1.Scikit-learn入门

# 1 实验目的

1.了解机器学习的基础知识、常用术语；

2.学会搭建机器学习环境；

3.了解Scikit-learn算法库；

4.通过一个简单的线性回归实验，了解利用sklearn算法库进行机器学习的基本流程。

# 2 实验要求

本章简要介绍机器学习，以及numpy、matplotlib、sklearn等模块的常用算法。然后以一个简单的线性回归实验为例，帮助同学们熟悉用Scikit-learn库进行机器学习实验的基本流程。

对于本章的实验内容，同学们只需要大致了解即可，sklearn库中机器学习算法的细节会在之后的章节中具体讲解。

# 3实验原理

## 3.1 机器学习简介

### 3.1.1 什么是机器学习？

机器学习相比于传统的编程算法，是一种让计算机获得学习能力的方法，能够解决仅用传统方法编程无法直接实现的功能。能够让计算机自动地处理海量的数据，从复杂的数据中挖掘出有价值的信息。

人工智能、机器学习、深度神经网络其实是一个单向的包含关系，人工智能是计算机科学的一个分支，而机器学习又是人工智能中一个重要的分支，现在很流行的深度学习（深度神经网络）又是机器学习的一个分支。

### 3.1.2 机器学习算法分类

机器学习发展至今已有很多成熟且优秀的算法，可以对它们按学习时是否需要人类监督进行大致分类。

（1）**有监督学习**是需要在人类监督下进行学习的算法，要给算法提供标注好的数据集。

有监督学习常用来解决分类问题（根据已有的训练数据，学会如何对新数据样本进行分类）、回归问题（根据一组已知最终结果的样本，来预测新样本的结果）等。

常见的有监督学习有：K-邻近算法、线性回归、逻辑回归、支持向量机、决策树和随机森林、有监督的神经网络。

（2）**无监督学习**是在没有人类监督的情况下学习，所有的训练数据都是没有被标注的。

无监督学习常用来解决聚类问题（将数据自动分为n类，与分类问题很像但是训练用的数据都是未经标注的）、降维（在尽可能不丢失信息的前提下简化数据维数，即提取特征，减少用于训练的属性个数）、异常检测（如自动判断数据集的缺陷）等。

常见的无监督学习有：K-means算法、分层聚类算法、主成分分析等。

（3）**半监督学习**就是无监督学习和有监督学习的结合，通过大量未标注数据和少量的人类监督来学习。

比如智能手机上的相册功能就是典型的半监督学习。当用户在手机相册中保存了大量亲朋好友的照片后，手机相册会自动根据未标注的照片通过聚类算法按人像分成不同类别，把不同人的照片归在不同类别里，之后只需要用户对每个类标注人名，进行一些简单的合并修改，就实现了照片按人分类。

### 3.1.3 机器学习主要挑战

（1）数据不足或质量差

大部分机器学习算法都是通过大量的数据训练得来，在某些复杂问题上，好的数据集有时比算法更重要。可能会遇到的问题有：训练数据不足、数据集标注困难、数据不具有代表性、数据中异常值和噪声过多、无关特征过多等。

（2）过拟合与欠拟合

拟合在数学上是指把散落在坐标系中的点，用一条光滑的曲线连接起来。这种拟合曲线有无数种可能，从而有各种拟合方法。图1即为一个线性拟合的示例。

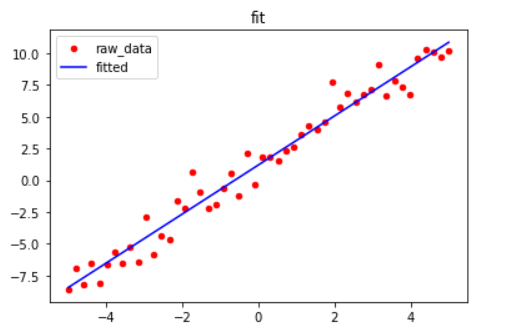


图1 线性拟合示例

过拟合是机器学习训练中常遇到的问题，算法过度利用训练数据的同时，也把数据中的噪声充分考虑了进去。这样的模型在训练集上可能表现优越，但是泛化能力弱，在未知的新数据上往往表现不佳。

