Unsupervised Learning with Clustering

K-means

EMclustering

Hierarchical Clustering

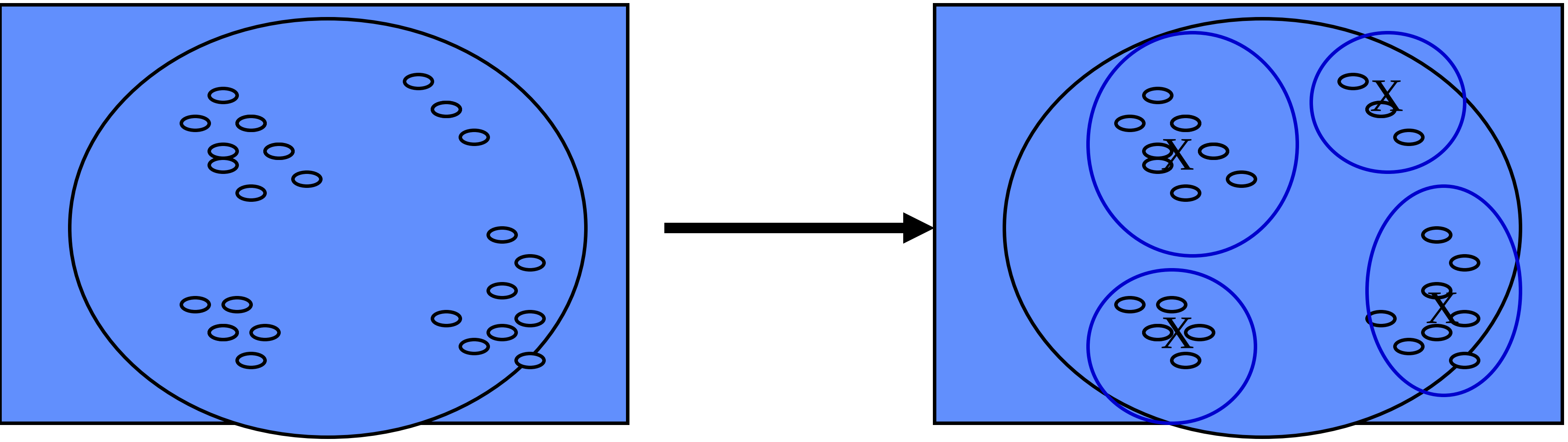
Why Clustering

- **A good grouping implies some structure**
- **In other words, given a good grouping, we can then:**
  - Interpret and label clusters
  - Identify important features
  - Characterize new points by the closest cluster (or nearest neighbors)
  - Use the cluster assignments as a compression or summary of the data

Clustering

- **Basic idea: Group similar things together**
- **Unsupervised Learning – Useful when no other info is available**
- **K-means**
  - Partitioning instances into k disjoint clusters
  - Measure of similarity

Clustering



Clustering Techniques

- **K-means clustering**
- **Hierarchical clustering**
- **Conceptual clustering**
- **Probability-based clustering**
- **Bayesian clustering**

Clustering Techniques

- **K-means clustering**
- **Hierarchical clustering**
- **Conceptual clustering**
- **Probability-based clustering**
- **Bayesian clustering**

Clustering Applications

- Example: Clustering of a large number of gene experiments
- Multiple sequence alignment of genes closely clustered together
- Search for metabolic pathways genes may be involved with
- Possible functional classification of genes in the same cluster
- Identifying co-regulated genes from expression arrays

Common uses of Clustering

- Often used as an exploratory data analysis tool
- In one-dimension, a good way to quantify real-valued variables into k non-uniform buckets
- Used on acoustic data in speech understanding to convert waveforms into one of k categories (known as Vector Quantization)
- Also used for choosing color palettes on old fashioned graphical display devices
- Color Image Segmentation

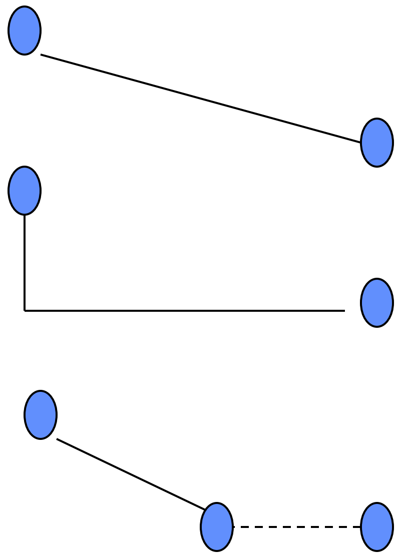
Clustering

- **Unsupervised: no target value to be predicted**
- **Differences ways clustering results can be produced/represented/learned**
  - Exclusive vs. overlapping
  - Deterministic vs. probabilistic
  - Hierarchical vs. flat
  - Incremental vs. batch learning

Clustering Objective

- **Objective: find subsets that are similar within cluster and dissimilar between clusters**
- **Similarity defined by distance measures**

  - Euclidean distance
  - Manhattan distance
  - Mahalanobis (Euclidean w/dimensions rescaled by variance)



Clustering Objective

- **Objective: find subsets that are similar within cluster and dissimilar between clusters**
- **Similarity defined by distance measures**

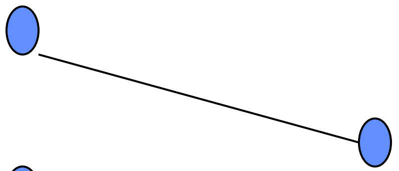
  - Euclidean distance =

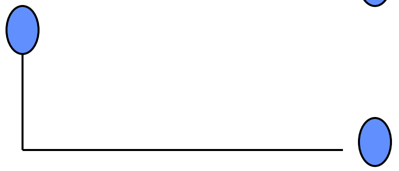
    sqrt[(a1 - b1)^2+ (a2 – b2)^2+...)]
  - Manhattan distance

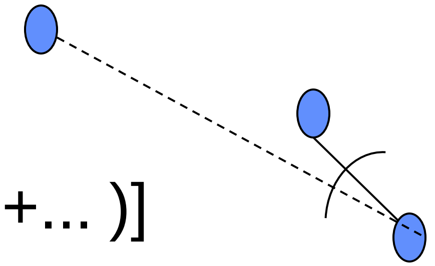
    [|a1 - b1|+ |a2 – b2|+...)]
  - Cosine (insensitive to size)

    Euc Dist/

    sqrt[(a1)^2 +(a2)^2..]*sqrt[(b1)^2+...)]







The k-means Algorithm Iterative Distance Based Clustering

- **Clusters the data into k groups where k is specified in advance**
  1. Cluster centers are chosen at random
  2. Instances are assigned to clusters based on their distance to the cluster centers
  3. Centroids of clusters are computed – “means”
  4. Go to 1st step until convergence

K-means Clustering

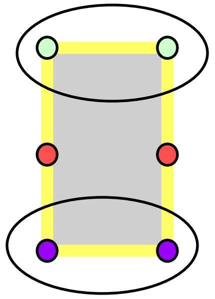
- **A simple, effective, and standard method**
    - Start with K initial cluster centers
    - Loop:
        - Assign each data point to nearest cluster center
        - Calculate mean of cluster for new center
    - Stop when assignments don't change
- **Issues:**
    - How to choose K?
    - How to choose initial centers?
    - Will it always stop?

K-Means Clustering Pros & Cons

- **Simple and reasonably effective**
- **The final cluster centers do not represent a global minimum but only a local one**
- **Result can vary significantly based on initial choice of seeds**
  - Completely different final clusters can arise from differences in the initial randomly chosen cluster centers
- **Algorithm can easily fail to find a reasonable clustering**

Getting Trapped in a Local Minimum

- **Example: four instances at the vertices of a two-dimensional rectangle**
  - Local minimum: two cluster centers at the midpoints of the rectangle's long sides



- **Simple way to increase chance of finding a global optimum: restart with different random seeds**

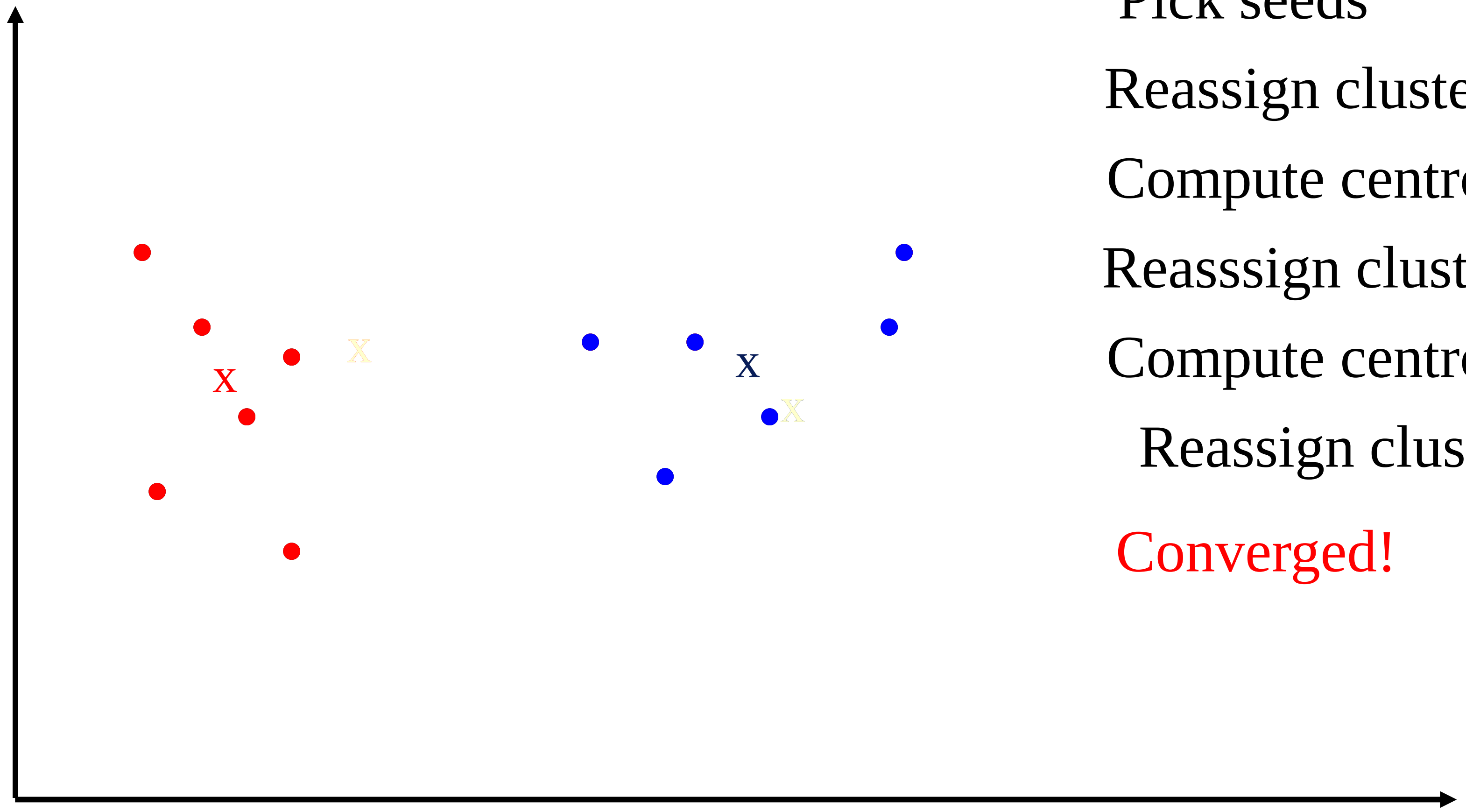
Clustering

- **Partition unlabeled examples into disjoint subsets of *clusters*, such that:**
  - Examples within a cluster are very similar
  - Examples in different clusters are very different
- **Discover new categories in an *unsupervised* manner (no sample category labels provided)**

K-Means Algorithm

```
Let $d$ be the distance measure between instances.
Select $k$ random instances $\{s_1, s_2, \ldots s_k\}$ as seeds.
Until clustering converges or other stopping criterion:
    For each instance $x_i$:
        Assign $x_i$ to the cluster $c_j$ such that $d(x_i, s_j)$ is
minimal.
    *(Update the seeds to the centroid of each cluster)*
    For each cluster $c_j$
        $s_j = \mu(c_j)$
```

K Means Example (K=2)



Pick seeds

Reassign clusters

Compute centroids

Reasssign clusters

Compute centroids

Reassign clusters

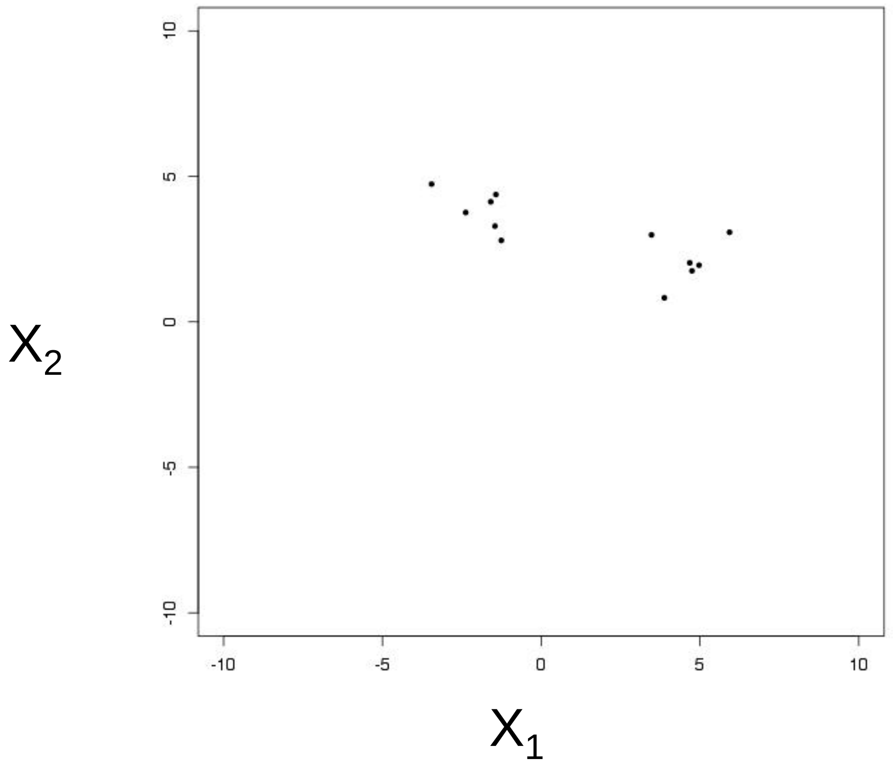
Converged!

Seed Choice

- Results can vary based on random seed selection
- Some seeds can result in poor convergence rate, or convergence to sub-optimal clusters
- Select good seeds using a heuristic or the results of another method

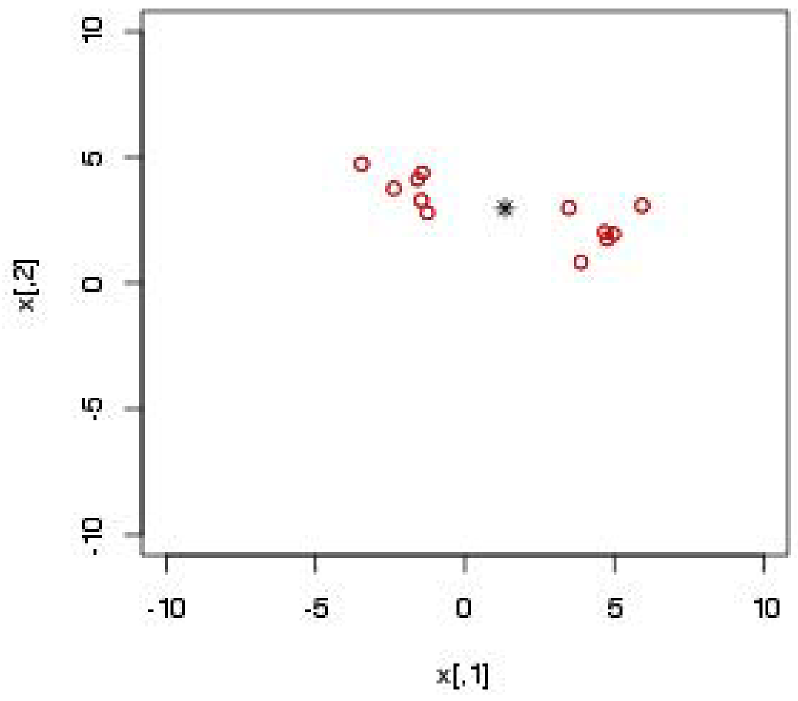
K-means Example

- **For K=1, using Euclidean distance, where will the cluster center be?**



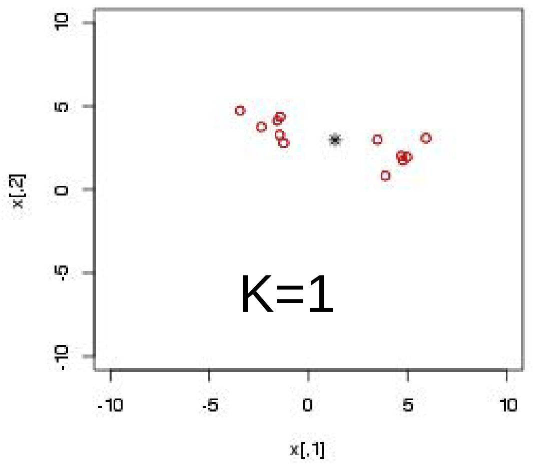
K-means Example

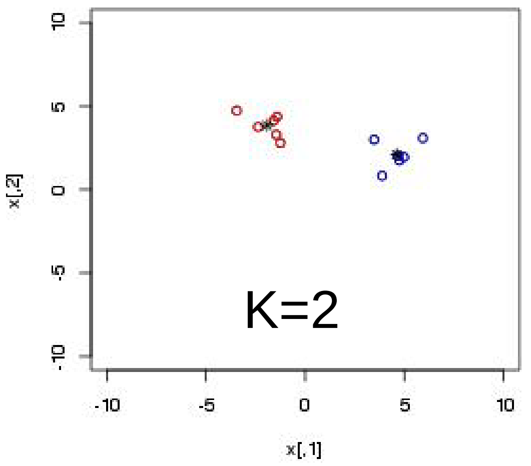
- **For K=1, the overall mean minimizes Sum Squared Error (SSE), aka Euclidean distance**

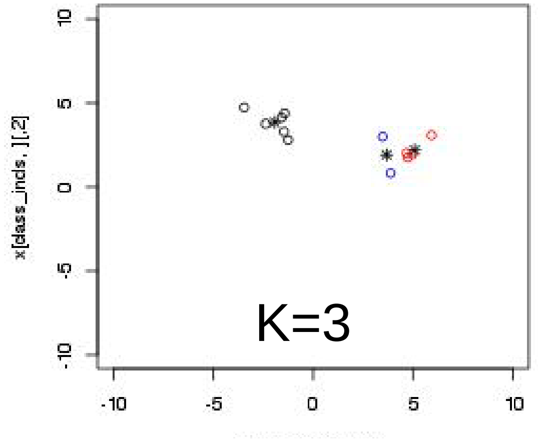


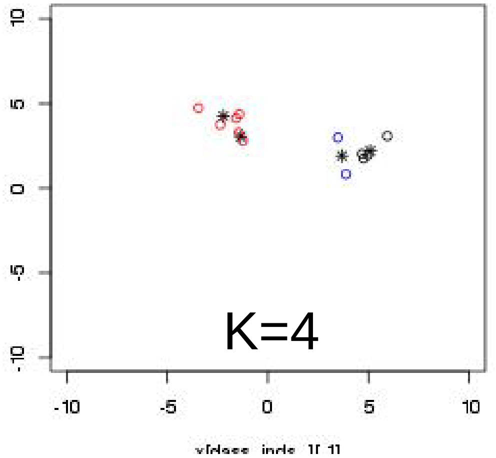
Simple example:
#choose 1 data point as initial K centers
#10 is max loop iterations
#1 is number of initial sets to try

