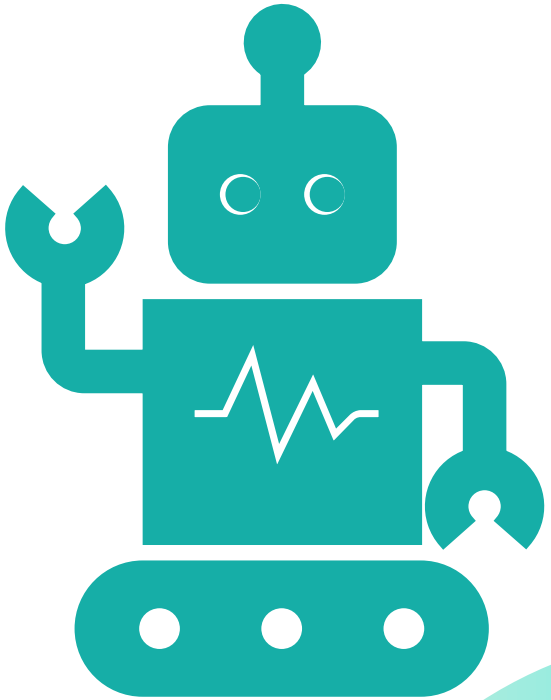
AI, Machine Learning and Deep Learning

Fundamental
Concepts and
Practical Tools

Hamid Yousefi

Summer 2024
Loan Team
Blu-Bank co

What is Artificial Intelligence



- AI is not a tool
- AI is an Agent
- It is the first technology in history that can make decision.
- It can create new ideas by itself (generative AI) and can empower human
  - Can Manipulate something like a human
  - Cannot Invent a new thing
- AI is the simulation of human intelligence in machines programmed to think and learn like humans.

What is the AI?

There is no precise definition for artificial intelligence that is in the case of all the scientists of this science. Artificial intelligence is the study of how computers can be made to do things that humans are currently doing better.
Artificial intelligence is a branch of computer science that examines the computational requirements of actions such as perception, reasoning, and learning and provides a system to perform such actions.
Artificial intelligence is the study of methods to transform a computer into a machine that can perform the tasks performed by humans.
Artificial intelligence is the science and engineering of creating intelligent machines by using computers and modeling human understanding and intelligence and finally achieving the mechanism of artificial intelligence at the level of human intelligence.
Artificial intelligence methods and techniques have been created to solve those problems that could not be easily solved by functional programming or mathematical methods.
Artificial intelligence methods are useful in areas whose problems are not well-defined Artificial intelligence, which has always been the ultimate goal of computer science, is now also serving the development of computer science

Historical Background of AI

1950s: The Birth of AI

1950: Alan Turing publishes "Computing Machinery and Intelligence," proposing the Turing Test to measure a machine's ability to exhibit intelligent behavior.

1956: The Dartmouth Conference, organized by John McCarthy, is considered the birth of AI as a field.

1960s-1970s: Early Developments

Development of early AI programs like ELIZA (a natural language processing program) and SHRLDU (a natural language understanding program).

1980s: The Rise of Expert Systems

Expert systems like MYCIN for medical diagnosis become popular, demonstrating AI's potential in specific domains.

1990s-2000s: Machine Learning and Expansion

Introduction of more sophisticated machine learning algorithms and data mining techniques.

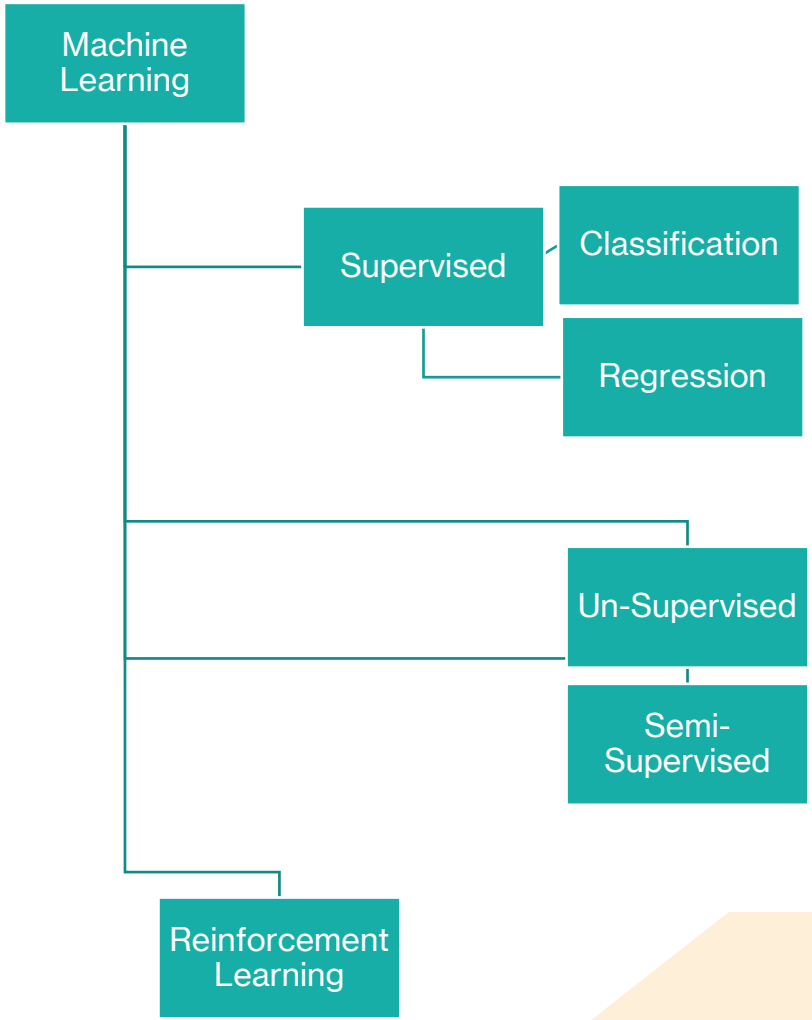
1997: IBM’s Deep Blue defeats world chess champion Garry Kasparov.

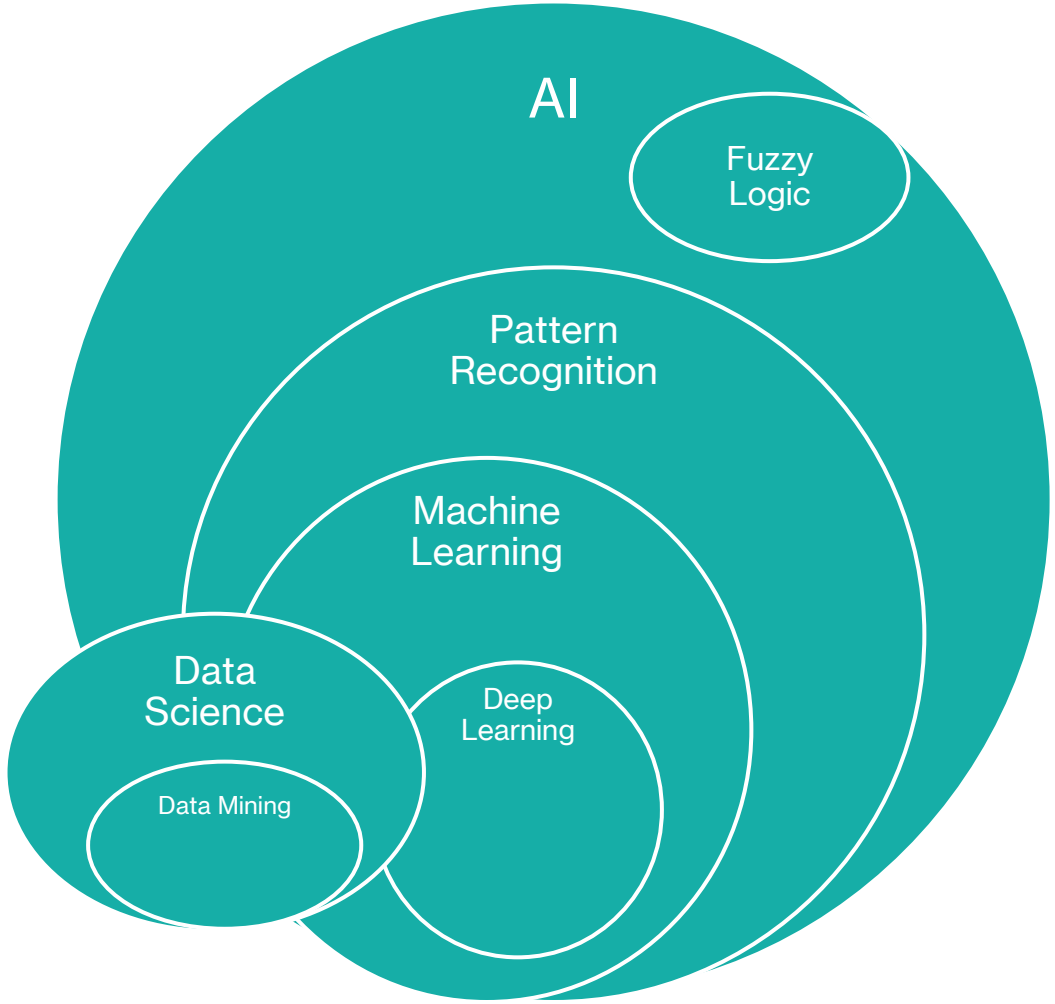
2010s-Present: AI Explosion

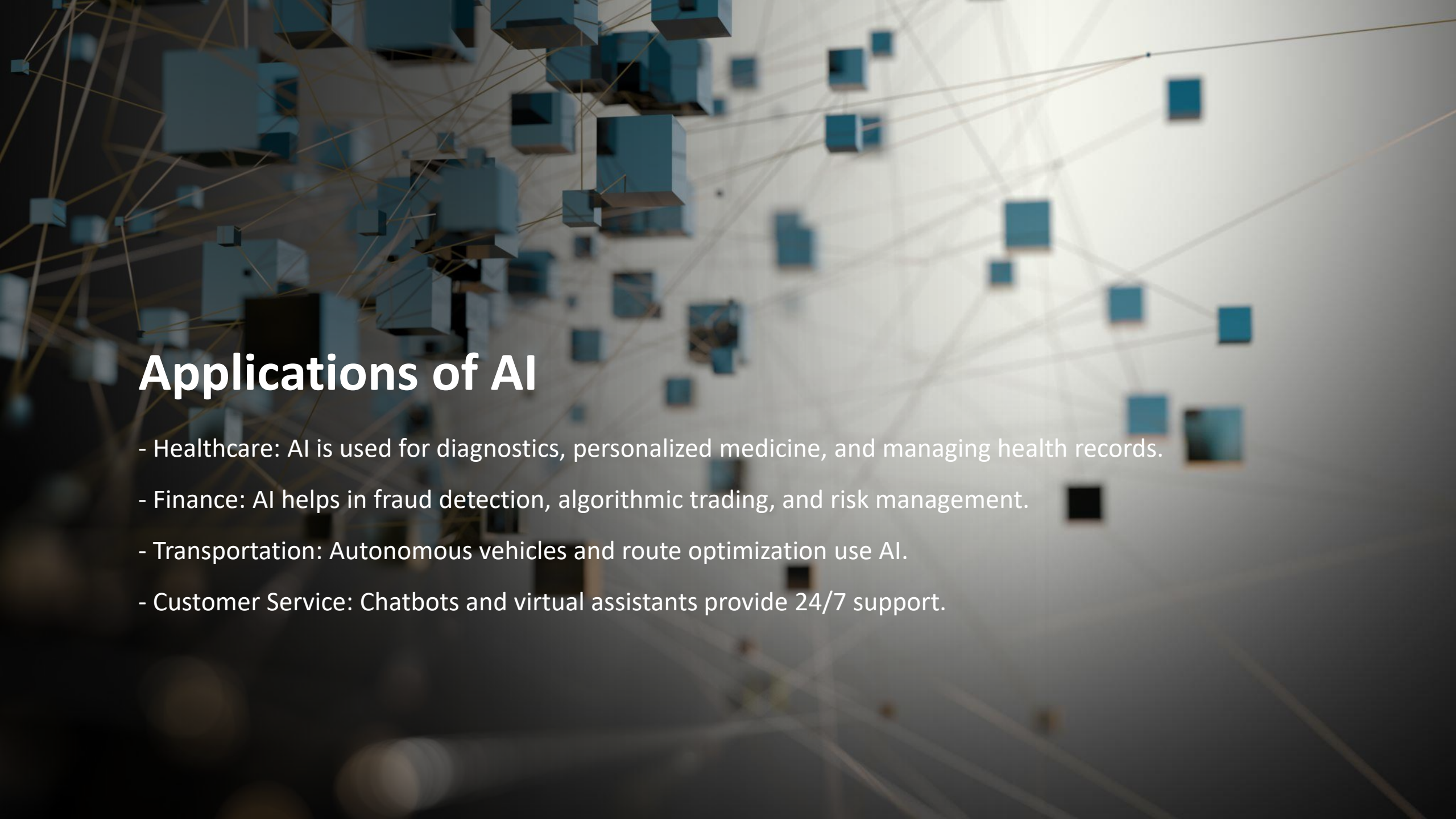
Significant advancements in deep learning and neural networks.

AI applications become more widespread, including in virtual assistants (e.g., Siri, Alexa), autonomous vehicles, and more.

AI Domain, Machine Learning Types







Applications of AI

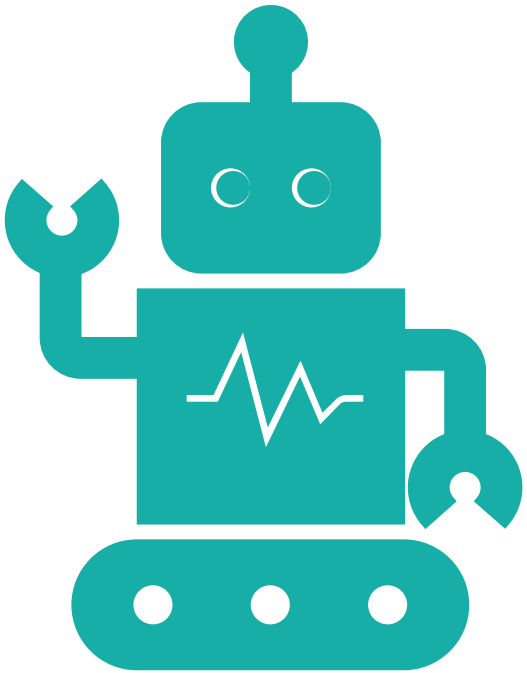
- Healthcare: AI is used for diagnostics, personalized medicine, and managing health records.
- Finance: AI helps in fraud detection, algorithmic trading, and risk management.
- Transportation: Autonomous vehicles and route optimization use AI.
- Customer Service: Chatbots and virtual assistants provide 24/7 support.

Machine Learning

ML is a subset of AI that focuses on building systems to learn from and make data-based decisions. Instead of being explicitly programmed to perform a task, ML algorithms use statistical techniques to identify patterns in data, make predictions, and improve over time as they are exposed to more data.

ML models are trained on a dataset, learning to recognize patterns and make decisions based on the input data.

Once trained, the model can make predictions or decisions on new, unseen data.



Types of ML

Supervised Learning: The model is trained on labeled data, where the input-output pairs are known. Example: Predicting house prices based on features like size, location, etc.

Unsupervised Learning: The model is trained on data without explicit labels, aiming to discover hidden patterns or groupings. Example: Clustering customers based on purchasing behavior.

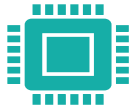
Reinforcement Learning: The model learns by interacting with an environment, receiving rewards or penalties based on its actions. Example: Training a robot to navigate a maze.

AI vs ML

Scope

AI is the broader concept of machines being able to perform tasks that typically require human intelligence. This includes reasoning, problem-solving, perception, language understanding, and learning.

ML is a specific subset of AI that focuses on the ability of machines to learn from data. It's one way to achieve AI but not the only way. Other approaches to AI can include rule-based systems, evolutionary algorithms, and symbolic AI.



Approach

AI: Encompasses various techniques, including traditional programming, where rules and logic are explicitly coded by humans.

ML: Relies on data-driven approaches, where the machine discovers patterns and makes decisions by itself after being exposed to large datasets.

Flexibility

AI: Can be as simple as a basic program that follows rules to solve a problem (like a calculator) or as complex as a self-learning system.

ML: Specifically focuses on systems that can improve and adapt without human intervention once trained, by identifying and learning from patterns in data.

Examples

Virtual assistants like Siri and Alexa, self-driving cars, and chatbots that follow scripted responses.

Email spam filters that learn to identify spam over time, recommendation systems like those used by Netflix or Amazon, and image recognition systems.

Example

A rule-based system that plays chess by following pre-defined strategies and rules coded by developers.

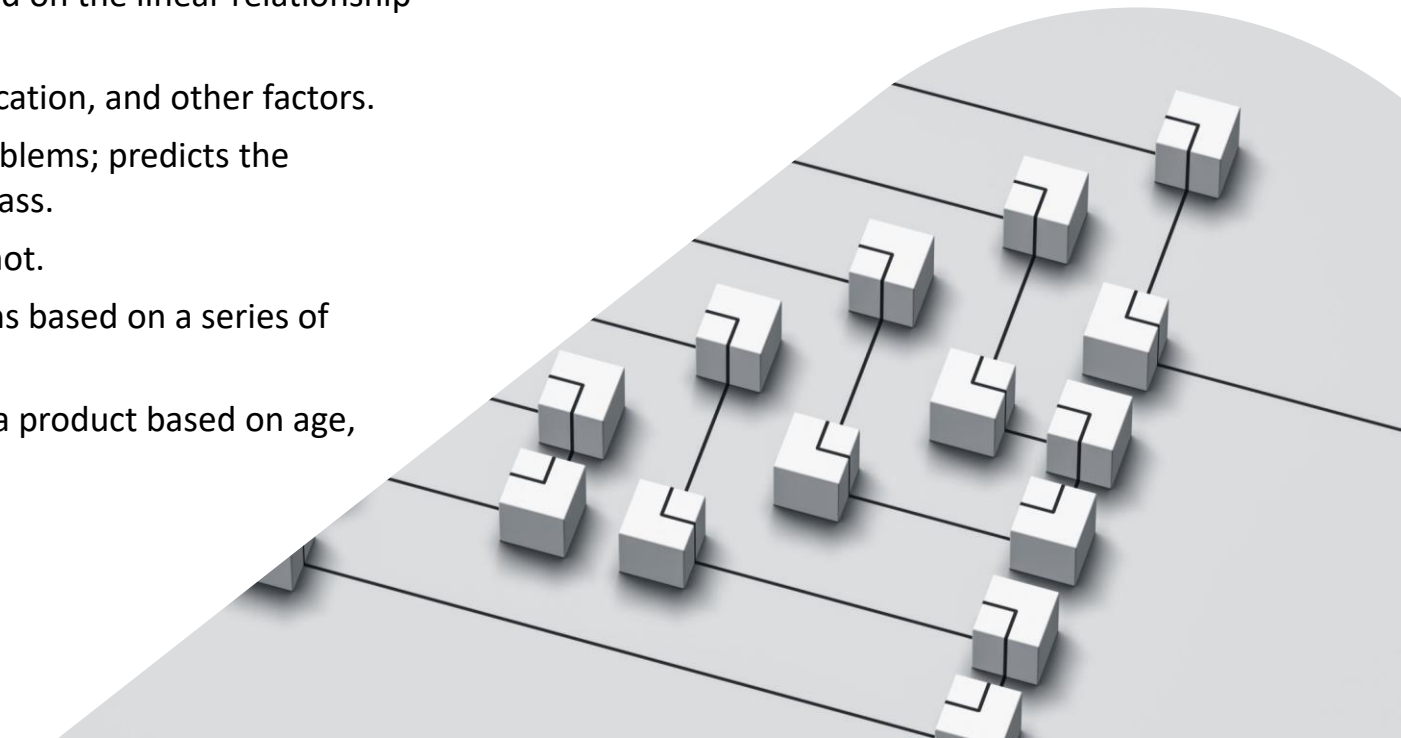
A model that learns to play chess by analyzing thousands of games and discovering patterns, then making moves based on the patterns it has learned.

