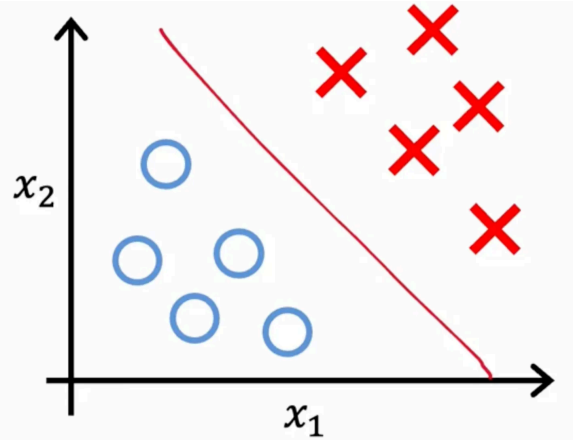
Clustering

What is clustering?

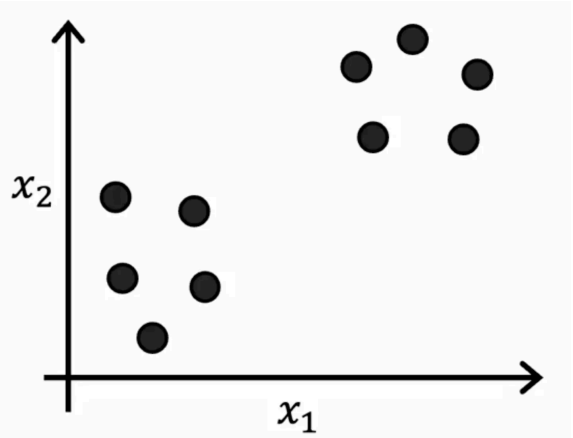
Supervised learning



Tranning Set:

$\{(x^{(1)}, y^{(1)}), (x^{(2)}, y^{(2)}), \ldots, (x^{(n)}, y^{(n)})\}$

Unsupervised learning



Tranning Set:

$\{x^{(1)}, x^{(2)}, \ldots, x^{(m)}\}$

Clustering

- Clustering algorithms identify groups of similar data points without prior labels.
- Unlike supervised learning, which uses labeled data to predict outcomes, clustering seeks to uncover patterns in unlabeled data.

Applications of Clustering

- Clustering can group similar news articles or segment markets based on user interests.
- It is also applied in analyzing DNA data to find individuals with similar genetic traits.
- Astronomers use clustering to analyze celestial bodies, identifying which belong to the same galaxy or structure.

K-means intuition

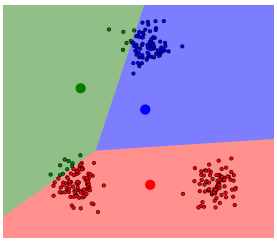
- The algorithm starts by making a random guess for the centers of the clusters, known as cluster centroids.
- In this example, two clusters are sought, and the algorithm randomly selects two initial points as centroids.

Steps of the K-means Algorithm

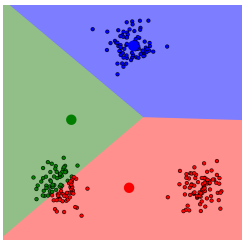
- **Assign Points to Centroids**: Each data point is evaluated to determine whether it is closer to one centroid or the other, and points are assigned accordingly.
- **Update Centroids**: After assigning points, the algorithm calculates the average position of the points in each cluster and moves the centroids to these new average locations.

Visualizing:

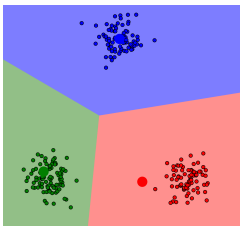
Thanks to https://www.naftaliharris.com/blog/visualizing-k-means-clustering/ for this comprehensive visualization.



