附加选题报告——用户聚类

第一章 概述

一、背景介绍

电网作为一个超大的复杂系统，必然会产生巨量的数据。在当下数据技术（大数据，分布式计算，数据挖掘等）蓬勃发展的背景下，很容易联想到，这些电网数据所带来的巨大潜在价值。

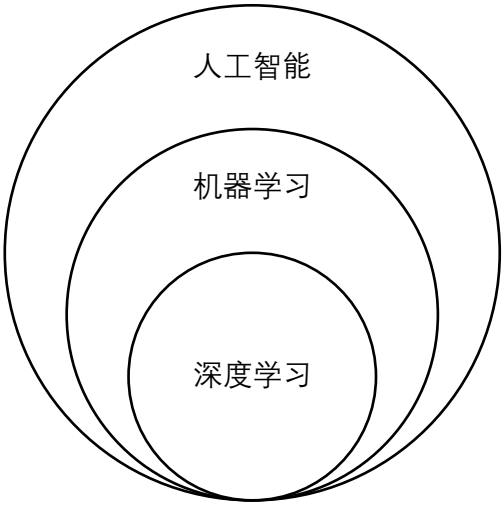
不管是数据挖掘，模式识别，还是机器学习，名称的不同并没有改变这些技术的核心思想：在数据中挖掘信息。人脑所无法处理与提取的数据，经过算法处理后便可大放异彩。

显然，电力系统早已应用起了这些技术。针对用户用电数据，我们可以对其进行特征的提取，然后根据特征对用户群体进行分类，这种聚类带来了大量潜在价值。

二、聚类

机器学习（machine learning）是一门致力于研究通过计算手段，利用经验（数据）来改善系统性能的学科。最早的机器学习算法可以追溯到上个世纪。然而当时的算法效果并不理想，实用性也相对较差，所以并没有获得大量的推广。直到几年前，机器学习的一个分支“深度学习”(deep learning)获得了显著战果（alphaGo, CNN）后,机器学习才重新回到主流视野中，并给人们带来了“人工智能”的理想信念。

以上概念的范畴如下



而在机器学习中，又根据有无数据标记，分“有监督学习”（supervised learning）和“无监督学习”(unsupervised learning)。无监督学习任务中研究最多，应用范围最广的便是“聚类”(clustering)。**此算法试图将数据集中的样本划分为若干个不相交的子集（即“簇”（cluster））。算法旨在寻找数据内在的分布结构。**

第二章 matlab 课题及其实现

一、原始数据生成

题目设想有五类用户，其用电特征都十分明显。

1) 数据储存方式

在大部分机器学习算法中，数据都是以向量的形式储存的，相应的数据集则是以矩阵的形式存起来的。这一存储方式不仅方便了数学推导，也为意外的给了 GPU 一个天然的优势。

