# 实验五 基于K-means的语音、噪声分类（男女声分类），采用MFCC为特征，用K-means做聚类来做男女声分类

# 实验目的

## 1.1学习与掌握K-means算法

## 1.2掌握如何用K-means做聚类来对男女声分类

# 2、实验设备

应用软件Matlab2018a

# 3、实验原理

### 3.1语音信号的采集

采集语音（男女声），具体方法原理实验一已有说明

### 3.2K-means聚类算法简介

K表示类别数，means表示均值，所以K-means是一种通过均值对数据点进行聚类的算法。

由于具有出色的速度和良好的可扩展性，K-means聚类算法算得上是最著名的聚类方法。K-means算法是一个重复移动类中心点的过程，把类的中心点，也称重心(centroids)，移动到其包含成员的平均位置，然后重新划分其内部成员。k是算法计算出的超参数，表示类的数量；K-means可以自动分配样本到不同的类，但是不能决定究竟要分几个类。k必须是一个比训练集样本数小的正整数。有时，类的数量是由问题内容指定的。例如，一个鞋厂有三种新款式，它想知道每种新款式都有哪些潜在客户，于是它调研客户，然后从数据里找出三类。也有一些问题没有指定聚类的数量，最优的聚类数量是不确定的。

K-means的参数是类的重心位置和其内部观测值的位置。与广义线性模型和决策树类似，K-means参数的最优解也是以成本函数最小化为目标。K-means成本函数公式如下：

μi是第k个类的重心位置。成本函数是各个类畸变程度(distortions)之和。每个类的畸变程度等于该类重心与其内部成员位置距离的平方和。若类内部的成员彼此间越紧凑则类的畸变程度越小，反之，若类内部的成员彼此间越分散则类的畸变程度越大。求解成本函数最小化的参数就是一个重复配置每个类包含的观测值，并不断移动类重心的过程。首先，类的重心是随机确定的位置。实际上，重心位置等于随机选择的观测值的位置。每次迭代的时候，K-means会把观测值分配到离它们最近的类，然后把重心移动到该类全部成员位置的平均值那里。

#### 3.2.1K值得确定

##### 3.2.1.1根据问题内容确定

##### 3.2.1.2肘部法则

如果问题中没有指定k的值，可以通过肘部法则这一技术来估计聚类数量。肘部法则会把不同k值的成本函数值画出来。随着k值的增大，平均畸变程度会减小；每个类包含的样本数会减少，于是样本离其重心会更近。但是，随着k值继续增大，平均畸变程度的改善效果会不断减低。k值增大过程中，畸变程度的改善效果下降幅度最大的位置对应的k值就是肘部。为了让读者看的更加明白，下面让我们通过一张图用肘部法则来确定最佳的k值。下图数据明显可分成两类：

从图中可以看出，k值从1到2时，平均畸变程度变化最大。超过2以后，平均畸变程度变化显著降低。因此最佳的k是2。

##### 3.2.1.3与层次聚类结合

经常会产生较好的聚类结果的一个有趣策略是，首先采用层次凝聚算法决定结果粗的数目，并找到一个初始聚类，然后用迭代重定位来改进该聚类。

##### 3.2.1.4稳定性方法

稳定性方法对一个数据集进行2次重采样产生2个数据子集，再用相同的聚类算法对2个数据子集进行聚类，产生2个具有k个聚类的聚类结果，计算2个聚类结果的相似度的分布情况。2个聚类结果具有高的相似度说明k个聚类反映了稳定的聚类结构，其相似度可以用来估计聚类个数。采用多次方法试探多个k，找到合适的k值。

##### 3.2.1.5系统演化方法

系统演化方法将一个数据集视为伪热力学系统，当数据集被划分为k个聚类时称系统处于状态k。系统由初始状态k=1出发，经过分裂过程和合并过程，系统将演化到它的稳定平衡状态 ki，其所对应的聚类结构决定了最优类数 ki 。系统演化方法能提供关于所有聚类之间的相对边界距离或可分程度，它适用于明显分离的聚类结构和轻微重叠的聚类结构。

