# Happy Wednesday!

- HW 1 is due on Feb 13
- Math part of course is over (mostly) ☺
- HW2 is releasing on Feb 13
  - Covers kMeans, EM, GMM.
- Mahdi will cover next week's lecture. GMM (Very Important topic).

*Machine Learning CS 4641*

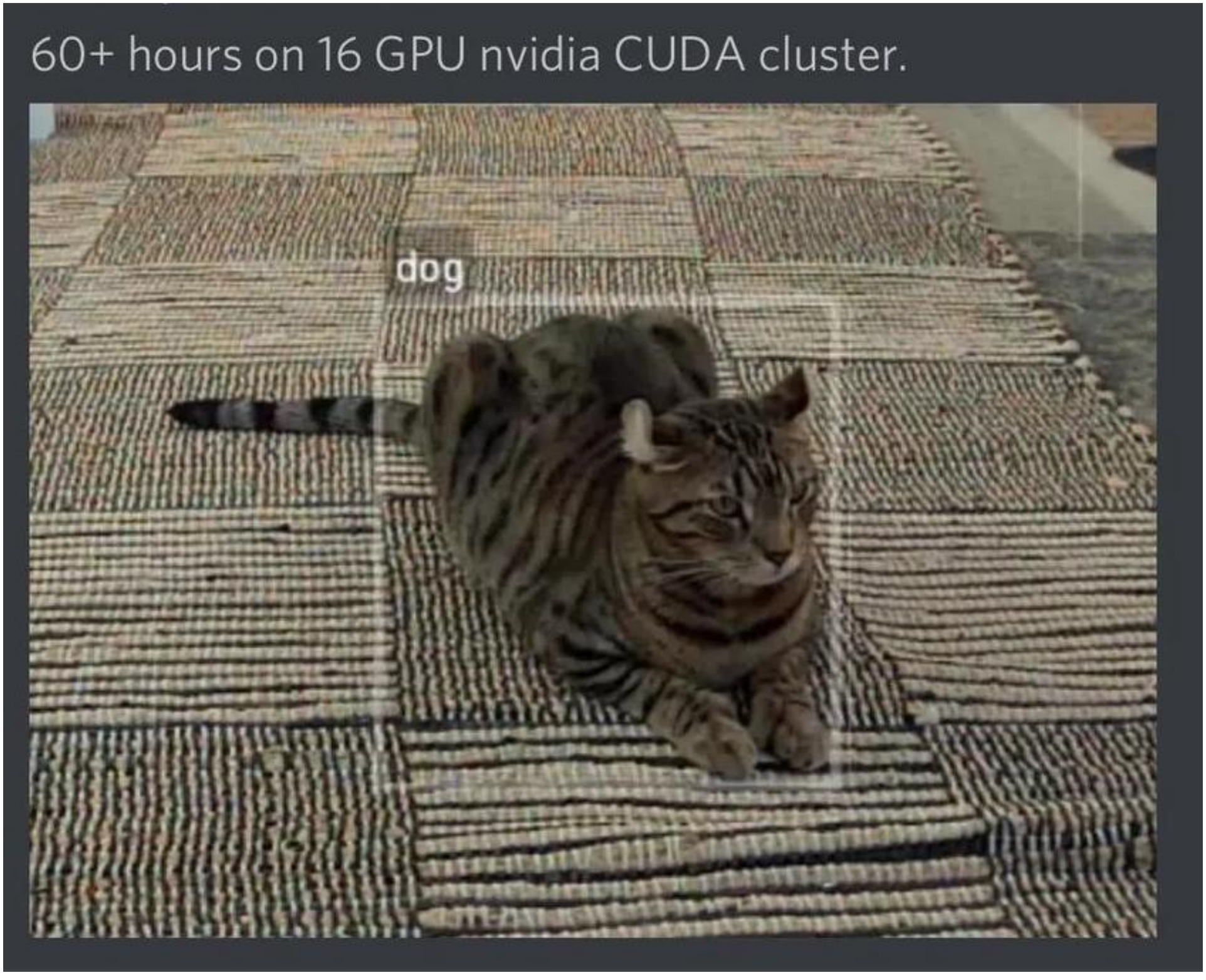
# Clustering Analysis and KMeans


Dr. Nimisha Roy

*Lecturer, SCI, College of Computing, Georgia Tech*

*Director, Online Undergraduate Initiatives*


*Slides adapted from Max Roozbahani, Rod Borela*

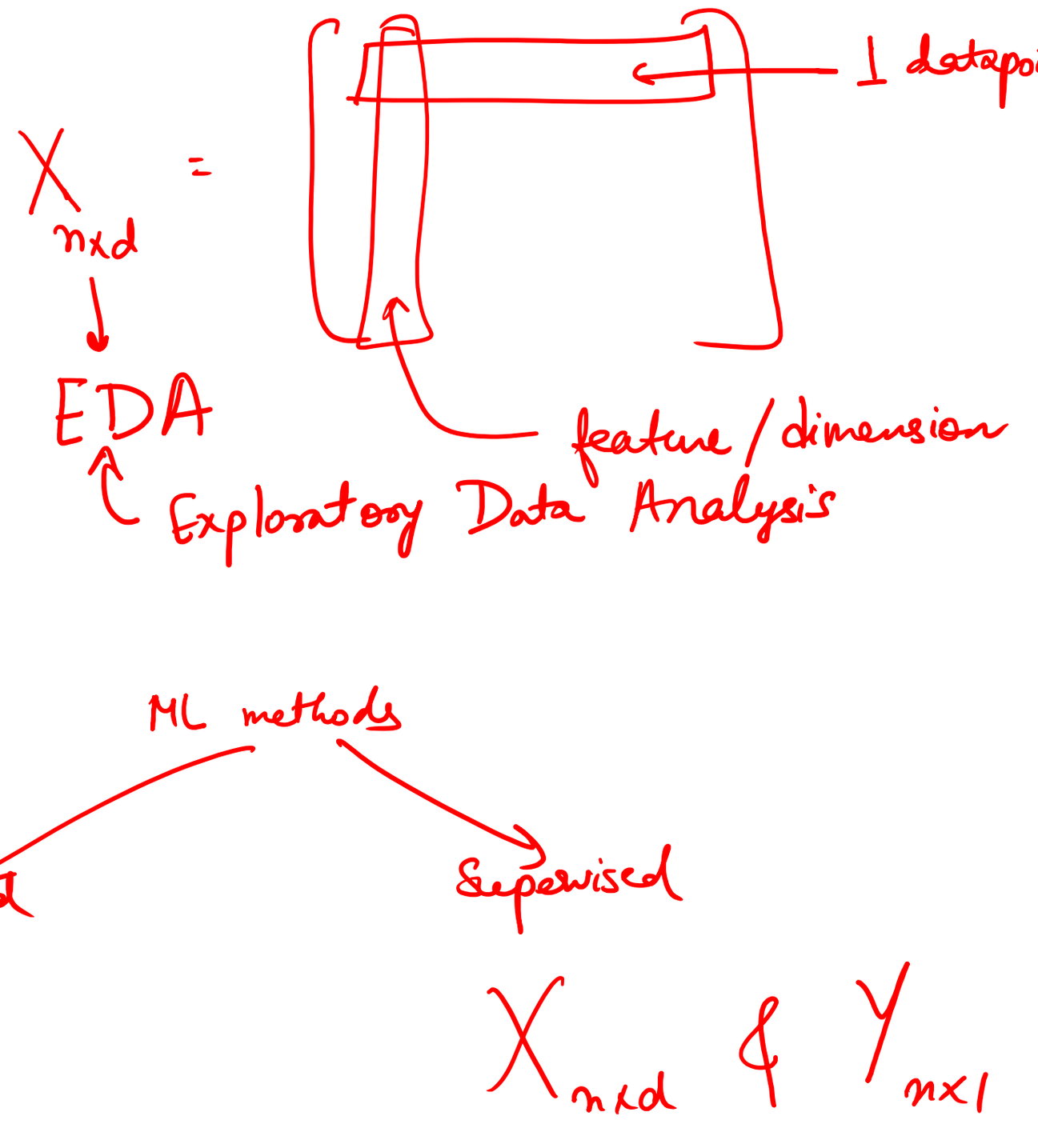

60+ hours on 16 GPU nvidia CUDA cluster.
dog

# Outline

- Clustering
- Distance functions
- K-Means algorithm
- Analysis of K-Means

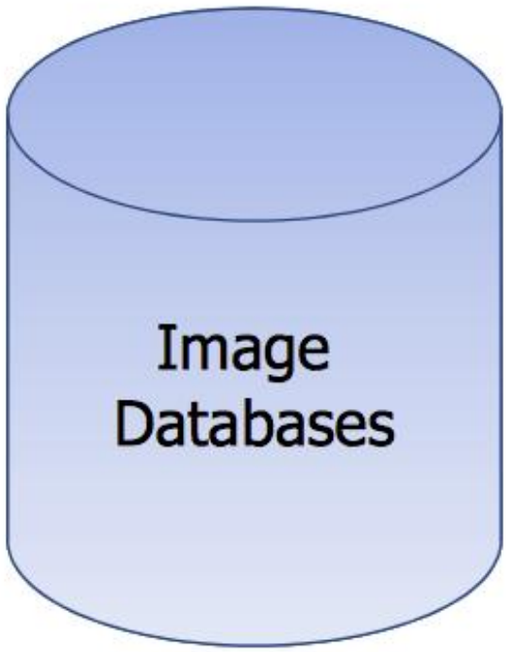
# Outline

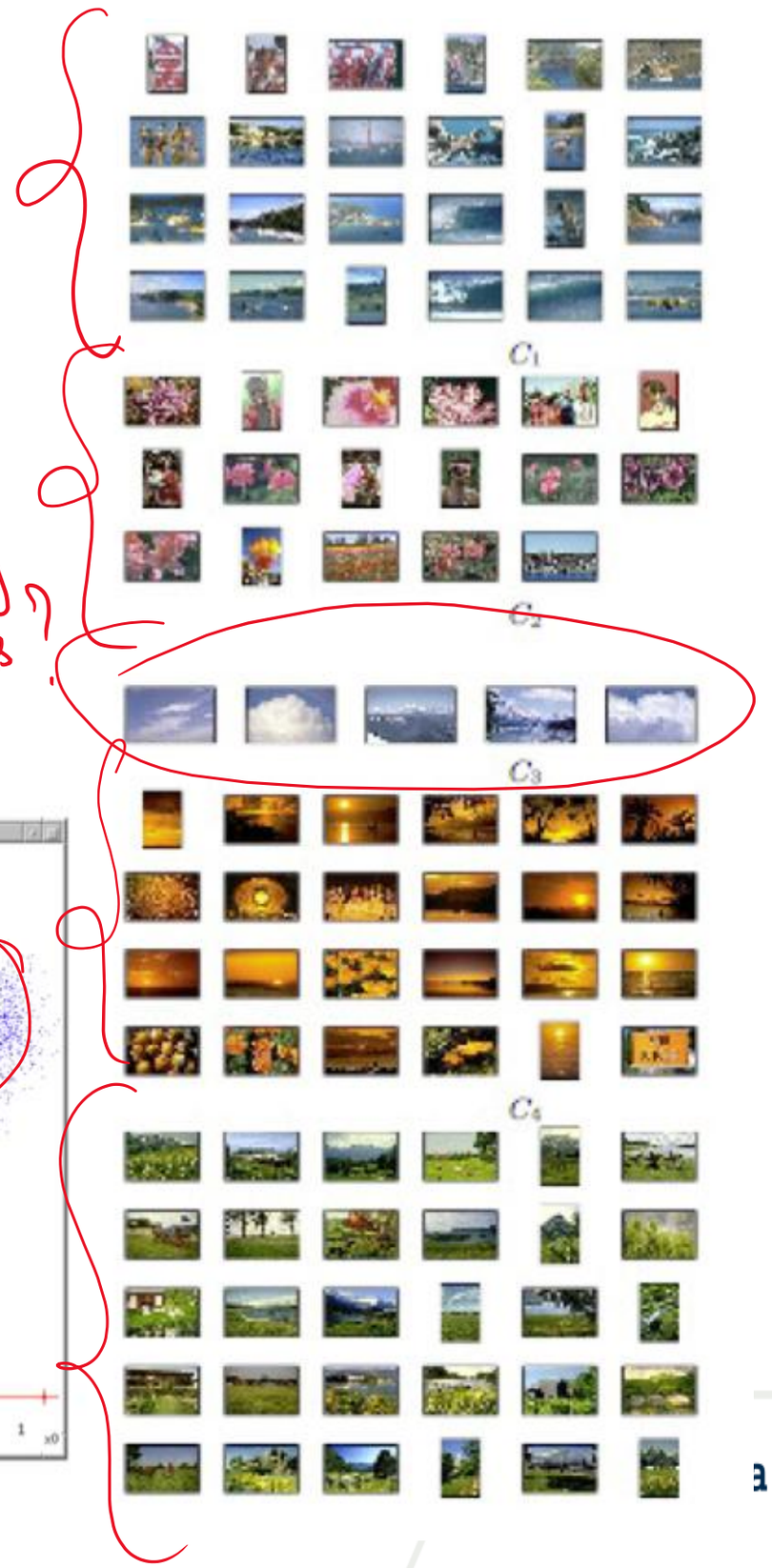
- Clustering
- Distance functions
- K-Means algorithm
- Analysis of K-Means

