MagDataAnalysis2019

Hierarchical clustering

1. Agglomerative/Divisive clustering
2. Ward clustering
3. Deriving Ward criterion
4. Nearest Neighbor clustering
5. Maximum/Minimum Spanning Tree; Prim Algorithm
6. Divisive NN Clustering

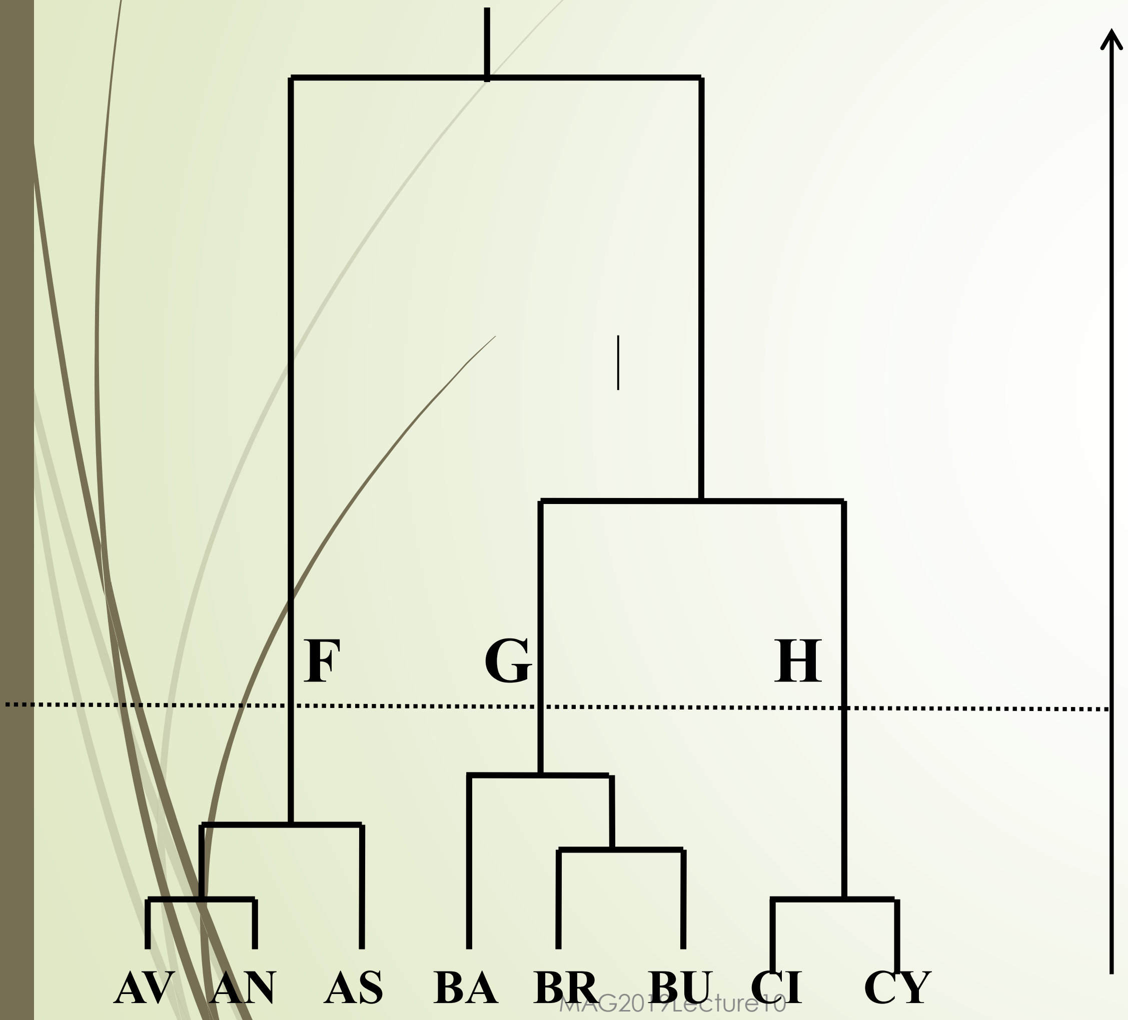
MDA subgroup
Home Work 2019: Deadline

A file with HomeWork project tasks is posted in LMS Materials.

