Scikit-learn机器学习包

本节代码来自: 黄海广-机器学习 https://github.com/fengdu78/WZU-machine-learning-course 推荐自学

Scikit-learn是基于NumPy、SciPy和Matplotlib的开源Python机器学习包，它封装了一系列数据预处理、机器学习算法、模型选择等工具，是数据分析师首选的机器学习工具包。

自2007年发布以来，Scikit-learn已经成为Python重要的机器学习库了，Scikit-learn简称sklearn，支持包括分类、回归、降维和聚类四大机器学习算法，还包括了特征提取、数据处理和模型评估三大模块。

1. Scikit-learn概述

Scikit-Learn (简称 Sklearn) 是基于 Python 语言的机器学习工具。它建立在 NumPy、 SciPy、Pandas和 Matplotlib 之上，里面的 API 的设计非常好，所有对象的接口简单，很适合新手上路。

Scikit-Learn库的算法主要有四类：分类、回归、聚类、降维。其中：

1. 常用的回归：线性回归、决策树回归、SVM回归、KNN 回归；集成回归：随机森林、Adaboost、GradientBoosting、Bagging、ExtraTrees。
2. 常用的分类：线性分类、决策树、SVM、KNN，朴素贝叶斯；集成分类：随机森林、Adaboost、GradientBoosting、Bagging、ExtraTrees。
3. 常用聚类：K均值(K-means)、层次聚类(Hierarchical clustering)、DBSCAN。
4. 常用降维：LinearDiscriminantAnalysis、PCA。

上图代表了Scikit-Learn算法选择的一个简单路径，这个路径图代表：蓝色圆圈是判断条件，绿色方框是可以选择的算法，我们可以根据自己的数据特征和任务目标去找一条自己的操作路线。

Scikit-Learn中包含众多数据预处理和特征工程相关的模块，但其实Sklearn六大板块中有两块都是关于数据预处理和特征工程的，两个板块互相交互，为建模之前的全部工程打下基础。

2.Scikit-learn主要用法

2.1.基本建模流程

基本建模的符号标记见表：

符号	代表含义	符号	代表含义
X_train	训练数据	y_train	训练集标签
X_test	测试数据	y_test	测试集标签
X	完整数据	y	数据标签
		y_pred	预测标签

2.1.1.导入工具包

导入工具包的方法如下(这里使用伪代码)：

```
from sklearn import 包名称
```

```
from sklearn.库名称 import 包名称
```

代码示例：

In [1]:
```
from sklearn import datasets, preprocessing
#导入数据集，数据预处理库
from sklearn.model_selection import train_test_split
#从模型选择库导入数据切分包
from sklearn.linear_model import LinearRegression
#从线性模型库导入线性回归包
from sklearn.metrics import r2_score
#从评价指标库导入R2评价指标
```

2.1.2 导入数据

导入数据的方法如下：

```
from sklearn.datasets import 数据名称
```

Scikit-learn支持以NumPy的arrays对象、Pandas对象、SciPy的稀疏矩阵及其他可转换为数值型arrays的数据结构作为其输入，前提是数据必须是数值型的。

sklearn.datasets模块提供了一系列加载和获取著名数据集如鸢尾花、波士顿房价、Olivetti人脸、MNIST数据集等的工具，也包括了一些toy data如S型数据等的生成工具。

Scikit-learn内置了很多可以用于机器学习的数据，可以用两行代码就可以使用这些数据。内置数据分为可以直接使用的数据集、需下载的数据集以及生成数据集。

- 1.可以直接使用的自带数据集

此类数据集可以直接导入使用数据，数据集和描述见下表：

数据集名称	描述	类型	维度
load_boston	Boston房屋价格	回归	506*13
fetch_california_housing	加州住房	回归	20640*9
load_diabetes	糖尿病	回归	442*10
load_digits	手写字	分类	1797*64
load_breast_cancer	乳腺癌	分类、聚类	(357+212)*30
load_iris	鸢尾花	分类、聚类	(50*3)*4
load_wine	葡萄酒	分类	(59+71+48)*13
load_linnerud	体能训练	多分类	20

