统计学习方法

第三章 k近邻法（KNN）

郑琦斌

2018.10.26

第三章 k近邻法

3.1 基本介绍

- k近邻法是一种基本的分类和回归方法
- 简单描述：
  - 问题：对一个类别未知的样本点A进行分类
  - 方法：
    - 从一个已知分类的数据集中，找出距离A最近的k个点
    - 依据这k个点的大多数类别，作为A的类别

3.1 基本介绍

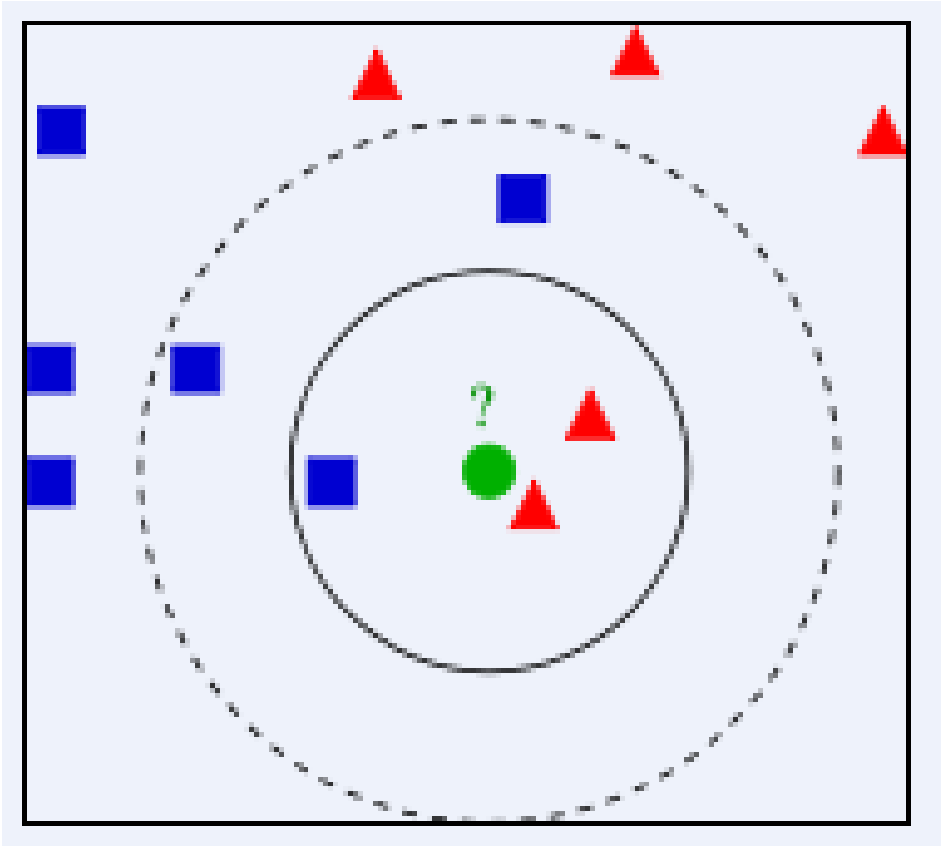
举例

右图说明：

1. 二维空间中的点
2. 数据集包括红蓝两种类别的点
3. 试得到未知点（绿色）的类别

当k=3时，判定未知点为红类；

当k=5时，判定未知点为蓝类。



第三章 k近邻法

3.2 k近邻算法

- 输入：　1. 训练数据集

$$T=\{(x_1,y_1),(x_2,y_2),\dots,(x_N,y_N)\},$$

其中 $x_i=\left(x_i^1,x_i^2,\dots,x_i^{\mathrm{n}}\right),\quad y_i\in\{c_1,c_2,\dots,c_{\mathrm{K}}\}$

