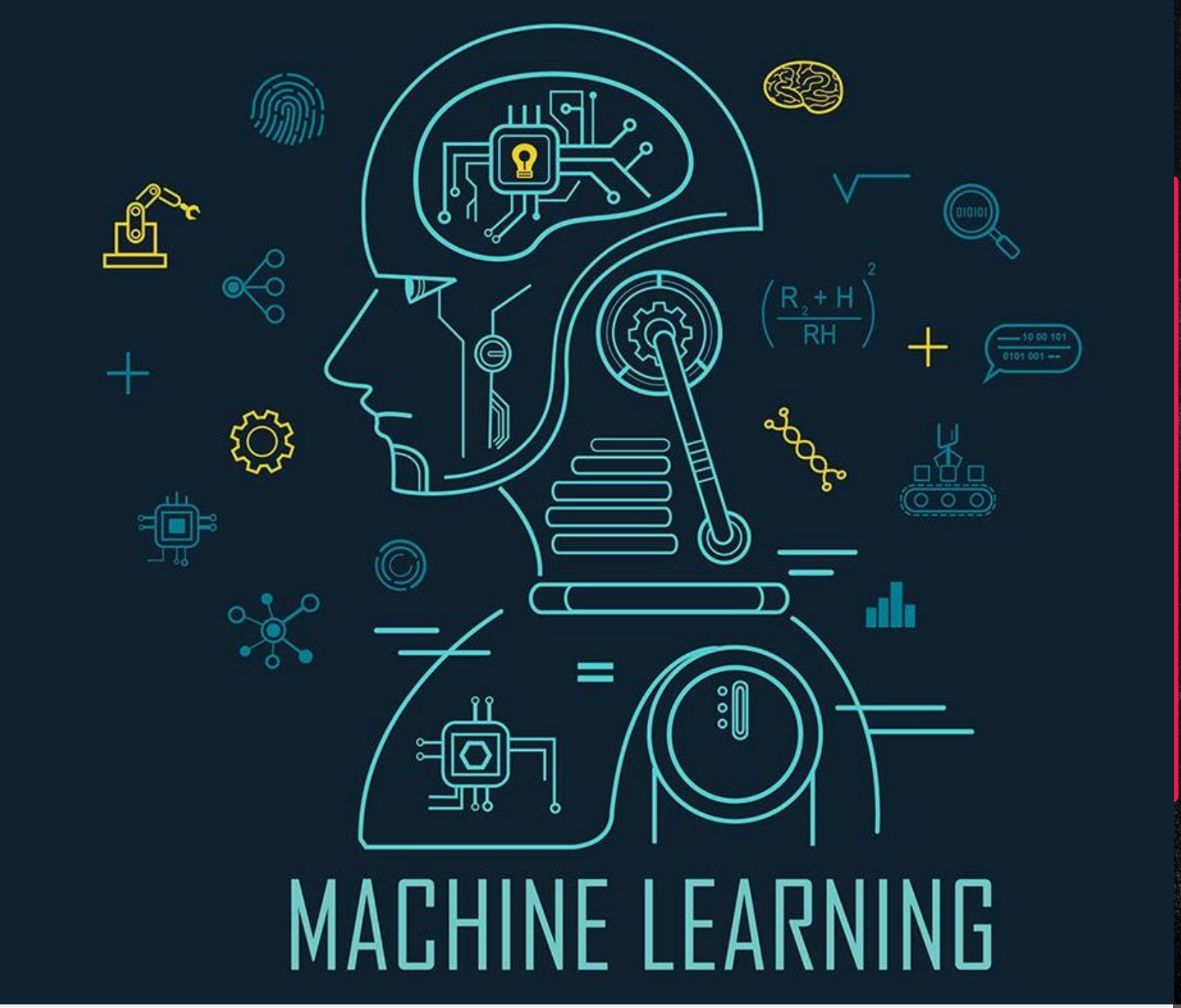


IT 323
ML: Machine Learning

Conducted by Ruchika Pharswan

at Delhi Technological University

8 Aug 2024

Brief About Me

Email : ruchikapharswan2024@gmail.com

B.Tech (CSE) M.Tech (CSE) || Ph.D. coursework graduated

Teaching Assistant IIT Delhi, Assistant Prof. GGSIPU, TA IIHT

Journal, Book Chapters, Conferences papers || Active Reviewer

Governance of AI ,Technology Adoption, Social Media Analysis, User Behavior,

INSTRUCTIONS

NO entry in class after 15 mins , once class the started.

NO proxies attendance. NO back attendances will be given.

If there will be mass bunk, then for next lecture, few of you will be randomly selected to present assigned topics.

Topic presentation (10), Assignments (5) will hold weightage of CWS component.

Also feel free to ask any doubts in class or even after class you can drop your query in AIES WhatsApp group. Kindly send your rest of the requests and queries to CR , refrain individual messages unless important. Thanks.

DO NOT UPLOAD THIS PPT. ANYWHERE !

B.Tech. Information Technology				
Course code: Course Title	**Course Structure**			**Pre-Requisite**
Machine Learning	**L**	**T**	**P**	Probability, Statistics and Stochastic Processes, Linear Algebra
	3	**0**	**2**	

Course Objective: 1. To understand various key paradigms for machine learning approaches.
2.To familiarize with the mathematical and statistical techniques used in machine learning.
3. To understand and differentiate among various machine learning techniques.

By the end of this course you can assure following :

- Understand the fundamental concepts and algorithms of machine learning.
- Develop a comprehensive understanding of fundamental machine learning concepts, algorithms, and techniques, including supervised and unsupervised learning, classification, regression, clustering, and dimensionality reduction.
- Apply principles and algorithms to evaluate models generated from data.
- Learn to critically evaluate the performance of machine learning models using appropriate metrics.
- Develop the ability to identify and formulate problems suitable for machine learning solutions, design appropriate models, and interpret results in practical applications.

Course Outline

S.No.	Contents	Contact Hours
1.	**Introduction to Machine Learning:** Overview of different tasks: classification, regression, clustering, control, Concept learning, information theory and decision trees	6
2.	**Supervised Learning:** Decision trees, nearest neighbors, linear classifiers and kernels, neural networks, linear regression; Support Vector Machines.	8
3.	**Unsupervised Learning:** Clustering, Expectation Maximization, Dimensionality Reduction, Feature Selection, PCA, factor analysis, manifold learning.	8
4.	**Reinforcement Learning:** Value iteration; policy iteration; TD learning; Q learning; actor-critic.	6
5.	**Other Topics:** Bayesian learning, online learning. Learning theory, Bias Variance trade-offs	6
6.	**Recent applications & ResearchTopics:** Applications in the fields of web and data mining, text recognition, speech recognition	8
TOTAL		**42**

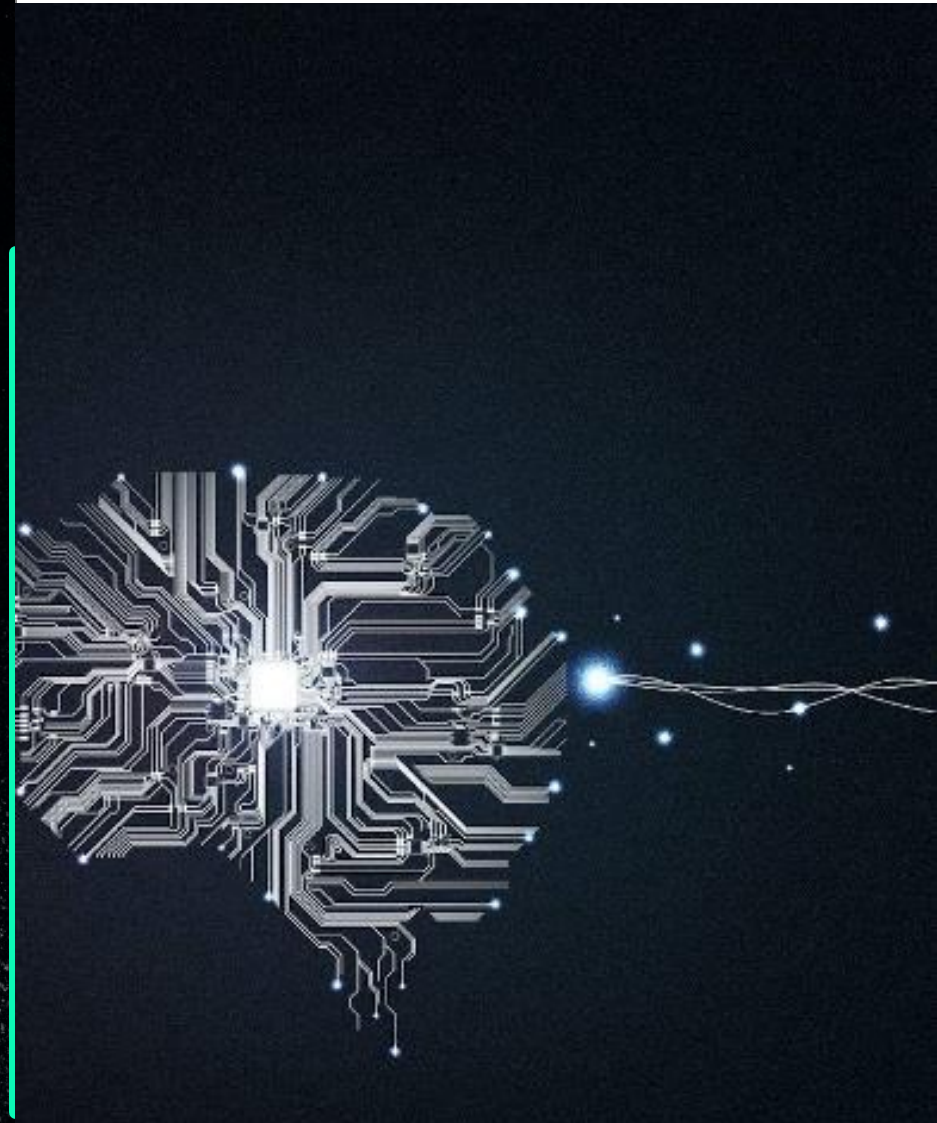
S.No.	Name of Books/Authors/Publishers	Year of Publication / Reprint
1	Introduction to Machine Learning, Alpaydin, E.,PHI Learning Pvt. Ltd.	2015
2	Machine Learning, Tom Mitchell, McGraw Hill	2017
3	Applied Machine Learning by M.Gopal|, McGraw Hill, ISBN: 978-9354601590	2021
4	Understanding Machine Learning: From Theory to Algorithms, 1st Edition,by Shai Shalev-Shwartz, Cambridge University Press	2015
5	Pattern Recognition and Machine Learning by Christopher Bishop, Springer Verlag	2006
6	Pattern Classification by Richard Duda, Wiley Publisher	2007

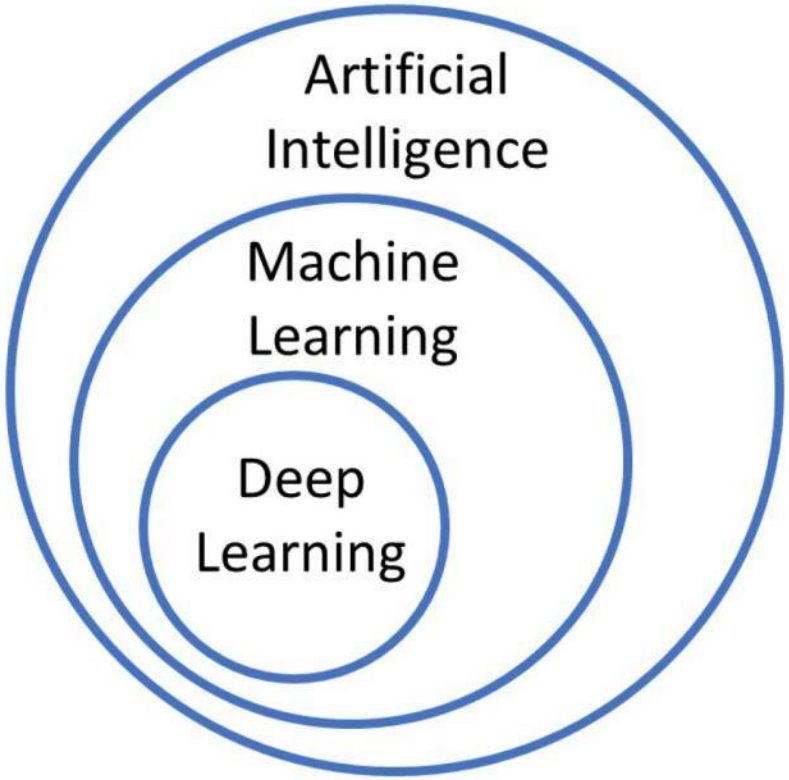
UNIT 1

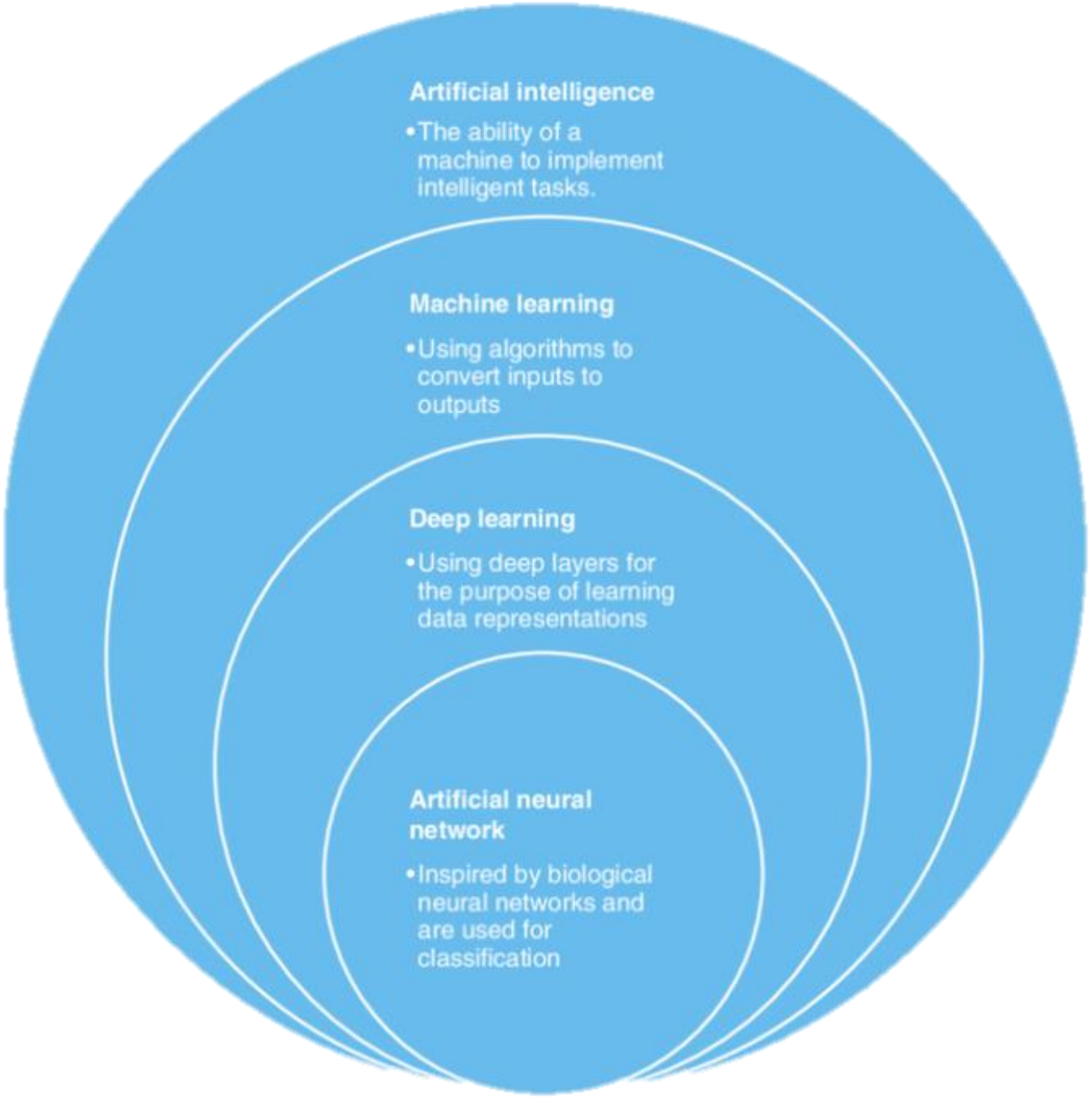
Introduction to machine learning

Sources : Books, Internet, Self

Referring to Prof. Dinesh Kumar Vishwakarma Lecture Notes.

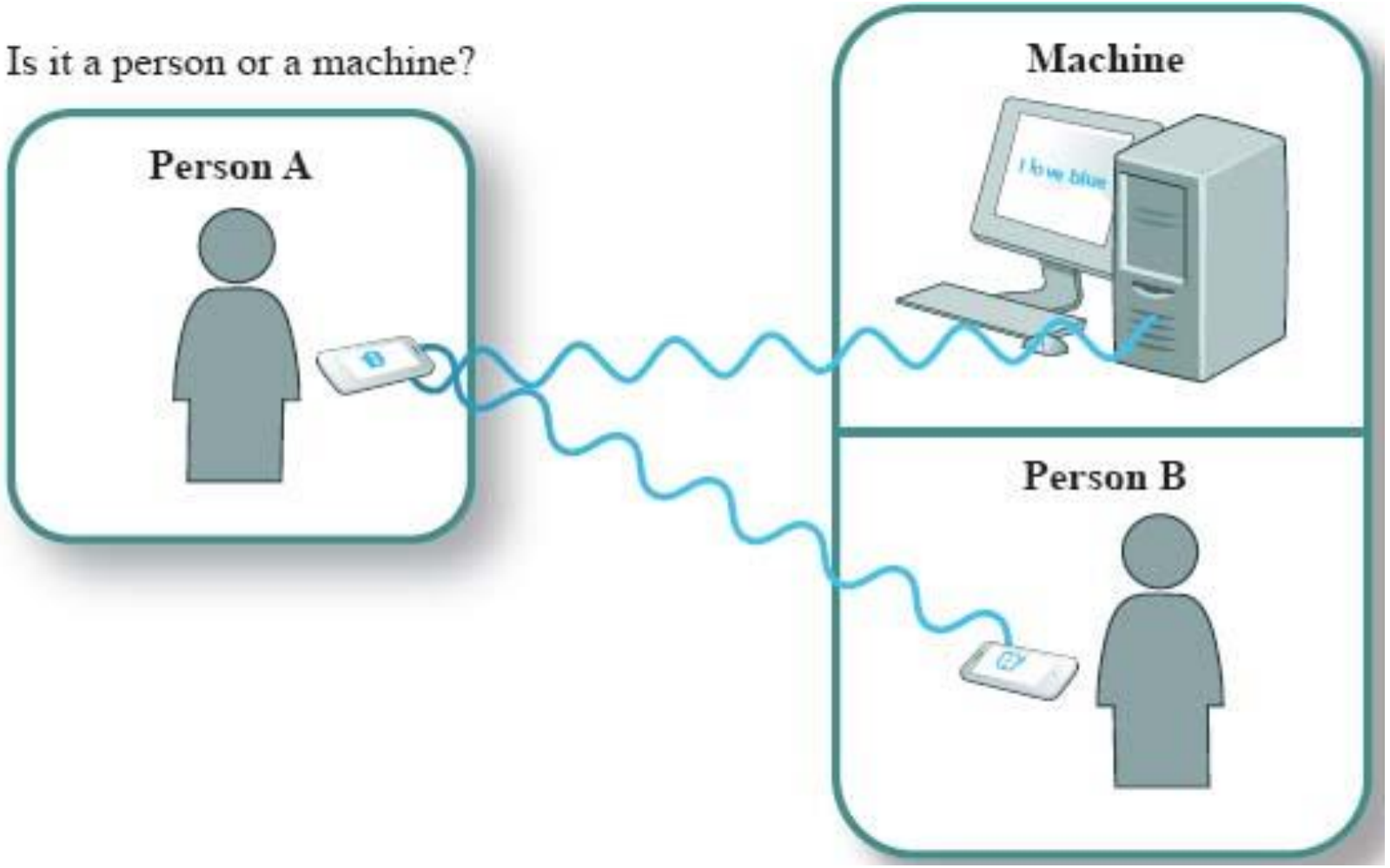






M**achine learning** is an application of artificial intelligence (AI) that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. **Machine learning** focuses on the development of computer programs that can access data and use it learn for themselves.

Any hunches ?



- Foundation of AI : Turing Test

Alan Turing

The Turing Test is a method of inquiry in artificial intelligence (AI) for determining whether or not a computer is capable of thinking like a human being.

Fool the tester

- **Alan Turing's proposal in his paper "Computing Machinery and Intelligence", in which the question "Can machines think?" is replaced with the question "Can machines do what we (as thinking entities) can do?".**

The Turing Test was the technique of exploration in AI for deciding whether or not a computer is competent of thinking like a human being.

Limitation of Turing test : *Nature of question was limited*

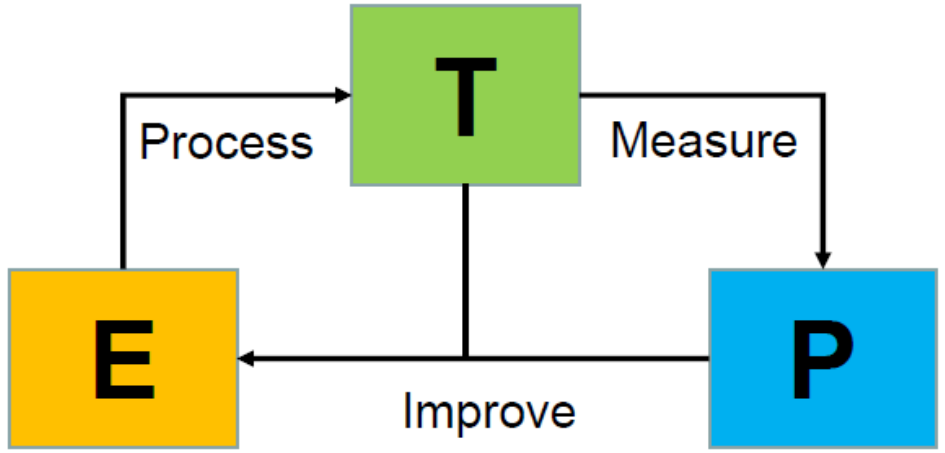
Computer score high only if the questions were formulated in the queries, that can be answered either in *"Yes" or "No"* or related to a narrow field of knowledge.

Whereas when questions were *open-ended* and needed *conversational answers,* computer scored less.

- **A computer program is said to learn from experience *E* with respect to some class of tasks *T* and performance measure *P* if its performance at tasks in *T*, as measured by *P*, improves with experience *E*.**

Defining the Learning Task

Improve on task T, with respect to
performance metric P, based on experience E



T: Playing checkers
P: Percentage of games won against an arbitrary opponent
E: Playing practice games against itself

T: Recognizing hand-written words
P: Percentage of words correctly classified
E: Database of human-labeled images of handwritten words

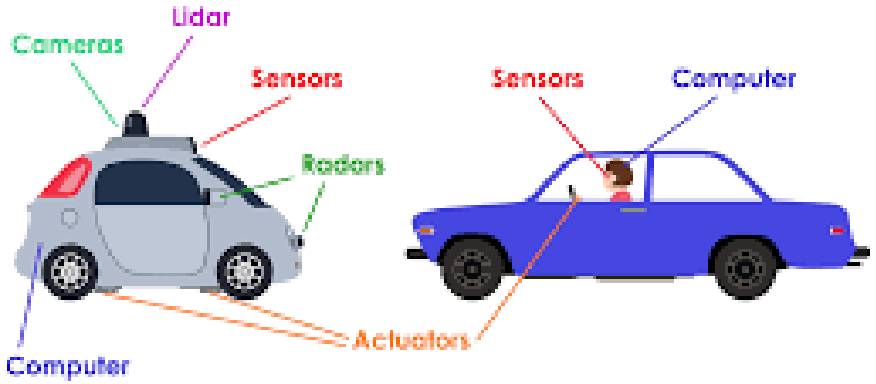
T: Driving on four-lane highways using vision sensors
P: Average distance traveled before a human-judged error
E: A sequence of images and steering commands recorded while observing a human driver.

PEAS Representation

- PEAS stands for **PERFORMANCE , ENVIRONMENT, ACTUATORS, SENSORS**.

PEAS is a type of model on which an AI agent works upon. When we define an AI agent or rational agent, then we can group its properties under PEAS representation model. Here performance measure is the objective for the success of an agent's behavior.

Let's suppose a self-driving car then PEAS representation will be:



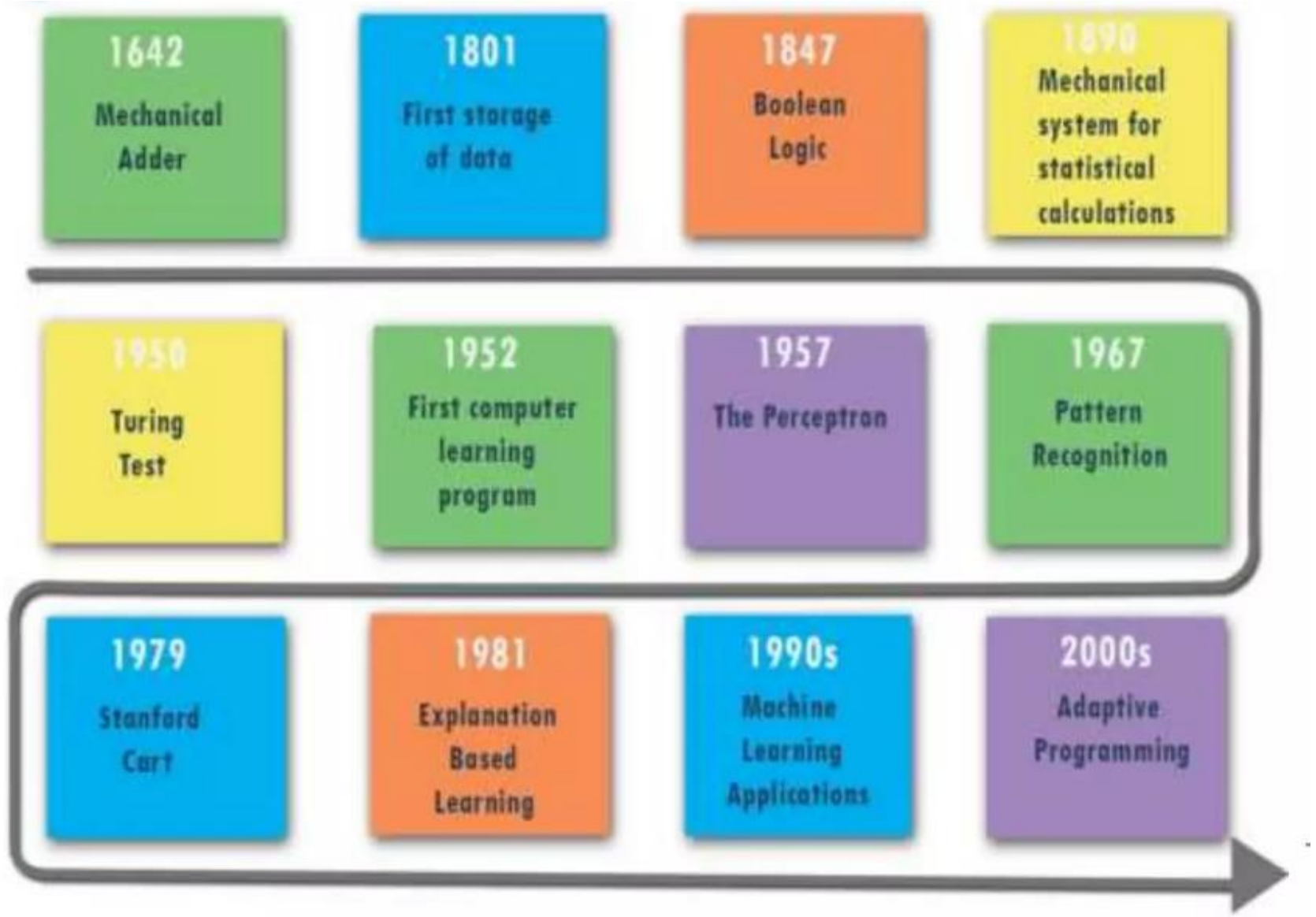
Performance: Safety, time, legal drive, comfort

Environment: Roads, other vehicles, road signs, pedestrian

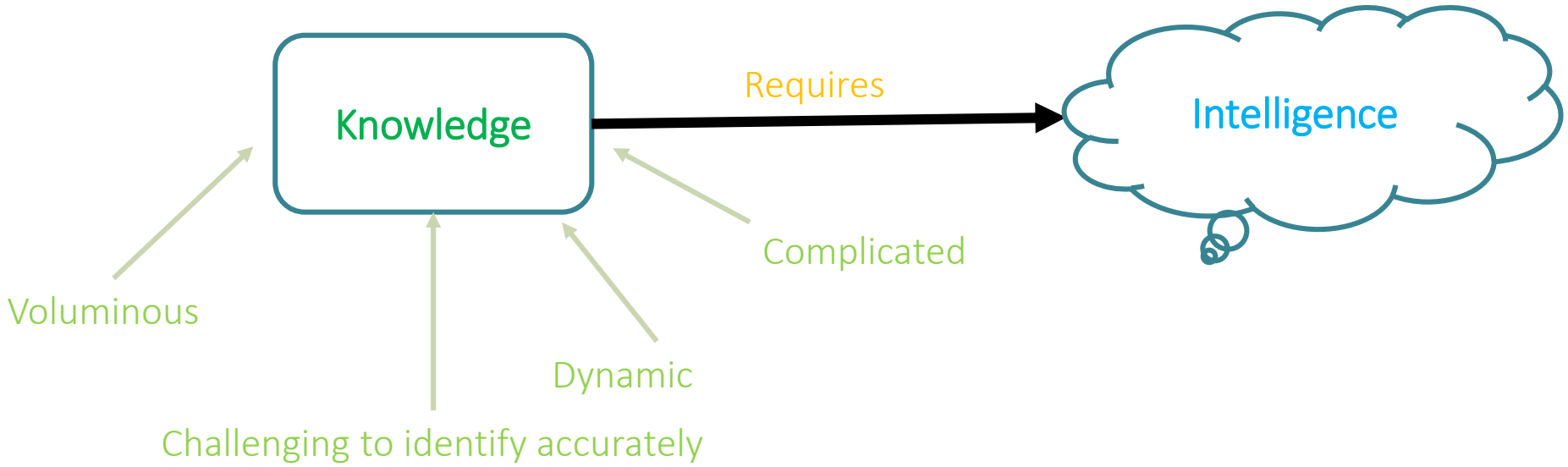
Actuators: Steering, accelerator, brake, signal, horn

Sensors: Camera, sonar.

History of Machine learning



- **As intelligence requires knowledge, it is necessary for the computers to acquire knowledge.**



Intelligence requires *knowledge*. **Knowledge** is *collection of facts*.
ML technique is a **method** that *achieves knowledge*.

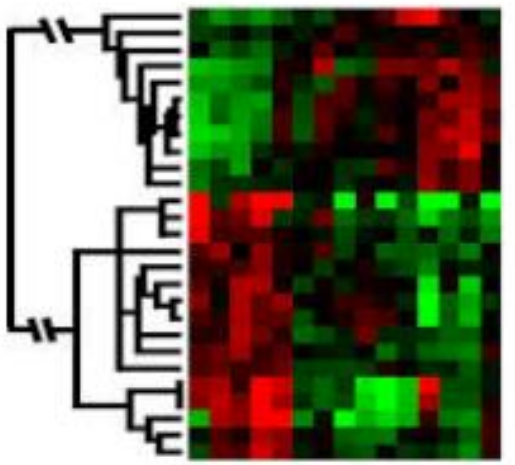
- ***Machine learning refers to a system capable of the autonomous acquisition and integration of knowledge***

When Do We Use Machine Learning?

