 \\ 2 \\ 4 \end{bmatrix} \\
&= \begin{bmatrix} 8 & 42 & 276 \\ 42 & 276 & 1998 \\ 276 & 1998 & 15252 \end{bmatrix}^{-1} \begin{bmatrix} 35 \\ 173 \\ 1037 \end{bmatrix} \\
&= \begin{bmatrix} 1.636 & -0.670 & 0.058 \\ -0.670 & 0.344 & -0.033 \\ 0.058 & -0.033 & 0.003 \end{bmatrix} \begin{bmatrix} 35 \\ 173 \\ 1037 \end{bmatrix} \\
&= \begin{bmatrix} 1.655 \\ 1.926 \\ -0.214 \end{bmatrix} = \begin{bmatrix} b_0 \\ b_1 \\ b_2 \end{bmatrix}
\end{aligned} \tag{7}
$$

这样我们便得到一元二阶多项式回归模型

$$
\hat{y} = 1.655 + 1.927x - 0.214x^2 \tag{8}
$$

图 7 中的红色抛物线对应上述解析式。

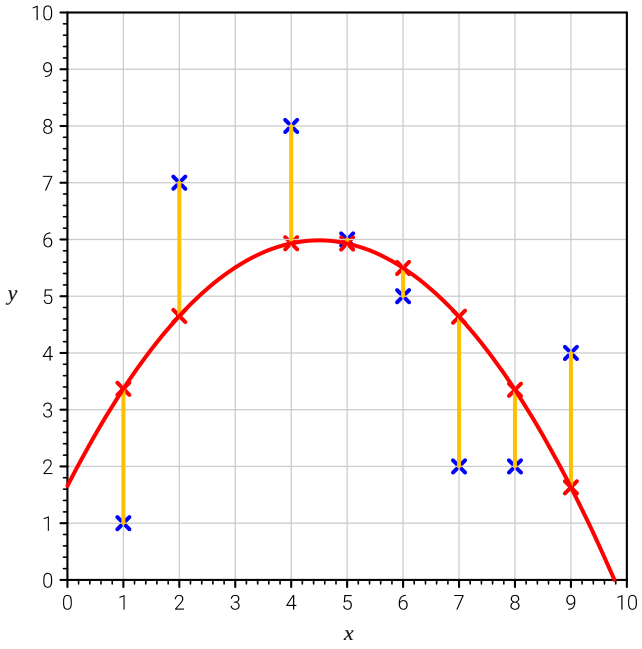


图 7. 阶数为 2 的一元多项式回归

图 7 中橙色线段为误差，对应向量 $\boldsymbol{\varepsilon}$

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。
版权归清华大学出版社所有，请勿商用，引用请注明出处。
代码及 PDF 文件下载：https://github.com/Visualize-ML
本书配套微课视频均发布在 B 站——生姜 DrGinger：https://space.bilibili.com/513194466
欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

$$\boldsymbol{\varepsilon} = \boldsymbol{y} - \hat{\boldsymbol{y}} = \begin{bmatrix} -2.368 \\ 2.349 \\ 2.068 \\ 0.070 \\ -0.498 \\ -2.638 \\ -1.35 \\ 2.367 \end{bmatrix} \tag{9}$$

误差的平方之和为

$$\|\boldsymbol{\varepsilon}\|_2^2 = \boldsymbol{\varepsilon}^{\mathrm{T}}\boldsymbol{\varepsilon} = \begin{bmatrix} -2.368 \\ 2.349 \\ 2.068 \\ 0.070 \\ -0.498 \\ -2.638 \\ -1.35 \\ 2.367 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} -2.368 \\ 2.349 \\ 2.068 \\ 0.070 \\ -0.498 \\ -2.638 \\ -1.35 \\ 2.367 \end{bmatrix} = 30.040 \tag{10}$$

一元三次多项式回归

较低的阶数可能无法很好地捕捉数据中的复杂关系。比如，图 7 中橙色线段长度还是很大。

下面让我们把阶数提高到 3!