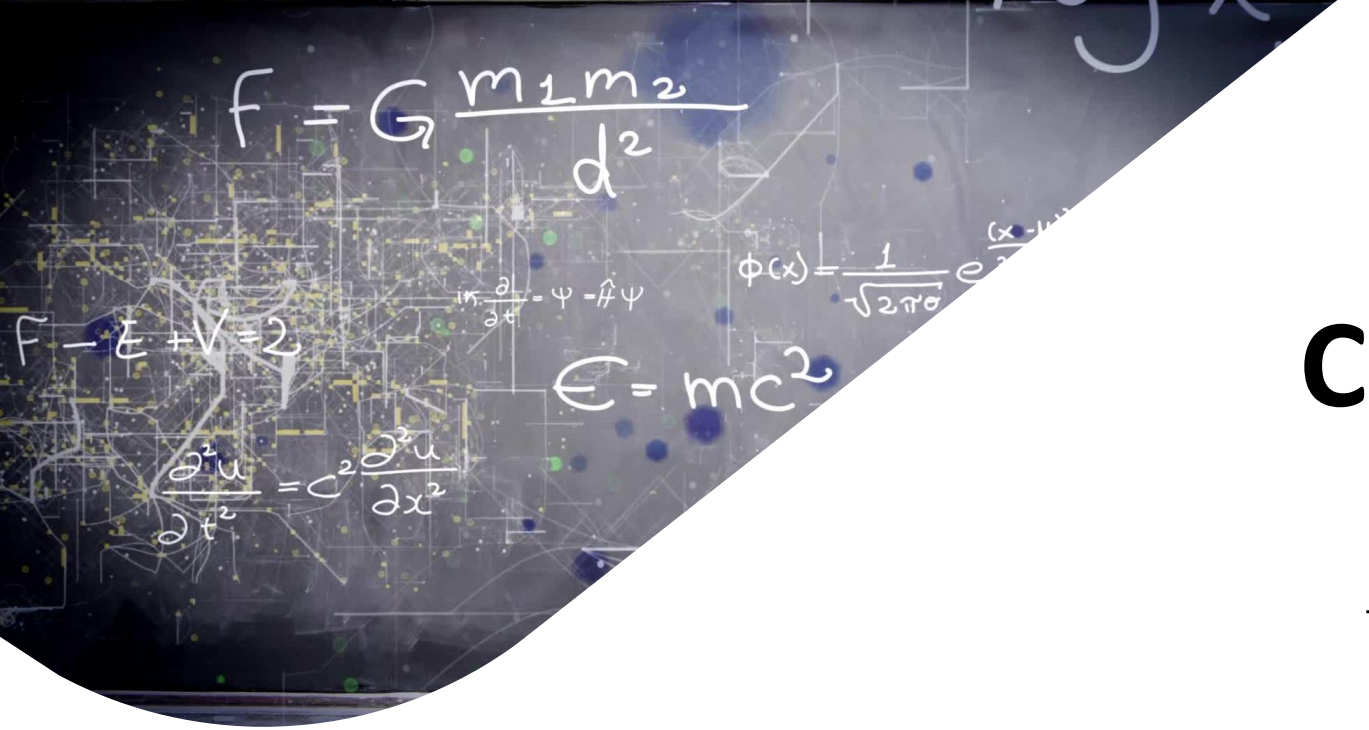
AI is the broad field that seeks to create machines capable of intelligent behavior, Machine Learning is a subset of AI that allows machines to learn from data and improve their performance over time without explicit programming.

Supervised Learning: Main Algorithms

- In supervised learning, the algorithm is trained on labeled data, meaning the input data comes with the corresponding output (target) values. The goal is to learn a mapping from inputs to outputs that can be applied to new, unseen data.

  - Linear Regression: Predicts a continuous output based on the linear relationship between the input features.
    - Example Predicting house prices based on size, location, and other factors.
  - Logistic Regression: Used for binary classification problems; predicts the probability that an instance belongs to a particular class.
    - Example Predicting whether an email is spam or not.
  - Decision Trees: A tree-like model that makes decisions based on a series of binary rules derived from the features.
    - Example Classifying whether a customer will buy a product based on age, income, etc.





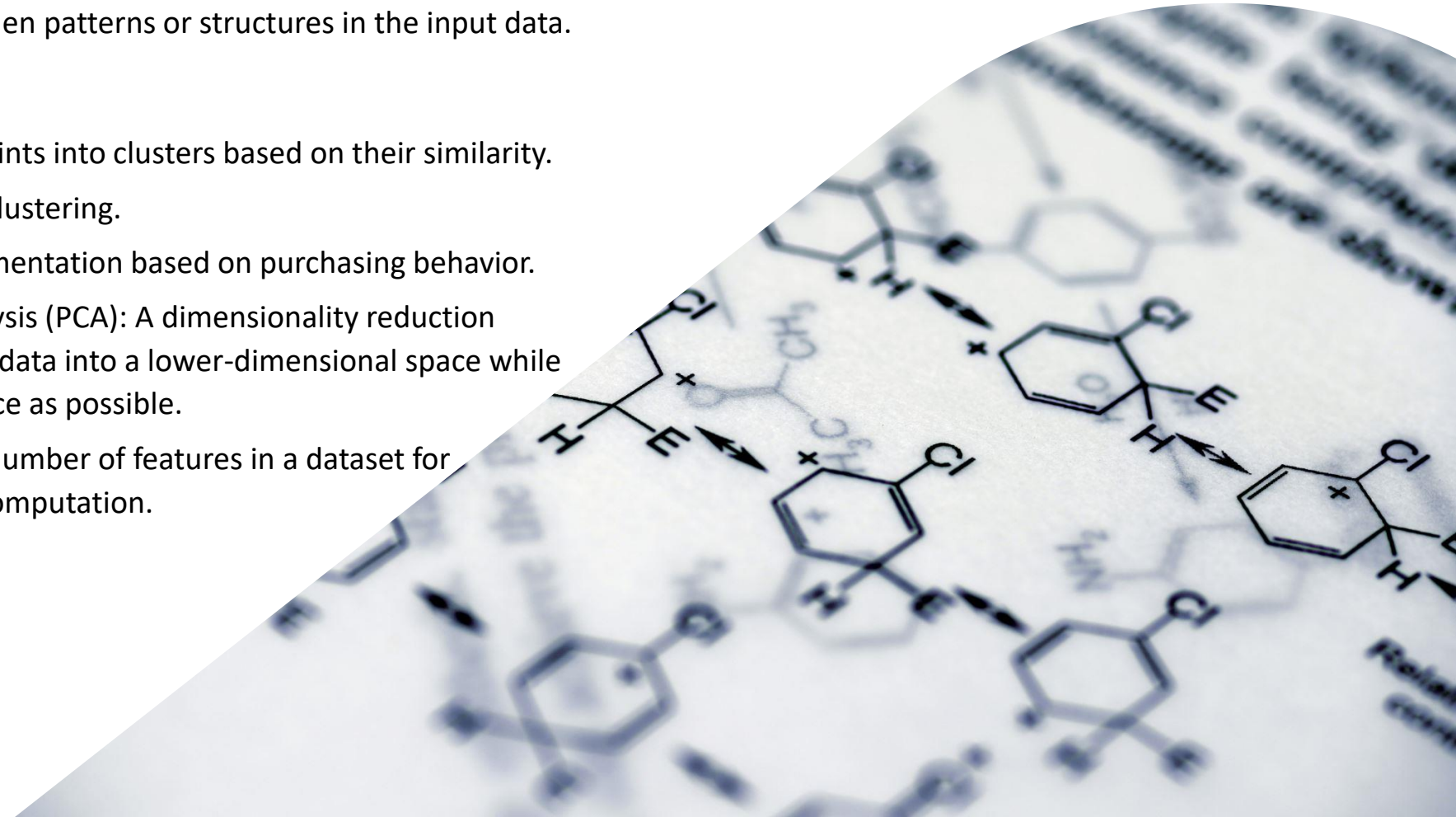
Continue…

- Support Vector Machines (SVM): Finds the optimal boundary (hyperplane) that separates different classes in the feature space.
  - Example Image classification tasks.
- k-Nearest Neighbors (k-NN): Classifies data points based on the labels of the closest data points in the feature space.
  - Example Recognizing handwritten digits.
- Neural Networks: Inspired by the human brain, these consist of layers of interconnected nodes (neurons) that can model complex patterns in data.
  - Example Image and speech recognition.

Unsupervised Learning: Main Algorithms

In unsupervised learning, the algorithm is trained on data without explicit labels. The goal is to discover hidden patterns or structures in the input data.

- Clustering: Groups data points into clusters based on their similarity.
  - k-Means, Hierarchical Clustering.
  - Example Customer segmentation based on purchasing behavior.
- Principal Component Analysis (PCA): A dimensionality reduction technique that transforms data into a lower-dimensional space while preserving as much variance as possible.
  - Example Reducing the number of features in a dataset for visualization or faster computation.



Continue…

- Autoencoders: Neural networks used for unsupervised learning of efficient representations (encoding) of data.
  - Example Anomaly detection in images.
- Association Rule Learning (e.g., Apriori, Eclat) Identifies relationships between variables in large datasets.
  - Example Market basket analysis to find items frequently bought together.

Semi-Supervised Learning

It is a hybrid approach that involves training the algorithm on a small amount of labeled data and a larger amount of unlabeled data. It’s used when labeling data is expensive or time-consuming.

- Self-Training: An initial model is trained on the labeled data, and then used to predict labels for the unlabeled data, which is iteratively added to the training set.
- Co-Training: Two models are trained on different views of the data, and each model's predictions on the unlabeled data are used to train the other

Classification vs Regression

Regression

Objective: Predict continuous numerical values.

Examples: Predicting the price of a house based on features like size and number of rooms, forecasting future temperatures, or estimating a company's revenue based on past data.

Output: A numerical value. For example, predicting that a house will cost $300,000.

Classification

Objective: Predict categorical labels or classes.

Examples: Identifying whether an email is spam or not, classifying an image as a cat or a dog, or predicting whether a patient has a certain disease.

Output: A label or category. For example, classifying an email as "spam" or "not spam."

Key Differences

Output Type: In regression, the output is a continuous numerical value, while in classification, the output is typically a categorical label or class.

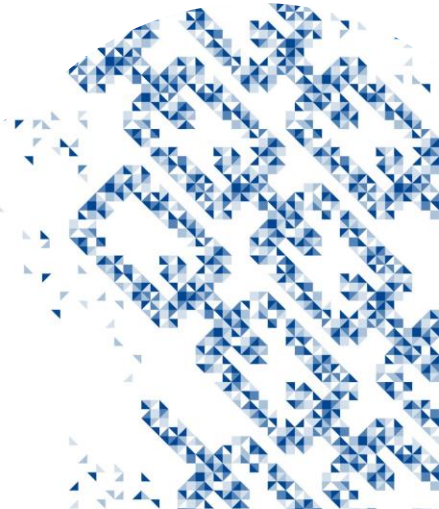
Models: Regression models include linear models (like linear regression) and non-linear models (like decision trees). Classification models include algorithms such as decision trees, support vector machines (SVM), and neural networks.

In summary, regression is used for predicting numerical values, while classification is used for predicting categories or labels.

KNN

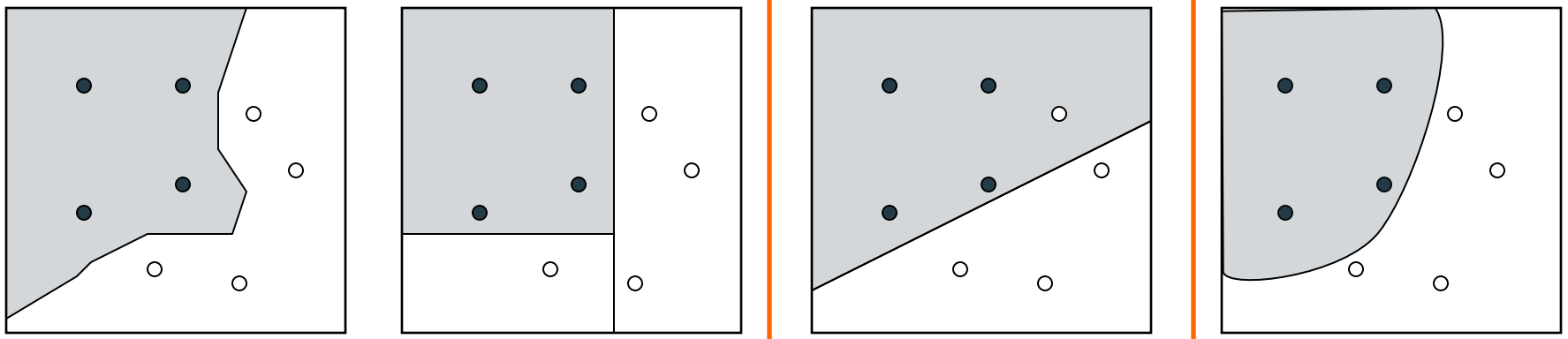
- **Definition:** K-Nearest Neighbors (KNN) is a simple, non-parametric machine learning algorithm used for classification and regression. It classifies a data point based on how its neighbors are classified.
- **How It Works:**

1. **Choose the Number of Neighbors (K):** Select how many neighbors to consider for classifying a new data point.
2. **Calculate Distances:** Compute the distance between the new data point and all other data points in the dataset.
3. **Identify Nearest Neighbors:** Find the K nearest data points to the new data point.
4. **Classify/Regress:** For classification, assign the most common label among the K neighbors. For regression, predict the average value of the K neighbors.



SVM, Discriminant Function

- It can be arbitrary functions of x, such as:



Nearest Neighbor

Decision Tree

Linear Functions

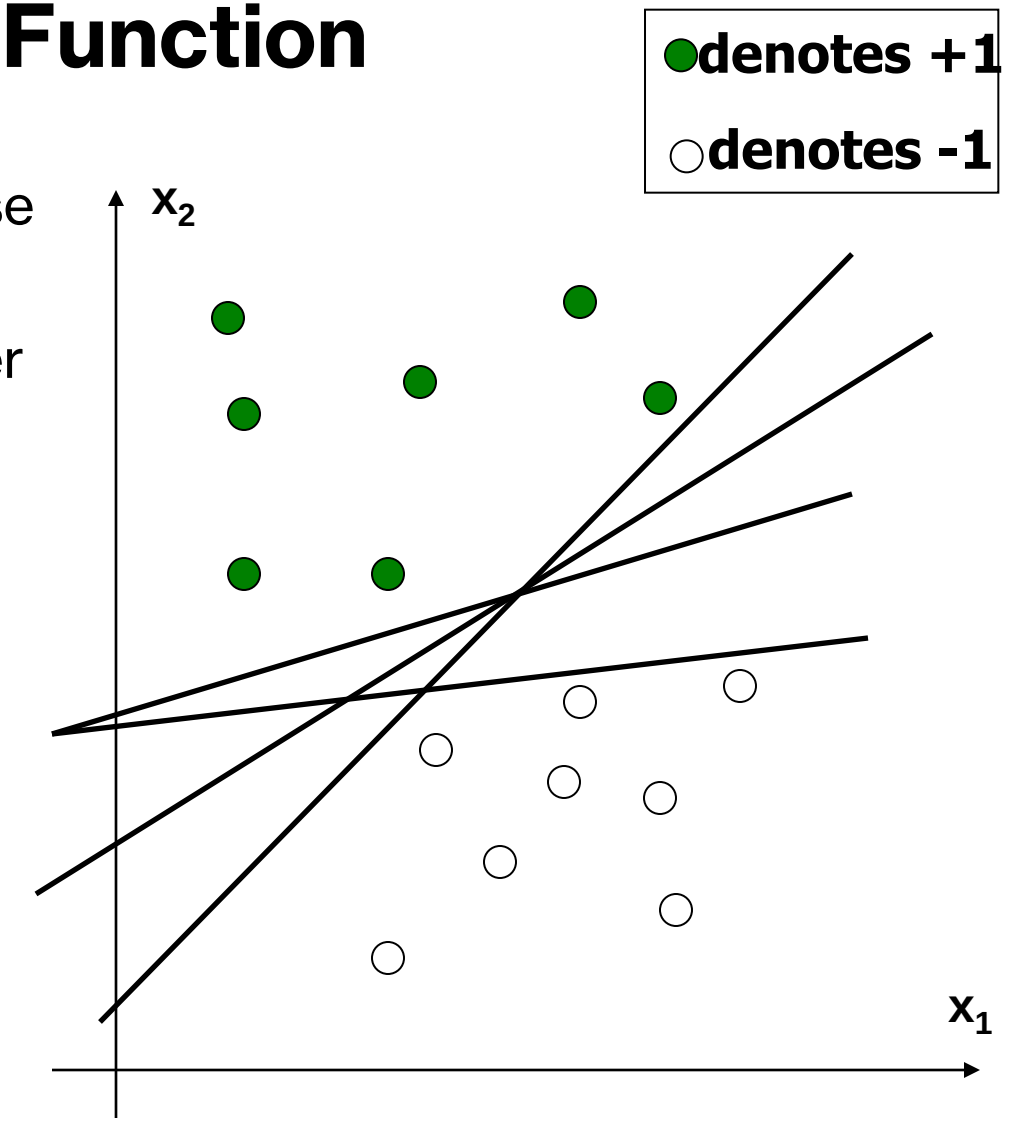
$$g(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b$$

Nonlinear Functions

- For Classification and Regression

SVM, Linear Discriminant Function

- How would you classify these points using a linear discriminant function in order to minimize the error rate?



