Computer Vision

Image Segmentation

Contents

- Segmentation
  - K-means
  - Mean shift

Why Image Segmentation?

Object detection

- Autonomous Vehicles need sensory input devices like cameras, radar, and lasers to allow the car to perceive the world around it, creating a digital map
- Autonomous driving is not possible without object detection which involves image classification/segmentation
- Detecting cancerous cell(s) as quickly as possible can potentially save millions of lives
- Shape of the cancerous cells plays a vital role in determining the severity of cancer which can be identified using image classification algorithms

Object Localization

- A classification model can classify the apple and orange with more than 95% accuracy
- If an Image contains both apple and orange the prediction accuracy reduces
- As the number of objects in the image increases the classification models' performances goes down

Why Image Segmentation?

- Object detection builds a bounding box corresponding to each class in the image
- Output of object detection contains bounding box coordinates
- Does not give information about the shape of the object

Object Detection

Why Image Segmentation?

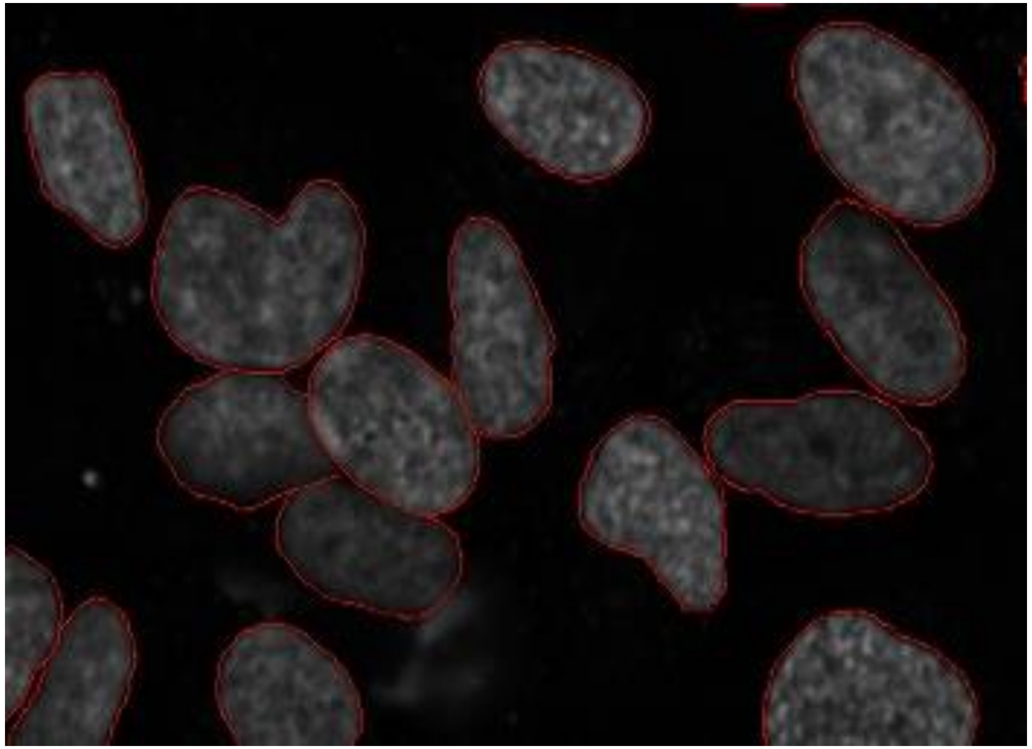
- Object detection builds a bounding box corresponding to each class in the image
- Output of object detection contains bounding box coordinates
- Does not give information about the shape of the object
- Image segmentation creates a pixel-wise mask for each object in the image
- This technique gives finer details of boundary of the objects

Object Detection

Image Segmentation

Ex: Image Segmentation?

- Image Segmentation can determine the shape of cancerous cells
- The shape of the cancerous cells plays a vital role in determining the severity of the cancer



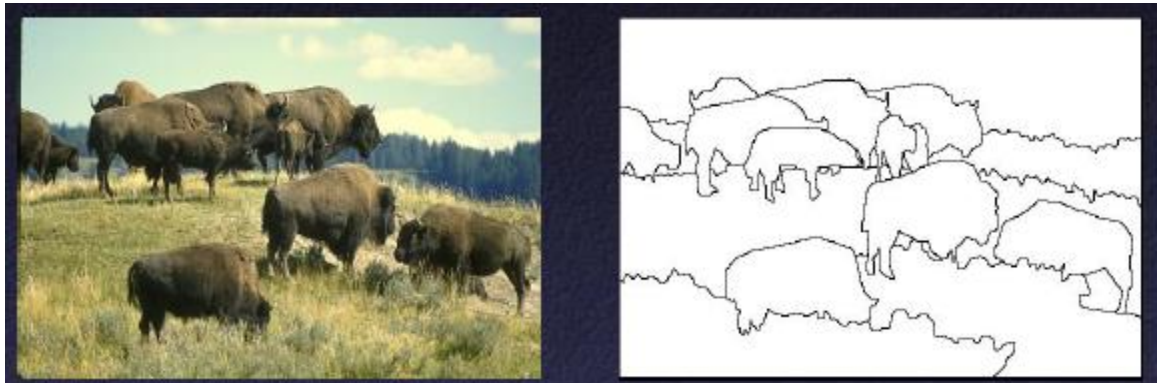
Introduction to segmentation

- Separate image into coherent regions
- Coherent means
  - Spatial proximity
  - Similar color
  - Similar texture

Introduction to segmentation

- Separate image into coherent regions
- Coherent means
  - Spatial proximity
  - Similar color
  - Similar texture

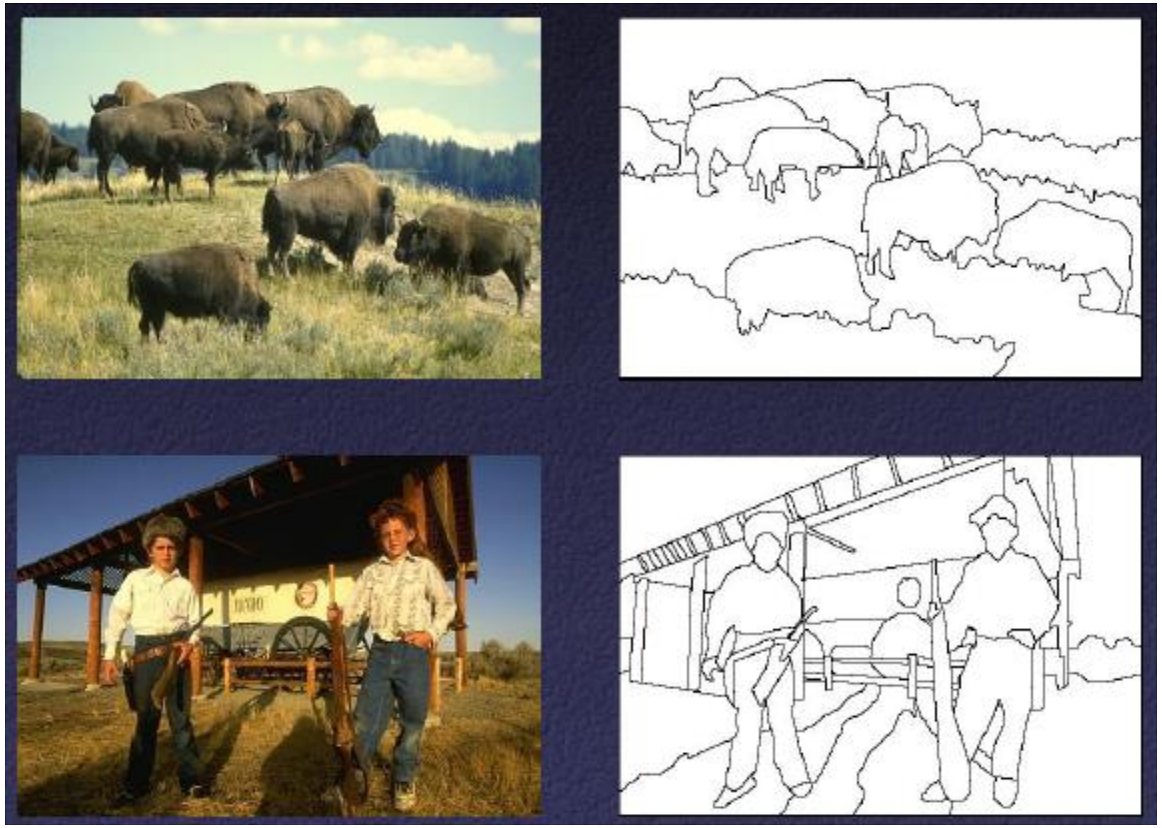
Texture Based segmentation



Introduction to segmentation

- Separate image into coherent regions
- Coherent means
  - Spatial proximity
  - Similar color
  - Similar texture

Texture Based segmentation



Color Based segmentation

Introduction to segmentation

- Partition an image into regions containing pixels with similar colors with connected pixels

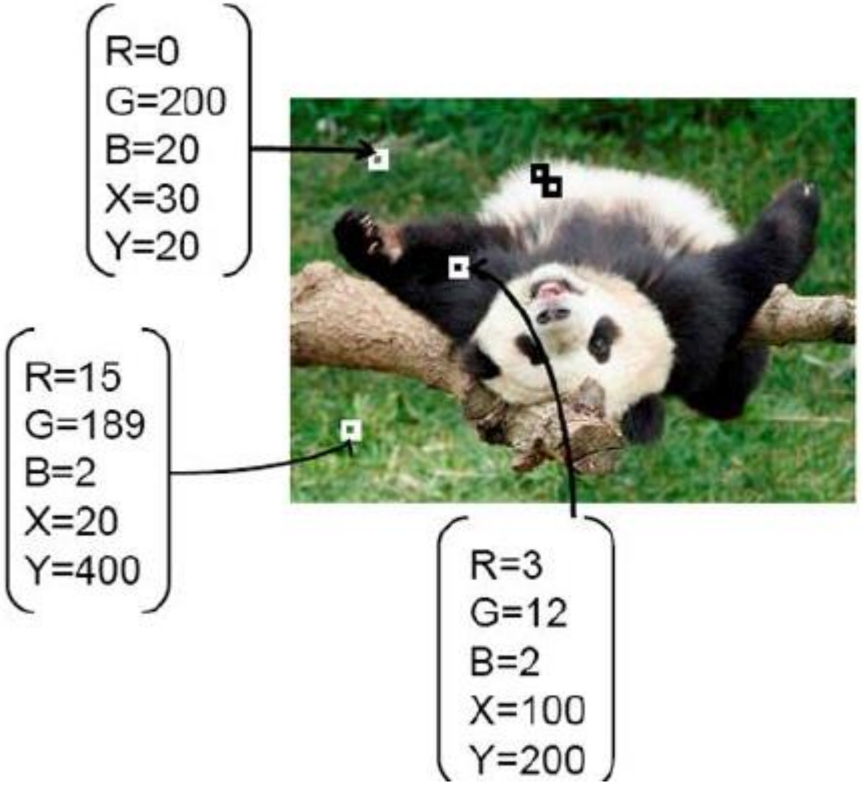


Image Segmentation

Only one object, dog

Cat-dog Classifier is simple

- Train a multi-label classifier
- Location of each object is also important
- For multiple objects, apply segmentation followed by localization and object detection

