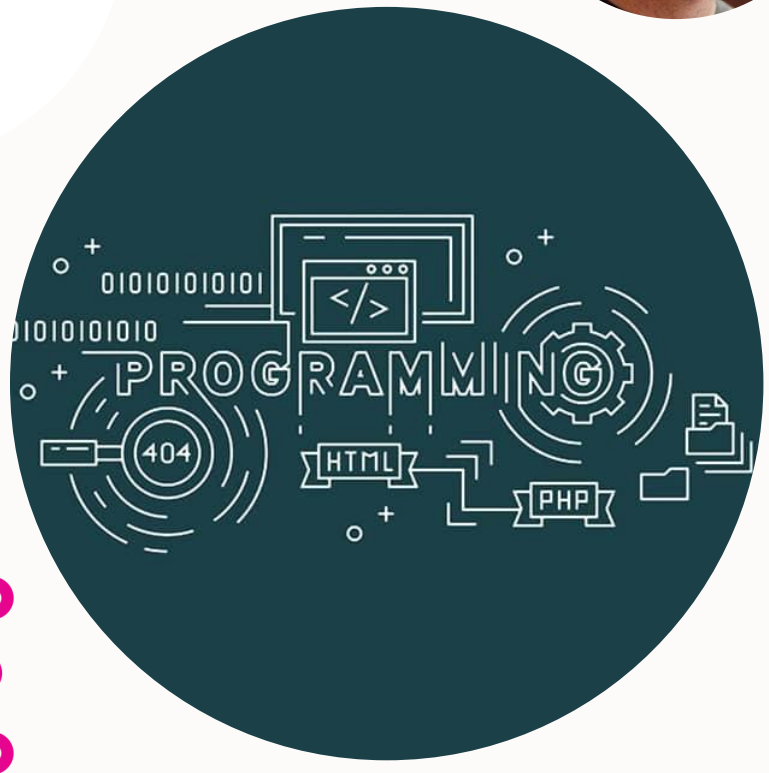
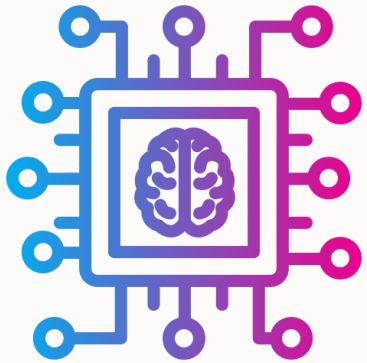
Guido Van Rossum



Machine learning and Data analytics using Python



PRAMOD NAIK

Module-1: Introduction to Machine Learning and Python:

Introduction to Machine Learning: Definition and importance of machine learning, Types of machine learning: Supervised, unsupervised, and reinforcement learning, Applications of machine learning in various domains.

Python for Data Analysis: Introduction to Python programming, Python libraries for data analysis: NumPy, Pandas, Matplotlib, Data manipulation and visualization using Pandas and Matplotlib.

Data Preprocessing: Data cleaning and transformation, Handling missing values and outliers, Feature scaling and normalization.

PRAMOD NAIK

Machine Learning (ML)

ML is an application of AI that provides **Systems or Models** the ability to automatically learn **and improve from experience without being explicitly programmed and** without human intervention .

Machine Learning (ML) is a **subfield** of artificial intelligence (**AI**) that focuses on the development of algorithms and models that enable computers to learn and make predictions or decisions without being explicitly programmed for a specific task.

The primary aim of machine learning is to allow the computers learn automatically without human intervention or assistance and adjust actions accordingly.

Examples:

1. The Self-Driving Car
2. The Spam Filter
3. Music Recommender
4. Personal Assistance (Alexa)

Definition:

According to **Tom Mitchell** Machine Learning **is the study of algorithms that improve** their **performance** P at some task T with experience E. A well-defined learning task for a system is given by <P, T, E>.

AI, ML and DL:

AI: Creation of Intelligent Machines that **think and act like** a Humans Ex: **Robots**.

ML: Approach for Computers to **learn without being explicitly programmed.**

DL: Training of Machines to think like a **human brains-** Application of Artificial Neural Networks.

Traditional Programming V/S Machine Learning :

Traditional Programming:
In traditional programming, a computer engineer writes a series of directions that instruct a computer how to transform input data into a desired output. Instructions are mostly based on an IF-THEN structure: when certain conditions are met, the program executes a specific action.

Machine Learning:
Machine learning, on the other hand, is an **automated process** that enables machines to solve problems **with little or no human input**, and take actions based on past observations.

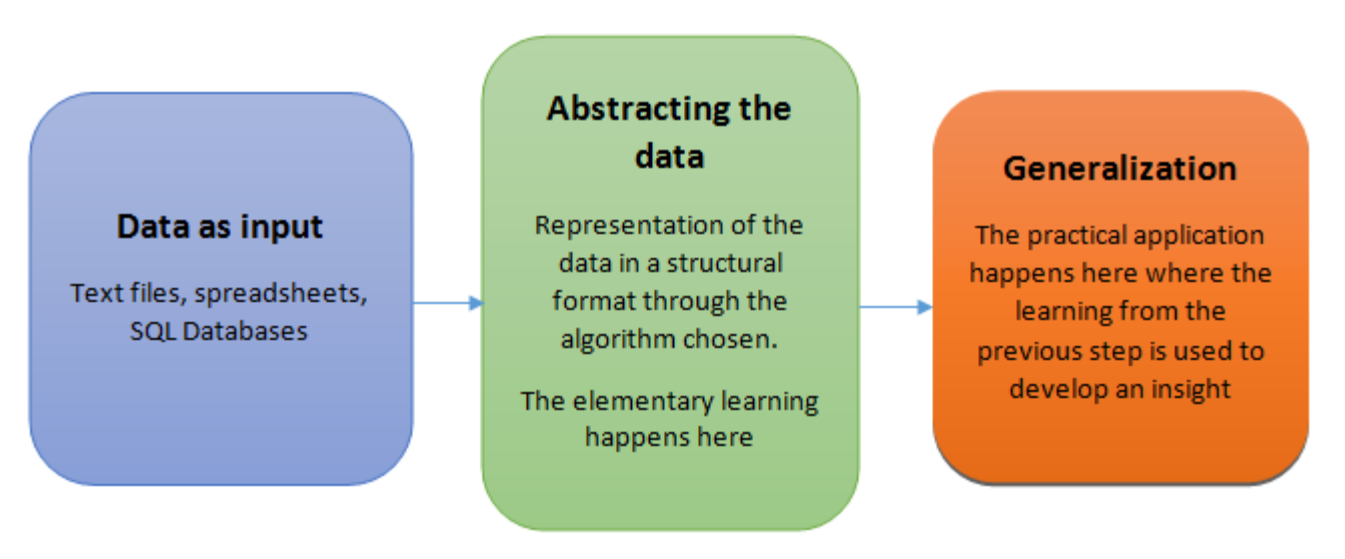
Basic machine learning process:

The basic machine learning process can be divided into **three** parts.

1. **Data Input:** Past data or information is utilized as a basis for future decision-making

2. **Abstraction (**Creating a **Internal representation of the data** by capturing the essential information)**:** The input data is represented in a broader way through the underlying algorithm

3. **Generalization:** Generalization refers to the ability of a **model to perform well on new, unseen data that it hasn't been trained on**.



Data Input:

1. This is the **foundation** of any machine learning task. Data can come from **various sources** like sensors, databases, logs, text files, and images.

2. The data should be **relevant** to the problem you're trying to solve, **diverse enough** to capture different scenarios, and **accurate** to **avoid misleading the model.**

3. Often, data needs **pre-processing** before feeding it into the model. This can involve cleaning, formatting, feature engineering (creating new features), and handling missing values.

Note:
The **quality** and **quantity** of the input data significantly impact the effectiveness of the machine learning model.

Abstraction: (Training the model):

- During the machine learning process, knowledge is fed in the form of input data.
- This is where the magic happens! The chosen machine learning algorithm analyzes the **processed** data, searching for patterns, relationships, and underlying structures.
- The algorithm creates an **internal representation of the data**, capturing the essential information it needs for prediction. This abstraction can be mathematical equations, rules, decision trees, or even complex neural networks.
- Think of it like **summarizing a book** into key themes and plot points; the abstraction captures the **essence** or **core** of the information without needing all the details.

The choice of the model used to solve a specific learning problem **is a human task**. The decision related to the choice of model is taken based on multiple aspects, some of which are listed below:

1. **The type of problem to be solved:** Whether the problem is related to prediction or forecast , analysis of trend, understanding the different segments or groups of objects, etc.

2. **Nature of the input data:** How exhaustive the input data is, whether the data has no values for many fields, the **data types** etc.

3. **Domain of the problem:** If the problem is in a business critical domain with a high rate of data input and need for immediate inference, e.g. fraud detection problem in banking domain.

Generalization:

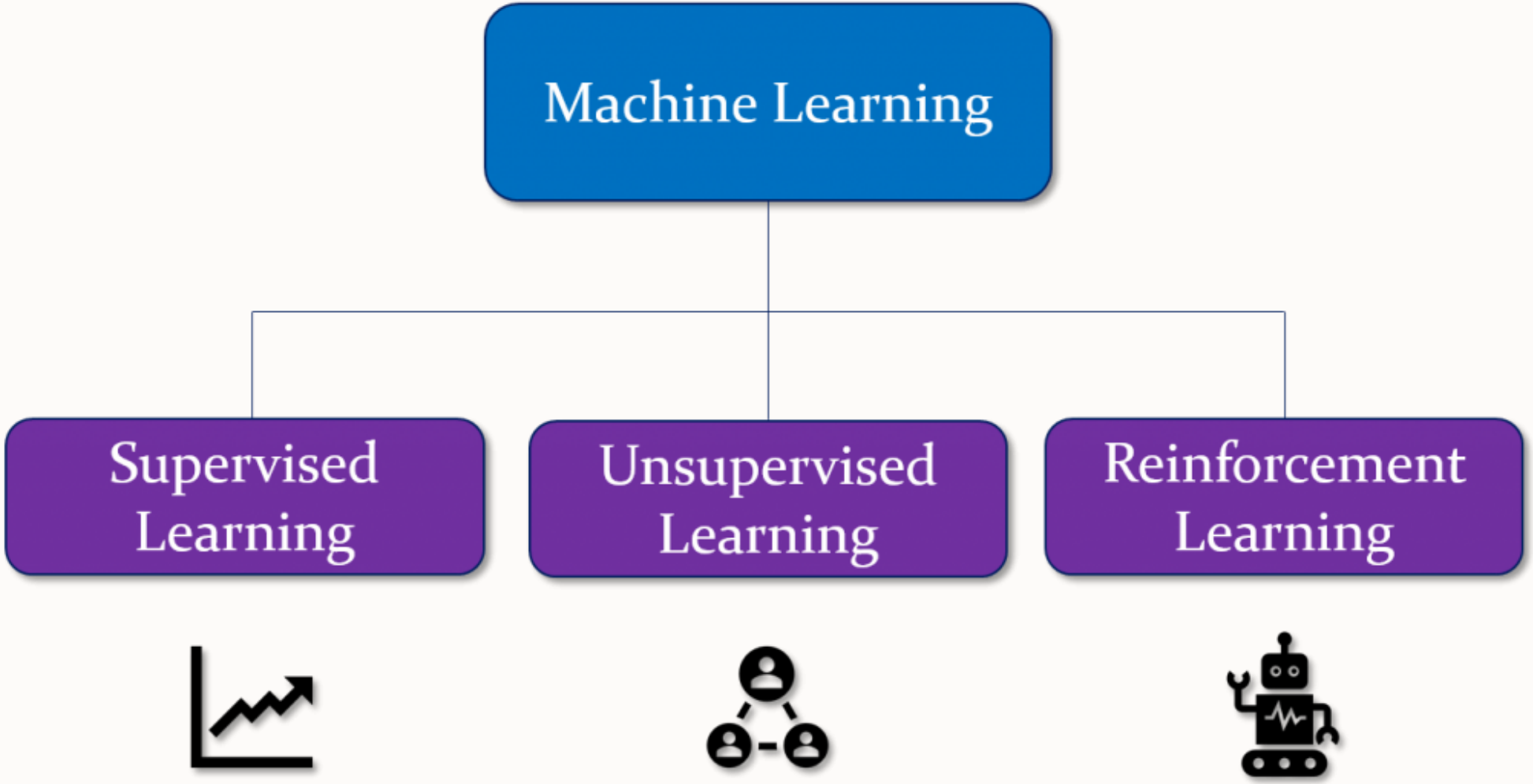
Generalization refers to the ability of a **model to perform well on new, unseen data that it hasn't been trained on**. The ultimate goal of a machine learning model is not just to memorize the training data but to learn underlying patterns and relationships that can be applied to new, unseen data in a meaningful way.

The first part of machine learning process is abstraction i.e. **abstract the knowledge** which comes as input data in the form of a model. However, this abstraction process, or **more popularly training the model**, is just one part of machine learning. The other key part is to **tune up** the abstracted knowledge to a form which can be used to take future decisions. This is achieved as a part of generalization.

