机器学习

概述

特征工程

- 特征预处理
- 特征抽取

监督学习（Supervised learning）

纵向上，数据包含两大部分：**特征值（feature）**、**目标值（label、target）**
横向上，人为的将数据分为两大部分：**训练集（train）**、**测试集（test）**。

数据类别	变量名称
训练集 特征值	x_train
训练集 目标值	y_train
测试集 特征值	x_test
测试集 目标值	y_test

- 回归（Regression）
- 分类（Classification）

无监督学习（Unsupervised learning）

与监督学习不同，无监督学习的数据在纵向上只有**特征值（feature）**。

- 聚类（Clustering）

混淆矩阵

	Positive	Negatie
True	真正例（TP）	伪反例（FN）

	Positive	Negatie
False	伪正例（FP）	真反例（TN）

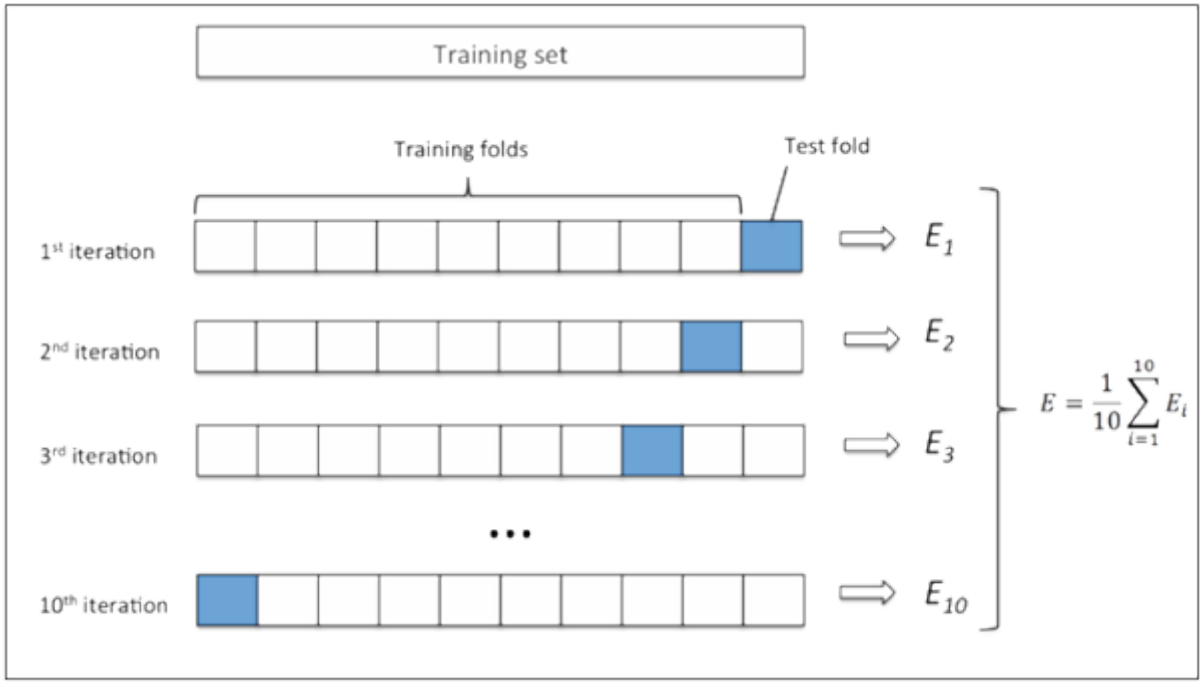
精确率（查准率、$Precision$）：$\frac{TP}{Positive}$

召回率（查全率、$Recall$）：$\frac{TP}{True}$

F1-score：$\frac{2TP}{2TP+FN+FP}=\frac{2\cdot Precision\cdot Recall}{Precision+Recall}$

```
from sklearn.metrics import classification_report
"""
    y_true       : Truth (correct) target.
    y_pred       : Estimated targets.
    target_names : list of strings, display names matching the labels.
    @return      : the precision, recall, F1-score for each class.
"""
```

