### K-NN算法

#### 3.2.1.K-NN算法简介

K-NN算法 ( K Nearest Neighbor, K近邻算法 ), 是机器学习中的一个经典算法, 比较简单且容易理解. K-NN算法通过计算新数据与训练数据特征值之间的距离, 然后选取 K (K>=1) 个距离最近的邻居进行分类或者回归. 如果K = 1 , 那么新数据将被分配给其近邻的类.

K-NN算法是一种有监督学习, K-NN算法用于分类时, 每个训练数据都有明确的label, 也可以明确的判断出新数据的label, K-NN用于回归时也会根据邻居的值预测出一个明确的值.

#### 3.2.2.K-NN算法的过程

1.选择一种距离计算方式, 通过数据所有的特征计算新数据与已知类别数据集中数据点的距离;

2.按照距离递增次序进行排序, 选取与当前距离最小的 k 个点;

3.对于离散分类, 返回 k 个点出现频率最多的类别作为预测分类; 对于回归, 返回 k 个点的加权值作为预测值.

#### 3.2.3.K-NN算法的关键

　K-NN 算法的理论和过程都很简单, 但有几个关键点需要特别注意.

##### (1).数据特征的量化

如果数据特征中存在非数值型的特征, 则需要采取手段将其量化为数值. 举个例子，若样本特征中包含颜色（红黑蓝）特征, 由于颜色之间没有距离可言, 所以可以通过将颜色转换为灰度值来实现距离计算. 另外, 一般样本有多个参数, 每个参数都有自己的定义域和取值范围, 因而它们对距离计算的影响也就不一样. 比如取值范围较大的参数影响力会盖过取值较小的参数. 所以, 为了公平起见, 样本参数必须做一些scale处理, 最简单的方式就是将所有特征的数值都采取归一化处理.

##### (2).计算距离的方法

距离的定义有很多种, 如欧氏距离, 余弦距离, 汉明距离, 曼哈顿距离等. 通常情况下，对于连续变量, 选取欧氏距离作为距离度量; 对于文本分类这种非连续变量, 选取汉明距离来作为度量. 通常如果运用一些特殊的算法来作为计算度量, 可以显著提高 K 近邻算法的分类精度, 如运用大边缘最近邻法或者近邻成分分析法.

##### (3).确定 K 值

　　K是一个自定义的常数, 它的值会直接影响最后的预测结果. 一种选择K值的方法是, 使用 cross-validate（交叉验证）误差统计选择法**.**交叉验证就是把数据样本的一部分作为训练样本, 另一部分作为测试样本. 比如选择95%作为训练样本, 剩下的用作测试样本, 通过训练数据集训练出一个机器学习模型, 然后利用测试数据测试其误差率. cross-validate（交叉验证）误差统计选择法就是比较不同K值时的交叉验证平均误差率, 选择误差率最小的那个K值. 例如选择K=1, 2, 3, ... ,  对每个K = i 做100次交叉验证, 计算出平均误差, 通过比较选出误差最小的那个.

#### 3.2.4.K-NN分类与K-NN回归

##### (1).K-NN分类

　　如果训练样本是多维特征空间向量, 其中每个训练样本都有一个类别标签（喜欢或者不喜欢、保留或者删除）. 分类算法常采用 " 多数表决 " 决定, 即k个邻居中出现次数最多的那个类作为预测类. “ 多数表决 ” 分类的一个缺点是出现频率较多的样本将会主导测试点的预测结果, 因为它们出现在测试点的K邻域的几率较大, 而测试点的属性又是通过K领域内的样本计算出来的. 解决这个缺点的方法之一是在进行分类时将K个邻居到测试点的距离考虑进去. 例如, 样本到测试点距离为d, 则选1/d为该邻居的权重（也就是得到了该邻居所属类的权重）, 然后统计出k个邻居所有类标签的权重和, 值最大的那个就是新数据点的预测类标签。  
　　举例，K=5, 计算出新数据点到最近的五个邻居的举例是（1, 3, 3, 4, 5）, 五个邻居的类标签是（yes, no, no, yes, no）. 如果按照多数表决法, 则新数据点类别为no（3个no, 2个yes）; 若考虑距离权重, 则类别为yes（no:2/3+1/5, yes:1+1/4）.

##### (2).K-NN回归

　　数据点的类别标签是连续值时应用K-NN算法就是回归, 与K-NN分类算法过程相同, 区别在于对K个邻居的处理上. K-NN回归是取K个邻居类标签值得加权作为新数据点的预测值. 加权方法有: K个近邻的属性值的平均值（最差）, 1/d为权重（有效的衡量邻居的权重, 使较近邻居的权重比较远邻居的权重大）, 高斯函数（或者其他适当的减函数）.

#### 3.2.5.K-NN总结

　　K-近邻算法是分类数据最简单最有效的算法, 其学习基于实例, 使用算法时我们必须有接近实际数据的训练样本数据. K-近邻算法必须保存全部数据集, 如果训练数据集的很大, 则会占用大量的存储空间. 此外, 由于必须对数据集中的每个数据计算距离值, 实际使用时可能非常耗时. k-近邻算法的另一个缺陷是它无法给出任何数据的基础结构信息, 因此我们也无法知晓平均实例样本和典型实例样本具有什么特征.