Decision Tree

决策树

OUTLINE

- 分类树
- 回归树
- 树剪枝
- R实现

决策树

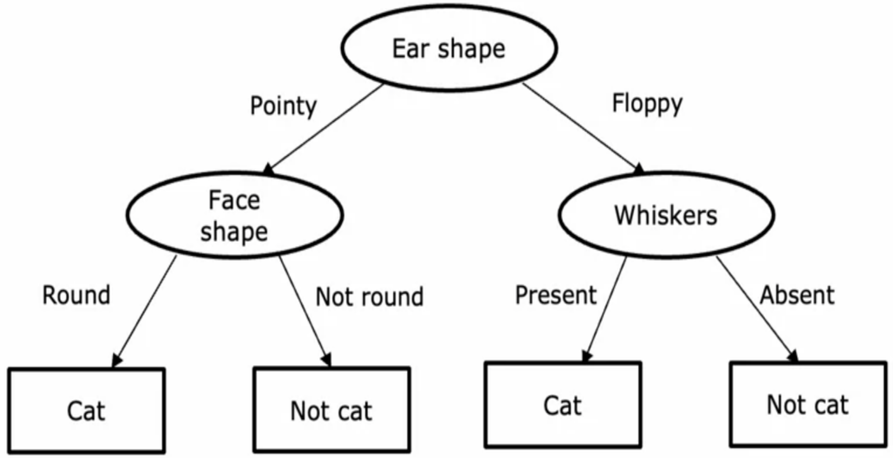
- 以两分类任务为例，我们希望从训练数据集中学得一个模型，用以对新数据进行分类
- 这样把样本分类的任务，可看作对 “当前样本属于A类吗？” 这一问题的 “决策” 过程
- 顾名思义，决策树基于**树状结构**进行决策，而这恰恰是人类在面临决策问题时一种很自然的处理机制
- 例如，我们要对 “这是好瓜吗？” 这个问题进行决策时，通常会进行一系列的判断：
  - 首先， 它是什么颜色？
  - 如果颜色是 “青绿色”，那么它的根蒂是什么形态？
  - 如果根蒂是 “蜷缩”，那么它敲起来是什么声音？
  - 最后我们得出最终决策：这是个好瓜！

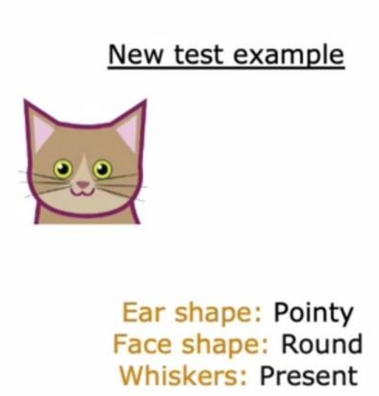
例：小猫分类

	Ear shape	Face shape	Whiskers	Cat
	Pointy	Round	Present	1
	Floppy	Not round	Present	1
	Floppy	Round	Absent	0
	Pointy	Not round	Present	0
	Pointy	Round	Present	1
	Pointy	Round	Absent	1
	Floppy	Not round	Absent	0
	Pointy	Round	Absent	1
	Floppy	Round	Absent	0
	Floppy	Round	Absent	0

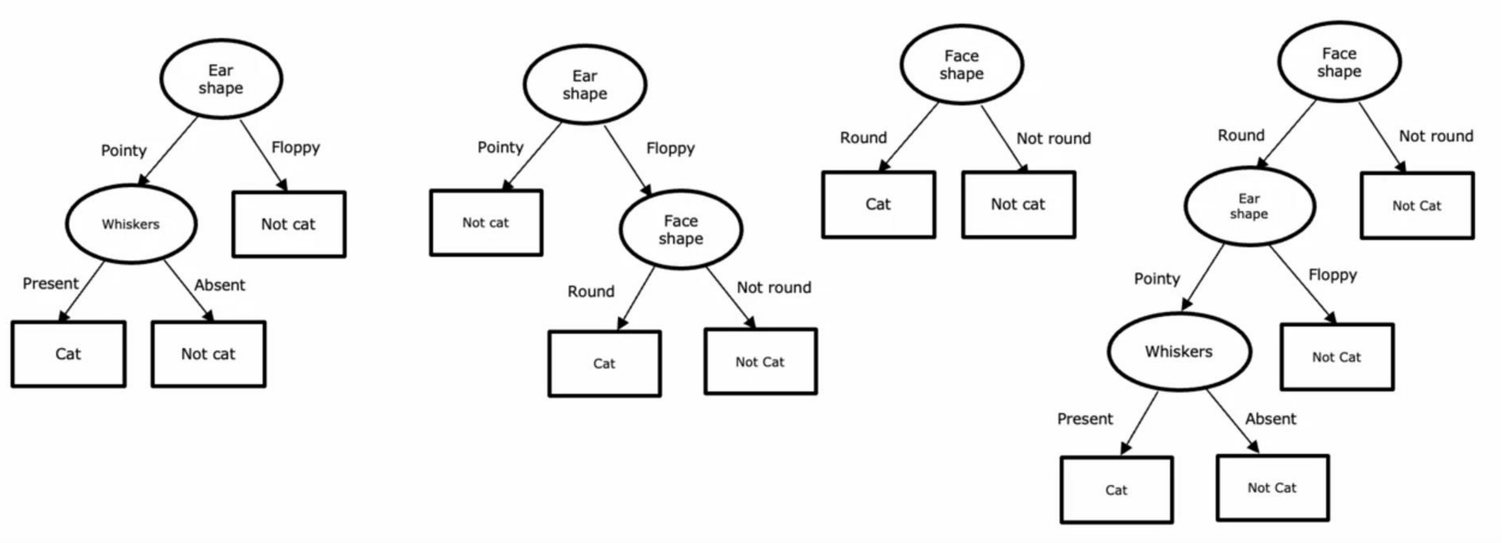
一个可行的小猫分类决策树

- 一个决策树包含一个根结点、若干个内部节点和若干个叶节点
- 叶节点对应于决策结果，其他每个节点则对应于一个属性测试
- 每个节点包含的样本集合根据属性测试的结果被划分到子节点中
- 根结点包含样本全集，从根节点到每个叶节点的路径对应了一个判定测试序列