Your report of the homework must reach Instructor at bmirkin@hse.ru by the end of 2 December 2019 (till morning of 3 December).

Reports submitted after this deadline but before the end of 12 December will be penalized by 20% off the mark. No reports are accepted after 12 December.

Cluster Hierarchy: Binary rooted tree



- Over data set as a leaf set
- Interior nodes: clusters of leaves; say F={AV,AN,AS}
- Height function defined at all nodes

h(leaf)=0,

if $t \subset s$, then $h(t)<h(s)$

MagDataAnalysis2019

Hierarchical clustering

1. Agglomertive clustering
2. Ward clustering
3. Deriving Ward criterion
4. Nearest Neighbor clustering
5. Maximum/Minimum Spanning Tree; Prim Algorithm
6. Divisive NN Clustering

Two types of algorithms:

agglomerative (bottom-up)

divisive (up-down)

Two types of criteria (among many others):

Square error

Nearest Neighbor

Agglomerative clustering, 1

Step 1: Start –

Trivial partition of the set of objects represented by their indices in N singletons:

Partition S={{1}, {2}, …, {i}, …, {N}},

(Dis)similarity function D=(d(i,j)), i,j= 1, 2,…, N (either distances or similarities from the object-to-feature data, or raw network interaction data)

Agglomerative clustering, 2

General step: given m part partition S ={S1, S2,…, Sm} and between-cluster dis(similarity) function d(s,t) (s,t=1,…, m),

- **G1:** Find s*, t* minimizing dissimilarity d(s,t) (maximizing similarity) – a costly operation
- **G2:** Merge clusters Ss* and St* to form Ss*t*=Ss* ∪ St*
- **G3:** Compute (dis)similarity between Ss*t* and every other cluster Su to form a new (dis)similarity matrix
- **G4:** Define and compute value of height h(Ss*t*)

- **Test:** Stopping condition. If Yes, stop; output the hierarchy. If not, m:= m-1 and go to G1.

Agglomerative clustering, 3

AgglClus Algorithms may differ by only this:

- **G3:** Compute (dis)similarity between Ss*t* and every other cluster Su to form a new (dis)similarity matrix

Two most popular versions:

- **Nearest neighbor** (according to dis(similarity) between the nearest points from clusters)
- **Ward algorithm** (according to change in the square error criterion)

In both agglomerative and divisive approaches

MagDataAnalysis2019

Hierarchical clustering

1. Agglomertive clustering
2. **Ward clustering**
3. Deriving Ward criterion
4. Nearest Neighbor clustering
5. Maximum/Minimum Spanning Tree; Prim Algorithm
6. Divisive NN Clustering

- **Ward algorithm,**
both divisive and agglomerative,
is based on the so-called Ward distance

- Nearest neighbor divisive algorithm is based on Minimum (Maximum) Spanning Tree (MST)

Ward distance between clusters S_k and S_l (to be derived further on)

$$wd(S_k, S_l) = \frac{N_k N_l}{N_k + N_l} d(c_k, c_l)$$

This combines distance between cluster centers c_k, c_l and a factor depending on the distribution of objects between the clusters: the smaller the difference in sizes N_k and N_l, the larger the value of the factor.
At Ward divisive clustering, a cluster is split in two parts maximizing wd. At Ward agglomerative clustering, two clusters are merged to minimize wd. In both cases, a balanced partition is preferred.

MagDataAnalysis2019

Hierarchical clustering

1. Agglomertive clustering
2. Ward clustering
3. Deriving Ward criterion
4. Nearest Neighbor clustering
5. Maximum/Minimum Spanning Tree; Prim Algorithm
6. Divisive NN Clustering

Derivation of Ward distance, 1:

$$Wd(S_k, S_l) = W(S(k,l), c^{kl}) - W(S, c) \;=\; (*)$$

$$\sum_{i\in S_k}\sum_{v\in V}(y_{iv} - c_{k\cup l,v})^2$$

$$-\sum_{i\in S_k}\sum_{v\in V}(y_{iv} - c_{kv})^2 + \sum_{i\in S_l}\sum_{v\in V}(y_{iv} - c_{k\cup l,v})^2$$

$$-\sum_{i\in S_l}\sum_{v\in V}(y_{iv} - c_{lv})^2.$$

Since $c_{k\cup l,v} = c_{kv} + N_l(c_{lv} - c_{kv})/(N_k + N_l) = c_{lv} + N_k(c_{kv} - c_{lv})/(N_k + N_l)$

and $(a+b)^2 = a^2+b^2+2ab$:

- $\sum_{i\in S_k}\sum_{v\in V}(y_{iv} - c_{k\cup l,v})^2 =$

$\sum_{i\in S_k}\sum_{v\in V}(y_{iv} - c_{kv})^2 +$

$$\sum_{i\in S_k}\sum_{v\in V}\left(\frac{N_l}{N_k+N_l}\right)^2 (c_{kv} - c_{lv})^2 +$$

$2\sum_{i\in S_k}\sum_{v\in V}\frac{N_l}{N_k+N_l}(y_{iv} - c_{kv})(c_{kv} - c_{lv})$

As proven above,

- $\sum_{i\in S_k}\sum_{v\in V}(y_{iv}-c_{k\cup l,v})^2 =$

$\sum_{i\in S_k}\sum_{v\in V}(y_{iv}-c_{kv})^2 +$

$$\sum_{i\in S_k}\sum_{v\in V}\left(\frac{N_l}{N_k+N_l}\right)^2 (c_{kv}-c_{lv})^2 +$$

$\boldsymbol{2\sum_{i\in S_k}\sum_{v\in V}\frac{N_l}{N_k+N_l}(y_{iv}-c_{kv})(c_{kv}-c_{lv})}$

The last item (in bold) =0 because $\sum_{i\in S_k}(y_{iv}-c_{kv})=0.$ The first item is part of W(S,c): to be annihilated by the subtraction in (*).

With a similar trick at S_l,

W(S(k,l), c^kl) – W(S, c)=

$$\sum_{v\in V} N_k\left(\frac{N_l}{N_k+N_l}\right)^2 (c_{kv}-c_{lv})^2 + \sum_{v\in V} N_l\left(\frac{N_k}{N_k+N_l}\right)^2 (c_{lv}-c_{kv})^2 = $$

$$=$$

$$\frac{N_k N_l}{N_k+N_l}\sum_{v\in V}(c_{kv}-c_{lv})^2$$

because

$$N_k\left(\frac{N_l}{N_k+N_l}\right)^2 + N_l\left(\frac{N_k}{N_k+N_l}\right)^2 = \frac{N_k N_l^2+N_l N_k^2}{(N_k+N_l)^2} = \frac{N_k N_l}{N_k+N_l},$$

q.e.d.

MagDataAnalysis2019

Hierarchical clustering

1. Agglomertive clustering
2. Ward clustering
3. Deriving Ward criterion
4. Nearest Neighbor clustering

Maximum/Minimum Spanning Tree; Prim Algorithm

6. Divisive NN Clustering

NN clustering relates to the graph-theoretic concept of Minimum Spanning Tree (MST), 1

- Given a dissimilarity matrix, it can be represented by a weighted graph.
- A tree is a subgraph with no cycles
- A spanning tree is a tree whose node set coincides with the set of all objects
- The length of a tree is the sum of weights of its edges
- The minimum spanning tree is a spanning tree of maximum length.
- If the data is a similarity matrix, we look for a maximum spanning tree.

NN clustering relates to the graph-theoretic concept of Minimum Spanning Tree (MST), 2

NN agglomerative clustering and NN divisive clustering over a (dis)similarity matrix

is equivalent to

agglomerative or divisive clustering

over its Min/Max spanning tree.

Prim's algorithm for MST: Building MST T by adding nodes one-by-one (greedy)

1. Initialization.

Start with tree T consisting of an arbitrary node $i \in I$ with no edges.

2. Tree update.

Find $j \in I\text{-}T$ maximizing a_{ij} over all $i \in T$ and $j \in I\text{-}T$. Add j and edge $\{i,j\}$ with the maximal a_{ij} to T.

3. Stop-condition.

If $I\text{-}T = \varnothing$, halt and output tree T. Otherwise, go to 2.

