Difference Between AI, ML, DL, and Data Science

1. Artificial Intelligence (AI):

- **Definition**: AI is the broad field of creating machines that can mimic human intelligence.
- **Scope**: Encompasses reasoning, learning, decision-making, and natural language understanding.
- **Examples**: Chatbots, recommendation systems, autonomous vehicles.
- **Key Techniques**: Rule-based systems, expert systems, and machine learning.

---

2. Machine Learning (ML):

- **Definition**: A subset of AI focused on algorithms that allow machines to learn from data without being explicitly programmed.
- **Scope**: Focuses on prediction and pattern recognition using historical data.
- **Examples**: Fraud detection, price prediction, spam filtering.
- **Key Techniques**: Supervised, unsupervised, and reinforcement learning.

---

3. Deep Learning (DL):

- **Definition**: A subset of ML that uses artificial neural networks with multiple layers to learn complex patterns.
- **Scope**: Specialized for processing large-scale data like images, audio, and text.
- **Examples**: Image recognition, natural language processing, speech synthesis.
- **Key Techniques**: Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), Transformers.

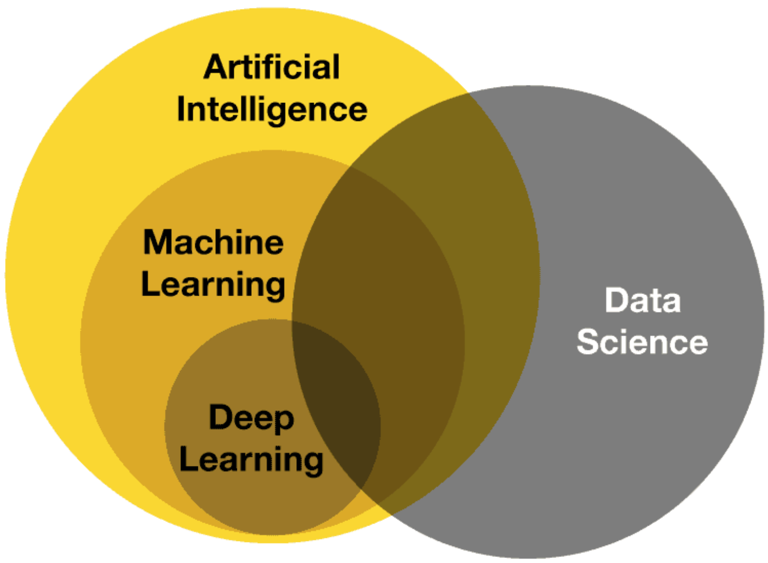
---

4. Data Science:

- **Definition**: An interdisciplinary field that combines statistics, programming, and domain expertise to extract meaningful insights from data.
- **Scope**: Focuses on data analysis, visualization, and building data-driven solutions.
- **Examples**: Business analytics, customer segmentation, forecasting.
- **Key Techniques**: Data wrangling, statistical modeling, machine learning, and visualization.

Key Differences

Feature	AI	ML	DL	Data Science
Definition	Broad intelligence	Learning from data	Neural network-based ML	Insights from data
Scope	Wide	Prediction and learning	Advanced ML with deep nets	Data preparation & analysis
Techniques Used	ML, rule-based	Algorithms like SVM	CNN, RNN, Transformers	Statistics, ML, visualization
Example	Chatbots	Spam filter	Face recognition	Customer behavior analytics



Machine Learning Techniques

1. Supervised Learning

- **Definition**: The model is trained on labeled data where the output is known.
- **Goal**: Predict outcomes for new data.
- **Techniques**:
  Regression:
  - **Description**: Predicts continuous values.
  - **Examples**:
    - **Predicting House Prices**:

            - **Data**: Features like number of bedrooms, location, house size, proximity to schools.
            - **Goal**: Predict house prices based on these features.
            - **Process**: Train a Linear Regression model with historical data to learn the relationship between features and price.
        - **Forecasting Sales Revenue**:
            - **Data**: Historical sales data (time of year, advertising spend, sales team size).
            - **Goal**: Predict future revenue based on seasonal trends and marketing efforts.
            - **Process**: Use Polynomial Regression to capture patterns in sales data.
- **Classification:**
    - **Description**: Predicts discrete categories.
    - **Examples**:
        - **Email Spam Detection**:
            - **Data**: Emails labeled as "Spam" or "Not Spam," with features like word frequency and sender reputation.
            - **Goal**: Classify new emails as spam or not spam.
            - **Process**: Use Logistic Regression or Naive Bayes to classify emails.
        - **Tumor Classification**:
            - **Data**: Patient data (e.g., tumor size, shape).
            - **Goal**: Identify if a tumor is benign or malignant.
            - **Process**: Train a Decision Tree using labeled medical data.