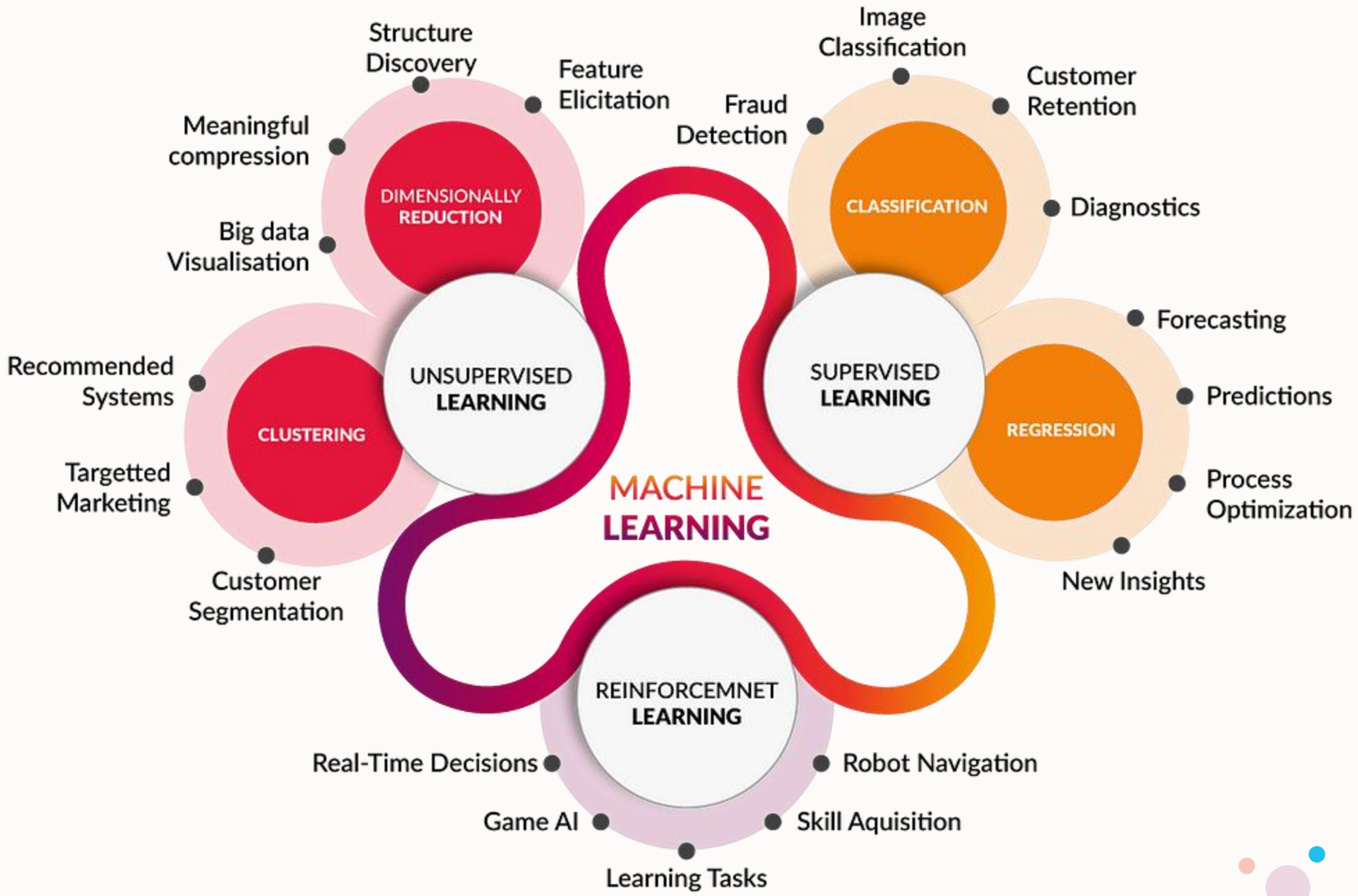
This is where the **model's power lies**. Based on the abstracted representation, the model goes beyond the specific data it was trained on and **learns to make general predictions for unseen data**.

This part is quite difficult to achieve. This is because the model is trained based on a finite set of data, which may possess a limited set of characteristics. But when we want to apply the model to take decision on a set of unknown data, usually termed as **test data**.

Types of Machine Learning



PRAMOD NAIK


MACHINE LEARNING
UNSUPERVISED LEARNING
SUPERVISED LEARNING
REINFORCEMNET LEARNING
DIMENSIONALLY REDUCTION
CLUSTERING
CLASSIFICATION
REGRESSION
Structure Discovery
Feature Elicitation
Meaningful compression
Big data Visualisation
Image Classification
Customer Retention
Fraud Detection
Diagnostics
Recommended Systems
Targetted Marketing
Customer Segmentation
Forecasting
Predictions
Process Optimization
New Insights
Real-Time Decisions
Robot Navigation
Game AI
Skill Aquisition
Learning Tasks

PRAMOD NAIK

TYPES OF MACHINE LEARNING

Supervised Machine Learning

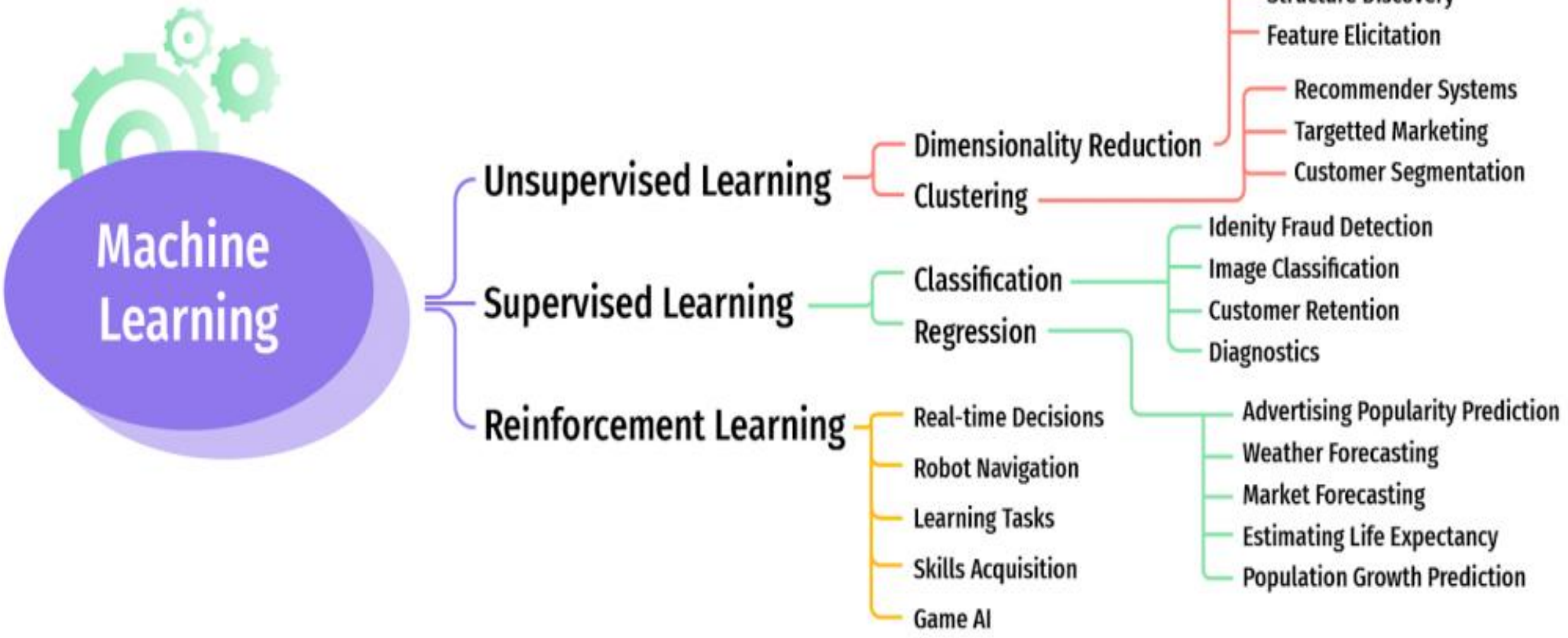
Unsupervised Machine Learning

Semi-Supervised Learning

Reinforcement Learning

PRAMOD NAIK

ML and It's Application



PRAMOD NAIK

Supervised learning:

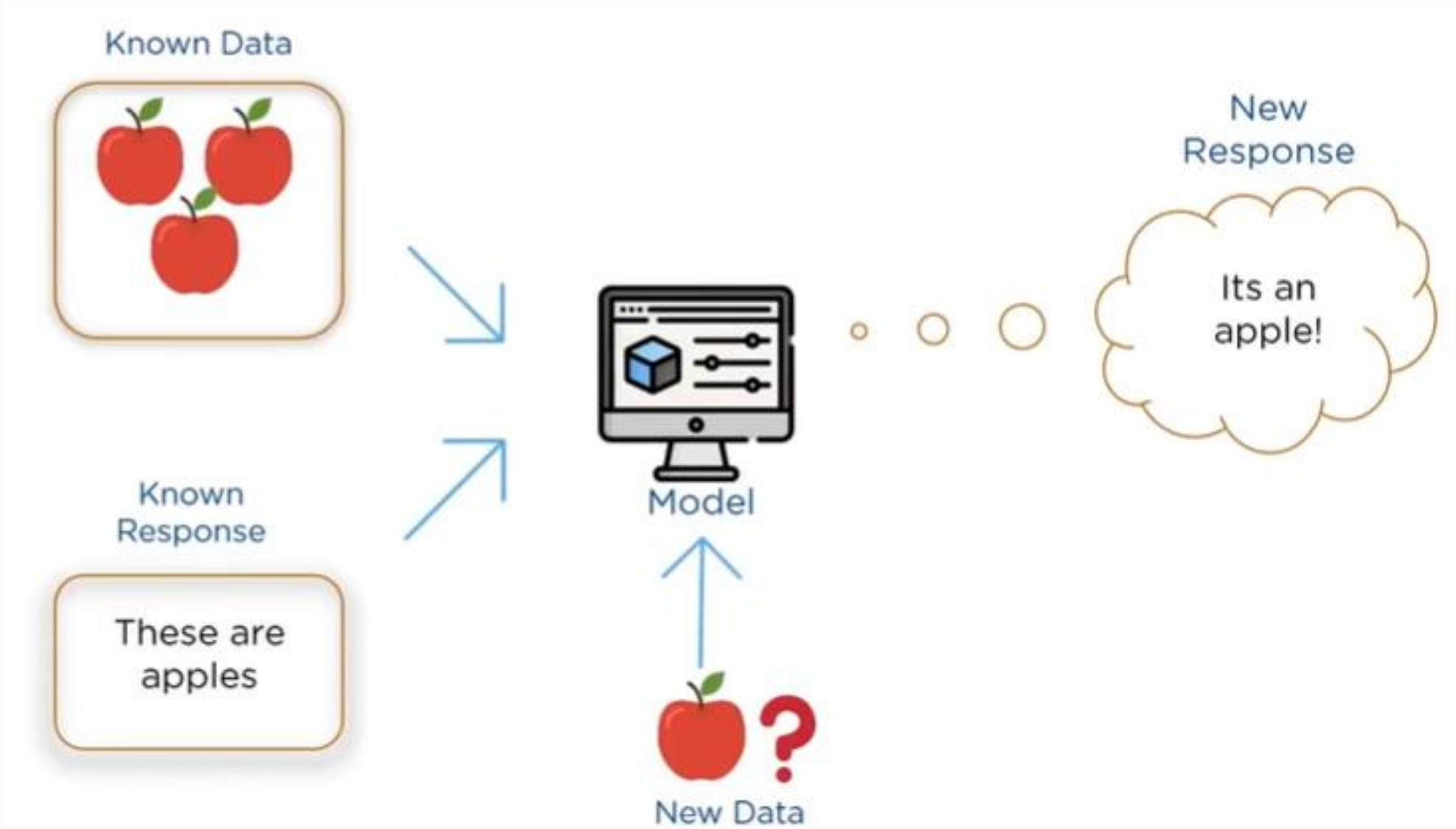
Supervised learning is the types of machine learning in which machines are trained using well "**labelled**" **training data**, and on basis of that data, machines predict the output. The labelled data means some input data is already **tagged** or **labelled** with the **correct output.**

In supervised learning, **the training data provided to the machines work as the supervisor** that teaches the machines to predict the output correctly. It applies the same concept as a student learns in the **supervision of the teacher**. Supervised learning is a process of providing input data as well as correct output data to the machine learning model. The aim of a supervised learning algorithm is to find a mapping function to **map** the **input variable(x)** with the **output variable(y).**

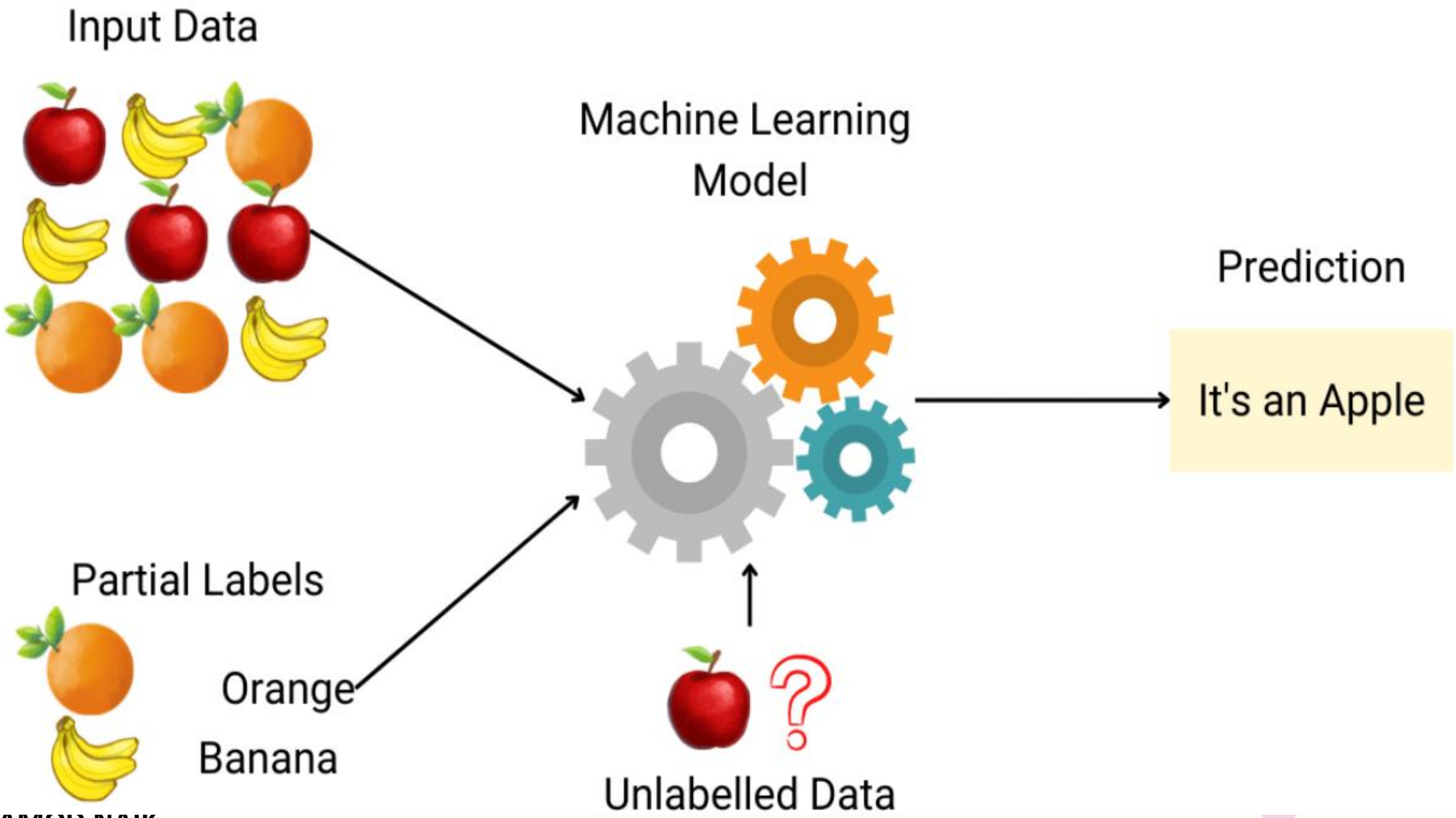
So, the goal of supervised learning is to learn a **mapping from inputs to outputs**, so that the model can make accurate predictions on **new**, **unseen data**.

