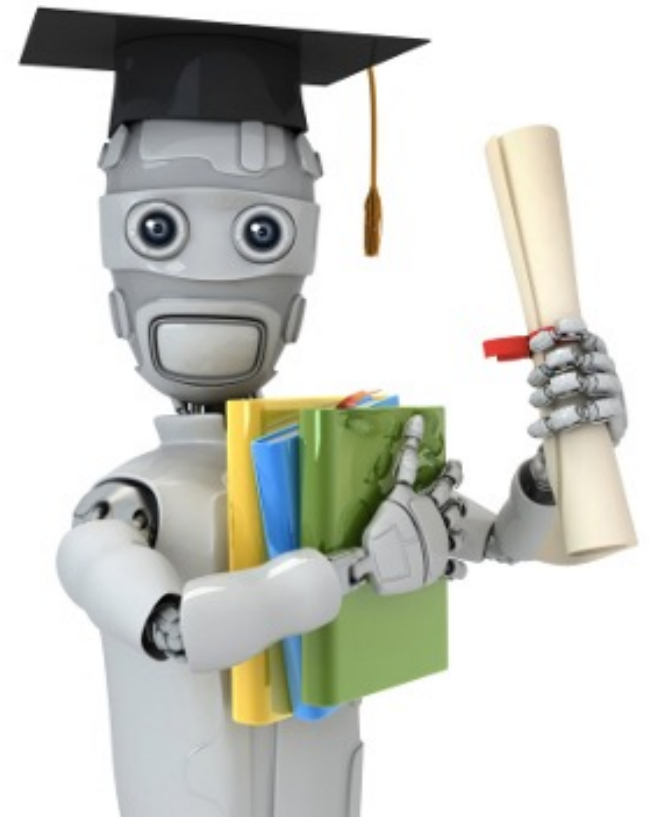


# 6.S079 MACHINE LEARNING 1

MARCH 5, 2024  
MIKE CAFARELLA

THANKS TO TIM KRASKA FOR  
SLIDES



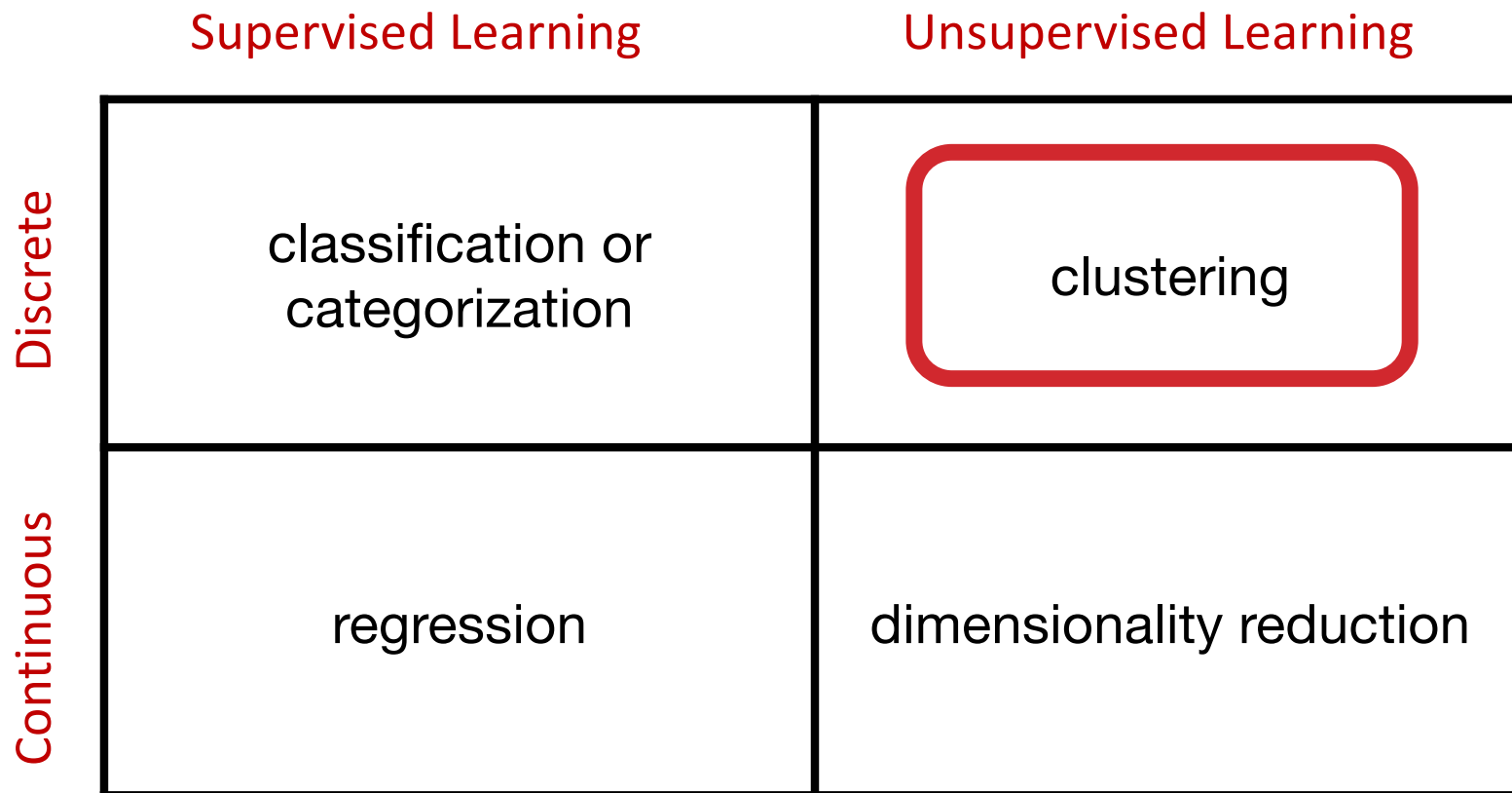
# MACHINE LEARNING PROBLEMS

(Boosted-) Decision Trees

K-Means

Agglomerative clustering

DBScan



(Boosted-) Decision Trees

PCA

# CLUSTERING STRATEGIES

## K-means

- Iteratively re-assign points to the nearest cluster center

## Agglomerative clustering

- Start with each point as its own cluster and iteratively merge the closest clusters

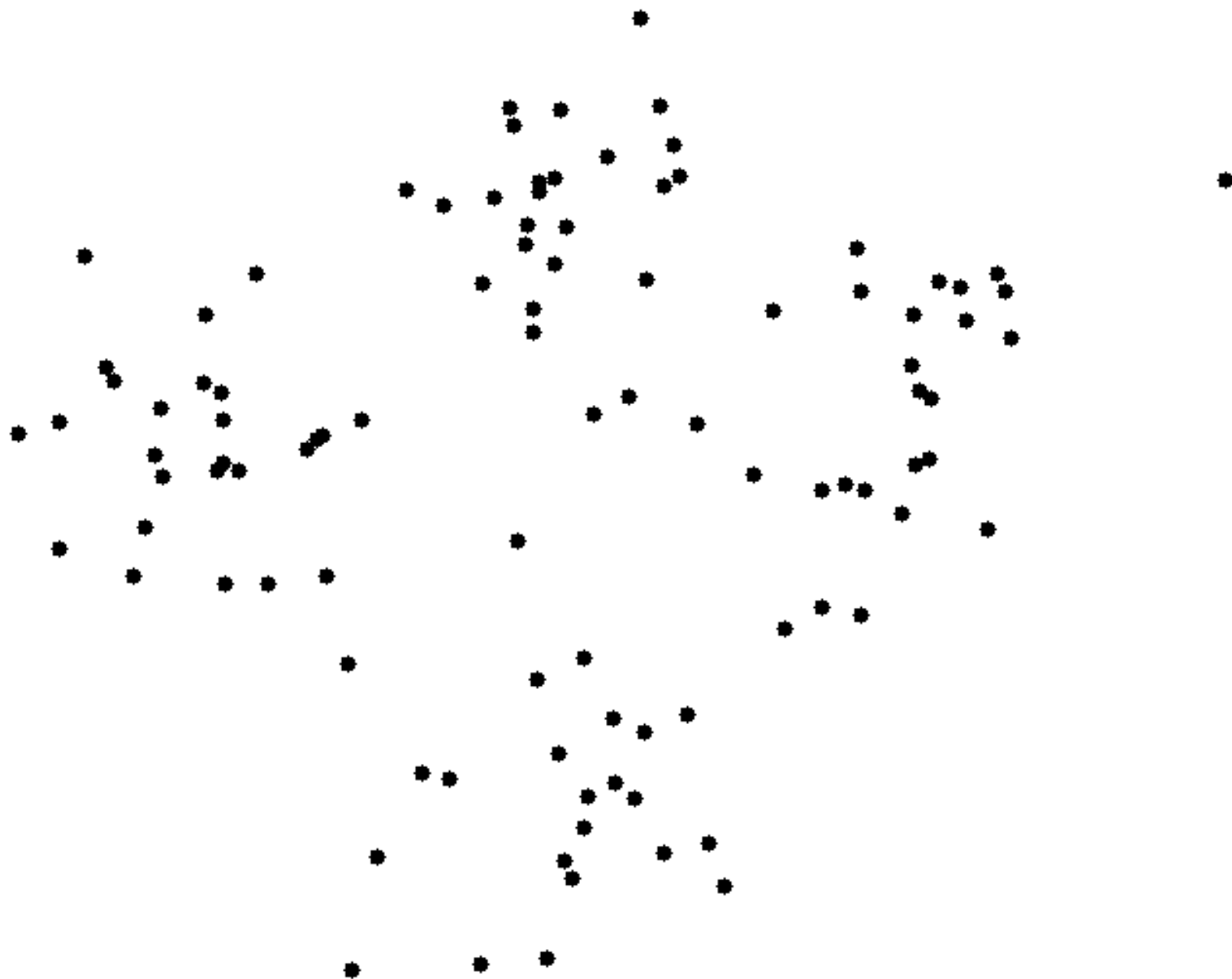
DBSCAN (Density-based spatial clustering of applications with noise)

EM Algorithm and Mixture Gaussian clustering

# K-MEANS

Lloyd's Algorithm is the most common, naïve approach

1. Choose  $k$  cluster centers. Place them randomly
2. Repeatedly:
  1. Find the data points closest to each center, assign them to that center's cluster
  2. Compute the centroid of each cluster
  3. Move each cluster's center to its centroid
3. Terminate when points don't move much



# CLUSTERING STRATEGIES

## K-means

- Iteratively re-assign points to the nearest cluster center

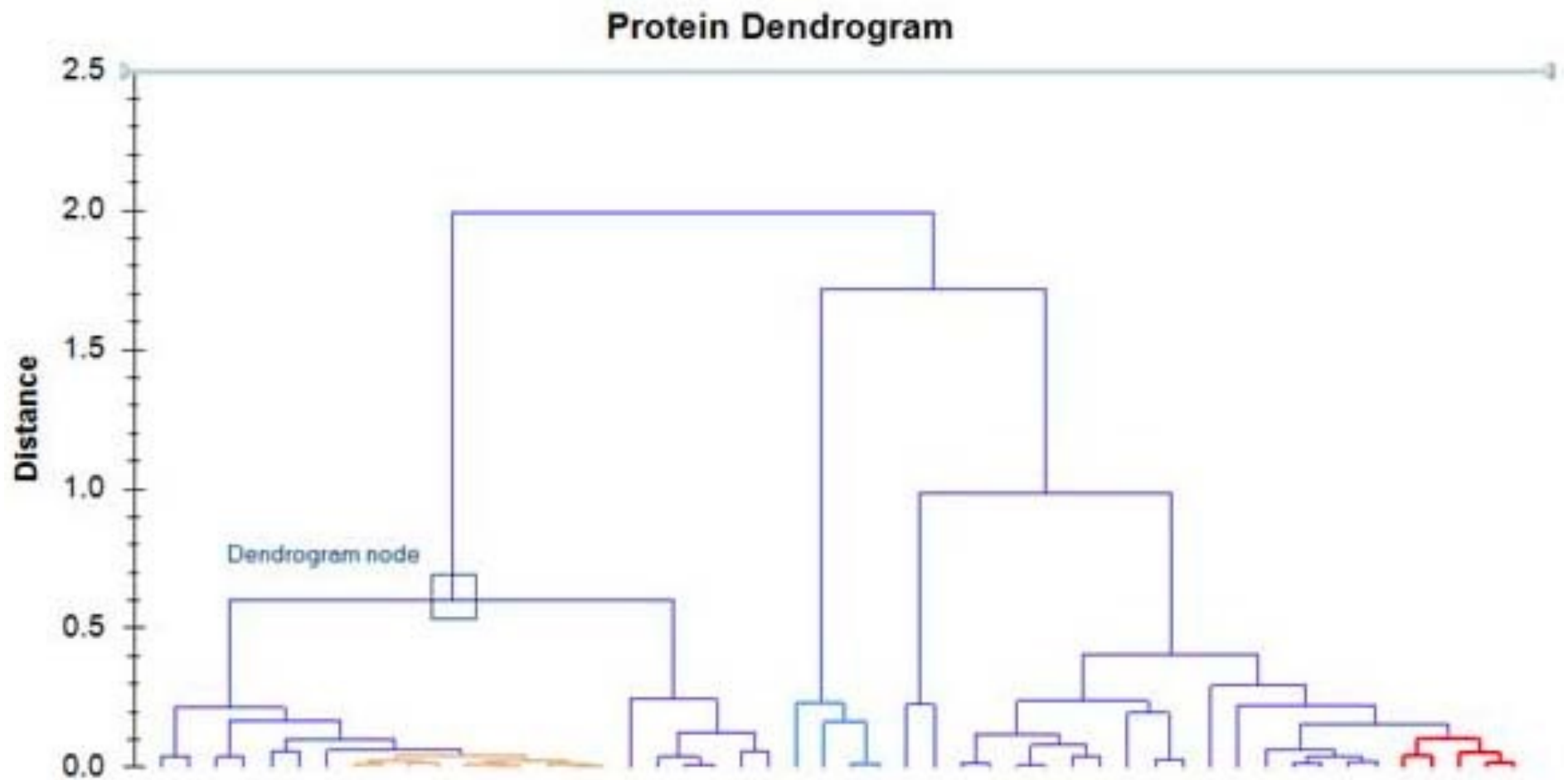
## Agglomerative clustering

- Start with each point as its own cluster and iteratively merge the closest clusters