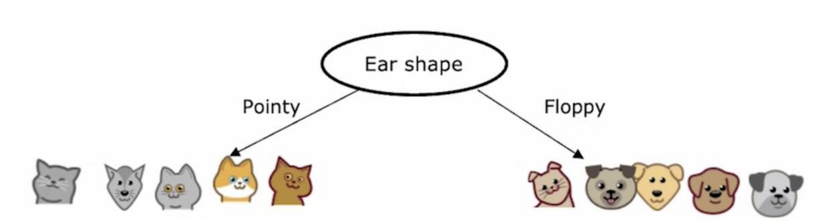
可行的决策树有很多

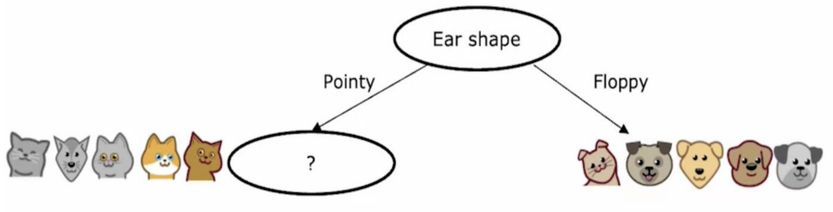


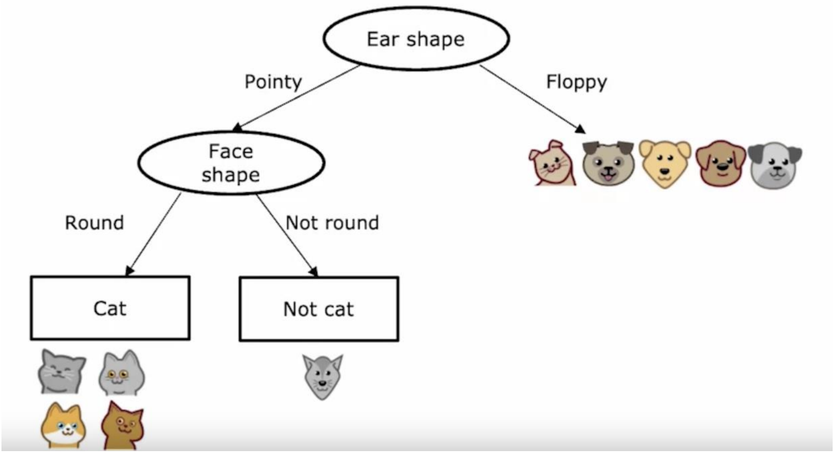
- 显然，有的决策树表现好，有的决策树表现不好
- 决策树学习的目的：在所有可能的决策树中，找到一颗泛化能力强，即处理从未见过的观测能力强的决策树

学习过程

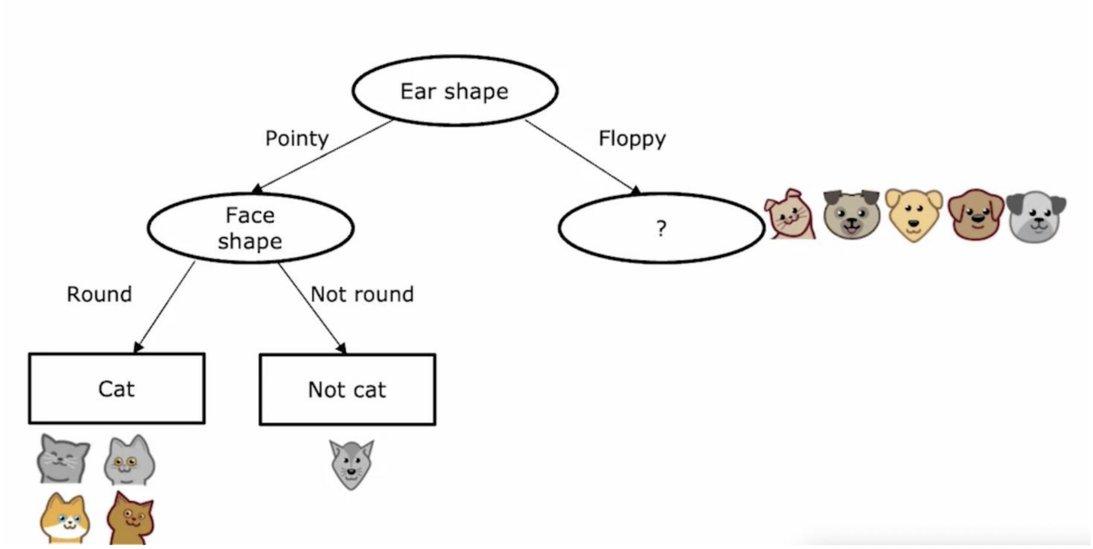
- 第一步：我们需要决定在根结点用哪个特征
- 假设我们决定用 ear shape 作为根节点，根据是 pointy 还是 floppy 将数据分成两类
- 第二步：我们需要分别决定在两个子节点上用哪个特征
- 假设我们决定在 pointy 指向的子节点用 face shape，根据是 round 还是 not round 继续将数据分成两类
- 最终得到的一类中只有猫，一类中只有狗，分类完成，形成两个叶节点

