

A cluster of colorful geometric shapes, including triangles and squares in shades of blue, yellow, green, and orange, arranged in a complex, overlapping pattern in the top-left corner.

分类算法

万永权

教学目标

- ◆理解并掌握分类算法的基础知识点:
 - 算法: KNN、朴素贝叶斯、决策树、
 - 评估指标: 准确率、精准率、召回率等;
- ◆掌握KNN模型的设计原理和构建流程;
- ◆熟悉使用Scikit-learn使用KNN进行分类的方法;

目录

CONTENTS

1. 分类问题定义
2. 模型评价指标
3. KNN分类算法
4. 朴素贝叶斯分类
5. 逻辑回归
6. 决策树



监督学习之分类问题

分类问题

- ◆ 分类是监督学习中重要的任务之一。
- ◆ 分类任务中，预测的因变量是离散型的类别标签。
- ◆ 例：
 - 反垃圾邮件系统：是/不是垃圾邮件（spam）
 - 新闻自动分类：



分类问题

◆ **分类问题**是**监督学习**的一个核心问题，它从数据中学习一个分类决策函数或分类模型(分类器 (classifier))，对新的输入进行输出预测，输出变量取有限个**离散**值。

◆ **核心算法**

➤ k近邻、决策树、贝叶斯、SVM、逻辑回归



分类



目前 ImageNet 中总共有14197122幅图像，总共分为21841个类别(synsets)，顶级类包括：amphibian、animal、appliance、bird、covering、device、fabric、fish、flower、food、fruit、fungus、furniture、geological formation、invertebrate、mammal、musical instrument、plant、reptile、sport、structure、tool、tree、utensil、vegetable、vehicle、person 共27个。

IMGENET

Large Scale Visual Recognition Challenge
(ILSVRC 2010-2017)

不同分类任务

Binary
Classification



- Spam
- Not spam

Multiclass
Classification



- Dog
- Cat
- Horse
- Fish
- Bird
- ...

Multi Label Classification



Horror: 0.02%
Romance: 0.02%
Adventure: 99.96%
Documentary: 0.0%



分类模型



- 监督学习中，**分类**任务就是从数据中学习一个**模型**（也叫**分类器**），应用这一模型，对给定的输入 X ，预测相应的输出 Y 。

- 这个模型的一般形式包括：

- **决策函数** $Y = f(X)$
- **条件概率分布** $P(Y|X)$



- 其中的条件概率又有两种方法获得：

- 直接对条件概率值 $P(Y|X)$ 建模
- 首先对**联合分布概率** $P(X, Y)$ 建模然后使用**贝叶斯公式**计算所有类别的**条件概率值** $P(Y|X)$ ，选择概率最大的一种或几种作为预测结果



模型的评价



正例、负例



二分类问题的混淆矩阵由 4 部分构成。

首先将二分类问题中，最关心的、数量为少数的那一部分数据，称之为正例（positive）。

例如，如果需要预测癌症，癌症患者就定义为正例，剩下的就定义为负例（negative）。



模型的评价



混淆矩阵

标识	预测类别	真实类别	预测是否正确
TP	P	P	T
TN	N	N	T
FP	P	N	F
FN	N	P	F



这 4 个定义由 2 个字母组成：

- 第 1 个字母表示算法预测正确或者错误；
- 第 2 个字母表示算法预测的结果。

准确率 (Accuracy)

教材P159, 5.5节



准确率是最好理解的评价指标，它是一个比值：

$$\text{准确率} = \frac{\text{算法分类正确的数据个数}}{\text{输入算法的数据的个数}}$$

$$\text{accuracy} = \frac{TP + TN}{TP + FN + FP + TN}$$



在数据的类别不均衡，特别是有极偏的数据存在的情况下，准确率不能客观评价模型的优劣。

例如，1000人中有5人患病：

$$\begin{aligned}\text{准确率} &= \frac{\text{算法分类正确的数据个数}}{\text{输入算法的数据的个数}} \\ &= \frac{995}{1000} = 99.5\%\end{aligned}$$

	预测类别 P	预测类别 N
真实类别 P	TP	FN
真实类别 N	FP	TN

精确率 (Precision)

教材P159, 5.5节

◆ 预测正确的数据 占 预测为正例的那些数据的比例

$$precision = \frac{TP}{TP + FP}$$

	预测类别 P	预测类别 N
真实类别 P	TP	FN
真实类别 N	FP	TN

召回率 (Recall)

