# 学习目标

|  |
| --- |
|  |

# 1.什么是集成学习方法

|  |
| --- |
|  |

# 2. .什么是随机森林

|  |
| --- |
|  |
|  |

## 随机森林是由决策树构成的,

# 3.随机森林原理过程

|  |
| --- |
| Bootstrap抽样  在统计学中，自助法（Bootstrap Method，Bootstrapping或自助抽样法）是一种从给定训练集中有放回的均匀抽样，也就是说，每当选中一个样本，它等可能地被再次选中并被再次添加到训练集中 |
|  |

## 注意理解,这里有2个随机,保证每一颗树的独立性

|  |
| --- |
|  |

# 4.API算法模块

|  |
| --- |
|  |

# 5.随机森林预测案例,把泰坦尼克号案例复制一份,然后把分类器改为随机森林分类器

|  |
| --- |
|  |
| *""" 决策树案例:泰坦尼克号乘客生存预测数据来做titanic.txt """* import pandas as pd from sklearn.feature\_extraction import DictVectorizer from sklearn.tree import DecisionTreeClassifier, export\_graphviz from sklearn.model\_selection import train\_test\_split, GridSearchCV # from sklearn.tree import DecisionTreeClassifier from sklearn.ensemble import RandomForestClassifier  # 设置数据格式 pd.set\_option("display.max\_column", None) pd.set\_option("display.width", 200) # 1.加载数据 titanic = pd.read\_csv("./titanic.txt") # 1.2 确定特征值和目标值 features = titanic[["pclass", "age", "sex"]] target = titanic["survived"] # print(features,target) # 2.数据处理: # 1>age字段的缺失值处理 features["age"].fillna(features["age"].mean(), inplace=True) # 缺失值用平均值来填补,数据就地处理 # print(features["age"]) # 2>把特征值转换为字典的形式 features = features.to\_dict(orient="records") # 先把特征值转换为字典列表格式 # 3.划分数据集 x\_train, x\_test, y\_train, y\_test = train\_test\_split(features, target) # 4.字典特征抽取 transfer = DictVectorizer() x\_train = transfer.fit\_transform(x\_train) x\_test = transfer.transform(x\_test) # print(transfer.get\_feature\_names\_out()) # 5.决策树的预估器流程 # estimator = RandomForestClassifier(n\_estimators=100,criterion="entropy",max\_depth=10) estimator = RandomForestClassifier() # 利用网格搜索来进行模型调优 estimator = GridSearchCV(estimator, {"n\_estimators": [10, 50, 70]  , "max\_depth": [5, 8]},cv=4) estimator.fit(x\_train, y\_train) # 6.模型评估 # 方法1,直接比对真实值与预测值 y\_predict = estimator.predict(x\_test) print("比对真实值与预测值的结果", y\_test == y\_predict) # 方法2.计算准确率 accuracy = estimator.score(x\_test, y\_test) print("准确率为:", accuracy) print("最佳参数:",estimator.best\_params\_) print("最佳结果:",estimator.best\_score\_) print("最佳预估器:",estimator.best\_estimator\_) print("交叉验证结果:",estimator.cv\_results\_) |

## 结果

|  |
| --- |
|  |

# 6.总结

|  |
| --- |
|  |