实验目录

实验内容

1. 基于逻辑回归的鸢尾花分类预测

知识点

1) 逻辑回归可以解决二分类问题
2) 混淆矩阵可以用于初步分析预测准确性
3) 精确率、召回率、准确率、f1-score 可以量化衡量模型
4) 交叉验证得出的结果往往更加客观

实验目的

1) 学习建立逻辑回归模型预测鸢尾花分类
2) 学习解读混淆矩阵和相关得分

实验步骤

2) 读取数据

1. Jupyter 输入代码后，使用 shift+enter 执行，下同。
2. 鸢尾花数据集是一个非常著名的数据集，数据包括三种鸢尾花花萼、花瓣的长宽（厘米计量），每种鸢尾花包括 50 个样本，共 150 个样本。本实验抽取其中两种鸢尾花的花瓣长宽数据，共 100 个样本进行试验。保留字段如下:
   petal_l：花瓣长度，厘米计量
   petal_w：花瓣宽度，厘米计量
   classes：鸢尾花种类，标记为 0-1
3. 使用 pandas 读取 csv 文件

[Code 001]:

```
import pandas as pd
data_iris = pd.read_csv('/root/experiment/datas/iris_partial.csv',index_col=0)
# 查看数据的维度
data_iris.shape
```

```
import pandas as pd

data_iris = pd.read_csv('/root/experiment/datas/iris_partial.csv',index_col=0)
data_iris.shape
```

```
(100, 3)
```

3) 描述性分析与可视化分析

1. 查看数据的随机五项

[Code 002]:

data_iris.sample(5)

```
data_iris.sample(5)
```

	petal_l	petal_w	classes
18	1.7	0.3	0
76	4.8	1.4	1
11	1.6	0.2	0
25	1.6	0.2	0
93	3.3	1.0	1

2. 查看数据的统计描述

[Code 003]:

data_iris.describe()

```
data_iris.describe()
```

	petal_l	petal_w	classes
count	100.000000	100.000000	100.000000
mean	2.862000	0.785000	0.500000
std	1.448565	0.566288	0.502519
min	1.000000	0.100000	0.000000
25%	1.500000	0.200000	0.000000
50%	2.450000	0.800000	0.500000
75%	4.325000	1.300000	1.000000
max	5.100000	1.800000	1.000000

3. 查看数据的缺失值

[Code 004]:

data_iris.isnull().sum()

```
data_iris.isnull().sum()
```

```
petal_l    0
petal_w    0
classes    0
dtype: int64
```

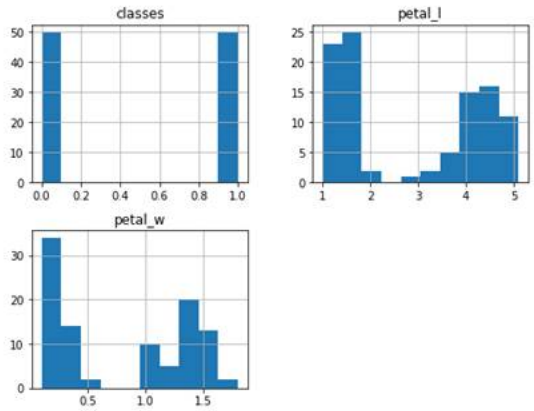
4. 查看数据分布（绘图时，由于 jupyter 的问题，执行时可能需重复执行才能显示绘图结果，下同）

[Code 005]:

import matplotlib.pyplot as plt
data_iris.hist(figsize=(8,6))
plt.show()

```
import matplotlib.pyplot as plt

data_iris.hist(figsize=(8,6))
plt.show()
```

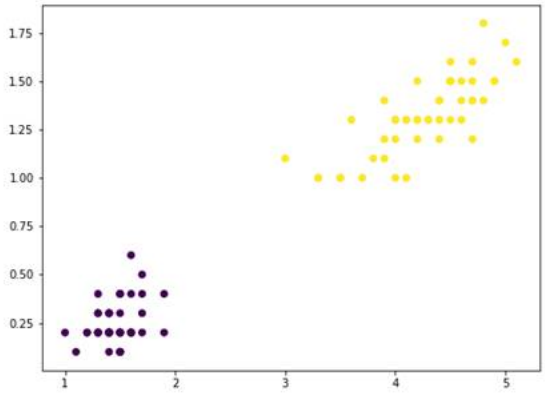


5. 数据可视化

[Code 006]:

plt.figure(figsize=(8,6))
plt.scatter(x=data_iris['petal_l'],
y=data_iris['petal_w'],c=data_iris['classes'])
plt.show()

```
plt.figure(figsize=(8,6))
plt.scatter(x=data_iris['petal_l'], y=data_iris['petal_w'],c=data_iris['classes'])
plt.show()
```



4) 数据预处理

1. 划分自变量和因变量，训练集和测试集

[Code 007]:

 # 定义自变量和因变量
X = data_iris.iloc[:,:-1]
y = data_iris.iloc[:,-1]
 # 划分训练集和测试集
from sklearn.model_selection import train_test_split
X_tr,X_ts,y_tr,y_ts = train_test_split(X,y,test_size=0.2)

5) 建立模型

1. 建立并训练模型

[Code 008]:

from sklearn.linear_model import LogisticRegression
model = LogisticRegression()
model.fit(X_tr,y_tr)

```
from sklearn.linear_model import LogisticRegression

model = LogisticRegression()
model.fit(X_tr,y_tr)
```

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=None, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

6) 模型预测与评估

1. 模型预测

[Code 009]:

y_pred = model.predict(X_ts)
y_pred

```
y_pred = model.predict(X_ts)
y_pred
```

```
array([1, 0, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 1, 0, 1, 0, 1, 1, 0, 1])
```

2. 查看混淆矩阵

[Code 0010]:

from sklearn.metrics import confusion_matrix
confusion_matrix(y_ts,y_pred)

```
from sklearn.metrics import confusion_matrix

confusion_matrix(y_ts,y_pred)
```

```
array([[ 9,  0],
       [ 0, 11]])
```

3. 查看相关得分

[Code 011]:

from sklearn.metrics import classification_report
print(classification_report(y_ts,y_pred))

```
from sklearn.metrics import classification_report

print(classification_report(y_ts,y_pred))
```

```
             precision    recall  f1-score   support

          0       1.00      1.00      1.00         9
          1       1.00      1.00      1.00        11

avg / total       1.00      1.00      1.00        20
```

4. 交叉验证计算准确率

[Code 012]:

from sklearn.model_selection import cross_val_score
scores = cross_val_score(model, X, y, cv=10, scoring='accuracy')
scores.mean()

```
from sklearn.model_selection import cross_val_score

scores = cross_val_score(model, X, y, cv=10, scoring='accuracy')
scores.mean()
```

```
1.0
```

7) 实验结论

1. 逻辑回归可以解决二分类问题。
2. 混淆矩阵对角线上的元素表示“预测正确”。
3. 逻辑回归模型 f1-score 得分为 1
4. 交叉验证模型准确率得分为 1