Python编程与人工智能实践

算法篇：KNN（K近邻）

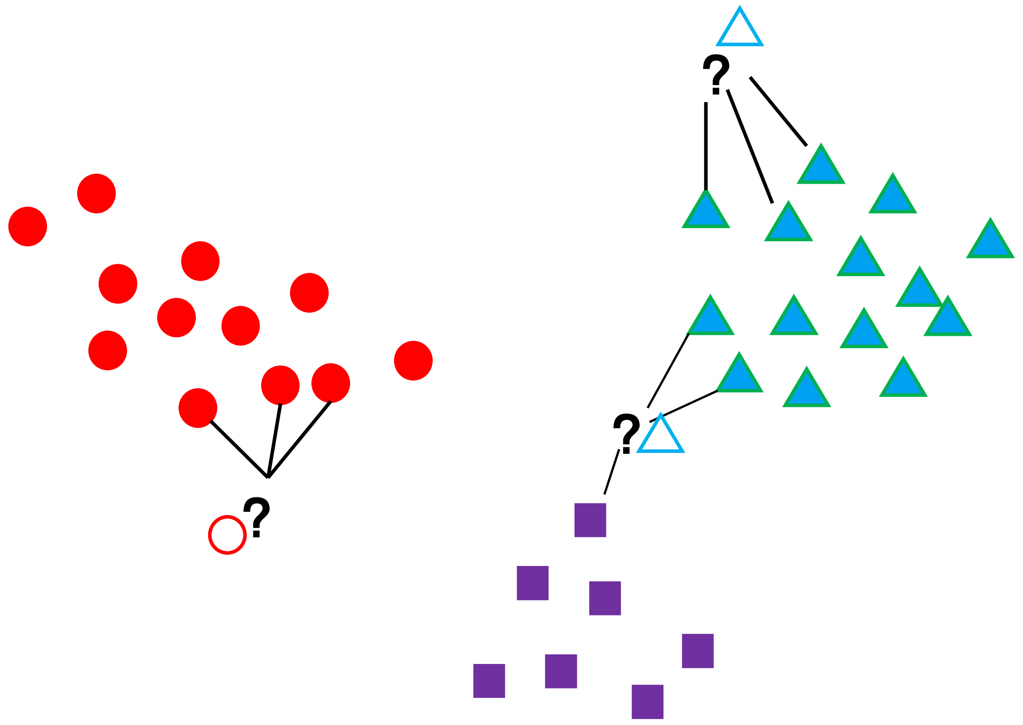
于泓
鲁东大学
信息与电气工程学院

2021.3.20

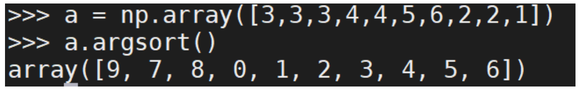
K近邻分类算法（K Nearest Neighbor）

- 在训练集中找寻距离测试样本最近的K个样本。通过统计这K个样本的类别，来判断测试样本的类别

是否放贷?



```python
'''
    trainData - 训练集  N,D
    testData - 测试   1,D
    labels - 训练集标签
'''
def knn(trainData, testData, labels, k):
    # 计算训练样本的行数
    rowSize = trainData.shape[0]
    # 计算训练样本和测试样本的差值
    diff = np.tile(testData, (rowSize, 1)) - trainData
    # 计算差值的平方和
    sqrDiff = diff ** 2
    sqrDiffSum = sqrDiff.sum(axis=1)
    # 计算距离
    distances = sqrDiffSum ** 0.5
    # 对所得的距离从低到高进行排序
    sortDistance = distances.argsort()

    count = {}

    for i in range(k):
        vote = labels[sortDistance[i]]
        # print(vote)
        count[vote] = count.get(vote, 0) + 1
    # 对类别出现的频数从高到低进行排序
    sortCount = sorted(count.items(), key=operator.itemgetter(1), reverse=True)

    # 返回出现频数最高的类别
    return sortCount[0][0]
```



