数据科学与工程导论

Introduction to Data Science and Engineering

第11章 机器学习方法

1 机器学习的基础

2 Sklearn库的基本使用

- **什么是机器学习**
  - 机器学习是一门多领域交叉学科，涉及概率论、统计学、逼近论、凸分析、算法复杂度理论等多门学科。
  - 机器学习专门研究计算机怎样模拟或实现人类的学习行为，以获取新的知识或技能，重新组织已有的知识结构使之不断改善自身的性能。
  - 机器学习是人工智能的核心，是使计算机拥有智能的重要途径。
  - 应用遍及人工智能的各个领域，它主要使用归纳，综合而不是演绎。

何为机器学习

- 机器学习的定义
  - **机器学习**（Machine Learning）是一门致力于通过数据以及以往的经验，优化计算机程序性能的学科。
  - 假设采用T代表**任务**(task)，P代表任务T的**性能**(performance)，E代表**经验**(experience)。机器学习研究的主要内容是利用经验E通过**学习算法**(learning algorithm)提高任务T的性能P，最终从经验中产生**模型**(model)。

何为机器学习

- 垃圾邮件分类系统
  - 任务T：判断给定邮件是否是垃圾邮件
  - 性能指标P：分类的准确度
  - 经验来源E：大量的邮件数据以及其对应的类别
- 鸢尾花分类系统
  - 任务T：确定给定鸢尾花的类别
  - 性能指标P：分类的准确度
  - 经验来源E：大量的鸢尾花数据以及其对应的品种

何为机器学习

- **机器学习**
  - **机器学习的目的**：机器学习的目的是**分类**和**预测**。所谓分类是根据输入数据，判别这些数据隶属于哪个类别(Category)。而预测则是根据输入数据，计算一个输出值(Numeric)。输入数据一般为一个**向量**，也称为**特征**(Feature)，输出则是一个**分类**或者一个**数值**。
  - **机器学习的基本过程**：机器学习的基本过程是用**训练数据**(包含输入数据和预期输出的分类或者数值)训练一个**模型**(Model)，利用这个模型，就可以对新的实例数据(Instances)进行分类和计算一个预测值。

何为机器学习

- **机器学习应用**
  - **数据挖掘**：从大量的数据中通过算法搜索隐藏于其中信息的研究。
    - 数据挖掘目前主要应用于**数据统计分析，趋势预测，异常检测，流量数据分析，风险评估**等多个方面。
  - **计算机视觉**：通过对采集的**图片**或者**视频**进行处理，最终获取所需被拍摄对象的数据和信息的一门学科。
    - 计算机视觉是使用计算机及相关的设备对生物视觉的一种模拟，目前主要用于**医学图像处理，导弹制导**以及**无人机**和**无人驾驶车辆**等应用领域。

何为机器学习

- **机器学习应用**
  - **自然语言处理**：实现人与计算机之间通过自然语言进行有效通信的理论与方法。
    - 自然语言即人们日常使用的语言，通过自然语言处理，主要实现**文本分类与聚类**，**信息检索和过滤**，**机器翻译**等多种应用。
  - **生物特征识别**：利用人体固有的**生理特征**或**行为特征**实现个人身份鉴定的技术。
    - 生理特征（虹膜，指纹，声纹，DNA等固有特征）、行为特征（步态，签名等习惯）。目前较为火热的研究方向**有人脸识别**，**亲子鉴定**，**指纹识别**，**虹膜识别**等身份鉴定技术。

何为机器学习

- **机器学习**
  - 机器学习是一类算法的总称，这些算法企图从大量历史数据中挖掘出其中隐含的规律，并用于对新的数据进行预测或者分类。更具体的说，机器学习可以看作是寻找一个**函数**，输入是大量的**样本数据**，输出是**期望结果**。
  - 要进行机器学习，必须要获取**经验E**，而计算机中经验一般以数据的形式存在，通过使用学习算法对数据进行学习我们最终可以获取模型。
  - 机器学习中的经验通常从**数据集**中获取，数据集中包含有若干的样本数据，每个样本数据都包含若干个**属性**对应的**属性值**。

何为机器学习

- **数据集**(data set)
  - 一组记录的集合称为一个**数据集**，其中每条记录是关于一个事件或对象的描述，称为一个**示例**(instance)或**样本**(sample) 。
  - 样本中内容包含有对事物一些方面的性质的描述，称为**属性**(attribute)或者**特征**(feature)。属性上的取值成为**属性值**(attribute value)。
  - 属性张成的空间称为**属性空间**(attribute space)。例如一个对象有三个属性，则它们张成一个用于描述该对象的三维空间，每一个对象都可以在这个空间中找到自己的坐标位置。由于空间中的每个点对应一个坐标向量，因此我们也把一个示例称为一个**特征向量**（feature vector）。

机器学习基本术语

- **数据集**(data set)
  - $D = \{x_1, x_2, x_3, \cdots, x_m\}$表示包含m个示例的数据集D
  - $x_i = (x_{i1}, x_{i2}, x_{i3}, \cdots, x_{id})$是$d$维样本空间$X$中的一个向量$x_i$，$x_i \in X$。其中$x_{ij}$是$x_i$在第j个属性上的取值，$d$被称为样本$x_i$的维数

机器学习基本术语

- **训练**(training)
  - 从数据中学得模型的过程被称之为**学习**(learning)或**训练**(training)，在通过使用学习算法对数据进行训练的过程中，每一个样本被称之为**训练样本**，训练样本组成的数据集称为**训练集**。
  - 训练模型的目的是**预测**(prediction)关于数据的某种潜在的规律，通过学习过程逐渐逼近这个规律。而学习过程只依靠训练样本的数据信息是不够的，我们需获得训练样本的结果信息，示例结果的信息，称为**标记**(label)。
  - 一般的，用(x_i, y_i)表示第i个样例，其中$\mathrm{y}_i \in \mathrm{Y}$是示例$\mathrm{x}_i$的标记，Y是所有标记的集合。

