第4.1节基于样本的方法

提纲

- 概述
- k-近邻分类方法：K-Nearest-Neighbors
- 支撑向量机：Support Vector Machine (SVM)

概 述

- 已知一系列的训练样本，许多学习方法为样本数据建立起明确的一般化的模型与描述，
- 基于样本的学习方法只是**把训练样本存储起来，从这些样本中泛化的工作被推迟到分类新样本**
- 给定一个新的测试样本，它分析这个新样本与以前存储的样本的关系，并据此把一个目标函数值赋给新样本

概 述

- 基于样本的学习方法有时被称为消极学习法，它把处理工作延迟到分类新样本时
- 与其他方法相比，基于样本的方法的特点是：
  - 可以为不同的待分类查询样本建立不同的目标函数逼近，无须假想合理的假设
  - 许多技术不建立目标函数在整个样本空间上的逼近，只建立局部逼近，并将其用于与新样本邻近的样本
  - 解决目标函数很复杂，但期望具有不太复杂的局部逼近描述

提纲

- 概述
- k-近邻分类方法：K-Nearest-Neighbors
- 支撑向量机：Support Vector Machine (SVM)

k-近邻算法

- k-近邻算法是最基本的基于实例的学习方法
  - k-近邻算法假定所有的实例对应于*d*维空间R^d中的点，任意的实例表示为一个特征向量 $X=(X_1,X_2,...,X_d)$
  - 根据欧氏距离定义实例的距离。两个实例 X_i 和X_j的距离)定义为

$$D(X^{(i)},X^{(j)})=\sqrt{\sum_{k=1}^{d}\left(X_k^{(i)}-X_k^{(j)}\right)^2}$$

  - 在最近邻学习中，目标函数值可以是离散的也可以是连续的，以下先考虑离散的情况。

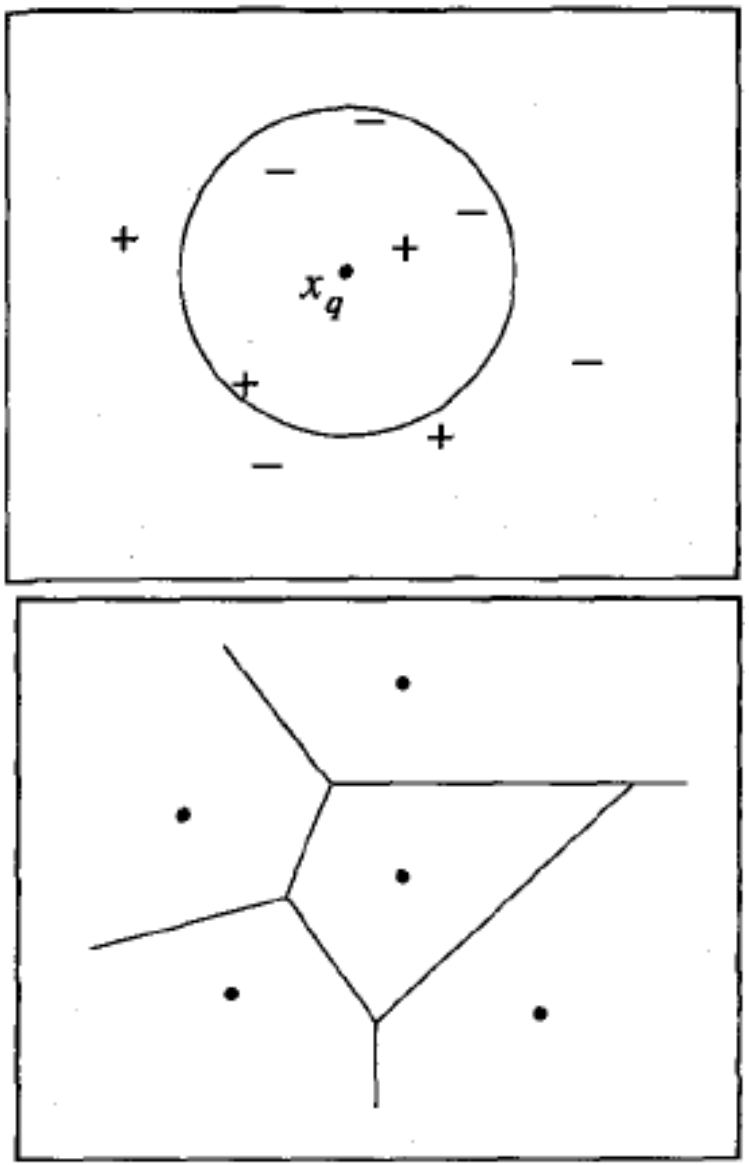
k-近邻算法

- 考虑离散目标函数 $f: R^d \to Y,\ Y = \{y^{(1)}, ..., y^{(N)}\}$
- 逼近离散值函数的k-近邻算法
  - 训练算法
    - 将每个训练样本<x,f(x)>加入到列表training_examples
  - 分类算法
    - 给定一个要分类的查询样本$x^{(q)}$
      - 在training_examples中选出最靠近$x^{(q)}$的k个样本，用这些样本表示
      - 最后的分类结果如下：