　2. 新实例x

- 输出：　x对应的y

3.2 k近邻算法

算法：

1．根据给定的距离度量，在训练集T中找出与x最邻近的k个点，涵盖这k个点的x的领域记作$N_k(x)$；

2．在$N_k(x)$中根据分类决策规则（如少数服从多数），决定x的类别y。

$$y = argmax_{cj} \sum_{xi \in N_k(x)} I(yi = cj)$$

其中，$i = 1,2,\ldots,N$；$j = 1,2,\ldots,K$

第三章 k近邻法

3.1　基本介绍

3.2　k近邻算法

3.3　k近邻模型和三要素

3.4　k近邻实现——kd树

3.3 k近邻模型

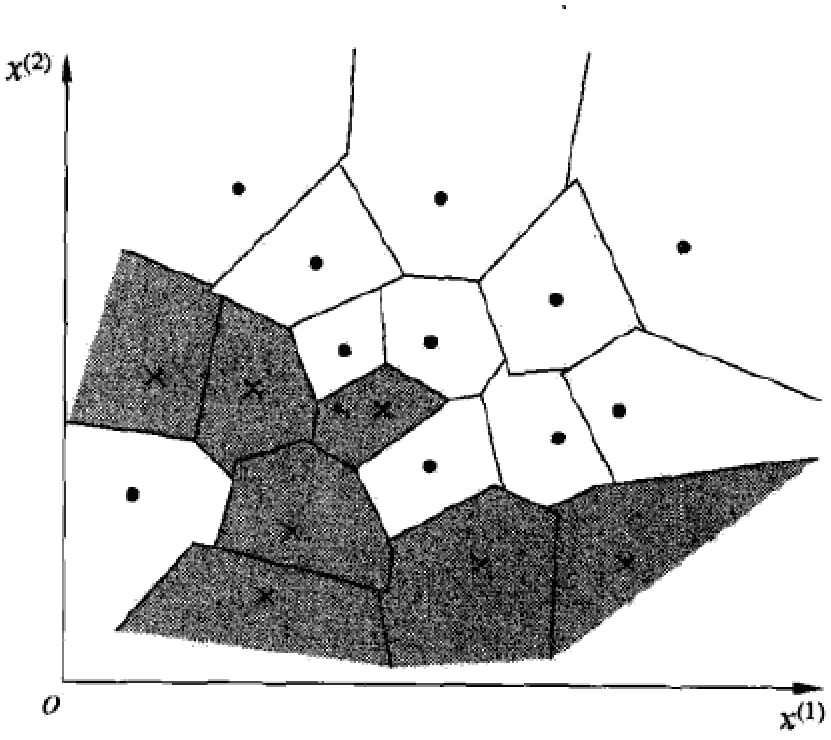
- 模型的三要素
  - 距离度量
  - k值选择
  - 分类决策规则
- k近邻模型由三要素决定，当三要素和训练集确定时，实例对应的类别也就确定

3.3 k近邻模型

- k近邻模型实际上对应于对特征空间的划分

右图说明：

1. 二维空间
2. 最近邻法（k=1）的划分



3.3.1 距离度量

- 距离反应的是两个实例相似的程度
- 其他的度量？
  - Lp距离
- 由不同的距离度量得到的k邻近点是不一样的

3.3.1 距离度量

- Lp距离公式：$L_p(x_i, x_j) = \left(\sum_{l=1}^{n} |x_i^{(l)} - x_j^{(l)}|^p\right)^{\frac{1}{p}}$，p ≥ 1
- 当p = 1，为曼哈顿距离，
  - $L_1(x_i, x_j) = \sum_{l=1}^{n} |x_i^{(l)} - x_j^{(l)}|$
- 当p = 2，为欧式距离，
  - $L_2(x_i, x_j) = \left(\sum_{l=1}^{n} |x_i^{(l)} - x_j^{(l)}|^2\right)^{\frac{1}{2}}$

