CSCI446/946 Big Data Analytics

Week 4 – Lecture: Clustering

School of Computing and Information Technology
University of Wollongong Australia
Spring 2022

Content

- Brief Recap
  - Hypothesis Testing
- Statistical Methods for Evaluation
  - Analysis of Variance (ANOVA)
- Clustering Analysis
  - K-means, DBSCAN, SOM

Content

- Brief Recap
  - Hypothesis Testing

- Statistical Methods for Evaluation
  - Analysis of Variance (ANOVA)
- Clustering Analysis
  - K-means, DBSCAN, SOM

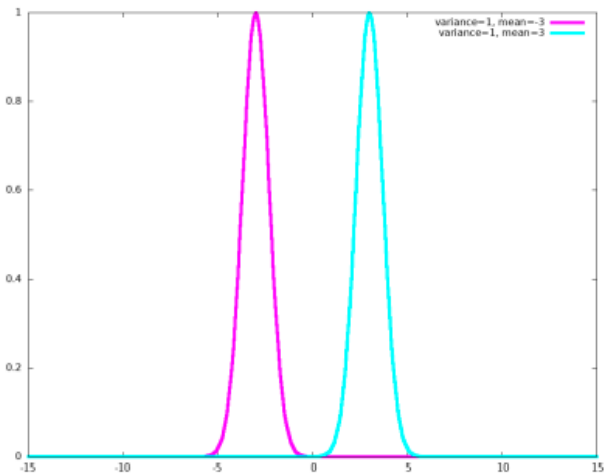
Statistical Methods for Evaluation

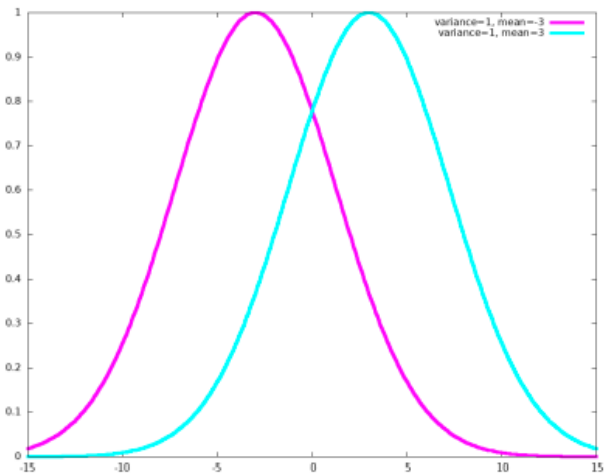
- Hypothesis Testing
  - Form an assertion and test it with data
  - Common assumption (there is no difference)
  - Null hypothesis (H_0) vs Alternative hypothesis (H_A)

- A hypothesis is formed before validation
  - It can define expectations.

Statistical Methods for Evaluation

- Analysis of the difference of two Means
  - Very common hypothesis test.
  - But simple comparison is often not sufficient.
  - Example: Assume we have two populations, one with mean=-3 and the other with mean=3
    - By comparing the means can we say that the difference between the two **populations** is significant?
    - Answer depends on variance.





Statistical Methods for Evaluation

- Student's t-test
  - Assumptions: Two populations, normally distributed and have a similar variance.
- Welch's t-test
  - Assumptions: Two populations, normally distributed.
- Wilcoxon Rank-Sum Test
  - Assumptions Two populations, not normal distributed.

- ANOVA
  - When: More than two populations.

- Many others: Mann-Whitney test, Kruskal-Wallis test, Fisher's exact test, chi-square test, McNemartest, Friedman test, log rank test, spearman correlation test, Pearson correlation test….

Statistical Methods for Evaluation

- Wilcoxon Rank-Sum Test – Suppose we have the following data:
  - Group A: [85, 80, 78, 90, 95]; Group B: [88, 82, 85, 87, 92]
- Step 1: Combine and Rank the Data
  - Combine: [85, 80, 78, 90, 95, 88, 82, 85, 87, 92]
  - Rank: [4.5, 2, 1, 8, 10, 7, 3, 4.5, 6, 9]
- Step 2: Sum the Ranks for Each Group
  - Group A: [4.5, 2, 1, 8, 10]; Sum of ranks W_1=4.5+2+1+8+10=25.5
  - Group B: [7, 3, 4.5, 6, 9]; Sum of ranks W_2=6+3+4.5+5+9=29.5
- Step 3: Choose the Test Statistic
  - W can be either 25.5 or 29.5 depending on the test design, but usually, the smaller sum is used if conducting a one-sided test.
- Step 4: Determine Significance
  - Compare the test statistic W to a critical value from the Wilcoxon rank-sum distribution or use a p-value from statistical software.

Brief Recap

Content

- Brief Recap
  - Hypothesis Testing
- Statistical Methods for Evaluation
  - Analysis of Variance (ANOVA)
- Clustering Analysis
  - K-means, DBSCAN, SOM

Statistical Methods for Evaluation

- ANOVA (Analysis of Variance)
  - What if there are more than two populations?
  - Multiple *t*-test may not perform well now
- A generalization of the hypothesis testing
  - ANOVA tests if any of the population means differ from the other population means
  - Each population is assumed to be normal and have the same variance

Statistical Methods for Evaluation

- ANOVA (Analysis of Variance)

$$\boldsymbol{H_0}: \mu_1 = \mu_2 = \ldots = \mu_n$$

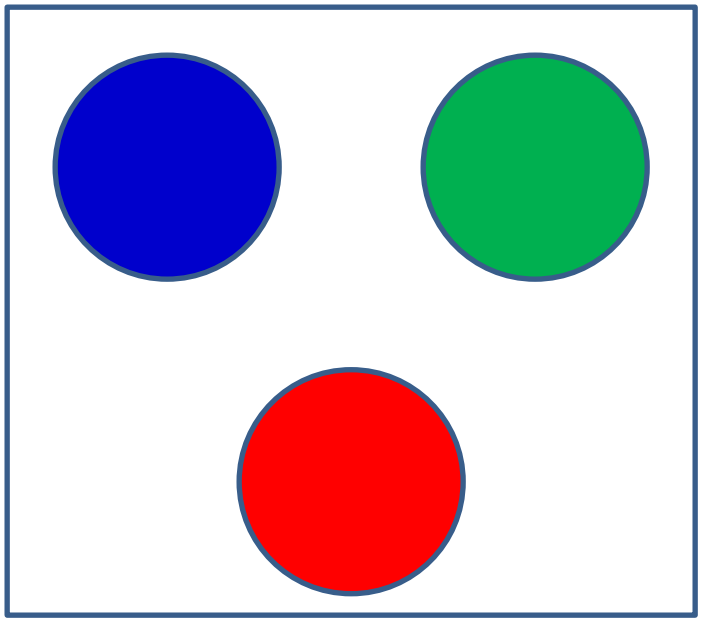
$$\boldsymbol{H_A}: \mu_i \neq \mu_j \text{ for at least one pair of } i, j$$

- Compute *F*-test statistic
  - Between-groups mean sum of squares
  - Within-groups mean sum of squares

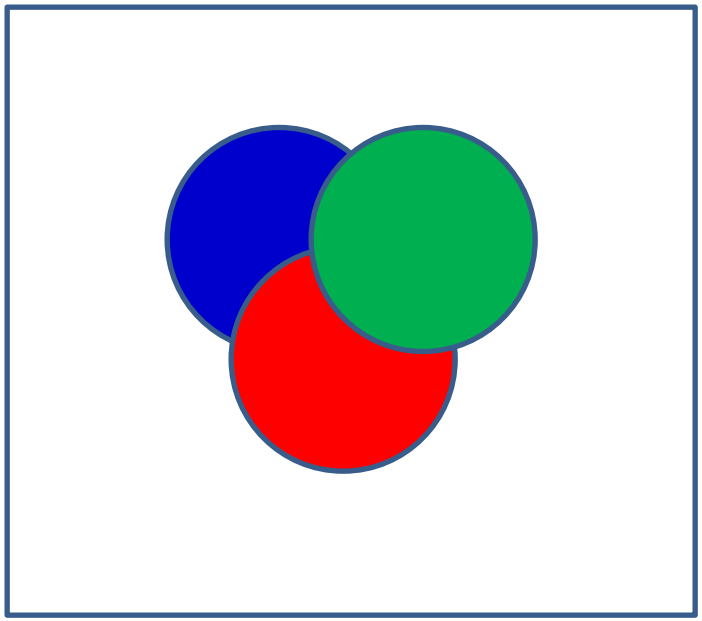
$$S_B^2 = \frac{1}{k-1}\sum_{i=1}^{k} n_i \cdot (\bar{x}_i - \bar{x}_0)^2 \qquad S_W^2 = \frac{1}{n-k}\sum_{i=1}^{k}\sum_{j=1}^{n_i}(x_{ij} - \bar{x}_i)^2$$

Statistical Methods for Evaluation

- ANOVA (Analysis of Variance)



$$F = \frac{S_B^2}{S_W^2}$$



$$S_B^2 = \frac{1}{k-1}\sum_{i=1}^{k} n_i \cdot (\bar{x}_i - \bar{x}_0)^2 \qquad S_W^2 = \frac{1}{n-k}\sum_{i=1}^{k}\sum_{j=1}^{n_i}(x_{ij} - \bar{x}_i)^2$$

Statistical Methods for Evaluation

- ANOVA (Analysis of Variance)
  - Measures how different the means are relative to the variability within each group
  - The larger the *F*-test statistic, the greater the likelihood that the difference of means are due to something other than chance alone
  - The *F*-test statistic follows an *F*-distribution

$$F = \frac{S_B^2}{S_W^2}$$

Statistical Methods for Evaluation

- ANOVA (Analysis of Variance)

```
# fit ANOVA test
model <- aov(purchase_amt ~ offers, data=offertest)

summary(model)
             Df Sum Sq Mean Sq F value Pr(>F)
offers        2 225222  112611   130.6 <2e-16 ***
Residuals   497 428470     862
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

- Shall we accept or reject the null hypothesis?

Statistical Methods for Evaluation

- ANOVA (Analysis of Variance)
  - One-Way ANOVA
    - Compares means across different groups based on a single independent variable (factor).
    - e.g. Comparing the mean test scores of students across different teaching methods (Method A, Method B, Method C).
  - Two-Way ANOVA:
    - Compares means across groups based on two independent variables (factors), and can also evaluate the interaction between the factors.
    - e.g. Comparing test scores based on teaching methods (Factor 1) and study times (Factor 2).

Statistical Methods for Evaluation

- Limitations of ANOVA (Analysis of Variance)
  - Assumptions
    - Normality: Data should be approximately normally distributed.
    - Homogeneity of Variances: Variances within each group should be equal (tested using Levene's test).
    - Independence: Observations should be independent of each other.
  - Limitations:
    - Sensitivity to Outliers: Outliers can affect the F-statistic and lead to misleading results.
    - Assumes Equal Variances: Violations of this assumption can impact the validity of the results.
    - Identifies Differences but Not Specifics: ANOVA indicates whether a difference exists but does not specify which groups are different without further tests (post-hoc).

Statistical Methods for Evaluation

- ANOVA (Analysis of Variance)
  - Additional tests for each pair of groups
  - Tukey's Honest Significant Difference (HSD)