ML is used when: No Human Expertise, Unknown Challenges, data driven learning

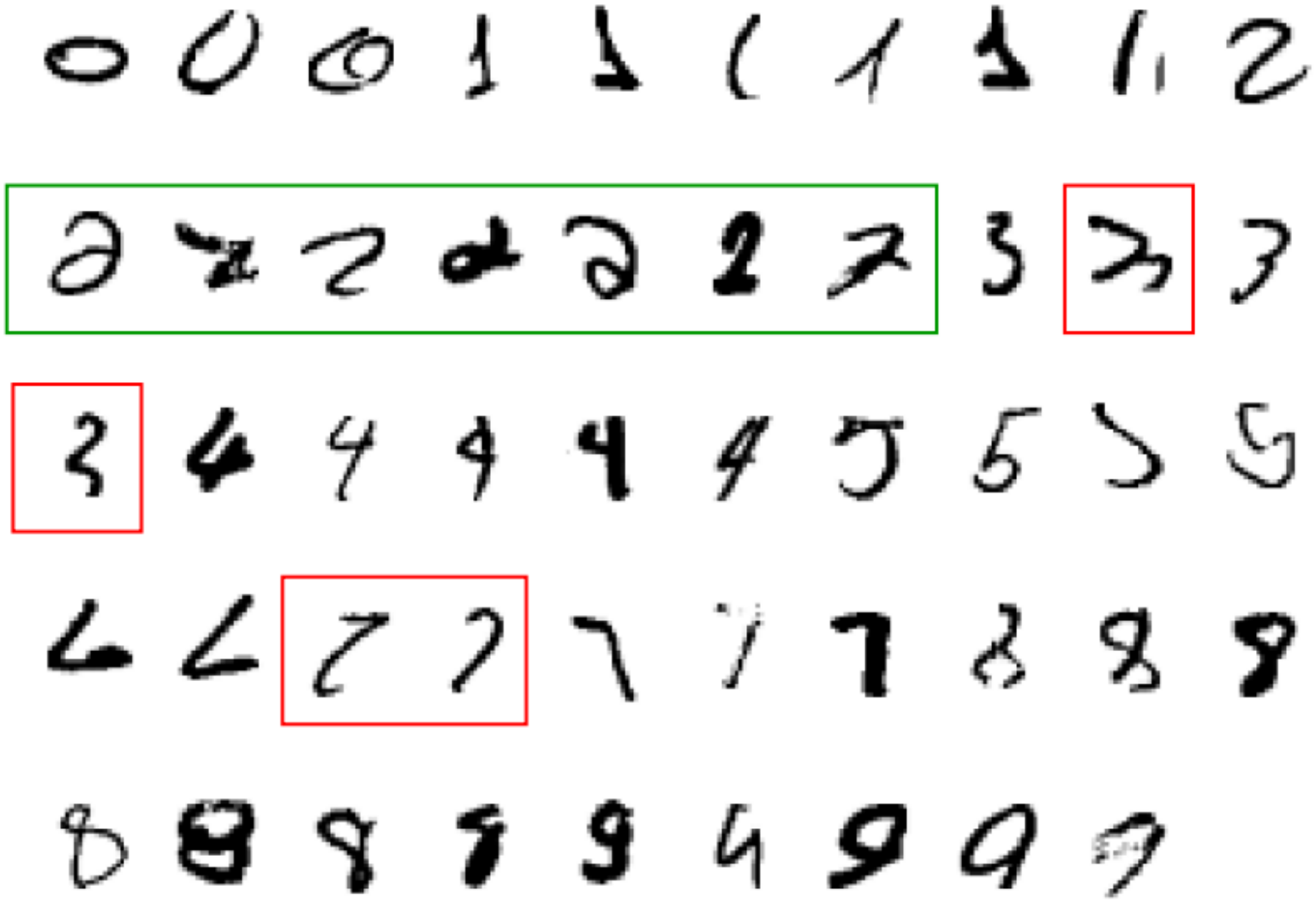
- Human expertise does not exist (navigating on Mars)
- Humans can't explain their expertise (speech recognition)
- Models must be customized (personalized medicine)
- Models are based on huge amounts of data (genomics)



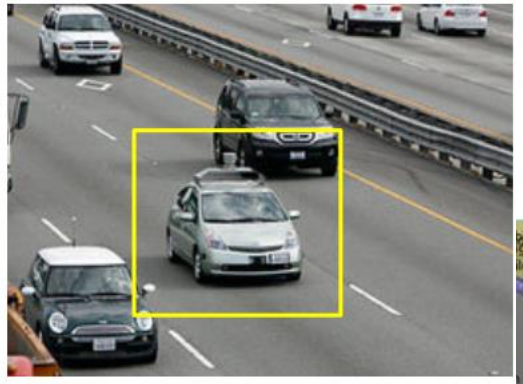



A classic example of a task that requires machine learning:

It is very hard to say what makes a 2



Autonomous Cars

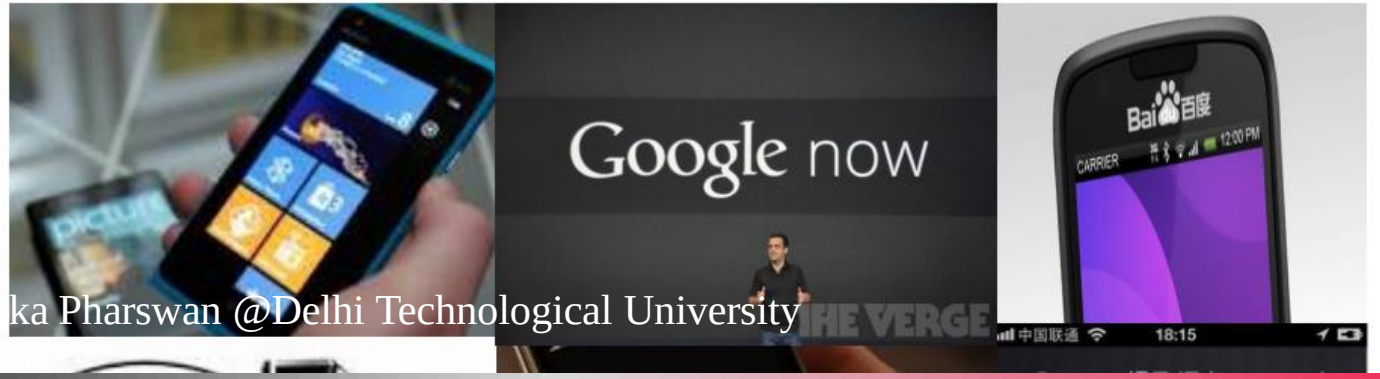


- Nevada made it legal for autonomous cars to drive on roads in June 2011
- As of 2013, four states (Nevada, Florida, California, and Michigan) have legalized autonomous cars

Penn's Autonomous Car →
(Ben Franklin Racing Team)

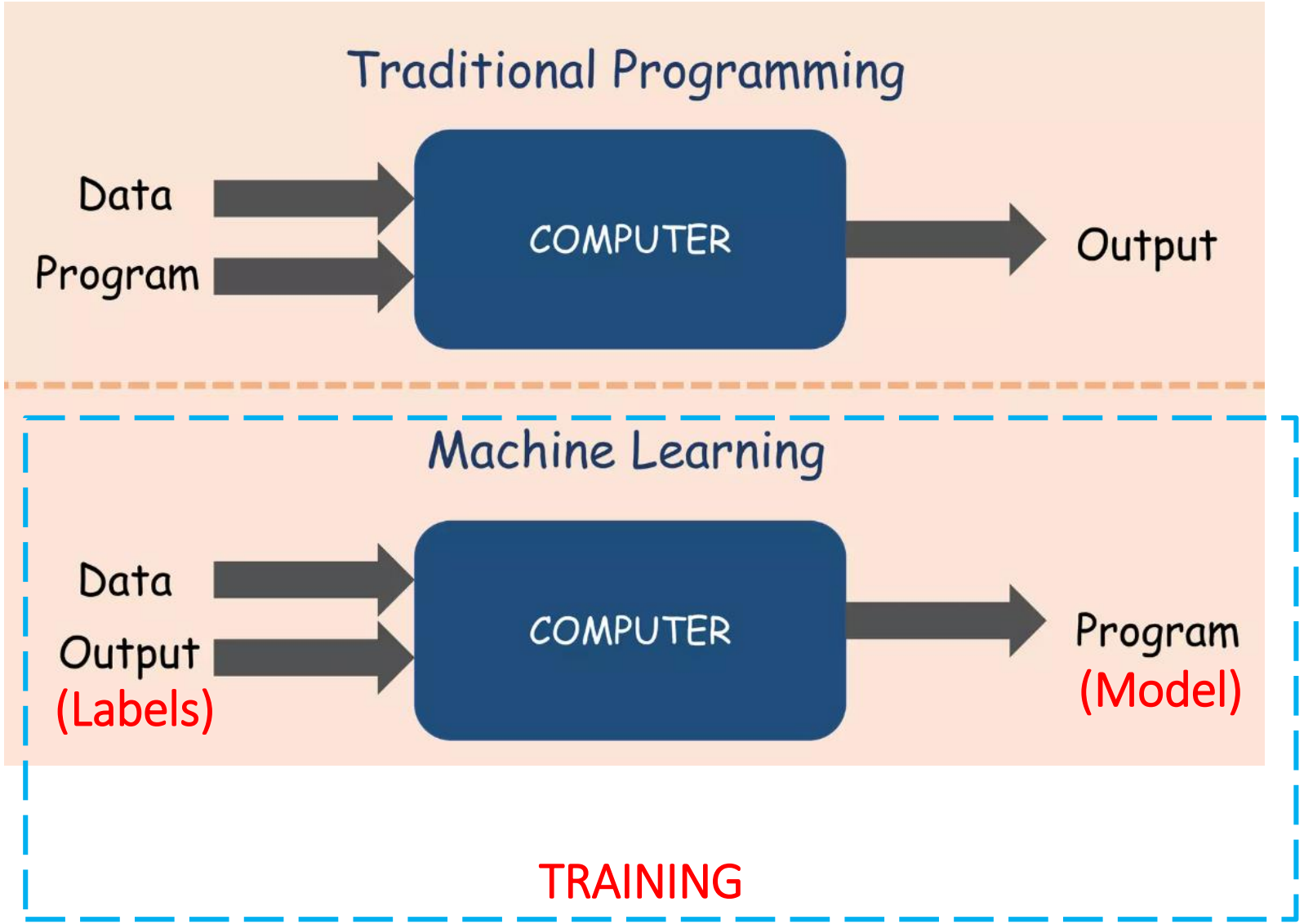
Scene Labeling via Deep Learning

Impact of Deep Learning in Speech Technology



Some more examples of tasks that are best solved by using a learning algorithm

- Recognizing patterns:
  - Facial identities or facial expressions
  - Handwritten or spoken words
  - Medical images
- Generating patterns:
  - Generating images or motion sequences
- Recognizing anomalies:
  - Unusual credit card transactions
  - Unusual patterns of sensor readings in a nuclear power plant
- Prediction:
  - Future stock prices or currency exchange rates



Machine Learning System

Inputs: Data + Output

Data: Similar to the traditional system, this is the information you want to work with.

However, in a machine learning context, this data is typically much larger and more complex.

Output: This is the desired result that you want the system to produce based on the data.

For example, if you want a machine learning model to recognize cats in images, the output could be **labels** indicating whether a cat is present in each image.

Output: Program (Model)

1. Representation of Some Phenomenon
2. Non-Maths /Stats Model

A **MODEL** is a mathematical or computational representation that encapsulates patterns, relationships, or rules that have been learned from data.

A **model** in machine learning is a learned representation that can make predictions or decisions based on input data. It's the core of what makes machine learning systems powerful and adaptable.

Often Describe Relationship between Variables

Types

- **Deterministic Models (no randomness)**
- **Probabilistic Models (with randomness)**

How a Model Works

1.Training:

1. During the training phase, a machine learning algorithm processes a dataset that includes both input data and the corresponding desired outputs (also known as labels or targets).
2. The algorithm adjusts its internal parameters in such a way that it can map the input data to the correct output. These internal parameters form the model.

2.Inference:

1. After training, the model can take new, unseen input data and apply the learned rules or patterns to predict an output.

Model Output After training, the model can make predictions, classify data, or even generate new content based on what it has learned.

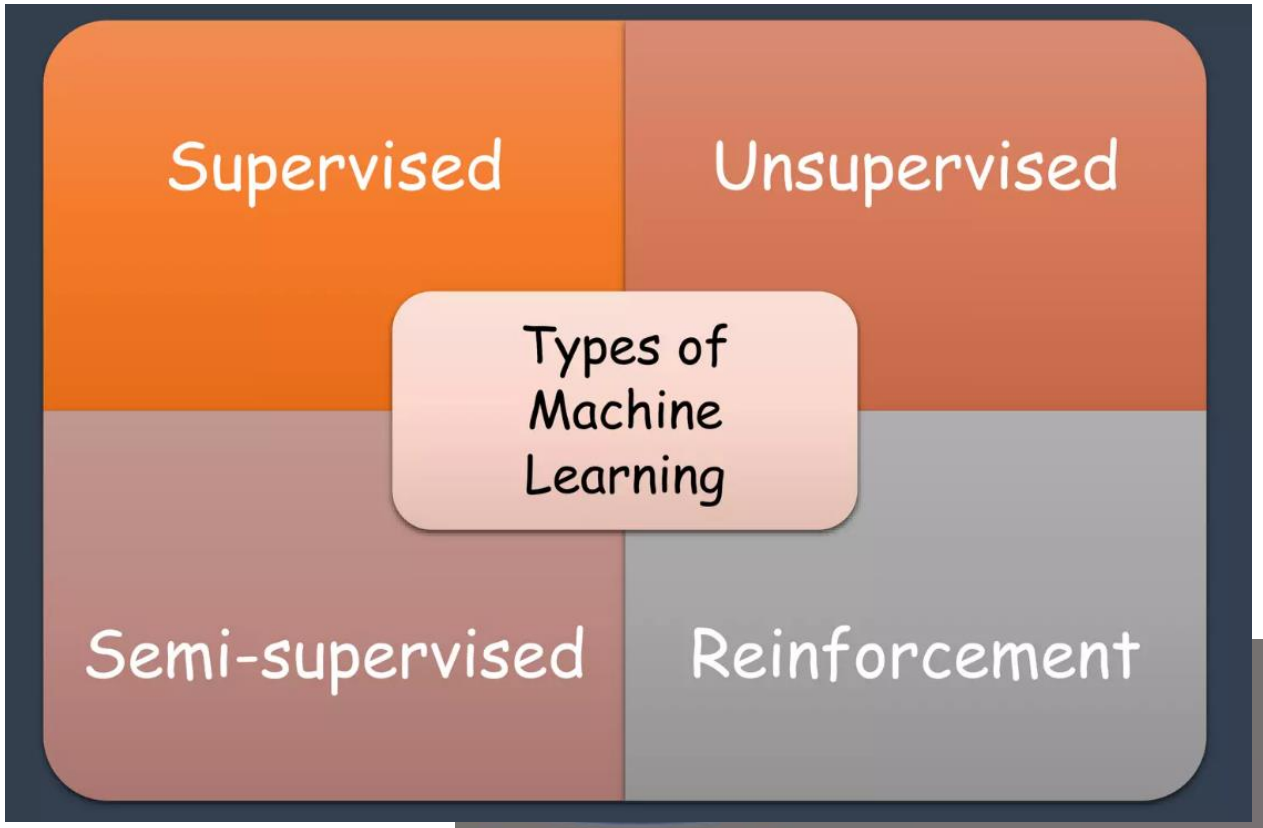
TASK OF MACHINE LEARNING

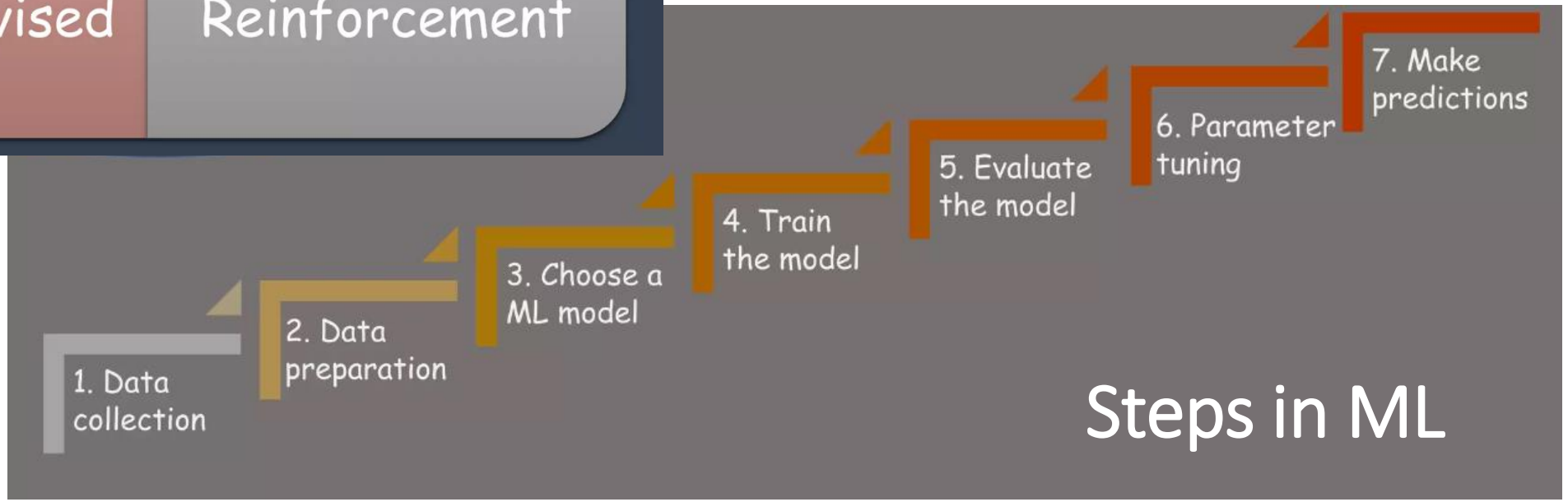
Types of Models

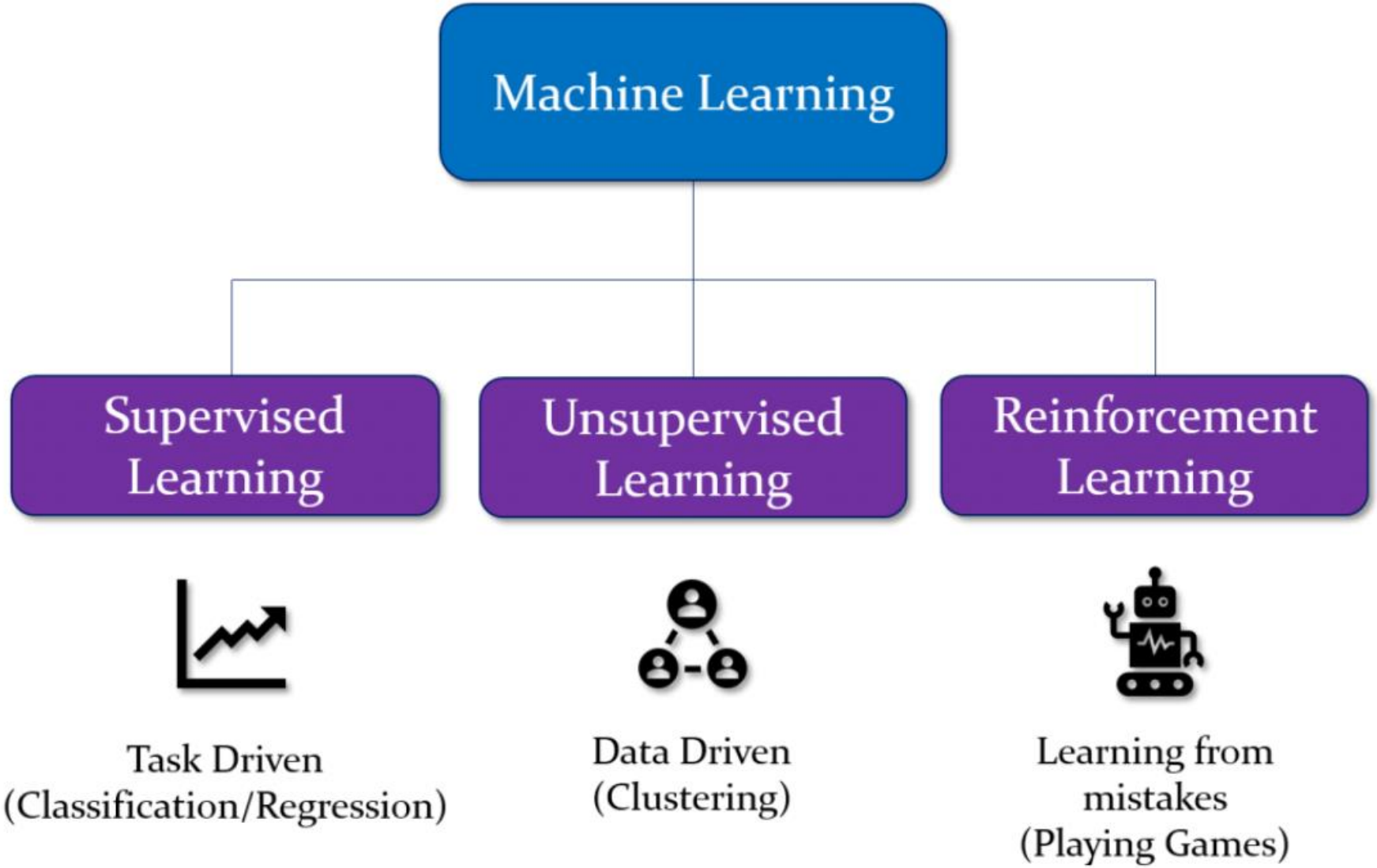
•**Supervised Learning Models**: Trained on labeled data, where the correct output is provided during training.
•**Unsupervised Learning Models**: Learn from unlabeled data, identifying patterns or structures in the data without explicit outputs.
•**Reinforcement Learning Models**: Learn by interacting with an environment, receiving feedback in the form of rewards or penalties.

- **Supervised (inductive) learning**
  - Given: training data + desired outputs (labels)
- **Unsupervised learning**
  - Given: training data (without desired outputs)
- **Semi-supervised learning**
  - Given: training data + a few desired outputs
- **Reinforcement learning**
  - Rewards from sequence of actions

Aspect	Semi-Supervised Learning	Reinforcement Learning
Objective	Improve learning by leveraging a small amount of labeled data along with a large amount of unlabeled data.	Train an agent to make decisions by maximizing cumulative rewards through interactions with an environment.
Data Usage	Uses both labeled and unlabeled data.	Uses feedback in the form of rewards and penalties from interactions with an environment.
Learning Process	Learns from the labeled data and refines the model using the structure in the unlabeled data.	Learns through trial and error, adjusting strategies based on received rewards and penalties.
Environment	Typically static datasets (labeled and unlabeled).	Involves an agent interacting dynamically with an environment.
Exploration vs. Exploitation	Not a primary focus. Relies on labeled data and generalizes with unlabeled data.	Balances exploration (trying new actions) and exploitation (using known successful actions) to maximize rewards.
Example Applications	Image classification with limited labeled images, text classification.	Game playing (e.g., AlphaGo), robotics, autonomous vehicles.
Key Challenge	Efficiently using the large amount of unlabeled data to improve model accuracy.	Learning the optimal policy to maximize long-term rewards through sequential decision-making.


Supervised
Unsupervised
Types of Machine Learning
Semi-supervised
Reinforcement


1. Data collection
2. Data preparation
3. Choose a ML model
4. Train the model
5. Evaluate the model
6. Parameter tuning
7. Make predictions
Steps in ML

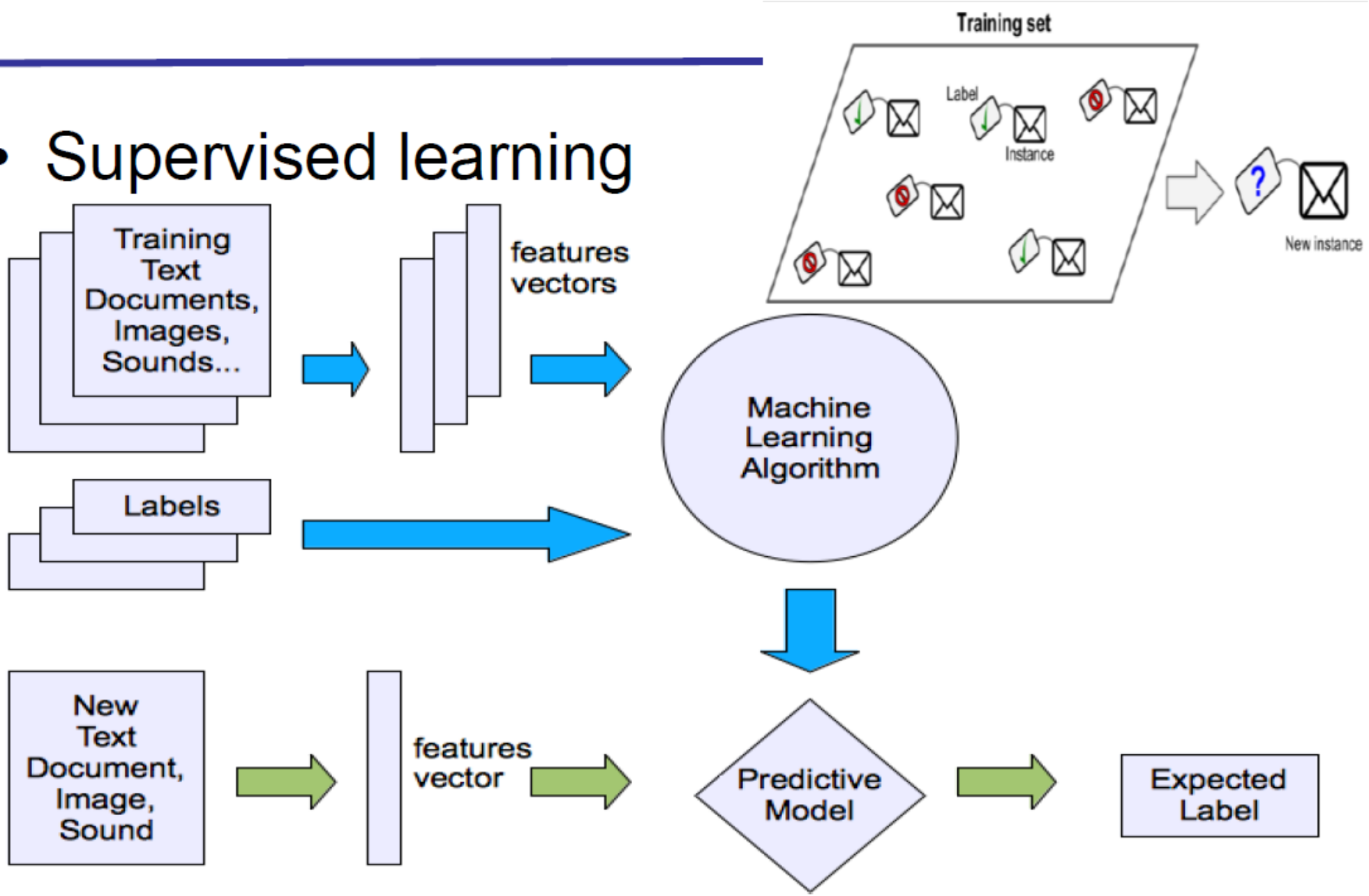

Machine Learning
Supervised Learning
Unsupervised Learning
Reinforcement Learning
Task Driven
(Classification/Regression)
Data Driven
(Clustering)
Learning from
mistakes
(Playing Games)

Supervised Learning

- Supervised learning is one of the most basic types of machine learning. In this type, the machine learning algorithm is trained on labeled data. Even though the data needs to be labeled accurately for this method to work, supervised learning is extremely powerful when used in the right circumstances.

- Great example of supervised learning is text classification problems. In this set of problems, the goal is to predict the class label of a given piece of text. One particularly popular topic in text classification is to predict the sentiment of a piece of text, like a tweet or a product review.

- Supervised learning



Supervised Learning

Supervised learning involves learning a function that maps input data to output labels, based on a set of input-output pairs. The model is trained on labeled data, where the correct output is provided for each input in the training set.

•**Classification**:

- **Objective**: Predict a categorical label for given input data.
- **Example**: Spam detection in emails (spam or not spam), image recognition (cat or dog).
- **Algorithms**: Logistic Regression, Decision Trees, Support Vector Machines (SVM), k-Nearest Neighbors (k-NN), Neural Networks.

•**Regression**:

- **Objective**: Predict a continuous value for given input data.
- **Example**: Predicting house prices, forecasting stock prices.
- **Algorithms**: Linear Regression, Ridge Regression, Lasso Regression, Polynomial Regression, Neural Networks.

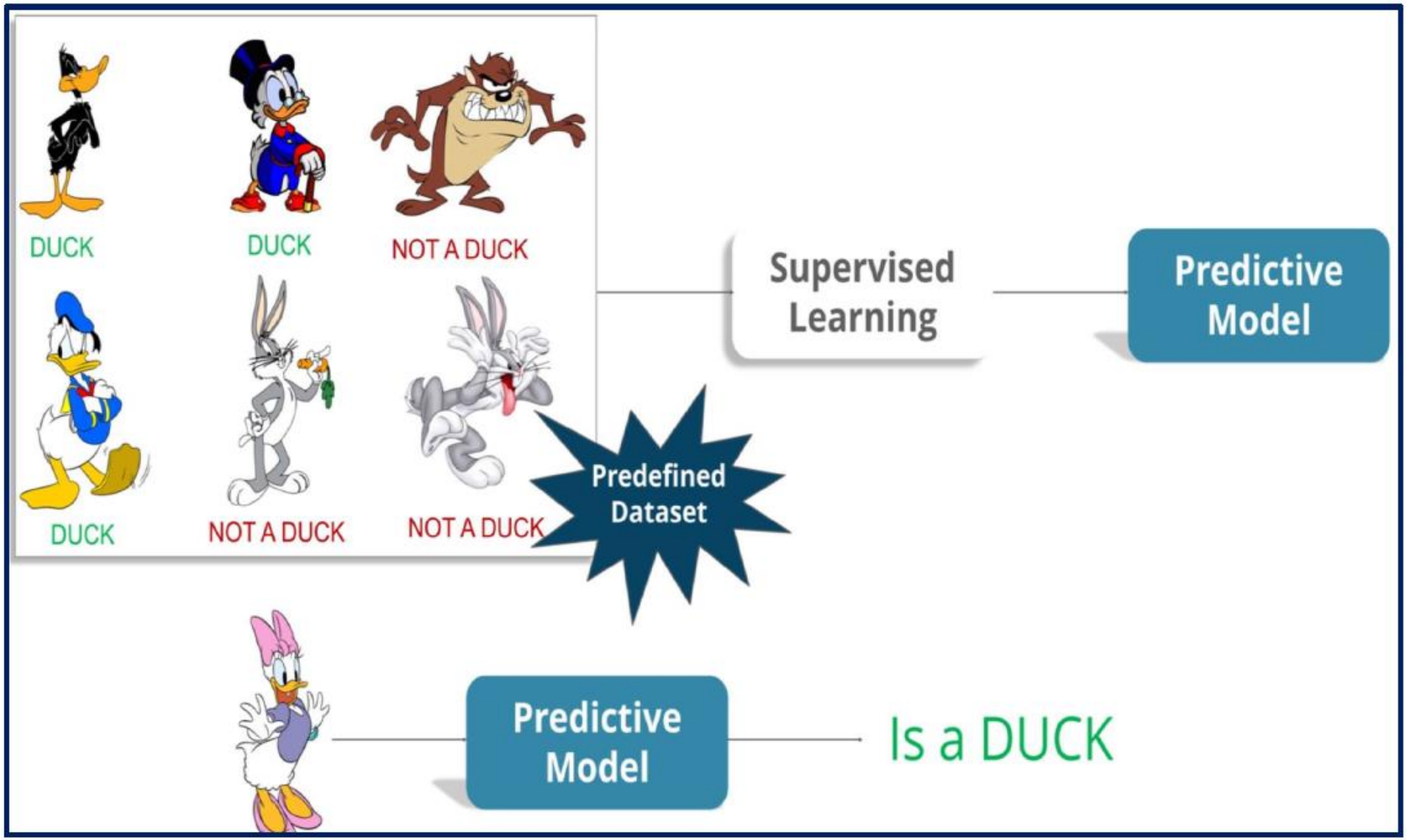
Iris Flower Classification

The iris dataset contains three classes of flowers, **Versicolor, Setosa, Virginica,** and each class contains 4 features, 'Sepal length', 'Sepal width', 'Petal length', 'Petal width'. The aim of the iris flower classification is to predict flowers based on their specific features.

1. Load the data
2. Analyze and visualize the dataset
3. Model training.
4. Model Evaluation.
5. Testing the model.

	Sepal length	Sepal width	Petal length	Petal width	Class_labels
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

E.g. Supervised Learning



Unsupervised Learning

Unsupervised learning involves finding hidden patterns or intrinsic structures in input data that is not labeled. The model tries to learn the structure or distribution of the data without any supervision.

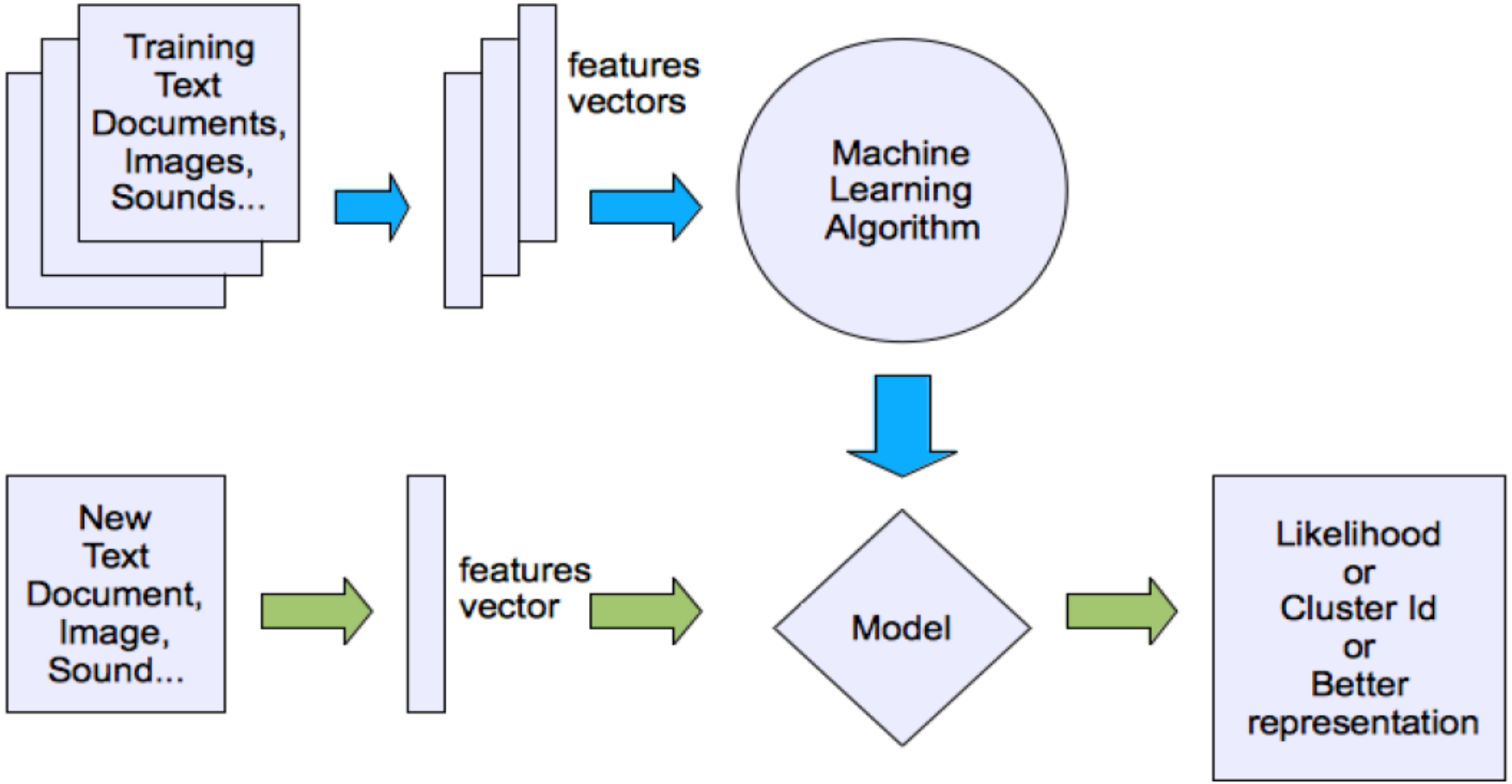
•Clustering:

- **Objective**: Group similar data points together based on some notion of similarity.
- **Example**: Customer segmentation, grouping similar documents.
- **Algorithms**: k-Means, Hierarchical Clustering, DBSCAN, Gaussian Mixture Models (GMM).

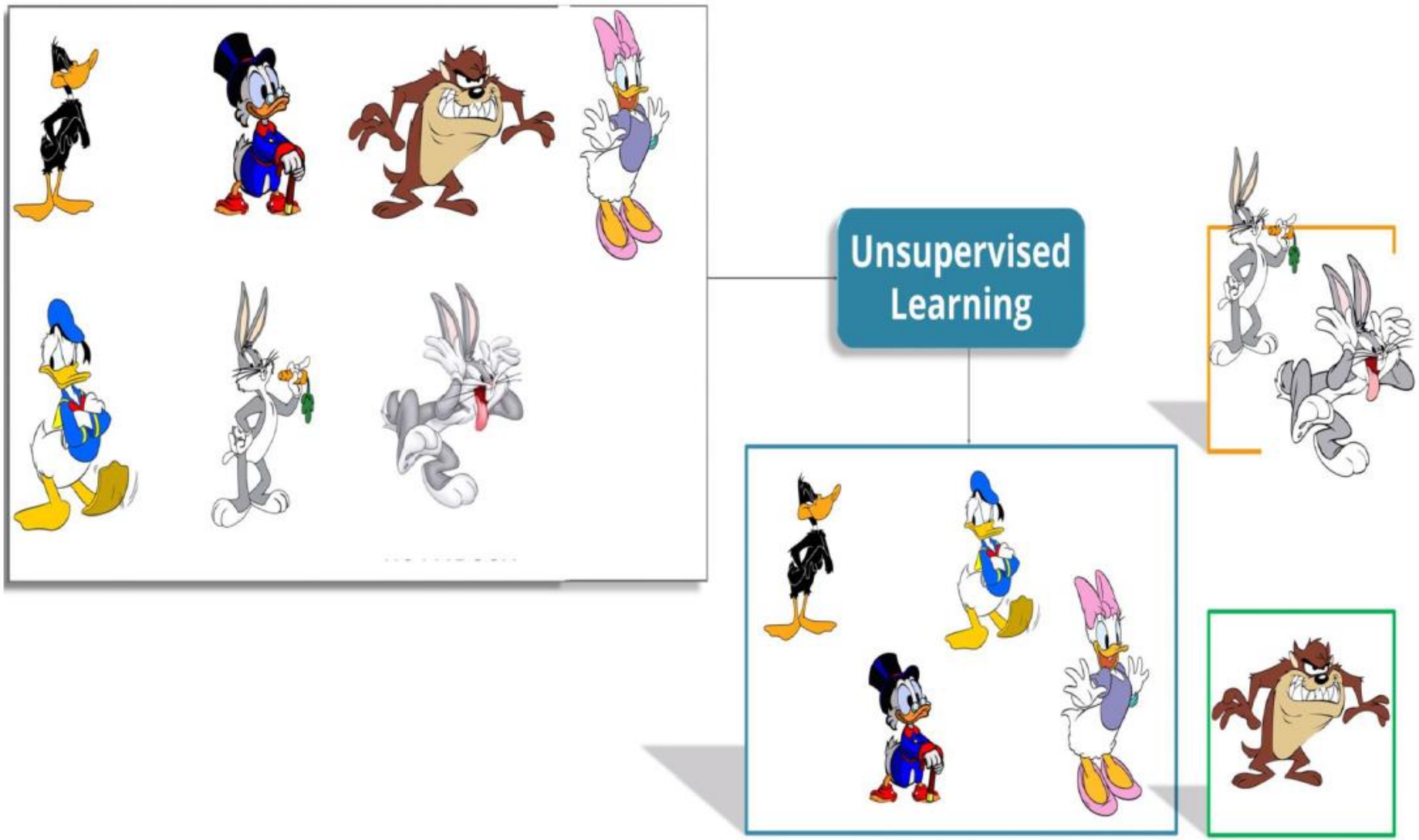
Anomaly Detection:

- **Objective**: Identify rare items, events, or observations that do not conform to the general distribution of the data.
- **Example**: Fraud detection, network security.
- **Algorithms**: Isolation Forest, One-Class SVM, Auto encoders.