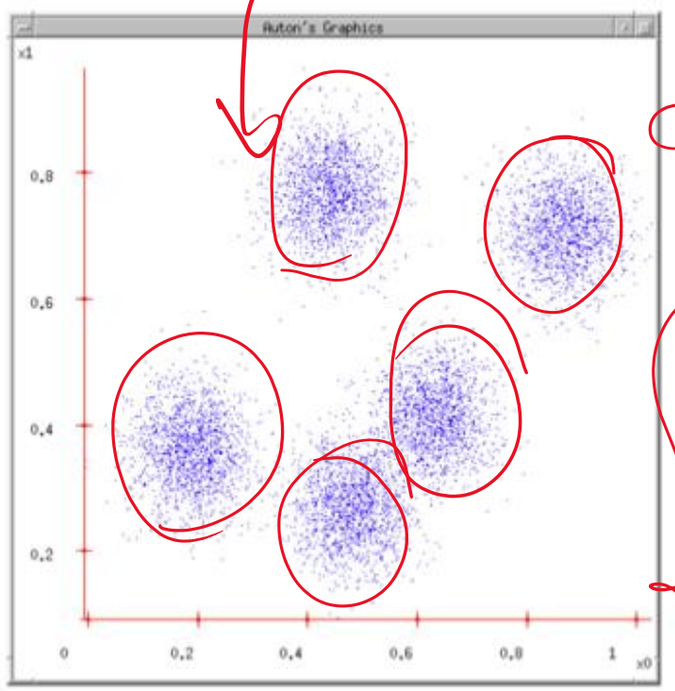


# Clustering Images

What is the similarity metric?



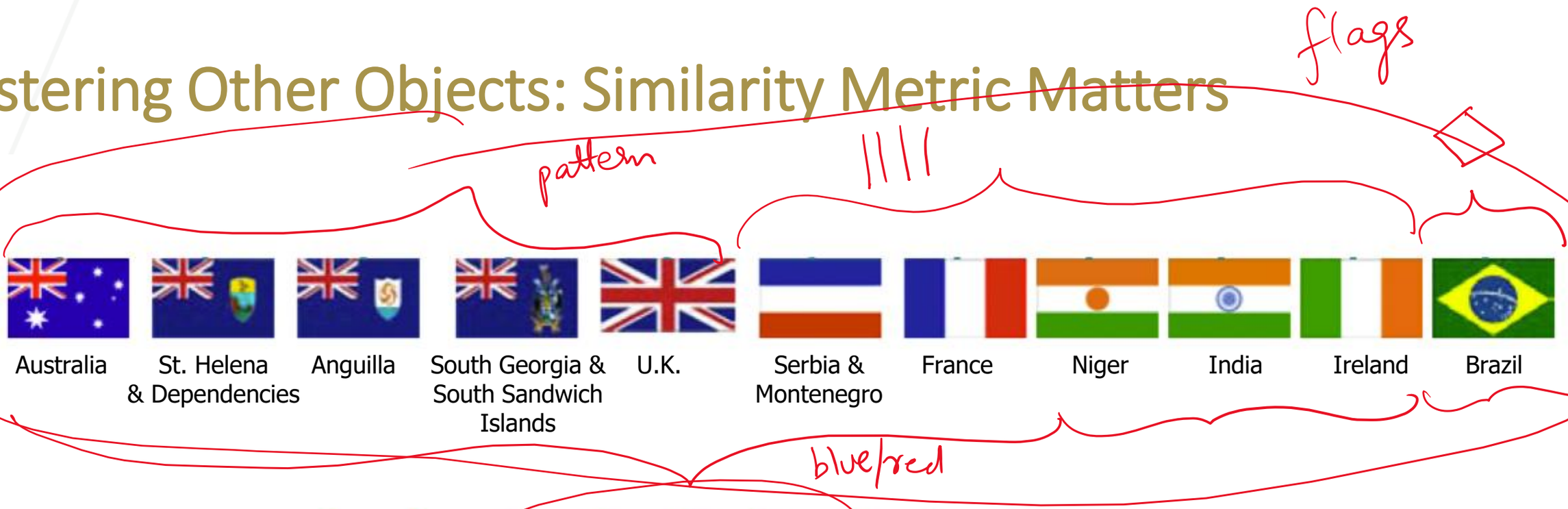




**Goal of clustering:**

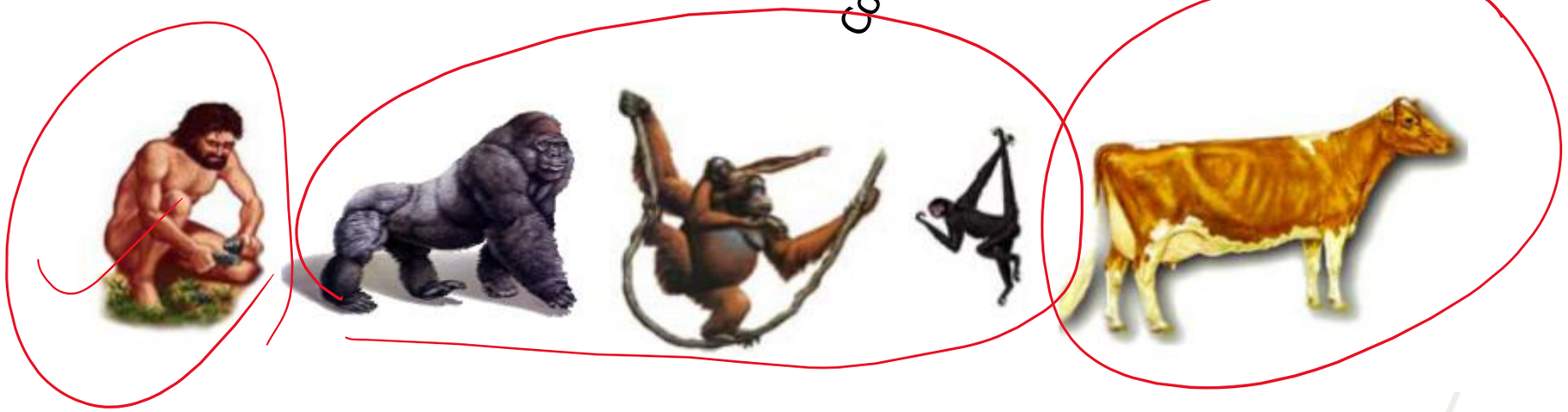
Divide object into groups, and objects within a group are more similar than those outside the group