---

2. Unsupervised Learning

- **Definition**: The model is trained on data without labeled outcomes.
- **Goal**: Discover hidden patterns or structures in the data.
- **Techniques**:
  Clustering:
    - **Description**: Groups similar data points.
    - **Examples**:
        - **Customer Segmentation**:
            - **Data**: Customer data (purchase history, income, age).
            - **Goal**: Group customers with similar behaviors for marketing.
            - **Process**: Apply K-Means clustering to find patterns in customer behavior.
        - **Document Categorization**:
            - **Data**: Text data from various documents.

            - **Goal**: Cluster similar documents for easier retrieval.
            - **Process**: Use Hierarchical Clustering to organize documents into themes.
- **Dimensionality Reduction:**
    - **Description**: Reduces the number of features while retaining essential information.
    - **Examples**:
        - **Visualizing High-Dimensional Data**:
            - **Data**: Datasets with many features (e.g., genome data).
            - **Goal**: Simplify the data for visualization.
            - **Process**: Apply PCA or t-SNE to reduce dimensions and plot the data.
        - **Speeding Up Computations**:
            - **Data**: Sensor readings from IoT devices.
            - **Goal**: Retain important features for faster processing.
            - **Process**: Use Autoencoders for feature compression.

---

3. Semi-Supervised Learning (Not main category)

- **Definition**: A mix of labeled and unlabeled data is used for training.
- **Goal**: Improve learning accuracy with minimal labeled data.
- **Examples**:
    - **Medical Diagnosis**:
        - **Data**: A small number of labeled cases (disease/healthy) and many unlabeled records.
        - **Goal**: Classify patient records effectively.
        - **Process**: Use Generative Models to label and learn from unlabeled cases.
    - **Webpage Categorization**:
        - **Data**: A few labeled web pages and many unlabeled ones.
        - **Goal**: Automatically assign categories to unlabeled pages.
        - **Process**: Use Self-Training to iteratively label and improve the model.

---

4. Reinforcement Learning

- **Definition**: The model learns by interacting with an environment and receiving rewards or penalties.
- **Goal**: Learn a sequence of actions to maximize cumulative reward.
- **Examples**:
    - **AlphaGo**:
        - **Data**: The board state (positions of pieces).

        - **Goal**: Maximize the chance of winning.
        - **Process**: The algorithm plays games to learn which moves yield the highest rewards.
    - **Robotics**:
        - **Data**: Sensor readings (position, obstacle distance).
        - **Goal**: Navigate a space without collisions.
        - **Process**: The robot learns optimal movements through rewards and penalties.

---

5. Ensemble Learning (Not main category)

- **Definition**: Combines multiple models to improve performance.
- **Goal**: Enhance prediction accuracy and reduce overfitting.
- **Examples**:
    - **Fraud Detection**:
        - **Data**: Transaction history (amount, merchant type, location).
        - **Goal**: Detect fraudulent transactions.
        - **Process**: Combine predictions from Random Forest and Gradient Boosting models.
    - **Predicting Stock Prices**:
        - **Data**: Historical stock data (prices, volume).
        - **Goal**: Predict future stock prices.
        - **Process**: Use Stacking to combine predictions from different regression models.

---

6. Neural Networks (Deep Learning) (Not main category)

- **Definition**: A subset of ML focused on learning from large datasets using neural networks.
- **Goal**: Handle complex tasks like image and speech recognition.
- **Examples**:
    - **Object Detection**:
        - **Data**: Images with bounding boxes around objects.
        - **Goal**: Identify and locate objects in images.
        - **Process**: Train CNN-based models like YOLO to learn object features.
    - **Machine Translation**:
        - **Data**: Parallel text (e.g., English sentences and French translations).
        - **Goal**: Translate text from one language to another.
        - **Process**: Use Transformer models like BERT for context-aware translation.

---

7. Transfer Learning (Not main category)

- **Definition**: Using a pre-trained model on a related task for a new task.
- **Goal**: Reduce training time and improve performance with limited data.
- **Examples**:
  - **Medical Imaging**:
    - **Data**: Small dataset of labeled X-rays.
    - **Goal**: Classify X-rays into disease categories.
    - **Process**: Fine-tune a pre-trained CNN like ResNet for medical imaging.
  - **Sentiment Analysis**:
    - **Data**: Reviews labeled as positive or negative.
    - **Goal**: Classify the sentiment of new reviews.
    - **Process**: Fine-tune a pre-trained BERT model on the dataset.

Lines in 2D Geometry

- **Definition**: A line in 2D space is the locus of all points satisfying a linear equation.
  Equation of a Line:
  - **General Form**: ax+by+c=0
  - **Slope-Intercept Form**: y=mx+c
    - m: Slope of the line.
    - c: Y-intercept (value of y when x=0).

