K-means Clustering

Rebecca C. Steorts, Duke University

STA 325, Chapter 10 ISL

Agenda

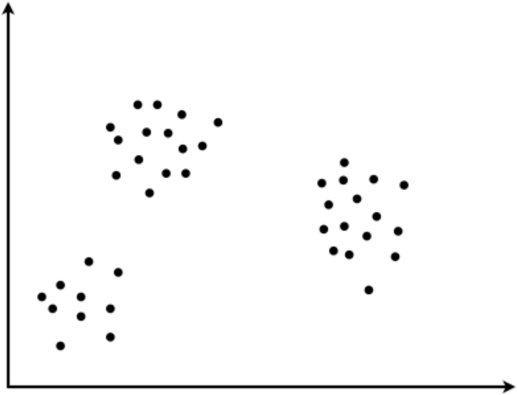
- Clustering
- Examples
- K-means clustering
- Notation
- Within-cluster variation
- K-means algorithm
- Example
- Limitations of K-means

Clustering

What is clustering? Why do we use it?

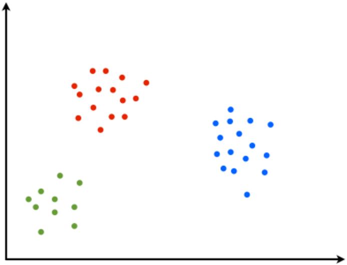
Clustering

Clustering/Partition



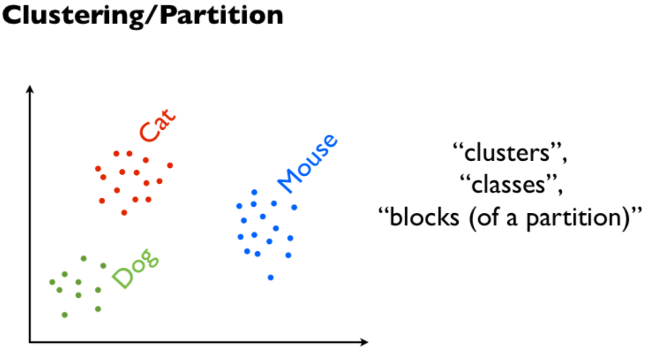
Clustering

Clustering/Partition



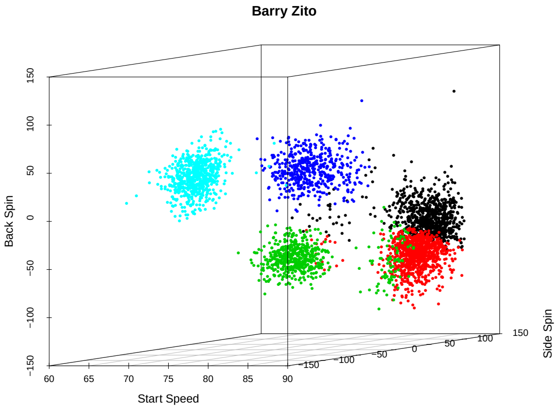
"clusters",
"classes",
"blocks (of a partition)"

Clustering



Note: we don't typically know the labels or the class groups (cat, dog, mouse) when we are clustering data points. Hence, one goal is to look for separation of the data points into groups.

Clustering of Baseball Pitches

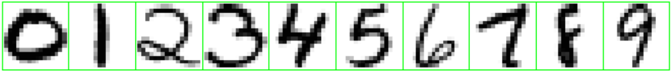


Inferred meaning of clusters: black – fastball, red – sinker, green – changeup, blue – slider, light blue – curveball

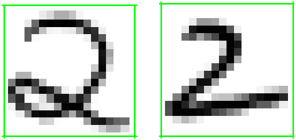
(Example from Mike Pane, former student at CMU)

Clustering versus Classification

- In classification, we have data for which the class labels are known,
- Try to learn what differentiates these groups (i.e., classification function) to properly classify future data



- In clustering, we look at data, where groups are unknown and undefined,
- Try to learn the groups themselves, as well as what differentiates them



Clustering algorithms

We will cover two clustering algorithms that are very simple to understand, visualize, and use.

The first is the k-means algorithm.