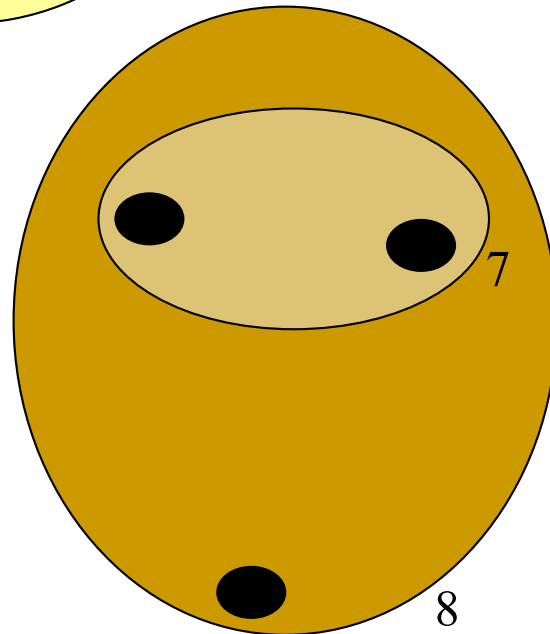
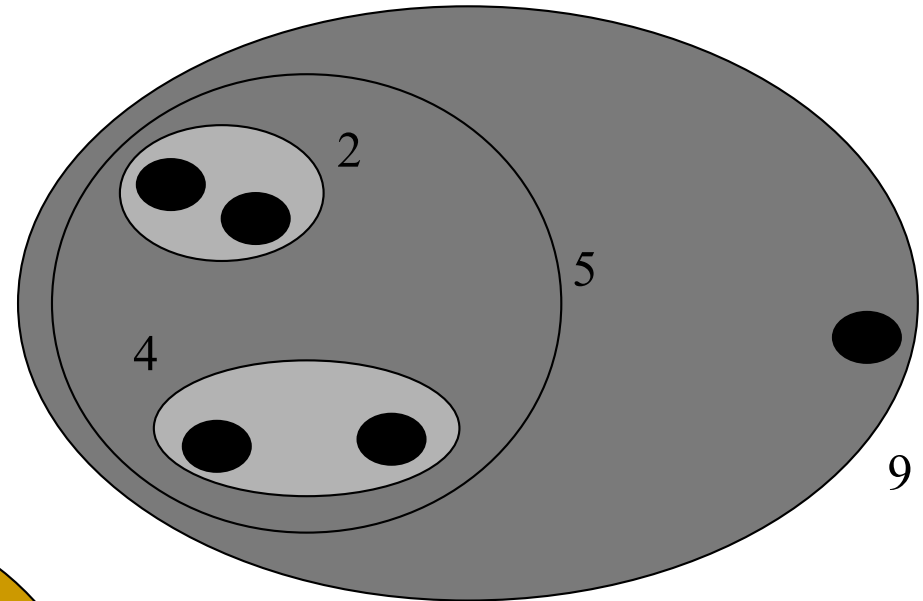
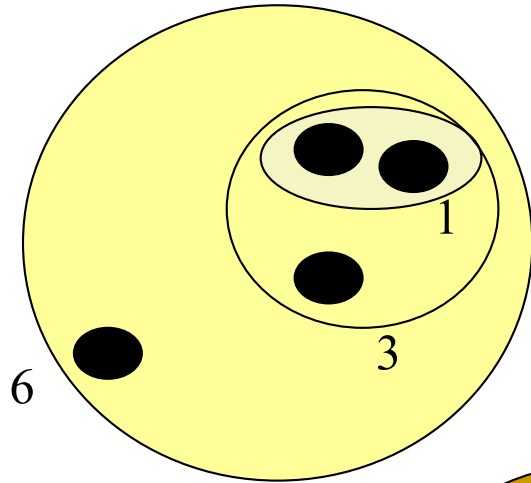
DBSCAN (Density-based spatial clustering of applications with noise)

EM Algorithm and Mixture Gaussian clustering

# DENDROGRAM EXAMPLE



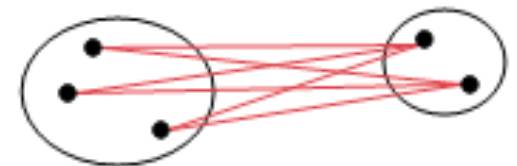
# Group Agglomerative Clustering



Which linkage scheme potentially yields long, skinny clusters?

Which yields compact clusters?

Average linkage



Complete linkage



Single linkage





# CLUSTERING STRATEGIES

## K-means

- Iteratively re-assign points to the nearest cluster center

## Agglomerative clustering

- Start with each point as its own cluster and iteratively merge the closest clusters

**DBSCAN** (Density-based spatial clustering of applications with noise)

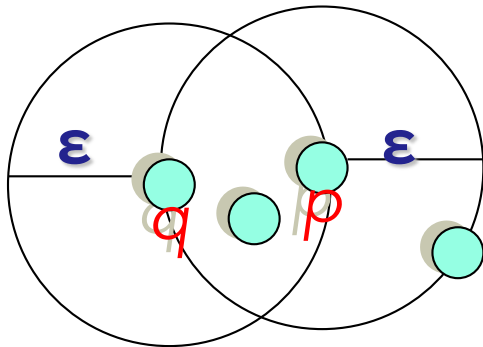
## EM Algorithm and Mixture Gaussian clustering

# $\epsilon$ -NEIGHBORHOOD

$\epsilon$ -Neighborhood – Objects within a radius of  $\epsilon$  from an object.

$$N_{\epsilon}(p) : \{q \mid d(p, q) \leq \epsilon\}$$

“High density” -  $\epsilon$ -Neighborhood of an object contains at least *MinPts* of objects.



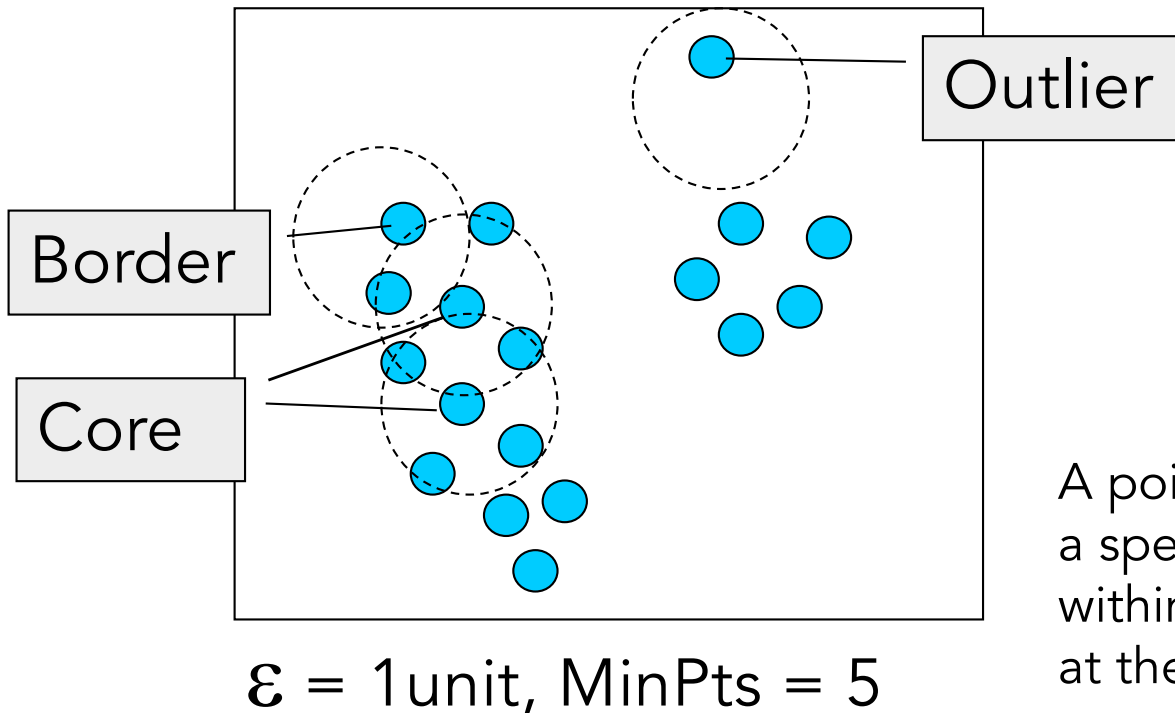
$\epsilon$ -Neighborhood of  $p$

$\epsilon$ -Neighborhood of  $q$

Density of  $p$  is “high” (MinPts = 4)

Density of  $q$  is “low” (MinPts = 4)

# CORE, BORDER & OUTLIER (NOISE)



Given  $\epsilon$  and *MinPts*, categorize the objects into three exclusive groups.

A point is a **core point** if it has more than a specified number of points (MinPts) within Epsilon. These are points that are at the interior of a cluster.

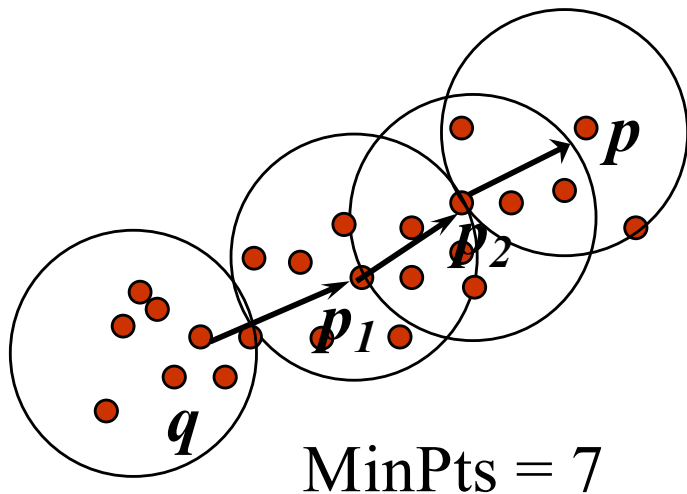
A **border point** has fewer than MinPts within Epsilon, but is in the neighborhood of a core point..

A **noise point (outlier)** is any point that is not a core point nor a border point.

# DENSITY-REACHABILITY

Density-Reachable (directly and indirectly):

- A point  $p$  is directly density-reachable from  $p_2$ ;
- $p_2$  is directly density-reachable from  $p_1$ ;
- $p_1$  is directly density-reachable from  $q$ ;
- $p \leftarrow p_2 \leftarrow p_1 \leftarrow q$  form a chain.



**$p$  is (indirectly) density-reachable from  $q$**

**$q$  is not density-reachable from  $p$ ?**

# DBSCAN ALGORITHM

Input: The data set D

Parameter:  $\varepsilon$ , MinPts

For each object p in D

    if p is a core object and not processed then

        C = retrieve all objects density-reachable from p

        mark all objects in C as processed

        report C as a cluster

    else mark p as outlier

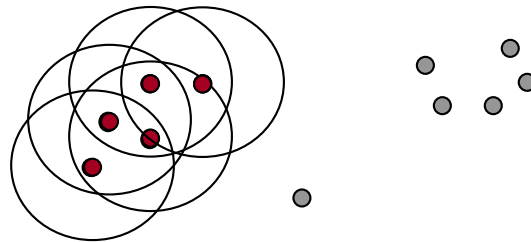
    end if

End For

# DBSCAN ALGORITHM: EXAMPLE

## Parameter

- $\varepsilon = 2$  cm
- $MinPts = 3$

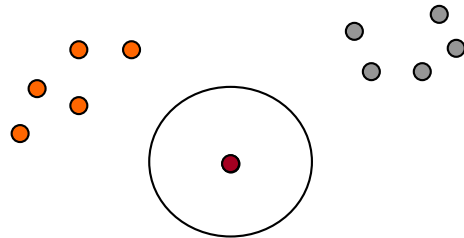


```
for each  $o \in D$  do  
  if  $o$  is not yet classified then  
    if  $o$  is a core-object then  
      collect all objects density-reachable from  $o$   
      and assign them to a new cluster.  
    else  
      assign  $o$  to NOISE
```

# DBSCAN ALGORITHM: EXAMPLE

## Parameter

- $\varepsilon = 2$  cm
- $MinPts = 3$

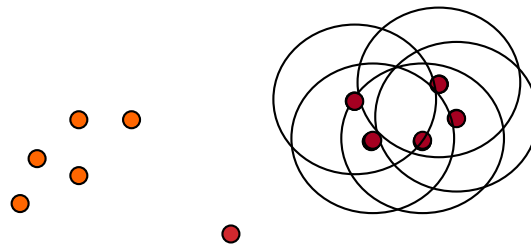


```
for each  $o \in D$  do  
  if  $o$  is not yet classified then  
    if  $o$  is a core-object then  
      collect all objects density-reachable from  $o$   
      and assign them to a new cluster.  
    else  
      assign  $o$  to NOISE
```

# DBSCAN ALGORITHM: EXAMPLE

## Parameter

- $\varepsilon = 2$  cm
- $MinPts = 3$



**for each  $o \in D$  do**

**if  $o$  is not yet classified then**

**if  $o$  is a core-object then**

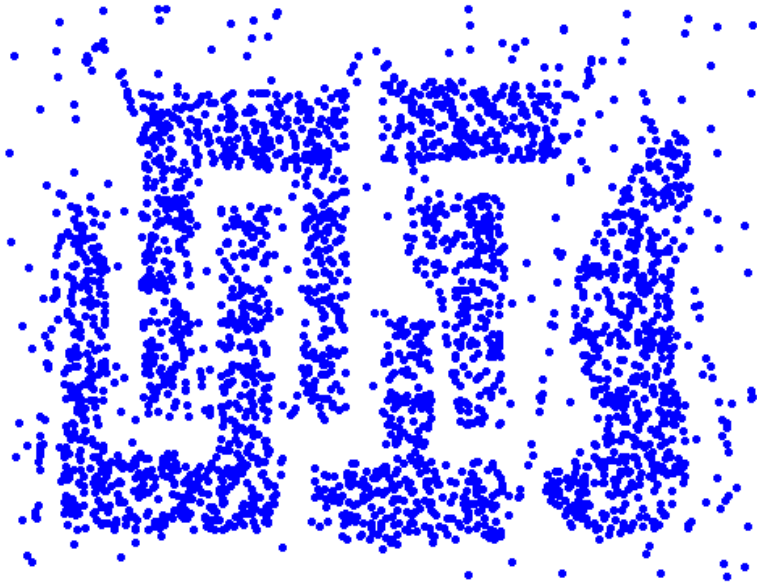
collect all objects density-reachable from  $o$   
and assign them to a new cluster.

**else**

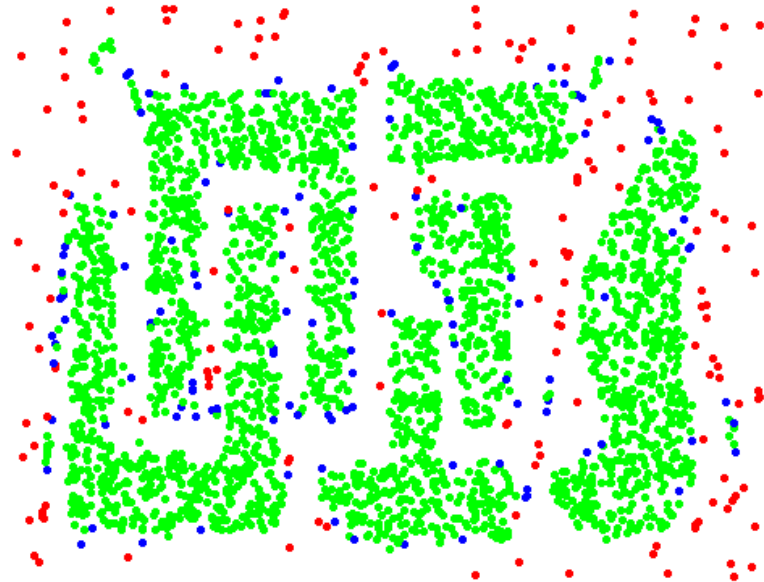
assign  $o$  to NOISE



# EXAMPLE



Original Points



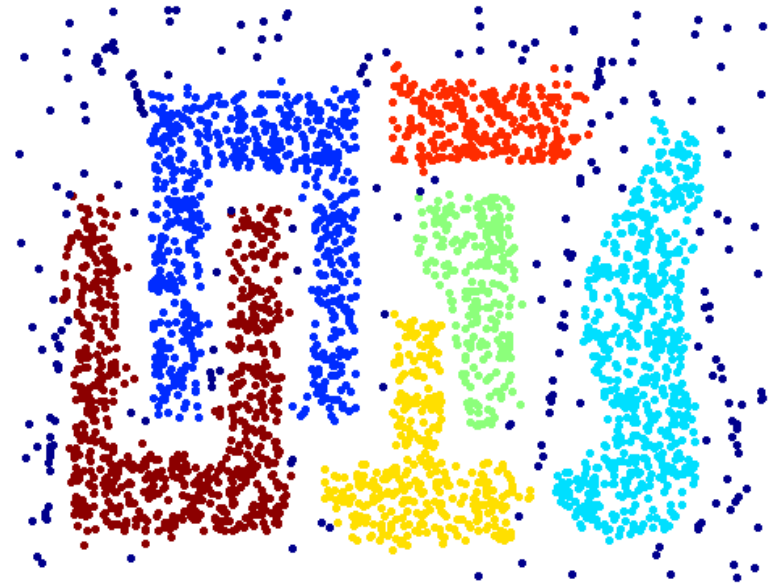
Point types: **core**,  
**border** and **outliers**

$\epsilon = 10$ , MinPts = 4

# WHEN DBSCAN WORKS WELL



Original Points



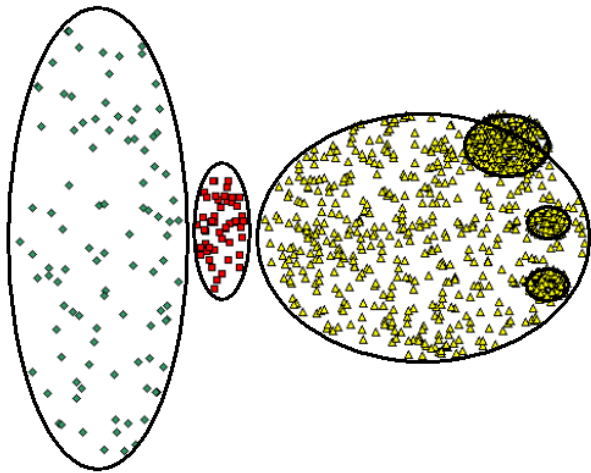
Clusters

- Resistant to Noise
- Can handle clusters of different shapes and sizes
- You don't need to specify the number of clusters in advance

CAN YOU CREATE AN EXAMPLE FOR  
WHICH DBSCAN WILL NOT WORK WELL?