Key Concepts:

- **Slope**:
  - Describes the steepness of the line.
  - $m = \frac{\text{rise}}{\text{run}} = \frac{y_2 - y_1}{x_2 - x_1}$.
- **Intercepts**:
  - **X-intercept**: Point where the line crosses the x-axis (y=0).
  - **Y-intercept**: Point where the line crosses the y-axis (x=0).

Let's look at the general form of the equation of the straight line.

Equation of the form ax + by + c = 0, will represent a straight line. Here a, b, c are arbitrary constants (a and b cannot be both 0), and x and y are variables (which represent the coordinates of points on the line).

ax + by + c = 0

Or

y = (-a/b)x + (-c/b)

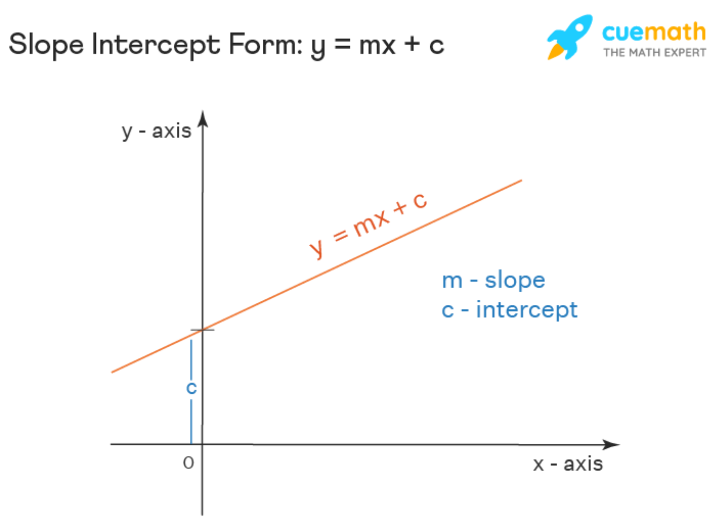
By putting -a/b = m and -c/b = c, the above equation becomes

y = mx + c

Therefore, the equation ax + by + c = 0 represents a line with slope −a/b and y-intercept −c/b.

Example:

- Line equation: y=2x+3
  - Slope (m): 2 (line rises 2 units for every 1 unit run).
  - Y-intercept (c): 3 (line crosses y-axis at 3).



Applications in ML:

- Linear regression in 2D finds the best-fit line: y=w1x+b
- w1: Weight (slope).
- b: Bias (intercept).

Lines in 3D Geometry

- **Definition**: A line in 3D is represented by a vector equation, describing all points along the line.

Vector Form:

$$\vec{r} = \vec{a} + t\vec{b}$$

- $\vec{r}$: Position vector of a point on the line.
- $\vec{a}$: Position vector of a specific point the line passes through.
- $\vec{b}$: Direction vector of the line.
- t: Scalar parameter.

Parametric Form:

$$x = x_1 + tb_x, \quad y = y_1 + tb_y, \quad z = z_1 + tb_z$$

- (x_1, y_1, z_1): Coordinates of a point on the line.
- (b_x, b_y, b_z): Direction components.

Example:

- Given point $(1, 2, 3)$ and direction vector $(2, -1, 4)$:
  - Parametric equations: $x = 1 + 2t,\ y = 2 - t,\ z = 3 + 4t$.

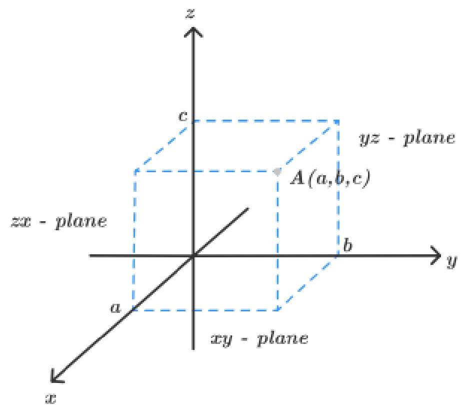
Equation of a Plane:

$$ax + by + cz = d$$

- (a, b, c): Coefficients forming the **normal vector** $\vec{n} = (a, b, c)$, perpendicular to the plane.
- d: Distance from the origin along the normal vector.

Plane Defined by Points:

- Three non-collinear points P_1, P_2, P_3 determine a unique plane.
- Compute two vectors on the plane:
$$\vec{v_1} = P_2 - P_1, \quad \vec{v_2} = P_3 - P_1$$
- Normal vector:
$$\vec{n} = \vec{v_1} \times \vec{v_2}$$



Hyperplanes

- **Definition**: A hyperplane is a subspace of one dimension less than its ambient space. In 2D, it's a line; in 3D, it's a plane.

