



# K-Means Clustering



# K-Means Clustering

- Section Overview
  - Understanding Clustering
  - History and Intuition of K-Means
  - Mathematical Theory of K-Means
  - Example of K-Means
  - K-Means Project Exercise
  - K-Means Project Exercise Solution



# K-Means Clustering

- **Important Note:**

- *Do not confuse K-Means with KNN!*
- *The “K” is completely different in both algorithms, they solve completely different problems and are not related in any way!*



# Let's get started!



# Clustering

General Concepts



# Clustering

- Clustering uses **unlabeled data** and looks for similarities between groups (clusters) in order to attempt to segment the data into separate clusters.
- Keep in mind that we don't actually know the true correct label for this data!



# Clustering

- Imagine an example data set:

X1	X2
2	4
6	3
...	...
1	2



# Clustering

- Notice again we only have features!

X1	X2
2	4
6	3
...	...
1	2





# Clustering

- How could we cluster this data together?

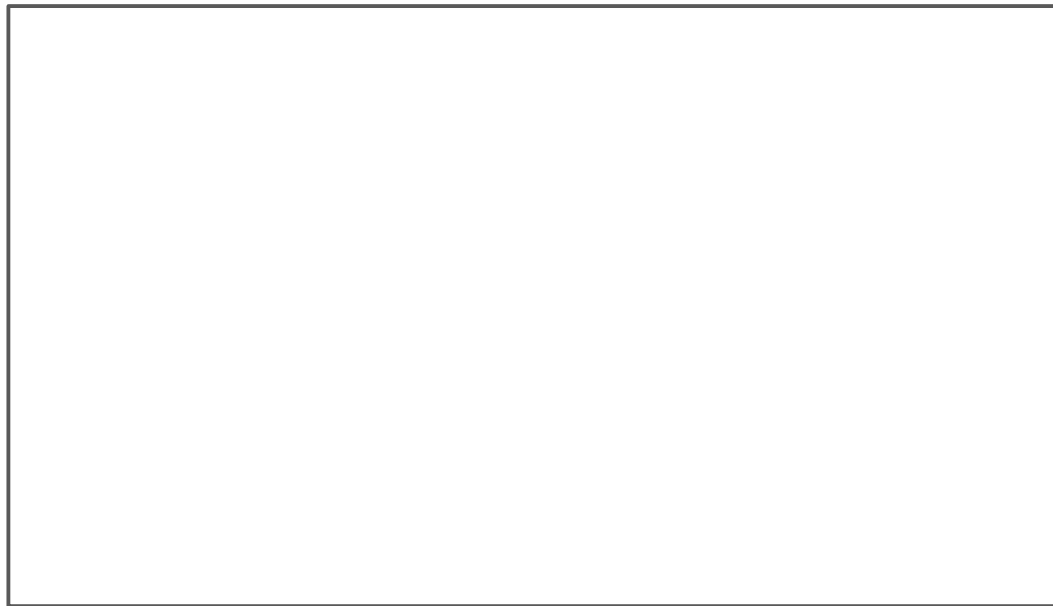
X1	X2
2	4
6	3
...	...
1	2



# Clustering

- Could simply plot and discover patterns:

x2

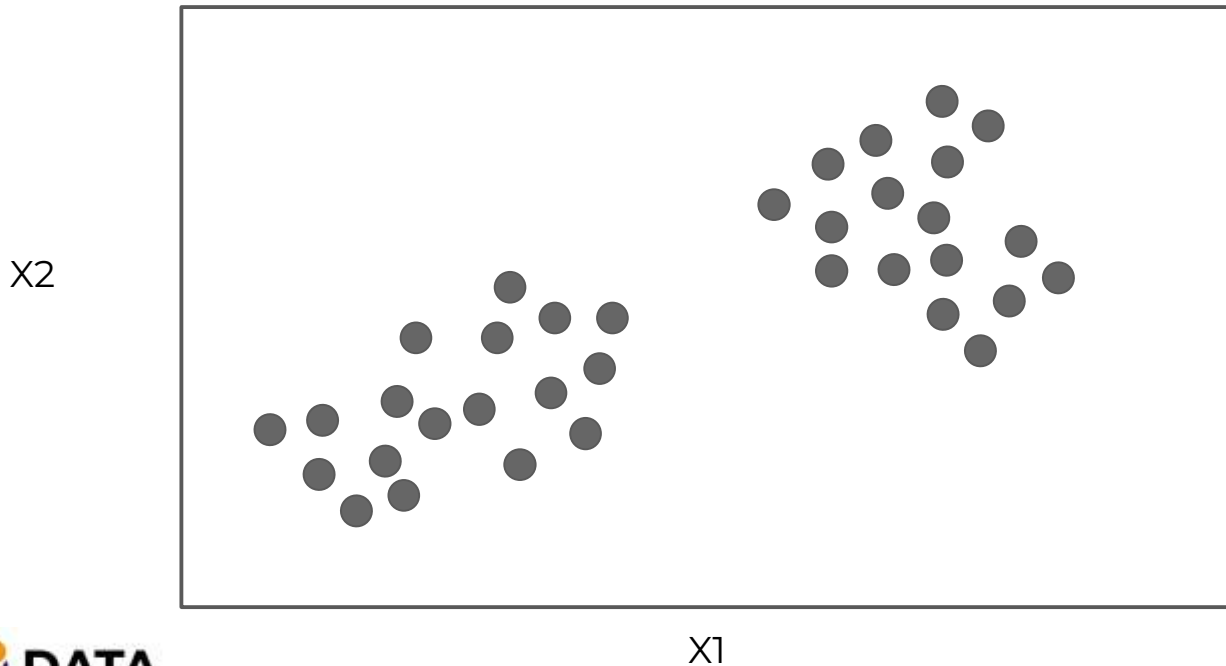


x1



# Clustering

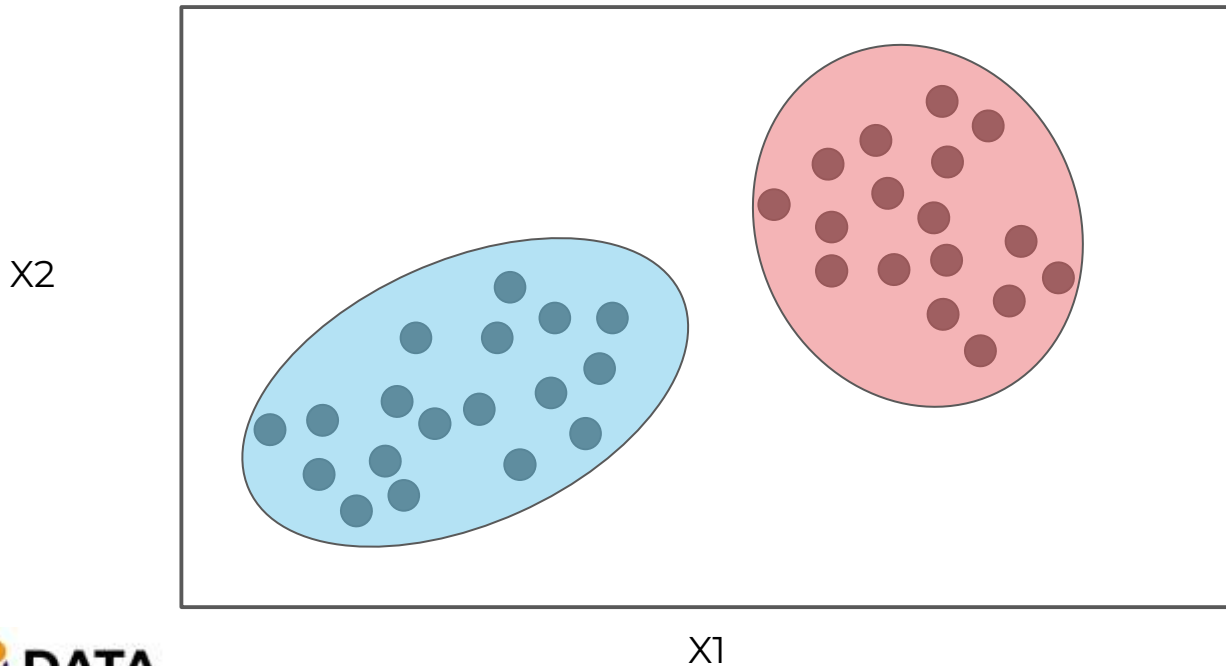
- Here we intuitively see 2 groupings:





# Clustering

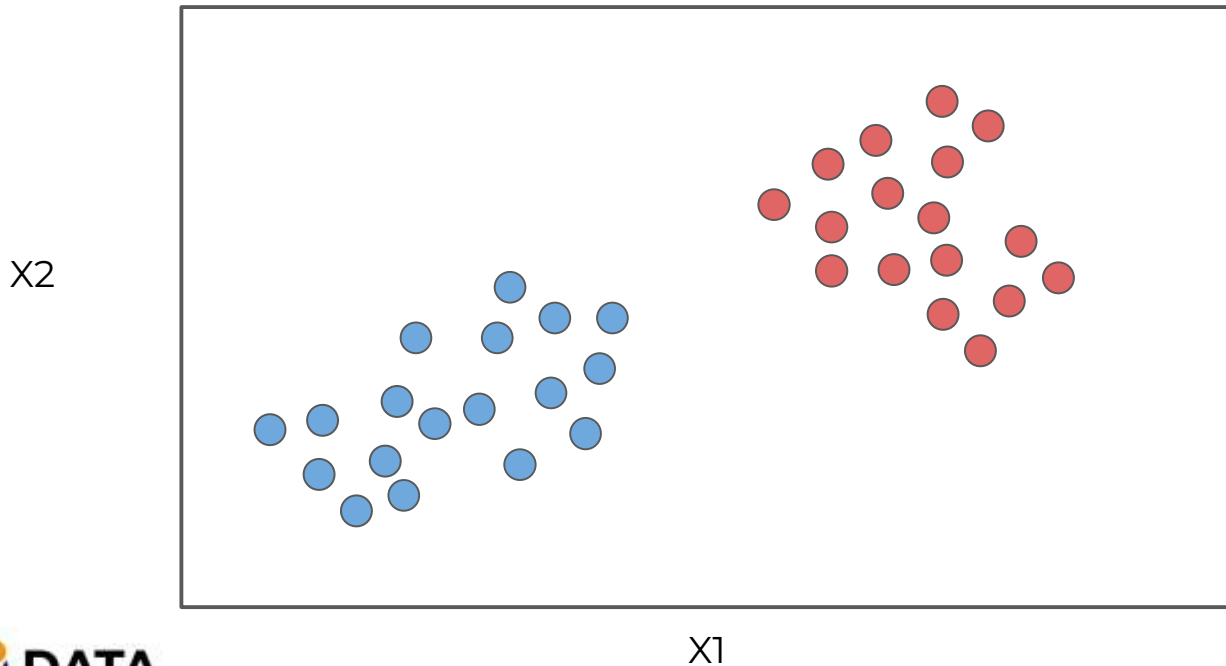
- Note how distance is the intuitive metric:





# Clustering

- We could then assign clusters:





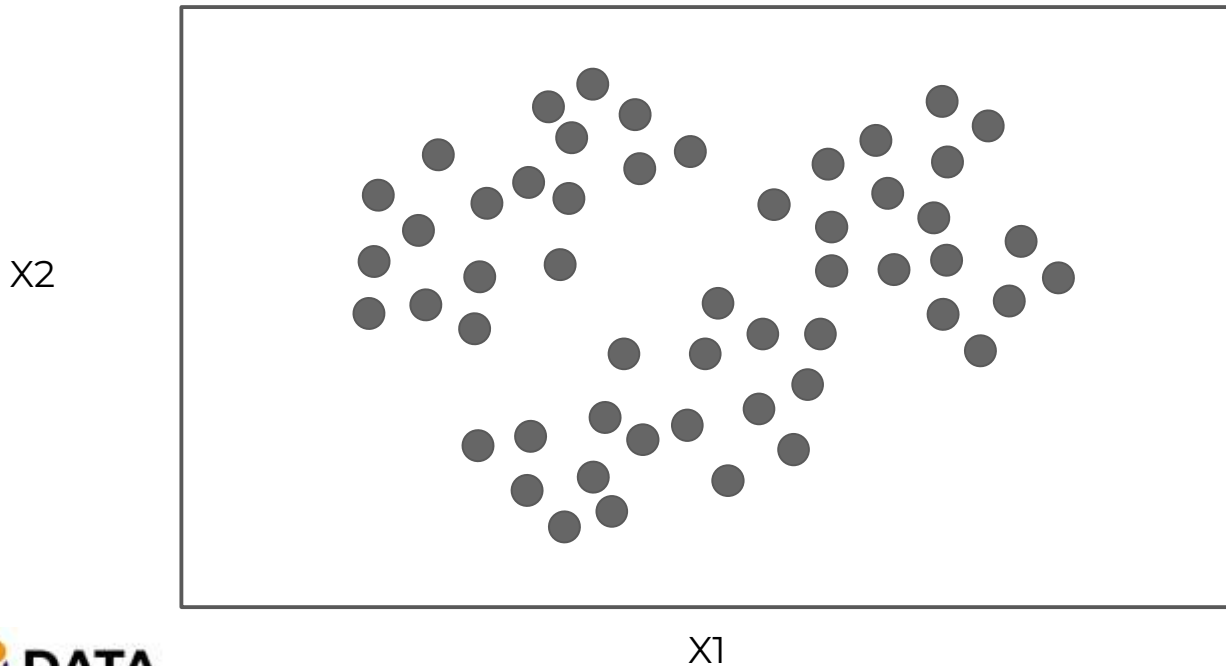
# Clustering

- Notice how we don't actually know for sure if this is a correct way of grouping together these data points, there was no correct label to begin with!
- And what about situations that are not so obvious or multi-dimensional?



# Clustering

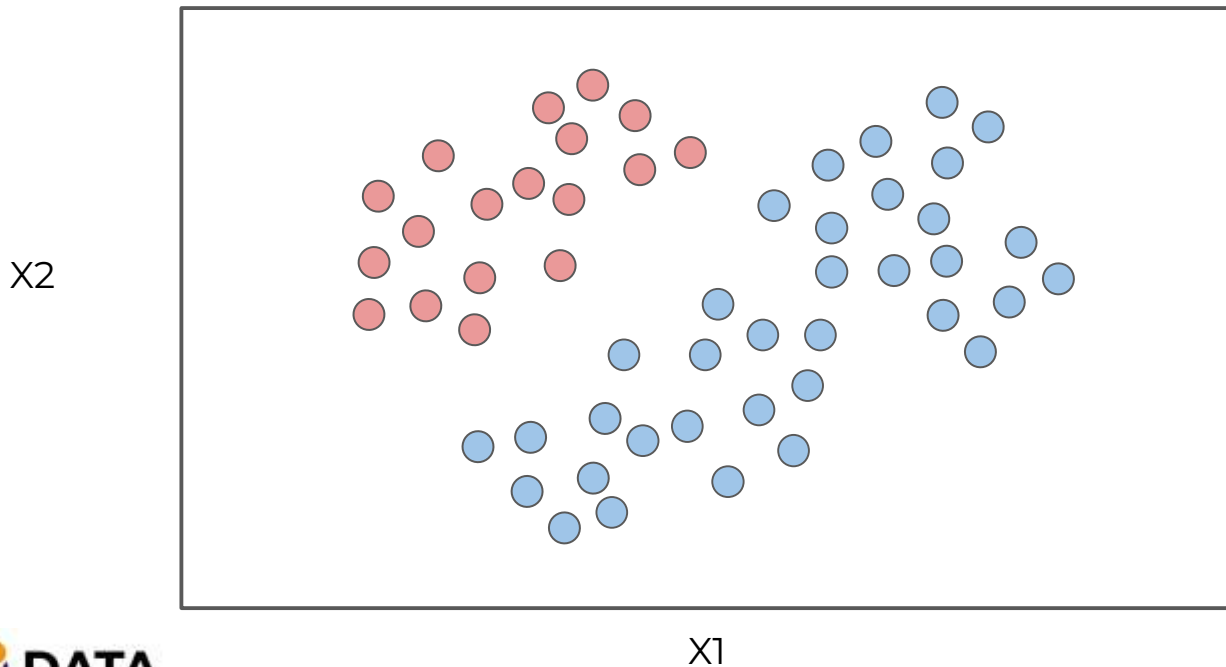
- 2 or 3 clusters could both be reasonable:





# Clustering

- 2 or 3 clusters could both be reasonable:

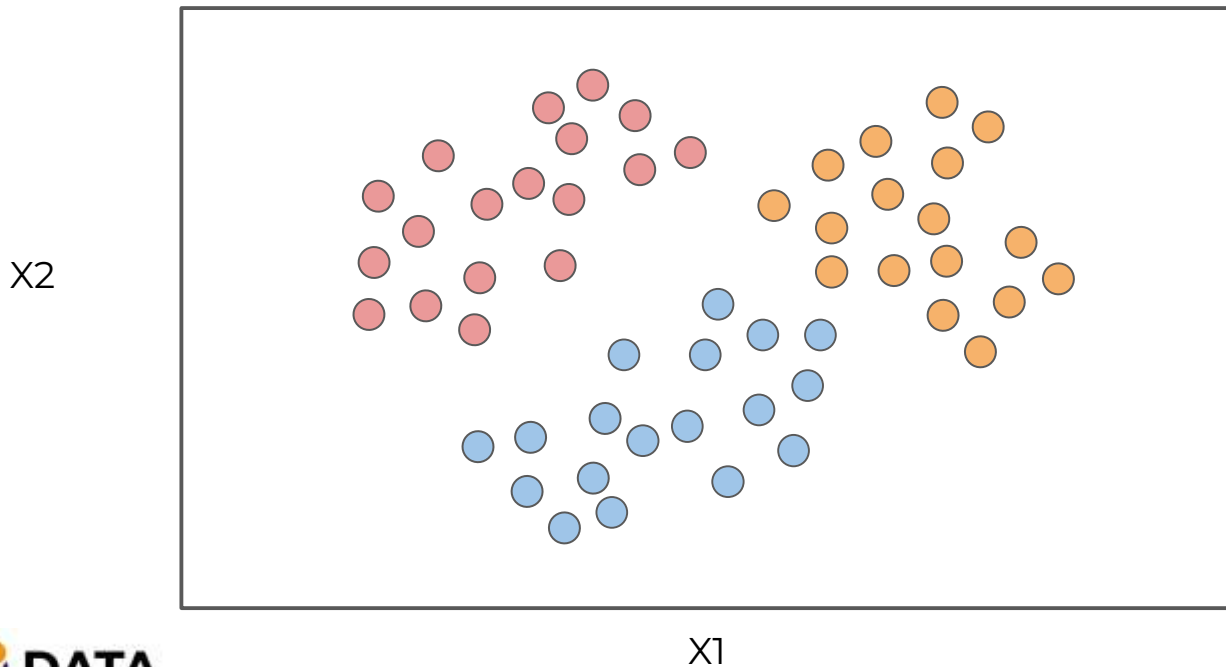






# Clustering

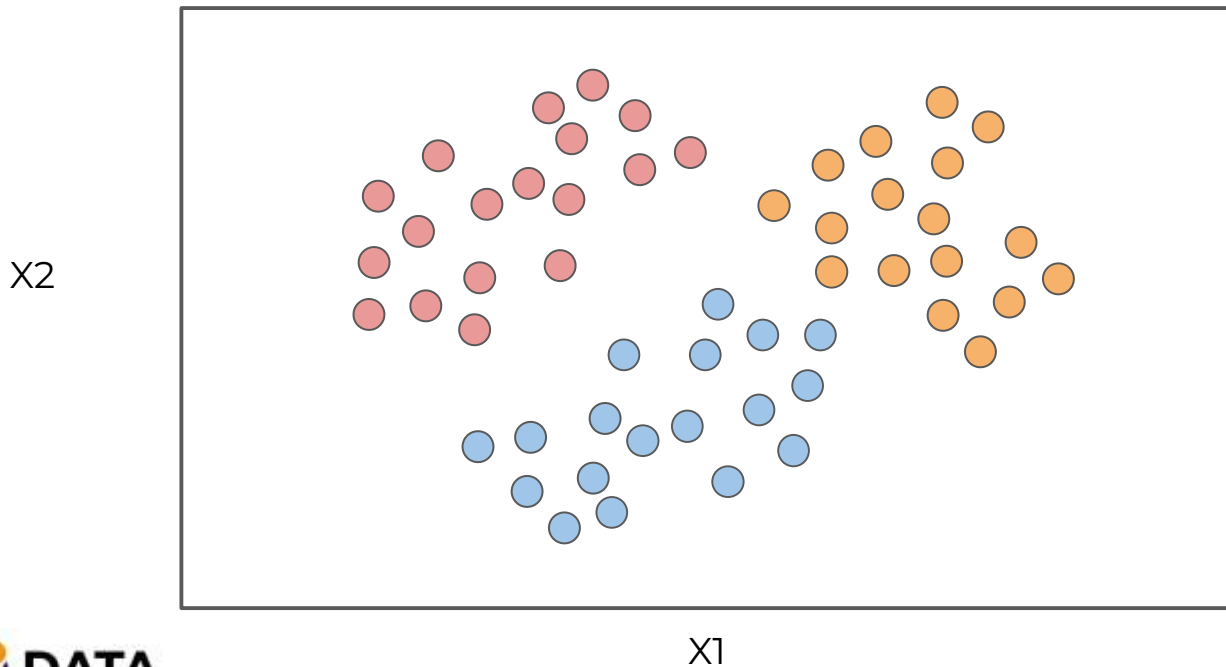
- 2 or 3 clusters could both be reasonable:





# Clustering

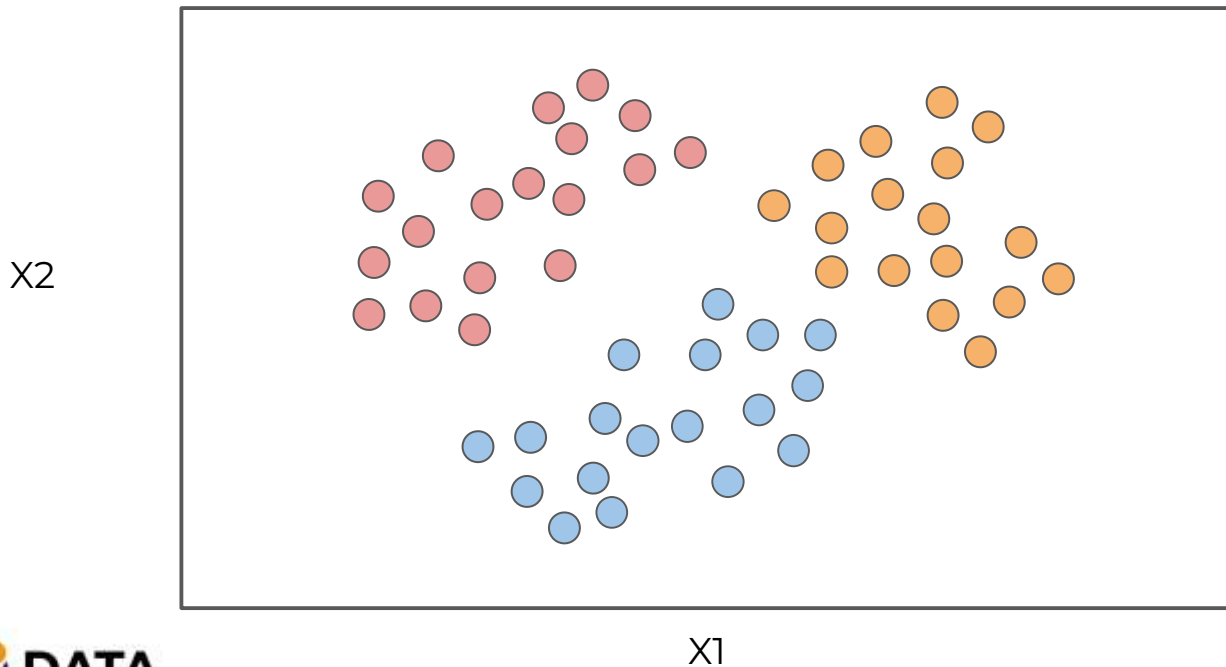
- Different methods can be used to decide!





# Clustering

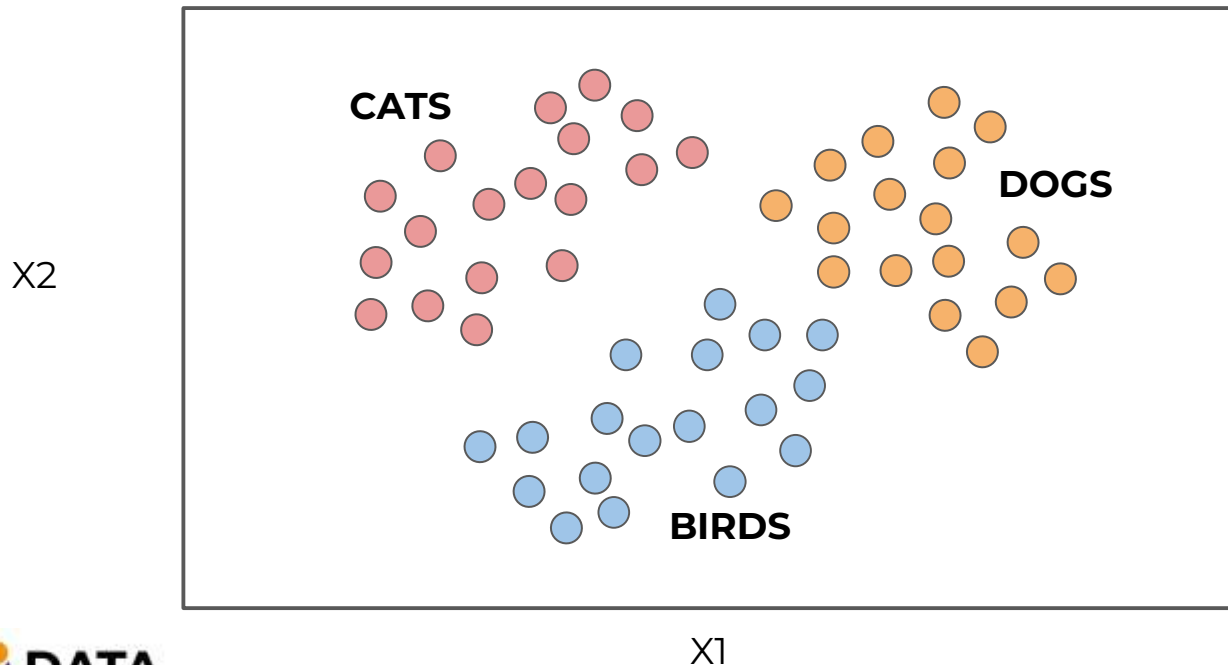
- Clustering doesn't "label" these for you!





# Clustering

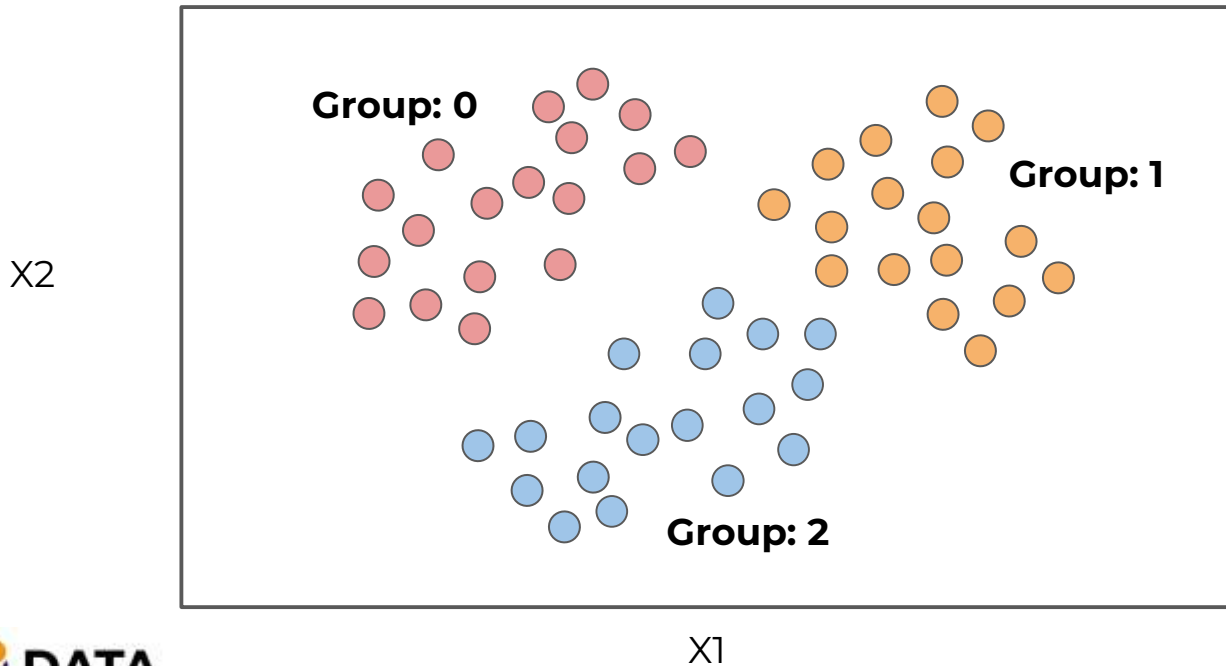
- Clustering doesn't "label" these for you!





# Clustering

- Clustering doesn't "label" these for you!





# Clustering

- Main Clustering Ideas:
  - Use features to decide which points are most similar to other points.
  - Realize that there is no final correct **y** label to compare cluster results to.
  - We can think of clustering as an unsupervised learning process that “discovers” potential labels.



# Clustering

- Unsupervised Learning Paradigm Shift:
  - *What about a new unlabeled data point?*
  - *How do we assign it to a cluster?*
  - *Was it the correct cluster for assignment?*



# Clustering

- Unsupervised Learning Paradigm Shift:
  - *How do we assign a new data point to a cluster?*
    - Different approaches depending on the unsupervised learning algorithm used.
    - Use features to assign most appropriate cluster.





# Clustering

- Unsupervised Learning Paradigm Shift:
  - *How do we assign a new data point to a cluster?*
    - Just as before, no way to measure if this was the “correct” assignment.



# Clustering

- Unsupervised Learning Paradigm Shift:
  - *If we've discovered these new cluster labels, could we use that as a **y** for supervised training?*
    - Yes! We can use unsupervised learning to discover possible labels, then apply supervised learning on new data points.



# Clustering

- Unsupervised Learning Paradigm Shift:
  - *If we've discovered these new cluster labels, could we use that as a **y** for supervised training?*
    - What's the trade-off?



# Clustering

- Unsupervised Learning Paradigm Shift:
  - *If we've discovered these new cluster labels, could we use that as a **y** for supervised training?*
    - Clustering doesn't tell you what these new cluster labels represent, no real way of knowing if these clusters are truly significant.



# Clustering

- Clustering ideas still to come:
  - How do we decide which number of clusters is best?
  - Do we decide or let the algorithm decide?
  - How can we measure “goodness of fit” for clustering without a **y** label for comparison?



# Clustering

- Machine Learning as an art:
  - *What is ground truth?*
  - *What trade-offs are we making by using unsupervised learning as a substitute for ground truth of the  $y$  label that was not given?*



# Clustering

- Keep these ideas in mind as we discover the different algorithms behind clustering!
- Also keep in mind that it's much harder to compare unsupervised algorithms against each other due to the lack of ground truth based performance metrics (e.g. can't use accuracy or RMSE).



# K-Means Clustering

History, Intuition and Theory





# K-Means Clustering

- Let's briefly go through a tour of the timeline of the development of K-Means clustering.
- Afterwards, we'll dive deeper into the main method and concepts.



# K-Means Clustering

- Hugo Steinhaus was the mathematician who began laying out the groundwork for clustering methods.





# K-Means Clustering

- 1940s-1950s: Hugo Steinhaus Polish mathematician and professor.
- During WWII he was in hiding from Nazi occupation, yet still taught clandestine math classes from memory, since higher education for Poles was forbidden!





# K-Means Clustering

- He even established statistical methods for estimating German casualty rates based simply on relative frequencies of obituaries in local newspapers.
- After WWII he helped establish the mathematics department at the University of Wrocław.





# K-Means Clustering

- It was at the University of Wrocław where he began to develop the ideas that would become the foundation of K-Means clustering!
- *Check out his Wikipedia page for more interesting stories from his incredible life!*





# Clustering

- 1957: Stuart Lloyd of Bell Labs developed the standard algorithmic technique for clustering related to K-Means.
- However Lloyd did not publish his technique in a journal until 1982.
- In 1965 Edward W. Forgy published the clustering technique, thus both Lloyd and Forgy are credited with the development.



# Clustering

- 1967: James MacQueen of UCLA develops and publishes “*Some methods for classification and analysis of multivariate observations*”.
- This is the first publication to introduce the term “K-Means” as a sequential clustering algorithm.



# Clustering

- So how does K-Means clustering actually work?
  - The main concept is actually very simple!
  - Let's walk through an example of clustering unlabeled data.





# Clustering

- First a set of properties each point must satisfy:
  - Each point must belong to a cluster.
  - Each point can only belong to one cluster (no single point can belong to multiple clusters).



# Clustering

- So how does K-Means clustering actually work?

x2



x1



# Clustering

- Note: For visualization purposes, we'll work with a simple dataset with only 2 features.

X2



X1



# Clustering

- The process shown here easily extends to **N** feature dimensions.

x2



x1



# Clustering

- Step 1: Start with unlabeled data (only features).

X2



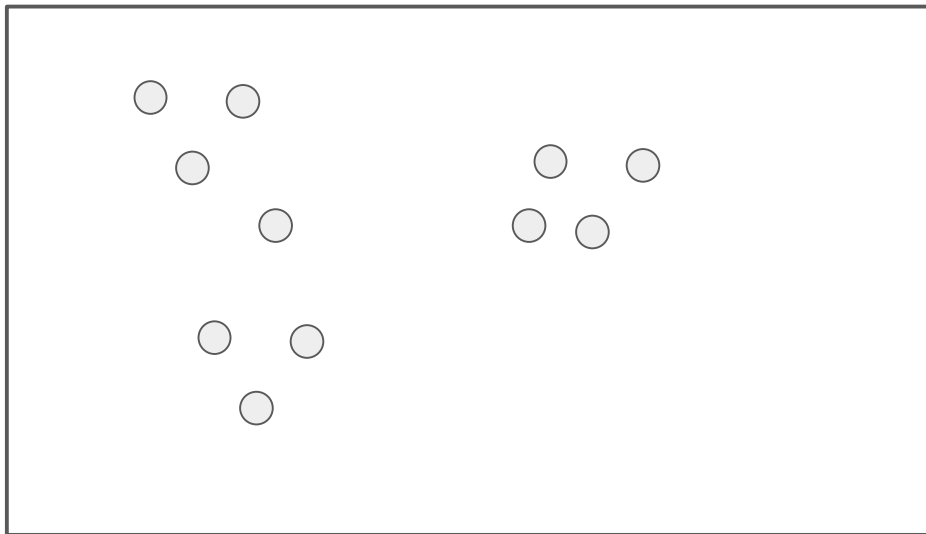
X1



# Clustering

- Step 0: Start with unlabeled data (only features).

x2



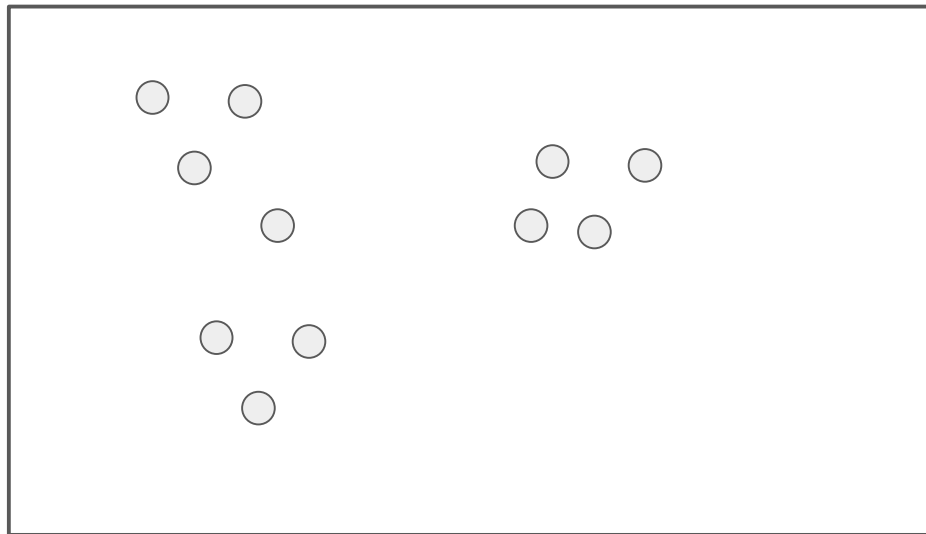
x1



# Clustering

- Note: *If we had the group labels, it wouldn't make sense to cluster!*

x2



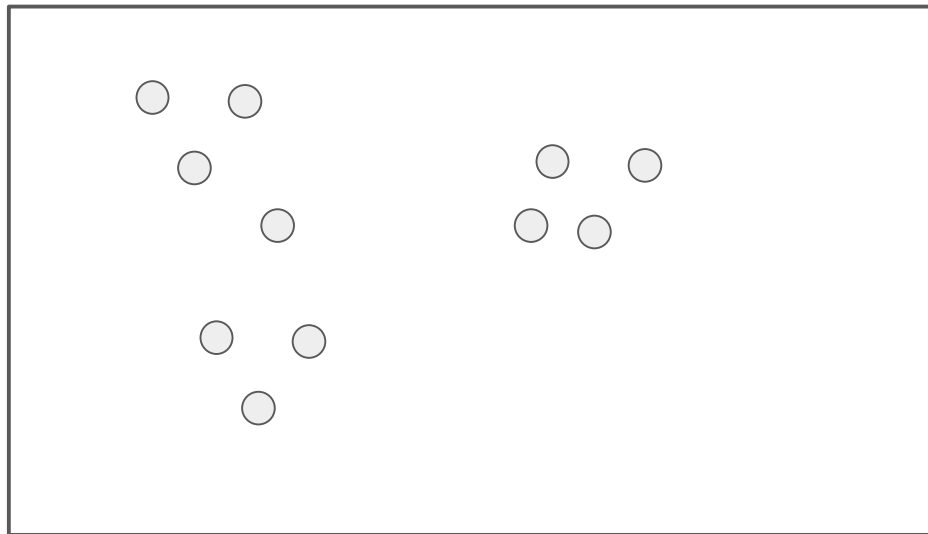
x1



# Clustering

- Step 1: Choose the number of clusters to create (this is the K value).

x2



x1

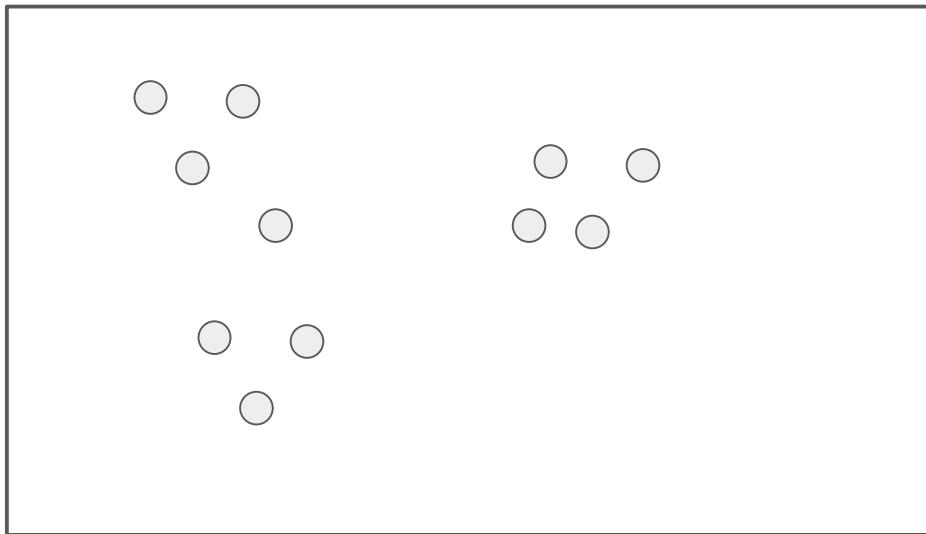




# Clustering

- Step 1: We'll choose  $K=3$ . Note in most situations you won't visualize the data!

