19ECCN1702 - Machine Learning

UNIT 4-UNSUPERVISED LEARNING

Unit I INTRODUCTION **9 Hours**

Introduction to Machine Learning - Types of Machine Learning systems - Challenges in Machine Learning - Overfitting and Under fitting - Testing and Validating the model - Bias and Variance

Unit II MACHINE LEARNING FRAMEWORK **9 Hours**

Problem Formulation - Get the data - analyze and visualize the data - Prepare the data for ML algorithms - sample complexity - Hypothesis space - Model evaluation and Improvement: Cross validation - Grid search - Evaluation Metrics - Kernel functions

Unit III SUPERVISED LEARNING **9 Hours**

Linear and Logistic Regression – Eigen Values and Eigen vectors - Naïve Bayes Classifier: Maximum Likelihood, Minimum Description Length – Gradient Descent - Decision Trees - Ensembles of Decision Trees - Support Vector Machine(SVM)

Unit IV UNSUPERVISED LEARNING 9 Hours

Clustering: k-Means clustering- Agglomerative Clustering - DBSCAN- Gaussian Mixtures- precision and recall - Collaborative filtering and Content Filtering

Unit V NEURAL NETWORK AND DEEP LEARNING 9 Hours

Biological Neuron - Logical computation with Neuron - Perceptron - Sigmoid and softmax functions - Multi Layer Perceptron(MLP) with Back propagation - Regression MLPs - Classification MLPs - Fine Tuning NN models - Convolutional Neural Network: Architecture of Visual cortex - Convolutional Layers - Stacking Multiple Feature Maps- CNN architectures

Course Outcomes	**Cognitive Level**
At the end of this course, students will be able to:	
CO1:Describe the types and challenges in Machine learning for exploring the machine learning concepts	Understand
CO2:Illustrate the machine learning framework for implementation of machine learning projects	Apply
CO3:Interpret the supervised learning techniques for classification	Apply
CO4:Demonstrate the un-supervised learning methods for clustering and classification	Apply
CO5:Construct the Neural network and deep learning models for classification	Apply

Text Book(s):

T1. AurélienGéron," Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow", Second edition, O'Reilly Media, Inc,2019

T2. Andreas C. Müller and Sarah Guido, "Introduction to Machine Learning with Python A Guide for Data Scientists", First Edition,O'Reilly,2017

Reference Book(s):

R1. Ethem Alpaydin, "Introduction to Machine Learning 3e (Adaptive Computation and Machine Learning Series)", 3rd Edition, MIT Press, 2014

R2. Jason Bell, "Machine learning - Hands on for Developers and Technical Professionals",1st Edition, Wiley, 2014

R3. Peter Flach, "Machine Learning: The Art and Science of Algorithms that Make Sense of Data", 1st Edition, Cambridge University Press, 2012.

Web References:

1. https://www.kaggle.com/kanncaa1/machine-learning-tutorial-for-beginners
2. https://nptel.ac.in/courses/106/106/106106139/
3. https://archive.ics.uci.edu/ml/datasets.php

Topics

- K means clustering
- Agglomerative clustering
- DBSCAN
- Gaussian Mixtures
- Precision and Recall
- Collaborative filtering and Content filtering

Unsupervised Learning

Two kinds of unsupervised learning are Transformations of the dataset and clustering.

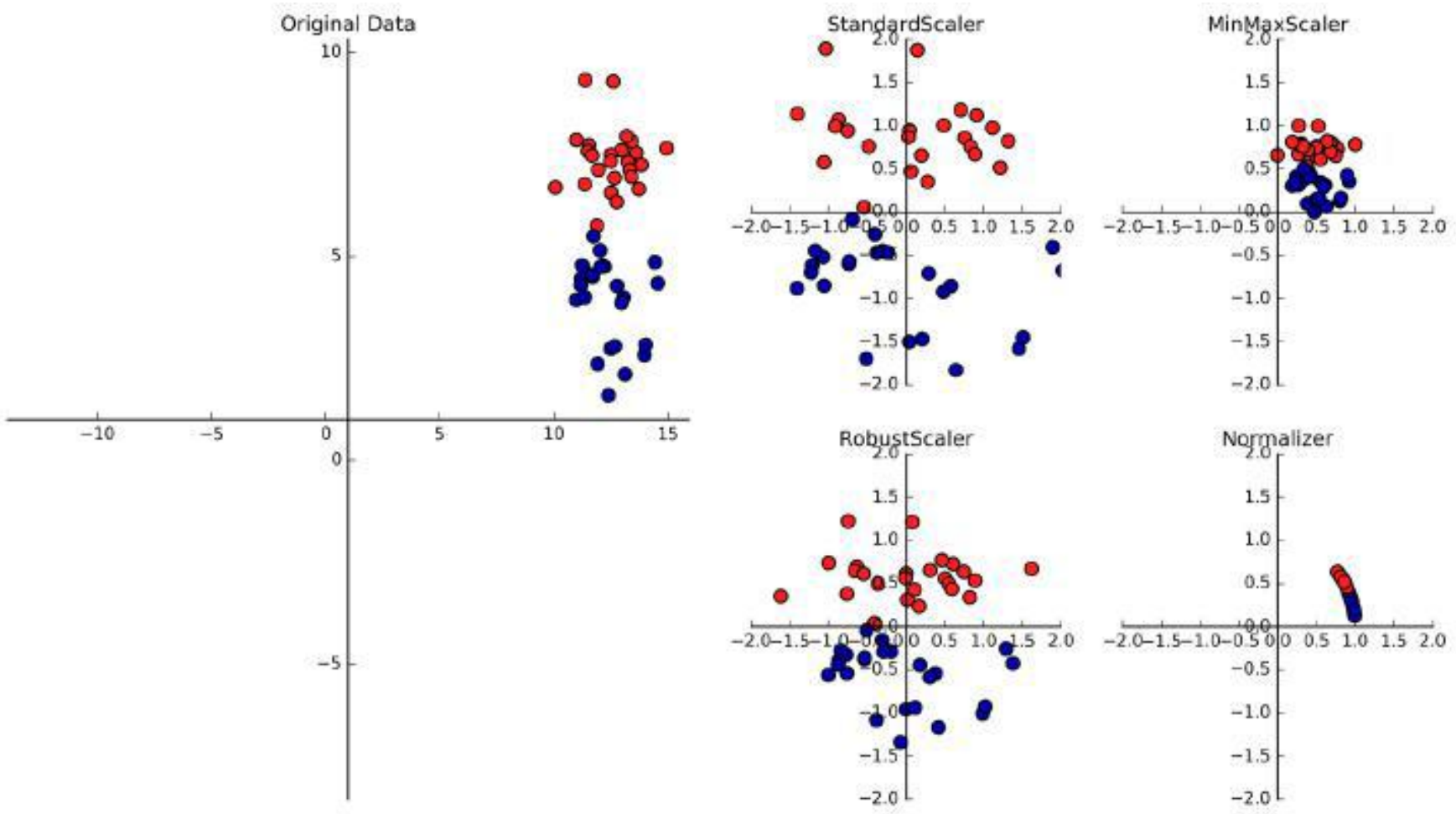
Unsupervised transformations of a dataset are algorithms that create a new representation of the data which might be easier for humans or other machine learning algorithms to understand compared to the original representation of the data.

A common application of unsupervised transformations is **dimensionality reduction,** which takes a high-dimensional representation of the data, consisting of many features, and finds a new way to represent this data that summarizes the essential characteristics with fewer features.

Unsupervised Learning

- *Clustering algorithms*, on the other hand, partition data into distinct groups of similar items. Eg: Google Photos
- A major challenge in unsupervised learning is evaluating whether the algorithm learned something useful. (unlabeled data)
- As a consequence, unsupervised algorithms are used often in an exploratory setting, when a data scientist wants to understand the data better, rather than as part of a larger automatic system.

Preprocessing and Scaling



K – means clustering

Types of scales are,

- Standard Scale - ensures that for each feature the mean is 0 and the variance is 1, bringing all features to the same magnitude.
- Robust Scaler uses the median and quartiles,1 instead of mean and variance.
- The Min Max Scaler, on the other hand, shifts the data such that all features are exactly between 0 and 1.
- the Normalizer does a very different kind of rescaling. It scales each data point such that the feature vector has a Euclidean length of 1.

Principal Component Analysis (PCA)

Principal component analysis is a method that rotates the dataset in a way such that the rotated features are statistically uncorrelated.

This rotation is often followed by selecting only a subset of the new features, according to how important they are for explaining the data.