Equation of a Hyperplane:

$$w_1x_1 + w_2x_2 + \cdots + w_nx_n = b$$

- $\vec{w} = (w_1, w_2, \ldots, w_n)$: Normal vector.
- b: Bias or offset.

Key Properties:

- **Normal Vector**: Perpendicular to the hyperplane, indicating its orientation.
- **Separability**: Hyperplanes are used in machine learning to separate data into classes (e.g., in Support Vector Machines).

Dot Product and Hyperplanes

- The equation of a **hyperplane** can be written using the **dot product**:

$$\vec{w} \cdot \vec{x} = b$$

- $\vec{w}$: Normal vector of the hyperplane.
- $\vec{x}$: Position vector of a point on the hyperplane.
- b: Offset from the origin.

Applications in Machine Learning:

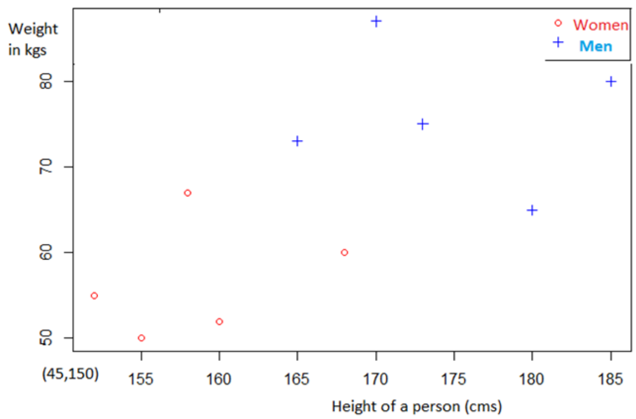
1. **Linear Classification**:

   - The decision boundary in a linear classifier (e.g., Logistic Regression, SVM) is a hyperplane:
$$\text{Class 1 if } \vec{w} \cdot \vec{x} + b > 0, \text{ Class 2 if } \vec{w} \cdot \vec{x} + b < 0.$$

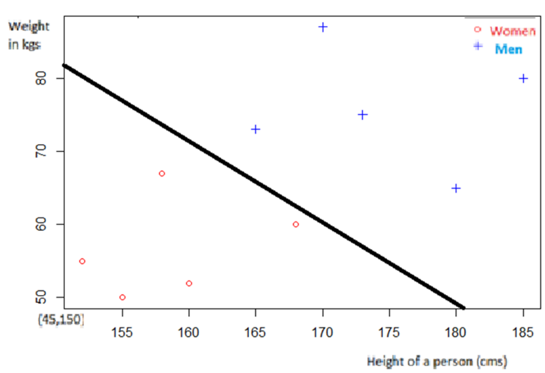
2. **Feature Importance**:

   - The normal vector $\vec{w}$ indicates the importance of features in separating data.

Example:
We will start with an example. We have plotted the height and weight of several people, and we want to distinguish between men and women from this height and weight data points. For instance: if someone measures height as 175 cm and weighs as 80 kg, is it a man or a woman?



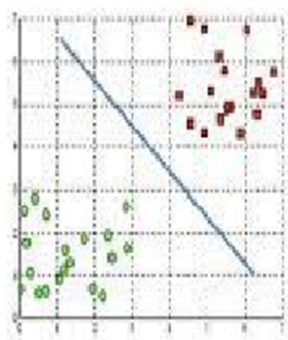
From the above plot, we can see that it is possible to separate the above data points because they are linearly separable data points i.e. a straight line can be drawn to separate all the tuples of class +1 from all the tuples of class −1. For instance, we could draw a line and then all the data points representing men will be above the line, and all the data points representing women will be below the line. Such a line is called a hyperplane and is depicted below:



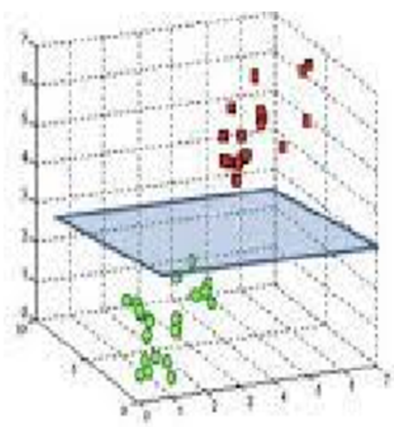
Thus data points falling on either side of the hyperplane can be attributed to different classes. So when we get a new point like height as 175 cm and weights as 80 kg then we can classify it either men or women after plotting and finding its location with reference to the hyperplane drawn above. In Machine Learning, a hyperplane is a decision boundary that divides the input space into two or more regions, each corresponding to a different class or output label. In a 2D space, a hyperplane is a straight line that divides the space into two halves. In a 3D space, however, a hyperplane is a plane that divides the space into two halves.

