POORNIMA
INSTITUTE OF ENGINEERING & TECHNOLOGY

Affiliated to RTU, Kota • Approved by AICTE & UGC under 2(f) • Accredited by NAAC and NBA

DEPARTMENT OF COMPUTER ENGINEERING

WELCOMES

5-Day AICTE Training And Learning (ATAL))

Faculty Development Programme

on

"Big Data Analytics With Deep Learning"

August 18-22, 2020

Unsupervised Machine learning Hands on Session using Python on K-Means Algorithm, Hierarchical Clustering.

What is K means Clustering?

- K-Means Clustering is an unsupervised learning algorithm that is used to solve the clustering problems in machine learning or data science.
- K-Means Clustering is an Unsupervised Learning algorithm, which groups the unlabeled dataset into different clusters. Here K defines the number of pre-defined clusters that need to be created in the process.
- It is a centroid-based algorithm, where each cluster is associated with a centroid. The main aim of this algorithm is to minimize the sum of distances between the data point and their corresponding clusters.
- The k-means clustering algorithm mainly performs two tasks:

1. Determines the best value for K center points or centroids by an iterative process.
2. Assigns each data point to its closest k-center. Those data points which are near to the particular k-center, create a cluster.

What is K means Clustering?

The working of the K-Means algorithm is explained in the below steps:

Step-1: Select the number K to decide the number of clusters.

Step-2: Select random K points or centroids. (It can be other from the input dataset).

Step-3: Assign each data point to their closest centroid, which will form the predefined K clusters.

Step-4: Calculate the variance and place a new centroid of each cluster.

Step-5: Repeat the third steps, which means reassign each datapoint to the new closest centroid of each cluster.

Step-6: If any reassignment occurs, then go to step-4 else go to FINISH.

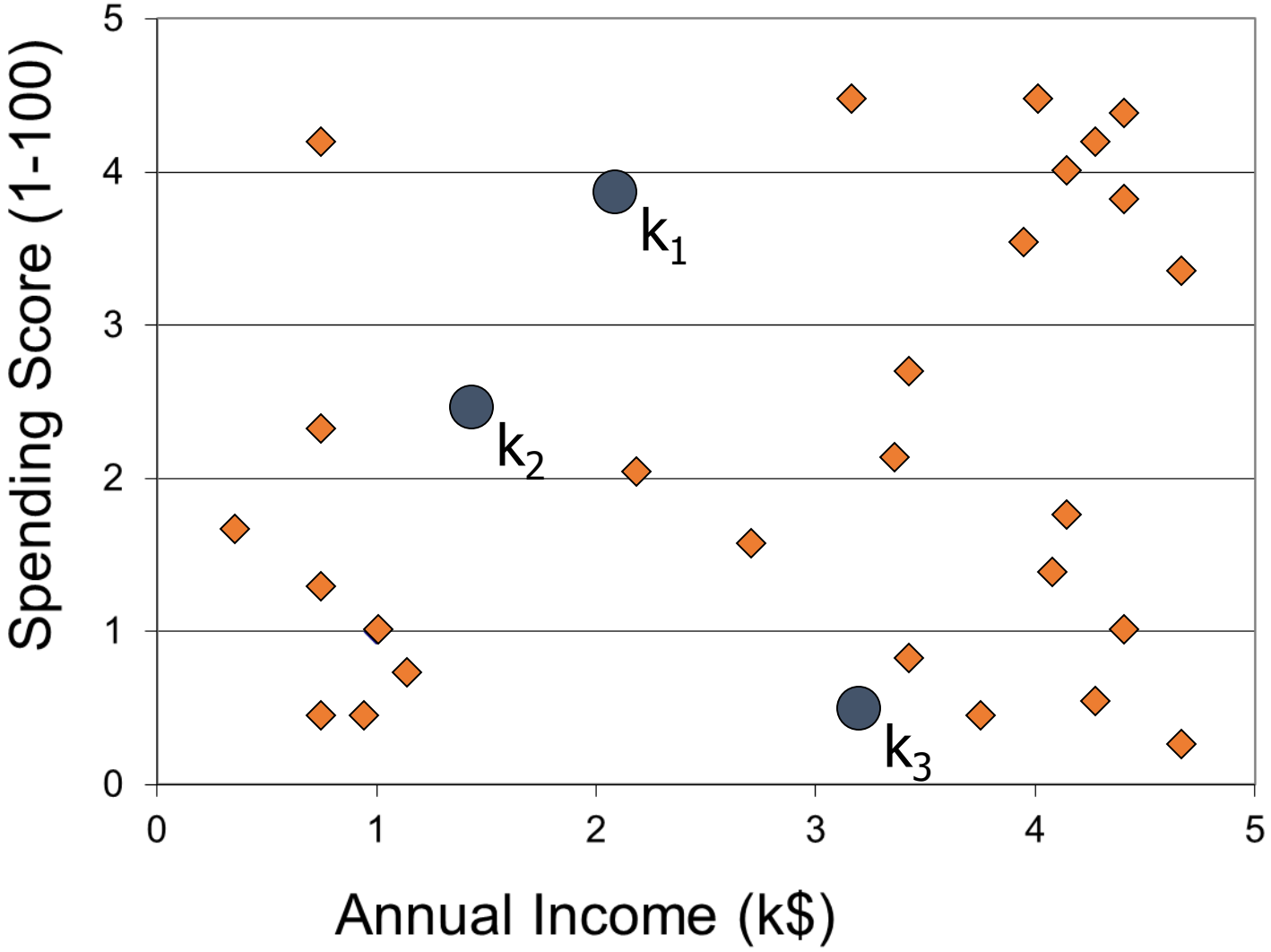
Step-7: The model is ready.

