k近邻算法

李 波

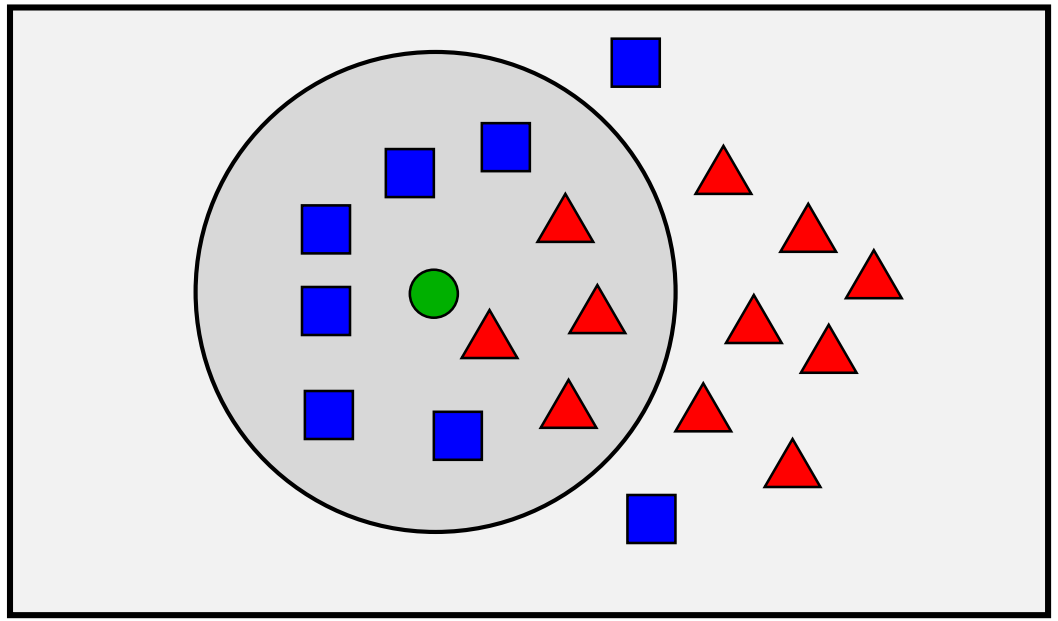
一个人的水平，就是他平时共处时间最长五个人的平均水平。

吉姆 约翰

k近邻算法 (k Nearest Neighbor algorithm, kNN)

- *K*近邻是一个有监督学习算法。
- *K*近邻算法没有训练过程，即无需训练。
- 给定一个测试样本，*k*近邻算法在训练数据里找到与测试样本距离最近的*k*个样本。
- 利用这*k*个训练样本标签，预测测试样本的标签。

k近邻算法 (k Nearest Neighbor algorithm, kNN)



k = 9

训练数据

测试数据输入

1. 计算测试样本输入与所有训练样本距离.
2. 找出与测试样本距离最近的k个训练样本.
   (6个蓝色，4个红色)
3. k个训练样本标签投票，投票结果作为测试样本的标签（6个蓝色，4个红色，投票结果为蓝色，预测标签为蓝色)

k近邻算法 (k Nearest Neighbor algorithm, kNN)

- K近邻算法伪代码

输入：$\boldsymbol{X}_{train}, \boldsymbol{y}_{train}, k, \boldsymbol{x}_{test}$.

输出：测试样本x_{test}的预测标签.

(1) 对于训练数据 $\boldsymbol{X}_{train}$中的每一个样本x:

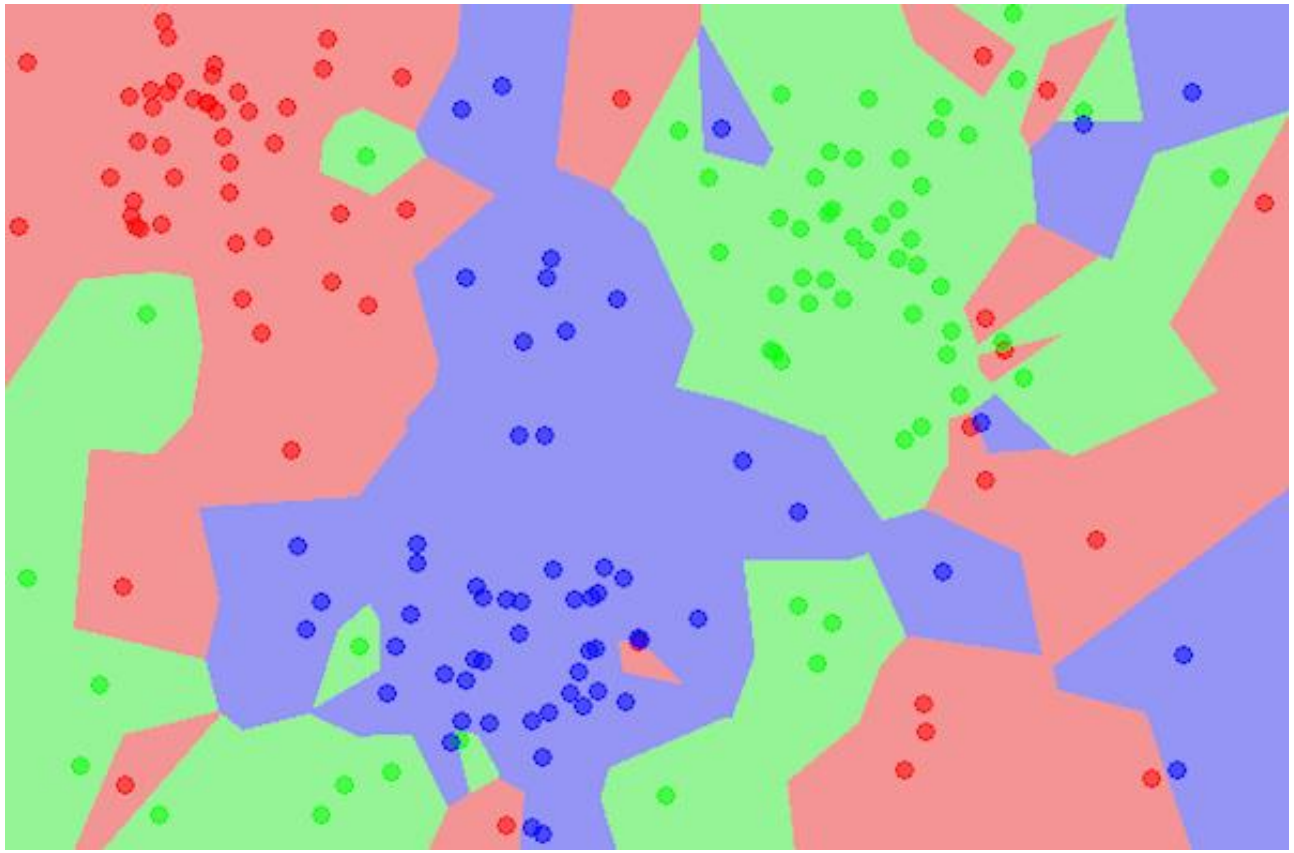
(2) 计算d_i为x与x_{test}的距离.