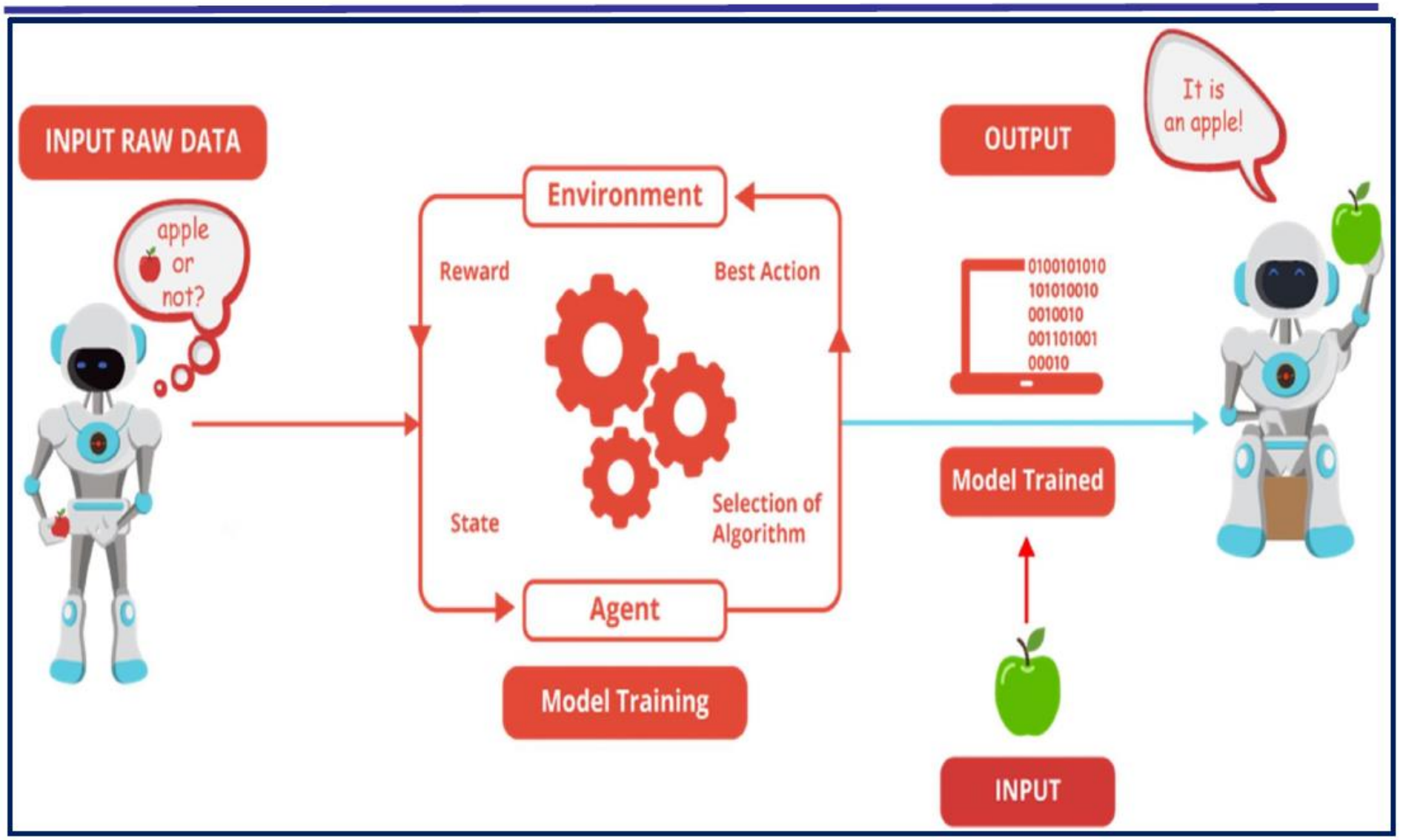
- Unsupervised learning



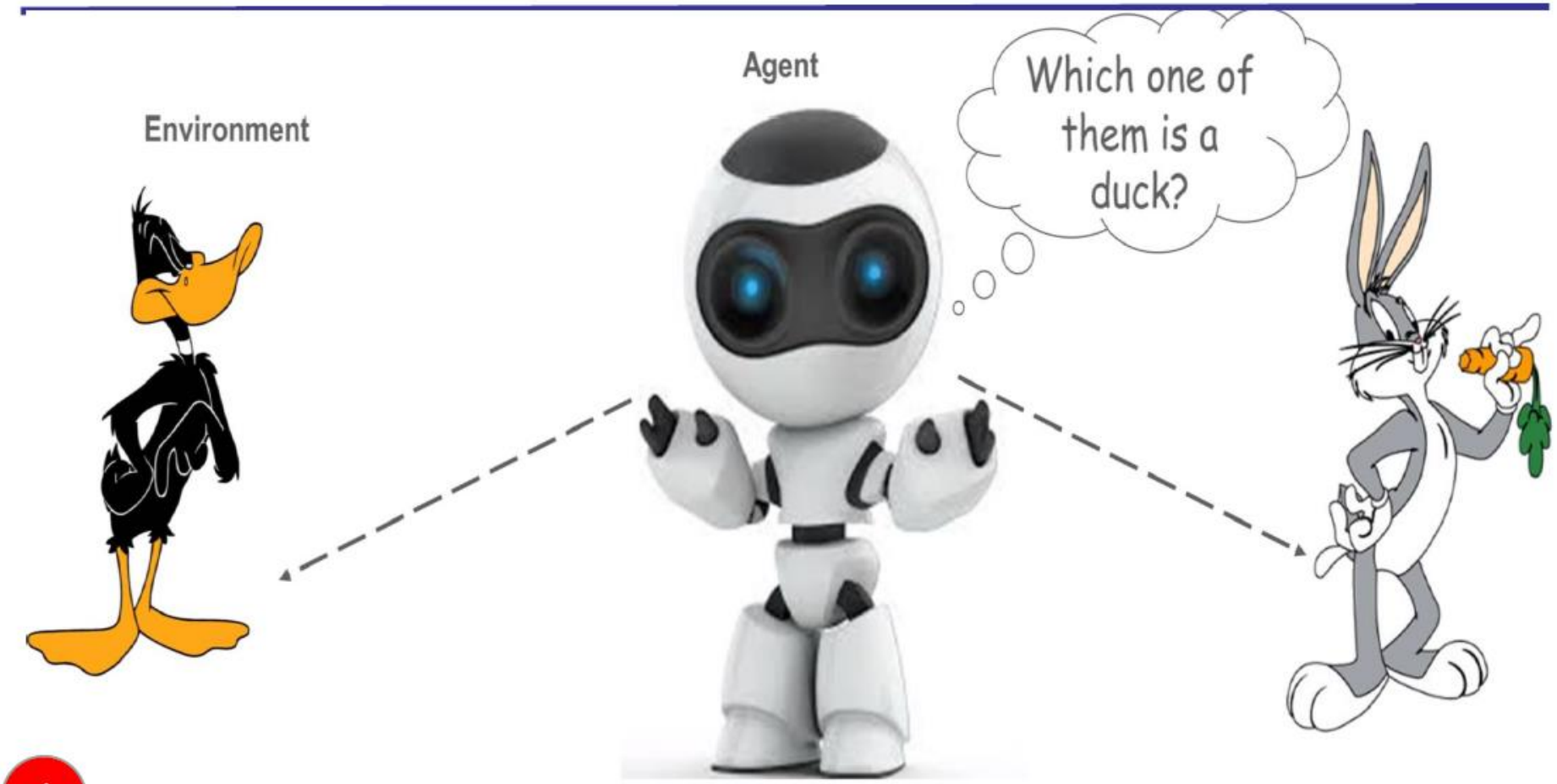
E.g. Unsupervised Learning



Reinforcement Learning

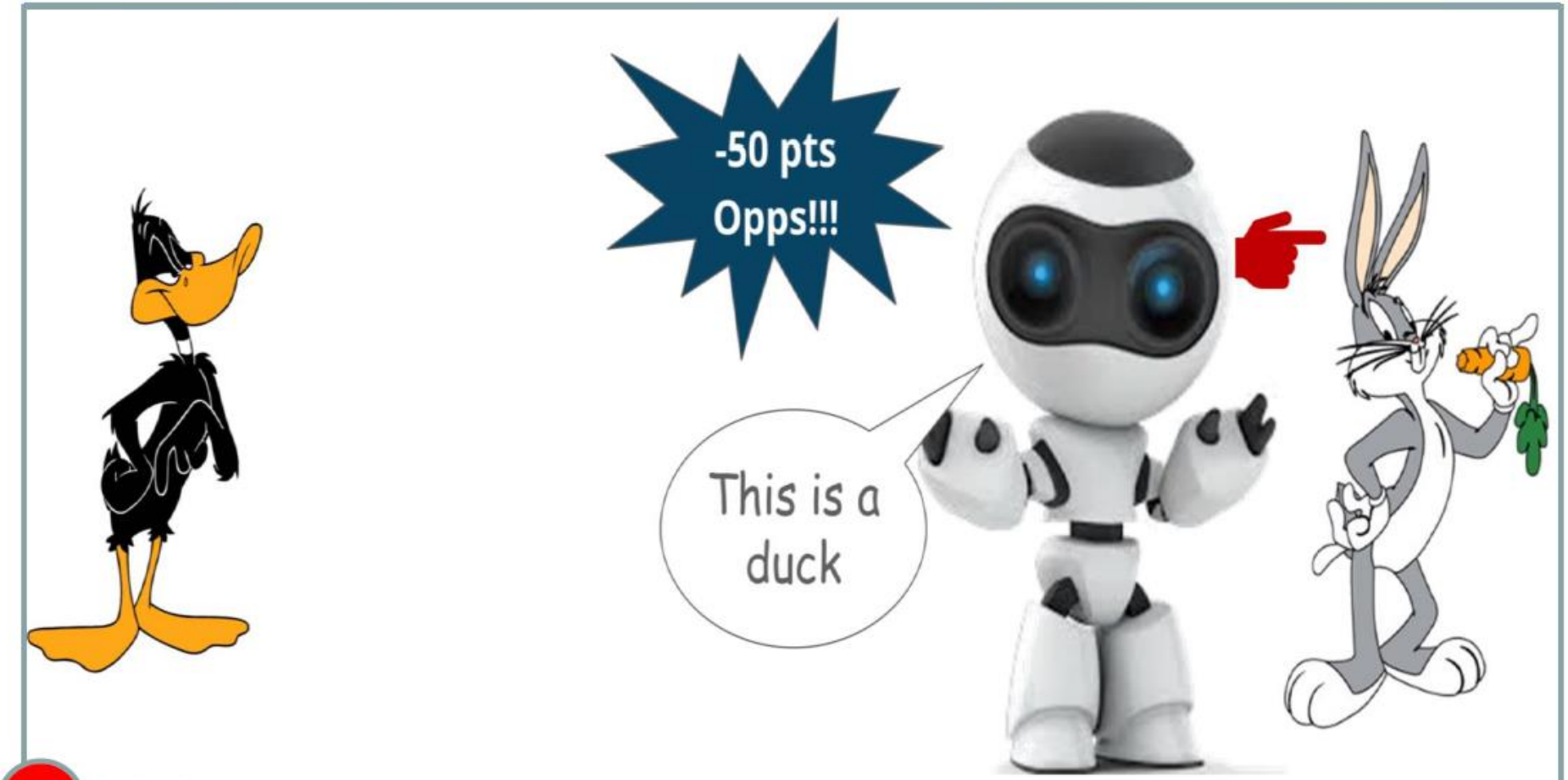


Reinforcement Learning



1. Observe
2. Select Action Using Policy

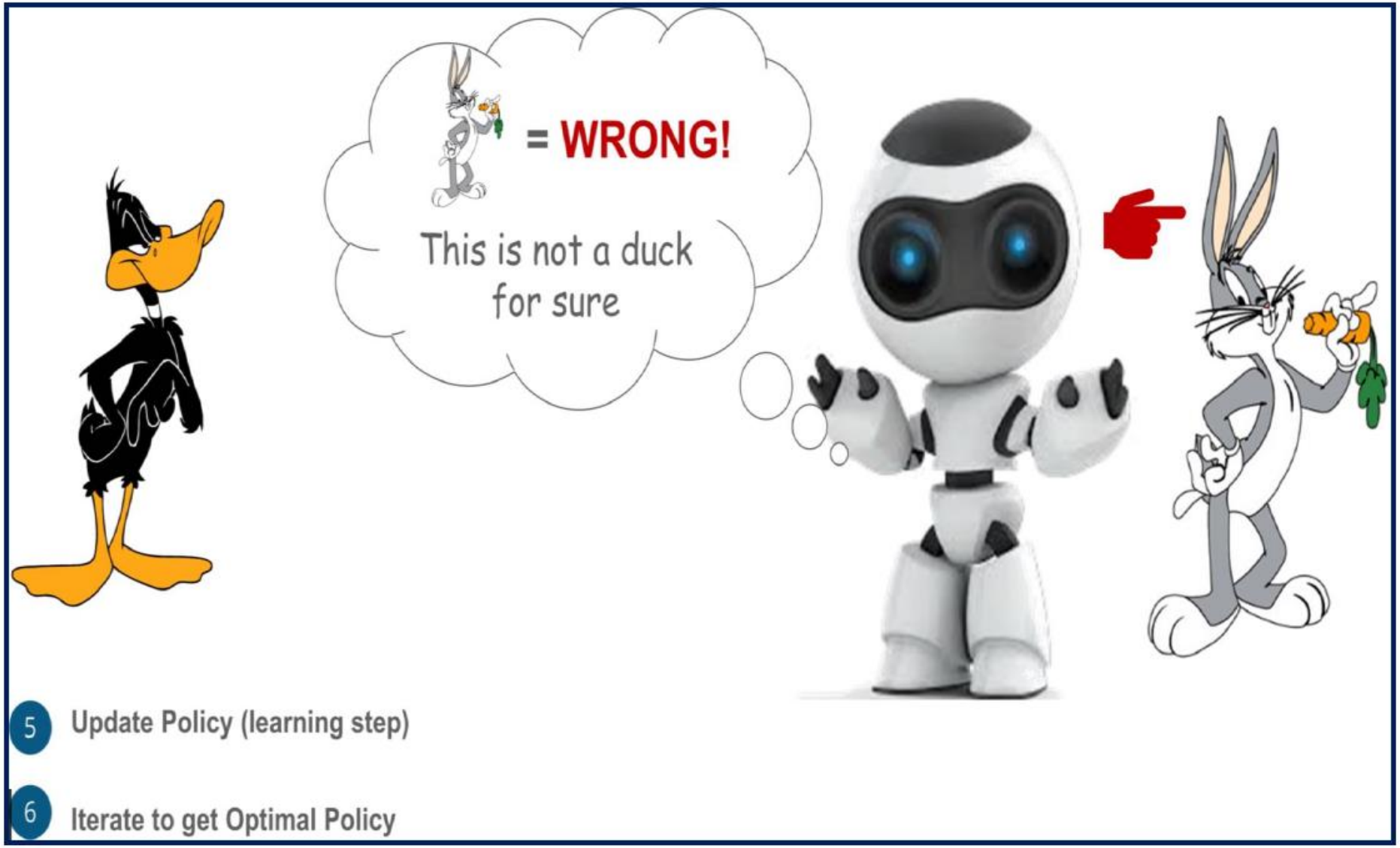
Reinforcement Learning



3 Action!

4 Get Reward or Penalty

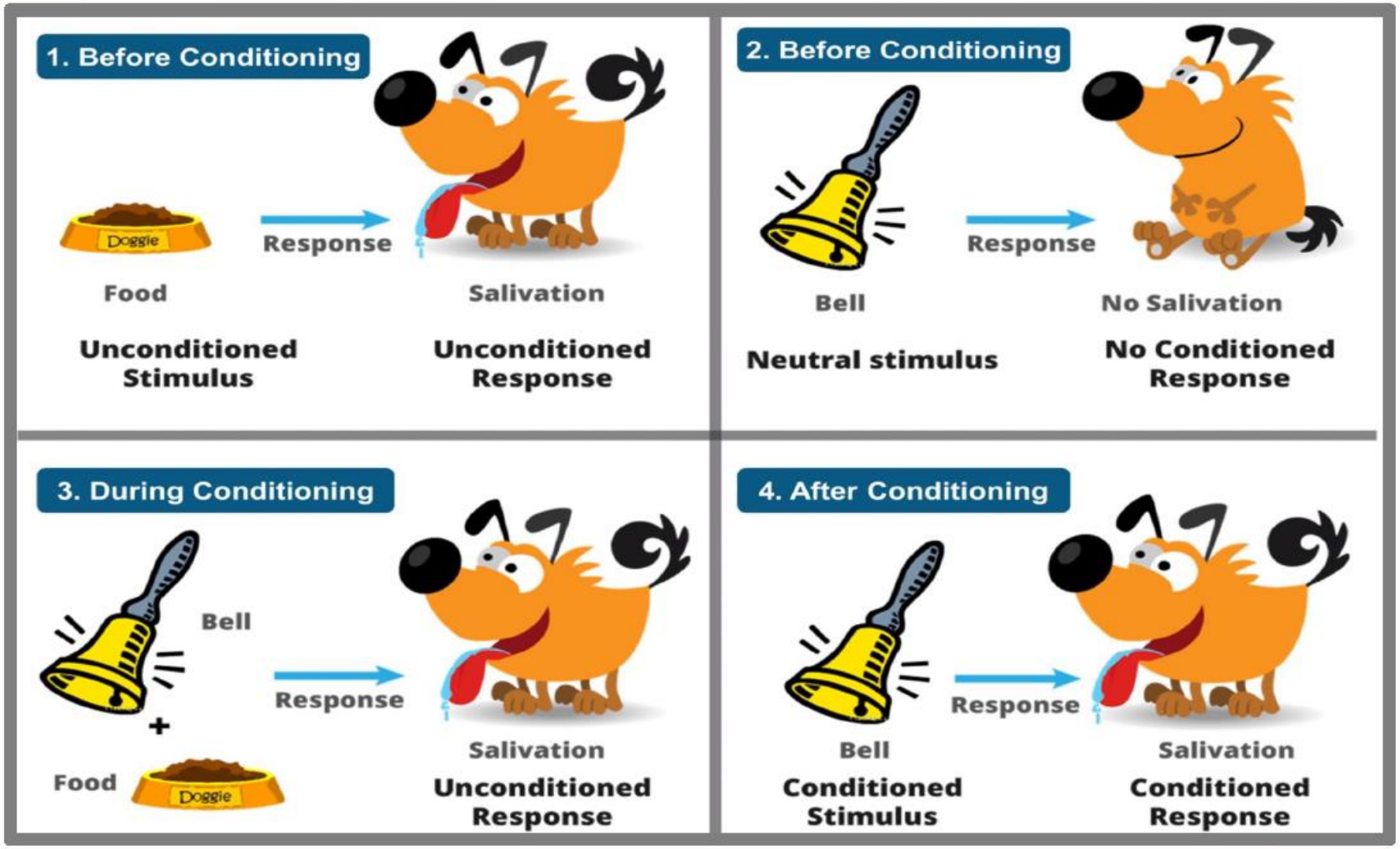
Reinforcement Learning



5 Update Policy (learning step)

6 Iterate to get Optimal Policy

E.g. Reinforcement Learning



Data Representation

- **Information systems:**
  - ❖It represents knowledge from RAW data, which is used for decision making.
- **Data warehousing**
  - ❖It provide integrated, consistent and cleaned data to machine learning algorithms.
- **Data Table:**
  - ❖It is used to represent information.

DATA TABLE

- Each row represents a measurements/ observations and each column gives the value of an attribute of the information system for all measurements/ observations.
- Different terms are used to call **'Rows'** information such as **"Instances, examples, samples, measurements, observations, records, patterns, objects, cases, events"**
- Similarly, the '**Column**' information is used to call **"attributes and features"**.

E.G. DATA TABLE

- Consider a patient information in the data table.
- **Features and attributes:** Headache, Muscle-Pain, Temperature. These attributes represented in linguistic form.

Patient	Headache	Muscle Pain	Temperature	Flu
1	NO	YES	HIGH	**YES**
2	YES	YES	HIGH	**YES**
3	YES	YES	VERY HIGH	**YES**
4	NO	YES	NORMAL	**NO**
5	YES	NO	HIGH	**NO**
6	NO	YES	VERY HIGH	**YES**

- **Decision Attribute:** one distinguished attributes that represent knowledge and information system of this kind called decision system.
- E.g. 'FLU' is decision attribute
- {Flu: Yes}, {Flu; No}.
- **Flu** is a decision attribute with respect to condition attributes: *headache, muscle-pain, temperature.*

- A data file represents inputs as N instances: $S^{(1)}, S^{(2)}, S^{(3)}, \ldots\ldots\ldots S^{(N)}$.
- Each individual instances $S^{(i)}; i = 1, 2, \ldots\ldots N$ that provides the input to the machine learning tools is characterized by its predefined values for a set of features/attributes $x_1, x_2, \ x_3, \ldots\ldots\ldots\ x_n$ *or* $x_j; j = 1,2,3, \ldots\ldots n$

x_j $S^{(i)}$	x_1	x_2	x_3	x_3	.....	x_n	Decision y
$S^{(1)}$							
$S^{(2)}$							
$S^{(3)}$							
$S^{(4)}$							
. .							
$S^{(N)}$							

Training experience is available in the form of N examples: $S^{(i)} \in S; i = 1, 2, 3 \ldots N$. Where S is a set of possible instances, which come from real world.

DATA REPRESENTATION

- An instance can be represented for n attribute/features: $x_j; j = 1, 2, 3, \ldots\ldots\ldots n$.
- These features can be visualize as n numerical features as a point in n -dimensional state space $\Re^n$.
- $\boldsymbol{x} = [x_1\ x_2\ x_3\ x_4 \ldots\ldots x_n]^{\boldsymbol{T}} \in \Re^n$. The set X is a finite set of feature vector $x^{(i)}$ for all possible instance.
- Also visualized as X region in the state space $\Re^n$ to which instance belongs, i.e. $X \subset \Re^n$

DATA REPRESENTATION

- Here, $x^{(i)}$ is a representation of $s^{(i)}$, X is the *representation space*.
- The pair of (S, X) constitutes the information system. Where S is non-empty set of instances and X is non-empty features.
- Here, index i represents instances and j represents features.
  - ❖ $\{s^{(i)}; i = 1, 2, 3, \ldots . N\} \in S$
  - ❖ $\{x^{(i)}; i = 1, 2, 3, \ldots . N\} \in X$ *(set of features)*
  - ❖ $\left\{x_j^{(i)}; j = 1, 2, 3, \ldots . N\right\} = x^{(i)}$
  - ❖ **Features $x_j; j = 1, 2, \ldots . n$, may be viewed as state variables and feature vector x as a state vector in n-dimensional space.**

Data Diversity

Diversity of data -refers to the variety or heterogeneity of the data used to train and evaluate models.

Importance of Data Diversity

1.Improves Generalization:

A diverse dataset helps the model learn patterns that generalize well to new, unseen data. If the training data is diverse, the model is less likely to **overfit** to specific patterns that don't hold across all data points.

2.Avoids Bias:

Lack of diversity can lead to **biased models**.

For example, if a facial recognition system is trained only on images of people with a certain skin tone, it may perform poorly on images of people with different skin tones.

3.Captures Edge Cases:

Diverse data includes edge cases and **rare situations**, which are important for building robust models that perform well across all possible inputs, not just the most common ones.

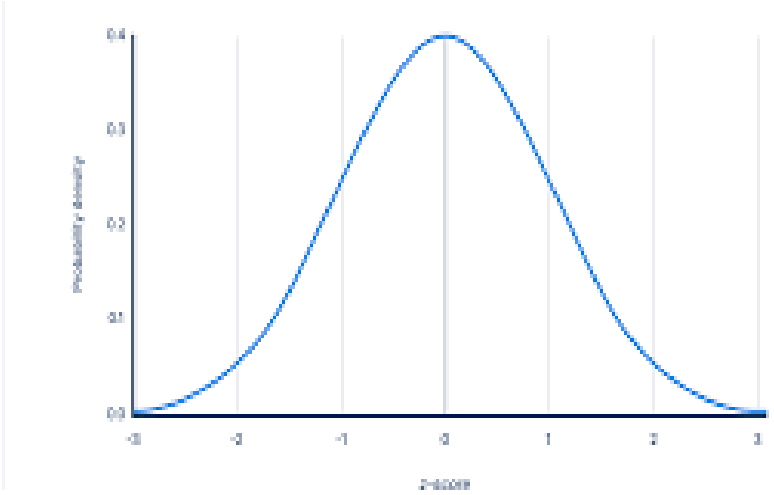
4.Reflects Real-World Complexity:

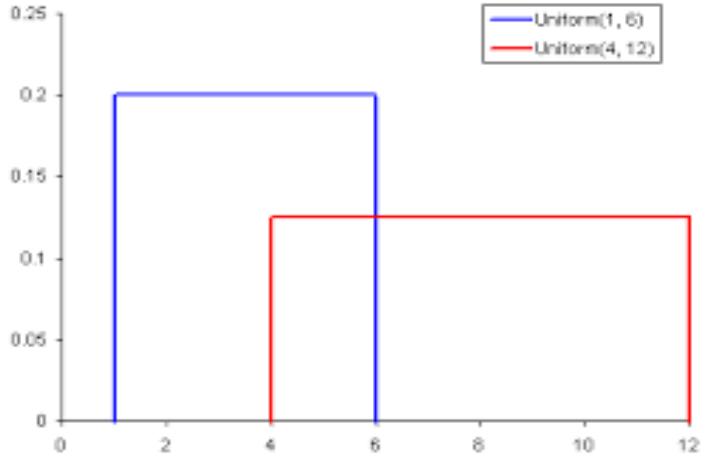
Real-world problems often involve complex and varied scenarios. Diverse data ensures that the model can handle this complexity and doesn't just perform well on a narrow subset of possible situations.

Data Distribution

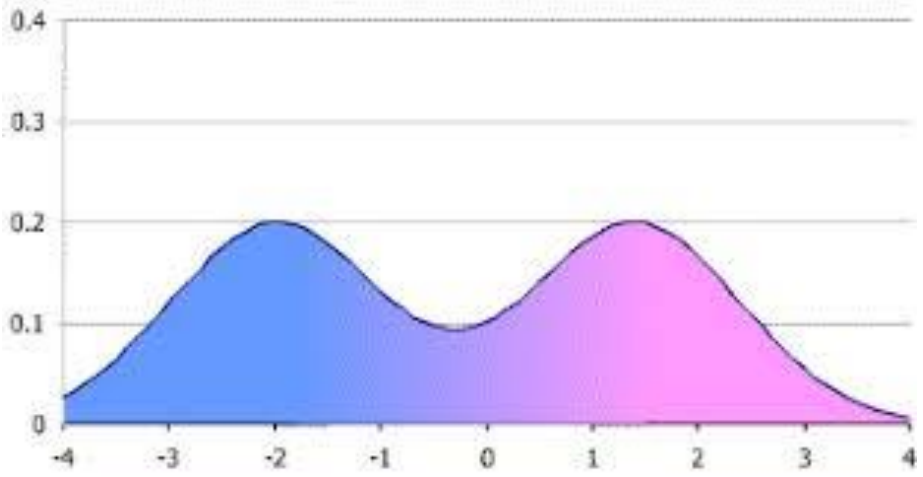
Data distribution refers to how the values of data points (features and labels) are spread or distributed within a dataset. It includes the range, frequency, and patterns of data points across different variables.

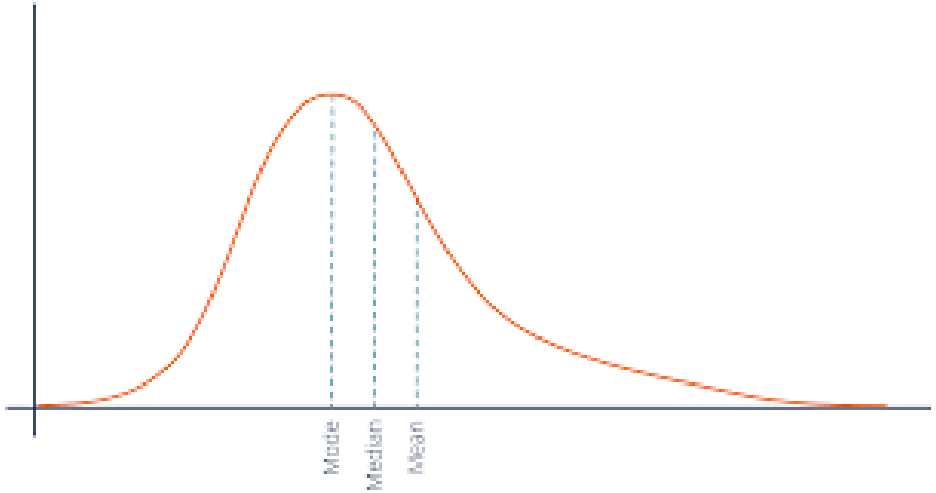
- **Uniform Distribution**: Data points are evenly spread across the range of possible values.
- **Normal Distribution**: Data points are symmetrically distributed around a central mean, forming a bell curve.
- **Skewed Distribution**: Data points are asymmetrically distributed, with a longer tail on one side of the mean.
- **Multimodal Distribution**: Data points are distributed in a way that there are multiple peaks or modes, indicating the presence of different subgroups or clusters within the data.





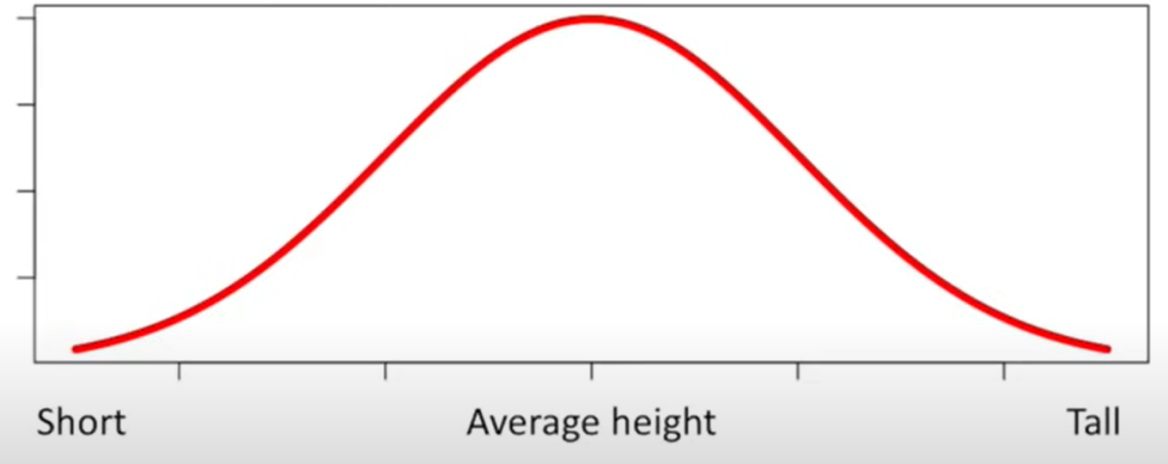
Guess the distribution ?





Normal Distribution: More common in real-life data due to natural variations, the central limit theorem, and the way multiple factors interact in complex systems.

A normal distribution, also known as a Gaussian distribution, is characterized by a symmetric, bell-shaped curve where most data points cluster around a central mean, with fewer points appearing as you move further from the mean.



Feature Space

Feature Space refers to the multidimensional space where each dimension corresponds to one feature (or attribute) of the data. It is the space in which machine learning algorithms operate to classify or predict data points.

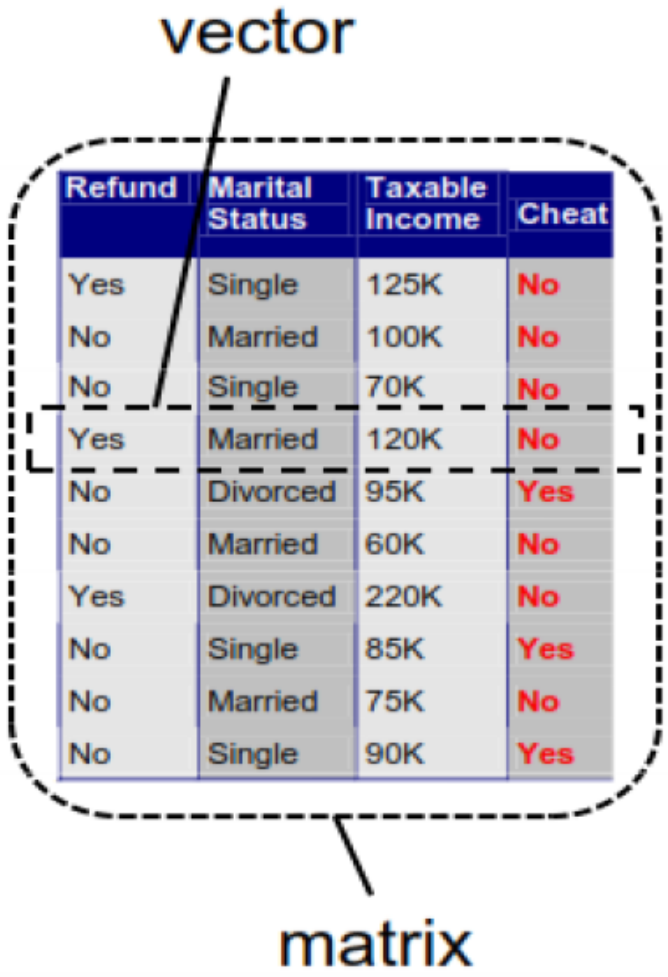
- **Feature Space**: The geometric space defined by the features of the data. Each data point in this space is represented by a vector of feature values. For instance, if you have a dataset with three features, the feature space is a three-dimensional space.

Representation:

- **Data Points**: Each point in the feature space represents a single data instance, characterized by its feature values. For example, in a 2D feature space, a point might be represented by coordinates (x1, x2) corresponding to two features.

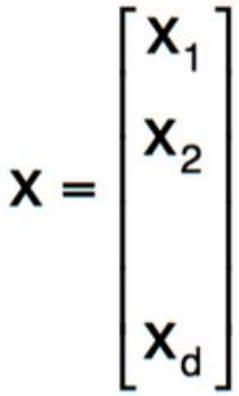
Why vectors and matrices?