Image Localization

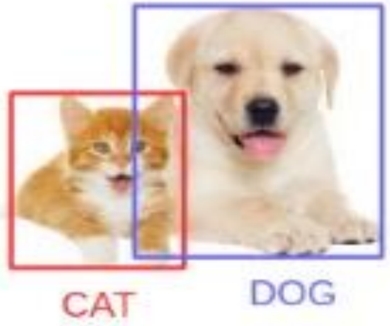
Image Segmentation

Only one object, dog

Cat-dog Classifier is simple

- Train a multi-label classifier
- Location of each object is also important
- For multiple objects, apply segmentation followed by localization and object detection

Image Localization



Object Detection

Image Segmentation

- Partitioning of image into connected homogeneous regions
- Homogeneity is defined in terms of:
  - Gray value
  - Color
  - Texture
  - Shape
  - Motion

Image Segmentation

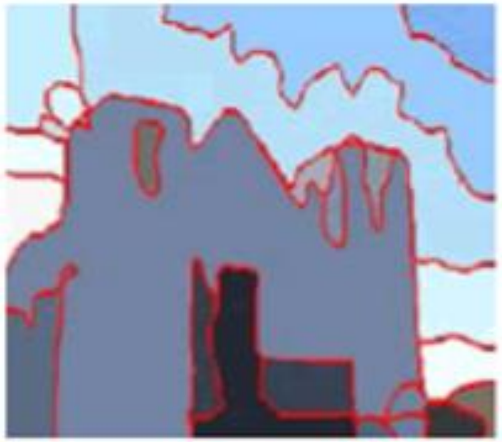
- Partition an image into multiple regions based on the characteristics of the pixels for each region
- Each region contains pixels with similar attributes
- Cluster similar pixels using a clustering algorithm to make a cluster
- Assign region label to pixels of each region
- Label identifies the region
- Typically used to locate objects and boundaries (lines, curves, etc.)

Types of segmentation algorithms

- Divisive clustering
- Hierarchical clustering
- K-means clustering
- Mean shift clustering
- Graph cuts…

- Can be divided into many narrower categories
  - Region-Based Segmentation
  - Edge Detection based Segmentation
  - Cluster-based segmentation
  - CNN based Segmentation

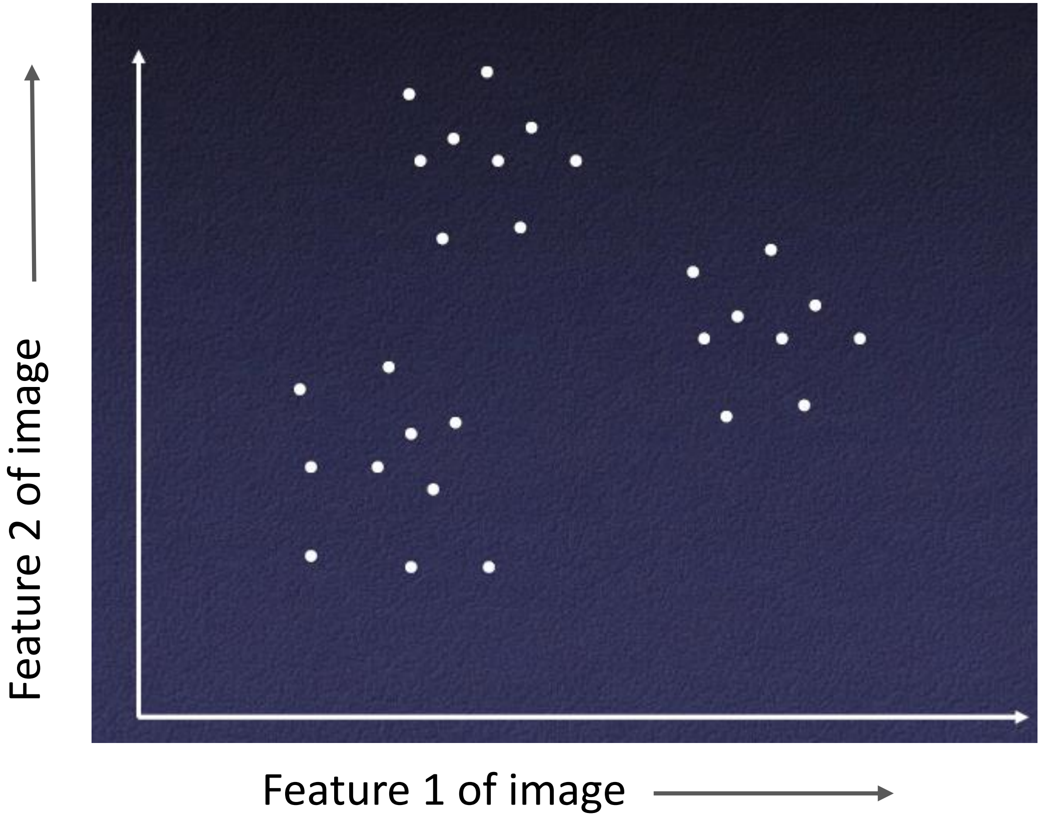
Image



Edge Detected

Segmented

Segmentation using clustering



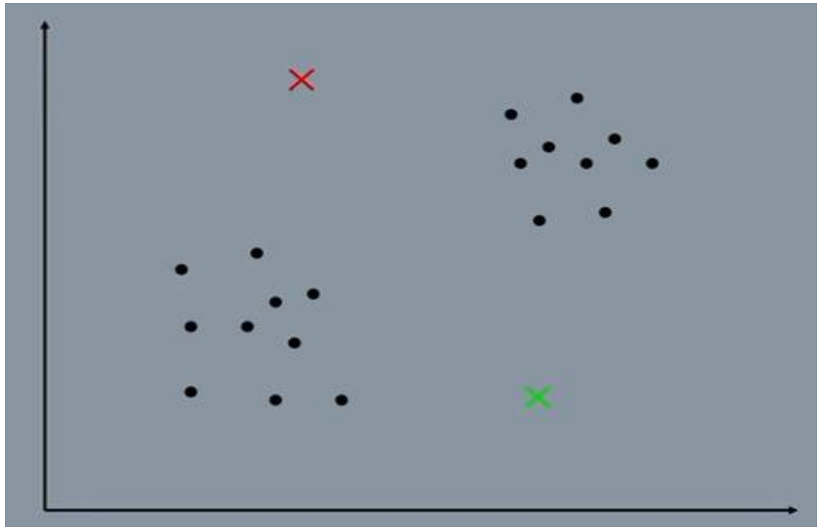
K-means clustering

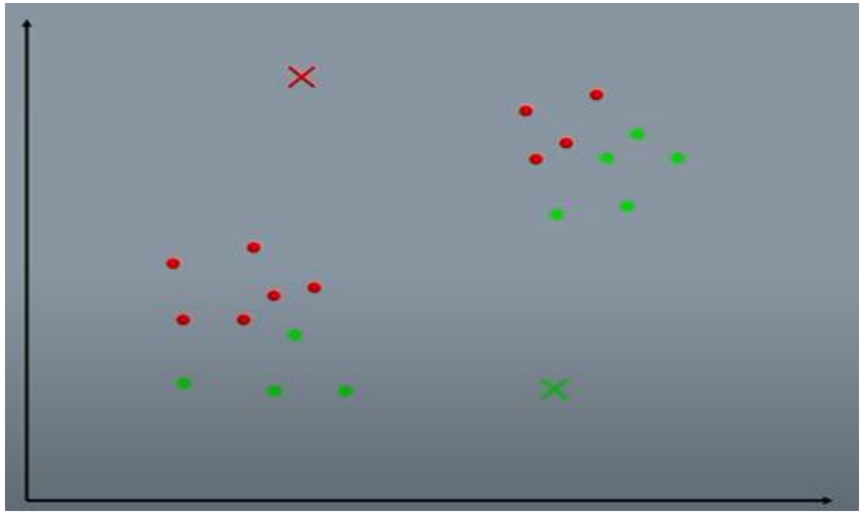
- Used to group data points that are more similar to each other, from other group data points
- Features could be texture, pixel values etc
- Unsupervised algorithm
- Steps

1. First, randomly select k initial clusters
2. Randomly assign each data point to any one of the k clusters
3. Calculate the centers of these clusters
4. Calculate the distance of all the points from the center of each cluster
5. Depending on the distance, the points are reassigned to the nearest cluster
6. Calculate the center of the newly formed clusters
7. Finally, repeat steps (4), (5) and (6) until either the center of the clusters does not change or we reach the set number of iterations

K-means clustering

- Selects k initial points, where k is the number of clusters
- Each of k points serves as an initial centroid for a cluster
- Assign closest points to the centroid