K-means algorithm Example

CustomerID	Genre	Age	Annual Income (k$)	Spending Score (1-100)
1	Male	19	15	39
2	Male	21	15	81
3	Female	20	16	6
4	Female	23	16	77
5	Female	31	17	40
6	Female	22	17	76
7	Female	35	18	6
8	Female	23	18	94
9	Male	64	19	3
10	Female	30	19	72
11	Male	67	19	14
12	Female	35	19	99
13	Female	58	20	15
14	Female	24	20	77

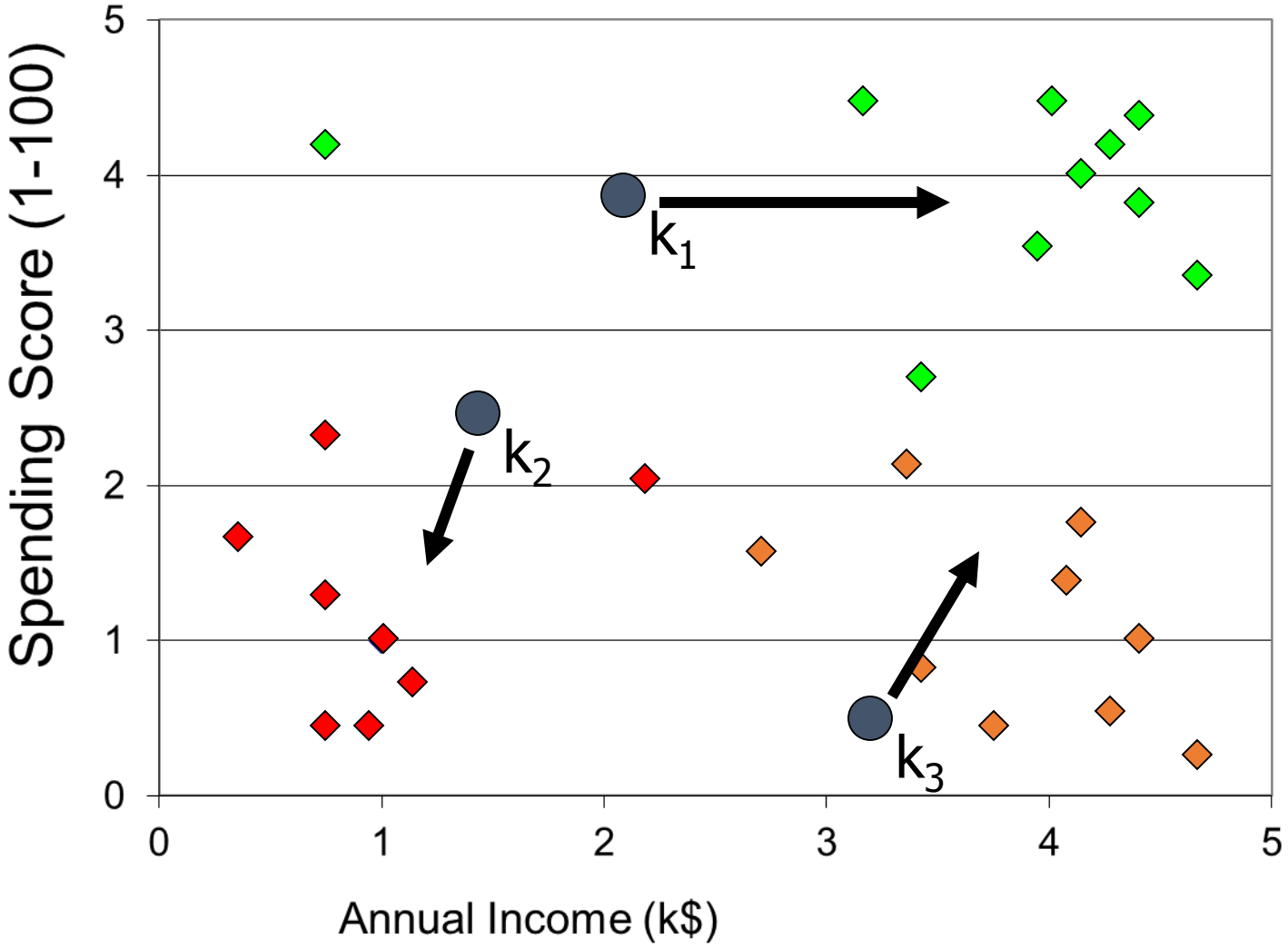
Clustering: Example 1, Step 1

Algorithm: k-means, Distance Metric: Euclidean Distance



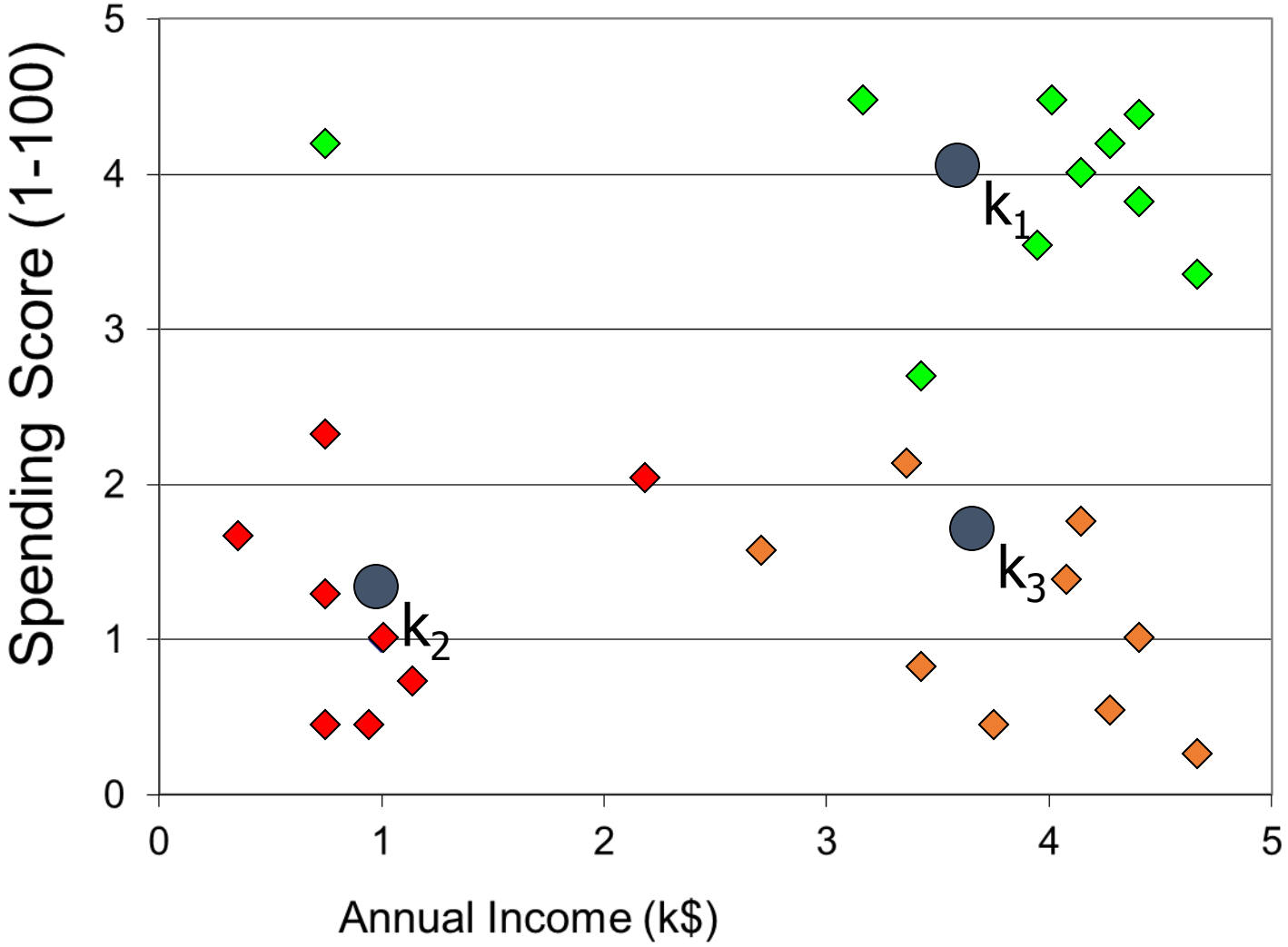
Clustering: Example 1, Step 2

Algorithm: k-means, Distance Metric: Euclidean Distance



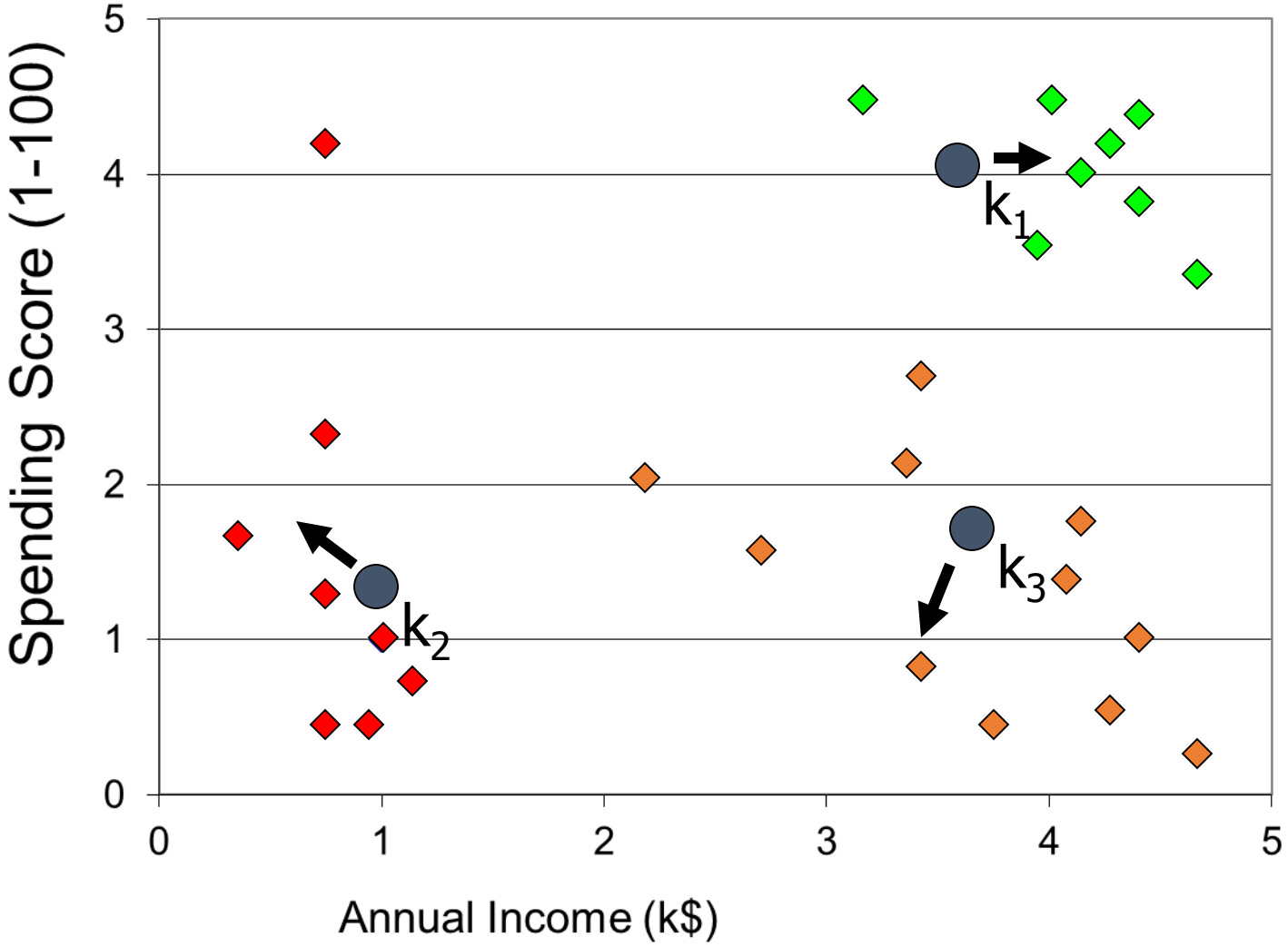
Clustering: Example 1, Step 3

