Contents

[AI, ML AND DEEP LEARNING 3](#_Toc204867763)

[TYPES OF AI 5](#_Toc204867764)

[COMPONENTS OF AI 5](#_Toc204867765)

[HISTORY OF AI 6](#_Toc204867766)

[WHAT IS MACHINE LEARNING 7](#_Toc204867767)

[MACHINE LEARNING WORKFLOW 8](#_Toc204867768)

[MACHINE LEARNING PROCESS 9](#_Toc204867769)

[TRAINING SET AND TEST SET 10](#_Toc204867770)

[FEATURES AND LABELS 12](#_Toc204867771)

[FEATURES 12](#_Toc204867772)

[LABELS 13](#_Toc204867773)

[FEATURE AND LABELS 13](#_Toc204867774)

[REINFORCEMENT LEARNING 14](#_Toc204867775)

[DEEP LEARNING 16](#_Toc204867776)

[HOW DEEP LEARNING DIFFERENT FROM TRADITIONAL ML 16](#_Toc204867777)

[DEEP LEARNING ARCHITECTURE 18](#_Toc204867778)

[EXAMPLES 18](#_Toc204867779)

[ADVANTAGES OF DEEP LEARNING 19](#_Toc204867780)

[TRANSFORMER ARCHITECTURE 19](#_Toc204867781)

[ATTENTION MECHANISM 19](#_Toc204867782)

[WHY TRANSFORMERS MATTER FOR GENERATIVE AI? 20](#_Toc204867783)

[DIFFERENT ARCHITECTURE OF DEEP LEARNING 20](#_Toc204867784)

[RECURRENT NEURAL NETWORKS (RNNS) 20](#_Toc204867785)

[CONVOLUTIONAL NEURAL NETWORKS (CNNS) 21](#_Toc204867786)

[FOUNDATION MODEL 21](#_Toc204867787)

[KEY CHARACTERISTICS 22](#_Toc204867788)

[TYPES OF ML MODELS 23](#_Toc204867789)

[TRADITIONAL MACHINE LEARNING MODELS 23](#_Toc204867790)

[REPRESENTATIONAL (OR REPRESENTATION LEARNING) MODELS 23](#_Toc204867791)

[HOW MACHINE LEARNS 24](#_Toc204867792)

[SPECIALIZED ML PROBLEMS 25](#_Toc204867793)

[RECOMMENDATION SYSTEMS 26](#_Toc204867794)

[TYPES OF RECOMMENDATION SYSTEMS 26](#_Toc204867795)

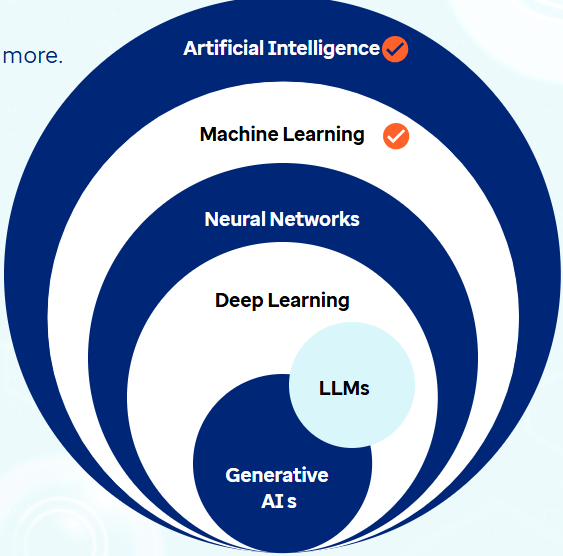
[ASSOCIATIONS RULES DETECTION 28](#_Toc204867796)

[APPLYING ML IN COMPLEX DATA 28](#_Toc204867797)

[APPLYING ML TO TEXT DATA 29](#_Toc204867798)

[ENCODING TECHNIQUES 29](#_Toc204867799)

# AI, ML AND DEEP LEARNING



Artificial Intelligence

* A broad field in computer science, encompassing a variety of technologies that work together to enable machines to think, act, and learn in ways that mimic human intelligence.

Machine Learning

* It is a key subset of AI that focuses on training algorithms to learn from data and make decisions based on that knowledge. It allows machines to identify patterns and make predictions.

How Machine Learns?

|  |  |
| --- | --- |
| Learning Type | Description |
| Supervised learning | * Training the model on labelled data |
| Unsupervised learning | * Working with unlabeled data to find hidden patterns |
| Semi-supervised learning | * Mix of both labelled and unlabeled data |
| Self-supervised learning | * Models sequentially generate their own labels from input data to learn patterns and features. * Large Language Models are based on this technique |
| Reinforcement learning | * Agent learning by interacting with an environment and receiving feedback; improves performance based on positive reinforcement and penalizes areas where negative feedback is received |

Neural Networks

* It is a subset of ML that is inspired by the structure of human brain.
* The networks consist of layers of interconnected 'neurons' or nodes, each processing inputs and passing results to the next layer.
* ***Neural networks are designed to model complex non-linear relationships within data by adjusting the weights of these connections as they process information.***

Deep Learning

* It is an advanced subset of neural networks used to analyze and learn from vast amounts of unstructured data like images, videos etc.
* The 'deep' in deep learning refers to the multiple layers in a neural network.

|  |
| --- |
| **While a basic neural network might have three layers (input, hidden, and output), deep learning algorithms utilize many more hidden layers, allowing them to process and understand much more complex patterns in data.** |

* For example, deep learning excels in tasks involving unstructured data like images, text, or audio, by allowing **multi-dimensional input data** to efficiently recognize patterns across multiple layers in a way that traditional machine learning methods cannot.

|  |
| --- |
| **Multi-dimensional input data** refers to data that has more than one feature or attribute, often organized in a structured format like arrays, matrices, or tensors. In the context of **deep learning**, this is crucial because models need to process complex data types like:  Examples of Multi-Dimensional Input Data:  **1. Images**   * Represented as 3D arrays: **Height × Width × Channels** * Example: A 256×256 RGB image → shape: **(256, 256, 3)**   **2. Text**   * Represented as sequences of word embeddings or token vectors. * Example: A sentence of 10 words, each represented by a 300-dimensional vector → shape: **(10, 300)**   **3. Audio**   * Represented as 2D spectrograms or 1D waveforms over time. * Example: A 5-second audio clip sampled at 16kHz → shape: **(80,000, 1)**   **4. Video**   * Represented as 4D arrays: **Frames × Height × Width × Channels** * Example: A 10-second video at 30 fps, 128×128 resolution → shape: **(300, 128, 128, 3)** |

Generative Ai

* A type of AI that can **create new content—like text, images, music, or code**.
* It learns patterns from existing data and uses that to generate new, similar content.
* Examples: Writing a poem or story, Creating artwork from a text prompt, Generating music or voice etc..

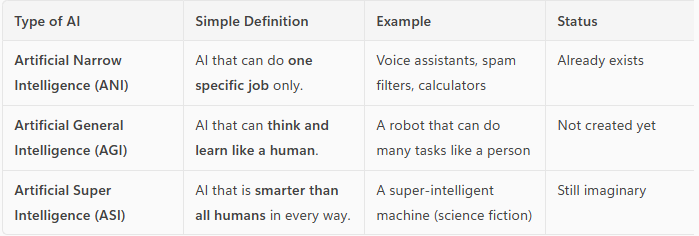
Large Language Models (LLMs)

* A subset of GenAI focused specifically on language. They understand and **generate human-like text**.
* **Examples**: Chatbots (like ChatGPT); Auto-completing emails;Translating languages;Writing code
* Within deep learning, two significant subsets are
  + **Large Language Models (LLMs)** and
  + **Generative AI (GenAI).**
* LLMs, like GPT-3.5, GPT-4, and Google Gemini, are advanced AI models designed to understand, generate, and interact with human language.
* These models are called 'large' because they are trained on massive amounts of text data, containing billions or even trillions of parameters.
* These models excel at natural language processing tasks, including text generation, summarization, question answering, and code generation by **leveraging deep neural network architectures like transformers and encoders.**

## TYPES OF AI

A pink square with black text

AI-generated content may be incorrect.



