**🧠 One-Day Workshop: Unsupervised Learning with Python**

**📅 Agenda (One Full Day – ~6 Hours Effective Learning)**

| **Topic** |
| --- |
| Welcome & Introduction to Unsupervised Learning |
| Clustering Techniques: K-Means, Hierarchical, DBSCAN |
|  |
| Dimensionality Reduction: PCA, t-SNE |
| 🍽️ Lunch Break |
| Association Rule Learning (Apriori, Eclat) |
| Real-world Dataset Hands-On (Clustering + PCA) |
|  |
| Evaluation Metrics + Challenges in Unsupervised Learning |
| Quiz + Wrap-Up + Certificate Distribution (if applicable) |

**📘 Theory + Code for Each Topic**

**1️⃣ What is Unsupervised Learning?**

* No labelled data
* Goal: Discover structure/patterns in data
* Applications:
  + Market segmentation
  + Anomaly detection
  + Recommender systems
  + Image compression

**2️⃣ Clustering Algorithms**

**A. K-Means Clustering**

**🧠 Theory:**

* Partitions data into **K clusters**
* Minimizes **intra-cluster distance**
* Uses **centroids**

**🧪 Python Code:**

from sklearn.datasets import make\_blobs

from sklearn.cluster import KMeans

import matplotlib.pyplot as plt

X, \_ = make\_blobs(n\_samples=300, centers=4, random\_state=42)

kmeans = KMeans(n\_clusters=4)

y\_kmeans = kmeans.fit\_predict(X)

plt.scatter(X[:, 0], X[:, 1], c=y\_kmeans)

plt.scatter(kmeans.cluster\_centers\_[:, 0], kmeans.cluster\_centers\_[:, 1], s=200, c='red')

plt.title("K-Means Clustering")

plt.show()

**B. Hierarchical Clustering**

**🧠 Theory:**

* Builds a tree of clusters (dendrogram)
* Agglomerative (bottom-up) or Divisive (top-down)

**🧪 Python Code:**

from scipy.cluster.hierarchy import dendrogram, linkage

from sklearn.datasets import make\_blobs

X, \_ = make\_blobs(n\_samples=100, centers=3)

linked = linkage(X, 'ward')

plt.figure(figsize=(10, 7))

dendrogram(linked)

plt.title("Hierarchical Clustering Dendrogram")

plt.show()

**C. DBSCAN**

**🧠 Theory:**

* Density-based
* No need to specify number of clusters
* Good for irregular shapes

**🧪 Python Code:**

from sklearn.cluster import DBSCAN

from sklearn.datasets import make\_moons

X, \_ = make\_moons(n\_samples=300, noise=0.05)

db = DBSCAN(eps=0.2, min\_samples=5)

labels = db.fit\_predict(X)

plt.scatter(X[:, 0], X[:, 1], c=labels)

plt.title("DBSCAN Clustering")

plt.show()

**3️⃣ Dimensionality Reduction**

**A. PCA – Principal Component Analysis**

**🧠 Theory:**

* Converts correlated features into fewer uncorrelated components
* Used for visualization and speeding up models

**🧪 Python Code:**

from sklearn.decomposition import PCA

from sklearn.datasets import load\_iris

import matplotlib.pyplot as plt

data = load\_iris()

X = data.data

pca = PCA(n\_components=2)

X\_pca = pca.fit\_transform(X)

plt.scatter(X\_pca[:, 0], X\_pca[:, 1], c=data.target)

plt.xlabel("PC1")

plt.ylabel("PC2")

plt.title("PCA on Iris Dataset")

plt.show()

**B. t-SNE – t-Distributed Stochastic Neighbor Embedding**

**🧠 Theory:**

* Non-linear dimensionality reduction
* Good for visualization of high-dimensional data

**🧪 Python Code:**

from sklearn.manifold import TSNE

from sklearn.datasets import load\_digits

digits = load\_digits()

tsne = TSNE(n\_components=2, perplexity=30, n\_iter=300)

X\_tsne = tsne.fit\_transform(digits.data)

plt.scatter(X\_tsne[:, 0], X\_tsne[:, 1], c=digits.target, cmap='tab10')

plt.title("t-SNE on Digits Dataset")

plt.show()

**4️⃣ Association Rule Learning**

**A. Apriori Algorithm**

**🧠 Theory:**

* Used for market basket analysis
* Finds frequent itemsets and association rules

**🧪 Python Code using mlxtend:**

import pandas as pd

from mlxtend.frequent\_patterns import apriori, association\_rules

# Sample transactions

dataset = [

['milk', 'bread', 'butter'],

['bread', 'butter'],

['milk', 'bread'],

['milk', 'bread', 'butter'],

['bread', 'butter']