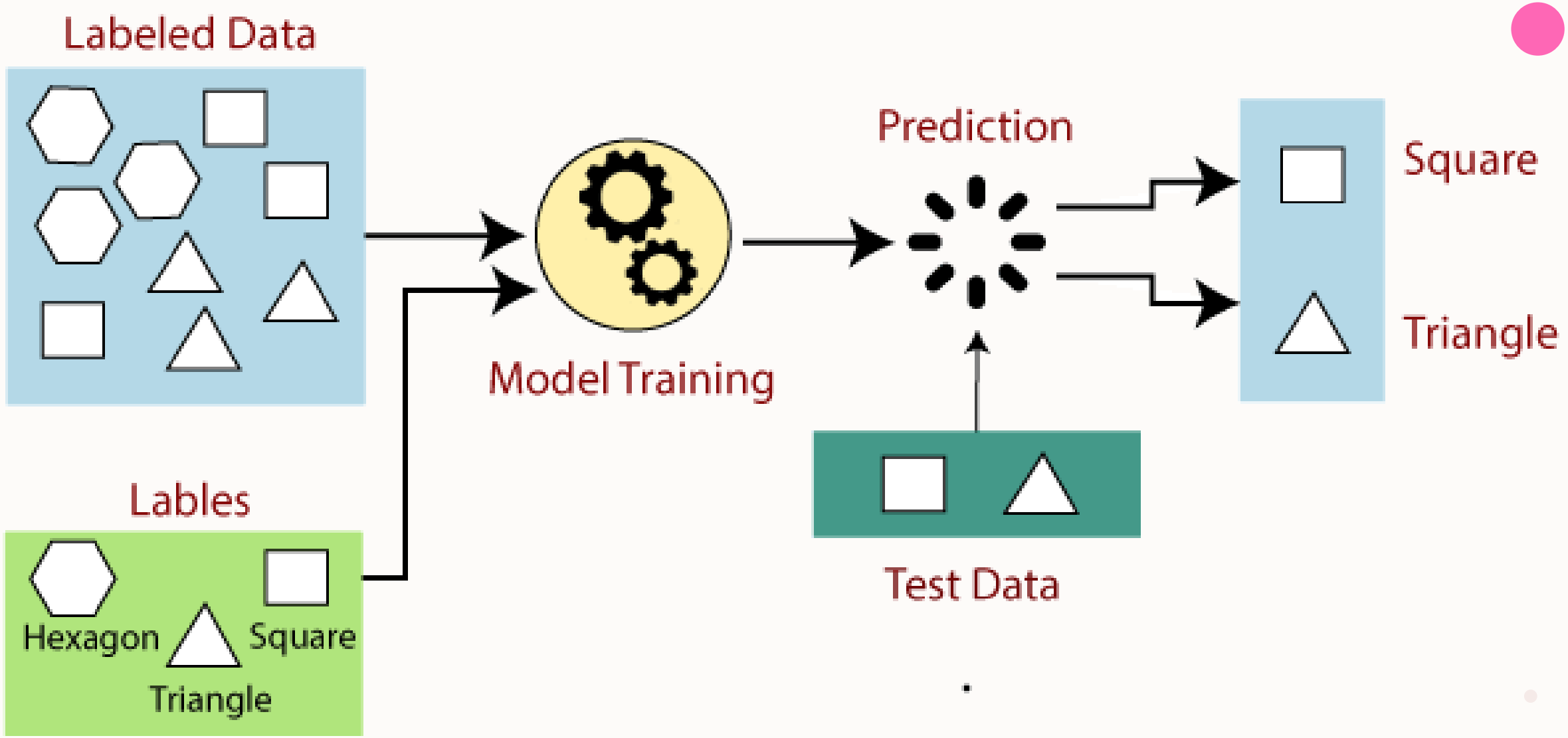
Note:

In supervised learning, models are trained using labelled dataset, where the model learns about each type of data. Once the **training process** is completed, the model is **tested** on the basis of **test data** (a subset of the training set), and then it predicts the output.


Known Data
New Response
Its an apple!
Known Response
These are apples
Model
New Data

PRAMOD NAIK


Input Data
Machine Learning Model
Prediction
It's an Apple
Partial Labels
Orange
Banana
Unlabelled Data


Labeled Data
Lables
Hexagon
Square
Triangle
Model Training
Prediction
Test Data
Square
Triangle

PRAMOD NAIK

Explanation:

Suppose we have a **dataset** of different types of **shapes** which includes **square, rectangle, triangle, and Polygon**. Now the first step is that we need to train the model for each shape.

1. If the given shape has **four sides**, and all the sides are equal, then it will be labelled as a Square.
2. If the given shape has **three sides**, then it will be labelled as a triangle.
3. If the given shape has **six equal sides** then it will be labelled as hexagon.

Now, after training, we test our model using the test set, and the task of the model is to identify the shape. The machine is already trained on all types of shapes, and when it finds a new shape, it classifies the shape on the bases of a number of sides, and predicts the output

Example-3: Predicting Housing Prices

Let's consider a common example of supervised learning: predicting housing prices based on various features.

1.Input Features:

1. Size of the house (in square feet)
2. Number of bedrooms
3. Number of bathrooms
4. Distance to the city center
5. Presence of nearby amenities (e.g., schools, parks)

2.Labels (Outputs):

- Sale price of the house

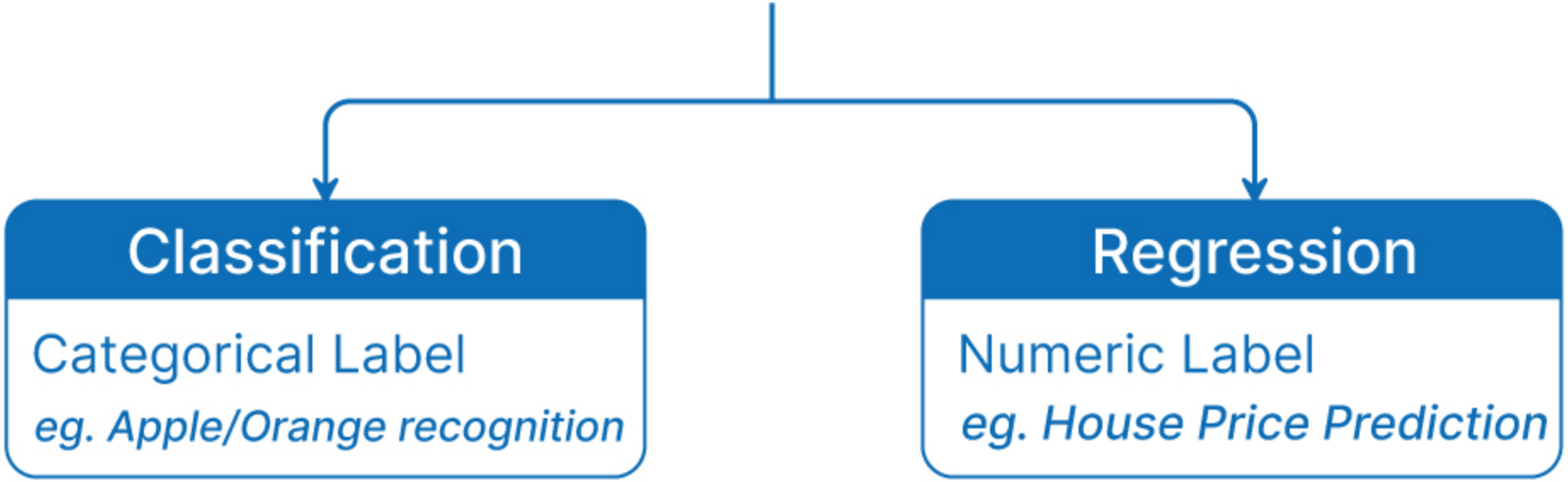
3.Training Dataset:

- A dataset containing historical examples of houses with features and their corresponding sale prices.

4.Model:

- A supervised learning algorithm, such as linear regression, is chosen to learn the relationship between the input features and the sale prices.

TYPE OF
SUPERVISED LEARNING



PRAMOD NAIK

1. Classification:

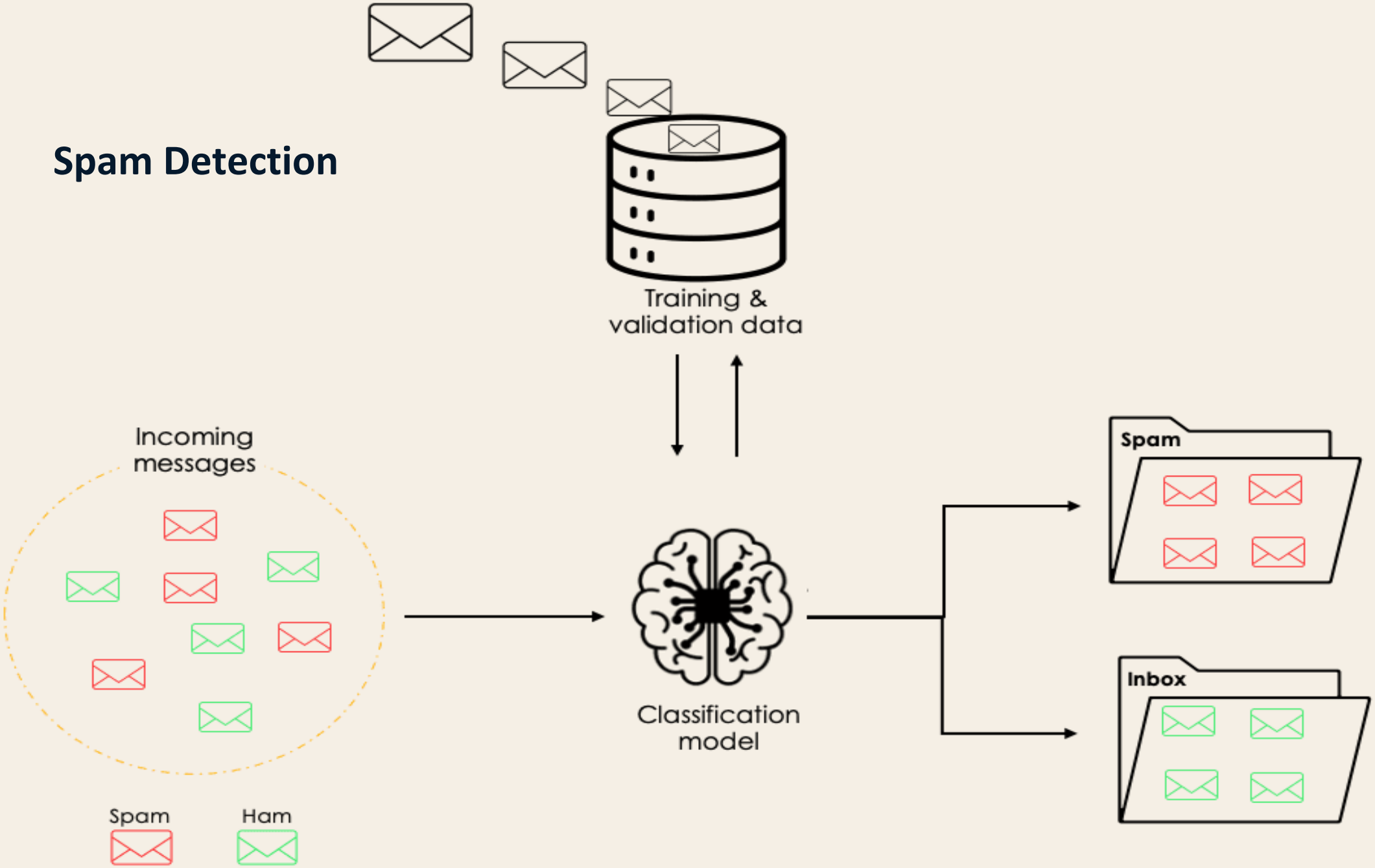
Classification is a type of supervised machine learning task where the goal is to **predict the category or class** that a new instance or observation belongs to, **on the basis of training data**. The output variable in classification is **discrete** and represents different classes or labels.

In Classification, a program learns from the given dataset or observations and then classifies new observation into a number of classes or groups. Such as, Yes or No, 0 or 1, Spam or Not Spam, cat or dog, etc. Classes can be called as targets/labels or categories.

Unlike regression, **the output variable of Classification is a category**, not a value, such as "Green or Blue", "fruit or animal", etc. Since the Classification algorithm is a Supervised learning technique, hence it takes labeled input data, which means it contains input with the corresponding output. The best example of an ML classification algorithm is **Email Spam Detector.**

The main goal of the Classification algorithm is to identify the category of a given dataset, and these algorithms are mainly used to predict the output for the categorical data.

