KNN最近邻算法（K - Nearest Neighbors）

- K最近邻算法是一种分类算法，算法思想是一个样本与数据集中的K个样本最相似，如果这K个样本中的大多数属于某一类别，则该样本也属于某一类别。
- 步骤：
  - 构建一个已经分类好的数据集。
  - 计算一个新样本与数据集中所有数据的距离。
  - 按照距离大小进行递增排序。
  - 选取距离最小的K个样本。
  - 确定前K个样本所在类别出现的频率，并输出出现频率最高的类别。
- 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)，适用于样本较少的数据集。
  - K取不同值时，分类结果可能不同。
- 距离计算方法：
  - 欧氏距离：

$$d = sqrt(\sum(xi - yi)^2)$$

    - x，y：两个样本
    - n：维度
    - xi，yi：x，y在第i个维度上的特征值
  - 曼哈顿距离：

$$d = sqrt(\sum|xi - yi|)$$

- K近邻模型的三个基本要素：距离度量，K值的选择，分类决策规则。
- K值的选择
  - 选择较小的K值，学习的近似误差会减小，缺点是学习的估计误差会增大。
  - 选择较大的K值，可以减少学习的估计误差，缺点是学习的近似误差会增大。
  - 在应用中，K值一般采用交叉验证的方法来选取最优的K值。
- 分类决策规则：多数表决规则。

```
import math
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"],
             "功夫熊猫3": [39, 0, 31, "喜剧片"],
             "谍影重重": [5, 2, 57, "动作片"],
             "叶问3": [3, 2, 65, "动作片"],
             "伦敦陷落": [2, 3, 55, "动作片"],
             "我的特工爷爷": [6, 4, 21, "动作片"],
             "奔爱": [7, 46, 4, "爱情片"],
             "夜孔雀": [9, 39, 8, "爱情片"],
             "代理情人": [9, 38, 2, "爱情片"],
```

```
                "新步步惊心": [8, 34, 17, "爱情片"]}

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)

KNN.sort(key=lambda dis : dis[1])

print(KNN)

#这里K取5
KNN = KNN[:5]
print(KNN)

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')
# 结果输出：喜剧片
```

K折交叉验证

- K折交叉验证：将原始数据集随机分为K份，K-1份数据用于模型训练，剩下一份用于测试模型。重复第二步K次，得到K个模型和评估结果。