Machine Learning

Documentation!!

1.Machine Learning

Introduction to Machine Learning

Machine learning (ML) is a branch of AI that enables algorithms to learn patterns from data and improve performance without explicit programming.

Types of Machine Learning

- **Supervised Learning**: Uses labeled data for regression (numeric prediction) and classification (categorical prediction).
- **Unsupervised Learning**: Finds patterns in unlabeled data (clustering, dimensionality reduction).
- **Reinforcement Learning**: Learns through trial and error to maximize rewards.

Key Algorithms

- **Classification**: KNN, Naive Bayes, Decision Trees/Random Forest, SVM, Logistic Regression.
- **Regression**: Linear Regression, SVR, Random Forest, Gaussian Processes, Ensemble Methods.
- **Clustering**: K-Means, Gaussian Mixtures, Hierarchical, Spectral Clustering, Boosting.

Generalization

- Ability of models to perform well on unseen data, not just training data.

Full ML Workflow

- Define problem
- Collect data
- Clean/prepare data
- Split (train/validation/test)
- Choose algorithm
- Set hyperparameters
- Train model
- Validate
- Tune hyperparameters
- Select best model
- Test final model
- Deploy for inference
- Monitor performance & data drift

2.Model Selection

Hyperparameters

- Defined as settings chosen *before* training that control how a model learns.

Types of Models

- **Classification models:** assign data to categories (e.g., A, B, C).
- **Regression models:** predict values on a numerical scale.

Model Selection Techniques

- **Grid Search** exhaustively tests all hyperparameter combinations.
- **Random Search:** samples combinations randomly.
- **Bayesian Optimization:** uses probability models to select promising hyperparameters.
- **Cross Validation:** evaluates models using multiple train-test splits for robustness.

3.Linear Regression

What Linear Regression Is

- A supervised machine-learning algorithm that learns from labeled data.
- Assumes a linear relationship between input (X) and output (Y).
- Represents this relationship using a straight-line equation:
- $y = mx + b$

Simple Linear Regression: Models the relationship between one independent variable (predictor) and one dependent variable (outcome).

Multiple Linear Regression: Extends the idea to multiple predictors.

Key Metrics

- **Coefficients (β)**: Show how much the dependent variable changes with a unit change in the predictor.
- **R-squared**: Proportion of variance in the dependent variable explained by the model.
- **Adjusted R-squared**: Corrects R-squared for multiple predictors.
- **p-values**: Test whether coefficients are statistically significant.
- **F-statistic**: Tests overall model significance

Assumptions

- Linearity: Relationship between predictors and outcome is linear.
- Independence: Observations are independent.
- Homoscedasticity: Constant variance of errors.
- Normality: Errors are normally distributed.

Applications

- **Prediction**: Forecasting outcomes (e.g., sales based on advertising spend).
- **Inference**: Understanding relationships (e.g., how education affects income).
- **Feature importance**: Identifying which variables most strongly influence outcomes.

3-1.Linear and Non-Linear Models in Machine Learning

Linear Models Models that assume a straight-line (linear) relationship between input features and the target.

Key Characteristics:

- Simple, fast, and easy to interpret
- Work well when data is linearly separable
- Low risk of overfitting
- Require fewer data points, Limited ability to capture complex patterns

Common Examples:

- Linear Regression
- Logistic Regression
- Ridge/Lasso Regression
- Linear SVM

When to Use:

- When interpretability matters, When the dataset is small or clean, When relationships between variables are roughly linear.

Non-Linear Models that can learn curved, complex, or irregular relationships between features and the target.

