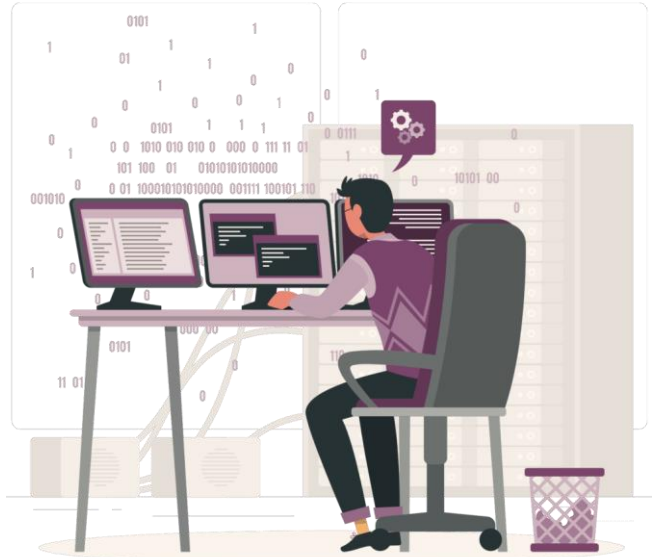
Machine Learning

Prof. Manisha

WHAT IS MACHINE LEARNING?

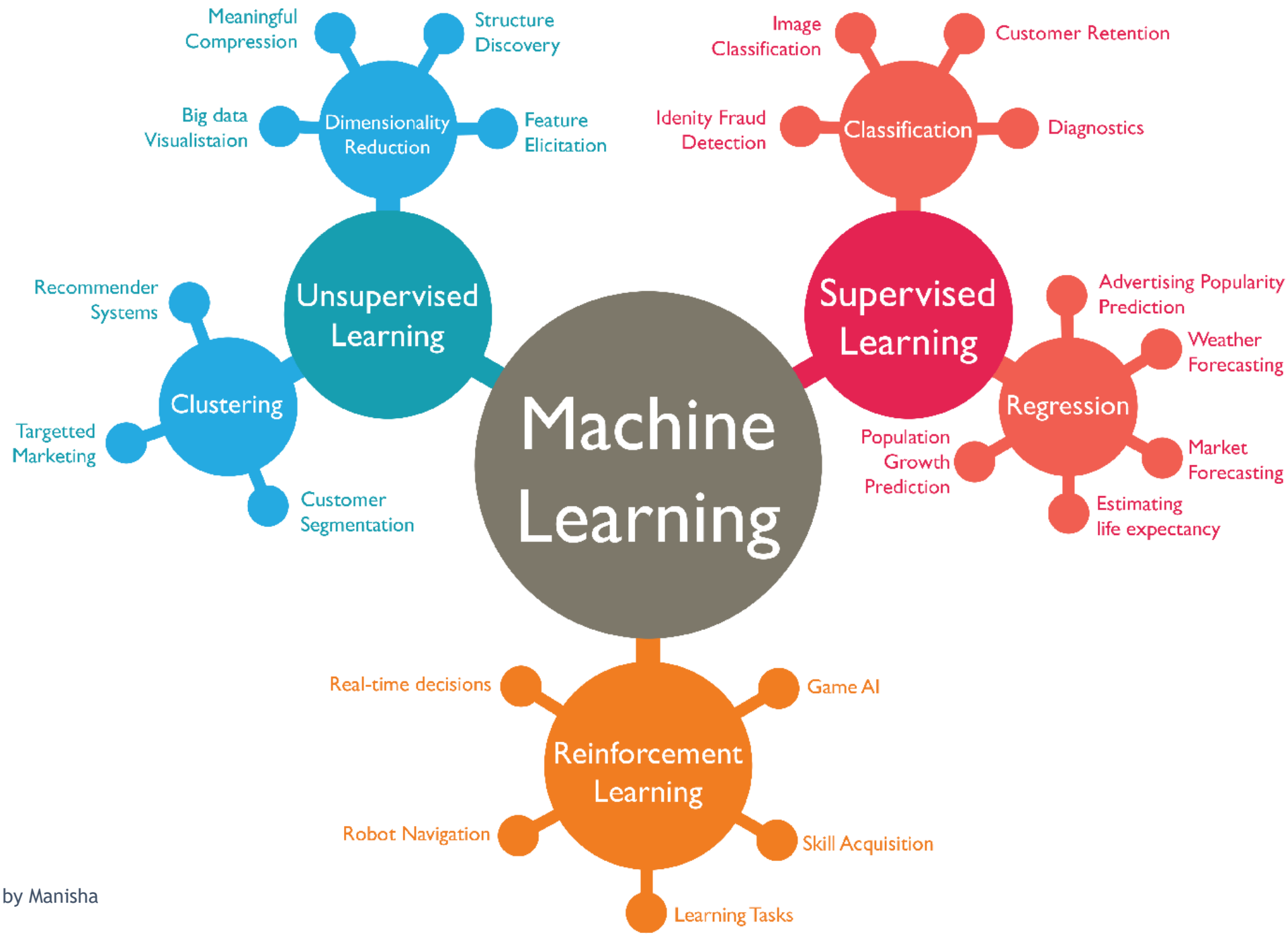
"Machine learning is like teaching computers to learn and make decisions without explicit programming. It enables computers to improve and adapt by learning from data."