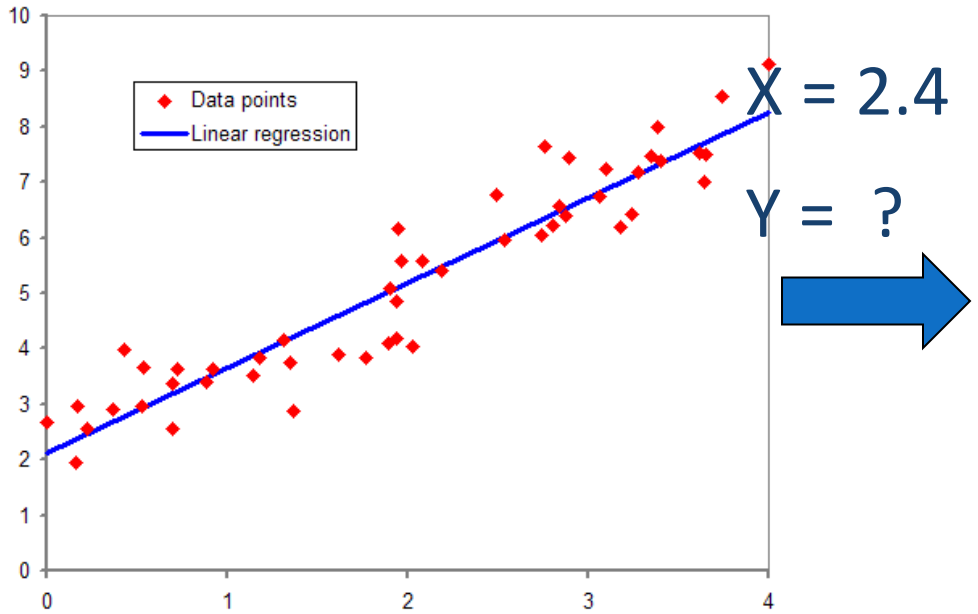
机器学习基本术语

- **预测**(prediction)
  - 根据预测的结果不同可以将学习任务分为不同类别。如果需要预测的是离散值，则此类任务称为**分类**(classification)，如果预测的是连续值，则将此类任务称之为**回归**(regression)。
  - **聚类**(clustering)将训练集的样本分为若干组，每个组称为一个**簇**(cluster)，这些生成的簇对应一些潜在的概念划分，而究竟按照何种策略进行聚类是没有预先定义的，通过聚类算法在学习过程中自己生成。
  - 回归以及分类都需要使用到标记信息，而聚类不需要使用标记信息。
  - 按照训练数据有无标记信息可以将学习任务分为两大类：**监督学习**(supervised learning)和**无监督学习**(unsupervised learning)，其中分类和回归是监督学习的代表，聚类是无监督学习的代表。

机器学习基本术语

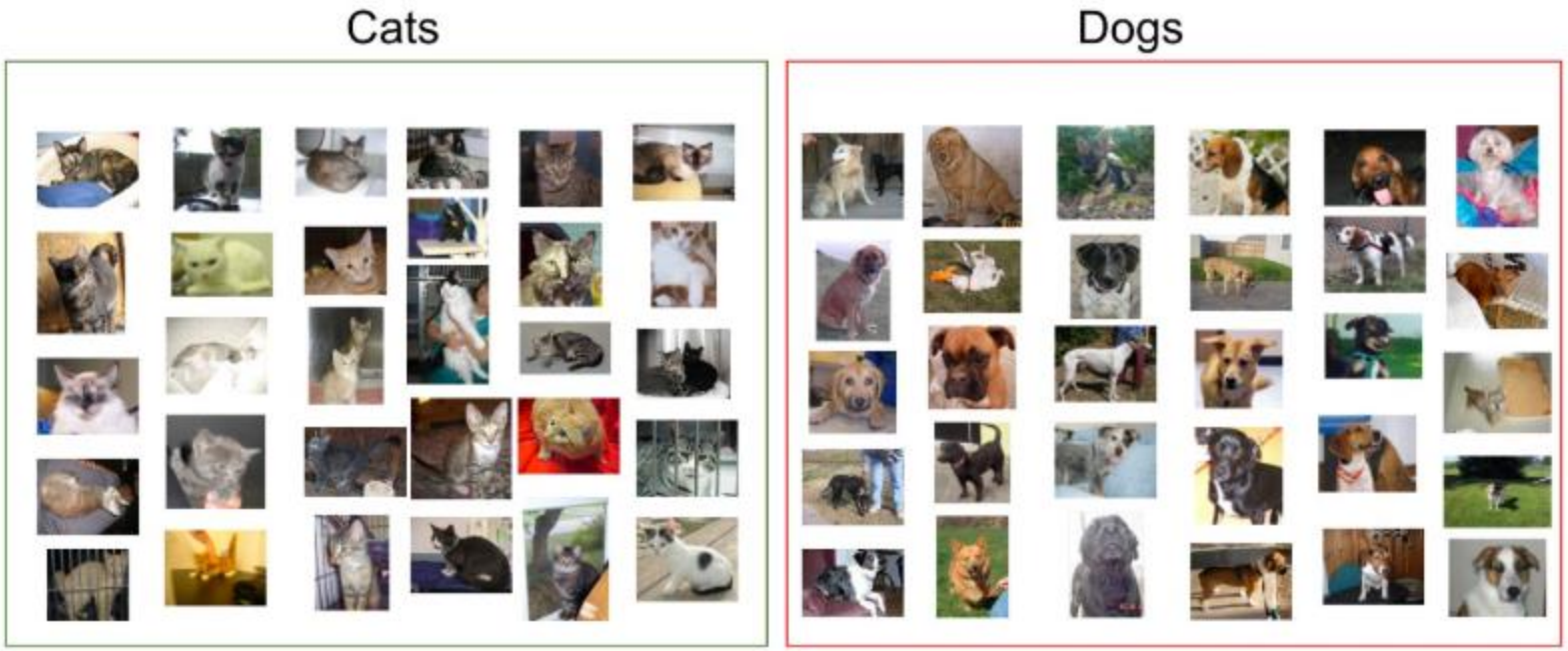
• 回归

- 回归方法是一种对数值型连续随机变量进行预测和建模的监督学习算法。例如，房价预测、股票走势或测试成绩等连续变化的案例。
- 通过给定数据集拟合出一条曲线，对于将要预测的样本放入曲线计算出预测结果，其输出值为连续的。

