

# 第四讲 数学基础和线性回归

2025/03/29

文件: [PJ-regression.zip](#)、[requirements.txt](#)、[Anaconda for Windows](#)、[Anaconda for Apple M](#)、[Anaconda for Apple Intel](#)

## 1. 机器学习简介

### 1.1 什么是机器学习？

- **机器学习 (Machine Learning)** 是指计算机在**数据**中学习**规律**，并根据所得规律对未来数据进行**预测**，是人工智能的一个重要分支
- Arthur Samuel的定义：在进行特定编程的情况下，给予计算机学习能力的领域
- Tom Mitchell的定义：如果计算机程序在**任务T**上的性能（由**P**衡量）随着**经验E**而提高，则称计算机程序从经验E中学习某类任务T

### 1.2 机器学习的分类

- 按学习方法分：
  - **监督学习 (Supervised Learning)**：
    - 我们教计算机如何去完成任务
    - 回归、分类
  - **无监督学习 (Unsupervised Learning)**：
    - 计算机自己进行学习（无教师）
    - 聚类、降维
- 按输出任务分：
  - 预测气温：**回归 (Regression)**，输出变量是**连续的**
  - 预测天气：**分类 (Classification)**，输出变量是**离散的**

## 2. 数学基础

### 2.1 导数的定义

- 导数的定义：
  - 设函数  $y = f(x)$  在点  $x_0$  的某个领域内有定义，当自变量  $x$  在  $x_0$  处有增量  $\Delta x$  也在该领域时，对应的函数取得增量：
$$\Delta y = f(x_0 + \Delta x) - f(x_0)$$
  - 如果  $\Delta y$  与  $\Delta x$  之比，当  $\Delta x \rightarrow 0$  时，极限存在，则称函数  $f(x)$  在  $x_0$  处可导，并称这个极限为函数  $y = f(x)$  在点  $x_0$  处的导数记为  $f'(x_0)$ ，即：

$$f'(x_0) = \lim_{\Delta x \rightarrow 0} \frac{\Delta y}{\Delta x} = \lim_{\Delta x \rightarrow 0} \frac{f(x_0 + \Delta x) - f(x_0)}{\Delta x}$$

- 导数的**几何意义**：**函数斜率**

### 2.2 多元函数与偏导数

- 多元函数
  - 有两个及以上自变量的函数统称为多元函数
  - 例：  $f(x, y) = x^2 + 3xy + y^2$
- 偏导数
  - 多元函数关于其中一个变量的导数而保持其他变量恒定
  - 例：  $f'_x = 2x + 3y$  及  $f'_y = 3x + 2y$

### 2.3 常用函数的导数

- $y = c$  ( $c$  为常数)，则  $y' = 0$
- $y = x^\alpha$ ，则  $y' = \alpha x^{\alpha-1}$
- $y = \alpha^x$ ，则  $y' = \alpha^x \ln \alpha$ ，特例：  $(e^x)' = e^x$
- $y = \log_a x$ ，则  $y' = \frac{1}{x \ln a}$ ，特例：  $(\ln x)' = \frac{1}{x}$
- $y = \sin x$ ，则  $y' = \cos x$
- $y = \cos x$ ，则  $y' = -\sin x$
- $y = \tan x$ ，则  $y' = \frac{1}{\cos^2 x} = \sec^2 x$

## 2.4 复合函数求导

- 复合函数求导的**链式法则**
  - $y = f(u)$  和  $u = g(x)$  , 其复合函数为  $y = f(g(x))$
  - 对  $y = f(g(x))$  求导, 有:  $y'_x = y'_u \cdot u'_x$

## 3. 寻优算法

### 3.1 什么是寻优?

- 寻优算法的任务, 就是找到函数的最小值
- 例如: 计算最小成本、最短路径、最小损失.....

