

分类回归树与随机森林

Machine Learning Engineer

机器学习工程师

寒小阳

目录

CONTENTS

- 01 连续值和缺省值的处理
- 02 回归树模型
- 03 随机森林
- 04 数据案例讲解



01 连续值和缺省值的处理

1.1 连续值处理

1.2 缺省值处理

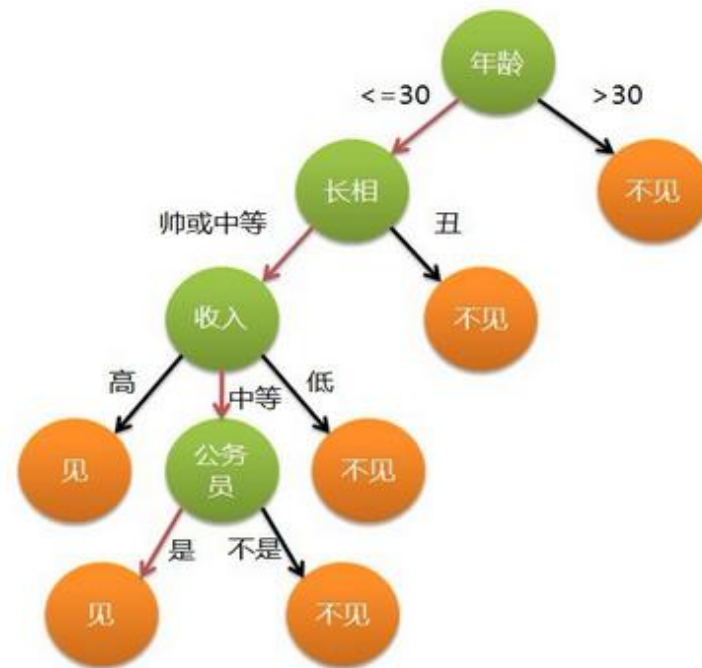
决策树模型

决策树基于“树”结构进行决策

- 每个“内部结点”对应于某个属性上的“测试”
- 每个分支对应于该测试的一种可能结果（即该属性的某个取值）
- 每个“叶结点”对应于一个“预测结果”

学习过程：通过对训练样本的分析来确定“划分属性”（即内部结点所对应的属性）

预测过程：将测试示例从根结点开始，沿着划分属性所构成的“判定测试序列”下行，直到叶结点



决策树基本流程

总体流程：

“分而治之” (divide-and-conquer)

- 自根至叶的递归过程
- 在每个中间结点寻找一个“划分” (split or test) 属性

三种停止条件：

1. 当前结点包含的样本全属于同一类别，无需划分；
2. 当前属性集为空，或是所有样本在所有属性上取值相同，无法划分；
3. 当前结点包含的样本集合为空，不能划分。

决策树基本流程

输入: 训练集 $D = \{(x_1, y_1), (x_2, y_2), \dots, (x_m, y_m)\}$;

属性集 $A = \{a_1, a_2, \dots, a_d\}$.

过程: 函数 TreeGenerate(D, A)

1: 生成结点 node;

2: **if** D 中样本全属于同一类别 C **then**

3: 将 node 标记为 C 类叶结点; **return**

4: **end if**

前面的(1)情形
递归返回

5: **if** $A = \emptyset$ **OR** D 中样本在 A 上取值相同 **then**

6: 将 node 标记为叶结点, 其类别标记为 D 中样本数最多的类; **return**

7: **end if**

前面的(2)情形
递归返回

8: 从 A 中选择最优划分属性 a_* ;

利用当前结点的后验分布

9: **for** a_* 的每一个值 a_*^v **do**

10: 为 node 生成一个分支; 令 D_v 表示 D 中在 a_* 上取值为 a_*^v 的样本子集;