3.3.1 距离度量

- 当$p = \infty$，可以证明，
  - $L_\infty(x_i, x_j) = \max_l |x_i^{(l)} - x_j^{(l)}|$

3. 3. 1 距离度量

- 证明： $L_\infty(x_i, x_j) = \max_l |x_i^{(l)} - x_j^{(l)}|$

$L_\infty(x_i, x_j) = \lim_{p\to\infty} \left(\sum_{l=1}^{n} |x_i^{(l)} - x_j^{(l)}|^p\right)^{\frac{1}{p}}$ ，

记 $\mathrm{M} = \max_l |x_i^{(l)} - x_j^{(l)}|$，则 $\mathrm{M}^p \le \sum_{l=1}^{n} \left|x_i^{(l)} - x_j^{(l)}\right|^p \le nM^p$ ，

$\therefore \mathrm{M} \le L_p(x_i, x_j) \le \sqrt[p]{n} * M$,

当 $p \to \infty$时，$\mathrm{M} \le L_p(x_i, x_j) \le M$,

$\therefore L_\infty(x_i, x_j) = \mathrm{M} = \max_l |x_i^{(l)} - x_j^{(l)}|$

3.3.2 k值的选择

- 当k值较小时，代表从较小的领域中进行预测，近似误差较小，但估计误差较大
- 当k值较大时，代表从更大的领域中预测，近似误差较大，估计误差较小
- 原因：模型的复杂程度、过拟合的问题

3.3.2 k值的选择

- 当k等于N时，不好，因为全部选择，没有考虑距离远近。
- 采用交叉验证法选取最优的k值。

3.3.3 分类决策规则

- 一般采用多数表决
- 多数表决的合理解释
  - 对于x，假设涵盖$N_k(x)$区域的类别为C_j，那么误分类率是：

$$\frac{1}{k}\sum_{x_i \in N_k(x)} I(y_i \neq c_j) = 1 - \frac{1}{k}\sum_{x_i \in N_k(x)} I(y_i = c_j)$$

  - 要使误分类率（经验风险）最小化，则选择多数表决。

第三章 k近邻法

3.4. k近邻实现——kd树

- 问题：如何找到k个近邻点？
- 方法1：线性扫描
  - 方法1存在的问题：需要计算到训练集中所有点的距离。当训练集与特征数量很多时，计算开销很大。
- 方法2：使用kd树
  - kd树的平均搜索效率为$O(\log N)$。