# Clustering Other Objects: Similarity Metric Matters

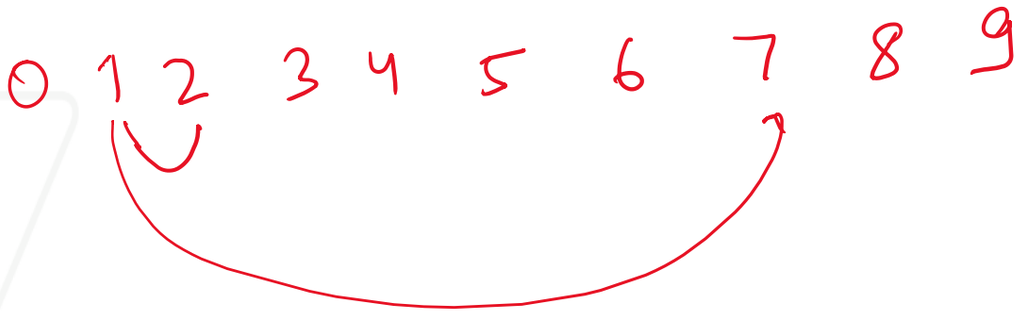


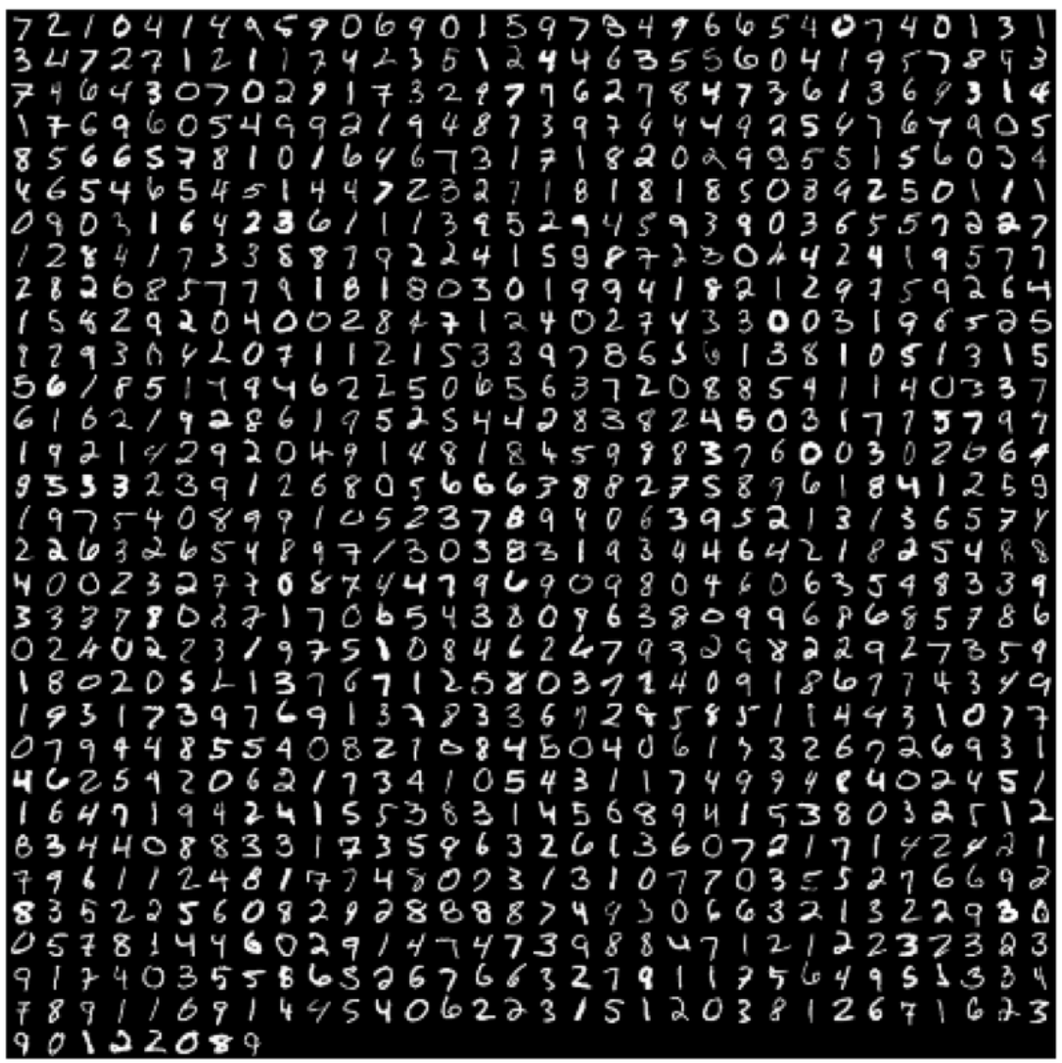


Linguistic Similarity



# Clustering hand digits





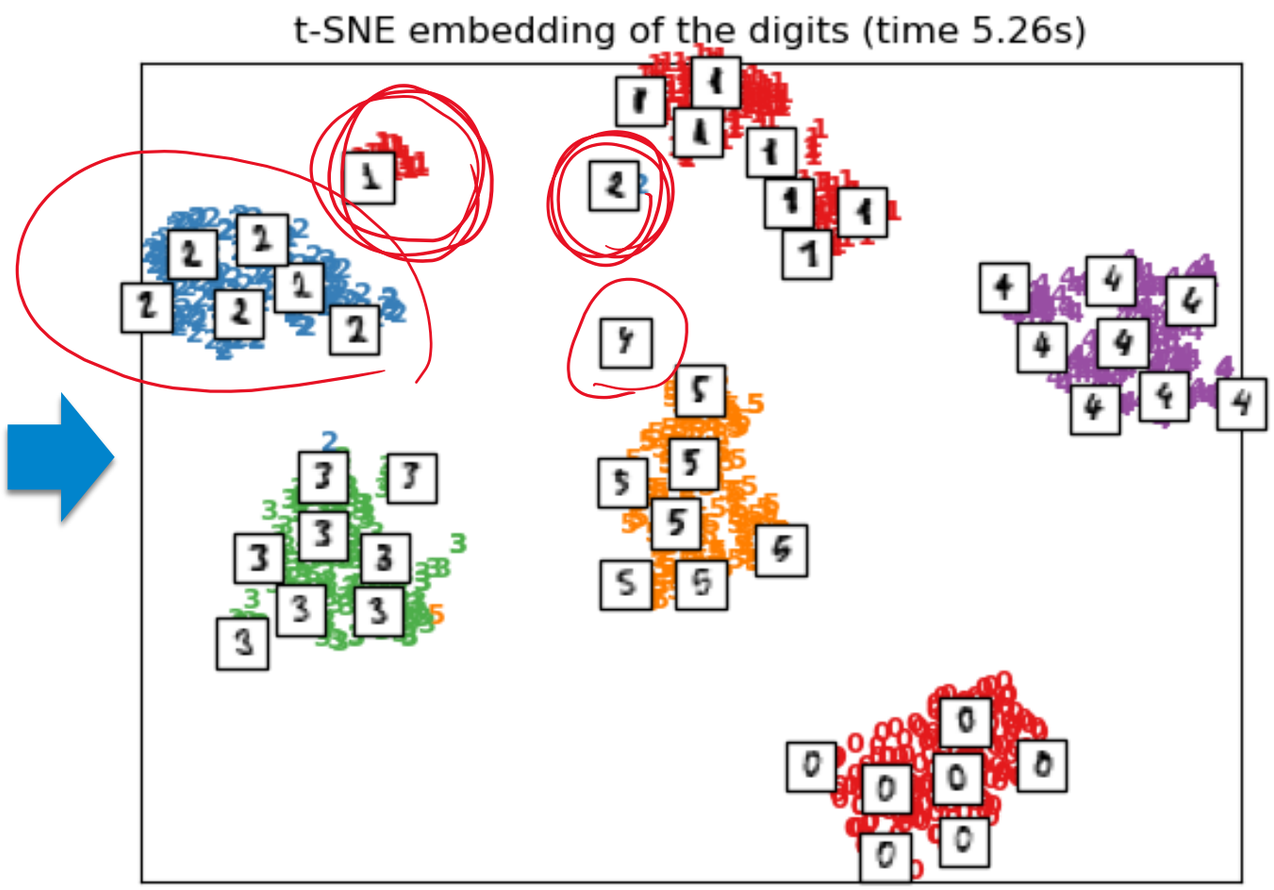
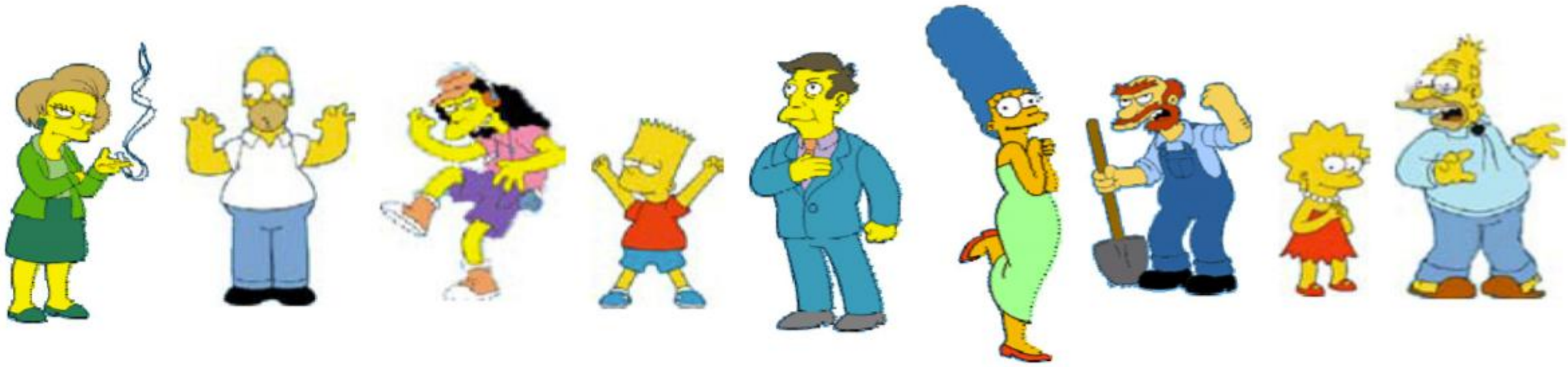



Image credit: Scikit learn



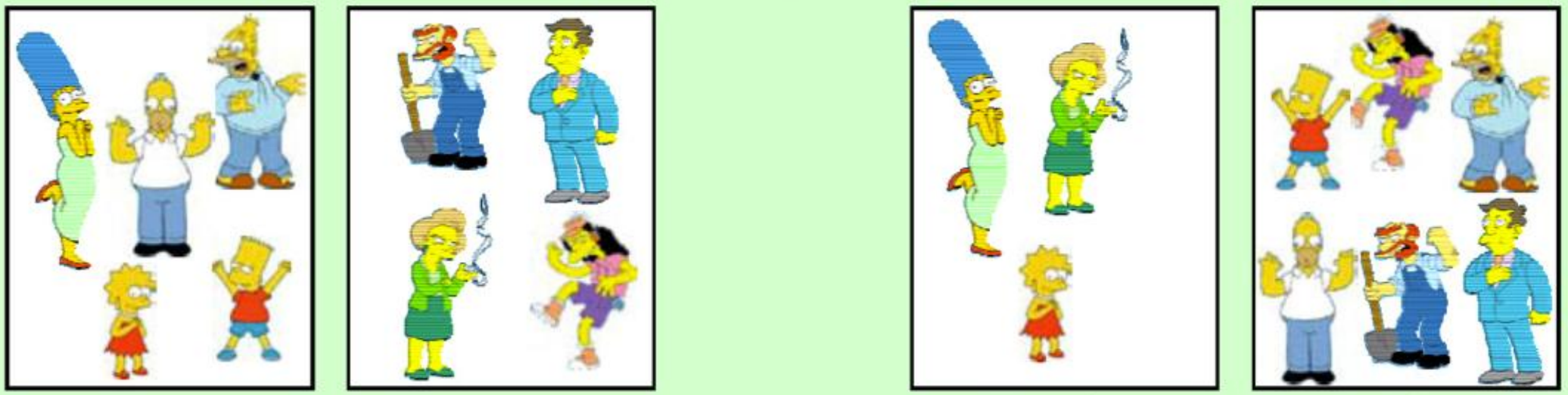
Georgia Tech

# Clustering is Subjective



What is consider similar/dissimilar?

## Clustering is subjective



# So, What is Clustering in General?

- You pick your similarity/dissimilarity function
- The algorithm figures out the grouping of objects based on the chosen similarity/dissimilarity function
  - Points within a cluster is similar
  - Points across clusters are not so similar
- Issues for clustering
  - How to represent objects? (Vector space? Normalization?)
  - What is a similarity/dissimilarity function for your data?
  - What are the algorithm steps?

# How is clustering useful?

EDA

- Helping us make decisions about the quality of our data
- Identifying communities or categories of items to better understand our data
  - Are there unexpected clusters?
  - Are there more clusters than classes in a labeled dataset?
  - Are there data points that don't fall in any groups?
- Streamlining tasks such as database search and data labeling
- Identifying whether there are natural separations in our dataset before proceeding to a classification task
- *Many other use cases…*

# Outline

1 of the similarity metrics

- Clustering
- **Distance functions**
- K-Means algorithm
- Analysis of K-Means

# Properties of distance functions

- Desired properties of distance functions:

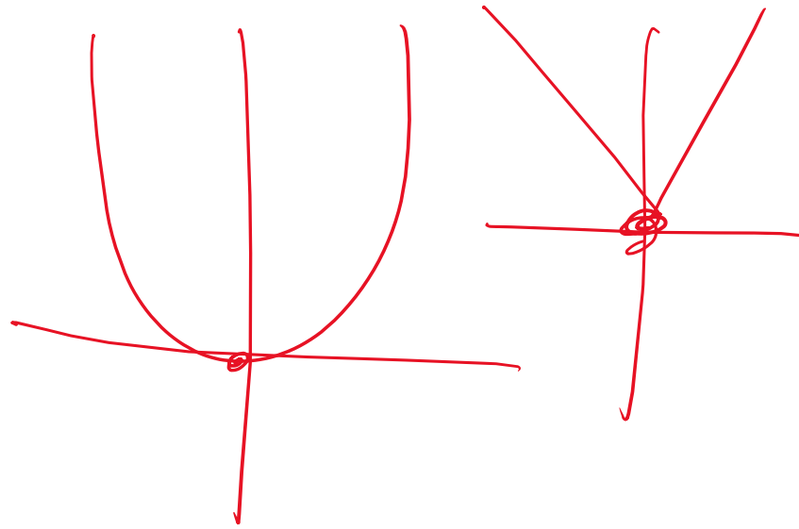
- Symmetry: $d(\mathbf{x}, \mathbf{y}) = d(\mathbf{y}, \mathbf{x})$
  - Otherwise you could claim "Alex looks like Taylor, but Taylor looks nothing like Alex"

- Positive separability: $d(\mathbf{x}, \mathbf{y}) = 0$, if and only if $\mathbf{x} = \mathbf{y}$
  - Otherwise there are objects that are different, but you cannot tell them apart

- Triangular inequality: $d(\mathbf{x}, \mathbf{y}) \leq d(\mathbf{x}, \mathbf{z}) + d(\mathbf{z}, \mathbf{y})$
  - Otherwise you could claim "Alex looks like Taylor, and Alex looks like Chris, but Taylor does not look like Chris"

# Distance functions for vectors

p=1 → manhattan
p=2 → euclidean
p=∞ → inf. dist.