11: **if** D_v 为空 **then**

12: 将分支结点标记为叶结点, 其类别标记为 D 中样本最多的类; **return**

13: **else**

前面的(3)情形
递归返回

14: 以 TreeGenerate($D_v, A \setminus \{a_*\}$) 为分支结点

将父结点的样本分布作为
当前结点的先验分布

15: **end if**

16: **end for**

决策树算法的
核心

输出: 以 node 为根结点的一棵决策树

如果数据中有连续值，如何处理？

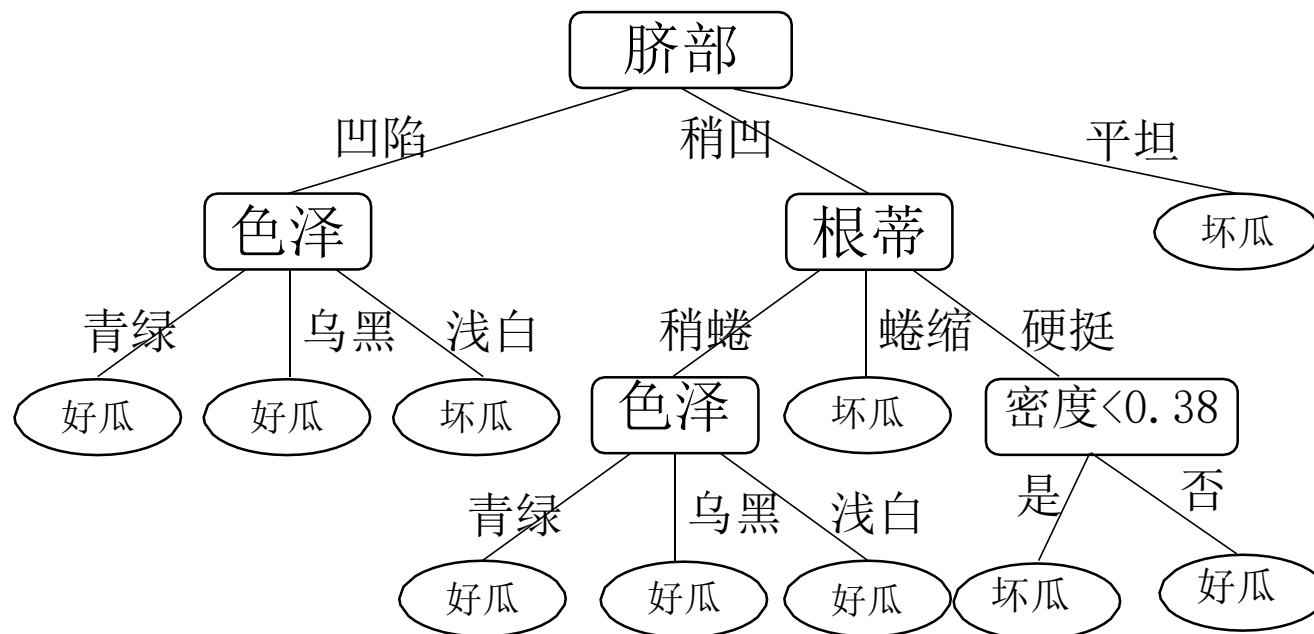
周志华老师《机器学习》西瓜数据集

| 编号 | 色泽 | 根蒂 | 敲声 | 纹理 | 脐部 | 触感 | 密度 | 含糖率 | 好瓜 |
|----|----|----|----|----|----|----|-------|-------|----|
| 1 | 青绿 | 蜷缩 | 浊响 | 清晰 | 凹陷 | 硬滑 | 0.697 | 0.460 | 是 |
| 2 | 乌黑 | 蜷缩 | 沉闷 | 清晰 | 凹陷 | 硬滑 | 0.774 | 0.376 | 是 |
| 3 | 乌黑 | 蜷缩 | 浊响 | 清晰 | 凹陷 | 硬滑 | 0.634 | 0.264 | 是 |
| 4 | 青绿 | 蜷缩 | 沉闷 | 清晰 | 凹陷 | 硬滑 | 0.608 | 0.318 | 是 |
| 5 | 浅白 | 蜷缩 | 浊响 | 清晰 | 凹陷 | 硬滑 | 0.556 | 0.215 | 是 |
| 6 | 青绿 | 稍蜷 | 浊响 | 清晰 | 稍凹 | 软粘 | 0.403 | 0.237 | 是 |
| 7 | 乌黑 | 稍蜷 | 浊响 | 稍糊 | 稍凹 | 软粘 | 0.481 | 0.149 | 是 |
| 8 | 乌黑 | 稍蜷 | 浊响 | 清晰 | 稍凹 | 硬滑 | 0.437 | 0.211 | 是 |
| 9 | 乌黑 | 稍蜷 | 沉闷 | 稍糊 | 稍凹 | 硬滑 | 0.666 | 0.091 | 否 |
| 10 | 青绿 | 硬挺 | 清脆 | 清晰 | 平坦 | 软粘 | 0.243 | 0.267 | 否 |
| 11 | 浅白 | 硬挺 | 清脆 | 模糊 | 平坦 | 硬滑 | 0.245 | 0.057 | 否 |
| 12 | 浅白 | 蜷缩 | 浊响 | 模糊 | 平坦 | 软粘 | 0.343 | 0.099 | 否 |
| 13 | 青绿 | 稍蜷 | 浊响 | 稍糊 | 凹陷 | 硬滑 | 0.639 | 0.161 | 否 |
| 14 | 浅白 | 稍蜷 | 沉闷 | 稍糊 | 凹陷 | 硬滑 | 0.657 | 0.198 | 否 |
| 15 | 乌黑 | 稍蜷 | 浊响 | 清晰 | 稍凹 | 软粘 | 0.360 | 0.370 | 否 |
| 16 | 浅白 | 蜷缩 | 浊响 | 模糊 | 平坦 | 硬滑 | 0.593 | 0.042 | 否 |
| 17 | 青绿 | 蜷缩 | 沉闷 | 稍糊 | 稍凹 | 硬滑 | 0.719 | 0.103 | 否 |

基本思路：连续属性离散化

常见做法：二分法 (bi-partition)

- n 个属性值可形成 $(n-1)$ 个候选划分
- 把候选划分值当做离散属性处理，寻找最佳划分



现实应用中，经常会遇到属性值“缺失” (missing) 现象

只使用没有缺失值的样本/属性？

- 会造成数据的极大浪费

如果使用带缺失值的样例，需解决几个问题：

Q1：如何进行划分属性选择？

Q2：给定划分属性，若样本在该属性上的值缺失，如何进行划分？

基本思路：样本赋权，权重划分

分辨西瓜的例子

仅通过无缺失值的样例来判断划分属性的优劣

学习开始时，根结点包含样例集 D 中全部17个 样例，权重均为 1

以属性“色泽”为例，该属性上无缺失值的样例子集 \tilde{D} 包含14个样本，信息熵为

$$\text{Ent}(\tilde{D}) = - \sum_{k=1}^2 \tilde{p}_k \log_2 \tilde{p}_k = -(\frac{6}{14} \log_2 \frac{6}{14} + \frac{8}{14} \log_2 \frac{8}{14}) = 0.985$$

有缺失值的西瓜数据集

| 编号 | 色泽 | 根蒂 | 敲声 | 纹理 | 脐部 | 触感 | 好瓜 |
|----|----|----|----|----|----|----|----|
| 1 | - | 蜷缩 | 浊响 | 清晰 | 凹陷 | 硬滑 | 是 |
| 2 | 乌黑 | 蜷缩 | 沉闷 | 清晰 | 凹陷 | - | 是 |
| 3 | 乌黑 | 蜷缩 | - | 清晰 | 凹陷 | 硬滑 | 是 |
| 4 | 青绿 | 蜷缩 | 沉闷 | 清晰 | 凹陷 | 硬滑 | 是 |
| 5 | - | 蜷缩 | 浊响 | 清晰 | 凹陷 | 硬滑 | 是 |
| 6 | 青绿 | 稍蜷 | 浊响 | 清晰 | - | 软粘 | 是 |
| 7 | 乌黑 | 稍蜷 | 浊响 | 稍糊 | 稍凹 | 软粘 | 是 |
| 8 | 乌黑 | 稍蜷 | 浊响 | - | 稍凹 | 硬滑 | 是 |
| 9 | 乌黑 | - | 沉闷 | 稍糊 | 稍凹 | 硬滑 | 否 |
| 10 | 青绿 | 硬挺 | 清脆 | - | 平坦 | 软粘 | 否 |
| 11 | 浅白 | 硬挺 | 清脆 | 模糊 | 平坦 | - | 否 |
| 12 | 浅白 | 蜷缩 | - | 模糊 | 平坦 | 软粘 | 否 |
| 13 | - | 稍蜷 | 浊响 | 稍糊 | 凹陷 | 硬滑 | 否 |
| 14 | 浅白 | 稍蜷 | 沉闷 | 稍糊 | 凹陷 | 硬滑 | 否 |
| 15 | 乌黑 | 稍蜷 | 浊响 | 清晰 | - | 软粘 | 否 |
| 16 | 浅白 | 蜷缩 | 浊响 | 模糊 | 平坦 | 硬滑 | 否 |
| 17 | 青绿 | - | 沉闷 | 稍糊 | 稍凹 | 硬滑 | 否 |

分辨西瓜的例子

令 $\tilde{D}^1, \tilde{D}^2, \tilde{D}^3$ 分别表示在属性“色泽”上取值为“青绿”“乌黑”以及“浅白”的样本子集，有

$$\begin{aligned} \text{Ent}(\tilde{D}^1) &= -\left(\frac{2}{4}\log_2\frac{2}{4} + \frac{2}{4}\log_2\frac{2}{4}\right) = 1.000 & \text{Ent}(\tilde{D}^2) &= -\left(\frac{4}{6}\log_2\frac{4}{6} + \frac{2}{6}\log_2\frac{2}{6}\right) = 0.918 \\ \text{Ent}(\tilde{D}^3) &= -\left(\frac{0}{4}\log_2\frac{0}{4} + \frac{4}{4}\log_2\frac{4}{4}\right) = 0.000 \end{aligned}$$