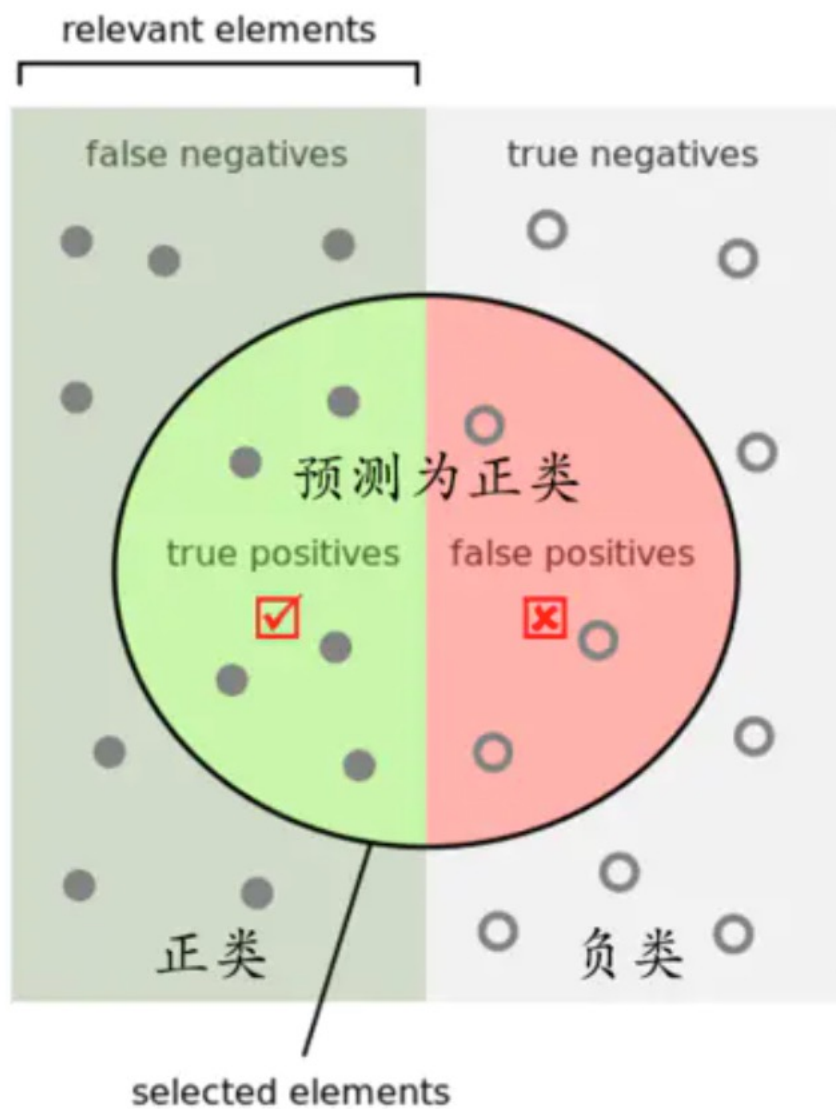
教材P159, 5.5节

◆ 预测正确的数据 占 真实为正例的那些数据的比例

$$recall = \frac{TP}{TP + FN}$$

	预测类别 P	预测类别 N
真实类别 P	TP	FN
真实类别 N	FP	TN

精确率和召回率



How many selected items are relevant?

$$\text{Precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}}$$

How many relevant items are selected?

$$\text{Recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}}$$



模型的评价



- 在不同的应用场景下，**关注点是不同的**。
- 例如，在预测股票的时候，更关心**精确率**，即预测涨的那些股票里，真的涨了有多少，因为那些预测涨的股票已经投资买入了。



- 而在预测病患的场景下，更关注**召回率**，即真的患病的那些人里预测错误的情况应该越少越好，因为真的患病如果没有检测出来，后果很严重。而之前那个预测所有人都健康的模型，其召回率就是 0。

不同的应用场景，关注点不同

1. 地震的预测

对于地震的预测，我们希望的是召回率非常高，也就是说每次地震我们都希望预测出来。

宁可错杀三千，不可放过一个

2. 嫌疑人定罪

基于不错怪一个好人的原则，对于嫌疑人的定罪我们希望是非常准确的。及时有时候放过了一些罪犯（召回率低），但也是值得的。

宁可放过三千，不可错杀一个



模型的评价



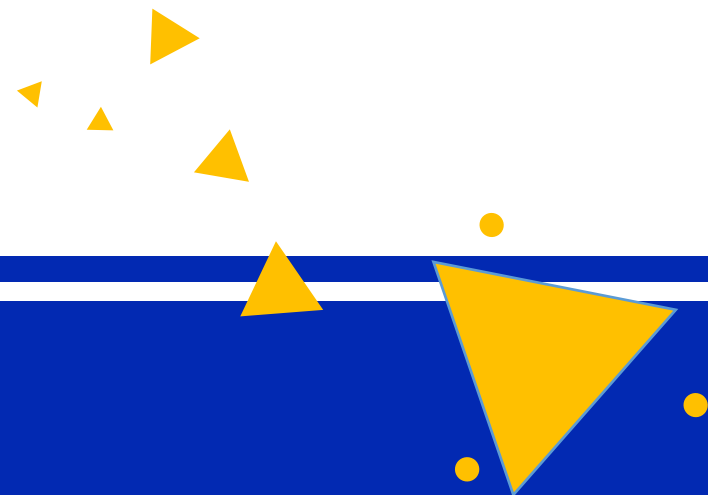
- 精准率和召回率是此消彼长的，即精准率高了，召回率就下降，在一些场景下要兼顾精准率和召回率，这就是 F1 score (F1测度)。

$$\frac{1}{F1} = \frac{1}{2} \left(\frac{1}{precision} + \frac{1}{recall} \right)$$

$$F1 = \frac{2 \times precision \times recall}{precision + recall}$$



- F1测度是精准率和召回率的调和平均数。
 - 只有当精准率和召回率二者都非常高的时候，它们的调和平均才会高。
 - 如果其中之一很低，调和平均就会被拉得接近于那个很低的数。



KNN算法

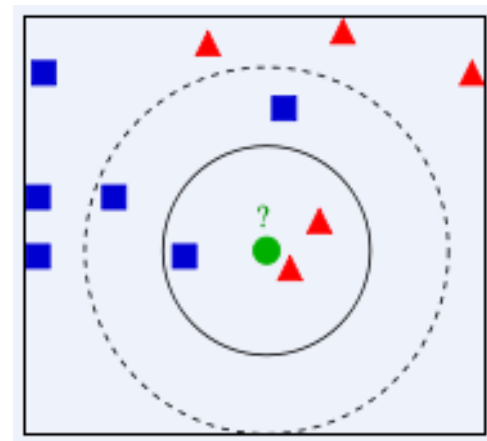
02

- ◆ KNN分类算法 (K-Nearest-Neighbors Classification),又叫K近邻算法。
- ◆ 1967年由Cover T和Hart P提出。
- ◆ 在分类任务中基于实例的学习, 没有训练阶段, 它是"懒惰学习"(lazy learning) 的著名代表训练时间开销为零, 待收到测试样本后再进行处理。
- ◆ 选择与其最近的K个训练样本, 类似于“投票法”选择K个中特征值出现较多的类别作为预测结果。