Key Characteristics:

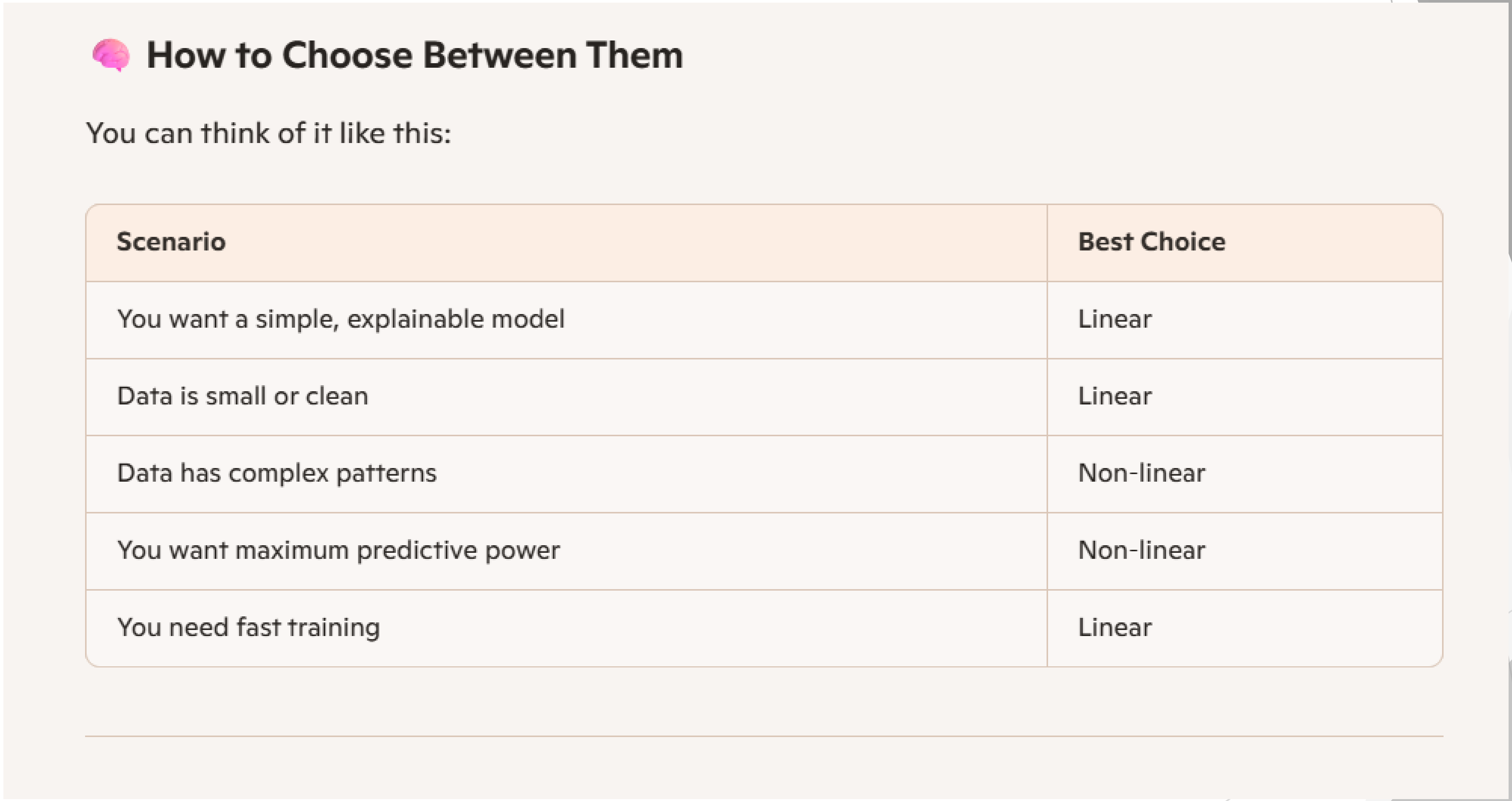
- Capture complex patterns and interactions
- More flexible and powerful
- Higher risk of overfitting
- Often require more data and tuning, Less interpretable

Common Examples:

- Decision Trees
- Random Forest
- Gradient Boosting (XGBoost, LightGBM, CatBoost), k-Nearest Neighbors, Neural Networks
- Non-linear SVM (RBF kernel)

When to Use:

- When data has complex, non-linear relationships
- When accuracy is more important than interpretability
- When you have enough data to avoid overfitting



🧠 How to Choose Between Them

You can think of it like this:

Scenario	Best Choice
You want a simple, explainable model	Linear
Data is small or clean	Linear
Data has complex patterns	Non-linear
You want maximum predictive power	Non-linear
You need fast training	Linear

4.Gradient Descent

What Gradient Descent Is A fundamental optimization technique that minimizes a model's cost function by iteratively adjusting parameters to reduce prediction error.

