# 机器学习导论
# 第七章

王小航

# 一个例子

- a. 生产1000个训练集子集，每个子集包含随机挑选的100个实例。
- b.使用前面得到的最佳超参数值，在每个子集上训练一个决策树。在测试集上评估这1000个决策树。因为训练集更小，所以这些决策树的表现可能比第一个决策树要差一些，只能达到约80%的准确率。
- c. 对于每个测试集实例，生成1000个决策树的预测，然后仅保留次数最频繁的预测。这样你在测试集上可获得大多数投票的预测结果。
- d.评估测试集上的这些预测，你得到的准确率应该比第一个模型更高（高出0.5%～1.5%）。

# 随机森林分类器

- 恭喜，你已经训练出了一个随机森林分类器！

# 集成学习

- 如果你随机向几千个人询问一个复杂问题，然后汇总他们的回答。在许多情况下，你会发现，这个汇总的回答比专家的回答还要好，这被称为群体智慧。
- 同样，如果你聚合一组预测器（比如分类器或回归器）的预测，得到的预测结果也比最好的单个预测器要好。
- 这样的一组预测器称为集成，所以这种技术也被称为集成学习，而一个集成学习算法则被称为集成方法。

# Bagging

- 自助采样法（bootstrap sampling）：

给定 m 个样本的数据集 D，我们对他进行采样产生数据集 D’，每次随机从D 中挑选一个样本，将其拷贝放入 D’，然后再将该样本放回初始数据集 D 中，使得该样本在下次采样中仍有可能被采样到，这个过程执行m次以后，我们就得到了包含 m个样本的数据集 D’，这就是自助采样的结果。

# Bsagging

- 给定一个训练集，基于自助采样法采样出T个含m个训练样本的采样集
- 然后基于每个采样集训练一个基学习器，再将这些基学习器进行结合。
- 每个预测器使用的算法相同，但是在不同的训练集随机子集上进行训练。

# Bagging

输入为样本集$D=\{(x,y_1),(x_2,y_2),...(x_m,y_m)\}$，弱学习器算法，弱分类器迭代次数T。

输出为最终的强分类器$f(x)$

1）对于t=1,2...,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$D_t$

b)用采样集$D_t$训练第t个弱学习器$G_t(x)$

2) 如果是分类算法预测，则T个弱学习器投出最多票数的类别或者类别之一为最终类别。如果是回归算法，T个弱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。

