IN4151 – Information Engineering

Unit 3: Introduction to Data and Information Science for Management (part IV)

Ángel Jiménez, angeljim@uchile.cl

Universidad de Chile
Facultad de Ciencias Físicas y Matemáticas
Departamento de Ingeniería Industrial

25 de abril de 2022

Agenda

- 3.5 Supervised Modeling II.
  - Bayesian Classifiers (Naïve Bayes).
  - Support Vector Machines.
  - Ensemble Methods.

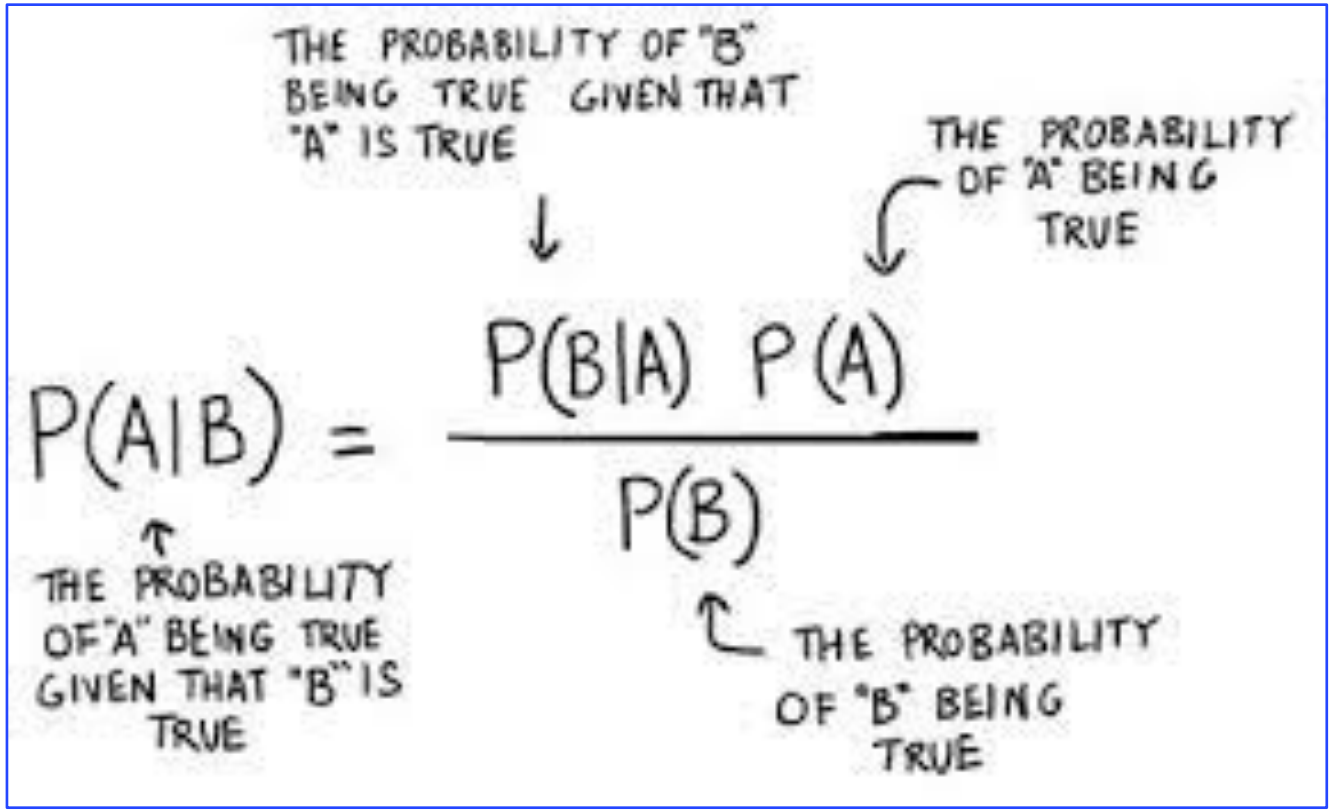
- 3.6 Unsupervised Modeling.
  - Definitions.
  - K-means
  - Hierarchical model

3.5 Supervised Modeling II

Disclaimer

- This presentation has slides adapted from Tan et al., 2019, while some were made by Ángel Jiménez.
  - Tan, P., Steinbach, M., Kumar, V. (2019). Introduction to Data Mining, Pearson Higher Education.

Bayesian Classifiers



Bayes Classifier

- A probabilistic framework for solving classification problems
- Conditional Probability:

$$P(Y \mid X) = \frac{P(X,Y)}{P(X)}$$

$$P(X \mid Y) = \frac{P(X,Y)}{P(Y)}$$

- Bayes theorem:

$$P(Y \mid X) = \frac{P(X \mid Y)P(Y)}{P(X)}$$

Using Bayes Theorem for Classification

- Consider each attribute and class label as random variables
- Given a record with attributes $(X_1, X_2, \ldots, X_d)$, the goal is to predict class Y
  - Specifically, we want to find the value of Y that maximizes $P(Y \mid X_1, X_2, \ldots, X_d)$
- Can we estimate $P(Y \mid X_1, X_2, \ldots, X_d)$ directly from data?

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Source: Tan et al., 2019

Using Bayes Theorem for Classification

- Approach:
  - Compute posterior probability $P(Y \mid X_1, X_2, \ldots, X_d)$ using the Bayes theorem

$$P(Y \mid X_1 X_2 \ldots X_n) = \frac{P(X_1 X_2 \ldots X_d \mid Y) P(Y)}{P(X_1 X_2 \ldots X_d)}$$

  - *Maximum a-posteriori*: Choose Y that maximizes $P(Y \mid X_1, X_2, \ldots, X_d)$
  - Equivalent to choosing value of Y that maximizes $P(X_1, X_2, \ldots, X_d \mid Y)\, P(Y)$

- How to estimate $P(X_1, X_2, \ldots, X_d \mid Y)$?

Source: Tan et al., 2019

Example Data

Given a Test Record: $X = (\text{Refund} = \text{No}, \text{Divorced}, \text{Income} = 120\text{K})$

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

- We need to estimate
  - P(Evade = Yes | X) and P(Evade = No | X)
  - In the following we will replace
    Evade = Yes by Yes, and
    Evade = No by No

Source: Tan et al., 2019

Example Data

Given a Test Record: $X = (\text{Refund} = \text{No}, \text{Divorced}, \text{Income} = 120\text{K})$

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Using Bayes Theorem:

- $P(\text{Yes} \mid X) = \dfrac{P(X \mid \text{Yes})P(\text{Yes})}{P(X)}$
- $P(\text{No} \mid X) = \dfrac{P(X \mid \text{No})P(\text{No})}{P(X)}$
- How to estimate P(X | Yes) and P(X | No)?

Source: Tan et al., 2019

Conditional Independence

- **X** and **Y** are conditionally independent given **Z** if $P(\mathbf{X}|\mathbf{YZ}) = P(\mathbf{X}|\mathbf{Z})$

- Example: Arm length and reading skills
  - Young child has shorter arm length and limited reading skills, compared to adults.
  - If age is fixed, no apparent relationship between arm length and reading skills.
  - Arm length and reading skills are conditionally independent given age.

Source: Tan et al., 2019

Naïve Bayes Classifier

- Assume independence among attributes X_i when class is given:
  - $P(X_1, X_2, \ldots, X_d \mid Y_j) = P(X_1 \mid Y_j)\, P(X_2 \mid Y_j) \ldots P(X_d \mid Y_j)$
  - Now we can estimate $P(X_i \mid Y_j)$ for all X_i and Y_j combinations from the training data
  - New point is classified to Y_j if $P(Y_j) \prod P(X_i \mid Y_j)$ is maximal.

Source: Tan et al., 2019

Naïve Bayes on Example Data

Given a Test Record: $X = (\text{Refund} = \text{No}, \text{Divorced}, \text{Income} = 120\text{K})$

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

P(X | Yes) =

P(Refund = No | Yes) x

P(Divorced | Yes) x

P(Income = 120K | Yes)

P(X | No) =

P(Refund = No | No) x

P(Divorced | No) x

P(Income = 120K | No)

Source: Tan et al., 2019

Estimate Probabilities from Data

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

- P(y) = fraction of instances of class y, e.g.:
  - P(No) = 7/10
  - P(Yes) = 3/10
- For categorical attributes, $P(X_i = c \mid y) = n_c / n$
  - Where $|X_i = c|$ is number of instances having attribute value $X_i = c$ and belonging to class y.
  - Examples:
    - P(Status=Married | No) = 4/7
    - P(Refund=Yes | Yes)=0

Source: Tan et al., 2019

Estimate Probabilities from Data

- For continuous attributes:
  - Use discretization, i.e., partition of the range into bins:
    - Replace continuous value with bin value (attribute changed from continuous to ordinal)
  - Probability density estimation:
    - Assume attribute follows a normal distribution.
    - Use data to estimate parameters of distribution, e.g., mean and standard deviation.
    - Once probability distribution is known, use it to estimate the conditional probability P(Xi|Y).

Source: Tan et al., 2019

Estimate Probabilities from Data

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

- Normal distribution:

$$P(X_i \mid Y_j) = \frac{1}{\sqrt{2\pi\sigma_{ij}^2}} e^{-\frac{(X_i-\mu_{ij})^2}{2\sigma_{ij}^2}}$$

- One for each (Xi,Yi) pair.
- For (Income, Class=No):
  - If Class=No
    - Sample mean = 110
    - Sample variance = 2975

$$P(Income = 120 \mid No) = \frac{1}{\sqrt{2\pi}(54.54)} e^{-\frac{(120-110)^2}{2(2975)}} = 0.0072$$

Source: Tan et al., 2019

Numeric Example of Naïve Bayes Classifier

Given a Test Record: $X = (\text{Refund} = \text{No}, \text{Divorced}, \text{Income} = 120\text{K})$

- **Naïve Bayes Classifier:**
  - P(Refund = Yes | No) = 3/7
  - P(Refund = No | No) = 4/7
  - P(Refund = Yes | Yes) = 0
  - P(Refund = No | Yes) = 1
  - P(Marital Status = Single | No) = 2/7
  - P(Marital Status = Divorced | No) = 1/7
  - P(Marital Status = Married | No) = 4/7
  - P(Marital Status = Single | Yes) = 2/3
  - P(Marital Status = Divorced | Yes) = 1/3
  - P(Marital Status = Married | Yes) = 0

- **For Taxable Income:**
  - If class = No, sample mean = 110, sample variance = 2975
  - If class = Yes, sample mean = 90, sample variance = 25

- P(X | No) = P(Refund=No | No)
  × P(Divorced | No)
  × P(Income=120K | No)
  = 4/7 × 1/7 × 0.0072 = 0.0006

- P(X | Yes) = P(Refund=No | Yes)
  × P(Divorced | Yes)
  × P(Income=120K | Yes)
  = 1 × 1/3 × 1.2 × 10-9 = 4 × 10-10

- Since P(X|No)P(No) > P(X|Yes)P(Yes)

- Therefore P(No|X) > P(Yes|X) => Class = No

Source: Tan et al., 2019

Naïve Bayes Classifiers can make decisions with partial information about attributes in the test record

- Even in absence of information about any attributes, we can use Apriori Probabilities of Class Variable, i.e.:
  - P(Yes)=3/10
  - P(No)=7/10.

- **If we only know that marital status is Divorced, then:**
  - P(Yes | Divorced) = 1/3 x 3/10 / P(Divorced)
  - P(No | Divorced) = 1/7 x 7/10 / P(Divorced)

- **If we also know that Refund = No, then:**
  - P(Yes | Refund = No, Divorced) = 1 x 1/3 x 3/10 /P(Divorced, Refund = No)
  - P(No | Refund = No, Divorced) = 4/7 x 1/7 x 7/10 /P(Divorced, Refund = No)

- **If we also know that Taxable Income = 120, then:**
  - P(Yes | Refund = No, Divorced, Income = 120) =1.2 x10^{-9} x 1 x 1/3 x 3/10 / P(Divorced, Refund = No, Income = 120)
  - P(No | Refund = No, Divorced Income = 120) = 0.0072 x 4/7 x 1/7 x 7/10 /P(Divorced, Refund = No, Income = 120)

Source: Tan et al., 2019

Issues with Naïve Bayes Classifier

Given a Test Record: X = (Married)

- **Naïve Bayes Classifier:**
  - P(Refund = Yes | No) = 3/7
  - P(Refund = No | No) = 4/7
  - P(Refund = Yes | Yes) = 0
  - P(Refund = No | Yes) = 1
  - P(Marital Status = Single | No) = 2/7
  - P(Marital Status = Divorced | No) = 1/7
  - P(Marital Status = Married | No) = 4/7
  - P(Marital Status = Single | Yes) = 2/3
  - P(Marital Status = Divorced | Yes) = 1/3
  - P(Marital Status = Married | Yes) = 0

- **For Taxable Income:**
  - If class = No, sample mean = 110, sample variance = 2975
  - If class = Yes, sample mean = 90, sample variance = 25

P(Yes) = 3/10

P(No) = 7/10

P(Yes | Married) = 0 x 3/10 / P(Married)

P(No | Married) = 4/7 x 7/10 / P(Married)

Source: Tan et al., 2019

Issues with Naïve Bayes Classifier

Consider the table with Tid = 7 deleted

Tid	Refund	Marital Status	Taxable Income	Evade
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Naïve Bayes Classifier:

P(Refund = Yes | No) = 2/6
P(Refund = No | No) = 4/6
P(Refund = Yes | Yes) = 0
P(Refund = No | Yes) = 1
P(Marital Status = Single | No) = 2/6
P(Marital Status = Divorced | No) = 0
P(Marital Status = Married | No) = 4/6
P(Marital Status = Single | Yes) = 2/3
P(Marital Status = Divorced | Yes) = 1/3
P(Marital Status = Married | Yes) = 0/3
For Taxable Income:
If class = No: sample mean = 91
sample variance = 685
If class = No: sample mean = 90
sample variance = 25

Given X = (Refund = Yes, Divorced, 120K)

$P(X \mid No) = 2/6 \times 0 \times 0.0083 = 0$

$P(X \mid Yes) = 0 \times 1/3 \times 1.2 \times 10^{-9} = 0$

Naïve Bayes will not be able to classify X as Yes or No!

Source: Tan et al., 2019

Issues with Naïve Bayes Classifier

- If one of the conditional probabilities is zero, then the entire expression becomes zero
- Need to use other estimates of conditional probabilities than simple fractions.
- Probability estimation:

original: $P(X_i = c|y) = \frac{n_c}{n}$

Laplace Estimate: $P(X_i = c|y) = \frac{n_c + 1}{n + v}$

m − estimate: $P(X_i = c|y) = \frac{n_c + mp}{n + m}$

- n: number of training instances belonging to class y.
- nc: number of instances with Xi = c and Y = y.
- v: total number of attribute values that Xi can take.
- p: initial estimate of (P(Xi = c|y) known apriori.
- m: hyper-parameter for our confidence in p.

Source: Tan et al., 2019

Numeric Example of Naïve Bayes Classifier

Name	Give Birth	Can Fly	Live in Water	Have Legs	Class
human	yes	no	no	yes	mammals
python	no	no	no	no	non-mammals
salmon	no	no	yes	no	non-mammals
whale	yes	no	yes	no	mammals
frog	no	no	sometimes	yes	non-mammals
komodo	no	no	no	yes	non-mammals
bat	yes	yes	no	yes	mammals
pigeon	no	yes	no	yes	non-mammals
cat	yes	no	no	yes	mammals
leopard shark	yes	no	yes	no	non-mammals
turtle	no	no	sometimes	yes	non-mammals
penguin	no	no	sometimes	yes	non-mammals
porcupine	yes	no	no	yes	mammals
eel	no	no	yes	no	non-mammals
salamander	no	no	sometimes	yes	non-mammals
gila monster	no	no	no	yes	non-mammals
platypus	no	no	no	yes	mammals
owl	no	yes	no	yes	non-mammals
dolphin	yes	no	yes	no	mammals
eagle	no	yes	no	yes	non-mammals

Give Birth	Can Fly	Live in Water	Have Legs	Class
yes	no	yes	no	?

A: attributes

M: mammals

N: non-mammals

$$P(A \mid M) = \frac{6}{7} \times \frac{6}{7} \times \frac{2}{7} \times \frac{2}{7} = 0.06$$

$$P(A \mid N) = \frac{1}{13} \times \frac{10}{13} \times \frac{3}{13} \times \frac{4}{13} = 0.0042$$

$$P(A \mid M)P(M) = 0.06 \times \frac{7}{20} = 0.021$$

$$P(A \mid N)P(N) = 0.004 \times \frac{13}{20} = 0.0027$$

$P(A \mid M)P(M) > P(A \mid N)P(N) \Rightarrow$ Mammals

Source: Tan et al., 2019

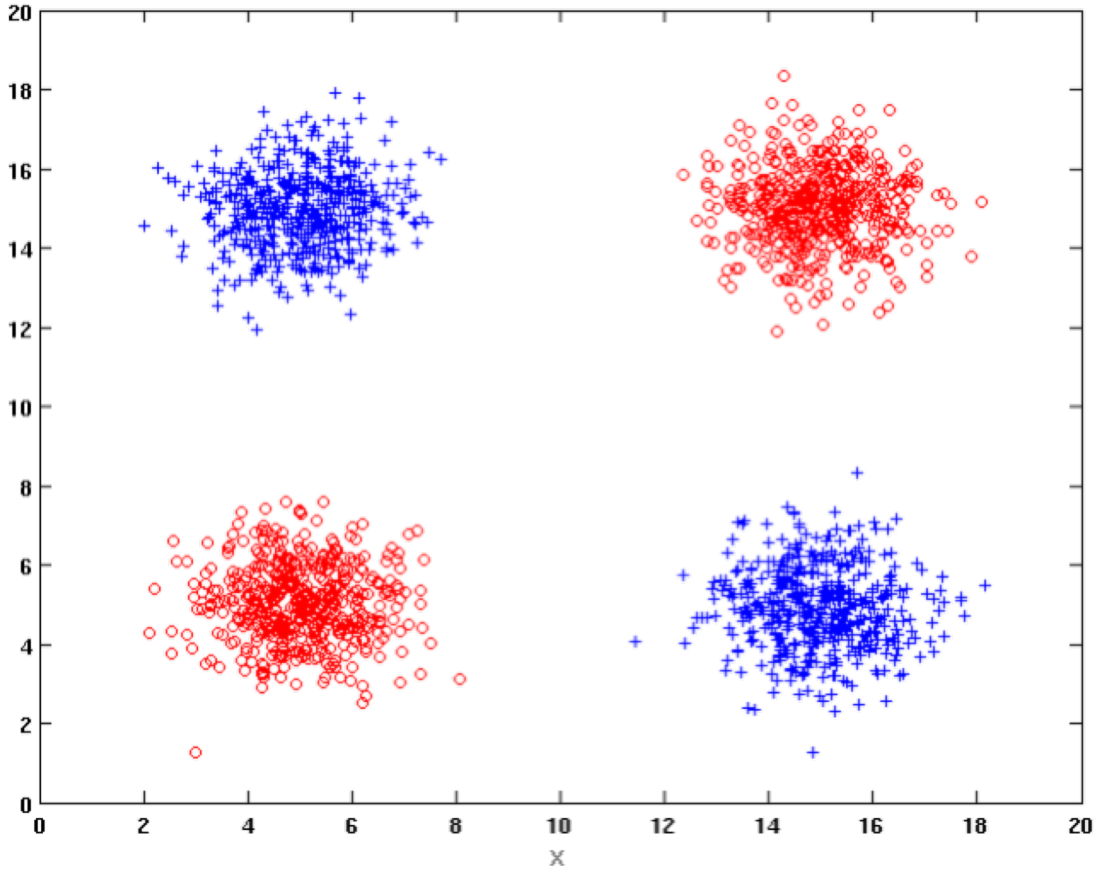
Naïve Bayes (Summary)

- Robust to isolated noise points
- Handle missing values by ignoring the instance during probability estimate calculations
- Robust to irrelevant attributes
- Redundant and correlated attributes will violate class conditional assumption
  - Use other techniques such as Bayesian Belief Networks (BBN)

Source: Tan et al., 2019

Naïve Bayes

- How does Naïve Bayes perform on the following dataset?

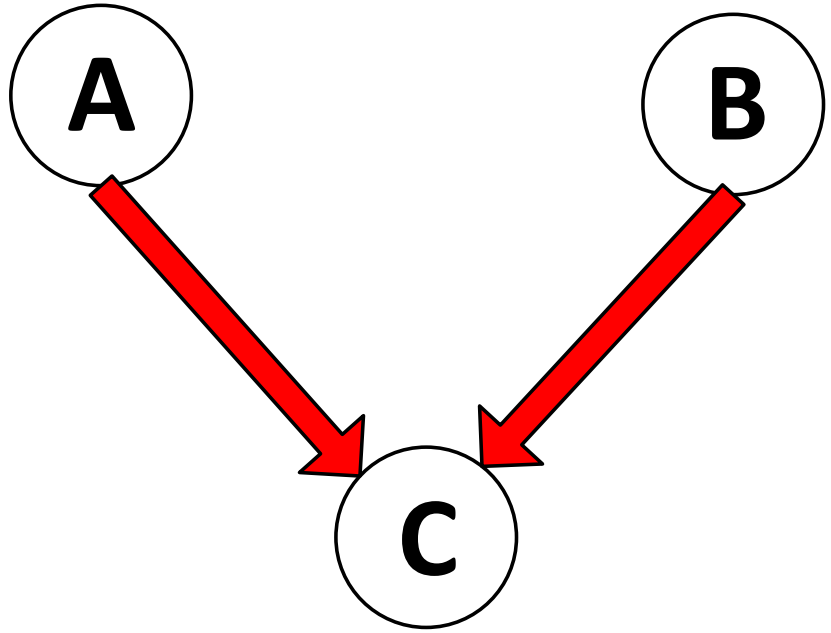


Conditional independence of attributes is violated

Source: Tan et al., 2019

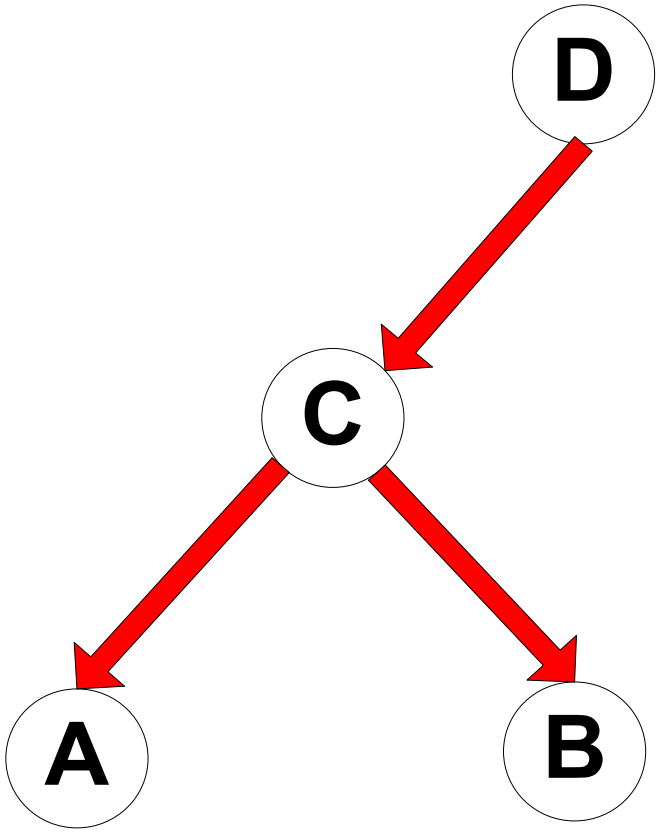
Bayesian Belief Networks

- Provides graphical representation of probabilistic relationships among a set of random variables.
- Consists of a directed acyclic graph (DAG):
  - Each node corresponds to a variable.
  - Each arc corresponds to dependence relationship between a pair of variables.
- A probability table associating each node to its immediate parent.



Source: Tan et al., 2019

Conditional Independence



D is parent of C

A is child of C

B is descendant of D

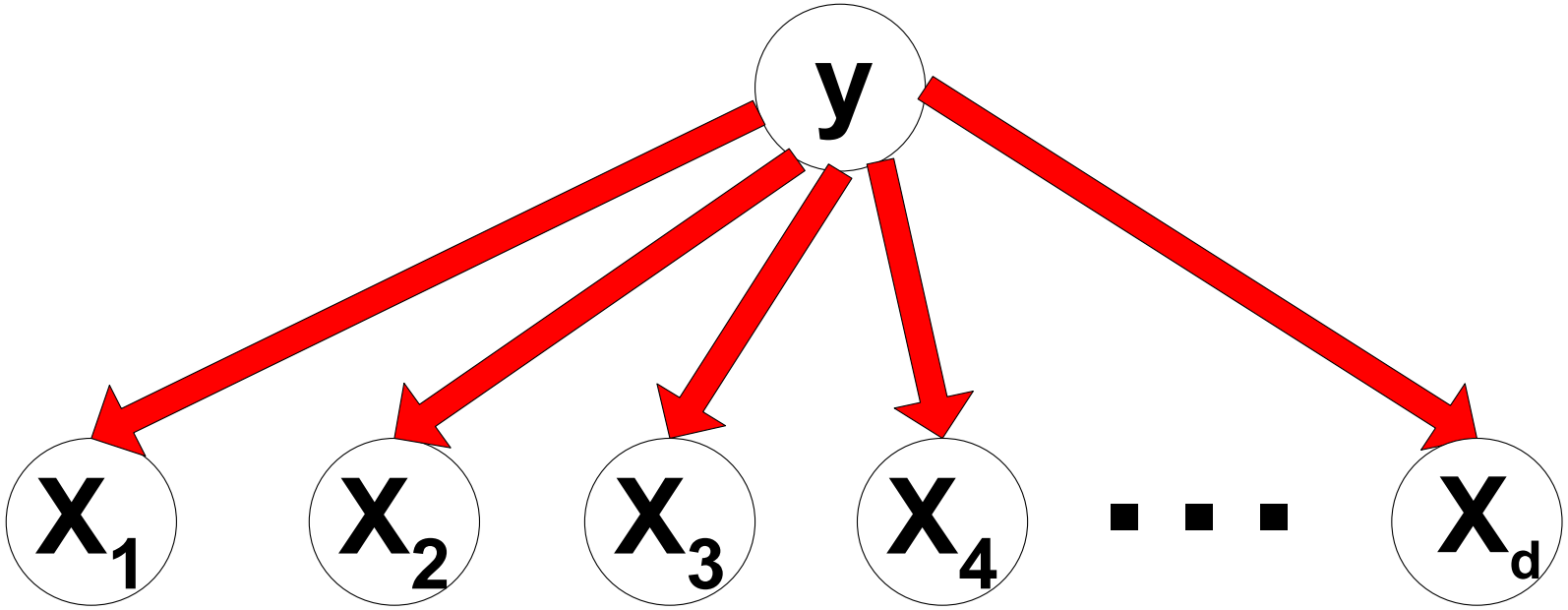
D is ancestor of A

- A node in a Bayesian network is conditionally independent of all of its nondescendants, if its parents are known.

Source: Tan et al., 2019

Conditional Independence

- Naïve Bayes assumption:

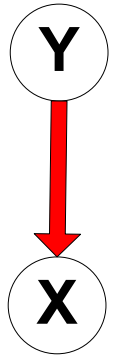


Source: Tan et al., 2019

Probability Tables

- If X does not have any parents, table contains prior probability P(X).

- If X has only one parent (Y), table contains conditional probability P(X|Y)



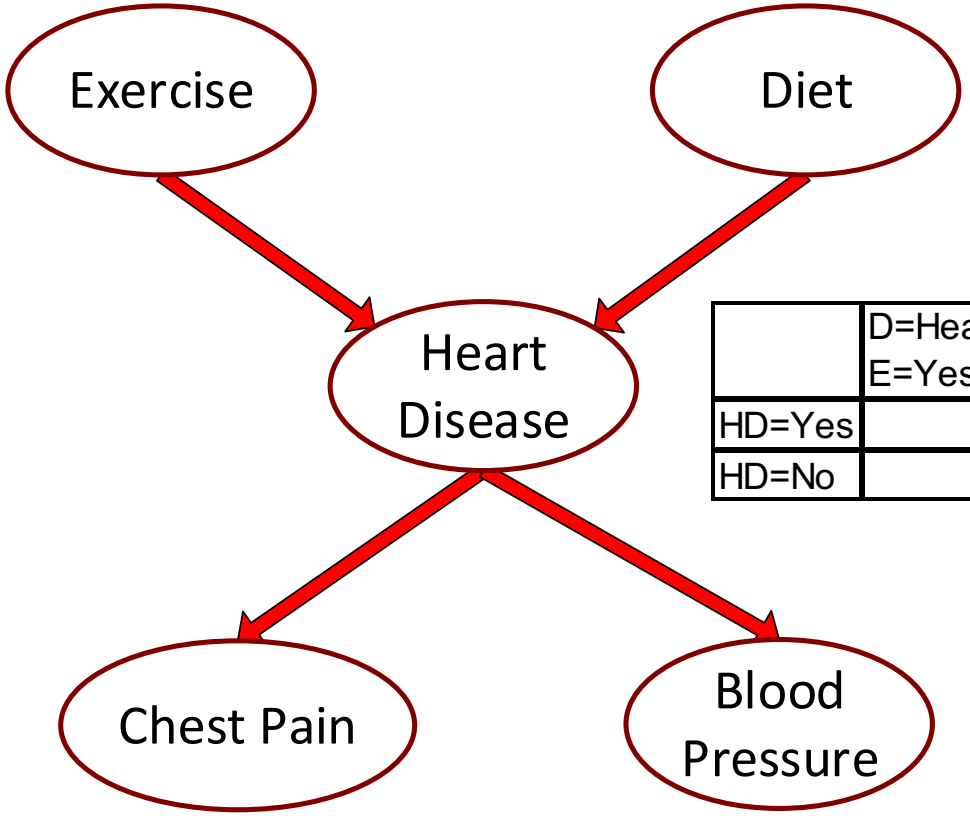
- If X has multiple parents (Y_1, Y_2,…, Y_k), table contains conditional probability P(X|Y_1, Y_2,…, Y_k)

Source: Tan et al., 2019

Example of Bayesian Belief Network

Exercise=Yes	0.7
Exercise=No	0.3

Diet=Healthy	0.25
Diet=Unhealthy	0.75



	D=Healthy E=Yes	D=Healthy E=No	D=Unhealthy E=Yes	D=Unhealthy E=No
HD=Yes	0.25	0.45	0.55	0.75
HD=No	0.75	0.55	0.45	0.25

	HD=Yes	HD=No
CP=Yes	0.8	0.01
CP=No	0.2	0.99

	HD=Yes	HD=No
BP=High	0.85	0.2
BP=Low	0.15	0.8

Source: Tan et al., 2019

Numeric Example of Inferencing using BBN

- Given: X = (E=No, D=Yes, CP=Yes, BP=High)
  - P(HD|E,D,CP,BP) ?

