硕士第3周作业

在“1、实现交叉验证法”基础上实现留10法

交叉验证法先将数据集划分为k个大小相似的互斥子集，每次采用$k-1$个子集的并集作为训练集，剩下的那个子集作为测试集。进行k次训练和测试，最终返回k个测试结果的均值。又称为“k折交叉验证”（k-fold cross validation）。

代码：

```
from sklearn.datasets import load_iris
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression

# X为数据的特征，y为数据的标签
X, y = load_iris(return_X_y=True)

# num_folds = 5
# num_fold_samples = len(X)/num_folds
num_fold_samples = 10
num_folds = int(len(X) / num_fold_samples)

res = []
shuffle_indexes = np.random.permutation(len(X))
X,y = X[shuffle_indexes], y[shuffle_indexes]
for fold in range(num_folds):
    #计算测试集开始和结束的索引
    start_inx = int(np.ceil(fold*num_fold_samples))
    end_inx = int(np.ceil((fold+1)*num_fold_samples))
    print(start_inx,end_inx)
    #取到训练集和测试集
    X_train = X[np.r_[:start_inx,end_inx:len(X)]]
    y_train = y[np.r_[:start_inx,end_inx:len(X)]]
    X_test = X[start_inx:end_inx]
    y_test = y[start_inx:end_inx]
    #训练和测试
    clf = LogisticRegression().fit(X_train, y_train)
    res.append(clf.score(X_test, y_test))
print("结果列表：", res)
print("最终结果：", np.average(res))
```

结果：

```
0 10
10 20
20 30
30 40
40 50
50 60
60 70
70 80
80 90
90 100
100 110
110 120
120 130
130 140
140 150
结果列表： [1.0, 1.0, 1.0, 1.0, 1.0, 0.9, 0.9, 1.0, 0.8, 1.0, 1.0, 0.9, 1.0, 1.0, 1.0]
最终结果： 0.9666666666666667
```

在“5、多分类混淆矩阵(Sklearn)”基础上附上手动计算宏精确率、微精确率的代码

宏平均（Macro-averaging），是先对每一个类统计指标值，然后在对所有类求算术平均值。

$$P_{macro} = \frac{1}{n}\sum_{i=1}^{n} P_i$$

微平均（Micro-averaging），是对数据集中的每一个实例不分类别进行统计建立全局混淆矩阵，然后计算相应指标。

$$P_{micro} = \frac{\bar{TP}}{\bar{TP} + \bar{FP}} = \frac{\sum_{i=1}^{n} TP_i}{\sum_{i=1}^{n} TP_i + \sum_{i=1}^{n} FP_i}$$

代码：

```
import numpy as np
from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn import metrics

X, y = load_iris(return_X_y=True)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.5)

#训练与测试
clf = LogisticRegression().fit(X_train, y_train)
y_pred = clf.predict(X_test)

print("混淆矩阵:\n",metrics.confusion_matrix(y_test,y_pred))
```

```
print("宏精确率:",metrics.precision_score(y_test,y_pred, average="macro"))
print("微精确率:",metrics.precision_score(y_test,y_pred, average="micro"))

# 手动计算
precision_macro = 0
for i in range( 3 ):
    precision_macro += np.sum((y_pred == i) & (y_test == i)) / np.sum(y_pred == i)
precision_macro /= 3

precision_micro = np.sum(y_pred == y_test) / len(y_test)

print("**********手动计算**********")
print("宏精确率:", precision_macro)
print("微精确率:", precision_micro)
```

结果:

```
混淆矩阵:
 [[26  0  0]
 [ 0 26  1]
 [ 0  1 21]]
宏精确率: 0.9725028058361391
微精确率: 0.9733333333333334
**********手动计算**********
宏精确率: 0.9725028058361391
微精确率: 0.9733333333333334
```