Algorithm: k-means, Distance Metric: Euclidean Distance



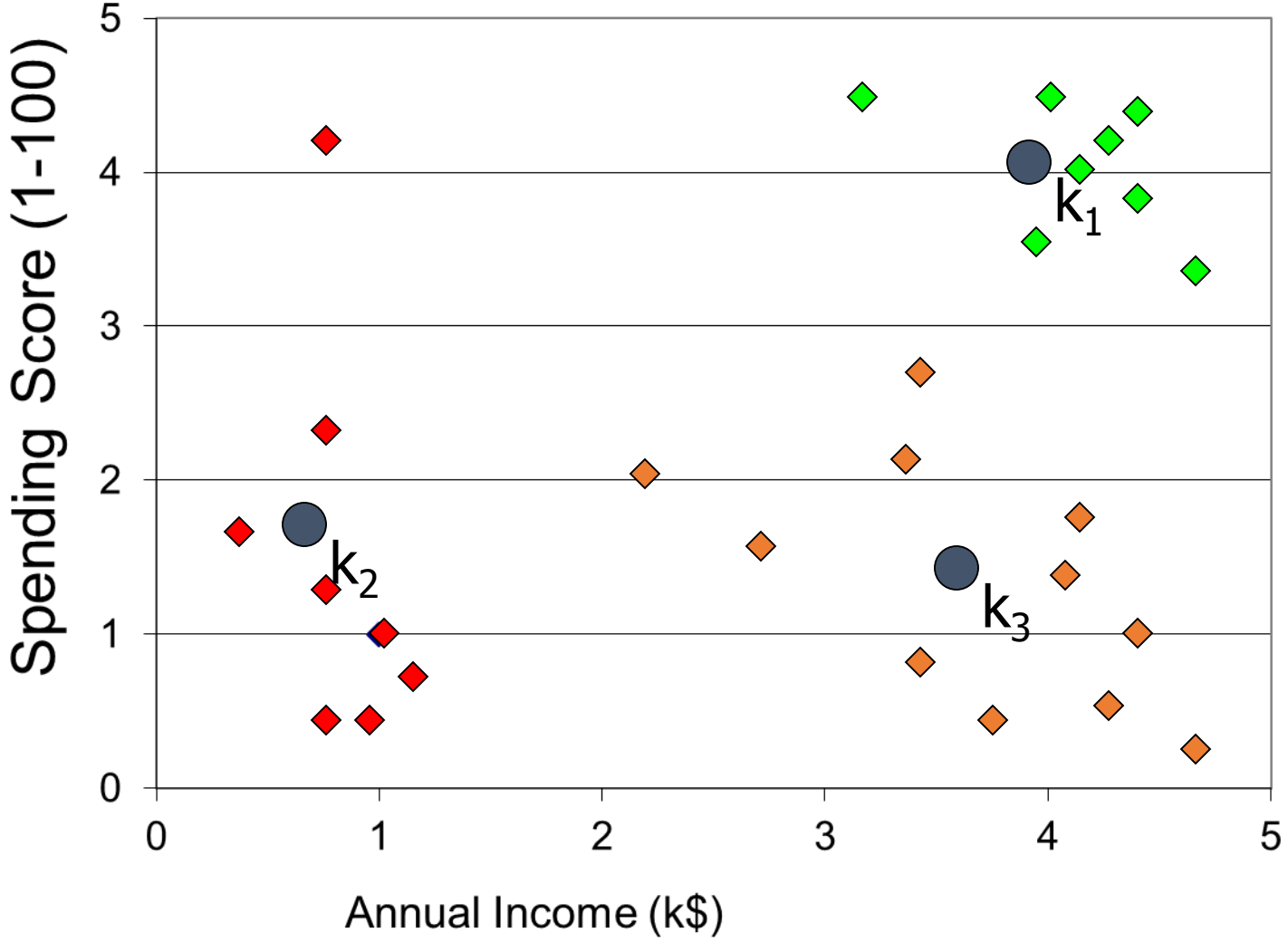
Clustering: Example 1, Step 4

Algorithm: k-means, Distance Metric: Euclidean Distance

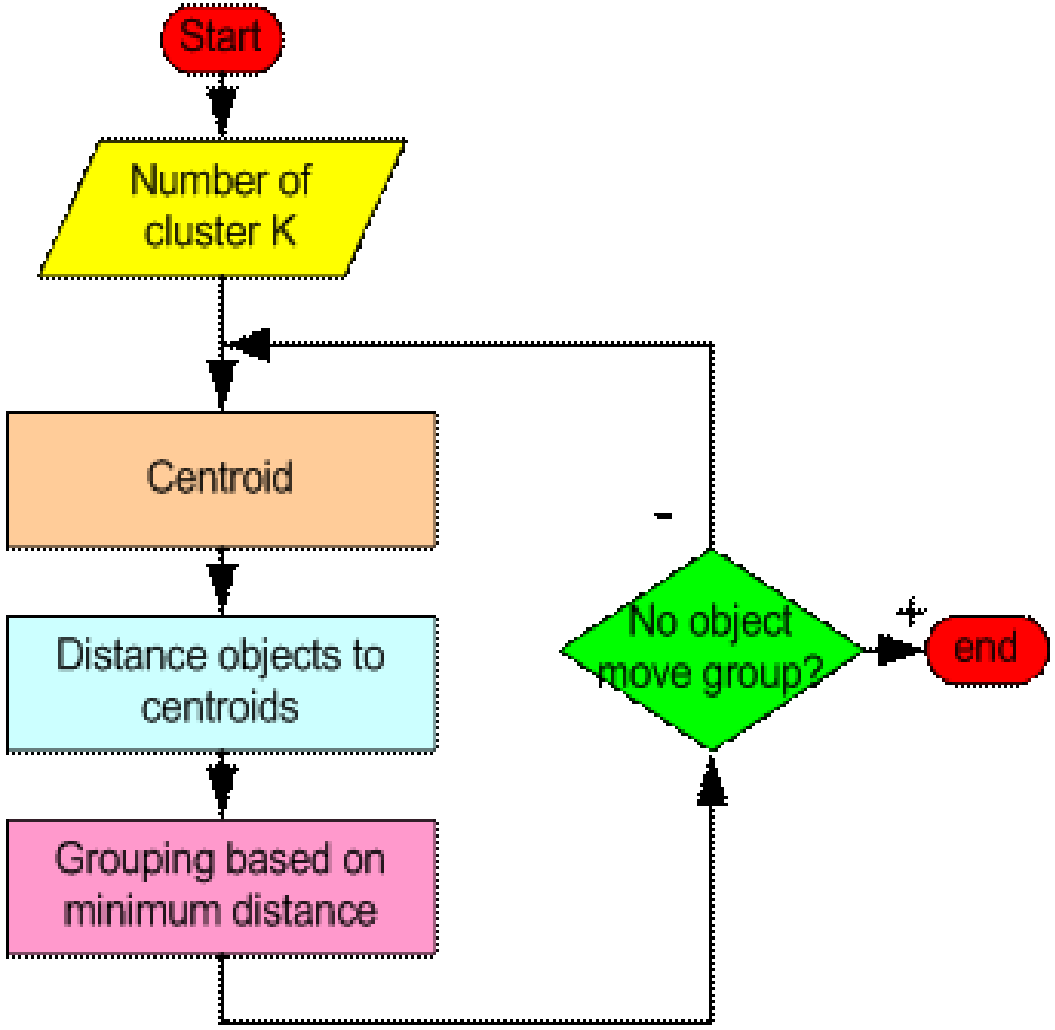


Clustering: Example 1, Step 5

Algorithm: k-means, Distance Metric: Euclidean Distance



How the K-Mean Clustering algorithm works?