图2给出了一个过拟合的例子，虽然曲线对训练数据拟合程度很高，但是并不能代表数据的真实的情况，无法进行准确的预测。

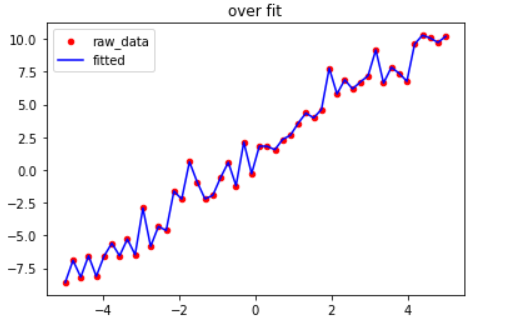


图2 过拟合示例

欠拟合与过拟合相反，指拟合曲线不能很好地拟合数据，拟合结果在训练集和测试集上效果均不理想，可能是由于选用了不合适的模型或学习不完善导致。图3给出了欠拟合的示例。

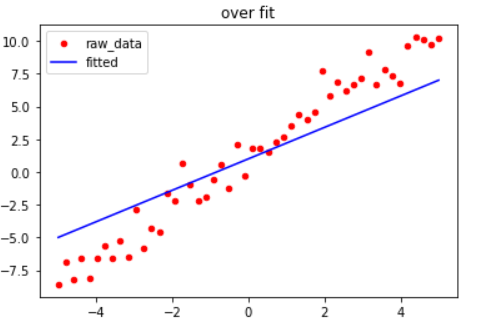


图3 欠拟合示例

## 3.2 Scikit-learn简介

Scikit-learn主要用 Python 语言开发，在 NumPy、Scipy 与 Matplotlib 的基础上，它提供了数十种经典的机器学习算法接口（API）供用户调用。提供了分类器、聚类器、回归器等估算器（estimator），每个估算器都是通过fit()方法训练数据，用transform()方法转换数据。如果是分类器、回归器等，还可以用predict()方法进行预测，用score()方法查看算法的评分。

Scikit-learn极大地提高了机器学习的效率，让研究者能够真正地“站在巨人的肩膀上”更进一步。也让开发者无须在意数学层面的公式、计算过程，有更多的时间与精力专注于业务层面，解决实际的应用问题。

# 4 实验步骤

## 4.1 搭建机器学习环境

Numpy是python的一个第三方扩展程序包，提供了多维数组处理、线性代数、傅里叶变换、随机数生成等非常有用的数学工具，主要用它来进行数据处理。

Matplotlib是 Python 的绘图库，它能让使用者很轻松地将数据图形化，并且提供多样化的输出格式。

Scikit-learn前文已经介绍过，安装命令如下：

pip3 install numpy matplotlib scikit-learn -i https://pypi.tuna.tsinghua.edu.cn/simple

## 4.2 简单线性回归实验

### 4.2.1 导入依赖

Numpy用于数据处理，Matplotlib用于数据可视化。sklearn.linear\_model是sklearn中线性模型算法模块，从中导入普通线性回归算法LinearRegression类。sklearn.model\_selection 中的train\_test\_split方法用于划分数据集。

# 导入numpy用于数据处理

import numpy as np

# 导入matplotlib库用于数据可视化

import matplotlib.pyplot as plt

# 导入线性模型中的线性回归算法

from sklearn.linear\_model import LinearRegression

# 导入划分数据集方法

from sklearn.model\_selection import train\_test\_split

### 4.2.2 生成实验数据

（1）生成实验数据

生成200个0到10之间的随机数数组X，计算数组，并给Y加上一个服从正态分布的扰动作为数据的标签。

scikit-learn中线性模型算法不接受一维数组作为训练数据，因为它无法区分n个元素的一维数组是1个有n个特征的样本，还是n个只有1个特征的样本。所以把X和Y改为二维上的列向量，表示x特征有200个样本。

# 生成100个[0， 10]的随机数，改为列向量

X = np.random.random(200) \* 20 - 10

Y = 1.5 \* X - 1.2 + np.random.randn(200)

# 改为scikit-learn库中算法可以直接接受的二维上的列向量

data = X.reshape(-1, 1)

target = Y.reshape(-1, 1)

