# CHAPTER 4

# 机器学习和神经网络

主讲人：吴春彪

2025年5月25日

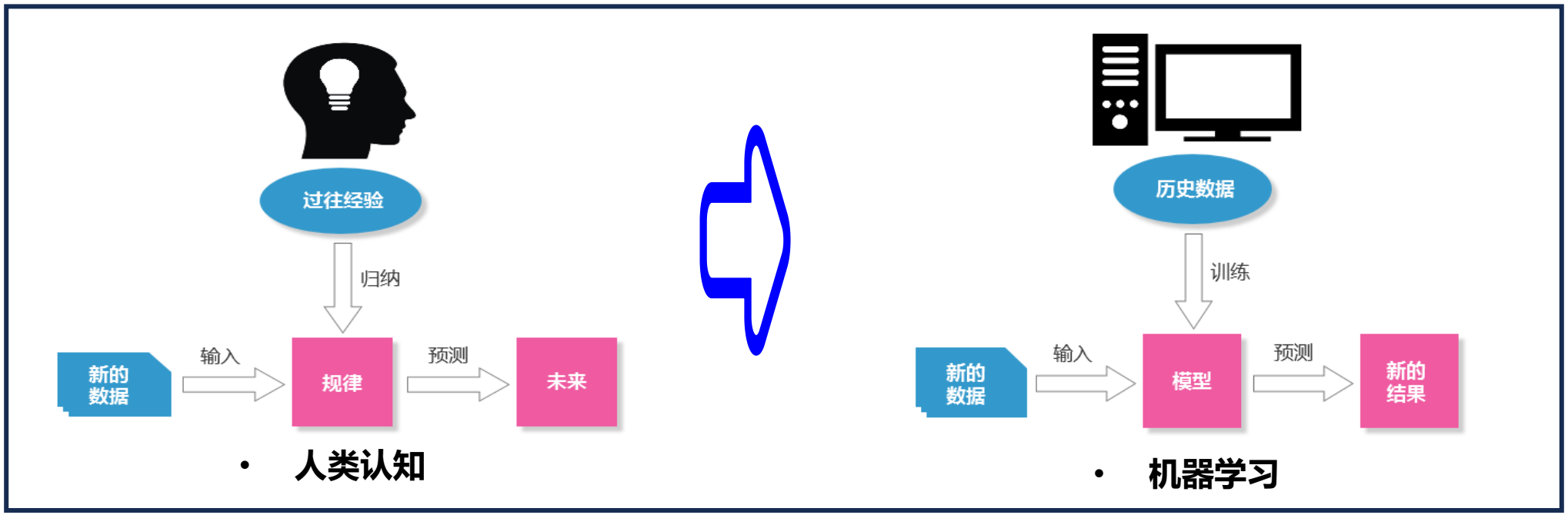
# 01

# 机器学习

# 机器学习

## 一、机器学习的概念

- 机器学习（Machine Learning，ML）：1959年提出，是人工智能 (AI) 的核心和重要的一个分支,机器学习能从有限的观测数据中学习（或“猜测”）出具有一般性的规律，并利用这些规对未知数据进行预测的方法，**机器通过数据（经验）来决策和预测。**




- 人类认知
- 机器学习

# 机器学习

## ■ 人工智能（ARTIFICIAL INTELLIGENCE）

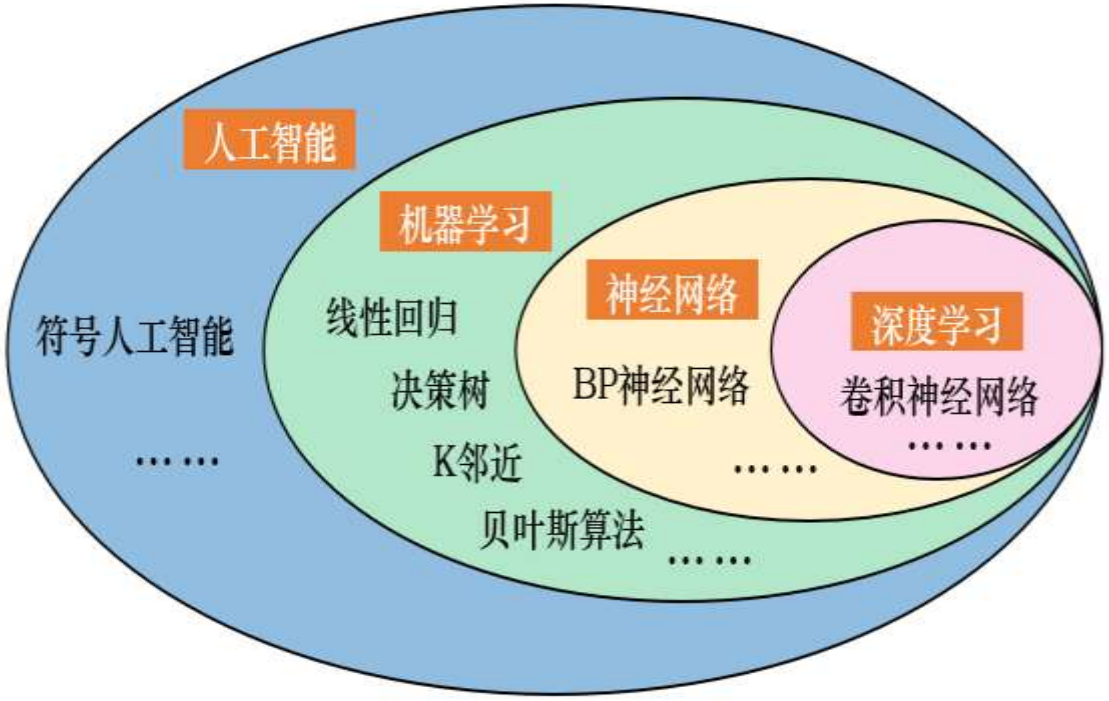
- 融汇工学、数学、医学、认知学等很多学科的一个交叉学科

## ■ 机器学习（MACHINE LEARNING）

- 人工智能中最火热的研究领域之一

## ■ 深度学习（DEEP LEARNING）

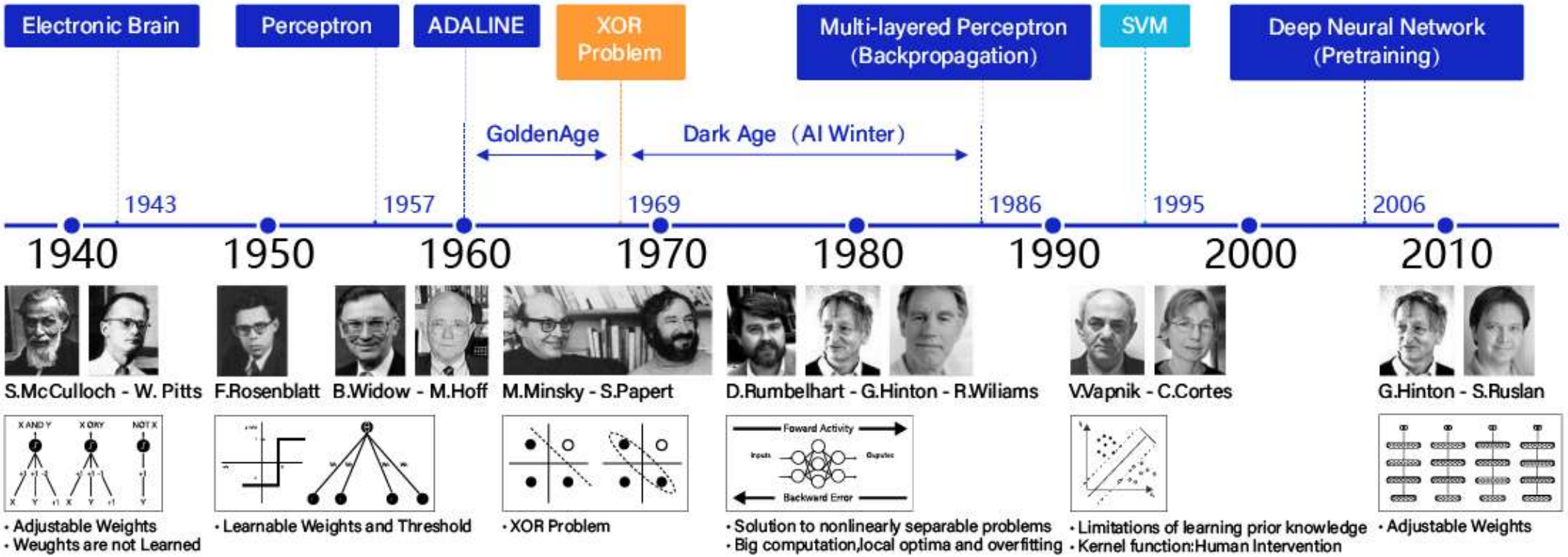
- 最火热的机器学习方法之一

# 机器学习

## 二、机器学习的历史

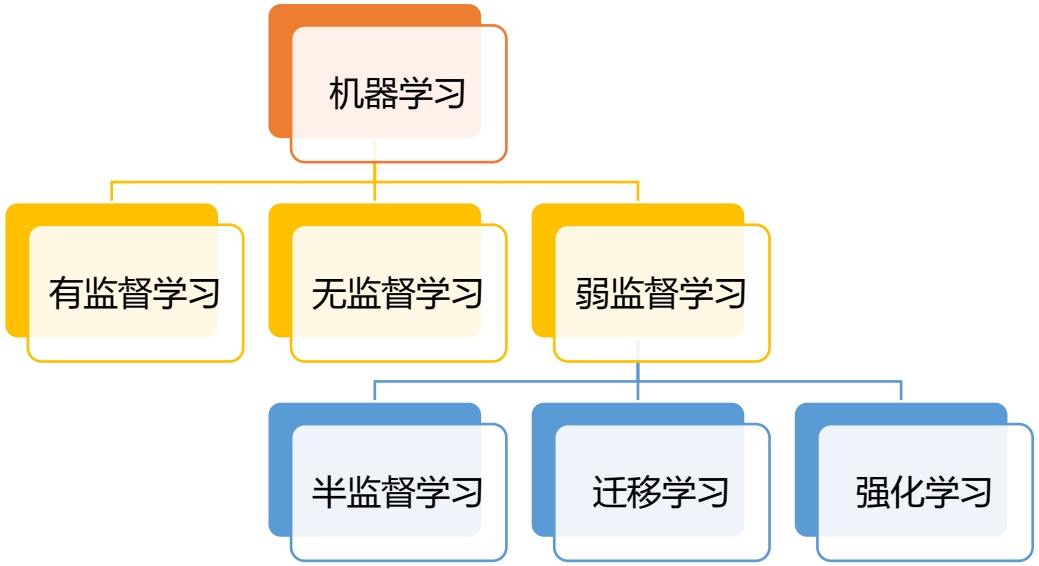
- 发展历程：1959s 感知机 → 1980s 神经网络 → 1990s SVM → 2006s 深度学习 → 2012s AlexNet →2016s AlphGo → 2017s Transformer → 2020s GPT-3 → 2023s ChatGPT → 多模态大模型 。**从符号学习到数据驱动，算力与算法协同推动AI革命。**



# 机器学习

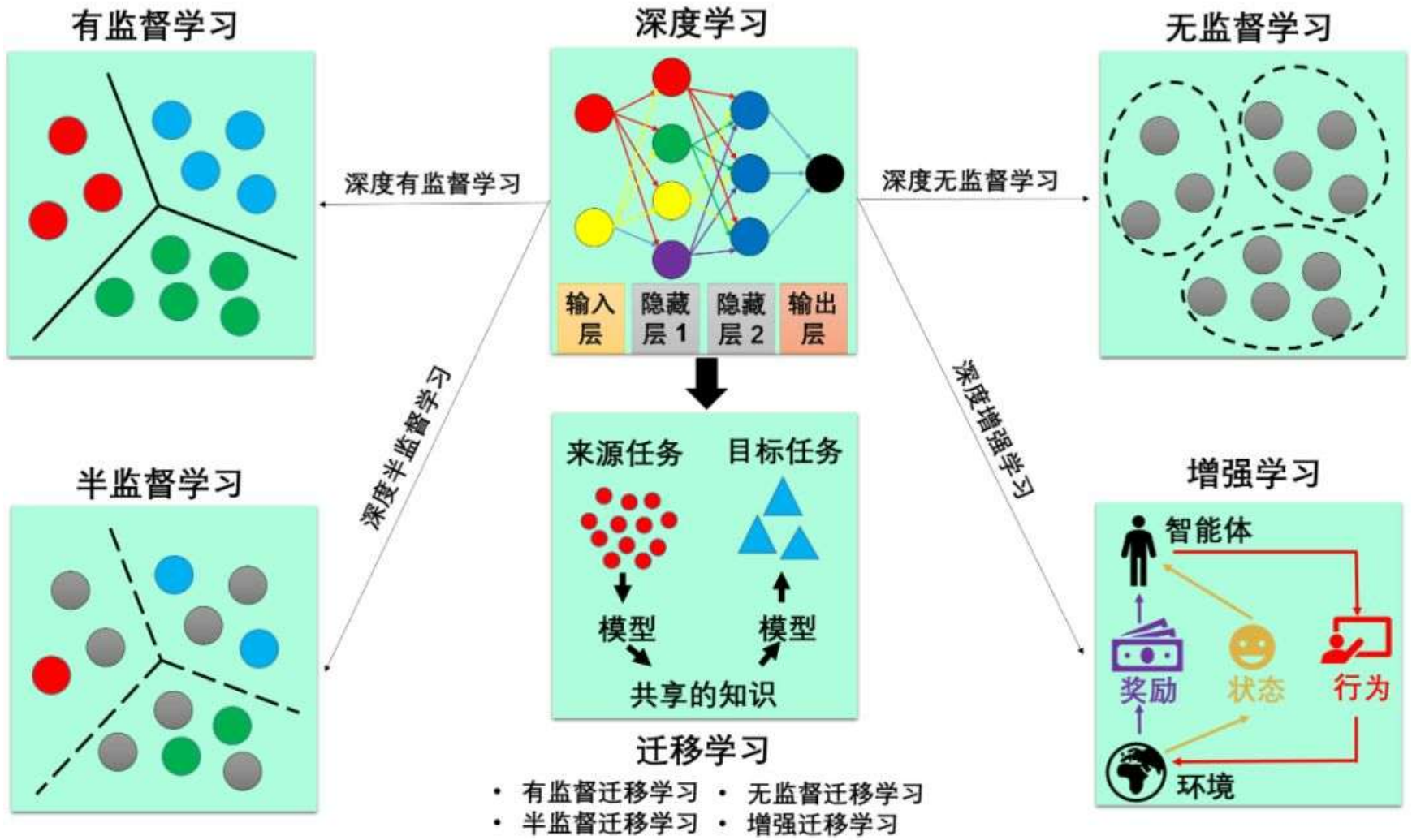
## 三、机器学习的类别

- 机器学习的主要类型：**监督学习**（或有导师学习）、**无监督学习**（或无导师学习）和**弱监督学习**




# 机器学习

## 三、机器学习的类别

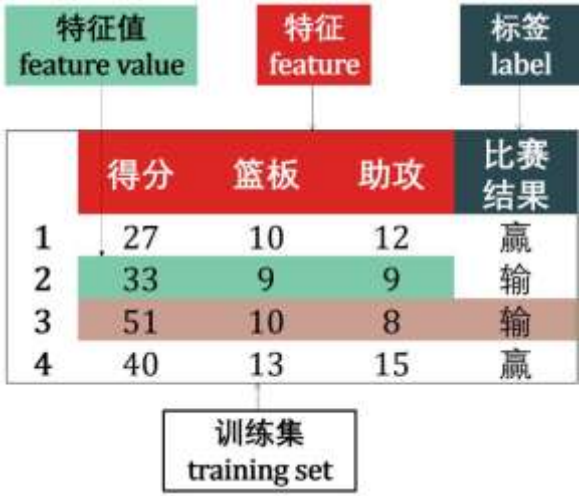


# 机器学习

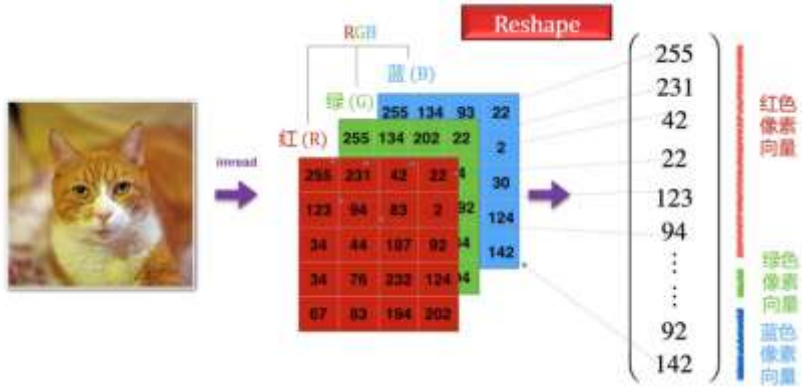
## 四、机器学习的三要素

- **机器学习的三要素：**


（1）**数据**：数据是机器学习的核心原料，包括**数字，图片，文字，语音**和**视频**等









# 机器学习

（2）**算法**：模型的设计框架，包括**监督学习、无监督学习和强化学习**等。选择合适算法（如神经网络、决策树）需结合任务需求，优化目标函数以**提升泛化能力**等


**监督学习：**

- 线性回归（Linear Regression）
- 逻辑回归（Logistic Regression）
- 支持向量机（SVM）
- K-近邻算法（KNN）
- 决策树（Decision Tree）
- 随机森林（Random Forest）