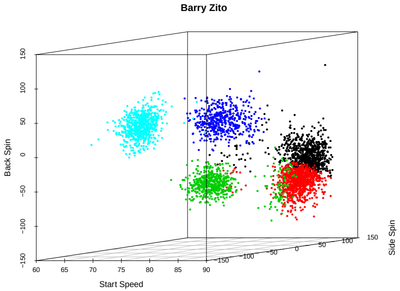
The second is hierarchical clustering.

K-means clustering algorithm

- K-means clustering: simple approach for partitioning a dataset into K distinct, non-overlapping clusters.
  1. To perform K-means clustering: specify the desired number of clusters K.
  2. Then the K-means algorithm will assign each observation to exactly one of the K clusters.

Notation

- Observations $X_1, \dots X_n$
- dissimilarites $d(X_i, X_j)$.
- Let K be the number of clusters (fixed).
- A clustering of points $X_1, \dots X_n$ is a function C that assigns each observation X_i to a group $k \in \{1, \dots K\}$



Within-cluster variation

- $C(i) = k$ means that observation X_i is assigned to group k
- $|C_k|$ is the number of points in group k
- Let $d_{ij} = d(X_i, X_j)$, for some distance function d.

The within-cluster variation is defined as

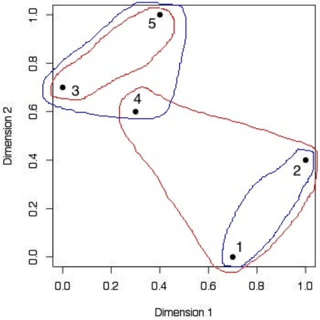
$$W = \sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{C(i)=k,\, C(j)=k} d_{ij}$$

Smaller W is better

Simple example

Here $n = 5$ and $K = 2$,
$X_i \in \mathbb{R}^2$ and $d_{ij} = \|X_i - X_j\|_2^2$

	1	2	3	4	5
1	0	0.25	0.98	0.52	1.09
2	0.25	0	1.09	0.53	0.72
3	0.98	1.09	0	0.10	0.25
4	0.52	0.53	0.10	0	0.17
5	1.09	0.72	0.25	0.17	0



- Red clustering: $W_{\text{red}} = (0.25 + 0.53 + 0.52)/3 + 0.25/2 = 0.56$
- Blue clustering:
  $W_{\text{blue}} = 0.25/2 + (0.10 + 0.17 + 0.25)/3 = 0.30$

(Tip: dist function in R)

Finding the best group assignments

Smaller W is better, so why don't we just directly find the clustering C that minimizes W?

Finding the best group assignments

Smaller W is better, so why don't we just directly find the clustering C that minimizes W?

Problem: doing so requires trying all possible assignments of the n points into K groups. The number of possible assignments is

$$A(n, K) = \frac{1}{K!} \sum_{k=1}^{K} (-1)^{K-k} \binom{K}{k} k^n$$

Note that $A(10, 4) = 34,105$, and $A(25, 4) \approx 5 \times 10^{13}$

See, Jain and Dubes (1998), "Algorithms for Clustering Data"

Most problems we look at are going to have way more than $n = 25$ observations, and potentially more than $K = 4$ clusters too (but $K = 4$ is not unrealistic)

Finding the best group assignments

How do we get around this?

We will end up making an approximation. Let's walk through all the details now of K-means clustering.

K-means clustering

- K-means is a simple way to paritition a data set into K distinct, non-overlapping clusters.
- To perform K-means clustering, we must

1. first specify the desired number of clusters K
2. then the K-means algorithm will assign each observation to exactly one of the K clusters.

How does this work?

Notation

Let n denote the number of data points in our data set.

Let

$$C_1, C_2, \ldots C_k$$

denoting sets containing the indices of the observations in each cluster. (This means that each data point is in only one cluster C_j).

This means that these sets satisfy two properties:

1. $$C_1 \cup C_2 \cup \cdots C_K = \{1, \ldots, n\}.$$

   This means that each observations belongs to at least one of the K clusters.

2. $$C_k \cap C_{k'} = \varnothing$$

   for all $k \neq k'$. This means the clusters are non-overlapping and so no observation belongs to more than one cluster.

Intuition

As an example, if the ith observation is in cluster k then data point $i \in C_k$.

