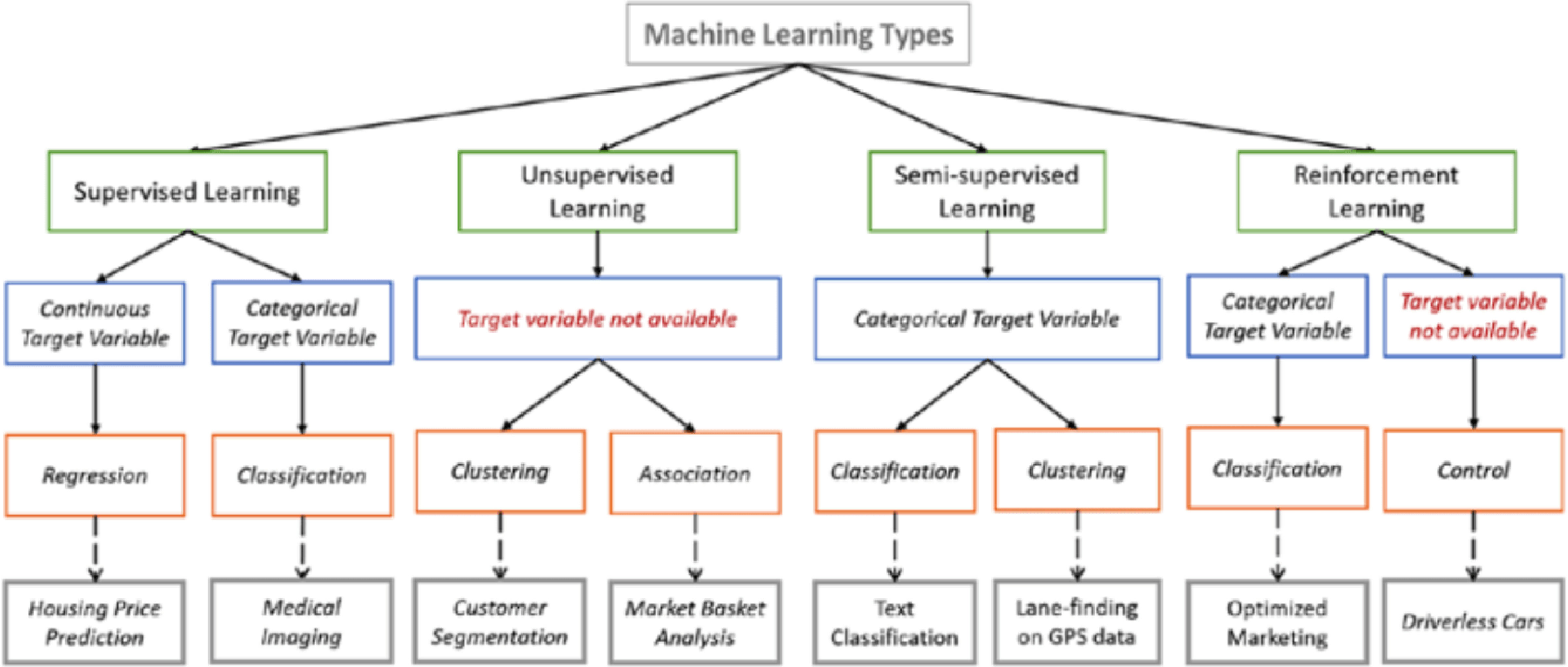
Supervised and Unsupervised Learning Algorithms