Spam Detection



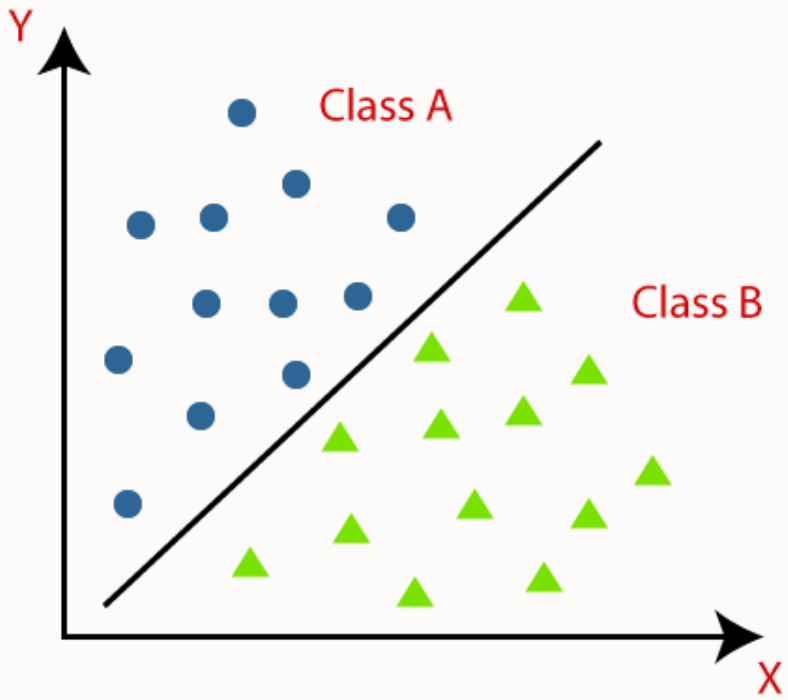
Example-2:

Classification algorithms can be better understood using the below diagram. In the below diagram, there are two classes, class A and Class B. These classes have features that are similar to each other and dissimilar to other classes.

The algorithm which implements the classification on a dataset is known as a **classifier**.

There are two types of Classifications:

1. Binary Classification
2. Multi-class Classifier



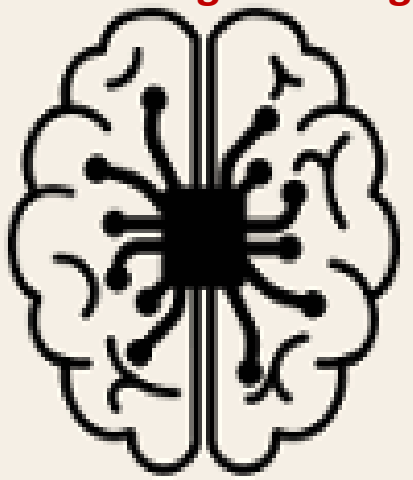
Truck

Binary Classification

In a binary classification task, the goal is to classify the input data **into two mutually exclusive categories**. The training data in such a situation is labeled in a binary format: **true** and **false**; **positive** and **negative**; O and 1; spam and not spam, male or female etc. depending on the problem being tackled.

For instance, we might want to detect whether a **given image is a truck or a boat**.

In Binary classification task, "**Mutually Exclusive**" means that the two categories or classes into which the input data is classified are distinct and cannot occur simultaneously for a single instance of input data.
An email cannot be both "spam" and "not spam" at the same time.

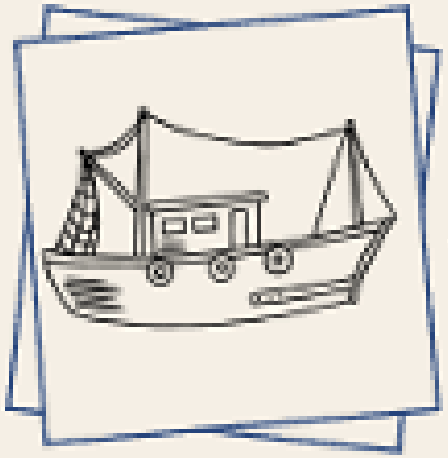


Machine Learning Model

Classification

Boat

Boat



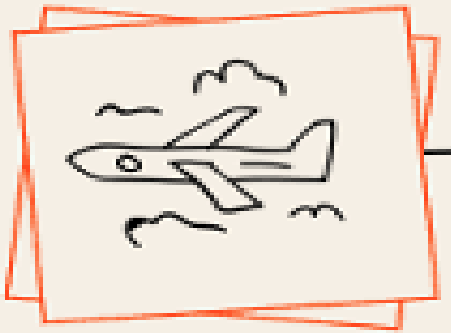
Note: If the classification problem has only two possible outcomes, then it is called as Binary Classifier.

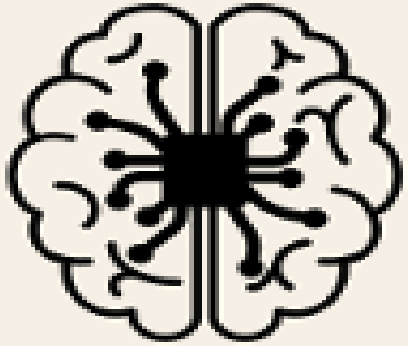
Zoumana KEITA

Truck

If a classification problem has more than two outcomes, then it is called as Multi-class Classifier.
Example: Classifications of types of crops, Classification of types of music.

Plane



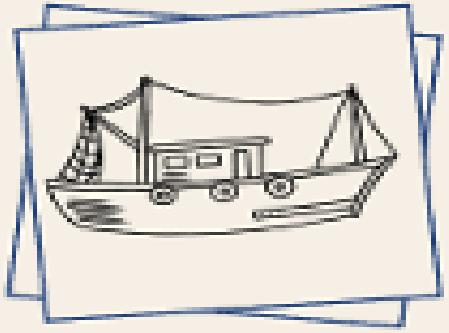


Machine Learning Model

Classification

Plane

Boat



Multi-Class Classification:
The multi-class classification, on the other hand, has at least two mutually exclusive class labels, where the goal is to predict to which class a given input example belongs to. In the following case, the model correctly classified the image to be a plane.

Regression :

Regression is a type of supervised machine learning task where the goal is to predict a **continuous numerical value** or outcome based on input features.

Regression is a type of supervised machine learning where algorithms **learn** from the data to predict continuous values such as sales, salary, weight, or temperature.

Note:

In the context of regression in machine learning, a continuous numerical value refers to an outcome or target variable that can take on an infinite number of values within a specific range. In case of predicting a person's age, age is considered a continuous numerical value because it can theoretically take on any value within a certain range (for example, from 0 to 100+ years). **There are no gaps or intervals between possible ages**, and age can be expressed as a decimal or fraction if necessary (e.g., 25.5 years).

It is a variable that can have any real number value, and there are no distinct categories or classes. The term "continuous" implies that the variable can vary over a continuous range, and there are no gaps or interruptions in the possible values it can take. In contrast, in a classification task, the target variable would be a **discrete set of categories or classes.**

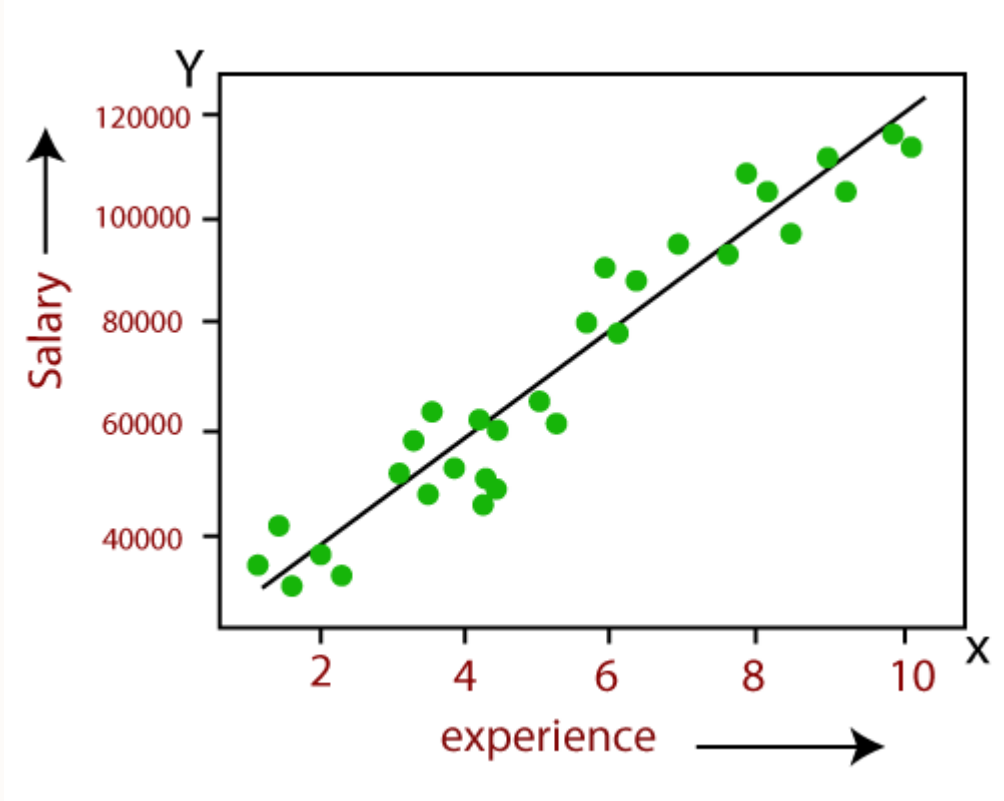
Note:

Here, “discrete set of categories or classes" refers to a set of distinct and separate values that a variable can take

Example:

Predicting the **salary** of an employee on the basis of the **year of experience**.

Linear Regression:



PRAMOD NAIK

Example:

1. **Predicting age of a person:** Given certain features or attributes of a person, such as **height, weight, gender**, and other relevant factors, the task is to predict the person's age in years.

2. **Predicting the price of houses based on their features:**

In real estate markets, house prices can vary continuously based on factors such as location, size, amenities, market conditions, and other features. The House prices can range from a few thousand dollars for smaller properties in certain areas to millions of dollars for luxury properties in prime locations.

Important to Note:

Let us consider two examples, say ‘predicting whether a tumour is malignant or benign’ and ‘price prediction in the domain of real estate? **Are these two problems same in nature?**

The answer is ‘no’ It is true that both of them are problems related to prediction. However, for tumour prediction, we are trying to predict which **category or class**, i.e. ‘malignant’ or ‘benign’, an unknown input data related to tumour belongs to.

In the other case, that is, for price prediction, we are trying to predict an **absolute value and not a class.** When we are trying to predict a **categorical or nominal variable**, the problem is known as a classification problem. A classification problem is one where the output variable is a category such as ‘red’ or ‘blue’ or ‘malignant tumour’ or ‘benign tumour’ etc. Whereas when we are trying to predict a numerical variable such as ‘price’, ‘weight’, etc. the problem falls under the category of **regression**.

Regression analysis :

Regression analysis is a statistical method to **model the relationship between a dependent (target) and independent (predictor) variables with one or more independent variables**. More specifically, Regression analysis helps us to understand how the value of the dependent variable is changing corresponding to an independent variable when other independent variables are held fixed. It predicts continuous/real values such as temperature, age, salary, price etc. Regression is a supervised learning which helps in finding the correlation between variables and enables us to predict the continuous output variable based on the one or more predictor variables.

It is mainly used for prediction, forecasting, time series modeling, and determining the causal-effect relationship between variables.

Unsupervised Learning:

Unsupervised Machine learning is a type of machine learning where the algorithm is trained on **unlabeled data**, and the **objective is to find patterns, relationships, or structures within the data** without explicit guidance or labeled outcomes. i.e Unsupervised learning is a method we use to **group data** when no **labels are present**.

As the **name suggests**, unsupervised learning is a machine learning technique in which models are **not supervised using training dataset**. Instead, **models itself find** the hidden patterns and insights from the given data. It can be compared to **learning which takes place in the human brain while learning new things**.

It can be defined as: **"**Unsupervised learning is a type of machine learning in which models are trained using unlabeled dataset and are allowed to act on that data without any supervision.**"**

Unsupervised learning cannot be directly applied to a regression or classification problem because unlike supervised learning, we have the input data but no corresponding output data.

Example:

Suppose the unsupervised learning algorithm is given an input dataset containing **images of different types of cats and dogs**. The algorithm is never trained upon the given dataset, which means it does not have any idea about the features of the dataset. The task of the unsupervised learning algorithm is to identify the image features on their own. Unsupervised learning algorithm will perform this task **by clustering the image dataset into the groups according to similarities between images**.

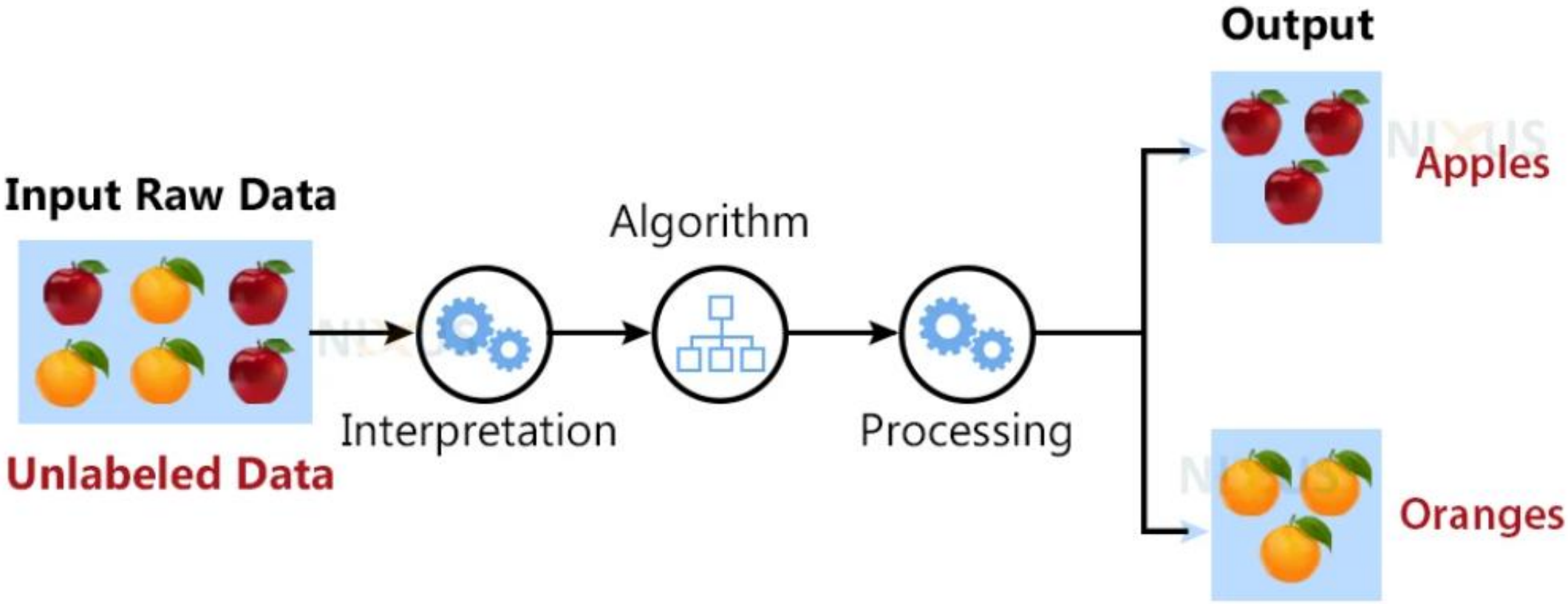
Why use Unsupervised Learning?