- Suppose two data points, both in $\mathbb{R}^D$

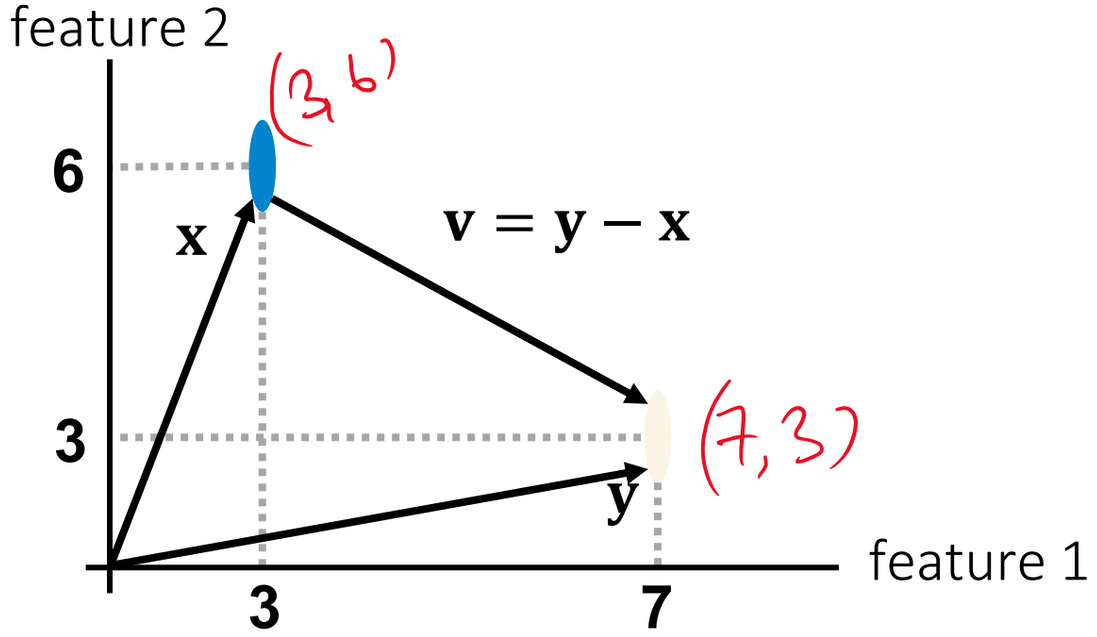
$$\mathbf{x} = (x_1, x_2, \ldots, x_D)$$
$$\mathbf{y} = (y_1, y_2, \ldots, y_D)$$

- Euclidean distance (L2-norm): $d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{D}(x_i - y_i)^2}$

- Minkowski distance (Lp-norm): $d(\mathbf{x}, \mathbf{y}) = \sqrt[p]{\sum_{i=1}^{D}(x_i - y_i)^p}$
  - Euclidean distance: $p = 2$
  - Manhattan distance: $p = 1, d(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{D} |x_i - y_i|$
  - "Inf"-distance: $p = \infty, d(\mathbf{x}, \mathbf{y}) = \max_i |x_i - y_i|$

# Example

- Euclidean distance: $d(\mathbf{x}, \mathbf{y}) = \sqrt{\sum_{i=1}^{D}(x_i - y_i)^2} = \sqrt{(7-3)^2 + (3-6)^2} = 5$
- Manhattan distance: $d(\mathbf{x}, \mathbf{y}) = |7-3| + |3-6| = 7$
- "Inf"-distance: $d(\mathbf{x}, \mathbf{y}) = \max(|7-3|, |3-6|) = 4$



# Hamming distance

- Manhattan distance is also called **Hamming distance when all features are binary**
  - Count the number of difference between two binary vectors
  - Example, $\mathbf{x}, \mathbf{y} \in \{0,1\}^{17}$

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| $x$ | 0 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 0 | 1 |
| $y$ | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |

$d(\mathbf{x}, \mathbf{y}) = 5$

- When is the Hamming distance used?
  - When features are categorical
  - e.g., DNA sequences, yes/no survey answers, on/off flags.

# Edit distance

≠ distance metric

$d(y,x) \neq d(x,y)$

- Transform one of the objects into the other, and measure how much effort it takes

$x$ I N T E * N T I O N

$y$ * E X E C U T I O N

d s s i s

5 + 1 + 1 + 2 + 1

$d(\mathbf{x}, \mathbf{y}) = 5 \times 1 + 3 \times 1 + 1 \times 2 = 10$

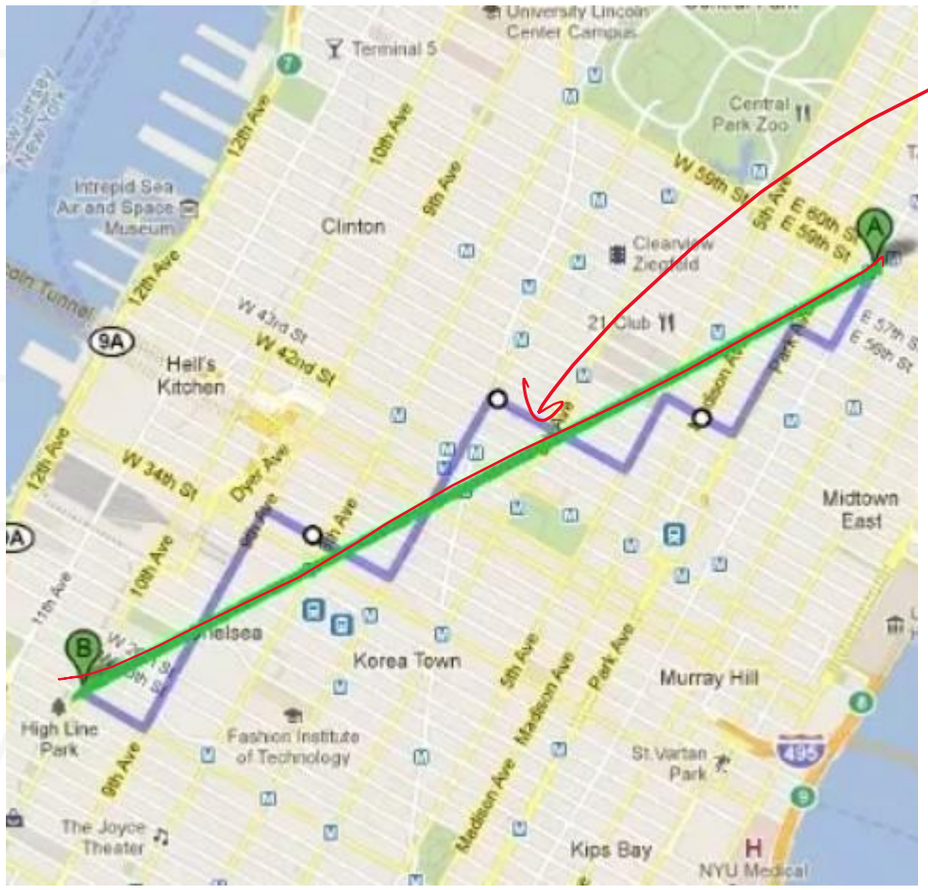
$d(y,x) = 2$

- d: deletion (cost 5)
- s: substitution (cost 1)
- i: insertion (cost 2)

*(these costs are arbitrary)*

if all costs are same, then it is a valid distance metric

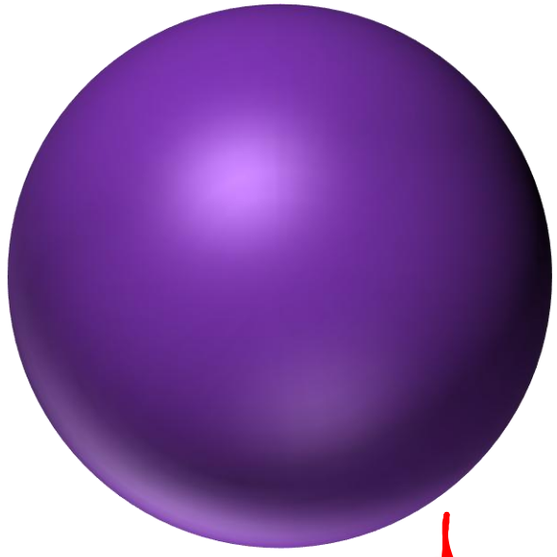
# Some problems with (Euclidean) distance

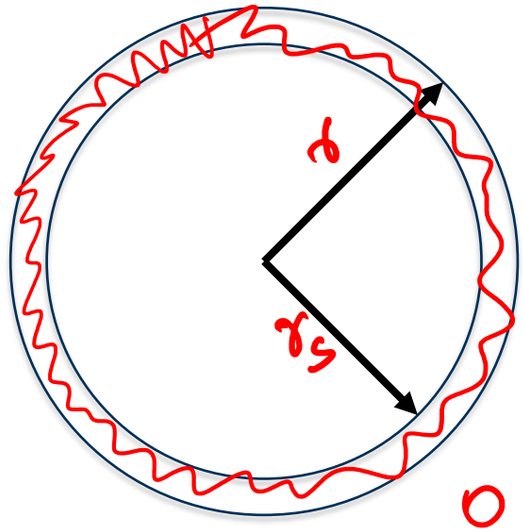



Manhattan distance

$V = \frac{4}{3}\pi r^3$

$V_T = c \cdot r^d$ ← dimension






$V_{shell} = c r^d - c r_s^d$

$\frac{V_{shell}}{V_T} = \frac{c r^d - c r_s^d}{c r^d} = 1 - k \cdot \left(\frac{r_s}{r}\right)^d$

$0 < \frac{r_s}{r} < 1$

As $d \uparrow$, $\left(\frac{r_s}{r}\right)^d \approx 0$

① In very high dimensions, $V_{shell} \approx V_T$

Distance metric loses its meaning in high dimensional space for calculating similarity

② In high-dim. space, model becomes very data hungry.

# Curse of Dimensionality

- Refers to the phenomenon where the efficiency and effectiveness of algorithms deteriorate exponentially as the dimensionality of the data increases.

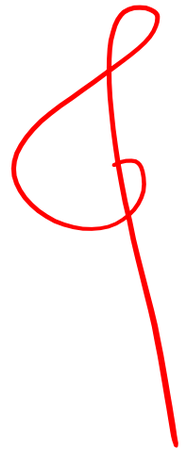
- In high-dimensional spaces, data points become sparse, making it challenging to discern meaningful patterns or relationships due to the vast amount of data required to adequately sample the space.