##### 3.2.1.6使用canopy算法进行初始划分

基于Canopy Method的聚类算法将聚类过程分为两个阶段

1、 聚类最耗费计算的地方是计算对象相似性的时候，Canopy Method在第一阶段选择简单、计算代价较低的方法计算对象相似性，将相似的对象放在一个子集中，这个子集被叫做Canopy，通过一系列计算得到若干Canopy，Canopy之间可以是重叠的，但不会存在某个对象不属于任何Canopy的情况，可以把这一阶段看做数据预处理；

2、在各个Canopy内使用传统的聚类方法(如K-means)，不属于同一Canopy的对象之间不进行相似性计算。

#### 3.2.2初始质心得选取

选择适当的初始质心是基本k-means算法的关键步骤。常见的方法是随机的选取初始中心，但是这样簇的质量常常很差。处理选取初始质心问题的一种常用技术是：多次运行，每次使用一组不同的随机初始质心，然后选取具有最小SSE(误差的平方和)的簇集。这种策略简单，但是效果可能不好，这取决于数据集和寻找的簇的个数。

第二种有效的方法是，取一个样本，并使用层次聚类技术对它聚类。从层次聚类中提取k个簇，并用这些簇的质心作为初始质心。该方法通常很有效，但仅对下列情况有效：(1)样本相对较小，例如数百到数千(层次聚类开销较大)；(2) k相对于样本大小较小。

第三种选择初始质心的方法，随机地选择第一个点，或取所有点的质心作为第一个点。然后，对于每个后继初始质心，选择离已经选取过的初始质心最远的点。使用这种方法，确保了选择的初始质心不仅是随机的，而且是散开的。但是，这种方法可能选中离群点。此外，求离当前初始质心集最远的点开销也非常大。为了克服这个问题，通常该方法用于点样本。由于离群点很少(多了就不是离群点了)，它们多半不会在随机样本中出现。计算量也大幅减少。

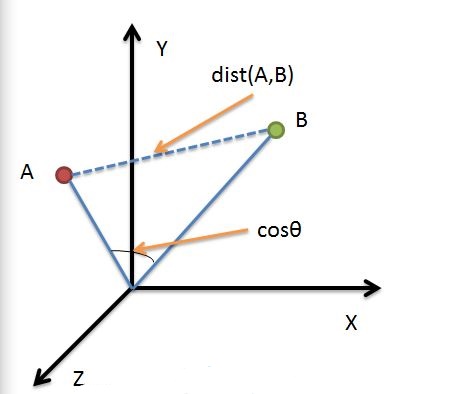
第四种方法就是上面提到的canopy算法。

#### 3.2.3距离得度量

常用的距离度量方法包括：欧几里得距离和余弦相似度。两者都是评定个体间差异大小。

欧氏距离是最常见的距离度量，而余弦相似度则是最常见的相似度度量，很多的距离度量和相似度度量都是基于这两者的变形和衍生

借助三维坐标系来看下欧氏距离和余弦相似度的区别：



从图上可以看出距离度量衡量的是空间各点间的绝对距离，跟各个点所在的位置坐标(即个体特征维度的数值)直接相关；而余弦相似度衡量的是空间向量的夹角，更加的是体现在方向上的差异，而不是位置。如果保持A点的位置不变，B点朝原方向远离坐标轴原点，那么这个时候余弦相似是保持不变的，因为夹角不变，而A、B两点的距离显然在发生改变，这就是欧氏距离和余弦相似度的不同之处。

欧氏距离能够体现个体数值特征的绝对差异，所以更多的用于需要从维度的数值大小中体现差异的分析，如使用用户行为指标分析用户价值的相似度或差异；而余弦相似度更多的是从方向上区分差异，而对绝对的数值不敏感，更多的用于使用用户对内容评分来区分用户兴趣的相似度和差异，同时修正了用户间可能存在的度量标准不统一的问题(因为余弦相似度对绝对数值不敏感)。