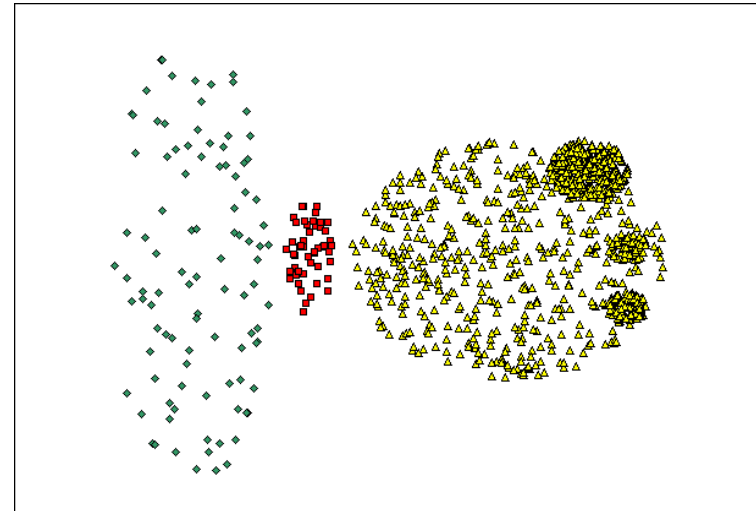


# WHEN DBSCAN DOES NOT WORK WELL

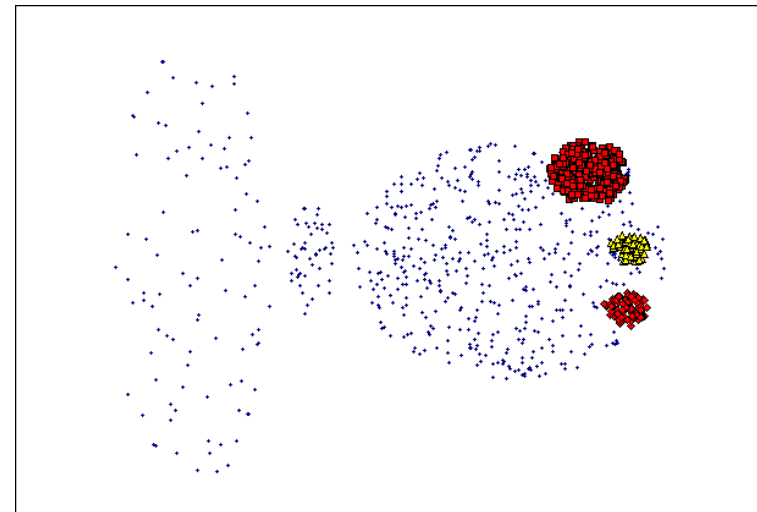


Original Points

- Cannot handle varying densities
- Sensitive to parameters

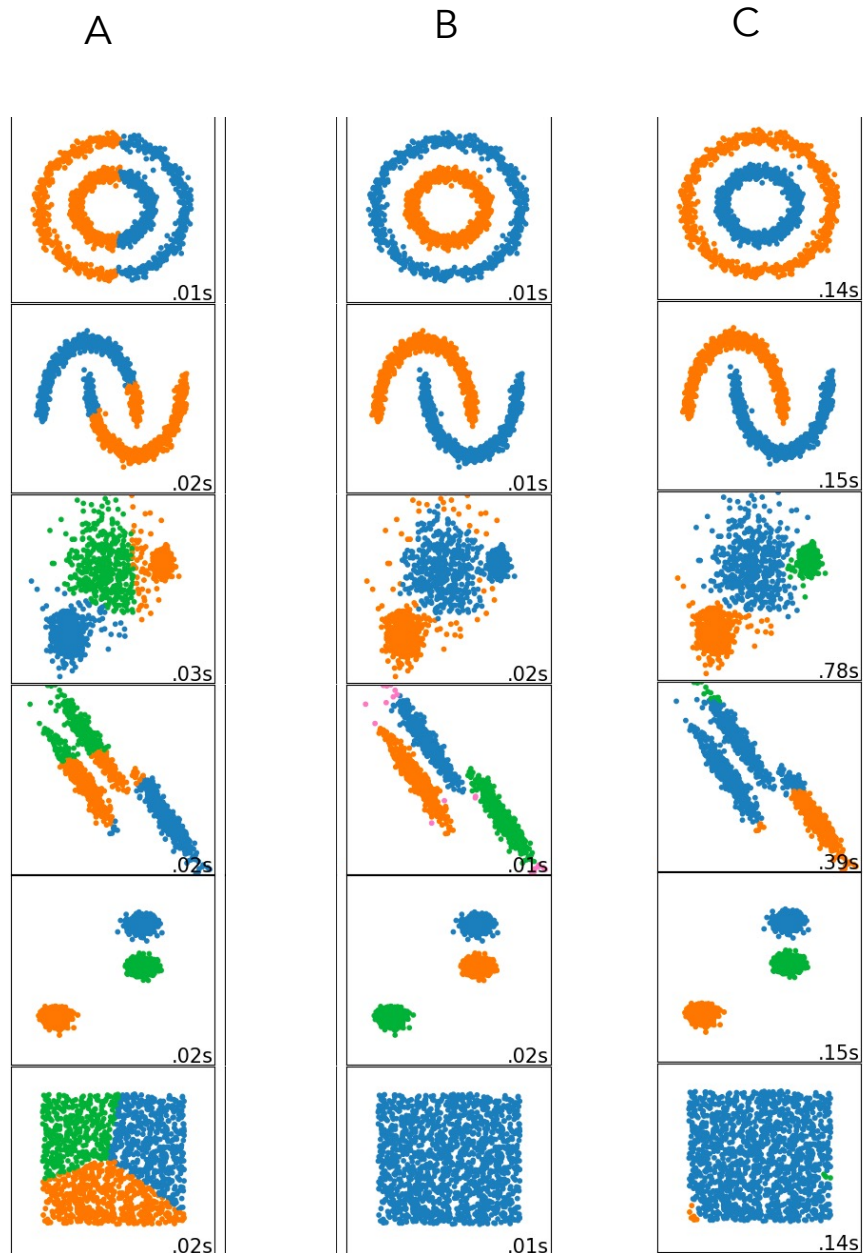


(MinPts=4, Eps=9.92).



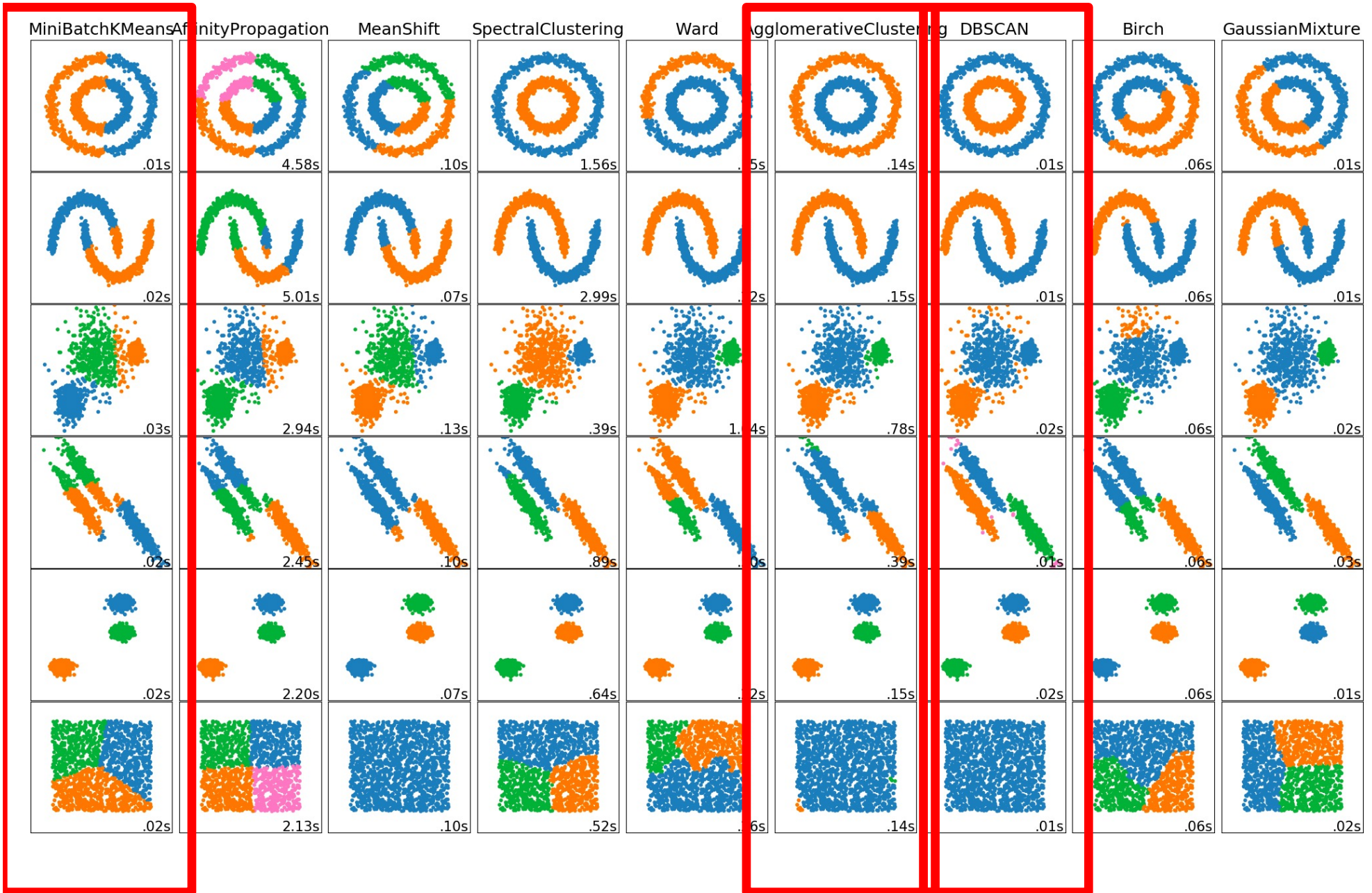
(MinPts=4, Eps=9.75)

# WHO WORE IT BEST?



A	KMeans	DBScan	Aggl. clustering
B	Aggl. clustering	KMeans	DBScan
C	KMeans	Aggl. clustering	DBScan

# CLUSTERING ANSWER: A WINS





# CLUSTERING STRATEGIES

## K-means

- Iteratively re-assign points to the nearest cluster center

## Agglomerative clustering

- Start with each point as its own cluster and iteratively merge the closest clusters

DBSCAN (Density-based spatial clustering of applications with noise)

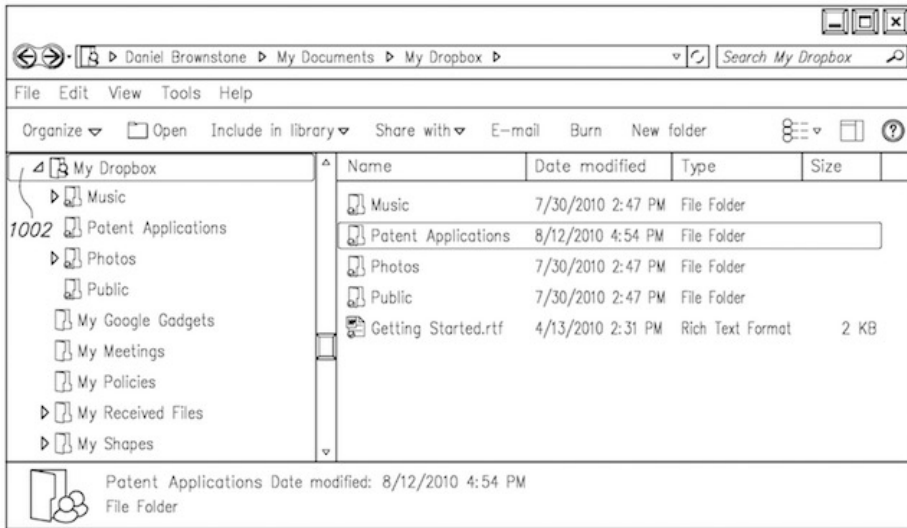
## EM Algorithm and Mixture Gaussian clustering

# Motivational Example

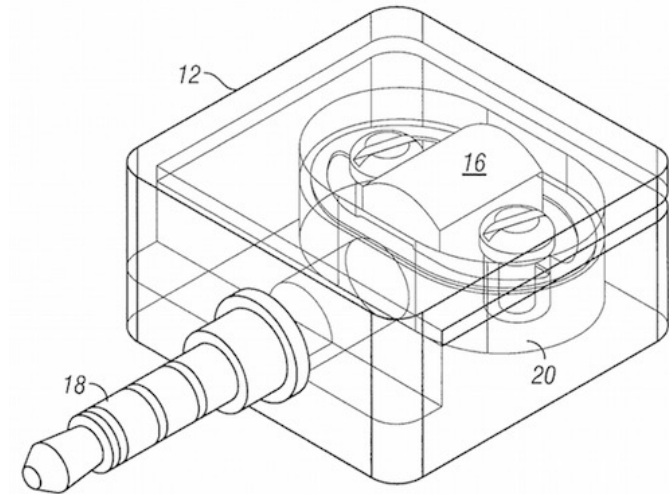


Around **300.000**  
US Patent Applications  
Granted **per Year**





Network folder synchronization (DropBox)



Systems and methods for decoding card swipe signals (Square)

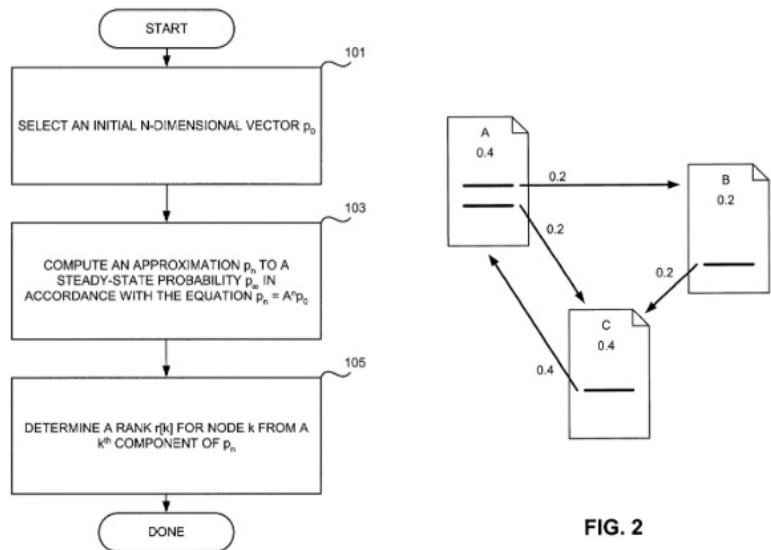


FIG. 2

Method for node ranking in a linked database (Google)