因此，样本子集 \tilde{D} 上属性“色泽”的信息增益为

$$\begin{aligned} \text{Gain}(\tilde{D}, \text{色泽}) &= \text{Ent}(\tilde{D}) - \sum_{v=1}^3 \tilde{r}_v \text{Ent}(\tilde{D}^v) \\ &= 0.985 - \left(\frac{4}{14} \times 1.000 + \frac{6}{14} \times 0.918 + \frac{4}{14} \times 0.000\right) \\ &= 0.306 \end{aligned}$$

无缺失值样例中属性 a 取值为 v 的占比

于是，样本集 D 上属性“色泽”的信息增益为

$$\text{Gain}(D, \text{色泽}) = \rho \times \text{Gain}(\tilde{D}, \text{色泽}) = \frac{14}{17} \times 0.306 = 0.252$$

无缺失值样例占比


1.2 缺失值处理


类似地可计算出所有属性在数据集上的信息增益


$$\text{Gain}(D, \text{色泽}) = 0.252 \quad \text{Gain}(D, \text{根蒂}) = 0.171$$

$$\text{Gain}(D, \text{敲声}) = 0.145 \quad \text{Gain}(D, \text{纹理}) = 0.424$$


$$\text{Gain}(D, \text{脐部}) = 0.289 \quad \text{Gain}(D, \text{触感}) = 0.006$$

 进入“纹理=清晰”分支 进

 入“纹理=稍糊”分支 进入

 “纹理=模糊”分支

样本权重在各子结点仍为1

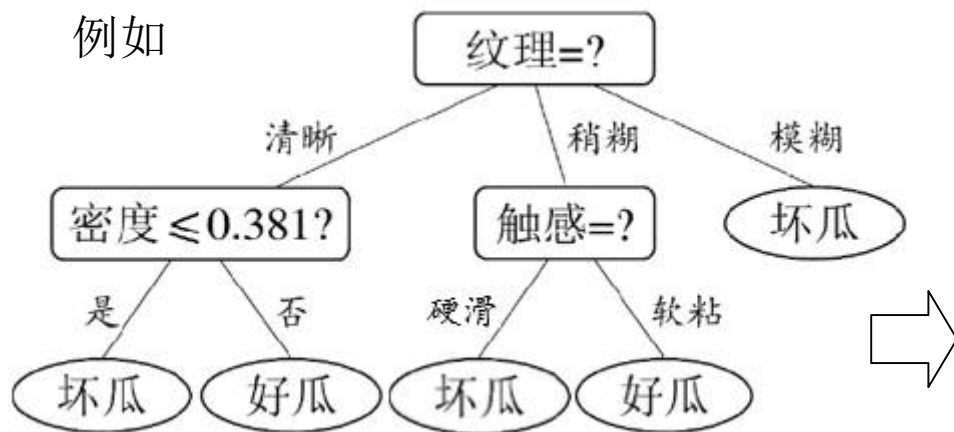
 在“纹理”上出现缺失值，
样本 8, 10 同时进入三个分支，三支上的权重分别为
7/15, 5/15, 3/15

| 编号 | 色泽 | 根蒂 | 敲声 | 纹理 | 脐部 | 触感 | 好瓜 |
|----|----|----|----|----|----|----|----|
| 1 | — | 蜷缩 | 浊响 | 清晰 | 凹陷 | 硬滑 | 是 |
| 2 | 乌黑 | 蜷缩 | 沉闷 | 清晰 | 凹陷 | — | 是 |
| 3 | 乌黑 | 蜷缩 | — | 清晰 | 凹陷 | 硬滑 | 是 |
| 4 | 青绿 | 蜷缩 | 沉闷 | 清晰 | 凹陷 | 硬滑 | 是 |
| 5 | — | 蜷缩 | 浊响 | 清晰 | 凹陷 | 硬滑 | 是 |
| 6 | 青绿 | 稍蜷 | 浊响 | 清晰 | — | 软粘 | 是 |
| 7 | 乌黑 | 稍蜷 | 浊响 | 稍糊 | 稍凹 | 软粘 | 是 |
| 8 | 乌黑 | 稍蜷 | 浊响 | — | 稍凹 | 硬滑 | 是 |
| 9 | 乌黑 | — | 沉闷 | 稍糊 | 稍凹 | 硬滑 | 否 |
| 10 | 青绿 | 硬挺 | 清脆 | — | 平坦 | 软粘 | 否 |
| 11 | 浅白 | 硬挺 | 清脆 | 模糊 | 平坦 | — | 否 |
| 12 | 浅白 | 蜷缩 | — | 模糊 | 平坦 | 软粘 | 否 |
| 13 | — | 稍蜷 | 浊响 | 稍糊 | 凹陷 | 硬滑 | 否 |
| 14 | 浅白 | 稍蜷 | 沉闷 | 稍糊 | 凹陷 | 硬滑 | 否 |
| 15 | 乌黑 | 稍蜷 | 浊响 | 清晰 | — | 软粘 | 否 |
| 16 | 浅白 | 蜷缩 | 浊响 | 模糊 | 平坦 | 硬滑 | 否 |
| 17 | 青绿 | — | 沉闷 | 稍糊 | 稍凹 | 硬滑 | 否 |

权重划分

从树到规则

- 一棵决策树对应于一个“规则集”
- 每个从根结点到叶结点的分支路径对应于一条规则



好处:

- 改善可理解性
- 进一步提升泛化能力

- IF (纹理=清晰) \wedge (密度 ≤ 0.381) THEN 坏瓜
- IF (纹理=清晰) \wedge (密度 > 0.381) THEN 好瓜
- IF (纹理=稍糊) \wedge (触感=硬滑) THEN 坏瓜
- IF (纹理=稍糊) \wedge (触感=软粘) THEN 好瓜
- IF (纹理=模糊) THEN 坏瓜