- 2.需要下载的自带数据集

此类数据集第一次使用，需要联网下载数据，数据集和描述见下表：

数据集名称	描述
fetch_20newsgroups	用于文本分类、文本挖据和信息检索研究的国际标准数据集之一。数据集收集了大约20,000左右的新闻组文档，均匀分为20个不同主题的新闻组集合。返回一个可以被文本特征提取器

- 3.生成数据集

此类数据集可以用来分类任务，可以用来回归任务，可以用来聚类任务，用于流形学习的，用于因子分解任务的，用于分类任务和聚类任务的：这些函数产生样本特征向量矩阵以及对应的类别标签集合，数据集和描述见下表：

数据集名称	描述
make_blobs	多类单标签数据集，为每个类分配一个或多个正态分布的点集
make_classification	多类单标签数据集，为每个类分配一个或多个正态分布的点集，提供了为数据添加噪声的方式，包括维度相关性，无效特征以及冗余特征等
make_circle和make_moons	产生二维二元分类数据集来测试某些算法的性能，可以为数据集添加噪声，可以为二元分类器产生一些球形判决界面的数据

代码示例：

In [2]:
```
#导入内置的鸢尾花数据
from sklearn.datasets import load_iris

iris = load_iris()
#定义数据、标签
X = iris.data
y = iris.target
```

2.2.数据预处理

2.2.1.数据划分

机器学习的数据，可以划分为训练集、验证集和测试集，也可以划分为训练集和测试集。

代码示例：

In [3]:
```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X,
                                                    y,
                                                    random_state=2023,
                                                    test_size=0.3)
#将完整数据集的70%作为训练集，30%作为测试集，
```

```
#并使得测试集和训练集中各类别数据的比例与原始数据集比例一致(stratify分层策略)，另外可通过设置shuffle=True 提前打乱数据。

X_train[:3]
```

Out[3]:
```
array([[6.2, 2.8, 4.8, 1.8],
       [5.2, 2.7, 3.9, 1.4],
       [6.6, 3. , 4.4, 1.4]])
```

2.2.2.数据变换操作

sklearn.preprocessing模块包含了数据变换的主要操作（表3-5），数据变换的方法如下：

`from sklearn.preprocessing import库名称`

下表使用Scikit-learn进行数据变换

预处理操作	库名称
标准化	StandardScaler
最小最大标准化	MinMaxScaler
One-Hot编码	OneHotEncoder
归一化	Normalizer
二值化(单个特征转换)	Binarizer
标签编码	LabelEncoder
缺失值填补	Imputer
多项式特征生成	PolynomialFeatures

代码示例：

In [4]:
```
#使用Scikit-Learn进行数据标准化

from sklearn.preprocessing import StandardScaler

#构建转换器实例

scaler = StandardScaler( )
```

```
#拟合及转换

scaler.fit_transform(X_train)[:3]
```

Out[4]:
```
array([[ 0.39430793, -0.6308404 ,  0.5316451 ,  0.70772787],
       [-0.82340773, -0.86996762,  0.01635831,  0.18624418],
       [ 0.88139419, -0.15258594,  0.30262875,  0.18624418]])
```

2.2.3.特征选择

关于特征选择等：在EXP2.4-特征工程进行细致讨论。

特征选择的方法如下：

- 导入特征选择库

```
from sklearn import feature_selection as fs
```

- 过滤式(Filter): 保留得分排名前k的特征(top k方式)

```
fs.SelectKBest(score_func, k)
```

- 交叉验证特征选择

```
fs.RFECV(estimator, scoring="r2")
```

- 封装式(Wrapper)，结合交叉验证的递归特征消除法，自动选择最优特征个数：

```
fs.SelectFromModel(estimator)
```

- 嵌入式(Embedded)，从模型中自动选择特征，任何具有coef_或者feature_importances_的基模型都可以作为estimator参数传入。

2.3监督学习算法

2.3.1.监督学习算法-回归

表常见的回归模型

回归模型名称	库名称

回归模型名称	库名称
线性回归	LinearRegression
岭回归	Ridge
LASSO回归	LASSO
ElasticNet回归	ElasticNet
决策树回归	tree.DecisionTreeRegressor

代码示例：

In [5]:
```
#从线性模型库导入线性回归模型
from sklearn.linear_model import LinearRegression
# 构建模型实例
lr = LinearRegression()
# 训练模型
lr.fit(X_train, y_train)
# 作出预测
y_pred = lr.predict(X_test)
y_pred[:3]
```

