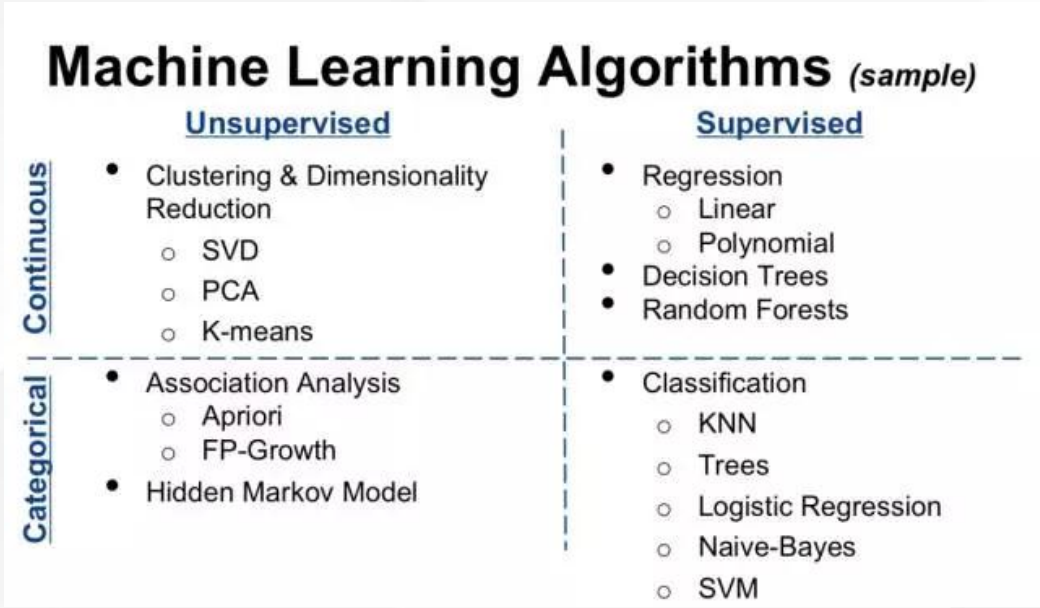
机器学习与python实践

讲师：裴得利

大纲

- 机器学习概述
  - 监督学习与无监督学习， 特征工程
- 回归模型
  - 线性回归， Logistic 回归
- 决策树类模型
  - 不同决策树模型， 兼谈 Bagging, Boosting和Stacking思想
- 评价体系
  - 评价指标及其误区

机器学习概述



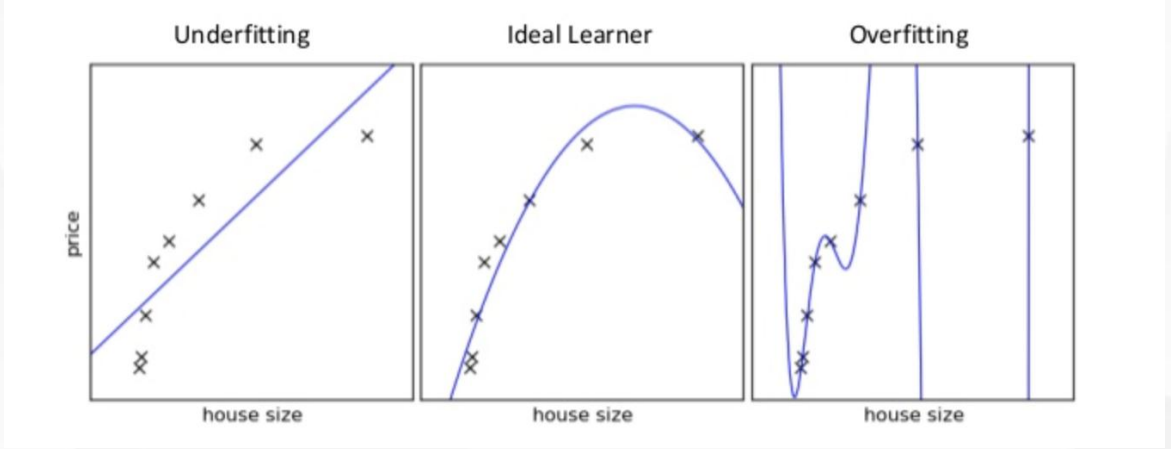
- 常见分类
  - 监督学习
    - 给定数据集并知道其正确的输出，即有反馈
    - 回归（Regression）　：　特征输入 → 连续值输出
    - 分类（Classification）：　特征输入 → 离散值输出
  - 非监督学习
    - 给定数据集，不知道其正确的输出，无反馈
    - 聚类（Clustering）：输入一批样本数据 → 划分为若干簇
    - 关联分析：　给定一批记录 → 记录中各项的关联关系

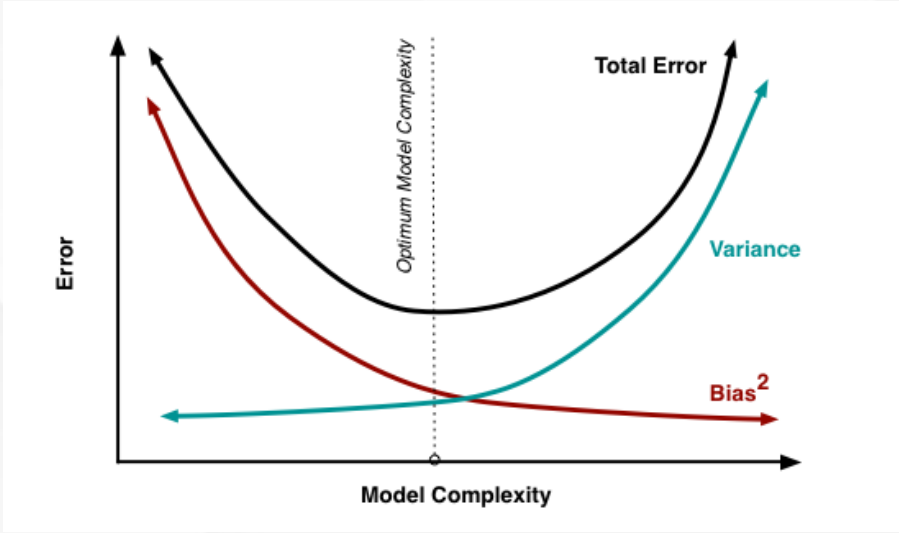
监督学习

- 监督学习
  - 要素： **特征，目标值， 模型， 数据集**
    - 目标值 = 模型（特征 | 模型参数）
  - 模型训练
    - 由训练数据集获取最优模型参数 → 模型
  - 预测
    - 利用已有模型，对未知结果做出预测
  - 老司机的例子
    - 过往的经历（数据集）， 每条经历的描述（特征）， 人生经验（模型）
    - 成长（训练过程），教你做人（预测过程）
    - 老司机带你买二手车
      - 分类：这辆车是否值得买；　　回归： 这辆车值多少钱