**无监督学习：**

- K-均值聚类（K-Means Clustering）
- 主成分分析（PCA）

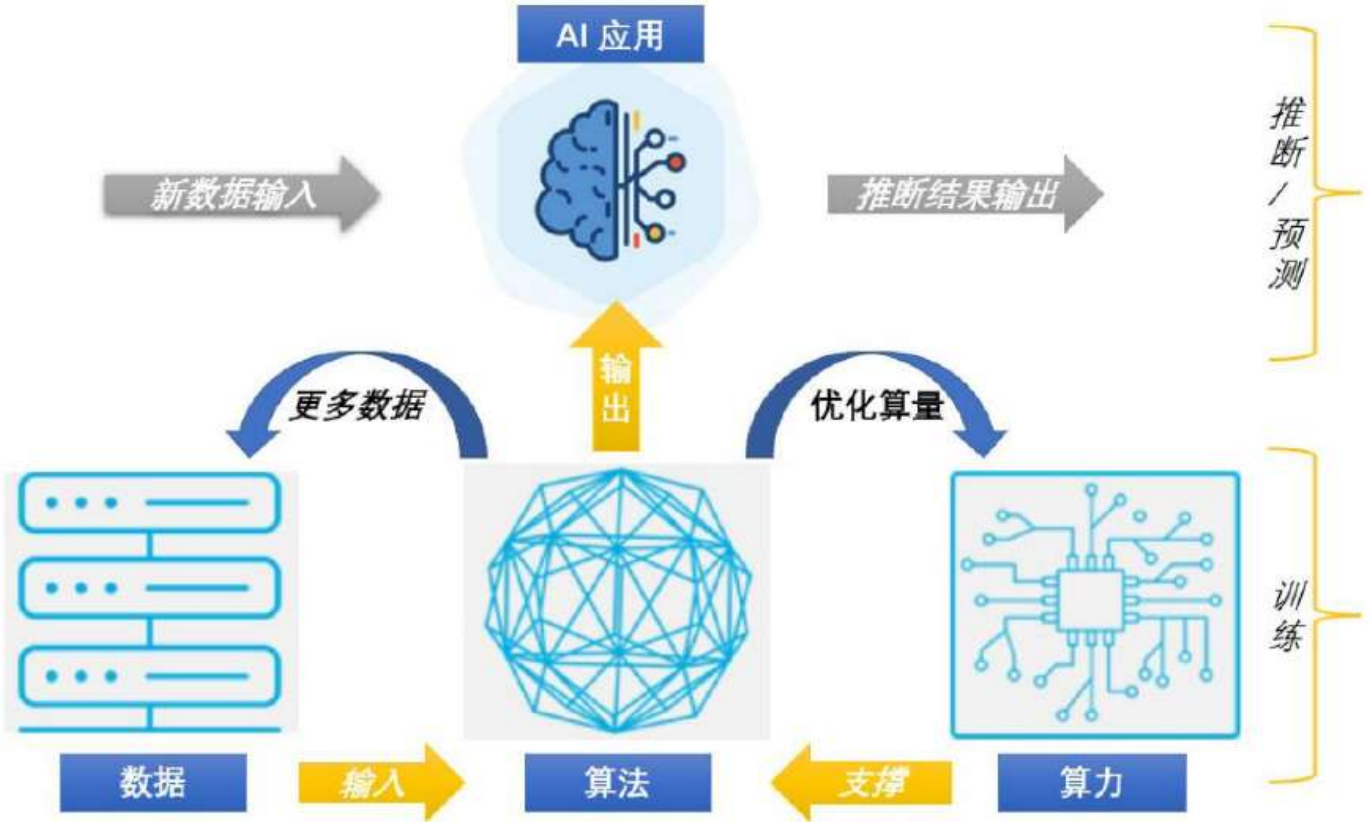
**深度学习：**

- 神经网络（Neural Networks）
- 卷积神经网络（CNN）
- 循环神经网络（RNN）

# 机器学习

（3）**算力**：指**硬件资源**（如GPU/TPU）和计算效率，支撑复杂模型训练与推理。分布式计算和并行优化可加速迭代，算力提升推动深度学习等大规模模型发展

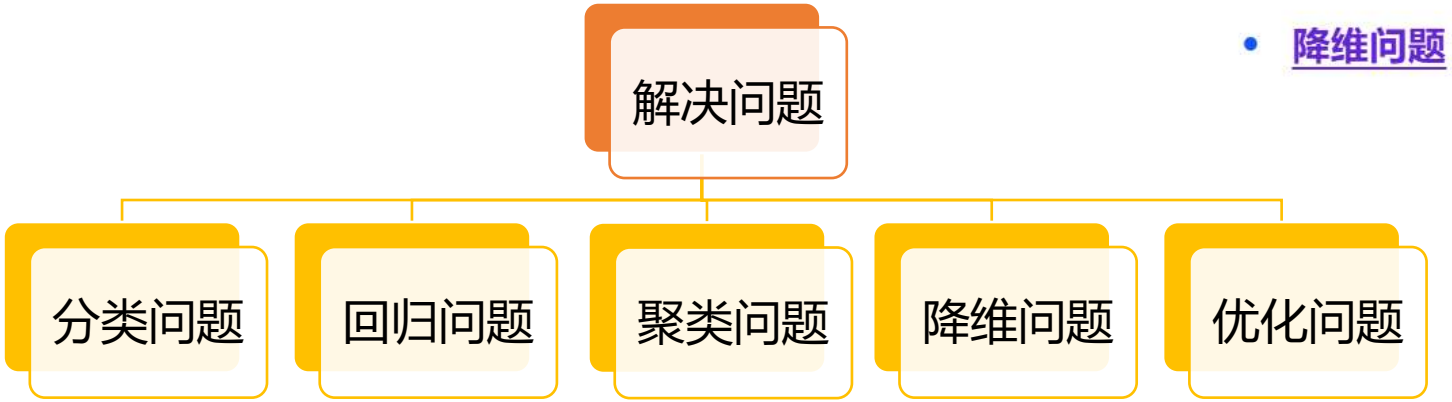


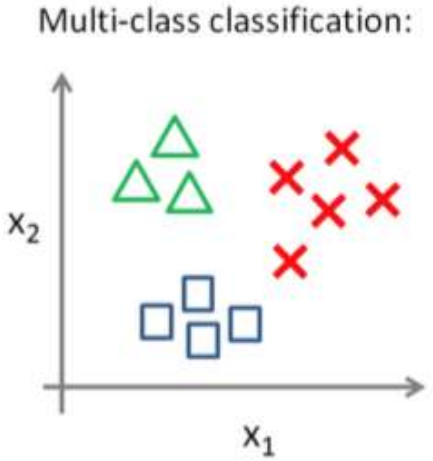


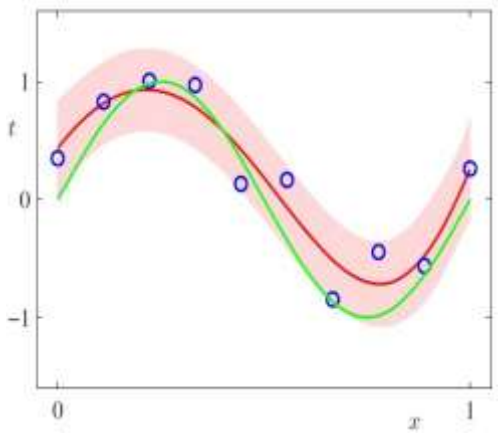
# 机器学习

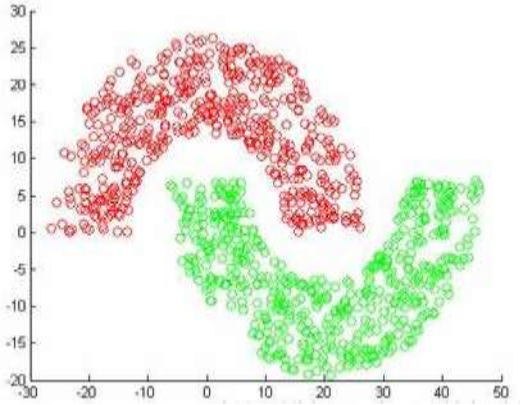
## 五、“机器学习”解决问题

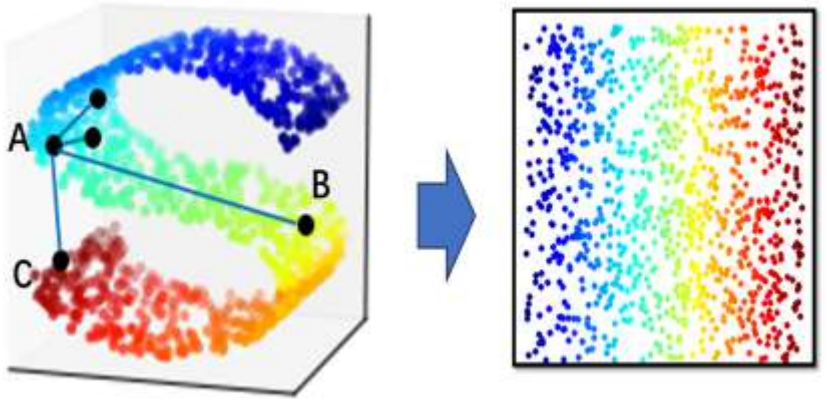
- **回归问题**：预测连续值，例如房价预测。
- **分类问题**：将样本分为不同类别，例如垃圾邮件检测。
- **聚类问题**：将数据自动分组，例如客户细分。
- **降维问题**：将数据降到低维度，例如主成分分析（PCA）。



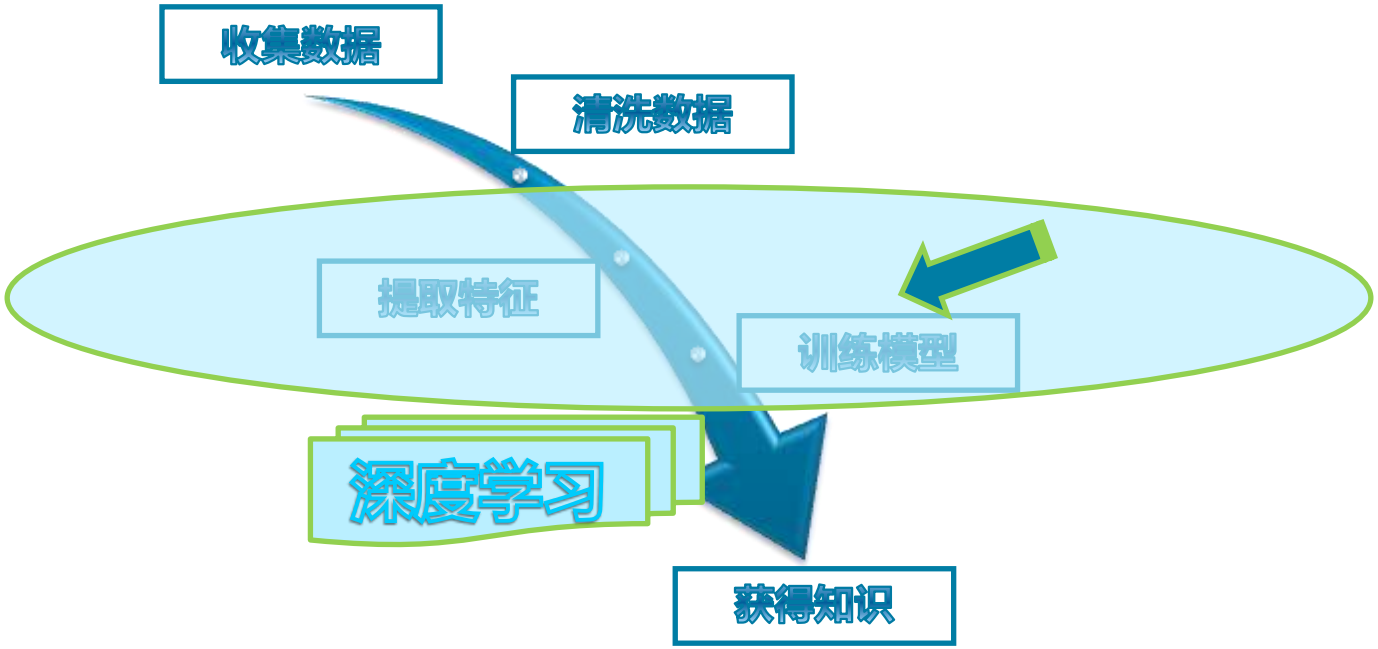





# 机器学习

## 六、机器学习是如何工作?

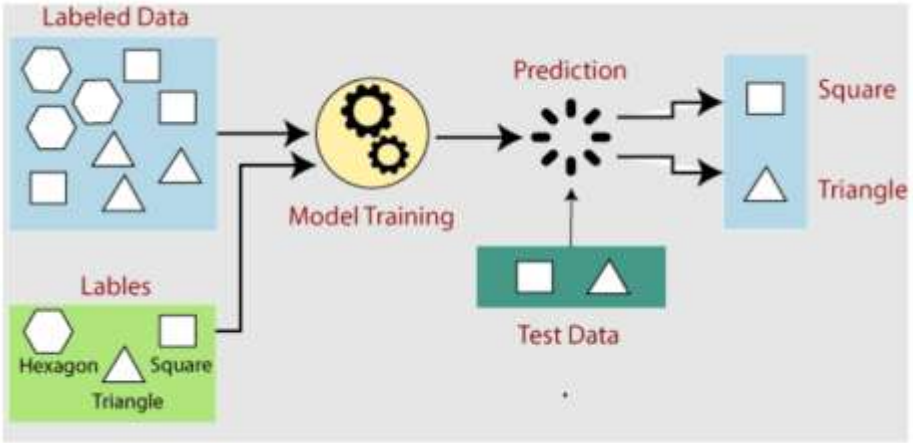
- “机器学习”的基本流程:

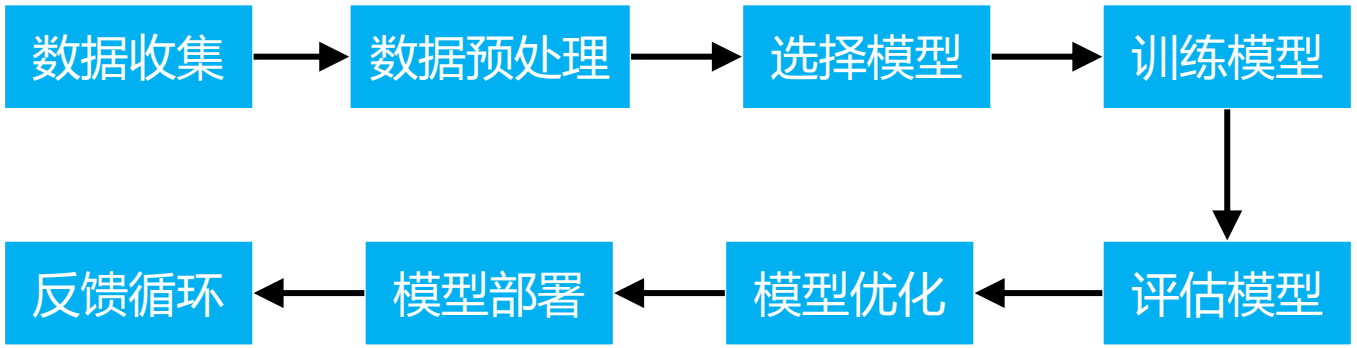
# 机器学习

## 六、机器学习是如何工作?

- “机器学习”的基本流程：




1. **Labeled Data（标记数据）**：：图中蓝色区域显示了标记数据，这些数据包括了不同的几何形状（如六边形、正方形、三角形）。
2. **Model Training（模型训练）**：：在这个阶段，机器学习算法分析数据的特征，并学习如何根据这些特征来预测标签。
3. **Test Data（测试数据）**：：图中深绿色区域显示了测试数据，包括一个正方形和一个三角形。
4. **Prediction（预测）**：：模型使用从训练数据中学到的规则来预测测试数据的标签。在图中，模型预测了测试数据中的正方形和三角形。
5. **Evaluation（评估）**：：预测结果与测试数据的真实标签进行比较，以评估模型的准确性。




**收集数据**：准备包含特征和标签的数据。

**选择模型**：根据任务选择合适的机器学习算法。

**训练模型**：让模型通过数据学习模式，最小化误差。

**评估与验证**：通过测试集评估模型性能，并进行优化。

**部署模型**：将训练好的模型应用到实际场景中进行预测。

**持续改进**：随着新数据的产生，模型需要定期更新和优化。

# 机器学习

## ■ “机器学习”结果评估：

- 评估方法：通过测试来评估机器学习结果。通常，将数据分为训练集(Training set)和测试集(Test set)，训练集用于执行机器学习；测试集用于执行结果评估。

- 数据分割方法：

  ✓ 保留法（Holdout）:取数据S的一部分（通常为2/3）作为训练数据，剩下的部分（通常为1/3）作为测试数据。

  ✓ 交叉验证法（Cross Validation）:把数据S划分为k个不相交的子集，然后取其中一个子集作测试集，剩下数据作训练集，重复k次，把每一个子集都做一次测试集，于是会得到k个测试结果，最终的测试结果就是这k个测试结果的平均值。

  ✓ 随机法:随机抽取S中的一部分数据作为测试数据，把剩下的数据作为训练数据。

# 机器学习

## ■ “机器学习”结果评估指标：

- 误差（Error）：测试数据集T上的误差 --- 回归预测

I. 平均误差(Mean error): $\text{Error} = \frac{1}{n}\sum_{i=1}^{n}(o_i - t_i)$

*II.* 平均绝对值误差 (Mean absolute percentage error): $\text{MAPE}(\%) = \frac{1}{n}\sum_{i=1}^{n}\left(\left|\frac{o_i - t_i}{o_i}\right| \times 100\right)$

*III.* 均方误差(Mean square error)： $\text{MSE} = \frac{1}{n}\sum_{i=1}^{n}(e_i)^2 = \frac{1}{n}\sum_{i=1}^{n}(t_i - o_i)^2$

*IV.* 均方根误差(Root mean square error)： $\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(e_i)^2} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(t_i - o_i)^2}$

*V.* 决定系数 (Coefficient of determination): $R^2 = 1 - \frac{\sum_{i=1}^{n}(t_i - o_i)^2}{\sum_{i=1}^{n}(t_i - \bar{t})^2}$

其中：$t_i$ 是实际输出值； $o_i$ 是神经网络预测输出值

# 机器学习

## ■ “机器学习”结果评估指标：

- 正确率（Accuracy）或错误率（Error Rate）--- 分类问题

✓ 正确率：$$Accuracy(T) = \frac{|T_{Error<\varepsilon}|}{|T|}$$

✓ 错误率：$$ErrorRate(T) = \frac{|T| - |T_{Error<\varepsilon}|}{|T|} = 1 - \frac{|T_{Error<\varepsilon}|}{|T|} = 1 - Accuracy(T)$$

