Partial Kmeans

Wenxi Zhang and Norman Matloff

What is k-means clustering

- K-means clustering is a type of unsupervised learning.
- Partition n observations into k clusters in which each observation belongs to the cluster with the nearest cluster centroid.
- What to do with missing data?

House votes dataset

This data set includes votes for each of the U.S. House of Representatives Congresspeople on the 16 key votes. Different types of votes are simplified to yes(y), no(n), and unknown(NA).

```
##     V2 V3 V4   V5   V6 V7 V8 V9 V10
## 1    n  y  n    y    y  y  n  n   n
## 2    n  y  n    y    y  y  n  n   n
## 3 <NA>  y  y <NA>    y  y  n  n   n
## 4    n  y  y    n <NA>  y  n  n   n
## 5    y  y  y    n    y  y  n  n   n
## 6    n  y  y    n    y  y  n  n   n

##   V11  V12  V13 V14 V15 V16  V17
## 1   y <NA>    y   y   y   n    y
## 2   n    n    y   y   y   n <NA>
## 3   n    y    n   y   y   n    n
## 4   n    y    n   y   n   n    y
## 5   n    y <NA>   y   y   y    y
## 6   n    n    n   y   y   y    y
```

- Pretend we have no information on parties
- Let's try k=2 for demonstration
- See if we can predict the correct group

```
Train_house <- PartialKmeans::gen_train_test(house.votes,seed=78
Valid_house <- PartialKmeans::gen_train_test(house.votes,78)$tes
house_model <- PartialKmeans::Partial_km(m = Train_house[,2:17],
house_test <- PartialKmeans::fitted.test(Valid_house[,-1],2,hous
house_test_fitted <- factor(house_test$fitted_values,labels = c(
Valid_house[,1][1:30]
```

```
##  [1] 1 1 2 1 1 1 2 1 2 2 1 1 1 1 1 2 2 2 1 1 1 2 2 1 1 1 1 1 1 1
```

```
house_test_fitted[1:30]
```

```
##  [1] 2 2 2 1 1 1 2 1 2 2 1 1 1 1 1 2 2 2 1 1 1 2 2 1 1 2 2 2 2 1
## Levels: 2 1
```

```
PartialKmeans::test_accuracy(Valid_house[,1],house_test_fitted)
```

```
## [1] 0.71875
```

Why PartialKmeans

- a easy remedy for datasets of missing values

```
(example<-matrix(c(2,4,NA,5,6,8,NA,5,8),nrow=3,byrow = TRUE))
```

```
##      [,1] [,2] [,3]
## [1,]    2    4   NA
## [2,]    5    6    8
## [3,]   NA    5    8
```

```
c1<-c(2,3,4)
c2<-c(6,2,4)
d11<- ((2-2)^2+(4-3)^2)/2
d12<- ((2-6)^2+(4-2)^2)/2
d21<- ((5-2)^2+(6-3)^2+(8-4)^2)/3
d22<- ((5-6)^2+(6-2)^2+(8-4)^2)/3
d31<-((5-3)^2+(8-4)^2)/2
d32<- ((5-2)^2+(8-4)^2)/2
```

```
(d<-matrix(c(d11,d12,d21,d22,d31,d32),nrow=3,byrow = TRUE))
```

```
##            [,1] [,2]
## [1,]  0.50000 10.0
## [2,] 11.33333 11.0
## [3,] 10.00000 12.5
```

```
matrix(c(2,4,NA,NA,5,8),nrow=2,byrow = TRUE)
```

```
##      [,1] [,2] [,3]
## [1,]    2    4   NA
## [2,]   NA    5    8
```

```
(newc1<-c(2,(4+5)/2,8))
```

```
## [1] 2.0 4.5 8.0
```

```
(newc2<-c(5,6,8))
```

```
## [1] 5 6 8
```

Given a training set $x^{(1)}, \dots, x^{(m)}$

1. initialize cluster centroids $\mu_1, \mu_2 \dots \mu_j \in R^n$ randomly
2. Repeat until convergence/number of iterations usr defined