要点总结

- 连续值处理

- 二分思路

- n 个属性值可形成 $n-1$ 个候选划分,
当做离散值来处理

- 缺失值处理

- 样本赋权, 权重划分



02

回归树模型

2.1

回归树模型

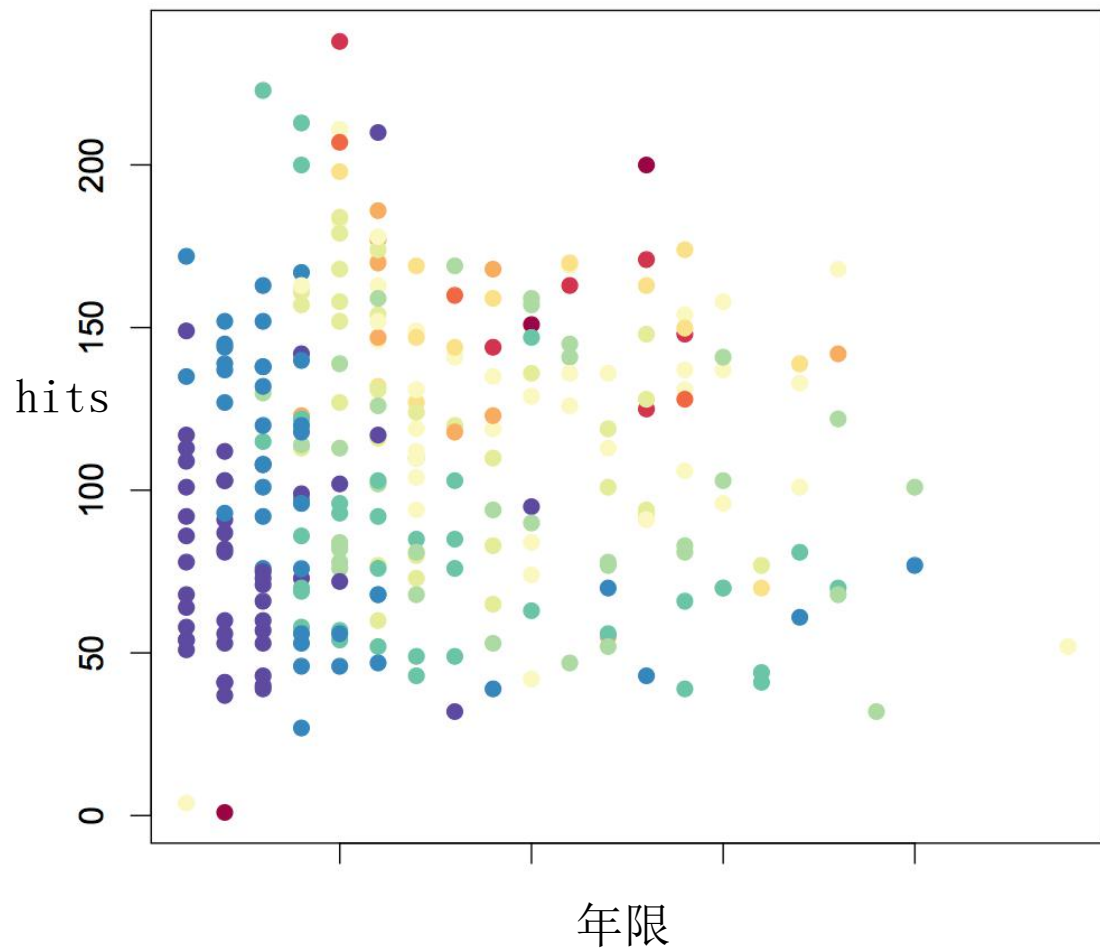
2.2

回归树构建方法

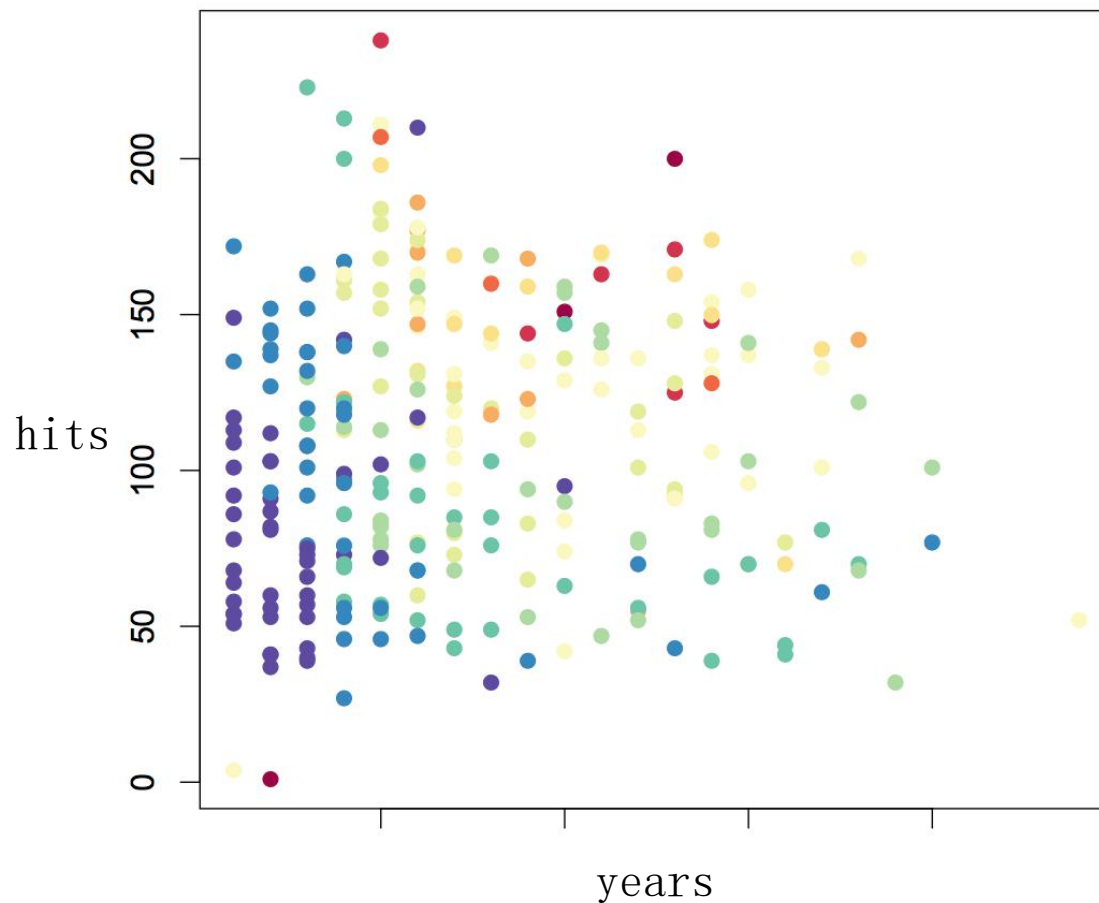
经典案例：

根据从业年限和表现，去预估棒球运动员的工资。

如右图所示，有1987个数据样本，包含322个棒球运动员。红黄表示高收入，蓝绿表示低收入。横坐标是年限，纵坐标是表现。



我们最终可以得到下面这样一颗回归树：



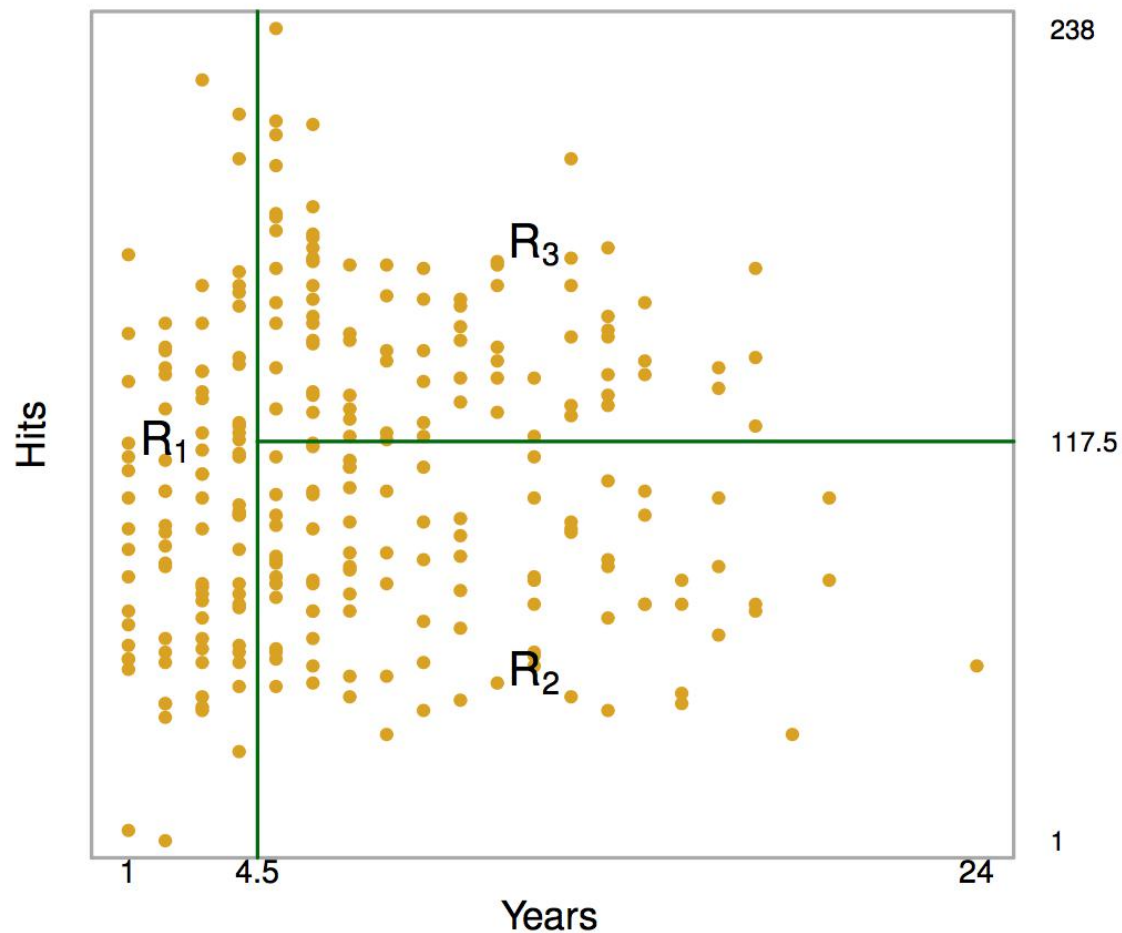
回归树背后的含义：

对空间的划分，整个平面被划分成3部分。

$$R_1 = \{X \mid \text{Years} < 4.5\},$$

$$R_2 = \{X \mid \text{Years} \geq 4.5, \text{Hits} < 117.5\}.$$

$$R_3 = \{X \mid \text{Years} \geq 4.5, \text{Hits} \geq 117.5\}$$



我们来正式介绍一下回归树的构建方法。