K-means Example

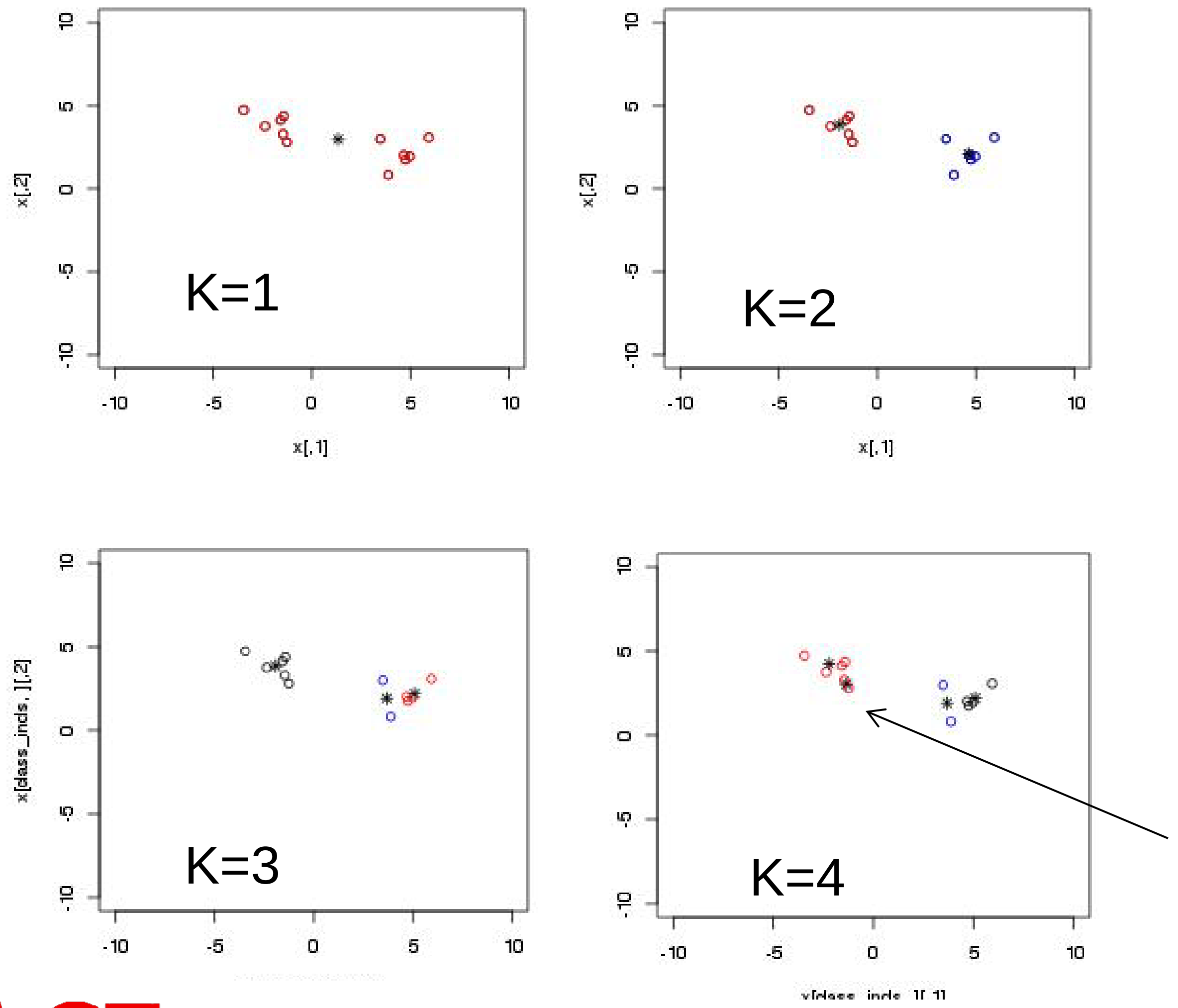








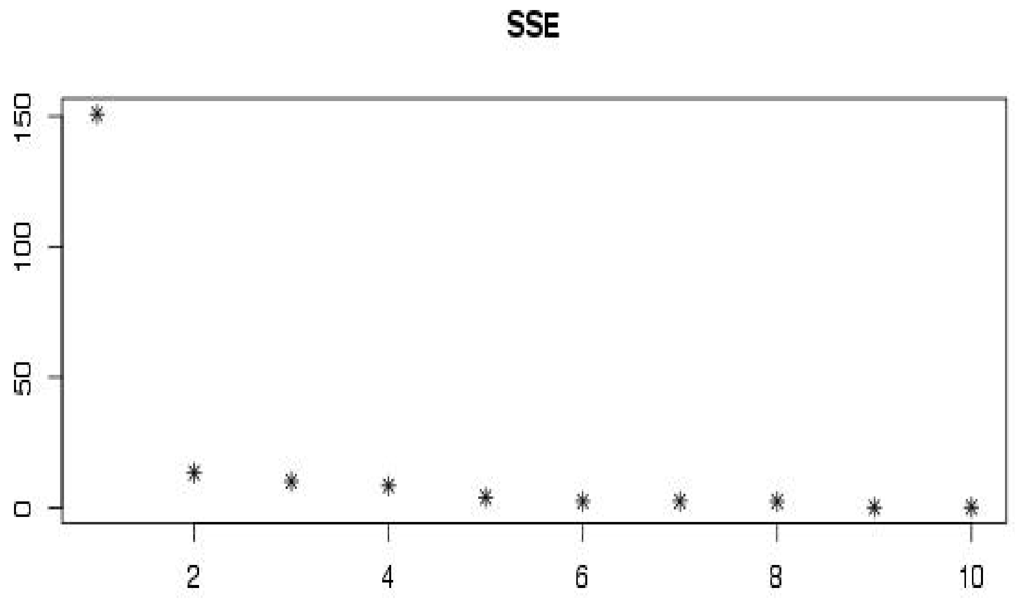
K-means Example



As K increases individual points get a cluster

Choosing K for K-means

Total
Within
Cluster
SSE

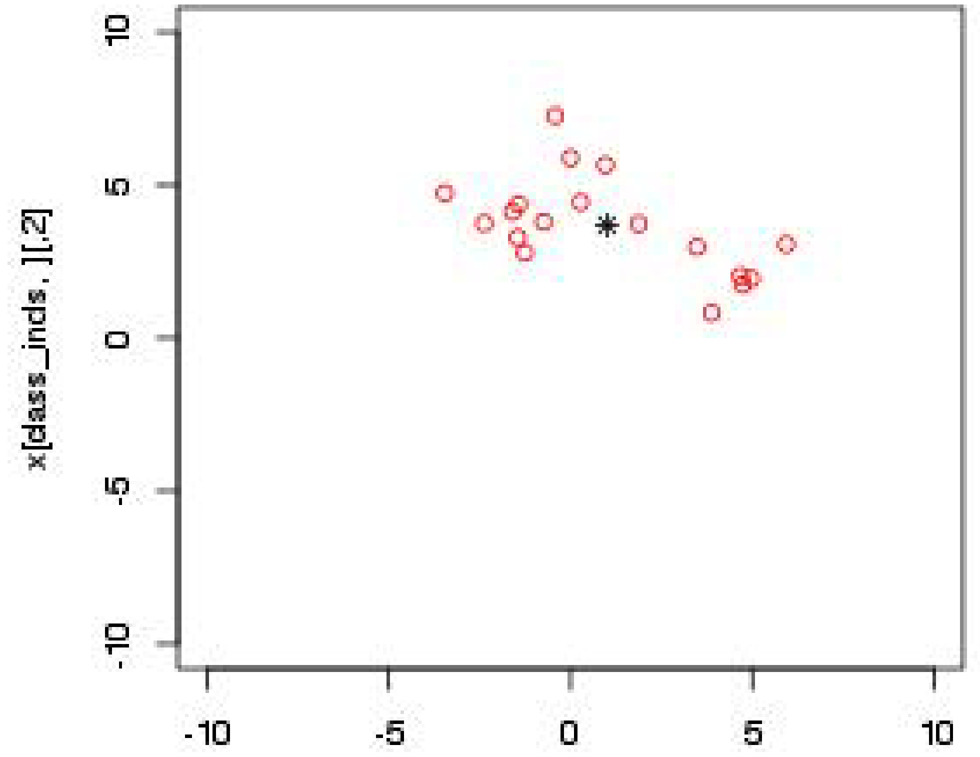


K=1 to 10

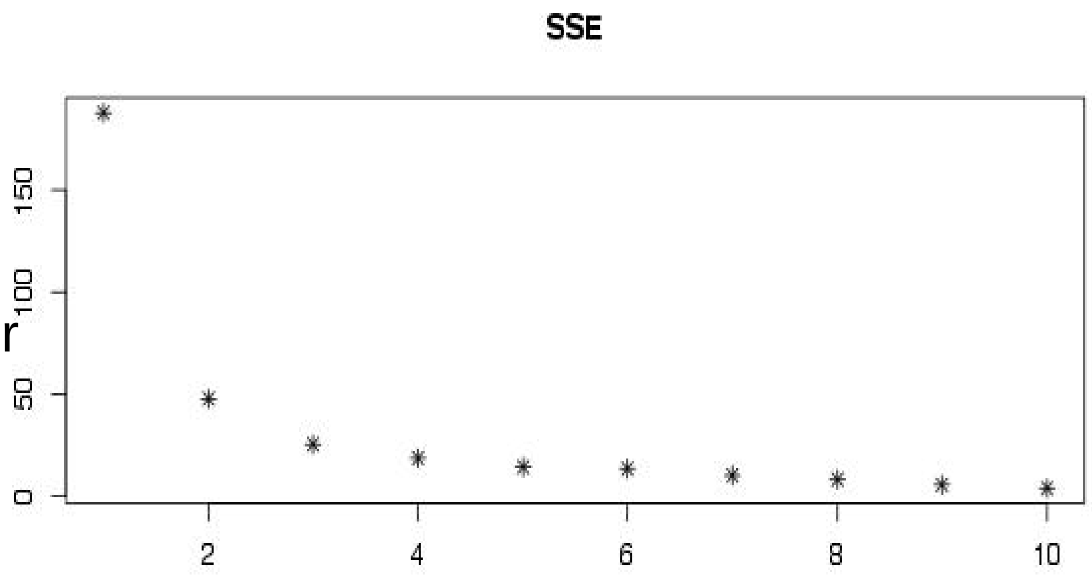
- Not much improvement after K=2 ("elbow")

K-means Example – more points

How many clusters should there be?



Choosing K for K-means



Total Within Cluster SSE

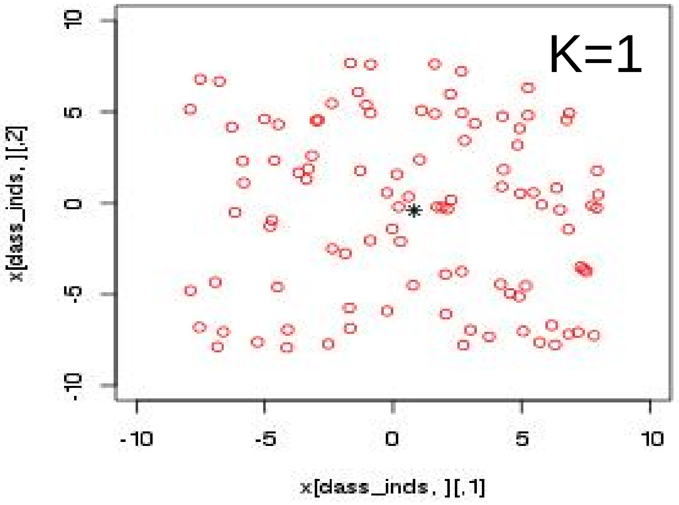
K=1 to 10

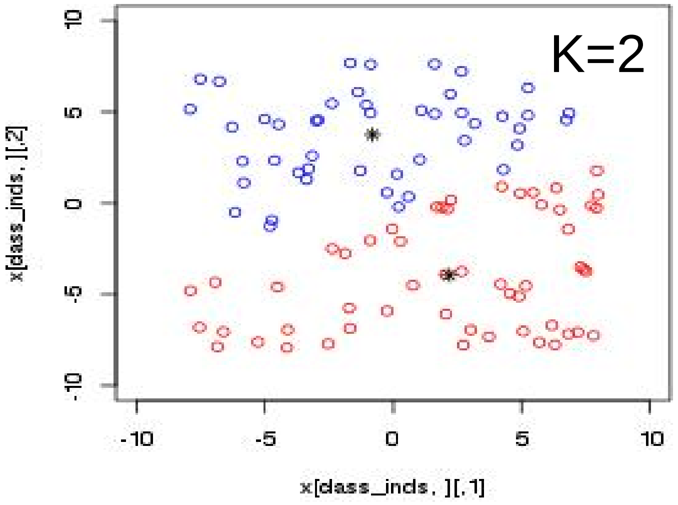
- Smooth decrease at K ≥ 2, harder to choose
- In general, smoother decrease => less structure

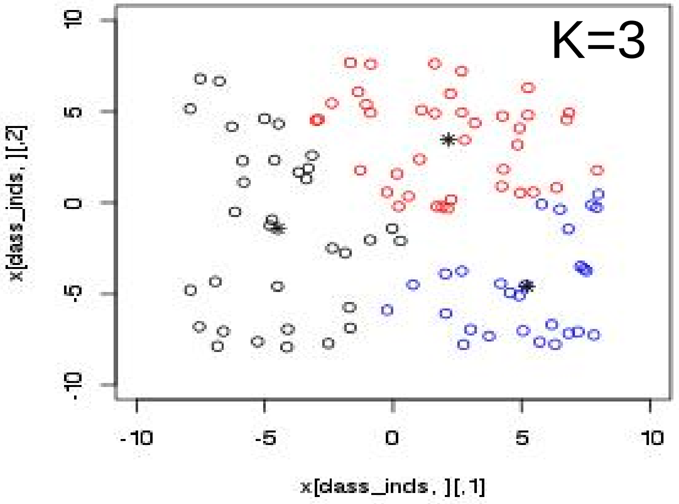
K-means Guidelines

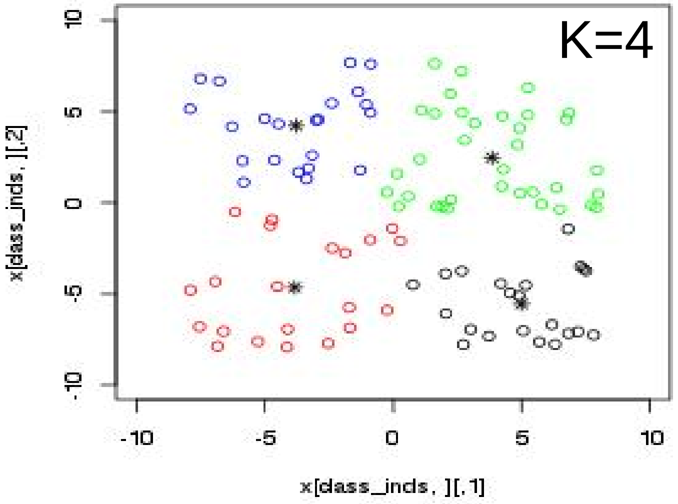
- **Choosing K:**
  - "Elbow" in total-within-cluster SSE as K=1…N
  - Cross-validation: hold out points, compare fit as K=1…N
- **Choosing initial starting points:**
  - take K random data points, do several K-means, take best fit
- **Stopping:**
  - may converge to sub-optimal clusters
  - may get stuck or have slow convergence (point assignments bounce around), 10 iterations is often good

K-means Example: uniform dist.



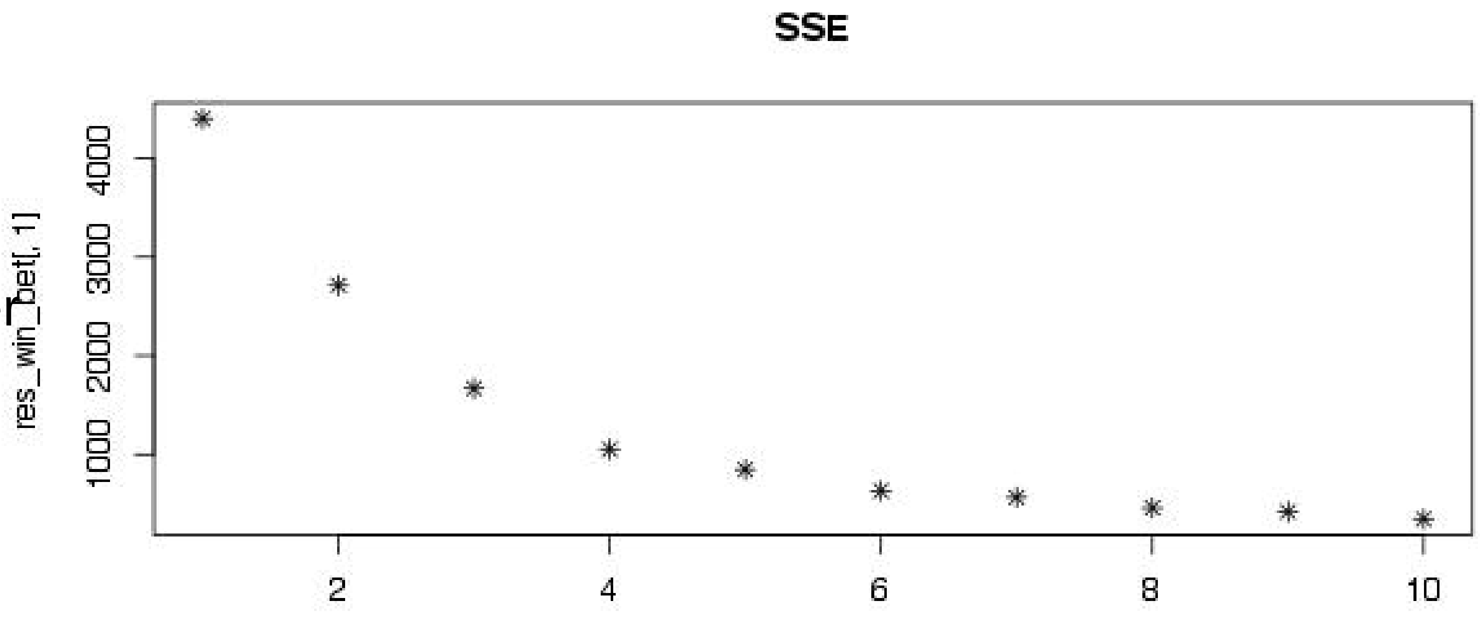






Choosing K - uniform

Total Within Cluster SSE



K=1 to 10

- Smooth decrease across K => less structure

K-means Clustering Issues

- **Scale:**
  - Dimensions with large numbers may dominate distance metrics

- **Outliers:**
  - Outliers can pull cluster mean, K-mediods uses median instead of mean

Probability-based Clustering

- **Problems with K-means & Hierarchical methods:**
  - Division by k
  - Order of examples
  - Merging/splitting operations might not be sufficient to reverse the effects of bad initial ordering
  - Is result at least local minimum of category utility?
- **Solution:**
  - Find the most likely clusters given the data
- **Instance has certain probability of belonging to a particular cluster**

Soft Clustering

- **So far clustering methods assumed that each instance has a "hard" assignment to exactly one cluster**
- **No uncertainty about class membership or an instance belonging to more than one cluster**
- ***Soft clustering* gives probabilities that an instance belongs to each of a set of clusters**
- **Each instance is assigned a probability distribution across a set of discovered clusters**
  - probabilities of all categories must sum to 1

Soft Clustering Methods

- **Fuzzy Clustering**
  - Use weighted assignments to all clusters
  - Weights depend on relative distance
  - Find min weighted SSE
- **Expectation-Maximization:**
  - Initialize a mixture of multivariate Gaussian distributions
  - Find means, variances, and mixture weights that maximize probability of data

Finite mixtures

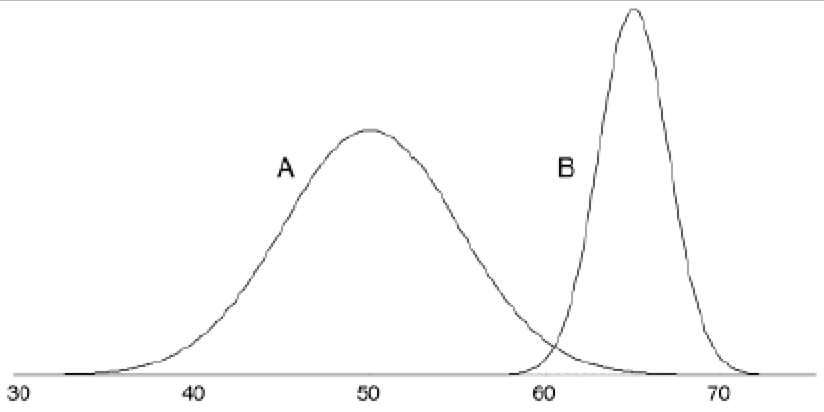
- **Probabilistic clustering algorithms model the data using a mixture of distributions**
- **Each cluster is represented by one distribution**
  - The distribution governs the probabilities of attributes values in the corresponding cluster
- **They are called finite mixtures because there is only a finite number of clusters being represented**
- **Usually individual distributions are normal**
- **Distributions are combined using cluster weights**

A Two-Class Mixture Model

data

A	51	B	62	B	64	A	48	A	39	A	51
A	43	A	47	A	51	B	64	B	62	A	48
B	62	A	52	A	52	A	51	B	64	B	64
B	64	B	64	B	62	B	63	A	52	A	42
A	45	A	51	A	49	A	43	B	63	A	48
A	42	B	65	A	48	B	65	B	64	A	41
A	46	A	48	B	62	B	66	A	48		
A	45	A	49	A	43	B	65	B	64		
A	45	A	46	A	40	A	46	A	48		

model



μ_A=50, σ_A=5, p_A=0.6 μ_B=65, σ_B=2, p_B=0.4

Using the Mixture Model

- **The probability of an instance x belonging to cluster A is:**

$$PR[A|x] = \frac{Pr[x|A]Pr[A]}{Pr[x]} = \frac{f(x;\mu_A,\sigma_A)p_A}{Pr[x]}$$

Learning the Clusters

- **Assume we know that there are k clusters**
- **To learn the clusters we need to determine their parameters**
  - I.e. their means and standard deviations
- **Start with the initial guess for the 5 parameters use them to calculate cluster probabilities for each instance, use these probabilities to re estimate the parameters and repeat**
- **We actually have a performance criterion: the likelihood of the training data given the clusters**

Expectation Maximization (EM)

- **Probabilistic method for soft clustering**
- **Iterative method for learning probabilistic categorization model from unsupervised data**
- **Direct method that assumes *k* clusters:**$\{c_1, c_2, \ldots c_k\}$
- **Soft version of *k*-means**
- **Assumes a probabilistic model of categories that allows computing P(c_i | *E*) for each category, c_i, for a given example, *E***