(3) 从距离集合$[d_1, d_2, \cdots d_n]$中选取k个最小距离，其索引组成集合Γ.

(4) $\boldsymbol{y}_{train}[i]$，$i \in \Gamma$中投票决定预测标签$\hat{y}$.

(5) 返回$\hat{y}$.

k近邻算法 (k Nearest Neighbor algorithm, kNN)

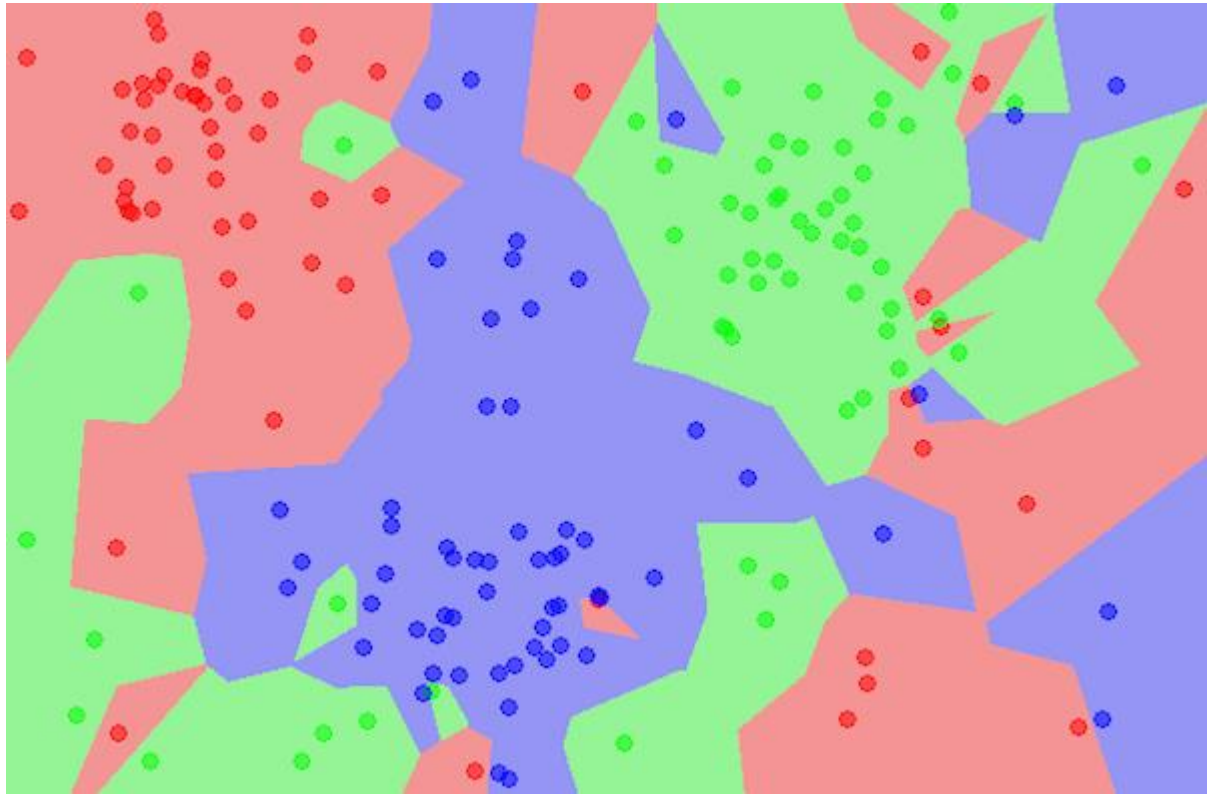


K近邻算法分类后的区域

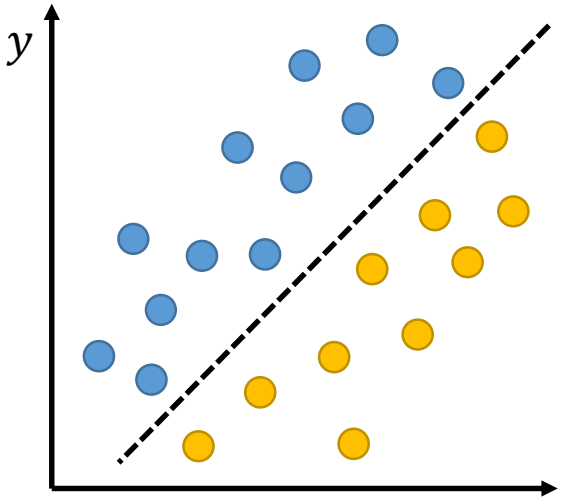
- 对二维平面网格化.
- 每个网格点作为测试样本输入.
- 利用k近邻算法预测每个网格颜色.

k近邻算法 (k Nearest Neighbor algorithm, kNN)

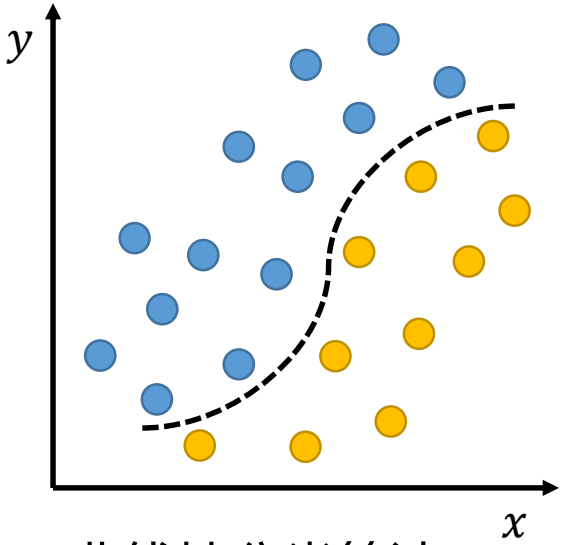
- K近邻算法是线性分类算法还是非线性分类算法?



K近邻算法分类后的区域



线性分类算法



非线性分类算法

k近邻算法 (k Nearest Neighbor algorithm, kNN)

K近邻算法需要关注的两个问题：

输入：$\boldsymbol{X}_{train}, \boldsymbol{y}_{train}, k, \boldsymbol{x}_{test}$.

输出：测试样本x_{test}的预测标签.

(1) 对于训练数据 $\boldsymbol{X}_{train}$中的每一个样本x：

(2) 计算d_i为x与x_{test}的距离.

如何定义距离?

