. https://github.com/syskantechnosoft/MRECW-ALMLDS2025

**🧠 1-Day Machine Learning with Python – Bootcamp**

**📅 Agenda Overview**

| **Topic** |
| --- |
| Introduction to ML & Python Ecosystem |
| Supervised Learning - Concepts & Hands-on |
| Unsupervised Learning - Concepts & Hands-on |
| Semi-Supervised & Self-Supervised Learning |
| **Lunch** |
| Model Evaluation, Overfitting, and Regularization |
| Real-Life Projects (Classification + Clustering) |
| Advanced ML Concepts: Ensemble, Dimensionality Reduction |
| Modern ML Trends & Tools (AutoML, HuggingFace, etc.) |
| Q&A + Next Steps for Learning |

**🔧 Platform for Hands-On**

**Cloud Run:** ✅ [Google Colab](https://colab.research.google.com/) (No installation required but internet connection is needed completely to run the code)  
**Local Run:** Jupyter Notebooks via [Anaconda](https://www.anaconda.com/) (if local install preferred – internet is only to download all the requirements not to run the code)

**🧑‍🏫 Session 1: Introduction to Machine Learning**

**✅ Topics:**

* What is Machine Learning?
* Difference between AI, ML, DL
* Types of Learning: Supervised, Unsupervised, Semi-Supervised
* Python Libraries: pandas, numpy, scikit-learn, matplotlib, seaborn

**✅ Code Exercise:**

import pandas as pd

import seaborn as sns

df = sns.load\_dataset('iris')

df.head()

**✅ Real-World Example:**

* Predicting house prices
* Email spam detection

**🔍 Session 2: Supervised Learning**

**✅ Concepts:**

* Features and Labels
* Classification vs Regression

**✅ Algorithms:**

* Linear Regression
* Logistic Regression
* Decision Trees

**✅ Hands-On Example: Predict House Price**

from sklearn.linear\_model import LinearRegression

from sklearn.datasets import fetch\_california\_housing

from sklearn.model\_selection import train\_test\_split

data = fetch\_california\_housing()

X\_train, X\_test, y\_train, y\_test = train\_test\_split(data.data, data.target)

model = LinearRegression()

model.fit(X\_train, y\_train)

print("Predicted:", model.predict(X\_test[:1]))

**✅ Simple Exercise:**

* Predict salary based on years of experience (use synthetic data)

**🧩 Session 3: Unsupervised Learning**

**✅ Concepts:**

* No labels
* Grouping or dimensionality reduction

**✅ Algorithms:**

* K-Means Clustering
* PCA

**✅ Real-World Example:**

* Customer segmentation
* Market basket analysis

**✅ Hands-On Example: KMeans Clustering**

from sklearn.cluster import KMeans

from sklearn.datasets import make\_blobs

import matplotlib.pyplot as plt

X, \_ = make\_blobs(n\_samples=300, centers=4, cluster\_std=0.60)

model = KMeans(n\_clusters=4)

model.fit(X)

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

plt.title("Clustered Data")

plt.show()

**🤖 Session 4: Semi-Supervised & Self-Supervised Learning (11:30–12:00 PM)**

**✅ Concepts:**

* Few labeled, many unlabeled data
* Label propagation techniques
* Used in NLP, CV

**✅ Real-World Use Case:**

* Sentiment analysis with limited labeled reviews

**🧪 Session 5: Model Evaluation**

**✅ Concepts:**

* Accuracy, Precision, Recall, F1
* Train/Test Split
* Overfitting vs Underfitting
* Cross-validation

**✅ Hands-On:**

from sklearn.metrics import accuracy\_score

y\_pred = model.predict(X\_test)

accuracy\_score(y\_test, y\_pred)

**💼 Session 6: Mini-Projects**

**✅ Project 1: Iris Classification**

* Dataset: seaborn.load\_dataset('iris')
* Classify flower type based on petal & sepal

**✅ Project 2: Customer Segmentation**

* Dataset: Synthetic
* Apply KMeans and visualize clusters

**🧠 Session 7: Advanced ML Concepts**

**✅ Topics:**

* Ensemble Methods: Random Forest, XGBoost
* Dimensionality Reduction: PCA, t-SNE
* Feature Engineering