We think of k-means being a good clustering algorithm when the within-cluster variation is as small as possible.

What is this?

The within cluster variation

The within cluster variation of cluster C_k is a measure of $W(C_k)$ of the amount by which the observations within a cluster differ from each other.

Mathematically, we want to solve the following optimization problem:

$$\min_{C_1,\ldots,C_K} \sum_{k=1}^{K} W(C_k)$$

In words, this means that we want to partition the data points into clusters such that the total within-cluster variation summed over all K clusters is as small as possible.

This seems reasonable, but how do we define the within-cluster variation $W(C_k)$? Thoughts?

The within cluster variation

There are many possible ways to define this concept, but by far the most common choice involves squared Euclidean distance.

Can you think of why we use this based on previous discussions that we have had in class?

The within cluster variation

Thus, we define

$$W(C_k) = \frac{1}{|C_k|} \sum_{i,i'} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2$$

where $|C_k|$ denotes the number of observations in the kth cluster.

In words, the within-cluster variation for the kth cluster is the sum of all of the pairwise squared Euclidean distances between the observations in the kth cluster, divided by the total number of observations in the kth cluster.

Back to the optimization problem

We return now to the optimization problem that defines K-means clustering (under the Euclidean norm):

$$\min_{C_1,\ldots,C_K} \sum_{k=1}^{K} W(C_k) = \min_{C_1,\ldots,C_K} \{\sum_{k=1}^{K} \frac{1}{|C_k|} \sum_{i,i'} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2\} \qquad (1)$$

Now, we would like to find an algorithm to solve equation 1.

That is, we want a method to partition the observations into K clusters such that the objective of equation 1 is minimized.

K-means continued

This is in fact a very difficult problem to solve precisely, since there are almost K^n ways to partition n data points into K clusters.

This is a very large number unless K and n are both small. (In practice, they are not)!

Fortunately, a very simple algorithm can be shown to provide a local optimum — a pretty good solution — to the K-means optimization problem.

K-means clustering algorithm

1. Randomly assign a number, from 1 to K, to each of the observations. These serve as initial cluster assignments for the observations.
2. Iterate until the clustering algorithm stops changing:
   - a For each of the K clusters, compute the centroid. The kth cluster centriod is the vector of the p feature averages for the observations in the cluster k.
   - b Assign each observation to the cluster whose centroid is closest (where closest is defined using Euclidean distance).

Note: $\bar{x}_{kj} = \frac{1}{|C_k|} \sum_{i \in C_k} x_{ij}$ is the average for feature j in cluster C_k.

K-means clustering algorithm

The above algorithm is guaranteed to decrease the value of the objective function at each step.

Why is this true?

$$\frac{1}{|C_k|} \sum_{i,i' \in C_k} \sum_{j=1}^{p} (x_{ij} - x_{i'j})^2 = 2 \sum_{i \in C_k} \sum_{j=1}^{p} (x_{ij} - \bar{x}_{kj})^2, \qquad (2)$$

where $\bar{x}_{kj} = \frac{1}{|C_k|} \sum_{i \in C_k} x_{ij}$ is the mean for data point (feature) j in cluster C_k.

K-means clustering algorithm

- In Step 2(a), the cluster means for each data point are the constants that minimize the sum of squared deviations
- In Step 2(b), reallocating the data points can only improve the the within sum of squares.
- This means that as the algorithm is run, the clustering obtained will continually improve until the result no longer changes and the objective of equation 1 will never increase!
- When the result no longer changes, we reach a local optimum.

K-means clustering algorithm

Since the algorithm reaches a local optimum and not a global optimum, the results obtained will depend on the initial (random) cluster assignment of each observation in Step 1.

- Due to this, it's crucial to run the algorithm many times and from multiple (random) starting points.
- One should select the best solution, namely, the one where the objective function is the smallest.