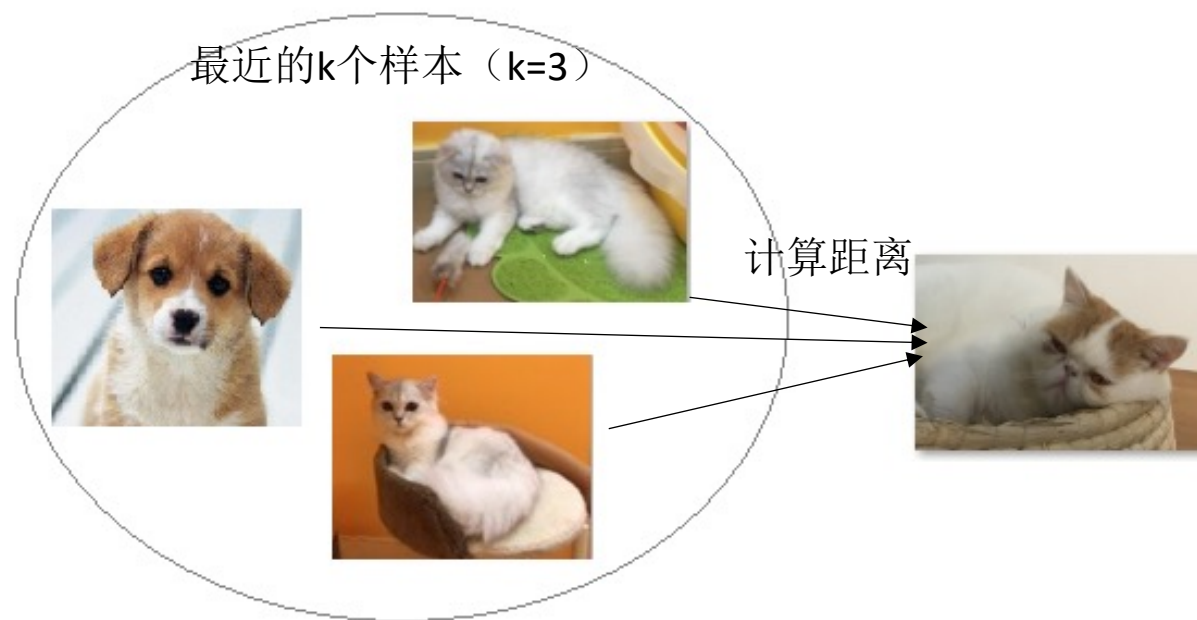
◆ KNN分类算法的**核心思想**:

- 如果一个样本在特征空间中的k个最相似(即特征空间中最邻近)的样本中的大多数属于某一个类别, 则该样本也属于这个类别。

近朱者赤、近墨者黑
——晋·傅玄《太子少傅箴》



如图，假设已经获取一些动物的特征，且已知这些动物的类别。现在需要识别一只新动物，判断它是哪类动物。



KNN分类示意图

首先找到与这个物体最接近的k个动物。假设 $k=3$ ，则可以找到2只猫和1只狗。由于找到的结果中大多数是猫，则把这个新动物划分为猫类。



K最近邻方法



- K最近邻法的基本规则是：
- 在所有N个样本中，找到测试样本的K ($K \leq N$) 个最近邻样本，当 $k=1$ 时，KNN方法就变成了最近邻方法。
- KNN 方法中，**K 一般为奇数**，跟投票表决一样，避免因两种票数相等而难以决策。



计算步骤如下：

- (1) **算距离**：给定测试对象，计算它与训练集中的每个对象的距离；
- (2) **找邻居**：圈定距离最近的 K 个训练对象，作为测试对象的近邻；
- (3) **做分类**：根据这 K 个近邻归属的主要类别，来对测试对象分类。

K-近邻算法 要素之一

◆ 距离度量选择

- 特征空间中两个实例点的距离是两个实例点相似程度的反映。
- KNN模型的特征空间一般是n维实数向量空间 \mathbf{R}^n 。

欧氏距离 $d_{ij} = \sqrt{\sum_{k=1}^N (x_{ik} - x_{jk})^2}$

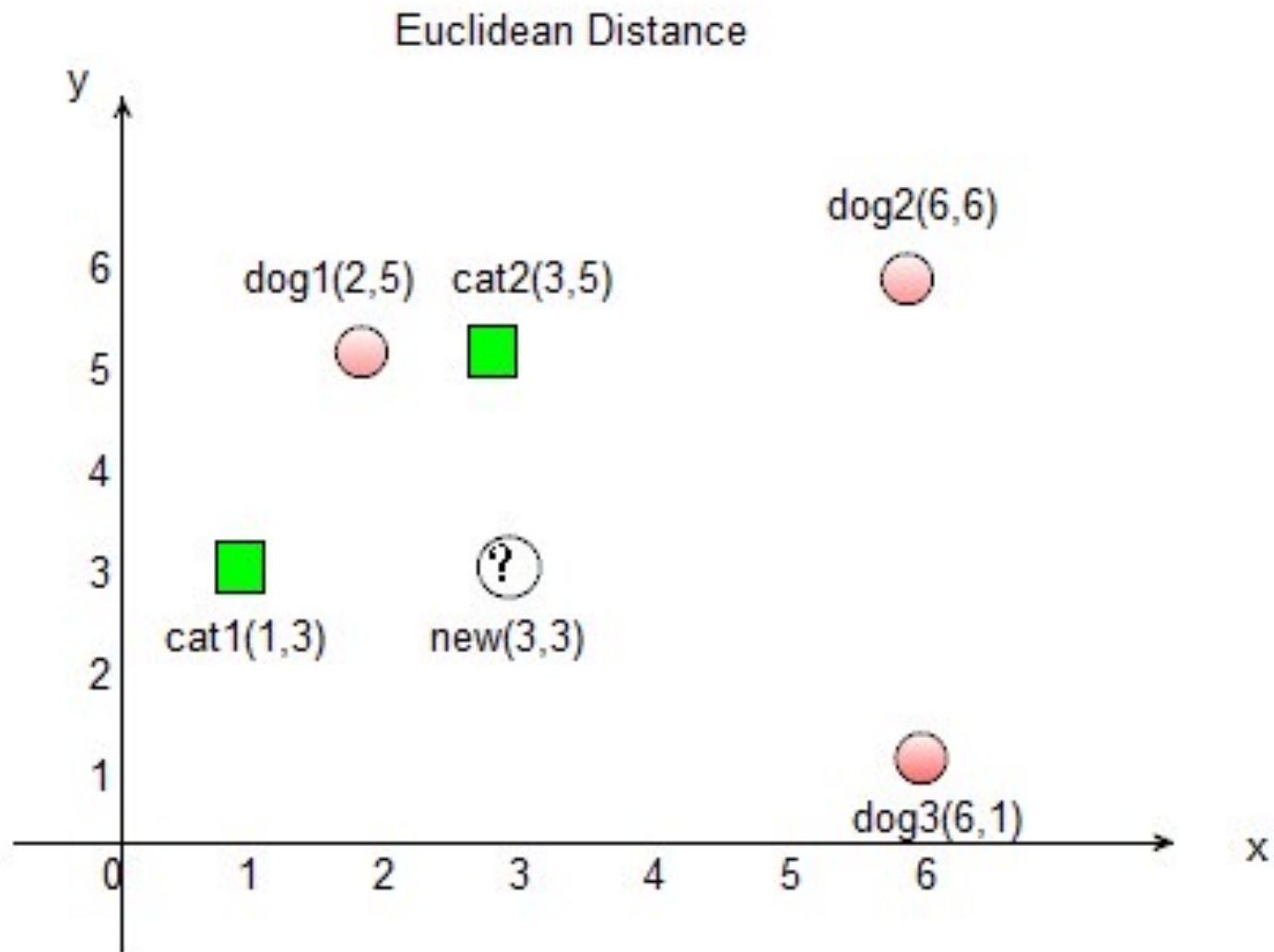
余弦相似度 $d_{ij} = \frac{\sum_{k=1}^N x_{ik} \cdot x_{jk}}{\sqrt{\sum_{k=1}^N x_{ik}^2} \sqrt{\sum_{k=1}^N x_{jk}^2}}$