- P(HD=Yes| E=No,D=Yes) = 0.55
  P(CP=Yes| HD=Yes) = 0.8
  P(BP=High| HD=Yes) = 0.85

  - P(HD=Yes|E=No,D=Yes,CP=Yes,BP=High)
    $\propto 0.55 \times 0.8 \times 0.85 = 0.374$

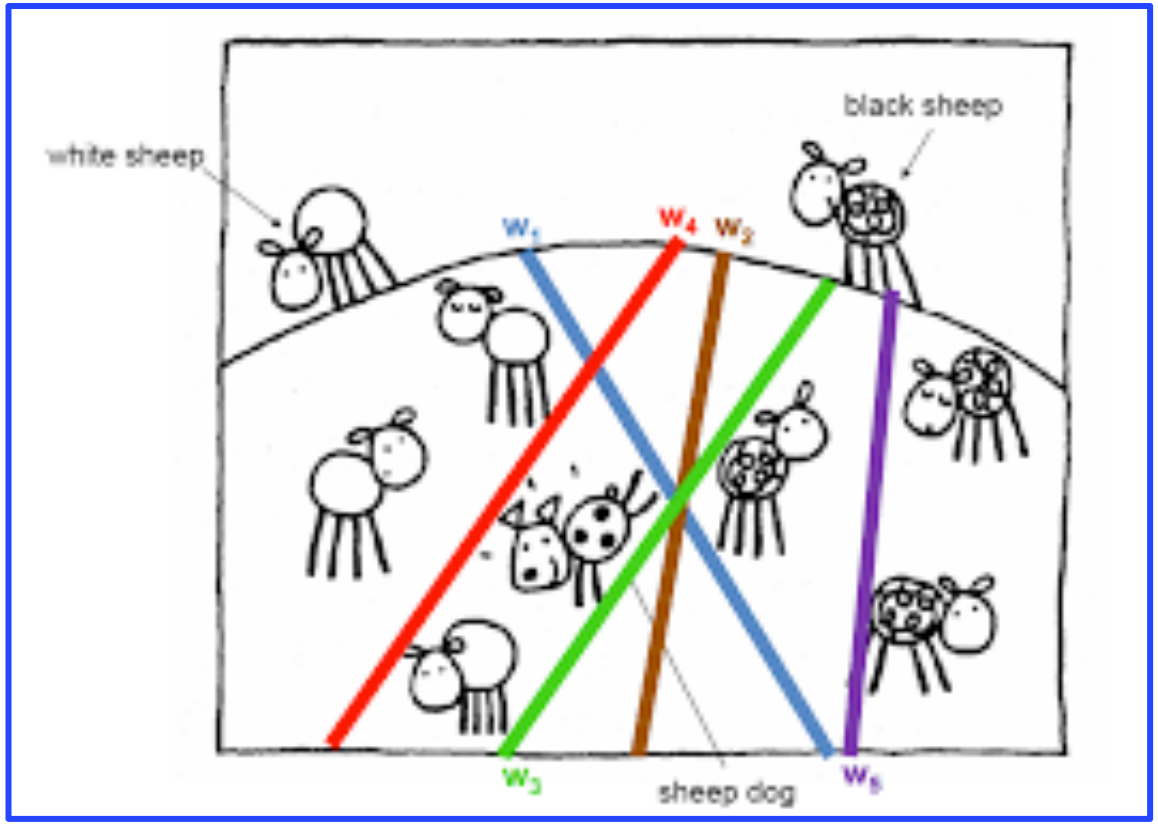
- P(HD=No| E=No,D=Yes) = 0.45
  P(CP=Yes| HD=No) = 0.01
  P(BP=High| HD=No) = 0.2

  - P(HD=No|E=No,D=Yes,CP=Yes,BP=High)
    $\propto 0.45 \times 0.01 \times 0.2 = 0.0009$

Classify X as Yes

Source: Tan et al., 2019

Support Vector Machines



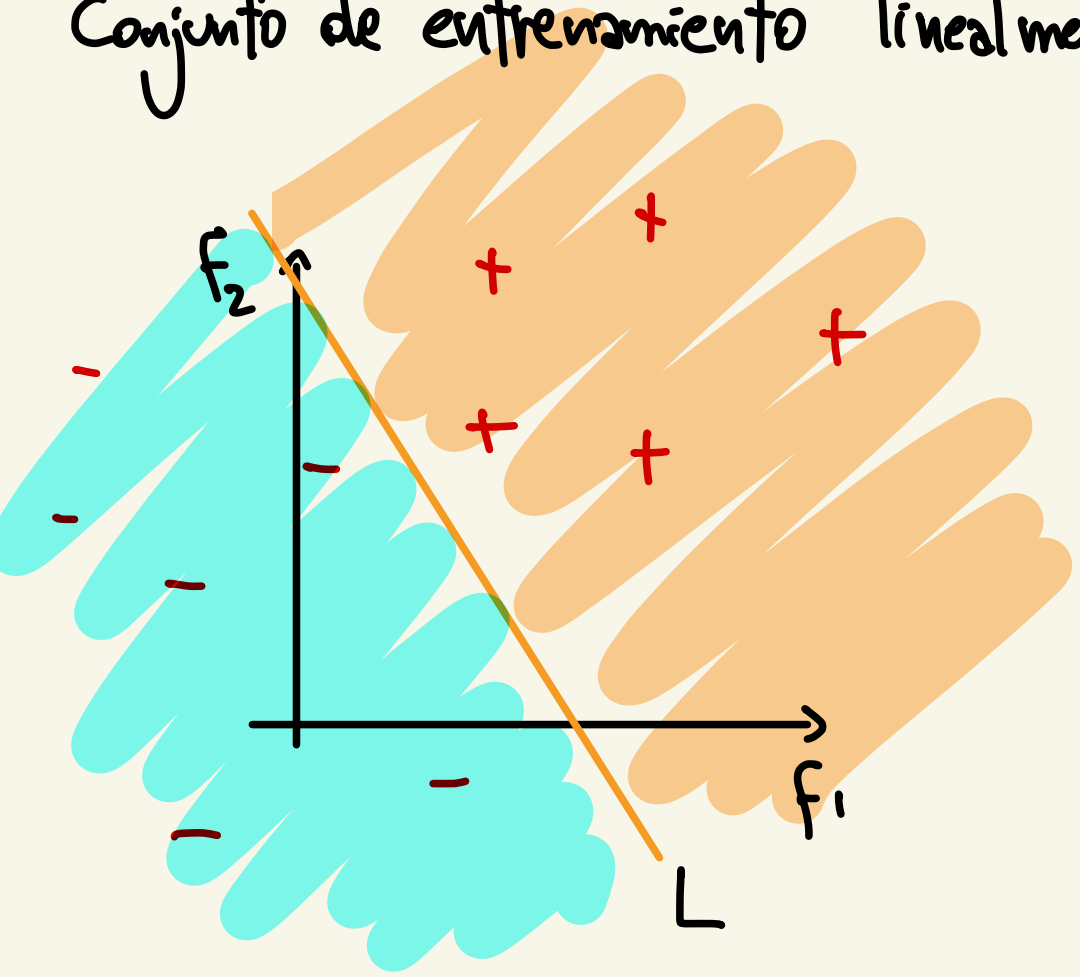
Support Vector Machine

Prof. Angel Jimenez M.

SVM

Clasificación binaria.

Conjunto de entrenamiento linealmente separable.



+ : diabetes

− : ~

$x \in \mathbb{R}^2$, $x = (f_1, f_2)$

$y \in \{+1, -1\}$

↳ $x \in \mathbb{R}^n$, $y \in \{+1, -1\}$

↳ Hiperplano separador

f_1	f_2	y

$x \in \mathbb{R}^2 \longrightarrow$ Ecuación de la recta:

$$x_2 = m x_1 + n$$

$$w_1 x_1 + w_2 x_2 + b = 0$$

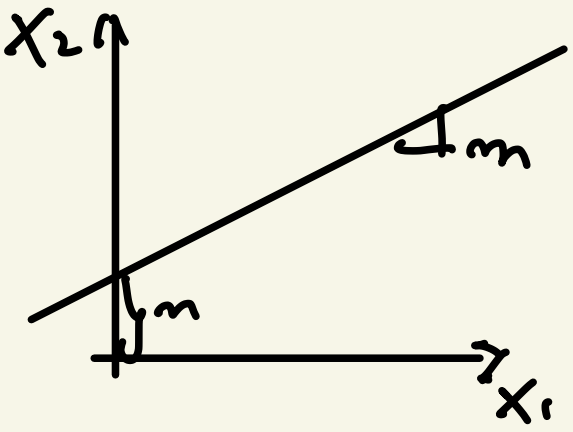
$$x_2 = \underbrace{\frac{-w_1}{w_2}}_{\text{pendiente}} x_1 - \underbrace{\frac{b}{w_1}}_{\text{sesgo}}$$

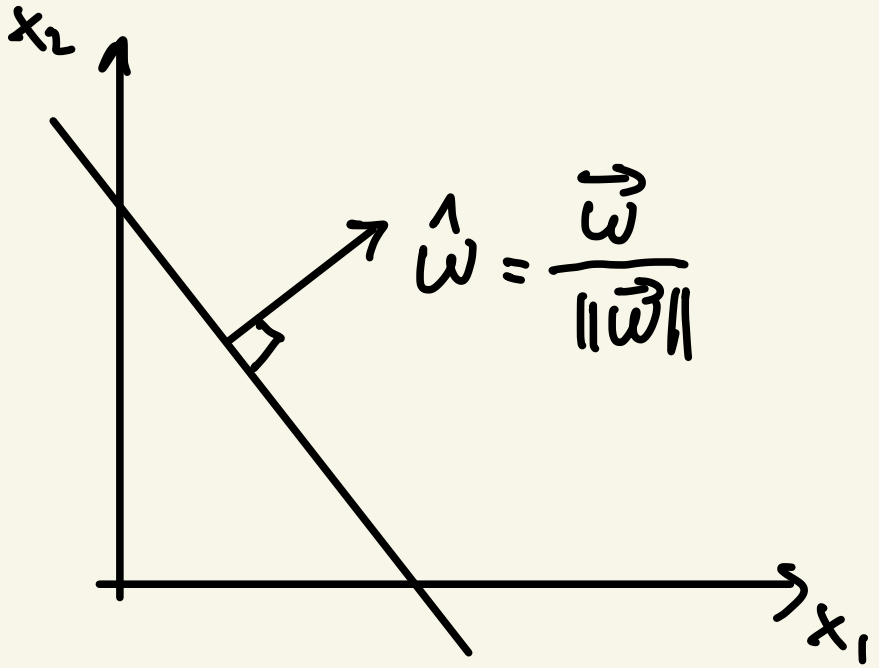
$$\vec{X} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \quad \vec{w} = \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$

Ecuación del hiperplano:

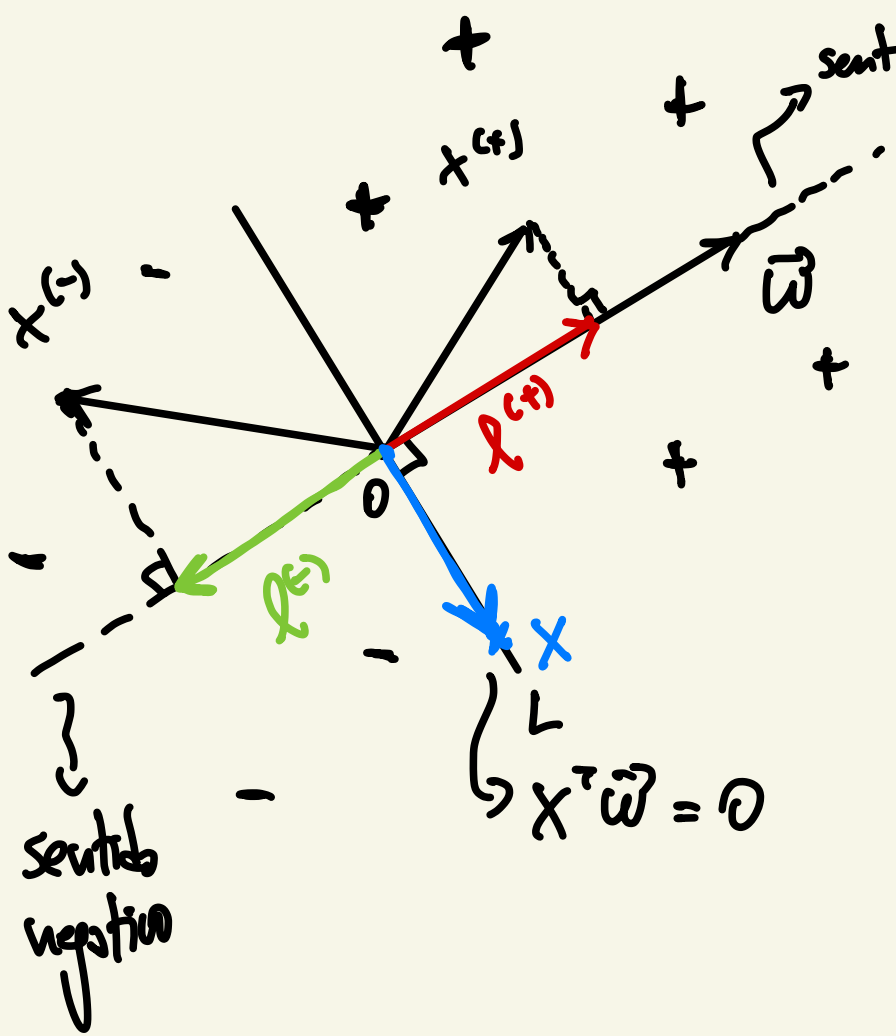
$$\boxed{\vec{w}^T \cdot X + b = 0}$$

$$x \in \mathbb{R}^n, \ \vec{w} \in \mathbb{R}^n$$





(Paréntesis: producto punto)



$$x^T \vec{w} = [x_1 \quad x_2] \cdot \begin{bmatrix} w_1 \\ w_2 \end{bmatrix}$$

$$= x_1 w_1 + x_2 w_2$$

$$= l^{(+)} \cdot \|\vec{w}\|$$

(1) Para puntos positivos

$$x^T \vec{w} \geq C \text{, semiplano (+)}$$

(2) Para puntos negativos:

$$x^T \vec{w} \leq C \text{, semiplano (-)}$$

(3) Para puntos en el hiperplano:

$$x^T \vec{w} = 0$$

Sea T: conjunto de entrenamiento

$$T = \{(x^{(i)}, y^{(i)}), i = 1, \dots m\}, \quad x^{(i)} \in \mathbb{R}^m, \quad y^{(i)} \in \{+1, -1\}$$

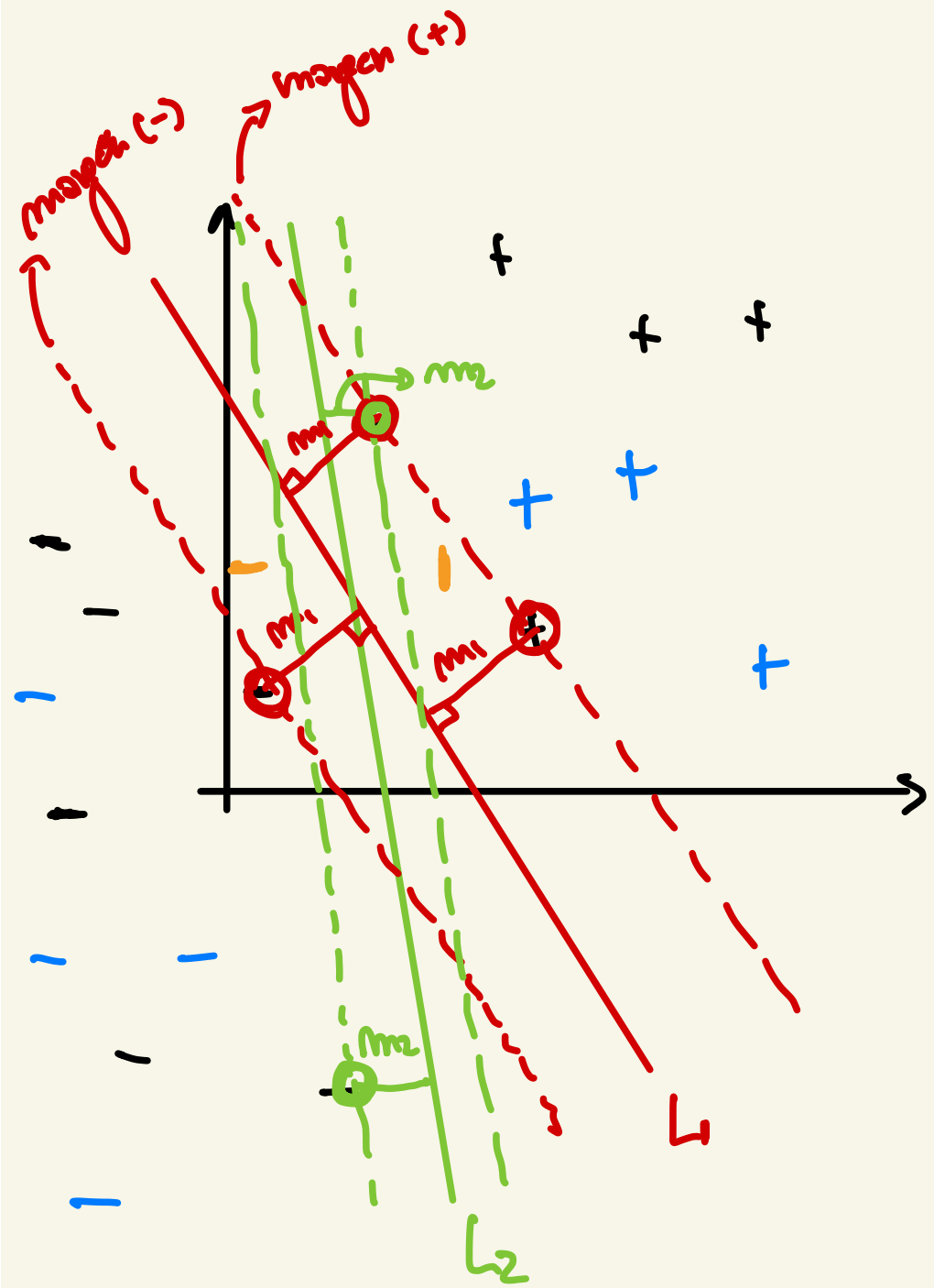
Si T es linealmente separable: $\exists\ \vec{w} \in \mathbb{R}^n \wedge b \in \mathbb{R}$, t.q.:

$$\vec{w}^T x^{(i)} + b \geq 0, \quad \forall i \,/\, y^{(i)} = +1$$

$$\vec{w}^T x^{(i)} + b < 0, \quad \forall i \,/\, y^{(i)} = -1$$

↑ Estrictas (*)

$$\vec{w}^T x^{(i)} + b = 0 \leftarrow \text{Hiperplano}$$



¿ $L_1, L_2, \ldots$?

<u>Intuición</u>: Hay hiperplanos mejores que otros

$m_1 > m_2$

<u>Margen</u>: distancia de los puntos más cercanos al hiperplano.

Como $m_1 > m_2 \Rightarrow L_1$ es mejor que L_2.

Hiperplano de **máximo** margen

Dado (*) puedo decir que $\exists \varepsilon_1$, t.q.:

$$\vec{w}^T x^{(i)} + b \geq \varepsilon_1, \quad \forall_i / y^{(i)} = +1, \wedge$$

$\exists \varepsilon_2$, t.q.:

$$\vec{w}^T x^{(i)} + b \leq -\varepsilon_2, \quad \forall_i / y^{(i)} = -1.$$

Sea $\varepsilon = \min\{\varepsilon_1, \varepsilon_2\}$

$$\Rightarrow \vec{w}^T x^{(i)} + b \geq \varepsilon, \quad \forall_i / y^{(i)} = +1,$$
$$\vec{w}^T x^{(i)} + b \leq -\varepsilon, \quad \forall_i / y^{(i)} = -1.$$

Si divido por ε:

$$\vec{w}^T x^{(i)} + b \geq 1, \quad \forall_i / y^{(i)} = +1,$$
$$\vec{w}^T x^{(i)} + b \leq -1, \quad \forall_i / y^{(i)} = -1$$

De manera compacta:

$$y^{(i)} (\vec{w}^T x^{(i)} + b) \geq 1$$

$$\forall i = 1, \dots, n. \quad (1)$$

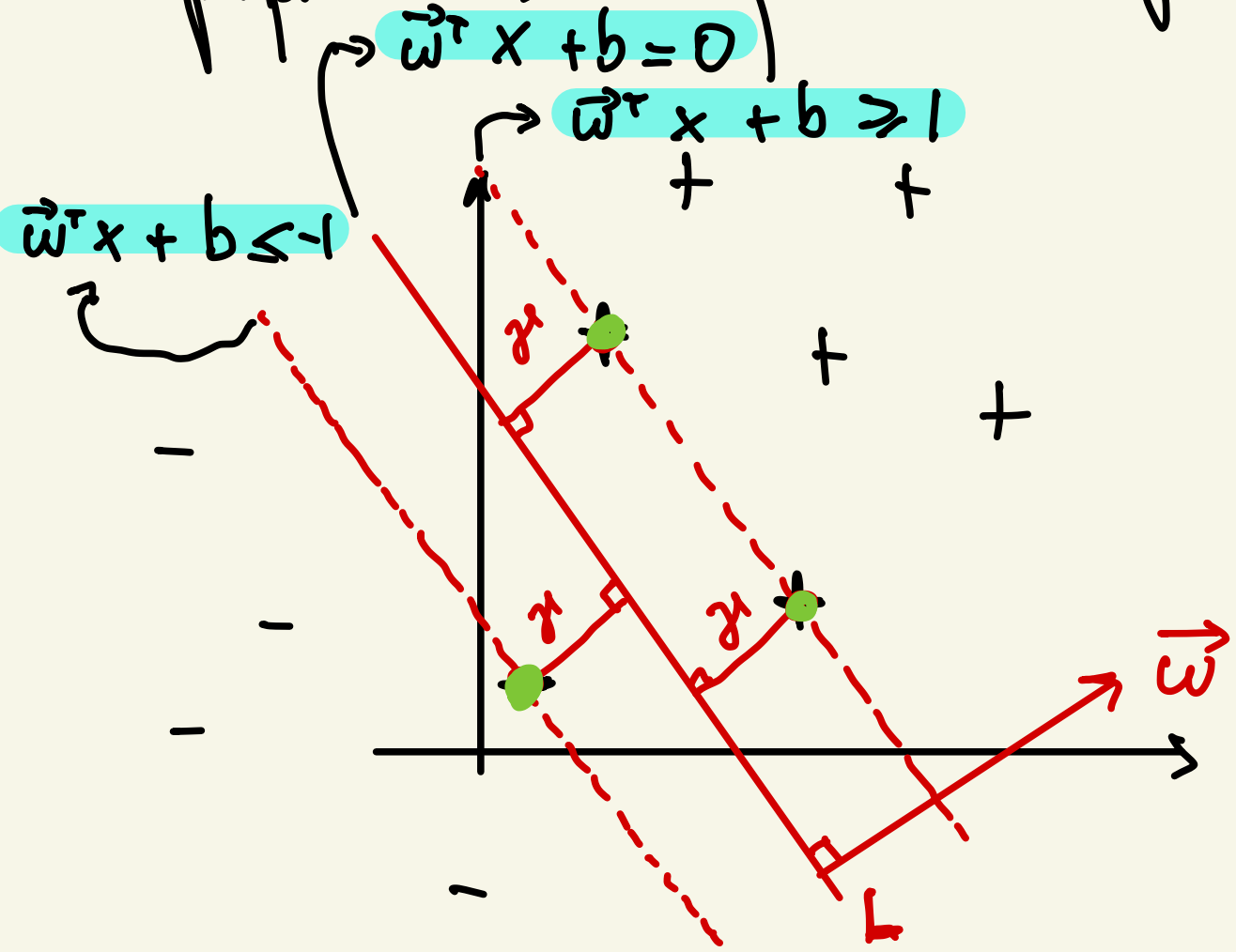
n desigualdades lineales.

Cuando el conjunto de entrenamiento T es linealmente separable, siempre se puede encontrar un hiperplano separador que satisfaga (1), escalando $\vec{w}$ y b adecuadamente.

Hiperplano $\longrightarrow$ clasificador : signo $(\vec{w}^T x^* + b)$

↳ $0.0000001 > 0 \rightarrow +1$

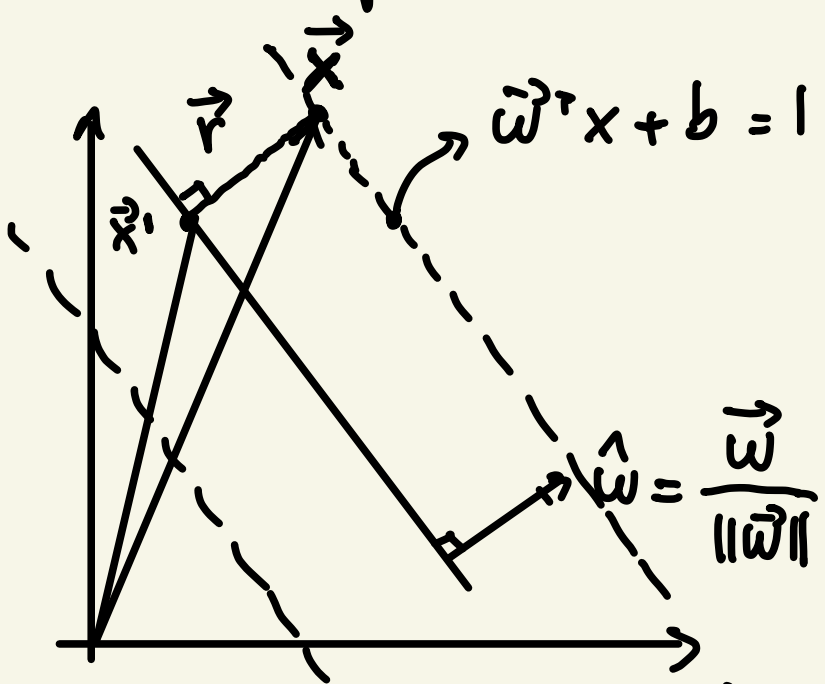
↳ $9999999 > 0 \rightarrow +1$



● : Vectores de soporte.

Buscar el hiperplano separador óptimo de máximo margen.

Buscando expresión para γ:



① : $\vec{r} = \vec{x} - \vec{x}'$,

② : $\vec{w}^T \vec{x}' + b = 0$.

Recordar:
$\vec{w}^T \vec{w} = \|\vec{w}\|^2$

Sea γ el margen:

$$\vec{r} = \gamma \cdot \hat{w} = \gamma \cdot \frac{\vec{w}}{\|\vec{w}\|}$$

De ①: $\vec{x}' = \vec{x} - \vec{r} = \vec{x} - \gamma \frac{\vec{w}}{\|\vec{w}\|}$

En ②: $\vec{w}^T \left(\vec{x} - \gamma \frac{\vec{w}}{\|\vec{w}\|} \right) + b = 0$

$$\vec{w}^T \vec{x} - \frac{\gamma \, \vec{w}^T \cdot \vec{w}}{\|\vec{w}\|} + b = 0$$

($\vec{w}^T \cdot \vec{w} = \|\vec{w}\|^2$)

$$\gamma = \frac{\vec{w}^T x^{(i)} + b}{\|\vec{w}\|} \, , \; y^{(i)} = +1 \; ; \quad \gamma = -\left(\frac{\vec{w}^T x^{(i)} + b}{\|\vec{w}\|} \right) , \; y^{(i)} = -1$$

$$\gamma = y^{(i)} \left(\frac{\vec{w}^T x^{(i)} + b}{\|\vec{w}\|} \right) \quad \forall i = 1, \ldots, n.$$

$$\text{Margen total} = +1 \cdot \frac{1}{\|\vec{w}\|} + (-1) \cdot \frac{(-1)}{\|\vec{w}\|} = \frac{1}{\|\vec{w}\|} + \frac{1}{\|\vec{w}\|} = \frac{2}{\|\vec{w}\|}$$

$$\boxed{\text{Margen total} = \frac{2}{\|\vec{w}\|}}$$

↳ Buscamos maximizar este margen total

Formulando el problema de optimización:

$$\max_{\vec{w}} \frac{2}{\|\vec{w}\|} \iff \min_{\vec{w}} \frac{1}{2}\|\vec{w}\| \iff \min_{\vec{w}} \frac{1}{2}\|\vec{w}\|^2$$

Encontrar $\vec{w} \in \mathbb{R}^n$, $b \in \mathbb{R}$, a través de:

$$\min_{\vec{w}, b} \frac{1}{2}\|\vec{w}\|^2$$

$$\text{s.a.}: \; y^{(i)}\left(\vec{w}^T x^{(i)} + b\right) \geq 1, \quad \forall i = 1, \ldots, m.$$

Optimización de una (fcn.) cuadrática con "m" restricciones de desigualdad lineales.

Support Vector Machine (Continuación)

Caso binario
T → linealmente separable.

Profesor : Ángel Jiménez.

Recapitulando:

Encontrar $\vec{w} \in \mathbb{R}^n$ ∧ $b \in \mathbb{R}$, a través de:

$$\min_{\vec{w}, b} \quad \frac{1}{2} \|\vec{w}\|^2$$

$$\text{s.a.:} \quad y^{(i)}(\vec{w}^T x^{(i)} + b) - 1 \geq 0, \quad \forall i = 1, \dots, n.$$

(Paréntesis: Optimización Lagrangeana).

PRIMAL:

$$\min_{\vec{w}} \quad l(\vec{w})$$

$$\text{s.a.:} \quad g_i(\vec{w}) \geq 0 \,, \quad (\forall i = 1, \dots, n)$$

Si w^* es la solución del primal:

① $g_i(w^*) \geq 0$

② $l(w^*) \leq l(w)$, $\forall w \in \mathbb{R}^n$, t.q.: $g_i(w) \geq 0$.

Lagrangeano: $\mathcal{L}(w) = l(w) - \sum_{i=1}^{n} \alpha_i \, g_i(w)$, $\forall i$

↳ Multiplicadores de Lagrange.

DUAL:

$$\max_{\alpha_i} \quad \mathcal{L}(w^*, \alpha)$$

$$s.t.: \quad \alpha_i \geq 0, \ \forall i$$

Teorema (Karush - Kuhn - Tucker): Si en el primal, las funciones $f: \mathbb{R}^n \to \mathbb{R}$ y $g_i: \mathbb{R}^n \to \mathbb{R}$, son convexas y:

C.1: $\nabla_w \mathcal{L}(w^*) = 0$,

C.2 $\alpha_i \cdot g_i(w^*) = 0 \quad \forall i$,

C.3 $\alpha_i \geq 0$.

Entonces: w^* es el mínimo global del problema primal.

① Lagrangeano:

$\mathcal{L}$ = Función de costo a optimizar $-$ $\alpha \cdot$ [restricciones]

$\vec{w} \in \mathbb{R}^n$, $b \in \mathbb{R}$, $\alpha \in \mathbb{R}^n$

② Resolver para $(\vec{w}, b, \alpha)$:

$\nabla \mathcal{L} = 0$

+ restricciones

} Conjunto de ecuaciones lineales

w^*, b^*

¿α? ← problema dual

$$\max_{\substack{\alpha \\ \alpha_i \geq 0}} \mathcal{L}(w^*, \alpha)$$

Lagrangeano:

$w^T w$, $\frac{\partial w^T w}{\partial w} = 2w$

$$\mathcal{L}(w, b) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{n} \alpha_i \left[y^{(i)} (w^T x^{(i)} + b) - 1 \right]$$

$x^T w$, $\frac{\partial x^T w}{\partial w} = x$

$$\frac{\partial \mathcal{L}(w)}{\partial w} = \begin{bmatrix} \frac{\partial \mathcal{L}}{\partial w_1} \\ \vdots \\ \frac{\partial \mathcal{L}}{\partial w_m} \end{bmatrix} = \nabla_w \mathcal{L}(w), \quad w \in \mathbb{R}^n$$

$$\frac{\partial \mathcal{L}(w)}{\partial w} = \frac{1}{2} 2w - \sum_{i=1}^{n} \alpha_i y^{(i)} x^{(i)}$$

$$\frac{\partial \mathcal{L}(w)}{\partial w} = 0 \Rightarrow \boxed{w^* = \sum_{i=1}^{n} \alpha_i y^{(i)} x^{(i)}} \quad (1)$$

¿Qué pasa si: hago un K-fold cross validation?

¡Los w's van a ser diferentes!

$$\frac{\partial L(w)}{\partial b} = -\sum_{i=1}^{n} \alpha_i y^{(i)}, \quad (\forall_i)$$

$$\frac{\partial L(w)}{\partial b} = 0 \Rightarrow \boxed{\sum_{i=1}^{n} \alpha_i y^{(i)} = 0} \qquad (2)$$

Reemplazando (1) y (2) en $\mathcal{L}(w, b)$:

$$\mathcal{L}(w, b) = \frac{1}{2} \underbrace{\|w\|^2}_{\hookrightarrow w^T w} - \sum_{i=1}^{n} \alpha_i \left[y^{(i)} \left(w^T x^{(i)} + b \right) - 1 \right]$$

$$\mathcal{L}(w^*) = \frac{1}{2} \left(\sum_{i=1}^{n} \alpha_i y^{(i)} x^{(i)} \right) \cdot \left(\sum_{j=1}^{n} \alpha_j y^{(j)} x^{(j)T} \right)$$

$$- \left(\sum_{i=1}^{n} \alpha_i y^{(i)} x^{(i)} \right) \cdot \left(\sum_{j=1}^{n} \alpha_j y^{(j)} x^{(j)T} \right)$$

$$- \sum_{i=1}^{n} \alpha_i y^{(i)} b + \sum_{i=1}^{n} \alpha_i .$$

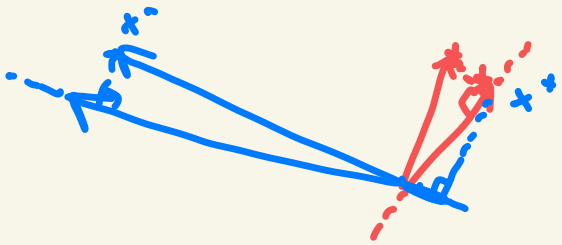
$\underbrace{\quad}_{0}$

$$\mathcal{L}(w^*) = -\frac{1}{2}\left(\sum_{i=1}^{n} \alpha_i\, y^{(i)}\, x^{(i)}\right)\cdot\left(\sum_{j=1}^{n} \alpha_j\, y^{(j)}\, x^{(j)^T}\right)$$

$$+ \sum_{i=1}^{n} \alpha_i \,.$$

$$\mathcal{L}(w^*) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \alpha_i \cdot \alpha_j \cdot \underbrace{y^{(i)} \cdot y^{(j)}}_{\text{Etiquetas}} \cdot x^{(i)} \cdot x^{(j)^T}$$

Producto punto entre dos ejemplos de entrenamiento.