Steps of Gradient Descent:

- **Initialize parameters** (weights & bias) randomly
- **Compute predictions** using current parameters
- **Calculate loss** (MSE for regression, Log Loss for classification)
- **Compute gradients** of the loss w.r.t. parameters
- **Update parameters** using the learning rate
- **Repeat** for many iterations
- **Converge** to a minimum of the loss function

Types of Gradient Descent:

1. Batch Gradient Descent

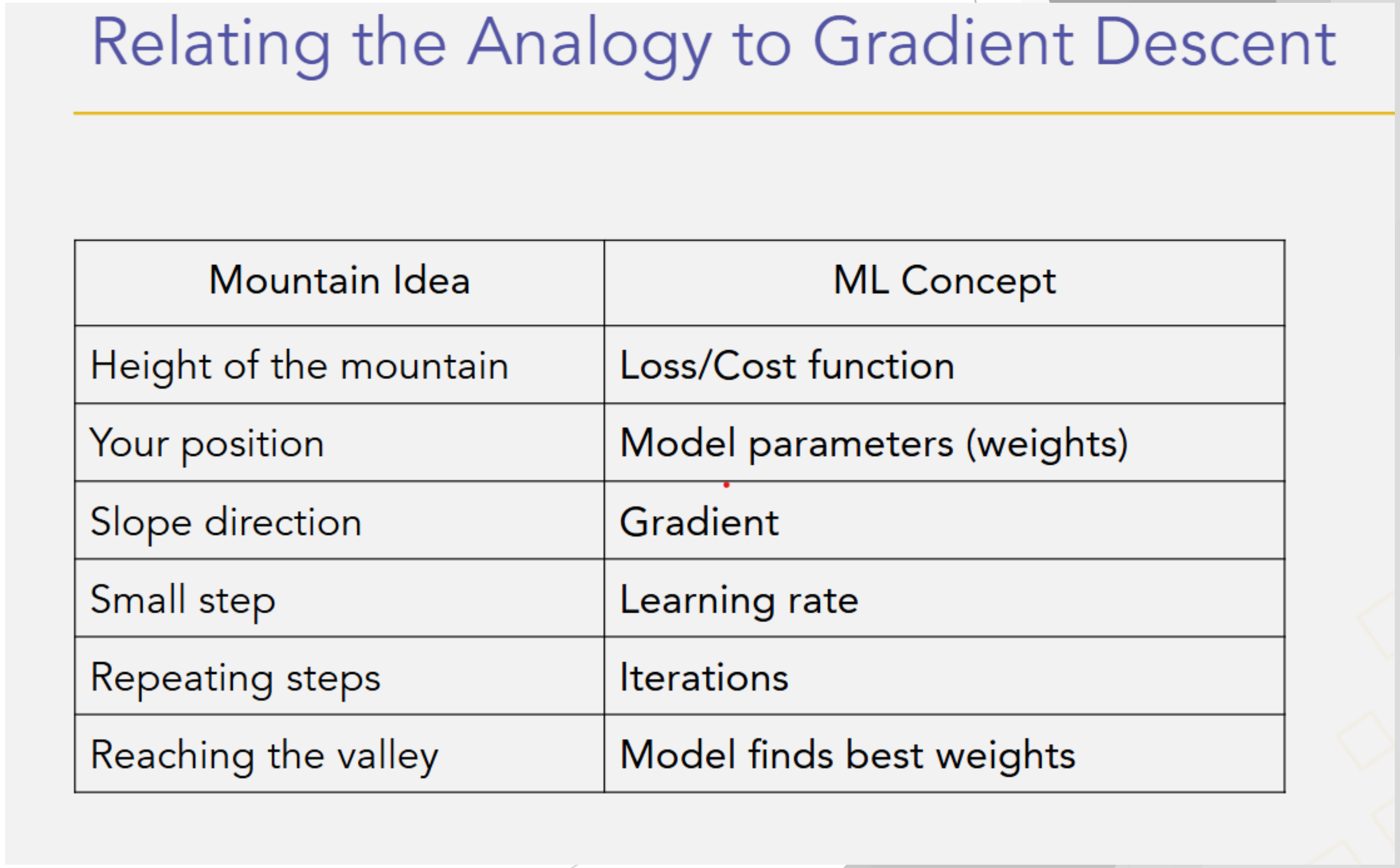
- Uses the entire dataset
- Very stable but slow
- Best for small/medium datasets

