

STAT562 Lecture 19: Unsupervised Learning

Beidi Qiang

SIUE

So far in the semester, we mostly concerns supervised learning methods such as regression and classification.

- ▶ In the supervised learning setting, we typically have access to a set of features X_1, \dots, X_p , measured on n observations, and a response Y also measured on the same observations. The goal is then to predict Y using X_1, \dots, X_p .
- ▶ In unsupervised learning, we have only a set of features X_1, \dots, X_p measured on n observations. The goal is to discover interesting things about the measurements on X_1, \dots, X_p , such as patterns and clusters.

The Challenge of Unsupervised Learning

Supervised learning has very well developed set of tools at your disposal and a clear way to assess the quality of the results obtained
However, unsupervised learning is often more subjective, and hard to assess.

- ▶ Unsupervised learning is often performed as part of an exploratory data analysis
- ▶ There is no simple goal for the analysis, such as prediction of a label
- ▶ There is no universally accepted mechanism for performing validation

Unsupervised learning are of growing importance in a number of fields:

- ▶ Targeted Ads: Identify groups of shoppers with similar browsing and purchase histories
- ▶ Recommendation Systems: identify users with similar preferences
- ▶ Anomaly detection: Identify rare items, events or observations which differ significantly from the majority of the data

Clustering refers to a very broad set of techniques for finding subgroups, or clusters, in a data set.

- ▶ We seek to partition data into distinct groups.
- ▶ Observations within each group should be quite similar to each other
- ▶ Observations in different groups should be quite different from each other.
- ▶ We must first answer the question: what it means for two or more observations to be similar or different.

- ▶ **Marketing:** We may have a large number of measurements (e.g.household income, occupation, and so forth) for a large number of people. Our goal is to perform market segmentation by identifying subgroups of people who might be more likely to purchase a particular product.

Two best-known Clustering Approaches

- ▶ K-means clustering: partition the observations into a pre-specified K distinct clusters.
- ▶ Hierarchical clustering: No pre-specified number of clusters. Create a tree-like visual representation of the observations, that allows us to view clusterings obtained for each possible number of clusters.

A simple approach for partitioning a data set into K distinct, non-overlapping clusters.

- ▶ K , the number of clusters, is pre-defined.
- ▶ Each observation belongs to exactly one of the K clusters.
- ▶ The clusters are non- overlapping
- ▶ The idea behind K-means clustering is that a good clustering is one for which the within-cluster variation is as small as possible.

- ▶ Let C_k be the indices of the observations in cluster k .
- ▶ We seek to minimize the total within-cluster variation, $W(C_k)$, i.e.,

$$\text{minimize } \sum_{k=1}^K W(C_k)$$

- ▶ A common choice is of measuring how observations are differ from each other is the squared Euclidean distance:

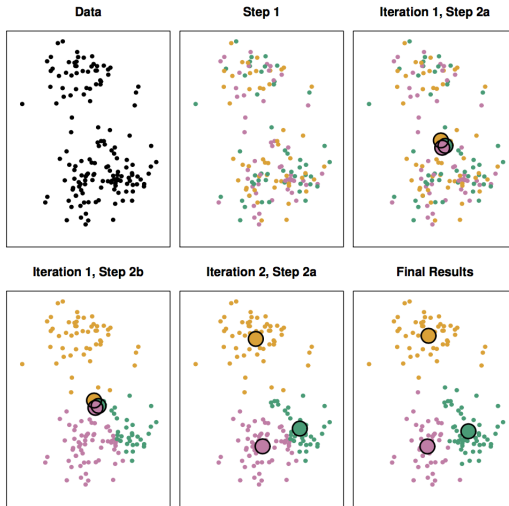
$$W(C_k) = \frac{1}{|C_k|} \sum_{i, i' \in C_k} \sum_{j=1}^p (x_{ij} - x_{i'j})^2$$

- ▶ This is in fact a very difficult optimization problem to solve precisely, since there are almost K^n ways to partition. We would like to find an algorithm find a solution.