3.4.1 kd树的构造

- 输入：　　k维空间数据集

$$T=\{x_1,x_2,\dots,x_N\},$$

其中 $x_i=\left(x_i^1,x_i^2,\dots,x_i^k\right)^T,i=1,2,\dots,N$

- 输出：　　kd树

3.4.1 kd树的构造

- 算法

1. 构造根结点。根结点对应于包含T的k维空间的超矩形区域。

2. 切分。选择$x^{(1)}$为坐标轴，以T中所有实例的$x^{(1)}$坐标中位数为切分点，将区域一分为二。

3. 生成深度为1的左右子结点。子结点对应于两个子区域。

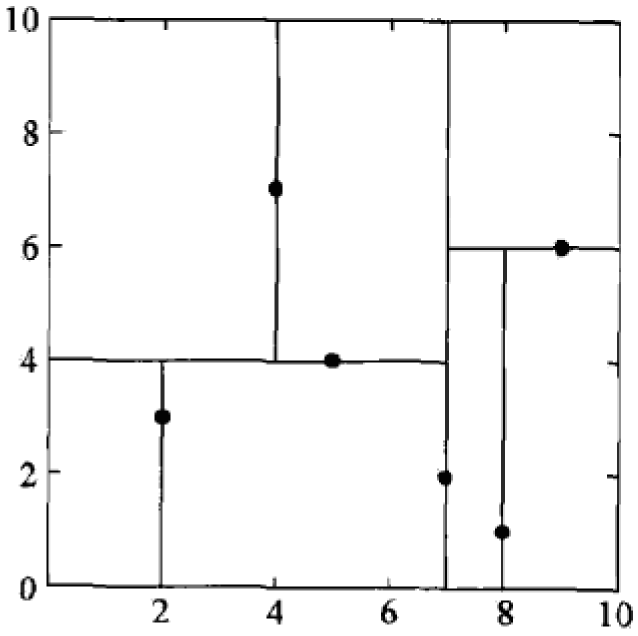
4. 将切分点实例保存在根结点。

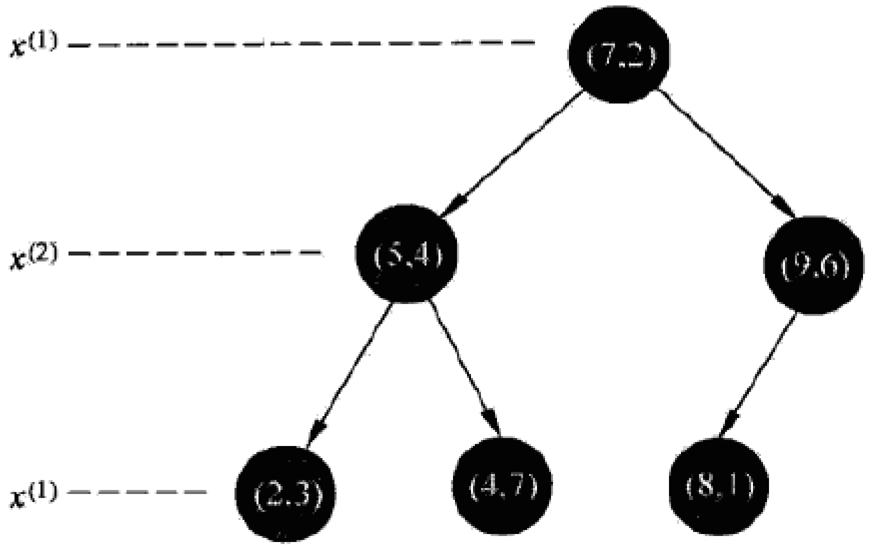
5. 重复生成下层子结点。

深度为j的结点，选择$x^{(l)}$为切分坐标轴，$l = j(mod k) + 1$

3.4.1 kd树的构造

$$T = \{(2,3)^T, (5,4)^T, (9,6)^T, (4,7)^T, (8,1)^T, (7,2)^T\}$$





3.4.1 kd树的构造

- 对于kd树的理解
  - kd树是二叉树，左右子树表示对k维空间的一个划分
  - 通常依次选择坐标轴，并以中位数作为切分点。这样得到的kd树是平衡的。

3.4.2 kd树的搜索

- 算法
  1. 找出包含目标点x的叶结点

     从根结点出发，递归地向下访问到kd树。若目标点x当前维的坐标小于切分点的坐标，则移动到左子结点，否则移动到右子结点。直到子结点为叶结点为止。
  2. 以此叶结点为“当前最近点”
  3. 递归的向上回退，在每个结点进行以下操作：

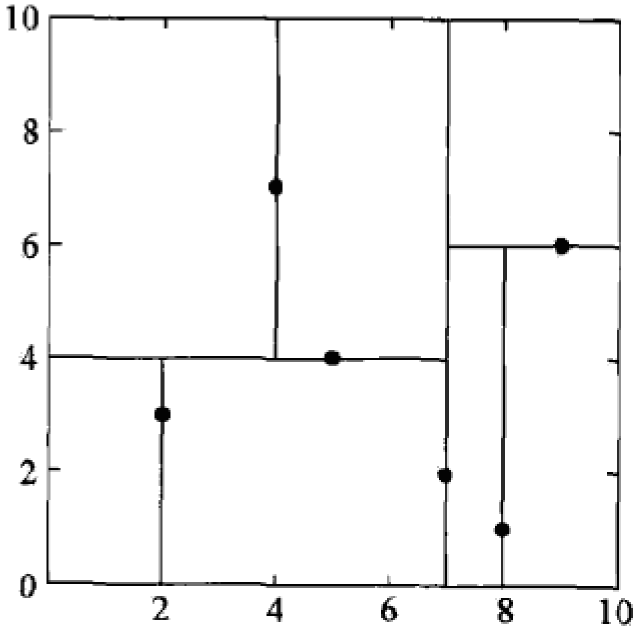
     a）如果该结点保存的实例点比当前最近点更近，则以该实例点为当前最近点。