DUAL:

$$\max_{\alpha} \quad \mathcal{L}(w^*, \alpha)$$

s.a.: $\alpha_i \geq 0, \ (\forall i)$

$$\mathcal{L}(w^*, \alpha) = f(w^*) - \sum \alpha_i \, g_i(w^*)$$

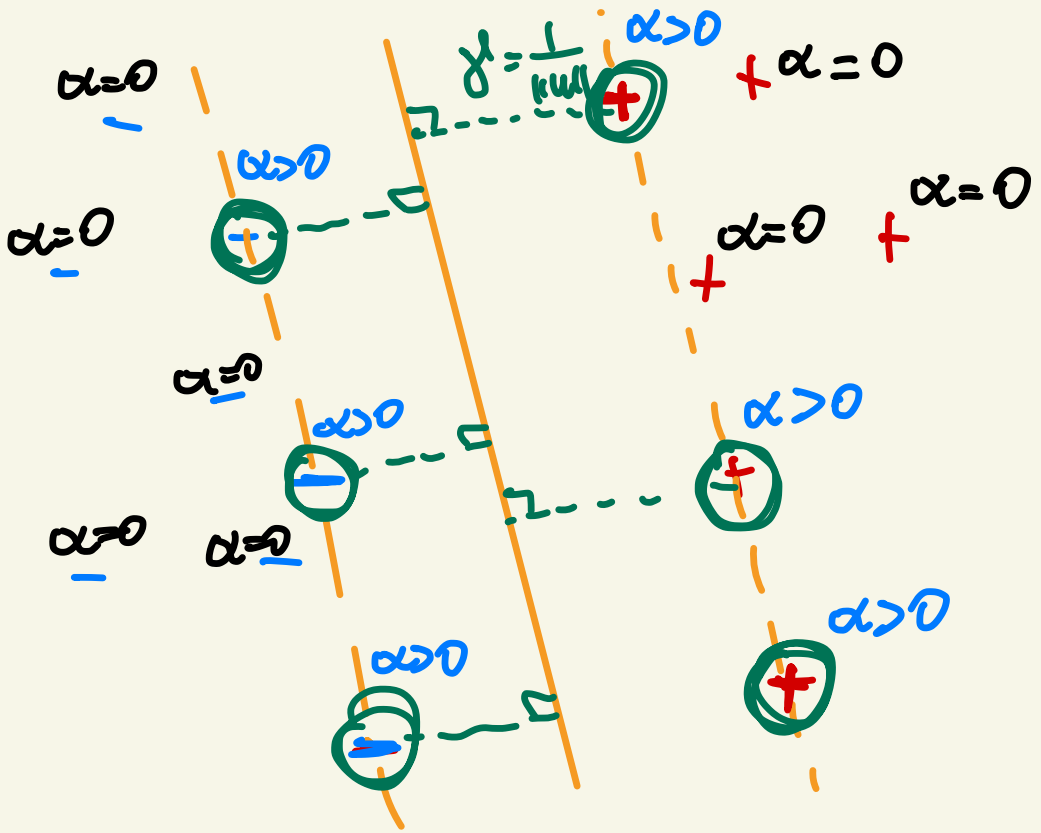
$g_i(w^*) = y^{(i)}[w^{*T} x^{(i)} + b] - 1 \geq 0$

$\alpha_i \geq 0$, $g_i(w^*) \geq 0$

$\sum \alpha_i \, g_i(w^*) \geq 0$ → Debe ser $\emptyset$.

Si $g_i(w^*) = 0 \ \wedge \ \alpha_i > 0 \implies y^{(i)}[w^{*T} x^{(i)} + b] - 1 = 0$

i.e.: $y^{(i)}[w^{*T} x^{(i)} + b] = 1$ ← Ecuaciones de los hiperplanos marginales.



Para los ejemplos del conjunto de entrenamiento que están en los hiperplanos marginales, $\alpha_i > 0$ y para el resto de los ejemplos $\alpha_i = 0$

$$\alpha = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ >0 \\ 0 \\ >0 \\ >0 \end{bmatrix}$$

→ Ejemplos sobre los hiperplanos marginales.

↳ Support Vectors

¡ Pues !

Sea $S = \{ x^s \;/\; y^{(i)}(w^T x^{(i)} + b) = 1 \;;\; s \in \{1, \dots, n\} \}$,
el conjunto de ejemplos en los hiperplanos marginales,
que denominaremos vectores de soporte. Entonces:

$$\boxed{w^* = \sum_{x^s \in S} \alpha_s y_s x^s}, \quad \alpha_s > 0.$$

Luego: $w^{*T} x^s + b = y_s$

$$\boxed{b^* = y_s - w^{*T} x^s}$$

Testing : Sea $X^{test} \in \mathbb{R}^n$ un conjunto de prueba:

$$\text{Signo}\left(w^{*T} X^{test} + b^* \right)$$

$$\text{Signo}\left((X^{test})^T \sum_{x^s \in S} \alpha_s y_s x^s + b^* \right)$$

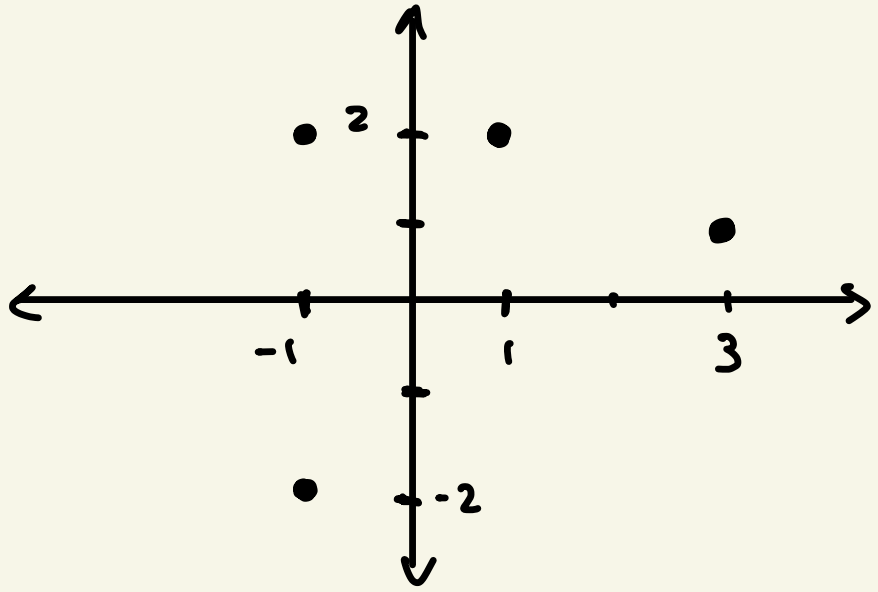
$$\in \{+1, -\}.$$

Ejercicio: (1) Encuentre el hiperplano separador óptimo

$x^{(i)} \in \mathbb{R}^2$

$x^{(1)} = \begin{bmatrix} 1 \\ 2 \end{bmatrix}$, $x^{(3)} = \begin{bmatrix} -1 \\ -2 \end{bmatrix}$

$x^{(2)} = \begin{bmatrix} -1 \\ 2 \end{bmatrix}$



(2) $x^{(4)} = \begin{bmatrix} 3 \\ 1 \end{bmatrix}$

Ensemble Methods

Ensemble Methods

- Construct a set of base classifiers learned from the training data
- Predict class label of test records by combining the predictions made by multiple classifiers (e.g., by taking majority vote)

Source: Tan et al., 2019

Example: Why Do Ensemble Methods Work?

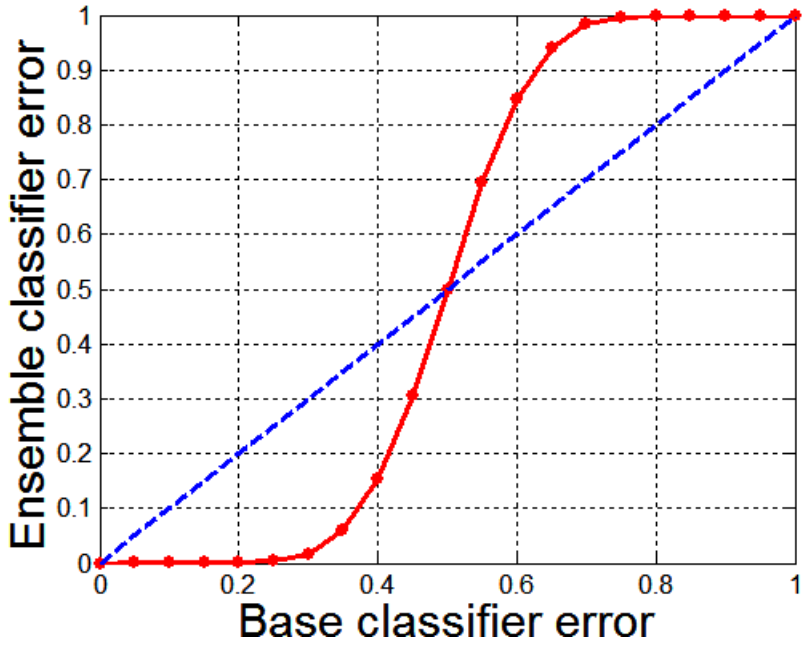
- Suppose there are 25 base classifiers
  - Each classifier has error rate, $\epsilon = 0.35$
  - Majority vote of classifiers used for classification
  - If all classifiers are identical:
    - Error rate of ensemble = ϵ (0.35)
  - If all classifiers are independent (errors are uncorrelated):
    - Error rate of ensemble = probability of having more than half of base classifiers being wrong

$$e_{\text{ensemble}} = \sum_{i=13}^{25} \binom{25}{i} \epsilon^i (1-\epsilon)^{25-i} = 0.06$$

Source: Tan et al., 2019

Necessary Conditions for Ensemble Methods

- Ensemble Methods work better than a single base classifier if:
  1. All base classifiers are independent of each other.
  2. All base classifiers perform better than random guessing (error rate < 0.5 for binary classification)



Classification error for an ensemble of 25 base classifiers, assuming their errors are uncorrelated.

Source: Tan et al., 2019

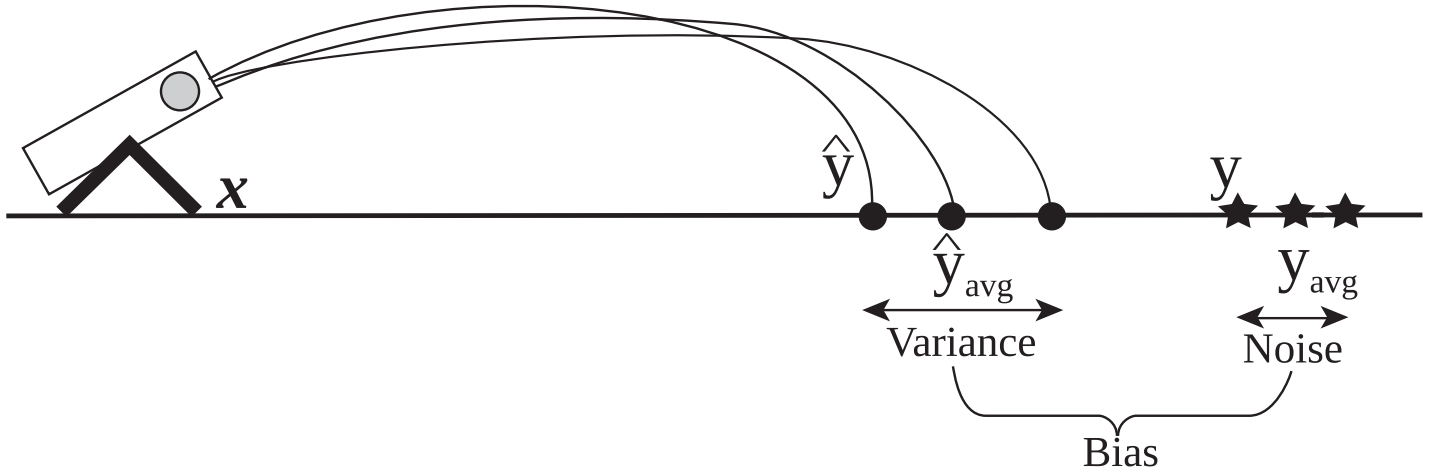
Rationale for Ensemble Learning

- Ensemble Methods work best with **unstable base classifiers.**
  - Classifiers that are sensitive to minor perturbations in training set, due to *high model complexity*
  - Examples: Unpruned decision trees, artificial neural networks, etc.

Source: Tan et al., 2019

Bias-Variance Decomposition

- Analogous problem of reaching a target y by firing projectiles from x (regression problem)

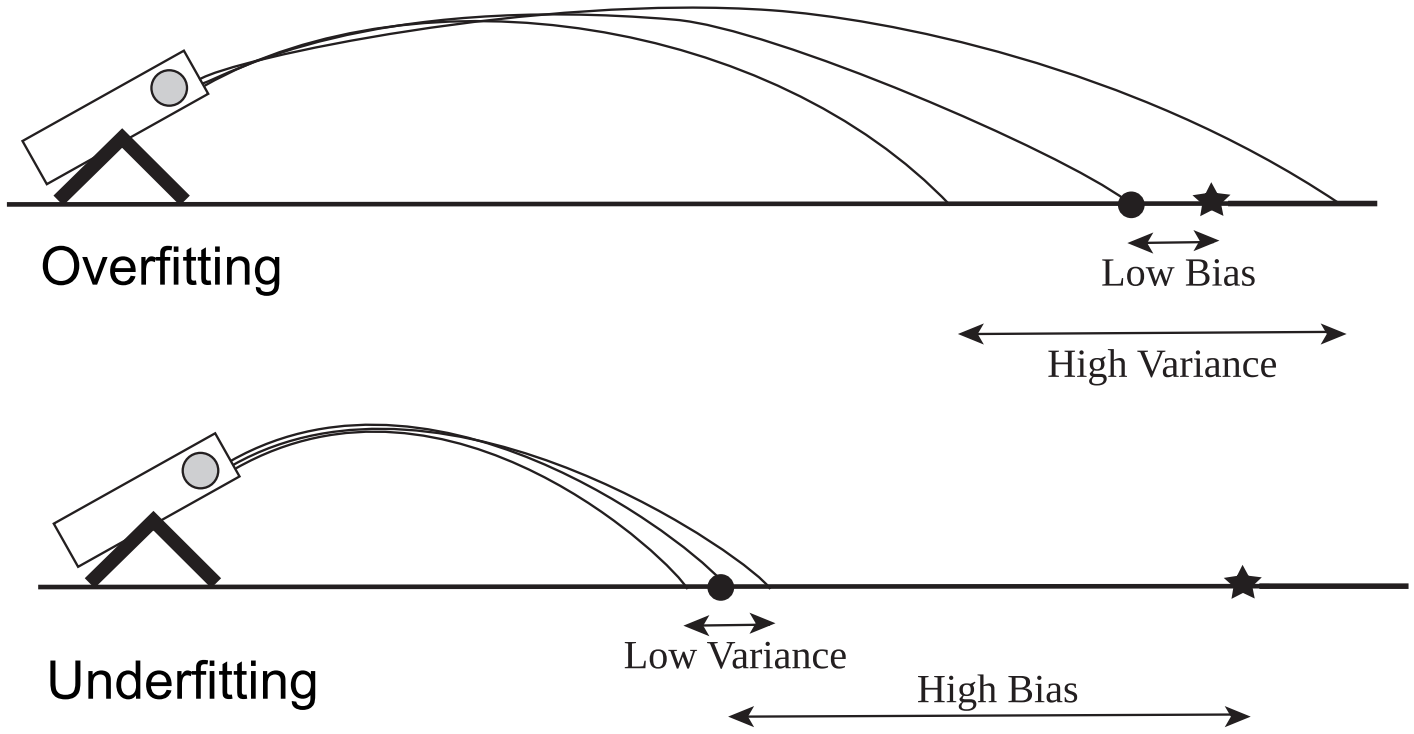


- For classification, the generalization error of model m can be given by:

$$gen.error(m) = c_1 + bias(m) + c_2 \times variance(m)$$

Source: Tan et al., 2019

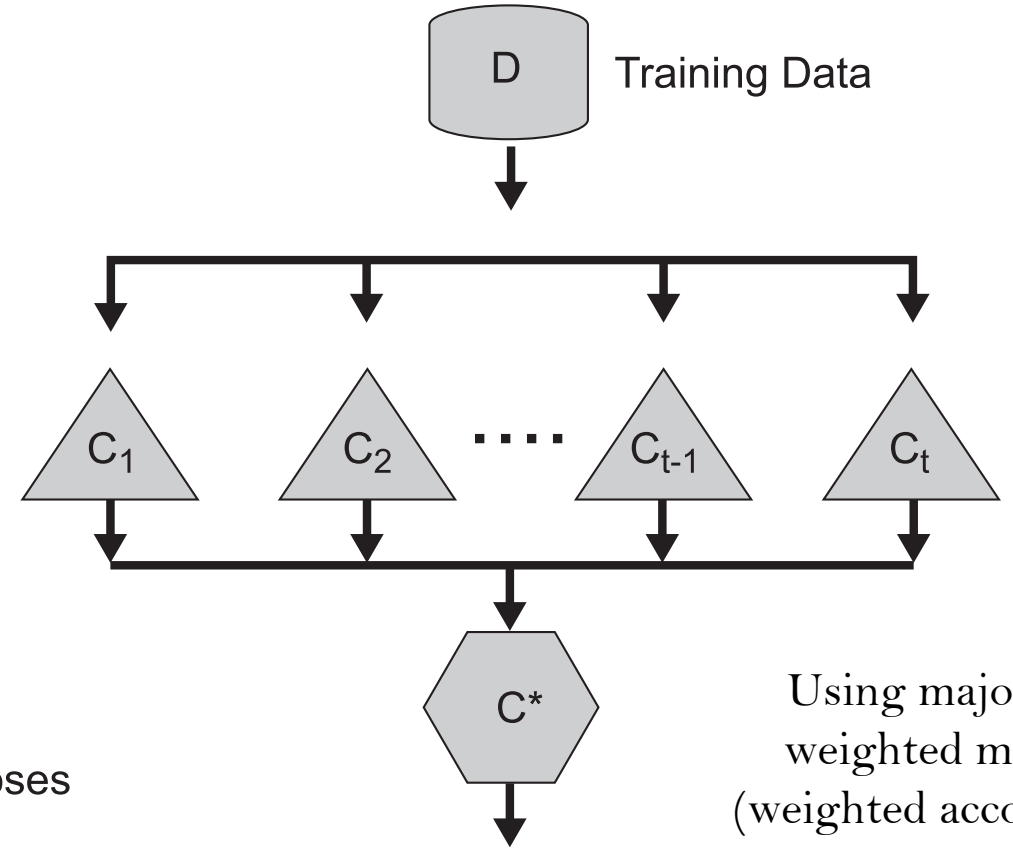
Bias-Variance Trade-off and Overfitting



- Ensemble methods try to reduce the variance of complex models (with low bias) by *aggregating* responses of multiple base classifiers

Source: Tan et al., 2019

General Approach of Ensemble Learning



Using majority vote or weighted majority vote (weighted according to their accuracy or relevance)

Source: Tan et al., 2019

Constructing Ensemble Classifiers

- By manipulating training set.
  - Example: bagging, boosting, random forests.
- By manipulating input features.
  - Example: random forests.
- By manipulating class labels.
  - Example: error-correcting output coding.
- By manipulating learning algorithm.
  - Example: injecting randomness in the initial weights of artificial neural network (ANN)

Source: Tan et al., 2019

Bagging (Bootstrap Aggregating)

- Bootstrap sampling: sampling with replacement

Original Data	1	2	3	4	5	6	7	8	9	10
Bagging (Round 1)	7	8	10	8	2	5	10	10	5	9
Bagging (Round 2)	1	4	9	1	2	3	2	7	3	2
Bagging (Round 3)	1	8	5	10	5	5	9	6	3	7

- Build classifier on each bootstrap sample.

- Probability of a training instance being selected in a bootstrap sample is:
  - $1 - (1 - 1/n)^n$ (n: number of training instances)
  - ~0.632 when n is large.

Source: Tan et al., 2019

Bagging Algorithm

Algorithm 4.5 Bagging algorithm.

1: Let k be the number of bootstrap samples.
2: **for** $i = 1$ to k **do**
3: Create a bootstrap sample of size N, D_i.
4: Train a base classifier C_i on the bootstrap sample D_i.
5: **end for**
6: $C^*(x) = \underset{y}{\text{argmax}} \sum_i \delta\big(C_i(x) = y\big)$.

$\{\delta(\cdot) = 1$ if its argument is true and 0 otherwise.$\}$

Source: Tan et al., 2019

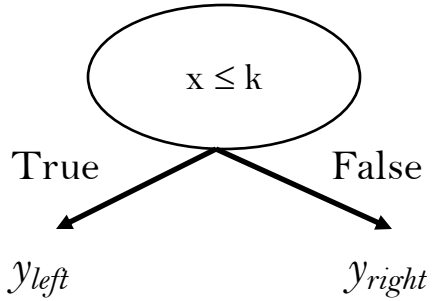
Bagging Example

- Consider 1-dimensional data set:

Original Data:

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
y	1	1	1	-1	-1	-1	-1	1	1	1

- Classifier is a decision stump (decision tree of size 1)
  - Decision rule: $x \leq k$ versus $x > k$
  - Split point k is chosen based on entropy



Source: Tan et al., 2019

Bagging Example

Bagging Round 1:

x	0.1	0.2	0.2	0.3	0.4	0.4	0.5	0.6	0.9	0.9
y	1	1	1	1	-1	-1	-1	-1	1	1

x <= 0.35 ➔ y = 1
x > 0.35 ➔ y = -1

Bagging Round 2:

x	0.1	0.2	0.3	0.4	0.5	0.5	0.9	1	1	1
y	1	1	1	-1	-1	-1	1	1	1	1

x <= 0.7 ➔ y = 1
x > 0.7 ➔ y = 1

Bagging Round 3:

x	0.1	0.2	0.3	0.4	0.4	0.5	0.7	0.7	0.8	0.9
y	1	1	1	-1	-1	-1	-1	-1	1	1

x <= 0.35 ➔ y = 1
x > 0.35 ➔ y = -1

Bagging Round 4:

x	0.1	0.1	0.2	0.4	0.4	0.5	0.5	0.7	0.8	0.9
y	1	1	1	-1	-1	-1	-1	-1	1	1

x <= 0.3 ➔ y = 1
x > 0.3 ➔ y = -1

Bagging Round 5:

x	0.1	0.1	0.2	0.5	0.6	0.6	0.6	1	1	1
y	1	1	1	-1	-1	-1	-1	1	1	1

x <= 0.35 ➔ y = 1
x > 0.35 ➔ y = -1

Source: Tan et al., 2019

Bagging Example

Bagging Round 6:

x	0.2	0.4	0.5	0.6	0.7	0.7	0.7	0.8	0.9	1
y	1	-1	-1	-1	-1	-1	-1	1	1	1

x <= 0.75 ➔ y = -1
x > 0.75 ➔ y = 1

Bagging Round 7:

x	0.1	0.4	0.4	0.6	0.7	0.8	0.9	0.9	0.9	1
y	1	-1	-1	-1	-1	1	1	1	1	1

x <= 0.75 ➔ y = -1
x > 0.75 ➔ y = 1

Bagging Round 8:

x	0.1	0.2	0.5	0.5	0.5	0.7	0.7	0.8	0.9	1
y	1	1	-1	-1	-1	-1	-1	1	1	1

x <= 0.75 ➔ y = -1
x > 0.75 ➔ y = 1

Bagging Round 9:

x	0.1	0.3	0.4	0.4	0.6	0.7	0.7	0.8	1	1
y	1	1	-1	-1	-1	-1	-1	1	1	1

x <= 0.75 ➔ y = -1
x > 0.75 ➔ y = 1

Bagging Round 10:

x	0.1	0.1	0.1	0.1	0.3	0.3	0.8	0.8	0.9	0.9
y	1	1	1	1	1	1	1	1	1	1

x <= 0.05 ➔ y = 1
x > 0.05 ➔ y = 1

Source: Tan et al., 2019

Bagging Example

- Summary of Trained Decision Stumps:

Round	Split Point	Left Class	Right Class
1	0.35	1	-1
2	0.7	1	1
3	0.35	1	-1
4	0.3	1	-1
5	0.35	1	-1
6	0.75	-1	1
7	0.75	-1	1
8	0.75	-1	1
9	0.75	-1	1
10	0.05	1	1

Source: Tan et al., 2019

Bagging Example

- Use majority vote (sign of sum of predictions) to determine class of ensemble classifier:

Round	x=0.1	x=0.2	x=0.3	x=0.4	x=0.5	x=0.6	x=0.7	x=0.8	x=0.9	x=1.0
1	1	1	1	-1	-1	-1	-1	-1	-1	-1
2	1	1	1	1	1	1	1	1	1	1
3	1	1	1	-1	-1	-1	-1	-1	-1	-1
4	1	1	1	-1	-1	-1	-1	-1	-1	-1
5	1	1	1	-1	-1	-1	-1	-1	-1	-1
6	-1	-1	-1	-1	-1	-1	-1	1	1	1
7	-1	-1	-1	-1	-1	-1	-1	1	1	1
8	-1	-1	-1	-1	-1	-1	-1	1	1	1
9	-1	-1	-1	-1	-1	-1	-1	1	1	1
10	1	1	1	1	1	1	1	1	1	1
Sum	2	2	2	-6	-6	-6	-6	2	2	2
Sign	1	1	1	-1	-1	-1	-1	1	1	1

Predicted Class

- Bagging can also increase the complexity (representation capacity) of simple classifiers such as decision stumps.

Source: Tan et al., 2019

Boosting

- An iterative procedure to adaptively change distribution of training data by focusing more on previously misclassified records.
  - Initially, all N records are assigned equal weights (for being selected for training)
  - Unlike bagging, weights may change at the end of each boosting round

Source: Tan et al., 2019

Boosting

- Records that are wrongly classified will have their weights increased in the next round
- Records that are classified correctly will have their weights decreased in the next round

Original Data	1	2	3	4	5	6	7	8	9	10
Boosting (Round 1)	7	3	2	8	7	9	4	10	6	3
Boosting (Round 2)	5	4	9	4	2	5	1	7	4	2
Boosting (Round 3)	4	4	8	10	4	5	4	6	3	4

- Example 4 is hard to classify.
- Its weight is increased, therefore it is more likely to be chosen again in subsequent rounds.

Source: Tan et al., 2019

AdaBoost

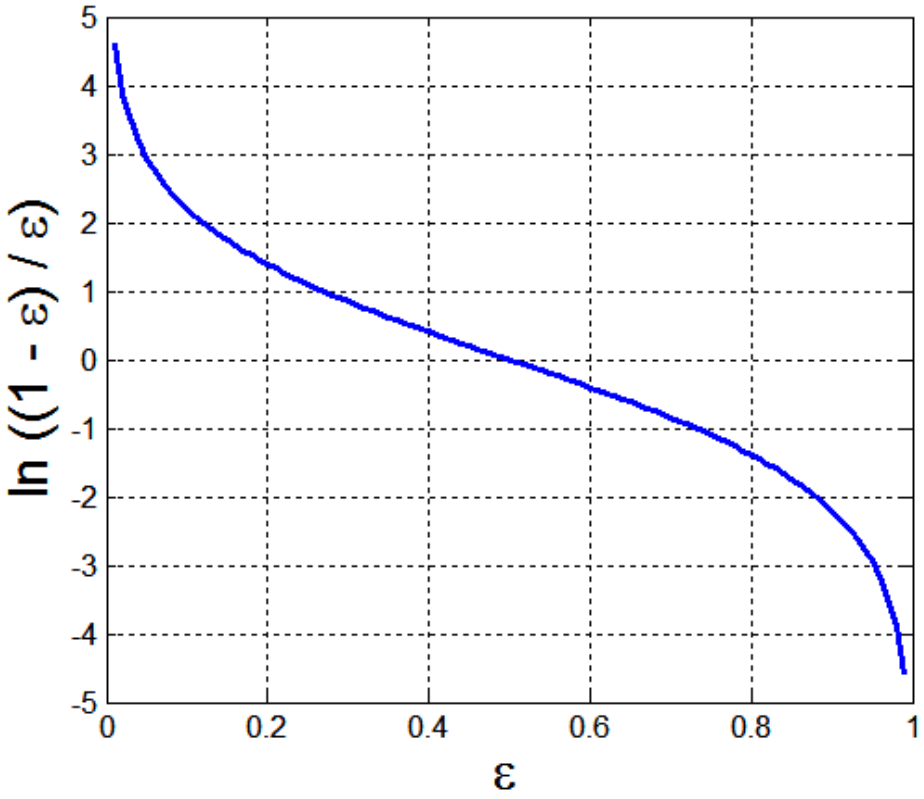
- Base classifiers: $C_1, C_2, \ldots, C_T$

- Error rate of a base classifier:

$$\epsilon_i = \frac{1}{N}\sum_{j=1}^{N} w_j^{(i)}\, \delta(C_i(x_j) \neq y_j)$$

- Importance of a classifier:

$$\alpha_i = \frac{1}{2}\ln\left(\frac{1-\varepsilon_i}{\varepsilon_i}\right)$$



Source: Tan et al., 2019

AdaBoost Algorithm

- Weight update:

$$w_j^{(i+1)} = \frac{w_j^{(i)}}{Z_i} \times \begin{cases} e^{-\alpha_i} & \text{if } C_i(x_j) = y_j \\ e^{\alpha_i} & \text{if } C_i(x_j) \neq y_j \end{cases}$$

Where Z_i is the normalization factor

- If any intermediate rounds produce error rate higher than 50%, the weights are reverted back to 1/n and the resampling procedure is repeated
- Classification:

$$C^*(x) = \arg\max_y \sum_{i=1}^{T} \alpha_i \delta(C_i(x) = y)$$

Source: Tan et al., 2019

AdaBoost Algorithm

Algorithm 4.6 AdaBoost algorithm.

1: $\mathbf{w} = \{w_j = 1/N \mid j = 1, 2, \ldots, N\}$. {Initialize the weights for all N examples.}
2: Let k be the number of boosting rounds.
3: **for** $i = 1$ to k **do**
4: Create training set D_i by sampling (with replacement) from D according to $\mathbf{w}$.
5: Train a base classifier C_i on D_i.
6: Apply C_i to all examples in the original training set, D.
7: $\epsilon_i = \frac{1}{N}\left[\sum_j w_j \, \delta\big(C_i(x_j) \neq y_j\big)\right]$ {Calculate the weighted error.}
8: **if** $\epsilon_i > 0.5$ **then**
9: $\mathbf{w} = \{w_j = 1/N \mid j = 1, 2, \ldots, N\}$. {Reset the weights for all N examples.}
10: Go back to Step 4.
11: **end if**
12: $\alpha_i = \frac{1}{2} \ln \frac{1-\epsilon_i}{\epsilon_i}$.
13: Update the weight of each example according to Equation 4.103.
14: **end for**
15: $C^*(\mathbf{x}) = \underset{y}{\operatorname{argmax}} \sum_{j=1}^{T} \alpha_j \delta(C_j(\mathbf{x}) = y)\big)$.