曼哈顿距离 $d_{ij} = \sum_{k=1}^N |x_{ik} - x_{jk}|$



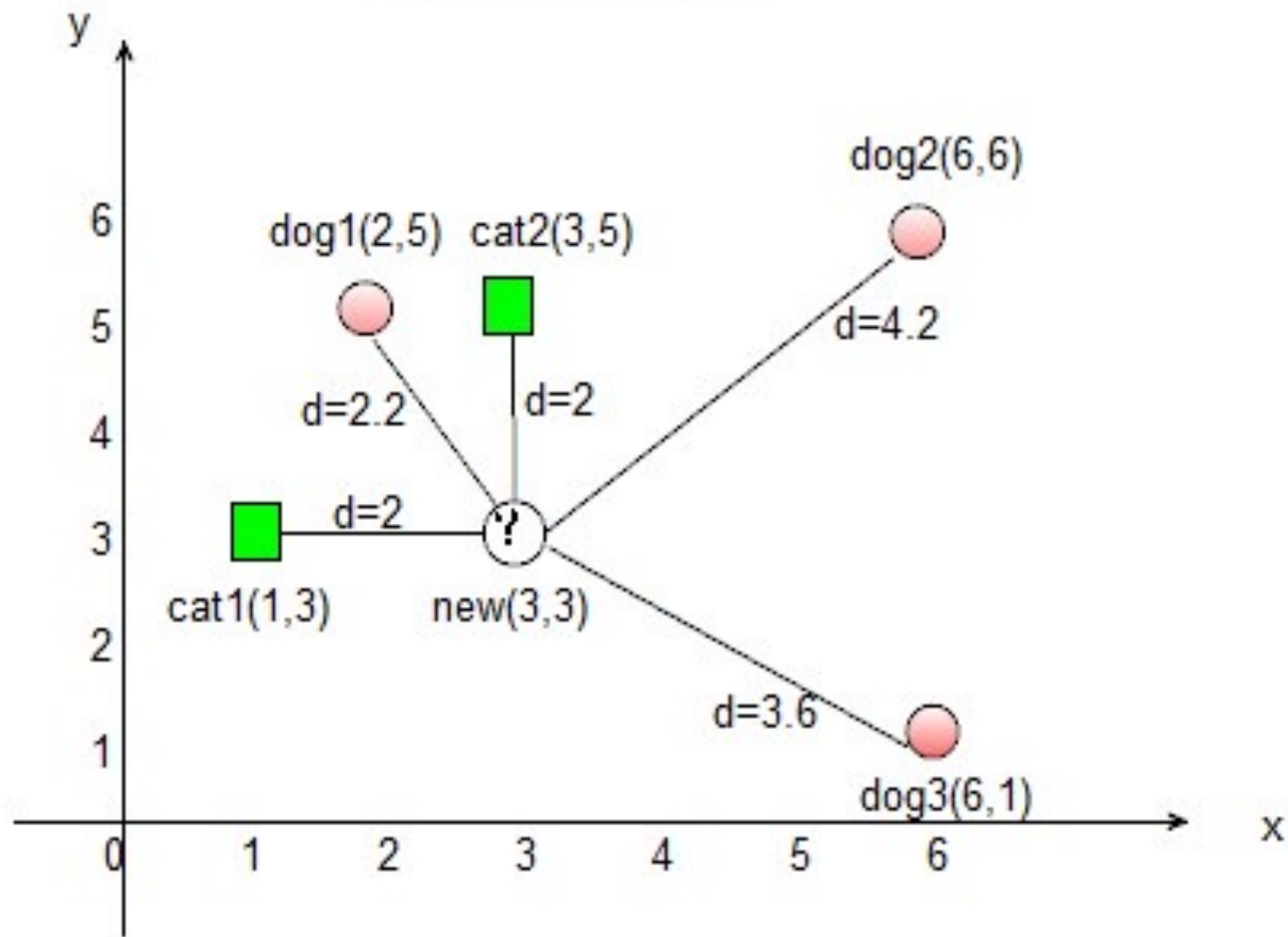
欧式距离公式:

$$\rho = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$





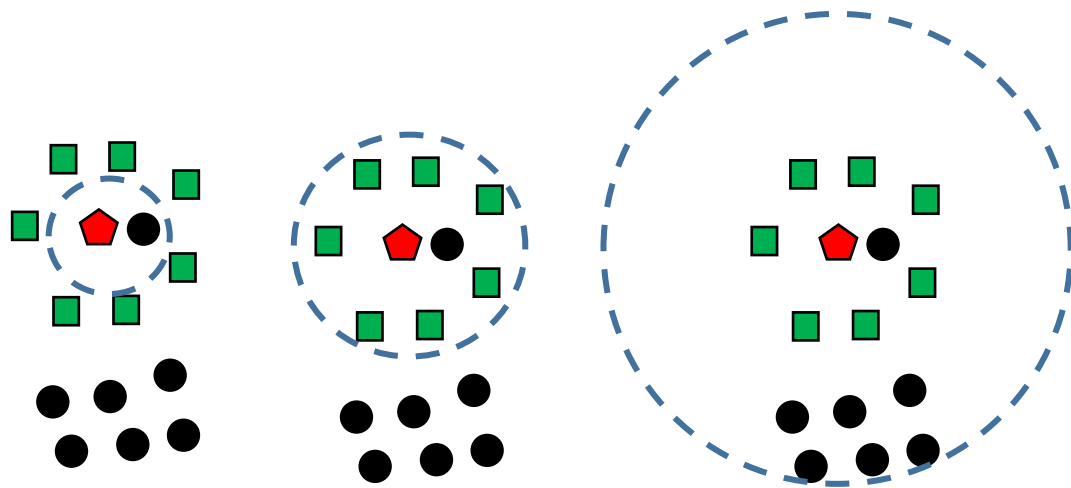
Euclidean Distance



K-近邻算法 要素之二

◆ K值的选择

- K值的选择会对KNN算法的结果产生重大影响。
- 如果选择较小的K值，学习”的近似误差会减小，预测结果会对近邻的实例点非常敏感，整体模型变得更复杂，容易发生过拟合。
- 如果选择较大的K值，模型变得更简单，但近似误差会增大，导致分类模糊，即欠拟合。



K值太小

K值适中

K值太大



下面举例看k值对预测结果的影响。对图5.2中的动物进行分类，当 $k=3$ 时，分类结果为“猫：狗=2：1”，所以属于猫；当 $k=5$ 时，表决结果为“猫：狗：熊猫=2：3：1”，所以判断目标动物为狗。

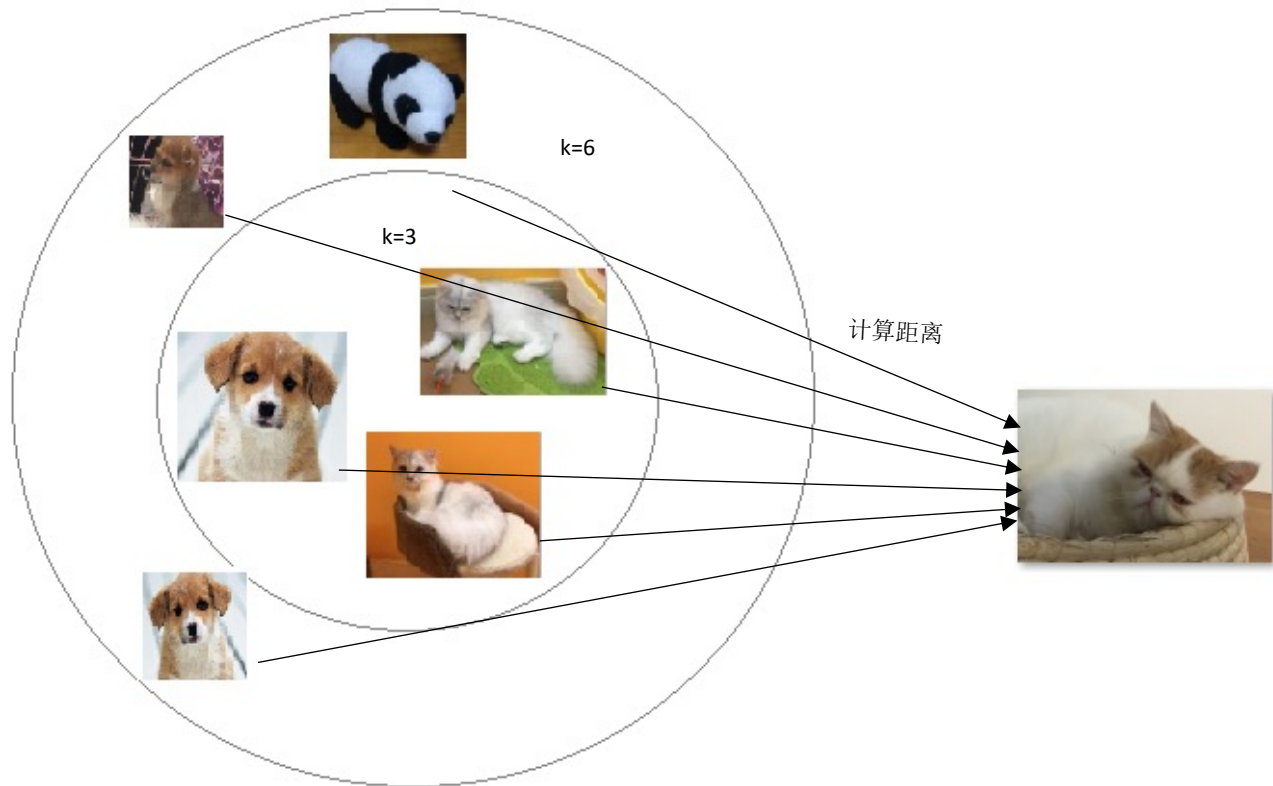


图5.2 不同K值对结果的影响示意图

K值的选择

- ◆ **K值得确定：** K的取值没有特定的算法，一般是通过分析师的经验来选取，推荐使用相对较小的值，通过交叉验证来选取最优解。
K 太小，分类结果易受噪声点影响；
- K 太大，近邻中又可能包含太多的其它类别的点。
- K 值通常是采用交叉检验来确定。
- 经验规则： K 一般低于训练样本数的平方根。

K -近邻算法 要素之三

◆ 分类决策规则

分类结果的确定往往采用**多数表决**原则，即由输入实例的 k 个最邻近的训练实例中的多数类决定输入实例的类别

KNN流程

- ◆ KNN算法完成**分类**任务时的流程：
- ◆ 对数据集中每个未知类别的样本依次执行以下操作。
 - ◆ ①准备好已知类别标签的数据集 D ，对数据进行预处理，假设 D 中共有 N 个数据样本。
 - ◆ ②计算测试样本 x （即待分类样本）到 D 中每个样本的距离，存入数组 $\text{Dist}[1..N]$ 。
 - ◆ ③对 $\text{Dist}[1..N]$ 中元素进行增序排序，找出其中距离最小的 K 个样本。
 - ◆ ④统计这 K 个样本所属类别出现的频率。
 - ◆ ⑤找出出现频率最高的类别，记为 c ，作为测试样本 x 的预测类别。