- **Infinite number of answers!**
- **Which one is the best?**

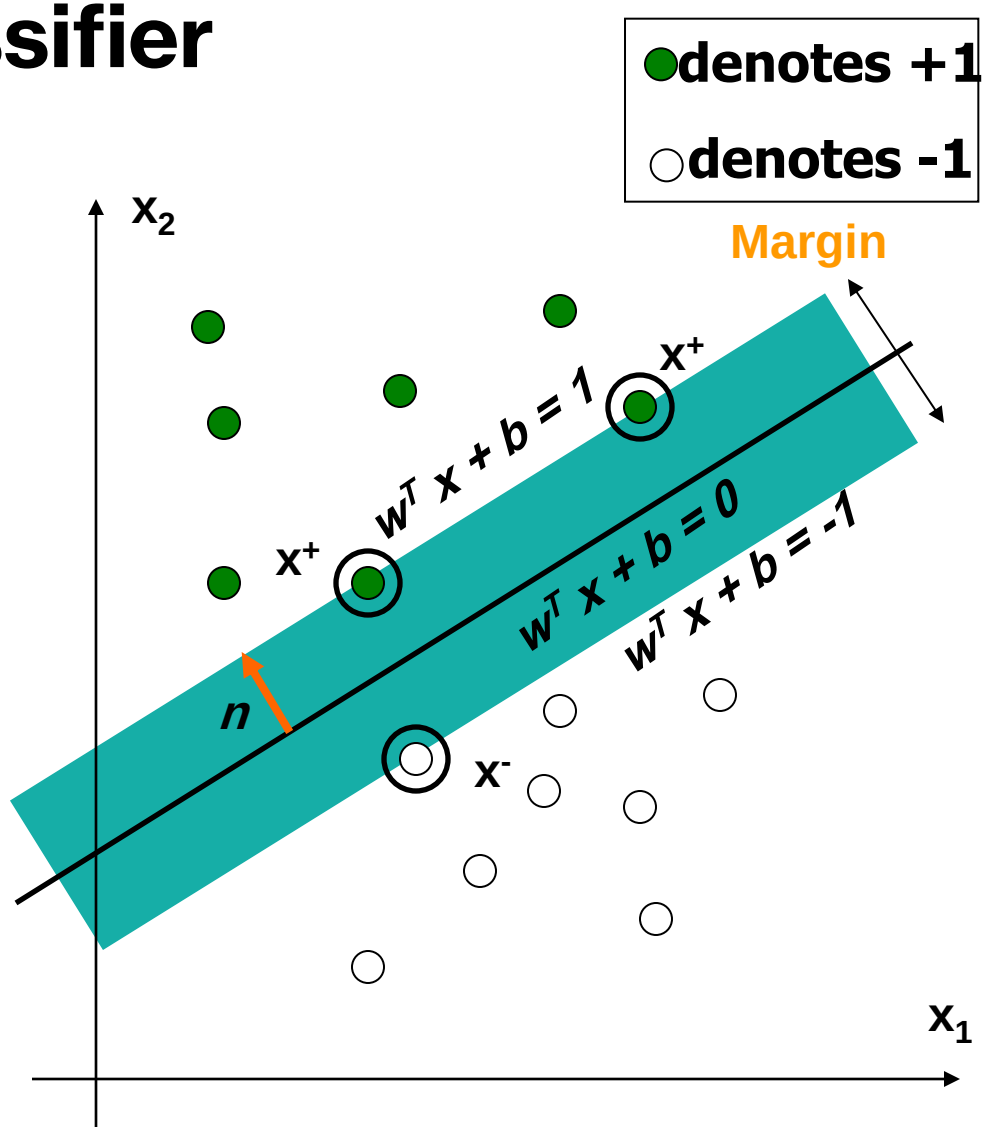
Large Margin Linear Classifier

- Formulation:

$$\text{minimize} \quad \frac{1}{2}\|\mathbf{w}\|^2$$

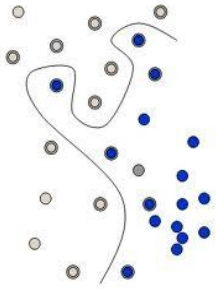
such that

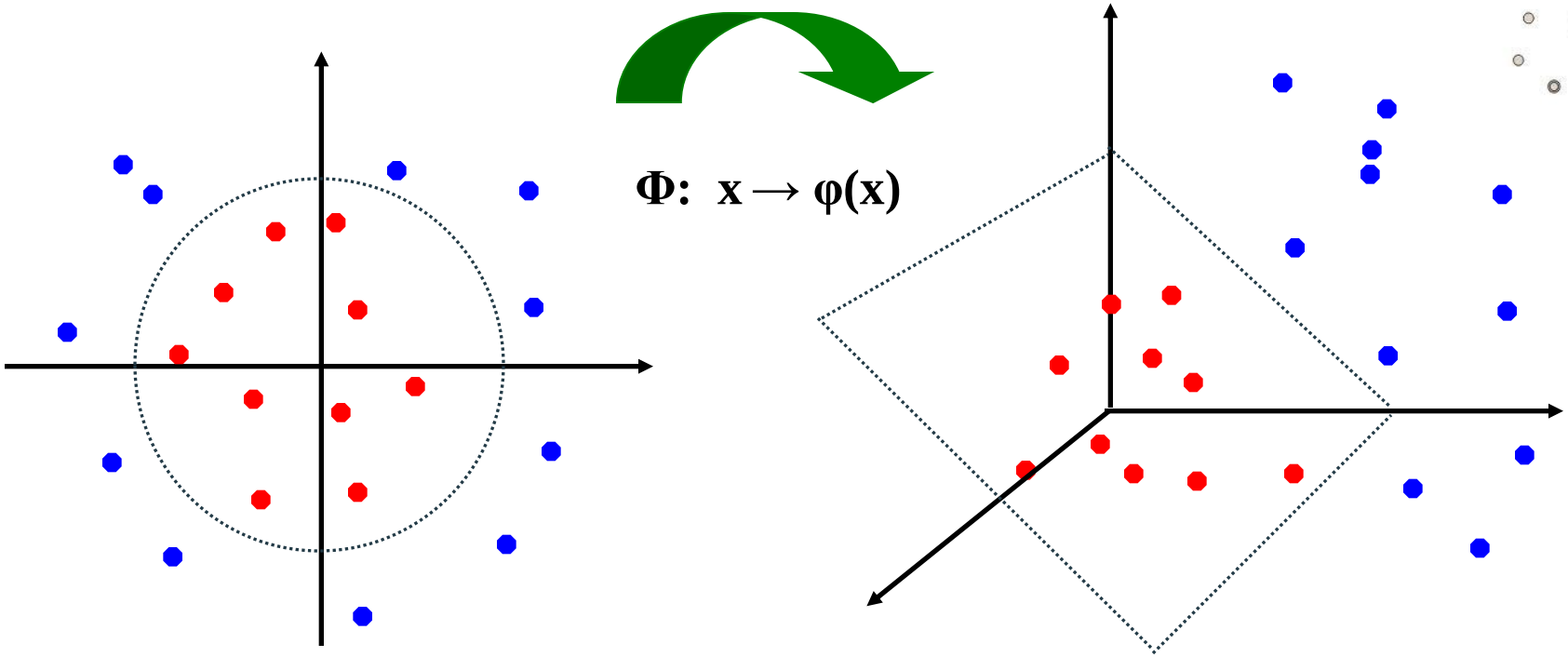
$$y_i(\mathbf{w}^T\mathbf{x}_i + b) \geq 1$$



Non-linear SVMs: Feature Space

- **General idea: the original input space can be mapped to some higher-dimensional feature space where the training set is separable:**





This slide is courtesy of *www.iro.umontreal.ca/~pift6080/documents/papers/svm_tutorial.ppt*

Neural Networks (NN)

NNs are a class of machine learning models inspired by the structure and function of the human brain. They consist of layers of interconnected nodes, called neurons, which process input data and learn to make predictions or classifications. Each connection between neurons has an associated weight that adjusts as the network learns, allowing the model to improve its performance over time.

1.Neurons:

1. The basic units of a neural network receive inputs, process them, and produce outputs. Each neuron can be thought of as a small computational unit that performs a mathematical operation.

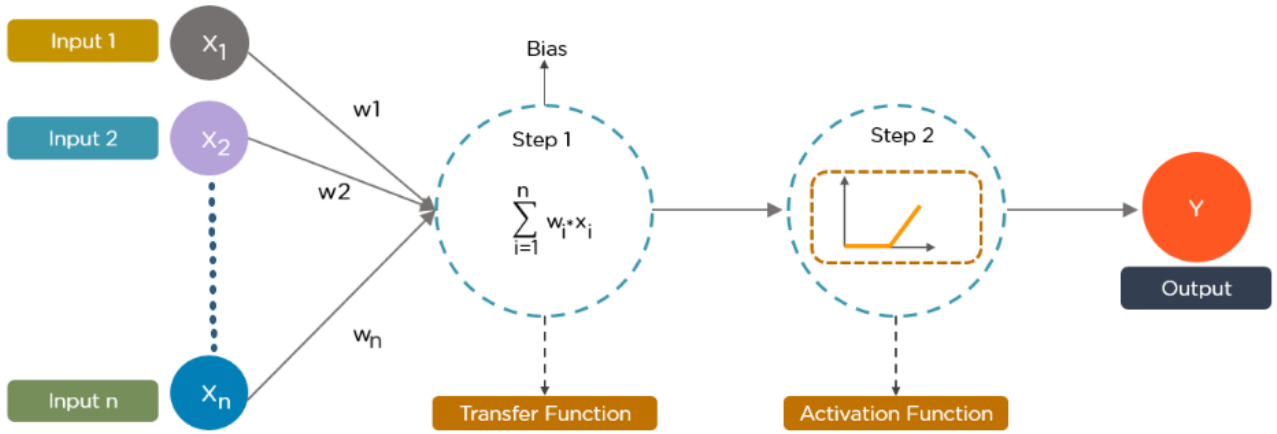
2.Layers

3.Weights:

1. Each connection between neurons has an associated weight that signifies the strength or importance of that connection. During training, these weights are adjusted to minimize the error in predictions.

4.Activation Functions:

1. Functions that determine the output of a neuron. They introduce non-linearity into the network, allowing it to learn complex patterns. Common activation functions include **ReLU** (Rectified Linear Unit), **Sigmoid**, and **Tanh**.



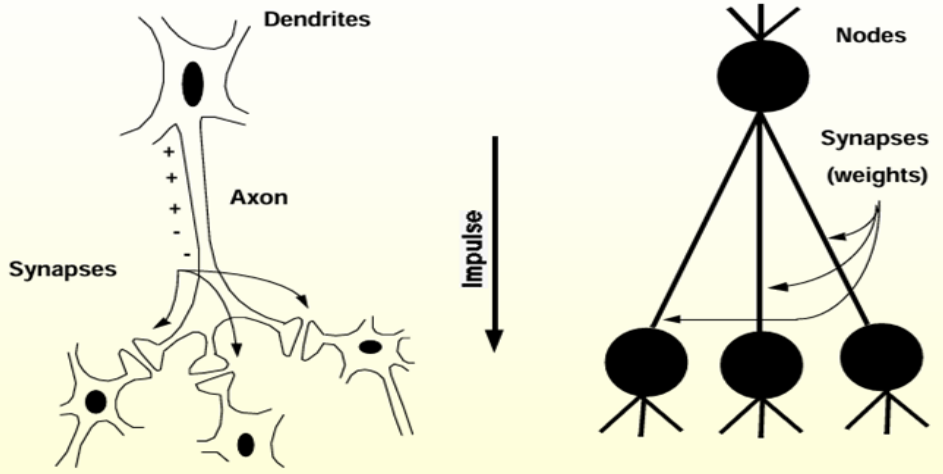
- Input Layer:
  - The input layer receives the raw data (e.g., images, text, numerical data) and passes it to the network for processing.
- Hidden Layers:
  - Hidden layers perform computations on the input data. Each hidden layer comprises neurons, and the number of layers and neurons determines the network's complexity.
  - Neurons in each layer apply an activation function to their inputs, which allows the network to learn complex patterns.
- Output Layer:
  - The output layer provides the final prediction or classification result. For instance, in a binary classification problem, the output might be a single value indicating the probability of the input belonging to one of two classes.

NNs

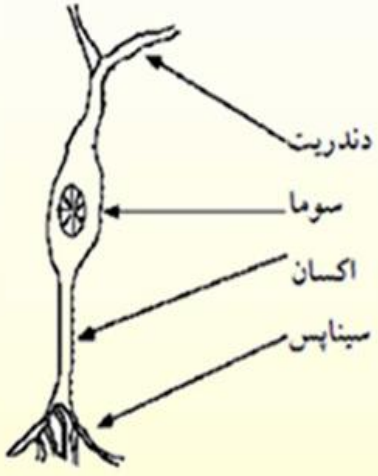
- **Learning Process in Neural Networks**

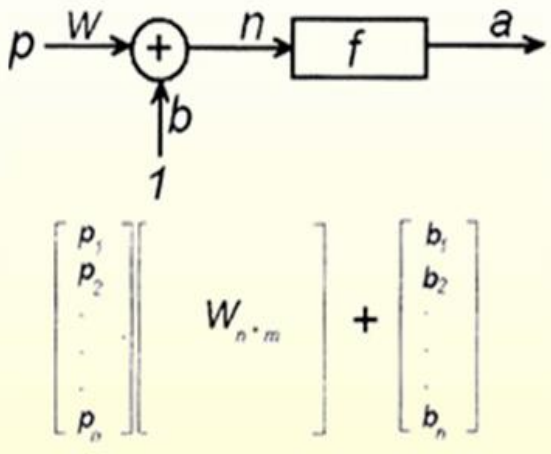
1. **Training**:
   - During training, the neural network learns from labeled data. The data is fed through the network, and predictions are made. These predictions are compared to the actual outcomes, and the error is computed.
2. **Error Backpropagation**:
   - This is the process where the error is propagated back through the network from the output layer to the input layer. This method, known as "backpropagation," adjusts the weights of the connections to reduce the error.
3. **Weight Update**:
   - Weights are updated using optimization algorithms like **Gradient Descent**. The goal is to adjust the weights in a way that minimizes the error and improves the accuracy of the predictions.

Neural Networks (NN)



- شبکه‌های عصبی را می‌توان با اغماض زیاد، مدل‌های الکترونیکی از ساختار عصبی مغز انسان نامید.
- مکانیسم فراگیری و آموزش مغز اساساً بر تجربه استوار است.





$$\begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_n \end{bmatrix} \begin{bmatrix} \quad \\ W_{n \cdot m} \\ \quad \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

Deep Learning

Intro

- زیرمجموعه ای از یادگیری ماشین که بر مبنای شبکه ی عصبی کار میکند.
- یادگیری عمیق که بیشتر در شبکه های عصبی استفاده میشود شامل ساخت نورون های عصبی است (شبکه‌هایی که همانند مغز انسان رفتار می‌کنند).
- این الگوریتم های یادگیری معماری چند لایه و بسیار شبیه به مغز دارند که می‌توانند اطلاعات را جمع‌آوری و نسبت به آن عکس‌العمل نشان دهند.
- آن‌ها می‌توانند ماهیت اجسام یا صدا را نیز درک کنند. به‌عنوان مثال، بازشناسی چشم انسان این چنین است که یک لایه ساده از نورون‌های وجود دارد که می‌تواند چیزهای ساده‌ای مثل شناسایی سطوح یک شکل خاص را شناسایی کند، سپس لایه بعدی می‌تواند این سطوح را در کنار یکدیگر قرار داده و شکل بزرگتری را شناسایی کند و درنهایت این اشکال می‌توانند در کنار یکدیگر باعث شناسایی یک جسم شوند.
- دانشمندان با الهام از همین ایده به نظریه معماری های عمیق و چند سطحی دست یافتند که در هر سطح یک بازنمایی انتزاعی از داده های خام اولیه ورودی را نشان میدهد و لایه های بالاتر مفاهیم مربوط به داده ها را بیان میدارد.

History

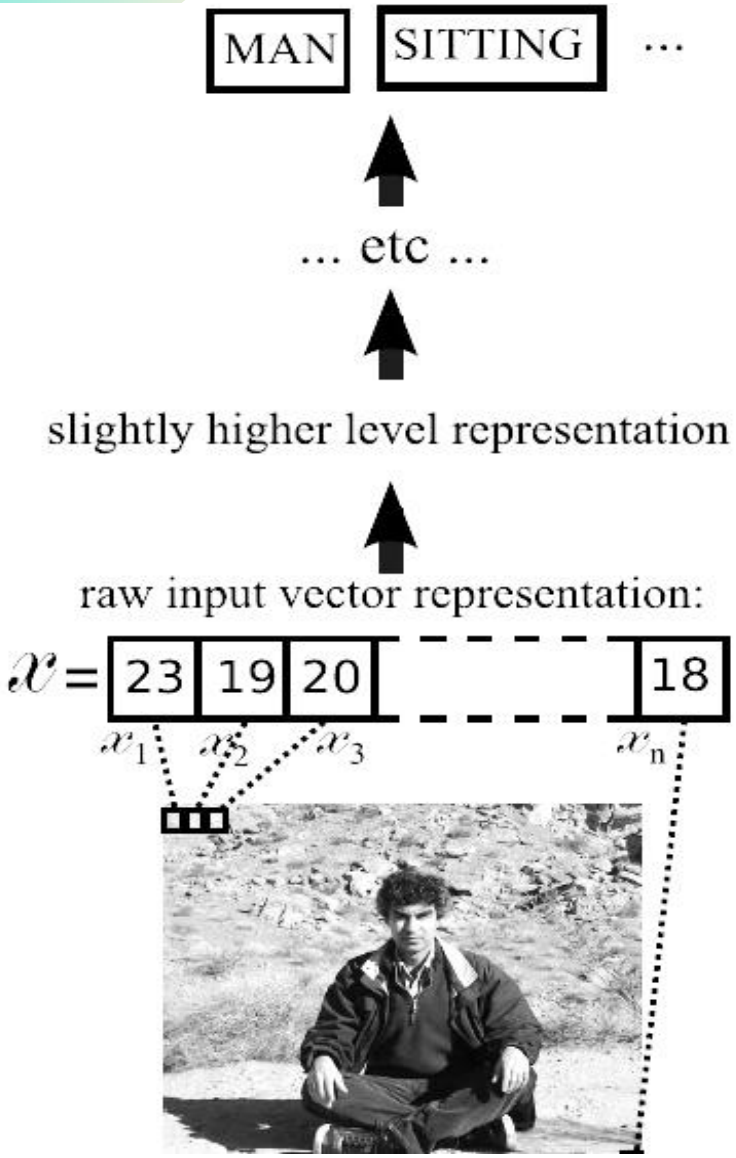
- یادگیری عمیق به طور خاص برای شبکه های عصبی مصنوعی ساخته شد و تاریخ استفاده از معماری های عمیق روی شبکه های عصبی به سال ۱۹۸۰، توسط فوکوشیما برمیگردد.
- در سال ۱۹۸۰شخصی به نام Yann LeCunو همکارانش با استفاده از شبکه های عصبی استاندارد پس انتشار که در سال ۱۹۷۴ارایه شده بود،توانستند یک شبکه ی عصبی عمیق بسازند .این شبکه برای تشخیص کد پستی نوشته شده روی پاکتهای نامه بود ولی زمان تقریبی آموزش این الگوریتم روی نمونه ها ۳روز بود و همین زمان طولانی امکان استفاده به صورت عمومی از این شبکه عصبی عمیق را فراهم نمیکرد .
- .تا اینکه در سال ۱۹۹۱، Jürgen Schmidhuberبه همراه دانشجوی خود به نام Sepp Hochreiterتوانست این کندی را حل کند .آنها متوجه تاثیر گرادیان در روی شبکه های عصبی عمیق و فرایند آموزش شدند .در سال ۱۹۹۱چندین شبکه ی عصبی برای تشخیص حروف و ارقام روی تصاویر ۲بعدی و ۳بعدی دست خط افراد مختلف آزمایش شد و از آن سال به بعد رویکرد مبتنی بر تصاویر سه بعدی به رسمیت شناخته شد.
- آقای Juyang Wengبه همراه همکارانش نظریه ای را بیان داشت که نشان داد، مغز انسان یک مدل یکپارچه و سه بعدی از اشیایی را که میبیند در ذهنش نگاه میدارد .در سال ۱۹۹۲، Schmidhuber، یک ایده بسیار مشابه برای حالت کلی تر از سلسله مراتب عمیق بدون نظارت از شبکه های عصبی مکرر، و همچنین مزایای این شبکه برای بالا بردن سرعت یادگیری نظارت شده را نشان داد.

Algorithm

یک معماری عمیق حالتی است که در آن سطوح مختلفی از بازنمایی ها وجود دارند، با سطوح بالاتر ساخته شده در بالای سطوح پایین تر(ورودی خام اصلی در پایین ترین سطوح است).

سطوح بالاتر نشان دهنده مفاهیم انتزاعی تری هستند و انتزاع کمتر، در سطوح پایین تر قرار دارند.

فرض کنید تصویر یک خیابان که ماشینی در آن در حال حرکت است، آنگاه این تصویر به عنوان ورودی خام در لایه ی سطح اول است و در سطوح بالاتر به صورت انتزاعی اشیا موجود در تصویر بازنمایی میشود.



Example

In the image, for each pixel, a value of the gray color feature is stored in the input vector x. In the figure, you can see that by recognizing the objects in the gray image, it has come to the knowledge that there is a man in a sitting position in the image.