Note that the dimension of the hyperplane depends upon the number of features. If the number of input features is 2, then the hyperplane is just a line. If the number of input features is 3, then the hyperplane becomes a two-dimensional plane. It becomes difficult to imagine when the number of features exceeds 3. The dimension of the hyperplane is directly proportional to the number of features. As features get increased then the numbers of dimension for ML model also increases and in higher dimension visualization is almost impossible so we need hyperplanes which can be easily extended to n dimensions.

Hyperplane in 2D



Hyperplane in 3D



In coordinate geometry we cannot have more than 3 axes namely x, y and z. So to designate axes in higher dimensions in machine learning, we will use x1, x2, x3,…xn as axes labels and correspondingly w1, w2, w3….wn as coefficients of x1, x2, x3… etc. instead of a, b, …etc.

Thus ax + by + c = 0 will be replaced by

$$w_1x_1+w_2x_2+w_0=0.... (1)$$

Extending the same concept to 4D, we get equation of hyperplane as

$$w_1x_1+w_2x_2+ w_3x_3+ w_4x_4+w_0=0.... (3)$$

Thus in n dimension equation of hyperplane denoted as **'π'** is

$$w_1x_1+w_2x_2+ w_3x_3+ w_4x_4+.....+w_nx_n+w_0=0.... (4)$$

Now we will write the equation (4) in vector notation.

What is a vector? In simple terms we can define a vector as a tuple of one or more values called scalars. Vectors are built from ordinary numbers. You can think of a vector as a list of numbers, and vector algebra operations are performed on the numbers in the list. It is most commonly used in machine learning to represent the data in the most optimized and organized way.

Example:
Row vector:

$$P= [2, 4, 5, 6]$$

Column vector:

$$Q=\begin{bmatrix}2\\4\\5\end{bmatrix}$$

For detailed explanation of vectors you can refer to any basic mathematical site.
From equation (4) we can define the w vector and x vector as

$$w = \begin{bmatrix} w1 \\ w2 \\ w3 \\ w4 \\ . \\ . \\ wn \end{bmatrix} \text{ and } x = \begin{bmatrix} x1 \\ x2 \\ x3 \\ x4 \\ . \\ . \\ xn \end{bmatrix}$$

where w and x are n-dimensional column vectors.
So From equation (4) we write

$$\boxed{\mathbf{w.x + w_o = 0 \ldots (5)}}$$

w.x is the Dot product of two vectors w and x.

To get rid of dot product notation we use matrix multiplication notation in equation (5).

We have to write the column vector w in row format, because rule of matrix multiplication suggests that for two matrices to be multiplied the number of columns in 1st matrix should be equal to no. of rows in 2nd matrix. so we have transposed w as

w^T

(Column vector transposed to row vector) .i.e.

$$w^{\mathbf{T}} = [w_1, w_2, w_3, w_4 \ldots w_n]$$

is a row matrix and equation (5) is written as

$$[w_1, w_2, w_3, w_4 \ldots w_n] \begin{bmatrix} x1 \\ x2 \\ x3 \\ x4 \\ . \\ . \\ xn \end{bmatrix} + w_{\mathbf{0}} = 0$$

Thus

$$\boxed{w^{\mathbf{T}}x + w_{\mathbf{0}} = 0 \ldots (6)}$$

Equation (6) is the general equation of a hyperplane in any dimension.

What is w0?
In y=mx + c, c is the intercept with the y axis and the equation of a straight line passing through the origin is y-mx=0, where c=0.
Similarly from general equation of line from (1)

$$w_1x_1+w_2x_2+w_0=0$$

$$x2= -\frac{wo}{w2}+\frac{w1}{w2}x1$$

Thus after comparing with y=mx + c , we get m=w1/w2 and c= — wo/w2

$$\text{When c=0 then } \frac{wo}{w2}=0$$

$$\text{So } w_0=0$$

Thus equation (6) becomes

$$w^Tx=0\ldots (7)$$

Equation (7) is the general equation of a hyperplane passing through the origin which is valid in any dimension.

Why do we use the hyperplane equation wTx+w0=0 instead of y=mx+c?

For two reasons:
- it is easier to work in more than two dimensions with this notation,
- The vector w will always be normal to the hyperplane which helps in classification in machine learning. We will prove this statement now.

What is w?
W is a column weight vector containing coefficients of features x1, x2, x3…etc.

Now we have seen that

$$w^Tx=w.x$$

Now by the dot product definition of vectors,

$$w.x = ||w||\ ||x||\ \cos\theta$$

As $w^T x = 0$ so $||w||\ ||x||\ \cos\theta = 0$

Thus $\theta = 90^\circ$

This means that w and x are perpendicular. As the hyperplane passes through the x only, so we conclude that w or the weight vector is perpendicular to the hyperplane.