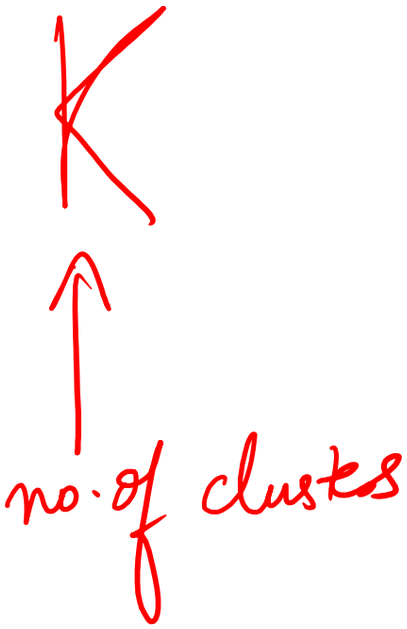
- Curse of Dimensionality significantly impacts ML algorithms in various ways. It leads to increased computational complexity, longer training times, and higher resource requirements. Moreover, it escalates the risk of overfitting and spurious correlations, hindering the algorithms' ability to generalize well to unseen data.

- The curse of dimensionality is both geometric and statistical: distances lose contrast, and learning requires far more data. Dimensionality reduction mitigates both effects.

# Outline

2 inputs :




- Clustering
- Distance functions
- **K-Means algorithm**
- Analysis of K-Means



Georgia Tech

# **Motivating example:** Image compression

Source: Reddit

Communication bottlenecks

Source: Giphy

# How is image data stored?


R → 0 - 255

G → 0 - 255

B → 0 - 255

8 bits

# How is image data stored?

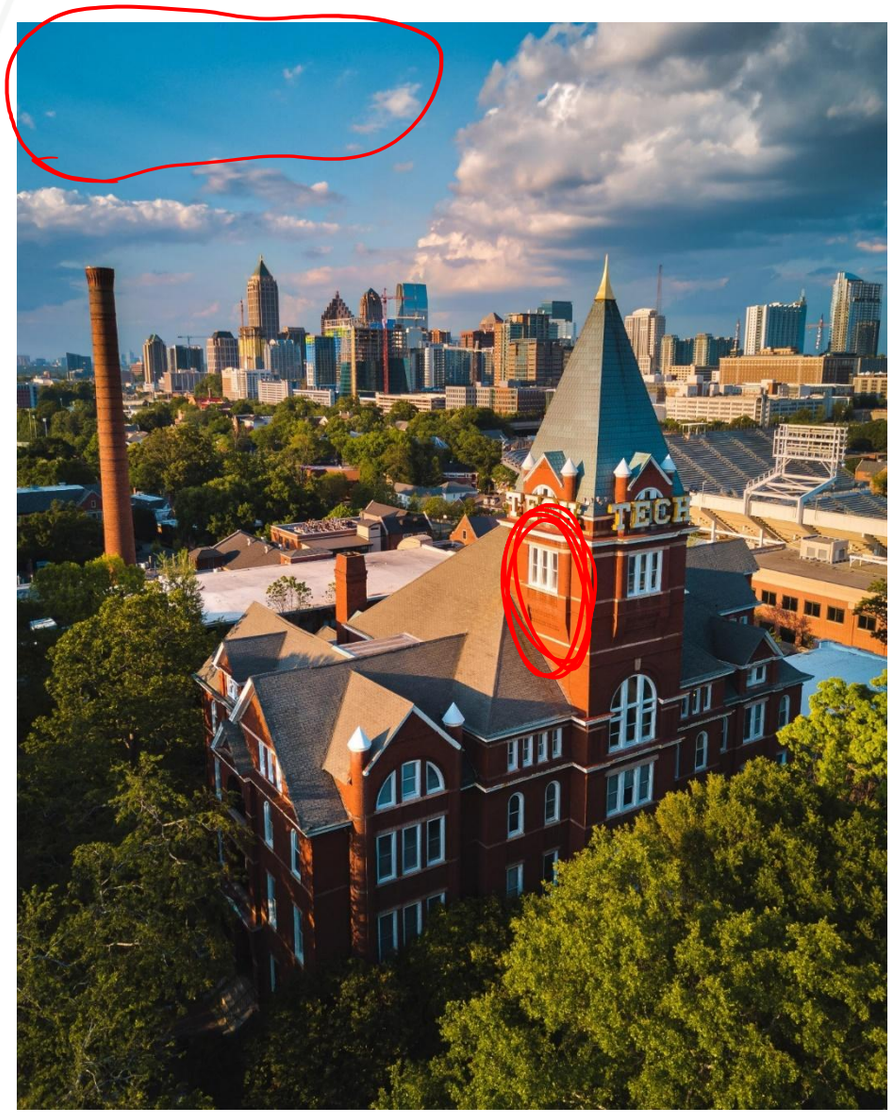
# How is image data stored?

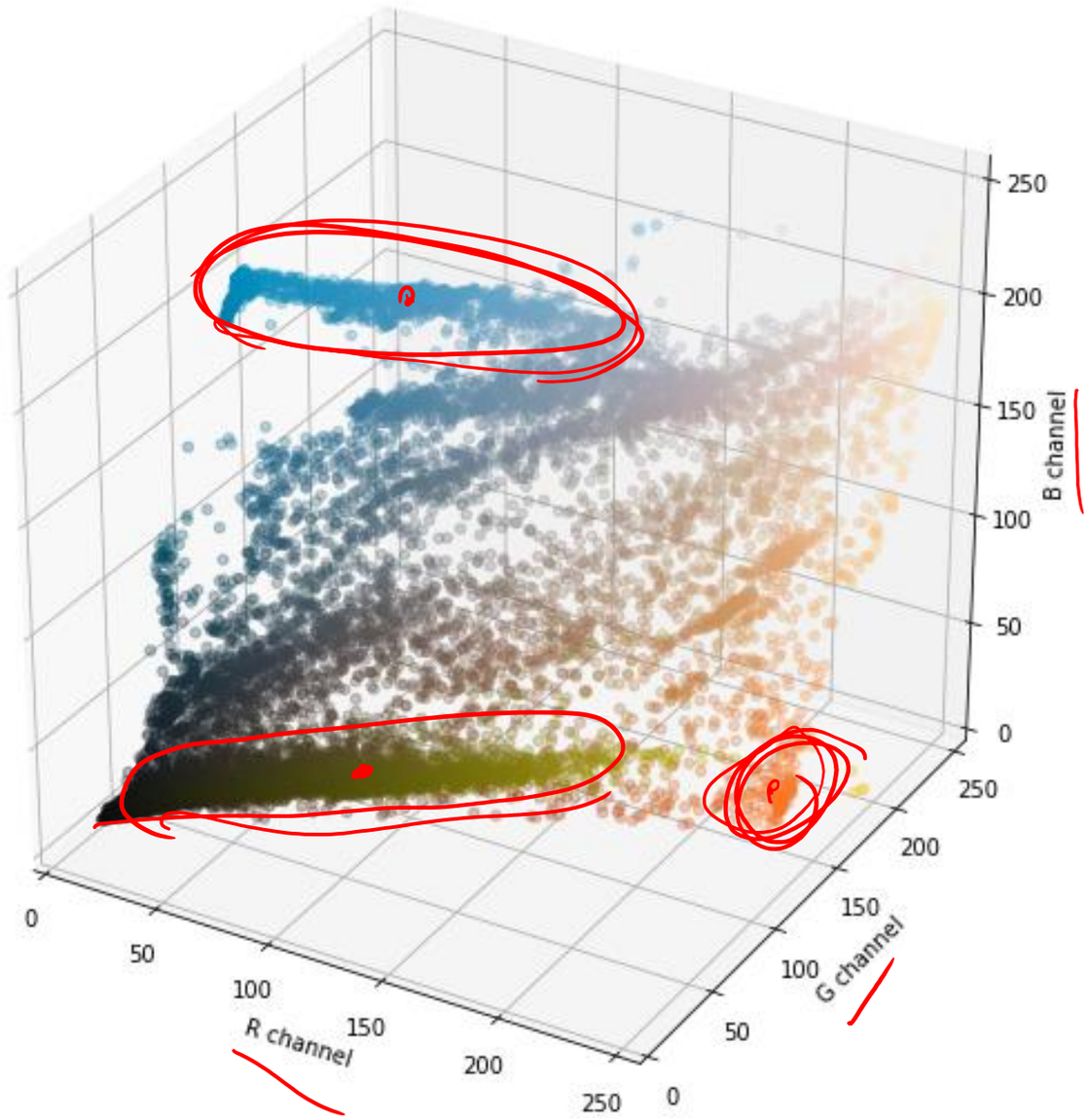
# How is image data stored?

600 X 400 X 3 X 8  bits of storage

# How are colors distributed in RGB space?

K=6





# What if we encoded this image?


Color dictionary

| | R | G | B |
|---|---|---|---|
| 1 | 0 | 100 | 250 |
| 2 | 100 | 100 | 100 |
| 3 | 10 | 250 | 10 |
| 4 | 250 | 50 | 0 |
| 5 | 255 | 255 | 255 |
| 6 | 0 | 0 | 0 |

Centroids of 6 clusters

| | | | | |
|---|---|---|---|---|
| 1 | 1 | … | … | 2 |
| 1 | 1 | … | … | … |
| 1 | … | … | … | … |
| … | … | … | … | 3 |
| … | … | … | … | 3 |
| 3 | 3 | … | 3 | 3 |



Georgia Tech

# What if we encoded this image?

**Original image**

•Size: 600 × 400 pixels

•Each pixel: 3 channels × 8 bits = 24 bits. $600 \times 400 \times 24 = 5{,}760{,}000\ bits$

**Compressed image**

**(a) Dictionary cost (negligible)**

•6 colors

•Each color = 3 channels × 8 bits

$3 \times 6 \times 8 = 144\ bits$ ← negligible

**(b) Pixel indices**

We only need to store **which of the 6 colors** each pixel uses.