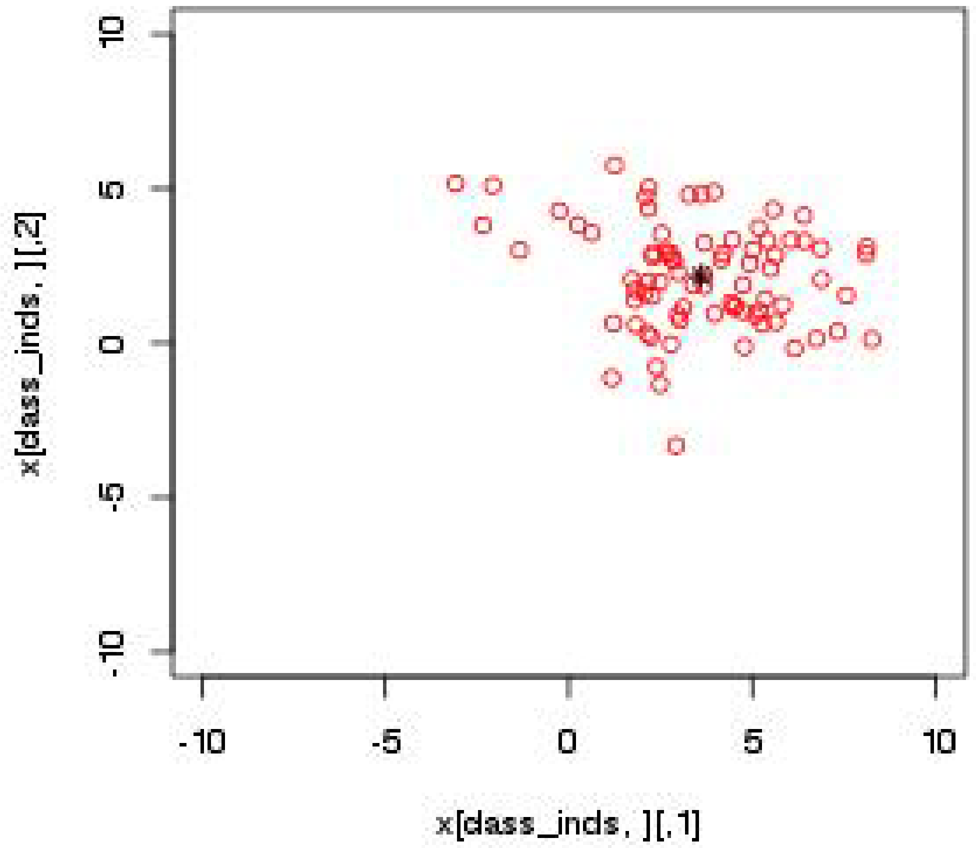
EM Algorithm

- Initially assume random assignment of examples to categories
- Learn an initial probabilistic model by estimating model parameters from this randomly labeled data

The EM Algorithm

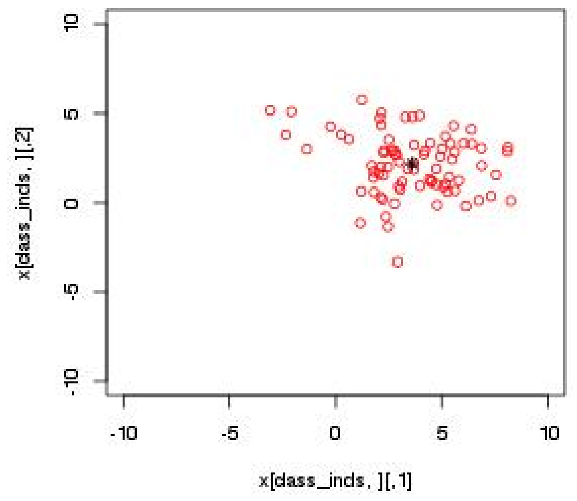
- **EM algorithm:**
  - **expectation-maximization algorithm**
- **Generalization of k-means to probabilistic setting**
- **Similar iterative procedure**
  1. Calculate cluster probability for each instance (expectation step)
  2. Estimate distribution parameters based on the cluster probabilities (maximization step)
- **Cluster probabilities are stored as instance weights**

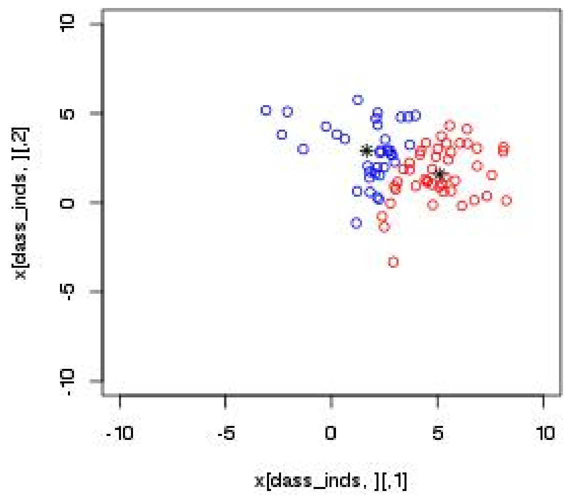
Kmeans – unequal cluster variance

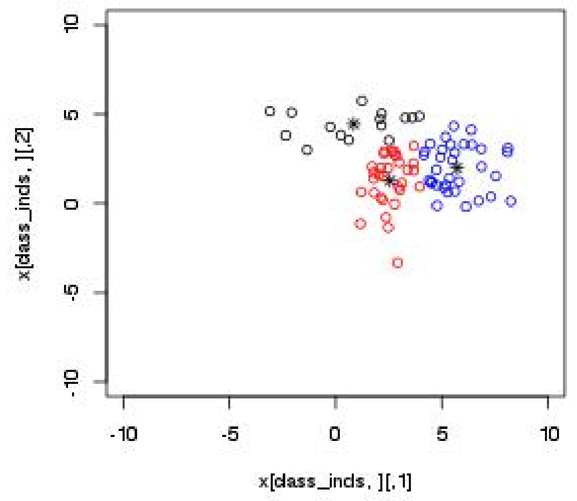


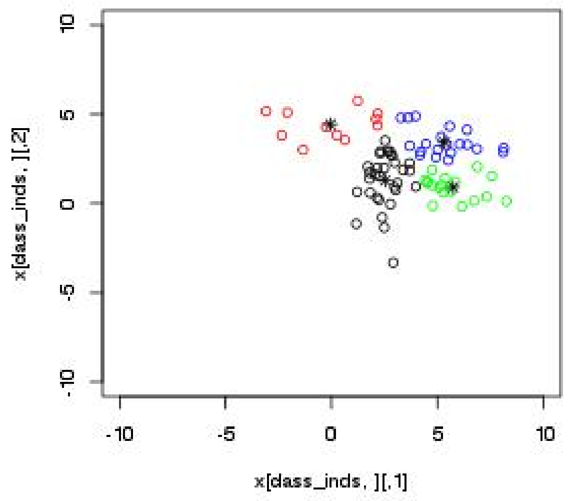
Can you guess K?

K-means – unequal cluster variance

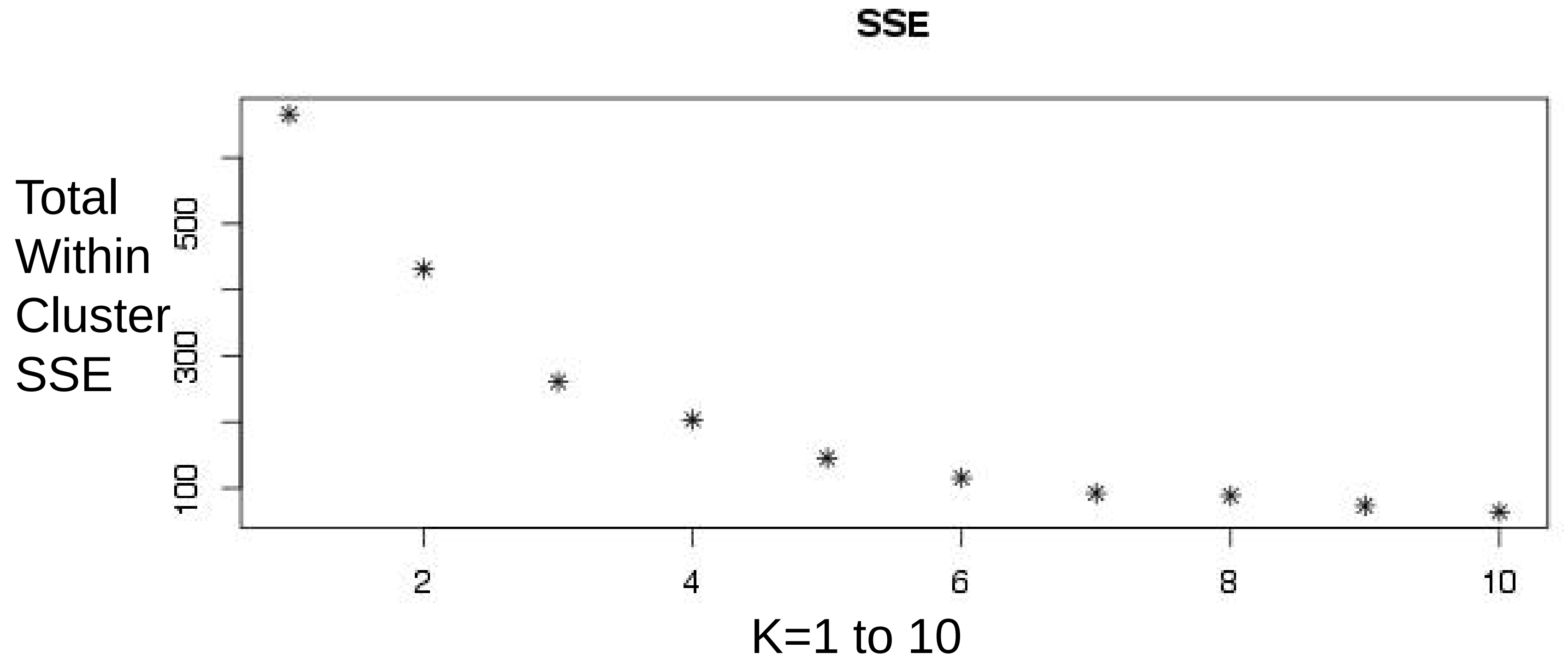






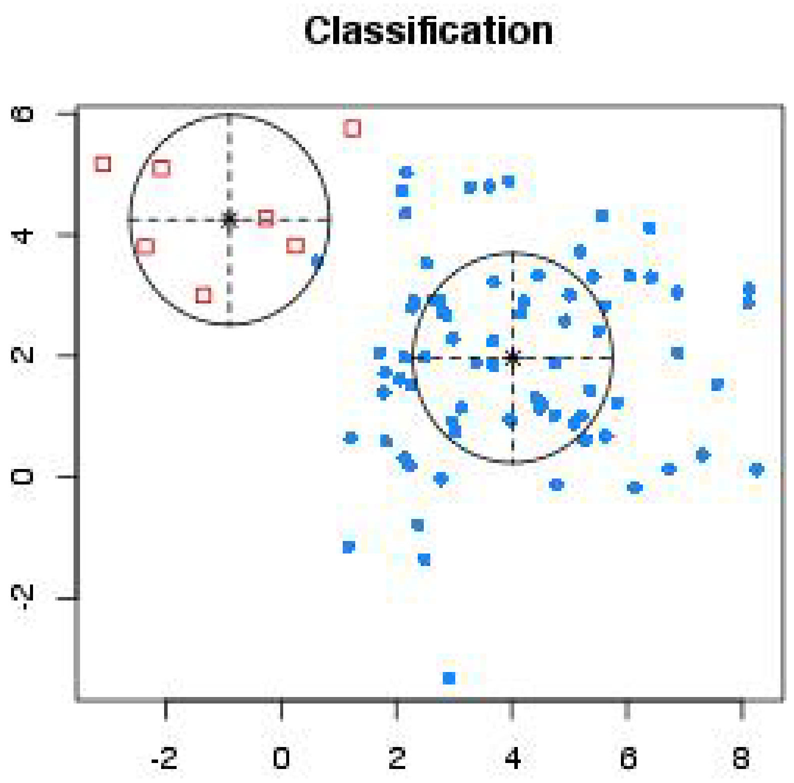


Choosing K – unequal distributions



- Smooth decrease across K => less structure

EM clustering



- Selects K=2
  (either by Information Criterion=
  min of SSE+ K*logN,
  Or by cross-validation)

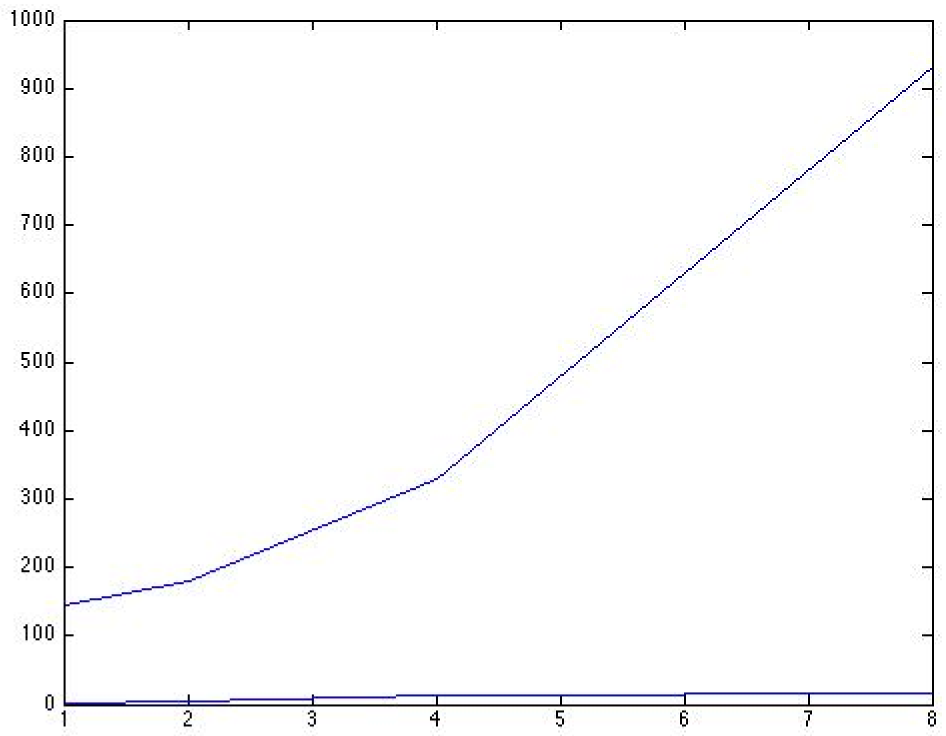
- Handles unequal variance

K-means computations

- **Distance of each point to each cluster center**
  - For *N* points, *D* dimensions: each loop requires *N*D*K* operations
- **Update Cluster centers**
  - only track points that change, get change in cluster center
- **But for EM errors to each cluster center update a probability function**

K-means vs EM performance

1 Gordon compute node, normal random matrices
R: system.time(Mclust())



Wall Time (secs)

15 min

10 min

EM: 1000 pts

Kmeans:1000 pts

1K 4K 8K

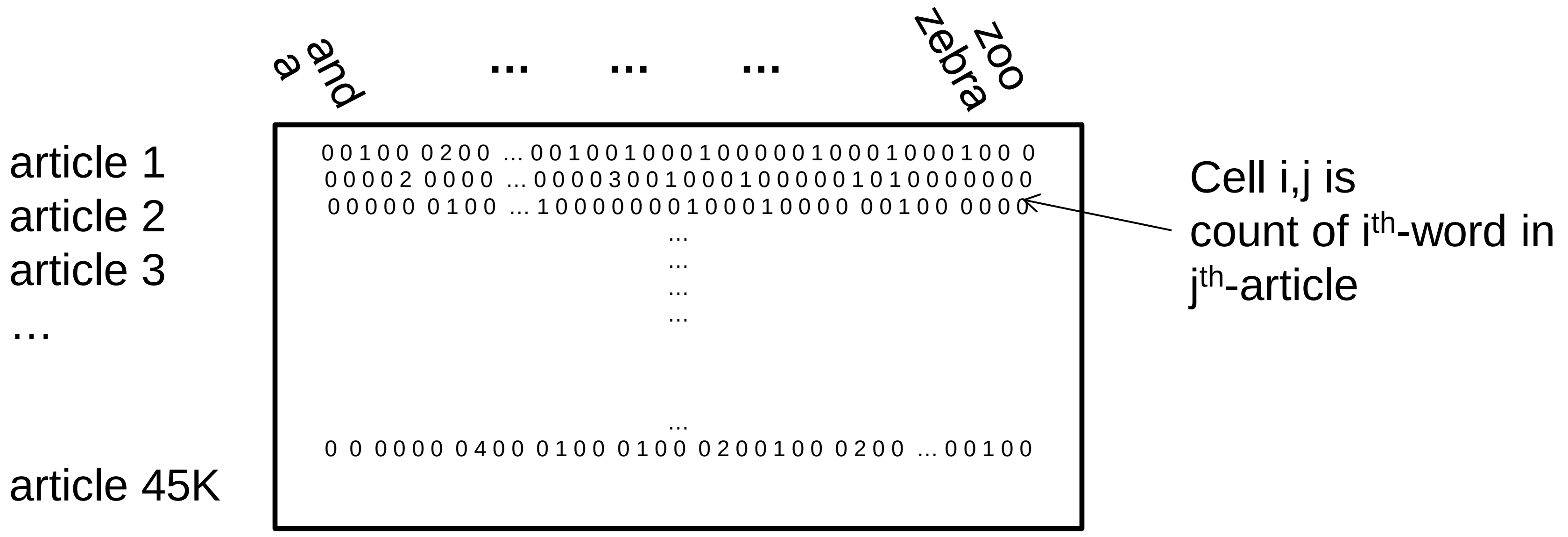
Number of Dimensions (i.e. columns in data matrix)

Kmeans big data example

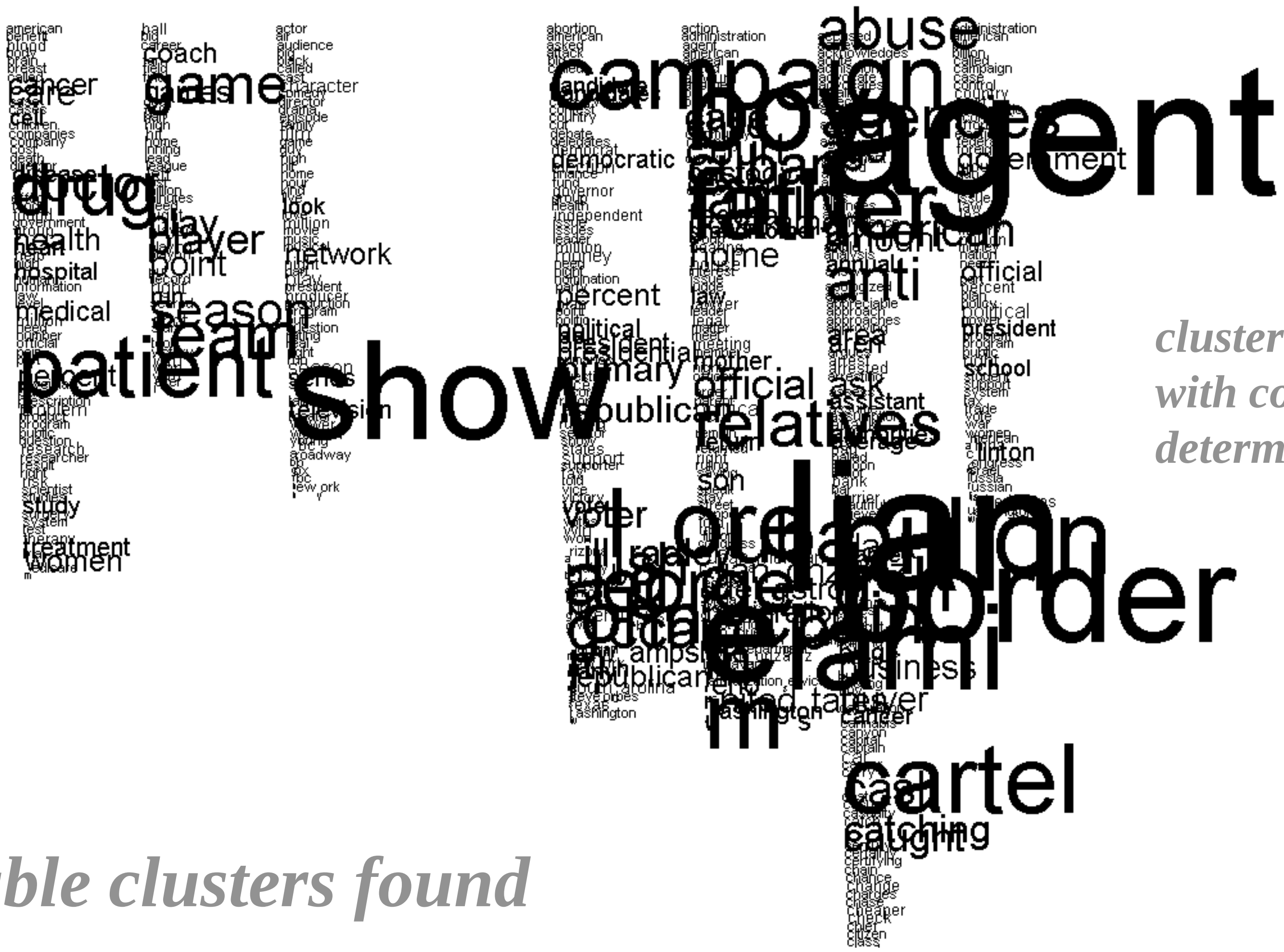
- **45,000 NYTimes articles, 102,000 unique words**