假设一个回归问题，预估结果 $y \in R$ ，特征向量为 $X = [x_1, x_2, x_3 \dots x_p] \in R$ ，回归树的2个步骤是：

1. 把整个特征空间 X 切分成 J 个没有重叠的区域 $R_1, R_2, R_3 \dots R_J$
2. 其中区域 R_j 中的每个样本我们都给一样的预测结果 $\tilde{y}_{R_j} = \frac{1}{n} \sum_{i \in R_j} y_i$ ，其中 n 是 R_j 中的总样本数。

我们仔细观察一下上面的①过程，实际上我们希望能找到如下的RSS最小的划分方式 $R_1, R_2, R_3 \dots R_J$

$$RSS = \sum_{j=1}^J \sum_{i \in R_j} (y_i - \tilde{y}_{R_j})^2$$

但是这个最小化和探索的过程，计算量是非常非常大的。

我们采用探索式的递归二分来尝试解决这个问题。

递归二分

- 自顶向下的贪婪式递归方案

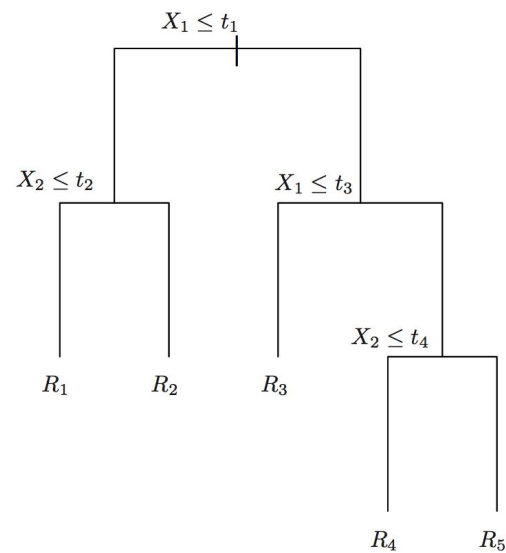
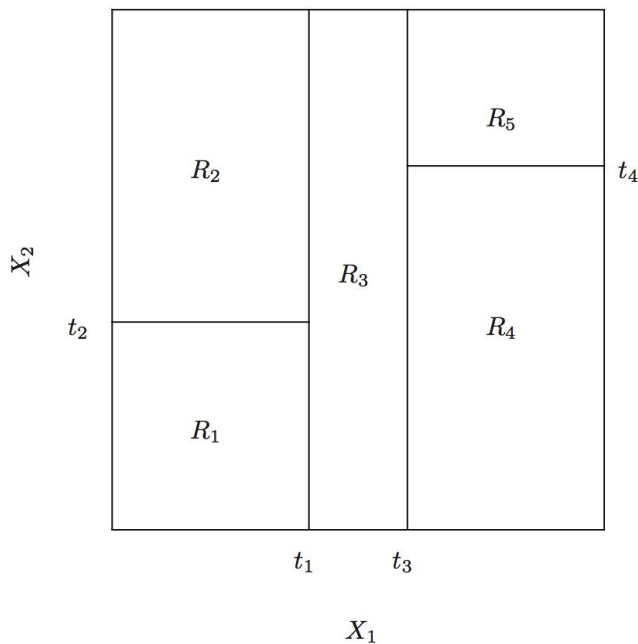
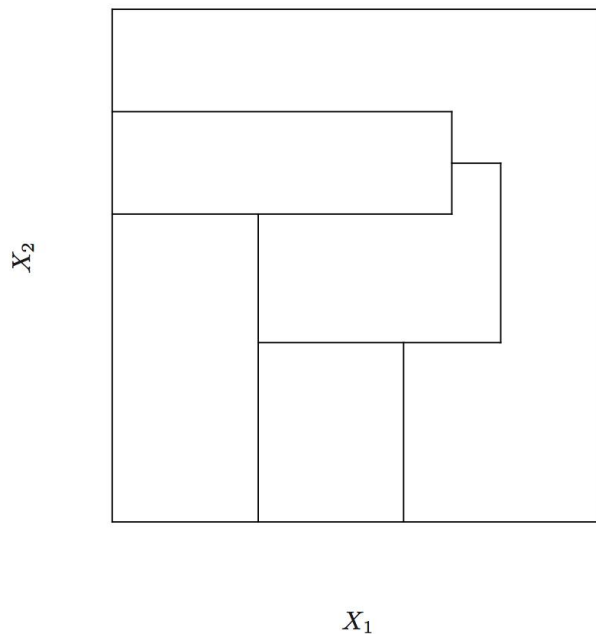
- 自顶向下：从所有样本开始，不断从当前位置，把样本切分到2个分支里
- 贪婪：每一次的划分，只考虑当前最优，而不回过头考虑之前的划分