How to choose the value of "K number of clusters" in K-means Clustering?

Elbow Method

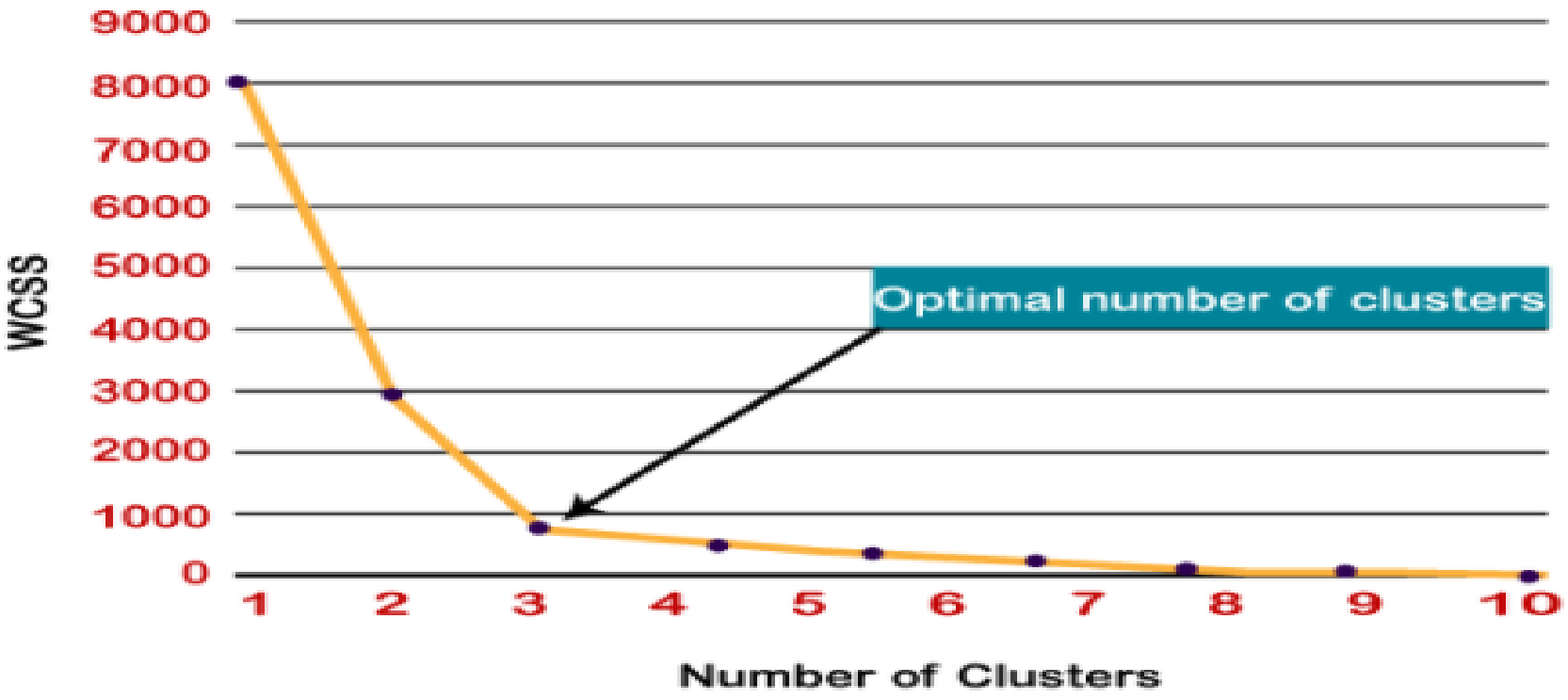
- The Elbow method is one of the most popular ways to find the optimal number of clusters. This method uses the concept of WCSS value. **WCSS** stands for **Within Cluster Sum of Squares**, which defines the total variations within a cluster.

- The formula to calculate the value of WCSS (for 3 clusters) is given below:

$$WCSS = \sum_{Pi\ in\ Cluster1} distance(P_i\ C_1)^2 + \sum_{Pi\ in\ Cluster2} distance(P_i\ C_2)^2 + \sum_{Pi\ in\ CLuster3} distance(P_i\ C_3)^2$$

How to choose the value of "K number of clusters" in K-means Clustering?

Elbow Method



Python Implementation of K-means Clustering Algorithm

The steps to be followed for the implementation are given below:

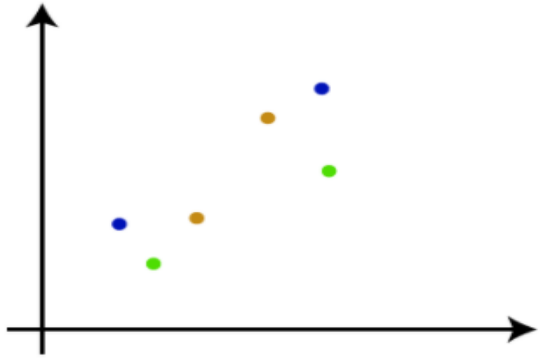
1. **Data Pre-processing**
2. **Finding the optimal number of clusters using the elbow method**
3. **Training the K-means algorithm on the training dataset**
4. **Visualizing the clusters**

Hierarchical Clustering in Machine Learning

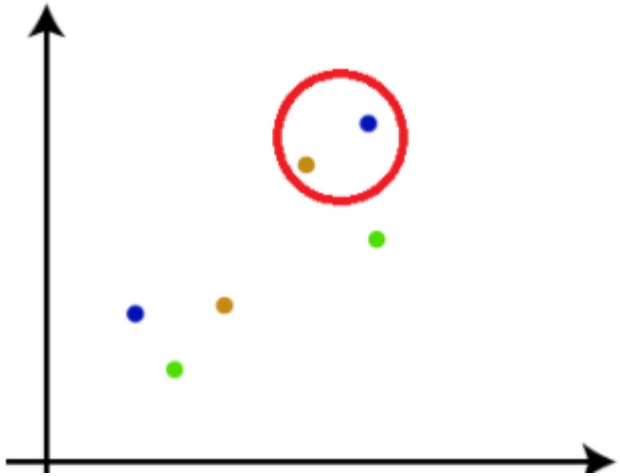
- Hierarchical clustering is another unsupervised machine learning algorithm, which is used to group the unlabeled datasets into a cluster and also known as **hierarchical cluster analysis** or HCA.
- we develop the hierarchy of clusters in the form of a tree, and this tree-shaped structure is known as the **dendrogram**.
- The hierarchical clustering technique has two approaches:
  1. Agglomerative: Agglomerative is a bottom-up approach, in which the algorithm starts with taking all data points as single clusters and merging them until one cluster is left.
  2. Divisive: Divisive algorithm is the reverse of the agglomerative algorithm as it is a top-down approach.