  (UCI Machine Learning repository)

- **Full Data Matrix: 45Kx102K ~ 40Gb**



Kmeans results

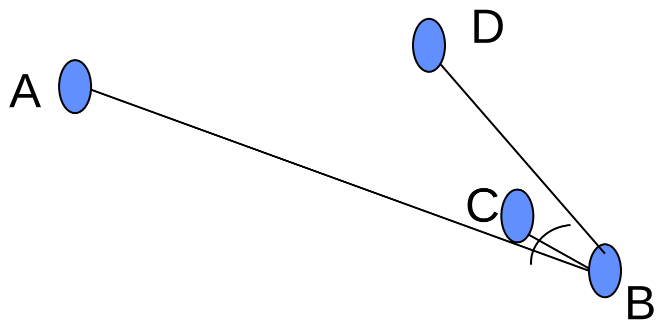


cluster means shown with coordinates determining fontsize

7 viable clusters found

Other distance measures

- Cosine: each row is treated as vector in R^p, then take angles



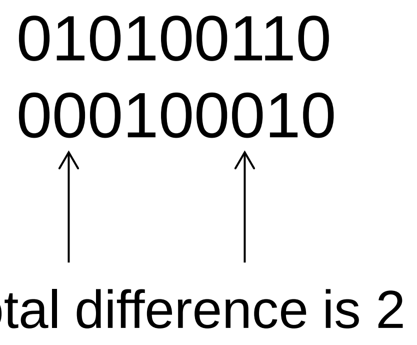
C is closer to A because ABCº<ABDº

- Jaccard (over sets A,B): $1- (|A \cap B| / |A \cup B|)$

Other distance measures

- Hamming distance: count 1 if values different

    e.g. appropriate for binary strings



010100110

000100010

Total difference is 2

Summary

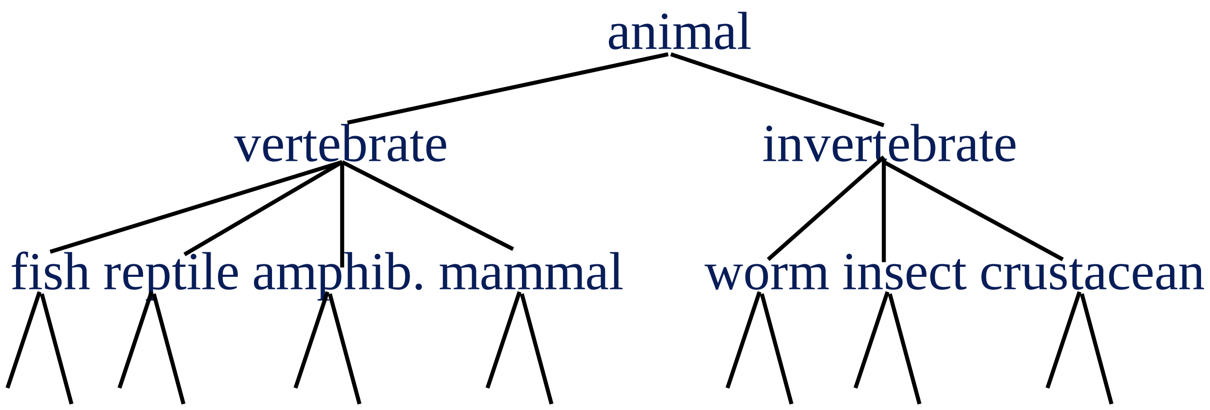
- **Labeled clusters can be interpreted by using supervised learning - train a tree or learn rules**
- **Can be used to fill in missing attribute values**
- **All methods have a basic assumption of independence between the attributes**
    - Some methods allow the user to specify in advanced that two of more attributes are dependent and should be modeled with a joint probability

Clustering algorithms in scikit-learn

Method name	Parameters	Scalability	Usecase	Geometry (metric used)
K-Means	number of clusters	Very large n_samples, medium n_clusters with MiniBatch code	General-purpose, even cluster size, flat geometry, not too many clusters	Distances between points
Affinity propagation	damping, sample preference	Not scalable with n_samples	Many clusters, uneven cluster size, non-flat geometry	Graph distance (e.g. nearest-neighbor graph)
Mean-shift	bandwidth	Not scalable with n_samples	Many clusters, uneven cluster size, non-flat geometry	Distances between points
Spectral clustering	number of clusters	Medium n_samples, small n_clusters	Few clusters, even cluster size, non-flat geometry	Graph distance (e.g. nearest-neighbor graph)
Ward hierarchical clustering	number of clusters	Large n_samples and n_clusters	Many clusters, possibly connectivity constraints	Distances between points
Agglomerative clustering	number of clusters, linkage type, distance	Large n_samples and n_clusters	Many clusters, possibly connectivity constraints, non Euclidean distances	Any pairwise distance
DBSCAN	neighborhood size	Very large n_samples, medium n_clusters	Non-flat geometry, uneven cluster sizes	Distances between nearest points
Gaussian mixtures	many	Not scalable	Flat geometry, good for density estimation	Mahalanobis distances to centers
Birch	branching factor, threshold, optional global clusterer.	Large n_clusters and n_samples	Large dataset, outlier removal, data reduction.	Euclidean distance between points

Hierarchical Clustering

- Build a tree-based hierarchical taxonomy (*dendrogram*) from a set of unlabeled examples



- Recursive application of a standard clustering algorithm can produce a hierarchical clustering

Incremental & Hierarchical Clustering

- **Start with 1 cluster (all instances) and do splits**

  OR

  Start with N clusters (1 per instance) and do merges

- **Can be greedy & expensive in its search**

  some algorithms might merge & split

  algorithms need to store and recalculate distances

- **Need distance between groups**

  in contrast to K-means

Incremental & Hierarchical Clustering

- **Result is a hierarchy of clusters**
  - displayed as a ‘dendrogram’ tree

- **Useful for tree-like interpretations**
  - syntax (e.g. word co-occurences)
  - concepts (e.g. classification of animals)
  - topics (e.g. sorting Enron emails)
  - spatial data (e.g. city distances)
  - genetic expression (e.g. possible biological networks)
  - exploratory analysis

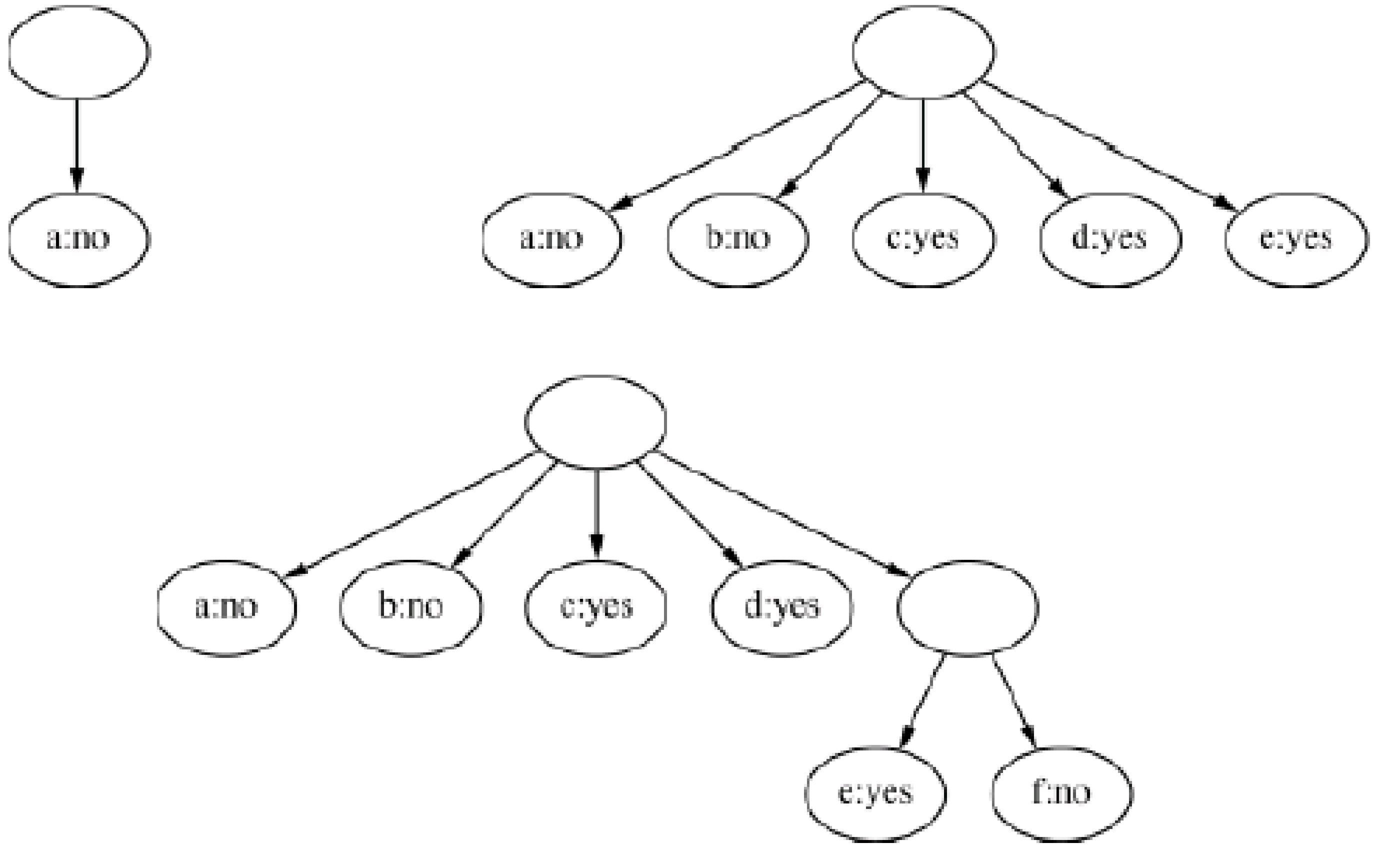
Incremental Clustering

- **Works incrementally instance by instance forming a a hierarchy of clusters**
- **COBWEB - nominal; CLASSIT– numeric attributes**
- **Instances are added one at the time**
  - Tree is updated appropriately at each step
  - Finding the right leaf for an instance
  - Restructuring the tree
- **How and where to update based on category utility value**

Clustering: Weather Data

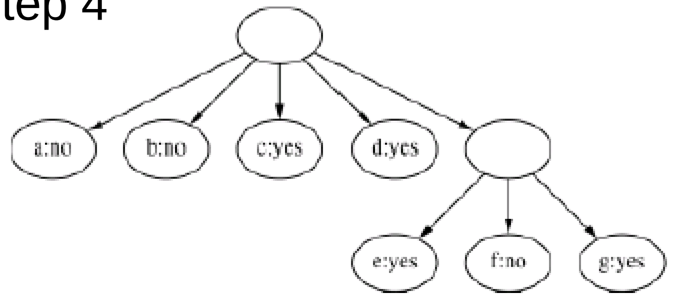
Weather Data Set				
ID Code	Outlook	Temperature	Humidity	Windy
A	Sunny	Hot	High	False
B	Sunny	Hot	High	True
C	Overcast	Hot	High	False
D	Rain	Mild	High	False
E	Rain	Cool	Normal	False
F	Rain	Cool	Normal	True
F	Overcast	Cool	Normal	True
H	Sunny	Mild	High	False
I	Sunny	Cool	Normal	False
J	Rain	Mild	Normal	False
K	Sunny	Mild	Normal	True
L	Overcast	Mild	High	True
M	Overcast	Hot	Normal	False
N	Rain	Mild	High	True

Clustering

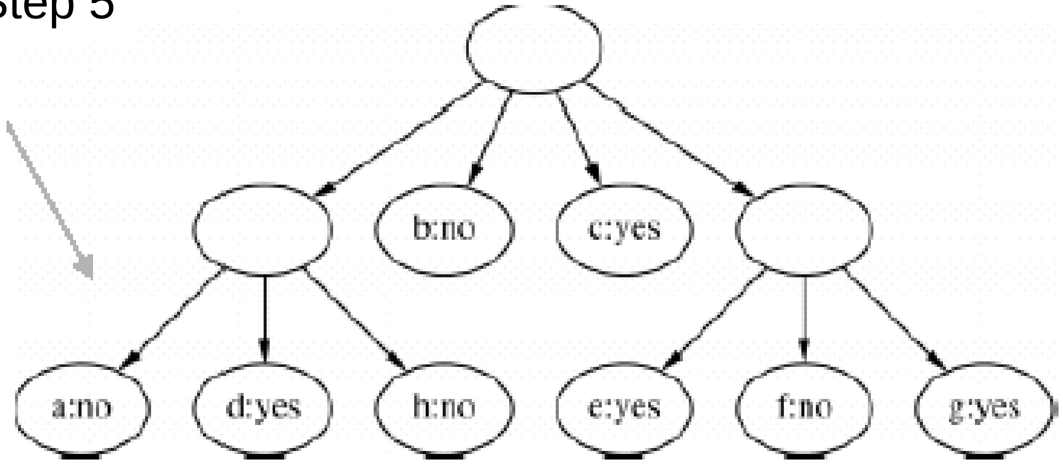


Clustering

Step 4



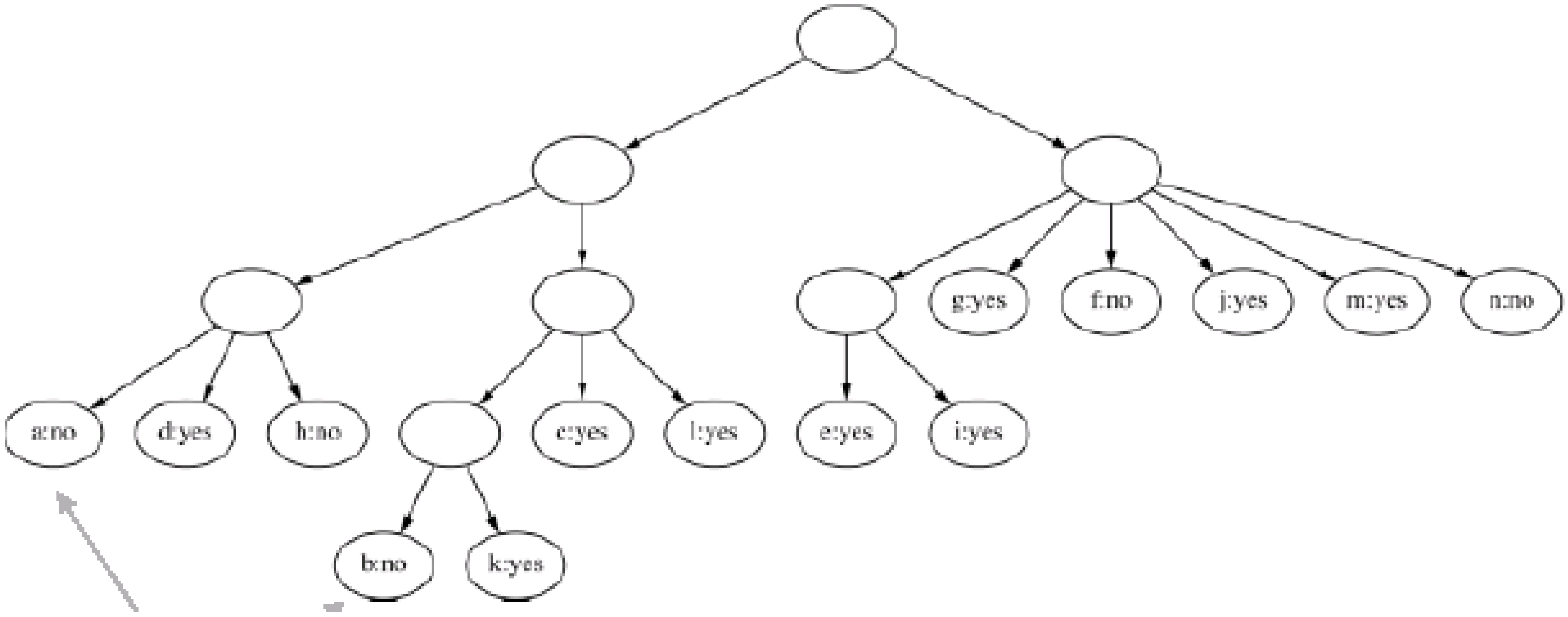
Step 5



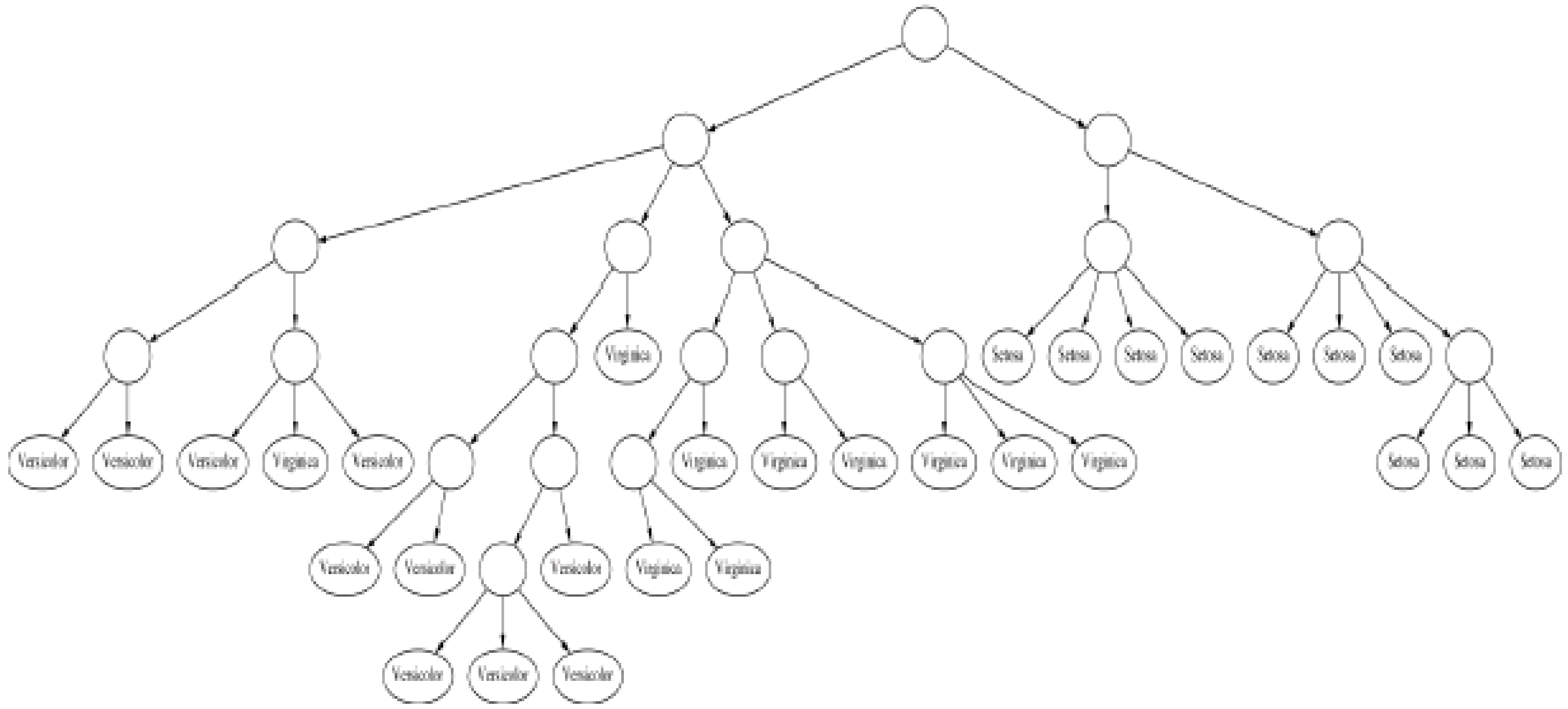
Merging

- **Consider all pairs of nodes for merging and evaluate category utility of each**
  - Computationally expensive
- **When scanning nodes for a suitable host – both the best matching node and the runner-up are noted**
- **The best will form the host for new instance unless merging host and runner-up produces better CU**

Final Hierarchy



Iris Data


Versicolor
Versicolor
Versicolor
Virginica
Versicolor
Versicolor
Versicolor
Versicolor
Versicolor
Versicolor
Versicolor
Virginica
Virginica
Virginica
Virginica
Virginica
Virginica
Virginica
Virginica
Virginica
Virginica
Setosa
Setosa
Setosa
Setosa
Setosa
Setosa
Setosa
Setosa
Setosa
Setosa

Category utility

- Category utility is a kind of quadratic loss function defined on conditional probabilities:

$$CU(C_1, C_2, \ldots, C_k) = \frac{\sum_l \Pr[C_l] \sum_i \sum_j \left(\Pr[a_i = v_{ij} \mid C_l]^2 - \Pr[a_i = v_{ij}]^2\right)}{k}$$