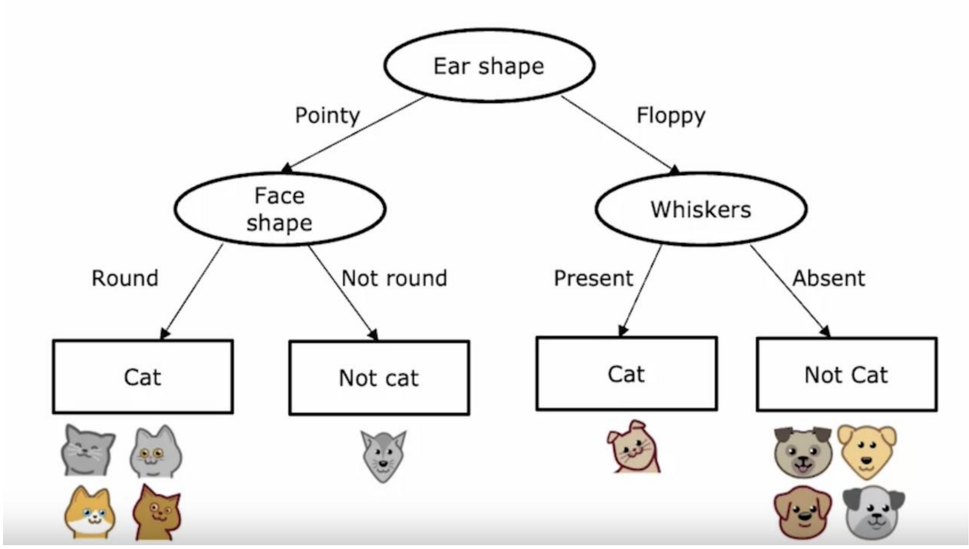






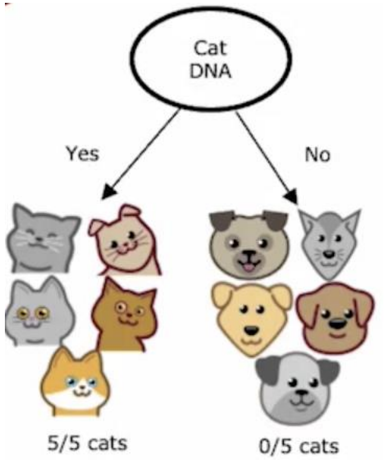
学习过程：续

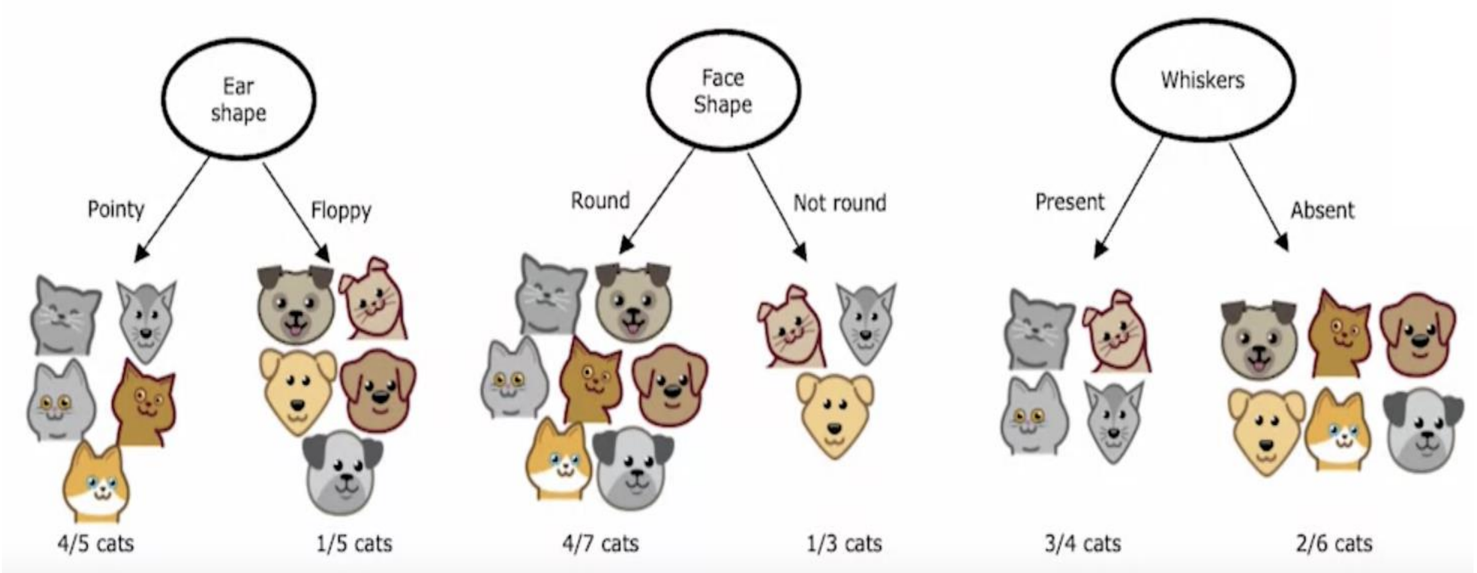

Ear shape
Pointy
Floppy
Face shape
?
Round
Not round
Cat
Not cat


Ear shape
Pointy
Floppy
Face shape
Whiskers
Round
Not round
Present
Absent
Cat
Not cat
Cat
Not Cat

关键问题

- 关键问题1：在每个节点，怎样选择使用哪个特征？
- 最大化纯度（Maximize purity or minimize impurity）

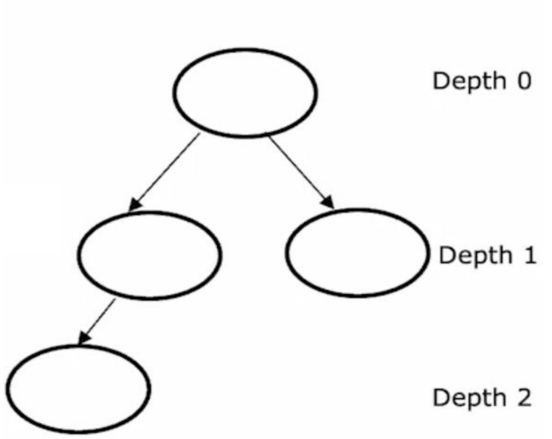


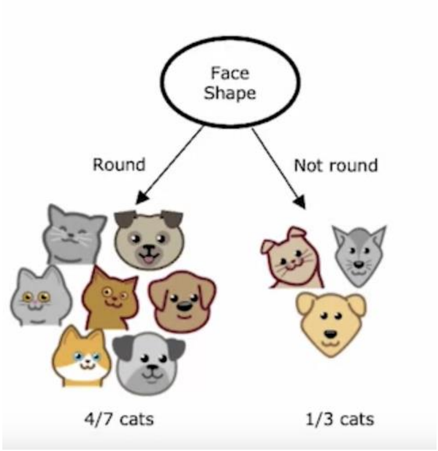


- 那么，上面哪一个特征可以最大化纯度呢？

关键问题

- 关键问题2：什么时候停止拆分节点？
  - 当一个节点中的数据100%属于一个分类
  - 当拆分一个节点会导致决策树超过设定的最大深度
  - 当对于纯度的提升小于给定临界值
  - 当节点中的观测个数少于一定数量





纯度

- 当一个集合中全是猫，我们说该集合纯度很高。
- 当一个集合中全都不是猫、全是狗，我们也说这个集合纯度很高。
- 当一个集合中有猫有狗，该怎么去量化它的纯度呢？

Entropy作为非纯度的测量

- 令p_k表示第k类样本所占的比例
- 例：令p_1表示小猫样本所占的比例

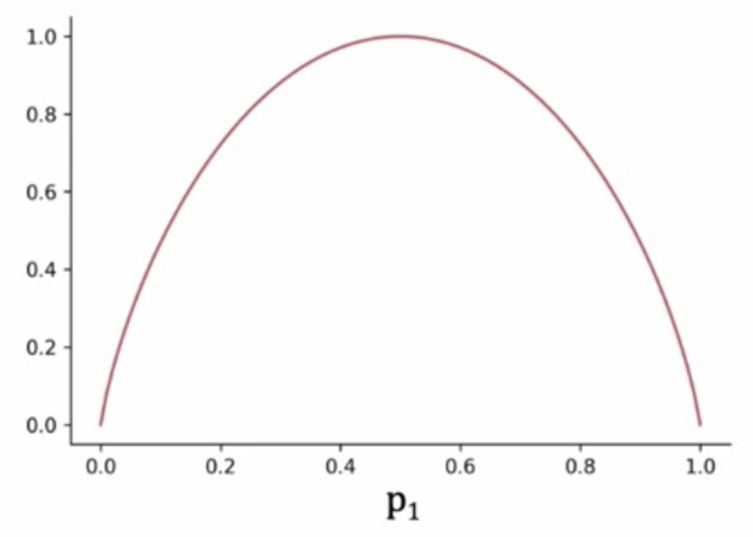
- 我们用信息熵（Information Entropy）来衡量一组数据的非纯度
- 信息熵是信息论中一个非常抽象的概念，是为了量化信息的不确定性而设计的度量。一个系统越是有序，不确定性越低，信息熵就越低；反之，信息熵就越高
- 对一个样本集合D来说，其信息熵定义为

