K-Means 简介，前面说的 KNN 是分类算法

一句话讲一下什么是机器学习，通过算法引导模型训练数据使模型可用

举个例子讲一下什么算法是什么算法，模型是个啥

假如说你现在带着一个小孩逛公园，公园有很多人在遛狗。

现在你教这个小孩辨别哪个是狗。那这个小孩就是你要训练的模型。

你们走着走着就会有小动物迎面上来，你就告诉他哪个是，哪个不是，这个过程就是训练，也是小孩学习的过程。

上来的那些小动物就是你拿来训练的数据。

在这个训练，小孩学习的过程中，小孩会形成自己的认知模式，他自己会辨别哪个是狗哪个不是，等识别有了一定的准确度的时候，这个小孩就可以被称为一个可用模型。

根据这个例子大家对模型应该有个认识了。模型，简单来说可以是函数，函数根据已有的数据去完善自己的执行过程，比如，我给到模型 10000 组输入输出的数据集，模型根据数据集自己完善内部的参数，

等训练结束的时候，面对新的输入可以执行得到一个可能符合期望可能不符合的输出。

抽象一点，模型可以是一种认知模式，像之前辨别小动物的例子，如果模型再复杂一点，小孩可以根据动物的特征分辨猫，狗，鸟，松鼠等等，那这种认知模式就是模型。

模型讲清楚了，我们来看看机器学习中常见的算法。

算法就是引导模型利用数据进行学习的方法。

在上面的例子中，如果有一个小动物过来，小孩根据目前的认知模式做出判断并问你：这个小动物是狗吧？你回答是，他就知道自己判断对了，回答错他就知道自己的判断是有误的，也就是说这时你的回答可以帮他验证和纠错，在有人告诉模型样本的正确分类的时候，就是监督式学习。

非监督式学习就是相对的，我也举不出更好的例子。但是可以说的是，K 均值算法是一种非监督式学习。大家自己体会一下或者去网上找找更严谨一些的定义吧。

还有一种叫强化学习，因为今天讲到的这两个算法没有涉及到，而且我自己了解的也还不太多，就不误导大家了。

之前还有提到，K 均值算法是一种聚类算法，而学姐讲的 KNN 算法是一种分类算法，我把分类和聚类作为基础知识再唠叨一下哈。

分类算法是根据可以比较的特性或者特征来分配已确定的标签，而聚类算法是根据相似度将样本聚为几类，打个比方来解释一下，还是给小动物分类的例子。

如果现在要把面前的小动物分成猫，狗，鸟和松鼠几类的话，我可以根据叫声，身形，还有会不会飞，吃什么就进行简单的区分，然后给每一个待处理的样本贴上一个明确的标签；而另一种情况，比方说现在有一堆动物，我把有羽毛的归为一簇，吃荤的归为一簇，然后卵生的归为一簇，我并不关心类别具体是什么，但是我知道同类的样本具有一定的相似度。

分类的目的是利用分类器将信息映射成一个具体的类别；而聚类的目的是找出样本的相似之处。

分类是一种典型的带有监督的学习，聚类是非监督式的。具体怎么解释监督式和非监督式我还不太行，但是在分类和聚类问题上，对于分类，每个样本都是带有标签的，也就是说每一个样本都有一个正确答案的，而聚类不是这样的。

终于切入正题了哈哈，我们说一说算法的具体步骤，前面在说主要思想的时候已经概括了，在这看一看算法的细节，另外再举两个例子给大家理解一下。

K 均值算法第一步先要确定初始质心，在初始质心的基础上开始迭代。初始质心的选择会直接影响到聚类算法最后的效果，等会有一个聚类结果的对比，我们先看完这些步骤。

K 值决定了我们最后要聚成几类，这个 K 值是要预先设置好的，要求我们对数据集有一定的认识，可以是根据经验，也可以是根据事先试验的结果。这个 K 值也会影响到聚类的效果。

