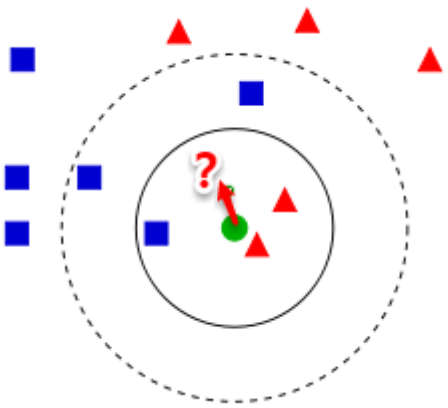
K 近邻（KNN）

遍历训练集中的所有样本，计算每个样本与待测样本的距离，并从中挑选出最近邻，根据与

距离成反比的权重，做加权投票（分类）或者平均（回归），得到待测样本的类别标签或预测数值。

k 近邻算法简称 kNN（k-Nearest Neighbor），是一种经典的监督学习方法，同时也实力担当入选数据挖掘十大算法。其工作机制十分简单粗暴：给定某个测试样本，kNN 基于某种**距离度量**在训练集中找出与其距离最近的 k 个带有真实标记的训练样本，然后给基于这 k 个邻居的真实标记来进行预测，类似于前面集成学习中所讲到的基学习器结合策略：分类任务采用投票法，回归任务则采用平均法。接下来本篇主要就 kNN 分类进行讨论。



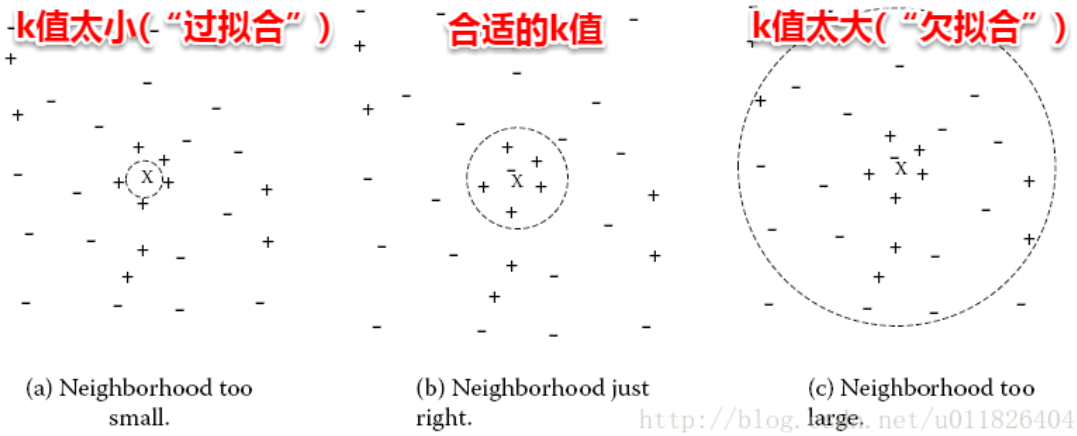
从左图中我们可以看到，图中有两种类型的样本，一类是蓝色正方形，另一类是红色三角形。而那个绿色圆形是我们待分类的样本。基于 kNN 算法的思路，我们很容易得到以下结论：

如果 K=3，那么离绿色点最近的有 2 个红色三角形和 1 个蓝色的正方形，这 3 个点投票，于是绿色的这个待分类点属于红色的三角形。
如果 K=5，那么离绿色点最近的有 2 个红色三角形和 3 个蓝色的正方形，这 5 个点投票，于是绿色的这个待分类点属于蓝色的正方形。

可以发现：**kNN 虽然是一种监督学习方法，但是它却没有显式的训练过程**，而是当有新样本需要预测时，才来计算出最近的 k 个邻居，因此 **kNN 是一种典型的懒惰学习方法**，再来回想一下朴素贝叶斯的流程，训练的过程就是参数估计，因此朴素贝叶斯也可以懒惰式学习，此类技术在**训练阶段开销为零**，待收到测试样本后再进行计算。相应地我们称那些一有训练数据立马开工的算法为“**急切学习**”，可见前面我们学习的大部分算法都归属于急切学习。

很容易看出：**kNN 算法的核心在于 k 值的选取以及距离的度量**。k 值选取太小，模型很容易受到噪声数据的干扰，例如：极端地取 k=1，若待分类样本正好与一个噪声数据距离最近，就导致了分类错误；若 k 值太大， 则在更大的邻域内进行投票，此时模型的预测能力大大减弱，例如：极端取 k=训练样本数，就相当

于模型根本没有学习，所有测试样本的预测结果都是一样的。**一般地我们都通过交叉验证法来选取一个适当的 k 值。**



对于距离度量，不同的度量方法得到的 k 个近邻不尽相同，从而对最终的投票结果产生了影响，因此选择一个合适的距离度量方法也十分重要。在上一篇聚类算法中，在度量样本相似性时介绍了常用的几种距离计算方法，包括闵可夫斯基距离，曼哈顿距离，VDM 等。在实际应用中，kNN 的距离度量函数一般根据样本的特性来选择合适的距离度量，同时应对数据进行去量纲/归一化处理来消除大量纲属性的强权政治影响。k-近邻算法采用测量不同特征值之间的距离来进行分类优点：精度高、对异常值不敏感、无数据输入假定缺点：计算复杂度高、空间复杂度高