- 选择切分的维度(特征) x_j 以及切分点 s 使得划分后的树RSS结果最小

$$R_1(j, s) = \{x | x_j < s\}$$

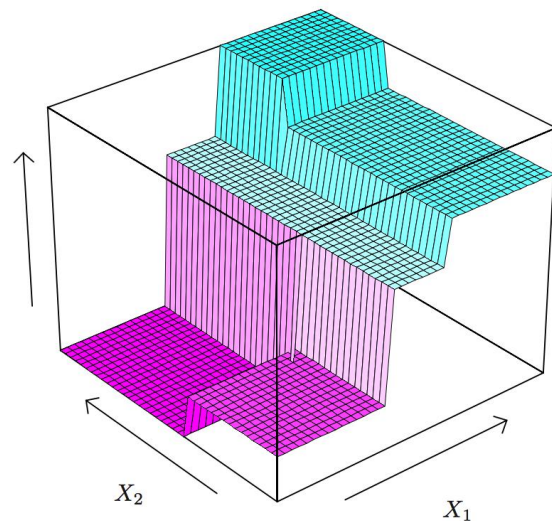
$$R_2(j, s) = \{x | x_j \geq s\}$$

$$RSS = \sum_{x_i \in R_1(j, s)} (y_i - \tilde{y}_{R_1})^2 + \sum_{x_i \in R_2(j, s)} (y_i - \tilde{y}_{R_2})^2$$



从左到右:

- 上左1: 非二分切分得到的回归树空间划分
- 上左2: 二分递归切分得到回归树空间划分
- 上左3: 对应左2的回归树
- 下: 对应左2的回归树空间划分与预估结果可视化



回归树剪枝

如果让回归树充分“生长”，同样会有过拟合的风险

- 解决办法：添加正则化项衡量
- 考虑剪枝后得到的子树 $\{T_\alpha\}$ ，其中 α 是正则化项的系数，当我固定一个 α 后，最佳的 T_α 就是使得下列式子值最小的子树。

$$\sum_{m=1}^{|T|} \sum_{x_i \in R_m} (y_i - \tilde{y}_{R_m})^2 + \alpha |T|$$

其中 $|T|$ 是回归树叶子节点的个数

- 其中的 α 可以通过交叉验证去选择

要点总结

● 回归树模型

- 把整个特征空间切分成不相交的子区域
- 每个区域预估成该区域样本的均值

● 回归树构建

- 自顶向下的贪婪式二分希望最小化RSS
- 可以通过正则化项去控制过拟合



03

bagging与随机森林

3.1

Bagging思想

3.2

随机森林

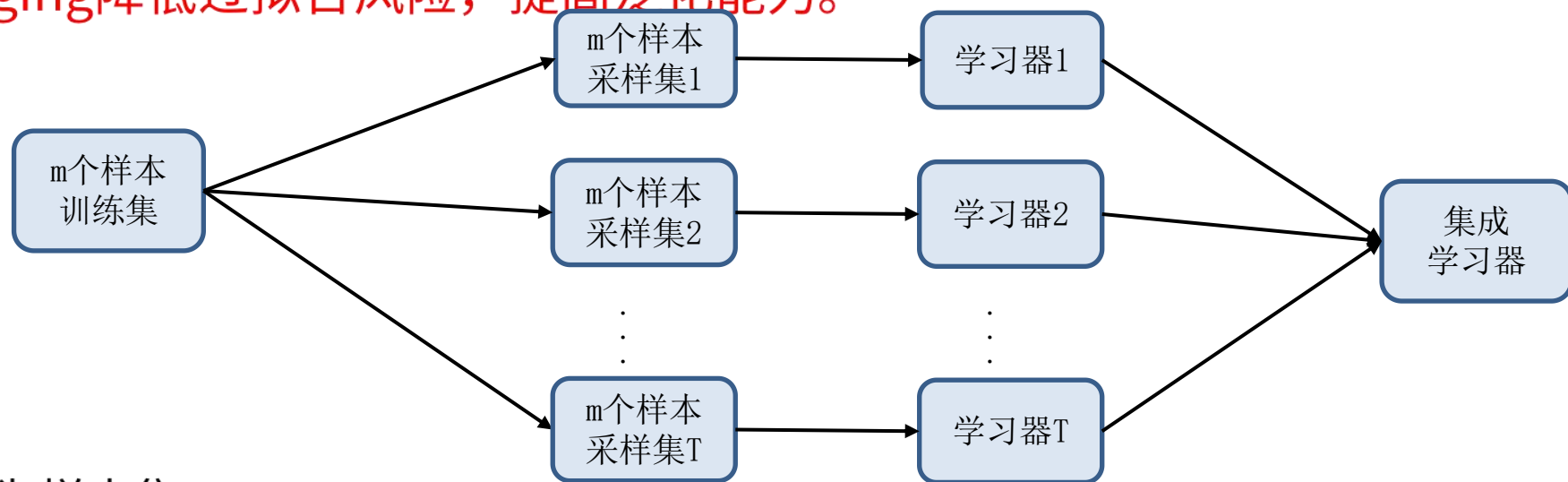
在介绍强大的RandomForest (随机森林) 之前，我们先介绍一下Bootstrapping和Bagging

Bootstrapping: 名字来自成语 “pull up by your own bootstraps”，意思是依靠你自己的资源，称为自助法，它是一种有放回的抽样方法，它是非参数统计中一种重要的估计统计量方差进而进行区间估计的统计方法。其核心思想和基本步骤如下：

- (1) 采用重抽样技术从原始样本中抽取一定数量（自己给定）的样本，此过程允许重复抽样。
- (2) 根据抽出的样本计算给定的统计量 T 。
- (3) 重复上述 N 次（一般大于1000），得到 N 个统计量 T 。
- (4) 计算上述 N 个统计量 T 的样本方差，得到统计量的方差。