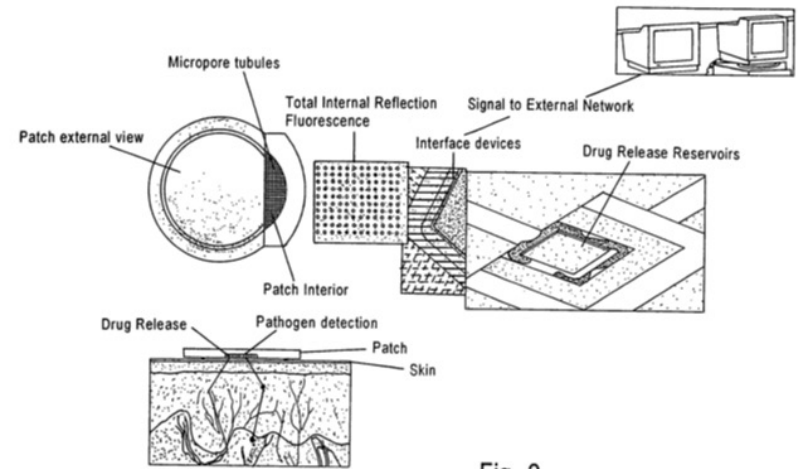
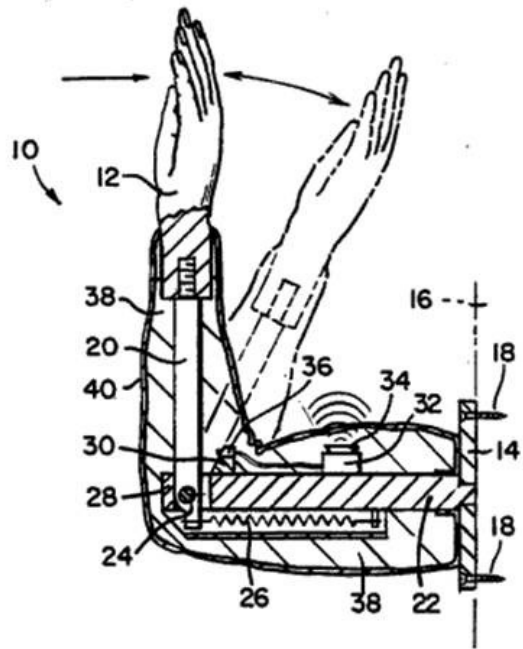


Fig. 2

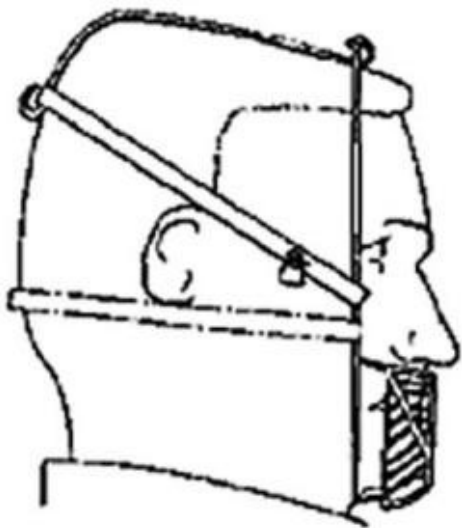
Medical device for analyte monitoring and drug delivery (Theranos)



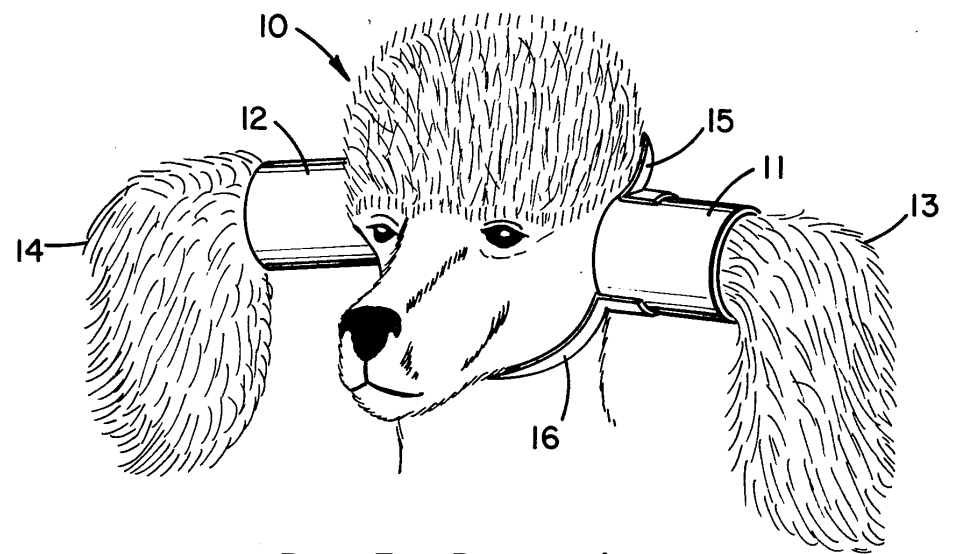
High-Five Machine



Gerbil Shirt



Anti Eating Device



Dog Ear Protection





*Ein Stück Gesundheit, dessen Erhaltung mehr als wichtig für Sie ist.*

Um Ihre Zähne geht es hier, von denen es abhängt, ob Ihnen Essen, Lachen, Sprechen immer eine Freude sein werden, ob Ihr Mund und Ihr Gesicht ihr glattes, gepflegtes Aussehen behalten, ob Ihre Kaukraft erhalten bleibt, die bekanntlich eine wichtige Rolle für die Verdauung spielt.

*Ein hohler Zahn ist Warnung genug!*

Ihm fehlte die Zufuhr notwendiger Aufbaustoffe und Abwehrkräfte. Darum ist er erkrankt. Heute geht es dem einen Zahn so. Ein Jahr später aber vielleicht vielen! Schützen Sie sich durch Pflege mit der biologisch wirksamen, radioaktiven „Doramad-Zahncreme“. Durch ihre feine radioaktive Strahlung - welche noch lange nach dem Putzen das Zahnfleisch massiert - werden Zellstoffwechsel, Nahrungszufuhr und Abwehrkräfte wesentlich gesteigert und angreifende Krankheitserreger vernichtet.



*Leiden Sie unter Zahnfleischbluten, krankem Zahnfleisch oder Zahnlockerung?*

Dann benutzen Sie „Doramad“ erst recht. Das Zahnfleisch blutet bald nicht mehr beim Bürsten, es wird straff und bekommt gesunde, schöne Farbe. Eiterungen verschwinden und lockere Zähne festigen sich häufig wieder, wenn es nicht zu spät ist und nur der Facharzt helfen kann. Zur Vorbeugung gegen das Entstehen derartiger Erkrankungen sollte jeder „Doramad“ benutzen. — „Doramad“ ist radioaktiv — Wissenschaftliche Zusammensetzung und edelste Rohstoffe geben ihr aber noch weitere Vorteile. Die 5 Zahnpfleger der „Doramad“ sagen sie Ihnen rückseitig.



Genau wie im Körper überall herrscht auch in der Mundhöhle, dem Einfallstor für viele Krankheitserreger, ein fortwährender Kampf zwischen den natürlichen Abwehrkräften und den eingedrungenen schädlichen Bakterien. Diese Krankheitserreger können auf natürlichem — biologischem — Wege erfolgreich bekämpft werden, weil „Doramad“ die Abwehrkräfte des Organismus unterstützt.

Note, that I could not actually verify them as real patents, but they could easily be some

Goal:

We want to build a  
**model** to automatically  
**classify patents** into  
useful or bogus?

# What do we need?

1. The patent data (easy thanks to Google Patents)
2. A training data set:  
some pre-labeled patents
3. A model

# What do we need?

1. The patent data (easy thanks to Google Patents)
2. A training data set:  
some pre-labeled patents
3. A model

How do we get a labeled data set?

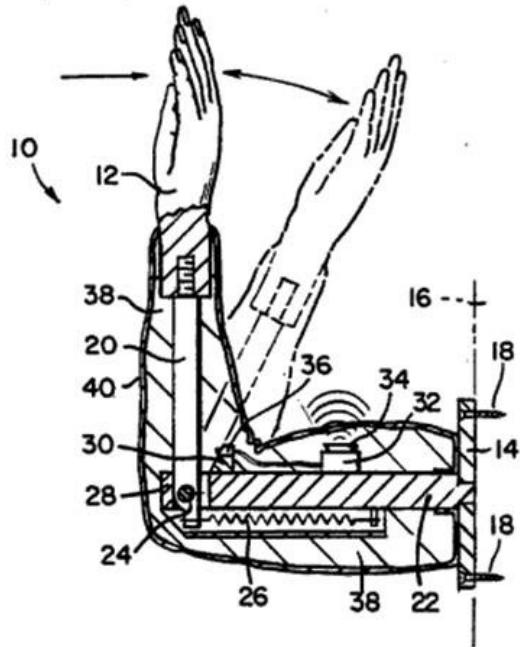
How do we get a labeled data set?





# A Crowd Task

Is this Patent Bogus?



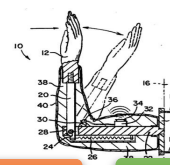
$l = 1$

Yes

No

# A Crowd Task

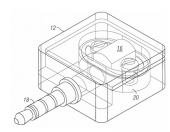
Is this Patent Bogus?



Yes No

$O_1$

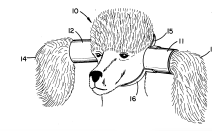
Is this Patent Bogus?



Yes No

$O_2$


Is this Patent Bogus?



Yes No

$O_3$

Is this Patent Bogus?



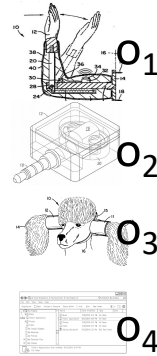
Yes No

$O_4$

⋮

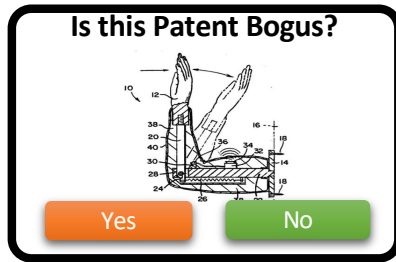


$$l[n] =$$

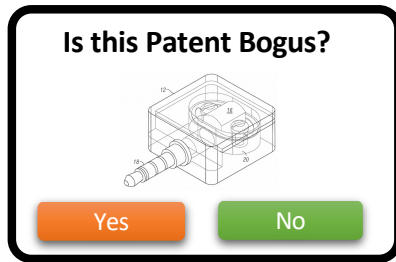


$$\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ \vdots \end{pmatrix}$$

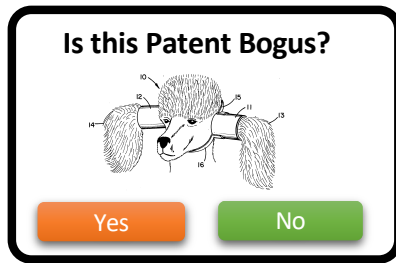
# A Crowd Task



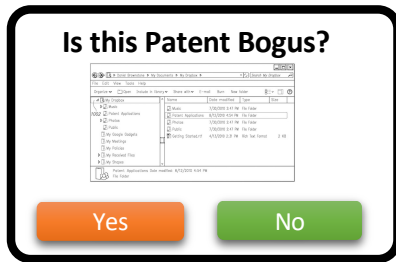
$O_1$



$O_2$

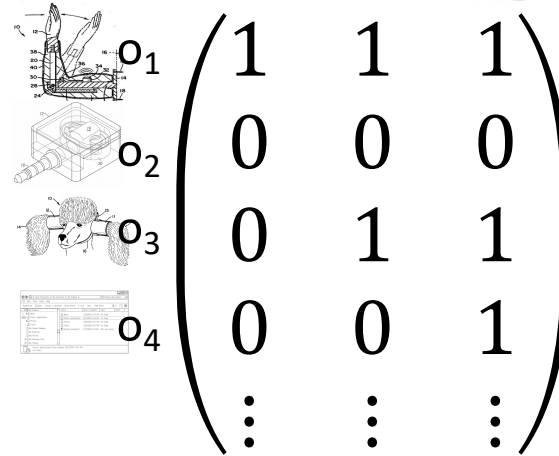
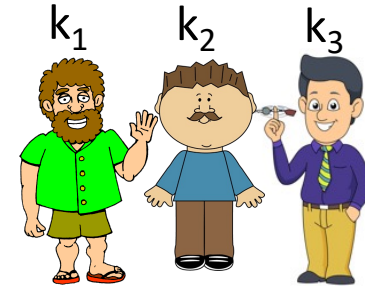


$O_3$



$O_4$

⋮



$$l[k][n] =$$

# What should the final labels be?

$$l[k][n] = \begin{matrix} & \begin{matrix} k_1 & k_2 & k_3 \end{matrix} \\ \begin{matrix} \text{img}_1 & \text{img}_2 & \text{img}_3 \\ \text{img}_4 & \text{img}_5 & \text{img}_6 \\ \vdots & \vdots & \vdots \end{matrix} & \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \\ \vdots & \vdots & \vdots \end{pmatrix} \end{matrix}$$

$$T(n) = \begin{matrix} \text{img}_7 & o_1 \\ \text{img}_8 & o_2 \\ \text{img}_9 & o_3 \\ \text{img}_{10} & o_4 \\ \vdots & \vdots \end{matrix} \begin{pmatrix} ? \\ ? \\ ? \\ ? \\ \vdots \end{pmatrix}$$

# Maximum Likelihood Estimate

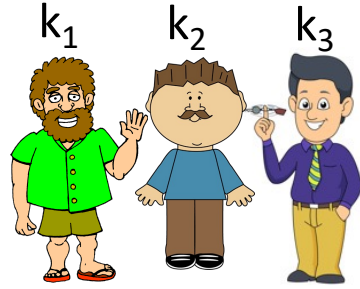
- Given some data  $X = (x_1, \dots, x_n)$

- Model  $\mathcal{L}(\theta, X) = p_\theta(X) = \prod_i^n p_\theta(x_i)$

- **Maximum Likelihood Estimator (MLE)**

$$\hat{\theta} = \operatorname{argmax}_{\theta \in \Theta} \mathcal{L}(\theta, X)$$

# A Maximum Likelihood Estimate (MLE)



$$l[k][n] = \begin{matrix} & & \begin{matrix} k_1 & k_2 & k_3 \end{matrix} \\ \begin{matrix} o_1 \\ o_2 \\ o_3 \\ o_4 \\ \vdots \end{matrix} & \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \\ \vdots & \vdots & \vdots \end{pmatrix} \end{matrix}$$

What should the final labels be?

$$T(n) = \begin{matrix} & \begin{matrix} o_1 \\ o_2 \\ o_3 \\ o_4 \\ \vdots \end{matrix} & \begin{pmatrix} ? \\ ? \\ ? \\ ? \\ \vdots \end{pmatrix} \end{matrix}$$

**BOGUS**

**Not  
BOGUS**

$$\begin{pmatrix} 3/3 & 0/3 \\ 0/3 & 3/3 \\ 2/3 & 1/3 \\ 1/3 & 2/3 \\ \vdots & \vdots \end{pmatrix}$$

# A Maximum Likelihood Estimate (MLE)

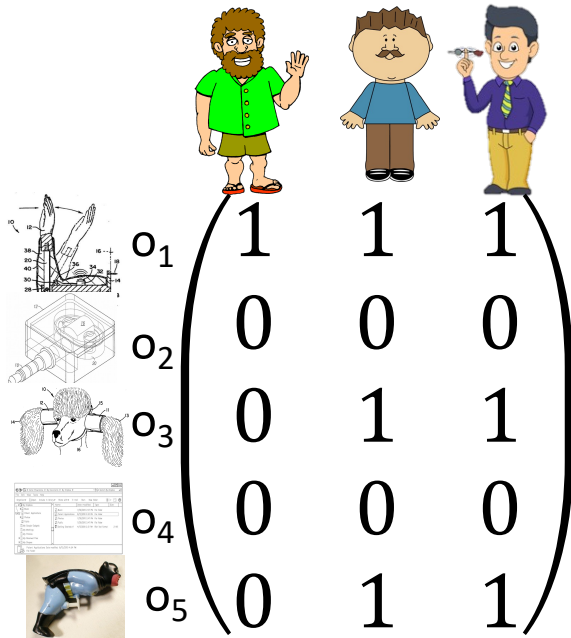
$$l[k][n] = \begin{matrix} & \begin{matrix} k_1 & k_2 & k_3 \end{matrix} \\ \begin{matrix} \text{Person 1} \\ \text{Person 2} \\ \text{Person 3} \end{matrix} & \begin{pmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 1 \\ \vdots & \vdots & \vdots \end{pmatrix} \end{matrix}$$

