KNN regression

In [1]:
```
import numpy as np
import pandas as pd
```

In [2]:
```
data = pd.read_csv(r"iris.csv")
# 删除不需要的列：Unnamed:0,Species
data.drop(["Unnamed: 0","Species"],axis=1,inplace=True)
# 删除重复的记录
data.drop_duplicates(inplace=True)
data.head()
```

Out[2]:

	Sepal.Length	Sepal.Width	Petal.Length	Petal.Width
0	5.1	3.5	1.4	0.2
1	4.9	3.0	1.4	0.2
2	4.7	3.2	1.3	0.2
3	4.6	3.1	1.5	0.2
4	5.0	3.6	1.4	0.2

In [3]:
```
class KNN:
    """
    使用python 实现K近邻算法（回归预测）
    根据前3个特征属性，寻找最近的k个邻居，再根据k个邻居的第4个特征属性，预测当前样
    """
    def __init__(self,k):
        """初始化方法
        Parameters
        ------
        k:int
          邻居的个数
        """
        self.k = k
    def fit(self,X,y):
        """训练方法
        Parameters
        ------
        X:类数组类型（特征矩阵），形状为[样本数量,特征数量]
          待训练的样本特征（属性）
        y:类数组类型（目标标签），形状为[样本数量]
          每个样本的目标值（标签）
        """
        self.X = np.asarray(X)
        self.y = np.asarray(y) # 将X与y转换成 ndarray数组形式，方便统一进行操作
    def predict(self,X):
        """根据参数传递的X，对样本数据进行预测
        Parameters:
        ------
        X:类数组类型，形状为[样本数量，特征数量]
          待测试的样本特征（属性）
        Returns
        ------
        result:数组类型
               预测的结果值
```

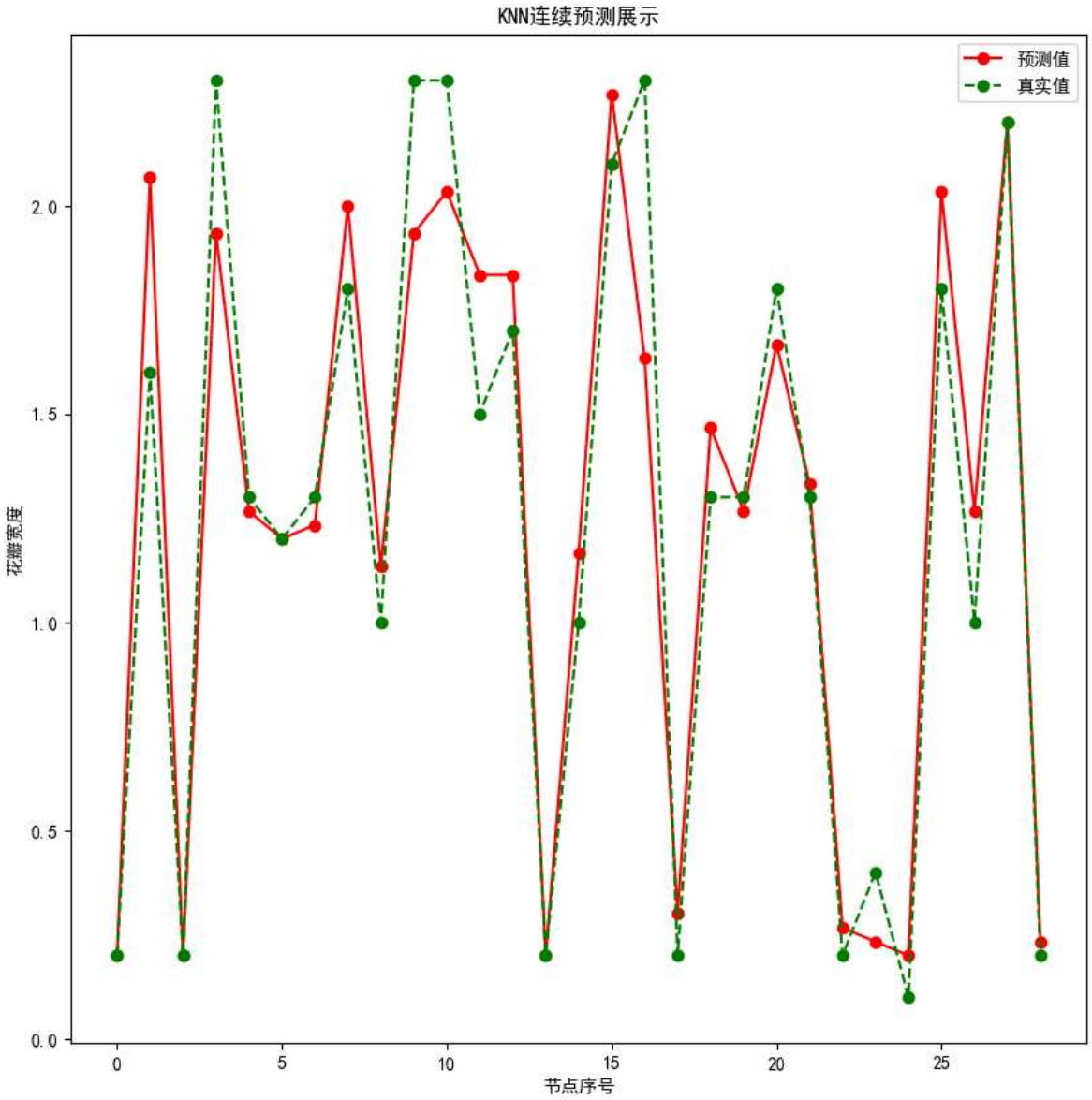
```
        """
        # 转换成数组类型
        X = np.asarray(X)
        result = [] # 保存预测的结果值
        for x in X:
            # 计算与训练集中每个X的距离
            dis = np.sqrt(np.sum((x-self.X) ** 2, axis=1))
            # 返回数组排序后，每个元素在原数组中（排序之前的数组）的索引
            index = dis.argsort()
            # 取原数组中前k个距离最近的索引
            index = index[:self.k]
            # 计算均值，并加入到结果列表中
            result.append(np.mean(self.y[index]))
        return np.array(result)
```