- C_1, ..C_k are k clusters
- a_i is the *i*th attribute
- Takes on values v_{i1}, v_{i2}, …

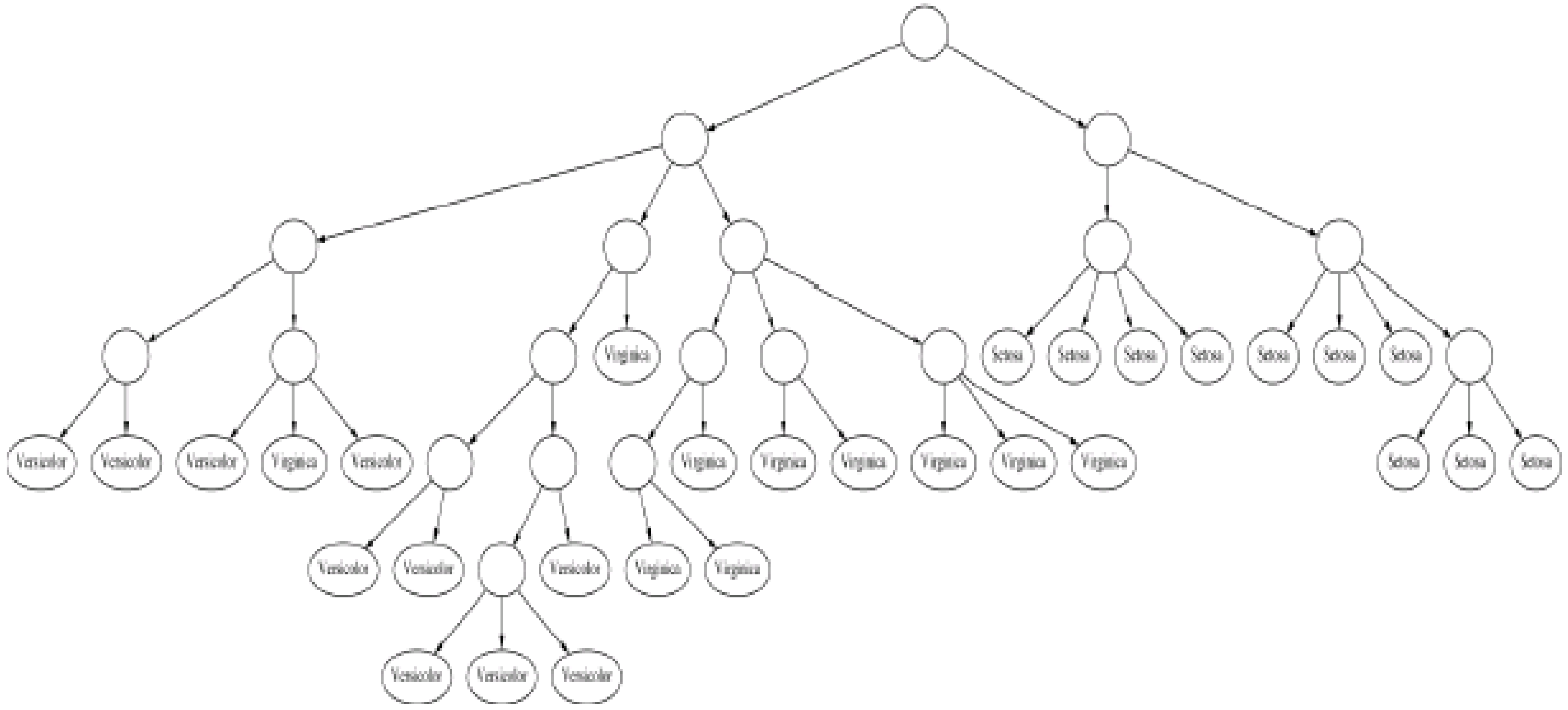
Category Utility Extended to Numeric Attributes

- **Assuming normal distribution:**

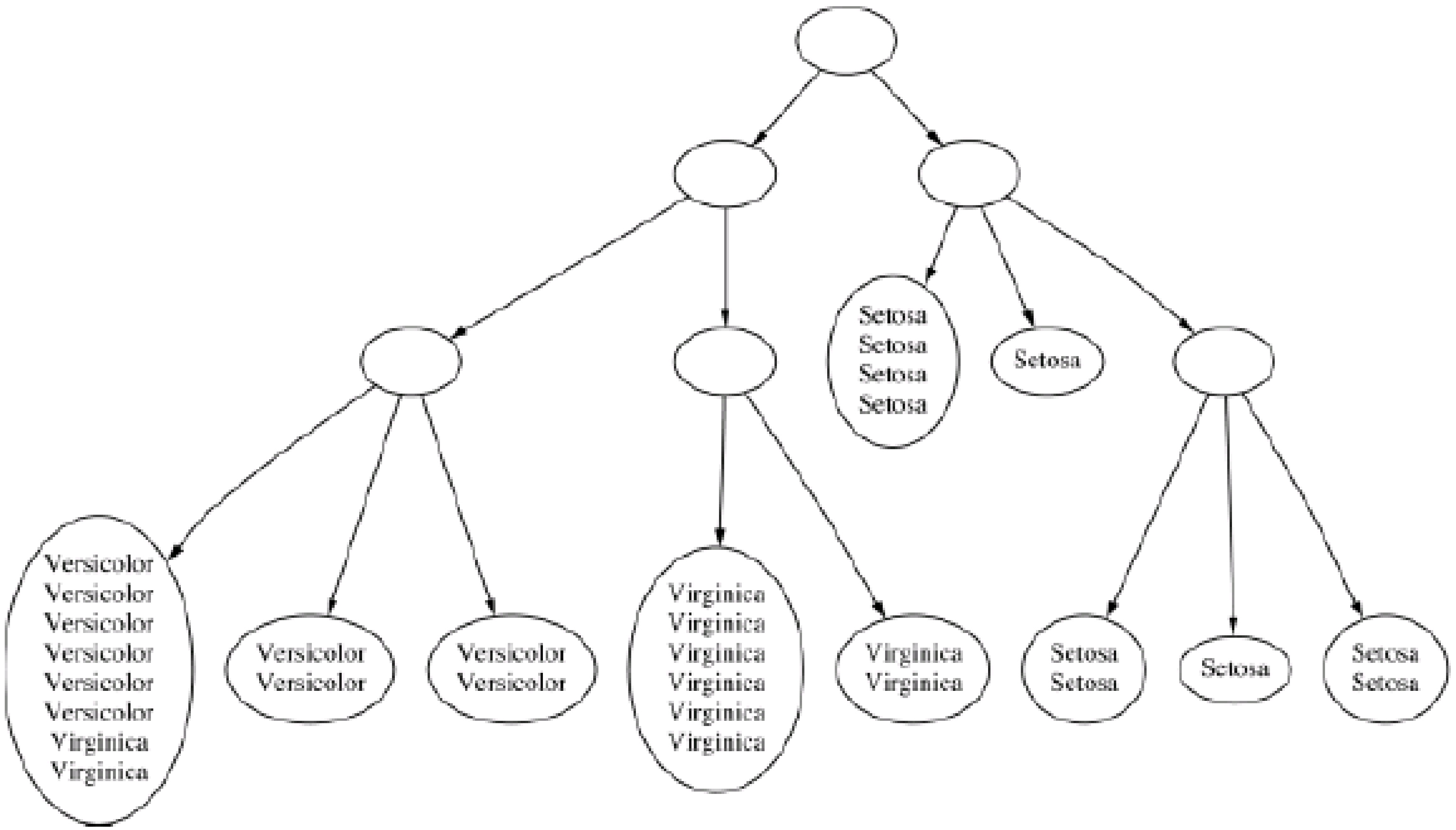
$$CU = \frac{\sum_l \Pr[C_l] \frac{1}{2\sqrt{\pi}} \sum_i \left(\frac{1}{\sigma_{il}} - \frac{1}{\sigma_i} \right)}{k}$$

- **When Standard deviation of attribute a_i is zero it produced infinite value of the category utility formula**
- **Acuity parameter: pre-specified minimum variance on each attribute**
  - only one instance in a node produces 0 variance

Iris Data Final Hierarchy



Clustering with Cutoff



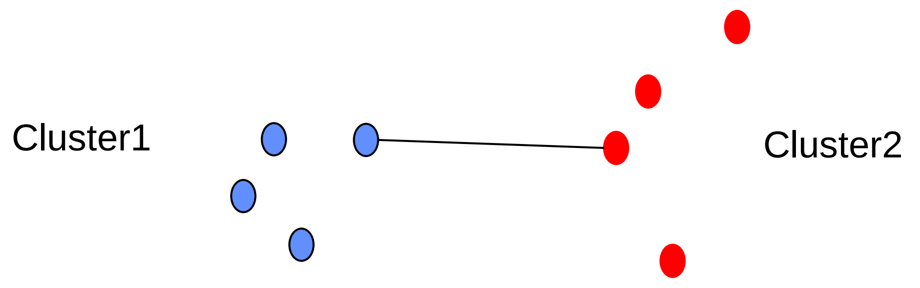
Incremental & Hierarchical Clustering

- **Clusters are merged/split according to distance or utility measure**
  - Euclidean distance (squared differences)
  - conditional probabilities (for nominal features)

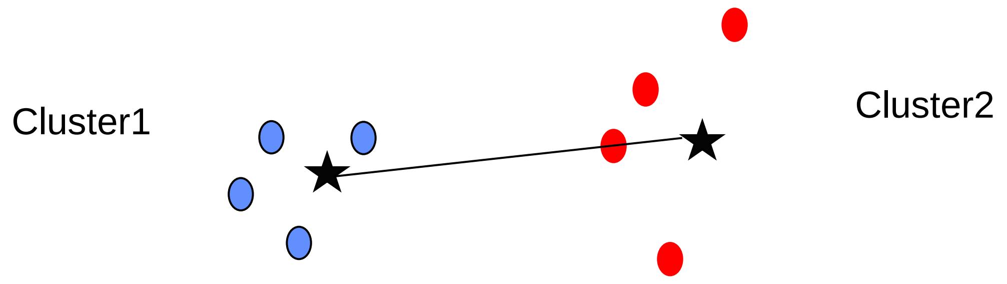
- **Options to choose which clusters to 'Link'**
  - single linkage, mean, average (w.r.t. points in clusters)
    (may lead to different trees, depending on spreads)
  - Ward method (smallest increase within cluster variance)
  - change in probability of features for given clusters

Linkage options

- e.g. single linkage (closest to any cluster instance)

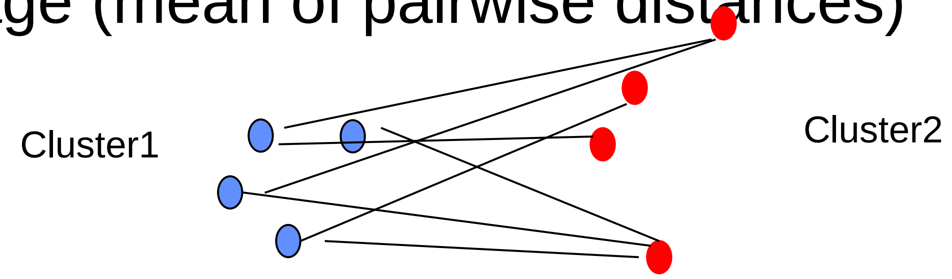


- e.g. mean (closest to mean of all cluster instances)

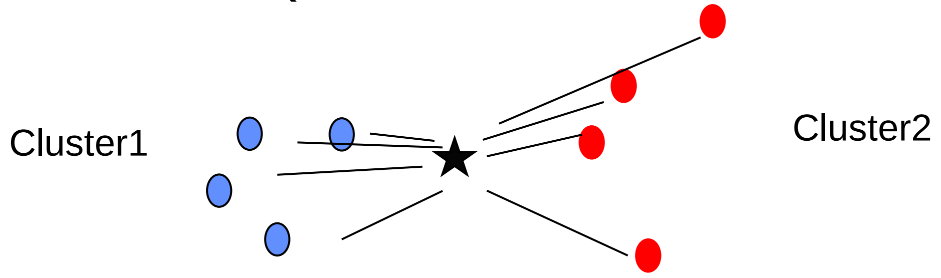


Linkage options (cont')

- e.g. average (mean of pairwise distances)



- e.g. Ward's method (find new cluster with min. variance)



Hierarchical Clustering Demo

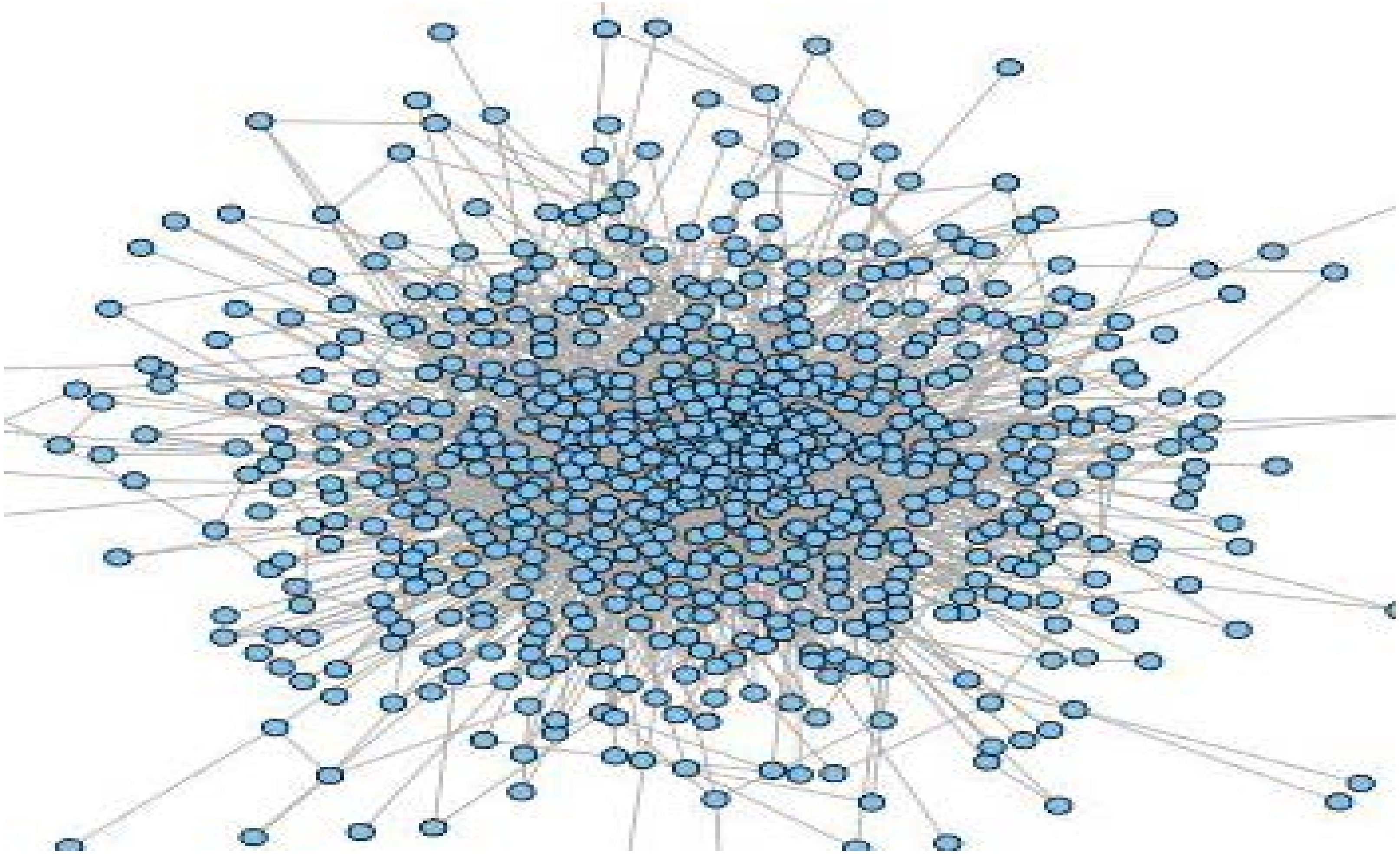
- **3888 Interactions among 685 proteins**

  From Hu et.al. TAP dataset
  http://www.compsysbio.org/bacteriome/dataset/)

b0009	b0014	0.92
b0009	b2231	0.87
b0014	b0169	1.0
b0014	b0595	0.76
b0014	b2614	1.0
b0014	b3339	0.95
b0014	b3636	0.9
b0015	b0014	0.99

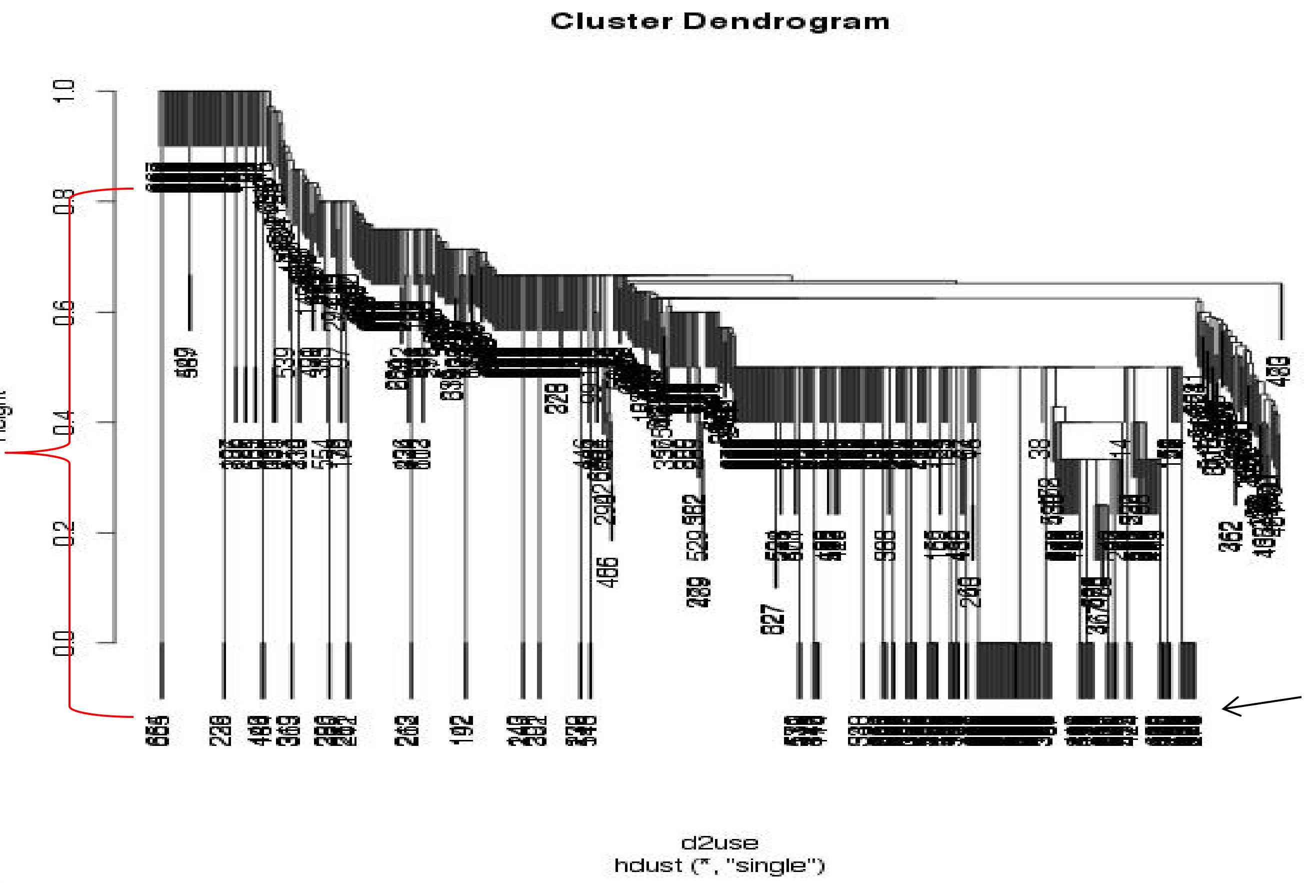
…….