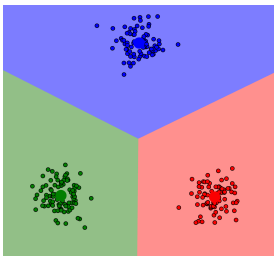
1.1 Assign Points



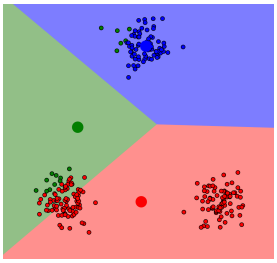
2.1 Assign Points



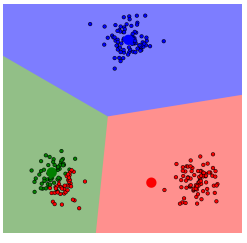
3.1 Assign Points



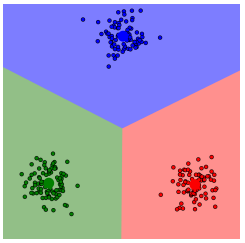
4.1 Assign Points



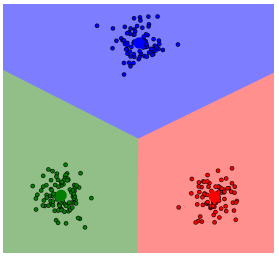
1.2 Update centroids



2.2 Update centroids



3.2 Update centroids



4.2 Update centroids

K-means algorithm

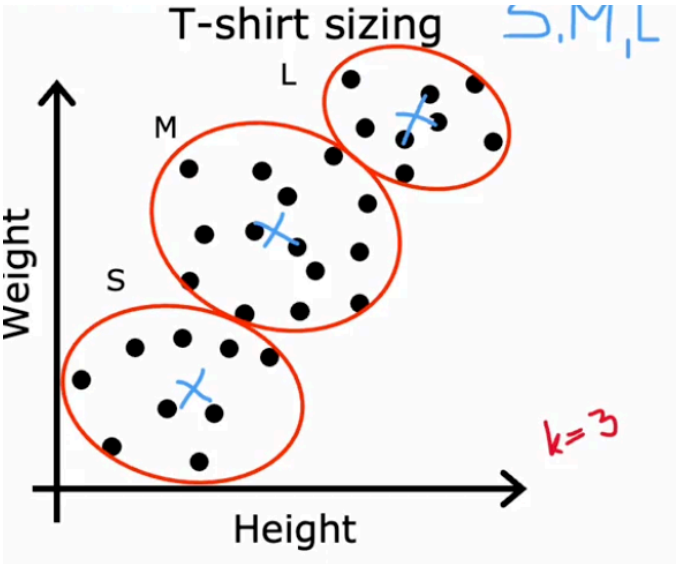
Randomly initializing K cluster centroids $(\mu_1, \mu_2, \ldots, \mu_k)$, m examples and n features,

Algorithm 1 K-Means Clustering Algorithm

repeat
  Assign points to cluster centroids
  for $i = 1$ to m **do**
    $c^{(i)} :=$ index (from 1 to K) of cluster centroid closest to $x^{(i)}$
      $= \min_{k \in \{1,\ldots,K\}} \left\| x^{(i)} - \mu_k \right\|^2$
  Move cluster centroids
  for $k = 1$ to K **do**
    $\mu_k :=$ mean of points (vector of size n) assigned to cluster k
until the algorithm converges

K-means for clusters that are not well separated

- K-means can be applied even when clusters are not well-separated, such as in sizing t-shirts based on customer height and weight data.



Optimization objective

K-means optimization objective

Given the following definition,

$c^{(i)}$ = index of cluster $(1, \dots, K)$ to which example training example $x^{(i)}$ is currently assigned.

μ_k = cluster centroid k

$\mu_{c^{(i)}}$ = cluster centroid of cluster to which example $x^{(i)}$ has been assigned.

The cost function (called **distortion function**) is defined as,

$$J(c^{(1)}, \dots, c^{(m)}, \mu^{(1)}, \dots, \mu^{(k)}) = \frac{1}{m} \sum_{i=1}^{m} ||x^{(i)} - \mu_{c^{(i)}}||^2$$

⇒ try to minimize J:

$$min_{\substack{c^{(1)}, \dots, c^{(m)} \\ \mu^{(1)}, \dots, \mu^{(k)}}} J(c, \mu)$$

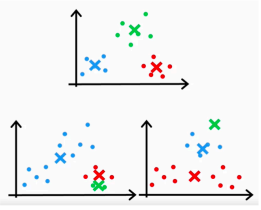
Cost function for K-means

$$J(c^{(1)}, \dots, c^{(m)}, \mu^{(1)}, \dots, \mu^{(k)}) = \frac{1}{m} \sum_{i=1}^{m} ||x^{(i)} - \mu_{c^{(i)}}||^2$$

Due to how the cost function is defined in terms of distances, minimizing it should always lead to convergence and never result in an increase. If the cost function value rises, it's likely there's an error in the code. Similar to gradient descent, the cost function may decrease in progressively smaller amounts, and the algorithm can be halted once these changes become negligibly small.