## COMPONENTS OF AI

A group of icons with text

AI-generated content may be incorrect.

|  |  |  |
| --- | --- | --- |
| Perception   * AI’s ability to **sense the world** using data from cameras, microphones, sensors, etc. * *Example:* A self-driving car “seeing” a stop sign using a camera. | Learning   * AI’s ability to **learn from data or experience** to improve its performance over time. * *Example:* A music app learning your favorite songs and suggesting similar ones. | Knowledge   * The **information and facts** that AI stores and uses to make decisions. * *Example:* A medical AI system storing information about diseases and treatments. |
| Reasoning   * AI’s ability to **think logically** and solve problems using the knowledge it has. * *Example:* A chess AI figuring out the best move to win the game. | Planning   * AI’s ability to **set goals and decide steps** to reach them. * *Example:* A robot planning the best path to clean a room without missing spots. |  |

## HISTORY OF AI

Classical AI (Symbolic AI)

* Classical AI, also known as **symbolic AI**, emerged in the 1950s–1980s.
* It focused on using **explicit rules and logic** to represent knowledge and solve problems.
* It was based on the idea that intelligence could be achieved by manipulating symbols according to formal rules.

Key Features

* Rule-based systems
* Logic programming (e.g., Prolog)
* Expert systems (e.g., MYCIN, DENDRAL)

Setbacks and Failures

* **Brittleness:** Systems failed outside narrow domains.
* **Scalability issues:** Hard to encode all human knowledge as rules.
* **Lack of learning:** Could not adapt or learn from data.
* **Commonsense reasoning:** Struggled with ambiguity, context, and real-world variability.

Knowledge-Based AI

* A subfield of classical AI that focuses on **encoding expert knowledge** into a system using rules, facts, and ontologies to perform reasoning.

Examples

* Expert systems (e.g., MYCIN for medical diagnosis)
* Knowledge graphs
* Semantic web technologies

Strengths

* Transparent reasoning
* Useful in well-defined domains

Limitations

* Difficult to maintain and scale
* Inflexible in dynamic environments

Data-Driven AI (Statistical AI / Machine Learning)

* Data-driven AI refers to systems that **learn patterns and make decisions from data**, rather than relying on hand-coded rules.

Key Technologies

* Machine Learning (ML)
* Deep Learning (DL)
* Neural Networks

Advantages

* Can handle large, complex, and unstructured data
* Learns and improves over time
* Powers modern applications like image recognition, NLP, and recommendation systems

Challenges

* Requires large datasets
* Often lacks interpretability (black-box models)
* Bias and fairness concerns

AI Winter

AI Winter refers to **periods of reduced funding, interest, and progress** in AI research due to unmet expectations and repeated failures.

Major AI Winters

* **First AI Winter (mid-1970s):** Due to the failure of early machine translation and robotics.
* **Second AI Winter (late 1980s–early 1990s):** Caused by the collapse of expert systems and unmet commercial promises.

Consequences

* Funding cuts from governments and companies
* Decline in AI research and job opportunities
* Skepticism about AI’s potential

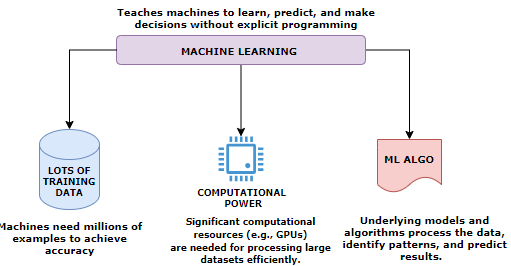
# WHAT IS MACHINE LEARNING

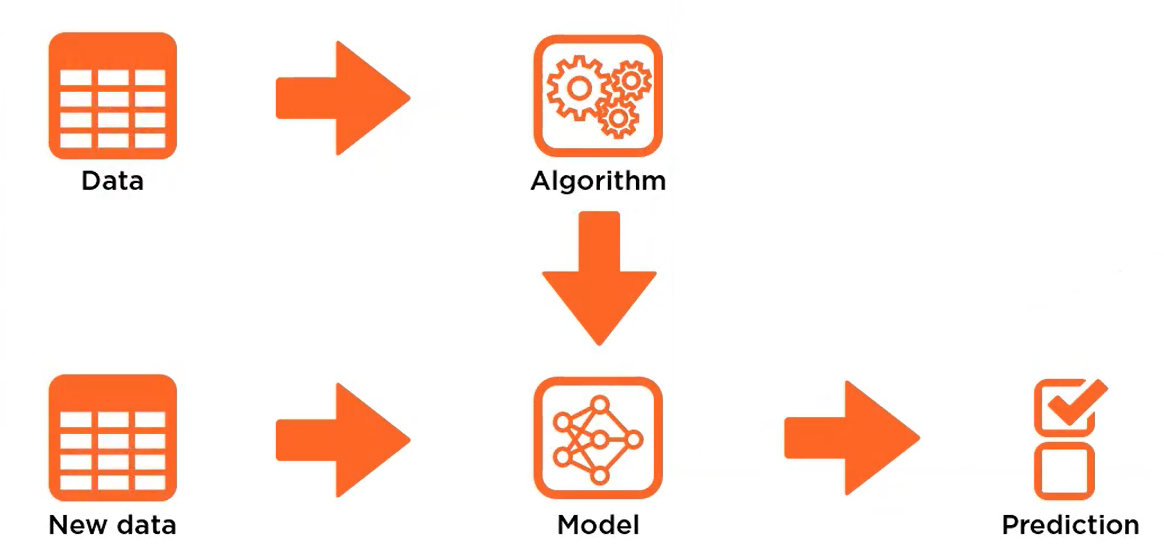
* Machine learning is a subfield of artificial intelligence.

A close-up of a pattern

AI-generated content may be incorrect.

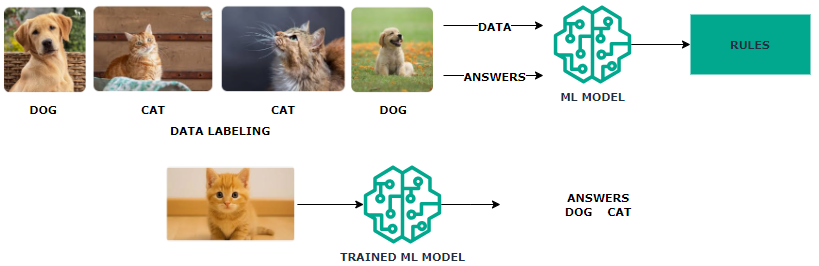
* It involves training computers on large amounts of data.
* Computers learn from data and make predictions or decisions.
* It allows computers to recognize patterns and improve their performance over time.
* Machine learning is used in various applications like image and speech recognition, recommendation systems, and predictive analytics.





We feed a set of data into a training algorithm 🡪 the algorithm produces a model 🡪then we feed new previously unseen data into that model and the 🡪model will predict what the data represent.

## MACHINE LEARNING WORKFLOW

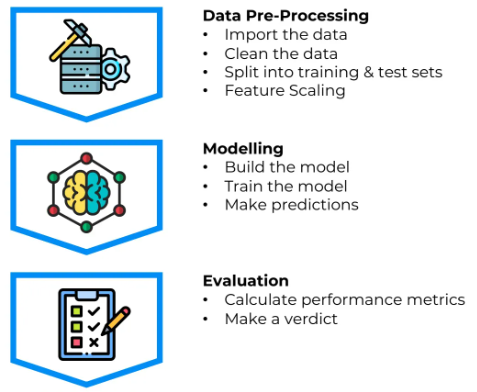


* **Machine Learning Model** learns by extracting patterns from the existing data, establishing rules automatically.
* In above example with multiple images of “dogs” & “cats” we train the model by labeling the training data. This common type of machine learning where the system is trained with labeled data (e.g., pictures of dogs and cats) is called **Supervised Learning**
* After Training the **Trained Model** can be used to classify new, unseen data.
* Machine learning systems adapt to new data without changing the program's code.

# MACHINE LEARNING PROCESS

A diagram of a model training

AI-generated content may be incorrect.