监督学习

- Bias – Variance tradeoff
  - 偏差 Bias
    - 预测值与真实值的平均偏差
    - Bias过大：欠拟合 underfitting
      - 没有学习到特征值与目标值之间的偏差
  - 方差 Variance
    - 同等大小数据集变动导致学习性能的波动
    - Variance 过大：过拟合overfitting
      - 对训练集噪声过于敏感，泛化能力差





特征工程

- 特征
  - 数据的预处理：将样本的属性转化为数据特征，刻画样本
  - 问题： 描述那些方面，以及怎样描述
- 特征工程
  - 时间戳处理
    - 分解成多维度如年、月、日、小时， 区分场景
    - 如交通状况（天级别，小时级别），天气预测（月级别，季度级别）
  - 类别属性处理
    - 误区： 将类别属性转换成标量， 误导模型（排序，平均）
    - 颜色属性： 用 {1,2,3} 表示{红，绿，蓝}

特征工程

- 特征工程
  - 类别属性处理
    - one-hot 编码
      - 颜色属性 {红，绿，蓝} 用 { (1，0，0), (0，1，0), (0，0，1) }表示

class sklearn.preprocessing.**OneHotEncoder**(*n_values='auto', categorical_features='all', dtype=<type 'numpy.float64'>, sparse=True, handle_unknown='error'*)

    - Hash编码
      - 近似 one-hot编码，对特征的每一种取值做hash
    - 缺点
      - 维数爆炸： 个性化特征，userid， 广告id，商品id，几百万上千万维

代码演示 one-hot 编码

特征工程

- 特征工程
  - 分箱或者分区
    - 特征离散化： 数值落入同一分区时能够呈现出共同特征
    - 增强鲁棒性，减少噪声干扰
    - 如时间分组，年龄段分组，位置分组（县乡镇 => 区市省）
  - 交叉特征
    - 两个或者更多类别属性组合成一个，比单独两个特征更有意义
    - 常与one-hot编码方式结合
    - 如地理位置服务中（经度，纬度），个性化推荐中（性别，年龄）

特征工程

- 特征工程
  - 特征选择
    - 解决“从哪些方面描述”的问题，领域知识要求强
    - 特征与目标值的相关性， 前向/后向特征搜索
  - 特征缩放
    - 回归模型中尤为突出， 不同量纲的特征值
    - 如Min-Max缩放

$$x^* = \frac{x - \min}{\max - \min}$$

class sklearn.preprocessing.**MinMaxScaler**(*feature_range=(0, 1), copy=True*)

class sklearn.preprocessing.**Normalizer**(*norm='l2', copy=True)*

$$x^* = \frac{x - \mu}{\sigma}$$

大纲

- 机器学习概述
  - 监督学习与无监督学习， 特征工程
- 回归模型
  - 线性回归， Logistic 回归
- 决策树类模型
  - 不同决策树模型， 兼谈 Bagging, Boosting和Stacking思想
- 评价体系
  - 评价指标及其误区

监督学习之回归分析

- 回归分析（Regression）
  - 回归分析是解决预测建模任务时的一种方法，用于研究自变量与因变量之间的关系
  - 典型方法
    - 线性回归 Linear Regression
    - Logistic 回归 Logistic Regression

方法	自变量（特征）	因变量（结果）	关系
线性回归	连续或离散	连续实数	线性
Logistic回归	连续或离散	(0,1)之间连续值	非线性

监督学习之回归分析

- 线性回归
  - 模型表达

$$y(x,w) = w_0 + w_1 x_1 + \cdots + w_n x_n \quad \Longrightarrow \quad h_w(x) = \sum_{i=0}^{n} w_i x_i = w^T x$$

    - 特征：　对样本的多维度描述　$x_1, x_2, \cdots, x_n$
    - 模型参数：　w为参数向量；w_i表示对应自变量（特　征）的权重，
    - 目标值　　y是因变量
  - 老司机买二手车
    - 特征：　品牌，出厂日期/价格，里程数，外观及内饰的折旧，有无事故
    - 模型参数：　每个特征的重要程度（权重）
    - 目标值：　二手车估价

监督学习之回归分析

- 线性回归
  - 特征工程的重要性
    - 领域知识：与目标值（因变量）有关的因素
    - 直接特征和**挖掘特征**
  - 训练数据集和目标函数
    - 训练集：　过往二手车销售记录（车的特征，车的价格）
    - 预测集：　给二手车一个合理的估价
    - 目标函数：预测越接近真实的越好

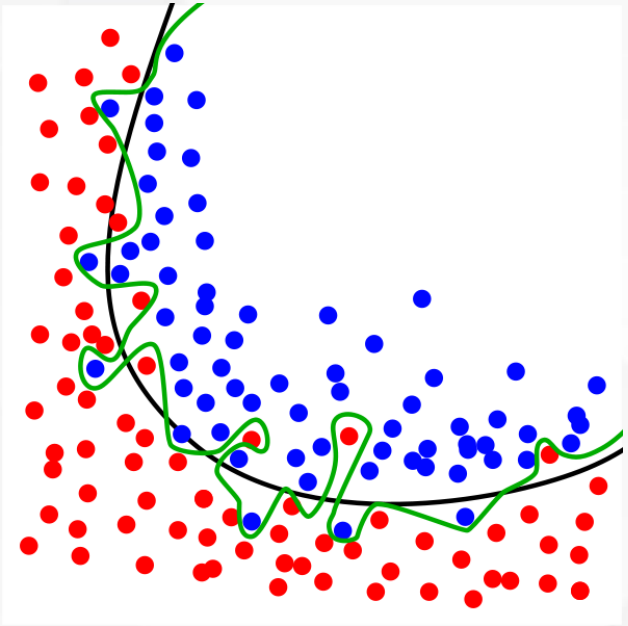
$$h_w(x) = \sum_{i=0}^{n} w_i x_i = w^T x$$