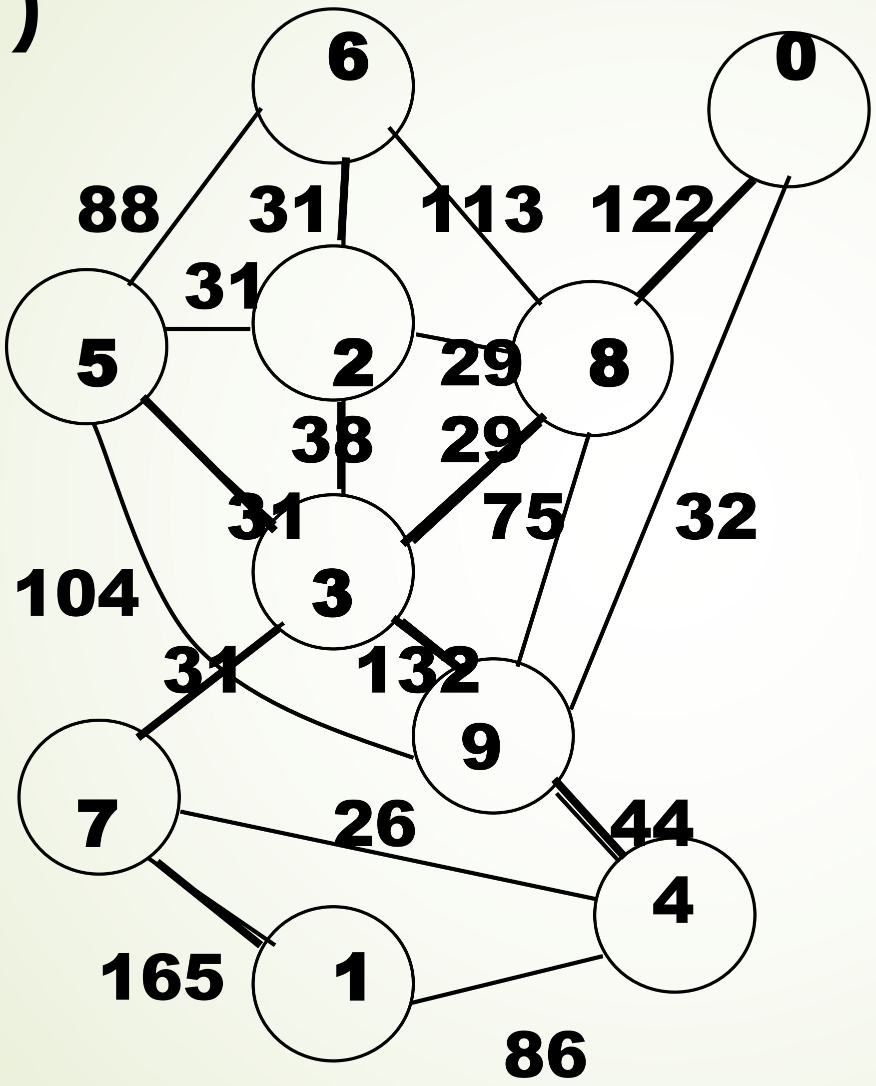
Quest: What MST is built with this algorithm: Maximal or Minimal?

Example: Digits 0, 1, 2,…, 9 confusion data (symmetrized)