How the Agglomerative Hierarchical clustering Work?

- **Step-1:** Create each data point as a single cluster. Let's say there are N data points, so the number of clusters will also be N.

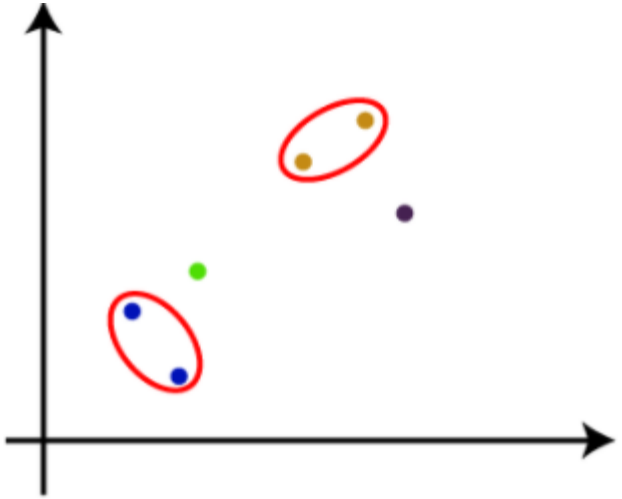


- **Step-2:** Take two closest data points or clusters and merge them to form one cluster. So, there will now be N-1 clusters.

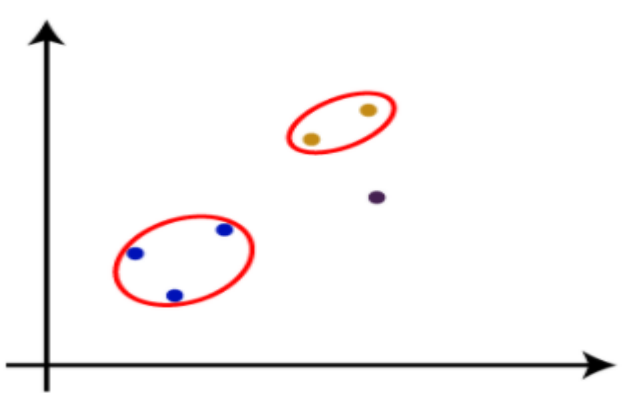


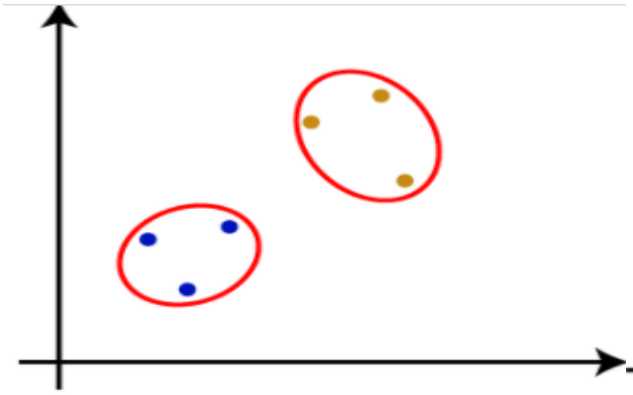
How the Agglomerative Hierarchical clustering Work?

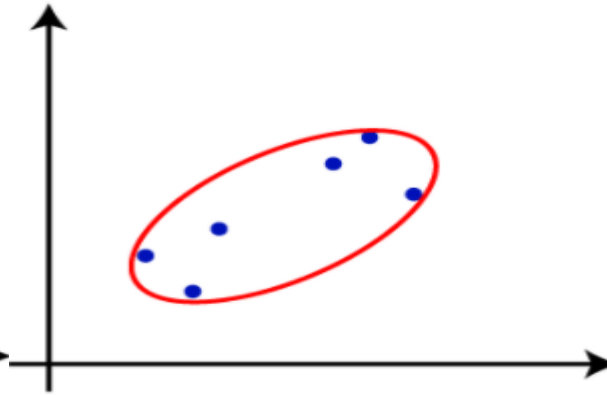
- **Step-3**: Again, take the two closest clusters and merge them together to form one cluster. There will be N-2 clusters



- **Step4**: Repeat Step 3 until only one cluster left. So, we will get the following clusters. Consider the below images:





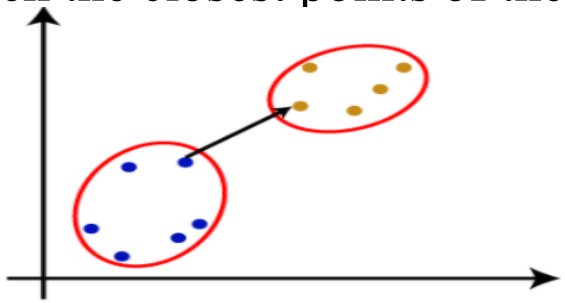


How the Agglomerative Hierarchical clustering Work?

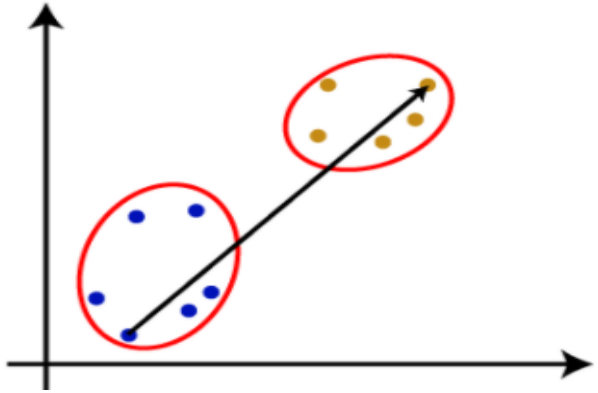
- **Step-5:** Once all the clusters are combined into one big cluster, develop the dendrogram to divide the clusters as per the problem..