1. Unsupervised learning is helpful for finding useful insights from the data.
2. Unsupervised learning is much similar as a **human learns to think by their own experiences**, which makes it closer to the real AI.
3. Unsupervised learning works on unlabeled and uncategorized data which make unsupervised learning more important.
4. In real-world, we do not always have input data with the corresponding output so to solve such cases, we need unsupervised learning.
5. **Semi-supervised Learning:** Unsupervised ML techniques can also be combined with supervised ML in semi-supervised learning approaches. In this paradigm, a small amount of labeled data is augmented with a larger amount of unlabeled data to improve the performance of supervised models.

Note:

Unsupervised machine learning (ML) works differently from supervised ML because **it doesn't rely on labeled data to learn patterns or make predictions**. Instead, unsupervised ML algorithms focus on finding hidden structure or patterns within the data without explicit guidance from predefined labels.

Unsupervised Machine Learning



PRAMOD NAIK

Here, we have taken an unlabeled input data, which means it is not categorized and corresponding outputs are also not given. Now, this unlabeled input data is fed to the machine learning model in order to train it. Firstly, it will interpret the raw data to find the hidden patterns from the data and then will apply suitable algorithms such as k-means clustering, Decision tree, etc. Once it applies the suitable algorithm, the algorithm divides the data objects into groups according to the similarities and dissimilarities between the objects.

Working of Unsupervised learning models:

1. We feed the model data with no categories or outputs for training
2. Model interprets raw data to identify hidden patterns
3. Depending on data, we use suitable algorithms
4. Algorithm groups data

Disadvantages of Unsupervised Learning:

- 1. Unsupervised learning is fundamentally **more difficult** than supervised learning since it lacks comparable results.
  2. The outcome of an unsupervised learning approach may be **less accurate** since the input data does not have labels and algorithms do not know the precise output in advance.
  3. We **need humans to validate the results** of unsupervised learning models which convolutes the process.
  4. The operations involved **require a high level of computational power and also take up a lot of time**.

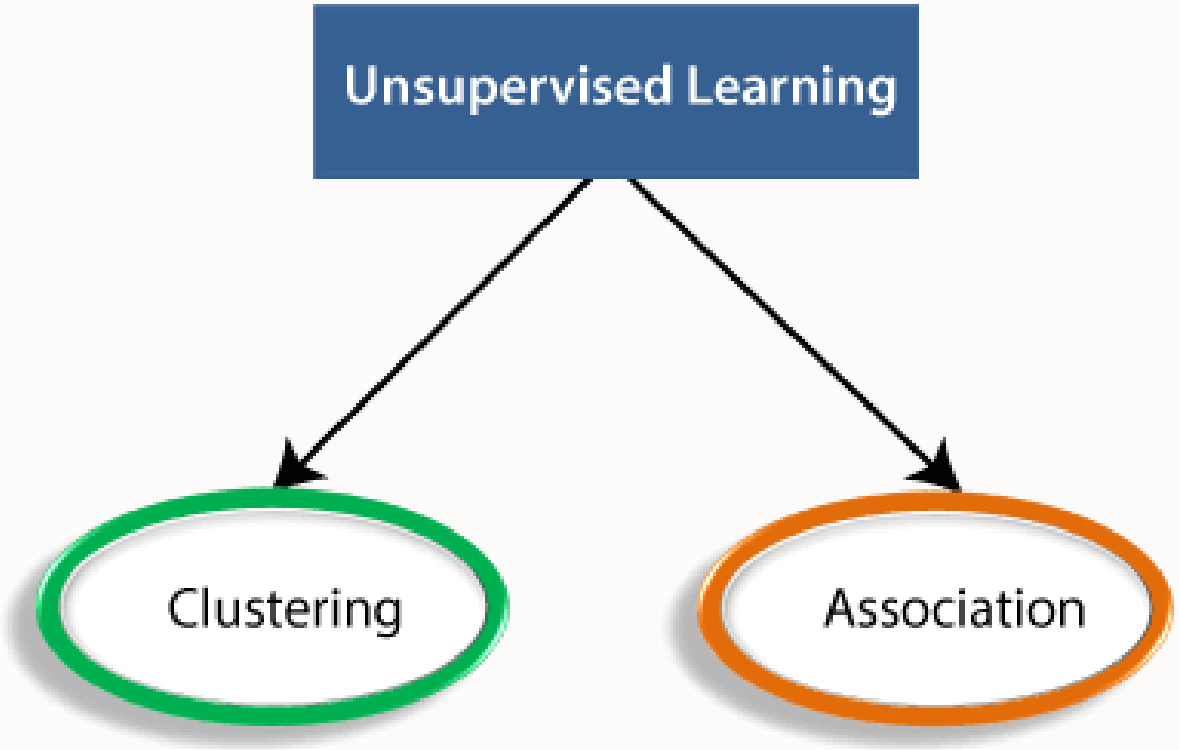
Alternative to Unsupervised Learning:

Semi-supervised learning may be a viable alternative to unsupervised learning. It is a combination or even a min-and-match of both unsupervised and supervised learning methods. The primary benefit of this sort of training would be that it lowers errors.

For example, it will only cluster the unlabeled data that fits the clustering criteria, and it will categorize the output automatically once it has labels. This uses less computer power and takes less time.

Types of Unsupervised Learning:

1. Clustering
2. Association rules
3. Dimensionality reduction.

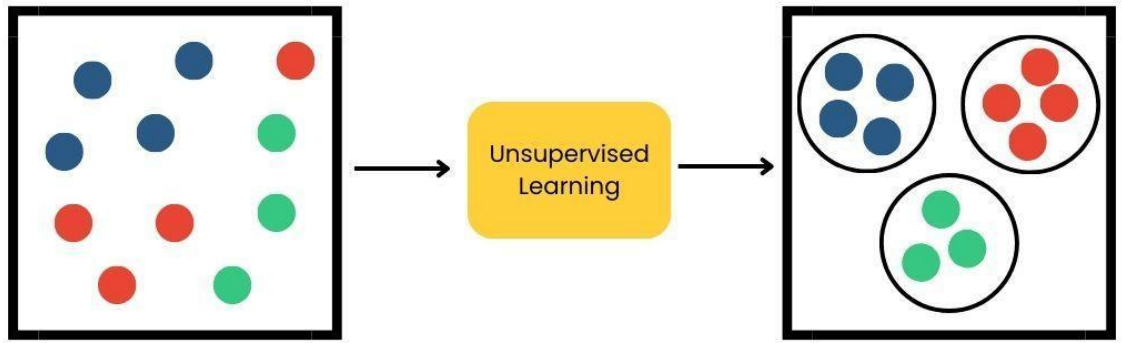


Clustering:

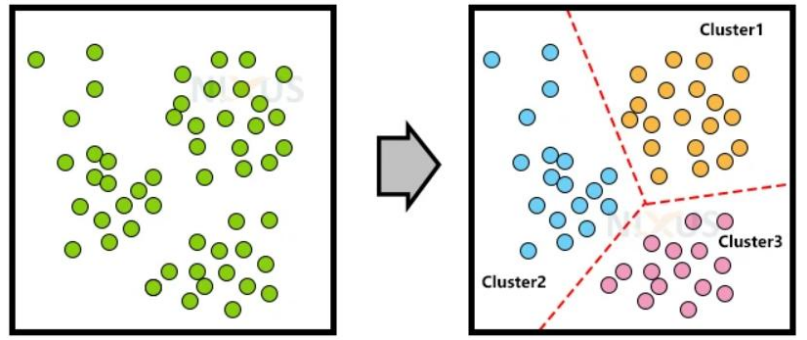
- Clustering methods involve **grouping untagged** data based on their similarities and differences. When two instances appear in different groups, we can infer they have dissimilar properties.

Clustering is a type of unsupervised learning, meaning that we do not need labeled data for clustering algorithms; this is one of the biggest advantages of clustering over other supervised learning like Classification.

Clustering is the process of **arranging** a group of objects in such a manner that the objects in the same group (which is referred to as a cluster) are more **similar** to **each other** than to the objects in any **other group**.



OR



Application of Clustering:

1. Customer Segmentation:

- **Objective:** Grouping customers based on similarities in purchasing behavior, demographics, or preferences.
- **Application:** Targeted marketing, personalized recommendations, and tailored customer experiences.

2. Document Clustering:

- **Objective:** Grouping similar documents based on content, topic, or language.
- **Application:** Document categorization, information retrieval, and organizing large document repositories.

3. Anomaly Detection:

- **Objective:** Identifying unusual patterns or outliers in a dataset.
- **Application:** Fraud detection in financial transactions, network intrusion detection, and fault detection in industrial systems.

4. Retail Market Basket Analysis:

- **Objective:** Identifying associations and patterns in the purchasing behavior of customers.
- **Application:** Recommender systems, optimizing product placements, and understanding customer buying habits.

5. E-commerce Product Bundling:

- **Objective:** Grouping products that are often purchased together.
- **Application:** Creating product bundles, improving cross-selling strategies, and enhancing the user shopping experience.

Association Rule:

An association rule is an unsupervised learning method which is used **for finding the relationships between variables in the large database**. It determines the set of items that occurs together in the dataset. Association rule makes marketing strategy more effective. Such as people who buy X item (suppose a bread) are also tend to purchase Y (Butter/Jam) item. A typical example of Association rule is **Market Basket Analysis(MBA)**.

We typically see association rule mining used for market basket analysis: this is a data mining technique retailers use to gain a better understanding of customer purchasing patterns based on the relationships between various products.

So Association is the process of discovering interesting relationships, associations, or patterns within a dataset. This type of analysis is often applied to **transactional data**, where the goal is to identify associations between items or events that frequently co-occur. Association rules are used to express these relationships, and they help reveal hidden connections in the data.

For example, if a customer buys bread, he most likely can also buy butter, eggs, or milk, so these products are stored within a shelf or mostly nearby.

Customer 1

Customer 2

Customer 3

Customer n

PRAMOD NAIK

3. Dimensionality Reduction:

Dimensionality reduction is a technique used in unsupervised machine learning **to reduce the number of input features or variables in a dataset while preserving as much of its essential information**. In other words, it is a process of **transforming high-dimensional data into a lower-dimensional space** that still preserves the essence of the original data. The primary goal is to overcome the **curse of dimensionality**, improve computational efficiency, and sometimes enhance the interpretability of the data. Two common methods for dimensionality reduction are **Principal Component Analysis (PCA) and t-Distributed Stochastic Neighbor Embedding (t-SNE).**

In machine learning, high-dimensional data refers to data with a large number of features or variables. The curse of dimensionality is a common problem in machine learning, where the performance of the model deteriorates or drops as the number of features increases. This is because the complexity of the model increases with the number of features, and it becomes more difficult to find a good solution. In addition, high-dimensional data can also lead to overfitting, where the model fits the training data too closely and does not generalize well to new data.

It's helpful to reduce the dimensionality of a dataset during EDA (Exploratory Data Analysis) to help visualize data: this is **because visualizing data in more than three dimensions is difficult**. From a data processing perspective, reducing the dimensionality of the data simplifies the modeling problem.

When more input features are being fed into the model, the model must learn a more complex approximation function. This phenomenon can be summed up by a saying called the "curse of dimensionality."

PRAMOD NAIK

Dimensionality reduction can help to mitigate these problems by reducing the complexity of the model and improving its generalization performance. There are two main approaches to dimensionality reduction: feature selection and feature extraction.

Feature Engineering:

1. Feature Selection:

Feature selection involves **selecting a subset of the original features** that are most relevant to the problem at hand. The goal is to reduce the dimensionality of the dataset while retaining the most important features. There are several methods for feature selection, including filter methods, wrapper methods, and embedded methods. Filter methods rank the features based on their relevance to the target variable, wrapper methods use the model performance as the criteria for selecting features, and embedded methods combine feature selection with the model training process.

2. Feature Extraction:

Feature extraction involves **creating new features by combining or transforming the original features**. The goal is to create a set of features that captures the essence of the original data in a lower-dimensional space. There are several methods for feature extraction, including principal component analysis (PCA), linear discriminant analysis (LDA), and t-distributed stochastic neighbor embedding (t-SNE). PCA is a popular technique that projects the original features onto a lower-dimensional space while preserving as much of the variance as possible.

