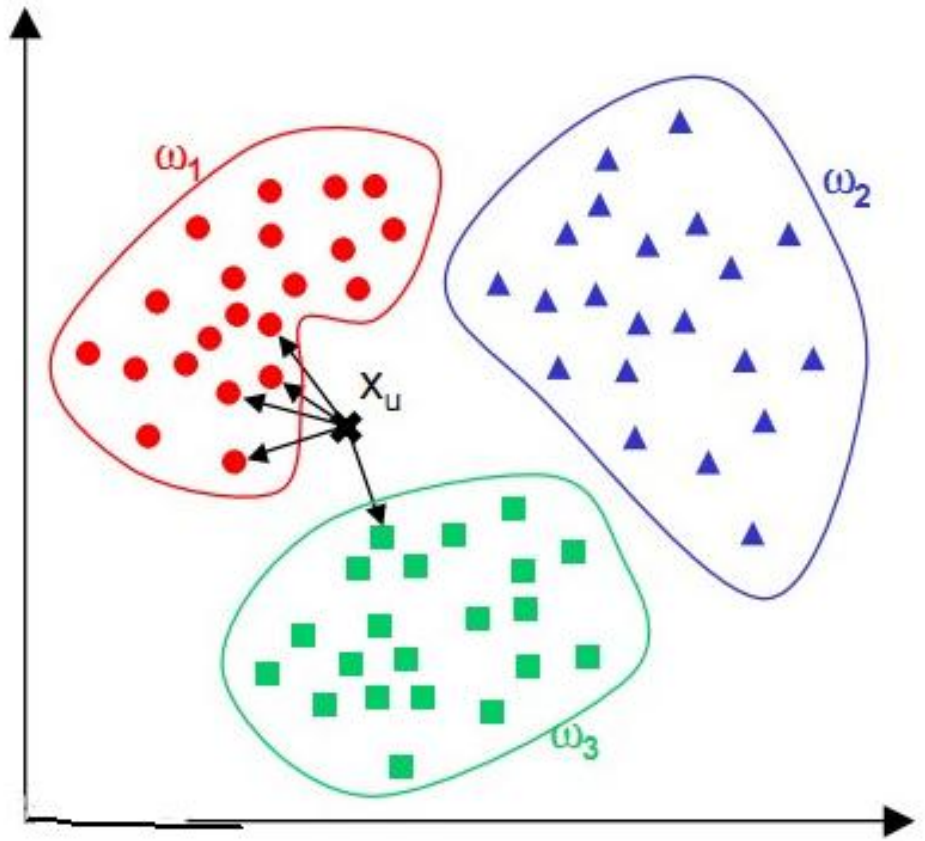
人工智能之机器学习

K近邻算法(KNN)

主讲人：李老师

课程内容

- KNN算法
- KD-Tree



KNN直观解释


迟到党
睡神
LOL队友
深藏功与名
考神
女神
学霸
忧伤少年
妹子区
伪学霸区

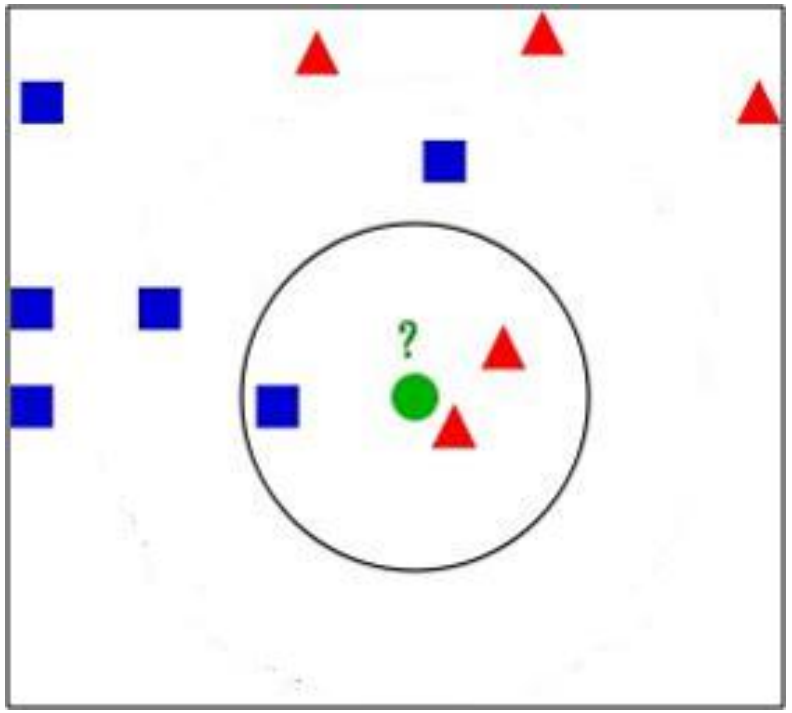

讲 台
左护法
右护法
户外个人VIP休闲场所
至尊VIP清凉楼道专区
高级阳光SPA专区
学霸区
VIP休息
VIP娱乐
高级避暑VIP专区
免费苍蝇按摩区
超级VIP休闲专区