- Most common form of data organization for machine learning is a 2D array, where
  - *rows* represent samples (records, items, datapoints)
  - *columns* represent attributes (features, variables)
- Natural to think of each sample as a *vector* of attributes, and whole array as a *matrix*

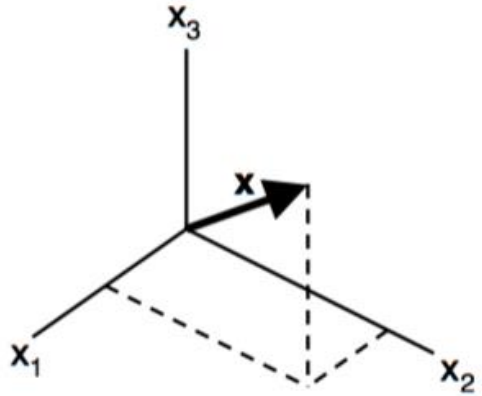


Dimensions:

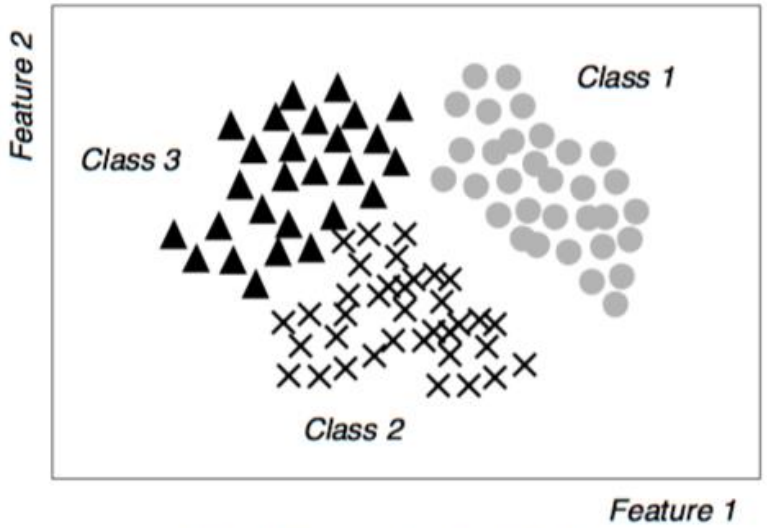
- **Dimensionality**: The number of dimensions in the feature space corresponds to the number of features. For instance:
  - **1D Feature Space**: A single feature (e.g., height).
  - **2D Feature Space**: Two features (e.g., height and weight).
  - **3D Feature Space**: Three features (e.g., height, weight, and age).
  - **Higher Dimensions**: More than three features create a higher-dimensional space that is harder to visualize but still conceptually similar.



Feature vector



Feature space (3D)



Scatter plot (2D)

Inductive Learning

Inductive learning is a key concept in machine learning where a model learns general rules from specific examples or data points. In this learning approach, the machine is provided with labeled training data, and from this, it attempts to infer a generalized function that can make predictions on unseen data

Key Idea of Inductive Learning

•Specific to General: The model starts with specific instances (training data) and generalizes from these examples to learn a broader pattern or rule.

•Learning from Observations: Inductive learning relies on learning patterns from observed data without assuming the data fits into a pre-defined theory or structure.

How It Works

1.Training Data
2. Generalization
3.Prediction (tasks)

Learner Discovers Rules by Observing Examples

- **Learner**: This refers to the machine learning algorithm or model.
- **Observing Examples**: The learner does not have predefined rules but instead discovers patterns or rules by examining examples (data points) provided in the training dataset.

The learner studies these examples and uses them to infer a relationship (or function) between input variables and the output. This process is known as **training** the model.

- Learner discovers rules by observing examples
- **Given** examples of a function *(X, F(X))*
- **Predict** function *F(X)* for new examples *X*
  - ❖Discrete *F(X)*: Classification
  - ❖Continuous *F(X)*: Regression
  - ❖*F(X)* = Probability(*X*): Probability estimation

EXAMPLE : Lets take an example of loan approval system: learner uses examples from historical data to learn the function F(X), then applies it to new data for making predictions. (income, credit score, and employment history.)

- **Classification**:
  - $F(X)$ could be a function that classifies whether a loan will be approved or rejected based on features such as income, credit score, and employment history.
  - The model outputs a **discrete value**: "Approved" or "Rejected."
- **Regression**:
  - In another case, $F(X)$ could predict the **amount** of loan that a customer is likely to get. Based on the same input features (income, credit score), the output will be a **continuous value** like $50,000 or $75,000.
- **Probability Estimation**:
  - The model could estimate the probability that a customer will default on a loan, giving a result like **80% chance** of defaulting based on the customer's income and credit score.

Hyper parameter vs Parameter / features

1. Parameters

Definition: Parameters are the internal variables of a model that are learned from the training data. These values are adjusted during the training process to minimize the model's loss function and improve performance.

Examples:

- **Weights** in a linear regression model.
- **Coefficients** in a logistic regression model.
- **Weights and biases** in a neural network.

2. Hyperparameters

Definition: Hyperparameters are the external settings or configurations of the model that are not learned from the training data but set before the training process begins. These values are used to control the learning process and the structure of the model.

Examples:

- **Learning rate** in gradient descent.
- **Number of trees** in a random forest.
- **Number of layers** and **neurons per layer** in a neural network.
- **Regularization strength** in logistic regression (like L1 or L2 regularization).
- **Kernel type** in a Support Vector Machine (SVM).

Classification is a process of categorizing a given set of data into classes, It can be performed on both structured or unstructured data. The process starts with predicting the class of given data points. The classes are often referred to as target, label or categories.

Essentially, the terms “classifier” and “model” are synonymous in certain contexts; however, **sometimes people refer to “classifier” as the learning algorithm that learns the model from the training data**.

Model : It is what you get once you have finished training your classifier, it's the resulting object of the training phase. You can see it as an "intelligent" black box to whom you feed and input sample and it gives you a label as an ouput.

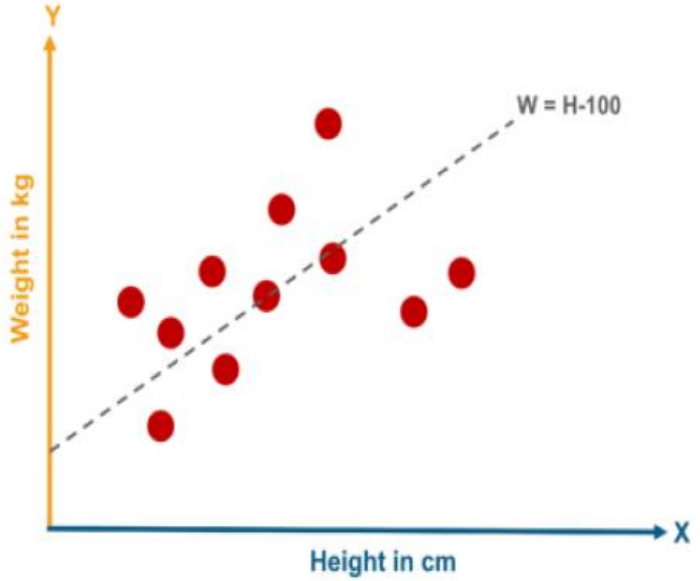
Types Of Classification

- **Binary Classification**
  - Classifies data into two distinct classes
  - Example: Spam vs. Not Spam.
- **Multi-Class Classification**
  - Classifies data into more than two classes.
  - Example: Handwritten digit recognition (0-9).
- **Multi-Label Classification**
  - Assigns multiple labels to a single instance.
  - Example: Image tagging where an image can have multiple tags like "cat," "outdoor," "daytime."

Which type of classification ?

Linear Regression

- A linear regression is one of the easiest statistical models in machine learning.
- It is used to show the linear relationship between a dependent variable and one or more independent variables.
- Relationship between one dependent variable (y) and explanatory variable (s).
- Regression analysis is a form of predictive modelling technique which investigates the relationship between a dependent and independent variable

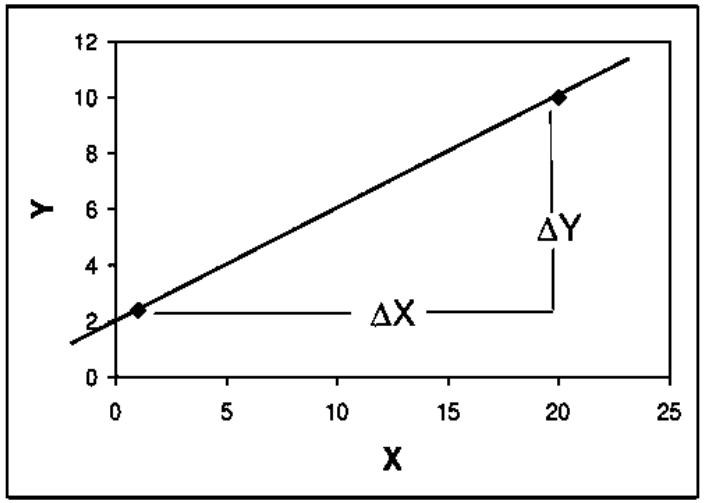


- **The Linear Model**

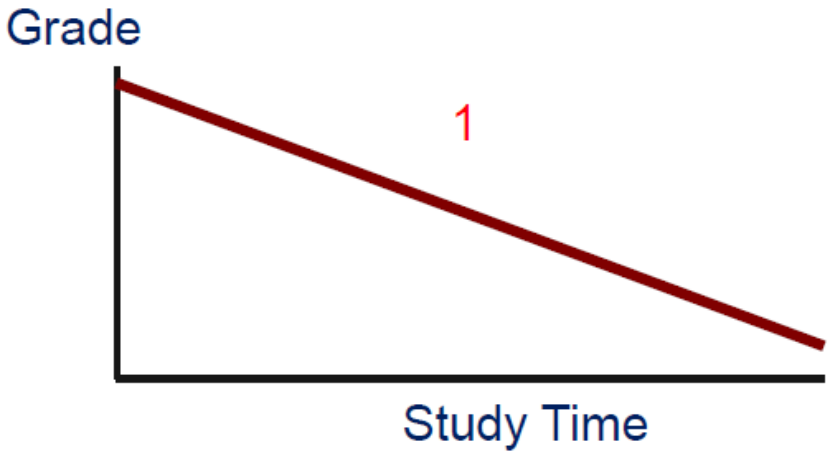
$$Y = mX + b$$

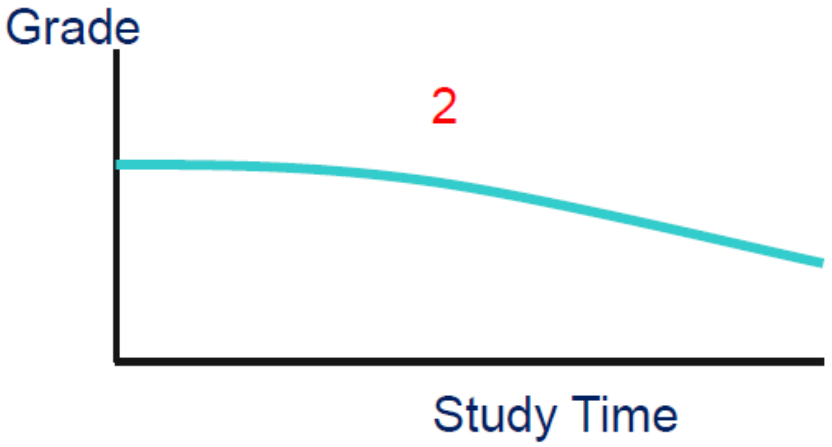
Y = Dependent variable
X = Independent variable
m = slope = $\Delta Y/\Delta X$
b = y-intercept (point where line crosses y-axis at x=0)

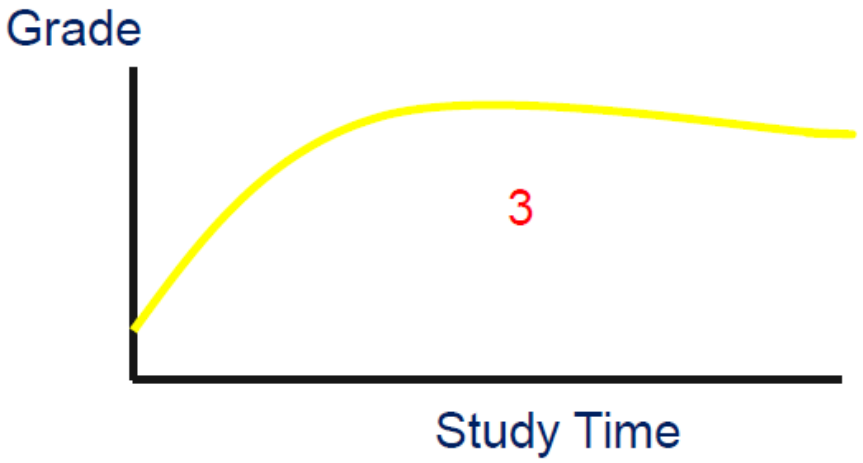
$X_1=1, Y_1=2.4$
$X_2=20, Y_2=10$

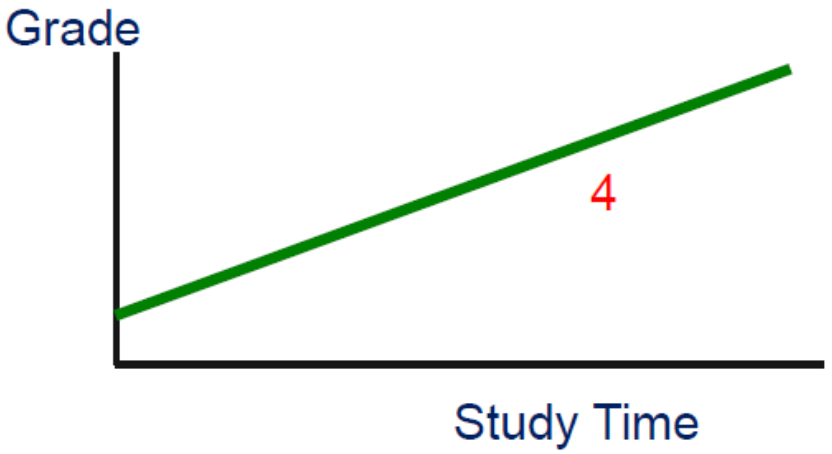


Thinking Challenge: Which is more logical?

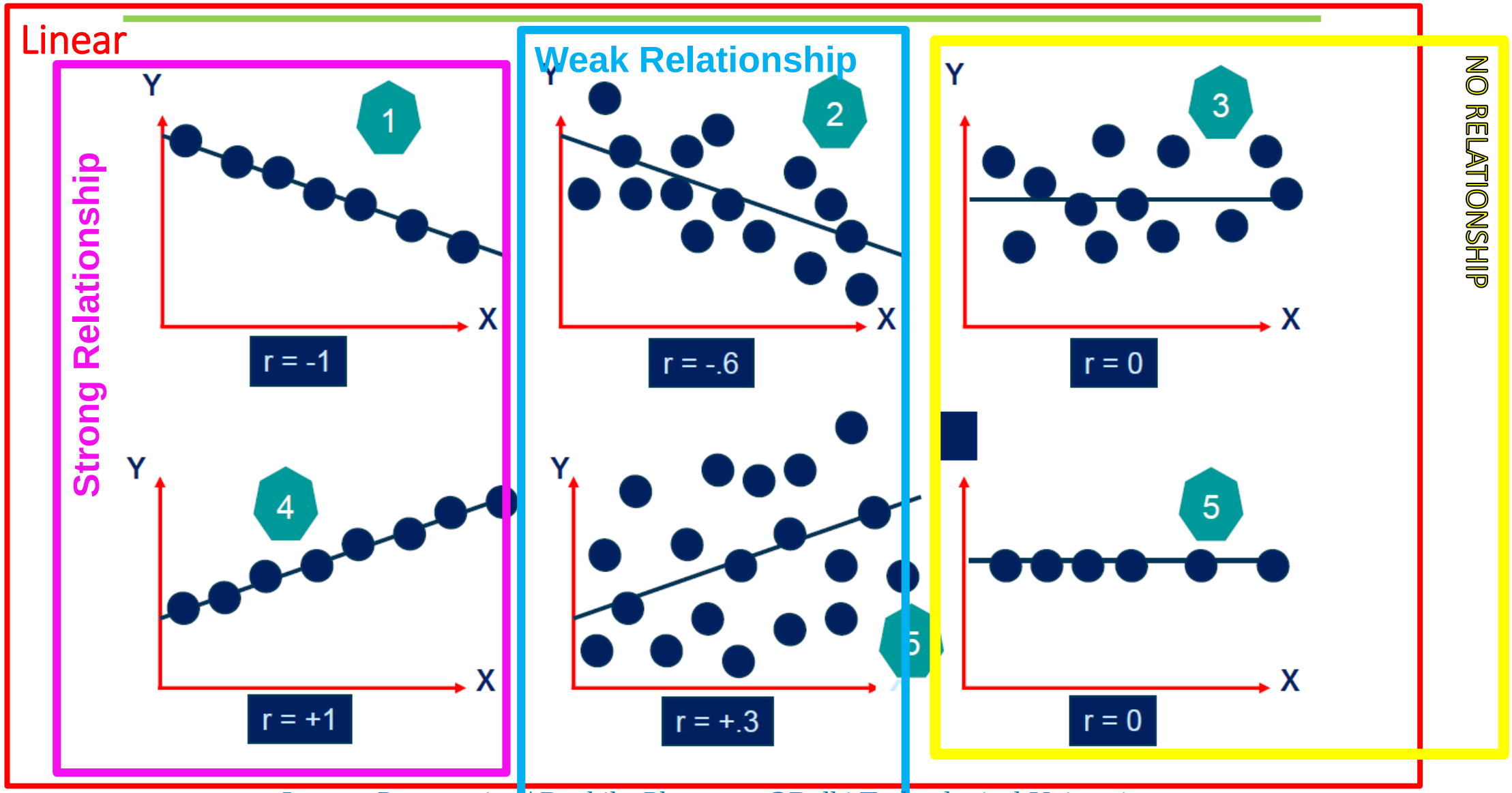






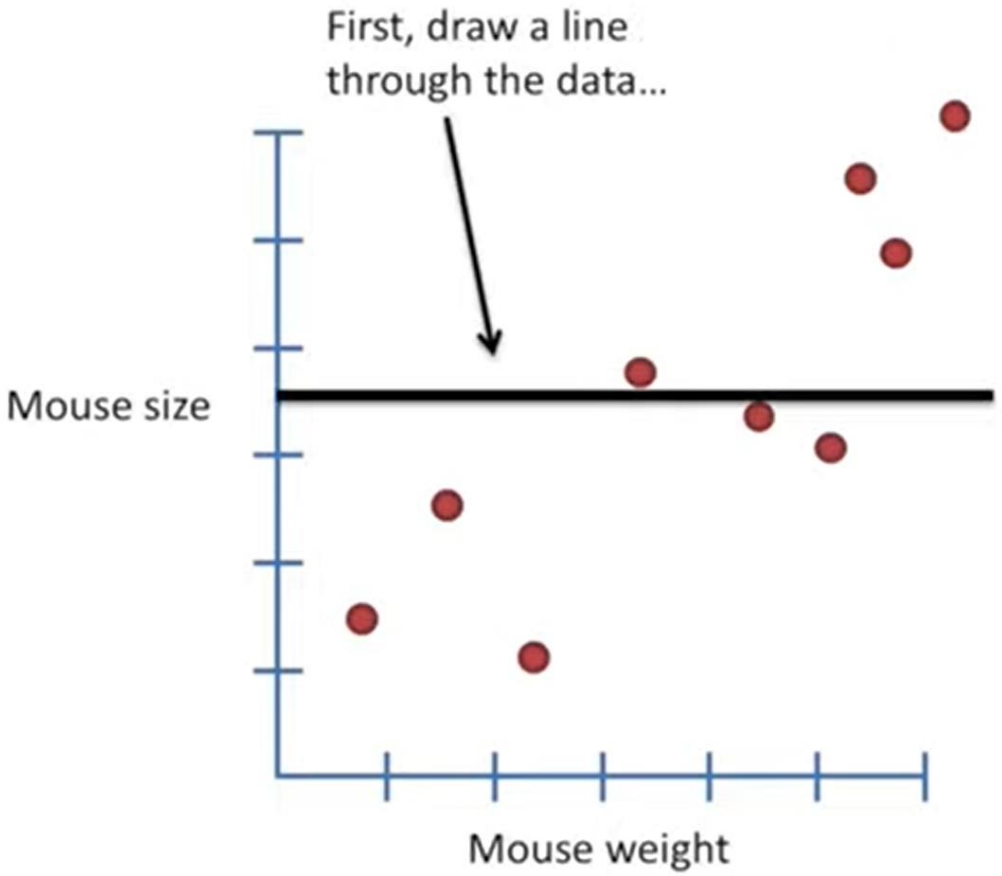


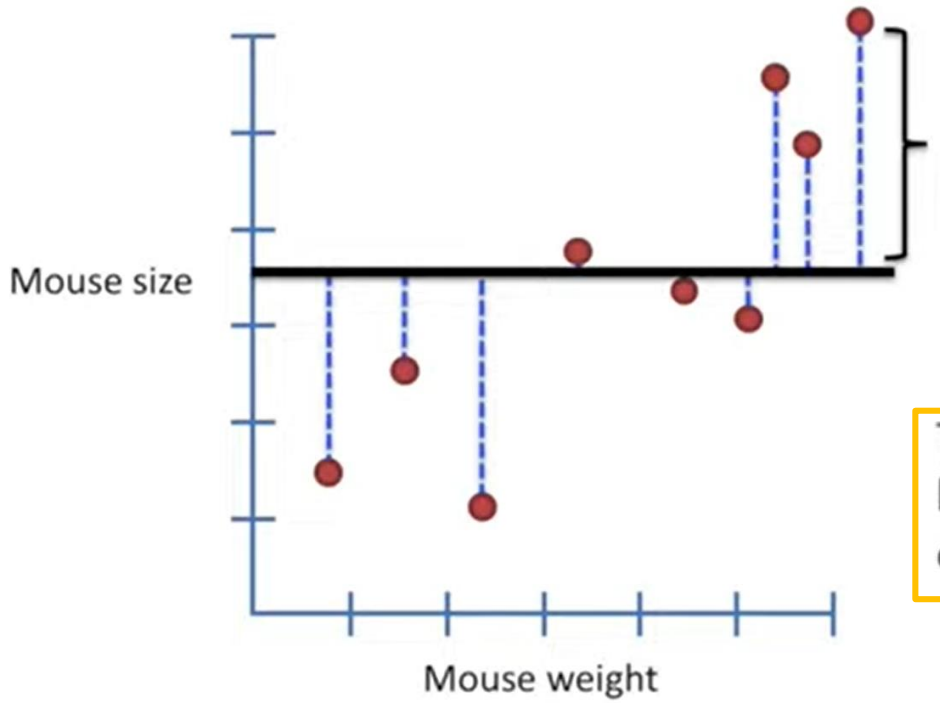
Scatter Plot of Data



What is the Least Squares Regression Method?

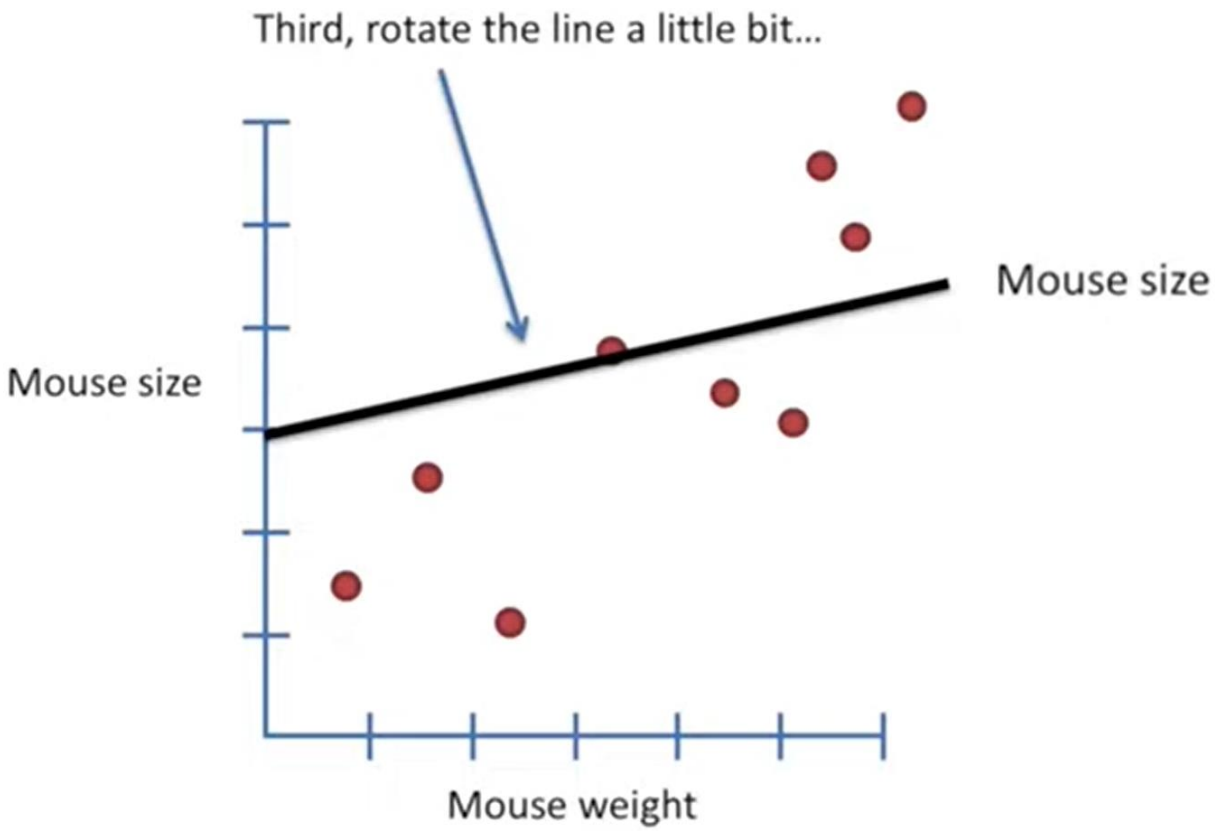
The least-squares regression method is a technique commonly used in Regression Analysis. It is a mathematical method used to find the best fit line that represents the relationship between an independent and dependent variable in such a way that the error is minimized.

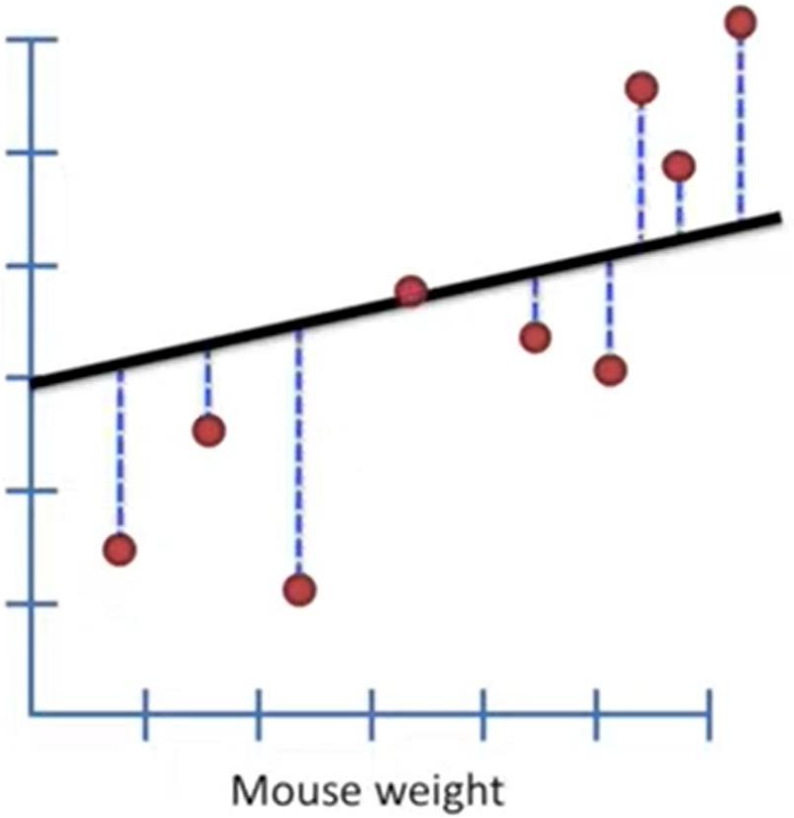




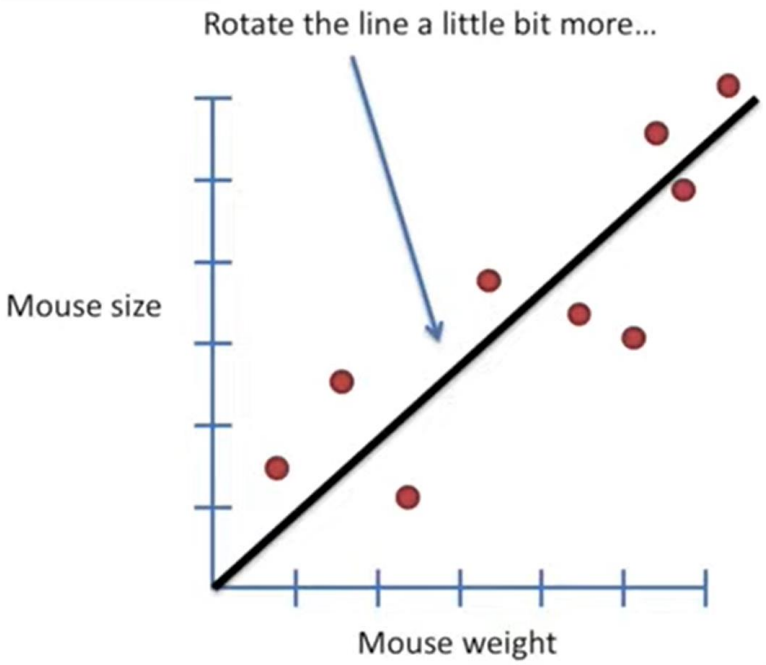
Second, measure the distance from the line to the data, square each distance, and then add them up.

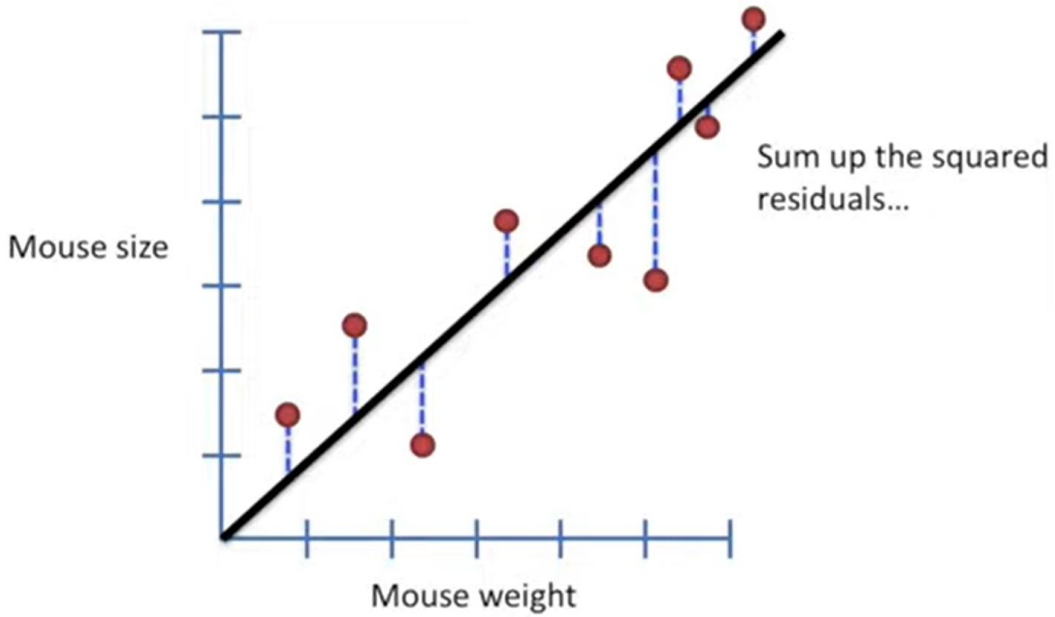
The distance from a line to a data point is called a "**residual**".

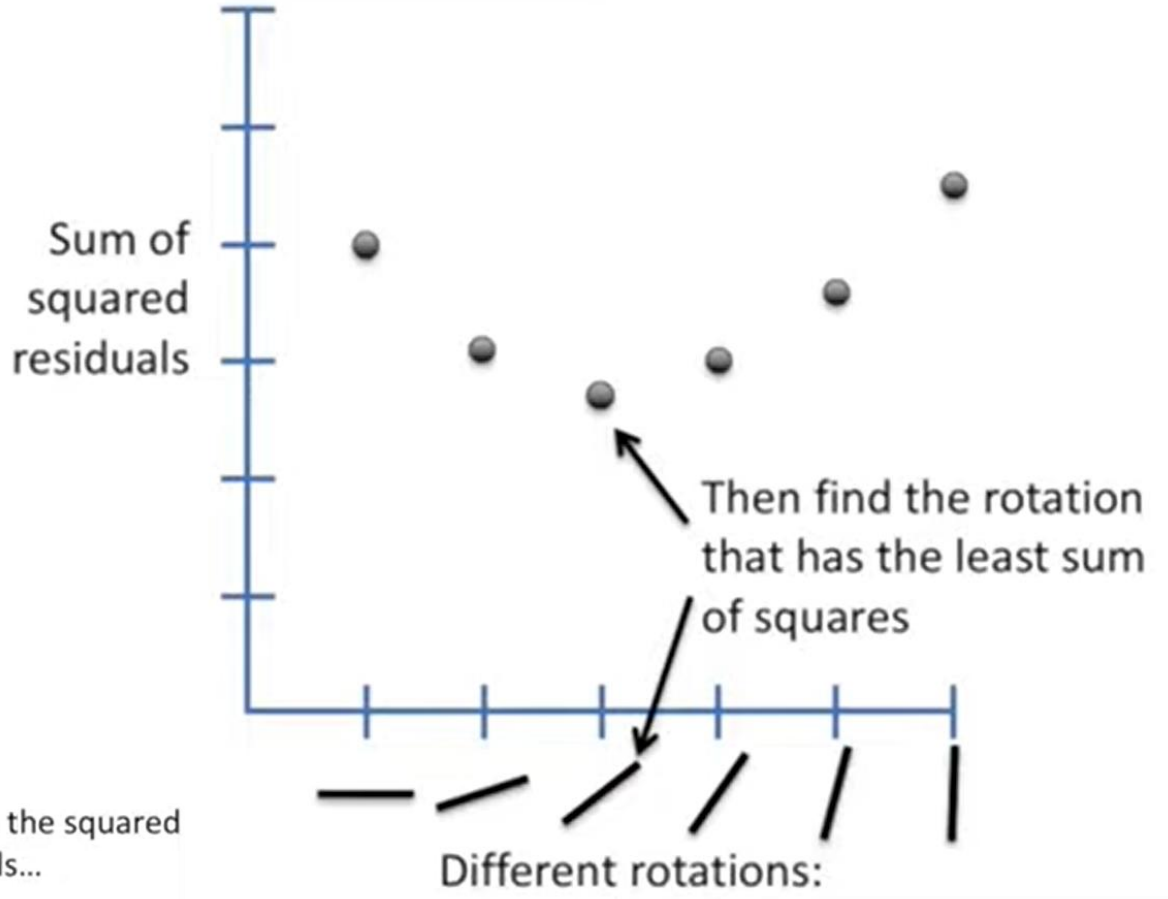


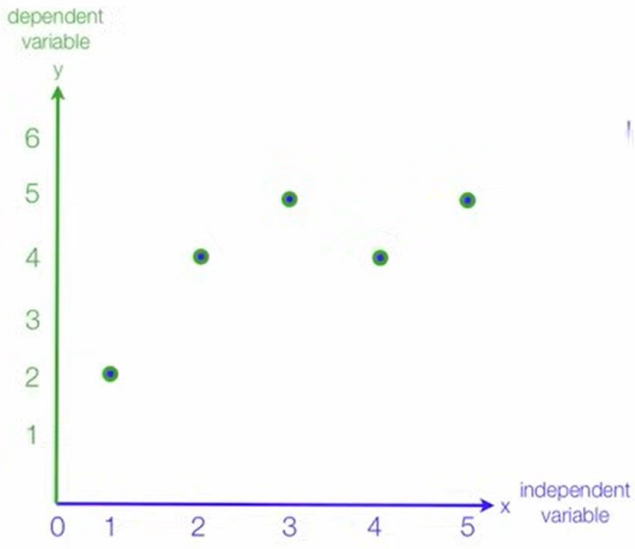


With the new line, measure the residuals, square them, and then sum up the squares.