- Imagine a baby learning to walk. At first, they stumble and fall. But with each attempt, they get better, eventually taking their first steps without support. That's essentially what machine learning is all about!
- Instead of babies, think of computers. We feed them data (like the baby's experiences of falling and trying again), and they learn from it. They identify patterns and connections, allowing them to make predictions or decisions without being explicitly programmed.



Here are some key points to remember:

- **No magic involved**: Machine learning isn't about robots becoming sentient. It's about using math and data to train computers to do specific tasks better.
- **Different types of learning**: Just like babies learn in different ways, there are different types of machine learning:
  - Supervised learning: Like a teacher guiding a student, the computer is given labeled examples (e.g., pictures of cats and dogs) and learns to categorize new data.
  - Unsupervised learning: Imagine a child exploring their surroundings. The computer finds patterns and structures in unlabeled data without specific instructions.
  - Reinforcement learning: The computer learns by trial and error, like playing a game and getting rewards for good moves.
- **Real-world applications**: Machine learning is everywhere! From recommending movies to filtering spam emails, it's used in various fields:
  - Entertainment: Suggesting music you might like
  - Finance: Detecting fraudulent transactions
  - Healthcare: Diagnosing diseases based on medical scans
  - Transportation: Self-driving cars



ML by Manisha

The Machine Learning Process

Data Pre Processing

- Import the data
- Clean the data
- Split into training & and test sets
- Feature Scaling