Surendra Panpaliya

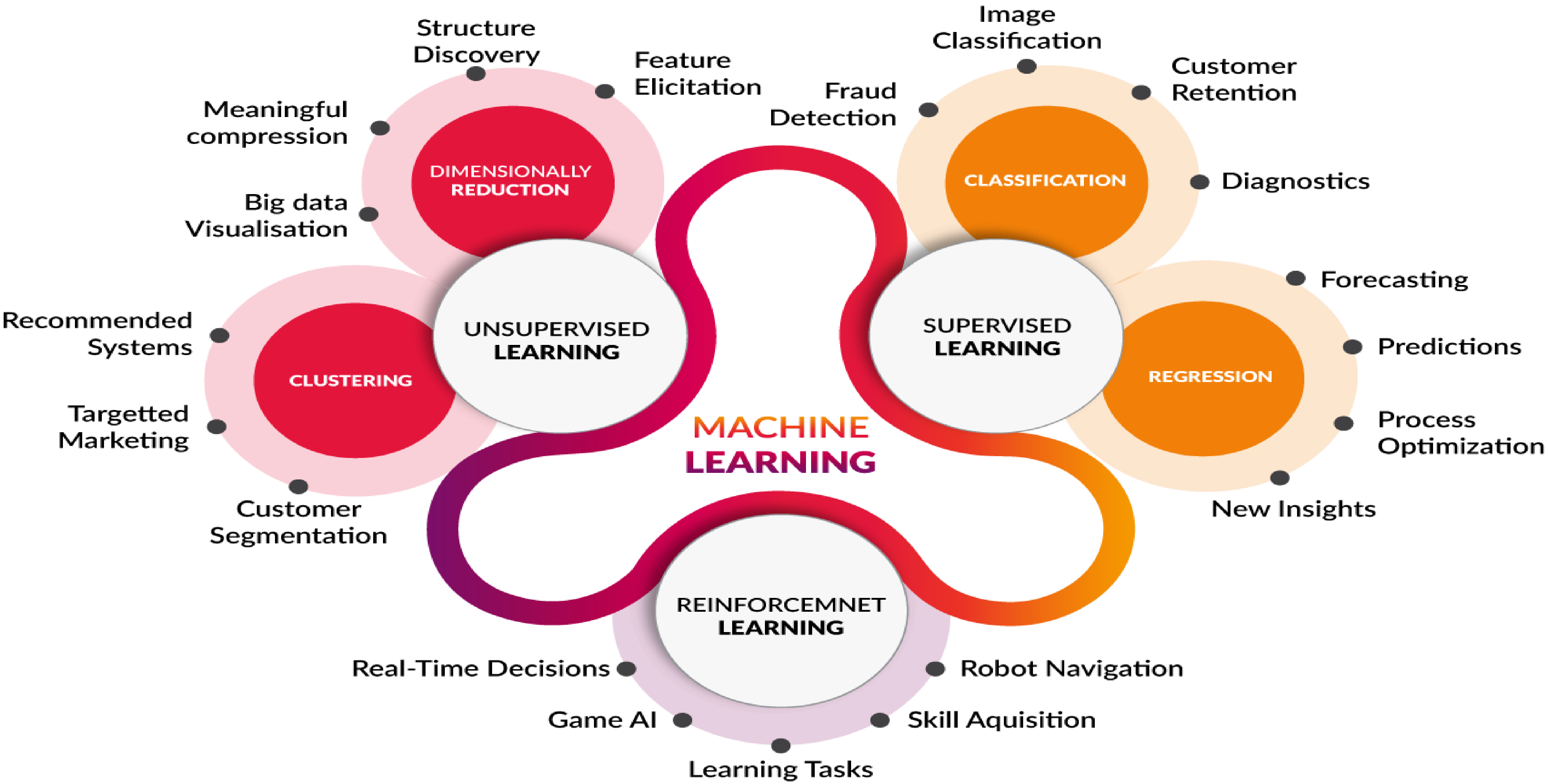
Agenda

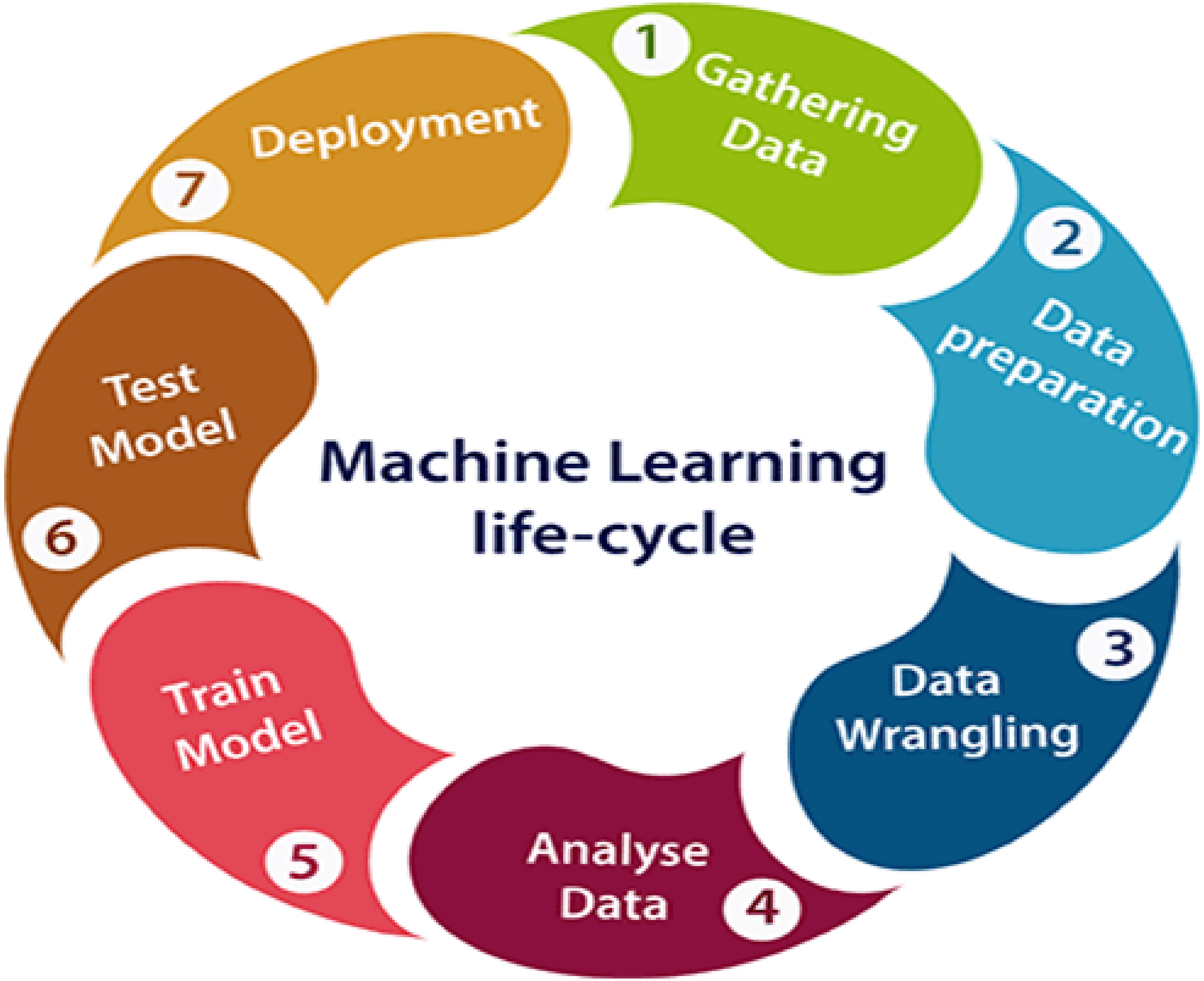
- Supervised vs Unsupervised
- Linear regression vs Logistic regression
- Decision Trees vs Random Forests
- K-means clustering vs Principal Component Analysis (PCA)

Types of Machine Learning



Machine learning for different type of business problem, regardless of industry.




Machine Learning life-cycle
1 Gathering Data
2 Data preparation
3 Data Wrangling
4 Analyse Data
5 Train Model
6 Test Model
7 Deployment

Supervised vs Unsupervised Learning

Aspect	Supervised Machine Learning	Unsupervised Machine Learning
Definition	The model is trained on labeled data, where each input is paired with a corresponding output label.	The model is trained on unlabeled data, and it tries to infer patterns or relationships within the dataset.

Supervised vs Unsupervised Learning

Aspect	Supervised Machine Learning	Unsupervised Machine Learning
Goal	To predict outcomes for new data based on learned patterns from labeled data.	To discover the underlying structure of data and group or categorize data based on similarities.

Supervised vs Unsupervised Learning

Aspect	Supervised Machine Learning	Unsupervised Machine Learning
Example Algorithms	Linear Regression, Logistic Regression, Decision Trees, Support Vector Machines, k-Nearest Neighbors (k-NN).	k-Means Clustering, Hierarchical Clustering, Principal Component Analysis (PCA), Association Rules.