Bootstrap是现代统计学较为流行的一种统计方法，在小样本时效果很好。通过方差的估计可以构造置信区间等，其运用范围得到进一步延伸。

Bagging是bootstrap aggregating的缩写，使用了上述的bootstrapping思想。
Bagging降低过拟合风险，提高泛化能力。



输入为样本集 $D = \{(x, y_1), (x_2, y_2), \dots (x_m, y_m)\}$

1) 对于 $t = 1, 2, \dots, T$:

a) 对训练集进行第 t 次随机采样，共采集 m 次，得到包含 m 个样本的采样集 D_m

b) 用采样集 D_m 训练第 t 个基学习器 $G_t(x)$

2) 分类场景，则 T 个学习器投出最多票数的类别为最终类别。回归场景， T 个学习器得到的回归结果进行算术平均得到的值为最终的模型输出。

RandomForest(随机森林)是一种基于树模型的Bagging的优化版本。核心思想依旧是bagging，但是做了一些独特的改进。

RF使用了CART决策树作为基学习器，具体过程如下：

输入为样本集 $D = \{(x, y_1), (x_2, y_2), \dots (x_m, y_m)\}$

1) 对于 $t = 1, 2, \dots, T$:

a) 对训练集进行第t次随机采样，共采集m次，得到包含m个样本的采样集 D_m

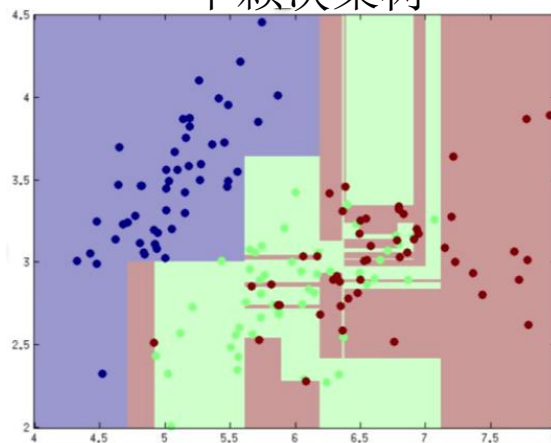
b) 用采样集 D_m 训练第m个决策树模型 $G_m(x)$ ，在训练决策树模型的节点的时候，在节点上所有的样本特征中选择一部分样本特征，在这些随机选择的部分样本特征中选择一个最优的特征来做决策树的左右子树划分

2) 分类场景，则T个基模型(决策树)投出最多票数的类别为最终类别。回归场景，T个基模型(回归树)得到的回归结果进行算术平均得到的值为最终的模型输出。

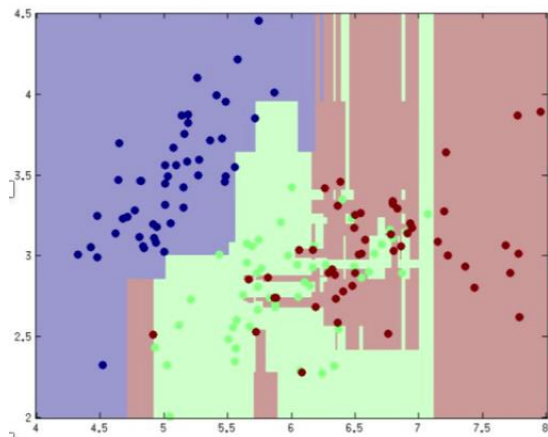
3.2 随机森林

RandomForest（随机森林） 在iris数据集上的分类表现：

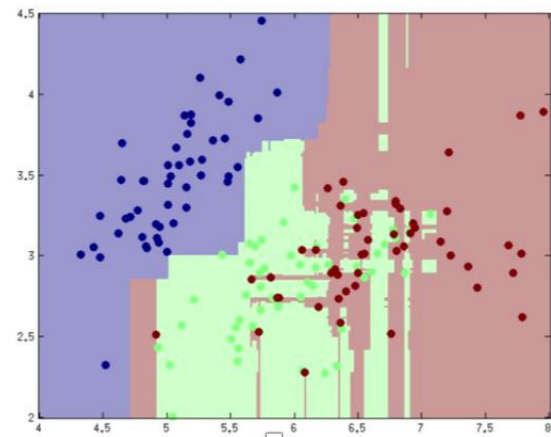
单颗决策树



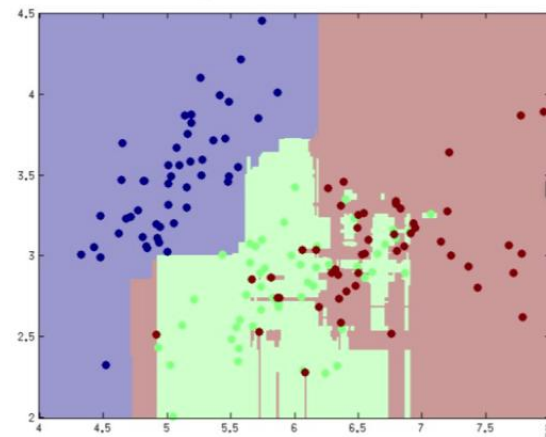
5颗决策树的随机森林



25颗决策树的随机森林



100颗决策树的随机森林





04

数据案例讲解

4.1

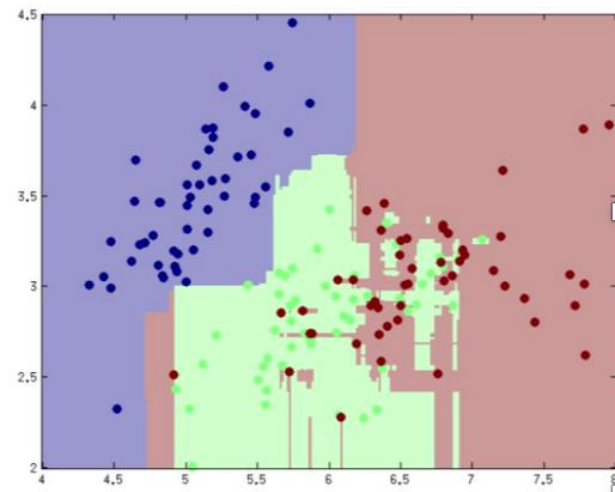
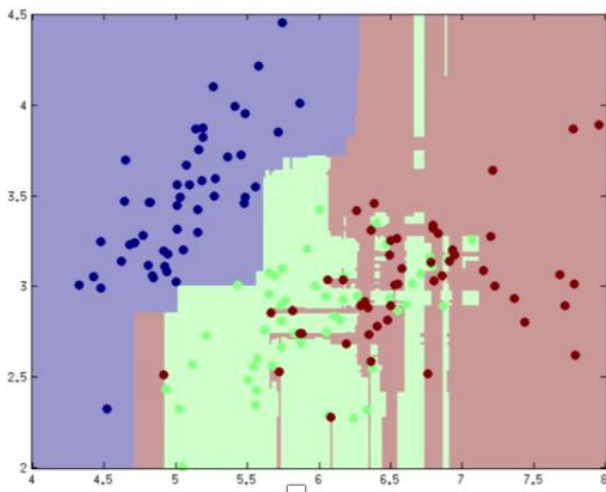
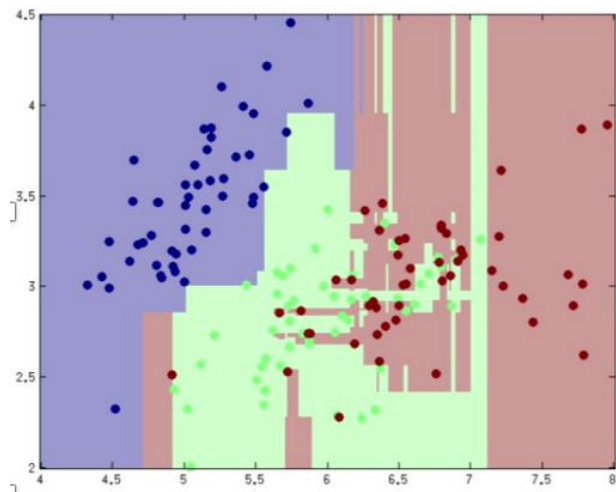
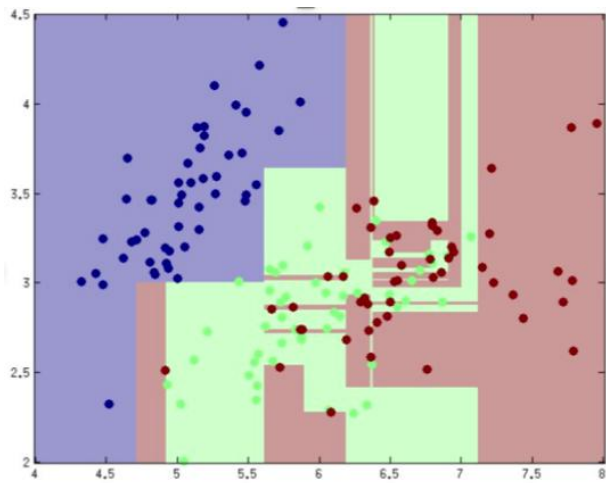
Iris数据集决策树与随机森林建模可视化