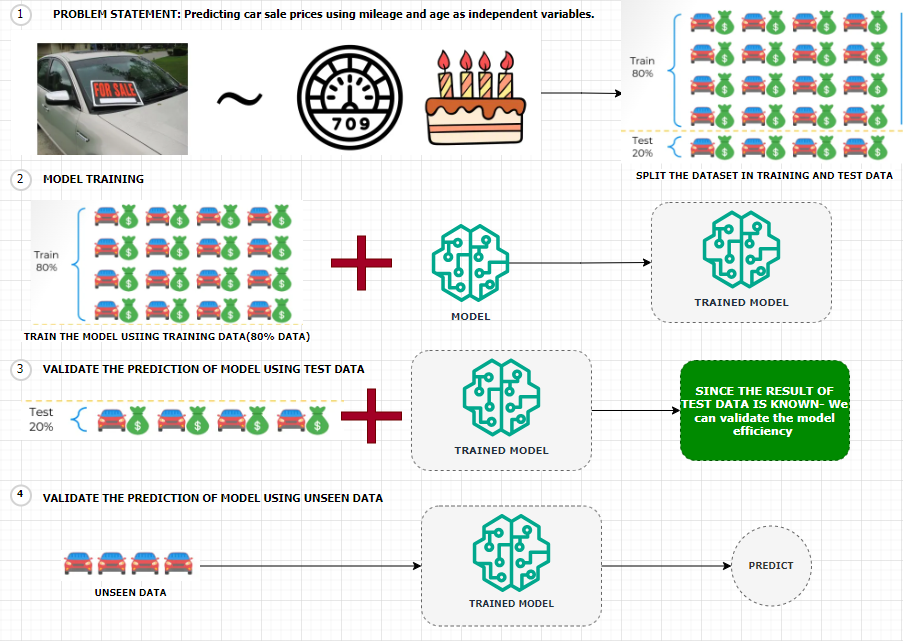
1. **DATA PREPARATION**:
   1. This step involves gathering and cleaning the data to make it suitable for analysis.
   2. It includes tasks like handling missing values, removing outliers, normalizing or standardizing the data, and splitting the data into training and testing sets.
2. **FEATURE EXTRACTION**:
   1. Feature extraction is the process of selecting or creating relevant features from the data that will be used as input for the machine learning model.
   2. It involves identifying the most informative attributes or transforming the data into a more meaningful representation that captures the underlying patterns and relationships.
3. **MODEL TRAINING**:
   1. Model training is the process of using the prepared dataset to teach the machine learning model to recognize patterns and make predictions.
   2. During this step, the model learns from the input data and adjusts its internal parameters to optimize its performance.
4. **MODEL**:
   1. The model is the representation of the learned patterns and relationships from the training data.
   2. It encapsulates the knowledge gained during the training process and can be used to make predictions or decisions on new, unseen data.
5. **TEST DATA**:
   1. Test data is a separate set of data that is not used during the model training. It is used to evaluate the performance of the trained model and measure its accuracy.
   2. The test data should be representative of the real-world scenarios the model will encounter.
6. **PREDICTION**:
   1. Once the model is trained and evaluated, it can be used to make predictions on new, unseen data.
   2. The model takes the input data, processes it using the learned patterns, and generates predictions or decisions based on the trained knowledge.
   3. The accuracy and reliability of the predictions depend on the quality of the model and the suitability of the input data.

## TRAINING SET AND TEST SET

A diagram of a car with a price tag

AI-generated content may be incorrect.

Importance Of Splitting Data Set into a training set and a test set.



Predicting Car Sale Prices Using Linear Regression

Let’s consider a scenario where we aim to **predict the sale prices of cars** using a **linear regression model**. In this case:

* **Dependent Variable**: Sale price of the car
* **Independent Variables**:
  + Mileage of the car
  + Age of the car

**Dataset Overview**

We are provided with a dataset containing information on **20 cars**. To build and evaluate our model effectively, we split the dataset into two parts:

* **Training Set (80%)**:
  + Contains data for **16 cars**
  + Used to train the linear regression model
* **Test Set (20%)**:
  + Contains data for **4 cars**
  + Set aside before training to evaluate the model’s performance

Model Training and Evaluation

* **Model Building**:  
  We train the linear regression model using only the training set. This means the model learns the relationship between mileage, age, and sale price from these 16 cars.
* **Prediction on Test Set**:  
  After training, we apply the model to the 4 cars in the test set. These cars were **not part of the training process**, so the model has **no prior knowledge** of them.
* **Comparison with Actual Prices**:  
  Since we already know the **actual sale prices** of the test set cars, we can now compare:
  + **Predicted Prices** (from the model)
  + **Actual Prices** (from the dataset)
* **Model Evaluation**:  
  This comparison allows us to assess how well our model performs on unseen data. Metrics such as **Mean Absolute Error (MAE)** or **Root Mean Squared Error (RMSE)** can be used to quantify the model’s accuracy.

## FEATURES AND LABELS

A screenshot of a white grid

AI-generated content may be incorrect.

* Let's say we are building a ML model that can predict the house prices over time. For that we need to have prior information in place about the houses that were already sold over the previous years.
* The information can be the location of the house, the number of rooms, the agent involved in selling the house,the area in square feet, and what was the price in USD. (as shown in above table)
* To train the model using historical data, we need to select the relevant features/column. ( for example, Agent column is irrelevant for price prediction)
* **This is the historical data is then used to training the ML model. Hence the data like - location, the number of rooms, and the area will become the features. Features are the measurable property within the dataset and the price is going be the label.**

### FEATURES

* Features, also known as input variables or independent variables, are the measurable properties or characteristics of the data that are used to make predictions or classifications.
* They represent the input data that the model will analyze and learn patterns from. For example, in the context of predicting house prices, features could include the location, number of rooms, and area of a house.
* Each feature is represented by a specific value or attribute.

A diagram of a brain

AI-generated content may be incorrect.

* Selecting the right features is a critical step to ensure the accuracy of ML models, something that is called feature selection.

A black arrow pointing to a white background

AI-generated content may be incorrect.

***Mathematically the list of input features will be represented as a vector with the size of N like x1,x2, x3 until Xn***

### LABELS

* Labels, also known as target variables or dependent variables, are the values that the model aims to predict or classify.
* Labels represent the desired output or outcome based on the input data. In the house price prediction example, the label would be the actual price of the house.
* The model analyzes the provided features and attempts to learn patterns that will help predict or estimate the label.

A black sign with green arrow

AI-generated content may be incorrect.

### FEATURE AND LABELS

|  |
| --- |
| * During the training process, the model is presented with a dataset that contains both the features and the corresponding labels.      * The model learns the relationship between the features and labels by adjusting its internal parameters to minimize the difference between its predicted output and the actual labels. * Once trained, the model can then be used to make predictions or classifications on new, unseen data by analyzing the features and producing a predicted label. * In summary, features are the measurable properties of the data that serve as input for the model, while labels are the desired output or outcome that the model aims to predict or classify. The relationship between the features and labels is learned by the model during the training process. |

**IN A NUTSHELL, the summary is**

|  |
| --- |
| * Building a model to predict house prices over time * Prior information about sold houses including location, number of rooms, agent, area, and price in USD * Training the model using historical data, selecting relevant features and the price as the label * Cleaning the data, removing missing values, and performing feature engineering * Splitting the dataset into training and testing sets, using the training set to train the model and the testing set to evaluate its accuracy * Testing the accuracy and precision of the model by comparing its predictions with the actual values * Deploying the model and allowing users to make predictions based on new input data, such as location, number of rooms, and area. |

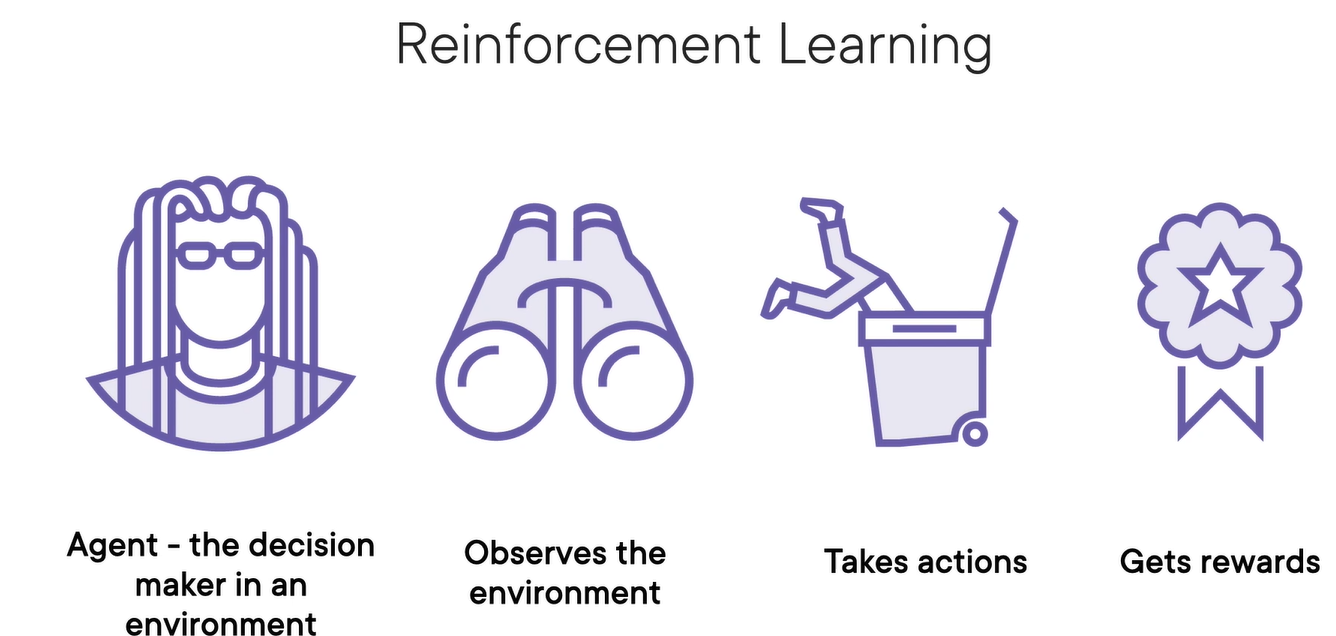
# REINFORCEMENT LEARNING

Reinforcement Learning (RL): Solving Multi-Step Problems

* Reinforcement Learning is a type of machine learning where an agent learns by trial and error to achieve a goal in an environment.