Supervised vs Unsupervised Learning

Aspect	Supervised Machine Learning	Unsupervised Machine Learning
Use Cases	Spam detection, sentiment analysis, fraud detection, sales forecasting, object recognition.	Customer segmentation, anomaly detection, market basket analysis, dimensionality reduction.

Supervised vs Unsupervised Learning

Aspect	Supervised Machine Learning	Unsupervised Machine Learning
Output	Predicts a specific output value or class label.	Finds hidden patterns, relationships, or groupings in data.

Supervised vs Unsupervised Learning

Aspect	Supervised Machine Learning	Unsupervised Machine Learning
Data Requirement	Requires labeled data, which can be more resource-intensive to obtain.	Works with unlabeled data, which is often easier to collect and less costly.

Supervised vs Unsupervised Learning

Aspect	Supervised Machine Learning	Unsupervised Machine Learning
Evaluation	Easier to evaluate as there is a ground truth (labels) for comparison.	More challenging to evaluate due to the absence of predefined labels or a ground truth.

Summary

Supervised Learning

Used when you have known outcomes and

need to train the model

to predict similar results for new inputs.

Summary

UNSUPERVISED LEARNING

USEFUL WHEN YOU’RE EXPLORING

THE DATA WITHOUT PREDEFINED LABELS,

LOOKING TO DISCOVER PATTERNS,

CLUSTERS, OR INSIGHTS ON THE DATASET'S STRUCTURE.

Linear Regression vs Logistic Regression

Aspect	Linear Regression	Logistic Regression
Purpose	Used for predicting continuous numerical values.	Used for predicting categorical outcomes (usually binary).

Linear Regression vs Logistic Regression

Aspect	Linear Regression	Logistic Regression
Output	Produces a continuous output (e.g., price, temperature).	Produces a probability value, typically between 0 and 1, for classification.

Linear Regression vs Logistic Regression

Aspect	Linear Regression	Logistic Regression
Algorithm	Fits a line to minimize the sum of squared differences between predicted and actual values.	Uses the sigmoid function to map predicted values to a probability.

Linear Regression vs Logistic Regression

Aspect	Linear Regression	Logistic Regression
Equation	$Y=b0+b1X$	$P(Y=1)=1/1+e-(b0+b1X)$

$$Y = b0 + b1X + \epsilon.$$

Y - dependent variable

X - independent variable,

b0 - intercept,

b1 - coefficient (slope)

ϵ (epsilon) - error term.

sigmoid function

Maps the output to a value

Between 0 and 1

$P(Y=1)= 1 / 1+e-(b0+b1X)$

Linear Regression vs Logistic Regression

Aspect	Linear Regression	Logistic Regression
Type of Model	Regression (predicts a value along a continuum).	Classification (predicts discrete classes like 0 or 1).

Linear Regression vs Logistic Regression

Aspect	Linear Regression	Logistic Regression
Example Use Cases	Predicting house prices, sales forecasting, salary prediction.	Spam detection, disease diagnosis (e.g., predicting yes/no outcomes), credit scoring.

Linear Regression vs Logistic Regression

Aspect	Linear Regression	Logistic Regression
Assumptions	Assumes a linear relationship between input and output.	Assumes a relationship based on the log-odds of the outcome.

Linear Regression vs Logistic Regression

Aspect	Linear Regression	Logistic Regression
Error Measurement	Measured by Mean Squared Error (MSE) or Root Mean Squared Error (RMSE).	Measured by Log Loss or Cross-Entropy Loss.

Linear Regression vs Logistic Regression

Aspect	Linear Regression	Logistic Regression
Decision Boundary	No clear boundary; produces continuous values.	Has a distinct decision boundary at a probability threshold (often 0.5).

Decision Trees vs Random Forests

Aspect	Decision Tree Algorithms	Random Forest Algorithm
Definition	A single tree that splits data based on feature values to make predictions.	An ensemble of multiple decision trees that work together for a final prediction.