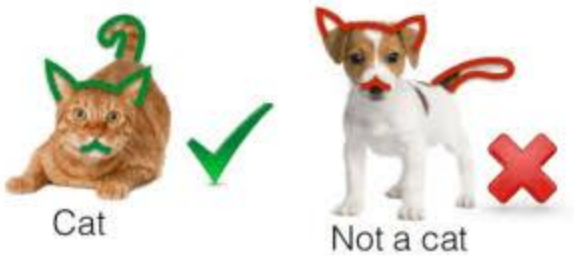


机器学习基本术语

- 分类

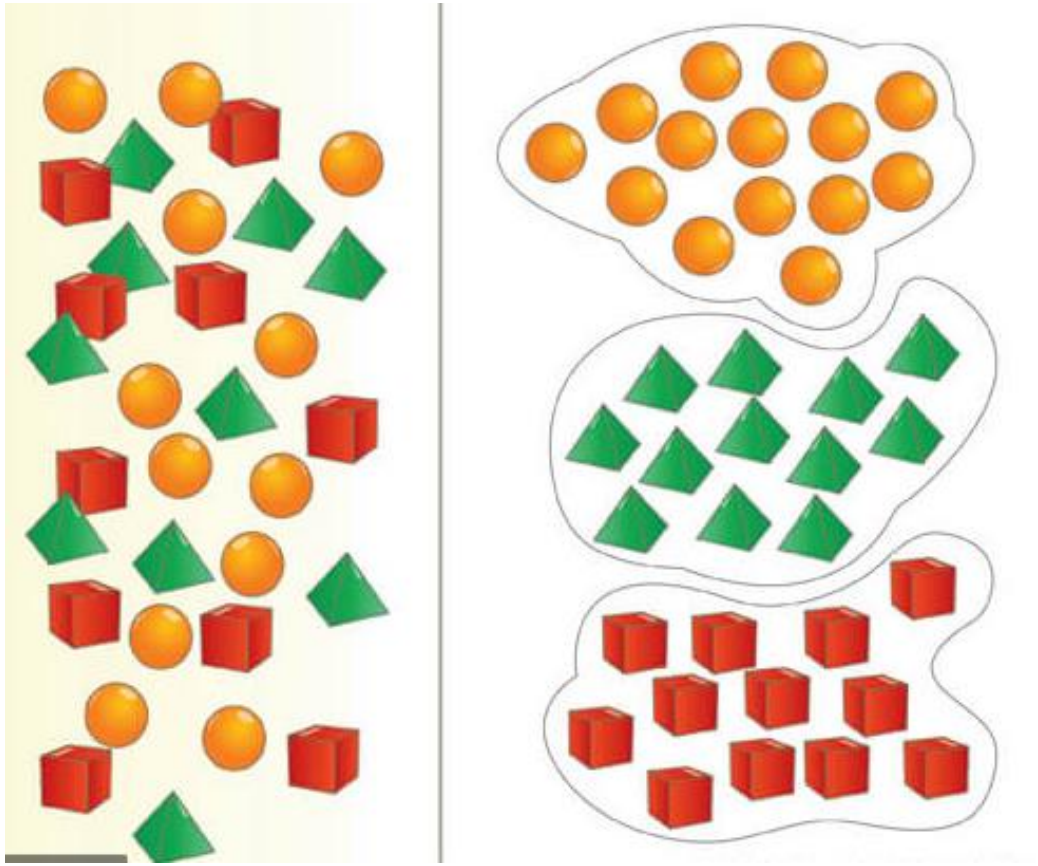


Sample of cats & dogs images from Kaggle Dataset



机器学习基本术语

- 聚类



机器学习基本术语

- **预测**(prediction)
  - 学得模型后可以采用样本对其进行预测，过程称为**测试**(testing)，被预测的样本称为**测试样本**(testing sample) 。例如在习得f之后，对测试样本x进行测试，测试的标记为y = f(x)。
  - 机器学习的目标是由训练集学得的模型可以适用于非训练集样本的预测。我们希望对于未出现在训练集中的数据采用预测模型也可以得到较好的预测效果。
  - 学得模型适用于新样本的能力称为**泛化**(generalization)。我们的模型设计虽然只是基于整个样本空间很少的部分进行预测，但是我们的目标是其在整个样本空间都有较好的预测能力，所以必须保证我们的模型拥有较强的泛化能力。

机器学习基本术语

• 模型评估方法

- 模型评估方法是对数据集D如何划分为**训练集**S和**测试集**T的方法。为了得到更加准确的测试结果，测试集应该尽可能与训练集互斥。即测试样本尽量不在训练集中出现，未在训练过程中使用。
- 目前常见的做法有**留出法**、**交叉验证法**和**自助法**。

机器学习基本术语

- **模型评估方法**
  - **留出法**（hold-out）
    - 将数据集D划分为两个互斥的集合，其中一个集合作为**训练集**S，另一个作为**测试集**T。在S上训练出模型后，用T来评估其测试误差，作为对泛化误差的估计。训练集和测试集的划分要尽可能保持**数据分布的一致性**，避免因数据划分过程引入额外的偏差而对最终结果产生影响。

- 模型评估方法
  - 留出法（hold-out）
    - 例：数据集D包含600个正样本，400个负样本，数据集D划分为70%样本的训练集和30%样本的测试集。
    - 为了保证训练和测试正负样本的比例与数据D比例相同，采用分层抽样的方法。先从600个正样本随机抽取420次，400个负样本随机抽取280次，然后剩下的样本集作为测试集，分层抽样保证了训练集的正负样本的比例与数据集D的正负样本比例相同。

- **模型评估方法**
  - **交叉验证法**（cross validation）
    - 先将数据集D划分为**K**个大小相似的**互斥子集**，每个子集Di通过分层采样得到（如上页所述，为了保证正负样本的比例与数据集D的比例相同）。
    - 然后用k-1个子集的并集作为训练集，余下的子集作为测试集；这样就获得k组训练/测试集。
    - 从而进行k次训练和测试，最终返回的是这k个测试结果的均值。通常把交叉验证法称为**k折交叉验证法**，k最常用的取值是10，此时称为10折交叉验证。

- **模型评估方法**
  - **交叉验证法**（cross validation）
    - 若数据集D中包含m个样本，先将数据集D划分为m个大小相似的子集，每个子集只有**一个样本数据**，则得到了交叉验证法的一个特例，**留一法**（Leave-One-Out，简称**LOO**）。
      - **优点**:它不受随机样本划分的影响，因为每个子集都会包含一个样本，所以留一法的评估结果最准确。
      - **缺点**:数据集较大时，训练m个模型计算开销太大。

- **模型评估方法**
  - **自助法**（bootstrapping）
    - 我们希望评估的是用**原始数据集**D训练出的模型，但是留出法和交叉验证法训练的数据集比原始的数据集D小，这必然会引入因训练数据集不同导致的估计偏差，所以我们引入了自助采样法进行模型评估。

- 模型评估方法
  - 自助法（bootstrapping）
    - 自助法是**有放回抽样**，给定包含m个样本的数据集D，我们对它进行采样产生数据集D’：
    - 每次有放回的随机从D中挑选一个样本，将该样本拷贝放入D’；
    - 重复执行m次，就得到了包含m个样本数据集D’，这就是自助采样的结果。初始数据集D中有一部分样本会在数据集D’中多次出现，也有一部分样本不会在数据集D’中出现。
    - 通过自助采样，初始数据集D中约有**36.8%**的样本未出现在采样数据集D’，于是我们可将D’用作训练集，我们仍有数据总量约1/3的、没在训练集中出现的样本作为测试集用于测试。

• 模型评估方法

	采样方法	与原始训练数据集的分布是否相同	相比原始训练数据集的容量	是否适用小数据集	是否适用大数据集	是否存在估计偏差
留出法	分层抽样	否	变小	否	是	是
交叉验证法	分层抽样	否	变小	否	是	是
自助法	放回抽样	否	不变	是	否	是

机器学习基本术语

• 性能度量

机器学习基本术语

• 分类性能度量

• 混淆矩阵（confusion matrix）

- 它是一种特定的矩阵用来呈现算法性能的可视化效果,每一列代表预测值，每一行代表的是实际的类别。
- **真正**(True Positive , **TP**)：被模型预测为正的正样本。
- **假正**(False Positive , **FP**)：被模型预测为正的负样本。
- **假负**(False Negative , **FN**)：被模型预测为负的正样本。
- **真负**(True Negative , **TN**)：被模型预测为负的负样本。