```
TukeyHSD(model)
  Tukey multiple comparisons of means
    95% family-wise confidence level

Fit: aov(formula = purchase_amt ~ offers, data = offertest)

$offers
                    diff        lwr      upr     p adj
offer1-nopromo 40.961437 33.4638483 48.45903 0.0000000
offer2-nopromo 48.120286 40.5189446 55.72163 0.0000000
offer2-offer1   7.158849 -0.4315769 14.74928 0.0692895
```

Statistical Methods for Evaluation

- Tukey's Honest Significant Difference (HSD)
  - Assumptions
    - Norm + equal variance + sample sizes are approximately equal (though it can still be used if they are not).
  - Perform ANOVA test
    - establish whether there is a significant difference between the means of the groups
  - Calculation of the HSD
    - Critical value from studentized range distribution, Mean square within groups(from ANOVA), number of groups
  - Decision Rule
    - For each pair of means, calculate the absolute difference.
    - Compare the absolute difference to the HSD value.
    - If the absolute difference is greater than the HSD, the pair of means is considered significantly different.

Content

- Brief Recap
  - Hypothesis Testing
- Statistical Methods for Evaluation
  - Analysis of Variance (ANOVA)
- Clustering Analysis
  - K-means, DBSCAN, SOM

Clustering

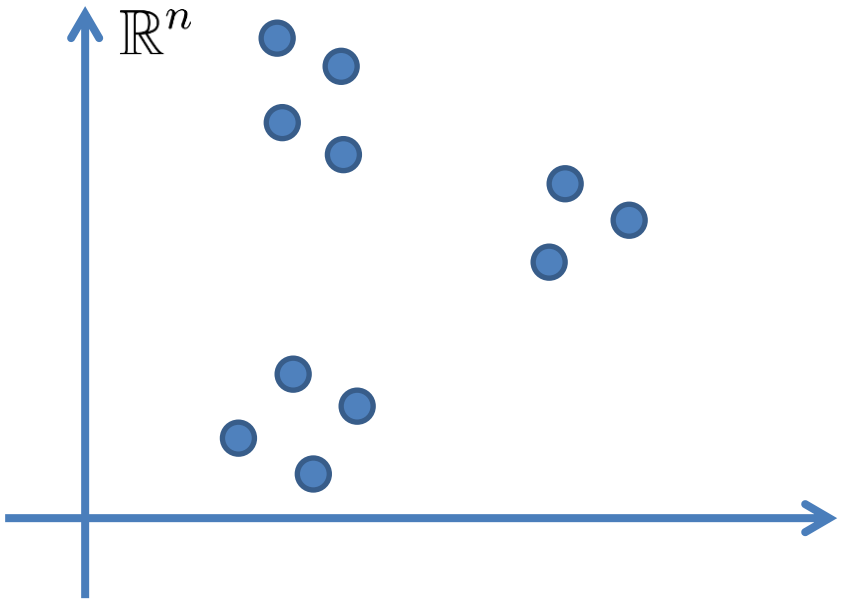
- Overview of Clustering
- K-means clustering
  - Overview of the Method
  - Determining the Number of Clusters
  - Diagnostics
  - Reasons to Choose and Cautions
- Additional Algorithms
  - Density Based Clustering
  - Self-Organize Map (SOM)
  - Hierarchical Clustering

Overview of Clustering

- Supervised vs. Unsupervised Techniques
  - Labelled data vs. Unlabelled data
- Unsupervised Techniques
  - Refers to the problem of finding hidden structure within unlabelled data
  - Clustering, density estimation, dimensionality reduction, etc.
- Clustering is an unsupervised technique

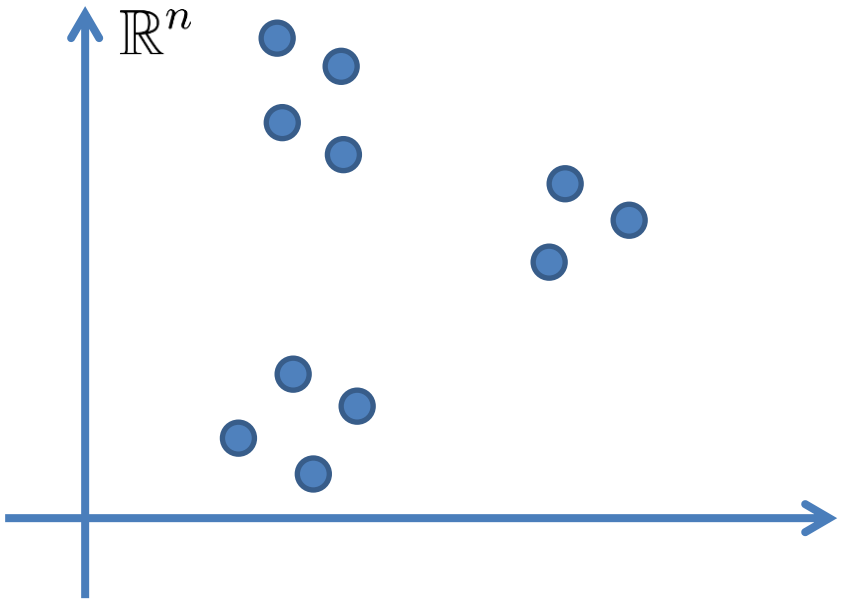
K-means Clustering

- Given a collection of m objects each with n measurable attributes
  - Mathematically, $\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_m \in \mathbb{R}^n$
  - Each object is a point in an n-dimensional space



K-means Clustering

- For a chosen value of k, identify k clusters of objects based on the objects' proximity to the centre of the k groups



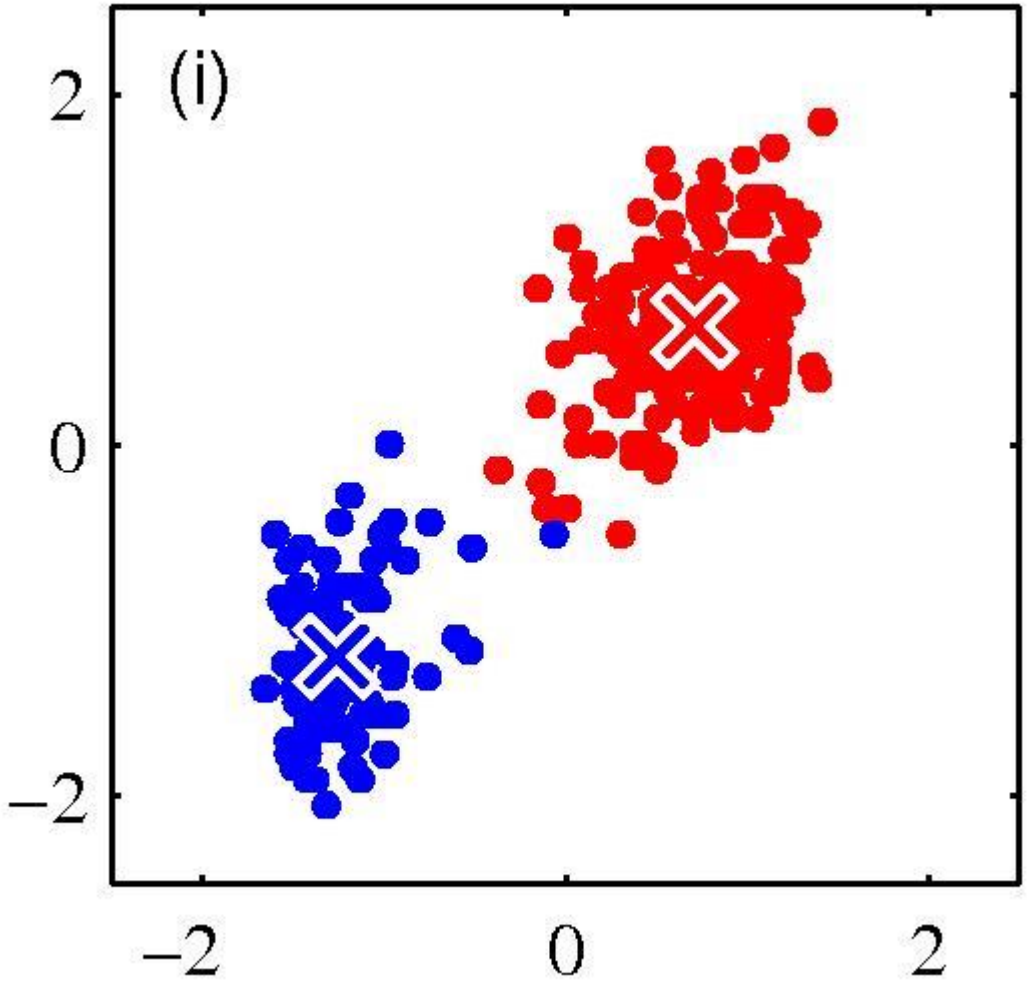
K-means Clustering

- Use Cases
  - Often used as a lead-in to classification
    - Once clusters are identified, labels can be applied to each cluster to do classification
  - Find out whether the data is organized in cohorts, and how the cohorts align with labels (if available)
- Applications
  - Image Processing
  - Medical (Clustering patients)
  - Customer grouping (find similar customers)

Overview of K-means Clustering

- Three steps
  1. Choose a value of k, create k centroids, then initialize them by "guessing" their value.
     - Use k-random selected data points to initialize the centroids. *(pick a suitable random algorithm)*
  2. Compute the distance from each data point to each centroid. Assign each point to the closest centroid.
  3. Update the centroid of each cluster
- Repeat Steps 2 and 3 until convergence, i.e., centroids don't change.

Overview of K-means Clustering



Overview of K-means Clustering

- Compute the Euclidean distance

$$d(\mathbf{x}, \mathbf{y}) = \|\mathbf{x} - \mathbf{y}\|_2 = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2}$$

- Compute the centroid for a cluster (centre of gravity)

$$\bar{\mathbf{x}} = \frac{\sum_{i=1}^{m} \mathbf{x}_i}{m}$$

Determine the Number of Clusters

- What value of k shall be selected?
  - A reasonable guess, some predefined requirement
  - k-1, k, or k+1?
- Within Sum of Squares (WSS)
  - A heuristic
  - Sum of the squares of the distances between each data point and the closest centroid

$$J = \sum_{i=1}^{n} \sum_{j=1}^{k} r_{ij} \|\mathbf{x}_i - \bar{\mathbf{x}}_j\|_2^2; \quad r_{ij} \in \{0, 1\}$$

Determine the Number of Clusters

- An optimization point of view
  - A combinatorial partition problem

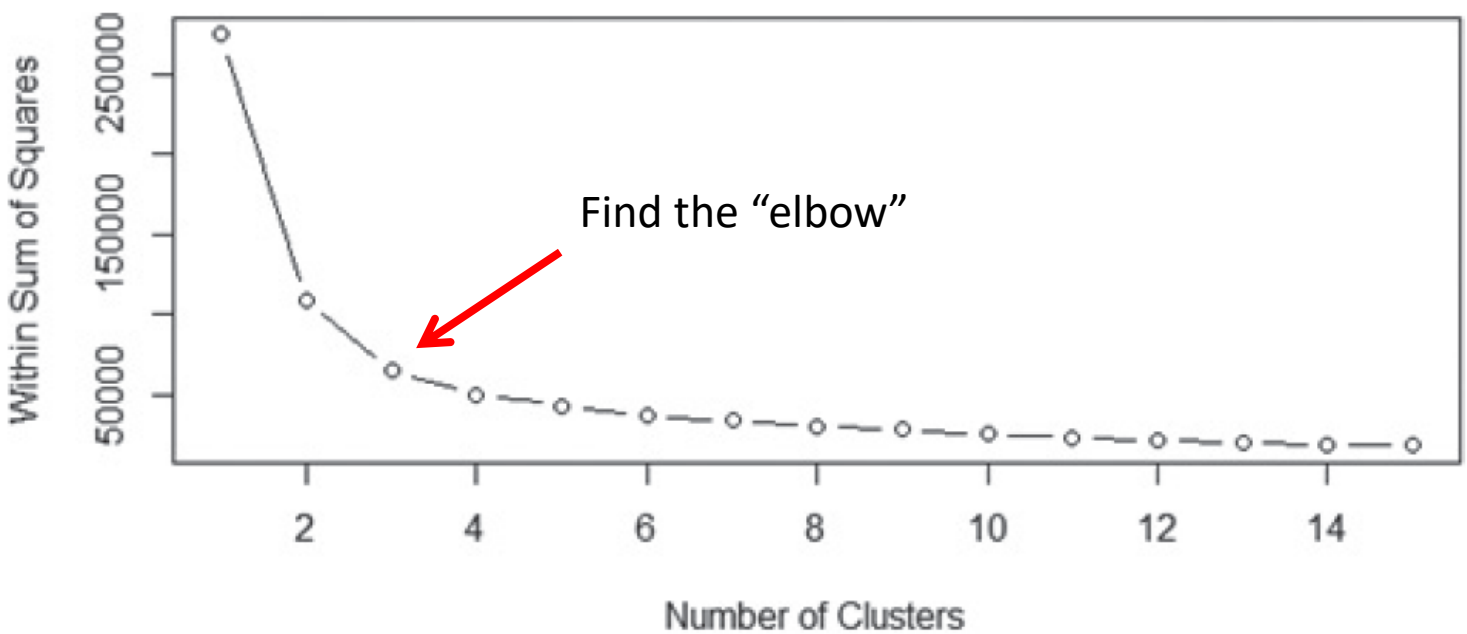
$$J = \sum_{i=1}^{n} \sum_{j=1}^{k} r_{ij} \|\mathbf{x}_i - \bar{\mathbf{x}}_j\|_2^2; \quad r_{ij} \in \{0, 1\}$$

$$\{r_{ij}^*\} = \arg \min_{r_{ij} \in \{0,1\}} J$$

Determine the Number of Clusters

- Within Sum of Squares (WSS)

$$J = \sum_{i=1}^{n} \sum_{j=1}^{k} r_{ij} \|\mathbf{x}_i - \bar{\mathbf{x}}_j\|_2^2; \quad r_{ij} \in \{0, 1\}$$



Using R to Perform K-mean Clustering

- Task is to
  - Group 620 high school seniors based on their grades in “English”, “Math”, and “Science”

