案例八：预测鸢尾花的种类

1. 案例目的

• 理解KNN分类算法；

• 理解分类的场景；

• 掌握模型评估方法。

2. 案例内容

本案例将对150条鸢尾花的5个属性（花萼长度、花萼宽度、花瓣长度、花瓣宽度、鸢尾花种类）进行分析，建立分类模型，评价模型性能，然后使用该模型预测鸢尾花的种类。

3. 案例知识点

• Python语言编程；

• numpy和pandas库的使用；

• KNN分类算法；

• 分类报告；

• 使用模型进行预测。

4. 案例时长

共1学时，具体安排如下：

• 数据处理（0.4学时）

• 训练模型（0.2学时）

• 评估模型（0.2学时）

• 测试新数据（0.2学时）

5. 案例实验环境

**•操作系统：**

1）Linux Ubuntu 16.04 64位操作系统

**•软件环境：**

1）Python 3.7

**•开发环境与工具：**

1）Spyder 4

6. 案例分析

本案例主要分为以下4部分：

1）处理数据，为建立分类模型做准备；

2）建立分类模型；

3）评估模型性能；

4）使用训练好的模型对新数据做分类。

7. 案例实验步骤

模型预测鸢尾花的种类，可分为以下4个步骤：

1、数据预处理；

1.1 导入包

1.2 导入数据集

1.3 将数据分成自变量和因变量

1.4 处理类别型字段

1.5 特征缩放

2. 训练模型；

3、评估模型；

3.1 混淆矩阵

3.2 模型分类报告

4、测试新数据。

7.1开发准备

在桌面上新建一个目录，名字叫iris-species。案例中所有的数据集和代码都放在该目录下。

7.1.1 打开Spyder软件

在桌面上点击Spyder图标打开Spyder，并把工作目录切换至iris-species目录下。

7.1.2 获取数据

本案例需要1个数据集，Iris.csv。

点击右侧菜单栏中的【资源下载】按钮，复制提供的URL前缀，如https://xxx.eec-cn.com/。

将URL前缀和数据集地址拼接到一起，如[https://xxx.eec-cn.com/dataSet/systemLib/a2dac8599d85447192830c4a9866c78b.csv](https://xxx.eec-cn.com/dataSet/systemLib/0c3e785811b1437095486e6978bc3836.csv" \t "_blank)。下载数据集，将Iris.csv复制到刚刚创建的iris-species目录下。

7.1.3 新建代码文件

确保Spyder的工作目录为iris-species。使用Spyder新建一个Python文件，命名为iris\_classification.py。

至此，整个案例的目录结构如下图所示：



图1 案例的目录结构

7.1.4 了解数据

Iris.csv存储着150条鸢尾花数据，用于训练模型。

数据集有6个属性，分别为：

• Id：数据编号

• SepalLengthCm：花萼长度（单位厘米）

• SepalWidthCm：花萼宽度（单位厘米）

• PetalLengthCm：花瓣长度（单位厘米）

• PetalWidthCm：花瓣宽度（单位厘米）

• Species：鸢尾花种类，一共有3个种类，分别是山鸢尾（Iris-setosa）、杂色鸢尾（Iris-versicolor）、维吉尼亚鸢尾（Iris-virginica）。

7.2数据预处理

向iris\_classification.py文件中添加代码。

7.2.1 导入包

导入用到的库，代码如下：

【代码7-2-1】iris\_classification.py

# 导入包

import numpy as np

import pandas as pd

from sklearn.preprocessing import LabelEncoder

from sklearn.preprocessing import StandardScaler

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import confusion\_matrix

from sklearn.metrics import classification\_report

执行代码块。

7.2.2 导入数据集

导入数据集，代码如下：

【代码7-2-2】iris\_classification.py

# 导入数据集

dataset = pd.read\_csv('Iris.csv')