A hyperplane is a generalization of a plane.

- in one dimension, a hyperplane is called a point
- in two dimensions, it is a line
- in three dimensions, it is a plane
- in more dimensions you can call it an hyperplane

Distance of a Point from a Plane

Definition

The distance of a point P(x1,y1,z1) from a plane ax+by+cz+d=0 is the shortest distance from the point to the plane. This is computed as the perpendicular distance from the point to the plane.

Formula

The distance d is given by:

$$d = \frac{|ax_1 + by_1 + cz_1 + d|}{\sqrt{a^2 + b^2 + c^2}}$$

Where:

- (x_1, y_1, z_1): Coordinates of the point.
- a, b, c: Coefficients of the plane equation representing the normal vector.
- d: Constant term in the plane equation.

Derivation

1. The plane is represented by $ax + by + cz + d = 0$.
2. The normal vector to the plane is $\vec{n} = (a, b, c)$.
3. The vector from the point $P(x_1, y_1, z_1)$ to any point $Q(x, y, z)$ on the plane is:
$$\vec{PQ} = (x - x_1, y - y_1, z - z_1)$$
4. The perpendicular distance is the projection of $\vec{PQ}$ onto the normal vector $\vec{n}$:
$$d = \frac{|\vec{PQ} \cdot \vec{n}|}{|\vec{n}|}$$
5. Substituting $\vec{PQ} \cdot \vec{n} = a(x - x_1) + b(y - y_1) + c(z - z_1)$ and $|\vec{n}| = \sqrt{a^2 + b^2 + c^2}$, we simplify to:
$$d = \frac{|ax_1 + by_1 + cz_1 + d|}{\sqrt{a^2 + b^2 + c^2}}$$

To calculate the distance from a point P(x1,y1,z1) to a plane ax+by+cz+d=0 using the angle, we utilize the concept of the **dot product** and the **cosine of the angle** between vectors.

Steps:

1. **Understand the plane's normal vector**: The coefficients a, b, c of the plane equation represent a normal vector $\mathbf{n} = (a, b, c)$ to the plane.
2. **Form the point-to-plane vector**: Take any point $Q(x_0, y_0, z_0)$ on the plane (usually the origin or any point satisfying the plane equation) and calculate the vector from Q to P, denoted as $\mathbf{v} = \mathbf{P} - \mathbf{Q}$.
3. **Project $\mathbf{v}$ onto $\mathbf{n}$**: The perpendicular distance is the projection of $\mathbf{v}$ onto $\mathbf{n}$. Use the formula for the projection length:

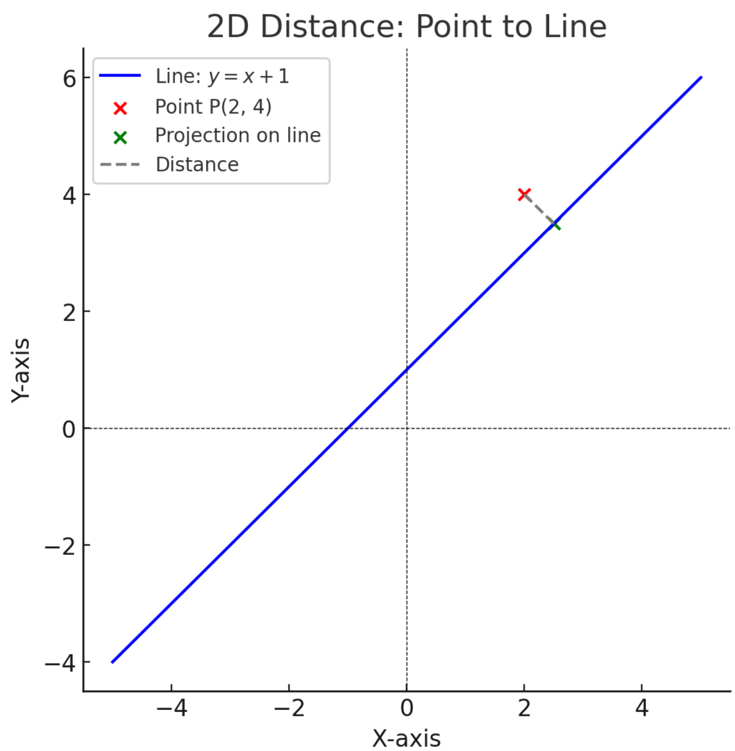
$$\text{Distance} = |\mathbf{v} \cdot \mathbf{n}| / \|\mathbf{n}\|$$