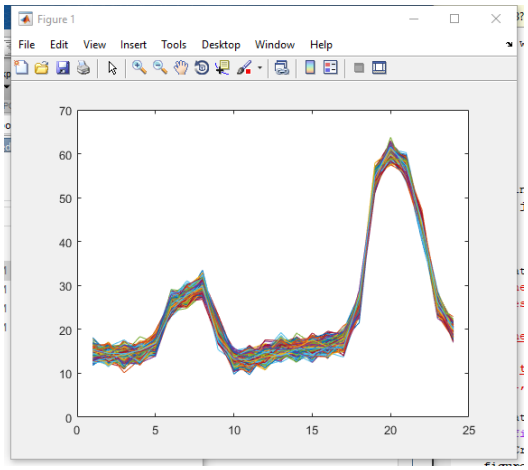
显然，用户的数据应当用一个由 24 个不同时间对应值的向量储存。

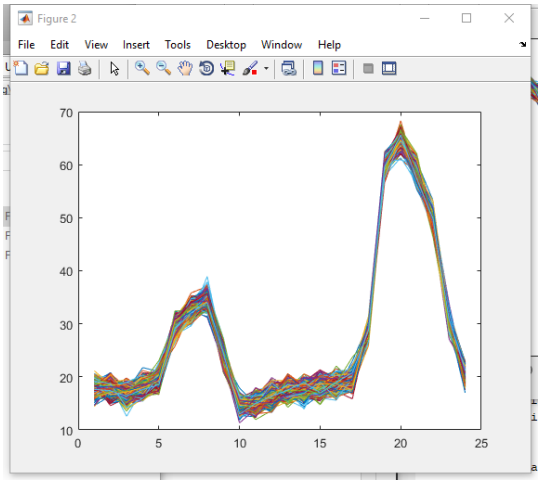
2) 数据生成方式

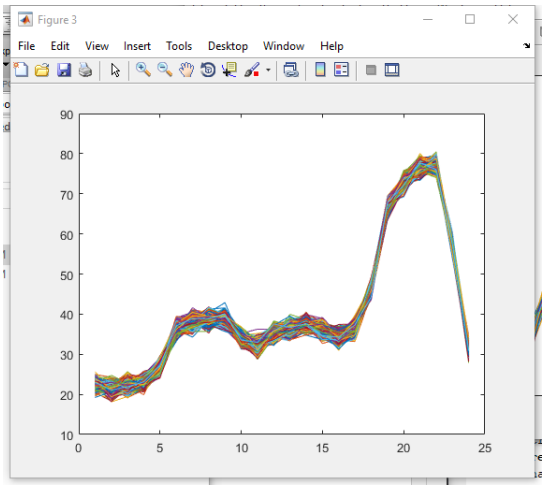
经过与老师的商讨，数据生成的方法如下：

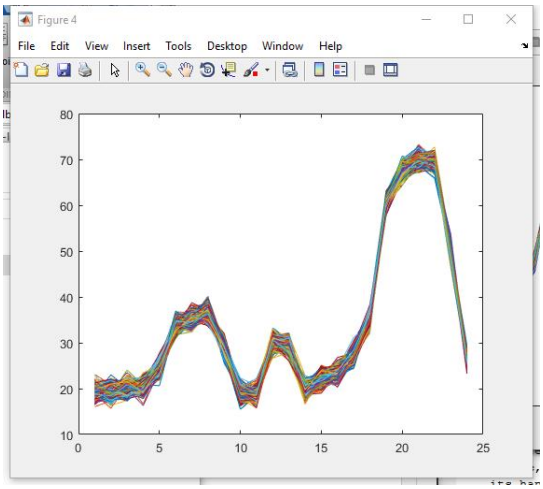
- 首先根据五类用户的特征描述手动创建五个数据
- 对于每个数据向量，叠加由正态分布产生的随机偏差 delta，重复 1000 次