What should the final labels be?

$$T(n) = \begin{matrix} \text{Image 1} \\ \text{Image 2} \\ \text{Image 3} \\ \text{Image 4} \end{matrix} \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ \vdots \end{pmatrix}$$

BOGUS	Not BOGUS
3/3	0/3
0/3	3/3
2/3	1/3
1/3	2/3
⋮	⋮

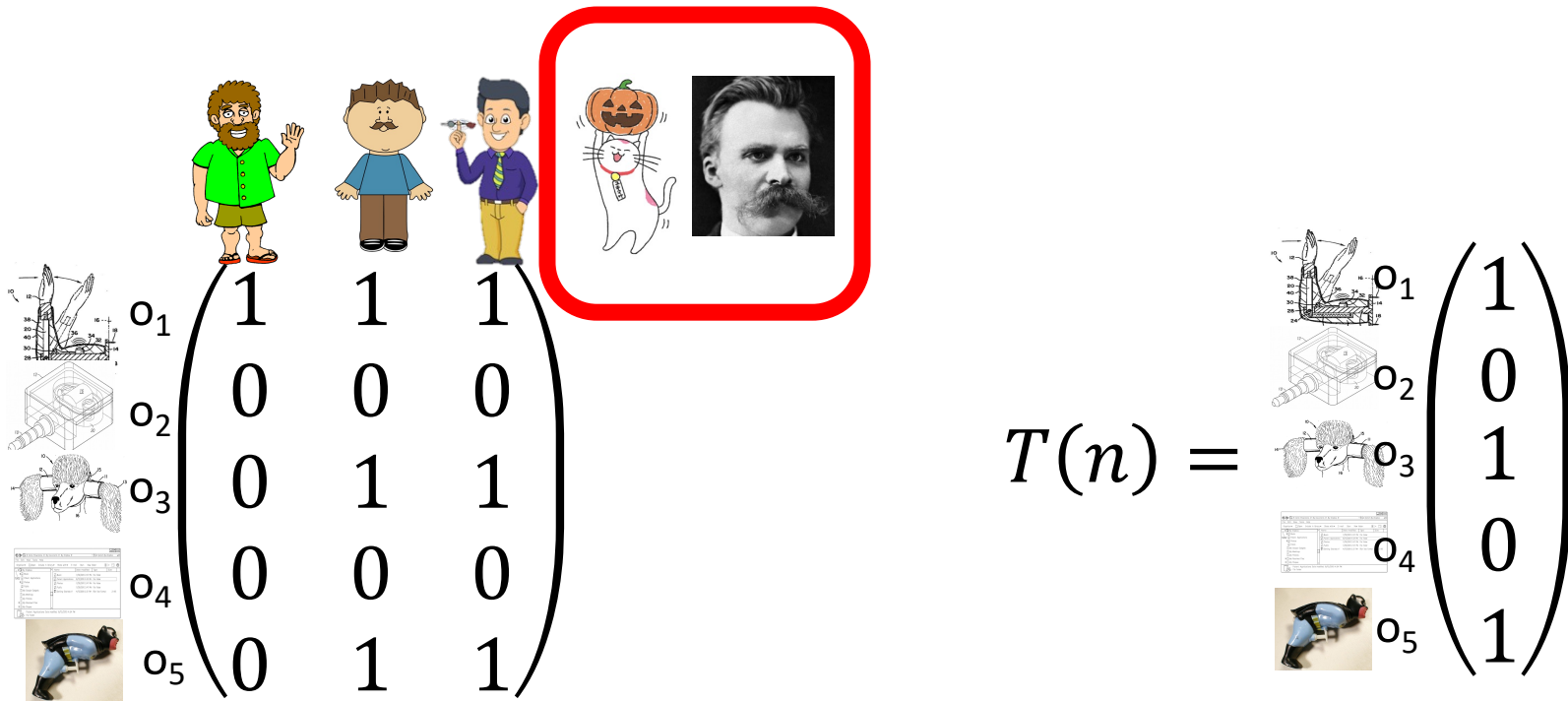
# So Everything is Good



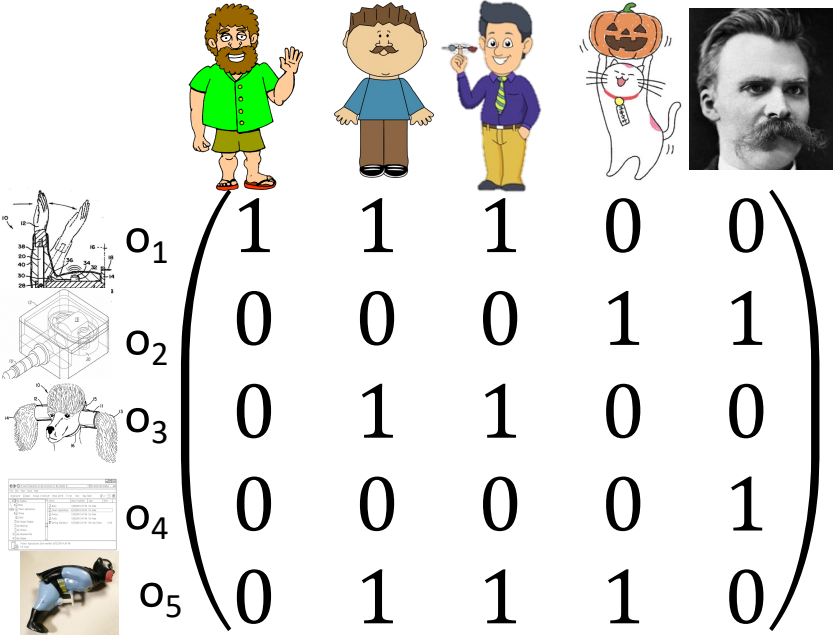
$$T(n) = \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}$$



# But what happens if we add Crazy Cat with Pumpkin and the Nihilist?



# What if the Workers do not have the same Quality?

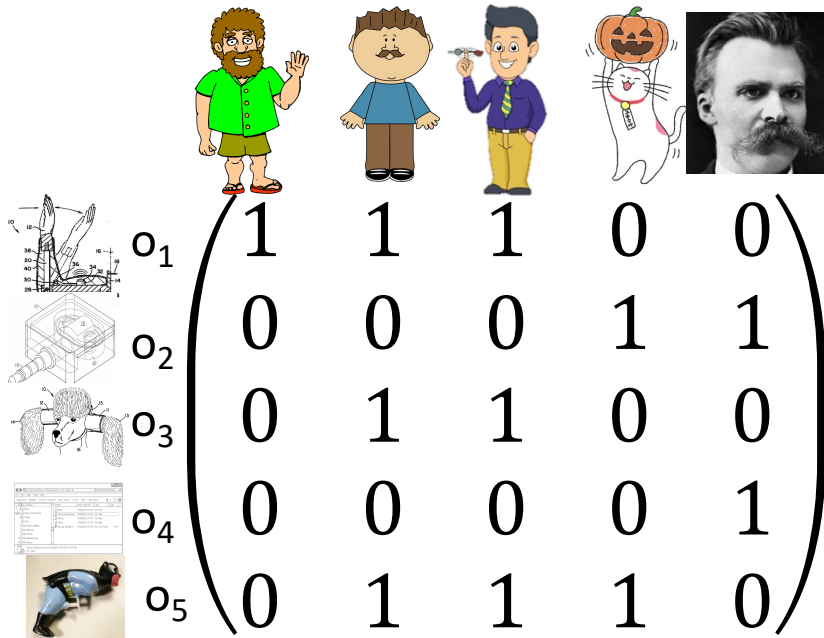
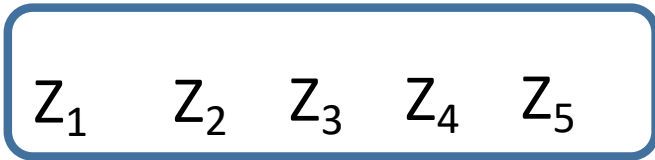


$$T(n) = \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

The diagram shows five tasks and a column vector of 1s and 0s. The tasks are: a technical drawing of a wing, a technical drawing of a box, a technical drawing of a propeller, a screenshot of a software interface, and a photo of a person in a blue uniform. The vector is: 1, 0, 0, 0, 1. The third element (0) is highlighted in red.

# What if the Workers do not have the same Quality?

Latent (hidden)  
Variables



$$T(n) = \begin{matrix} \begin{matrix} \text{Diagram 1} \\ \text{Diagram 2} \\ \text{Diagram 3} \\ \text{Diagram 4} \\ \text{Diagram 5} \end{matrix} \begin{matrix} o_1 \\ o_2 \\ o_3 \\ o_4 \\ o_5 \end{matrix} \end{matrix} \begin{pmatrix} ? \\ ? \\ ? \\ ? \\ ? \end{pmatrix}$$

# Maximum Likelihood Estimate

- Given some data  $X = (x_1, \dots, x_n)$

- Model  $\mathcal{L}(\theta, X) = p_\theta(X) = \prod_i^n p_\theta(x_i)$

- **Maximum Likelihood Estimator (MLE)**

$$\hat{\theta} = \operatorname{argmax}_{\theta \in \Theta} \mathcal{L}(\theta, X)$$

# Maximum Likelihood Estimate

- Given some data  $X = (x_1, \dots, x_n)$

- Model  $\mathcal{L}(\theta, X, Z) = p_\theta(X, Z) = \prod_i^n p_\theta(x_i, z)$

- **Maximum Likelihood Estimator (MLE)**

$$\hat{\theta} = \operatorname{argmax}_{\theta} \mathcal{L}(\theta, X) = \sum_z p_\theta(X, Z)$$

- Z has been marginalized
- Hard to compute

# Expectation Maximization Algorithm

Initialize  $\theta \in \Theta$

For  $t = 0, 1, 2, \dots$

E-Step: Calculate the expected value of the log likelihood function, with respect to the conditional distribution of  $Z$  given  $X$  under the current estimate of the parameters  $\theta_t$ :

$$Q(Q|\theta_t) = E_{Z|X, \theta_t}[\log \mathcal{L}(\theta, X, Z)]$$

M-Step: Find the parameter that maximizes this quantity

$$\theta_{t+1} = \operatorname{argmax}_{\theta} Q(Q|\theta_t)$$

# EM – In our Example

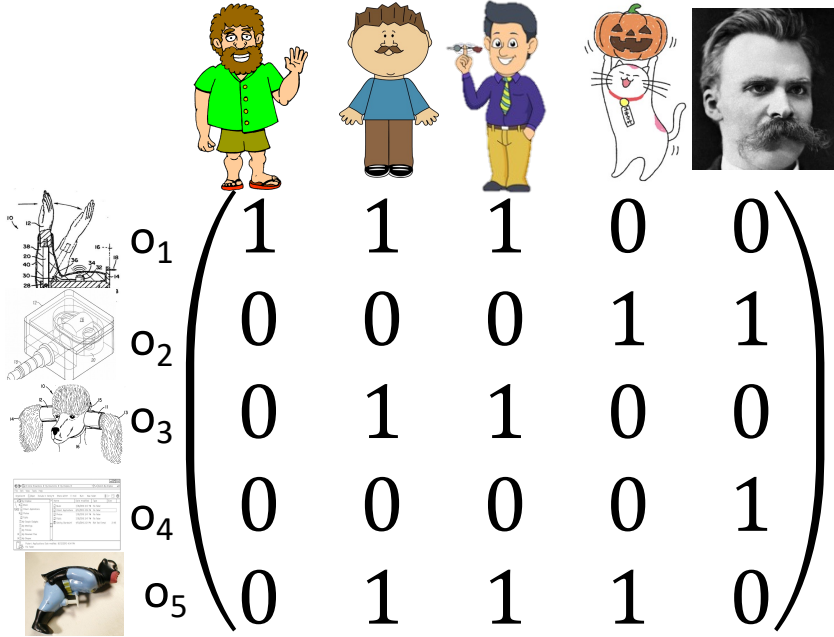
Initialize  $\theta_0$

For  $t = 0, 1, 2, \dots$

E-Step: Calculate the expected labels  
(e.g., bogus or not-bogus)  
given  $\theta_t$

M-Step: Given the estimated label,  
optimize  $\theta$  and set it to  $\theta_{t+1}$

# EM Algorithm - Example



		Guess	
		Bogus	!Bogus
True	Bogus	1	0
	!Bogus	0	1

		Guess	
		Bogus	!Bogus
True	Bogus	1	0
	!Bogus	0	1

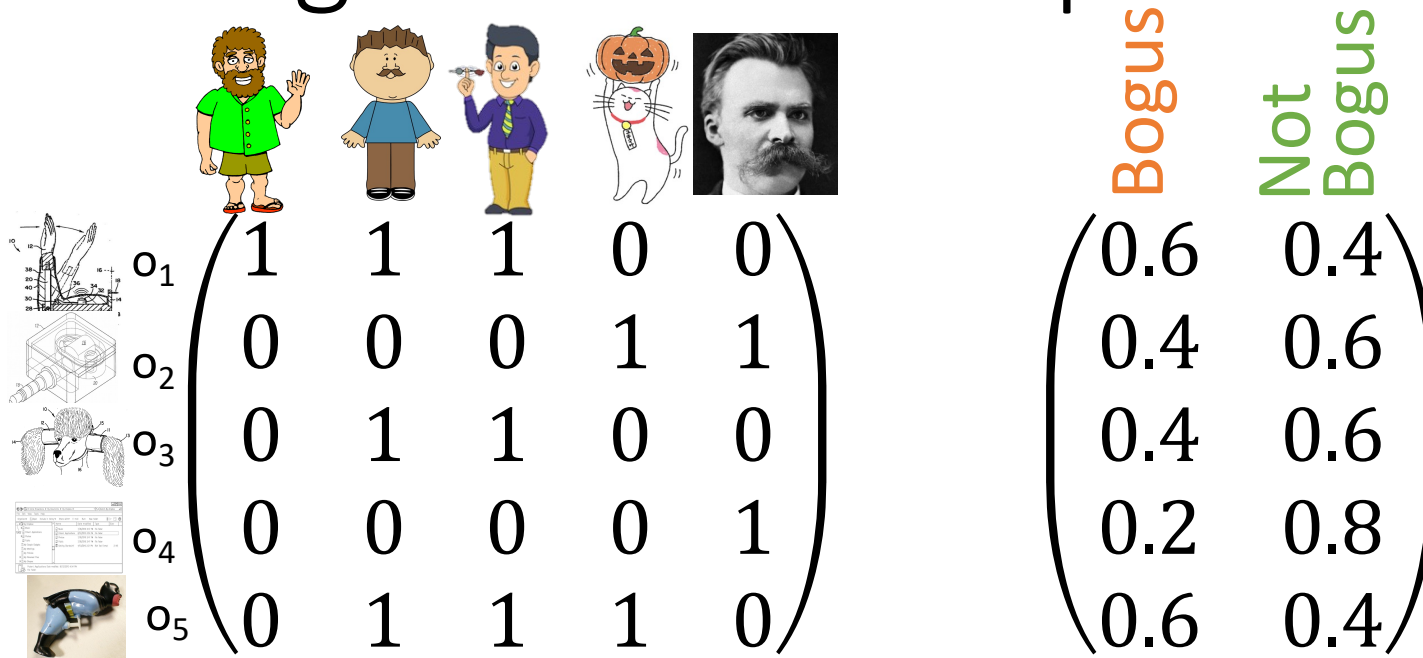
		Guess	
		Bogus	!Bogus
True	Bogus	1	0
	!Bogus	0	1


