KNN最近邻算法

构建一个已经分类好的数据集

```
movie_data= {"宝贝当家":[45,2,9,"喜剧片"],
             "美人鱼":[21,17,5,"喜剧片"],
             "澳门风云3":[54,9,11,"喜剧片"],
             "谍影重重":[5,2,57,"动作片"],
             "叶问3":[3,2,65,"动作片"],
             "奔爱":[7,46,4,"爱情片"],
             "代理情人":[9,38,2,"爱情片"],
             "我的特工爷爷":[6,4,21,"动作片"]}

print(movie_data)
{'澳门风云3': [54, 9, 11, '喜剧片'], '宝贝当家': [45, 2, 9, '喜剧片'], '叶问3': [3, 2, 65, '动作
片'], '美人鱼': [21, 17, 5, '喜剧片'], '代理情人': [9, 38, 2, '爱情片'], '奔爱': [7, 46, 4, '爱情
片'], '谍影重重': [5, 2, 57, '动作片']}
```

计算一个新样本与数据集中所有数据的距离

- 这里的新样本是："唐人街探案":[23,3,17,"?片"]。欧氏距离是常用的距离计算方法。d = 根号（求和（Xi - Yi）平方），其中x,y为2个样本，n为维度，Xi,Yi为x,y第i个维度上的特征值。如x为:"唐人街探案":[23,3,17,"?"],y为："伦敦陷落":[2,3,55,"动作片"]，则二者距离为：d = 根号((23-2)2 + (3 - 3)2 + (17 - 55)2) = 43.42，具体代码如下：

```
import math
x = [23,3,17]
KNN = []
for key,v in movie_data.items():
    d = math.sqrt((x[0] - v[0]) **2 + (x[1] - v[1]) **2 + (x[2] - v[2]) **2)
    KNN.append([key,round(d,2)])
print(KNN)
[['澳门风云3', 32.14], ['宝贝当家', 23.43], ['叶问3', 52.01], ['美人鱼', 18.55], ['代理情人',
40.57], ['奔爱', 47.69], ['谍影重重', 43.87]]
```

按照距离大小进行递增排序

```
KNN.sort(key = lambda dis : dis[1])
print(KNN)
[['美人鱼', 18.55], ['宝贝当家', 23.43], ['澳门风云3', 32.14], ['代理情人', 40.57], ['谍影重重',
43.87], ['奔爱', 47.69], ['叶问3', 52.01]]
```

选取距离最小的K个样本

```
#这里K取5
KNN = KNN[:5]
print(KNN)
[['美人鱼', 18.55], ['宝贝当家', 23.43], ['澳门风云3', 32.14], ['代理情人', 40.57], ['谍影重重',
43.87]]
```

确定前k个样本所在的类别出现的频率，并输出出现频率最高的类别

```
labels = {"喜剧片":0,"动作片":0,"爱情片":0}
for s in KNN:
    label = movie_data[s[0]]
    labels[label[3]] += 1
labels = sorted(labels.items(),key = lambda l:l[1],reverse = True)
print(labels,labels[0][0],sep = '\n')
[('喜剧片', 3), ('爱情片', 1), ('动作片', 1)]
喜剧片
```

KNN特点：

- KNN属于惰性学习
- KNN的计算复杂度高，时间复杂度为O（n），适用于样本数较少的数据集。
- k取不同值时，分类结果可能会有不同，一般取值不超过20。