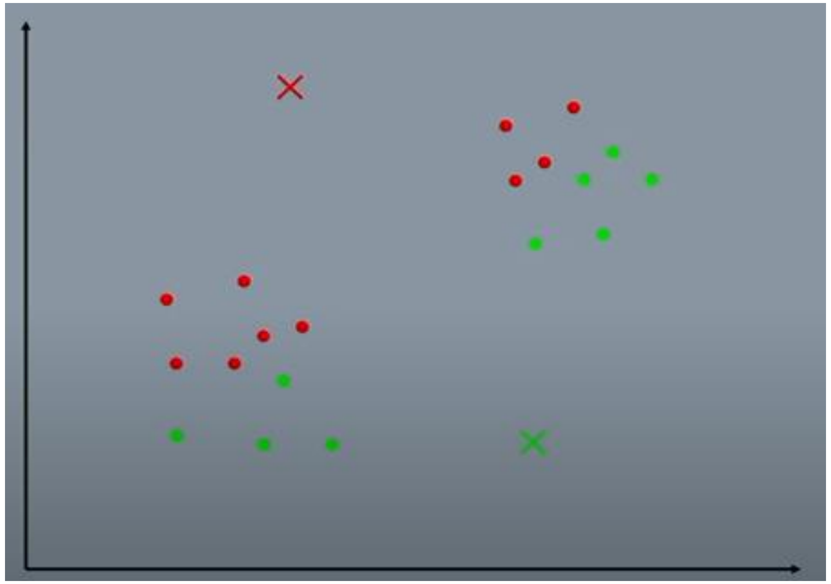




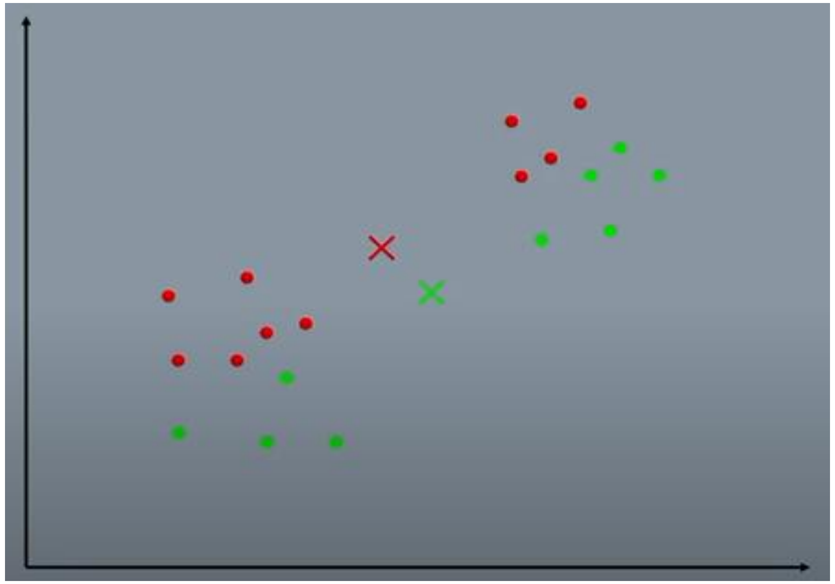
Assign data points to clusters

K-means clustering

- Recalculate the locations of the centroids
- Coordinate of the centroid is the mean value of all points of the cluster
- Reassign other points to new centroid which is closest
- The recalculation of centroids is repeated until a stopping condition is satisfied



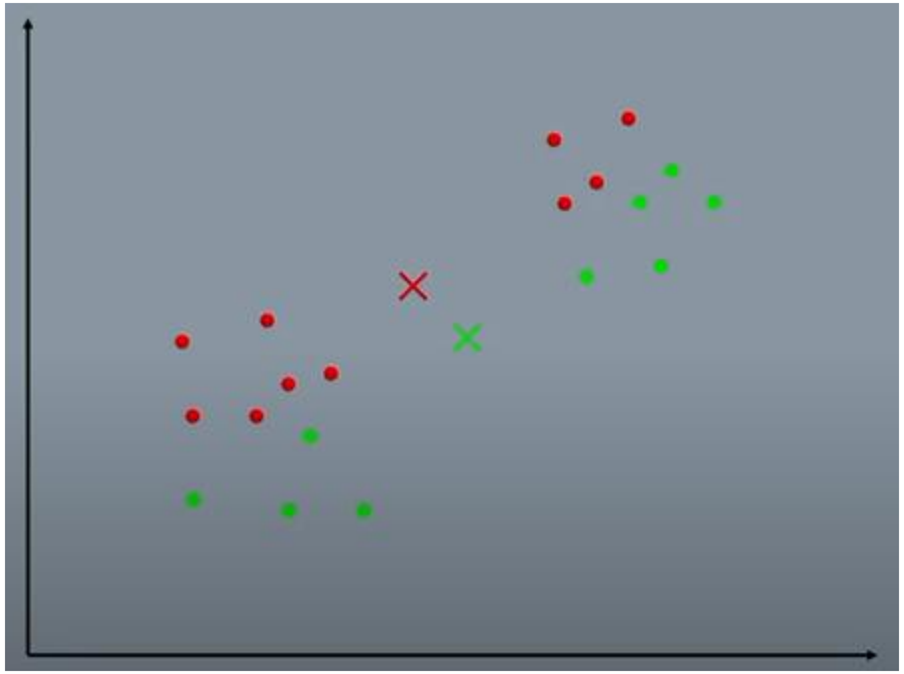
Assign data points to clusters



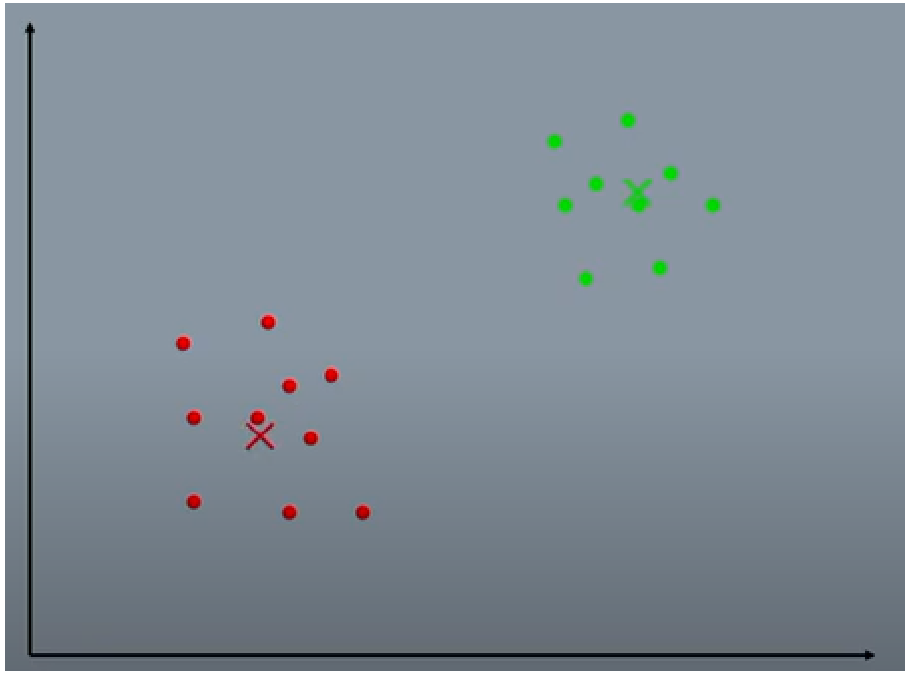
Recompute centroids

K-means clustering

The recalculation of centroids is repeated until a stopping condition is satisfied



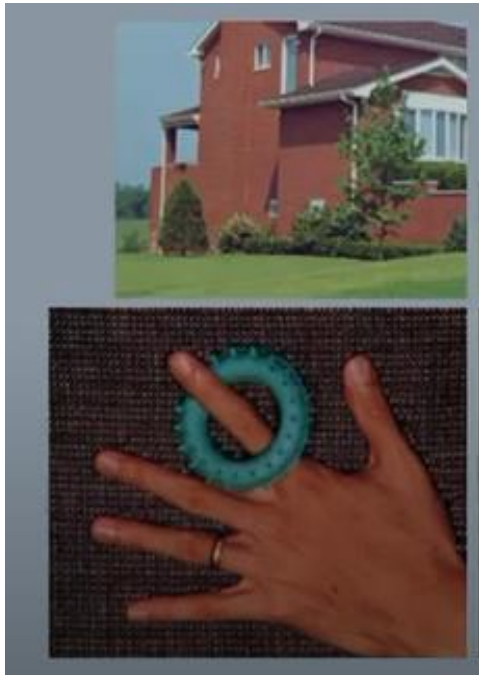
Recompute centroids



After a few iterations

K-means clustering

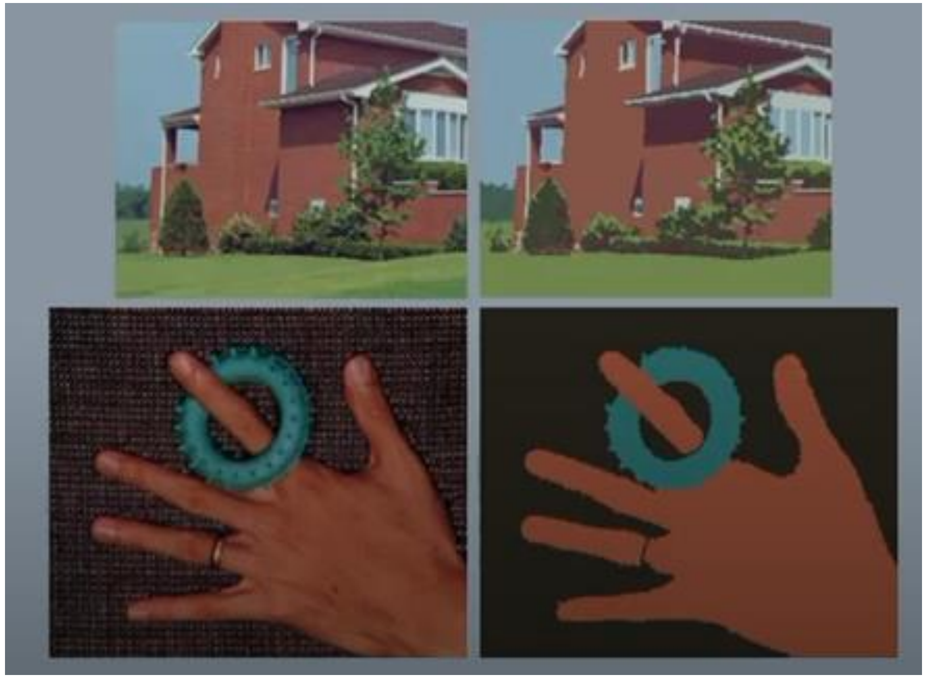
- Some common stopping conditions for k-means clustering are:
  - Centroids don’t change location anymore
  - Data points don’t change clusters anymore
  - Terminate training after a set number of iterations



examples

K-means clustering

- Some common stopping conditions for k-means clustering are:
  - Centroids don’t change location anymore
  - Data points don’t change clusters anymore
  - Terminate training after a set number of iterations



examples

K-means clustering (optimum value of ‘k’)

- Correct choice of number of clusters, K is ambiguous
- Choice of K depends on the shape and scale of the distribution of points in a data set
- And the desired clustering resolution
- Increasing K reduce the amount of error in the resulting clustering
- Elbow method can be used to determine the value of K

K means clustering (Elbow Method)