The agent:

* + Observes the environment’s state (e.g., road conditions for a self-driving car)
  + Take actions that change the state (e.g., turning, accelerating)
  + Receives rewards or penalties based on the outcome of its actions
  + Over time, the agent learns which actions lead to better outcomes and refines its strategy.



**Reinforcement Learning (RL)** is a type of machine learning where an agent learns to make decisions by interacting with an environment to maximize some notion of cumulative reward.

|  |
| --- |
| STEP TO TRAIN MODEL   * We train the agent in a computer simulation, which acts as a virtual sandbox for the real‑world environment. * Then, we fine tune the agent's behavior in a safe, real‑world environment to iron out any training issues. * Finally, we deploy the agent into the real world while we monitor its behavior and collect new data to further improve the model |

How It Works (The RL Loop)

|  |  |
| --- | --- |
|  | 1. The agent is the decision in an environment which has to constantly look around and observe the **current state/environment**. The environment is completely new for Agent 2. It takes certain **actions** in the environment. 3. If these actions are correct, the agent will receive a reward; otherwise, the agent will incur a penalty. 4. The output of Reinforced learning is set of action rather than set of predictions 5. The algorithm that determines this action is called **policy** 6. Actions are optimized to earn rewards and avoid punishments |

Example: Game Playing (e.g., Tic-Tac-Toe)

1. **State**: Current board configuration.
2. **Action**: Place X or O in an empty cell.
3. **Reward**: +1 for win, -1 for loss, 0 for draw.
4. **Goal**: Learn a policy that maximizes the chance of winning.

Real-World Applications

1. **Robotics**: Teaching robots to walk or grasp objects.
2. **Finance**: Portfolio optimization.
3. **Healthcare**: Personalized treatment strategies.
4. **Games**: AlphaGo, OpenAI Five, etc.
5. **Recommendation Systems**: Dynamic content personalization.

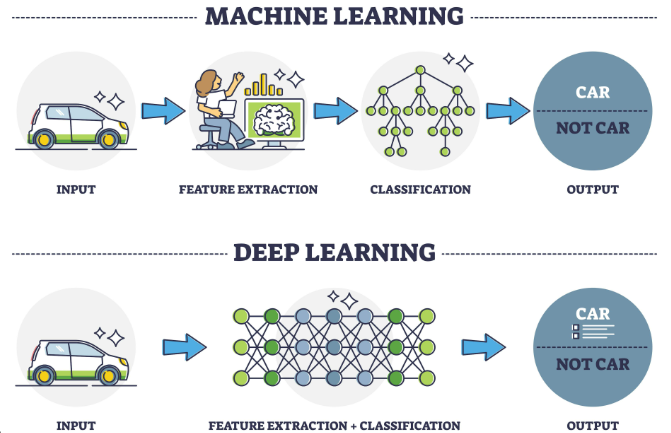
# DEEP LEARNING

A close-up of a human head

AI-generated content may be incorrect.

* Deep learning is a subset of Machine Learning based on neural networks which mimic the structure of human neurons.
* Neural networks consist of multiple computational layers.
* Each layer processes and enhances the data before passing it to the next layer.
* Deep learning trainings requires large amount of labelled data for training , hence it needs significant amount of computational power, of using GPUs or specialized hardware.
* *Neural network is the model that generative tools like ChatGPT are also using*.
* **Deep learning** enables models to handle unstructured inputs (like images, text or sound) and produce complex outputs. For example – Face Recognition using Convolution Neural Network
* Framework and Tools: Tools like Tensorflow, PyTorch, Keras etc.. are used building and training Deep learning models

## HOW DEEP LEARNING DIFFERENT FROM TRADITIONAL ML



**In Traditional Machine Learning (ML)**

1. Input: An image of a car is given to the system.
2. Manual Feature Extraction: A human expert selects important features from the image (like shape, color, wheels) using a computer.
3. Classification: These features are passed to a machine learning model (like a decision tree) to decide whether the image is a "CAR" or "NOT CAR".
4. Output: The model gives the result.

Key Idea: The system needs human help to decide what features are important.

While in Deep Learning (DL)

1. Input: The same image of a car is given.
2. Automatic Feature Extraction + Classification: A deep neural network processes the image through multiple layers. It automatically learns which features are important.
3. Output: The model predicts whether it’s a "CAR" or "NOT CAR".

Key Idea: The system learns everything on its own — no manual feature selection needed.

Summary Table

|  |  |  |
| --- | --- | --- |
| Aspect | Machine Learning | Deep Learning |
| Feature Extraction | Manual (by humans) | Automatic (by neural network) |
| Model Type | Decision Tree (example) | Neural Network |
| Input | Image | Image |
| Output | CAR / NOT CAR | CAR / NOT CAR |

How Neural Networks Work

* **Input Layer**: Data enters through input neurons.
* **Processing**: Each neuron performs mathematical operations.
* **Forwarding**: Neurons pass results to connected neurons.
* **Hidden Layers**: This process repeats across hidden layers.
* **Output Layer**: The network generates a prediction.
* A **deep neural network** has multiple hidden layers, allowing it to learn increasingly abstract features.

Applications

* Deep learning handles diverse data types: **text, images, audio, video**.
* Used for tasks like **object detection**, **video segmentation**, and **real-time motion tracking**.
* Can also generate **synthetic content**.

Limitations

* **There is limitation of Deep learning when it comes to solving multi‑step problems. To solve these types of problems, we need reinforcement learning.**

## DEEP LEARNING ARCHITECTURE

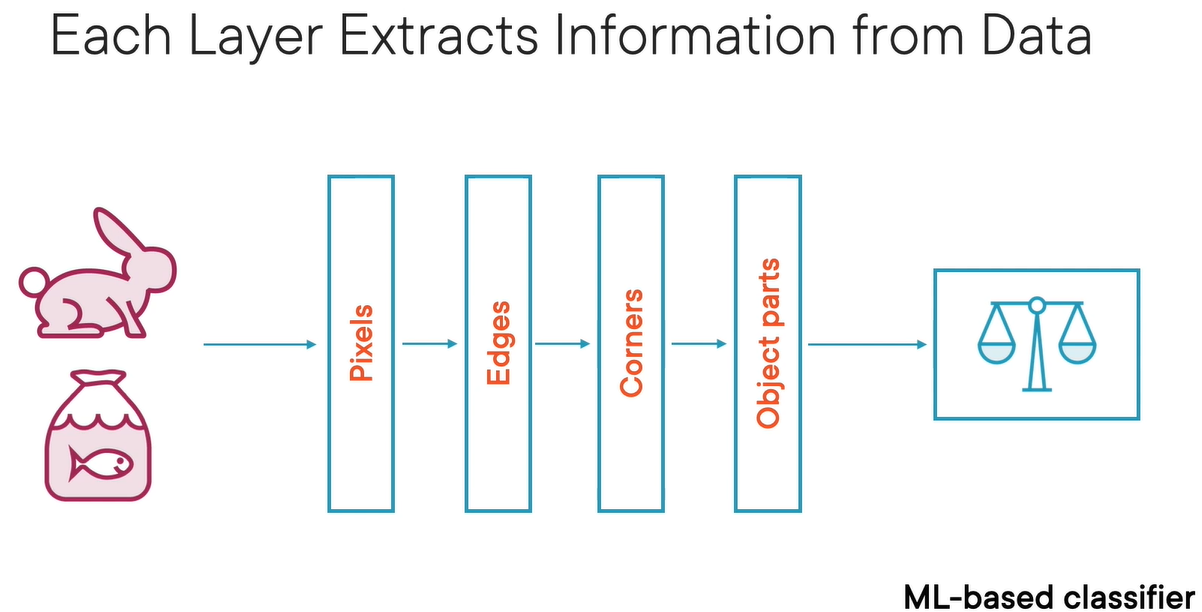
A diagram of a machine learning

AI-generated content may be incorrect.