Source: Tan et al., 2019

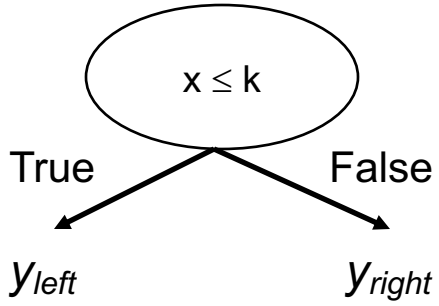
AdaBoost Example

- Consider 1-dimensional data set:

Original Data:

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
y	1	1	1	-1	-1	-1	-1	1	1	1

- Classifier is a decision stump
  - Decision rule: $x \leq k$ versus $x > k$
  - Split point k is chosen based on entropy



Source: Tan et al., 2019

AdaBoost Example

- Training sets for the first 3 boosting rounds:

Boosting Round 1:

x	0.1	0.4	0.5	0.6	0.6	0.7	0.7	0.7	0.8	1
y	1	-1	-1	-1	-1	-1	-1	-1	1	1

Boosting Round 2:

x	0.1	0.1	0.2	0.2	0.2	0.2	0.3	0.3	0.3	0.3
y	1	1	1	1	1	1	1	1	1	1

Boosting Round 3:

x	0.2	0.2	0.4	0.4	0.4	0.4	0.5	0.6	0.6	0.7
y	1	1	-1	-1	-1	-1	-1	-1	-1	-1

- Summary:

Round	Split Point	Left Class	Right Class	alpha
1	0.75	-1	1	1.738
2	0.05	1	1	2.7784
3	0.3	1	-1	4.1195

Source: Tan et al., 2019

AdaBoost Example

- Weights

Round	x=0.1	x=0.2	x=0.3	x=0.4	x=0.5	x=0.6	x=0.7	x=0.8	x=0.9	x=1.0
1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
2	0.311	0.311	0.311	0.01	0.01	0.01	0.01	0.01	0.01	0.01
3	0.029	0.029	0.029	0.228	0.228	0.228	0.228	0.009	0.009	0.009

- Classification

Round	x=0.1	x=0.2	x=0.3	x=0.4	x=0.5	x=0.6	x=0.7	x=0.8	x=0.9	x=1.0
1	-1	-1	-1	-1	-1	-1	-1	1	1	1
2	1	1	1	1	1	1	1	1	1	1
3	1	1	1	-1	-1	-1	-1	-1	-1	-1
Sum	5.16	5.16	5.16	-3.08	-3.08	-3.08	-3.08	0.397	0.397	0.397
Sign	1	1	1	-1	-1	-1	-1	1	1	1

Predicted Class

Source: Tan et al., 2019

Random Forest Algorithm

- Construct an ensemble of decision trees by manipulating training set as well as features.
  - Use bootstrap sample to train every decision tree (similar to Bagging)
  - Use the following tree induction algorithm:
    - At every internal node of decision tree, randomly sample p attributes for selecting split criterion
    - Repeat this procedure until all leaves are pure (unpruned tree)

Source: Tan et al., 2019

Characteristics of Random Forest

- Base classifiers are unpruned trees and hence are *unstable classifiers*
- Base classifiers are *decorrelated* (due to randomization in training set as well as features)
- Random forests reduce variance of unstable classifiers without negatively impacting the bias
- Selection of hyper-parameter p
  - Small value ensures lack of correlation
  - High value promotes strong base classifiers
  - Common default choices: $\sqrt{d}$, $\log_2(d+1)$

Source: Tan et al., 2019

Gradient Boosting

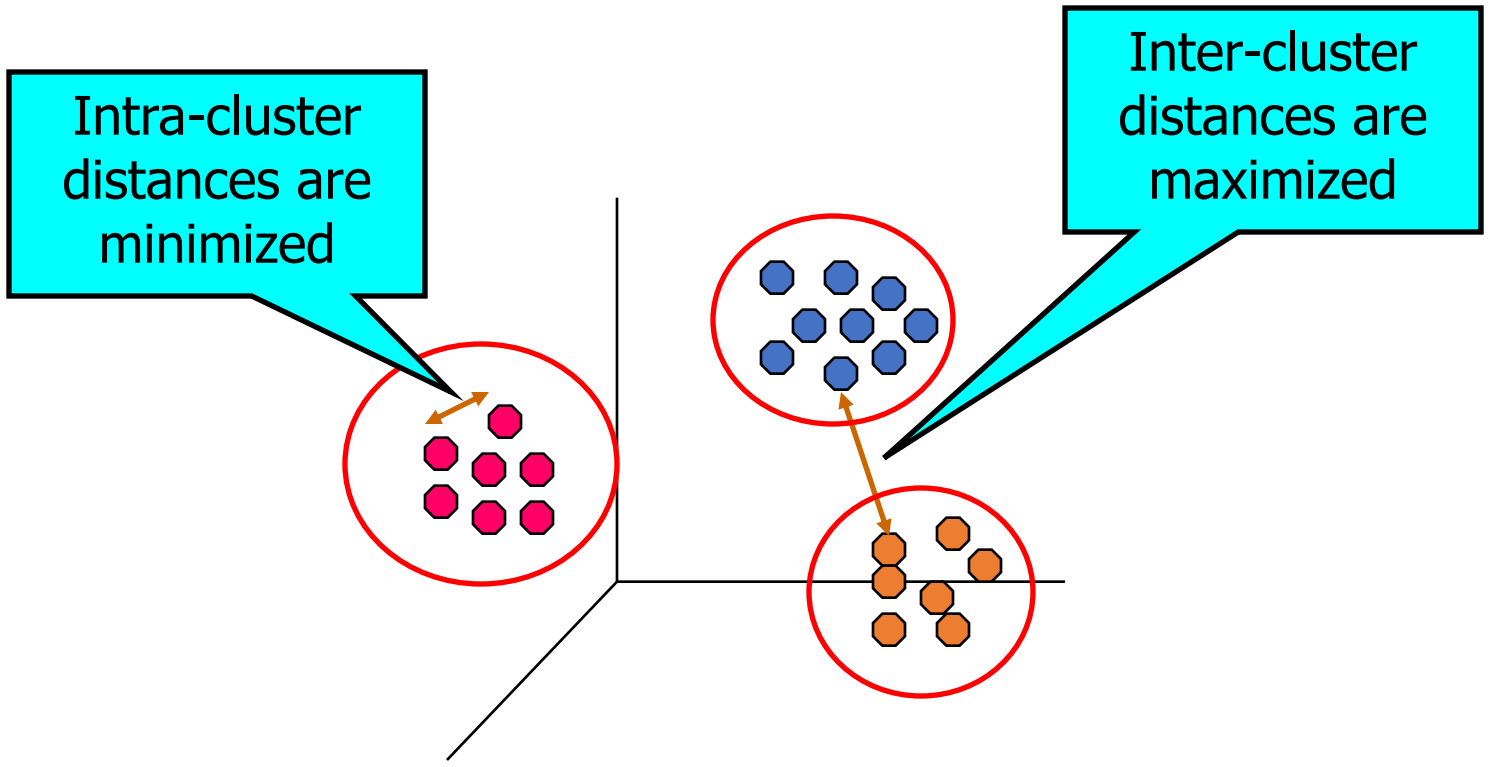
- Constructs a series of models
  - Models can be any predictive model that has a differentiable loss function
  - Commonly, trees are the chosen model
    - XGboost (extreme gradient boosting) is a popular package because of its impressive performance
- Boosting can be viewed as optimizing the loss function by iterative functional gradient descent.
- Implementations of various boosted algorithms are available in Python, R, Matlab, and more.

Source: Tan et al., 2019

3.6 Unsupervised Modeling

What is Cluster Analysis?

- Given a set of objects, place them in groups such that the objects in a group are similar (or related) to one another and different from (or unrelated to) the objects in other groups



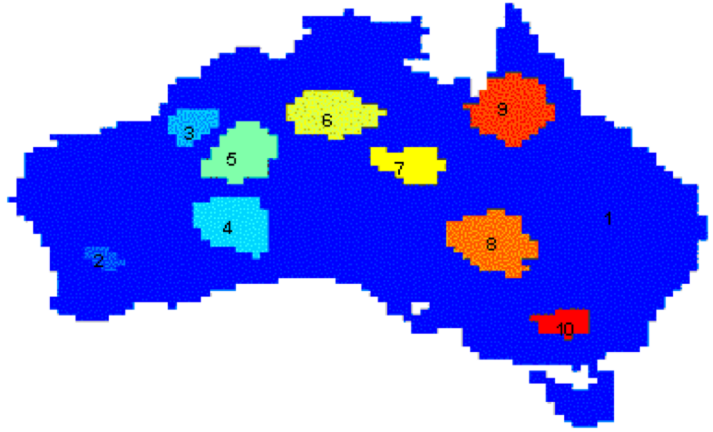
Source: Tan et al., 2019

Applications of Cluster Analysis

- **Understanding**
  - Group related documents for browsing, group genes and proteins that have similar functionality, or group stocks with similar price fluctuations

	Discovered Clusters	*Industry Group*
1	Applied-Matl-DOWN,Bay-Network-Down,3-COM-DOWN, Cabletron-Sys-DOWN,CISCO-DOWN,HP-DOWN, DSC-Comm-DOWN,INTEL-DOWN,LSI-Logic-DOWN, Micron-Tech-DOWN,Texas-Inst-Down,Tellabs-Inc-Down, Natl-Semiconduct-DOWN,Oracl-DOWN,SGI-DOWN, Sun-DOWN	Technology1-DOWN
2	Apple-Comp-DOWN,Autodesk-DOWN,DEC-DOWN, ADV-Micro-Device-DOWN,Andrew-Corp-DOWN, Computer-Assoc-DOWN,Circuit-City-DOWN, Compaq-DOWN, EMC-Corp-DOWN, Gen-Inst-DOWN, Motorola-DOWN,Microsoft-DOWN,Scientific-Atl-DOWN	Technology2-DOWN
3	Fannie-Mae-DOWN,Fed-Home-Loan-DOWN, MBNA-Corp-DOWN,Morgan-Stanley-DOWN	Financial-DOWN
4	Baker-Hughes-UP,Dresser-Inds-UP,Halliburton-HLD-UP, Louisiana-Land-UP,Phillips-Petro-UP,Unocal-UP, Schlumberger-UP	Oil-UP

- **Summarization**
  - Reduce the size of large data sets

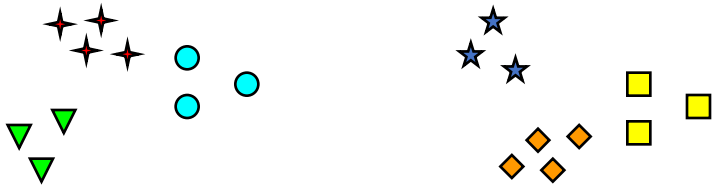


Clustering precipitation in Australia

Source: Tan et al., 2019

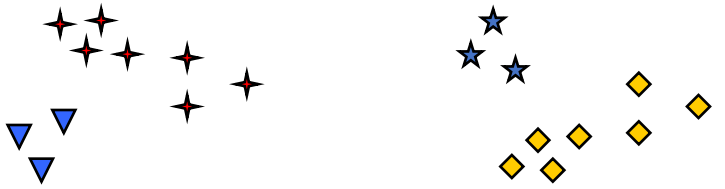
Notion of a Cluster can be Ambiguous

How many clusters?



Six Clusters

Two Clusters



Four Clusters

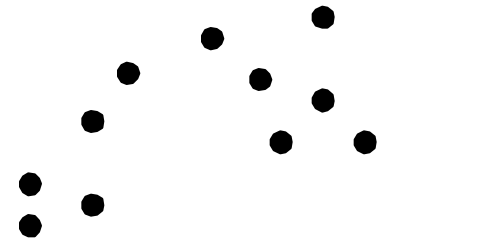
Source: Tan et al., 2019

Types of Clusterings

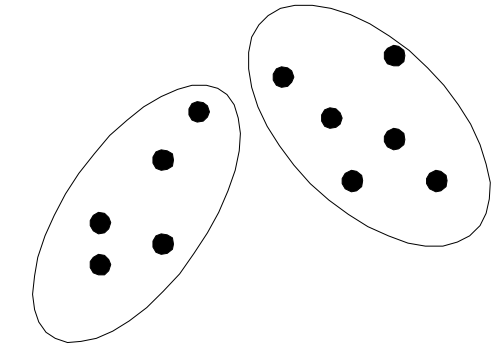
- A clustering is a set of clusters
- Important distinction between hierarchical and partitional sets of clusters
  - Partitional Clustering
    - A division of data objects into non-overlapping subsets (clusters)
  - Hierarchical clustering
    - A set of nested clusters organized as a hierarchical tree

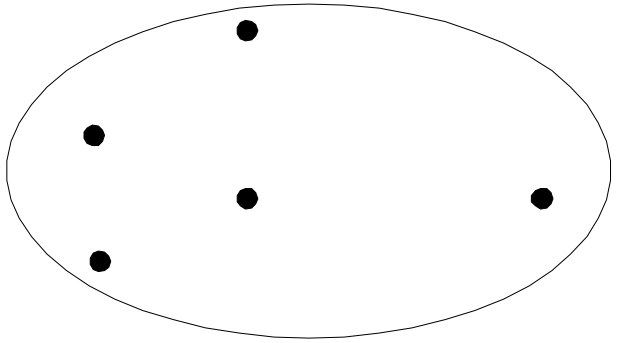
Source: Tan et al., 2019

Partitional Clustering



Original Points

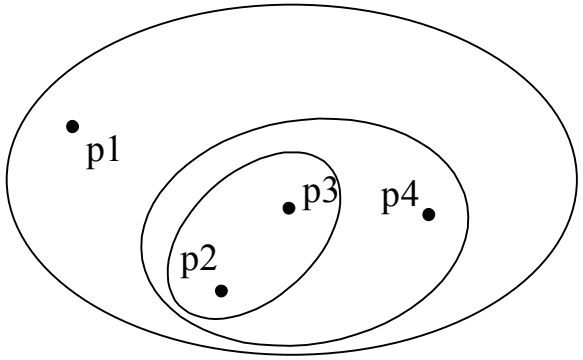




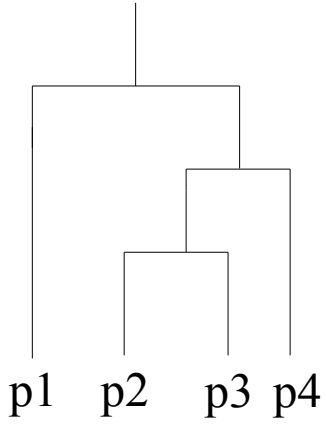
A Partitional Clustering

Source: Tan et al., 2019

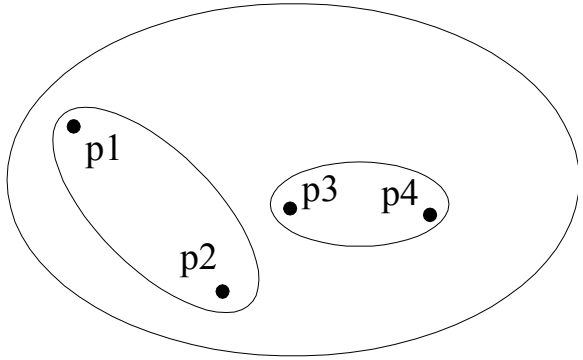
Hierarchical Clustering



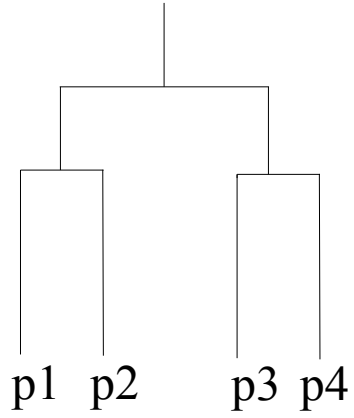
Traditional Hierarchical Clustering



Traditional Dendrogram



Non-traditional Hierarchical Clustering



Non-traditional Dendrogram

Source: Tan et al., 2019

Other Distinctions Between Sets of Clusters

- Exclusive versus non-exclusive
  - In non-exclusive clusterings, points may belong to multiple clusters.
    - Can belong to multiple classes or could be 'border' points
    - Fuzzy clustering (one type of non-exclusive)
    - In fuzzy clustering, a point belongs to every cluster with some weight between 0 and 1
    - Weights must sum to 1
    - Probabilistic clustering has similar characteristics
- Partial versus complete
  - In some cases, we only want to cluster some of the data

Source: Tan et al., 2019

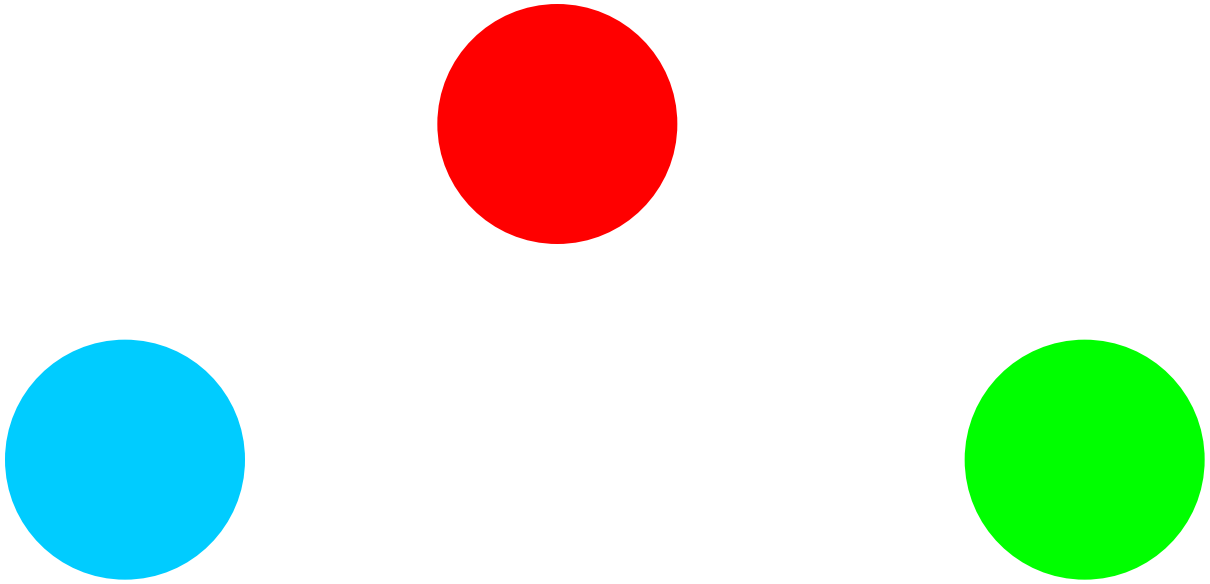
Types of Clusters

- Well-separated clusters
- Prototype-based clusters
- Contiguity-based clusters
- Density-based clusters
- Described by an Objective Function

Source: Tan et al., 2019

Types of Clusters: Well-Separated

- Well-Separated Clusters:
  - A cluster is a set of points such that any point in a cluster is closer (or more similar) to every other point in the cluster than to any point not in the cluster.



3 well-separated clusters

Source: Tan et al., 2019

Types of Clusters: Prototype-Based

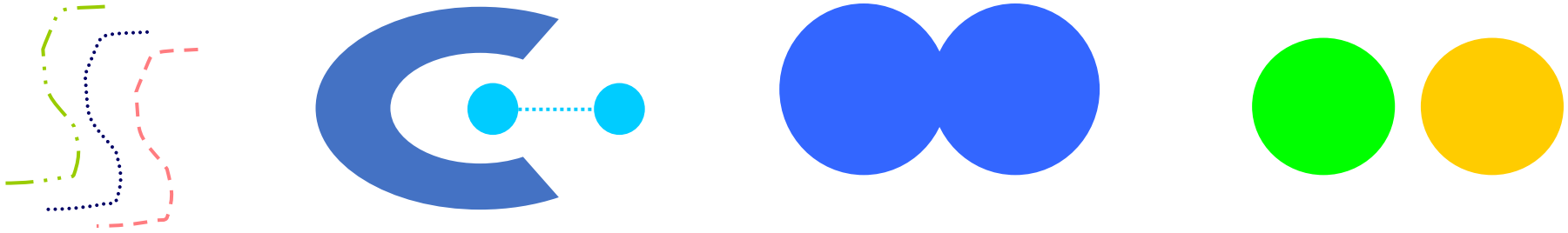
- Prototype-based
  - A cluster is a set of objects such that an object in a cluster is closer (more similar) to the prototype or "center" of a cluster, than to the center of any other cluster
  - The center of a cluster is often a centroid, the average of all the points in the cluster, or a medoid, the most "representative" point of a cluster

4 center-based clusters

Source: Tan et al., 2019

Types of Clusters: Contiguity-Based

- Contiguous Cluster (Nearest neighbor or Transitive)
  - A cluster is a set of points such that a point in a cluster is closer (or more similar) to one or more other points in the cluster than to any point not in the cluster.

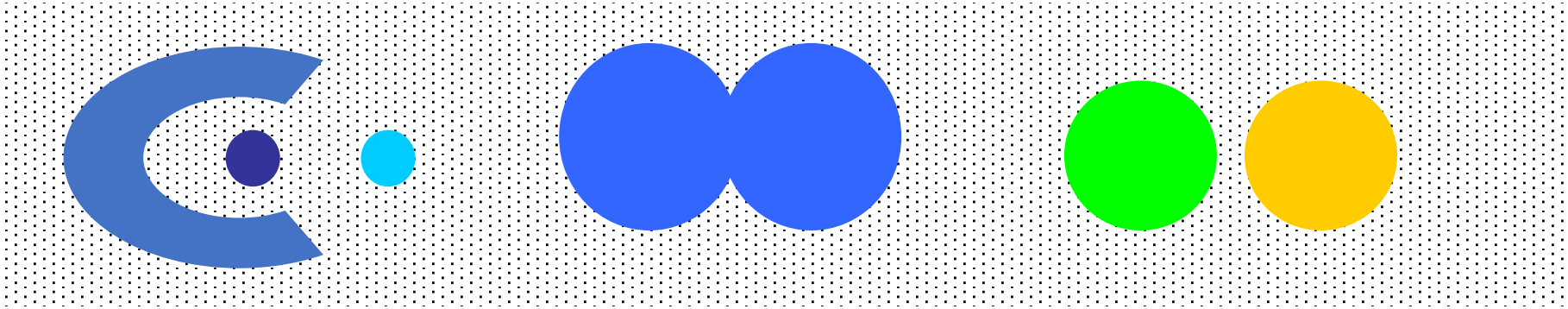


8 contiguous clusters

Source: Tan et al., 2019

Types of Clusters: Density-Based

- Density-based
  - A cluster is a dense region of points, which is separated by low-density regions, from other regions of high density.
  - Used when the clusters are irregular or intertwined, and when noise and outliers are present.



6 density-based clusters

Source: Tan et al., 2019

Types of Clusters: Objective Function

- Clusters Defined by an Objective Function
  - Finds clusters that minimize or maximize an objective function.
  - Enumerate all possible ways of dividing the points into clusters and evaluate the "goodness" of each potential set of clusters by using the given objective function. (NP Hard)
  - Can have global or local objectives.
    - Hierarchical clustering algorithms typically have local objectives
    - Partitional algorithms typically have global objectives
  - A variation of the global objective function approach is to fit the data to a parameterized model.
    - Parameters for the model are determined from the data.
    - Mixture models assume that the data is a 'mixture' of a number of statistical distributions.

Source: Tan et al., 2019

Characteristics of the Input Data Are Important

- Type of proximity or density measure
  - Central to clustering
  - Depends on data and application

- Data characteristics that affect proximity and/or density are
  - Dimensionality
    - Sparseness
  - Attribute type
  - Special relationships in the data
    - For example, autocorrelation
  - Distribution of the data

- Noise and Outliers
  - Often interfere with the operation of the clustering algorithm
- Clusters of differing sizes, densities, and shapes

Source: Tan et al., 2019

Clustering Algorithms

- K-means and its variants
- Hierarchical clustering
- Density-based clustering

Source: Tan et al., 2019

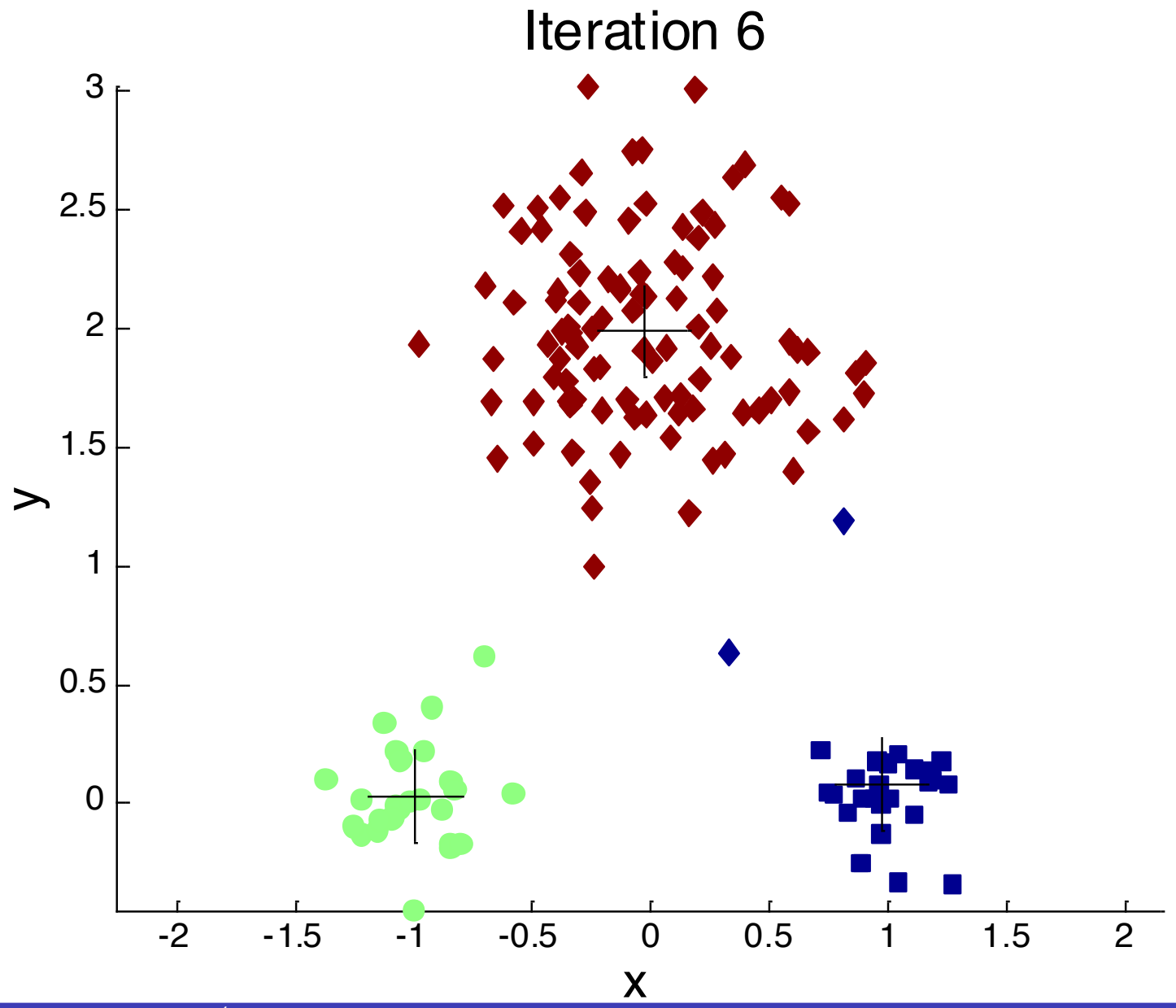
K-means Clustering

- Partitional clustering approach
- Number of clusters, K, must be specified
- Each cluster is associated with a centroid (center point)
- Each point is assigned to the cluster with the closest centroid
- The basic algorithm is very simple

1: Select K points as the initial centroids.
2: **repeat**
3: Form K clusters by assigning all points to the closest centroid.
4: Recompute the centroid of each cluster.
5: **until** The centroids don't change

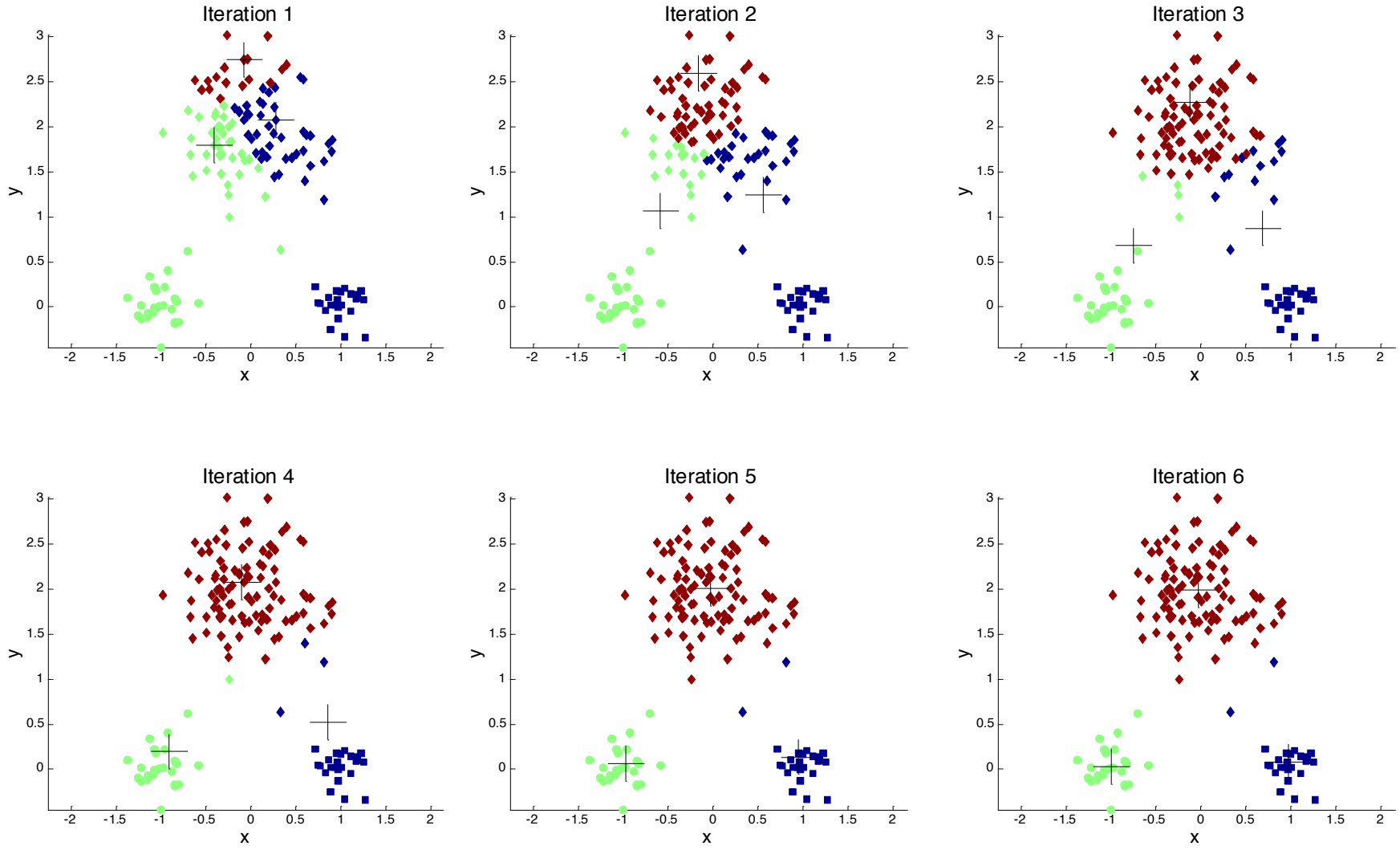
Source: Tan et al., 2019

Example of K-means Clustering



Source: Tan et al., 2019

Example of K-means Clustering



Source: Tan et al., 2019

K-means Clustering – Details

- Simple iterative algorithm.
  - Choose initial centroids
  - Repeat {assign each point to a nearest centroid; re-compute cluster centroids}
  - Until centroids stop changing.
- Initial centroids are often chosen randomly.
  - Clusters produced can vary from one run to another
- The centroid is (typically) the mean of the points in the cluster, but other definitions are possible (see Table 7.2).
- K-means will converge for common proximity measures with appropriately defined centroid (see Table 7.2)
- Most of the convergence happens in the first few iterations.
  - Often the stopping condition is changed to 'Until relatively few points change clusters'
- Complexity is O(n * K * I * d)
  - n = number of points, K = number of clusters,
    I = number of iterations, d = number of attributes

Source: Tan et al., 2019

K-means Objective Function

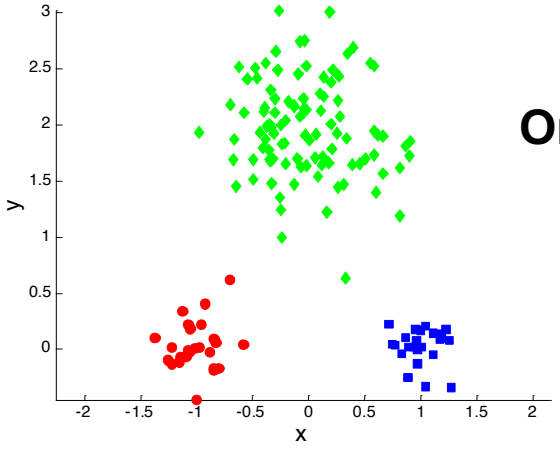
- A common objective function (used with Euclidean distance measure) is Sum of Squared Error (SSE)
  - For each point, the error is the distance to the nearest cluster center
  - To get SSE, we square these errors and sum them.

$$SSE = \sum_{i=1}^{K} \sum_{x \in C_i} dist^2(m_i, x)$$

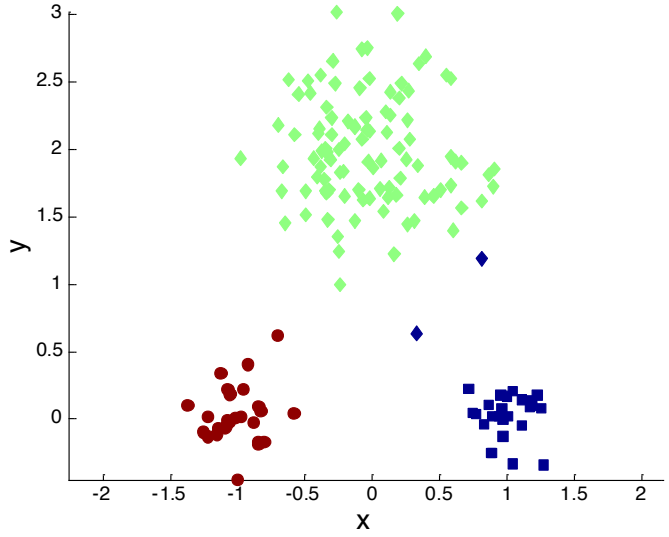
  - x is a data point in cluster C_i and m_i is the centroid (mean) for cluster C_i
  - SSE improves in each iteration of K-means until it reaches a local or global minima.