- Define clusters such that the total intra-cluster variation is minimized
- Within Cluster Sum of Square (WCSS) measures the compactness of the clustering
- Objective is to minimize WCSS between all points and the cluster centre within a cluster

number of clusters

number of cases

centroid for cluster *j*

case *i*

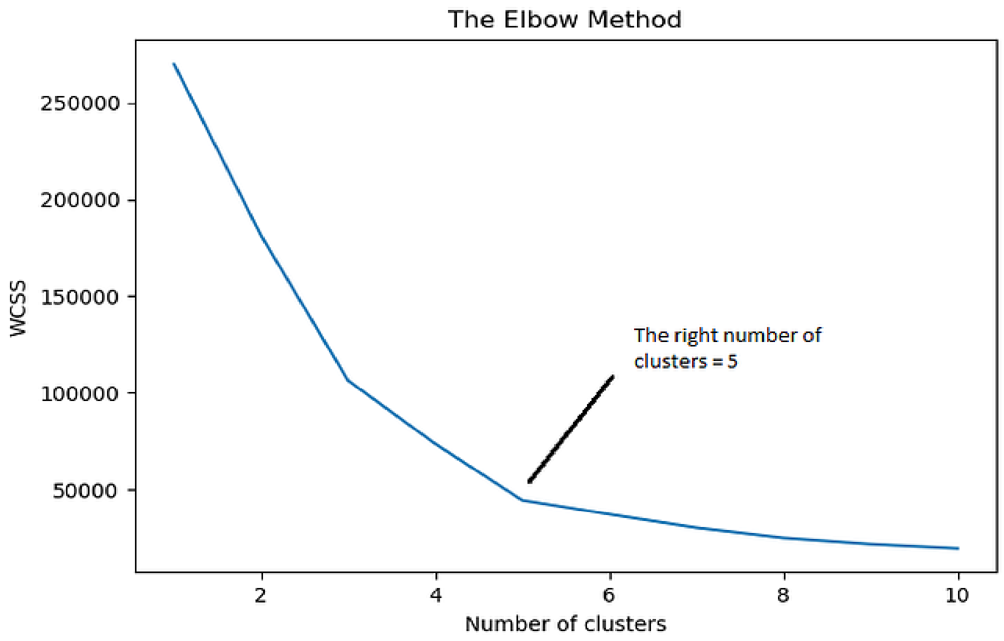
objective function

$$J = \sum_{j=1}^{k} \sum_{i=1}^{n} \underbrace{\left\| x_i^{(j)} - c_j \right\|^2}_{\text{Distance function}}$$

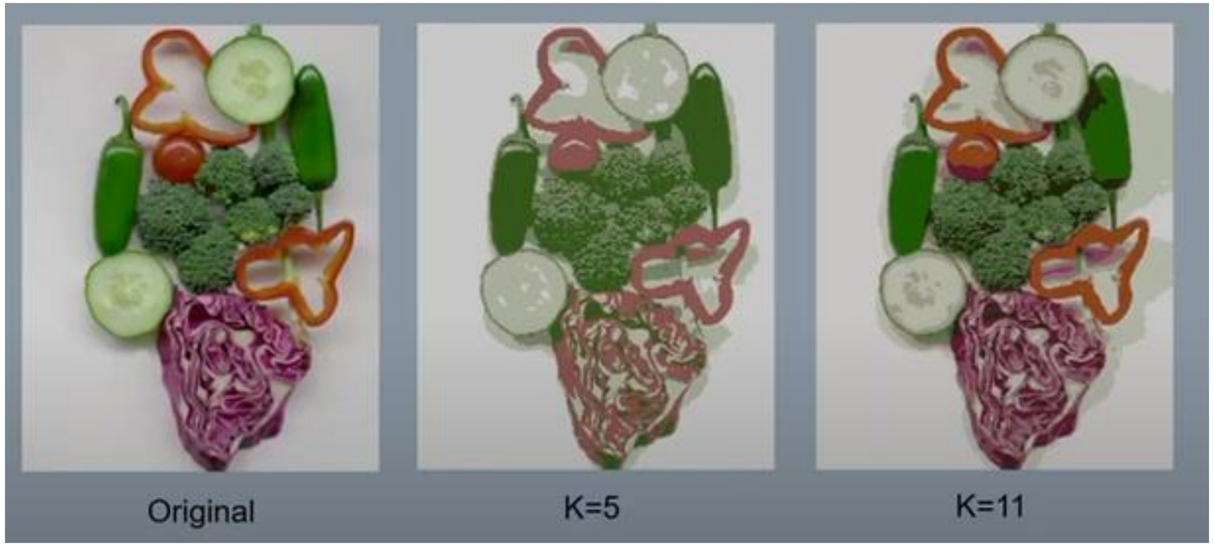
K means clustering (Elbow Method)

- Compute WCSS for different values of K by varying K
- For each K, calculate the total Within Cluster Sum of Square (WCSS)
- Plot the curve of WCSS vs the number of clusters K
- The location of a bend (knee) in the plot is considered as an indicator of the appropriate number of clusters

- K-Means can fail if choice of centroids is not correct
- Called The Random Initialization Trap

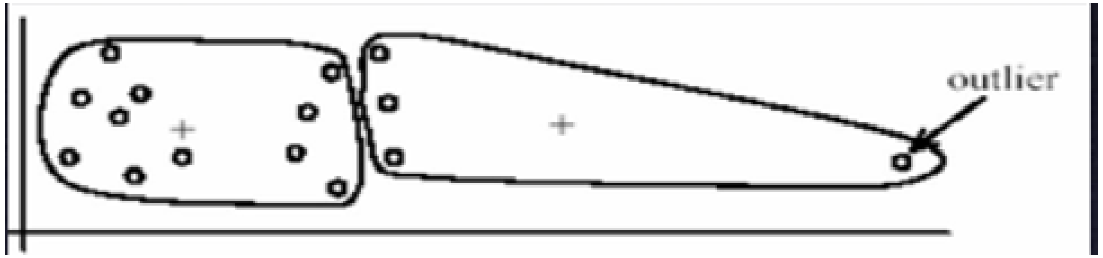


K means clustering (optimum value of ‘k’)



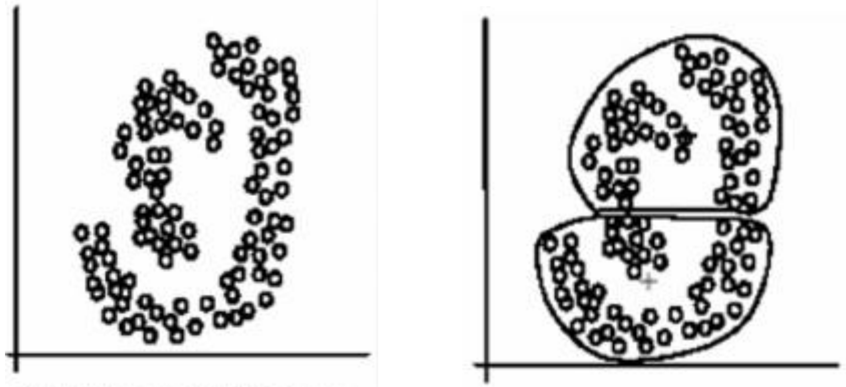
K-means algorithm

- Pros
  - Fastest unsupervised machine learning algorithm to break down data points into groups
  - Therefore, it is a good choice for large dataset
  - Complexity of algorithm is low
- Cons
  - Need to choose the value of K
  - Converges to a local minimum
  - Sensitive to initialization of centroid
  - It is sensitive to rescaling
  - Sensitive to outliers
  - Segments “spherical” clusters, does not work if clusters have a complex geometric shape



K-means segmentation

Ideal segmentation



(A): Two natural clusters (B): *k*-means clusters

Segmentation Techniques

Algorithm	Description	Advantages	Limitations
Region-Based Segmentation	• Separates the objects into different regions based on a threshold value(s)	• Simple calculations • Fast operation speed • When the object and background have high contrast, this method performs well	• When there is no significant grayscale difference or an overlap of the grayscale pixel values, it becomes very difficult to get accurate segments
Edge Detection Segmentation	• Makes use of discontinuous local features of an image to detect edges • hence define a boundary of the object	• It is good for images having better contrast between objects	• Not suitable when there are too many edges in the image and if there is less contrast between objects

Segmentation Techniques

Algorithm	Description	Advantages	Limitations
Segmentation based on Clustering	Divides the pixels of the image into homogeneous clusters.	• Works well on small datasets and generates excellent clusters	• Computation time is too large and expensive • k-means is not suitable for clustering non-convex clusters
Mask R-CNN	• Gives three outputs for each object in the image • its class, bounding box coordinates, and object mask	• Simple, flexible and general approach • It is also the current state-of-the-art for image segmentation	• High training time

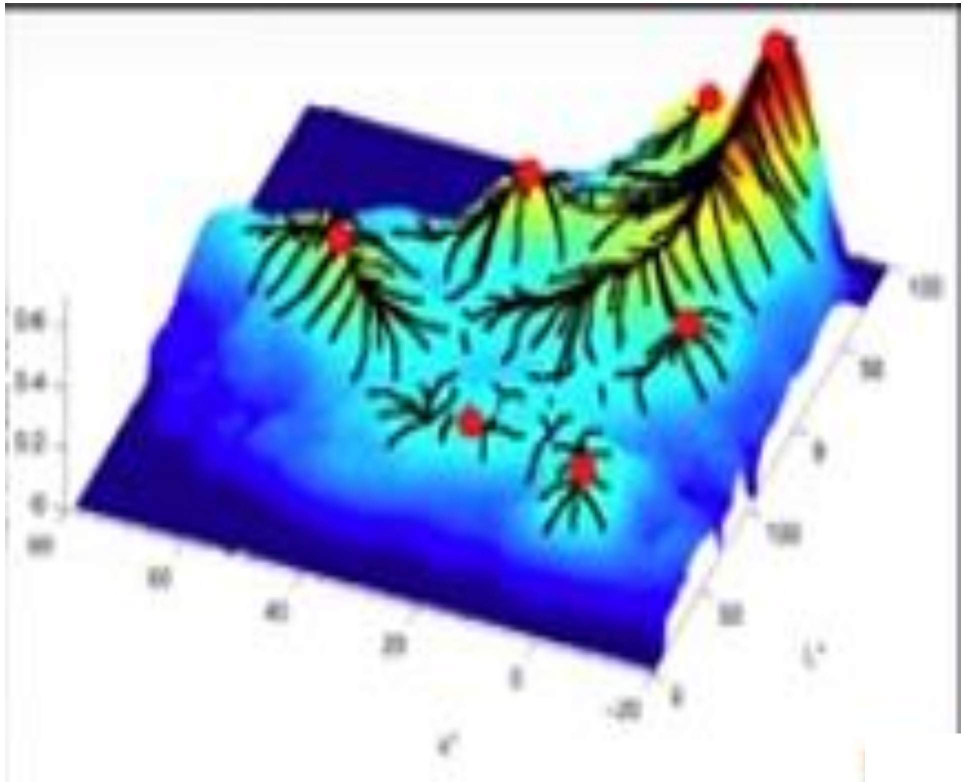
Mean Shift Algorithm

- Is a well-known method in computer vision for image segmentation
- Divides an image into meaningful zones according to color and space
- find clusters in data without specifying the number of clusters beforehand

