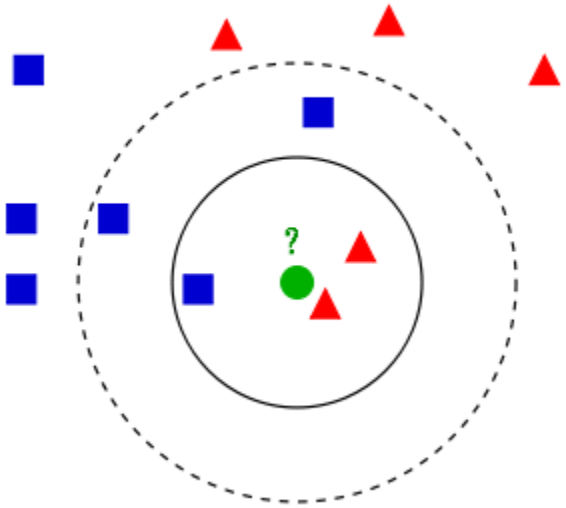
一、 KNN 算法简介

KNN（k-Nearest Neighbor）算法，或者叫 K 近邻，K 最近邻算法，是一个理论上比较成熟的方法，也是最简单的机器学习算法之一，适合刚接触机器学习领域的新人学习和掌握。



如图所示，蓝色方块和红色三角是已经分类好的数据，这时又输入了一个绿色圆圈数据，如何对其进行分类呢？我们需要测量一下绿圆圈到已经分类好的数据的距离，然后对测量出的距离进行分类，取出距离最近的几个（例如 3 个）数据，然后再观测到这距离最近的三个数据中有两个是蓝方块，于是就把绿圆圈划分到蓝方块这一个类别中。

这实际上就是一个 KNN 算法的通俗表示。科学一点来描述的话，就是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。就像一句经常被提到的话：物以类聚人以群分，观其友而识其人。

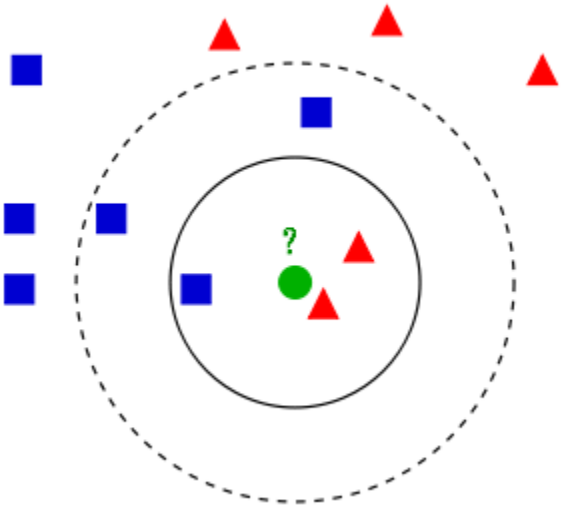
二、 KNN 算法的流程：

1. 给定一个训练集数据，在训练集中，每一个数据都是已经分好类的。
2. 输入一个数据 a，计算 a 到训练集所有数据的欧几里得距离，并排序。
3. 选出训练集中离 a 距离最近的 K 个训练集数据。
4. 在这 K 个训练集数据中，选出里面出现次数最多的分类类型，此分类类型即为数据 a 的分类。