		Guess	
		Bogus	!Bogus
True	Bogus	1	0
	!Bogus	0	1


		Guess	
		Bogus	!Bogus
True	Bogus	1	0
	!Bogus	0	1

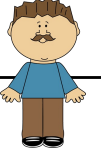



# EM Algorithm - Example

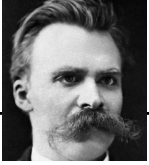


		Guess	
		Bogus	!Bogus
True	Bogus	1	0
	!Bogus	0	1


		Guess	
		Bogus	!Bogus
True	Bogus	1	0
	!Bogus	0	1

		Guess	
		Bogus	!Bogus
True	Bogus	1	0
	!Bogus	0	1


		Guess	
		Bogus	!Bogus
True	Bogus	1	0
	!Bogus	0	1

		Guess	
		Bogus	!Bogus
True	Bogus	1	0
	!Bogus	0	1

# EM Algorithm - Example




$$\begin{matrix}
 o_1 \\
 o_2 \\
 o_3 \\
 o_4 \\
 o_5
 \end{matrix}
 \begin{pmatrix}
 1 & 1 & 1 & 0 & 0 \\
 0 & 0 & 0 & 1 & 1 \\
 0 & 1 & 1 & 0 & 0 \\
 0 & 0 & 0 & 0 & 1 \\
 0 & 1 & 1 & 1 & 0
 \end{pmatrix}
 \begin{matrix}
 \text{Bogus} \\
 \text{Not Bogus}
 \end{matrix}
 \begin{pmatrix}
 0.6 & 0.4 \\
 0.4 & 0.6 \\
 0.4 & 0.6 \\
 0.2 & 0.8 \\
 0.6 & 0.4
 \end{pmatrix}
 \begin{matrix}
 \text{“Correct”} \\
 \text{Labels}
 \end{matrix}
 \begin{pmatrix}
 1 \\
 0 \\
 0 \\
 0 \\
 1
 \end{pmatrix}$$



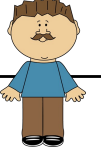
Guess

	Bogus	!Bogus
True Bogus	1	0
True !Bogus	0	1




Guess

	Bogus	!Bogus
True Bogus	1	0
True !Bogus	0	1



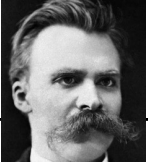
Guess

	Bogus	!Bogus
True Bogus	1	0
True !Bogus	0	1



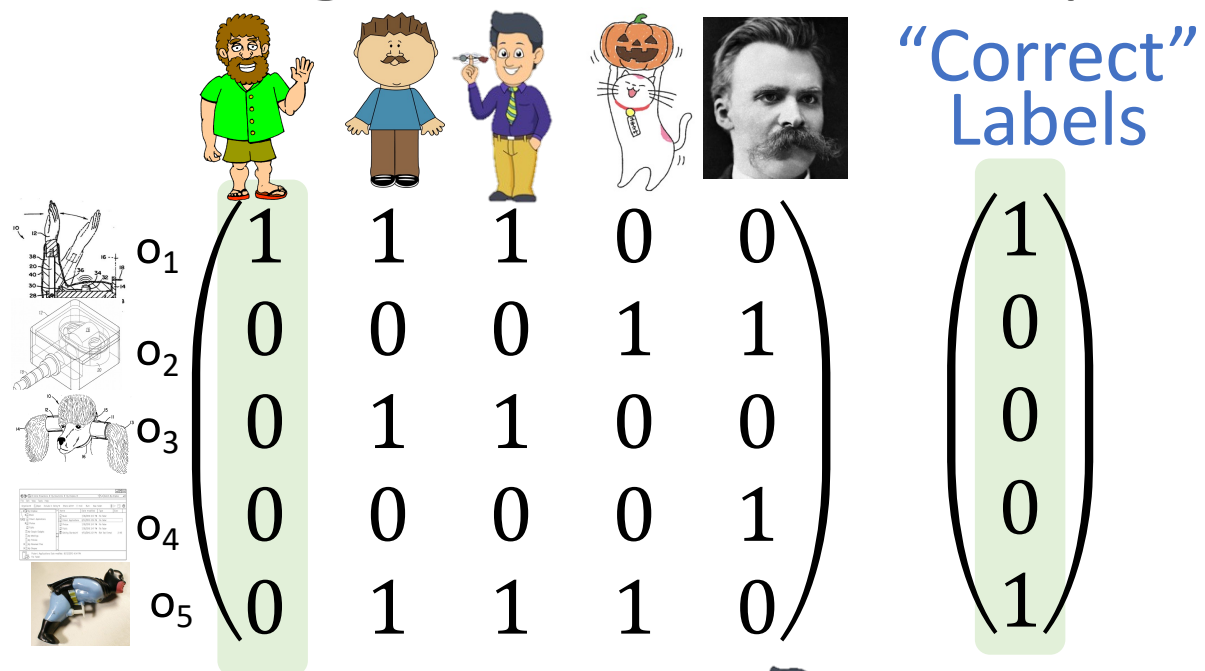
Guess

	Bogus	!Bogus
True Bogus	1	0
True !Bogus	0	1



	Bogus	!Bogus
Bogus	1	0
!Bogus	0	1

# EM Algorithm - Example



Guess

	Bogus	!Bogus
True Bogus	?	?
True !Bogus	?	?

Guess

	Bogus	!Bogus
True Bogus	?	?
True !Bogus	?	?

Guess

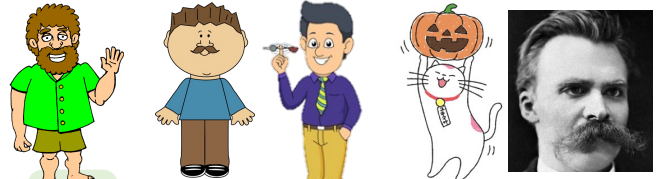
	Bogus	!Bogus
True Bogus	?	?
True !Bogus	?	?

Guess

	Bogus	!Bogus
True Bogus	?	?
True !Bogus	?	?

	Bogus	!Bogus
Bogus	?	?
!Bogus	?	?


# EM Algorithm - Example



“Correct” Labels


$o_1$	1	1	1	0	0	1
$o_2$	0	0	0	1	1	0
$o_3$	0	1	1	0	0	0
$o_4$	0	0	0	0	1	0
$o_5$	0	1	1	1	0	1

Guess

	Bogus	!Bogus
Bogus	1	0.25
!Bogus	0	0.75

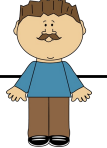
True

Guess

	Bogus	!Bogus
Bogus	?	?
!Bogus	?	?


True

Guess

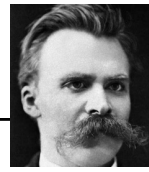
	Bogus	!Bogus
Bogus	?	?
!Bogus	?	?

True

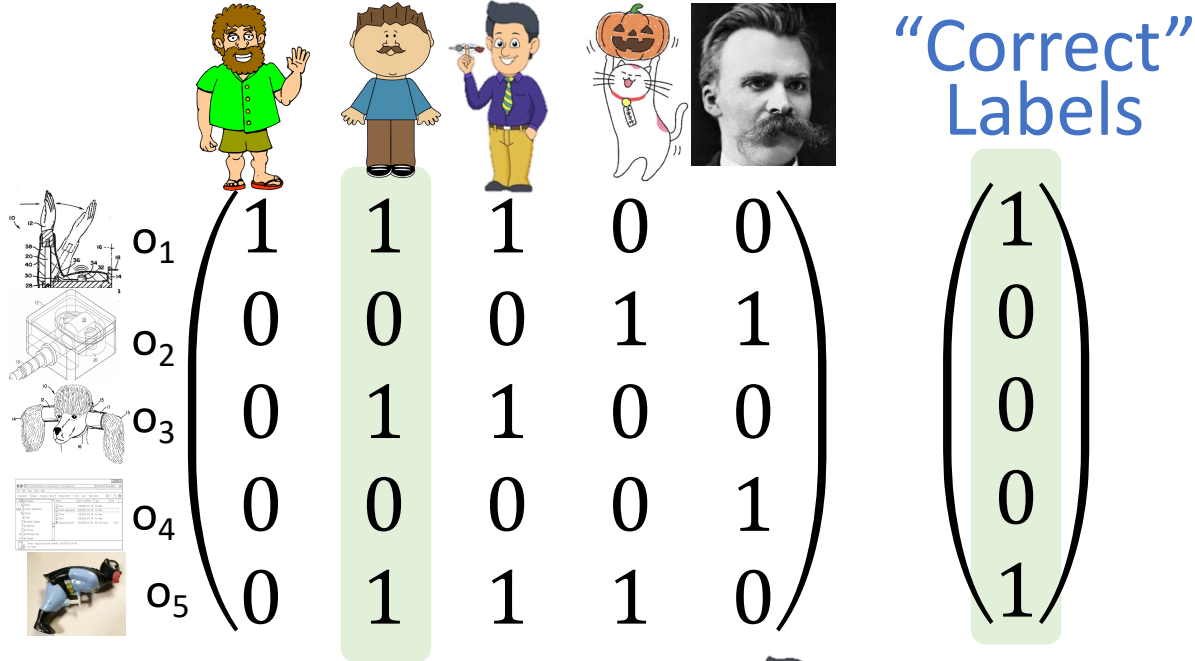
Guess

	Bogus	!Bogus
Bogus	?	?
!Bogus	?	?


True

	Bogus	!Bogus
Bogus	?	?
!Bogus	?	?


# EM Algorithm - Example



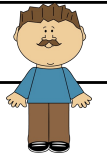
Guess

	Bogus	!Bogus
True Bogus	1	0.25
True !Bogus	0	0.75


Guess

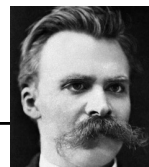
	Bogus	!Bogus
True Bogus	?	?
True !Bogus	?	?

Guess

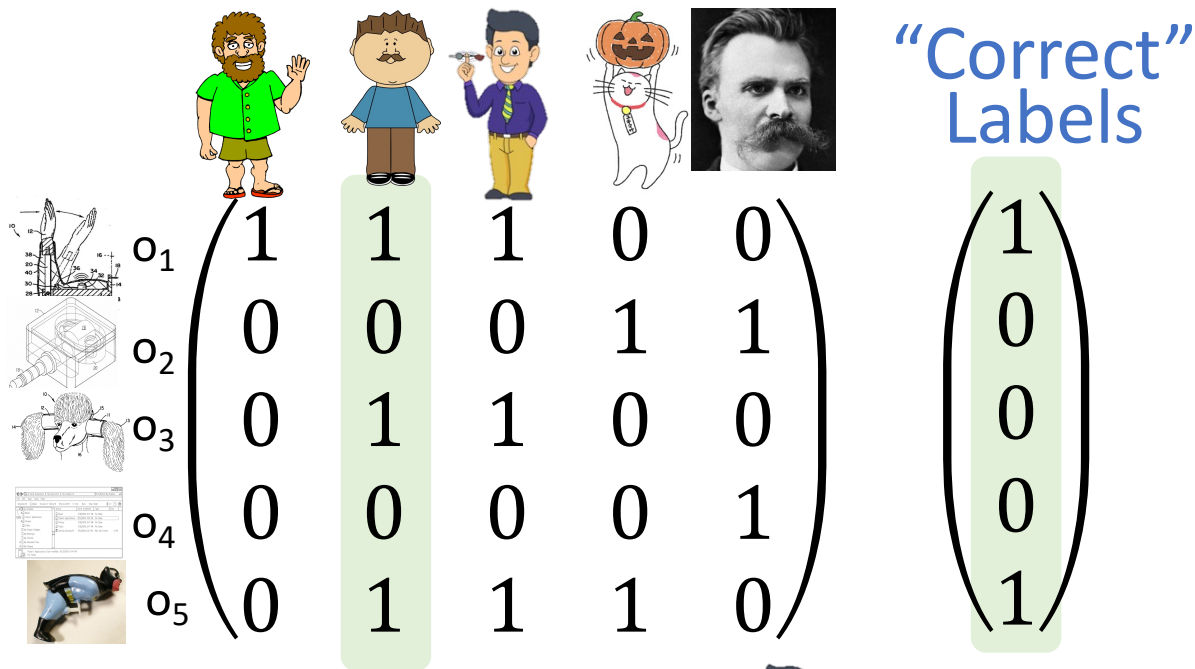
	Bogus	!Bogus
True Bogus	?	?
True !Bogus	?	?

Guess

	Bogus	!Bogus
True Bogus	?	?
True !Bogus	?	?

	Bogus	!Bogus
Bogus	?	?
!Bogus	?	?

# EM Algorithm - Example



Guess

	Bogus	!Bogus
True Bogus	1	0.25
True !Bogus	0	0.75

Guess

	Bogus	!Bogus
True Bogus	?	?
True !Bogus	?	?

Guess

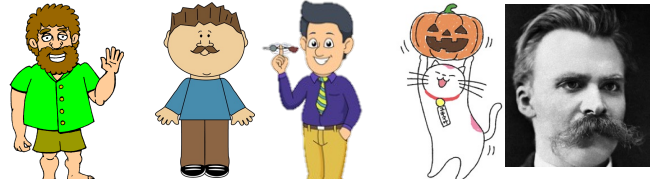
	Bogus	!Bogus
True Bogus	0.66	0
True !Bogus	0.33	1

Guess

	Bogus	!Bogus
True Bogus	?	?
True !Bogus	?	?


	Bogus	!Bogus
Bogus	?	?
!Bogus	?	?

# EM Algorithm - Example




"Correct"  
Labels

$o_1$	1	1	1	0	0	$\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$
$o_2$	0	0	0	1	1	
$o_3$	0	1	1	0	0	
$o_4$	0	0	0	0	1	
$o_5$	0	1	1	1	0	



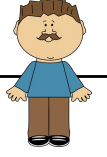
Guess

		Bogus	!Bogus
True	Bogus	1	0.25
	!Bogus	0	.75




Guess

		Bogus	!Bogus
True	Bogus	.66	0
	!Bogus	.33	1



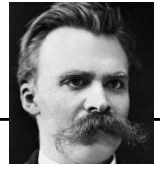
Guess

		Bogus	!Bogus
True	Bogus	0.66	0
	!Bogus	0.33	1




Guess

		Bogus	!Bogus
True	Bogus	.5	0.33
	!Bogus	.5	0.66




		Bogus	!Bogus
True	Bogus	0	0.66
	!Bogus	1	0.33

# EM Algorithm - Example




“Correct” Labels

$o_1$	1	1	1	0	0	$\begin{pmatrix} ? \\ ? \\ ? \\ ? \\ ? \end{pmatrix}$
$o_2$	0	0	0	1	1	
$o_3$	0	1	1	0	0	
$o_4$	0	0	0	0	1	
$o_5$	0	1	1	1	0	



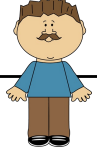
Guess

	Bogus	!Bogus
True Bogus	1	0.25
True !Bogus	0	.75




Guess

	Bogus	!Bogus
True Bogus	.66	0
True !Bogus	.33	1



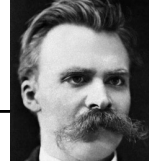
Guess

	Bogus	!Bogus
True Bogus	0.66	0
True !Bogus	0.33	1



Guess

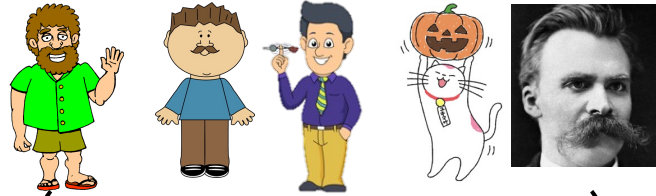
	Bogus	!Bogus
True Bogus	.5	0.33
True !Bogus	.5	0.66




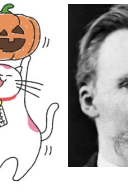



	Bogus	!Bogus
Bogus	0	0.66
!Bogus	1	0.33



# EM Algorithm - Example




				
$o_1$	1	1	1	0
$o_2$	0	0	0	1
$o_3$	0	1	1	0
$o_4$	0	0	0	1
$o_5$	0	1	1	0


Bogus

Not Bogus


$$\begin{pmatrix} 1 + .66 + .66 + .33 + .66 & 0 + .33 + .33 + .66 + .33 \\ 0.25 + 0 + 0 + .5 + 0 & .75 + 1 + 1 + 0.5 + 1 \\ 0.25 + .66 + .66 + 0.33 + .66 & .75 + .33 + .33 + 0.66 + .33 \\ 0.25 + 0 + 0 + .33 + 0 & .75 + 1 + 1 + .66 + 1 \\ .25 + .66 + .66 + .5 + .66 & .75 + .33 + .33 + .5 + .33 \end{pmatrix}$$