Modelling

- Build the model
- Train the model
- Make predictions

Evaluation

- Calculate performance metrics
- Make a verdict

Data Pre Processing

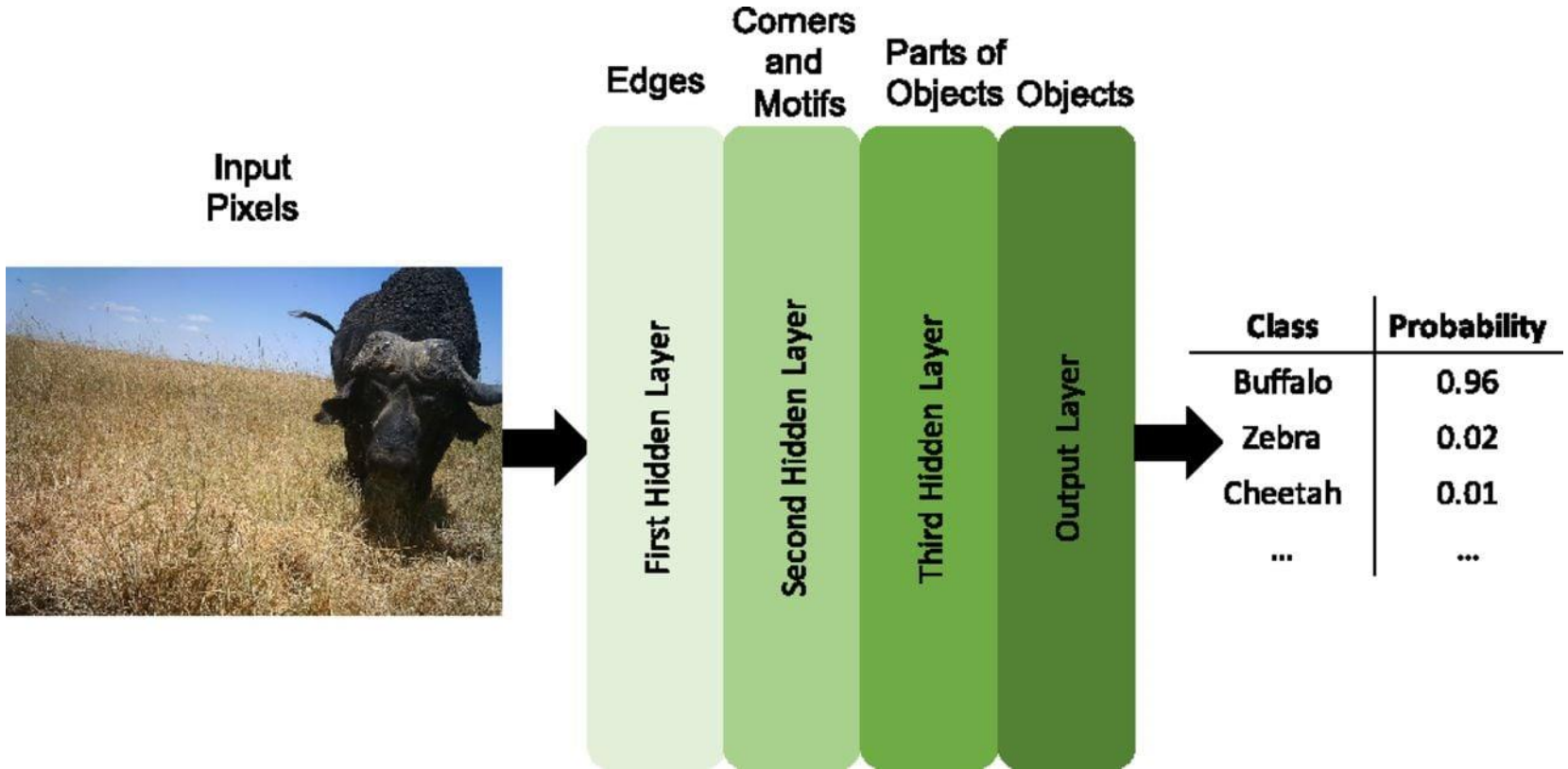
Training and Testing Data

Training Set & Test SeT

- In machine learning, we use two important sets of data: the training set and the test set. Let's explore how these sets play a crucial role in building and evaluating our models.
- Think of the **training set** as the study material for our computer model. It's like giving our computer a set of examples to learn from. The model studies these examples to understand patterns and make predictions.
- Imagine teaching a computer to recognize animals. The training set would be like showing pictures of various animals, allowing the model to learn the distinctive features of each animal.
- Now, after our computer has studied with the training set, we want to check how well it has learned. This is where the **test set** comes in. It's a set of new, unseen examples that the model hasn't encountered before.

Analogy - Pop Quiz for Animal Recognition

Think of the test set as a pop quiz for our computer model. We show it new animal pictures that it hasn't seen during its 'study' phase and evaluate how well it can correctly identify the animals.



Data Pre Processing

Feature Scaling

Feature Scaling

Imagine you're having a pizza party with friends, but everyone brought pizzas of different sizes! Comparing the deliciousness wouldn't be fair, right? That's where feature scaling comes in for machine learning models.

Why does it matter?
Machine learning models often deal with data that has features measured in different units or scales. Think of it like comparing pizza sizes: inches, centimeters, even slices! This difference can confuse the model, making it prioritize features with larger values, even if they're not necessarily more important.

Feature scaling to the rescue:
Think of scaling as putting all the pizzas on the same plate, cut into equal slices. This way, the model can focus on the actual flavors and toppings, not just the size of the pizza (or feature).