表 5-2 混淆矩阵

真实数据	预测结果	
	正样本	负样本
正样本	TP	FN
负样本	FP	TN

- **分类性能度量**
  - **准确率（Accuracy）**
    - 所有被分类正确的点在所有点中的概率
    - Accuracy = (TP+TN)/(TP+FN+FP+TN)
  - **精确率（Precision）**
    - 针对预测正确的正样本而不是所有预测正确的样本。表现为预测出是正的里面有多少真的为正的概率，可理解为**查准率**
    - Precision = TP/(TP+FP)

机器学习基本术语

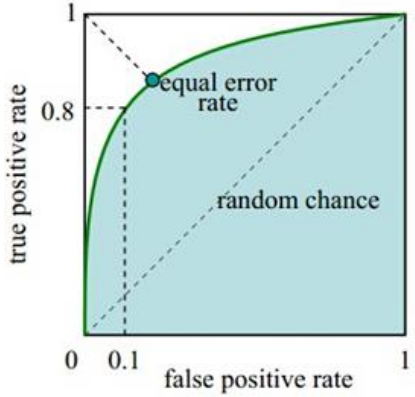
- **分类性能度量**
  - **召回率（Recall）**
    - 正确预测的正例数 /实际正例总数，也称**查全率**
    - Recall = TP/(TP+FN)
  - **F1 score**
    - 2/F1 = 1/Precision + 1/Recall
    - $\mathrm{F1} = \frac{2 \times \mathrm{Precision} \times \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}$
    - F值是精确率和召回率的调和值，更接近于两个数较小的那个，所以精确率和召回率接近时，F值最大。

机器学习基本术语

• 分类性能度量

• ROC曲线

- ROC空间将伪阳性率（FPR）定义为 *X* 轴，真阳性率（TPR）定义为 *Y* 轴。
  - TPR：在所有实际为阳性的样本中，被**正确地**判断为阳性的比率。 TPR = TP / (TP + FN)
  - FPR：在所有实际为阴性的样本中，被**错误地**判断为阳性的比率。 FPR = FP / (FP + TN)
- 从 (0, 0) 到 (1,1) 的对角线将ROC空间
  - 划分为左上 / 右下两个区域，在这条线
  - 的以上的点代表了一个好的分类结果（
  - 胜过随机分类）。



- 分类性能度量
  - AUC
    - **AUC**（Area Under Curve）被定义为ROC曲线下的面积(ROC的积分)，通常大于0.5小于1。随机挑选一个正样本以及一个负样本，分类器判定正样本的值高于负样本的概率就是AUC 值。AUC值(面积)越大的分类器，性能越好。
      - AUC = 1，是完美分类器，采用这个预测模型时，存在至少一个阈值能得出完美预测
      - 0.5 < AUC < 1，优于随机猜测。这个分类器（模型）妥善设定阈值的话，能有预测价值
      - AUC = 0.5，跟随机猜测一样（例：丢硬币），模型没有预测价值
      - AUC < 0.5，比随机猜测还差；但只要总是反预测而行，就优于随机猜测

- **回归与性能度量**
  - **平均绝对误差**(Mean absolute error，MAE)
    - $MAE = \frac{\sum_{i=1}^{n} |y_i - \widehat{y_i}|}{n}$，$\widehat{y_i} = f(x_i)$
    - MAE又被称为**L1范数损失**，表示预测值与观察值之间的绝对误差的平均值。
  - **均方误差**(Mean squared error MSE)
    - $MSE = \frac{\sum_{i=1}^{n} (y_i - \widehat{y_i})^2}{n}$
    - MSE又被称为**L2范数损失**，表示预测值与观察值之间的误差平方的平均值。

机器学习基本术语

- **回归性能度量**
  - **均方根误差**（RMSE）
    - $RMSE = \sqrt{MSE}$
  - 以上三种回归评价指标的取值大小与应用场景有关，很难定义统一的规则评判模型的好坏，所以是否存在类似于分类中的准确率的评价指标，取值范围为0-1，在不同的应用场景下都可以使用这一评价标准呢？

- **回归性能度量**
  - **决定系数：** R^2（R-Square）
    - $R^2 = 1 - \frac{\sum_i (\hat{y}_i - y_i)^2}{\sum_i (\bar{y}_i - y_i)^2}$
    - 其中分母为标签Y的方差，分子为MSE。我们可以根据决定系数的取值，判断模型性能的好坏。R^2的取值范围为[0,1]。如果模型的决定系数为0，表示模型的拟合效果很差，如果结果为1，说明拟合曲线无错误。R^2取值越大，说明模型的拟合效果越好。
    - 缺点：随着样本数量的增加R^2值会随之增加，所以无法定量的说明准确程度。

- **回归性能度量**
  - **校正决定系数：** Adjusted R^2（Adjusted R-Square）
    - $Adjusted\ R^2 = 1 - \frac{(1-R^2)(n-1)}{n-p-1}$
    - n:样本数量
    - p:特征数量
    - $Adjusted\ R^2$抵消了样本数量对R^2的影响，可以定量的说明准确程度。

- **聚类性能度量**
  - 聚类性能度量也称聚类**有效性指标**，用来评估聚类结果的好坏。
  - 好的聚类结果→聚类结果的**簇内相似度高**且保证**簇间相似度低**。
  - 聚类性能度量分类：
    - **外部指标**
    - **内部指标**

- **聚类性能度量**
  - **外部指标：**
    - 将聚类结果与某个“**参考模型**”进行比较，例如与领域专家的划分结果进行比较（类似对数据进行标记）。我们默认参考模型的性能指标是对样本的最优划分，度量的目的就是使聚类结果与参考模型尽可能相近。
    - 核心思想是聚类结果中被划分在同一簇样本与参考模型样本也被同样划分到一个簇的概率越高越好。
    - 常用的外部指标有：**Jaccard系数，FM指数，Rand指数**