Initializing K-means

The first step in the algorithm is to randomly initialize the cluster centroids $\mu_1, \dots, \mu_K$. The number of clusters K should be less than the number of training examples m, as having more clusters than data points is impractical. The outcome of the algorithm depends on the initial centroid positions—for example, centroids grouped in one corner versus evenly spread out can lead to very different clusters.



- To address this variability, the algorithm is typically run multiple times with different initializations. For each run, the cost function J is calculated; suboptimal clusters result in a **higher cost** due to data points being **farther** from their centroids.
- The clustering result with the lowest cost is then chosen. In practice, 50 to 1000 random initializations are used, as going beyond that tends to be computationally expensive with little added benefit.

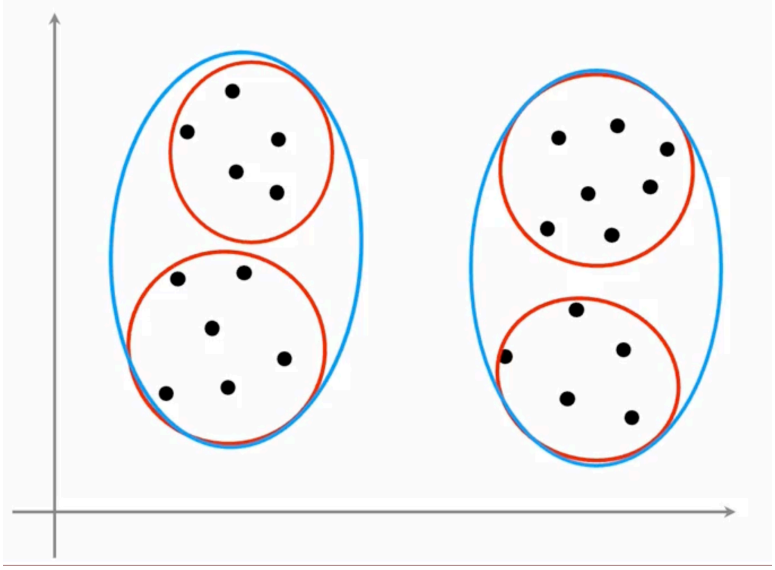
There are a few ways to address this:

- Run K-means clustering multiple times with different initial centers and choose the result where the final loss function reaches the smallest value.
- **K-means++** – an improved initialization algorithm.
- For those who want to explore further, you can refer to the scientific paper *"Cluster center initialization algorithm for K-means clustering."*

Choosing the number of clusters

What is right value of K?

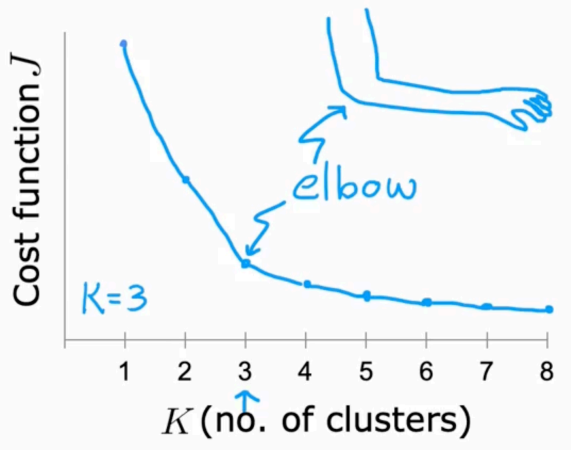
In the following example, $K \text{ should be } 2 \text{ or } 4?$



Choosing the Value of K

Elbow Method

The number of clusters K largely depends on the specific goal of the clustering task. However, when there's no prior insight into the appropriate number of clusters, the **elbow method** can be used to determine a suitable value. This approach involves computing the cost function (or distortion) for various values of K. By plotting these values, we look for a point where the cost starts decreasing at a slower rate—forming a shape similar to an "elbow." That point is considered a good choice for K. The following plot illustrates this elbow effect.



