Unit 3

20 August 2025 13:43

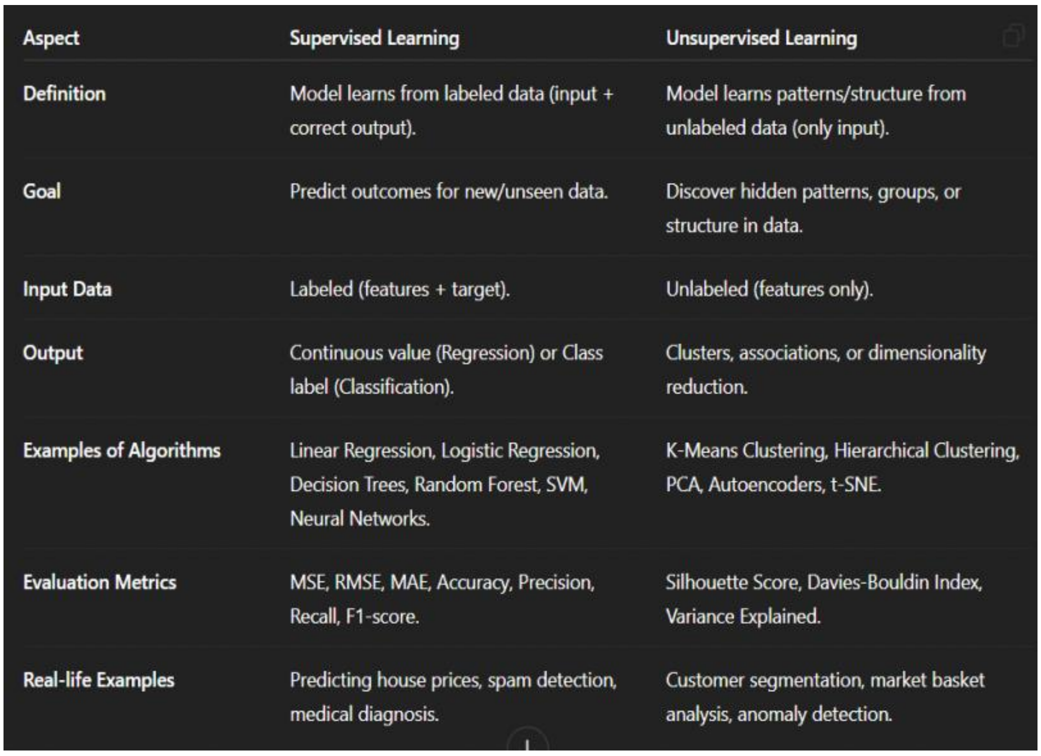
Supervised Learning Models

Learn from labelled data, predict output from input features.

- **Regression Models:** Predict continuous values.
  - Linear Regression
  - Multiple Linear Regression
  - Polynomial Regression
  - Support Vector Regression (SVR)
  - Decision Tree Regression
  - Random Forest Regression
  - K-Nearest Neighbours Regression (KNN Regression)

- **Classification Models**: Predict discrete classes.
  - Logistic Regression
  - K-Nearest Neighbours (KNN)
  - Decision Trees
  - Random Forest Classifier
  - Naive Bayes Classifier
    - Gaussian NB, Multinomial NB, Bernoulli NB
  - Support Vector Machine (SVM)

Supervised vs Unsupervised Learning



Aspect	Supervised Learning	Unsupervised Learning
Definition	Model learns from labeled data (input + correct output).	Model learns patterns/structure from unlabeled data (only input).
Goal	Predict outcomes for new/unseen data.	Discover hidden patterns, groups, or structure in data.
Input Data	Labeled (features + target).	Unlabeled (features only).
Output	Continuous value (Regression) or Class label (Classification).	Clusters, associations, or dimensionality reduction.
Examples of Algorithms	Linear Regression, Logistic Regression, Decision Trees, Random Forest, SVM, Neural Networks.	K-Means Clustering, Hierarchical Clustering, PCA, Autoencoders, t-SNE.
Evaluation Metrics	MSE, RMSE, MAE, Accuracy, Precision, Recall, F1-score.	Silhouette Score, Davies-Bouldin Index, Variance Explained.
Real-life Examples	Predicting house prices, spam detection, medical diagnosis.	Customer segmentation, market basket analysis, anomaly detection.