KNN算法原理

- K近邻(K-nearst neighbors, KNN)是一种基本的机器学习算法，所谓k近邻，就是k个最近的邻居的意思，说的是每个样本都可以用它最接近的k个邻居来代表。比如：判断一个人的人品，只需要观察与他来往最密切的几个人的人品好坏就可以得出，即“近朱者赤，近墨者黑”；KNN算法既可以应用于分类应用中，也可以应用在回归应用中。
- KNN在做回归和分类的主要区别在于最后做预测的时候的决策方式不同。KNN在分类预测时，一般采用**多数表决法**；而在做回归预测时，一般采用**平均值法**。

KNN算法原理

- 1. 从训练集合中获取K个离待预测样本距离最近的样本数据；
- 2. 根据获取得到的K个样本数据来预测当前待预测样本的目标属性值。



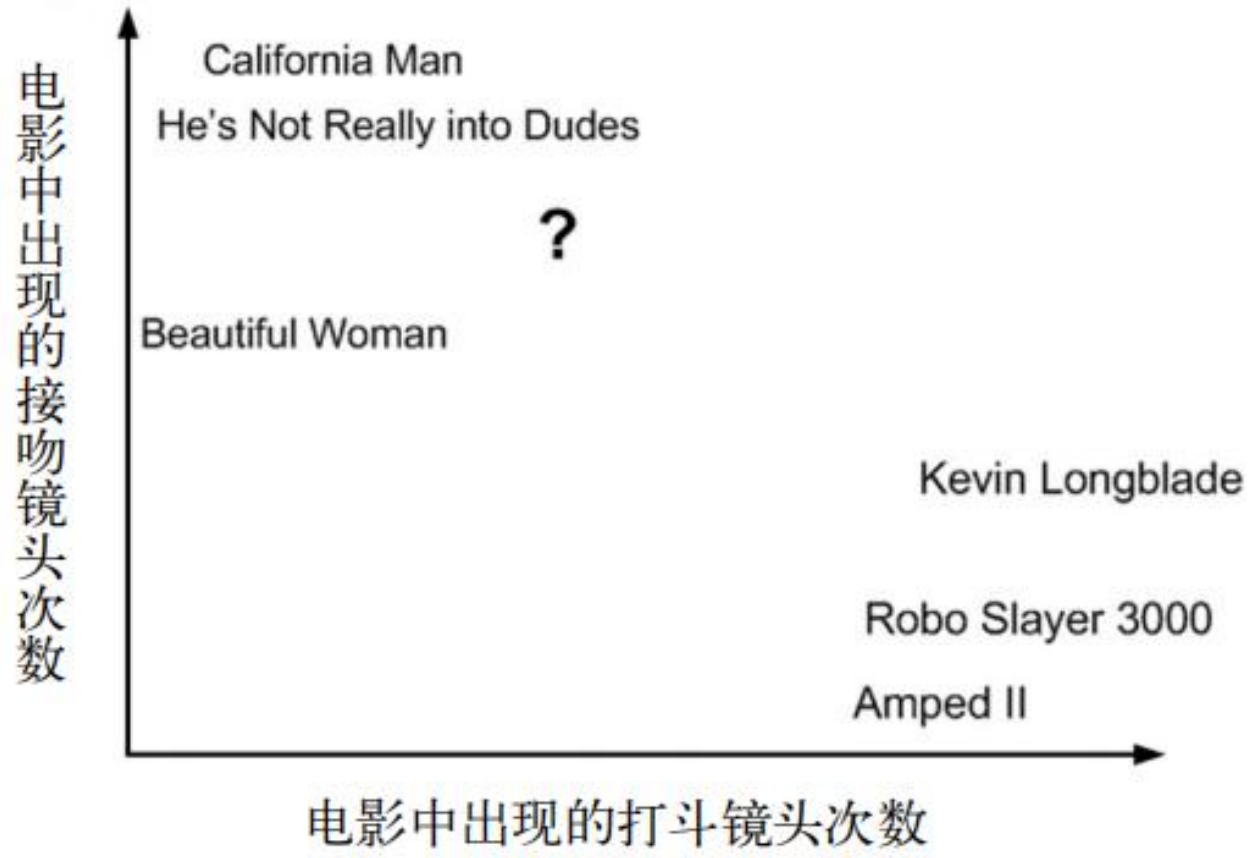
一个案例了解KNN

电影名称	打斗镜头	接吻镜头	电影类型
California Man	3	104	爱情片
He's Not Really into Dudes	2	100	爱情片
Beautiful Woman	1	81	爱情片
Kevin Longblade	101	10	动作片
Robo Slayer 3000	99	5	动作片
Amped ll	98	2	动作片
?	18	90	未知