Compare on MNIST

Error Rate	Processing Function	Name
7.6	Deskewing	**Linear classifier**
0.52	Shiftable edges	**K-Nearest Neighbors**
0.56	Deskewing	**Support vector machine**
0.35	None	**Neural network**
0.23	Width normalizations	**Convolutional neural network**

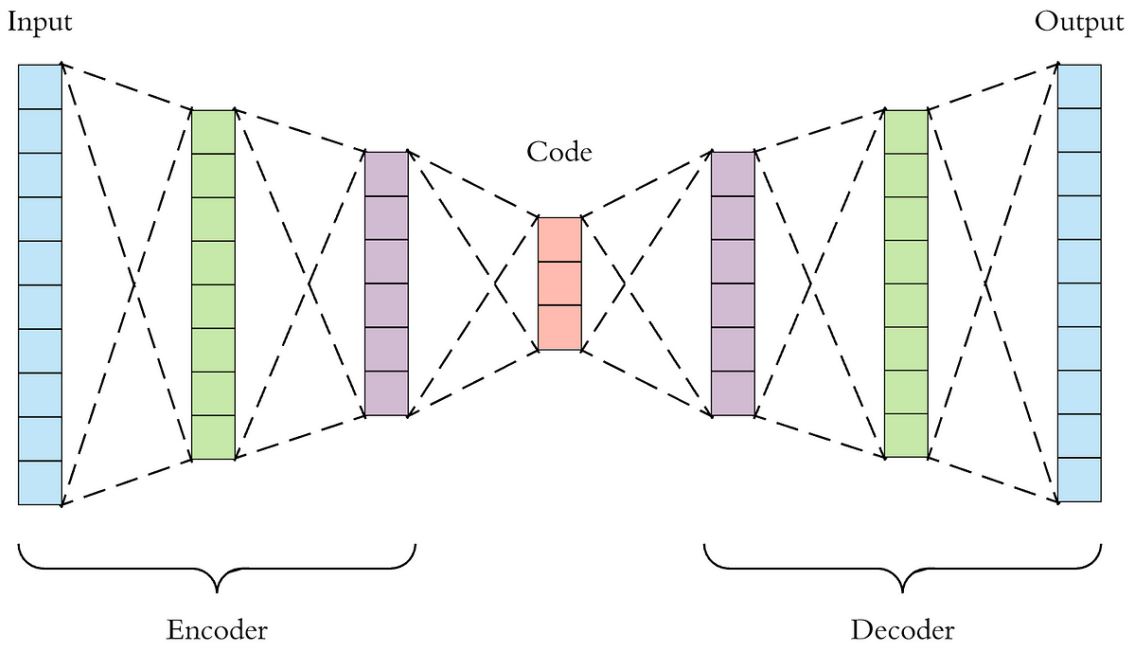
Top Deep Learning Algorithms

- **Deep Belief Networks (DBNs)**
  - Layer-by-Layer Training: DBNs are trained in a greedy, layer-by-layer fashion. Each layer is trained as a Restricted Boltzmann Machine (RBM), which learns to reconstruct its input.
  - Fine-tuning: After pretraining the layers, the entire network can be fine-tuned using backpropagation for specific tasks.
- **Autoencoders**
  - Encoder: Maps the input data to a lower-dimensional latent space representation.
  - Latent Space: Represents the compressed version of the input data.
  - Decoder: Reconstructs the input data from the latent representation.
  - Training: The network minimizes the difference between the input and the reconstructed output.
- **Transformer Networks**
  - Self-Attention Mechanism: This mechanism computes the importance of each part of the input relative to every other part, enabling the model to weigh the significance of different words in a sentence differently.
  - Positional Encoding: Adds information about the position of words in the sequence since self-attention doesn't inherently capture sequence order.
  - Encoder-Decoder Architecture: Consists of an encoder that processes the input sequence and a decoder that generates the output sequence. Each consists of multiple layers of self-attention and feed-forward networks.

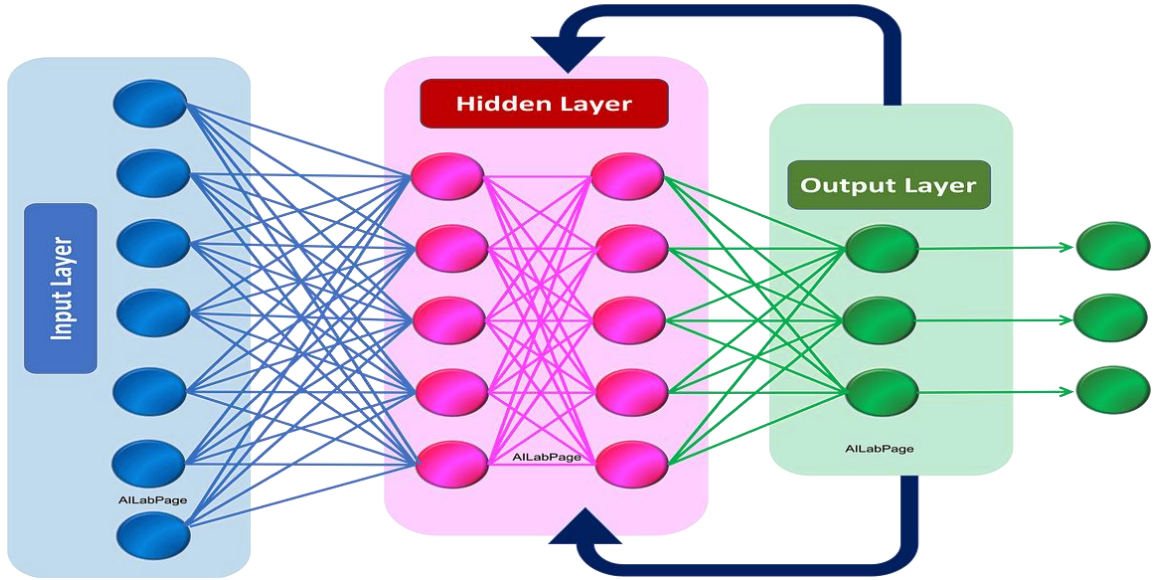
Top Deep Learning Algorithms

- **Recurrent Neural Networks (RNNs)**
  - Hidden State: At each time step, the hidden state is updated based on the current input and the previous hidden state. This allows the network to maintain a memory of past inputs.
  - Output: The hidden state generates an output at each time step. The network is trained using backpropagation through time (BPTT) to minimize prediction error.
- **Convolutional Neural Networks (CNNs)**
  - Convolutional Layer: This layer applies a set of filters (kernels) to the input image, where each filter slides (convolves) across the image to produce a feature map. This helps detect various features such as edges, textures, and patterns.
  - Pooling Layer: This layer reduces the dimensionality of the feature maps while retaining the most essential information. Common types include max pooling and average pooling.
  - Fully Connected Layer: After several convolutional and pooling layers, the output is flattened and fed into one or more fully connected (dense) layers, culminating in the output layer that makes the final classification or prediction.
- **Graph Neural Networks (GNNs)**
  - Graph Representation: Nodes represent entities, and edges represent relationships between entities.
  - Message Passing: Nodes aggregate information from their neighbors to update their representations. This process can be repeated for several iterations.
  - Readout Function: After message passing, a readout function aggregates node representations to produce a graph-level representation for tasks like classification or regression.

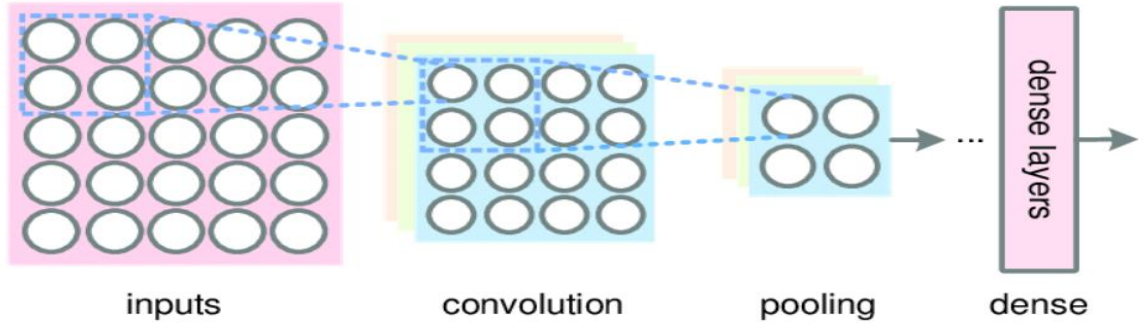
Autoencoders



Recurrent Neural Networks



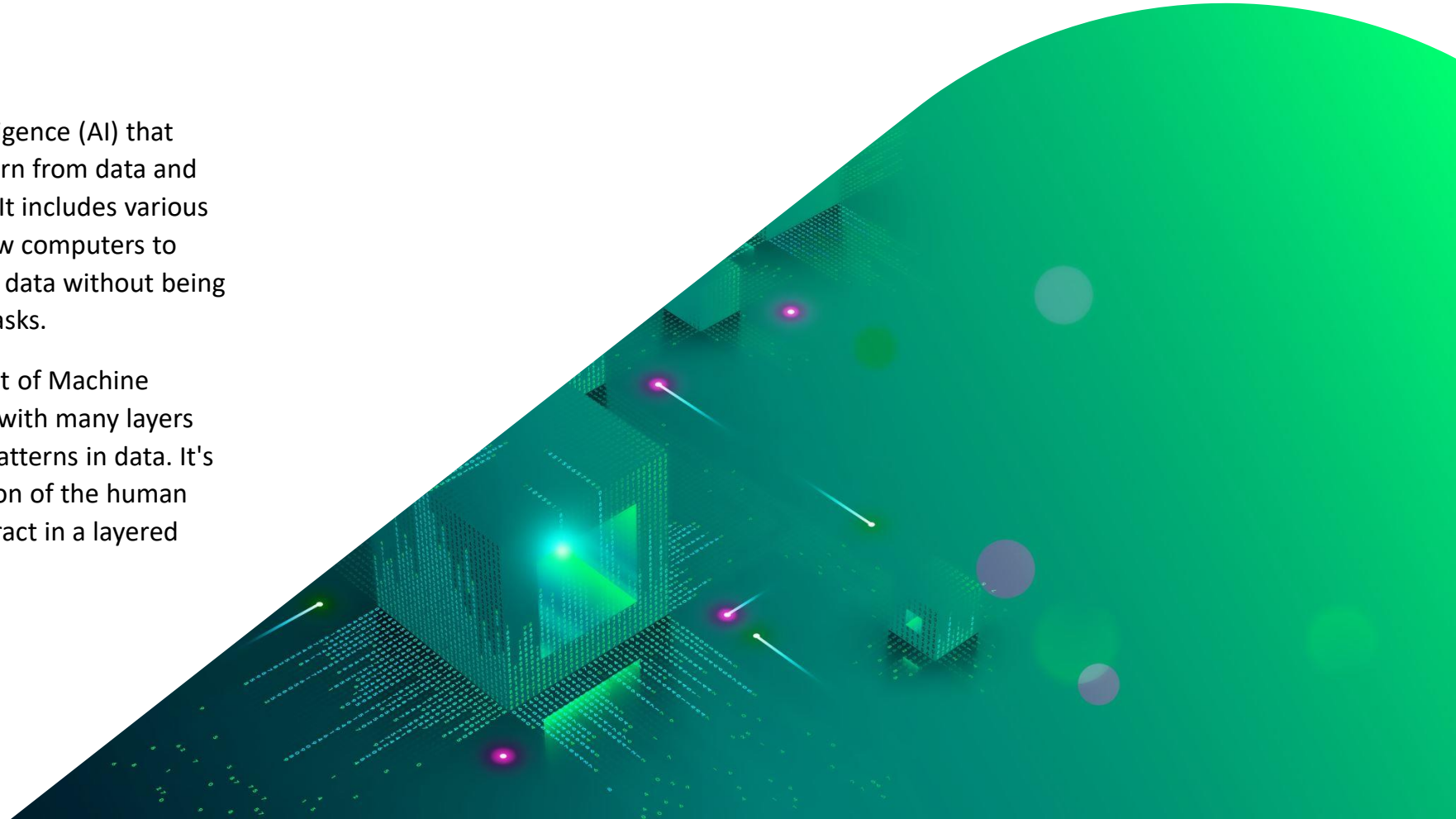
CNN



-1	0	1
-2	0	2
-1	0	1

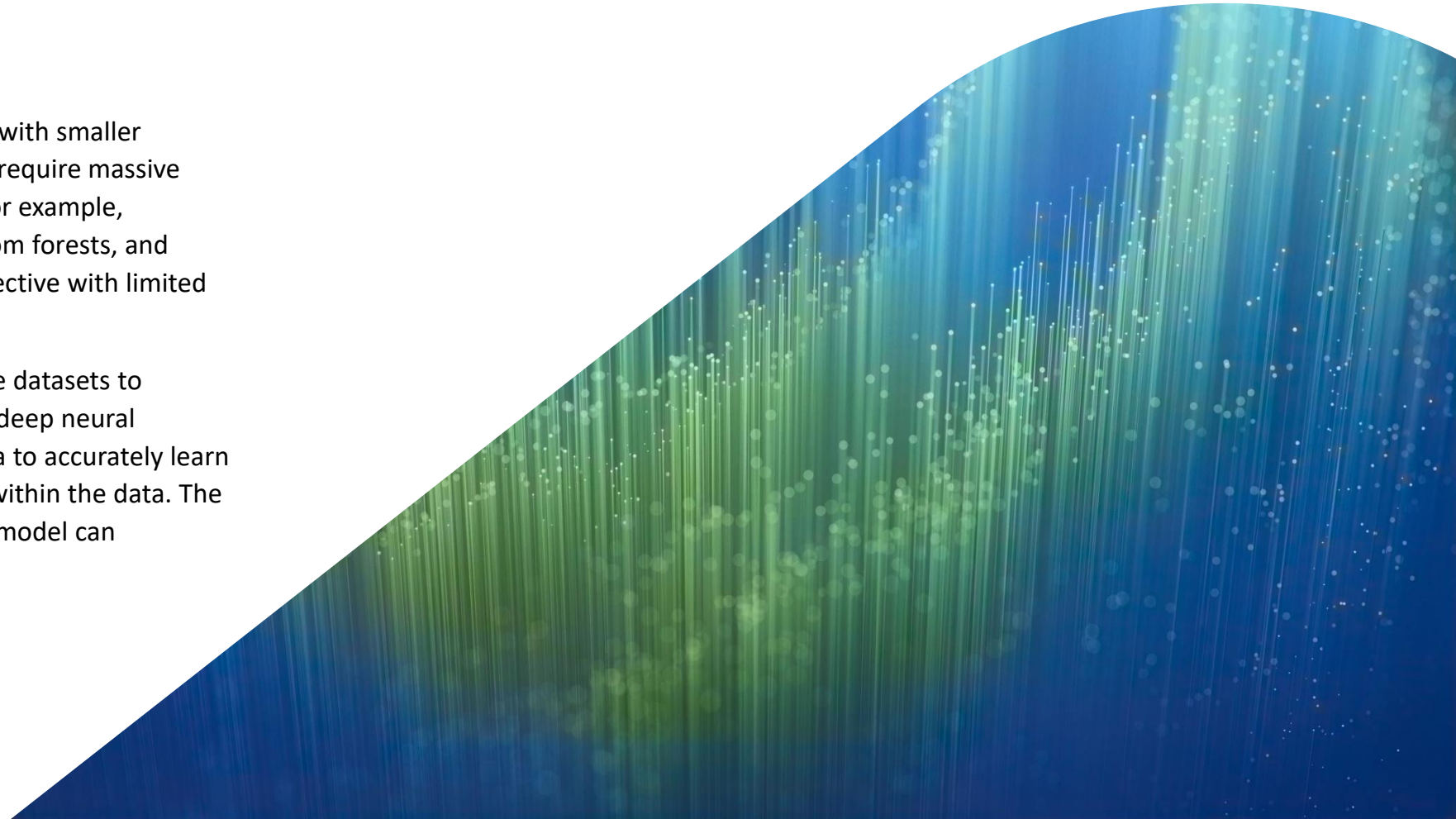
ML vs DL

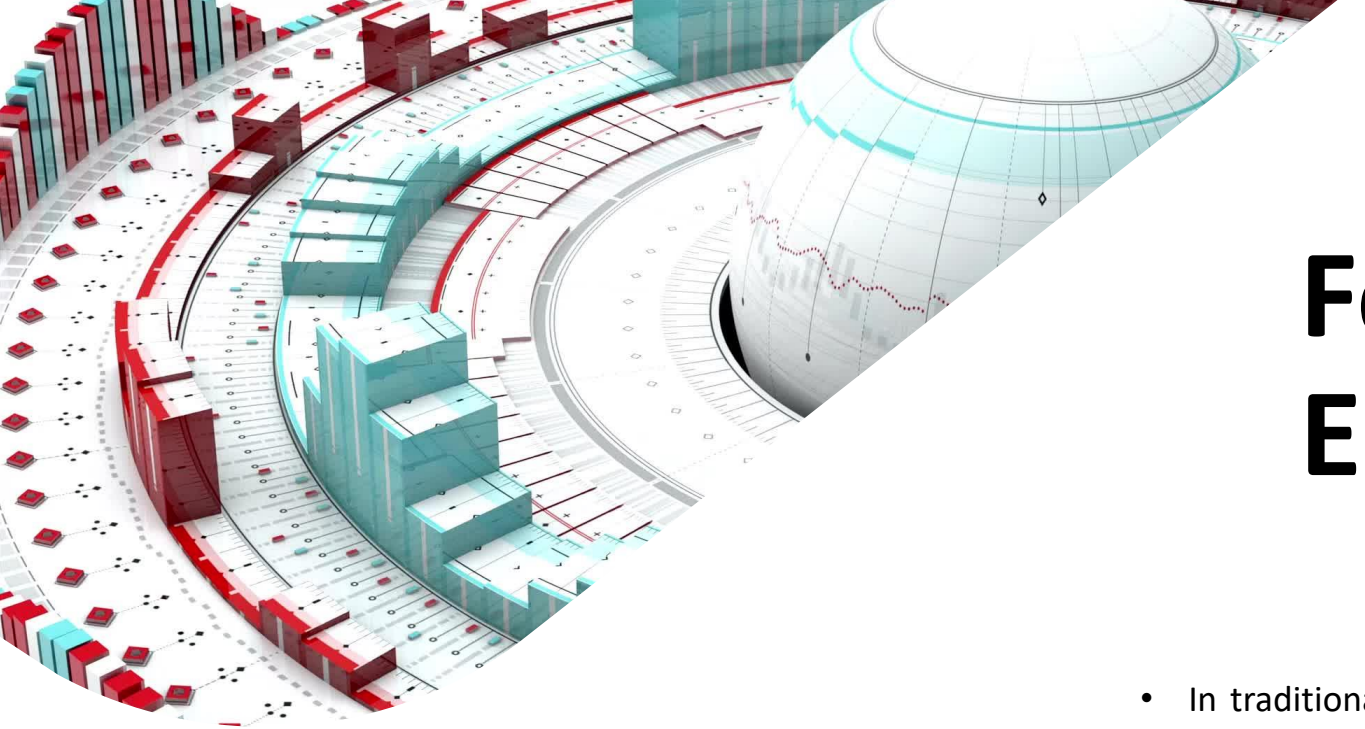
- ML is a broad field of Artificial Intelligence (AI) that focuses on building systems that learn from data and make decisions based on that data. It includes various algorithms and techniques that allow computers to learn from and make predictions on data without being explicitly programmed for specific tasks.
- Deep Learning is a specialized subset of Machine Learning that uses neural networks with many layers (hence "deep") to model complex patterns in data. It's inspired by the structure and function of the human brain, specifically how neurons interact in a layered network.



Data Requirements

- Traditional ML algorithms can work with smaller datasets, and many of them do not require massive amounts of data to perform well. For example, algorithms like decision trees, random forests, and support vector machines can be effective with limited data.
- DL algorithms typically require large datasets to perform effectively. This is because deep neural networks need vast amounts of data to accurately learn the intricate patterns and features within the data. The more data available, the better the model can generalize.