**✅ Real-World Example:**

* Credit card fraud detection (use outliers)

**✅ Code Snippet:**

from sklearn.ensemble import RandomForestClassifier

model = RandomForestClassifier()

model.fit(X\_train, y\_train)

**🔬 Session 8: Modern ML Trends & Tools**

**✅ Tools:**

| **Tool** | **Use Case** |
| --- | --- |
| AutoML (Google, AWS SageMaker) | No-code ML |
| HuggingFace | Pretrained models (NLP/CV) |
| PyCaret | Rapid prototyping |
| Gradio / Streamlit | Build ML web apps |
| ONNX | ML model interoperability |
| DVC | Data versioning |

**✅ AutoML Demo (Optional):**

!pip install pycaret

from pycaret.datasets import get\_data

data = get\_data('diabetes')

from pycaret.classification import \*

setup(data, target='Class variable')

compare\_models()

**✅ Final Exercise Suggestions:**

1. Predict diabetes from patient info (sklearn.datasets.load\_diabetes)
2. Build a Streamlit app for salary prediction
3. Explore HuggingFace transformers for sentiment analysis

**📘 Resources for Continued Learning**

* 📚 Book: *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow* – Aurélien Géron
* 🧑‍💻 Course: [Google ML Crash Course](https://developers.google.com/machine-learning/crash-course)
* 🎓 Platform: [Kaggle](https://www.kaggle.com/), [Coursera ML by Andrew Ng](https://www.coursera.org/learn/machine-learning)
* 🛠 Tools: [Colab](https://colab.research.google.com/), [HuggingFace](https://huggingface.co/), [Streamlit](https://streamlit.io/)

**🍎 Supervised Learning**

**Think of it like a teacher helping you.**

📘 **Real-life example**: Your mom shows you many pictures of fruits and tells you the name of each. After seeing enough, you can tell the fruit name yourself.

🧠 **You learn with labels!**

📊 **Diagram**:

[Image of Apple] --> "Apple"

[Image of Banana] --> "Banana"

**🔍 Unsupervised Learning**

**No teacher, you figure it out yourself!**

📘 **Real-life example**: You have a box of buttons. You don’t know their names, but you group them by size or color.

🧠 **You group things without knowing names!**

📊 **Diagram**:

[Red Button] [Green Button] [Blue Button] → Grouped by Colour

**🧠 Semi-Supervised Learning**

**You get some help and figure out the rest.**

📘 **Real-life example**: You are shown 3 labelled fruits and 10 without labels. You use what you learned from 3 to guess the rest.

📊 **Diagram**:

[Labelled: Apple, Banana] + [Unlabelled: ???, ???] → Learn & Predict

**📘 ML Algorithms Overview**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Algorithm | Type | Usage Example | Sample Colab Link | GitHub Repo |
| Linear Regression | Supervised | Predict house prices | Colab | [Repo](https://github.com/scikit-learn/scikit-learn) |
| Logistic Regression | Supervised | Email spam detection | Colab | [Repo](https://github.com/scikit-learn/scikit-learn) |
| Decision Tree | Supervised | Student pass/fail prediction | Colab | [Repo](https://github.com/rasbt/python-machine-learning) |
| K-Nearest Neighbors | Supervised | Handwriting recognition | Colab | [Repo](https://github.com/scikit-learn/scikit-learn) |
| K-Means Clustering | Unsupervised | Customer segmentation | Colab | [Repo](https://github.com/scikit-learn/scikit-learn) |
| PCA | Unsupervised | Dimensionality reduction | Colab | [Repo](https://github.com/scikit-learn/scikit-learn) |
| Label Propagation | Semi-Supervised | NLP label inference | Colab | [Repo](https://github.com/scikit-learn/scikit-learn) |
| Random Forest | Supervised | Fraud detection | Colab | [Repo](https://github.com/scikit-learn/scikit-learn) |
| XGBoost | Supervised | Loan default prediction | Colab | [Repo](https://github.com/dmlc/xgboost) |