x2



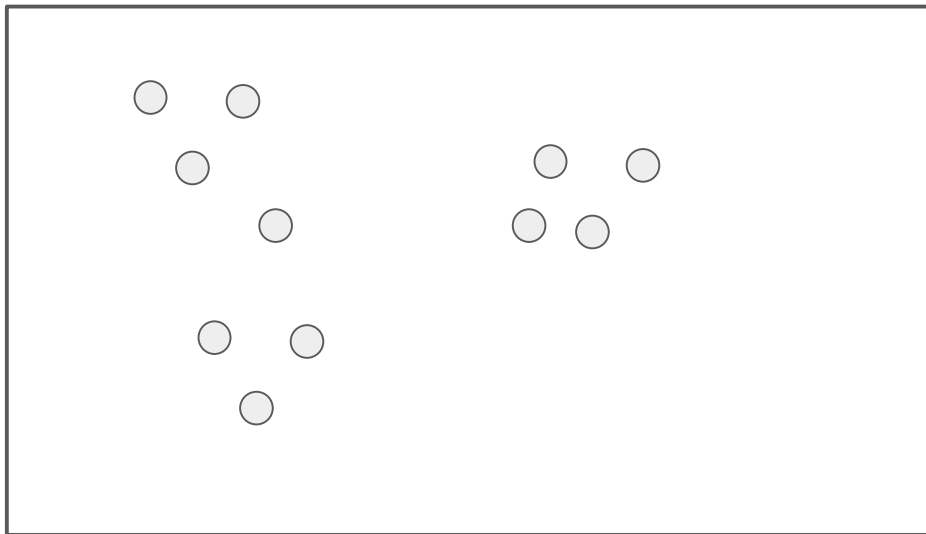
x1



# Clustering

- Step 2: Randomly select  $K$  distinct data points.

x2



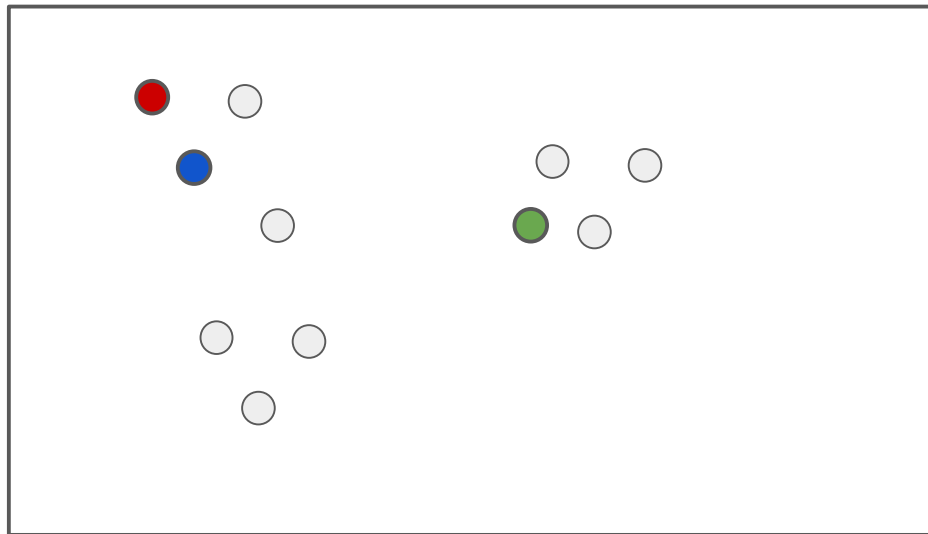
x1



# Clustering

- Step 2: Randomly select K distinct data points. Our  $K=3$ :

x2



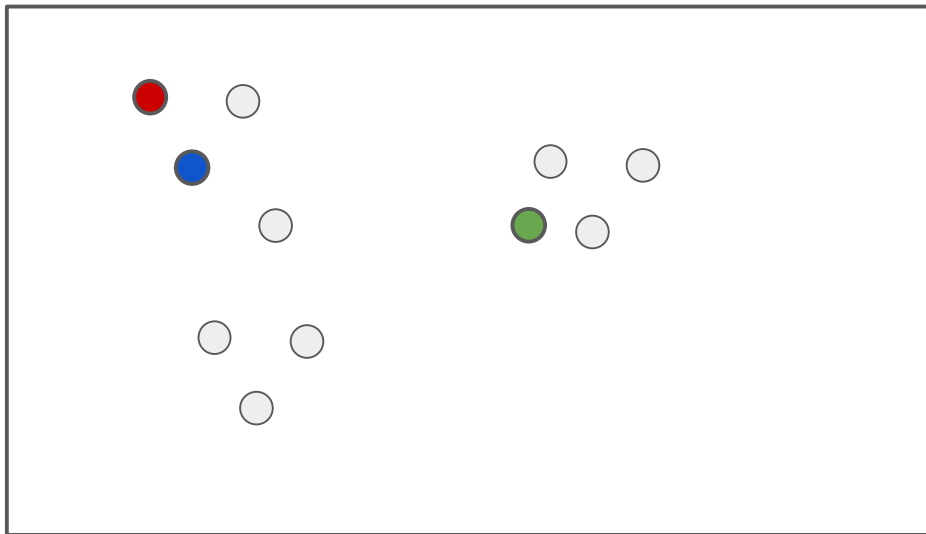
x1



# Clustering

- Step 2: We'll treat these new K points as our “cluster” points.

x2



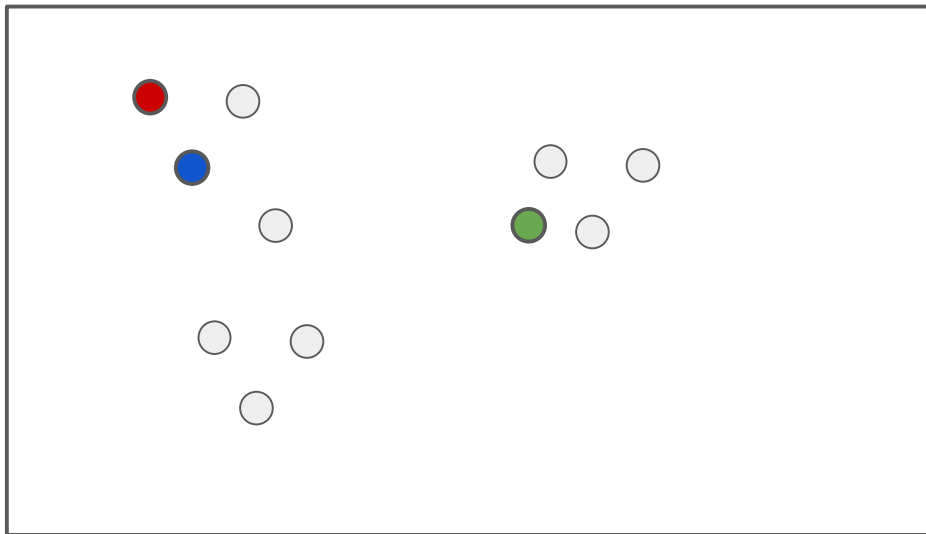
x1



# Clustering

- Step 3: Assign each remaining point to the nearest “cluster” point.

x2



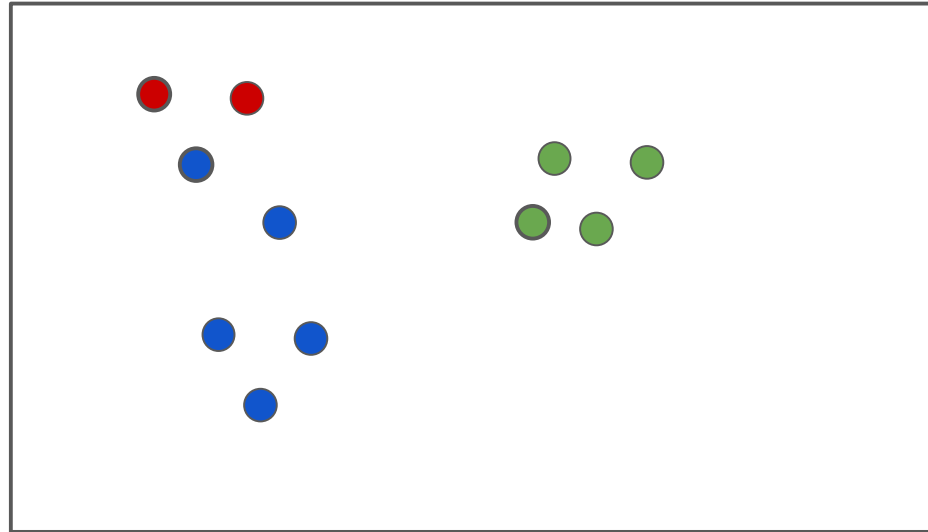
x1



# Clustering

- Step 3: Assign each remaining point to the nearest “cluster” point.

x2



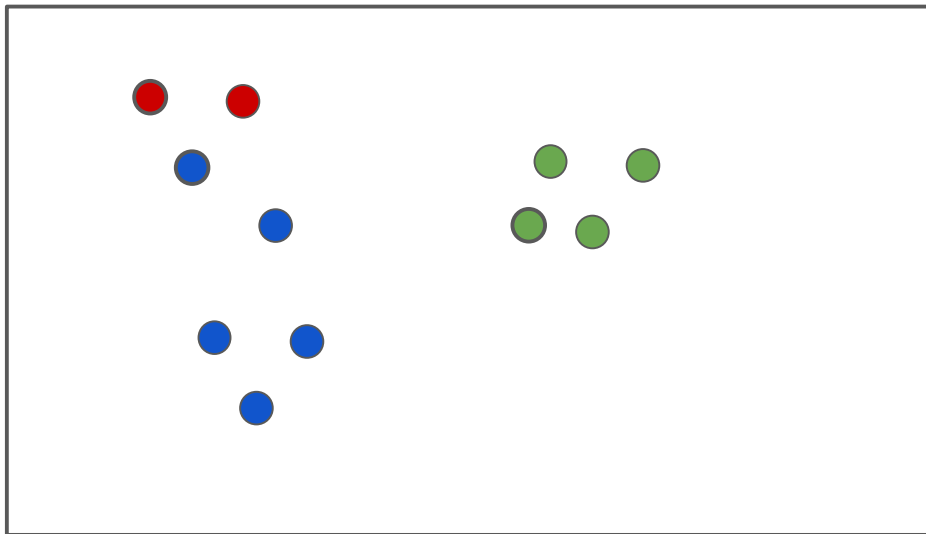
x1



# Clustering

- Step 3: Note how this is using a distance metric to judge the nearest point.

x2



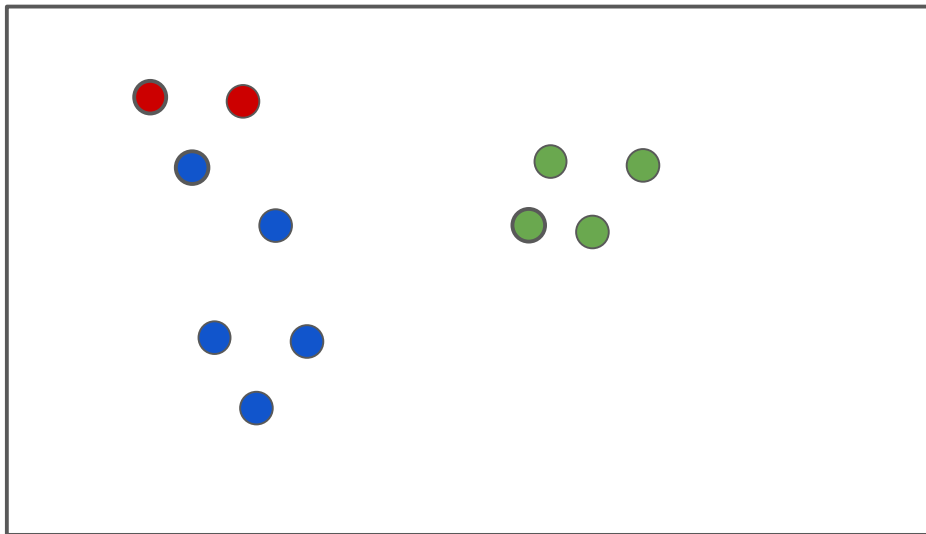
$X_1$



# Clustering

- Step 4: Calculate the center of the cluster points (mean value of each point vector).

x2



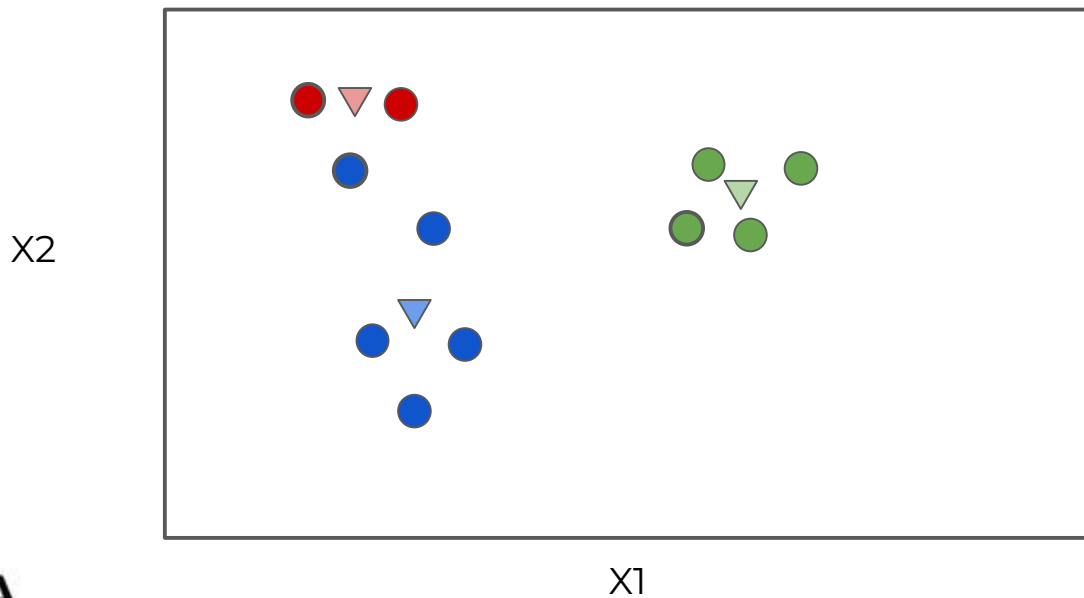
x1





# Clustering

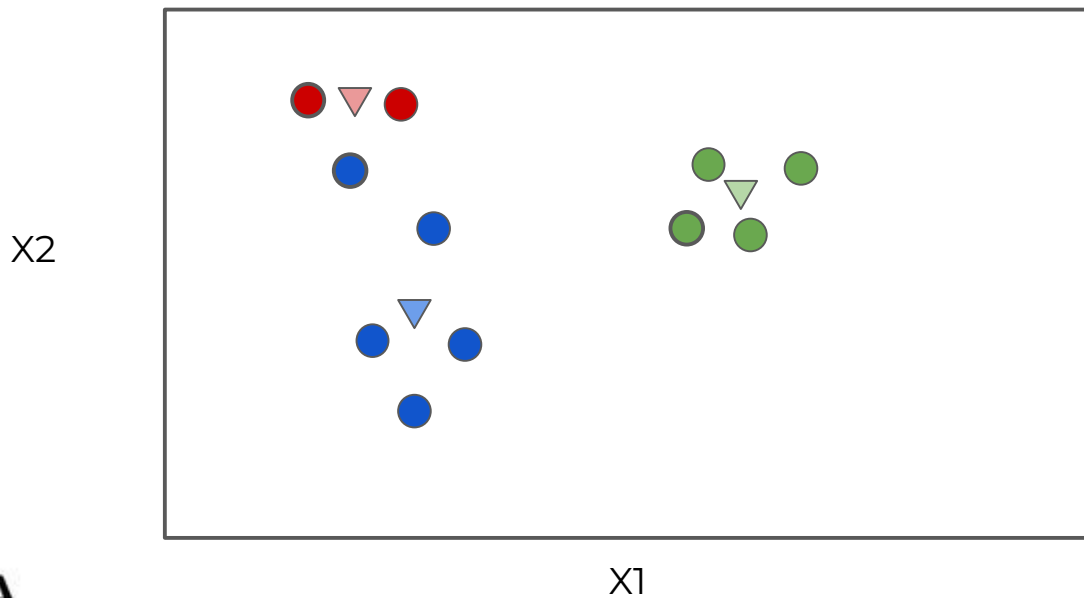
- Step 4: Calculate the center of the cluster points (mean value of point vectors).





# Clustering

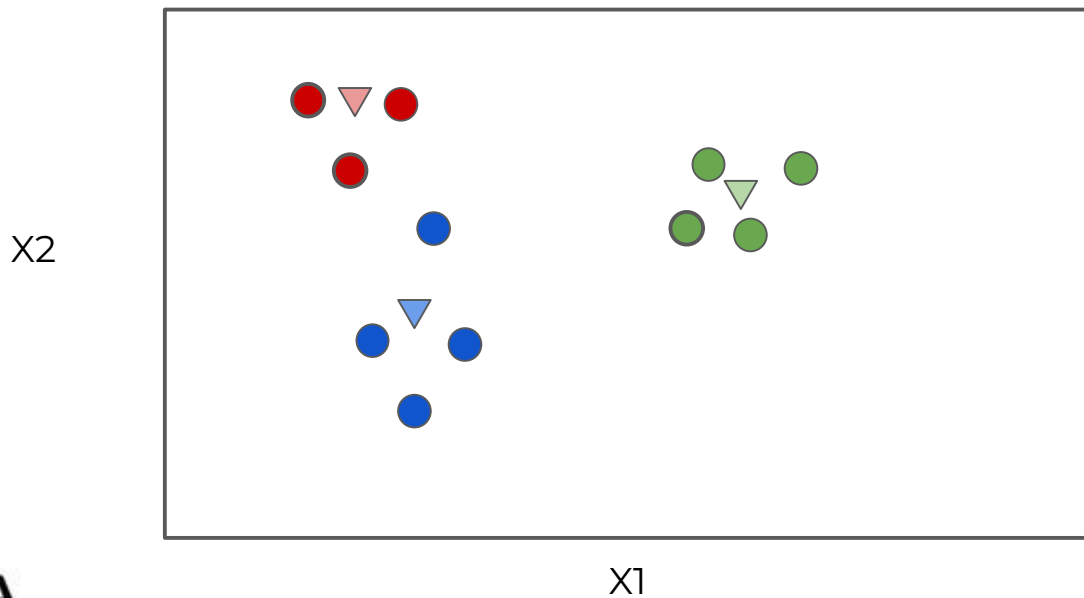
- Step 5: Now assign each point to the nearest cluster center.





# Clustering

- Step 5: Now assign each point to the nearest cluster center.