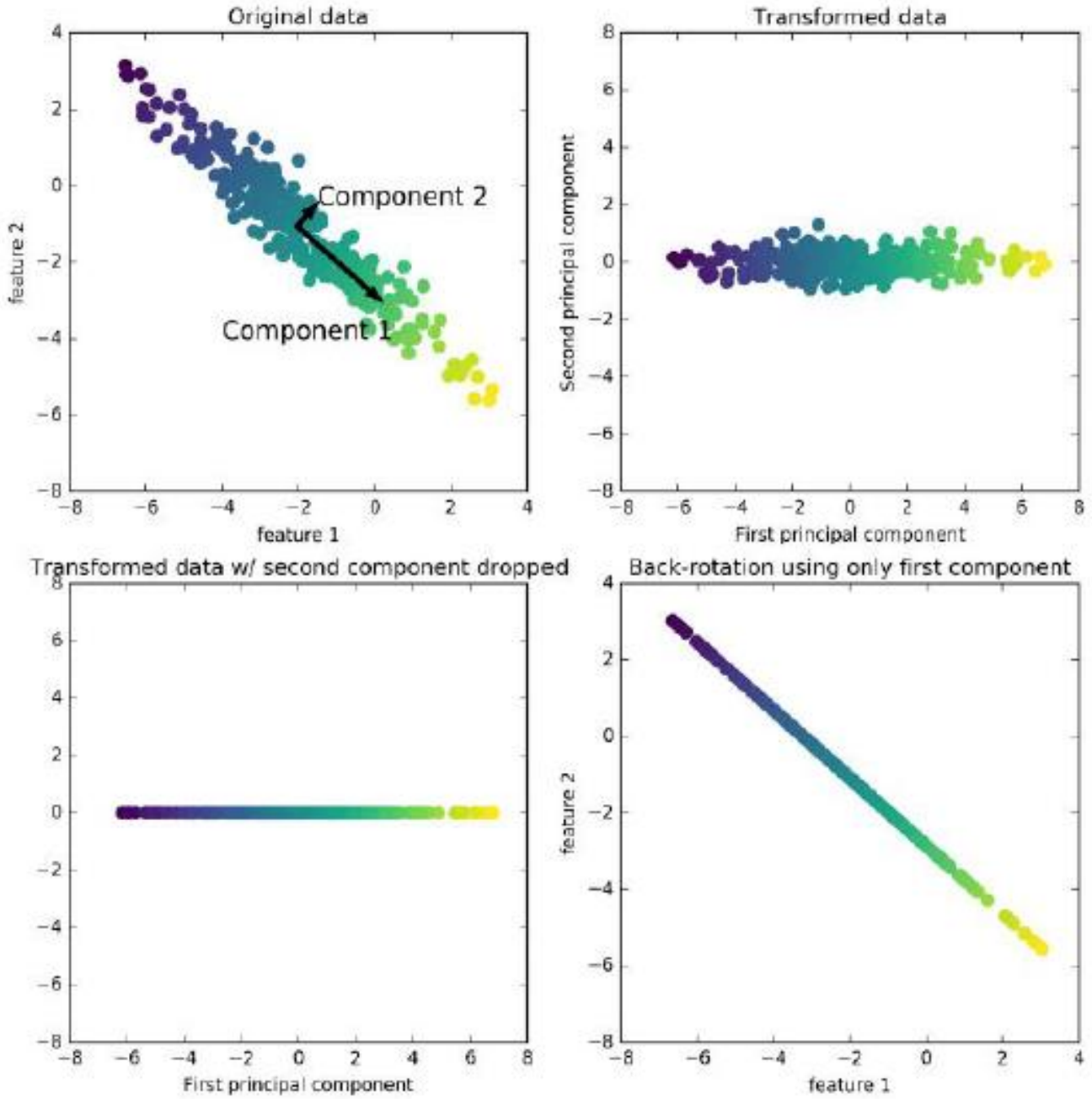


Figure 3-3. Transformation of data with PCA

Clustering

Clustering is the task of partitioning the dataset into groups, called clusters.

The goal is to split up the data in such a way that points within a single cluster are very similar and points in different clusters are different.

k-Means Clustering

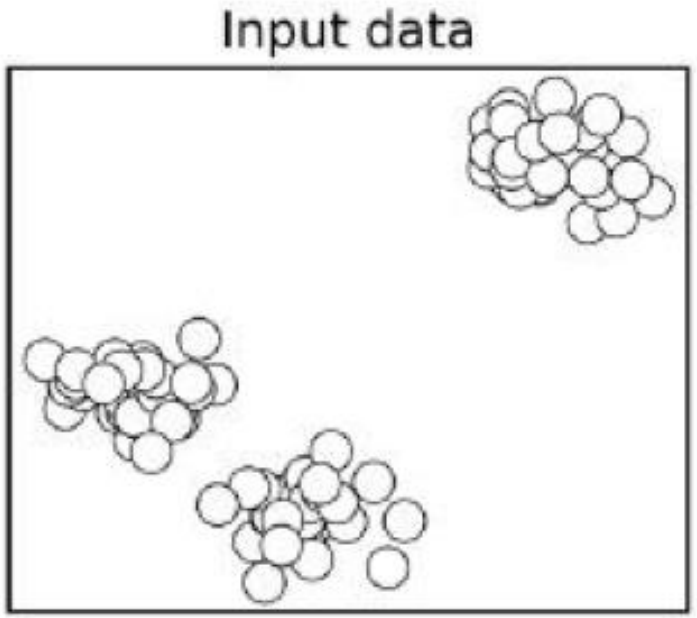
k-Means Clustering

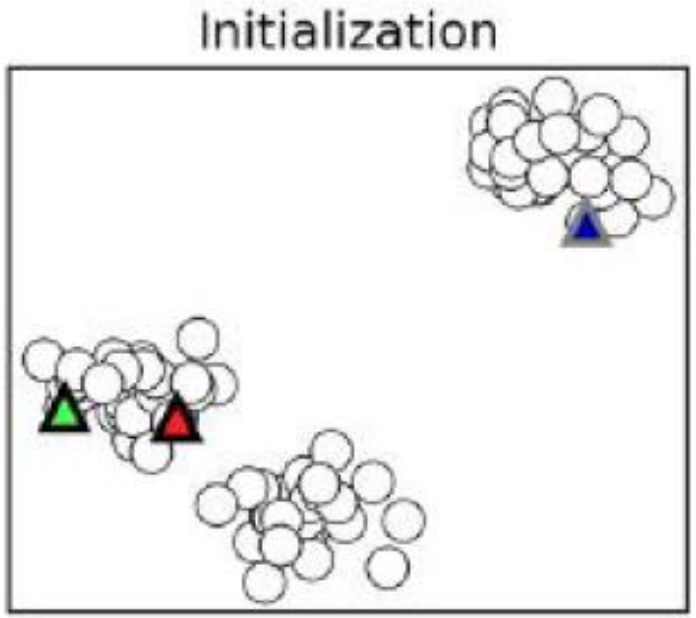
It tries to find cluster centers that are representative of certain regions of the data. The algorithm alternates between two steps:

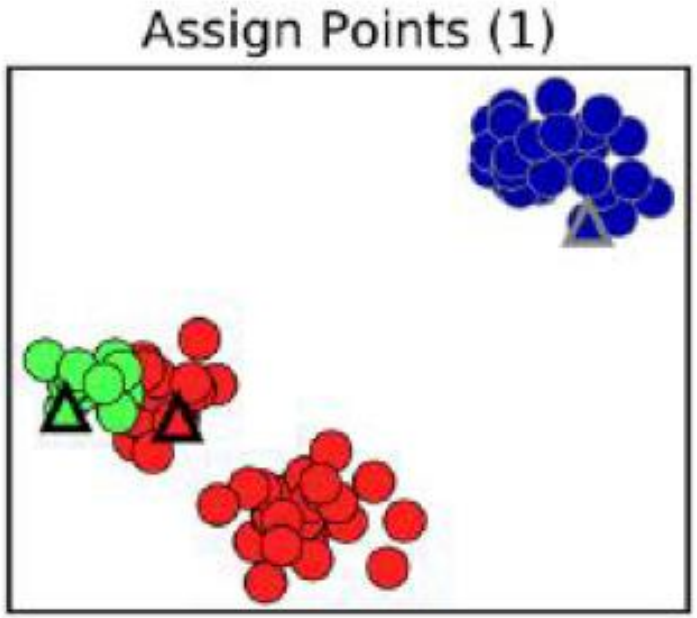
- assigning each data point to the closest cluster center, and then
- setting each cluster center as the mean of the data points that are assigned to it.

The algorithm is finished when the assignment of instances to clusters no longer changes.