(3) 从距离集合$[d_1, d_2, \cdots d_n]$中选取k个最小距离，其索引组成集合Γ.

(4) $\boldsymbol{y}_{train}[i]$，$i \in \Gamma$中投票决定预测标签$\hat{y}$.

如何选择k值?

(5) 返回$\hat{y}$.

k近邻算法 (k Nearest Neighbor algorithm, kNN)

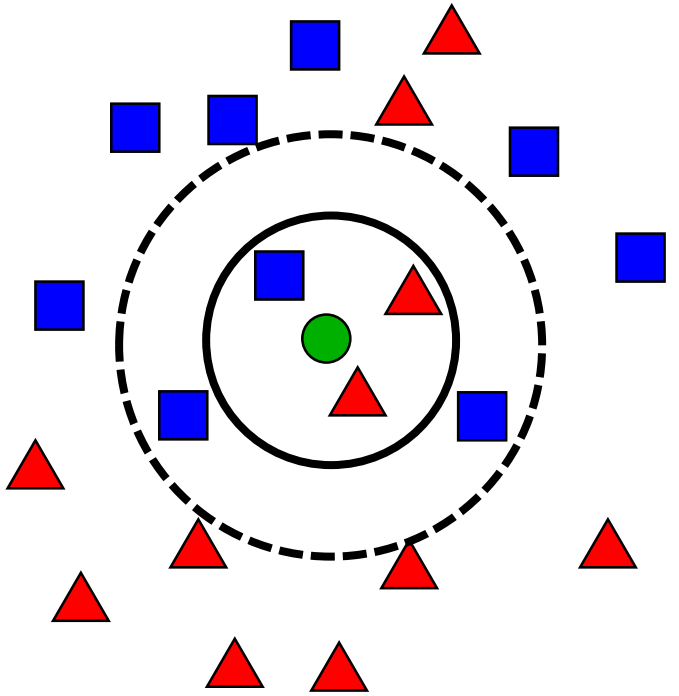
根据k近邻算法来给绿色圆点进行分类。如何选择*k*的值?