```
grade_input = as.data.frame(read.csv('grades_km_input.csv’))
kmdata_orig =
as.matrix(grade_input[,c("Student","English","Math","Science")])
kmdata <- kmdata_orig[,2:4]
```

clustering-demo.R × | kmdata × | kmdata_orig × | grade_input ×

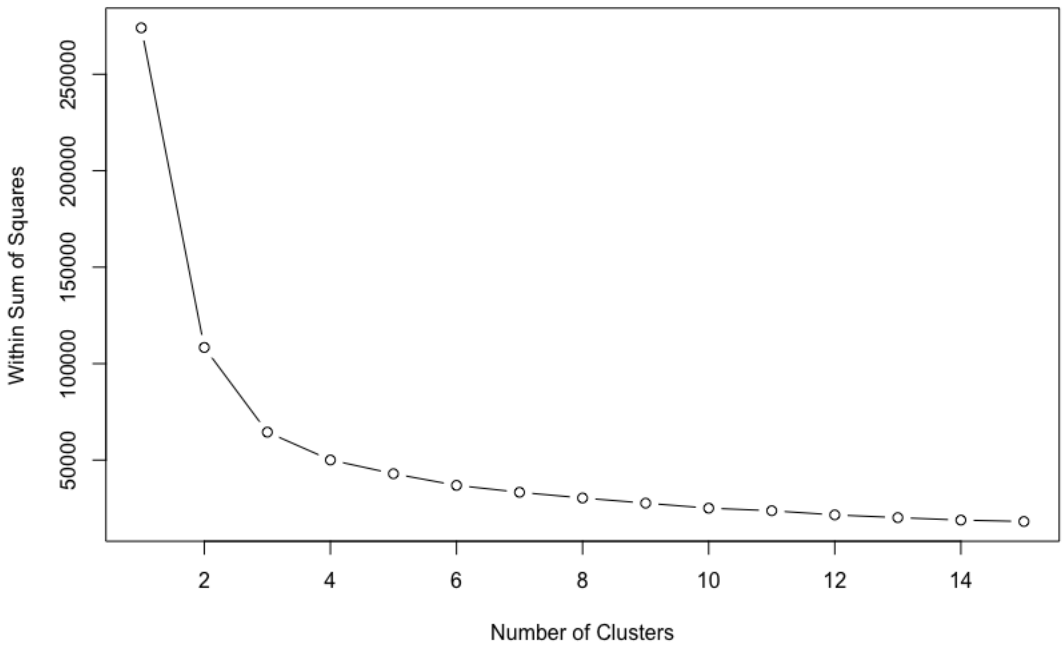
Filter

	English	Math	Science
1	99	96	97
2	99	96	97
3	98	97	97
4	95	100	95
5	95	96	96
6	96	97	96
7	100	96	97

Using R to Perform K-mean Clustering

- Compute and plot WSS to choose k value

```
wss <- numeric(15)
for (k in 1:15) wss[k] <- sum(kmeans(kmdata, centers=k, nstart=25)$withinss)

plot(1:15, wss, type = "b", xlab="Number of Clusters", ylab="Within Sum of
Squares")
```



nstart option attempts multiple initial configurations and reports on the best one. For example, adding nstart=25 will generate 25 initial random centroids and choose the best one for the algorithm

Using R to Perform K-means Clustering

- Perform K-means Clustering

```
km = kmeans(kmdata,3, nstart=25)
km

K-means clustering with 3 clusters of sizes 158, 218, 244

Cluster means:
   English     Math  Science
1 97.21519 93.37342 94.86076
2 73.22018 64.62844 65.84862
3 85.84426 79.68033 81.50820