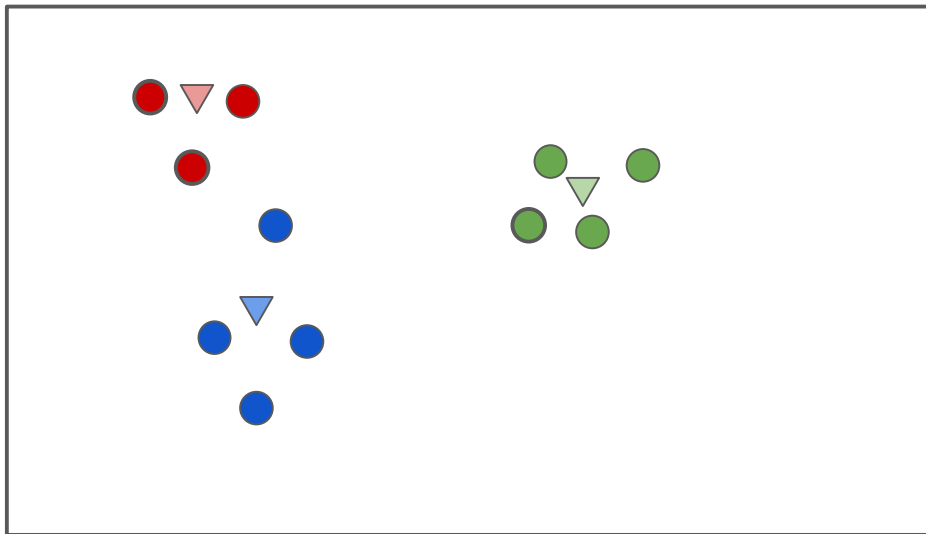




# Clustering

- We repeat steps 4 and 5 until there are no more cluster reassignments.

x2

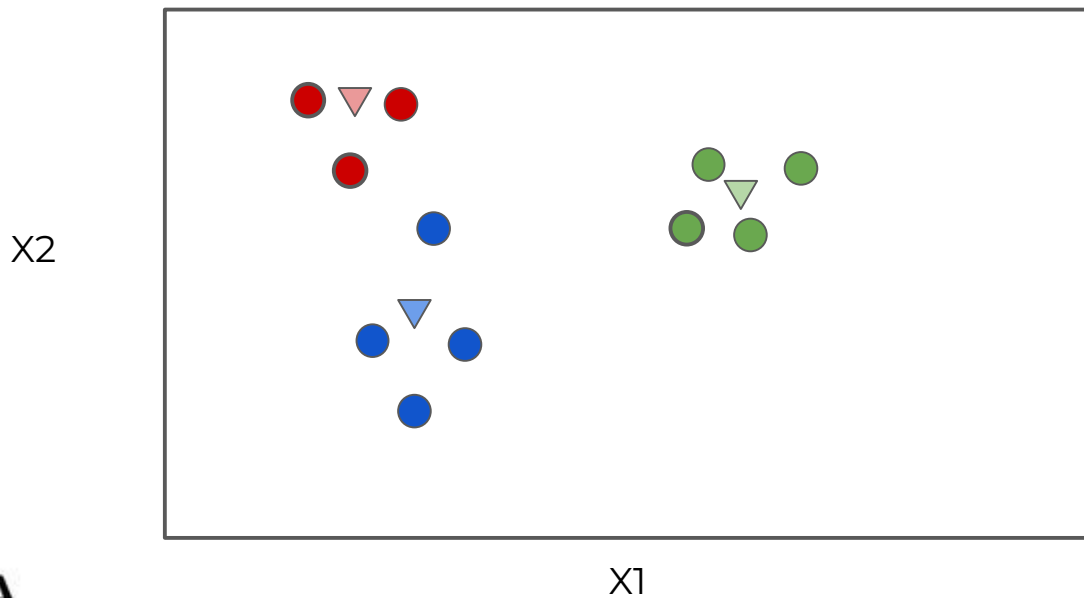


x1



# Clustering

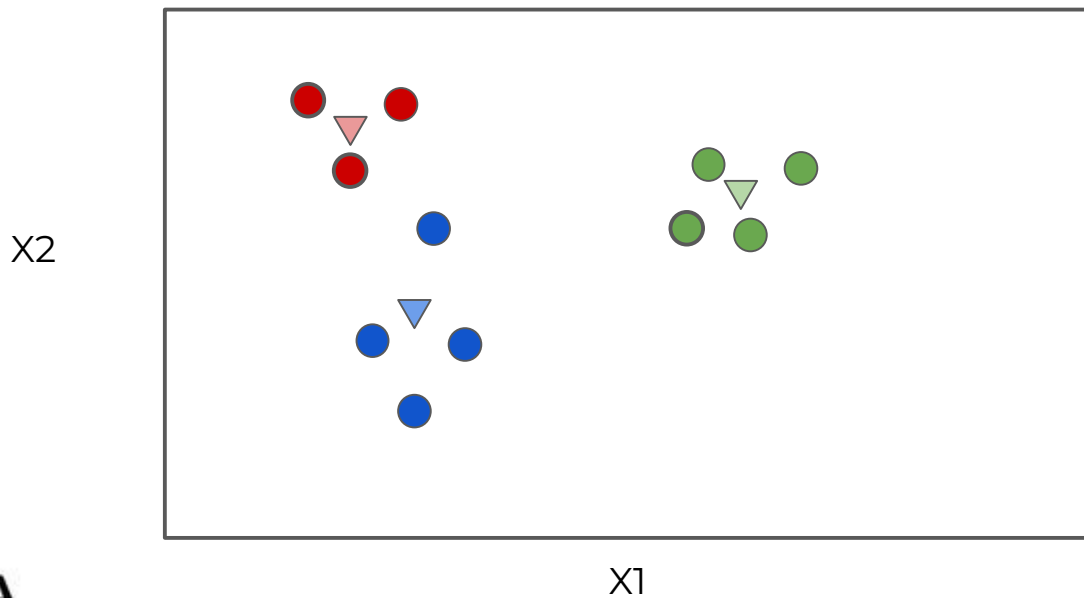
- Step 4b: Recalculate new cluster centers:





# Clustering

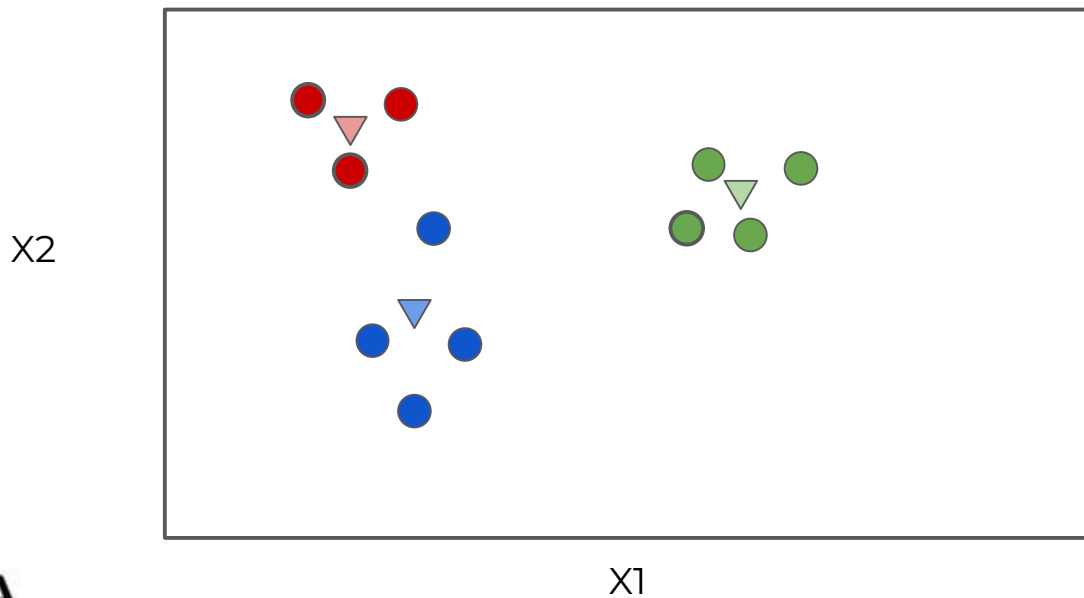
- Step 4b: Recalculate new cluster centers:





# Clustering

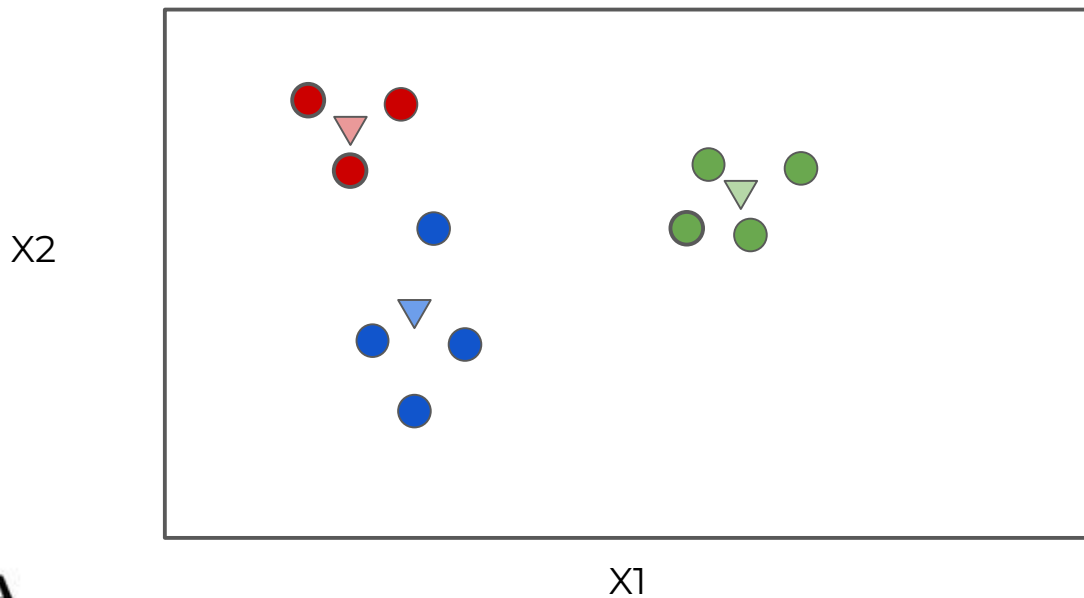
- Step 5b: Assign points to nearest cluster center.





# Clustering

- If there are no more reassignments, we're done! The clusters have been found.







# Clustering

- **Upcoming considerations:**

- How do we choose a reasonable value for  $K$  number of clusters?
- Is there any way we can evaluate how good our current  $K$  value is at determining clusters?



# Clustering

- Let's continue by coding out an example of K-Means clustering, then we'll revisit these considerations when they naturally appear after we find the first set of clusters for a given  $K$ .



# K-Means Clustering Example

Exploratory Data Analysis



# K-Means Clustering Example

Data Preparation and Model Fitting



# K-Means Clustering Example

Choosing a K Value



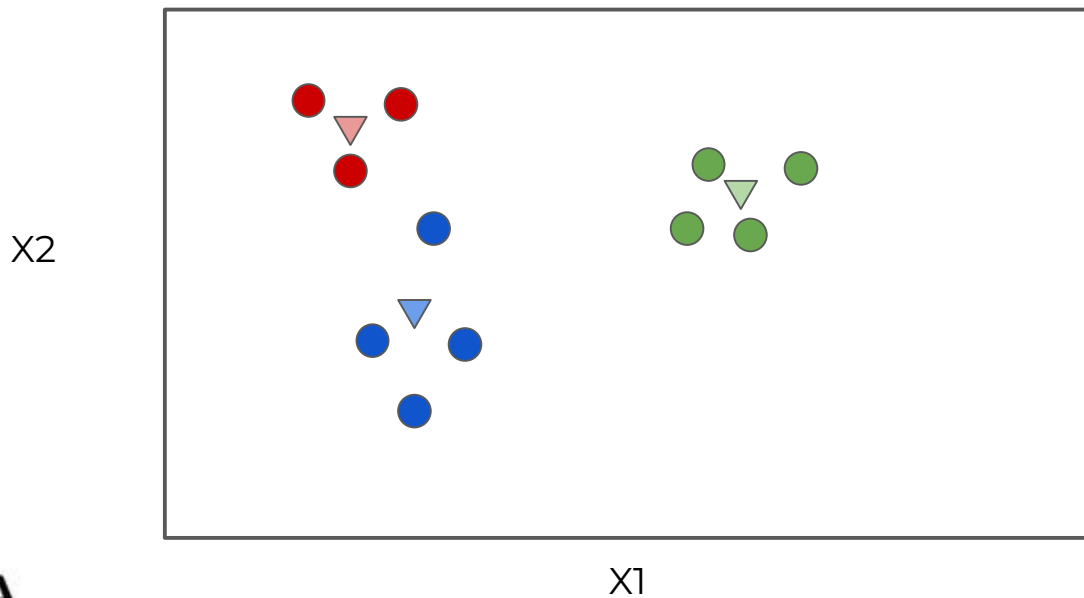
# Clustering

- **Recall our previous considerations:**
  - How do we choose a reasonable value for  $K$  number of clusters?
  - Is there any way we can evaluate how good our current  $K$  value is at determining clusters?



# Clustering

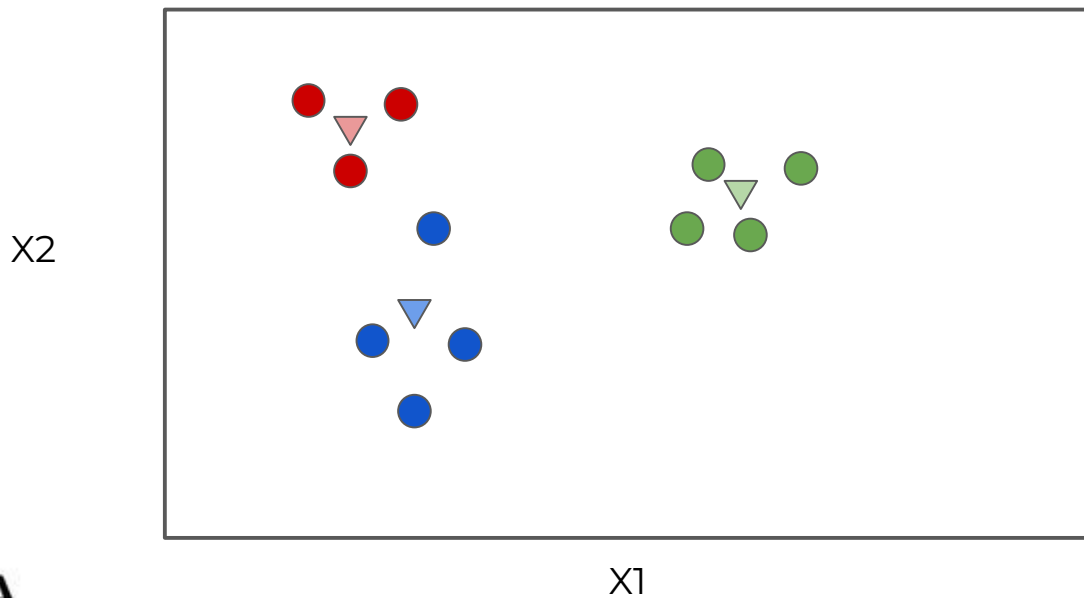
- Here we have 3 clusters, how can we measure “goodness of fit”?





# Clustering

- We could measure the sum of the distances from points to cluster centers.

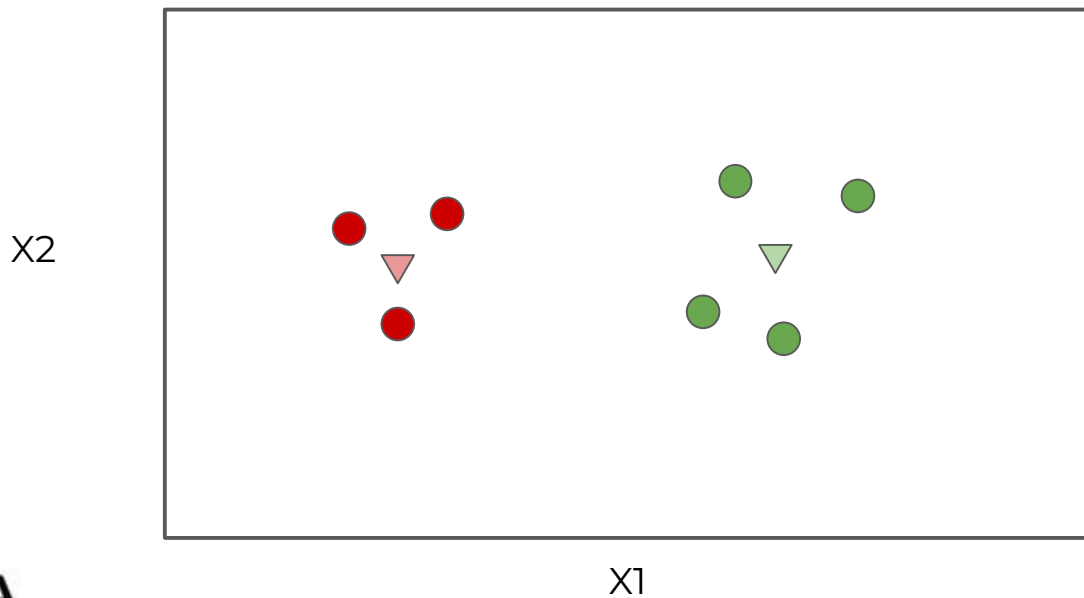






# Clustering

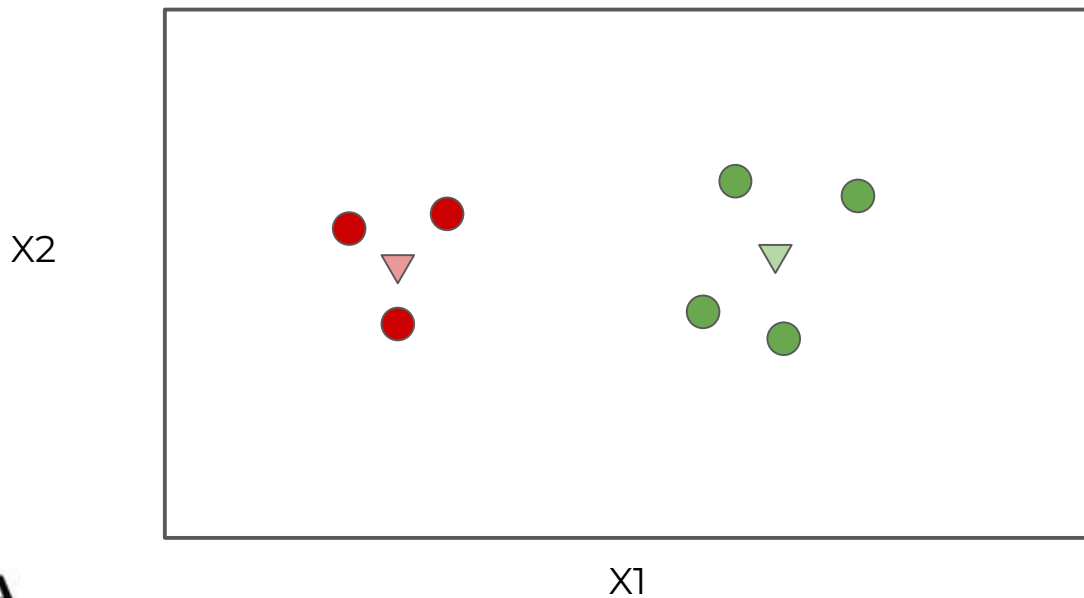
- Imagine a simple example starting with  $K=2$ .