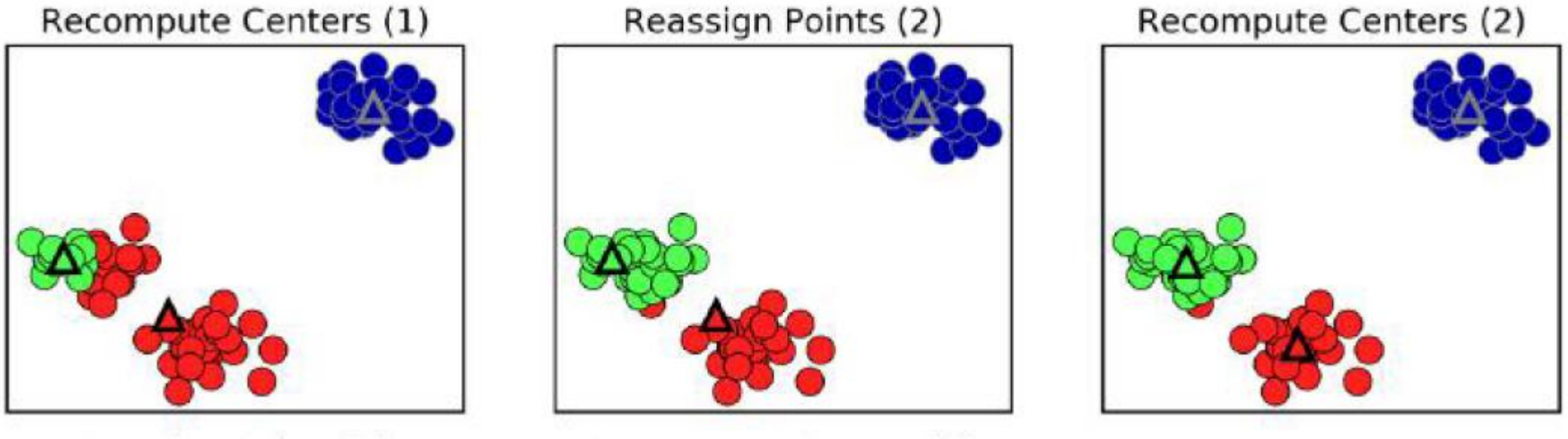
k-Means Clustering





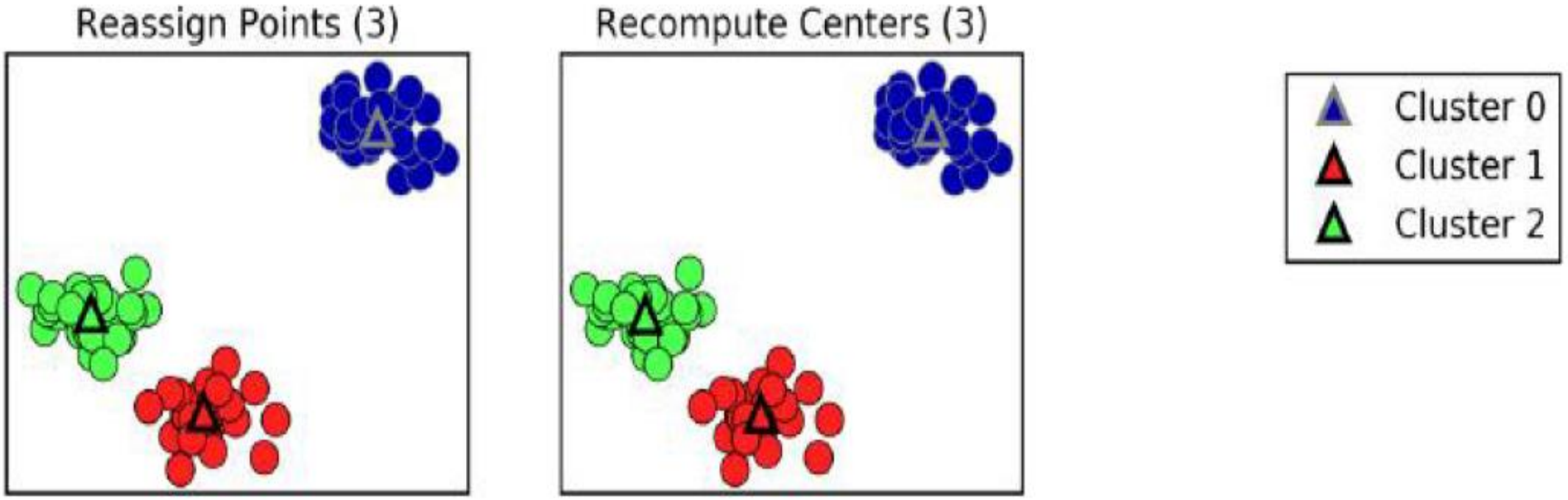


k-Means Clustering

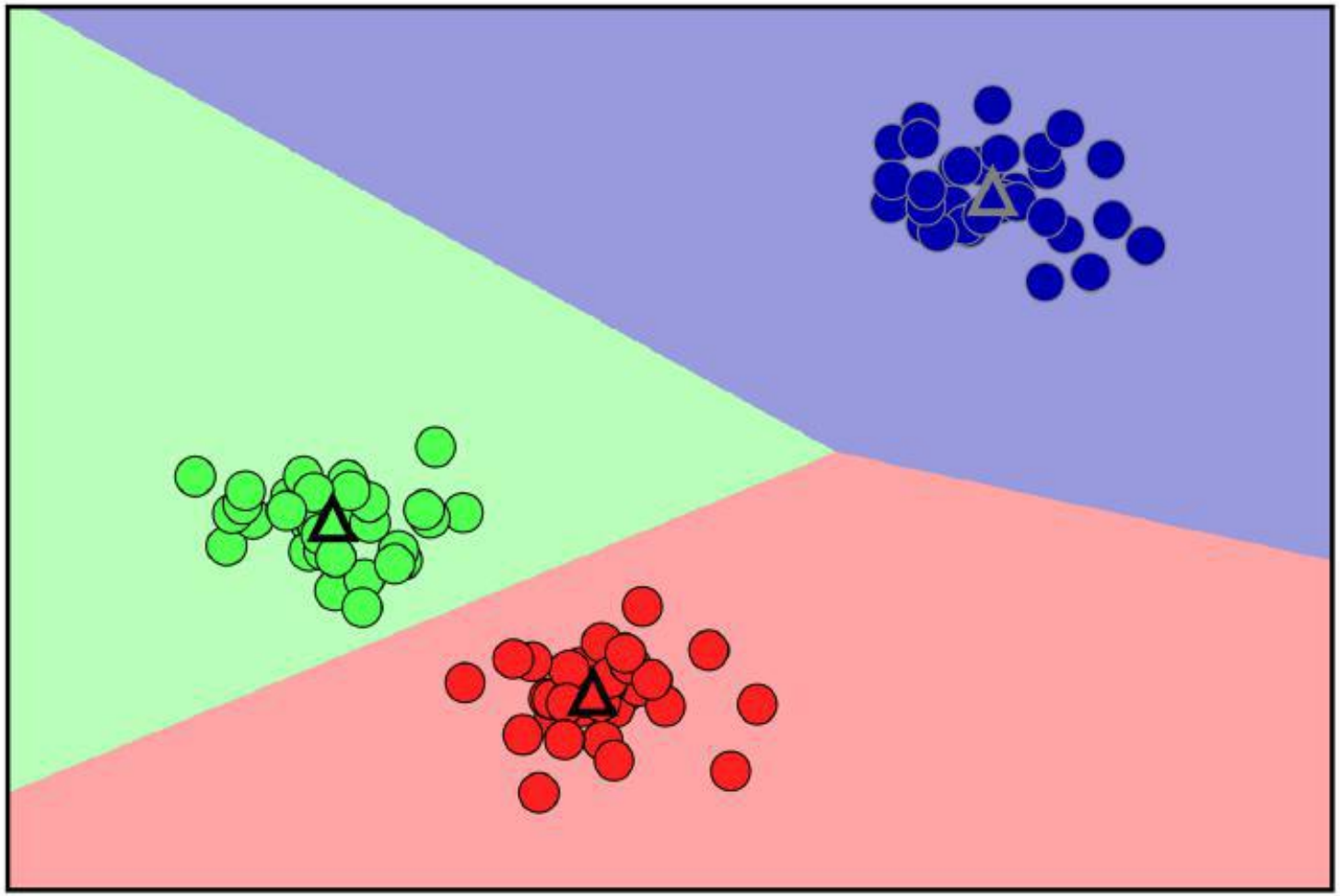


The cluster centers are updated to be the mean of the assigned points

k-Means Clustering



Cluster Boundaries



k-means with scikit-learn

In[49]:

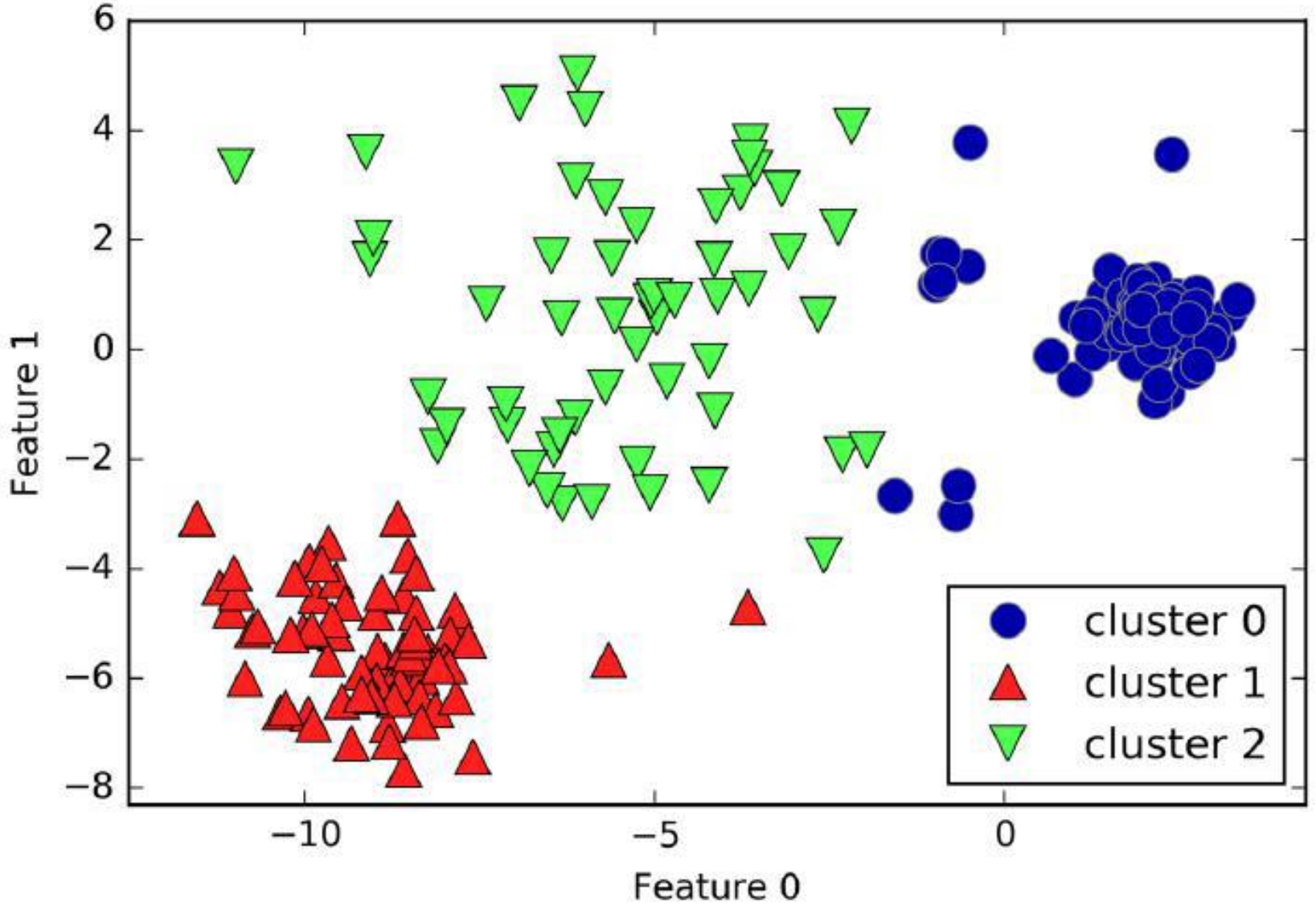
```
from sklearn.datasets import make_blobs
from sklearn.cluster import KMeans

# generate synthetic two-dimensional data
X, y = make_blobs(random_state=1)

# build the clustering model
kmeans = KMeans(n_clusters=3)
kmeans.fit(X)
```

Failure cases of k-means

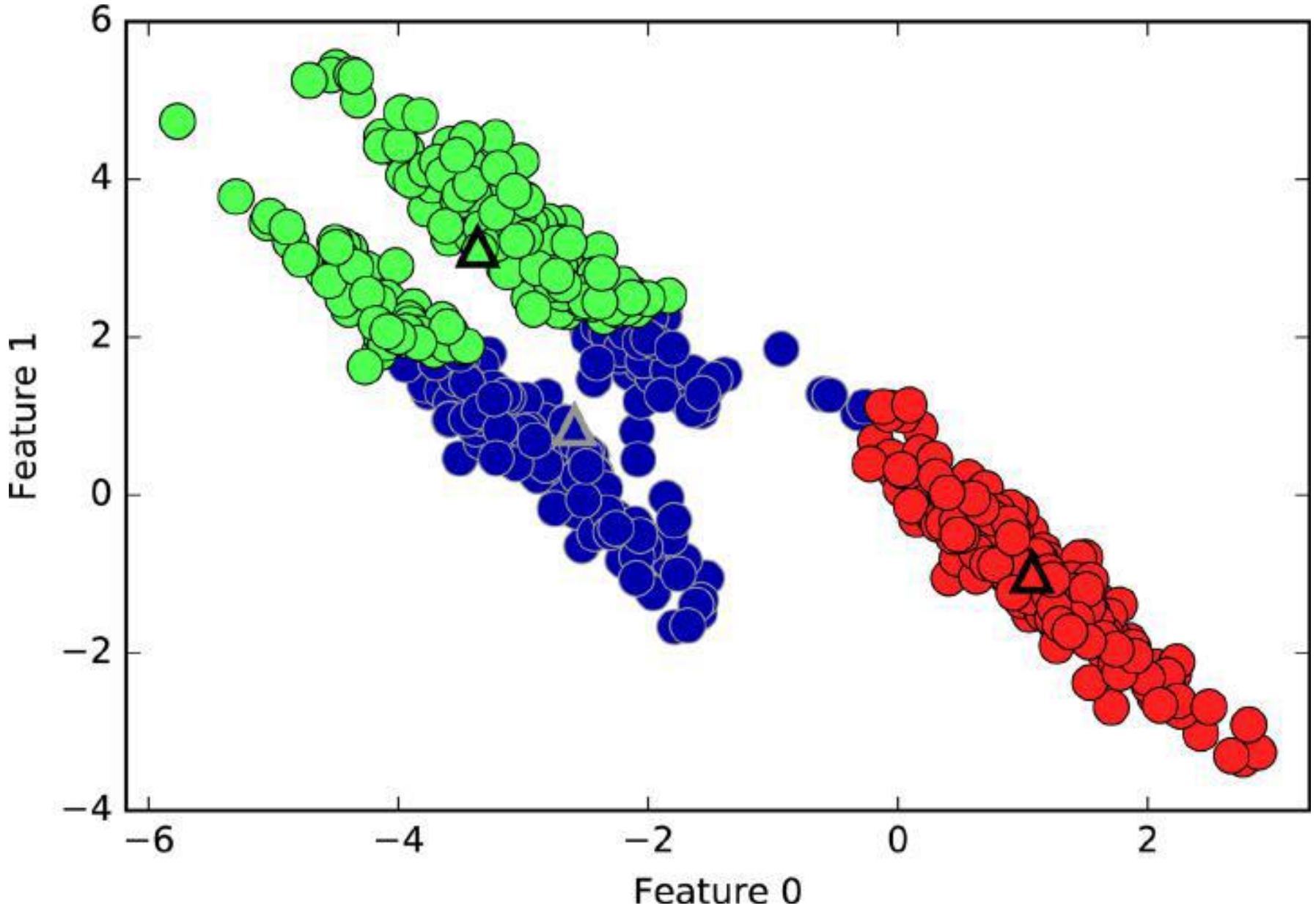
Each cluster is defined solely by its center, which means that each cluster is a convex shape.

As a result of this, *k*-means can only capture relatively simple shapes. *k*-means also assumes that all clusters have the same “diameter” in some sense;