三、 KNN 中需要注意的点

KNN 算法的基本思想和流程都是非常简单的，但它也有许多需要注意的点。

1. K 值的确定



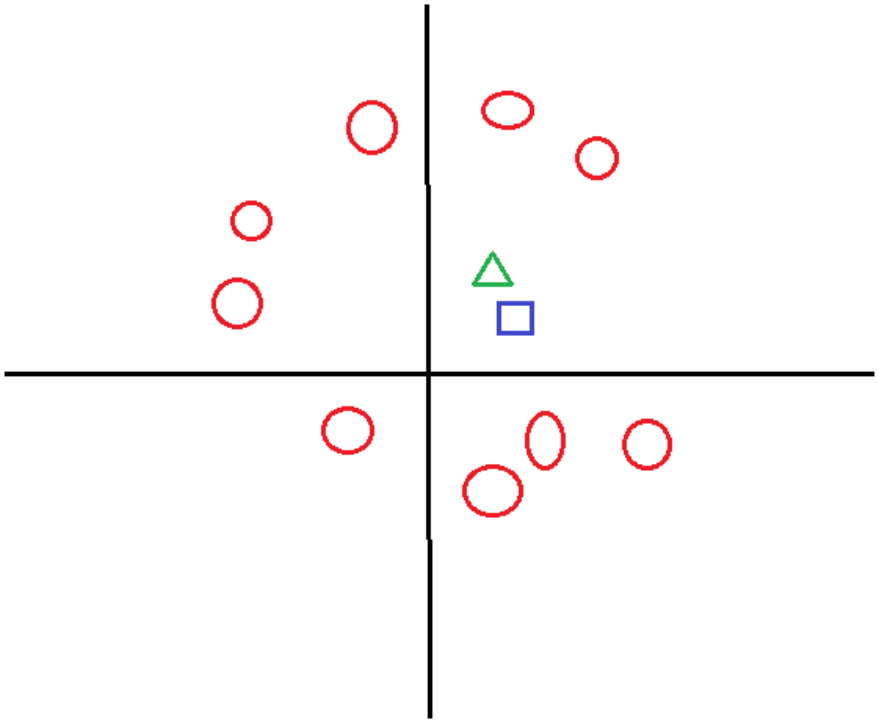
上图中，蓝方块和红三角分别是已经完成好分类的训练集数据，而绿圆形则是待分类的数据。如何对其进行分类呢？

当 K=3 时，显然距离绿色圆圈最近的数据是两个红三角和一个蓝方块，这时候自然应该把绿圆圈分在红三角一类中；

当 k=5 时，情况就不一样了，距离绿圆圈最近的数据是三个蓝方块和两个红三角，此时就应该把绿圆圈分在蓝方块一类中。

在上面的例子中 ,我们已经发现了 ,当 K 取不同的值时候分类结果也不一样。举一个极端的例子，当 K 取一个非常非常大的值，甚至超过了训练集数据量，这时很明显，无论再输入一个什么数据，这都是在把它划分进当前训练集中实例最多的类，此时分类算法就失效了；

而当 K 非常小，例如取 K=1 时，则会导致过拟合，很容易学习到噪声，这时分类效果也会非常差。如下图



当取 K=1 时很显然绿三角应该会被分在蓝方块一类中。因此，如何选一个合适的 K 值，是非常重要的问题。而在实践过程中，如何选择合适的 K 值，通过实验调参是一个很重要的步骤。一般来说，在实际应用中，选用一个比较小的数值，再采取交叉验证法来选取最优的 K 值。

交叉验证法

什么是交叉验证法呢？一般情况下，当给定的样本数据充足时，我们通过随机的把训练集切分成训练集、验证集和测试集来进行模型选择。其中训练集用于训练模型，验证集用于模型的选择，测试集用于最终对学习方法的评估。

但是实际应用中往往没有这样的理想条件，数据经常是不充足的，这时候就需要用到交叉验证法。交叉验证法的基本思想是重复地使用数据，把给定的数据进行切分，再把切分出的数据集组合成训练集和测试集，在此基础上反复进行训

练、测试和模型选择。

例如：简单交叉验证法就是随机将数据分成两部分，一部分（如 70%）作为训练集；另一部分（如 30%）作为测试集；然后用训练集在各种条件（例如不同的参数）下进行模型的训练，从而得到不同的模型；再在测试集上评价各个模型的测试误差，选出误差最小的模型。另外还有 S 折交叉验证和留一交叉验证，其基本思路也是这样的。