]

df = pd.DataFrame(dataset)

df = df.apply(lambda x: pd.Series(x.dropna().values), axis=1).stack().reset\_index(level=1, drop=True).to\_frame('item')

df['transaction'] = df.index

basket = pd.get\_dummies(df.pivot\_table(index='transaction', columns='item', aggfunc=len, fill\_value=0).astype(bool))

frequent\_itemsets = apriori(basket, min\_support=0.4, use\_colnames=True)

rules = association\_rules(frequent\_itemsets, metric="lift", min\_threshold=1)

print(rules[['antecedents', 'consequents', 'support', 'confidence', 'lift']])

**5️⃣ Evaluation Metrics for Clustering**

* **Silhouette Score**
* **Davies–Bouldin Index**
* **Inertia (for K-means)**
* **Visual inspection (for low dimensions)**

**🧪 Sample Code:**

from sklearn.metrics import silhouette\_score

score = silhouette\_score(X, y\_kmeans)

print(f"Silhouette Score: {score}")

**6️⃣ Real-World Example Exercise (Hands-On)**

**Dataset: Mall Customers Dataset**

* Features: Age, Annual Income, Spending Score

import pandas as pd

from sklearn.cluster import KMeans

import matplotlib.pyplot as plt

df = pd.read\_csv('Mall\_Customers.csv')

X = df[['Annual Income (k$)', 'Spending Score (1-100)']]

kmeans = KMeans(n\_clusters=5)

df['Cluster'] = kmeans.fit\_predict(X)

plt.scatter(X['Annual Income (k$)'], X['Spending Score (1-100)'], c=df['Cluster'], cmap='rainbow')

plt.xlabel("Annual Income")

plt.ylabel("Spending Score")

plt.title("Customer Segmentation")

plt.show()

**📊 Tools/Platforms for Hands-On**

* Google Colab (no setup needed)
* Jupyter Notebook
* Kaggle Notebooks

Traditional Programming

Input Data :

1. Input Data
2. Program

Output Data:

1. Program generates output by processing the input.

Addition of Two numbers

Public int add(int a,int b) {

Return (a+b);

}

Add(4,5) 🡪 9

Add(3,2) 🡪 5

In Machine Learning

Input Data:

1. Set of labelled or unlabelled data (a csv file containing m,x,c) – csv will have 3 columns each one for m,x,c
2. It will also have output data in the form of csv or dataset (a csv containing result of y=mx+c)

Expected Output

1. Machines learns the relationship between input and output data and generate the formula i.e y=mx+c

Preparing the data

1. Preparing the test data - <https://www.mockaroo.com/>
2. Loading and pre-processing the data – using python libraries pandas, numpy
3. Train the model – scikit-learn, tensor-flow, seaborn
4. Predict/classify the result by testing the model
5. Check precision and other parameters of model
6. Visualize the data using libs such as matplotlib,

<https://github.com/syskantechnosoft/education>

**1. Clustering Algorithms**

[Clustering](https://www.geeksforgeeks.org/clustering-in-machine-learning/) in unsupervised machine learning is the process of grouping unlabeled data into clusters based on their similarities. The goal of clustering is to identify patterns and relationships in the data without any prior knowledge of the data's meaning.

Broadly this technique is applied to group data based on different patterns, such as similarities or differences, our machine model finds. These algorithms are used to process raw, unclassified data objects into groups. For example, in the above figure, we have not given output parameter values, so this technique will be used to group clients based on the input parameters provided by our data.

***Some common clustering algorithms:***

* [***K-means Clustering***](https://www.geeksforgeeks.org/k-means-clustering-introduction/)***:*** *Groups data into K clusters based on how close the points are to each other.*
* [***Hierarchical Clustering***](https://www.geeksforgeeks.org/ml-hierarchical-clustering-agglomerative-and-divisive-clustering/)***:*** *Creates clusters by building a tree step-by-step, either merging or splitting groups.*
* [***Density-Based Clustering (DBSCAN)***](https://www.geeksforgeeks.org/dbscan-clustering-in-ml-density-based-clustering/)***:*** *Finds clusters in dense areas and treats scattered points as noise.*
* [***Mean-Shift Clustering***](https://www.geeksforgeeks.org/ml-mean-shift-clustering/)***:*** *Discovers clusters by moving points toward the most crowded areas.*
* [***Spectral Clustering***](https://www.geeksforgeeks.org/ml-spectral-clustering/)***:*** *Groups data by analyzing connections between points using graphs.*

**2. Association Rule Learning**

[Association rule learning](https://www.geeksforgeeks.org/association-rule/) is also known as association rule mining is a common technique used to discover associations in unsupervised machine learning. This technique is a rule-based ML technique that finds out some very useful relations between parameters of a large data set. This technique is basically used for market basket analysis that helps to better understand the relationship between different products.