Source: Tan et al., 2019

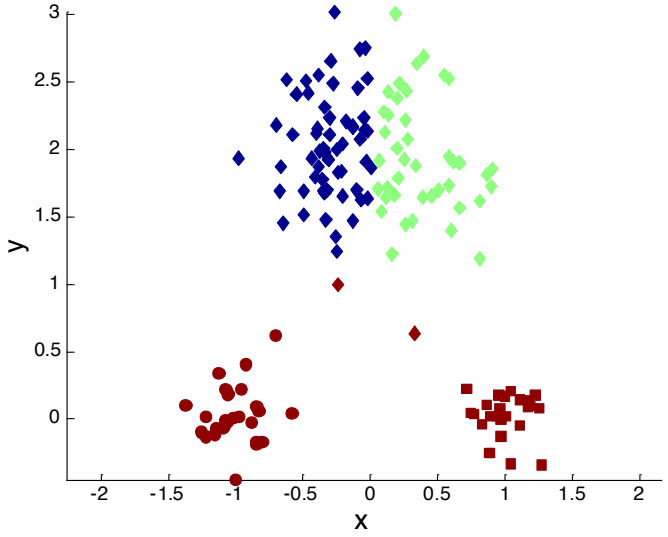
Two different K-means Clusterings



Original Points



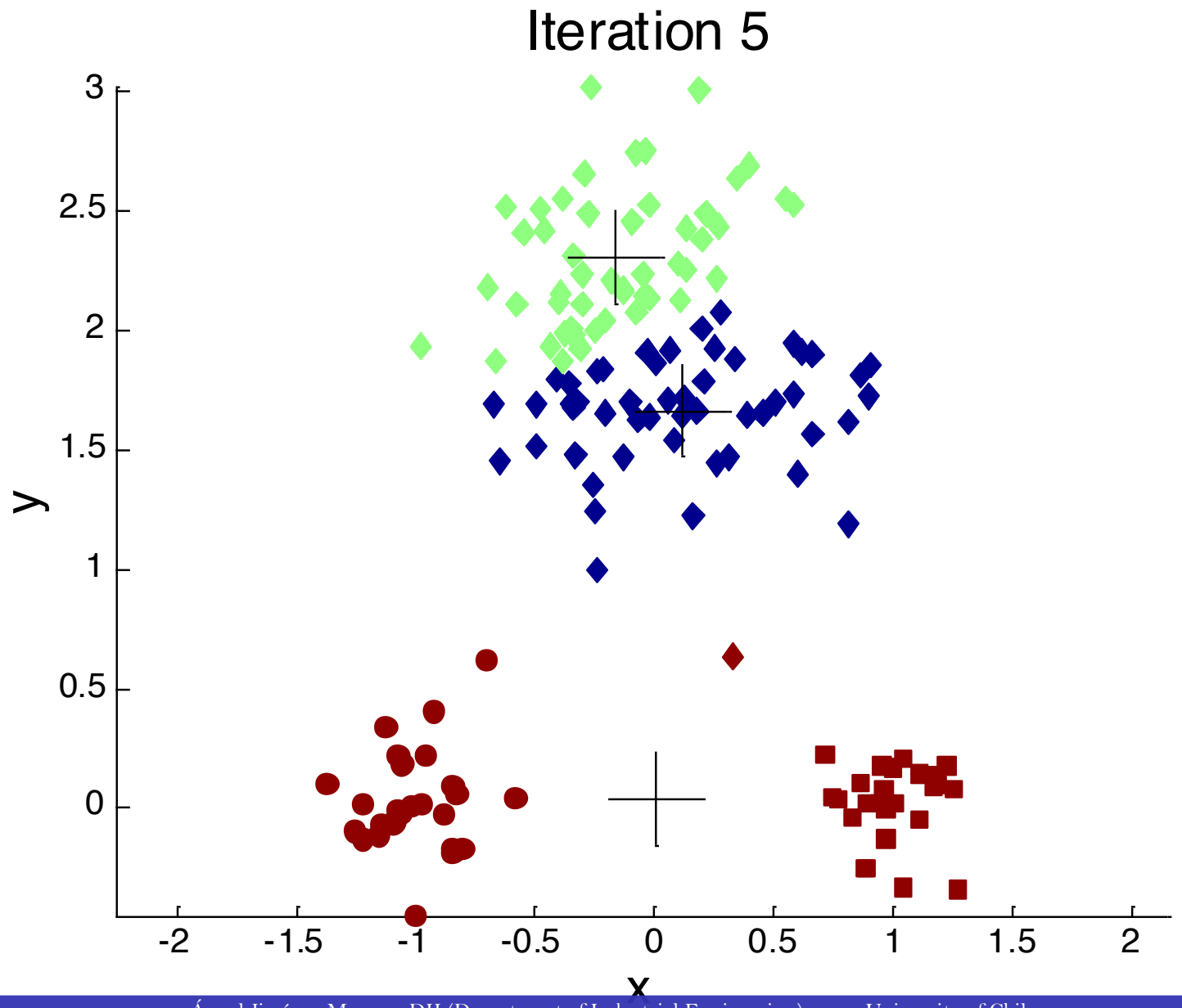
Optimal Clustering



Sub-optimal Clustering

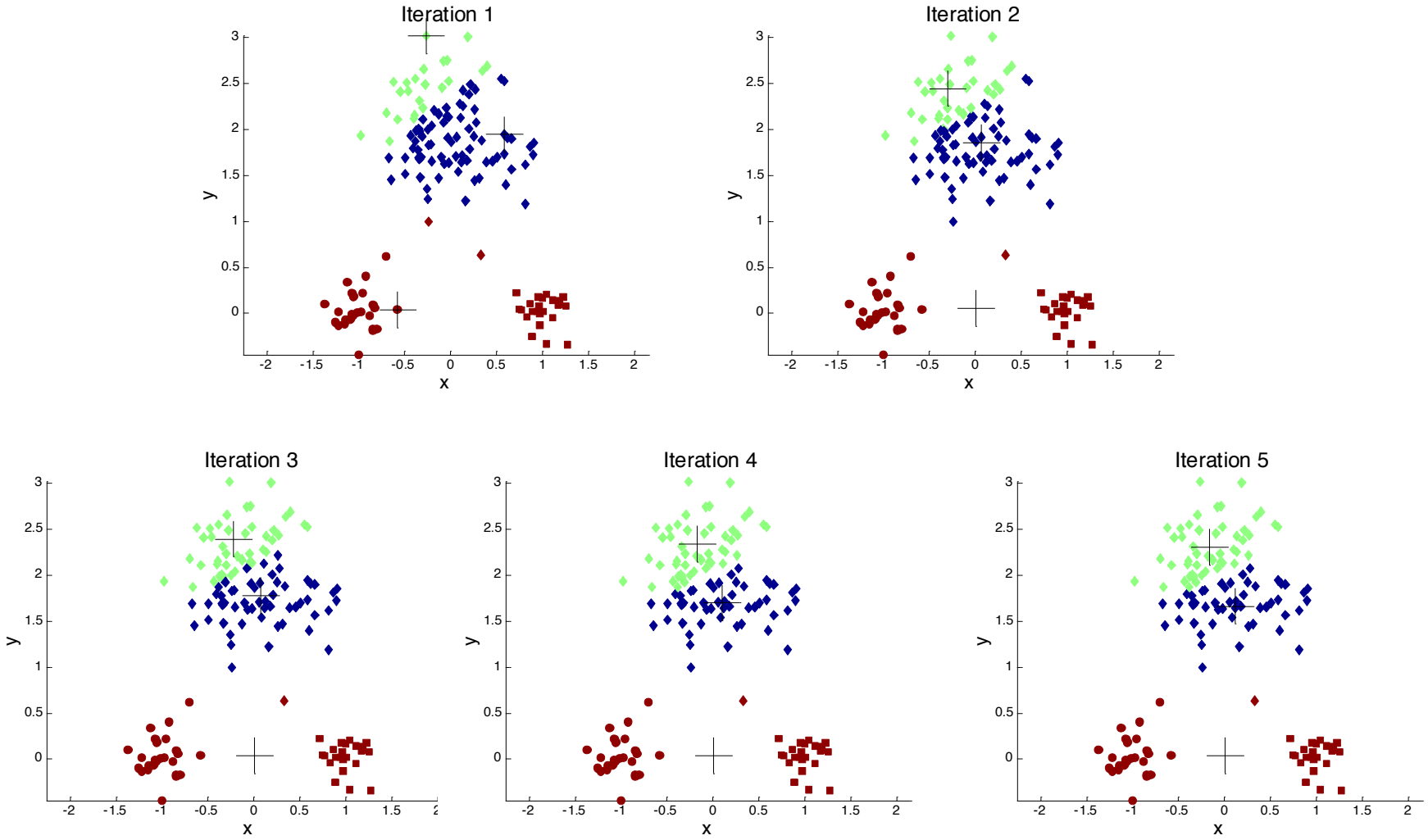
Source: Tan et al., 2019

Importance of Choosing Initial Centroids …



Source: Tan et al., 2019

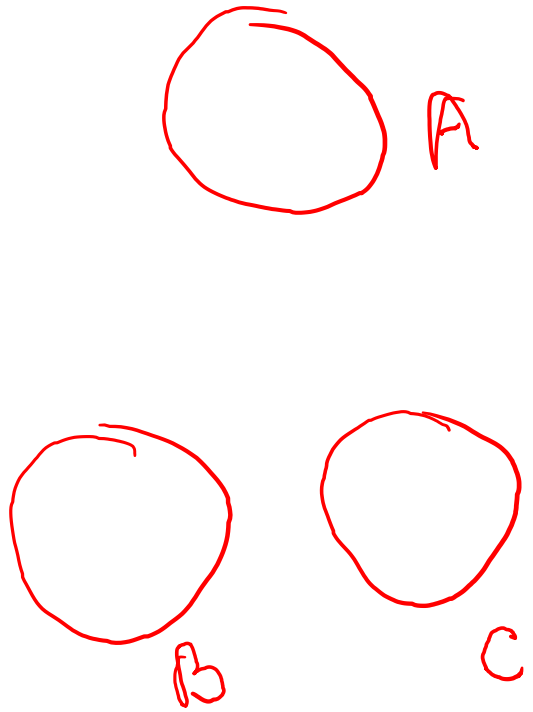
Importance of Choosing Initial Centroids …



Source: Tan et al., 2019

Importance of Choosing Intial Centroids

- Depending on the choice of initial centroids, B and C may get merged or remain separate.



Source: Tan et al., 2019

Problems with Selecting Initial Points

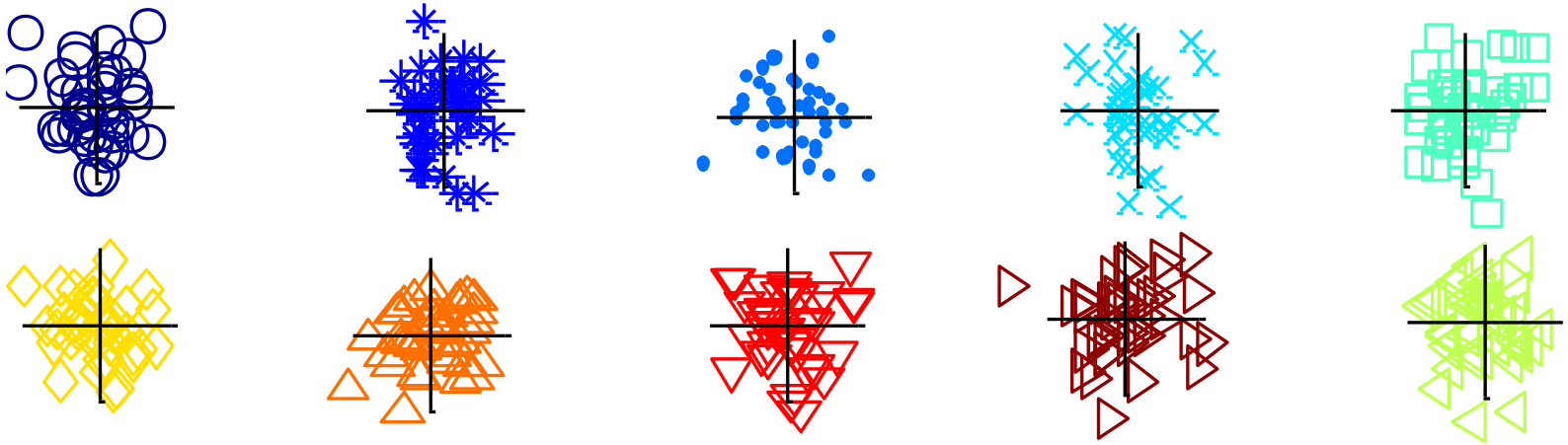
- If there are K 'real' clusters then the chance of selecting one centroid from each cluster is small.
  - Chance is relatively small when K is large
  - If clusters are the same size, n, then

$$P = \frac{\text{number of ways to select one centroid from each cluster}}{\text{number of ways to select } K \text{ centroids}} = \frac{K!n^K}{(Kn)^K} = \frac{K!}{K^K}$$

  - For example, if K = 10, then probability = $10!/10^{10} = 0.00036$
  - Sometimes the initial centroids will readjust themselves in 'right' way, and sometimes they don't
  - Consider an example of five pairs of clusters

Source: Tan et al., 2019

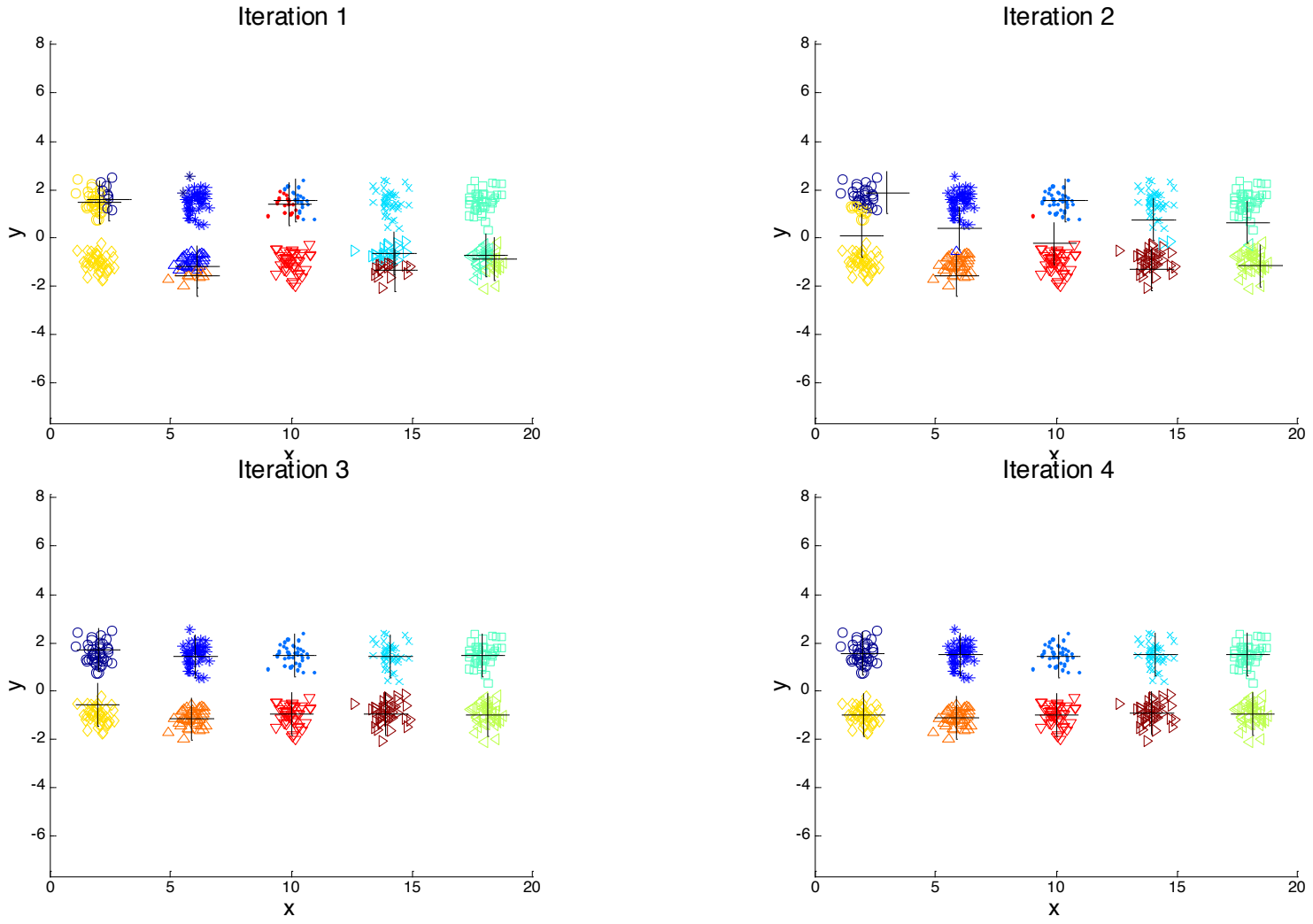
10 Clusters Example



Starting with two initial centroids in one cluster of each pair of clusters

Source: Tan et al., 2019

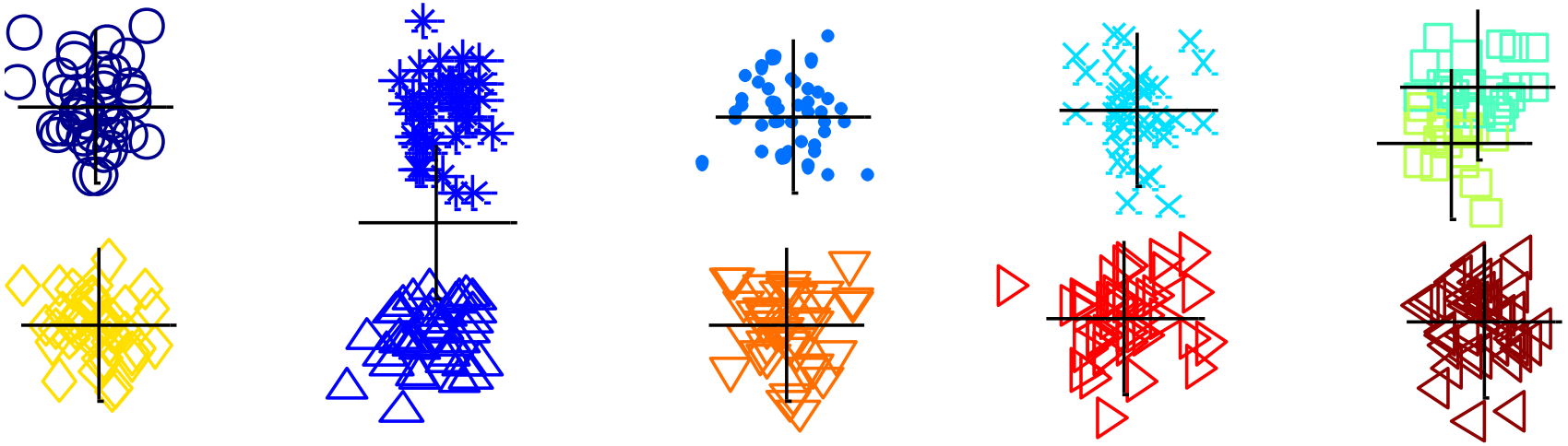
10 Clusters Example



Starting with two initial centroids in one cluster of each pair of clusters

Source: Tan et al., 2019

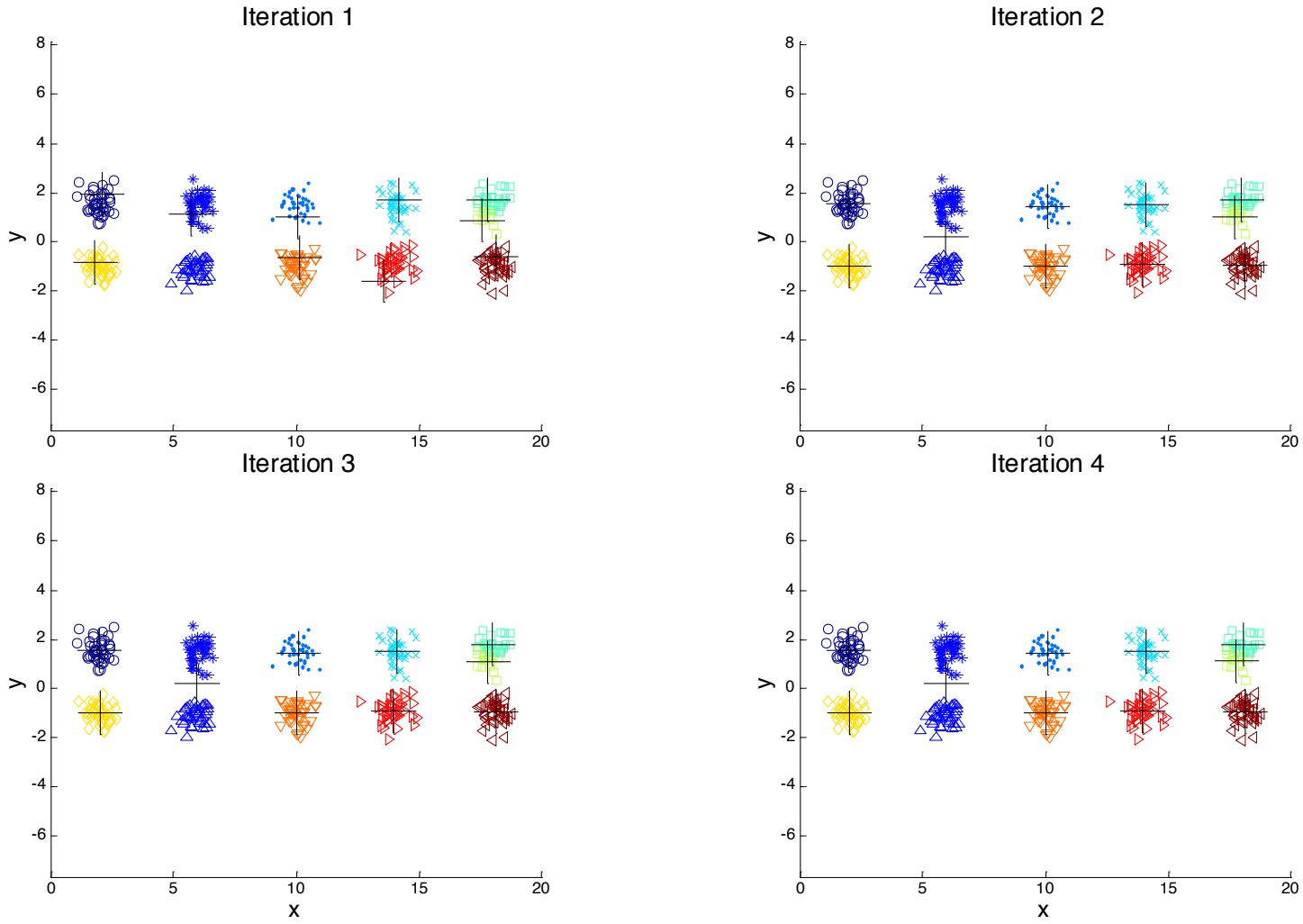
10 Clusters Example



Starting with some pairs of clusters having three initial centroids, while other have only one.

Source: Tan et al., 2019

10 Clusters Example



Starting with some pairs of clusters having three initial centroids, while other have only one.

Source: Tan et al., 2019

Solutions to Initial Centroids Problem

- Multiple runs
  - Helps, but probability is not on your side
- Use some strategy to select the k initial centroids and then select among these initial centroids
  - Select most widely separated
    - K-means++ is a robust way of doing this selection
  - Use hierarchical clustering to determine initial centroids
- Bisecting K-means
  - Not as susceptible to initialization issues

Source: Tan et al., 2019

K-means++

- This approach can be slower than random initialization, but very consistently produces better results in terms of SSE
  - The k-means++ algorithm guarantees an approximation ratio O(log k) in expectation, where k is the number of centers
- To select a set of initial centroids, *C*, perform the following:

1. Select an initial point at random to be the first centroid
2. For k – 1 steps
3. For each of the N points, x_i, $1 \leq i \leq N$, find the minimum squared distance to the currently selected centroids, C_1, …, C_j, $1 \leq j < k$, i.e., $\min_j d^2(C_j, x_i)$
4. Randomly select a new centroid by choosing a point with probability proportional to $\frac{\min_j d^2(C_j, x_i)}{\sum_i \min_j d^2(C_j, x_i)}$ is
5. End For

Source: Tan et al., 2019

Bisecting K-means

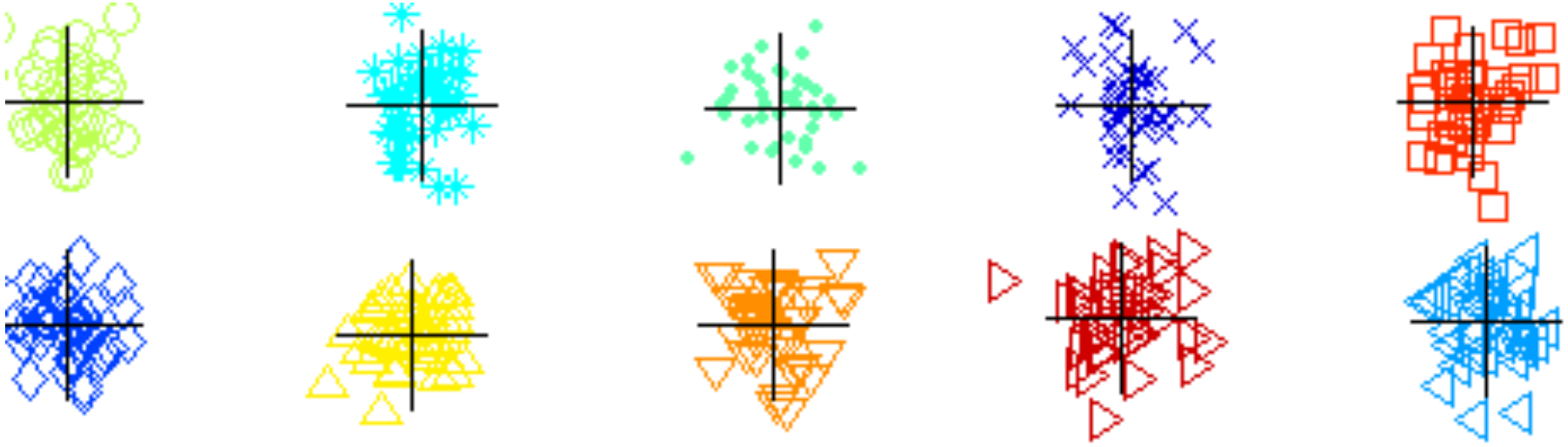
- Bisecting K-means algorithm
  - Variant of K-means that can produce a partitional or a hierarchical clustering

1: Initialize the list of clusters to contain the cluster containing all points.
2: **repeat**
3: Select a cluster from the list of clusters
4: **for** $i = 1$ to *number_of_iterations* **do**
5: Bisect the selected cluster using basic K-means
6: **end for**
7: Add the two clusters from the bisection with the lowest SSE to the list of clusters.
8: **until** Until the list of clusters contains K clusters

CLUTO: http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview

Source: Tan et al., 2019

Bisecting K-means Example



Source: Tan et al., 2019

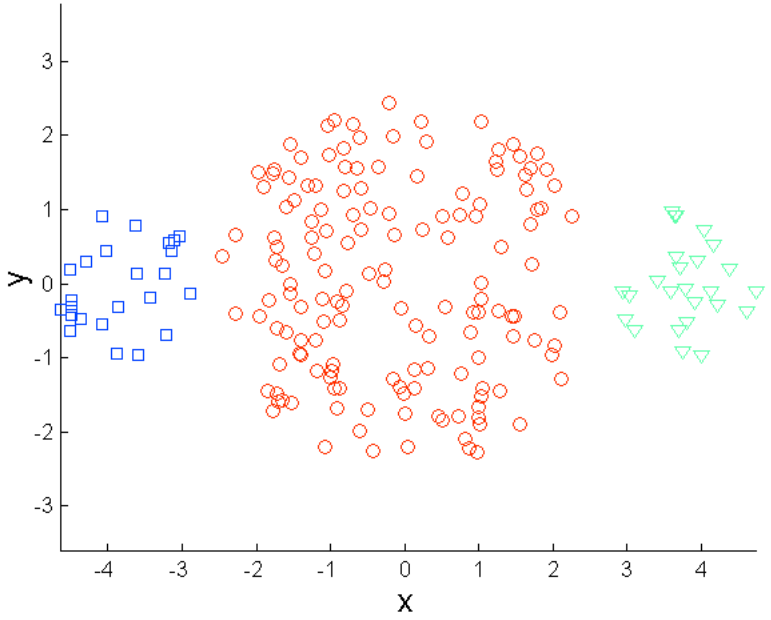
Limitations of K-means

- K-means has problems when clusters are of differing
  - Sizes
  - Densities
  - Non-globular shapes

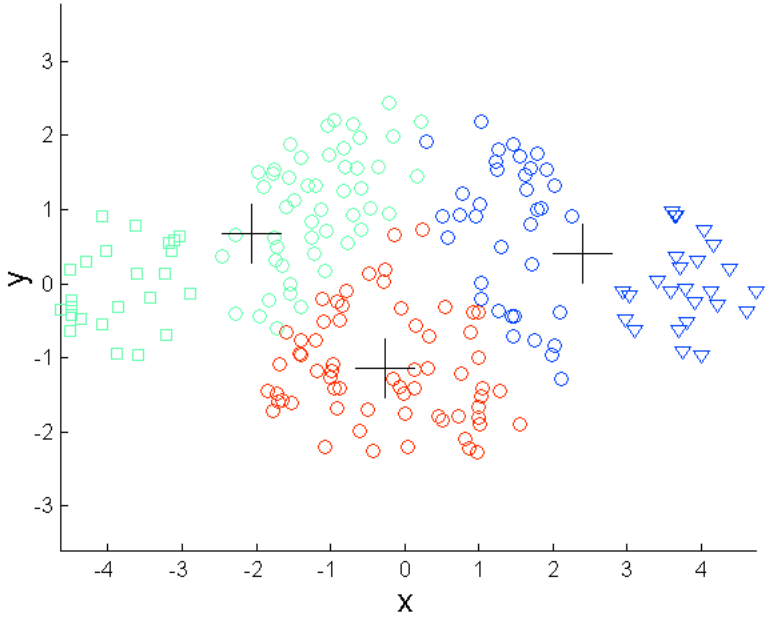
- K-means has problems when the data contains outliers.
  - One possible solution is to remove outliers before clustering