- 我们标记电影的类型：爱情片，动作片
- 每个电影有两个特征属性：打斗镜头，接吻镜头
- 预测一个新的电影的电影类型

一个案例了解KNN

- 第一步：将训练集中的所有样例画入坐标系，也将待测样例画入



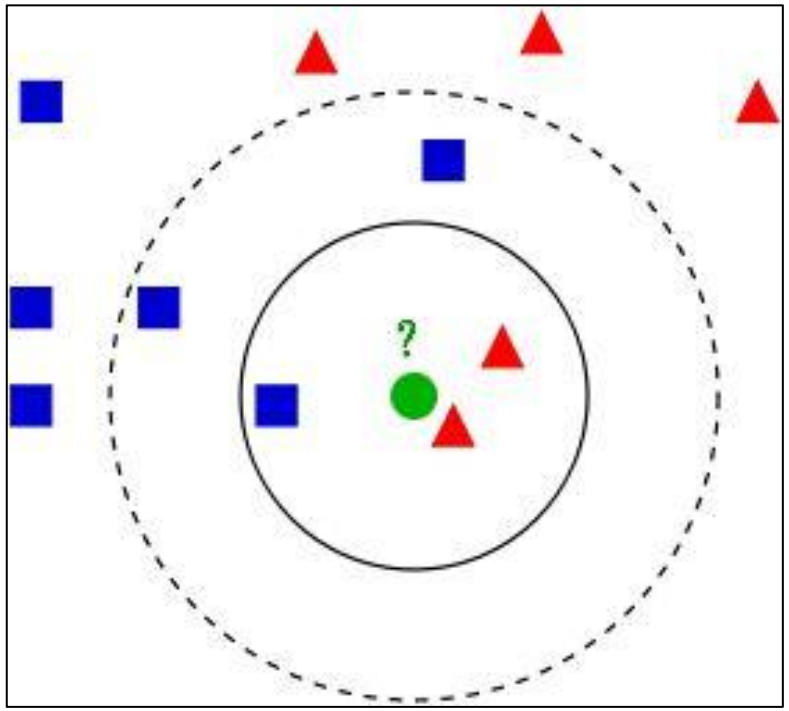
一个案例了解KNN

- 第二步：计算待测分类的电影与所有已知分类的电影的欧式距离

电影名称	与未知电影的距离
California Man	20.5
He's Not Really into Dudes	18.7
Beautiful Woman	19.2
Kevin Longblade	115.3
Robo Slayer 3000	117.4
Amped ll	118.9

- 第三步：将这些电影按照距离升序排序，取前k个电影，假设k=3，那么我们得到的电影依次是《He's Not Really Into Dudes》、《Beautiful Woman》和《California Man》。而这三部电影全是爱情片，因此我们判定未知电影是爱情片。

KNN三要素



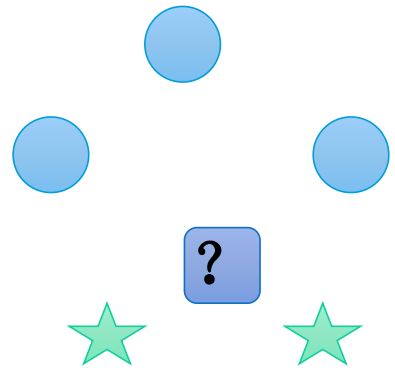
- 如左图中，绿色圆要被决定赋予哪个类，是红色三角形还是蓝色四方形?
- 如果K=3，由于红色三角形所占比例为2/3，绿色圆将被赋予红色三角形那个类；
- 如果K=5，由于蓝色四方形比例为3/5，因此绿色圆被赋予蓝色四方形类。

KNN三要素

- 在KNN算法中，非常重要的主要是三个因素：
  - **K值的选择**：对于K值的选择，一般根据样本分布选择一个较小的值，然后通过交叉验证来选择一个比较合适的最终值；当选择比较小的K值的时候，表示使用较小领域中的样本进行预测，训练误差会减小，但是会导致模型变得复杂，容易过拟合；当选择较大的K值的时候，表示使用较大领域中的样本进行预测，训练误差会增大，同时会使模型变得简单，容易导致欠拟合；
  - **距离的度量**：一般使用欧氏距离(欧几里得距离)；
  - **决策规则**：在分类模型中，主要使用多数表决法或者加权多数表决法；在回归模型中，主要使用平均值法或者加权平均值法。

KNN分类预测规则

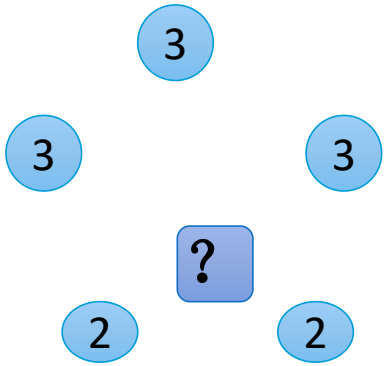
- 在KNN分类应用中，一般采用多数表决法或者加权多数表决法。
- **多数表决法**：每个邻近样本的权重是一样的，也就是说最终预测的结果为出现类别最多的那个类，比如右图中蓝色圆圈的最终类别为红色；