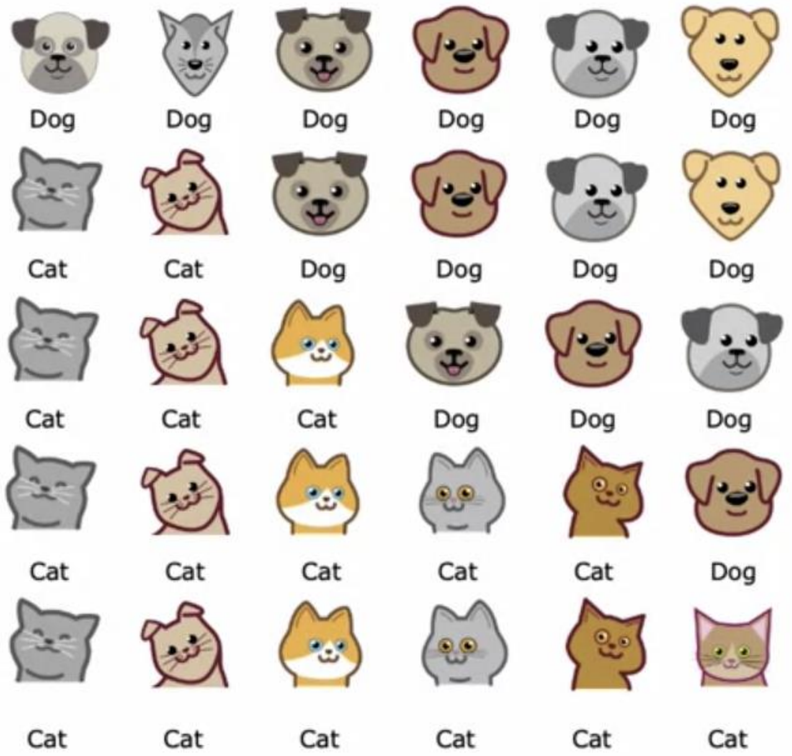
$$H(D) = -\sum_{k} p_k \log_2 p_k$$

- 计算信息熵时约定：若$p_k = 0$，则$p_k \log_2 p_k = 0$

信息熵：例

- $H(D) = -p_1 \log_2 p_1 - p_0 \log_2 p_0 = -p_1 \log_2 p_1 - (1 - p_1) \log_2 (1 - p_1)$





$p_1 = \frac{0}{6}, H(D) = 0$

$p_1 = \frac{2}{6}, H(D) = 0.92$

$p_1 = \frac{1}{2}, H(D) = 1$

$p_1 = \frac{5}{6}, H(D) = 0.65$

$p_1 = \frac{6}{6}, H(D) = 0$

选择划分：信息增益

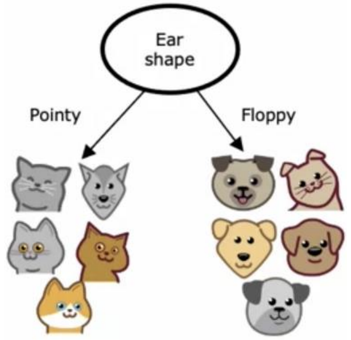
- 构造决策树时，我们选择使用哪个特征进行划分的方式是：

  哪个特征可以最大程度地　　增大纯度 / 减小非纯度 / 减小信息熵

- 在决策树学习中，信息熵的减小被称作信息增益（Information Gain）

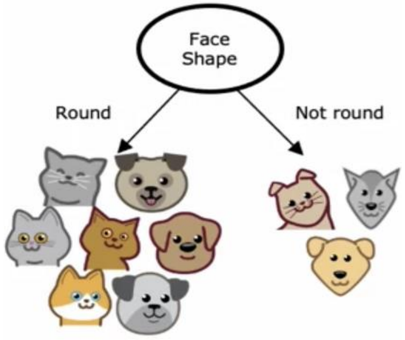
- 特征 A 的信息增益越大，说明信息熵的减小更多，意味着使用特征 A 进行划分所获得的**纯度提升**越大

选择划分：信息增益



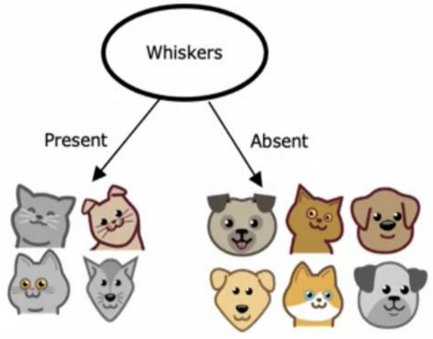
$p_1 = \frac{4}{5} = 0.8$ $p_1 = \frac{1}{5} = 0.2$

$H(D_1) = 0.72$ $H(D_2) = 0.72$



$p_1 = \frac{4}{7} = 0.57$ $p_1 = \frac{1}{3} = 0.33$

$H(D_1) = 0.99$ $H(D_2) = 0.92$



$p_1 = \frac{3}{4} = 0.75$ $p_1 = \frac{2}{6} = 0.33$

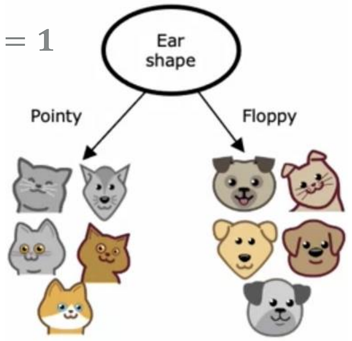
$H(D_1) = 0.81$ $H(D_2) = 0.92$

- 给定这三个划分根节点的选择，哪一个最好呢？
- 与其肉眼观察并比较这些信息熵的值，更好的方式是考虑它们的加权平均
- 背后思想：节点有很多的观测且很高的信息熵，要比有很少的观测且很高的信息熵情况要坏
- 根据分支的观测数量对信息熵加权，观测数量越多，信息熵权重越大

选择划分：信息增益

$p_1 = \frac{5}{10} = 0.5$

$H(D) = 1$



$p_1 = \frac{4}{5} = 0.8$

$H(D_1) = 0.72$

$p_1 = \frac{1}{5} = 0.2$

$H(D_2) = 0.72$

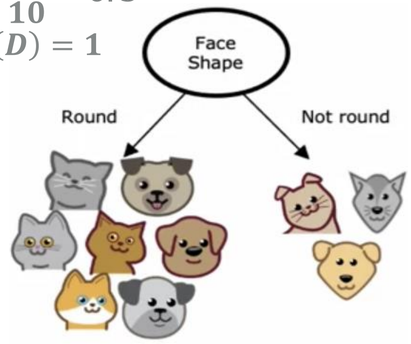
$\frac{5}{10}H(D_1) + \frac{5}{10}H(D_2)$

$H(D) - \left(\frac{5}{10}H(D_1) + \frac{5}{10}H(D_2)\right)$

$= 0.28$

$p_1 = \frac{5}{10} = 0.5$

$H(D) = 1$



$p_1 = \frac{4}{7} = 0.57$

$H(D_1) = 0.99$

$p_1 = \frac{1}{3} = 0.33$

$H(D_2) = 0.92$

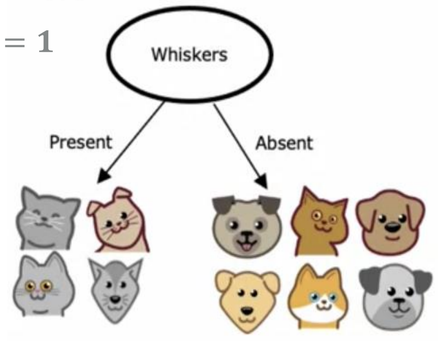
$\frac{7}{10}H(D_1) + \frac{3}{10}H(D_2)$

