应用sklearn的K近邻算法进行类别预测

In [ ]:

```
import pandas as pd
from sklearn.model_selection import train_test_split     #导入切分训练集、测试集模
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import normalize
import numpy as np
```

读取数据，替换掉字符串

In [ ]:

```
fruit=pd.read_csv('./data/fruit_data_with_colors.txt',sep='\t')
fruit.replace('granny_smith',1,inplace=True)
fruit.replace('mandarin',2,inplace=True)
fruit.replace('braeburn',3,inplace=True)
fruit.replace('golden_delicious',4,inplace=True)
fruit.replace('cripps_pink',5,inplace=True)
fruit.replace('spanish_jumbo',6,inplace=True)
fruit.replace('selected_seconds',7,inplace=True)
fruit.replace('turkey_navel',8,inplace=True)
fruit.replace('spanish_belsan',9,inplace=True)
fruit.replace('unknown',9,inplace=True)
fruit.head()
```

Out[ ]:

	fruit_label	fruit_name	fruit_subtype	mass	width	height	color_score
0	1	apple	1	192	8.4	7.3	0.55
1	1	apple	1	180	8.0	6.8	0.59
2	1	apple	1	176	7.4	7.2	0.60
3	2	2	2	86	6.2	4.7	0.80
4	2	2	2	84	6.0	4.6	0.79

In [ ]:

```
fruit_x=fruit.iloc[:,2:]
fruit_x.head()
```

Out[ ]:

	fruit_subtype	mass	width	height	color_score
0	1	192	8.4	7.3	0.55
1	1	180	8.0	6.8	0.59
2	1	176	7.4	7.2	0.60
3	2	86	6.2	4.7	0.80
4	2	84	6.0	4.6	0.79

In [ ]:

```
fruit_y=fruit.iloc[:,0]
fruit_y.head()
```

Out[ ]:

```
0    1
1    1
2    1
3    2
4    2
Name: fruit_label, dtype: int64
```

按不同类别的数据进行归一化

```
In [ ]: fruit_x=normalize(fruit_x,'l1',axis=0)
        fruit_x[:5,:]
```

```
Out[ ]: array([[0.00276243, 0.01995012, 0.02003817, 0.01608284, 0.01221951],
               [0.00276243, 0.01870324, 0.01908397, 0.01498127, 0.0131082 ],
               [0.00276243, 0.01828761, 0.01765267, 0.01586252, 0.01333037],
               [0.00552486, 0.00893599, 0.01479008, 0.0103547 , 0.01777383],
               [0.00552486, 0.00872818, 0.01431298, 0.01013439, 0.01755166]])
```

分为训练数据和测试数据

```
In [ ]: x_train, x_test , y_train, y_test = train_test_split(fruit_x, fruit_y, test_size =
```

将训练样本放入K近邻算法，并预测测试数据集类型，输出准确率

```
In [ ]: knn = KNeighborsClassifier(n_neighbors=3)    #实例化KNN模型
        knn.fit(x_train, y_train)      #放入训练数据进行训练
        print("预测： ",knn.predict(x_test))
        print("实际： ",np.array(y_test))
        print('测试集精确度',(knn.predict(x_test)==y_test).sum()/len(x_test))          #打
```

```
预测：  [3 4 1 4 4 4 3 4 1 3 2 1 1 1 3 1 2 4]
实际：  [3 4 1 4 4 4 3 4 1 3 2 1 1 1 3 1 2 4]
测试集精确度 1.0
```