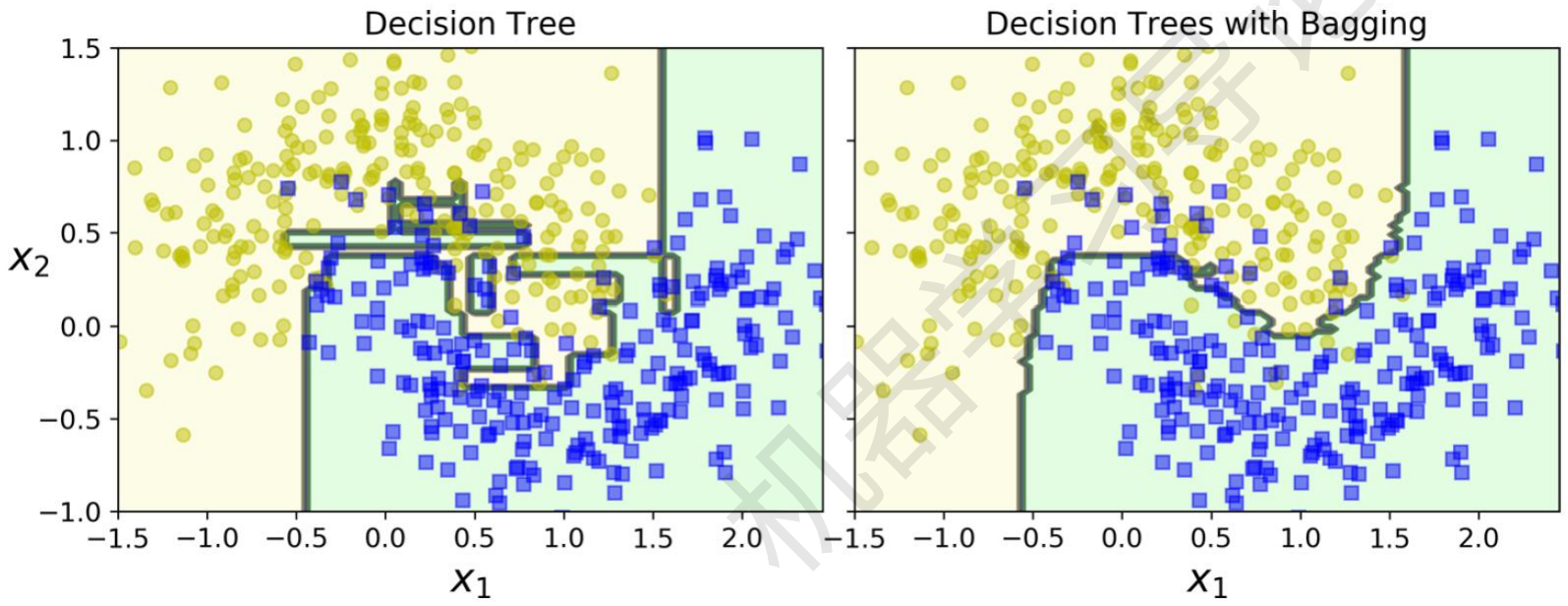
# Bagging

| | 测试例1 | 测试例2 | 测试例3 |
|---|---|---|---|
| h1 | √ | √ | × |
| h2 | × | √ | √ |
| h3 | √ | × | √ |
| 集成 | √ | √ | √ |

# Sklearn代码实现

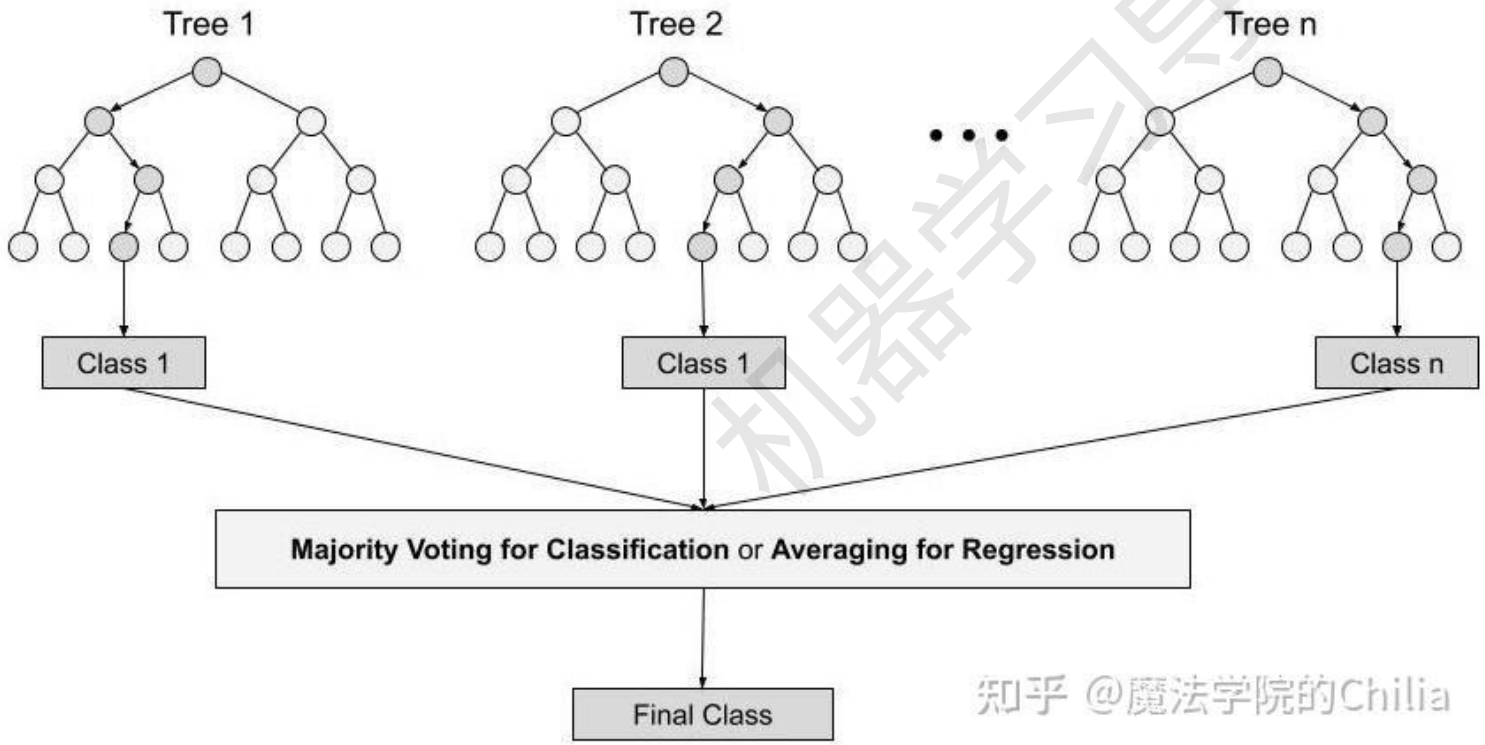
```
from sklearn.ensemble import BaggingClassifier
from sklearn.tree import DecisionTreeClassifier

bag_clf = BaggingClassifier(
    DecisionTreeClassifier(), n_estimators=500,
    max_samples=100, bootstrap=True, n_jobs=-1)
bag_clf.fit(X_train, y_train)
y_pred = bag_clf.predict(X_test)
```