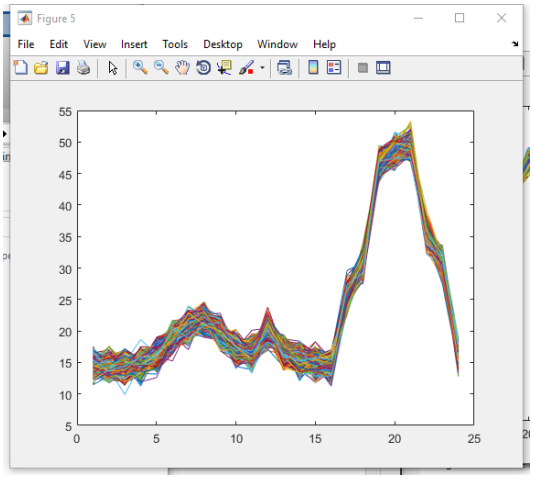
单个用户数据与混合后的数据结果如图

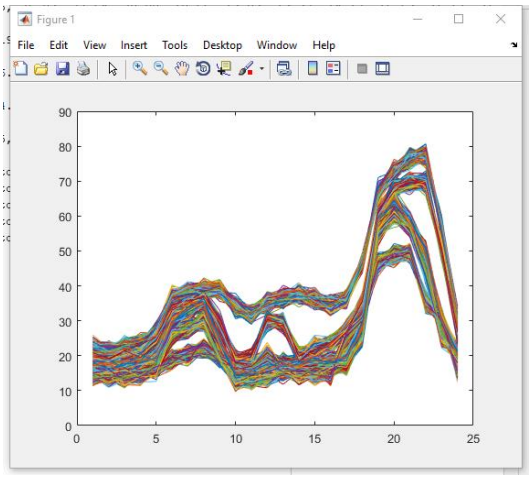












二、聚类算法

题目要求采用的聚类算法是基于原型的 k-mean 聚类算法。通常情况下，算法相对原型进行初始化，然后对原型进行迭代更新求解。

k-mean 求解样本集的最小化平方误差

$$E = \sum_{i=1}^{k} \sum_{x \in C_i} \|x - \mu_i\|_2^2$$

其中样本集D $= \{x_1, x_2, \ldots, x_m\}$, 簇划分为C $= \{C_1, C_2, \ldots, C_k\}$, μ表示样本中心点。然而直接求解 E 的最小化式是一个 NP 难问题，k-mean 算法转而采用贪心策略进行求解，通过迭代进行最优化求解。算法的伪码如下。

输入: 样本集 $D = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_m\}$;
聚类簇数 k.

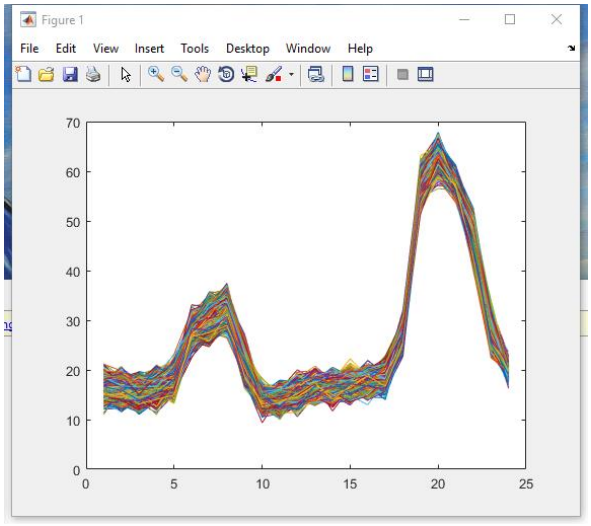
过程:

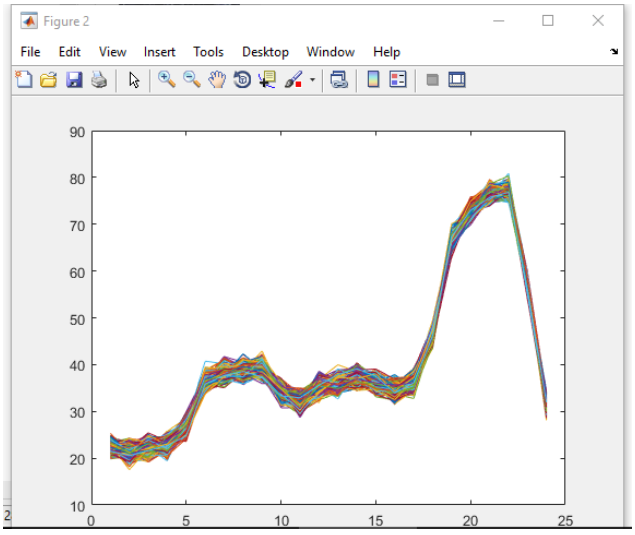
从 D 中随机选择 k 个样本作为初始均值向量 $\{\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, \ldots, \boldsymbol{\mu}_k\}$
repeat
  令 $C_i = \varnothing \ (1 \leqslant i \leqslant k)$
  for $j = 1, 2, \ldots, m$ **do**
    计算样本 $\boldsymbol{x}_j$ 与各均值向量 $\boldsymbol{\mu}_i \ (1 \leqslant i \leqslant k)$ 的距离: $d_{ji} = ||\boldsymbol{x}_j - \boldsymbol{\mu}_i||_2$;
    根据距离最近的均值向量确定 $\boldsymbol{x}_j$ 的簇标记: $\lambda_j = \arg\min_{i \in \{1,2,\ldots,k\}} d_{ji}$;
    将样本 $\boldsymbol{x}_j$ 划入相应的簇: $C_{\lambda_j} = C_{\lambda_j} \bigcup \{\boldsymbol{x}_j\}$;
  end for
  for $i = 1, 2, \ldots, k$ **do**
    计算新均值向量: $\boldsymbol{\mu}_i' = \frac{1}{|C_i|} \sum_{\boldsymbol{x} \in C_i} \boldsymbol{x}$;
    if $\boldsymbol{\mu}_i' \neq \boldsymbol{\mu}_i$ **then**
      将当前均值向量 $\boldsymbol{\mu}_i$ 更新为 $\boldsymbol{\mu}_i'$
    else
      保持当前均值向量不变
    end if
  end for
until 当前均值向量均未更新