Interactions as connections – structure is hard to see



Hierarchical Clustering Demo

- **hclust with "single" distance: chaining**



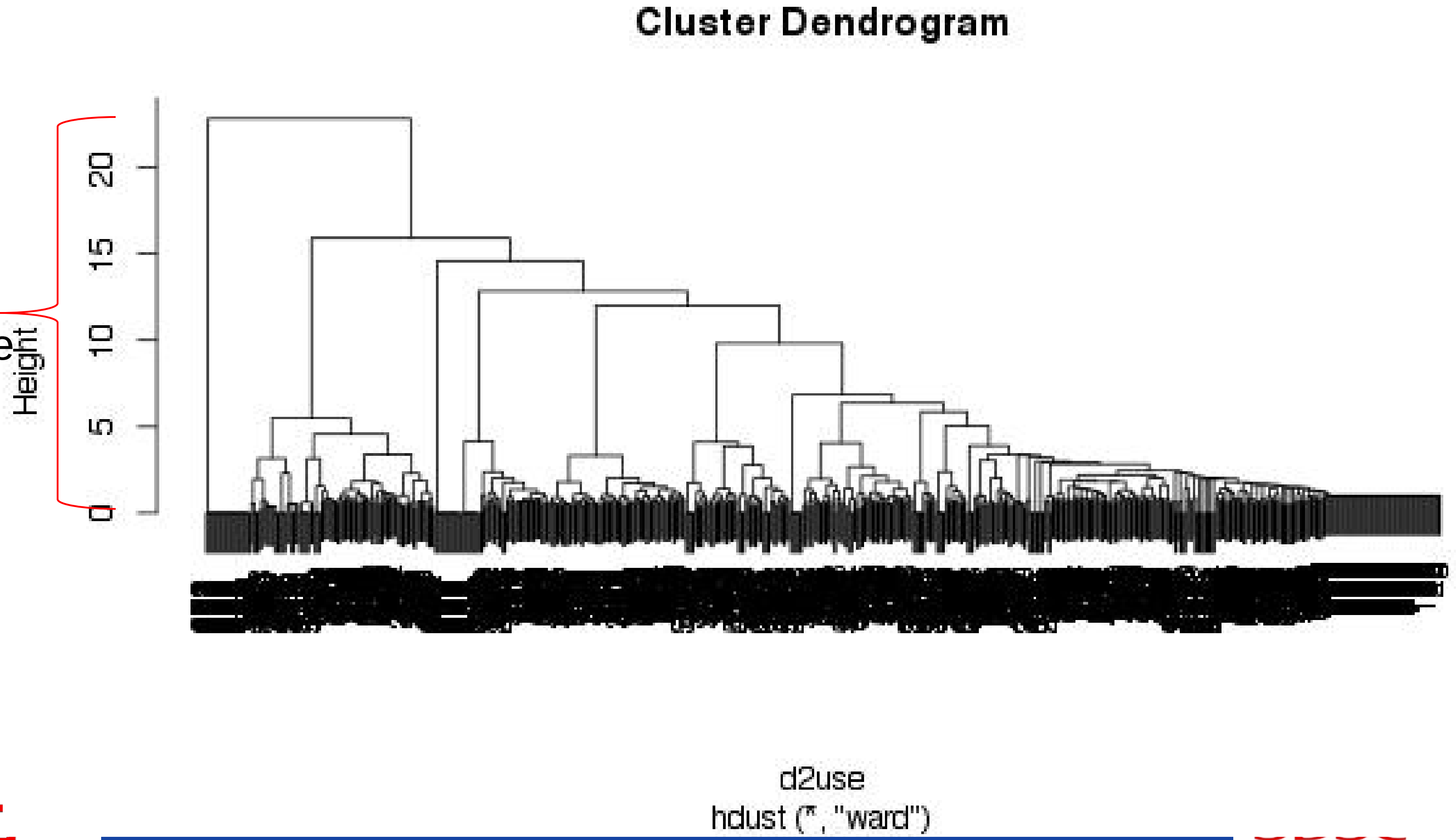
the cluster distance when 2 are combined

Items that cluster first

Hierarchical Clustering Demo

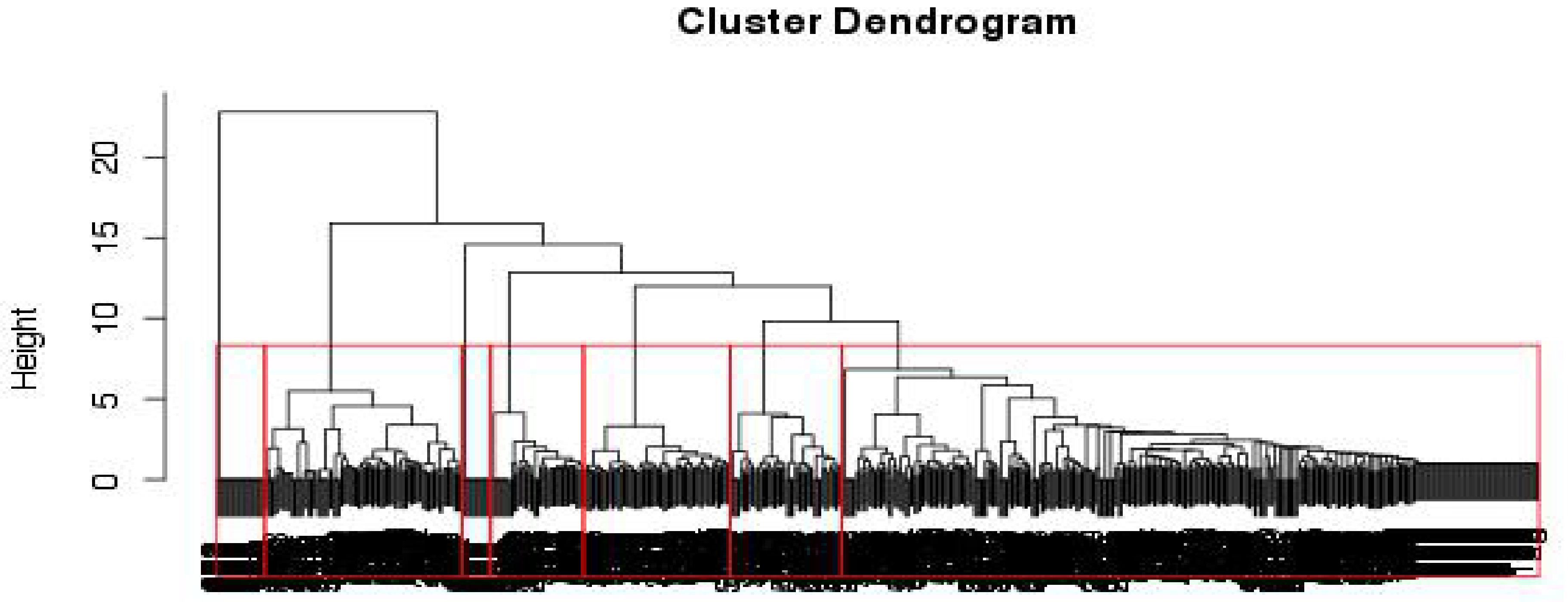
- **hclust with "Ward" distance: spherical clusters**

the cluster distance when 2 are combined



Hierarchical Clustering Demo

- **Where height change looks big, cut off tree**



Summary

- **Having no label doesn't stop you from finding structure in data**
- **Labeled clusters can be interpreted by using supervised learning - train a tree or learn rules**
- **Can be used to fill in missing attribute values**
- **All methods have a basic assumption of independence between the attributes**
    - Some methods allow the user to specify in advanced that two of more attributes are dependent and should be modeled with a joint probability
- **Unsupervised methods are somewhat related**

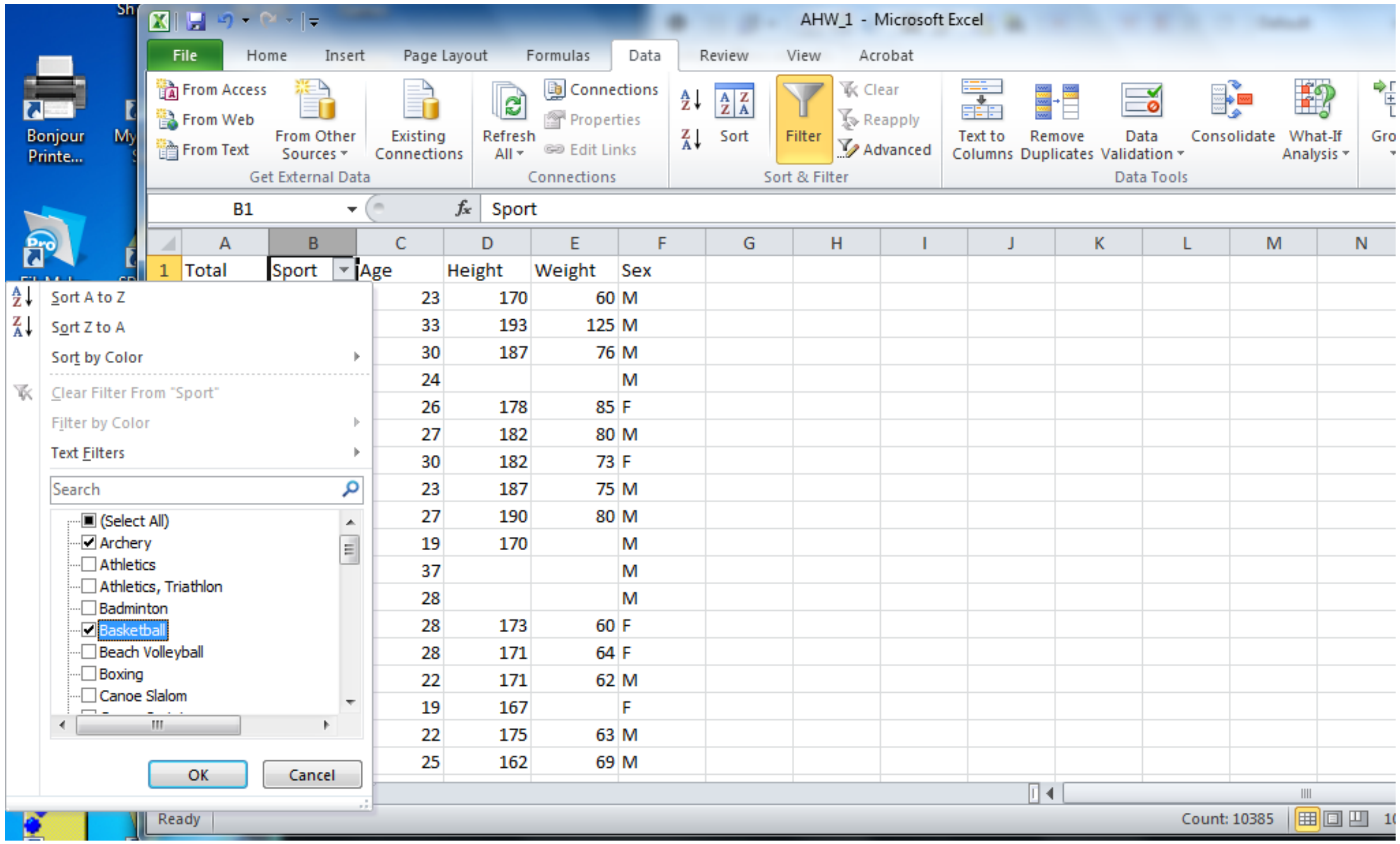
Exercise:
Clustering Athlete’s Data in Weka

Is there a relationship between athlete’s physical attributes and their sport?

Let’s filter data to try a few, good, candidate sport categories.

Weka not nice for picking out instances with multi-criteria, so I used excel:
(Excel, data -> advanced -> click on column and then drop down selection menu. Include only a few sports)

Download AHW_withHWbmi_cycbaskten2.csv from pace.sdsc.edu, open in weka

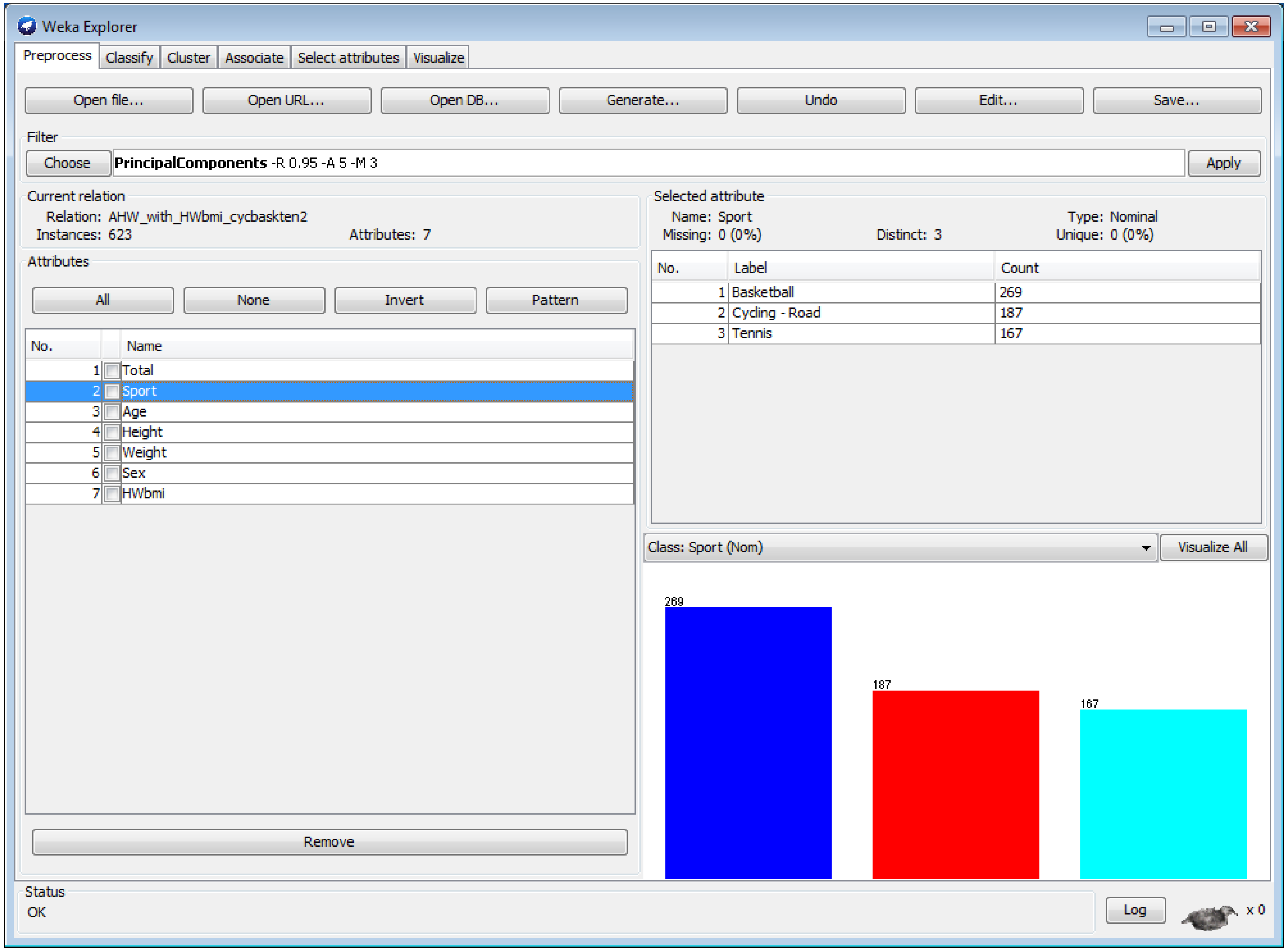


B1 fx Sport

- ■ (Select All)
- ☑ Archery
- ☐ Athletics
- ☐ Athletics, Triathlon
- ☐ Badminton
- ☑ Basketball
- ☐ Beach Volleyball
- ☐ Boxing
- ☐ Canoe Slalom

Total	Sport	Age	Height	Weight	Sex
		23	170	60	M
		33	193	125	M
		30	187	76	M
		24			M
		26	178	85	F
		27	182	80	M
		30	182	73	F
		23	187	75	M
		27	190	80	M
		19	170		M
		37			M
		28			M
		28	173	60	F
		28	171	64	F
		22	171	62	M
		19	167		F
		22	175	63	M
		25	162	69	M

Notice the number of sports and their nominal values



Filter: **PrincipalComponents** -R 0.95 -A 5 -M 3

Current relation
Relation: AHW_with_HWbmi_cycbaskten2
Instances: 623   Attributes: 7

Attributes

No.	Name
1	Total
2	Sport
3	Age
4	Height
5	Weight
6	Sex
7	HWbmi

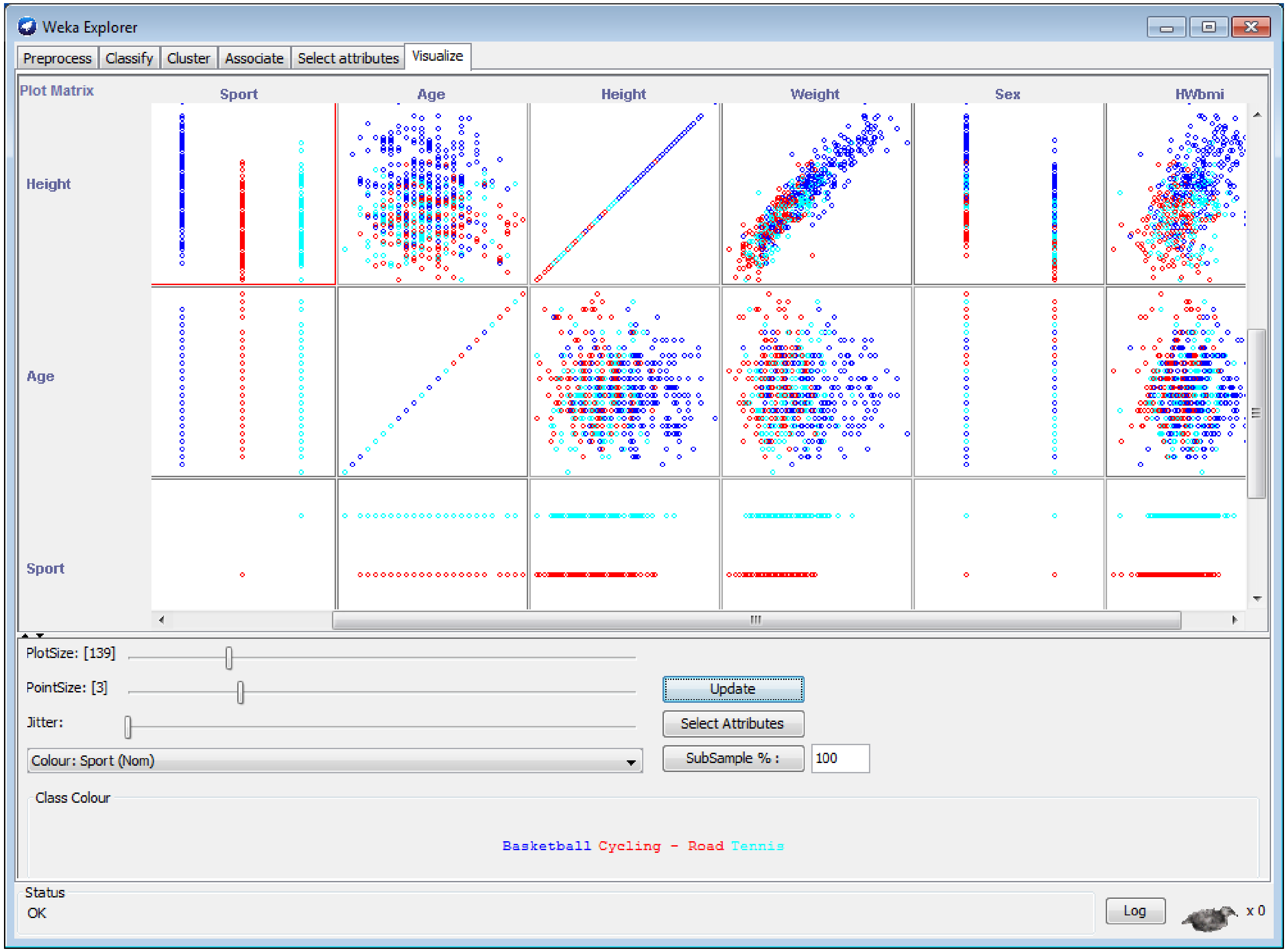
Selected attribute
Name: Sport   Type: Nominal
Missing: 0 (0%)   Distinct: 3   Unique: 0 (0%)

No.	Label	Count
1	Basketball	269
2	Cycling - Road	187
3	Tennis	167

Class: Sport (Nom)

Status
OK

- Visualize correlation with sport as class – any problems, any promising relationships?