- **加权多数表决法**：每个邻近样本的权重是不一样的，一般情况下采用权重和距离成反比的方式来计算，也就是说最终预测结果是出现权重最大的那个类别；比如右图中，假设三个红色点到待预测样本点的距离均为2，两个黄色点到待预测样本点距离为1，那么蓝色圆圈的最终类别为黄色。

KNN回归预测规则

- 在KNN回归应用中，一般采用平均值法或者加权平均值法。
- **平均值法**：每个邻近样本的权重是一样的，也就是说最终预测的结果为所有邻近样本的目标属性值的均值；比如右图中，蓝色圆圈的最终预测值为：2.6；



- **加权平均值法**：每个邻近样本的权重是不一样的，一般情况下采用权重和距离成反比的方式来计算，也就是说在计算均值的时候进行加权操作；比如右图中，假设上面三个点到待预测样本点的距离均为2，下面两个点到待预测样本点距离为1，那么蓝色圆圈的最终预测值为：2.43。(权重分别为: 1/7和2/7)

编程——经典面试题KNN的实现

- 使用Python的二维List对KNN进行实现（使用等权投票），然后对未知影片的类型进行预测

电影名称	打斗镜头	接吻镜头	电影类型
California Man	3	104	爱情片
He's Not Really into Dudes	2	100	爱情片
Beautiful Woman	1	81	爱情片
Kevin Longblade	101	10	动作片
Robo Slayer 3000	99	5	动作片
Amped ll	98	2	动作片
?	18	90	未知

思考

- 训练KNN模型需要进行特征标准化吗?

KNN算法实现方式

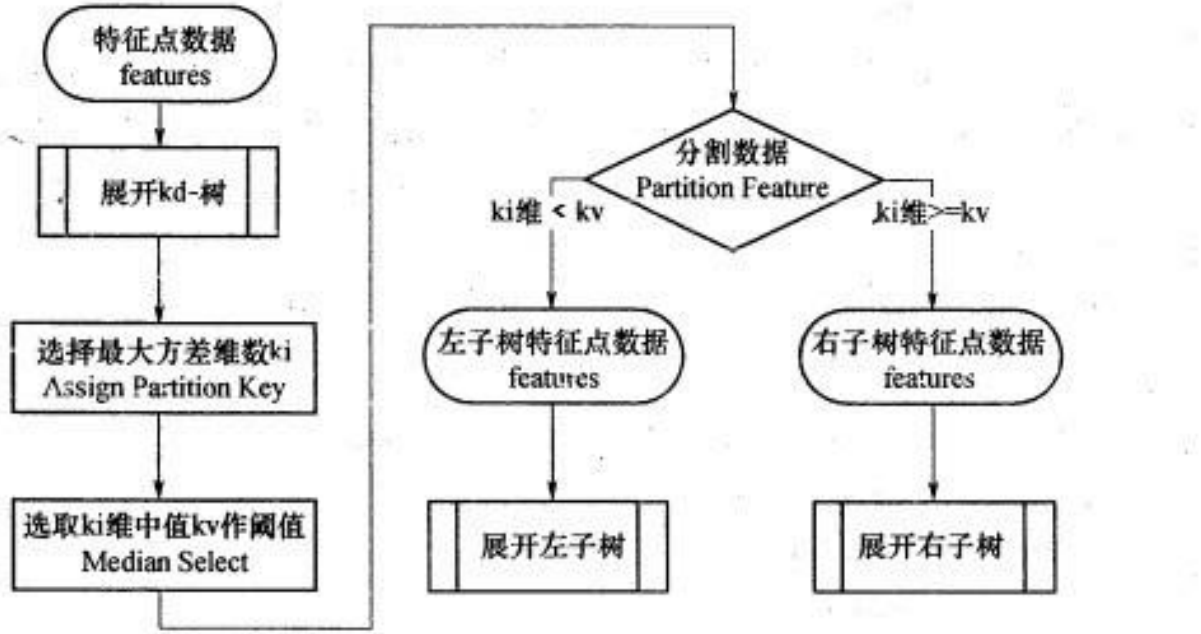
- KNN算法的重点在于找出K个最邻近的点，主要方式有以下几种：
  - **蛮力实现(brute)**：计算预测样本到所有训练集样本的距离，然后选择最小的k个距离即可得到K个最邻近点。缺点在于当特征数比较多、样本数比较多的时候，算法的执行效率比较低;
  - **KD树(kd_tree)**：KD树算法中，首先是对训练数据进行建模，构建KD树，然后再根据建好的模型来获取邻近样本数据。
- 除此之外，还有一些从KD_Tree修改后的求解最邻近点的算法，比如：Ball Tree、BBF Tree、MVP Tree等。

KD Tree

- KD Tree是KNN算法中用于计算最近邻的快速、便捷构建方式。
- 当样本数据量少的时候，我们可以使用brute这种暴力的方式进行求解最近邻，即计算到所有样本的距离。但是当样本量比较大的时候，直接计算所有样本的距离，工作量有点大，所以在这种情况下，我们可以使用kd tree来快速的计算。