Decision Trees vs Random Forests

Aspect	Decision Tree Algorithms	Random Forest Algorithm
Structure	Single tree structure, with nodes and branches based on features.	Collection of decision trees, each trained on random subsets of data and features.

Decision Trees vs Random Forests

Aspect	Decision Tree Algorithms	Random Forest Algorithm
Training Process	Splits data based on features that best separate classes (classification) or minimize error (regression).	Trains multiple decision trees using bootstrapped (randomly sampled) data and randomly selected features.

Decision Trees vs Random Forests

Aspect	Decision Tree Algorithms	Random Forest Algorithm
Output	Direct prediction from a single decision path through the tree.	Aggregates predictions from all trees (majority vote for classification, average for regression).

Decision Trees vs Random Forests

Aspect	Decision Tree Algorithms	Random Forest Algorithm
Interpretability	Highly interpretable, easy to visualize, especially with small trees.	Less interpretable as it combines predictions from many trees.

Decision Trees vs Random Forests

Aspect	Decision Tree Algorithms	Random Forest Algorithm
Accuracy	Can be lower and highly dependent on the dataset; prone to overfitting.	Generally more accurate due to averaging, which reduces variance and stabilizes predictions.

Decision Trees vs Random Forests

Aspect	Decision Tree Algorithms	Random Forest Algorithm
Overfitting Tendency	Prone to overfitting, especially with deep trees on complex data.	Less likely to overfit due to ensemble approach and averaging.

Decision Trees vs Random Forests

Aspect	Decision Tree Algorithms	Random Forest Algorithm
Computational Complexity	Low computational cost, quick to train on small datasets.	Higher computational cost due to multiple trees; slower training but generally efficient in parallel processing.

Decision Trees vs Random Forests

Aspect	Decision Tree Algorithms	Random Forest Algorithm
Robustness to Noise	Sensitive to noise, can easily be influenced by outliers and small changes in data.	More robust to noise and outliers, as ensemble averaging reduces their effect.

Decision Trees vs Random Forests

Aspect	Decision Tree Algorithms	Random Forest Algorithm
Scalability	Suitable for small to medium datasets; can become inefficient with very large data.	Better suited for large datasets, as it handles high-dimensional data well with averaging across multiple trees.

Decision Trees vs Random Forests

Aspect	Decision Tree Algorithms	Random Forest Algorithm
Feature Selection	Considers all features at each split, making it more prone to biases.	Randomly selects subsets of features for each tree, reducing bias and improving model stability.

Decision Trees vs Random Forests

Aspect	Decision Tree Algorithms	Random Forest Algorithm
Use Cases	Interpretability-focused applications, basic decision-making processes, small datasets.	Complex applications like fraud detection, customer churn prediction, image classification, where accuracy is critical.

Decision Trees vs Random Forests

Aspect	Decision Tree Algorithms	Random Forest Algorithm
Feature Importance	Provides feature importance based on a single tree, which may be biased.	Aggregates feature importance across multiple trees, providing a more reliable estimate.

Decision Trees vs Random Forests

Aspect	Decision Tree Algorithms	Random Forest Algorithm
Example Algorithms	ID3, C4.5, CART (Classification and Regression Trees)	Ensemble method of decision trees; no specific algorithm for individual trees, often uses CART.

Summary

Decision Trees

Simple, interpretable, and

suitable for smaller datasets,

but prone to overfitting.

Summary

Random Forest

More complex

robust, accurate for larger datasets

less likely to overfit due

to ensemble structure.

Summary

Random Forest

outperforms Decision Trees in

predictive accuracy,

particularly for larger,

complex datasets.

Summary

DECISION TREES PROVIDE

EASE OF INTERPRETATION AND

FASTER TRAINING FOR SIMPLER PROBLEMS.

K-means clustering vs Principal Component Analysis (PCA)

Surendra Panpaliya

K-means clustering vs Principal Component Analysis (PCA)

Aspect	K-Means Clustering	Principal Component Analysis (PCA)
Purpose	Used for clustering, i.e., grouping data points into distinct clusters based on similarity.	Used for dimensionality reduction, i.e., reducing the number of features while retaining maximum variance.

K-means clustering vs Principal Component Analysis (PCA)

Aspect	K-Means Clustering	Principal Component Analysis (PCA)
Type of Technique	Unsupervised clustering technique.	Unsupervised dimensionality reduction technique.