Reinforcement Learning:

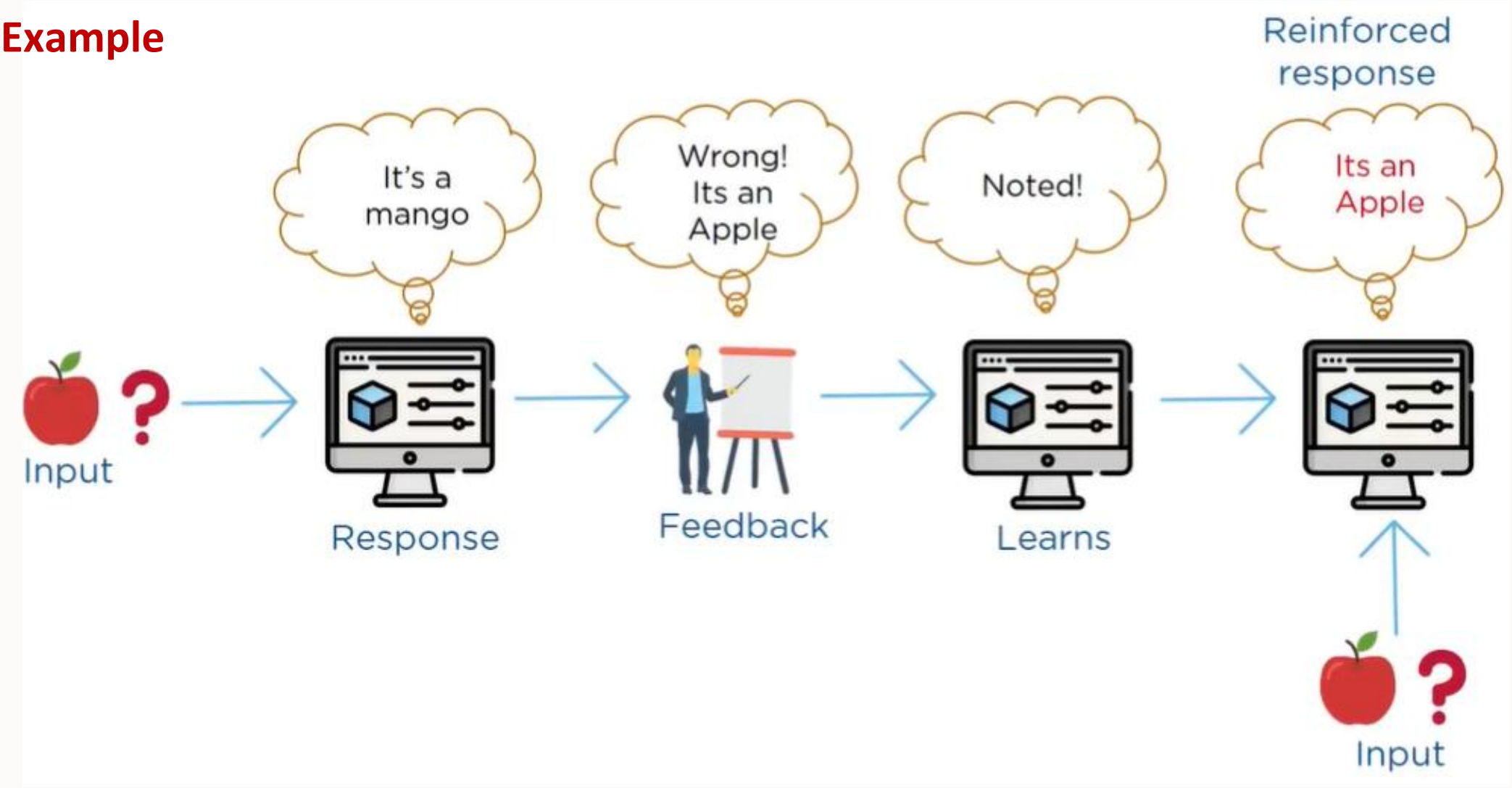
- Reinforcement Learning (RL) is a type of machine learning paradigm in which an **agent learns to make decisions by interacting with an environment**. The agent takes actions in the environment, and in **return, it receives feedback in the form of rewards or punishments**.

Reinforcement Learning is a **feedback-based Machine learning technique** in which an agent learns to behave in an environment by performing the actions and seeing the results of actions. For each good action, the agent gets **positive feedback**, and for each bad action, the agent gets **negative feedback or penalty**. In Reinforcement Learning, the agent learns automatically using feedbacks without any labeled data, unlike supervised learning Since there is no labeled data, so the agent is bound to learn by its experience only. RL solves a specific type of problem where decision making is sequential, and the goal is long-term, such as game-playing, robotics, etc. The agent interacts with the environment and explores it by itself. The primary goal of an agent in reinforcement learning is to **improve the performance by getting the maximum positive rewards.**

The agent learns with the process of **hit and trial**, and based on the experience, it learns to perform the task in a better way. Hence, we can say that "Reinforcement learning is a type of machine learning method where an intelligent agent (computer program) interacts with the environment and learns to act within that." How a **Robotic dog learns** the movement of his arms is an example of Reinforcement learning.

It is a **core part of Artificial Intelligence**, and all AI agent works on the concept of reinforcement learning. Here we do not need to pre-program the agent, as it learns from its own experience without any human intervention

Example



PRAMOD NAIK

MACHINE LEARNING ACTIVITIES:

The first step in machine learning activity starts with **data**. In case of supervised learning, it is the labelled training data set followed by **test data which is not labelled**. In case of unsupervised learning, there is no question of labelled data but the task is to find patterns in the input data.

A thorough review and exploration of the data is needed to understand the type of the data, the quality of the data and relationship between the different data elements. Based on that, multiple pre-processing activities may need to be done on the input data before we can go ahead with core machine learning activities. Following are the typical preparation activities done once the input data comes into the machine learning system:

- Understand the type of data in the given input data set (For example Numerical Data).
- Explore the data to understand the nature and quality.
- Explore the relationships amongst the data elements, e.g. inter-feature relationship.
- Find potential issues in data. (you might find missing values, outliers, duplicate entries, or data entry errors)
- Do the necessary remediation, e.g. impute missing data values, etc., if needed. Once issues are identified, you take steps to address them.
- Apply the following pre-processing steps, as necessary.
    1. Dimensionality Reduction
    2. Feature sub-set selection

Once the data is prepared for modelling, then the learning tasks start off.

Once the data is prepared for modelling, then the learning tasks start off. As a part of it, do the following activities:

1. " The input data is **first divided into parts** — the **training data** and the **test data** (called holdout). This step is applicable for supervised learning only.
2. Consider different **models** or **learning algorithms** for selection. "Train the model based on the training data for supervised learning problem and apply to unknown data. Directly apply the chosen unsupervised model on the input data for unsupervised learning problem.
3. After the model is selected, trained (for supervised learning), and applied on input data, the performance of the model is evaluated. Based on options available, specific actions can be taken to improve the performance of the model, if possible.

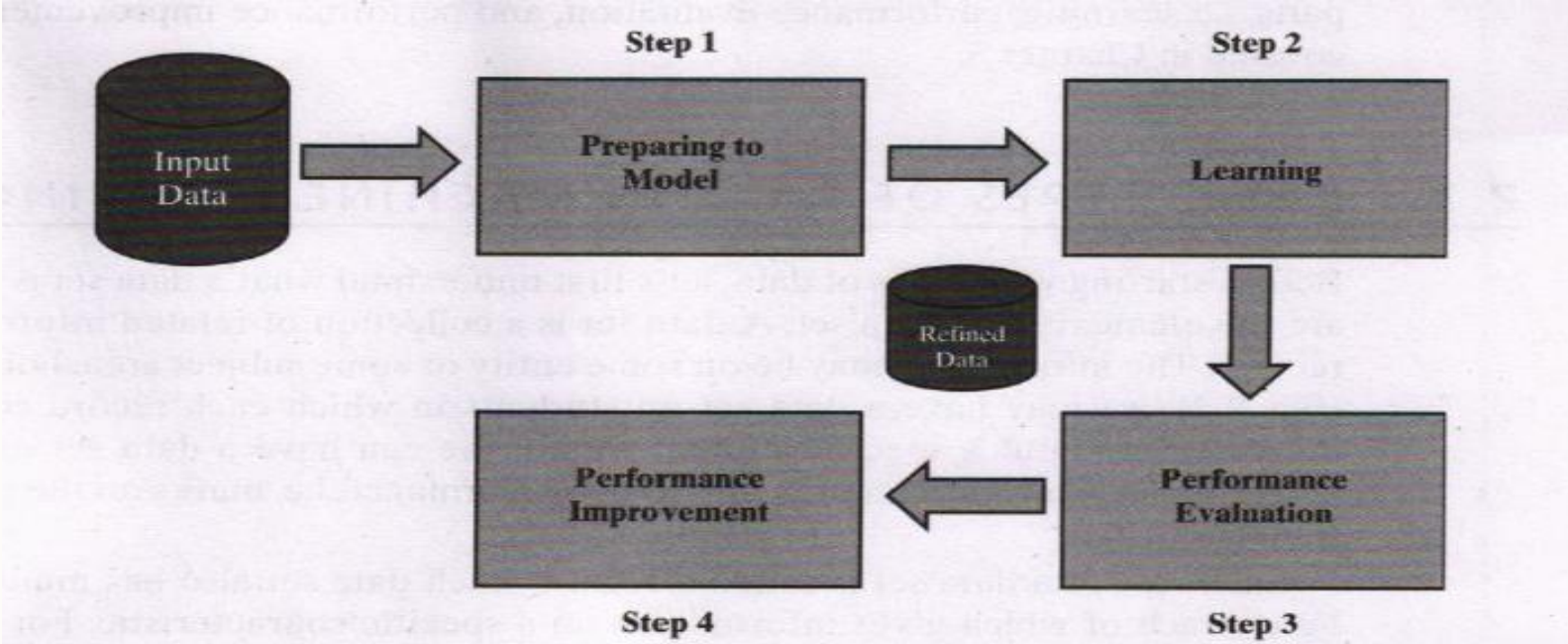


FIG. 2.1
Detailed process of machine learning

INTRODUCTION TO PYTHON

TRAINING COURSE

What is Python:

Python is an general purpose, **Object-oriented**, **High-level, Dynamically Typed**, **Compiled and Interpreted** **programming language** known for its **simplicity** and **readability**.

Common Uses of Python:

1. **Web Development**: Using frameworks like **Django** and **Flask**.
2. **Data Analysis and Visualization**: With libraries like **Pandas**, **Matplotlib**, and **Seaborn**.
3. **Artificial Intelligence and Machine Learning**: Leveraging tools like **TensorFlow**, **PyTorch**, and **Scikit-learn**.
4. **Automation/Scripting**: **Automating repetitive tasks** using Python scripts.
5. **Game Development**: Using libraries like **Pygame**.
6. **Scientific Computing**: Libraries like **SciPy** and **SymPy** are used for simulations and computations.
7. **Software Development**: For prototyping and backend development.

Python Tokens:

- In Python, **tokens** are the smallest units of the program that have a meaningful role. Python breaks every program into these tokens during the lexical analysis phase.

There are **5 types of Python tokens**:

1. Keywords
2. Identifiers
3. Literals
   - String Literals
   - Numeric Literals
   - Boolean Literals
   - Special Literal (None)
   - Collection Literals
4. Operators
5. Punctuators

Interpretation in Python

The **bytecode** produced by the compilation step is executed by the **Python Virtual Machine (PVM)**. The PVM **interprets the bytecode line by line** and **executes it**. This is what gives Python its **"interpreted"** behavior.

Key Points

- Python is **not a purely compiled language** like C or C++, where code is compiled into machine code before execution.
- Python is **not a purely interpreted language** like early versions of BASIC, which execute source code directly without compiling it.
- Instead, Python uses a **hybrid model**: it compiles source code to bytecode (a form of intermediate code), which is then interpreted.

How Python is Compiled and Interpreted:

Python is often described as both a compiled and an interpreted language, depending on how you look at its execution process.

Compilation in Python:

When you **run** a Python program, it first goes through a **compilation step**.

The **Python interpreter** compiles the source code (**.py** file) into **bytecode** (**a lower-level, platform-independent representation**). This bytecode is stored in **.pyc** files (in the __pycache__ directory).

Note:

- This compilation step is **implicit** and **happens behind the scenes**.
- The **bytecode is not directly executed by your machine**; it is executed by the Python interpreter.

What is Compiler?

- A **compiler** is a **program** that **translates the entire source code** of a programming language into **machine code** or **intermediate code** before execution. The compiled code is then executed directly by the **computer's processor**.
- **Example: C Language**

```c
#include <stdio.h>
int main() {
    printf("Hello, World!");
    return 0;
}
```

- **Process:** A C compiler (like **GCC**) compiles the code into a machine-readable **.exe** file, which can then be **executed**.

What is Interpreter?

- An **interpreter is a program** that **executes code line by line, translating it directly into machine code as it runs.**
- **Example: Python:**

```
print("Hello, World!")
```

- **Process:** The Python **interpreter** reads the script **line by line** and executes it immediately.

📘 Definition: Python for Data Analysis

Python for Data Analysis refers to the use of the Python programming language to **explore**, **clean**, **transform**, **visualize**, and **model data** in order to gain **insights** and **make data-driven decisions**.

It combines **programming** with **data manipulation**, **statistical analysis**, and **visualization techniques** using powerful libraries like:

- **Pandas** – for data manipulation and analysis
- **NumPy** – for numerical operations
- **Matplotlib / Seaborn** – for data visualization
- **Scikit-learn** – for machine learning and predictive modeling
- **Jupyter Notebook** – for interactive coding and exploration

🧠 Example Use Cases:

- **Analyzing** sales trends from Excel/CSV files
- **Cleaning** messy datasets (removing nulls, converting data types)
- **Creating** dashboards and graphs for presentations
- **Building** machine learning models to predict outcomes

pandas

PRAMOD NAIK

🐼 Pandas in Python for Data Analysis?

Pandas is a **powerful** and **popular** **open-source Python library** used for **data manipulation**, **analysis**, and **cleaning**. It provides easy-to-use data structures and functions to work with structured (tabular) data efficiently.

🧠 Why Pandas?

- It simplifies data analysis tasks.
- Makes large data easy to understand and manipulate.
- Built on top of NumPy, so it's very fast.

PRAMOD NAIK

NumPy

NumPy : TheNumPy

NumPy (**Numerical Python**) is a **powerful library** in Python used for **numerical computing**. It provides support for **large**, **multi-dimensional arrays** and **matrices**, along with a collection of mathematical functions to operate on these arrays efficiently.

Why Use NumPy?

- **Faster than Python lists:** NumPy arrays are more efficient than Python lists due to optimized memory and vectorized operations.
- **Supports Multi-dimensional Arrays:** Easily handles 1D, 2D, and ND arrays.
- **Rich Mathematical Functions:** Offers functions like mean, median, standard deviation, linear algebra, etc.
- **Integration with Libraries:** Used in Pandas, Matplotlib, Scikit-Learn, TensorFlow, etc.