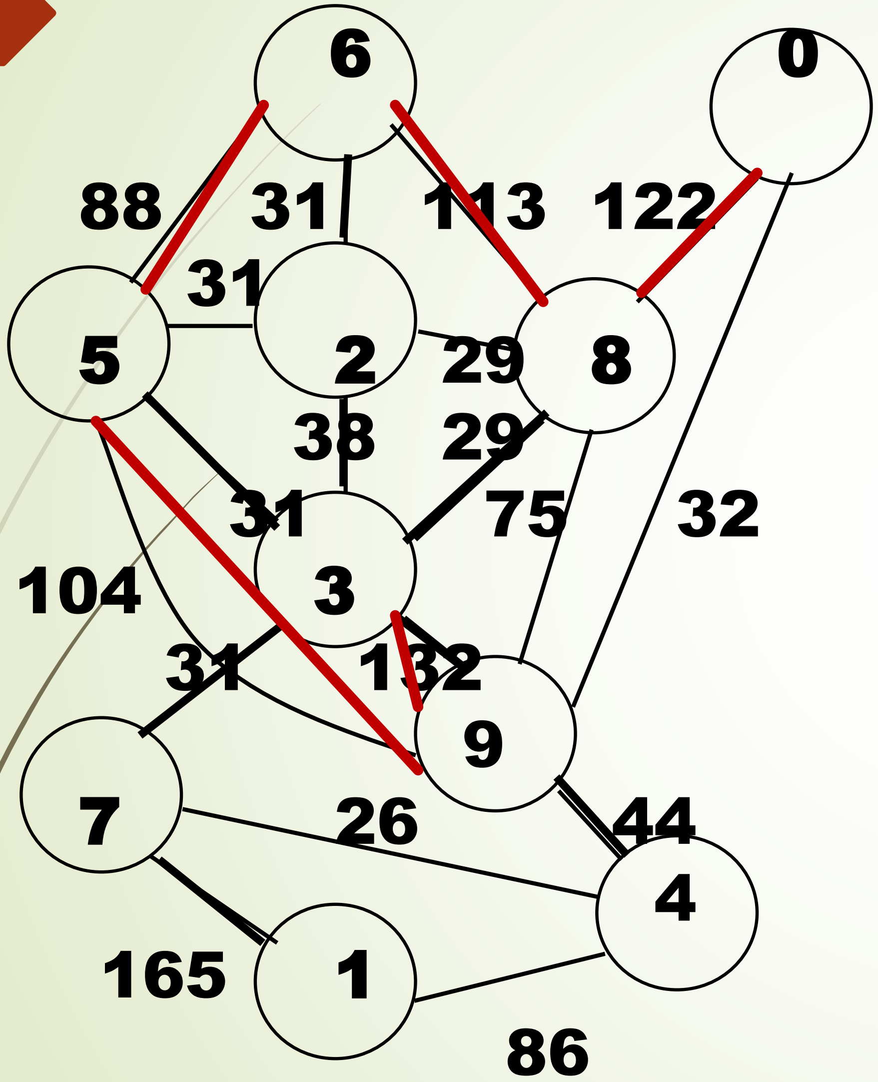
Response

Stimulus	1	2	3	4	5	6	7	8	9	0
	877	11	18	86	9	20	165	6	15	11
	11	782	38	13	31	31	9	29	18	11
	18	38	681	6	31	4	31	29	132	11
	86	13	6	732	9	11	26	13	44	6
	9	31	31	9	669	88	7	13	104	11
	20	31	4	11	88	633	2	113	11	31
	165	9	31	26	7	2	667	6	13	16
	6	29	29	13	13	113	6	577	75	122
	15	18	132	44	104	11	13	75	550	32
	11	11	11	6	11	31	16	122	32	818

Example: Digits 0, 1, 2,…, 9 confusion data as a graph (weights > 21)