Rotate the line a little bit more...
Mouse size
Mouse weight


Sum up the squared residuals...
Mouse size
Mouse weight

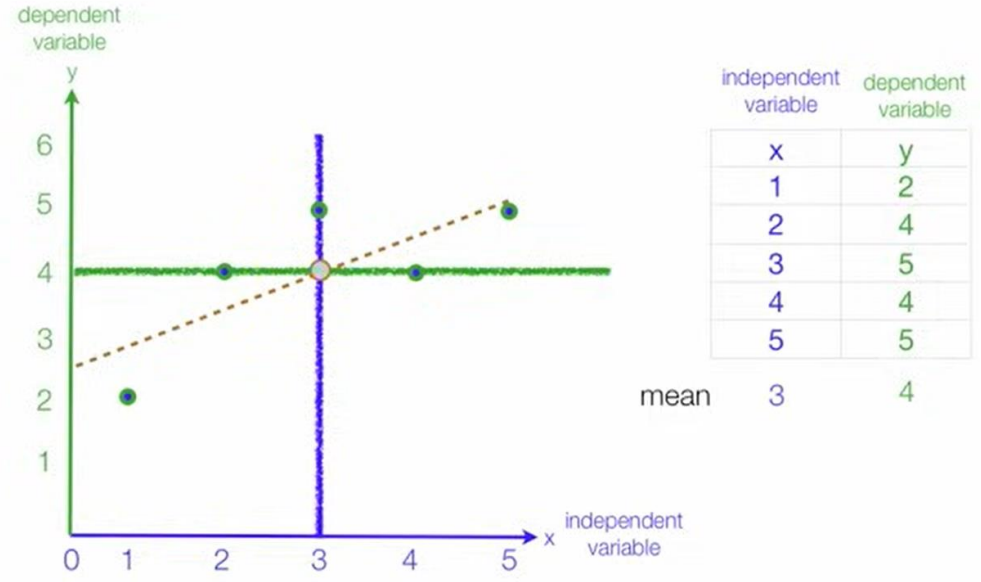

Sum of squared residuals
Then find the rotation that has the least sum of squares
Different rotations:



independent variable	dependent variable
x	y
1	2
2	4
3	5
4	4
5	5

The reason the least squares regression line passes through the mean coordinates $(\bar{x}, \bar{y})$ is because this point represents the center of the data, and passing through this point ensures that the regression line is optimally positioned to minimize the sum of squared residuals (errors). It's a balance point that ensures the errors are evenly distributed around the line. If the line didn't pass through the mean point, it would not be the best fit for the data.

$$\hat{y} = b_0 + b_1 x$$



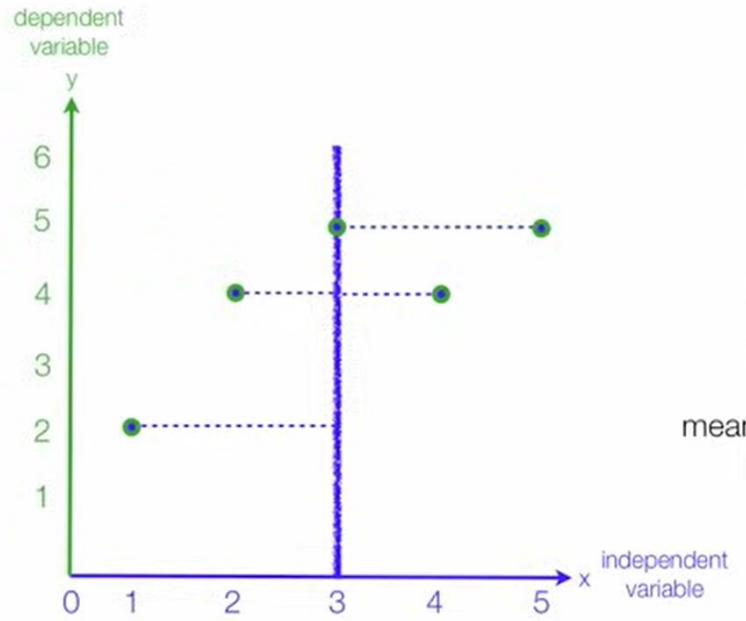
independent variable	dependent variable
x	y
1	2
2	4
3	5
4	4
5	5
mean	3	4

$$b_1 = \frac{\Sigma (x - \bar{x})(y - \bar{y})}{\Sigma (x - \bar{x})^2}$$

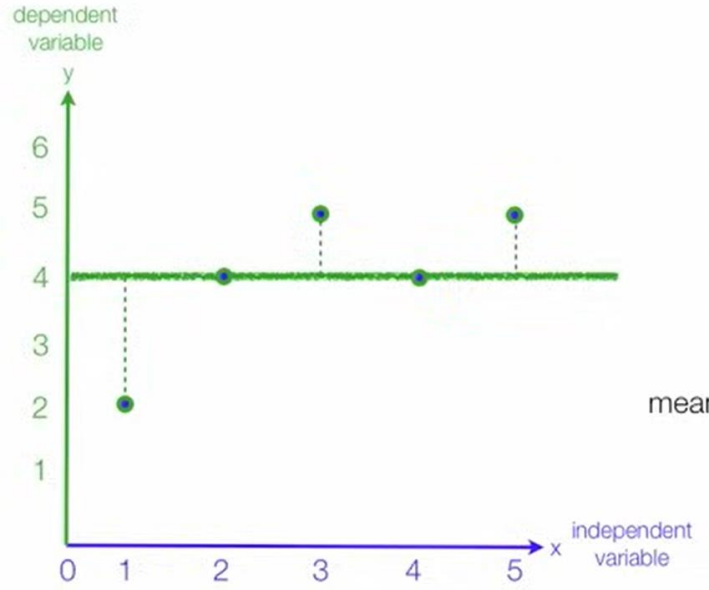
$$\beta_0 = \bar{y} - \beta_1 \cdot \bar{x}$$

Where:

- β_0 is the **intercept**,
- β_1 is the **slope**,
- $\bar{y}$ is the **mean** of the dependent variable y,
- $\bar{x}$ is the **mean** of the independent variable x.



independent variable	dependent variable
x	y
1	2
2	4
3	5
4	4
5	5
mean	3	4



independent variable	dependent variable
x	y
1	2
2	4
3	5
4	4
5	5
mean	3	4

x	y	$x - \bar{x}$	$y - \bar{y}$
1	2	-2	-2
2	4	-1	0
3	5	0	1
4	4	1	0
5	5	2	1
mean	3	4	

x	y	$x - \bar{x}$	$y - \bar{y}$	$(x - \bar{x})^2$	$(x - \bar{x})(y - \bar{y})$
1	2	-2	-2	4	4
2	4	-1	0	1	0
3	5	0	1	0	0
4	4	1	0	1	0
5	5	2	1	4	2
mean	3	4		10	6

$$b_1 = \frac{6}{10} = .6 = \frac{\Sigma(x - \bar{x})(y - \bar{y})}{\Sigma(x - \bar{x})^2}$$

$b_0 = 2.2$

$b_1 = .6$

$\hat{y} = 2.2 + .6x$

Simple Linear Regression

Given the observations (x_1, y_1), (x_2, y_2), $\cdots$, (x_n, y_n), we can write the regression line as

$$\hat{y} = \beta_0 + \beta_1 x.$$

We can estimate β_0 and β_1 as

$$\hat{\beta}_1 = \frac{s_{xy}}{s_{xx}},$$
$$\hat{\beta}_0 = \overline{y} - \hat{\beta}_1 \overline{x},$$

where

$$s_{xx} = \sum_{i=1}^{n} (x_i - \overline{x})^2,$$
$$s_{xy} = \sum_{i=1}^{n} (x_i - \overline{x})(y_i - \overline{y}).$$

For each x_i, the **fitted value** $\hat{y}_i$ is obtained by

$$\boxed{\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i.}$$

The quantities

$$e_i = y_i - \hat{y}_i$$

are called the **residuals**.

Another formula

$$m = \frac{n\sum xy - (\Sigma x)(\Sigma y)}{n\Sigma x^2 - (\Sigma x)^2}$$

m – slope of the line

n – total number of data points

x – Independent variable

y – Dependent variable

$$b = \frac{\left(\sum y \cdot \sum x^2\right) - \left(\sum x \cdot \sum xy\right)}{n \cdot \sum x^2 - \left(\sum x\right)^2}$$

Where:

- $\sum y$: The sum of the y-values (dependent variable),
- $\sum x$: The sum of the x-values (independent variable),
- $\sum x^2$: The sum of the squared x-values,
- $\sum xy$: The sum of the product of x and y,
- n: The number of data points.

Interpretation of Coefficients

- $\beta_1 > 0 \Rightarrow$ Positive Association
- $\beta_1 < 0 \Rightarrow$ Negative Association
- $\beta_1 = 0 \Rightarrow$ No Association

- Slope ($\widehat{\beta_1}$)
  - Estimated Y Changes by $\widehat{\beta_1}$ for each 1 Unit Increase in X
    - If $\widehat{\beta_1} = 2$, then Y Is Expected to Increase by 2 for each 1 Unit Increase in X

- Y-Intercept (β_0)
  - Average Value of Y When $X = 0$
    - If $\beta_0 = 4$, then Average Y is expected to be 4 When X is 0

Practice Question

X	Y
2	3
4	7
6	5
8	10

$$\hat{y} = b_0 + b_1 x$$

$$b_1 = \frac{\Sigma(x - \bar{x})(y - \bar{y})}{\Sigma(x - \bar{x})^2}$$

$$\beta_0 = \bar{y} - \beta_1 \cdot \bar{x}$$

$$m = \frac{n\Sigma xy - (\Sigma x)(\Sigma y)}{n\Sigma x^2 - (\Sigma x)^2}$$

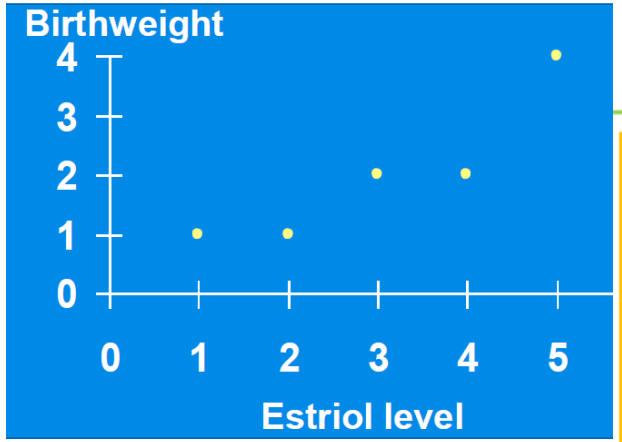
$$b = \frac{(\sum y \cdot \sum x^2) - (\sum x \cdot \sum xy)}{n \cdot \sum x^2 - (\sum x)^2}$$

$$\hat{Y} = 0.95x + 1.5$$

E.g. Parameter Estimation

- What is the **relationship** between Mother's Estriol level & Birthweight using the following data?

Estriol (mg/24h)	Birthweight (g/1000)
1	1
2	1
3	2
4	2
5	4



Decide –X and Y

Parameter Estimation Solution Table

X_i	Y_i	X_i^2	Y_i^2	X_iY_i
1	1	1	1	1
2	1	4	1	2
3	2	9	4	6
4	2	16	4	8
5	4	25	16	20
15	10	55	26	37

Coefficient Interpretation

1. Slope ($\hat{\beta}_1$)
   - Birthweight (Y) is Expected to Increase by .7 Units for Each 1 unit Increase in Estriol (X).
2. Intercept (β_0)
   - Average Birthweight (Y) is -.10 Units When Estriol level (X) Is 0
     - Difficult to explain
     - The birthweight should always be positive

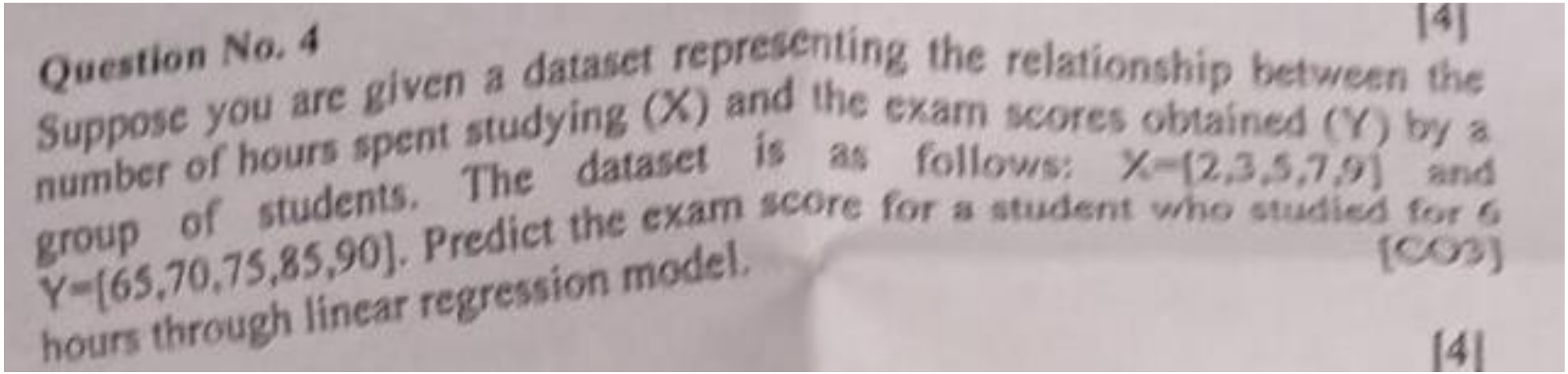
Parameter Estimation Solution

$$\hat{\beta}_1 = \frac{\sum_{i=1}^{n} X_i Y_i - \frac{\left(\sum_{i=1}^{n} X_i\right)\left(\sum_{i=1}^{n} Y_i\right)}{n}}{\sum_{i=1}^{n} X_i^2 - \frac{\left(\sum_{i=1}^{n} X_i\right)^2}{n}} = \frac{37 - \frac{(15)(10)}{5}}{55 - \frac{(15)^2}{5}} = 0.70$$

$$\hat{\beta}_0 = \bar{Y} - \hat{\beta}_1 \bar{X} = 2 - (0.7)(3) = -0.1$$

$$\hat{y} = -.1 + .7x$$

Previous year question



Question No. 4 [4]

Suppose you are given a dataset representing the relationship between the number of hours spent studying (X) and the exam scores obtained (Y) by a group of students. The dataset is as follows: X=[2,3,5,7,9] and Y=[65,70,75,85,90]. Predict the exam score for a student who studied for 6 hours through linear regression model. [CO3]

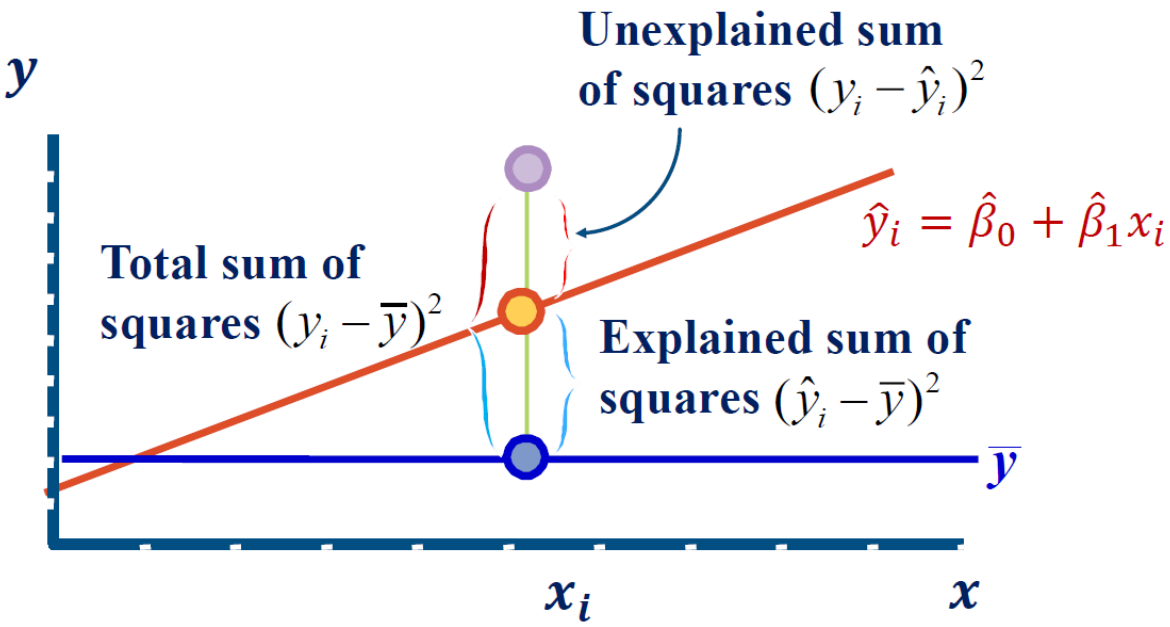
[4]

Goodness of Fit

Goodness of Fit refers to how well a model fits the observed data. It measures the discrepancy between the observed data and the values predicted by the model. In simple terms, it helps us assess how well the chosen model explains the variability in the data.

Why is Goodness of Fit Important?

When building a model, especially in machine learning or statistical analysis, it's crucial to know whether the model you've created accurately represents the data or if it's underfitting or overfitting. A model with a good fit will make accurate predictions on both training data and unseen data (generalization).



SST: The total variability in the dependent variable.
SSR: The variability explained by the model.
SSE: The remaining variability not explained by the model.

Together, these terms help evaluate the performance of a regression model and indicate how well it fits the data.

Coefficient of Correlation

X	Y
1	1
2	1
3	2
4	2
5	4

Calculate the Slope (b_1) and Intercept (b_0)

The formulas for the slope (b_1) and intercept (b_0) of the linear regression line are:

$$b_1 = \frac{n\sum xy - \sum x \sum y}{n\sum x^2 - (\sum x)^2}$$

$$b_0 = \frac{\sum y - b_1 \sum x}{n}$$

Where:

- n is the number of data points
- x and y are the given data points

Let's calculate these values.

1. $\sum x = 1 + 2 + 3 + 4 + 5 = 15$
2. $\sum y = 1 + 1 + 2 + 2 + 4 = 10$
3. $\sum xy = (1)(1) + (2)(1) + (3)(2) + (4)(2) + (5)(4) = 1 + 2 + 6 + 8 + 20 = 37$
4. $\sum x^2 = 1^2 + 2^2 + 3^2 + 4^2 + 5^2 = 1 + 4 + 9 + 16 + 25 = 55$
5. $n = 5$

Plugging these into the formulas:

$$b_1 = \frac{5(37) - 15(10)}{5(55) - 15^2} = \frac{185 - 150}{275 - 225} = \frac{35}{50} = 0.7$$

$$b_0 = \frac{10 - 0.7(15)}{5} = \frac{10 - 10.5}{5} = \frac{-0.5}{5} = -0.1$$

3. Calculate the Predicted $\hat{y}$ Values

The predicted $\hat{y}$ values can be calculated using the regression equation:

$$\hat{y} = b_0 + b_1 x$$

Using $b_0 = -0.1$ and $b_1 = 0.7$:

x	y	$\hat{y}$
1	1	$-0.1 + 0.7(1) = 0.6$
2	1	$-0.1 + 0.7(2) = 1.3$
3	2	$-0.1 + 0.7(3) = 2.0$
4	2	$-0.1 + 0.7(4) = 2.7$
5	4	$-0.1 + 0.7(5) = 3.4$

4. Calculate SSE, SSR, and SST

Formulas:

- Sum of Squares due to Error (SSE):

$$SSE = \sum (y - \hat{y})^2$$

- Sum of Squares due to Regression (SSR):

$$SSR = \sum (\hat{y} - \bar{y})^2$$

- Total Sum of Squares (SST):

$$SST = \sum (y - \bar{y})^2$$

Let's calculate these values.

1. $\bar{y} = \frac{10}{5} = 2$
2. Calculating the values:

x	y	$\hat{y}$	$y - \hat{y}$	$(y - \hat{y})^2$	$\hat{y} - \bar{y}$	$(\hat{y} - \bar{y})^2$	$y - \bar{y}$	$(y - \bar{y})^2$
1	1	0.6	0.4	0.16	-1.4	1.96	-1	1.0
2	1	1.3	-0.3	0.09	-0.7	0.49	-1	1.0
3	2	2.0	0.0	0.00	0.0	0.00	0	0.0
4	2	2.7	-0.7	0.49	0.7	0.49	0	0.0
5	4	3.4	0.6	0.36	1.4	1.96	2	4.0

Now we calculate the sums:

$$SSE = 0.16 + 0.09 + 0.00 + 0.49 + 0.36 = 1.1$$

$$SSR = 1.96 + 0.49 + 0.00 + 0.49 + 1.96 = 4.9$$

$$SST = 1.0 + 1.0 + 0.0 + 0.0 + 4.0 = 6.0$$

5. Calculate the Coefficient of Determination (R^2)

The coefficient of determination R^2 is given by:

$$R^2 = \frac{SSR}{SST}$$

$$R^2 = \frac{4.9}{6.0} = 0.8167 \approx 0.82$$

6. Calculate the Correlation Coefficient (r)

The correlation coefficient r is the square root of R^2:

$$r = \sqrt{R^2} = \sqrt{0.82} \approx 0.9055$$

ANOTHER FORMULA

Coefficient of Correlation

$$r = \frac{SS_{xy}}{\sqrt{SS_{xx} SS_{yy}}}$$

$$SS_{xx} = \sum x^2 - \frac{\left(\sum x\right)^2}{n}$$

$$SS_{yy} = \sum y^2 - \frac{\left(\sum y\right)^2}{n}$$

$$SS_{xy} = \sum xy - \frac{\left(\sum x\right)\left(\sum y\right)}{n}$$

Coefficient of Determination

Proportion of variation 'explained' by relationship between x and y

$$r^2 = \frac{\text{Explained Variation}}{\text{Total Variation}} = \frac{SS_{yy} - SSE}{SS_{yy}}$$

$0 \le r^2 \le 1$

$r^2 = (\text{coefficient of correlation})^2$

You're a marketing analyst for any Toys.

Ad (₹)	Sales (Qty)
1	1
2	1
3	2
4	2
5	4

x_i	y_i	x_i^2	y_i^2	$x_i y_i$
1	1	1	1	1
2	1	4	1	2
3	2	9	4	6
4	2	16	4	8
5	4	25	16	20
15	10	55	26	37

Calculate the **coefficient of correlation**.

$$SS_{xx} = \sum x^2 - \frac{\left(\sum x\right)^2}{n} = 55 - \frac{(15)^2}{5} = 10$$

$$SS_{yy} = \sum y^2 - \frac{\left(\sum y\right)^2}{n} = 26 - \frac{(10)^2}{5} = 6$$

$$SS_{xy} = \sum xy - \frac{\left(\sum x\right)\left(\sum y\right)}{n} = 37 - \frac{(15)(10)}{5} = 7$$

$$r = \frac{SS_{xy}}{\sqrt{SS_{xx} SS_{yy}}} = \frac{7}{\sqrt{10 \cdot 6}} = .904$$

r^2 = (coefficient of correlation)^2

$r^2 = (.904)^2$

$r^2 = .817$

Interpretation: About 81.7% of the sample variation in Sales (y) can be explained by using Ad ₹ (x) to predict Sales (y) in the linear model.

You're an economist for the county cooperative. You gather the following data:

Fertilizer (lb.)	Yield (lb.)
4	3.0
6	5.5
10	6.5
12	9.0

Find the **coefficient of correlation**.

x_i	y_i	x_i^2	y_i^2	$x_i y_i$
4	3.0	16	9.00	12
6	5.5	36	30.25	33
10	6.5	100	42.25	65
12	9.0	144	81.00	108
32	24.0	296	162.50	218

$$SS_{xx} = \sum x^2 - \frac{\left(\sum x\right)^2}{n} = 296 - \frac{(32)^2}{4} = 40$$

$$SS_{yy} = \sum y^2 - \frac{\left(\sum y\right)^2}{n} = 162.5 - \frac{(24)^2}{4} = 18.5$$

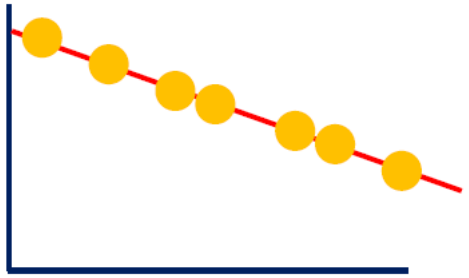
$$SS_{xy} = \sum xy - \frac{\left(\sum x\right)\left(\sum y\right)}{n} = 218 - \frac{(32)(24)}{4} = 26$$

$$r = \frac{SS_{xy}}{\sqrt{SS_{xx} SS_{yy}}} = \frac{26}{\sqrt{40 \cdot 18.5}} = .956 \qquad R^2 = (0.956)^2 = 0.914$$

A correlation coefficient of **0.956** indicates a **very strong positive relationship** between the two variables.

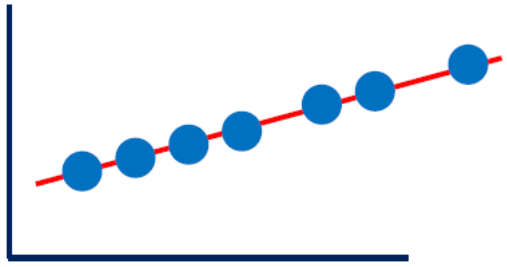
This means that 91.4% of the variability in the dependent variable Y can be explained by the independent variable X. The remaining 8.6% is due to factors not captured by the model.

Coefficient of Determination



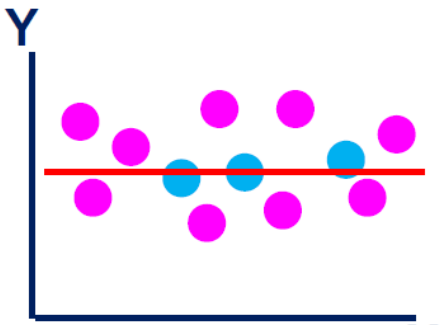
$r^2 = 1$

Perfect linear relationship between X and Y:



$r^2 = 1$

100% of the variation in Y is explained by variation in X



$r^2 = 0$

$r^2 = 0$

- No linear relationship between X and Y:
- The value of Y does not depend on X. (None of the variation in Y is explained by variation in X)

Answers 'How strong is the **linear** relationship between two variables?'

Coefficient of correlation

- Sample correlation coefficient denoted *r*
- Values range from –1 to +1
- Measures degree of association
- Does not indicate cause–effect relationship

Residual Analysis (also called errors)

A residual is the individual difference between the observed value and the predicted value for each data point in a regression model. Residuals show how much each data point deviates from the regression line.

Formula for Residuals:

For each observation i,

$$\text{Residual} = y_i - \hat{y}_i$$

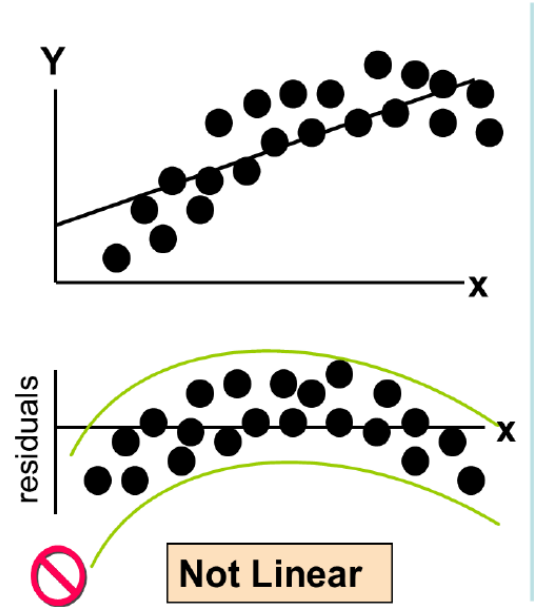
Where:

- y_i is the **actual value** of the dependent variable for observation i,
- $\hat{y}_i$ is the **predicted value** for the same observation i.

Characteristics:

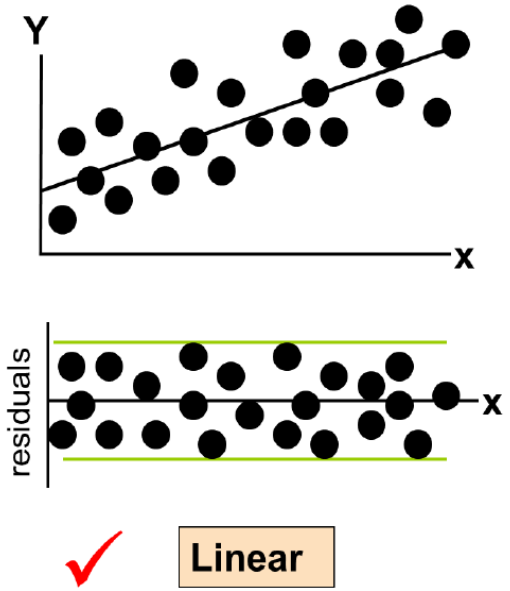
- Residuals are **specific to each data point**.
- They can be positive (if the predicted value is less than the actual value) or negative (if the predicted value is more than the actual value).
- The goal of a regression model is to minimize the sum of the squared residuals (this is called **Ordinary Least Squares (OLS)** in linear regression).
- Each residual gives a local view of the error, one per data point.

- Check the assumptions of regression by examining the residuals
  - Examine for linearity assumption
  - Evaluate independence assumption
  - Evaluate normal distribution assumption
  - Examine for constant variance for all levels of X (homoscedasticity)



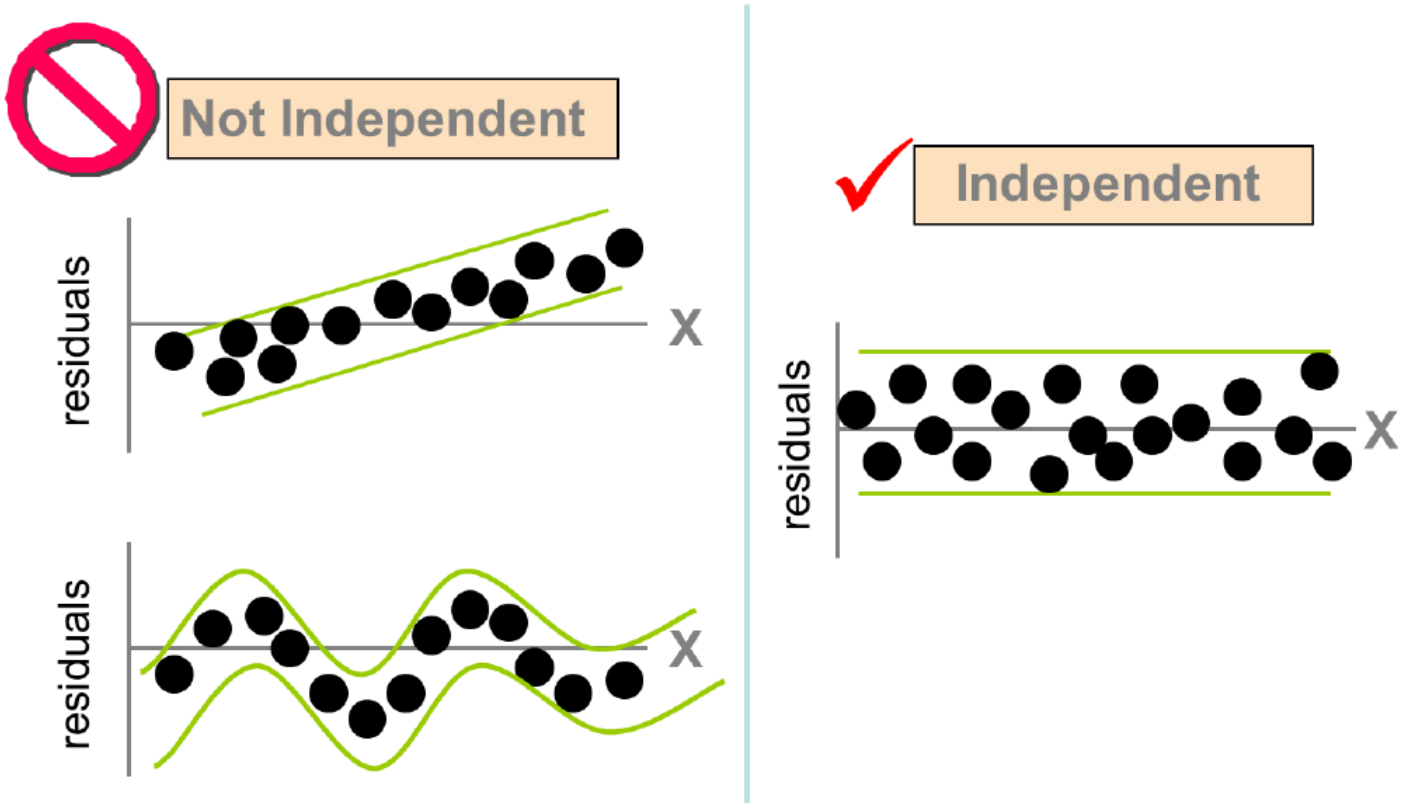
1. Linearity Assumption

The **linearity assumption** means that the relationship between the independent variable(s) (X) and the dependent variable (Y) should be **linear**. This implies that the residuals should not show any patterns and should scatter randomly around zero.



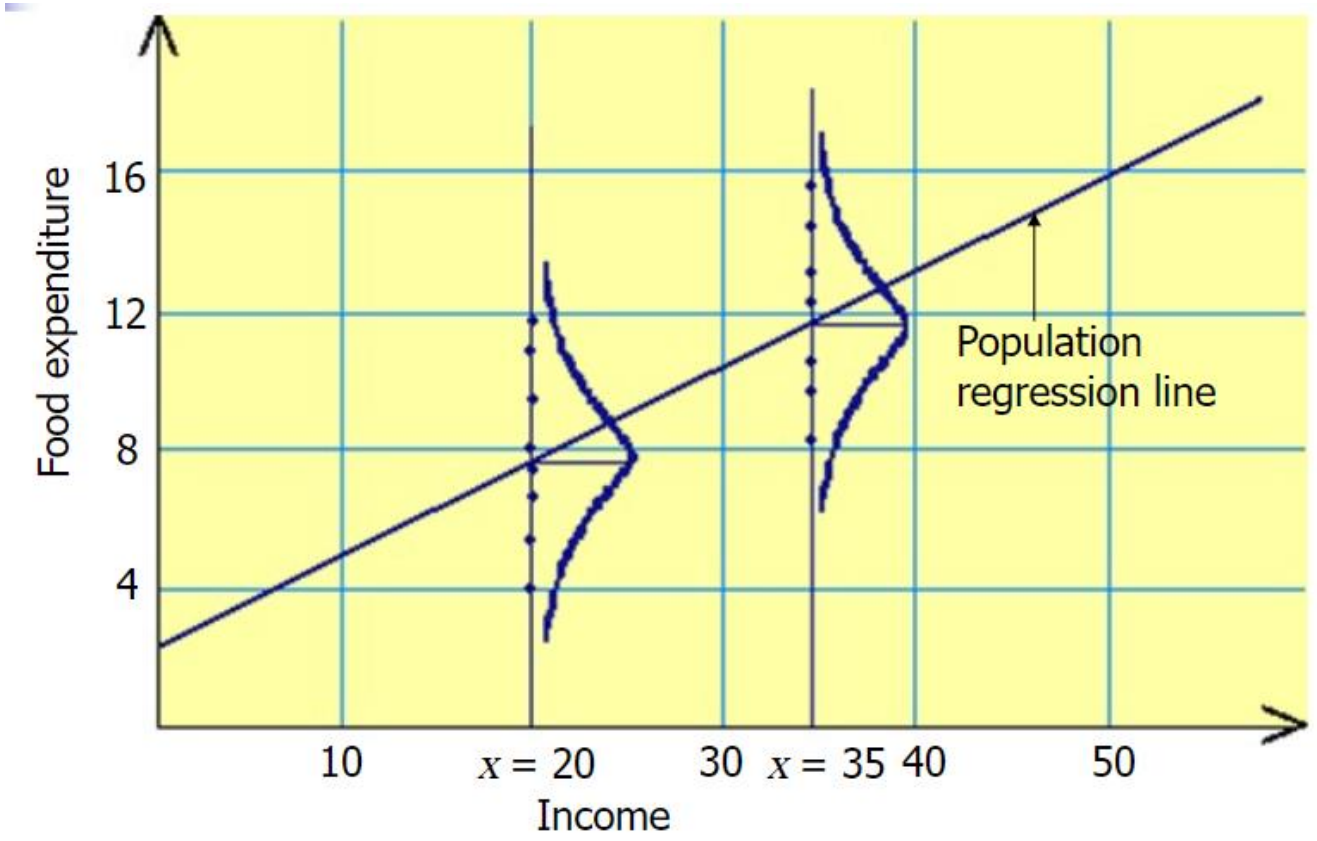
2. Independence Assumption

The **independence assumption** means that the residuals should not be **correlated** with each other. In other words, the value of one residual should not depend on the value of another residual. This is especially important in time-series data or when there are repeated measurements.