* **Model Training**: In machine learning, training a model means teaching it to recognize patterns in data.
* **Model = Knowledge**: The trained model represents the system’s learned knowledge from the data.
* **Complex Patterns**: When data patterns are too complex for traditional methods, **deep learning** is used.
* **Deep Learning**: Uses **multiple layers** in a neural network to detect and learn complex patterns.
* **Layered Learning**: Each layer in the network captures increasingly abstract features from the data.
* **Evolution of ML**: Deep learning is a more advanced stage in the evolution of model training, enabling machines to handle tasks like image recognition, language understanding, and more.

## EXAMPLES

1. IMAGE CLASSFICATION SYSTEM



* Different layers of data extract different data from the corpus of data fed into it.
* In the above example of image classification system
  + The first layer extracts the pixels from the image
  + The second layer will put those pixels together to find edges
  + The second layer will put those edges together to find corners
  + The corners put together to view the object (image) and finally we get image recognition/classification

1. Face Recognition
   * Input: Labeled face images.
   * Layer 1: Detects basic lines and edges.
   * Layer 2: Identifies facial features (eyes, nose, mouth).
   * Layer 3: Recognizes full face patterns.
   * Output: Predicts the person’s name.
   * Each layer extracts more complex features than the previous one.

## ADVANTAGES OF DEEP LEARNING

* **Higher Accuracy**: Suitable for solving complex problems.
* **Generative AI Application**: Tools like ChatGPT use deep learning neural networks to enhance text generation accuracy.
* **Utilization of GPUs**: Recent advancements in computational power have made deep learning models more accessible and efficient.

## DEEP LEARNING MODELS

### RECURRENT NEURAL NETWORKS (RNNS)

* RNNs are a type of neural network designed to handle **sequential data** (like text, time series, or speech).
* They are good at remembering **previous inputs** using a concept called **"memory"**, which helps them understand context.

**HOW DO THEY WORK?**

* They process one element at a time in a sequence.
* Each step’s output depends on:
  1. The current input.
  2. The previous step’s output (memory).
* This loop-like structure gives RNNs their “recurrent” nature.

**WHERE ARE RNN USED?**

1. Text generation (e.g., writing like a human).
2. Language translation (e.g., English to Hindi).
3. Speech recognition (e.g., Siri or Google Assistant).
4. Stock price prediction (based on past trends).

### CONVOLUTIONAL NEURAL NETWORKS (CNNS)

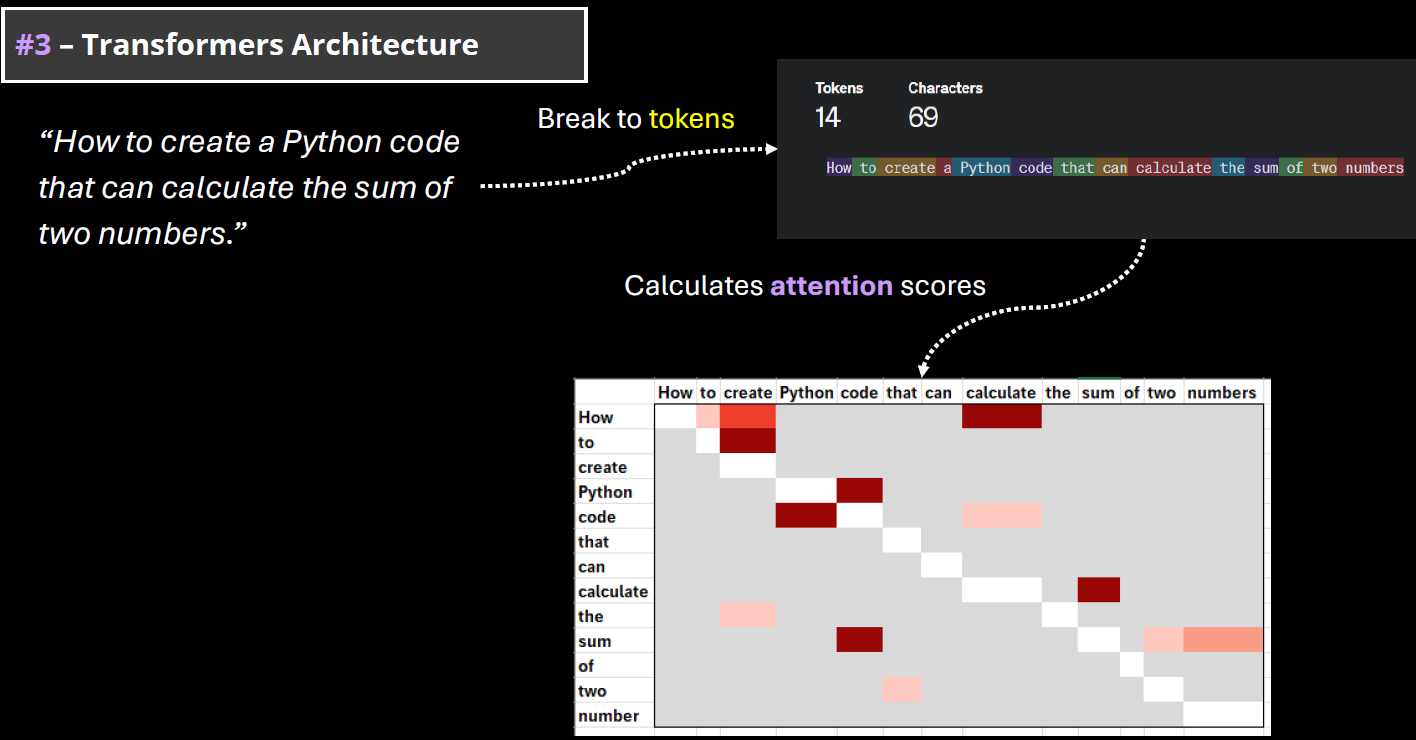
* Designed for image and video processing.
* Used in tasks like image classification, object detection, and segmentation.
* Not ideal for language tasks, but powerful in visual domains.

## TRANSFORMER ARCHITECTURE

* **Transformer Architecture is now the standard for generative AI.**
* Processes data in parallel, not sequentially like RNNs → much faster and scalable.
* Uses GPUs (Graphics Processing Units) for efficient training and inference.
* Cloud platforms (AWS, Google Cloud) use GPUs to train large models.
* It makes use of something **Attention Mechanism**

### ATTENTION MECHANISM

* Like the way we use **colored markers** to highlight important words in a sentence, transformer architecture **focus on key parts** of the input.
* In this architecture – it breaks down the word in multiple tokens called **tokenization**
* Then it calculates the attention **score** of each pair of words to understand relationships between words (e.g., “calculate” is linked to “sum”) – as shown below!
* Hence - using the transformer architecture, the parallel processing coupled with the attention mechanism enables Gen-AI system to digest more data, process it faster, and catch more complex patterns.