Algorithm: K-Means Clustering

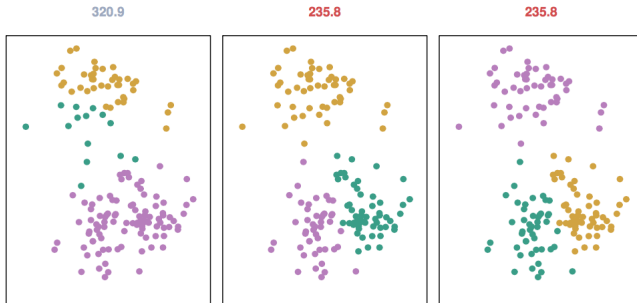
- ▶ Randomly assign an initial cluster, from 1 to K , to each of the observations.
- ▶ Iterate the following until the cluster assignments stop changing:
 - (1) For each of the K clusters, compute the cluster centroid, which is the vector of the p feature means for the observations in the cluster.
 - (2) Assign each observation to the cluster whose centroid is closest (using Euclidean distance).

K-means




- ▶ The algorithm of K-means is guaranteed to decrease the total within-cluster variation at each step.
- ▶ As the algorithm is run, the clustering obtained will continually improve until the result no longer changes.
- ▶ K-means algorithm finds a local rather than a global optimum. So the results obtained will depend on the initial (random) cluster.
- ▶ It is important to run the algorithm multiple times from different random initial configurations.

K-means with Different Initial Clusters



Working Example

```
R 4.3.1 · ~/ 
```

```
> x=c(1,2,3,5,8,9,10)
> A0=c(1,2,8,9)
> B0=c(3,5,10)
> centroid0=c(mean(A0),mean(B0))
> centroid0
[1] 5 6
> A1=c(1,2,3,5)
> B1=c(8,9,10)
> centroid1=c(mean(A1),mean(B1))
> centroid1
[1] 2.75 9.00
>
> kmeans(x,2)
K-means clustering with 2 clusters of sizes 4, 3

Cluster means:
[,1]
1 2.75
2 9.00

Clustering vector:
[1] 1 1 1 1 2 2 2

Within cluster sum of squares by cluster:
[1] 8.75 2.00
(between_SS / total_SS = 86.2 %)
```

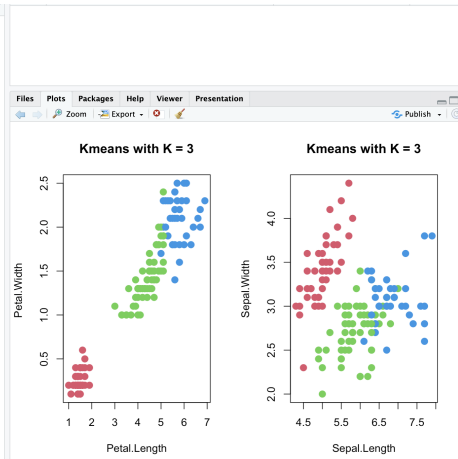
Implement in R

```
R 4.3.1 ~ /  
> x=iris[,1:4]  
> km.iris=kmeans(x, 3)  
> par(mfrow = c(1, 2))  
> plot(x[,3:4], col = (km.iris$cluster + 1), main = "Kmeans with K = 3",  
xlab = "Petal.Length", ylab = "Petal.Width", pch = 20, cex = 2)  
> plot(x[,1:2], col = (km.iris$cluster + 1), main = "Kmeans with K = 3",  
xlab = "Sepal.Length", ylab = "Sepal.Width", pch = 20, cex = 2)  
>  
> km.iris  
K-means clustering with 3 clusters of sizes 50, 62, 38  
  
Cluster means:  
  Sepal.Length Sepal.Width Petal.Length Petal.Width  
1    5.006000    3.428000    1.462000    0.246000  
2    5.901613    2.748387    4.393548    1.433871  
3    6.850000    3.073684    5.742105    2.071053  
  
Clustering vector:  
[1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
[26] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
[51] 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 2 2 2 2 2 2 2  
[76] 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 2 2 2 2 2 2 2  
[101] 3 2 3 3 3 3 2 3 3 3 3 3 2 2 3 3 3 3 3 3 3 3 2 3 3 2 3 3 2 3 3 3  
[126] 3 2 3 3 3 3 3 2 3 3 3 3 2 3 3 3 3 2 3 3 3 2 3 3 3 2 3 3 3 2
```