Here are the common types of scaling:

- **Normalization:** Like adjusting all pizzas to have the same diameter (e.g., between 0 and 1).
- **Standardization:** Making sure all pizzas have the same average size and "thickness" (e.g., mean of 0 and standard deviation of 1).

Feature scaling

Normalization

$$X_{new} = \frac{X - X_{min}}{X_{max} - X_{min}}$$

Standardization

$$X' = \frac{X - \text{Mean}}{\text{Standard deviation}}$$

Normalization

Think of this like setting a standard size for all your pizzas, like 12 inches in diameter. There are two main types of normalization:

Min-Max Normalization:

- This scales each feature to a specific range, typically between 0 and 1.
- Imagine stretching or shrinking all your pizzas to fit within a 12-inch diameter limit.
- Formula: (x - min) / (max - min)
- Useful when you want all features to contribute equally, regardless of their original range.
- Can be sensitive to outliers.

L1 Normalization (Manhattan scaling):

- This scales each feature so the sum of its absolute values equals 1.
- Imagine adjusting the "thickness" of each pizza slice so the total thickness of all slices adds up to 1 inch.
- Formula: x / ||x||_1
- Useful when feature sparsity (having many zeros) is important.
- Can be less intuitive than Min-Max scaling.

Standardization

Instead of setting a fixed size, this makes sure all pizzas have the same average "size" and "thickness". Imagine having an "average pizza" and adjusting all others to match its characteristics.

This involves two steps:

- Centering: Subtract the mean value of each feature from all its data points. This makes the mean of each feature become 0.
- Scaling: Divide each centered feature by its standard deviation. This makes the standard deviation of each feature become 1.
- Formula (combined): (x - mean) / standard_deviation
- Useful when features have different units or scales and the algorithm is sensitive to distance (e.g., K-Nearest Neighbors).
- Doesn't preserve the original range of data, which might be important in some cases.

Normalization

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}}$$

[0 ; 1]

Standardization

$$X' = \frac{X - \mu}{\sigma}$$

[-3 ; +3]

Data Pre Processing

Encoding

What is Encoding in Machine Learning?

Imagine you're teaching a computer to understand and make decisions based on data. Data can be numbers, but sometimes it's words or categories, like "red," "blue," or "green." Computers are great with numbers but not with words. So, we need to convert these words into numbers that the computer can understand and use. This process is called **encoding**.

Why Do We Need Encoding?

- Suppose you want to teach the computer to identify different types of fruits based on their color, shape, and size. The computer can easily understand sizes (like weight or height in numbers) but not colors or shapes described in words. Encoding converts these descriptive words into numbers.

Common Encoding Methods

Label Encoding

One-Hot Encoding

Binary Encoding

1. Label Encoding

Label Encoding is like giving each item a unique ID number.

Example:

Colors: "red," "blue," "green"

After Label Encoding:

"red" becomes 0

"blue" becomes 1

"green" becomes 2

When to use: When the categories have an order (like small, medium, large).

Common Encoding Methods

Label Encoding

One-Hot Encoding

Binary Encoding

2. One-Hot Encoding

One-Hot Encoding creates a column for each category. Each column has a 1 or 0 indicating whether a particular category is present.

Example:

•Colors: "red," "blue," "green"

•After One-Hot Encoding:

- "red" becomes [1, 0, 0]
- "blue" becomes [0, 1, 0]
- "green" becomes [0, 0, 1]

When to use: When the categories don't have an order and each category should be treated separately.

Common Encoding Methods

Label Encoding

One-Hot Encoding

Binary Encoding

3. Binary Encoding

Binary Encoding is a bit more complex. It converts each category into binary (like how computers represent data), then splits that binary number into separate columns.

Example:

•Colors: "red," "blue," "green," "yellow"

•After Binary Encoding:

- "red" might become [00, 01]
- "blue" might become [00, 10]
- "green" might become [00, 11]
- "yellow" might become [01, 00]

When to use: For large sets of categories to reduce the number of columns and still keep each category distinct.