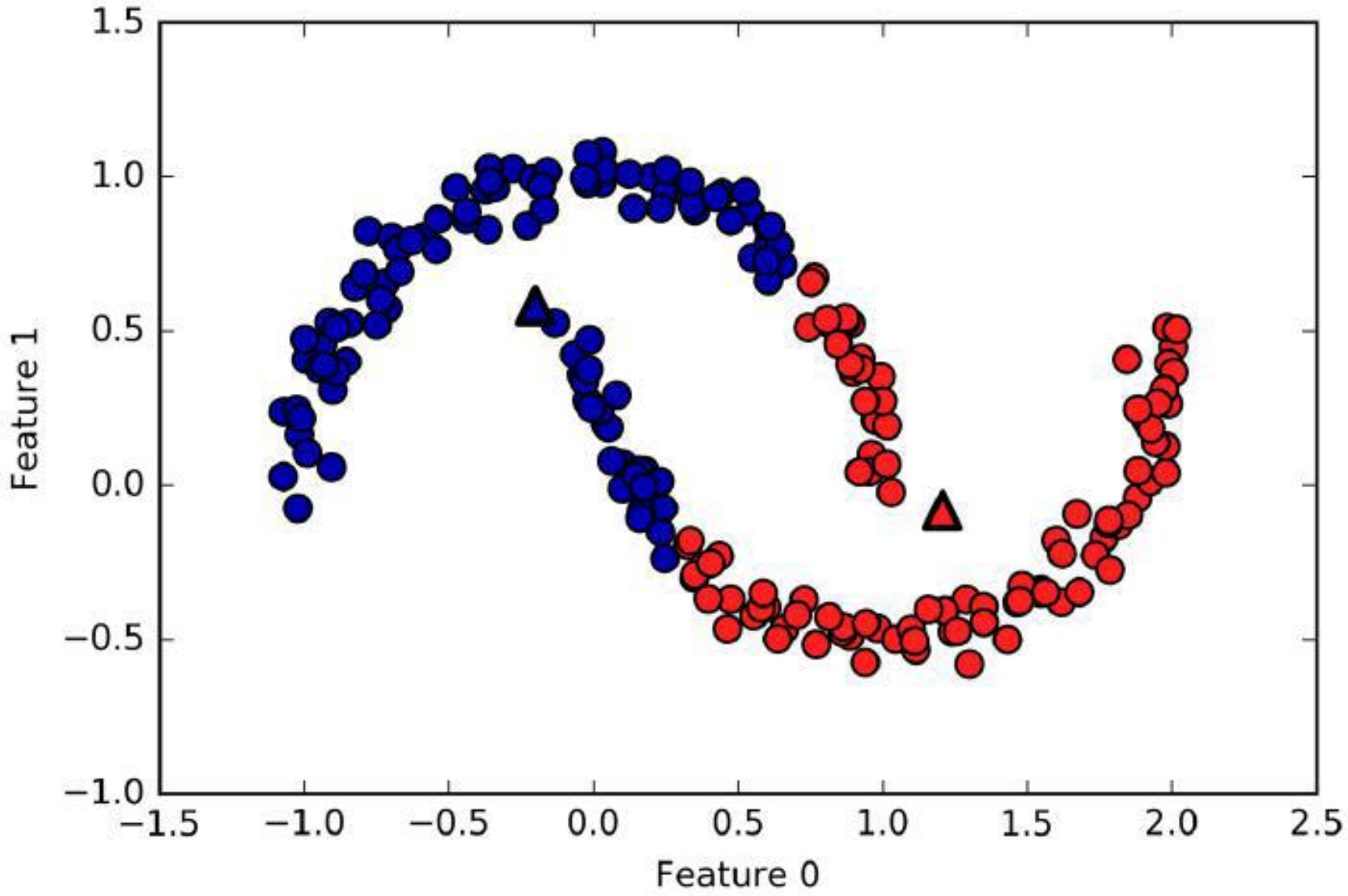
it always draws the boundary between clusters to be exactly in the middle between the cluster centers.



Cluster assignments found by k-means when clusters have different densities

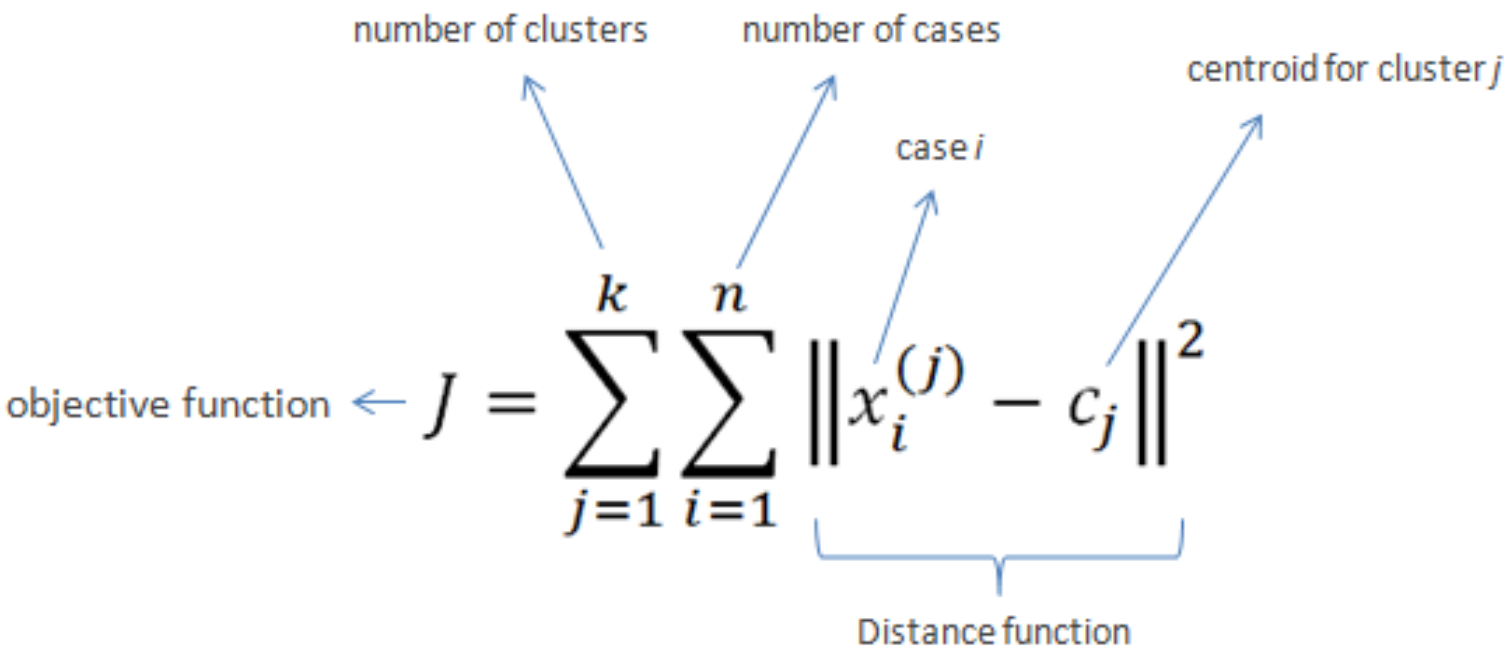


k-means fails to identify non spherical clusters



k-means fails to identify clusters with complex shapes

Cost function



Animations

http://shabal.in/visuals/kmeans/5.html

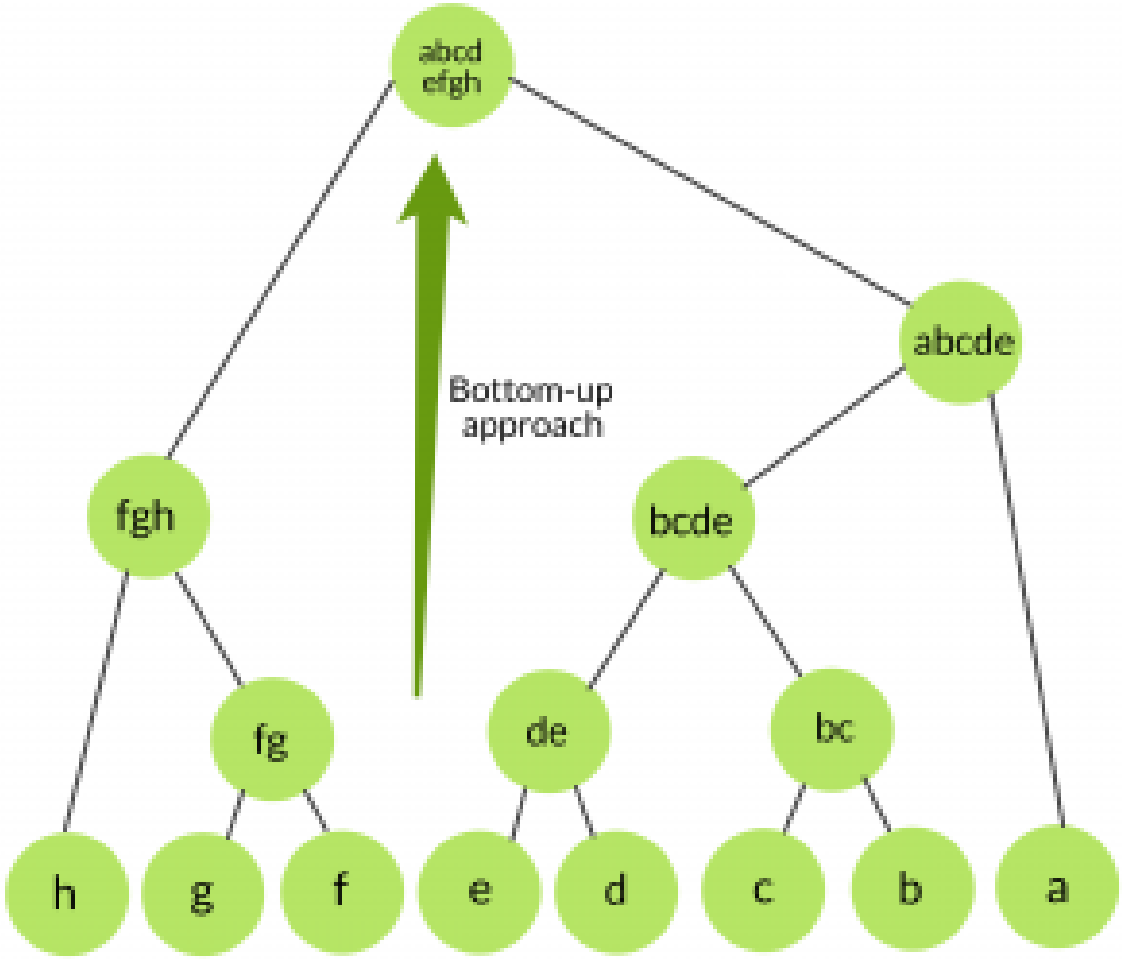
https://www.naftaliharris.com/blog/visualizing-k-means-clustering/

Agglomerative Clustering

- *Agglomerative Clustering is also known as bottom-up approach or hierarchical agglomerative clustering (HAC)*
- *Agglomerative clustering refers to a collection of clustering algorithms that all build* upon the same principles
- The algorithm starts by declaring each point its own cluster, and then merges the two most similar clusters until some stopping criterion is satisfied

Agglomerative Clustering

- This clustering algorithm does not require us to prespecify the number of clusters.
- Bottom-up algorithms treat each data as a singleton cluster at the outset and then successively agglomerates pairs of clusters until all clusters have been merged into a single cluster that contains all data.