4.2

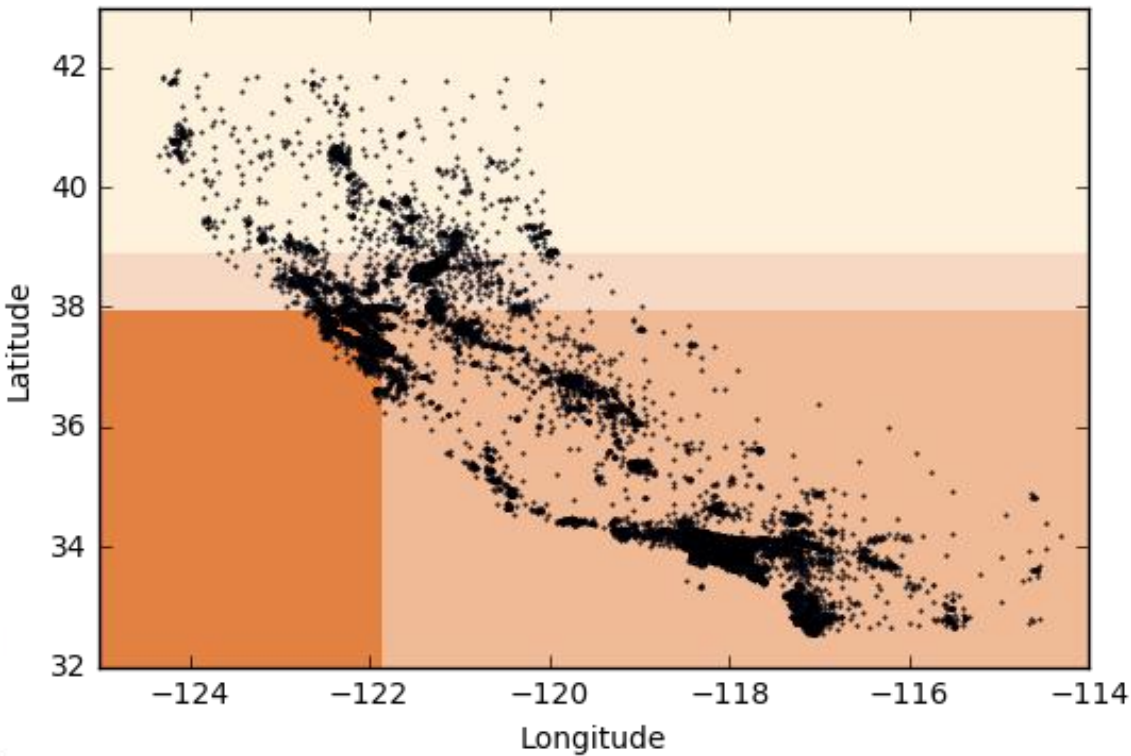
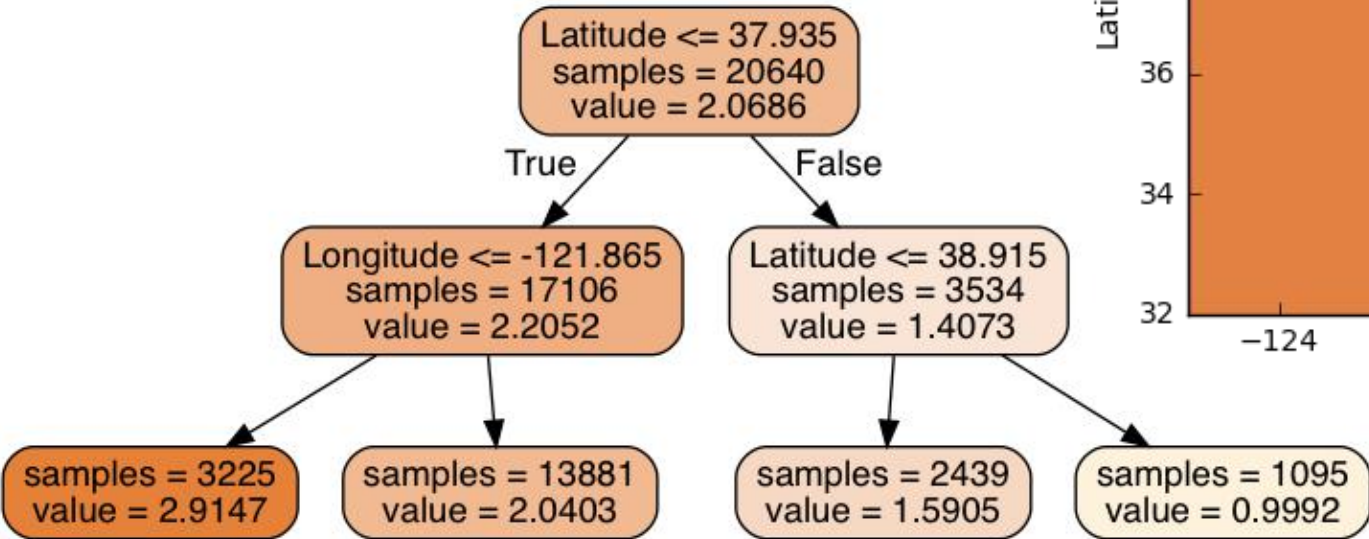
房价预测回归树与随机森林建模

3.1 iris决策树与随机森林分类与可视化

详见课程案例讲解



详见课程案例讲解



参考文献/Reference

- 周志华，机器学习，清华大学出版社，2016
- 李航，统计学习方法，清华大学出版社，2012
- Thomas M. Cover, Joy A. Thomas. Elements of Information Theory. 2006
- Christopher M. Bishop. Pattern Recognition and Machine Learning. Springer-Verlag. 2006
- Scikit-learn, <http://scikit-learn.org/stable/index.html>



THANK YOU !

Machine Learning Engineer
机器学习工程师微专业