Mean Shift Algorithm

- Also known as Mode-seeking algorithm
- Is an unsupervised learning clustering algorithm
- Number of clusters is dependent on the data
- Every data point is shifted to the “regional mean” in each iteration
- Location of the final destination of each point represents the cluster it belongs to
- Useful for datasets where the clusters have arbitrary shapes and are not well-separated by linear boundaries

Mean Shift Algorithm



Feature 2 →

Feature 1→

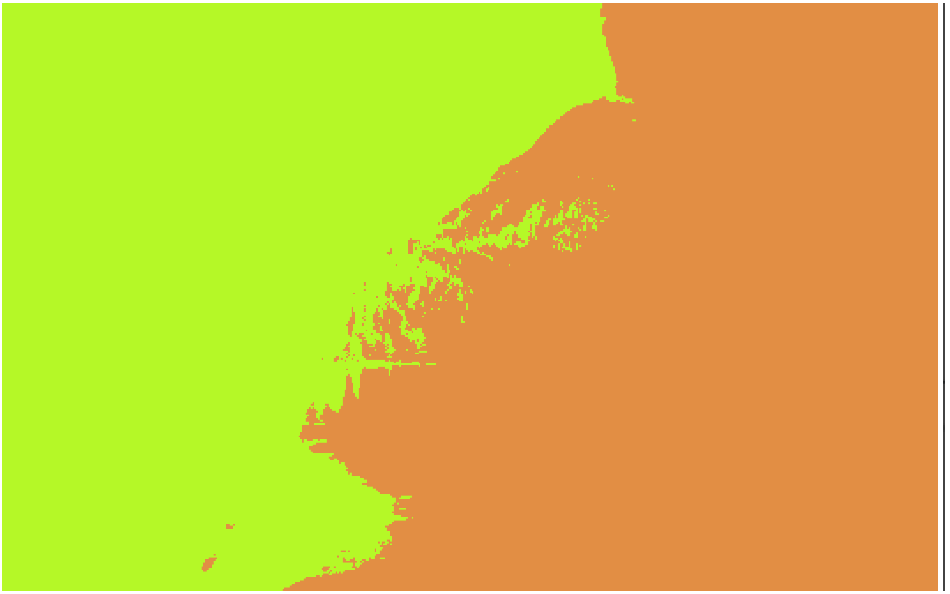
- Each hill represents one cluster
- Height of cluster is number of data points
- Peak (mode) of the hill represents the center of cluster
- Is based on the density of pixels with the same feature values
- Each pixel climbs up the hill within its neighbourhood

Steps: Mean Shift Algorithm

1. Convert Image to Feature Space:
   - Feature space for color image has 3 dimensions
   - A common feature space for images includes the spatial (x, y) coordinates and the color values (e.g., RGB or Lab color space), texture features etc
   - It can be a pixel distribution like histogram backprojection
   - Histogram backprojection is histogram of object which is searched in the reference image
2. Mean Shift Clustering:
   - For each pixel, perform Mean shift
   - Kernel moves to the nearest region of highest data density (mode)
3. Assign Labels:
   - Pixels that converge to the same mode are assigned the same label

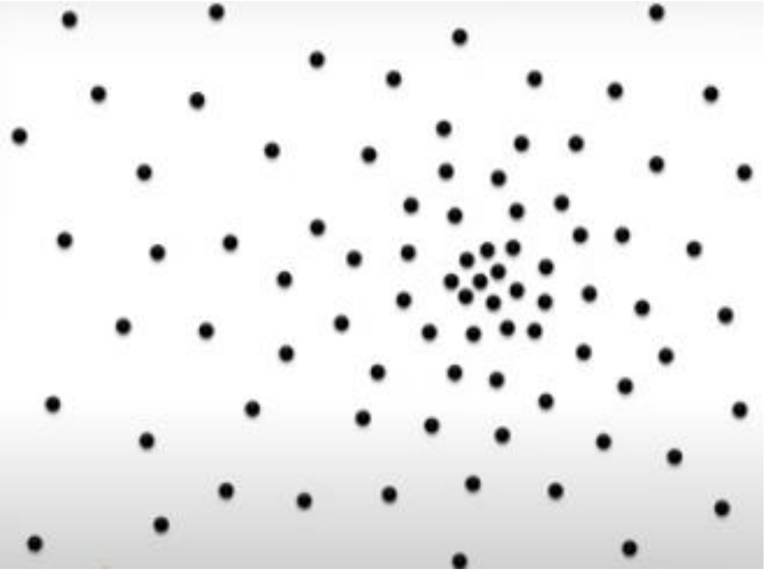
Mean Shift Algorithm

Image

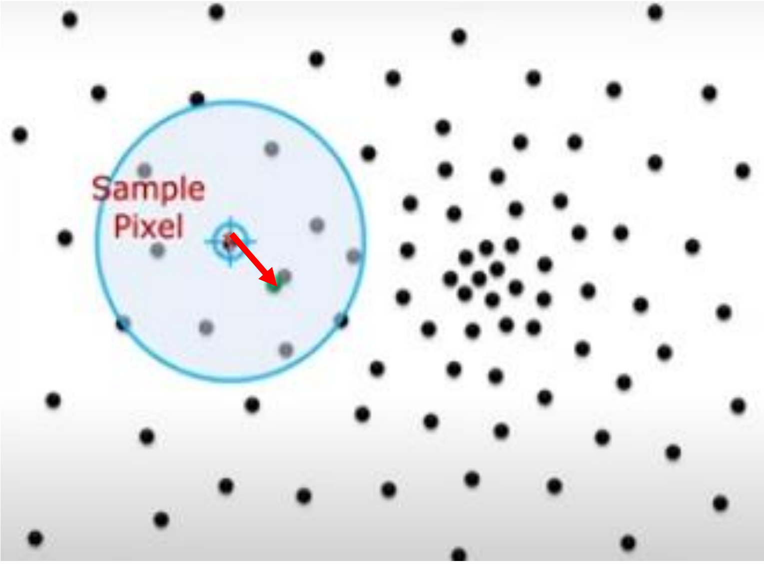


Segmented Image

Mean Shift Algorithm

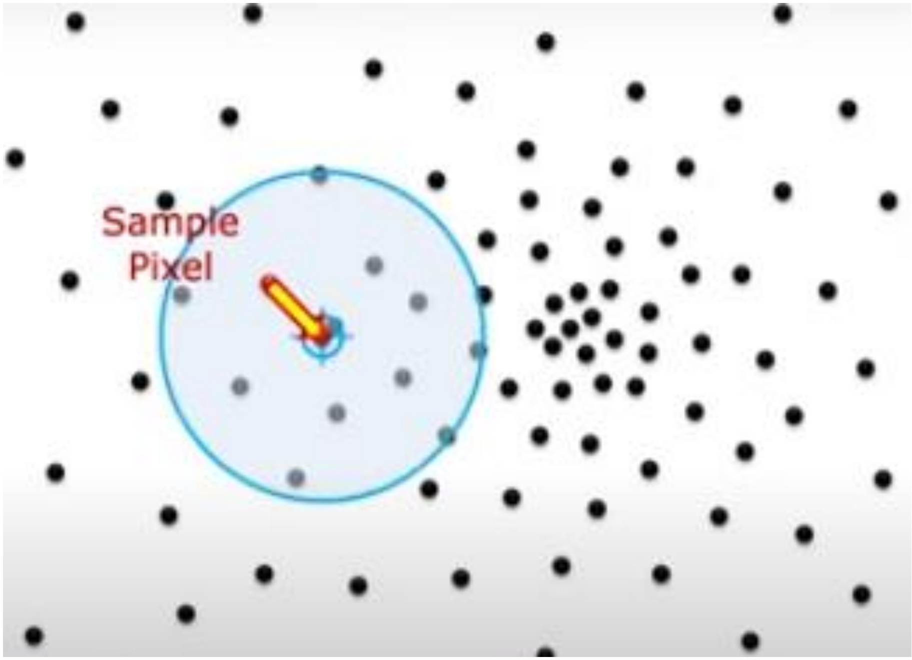


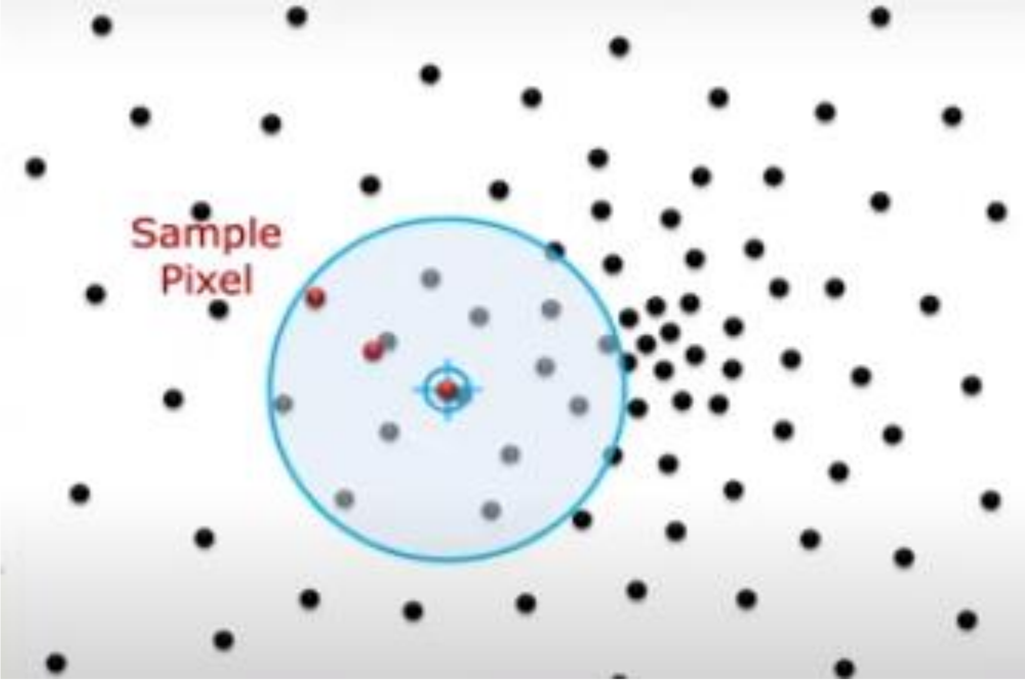
2D feature space



- Initially, mean value is pixel at a random location
- Choose window of size, R
- Determine mean
- Center window at new mean