Measure for the distance between two clusters

- There are various ways to calculate the distance between two clusters, and these ways decide the rule for clustering. These measures are called Linkage methods. Some of the popular linkage methods are given below:
- **Single Linkage:** It is the Shortest Distance between the closest points of the clusters. Consider the below image:



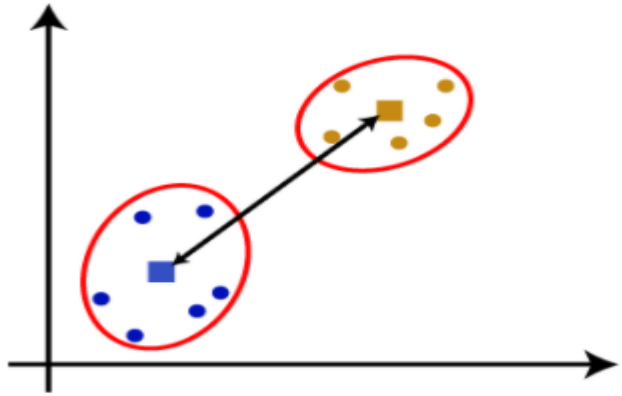
- **Complete Linkage:** It is the farthest distance between the two points of two different clusters. It is one of the popular linkage methods as it forms tighter clusters than single-linkage.



Measure for the distance between two clusters

Average Linkage: It is the linkage method in which the distance between each pair of datasets is added up and then divided by the total number of datasets to calculate the average distance between two clusters. It is also one of the most popular linkage methods.

Centroid Linkage: It is the linkage method in which the distance between the centroid of the clusters is calculated. Consider the below image:



Distance between two clusters

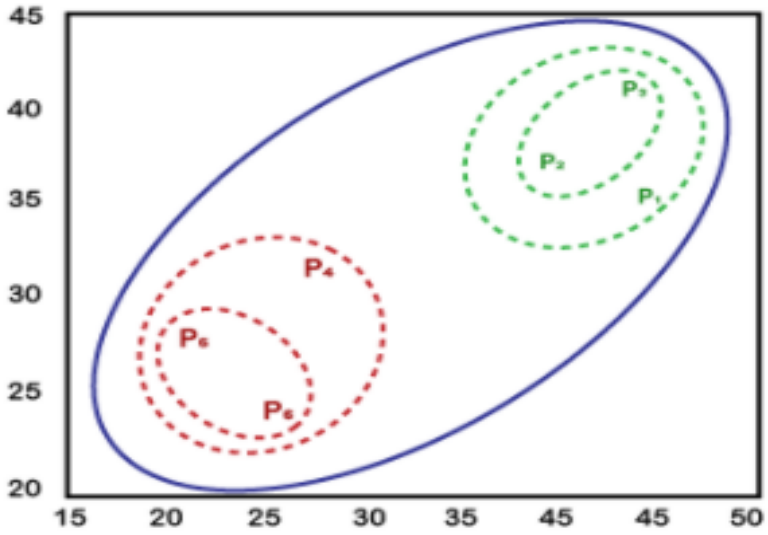
- Ward's distance between clusters Ci and Cj is the difference between the total within cluster sum of squares for the two clusters separately, and the within cluster sum of squares resulting from merging the two clusters in cluster Cij

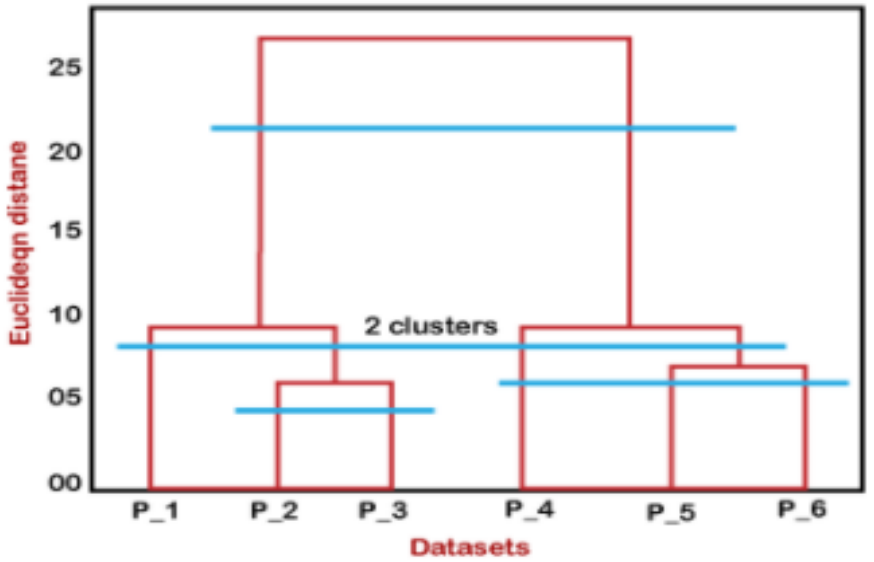
$$D_W(C_i, C_j) = \sum_{x \in C_i} (x - r_i)^2 + \sum_{x \in C_j} (x - r_j)^2 - \sum_{x \in C_{ij}} (x - r_{ij})^2$$