K-means clustering vs Principal Component Analysis (PCA)

Aspect	K-Means Clustering	Principal Component Analysis (PCA)
Output	Assigns each data point to a cluster, resulting in groups of similar points.	Produces new uncorrelated features (principal components) that capture the maximum variance.

K-means clustering vs Principal Component Analysis (PCA)

Aspect	K-Means Clustering	Principal Component Analysis (PCA)
Algorithm	Iteratively assigns data points to the nearest cluster center (centroid) and updates centroids until convergence.	Computes eigenvectors and eigenvalues of the covariance matrix to find principal components.

K-means clustering vs Principal Component Analysis (PCA)

Aspect	K-Means Clustering	Principal Component Analysis (PCA)
Use Cases	Customer segmentation, document clustering, image compression, market basket analysis.	Reducing dimensionality for visualization, noise reduction, speeding up machine learning algorithms, feature extraction.

K-means clustering vs Principal Component Analysis (PCA)

Aspect	K-Means Clustering	Principal Component Analysis (PCA)
Interpretability	Results in distinct clusters, easy to interpret if clusters are well-separated.	Reduces data dimensions, which may reduce interpretability of individual principal components.

K-means clustering vs Principal Component Analysis (PCA)

Aspect	K-Means Clustering	Principal Component Analysis (PCA)
Assumptions	Assumes clusters are spherical and of similar size (sensitive to shape and size).	Assumes data is linearly correlated and maximizes variance along orthogonal axes.

K-means clustering vs Principal Component Analysis (PCA)

Aspect	K-Means Clustering	Principal Component Analysis (PCA)
Dependency on Scale	Sensitive to feature scale; standardizing features often improves clustering performance.	Highly sensitive to feature scale; standardization is necessary for meaningful principal components.

K-means clustering vs Principal Component Analysis (PCA)

Aspect	K-Means Clustering	Principal Component Analysis (PCA)
Dimensionality Reduction	Does not reduce dimensions but assigns data points to clusters.	Reduces dimensions by transforming original features into principal components.

K-means clustering vs Principal Component Analysis (PCA)

Aspect	K-Means Clustering	Principal Component Analysis (PCA)
Feature Selection	Does not inherently reduce or select features; assigns data points to clusters based on all features.	Selects features by reducing data to a subset of principal components that capture the most variance.

K-means clustering vs Principal Component Analysis (PCA)

Aspect	K-Means Clustering	Principal Component Analysis (PCA)
Distance Metric	Commonly uses Euclidean distance to measure similarity between points and centroids.	Does not rely on a specific distance metric; instead, it uses variance to select principal components.

K-means clustering vs Principal Component Analysis (PCA)

Aspect	K-Means Clustering	Principal Component Analysis (PCA)
Interpretation of Components	Produces discrete clusters that represent groups of similar data points.	Produces continuous components that represent the direction of maximum variance in data.

K-means clustering vs Principal Component Analysis (PCA)

Aspect	K-Means Clustering	Principal Component Analysis (PCA)
Visualization	Useful for visualizing groups in a scatter plot with assigned cluster colors.	Useful for visualizing high-dimensional data in 2D or 3D by projecting onto the principal components.

Summary

K-Means Clustering

Groups similar data points into clusters

based on distance from centroids,

used for identifying patterns or

groupings in data.

Summary

PCA

Reduces the number of features

by identifying new uncorrelated

principal components that retain

the maximum variance in data.

Agenda

Model Evaluation vs Validation

Techniques for cross-validation

Performance metrics Evaluating models

Model Evaluation vs Validation

Aspect	Model Evaluation	Model Validation
Purpose	Measures the performance of a trained model to assess its effectiveness on unseen data.	Confirms that the model's performance is consistent and generalizable on new data, typically by splitting data or using specific resampling methods.

Model Evaluation vs Validation

Aspect	Model Evaluation	Model Validation
Focus	Focuses on metrics that indicate the model’s accuracy, error, and performance on test data.	Focuses on verifying the model’s stability and ability to generalize well by evaluating it on multiple subsets of data.