Mean Shift Algorithm

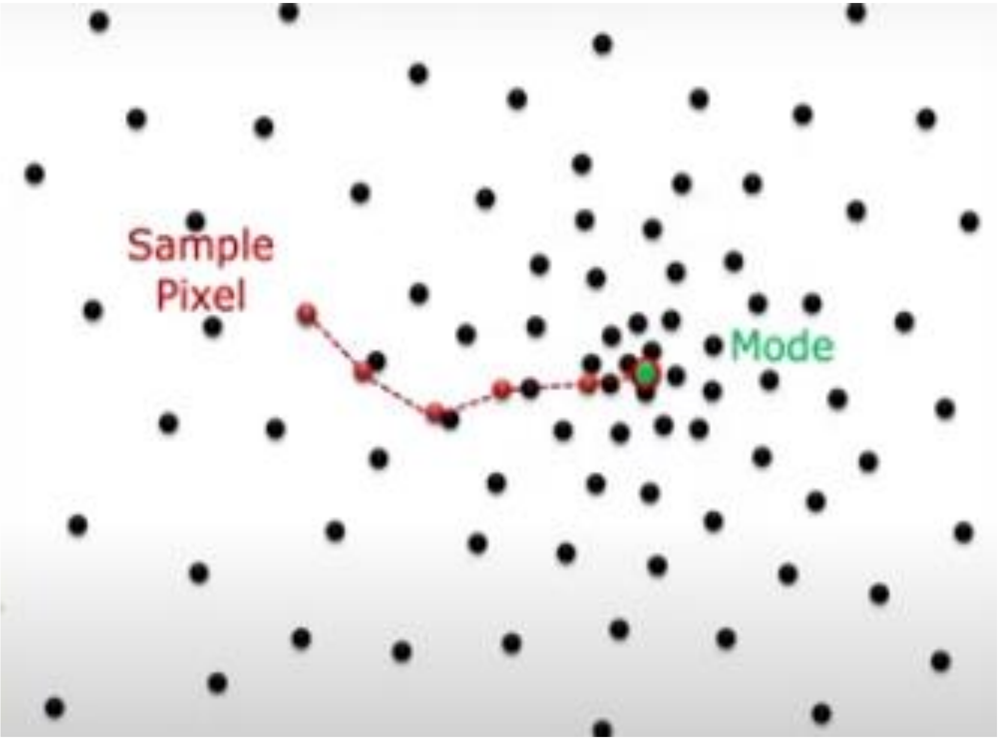




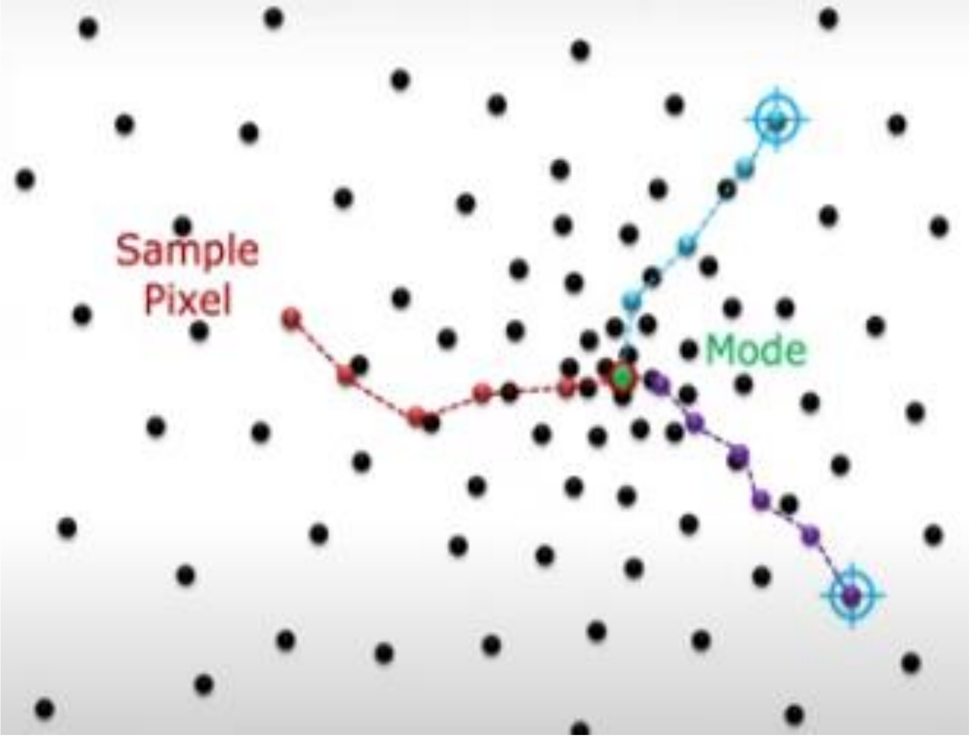
- Mean has shifted to new location
- Determine new mean

Repeat till there is no significant change (less than a threshold) in mean value

Mean Shift Algorithm



First Mode



Three modes

Example: Mean Shift Algorithm

$$\begin{bmatrix} 30 & 32 & 34 & 201 & 203 \\ 31 & 33 & 200 & 202 & 204 \end{bmatrix}$$

Image

- Color bandwidth, hc = 10
- Space bandwidth is not specified
- Therefore consider complete image

- Assume initial pixel with value, 33
  - Points within color bandwidth are {30, 31, 32, 33, 34}
  - Mean of points = 32
  - Shift 33 to 32

$$\begin{bmatrix} 30 & 32 & 34 & 201 & 203 \\ 31 & 32 & 200 & 202 & 204 \end{bmatrix}$$

- For pixel value, 32
  - Points are {30, 31, 32, 32, 34}
  - Mean = 32
  - Pixel 32 remains 32

Example: Mean Shift Algorithm

$$\begin{bmatrix} 30 & 32 & 34 & 201 & 203 \\ 31 & 32 & 200 & 202 & 204 \end{bmatrix}$$

Image

Color bandwidth, hc = 10

- For pixel value, 202
  - Points within color bandwidth are {200, 201, 202, 203, 204}
  - Mean = 202
  - No change in mean
  - Image is

$$\begin{bmatrix} 30 & 32 & 34 & 201 & 203 \\ 31 & 32 & 200 & 202 & 204 \end{bmatrix}$$

- Cluster 1: 30,31,32,33,34 (mean = 32)
- Cluster 2: 201,202,203,204 (mean = 202)

$$\begin{bmatrix} 30 & 32 & 34 & 201 & 203 \\ 31 & 33 & 200 & 202 & 204 \end{bmatrix}$$

Example: Mean Shift Algorithm

$$\begin{bmatrix} \color{red}{(255, 0, 0)} & (255, 0, 0) & (0, 255, 0) \\ (255, 0, 0) & (0, 255, 0) & (0, 255, 0) \\ (0, 255, 0) & (0, 255, 0) & (255, 0, 0) \end{bmatrix}$$ Color Image

- Bandwidth in RGB space = 50
- For (255, 0, 0)
  - Pixels within the bandwidth are (255, 0, 0), (255, 0, 0), (255, 0, 0) and (255, 0, 0)
  - Mean of these pixels = (255, 0, 0)
- For (0, 255, 0)
  - Pixels within the bandwidth are (0, 255, 0), (0, 255, 0), (0, 255, 0), (0, 255, 0), (0, 255, 0)
  - Mean of these pixels = (0, 255, 0)

Example: Mean Shift Algorithm

$$\begin{bmatrix} (255,0,0) & (255,0,0) & (0,255,0) \\ (255,0,0) & (0,255,0) & (0,255,0) \\ (0,255,0) & (0,255,0) & (255,0,0) \end{bmatrix}$$ Color Image

- Repeat the process for all pixels in the grid
- Clusters are red and green

Segmented Image

$$\begin{bmatrix} (255,0,0) & (255,0,0) & (0,255,0) \\ (255,0,0) & (0,255,0) & (0,255,0) \\ (0,255,0) & (0,255,0) & (255,0,0) \end{bmatrix}$$

Example: Mean Shift Algorithm

$$\begin{bmatrix} (255,0,0) & (255,100,0) & (0,200,0) & (0,200,200) \\ (200,0,0) & (0,255,0) & (0,255,255) & (255,255,0) \end{bmatrix}$$ Color Image

- Color bandwidth, hc=60
- For (255, 0, 0)
  - Distance from (200,0,0) $=\sqrt{(255-200)^2+(0-0)^2+(0-0)^2}$ =55
  - Distance from (255,100,0) = $\sqrt{(255-255)^2+(0-100)^2+(0-0)^2}$ =100 (not considered)
  - Distance from (0,255,0) = 255 (not considered)
  - Distance from (0,200,0) = 200 (not considered)
  - Distance from (0,255,255) = 360.62 (not considered)
  - Distance from (0,200,200) = 360.62 (not considered)
  - Distance from (255,255,0) = 255 (not considered)
  - Mean of (255, 0, 0) and (200,0,0)

    = (255+200, 0+0, 0+0)/2

    = 227.5→ 228

Example: Mean Shift Algorithm

$$\begin{bmatrix} (228,0,0) & (255,100,0) & (0,200,0) & (0,200,200) \\ (200,0,0) & (0,255,0) & (0,255,255) & (255,255,0) \end{bmatrix}$$

Color Image

- Color bandwidth, hc=60
- For (0, 255, 0)
  - Distance from (228,0,0) =
  - Distance from (0,200,0) = 55
  - Distance from (0,255,255) = 255
  - Distance from (0,200,200) = 206.15
  - Distance from (200,0,0) =
  - Mean of (0, 255, 0) and (0, 200, 0) = (0, 227.25, 0)
  - (0, 255, 0) → (0, 227, 0)
- For (0, 255, 255)
  - Mean = (0, 255, 255) and (0, 200, 200) = (0, 227.5, 227.5)
  - (0, 255, 255) → (0, 228, 228)

Example: Mean Shift Algorithm

$$\begin{bmatrix} (255,0,0) & (255,100,0) & (0,200,0) & (0,200,200) \\ (200,0,0) & (0,255,0) & (0,255,255) & (255,255,0) \end{bmatrix}$$ Color Image

- Color bandwidth, hc=60
- For (0, 255, 255)
  - Mean = (0, 255, 255) and (0, 200, 200) = (0, 227.5, 227.5)
  - (0, 255, 255) → (0, 227, 227)