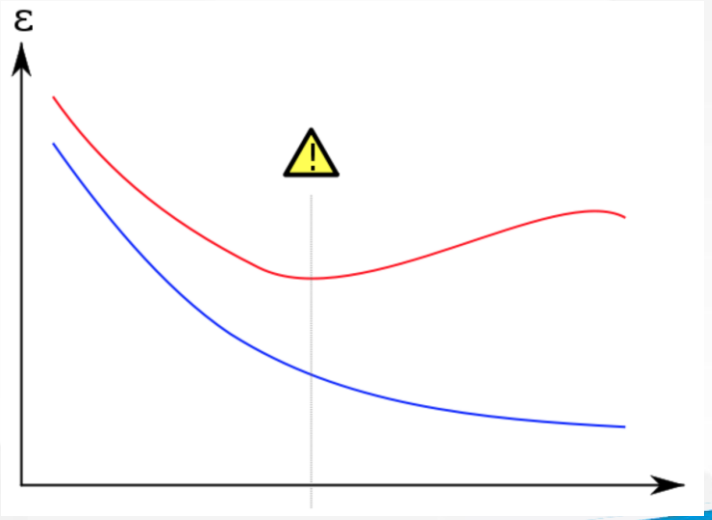
$$J(w) = \frac{1}{2} \sum_{i=1}^{m} \left(h_w(x^{(i)}) - y^{(i)} \right)^2$$

$$\min_{w} \quad J(w)$$

监督学习之回归分析

- 线性回归
  - 目标函数
    - 最小平方误差（MSE）， 最小绝对误差（MAE）
  - 优化方法
    - 最小二乘法， 梯度下降类（Newton, SGD, L-BFGS)
  - 正则化
    - 模型复杂度与推广能力
    - 过于不及：过拟合问题





监督学习之回归分析

- 线性回归
  - 正则化的意义
    - Bias-variance tradeoff : 通过模型参数的稀疏度控制模型复杂程度
  - L2正则化
    - Ridge Regression

$$\min_{w} \quad \sum_{i=1}^{m} \left(w^T x^{(i)} - y^{(i)} \right)^2 + \lambda \|w\|_2^2$$

  - L1正则化
    - Lasso Regression

$$\min_{w} \quad \sum_{i=1}^{m} \left(w^T x^{(i)} - y^{(i)} \right)^2 + \lambda \|w\|_1$$

监督学习之回归分析

- 线性回归示例
  - Python-Sklearn实现
    - 类定义

class sklearn.linear_model.**LinearRegression**(*fit_intercept=True*, *normalize=False*, *copy_X=True*, *n_jobs=1*)

    - 参数
      - Fit_intercept: 是否计算截距 w0
      - Normalize: 是否归一化
      - Copy_x: 对原数据操作，还是复制后操作
    - 方法
      - Fit（x, y), predict(x)

$$y(x, w) = w_0 + w_1 x_1 + \cdots + w_n x_n$$

代码示例 intercept

监督学习之回归分析

- 线性回归示例
  - Python-Sklearn实现
    - Ridge Regression类定义：带L2正则约束

class sklearn.linear_model.**Ridge**(*alpha=1.0*, *fit_intercept=True*, *normalize=False*, *copy_X=True*, *max_iter=None*, *tol=0.001*, *solver='auto'*, *random_state=None*)

    - 参数
      - Alpha: 正则化权重
      - Solver: 求解方法 auto, svd， cholesky， lsqr
    - Lasso Regresion 类定义： 带L1正则约束

class sklearn.linear_model.**Lasso**(*alpha=1.0*, *fit_intercept=True*, *normalize=False*, *precompute=False*, *copy_X=True*, *max_iter=1000*, *tol=0.0001*, *warm_start=False*, *positive=False*, *random_state=None*, *selection='cyclic'*)

代码示例 L1-L2正则

监督学习之回归分析

- Logitstic Regression
  - 与线性回归的区别
  - 分类模型
  - 工业界中应用最多的模型

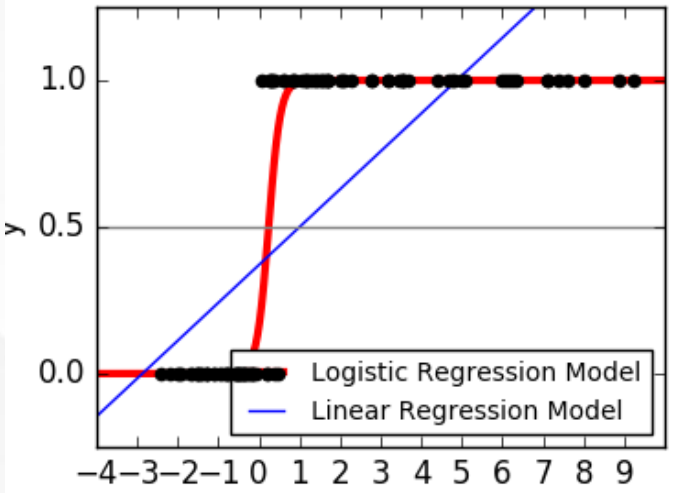
方法	自变量（特征）	因变量（结果）	关系
线性回归	连续或离散	连续实数	线性
Logistic回归	连续或离散	(0,1)之间连续值	非线性

$$h_w(x) = \sum_{i=0}^{n} w_i x_i = w^T x$$

线性回归

$$h_w(x) = \frac{1}{1 + e^{-w^T \cdot x}}$$

Logistics回归



Logistic 函数

监督学习之回归分析

- Logistic Regression
  - 在工业界的应用
    - 预估场景：推荐系统，广告系统中的点击率预估，转化率预估
    - 分类场景：用户画像标签预测， 反作弊，反垃圾
    - 难点： 海量数据， 稀疏特征， 实时性
  - 优点
    - 易用高效： LR模型建模简单清晰，能够满足大规模数据处理和实时系统的要求
    - 概率结果： 输出结果可用概率解释，天然适用于预估问题
    - 强解释性：特征与标签之间建立关联，参数取值直接反应特征强弱
    - 资源丰富：有大量的机器学习开源工具包含LR模型， 如sk-learn, spark-mllib：

监督学习之回归分析

- Logistic Regrssion 示例
  - Python-sklearn 实现

class sklearn.linear_model.**LogisticRegression**(*penalty='l2', dual=False, tol=0.0001, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='liblinear', max_iter=100, multi_class='ovr', verbose=0, warm_start=False, n_jobs=1*）

  - 参数类型
    - Penalty： 正则化约束， L1 或者L2正则
    - C： 正则化的权重
    - Solver： 优化求解方法，newton-cg, lbfgs, sag, liblinear
    - Multi_class: 多分类的处理方法，ovr和multinominal
    - Class_weight: 类别权重，数据集不均衡时尤其有用