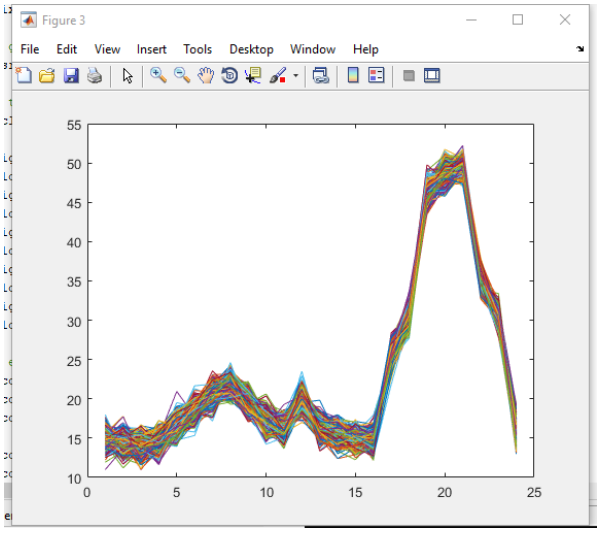
输出: 簇划分 $C = \{C_1, C_2, \ldots, C_k\}$

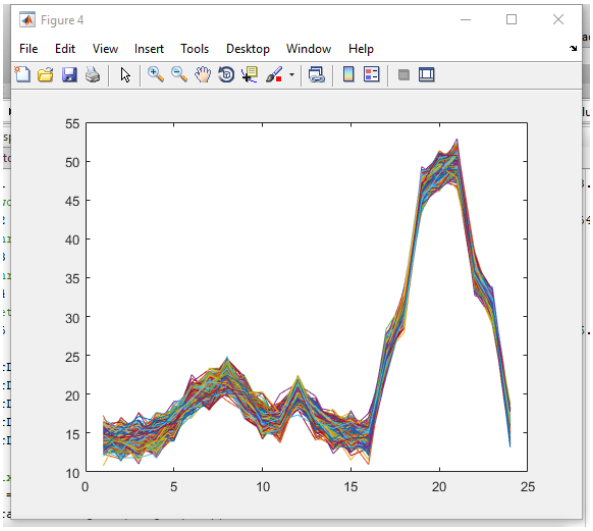
通常为了避免运行时间过长，会给算法设置一个最大运行轮数或者最小调整幅度阈值。然而在实际的算法实现中，发现没有设置这个，也能在 30 次迭代以内完成所有调整。