因为欧几里得距离度量会受指标不同单位刻度的影响，所以一般需要先进行标准化，同时距离越大，个体间差异越大；空间向量余弦夹角的相似度度量不会受指标刻度的影响，余弦值落于区间[-1,1]，值越大，差异越小。

#### 3.2.4聚类效果评估

K-means是一种非监督学习，没有标签和其他信息来比较聚类结果。但是，我们还是有一些指标可以评估算法的性能。我们已经介绍过类的畸变程度的度量方法。介绍另一种聚类算法效果评估方法称为轮廓系数(Silhouette Coefficient)。轮廓系数是类的密集与分散程度的评价指标。它会随着类的规模增大而增大。彼此相距很远，本身很密集的类，其轮廓系数较大，彼此集中，本身很大的类，其轮廓系数较小。轮廓系数是通过所有样本计算出来的，计算每个样本分数的均值，计算公式如下：

a是每一个类中样本彼此距离的均值，b是一个类中样本与其最近的那个类的所有样本的距离的均值

#### 3.2.5K-means算法流程

输入：聚类个数k，数据集Xmxn。

输出：满足方差最小标准的k个聚类。

(1) 选择k个初始中心点，例如c[0]=X[0] , … , c[k-1]=X[k-1]；

(2) 对于X[0]….X[n]，分别与c[0]…c[k-1]比较，假定与c[i]差值最少，就标记为i；

(3) 对于所有标记为i点，重新计算c[i]={ 所有标记为i的样本的每个特征的均值}；

(4) 重复(2)(3)，直到所有c[i]值的变化小于给定阈值或者达到最大迭代次数。

K-means的时间复杂度：O(tkmn)，空间复杂度：O((m+k)n)。其中，t为迭代次数，k为簇的数目，m为样本数，n为特征数。

**基于以上5点总结一下K-means算法的原理**

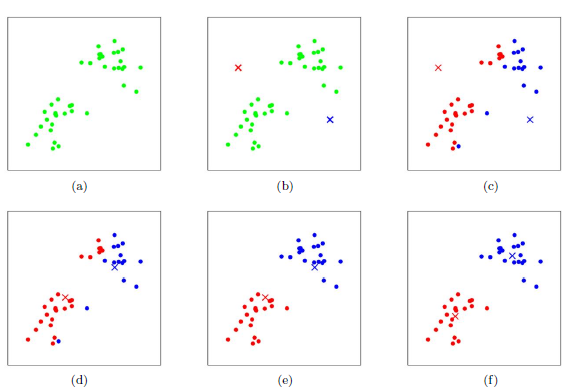
K-means算法的思想很简单，对于给定的样本集，按照样本之间的距离大小，将样本集划分为K个簇。让簇内的点尽量紧密的连在一起，而让簇间的距离尽量的大。

如果用数据表达式表示，假设簇划分为(C1,C2,...Ck)，则我们的目标是最小化平方误差E：

其中μi是簇Ci的均值向量，有时也称为质心，表达式为：

如果我们想直接求上式的最小值并不容易，这是一个NP难的问题，因此只能采用启发式的迭代方法。

K-means采用的启发式方式很简单，用下面一组图就可以形象的描述。



上图a表达了初始的数据集，假设k=2。在图b中，我们随机选择了两个k类所对应的类别质心，即图中的红色质心和蓝色质心，然后分别求样本中所有点到这两个质心的距离，并标记每个样本的类别为和该样本距离最小的质心的类别，如图c所示，经过计算样本和红色质心和蓝色质心的距离，我们得到了所有样本点的第一轮迭代后的类别。此时我们对我们当前标记为红色和蓝色的点分别求其新的质心，如图4所示，新的红色质心和蓝色质心的位置已经发生了变动。图e和图f重复了我们在图c和图d的过程，即将所有点的类别标记为距离最近的质心的类别并求新的质心。最终我们得到的两个类别如图f。