# Clustering

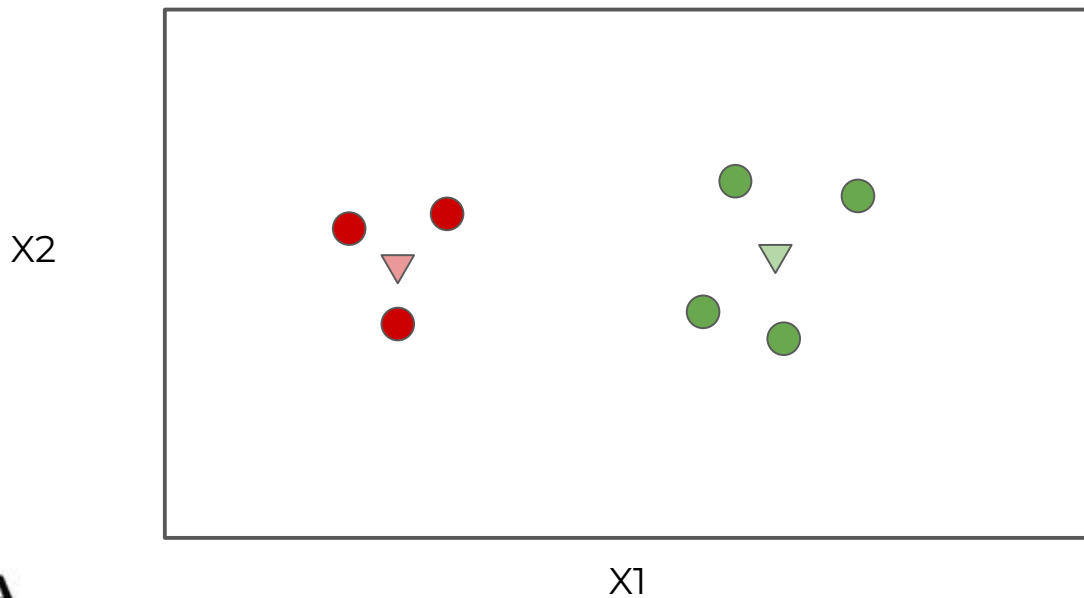
- We measure the sum of the squared distances from points to the cluster center:





# Clustering

- Then we fit an entirely new KMeans model with  $K+1$ :

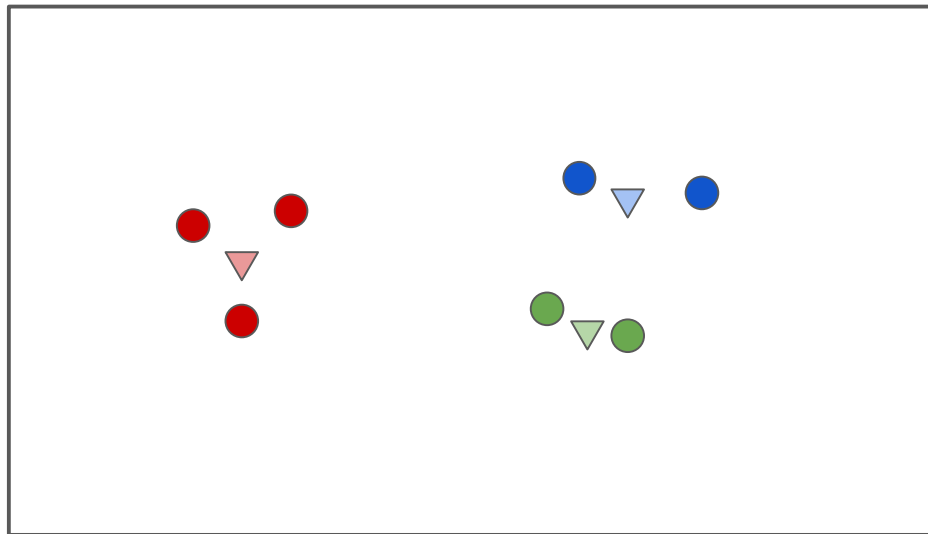




# Clustering

- Then we fit an entirely new KMeans model with  $K+1$ :

x2



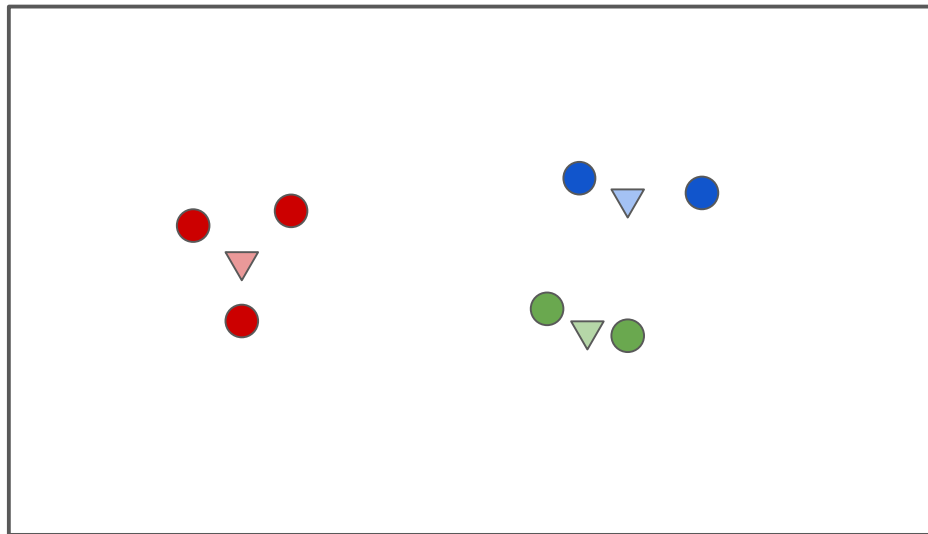
x1



# Clustering

- Then measure again the sum of the squared distance (SSD) to center.

x2



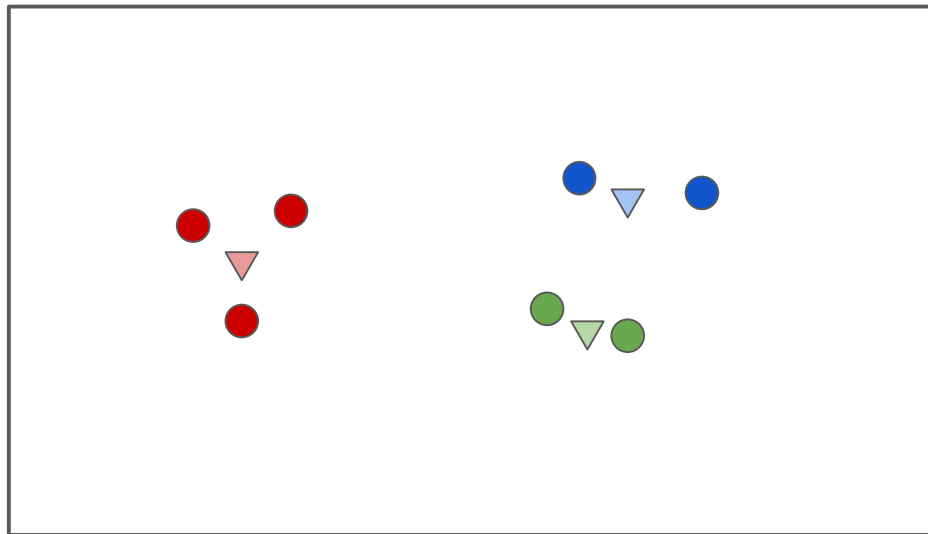
x1



# Clustering

- In theory this SSD would go to zero once  $K$  is equal to the number of points.

x2



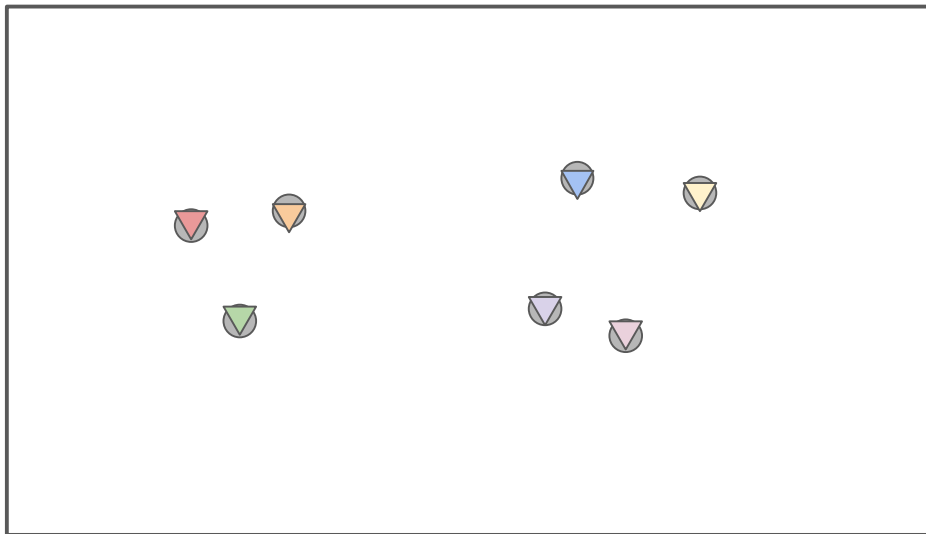
x1



# Clustering

- You would have a cluster for each point!  
SSD would be perfect at 0!

X2



X1



# Clustering

- We keep track of this SSD value for a range of different K values.
- We then look for a K value where **rate of reduction in SSD** begins to decline.
- This signifies that adding an extra cluster is **not** obtaining enough clarity of cluster separation to justify increasing K.





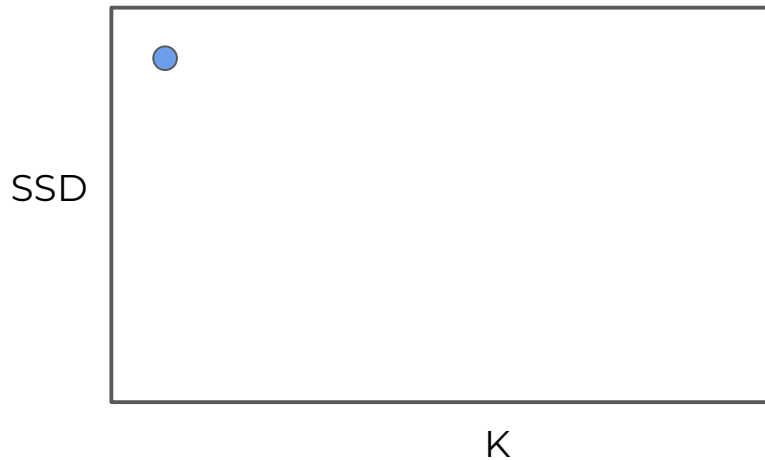
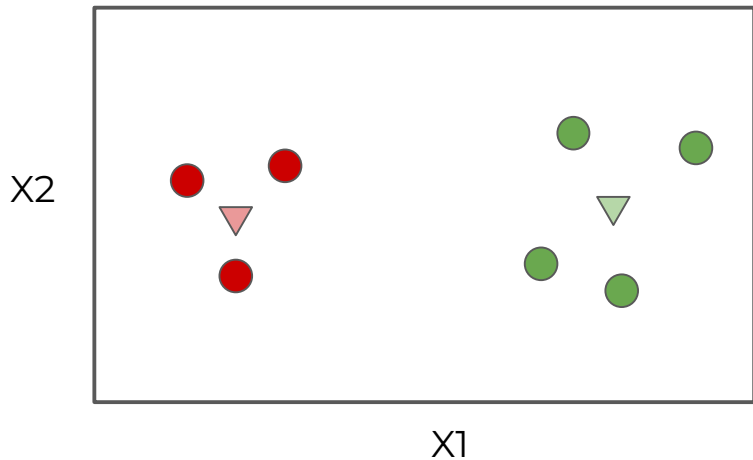
# Clustering

- This is known as the “elbow” method since we will track where decrease in SSD begins to flatten out compared to increasing K values.
- Let’s walk through what this chart would look like...



# Clustering

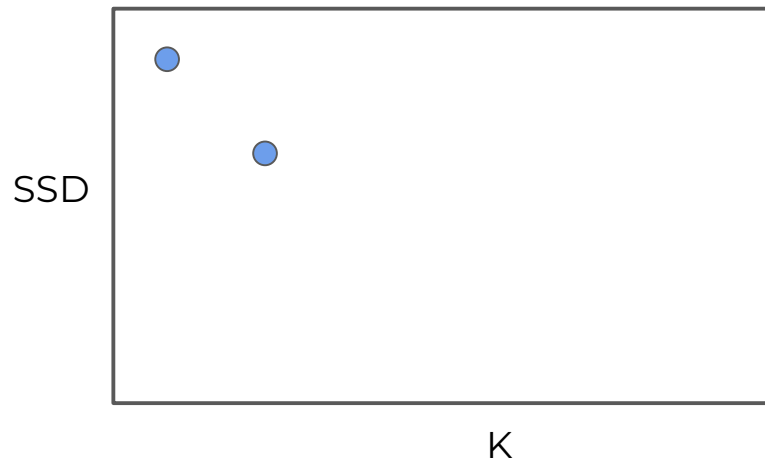
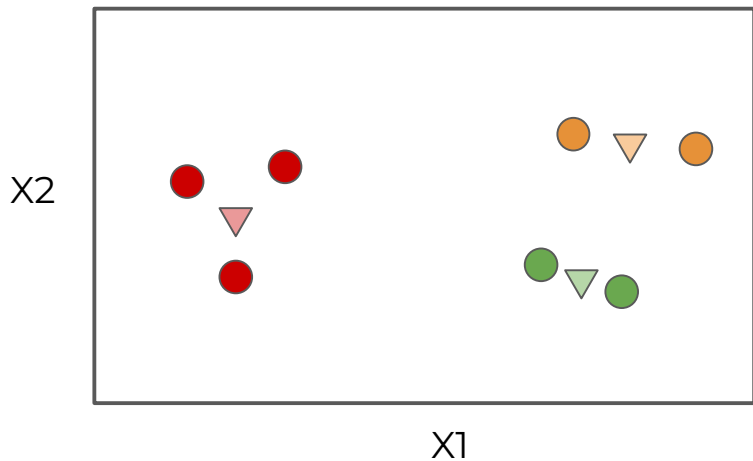
- Start with  $K=2$ :





# Clustering

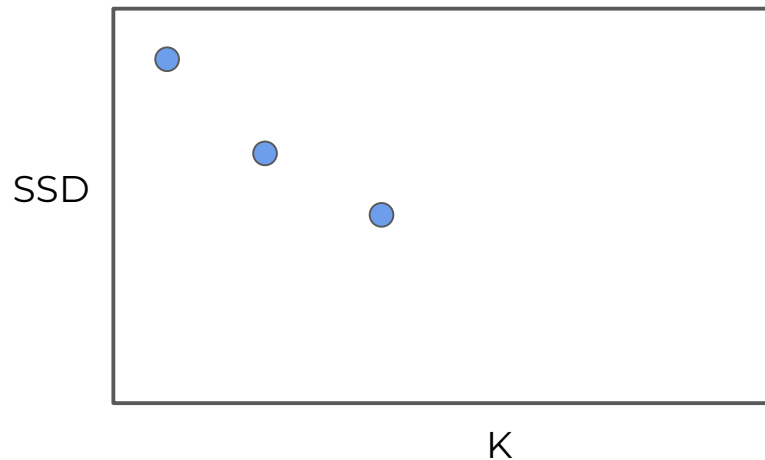
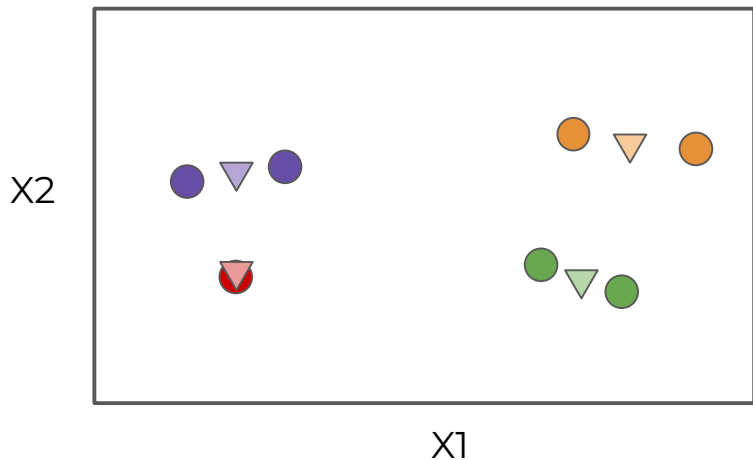
- Increase K and measure SSD:





# Clustering

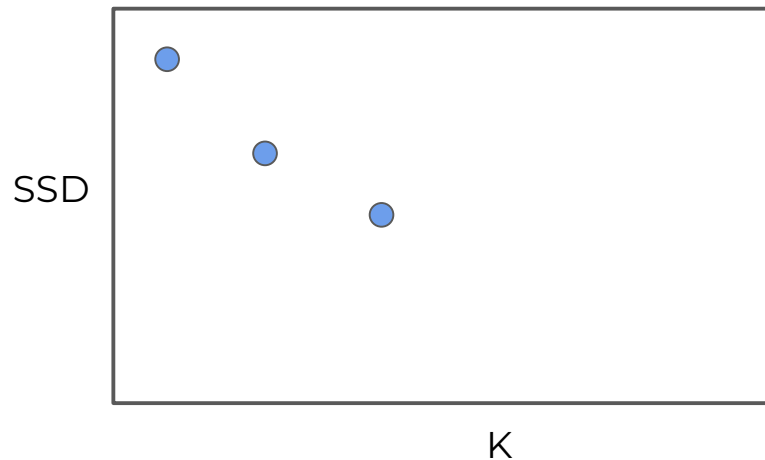
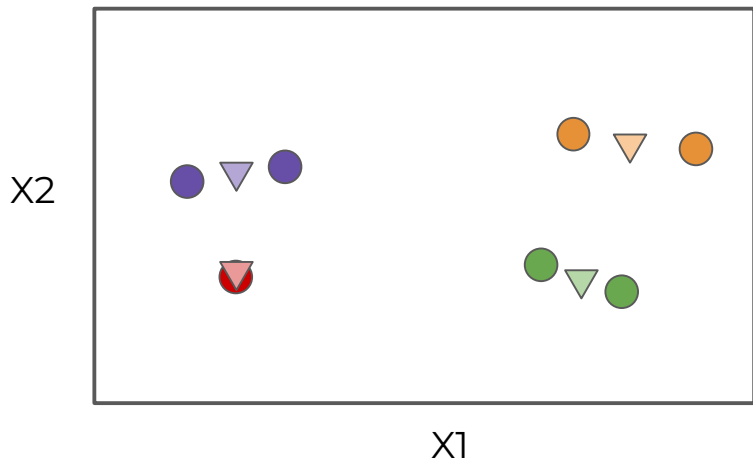
- Increase K and measure SSD:





# Clustering

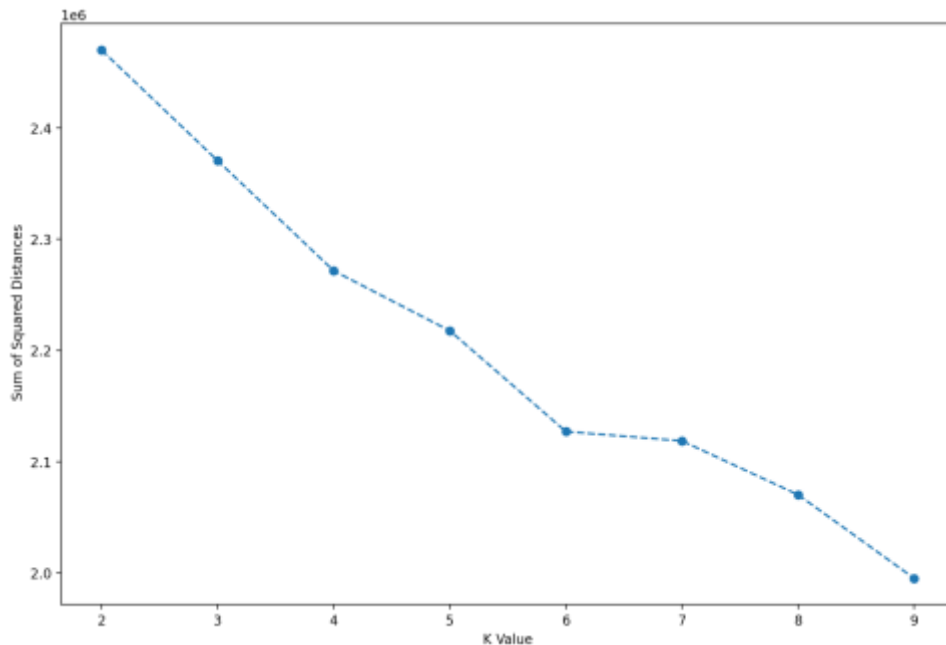
- Repeat this process for some set number of K values:





# K Means Clustering

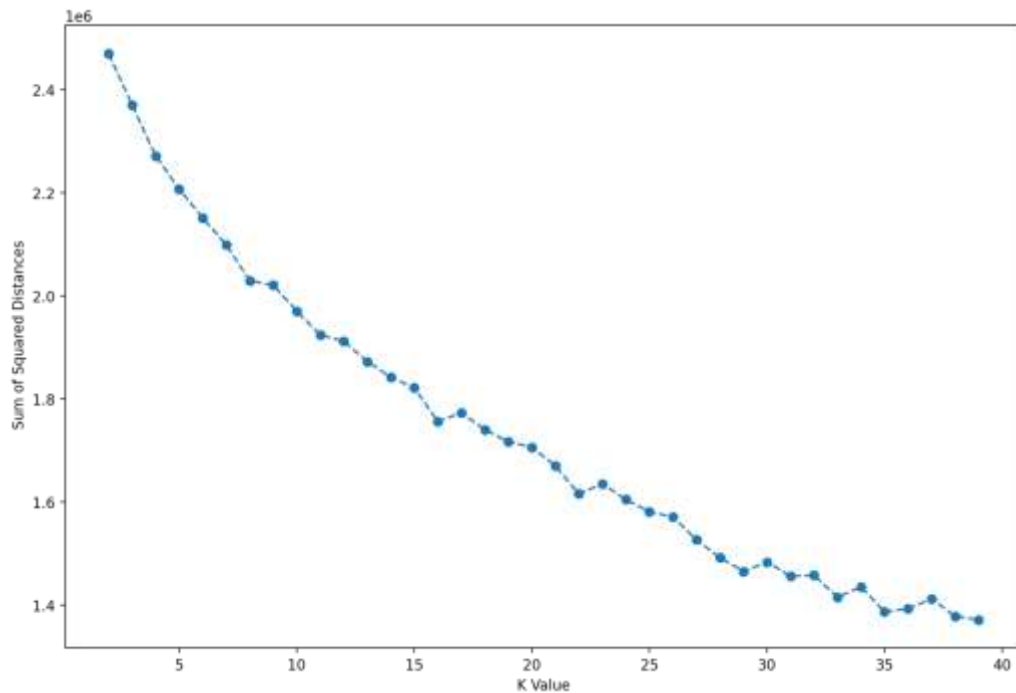
- You will see a continuous decline.