Hierarchical agglomerative clustering

Merging of Clusters

While merging two clusters we check the distance between two every pair of clusters and merge the pair with least distance/most similarity.

Some of the methods are:

1. Min Distance: Find minimum distance between any two points of the cluster.

2. Max Distance: Find maximum distance between any two points of the cluster.

3. Group Average: Find average of distance between every two points of the clusters.
4. Ward's Method: Similarity of two clusters is based on the increase in squared error when two clusters are merged

Clustering Choices

ward

The default choice, ward picks the two clusters to merge such that the variance within all clusters increases the least. This often leads to clusters that are relatively equally sized.

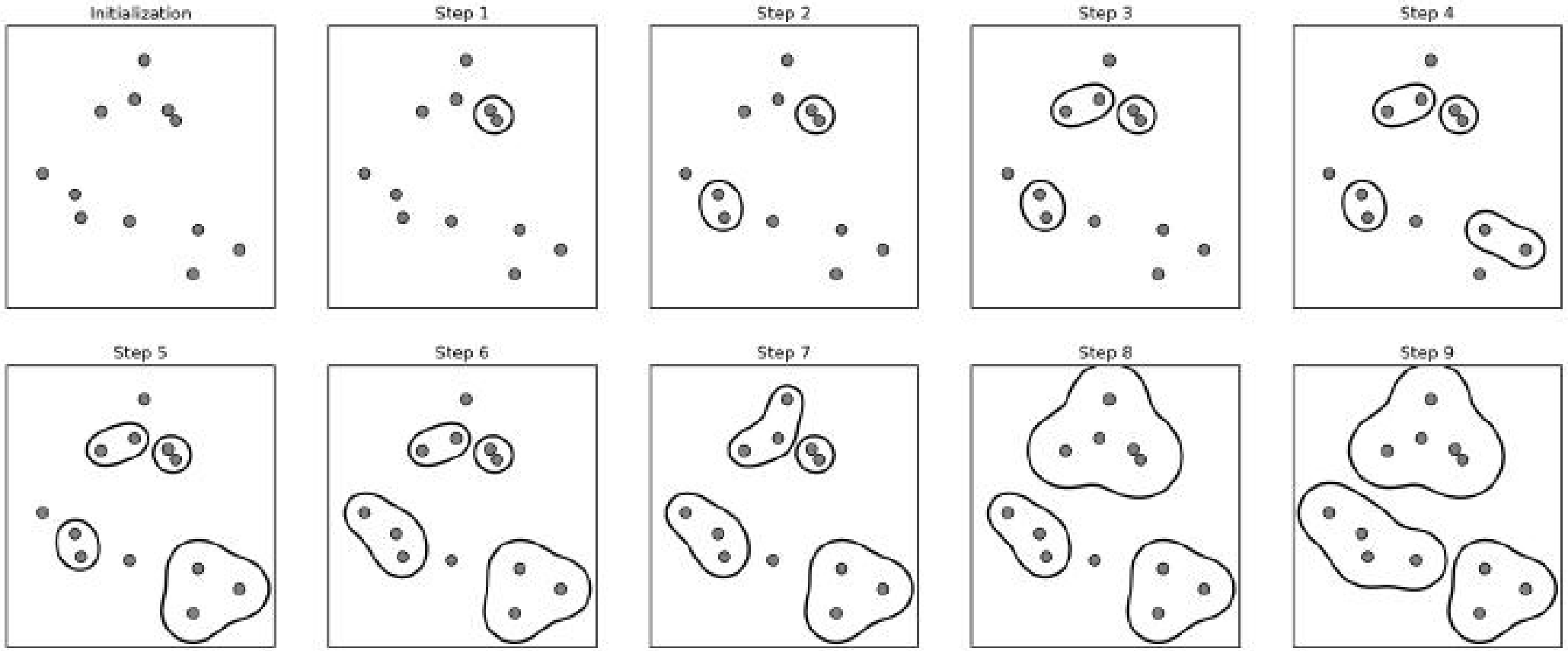
average

average linkage merges the two clusters that have the smallest average distance between all their points.

complete

complete linkage (also known as maximum linkage) merges the two clusters that have the smallest maximum distance between their points.

Agglomerative clustering iteratively joins the two closest clusters

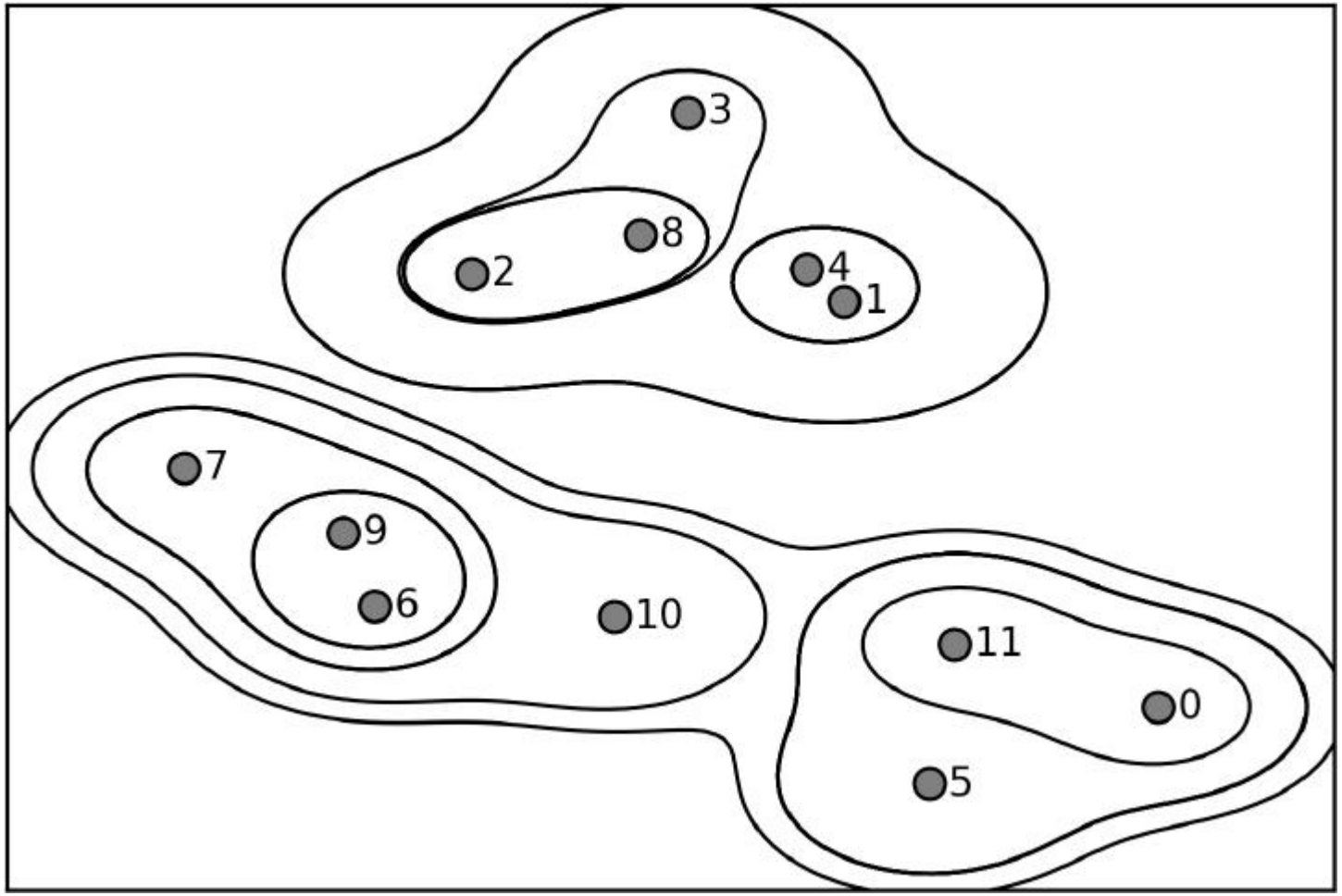


looking for three clusters:

fit_predict

Agglomerative Clustering has no predict method. To build the model and get the cluster memberships on the training set, use the fit_predict method instead.