阶数为 3 的一元 (x) 多项式回归 (一元三次多项式回归) 对应的模型为

$$\hat{y} = b_0 + b_1 x + b_2 x^2 + b_3 x^3 \tag{11}$$

请大家自行写出一元三次多项式回归设计矩阵 $\boldsymbol{X}$，并计算 $\boldsymbol{b}$。

图 8 中的红色曲线便是一元三次多项式回归模型，对应解析式

$$\hat{y} = -8.539 + 12.211x - 2.643x^2 + 0.16x^3 \tag{12}$$

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。
版权归清华大学出版社所有，请勿商用，引用请注明出处。
代码及 PDF 文件下载：https://github.com/Visualize-ML
本书配套微课视频均发布在 B 站——生姜 DrGinger：https://space.bilibili.com/513194466
欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

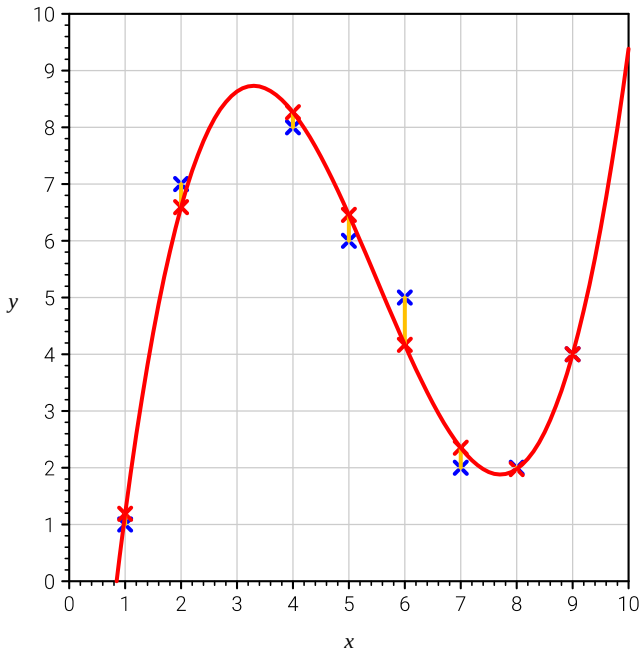


图 8. 阶数为 3 的一元多项式回归

图 8 中橙色线段为误差，对应向量 $\boldsymbol{\varepsilon}$

$$\boldsymbol{\varepsilon} = \boldsymbol{y} - \hat{\boldsymbol{y}} = \begin{bmatrix} -0.189 \\ 0.407 \\ -0.266 \\ -0.457 \\ 0.834 \\ -0.352 \\ 0.023 \\ -0.001 \end{bmatrix} \tag{13}$$

误差的平方之和为

$$\|\boldsymbol{\varepsilon}\|_2^2 = \boldsymbol{\varepsilon}^{\mathrm{T}}\boldsymbol{\varepsilon} = \begin{bmatrix} -0.189 \\ 0.407 \\ -0.266 \\ -0.457 \\ 0.834 \\ -0.352 \\ 0.023 \\ -0.001 \end{bmatrix}^{\mathrm{T}} \begin{bmatrix} -0.189 \\ 0.407 \\ -0.266 \\ -0.457 \\ 0.834 \\ -0.352 \\ 0.023 \\ -0.001 \end{bmatrix} = 1.302 \tag{14}$$

和二次多项式回归相比，三次多项式回归的误差明显减小。

过拟合

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。
版权归清华大学出版社所有，请勿商用，引用请注明出处。
代码及 PDF 文件下载：https://github.com/Visualize-ML
本书配套微课视频均发布在 B 站——生姜 DrGinger：https://space.bilibili.com/513194466
欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

然而，较高的阶数可能会导致过拟合。过拟合 (overfitting) 是指模型在训练数据上表现得很好，但在新数据上表现较差的现象。当多项式回归的阶数过高时，模型可能会过度适应训练数据中的噪声和细节，从而失去了泛化能力 (generalization capability, generalization)。

这意味着模型对于新的、未见过的数据可能无法进行准确的预测，因为它在训练数据上“记住了”许多细微的变化，而这些变化可能在真实数据中并不存在。

以学习线性代数为例，欠拟合就像“学业不精”，学生只掌握了皮毛知识，可能连矩阵乘法都没有搞清楚，基础题都常常做错。这说明说明学习内容太少、太浅，练习不够，没有掌握线性代数的基础知识。

而过拟合就像“死读书”，学生死记硬背了所有例题的解解题技巧；但一旦题型稍有变化，或者要求把书本知识和实践应用相结合，就无所适从。这说明学习太机械、太依赖细节，缺乏提炼和总结，无法举一反三。