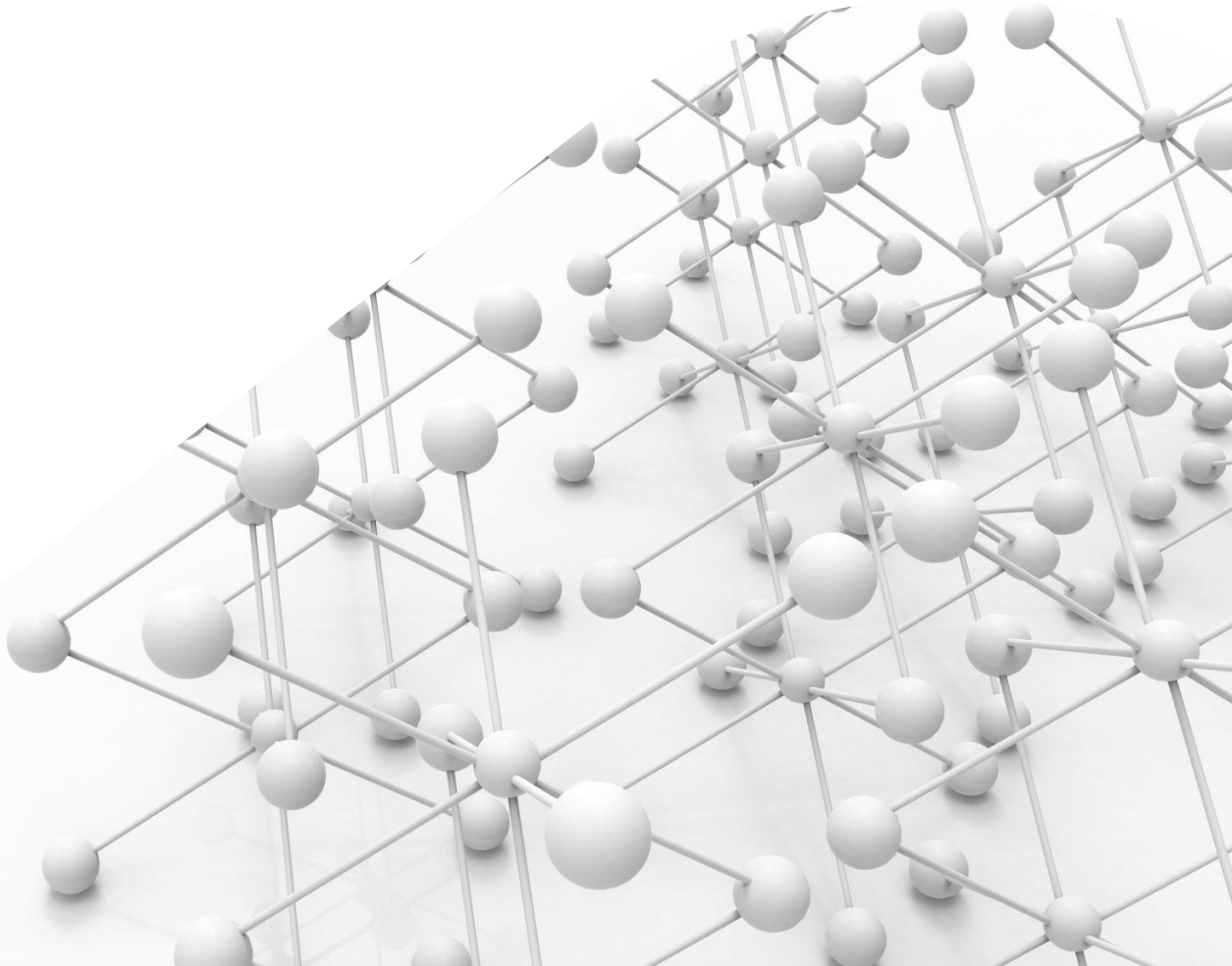


Feature Engineering

- In traditional ML, a significant amount of effort is often put into feature engineering, which involves manually selecting, modifying, or creating features that can help the algorithm make better predictions. The success of an ML model can heavily depend on the quality and relevance of the features used.
- DL models automatically learn to extract relevant features from raw data. This is one of the key advantages of DL—it reduces the need for manual feature engineering. For instance, in image recognition, a deep learning model can learn to identify edges, textures, and shapes in the data without manual intervention.

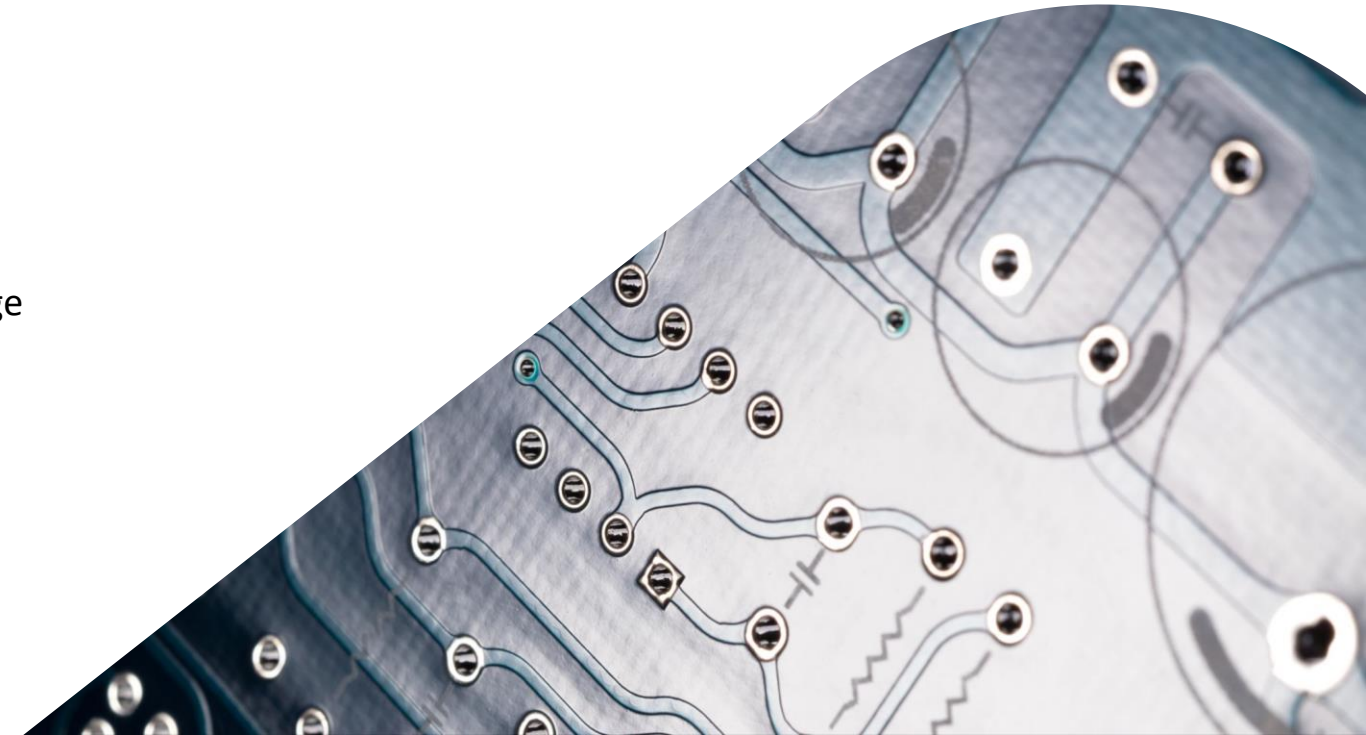
Complexity of Models

- ML models can range from simple to moderately complex, depending on the algorithm used. For example, a linear regression model is relatively simple, while a random forest is more complex but still interpretable.
- DL models, especially deep neural networks with many layers, are much more complex. They consist of numerous layers of neurons that process data in multiple stages, making them powerful but also more challenging to interpret.



Computational Requirements

- Traditional ML models are generally less computationally intensive, making them easier and faster to train on standard hardware. They can often be trained on CPUs.
- DL models are highly computationally demanding, especially as the number of layers and parameters increases. Training deep neural networks often requires specialized hardware such as Graphics Processing Units (GPUs) or Tensor Processing Units (TPUs) to process large amounts of data efficiently.



Summary

- Machine Learning is a broad field that encompasses various techniques for building systems that learn from data. It includes many different types of algorithms, from simple linear regression to more complex ensemble methods.
- Deep Learning is a more specialized area within ML, focused on using deep neural networks to model complex patterns in large datasets. It is particularly powerful for tasks involving unstructured data like images, text, and audio, but it requires more data, and computational power, and is often harder to interpret.

In short, while all deep learning is machine learning, not all machine learning is deep learning.

Deep Learning in Computer Vision

1

Image Classification:

- Example: Automatically tagging photos on social media platforms like Facebook using models trained to recognize objects, scenes, and faces in images.
- Impact: Enhances user experience by simplifying the organization and search of photos.

2

Object Detection:

- Example: Self-driving cars use object detection to identify and track objects such as pedestrians, other vehicles, and road signs in real time.
- Impact: Improves safety and navigation in autonomous vehicles.

3

Facial Recognition:

- Example: Security systems use facial recognition to authenticate users, such as Apple's Face ID.
- Impact: Enhances security in devices, banking, and public surveillance.

4

Medical Imaging:

- Example: Deep learning models analyze X-rays, MRIs, and CT scans to detect diseases like cancer or neurological disorders.
- Impact: Increases accuracy in early diagnosis and treatment planning, reducing human error.

Deep Learning in Autonomous Systems

- Self-Driving Cars
  - Companies like Tesla and Waymo use deep learning for perception, decision-making, and control in autonomous vehicles.
  - Potential to reduce traffic accidents, enhance mobility, and lower transportation costs.
- Drones
  - Deep learning enables drones to autonomously navigate and perform tasks like aerial photography, surveillance, and delivery.
  - Expand the capabilities of drones in various industries, from agriculture to logistics.
- Robotics
  - Robots in manufacturing and logistics use deep learning for tasks such as object manipulation, quality inspection, and autonomous navigation in warehouses.
  - Increases efficiency, reduces labor costs and improves precision in manufacturing processes.

Deep Learning in Finance

Algorithmic Trading

- Deep learning models analyze market data to identify patterns and make real-time trading decisions.
- Increases trading efficiency and profitability by executing trades faster than humans.

Fraud Detection

- Banks and financial institutions use deep learning to detect fraudulent transactions by recognizing unusual patterns in transaction data.
- Reduces financial losses and enhances security in online transactions.

Risk Management

- Deep learning models assess credit risk by analyzing vast amounts of financial data, including customer history and market trends.
- Improves the accuracy of credit scoring and risk assessment..

TensorFlow for Deep Learning

- Overview: Developed by Google, TensorFlow is one of the most popular open-source deep learning frameworks. It provides a comprehensive ecosystem for building, training, and deploying deep learning models.

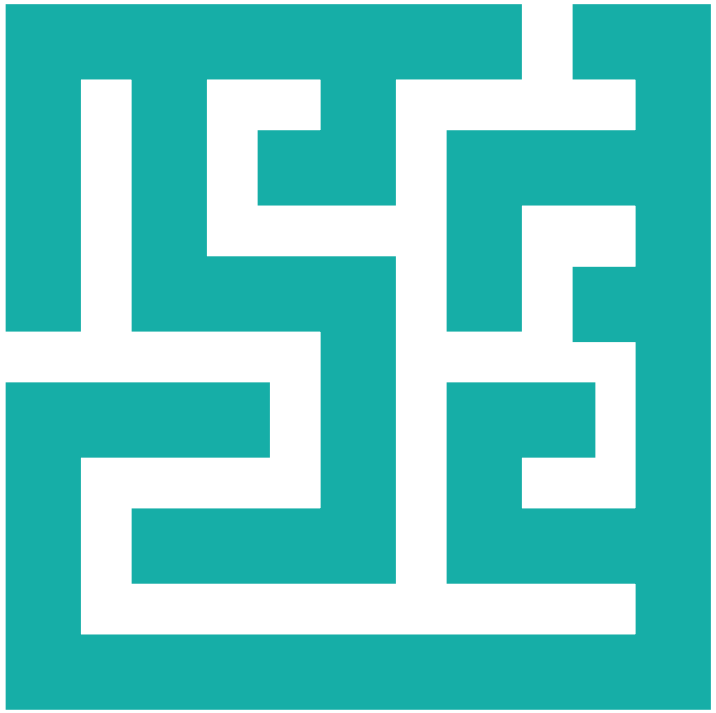
- Features: Supports deep learning, machine learning, and AI across various platforms (desktops, servers, mobile devices, and edge devices).

  - TensorFlow Hub for sharing pre-trained models.

  - TensorFlow Lite for deploying models on mobile and IoT devices.

  - TensorFlow Extended (TFX) for production-level ML pipelines.

- Use Cases: Image recognition, natural language processing, recommendation systems.



Keras for Deep Learning

Overview: Keras is an open-source neural network API written in Python, designed to enable fast experimentation. It is user-friendly, modular, and easy to extend. Keras is now part of the TensorFlow library.

- Features:

  - High-level API for building and training models.

  - Easy to switch between different backends like TensorFlow, Theano, and CNTK (although TensorFlow is the primary backend now).

  - Supports Convolutional Networks, Recurrent Networks, and more.

- Use Cases: Rapid prototyping, educational purposes, model deployment.

Caffe for Deep Learning

- Overview: Developed by the Berkeley Vision and Learning Center (BVLC), Caffe is an open-source deep learning framework primarily focused on speed and modularity.

- Features:

  - Specializes in convolutional neural networks (CNNs) for visual recognition tasks.

  - Highly efficient, making it suitable for production use.

  - Extensive support for various pre-trained models via Caffe Model Zoo.

- Use Cases: Image classification, convolutional neural networks, transfer learning.

PyTorch for Deep Learning

- Overview: Developed by Facebook's AI Research lab (FAIR), PyTorch is another leading open-source deep learning framework known for its flexibility and ease of use, particularly in research settings.

- Features:

  - Dynamic computation graph (eager execution), making it intuitive and easy to debug.

  - Strong community support with extensive libraries and tools.

  - PyTorch Lightning for managing complex model training.

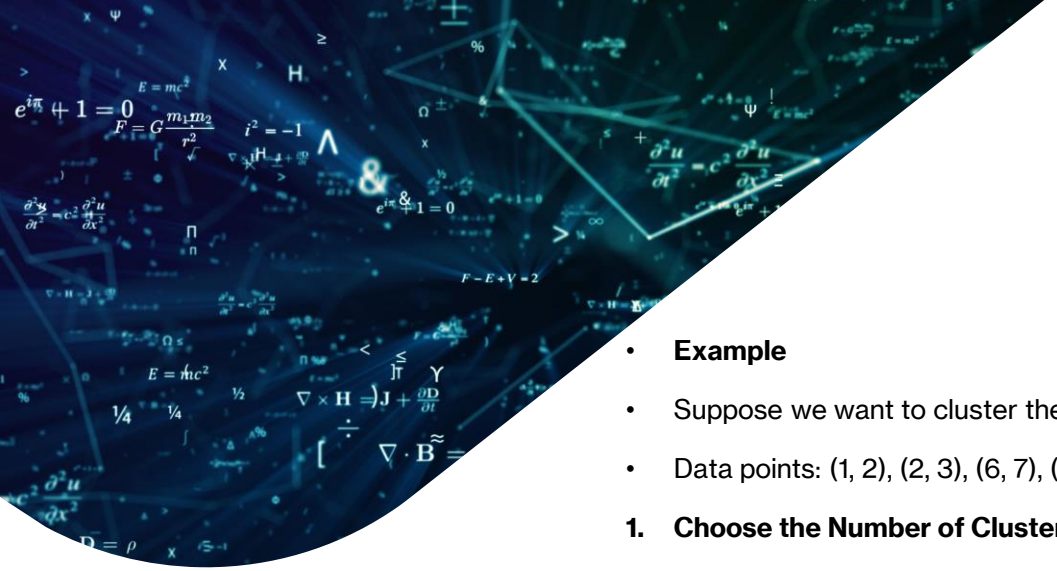
  - TorchServe for deploying PyTorch models at scale.

- Use Cases: Computer vision, natural language processing, reinforcement learning.

K-means Algorithm

- The K-means algorithm is a popular method used in machine learning and data analysis for clustering data. The goal of the algorithm is to partition data into K groups (or clusters) so that data points in the same cluster are more similar to each other than to those in other clusters.
- **Steps of the K-means Algorithm**

1. **Choose the Number of Clusters (K):** Determine the number of clusters you want to divide your data into.
2. **Initialize Cluster Centers:** Randomly select K data points as the initial cluster centers.
3. **Assign Data Points to the Nearest Cluster Center:** For each data point, assign it to the nearest cluster center based on a distance metric (typically Euclidean distance).
4. **Update Cluster Centers:** After assigning all data points to clusters, recalculate the position of each cluster center as the mean of the data points assigned to that cluster.
5. **Repeat Steps 3 and 4:** Repeat the process of assigning data points to the nearest cluster center and updating cluster centers until the positions of the cluster centers stabilize (i.e., they do not change significantly) or until a stopping criterion is met.



K-means Algorithm

- **Example**
- Suppose we want to cluster the following 2D data points into two clusters (K=2):
- Data points: (1, 2), (2, 3), (6, 7), (7, 8), (8, 9)

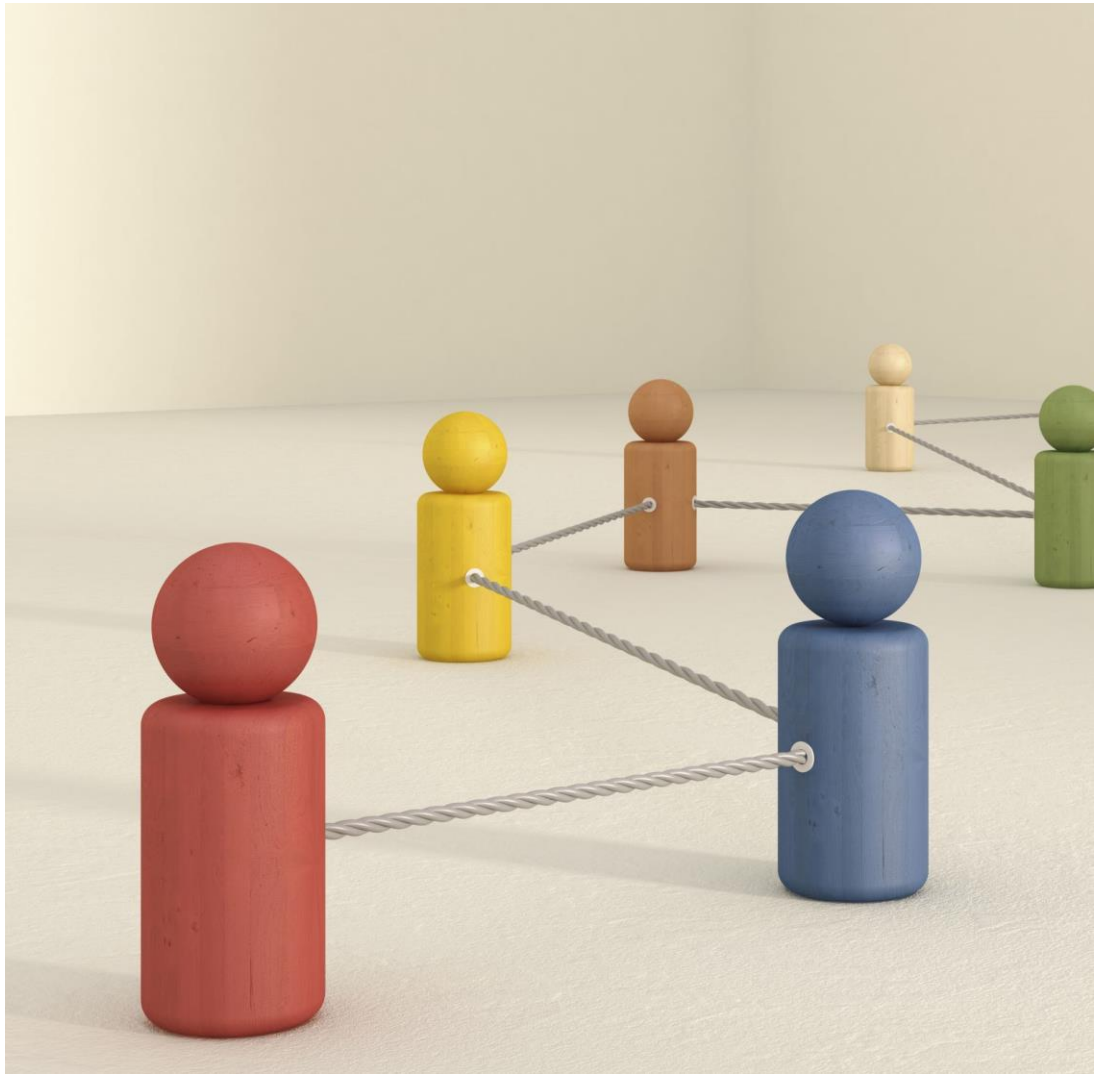
1. **Choose the Number of Clusters:** K = 2
2. **Initialize Cluster Centers:** Suppose we randomly choose:
    1. Cluster Center 1: (1, 2)
    2. Cluster Center 2: (6, 7)
3. **Assign Data Points to Nearest Cluster Center:**
    1. (1, 2) is closer to Cluster Center 1.
    2. (2, 3) is closer to Cluster Center 1.
    3. (6, 7) is closer to Cluster Center 2.
    4. (7, 8) is closer to Cluster Center 2.
    5. (8, 9) is closer to Cluster Center 2.
4. After this step, the clusters are:
    1. Cluster 1: [(1, 2), (2, 3)]
    2. Cluster 2: [(6, 7), (7, 8), (8, 9)]
5. **Update Cluster Centers:**
    1. New Center for Cluster 1: Mean of points in Cluster 1 = ((1+2)/2, (2+3)/2) = (1.5, 2.5)
    2. New Center for Cluster 2: Mean of points in Cluster 2 = ((6+7+8)/3, (7+8+9)/3) = (7, 8)
6. **Repeat:** With the new centers, reassign data points to the nearest cluster center and update the cluster centers. This process continues until the cluster centers no longer change significantly.