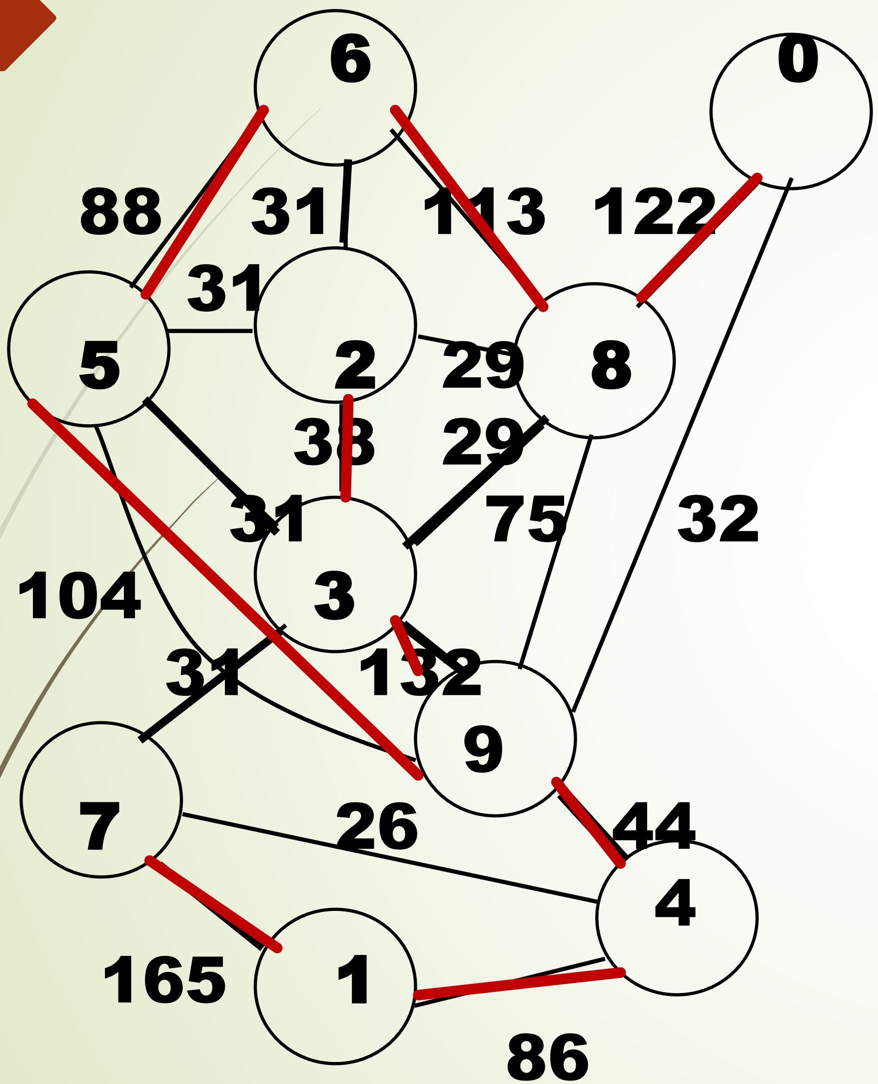


Maximum ST to build



1. Start with arbitrary node. Let it be 0.
2. A. Find a node nearest to 0:
0 --- 8 (122) which is current T.
   B. Find a node which is the nearest to either 0 or 8: 0---8(122)---6(113)
   C. Find a node nearest to either 0 or 8 or 6: 0---8(122)---6(113)----5 (88)
   D. Find a node nearest to 0, 8, 6, or 5:
0---8(122)---6(113)----5(88)----9(104)
   E. Find a node which is the nearest to either of 0,8,6,5,9:

Maximum ST built (see in red)



F. Find a node which is the nearest to either 0,8,6,5,9,3:
0---8(122)---6(113)----5(88)----9(104)----3 (132)

|
4 (44)

G. F. E. A node which is the nearest to either 0,8,6,5,9,3, 4:
0---8(122)---6(113)----5(88)----9(104)----3 (132)

|
4 (44)----1 (86)

H. **Final MST:**

2(38)
|
0--8(122)--6(113)--5(88)--9(104)--3 (132)
|

MagDataAnalysis2019

Hierarchical clustering

1. Agglomertive clustering
2. Ward clustering
3. Deriving Ward criterion
4. Nearest Neighbor clustering

Maximum/Minimum Spanning Tree; Prim Algorithm

6. **Divisive NN Clustering**

Convert an MST to a Nearest Neighbor Hierarchy/Partition

- **K-part partition**: **Cut K-1 weakest links**

- **NN binary hierarchy**: **build a hierarchy top-down by cutting the weakest link at each step**

NN Partition: at K=4, cut 3 weakest links

2(38)

|

0--8(122)--6(113)--5(88)--9(104)--3 (132)

|

4(44)--1 (86)–7

(165)

NN Hierarchy Divisively: Sort MST links, cut over the weakest link one by one

All

2(38)