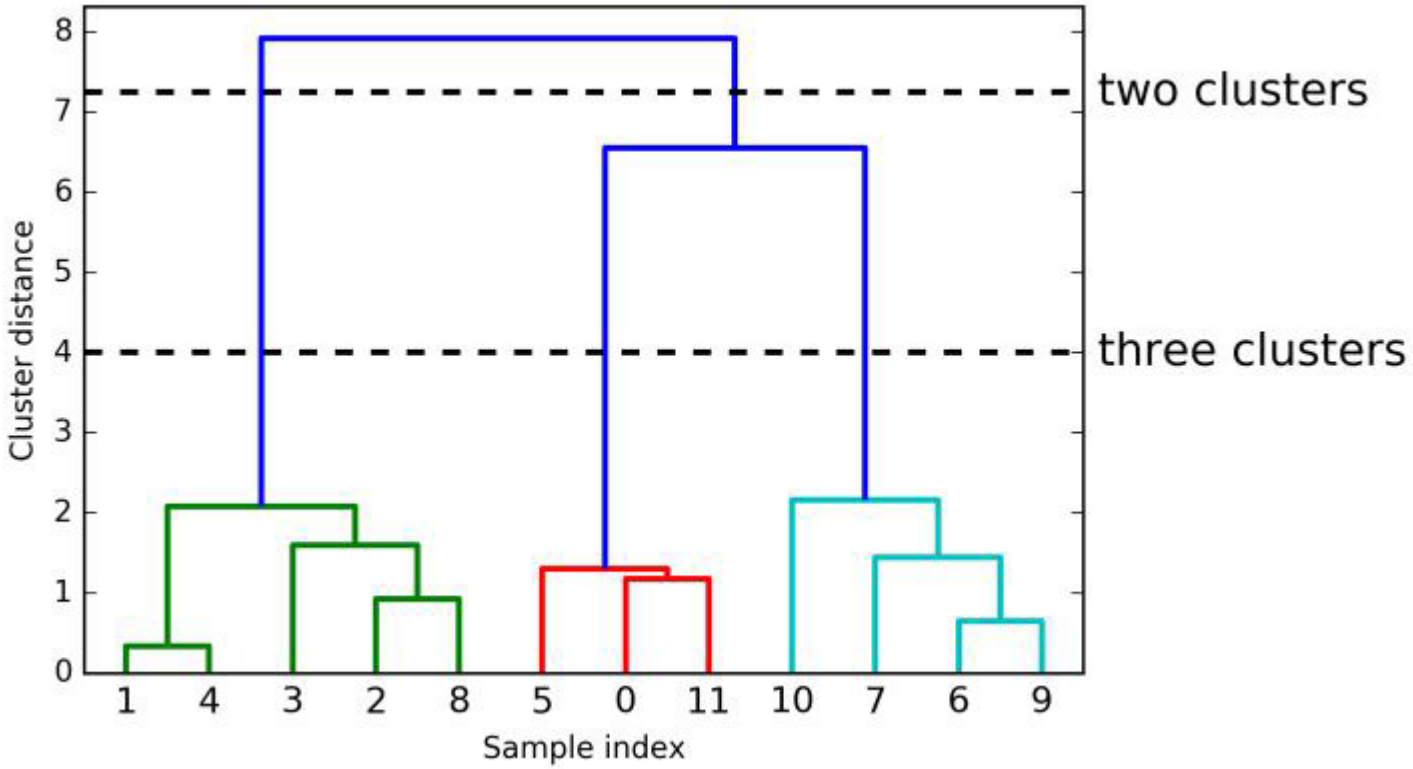
Hierarchical clustering and dendrograms



Hierarchical cluster assignment (shown as lines) generated with agglomerative clustering, with numbered data points

Dendrogram

Tool to visualize hierarchical clustering, called a *dendrogram, that can handle multidimensional* datasets.