KNN分类实例-蚊子分类



1981年生物学家格若根（W. Grogan）和维什（W. Wirth）发现了两类蚊子(或飞蠓 midges)。他们测量了这两类15只蚊子的翼长和触角长，数据如下：

翼长	触角长	类别
• 1.78	1.14	Apf
• 1.96	1.18	Apf
• 1.86	1.20	Apf
• 1.72	1.24	Af
• 2.00	1.26	Apf
• 2.00	1.28	Apf
• 1.96	1.30	Apf
• 1.74	1.36	Af

翼长	触角长	类别
• 1.64	1.38	Af
• 1.82	1.38	Af
• 1.90	1.38	Af
• 1.70	1.40	Af
• 1.82	1.48	Af
• 1.82	1.54	Af
• 2.08	1.56	Af

问：如果抓到三只新的蚊子，它们的触角长和翼长分别为：

#1: 1.90, 1.25;

#2: 1.82, 1.50;

#3: 1.96, 1.43;

问它们应分别属于哪一个种类？

结果

翼长	触角长	类别	#1 距离	#2 距离	#3距离
1.78	1.14	1	0.16	0.36	0.34
1.96	1.18	1	0.09	0.35	0.25
1.86	1.2	1	0.06	0.30	0.25
1.72	1.24	0	0.18	0.28	0.31
2	1.26	1	0.10	0.30	0.17
2	1.28	1	0.10	0.28	0.16
1.96	1.3	1	0.08	0.24	0.13
1.74	1.36	0	0.19	0.16	0.23
1.64	1.38	0	0.29	0.22	0.32
1.82	1.38	0	0.15	0.12	0.15
1.9	1.38	0	0.13	0.14	0.08
1.7	1.4	0	0.25	0.16	0.26
1.82	1.48	0	0.24	0.02	0.15
1.82	1.54	0	0.30	0.04	0.18
2.08	1.56	0	0.36	0.27	0.18

当K=3时:

#1: 为1类Apf;

#2: 为0类Af;

#3: 为0类Af;

若K=5时, 结果是否发生改变?

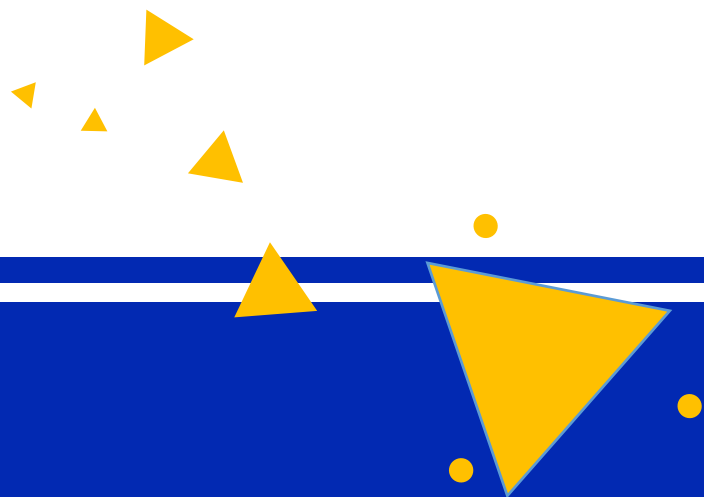
KNN算法的优缺点

◆ KNN算法的优点有：

- 思路简单，易于理解，易于实现，无需训练过程，不必估计参数，可直接用训练数据来实现分类。
- 只需人为确定两个参数，即K的值和距离函数。KNN算法支持多分类。

◆ KNN算法有以下4点不足：

- ① 对超参数K的选择十分敏感，针对同一个数据集和同一个待测样本选择不同K值，会导致得到完全不同的分类结果。
- ② 当样本不平衡时，例如一个类中样本数很多，而其他类中样本数很少，有可能导致总是将新样本归入大容量类别中，产生错误分类。
- ③ 当不同类别的样本数接近时或有噪声时，会增加决策失误的风险。
- ④ 当样本的特征维度很高或训练数据量很大时，计算量较大，KNN算法效率会降低，因为对每一个待分类样本都要计算它与全体已知样本之间的距离，才能找到它的K个最近邻点。目前常用的解决方法是事先对已知样本进行剪枝，去除对分类作用不大的样本；另外，可以对训练数据进行快速K近邻搜索。



分类案例

机器学习案例

◆ **问题：** 如何对如下三种鸢尾花进行分类？



山鸢尾



变色鸢尾



维吉尼亚鸢尾

实践： 鸢尾花分类

- ◆ 问题描述： 一名植物学爱好者对发现的鸢尾花的品种很感兴趣。他收集了每朵鸢尾花的一些测量数据：**花瓣的长度和宽度以及花萼的长度和宽度**。他还有一些鸢尾花分类的测量数据，这些花之前已经被植物学专家鉴定为属于 **setosa**、**versicolor** 或 **virginica** 三个品种之一。
- ◆ 对于这些测量数据，他可以确定每朵鸢尾花所属的品种。我们的目标是**构建一个机器学习模型**，可以从这些已知品种的鸢尾花测量数据中进行学习，从而能够**预测新鸢尾花的品种**。

机器学习案例

如何根据鸢尾花的花萼和花瓣大小将其分为三种不同的品种？



山鸢尾



变色鸢尾



维吉尼亚鸢尾

特征

花萼长度	花萼宽度	花瓣长度	花瓣宽度
5.1	3.3	1.7	0.5
5.0	2.3	3.3	1.0
6.4	2.8	5.6	2.2

model



结果

品种 (标签)
0 (山鸢尾)
1 (变色鸢尾)
2 (维吉尼亚鸢尾)

5.2 初识KNN——鸢尾花分类

1. 查看数据

SKlearn中的iris数据集有5个key，分别如下：

- ◆ target_names：分类名称，包括setosa、versicolor和virginica类。
- ◆ data：特征数据值。
- ◆ target：分类（150个）。
- ◆ DESCR：数据集的简介。
- ◆ feature_names：特征名称。