# 机器学习

## ■ “机器学习”结果评估指标：

- 精度（Precision，命中率或准确率）和 召回率（Recall，真阳性率或敏感度）--- 分类问题

✓ 精度：$precision(T) = \frac{|a|}{|a+b|}$

✓ 召回率：$Recall(T) = \frac{|a|}{|a+d|}$

✓ 正确率：$Accuracy(T) = \frac{|a+c|}{|a+b+c+d|}$

a：判定属于类且判定正确；

b：判定属于类且判定错误；

c：判定不属于类且判定正确；

d：判定不属于类且判定错误。

# 机器学习

■ 案例：医学检验结果如下

| 真实结果 | | | |
|---|---|---|---|
| 阳性 | | 阴性 | |
| 10 | | 90 | |
| **检测结果** | | | |
| 真阳性 | 假阳性 | 真阴性 | 假阴性 |
| 8 | 2 | 86 | 4 |

a：判定属于类且判定正确；

b：判定属于类且判定错误；

c：判定不属于类且判定正确；

d：判定不属于类且判定错误。

- 正确率：$Accuracy(T) = \frac{8+86}{100} = 94\%$
- 错误率：$ErrorRate(T) = 1 - Accuracy(T) = 6\%$
- 精度：

$$precision(\text{阳性}) = \frac{|a|}{|a+b|} = \frac{8}{10} = 80\%$$

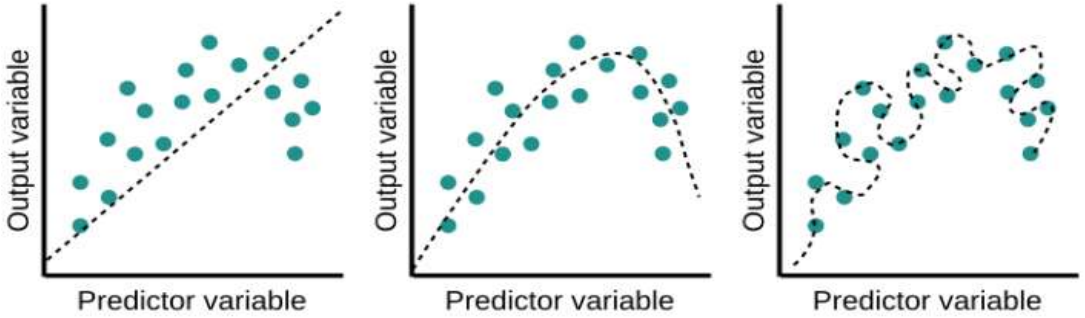
$$precision(\text{阴性}) = \frac{|a|}{|a+b|} = \frac{86}{90} = 95.6\%$$

- 敏感度（真阳性率）：

$$Recall(\text{阳性}) = \frac{|a|}{|a+d|} = \frac{8}{12} = 66.7\%$$

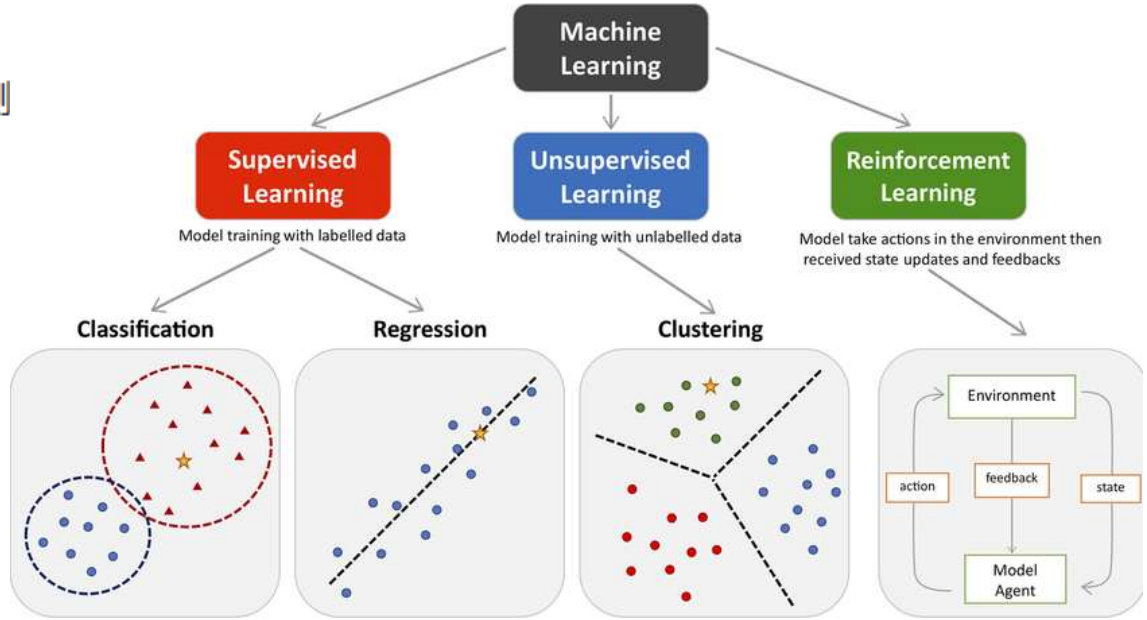
# 机器学习

- 机器学习中的基本概念：




- 欠拟合
- 拟合
- 过拟合

- **训练集、测试集和验证集**：帮助训练、评估和调优模型。
- **特征与标签**：特征是输入，标签是模型预测的目标。
- **模型与算法**：模型是通过算法训练得到的，算法帮助模型学习数据中的模式。
- **监督学习、无监督学习和强化学习**：三种常见的学习方式，分别用于不同的任务。
- **过拟合与欠拟合**：两种常见的问题，影响模型的泛化能力。
- **训练误差与测试误差**：反映模型是否能适应数据，并进行有效预测
- **评估指标**：衡量模型好坏的标准，根据任务选择合适的指标。



# 02

# 机器学习：scikit-learn

# 机器学习：scikit-learn

## 一、scikit-learn 库

- scikit-learn库 (sklearn) 是基于 Python 语言的机器学习工具，对常用的机器学习方法进行封装，提供了强大的数据挖掘和分析工具

- <官网>：https://scikit-learn.org/stable/index.html
- <中文社区>：https://scikit-learn.org.cn/
- <安装> anaconda模式 → conda install sklearn/ scikit-learn
- Scikit-learn包含大量用于机器学习相关的模块

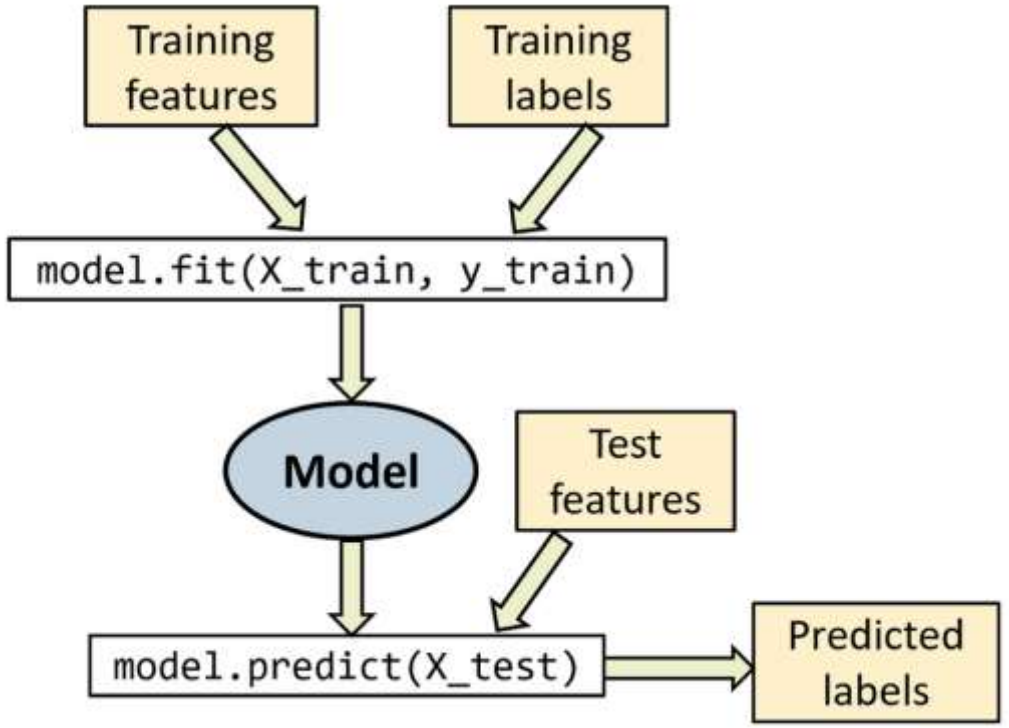

- 分类(Classfication)
- 回归(Regression)
- 聚类(Clustering)
- 降维(Dimensionality Reduction)
- 模型选择(Model selection)
- 预处理(Preprocessing)

# 机器学习：scikit-learn

## 二、scikit-learn 如何工作?

- 在机器学习scikit-learn库中，机器学习的流程遵循一定的模式：**数据加载、数据预处理、训练模型、评估模型和调优模型。**

- **数据加载**：使用 Sklearn 或其他库加载数据集，或使用 train_test_split() 分割数据。
- **数据预处理**：根据数据的类型，可能需要进行标准化、去噪、缺失值填充等操作。
- **选择算法和训练模型**：选择适合的算法（如逻辑回归、支持向量机等），使用 .fit() 方法对模型进行训练。
- **模型评估**：使用交叉验证或单一训练/测试集来评估模型的准确性、召回率、F1分数等性能指标。
- **模型优化**：使用网格搜索或随机搜索对模型进行超参数优化，提高模型性能。



# 机器学习：scikit-learn

## 三、 scikit-learn 中常见的模块和类

- **分类（Classification）**
  - sklearn.linear_model.LogisticRegression：逻辑回归
  - sklearn.svm.SVC：支持向量机分类
  - sklearn.neighbors.KNeighborsClassifier：K近邻分类
  - sklearn.ensemble.RandomForestClassifier：随机森林分类
- **回归（Regression）**
  - sklearn.linear_model.LinearRegression：线性回归
  - sklearn.linear_model.Ridge：岭回归
  - sklearn.ensemble.RandomForestRegressor：随机森林回归
- **聚类（Clustering）**
  - sklearn.cluster.KMeans：K均值聚类
  - sklearn.cluster.DBSCAN：基于密度的空间聚类
- **模型选择（Model Selection）**
  - sklearn.model_selection.train_test_split：将数据集划分为训练集和测试集
  - sklearn.model_selection.GridSearchCV：网格搜索，寻找最佳超参数
- **数据预处理（Preprocessing）**
  - sklearn.preprocessing.StandardScaler：标准化
  - sklearn.preprocessing.MinMaxScaler：最小-最大标准化

# 机器学习：scikit-learn

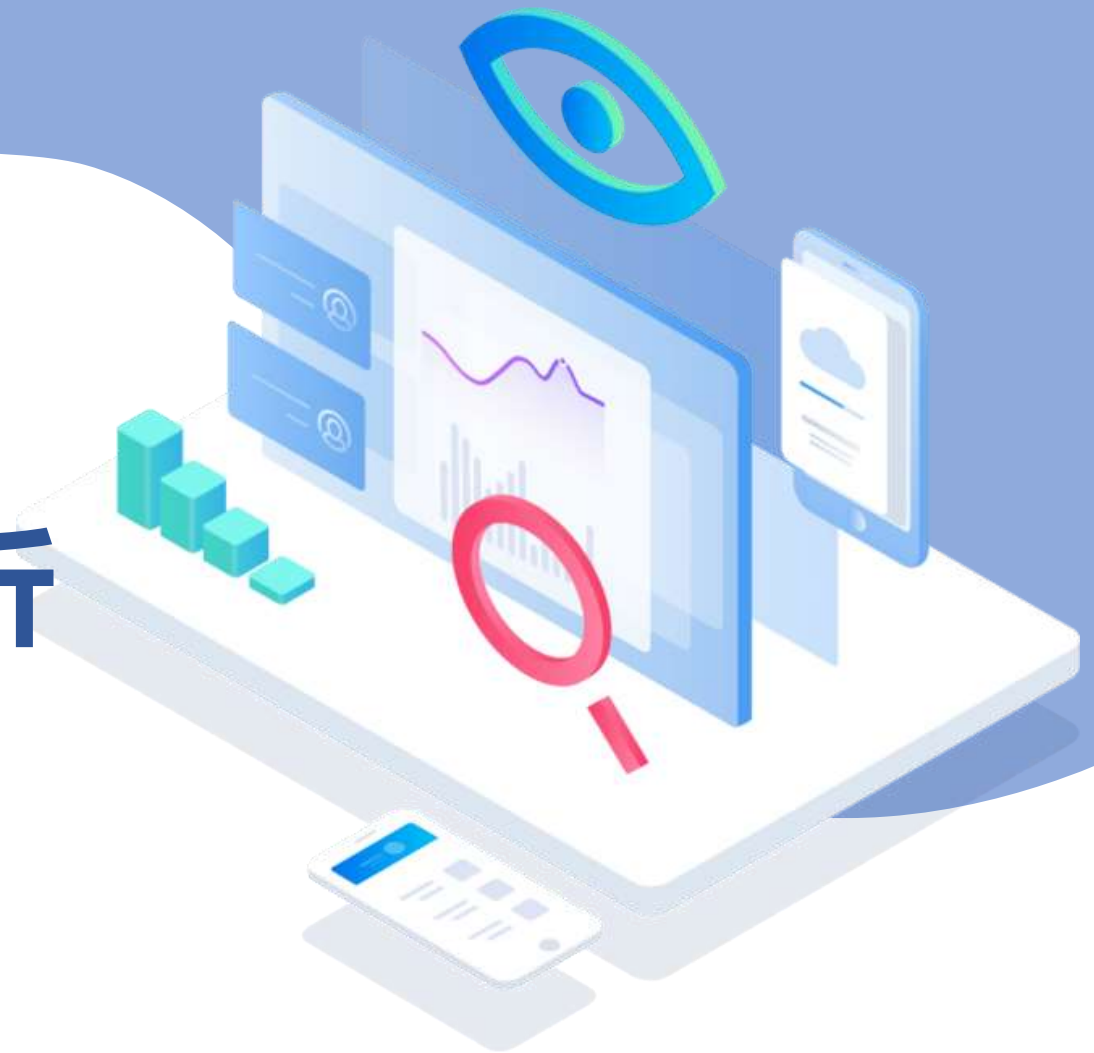
## 四、 scikit-learn 中常见的数据集

| 类型 | 数据集名称 | 调用方法 | 适用算法 | 数据规模 |
| --- | --- | --- | --- | --- |
| 小数据集 | 波士顿房价数据集 | load_boston() | 回归 | 506*13 |
| | 鸢尾花数据集 | load_iris() | 分类 | 150*4 |
| | 糖尿病数据集 | load_diabetes() | 回归 | 442*10 |
| | 手写数字数据集 | load_digits() | 分类 | 1797*64 |
| 大数据集 | Olivetti脸部图像数据集 | fetch_olivetti_faces() | 降维 | 400*64*64 |
| | 新闻分类数据集 | fetch_20newsgroups() | 分类 | |
| | 带标签的人脸数据集 | fetch_lfw_people() | 分类，降维 | |
| | 路透社新闻数据集 | fetch_rcv1() | 分类 | 804414*47236 |

小规模数据集可以使用datasets.load_*函数获取。例如，使用sklearn.datasets.load_iris()加载鸢尾花数据集，利用分类算法对Iris数据集进行分类。

# 03

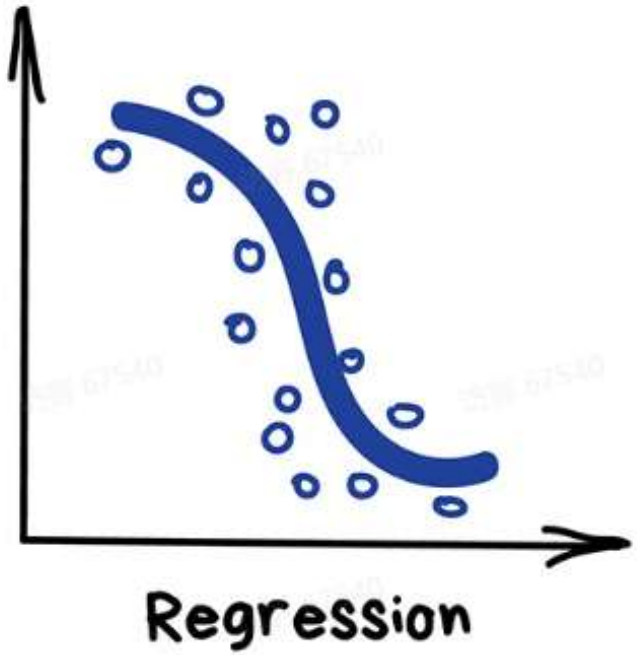
# scikit-learn：回归分析



# scikit-learn：回归分析

## 一、 回归分析

- 回归分析（ regression analysis ）：研究自变量与因变量之间的关系的一种统计分析方法，用于预测一组数据的连续输出变量。**研究的是因变量（目标）和自变量（预测器）之间的关系。** 。



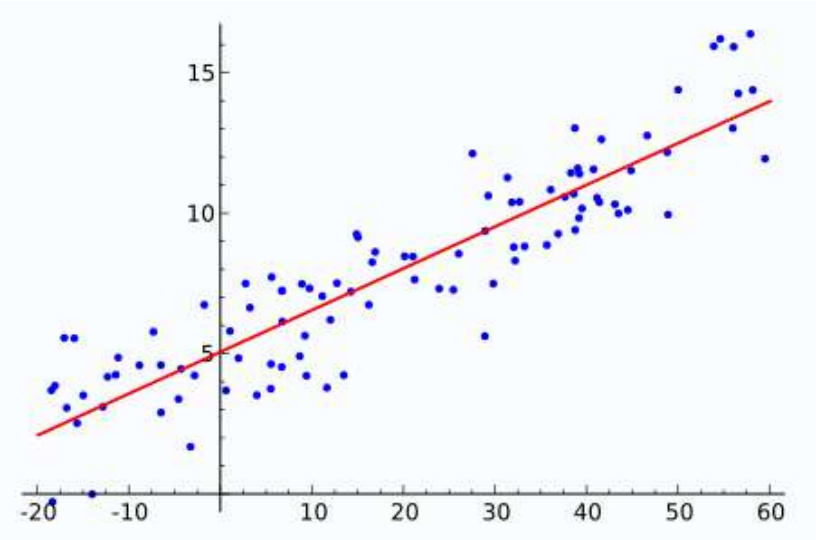

| 回归模型 | 加载模块 |
| --- | --- |
| 线性回归 | linear_model.LinearRegression |
| 岭回归 | linear_model.Ridge |
| Lasso回归 | linear_model.Lasso |
| 弹性网络 | linear_model.ElasticNet |
| 最小角回归 | linear_model.Lars |
| 贝叶斯回归 | linear_model.BayesianRidge |