2. Stochastic Gradient Descent (SGD)

- Uses **one** data point per update
- Very fast, noisy updates
- Can escape local minima
- Good for large/streaming data

3. Mini-Batch Gradient Descent

- Uses small batches
- Balanced speed + stability
- Most common in practice



Relating the Analogy to Gradient Descent

Mountain Idea	ML Concept
Height of the mountain	Loss/Cost function
Your position	Model parameters (weights)
Slope direction	Gradient
Small step	Learning rate
Repeating steps	Iterations
Reaching the valley	Model finds best weights

5.Regularization

Generalization: The ability of the model to preform good on unseen data.

Regularization : is a technique used to **reduce overfitting** by adding a penalty for model complexity. This encourages simpler models that perform better on unseen data.

Overfitting : Overfitting happens when the model learns too much from the training data, including noise and outliers.

Underfitting: Underfitting happens when the model fails to learn important patterns.

How Regularization Controls Overfitting

- The cost function becomes: Total Cost = Loss Function + **λ** × Penalty Term
- The loss function measures training error.
- The penalty term discourages large weights.
- **λ** (lambda) controls how strong the penalty is.

Types of Regularization

- L2 (Ridge): Adds the squared magnitude of coefficients to the loss.
- L1 (Lasso): Adds the absolute magnitude of coefficients, which can shrink some weights to zero—useful for feature selection.

Bias–Variance Tradeoff

- **Bias:** Error from overly simple assumptions. High bias → underfitting.
- **Variance:** Sensitivity to small changes in training data. High variance → overfitting.
- The goal is to **balance bias and variance** for optimal generalization.

Model Complexity

- **Bias decreases**
- **Variance increases** And vice versa. Regularization helps find the sweet spot between the two.

6.Classification

Classification : Classification is defined as the process of using machine learning algorithms to categorize data into predefined classes or labels.

- Types of Classification
- The slides outline three major types:
- **Binary Classification** – Two possible outcomes (e.g., spam vs. not spam).
- **Multiclass Classification** – More than two classes, but each data point belongs to only one.
- **Multi-Label Classification** – A single data point can belong to multiple classes simultaneously.

- Key Characteristics of Classification Models
- The slides highlight several important aspects:
- Class separation
- Decision boundaries
- Sensitivity to data quality
- Handling imbalanced data
- Interpretability

Classification Algorithms

- The algorithms are grouped into linear and non-linear models:
- Linear Models
- Logistic Regression
- Linear SVM
- Single-layer Perceptron
- SGD Classifier
- Non-Linear Models
- K-Nearest Neighbors
- Kernel SVM
- Naive Bayes
- Decision Trees
- Ensemble Methods (Random Forest, AdaBoost)
- Multi-layer Neural Networks

7.Errors in ML

What Machine Learning Errors Are

- Any deviation from the expected model behavior.

Includes:

- **Classification errors**: wrong category predictions.
- **Regression errors**: inaccurate numerical predictions.
- **Outliers**: unusual data points that don’t fit the pattern.

Sources of Errors:

- **Poor data quality**: noise, missing values, wrong types, inconsistent formatting.
- **Model selection & hyperparameters**: choosing unsuitable algorithms or tuning without understanding.
- **Algorithmic issues**: overfitting, underfitting, excessive complexity.
- **Computational limits**: insufficient memory or processing power affecting training.

How to Detect Errors

- **Model interpretability tools**: feature importance, partial dependence plots.
- **Data visualization**: inspecting data and predictions to spot issues.