Source: Tan et al., 2019

Limitations of K-means: Differing Sizes



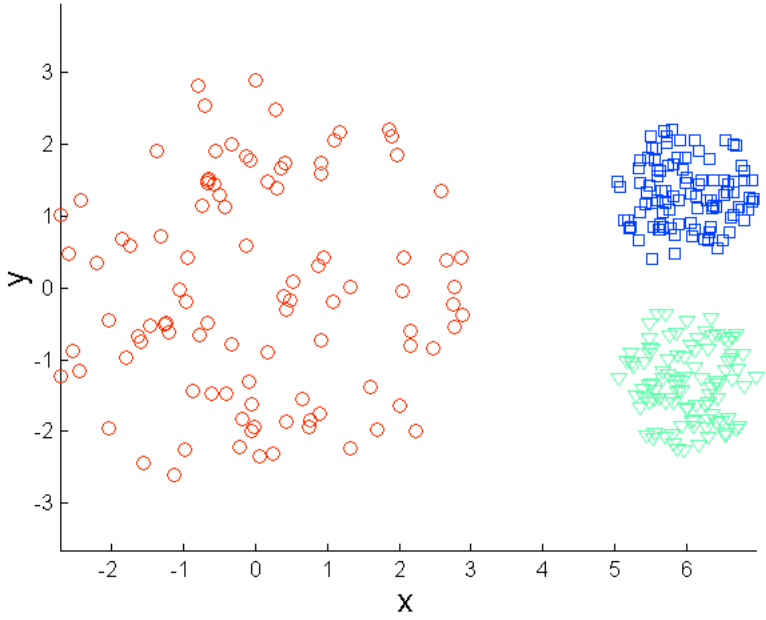
Original Points



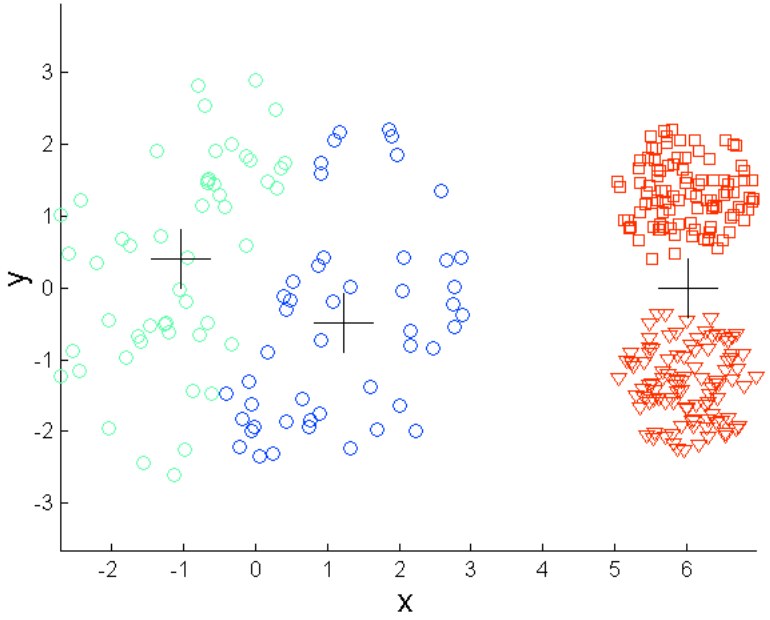
K-means (3 Clusters)

Source: Tan et al., 2019

Limitations of K-means: Differing Density



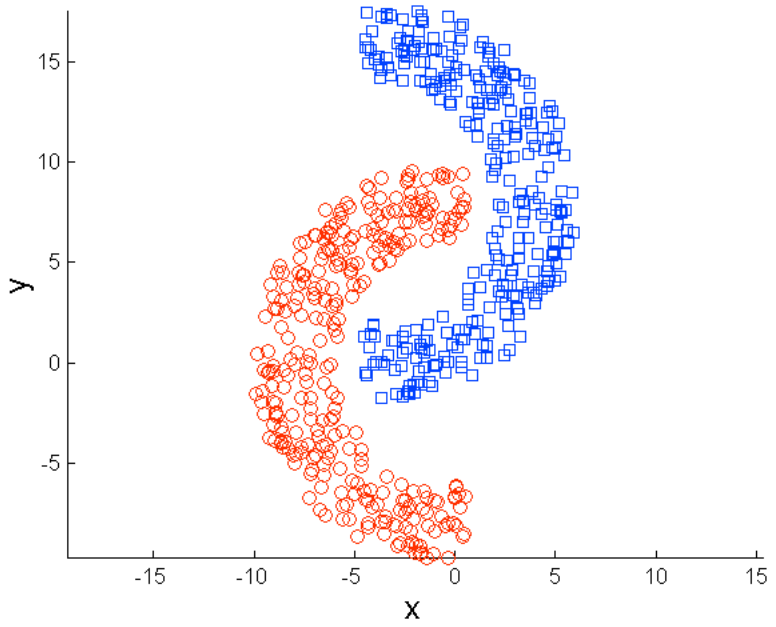
Original Points



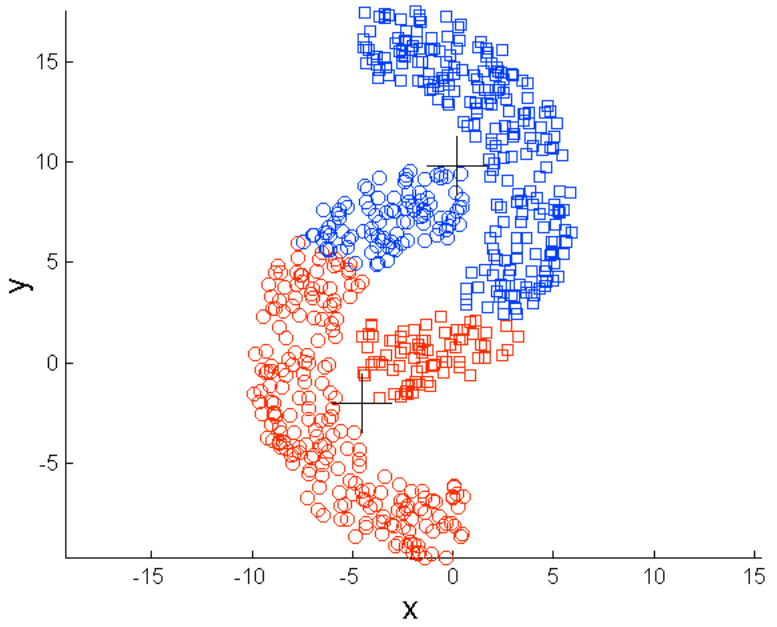
K-means (3 Clusters)

Source: Tan et al., 2019

Limitations of K-means: Non-globular Shapes



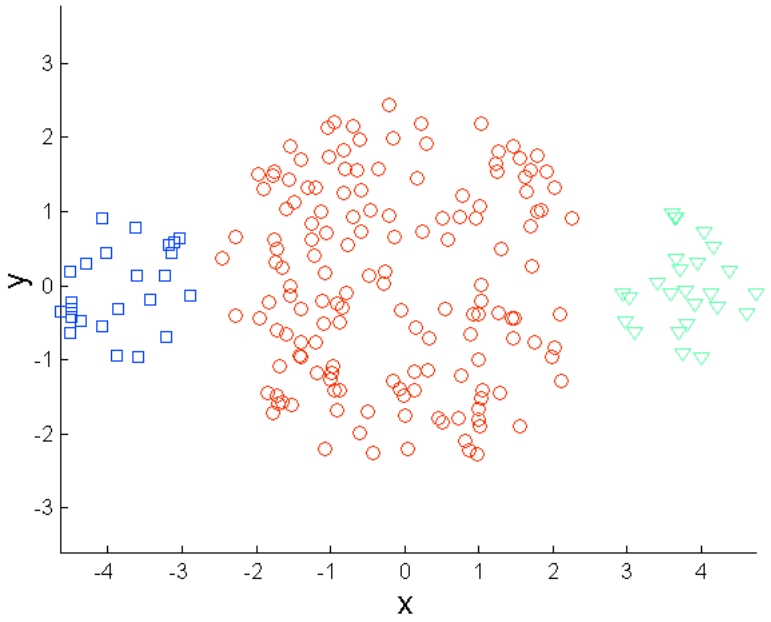
Original Points



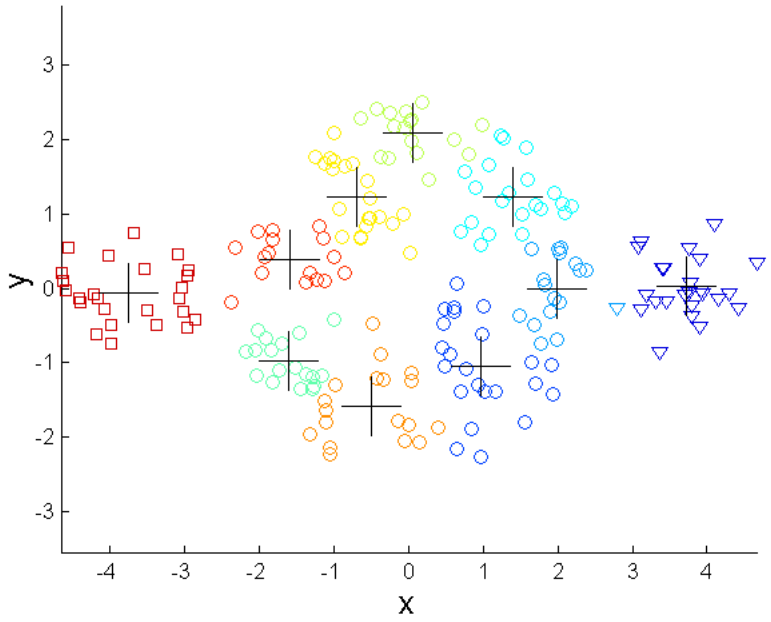
K-means (2 Clusters)

Source: Tan et al., 2019

Overcoming K-means Limitations



Original Points

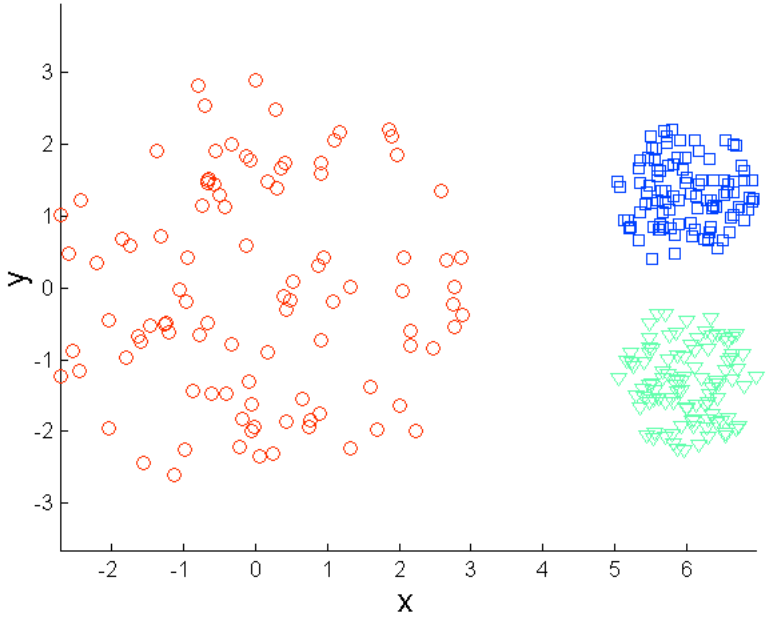


K-means Clusters

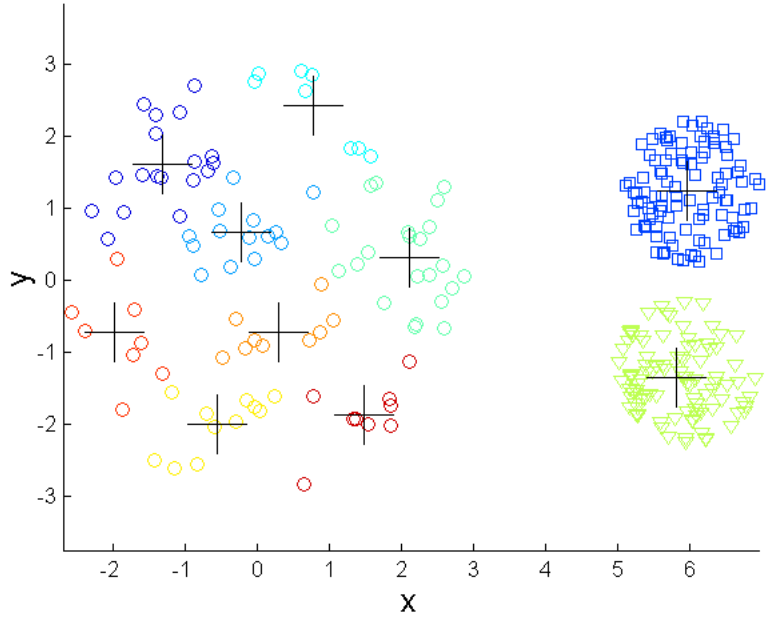
One solution is to find a large number of clusters such that each of them represents a part of a natural cluster. But these small clusters need to be put together in a post-processing step.

Source: Tan et al., 2019

Overcoming K-means Limitations



Original Points

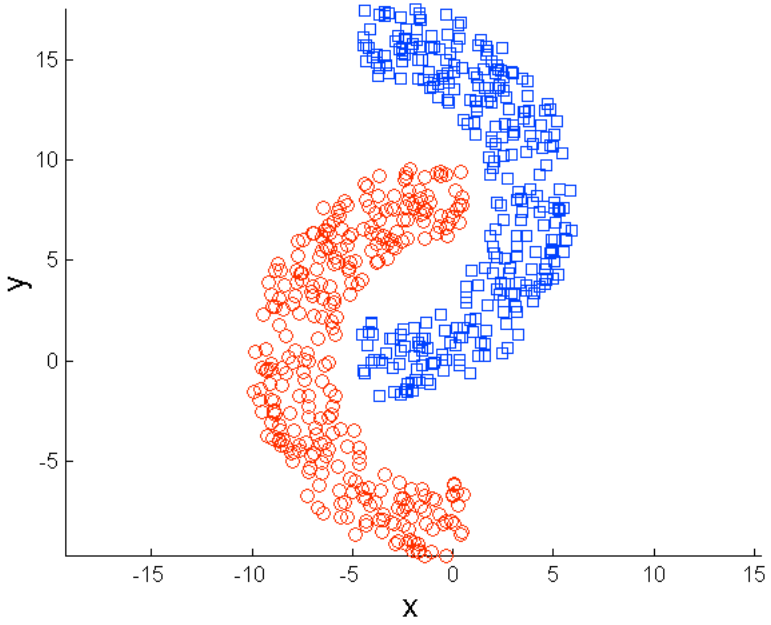


K-means Clusters

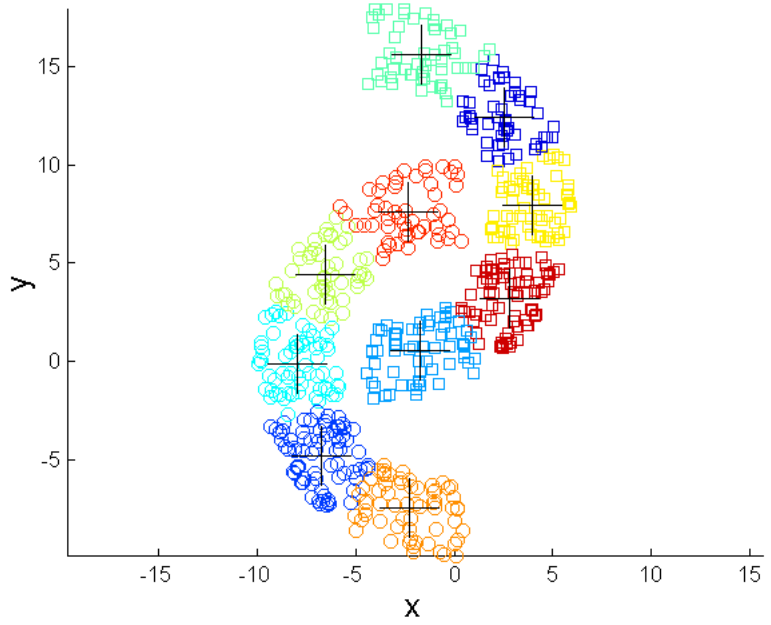
One solution is to find a large number of clusters such that each of them represents a part of a natural cluster. But these small clusters need to be put together in a post-processing step.

Source: Tan et al., 2019

Overcoming K-means Limitations



Original Points



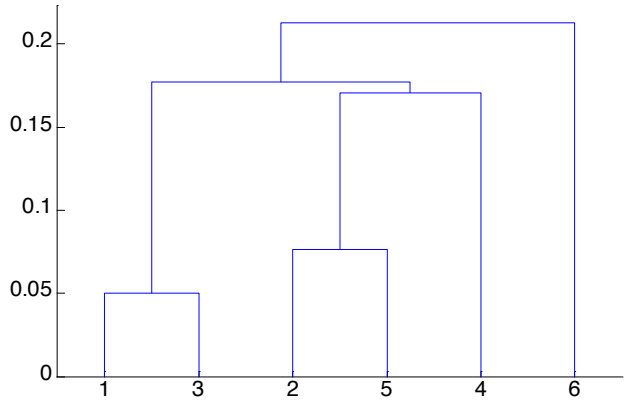
K-means Clusters

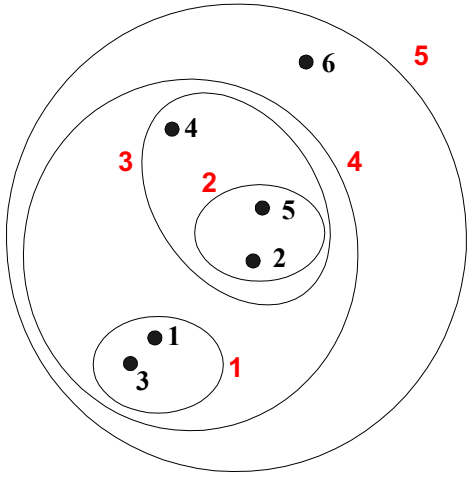
One solution is to find a large number of clusters such that each of them represents a part of a natural cluster. But these small clusters need to be put together in a post-processing step.

Source: Tan et al., 2019

Hierarchical Clustering

- Produces a set of nested clusters organized as a hierarchical tree
- Can be visualized as a dendrogram
  - A tree like diagram that records the sequences of merges or splits





Source: Tan et al., 2019

Strengths of Hierarchical Clustering

- Do not have to assume any particular number of clusters
  - Any desired number of clusters can be obtained by 'cutting' the dendrogram at the proper level

- They may correspond to meaningful taxonomies
  - Example in biological sciences (e.g., animal kingdom, phylogeny reconstruction, …)

Source: Tan et al., 2019

Hierarchical Clustering

- Two main types of hierarchical clustering
  - Agglomerative:
    - Start with the points as individual clusters
    - At each step, merge the closest pair of clusters until only one cluster (or k clusters) left
  - Divisive:
    - Start with one, all-inclusive cluster
    - At each step, split a cluster until each cluster contains an individual point (or there are k clusters)
- Traditional hierarchical algorithms use a similarity or distance matrix
  - Merge or split one cluster at a time

Source: Tan et al., 2019

Agglomerative Clustering Algorithm

- **Key Idea: Successively merge closest clusters**

- Basic algorithm

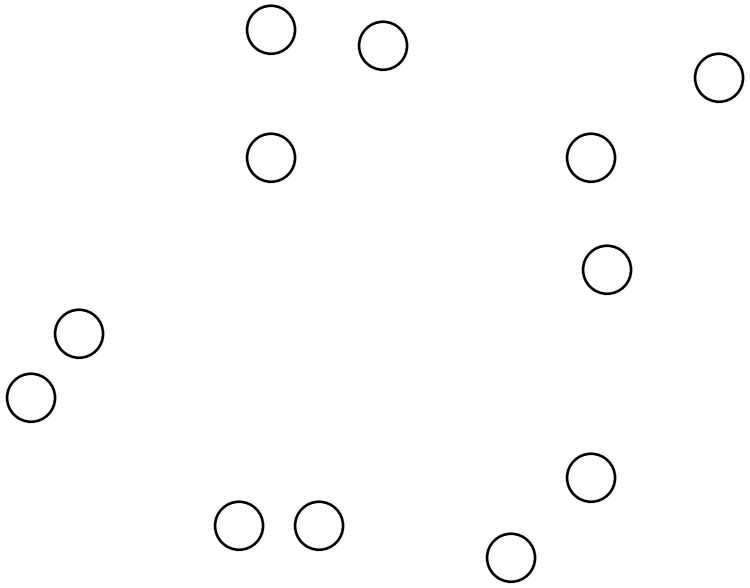
```
1. Compute the proximity matrix
2. Let each data point be a cluster
3. Repeat
4.         Merge the two closest clusters
5.         Update the proximity matrix
6. Until only a single cluster remains
```

- Key operation is the computation of the proximity of two clusters
  - Different approaches to defining the distance between clusters distinguish the different algorithms

Source: Tan et al., 2019

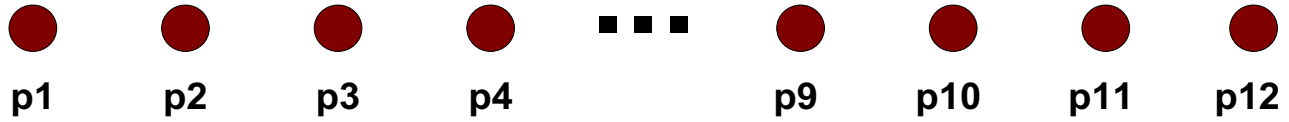
Steps 1 and 2

- Start with clusters of individual points and a proximity matrix



	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

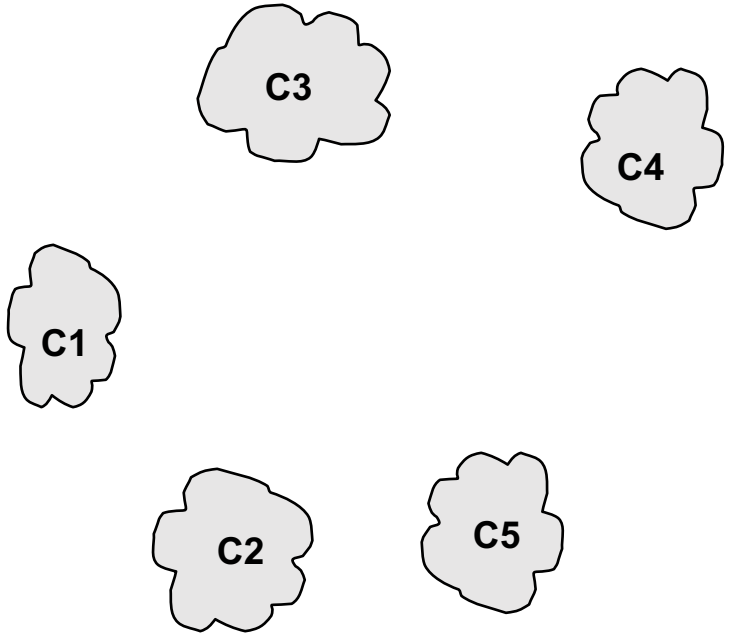
Proximity Matrix



Source: Tan et al., 2019

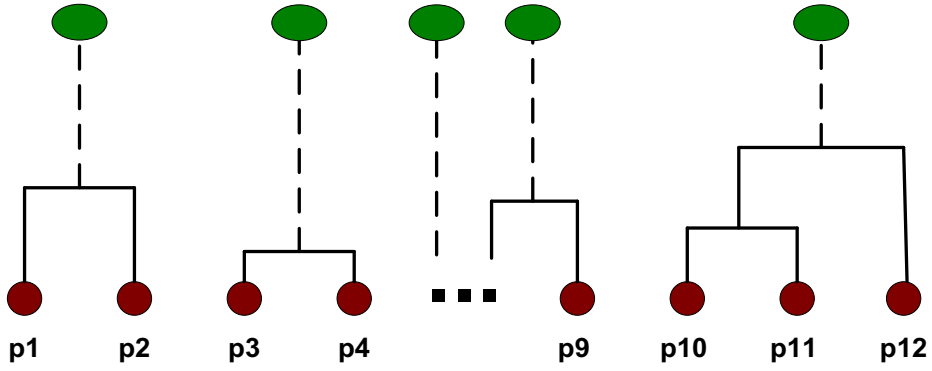
Intermediate Situation

- After some merging steps, we have some clusters



	C1	C2	C3	C4	C5
C1					
C2					
C3					
C4					
C5					

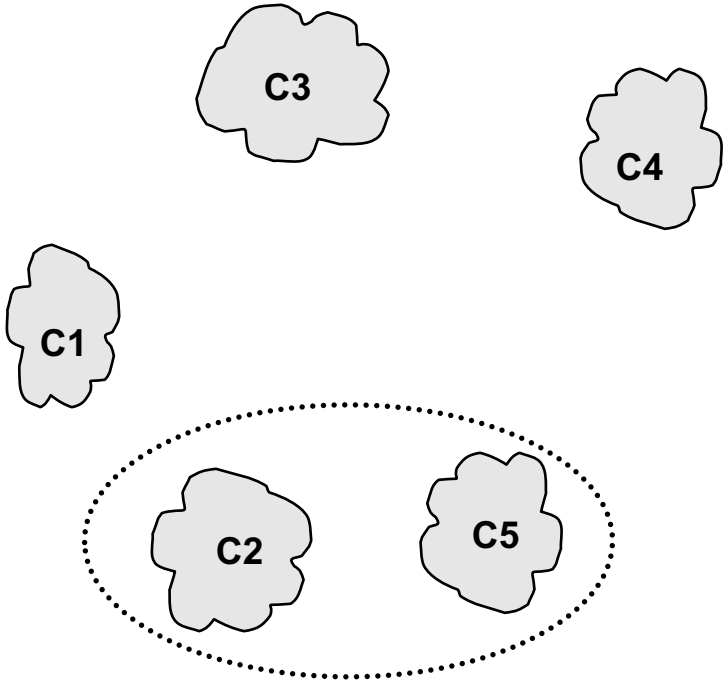
Proximity Matrix

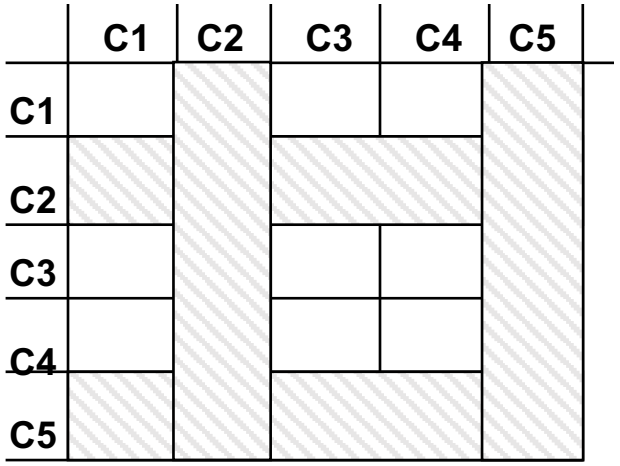


Source: Tan et al., 2019

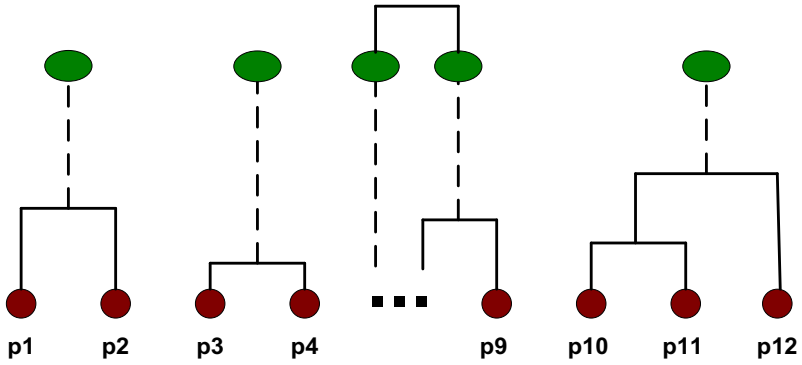
Step 4

- We want to merge the two closest clusters (C2 and C5) and update the proximity matrix.





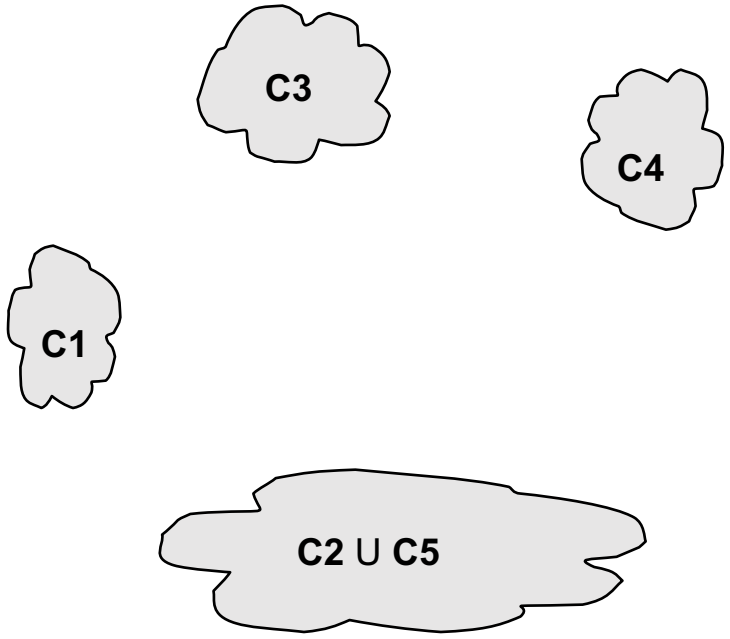
Proximity Matrix



Source: Tan et al., 2019

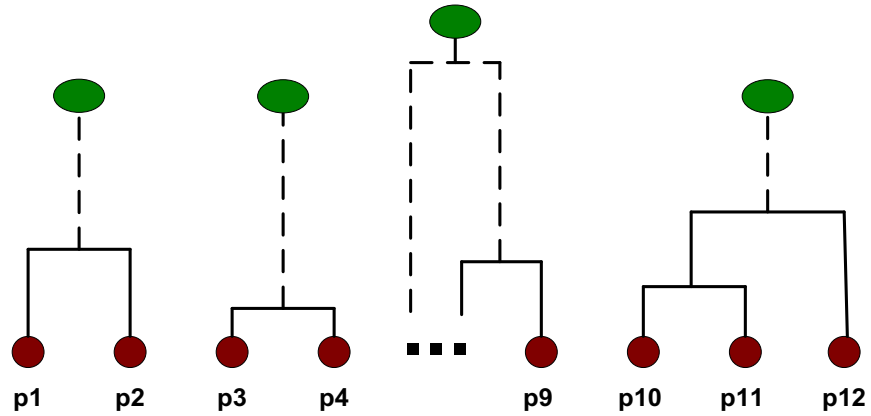
Step 5

- The question is "How do we update the proximity matrix?"



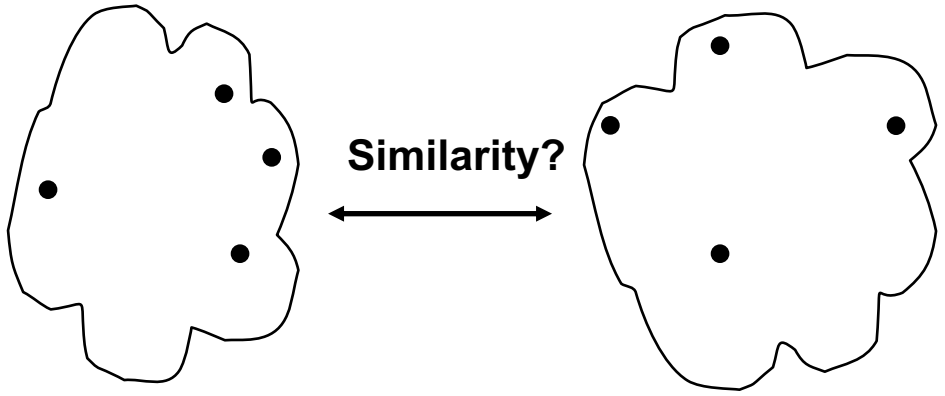
	C1	C2 U C5	C3	C4
C1		?		
C2 U C5	?	?	?	?
C3		?		
C4		?		