• 聚类性能度量

• 外部指标：

- 对于给定数据集$\boldsymbol{D}=\{\boldsymbol{x_1},\boldsymbol{x_2},\boldsymbol{x_3},\cdots,\boldsymbol{x_n}\}$，经过聚类算法划分的簇为$\boldsymbol{C}=\{\mathbf{C_1},\mathbf{C_2},\cdots,\mathbf{C_k}\}$，参考模型给出的簇划分为$\boldsymbol{C}^*=\{\mathbf{C}_1^*,\mathbf{C}_2^*,\cdots,\mathbf{C}_s^*\}$。同时我们令$\boldsymbol{l}$与$\boldsymbol{l}^*$分别表示数据在$\boldsymbol{C}$与$\boldsymbol{C}^*$中的簇标记向量。将样本两两配对考虑，定义
  - a= |SS|，SS ={ $(x_i,x_j)\mid l_i=l_j,l_i^*=l_j^*,i<\mathrm{j}$ }
  - b= |SD|，SD ={ $(x_i,x_j)\mid l_i=l_j,l_i^*\neq l_j^*,i<\mathrm{j}$ }
  - c= |DS|，DS ={ $(x_i,x_j)\mid l_i\neq l_j,l_i^*=l_j^*,i<\mathrm{j}$ }
  - d= |DD|，DD ={ $(x_i,x_j)\mid l_i\neq l_j,l_i^*\neq l_j^*,i<\mathrm{j}$ }
  - 其中集合SS包含在$\boldsymbol{C}$中隶属相同簇并且在$\boldsymbol{C}^*$中也隶属相同簇的样本对，a为集合SS中样本对的个数，由于每对样本只可能出现在其中一个集合，所以$a+b+c+d=n(n-1)/2$。

- **聚类性能度量**
  - **外部指标：**
    - Jaccard系数（Jaccard Coefficient,JC）
      - $\mathrm{JC} = \frac{a}{a+b+c}$
    - FM指数（Fowlkes and Mallows Index,FMI）
      - $\mathrm{FMI} = \sqrt{\frac{a}{a+b} \cdot \frac{a}{a+c}}$
    - Rand指数（Rand Index,RI）

$$\mathrm{RI} = \frac{2(a+d)}{n(n-1)}$$

    - 以上性能度量指标的取值均在区间[0,1]内，取值越大代表聚类效果越好。

- **聚类性能度量**
  - **内部指标：**
    - 直接考察聚类结果而不利用参考模型，通过计算簇内样本间的距离以及簇间样本的距离评估模型的性能。
    - 核心思想是用簇内样本间距离模拟簇内相似度，簇间样本距离模拟簇间相似度，通过计算距离构建性能指标。
    - 常用的内部指标有：**DB指数，Dunn指数**

- **聚类性能度量**
  - **内部指标：**
    - 对于给定数据集 $D=\{x_1,x_2,x_3,\cdots,x_n\}$，经过聚类算法划分的簇为 $C=\{C_1,C_2,\cdots,C_k\}$，定义
      - $avg(C)=\frac{2}{|C|(|C|-1)}\sum_{1\le i<j\le|C|}dist(x_i,x_j)$
      - $diam(C)=max_{1\le i<j\le|C|}\ dist(x_i,x_j)$
      - $d_{min}(C_i,C_j)=min_{x_i\in C_i,x_j\in C_j}\ dist(x_i,x_j)$
      - $d_{cen}(C_i,C_j)=dist(u_i,u_j)$
      - 其中dist(,)用于计算两个样本之间的距离；u代表簇C的中心点$u=\frac{1}{|C|}\sum_{1\le i\le|C|}x_i$

机器学习基本术语

- **聚类性能度量**
  - **内部指标：**
    - $\mathbf{avg(C)}$对应簇内样本间的平均距离，$\mathbf{diam(C)}$对应于该簇内样本间的最远距离，$\mathbf{d_{min}(C_i, C_j)}$对应两簇间样本的最近距离，$\mathbf{d_{cen}(C_i, C_j)}$对应两簇中心点的距离，基于以上指标可得以下的聚类性能度量的内部指标,其中DBI值越小越好，DI值越大越好。
    - **DB指数（Davies-Bouldin Index,DBI）**
      - $\mathbf{DBI} = \frac{1}{k}\sum_{i=1}^{k} \max_{j \neq i} \left(\frac{avg(C_i)+avg(C_j)}{d_{cen}(C_i,C_j)}\right)$
    - **Dunn指数（Dunn Index,DI）**

$$\mathbf{DI} = \min_{1 \le i \le k} \left\{ \min_{j \neq i} \left(\frac{d_{min}(C_i, C_j)}{\max_{1 \le x \le k} diam(C_x)} \right) \right.$$

- 机器学习的发展是整个人工智能发展史上颇为重要的一个分支。下面介绍人工智能的发展阶段以及同一时期机器学习的主要成果：

• 第一阶段：

- 20世纪50年代初到60年代初：人工智能研究进入**推理期**。认为只需要为机器赋予逻辑推理能力，机器就会拥有智能。
- 1952年，IBM科学家**亚瑟·塞缪尔**研制了一个**西洋跳棋程序**，这是**人工智能下棋问题**的由来。
- 基于贝叶斯决策理论的**贝叶斯分类器**也起步于20世纪50年代，通过计算后验概率实现数据分类任务。
- 50年代中期开始出现基于神经网络的“**连接主义**”（Connectionism）学习；1957年，**罗森·布拉特**（F. Rosenblatt）提出了**感知机**（Perceptron），感知机成功处理线性分类问题，为现在的神经网络以及深度学习开创了基础。

发展历程

- **第二阶段：**
  - 20世纪60年代中叶到70年代初：**冷静时期**。人们发现仅有逻辑推理能力无法使机器具有智能。
  - 这一时期基于逻辑表示的“**符号主义**”（Symbolism）学习技术蓬勃发展，**P. Winston**的结构学习系统，**R. S. Michalski**的基于逻辑的归纳学习系统，以及**E. B. Hunt**的概念学习系统。
  - 这一时期出现了大量的机器学习算法。例如1967年的**最近邻算法**（The nearest neighbor algorithm），一种基于模板匹配思想的分类算法，虽然简单，但很有效，至今仍在被使用。
  - 同年提出的**k均值算法**是所有聚类算法中变种和改进型最多的算法，此后出现了大量的改进算法，也有大量成功的应用。

发展历程

- **第三阶段：**
  - 20世纪70年代中叶到80年代初：**知识期**。人们提出要想使机器拥有智能，必须设法让机器拥有知识，这个阶段大量的**专家系统**的问世获得了较多的应用领域的成果。
  - 这一时期机器学习的研究在全世界开始兴起。1980年，在**卡内基梅隆大学**(CMU)召开了第一届**机器学习国际研讨会。**
  - 1981年，**多层感知器**(MLP)在**伟博斯**的神经网络**反向传播(BP)算法**中具体提出，BP算法仍然是今天神经网络架构的关键因素。

- **第四阶段：**
  - 20世纪80年代中期到现在：**学习期**。由于专家系统面临着人将知识总结出来相当困难，于是提出让机器能够自主学习知识，机器学习正式走入人工智能舞台。八十年代中期到九十年代中期是机器学习成为一个独立的学科领域，各种机器学习技术百花初绽的时期。
  - 基于神经网络的**连接主义**学习的主要成果有于1986年诞生的用于训练多层神经网络的真正意义上的**反向传播算法**，奠定了神经网络走向完善和应用的基础。
  - 1989年，**LeCun**设计出了第一个真正意义上的**卷积神经网络**，用于手写数字的识别，是现在被广泛使用的深度卷积神经网络的鼻祖
  - 基于逻辑学习的**符号主义**学习的研究成果有1986年**昆兰**提出的**ID3决策树算法**，虽然简单，但可解释性强，这使得决策树至今在一些问题上仍被使用。