Within cluster sum of squares by cluster:
[1] 15.15100 39.82097 23.87947
(between_SS / total_SS = 88.4 %)

Available components:

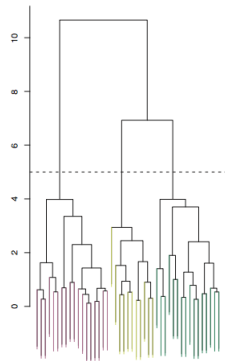
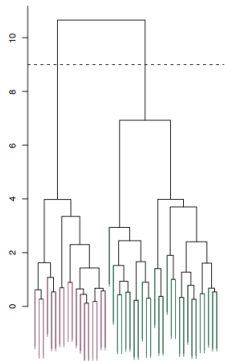
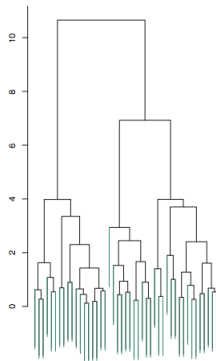
```
[1] "cluster"      "centers"      "totss"  
[4] "withinss"     "tot.withinss" "betweenss"  
[7] "size"         "iter"         "ifault"
```



Hierarchical clustering is an alternative clustering approach which does not require that we commit to a particular choice of K .

- ▶ Hierarchical Clustering results in an attractive tree-based representation, called a dendrogram.
- ▶ Most common type of hierarchical clustering: bottom-up. The dendrogram is built starting from the leaves and combining clusters up to the trunk.

Dendrogram



- ▶ Each leaf of the dendrogram represents one of the observations.
- ▶ As we move up the tree, some leaves begin to fuse into branches. and as we move higher up the tree, branches themselves fuse, either with leaves or other branches.
- ▶ For any two observations, we can look for the point in the tree where branches containing those two observations are first fused. The height (vertical axis) of this fusion indicates how different the two observations are.
- ▶ We make a horizontal cut across the dendrogram to identify clusters. Cutting at different height allow us to obtain different number of clusters.

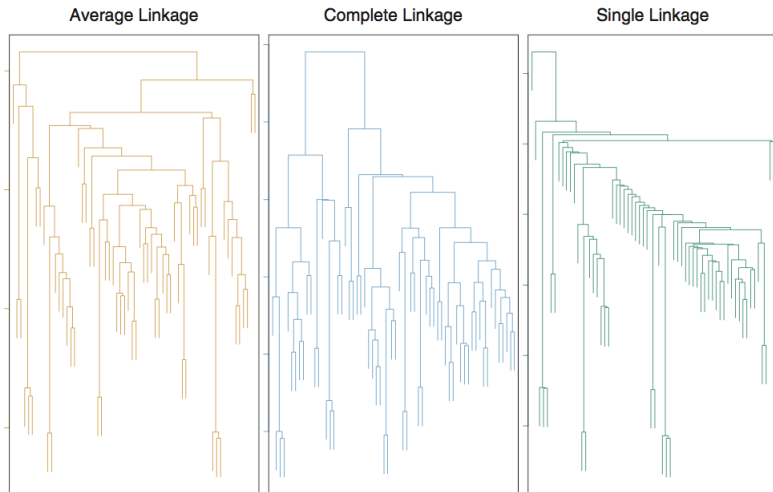
The Hierarchical Clustering Algorithm

- ▶ Start with each observation as its own cluster. Define a measure (such as Euclidean distance) of dissimilarities.
- ▶ In each iteration,
 - Examine all pairwise inter-cluster dissimilarities among the clusters and identify the pair of clusters that are least dissimilar. Fuse these two clusters. The dissimilarity between these two clusters indicates the height at which the fusion should be placed.
 - Compute the new pairwise inter-cluster dissimilarities among the remaining clusters.