All without "2" cut38

cut44

0--8--6--5--9--3 4--1–7

cut88 cut86

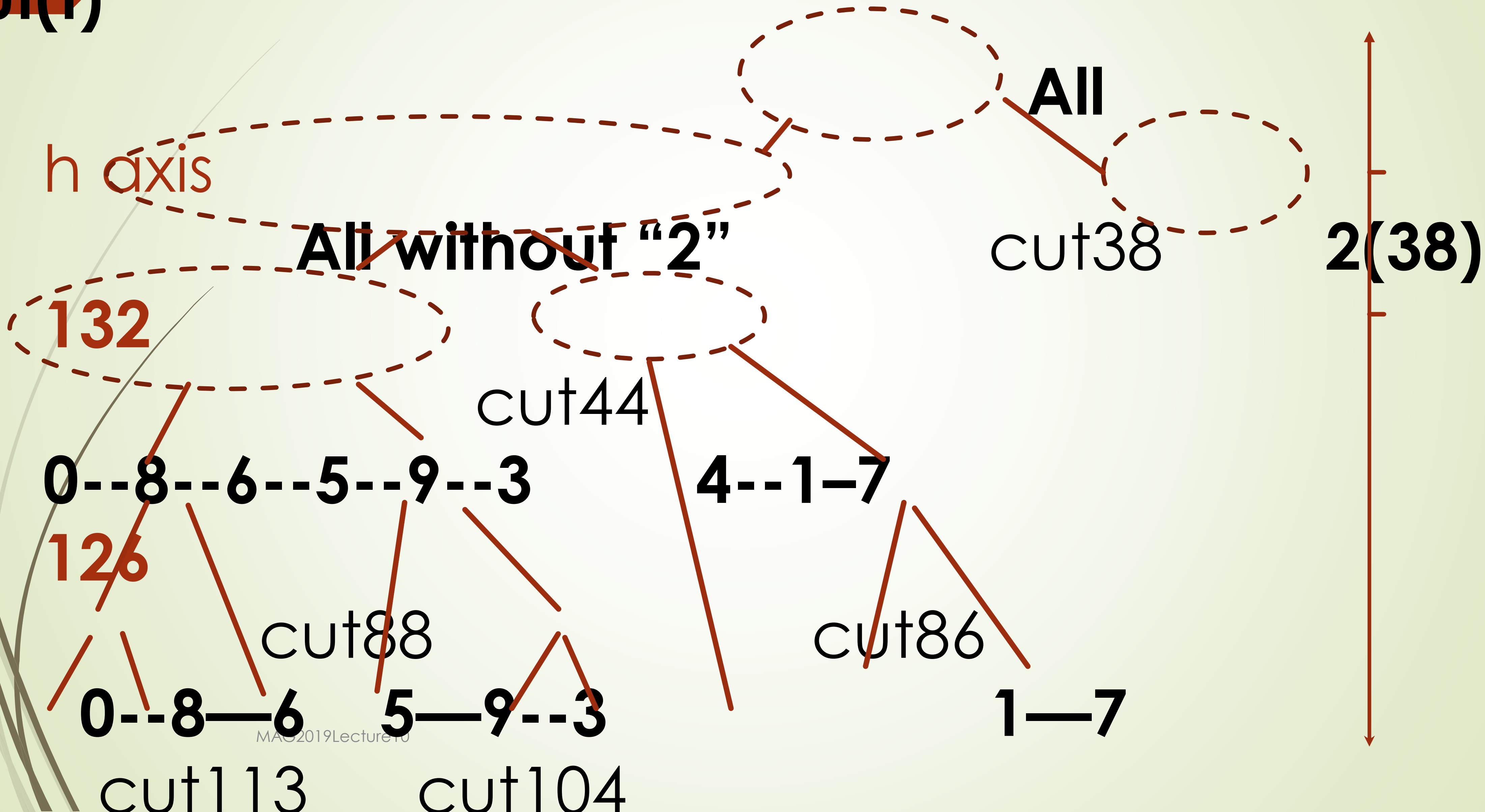
0--8—6 5—9--3 4 1—7

cut113 cut104

0—8 6 5 9—3 1 7

0 8 9 3

NN Hierarchy with a height function, say h(t)=170-cut(t)



Summary of the lecture

- Cluster hierarchy as a binary rooted tree with a height function, whose leaves are one-to-one labeled by dataset entities
- Agglomerative clustering algorithm
- Distance between clusters:
  - Ward distance
  - Nearest Neighbor distance
- Derivation of Ward cluster-to-cluster distance as the increment of the K-means square error criterion at the cluster merger
- Max/Min Spanning Tree and Prim's algorithm
- NN Divisive clustering with MST