		Guess	
		Bogus	!Bogus
True	Bogus	1	0.25
	!Bogus	0	.75



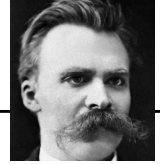
		Guess	
		Bogus	!Bogus
True	Bogus	.66	0
	!Bogus	.33	1



		Guess	
		Bogus	!Bogus
True	Bogus	0.66	0
	!Bogus	0.33	1



		Guess	
		Bogus	!Bogus
True	Bogus	.5	0.33
	!Bogus	.5	0.66



		Guess	
		Bogus	!Bogus
True	Bogus	0	0.66
	!Bogus	1	0.33

# EM Algorithm - Example



Bogus

Not Bogus

	$o_1$	1	1	1	0	0
	$o_2$	0	0	0	1	1
	$o_3$	0	1	1	0	0
	$o_4$	0	0	0	0	1
	$o_5$	0	1	1	1	0

$$\begin{pmatrix}
 1 + .66 + .66 + .33 + .66 & 0 + .33 + .33 + .66 + .33 \\
 0.25 + 0 + 0 + .5 + 0 & .75 + 1 + 1 + 0.5 + 1 \\
 0.25 + .66 + .66 + 0.33 + .66 & .75 + .33 + .33 + 0.66 + .33 \\
 0.25 + 0 + 0 + .33 + 0 & .75 + 1 + 1 + .66 + 1 \\
 .25 + .66 + .66 + .5 + .66 & .75 + .33 + .33 + .5 + .33
 \end{pmatrix}$$

		Guess	
		Bogus	!Bogus
True	Bogus	1	0.25
	!Bogus	0	.75

		Guess	
		Bogus	!Bogus
True	Bogus	.66	0
	!Bogus	.33	1

		Guess	
		Bogus	!Bogus
True	Bogus	0.66	0
	!Bogus	0.33	1

		Guess	
		Bogus	!Bogus
True	Bogus	.5	0.33
	!Bogus	.5	0.66

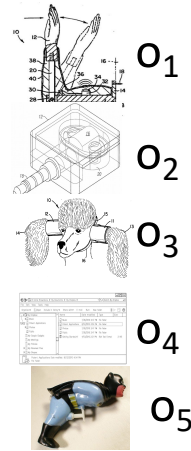
		Guess	
		Bogus	!Bogus
True	Bogus	0	0.66
	!Bogus	1	0.33

# EM Algorithm - Example



Bogus

Not Bogus



$o_1$	1	1	1	0	0
$o_2$	0	0	0	1	1
$o_3$	0	1	1	0	0
$o_4$	0	0	0	0	1
$o_5$	0	1	1	1	0

$$\begin{pmatrix} 1 + .66 + .66 + .33 + .66 \\ 0.25 + 0 + 0 + .5 + 0 \\ 0.25 + .66 + .66 + 0.33 + .66 \\ 0.25 + 0 + 0 + .33 + 0 \\ .25 + .66 + .66 + .5 + .66 \end{pmatrix}$$

$$\begin{pmatrix} 0 + .33 + .33 + .66 + .33 \\ .75 + 1 + 1 + 0.5 + 1 \\ .75 + .33 + .33 + 0.66 + .33 \\ .75 + 1 + 1 + .66 + 1 \\ .75 + .33 + .33 + .5 + .33 \end{pmatrix}$$

		Guess	
		Bogus	!Bogus
True	Bogus	1	0.25
	!Bogus	0	.75


		Guess	
		Bogus	!Bogus
True	Bogus	.66	0
	!Bogus	.33	1


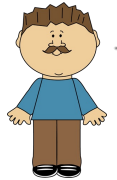



		Guess	
		Bogus	!Bogus
True	Bogus	0.66	0
	!Bogus	0.33	1

		Guess	
		Bogus	!Bogus
True	Bogus	.5	0.33
	!Bogus	.5	0.66

		Guess	
		Bogus	!Bogus
Bogus		0	0.66
!Bogus		1	0.33

# EM Algorithm - Example




				
$o_1$	1	1	1	0
$o_2$	0	0	0	1
$o_3$	0	1	1	0
$o_4$	0	0	0	1
$o_5$	0	1	1	1


Bogus

Not Bogus

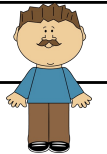
"Correct" Labels

$$\begin{pmatrix} 0.66 & .33 \\ 0.15 & .85 \\ 0.52 & 0.48 \\ .12 & 0.88 \\ 0.55 & 0.45 \end{pmatrix} \quad \begin{pmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}$$



	Guess	
	Bogus	!Bogus
True Bogus	1	0.25
True !Bogus	0	.75



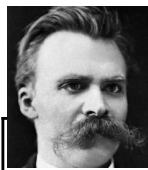
	Guess	
	Bogus	!Bogus
True Bogus	.66	0
True !Bogus	.33	1



	Guess	
	Bogus	!Bogus
True Bogus	0.66	0
True !Bogus	0.33	1



	Guess	
	Bogus	!Bogus
True Bogus	.5	0.33
True !Bogus	.5	0.66



	Bogus	!Bogus
Bogus	0	0.66
!Bogus	1	0.33


# Dawid and Skene EM Algorithm [1]

- Input:** Labels  $l[k][n]$  from worker ( $k$ ) to object  $o_n$ ,
- Output:** Confusion matrix  $\pi_{ij}^{(k)}$  for each worker ( $k$ ), Correct labels  $T(o_n)$  for each object  $o_n$ , Class priors  $Pr\{C\}$  for each class  $C$
- 1 Initialize error rates  $\pi_{ij}^{(k)}$  for each worker ( $k$ ) (e.g., assume each worker is perfect);
  - 2 Initialize correct label for each object  $T(o_n)$  (e.g., using majority vote);
  - 3 **while not converged do**
  - 4     Estimate the correct label  $T(o_n)$  for each object, using the labels  $l[\cdot][n]$  assigned to  $o_n$  by workers, weighting the votes using the error rates  $\pi_{ij}^{(k)}$ ;
  - 5     Estimate the error rates  $\pi_{ij}^{(k)}$ , for each worker ( $k$ ), using the correct labels  $T(o_n)$  and the assigned labels  $l[k][n]$ ;
  - 6     Estimate the class priors  $Pr\{C\}$ , for each class  $C$ ;
  - 7 **end**
  - 8 **return** *Estimated error rates  $\pi_{ij}^{(k)}$ , Estimated correct labels  $T(o_n)$ , Estimated class priors  $Pr\{C\}$*


[1] Panos Ipeirotis, Foster Provost, Jing Wang: **Quality management on Amazon Mechanical Turk**. Proceedings of the ACM SIGKDD Workshop on Human Computation, 2010

[2] Dawid, A. P., and Skene, A. M. **Maximum likelihood estimation of observer error-rates using the EM algorithm**. Applied Statistics 28, 1 (Sept. 1979), 20–28.

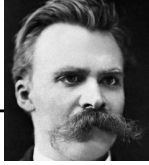
# Confusion Matrices in the 2<sup>nd</sup> iteration



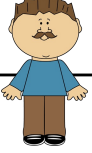
	Bogus	!Bogus
True Bogus	1	0.5
True !Bogus	0	.5




	Bogus	!Bogus
True Bogus	1	0
True !Bogus	0	1



	Bogus	!Bogus
True Bogus	0	1
True !Bogus	1	0



	Bogus	!Bogus
True Bogus	1	0
True !Bogus	0	1



	Bogus	!Bogus
True Bogus	.5	0.66
True !Bogus	.5	0.33

**Which worker is the worst?**

# EM-Algorithm:

## Many other applications

Initialize  $\theta \in \Theta$

For  $t = 0, 1, 2, \dots$

E-Step: Calculate the expected value of the log likelihood function, with respect to the conditional distribution of  $Z$  given  $X$  under the current estimate of the parameters  $\theta_t$ :

$$Q(Q|\theta_t) = E_{Z|X, \theta_t}[\log \mathcal{L}(\theta, X, Z)]$$

M-Step: Find the parameter that maximizes this quantity

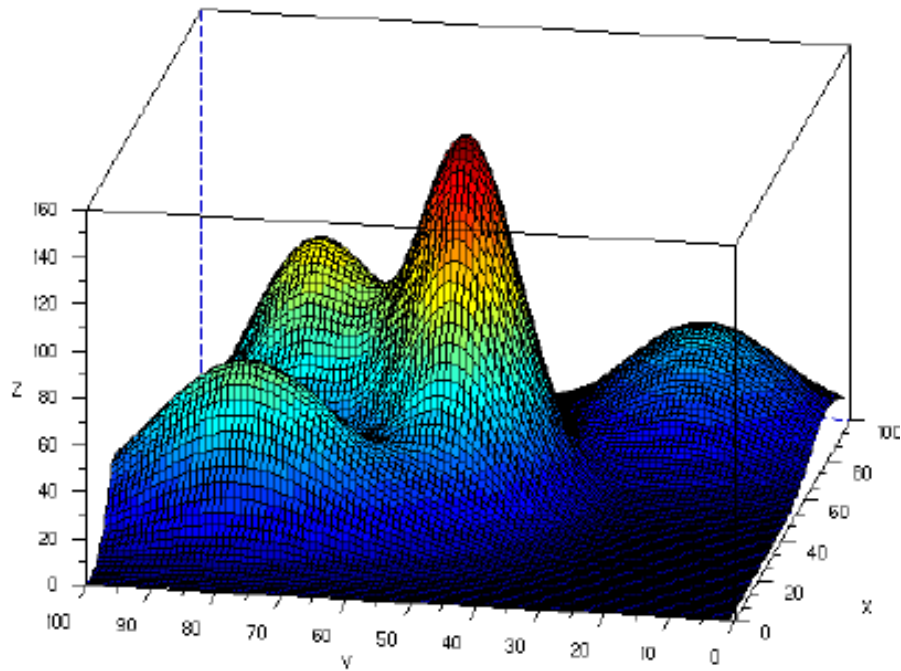
$$\theta_{t+1} = \underset{\theta}{\operatorname{argmax}} Q(Q|\theta_t)$$



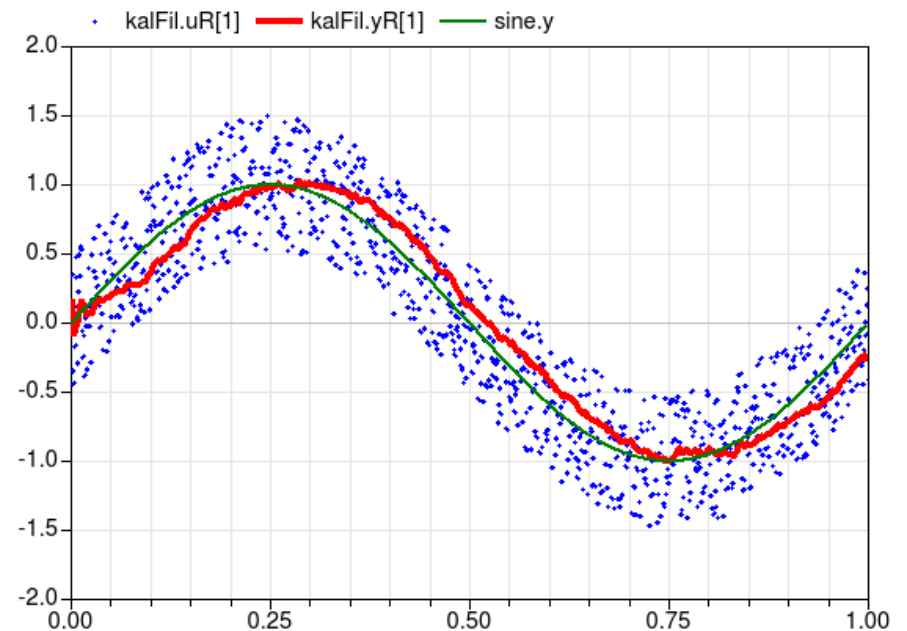
# EM-Algorithm: Many other applications

Initialize  $\theta \in \Theta$

## Gaussian Mixture Models (GMM)



## Kalman filter



maximizes this quantity

$$\theta_{t+1} = \operatorname{argmax}_{\theta} Q(Q|\theta_t)$$



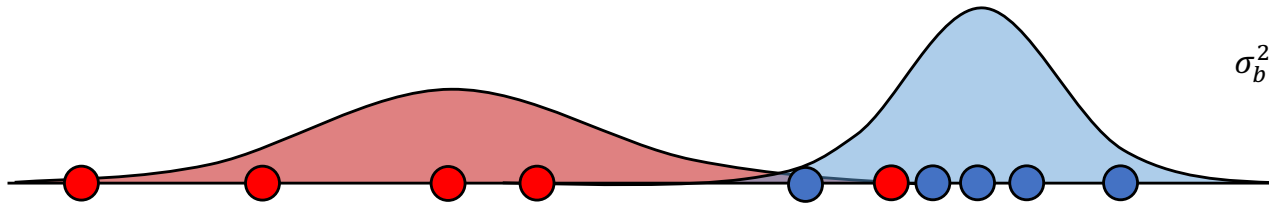
# EM-Algorithm: Gaussian Mixture Models

Unknown distributions parameters, known data point labels

- K=2 Gaussians with unknown  $\mu, \sigma$
- Assume you know if data point comes from the red or blue distribution.
- Estimating the distribution parameters is trivial

$$\mu_b = \frac{x_1 + x_2 + \dots + x_n}{n_b}$$

$$\sigma_b^2 = \frac{(x_1 - \mu_b)^2 + (x_2 - \mu_b)^2 + \dots + (x_n - \mu_b)^2}{n_b}$$



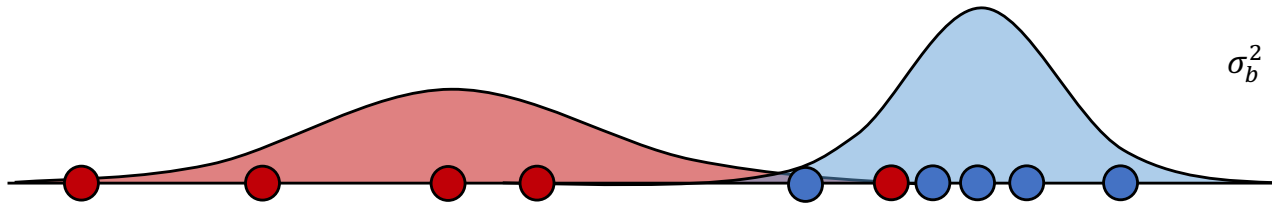
# EM-Algorithm: Gaussian Mixture Models

## Unknown distributions parameters, known data point labels

- K=2 Gaussians with unknown  $\mu, \sigma$
- Assume you know if data point comes from the red or blue distribution.
- Estimating the distribution parameters is trivial

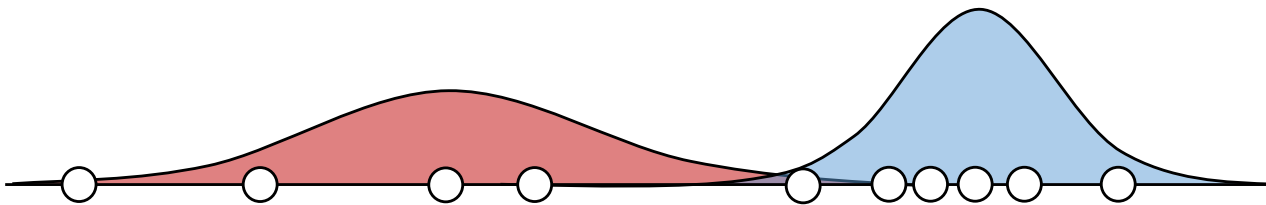
$$\mu_b = \frac{x_1 + x_2 + \dots + x_n}{n_b}$$

$$\sigma_b^2 = \frac{(x_1 - \mu_b)^2 + (x_2 - \mu_b)^2 + \dots + (x_n - \mu_b)^2}{n_b}$$



## Known distributions parameters, unknown data point labels

- We can guess whether the point is more likely from the blue or red distribution