Example

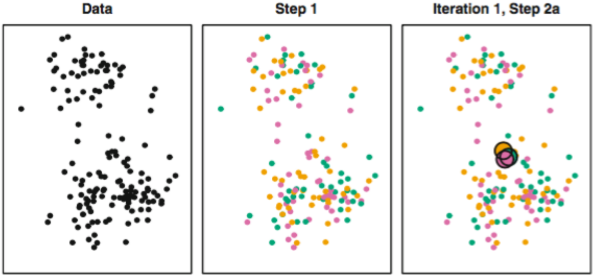


Figure 1: Top left: the data is shown. Top center: in Step 1 of the algorithm, each observation is randomly assigned to a cluster. Top right: in Step 2(a), the cluster centroids are computed. These are shown as large colored disks. Initially the centroids are almost completely overlapping because the initial cluster assignments were chosen at random.

Example (continued)

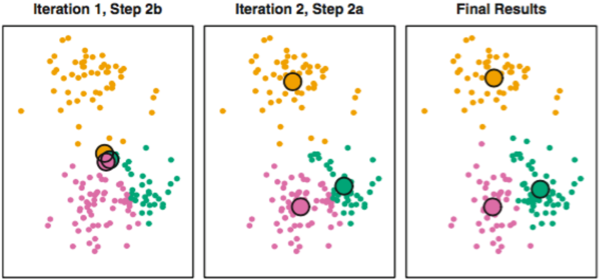


Figure 2: Bottom left: in Step 2(b), each observation is assigned to the nearest centroid. Bottom center: Step 2(a) is once again performed, leading to new cluster centroids. Bottom right: the results obtained after ten iterations.

Example 2

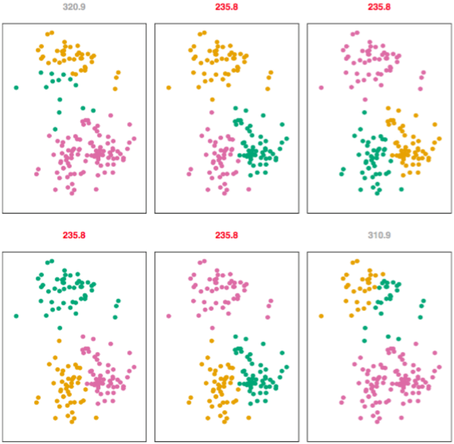


Figure 3: K-means clustering performed six times; K = 3, each time with a different random assignment of the observations in Step 1 of the K-means algorithm. Above each plot is the value of the objective (10.11). Those labeled in red all achieved the same best solution, with an objective value of 235.8.

Application

We begin with a simple simulated example in which there truly are two clusters in the data: the first 25 observations have a mean shift relative to the next 25 observations.

```r
# simulated some data
set.seed(2)
x=matrix(rnorm(50*2), ncol=2)
x[1:25,1]=x[1:25,1]+3
x[1:25,2]=x[1:25,2]-4
```

Perform K-means clustering, $K = 2$

```
km.out=kmeans(x,centers= 2,nstart=20)
```

- centers is the number of clusters K.
- nstart tells us how many sets should be chosen.

Perform K-means clustering, $K = 2$

The cluster assignments of the 50 observations can be found by the following:

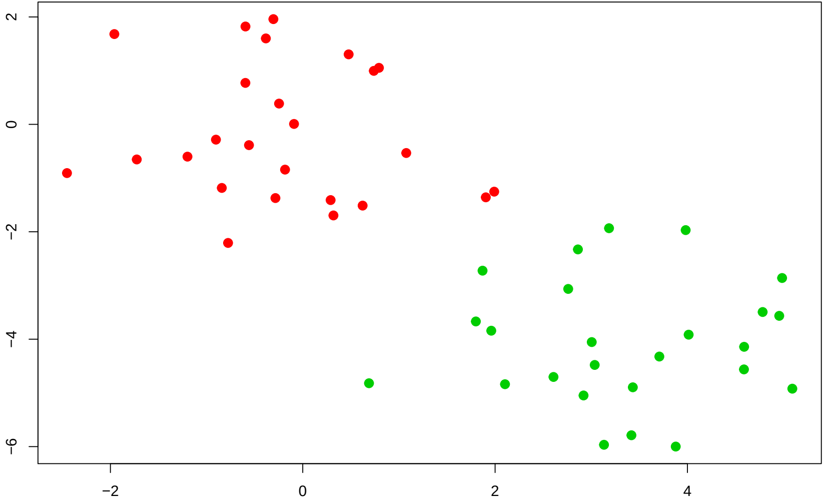
```
km.out$cluster
```

```
##  [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 1
## [36] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

The K-means clustering perfectly separated the observations into two clusters even though we did not supply any group information to kmeans

Plotting with cluster assignment