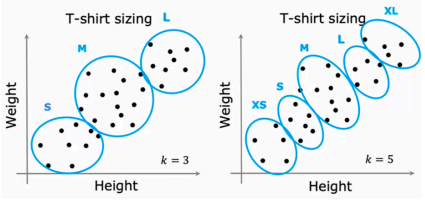
1. The **value** of K is often **ambiguous**; different observers may identify different numbers of clusters in the same dataset.
2. With a lot of cost functions just decreases smoothly and it doesn't have a clear elbow. We should remember that: "**Don't choose K to minimize cost J.**"
    - Because $J(c, \mu)$ is a **decreasing** function.

What if there is no elbow? How ?

- Often, you want to get clusters for some later (downstream) purpose.
- Evaluate K-means based on how well it performs on that later pupose.

For example, when sizing t-shirts based on height and weight data. Instead of picking arbitrary K values, consider the trade-off between the number of sizes and the fit versus manufacturing costs.

- That means it's more **practical** to choose values based on established sizing conventions (such as **S, M, L**).
- However, this can still be adjusted depending on the desired level of detail in the sizing.)

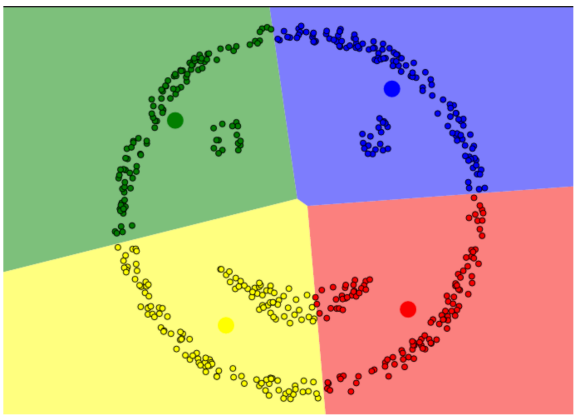


When a cluster is nested inside another cluster

source: MachineLearningCoBan

When one cluster lies inside another cluster

This is a classic example where K-means clustering fails to properly cluster the data. Naturally, we would separate the data into four clusters: left eye, right eye, mouth, and the area around the face. However, because the eyes and mouth are located within the face, K-means clustering is unable to perform the clustering correctly.



source: MLCoBan

Despite its limitations, **K-means clustering** remains extremely important in Machine Learning and serves as the foundation for many more advanced algorithms later on. We need to start with the basics. **Simple is best!**

Example

K-means

- At the end, your figure should look like the one displayed in Figure 1.
- The final centroids are the black X-marks in the middle of the colored clusters.
- You can see how these centroids got to their final location by looking at the other X-marks connected to it.

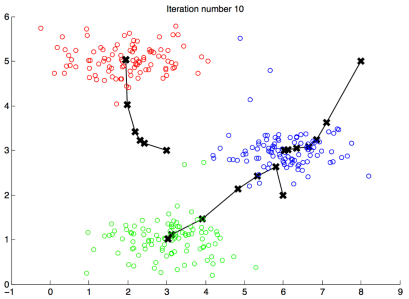


Figure 1: The expected output.

Random initialization

The initial assignments of centroids for the example dataset was designed so that you will see the same figure as in Figure 1. In practice, a good strategy for initializing the centroids is to select random examples from the training set.

This is how the function `kMeans_init_centroids` is implemented.

- The code first randomly shuffles the indices of the examples (using `np.random.permutation()`).
- Then, it selects the first K examples based on the random permutation of the indices.
- This allows the examples to be selected at random without the risk of selecting the same example twice.