Cluster 1, Red: (255,0,0), (200,0,0)
Cluster 2, Green: (0,255,0), (0,200,0)
Cluster 3, Cyan: (0,255,255), (0,200,200)

$$\begin{bmatrix} (255,0,0) & (255,100,0) & (0,200,0) & (0,200,200) \\ (200,0,0) & (0,255,0) & (0,255,255) & (255,255,0) \end{bmatrix}$$

Example: Mean Shift Algorithm

$$\begin{bmatrix} {\color{red}(255,0,0)} & (200,0,0) & (255,100,0) \\ (0,255,0) & (0,200,0) & (0,255,255) \\ (255,255,0) & (255,255,255) & (0,0,0) \end{bmatrix}$$

Color Image

- Consider color bandwidth, hc = 60 and spatial bandwidth, hs = 1
- Initial Point (1,1): (255,0,0)
  - Within hs, color distance from (200,0,0) at (1,2)

    $= \sqrt{(255-200)^2+(0-0)^2+(0-0)^2} = 55$
  - For (255,100,0) at (1, 3), do not consider as 2>hs
  - For (0,255,0) at (2,1),
    - Color distance = 360.62

Example: Mean Shift Algorithm

$$\begin{bmatrix} (255,0,0) & (200,0,0) & (255,100,0) \\ (0,255,0) & (0,200,0) & (0,255,255) \\ (255,255,0) & (255,255,255) & (0,0,0) \end{bmatrix}$$

Color Image

- Consider color bandwidth, hc = 60 and spatial bandwidth, hs = 1
  - Others are out of spatial bandwidth
  - Color mean= (255,0,0) + (200,0,0) = (227.5, 0, 0)
    - $(255,0,0) \rightarrow (227.5,0,0)$

Example: Mean Shift Algorithm

$$\begin{bmatrix} (255,0,0) & (200,0,0) & (255,100,0) \\ (0,255,0) & (0,200,0) & (0,255,255) \\ (255,255,0) & (255,255,255) & (0,0,0) \end{bmatrix}$$

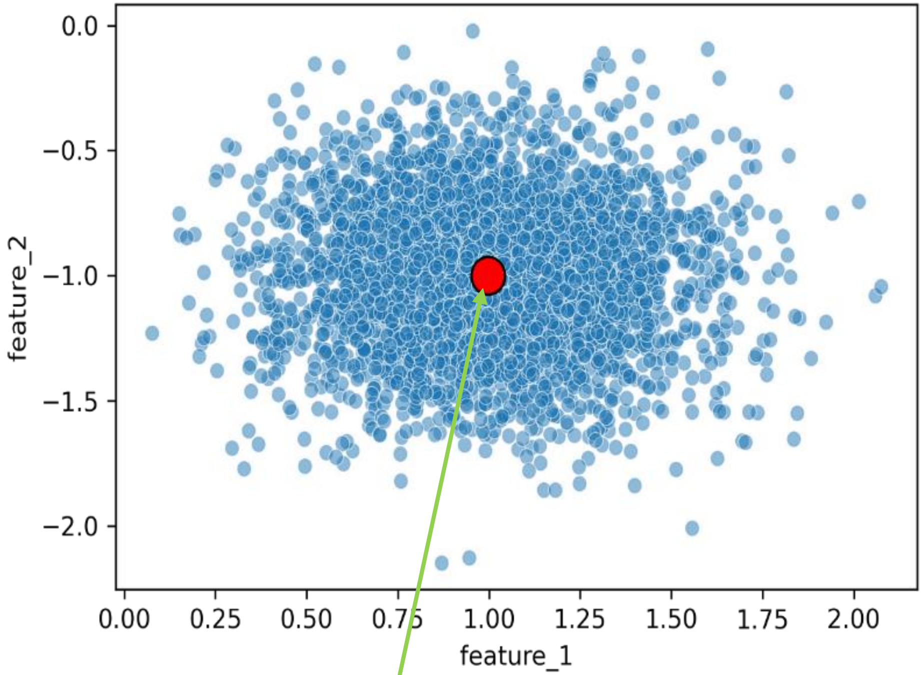
- Consider color bandwidth, hc = 60 and spatial bandwidth, hs = 1
- Initial Point (2,1): (0,255,0)
  - Point within spatial distance at (2,2) is (0,200,0)
  - Color distance = 55
  - Color mean = 227.5
- Initial Point (2,3): (0,255,255)
  - Point within hs is (255,100,0)
  - Color distance = 261.16 (is > hc)
  - Therefore Color mean is (0,255,255)

Example: Mean Shift Algorithm

- Repeat till cluster centers converge
- Cluster 1: (255,0,0) and (200,0,0)
- Cluster 2:$(0,255,0)$ and $(200,0,0)$
- Cluster 3: (0,255,255)

$$\begin{bmatrix} (255,0,0) & (200,0,0) & (255,100,0) \\ (0,255,0) & (0,200,0) & (0,255,255) \\ (255,255,0) & (255,255,255) & (0,0,0) \end{bmatrix}$$

Mean Shift Algorithm

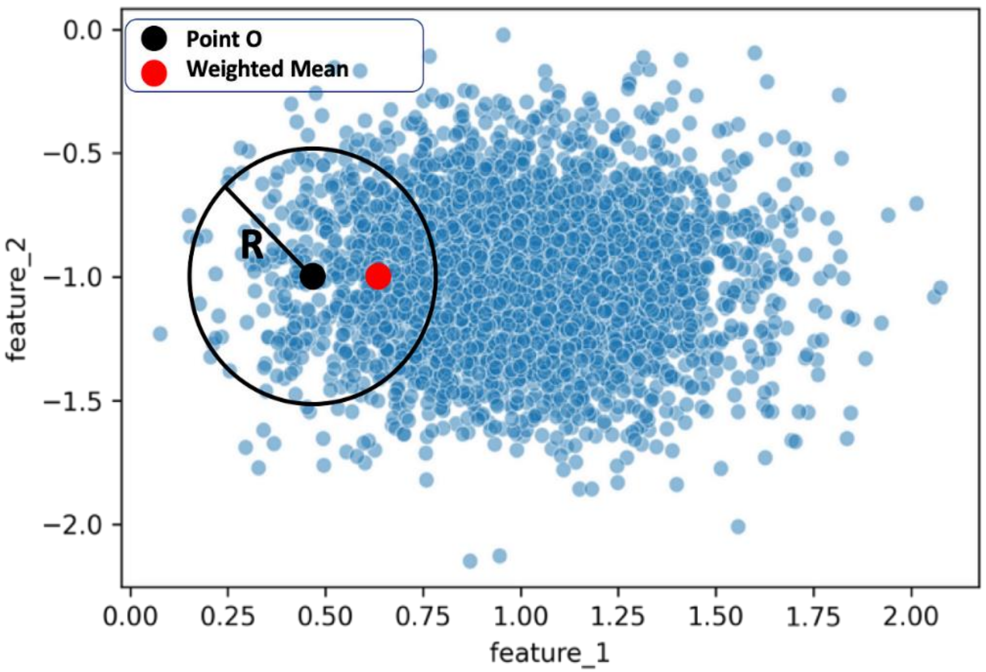


Mean point, M_A of all samples

$$M_A = \frac{1}{n}\sum_{i=1}^{n} x_i$$

Each point is given equal weight

Mean Shift Algorithm



Samples inside the circle are considered and outside the circle are ignored

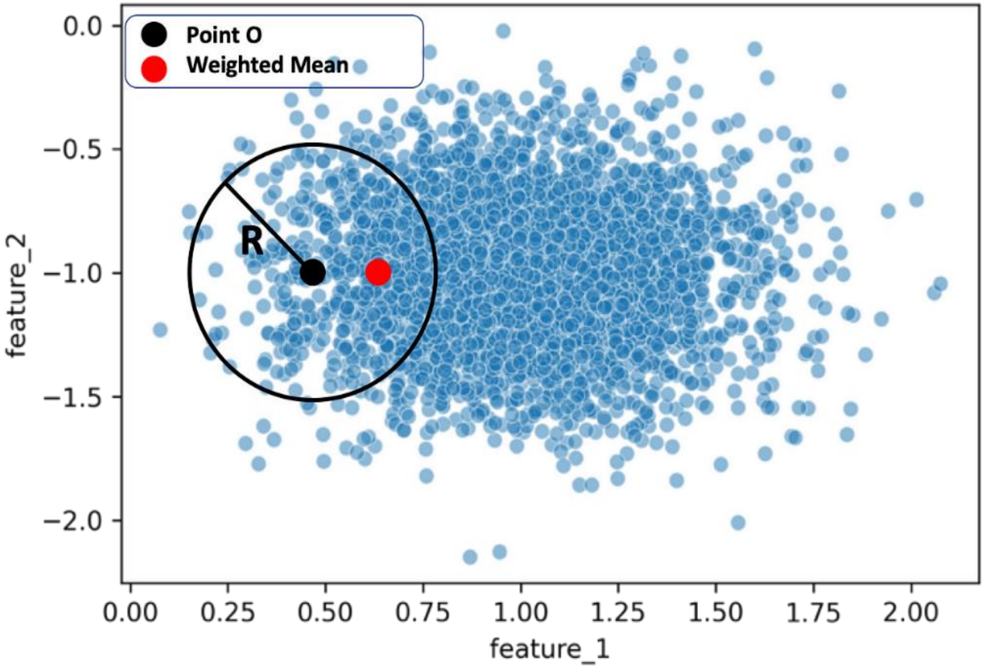
Weighted mean function

$$M_W = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i} \qquad w(d) = \begin{cases} 1, & \text{if } d \le R \\ 0, & \text{if } d > R \end{cases}$$

where

- d is the distance between any data point to the current mean
- R is the radius of the circle at initial point or mean in the previous iteration

Mean Shift Algorithm



Gaussian weight function

$$M_W = \frac{\sum_{i=1}^{n} w_i x_i}{\sum_{i=1}^{n} w_i} \qquad w(d) = e^{-\frac{d}{2\sigma^2}}$$

where

- d is the distance between the center point to current mean
- sigma is used to adjust how fast the weight decreases with the increase of d
- Closer is the pixel to centroid more is the weighted mean