In [4]:
```
t = data.sample(len(data), random_state=0)
train_X = t.iloc[:120,:-1]
train_y = t.iloc[:120,-1]
test_X = t.iloc[120:,:-1]
test_y = t.iloc[120:,-1]
knn = KNN(k=3) # 选择3个邻居
knn.fit(train_X,train_y) # 训练
result = knn.predict(test_X)
display(result)
np.mean(np.sum((result-test_y) ** 2))
display(test_y.values)
```

```
array([0.2       , 2.06666667, 0.2       , 1.93333333, 1.26666667,
       1.2       , 1.23333333, 2.        , 1.13333333, 1.93333333,
       2.03333333, 1.83333333, 1.83333333, 0.2       , 1.16666667,
       2.26666667, 1.63333333, 0.3       , 1.46666667, 1.26666667,
       1.66666667, 1.33333333, 0.26666667, 0.23333333, 0.2       ,
       2.03333333, 1.26666667, 2.2       , 0.23333333])
array([0.2, 1.6, 0.2, 2.3, 1.3, 1.2, 1.3, 1.8, 1. , 2.3, 2.3, 1.5, 1.7,
       0.2, 1. , 2.1, 2.3, 0.2, 1.3, 1.3, 1.8, 1.3, 0.2, 0.4, 0.1, 1.8,
       1. , 2.2, 0.2])
```

In [5]:
```
# 可视化
import matplotlib as mpl
import matplotlib.pyplot as plt
mpl.rcParams["font.family"] = "SimHei"
mpl.rcParams["axes.unicode_minus"] = False
```

In [6]:
```
plt.figure(figsize=(10,10))
# 绘制预测值
plt.plot(result,"ro-",label="预测值")
# 绘制真实值
plt.plot(test_y.values,"go--",label="真实值")
plt.title("KNN连续预测展示")
plt.xlabel("节点序号")
plt.ylabel("花瓣宽度")
plt.legend()
plt.show()
```


KNN连续预测展示
预测值
真实值
花瓣宽度
节点序号
0.0
0.5
1.0
1.5
2.0
0
5
10
15
20
25