# scikit-learn：回归分析

## （1） 线性回归 （Line Regression）

- 线性回归通过拟合一条直线来预测目标变量。其核心假设是特征与目标变量之间存在线性关系。
  - 一元线性回归模型如下：$y = kx + b$
  - 多元线性回归模型如下：$y = w_1 x_1 + w_2 x_2 + \cdots w_n x_n + b$

- y 是预测值（目标值）。
- $x_1$，$x_2$，$x_n$ 是输入特征。
- $w_1$，$w_2$，$w_n$是待学习的权重（模型参数）。
- b 是偏置项。



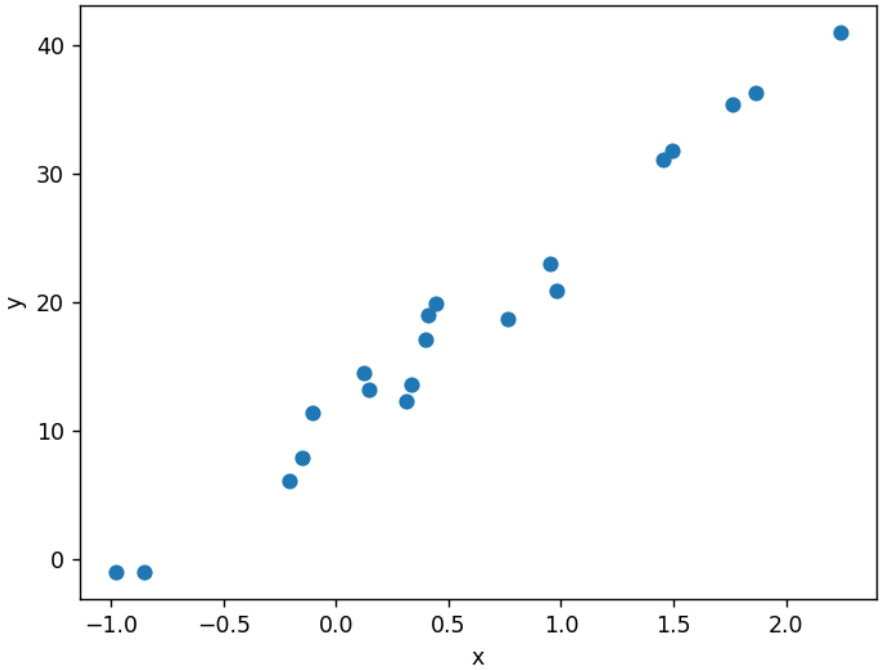
# scikit-learn：回归分析

■ 案例1： make_regression函数，**生成回归样本数据**

x , y = sklearn. datasets. make_regression (n_samples, n_features, n_informative, n_targets, bias , effective_rank, tail_strength, noise, shuffle , coef , random_state )

**参数说明：**

- n_samples：样本数
- n_features：特征数(自变量个数)
- n_informative：参与建模特征数
- n_targets：因变量个数
- noise：噪音
- bias：偏差(截距)
- coef：是否输出coef标识
- random_state：随机状态若为固定值则每次产生的数据都一样



# scikit-learn：回归分析

- train_test_split函数：划分训练集和测试集

X_train， X_test， y_train， y_test = sklearn.model_selection. train_test_split (train_data, train_target, test_size, random_state, shuffle)

| 变量 | 描述 |
| --- | --- |
| X_train | 划分的训练集数据 |
| X_test | 划分的测试集数据 |
| y_train | 划分的训练集标签 |
| y_test | 划分的测试集标签 |

| 参数 | 描述 |
| --- | --- |
| train_data | 还未划分的数据集 |
| train_target | 还未划分的标签 |
| test_size | 分割比例，默认为0.25，即测试集占完整数据集的比例 |
| random_state | 随机数种子，应用于分割前对数据的洗牌。可以是int，RandomState实例或None，默认值=None。设成定值意味着，对于同一个数据集，只有第一次运行是随机的，随后多次分割只要rondom_state相同，则划分结果也相同。 |
| shuffle | 是否在分割前对完整数据进行洗牌（打乱），默认为True，打乱 |

# scikit-learn：回归分析

- Scikit-learn 中常用方法

| 方法名称 | 方法功能 |
| --- | --- |
| fit() | 使用数据样本X和目标y对模型进行拟合，即训练模型。 |
| predict() | 模型训练完成后，使用此方法对未知样本进行预测。 |

- Scikit-learn 中评价函数

| 方法名称 | 方法功能 |
| --- | --- |
| mean_squared_error | 均方误差 MSE |
| r2_score | 决定系数 R2 |

# scikit-learn：回归分析

## ■ Sklearn 模型保存与加载

- joblib 是一个高效的 Python 序列化工具，特别适合用于保存包含大量数值数组（如 numpy 数组、scikit-learn 模型等）的对象，joblib 在处理大规模数据时更高效。

✓ 使用 joblib.dump() 方法将模型保存

```
joblib.dump(model, 'model.joblib')
```

✓ 使用 joblib.load() 方法加载保存的模型对象

```
loaded_model = joblib.load('model.joblib')
```

# scikit-learn：回归分析

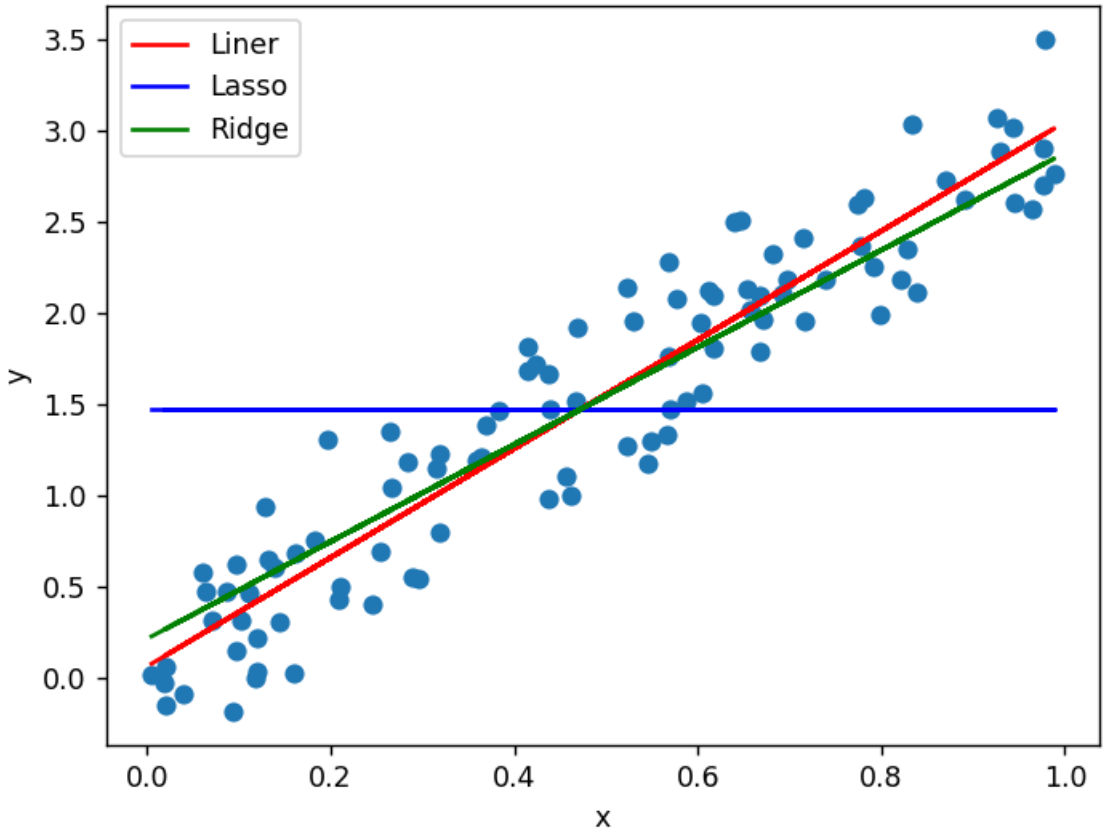
- 线性回归： 岭回归 vs Lasso 回归

  - Lasso 回归（Lasso Regression）：是线性回归的一种形式，它使用 **L1 正则化** 来对回归系数进行惩罚

$$L_{\mathrm{L1}} = L_{\mathrm{data}} + \lambda \sum_{i=1}^{n} |w_i|$$

  - 岭回归（Ridge Regression）：是线性回归的一个变种，使用 **L2 正则化** 来约束模型的复杂度，避免过拟合

$$L_{\mathrm{L2}} = L_{\mathrm{data}} + \lambda ||\mathbf{w}||_2^2$$



# scikit-learn：回归分析

（2） 非线性回归 （Nonline Regression）

- 多项式回归（Polynimial Regression）：多项式回归是一种用于拟合非线性数据的回归方法

$$y = w_0 x_0 + w_1 x + w_2 x^2 \cdots + w_n x^n$$

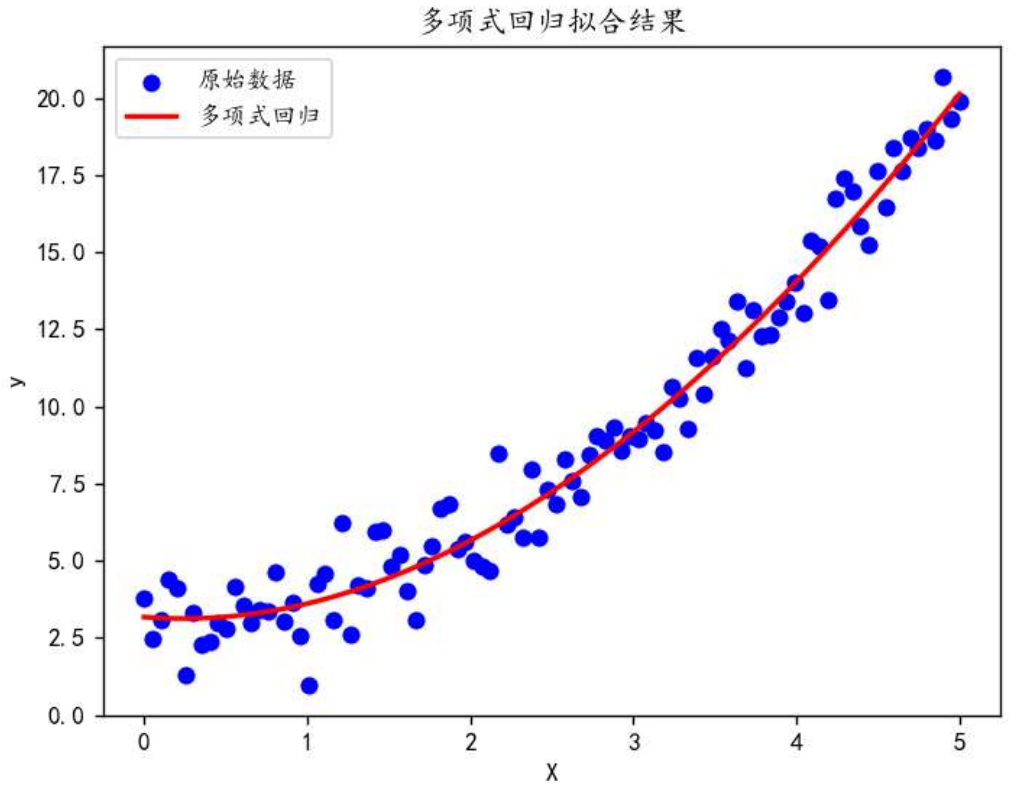
sklearn.preprocessing.PolynomialFeatures(degree, *, interaction_only, include_bias=True)

| 参数 | 含义 |
| --- | --- |
| degree | 多项式中的次数，默认为2 |
| interaction_only | 布尔值是否只产生交互项，默认为False |
| include_bias | 布尔值，是否产出与截距项相乘的$x_0$，默认True |

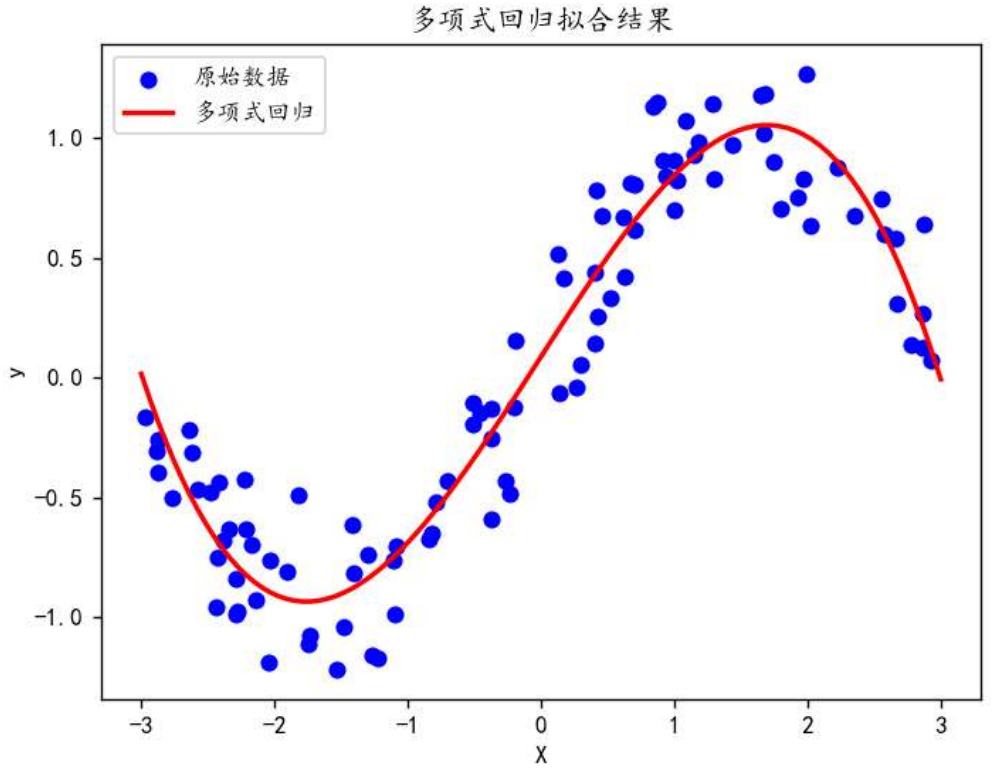
# scikit-learn：回归分析

（2） 非线性回归 （Nonline Regression）

案例： $y = 2 + x + 0.5x^2 + \theta$




案例： $y = \sin(x) + 0.2\theta$

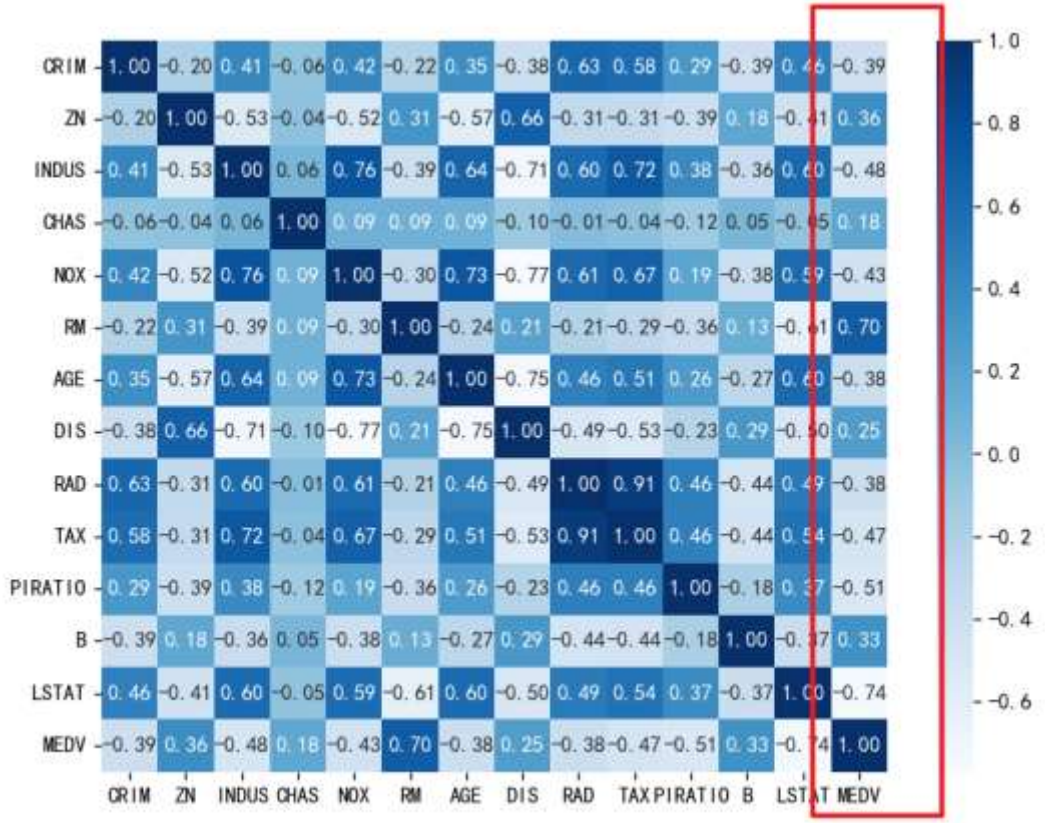


# scikit-learn：回归分析

## ■ 案例2： Boston波士顿房价预测

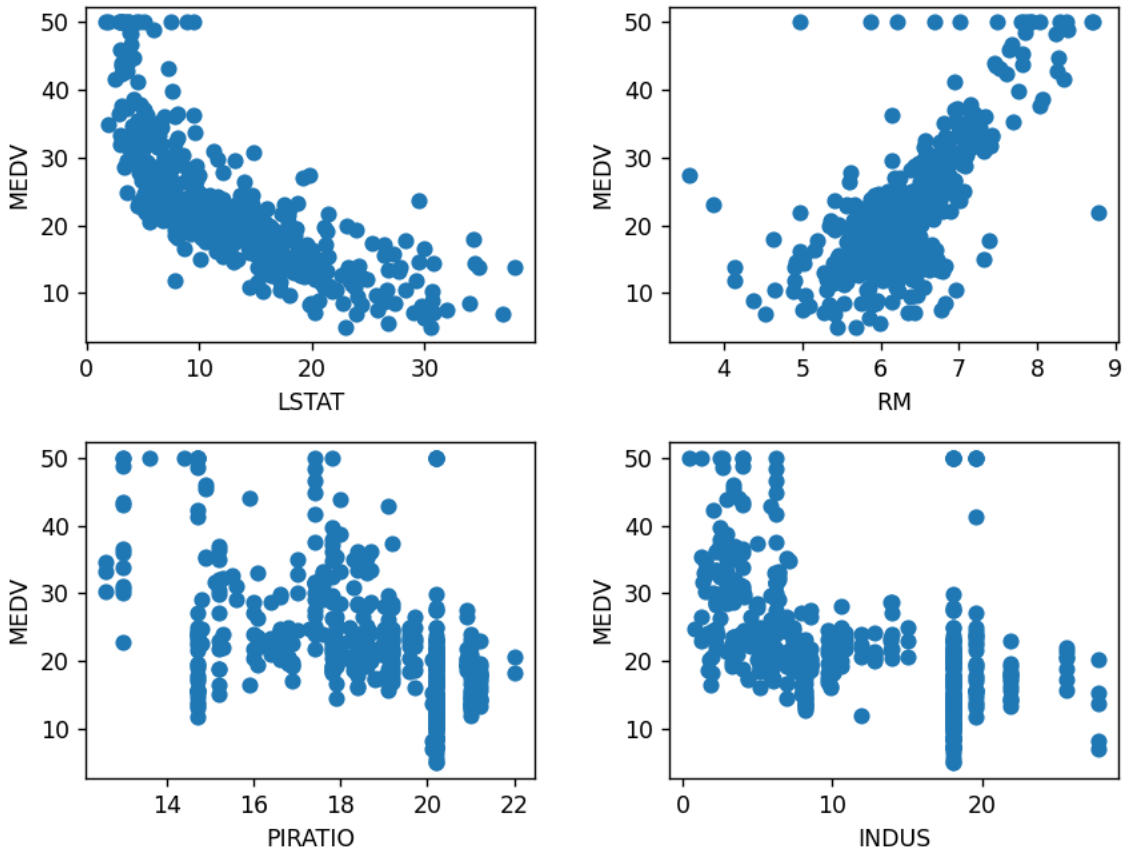
- 该数据集包含美国人口普查局收集的**美国马萨诸塞州波士顿住房价格**的有关信息, 数据集有506个案例。数据集有14个属性:

- CRIM--城镇人均犯罪率 ------【城镇人均犯罪率】
- ZN - 占地面积超过25,000平方英尺的住宅用地比例。 ------【住宅用地所占比例】
- INDUS - 每个城镇非零售业务的比例。 ------【城镇中非商业用地占比例】
- CHAS - Charles River虚拟变量（如果是河道，则为1;否则为0 ------【查尔斯河虚拟变量，用于回归分析】
- NOX - 一氧化氮浓度（每千万份） ------【环保指标】
- RM - 每间住宅的平均房间数 ------【每栋住宅房间数】
- AGE - 1940年以前建造的自住单位比例 ------【1940年以前建造的自住单位比例 】
- DIS -波士顿的五个就业中心加权距离 ------【与波士顿的五个就业中心加权距离】
- RAD - 径向高速公路的可达性指数 ------【距离高速公路的便利指数】
- TAX - 每10,000美元的全额物业税率 ------【每一万美元的不动产税率】
- PTRATIO - 城镇的学生与教师比例 ------【城镇中教师学生比例】
- B - 1000（Bk - 0.63）^ 2其中Bk是城镇黑人的比例 ------【城镇中黑人比例】
- LSTAT - 人口状况下降% ------【房东属于低等收入阶层比例】
- MEDV - 自有住房的中位数报价, 单位1000美元 ------【自住房屋房价中位数】

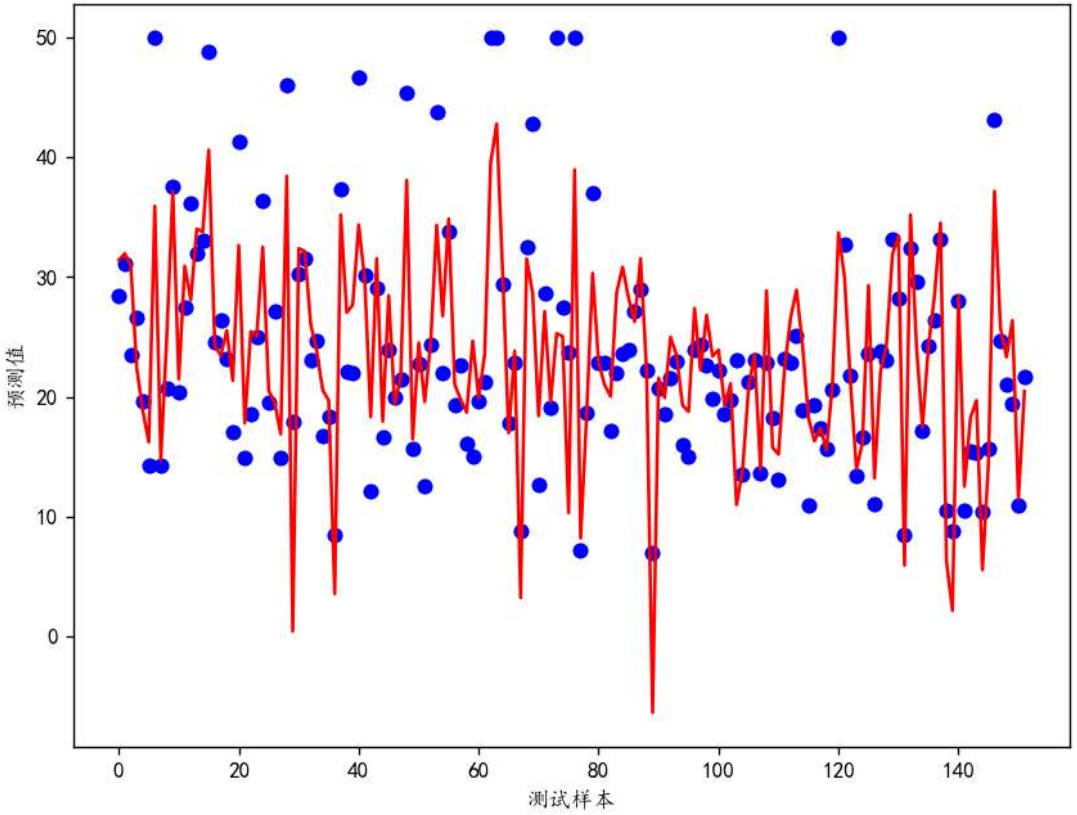


# scikit-learn：回归分析

- 相关性分析



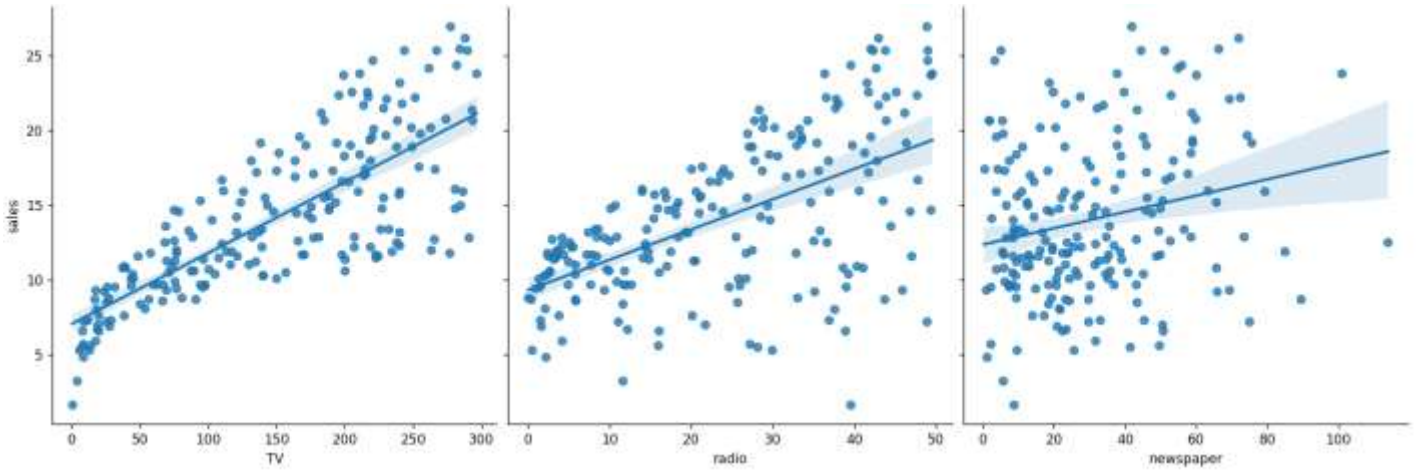

- 真实 vs 预测



# scikit-learn：回归分析

## ■ 案例3：广告费分析

- TV：在电视上投资的广告费用（以千万元为单位）；
- Radio：在广播媒体上投资的广告费用；
- Newspaper：用于报纸媒体的广告费用；
- Sales：对应产品的销量




| | TV | Radio | Newspaper | Sales |
|---|---|---|---|---|
| **1** | 230.1 | 37.8 | 69.2 | 22.1 |
| **2** | 44.5 | 39.3 | 45.1 | 10.4 |
| **3** | 17.2 | 45.9 | 69.3 | 9.3 |
| **4** | 151.5 | 41.3 | 58.5 | 18.5 |
| **5** | 180.8 | 10.8 | 58.4 | 12.9 |
| **...** | ... | ... | ... | ... |
| **196** | 38.2 | 3.7 | 13.8 | 7.6 |
| **197** | 94.2 | 4.9 | 8.1 | 9.7 |
| **198** | 177.0 | 9.3 | 6.4 | 12.8 |
| **199** | 283.6 | 42.0 | 66.2 | 25.5 |
| **200** | 232.1 | 8.6 | 8.7 | 13.4 |

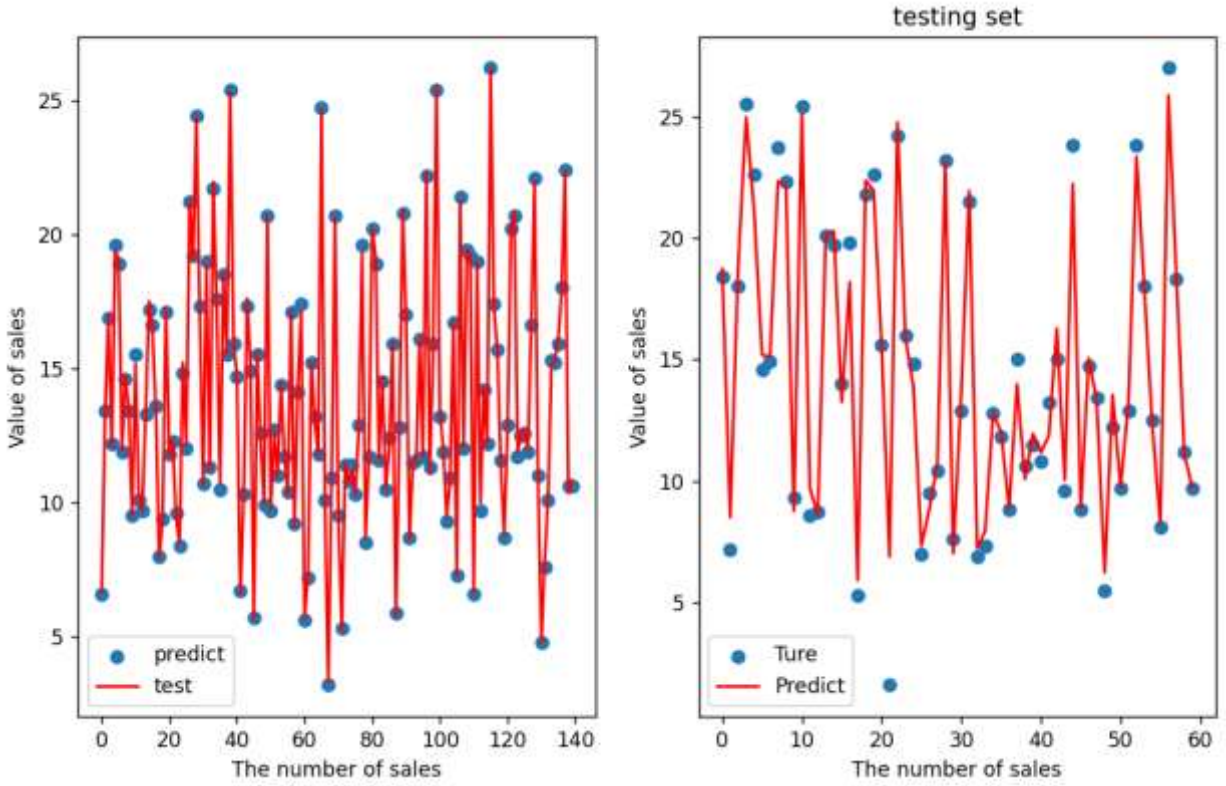
# scikit-learn：回归分析

■ 回归模型比较

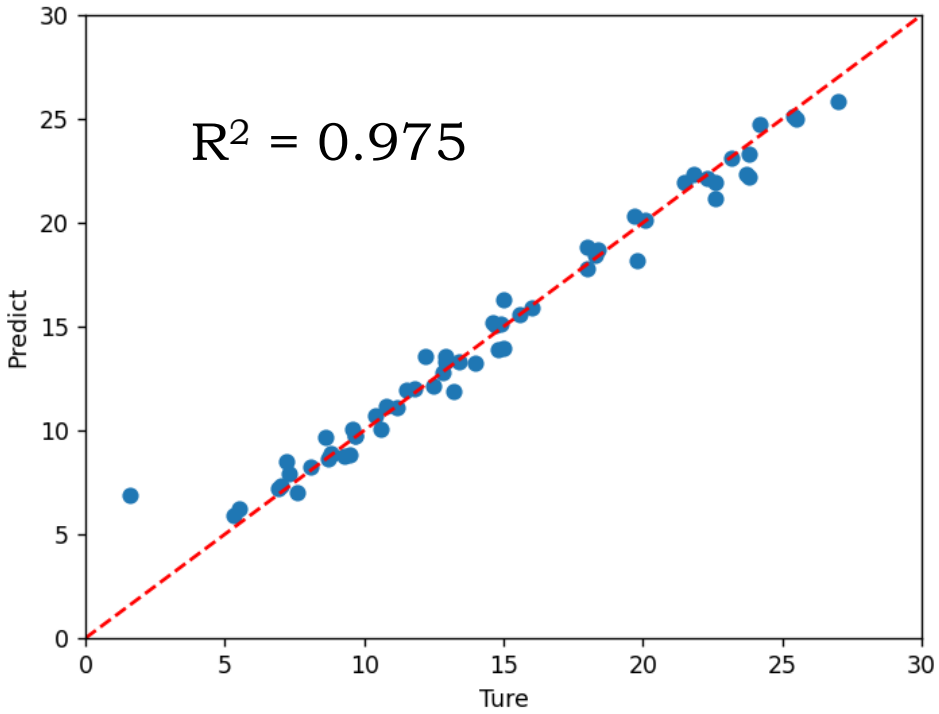
| 序号 | 回归模型 | 平均绝对值误差MAE | 均方误差MSE | 决定系数$R^2$ |
|---|---|---|---|---|
| **1** | **线性模型line** | 1.709 | 5.374 | 0.857 |
| **2** | **决策树tree** | 0.937 | 1.281 | 0.966 |
| **3** | **支持向量机SVR** | 2.008 | 7.741 | 0.794 |
| **4** | **K近邻 KNN** | 1.300 | 3.191 | 0.915 |
| **5** | **Adaboost** | 0.924 | 1.397 | 0.963 |
| **6** | **Boosting** | **0.621** | **0.944** | **0.975** |

# scikit-learn：回归分析

■ 回归模型：**Boosting**



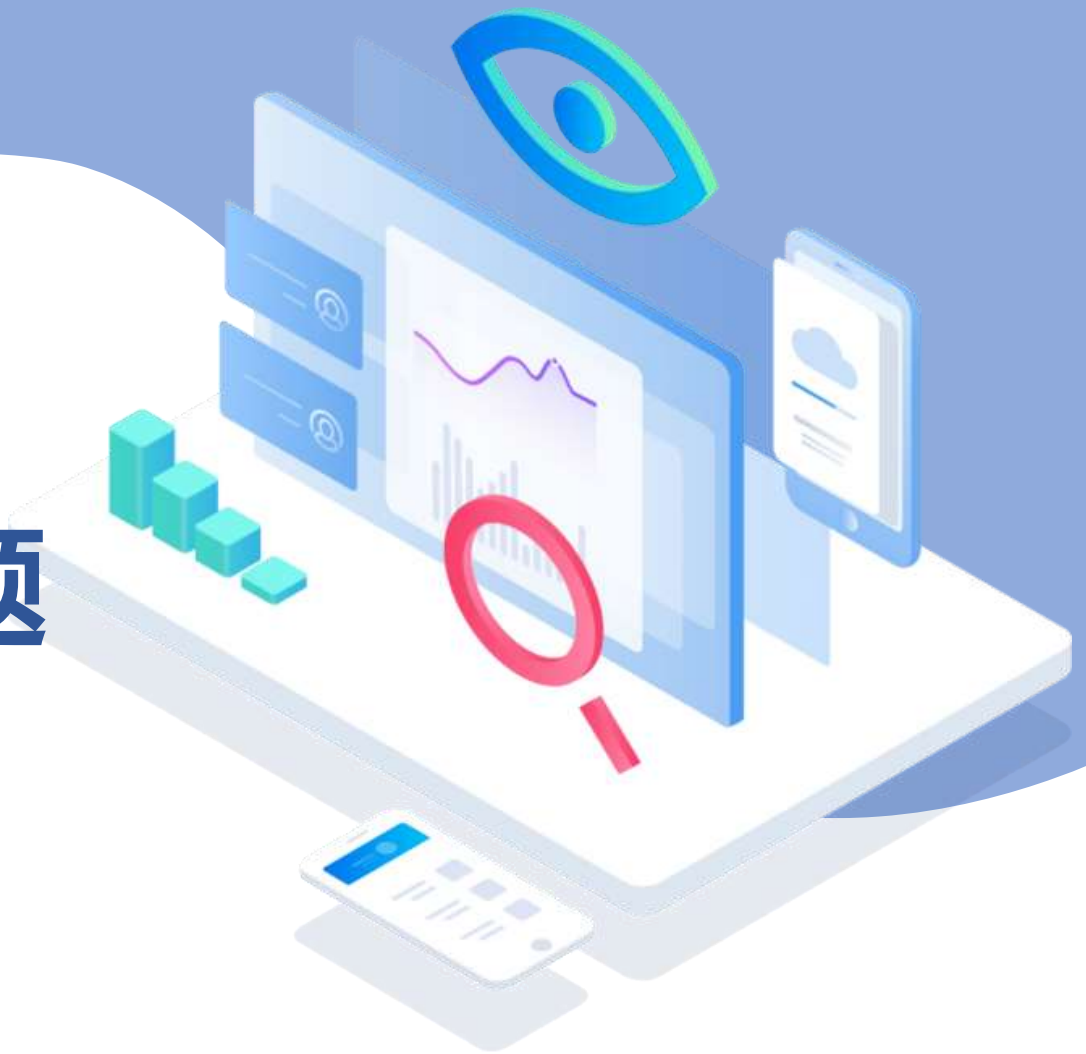

- 真实 vs 预测




- 真实 vs 预测

# 04

# scikit-learn：分类问题



# scikit-learn：分类问题

## 一、 分类模型

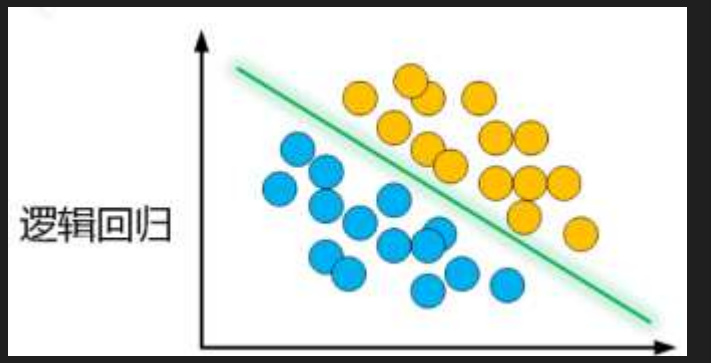
- **逻辑回归**（Logistic Regression）是一种广泛应用于二分类问题的统计学习方法。