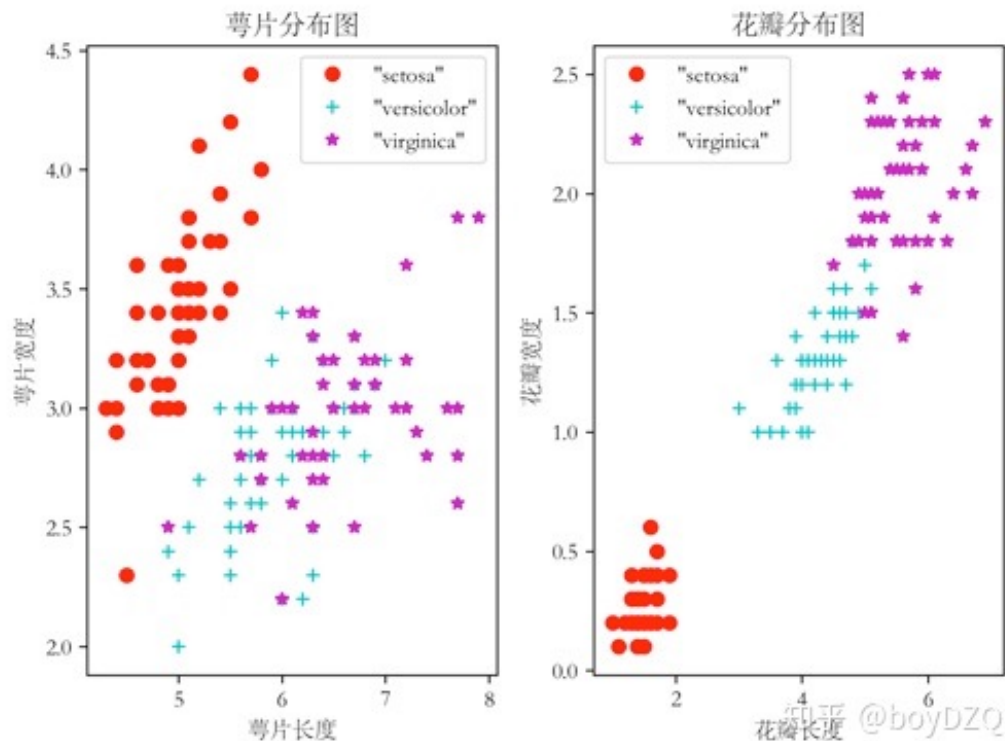
【例】查看鸢尾花iris数据集。

数据可视化分析

- ◆以图表的形式观察鸢尾花花瓣长度和宽度与鸢尾花品种的关系、鸢尾花花萼长度和宽度与鸢尾花品种的关系。关键代码如下：
- ◆生成的数据图效果如下，通过图可以观察到花瓣的长度和宽度与鸢尾花品种的相关性更强。

```
def draw(f_datas, labels): # 绘制数据图
    d_dict = defaultdict(list) # 创建字典，默认值为列表类型
    for i in range(len(labels)): # 循环遍历每一个样本
        d_dict[labels[i]].append(f_datas[i]) # 将同一品种的鸢尾花放在一起
    styles = ["ro", "b+", "g*"] # 设置一些样式
    plt.rcParams["font.family"] = "STSong" # 设置支持中文
    plt.subplot(1, 2, 1) # 添加子块
    for i, (key, values) in enumerate(d_dict.items()):
        values = np.array(values) # 获取每一个品种对应的样本
        plt.plot(values[:, 0], values[:, 1], styles[i], label=key)
    plt.legend() # 显示图例
    plt.title("花萼的分布图") # 显示标题
    plt.xlabel("花萼的长度") # 显示X轴标签
    plt.ylabel("花萼的宽度") # 显示Y轴标签
    plt.subplot(1, 2, 2) # 添加子块
    for i, (key, values) in enumerate(d_dict.items()):
        values = np.array(values)
        plt.plot(values[:, 2], values[:, 3], styles[i], label=key)
    plt.legend() # 显示图例
    plt.title("花瓣的分布图") # 显示标题
    plt.xlabel("花瓣的长度") # 显示X轴标签
    plt.ylabel("花瓣的宽度") # 显示Y轴标签
    plt.subplots_adjust(wspace=0.4) # 调整位置
    plt.show() # 显示图片
```

知乎 @boyDZQ



知乎 @boyDZQ



2 . 数据集拆分

使用train_test_split函数。train_test_split函数属于sklearn.model_selection类中的交叉验证功能，能随机地将样本数据集合拆分成训练集和测试集。

【例】 对iris数据集进行拆分，并查看拆分结果。

```
X_train, X_test, y_train, y_test = train_test_split( iris_dataset['data'],  
iris_dataset['target'], random_state=2)
```

3. 使用散点矩阵查看数据特征关系

在数据分析中，同时观察一组变量的散点图是很有意义的，这也被称为散点图矩阵（scatter plot matrix）。创建这样的图表工作量巨大，可以使用scatter_matrix函数。scatter_matrix函数是Pandas提供了一个能从DataFrame创建散点图矩阵的函数。

函数格式：

```
scatter_matrix(frame, alpha=0.5, c,figsize=None, ax=None, diagonal='hist',  
marker='.', density_kwds=None,hist_kwds=None, range_padding=0.05, **kwds)
```