$H(D) - \left(\frac{7}{10}H(D_1) + \frac{3}{10}H(D_2)\right)$

$= 0.03$

$p_1 = \frac{5}{10} = 0.5$

$H(D) = 1$



$p_1 = \frac{3}{4} = 0.75$

$H(D_1) = 0.81$

$p_1 = \frac{2}{6} = 0.33$

$H(D_2) = 0.92$

$\frac{4}{10}H(D_1) + \frac{6}{10}H(D_2)$

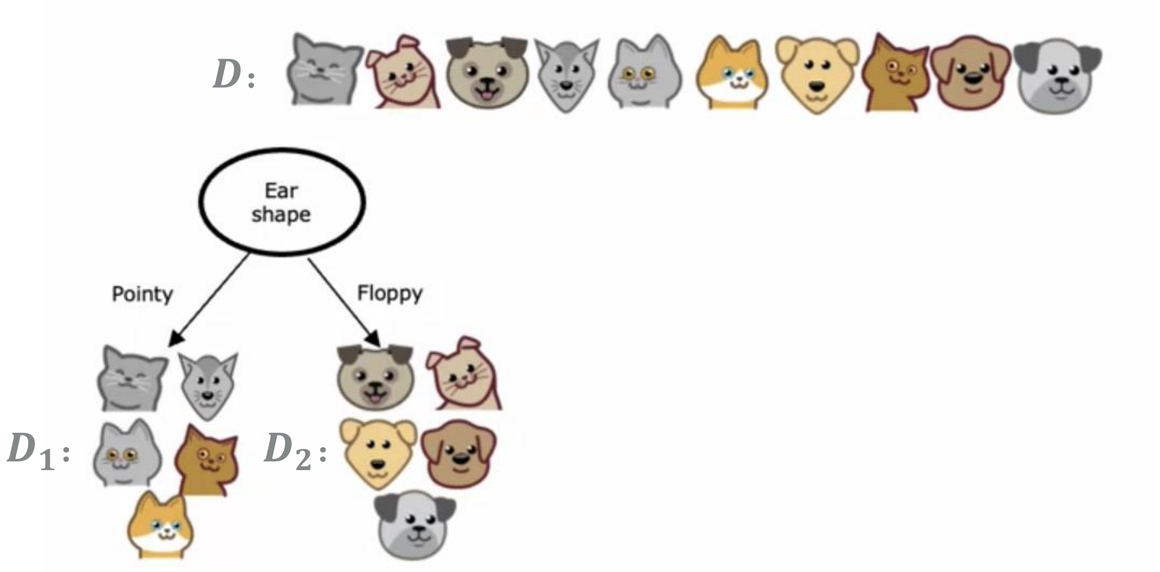
$H(D) - \left(\frac{4}{10}H(D_1) + \frac{6}{10}H(D_2)\right)$

$= 0.12$

信息熵下降最多

信息增益最大

信息增益：正式定义



- 用特征a对样本集D进行划分的信息增益：

$$\mathrm{Gain}(D,a) = H(D) - \sum_{v} \frac{|D_v|}{|D|} H(D_v)$$

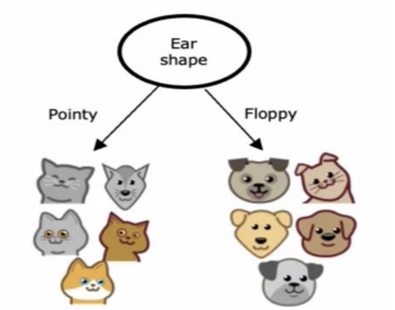
- 通过信息增益，我们就掌握了如何对一个单一的节点选择拆分特征的方法
- 著名的ID3决策树算法（Quinlan, 1986）就是以信息增益为准则来选择划分属性

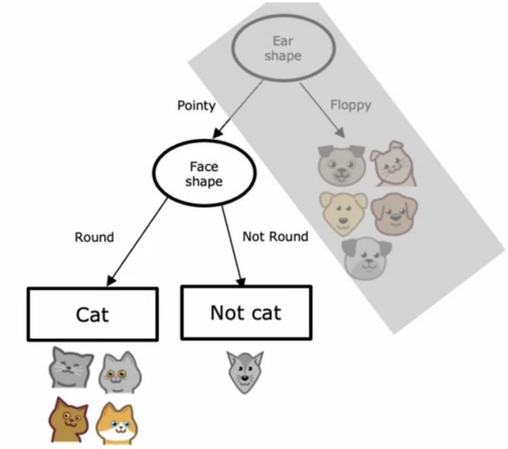
基于信息增益的决策树学习全过程

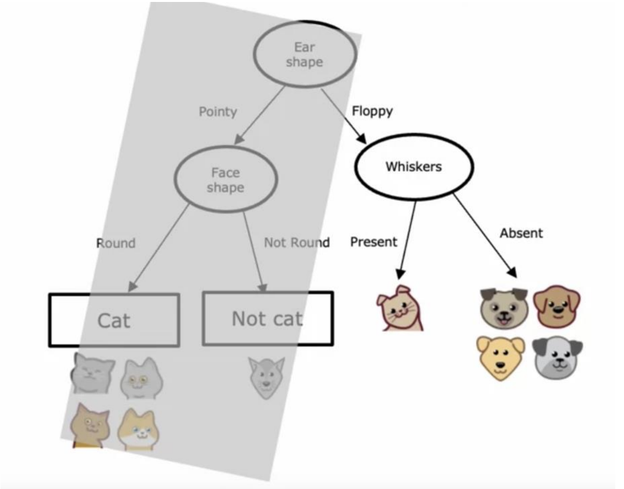
- 从根节点的所有训练数据出发
- 对所有的特征计算信息增益，选取信息增益最大的特征进行划分
- 根据所选择的特征划分训练数据，并创造出决策树的左右分支
- 重复划分过程，直到满足停止准则。停止准则可能是下列的一个或多个的组合：
  - 当一个节点中的数据100%属于一个分类
  - 当拆分一个节点会导致决策树超过设定的最大深度
  - 当拆分的信息增益小于一定的临界值
  - 当节点中的观测数量小于一定的临界值
- 当叶节点中不是100%属于一个分类时，这个叶节点所标识的类别通过大多数原则(majority class criterion)确定