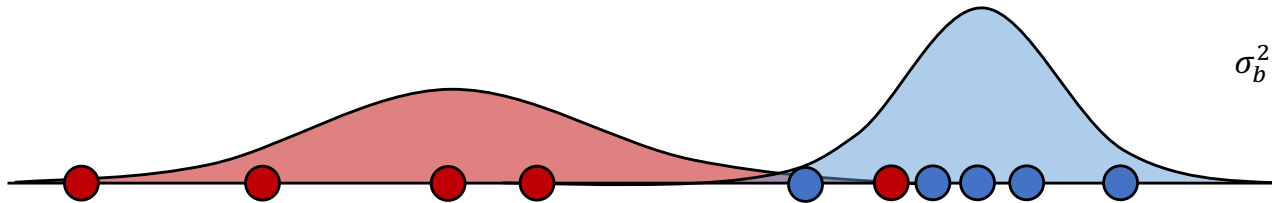
# EM-Algorithm: Gaussian Mixture Models

## Unknown distributions parameters, known data point labels

- K=2 Gaussians with unknown  $\mu, \sigma$
- Assume you know if data point comes from the red or blue distribution.
- Estimating the distribution parameters is trivial

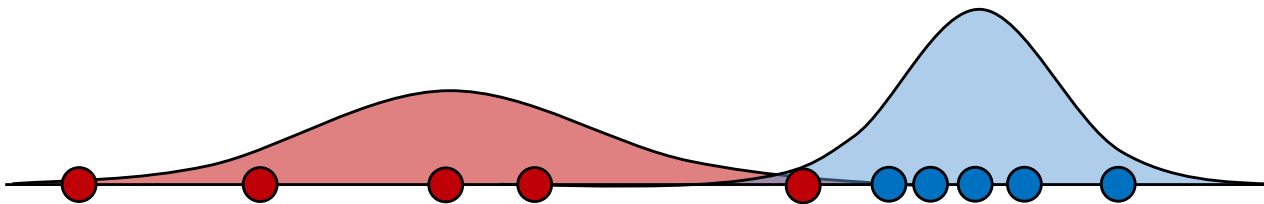
$$\mu_b = \frac{x_1 + x_2 + \dots + x_n}{n_b}$$

$$\sigma_b^2 = \frac{(x_1 - \mu_b)^2 + (x_2 - \mu_b)^2 + \dots + (x_n - \mu_b)^2}{n_b}$$



## Known distributions parameters, unknown data point labels

- We can guess whether the point is more likely from the blue or red distribution



$$P(b|x_1) = \frac{P(x_1|b)P(b)}{P(x_1|b)P(b) + P(x_1|r)P(r)}$$

$$P(x_1|b) = \frac{1}{\sqrt{2\pi\sigma_b^2}} \exp\left(-\frac{(x_1 - \mu_b)^2}{2\sigma_b^2}\right)$$

# EM-Algorithm: Gaussian Mixture Models

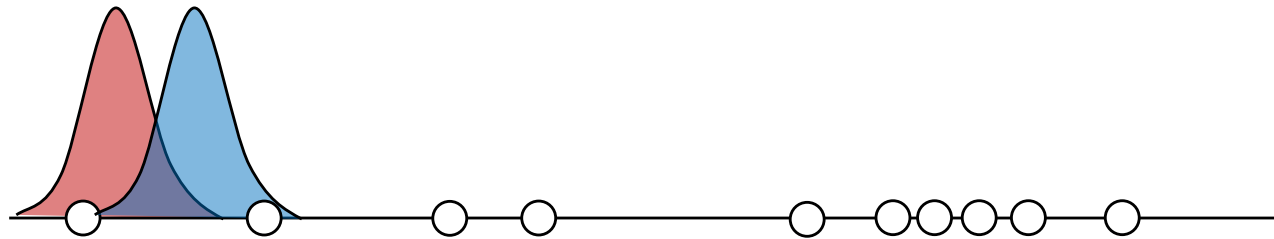
## Chicken and egg problem

- Need  $(\mu_r, \sigma_r^2)$  and  $(\mu_b, \sigma_b^2)$  to guess source of points
- Need to know the source to estimate  $(\mu_r, \sigma_r^2)$  and  $(\mu_b, \sigma_b^2)$

## EM algorithm

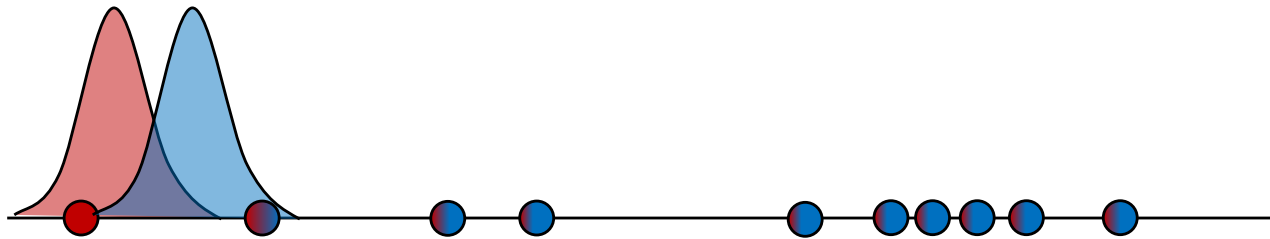
- Start with two randomly placed Gaussians  $(\mu_r, \sigma_r^2)$  and  $(\mu_b, \sigma_b^2)$
- E-step: For each point:  $P(b|x_1)$  = does it look it came from blue (red)
- M-Step: adjust  $(\mu_r, \sigma_r^2)$  and  $(\mu_b, \sigma_b^2)$  to fit points assigned to them
- Iterate until convergence

# EM-Algorithm: Gaussian Mixture Models



- Start with two randomly placed Gaussians  $(\mu_r, \sigma_r^2)$  and  $(\mu_b, \sigma_b^2)$

# EM-Algorithm: Gaussian Mixture Models



- Start with two randomly placed Gaussians  $(\mu_r, \sigma_r^2)$  and  $(\mu_b, \sigma_b^2)$
- E-step: For each point:  $P(b|x_1)$  = does it look it came from blue (red)

*E-Step:*

$$P(x_1|b) = \frac{1}{\sqrt{2\pi\sigma_b^2}} \exp\left(-\frac{(x-\mu_b)^2}{2\sigma_b^2}\right)$$

Note: we could estimate priors  $P(b)$  and  $P(r)$ , but often left at equal chance

$$b_i = P(b|x_i) = \frac{P(x_i|b)P(b)}{P(x_i|b)P(b) + P(x_i|r)P(r)}$$

$$r_i = 1 - b_i$$

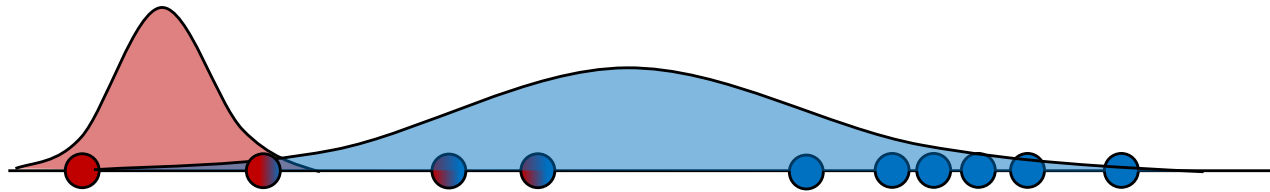
- M-Step: adjust  $(\mu_r, \sigma_r^2)$  and  $(\mu_b, \sigma_b^2)$  to fit points assigned to them

*M-Step:*

$$\mu_b = \frac{b_1x_1 + b_2x_2 + \dots + b_nx_n}{b_1 + b_2 + \dots + b_n}$$

$$\sigma_b^2 = \frac{b_1(x_1-\mu_b)^2 + b_2(x_2-\mu_b)^2 + \dots + b_n(x_n-\mu_b)^2}{n_b}$$

# EM-Algorithm: Gaussian Mixture Models



- Start with two randomly placed Gaussians  $(\mu_r, \sigma_r^2)$  and  $(\mu_b, \sigma_b^2)$
- E-step: For each point:  $P(b|x_1)$  = does it look it came from blue (red)

*E-Step:*

$$P(x_1|b) = \frac{1}{\sqrt{2\pi\sigma_b^2}} \exp\left(-\frac{(x-\mu_b)^2}{2\sigma_b^2}\right)$$

Note: we could estimate priors  $P(b)$  and  $P(r)$ , but often left at equal chance

$$b_i = P(b|x_i) = \frac{P(x_i|b)P(b)}{P(x_i|b)P(b) + P(x_i|r)P(r)}$$

$$r_i = 1 - b_i$$

- M-Step: adjust  $(\mu_r, \sigma_r^2)$  and  $(\mu_b, \sigma_b^2)$  to fit points assigned to them

*M-Step:*

$$\mu_b = \frac{b_1x_1 + b_2x_2 + \dots + b_nx_n}{b_1 + b_2 + \dots + b_n}$$

$$\sigma_b^2 = \frac{b_1(x_1-\mu_b)^2 + b_2(x_2-\mu_b)^2 + \dots + b_n(x_n-\mu_b)^2}{n_b}$$

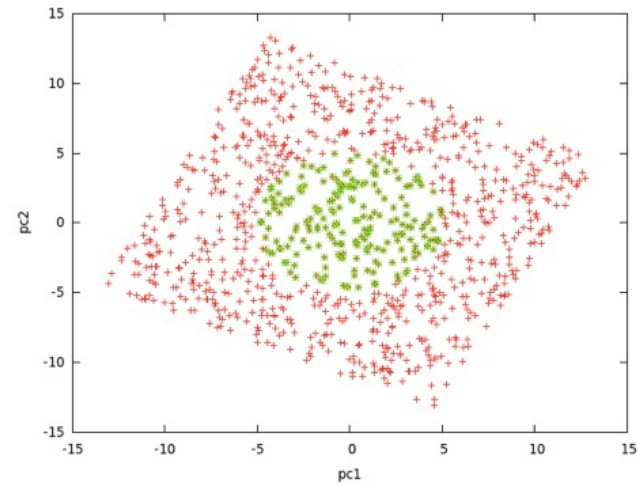
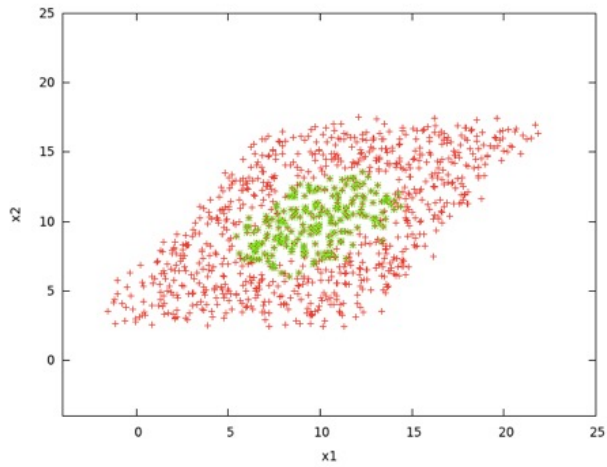
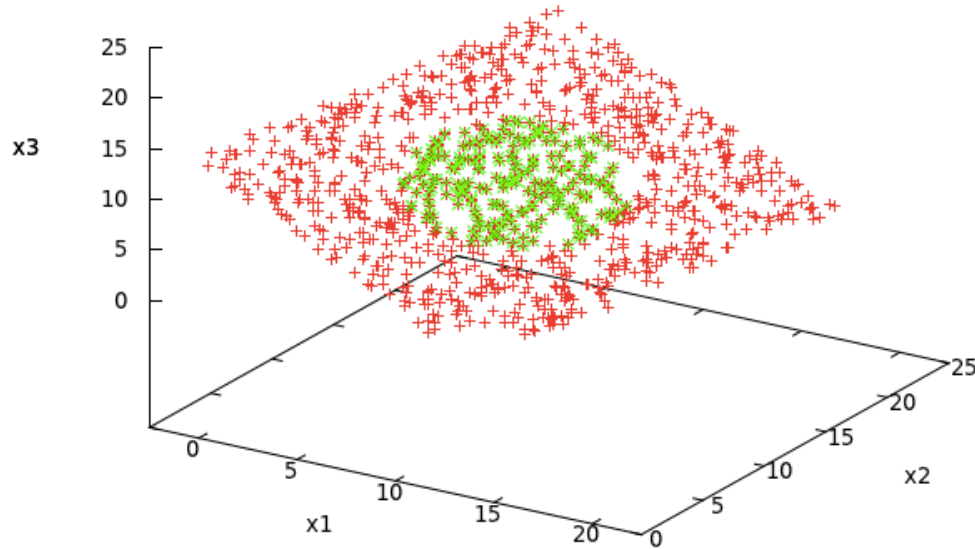
In what way is the algorithm similar to k-means and in what ways is it different?

# Machine Learning Problems

	Supervised Learning	Unsupervised Learning
Discrete	classification or categorization	clustering
Continuous	regression	dimensionality reduction

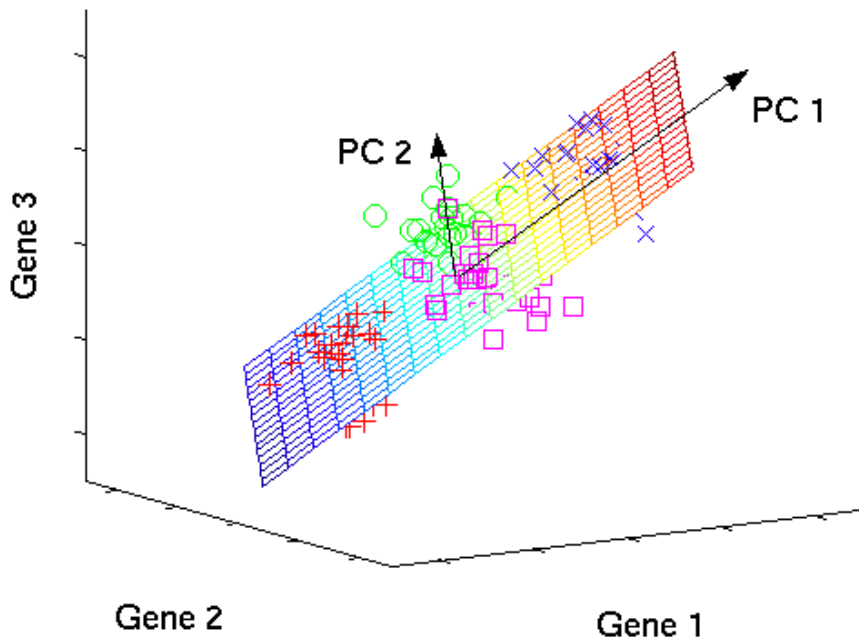


# PRINCIPAL COMPONENT ANALYSIS (PCA)



# PCA

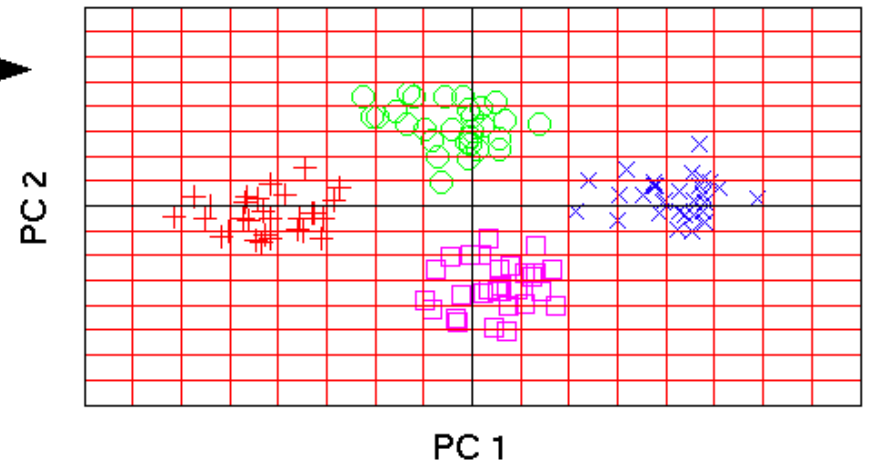
original data space



PCA



component space



# PCA Intuition