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 1 1
      1 1 1 1 1 1 1 1 1 1
 [41] 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 1
      1 1 1 1 1 1 1 1 1 1
 [81] 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 1
      1 1 1 1 1 1 1 1 1 1
[121] 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 1
      3 3 3 3 3 3 3 3 3 3
```

Using R to Perform K-means Clustering

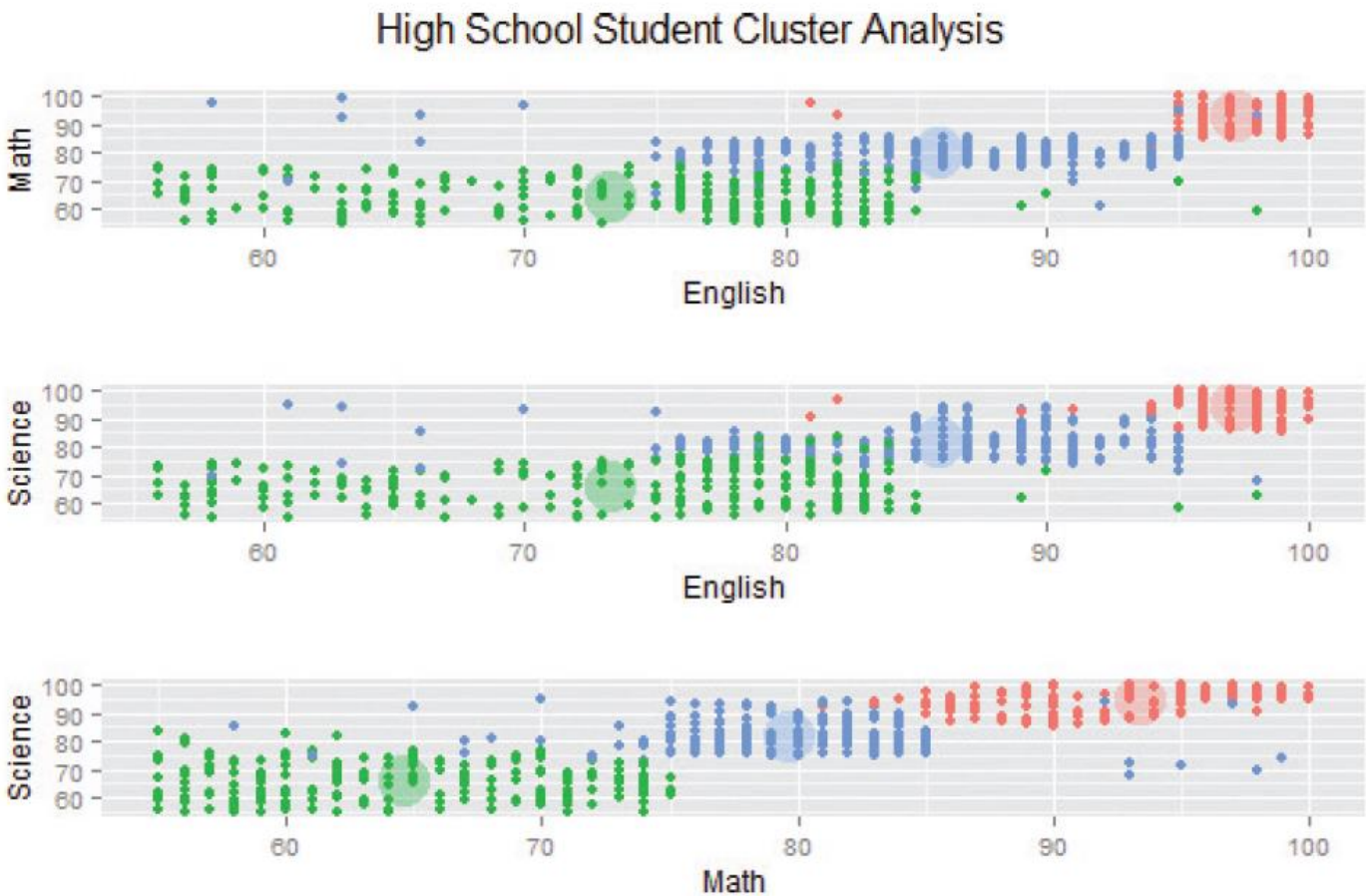
- Perform K-means Clustering

```
[521] 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
      2 2 2 2 2 2 2 2 2 2
[561] 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
      2 2 2 2 2 2 2 2 2 2
[601] 3 3 2 2 3 3 3 3 1 1 3 3 3 2 2 3 2 3 3 3

Within cluster sum of squares by cluster:
[1]  6692.589 34806.339 22984.131
 (between_SS / total_SS =  76.5 %)

Available components:

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

```
c( wss[3] , sum(km$withinss) )

[1] 64483.06 64483.06
```

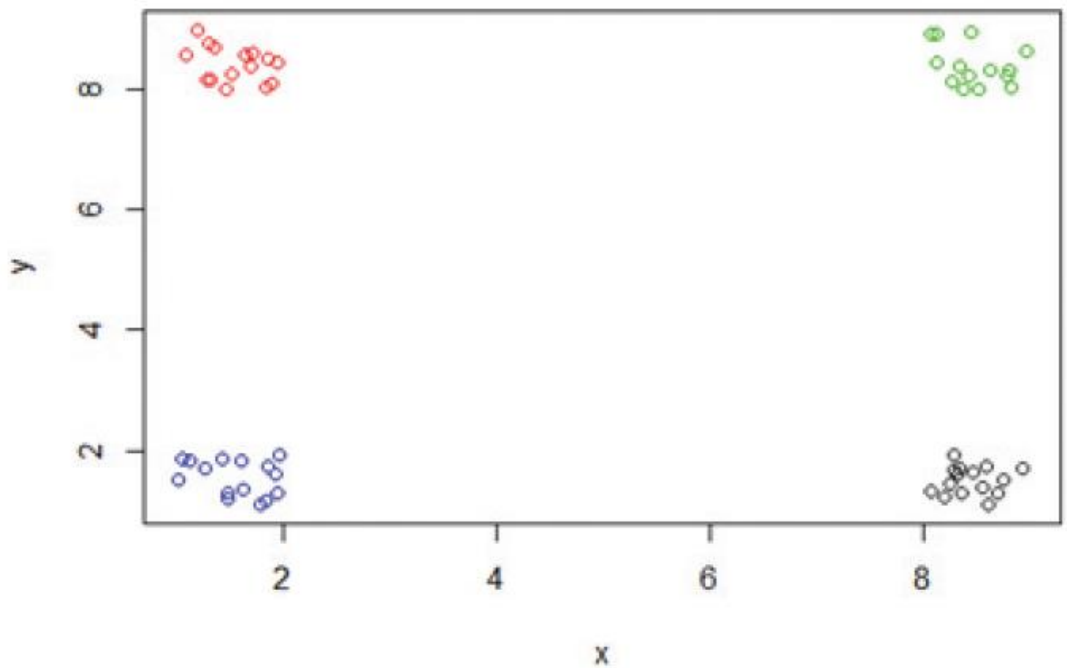
Using R to Perform K-means Clustering

- Visualize the identified clusters and centroids



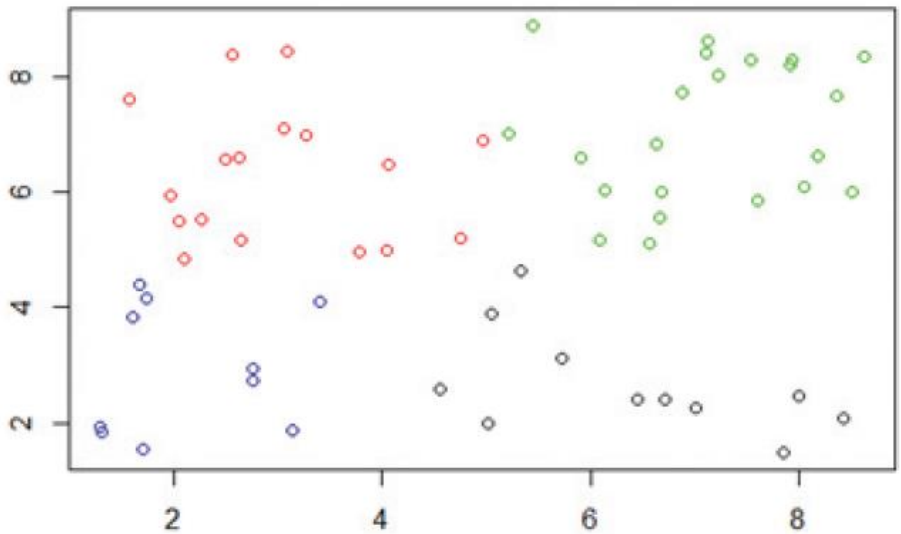
Diagnostics – K-means (clustering)

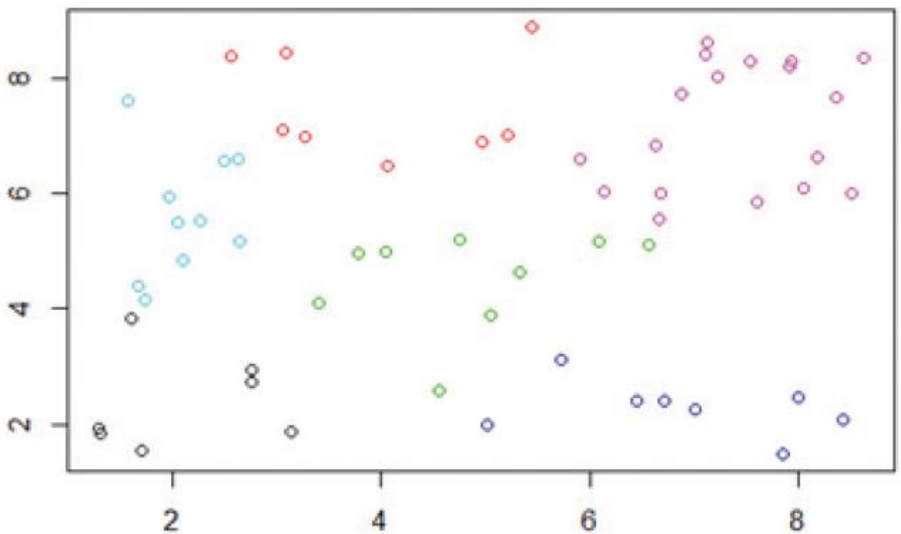
- The following questions shall be asked
  - Are the clusters well separated from each other?
  - Do any of the clusters have only a few points?
  - Do any of the centroids appear to be too close to each other?



Diagnostics

- A principle
  - If using more clusters does not better distinguish the groups, it is almost certainly better to go with fewer clusters





Reasons to Choose and Cautions

- Several decisions that must be made
  - What object attributions shall be included in clustering analysis?
  - What unit of measure shall be used for each attribute?
  - Do the attributes need to be rescaled?
    - One attribute could have a disproportionate effect