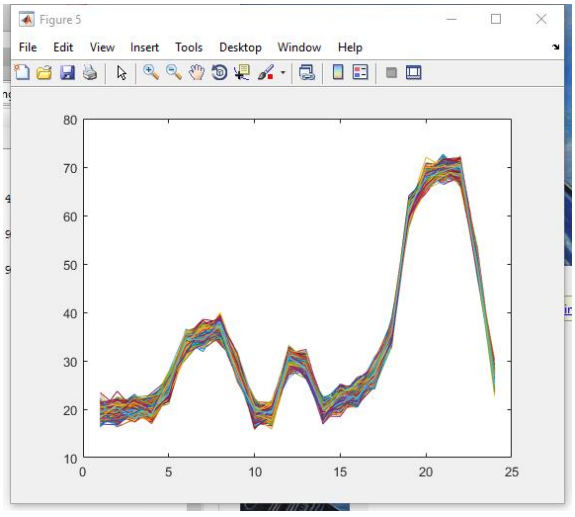
使用 matlab 实现这个算法，最后的分类结果如下











由于理论上每次初始化的样本中心点都是随机的，所以每次聚类的结果都有所不同。而我们确实能看出，这些数据分类都具有明显的规律性。

三、聚类性能度量

性能度量（validity index）是聚类的一大问题。参考周志华教授的《机器学习》西瓜书，性能度量分“外部指标”和“内部指标”两种方法。

1) “外部指标”指讲模型分类结果与专家的参考结果进行比较。我们的题目是可以采用这种方式的，所有的样本在生成时，我们都给其加上了标签。

具体的指标如下

对数据集 $D=\{\boldsymbol{x}_1,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_m\}$, 假定通过聚类给出的簇划分为 $\mathcal{C}=\{C_1, C_2,\ldots,C_k\}$, 参考模型给出的簇划分为 $\mathcal{C}^*=\{C_1^*,C_2^*,\ldots,C_s^*\}$. 相应地, 令 $\boldsymbol{\lambda}$ 与 $\boldsymbol{\lambda}^*$ 分别表示与 $\mathcal{C}$ 和 $\mathcal{C}^*$ 对应的簇标记向量. 我们将样本两两配对考虑, 定义

$$a=|SS|,\quad SS=\{(\boldsymbol{x}_i,\boldsymbol{x}_j)\mid \lambda_i=\lambda_j,\lambda_i^*=\lambda_j^*,i<j)\},\tag{9.1}$$

$$b=|SD|,\quad SD=\{(\boldsymbol{x}_i,\boldsymbol{x}_j)\mid \lambda_i=\lambda_j,\lambda_i^*\neq\lambda_j^*,i<j)\},\tag{9.2}$$

$$c=|DS|,\quad DS=\{(\boldsymbol{x}_i,\boldsymbol{x}_j)\mid \lambda_i\neq\lambda_j,\lambda_i^*=\lambda_j^*,i<j)\},\tag{9.3}$$

$$d=|DD|,\quad DD=\{(\boldsymbol{x}_i,\boldsymbol{x}_j)\mid \lambda_i\neq\lambda_j,\lambda_i^*\neq\lambda_j^*,i<j)\},\tag{9.4}$$

其中集合 SS 包含了在 $\mathcal{C}$ 中隶属于相同簇且在 $\mathcal{C}^*$ 中也隶属于相同簇的样本对, 集合 SD 包含了在 $\mathcal{C}$ 中隶属于相同簇但在 $\mathcal{C}^*$ 中隶属于不同簇的样本对, ……由于每个样本对 $(\boldsymbol{x}_i,\boldsymbol{x}_j)$ $(i<j)$ 仅能出现在一个集合中, 因此有 $a+b+c+d=m(m-1)/2$ 成立.

基于式(9.1)~(9.4)可导出下面这些常用的聚类性能度量外部指标:

- Jaccard 系数(Jaccard Coefficient, 简称 JC)

$$\mathrm{JC}=\frac{a}{a+b+c}.\tag{9.5}$$

- FM 指数(Fowlkes and Mallows Index, 简称 FMI)

$$\mathrm{FMI}=\sqrt{\frac{a}{a+b}\cdot\frac{a}{a+c}}.\tag{9.6}$$

- Rand 指数(Rand Index, 简称 RI)

$$\mathrm{RI} = \frac{2(a+d)}{m(m-1)} . \tag{9.7}$$

显然, 上述性能度量的结果值均在 $[0,1]$ 区间, 值越大越好.

我们使用 matlab 实现了外部评估方式，并为外部指标函数 exIndex 预留了接口，供用户选择评估指标种类（Jaccard, FMI & Rand）

2) 内部指标是根据计算划分结果的内聚程度来度量的。具体算法如下：

考虑聚类结果的簇划分 $\mathcal{C} = \{C_1, C_2, \ldots, C_k\}$, 定义

$$\mathrm{avg}(C) = \frac{2}{|C|(|C|-1)} \sum_{1 \leqslant i < j \leqslant |C|} \mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j) , \tag{9.8}$$

$$\mathrm{diam}(C) = \max_{1 \leqslant i < j \leqslant |C|} \mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j) , \tag{9.9}$$

$$d_{\min}(C_i, C_j) = \min_{\boldsymbol{x}_i \in C_i, \boldsymbol{x}_j \in C_j} \mathrm{dist}(\boldsymbol{x}_i, \boldsymbol{x}_j) , \tag{9.10}$$

$$d_{\mathrm{cen}}(C_i, C_j) = \mathrm{dist}(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j) , \tag{9.11}$$