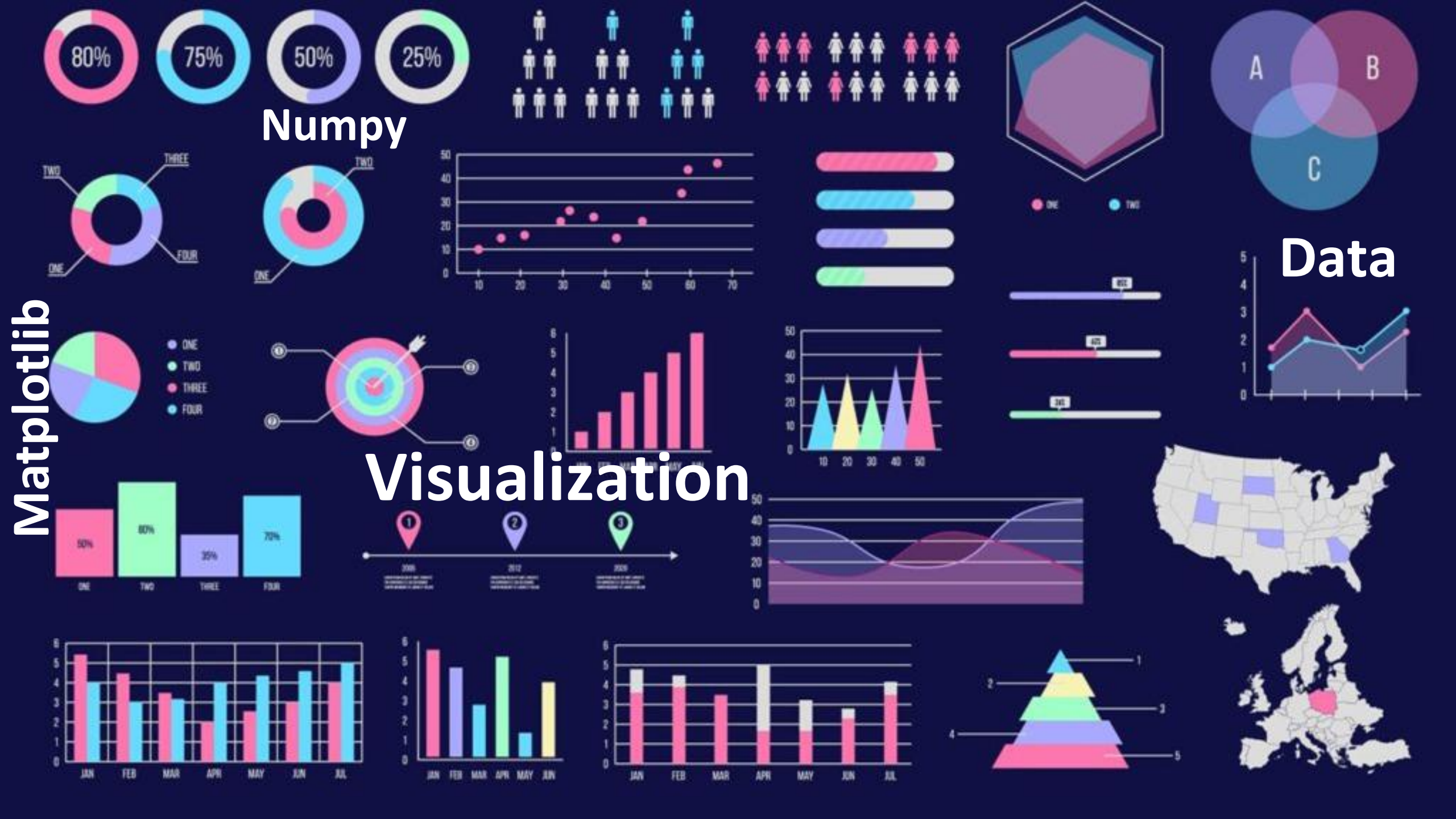
Installing and Importing NumPy

If you don’t have NumPy installed, install it using:

```
pip install numpy
```

Then, **import** it in Python:

```
import numpy as np
```


Numpy
Matplotlib
Visualization
Data

Data Visualization:

Data Visualization is the **graphical representation** of **data** and **information**. It helps in identifying **trends**, **patterns**, and **insights** in large datasets by using visual elements like **charts**, **graphs**, **maps**, and **dashboards**.

Why is Data Visualization Important?

- **Better Data Understanding** – Makes **complex data** easier to comprehend.
- **Identifies Trends and Patterns** – Helps **spot correlations** and **anomalies** quickly.
- **Improves Decision Making** – Supports **data-driven** business strategies.
- **Enhances Communication** – **Presents** data effectively for both technical and non-technical audiences.

Popular Data Visualization Tools

- Tableau
- Power BI
- **Matplotlib & Seaborn (Python)**
- D3.js (JavaScript)
- Google Data Studio

matplotlib

Matplotlib

Matplotlib is one of the **most widely used Python libraries** for **data visualization**. It provides tools to **create static**, **animated**, and **interactive plots** with a high degree of customization. It is particularly useful for plotting datasets in a variety of formats.

Installation: You can install Matplotlib using **pip**:

```
pip install matplotlib
```

Importing Matplotlib:

```
import matplotlib.pyplot as plt
```

Matplotlib Package:

Matplotlib provides functionalities to create various types of charts, graphs, and plots to visualize data effectively. It is particularly useful for **scientific computing**, **machine learning**, and **statistical analysis**.

Key Features of Matplotlib:

✅ Supports **line plots**, **bar charts**, **histograms**, **scatter plots**, **pie charts**, **3D plots**, and more.

✅ Customizable plots with **labels**, **legends**, **colors**, **styles**, and **annotations**.

✅ Can **save plots** in PNG, JPG, PDF, SVG, etc.

✅ Works well with **NumPy**, **Pandas**, and **SciPy**.

✅ Allows interactive plotting using **plt.ion()**.

Data manipulation and visualization using Pandas and Matplotlib.

🛠️ Data Manipulation with Pandas

Pandas helps us **clean**, **reshape**, **analyze**, and **prepare data for visualization** or further processing.

🔢 Basic Workflow:

- Import Data
- Clean Data
- Transform & Filter
- Aggregate/Summarize
- Prepare for Visualization

Example:

Input File: sales.csv

Date	Region	Sales	Profit
2023-01-01	East	3000	300
2023-01-02	West	7000	1000
2023-01-03	South	4500	500
2023-01-04	East		400
2023-01-05	West	8000	1200
2023-01-06	South	6000	700
2023-01-07	North	2000	300

```python
import pandas as pd

# Load dataset
df = pd.read_csv("sales.csv")

# Check top rows
print(df.head())

# Drop missing values
df = df.dropna()

# Filter: Get sales above 5000
filtered_df = df[df['Sales'] > 5000]

# Add new column: profit margin
df['Profit Margin'] = df['Profit'] / df['Sales']

# Group by Region
region_sales = df.groupby('Region')['Sales'].sum()
print(region_sales)
```

Output:

1. filtered_df (Sales > 5000):

Date	Region	Sales	Profit
2023-01-02	West	7000.0	1000
2023-01-05	West	8000.0	1200
2023-01-06	South	6000.0	700

2. df['Profit Margin']:

Date	Region	Sales	Profit	Profit Margin
2023-01-01	East	3000.0	300	0.100000
2023-01-02	West	7000.0	1000	0.142857
2023-01-03	South	4500.0	500	0.111111
2023-01-05	West	8000.0	1200	0.150000
2023-01-06	South	6000.0	700	0.116667
2023-01-07	North	2000.0	300	0.150000

📊 Data Visualization with Matplotlib

Matplotlib is used to create visual representations of the data. Often used alongside Pandas for quick plotting.

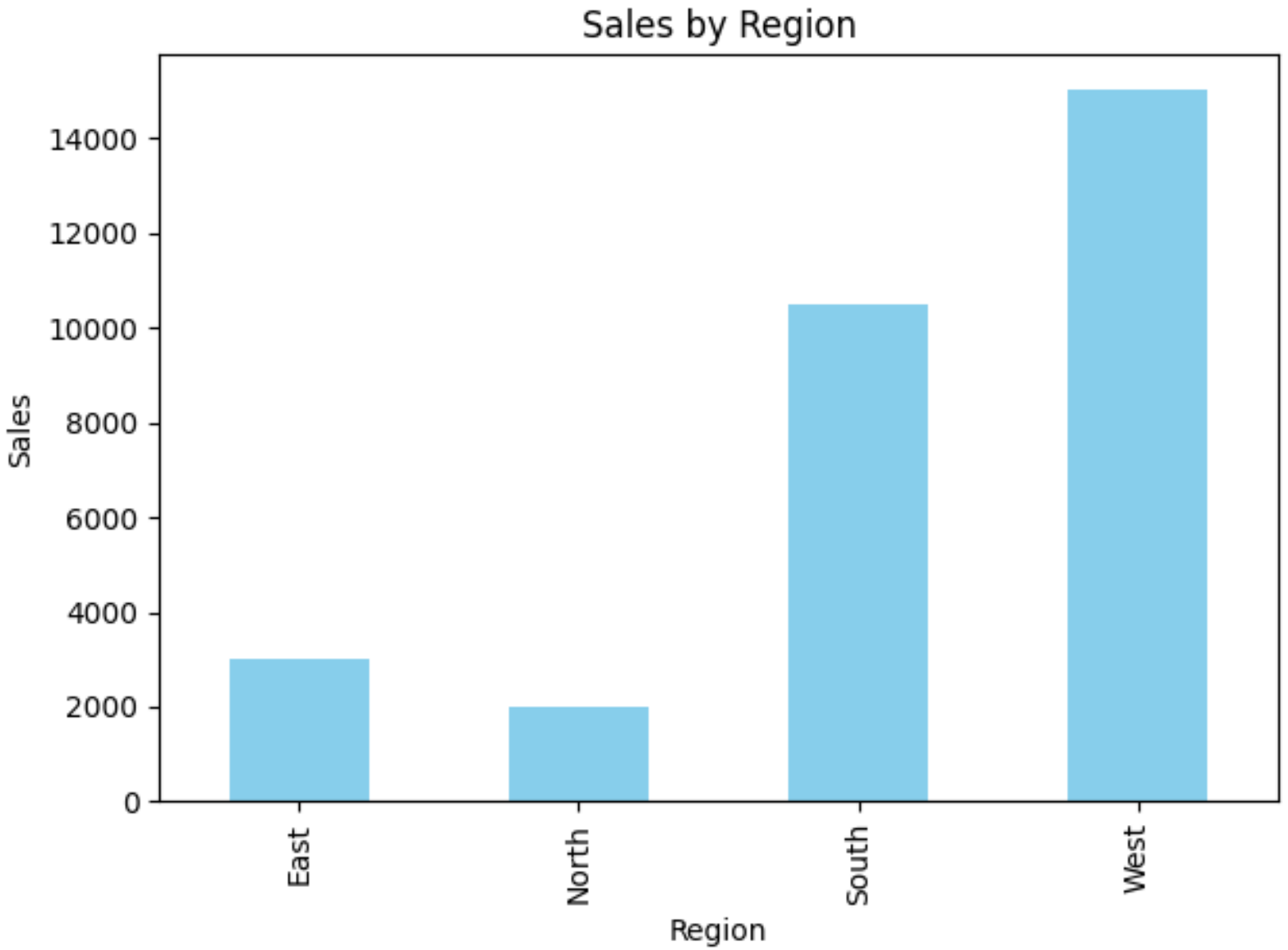
- 📈 **Basic Plotting Example:**

```
import matplotlib.pyplot as plt
```

```
# Bar chart for region sales
region_sales.plot(kind='bar', color='skyblue')
plt.title('Sales by Region')
plt.xlabel('Region')
plt.ylabel('Sales')
plt.tight_layout()
plt.show()
```

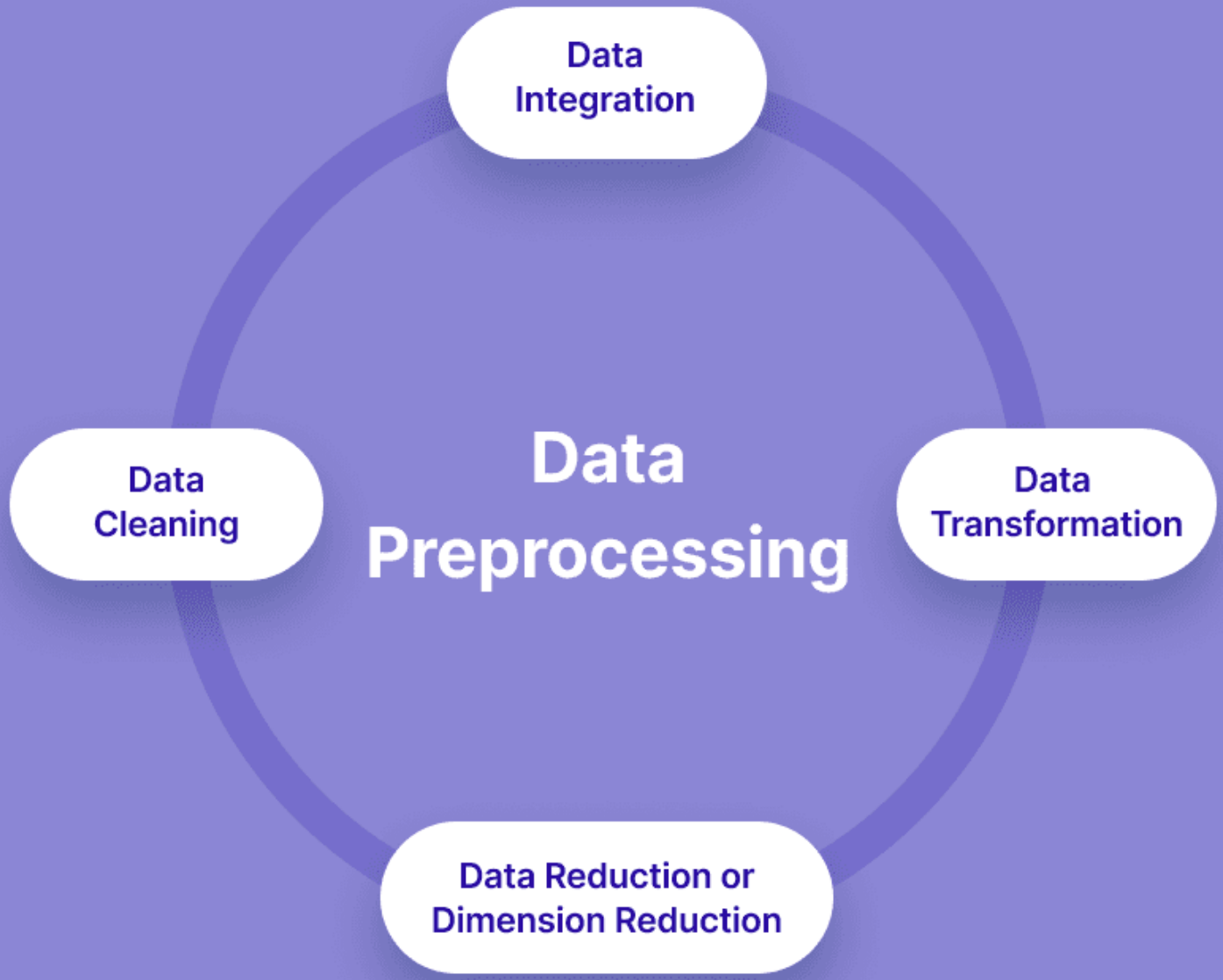
PRAMOD NAIK

Output:



PRAMOD NAIK

Data Preprocessing



✅ What is Data Wrangling?