2. 最邻近的度量

k 近邻算法是在训练数据集中找到与该实例最邻近的 K 个实例，这 K 个实例的多数属于某个类，我们就说预测点属于哪个类。那么，距离有多少才算是最邻近呢？常规情况下，我们只需要采用欧几里得距离来衡量两个点之间的距离就可以了，然后把待分类的点到训练集中其他所有的点的欧几里得距离做一个排列，取最小值，就可以得到最邻近的结果。

欧几里得距离：欧几里得距离或欧几里得度量是欧几里得空间中两点间“普通”（即直线）距离。在欧几里得空间中，点 $x=(x_1,\ldots,x_n)$和 $y=(y_1,\ldots,y_n)$之间的欧氏距离为

$$d(x,y)=\sqrt{(x_1-y_1)^2+(x_2-y_2)^2+\cdots+(x_n-y_n)^2}$$

3. 多特征的归一化

在机器学习领域，不同的评价指标往往具有不同的量纲和单位。如果一个训练样本有两种特征值，且两种特征值的数值差异较大的时候，我们在度量距离时

候就会更偏向于数值较大的那个特征值，这是因为在这时候两个特征值并不是同等重要的。这样的情况会影响到数据分析的结果，因此就需要去进行数据标准化处理，来解决数据指标之间的可比性，在这一过程中，最典型的就是数据归一化处理。

举一个简单的例子，当前数据集中有五个数据，特征值是一个人的身高和鞋码，类别为男性和女性。这五个数据如下：A [(179,42),男] B [(178,43),男] C [(165,36)女] D [(177,42),男] E [(160,35),女]。此时再加入一个待分类的样本 F（167,43），对其进行预测，取 K=3。

采用欧拉距离计算 F 距离五个训练样本的欧氏距离并且取距离最近的三个，其中多数的类别就是我们所需的最终分类结果，计算如下：

$$AF = \sqrt{(167-179)^2 + (43-42)^2} = \sqrt{145}$$
$$BF = \sqrt{(167-178)^2 + (43-43)^2} = \sqrt{121}$$
$$CF = \sqrt{(167-165)^2 + (43-36)^2} = \sqrt{53}$$
$$DF = \sqrt{(167-177)^2 + (43-42)^2} = \sqrt{101}$$
$$EF = \sqrt{(167-160)^2 + (43-35)^2} = \sqrt{103}$$

由此得出，最近的三个样本分别是 C D E，由于 CE 为女性，D 为男性，那么预测结果就是女性。

然而，根据我们的常识，一个女性鞋码是 43 的概率远远小于一个男性鞋码是 43 的概率。出现这样的问题就是因为不同特征的量纲不同，导致了身高的重要

性远远超过了鞋码，这样就会导致预测结果出现偏差，同时这也就是我们要进行归一化的原因。

在实际操作中常用的归一化方法有：

1. 最大最小标准化（离差标准化）：

思路是把结果映射到[0,1]之间，其转换函数为

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

这种归一方法比较适合于数值集中的场合，但是当 max 和 min 值不稳定的时候，归一化结果不稳定，后续的使用效果也不稳定。实践过程中可以使用经验常量代替 max 和 min 值。

2. z-score 标准化

数据处理后符合标准正态分布，均值为 0，标准差为 1，其转换函数为:

$$x^* = \frac{x - \mu}{\sigma}$$

其中μ为所有样本数据的均值，σ为所有样本数据的标准差。这种归一方法要求原始数据的分布近似为高斯分布，否则的话效果非常差。

KNN 算法可以借助 Scikit-learn 包来直接进行调用，比较适合新手，可以尝试运行其 demo 研究算法的流程，例如基于鸢尾花数据集的分类: https://scikit-learn.org/stable/auto_examples/datasets/plot_iris_dataset.html#

当然，也可以考虑亲自研究一下源代码来加深对于 KNN 算法的理解。

四、 KNN 算法的优缺点

1. 优点

   非常简单且人性化 ,利于理解和实现 ;适合处理多分类问题 ,例如推荐算法。

2. 缺点

   时间复杂度较高，对样本的平衡度依赖性也高，当样本极端不平衡时候分类一定会出现偏差；并且可解释性比较差，不太适用于向量维度过高的场景。