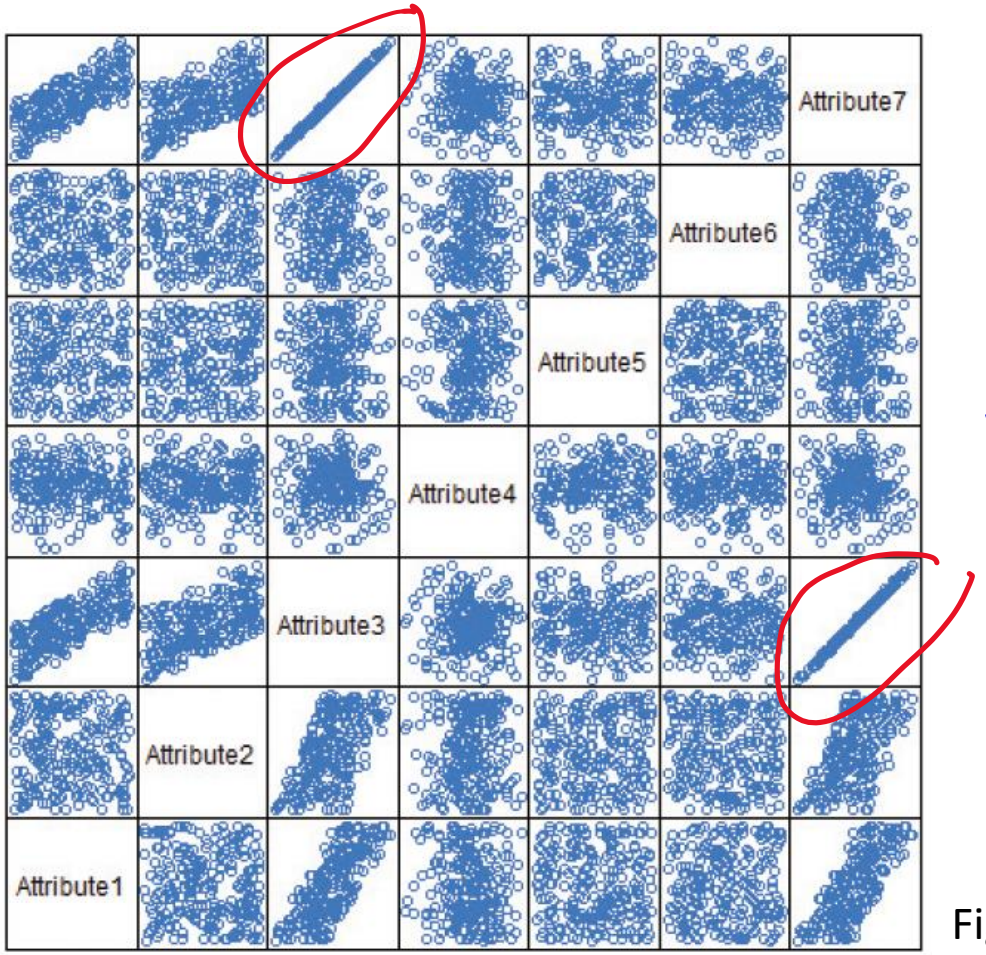
May lead back to Phase 2 data preparation!

Reasons to Choose and Cautions

- Object attributes
  - Whether it will be known for a new object?
  - Best to reduce the number of attributes to the extent of possible
    - Avoid using too many variables (Why?)
    - Avoid using several similar variables (Why?)
- Identify any highly correlated attributes
- Feature selection, PCA, etc.

Reasons to Choose and Cautions

- Identify any highly correlated attributes



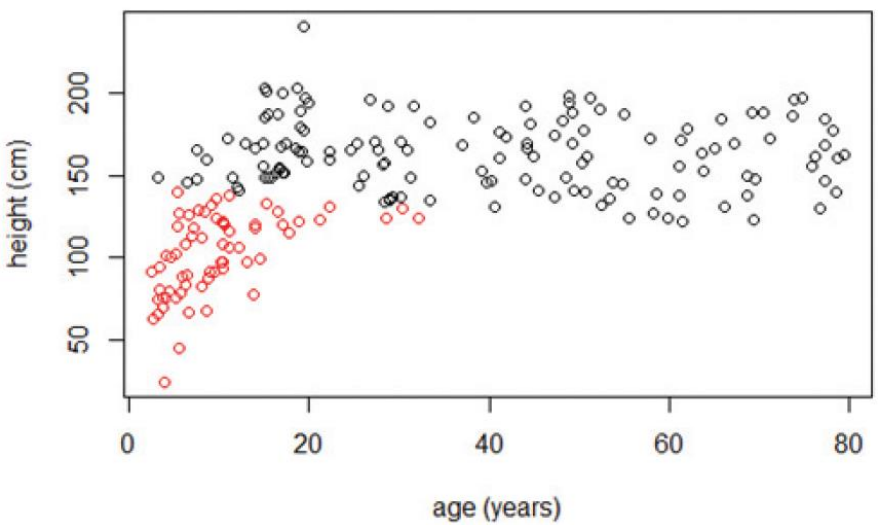
R-Squared or **Pearson's r**

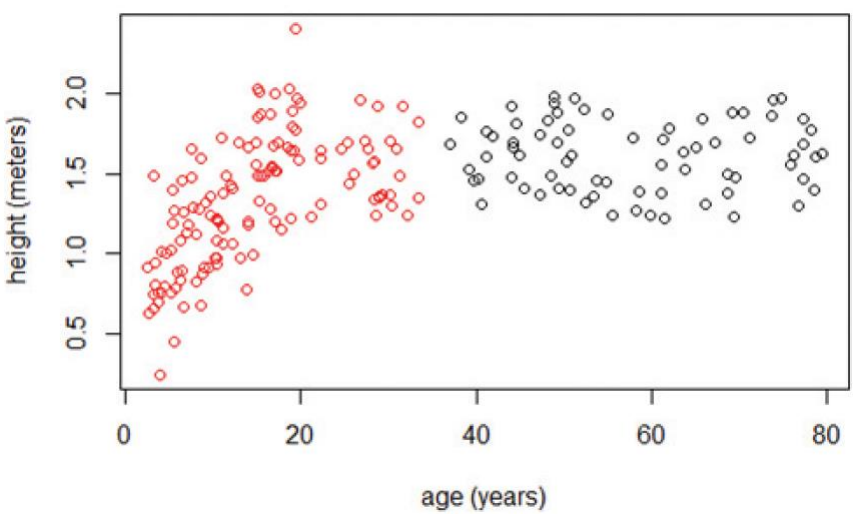
What is your observation?

Fig. Scatterplot matrix for 7 attributes

Reasons to Choose and Cautions

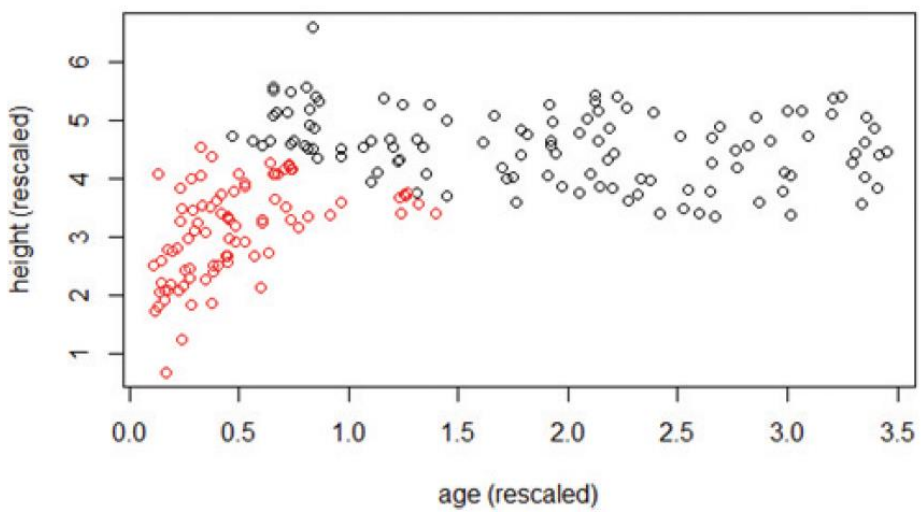
- Units of measure could affect clustering result

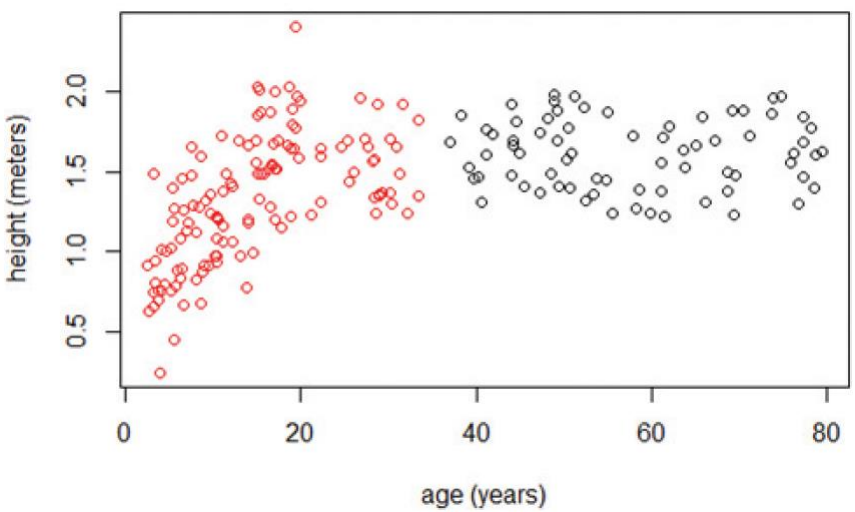




Reasons to Choose and Cautions

- Rescaling attributes affect clustering result
  - Divide each attribute by its standard deviation
  - Normalisation: mean=0, sdev=1, particularly when Euclidean distance is used





Additional Algorithms

- K-means clustering is easily applied to numeric data where the concept of distance can naturally be applied
- K-modes handles categorical data
  - Use the number of differences in the respective components of the attributes
    - What is the distance between (a,b,e,d) and (d,d,d,d)?
  - Implemented by the `kmode()` function

- Caution: Sometimes it is better to convert categorial (or symbolic) data to numerical i.e. {hot, warm, cold} to {1,0,-1}, or use one-hot encoding.
  - Understand why!
  - Understand how to encode categorial values.

Additional Considerations

- Despite its popularity, K-means has problems:
  - When data contains noise and/or outliers
  - When clusters have non-globular shapes
    - Cluster shape affect the choice of the distance metric
  - When clusters vary in densities
  - When clusters differ significantly in size
  - Can reveal "empty" clusters
  - Sensitive to the starting positions of the initial centroids
    - Sunning multiple times with different initialization and choose the one with lowest WSS

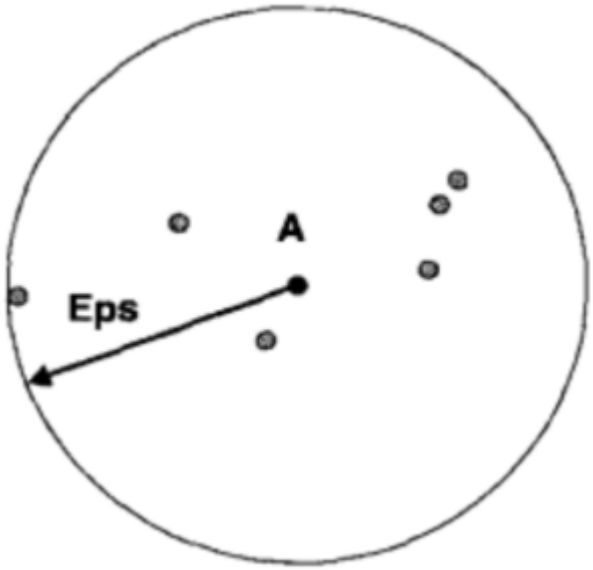
- Know your data (i.e via visualization) to verify whether K-means is suitable.