We use linkage to define the dissimilarity between two groups of observations. The common choices of linkage are:

- ▶ **Complete:** Compute all pairwise dissimilarities between the observations in cluster A and the observations in cluster B, and record the largest of these dissimilarities.
- ▶ **Single:** Compute all pairwise dissimilarities, and record the smallest of these dissimilarities.
- ▶ **Average:** Compute all pairwise dissimilarities, and record the average of these dissimilarities.
- ▶ **Centroid:** Compute dissimilarity between the centroid (mean) for cluster A and the centroid for cluster B.

Dendrogram typically depends quite strongly on the type of linkage used.



The choice of dissimilarity measure

The choice of dissimilarity measure also has a strong effect on the resulting dendrogram.

- ▶ Euclidean distance as a common dissimilarity measure, but we can also use correlation-based distance.
- ▶ Two observations to be similar if their features are highly correlated.
- ▶ Correlation-based distance focuses on the shapes of observation profiles rather than their magnitudes.
- ▶ In addition to selecting the dissimilarity measure, one must also consider whether or not the variables should be scaled: the choice of units can will greatly affect the dissimilarity measure obtained.

- ▶ The term hierarchical refers to the fact that clusters obtained using hierarchical clustering are nested.
- ▶ This means if we'd like to obtain K clusters by cutting the dendrogram, the result will be nested in clusters obtained by cutting at a higher level (resulting $K-1$ clusters).
- ▶ If this is not a desired assumption, K means will likely give better results, since the clusters obtained are not nested.

Example

R 4.3.1 · - /

```
> x=c(1,2,3,5,8,9,10)
```

```
>
```

```
> hc.complete=hclust(dist(x), method="complete")
```

```
> hc.single=hclust(dist(x), method="single")
```

```
> hc.average=hclust(dist(x), method="average")
```

```
> par(mfrow=c(1,3))
```

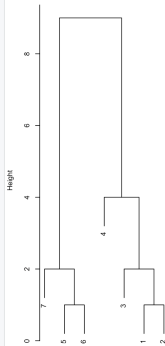
```
> plot(hc.complete,main="Complete Linkage")
```

```
> plot(hc.single,main="Single Linkage")
```

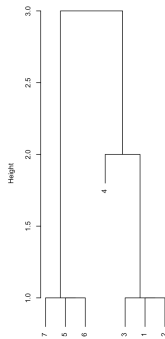
```
> plot(hc.average,main="Average Linkage")
```

```
>
```

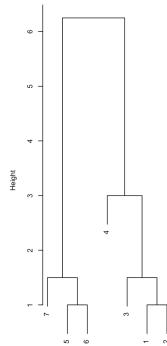
Complete Linkage



Single Linkage



Average Linkage



Implement in R

```
R 4.3.1 ~/  
> x=iris[,1:4]  
> hc.iris=hclust(dist(x), method="complete")  
> cls=cutree(hc.iris,3)  
> par(mfrow = c(1, 2))  
> plot(x[,3:4], col = (cls + 1), main = "hierarchical", xlab = "Petal.Length", ylab = "Petal.Width", pch = 20, cex = 2)  
> plot(x[,1:2], col = (cls + 1), main = "hierarchical", xlab = "Sepal.Length", ylab = "Sepal.Width", pch = 20, cex = 2)  
>  
> #You may scale the data first  
> x.scaled=scale(x)  
>  
>  
> #Using correlation as measure of dissimilarity  
> dd=as.dist(1-cor(t(x)))  
> hc.iris=hclust(dd, method="complete")  
> cls=cutree(hc.iris,3)  
> plot(x[,3:4], col = (cls + 1), main = "hierarchical", xlab = "Petal.Length", ylab = "Petal.Width", pch = 20, cex = 2)  
> plot(x[,1:2], col = (cls + 1), main = "hierarchical", xlab = "Sepal.Length", ylab = "Sepal.Width", pch = 20, cex = 2)  
>
```