Out[5]:
```
array([1.92581496, 1.11877566, 1.16609402])
```

2.3.2.监督学习算法-分类

表常见的分类模型

模型名称	库名称
逻辑回归	linear model.LogisticRearession
支持向量机	svm.SVC
朴素贝叶斯	naïve_bayes.GaussianNB
KNN	neighbors.NearestNeighbors
随机森林	ensemble.RandomForestClassifier
GBDT	ensemble.GradientBoostingClassifier

代码示例：

In [6]:
```
#从树模型库导入决策树
from sklearn.tree import DecisionTreeClassifier
#定义模型
clf = DecisionTreeClassifier(max_depth=5)
#训练模型
clf.fit(X_train, y_train)
#使用决策树分类算法解决二分类问题，得到的是类别
y_pred = clf.predict(X_test)
#y_prob 为每个样本预测为"0"和"1"类的概率
y_prob = clf.predict_proba(X_test)
y_prob[:3],y_test[:3]
```

Out[6]:
```
(array([[0., 0., 1.],
        [0., 1., 0.],
        [0., 1., 0.]]),
 array([2, 1, 1]))
```

2.4.无监督学习算法

2.4.1.聚类算法

sklearn.cluster模块包含了一系列无监督聚类算法，聚类使用的方法如下：

from sklearn.cluster import库名称

表常见的聚类模型

模型名称	库名称
K-means	KMeans
DBSCAN	DBSCAN
层次聚类	AgglomerativeClustering
谱聚类	SpectralClustering

代码示例：

In [7]:
```
#从聚类模型库导入kmeans
from sklearn.cluster import KMeans
#构建聚类实例
kmeans = KMeans(n_clusters=3, random_state=0)
#拟合
kmeans.fit(X_train)
#预测
kmeans.predict(X_test)[:3],y_test[:3]
```

```
D:\Python\python3.8\lib\site-packages\sklearn\cluster\_kmeans.py:870: FutureWarning: The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the value of `n_init` explicitly to suppress the warning
  warnings.warn(
```

Out[7]:
```
(array([2, 0, 0]), array([2, 1, 1]))
```

2.4.2.降维算法

Scikit-learn中降维算法都被包括在模块decomposition中，sklearn.decomposition模块本质是一个矩阵分解模块。最常见的降维方法是PCA(主成分分析)。

降维的使用的方法如下：

`from sklearn.decomposition import 库名称`

代码示例：

In [8]:
```
#导入PCA库
from sklearn.decomposition import PCA
#设置主成分数量为3，n_components代表主成分数量
pca = PCA(n_components=3)
#训练模型
pca.fit(X)
#投影后各个特征维度的方差比例(这里是三个主成分)
print(pca.explained_variance_ratio_)
#投影后的特征维度的方差
print(pca.explained_variance_)
```

```
[0.92461872 0.05306648 0.01710261]
[4.22824171 0.24267075 0.0782095 ]
```

2.5.评价指标

sklearn.metrics模块包含了一系列用于评价模型的评分函数、损失函数以及成对数据的距离度量函数。评价指标主要分为分类评价指标、回归评价指标等等，这里列举了常见的几种评价指标。

评价指标使用的方法如下：

```
from sklearn.metrics import 库名称
```

评价指标	库名称	使用范围
正确率	accuracy_score	分类
精确率	precision_score	分类
F1 值	f1_score	分类
对数损失	log_loss	分类
混淆矩阵	confusion_matrix	分类
含多种评价的分类报告	classification_report	分类
均方误差MSE	mean_squared_error	回归
平均绝对误差MAE	mean_absolute_error	回归
决定系数R2	r2_score	回归

代码示例：

In [9]:
```
#从评价指标库导入准确率
from sklearn.metrics import accuracy_score
#从树模型库导入决策树
from sklearn.tree import DecisionTreeClassifier
#定义模型
clf = DecisionTreeClassifier(max_depth=5)
#训练模型
clf.fit(X_train, y_train)
#使用决策树分类算法解决二分类问题，得到的是类别
y_pred = clf.predict(X_test)
#y_prob 为每个样本预测为“0”和“1”类的概率
y_prob = clf.predict_proba(X_test)
y_prob[:3],y_test[:3]

#计算样本的准确率
```

```
accuracy_score(y_test, y_pred)
#对于测试集而言，大部分函数都必须包含真实值y_test和预测值y_pred
```