**💻 How to Run Code in Cloud and Locally**

**🔹 On Google Colab (Cloud)**

1. Go to <https://colab.research.google.com>
2. Click **"New Notebook"** or **upload** from GitHub
3. Copy and paste code from above or GitHub repo
4. Click "Runtime > Run All"

**🔸 On Local System**

1. Install Anaconda from <https://anaconda.com>
2. Open Jupyter Notebook or VSCode
3. Create a new Python file or notebook
4. Install packages:

pip install numpy pandas matplotlib scikit-learn

1. Copy-paste the code and run cell by cell

**🔧 Platform for Hands-On**

* **Google Colab** - <https://colab.research.google.com>
* Alternative: **Jupyter via Anaconda** - <https://www.anaconda.com>

| **Algorithm** | **Type** | **Real-Life Example** | **Use Case / Domain** | **GitHub Repo / Notebook** |
| --- | --- | --- | --- | --- |
| **Linear Regression** | Supervised | Predicting house price from area and number of rooms | Price Prediction | [Linear Regression Example](https://github.com/krishnaik06/Linear-Regression) |
| **Logistic Regression** | Supervised | Predicting if an email is spam or not | Binary Classification | [Logistic Regression](https://github.com/krishnaik06/Logistic-Regression) |
| **Decision Tree** | Supervised | Making decisions like whether to play outside based on weather | Rules-Based Decision Making | [Decision Tree](https://github.com/krishnaik06/Decision-Tree) |
| **Random Forest** | Supervised | Voting among multiple opinions (ensemble of trees) | Classification, Regression | [Random Forest](https://github.com/krishnaik06/Random-Forest) |
| **Support Vector Machine** | Supervised | Separating apples and oranges by size and color | Image Classification | [SVM Demo](https://github.com/krishnaik06/SVM) |
| **K-Nearest Neighbors (KNN)** | Supervised | Recommending products based on similar users | Recommendation Engines | [KNN](https://github.com/krishnaik06/KNN) |
| **Naive Bayes** | Supervised | Predicting if a message is spam based on word probabilities | Spam Filtering, Text Mining | [Naive Bayes](https://github.com/krishnaik06/Naive-Bayes) |
| **K-Means Clustering** | Unsupervised | Grouping customers by behavior without knowing segments | Customer Segmentation | [K-Means](https://github.com/krishnaik06/K-Means-Clustering) |
| **Principal Component Analysis (PCA)** | Unsupervised | Compressing photo without much quality loss | Dimensionality Reduction | [PCA Example](https://github.com/krishnaik06/PCA-Example) |
| **DBSCAN** | Unsupervised | Detecting traffic patterns/clusters on road sensors | Anomaly Detection | [DBSCAN](https://github.com/abhishekkrthakur/DBSCAN) |
| **Label Propagation** | Semi-Supervised | Learning from a few labeled fruits to label others | NLP, Image Tagging | [Label Propagation](https://github.com/scikit-learn/scikit-learn/blob/main/examples/semi_supervised/plot_label_propagation_structure.py) |
| **XGBoost** | Supervised | Winning many Kaggle competitions with better accuracy | Tabular Data ML | [XGBoost](https://github.com/dmlc/xgboost) |
| **t-SNE** | Unsupervised | Visualizing high-dimensional data like face embeddings | Data Visualization | [t-SNE Example](https://github.com/oreillymedia/introduction_to_ml_with_python/blob/master/06_unsupervised_learning/tsne_example.py) |

**🔍 Example: Real-Life Mapping**

| **Scenario** | **Algorithm You Might Use** | **Why?** |
| --- | --- | --- |
| Predicting salary from experience | Linear Regression | Continuous output prediction |
| Detecting fraudulent transactions | Random Forest / Isolation Forest | Handles non-linear, complex patterns |
| Sorting users into behavior groups | K-Means Clustering | Discovering hidden group patterns (unsupervised) |
| Email classification (spam/ham) | Naive Bayes / Logistic Regression | Simple & effective for text classification |
| Visualizing customer clusters | t-SNE / PCA | High-dimensional data reduction |