Mean Shift Algorithm

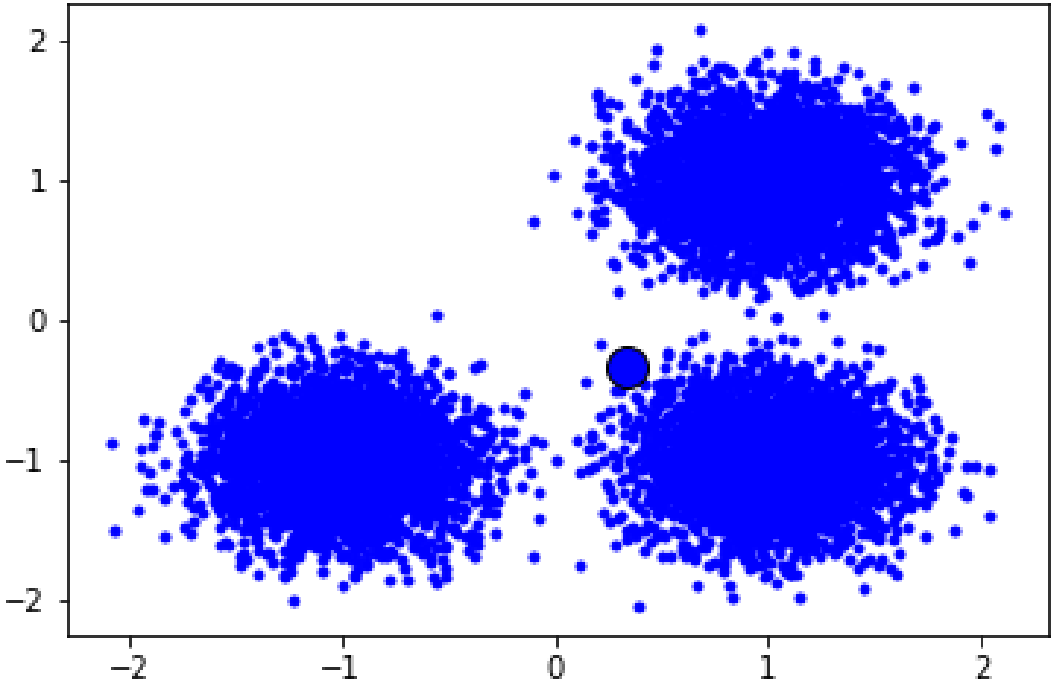
1. Kernel Density Estimation (KDE)
    - Define a kernel window (a circular or Gaussian window) around a preselected centroid
    - Calculate the mean of all the pixels within the window
    - Change pixel value to new mean
    - This process is repeated for each pixel until convergence of mean
2. Mode Seeking
    - Each pixel is associated with a nearby peak (or mode) in the data density
    - Pixels that converge to the same mode are grouped into the same cluster
    - Cluster represent segment of the image

Mean Shift Algorithm

3. Bandwidth Parameter:
    - Bandwidth is a crucial parameter that controls the size of the kernel window
    - It affects the scale of the clusters
    - Small bandwidth: More clusters, finer segmentation
    - Large bandwidth: Fewer clusters, coarser segmentation.
    - Selecting the optimal bandwidth is important for the quality of segmentation

Mean Shift Algorithm

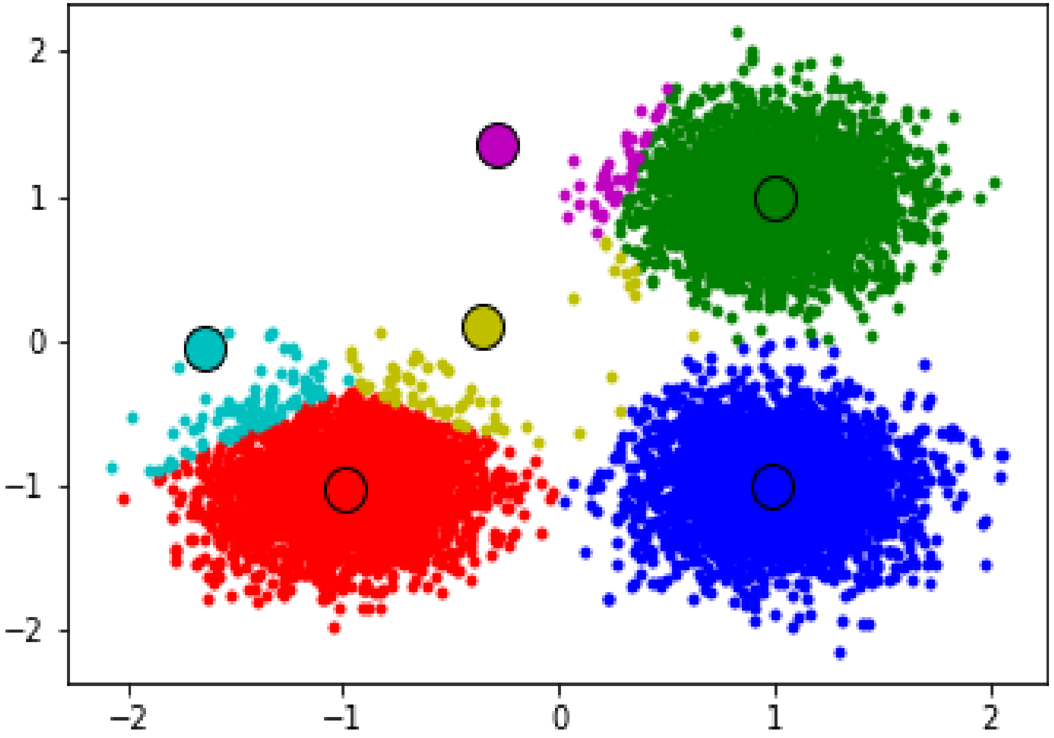
- Larger the widow size is, the closer the local mean point is to the global mean
- Large local region can ignore the local structure of the dataset



Mean shift with large bandwidth (window size)

Mean Shift Algorithm

- Mean value is calculated for a small local area
- If small bandwidth is used, several noisy clusters can appear



Mean shift with small bandwidth

Mean Shift Algorithm

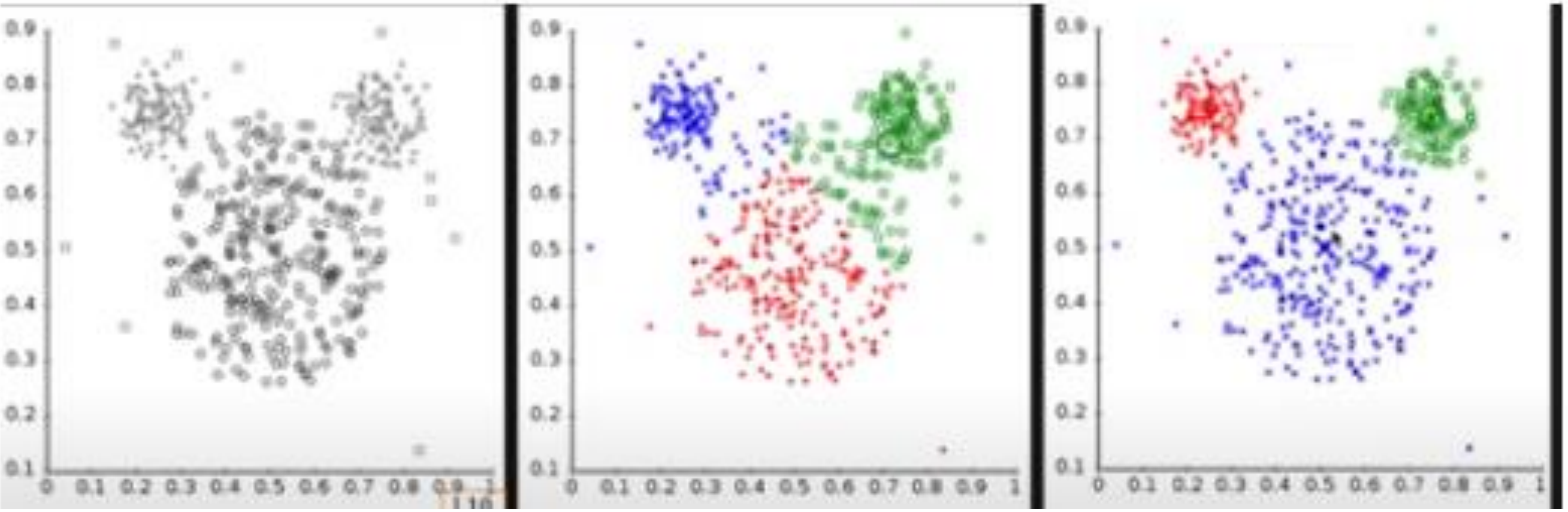
Pros:

- Finds number of modes depending on the data values
- Robust to noise or outliers: By focusing on regions of high density, it can avoid the influence of noise in the image
- Non parametric as it does not assume any prior shape like spherical, elliptical, etc. on data clusters

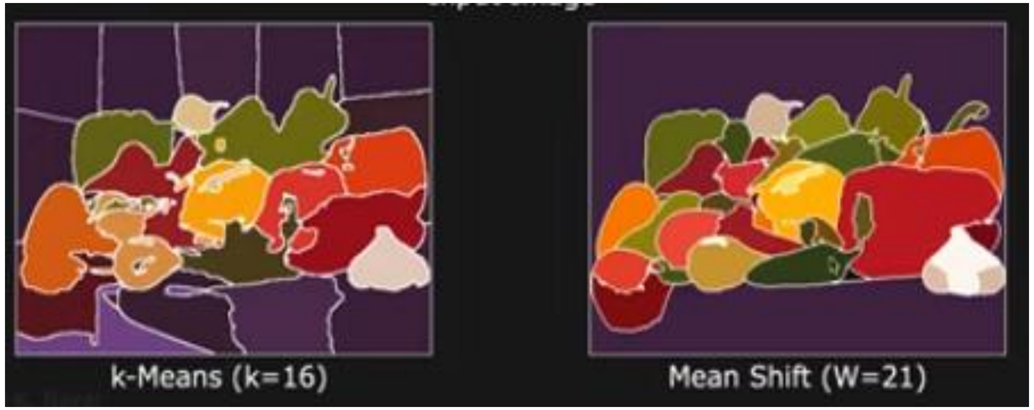
Cons:

- Clustering depends on the choice of window size (bandwidth) which can lead to under or over segmentation
- Computationally more expensive than K-means
- Algorithm can be slow, especially for high-resolution images because kernel moves for every pixel until convergence
- Can identify noisy pixel as clusters
- Finds arbitrary number of clusters

Mean Shift Algorithm



Original data | k-means (k=3) | Mean shift



K needs to be specified

Window size needs to be specified

Applications of Image Segmentation

1. Object Detection and Recognition:
    - Identify and segment distinct objects by grouping similar pixels together
2. Medical Imaging:
    - Segment medical images such as MRI, CT scans, or X-rays to highlight regions of interest like tumors, organs, or abnormalities
3. Texture Segmentation:
    - Segment images based on texture or color, helping in tasks like landscape segmentation or separating regions with different surface textures
4. Video Tracking:
    - Track objects in a video by clustering regions of similar features across frames