# Sklearn代码实现



# 随机森林

- 随机森林就是通过集成学习的Bagging思想将多棵树集成的一种算法：它的基本单元就是决策树。



# 随机森林

- 每棵树的按照如下规则生成：
  - （1）如果训练集大小为N，对于每棵树而言，随机且有放回地从训练集中的抽取N个训练样本作为该树的训练集；从这里我们可以知道：每棵树的训练集都是不同的，而且里面包含重复的训练样本。
  - （2）如果存在M个特征，则在每个节点分裂的时候，从M中随机选择m个特征维度（m << M），使用这些m个特征维度中最佳特征(最大化信息增益)来分割节点。在森林生长期间，m的值保持不变。

# 随机森林

- 随机森林分类效果（错误率）与两个因素有关：
  - 森林中任意两棵树的相关性：相关性越大，错误率越大；（弱分类器应该good且different）
  - 森林中每棵树的分类能力：每棵树的分类能力越强，整个森林的错误率越低。（弱分类器应该good且different）

# Sklearn代码实现

```
from sklearn.ensemble import RandomForestClassifier

rnd_clf = RandomForestClassifier(n_estimators=500, max_leaf_nodes=16, n_jobs=-1)
rnd_clf.fit(X_train, y_train)

y_pred_rf = rnd_clf.predict(X_test)
```

# Boosting

- Boosting是一种串行的工作机制，即个体学习器的训练存在依赖关系，必须一步一步序列化进行。
- 其基本思想是：增加前一个基学习器在训练过程中预测错误样本的权重，使得后续基学习器更加关注这些打标错误的训练样本，尽可能纠正这些错误，一直向下串行直至产生需要的T个基学习器，Boosting最终对这T个学习器进行加权结合，产生集成学习器。
- 最具代表性的是AdaBoost 算法

# AdaBoost

- 假设给定一个二类分类的训练数据集，
$$T = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$$
- 其中，每个样本点由实例与标记组成。实例$x_i \in \mathcal{X} \subseteq \mathbb{R}^n$，$y_i \in \mathcal{Y} = \{-1, +1\}$
- AdaBoost 利用以下算法，从训练数据中学习一系列弱分类器或基本分类器，并将这些弱分类器线性组合成为一个强分类器

# AdaBoost

- 输入：训练数集$T = \{(x_1, y_1), (x_2, y_2), ..., (x_N, y_N)\}$，其中$x_i \in \mathcal{X} \subseteq \mathbb{R}^n$，$y_i \in \mathcal{Y} = \{-1, +1\}$；弱学习算法；
- 输出：最终分类器$G(x)$
- （1）初始化训练数据的权值分布

$$D_1 = (w_{11}, ..., w_{1i}, ..., w_{1N}), w_{1i} = \frac{1}{N}, i = 1,2, ..., N$$

# AdaBoost

- （2）对m = 1,2, ..., $M$
  - （a）使用具有权值分布$D_m$的训练数据集学习，得到基本分类器：

$$G_m(x): \mathcal{X} \rightarrow \{-1, +1\}$$

  - （b）计算$G_m(x)$在训练数据集上的分类误差率

$$e_m = P(G_m(x_i) \neq y_i) = \sum_{i=1}^{N} w_{mi} I(G_m(x_i) \neq y_i)$$

# AdaBoost

- （3）计算$G_m(x)$的系数

$$\alpha_m = \frac{1}{2}\log\frac{1-e_m}{e_m}$$

这里的对数是自然对数

# AdaBoost

- （d）更新训练数据集的权值分布

$$D_{\mathrm{m+1}} = \left(w_{m+1,1}, \cdots, w_{m+1,i}, \cdots, w_{m+1,N}\right)$$

$$w_{m+1,i} = \frac{w_{mi}}{Z_m}\exp\left(-\alpha_m y_i G_m(x_i)\right),\ i = 1,2,\ldots,N$$