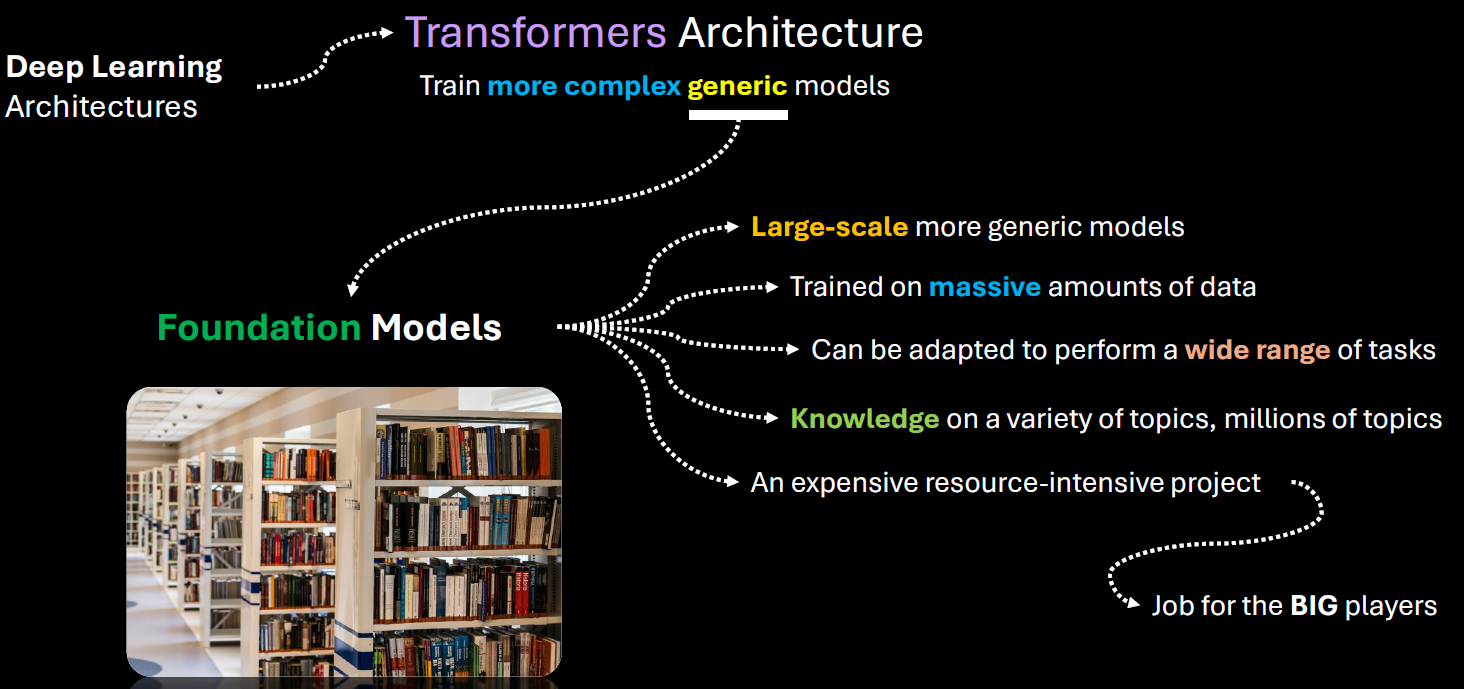
### WHY TRANSFORMERS MATTER FOR GENERATIVE AI?

A diagram of a computer

AI-generated content may be incorrect.

* The power of generative AI is based on the capabilities to digest and understand the human language. In machine learning terminology, it is called natural language processing(NLP).
* **Transformer Architecture is helping to train very complex models that can handle the human language. They are called LLMs (large language models)**
* Note: Training an LLM model requires huge amount of data and computing resources.

# FOUNDATION MODEL



* **Foundation models** are large-scale machine learning models trained on vast amounts of data that can be adapted to a wide range of tasks.
* Training foundation models is expensive and resource-intensive which requires massive data collection, storage, and processing capabilities.
* Needs advanced hardware, software infrastructure, and skilled teams.
* Big tech companies like Google, Microsoft, and Amazon have the resources to train such models.
* **Popular example**: GPT, the foundation model behind OpenAI’s ChatGPT, trained on vast amounts of data

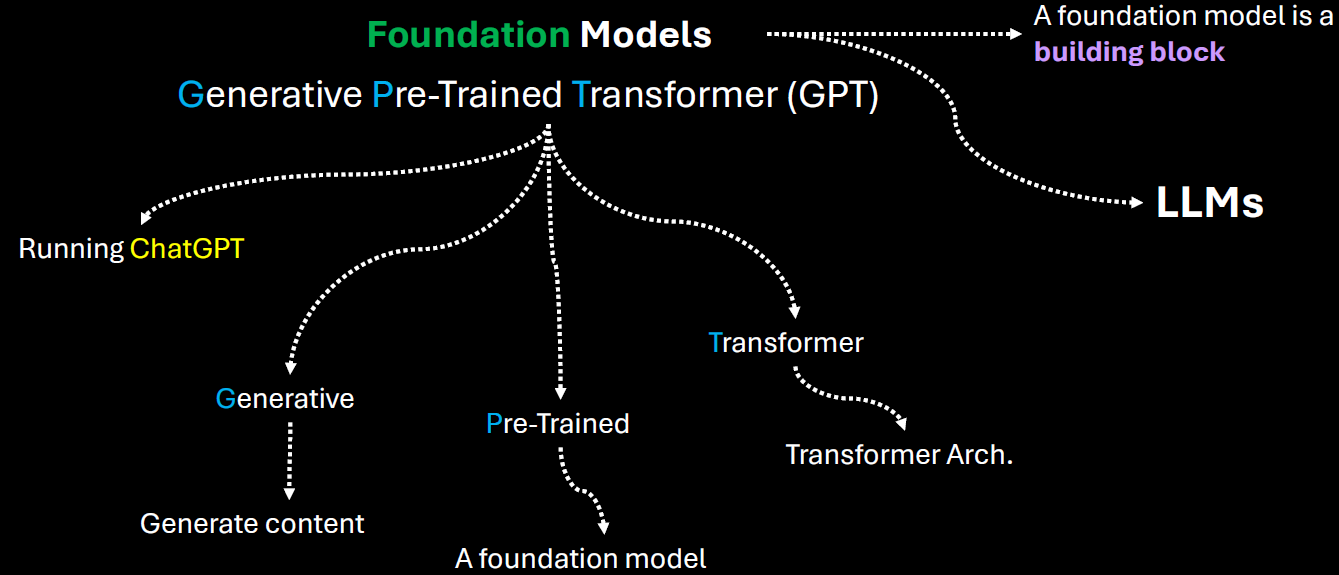
## KEY CHARACTERISTICS

1. **Scale**:  
   Foundation models are typically trained on massive datasets using billions (or even trillions) of parameters. Examples include GPT (like me), BERT, and CLIP.
2. **Generalization**:  
   They are not trained for a single task. Instead, they learn general patterns in data, which allows them to be fine-tuned or prompted for many different tasks—like translation, summarization, image recognition, or even coding.
3. **Multimodal Capabilities**:  
   Some foundation models can handle multiple types of data—like text, images, audio, or video—at the same time. For example, OpenAI’s CLIP can understand both images and text.
4. **Transfer Learning**:  
   Once trained, these models can be adapted to new tasks with relatively little additional data or training, making them highly efficient for downstream applications.

EXAMPLES OF FOUNDATION MODELS

* **GPT (Generative Pre-trained Transformer)** – for natural language understanding and generation.
* **BERT (Bidirectional Encoder Representations from Transformers)** – for understanding the context of words in text.
* **DALL·E** – for generating images from text prompts.
* **CLIP** – for connecting images and text.
* **Whisper** – for speech recognition.

ABOUT GPT



|  |  |
| --- | --- |
| **Generative** | It means that the model can generate content based on the input |
| **Pre-trained** | This model was trained on a large amount of data from diverse sources such as websites, books and articles. |
| **Transformer** | That's the internal architecture of the model, which is becoming the de facto architecture for creating foundation models. |

* Users can directly interact with the model using simple text as a prompt, ask a question and get an answer.
* Given this kind of versatility of a foundation model, smaller players, like medium sized companies or startups can leverage those foundation models developed or and provided by the big players. Hence - Instead of investing millions of dollars in training such models from scratch, they can adapt an existing foundation model for a fraction of that amount and introduce new AI based products more quickly.
* There are many types of foundation models, some focused-on handling natural language processing. Some of them are focused on computer vision tasks like image and video generation, speech recognition etc.
* One of the core foundation model types is for natural language processing. They are called LLMs

# TYPES OF ML MODELS

1. **TRADITIONAL MODEL**
2. **REPRESENTATIONAL ML MODEL**

## TRADITIONAL MACHINE LEARNING MODELS

These models rely on **manually engineered features** and are often simpler and more interpretable. They include:

**CHARACTERISTICS OF TRADITIONAL ML MODEL**

1. Require **feature extraction** and **preprocessing** by humans.
2. Perform well on **structured data** (like tables, spreadsheets).
3. Easier to interpret and debug.
4. Often faster to train and require less data.

**EXAMPLES**

* **Linear Regression**
* **Logistic Regression**
* **Decision Trees**
* **Random Forests**
* **Support Vector Machines (SVM)**
* **K-Nearest Neighbors (KNN)**
* **Naive Bayes**

## REPRESENTATIONAL (OR REPRESENTATION LEARNING) MODELS

Representation ML based systems figure out by themselves what features to pay attention to

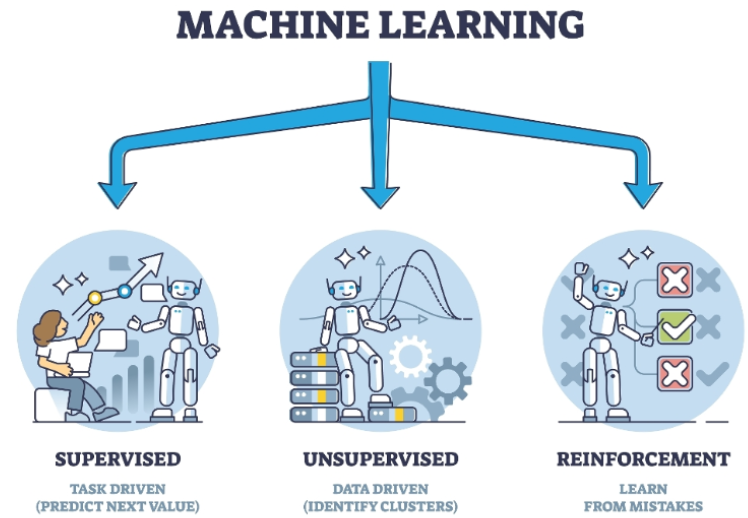
**CHARACTERISTICS OF REPRESENTATIONAL ML MODEL**

* Learn **significant features from the underlying data**
* Often based on **deep learning** architectures.
* Require large datasets and computational power.
* Less interpretable but more powerful for complex tasks.