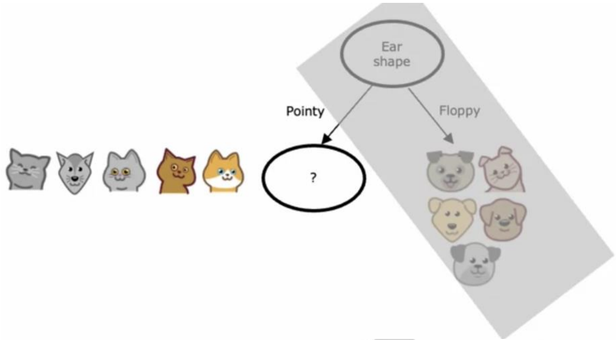
全过程演示

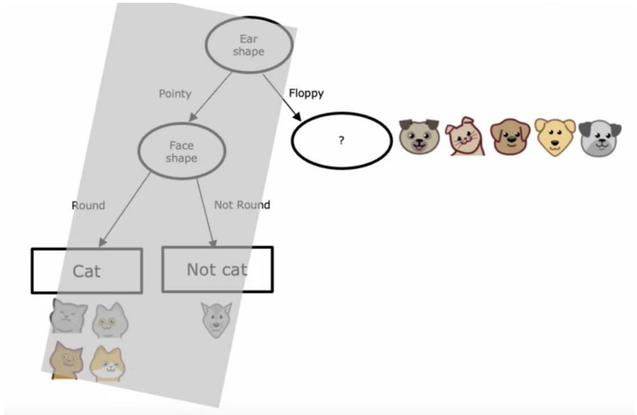

?

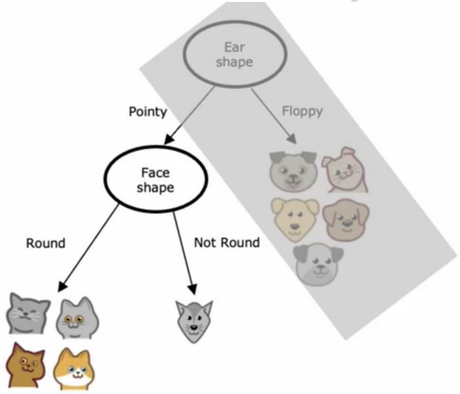

Ear shape
Pointy
Floppy

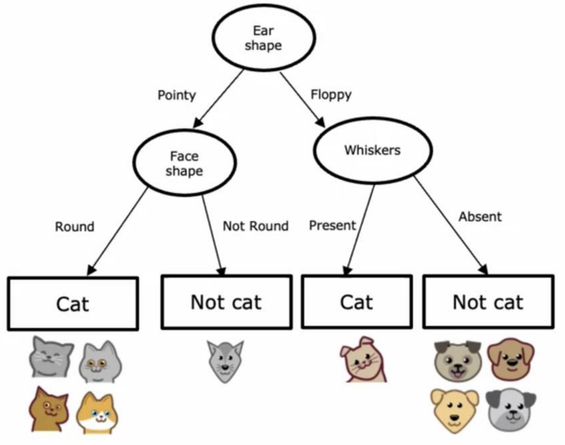

Ear shape
Pointy
Floppy
Face shape
Round
Not Round
Cat
Not cat


Ear shape
Pointy
Floppy
Face shape
Whiskers
Round
Not Round
Present
Absent
Cat
Not cat


Ear shape
Pointy
Floppy
?


Ear shape
Pointy
Floppy
?
Face shape
Round
Not Round
Cat
Not cat


Ear shape
Pointy
Floppy
Face shape
Round
Not Round


Ear shape
Pointy
Floppy
Face shape
Whiskers
Round
Not Round
Present
Absent
Cat
Not cat
Cat
Not cat

讨论

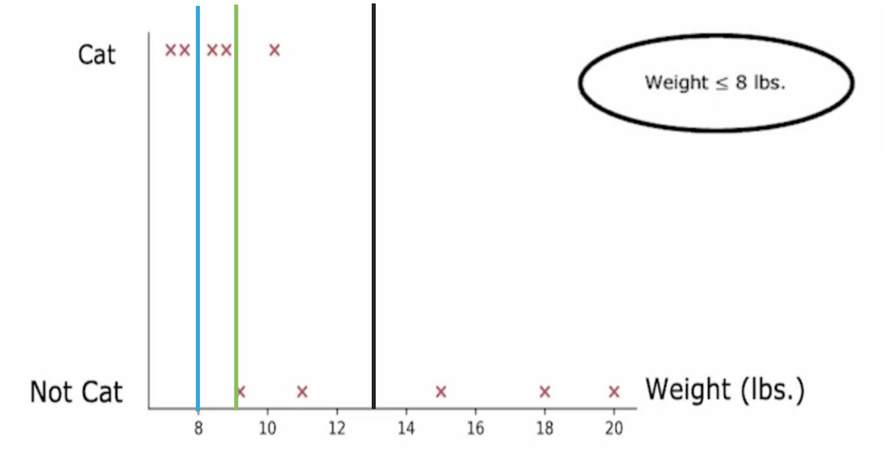
- 在我们决定了使用哪个特征作为根节点之后，我们构造左边和右边子树的方式是：
- 分别对一部分子集和另一部分子集构造决策树
- 这些子节点又可以作为子决策树的根节点，向下延伸子树
- 在计算机科学中，这叫做递归算法（Recursive Algorithm）：在根部构建决策树的方法是在左侧和右侧子分支中构建其他较小的决策树。
- 这也是为什么决策树的软件实现都会涉及到递归算法

连续特征

	Ear shape	Face shape	Whiskers	Weight (lbs.)	Cat
	Pointy	Round	Present	7.2	1
	Floppy	Not round	Present	8.8	1
	Floppy	Round	Absent	15	0
	Pointy	Not round	Present	9.2	0
	Pointy	Round	Present	8.4	1
	Pointy	Round	Absent	7.6	1
	Floppy	Not round	Absent	11	0
	Pointy	Round	Absent	10.2	1
	Floppy	Round	Absent	18	0
	Floppy	Round	Absent	20	0

- 现在除了三个两分类特征以外，我们有了一个连续特征
- 我们要在这四个特征中决定对哪个进行划分
- 怎样对连续特征进行拆分呢？

拆分连续特征

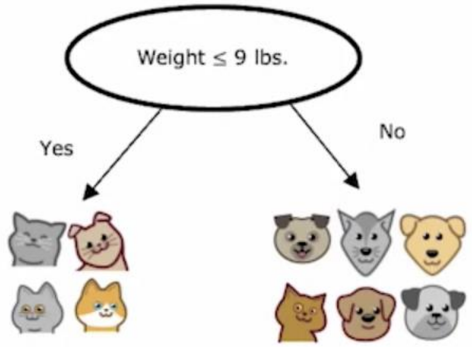


$$H(D) - \left(\frac{2}{10}H(D_1) + \frac{8}{10}H(D_2)\right) = 0.24$$

$$H(D) - \left(\frac{4}{10}H(D_1) + \frac{6}{10}H(D_2)\right) = 0.61$$

$$H(D) - \left(\frac{7}{10}H(D_1) + \frac{3}{10}H(D_2)\right) = 0.40$$

- 在实际中，我们会考虑更多的拆分临界值，从中选取一个信息增益最大的
- 一个常用的做法是把所有数据根据连续特征的大小排序，选择所有两个观测之间的中位点
- 如果在最优临界值点处的信息增益大于其他所有的特征得到的信息增益，那么我们就选择这个连续变量进行拆分