代码示例 logistic vs linear

监督学习之回归分析

- Logistic Regrssion 示例
  - Python-sklearn 实现
    - 使用建议

场景	Solver
小数据集或者L1 正则	liblinear
多项式损失 或者 较大数据集	Lbfgs, sag, newton-cg
超大数据集	sag

    - 正则化的影响

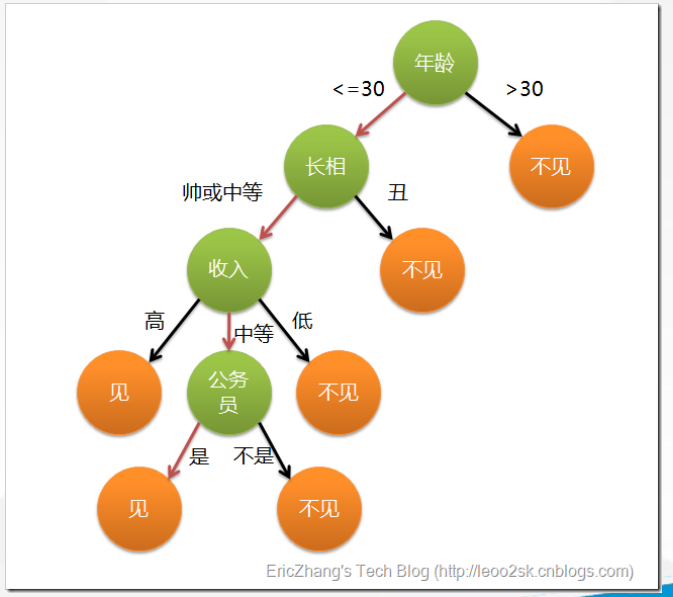
代码示例 L1-L2正则

大纲

- 机器学习概述
  - 监督学习与无监督学习， 特征工程
- 回归模型
  - 线性回归， Logistic 回归
- 决策树类模型
  - 不同决策树模型， 兼谈 Bagging, Boosting和Stacking思想
- 评价体系
  - 评价指标及其误区

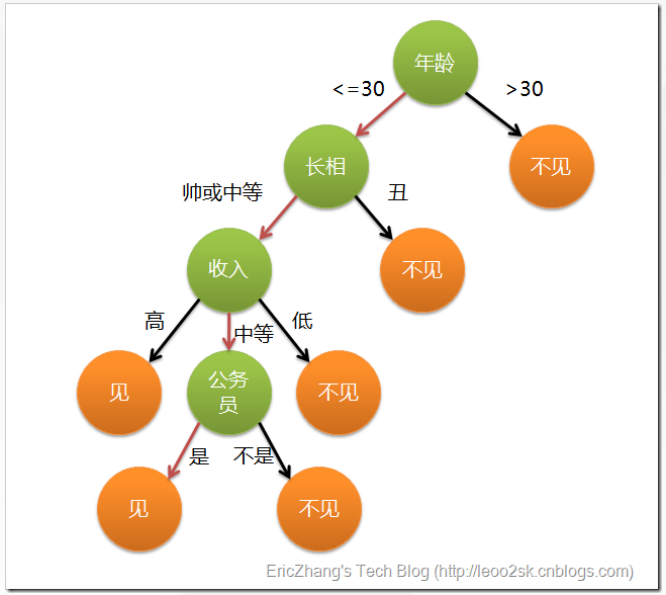
监督模型之决策树类模型

- 决策树类模型
  - 决策树（decision tree）是一个树结构（可以是二叉树或非二叉树）。其每个非叶节点表示一个特征属性上的测试，每个分支代表这个特征属性在某个值域上的输出，而每个叶节点存放一个类别
  - 兼谈bagging、boosting和stacking思想
    - Bagging： 随机森林（Random Forest)
    - Boosting： 梯度提升决策树（GBDT）
    - 不仅限于决策树类模型！



监督学习之决策树类模型

- 决策树类模型、
  - 优点：
    - 可解释性强，模型可视化
    - 无需太多数据预处理，如归一化等
    - 能够直接处理离散值和连续值
    - 对噪声值不敏感
  - 缺点
    - 容易过拟合，导致泛化能力下降
    - 不稳定，数据扰动可产出完全不同的树
    - 最优决策树是NP-完全问题， 一般用启发式



监督学习之决策树类模型

- 决策树
  - 构造： 分裂属性
    - 在某个节点处按照某一特征属性的划分成不同的分支
    - 目标：让分裂后子集尽可能的“纯”（属于同一类别）
      - 属性是离散值且不要求生成二叉决策树
      - 属性是离散值且要求生成二叉决策树
      - 属性是连续值
  - 回归树与分类树
    - 取决于应用场景和Label的取值类型
    - 回归树：Label是连续值
    - 分类树：Label是离散值

编号	得分	等级1	等级2
1	82	良好	过关
2	74	中等	不过关
3	68	中等	不过关
4	91	优秀	过关
5	88	良好	过关
6	53	较差	不过关
7	76	良好	过关
8	62	中等	不过关
9	58	较差	不过关
10	97	优秀	过关

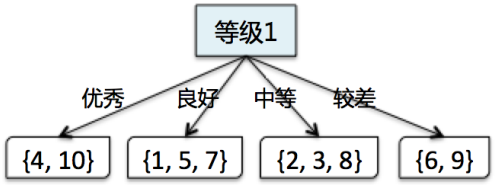


图4.1 单层决策树

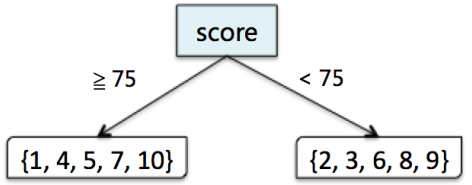


图4.2 连续值-离散化

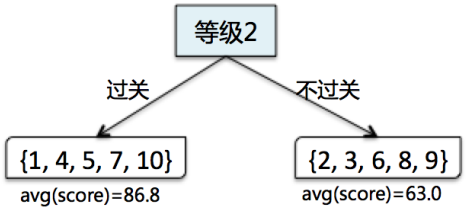


图4.3 CART - 回归树

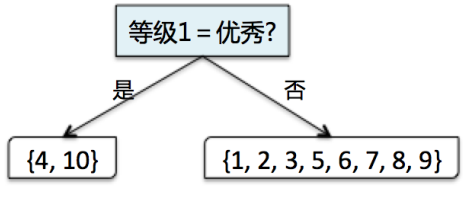


图4.4 CART - 分类树

监督学习之决策树类模型

- 决策树
  - ID3：熵增益

$$IG(T) = H(C) - H(C|T)$$
$$= -\sum_{i=1}^{k} p(c_i) \cdot \log_2 p(c_i) + \sum_{i=1}^{n} \sum_{j=1}^{k} p(c_j, t_i) \cdot \log_2 p(c_j|t_i)$$