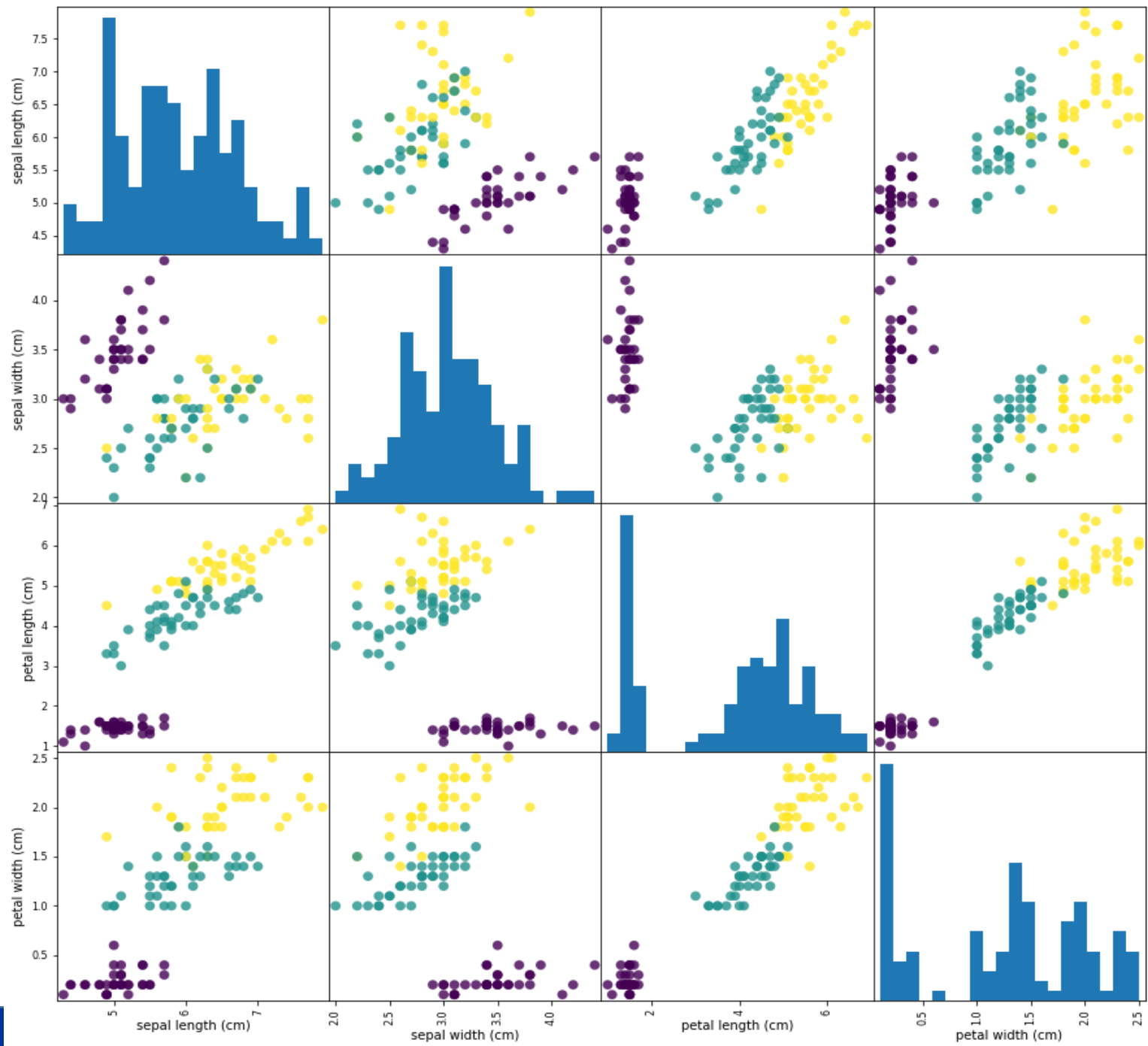
重要参数解释：

- ◆ frame: Pandas dataframe对象。
- ◆ alpha: 图像透明度，一般取(0,1)。
- ◆ figsize: 以英寸为单位的图像大小，一般以元组 (width, height) 形式设置。
- ◆ Diagonal: 必须且只能在{'hist','kde'}中选择1个，'hist'表示直方图(Histogram plot),'kde'表示核密度估计(Kernel Density Estimation); 该参数是scatter_matrix函数的关键参数。
- ◆ marker: Matplotlib可用的标记类型，如'.', ',', 'o'等。



【例】对鸢尾花数据结果，使用scatter_matrix显示训练集与测试集的散点图矩阵。

```
pd.plotting.scatter_matrix(iris_dataframe, c=y_train, figsize=(15, 15), marker='o',  
hist_kwds={'bins': 20}, s=60, alpha=.8)
```





4 . 建立KNN模型

在Python中，实现KNN方法使用的是KNeighborsClassifier类，KNeighborsClassifier类属于Scikit-learn的neighbors包。



KNeighborsClassifier使用很简单，核心操作包括三步：

1) 创建KNeighborsClassifier对象，并进行初始化。

基本格式：

```
sklearn.neighbors.KNeighborsClassifier(n_neighbors=5, weights='uniform', algorithm='auto', leaf_size=30, p=2, metric='minkowski', metric_params=None, n_jobs=None, **kwargs)
```

主要参数：

- ◆ `n_neighbors`: int型,可选，缺省值是5，代表KNN中的近邻数量k值。
- ◆ `weights`: 计算距离时使用的权重，缺省值是“uniform”，表示平等权重。也可以取值“distance”，则表示按照距离的远近设置不同权重。还可以自主设计加权方式，并以函数形式调用。
- ◆ `metric`: 距离的计算，缺省值是“minkowski”。当`p=2`, `metric='minkowski'`时，使用的是欧式距离。`p=1`, `metric='minkowski'`时为曼哈顿距离。



2) 调用`fit`方法，对数据集进行训练。

函数格式： `fit(X, y)`

说明： 以X为训练集， 以y为测试集对模型进行训练。



3) 调用`predict`函数，对测试集进行预测。

函数格式： `predict(X)`

说明： 根据给定的数据预测其所属的类别标签。


```
from sklearn import datasets
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
#导入鸢尾花数据并查看数据特征
iris = datasets.load_iris()
print('数据集结构: ',iris.data.shape)
# 获取属性
iris_X = iris.data
# 获取类别
iris_y = iris.target
# 划分成测试集和训练集
iris_train_X,iris_test_X,iris_train_y,iris_test_y=train_test_split(iris_X,iris_y,test_size=0.2, random_state=0)
#分类器初始化
knn = KNeighborsClassifier()
#对训练集进行训练
knn.fit(iris_train_X, iris_train_y)
#对测试集数据的鸢尾花类型进行预测
predict_result = knn.predict(iris_test_X)
print('测试集大小: ',iris_test_X.shape)
print('预测结果: ',predict_result)
print('预测精确率: ',knn.score(iris_test_X, iris_test_y))
```

综合实验：KNN对鸢尾花数据集进行分类

程序显示出了30个测试样本的预测结果
测试样本的真值是什么？
请设法显示出来。

```
数据集结构: (150, 4)
测试集大小: (30, 4)
真实结果: [2 1 0 2 0 2 0 1 1 1 2 1 1 1 1 0 1 1 0 0 2 1 0 0 2 0 0 1 1 0]
预测结果: [2 1 0 2 0 2 0 1 1 1 2 1 1 1 2 0 1 1 0 0 2 1 0 0 2 0 0 1 1 0]
预测精确率: 0.9666666666666667
```