决策树优缺点

- 决策树非常好解释
- 一些人认为决策树最能近似刻画人类决策的真实过程
- 决策树可以可视化，可以轻易地对非专业人士解释
- 决策树可以轻松处理分类特征而不用创造哑变量
- 不幸的是，决策树往往没有其他分类方法的预测精度
- 另外，决策树可能非常不稳健。 换句话说，数据的一个小变化可能会导致最终估计的树发生大的变化。
- 然而，通过聚合许多决策树，使用 bagging、随机森林和 boosting 等方法，可以显著提高树的预测性能。

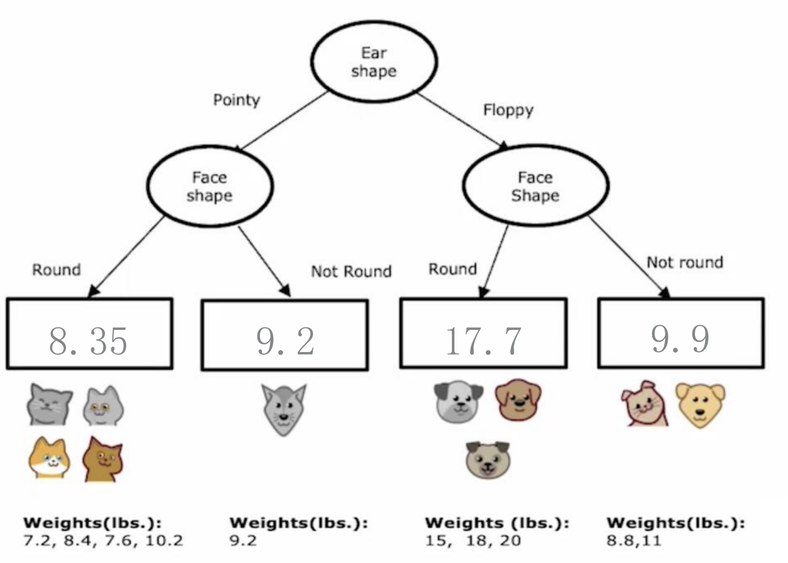
回归树

- 目前为止，我们都是用决策树解决分类问题
- 决策树能不能解决回归问题呢？

	Ear shape	Face shape	Whiskers	Weight (lbs.)
	Pointy	Round	Present	7.2
	Floppy	Not round	Present	8.8
	Floppy	Round	Absent	15
	Pointy	Not round	Present	9.2
	Pointy	Round	Present	8.4
	Pointy	Round	Absent	7.6
	Floppy	Not round	Absent	11
	Pointy	Round	Absent	10.2
	Floppy	Round	Absent	18
	Floppy	Round	Absent	20

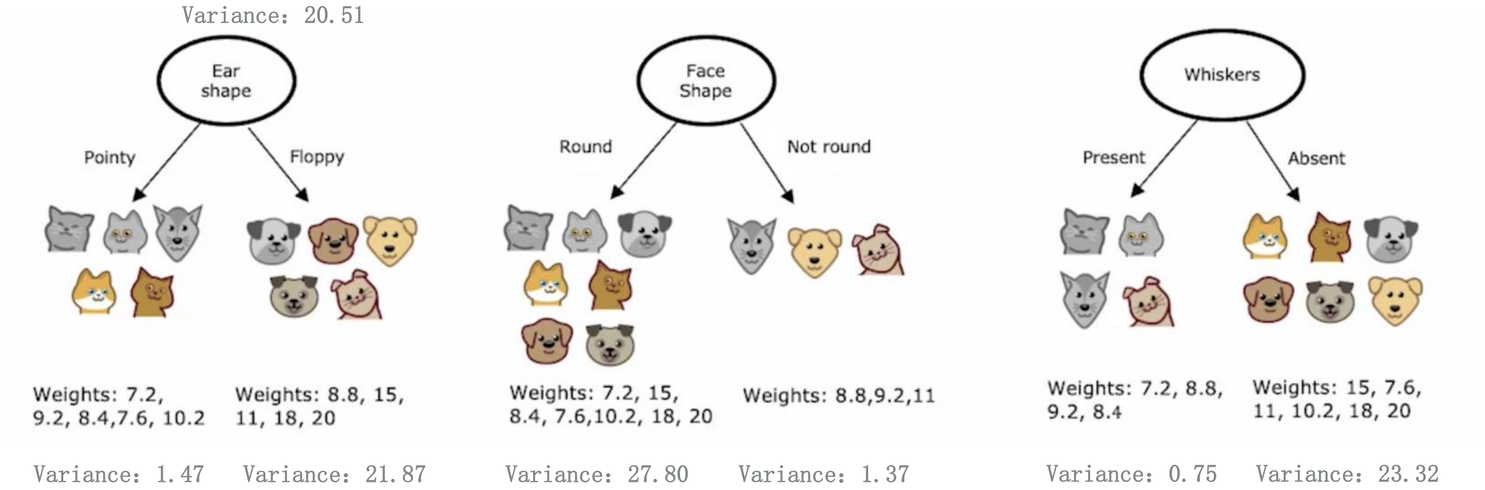
回归树

- 假设我们已经构建好了一个决策树如图所示
- 这时如果有一个新的数据（Pointy & Round），我们该如果对其响应变量进行预测呢
- 预测值：所对应的叶节点当中训练数据的响应变量平均值



回归树：选择拆分

- 回归树不再根据信息熵来选择拆分节点，而是根据**方差**来选择拆分节点
- 方差越大，表示该节点的数据越分散，越不像是来自同一类，预测效果就越差。
- 准则：选择具有最大方差下降的拆分特征



$$20.51 - \left(\frac{5}{10} * 1.47 + \frac{5}{10} * 21.87\right) = 8.84$$

$$20.51 - \left(\frac{7}{10} * 27.80 + \frac{3}{10} * 1.37\right) = 0.64$$

$$20.51 - \left(\frac{4}{10} * 0.75 + \frac{5}{10} * 23.32\right) = 6.22$$

过拟合

- 可以想像，如果有非常多的特征，最终完整的决策树可能非常庞大，有很多叶节点
- 在决策树学习中，为了尽可能正确分类训练样本，节点划分过程将不断重复，造成分支过多
- 此时，对训练数据学习的太好，很容易导致过拟合问题
- 一个更小的决策树有着更少的节点，可以在牺牲一点偏差的代价下带来更低的方差和更强的可解释性
- 对决策树进行剪枝可以有效地做到这一点，从而解决过拟合

决策树剪枝（Pruning）

- 决策树的剪枝有两种思路：预剪枝（pre-pruning）和后剪枝（post-pruning）
- 预剪枝：在构造决策树的同时进行剪枝。
  - 在降低信息熵的过程中，为了避免过拟合，可以设定一个阈值，熵的减小量如果大于这个阈值，则继续创建分支。
  - 也可以是别的指标：如比较剪枝前后的分类错误率或RSS，前提是有一个测试集
  - 这样的方法在实际中经常表现不好，过于短视：一个没必要拆分的节点可能紧接着一个非常值得拆分的节点。
- 后剪枝：先从训练集生成一颗完整的决策树，从下往上对非叶节点进行考察
  - 如果一个结点满足剪枝条件，则将该结点对应的子树替换为叶节点