How to Mitigate Errors

- **Data cleaning**: preprocessing, validating, fixing quality issues.
- **Model refinement**: adjusting parameters or trying different models.
- **Regularization**: L1, L2, dropout to reduce overfitting.

8.KNN Example on iphone purchase record

Dataset Overview

- Features typically include Gender, Age, Salary.
- Target variable: Purchased iPhone (Yes/No).
- Goal: Predict whether a customer will buy an iPhone based on demographic features.

Required Libraries

- pandas, numpy for data handling
- matplotlib, seaborn for visualization
- scikit-learn for preprocessing, model building, and evaluation

Data Loading & Exploration

- Load data using pd.read_csv().
- Inspect structure with df.info(), df.describe(), df.head().
- Check for missing values.

Data Preparation

- Select features: X = df[['Age', 'Salary']].
- Select target: y = df['Purchased'].
- Split into training/testing sets using train_test_split().

Training the KNN Model

- Initialize model: KNeighborsClassifier(n_neighbors=5)
- Fit using training data.
- Predict using knn.predict(X_test).

Model Evaluation

- Use:
  - accuracy_score
  - confusion_matrix
  - classification_report

Hyperparameters & Tuning K

- Hyperparameters are settings chosen before training.
- For KNN, the key hyperparameter is k (number of neighbors).
- A loop tests k from 1 to 20.
- Plot accuracy vs. k to find the best value.
- Optimal K found: 7, with accuracy 0.9375.

Advantages of KNN

- Simple and intuitive.
- No explicit training phase.
- Works for both classification and regression.
- Few parameters to tune.

Disadvantages of KNN

- Slow with large datasets.
- Performance drops with many features.
- Sensitive to noisy or unscaled data.
- Can overfit in high-dimensional spaces.

9.Evaluation Metrics for Classification

Confusion Matrix

A 2×2 table containing four outcomes:

- True Positive (TP): Correctly predicted positives
- False Positive (FP): Incorrectly predicted positives
- True Negative (TN): Correctly predicted negatives
- False Negative (FN): Incorrectly predicted negatives

Accuracy

- Measures the proportion of correct predictions:
- Accuracy $= \frac{TP+TN}{TP+TN+FP+FN}$

Precision

- Indicates how many predicted positives were actually positive:
- Precision $= \frac{TP}{TP+FP}$

Recall

- Shows how many actual positives were correctly identified:
- Recall $= \frac{TP}{TP+FN}$

F1-Score

- A harmonic mean of precision and recall, useful when you need a balance:
- F1 $= 2 \cdot \frac{precision \cdot recall}{precision + recall}$

10.Logistic Regression

What Logistic Regression Is

- A widely used algorithm for **binary classification**.

Why Not Use Linear Regression for Classification

- Linear regression predicts **continuous values**, not probabilities.
- It can output values outside the range **0–1**, making it unsuitable for classification.

Logistic vs. Linear Regression

- **Linear Regression:** continuous outputs.
- **Logistic Regression:** discrete categories (classification).

Assumptions of Logistic Regression

- Independent observations
- Binary target variable
- Linear relationship between features and log-odds
- No significant outliers
- Large sample size

Sigmoid Function

- Maps any real number to a value between 0 and 1.
- Smooth, continuous, symmetric around 0.
- Converts model output into a probability.

Decision Boundary

- The line (or surface) separating predicted classes.
- Determined by model weights.

Binary Cross-Entropy Loss

- Measures how far predicted probabilities are from true labels.
- Good predictions → small loss
- Wrong & confident predictions → large loss

Loss Function vs. Metric

- Loss function: optimized during training (must be differentiable).
- Metrics: used for evaluation after training (accuracy, precision, etc.).

Gradient Descent Steps

- Initialize weights and bias
- Compute predictions
- Calculate BCE loss
- Compute gradients
- Update weights
- Repeat until convergence

11.Clustering

What Clustering Is

- Clustering is an **unsupervised machine learning technique** used to group similar data points without predefined labels. It helps uncover hidden patterns and natural structures in data.