其中, $\mathrm{dist}(\cdot,\cdot)$ 用于计算两个样本之间的距离; $\boldsymbol{\mu}$ 代表簇 C 的中心点 $\boldsymbol{\mu} = \frac{1}{|C|} \sum_{1 \leqslant i \leqslant |C|} \boldsymbol{x}_i$. 显然, $\mathrm{avg}(C)$ 对应于簇 C 内样本间的平均距离, $\mathrm{diam}(C)$ 对应于簇 C 内样本间的最远距离, $d_{\min}(C_i, C_j)$ 对应于簇 C_i 与簇 C_j 最近样本间的距离, $d_{\mathrm{cen}}(C_i, C_j)$ 对应于簇 C_i 与簇 C_j 中心点间的距离.

基于式(9.8)~(9.11)可导出下面这些常用的聚类性能度量内部指标:

- DB 指数(Davies-Bouldin Index, 简称 DBI)

$$\mathrm{DBI} = \frac{1}{k} \sum_{i=1}^{k} \max_{j \neq i} \left(\frac{\mathrm{avg}(C_i) + \mathrm{avg}(C_j)}{d_{\mathrm{cen}}(\boldsymbol{\mu}_i, \boldsymbol{\mu}_j)} \right) . \tag{9.12}$$

- Dunn 指数(Dunn Index, 简称 DI)

$$\mathrm{DI} = \min_{1 \leqslant i \leqslant k} \left\{ \min_{j \neq i} \left(\frac{d_{\min}(C_i, C_j)}{\max_{1 \leqslant l \leqslant k} \mathrm{diam}(C_l)} \right) \right\} . \tag{9.13}$$

显然, DBI 的值越小越好, 而 DI 则相反, 值越大越好.

我们使用 matlab 实现了内部评估方式，并为函数 intIndex 提供了接口以供用户选择指标类型。

3) 数学工具箱

在聚类算法中有个很重要的主题是“距离计算”。样本点之间采用的计算方法是广义的距离函数“闵科夫斯基距离”。即：

$$dist_{mk}(x_i, x_j) \overset{\text{def}}{=} \left(\sum_{u=1}^{n} |x_{iu} - x_{ju}|^p \right)^{\frac{1}{p}}$$

当 p = 2 时，这个距离就是我们平时所用的欧式距离。

在 k-mean 算法及其评估中中也大量应用了闵科夫斯基距离的概念。

我们用 matlab 实现了这些前提的数学工具，并设计给函数 minkoDist 设计了接口，以便用户选择 p 的值。

4) 最后的测试中，发现我们的指标并没有相当理想。与深度学习方法比较起来，指标相对还是较低的。部分测试结果截图如下

```
>> testModel
iterations =
     1
iterations =
     2
a =
     0
b =
     0
c =
     0
score1 =
    0.1111
a =
     0
b =
     0
c =
     0
score2 =
    0.2001
a =
     0
b =
     0
c =
     0
score3 =
    0.6803
score4 =
    0.5716
score5 =
    0.9064
```

第三章 总结

整个 matlab 算法的实现到结果处理，我们都是通过数学公式全自主实现，最后的结果可观，也是最让我们欣慰的。

在整个程序设计流程中我们采用了非常好的程序设计规范，采用了规范化的命名，排版等，为代码提供了较高的可维护性。

我们的问题主要集中在算法的评估结果并不理想上。由于时间与精力问题，我们没有继续深究原因。

其次我们注意到，由于初始化数据的随机性，每次聚类的效果差异都是很大的。我们可以采用在图形界面手动选取偏中心的初始化数据点的方式进行改进。然而时间与能力不足，最终没有进行实践。

此外，我们的聚类方法采用了最基本的 k-mean 算法。其实还可以尝试更多的聚类方法如高斯混合聚类，密度聚类等。我们的模型也对用户开放了很多可定制的接口（如评估方法，样本数量, 闵氏距离的 p 值等），这些也会对模型的效果产生较大影响。当然，调参也是机器学习从业者的一大“乐趣”。

总之，我们不仅完成了项目要求，还尽量将我们的工作设计为了一个可复用的框架（即 toolbox）