# K Means Clustering

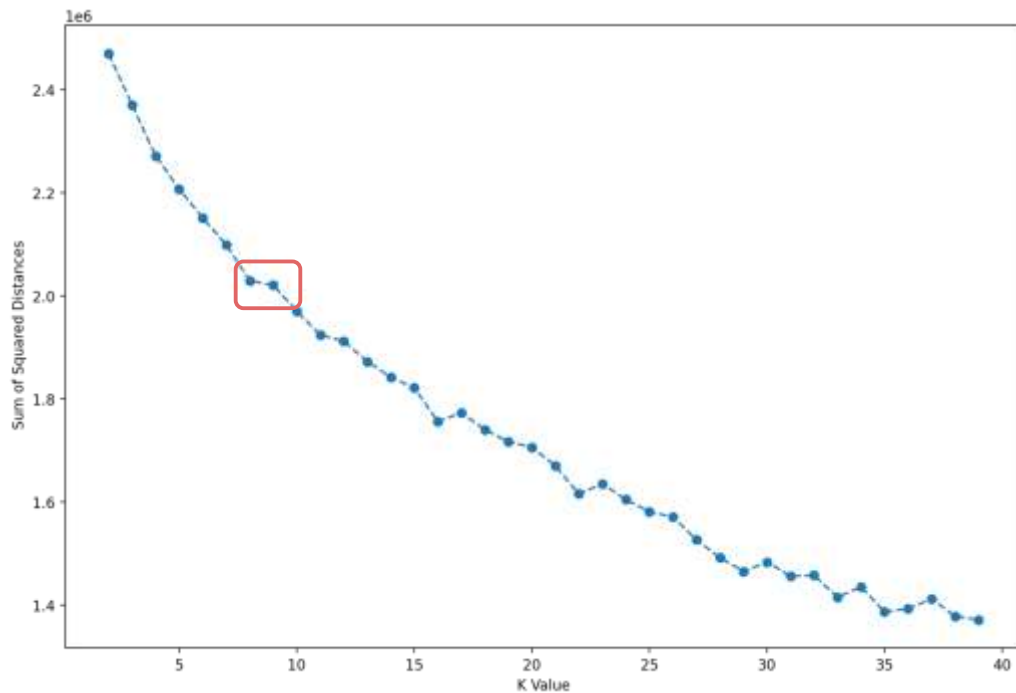
- Eventually you will see “elbow” points:





# K Means Clustering

- Eventually you will see “elbow” points:

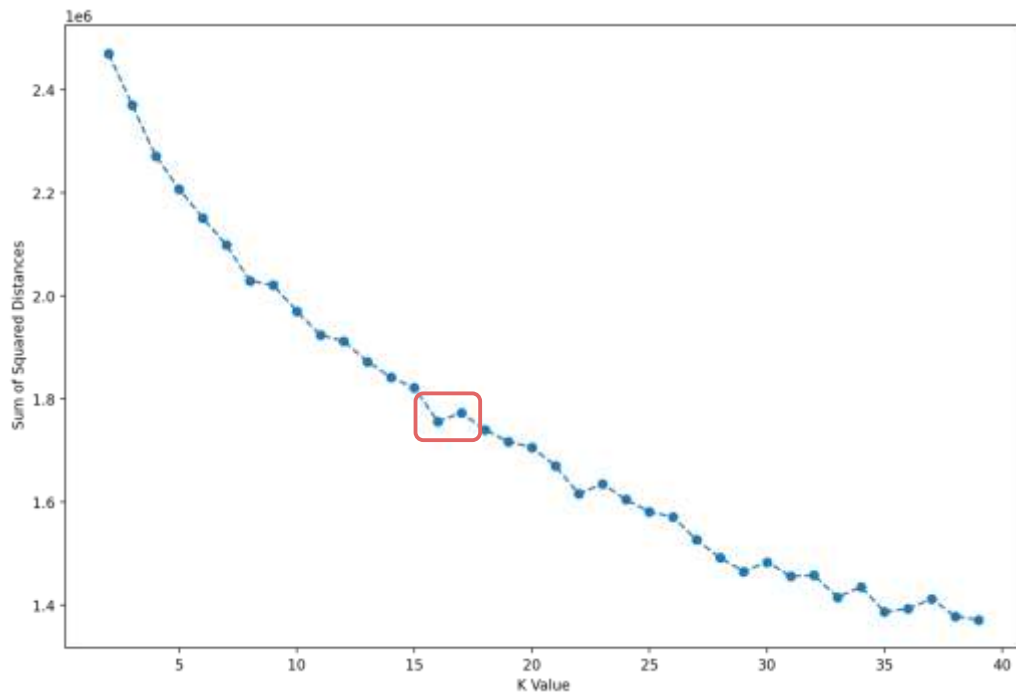






# K Means Clustering

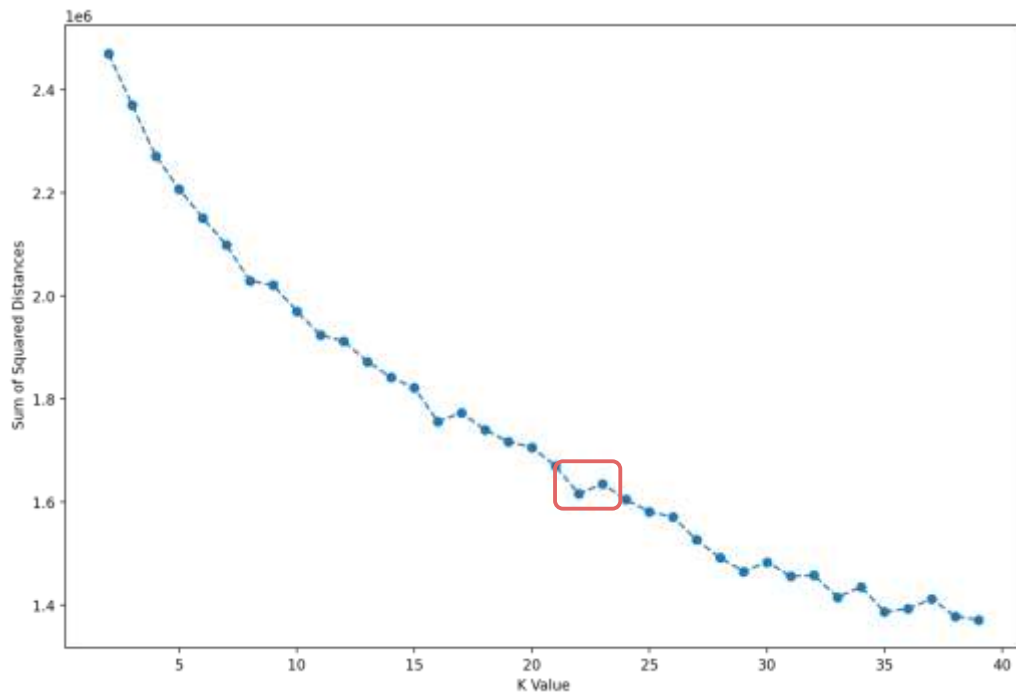
- Eventually you will see “elbow” points:





# K Means Clustering

- Eventually you will see “elbow” points:





# Clustering

- These points are strong indicators that increasing  $K$  further is no longer justified as it is not revealing more “signal”.
- You can also measure out this SSD in a barplot.
- Let's explore this further with code!



# K-Means Clustering Color Quantization

Part One: General Concepts



# K Means Clustering

- Unsupervised Learning provides opportunities for very creative use cases on algorithm applications.
- Searching for insights, patterns, and general understanding of our data allows us to apply methods to a variety of tasks.



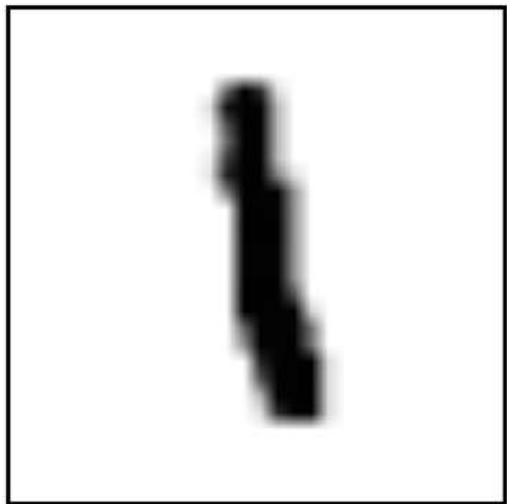
# K Means Clustering

- One interesting application of clustering is on image quantization.
- Let's discuss images, computers, colors, and quantization to get an idea of how K Means clustering can be applied to different fields.



# K Means Clustering

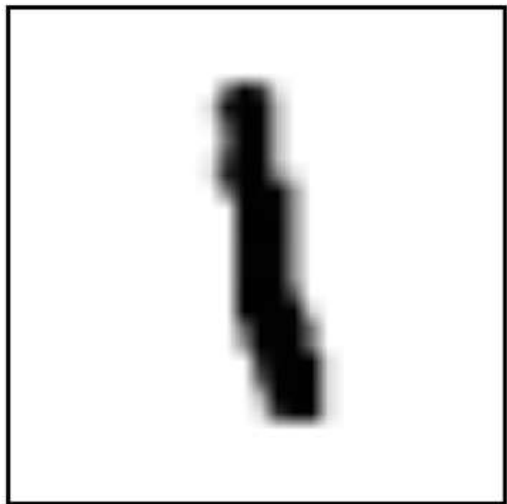
- Imagine an image of a single pen stroke:





## K Means Clustering

- This image is in **grayscale**, meaning the color range goes from black to white.

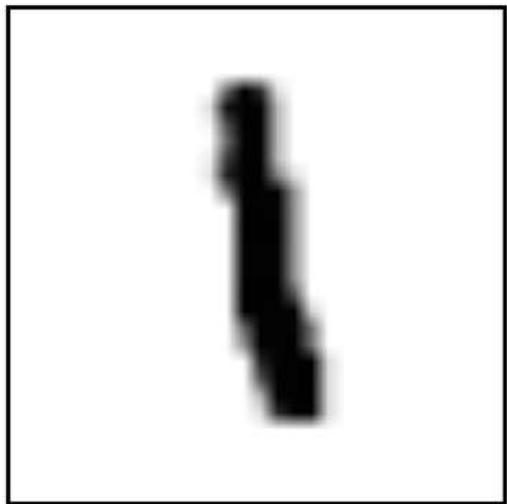






## K Means Clustering

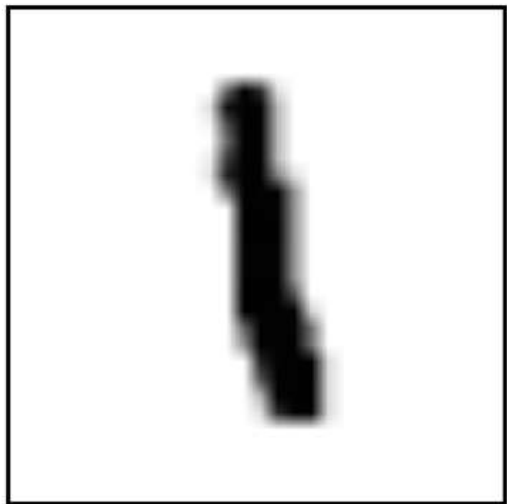
- You will notice on the edges there are gray colors between black and white.





# K Means Clustering

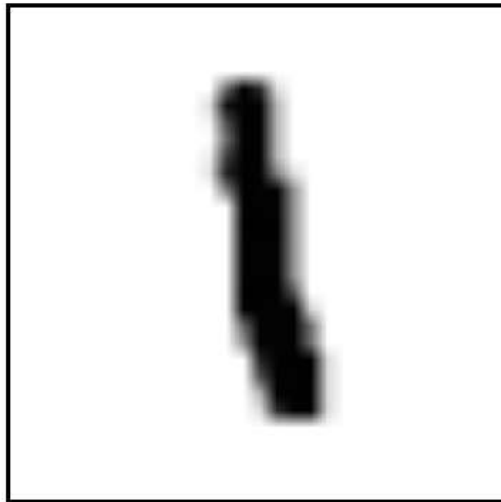
- A computer will store this information as an array with values between a range.





# K Means Clustering

- A computer will store this information as an array with values between a range.



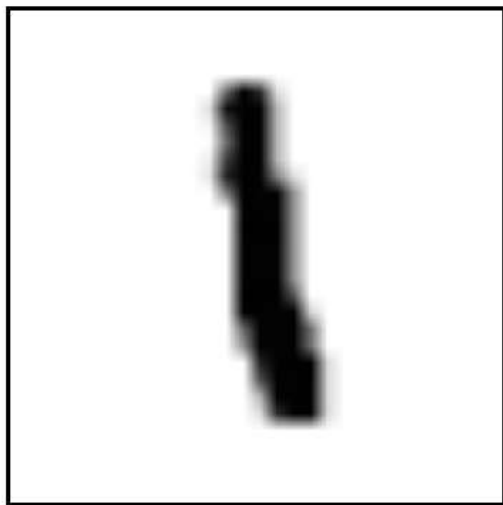
12

0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	.6	.8	0	0	0	0	0
0	0	0	0	0	0	0	.7	1	0	0	0	0	0
0	0	0	0	0	0	0	.7	1	0	0	0	0	0
0	0	0	0	0	0	0	.5	1	.4	0	0	0	0
0	0	0	0	0	0	0	0	1	.4	0	0	0	0
0	0	0	0	0	0	0	0	1	.4	0	0	0	0
0	0	0	0	0	0	0	0	1	.7	0	0	0	0
0	0	0	0	0	0	0	0	1	1	0	0	0	0
0	0	0	0	0	0	0	0	.9	1	.1	0	0	0
0	0	0	0	0	0	0	0	.3	1	.1	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0



# K Means Clustering

- Notice 0 is white and 1 is black, with values in between representing gray.



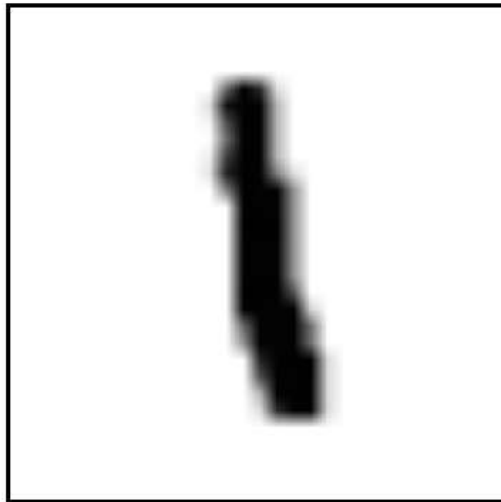
12

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	.6	.8	0	0	0	0	0	0
0	0	0	0	0	0	0	.7	1	0	0	0	0	0	0
0	0	0	0	0	0	0	.7	1	0	0	0	0	0	0
0	0	0	0	0	0	0	.5	1	.4	0	0	0	0	0
0	0	0	0	0	0	0	0	1	.4	0	0	0	0	0
0	0	0	0	0	0	0	0	1	.4	0	0	0	0	0
0	0	0	0	0	0	0	0	1	.7	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	.9	1	.1	0	0	0	0
0	0	0	0	0	0	0	0	.3	1	.1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



# K Means Clustering

- It is also very common for computers to store values from 0-255 for scales.



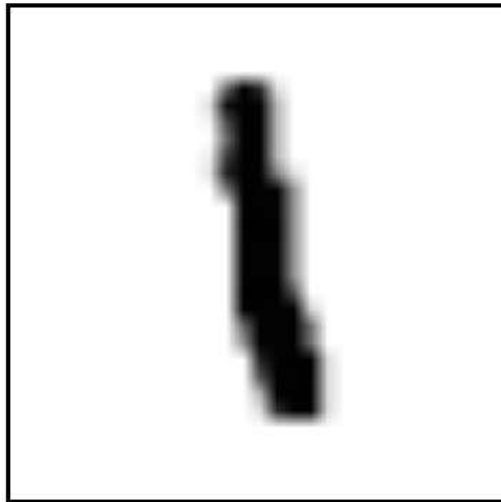
12

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	.6	.8	0	0	0	0	0	0	0
0	0	0	0	0	0	0	.7	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	.7	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	.5	1	.4	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	.4	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	.4	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	.7	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	.9	1	.1	0	0	0	0	0
0	0	0	0	0	0	0	0	.3	1	.1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



# K Means Clustering

- The range 0 to 255 has to do with how computers store 8-bit numbers.



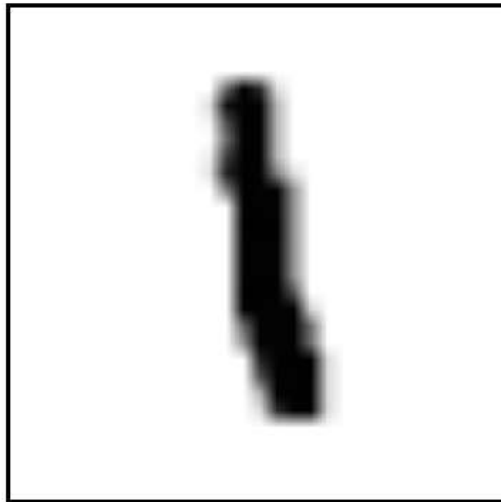
12

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	.6	.8	0	0	0	0	0	0	0
0	0	0	0	0	0	0	.7	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	.7	1	0	0	0	0	0	0	0
0	0	0	0	0	0	0	.5	1	.4	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	.4	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	.4	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	.7	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0
0	0	0	0	0	0	0	0	.9	1	.1	0	0	0	0	0
0	0	0	0	0	0	0	0	.3	1	.1	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



# K Means Clustering

- But you can always divide all the values by 255 to normalize to between 0 and 1



12

0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	.6	.8	0	0	0	0	0	0
0	0	0	0	0	0	0	.7	1	0	0	0	0	0	0
0	0	0	0	0	0	0	.7	1	0	0	0	0	0	0
0	0	0	0	0	0	0	.5	1	.4	0	0	0	0	0
0	0	0	0	0	0	0	0	1	.4	0	0	0	0	0
0	0	0	0	0	0	0	0	1	.4	0	0	0	0	0
0	0	0	0	0	0	0	0	1	.7	0	0	0	0	0
0	0	0	0	0	0	0	0	1	1	0	0	0	0	0
0	0	0	0	0	0	0	0	.9	1	.1	0	0	0	0
0	0	0	0	0	0	0	0	.3	1	.1	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0



## K Means Clustering

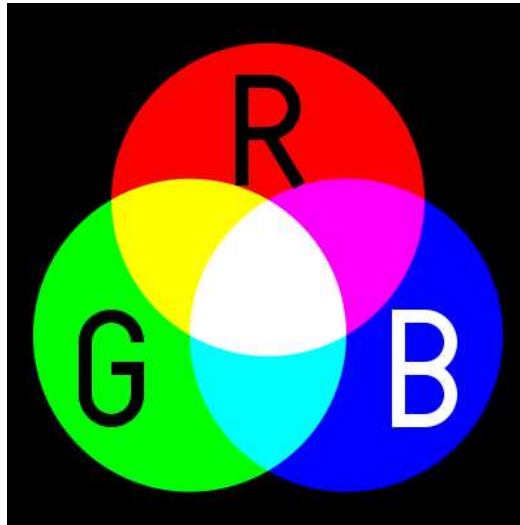
- Now that we've understood how grayscale images can be represented as arrays, what about color images?
- Color images can be represented as a combination of Red, Green, and Blue.