- *k*=3

  如果*k*=3，与绿色圆点的最邻近的3个点是2个红色三角形和1个蓝色方形，绿色点属于红色的三角形一类。

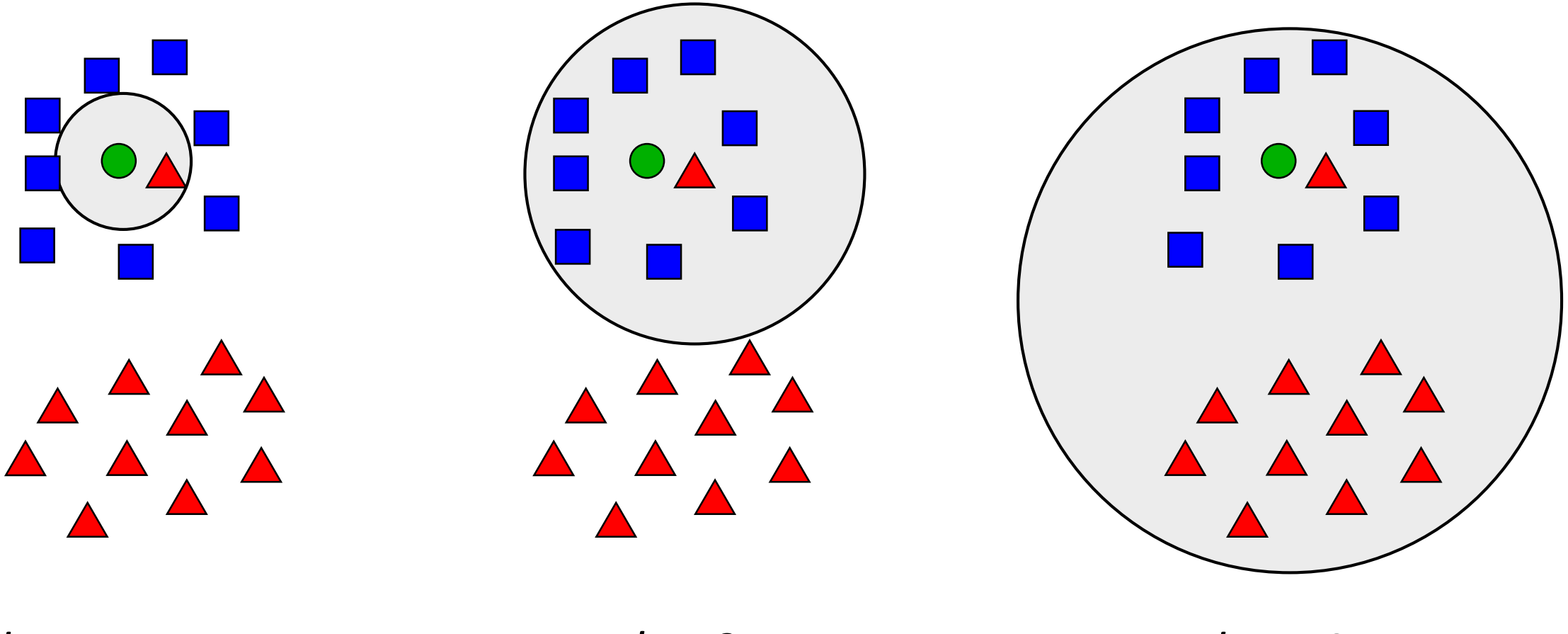
- *k*=5

  如果*k*=5，与绿色圆点的最邻近的5个点是2个红色三角形和3个蓝色方形，绿色点属于蓝色方形一类。

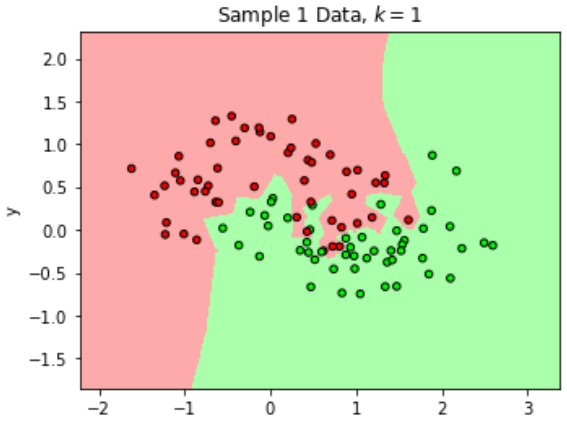
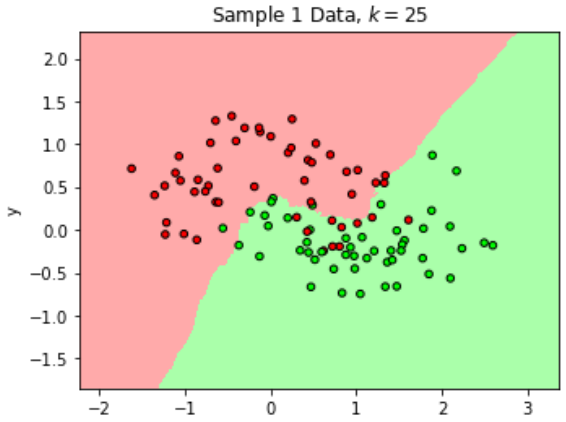
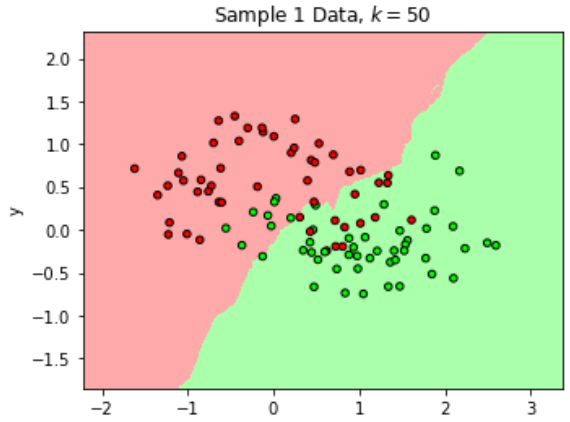
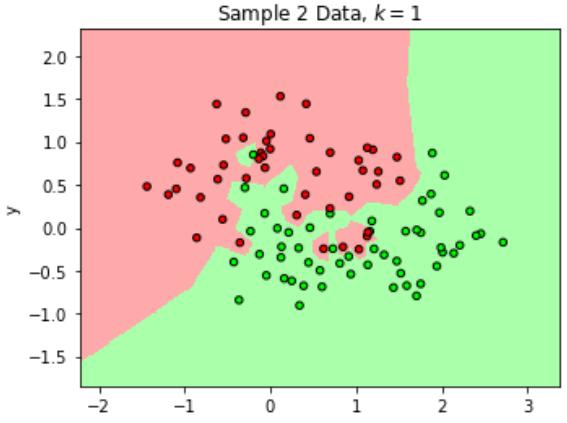
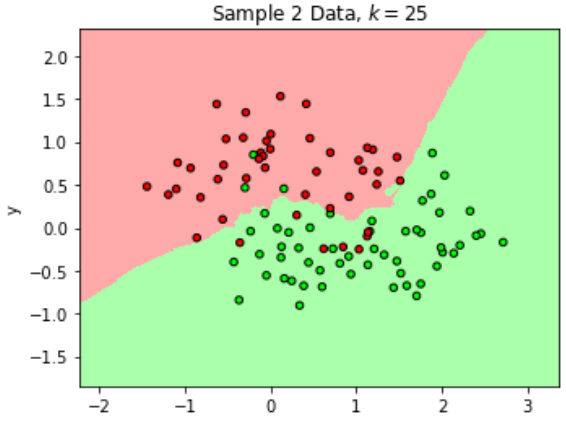
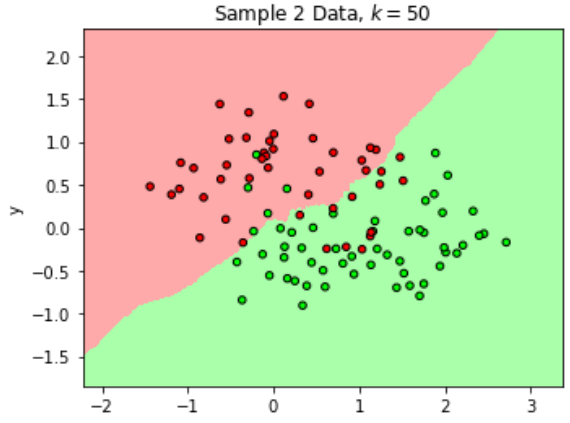
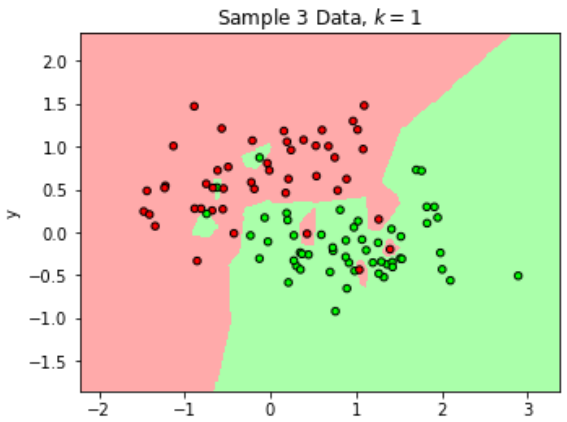
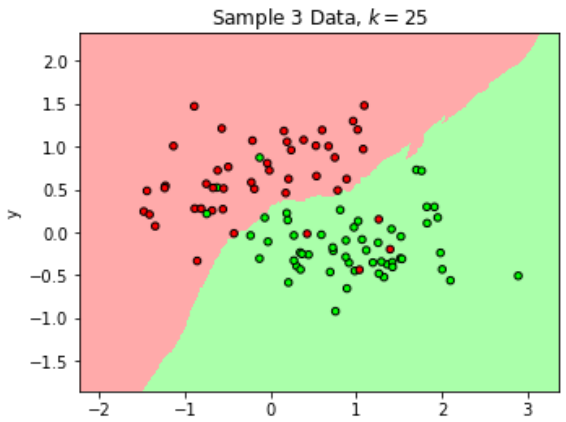
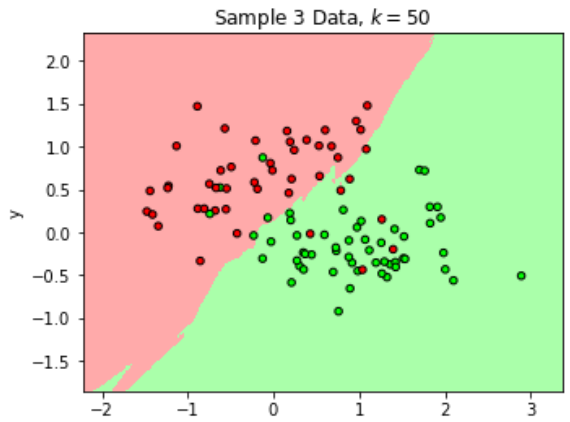


k近邻算法 (k Nearest Neighbor algorithm, kNN)

- 最优k值的确定



k = 1　　k = 9　　k = 19

	$k = 1$	$k = 25$	$k = 50$
Training data 1			
Training data 2			
Training data 3			

From https://github.com/inkawhich/machine-learning-course

k近邻算法 (k Nearest Neighbor algorithm, kNN)