Data Wrangling (also called **Data Munging**) is the process of **cleaning**, **transforming**, and **structuring** **raw data** into a usable format for analysis and machine learning.

💡 Why is it Important?

- **Real-world data** is **messy**, **inconsistent**, and **incomplete**
- Helps in **improving data quality** and model **performance**
- Essential for insightful **analysis** and **decision-making**

🔍 Data Preprocessing in Data Analytics with Python:

- ✅ **What is Data Preprocessing?** A **subset of** **data wrangling** focused on preparing data for ML models.

Data Preprocessing is the process of **cleaning**, **transforming**, and **preparing** **raw data** before using it for **analysis** or **machine learning models**. **Real-world data** is often **incomplete**, **inconsistent**, and **noisy**, so preprocessing is **crucial** to **improve accuracy** and **efficiency**.

📌 Steps in Data Preprocessing:

1. Handling **Missing Values**
2. Handling **Duplicate Data**
3. Handling **Outliers (**Ex: Invalid entries)
4. Encoding **Categorical Data** (Ex: Male, Female)
5. **Feature Scaling** (Normalization/Standardization)
6. **Splitting Data** for Training & Testing

PRAMOD NAIK

💡 **Example:** Data Preprocessing using Python (**Pandas & Scikit-Learn**)

Let's say we have a dataset of **customers** with **missing values**, **categorical features**, and **unscaled numerical data**.

📂 Sample Dataset (**Before Preprocessing**)

ID	Name	Age	Salary	Country	Purchased
1	Alice	25	50000	USA	Yes
2	Bob	NaN	60000	Canada	No
3	Carol	35	NaN	USA	Yes
4	David	40	80000	Canada	No
5	Eve	45	90000	India	Yes

Data cleaning and transformation:

Data cleaning and transformation in Python are **essential** steps in the data preprocessing phase, where you prepare raw data for analysis or machine learning.

1. Data Cleaning

This involves handling:

- Missing values
- Duplicates
- Incorrect data types
- Outliers or invalid entries

✅ 2. Data Transformation

This involves converting data into a suitable format:

- Normalization or scaling
- Encoding categorical variables (Ex: Converting Male & Female values to Numerical)
- Feature extraction or engineering

Example: Data Cleaning & Transformation with pandas:

```
import pandas as pd
import numpy as np

# Sample raw data
data = {
    'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Edward', 'Frank', 'Bob'],
    'Age': [25, np.nan, 35, 45, 55, 29, 29],
    'Gender': ['F', 'M', np.nan, 'M', 'M', 'M', 'M'],
    'Salary': ['50k', '40k', '70k', '90k', 'NaN', '60k', '40k']
}

df = pd.DataFrame(data)

print("Original DataFrame:\n", df)
```

Explanation:

```
df = pd.DataFrame(data)
```

This line creates a **DataFrame** object from the dictionary data.

📊 **Output of df:**

	Name	Age	Gender	Salary
0	Alice	25.0	F	50k
1	Bob	NaN	M	40k
2	Charlie	35.0	NaN	70k
3	David	45.0	M	90k
4	Edward	55.0	M	NaN
5	Frank	29.0	M	60k
6	Bob	29.0	M	40k

🧹 Step 1: Cleaning

```
# Remove duplicates
df = df.drop_duplicates()

# Handle missing values
df['Age'].fillna(df['Age'].mean(), inplace=True)
df['Gender'].fillna('Unknown', inplace=True)

# Remove rows with invalid salary
df = df[df['Salary'] != 'NaN']

# Convert 'Salary' to numeric
df['Salary'] = df['Salary'].str.replace('k', '').astype(float) * 1000
```

Here,

The parameter **inplace=True** means that the operation will modify the existing DataFrame directly — without creating a new one.

Convert Salary to numeric:

```
df['Salary'] = df['Salary'].str.replace('k', '').astype(float) * 1000
```

1. .str.replace('k', ''):

- Removes the 'k' from the end of salary strings (e.g., '50k' → '50').

2. .astype(float)

- Converts the resulting strings to numeric float values.

3. * 1000

- Converts the salary from thousands to actual amounts.

Example: '50k' → 50 → 50000.0

🔄 Step 2: Transformation

```
# Convert Gender to numerical values (encoding)
df['Gender'] = df['Gender'].map({'M': 1, 'F': 0, 'Unknown': -1})

# Normalize Age and Salary using Min-Max Scaling
df['Age_norm'] = (df['Age'] - df['Age'].min()) / (df['Age'].max() - df['Age'].min())
df['Salary_norm'] = (df['Salary'] - df['Salary'].min()) / (df['Salary'].max() - df['Salary'].min())
```

Normalization Formula:

$$\text{Normalized value} = \frac{\text{value} - \text{min}}{\text{max} - \text{min}}$$

PRAMOD NAIK

What is Normalization

Normalization is the process of **scaling numeric data** to a **standard range**, typically **between 0 and 1**.

🎯 Why Normalize?

1. **To treat all features equally:**

In datasets, features (columns) may have very different scales.

- **Example:**
  - Age might be 18–60
  - Salary might be 20,000–100,000
- Models like KNN, K-Means, Neural Networks are sensitive to such differences.

2. Improves model performance:

Helps models converge **faster** and **perform** better.

🔢 Min-Max Normalization Formula:

$$\text{Normalized value} = \frac{\text{value} - \text{min}}{\text{max} - \text{min}}$$

This **scales** all values to fall within the [0, 1] range.

PRAMOD NAIK

✅ Final Cleaned and Transformed DataFrame:

```
print("\nCleaned & Transformed DataFrame:\n", df)
```

📜 Final Transformed Table:

Name	Age	Gender	Salary	Age_norm	Salary_norm
Alice	25.0	0	50000	0.00	0.20
Bob	36.33	1	40000	0.5665	0.00
Charlie	35.0	-1	70000	0.50	0.60
David	45.0	1	90000	1.00	1.00
Frank	29.0	1	60000	0.20	0.40

PRAMOD NAIK

Handling missing values and outliers:

🧹 1. Handling Missing Values:

✅ What are missing values?

Missing values occur when **data is not available** for certain entries in your dataset. In Pandas, they are typically represented as **NaN**.

Example:

Name	Age	Salary
Alice	25.0	50000
Bob	NaN	60000
Charlie	35.0	NaN
David	45.0	80000

🔧 How to Handle Missing Values

Option 1: Remove rows with missing values

```
df.dropna(inplace=True)
```

Removes any row that has **at least one NaN**.

Option 2: Fill missing values with a default (like mean/median)

```
df['Age'].fillna(df['Age'].mean(), inplace=True)
df['Salary'].fillna(df['Salary'].median(), inplace=True)
```

After filling:

Name	Age	Salary
Alice	25.0	50000
Bob	35.0	60000
Charlie	35.0	60000
David	45.0	80000

⚠️ 2. Handling Outliers

✅ What are outliers?

Outliers are data points that are **extremely high** or **low** compared to the rest of the data. They can **distort** or **misrepresent** your analysis and models.

🔍 Example with Outlier

```
df = pd.DataFrame({
    'Name': ['Alice', 'Bob', 'Charlie', 'David', 'Edward'],
    'Salary': [50000, 52000, 51000, 49500, 200000]  # 200000 is an outlier
})
```

🔎 Detect Outliers (using IQR - Interquartile Range method)

```
Q1 = df['Salary'].quantile(0.25)
Q3 = df['Salary'].quantile(0.75)
IQR = Q3 - Q1

lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR

outliers = df[(df['Salary'] < lower_bound) | (df['Salary'] > upper_bound)]
print(outliers)
```

PRAMOD NAIK

🔍 Step-by-step Explanation:

1️⃣ Calculate Q1 and Q3 (25th and 75th percentiles):

```
Q1 = df['Salary'].quantile(0.25)
Q3 = df['Salary'].quantile(0.75)
```

- **Q1 (1st Quartile):** Value below which **25% of the data** falls.
- **Q3 (3rd Quartile):** Value below which **75% of the data** falls.

📌 Example (based on sorted Salary values):

Sorted Salaries: [49500, 50000, 51000, 52000, 200000]

- Q1 = 50000 (25% point)
- Q3 = 52000 (75% point)

2 Compute IQR (Interquartile Range):

```
IQR = Q3 - Q1
```

- IQR is the range of the **middle 50%** of the data.
- It tells you how spread out the central values are.

📌 In this example:

```
IQR = 52000 - 50000 = 2000
```

PRAMOD NAIK

3 Define lower and upper bounds for outliers:

```
lower_bound = Q1 - 1.5 * IQR
upper_bound = Q3 + 1.5 * IQR
```

- Any value **below** **lower_bound** or **above** **upper_bound** is considered an **outlier**.
- This is a **standard statistical rule** of thumb.
- The number **1.5** is a statistical rule of **thumb** commonly used in box plot analysis to detect mild outliers.

📌 In our example:

```
lower_bound = 50000 - 1.5 * 2000 = 47000
upper_bound = 52000 + 1.5 * 2000 = 55000
```

4️⃣ Identify outliers:

```
outliers = df[(df['Salary'] < lower_bound) | (df['Salary'] > upper_bound)]
print(outliers)
```

This filters rows where the salary is:

- **less than** 47000, or
- **greater than** 55000.

📌 Result:

```
      Name   Salary
4   Edward   200000
```

So Edward is identified as an **outlier** based on salary.

🔧 How to Handle Outliers

Option 1: Remove outliers

```
df = df[(df['Salary'] >= lower_bound) & (df['Salary'] <= upper_bound)]
```

Option 2: Cap or floor outliers:

```
df['Salary'] = np.where(df['Salary'] > upper_bound, upper_bound, df['Salary'])
```

It checks each element of **df['Salary']** :

- If the salary is greater than upper_bound, it replaces it with upper_bound.
- Otherwise, it keeps the original salary.

Feature scaling and Normalization:

🚀 What is Feature Scaling?

Feature scaling is a technique to bring all input features to the same scale or range.

👉 It's especially **important** for algorithms that rely on **distance** (like KNN, SVM, Logistic Regression, Neural Networks).

Two Common Types of Feature Scaling:

1. **Normalization** (Min-Max Scaling)
2. **Standardization** (Z-score Scaling)

1. Normalization (Min-Max Scaling)

- **Rescales** the data to a range between **0** and **1**.
- **Formula:**

$$X_{\text{norm}} = \frac{X - X_{\text{min}}}{X_{\text{max}} - X_{\text{min}}}$$

2. Standardization (Z-score Scaling):

- Rescales the data to have a **mean = 0** and **standard deviation = 1**.
- **Formula:**

$$X_{\text{std}} = \frac{X - \mu}{\sigma}$$

Where

- **μ** is the **mean**
- **σ** is the **standard deviation**

PRAMOD NAIK

🧪 Example Dataset

```
import pandas as pd
import numpy as np


# Sample DataFrame
df = pd.DataFrame({
    'Age': [25, 30, 45, 50, 35],
    'Salary': [50000, 60000, 80000, 120000, 70000]
})
```

PRAMOD NAIK

🔄 Normalization (Min-Max Scaling):

```
df['Age_norm'] = (df['Age'] - df['Age'].min()) / (df['Age'].max() - df['Age'].min())
df['Salary_norm'] = (df['Salary'] - df['Salary'].min()) / (df['Salary'].max() - df['Salary'].min())
```

👉 **Output:**

Age	Salary	Age_norm	Salary_norm
25	50000	0.00	0.00
30	60000	0.17	0.17
45	80000	0.67	0.50
50	120000	1.00	1.00
35	70000	0.33	0.33

PRAMOD NAIK

🔄 Standardization (Z-score Scaling):

```
df['Age_std'] = (df['Age'] - df['Age'].mean()) / df['Age'].std()
df['Salary_std'] = (df['Salary'] - df['Salary'].mean()) / df['Salary'].std()
```

👉 Output (approx):

Age	Salary	Age_std	Salary_std
25	50000	-1.41	-1.18
30	60000	-0.71	-0.56
45	80000	0.53	0.31
50	120000	1.24	1.86
35	70000	0.35	-0.43

PRAMOD NAIK

Module-1: Introduction to Machine Learning and Python:

- **Introduction to Machine Learning: Definition and importance of machine learning, Types of machine learning: Supervised, unsupervised, and reinforcement learning, Applications of machine learning in various domains.**
- **Python for Data Analysis: Introduction to Python programming, Python libraries for data analysis: NumPy, Pandas, Matplotlib, Data manipulation and visualization using Pandas and Matplotlib.**
- **Data Preprocessing: Data cleaning and transformation, Handling missing values and outliers**, Feature scaling and normalization.

PRAMOD NAIK

PRAMOD NAIK

PRAMOD NAIK

PRAMOD NAIK

PRAMOD NAIK