- r_i: centroid of C_i
- r_j: centroid of C_j
- r_{ij}: centroid of C_{ij}

Woking of Dendrogram in Hierarchical clustering

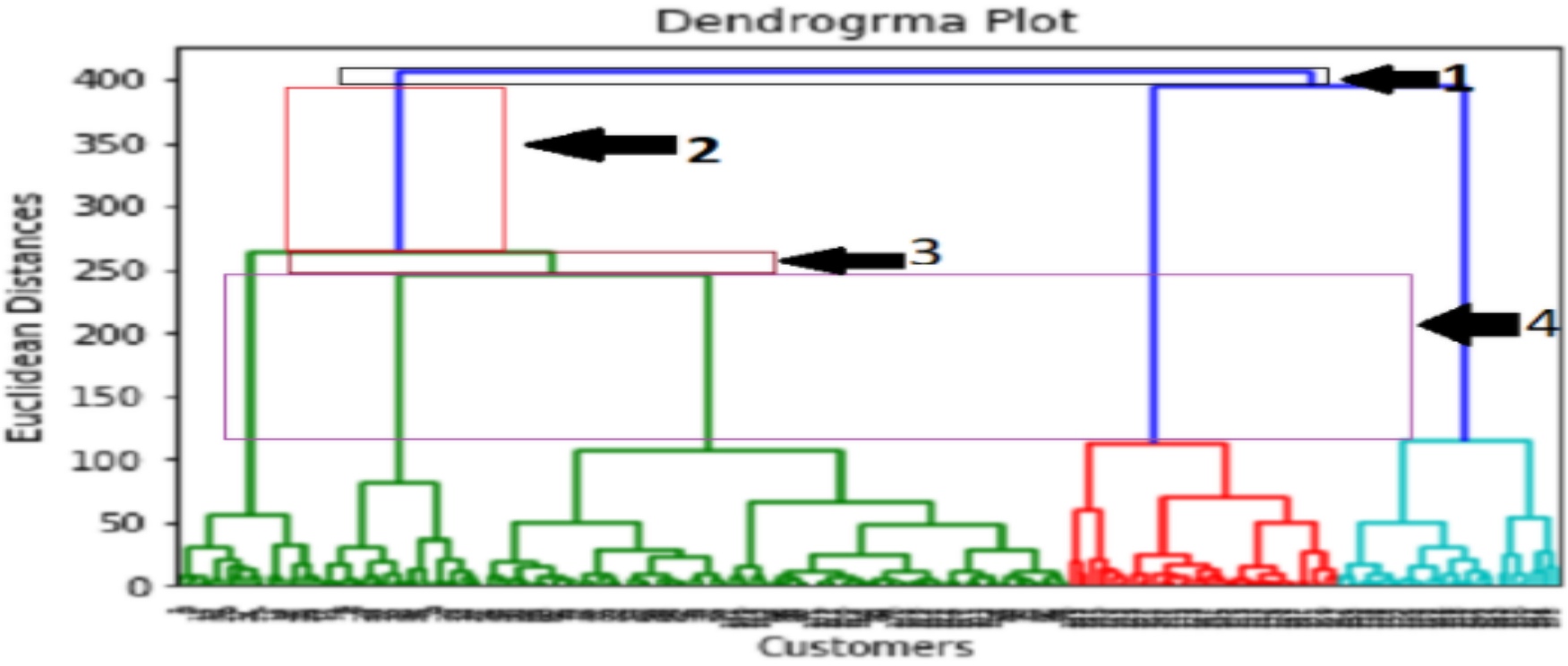
The dendrogram is a tree-like structure that is mainly used to store each step as a memory that the HC algorithm performs. In the dendrogram plot, the Y-axis shows the Euclidean distances between the data points, and the x-axis shows all the data points of the given dataset.





Woking of Dendrogram in Hierarchical clustering

we will now determine the optimal number of clusters for our model. For this, we will find the **maximum vertical distance** that does not cut any horizontal bar. Consider the below diagram:



Python Implementation of Agglomerative Hierarchical Clustering

.

1. **Data Pre-processing**
2. **Finding the optimal number of clusters using the Dendrogram**
3. **Training the hierarchical clustering model**
4. **Visualizing the clusters**

References

- *Srivastava, Nitish, et al. "**Dropout: a simple way to prevent neural networks from overfitting.**" Journal of machine learning research (2014)*
- *Bergstra, James, and Yoshua Bengio. "**Random search for hyper-parameter optimization.**" Journal of Machine Learning Research, Feb (2012)*
- *Kim, Y. "**Convolutional Neural Networks for Sentence Classification**", EMNLP (2014)*
- Severyn, Aliaksei, and Alessandro Moschitti. "**UNITN: Training Deep Convolutional Neural Network for Twitter Sentiment Classification.**" *SemEval@ NAACL-HLT* (2015)
- *Cho, Kyunghyun, et al. "**Learning phrase representations using RNN encoder-decoder for statistical machine translation**." EMNLP (2014)*
- Ilya Sutskever et al. "**Sequence to sequence learning with neural networks.**" *NIPS (*2014)
- *Bahdanau et al. "**Neural machine translation by jointly learning to align and translate.**" ICLR (2015)*
- *Gal, Y., Islam, R., Ghahramani, Z. "**Deep Bayesian Active Learning with Image Data**." ICML (2017)*
- *Nair, V., Hinton, G.E. "**Rectified linear units improve restricted boltzmann machines**." ICML (2010)*
- Ronan Collobert, et al. "**Natural language processing (almost) from scratch**." *JMLR* (2011)

- *Kumar, Shantanu. "A Survey of Deep Learning Methods for Relation Extraction." arXiv preprint arXiv:1705.03645 (2017)*
- *Lin et al. "Neural Relation Extraction with Selective Attention over Instances" ACL (2016) [code]*
- *Zeng, D.et al. "Relation classification via convolutional deep neural network". COLING (2014)*
- *Nguyen, T.H., Grishman, R. "Relation extraction: Perspective from CNNs." VS@ HLT-NAACL. (2015)*
- *Zhang, D., Wang, D. "Relation classification via recurrent NN." -arXiv preprint arXiv:1508.01006 (2015)*
- *Zhou, P. et al. "Attention-based bidirectional LSTM networks for relation classification . ACL (2016)*

References & Resources

- http://web.stanford.edu/class/cs224n
- https://www.coursera.org/specializations/deep-learning
- https://chrisalbon.com/#Deep-Learning
- http://www.asimovinstitute.org/neural-network-zoo
- *http://cs231n.github.io/optimization-2*
- *https://medium.com/@ramrajchandradevan/the-evolution-of-gradient-descend-optimization-algorithm-4106a6702d39*
- https://arimo.com/data-science/2016/bayesian-optimization-hyperparameter-tuning
- http://www.wildml.com/2015/12/implementing-a-cnn-for-text-classification-in-tensorflow
- http://www.wildml.com/2015/11/understanding-convolutional-neural-networks-for-nlp
- https://medium.com/technologymadeeasy/the-best-explanation-of-convolutional-neural-networks-on-the-internet-fbb8b1ad5df8
- *http://www.wildml.com/2015/09/recurrent-neural-networks-tutorial-part-1-introduction-to-rnns/*
- *http://www.wildml.com/2015/10/recurrent-neural-network-tutorial-part-4-implementing-a-grulstm-rnn-with-python-and-theano/*
- *http://colah.github.io/posts/2015-08-Understanding-LSTMs*
- https://github.com/hyperopt/hyperopt
- https://github.com/tensorflow/nmt


thank you