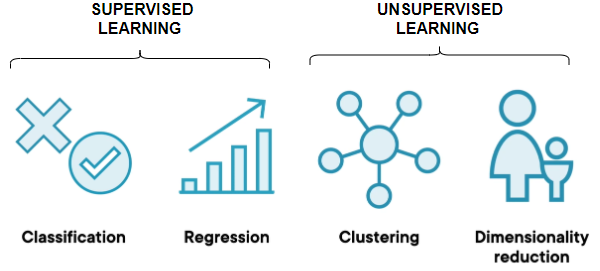
**EXAMPLES:**

* **Neural Networks (NNs)**
* **Convolutional Neural Networks (CNNs)** – for images
* **Recurrent Neural Networks (RNNs)** – for sequences
* **Transformers** – for text and more
* **Autoencoders**
* **Self-supervised learning models**

# HOW MACHINE LEARNS



1. **SUPERVISED LEARNING MODELS**
   1. In supervised learning, for training data set that we feed in the features along with the labels(output)
2. **UNSUPERVISED LEARNING MODELS**:
   1. In unsupervised learning – we don’t know what the output will be, hence we cannot provide labels in training data
   2. Here we just want the model to split our data into different groups or clusters.



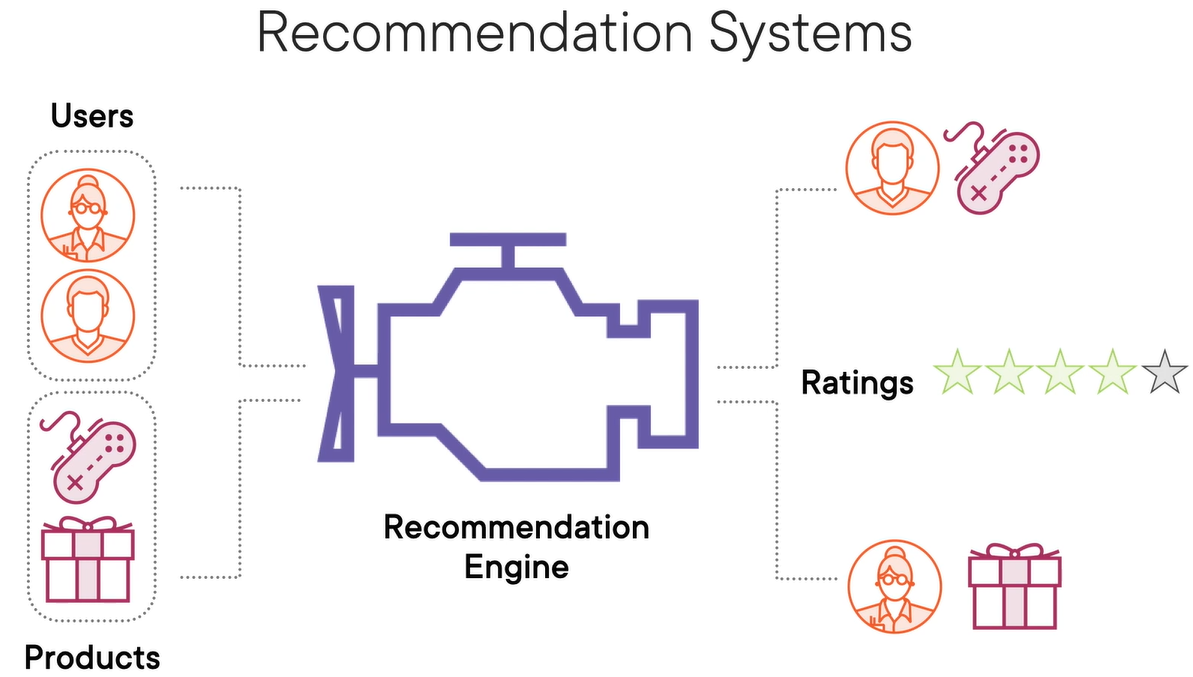
1. **REINFORCEMENT LEARNING MODELS**:
2. These models learn through interaction with an environment. They receive feedback in the form of rewards or penalties, allowing them to learn optimal strategies for decision-making.
3. Example:
   1. Let's take the example where we trying to teach a system how to play a game of chess against a human. So now, instead of we feeding in data about the different chess moves, we ask the model to play against a human.
   2. Hence, the system would go ahead and make a set of moves, and let's say it loses. So now it understands that if it performs set of actions, it results in a loss, then the next time around it will go and change its actions.
   3. It'll keep on doing this, so it's performing reinforcement of its actions. At the end when it does win a game, that’s the reward. So now it understands that based on certain set of actions that it takes, it's going to get a reward.

# SPECIALIZED ML PROBLEMS



|  |  |
| --- | --- |
| Recommendation System | * Recommend products which user might lik e * Typically used in E-commerce application |
| Association Rule Detection | * Detect transaction that can occur together * For example – **"If a customer buys Bread and Butter, they are likely to buy Milk** |
| Reinforcement Learning | * Robots learn to walk. |

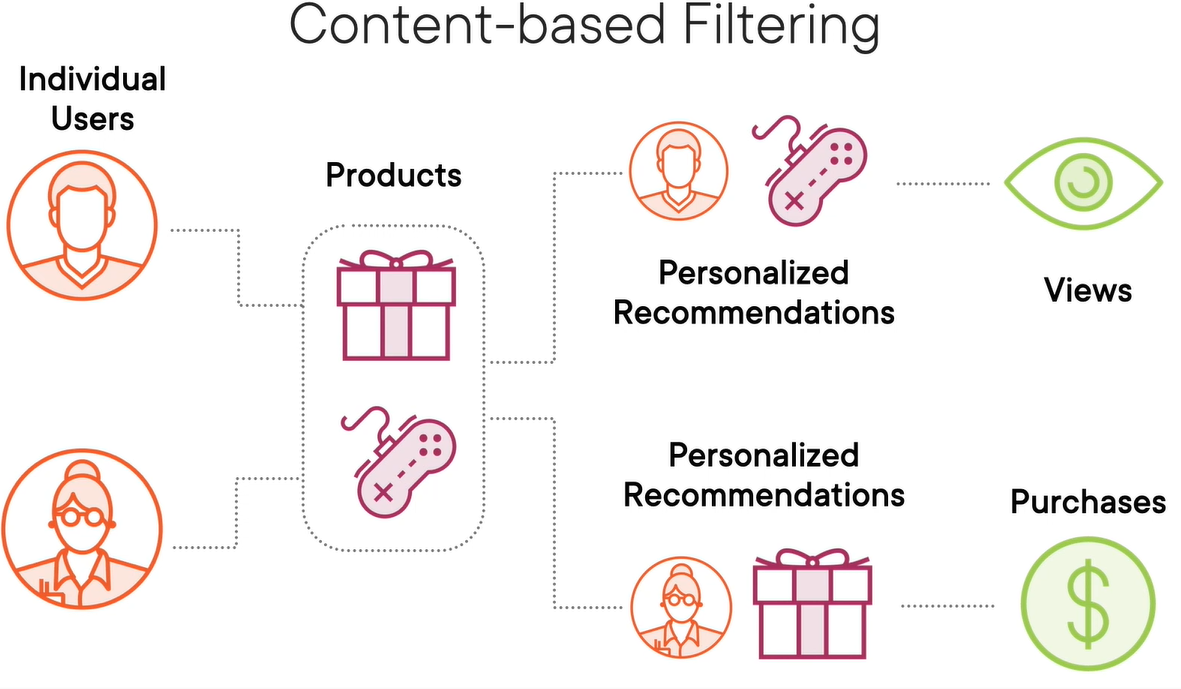
## RECOMMENDATION SYSTEMS



* A **Recommendation System** is a type of information filtering system that predicts and suggests items a user might like based on various data inputs.
* These systems are widely used in platforms like Netflix, Amazon, Spotify, and YouTube.