    - 天然倾向于分支比较多的属性（无用）：如利用编号进行分类
  - C4.5：增益率

$$IG_{ratio}(T) = \frac{IG(T)}{SplitInfo(T)}$$

    - 考虑属性本身的熵，作为归一化分母
    - 优化对连续数值分列： 排序 -> 属性变化时再切开

决策树算法	特征选择方法
ID3	信息增益
C4.5	增益率
CART	回归树： 最小二乘 分类树： 基尼指数

监督学习之决策树类模型

- 决策树
  - 分类与回归树（CART）
    - 二叉决策树，学习过程等价于递归的二分每个特征，将输入空间（特征空间）划分为有限个子空间，并且在子空间上确定预测的概率分布
    - ID3/C4.5：叶子节点对应数据子集通过“多数表决”的方式确定一个类别
    - CART　：叶节点对应类别的概率分布
  - 学习准则
    - 二叉分类树：基尼指数 Gini Index

$$Gini(D) = 1 - \sum_{i=1}^{k} p_i^2$$

    - 二叉回归树：平方误差最小化

$$\hat{v}_j := \arg\min \sum_{x^{(i)} \in R_j} \left(y^{(i)} - f(x^{(i)})\right)^2$$

监督学习之决策树类模型

- 决策树示例
  - Python-sklearn实现
    - 分类树-实现

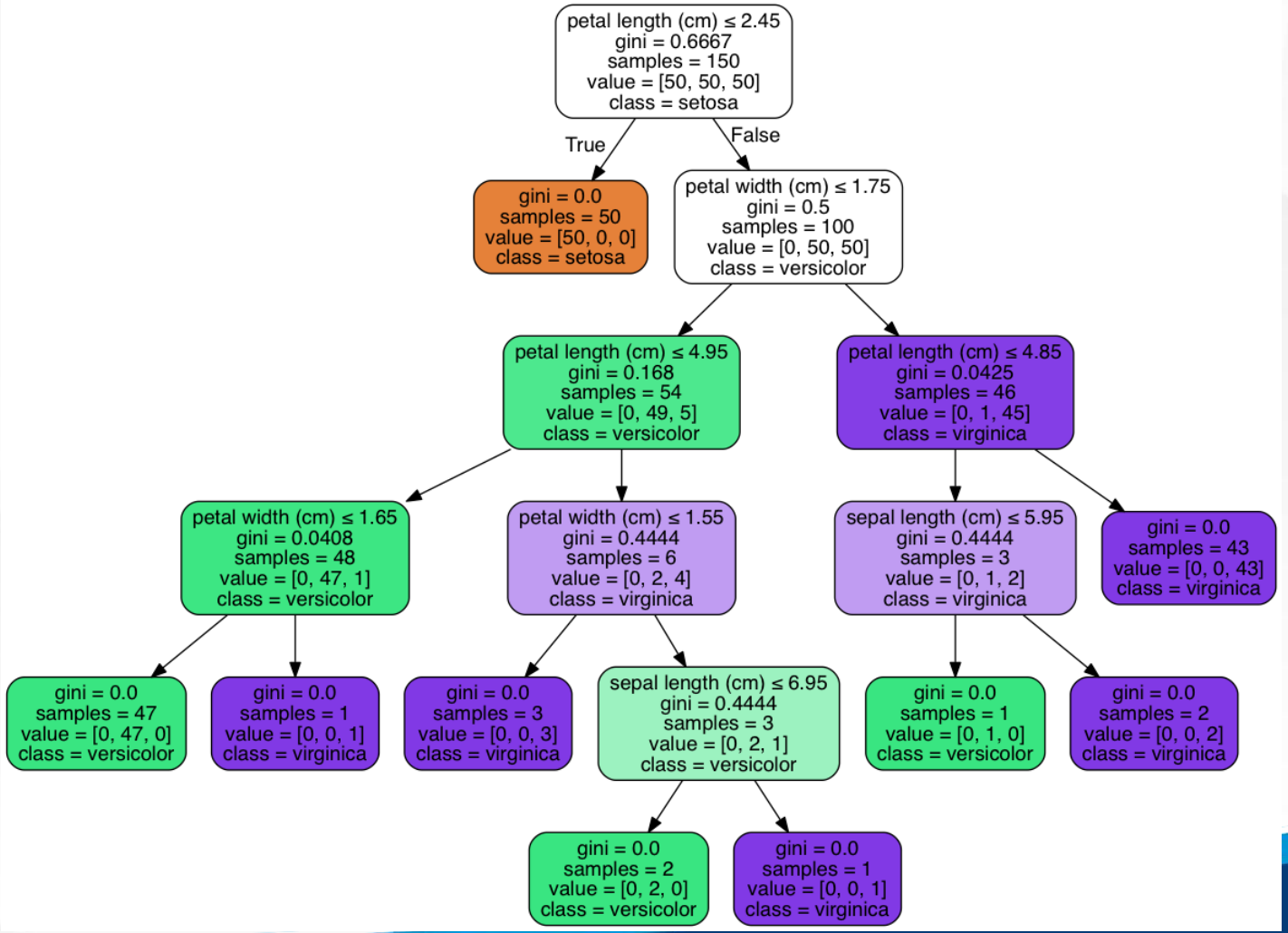
class sklearn.tree.**DecisionTreeClassifier**(*criterion='gini', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_split=1e-07, class_weight=None, presort=False*

    - 主要参数
      - Criterion：分裂选择准则， gini和entropy
      - Splitter： 每次分裂方法， best和random

监督学习之决策树类模型

- 决策树示例
  - 可视化工具GraphViz
    - 示例： iris 数据集分类模型

类别： 鸢尾属的三个亚属
Setosa, Versicolour, and Virginica）
特征：花萼(sepal）长度/宽度
花瓣(petal)长度/宽度
数据集： 共150个样本， 150*4的数据



监督学习之决策树类模型

- 决策树示例
  - Python-sklearn实现
    - 回归树-类实现

class sklearn.tree.**DecisionTreeRegressor**(*criterion='mse', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_split=1e-07, presort=False*)

    - 参数
      - Criterion: 目标函数，MSE/MAE
      - Max_depth: 最大树深度，控制过拟合
    - 目标函数
      - MSE： mean squared error