这里$Z_m$是规范化因子

$$Z_m = \sum_{i=1}^{N} w_{mi}\exp\left(-\alpha_m y_i G_m(x_i)\right)$$

它使$D_{\mathrm{m+1}}$成为一个概率分布

# AdaBoost

- （3）构建基本分类器的线性组合

$$f(x) = \sum_{m=1}^{M} \alpha_m G_m(x)$$

得到最终分类器

$$G(x) = sign\big(f(x)\big) = sign\Big(\sum_{m=1}^{M} \alpha_m G_m(x)\Big)$$

# AdaBoost的例子

- 给定如表 8.1 所示训练数据。假设弱分类器由$x < v$或$x > v$产生，其阈值v 使该分类器在训练数据集上分类误差率最低。试用 AdaBoost 算法学习一个强分类器。

| 序号 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|---|---|---|---|---|---|---|---|---|---|---|
| $x$ | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 |
| $y$ | 1 | 1 | 1 | −1 | −1 | −1 | 1 | 1 | 1 | −1 |

# AdaBoost的例子

- 初始化权值分布

$$D_1 = (w_{11}, w_{12}, \ldots, w_{110})$$
$$w_{1i} = 0.1,\ i = 1,2, \ldots, 10$$

- 对$m = 1,$
  - （a）在权值分布为$D_1$的训练数据上，阈值$v$取2.5时分类误差率最低，故基本分类器为

$$G_1(x) = \begin{cases} 1, & x < 2.5 \\ -1, & x > 2.5 \end{cases}$$

# AdaBoost的例子

- （b）$G_1(x)$在训练数集上的误差率$e_1 = P(G_1(x_i) \neq y_i) = 0.3$
- （c）计算$G_1(x)$的系数：$\alpha_1 = \frac{1}{2}\log\frac{1-e_1}{e_1} = 0.4236$
- （d）更新训练数据的权值分布：

$$D_2 = (w_{21}, w_{22}, \ldots, w_{210})$$

$$w_{2,i} = \frac{w_{1i}}{Z_1}\exp\left(-\alpha_1 y_i G_1(x_i)\right),\ i = 1,2,\ldots,10$$

$$D_2 = (0.07143,0.07143,0.07143,0.07143,0.07143,0.07143,$$
$$0.16667,0.16667,0.16667,0.07143)$$

$$f_1(x) = 0.4236G_1(x)$$

分类器$sign(f_1(x))$在训练数据集上有3个误分点

# AdaBoost的例子

- 对$m=2$,
  - （a）在权值分布为$D_2$的训练数据上，阈值$v$取8.5时分类误差率最低，故基本分类器为

$$G_2(x)=\begin{cases}1, & x<8.5\\ -1, & x>8.5\end{cases}$$

  - （b）$G_2(x)$在训练数集上的误差率$e_2=0.2143$
  - （c）计算$\alpha_2=0.6496$

# AdaBoost的例子

- （d）更新训练数据的权值分布：

$$D_3 = (0.0455,0.0455,0.0455,0.1667,0.1667,0.1667,$$
$$0.1060,0.1060,0.1060,0.0455)$$
$$f_2(x) = 0.4236G_1(x) + 0.6496G_2(x)$$

分类器$sign(f_2(x))$在训练数据集上有3个误分点

# AdaBoost的例子

- 对$m = 3$,
  - （a）在权值分布为$D_3$的训练数据上，阈值$v$取5.5时分类误差率最低，故基本分类器为

$$G_2(x) = \begin{cases} 1, & x < 5.5 \\ -1, & x > 5.5 \end{cases}$$

  - （b）$G_3(x)$在训练数集上的误差率$e_3 = 0.1820$
  - （c）计算$\alpha_3 = 0.7514$
  - （d）更新训练数据的权值分布：

$$D_4 = (0.125,0.125,0.125,0.102,0.102,0.102,$$
$$0.065,0.065,0.065,0.125)$$

# AdaBoost的例子

- 于是得到：

$$f_3(x) = 0.4236G_1(x) + 0.6496G_2(x) + 0.7514G_3(x)$$

- 分类器$\mathrm{sign}(f_3(x))$在训练数集上误分类点数为0
- 于是最终分类器为：

$$G(x) = \mathrm{sign}\big(f_3(x)\big)$$
$$= \mathrm{sign}(0.4236G_1(x) + 0.6496G_2(x) + 0.7514G_3(x))$$

# Sklearn代码实现

```
from sklearn.ensemble import AdaBoostClassifier

ada_clf = AdaBoostClassifier(
    DecisionTreeClassifier(max_depth=1), n_estimators=200,
    algorithm="SAMME.R", learning_rate=0.5)
ada_clf.fit(X_train, y_train)
```