PlotSize: [139]

PointSize: [3]

Jitter:

Colour: Sport (Nom)

SubSample % : 100

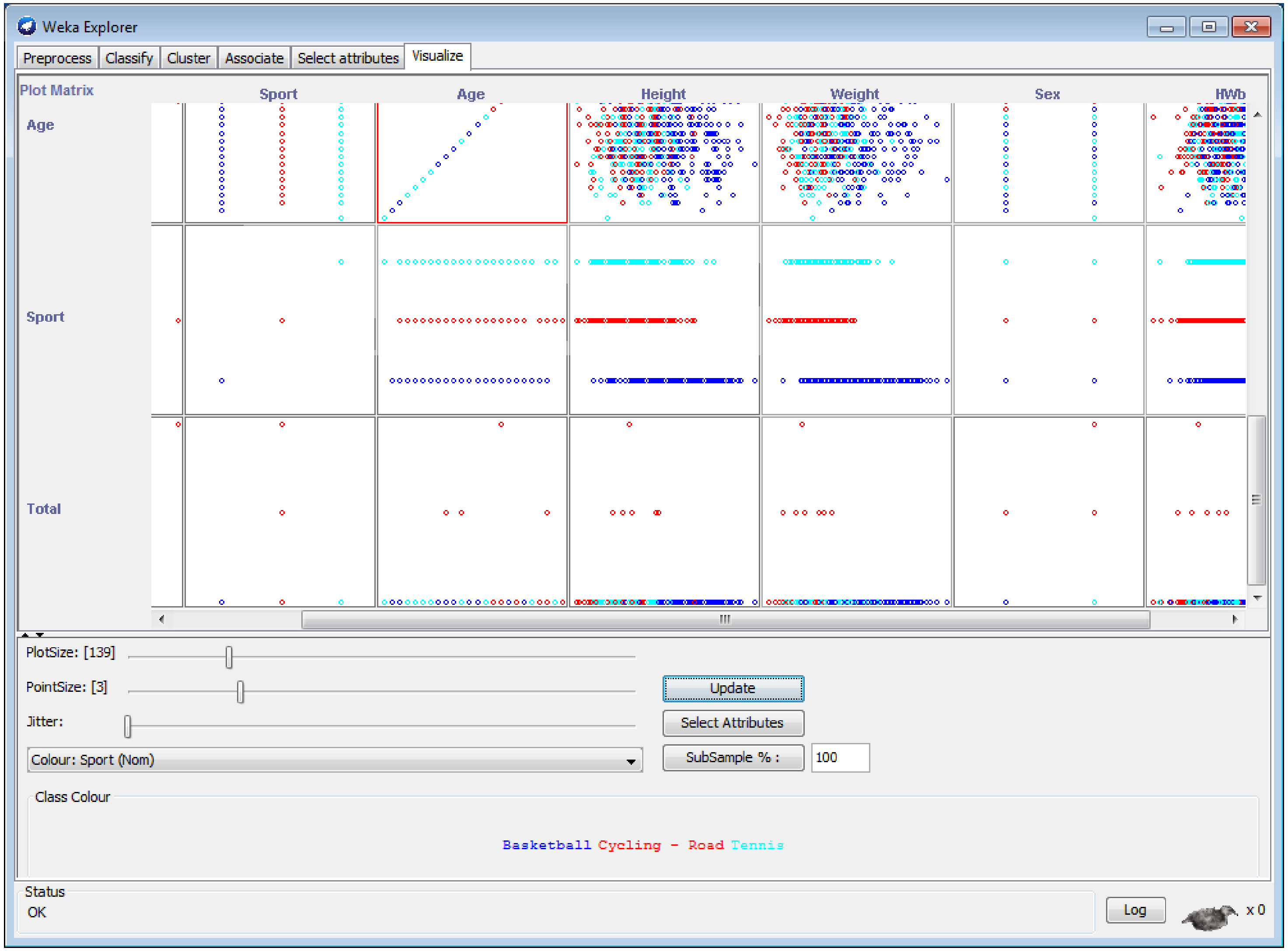
Class Colour

Basketball Cycling - Road Tennis

Status

OK

Are there one or more variables we should ignore?



PlotSize: [139]

PointSize: [3]

Jitter:

Colour: Sport (Nom)

SubSample % : 100

Class Colour

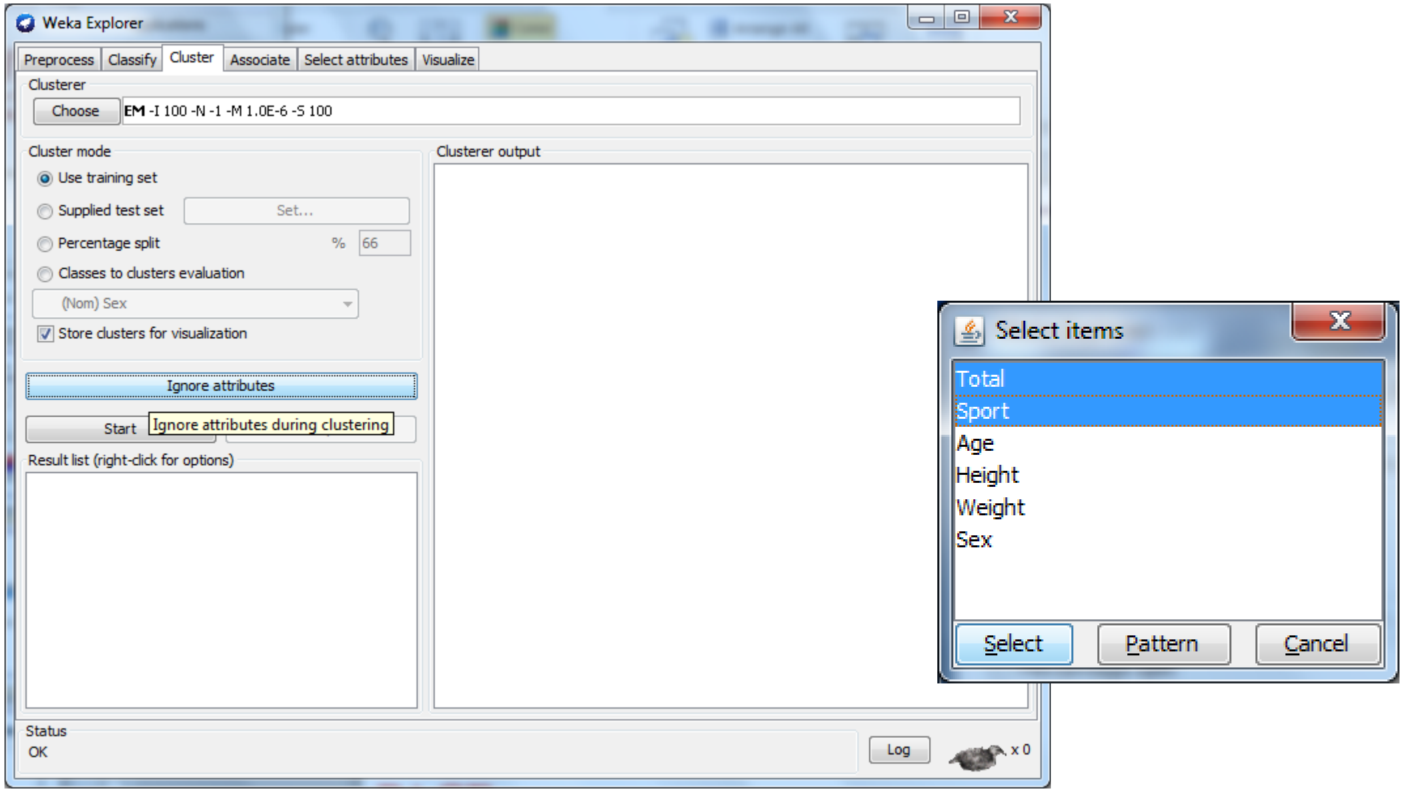
Basketball Cycling - Road Tennis

Status

OK

Predictive Analytics Center of Excellence

Select Ignore attributes -> ctrl-space to ignore sport (why?) and total medals won (why or why not?) What about sex attribute?



Clusterer: EM -I 100 -N -1 -M 1.0E-6 -S 100

Cluster mode
- (●) Use training set
- ( ) Supplied test set
- ( ) Percentage split % 66
- ( ) Classes to clusters evaluation
  - (Nom) Sex
- [✓] Store clusters for visualization

Ignore attributes during clustering

Clusterer output

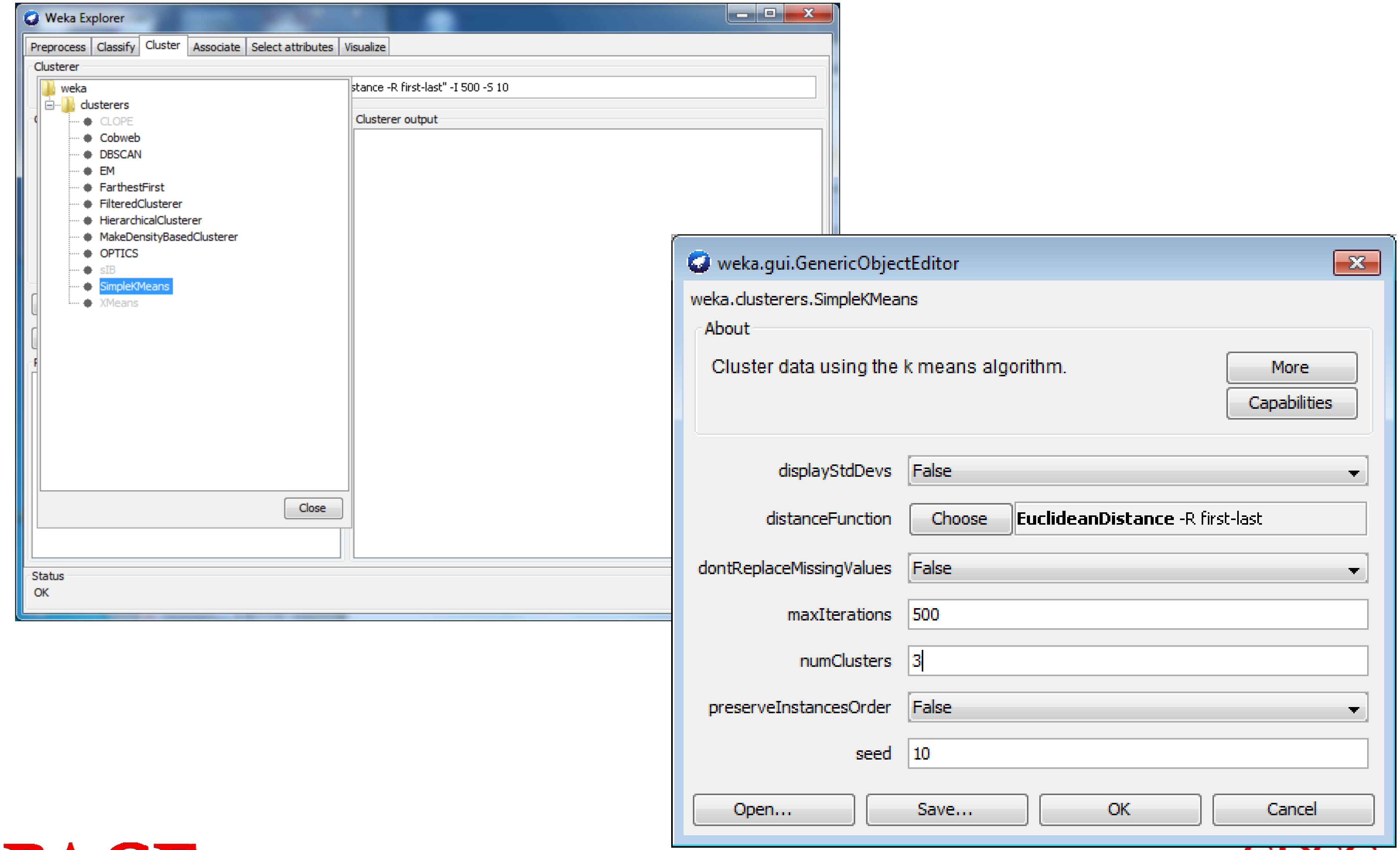
Result list (right-click for options)

Status
OK

x 0

Select items:
- Total
- Sport
- Age
- Height
- Weight
- Sex

select cluster tab; choose -> simple Kmeans ; choose 3 classes, accept other defaults



stance -R first-last" -I 500 -S 10

weka.gui.GenericObjectEditor

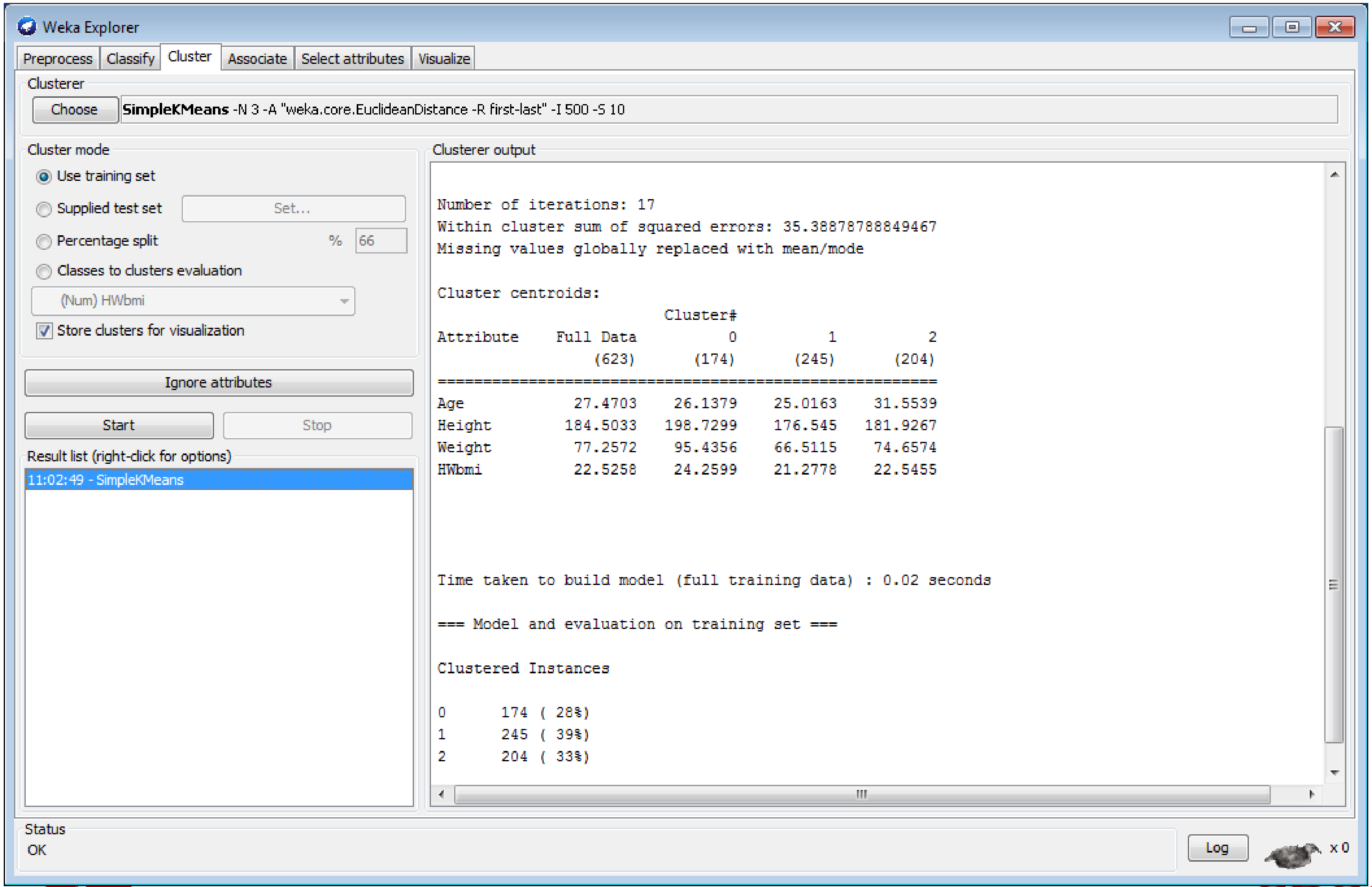
weka.clusterers.SimpleKMeans

About

Cluster data using the k means algorithm.

displayStdDevs	False
distanceFunction	**EuclideanDistance** -R first-last
dontReplaceMissingValues	False
maxIterations	500
numClusters	3
preserveInstancesOrder	False
seed	10

Results end up in output panel. Take note of sum squared error. Any other message to note?



SimpleKMeans -N 3 -A "weka.core.EuclideanDistance -R first-last" -I 500 -S 10

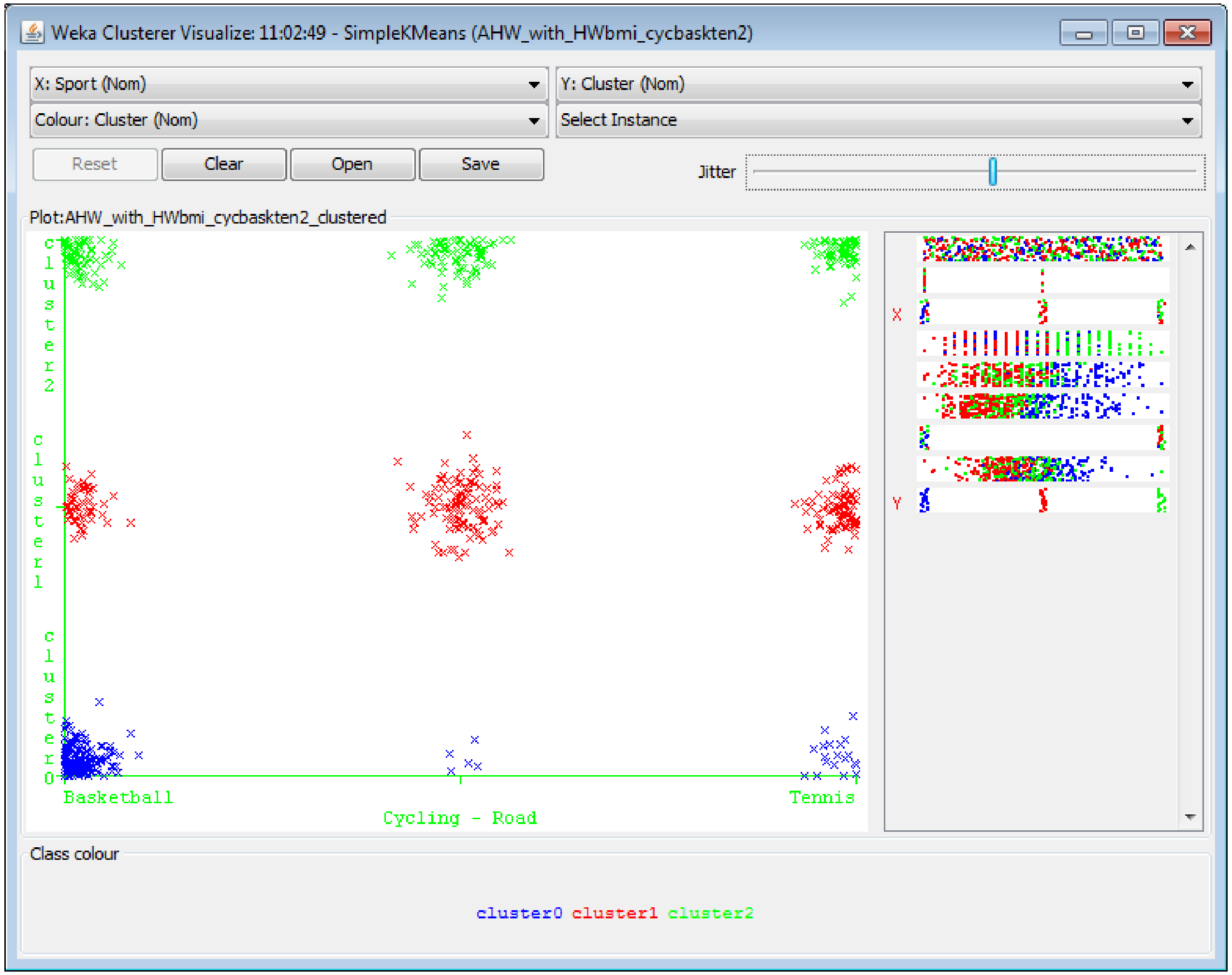
Cluster mode:
- [✓ Use training set]
- Supplied test set
- Percentage split % 66
- Classes to clusters evaluation: (Num) HWbmi
- [✓ Store clusters for visualization]

Result list: 11:02:49 - SimpleKMeans

```
Number of iterations: 17
Within cluster sum of squared errors: 35.38878788849467
Missing values globally replaced with mean/mode

Cluster centroids:
                              Cluster#
Attribute     Full Data          0          1          2
                  (623)      (174)      (245)      (204)
=======================================================
Age             27.4703    26.1379    25.0163    31.5539
Height         184.5033   198.7299    176.545   181.9267
Weight          77.2572    95.4356    66.5115    74.6574
HWbmi           22.5258    24.2599    21.2778    22.5455




Time taken to build model (full training data) : 0.02 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       174 ( 28%)
1       245 ( 39%)
2       204 ( 33%)
```

Status: OK

Rt click on model result, visualize cluster assignment, add jitter
Compare sport and cluster number – how do they correspond?
Are the clusters useful to distinguish (or predict) an athlete's sport?