Model Evaluation vs Validation

Aspect	Model Evaluation	Model Validation
Common Techniques	Accuracy, Precision, Recall, F1 Score, Confusion Matrix, ROC-AUC, Mean Squared Error, Mean Absolute Error.	Train-Test Split, K-Fold Cross-Validation, Stratified K-Fold, Leave-One-Out Cross-Validation, Hold-Out Validation.

Model Evaluation vs Validation

Aspect	Model Evaluation	Model Validation
Process	Evaluation is typically performed after training using a dedicated test set to measure final performance.	Validation occurs during model development to assess and improve the model’s ability to generalize, often using subsets of the training data.

Model Evaluation vs Validation

Aspect	Model Evaluation	Model Validation
Data Split Requirement	Requires a separate test set not used in training to ensure unbiased performance measurement.	Uses portions of the training data, usually through resampling or partitioning, to avoid data leakage and test for robustness.

Model Evaluation vs Validation

Aspect	Model Evaluation	Model Validation
Goal	To understand how well the model performs on real-world, unseen data.	To assess if the model's performance is consistent across different subsets of data, indicating good generalization.

Model Evaluation vs Validation

Aspect	Model Evaluation	Model Validation
Error Analysis	Analyzes errors on test data to identify weaknesses and potential areas for model improvement.	Helps tune hyperparameters, select features, and refine model architecture by revealing variance, bias, and overfitting tendencies.

Model Evaluation vs Validation

Aspect	Model Evaluation	Model Validation
Example Metrics	Classification: Accuracy, Precision, Recall, F1 Score, ROC-AUC; Regression: MSE, RMSE, R-squared, MAE.	Performance metrics are computed on multiple folds or subsets to ensure stable results (e.g., mean accuracy across K-Folds).

Model Evaluation vs Validation

Aspect	Model Evaluation	Model Validation
Hyperparameter Tuning	Not typically used for tuning; evaluation results reflect final model performance.	Often combined with techniques like K-Fold Cross-Validation to select optimal hyperparameters.

Model Evaluation vs Validation

Aspect	Model Evaluation	Model Validation
Reproducibility	Offers a single performance score for final assessment.	Provides insights across multiple iterations, making it easier to gauge model reliability and stability.

Model Evaluation vs Validation

Aspect	Model Evaluation	Model Validation
Use Cases	Final model assessment in production or reporting to understand expected performance.	Model development and tuning phases to ensure robustness, generalization, and prevent overfitting.

Summary

Model Evaluation: Measures final model performance on test data to assess real-world applicability.

Validation Techniques: Ensures the model generalizes well by testing on multiple data splits during training.

Summary

Validation techniques refine the model and

ensure consistency,

Evaluation metrics provide a final measure

of its expected real-world performance.

Techniques for Cross-Validation in Machine Learning Models

Surendra Panpaliya

Train-Test Split

How it works?

Splits data into

a training set and a testing set,

typically with an 80/20 or 70/30 ratio.

Train-Test Split

Example

A dataset with 100 observations.

An 80/20 split would

use 80 observations to train the model

20 to test it.

Train-Test Split

Limitation

Results depend on a single split,

so performance might vary

if the split changes.

K-Fold Cross-Validation

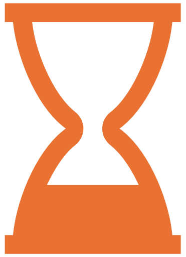
How it works

Divides the dataset into *k* equally-sized folds.

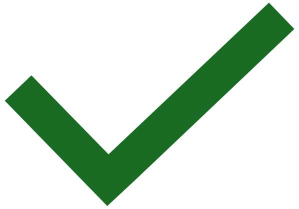
The model trains on *k-1* folds and

tests on the remaining fold.

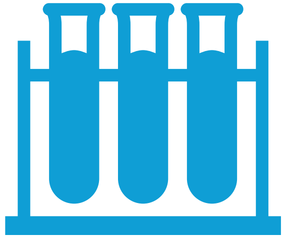
K-Fold Cross-Validation



This repeats *k* times,



using each fold



as the test set once.

K-Fold Cross-Validation

Example

For a dataset with

100 observations and k=5 folds,

each fold has 20 observations.