•Number of choices = 6

•Bits needed per pixel:

$$\lceil log_2 6 \rceil = 3\ bits$$

So:

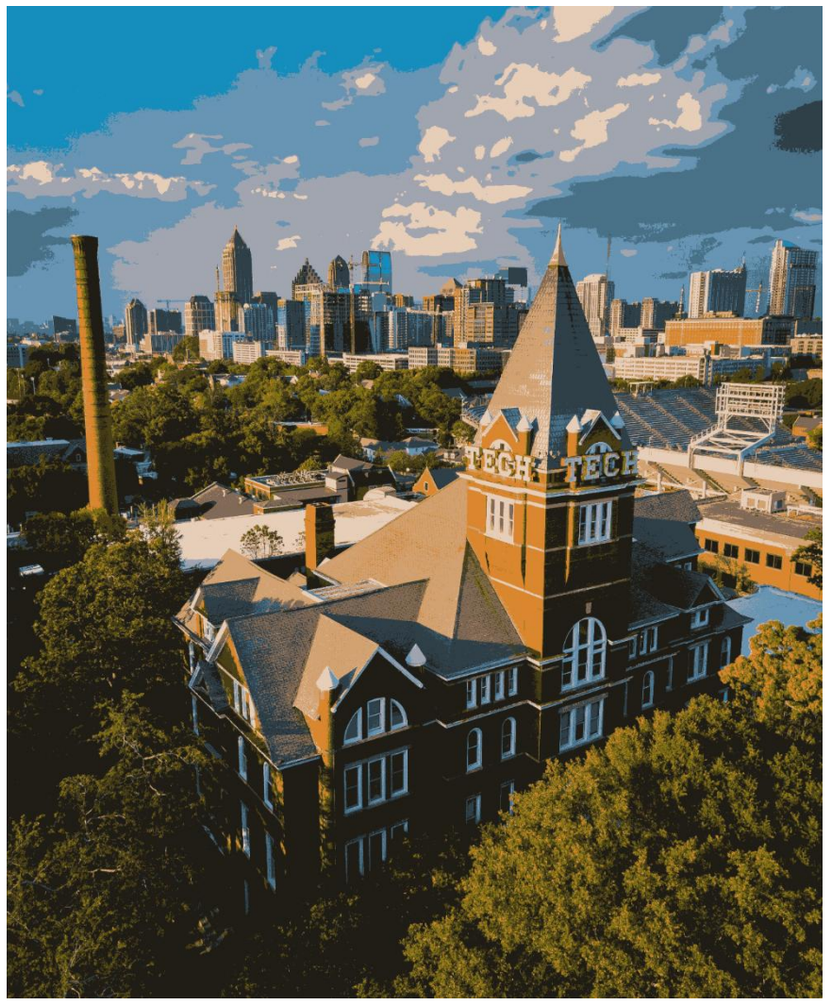
$$600 \times 400 \times 3 = 720{,}000\ bits$$

**4. Compression ratio**

$$\frac{5{,}760{,}000}{720{,}000} = 8$$

# Compressed image

K = 6

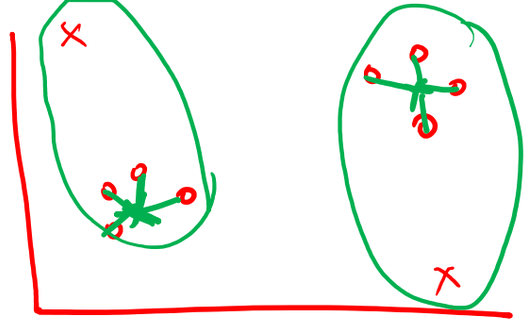


K = 12

K = 24

Motivation 2: We can use compressed image to insert into an ML algorithm for training detecting sky or building or grass. It has much processed input than a complex rgb image input.

# K-Means Algorithm




Step 0:
- Initialize $k$ cluster centers, $\{c_1, c_2, \ldots, c_k\}$ , randomly
- Do

Step 1
- Decide the cluster memberships of each data point, $x_i$ by assigning it to the nearest cluster center (cluster assignment)

EXPECTATION

$$\pi(i) = argmin_{j=1,\ldots,k} \quad \|x_i - c_j\|^2$$

Step 2
- Adjust the cluster centers (center adjustment)

MAXIMIZATION

$$c_j = \frac{1}{|\{i: \pi(i) = j\}|} \sum_{i:\pi(i)} x_i$$

- While any cluster center has been changed

Until convergence ⟶ Distortion Value stops changing

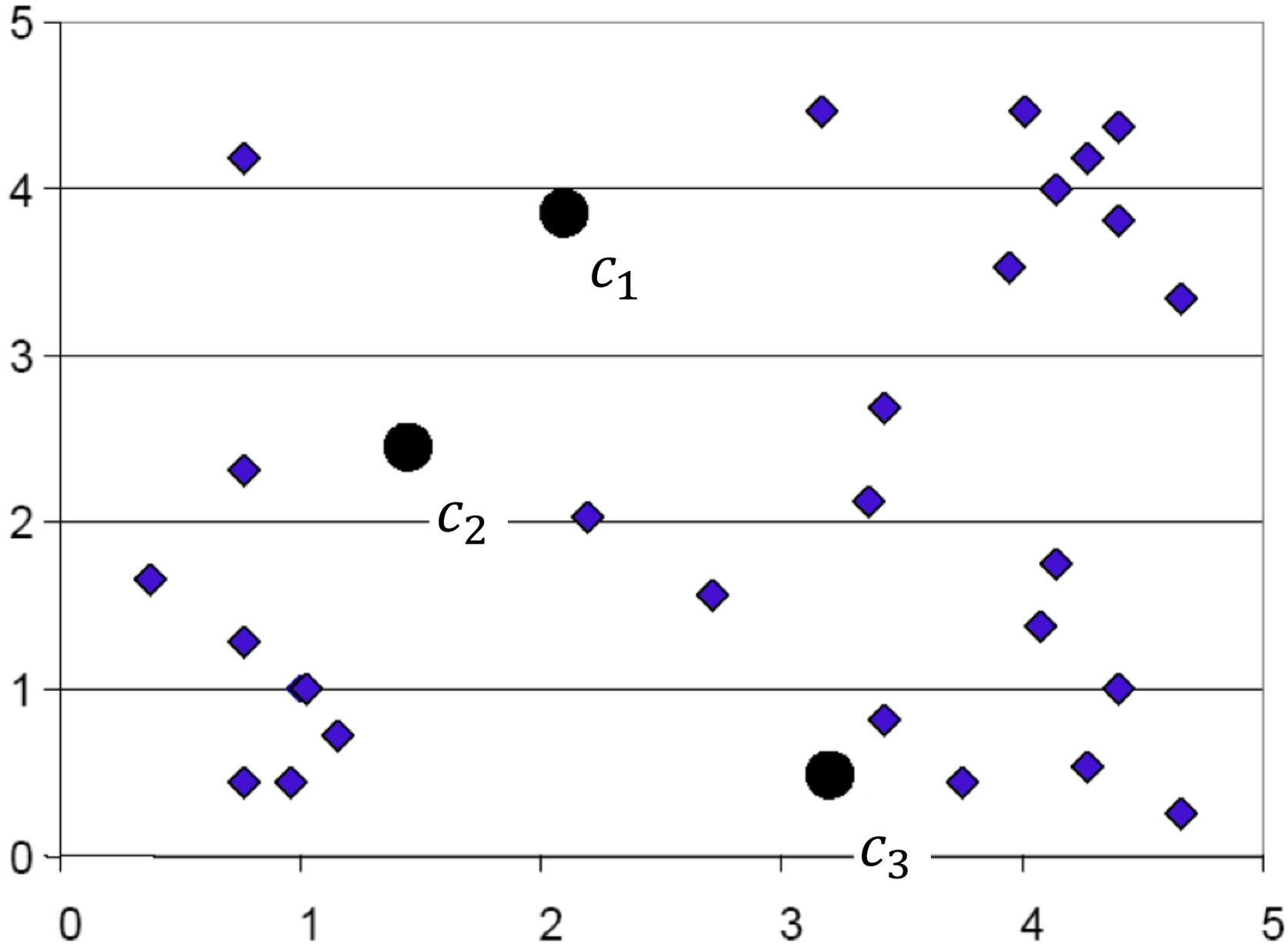
# K-means algorithm (in plain English)

1. Initialize the number of clusters and their centers
2. Compute the distance between each point and each cluster center.
3. Assign each point the cluster id of the nearest cluster center
4. Recompute the cluster centers based on the cluster assignment to each point
5. Repeat steps 2 and 3 until convergence