发展历程

- **第四阶段：**
  - 二十世纪九十年代中期，**统计学习**（Statistical Learning）发展迅速，期间出现了以**支持向量机**（Support Vector Machine，SVM）技术为代表的机器学习算法。由**瓦普尼克**和**科尔特斯**在大量理论和实证的条件下提出的支持向量机算法将机器学习社区划分为神经网络社区和支持向量机社区。
  - 同一时期**AdaBoost**代表**集成**学习算法，通过将一些简单的弱分类器集成起来使用，居然能够达到惊人的精度。

- **第四阶段：**
  - 二十一世纪初至今，随着大数据时代的到来，数据量的增大以及计算机计算能力的大幅增强，深度学习（Deep Learning）的时代到来。
  - 2006年，神经网络研究领域领军者**Hinton**提出了神经网络**Deep Learning**算法，使神经网络的能力大大提高，向支持向量机发出挑战，开启了**深度学习**在学术界和工业界的浪潮。

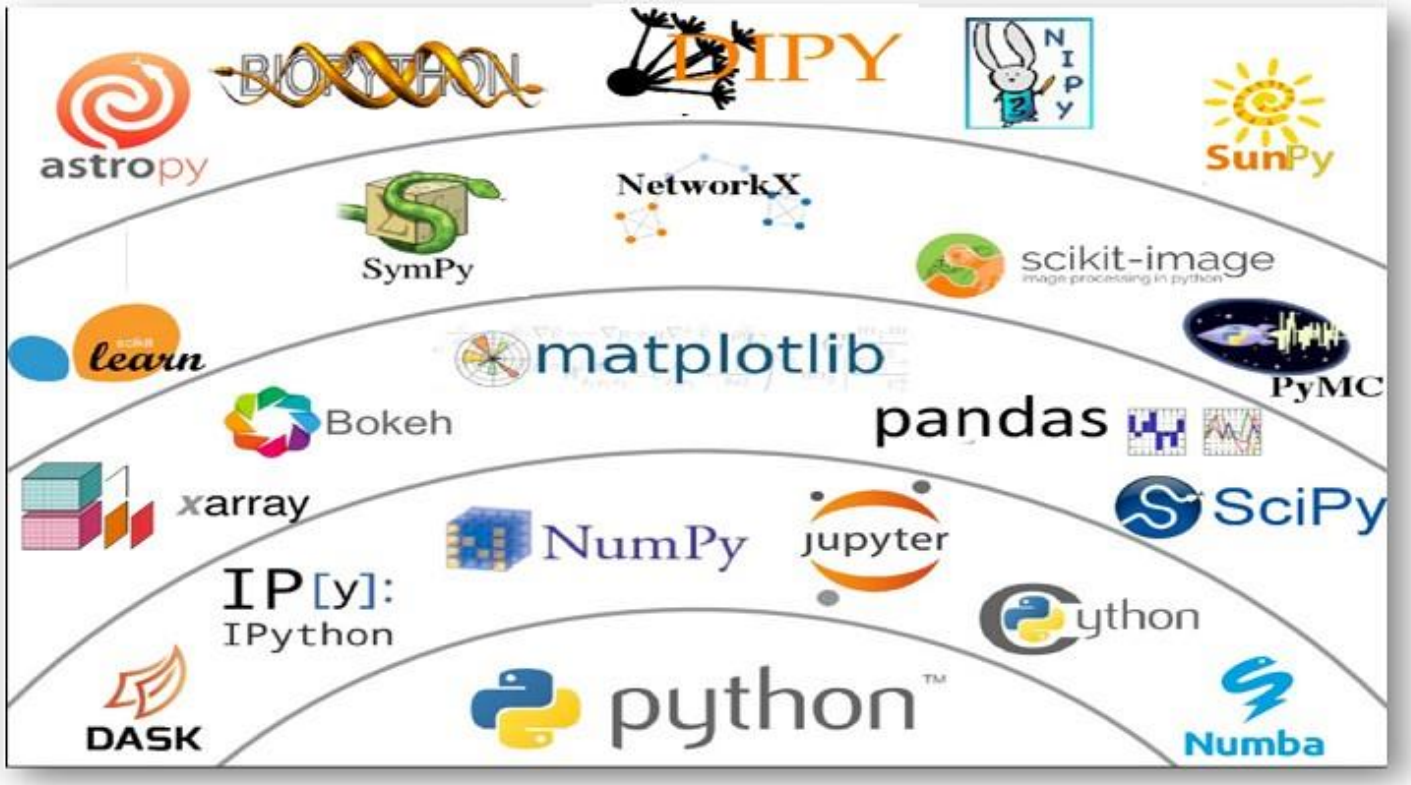
- 5.2 Sklearn库基本使用
  - 5.2.1 Sklearn库简介
  - 5.2.2 基本使用介绍

第 11 章
机器学习方法

1 机器学习的基础

2 Sklearn库的基本使用

- 机器学习常用工具



Sklearn库基本应用

• 机器学习工具包

- Scikit-learn是面向Python的机器学习软件包。Scikit-learn软件包支持主流的有监督机器学习方法(Supervised Learning Algorithm)、无监督机器学习方法(Unsupervised Learning Algorithm)。
  - 有监督的机器学习方法，包括通用的线性模型、支持向量机、决策树、贝叶斯方法等。
  - 无监督的机器学习方法，包括聚类、因子分析、主成份分析、无监督神经网络等。

简介

Scikit-learn安装

- 安装要求：
  - Python>=3.5,Numpy>=1.11.0,Scipy>=0.17.0
  - 如果使用Python2.7可以采用Scikit-learn0.20版本
- 安装方式：
  - Pip命令安装：pip install –u scikit-learn
  - Conda命令安装：conda install scikit-learn
- 官方文档：
  - Scikit-learn英文文档：https://scikit-learn.org/stable/
  - Scikit-learn中文文档： https://sklearn.apachecn.org/

简介

• Scikit-learn

Classification

Identifying to which category an object belongs to.

Applications: Spam detection, Image recognition.
Algorithms: SVM, nearest neighbors, random forest, ... — Examples

Regression

Predicting a continuous-valued attribute associated with an object.

Applications: Drug response, Stock prices.
Algorithms: SVR, ridge regression, Lasso, ... — Examples

Clustering

Automatic grouping of similar objects into sets.

Applications: Customer segmentation, Grouping experiment outcomes
Algorithms: k-Means, spectral clustering, mean-shift, ... — Examples

Dimensionality reduction

Reducing the number of random variables to consider.

Applications: Visualization, Increased efficiency
Algorithms: PCA, feature selection, non-negative matrix factorization. — Examples

Model selection

Comparing, validating and choosing parameters and models.

Goal: Improved accuracy via parameter tuning
Modules: grid search, cross validation, metrics. — Examples

Preprocessing

Feature extraction and normalization.