Proximity Matrix



Source: Tan et al., 2019

How to Define Inter-Cluster Distance



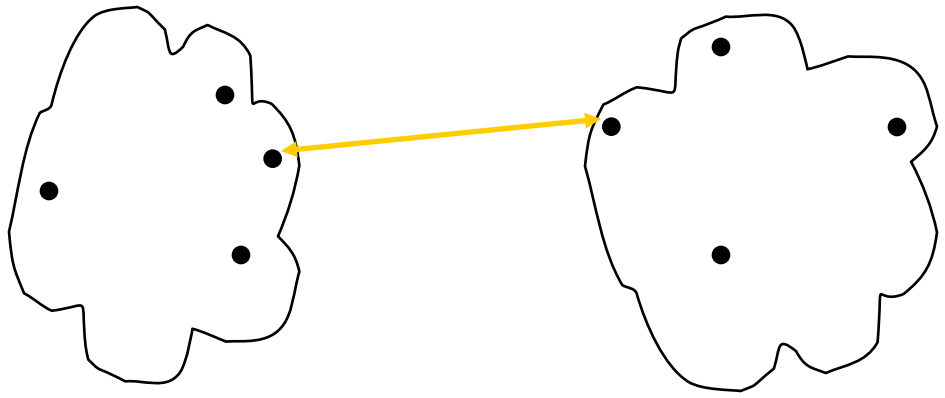
- MIN
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function.
  - Ward's Method uses squared error

	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

Source: Tan et al., 2019

How to Define Inter-Cluster Similarity



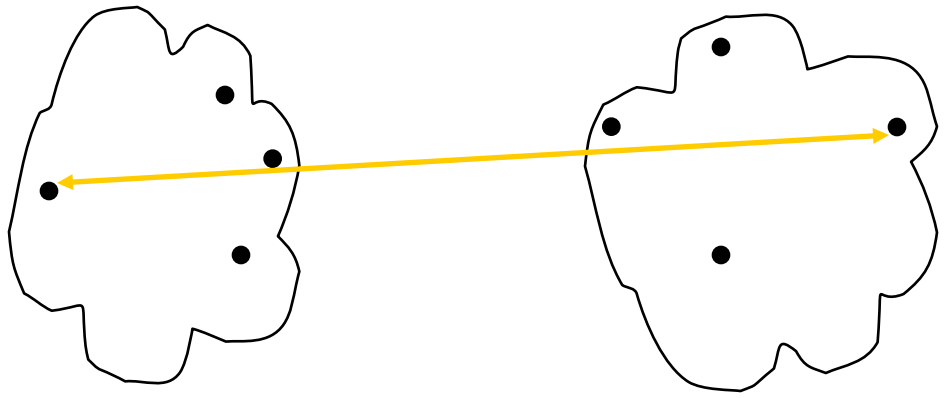
	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- **MIN**
- MAX
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function.
  - Ward's Method uses squared error

Source: Tan et al., 2019

How to Define Inter-Cluster Similarity



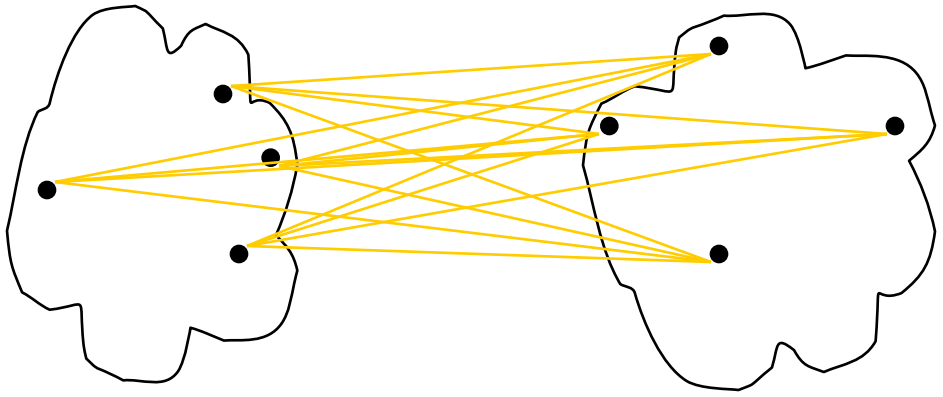
	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- **MAX**
- Group Average
- Distance Between Centroids
- Other methods driven by an objective function.
  - Ward's Method uses squared error

Source: Tan et al., 2019

How to Define Inter-Cluster Similarity



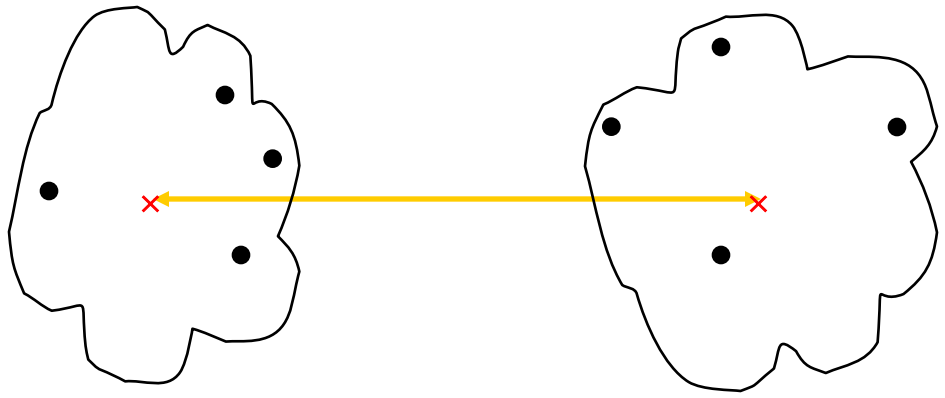
	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

Proximity Matrix

- MIN
- MAX
- **Group Average**
- Distance Between Centroids
- Other methods driven by an objective function.
  - Ward's Method uses squared error

Source: Tan et al., 2019

How to Define Inter-Cluster Similarity



	p1	p2	p3	p4	p5	. . .
p1						
p2						
p3						
p4						
p5						
.						
.						
.						

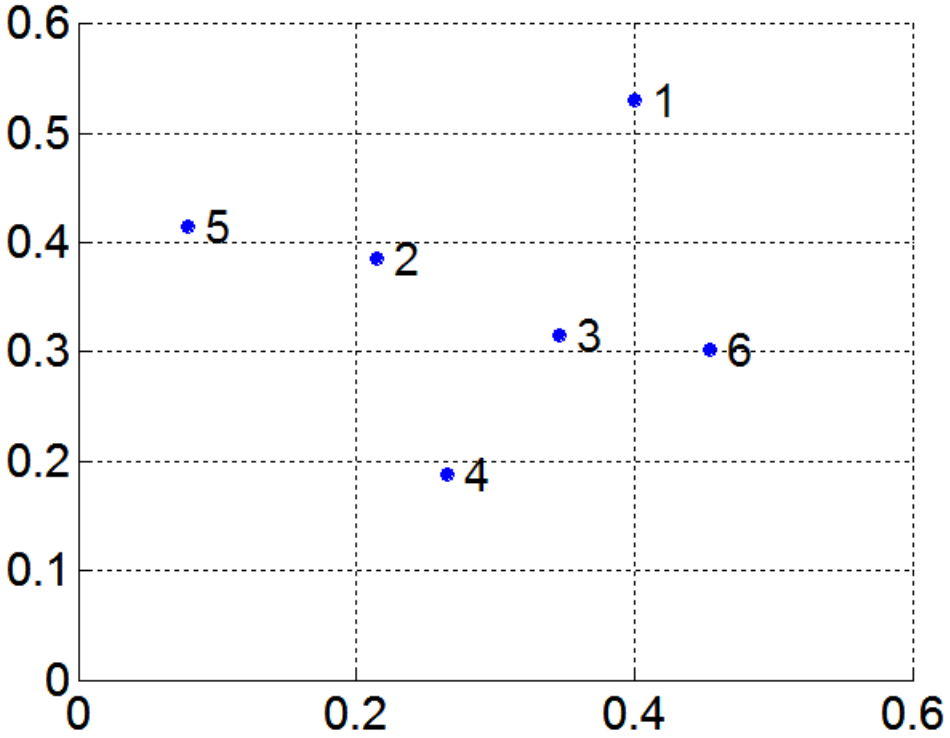
Proximity Matrix

- MIN
- MAX
- Group Average
- **Distance Between Centroids**
- Other methods driven by an objective function.
  - Ward's Method uses squared error

Source: Tan et al., 2019

MIN or Single Link

- Proximity of two clusters is based on the two closest points in the different clusters
  - Determined by one pair of points, i.e., by one link in the proximity graph
- Example:

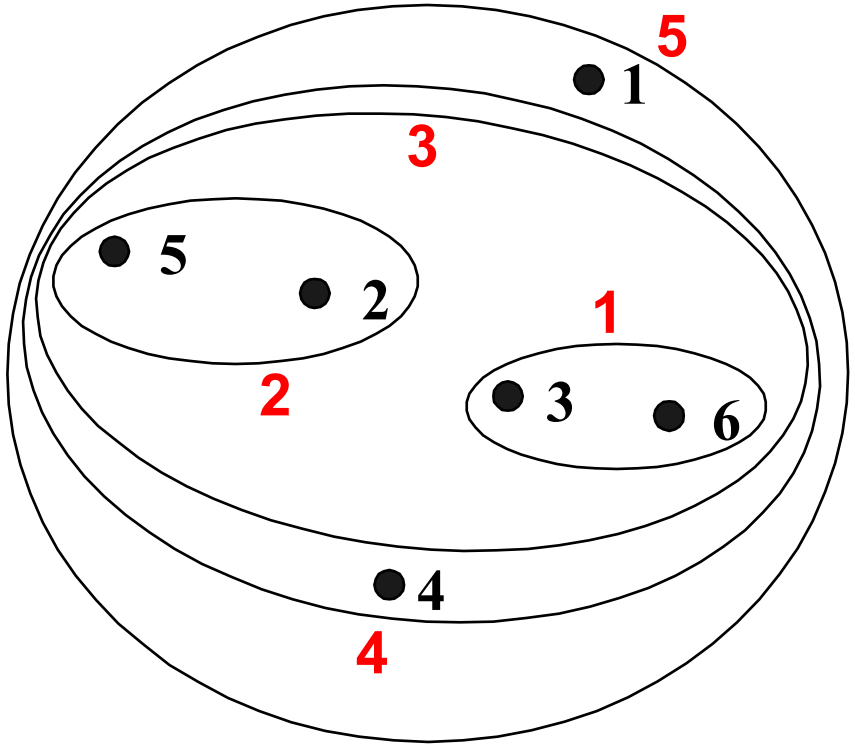


Distance Matrix:

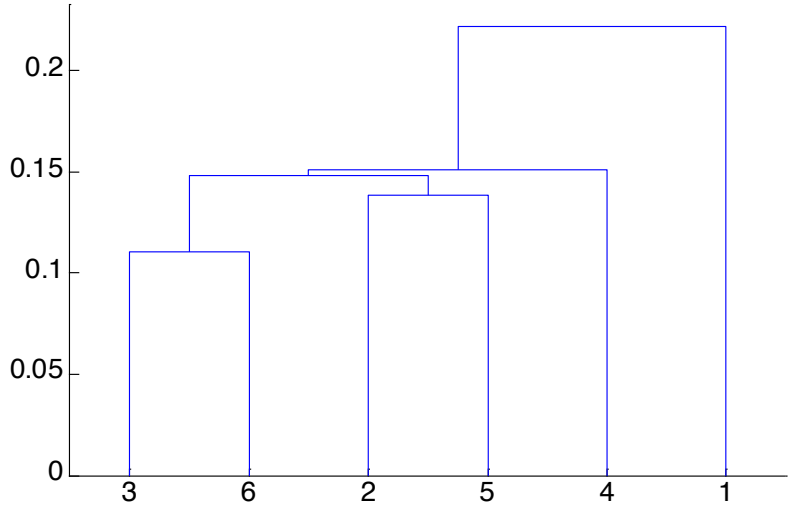
	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

Source: Tan et al., 2019

Hierarchical Clustering: MIN



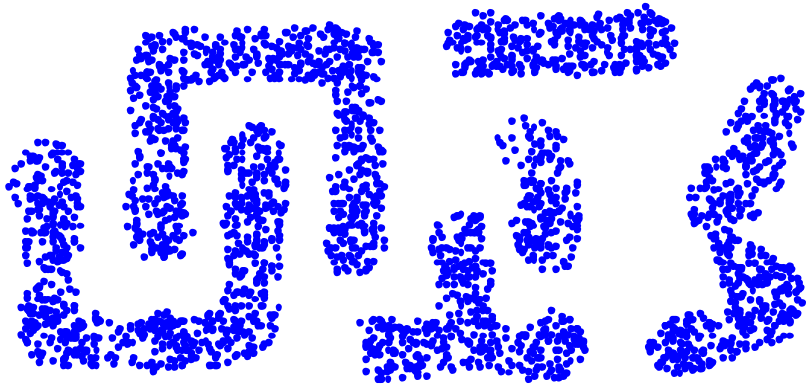
Nested Clusters



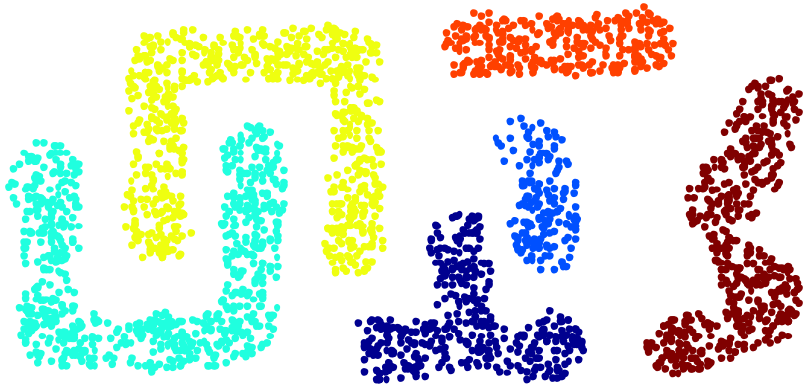
Dendrogram

Source: Tan et al., 2019

Strength of MIN



Original Points

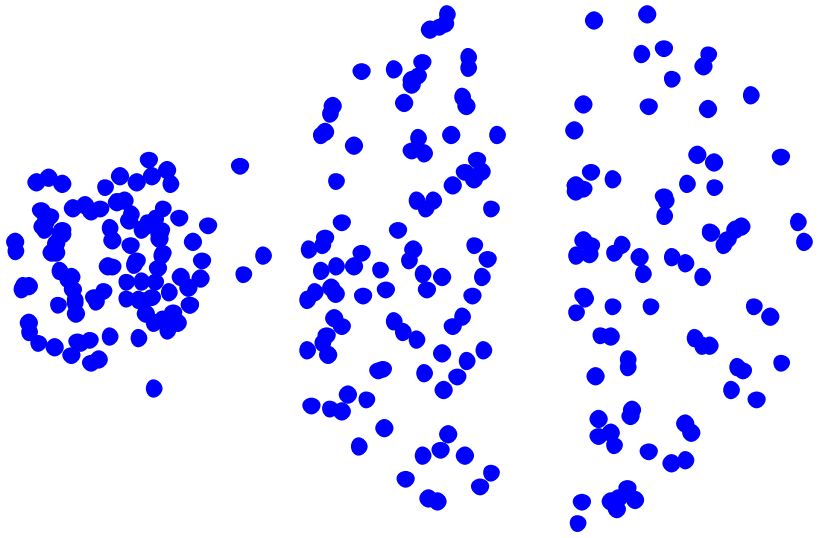


Six Clusters

- **Can handle non-elliptical shapes**

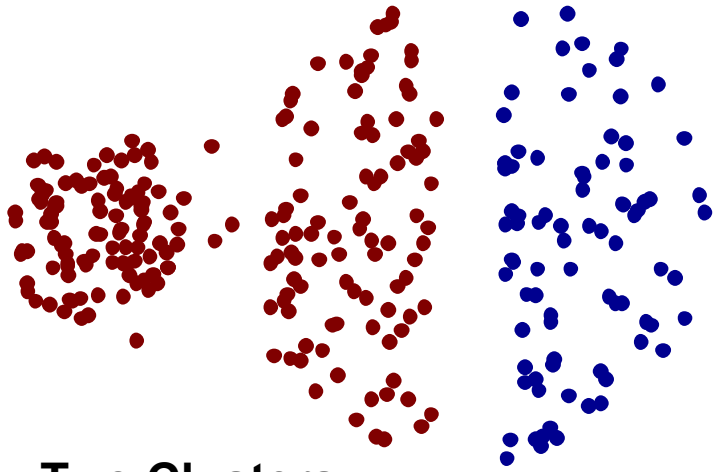
Source: Tan et al., 2019

Limitations of MIN

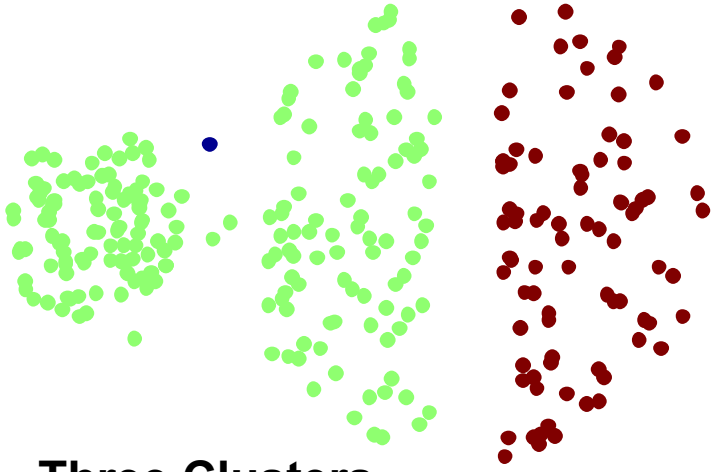


Original Points

- Sensitive to noise



Two Clusters

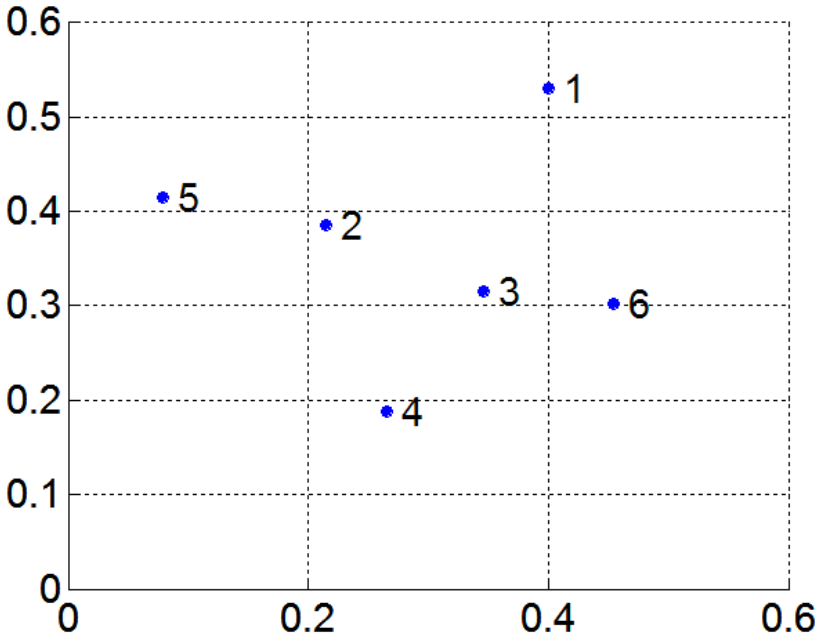


Three Clusters

Source: Tan et al., 2019

MAX or Complete Linkage

- Proximity of two clusters is based on the two most distant points in the different clusters
  - Determined by all pairs of points in the two clusters

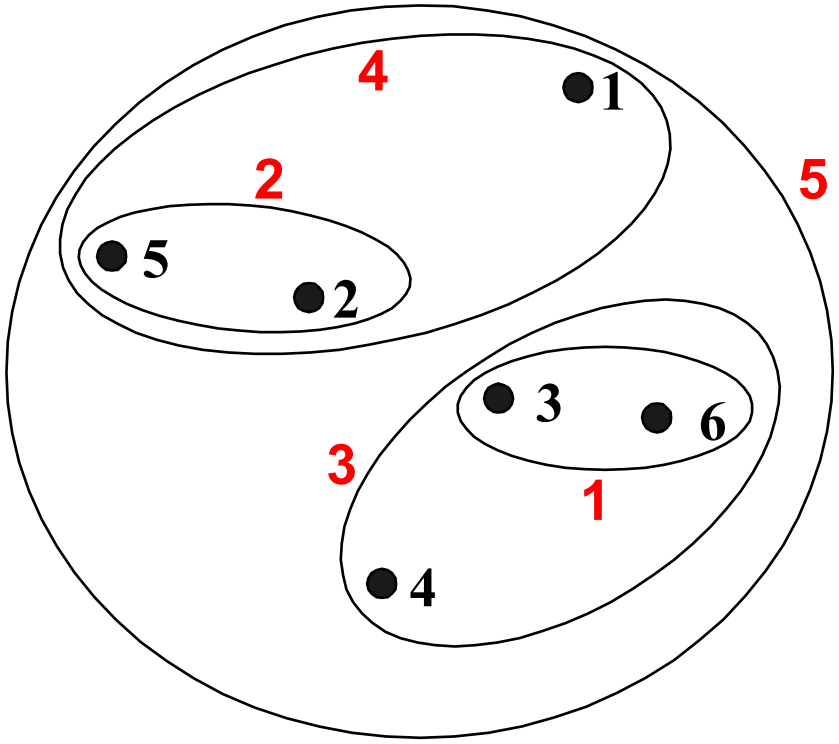


Distance Matrix:

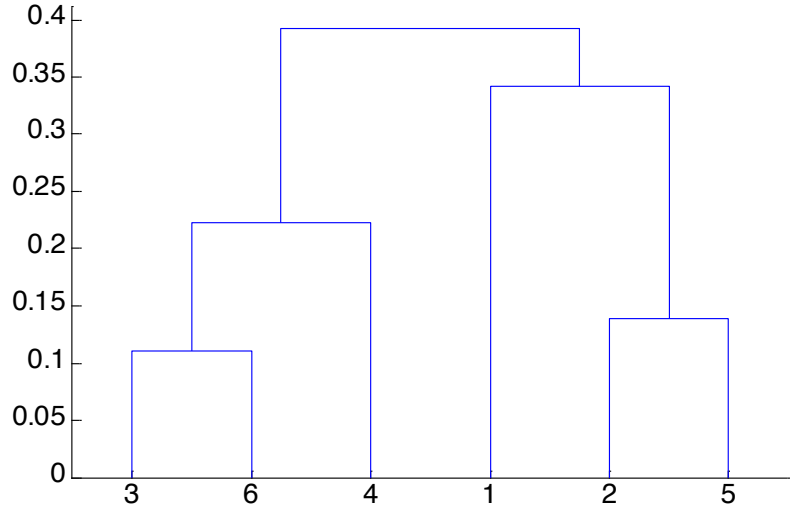
	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

Source: Tan et al., 2019

Hierarchical Clustering: MAX



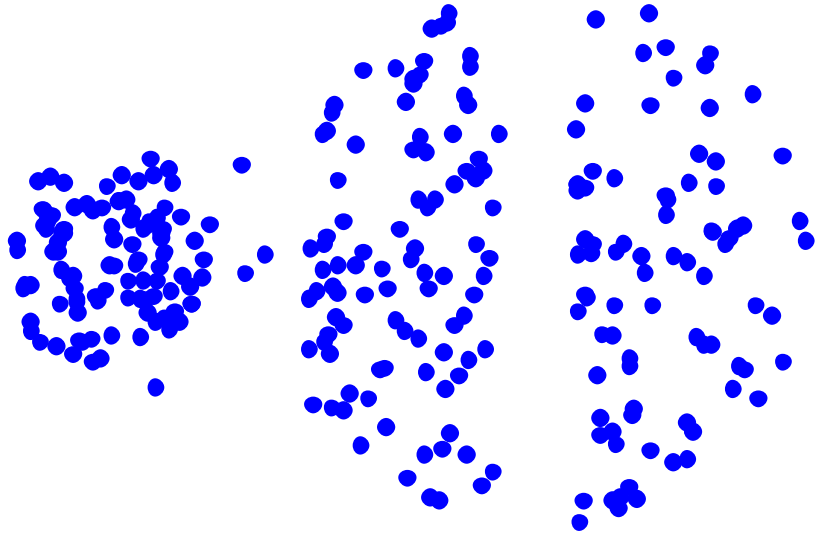
Nested Clusters



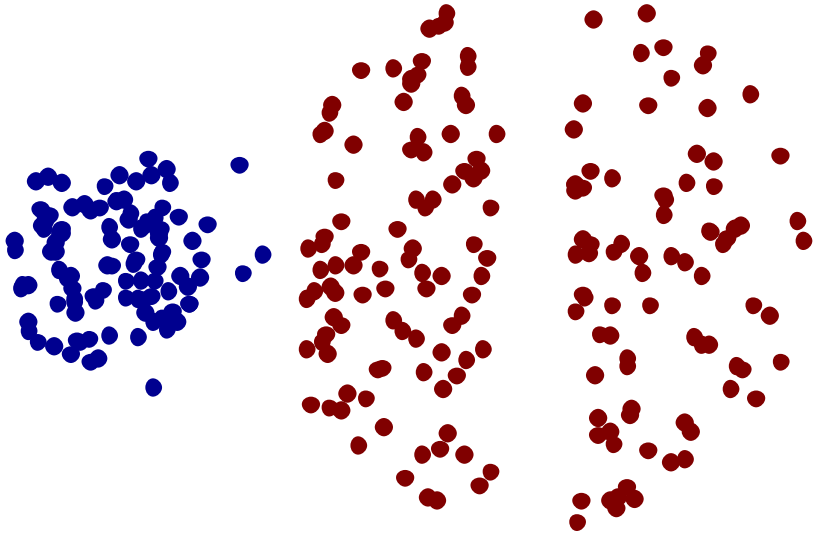
Dendrogram

Source: Tan et al., 2019

Strength of MAX



Original Points

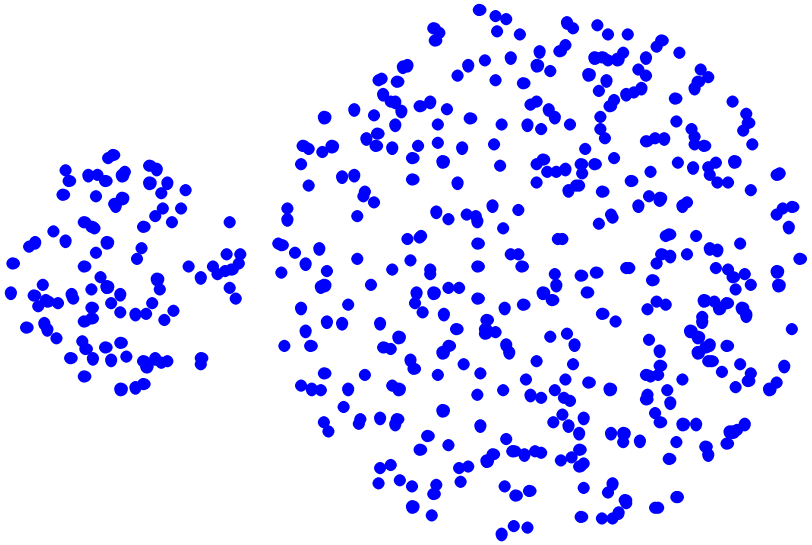


Two Clusters

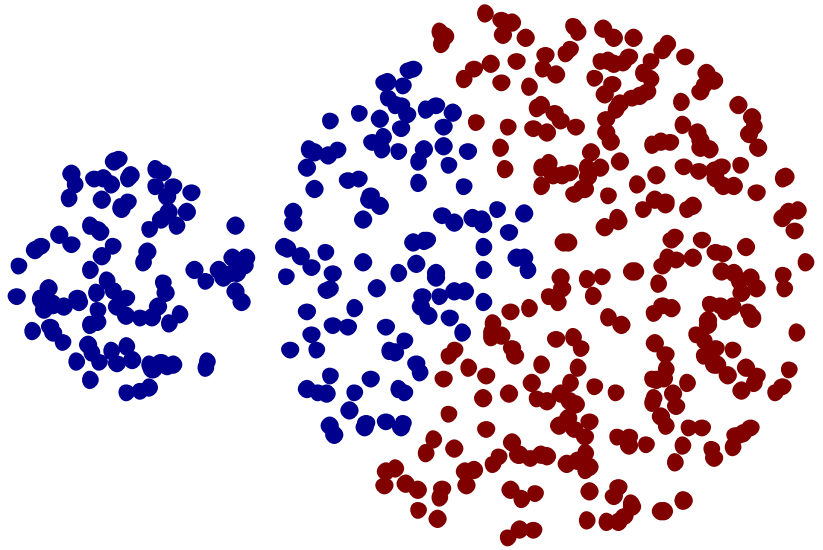
- **Less susceptible to noise**

Source: Tan et al., 2019

Limitations of MAX



Original Points



Two Clusters

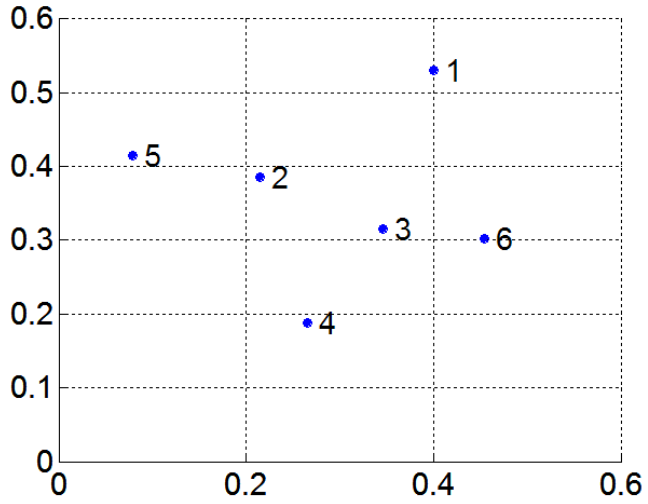
- Tends to break large clusters
- Biased towards globular clusters

Source: Tan et al., 2019

Group Average

- Proximity of two clusters is the average of pairwise proximity between points in the two clusters.

$$\text{proximity}(\text{Cluster}_i, \text{Cluster}_j) = \frac{\sum_{\substack{p_i \in \text{Cluster}_i \\ p_j \in \text{Cluster}_j}} \text{proximity}(p_i, p_j)}{|\text{Cluster}_i| \times |\text{Cluster}_j|}$$

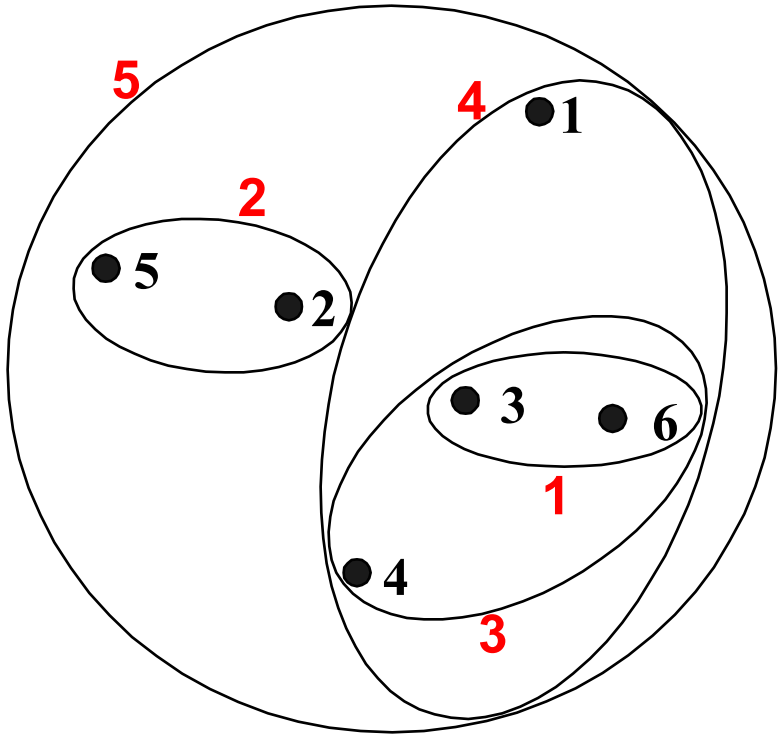


Distance Matrix:

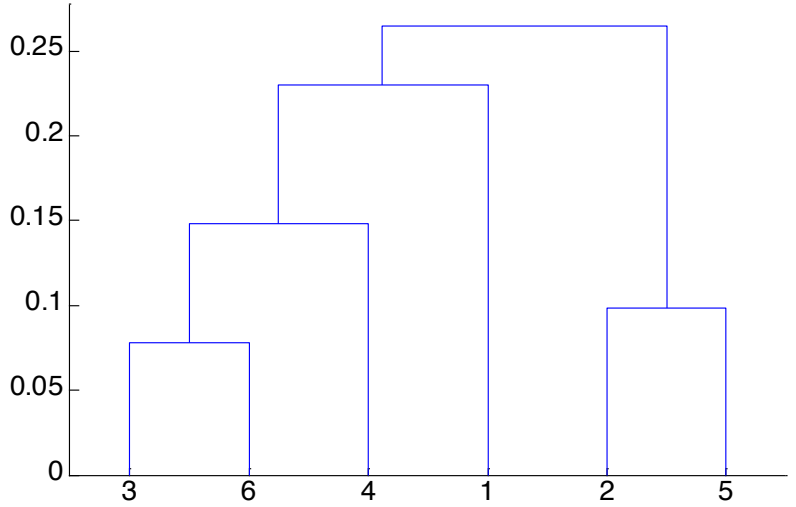
	p1	p2	p3	p4	p5	p6
p1	0.00	0.24	0.22	0.37	0.34	0.23
p2	0.24	0.00	0.15	0.20	0.14	0.25
p3	0.22	0.15	0.00	0.15	0.28	0.11
p4	0.37	0.20	0.15	0.00	0.29	0.22
p5	0.34	0.14	0.28	0.29	0.00	0.39
p6	0.23	0.25	0.11	0.22	0.39	0.00

Source: Tan et al., 2019

Hierarchical Clustering: Group Average



Nested Clusters



Dendrogram

Source: Tan et al., 2019

Hierarchical Clustering: Group Average

- Compromise between Single and Complete Link
- Strengths
  - Less susceptible to noise
- Limitations
  - Biased towards globular clusters

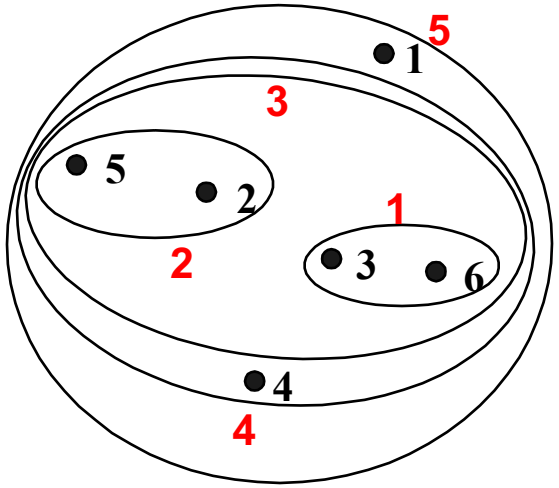
Source: Tan et al., 2019

Cluster Similarity: Ward's Method

- Similarity of two clusters is based on the increase in squared error when two clusters are merged
  - Similar to group average if distance between points is distance squared
- Less susceptible to noise
- Biased towards globular clusters
- Hierarchical analogue of K-means
  - Can be used to initialize K-means

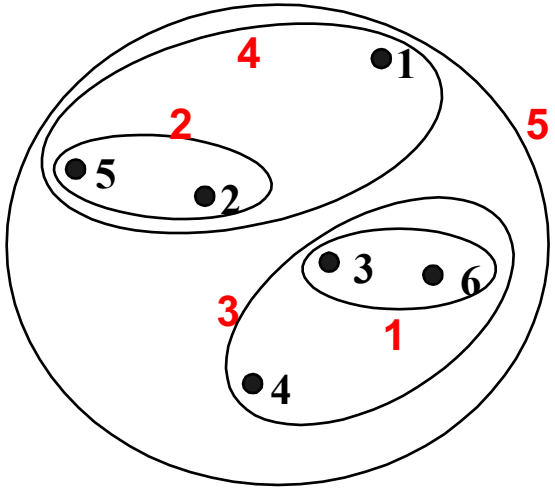
Source: Tan et al., 2019

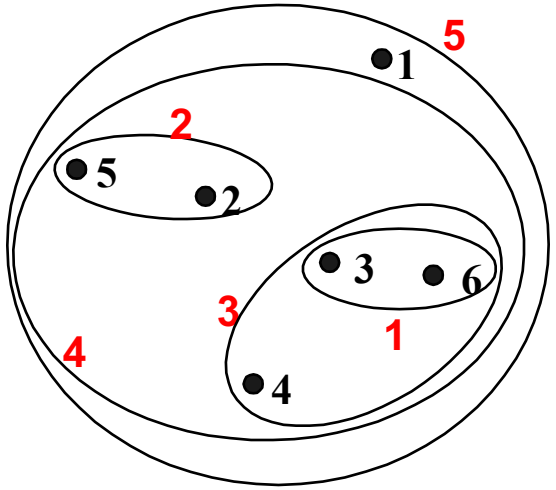
Hierarchical Clustering: Comparison



MIN

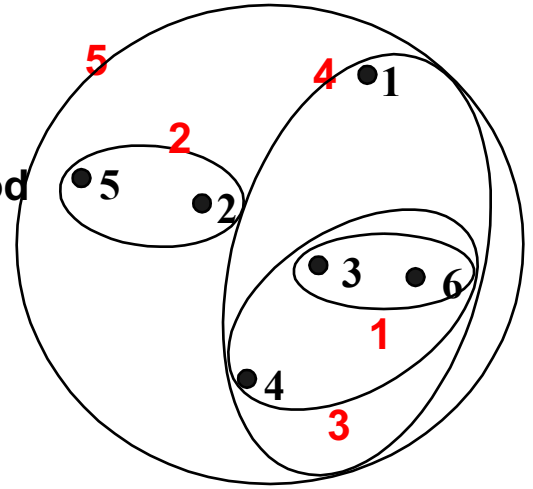
MAX





Group Average

Ward's Method



Source: Tan et al., 2019

Hierarchical Clustering: Time and Space requirements

- $O(N^2)$ space since it uses the proximity matrix.
  - N is the number of points.