# K Means Clustering

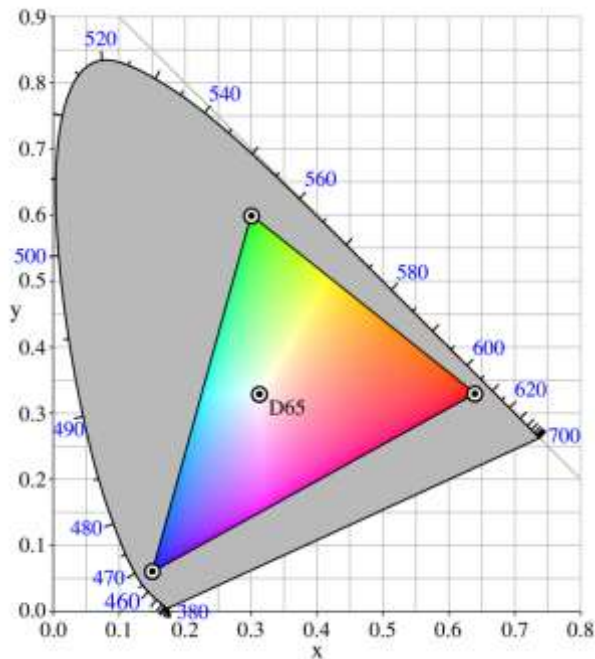
- Additive color mixing allows a wide variety of colors by simply combining different amounts of Red, Green, and Blue.





# K Means Clustering

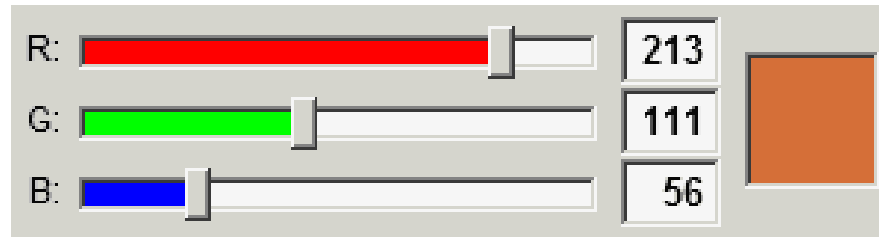
- RGB can produce a range of colors





# K Means Clustering

- Each color channel will have intensity values.
- You may have already seen this sort of representation in other software with RGB sliders.






# K Means Clustering

- Notice we now have 3 distinct values to track, with each value in a range (shown here from 0-255).
- Combining RGB to produce a distinct color.

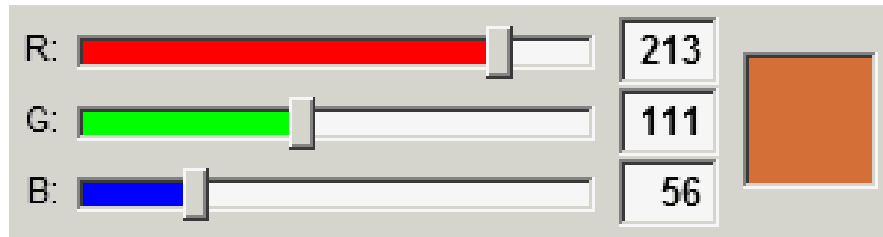
R:	<input type="range" value="213"/>	213
G:	<input type="range" value="111"/>	111
B:	<input type="range" value="56"/>	56





# K Means Clustering

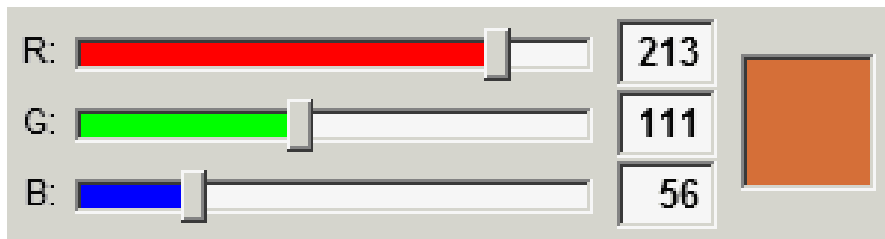
- From a computer perspective, this looks like 3 arrays, each array representing a color channel.
- For example, a single pixel (1 by 1 image) here is (213,111,56) for (R,G,B).





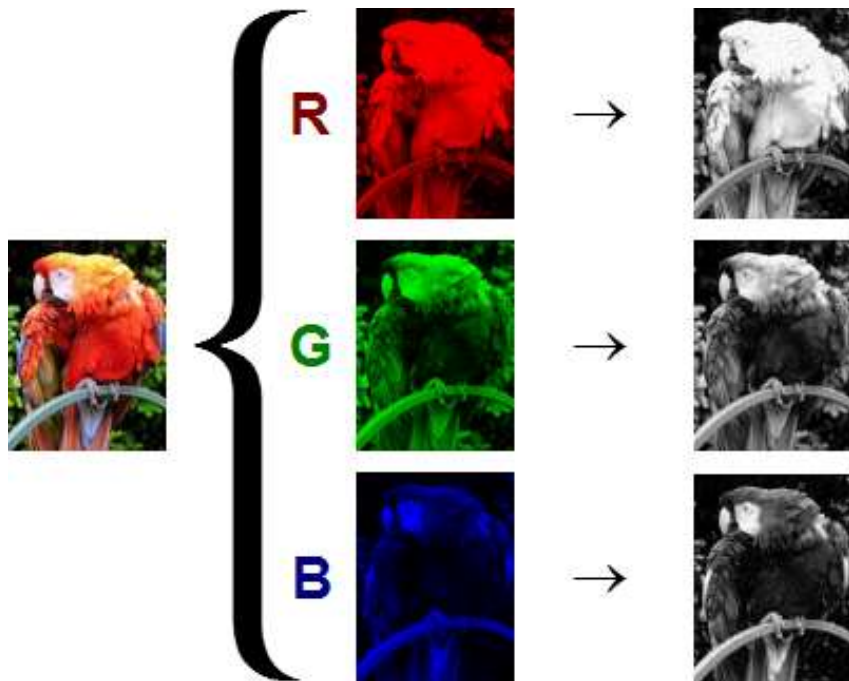
# K Means Clustering

- How is this stored for a larger color image?





# K Means Clustering



- The shape of the color array then has 3 dimensions.
- Height
- Width
- Color Channels



# K Means Clustering

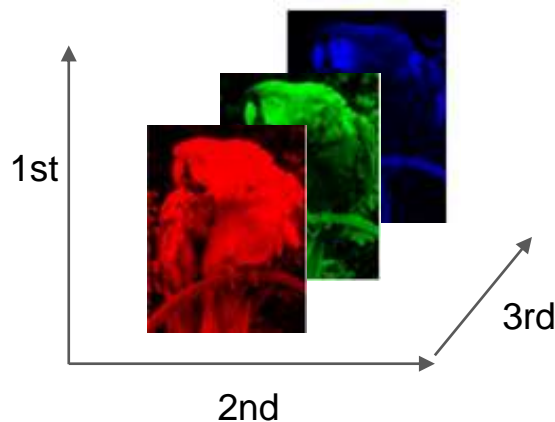
- This means when you read in an image and check its shape, it will look something like:
  - **(1280,720,3)**
  - **1280** pixel width
  - **720** pixel height
  - **3** color channels





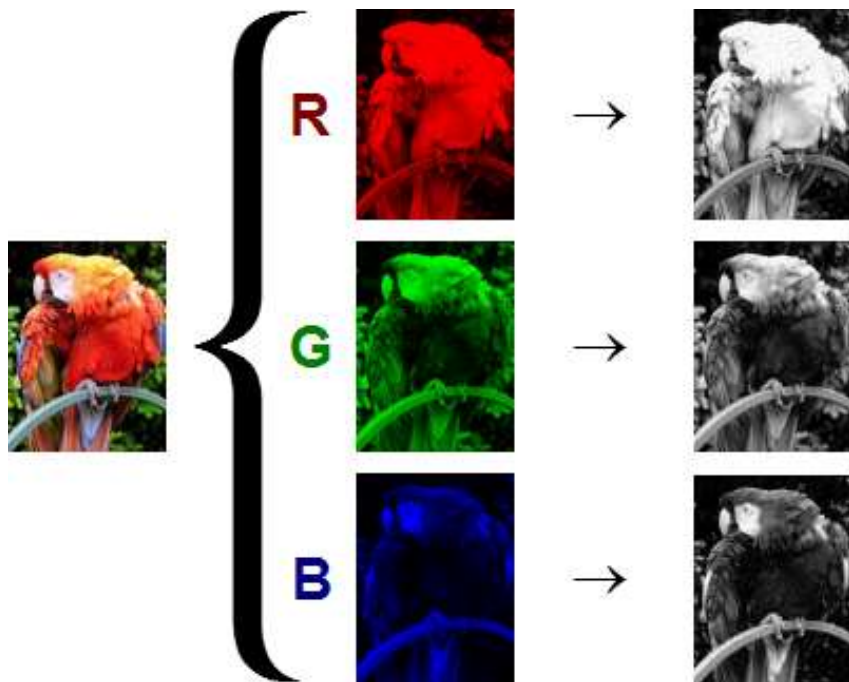
# K Means Clustering

- This means when you read in an image and check its shape, it will look something like:
  - **(720,1280,3)**
    - **720** pixel height
    - **1280** pixel width
    - **3** color channels





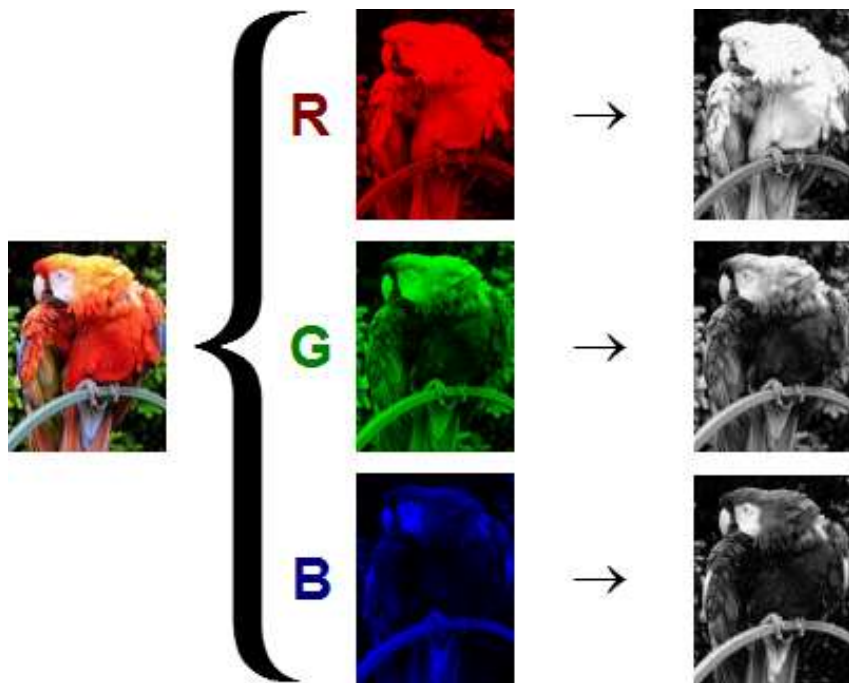
# K Means Clustering



- Keep in mind the computer won't "know" a channel is Red, it just knows that there are now 3 intensity channels.



# K Means Clustering



- The user needs to dictate which channel is for which color.
- Each channel alone is essentially the same as a grayscale image.



## K Means Clustering

- From the computer's perspective you simply have an array with 3 dimensions, where a user or display function can attribute each dimension to a color channel (e.g. red intensity).



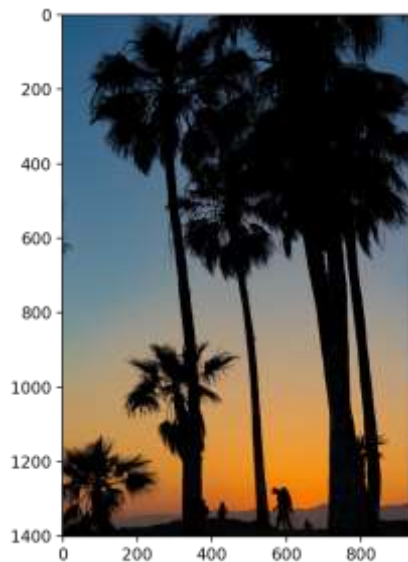
## K Means Clustering

- Swapping these arrays across channels would allow for effects such as color inversion.
- Now how can we apply clustering to RGB color channels and images?



# K Means Clustering

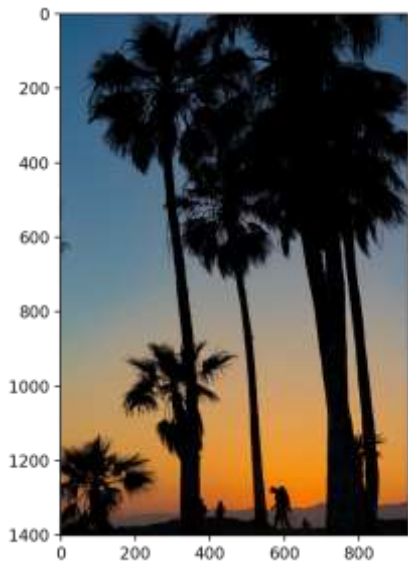
- Imagine the following image:





# K Means Clustering

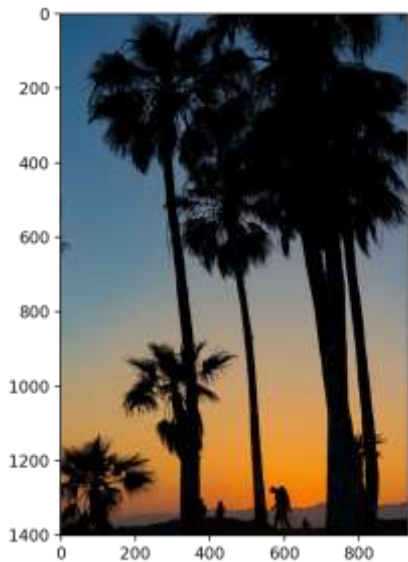
- There are many shades of colors in this image, with many (R,G,B) combinations.





# K Means Clustering

- What if we wanted to reduce this to 6 colors for simplified display purposes?

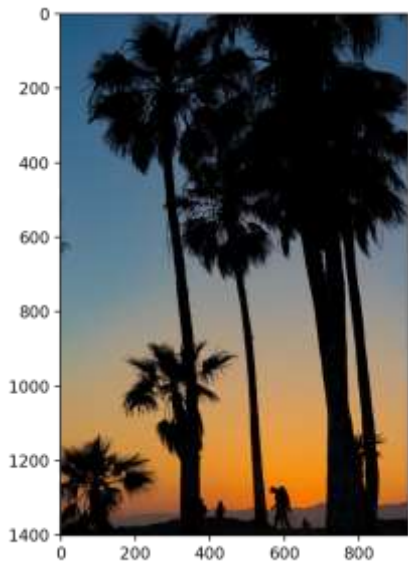






# K Means Clustering

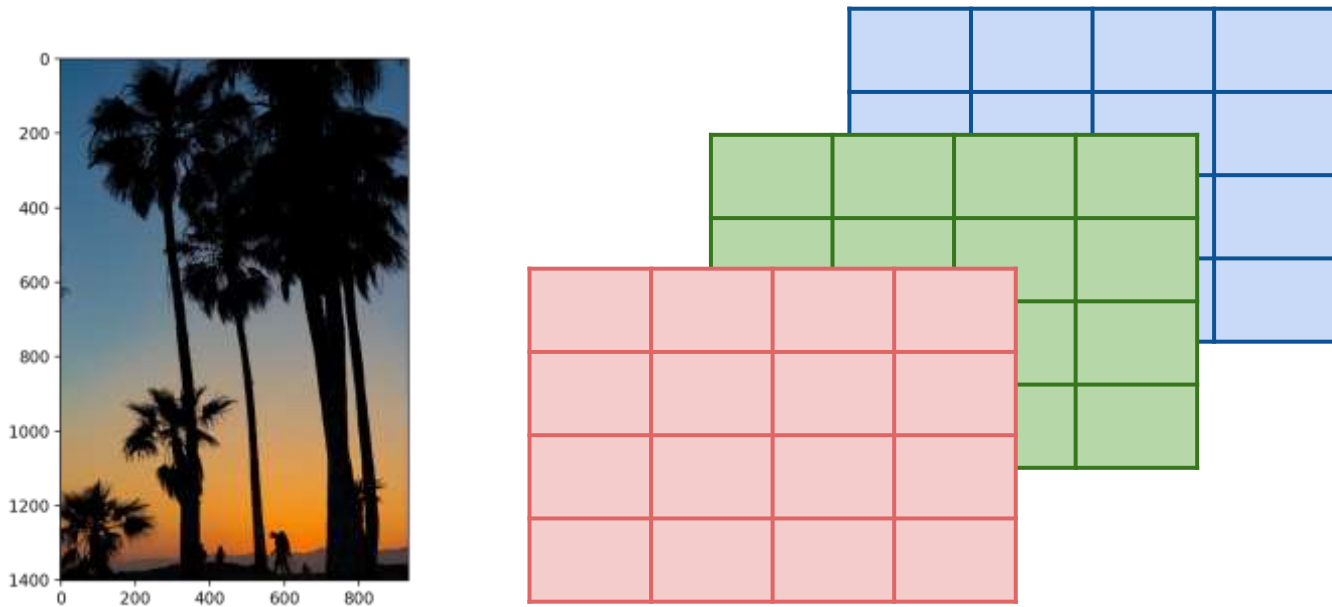
- What if we wanted to compress the image for a smaller screen with less colors?