使用数据范围：数值型和标称型电影可以按照题材分类，每个题材又是如何定义的呢？那么假如两种类型的电影，动作片和爱情片。动作片有哪些公共的特征？那么爱情片又存在哪些明显的差别呢？我们发现动作片中打斗镜头的次数较多，而爱情片中接吻镜头相对更多。当然动作片中也有一些接吻镜头，爱情片中也会有一些打斗镜头。所以不能单纯通过是否存在打斗镜头或者接吻镜头来判断影片的类别。那么现在我们有 6 部影片已经明确了类别，也有打斗镜头和接吻镜头的次数，还有一部电影类型未知。

电影名称	打斗镜头	接吻镜头	电影类型
California Man	3	104	爱情片
He's not Really into dues	2	100	爱情片
Beautiful Woman	1	81	爱情片
Kevin Longblade	101	10	动作片
Robo Slayer 3000	99	5	动作片
Amped II	98	2	动作片
?	18	90	未知

那么我们使用 K-近邻算法来分类爱情片和动作片：存在一个样本数据集合，也叫训练样本集，样本个数 M 个，知道每一个数据特征与类别对应关系，然后存在未知类型数据集合 1 个，那么我们要选择一个测试样本数据中与训练样本中 M 个的距离，排序过后选出最近的 K 个，这个取值一般不大于 20 个。选择 K 个最相近数据中次数最多的分类。那么我们根据这个原则去判断未知电影的分类

电影名称	与未知电影的距离
California Man	20.5
He's not Really into dues	18.7
Beautiful Woman	19.2
Kevin Longblade	115.3
Robo Slayer 3000	117.4
Amped II	118.9

我们假设 K 为 3，那么排名前三个电影的类型都是爱情片，所以我们判定这个未知电影也是一个爱情片。那么计算距离是怎样计算的呢？

欧氏距离：A(x1,y1)与 B(x2,y2)

$$\sqrt{(x_1-x_2)^2+(y_1-y_2)^2}$$ ，

如果说输入变量有四个特征，例如（1，3，5，2）和（7，6，9，4）之间的距离计算为：

$$\sqrt{(1-7)^2+(3-6)^2+(5-9)^2+(2-4)^2}$$

二、应用

1、寻找最近邻（FNN)

FNN 模型 = sklearn.neighbors.NearestNeighbors(n_neighbors=邻居数,algorithm=算法)

FNN.fit(已知样本集合)

1 x1 x2 .... xn

2 x1 x2 .... xn

....

....

n x1 x2 .... xn

FNN 模型.kneighbors(待求样本矩阵) --> 距离矩阵，近邻下标的索引矩阵

...假设邻居数为 3

3 x1 x2 .... xn 4 3 11 0.1 0.3 0.5

4 x1 x2 .... xn 20 10 11 0.2 0.6 0.9

...

示例代码：02-fnn.py

2、KNN 分类

import sklearn.neighbors as sn

KNN 分类模型 = sn.KNeighborsClassifier(n_neighbors=邻居数,weights=权重指标)

代码示例：01-knnc.py

3、KNN 回归

import sklearn.neighbors as sn

KNN 回归模型 = sn.KNeighborsRegressor(n_neighbors=邻居数,weights=权重指标)

01-knnr.py

4、欧式（欧几里得）距离得分

[x1,y1,z1,q1] <----> [x2,y2,z2,q2]

欧氏距离=sqrt[(x1-x2)^2+(y1-y2)^2+(z1-z2)^2+(q1-q2)^2]

$$\text{欧氏距离} = \frac{1}{1 + \text{欧式距离}}$$

0<-不相似 相似->1

	用户 1	用户 2	用户 3	用户 4 ...
用户 1	1.	0.29	0.47	0.39
用户 2	0.29	1.	0.34	0.28
用户 3	0.47	0.34	1.	0.54
...				

*欧氏距离可以用来评价用户的相似性，但无法描述用户的正负相关性。

示例代码：02-eus.py

5、皮氏（皮尔逊）距离得分

用于度量两个样本的相关（线性相关），其值介于-1 和 1 之间，可用来描述两个样本的相似度以及相关性。

1 表示完全正相关，0 表示无关，-1 表示完全负相关

*当 x 的值增大（减小），y 值增大（减小），两个变量为正相关，相关系数在 0 和 1 之间。

*当 x 的值增大（减小），y 值减小（增大），两个变量为负相关，相关系数在-1 和 0 之间。

示例代码：03-ps.py

6、根据样本的相似度排序，生成针对每个用户的推荐列表

示例代码：04-sim.py