Fuzzy Logic

- Fuzzy logic is a mathematical framework used to model complex systems with uncertainty, vagueness, and imprecision. It mimics human reasoning, allowing for reasoning in a way that is closer to natural language and human decision-making. Instead of a strict true/false evaluation, fuzzy logic enables the handling of situations where truth values are on a continuum between 0 and 1.

- 1965- Professor Lotfizade.

- Application:
  1. Industry Control Systems
  2. Traffic Control System
  3. Train Speed Traffic Control System
  4. Engine Control System
  5. Expert system
  6. Decision-Making Systems



Fuzzy Rules vs Traditional If-Then Rules

How they handle conditions and conclusions	
Conditions:	• Traditional if-then rules use crisp, binary conditions that are either completely true or completely false. For example: "If temperature is 25°C, then...“ • Fuzzy rules use fuzzy conditions that can be partially true to varying degrees. For example: "If temperature is hot, then..." where "hot" is a fuzzy condition that can be true to some degree between 0 and 1.
Conclusions:	• Traditional if-then rules produce crisp, binary conclusions that are either completely true or completely false. • Fuzzy rules produce conclusions that can also be partially true to varying degrees. The conclusion of a fuzzy rule is a fuzzy set that specifies the degree to which the output variable should take on different values.
Reasoning:	• Traditional if-then rules use Boolean logic for reasoning, where conditions are either true or false. • Fuzzy rules use fuzzy logic for reasoning, where conditions can be partially true. The degree to which the conclusion is true depends on the degree to which the conditions are satisfied.

Fundamental Concepts of Fuzzy Logic

1. Fuzzy Sets: The foundation of fuzzy logic, fuzzy sets allow for partial membership, meaning an element can belong to a set to a certain degree, expressed by a value between 0 and 1.

- Example: In a fuzzy set representing "tall people," someone who is 180 cm might belong to the set with a membership value of 0.7, while someone who is 170 cm might belong with a membership value of 0.4.

2. Membership Function: A membership function defines how each element in the input space is mapped to a membership value (degree of truth) between 0 and 1. Different shapes of membership functions (e.g., triangular, trapezoidal, Gaussian) can be used depending on the problem.

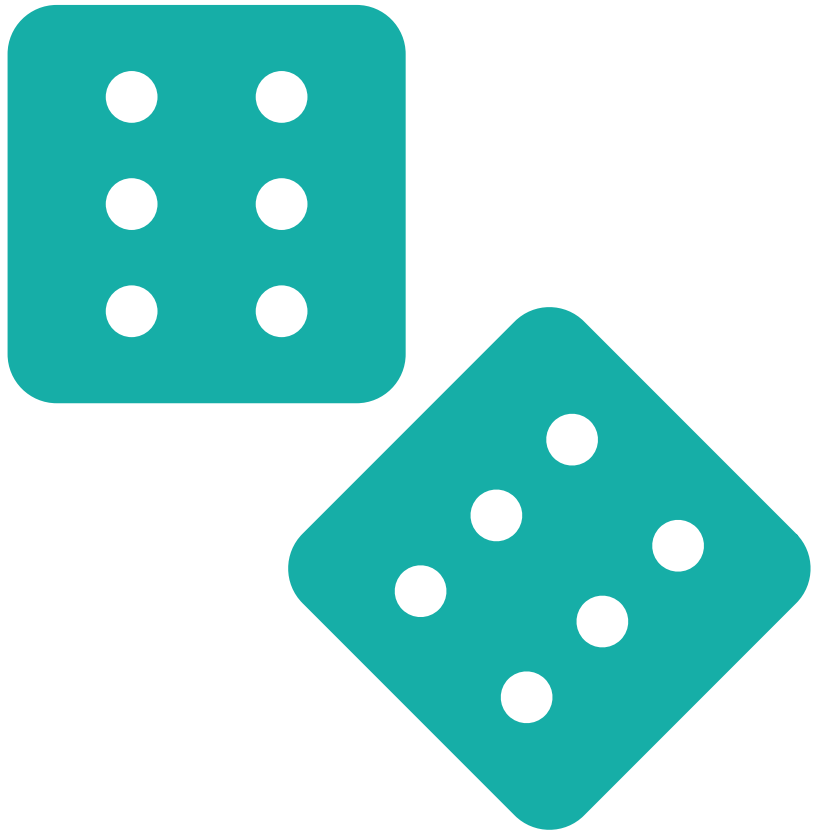
- Example: For the fuzzy set "tall people," a membership function might be defined so that heights between 160 cm and 180 cm have gradually increasing membership values from 0 to 1.

3. Linguistic Variables: These are variables whose values are words or sentences in natural language rather than numerical values. Each linguistic variable can have a set of linguistic terms (e.g., "low," "medium," "high") that correspond to fuzzy sets.

- Example: The linguistic variable "Temperature" could have terms like "cold," "warm," and "hot," each represented by fuzzy sets.

4. Fuzzy Rules: Fuzzy logic systems use a set of IF-THEN rules to model the relationship between input and output variables. These rules describe how the system should behave under different conditions, using fuzzy sets and linguistic variables.

- Example: A fuzzy rule might be: "IF temperature is warm AND humidity is high, THEN fan speed is medium."



Continue...

5. Fuzzy Inference System (FIS):

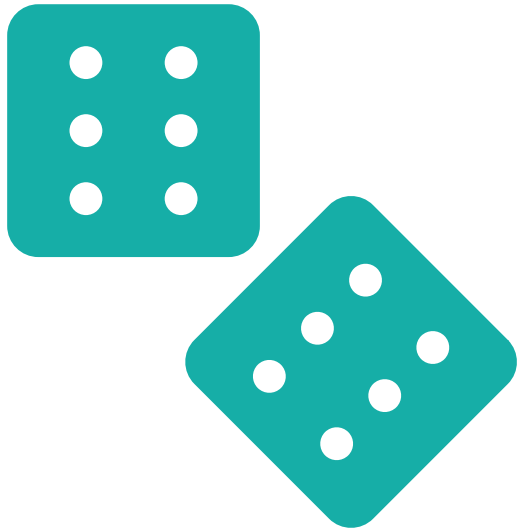
- This is the process of mapping inputs to outputs using fuzzy logic. The FIS consists of three main steps:

  - Fuzzification: Converts crisp inputs into degrees of membership in fuzzy sets.

  - Inference: Applies the fuzzy rules to the fuzzified inputs to generate fuzzy outputs.

  - Defuzzification: Converts the fuzzy output back into a crisp value to make a concrete decision or action.

  - Example: In a climate control system, the FIS takes temperature and humidity as inputs, applies the fuzzy rules, and determines the appropriate fan speed.



6. Defuzzification: The process of converting the fuzzy output of a fuzzy inference system into a single crisp value. Common defuzzification methods include the Centroid method (calculating the center of gravity of the output distribution) and the Maximum method (selecting the output with the highest membership value).

  - Example: After applying fuzzy rules to determine the fan speed, defuzzification might result in setting the fan speed to 75% of its maximum capacity.

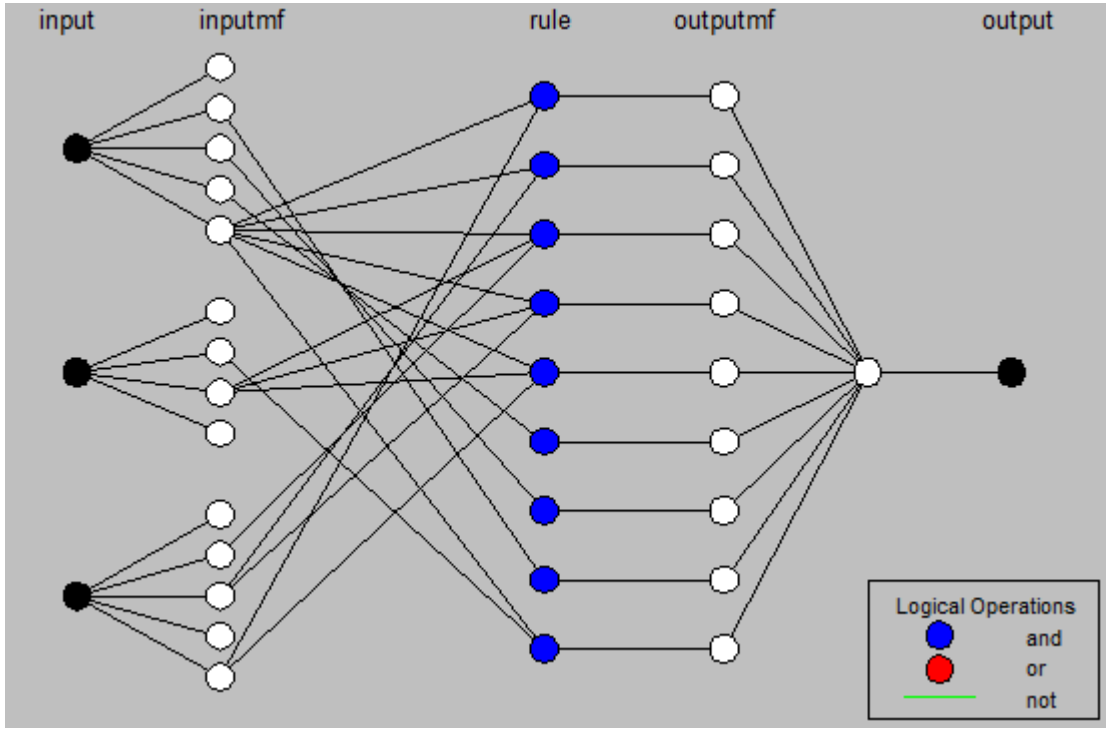
7. Fuzzy Operators: Used to combine fuzzy sets and apply logical operations such as AND, OR, and NOT. These operators are defined based on mathematical functions to manage the degrees of truth.

  - Example: In fuzzy logic, the AND operator might be implemented as the minimum of the membership values, and the OR operator as the maximum.

- if BI-RADS is very large and Shape is very large and Margin is very large then Malign;
- if BI-RADS is very large and Shape is very large and Margin is medium then Malign;
- if BI-RADS is very large and Shape is large and Margin is small then Malign;
- if BI-RADS is very large and Shape is large and Margin is medium then Malign;
- if BI-RADS is very large and Shape is large and Margin is very large then Malign;
- if BI-RADS is very large and Shape is small then Malign;
- if BI-RADS is medium then Benign;
- if BI-RADS is small then Benign;
- if BI-RADS is large then Benign.

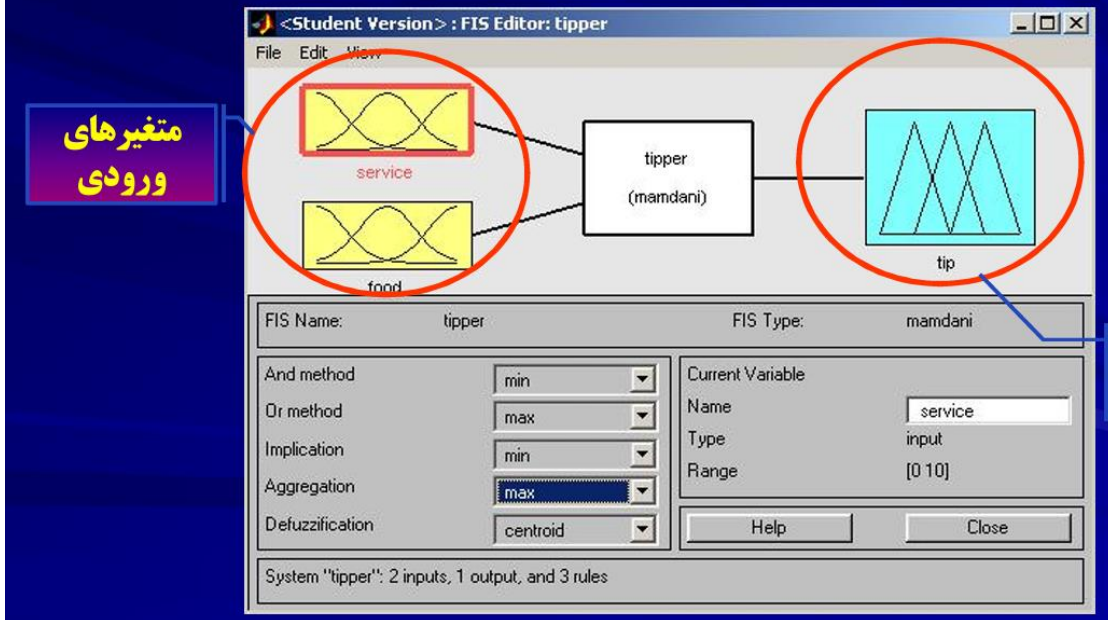
- در این شبکه عصبی فازی ساختار ۳ لایه است . لایه اول ورودی است . این لایه ۳ ویژگی به عنوان ورودی میگیرد که شامل ویژگی , BIRADS , Shape Margin میباشد . سپس برای لایه مخفی ار ۹ نود استفاده میشود که هر کدام یک قانون فازی را شامل میشوند و وزن ها بر اساس میزان عضویت در آن کلاس قانون مشخص میشود و تابع فعال ساز هم t-norm است ۹ نود شامل :
- برچسپ Benign به معنی بدخیم بودن با مقدار ۰ و برچسب Malign به معنی خوش خیم بودن و با مقدار ۱ مشخص شده است.
- مدل استنتاجی در این جا sugeno است و تابع عضویت ورودی ها هم trimf میباشد .
- یک خروجی هم داریم که دارای ۲ مقدار عضویت خوش خیم و بد خیم است که تابع عضویت آن هم trimf است.
- تعداد epoch ها ۱۰ است و روش استفاده شده در شبکه عصبی هم پس انتشار است.
- این تصویر مربوط به آموزش ۷۰۰ نمونه است در ۳۰ دوره آموزش.

Model Structure



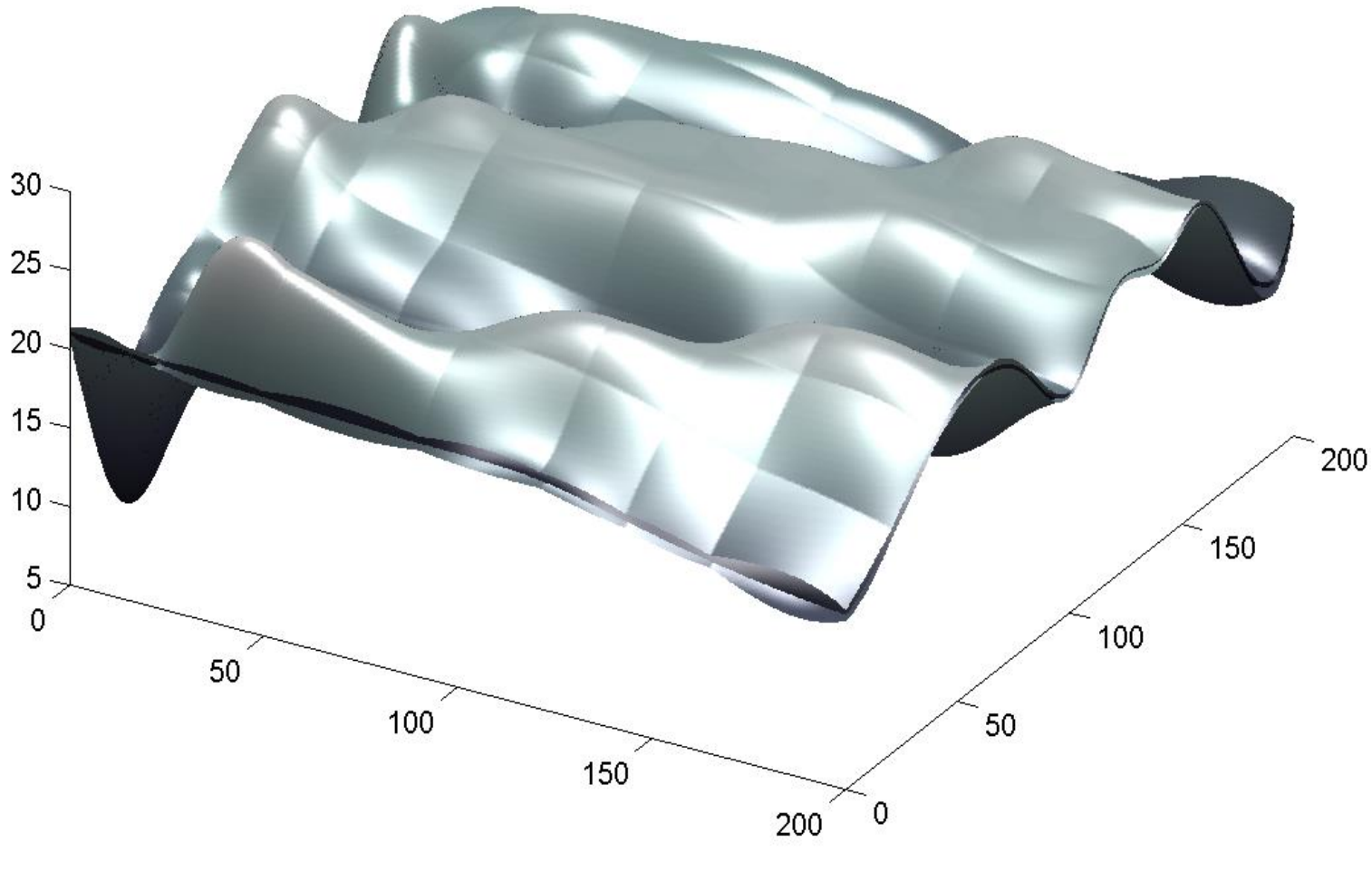
Example: Dinner for Two

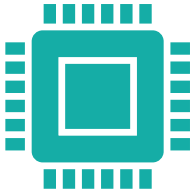
Fuzzy Inference System (FIS) Editor



Mathlab Fuzzy toolbox

Fuzzy Surfaces via Cubic Splines (figure from Jorge dos Santos & Lodwick)





Fuzzy Variables

1. Weather: (bad, ok, perfect)
2. Humidity: (dry, good, wet)
3. Temperature: (cold, good, hot)
4. Precipitation: (no rain, little rain, rain)

Fuzzy Rules

Rule 1: If temperature is hot and precipitation is little rain, then weather is ok.

Rule 2: If temperature is hot and humidity is dry, then weather is ok.

Rule 3: If precipitation is little rain, then weather is ok.

Rule 4: If temperature is hot and humidity is wet, then weather is bad.

Rule 5: If precipitation is rain, then weather is bad.

Rule 6: If temperature is good and humidity is dry, then weather is perfect.

Rule 7: If precipitation is no rain, then weather is perfect.

Rule 8: If temperature is good or humidity is good or precipitation is little rain, then weather is ok.

Rule 9: If temperature is cold, then weather is bad.

Reinforcement Learning

- It is a type of machine learning where an agent learns to make decisions by performing actions in an environment to maximize cumulative rewards. The key components of RL are:

  - Agent: The learner or decision-maker that interacts with the environment to achieve a goal.
  - Environment: The external system with which the agent interacts. It provides feedback to the agent in the form of rewards or penalties based on the agent's actions.
  - State: A representation of the environment at a particular time. The agent perceives the state and decides the next action based on it.
  - Action: A move or decision taken by the agent at any given state to influence the environment.

Reinforcement Learning

- Reward: A scalar feedback signal that tells the agent how good or bad the action taken was. The goal of the agent is to maximize the cumulative reward over time.
- Policy (π): A strategy used by the agent to decide the next action based on the current state. It maps states to actions.
- Value Function: A function that estimates the expected cumulative reward that can be obtained from a state or a state-action pair.
- Q-Value (Q-function): A function that represents the expected reward of taking an action in a specific state and following the policy thereafter.

Example

- Game Playing (e.g., Chess or Go)
  - Scenario: Consider an RL agent learning to play chess.
  - States: Each configuration of the chessboard (the positions of all pieces) is a state.
  - Actions: Possible moves (e.g., moving a pawn, knight, etc.) are the actions.
  - Rewards: Winning the game provides a positive reward, losing gives a negative reward, and each move during the game may have intermediate rewards based on strategy.
  - Goal: The agent's goal is to learn a policy that maximizes its chances of winning the game.

Reinforcement Learning Algorithms

- Q-Learning: A value-based approach where the agent learns the value of actions in each state and selects actions that maximize future rewards.

  - Example Training a robot to navigate a maze.