3. **Normal Distribution Assumption (Normality of Residuals)**

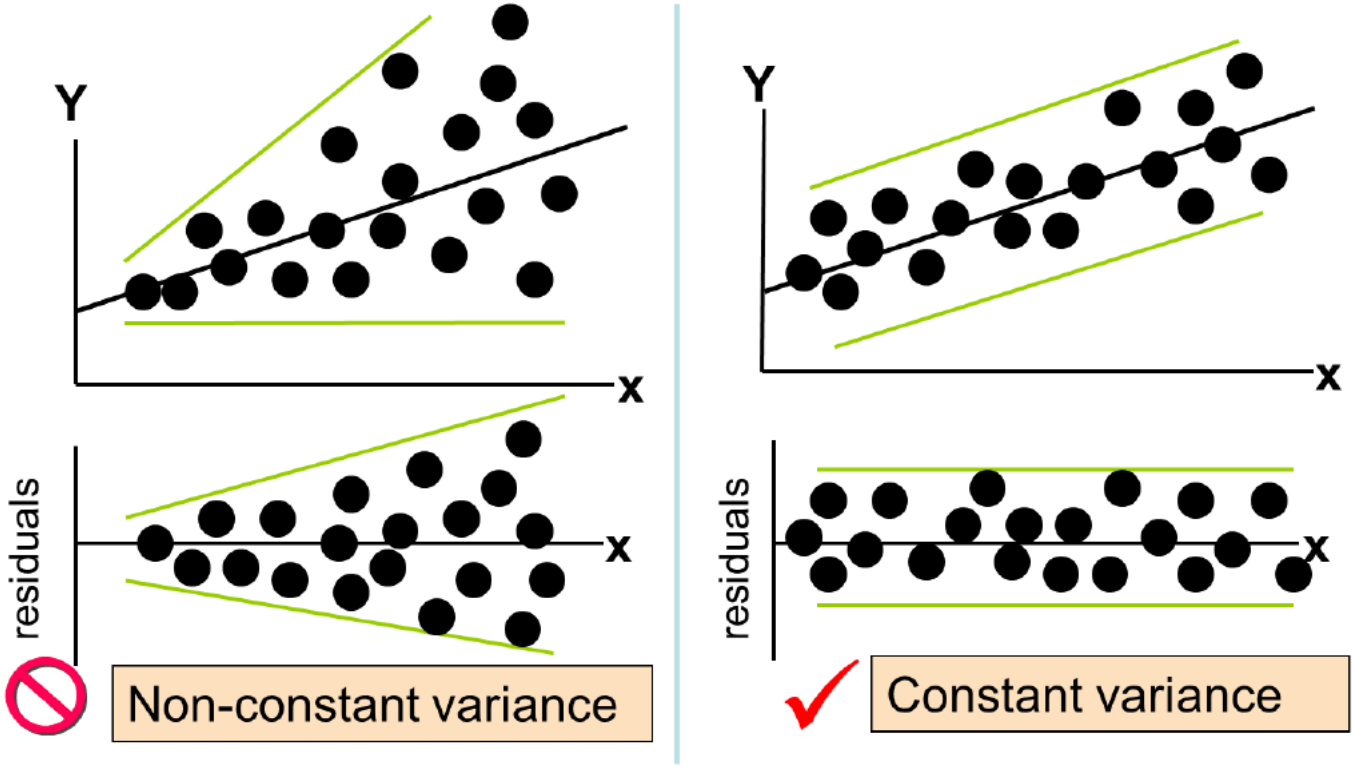
The **normal distribution assumption** suggests that the residuals should be **normally distributed** around zero. This is important because many statistical tests (like t-tests for significance) assume normally distributed residuals.



4. **Constant Variance Assumption (Homoscedasticity)**

The **constant variance assumption** means that the residuals should have **constant variance** across all levels of the independent variable(s) X. In other words, the spread (or dispersion) of the residuals should be approximately the same across the entire range of predicted values.

- When the variance of the residuals changes across the values of X, this is known as **heteroscedasticity**.



Standard Error of Estimate (SEE) (s) (σ ^2)

The **Standard Error of Estimate (SEE)**, also called the **Standard Error of the Regression**, is a **global measure** of the overall fit of the regression model. It calculates the **average magnitude** of the residuals (errors) across all the data points, showing how well the regression line approximates the actual data.

Formula for SEE:

For a simple linear regression with n data points:

(s) Or $$\text{SEE} = \sqrt{\frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n-2}}$$

Where:

- y_i are the actual values,
- $\hat{y}_i$ are the predicted values,
- n is the number of data points.

The subtraction of 2 can be thought of as the fact that we have estimated two parameters: β_0 and β_1

- **Sum of Squares due to Error (SSE):**

$$SSE = \sum (y - \hat{y})^2$$

Estimation of σ^2

$$s^2 = \frac{SSE}{n-2} \qquad \text{Where } SSE = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

The subtraction of 2 can be thought of as the fact that we have estimated two parameters: β_0 and β_1

$$s = \sqrt{s^2} = \sqrt{\frac{SSE}{n-2}}$$

Characteristics:

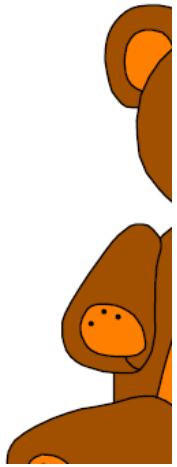
- SEE gives an **overall summary** of how far the data points are from the regression line on average.
- It is a single value that measures the typical error across the entire dataset.
- The SEE is always **positive** and is expressed in the same units as the dependent variable.
- A smaller SEE means that the residuals are smaller, meaning that the predictions are more accurate.

Practice Questions

You're a marketing analyst for any Toys. the following data:

Ad (₹)	Sales (Qty)
1	1
2	1
3	2
4	2
5	4

Find **SSE**, s^2, and **s**.



x_i	y_i	$\hat{y} = -.1 + .7x$	$y - \hat{y}$	$(y - \hat{y})^2$
1	1	.6	.4	.16
2	1	1.3	-.3	.09
3	2	2	0	0
4	2	2.7	-.7	.49
5	4	3.4	.6	.36
				SSE=1.1

$$s^2 = \frac{SSE}{n-2} = \frac{1.1}{5-2} = .36667 \qquad s = \sqrt{.36667} = .6055$$

The SEE of 0.60 means that, on average, the actual values of the dependent variable Y (observed values) differ from the predicted values $Y^{\wedge}$ by about 0.60 units.

In other words, the typical prediction error or residual is 0.60 units away from the actual data points.

Test of Slope Coefficient

Test of Slope Coefficient in Linear Regression

The **test of the slope coefficient** in linear regression is used to determine if there is a significant linear relationship between the independent variable X and the dependent variable Y. This is done through a **hypothesis test** for the slope (β_1) of the regression line.

The null and alternative hypotheses are:

- **Null Hypothesis** H_0: $\beta_1 = 0$ (There is no relationship between X and Y)
- **Alternative Hypothesis** H_a: $\beta_1 \neq 0$ (There is a significant relationship between X and Y)

Formula for the t-statistic:

estimated slope

$$t = \frac{\widehat{\beta_1}}{S_{\widehat{\beta}_1}} = \frac{\widehat{\beta_1}}{\frac{S}{\sqrt{SS_{xx}}}}, \qquad SS_{xx} = \sum_{i=1}^{n} x_i^2 - \frac{(\sum_{i=1}^{n} x_i)^2}{n} \qquad df = 2$$

standard error of the slope

Steps in Testing the Slope Coefficient:

1. **Fit the regression model** using least squares.
2. **Compute the standard error of the slope**.
3. **Calculate the t-statistic** for the slope.
4. **Compare the t-statistic** to the critical value from the t-distribution,

t-test table

cum. prob	$t_{.50}$	$t_{.75}$	$t_{.80}$	$t_{.85}$	$t_{.90}$	$t_{.95}$	$t_{.975}$	$t_{.99}$	$t_{.995}$	$t_{.999}$	$t_{.9995}$
one-tail	**0.50**	**0.25**	**0.20**	**0.15**	**0.10**	**0.05**	**0.025**	**0.01**	**0.005**	**0.001**	**0.0005**
two-tails	**1.00**	**0.50**	**0.40**	**0.30**	**0.20**	**0.10**	**0.05**	**0.02**	**0.01**	**0.002**	**0.001**
df											
1	0.000	1.000	1.376	1.963	3.078	6.314	12.71	31.82	63.66	318.31	636.62
2	0.000	0.816	1.061	1.386	1.886	2.920	4.303	6.965	9.925	22.327	31.599
3	0.000	0.765	0.978	1.250	1.638	2.353	3.182	4.541	5.841	10.215	12.924
4	0.000	0.741	0.941	1.190	1.533	2.132	2.776	3.747	4.604	7.173	8.610
5	0.000	0.727	0.920	1.156	1.476	2.015	2.571	3.365	4.032	5.893	6.869
6	0.000	0.718	0.906	1.134	1.440	1.943	2.447	3.143	3.707	5.208	5.959
7	0.000	0.711	0.896	1.119	1.415	1.895	2.365	2.998	3.499	4.785	5.408
8	0.000	0.706	0.889	1.108	1.397	1.860	2.306	2.896	3.355	4.501	5.041
9	0.000	0.703	0.883	1.100	1.383	1.833	2.262	2.821	3.250	4.297	4.781
10	0.000	0.700	0.879	1.093	1.372	1.812	2.228	2.764	3.169	4.144	4.587
11	0.000	0.697	0.876	1.088	1.363	1.796	2.201	2.718	3.106	4.025	4.437
12	0.000	0.695	0.873	1.083	1.356	1.782	2.179	2.681	3.055	3.930	4.318
13	0.000	0.694	0.870	1.079	1.350	1.771	2.160	2.650	3.012	3.852	4.221
14	0.000	0.692	0.868	1.076	1.345	1.761	2.145	2.624	2.977	3.787	4.140
15	0.000	0.691	0.866	1.074	1.341	1.753	2.131	2.602	2.947	3.733	4.073
16	0.000	0.690	0.865	1.071	1.337	1.746	2.120	2.583	2.921	3.686	4.015
17	0.000	0.689	0.863	1.069	1.333	1.740	2.110	2.567	2.898	3.646	3.965
18	0.000	0.688	0.862	1.067	1.330	1.734	2.101	2.552	2.878	3.610	3.922
19	0.000	0.688	0.861	1.066	1.328	1.729	2.093	2.539	2.861	3.579	3.883
20	0.000	0.687	0.860	1.064	1.325	1.725	2.086	2.528	2.845	3.552	3.850
21	0.000	0.686	0.859	1.063	1.323	1.721	2.080	2.518	2.831	3.527	3.819
22	0.000	0.686	0.858	1.061	1.321	1.717	2.074	2.508	2.819	3.505	3.792
23	0.000	0.685	0.858	1.060	1.319	1.714	2.069	2.500	2.807	3.485	3.768
24	0.000	0.685	0.857	1.059	1.318	1.711	2.064	2.492	2.797	3.467	3.745
25	0.000	0.684	0.856	1.058	1.316	1.708	2.060	2.485	2.787	3.450	3.725
26	0.000	0.684	0.856	1.058	1.315	1.706	2.056	2.479	2.779	3.435	3.707
27	0.000	0.684	0.855	1.057	1.314	1.703	2.052	2.473	2.771	3.421	3.690
28	0.000	0.683	0.855	1.056	1.313	1.701	2.048	2.467	2.763	3.408	3.674
29	0.000	0.683	0.854	1.055	1.311	1.699	2.045	2.462	2.756	3.396	3.659
30	0.000	0.683	0.854	1.055	1.310	1.697	2.042	2.457	2.750	3.385	3.646
40	0.000	0.681	0.851	1.050	1.303	1.684	2.021	2.423	2.704	3.307	3.551
60	0.000	0.679	0.848	1.045	1.296	1.671	2.000	2.390	2.660	3.232	3.460
80	0.000	0.678	0.846	1.043	1.292	1.664	1.990	2.374	2.639	3.195	3.416
100	0.000	0.677	0.845	1.042	1.290	1.660	1.984	2.364	2.626	3.174	3.390
1000	0.000	0.675	0.842	1.037	1.282	1.646	1.962	2.330	2.581	3.098	3.300
z	0.000	0.674	0.842	1.036	1.282	1.645	1.960	2.326	2.576	3.090	3.291
	0%	50%	60%	70%	80%	90%	95%	98%	99%	99.8%	99.9%
	Confidence Level										

You're a marketing analyst for any Toys. You find β0 = –0.1, β1 = 0.7 and s = 0.6055 Is the relationship significant the .05 level of significance?

X	Y
1	2
2	3
3	5
4	4
5	6

Given:

- Slope β_1 = 0.7
- Intercept β_0 = -0.1
- Standard error of the estimate $s = 0.6055$
- Significance level (α) = 0.05
- $X = [1, 2, 3, 4, 5]$, $Y = [1, 1, 2, 2, 4]$

Step 1: State the Hypotheses

We will test the null hypothesis that the slope $\beta_1 = 0$, which would mean there is no linear relationship between X and Y.

- Null Hypothesis $H_0 : \beta_1 = 0$ (No relationship between X and Y)
- Alternative Hypothesis $H_a : \beta_1 \neq 0$ (There is a significant relationship between X and Y)

Step 2: Calculate the Standard Error of the Slope

$$S_{\hat{\beta}_1} = \frac{s}{\sqrt{SS_{XX}}} \qquad Where \quad SS_{xx} = \sum_{i=1}^{n} x_i^2 - \frac{(\sum_{i=1}^{n} x_i)^2}{n} \qquad s = \sqrt{s^2} = \sqrt{\frac{SSE}{n-2}}$$

Where $SSE = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2$

x_i	y_i	x_i^2	y_i^2	$x_i y_i$
1	1	1	1	1
2	1	4	1	2
3	2	9	4	6
4	2	16	4	8
5	4	25	16	20
15	10	55	26	37

$$S_{\hat{\beta}_1} = \frac{s}{\sqrt{SS_{XX}}} = \frac{0.6055}{\sqrt{55 - \frac{15^2}{5}}} = .1914$$

Step 3: Calculate the t-Statistic

Given : s = 0.6055

$$t = \frac{0.70}{.1914} = 3.657$$ t- calculated value

Another Formula

To calculate the **standard error of the slope** SE_{β_1}, we use the formula:

$$SE_{\beta_1} = \frac{s}{\sqrt{\sum (X_i - \bar{X})^2}}$$

$$SE_{\beta_1} = \frac{0.6055}{\sqrt{10}} = \frac{0.6055}{3.1623} = 0.1915$$

- $s = 0.6055$ (the standard error of the estimate),
- $\bar{X} = \frac{1+2+3+4+5}{5} = 3$,
- $\sum (X_i - \bar{X})^2 = (1-3)^2 + (2-3)^2 + (3-3)^2 + (4-3)^2 + (5-3)^2 = 4 + 1 + 0 + 1 + 4 = 10$.

Step 3: Calculate the t-Statistic

$$t = \frac{0.7}{0.1915} = 3.654$$ t- calculated value

Step 4: Determine the Critical t-Value

The degrees of freedom (df) for this test is $n - 2$, where $n = 5$ is the number of data points.

- $df = 5 - 2 = 3$

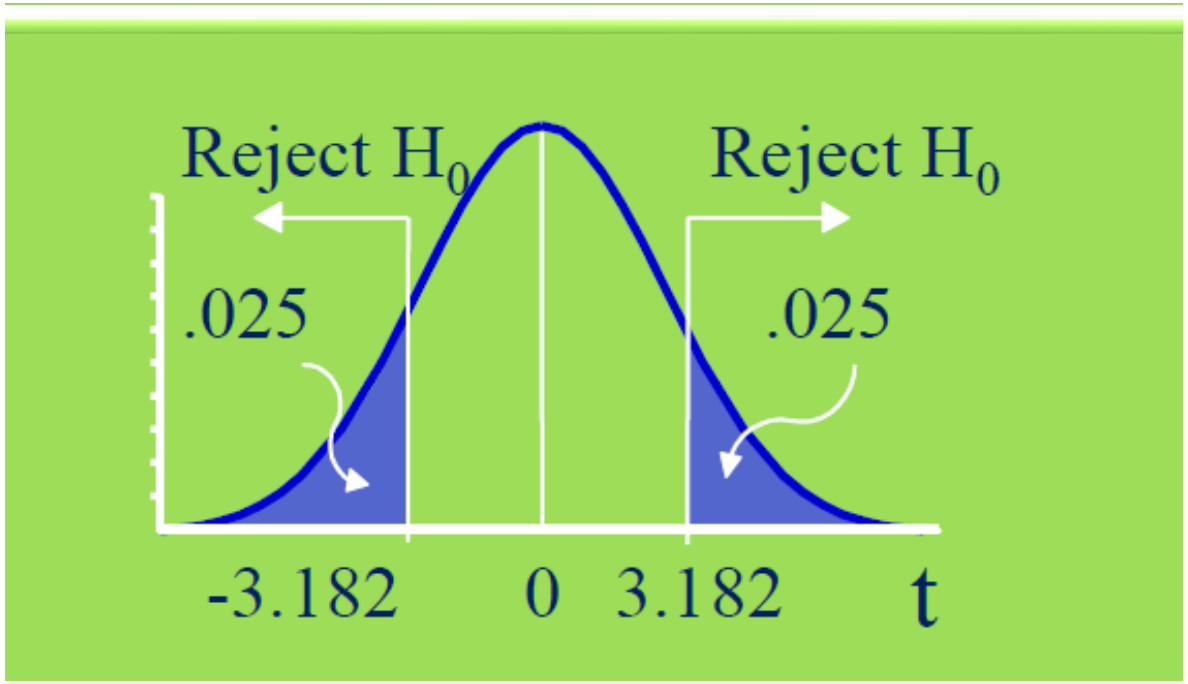
At the 0.05 significance level (two-tailed test), we look up the critical t-value for 3 degrees of freedom.
Using a t-distribution table, the critical value for α=0.05α=0.05 and df=3df=3 is approximately **3.182.**

t-test table

cum. prob	$t_{.50}$	$t_{.75}$	$t_{.80}$	$t_{.85}$	$t_{.90}$	$t_{.95}$	$t_{.975}$	$t_{.99}$	$t_{.995}$	$t_{.999}$	$t_{.9995}$
one-tail	0.50	0.25	0.20	0.15	0.10	0.05	0.025	0.01	0.005	0.001	0.0005
two-tails	1.00	0.50	0.40	0.30	0.20	0.10	0.05	0.02	0.01	0.002	0.001
df											
1	0.000	1.000	1.376	1.963	3.078	6.314	12.71	31.82	63.66	318.31	636.62
2	0.000	0.816	1.061	1.386	1.886	2.920	4.303	6.965	9.925	22.327	31.599
3	0.000	0.765	0.978	1.250	1.638	2.353	3.182	4.541	5.841	10.215	12.924
4	0.000	0.741	0.941	1.190	1.533	2.132	2.776	3.747	4.604	7.173	8.610
5	0.000	0.727	0.920	1.156	1.476	2.015	2.571	3.365	4.032	5.893	6.869
6	0.000	0.718	0.906	1.134	1.440	1.943	2.447	3.143	3.707	5.208	5.959
7	0.000	0.711	0.896	1.119	1.415	1.895	2.365	2.998	3.499	4.785	5.408

Step 5: Compare the t-Statistic to the Critical Value

The calculated t-statistic is **3.654.**
The critical t-value is **3.182.**



Decision: we reject the null hypothesis.

Since |3.654|>3.182

The calculated t-value exceeds the critical value, so we reject the null hypothesis. This means that the slope β_1 is significantly different from 0, and there **is a significant linear relationship** between X and Y at the 0.05 level of significance.

Other Evaluation Metrics

- **Mean Squared Error (MSE)**

$$MSE = \frac{1}{m}\sum_{i=1}^{m}(y_i - \hat{y}_i)^2$$

  - Most commonly used Metric
  - Differentiable due to convex shape
  - Easier to optimize.
  - Penalizes large errors

- **Mean Absolute Error (MAE)**

$$MAE = \frac{1}{m}\sum_{i=1}^{m}|y_i - \hat{y}_i|$$

  - Not preferred in cases where outliers are prominent.
  - MAE does not penalize large errors.
  - **Small MAE** suggests the model is great at prediction, while a **large MAE** suggests that model may have trouble in certain areas.

Linear regression with gradient Descent

At a theoretical level, gradient descent is an algorithm that minimizes functions. Given a function defined by a set of parameters, gradient descent starts with an initial set of parameter values and iteratively moves toward a set of parameter values that minimize the function. This iterative minimization is achieved using calculus, taking steps in the negative direction of the function gradient.

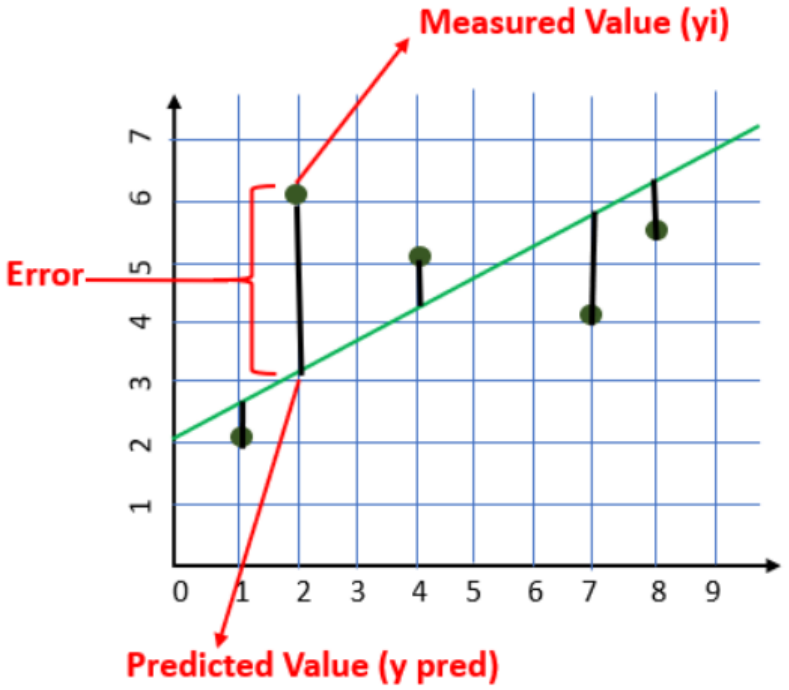
Cost Function

The cost is the error in our predicted value. We will use the Mean Squared Error function to calculate the cost.

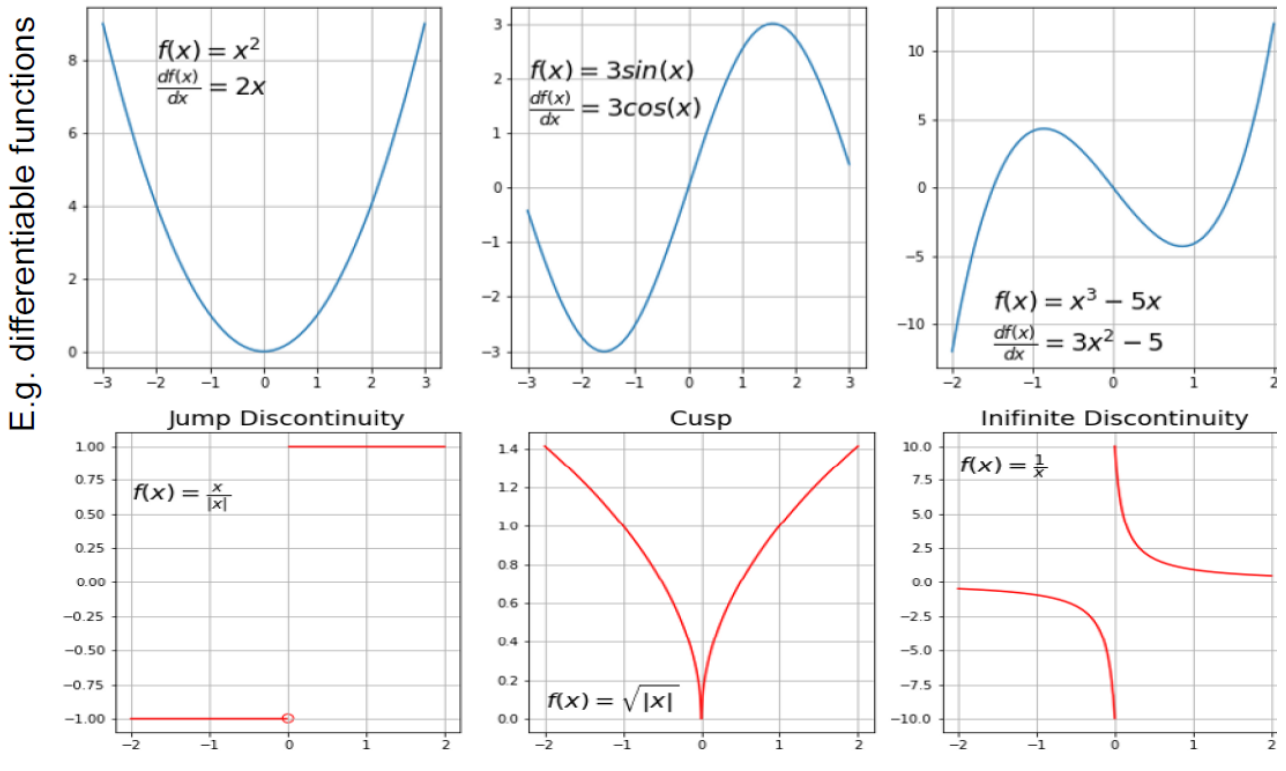
$$Cost\ Function(MSE) = \frac{1}{n}\sum_{i=0}(y_i - y_{i\,pred})^2$$

Replace $y_{i\,pred}$ with $mx_i + c$

$$Cost\ Function(MSE) = \frac{1}{n}\sum_{i=0}^{n}(y_i - (mx_i + c))^2$$

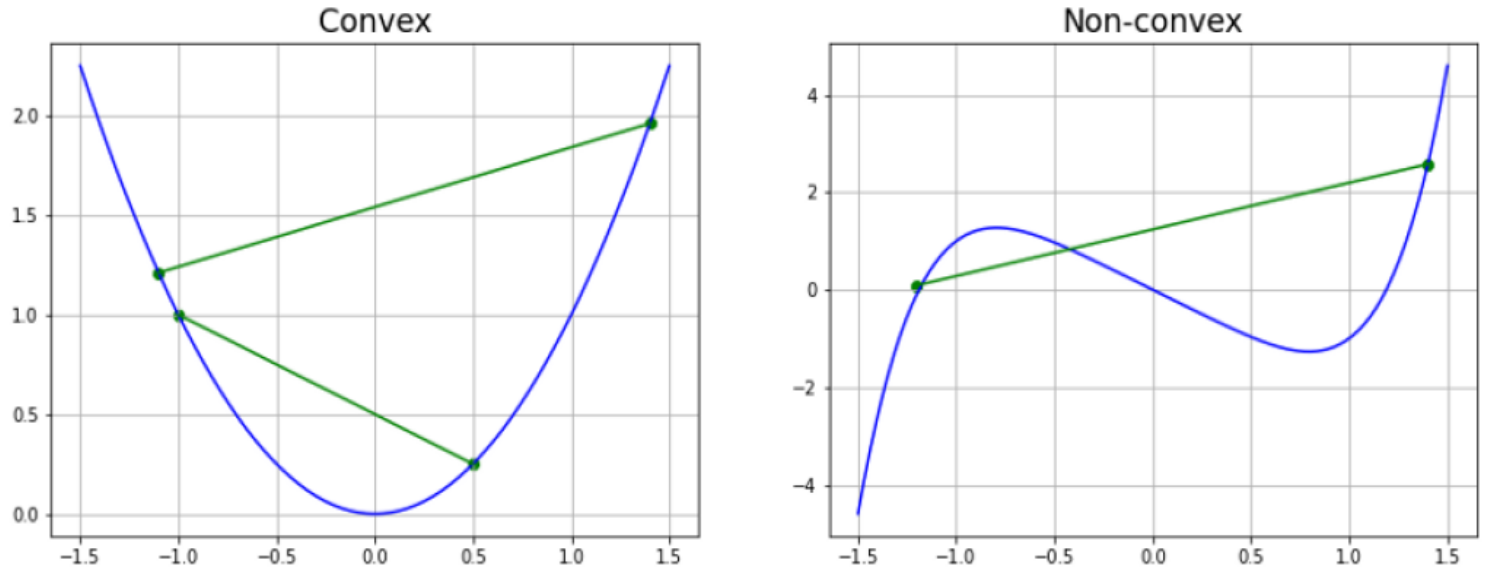


- **Function requirements**
- GD algorithm doesn't work for all functions. Hence, it has two specific requirements that a ***function*** has to be:
  - **Differentiable**
  - **Convex**
- A differentiable function has, its derivative for each point in its domain and not all functions meet this criteria, such as … next slide…



Non-differentiable functions have a step a cusp or a discontinuity

- **Convexity in GD optimization**
  - Our goal is **to minimize the cost function in order to improve the accuracy of the model**. MSE is a convex function (it is differentiable twice). This **means** there is **no local minimum**, but only the ***global minimum***. Thus gradient descent would converge to the global minimum.

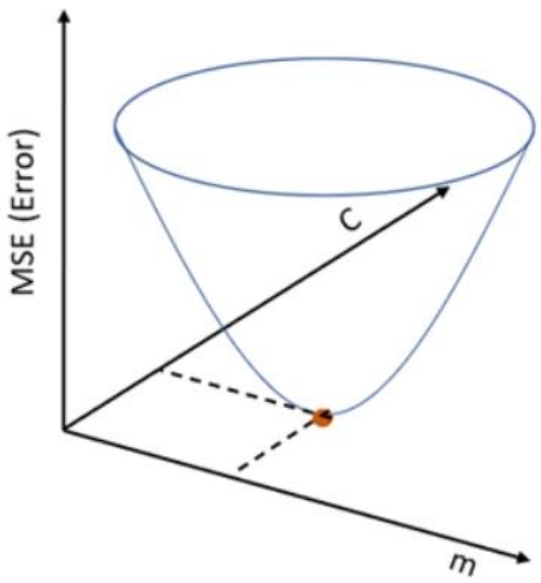


- **Convexity in GD optimization**
  - Another way to check mathematically if a univariate function is ***convex*** then the second derivative is always **greater than 0**.
  - $\frac{d^2f(x)}{dx^2} > 0$; E.g. $f(x) = x^2 - x + 3$; $\frac{df(x)}{dx} = 2x - 1$ and $\frac{d^2f(x)}{dx^2} = 2$
  - Hence, $f(x)$ is **convex**.

Gradient Descent Algorithm

Gradient Descent is an algorithm that finds the best-fit line for a given training dataset in a smaller number of iterations.

If we plot m and c against MSE, it will acquire a bowl shape (As shown in the diagram below)



1. Choose a starting point (initialization)
2. Calculate gradient at this point
3. Make a scaled step in the opposite direction to the gradient (objective: minimize)
4. Repeat points 2 and 3 until one of the criteria is met:
   - Maximum number of iterations reached
   - Step size is smaller than the tolerance (due to scaling or a small gradient)

Vanilla gradient descent, aka batch gradient descent, computes the gradient of the cost function w.r.t. to the parameters **θ** for the entire training dataset:

$$\boldsymbol{\theta_{t+1} = \theta_t - \eta \nabla J(\theta)} \; \boldsymbol{:} \; \boldsymbol{\eta \rightarrow Learning\ Rate}$$

GD algorithm iteratively calculates the next point using gradient at the current position, scales it (by a **learning rate**) and subtracts obtained value from the current position (makes a step).

How to update m and b

$$x(next) = x(current) - \gamma * \nabla f(x)$$

Updated value = old value - learning rare * gradient

Learning rate

The steps which are taken to reach optimal point decides the rate of gradient descent. It is often referred to as 'Learning rate'(i.e., The size of the steps).