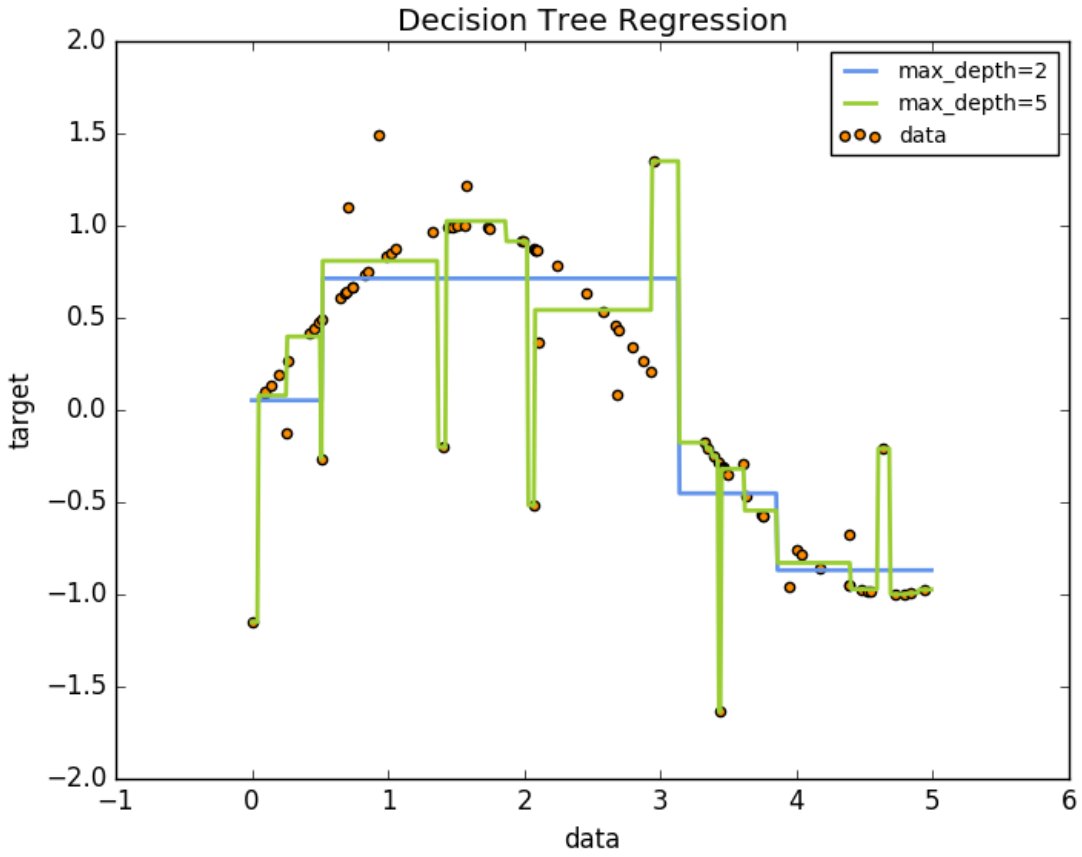
$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2$$

      - MAE： mean Absolute Error

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|f_i - y_i| = \frac{1}{n}\sum_{i=1}^{n}|e_i|\,.$$

监督学习之决策树类模型

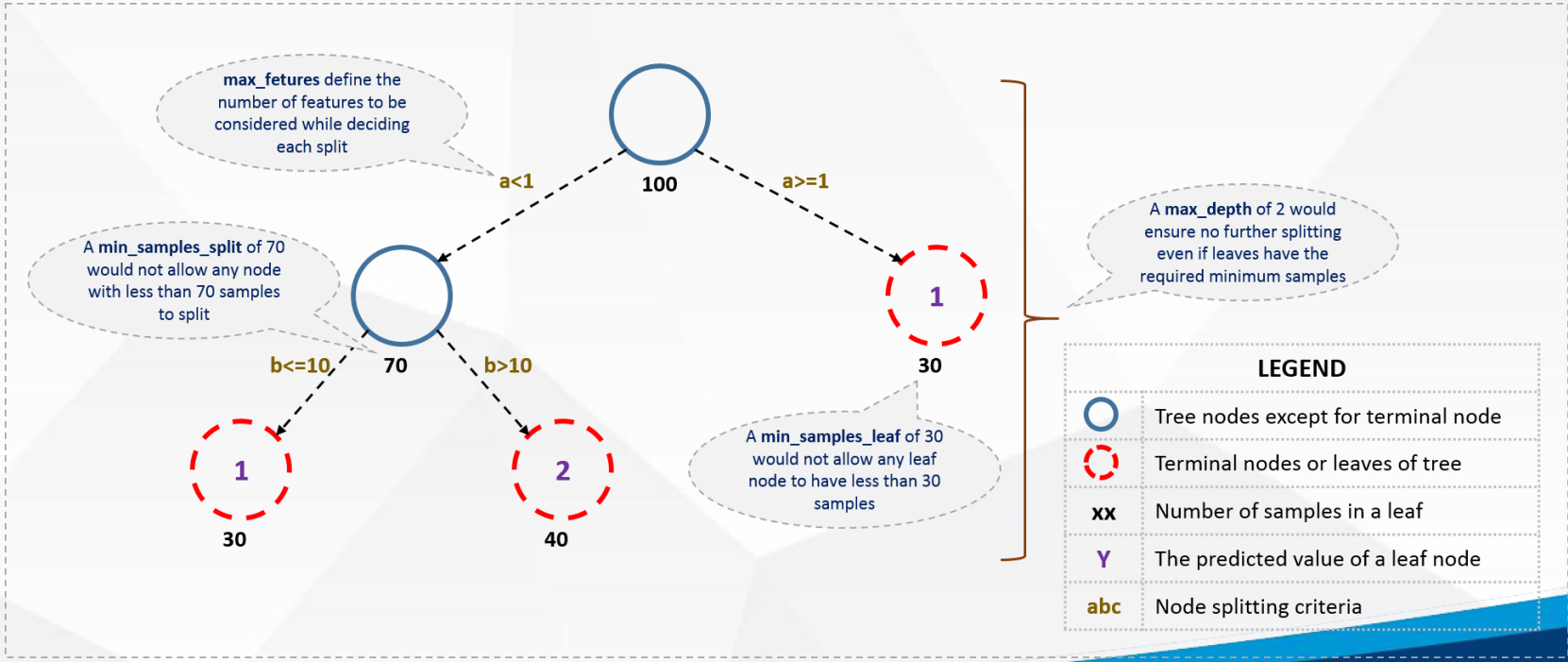
- 决策树示例
  - 决策回归树
    - 拟合带噪声sine曲线
    - 树的深度与过拟合问题



监督学习之决策树类模型

- 决策树示例
  - 控制参数
    - Max_depth
    - Min_samples_split
    - Min_samples_leaf
    - Max_features