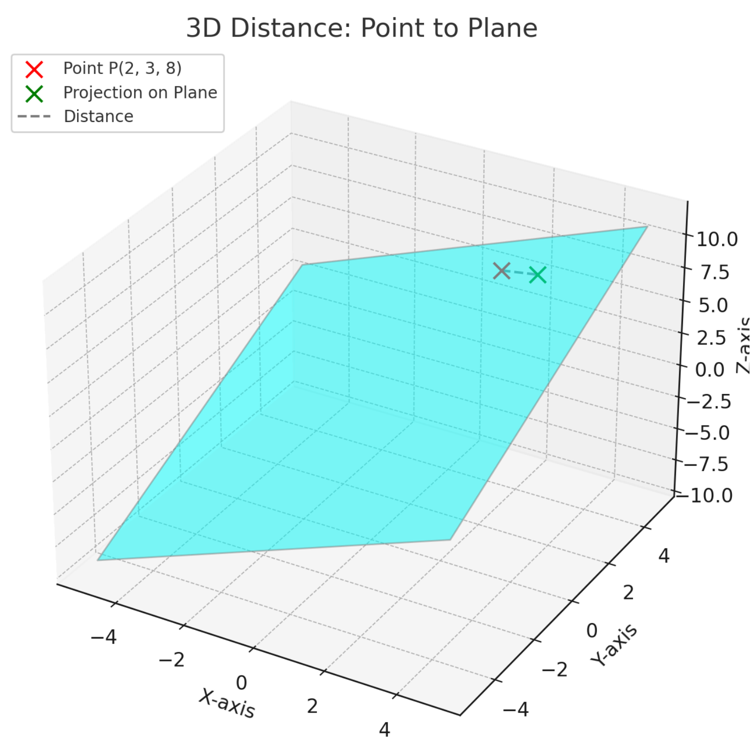
   - $\mathbf{v} \cdot \mathbf{n}$: Dot product between $\mathbf{v}$ and $\mathbf{n}$.
   - $\|\mathbf{n}\| = \sqrt{a^2 + b^2 + c^2}$: Magnitude of the normal vector.
4. **Using the cosine angle**:
   - The dot product $\mathbf{v} \cdot \mathbf{n} = \|\mathbf{v}\| \|\mathbf{n}\| \cos\theta$, where θ is the angle between $\mathbf{v}$ and $\mathbf{n}$.
   - Rearrange to find $\cos\theta = (\mathbf{v} \cdot \mathbf{n}) / (\|\mathbf{v}\| \|\mathbf{n}\|)$.

Short Formula:

For a point $P(x_1, y_1, z_1)$, the distance D to a plane $ax + by + cz + d = 0$ is:

$$D = \frac{|ax_1 + by_1 + cz_1 + d|}{\sqrt{a^2 + b^2 + c^2}}$$





Applications in Machine Learning

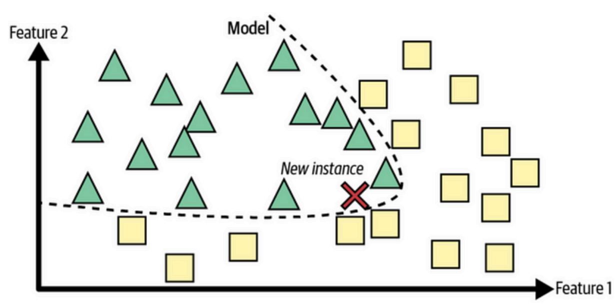
1. **Support Vector Machines (SVM)**:

- The distance helps determine the margin between the hyperplane and data points.
- Maximizing the margin improves generalization.

2. **Regression**:
    - The distance measures errors when fitting a plane to data in multivariate linear regression.
3. **Clustering**:
    - Planes are used as cluster boundaries in algorithms like K-Means or Gaussian Mixture Models.
4. **Dimensionality Reduction**:
    - Computing the distance to a subspace (plane) helps in Principal Component Analysis (PCA).

Model-Based Learning

In model-based learning, the training data is used to create a model that can be **generalized** to new data. The model is typically created using statistical algorithms such as linear regression, logistic regression, decision trees, and neural networks. These algorithms use the training data to create a mathematical model that can be used to predict outcomes.



Key Features:

1. **Eager Learning**:
   The algorithm builds the model during the training phase and uses it for predictions.
2. **Global Generalization**:
   The model attempts to generalize across the entire training dataset to capture the overall trends.
3. **Optimization**:
   Model training involves optimizing a function (e.g., minimizing a loss function).

Advantages of Model-Based Learning

- Faster predictions: Model-based learning is typically faster than instance-based learning because the model is already created and can be used to make predictions quickly.
- More accurate predictions: Model-based learning can often make more accurate predictions than instance-based learning because the model is trained on a large dataset and can generalize to new data.
- Better understanding of data Model-based learning allows you to gain a better understanding of the relationships between input and output variables. This can help identify which variables are most important in making predictions.