- **Too big**
  bounce between the convex function and may not reach the local minimum.
- **Too small**
  gradient descent will eventually reach the local minimum but it will take too much time for that
- **Just right**

  gradient descent will eventually reach the local minimum but it will take too much time for that

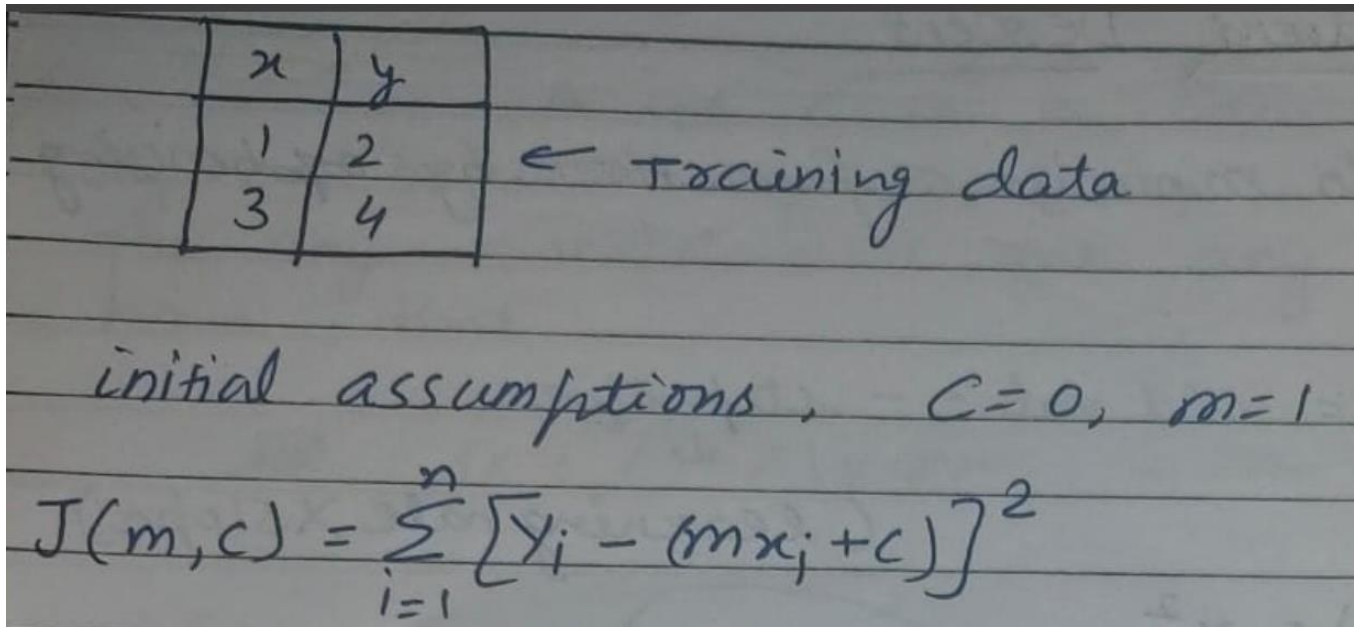
Big Learning Rate

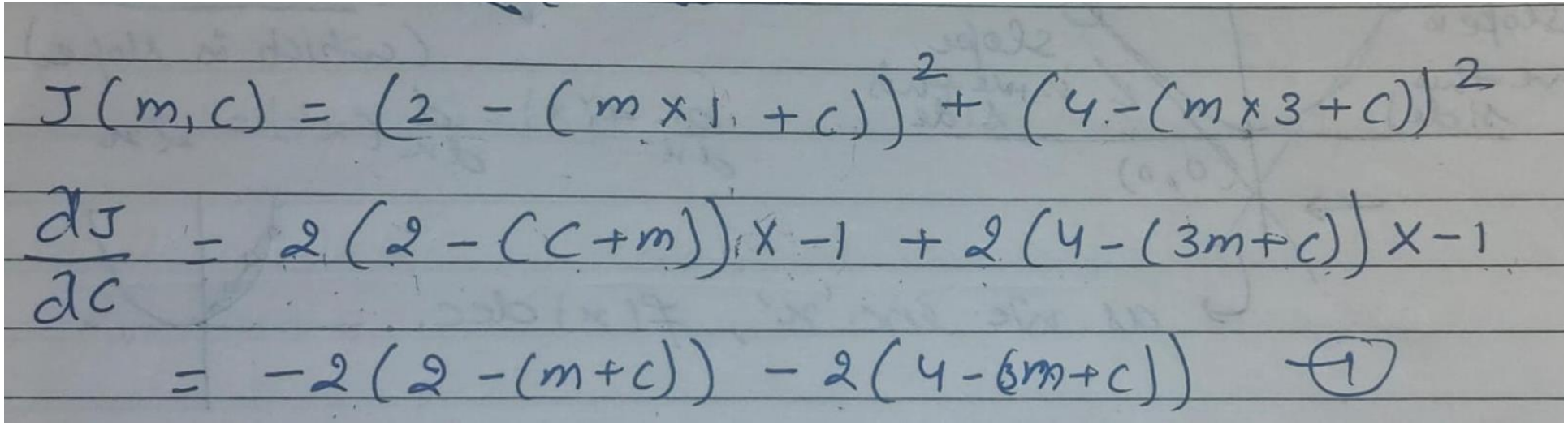
Too small

Just right

- A quadratic function : $f(x) = x^2 - 4x + 1$
- It is a univariate function. $\frac{df(x)}{dx} = 2x - 4$
- let us consider $\eta = 0.1$ and starting point $x = 9$. Then calculation is as follows
- $\boldsymbol{x_1 = 9 - 0.1 * (2 * 9 - 4) = 7.6}$
- $\boldsymbol{x_2 = 7.6 - 0.1 * (2 * 7.6 - 4) = 6.8}$
- $\boldsymbol{x_3 = 6.8 - 0.1 * (2 * 6.8 - 4) = 5.584}$

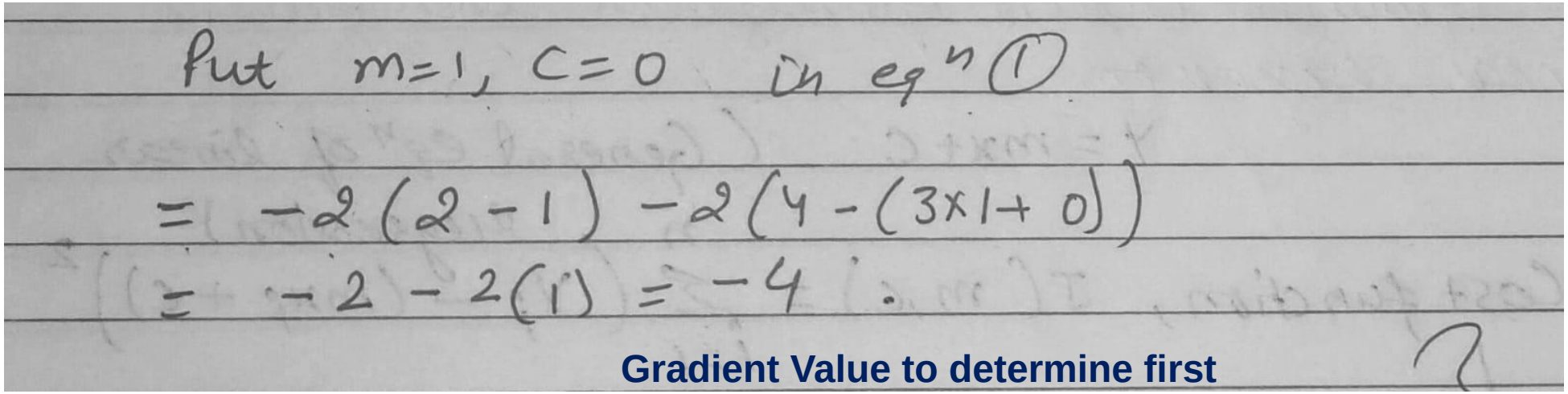
Updated value = old value - learning rare * gradient





EXPAND Function

Partial differentiation W.r.t "C"



Putting old value of "c" and "m "

Gradient Value to determine first iteration of "c" value i.e. new value

$$C_{new} = C_{old} - LR \times (slope)$$

$$= 0 - (0.001 \times -4)$$

$$= 0.004$$

New "c" Value

Likewise :

$$\frac{\partial J}{\partial m} = 2(2 - (m+c)) \times -1 + 2(4 - (3m+c)) \times -3$$

Partial differentiation W.r.t "m"

$$= -2(2 - (m+c)) - 6(4 - (3m+c))$$

Put $m = 1$, $C = 0$

Good Write

$$\frac{\partial J}{\partial m} = -2\left(2-(1+0)\right) - 6\left(4-(3\times 1+0)\right)$$

Putting old value of "c" and "m "

$$= -2(1) - 6(4-3)$$

$$= -2 - 6 = -8$$

Gradient Value to determine first iteration of "m" value i.e. new value

$$m_{new} = m_{old} - (LR \times slope)$$

$$= 1 - 0.001 \times -8$$

$$= 1 + 0.008 = 1.008$$

New "m" Value

Gradient Descent types

- **Batch Gradient Descent**
  A.k.a. Vanilla Gradient Descent. Calculates error for each example. Model is updated only after an epoch.
- **Stochastic Gradient descent**
  SGD unlike vanilla, iterates over each example while updating the model. Frequent updates can be computationally more expensive.
- **Mini Batch Gradient Descent**
  a combination of concepts of both SGD and Batch Gradient Descent.
    - Splits data into batches then performs update on batches balancing between the efficiency of batch gradient descent and the robustness of SGD.

In Batch Gradient Descent we were considering all the examples for every step of Gradient Descent. But what if our dataset is very huge. Deep learning models crave for data. The more the data the more chances of a model to be good. Suppose our dataset has 5 million examples, then just to take one step the model will have to calculate the gradients of all the 5 million examples. This does not seem an efficient way.

It requires to calculate the gradients for the whole data set to perform just one update.

BGD can be very slow and is intractable for datasets that don't fit In memory ,it also doesn't allow us to update the model online i.e BGD isn’t performed on data set that update continuously.

To tackle this problem we have Stochastic Gradient Descent.

Stochastic Gradient Descent

In Stochastic Gradient Descent (SGD), we consider just one example at a time to take a single step.
Stochastic gradient descent(SGD) performs a parameter update for *each* training example x(i) and label y(i).

$$-\theta_{t+1} = \theta_t - \eta \nabla J(\theta; x(i); y(i))$$

- θ: The model parameters (weights) that we want to optimize.
- t: The current iteration or time step.
- $t + 1$: The next iteration or time step.
- η (eta): The learning rate, a small positive scalar that controls the step size in the gradient descent. A high learning rate may cause the updates to overshoot the optimal solution, while a very small learning rate may make the convergence slow.
- $\nabla J(\theta; x_i; y_i)$: The gradient (vector of partial derivatives) of the cost function J with respect to the model parameters θ, calculated for a specific data point (x_i, y_i). This tells us the direction and rate of change of the loss function as we tweak the parameters.

At each step t, the parameters θ are updated by moving in the direction opposite to the gradient of the cost function J with respect to θ. This gradient indicates how to adjust θ to reduce the value of the cost function. The step size of the update is controlled by the learning rate η.

- BGD performs redundant computations for large datasets, SGD avoids this redundancy by performing one update at a time.
- It is therefore usually much faster and can also be used to learn online.
- SGD performs frequent updates with a high variance that cause the objective function to fluctuate heavily .

While SGD's fluctuation, on the one hand, enables it to jump to new and potentially better local minima
On the other hand, this ultimately complicates convergence to the exact minimum, as SGD will keep overshooting

Benefits of SGD Fluctuation:

1. **Escaping local minima**: The fluctuations allow SGD to escape from local minima (which might trap Batch Gradient Descent) and potentially find better solutions.
2. **Faster initial convergence**: Since SGD updates the parameters more frequently (after each data point), it often makes faster progress toward a solution in the early stages of training, especially in large-scale datasets.

Drawbacks of SGD Fluctuation:

1. **Slower final convergence**: The noise in SGD makes it difficult for the algorithm to settle at the exact global minimum, as it keeps oscillating or overshooting once it reaches a region near the minimum.
2. **Less stability**: The constant fluctuation means that SGD might struggle to converge in a smooth and predictable manner, especially with a poorly chosen learning rate.

To tackle this problem we have Mini-Batch Gradient Descent.

In **Mini-batch Gradient Descent**, instead of using:

- **Batch Gradient Descent (BGD)**, which computes the gradient over the **entire dataset** in one step, or
- **Stochastic Gradient Descent (SGD)**, which computes the gradient using only **one data point** at a time,

Mini-batch Gradient Descent computes the gradient using a **small subset** of the data (called a **mini-batch**) in each iteration.

Mini-batch Gradient Descent is a compromise between **Batch Gradient Descent** and **Stochastic Gradient Descent (SGD)**, combining the strengths of both to improve convergence, speed, and stability.

The number of **mini-batches** in one **epoch** depends on two factors:

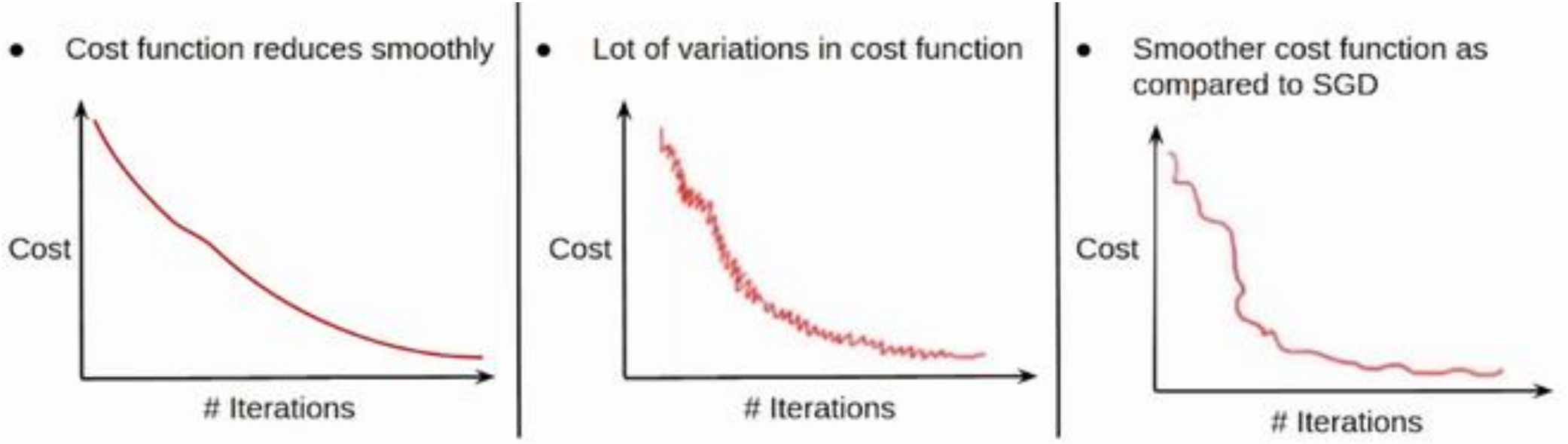
1. **Size of the dataset** n
2. **Mini-batch size** b

The number of mini-batches is simply the number of data points divided by the mini-batch size.

Formula to calculate the number of mini-batches:

$$\text{Number of mini-batches} = \frac{n}{b}$$

- If n is the total number of examples in the dataset.
- b is the mini-batch size.



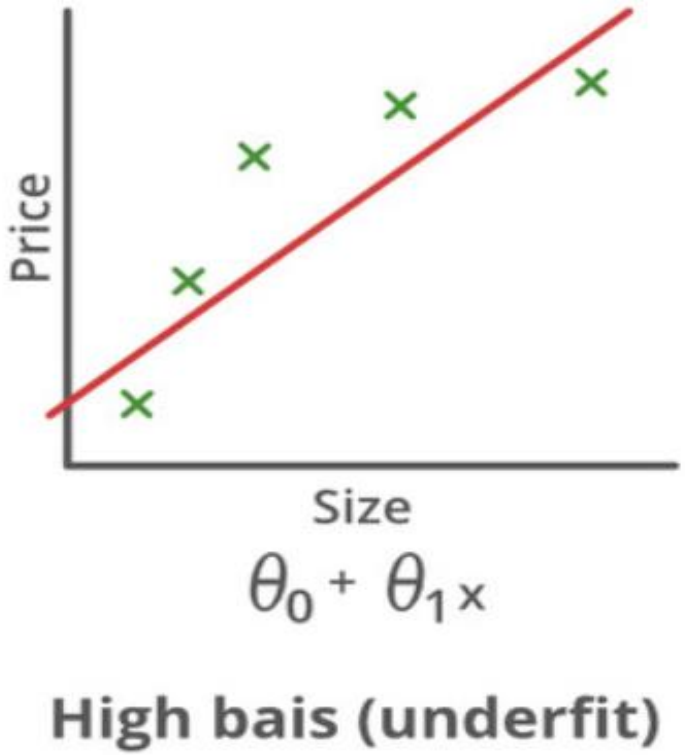
What are Bias and Variance?

Bias

It can be defined as an inability of machine learning algorithms such as Linear Regression to capture the true relationship between the training data points

.

High bias, also known as underfitting, means the model did not learn enough from the dataset.

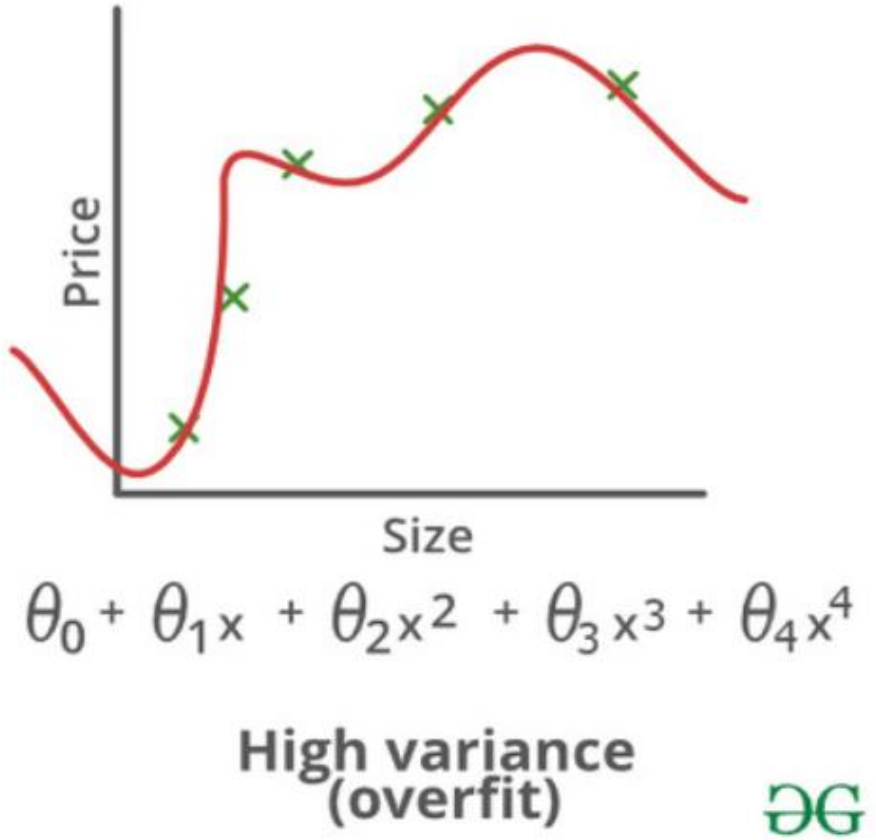


What are Bias and Variance?

Variance

variance implies the error value that occurs when we try to make predictions by using data that is not previously seen by the model

High variance, also known as overfitting, means the model focuses too much on specific patterns in the training dataset and does not generalize well on unseen data.



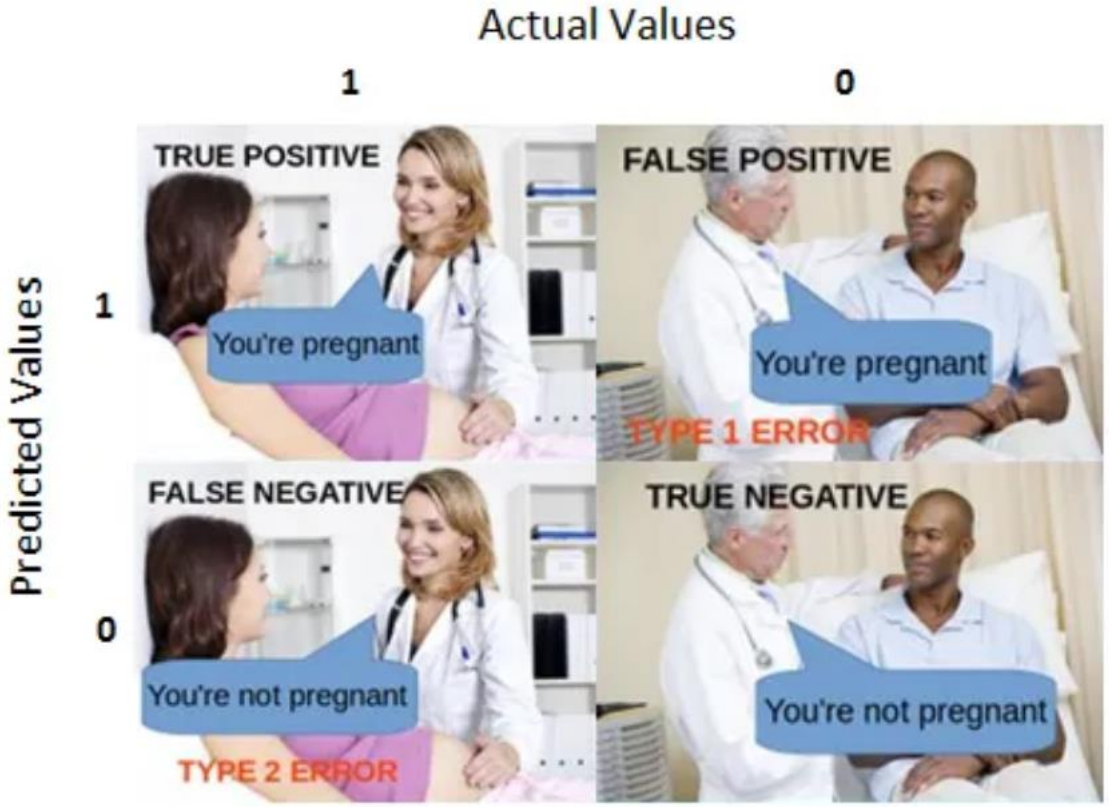
Performance Evaluation : Confusion matrix

The **Confusion Matrix** is a tool used to evaluate the performance of classification models in **Machine Learning (ML)**. It is a square matrix that compares the predicted labels (output) with the actual labels (ground truth) of the data, giving a comprehensive view of how well the model is performing.

For a **binary classification problem**, the confusion matrix is a 2x2 matrix. It consists of four components:

Confusion Matrix

	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)



True Positive

Interpretation: You predicted positive and it's true (Actual -positive).

You predicted that a woman is pregnant and she actually is.

True Negative:

Interpretation: You predicted negative and it's true (Actual-nve).

You predicted that a man is not pregnant and he actually is not.

False Positive: (Type 1 Error)

Interpretation: You predicted positive and it's false (actual –nve)

You predicted that a man is pregnant but he actually is not.

False Negative: (Type 2 Error)

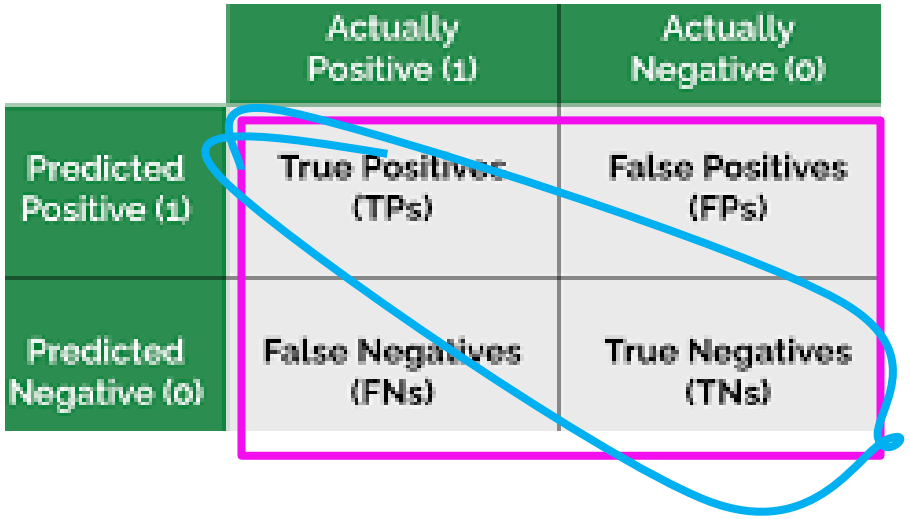
Interpretation: You predicted negative and it's false (actual –positive).

You predicted that a woman is not pregnant but she actually is.

Several important metrics can be calculated using the values from the confusion matrix:

1. **Accuracy**: The overall correctness of the model. It's the ratio of correctly predicted observations (TP + TN) to the total number of observations.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$



2. **Precision**: The ratio of correctly predicted positive observations to the total predicted positive observations (also called Positive Predictive Value).

$$\text{Precision} = \frac{TP}{TP + FP}$$

The above equation can be explained by saying, from all the classes we have predicted as positive, how many are actually positive.

Issues with Accuracy…

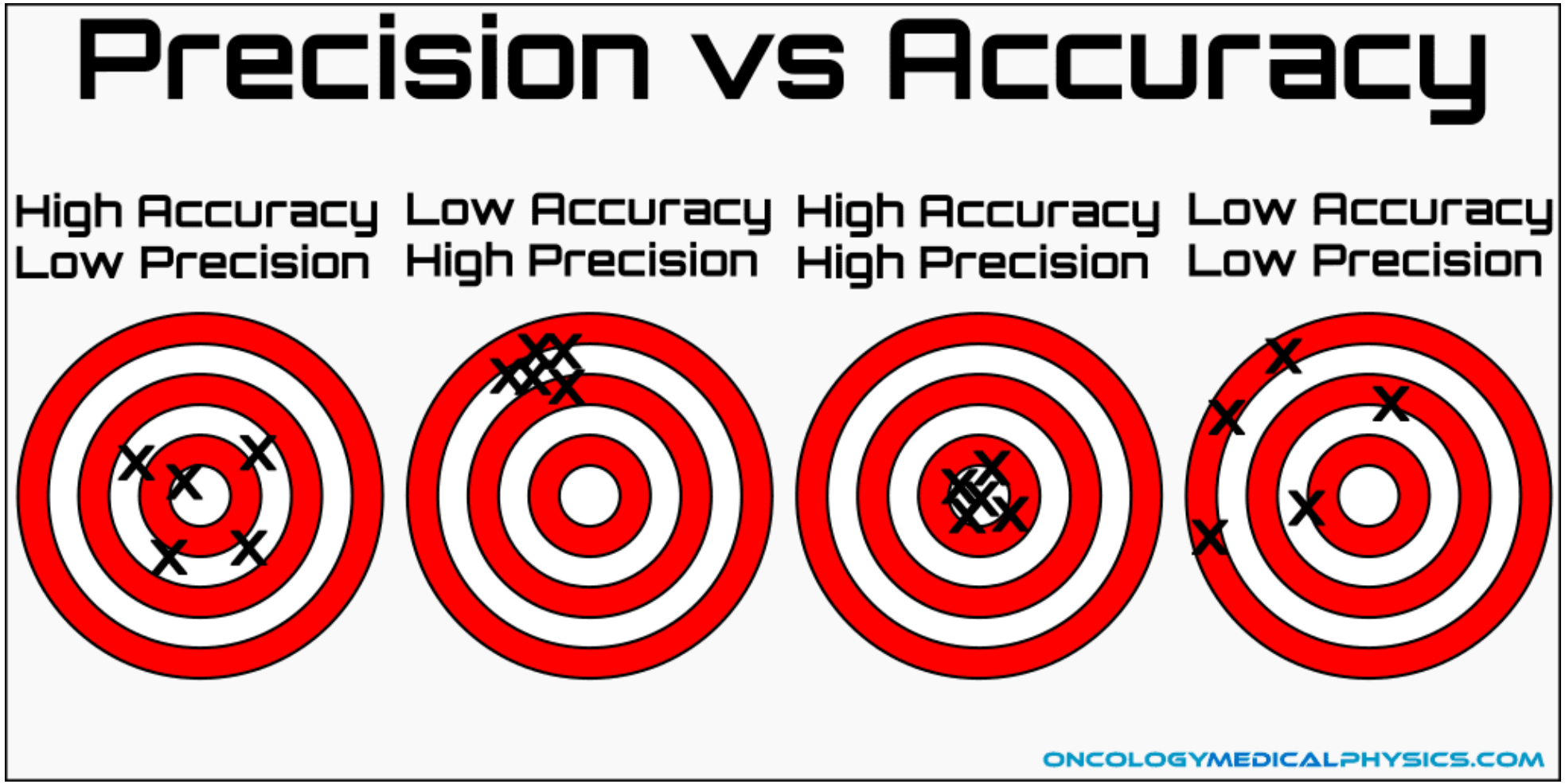
True class →	Pos	Neg
Yes	200	100
No	300	400
	P=500	N=500

True class →	Pos	Neg
Yes	400	300
No	100	200
	P=500	N=500

- Both classifiers gives 60% accuracy.
- They exhibit very different behaviors:
  - ✓ **On the left:** weak positive recognition rate/strong negative recognition rate
  - ✓ **On the right:** strong positive recognition rate/weak negative recognition rate

Is accuracy adequate measure?

- **Accuracy may not be useful measure in cases where**
  - there is a large class skew
    - ✓ Is 98% accuracy good if 97% of the instances are negative?
  - there are differential misclassification costs – say, getting a positive wrong costs more than getting a negative wrong.
    - ✓ Consider a medical domain in which a false positive results in an extraneous test but a false negative results in a failure to treat a disease
  - we are most interested in a subset of high-confidence predictions


Precision vs Accuracy
High Accuracy
Low Precision
Low Accuracy
High Precision
High Accuracy
High Precision
Low Accuracy
Low Precision
ONCOLOGYMEDICALPHYSICS.COM

3. **Recall (Sensitivity or True Positive Rate)**: The ratio of correctly predicted positive observations to all observations in the actual positive class.

$$\text{Recall} = \frac{TP}{TP + FN}$$

The above equation can be explained by saying, from all the positive classes, how many we predicted correctly.

	Actually Positive (1)	Actually Negative (0)
Predicted Positive (1)	True Positives (TPs)	False Positives (FPs)
Predicted Negative (0)	False Negatives (FNs)	True Negatives (TNs)

4. **F1 Score**: The harmonic mean of precision and recall. It provides a balance between precision and recall, especially when the dataset is imbalanced.

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

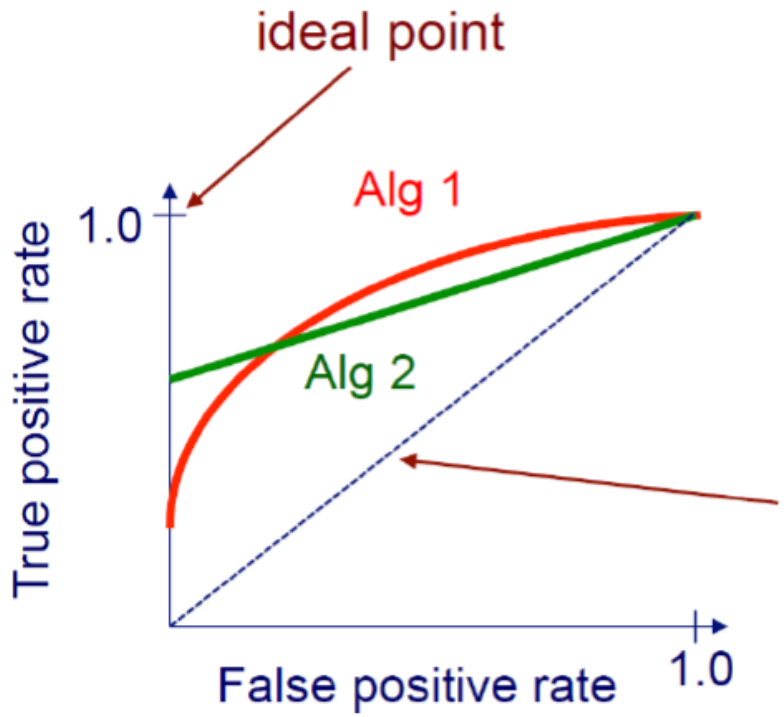
$$\text{Harmonic Mean} = \frac{2 \times a \times b}{a + b}$$

5. **Specificity (True Negative Rate)**: The ratio of correctly predicted negative observations to all observations in the actual negative class.

$$\text{Specificity} = \frac{TN}{TN + FP}$$

ROC/AUC

- A Receiver Operating Characteristic (ROC)/Area Under Curve plots the TP-rate vs. the FP-rate as a threshold on the confidence of an instance being positive is varied.

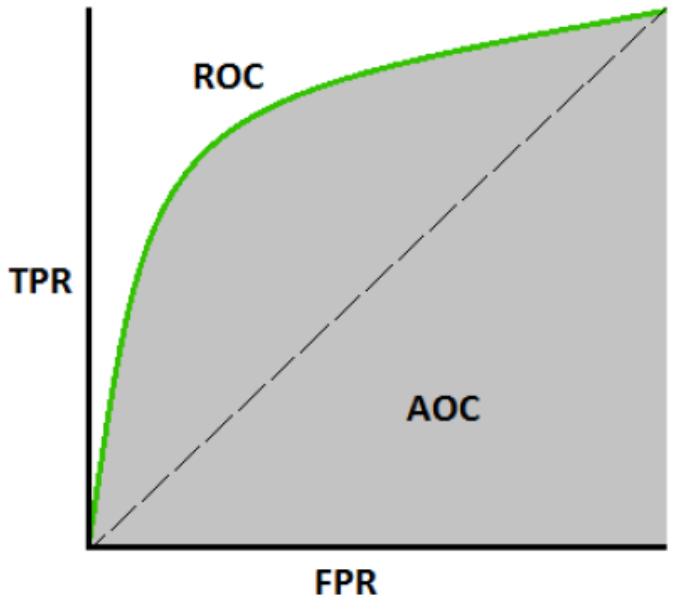


Different methods can work better in different parts of ROC space. This depends on cost of false + vs. false -

expected curve for random guessing

Area Under the Receiver Operating Characteristics

- AUC-ROC curve measure the performance at various **threshold settings**.
- ROC is a probability curve and AUC represents the degree or **measure of separability**.
- AUC tells the model **capability of distinguishing between classes**.
- **Higher AUC**, the better the model is at predicting 0 classes as 0 and 1 classes as 1.
- The ROC curve is plotted between TPR & FPR, where TPR is on the y-axis and FPR is on the x-axis.



$$\text{TPR /Recall / Sensitivity} = \frac{TP}{TP + FN}$$

$$\text{FPR} = 1 - \text{Specificity}$$

$$= \frac{FP}{TN + FP}$$

Create ROC of a model

- Consider a prediction table at different threshold setting

y labelled Value (0-Negative, 1-Positive)	$\hat{y}$ predicted value	Output at threshold (0.5)	Output at threshold (0.6)	Output at threshold (0.72)	Output at threshold (0.8)
0	0.3	0	0	0	0
1	0.55	1	0	0	0
0	0.75	1	1	1	0
1	0.8	1	1	1	1
0	0.4	0	0	0	0
1	0.7	1	1	0	0

Threshold Setting (0.5)	TP=3	FP=1	TN=2	FN=0	TPR=3/(3+0)=1	FPR=2/(2+1)=.66

Create ROC of a model…

- Threshold setting (0.6)

y labelled Value (0-Negative, 1-Positive)	$\hat{y}$ predicted value	Output at threshold (0.5)	Output at threshold (0.6)	Output at threshold (0.72)	Output at threshold (0.8)
0	0.3	0	0	0	0
1	0.55	1	0	0	0
0	0.75	1	1	1	0
1	0.8	1	1	1	1
0	0.4	0	0	0	0
1	0.7	1	1	0	0

Threshold Setting (0.6)	TP=2	FP=1	TN=2	FN=1	TPR=2/(2+1)=.66	FPR=1/(1+2)=.66

Create ROC of a model…

- Threshold setting (0.72)

y labelled Value (0-Negative, 1-Positive)	$\hat{y}$ predicted value	Output at threshold (0.5)	Output at threshold (0.6)	Output at threshold (0.72)	Output at threshold (0.8)
0	0.3	0	0	0	0
1	0.55	1	0	0	0
0	0.75	1	1	1	0
1	0.8	1	1	1	1
0	0.4	0	0	0	0
1	0.7	1	1	0	0

Threshold Setting (0.72)	TP=1	FP=1	TN=2	FN=2	TPR=1/(1+2)=.33	FPR=1/(1+2)=.33

Create ROC of a model…

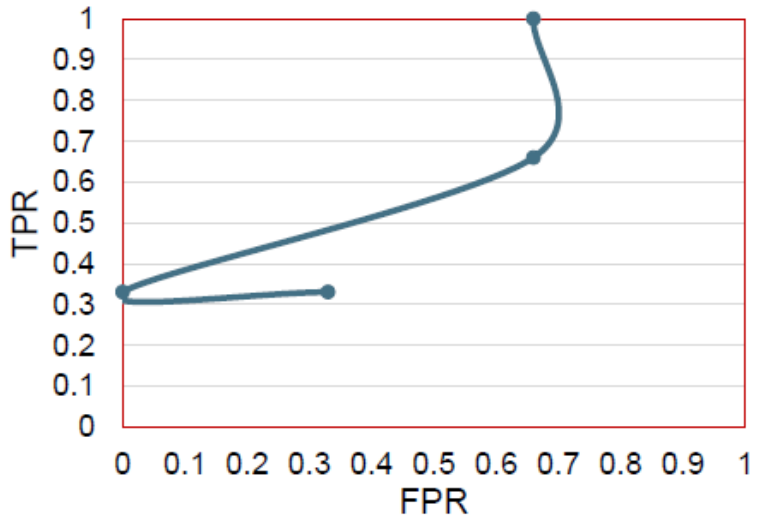
- Threshold setting (0.80)

y labelled Value (0-Negative, 1-Positive)	$\hat{y}$ predicted value	Output at threshold (0.5)	Output at threshold (0.6)	Output at threshold (0.72)	Output at threshold (0.8)
0	0.3	0	0	0	0
1	0.55	1	0	0	0
0	0.75	1	1	1	0
1	0.8	1	1	1	1
0	0.4	0	0	0	0
1	0.7	1	1	0	0