- $O(N^3)$ time in many cases
  - There are N steps and at each step the size, N^2, proximity matrix must be updated and searched
  - Complexity can be reduced to $O(N^2 \log(N))$ time with some cleverness

Source: Tan et al., 2019

Hierarchical Clustering: Problems and Limitations

- Once a decision is made to combine two clusters, it cannot be undone
- No global objective function is directly minimized
- Different schemes have problems with one or more of the following:
  - Sensitivity to noise
  - Difficulty handling clusters of different sizes and non-globular shapes
  - Breaking large clusters

Source: Tan et al., 2019

Density Based Clustering

- Clusters are regions of high density that are separated from one another by regions on low density.

Source: Tan et al., 2019

Cluster Validity

- For supervised classification we have a variety of measures to evaluate how good our model is
  - Accuracy, precision, recall

- For cluster analysis, the analogous question is how to evaluate the "goodness" of the resulting clusters?

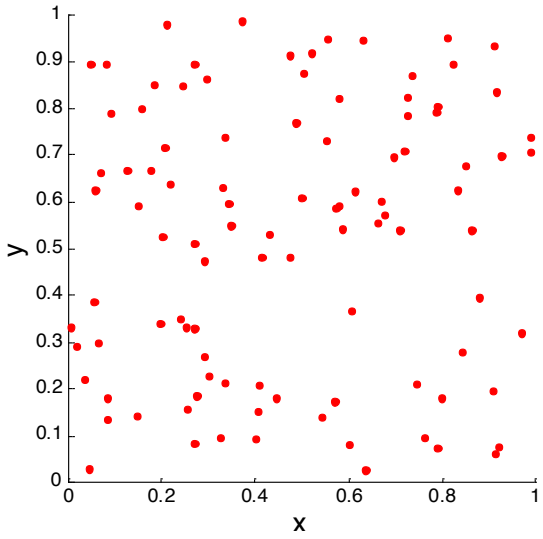
- But "clusters are in the eye of the beholder"!
  - In practice the clusters we find are defined by the clustering algorithm

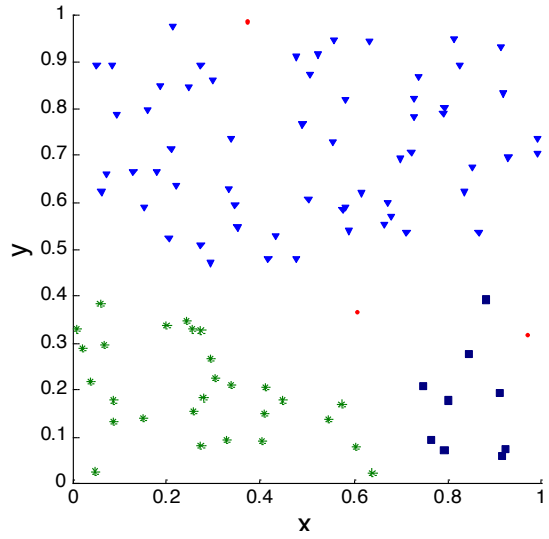
- Then why do we want to evaluate them?
  - To avoid finding patterns in noise
  - To compare clustering algorithms
  - To compare two sets of clusters
  - To compare two clusters

Source: Tan et al., 2019

Clusters found in Random Data

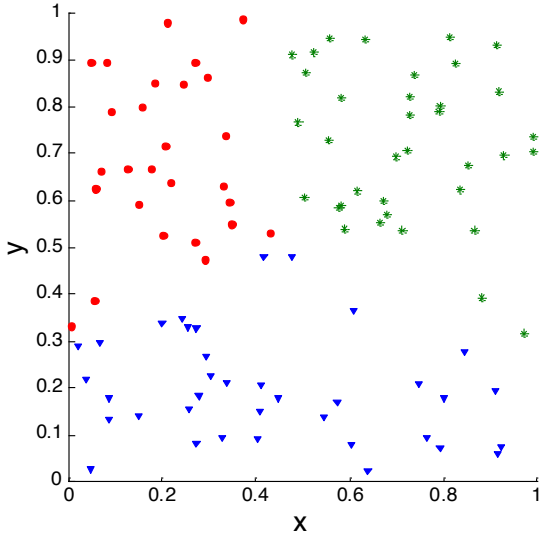
Random Points

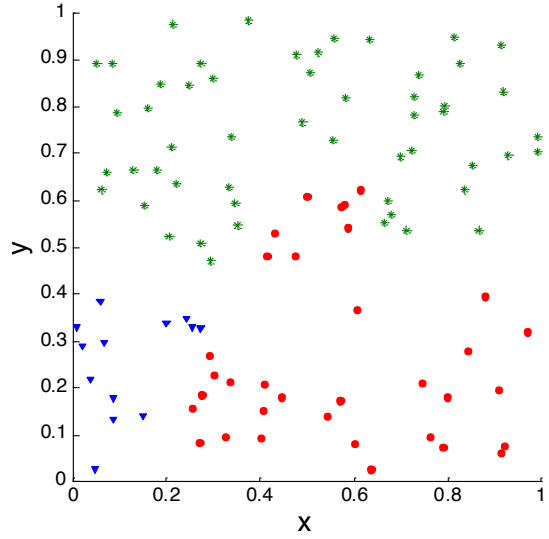




DBSCAN

K-means





Complete Link

Source: Tan et al., 2019

Measures of Cluster Validity

- Numerical measures that are applied to judge various aspects of cluster validity, are classified into the following two types.
  - Supervised: Used to measure the extent to which cluster labels match externally supplied class labels.
    - Entropy
    - Often called *external indices* because they use information external to the data
  - Unsupervised: Used to measure the goodness of a clustering structure *without* respect to external information.
    - Sum of Squared Error (SSE)
    - Often called *internal indices* because they only use information in the data

- You can use supervised or unsupervised measures to compare clusters or clusterings

Source: Tan et al., 2019

Unsupervised Measures: Cohesion and Separation

- Cluster Cohesion: Measures how closely related are objects in a cluster
  - Example: SSE
- Cluster Separation: Measure how distinct or well-separated a cluster is from other clusters
- Example: Squared Error
  - Cohesion is measured by the within cluster sum of squares (SSE):

$$SSE = \sum_{i} \sum_{x \in C_i} (x - m_i)^2$$

  - Separation is measured by the between cluster sum of squares:

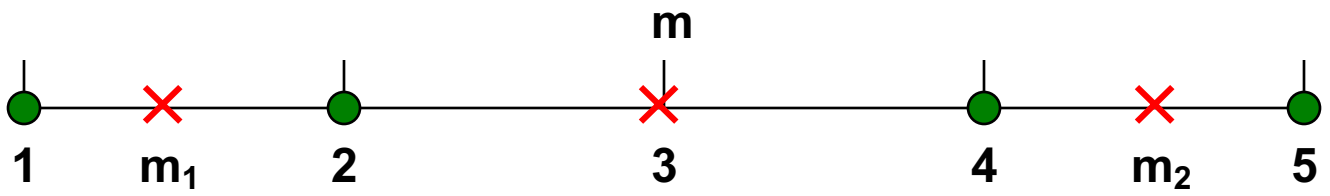
$$SSB = \sum_{i} |C_i| (m - m_i)^2$$

Where $|C_i|$ is the size of cluster i

Source: Tan et al., 2019

Unsupervised Measures: Cohesion and Separation

- Example: SSE
  - SSB + SSE = constant



K=1 cluster:

$$SSE = (1-3)^2 + (2-3)^2 + (4-3)^2 + (5-3)^2 = 10$$

$$SSB = 4 \times (3-3)^2 = 0$$

$$Total = 10 + 0 = 10$$

K=2 clusters:

$$SSE = (1-1.5)^2 + (2-1.5)^2 + (4-4.5)^2 + (5-4.5)^2 = 1$$

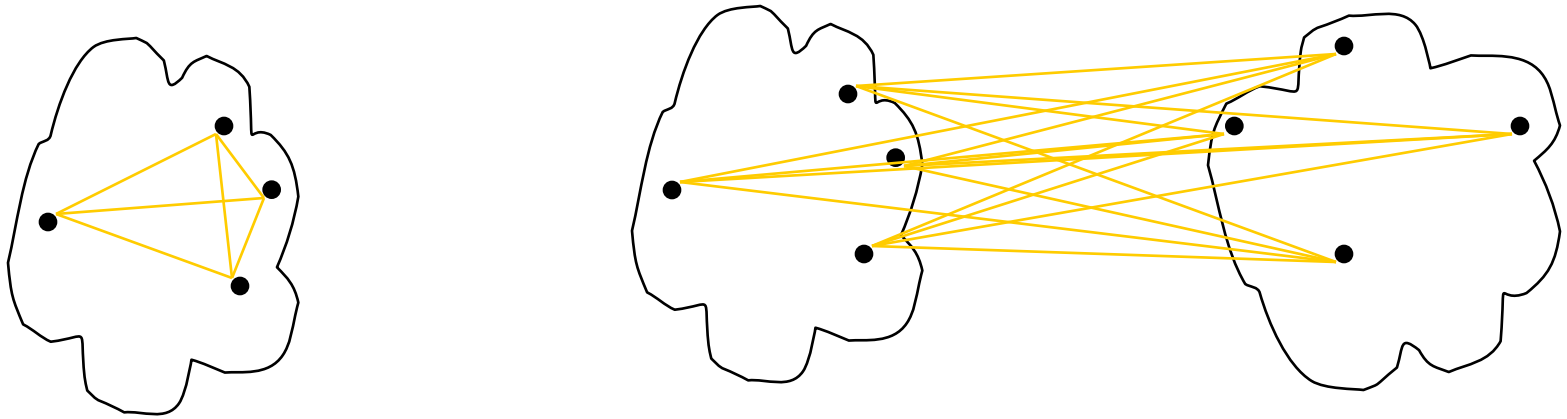
$$SSB = 2 \times (3-1.5)^2 + 2 \times (4.5-3)^2 = 9$$

$$Total = 1 + 9 = 10$$

Source: Tan et al., 2019

Unsupervised Measures: Cohesion and Separation

- A proximity graph-based approach can also be used for cohesion and separation.
  - Cluster cohesion is the sum of the weight of all links within a cluster.
  - Cluster separation is the sum of the weights between nodes in the cluster and nodes outside the cluster.



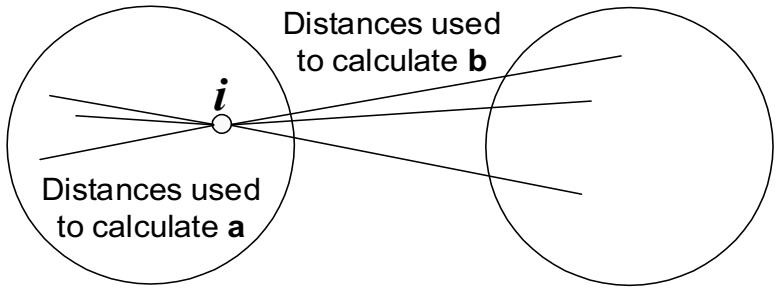
cohesion separation

Source: Tan et al., 2019

Unsupervised Measures: Silhouette Coefficient

- Silhouette coefficient combines ideas of both cohesion and separation, but for individual points, as well as clusters and clusterings
- For an individual point, ***i***
  - Calculate ***a*** = average distance of ***i*** to the points in its cluster
  - Calculate ***b*** = min (average distance of ***i*** to points in another cluster)
  - The silhouette coefficient for a point is then given by

$$s = (b - a) / max(a,b)$$



  - Value can vary between -1 and 1
  - Typically ranges between 0 and 1.
  - The closer to 1 the better.

- Can calculate the average silhouette coefficient for a cluster or a clustering

Source: Tan et al., 2019

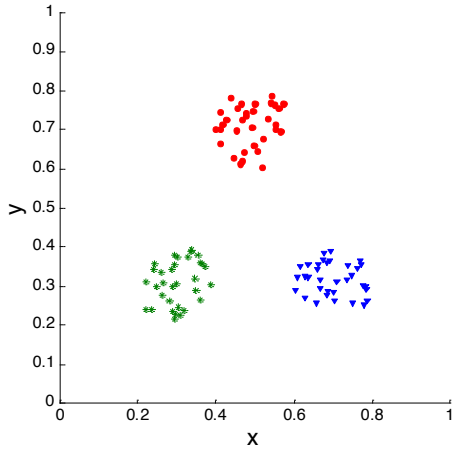
Measuring Cluster Validity Via Correlation

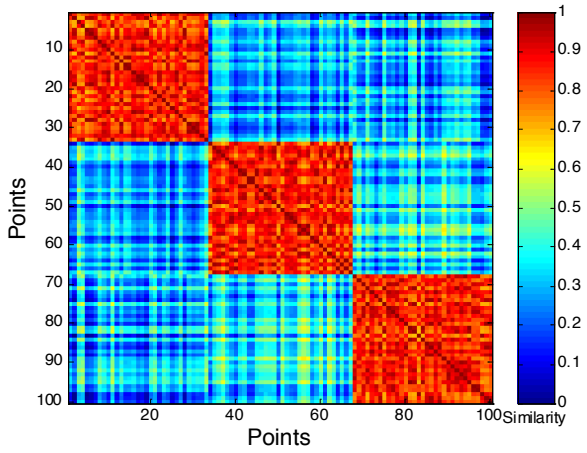
- Two matrices
  - Proximity Matrix
  - Ideal Similarity Matrix
    - One row and one column for each data point
    - An entry is 1 if the associated pair of points belong to the same cluster
    - An entry is 0 if the associated pair of points belongs to different clusters
- Compute the correlation between the two matrices
  - Since the matrices are symmetric, only the correlation between $n(n-1) / 2$ entries needs to be calculated.
- High magnitude of correlation indicates that points that belong to the same cluster are close to each other.
  - Correlation may be positive or negative depending on whether the similarity matrix is a similarity or dissimilarity matrix
- Not a good measure for some density or contiguity based clusters.

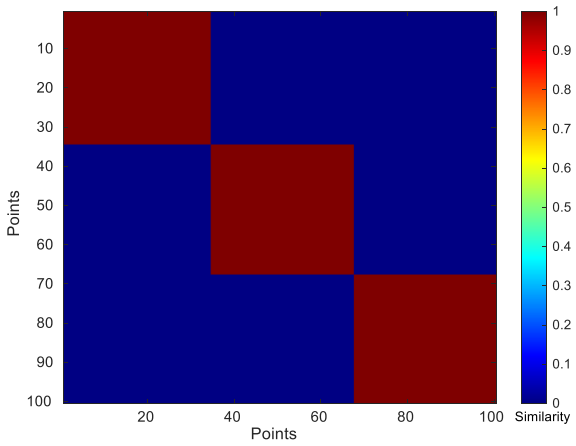
Source: Tan et al., 2019

Measuring Cluster Validity Via Correlation

- Correlation of ideal similarity and proximity matrices for the K-means clusterings of the following well-clustered data set.





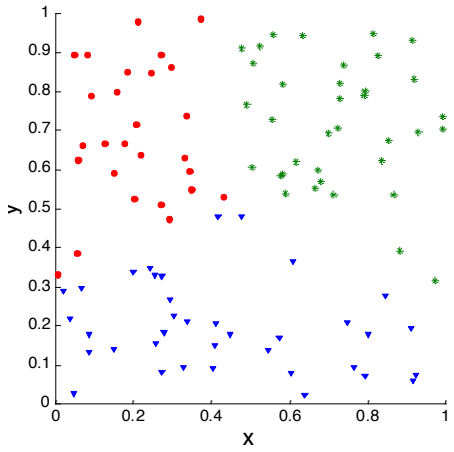


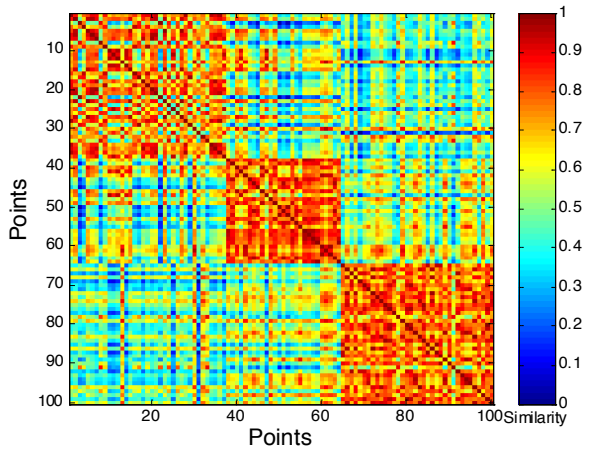
Corr = 0.9235

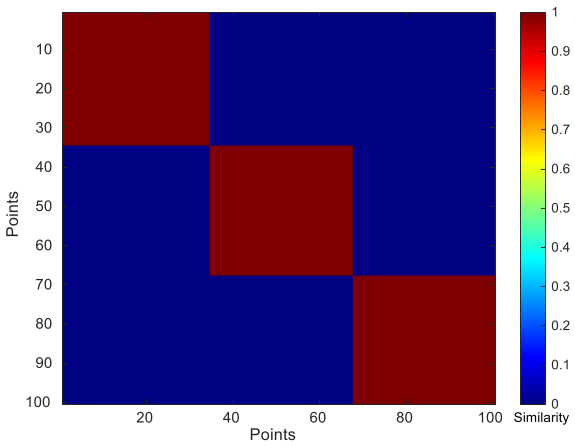
Source: Tan et al., 2019

Measuring Cluster Validity Via Correlation

- Correlation of ideal similarity and proximity matrices for the K-means clusterings of the following random data set.







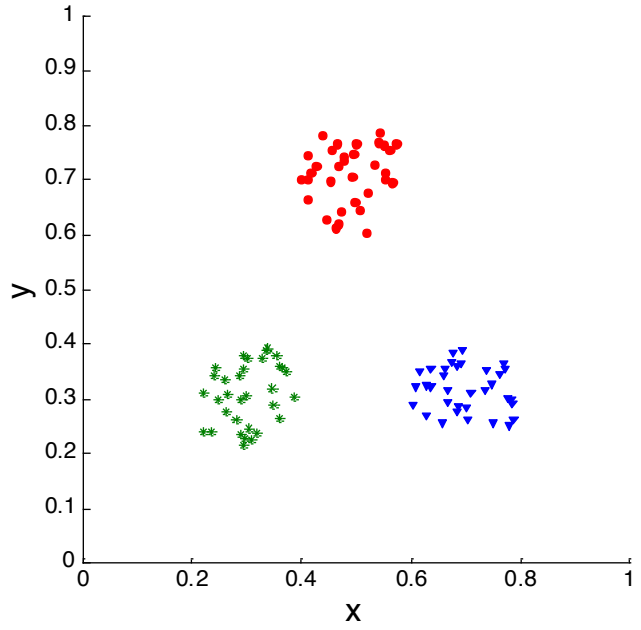
K-means

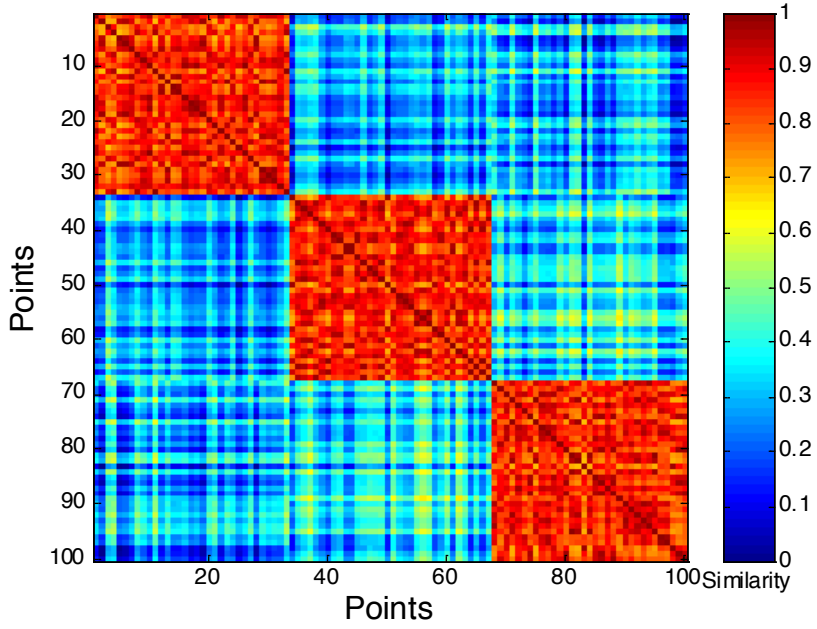
Corr = 0.5810

Source: Tan et al., 2019

Judging a Clustering Visually by its Similarity Matrix

- Order the similarity matrix with respect to cluster labels and inspect visually.

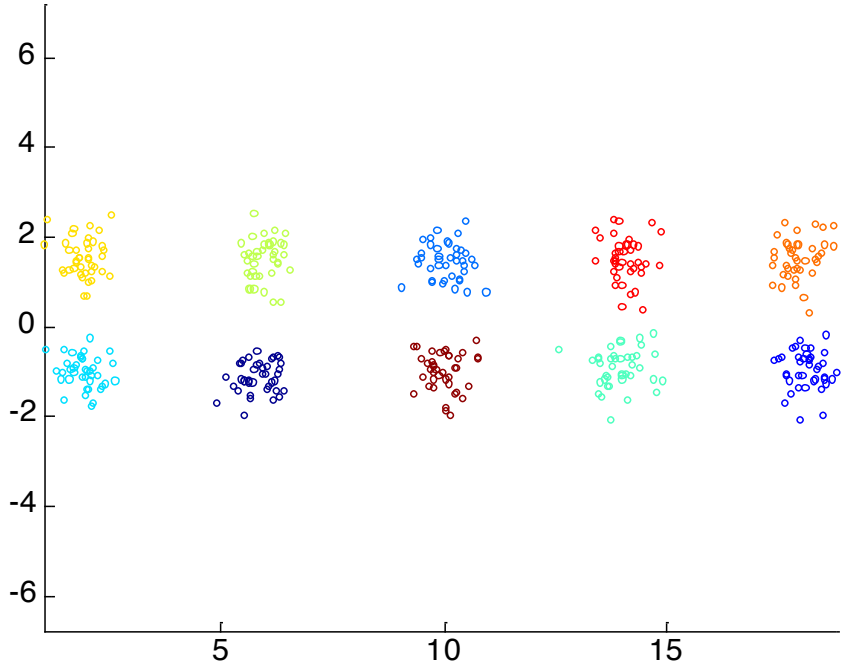


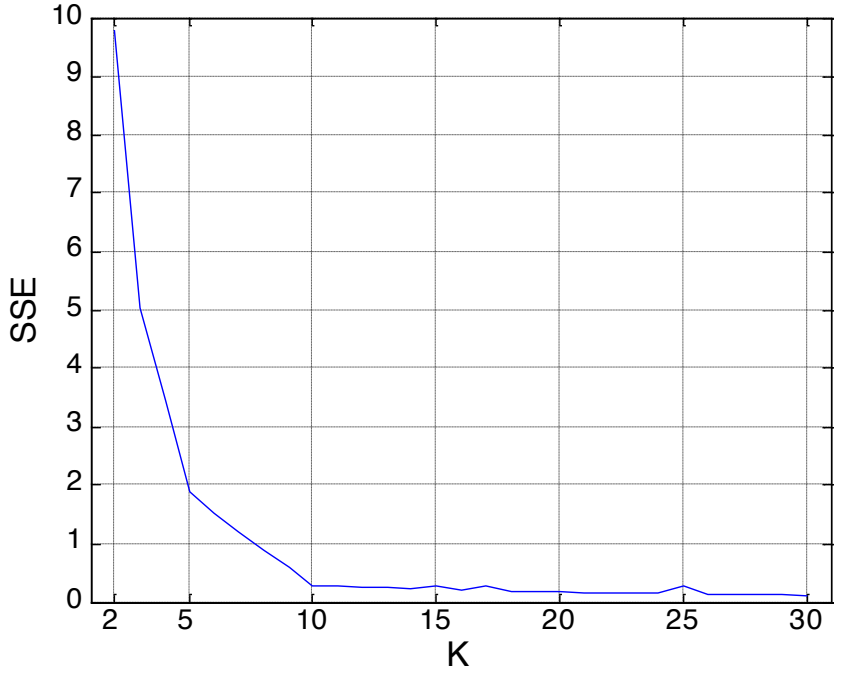


Source: Tan et al., 2019

Determining the Correct Number of Clusters

- SSE is good for comparing two clusterings or two clusters
- SSE can also be used to estimate the number of clusters

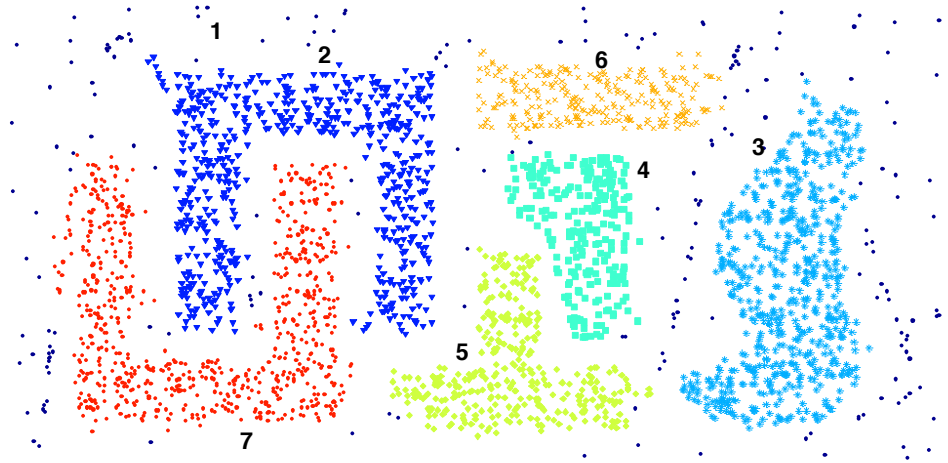


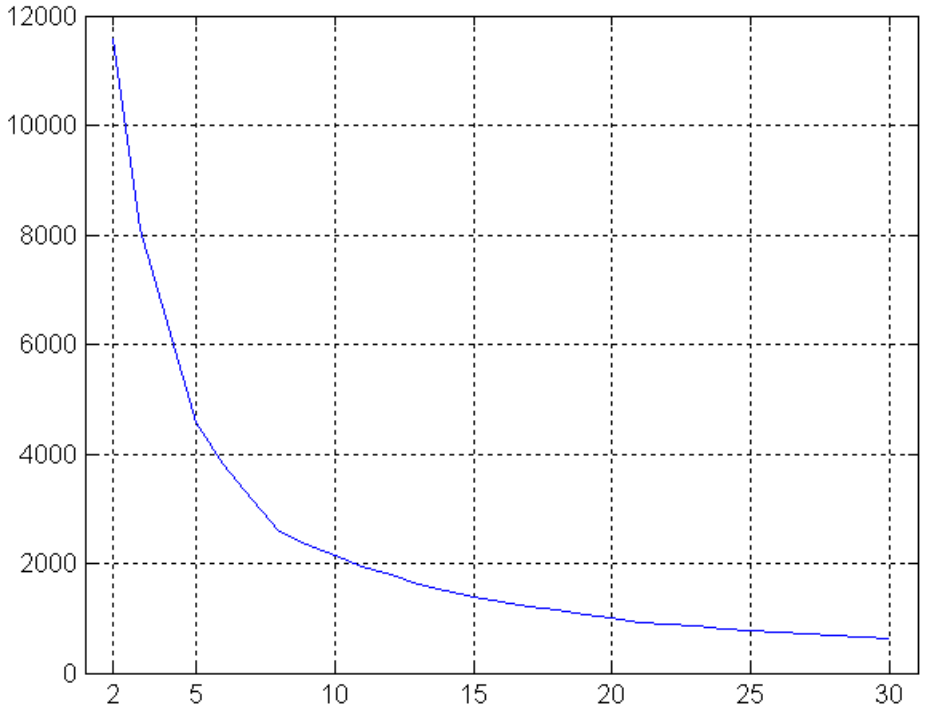


Source: Tan et al., 2019

Determining the Correct Number of Clusters

- SSE curve for a more complicated data set





SSE of clusters found using K-means

Source: Tan et al., 2019

IN4151 – Information Engineering

Unit 3: Introduction to Data and Information Science for Management (part IV)

Ángel Jiménez, angeljim@uchile.cl

Universidad de Chile
Facultad de Ciencias Físicas y Matemáticas
Departamento de Ingeniería Industrial

25 de abril de 2022