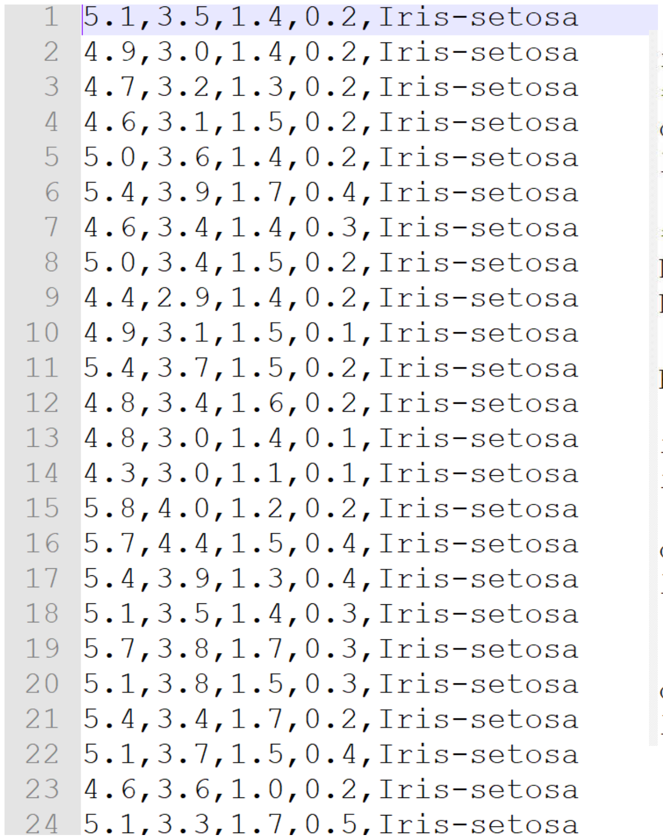
```
>>> a = np.array([3,3,3,4,4,5,6,2,2,1])
>>> a.argsort()
array([9, 7, 8, 0, 1, 2, 3, 4, 5, 6])
```

$\sqrt{(y_1-x_1)^2+(y_2-x_2)^2+}$

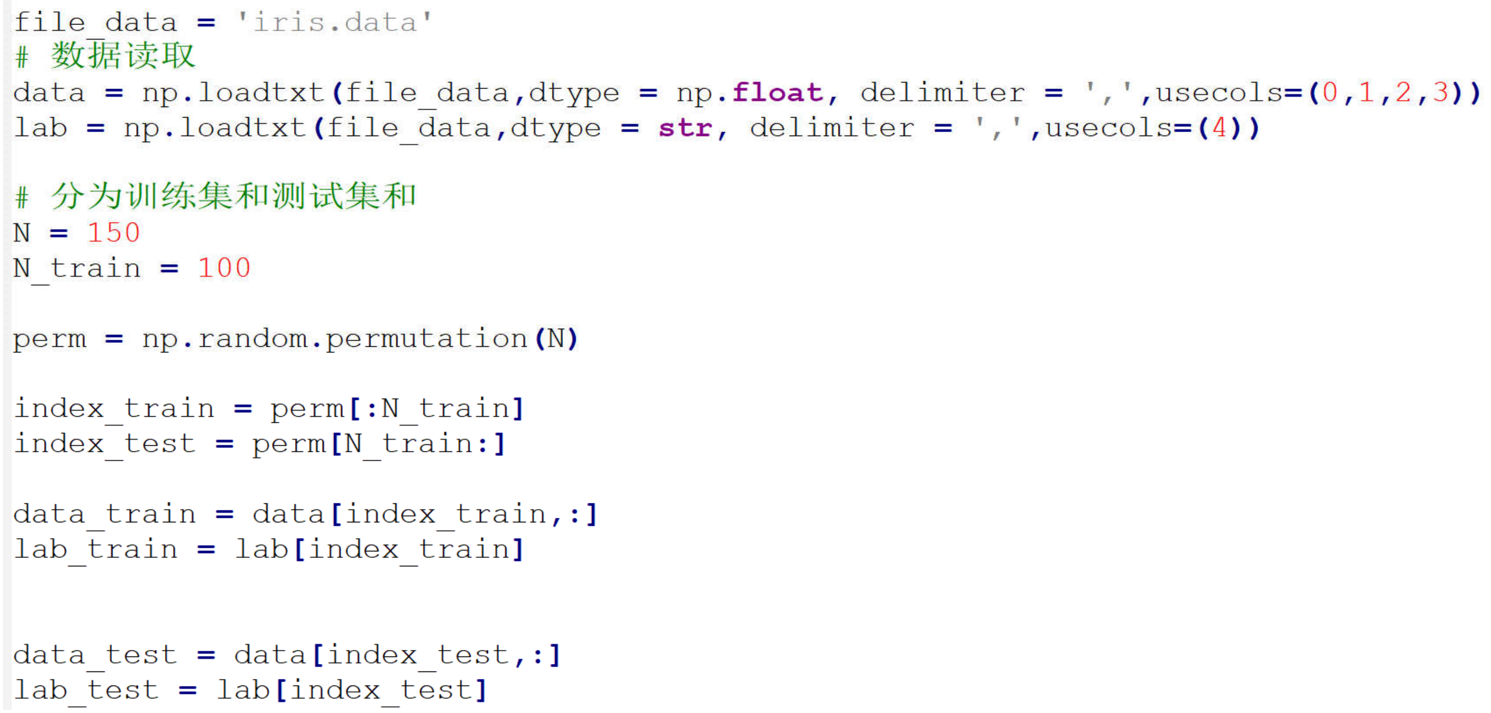
[vote]