Threshold Setting (0.80	TP=1	FP=0	TN=3	FN=2	TPR=1/(1+2)=.33	FPR=0

Plot of ROC

Threshold Setting (0.5)	TP=3	FP=1	TN=2	FN=0	TPR=3/(3+0)=1	FPR=2/(2+1)=.66
Threshold Setting (0.6)	TP=2	FP=1	TN=2	FN=1	TPR=2/(2+1)=.66	FPR=1/(1+2)=.66
Threshold Setting (0.72)	TP=1	FP=1	TN=2	FN=2	TPR=1/(1+2)=.33	FPR=1/(1+2)=.33
Threshold Setting (0.80	TP=1	FP=0	TN=3	FN=2	TPR=1/(1+2)=.33	FPR=0

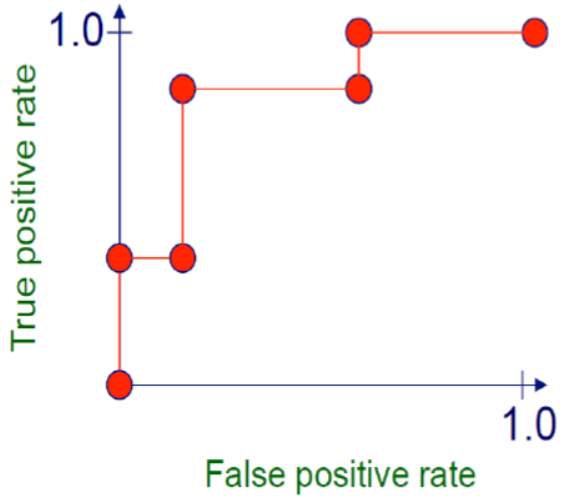


Step to create ROC

- Sort test-set predictions according to confidence that each instance is positive.
- Step through sorted list from high to low confidence
  - ✓ locate a *threshold* between instances with opposite classes (keeping instances with the same confidence value on the same side of threshold)
  - ✓ compute TPR, FPR for instances above threshold
  - ✓ output (FPR, TPR) coordinate

Example of ROC Plot

instance	confidence positive		correct class
Ex 9	.99		+
Ex 7	.98	TPR= 2/5, FPR= 0/5	+
Ex 1	.72	TPR= 2/5, FPR= 1/5	-
Ex 2	.70		+
Ex 6	.65	TPR= 4/5, FPR= 1/5	+
Ex 10	.51		-
Ex 3	.39	TPR= 4/5, FPR= 3/5	-
Ex 5	.24	TPR= 5/5, FPR= 3/5	+
Ex 4	.11		-
Ex 8	.01	TPR= 5/5, FPR= 5/5	-



Example of ROC Plot …

- **Rearrange the samples according to class**

Correct class	Instance	Confidence Positive
+	Ex 9	0.99
+	Ex 7	0.98
+	Ex 2	0.70
+	Ex 6	0.65
+	Ex 5	0.24
-	Ex 1	0.72
-	Ex10	0.51
-	Ex 3	0.39
-	Ex 4	0.11
-	Ex 8	0.01

Positive Class

Negative Class

Example of ROC Plot …

- **For Threshold 0.72**

Correct class	Instance	confidence positive	predicted class
+	Ex 9	0.99	+
+	Ex 7	0.98	+
+	Ex 2	0.70	-
+	Ex 6	0.65	-
+	Ex 5	0.24	-
-	Ex 1	0.72	+
-	Ex10	0.51	-
-	Ex 3	0.39	-
-	Ex 4	0.11	-
-	Ex 8	0.01	-

Confidence > threshold
Positive class
Else
Negative class

TP=2
FP=1
TN=4
FN=3
TPR=TP/TP+FN=2/5
FPR=FP/FP+TN=1/5

Example of ROC Plot …

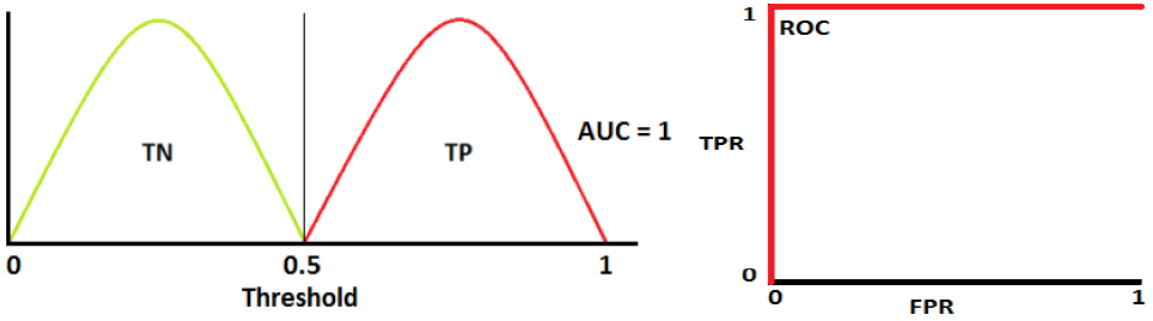
- **For Threshold 0.65**

Correct class	Instance	confidence positive	predicted class
+	Ex 9	0.99	+
+	Ex 7	0.98	+
+	Ex 2	0.70	+
+	Ex 6	0.65	+
+	Ex 5	0.24	-
-	Ex 1	0.72	+
-	Ex10	0.51	-
-	Ex 3	0.39	-
-	Ex 4	0.11	-
-	Ex 8	0.01	-

Confidence > threshold
Positive class
Else
Negative class

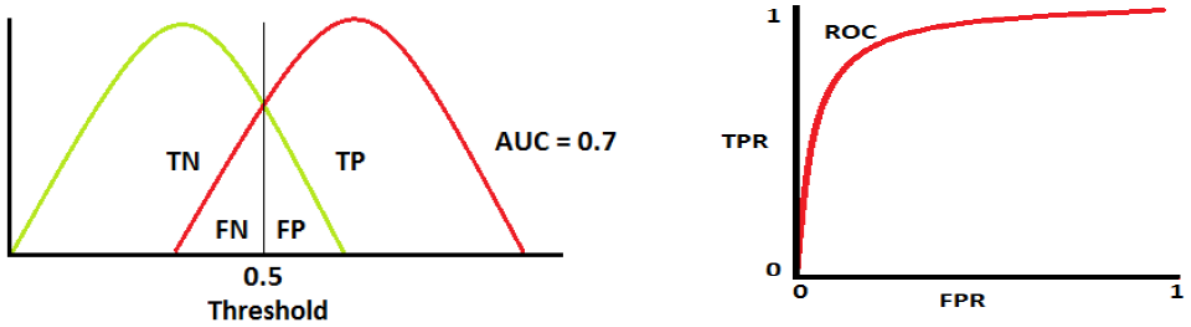
TP=4
FP=1
TN=4
FN=1
TPR=TP/TP+FN=4/5
FPR=FP/FP+TN=1/5

Significance of ROC



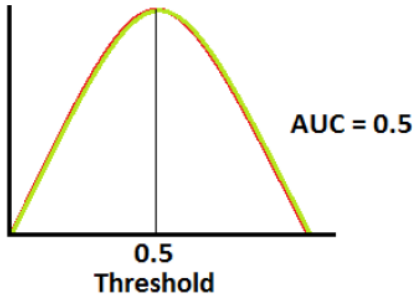
- This is an ideal situation, when two curves don't overlap at all, means model has an ideal measure of separability.
- It is perfectly able to distinguish between positive class and negative class.

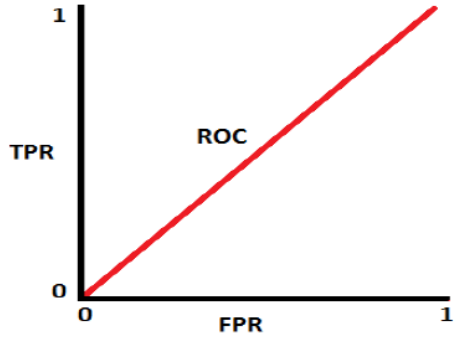
Significance of ROC…



- When two distributions overlap, then type 1 and type 2 errors are introduced.
- Depending upon the threshold, it can be minimized or maximized. When AUC is 0.7, it means there is a 70% chance that the model will be able to distinguish between positive class and negative class.

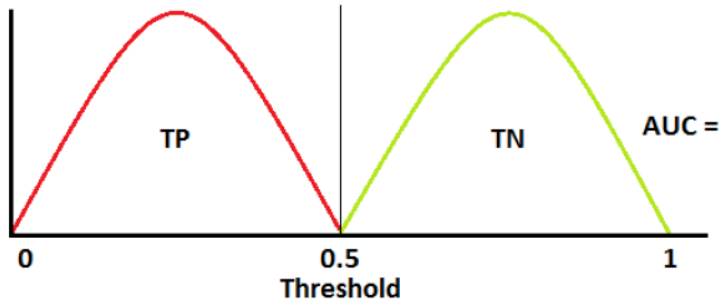
Significance of ROC…

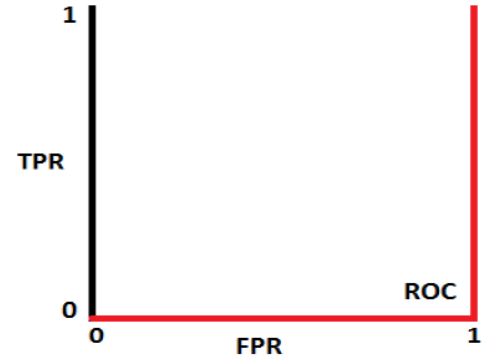




- This is the worst situation.
- When AUC is approximately 0.5, the model has no discrimination capacity to distinguish between positive class and negative class.

Significance of ROC…





- When AUC is approximately 0, the model is actually reciprocating the classes. It means the model is predicting a negative class as a positive class and vice versa.

TPR↑, FPR↑ and TPR↓, FPR↓

Steps to calculate AUC

Step 1: Sort Predictions by Probability

For a binary classifier, you'll have probabilities for each instance that it belongs to the positive class. Sort the dataset by predicted probability in descending order.

Step 2: Choose Thresholds

Vary the classification threshold from 0 to 1. For each threshold, classify instances as positive if their predicted probability is greater than or equal to the threshold, and negative otherwise. At each threshold, calculate TPR and FPR.

Step 3: Plot ROC Curve

For each threshold, plot the corresponding values of TPR (y-axis) and FPR (x-axis) to form the ROC curve.

Step 4: Compute AUC

The AUC is the area under the ROC curve. This can be computed using numerical methods, such as the **trapezoidal rule**, to approximate the area under the curve. The AUC value ranges from 0 to 1, where:

- **AUC = 1**: Perfect classifier (TPR = 1 and FPR = 0 for all thresholds).
- **AUC = 0.5**: Random guessing.
- **AUC < 0.5**: Worse than random guessing.

Where:

- **TPR (True Positive Rate)**: Measures the proportion of actual positives that are correctly identified.
- **FPR (False Positive Rate)**: Measures the proportion of actual negatives that are incorrectly identified as positives.

Formula for Trapezoidal Rule:

If you have multiple points (TPR1, FPR1), (TPR2, FPR2), ..., the AUC can be approximated as:

$$AUC = \sum \frac{(FPR_{i+1} - FPR_i) \times (TPR_{i+1} + TPR_i)}{2}$$

Where each interval between two consecutive points on the ROC curve forms a trapezoid, and you sum the areas of all these trapezoids.

Example Data:

Suppose we have the following predicted and actual values:

Actual	Predicted
1	0.9
1	0.8
0	0.6
1	0.4
0	0.3
0	0.2

Step 1: Sort the Predictions

We'll rank the predicted values in descending order, keeping track of the actual values:

Rank	Actual	Predicted
1	1	0.9
2	1	0.8
3	0	0.6
4	1	0.4
5	0	0.3
6	0	0.2

Step 2: Calculate TPR and FPR at each threshold

At each threshold, we will update the TPR and FPR based on the number of positive and negative instances classified correctly.

Let's define:

- $P = 3$ (total number of positives)
- $N = 3$ (total number of negatives)

Now, we'll calculate the TPR and FPR for each threshold:

Threshold	TPR	FPR
0.9	1/3	0/3
0.8	2/3	0/3
0.6	2/3	1/3
0.4	3/3	1/3
0.3	3/3	2/3
0.2	3/3	3/3

- At **threshold = 0.9**:
  - $TPR = 1/3, FPR = 0/3$
- At **threshold = 0.8**:
  - $TPR = 2/3, FPR = 0/3$
- At **threshold = 0.6**:
  - $TPR = 2/3, FPR = 1/3$
- At **threshold = 0.4**:
  - $TPR = 3/3, FPR = 1/3$
- At **threshold = 0.3**:
  - $TPR = 3/3, FPR = 2/3$
- At **threshold = 0.2**:
  - $TPR = 3/3, FPR = 3/3$

Step 3: Apply the Trapezoidal Rule to Calculate AUC

We now apply the formula for AUC:

$$AUC = \sum_{i=1}^{n-1} (FPR_{i+1} - FPR_i) \times (TPR_{i+1} + TPR_i) \, /2$$

We'll calculate the AUC incrementally for each segment:

1. From FPR 0 to FPR 0 (first segment):

$$(0 - 0) \times \frac{1/3 + 2/3}{2} = 0$$

2. From FPR 0 to FPR 1/3 (second segment):

$$(1/3 - 0) \times \frac{2/3 + 2/3}{2} = (1/3) \times 2/3 = 2/9$$

3. From FPR 1/3 to FPR 1/3 (third segment):

$$(1/3 - 1/3) \times \frac{2/3 + 3/3}{2} = 0$$

4. From FPR 1/3 to FPR 2/3 (fourth segment):

$$(2/3 - 1/3) \times \frac{3/3 + 3/3}{2} = (1/3) \times 1 = 1/3$$

5. From FPR 2/3 to FPR 3/3 (fifth segment):

$$(3/3 - 2/3) \times \frac{3/3 + 3/3}{2} = (1/3) \times 1 = 1/3$$

Step 4: Sum the AUC Segments

$$AUC = 0 + 2/9 + 0 + 1/3 + 1/3 = 2/9 + 2/3 = \frac{2 + 6}{9} = \frac{8}{9} \approx 0.89$$

Conclusion:

The **AUC** for this example is approximately **0.89**. This indicates a strong ability of the model to distinguish between the positive and negative classes.

Another formula for AUC Calculation

2. **Formula for AUC Using Only Predicted and Actual Values**

There is a simplified way to calculate the AUC if you only have predicted and actual values by using the **Mann–Whitney U-statistic**, which is equivalent to the **AUC**. The formula for AUC can be written as:

$$\text{AUC} = \frac{\sum \text{Ranks of Positive Class} - \frac{n_p(n_p+1)}{2}}{n_p \cdot n_n}$$

Where:

- n_p is the number of positive instances.
- n_n is the number of negative instances.
- The ranks are assigned based on sorting the predicted values in descending order.

This formula computes the AUC by ranking the predicted values, calculating the relative ranking of positive examples compared to negative examples, and then using that information to estimate the AUC.

3. **Interpretation**

- **AUC = 1**: The model perfectly distinguishes between the positive and negative classes.
- **AUC = 0.5**: The model is no better than random guessing.
- **AUC < 0.5**: The model is worse than random guessing.

Example:

Suppose you have the following predicted and actual values:

Actual	Predicted
1	0.9
1	0.8
0	0.6
1	0.4
0	0.3
0	0.2

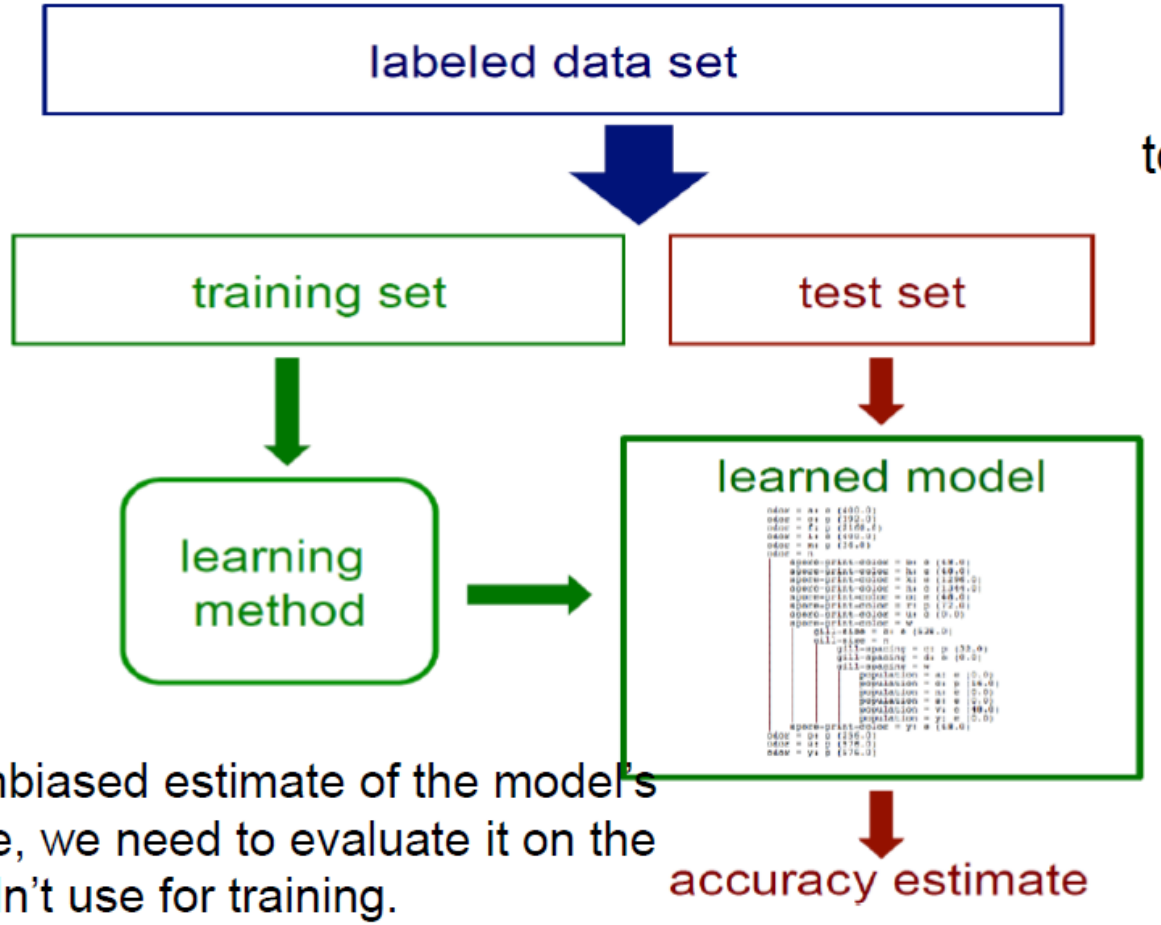
Step-by-Step Process:

1. Rank the predicted values:
   - Rank: $0.9(1), 0.8(2), 0.6(3), 0.4(4), 0.3(5), 0.2(6)$
2. Sum the ranks of positive instances:
   - Positive class ranks: $1 + 2 + 4 = 7$
3. Apply the formula:
   - $n_p = 3$ (number of positive examples), $n_n = 3$ (number of negative examples)
   - AUC = $\frac{7 - \frac{3(3+1)}{2}}{3 \cdot 3} = \frac{7-6}{9} = \frac{1}{9} = 0.89$

Thus, the AUC is **0.89**.

Experiment: Training and Testing

- Objective: Unbiased estimate of accuracy



Simplest way to split the data is to use the train-test split method

To get an unbiased estimate of the model's performance, we need to evaluate it on the data, we didn't use for training.

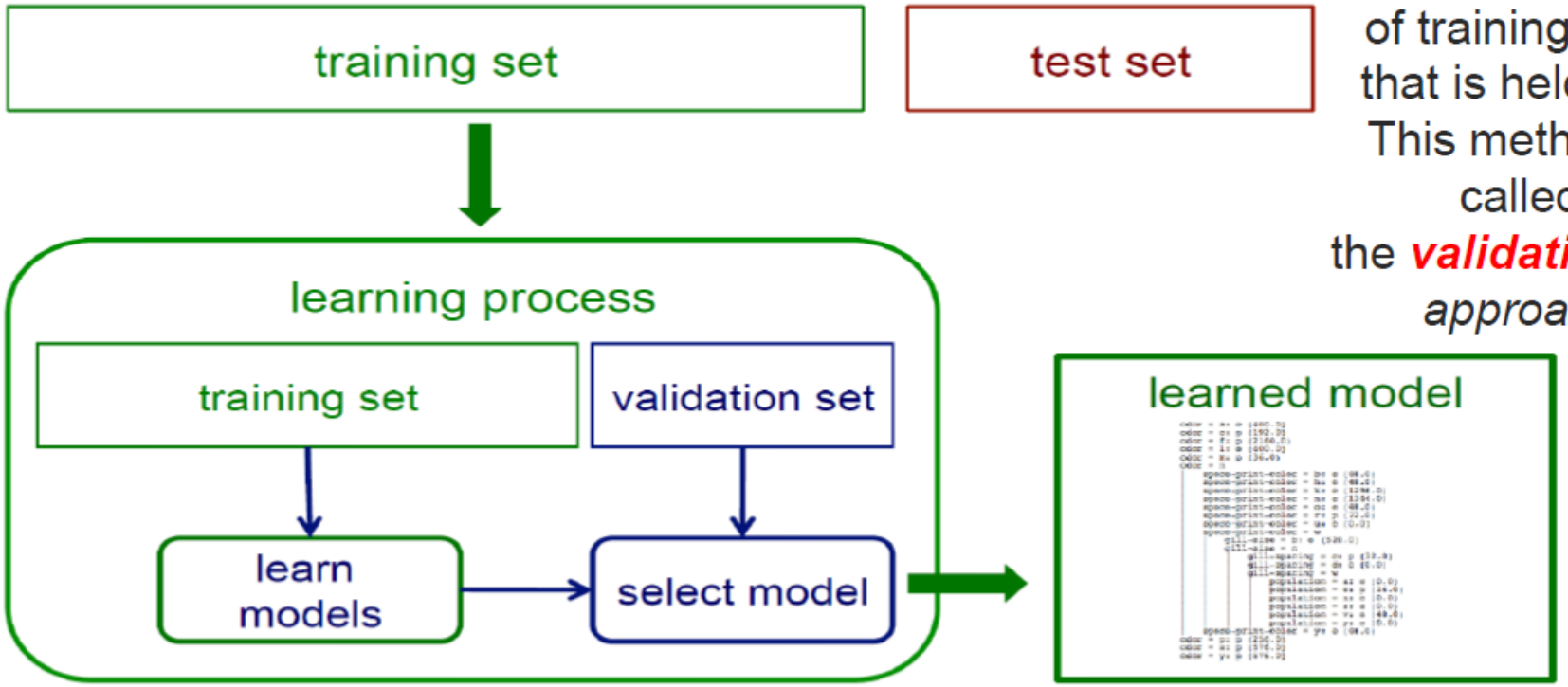
Experiment: Training and Testing…

- How can we get an unbiased estimate of the accuracy of a learned model?
  - ✓ when learning a model, you should pretend that you don't have the test data yet (it is "in the mail")*
  - ✓ if the test-set labels influence the learned model in any way, accuracy estimates will be biased
- * In some applications it is reasonable to assume that you have access to the feature vector (i.e. ***x***) but not the ***y*** part of each test instance

Validation (Tuning) Set

- Consider we want unbiased estimates of accuracy during the learning process (e.g. to choose the best level of decision-tree pruning)?

holding** out* a portion or subset of training data that is held out. This method is called the ***validation se *approach*



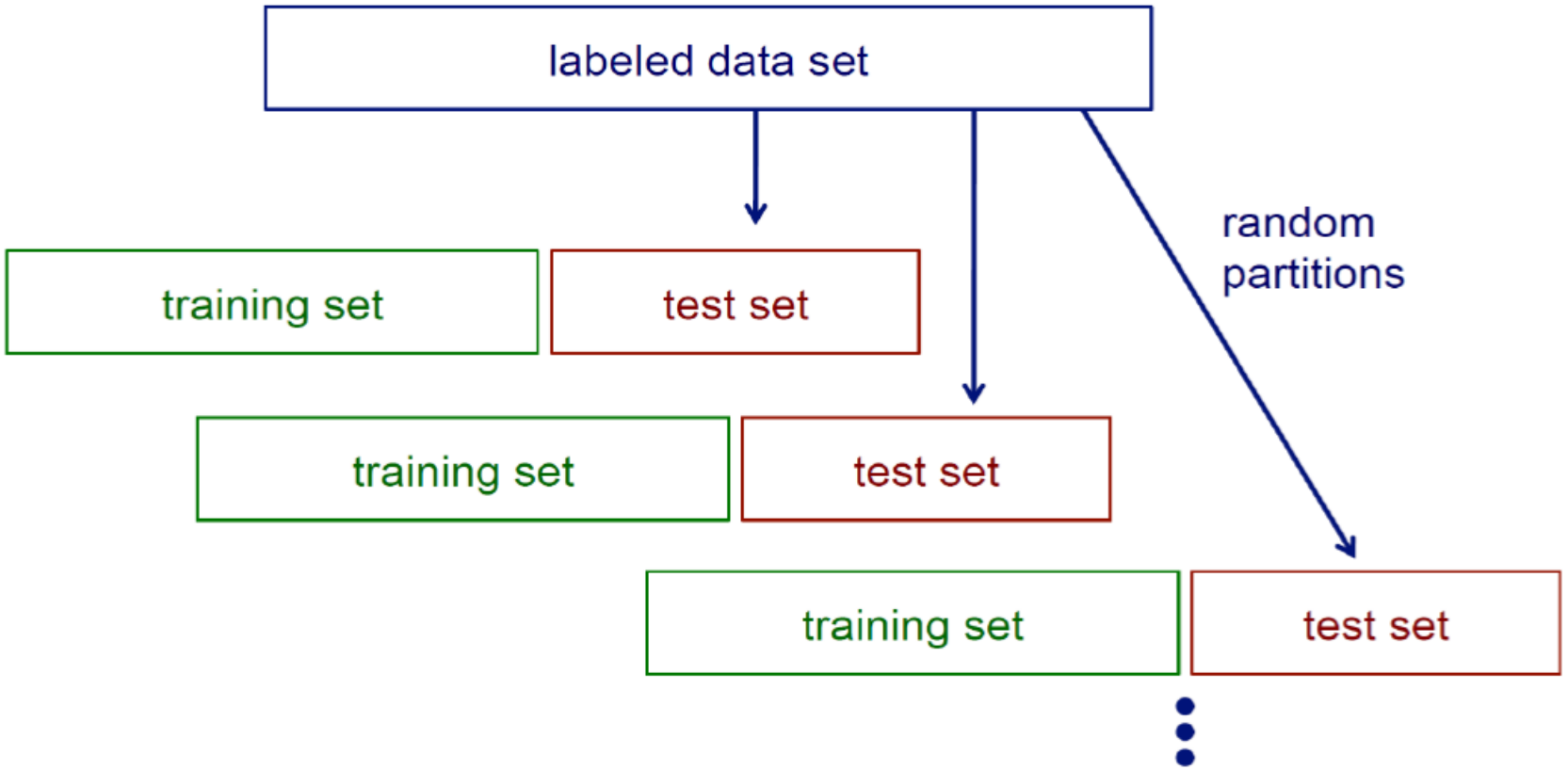
Partition training data into separate training/validation sets

Limitation of Single Training/Test Partition

- We may not have enough data to make sufficiently large
  - ✓ training and test sets a larger test set gives us more reliable estimate of accuracy (i.e. a lower variance estimate)
  - ✓ but… a larger training set will be more representative of how much data we actually have for learning process
- A single training set doesn't tell us how sensitive accuracy is to a particular training sample

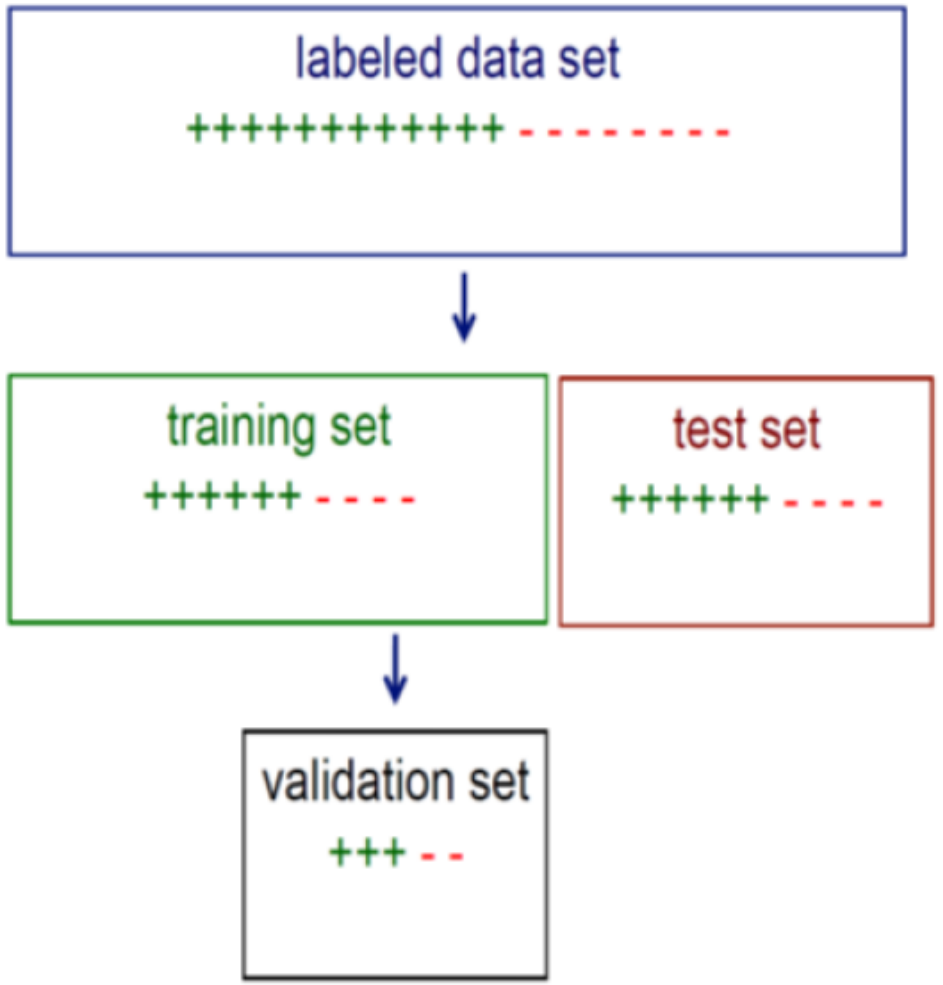
Random Sampling

- It can be addressed the second issue by repeatedly randomly partitioning the available data into training and set sets.



Random Sampling…

- When randomly selecting training or validation sets, we may want to ensure that class proportions are maintained in each selected set.
- This can be done via stratified sampling: first stratify (divider) instances by class, then randomly select instances from each class proportionally.



Cross Validation

- The train and test split has limitations such as the **dataset is small, the method is prone to high variance.**
- Due to the random partition, the results can be entirely different for different test sets because in some partitions, samples that are easy to classify get into the test set, while in others, the test set receives the 'difficult' ones.
- **To deal with this issue, we use cross-validation to evaluate the performance of a machine learning model.**

Cross Validation…

- **K-Fold Cross Validation**
  - In k-fold CV, we first divide our dataset into k equally sized subsets. Then, we repeat the train-test method **k times** such that each time one of the **k subsets** is used as a **test set** and the rest **k-1** subsets are used together as a training set.
  - Finally, we compute the estimate of the model's performance estimate by averaging the scores over the k trials.

K-Fold Cross Validation

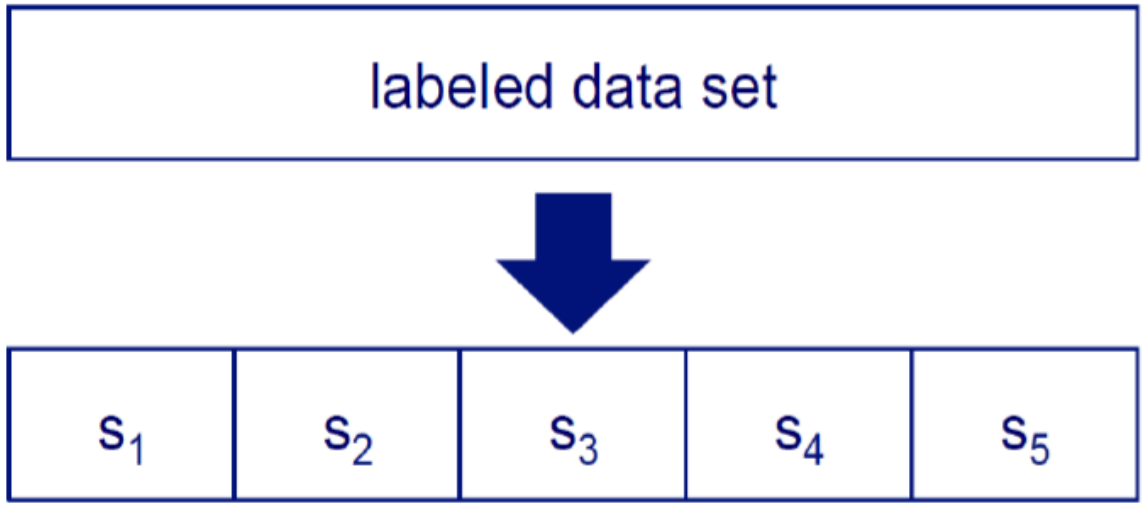
- **Example of 3-fold**

$$Overall\ Score = \frac{Score_1 + Score_2 + Score_3}{3}$$

- For example, let's suppose that we have a dataset $S = \{x_1, x_2, x_3, x_4, x_5, x_6\}$,
- First we divide the samples in to 3-fold as $S_1 = \{x_1, x_2), S_2 = \{x_3, x_4\}, S_3 = \{x_5, x_6\}$. Then we evaluate the model as

5-fold Cross Validation

Partition data into *n* subsamples (S)



Iteratively leave one subsample out for the test set, train on the rest

iteration	train on	test on
1	s_2 s_3 s_4 s_5	s_1
2	s_1 s_3 s_4 s_5	s_2
3	s_1 s_2 s_4 s_5	s_3
4	s_1 s_2 s_3 s_5	s_4
5	s_1 s_2 s_3 s_4	s_5

5-fold Cross Validation…

- Suppose we have 100 instances, and we want to estimate accuracy with cross validation.

iteration	train on	test on	correct
1	s_2 s_3 s_4 s_5	s_1	11 / 20
2	s_1 s_3 s_4 s_5	s_2	17 / 20
3	s_1 s_2 s_4 s_5	s_3	16 / 20
4	s_1 s_2 s_3 s_5	s_4	13 / 20
5	s_1 s_2 s_3 s_4	s_5	16 / 20

10-fold cross validation is common, but **smaller values** of *n* are often used when learning takes a **lot of time.**

accuracy = 73/100 = 73%

Leave-One-Out Cross-Validation

- LOOCV
  - we train our machine-learning model n times where n is dataset size.
  - Each time, only one sample is used as a test set while the rest are used to train our model.
  - **LOO on previous example** $S = \{x_1, x_2, x_3, x_4, x_5, x_6\}$,
  - $S_1 = \{x_1)$
  - $S_2 = \{x_2\}$
  - $S_3 = \{x_3\}$
  - $S_4 = \{x_4\}$
  - $S_5 = \{x_5\}$
  - $S_6 = \{x_6\}$

LOOCV...

$$Overall\ Score = \frac{Score_1 + Score_2 + Score_3 + Score_4 + Score_5 + Score_6}{6}$$

Cross Validation… Summary

- **When the size is small, LOOCV is more appropriate** since it will use more training samples in each iteration.
- Conversely, **we use k-fold cross-validation to train a model on a large dataset, which reduces the training time.**
- Using k-Fold Cross-Validation over LOOCV is one of the examples of Bias-Variance Trade-off. It reduces the variance shown by LOOCV and introduces some bias by holding out a substantially large validation set

All the Best, Prepare well