- 如果k太小，比如$k=1$，预测的标签对训练数据敏感，这时候**过拟合**发生。

- 如果k很大，比如$k=n$，即所有训练数据都用来预测一个测试样本的标签，这时候预测的标签对训练数据变化不敏感，**欠拟合**发生。

k近邻算法 (k Nearest Neighbor algorithm, kNN)

距离的度量

两个样本输入 $\boldsymbol{x}_i$, $\boldsymbol{x}_j$的L_p距离定义为:

$$L_p(\boldsymbol{x}_i, \boldsymbol{x}_j) = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_p = \left(\sum_{t=1}^{n} |x_{it} - x_{jt}|^p\right)^{\frac{1}{p}}$$

当 $p = 2$ 时，距离为欧氏距离(Euclidean distance)，即

$$L_2(\boldsymbol{x}_i, \boldsymbol{x}_j) = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2 = \left(\sum_{t=1}^{n} |x_{it} - x_{jt}|^2\right)^{\frac{1}{2}}$$

当$p = 1$时，距离为曼哈顿距离(Manhattan distance)，即

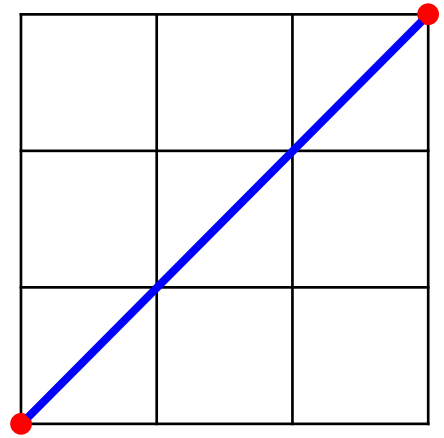
$$L_1(\mathrm{x}_i, \mathrm{x}_j) = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_1 = \sum_{l=1}^{n} |x_i^l - x_j^l|$$

k近邻算法 (k Nearest Neighbor algorithm, kNN)

距离的度量

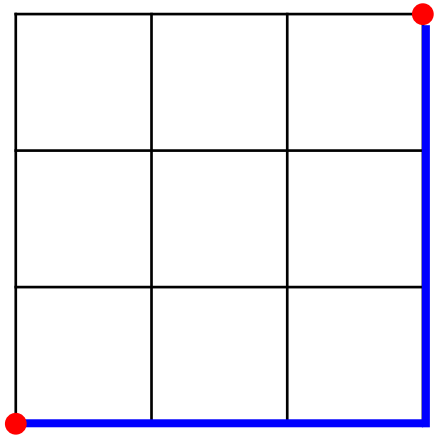
- 两个样本输入 $\boldsymbol{x}_i$, $\boldsymbol{x}_j$的L_p距离定义为:

$$L_p(\boldsymbol{x}_i, \boldsymbol{x}_j) = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_p = \left(\sum_{t=1}^{n} |x_{it} - x_{jt}|^p\right)^{\frac{1}{p}}$$



- 当 $p = 2$ 时，距离为欧氏距离(Euclidean distance)，即

$$L_2(\boldsymbol{x}_i, \boldsymbol{x}_j) = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_2 = \left(\sum_{t=1}^{n} |x_{it} - x_{jt}|^2\right)^{\frac{1}{2}}$$



- 当$p = 1$时，距离为曼哈顿距离(Manhattan distance)，即

$$L_1(\mathrm{x}_i, \mathrm{x}_j) = \|\boldsymbol{x}_i - \boldsymbol{x}_j\|_1 = \sum_{l=1}^{n} |x_i^l - x_j^l|$$

k近邻算法 (k Nearest Neighbor algorithm, kNN)

距离的度量

- 以人的身高(cm)与鞋码(欧制)作为特征值；类别为男性或者女性。有5个训练样本如下

x	(179,42)	(178,43)	(165,36)	(177,42)	(160,35)
y	男	男	女	男	女

- 测试样本输入为x(167,43)，kNN预测测试样本标签，取K=3.
- 计算测试输入与每个训练数据输入的距离

$$d_1 = \|\boldsymbol{x}_1 - \boldsymbol{x}_{test}\|_2 = \sqrt{(167-179)^2 + (43-42)^2} = \sqrt{145}$$
$$d_2 = \|\boldsymbol{x}_2 - \boldsymbol{x}_{test}\|_2 = \sqrt{(167-178)^2 + (43-43)^2} = \sqrt{121}$$
$$d_3 = \|\boldsymbol{x}_3 - \boldsymbol{x}_{test}\|_2 = \sqrt{(167-165)^2 + (43-36)^2} = \sqrt{53}$$
$$d_4 = \|\boldsymbol{x}_4 - \boldsymbol{x}_{test}\|_2 = \sqrt{(167-177)^2 + (43-42)^2} = \sqrt{101}$$
$$d_5 = \|\boldsymbol{x}_5 - \boldsymbol{x}_{test}\|_2 = \sqrt{(167-160)^2 + (43-45)^2} = \sqrt{103}$$

- 距离排序　$d_3 < d_4 < d_5 < d_2 < d_1$

k近邻算法 (k Nearest Neighbor algorithm, kNN)

距离的度量

- 训练样本

x	(179,42)	(178,43)	(165,36)	(177,42)	(160,35)
y	男	男	女	男	女

- 测试样本输入为x(167,43).
- 训练样本与测试输入距离排序为 $d_3 < d_4 < d_5 < d_2 < d_1$.
- 与测试样本输入距离最近的三个训练样本分别为**样本3**，**样本4**，**样本5**.
- 根据kNN算法，测试样本的预测标签是什么?
- 根据对数据的理解，预测结果有什么问题?
- 如何解决?

k近邻算法 (k Nearest Neighbor algorithm, kNN)

- **优点**：
  1. 无需训练。
  2. 是一种非参数分类器，简单直观，易于实现。
  3. 是一种在线分类器，可以直接在训练数据集加入新增数据，而不必重新训练模型。
  4. 具有可解释性，可以为预测标签提供预测证据（*k*个训练样本）。
- **缺点**：
  1. 计算量较大，预测过程需要计算测试样本输入与所有训练样本的距离。
  2. 难以选择好的距离度量。

k近邻算法 (k Nearest Neighbor algorithm, kNN)