   for each i,

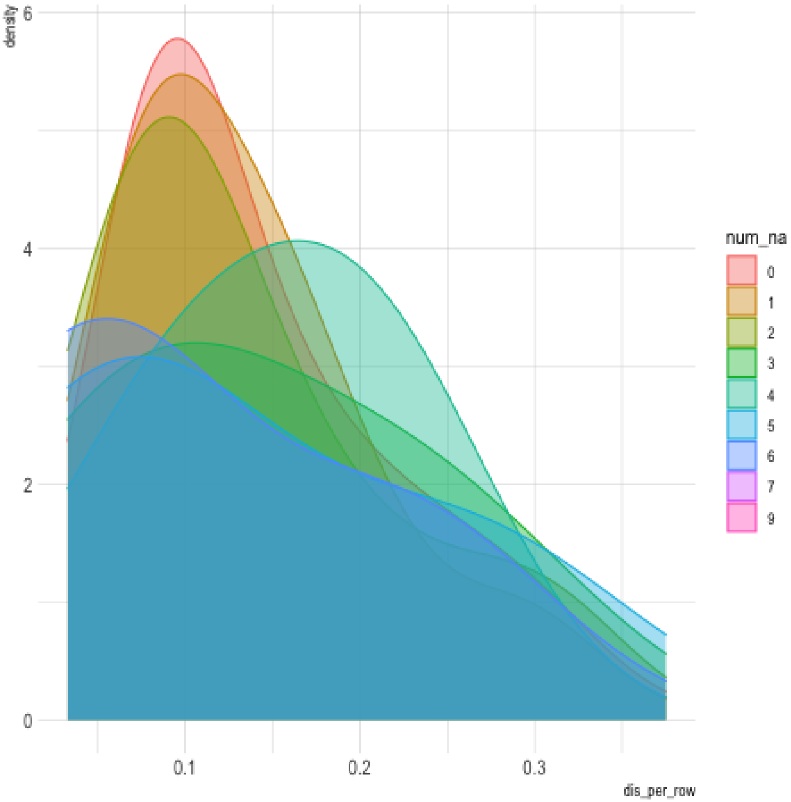
   $$d_i = \frac{||(x_i - \mu_j)[intact\ index]||^2}{\#x_i[intact\ index]}$$

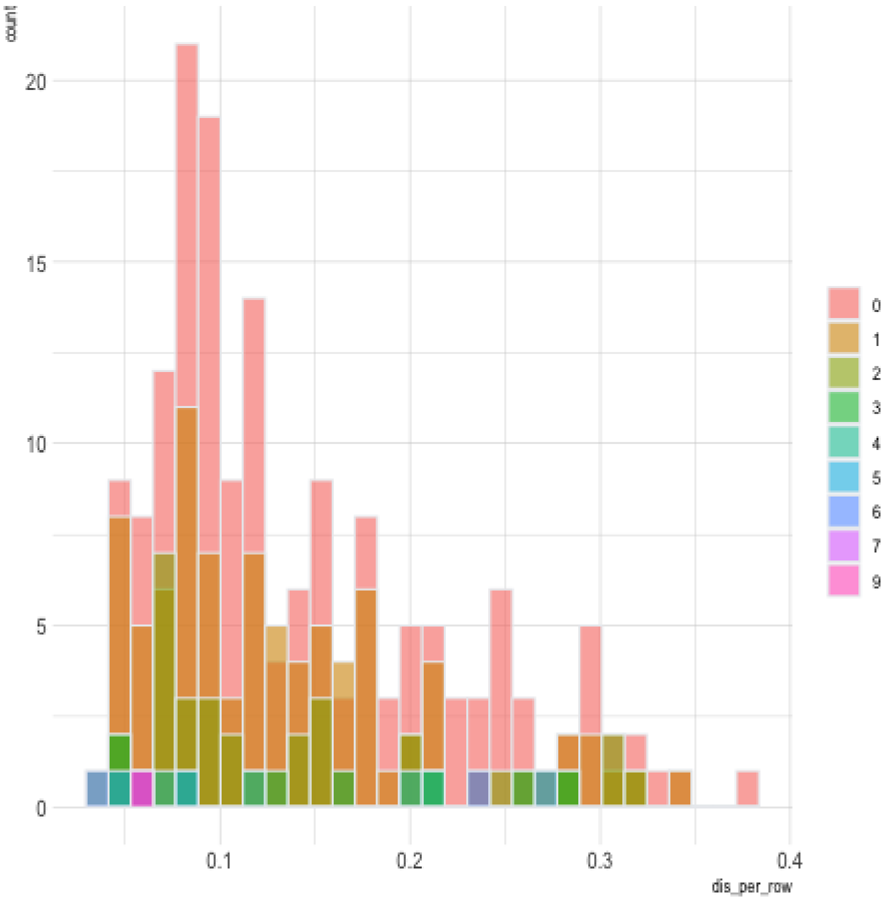
   $$c_i = \arg\min_j d_i^2$$

   for each j,

   $$\mu_j = \frac{\sum_{i=1}^{m} 1\{c_i = j\} x_i[intact\ index]}{\sum_{i=1}^{m} 1\{c_i = j\} \#x_i[intact\ index]}$$

some plots




count
20
15
10
5
0
0.1
0.2
0.3
0.4
dis_per_row
0
1
2
3
4
5
6
7
9

Performance comparison wrt %missing

- ClustImpute
- Synthetic data using a dietary survey of patients with irritable bowel syndrome (IBS)
- A data frame with 400 Instances and 43 attributes
- Attributes are intake of a list of foods

- Data has two groups: healthy-group vs. the IBS-patients
- Pretend we have no knowledge about the groups
- Given that we choose k = 2
- How well would PartialKmeans predict the correct group, relative to ClustImpute?

	Partialkmeans	ClustImpute
10%missing	0.9416667	0.9416667
20%missing	0.9250000	0.9250000
30%missing	0.9333333	0.9333333
40%missing	0.9166667	0.9000000
50%missing	0.8916667	0.8000000
60%missing	0.8833333	0.8333333
70%missing	0.8750000	0.7333333
80%missing	0.8416667	0.7083333
90%missing	0.7583333	0.6833333