```
plot(x, col=(km.out$cluster +1), main="", xlab="", ylab="", pch=20, cex=2)
```



Here the observations can be easily plotted because they are two-dimensional. If there were more than two variables then we could instead perform PCA and plot the first two principal components score vectors.

Other values of K

Here, we alread knew the value of K because we simulated the data and in general we don't know K, so we need to play around with this value.

What happens if we look at $K = 3$.

$K = 3$ for simulated example

```
set.seed(4)
km.out=kmeans(x,3,nstart=20)
km.out
```

```
## K-means clustering with 3 clusters of sizes 10, 23, 17
##
## Cluster means:
##         [,1]        [,2]
## 1  2.3001545 -2.69622023
## 2 -0.3820397 -0.08740753
## 3  3.7789567 -4.56200798
##
## Clustering vector:
##  [1] 3 1 3 1 3 3 3 1 3 1 3 1 3 1 3 1 3 3 3 3 3 1 3 3 3 2 2 2 2 2 2 2 2 2 2
## [36] 2 2 2 2 2 2 2 2 1 2 1 2 2 2 2
##
## Within cluster sum of squares by cluster:
## [1] 19.56137 52.67700 25.74089
##  (between_SS / total_SS =  79.3 %)
##
## Available components:
##
## [1] "cluster"      "centers"      "totss"        "withinss"
## [5] "tot.withinss" "betweenss"    "size"         "iter"
## [9] "ifault"
```

Go do this on your own and explain the results. What happens now with $K = 3$. (Take about 5 minutes to do this).

More about k-means

- To run the kmeans() function in R with multiple initial cluster assignments, we use the nstart argument.
- If a value of nstart greater than one is used, then K-means clustering will be performed using multiple random assignments in Step 1 of Algorithm 10.1, and the kmeans() function will report only the best results.

Here we compare using nstart$=1$ to nstart$=20$.

Varying the nstart value

```r
set.seed(3)
km.out=kmeans(x,3,nstart=1)
km.out$tot.withinss
```

```
## [1] 104.3319
```

```r
km.out=kmeans(x,3,nstart=20)
km.out$tot.withinss
```

```
## [1] 97.97927
```

- km.out$tot.withinss is the total within-cluster sum of squares, which we seek to minimize by performing K-means clustering
- The individual within-cluster sum-of-squares are contained in the vector km.out$withinss.

Recommended settings

- Recommend always running K-means clustering with a large value of nstart, such as 20 or 50, since otherwise an undesirable local optimum may be obtained.
- Make sure you always set a random seed as well so that you can reproduce your results.

Difficult questions posed by K-means

- The main question that is posed by k-means is how many clusters K should we choose?
- Anytime we make such a choice regarding K, this can have a strong impact on the results obtained.
- In practice, we try several different choices, and look for the one with the most useful or interpretable solution.
- With these methods, there is no single right answer—any solution that exposes some interesting aspects of the data should be considered.

Validating the Clusters Obtained

- Any time clustering is performed on a data set we will find clusters.
- But we really want to know whether the clusters that have been found represent true subgroups in the data, or whether they are simply a result of clustering the noise.
- For instance, if we were to obtain an independent set of observations, then would those observations also display the same set of clusters?
- There exist a number of techniques for assigning a p-value to a cluster in order to assess whether there is more evidence for the cluster than one would expect due to chance.
- However, there has been no consensus on a single best approach. More details can be found in Hastie et al. (2009).

Other Considerations in Clustering

- Both K-means and hierarchical clustering will assign each observation to a cluster.
- However, sometimes this might not be appropriate.
- For instance, suppose that most of the observations truly belong to a small number of (unknown) subgroups, and a small subset of the observations are quite different from each other and from all other observations.
- Then since K-means and hierarchical clustering force every observation into a cluster, the clusters found may be heavily distorted due to the presence of outliers that do not belong to any cluster.
- Mixture models are an attractive approach for accommodating the presence of such outliers.
- These amount to a soft version of K-means clustering, and are described in Hastie et al. (2009).