逻辑回归的目标是预测一个二分类结果 $y \in \{0, 1\}$，它通过如下的公式建立模型：

$$p(y = 1|X) = \sigma(w^T X + b)$$

其中：

- $X$ 是输入特征（可以是多个特征组成的向量）。
- $w$ 是权重向量。
- $b$ 是偏置项。
- $\sigma(z) = \frac{1}{1+e^{-z}}$ 是Sigmoid函数。

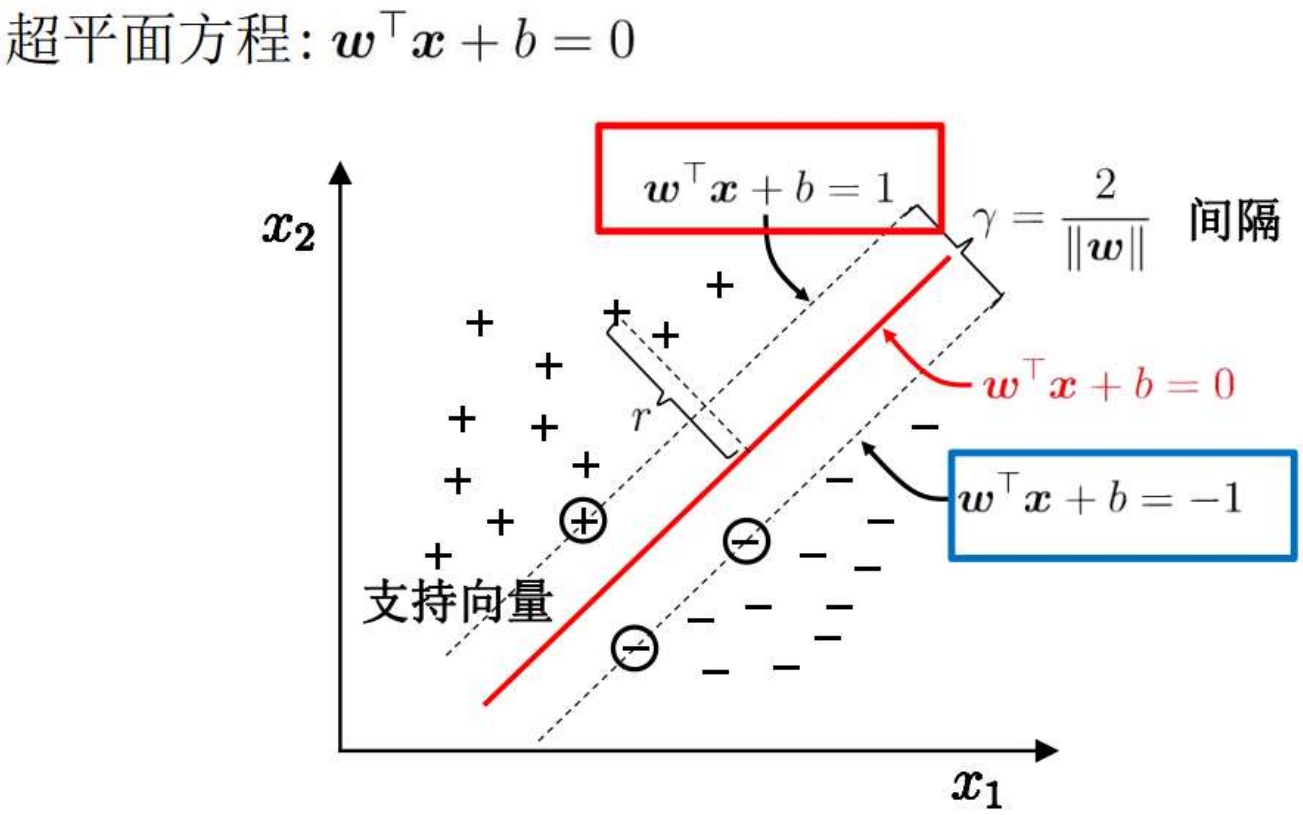



Sigmoid函数将模型的输出值（$w^T X + b$）映射到0到1之间，因此它可以看作是属于类别1的概率。

# scikit-learn：分类问题

## 一、 分类模型

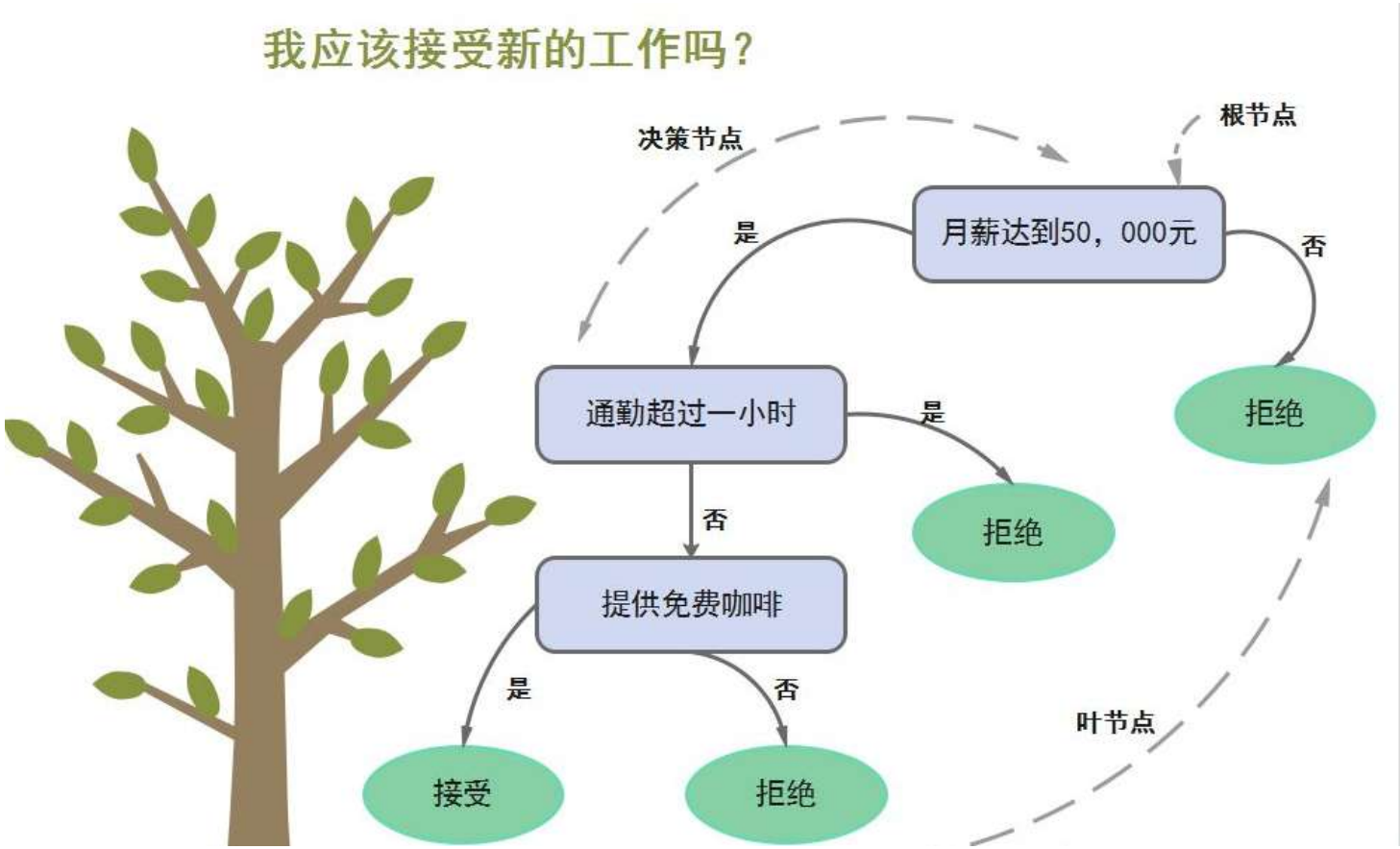
- **支持向量机**（support vector machines，SVM）是一种二分类模型，它的目的是寻找一个超平面来对样本进行分割。



# scikit-learn：分类问题

## 一、 分类模型

- **决策树**（Decision Tree），它是一种以树形数据结构来展示决策规则和分类结果的模型

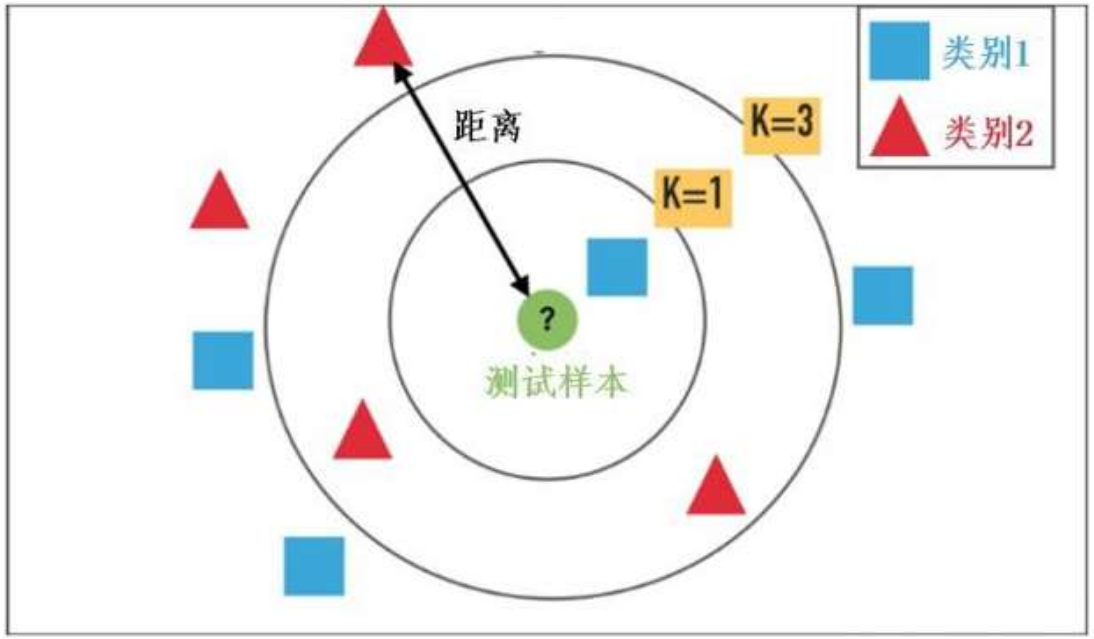


# scikit-learn：分类问题

## 一、 分类模型

- **K邻近算法**（k-Nearest Neighbor，KNN），算法是机器学习算法中最基础、最简单的算法之一。KNN通过测量不同特征值之间的距离来进行分类。

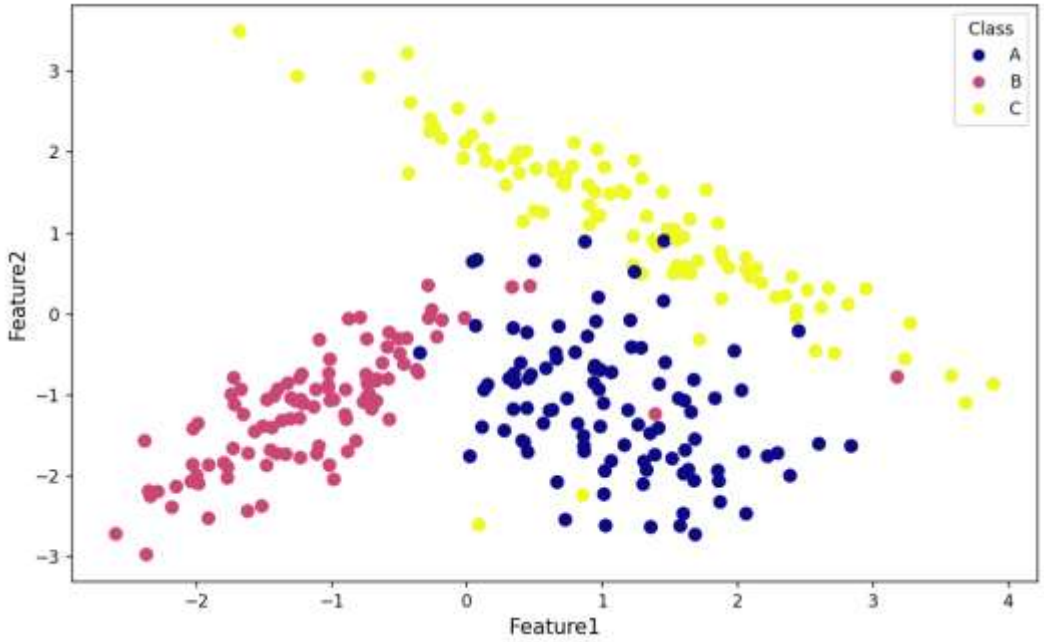
- 训练数据集中找到与该实例最近邻的K个实例，依据“少数服从多数”的原则，根据这K个实例中占多数的类，就把该实例分为这个类。



# scikit-learn：分类问题

■ **案例1**：make_classification() 函数，**生成分类样本数据**

```
make_classification（
    n_samples=100,
    n_features=20,
    n_informative=2,
    n_redundant=0,
    n_repeated=0,
    n_classes=2,
    n_clusters_per_class=1,
    weights=None,
    shuffle=True,
    random_state=None,
)
```




