KNN原理小结

KNN既可以做回归除了最后一个一般是多数表决法，一个一般是选择平均法。scikit-learn只使用了蛮力法(brute-force)KD树实现(kD-Tree)和球树(ball tree)

① KNN算法的三要素.

三个最originate的要素：K值的选取，距离度量的方式和分类决策规则。(分类决策规则一般用多数表决法，重点关注K值的选择和距离的度量方式。

K值的选取，没有固定的经验，一般根据样本的分布，选择一个较小的值，可以通过交叉验证来选择一个合适的K值。 K值左小，相似的训练实例对结果取作用，泛化误差大，K值小⇒模型复杂，容易发生过拟合。

K值左大，整体模型变得简单。泛化误差减小，训练误差增大。

距离的度量：常用的是欧式距离：对于2个n维向量 $x(x_1,\cdots x_n)$ 和 $y(y_1,\cdots,y_n)$ $D(x,y)=\sqrt{(x_1-y_1)^2+(x_2-y_2)^2+\cdots+(x_n-y_n)^2}$

② 曼哈顿距离 $|x_1-y_1|+|x_2-y_2|+\cdots+|x_n-y_n|=\sum_{i=1}^{n}|x_i-y_i|$ $=\sqrt{\sum_{i=1}^{n}(x_i-y_i)^2}$

③ 通用闵可夫斯基距离 $\sqrt[p]{|x_1-y_1|^p+|x_2-y_2|^p+\cdots+|x_n-y_n|^p}=\sqrt[p]{\sum_{i=1}^{n}(|x_i-y_i|)^p}$

② KNN的算法蛮力实现.

计算预测样本与训练集中样本的距离，然后计算最小的K个距离即可，接着多数表决。在特征数多，样本量大的时候，算法时间效率很成问题

③ KNN算法之KD树实现原理.

KD树没有一开始就对测试样本分类，而是先对训练集建模，建立的模型就是KD树。建好模型对test data做预测。这里的模型就是KD树：KD树中的K代表样本的特征维度。不是KNN中最近的K个最近样本的意思。后面用n表示

KD tree三步：建树、搜索最近邻、预测。

KD树建树：KD树是二叉树，每个节点有[特征维数，切分轴，指向左枝的指针，指向右枝的指针]

切分轴 r $1\le r\le n$ (n为特征维度的数量)

对每一个维度计算方差，方差大的那一维为 r，取第 r 维的中位数，假如是 $x_r^{(p)}$ 则 $x_r^{(i)}<x_r^{(p)}$ 分到左边，对 $x_r^{(i)}>x_r^{(p)}$ 分到右边，(若有m个样本，m为偶数，则取中位左边或右边。

搜索最近邻：对测试集P，把测集P放在叶子结点上，持续向上爬，当到 其它点和切割轴的距离 也挺复杂的，具体算法见聚完一kd树详解篇。