后剪枝：Cost complexity pruning

- **代价复杂度剪枝**不需要额外的测试数据集
- 对任意子树$T < T_{max}$，我们将其复杂度定义为$|T|$，即T中叶节点的数量
- 令$\alpha \geq 0$代表复杂度正则化参数，每一个α的取值都对应着一个子树T使得

$$R_\alpha(T) = R(T) + \alpha|T|$$

尽可能小。$R(T)$ 可以是 T 的训练分类错误率，RSS等指标

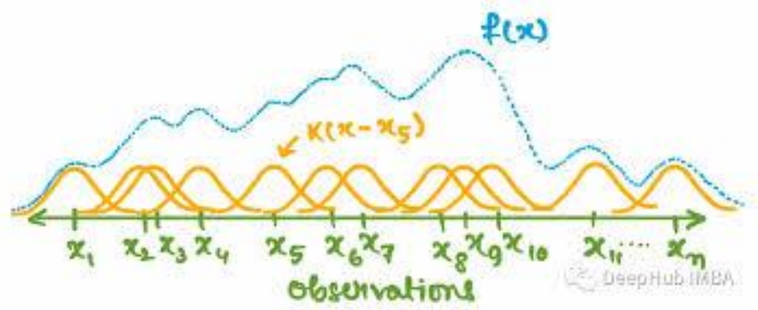
- α控制了子树T的复杂度和对训练数据拟合效果之间的平衡
  - $\alpha = 0$时，没有任何正则化，得到的就是完整的决策树
  - 当α增大时，一个有很多叶节点的树要付出更多代价，所以最小化$R_\alpha(T)$会倾向于得到更小的子树
  - 选择α：交叉验证

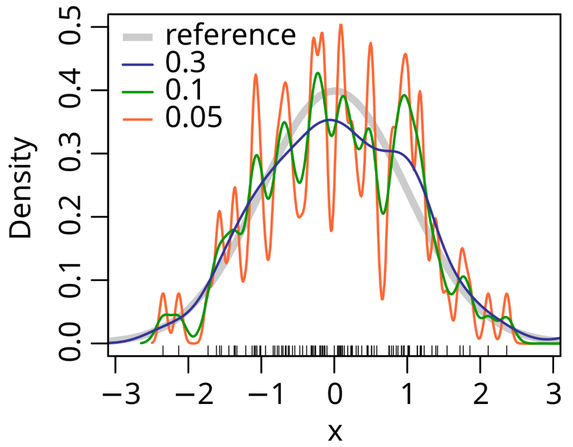
几种非参回归方法的比较

- 非参数回归
  - $\mathbb{E}[Y \mid \boldsymbol{X} = \boldsymbol{x}] = m(\boldsymbol{x}), \boldsymbol{X} \sim \boldsymbol{p}(\boldsymbol{X})$
  - 函数$m(\boldsymbol{x})$的具体形式未知（e.g. 线性模型），理论推导通常对其光滑性进行假设(e.g. 属于Sobolev空间，Hölder 空间)
- 采用不同的方法对$m(\boldsymbol{x})$建模
  - 决策树 (Decision tree)
  - 核光滑 (kernel smoothing method)
  - k近邻 (k-nearest neighbours)

核光滑（KERNEL SOOTHING METHOD）

- $m(\boldsymbol{x}) = \mathbb{E}[Y \mid \boldsymbol{X} = \boldsymbol{x}] = \frac{\int yp(\boldsymbol{x},y)dy}{p(\boldsymbol{x})}$
- 估计两个密度函数$p(\boldsymbol{x}), p(\boldsymbol{x},y)$
  - 核密度估计（kernel density estimation）
  - $\hat{p}(\boldsymbol{x}) = \frac{1}{nh}\sum_{i=1}^{n} K(\frac{\boldsymbol{x}_i - \boldsymbol{x}}{h})$
  - h：窗宽，　$K(\cdot)$：核函数
  - 超参数h控制平滑程度，h越大越平滑





核光滑（KERNEL SOOTHING METHOD）

- 将估计量代入：$m(\boldsymbol{x}) = \mathbb{E}[\, Y \mid \boldsymbol{X} = \boldsymbol{x}\,] = \frac{\int y p(\boldsymbol{x},y)dy}{p(\boldsymbol{x})}$
  - $\hat{m}_{ks}(\boldsymbol{x}) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x_i - \boldsymbol{x}}{h}\right) \cdot y_i$
- 写成加权平均的形式
  - $\hat{m}_{ks}(\boldsymbol{x}) = \frac{1}{n}\sum_{i=1}^{n} w_{ks}(\boldsymbol{x_i}) \cdot y_i$
  - 其中$w_{ks}(\boldsymbol{x_i}) = \frac{1}{h}K(\frac{x_i - \boldsymbol{x}}{h})$是$\frac{p(\boldsymbol{x},y)}{p(\boldsymbol{x})}$的估计
- 超参数h的选取需要做交叉验证

几种非参回归方法的比较

- 决策树
  - $\hat{m}_{dt}(\boldsymbol{x}) = \frac{1}{n_x}\sum_{\mathrm{i}=1}^{n} \mathbb{I}_{\{x_i \in \mathcal{A}_x\}} \cdot y_i$，$n_x = \#\{\boldsymbol{x}_i \in \mathcal{A}_x\}$
  - $\mathcal{A}_{\boldsymbol{x}}$为$\boldsymbol{x}$所在的叶子节点区域，$n_{\boldsymbol{x}}$为该区域的样本量
- 核光滑
  - $\hat{m}_{ks}(\boldsymbol{x}; h) = \frac{1}{n}\sum_{i=1}^{n} \frac{1}{h} K\left(\frac{x_i - \boldsymbol{x}}{h}\right) \cdot y_i$
- k近邻
  - $\hat{m}_{knn}(\boldsymbol{x}; k) = \frac{1}{k}\sum_{i=1}^{n} \mathbb{I}_{\{x_i \in \mathcal{N}_k\}} \cdot y_i$
  - $\mathcal{N}_k$为距离$\boldsymbol{x}$最近的k个点的集合

几种非参回归方法的比较

- 三种非参回归方法都是加权平均的形式，不同点在于权重的定义
  - 三者权重都涉及到超参数的选取
  - 决策树的权重是数据驱动的（data adaptive）
  - 核光滑的权重涉及到密度估计，当$\boldsymbol{x}$的维度很高时效果不好
  - k近邻的权重与定义的度量有关，需要对数据有一定的先验信息
  - 因此当我们对数据没有足够的先验信息并且$\boldsymbol{x}$的维度很高时，决策树具有一定的优势

本章小节

- 决策树学习过程
- 拆分特征选择
- 信息熵、信息增益
- 连续特征
- 回归树
- 树剪枝
- ……