真正好的学习状态，就像模型在训练集 (例题) 和测试集 (新题、实践应用) 上都表现良好，既掌握了规律，又具备了灵活应变的能力。

图 9 所示为阶数为 8 的一元多项式回归结果，对应的解析式为

$$\hat{y} = 7.123 - 11.430x - 2.828x^2 + 15.342x^3 - 9.463x^4 + 2.599x^5 - 0.370x^6 + 0.027x^7 - 0.001x^8 \quad (15)$$

如图 9 所示，红色回归曲线“完美”地穿越了所有数据点，这意味着误差几乎为 0！

但是，这并不是理想的模型。阶数越高，多项式回归模型越能够适应训练数据，但也越容易在测试数据或实际应用中表现不佳。

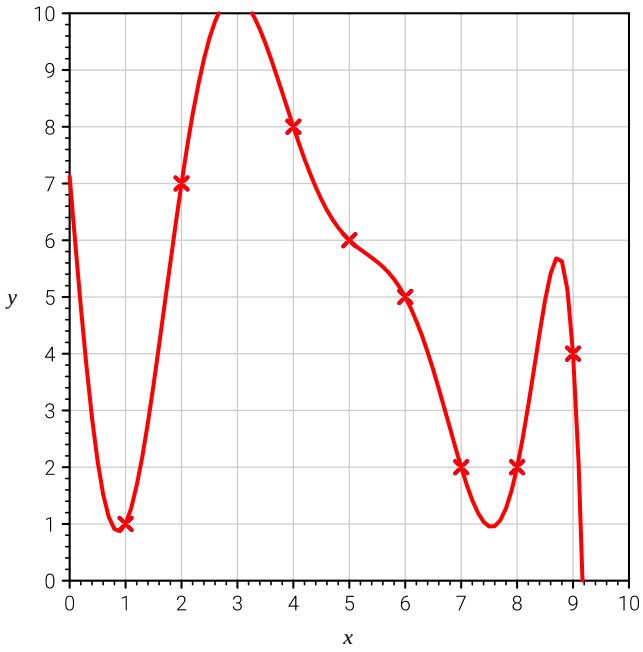
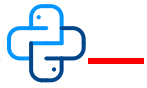


图 9. 阶数为 8 的一元多项式回归



LA_09_08_01.ipynb 求解次数为 3 的一元多项式回归，并绘制图 8。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。
版权归清华大学出版社所有，请勿商用，引用请注明出处。
代码及 PDF 文件下载：https://github.com/Visualize-ML
本书配套微课视频均发布在 B 站——生姜 DrGinger：https://space.bilibili.com/513194466
欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

LA_09_08_02.ipynb 则是用 Scikit-learn 提供的 LinearRegression 工具完成模型训练与预测，并绘制图 9。

二元多项式回归

既然有一元多项式回归，自然也就有二元，乃至多元的多项式回归。

我们前面讲过，一元二次多项式回归是在线性回归模型 $\hat{y} = b_0 + b_1 x$ 中引入 $b_1 x^2$ 这样的高次项，现在我们把变量扩展到两个：x_1、x_2，就可以构造二元多项式回归。

比如说，二元二次多项式回归，我们除了原始的 x_1、x_2 之外，还会引入如下组合 x_1^2 、 x_2^2 、以及交叉项 $x_1 x_2$ 。

因此，二元二次多项式回归模型的形式就是

$$\hat{y} = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_1^2 + b_4 x_2^2 + b_5 x_1 x_2 \tag{16}$$

上式相当于图 10 中图形的叠加。

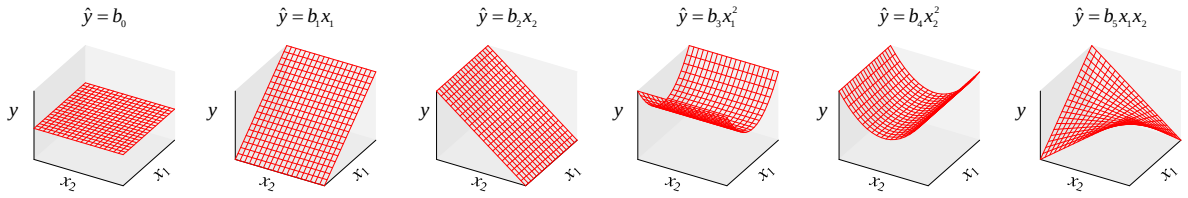


图 10. 二元二次多项式中各项图像

(16) 可以写成

$$\hat{y} = \begin{bmatrix} 1 & x_1 & x_2 & x_1^2 & x_2^2 & x_1 x_2 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \end{bmatrix} \tag{17}$$