n_samples：样本个数

n_features：样本特征个数，包括信息特征，冗余特征，重复特征，和除去以上特征的无用特征

n_informative：信息特征个数

n_redundant：冗余特征个数

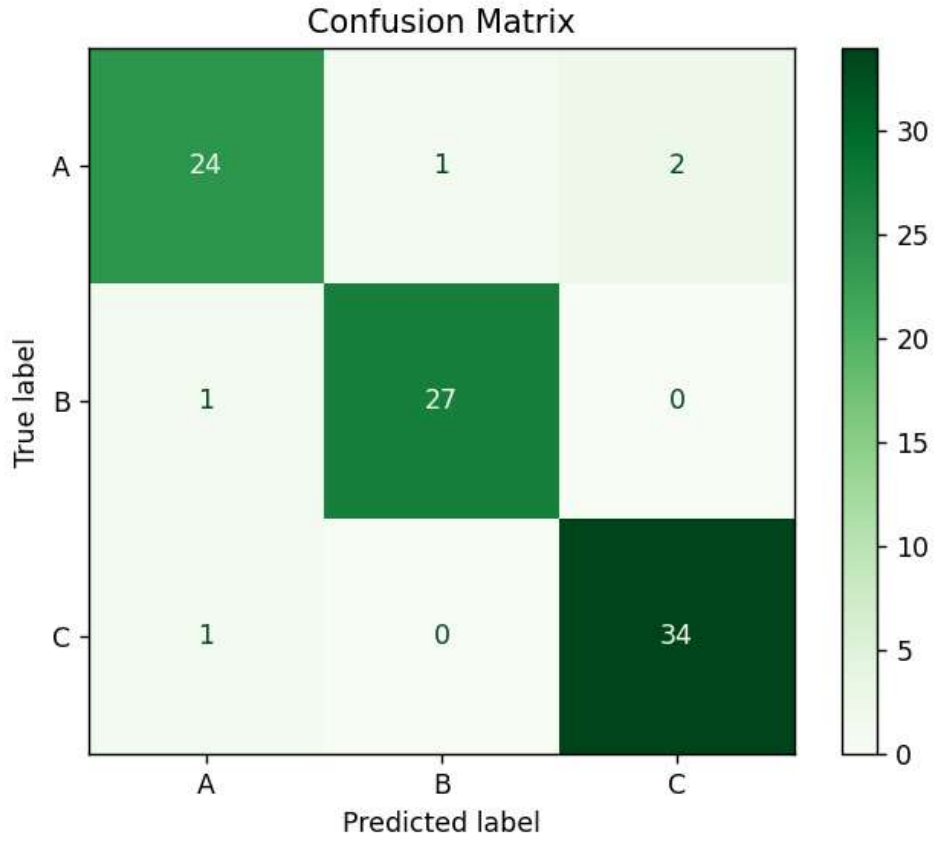
n_repeated：重复特征个数

n_classes：样本类别个数

n_clusters_per_class：每个类的簇的个数

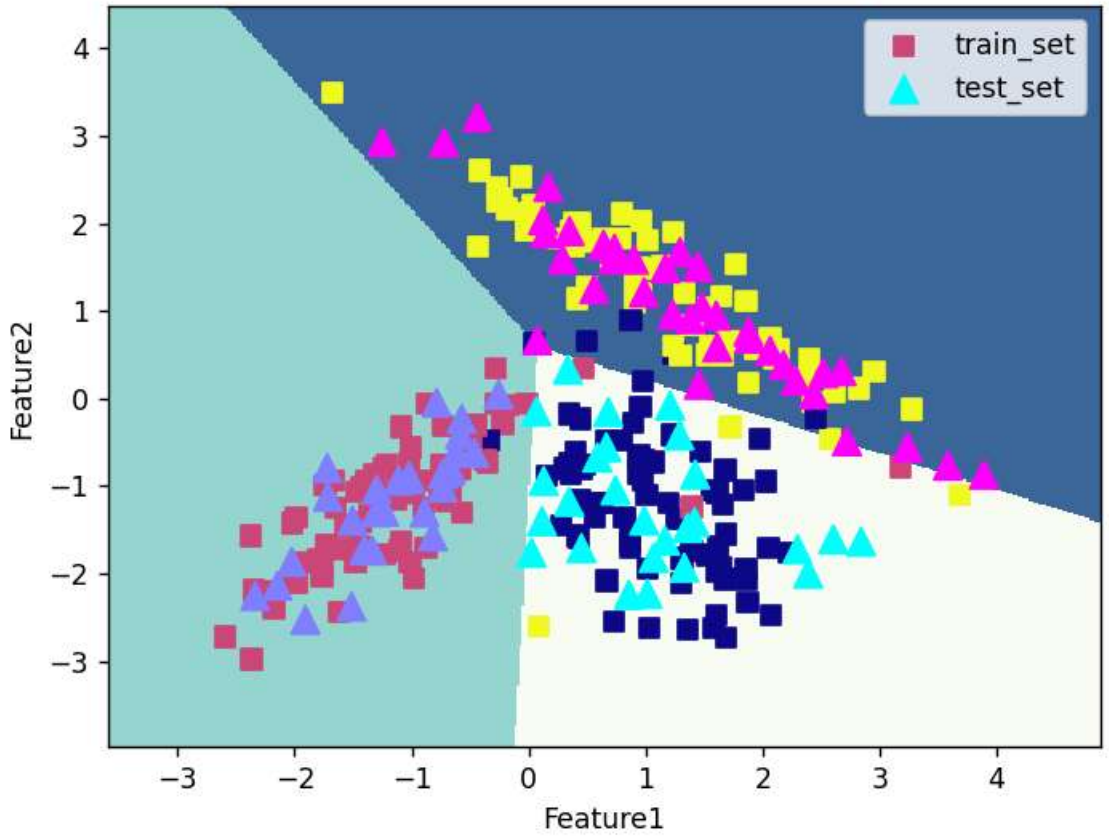
# scikit-learn：分类问题

- 混淆矩阵




测试集的预测精度 = 94.44%

- 分类数据可视化



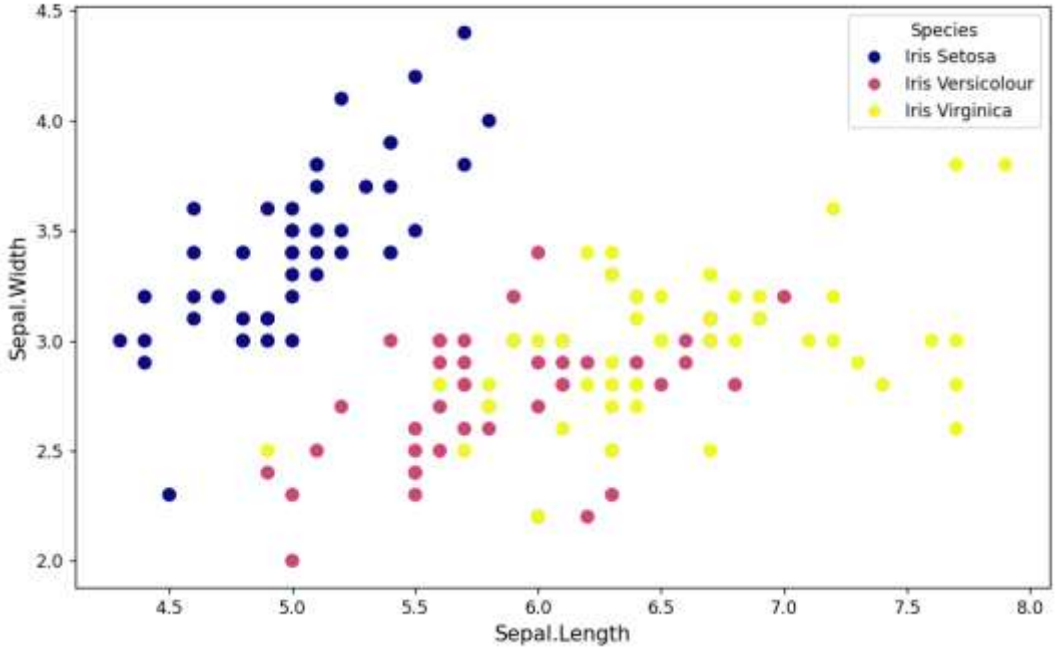
# scikit-learn：分类问题

■ **案例2**：鸢尾花数据集（iris）

数据集包含了150个鸢尾花样本，每个样本有**4个特征**（花萼长度(Sepal.Length)、花萼宽度(Sepal.Width)、花瓣长度(Petal.Length)和花瓣宽度(Petal.Width)）和**1个目标变量**（鸢尾花的品种:山鸢尾(Setosa)、变色鸢尾(Versicolour)、维吉尼亚鸢尾(Virginica)）

费雪鸢尾花卉数据集

| 花萼长度 | 花萼宽度 | 花瓣长度 | 花瓣宽度 | 属种 |
|---|---|---|---|---|
| 5.1 | 3.5 | 1.4 | 0.2 | *setosa* |
| 4.9 | 3.0 | 1.4 | 0.2 | *setosa* |
| 4.7 | 3.2 | 1.3 | 0.2 | *setosa* |
| 4.6 | 3.1 | 1.5 | 0.2 | *setosa* |
| 5.0 | 3.6 | 1.4 | 0.2 | *setosa* |
| 5.4 | 3.9 | 1.7 | 0.4 | *setosa* |
| 4.6 | 3.4 | 1.4 | 0.3 | *setosa* |



# scikit-learn：分类问题

## ■ 鸢尾花数据集（iris）--- 分类问题建模

- **数据准备**：获取并准备好数据集。在鸢尾花数据集中，这包括加载数据，了解数据的结构，检查是否有缺失值等；
- **数据预处理**：对数据进行清洗和转换，如处理缺失值、标准化或归一化特征等；
- **数据分割**：将数据分成训练集和测试集。训练集用于训练模型，测试集用于评估模型的性能；
- **选择模型**：选择合适的机器学习算法来训练模型，如决策树、逻辑回归、支持向量机（SVM）等；
- **训练模型**：使用训练数据来训练模型；
- **评估模型**：使用测试数据评估模型的性能，查看模型的准确率、混淆矩阵等评估指标；
- **模型优化**：根据评估结果调整模型参数，优化模型性能；
- **模型部署**：将训练好的模型应用到实际问题中进行预测。

# scikit-learn：分类问题

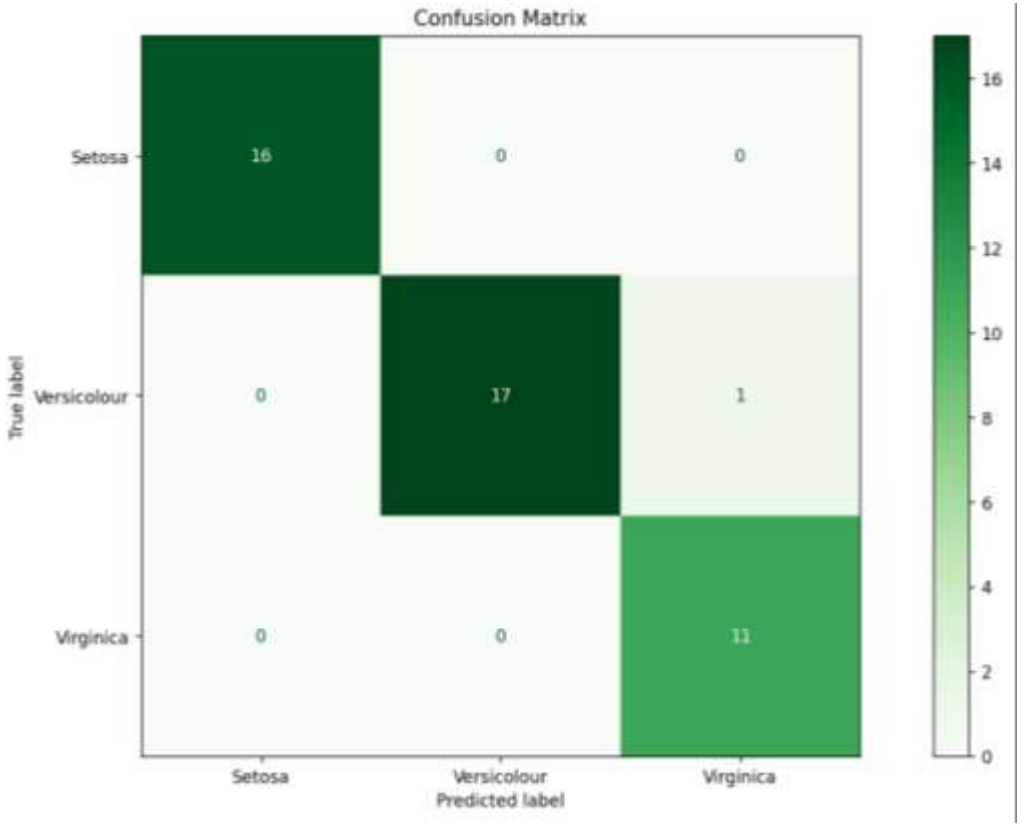
■ 鸢尾花数据集（iris）--- 分类问题建模

- 测试集的预测精度 = 97.78%

分类报告：

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 0 | 1.00 | 1.00 | 1.00 | 16 |
| 1 | 1.00 | 0.94 | 0.97 | 18 |
| 2 | 0.92 | 1.00 | 0.96 | 11 |
| accuracy | | | 0.98 | 45 |
| macro avg | 0.97 | 0.98 | 0.98 | 45 |
| weighted avg | 0.98 | 0.98 | 0.98 | 45 |

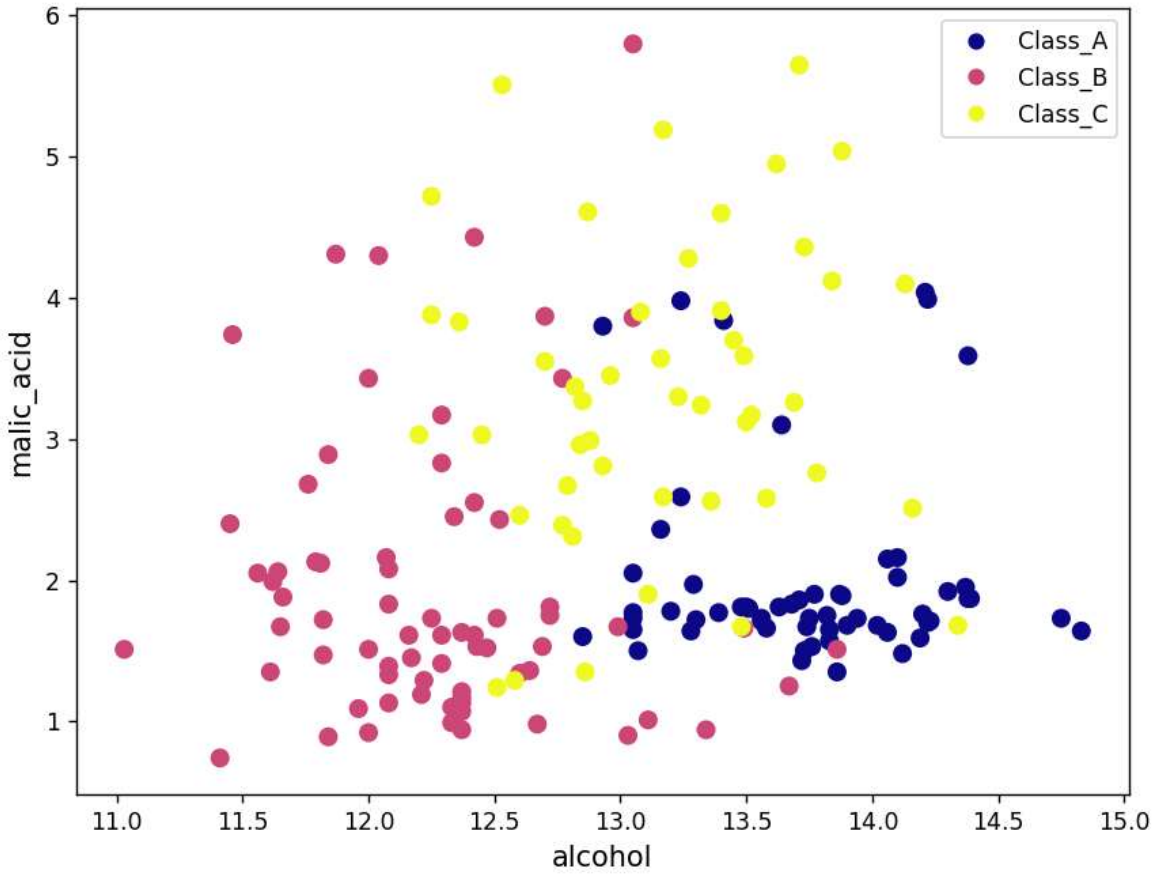
- 混淆矩阵



# scikit-learn：分类问题

## ■ 案例3：葡萄酒品种分类

- wine数据集中，包括三种酒(A、B和C)和13种不同成分的数量，共有178个样本。13个属性是，酒精、苹果酸、灰、灰分的碱度、镁、总酚、黄酮类化合物、非黄烷类酚类、原花色素、颜色强度、色调、稀释葡萄酒的OD280/OD315、脯氨酸



# scikit-learn：分类问题

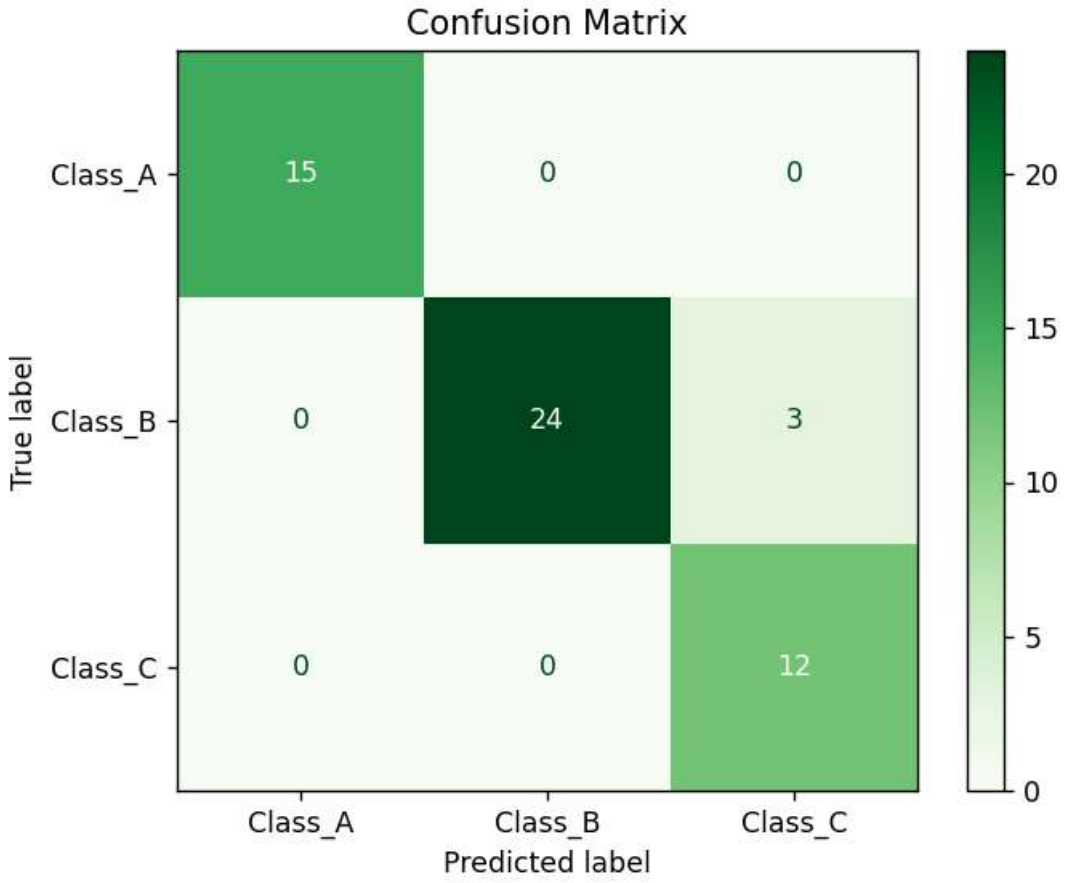
## ■ 案例3：葡萄酒品种分类

- 测试集的预测精度 = 94.44%

分类报告：

| | precision | recall | f1-score | support |
|---|---|---|---|---|
| 1 | 1.00 | 1.00 | 1.00 | 15 |
| 2 | 1.00 | 0.89 | 0.94 | 27 |
| 3 | 0.80 | 1.00 | 0.89 | 12 |
| accuracy | | | 0.94 | 54 |
| macro avg | 0.93 | 0.96 | 0.94 | 54 |
| weighted avg | 0.96 | 0.94 | 0.95 | 54 |