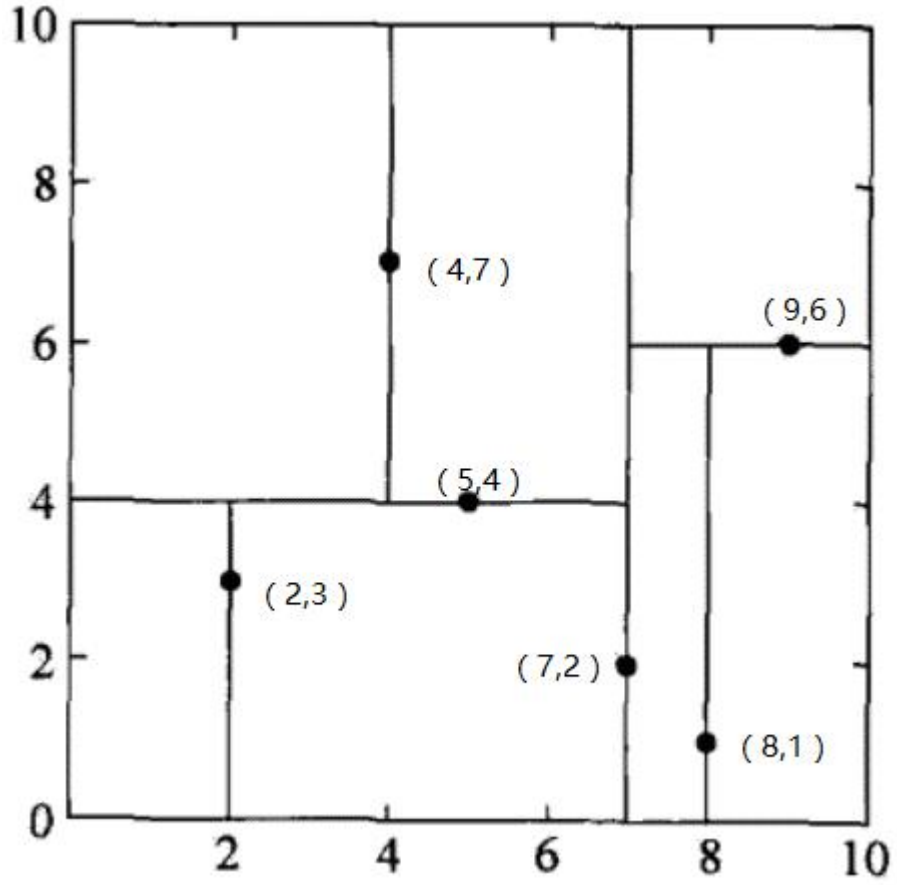
KD Tree构建方式

- KD树采用从m个样本的n维特征中，分别计算n个特征取值的方差，用方差最大的第k维特征n_k作为根节点。对于这个特征，选择取值的中位数n_{kv}作为样本的划分点，对于小于该值的样本划分到左子树，对于大于等于该值的样本划分到右子树，对左右子树采用同样的方式找方差最大的特征作为根节点，递归即可产生KD树。