当然在实际K-mean算法中，我们一般会多次运行图c和图d，才能达到最终的比较优的类别。

#### 3.2.6K-means算法优缺点

K-means的主要优点有：

1. 原理比较简单，实现也是很容易，收敛速度快。
2. 聚类效果较优。
3. 算法的可解释度比较强。
4. 主要需要调参的参数仅仅是簇数k。

K-Means的主要缺点有：

1. K值的选取不好把握
2. 对于不是凸的数据集比较难收敛
3. 如果各隐含类别的数据不平衡，比如各隐含类别的数据量严重失衡，或者各隐含类别的方差不同，则聚类效果不佳。
4. 采用迭代方法，得到的结果只是局部最优。
5. 对噪音和异常点比较的敏感。

## 3.3Matlab基本指令语法说明

#### ①k-means

idx = k-means(X,k)

idx = k-means(X,k,Name,Value)

[idx,C] = k-means(**\_\_\_**)

[idx,C,sumd] = k-means(**\_\_\_**)

[idx,C,sumd,D] = k-means(**\_\_\_**)

# 4、实验内容

## 4.1对声音信号进行采集

得到测试声音信号：[manwoman.wav](Test.wav)

## 4.2对采集的语音采用MFCC为特征，基于K-means算法做聚类进行男女声分类

# 5、实验结果

## 5.1对采集的语音采用MFCC为特征，基于K-means算法做聚类进行男女声分类

提取声音信号的MFCC特征（13维的倒谱参数）

代码：

clear all;

[x,fs]=audioread('D:\manwoman.m4a'); %声音的获取

x=filter([1-0.9375],[1],x); %预处理高频预加重，抵消频谱倾斜

wlen=882; %设置窗长

win=hamming(wlen);

inc=320; %设置帧移

xf=enframe(x,wlen,inc)'; %分帧、加汉明窗

n1=length(xf);

n2=fix(wlen/2)+1;

p=24;

bank=v\_melbankm(p,wlen,fs,0,0.5,'m'); %Mel滤波器

bank=full(bank);

bank=bank/max(bank(:)); %归一化Mel滤波器组系数

for k=1:13 %DCT系数

n=0:p-1;

detcoef(k,:)=cos((2\*n+1)\*k\*pi/(2\*p)); %p为滤波器个数

end

for i=1:size(xf,2) %计算mfcc参数

y1=xf(i,:);

s1=y1';

t1=abs(fft(s1));

t1=t1.^2;

c1=detcoef\*log(bank\*t1(1:n2));

m1(i,:)=c1';

end

data=m1(:,1:2)

N=2;

figure

[id,center]=k\_means(data,N);

plot(data(id==1,1),data(id==1,2),'r.');

hold on;

plot(data(id==2,1),data(id==2,2),'b.');

hold on;

plot(center(:,1),center(:,2),'Kx');

grid on;

function [id,center]=k\_means(data,K)

%K-means聚类

%id是表示数据点属于哪一类的标记，center是每个类的聚类中心

%data为输入数据,K为需要分的聚类

% data=datafile();

% K=3;

[m,n]=size(data); %求输入数据点的个数m

id=zeros(m,1);

%随机初始化聚类中心

C=zeros(K,n);

for i=1:K

C(i,:)=data(randi(m,1),:); %随机初始化聚类中心%randi(n,1)产生一个1到n的伪随机整数

end

%C(i)表示第i类的聚类中心

while 1

%分配簇

for x=1:m

for y=1:K

d(y)=norm(data(x,:)-C(y,:)); %计算数据点到每个聚类中心的距离

end

[~,idx]=min(d);

id(x)=idx;

end

%更新聚类中心

new\_C=zeros(K,n);

num=zeros(K);

q=0;

for y=1:K

for x=1:m

if id(x,1)==y

new\_C(y,:)=new\_C(y,:)+data(x,:);

num(y)=num(y)+1;

end

end

new\_C(y,:)=new\_C(y,:)/num(y);

if norm(new\_C(y,:)-C(y,:))<0.001 %判断是否收敛

q=q+1;

end

end

if q==K

break;

else

C=new\_C;

end

end

center=C;

end

结果：