Application: Transforming input data such as text for use with machine learning algorithms.
Modules: preprocessing, feature extraction. — Examples

简介

- **Scikit-learn快速使用**
  - 传统的机器学习任务从开始到建模的一般流程就是：获取数据→数据预处理→训练模型→模型预测，评估→保存。**1.获取数据：**
    - Sklearn中包含了大量的优质的数据集，我们可以使用这些数据集实现出不同的模型，从而提高动手实践能力，同时这个过程也可以加深对理论知识的理解和把握。要使用sklearn中的数据集，必须导入datasets模块。

```
In[1]    from sklearn import datasets
In[2]    dir(datasets)
```

应用

• Scikit-learn快速使用

• 1.获取数据：

	数据集名称	调用方式	适用算法	数据规模
小数据集	波士顿房价数据集	load_boston()	回归	506*13
	鸢尾花数据集	load_iris()	分类	150*4
	糖尿病数据集	load_diabetes()	回归	442*10
	手写数字数据集	load_digits()	分类	5620*64
大数据集	Olivetti脸部图像数据集	fetch_olivetti_faces()	降维	400*64*64
	新闻分类数据集	fetch_20newsgroups()	分类	-
	带标签的人脸数据集	fetch_lfw_people()	分类；降维	-
	路透社新闻语料数据集	fetch_rcv1()	分类	804414*47236
注：小数据集可以直接使用，大数据集在第一次使用的时会自动下载				

应用

- Scikit-learn快速使用
  - 1.获取数据：
    - Iris数据集导入：
      - Iris(鸢尾花)数据集是用于分类的经典数据集，其中数据集中包含三类鸢尾花数据，鸢尾花数据集是常用的分类实验数据集。

```
In[3]    Iris = datasets.load_iris()#导入数据集
In[4]    data = Iris.data#获取样本特征向量
In[5]    target = Iris.target#获得样本label
In[6]    print (data,target)
```

应用

- **Scikit-learn快速使用**
  - **1.获取数据：**
    - Iris数据集导入：
      - 数据集中的样本的特征向量以及样本标签的存储格式均为矩阵。

```
In[7]     print (type(data),type(target))
Out[7]    <class 'numpy.ndarray'>
          <class 'numpy.ndarray'>
```

      - 其中包含三种鸢尾花，分别是山鸢尾（setossa），变色鸢尾（versicolor）和维吉尼亚鸢尾（virginica）。

```
In[8]     print(Iris.target_names)
Out[8]    ['setosa' 'versicolor' 'virginica']
```

应用

- **Scikit-learn快速使用**
  - **1.获取数据：**
    - Iris数据集导入：
      - Iris每个样本包含4个特征：花萼长度，花萼宽度，花瓣长度，花瓣宽度。

```
In[9]     print(Iris.feature_names)
Out[9]    ['sepal length (cm)', 'sepal width (cm)',
           'petal length (cm)', 'petal width (cm)']
```

      - 每类包含50个样本，共150个样本。该数据集包含4个特征向量，1个类别向量（label）。

```
In[10]    print(Iris.data.shape,Iris.target.shape)
Out[10]   (150, 4) (150,)
```

应用

- Scikit-learn快速使用
  - 1.获取数据：
    - Boston数据集导入：
      - 波士顿房价数据集是常用来做回归分析的数据集，该数据集来源于1978年美国某经济学杂志上。该数据集包含若干波士顿房屋的价格及其各项数据，每个数据项包含14个数据，分别是住宅房间数目及周边犯罪率、是否在河边等13个房价相关信息，和最后一个房屋均价数据。
      - Sklearn自带有boston数据集，它包含506个样本、13个特征和一个实数表示的目标值。

```
In[11]    Boston = datasets.load_boston()
In[12]    print(Boston.data,Boston.target)
```

应用

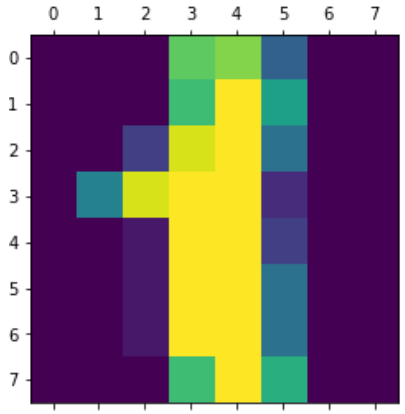
- Scikit-learn快速使用
  - 1.获取数据：
    - 手写数字数据集：
      - 手写数字数据集包含1797个0-9的手写数字数据，每个数据由8 * 8 大小的矩阵构成，矩阵中值的范围是0-16，代表颜色的深度。

```
In[13]    digits = datasets.load_digits()#导入数据集
In[14]    print(digits.data.shape)
In[15]    print(digits.target.shape)
In[16]    print(digits.images.shape)
```

应用

- Scikit-learn快速使用
  - 1.获取数据：
    - 手写数字数据集：
      - 使用matplotlib绘制一个手写数字数据。

```
In[17]    import matplotlib.pyplot as plt
In[18]    plt.matshow(digits.images[1])
In[19]    plt.show()
In[20]    print(digits.target[1])
```



应用

- Scikit-learn快速使用
  - 1.获取数据：
    - 创建数据集：可以利用Scikit-learn创建各种类型的人工数据集。

datasets.make_blobs ([n_samples, n_features, ...])	datasets.make_gaussian_quantiles ([mean, ...])
datasets.make_classification ([n_samples, ...])	datasets.make_hastie_10_2 ([n_samples, ...])
datasets.make_circles ([n_samples, shuffle, ...])	datasets.make_low_rank_matrix ([n_samples, ...])
datasets.make_friedman1 ([n_samples, ...])	datasets.make_moons ([n_samples, shuffle, ...])
datasets.make_friedman2 ([n_samples, noise, ...])	datasets.make_multilabel_classification ([...])
datasets.make_friedman3 ([n_samples, noise, ...])	datasets.make_regression ([n_samples, ...])
	datasets.make_s_curve ([n_samples, noise, ...])