KD tree

- 二维样本: {(2,3), (5,4), (9,6), (4,7), (8,1), (7,2)}

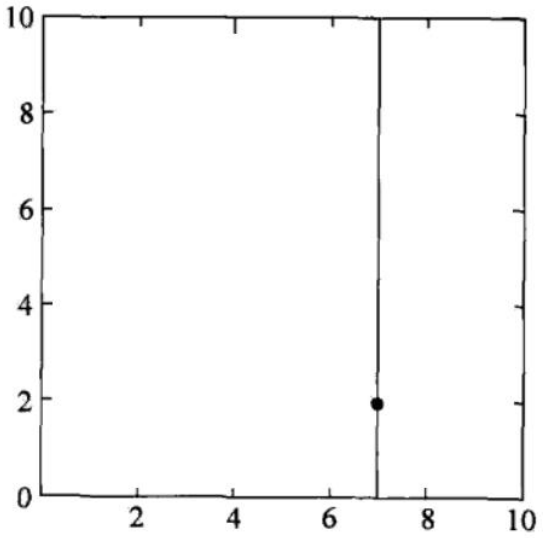


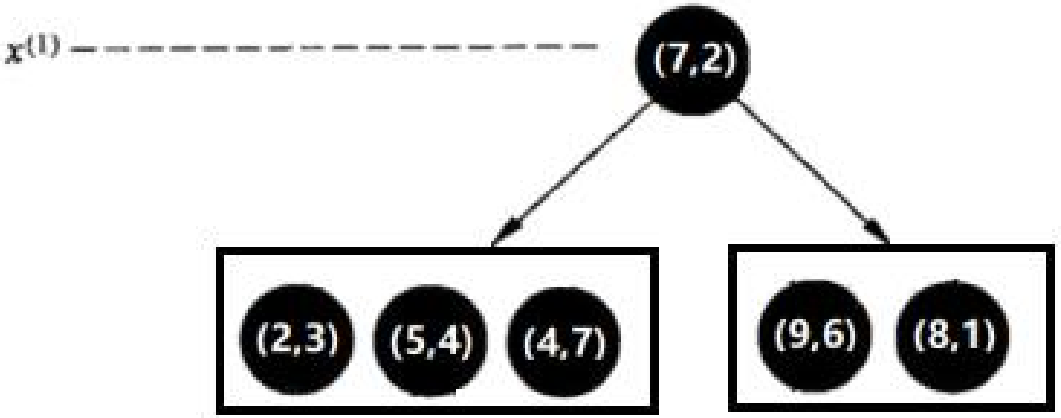
构造kd树

- 给定一个二维空间的数据集 $T=\{(2,3)^T,(5,4)^T,(9,6)^T,(4,7)^T,(8,1)^T,(7,2)^T\}$

请画出：特征空间的划分过程、kd树的构造过程。

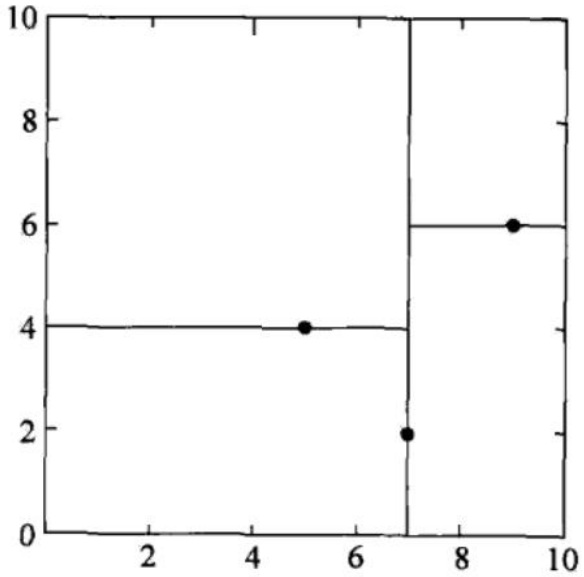
- 第一步：选择$x^{(1)}$轴，6个数据点的$x^{(1)}$坐标上的数字分别是2,5,9,4,8,7。取中位数7（不是严格意义的中位数，取较大的数），以$x^{(1)}=7$将特征空间分为两个矩形：