$(y_n-x_n)^2$

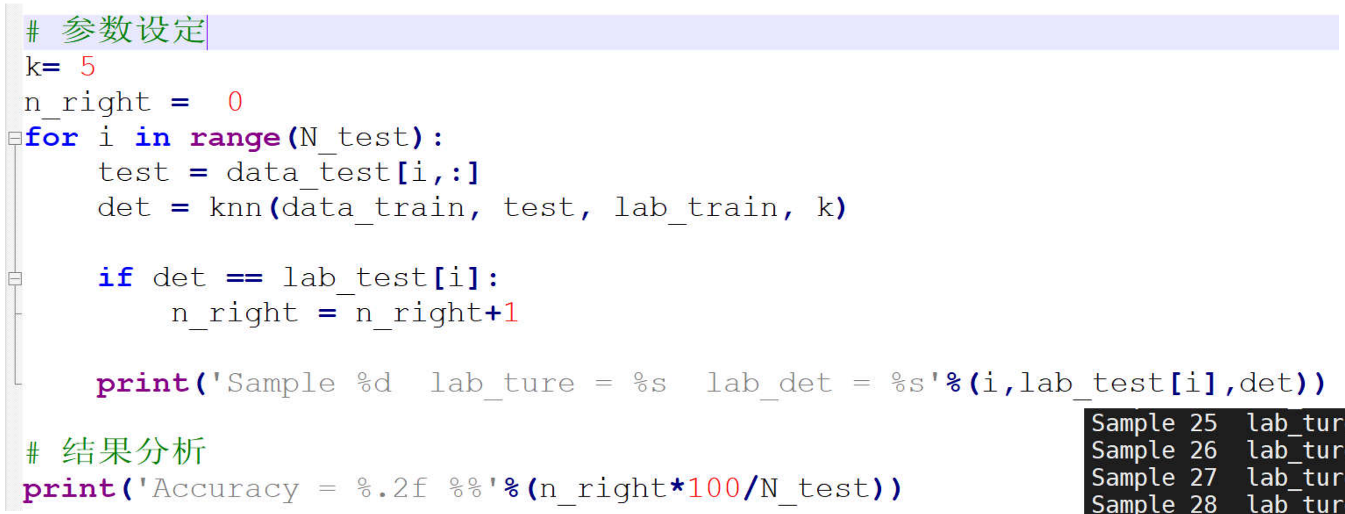
鸢尾花数据集（Iris）



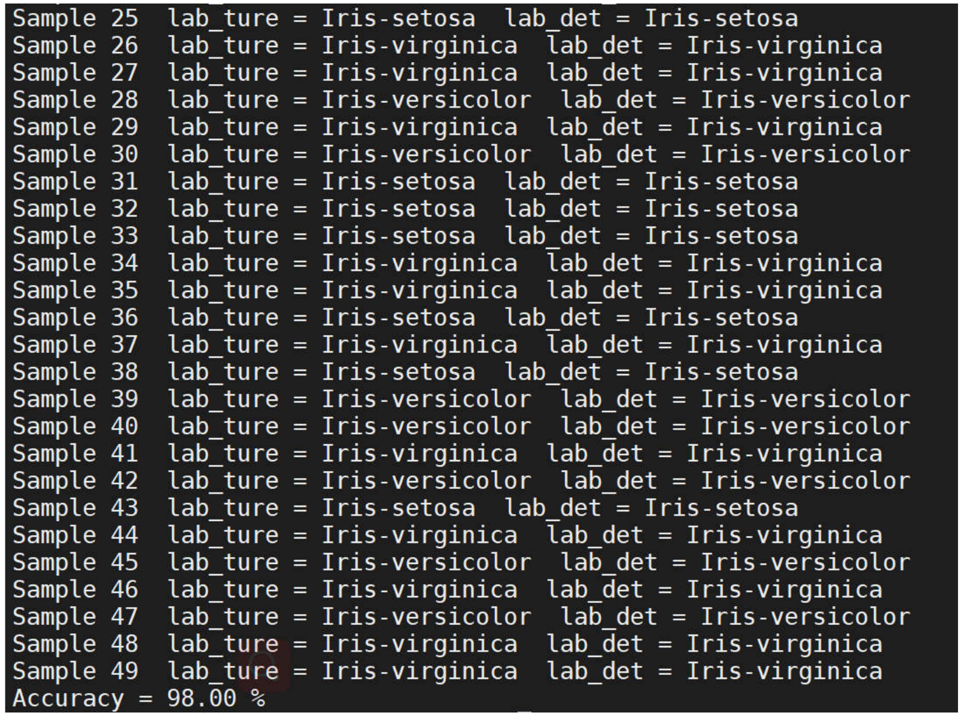
```
5.1,3.5,1.4,0.2,Iris-setosa
4.9,3.0,1.4,0.2,Iris-setosa
4.7,3.2,1.3,0.2,Iris-setosa
4.6,3.1,1.5,0.2,Iris-setosa
5.0,3.6,1.4,0.2,Iris-setosa
5.4,3.9,1.7,0.4,Iris-setosa
4.6,3.4,1.4,0.3,Iris-setosa
5.0,3.4,1.5,0.2,Iris-setosa
4.4,2.9,1.4,0.2,Iris-setosa
4.9,3.1,1.5,0.1,Iris-setosa
5.4,3.7,1.5,0.2,Iris-setosa
4.8,3.4,1.6,0.2,Iris-setosa
4.8,3.0,1.4,0.1,Iris-setosa
4.3,3.0,1.1,0.1,Iris-setosa
5.8,4.0,1.2,0.2,Iris-setosa
5.7,4.4,1.5,0.4,Iris-setosa
5.4,3.9,1.3,0.4,Iris-setosa
5.1,3.5,1.4,0.3,Iris-setosa
5.7,3.8,1.7,0.3,Iris-setosa
5.1,3.8,1.5,0.3,Iris-setosa
5.4,3.4,1.7,0.2,Iris-setosa
5.1,3.7,1.5,0.4,Iris-setosa
4.6,3.6,1.0,0.2,Iris-setosa
5.1,3.3,1.7,0.5,Iris-setosa
```



```python
file_data = 'iris.data'
# 数据读取
data = np.loadtxt(file_data,dtype = np.float, delimiter = ',',usecols=(0,1,2,3))
lab = np.loadtxt(file_data,dtype = str, delimiter = ',',usecols=(4))

# 分为训练集和测试集和
N = 150
N_train = 100

perm = np.random.permutation(N)

index_train = perm[:N_train]
index_test = perm[N_train:]

data_train = data[index_train,:]
lab_train = lab[index_train]


data_test = data[index_test,:]
lab_test = lab[index_test]
```



```
# 参数设定
k= 5
n_right =  0
for i in range(N_test):
    test = data_test[i,:]
    det = knn(data_train, test, lab_train, k)

    if det == lab_test[i]:
        n_right = n_right+1

    print('Sample %d  lab_ture = %s  lab_det = %s'%(i,lab_test[i],det))

# 结果分析
print('Accuracy = %.2f %%'%(n_right*100/N_test))
```



```