（2）数据划分

机器学习实验中，一般用划分训练集和测试集或交叉验证的方式对模型进行评估，在训练集上训练模型，在测试集上检验模型的泛化能力。

sklearn.model\_selection模块中提供了train\_teat\_split()方法进行数据划分，可以很方便地将数据集随机划分为训练集和测试集。其重要参数及含义如下：

**arrays：**传入待划分的数据，可以是列表、numpy数组、稀疏矩阵或是pandas中的DataFrame结构数据。此参数可传入多个，会将这多个数据按同样的方式划分。

**test\_size：**设置测试集大小（小数为测试集占比，整数为测试集中样本个数），也可以用**train\_size**参数设置训练集大小；

**random\_state：**固定划分的随机性，如果设置了相同的random\_state，则可以得到同一组随机数，从而得到同样的划分结果；

**shuffle：**是设置切分前是否对数据进行打乱。默认shuffle为True，即默认切分前打乱数据。

划分结束后函数会返回划分结果，划分训练集、测试集的代码如下：

# 划分训练集和测试集

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(

data,

target,

test\_size=0.2,

random\_state=1

)

### 4.2.3 创建并训练模型

Scikit-learn能够根据回归数据集进行拟合的学习器为回归器，回归器都可以通过fit()方法接受两个数据集作为参数（一个是数据，另一个是标签）进行训练。

模型训练完成后，算法模型实例的成员变量可以用参数名加下划线（形如：.param\_）直接访问。例如，本实验可以通过model.intercept\_检查训练出的截距，通过model.coef\_检查训练出的权值。

# 创建并训练

model = LinearRegression()

model.fit(X\_train, Y\_train)

# 查看训练出的参数信息

print("训练出的截距为：%.4f" % model.intercept\_)

print("训练出的权值为：%.4f\n" % model.coef\_)

### 4.2.4 测试模型

回归器能够基于给定的数据集进行预测，它的predict()方法被称为预测器，可以通过predict()方法对新样本进行预测并返回预测结果。

还可以用评估器score()方法来衡量训练出的模型在测试集上的预测效果。

# 预测

Y\_predict = model.predict(X\_test)

# 模型评分

print("训练出的模型在训练集上得分为：%.6f" % model.score(X\_train, Y\_train))

print("训练出的模型在测试集上得分为：%.6f\n" % model.score(X\_test, Y\_test))

### 4.2.5 数据可视化

将数据可视化化有助于让我们更直观地观察出算法模型的好坏，发现更多有价值的信息。

利用matplotlib模块进行数据可视化，title()方法用于设置图像标题，scatter()方法用于画散点图，plot()方法用于画折线图，legend()方法用于制定标签样式，最后调用show()方法展示图片。

# 预测结果图像

plt.title("Predict Result")

plt.scatter(X\_test, Y\_test, label="Test Data", s=30, c="r")

plt.plot(X\_test, Y\_predict, label="Fitted", c="b")

plt.legend(loc=0)

plt.show()

一般多数的机器学习问题都是高维空间中的问题，即每个样本可能具有多个特征，难以可视化，可以通过降维算法降至某两到三维来进行可视化观察，但是使用降维会损失很多数据信息。

# 5 实验结果

（1）模型训练出拟合直线的截距、权值如下：



图4 查看训练出的截距和权值

（2）训练出的模型评分如下：



图5 训练出的模型评分

（3）数据可视化

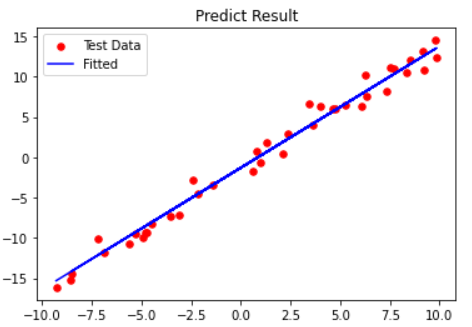


图6 拟合带有扰动的线性函数

红色的点是测试集样本的分布情况，蓝色的线是用普通线性模型拟合出的拟合曲线。可以看到，本实验中训练出的模型拟合效果很好。