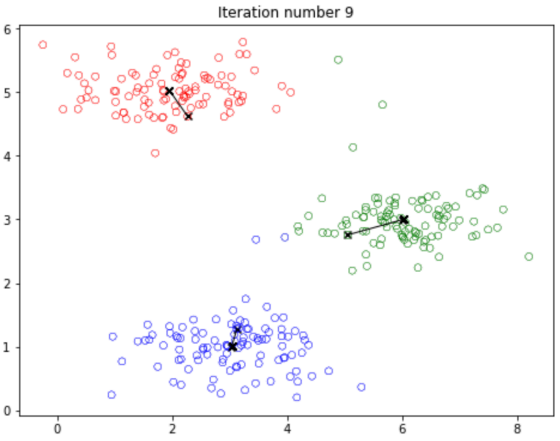
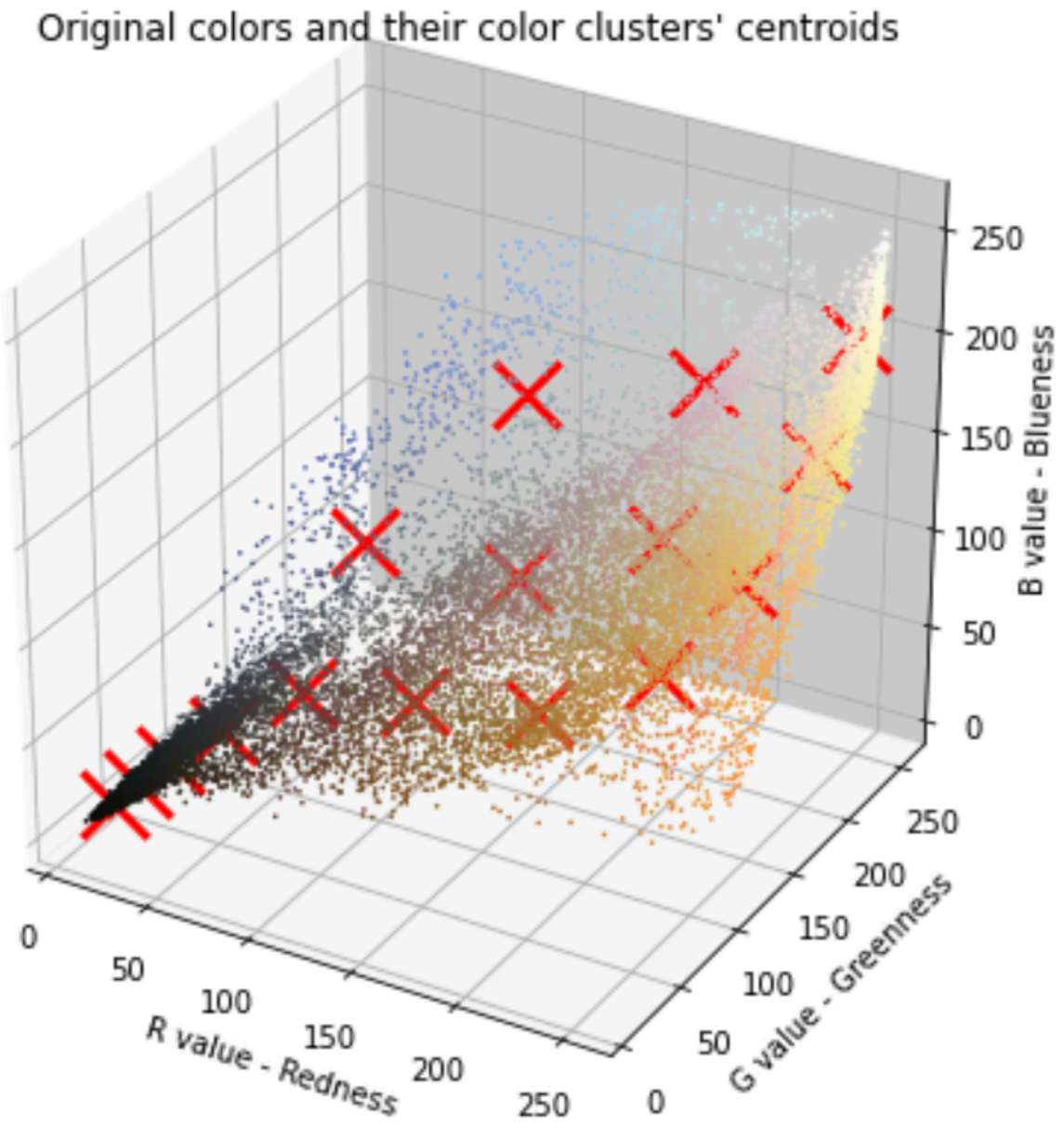