Out[9]: 0.9555555555555556

2.6.交叉验证及超参数调优

2.6.1.交叉验证

交叉验证的方法如图，具体原理将在第7章"机器学习实践"中讲解，本章仅讲解使用方法。

代码示例：

In [10]:
```
#从模型选择库导入交叉验证分数
from sklearn.model_selection import cross_val_score
clf = DecisionTreeClassifier(max_depth=5)
#使用5折交叉验证对决策树模型进行评估，使用的评分函数为F1值
scores = cross_val_score(clf, X_train, y_train,cv=5, scoring='f1_weighted')
scores
```

Out[10]: array([0.90391156, 1. , 0.9047619 , 1. , 0.9047619])

此外，Scikit-learn提供了部分带交叉验证功能的模型类如 `LogisticRegressionCV` 、 `LassoCV` 、等，这些类包含CV参数。

2.6.2.超参数调优

在机器学习中，超参数是指无法从数据中学习而需要在训练前提供的参数。机器学习模型的性能在很大程度上依赖于寻找最佳超参数集。

超参数调整一般是指调整模型的超参数，这基本上是一个非常耗时的过程。目前主要有 3 种最流行的超参数调整技术：网格搜索、随机搜索和贝叶斯搜索，其中Scikit-learn内置了网格搜索、随机搜索，本章进行简单讲解，其余调参方法如贝叶斯搜索，本章不进行讨论。

- 1.超参数调优——网格搜索

代码示例：

In [11]:
```
#从模型选择库导入网格搜索
from sklearn.model_selection import GridSearchCV
from sklearn import svm

svc = svm.SVC()
#把超参数集合作为字典
params = {'kernel': ['linear', 'rbf'], 'C': [1,5, 10]}
#进行网格搜索，使用了支持向量机分类器，并进行五折交叉验证
grid_search = GridSearchCV(svc, params, cv=5)
#模型训练
grid_search.fit(X_train, y_train)
#获取模型最优超参数组合
grid_search.best_params_
```

Out[11]:
```
{'C': 1, 'kernel': 'linear'}
```

在参数网格上进行穷举搜索，方法简单但是搜索速度慢(超参数较多时)，且不容易找到参数空间中的局部最优。

- 2.超参数调优——随机搜索

代码示例：

In [12]:
```
#从模型选择库导入随机搜索
from sklearn.model_selection import RandomizedSearchCV
from scipy.stats import randint

svc = svm.SVC()
#把超参数组合作为字典
param_dist = {'kernel': ['linear', 'rbf'], 'C': randint(1, 20)}
#进行随机搜索
random_search = RandomizedSearchCV(svc, param_dist, n_iter=10)
#模型训练
random_search.fit(X_train, y_train)
#获取最优超参数组合
random_search.best_params_
```

Out[12]:
```
{'C': 1, 'kernel': 'linear'}
```

在参数子空间中进行随机搜索，选取空间中的100个点进行建模(可从scipy.stats常见分布如正态分布norm、均匀分布uniform中随机采样得到)，时间耗费较少，更容易找到局部最优。

3.Scikit-learn总结

Scikit-learn是基于 Python 语言的机器学习工具，它建立在 NumPy、SciPy、Pandas和Matplotlib之上，被广泛地用于统计分析和机器学习建模等数据科学领域，其主要优点包括：

- 建模方便：用户通过Scikit-learn能够实现各种监督和非监督学习的模型，仅仅需要几行代码就可以实现。
- 功能多样：使用Scikit-learn还能够进行数据的预处理、特征工程、数据集切分、模型评估等工作。
- 数据丰富：内置丰富的数据集，比如：泰坦尼克、鸢尾花等，还可以生成数据，非常方便。

参考文献

[1] PEDREGOSA F, VAROQUAUX G, GRAMFORT A, et al. Scikit-learn: Machine Learning in Python[J]. Journal of Machine Learning Research, 2011, 12: 2825–2830.

[2] Andrew Ng. Machine Learning[EB/OL]. StanfordUniversity,2014.https://www.coursera.org/course/ml