Density Based Clustering

- Density-based clustering locates regions of high density that are separated from one another by regions of low density.
- In other words, clusters are dense regions in the data space, separated by regions of lower object density
- Major features of density-based clustering:
  - Discover clusters of arbitrary shape
  - Handle and identify noise
  - Need density parameters as termination condition

DBScan

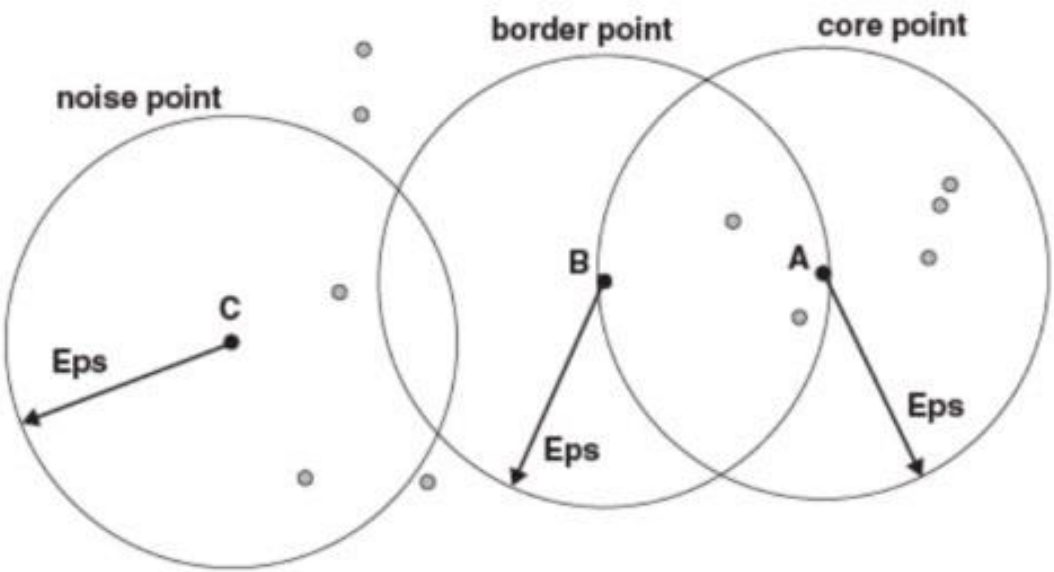
- Density is estimated for a particular point in the data set by counting the number of points within a specified radius, Eps, of that point. This includes the point itself.



- Example: the number of points within a radius of Eps of point A is 7, including A itself.
  - The density of A is 7.

DBScan

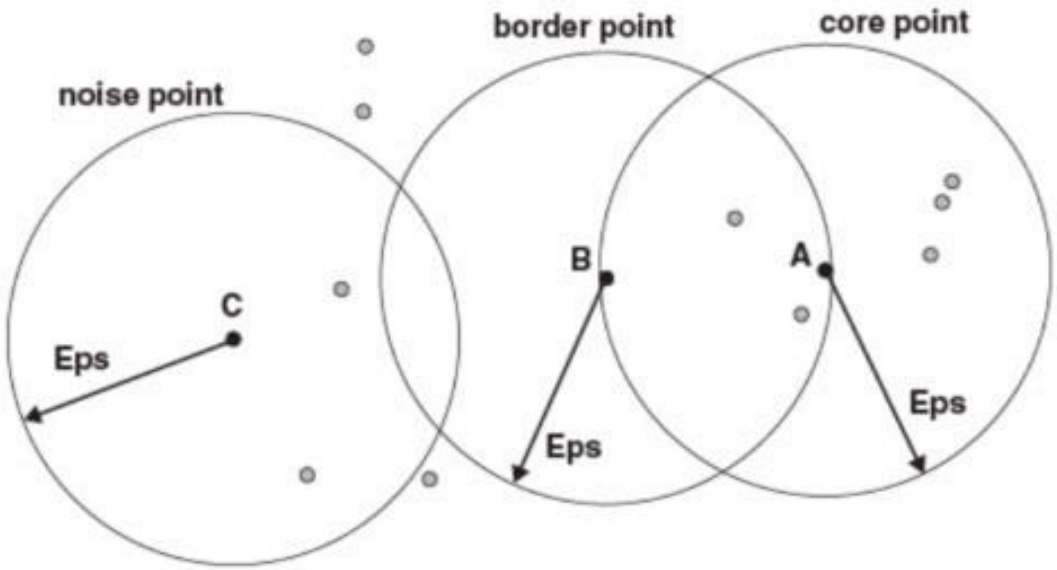
- Given a density threshold ($MinPts$) and a radius (Eps), the points in a dataset are classified into three types: core point, border point, and noise point.
  - Core points: $Point\ whose\ density\ >=\ MinPts$
  - Core points are in the interior of a density-based cluster.



Example: If $MinPts\ =\ 6$ then A is a core point because its density = 7 (7>6)

DBScan

- Three types: core point, border point, and noise point.
  - A border point is not a core point but falls within the neighborhood of a core point.

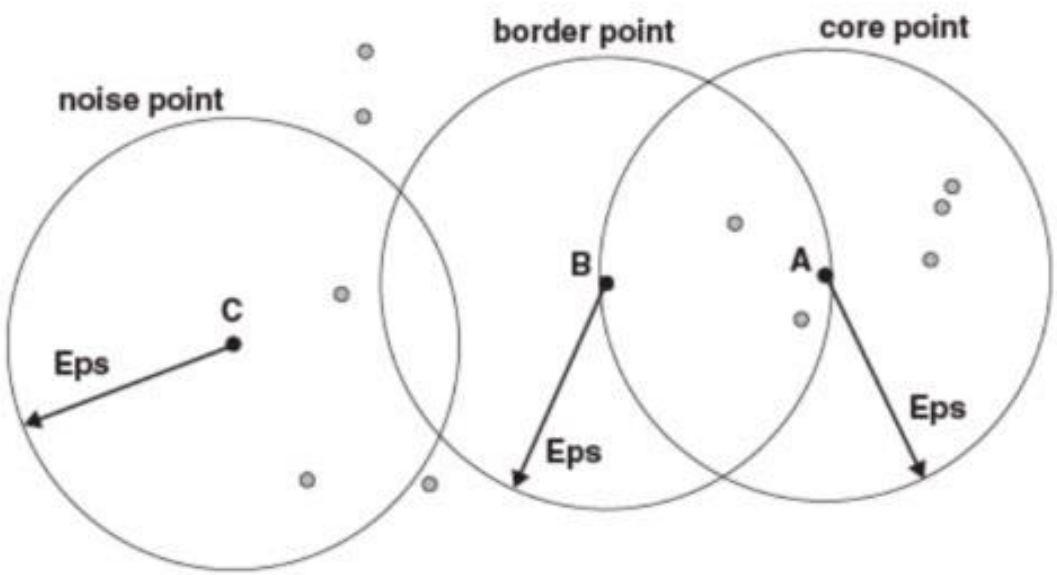


Example:

- The density of B is 4 and less than $MinPts = 6$, so B is not a core point.
- But B falls within the neighbor of A (a core point).
- So, B is a border point.

DBScan

- Three types: core point, border point, and noise point.
  - A noise point is any point that is neither a core point nor a border point.



Example:

- The density of C is 3 which is less than $MinPts = 6$, so C is not a core point.
- C doesn't fall within the neighborhood of any core point, so it is not a border point.
- So, C is a noise point..

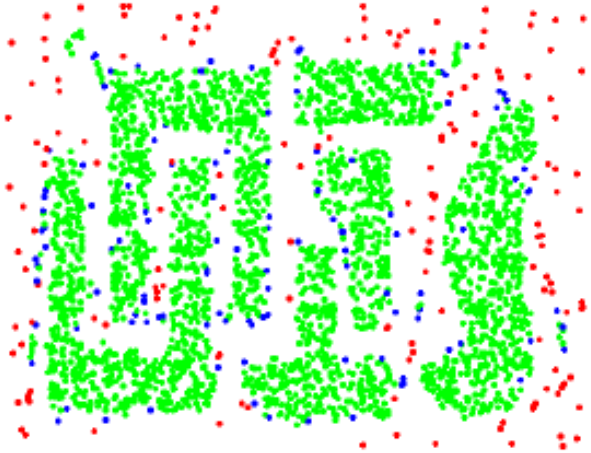
DBScan

- Steps of DBSCAN to identify clusters
  - Step 1: Label each point as either core, border, or noise point.
  - Step 2: Mark each group of Eps connected core points as a separate cluster.
  - Step 3: Assign each border point to one of the clusters of its associate core points.

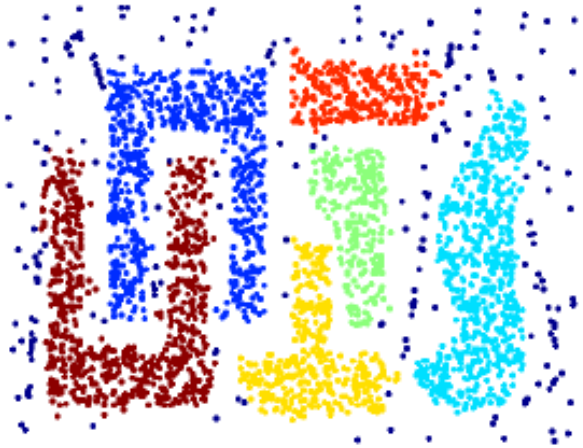
DBScan Example

Original Points

Eps = 10, MinPts = 4



Mark core, border and noise points



Mark connected core points

DBScan Properties

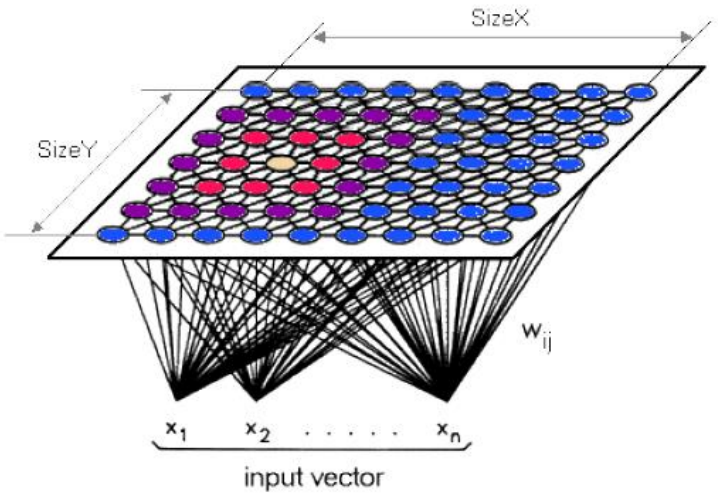
- DBSCAN:
  - Resistant to noise and outliers
  - Can handle clusters of different shapes and sizes
  - Computational complexity is similar to K-means
- When DBSCAN does not work well
  - Varying densities
    - Can be overcome by using sampling
  - Sparse and high-dimensional data
    - Can be overcome by using topology preserving dimension reduction techniques.