Image compression with K-means


Original colors and their color clusters' centroids
R value - Redness
G value - Greenness
B value - Blueness
0
50
100
150
200
250

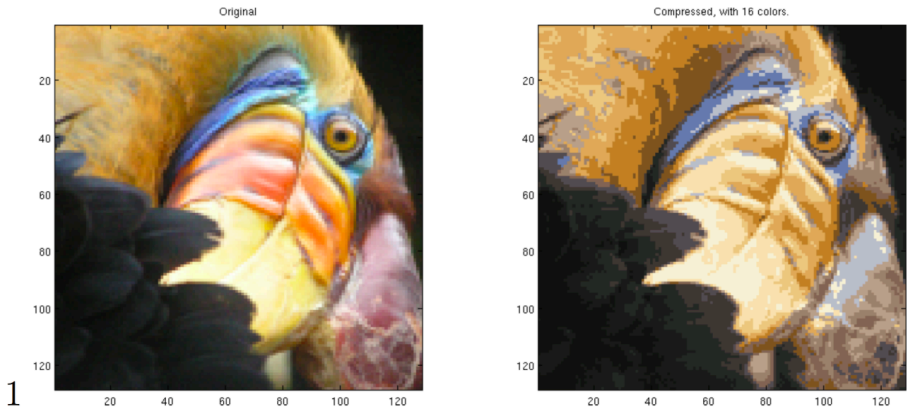


Figure 3: Original and reconstructed image (when using K-means to compress the image).

❓ **QUES:** Which of these best describes unsupervised learning?

- A form of machine learning that finds patterns without using a cost function.
- A form of machine learning that finds patterns using labeled data (x, y)
- A form of machine learning that finds patterns using unlabeled data (x). 👍
- A form of machine learning that finds patterns in data using only labels (y) but without any inputs (x).

Explain: Unsupervised learning uses unlabeled data. The training examples do not have targets or labels "y". Recall the T-shirt example. The data was height and weight but no target size.

❓ **QUES:** Which of these statements are true about K-means? Check all that apply.

- [x] ~~The number of cluster assignment variables $c^{(i)}$ is equal to the number of training examples.~~
- [ ] If you are running K-means with $K = 3$ clusters, then each $c^{(i)}$ should be 1, 2, or 3.
- [x] ~~The number of cluster centroids μ_k is equal to the number of examples.~~
- [x] ~~If each example x is a vector of 5 numbers, then each cluster centroid μ_k is also going to be a vector of 5 numbers.~~

❓ **QUES:** You run K-means 100 times with different initializations. How should you pick from the 100 resulting solutions?

- Pick the last one (i.e., the 100th random initialization) because K-means always improves over time
- Pick randomly -- that was the point of random initialization.
- Pick the one with the lowest cost J 👍
- Average all 100 solutions together.

Explain: K-means can arrive at different solutions depending on initialization. After running repeated trials, choose the solution with the lowest cost.

❓ **QUES:** You run K-means and compute the value of the cost function $J(c^{(1)}, \ldots, c^{(m)}, \mu^{(1)}, \ldots, \mu^{(k)})$ after each iteration. Which of these statements should be true?

- There is no cost function for the K-means algorithm.
- The cost can be greater or smaller than the cost in the previous iteration, but it decreases in the long run.
- The cost will either decrease or stay the same after each iteration. 👍
- Because K-means tries to maximize cost, the cost is always greater than or equal to the cost in the previous iteration.

Explain: The cost never increases. K-means always converges.

❓ **QUES:** In K-means, the elbow method is a method to

- Choose the maximum number of examples for each cluster
- Choose the best random initialization
- Choose the best number of samples in the dataset
- Choose the number of clusters K 👍

Explain: The elbow method plots a graph between the number of clusters K and the cost function. The 'bend' in the cost curve can suggest a natural value for K. Note that this feature may not exist or be significant in some data sets.