应用

- Scikit-learn快速使用
  - 1.获取数据：
    - 创建数据集：回归问题样本生成器

sklearn.datasets. **make_regression** (*n_samples=100, n_features=100, n_informative=10, n_targets=1, bias=0.0, effective_rank=None, tail_strength=0.5, noise=0.0, shuffle=True, coef=False, random_state=None*) [source]

    - 生成一个包含300个样本、10个特征和一个实数表示的目标值的数据集。

```
In[21]    from sklearn.datasets import make_regression
In[22]    X,y=make_regression(n_samples=300,n_features=10
          ,n_targets=1)
```

应用

- Scikit-learn快速使用
  - 1.获取数据：
    - 创建数据集：分类问题样本生成器

sklearn.datasets. **make_classification** (*n_samples=100, n_features=20, n_informative=2, n_redundant=2, n_repeated=0, n_classes=2, n_clusters_per_class=2, weights=None, flip_y=0.01, class_sep=1.0, hypercube=True, shift=0.0, scale=1.0, shuffle=True, random_state=None*) ¶ [source]

    - 生成一个包含300个样本、5个特征的二分类数据集。

```
In[23]    from sklearn.datasets import make_classification
In[24]    X, y = make_classification(n_samples=300,
          n_features=5, n_classes=2)
```

应用

• Scikit-learn快速使用

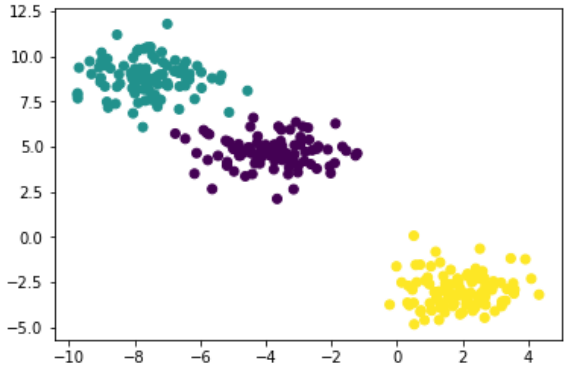
• 1.获取数据：

- 创建数据集：聚类问题样本生成器

sklearn.datasets. **make_blobs** (*n_samples=100, n_features=2, centers=None, cluster_std=1.0, center_box=(-10.0, 10.0), shuffle=True, random_state=None*) [source]

- 生成一个包含300个样本、2个特征的聚类数据集并可视化

```
In[25]    from sklearn.datasets import make_blobs
In[26]    X,y = make_blobs(n_samples=300,
          n_features=2,centers=3)
In[27]    plt.scatter(X[:,0],X[:,1],c = y)
In[28]    plt.show()
```



应用

• Scikit-learn快速使用

• 2.数据预处理：

```
In[29]    from sklearn import preprocessing
In[30]    dir(preprocessing)
```

官方文档：https://scikit-learn.org/stable/modules/preprocessing.html

- 标准化

sklearn.preprocessing. **scale** (*X, axis=0, with_mean=True, with_std=True, copy=True*) ¶ [source]

class sklearn.preprocessing. **StandardScaler** (*copy=True, with_mean=True, with_std=True*) ¶ [source]

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

class sklearn.preprocessing. **MaxAbsScaler** (*copy=True*) [source]

应用

• Scikit-learn快速使用

• 2.数据预处理：

- 非线性变换

class sklearn.preprocessing. **QuantileTransformer** (*n_quantiles=1000, output_distribution='uniform', ignore_implicit_zeros=False, subsample=100000, random_state=None, copy=True*) ¶ [source]

sklearn.preprocessing. **quantile_transform** (*X, axis=0, n_quantiles=1000, output_distribution='uniform', ignore_implicit_zeros=False, subsample=100000, random_state=None, copy='warn'*) [source]

class sklearn.preprocessing. **QuantileTransformer** (*n_quantiles=1000, output_distribution='uniform', ignore_implicit_zeros=False, subsample=100000, random_state=None, copy=True*) [source]

- 归一化

sklearn.preprocessing. **normalize** (*X, norm='l2', axis=1, copy=True, return_norm=False*) [source]

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

应用

- **Scikit-learn快速使用**
  - **数据预处理：**
  - 二值化

```
class sklearn.preprocessing. Binarizer (threshold=0.0, copy=True)                [source]
```

  - One-hot编码

```
class sklearn.preprocessing. OneHotEncoder (n_values=None, categorical_features=None, categories=None,
drop=None, sparse=True, dtype=<class 'numpy.float64'>, handle_unknown='error')   [source]
```

应用

- Scikit-learn快速使用
  - 3.数据集拆分：
    - 得到训练数据集后，通常我们会把训练数据进一步拆分成训练集和验证集，这样有助于模型参数的选取。train_test_split是交叉验证中常用的函数，功能是从样本中随机的按比例选取train data和test data，形式为：

```
In[31]    from sklearn.model_selection import
          train_test_split
In[32]    X_train,X_test, y_train, y_test
          =train_test_split(data,target,test_size=0.4,
          random_state=0)
In[33]    print(X_train.shape,X_test.shape)
```

应用

- Scikit-learn快速使用
  - 4.定义模型：
    - 常用模型：

```
In[34]  from sklearn.linear_model import LinearRegression#线性回归
In[35]  from sklearn.linear_model import LogisticRegression#逻辑回归
In[36]  from sklearn import naive_bayes#朴素贝叶斯算法NB
In[37]  from sklearn import tree #决策树算法
In[38]  from sklearn.svm import SVC#支持向量机算法
In[39]  from sklearn import neighbors#K近邻算法
In[40]  from sklearn import neural_network#神经网络
```

应用

- **Scikit-learn快速使用**
  - **5.模型预测与评估：**
    - sklearn为所有模型提供了非常相似的接口，这样使得我们可以更加快速的熟悉所有模型的用法，我们以支持向量机为例先来看看模型的常用属性和功能。

```
In[41]    from sklearn.svm import SVC
In[42]    model = SVC()
In[43]    model.fit(X_train, y_train)#拟合模型
In[44]    y_pre = model.predict(X_test)#预测模型
In[45]    print(model.get_params())#获得模型参数
In[46]    print(model.score(X_test,y_test))#模型得分
In[47]    print(y_pre)#模型预测结果
```

应用

- Scikit-learn快速使用
  - 5.模型预测与评估：
    - 交叉验证：
      - cv是我们使用的交叉验证的生成器或者迭代器，它决定了交叉验证的数据是如何划分的，当cv的取值为整数的时候，使用K折交叉验证方法。

```
In[48]    from sklearn.model_selection import
          cross_val_score
In[49]    scores = cross_val_score(model,data,target, cv=5)
In[50]    print(scores)
In[51]    print(scores.mean())
```

应用

- **Scikit-learn快速使用**
  - **5.模型保存：**
    - Pickle方法：

```
In[52]    import pickle
In[53]    with open('model.pickle', 'wb') as f:
              pickle.dump(model, f)
In[54]    with open('model.pickle', 'rb') as f:
              model = pickle.load(f)
In[55]    model.predict(X_test)
```

    - Joblib方法：

```
In[56]    import joblib
In[57]    joblib.dump(model, 'model.pickle')
In[58]    model = joblib.load('model.pickle')
```

应用

- 说出下面任务的T，P，E。T代表**任务**(task),P代表任务T的**性能**(performance),E代表**经验**(experience)
  - 预测学生是否能够考上研究生
  - 预测下节课有多少学生旷课
  - 学生根据兴趣聚成几个社团
- 课堂讨论
  - 列举身边采用机器学习方法解决实际问题的例子
  - 三人一组讨论，讲明问题的T, P, E

第11章
机器学习方法

1 机器学习的基础

2 Sklearn库的基本使用