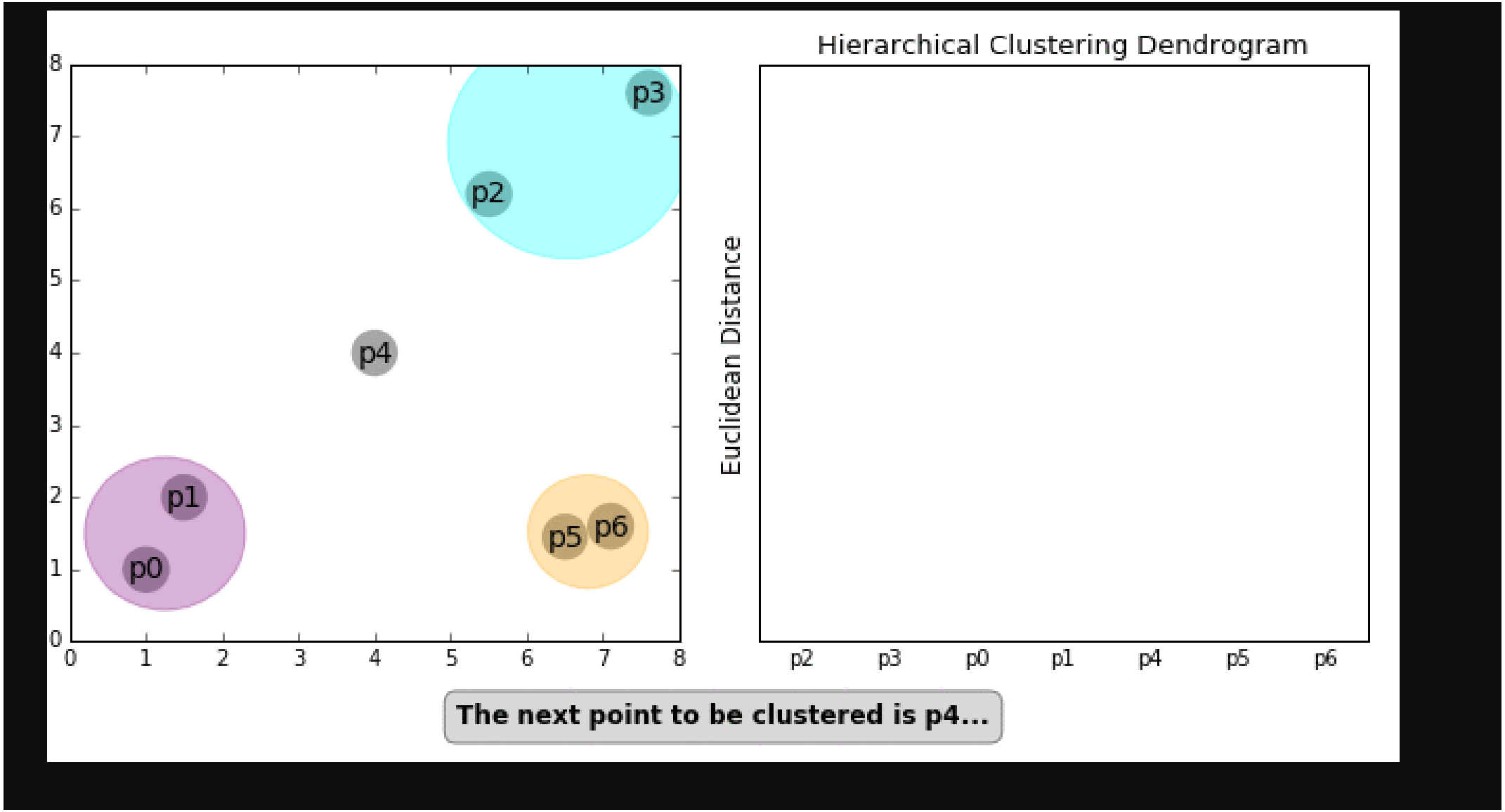


Animations

https://dashee87.github.io/images/hierarch.gif

https://dashee87.github.io/images/hierarch_1.gif

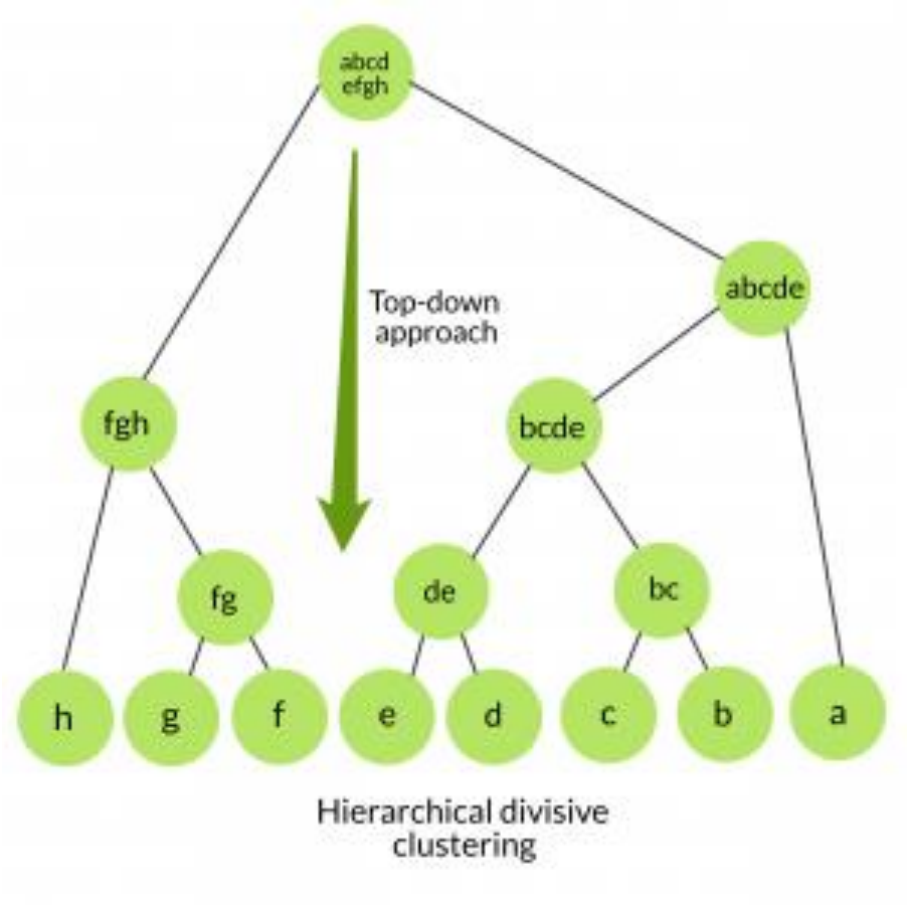
https://dashee87.github.io/data%20science/general/Clustering-with-Scikit-with-GIFs/


Hierarchical Clustering Dendrogram
Euclidean Distance
p0
p1
p2
p3
p4
p5
p6
The next point to be clustered is p4...

Divisive clustering

- Also known as a top-down approach.
- This algorithm also does not require to prespecify the number of clusters.
- Top-down clustering requires a method for splitting a cluster that contains the whole data and proceeds by splitting clusters recursively until individual data have been split into singleton clusters.

Divisive clustering



Hierarchical divisive clustering

DB SCAN

- **Density-based spatial clustering of application**
- Data mining
- Density based spatial clustering of applications with noise
- It can capture clusters of complex shapes, and it can identify points that are not part of any cluster. DBSCAN is somewhat slower than agglomerative clustering and *k*-means, but still scales to relatively large datasets.
- DBSCAN works by identifying points that are in “crowded” regions of the feature space, where many data points are close together.

DB SCAN

- ***eps**: specifies how close points should be to each other to be considered a part of a cluster. It means that if the distance between two points is lower or equal to this value (eps), these points are considered neighbors.*

- ***minPoints**: the minimum number of points to form a dense region. For example, if we set the minPoints parameter as 5, then we need at least 5 points to form a dense region.*

DB SCAN

- These regions are referred to as dense regions in feature space.
- The idea behind DBSCAN is that clusters form dense regions of data, separated by regions that are relatively empty.
- Points that are within a dense region are called core samples.
- If there are at least min_samples many data points within a distance of eps to a given data point, that data point is classified as a core sample.
- Core samples that are closer to each other than the distance eps are put into the same cluster by DBSCAN
- If there are less than min_samples points within distance eps of the starting point, this point is labeled as noise, meaning that it doesn't belong to any cluster.

DBSCAN

DBSCAN algorithm requires two parameters:

- **eps** : It defines the neighborhood around a data point i.e. if the distance between two points is lower or equal to ‘eps’ then they are considered neighbors.
- If the eps value is chosen too small then large part of the data will be considered as outliers. If it is chosen very large then the clusters will merge and the majority of the data points will be in the same clusters.
- One way to find the eps value is based on the ***k-distance graph***.

DB SCAN

- **MinPts**: Minimum number of neighbors (data points) within eps radius. Larger the dataset, the larger value of MinPts must be chosen.
- As a general rule, the minimum MinPts can be derived from the number of dimensions D in the dataset as, MinPts >= D+1.
- The minimum value of **MinPts** must be chosen at least 3.

DB SCAN

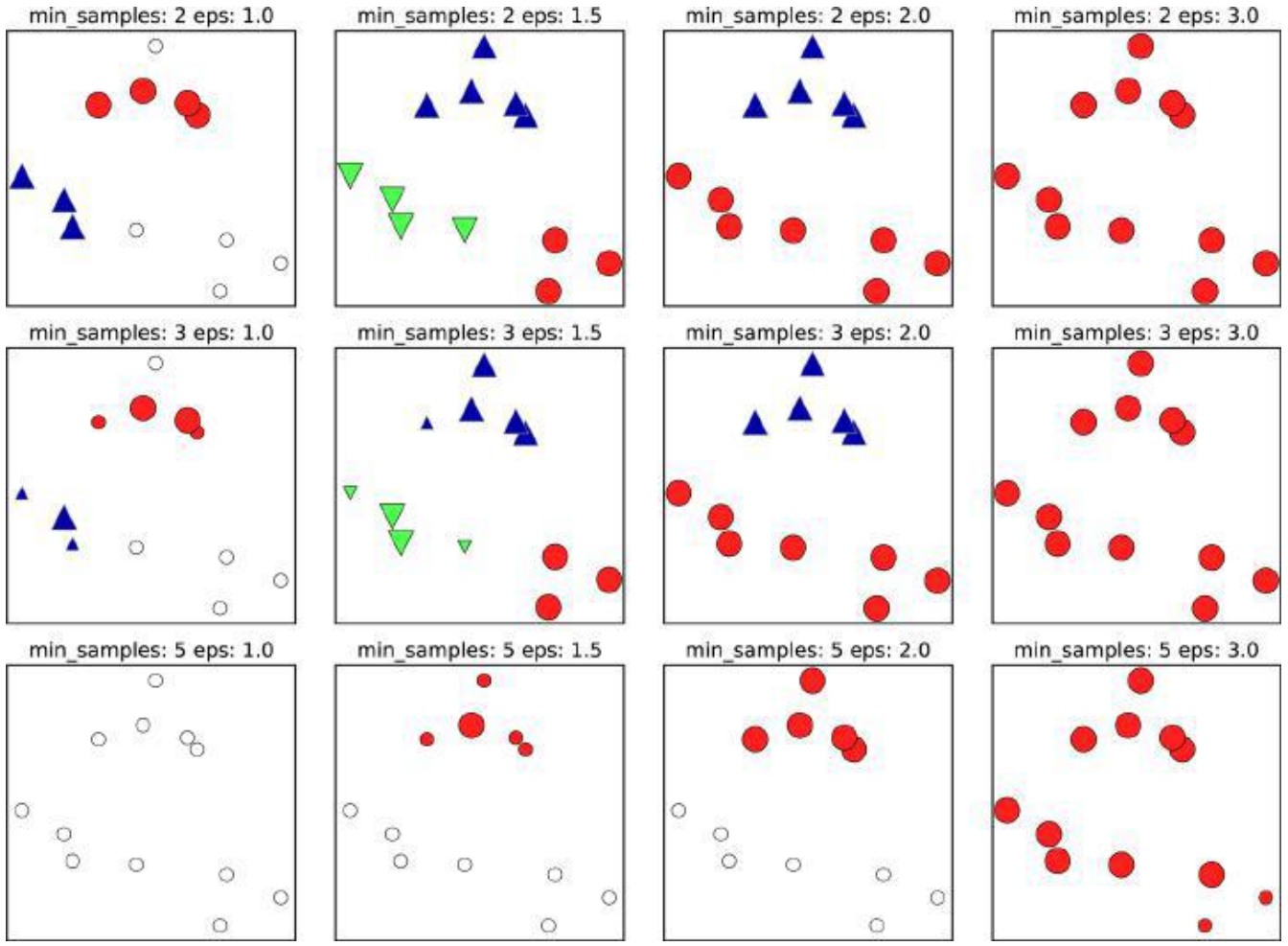
In the end, there are three kinds of points:

- core points,
- points that are within distance eps of core points (called *boundary points*),
- and noise.

When the DBSCAN algorithm is run on a particular dataset multiple times, the clustering of the core points is always the same, and the same points will always be labeled as noise.

Steps

- The algorithm works by picking an arbitrary point to start with. It then finds all points with distance eps or less from that point.
- If there are less than min_samples points within distance eps of the starting point, this point is labeled as *noise*, meaning that it doesn't belong to any cluster.
- If there are more than min_samples points within a distance of eps, the point is labeled a core sample and assigned a new cluster label.
- Then, all neighbors (within eps) of the point are visited.


min_samples: 2 eps: 1.0
min_samples: 2 eps: 1.5
min_samples: 2 eps: 2.0
min_samples: 2 eps: 3.0
min_samples: 3 eps: 1.0
min_samples: 3 eps: 1.5
min_samples: 3 eps: 2.0
min_samples: 3 eps: 3.0
min_samples: 5 eps: 1.0
min_samples: 5 eps: 1.5
min_samples: 5 eps: 2.0
min_samples: 5 eps: 3.0