Self-Organizing Maps

- Self organizing maps are a type of Neural Network (NN).
- Unsupervised algorithm.
- Project high dimensional data onto a n-dimensional display space (the feature map).
  - Commonly n=2
- Topology preserving mappings & clustering.
  - Data that is “similar” within the input space remain “close” to each other in the display space

Self-Organizing Maps

- Self-organizing maps have two layers:
  - An input layer and
  - An output layer called the feature map.
- The feature map consists of neurons.
  - organized on a regular grid.
  - Unlike other ANN types, the neuro in a SOM don't have an activation function.



- Each neuron in a SOM is assigned a weight vector with the same dimensionality as the input space.

Self-Organizing Maps

- The weights in a SOM are trained in a two-step algorithm:
  - Step 1: Competitive step
    - Every neuron is examined to calculate which one's weights is most similar to the input vector. The winning neuron is known as the Best Matching Unit (BMU).
  - Step 2: Cooperative step
    - The weights of the BMU and the weights of the neighboring neurons is updated.

Self-Organizing Maps

- Training Algorithm:
  1. Each neuron's weights is initialized with random values.
  2. A sample is chosen at random from the set of training data.
  3. Find the BMU.
  4. Identify the neighbourhood of the BMU. The size of the neighborhood decreases over time.
  5. Update the weights of the BMU and all of its neighbors so that they become more similar to the sample vector. The closer a node is to the BMU, the more its weights get altered and the farther away the neighbor is from the BMU, the less it is updated.
- Step 2 through to 5 are repeated N times.
  - Normally N is a multiple of the number of training samples.

Self-Organizing Maps in Python

```
#Step 1: Prepare the data
import pandas as pd
import numpyas np
from sklearn.utilsimport shuffle

df = pd.read_csv("A1_BC_SEER_data.csv", )
df = shuffle(df)
df = df[:int(df.shape[0]*0.2)] #use 20% subset for
                               #demonstration purposes
target = df['Survival months'] #Extract the target column

#Binarize target
target = np.where(df['Survival months'] < 60, 0, target)
target = np.where(df['Survival months'] >= 60, 1, target)
```

Self-Organizing Maps in Python

```
#Step 2: Preprocessthe data
from sklearn.model_selectionimport train_test_split

myseed=7 #Seed for the random number generator

#Remove irrelevant features, and targets from df
dropList= ['Patient ID', 'Survival months']
for item in dropList:
df.drop(item, axis=1, inplace=True)

#Scale the data?
#from sklearnimport preprocessing
#scaling = preprocessing.MinMaxScaler()
#data = scaling.fit_transform(data)

#Create a train, test, and validation set
X, X_tst, Y, Y_tst= train_test_split(df, target, test_size=.333,
random_state=myseed)
X_trn, X_val, Y_trn, Y_val= train_test_split(X, Y, test_size=.5,
random_state=myseed)

X_trn= X_trn.to_numpy()
X_tst= X_tst.to_numpy()
X_val= X_val.to_numpy()
```

Self-Organizing Maps in Python

```
#Step 3: Train the SOM
from myminisom import MiniSom #see Moodle site for myminisom

#Create the SOM
som_shape = (100, 100) #define the size of the som
som = MiniSom(som_shape[0], som_shape[1], X_trn.shape[1],
sigma=som_shape[0]/2, learning_rate=.9,
neighborhood_function='gaussian', random_seed=myseed)

#initialize the SOM, then train it
epochs=40
som.pca_weights_init(X_trn)
som.train_random(X_trn, epochs * len(X_trn), verbose=True)

#Find the BMU for each sample
BMU_trn = np.array([som.winner(x) for x in X_trn])
BMU_class0 = BMU_trn[Y_trn==0]
BMU_class1 = BMU_trn[Y_trn==1]
```

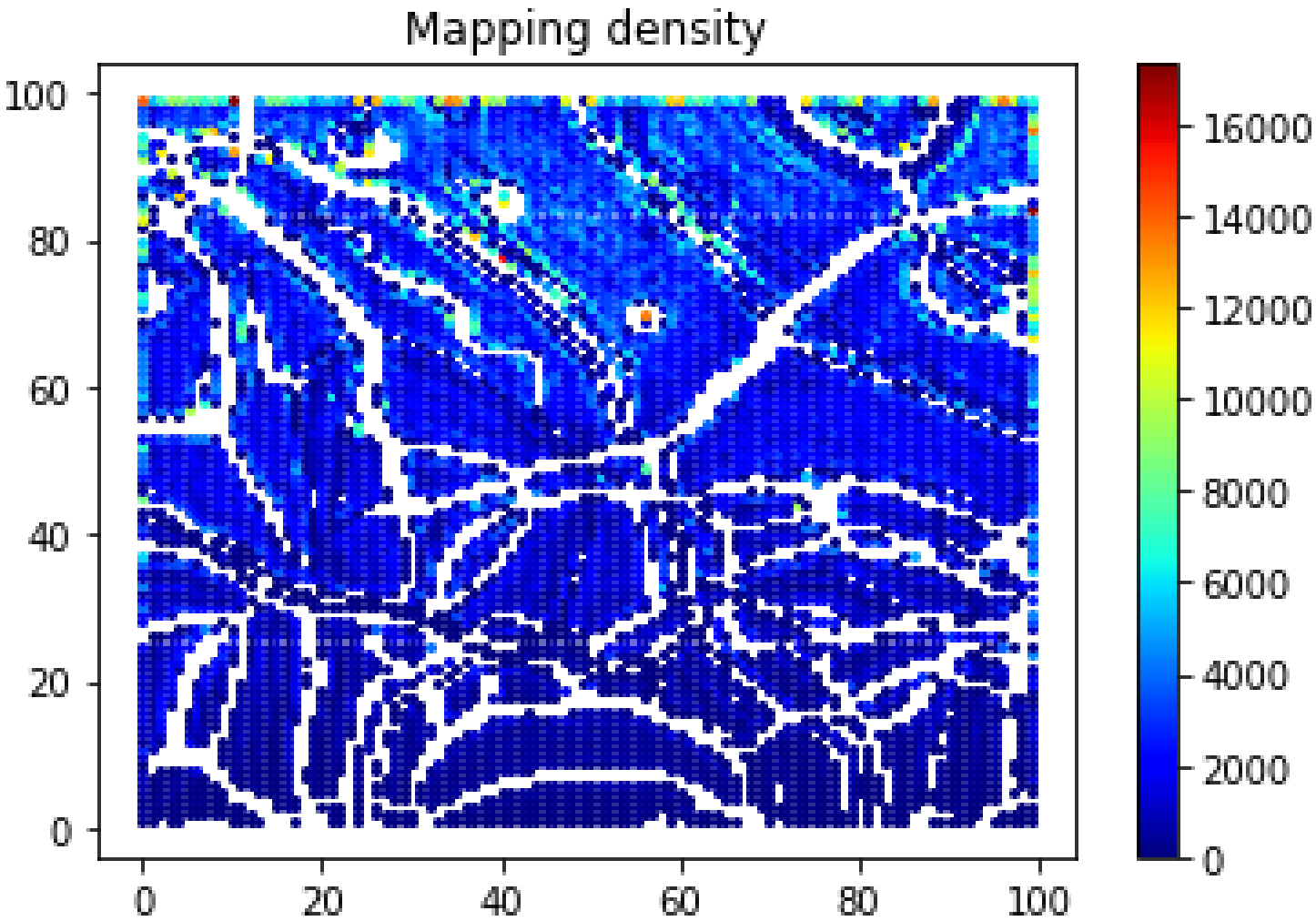
Self-Organizing Maps in Python

```
#Step 4: Plot some results (density map of all samples)
import matplotlib.pyplot as plt
from copy import copy

densitymap = np.zeros(som_shape)
for row in range(0,BMU_trn.shape[0]):
      x,y = BMU_trn[row]
      densitymap[y,x] += 1

densitymap[densitymap==0]=np.nan #mark zero values with nan
my_cmap = copy(plt.cm.jet)
my_cmap.set_bad(color=(1,1,1)) #plot nan in white color
plt.imshow(densitymap, cmap=my_cmap, interpolation="none",
origin="lower", aspect=0.75)
plt.colorbar()
plt.title('Mapping density')
plt.show()
```