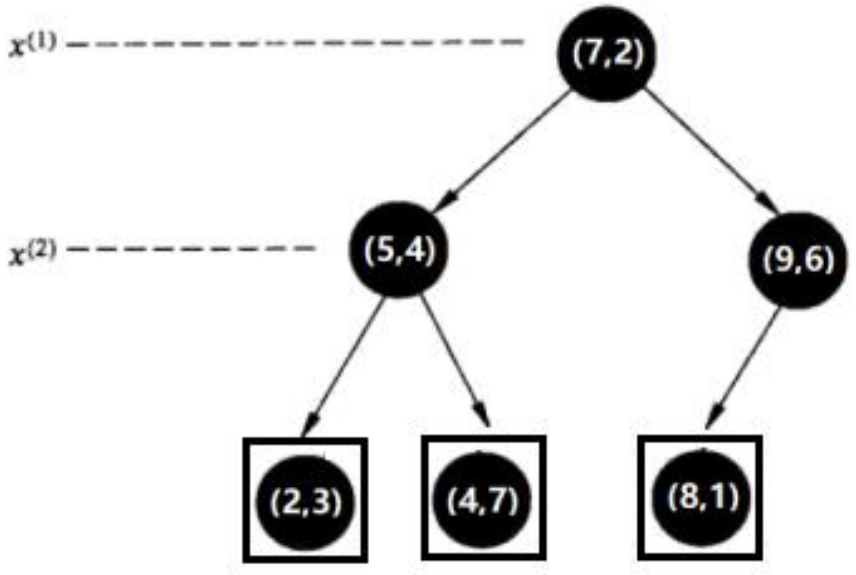




构造kd树

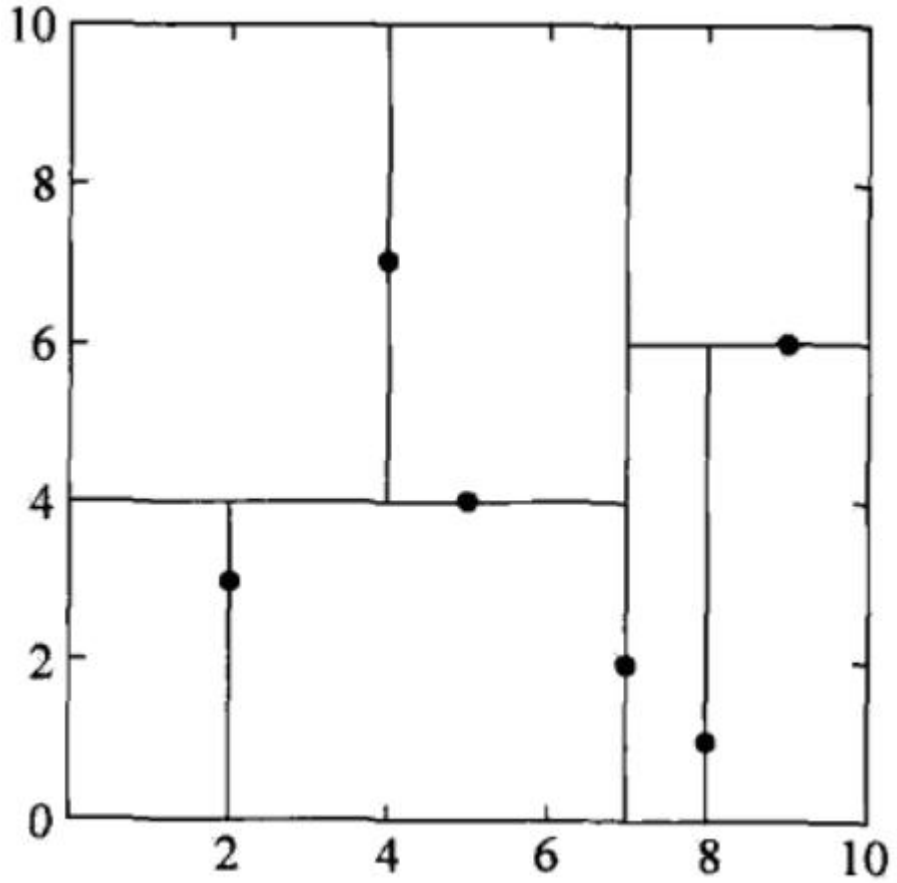
- 第二步：选择$x^{(2)}$轴，处理左子树，3个数据点的$x^{(2)}$坐标上的数字分别是3,4,7。取中位数4，以$x^{(2)}=4$将左子树对应的特征空间分为两个矩形；处理右子树，2个数据点的$x^{(2)}$坐标上的数字分别是6,1。取6，以$x^{(2)}=6$将右子树对应的特征空间分为两个矩形：