距离的计算

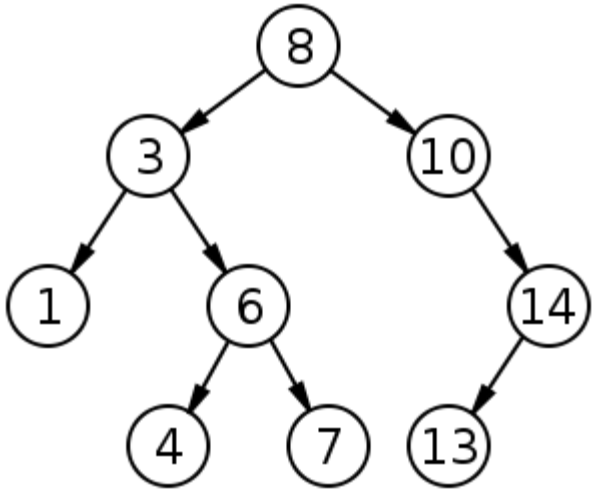
- *k*近邻算法需要计算测试样本输入与所有训练数据的距离。如果训练样本数量很大，距离的计算计算复杂度很高。
- kd树算法将训练数据结构化，从而降低距离计算的复杂度。

k近邻算法 (k Nearest Neighbor algorithm, kNN)

kd树(k-dimension tree)

- 二叉搜索树(kd树为二叉搜索树(Binary Search Tree, BST)

  1. 每个节点最多有两个子节点，即左子节点和右子节点.
  2. 一个节点的值大于等于左子树上所有节点的值.
  3. 一个节点的值小于右子树上所有节点的值.



k近邻算法 (k Nearest Neighbor algorithm, kNN)

kd树 (k-dimension tree):

构造步骤

Kd树的构造循环进行以下流程，直至划分的数据子集为一个样本:

1. 特征索引集合为$\Lambda = \{1,2,\cdots,k\}$.
2. 从Λ中选择方差最大的特征，索引为i，在该特征或维度选择中位数对训练数据进行切分，得到两个子数据集. $\Lambda = \Lambda - i$.
3. 选择一个点为节点。两个子数据集将分别用于构建左子节点和右子节点.
4. 对切分后的两个子集合重复（1）步骤的过程，直至$\Lambda = \emptyset$.

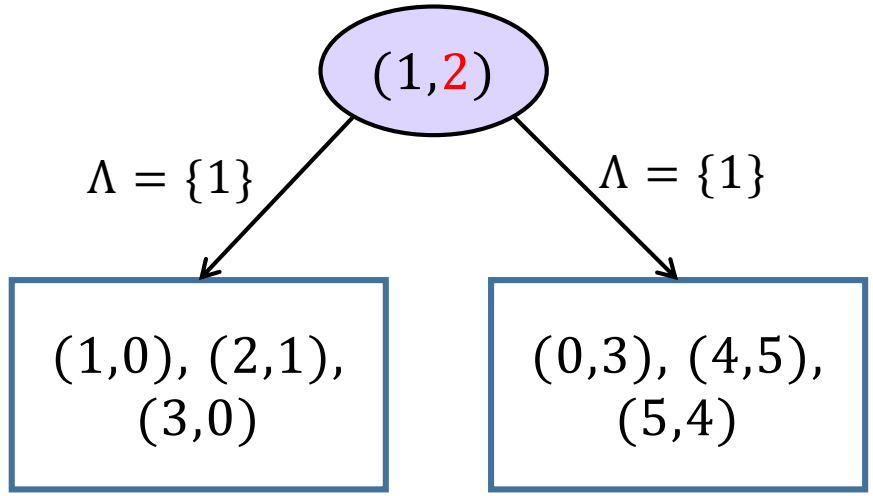
k近邻算法 (k Nearest Neighbor algorithm, kNN)

kd树 (k-dimension tree): 构造步骤

Kd树的构造循环进行以下流程，直至划分的数据子集为一个样本

1. 特征索引集合为$\Lambda = \{1,2,\cdots,k\}$.
2. 从Λ中选择方差最大的特征，索引为i，在该特征或维度选择中位数对训练数据进行切分，得到两个子数据集. $\Lambda = \Lambda - i$.
3. 选择一个点为节点。两个子数据集将分别用于构建左子节点和右子节点.
4. 对切分后的两个子集合重复（1）步骤的过程，直至$\Lambda = \emptyset$.

- (0,3) ,(1,0) ,(1,2), (2,1), (3,0), (4,5), (5,4)
- $\Lambda = \{1,2\}$
- 第一个维度方差约等于2.78
- 第二个维度方差约等于3.27
- 因为3.27>2.78，选择第二个维度切分
- $\Lambda = \{1\}$
- 第二个维度中位数是2



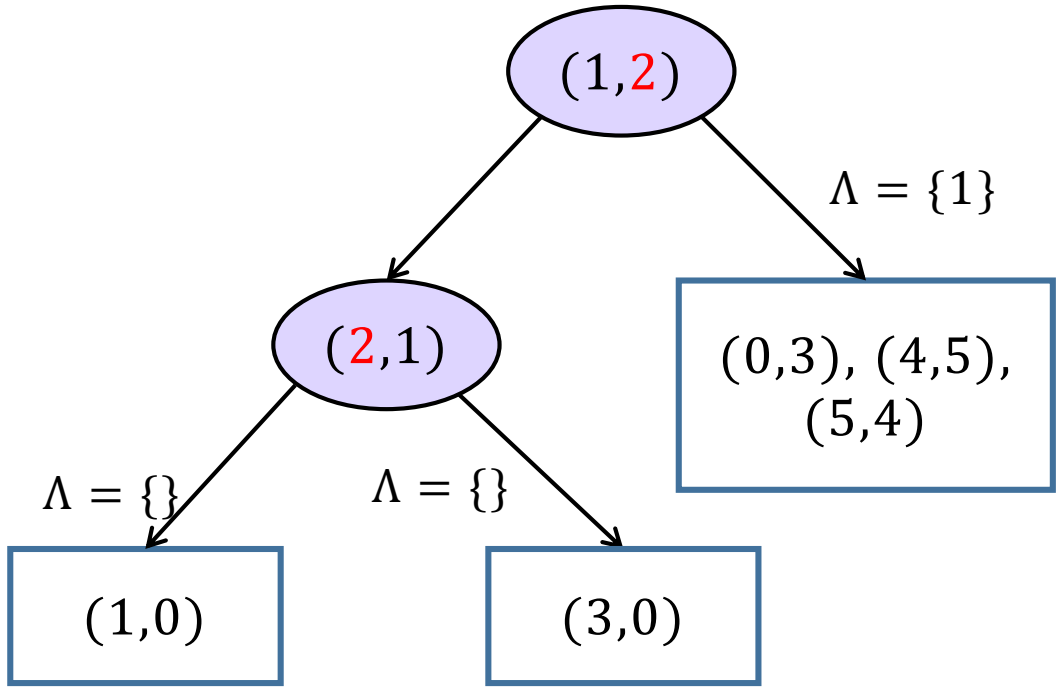
k近邻算法 (k Nearest Neighbor algorithm, kNN)

kd树 (k-dimension tree): 构造步骤

Kd树的构造循环进行以下流程，直至划分的数据子集为一个样本

1. 特征索引集合为$\Lambda = \{1,2,\cdots,k\}$.
2. 从Λ中选择方差最大的特征，索引为i，在该特征或维度选择中位数对训练数据进行切分，得到两个子数据集. $\Lambda = \Lambda - i$.
3. 选择一个点为节点。两个子数据集将分别用于构建左子节点和右子节点.
4. 对切分后的两个子集合重复（1）步骤的过程，直至$\Lambda = \emptyset$.