### TYPES OF RECOMMENDATION SYSTEMS

#### CONTENT-BASED FILTERING



* For any typical e-commerce site has large number user. This user will have history of brought and viewed products
* Based on user’s history and product characteristics can used to figure out – user will like a product or not – based on which a product is recommended.
* Items are recommended based on user profile , product features or items similar to those the user liked in the past.
  + Other Typical Example Based on item features (e.g., genre, keywords). If we watched many action movies, it recommends more action movies.

#### COLLABORATIVE FILTERING

A diagram of a product

AI-generated content may be incorrect.

* In a typical ecommerce platform there are certain user who are buying product and other set of users who are buy, like and reviewing the same product. Collaborative filtering techniques make use of this information.
* Recommend items based on the preferences of similar users.
  + Two types:
    - **User-based**: "Users like you also liked…"
    - **Item-based**: "Users who liked this item also liked…"
  + Example: Amazon’s “Customers who bought this also bought…”

#### HYBRID SYSTEMS

* + Combine content-based and collaborative filtering.
  + More accurate and robust.

Example Scenario: Movie Recommendation

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Let’s say a user has rated the following movies:   | **Movie** | **Genre** | **User Rating** | | --- | --- | --- | | Inception | Sci-Fi | 5 | | Interstellar | Sci-Fi | 5 | | The Notebook | Romance | 2 | | Titanic | Romance | 3 | | A **content-based system** might recommend more Sci-Fi movies like *The Matrix* or *Blade Runner*.  A **collaborative filtering system** might find other users who rated *Inception* and *Interstellar* highly and suggest movies they also liked. |

## ASSOCIATIONS RULES DETECTION

* **Association Rule Detection** is a data mining technique usually used to identify interesting relationships, patterns, or associations among a set of items in large datasets.
* It's commonly used in market basket analysis to find products that frequently co-occur in transactions.
* **It’s a rule based machine learning technique, where ML is used to create rules**
  + The rules are in the form “if x then y” e.g. If the shopping card has books its likely to contain stationaries
  + Association rule learning tries to identify strong rules which are supported by probability i.e. if the presence of “x” means there is high probability of “y”.
  + These strong rules can be extremely useful in
    - Recommendations
    - Cross-sell or Up-sell the produts

Key Concepts

* **Support**: How frequently an itemset appears in the dataset.
* **Confidence**: How often do items in Y appear in transactions that contain X.
* **Lift**: The ratio of observed support to that expected if X and Y were independent.

Example: Market Basket Analysis

Imagine a supermarket wants to analyze customer purchase behavior. Here's a sample dataset of transactions. **Goal: Find rules like "If a customer buys Bread and Butter, they are likely to buy Milk."**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| | Transaction ID | Items Bought | | --- | --- | | 1 | Milk, Bread, Butter | | 2 | Bread, Butter | | 3 | Milk, Bread | | 4 | Milk, Bread, Butter, Eggs | | 5 | Bread, Butter, Eggs | | Step-by-Step Rule Detection  Let’s say we want to evaluate the rule:**{Bread, Butter} ⇒ {Milk}**   1. Support = Transactions with Bread, Butter, and Milk / Total Transactions = 2 / 5 = 0.4 2. Confidence = Transactions with Bread, Butter, and Milk / Transactions with Bread and Butter = 2 / 3 = 0.67 3. Lift = Confidence / Support of Milk = 0.67 / 0.6 = 1.11   **Since Lift > 1, the rule suggests a positive association between {Bread, Butter} and {Milk}.** |

# APPLYING ML IN COMPLEX DATA

|  |  |
| --- | --- |
| Applying ML To Text Data  (NLP) | We apply ML for text in following use-cases   * Sentiment analysis of reviews on a site * Spam filtering * Language Translation * Natural Language Processing |
| Applying ML To Image Data  (CNN - convolutional neural networks) | We apply ML for image in following scenarios   * Image Classification * Object Detection * Facial recognition * Visual Search Engines * Medical imaging |
| Applying ML To Speech Data | We apply ML for image in following scenarios   * Speech Recognition * Speech to text conversion * Personalized Voice Assistants |

## APPLYING ML TO TEXT DATA

Why Text Needs Preprocessing?

* ML models work with **numbers**, not raw text.
* Text must be **converted into numeric form** before feeding into a model.

Steps to Prepare Text for ML

Step 1: Treat each sentence as a **document**.

A screenshot of a phone

AI-generated content may be incorrect.

* **Step 2: Tokenize** the document into individual words.
* **Step 3: Numerically encode** each word (e.g., X₀, X₁, …, Xₙ).
* **Step 4:** Combine encoded words into a **tensor** (multi-dimensional array).

A black text on a white background

AI-generated content may be incorrect.

### ENCODING TECHNIQUES

**There are various techniques to encode the text to numbers**

#### ONE-HOT ENCODING

What is One-Hot Encoding?

* One-hot encoding is a technique to convert **text data into numerical format** so that it can be used in machine learning models.  
  Each **unique word** in the text is represented as a **binary vector**:
* 1 indicates the presence of the word.
* 0 indicates absence.

Example

|  |  |
| --- | --- |
| Let’s say we have a small corpus of 3 reviews: | **"Amazing movie"**  **"Worst movie ever"**  **"Two thumbs up"** |
| **Step 1: Build the Vocabulary** | All unique words across the reviews:  ["Amazing", "movie", "Worst", "ever", "Two", "thumbs", "up"] |
| **Step 2: One-Hot Encode Each Review** | * **One-hot encoding** is like giving each word a unique ID in binary form * Each position in the vector corresponds to a word in the vocabulary.  | **Review** | **One-Hot Vector** | | --- | --- | | Amazing movie | [1, 1, 0, 0, 0, 0, 0] | | Worst movie ever | [0, 1, 1, 1, 0, 0, 0] | | Two thumbs up | [0, 0, 0, 0, 1, 1, 1] | |

Limitations

* **No word order** is preserved.
* **No semantic meaning** is captured (e.g., “good” and “great” are unrelated).
* Vectors can become **very large** with big vocabularies.

#### FREQUENCY-BASED ENCODING

What is Frequency-Based Encoding?

A method to convert text into numerical form by \*\*counting how often words appear\*\* in documents or across a corpus.

---

### 🔸 \*\*Common Techniques\*\*

#### 1. \*\*Count Vector\*\*

- Each document is represented as a vector of \*\*word counts\*\*.

- Simple but doesn’t account for word importance across documents.

#### 2. \*\*TF-IDF (Term Frequency–Inverse Document Frequency)\*\*

- Balances \*\*how often a word appears in a document\*\* with \*\*how rare it is across all documents\*\*.

- Formula:

$$ \text{TF-IDF}(i, j) = \text{TF}(i, j) \times \text{IDF}(i) $$

- \*\*TF(i, j)\*\*: Frequency of word \*i\* in document \*j\*.

- \*\*IDF(i)\*\*: Inverse frequency of word \*i\* across all documents.

#### 3. \*\*Co-occurrence Matrix\*\*

- Captures how often words appear \*\*together\*\* in a context window.

- Useful for understanding \*\*word relationships\*\*.

---

### 📝 \*\*Example (TF-IDF)\*\*

Corpus:

- Doc 1: "The movie was awesome"

- Doc 2: "The movie was terrible"

- \*\*TF\*\* for "movie" in Doc 1 = 1

- \*\*IDF\*\* for "movie" = low (appears in both docs)

- \*\*TF-IDF\*\* for "awesome" = high (unique to Doc 1)

---

### ✅ \*\*Advantages\*\*

- Captures \*\*importance\*\* of words.

- More informative than one-hot encoding.

### ⚠️ \*\*Limitations\*\*

- Still \*\*ignores word order\*\*.

- Doesn’t capture \*\*semantic meaning\*\* like embeddings.

#### PREDICTION-BASED ENCODING (WORD EMBEDDINGS)

* Learns **semantic relationships** between words.
* Words used in similar contexts have **similar vectors**.
* Examples: Word2Vec, GloVe, FastText.
* Captures meaning like:
  + *King – Man + Woman ≈ Queen*
  + *Paris : France :: London : England*
* Requires **large corpus** and **unsupervised learning**.

**🔹 Key Takeaways**

* Text must be **numerically encoded** for ML models.
* Choose encoding based on:
  + Simplicity → One-Hot
  + Importance → TF-IDF
  + Meaning → Word Embeddings