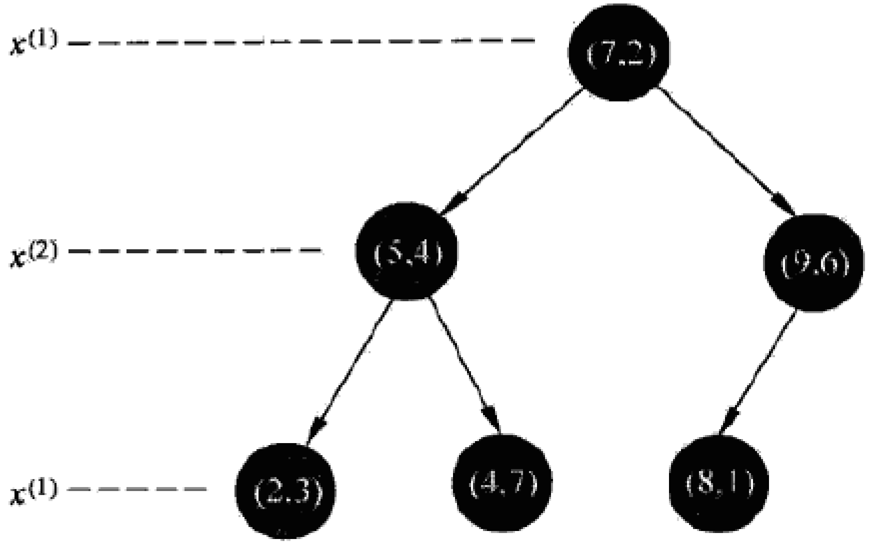
     b）检查另一子结点对应的区域是否相交。如果相交，递归搜索。
  4. 当回退到根结点时，搜索结束。

3.4.2 kd树的搜索

$$T = \{(2,3)^T, (5,4)^T, (9,6)^T, (4,7)^T, (8,1)^T, (7,2)^T\}$$

$x_1 = (2.1, 3.1)$，$x_2 = (2, 4.5)$





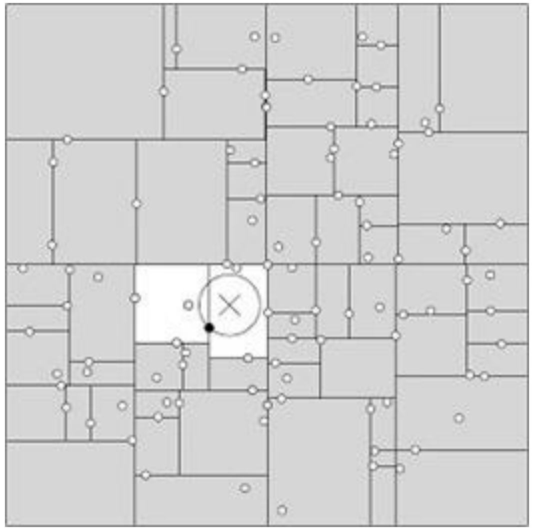
3.4.3 kd树的效率

- 研究表明N个节点的k维k-d树搜索过程时间复杂度为：

  $t_{worst}=O(kN^{1-1/k})$。

- 一般来说要求数据的规模N满足$N \gg 2^k$，才能达到高效的搜索。

查找效率好



查找效率坏