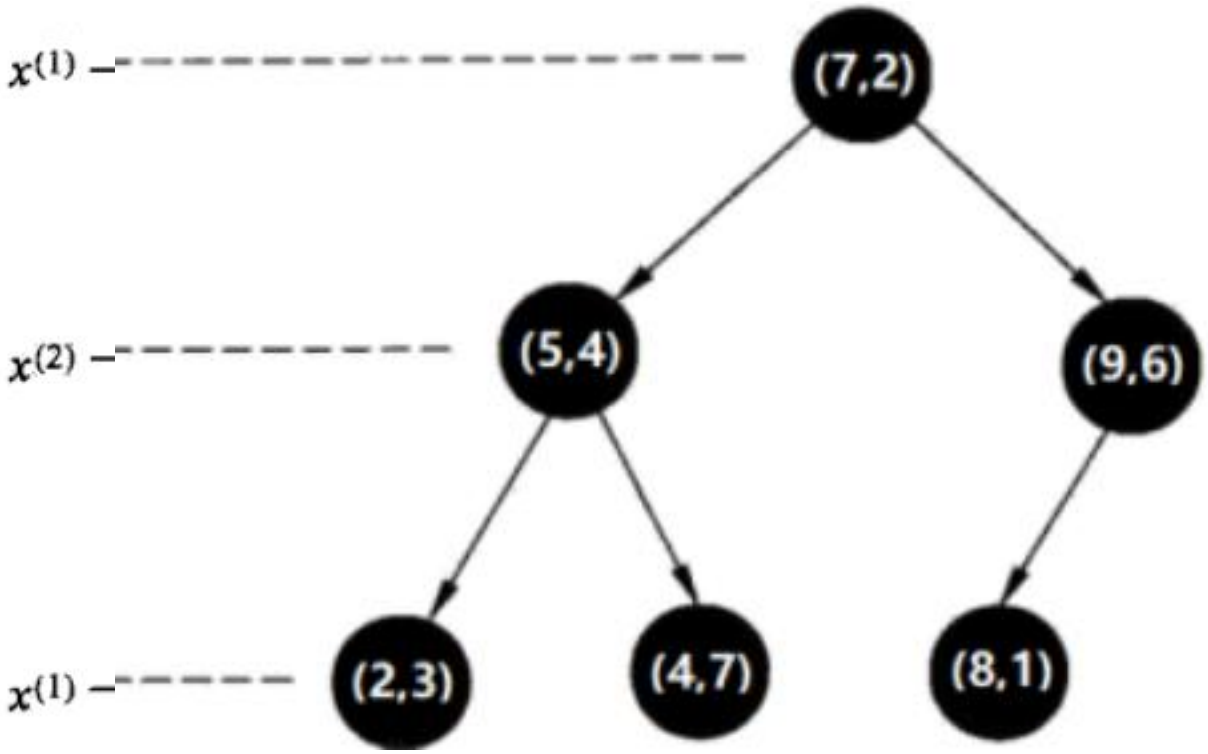




构造kd树

- 第三步：$x^{(1)}$轴，分别处理所有待处理的节点：





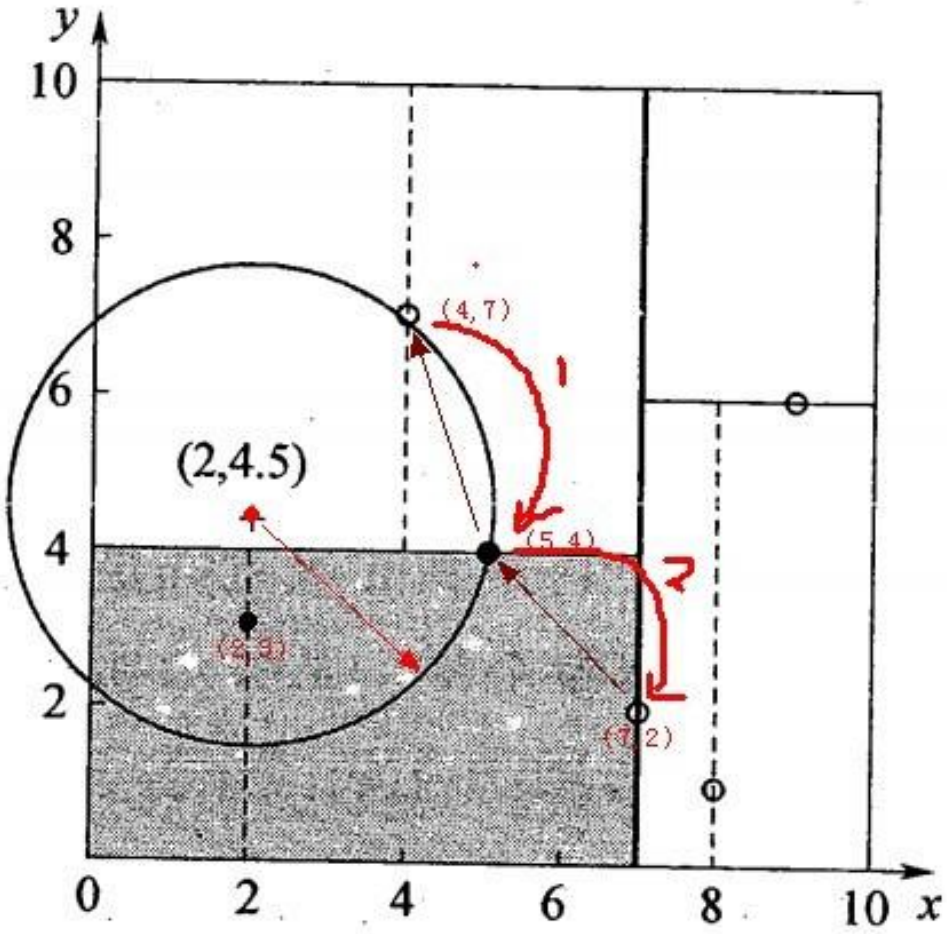
KD tree查找最近邻

- 当我们生成KD树以后，就可以去预测测试集里面的样本目标点了。对于一个目标点，我们首先在KD树里面找到包含目标点的叶子节点。以目标点为圆心，以目标点到叶子节点中样本实例的最短距离为半径，得到一个超球体，最近邻的点一定在这个超球体内部。然后返回叶子节点的父节点，检查另一个子节点包含的超矩形体是否和超球体相交，如果相交就到这个子节点寻找是否有更加近的近邻,有的话就更新最近邻。如果不相交那就简单了，我们直接返回父节点的父节点，在另一个子树继续搜索最近邻。当回溯到根节点时，算法结束，此时保存的最近邻节点就是最终的最近邻。

KD tree查找最近邻

- 找到所属的叶子节点后，以目标点为圆心，以目标点到最近样本点(一般为当前叶子节点中的其它训练数据或者刚刚经过的父节点)为半径画圆，从最近样本点往根节点进行遍历，如果这个圆和分割节点的分割线有交线，那么就考虑分割点的另外一个子树。如果在遍历过程中，找到距离比刚开始的样本距离近的样本，那就进行更新操作。
- 一直迭代遍历到根节点上，结束循环找到最终的最小距离的样本。

KD tree查找最近邻



KNN参数说明

参数	KNeighborsClassifier	KNeighborsRegressor
weights	样本权重，可选参数: uniform(等权重)、distance(权重和距离成反比，越近影响越强)；默认为uniform	
n_neighbors	邻近数目，默认为5	
algorithm	计算方式，默认为auto，可选参数: auto、ball_tree、kd_tree、brute；推荐选择kd_tree	
leaf_size	在使用KD_Tree、Ball_Tree的时候，允许存在最多的叶子数量，默认为30	
metric	样本之间距离度量公式，默认为minkowski（闵可夫斯基）；当参数p为2的时候，其实就是欧几里得距离	
p	给定minkowski距离中的p值，默认为2	