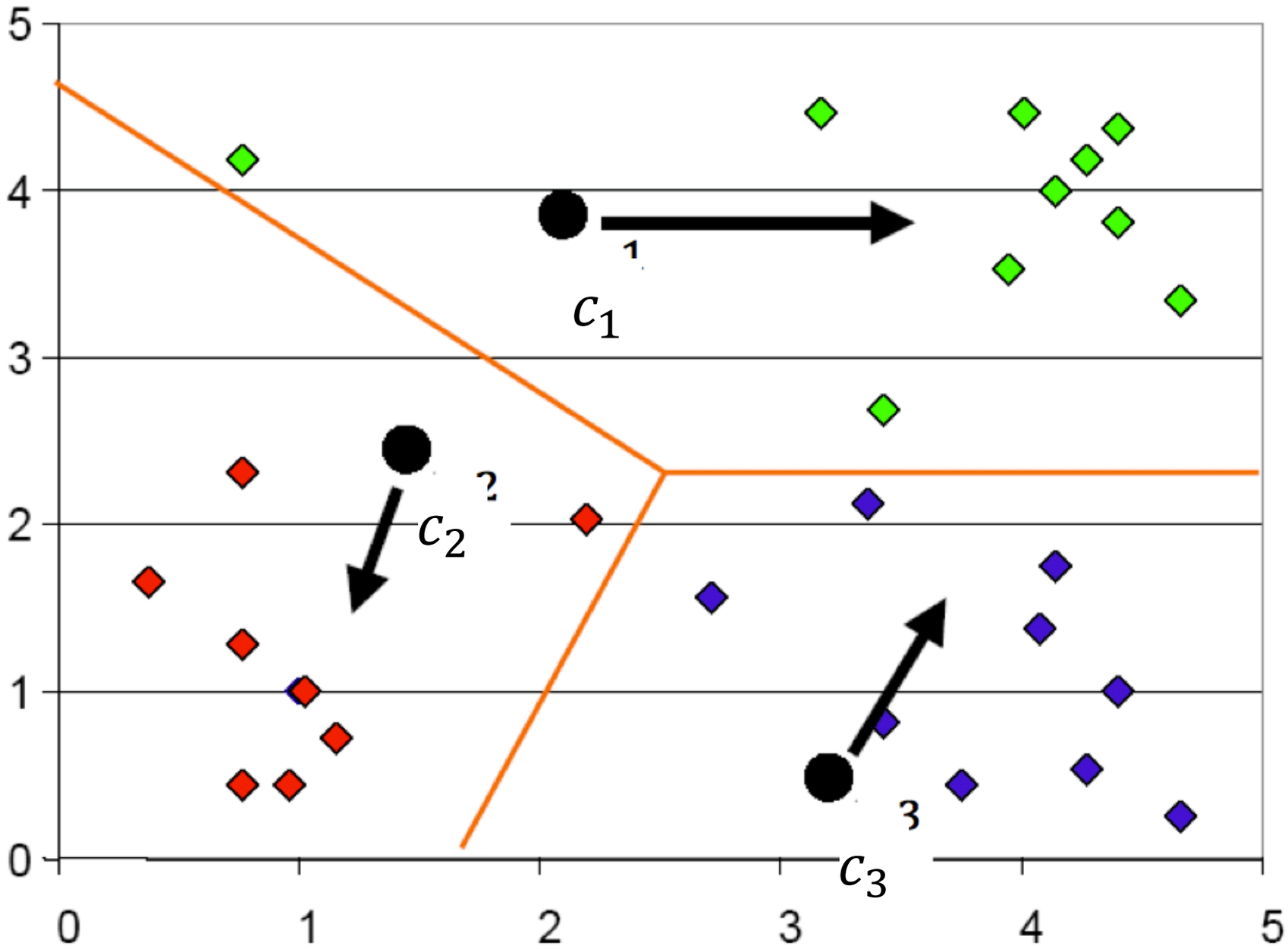
Visualizing K Means

# K-Means: Step 0

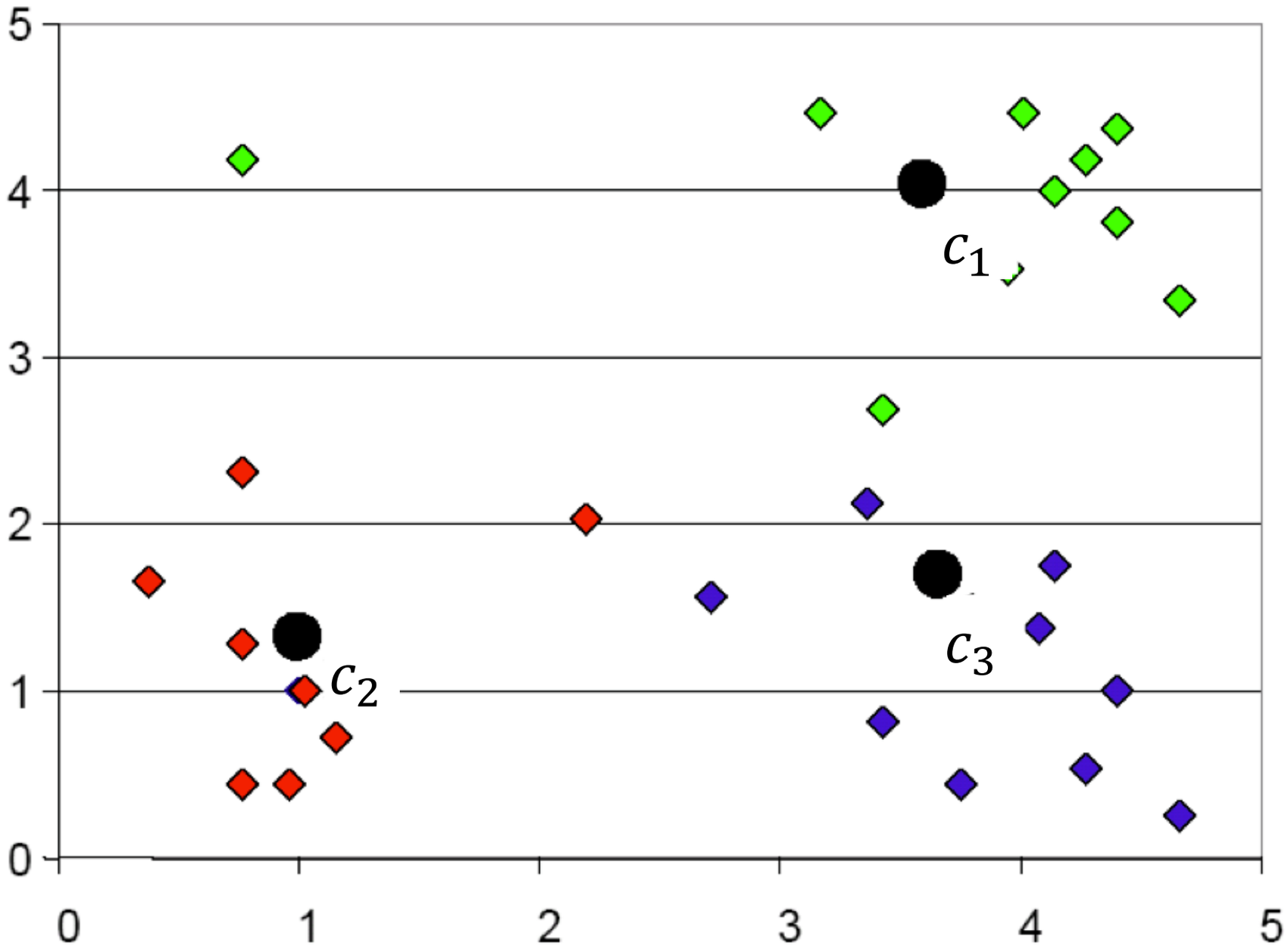
$K=3$



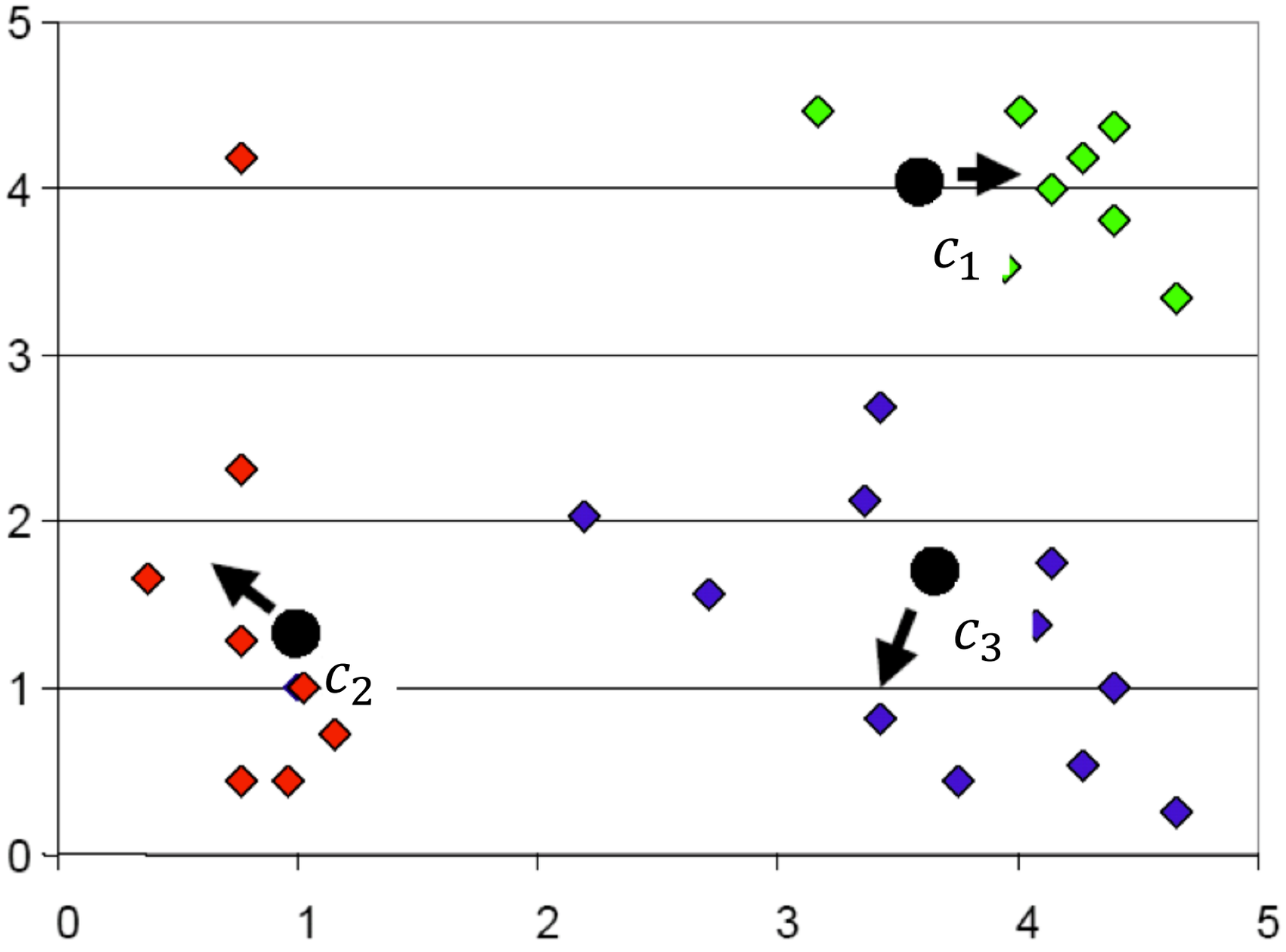
# K-Means: Step 1



# K-Means: Step 2



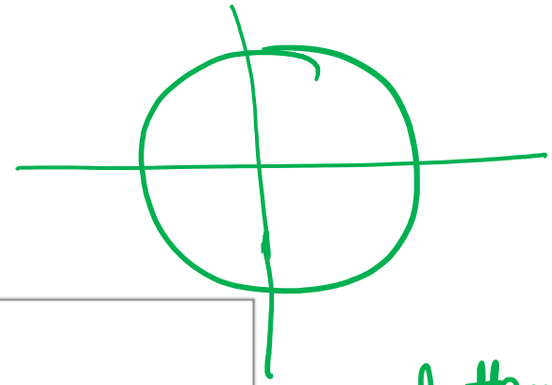
# K-Means: continue



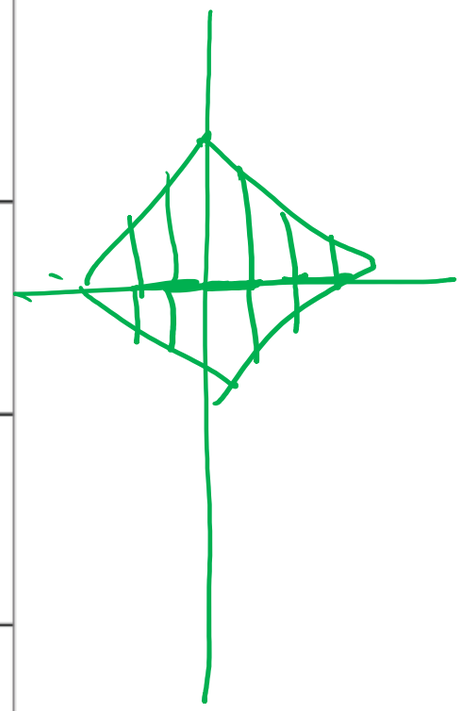
# K-Means: continue

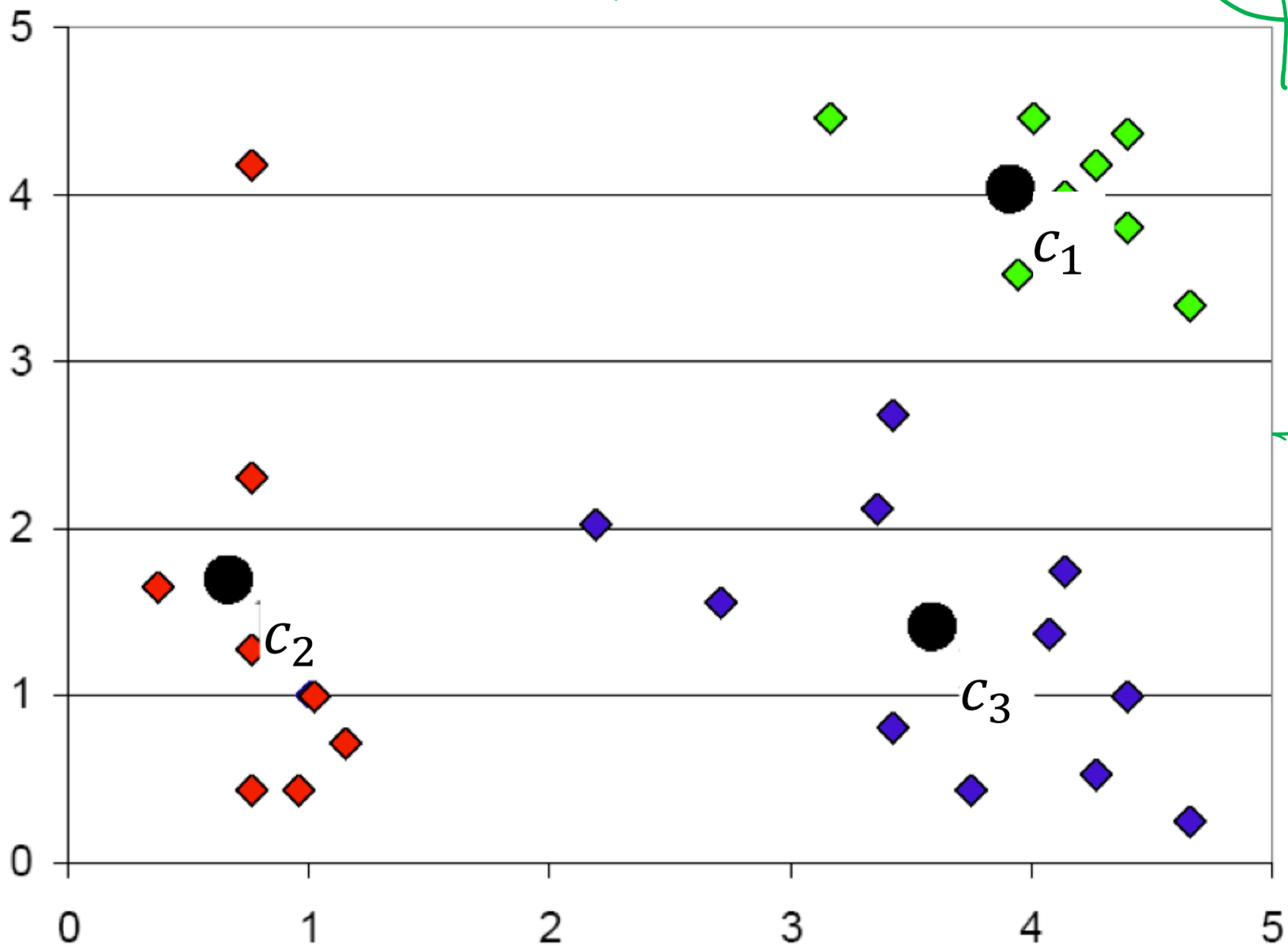
euclidean distance

Preference in kmeans to have spherical clusters



manhattan distance





# Some implementation questions

- Will different initializations lead to different results?
  a. Always
  b. Sometimes
  c. Never