### 3.2 梯度下降法

- **梯度下降法**就是通过沿着**负梯度方向**不断迭代, 找到梯度为 0 的位置
- 迭代公式:  $x_{n+1} = x_n - f'(x_n)$
- 梯度下降算法步骤:
  - i. 给定起始点  $x_0$  计算导数  $f'(x_0)$
  - ii. 迭代:  $x_{n+1} = x_n - \alpha f'(x_n)$   
 $\alpha$ : **步长**, 控制移动的距离
  - iii. 重复步骤 2, 直至  $f'(x_n)$  的绝对值足够小
- **步长  $\alpha$  影响**: 随着迭代次数增加, 导致函数值有不同变化:
  - $\alpha$ 太小: 迭代缓慢, 学习慢
  - $\alpha$ 适中: 适中
  - $\alpha$ 较大: 迭代来回震荡
  - $\alpha$ 过大: 反而会越过谷底, 不断上升

## 4. 线性回归

### 4.1 采样与回归

- **线性回归**: 用一条线来进行回归预测, 就是寻求一条直线尽可能拟合那些离散点
- **非线性回归**: 用曲线来进行回归预测

### 4.2 求解线性回归问题

- **求解线性回归的流程**
  - 建立合理的表达式  $f(x)$  :
    - **一元线性回归**:  $f(x) = wx + b$
    - **多元线性回归**:  $f(x) = w_1x_1 + w_2x_2 + \cdots + w_nx_n + b$

$$f(x) = \sum_{j=1}^n w_j x_j + b = W^T X + b$$

- 构建合理的准则以确定表达式中参数的值
  - $W$ : 权重 (weights)
  - $b$ : 偏置量 (bias)
- 对模型进行求解, 最终确定  $f(x)$

### 4.3 最小二乘

为使得误差最小, 就需要对误差进行计算和衡量:

与测试为  $w x_i + b$ , 真实值为  $y_i$

可以利用误差的平方和, 即  $\sum_{i=1}^m (y_i - (W^T x_i + b))^2$  对拟合效果进行衡量:

$$\min_{w, b} \sum_{i=1}^m (y_i - (W^T x_i + b))^2$$

其含义是寻找出一堆  $w$ 、 $b$  使得误差平方和最小。

因为是通过极小化误差平方和来求解回归问题, 因此被称为“**最小二乘方法 (Least Squares)**”。

为求解最小二乘问题, 需要将目标函数对参数  $w, b$  分别进行求导, 得到**最优解**需要满足的情况, 即

TODO: 实在懒得打了

由此得到了包含  $M$  个未知数的  $M$  个线性方程组，通过求解这个**线性方程组**（又被称为“**正则方程**” Normal Equation），就可以得到最小二乘问题的解。

## 4.4 过拟合问题

- 虽然利用最小二乘得到的多项式在每一个采样点上都拟合得很好，但在非采样点上的准确度却不高。
- 在采样有**噪声**的情况下，过拟合带来的危害更大。
- 如果有更多的数据，就能减轻过拟合问题。

## 4.5 模型泛化性

- **正确拟合**
  - 泛化能力是指模型不仅能在训练数据上表现良好，还能在测试数据（未见过的数据）上表现优异。
  - 模型适当复杂，能够准确捕捉数据的主要趋势和模式，能够泛化
  - 理想的模型应该能够在训练集上达到适度的拟合，同时能对测试集保持良好的预测效果。
- **欠拟合**
  - 模型是线性的，数据呈现复杂的非线性模式，线性模型无法很好地拟合这些数据。
  - 模型无法准确描述数据的趋势或模式
  - 在训练集和测试集上的误差都很高
- **过拟合**
  - 模型使用了非常高阶的多项式，拟合了每一个数据点，包括数据中的噪声，记住了细节，却忽略了数据的整体规律。
  - 在训练集上表现非常好，但在测试集上表现不佳。
  - 模型在新数据上的泛化能力较差。

## 作业：

Line 66~72:

```
# 构建并拟合线性回归模型
model = LinearRegression()    # 创建线性回归模型对象
model.fit(X_train, y_train)    # 使用数据拟合线性回归模型

# 使用训练好的模型预测
y_pred = model.predict(X_test) # 对特征进行预测
mse = mean_squared_error(y_test, y_pred)    # 计算均方误差（MSE）
```

Line 124~130:

```
# 定义线性回归模型并用训练数据进行拟合
lin_reg = LinearRegression()
lin_reg.fit(X_train, y_train)

# 输入测试数据并输出预测结果和评估指标
result(lin_reg, X_test, y_test)
```