Disadvantages of Model-Based Learning

- Requires a large dataset: model-based learning requires a large dataset to train the model. This can be a disadvantage if you have a small dataset.
- Requires expert knowledge: Model-based learning requires expert knowledge of statistical algorithms and mathematical modeling. This can be a disadvantage if you don't have the expertise to create the model.
- Requires expert knowledge: Model-based learning requires expert knowledge of statistical algorithms and mathematical modeling. This can be a disadvantage if you don't have the expertise to create the model.

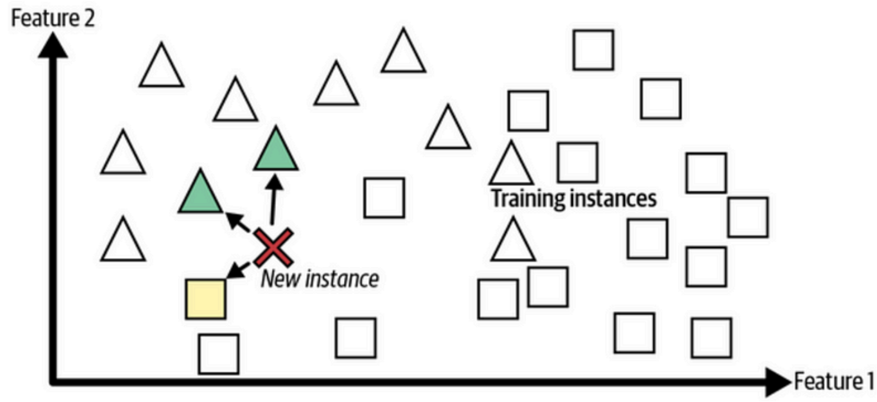
Example of Model-Based Learning

An example of model-based learning is predicting the price of a house based on its size, number of rooms, location, and other features. In this case, a model could be created using linear regression to predict the price of the house based on these features. The model would be trained on a dataset of house prices and features and then used to make predictions on new data.

Instance-Based Learning

Instance-based learning involves using the entire dataset to make predictions. The machine learns by storing all instances of data and then using these instances to make predictions on new data. The machine compares the new data to the instances it has seen before and uses the closest match to make a prediction.

In instance-based learning, no model is created. Instead, the machine stores all of the training data and uses this data to make predictions based on new data. Instance-based learning is often used in pattern recognition, clustering, and anomaly detection.



Key Features:

1. **Lazy Learning**:
   The algorithm defers the processing until a query is made (no explicit training phase). Example: K-Nearest Neighbors (KNN).
2. **Local Generalization**:
   Predictions depend only on a small subset of instances that are most similar to the query.
3. **Similarity Metric**:
   Distance measures like Euclidean distance, Manhattan distance, etc., are used to determine how close an input instance is to stored examples.

Advantages of Instance-Based Learning

- No need for model creation: Instance-based learning doesn't require creating a model, which can be an advantage if you don't have the expertise to create the model.
- Can handle small datasets: Instance-based learning can handle small datasets because it doesn't require a large dataset to create a model.
- More flexibility: Instance-based learning can be more flexible than model-based learning because the machine stores all instances of data and can use this data to make predictions.

Disadvantages of Instance-Based Learning

- Slower predictions: Instance-based learning is typically slower than model-based learning because the machine has to compare the new data to all instances of data in order to make a prediction.
- Less accurate predictions: Instance-based learning can often make less accurate predictions than model-based learning because it doesn't have a mathematical model to **generalize** from.

- Limited understanding of data: Instance-based learning doesn't provide as much insight into the relationships between input and output variables as model-based learning does.

Example of Instance-Based Learning

An example of instance-based learning is the k-nearest neighbor algorithm. This algorithm involves storing all instances of data and then using the closest k instances to make a prediction. For example, in a classification problem where the goal is to predict the species of a flower based on its petal length and width, the k-nearest neighbor algorithm would store all instances of flowers along with their species and petal length and width measurements. When a new flower is presented, the algorithm will find the k closest instances based on their petal length and width measurements and then assign the species of the majority of those k instances to the new flower.

Usual/Conventional Machine Learning	**Instance Based Learning**
Prepare the data for model training	Prepare the data for model training. No difference here
Train model from training data to estimate model parameters i.e. discover patterns	Do not train model. Pattern discovery postponed until scoring query received
Store the model in suitable form	There is no model to store
Generalize the rules in form of model, even before scoring instance is seen	No generalization before scoring. Only generalize for each scoring instance individually as and when seen
Predict for unseen scoring instance using model	Predict for unseen scoring instance using training data directly
Can throw away input/training data after model training	Input/training data must be kept since each query uses part or full set of training observations
Requires a known model form	May not have explicit model form
Storing models generally requires less storage	Storing training data generally requires more storage