这意味着二元二次多项式回归的设计矩阵 $\boldsymbol{X}$ 为

$$\boldsymbol{X} = \begin{bmatrix} \boldsymbol{1} & \boldsymbol{x}_1 & \boldsymbol{x}_2 & \boldsymbol{x}_1 \odot \boldsymbol{x}_1 & \boldsymbol{x}_2 \odot \boldsymbol{x}_2 & \boldsymbol{x}_1 \odot \boldsymbol{x}_2 \end{bmatrix} \tag{18}$$

这个模型不仅能拟合两个自变量的线性关系，也能捕捉它们之间的非线性交互。

如果我们继续往上扩展，比如看二元三次多项式回归，构造出的特征会更多，包括三次项。如图 11 所示，随着次数增多，我们不断纳入更多的高次项。通过这些人工构造的特征，模型可以更好地拟合曲面状的复杂数据分布。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。
版权归清华大学出版社所有，请勿商用，引用请注明出处。
代码及 PDF 文件下载：https://github.com/Visualize-ML
本书配套微课视频均发布在 B 站——生姜 DrGinger：https://space.bilibili.com/513194466
欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com

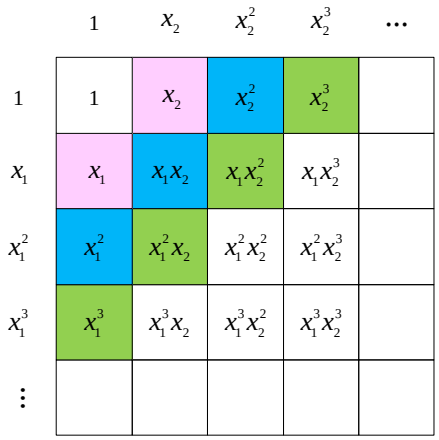


图 11. 多项式中的不同次数的项

此外，(16) 中的二次项部分可以写成矩阵乘法形式

$$b_3x_1^2+b_4x_2^2+b_5x_1x_2=\begin{bmatrix}x_1 & x_2\end{bmatrix}\begin{bmatrix}b_3 & b_5/2\\ b_5/2 & b_4\end{bmatrix}\begin{bmatrix}x_1\\ x_2\end{bmatrix}=\boldsymbol{x}^{\mathrm{T}}\begin{bmatrix}b_3 & b_5/2\\ b_5/2 & b_4\end{bmatrix}\boldsymbol{x} \tag{19}$$

这是本书后续要介绍的二次型 (quadratic form)。

请大家用 DeepSeek/ChatGPT 等工具完成本节如下习题。

Q1. 以下代码产生一元回归自变量、因变量数据，请大家用 Python 编程计算一元 3 阶多项式回归中的 $\boldsymbol{b}$、$\hat{\boldsymbol{y}}$、$\boldsymbol{\varepsilon}$。

```
## 初始化
import numpy as np

## 创建数据
np.random.seed(0)
n_samples = 100

# 自变量数据
x = np.random.uniform(0,4,num)

# 因变量数据
y = np.sin(0.4*np.pi * X) + 0.4 * np.random.randn(num)
```

Q2. 用 sklearn.preprocessing 模块中的 PolynomialFeatures，以及 sklearn.linear_model 模块中的 LinearRegression 求解 **Q1** 一元 3 阶多项式回归，并比较结果。

本 PDF 文件为作者草稿，发布目的为方便读者在移动终端学习，终稿内容以清华大学出版社纸质出版物为准。
版权归清华大学出版社所有，请勿商用，引用请注明出处。
代码及 PDF 文件下载：https://github.com/Visualize-ML
本书配套微课视频均发布在 B 站——生姜 DrGinger：https://space.bilibili.com/513194466
欢迎大家批评指教，本书专属邮箱：jiang.visualize.ml@gmail.com