执行代码块。

从现在开始，建议打开Spyder的Variable explorer面板，实时查看变量的信息。

dataset变量的信息如图2所示。

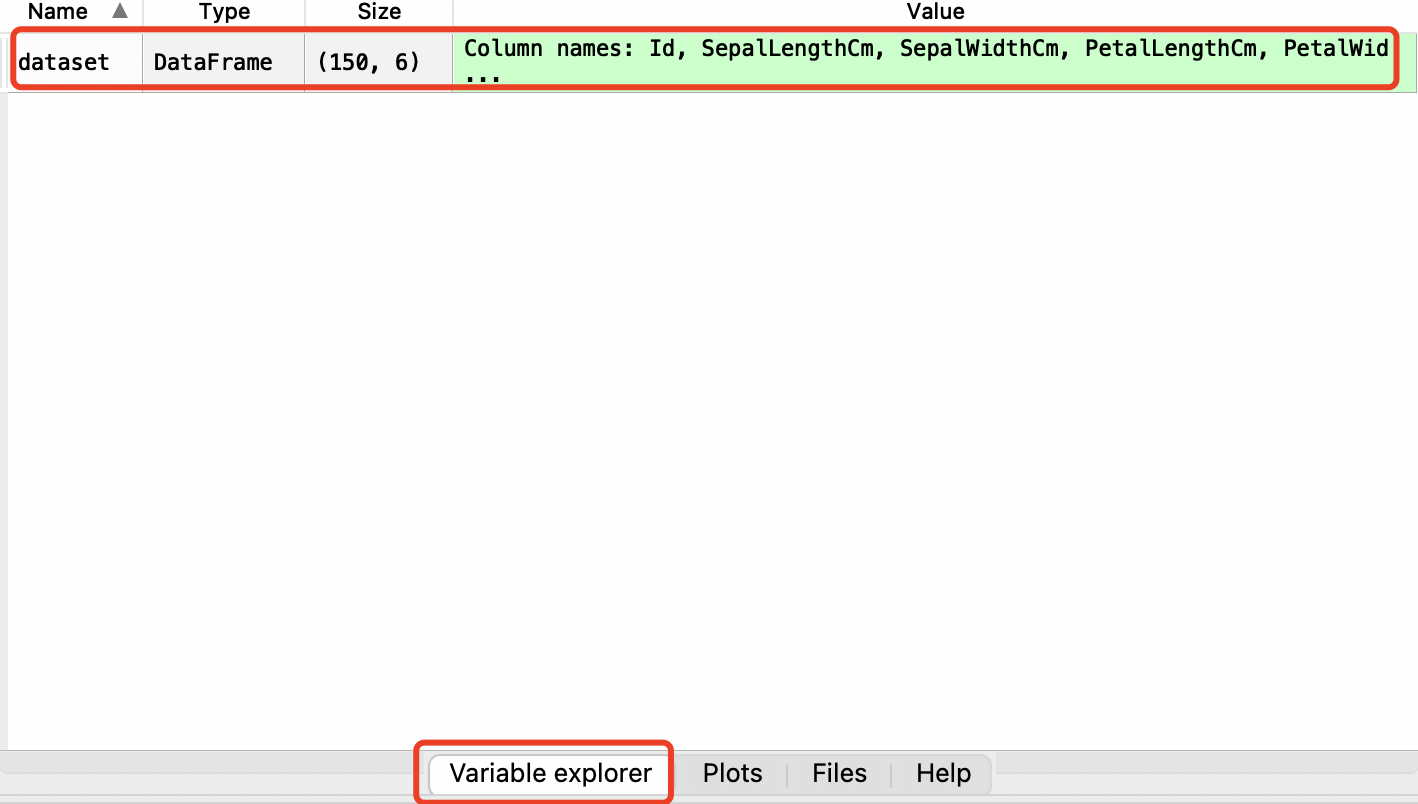


图2 dataset变量的信息

7.2.3 将数据分成自变量和因变量

Id是数据编号，对构建模型无用；

Species是鸢尾花种类，作为因变量。

其余4个字段作为自变量。

【代码7-2-3】iris\_classification.py

# 将数据分成自变量和因变量

X = dataset.iloc[:, 1:5].values

y = dataset.iloc[:, 5].values

执行代码块。

7.2.4 处理类别型字段

在数据集中，鸢尾花的种类都是英文单词。需要转换成数值。

转换之后，y的值Iris-setosa转换为0，'Iris-versicolor转换为1，Iris-virginica转换为2。

【代码7-2-4】iris\_classification.py

# 处理类别型字段

le = LabelEncoder()

y = le.fit\_transform(y)

print(le.classes\_)

执行代码块。

7.2.5 特征缩放

将大部分数据的值转换到-3至3之间。

【代码7-2-5】iris\_classification.py

# 特征缩放

sc = StandardScaler()

X = sc.fit\_transform(X)

执行代码块。

7.3训练模型

使用K近邻训练模型。

KNN（K- Nearest Neighbor）法即K最邻近法，最初由 Cover和Hart于1968年提出，是一个理论上比较成熟的方法，也是最简单的机器学习算法之一。该方法的思路非常简单直观：如果一个样本在特征空间中的K个最相似（即特征空间中最邻近）的样本中的大多数属于某一个类别，则该样本也属于这个类别。该方法在分类决策上只依据最邻近的一个或者几个样本的类别来决定待分样本所属的类别。

【代码7-3】iris\_classification.py

# 训练模型

classifier = KNeighborsClassifier(n\_neighbors = 5, metric = 'minkowski', p = 2)

classifier.fit(X, y)

y\_pred = classifier.predict(X)