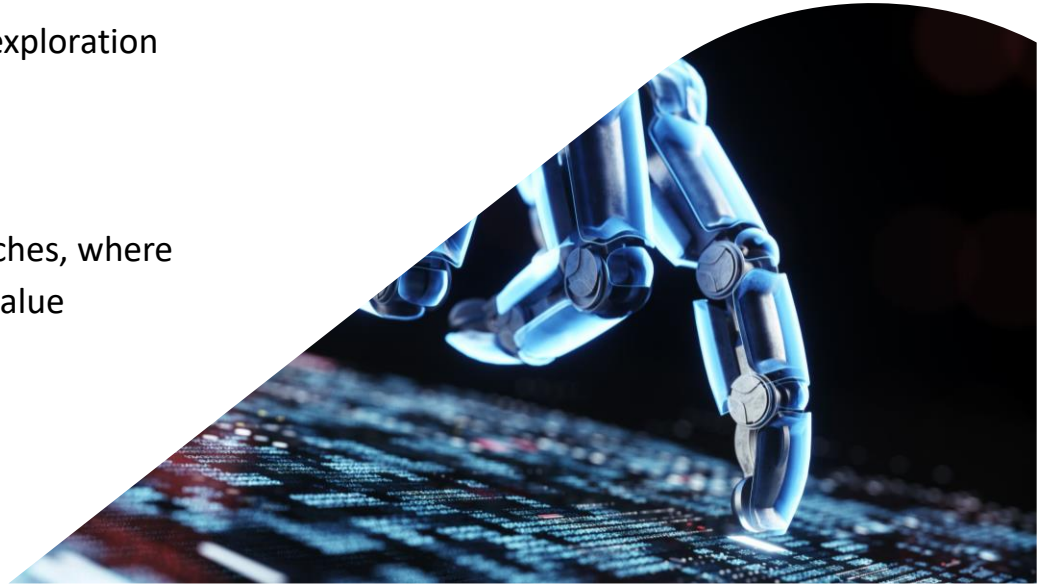
- Deep Q-Networks (DQN): Combines Q-learning with deep learning to handle environments with large state spaces.

  - Example Playing video games like Atari.

- Policy Gradient Methods: Learn a policy directly that maps states to actions, optimizing the expected reward.

  - Example Training robots for continuous control tasks like walking.

- Proximal Policy Optimization (PPO): A popular policy gradient method that balances exploration and exploitation by limiting the policy update step size, making it stable and efficient.
  - Key Concept: Introduces a clipping mechanism to ensure stable policy updates.
- Actor-Critic Methods: These methods combine policy-based and value-based approaches, where the "actor" updates the policy, and the "critic" evaluates the actions by learning the value function.
  - Key Concept: The actor learns to select actions, while the



OpenAI Gym	A popular toolkit for developing and comparing RL algorithms. It provides various environments (e.g., games, simulations) where agents can be trained.
TensorFlow Agents (TF-Agents)	A library for RL in TensorFlow. It provides tools to build, train, and evaluate RL agents.
Stable Baselines3	A set of reliable implementations of RL algorithms in Python, built on top of PyTorch. It includes popular algorithms like PPO, DQN, and A2C.
Ray RLlib	A scalable RL library for training RL agents, supporting a wide variety of RL algorithms and large-scale distributed training.
Keras-RL	A library that integrates with Keras to allow the implementation and experimentation with different RL algorithms, including DQN and Deep Deterministic Policy Gradient (DDPG)
Pytorch RL	A PyTorch-based library providing implementations of various RL algorithms and utilities for building custom agents

Library for Reinforcement Learning

Dimensionality Reduction

Dimensionality reduction techniques are used to reduce the number of input variables in a dataset, simplifying the model and reducing computational cost while preserving essential information.

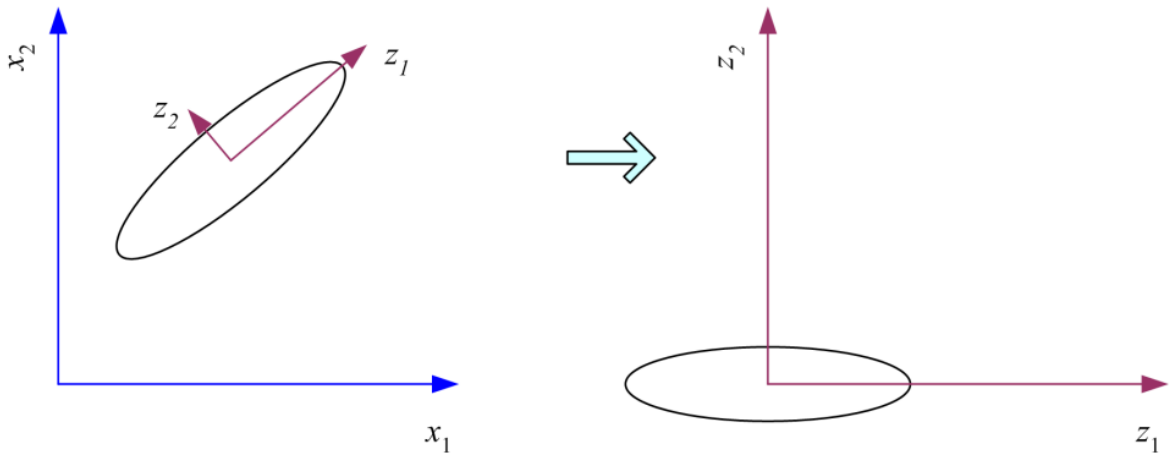
- Principal Component Analysis (PCA): Reduces the dimensionality by projecting the data into a lower-dimensional space.
  - Example: Data visualization and noise reduction.
- Distributed Stochastic Neighbor Embedding (t-SNE): A technique for dimensionality reduction, particularly useful for the visualization of high-dimensional datasets.
  - Example: Visualizing clusters of similar objects in large datasets.

What PCA does

$$z = W^T(x - m)$$

where the columns of **W** are the eigenvectors of ∑, and ***m*** is sample mean

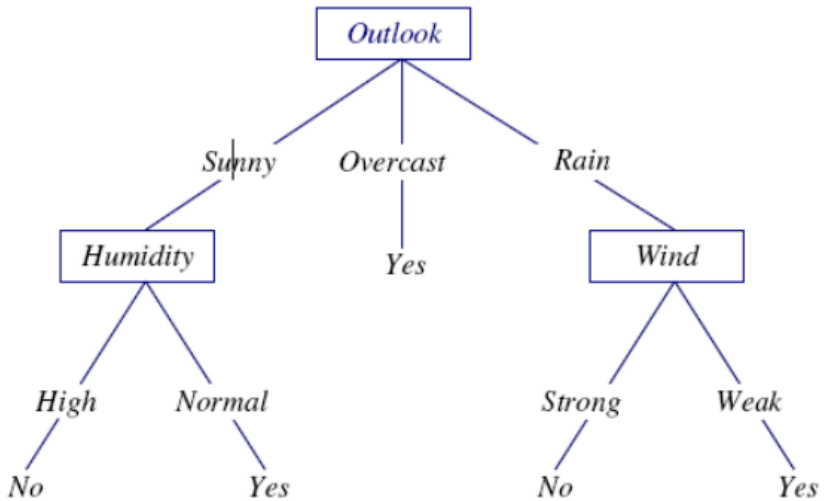
Centers the data at the origin and rotates the axes

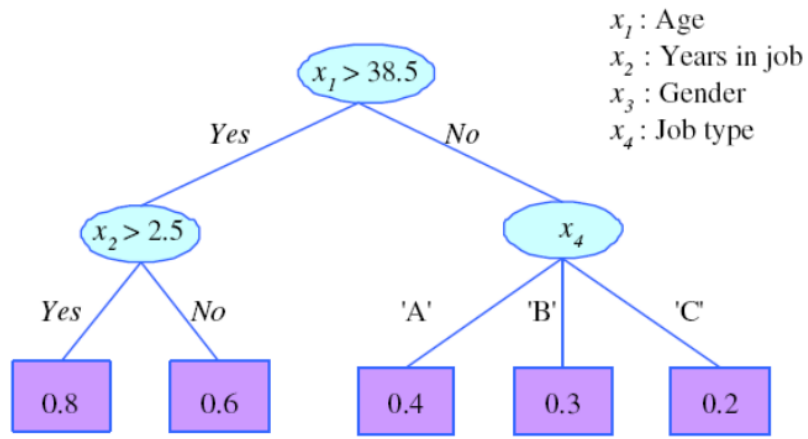


Rule Extraction from Trees

Decision Trees: Operat

C4.5Rules
(Quinlan, 1993)





R1: IF (age>38.5) AND (years-in-job>2.5) THEN y =0.8
R2: IF (age>38.5) AND (years-in-job≤2.5) THEN y =0.6
R3: IF (age≤38.5) AND (job-type='A') THEN y =0.4
R4: IF (age≤38.5) AND (job-type='B') THEN y =0.3
R5: IF (age≤38.5) AND (job-type='C') THEN y =0.2

- Each instance holds attribute values.
- Instances are classified by filtering the attribute values down the decision tree, down to a leaf which gives the final answer.
- Internal nodes: attribute names or attribute values. Branching occurs at attribute nodes.

Ensemble Learning

- Random Forest: An ensemble of decision trees, where each tree is trained on a random subset of the data, and the final prediction is based on the majority vote or average of the trees.

  - Example: Classifying whether a transaction is fraudulent.

- Boosting (e.g., AdaBoost, Gradient Boosting): Combines multiple weak models sequentially, where each new model focuses on correcting the errors made by the previous ones.

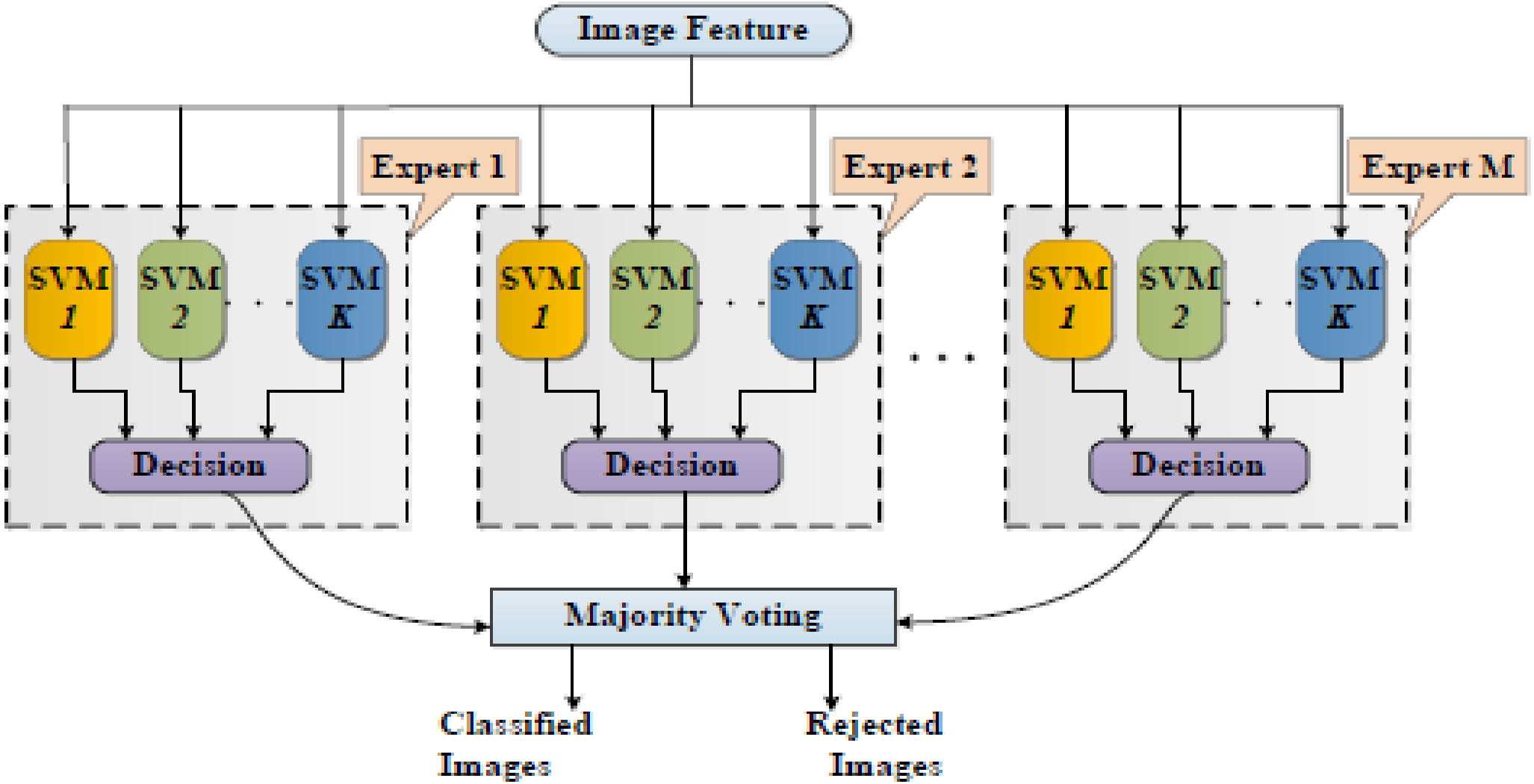
  - Example: Improving accuracy in predicting credit scores.

- Bagging (Bootstrap Aggregating): Reduces variance in the prediction by training the same algorithm multiple times on different subsets of the data.

  - Example: Improving model stability and reducing overfitting.

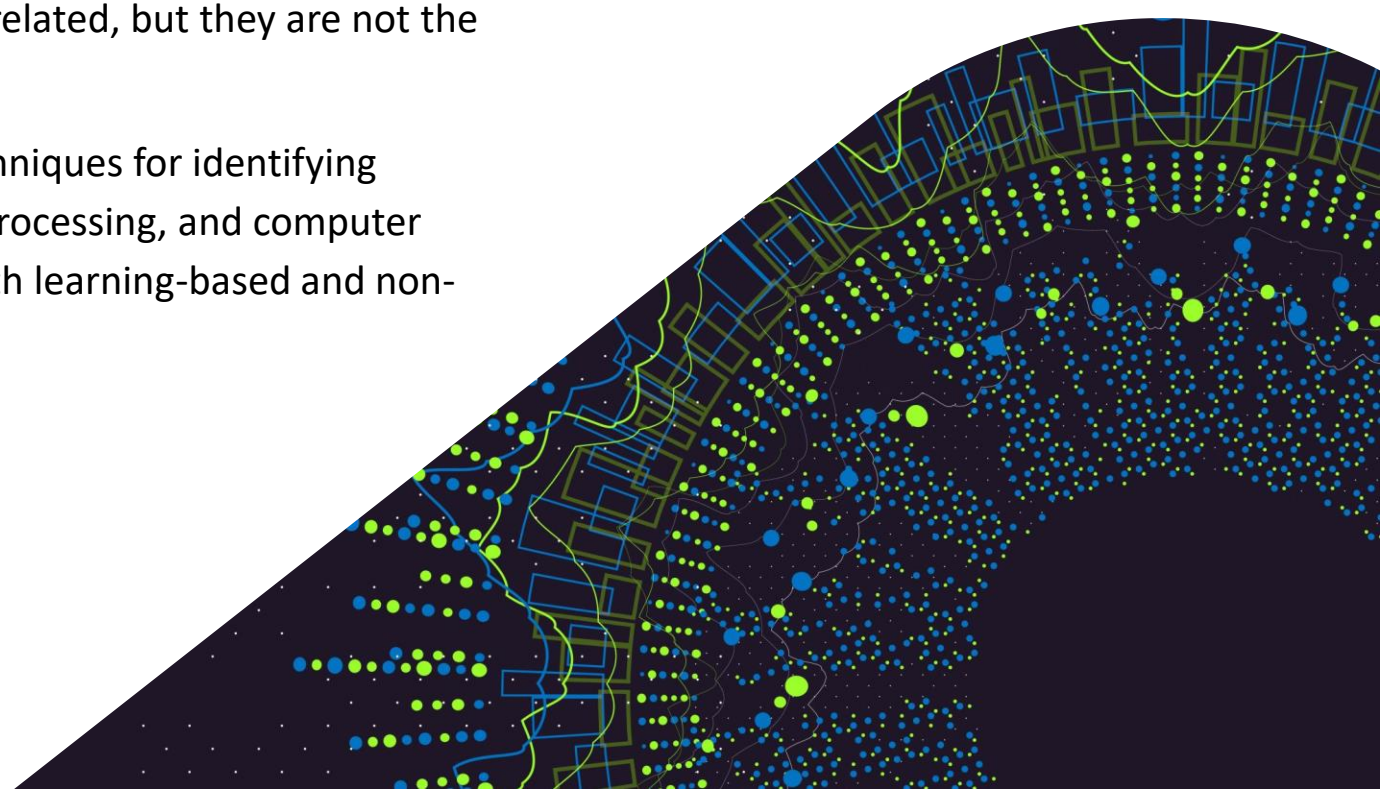
- These types of ML algorithms are used depending on the problem at hand, the nature of the data, and the desired outcome.

Group of SVM, Cascade Classification



Pattern Recognition

- Pattern Recognition is a branch of artificial intelligence that involves the identification and classification of patterns in data. It aims to detect regularities in data and categorize these into different classes.
- The primary goal is to develop systems that can automatically recognize patterns and make decisions based on them.
- Pattern Recognition and Machine Learning (ML) are closely related, but they are not the same thing.
- Pattern Recognition is a broader field that encompasses techniques for identifying patterns in data. It includes methods from statistics, signal processing, and computer vision, in addition to ML. Pattern recognition can involve both learning-based and non-learning-based methods.



Key Concepts in Pattern Recognition

1. Patterns and Features:

Pattern: A pattern is a set of features or attributes that represent an object or phenomenon. For example, a pattern could be a handwritten digit, where the features might include the number of loops, strokes, and overall shape.

Feature Extraction: This involves identifying and extracting the most relevant attributes or characteristics from the data that help in distinguishing different patterns. For example, in facial recognition, features might include the distance between the eyes, the shape of the nose, or the contours of the face.

2. Classification:

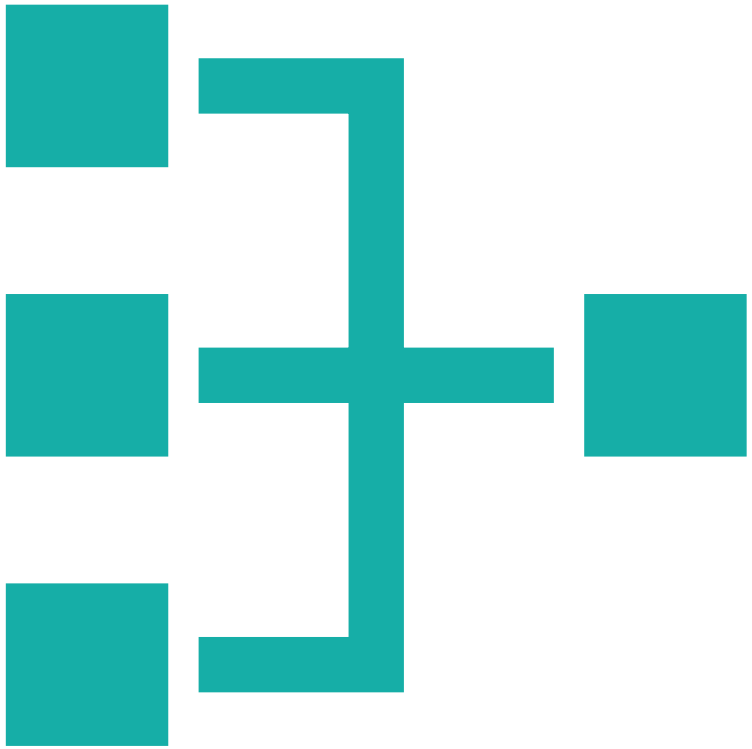
Supervised Learning: The system is trained on a labeled dataset where the correct output (class) is provided for each input pattern. The goal is to learn a mapping from inputs to outputs that can be generalized to new, unseen data.

Example: In a handwritten digit recognition system, the model is trained on labeled images of digits (0-9), where each image has a corresponding label indicating the digit.

Unsupervised Learning: The system tries to find patterns and group them into clusters without labeled data. The goal is to discover the inherent structure of the data.