交叉验证（Cross validation）



n等分的数据进行交叉验证称为——**n折交叉验证。**

超参数搜索（网格搜索）

最优参数选择模型

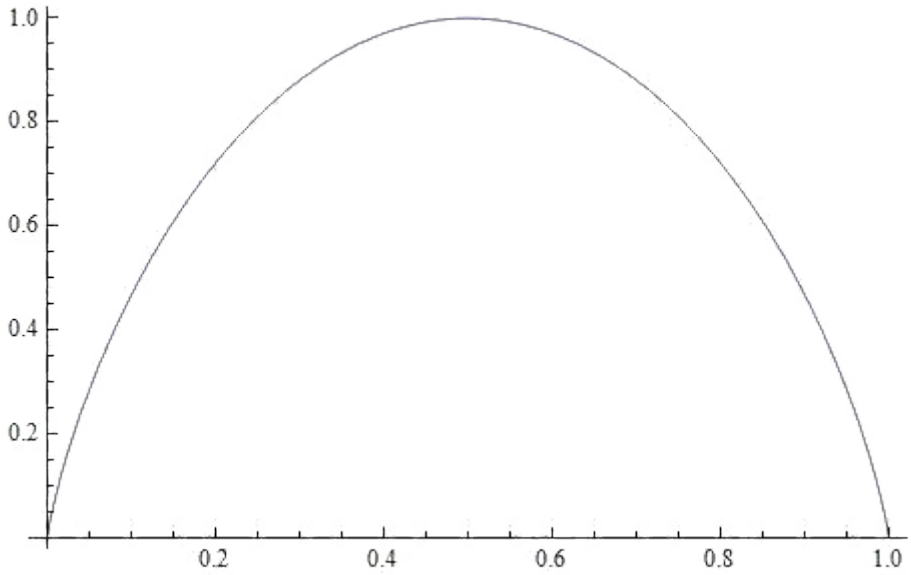
```
from sklearn.model_selection import GridSearchCV
"""
    estimator   : 估计对象（使用的算法实例）
    param_grid  :
    cv          : 使用几折交叉验证
    fit
    score
"""
```

信息熵

熵即不确定性，熵越大则不确定性越大，获取信息意味着消除熵。

信息熵：$H = \sum_i p_i \cdot log(p_i^{-1})$

二元信源（抛硬币）的信息熵：



信息增益

$g(data, feature) = H(data) - H(data|feature)$

表示得知该特征值信息使不确定性减少的程度。

算法

K近邻

欧氏距离：$\rho = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$

朴素贝叶斯

> 对数据缺失不敏感，分类准确性高、速度快、不需要调参。但由于使用了特征独立性假设，特征存在关联时，效果不好。

联合概率：A、B为独立事件，$P(AB) = P(A) \cdot (B)$。

条件概率：$P(A|B) = \dfrac{P(AB)}{P(B)}$

贝叶斯公式：$P(Class|feature) = \dfrac{P(feature|Class) \cdot P(Class)}{P(feature)}$

- $Class$：：文档类别。
- $feature_i$：给定文档的特征值（词频统计）。

拉普拉斯平滑系数：$P(F_1|C) = \dfrac{N_i + \alpha}{N + m\alpha}$

- α：指定的系数，一般为1
- m：文档中统计出的特征词个数

决策树

- ID3：信息增益，最大准则
- C4.5：信息增益比，最大准则
- CART
  - 回归树：平方误差最小
  - 分类树：基尼系数最小（默认）

随机森林

基于决策树的集成学习算法

线性回归

$$f(x) = b + \sum_{i=1}^{d} w_i \cdot x_i$$

- $w = [w_1, w_2, \ldots, w_n]^T$
- $x = [x_1, x_2, \ldots, x_m]^T$

正规方程

> 特征值过于复杂时，求解速度过慢。

$$w = (x^T x)^{-1} x^T y$$

- x：特征值矩阵
- y：目标值矩阵

梯度下降

损失函数（loss） / 代价函数（cost）：衡量**预测值**和**目标值**之间误差大小的函数。

$$w_i = -w_i - \alpha \frac{\partial \left[loss(w_0 + w_i \cdot x_i) \right]}{\partial w_i}$$

岭回归

带正则化的线性回归，可有效防止过拟合。

逻辑回归

带激活函数和正则化

聚类

无监督学习

模型保存与加载

```python
from sklearn.externals import joblib


def model_load(filename):
    """
    模型加载
    :param filename:
    :return: estimator
    """
    return joblib.load(filename)


def model_save(estimator, filename):
    """
    模型保存
    :param estimator: 估计器对象（已执行过训练操作）
    :param filename: 保存位置（文件以pkl为扩展名）
    :return:
    """
    joblib.dump(estimator, filename)
```