执行代码块。

7.4评估模型

7.4.1 混淆矩阵

首先，生成混淆矩阵。代码如下：

【代码7-4-1】iris\_classification.py

# 混淆矩阵

cm = confusion\_matrix(y, y\_pred)

混淆矩阵如下图所示：

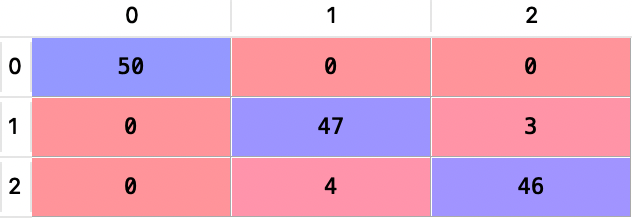


图3 混淆矩阵

在处理类别型变量部分，我们提到了我们把Iris-setosa转换为了0，'Iris-versicolor转换为了1，Iris-virginica转换为了2。在混淆矩阵中，0就代表Iris-setosa。而且，上面的0、1、2表示实际值，左边的0、1、2表示预测值。通过混淆矩阵我们可以看到，150条数据中只有7条数据预测错误，模型不错。模型对Iris-setosa的预测非常准确。

7.4.2 模型分类报告

【代码7-4-2】iris\_classification.py

# 评估模型分类报告

print(classification\_report(y,y\_pred))

模型分类报告如下图所示：

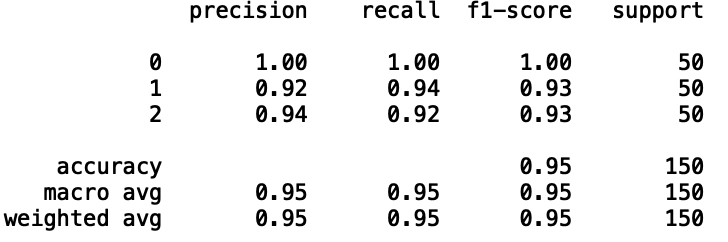


图4 模型性能报告

由此可见，precision、recall、f1-score三个指标的分数都很高，说明模型的性能不错。

7.5测试新数据

本阶段使用模型对一条新数据的鸢尾花种类进行预测。代码如下：

【代码7-5】iris\_classification.py

# 新数据预测

sepal\_length = 5.4 # 花萼长度，单位是cm

sepal\_width = 3.7 # 花萼宽度，单位是cm

petal\_length = 1.5 # 花瓣长度，单位是cm

petal\_width = 0.2 # 花瓣宽度，单位是cm

new\_data = np.array([[sepal\_length, sepal\_width, petal\_length, petal\_width]])

iris\_type = classifier.predict(sc.transform(new\_data))

if iris\_type == 0:

print('这是 山鸢尾（Iris Setosa）')

if iris\_type == 1:

print('这是 杂色鸢尾（Iris Versicolour）')

if iris\_type == 2:

print('这是 维吉尼亚鸢尾（Iris Virginica）')

输出结果如图5所示：



图5 执行结果

8. 案例代码

【案例代码】iris\_classification.py

# 导入包

import numpy as np

import pandas as pd

from sklearn.preprocessing import LabelEncoder

from sklearn.preprocessing import StandardScaler

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import confusion\_matrix

from sklearn.metrics import classification\_report

# 导入数据集

dataset = pd.read\_csv('Iris.csv')

# 将数据分成自变量和因变量

X = dataset.iloc[:, 1:5].values

y = dataset.iloc[:, 5].values

# 处理类别型字段

le = LabelEncoder()

y = le.fit\_transform(y)

print(le.classes\_)

# 特征缩放

sc = StandardScaler()

X = sc.fit\_transform(X)

# 训练模型

classifier = KNeighborsClassifier(n\_neighbors = 5, metric = 'minkowski', p = 2)

classifier.fit(X, y)

y\_pred = classifier.predict(X)

# 混淆矩阵

cm = confusion\_matrix(y, y\_pred)

# 评估模型分类报告

print(classification\_report(y,y\_pred))

# 新数据预测

sepal\_length = 5.4 # 花萼长度，单位是cm

sepal\_width = 3.7 # 花萼宽度，单位是cm

petal\_length = 1.5 # 花瓣长度，单位是cm

petal\_width = 0.2 # 花瓣宽度，单位是cm

new\_data = np.array([[sepal\_length, sepal\_width, petal\_length, petal\_width]])

iris\_type = classifier.predict(sc.transform(new\_data))

if iris\_type == 0:

print('这是 山鸢尾（Iris Setosa）')

if iris\_type == 1:

print('这是 杂色鸢尾（Iris Versicolour）')

if iris\_type == 2:

print('这是 维吉尼亚鸢尾（Iris Virginica）')