Example: Clustering customers into different groups based on purchasing behavior without pre-defined categories.

Key Concepts in Pattern Recognition



3. Decision Boundaries:

   Linear vs. Non-Linear Boundaries: The system learns decision boundaries that separate different classes of patterns. A linear boundary might be sufficient for simple problems, while complex patterns might require non-linear boundaries.

   Example: In a 2D space where each point represents a pattern (e.g., a type of fruit), the decision boundary could be a line that separates apples from oranges.

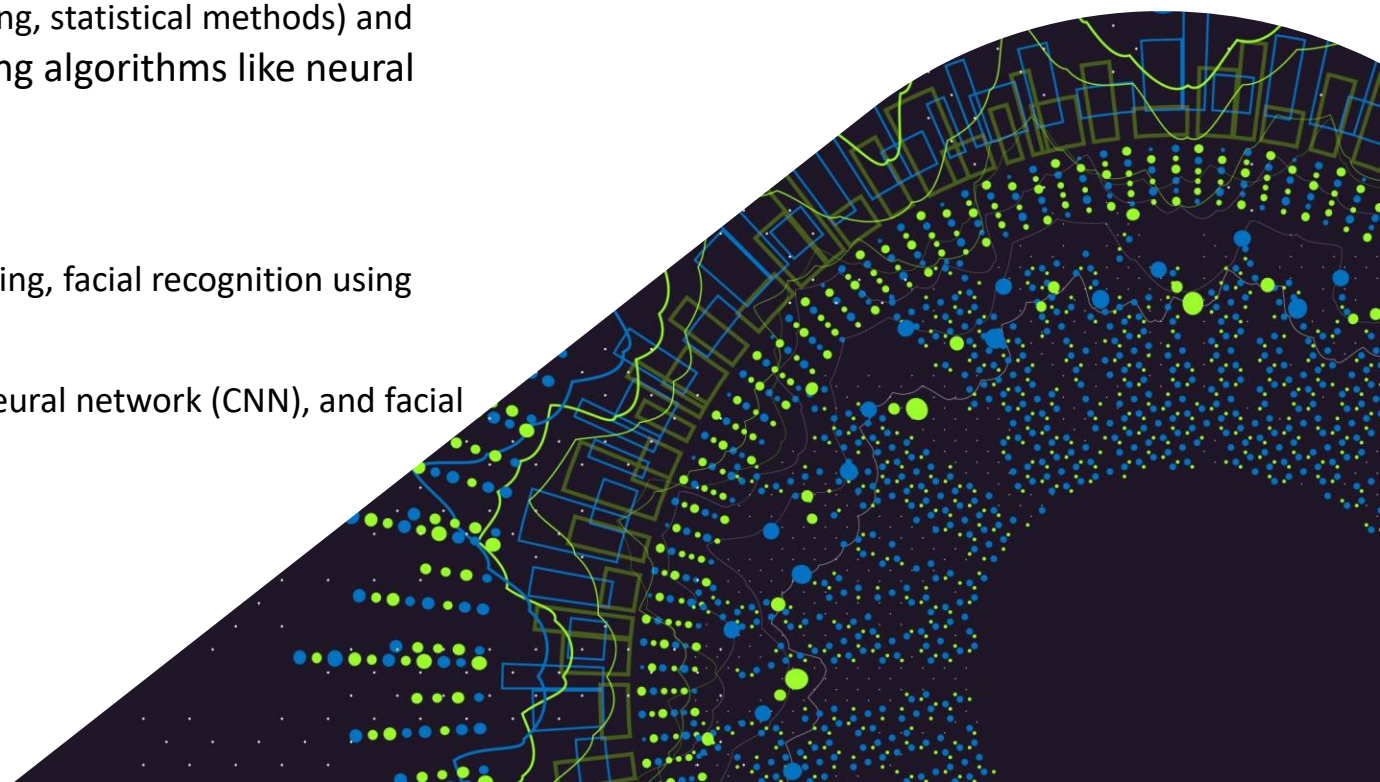
4. Distance Metrics:

   Pattern recognition often involves measuring the similarity or dissimilarity between patterns using distance metrics like Euclidean distance, Manhattan distance, or Mahalanobis distance.

   Example: In facial recognition, the system might measure the distance between two feature vectors (representing two faces) to determine if they belong to the same person.

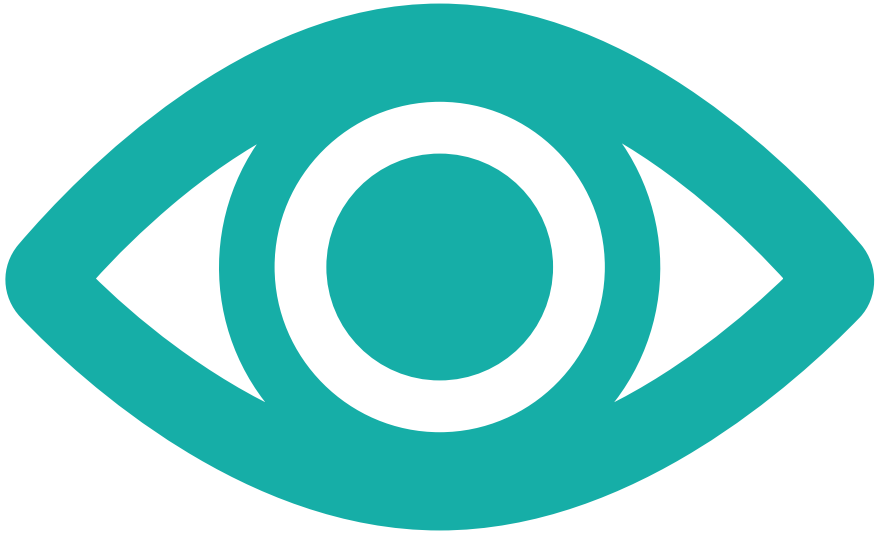
Pattern Recognition vs Machine Learning

- Scope
  - Focuses on the identification and classification of patterns, often using methods that do not involve learning vs Primarily concerned with developing models that learn from data to make predictions or decisions.
  - Approach
  - Can use both traditional rule-based methods (e.g., template matching, statistical methods) and machine learning methods, vs Relies on learning from data, using algorithms like neural networks, decision trees, and support vector machines.
- Examples:
  - Pattern Recognition: Handwriting recognition using template matching, facial recognition using geometric feature analysis.
  - Machine Learning: Handwriting recognition using a convolutional neural network (CNN), and facial recognition using a deep learning model.



Machine Vision

- Machine Vision is a field of computer science and engineering focused on enabling machines to interpret and make decisions based on environmental visual input. It involves using cameras, sensors, and software to capture, process, and analyze images or videos for various applications, such as quality inspection in manufacturing, autonomous vehicles, and robotics.



Fundamental Concept of Machine Vision

1. Image Acquisition:

- Cameras and Sensors: Machine vision systems use cameras and optical sensors to capture images or video from the real world. These images are the raw data that will be processed and analyzed by the system.

- Lighting: Proper lighting is critical for capturing high-quality images. Different lighting techniques (e.g., backlighting, structured lighting) are used to highlight features of interest and minimize shadows or reflections.

2. Image Processing:

- Preprocessing: Before analysis, images are often preprocessed to enhance quality, remove noise, and improve contrast. Techniques like filtering, thresholding, and edge detection are commonly used.

- Feature Extraction: This step involves identifying key features in the image that are relevant to the task, such as edges, corners, textures, or specific shapes.

3. Object Detection and Recognition:

- Object Detection: Identifying and locating objects within an image. This could involve simple tasks like finding a part on an assembly line or more complex tasks like detecting pedestrians in an autonomous vehicle's camera feed.

- Object Recognition: Once an object is detected, the system must recognize and classify it. This often involves pattern recognition and machine learning algorithms to match detected objects with known categories.

Fundamental Concept of Machine Vision

4. Measurement and Analysis:

- Dimensional Measurement: Machine vision systems can measure physical dimensions, such as length, width, height, and angles, with high precision. This is commonly used in quality control and inspection.

- Defect Detection: Identifying defects or anomalies in products, such as scratches, dents, or missing components, is a crucial application in manufacturing.

5. Decision Making:

- Automation and Control: Machine vision systems are often integrated with automation systems to make decisions based on visual input. For example, if a defect is detected, the system might reject the product or alert an operator.

6. System Integration:

- Integration with Machinery: Machine vision systems are often part of larger automated systems. They need to be integrated with other components like robotic arms, conveyors, and sorting mechanisms.

Image Processing

- Image processing involves the manipulation and analysis of digital images using various algorithms and techniques. It's a key area in computer vision and pattern recognition, with applications ranging from medical imaging to facial recognition

1. Pixels and Image Representation

- Pixels: The basic unit of a digital image. Each pixel represents a single point in the image and has a value that indicates its color or intensity.

- Grayscale Images: Each pixel is represented by a single value, typically ranging from 0 (black) to 255 (white).

- Color Images: Represented by three channels (Red, Green, Blue), where each channel is a grayscale image corresponding to one of the RGB colors.

2. Image Histograms

- Histogram: A graphical representation of the distribution of pixel intensities in an image. Histograms help in analyzing contrast, brightness, and intensity distribution.

- Application: Useful in tasks like contrast adjustment, thresholding, and image equalization.

3. Filtering and Convolution

- Convolution: A mathematical operation used to apply filters (kernels) to an image. This is a fundamental operation in image processing, especially in edge detection, blurring, and sharpening.

- Kernels/Filters: Small matrices applied to the image to perform various effects. Examples include:

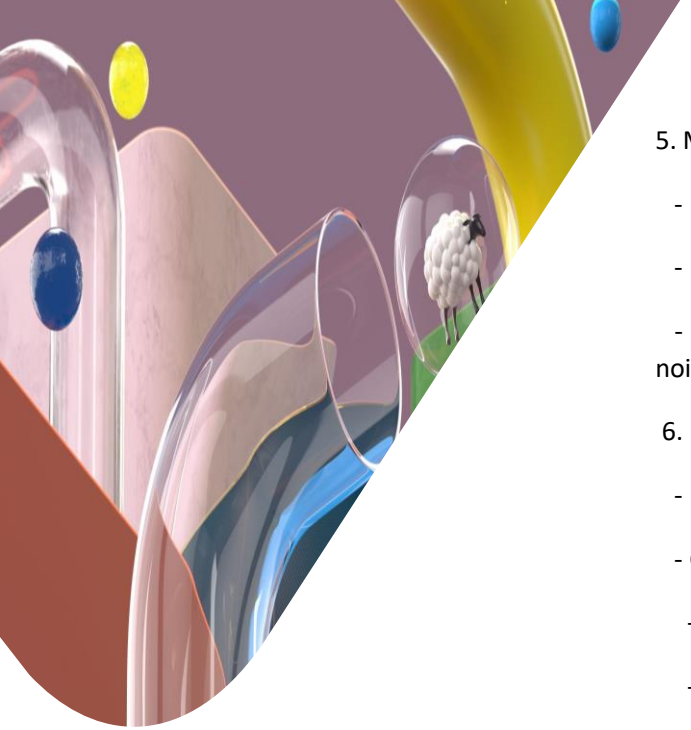
  - Gaussian Blur: Smooths the image by averaging the pixels with their neighbors.

  - Sobel Filter: Detects edges by highlighting regions of high-intensity change.

4. Thresholding

- Thresholding: A technique to convert a grayscale image into a binary image (black and white) by assigning pixel values as 0 or 255 based on a threshold value.

- Application: Common in object detection and image segmentation, where you need to distinguish objects from the background.



5. Morphological Operations

- Erosion: Removes pixels on object boundaries, making objects in the image smaller. It is used to eliminate small noise.

- Dilation: Adds pixels to the boundaries of objects, making them larger. It is used to fill small holes or gaps in objects.

- Opening and Closing: Combination of erosion followed by dilation (opening) or dilation followed by erosion (closing). These operations are used to remove noise and smooth objects.

6. Edge Detection

- Edge Detection: Identifies points in an image where the intensity changes sharply, which usually corresponds to object boundaries.

- Common Methods:

  - Canny Edge Detector: A multi-stage algorithm that detects a wide range of edges in images.

  - Sobel and Prewitt: Gradient-based methods that detect edges by computing the gradient of the image intensity.

7. Image Transformations

- Scaling: Resizing an image by increasing or decreasing its dimensions.

- Rotation: Rotating an image by a certain angle around a central point.

- Translation: Shifting an image along the x or y-axis.

- Affine Transformation: A combination of translation, rotation, scaling, and shearing.

8. Image Segmentation

- Segmentation: The process of partitioning an image into meaningful regions, typically to isolate objects of interest.

- Techniques:

  - Thresholding-based Segmentation: Using intensity thresholds to create segments.

  - Edge-based Segmentation: Detecting edges to define object boundaries.

  - Region-based Segmentation: Growing regions from seed points by adding neighboring pixels with similar properties.

Python vs Java

- Apache Spark Mllib vs Pyspark Python
- Weka Java vs Scikit Python
- DeepLearning4j vs Tensorflow Python
- Tensorflow Java API vs Tensorflow Python MLP

- https://flatlogic.com

Classifier Accuracy Measures

	C_1	C_2
C_1	True positive	False negative
C_2	False positive	True negative

classes	buy_computer = yes	buy_computer = no	total	recognition(%)
buy_computer = yes	6954	46	7000	99.34
buy_computer = no	412	2588	3000	86.27
total	7366	2634	10000	95.52

- Accuracy of a classifier M, acc(M): percentage of test set tuples that are correctly classified by the model M
  - Error rate (misclassification rate) of M = 1 – acc(M)
  - Given *m* classes, $CM_{i,j}$, an entry in a **confusion matrix**, indicates # of tuples in class *i* that are labeled by the classifier as class *j*
- Alternative accuracy measures (e.g., for cancer diagnosis)

  sensitivity = t-pos/pos /* true positive recognition rate */

  specificity = t-neg/neg /* true negative recognition rate */

  precision = t-pos/(t-pos + f-pos)

  accuracy = sensitivity * pos/(pos + neg) + specificity * neg/(pos + neg)
  - This model can also be used for cost-benefit analysis

Classifier Accuracy Measures (Information Retrieval)

- Precision :
  - Precision is the fraction of retrieved instances that are relevant

$$\text{precision} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{retrieved documents}\}|}$$

- Recall:
  - Recall is the fraction of relevant instances that are retrieved

$$\text{recall} = \frac{|\{\text{relevant documents}\} \cap \{\text{retrieved documents}\}|}{|\{\text{relevant documents}\}|}$$

- F-Measure:
  - A measure that combines precision and recall is the harmonic mean of precision and recall

$$F = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

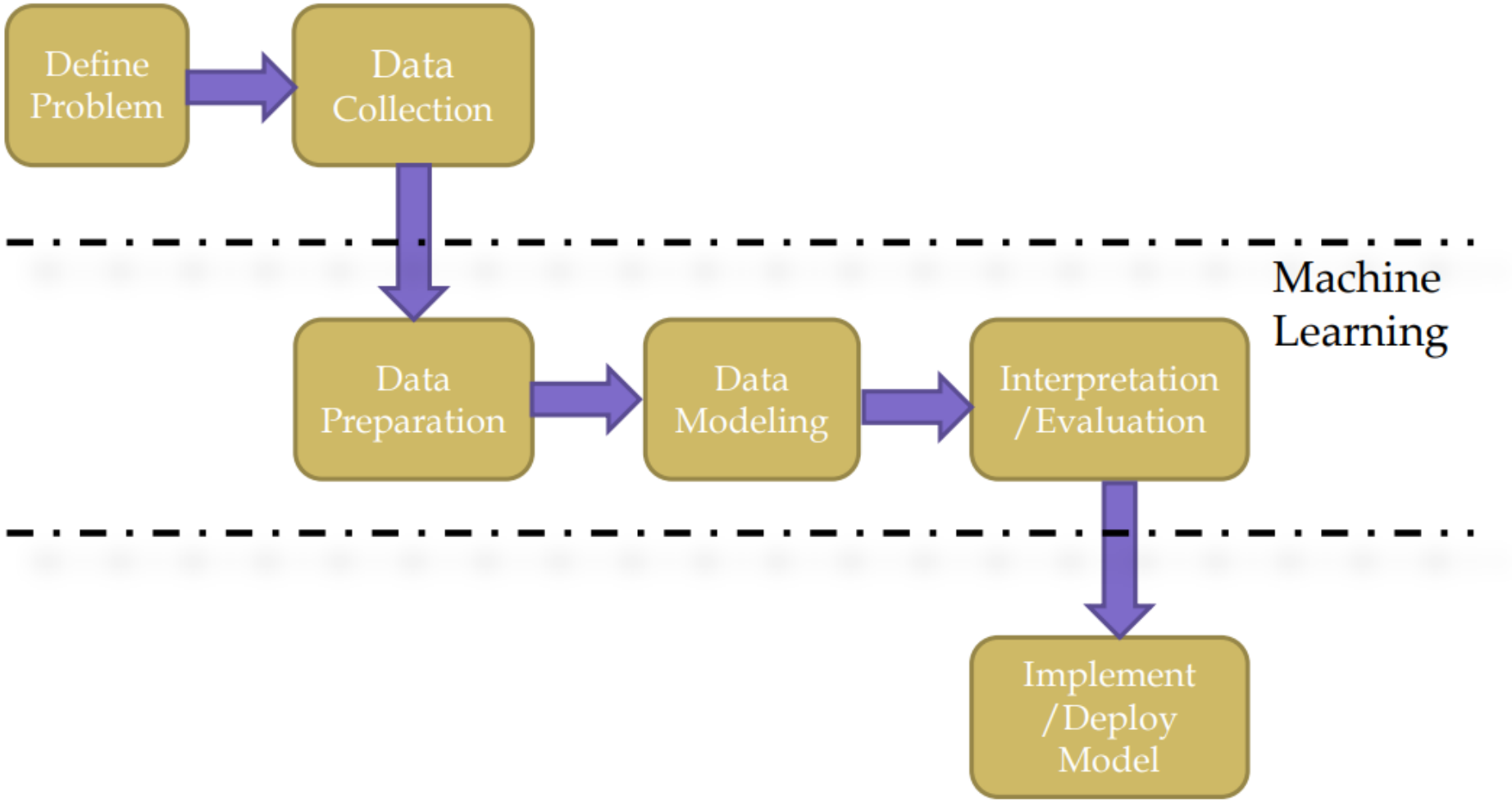
Predictor Error Measures

- Measure predictor accuracy: measure how far off the predicted value is from the actual known value
- **Loss function**: measures the error betw. y_i and the predicted value y_i'
  - Absolute error: $| y_i - y_i' |$
  - Squared error: $(y_i - y_i')^2$
- Test error (generalization error): the average loss over the test set
  - Mean absolute error: $\frac{\sum_{i=1}^{d} | y_i - y_i' |}{d}$ Mean squared error: $\frac{\sum_{i=1}^{d} (y_i - y_i')^2}{d}$
  - Relative absolute error: $\frac{\sum_{i=1}^{d} | y_i - y_i' |}{\sum_{i=1}^{d} | y_i - \bar{y} |}$ Relative squared error: $\frac{\sum_{i=1}^{d} (y_i - y_i')^2}{\sum_{i=1}^{d} (y_i - \bar{y})^2}$

  The mean squared-error exaggerates the presence of outliers

  Popularly use (square) root mean-square error, similarly, root relative squared error

Data Mining


Define Problem
Data Collection
Data Preparation
Data Modeling
Interpretation /Evaluation
Implement /Deploy Model
Machine Learning

Cross-Validation

- To estimate generalization error, we need data unseen during training. We split the data as
  - Training set (50%)
  - Validation set (25%)
  - Test (publication) set (25%)
- Resampling when there is few data

معیار های بررسی روش

- نرخ تشخیص (Recognition rate)= تعداد تصاویری که به درستی تشخیص داده شده است/(تقسیم بر) تعداد تصاویر بررسی شده
- نرخ رد کردن (Rejection rate)= تعداد تصاویر رد شده/ تعداد تصاویر بررسی شده
- قابلیت اطمینان (Reliability)= (تعداد تصاویری که به درستی تشخیص داده شده است+تعداد تصاویر رد شده) / تعداد تصاویر بررسی شده
- نرخ خطا (Error rate)=100% - قابلیت اطمینان
- قابلیت اطمینان= نرخ تشخیص+ نرخ رد کردن