- Will the algorithm always stop after some iteration?
  a. Yes
  b. No (we have to set a maximum number of iterations)

# Formal Statement of the Clustering Problem

- Given $n$ data points, $\{x_1, x_2, \dots, x_n\}\ x \in R^d$
- Find $k$ cluster centers, $\{c_1, c_2, \dots, c_k\}\ c \in R^d$
- And assign each datapoint $i$ to one cluster, $\pi(i) \in \{1, \dots, k\}$
- Such that the averaged square distances from each datapoint to its respective cluster center is small

FIRST OBJECTIVE FUNCTION

Distortion =

$$\min_{c,\pi} \sum_{i=1}^{n} \|x_i - c_{\pi(i)}\|^2$$

Georgia Tech

# Outline

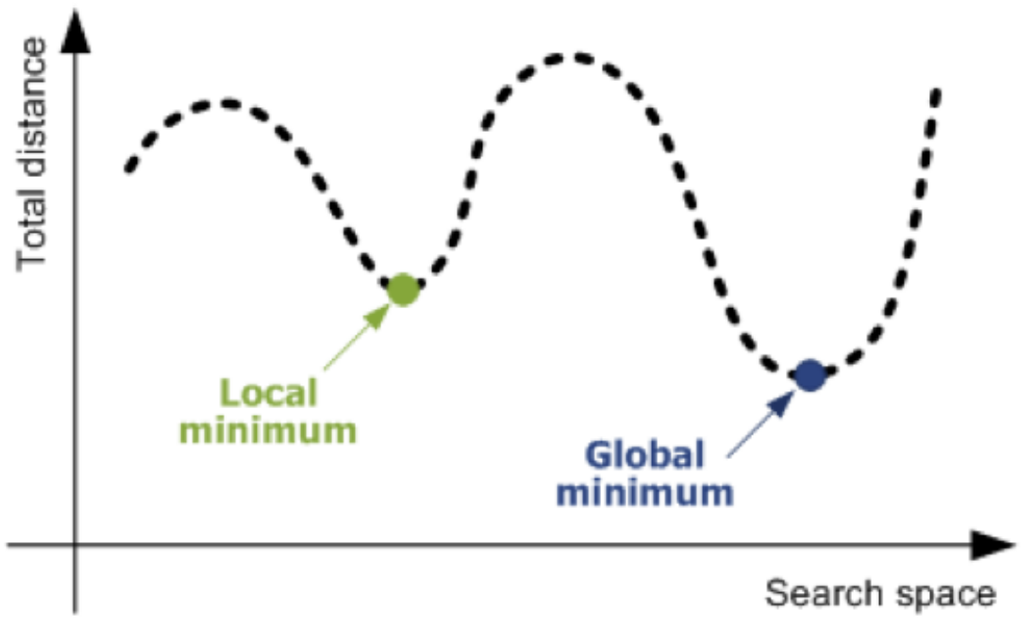
- Clustering
- Distance functions
- K-Means algorithm
- Analysis of K-Means

# Clustering is NP-Hard

- Find $k$ cluster centers, $\{c_1, c_2, \ldots, c_k\}$ c $\in R^{\mathrm{d}}$, and assign each data point $i$ to one cluster, $\pi(i) \in \{1, \ldots, k\}$, to minimize

$$\min_{c,\pi} \sum_{i=1}^{n} \left\| x_i - c_{\pi(i)} \right\|^2$$


- A search problem over the space of discrete assignments
  - For all n data point together, there are $k^{\mathrm{n}}$ possibility
  - The cluster assignment determines cluster centers, and vice versa



- For all n data point together, there are $k^n$ possibility

X = {A,B,C}

n=3 (data points)

k=2 clusters of two members

Cluster 1

ABC
{}
AB
C
AC
B
BC
A

Cluster 2

{}
ABC
C
AB
B
AC
A
BC

$2^3$ possibilities

# Convergence of K-Means

- Will kmeans objective oscillate?

Distortion

$$\min_{c,\pi} \sum_{i=1}^{n} \left\| x_i - c_{\pi(i)} \right\|^2$$

✗ No

- The minimum value of the objective is finite

- Each iteration of kmeans algorithm decrease the objective
  - Cluster assignment step decreases objective
    - $\pi(i) = argmin_{j=1,\ldots,k} \ \left\| x_i - c_{\pi(j)} \right\|^2$ for each data point $i$
  - Center adjustment step decreases objective
    - $c_i = \frac{1}{|\{i:\pi(i)=j\}|} \sum_{i:\pi(i)=j} x_i = argmin_c \sum_{i:\pi(i)=j} \left\| x_i - c_{\pi(j)} \right\|^2$

# Time Complexity

- Assume computing distance between two instances is O(d) where d is the dimensionality of the vectors.
- Reassigning clusters for all datapoints:
  - O(kn) distance computations (when there is one feature)
  - O(knd) (when there is d features)
- Computing centroids: Each instance vector gets added once to some centroid (Finding centroid for each feature): O(nd).
- Assume these two steps are each done once for I iterations: O(Iknd).

$x = \{ \}_d$

$y = \{ \}_d$

$\|x-y\|_2^2 = O(d)$

n datapoints $= O(nd)$

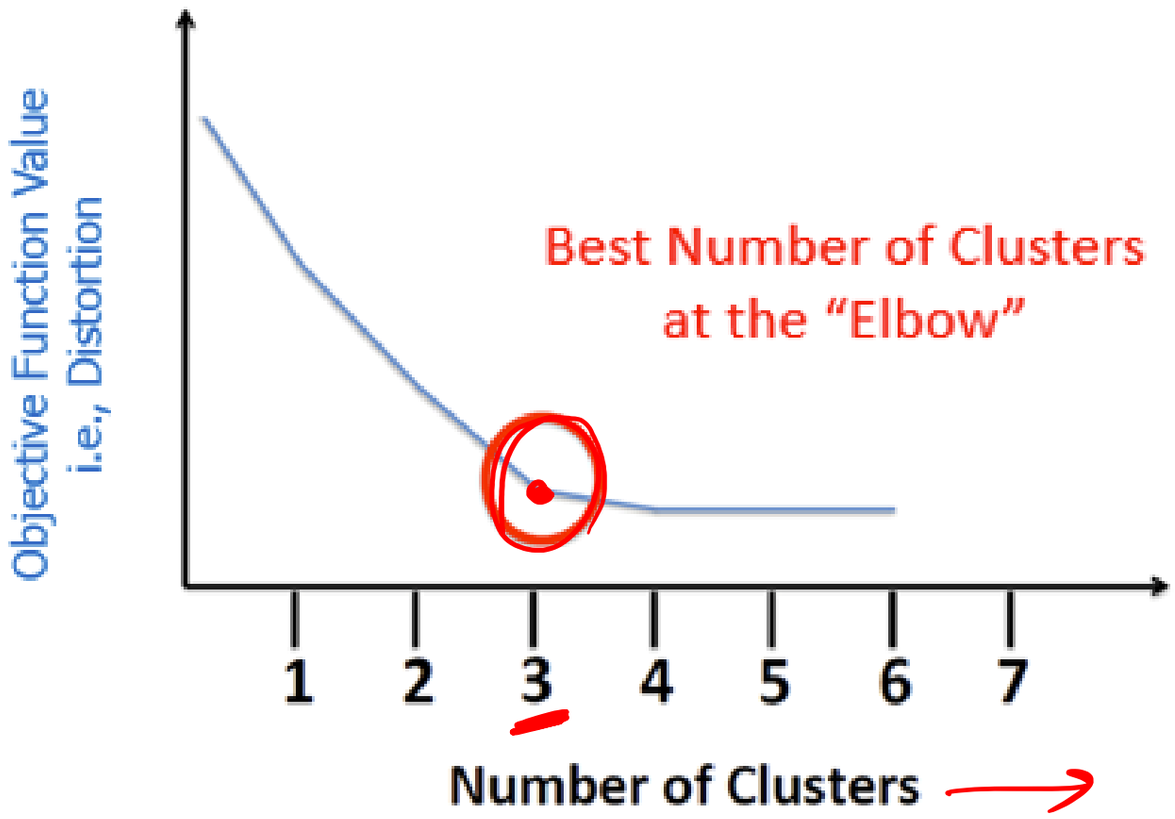
K centers $= O(knd)$

$\Sigma$ iterations $= O(Iknd)$

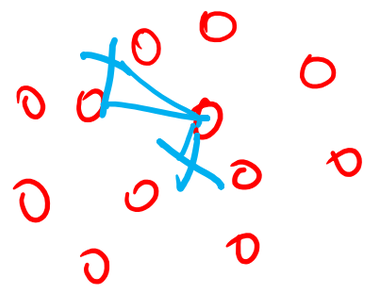
Slide credit: Ray Mooney.

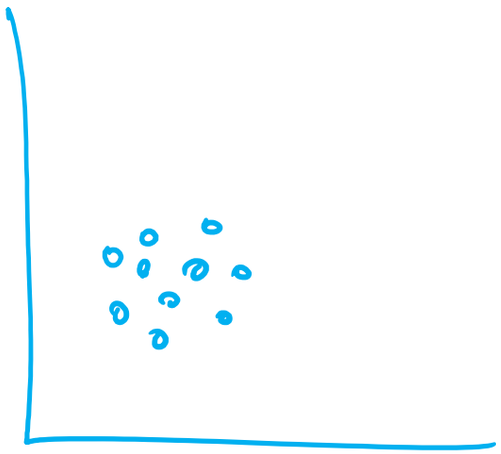
# How to Choose K?

Hyperparameters




k=1,2
Underclustering





Overclustering

**Distortion score:** computing the sum of squared distances from each point to its assigned center

Image credit: Dileka Madushan.

# Limitations of K-Means


- Local minima
  - Can we fix it?

Reinitializing & keep running
pick the one with min. distortion

- Cluster shape
  - Can we fix it?

K means → hard clustering

GMM → { Soft clustering based on probability

- Susceptibility to outliers
  - Can we fix it?

No