Weka Clusterer Visualize: 11:02:49 - SimpleKMeans (AHW_with_HWbmi_cycbaskten2)

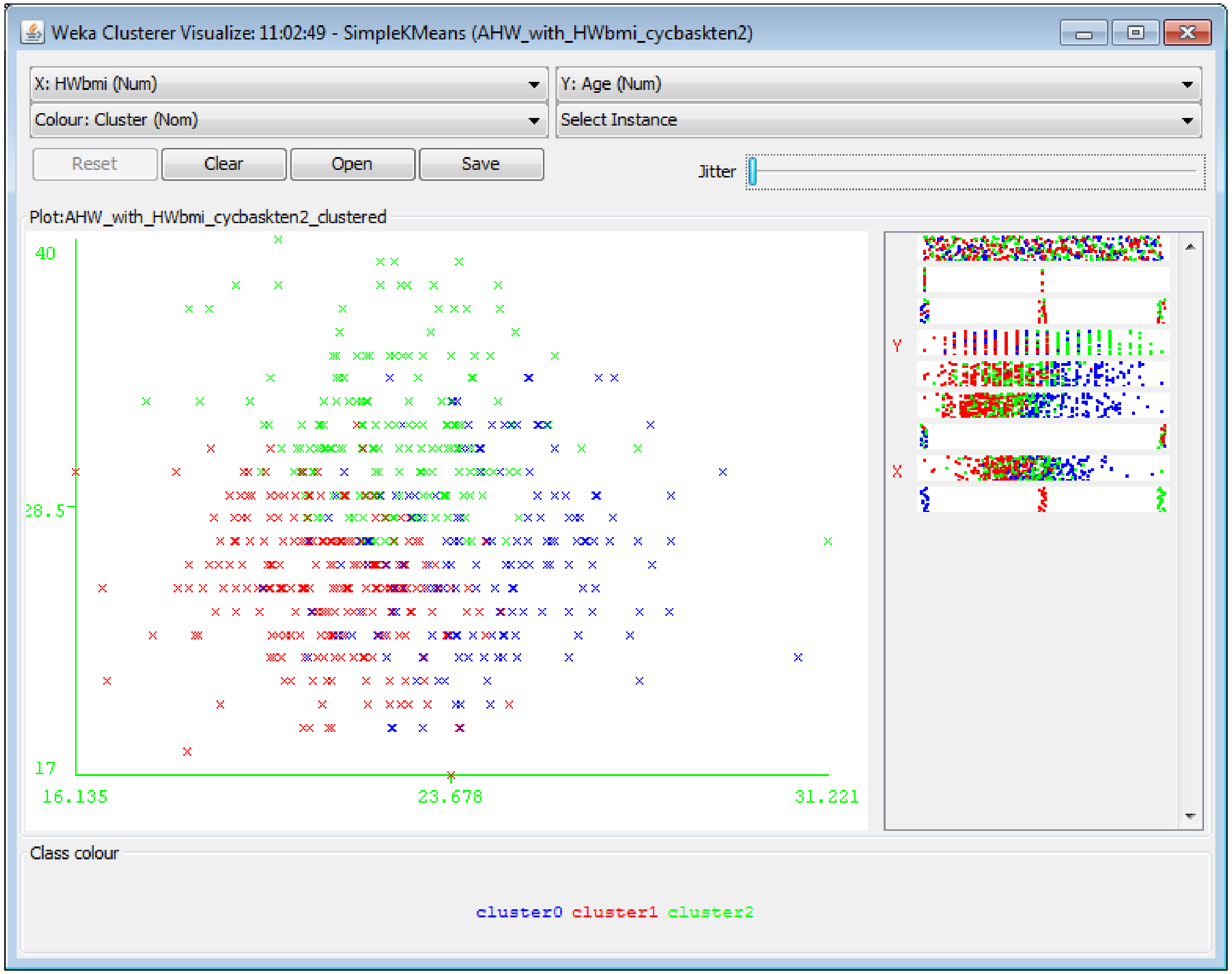
X: Sport (Nom)
Y: Cluster (Nom)
Colour: Cluster (Nom)
Select Instance
Jitter

Plot:AHW_with_HWbmi_cycbaskten2_clustered

Class colour

cluster0 cluster1 cluster2

Compare clusters in the 2 dimension subspace of BMI and Age - (or any other subspace)


Weka Clusterer Visualize: 11:02:49 - SimpleKMeans (AHW_with_HWbmi_cycbaskten2)

X: HWbmi (Num)

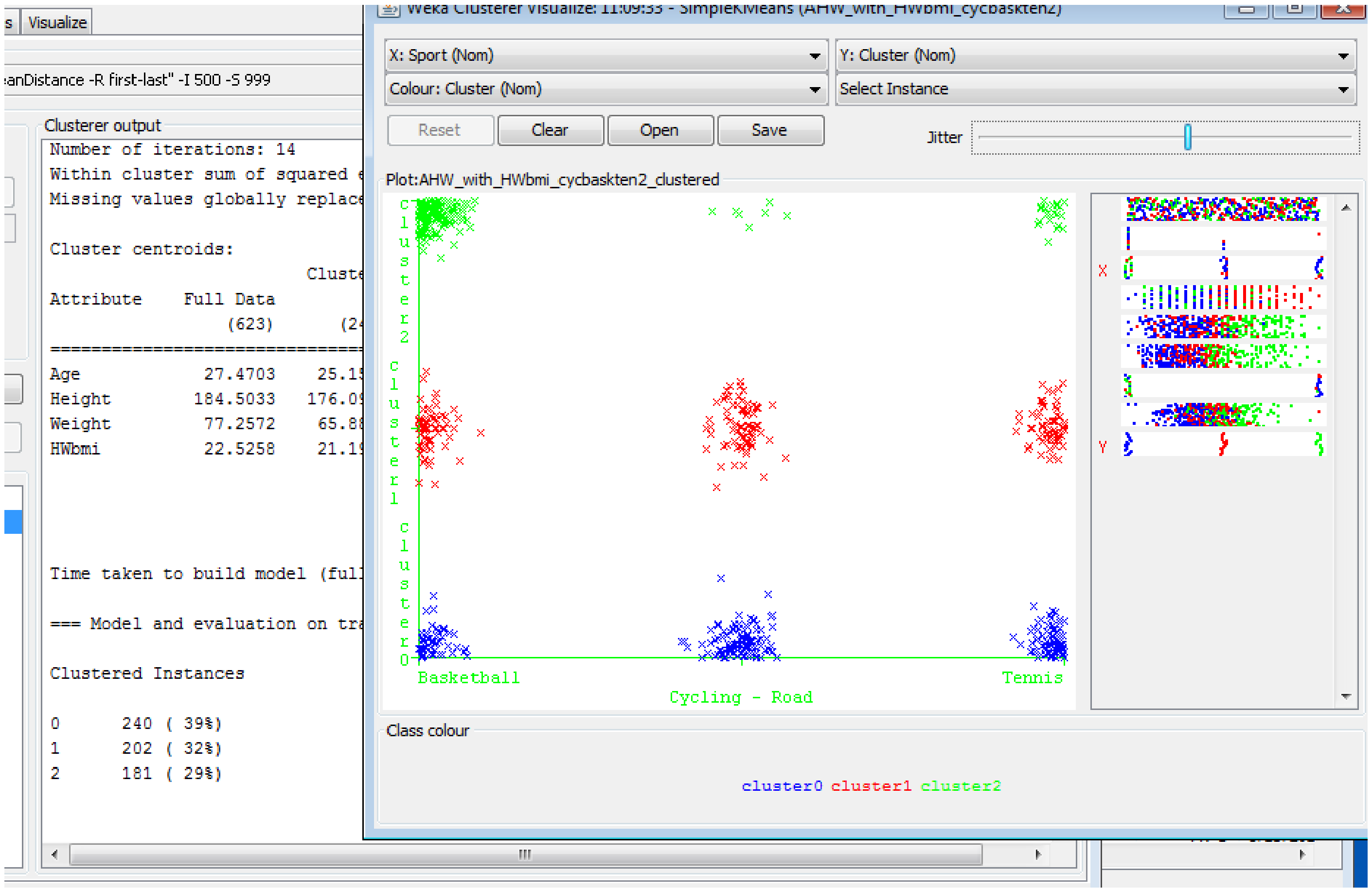
Y: Age (Num)

Colour: Cluster (Nom)

Select Instance

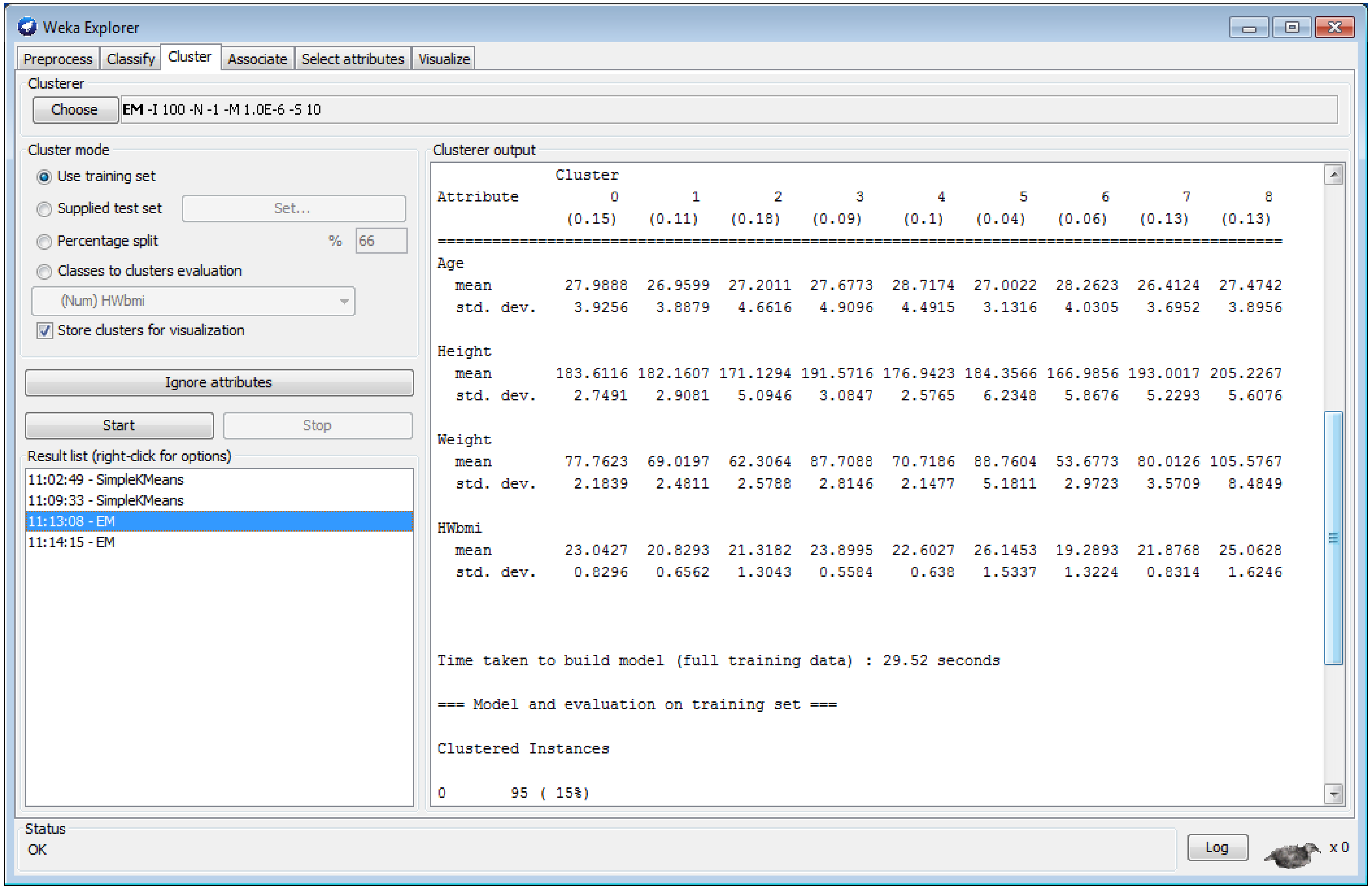
Reset
Clear
Open
Save
Jitter
Plot:AHW_with_HWbmi_cycbaskten2_clustered
40
28.5
17
16.135
23.678
31.221
Y
X
Class colour
cluster0 cluster1 cluster2

Try rerunning with different random seed, you get different clustering, but similar SSE



```
eanDistance -R first-last" -I 500 -S 999

Clusterer output
Number of iterations: 14
Within cluster sum of squared e
Missing values globally replace

Cluster centroids:
                           Cluste
Attribute      Full Data
                   (623)       (24
=================================
Age              27.4703    25.15
Height          184.5033   176.09
Weight           77.2572    65.88
HWbmi            22.5258    21.19



Time taken to build model (full

=== Model and evaluation on tra

Clustered Instances

0       240 ( 39%)
1       202 ( 32%)
2       181 ( 29%)
```

Try EM algorithm, accept defaults and start



Clusterer: EM -I 100 -N -1 -M 1.0E-6 -S 10

Cluster mode
- [✓ Use training set]
- Supplied test set
- Percentage split % 66
- Classes to clusters evaluation (Num) HWbmi
- [✓ Store clusters for visualization]

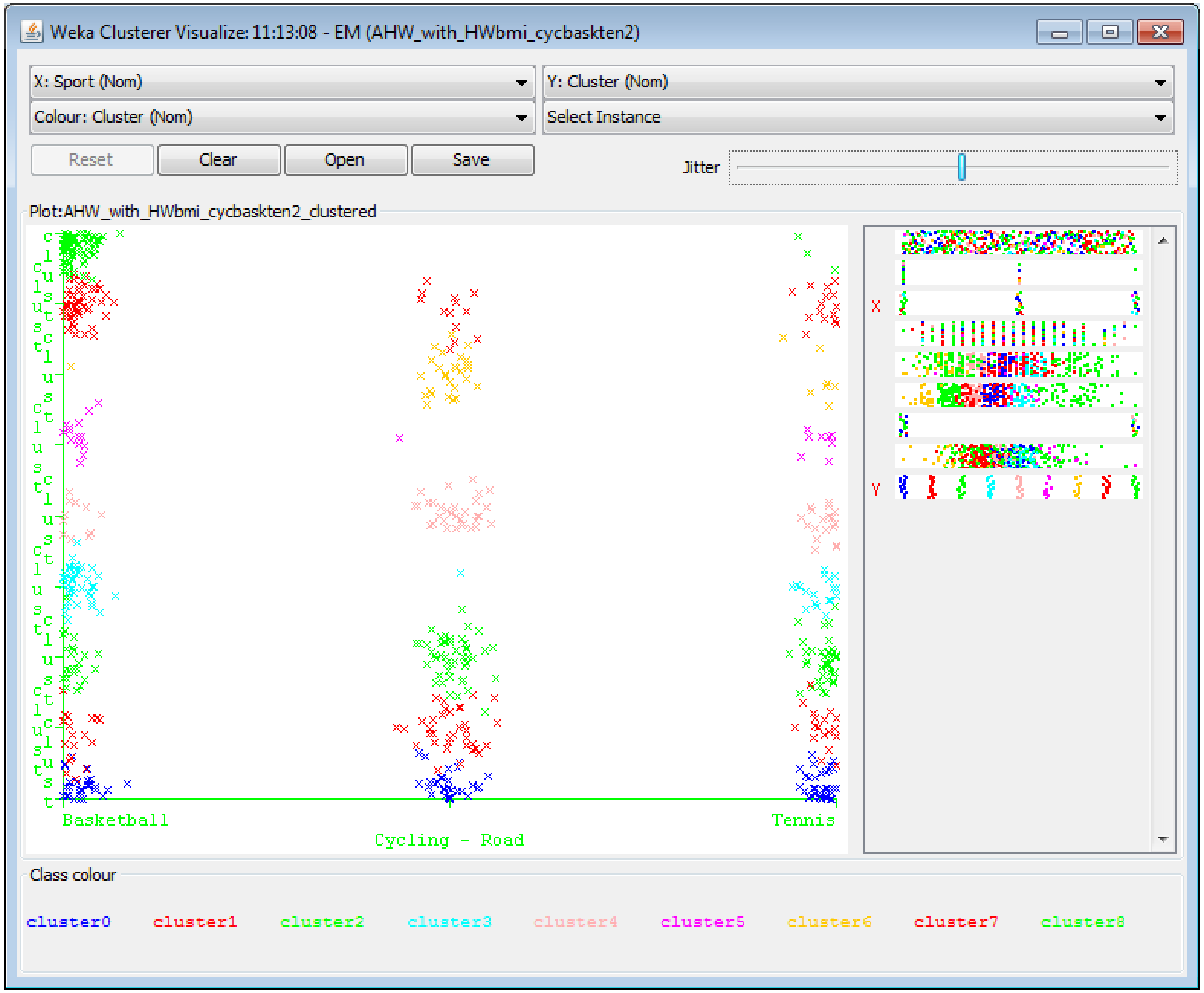
Result list (right-click for options)
11:02:49 - SimpleKMeans
11:09:33 - SimpleKMeans
11:13:08 - EM
11:14:15 - EM

Clusterer output

```
                 Cluster
Attribute              0        1        2        3        4        5        6        7        8
                  (0.15)   (0.11)   (0.18)   (0.09)    (0.1)   (0.04)   (0.06)   (0.13)   (0.13)
=============================================================================================
Age
  mean           27.9888  26.9599  27.2011  27.6773  28.7174  27.0022  28.2623  26.4124  27.4742
  std. dev.       3.9256   3.8879   4.6616   4.9096   4.4915   3.1316   4.0305   3.6952   3.8956

Height
  mean          183.6116 182.1607 171.1294 191.5716 176.9423 184.3566 166.9856 193.0017 205.2267
  std. dev.       2.7491   2.9081   5.0946   3.0847   2.5765   6.2348   5.8676   5.2293   5.6076

Weight
  mean           77.7623  69.0197  62.3064  87.7088  70.7186  88.7604  53.6773  80.0126 105.5767
  std. dev.       2.1839   2.4811   2.5788   2.8146   2.1477   5.1811   2.9723   3.5709   8.4849

HWbmi
  mean           23.0427  20.8293  21.3182  23.8995  22.6027  26.1453  19.2893  21.8768  25.0628
  std. dev.       0.8296   0.6562   1.3043   0.5584    0.638   1.5337   1.3224   0.8314   1.6246



Time taken to build model (full training data) : 29.52 seconds

=== Model and evaluation on training set ===

Clustered Instances

0       95 ( 15%)
```

Status
OK

Rt click on model result, visualize cluster assignment, Compare sport and cluster number


Weka Clusterer Visualize: 11:13:08 - EM (AHW_with_HWbmi_cycbaskten2)

X: Sport (Nom)

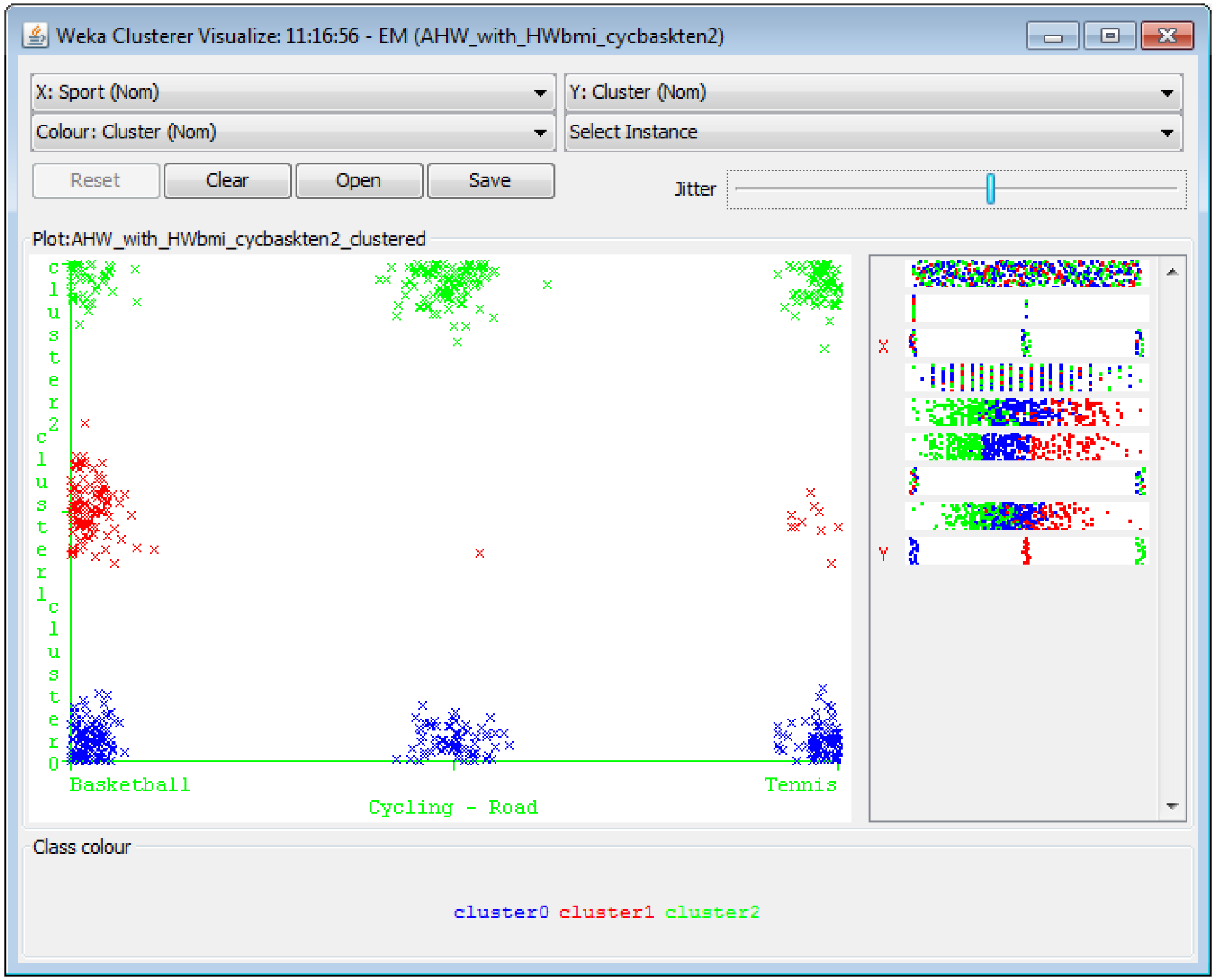
Y: Cluster (Nom)

Colour: Cluster (Nom)

Select Instance

Plot:AHW_with_HWbmi_cycbaskten2_clustered

Rerun with 5 clusters, compare sport and cluster number


Weka Clusterer Visualize: 11:16:56 - EM (AHW_with_HWbmi_cycbaskten2)

X: Sport (Nom)

Y: Cluster (Nom)

Colour: Cluster (Nom)

Select Instance

Plot:AHW_with_HWbmi_cycbaskten2_clustered

Class colour

cluster0 cluster1 cluster2

Exercise:

For simple Kmeans
Get a full sweep of K – ie change N to 1,3,4,5,
what K would you choose?