$$\hat{f}(x^{(q)}) \leftarrow \arg\max_{y} \sum_{i=1}^{k} \delta(y, f(x^{(i)}))$$ 其中，$$\delta(a,b) = \begin{cases} 1 & a = b \\ 0 & a \neq b \end{cases}$$

k-近邻算法

- 算法返回值是对 $f(x^{(q)})$ 的估计，它是距离 $x^{(q)}$ 最近的k个训练样本中最普遍的f值，结果与k的取值相关。
- k-近邻算法不形成关于目标函数f的明确的一般假设，仅在需要时计算每个新查询样本的分类
- 右图画出了1-近邻算法在整个样本空间上导致的决策面形状。这种图称为训练样本集合的Voronoi图
- **思考问题：**k-近邻算法隐含的一般函数是什么？



k-近邻算法

- 离散k-近邻算法简单修改后可逼近连续目标函数。
  - 即计算k个最接近样本的平均值，而不是计算其中的最普遍的值，为逼近f: $R^n \rightarrow R$，计算式如下：

$$\hat{f}(x^{(q)}) \leftarrow \frac{\sum_{i=1}^{k} f(x^{(q)})}{k}$$

  - 有情趣的同学可参考MeanShift算法

k-近邻算法

- 对k-近邻算法的一个改进是对k个近邻的贡献加权，越近的距离赋予越大的权值，如：

$$\hat{f}(x^{(q)}) \leftarrow \arg\max_{v \in V} \sum_{i=1}^{k} w^{(i)} \delta(v, f(x^{(i)})) \quad w^{(i)} = \frac{1}{d(x^{(q)}, x^{(i)})^2}$$

- 为了处理查询点$x^{(q)}$恰好匹配某个训练样本$x^{(i)}$，从而导致$d(x^{(q)}, x^{(i)})^2$为0的情况，令这种情况下的$\hat{f}(x^{(q)})$等于$f(x^{(i)})$ 如果有多个这样的训练样本，使用占多数的那个类作结果
- 也可以用类似的方式对实值目标函数进行距离加权，用下式替代表上面的计算式，w^i的定义与前相同

$$\hat{f}(x^{(q)}) \leftarrow \frac{\sum_{i=1}^{k} w^{(q)} f(x^{(q)})}{\sum_{i=1}^{k} w^{(q)}}$$

k-近邻算法

- k-近邻算法只考虑k个近邻用以分类查询点
  - 如果使用按距离加权，那么可以允许所有的训练样本影响$x^{(q)}$的分类，因为非常远的样本的影响很小
  - 考虑所有样本的不足是会使分类运行得更慢
  - 如果分类一个新样本时，考虑所有的训练样本，称为全局法；如果仅考虑靠近的训练样本，称为局部法

k-近邻算法

- 加权的k-近邻算法对训练数据中的噪声有很好的鲁棒性
  - 通过取k个近邻的加权平均，可以消除孤立的噪声样本的影响
- k-近邻方法的另外一个实践问题：**维数灾害**
  - 许多学习方法，比如决策树方法，选择部分属性作出判断，而k-近邻方法中样本间的距离是根据样本的所有属性计算的
  - 样本间距离会被大量的不相关属性所支配，可能导致相关属性的值很接近的样本相距很远

k-近邻算法

- 维数灾害的解决方法：
  - 对属性加权，相当于按比例缩放欧氏空间中的坐标轴，缩短对应不太相关的属性的坐标轴，拉长对应更相关属性的坐标轴
  - 线性数据降维
  - 流形数据降维

概 述

- 基于样本的方法的不足：
  - 分类新样本的开销可能很大。
    - 几乎所有的计算都发生在分类时，而不是在第一次遇到训练样本时。
    - 如何有效地索引训练样本是一个重要的问题
  - 当从存储器中检索相似的训练样本时，一般考虑样本的所有属性，如果目标概念仅依赖于很多属性中的几个，那么真正最“相似”的样本之间可能相距甚远

k-近邻算法

- k-近邻算法的另外一个实践问题：
  - 如何建立高效的索引
  - k-近邻算法推迟所有的处理，直到接收到一个新的查询，所以处理每个新查询可能需要大量的计算
  - kd-tree把样本存储在树的叶结点内，邻近的样本存储在同一个或附近的节点内，通过测试新查询$x^{(q)}$的选定属性，树的内部节点把查询$x^{(q)}$排列到相关的叶结点
  - Kd-tree例子