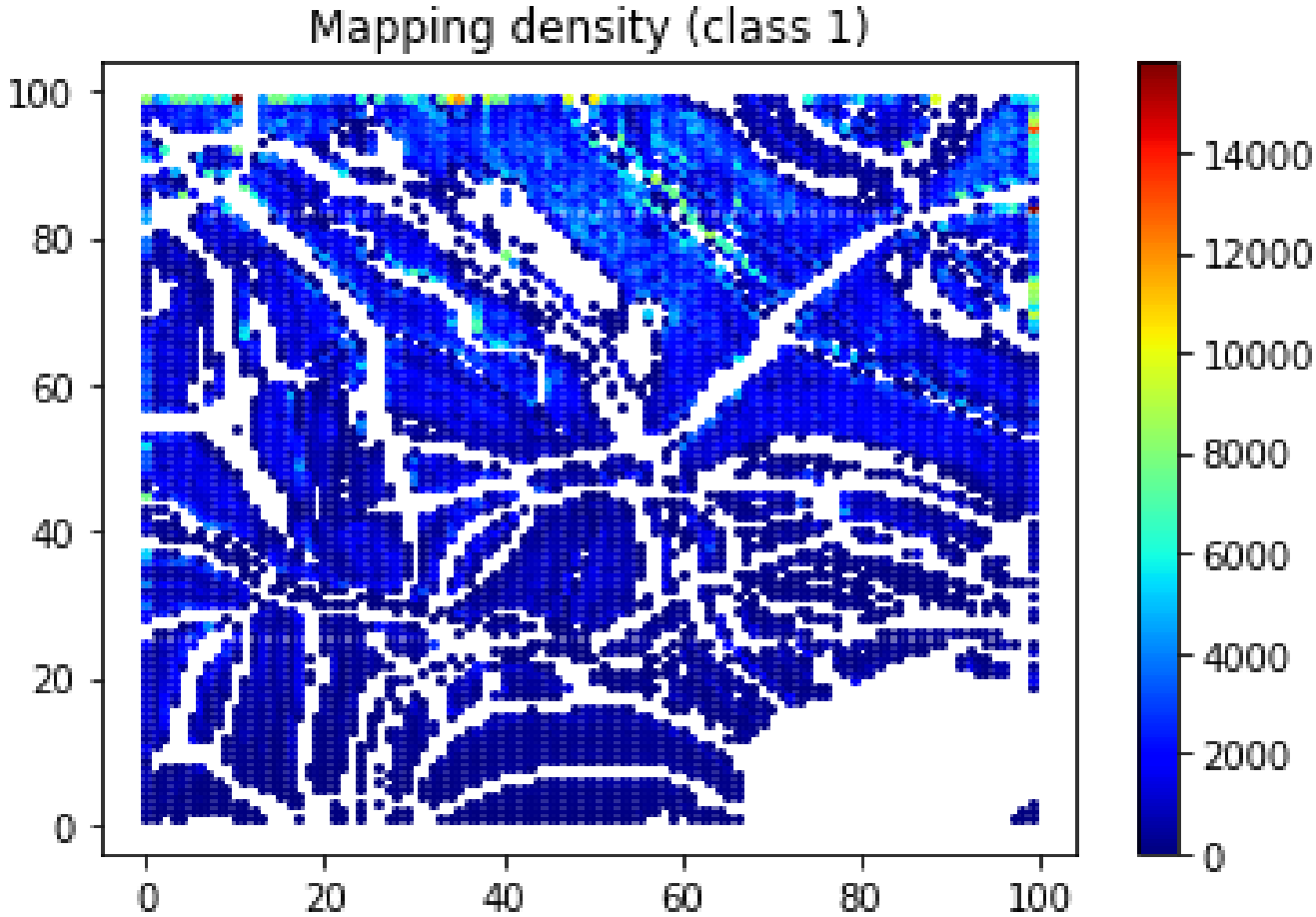
Self-Organizing Maps in Python

```
#density map of all samples from class 1
import matplotlib.pyplot as plt
densitymap = np.zeros(som_shape)
for row in range(0,BMU_class1.shape[0]):
    x,y = BMU_class1[row]
    densitymap[y,x] += 1

densitymap[densitymap==0]=np.nan #mask zero values
plt.imshow(densitymap, cmap=my_cmap, interpolation="none", origin="lower", aspect=0.75)
plt.colorbar()
plt.title('Mapping density (class 1)')
plt.show()
```

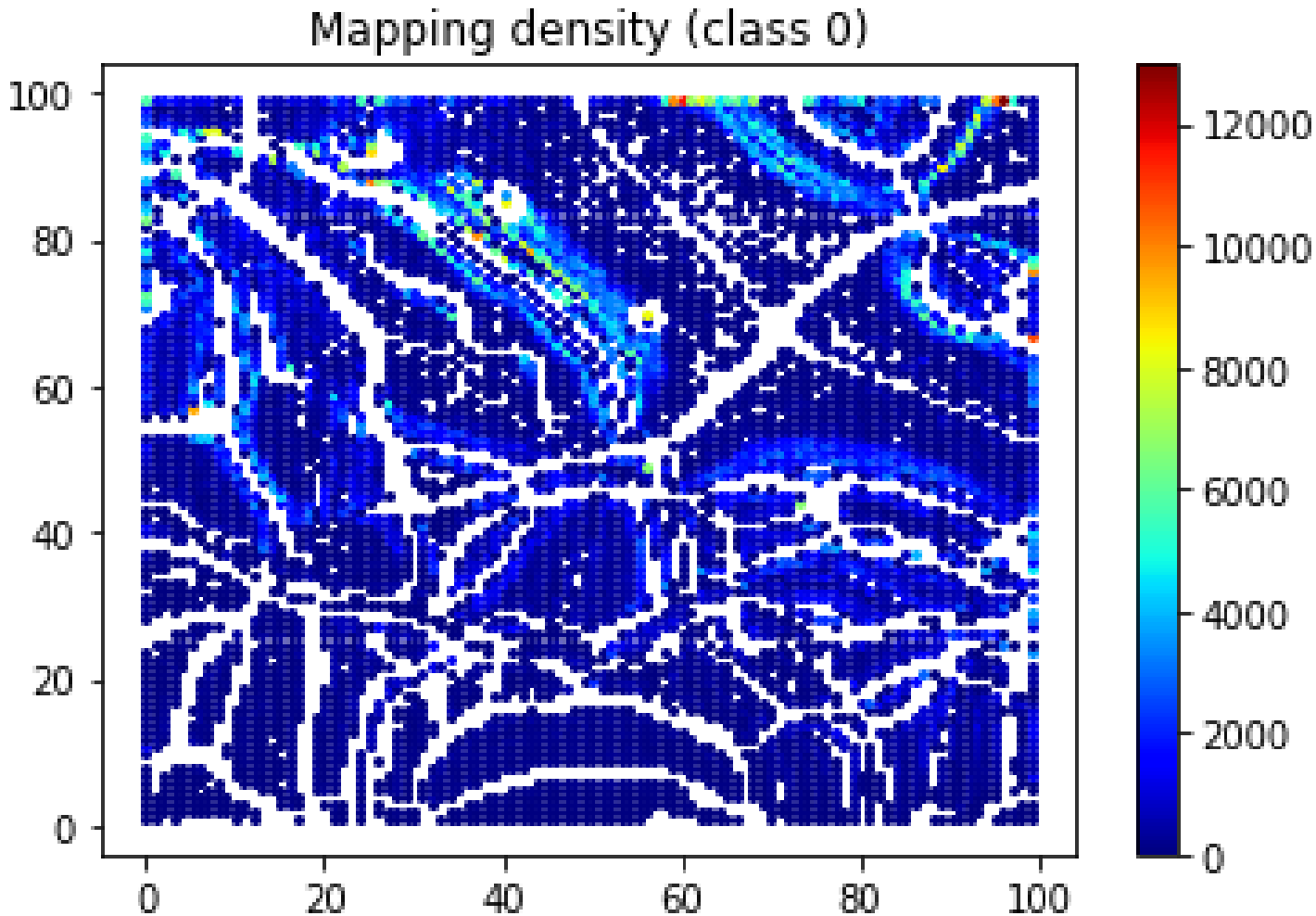
Self-Organizing Maps – An Example



Self-Organizing Maps – An Example



Self-Organizing Maps – An Example



Self-Organizing Maps - Note

- Each neuron clusters samples that are mapped to it.
  - $n\ x\ m$ clusters (size of the SOM)
- A group of neurons form larger cluster
  - Cluster analysis needed to detect these.

Self-Organizing Maps in Python

```
#Compute the quantization error
qerr = som.quantization_error(X_trn)
qerr
7.454546962215053
#inspect some weights

som.get_weights()[1,1]
array([1.45920861e+00, 2.94414892e+00, 1.58042285e+00, 1.98123243e+00,
       5.60639639e+01, 1.49714974e+02, 2.37928820e-01, 2.06261914e+02,
       5.05750682e+02, 1.57189567e+00, 1.03022154e+00, 8.98461177e+00,
       1.20844111e+00, 1.33963081e+00, 5.28842982e+01, 4.74582306e+01])
```

Self-Organizing Maps

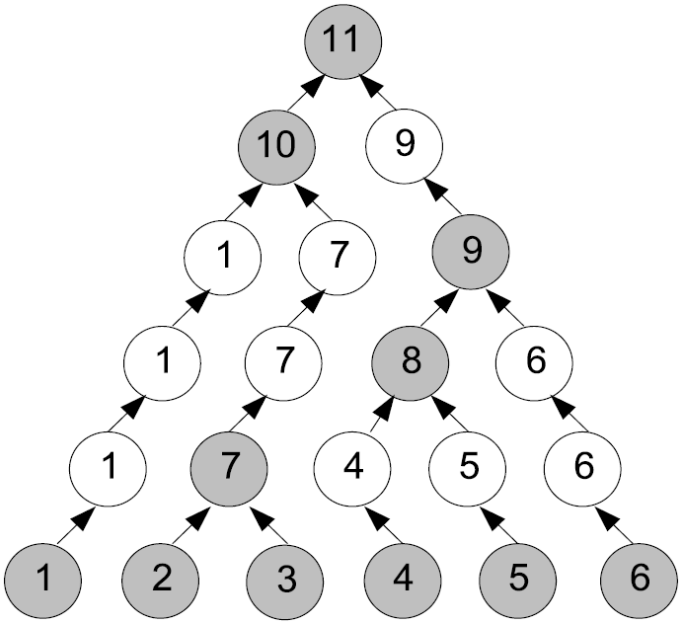
- SOMs are an excellent choice for data visualization
- Many visualization techniques
  - From exploratory data analytics
  - Dimension reduction techniques
    - i.e. PCA, t-SNE, SOM,…
- Why use Self-Organizing Maps (SOMs) in BDA?
  - Topology preservation (unlike PCA)
  - Able to deal with new data & missing values (unlike t-SNE)
  - Can reduces the amount of information that needs to be evaluated
  - Produces prototypes that represent the full set of attributes with their original meaning (unlike PCA)
  - …

Self-Organizing Maps

- When not to use SOMs in BDA:
  - When the data is very sparse
  - When cardinality (limited resolution) of the map is a problem.
  - When multi-core compute infrastructure is unavailable.

Hierarchical Clustering

- Hierarchical Clustering (`hclust()`)
  - Hierarchical agglomerative clustering
  - Hierarchical divisive clustering



1. Each object is initially treated as a cluster
2. The clusters are then combined with the most similar cluster in each step
3. This process is repeated until one cluster (containing all objects) exists

Computationally very expensive $O(n^2)$ to $O(n^3)$ and thus rarely used in Big Data analytics.

Q&A

Images Courtesy of Google Image