- 混淆矩阵

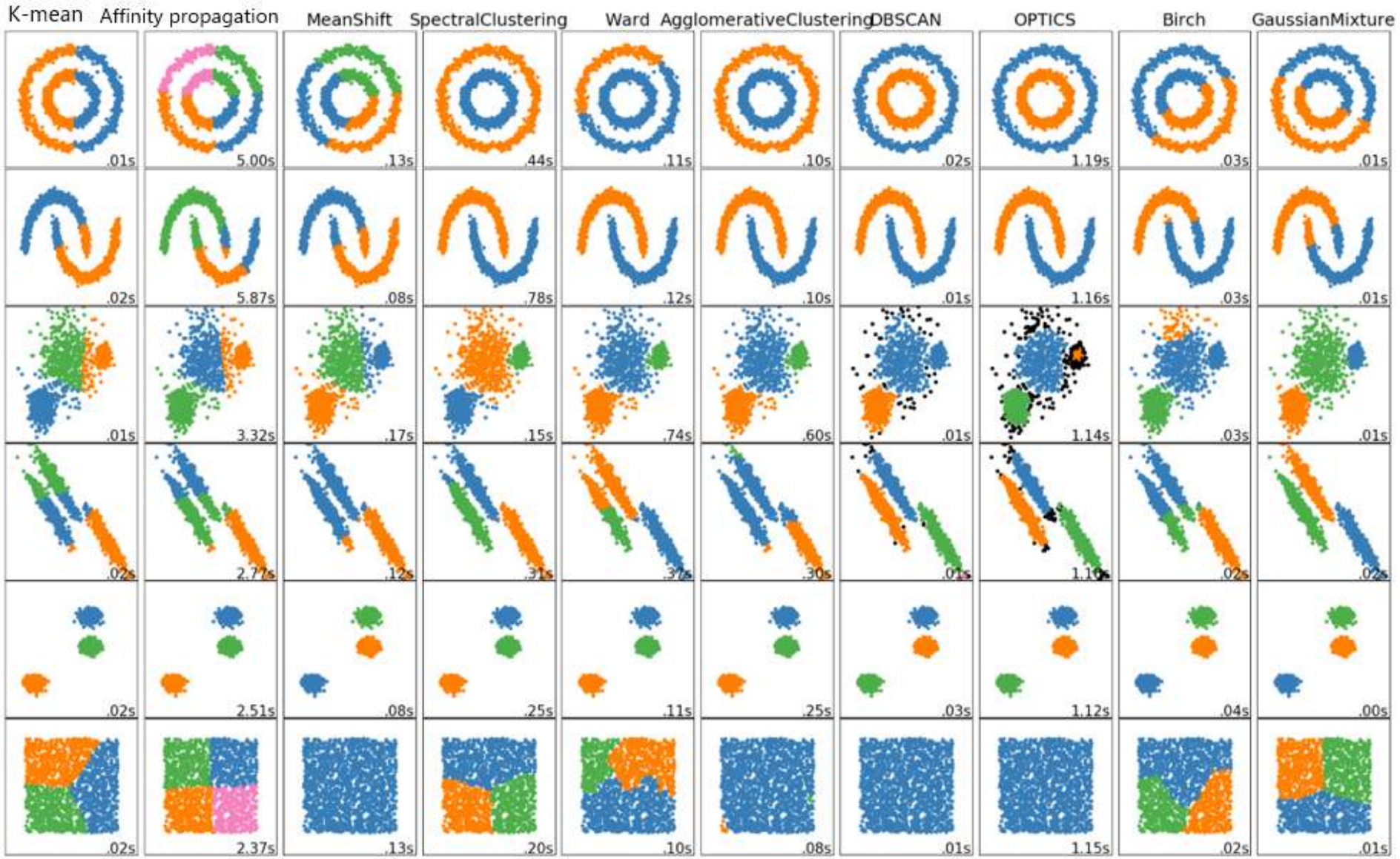


# 05

# scikit-learn：聚类问题

# scikit-learn：聚类问题

- 聚类模型



# scikit-learn：聚类问题

## ■ K-means 聚类

- K-means 的目标是将数据集划分为 K 个簇（clusters），使得每个数据点属于距离最近的簇中心。通过反复调整簇中心的位置，K-means 不断优化簇内的紧密度，从而获得尽量紧凑、彼此分离的簇。

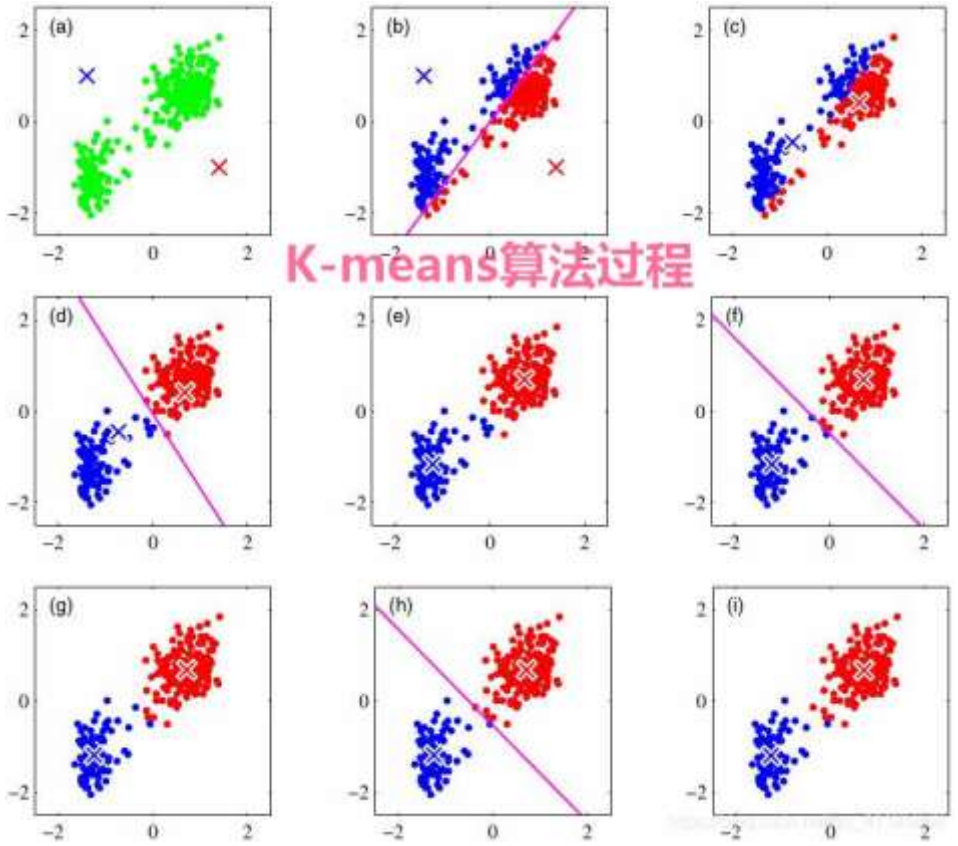
**核心思想：**K-means 使用“最近距离”来分组：

1.随机选择 K 个质心（初始中心点）。

2.每个数据点分配到距离最近的质心所属的簇。

3.重新计算每个簇的质心。

4.重复步骤 2 和 3，直到质心不再变化（或达到指定的迭代次数）。

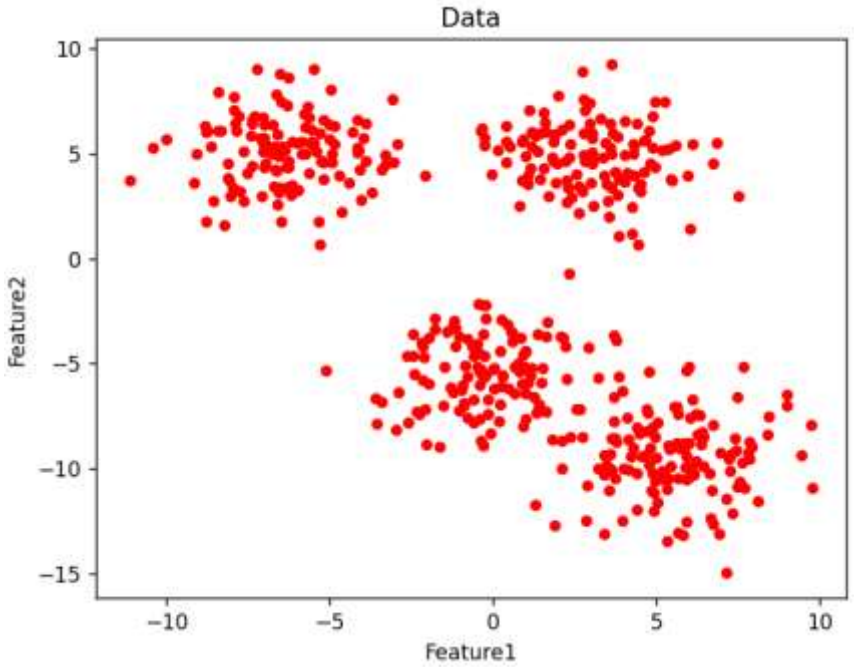


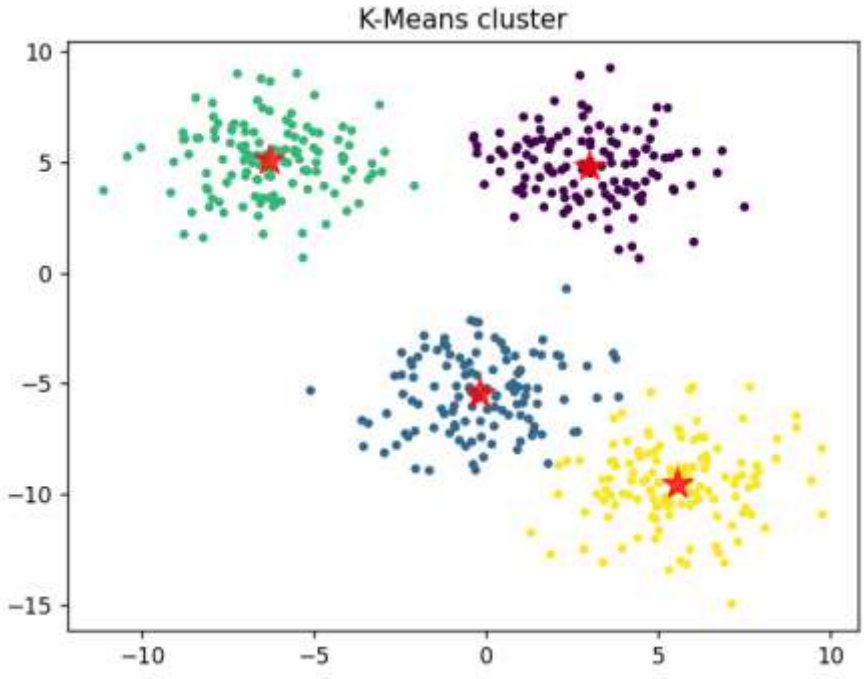
# scikit-learn：聚类问题

■ **案例1**：make_blobs() 函数，**生成聚类数据**

```
make_blobs（
    n_samples=100,
    n_features=2,
    centers = 4,
    cluster_std=1.8,
    shuffle=True,
    random_state=None,
)
```

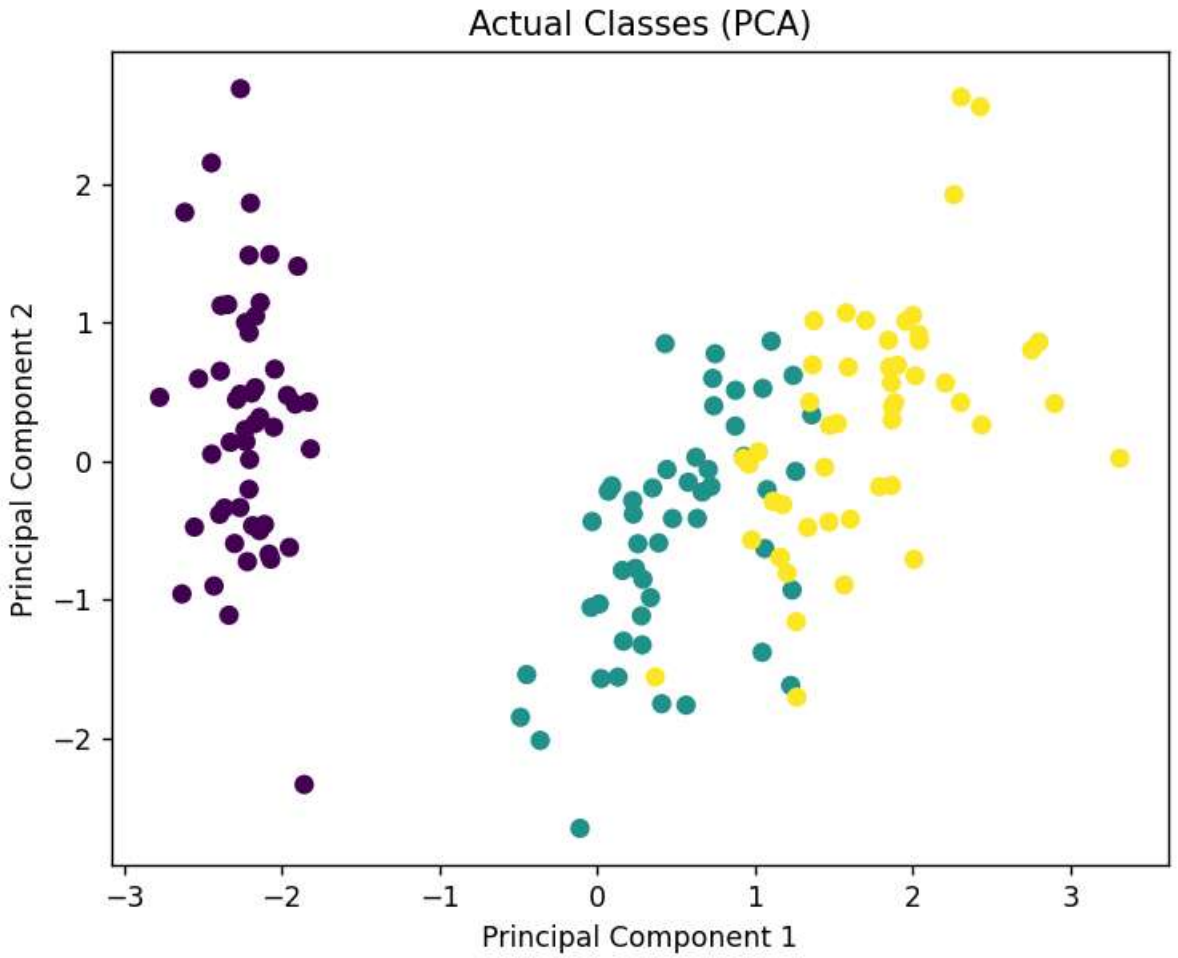
- **n_samples**：生成的样本点总数，默认为100。
- **n_features**：每个样本的特征数量，也就是数据的维度，默认为2。
- **centers**：要生成的数据中心（类别）数，或者是固定的中心点位置，默认生成3个中心。
- **cluster_std**：每个类别的标准偏差，可以是单个浮点数或浮点数序列，默认为1.0。
- **center_box**：随机生成中心时每个聚类中心的边界框，默认为(-10.0, 10.0)。
- **shuffle**：是否将数据进行洗牌，默认为True。
- **random_state**：随机数生成器的种子，可以固定生成的数据集。

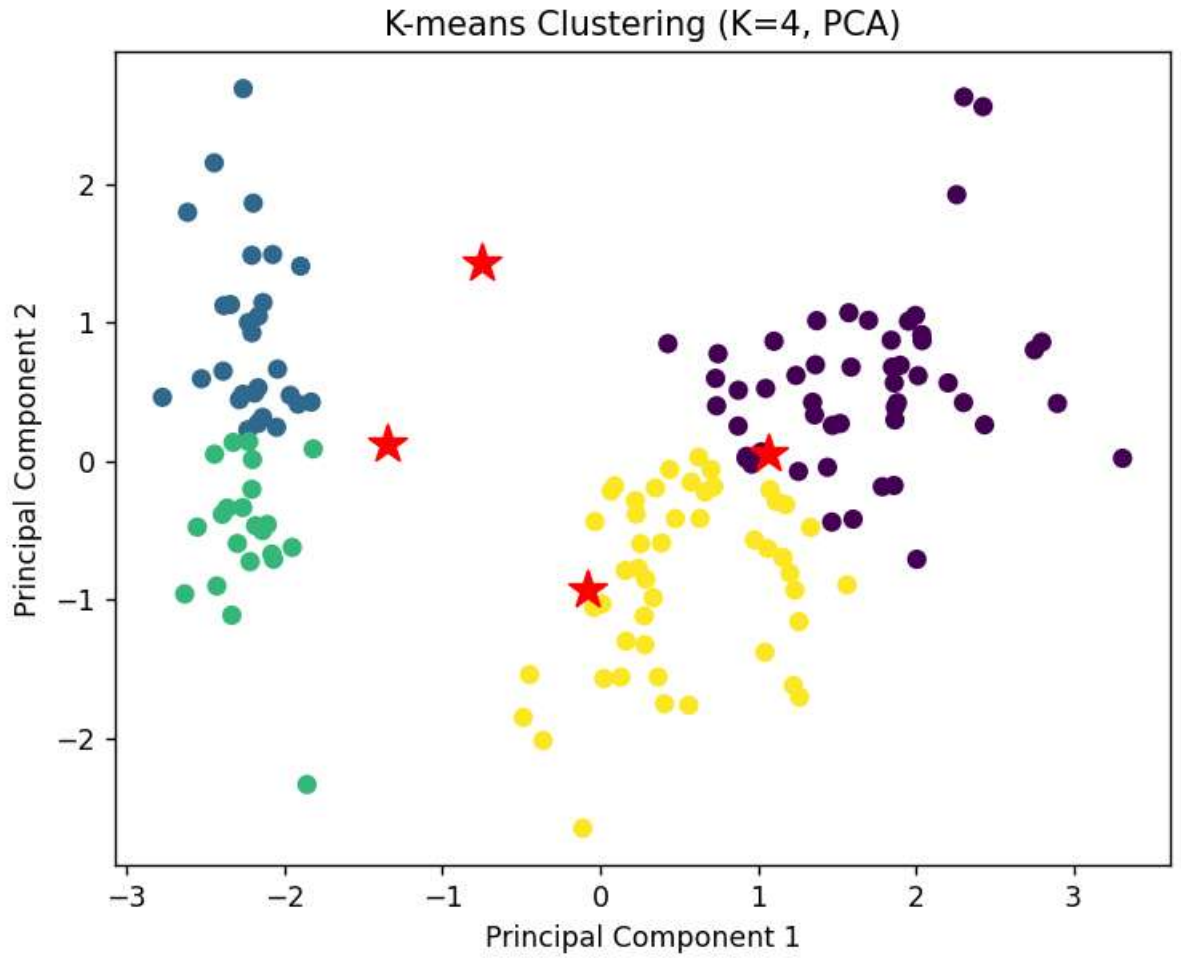


# scikit-learn：聚类问题

- **案例2**：鸢尾花（iris）聚类



# scikit-learn：聚类问题

- **案例3**：葡萄酒（wine）聚类