这就是算法在不同初始质心下的聚类效果。

左边这张图在结果上是更加符合我们预期的，根据我们肉眼的观察和直觉，大致也就是分类成这么个情况；而右边的图很明显超出我们的预料，但是从算法的执行过程上来看，又都是正确的。

对于这一现象的解释是说，因为在计算聚类质心的时候用的是平均值，然后我们也清楚，平均值是很容易受到异常点的影响的，所以如果真的不加限制的随机选择初始节点，就有可能因为初始节点太离谱导致这种结果。这是算法本身就有的缺陷，当然后面会介绍到一些方法针对缺陷来进行改进。

在这之后还有一张图，我们可以到散点最后聚成了 3 个类，显然这也是不符合我们预期的，导致这种结果既有可能是因为初始 K 值没设置好，也有可能是初始质心点选择的不合适。

----------切回去----------

下一步是计算相似度和更新各个类的质心。

K 均值算法用距离来代表相似度，这个距离其实是不仅限于欧式距离的，只是用其他的距离来代替欧式距离的话不能保证最后一定可以收敛，但是可以的到 K 均值算法的一些变体，比如说球体 K 均值算法和 K 中心点算法……

根据相似度完成聚类之后用新的质心代替原来的质心，最后重复这两个步骤直到质心不再发生变化就可以得到各个簇的质心点以及知道每个样本点分别属于哪个簇。

这就是 K 均值算法的所有步骤。下面有一个比较完整的流程。

大家看一下知道这个流程是怎么回事儿就好了。

先是初始化，然后随机确定初始质心点，之后计算相似度，重新计算质心点，迭代，出结果。

这么来看的话，K 均值算法还是不太难理解的。

然后我们来简单介绍一下 K-Means 算法的一些有意思的应用。

首先是可以用 K-Means 算法来实现图像压缩。
图像压缩就跟矩阵压缩很像，对与一个稀疏矩阵，我们只存储它有含义的部分，剩余的部分用很小的一部分空间就可以记录了。图像压缩也是一样的，将冗余的信息去掉，或是简单存储。

比如说这张图，两只老鼠的颜色，背后的墙，地板的颜色，这些大块面积的颜色都相同，就是我们压缩的对象。又或者说下面这是动画连续的几帧，在这连续的几帧中动画的背景甚至人物大面积的外形都没有变化，我们就可以在这上面进行压缩存储。

大家如果对图像压缩比较感兴趣的话可以去网上了解一下，我觉得还挺有意思的。

用 K-Means 算法如果可以实现精确的聚类，我们就可以把上述冗余的信息用很小的空间解决掉了。

K-Means 算法还可以用作其他算法的预处理。比如说像 KNN 这样的分类算法一般要求事先有几个已经确定的标签也就是确定的类，在要求没那么高的情况下，我可以先对已有的数据用 K 均值做预处理，然后再把新的数据导入进来做预测。

第三个应用其实我也不知道应该怎么来称呼，就是上面讲的 K 均值算法看起来还是蛮简单的，就是二位平面上对坐标点的一些操作，事实上当涉及到高维度空间的时候，人眼就派不上用场，只能用计算机来计算了。

最后对 K-Means 的优缺点做个简单的总结吧

首先 K 均值算法易于理解，而且简单易实现；

相比于 K 近邻算法，K 均值对存储空间的要求不大；

另外就是 K 均值对给的样本点是没有要求的，所有的点一开始就是无序的；

它的局限性在于，之前说过了，对异常值，也就是孤立数据点或者说噪声比较敏感；

然后就是这个聚类类别的个数是需要我们提前设置的，可能对于一个数据集我们也不清楚聚成几个类别合适，所以可能要求我们有一些先验知识；

上面这两点其实都是导致算法产生的不是全局最优解，而是局部最优解。

最后，这个算法其实是可以改进的，而且改进主要是针对初始点选择的方法进行改进的。

有一个很常见的就是 K-Means++。他的大致思路就是在选择的时候尽可能地让初始中心点相距较远，大家有兴趣的话可以去了解一下具体的操作。