Why Clustering Matters

- Identifies customer segments
- Detects anomalies
- Simplifies complex datasets
- Supports decision-making and personalization

Common Clustering Algorithms

- K-Means: Fast, simple, partitions data into *k* clusters
- Hierarchical Clustering: Builds a tree of clusters (dendrogram)
- DBSCAN: Finds clusters of arbitrary shape and detects noise
- Gaussian Mixture Models: Probabilistic, flexible cluster boundaries

How Clustering Works

- Choose features
- Measure similarity (distance metrics like Euclidean or Manhattan)
- Group points based on closeness
- Evaluate cluster quality (silhouette score, inertia, etc.)

Real-World Applications

- Customer segmentation in retail
- Image compression
- Document/topic grouping
- Fraud detection
- Market basket analysis.

12.K Means

What K-Means Is

- K-Means is an **unsupervised clustering algorithm** that groups data points into *K* clusters by minimizing the variance within each cluster.

How K-Means Works

- The algorithm follows an iterative cycle:
- Initialization: Choose *k* initial centroids randomly.
- Assignment: Assign each data point to the nearest centroid.
- Update: Recalculate centroids as the mean of assigned points.
- Repeat: Continue until assignments stop changing.
- Output: Final clusters and centroid positions.

Elbow Method

- A technique to choose the optimal number of clusters by plotting **WCSS (Within-Cluster Sum of Squares)** for different values of *k* and identifying the “elbow” point where improvement slows down.

13.Evaluation Metrics for Clustering

Clustering is unsupervised, so evaluation focuses on how well the algorithm groups data rather than accuracy.

internal Evaluation Metrics

- 1. Silhouette Score
- Measures how similar a point is to its own cluster vs. other clusters.
- Range: −1 to +1
- Interpretation:
  - Close to **+1** → strong, well-separated clusters
  - Around **0** → overlapping clusters
  - Negative → incorrect clustering

Davies–Bouldin Index (DBI)

- Evaluates cluster compactness and separation.
- Lower values are better.
- Simple rule: *clusters should be tight and far apart*.

14.Dimensionality Reduction

Why Dimensionality Reduction Matters

- Having too many features can hurt model performance due to noise, redundancy, and overfitting.
- Dimensionality reduction helps simplify datasets while keeping essential information.

What Dimensionality Reduction Is

- It reduces the number of features while preserving important structure in the data.
- Two main approaches:
  - Feature Selection: Keep the most relevant original features.
  - Feature Extraction: Create new features from transformations (e.g., PCA, SVD, LDA).

Principal Component Analysis (PCA)

- PCA transforms correlated variables into uncorrelated principal components (PCs).
- Each PC is a linear combination of original features.
- PCs are ordered:
  - PC1 captures the most variance.
  - PC2 captures the next most, and so on.
- PCA uses eigenvectors (directions) and eigenvalues (importance) from the covariance matrix.

Applications of PCA :

- **Data Visualization:** Reduce to 2D/3D for easier plotting.
- **Preprocessing:** Speed up training and improve generalization.
- **Feature Extraction:** Create meaningful, uncorrelated features.
- **Data Compression:** Store data efficiently with minimal loss.
- **Noise Reduction:** Remove low-variance components.

15.Support Vector Machines

What SVM Is: SVM is a **supervised machine learning algorithm** used for **classification and regression**, especially effective for **binary classification** tasks like spam detection or image classification.

Hard Margin vs Soft Margin:

- **Hard Margin**: Assumes data is perfectly separable; no misclassification allowed. (separation between classes)
- **Soft Margin**: Allows some misclassification using slack variables and hinge loss to improve generalization on real-world, noisy data.

Mathematical Foundation: The slides outline the linear hyperplane equation and how SVM optimizes margin maximization while balancing classification errors.

Non-Linear SVM & Kernels: When data isn't linearly separable, SVM uses **kernel functions** to map data into a higher-dimensional space where separation becomes possible. Examples include:

- Linear kernel
- Polynomial kernel
- Other kernels (implied)

Hyperplane : **best separating boundary.**