# K Means Clustering

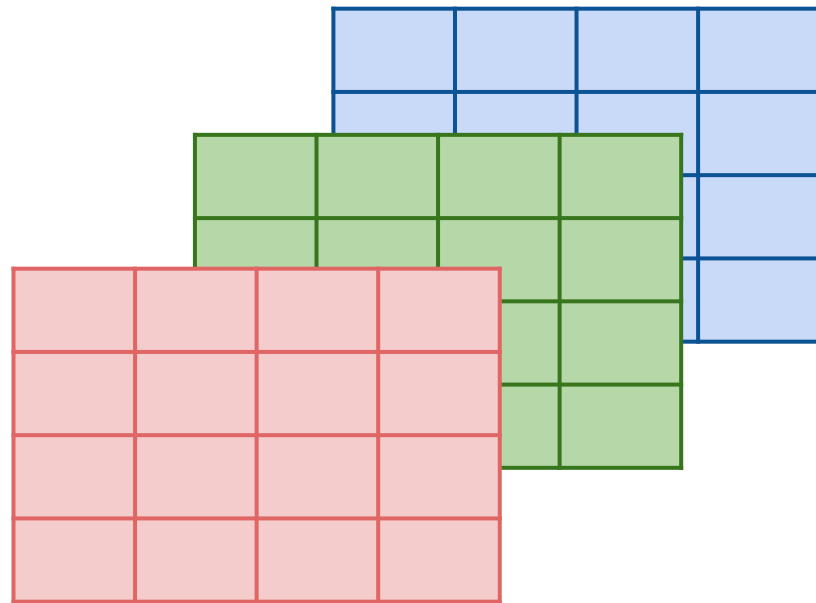
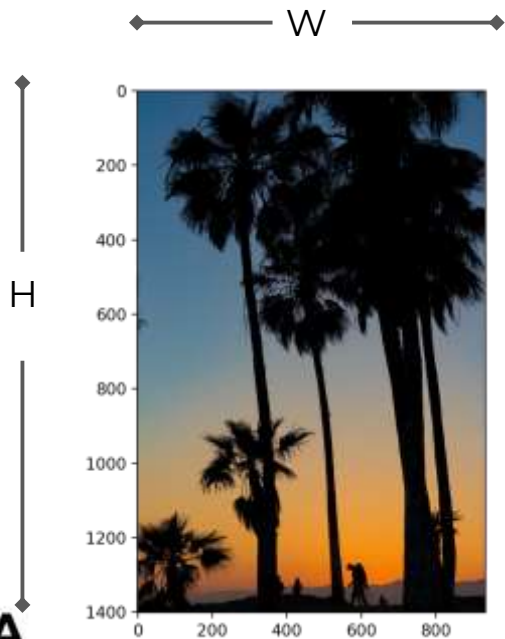
- Recall the image is a 3D array (H,W,C):





# K Means Clustering

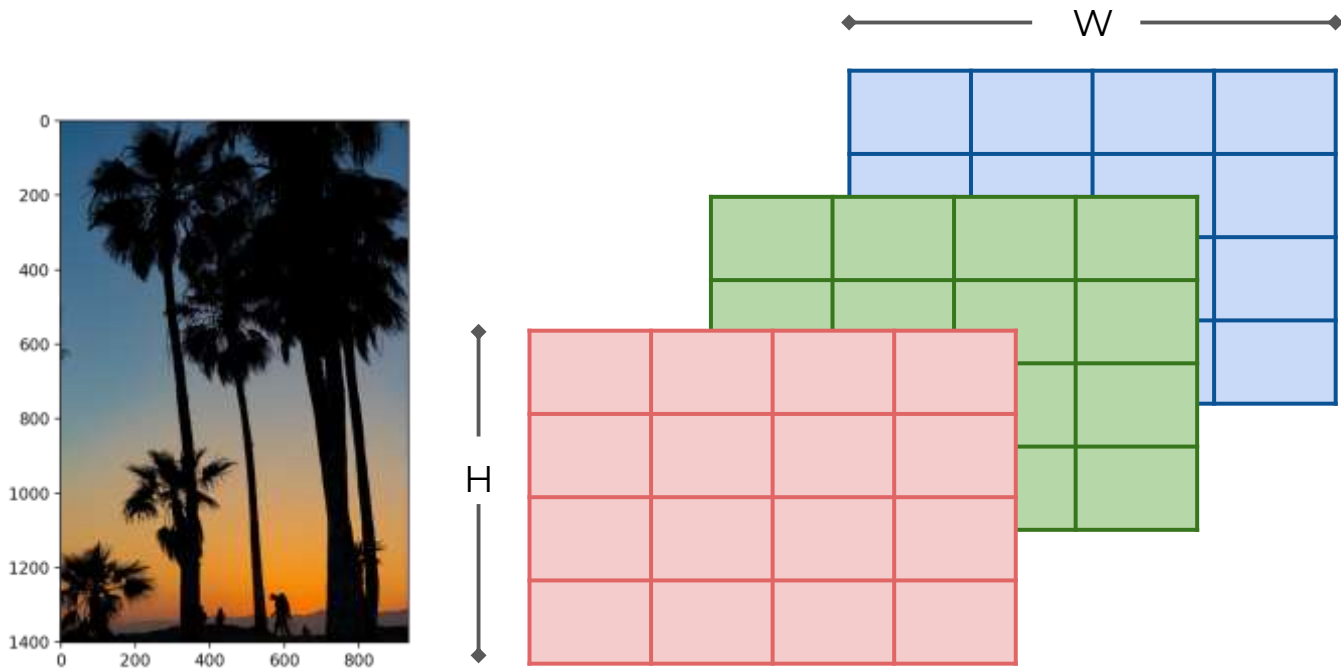
- Recall the image is a 3D array (H,W,C):





# K Means Clustering

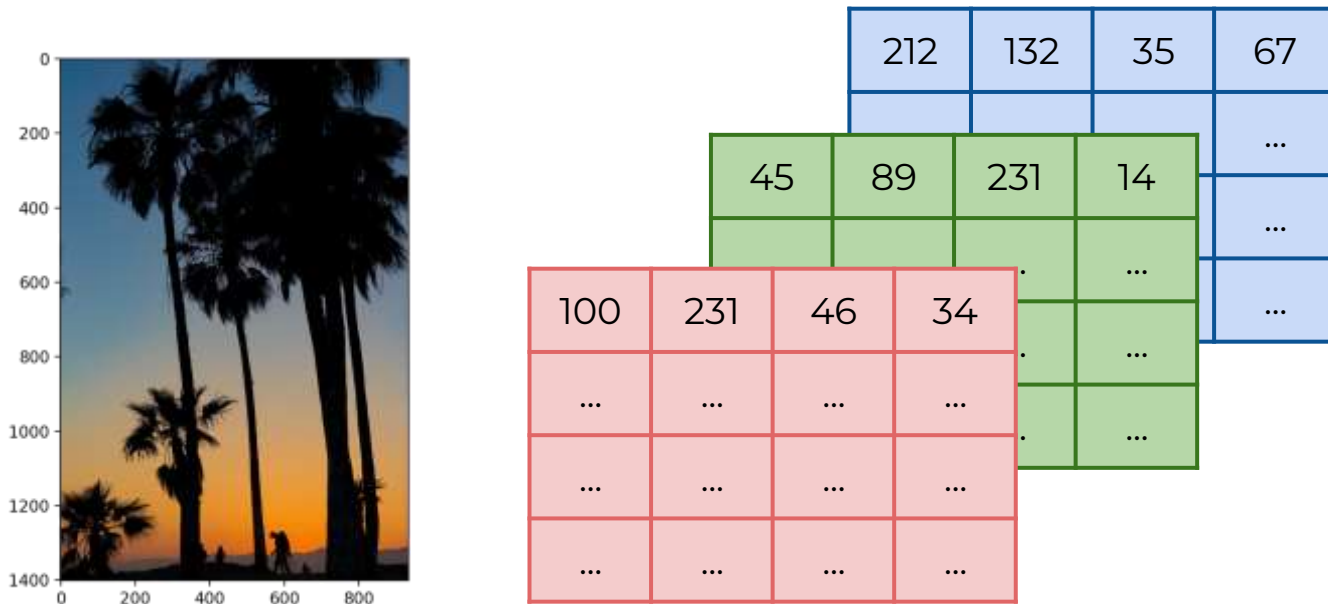
- Recall the image is a 3D array (H,W,C):





# K Means Clustering

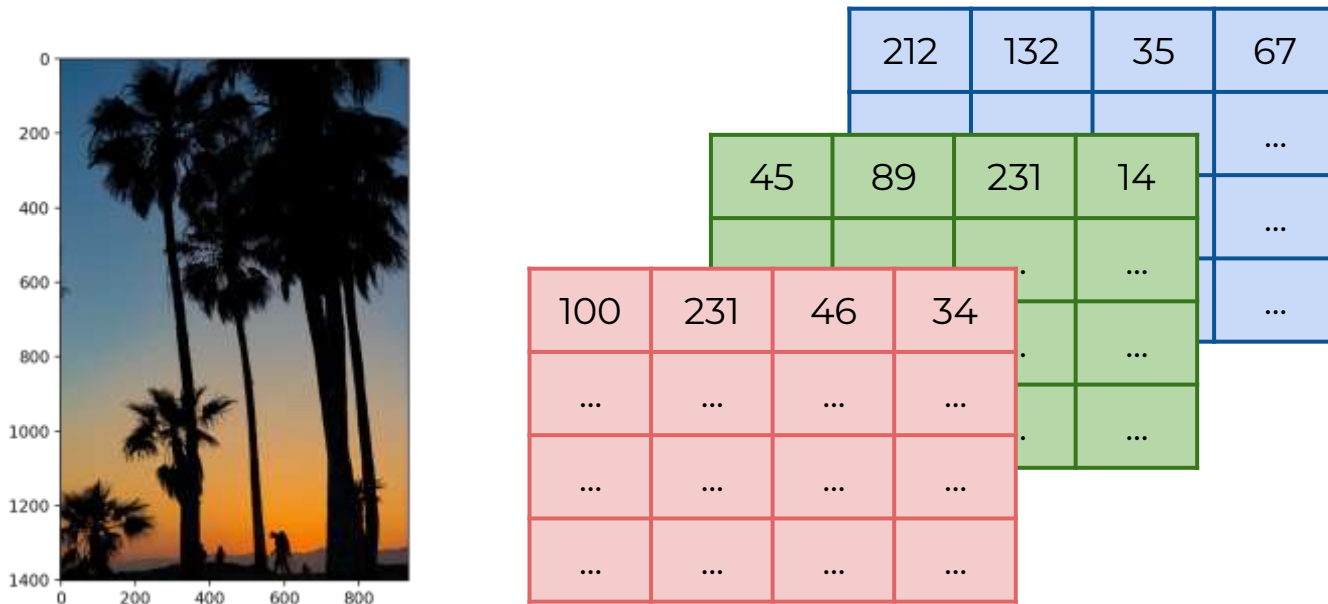
- Each pixel has an RGB value to create a color:





# K Means Clustering

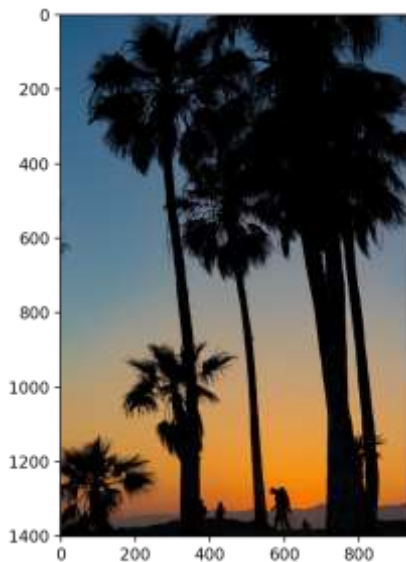
- We can reshape the image to an X array feature set, with features R,G,B:





# K Means Clustering

- We can reshape the image to an X array feature set, with features R,G,B:

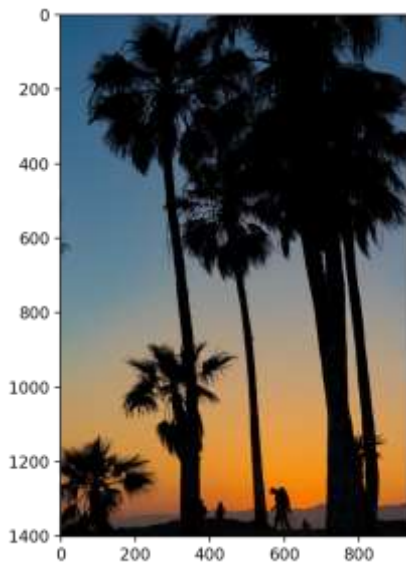


<b>R</b>	<b>G</b>	<b>B</b>
100	15	132
231	23	78
...	...	...
46	243	164
34	145	67



# K Means Clustering

- We can reshape the image to an X array feature set, with features R,G,B:



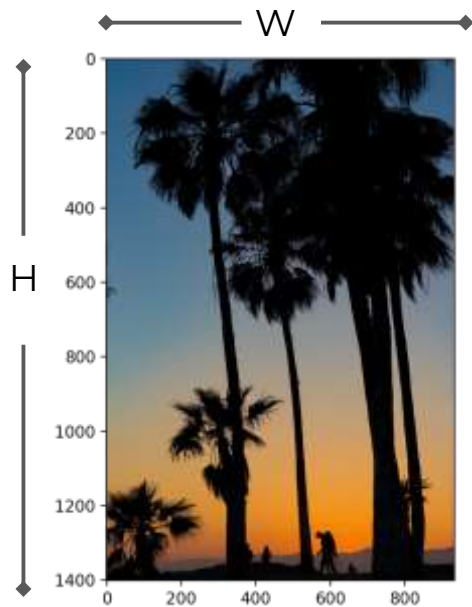
R	G	B
100	15	132
231	23	78
...	...	...
46	243	164
34	145	67





# K Means Clustering

- We can reshape the image to an X array feature set, with features R,G,B:



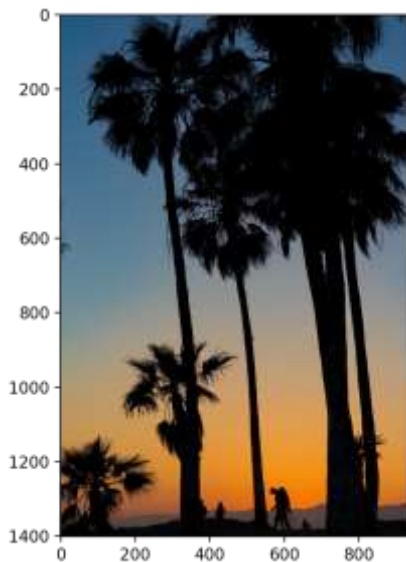
H x W

R	G	B
100	15	132
231	23	78
...	...	...
46	243	164
34	145	67



# K Means Clustering

- We then choose a K value of colors and use K Means clustering to create labels:

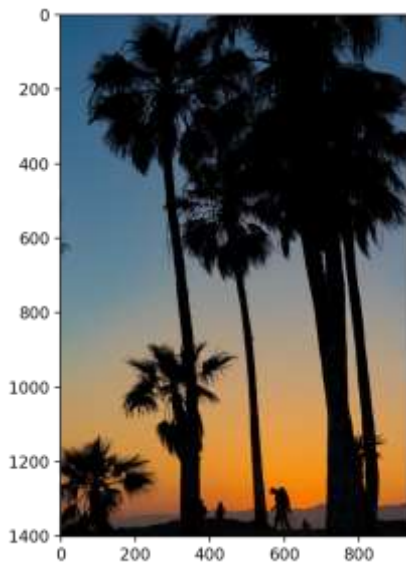


R	G	B
100	15	132
231	23	78
...	...	...
46	243	164
34	145	67



# K Means Clustering

- We then choose a K value of colors and use K Means clustering to create labels:

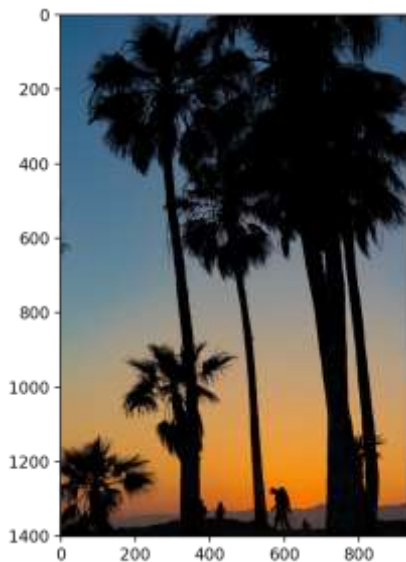


R	G	B	Cluster
100	15	132	0
231	23	78	1
...	...	...	...
46	243	164	2
34	145	67	0



# K Means Clustering

- Recall each cluster also has a **center** in the N dimensional feature space:

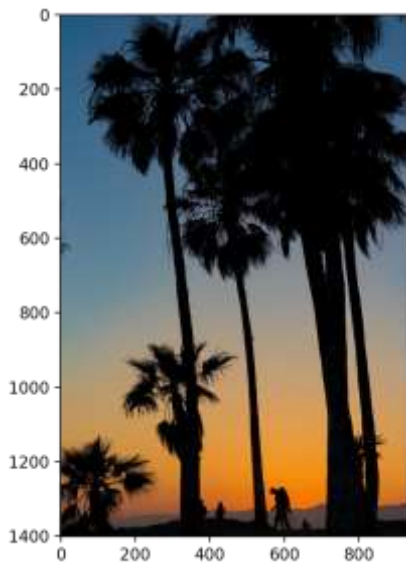


R	G	B	Cluster
100	15	132	0
231	23	78	1
...	...	...	...
46	243	164	2
34	145	67	0



# K Means Clustering

- Meaning each cluster center is an average (R,G,B) value we can use for reassignment!

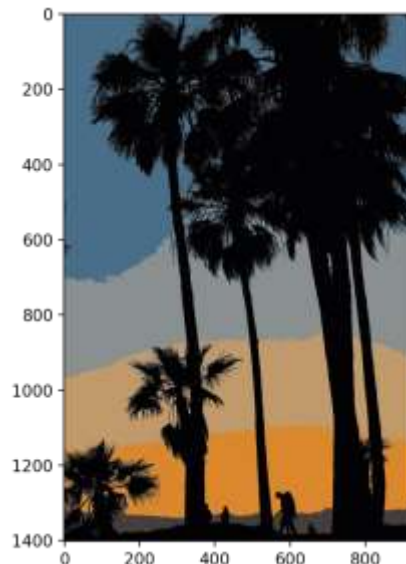
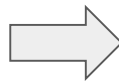
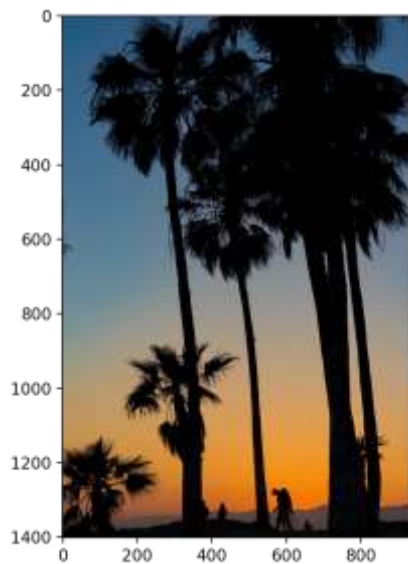


R	G	B	Cluster
100	15	132	0
231	23	78	1
...	...	...	...
46	243	164	2
34	145	67	0



# K Means Clustering

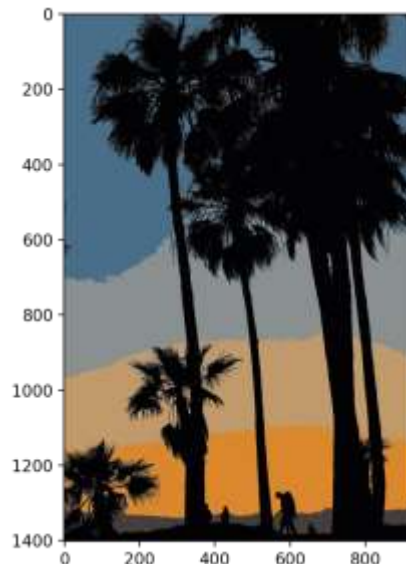
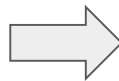
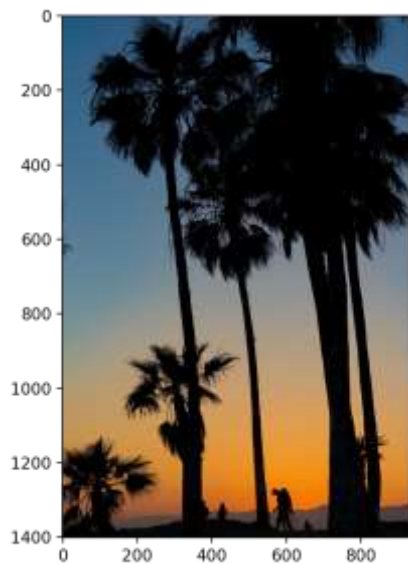
- We can then grab each data point and convert it to the same value as the center.





# K Means Clustering

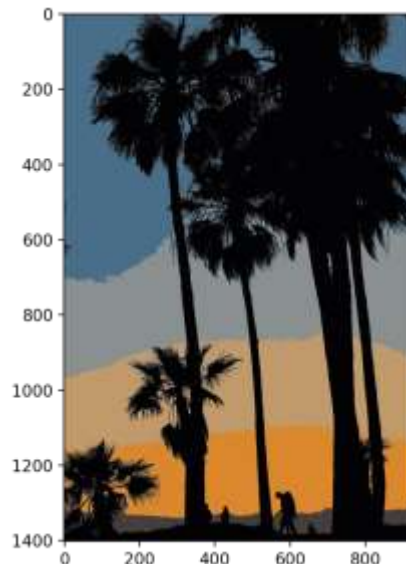
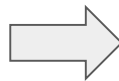
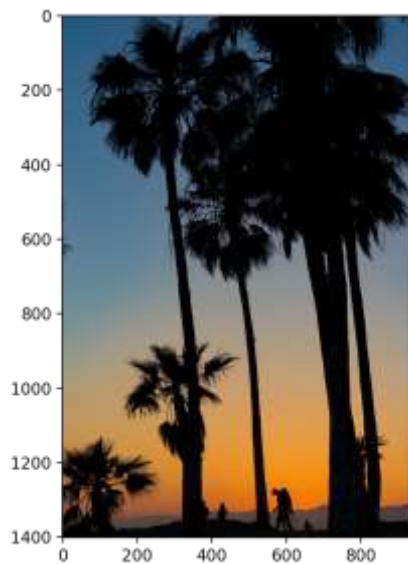
- This directly reduces to K color values (known as quantization).





# K Means Clustering

- Let's explore this in practice in the next lecture!







# K-Means Clustering Color Quantization

Part Two: Coding



# **K-Means Clustering Exercise Overview**



# K-Means Clustering Exercise Solutions

Part Three