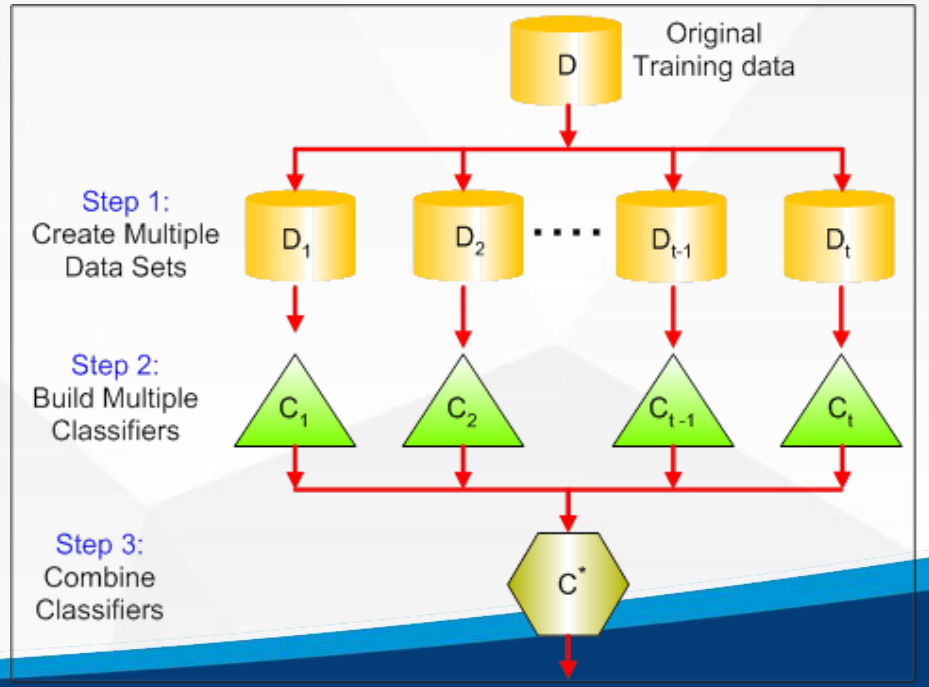
class sklearn.tree.**DecisionTreeRegressor**(*criterion='mse', splitter='best', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features=None, random_state=None, max_leaf_nodes=None, min_impurity_split=1e-07, presort=False*)



监督学习之决策树类模型

- Bagging思想
  - 什么是Bagging
    - 融合在多个不同子数据集上训练的分类器的预测结果
  - Bagging的 优势
    - 鲁棒性，泛化能力
  - 典型算法：随机森林 Random Forest
    - 随机性的体现
      - 特征随机划分和数据集随机划分

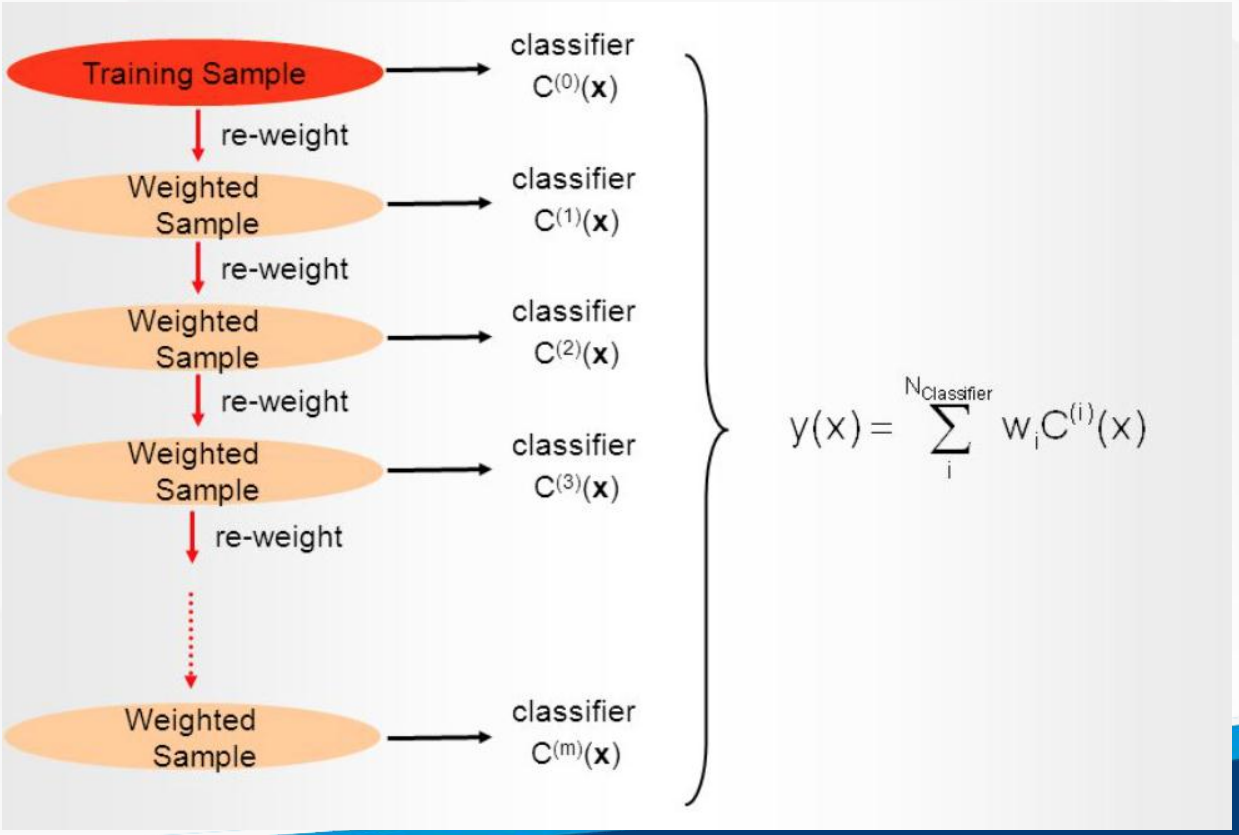
sklearn.ensemble.**RandomForestClassifier**



监督学习之决策树类模型

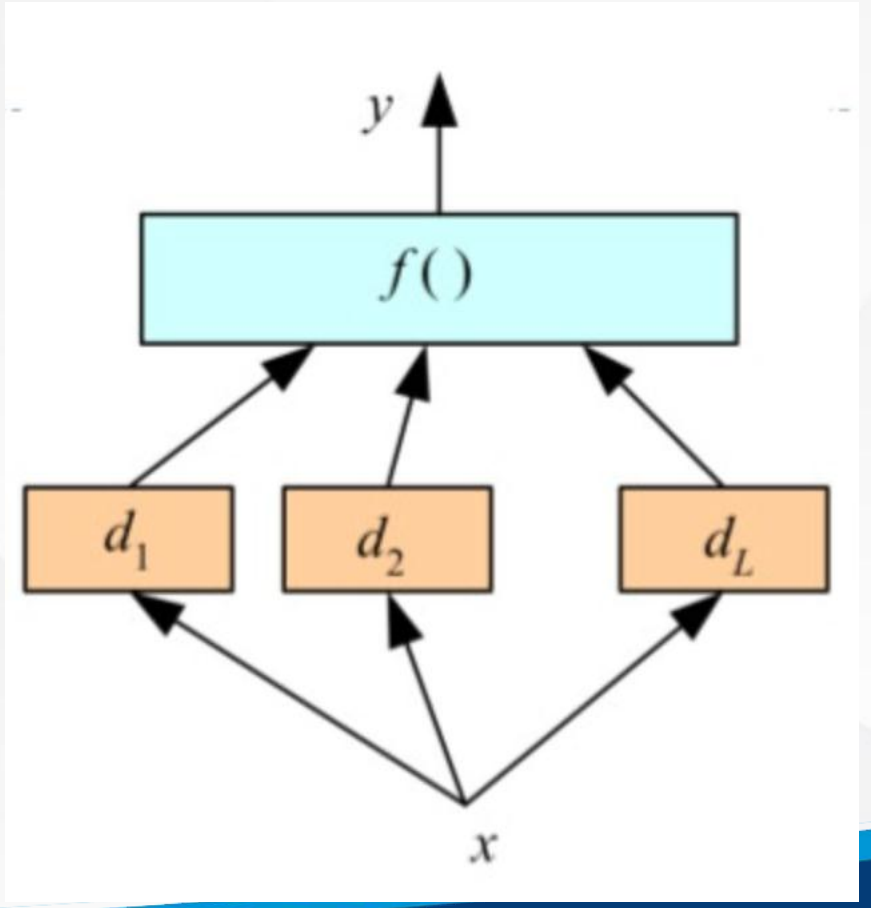
- Boosting思想
  - Boosting的优势
    - 通过多个子数据集上的分类器融合
    - 子数据集的划分不是随机
    - 在前一轮基础上迭代优化
  - 典型算法：GBDT/MART
    - GBDT 梯度提升决策树
    - 回归树（也可以用于分类）