- (1,0) , (2,1), (3,0)
- $\Lambda = \{1\}$
- 第一个维度中位数是2
- $\Lambda = \{\}$



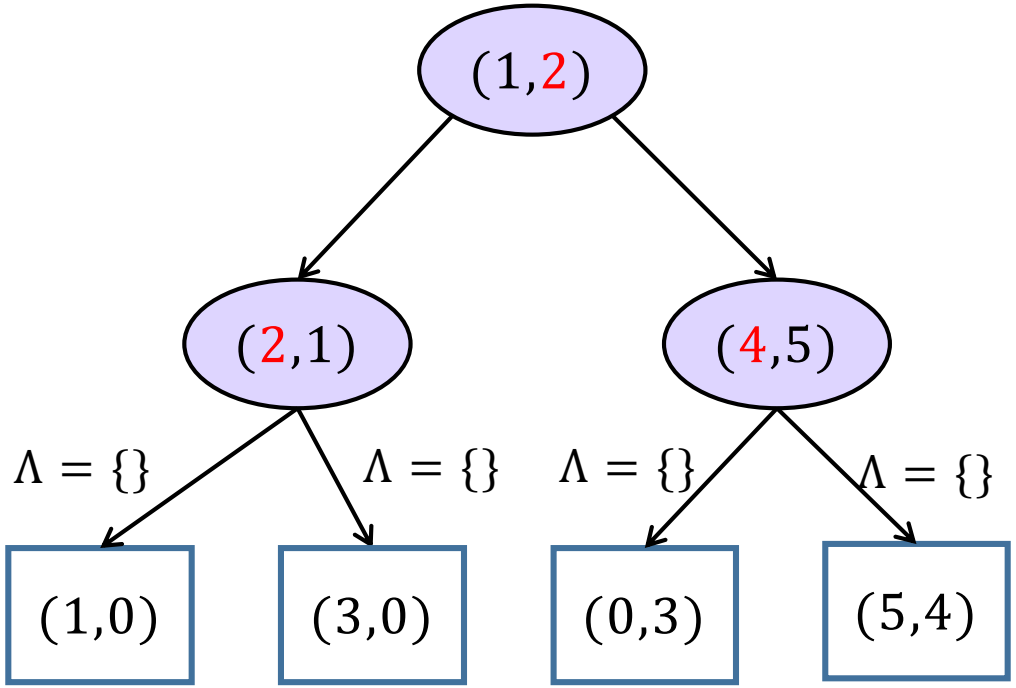
k近邻算法 (k Nearest Neighbor algorithm, kNN)

kd树 (k-dimension tree): 构造步骤

Kd树的构造循环进行以下流程，直至划分的数据子集为一个样本

1. 特征索引集合为$\Lambda = \{1,2,\cdots,k\}$.
2. 从Λ中选择方差最大的特征，索引为i，在该特征或维度选择中位数对训练数据进行切分，得到两个子数据集. $\Lambda = \Lambda - i$.
3. 选择一个点为节点。两个子数据集将分别用于构建左子节点和右子节点.
4. 对切分后的两个子集合重复（1）步骤的过程，直至$\Lambda = \emptyset$.

- (0,3) , (4,5), (5,4)
- $\Lambda = \{1\}$
- 第一个维度中位数是4
- $\Lambda = \{\}$



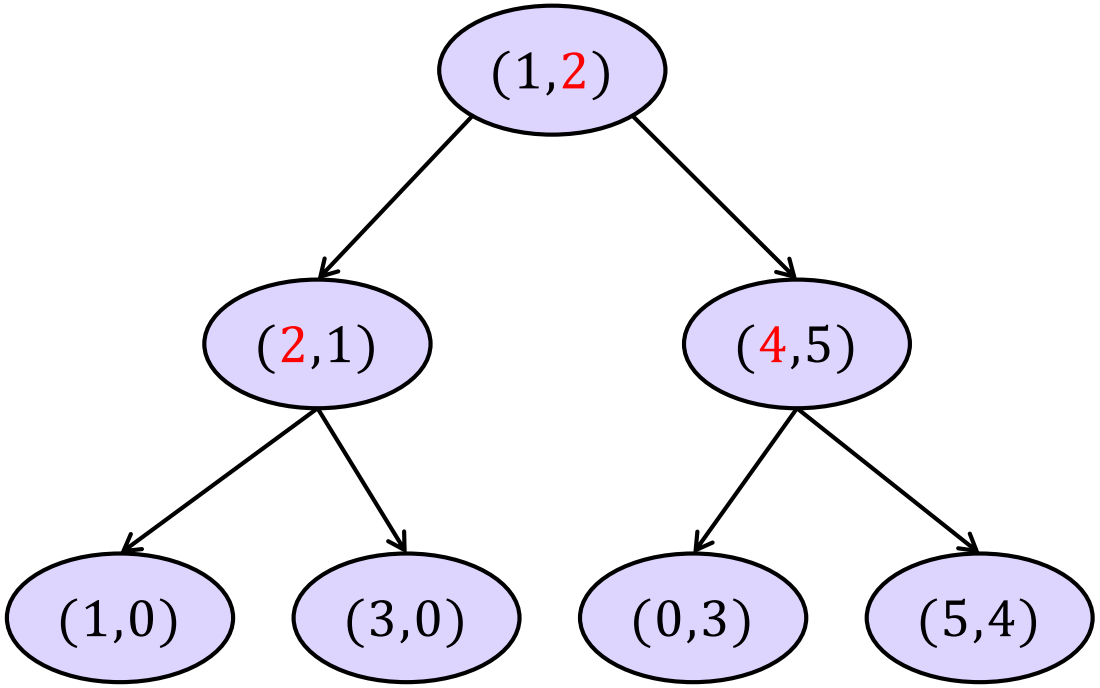
k近邻算法 (k Nearest Neighbor algorithm, kNN)