Sample 25  lab_ture = Iris-setosa  lab_det = Iris-setosa
Sample 26  lab_ture = Iris-virginica  lab_det = Iris-virginica
Sample 27  lab_ture = Iris-virginica  lab_det = Iris-virginica
Sample 28  lab_ture = Iris-versicolor  lab_det = Iris-versicolor
Sample 29  lab_ture = Iris-virginica  lab_det = Iris-virginica
Sample 30  lab_ture = Iris-versicolor  lab_det = Iris-versicolor
Sample 31  lab_ture = Iris-setosa  lab_det = Iris-setosa
Sample 32  lab_ture = Iris-setosa  lab_det = Iris-setosa
Sample 33  lab_ture = Iris-setosa  lab_det = Iris-setosa
Sample 34  lab_ture = Iris-virginica  lab_det = Iris-virginica
Sample 35  lab_ture = Iris-virginica  lab_det = Iris-virginica
Sample 36  lab_ture = Iris-setosa  lab_det = Iris-setosa
Sample 37  lab_ture = Iris-virginica  lab_det = Iris-virginica
Sample 38  lab_ture = Iris-setosa  lab_det = Iris-setosa
Sample 39  lab_ture = Iris-versicolor  lab_det = Iris-versicolor
Sample 40  lab_ture = Iris-versicolor  lab_det = Iris-versicolor
Sample 41  lab_ture = Iris-virginica  lab_det = Iris-virginica
Sample 42  lab_ture = Iris-versicolor  lab_det = Iris-versicolor
Sample 43  lab_ture = Iris-setosa  lab_det = Iris-setosa
Sample 44  lab_ture = Iris-virginica  lab_det = Iris-virginica
Sample 45  lab_ture = Iris-versicolor  lab_det = Iris-versicolor
Sample 46  lab_ture = Iris-virginica  lab_det = Iris-virginica
Sample 47  lab_ture = Iris-versicolor  lab_det = Iris-versicolor
Sample 48  lab_ture = Iris-virginica  lab_det = Iris-virginica
Sample 49  lab_ture = Iris-virginica  lab_det = Iris-virginica
Accuracy = 98.00 %
```