For e.g. shopping stores use algorithms based on this technique to find out the relationship between the sale of one product w.r.t to another's sales based on customer behavior. **Like if a customer buys milk, then he may also buy bread, eggs, or butter**. Once trained well, such models can be used to increase their sales by planning different offers.

***Some common Association Rule Learning algorithms:***

* [***Apriori Algorithm***](https://www.geeksforgeeks.org/apriori-algorithm/)***:****Finds patterns by exploring frequent item combinations step-by-step.*
* [***FP-Growth Algorithm***](https://www.geeksforgeeks.org/frequent-pattern-growth-algorithm/)***:****An Efficient Alternative to Apriori. It quickly identifies frequent patterns without generating candidate sets.*
* [***Eclat Algorithm***](https://www.geeksforgeeks.org/ml-eclat-algorithm/)***:****Uses intersections of itemsets to efficiently find frequent patterns.*
* [***Efficient Tree-based Algorithms***](https://www.geeksforgeeks.org/introduction-to-tree-data-structure-and-algorithm-tutorials/)***:****Scales to handle large datasets by organizing data in tree structures.*

**3. Dimensionality Reduction**

Dimensionality reduction is the process of reducing the number of features in a dataset while preserving as much information as possible. This technique is useful for improving the performance of machine learning algorithms and for data visualization.

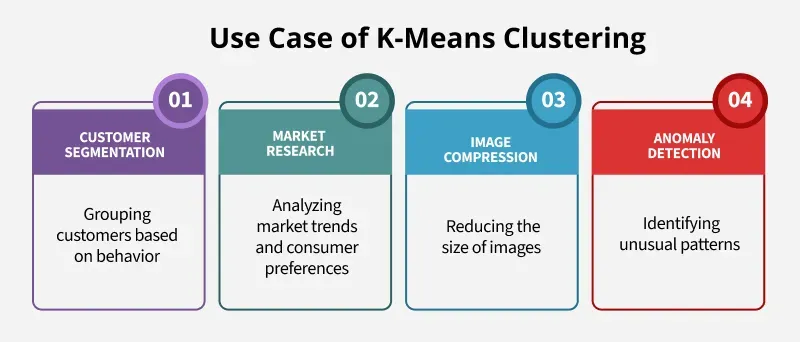
Imagine a dataset of 100 features about students (height, weight, grades, etc.). To focus on key traits, you reduce it to just 2 features: height and grades, making it easier to visualize or analyze the data.

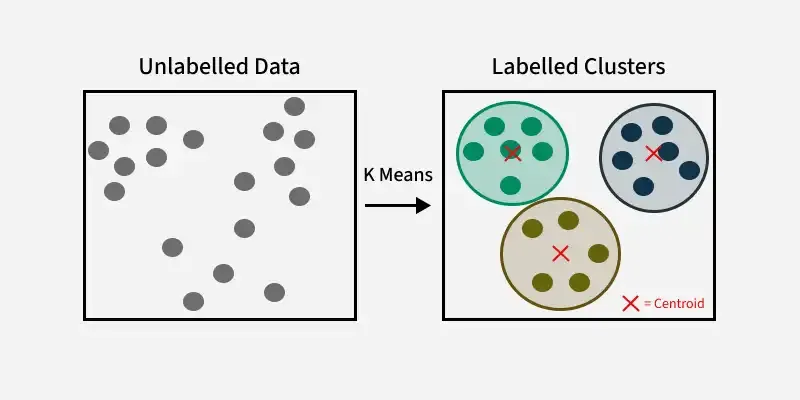
*Here are some popular* ***Dimensionality Reduction algorithms****:*

* [***Principal Component Analysis (PCA)***](https://www.geeksforgeeks.org/principal-component-analysis-pca/)***:****Reduces dimensions by transforming data into uncorrelated principal components.*
* [***Linear Discriminant Analysis (LDA)***](https://www.geeksforgeeks.org/ml-linear-discriminant-analysis/)***:****Reduces dimensions while maximizing class separability for classification tasks.*
* [***Non-negative Matrix Factorization (NMF***](https://www.geeksforgeeks.org/non-negative-matrix-factorization/)***):****Breaks data into non-negative parts to simplify representation.*
* [***Locally Linear Embedding (LLE)***](https://www.geeksforgeeks.org/locally-linear-embedding-in-machine-learning/)***:****Reduces dimensions while preserving the relationships between nearby points.*
* [***Isomap***](https://www.geeksforgeeks.org/isomap-a-non-linear-dimensionality-reduction-technique/)***:****Captures global data structure by preserving distances along a manifold.*

K-Means Clustering







Example Colab URL

<https://colab.research.google.com/drive/1HF6d8r1fksPpf9W_w_KQs_poqH5zs5vR#scrollTo=t1SAKLL-XY7V>