sklearn.ensemble.**GradientBoostingClassifier**



监督学习之决策树类模型

- Stacking思想
  - 什么是Stacking
    - 分层结构，在弱分类器基础上增加一层
    - 不限于bagging的Voting思路，也不限于线性
  - Stacking优势
    - 与bagging中的线性融合相比表达能力更强



监督学习

- Bagging, Boosting 和 Stacking
  - 不仅适用于决策树类模型
  - Stacking看起来是两者优点的融合，实际操作存在困难

	Bagging	Boosting	Stacking
划分子数据集	随机划分	错分样本更高 采样概率	不确定
目标	减少模型方差	增强预测能力	兼有
子模型的融合方法	（带权）平均	带权重的投票	Logistic Regression

大纲

- 机器学习概述
  - 监督学习与无监督学习， 特征工程
- 回归模型
  - 线性回归， Logistic 回归
- 决策树类模型
  - 不同决策树模型， 兼谈 Bagging, Boosting和Stacking思想
- 评价体系
  - 评价指标及其误区

评价体系

- 评价数据集
  - 验证集 与 测试集
- 分类场景
  - 除了准确率，召回率之外…
  - F1 score， ROC/AUC
- 回归场景
  - MSE， MAE
- 排序场景
  - MAP, DCG, NDCG

		predicted condition			
	total population	prediction positive	prediction negative	Prevalence $= \frac{\Sigma \text{ condition positive}}{\Sigma \text{ total population}}$	
true condition	condition positive	**True Positive (TP)**	**False Negative (FP)** (type II error)	True Positive Rate (TPR), Sensitivity, Recall, Probability of Detection $= \frac{\Sigma \text{ TP}}{\Sigma \text{ condition positive}}$	False Negative Rate (FNR), Miss Rate $= \frac{\Sigma \text{ FN}}{\Sigma \text{ condition positive}}$
	condition negative	**False Positive (FP)** (Type I error)	**True Negative (TN)**	False Positive Rate (FPR), Fall-out, Probability of False Alarm $= \frac{\Sigma \text{ FP}}{\Sigma \text{ condition negative}}$	True Negative Rate (TNR), Specificity (SPC) $= \frac{\Sigma \text{ TN}}{\Sigma \text{ condition negative}}$
	Accuracy $= \frac{\Sigma \text{ TP} + \Sigma \text{ TN}}{\Sigma \text{ total population}}$	Positive Predictive Value (PPV), Precision $= \frac{\Sigma \text{ TP}}{\Sigma \text{ prediction positive}}$	False Omission Rate (FOR) $= \frac{\Sigma \text{ FN}}{\Sigma \text{ prediction negative}}$	Positive Likelihood Ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$	Diagnostic Odds Ratio (DOR) $= \frac{\text{LR+}}{\text{LR−}}$
		False Discovery Rate (FDR) $= \frac{\Sigma \text{ FP}}{\Sigma \text{ prediction positive}}$	Negative Predictive Value (NPV) $= \frac{\Sigma \text{ TN}}{\Sigma \text{ prediction negative}}$	Negative Likelihood Ratio (LR−) $= \frac{\text{FNR}}{\text{TNR}}$	

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}}.$$

评价体系

- 尽信书不如无书
  - 很多时候单一指标会骗人
  - 数据不均衡造成指标失真
  - 不同业务场景对指标的偏重
    - 预警类：重视召回率
    - 判责类：重视准确率

评价体系

		predicted condition			
	total population	prediction positive	prediction negative	Prevalence $= \frac{\Sigma \text{ condition positive}}{\Sigma \text{ total population}}$	
true condition	condition positive	**True Positive (TP)**	**False Negative (FP)** (type II error)	True Positive Rate (TPR), Sensitivity, Recall, Probability of Detection $= \frac{\Sigma \text{ TP}}{\Sigma \text{ condition positive}}$	False Negative Rate (FNR), Miss Rate $= \frac{\Sigma \text{ FN}}{\Sigma \text{ condition positive}}$
	condition negative	**False Positive (FP)** (Type I error)	**True Negative (TN)**	False Positive Rate (FPR), Fall-out, Probability of False Alarm $= \frac{\Sigma \text{ FP}}{\Sigma \text{ condition negative}}$	True Negative Rate (TNR), Specificity (SPC) $= \frac{\Sigma \text{ TN}}{\Sigma \text{ condition negative}}$
	Accuracy $= \frac{\Sigma \text{ TP} + \Sigma \text{ TN}}{\Sigma \text{ total population}}$	$= \frac{\Sigma \text{ TP}}{\Sigma \text{ prediction positive}}$	se Omission Rate (FOR) $= \frac{\Sigma \text{ FN}}{\Sigma \text{ prediction negative}}$	Positive Likelihood Ratio (LR+) $= \frac{\text{TPR}}{\text{FPR}}$	Diagnostic Odds Ratio (DOR) $= \frac{\text{LR+}}{\text{LR−}}$
		False Discovery Rate (FDR) $= \frac{\Sigma \text{ FP}}{\Sigma \text{ prediction positive}}$	Negative Predictive Value (NPV) $= \frac{\Sigma \text{ TN}}{\Sigma \text{ prediction negative}}$	Negative Likelihood Ratio (LR−) $= \frac{\text{FNR}}{\text{TNR}}$	

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precision} \cdot \text{recall}}{(\beta^2 \cdot \text{precision}) + \text{recall}}.$$

谢谢！

Q&A