kd树 (k-dimension tree): 构造步骤

Kd树的构造循环进行以下流程，直至划分的数据子集为一个样本

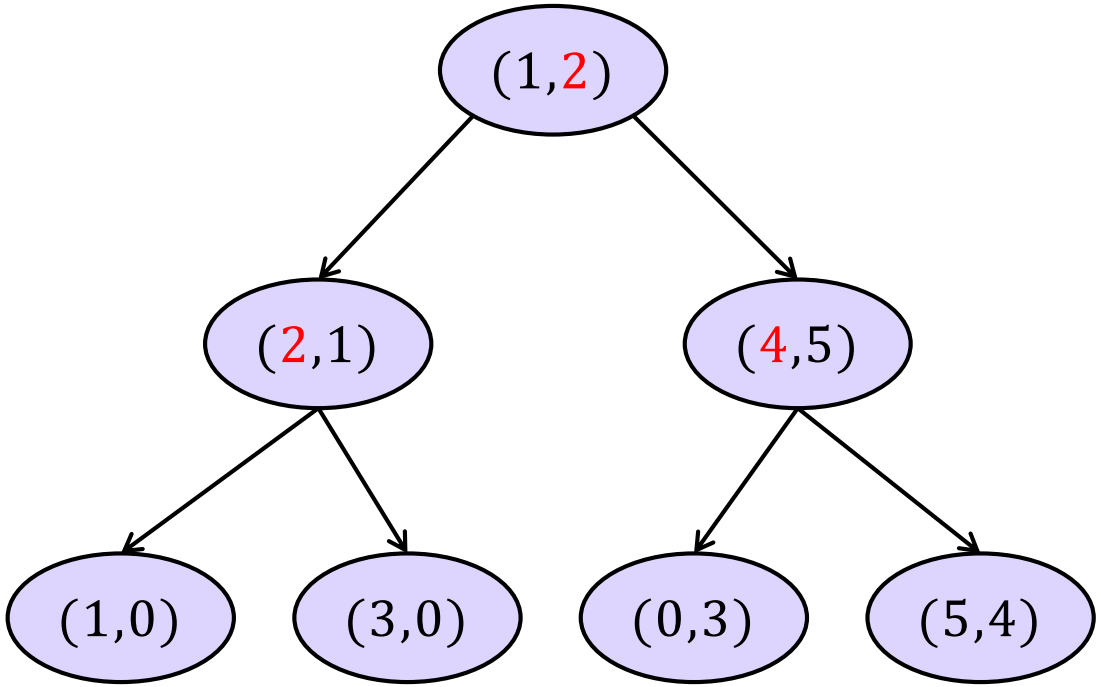
1. 特征索引集合为$\Lambda = \{1,2,\cdots,k\}$.
2. 从Λ中选择方差最大的特征，索引为i，在该特征或维度选择中位数对训练数据进行切分，得到两个子数据集. $\Lambda = \Lambda - i$.
3. 选择一个点为节点。两个子数据集将分别用于构建左子节点和右子节点.
4. 对切分后的两个子集合重复（1）步骤的过程，直至$\Lambda = \emptyset$.

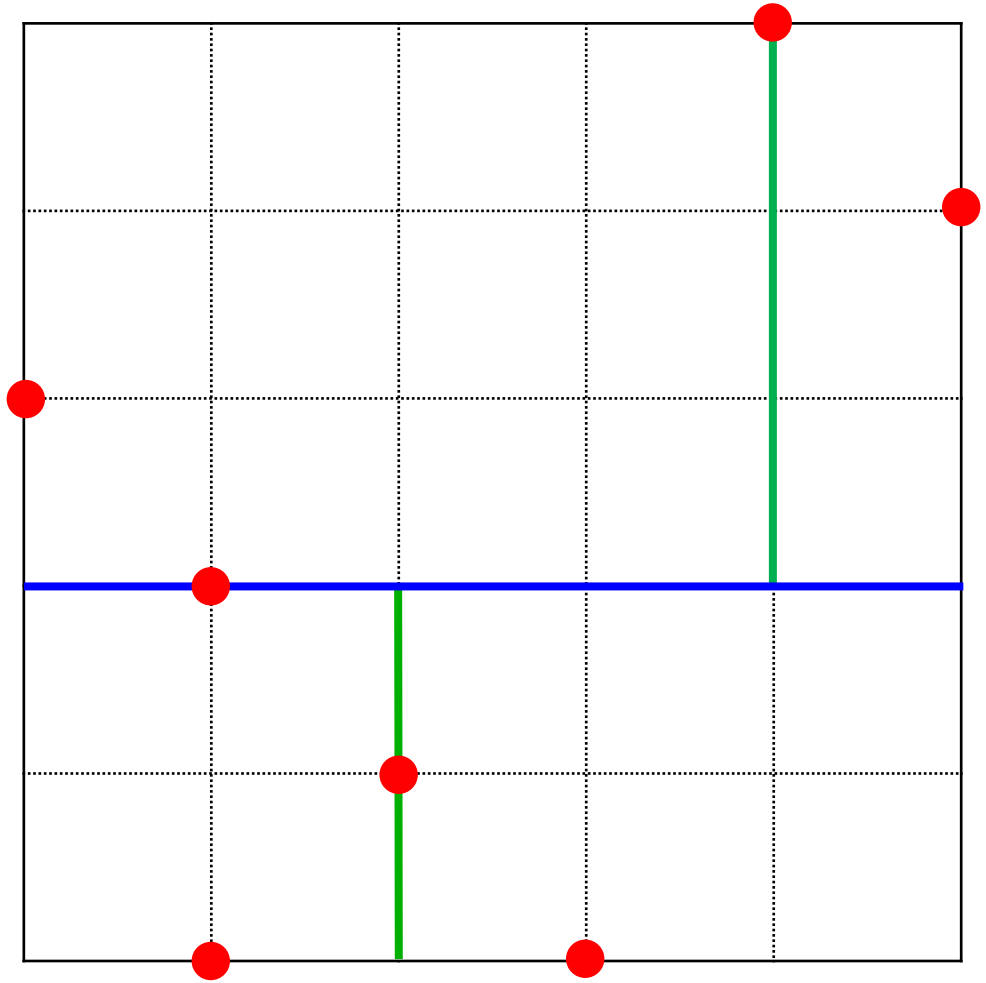
- (0,3) , (4,5), (5,4)
- $\Lambda = \{1\}$
- 第一个维度中位数是4
- $\Lambda = \{\}$



k近邻算法 (k Nearest Neighbor algorithm, kNN)

(0,3) ,(1,0) ,(1,2), (2,1), (3,0), (4,5), (5,4)





k近邻算法 (k Nearest Neighbor algorithm, kNN)

kd树 (k-dimension tree):

搜索最近点步骤

给定一个kd树和一个样本输入$\boldsymbol{x}$:

1. 从kd树根节点出发，在指定维上，比较$\boldsymbol{x}$和节点，直至到达叶子节点.
2. 从叶子节点，反向返回根节点，将遇到的节点存储在集合S中.
3. 计算$\boldsymbol{x}$与集合S中样本的距离，并选择距离最近的训练样本.

k近邻算法 (k Nearest Neighbor algorithm, kNN)

训练样本(0,3) ,(1,0) ,(1,2), (2,1), (3,0), (4,5), (5,4)

测试样本（1,1.8）