K-Fold Cross-Validation

The model trains on

80 observations and

tests on 20 in each round,

with results averaged across all 5 rounds.

K-Fold Cross-Validation

Advantage

Provides a more stable performance estimate.

Stratified K-Fold Cross-Validation

How it works?

Similar to K-Fold,

but each fold maintains

same class distribution as the full dataset,

Helpful for imbalanced datasets.

Stratified K-Fold Cross-Validation

Example

In a binary classification dataset with 100 observations,

80 of class A and 20 of class B,

each fold in k=5 would have

16 observations of class A and 4 of class B.

Stratified K-Fold Cross-Validation

Advantage

Ensures that each fold is

representative of the entire dataset,

improving reliability for imbalanced classes.

Leave-One-Out Cross-Validation (LOOCV)

How it works

Treats each observation as its own "fold."

The model trains on *n-1* data points and

tests on the one left out,

repeating this for all observations.

Leave-One-Out Cross-Validation (LOOCV)

EXAMPLE

FOR A DATASET OF 100 OBSERVATIONS,

THE MODEL TRAINS ON 99 AND

TESTS ON 1 IN EACH ITERATION,

REPEATING THIS PROCESS 100 TIMES.

Leave-One-Out Cross-Validation (LOOCV)

Advantage

Provides an unbiased estimate

But can be computationally intense

for large datasets.

Leave-P-Out Cross-Validation

How it works?

Similar to LOOCV

but leaves out *p* observations instead of 1.

The model trains on the remaining observations,

repeating for all possible combinations.

Leave-P-Out Cross-Validation

Example:

In a dataset of 100 observations, with $p=2$,

each iteration would leave out 2 observations

as the test set and use 98 to train.

Leave-P-Out Cross-Validation

All combinations of pairs

would be tested,

can be computationally demanding.

Conclusion

Each technique has unique strengths,

chosen based on dataset size,

class distribution, and

specific application requirements.

Performance Metrics for Evaluating Models in Embedded Applications

1. Accuracy

Measures the proportion

of correct predictions

out of total predictions.

1. Accuracy

Example

In an embedded device used for quality inspection

(e.g., identifying defective products),

if the model correctly identifies

90 out of 100 products, the accuracy is 90%.

2. Latency

THE TIME TAKEN BY THE MODEL

TO MAKE A PREDICTION,

KNOWN AS INFERENCE TIME.

2. Latency

EXAMPLE

FOR A REAL-TIME OBJECT DETECTION SYSTEM IN A DRONE,

LOW LATENCY IS CRUCIAL

TO REACT TO OBSTACLES INSTANTLY.

2. Latency

If it takes 100 milliseconds per prediction,

latency can determine whether

the drone's path is adjusted in time

to avoid collisions.

3. Memory Footprint

Measures the amount of memory

the model requires

to run on the embedded device.

3. Memory Footprint

Example

In a smart wearable device with limited memory,

a small memory footprint is essential.

3. Memory Footprint

For instance, if a model for

activity recognition requires only 1 MB,

it can run efficiently on wearables with 2 MB RAM,

leaving space for other processes.

4. Energy Consumption

The amount of power

the model consumes during inference,

crucial in battery-powered devices.

4. Energy Consumption

Example

In a smart home temperature sensor powered by batteries,

low energy consumption is essential

to extend battery life.

4. Energy Consumption

If a more optimized model consumes

10% less energy per prediction,

it can significantly prolong device usability.

5. Throughput

MEASURES THE NUMBER OF PREDICTIONS

A MODEL CAN MAKE PER SECOND.

5. Throughput

Example

For a network packet inspection model

embedded in a router,

high throughput is needed

to process hundreds of packets per second.

5. Throughput

If the model can inspect

200 packets per second,

it ensures smooth data flow

without lagging.

Summary and Conclusion

Each metric evaluates

different aspects of performance,

balancing accuracy,

efficiency, and usability

within the limited resources of

embedded applications.

Summary and Conclusion

The appropriate metric

depends on the device’s real-world

operational needs,

such as response time,

power efficiency, or

prediction accuracy.

Surendra Panpaliya

Founder and CEO

GKTCS Innovations

https://www.gktcs.com