Classification Model

A supervised learning model that predicts discrete outputs (categories/labels) based on input features. Instead of predicting a number like in regression, it assigns the input data to a class. Ex: Logistic Regression, Decision Trees, Random Forest, SVM, Naïve Bayes , etc.

Steps in Classification Model

1. **Define Classes / Labels:** Identify distinct categories the model should predict.

2. **Encode Labels:** Convert categorical labels into numeric form that the model can use.

3. **Choose a Classification Algorithm:** Depends on problem type, dataset size, and feature types.

4. **Train the Model on Labelled Data:** The model learns patterns between features and class labels.
    - Special to classification: The loss function is often **cross-entropy** (for multi-class) instead of MSE.

5. **Evaluate Model Using Classification Metrics:** Accuracy, Precision, Recall, F1 Score, Confusion Matrix.

6. **Adjust Thresholds (Optional):** Many classifiers (like Logistic Regression) output probabilities. Threshold can be tuned for better precision or recall, depending on the problem.

7. **Handle Multi-Class or Imbalanced Data (Optional):** Multi-class: One-vs-Rest (OvR) or One-vs-One (OvO) strategies
    - Imbalanced data: Use resampling or class weights

Classification Algorithms

1. **Logistic Regression:**
   A classification algorithm used to predict a categorical output (usually binary) from one or more input features, which can be continuous or categorical.

   Key Points:
    - **Purpose:** Predict the probability of an event belonging to a class.
    - **Output:** A value between 0 and 1 representing probability.
    - **Decision Rule:** For binary, probability of over 0.5 is considered positive result. For multi-class, class with highest probability is chosen as positive.
    - **Key Feature:** Uses the sigmoid function for binary classification and softmax function for multi-class classification.
    - **Use Case:** Binary classification, need probability of different classes, small to medium dataset.
    - **Cons:** Not suitable for large, imbalance, non-linear or complex datasets, sensitive to multicollinearity.

2. **KNN:** K-Nearest Neighbours
   It is a lazy, instanced based algorithm which doesn't train using any function, rather it stores all the test samples in memo ry and for each new sample it looks into the memory to categorize the new sample based on its K-nearest neighbours.
   Types:
    - KNN Classification
    - KNN Regression

   Distance Metrics:
    - Euclidean
    - Manhattan
    - Minkowski

   Key Points:
    - **Purpose:** Predict the category for new data using distance metrics and majority vote.
    - **Output:** Class label.
    - **Decision Rule:** Majority vote.
    - **Key Features:** No training phase, sensitive to scale, K value selection is critical (less value -> noisy, large value -> smooth).
        - Noisy: Less value of K could make the model overly sensitive to outliers if test sample is nearby outliers.
        - Smooth: Large value of K result into misclassification due to domination of faraway points (Imbalanced dataset).
    - **Use Case:** Handwriting recognition, Recommendation system, Medical Diagnosis.
    - **Cons:** Computationally expensive at prediction time, memory overhead for large datasets, struggles with high dimensionality datasets (curse of dimensionality).
        - Curse of Dimensionality: All points become far from each other, making distance less meaningful.

3. **Decision Trees**
   Predicts outcomes by **s**plitting data into branches based on feature values, forming a tree-like structure of decisions. The split occurs based on information gain/impurity eliminated, feature with highest information gain is used for first split and so on.
   Splitting Criteria:
    - Gini (CART)
    - Entropy (Information Gain)
    - MSE (for regression trees)

   Key Points:
    - **Purpose:** To classify or predict outcome using hierarchical splits.
    - **Output:** Class label/Continuous value.
    - **Decision Rule:** Follow branches (if-else rules) until leaf node reached.
    - **Key Features:** Supports non-linear relationships, easy to visualize and interpret, handles both cat and con features.
    - **Use Case:** Loan approval, credit card approval, medical diagnosis.
    - **Cons:** Unstable, biased towards features with more depth, less accurate than random forest.

4. **Random Forest**
Creates multiple decision trees using random subsets of rows, each subset tree prioritizing different set of randomly selecte d features for split at each node. Each tree gives its own result then majority vote is used to decide the outcome.
Splitting Criteria:
    - Gini
    - Entropy
    - MSE

Key Points:
    - **Purpose:** Reduce overfitting, reduce bias, improve prediction stability over decision trees.
    - **Output:** Majority Vote for classification and Average for regression.
    - **Decision Rule:** Majority Vote or Average.
    - **Key Features:** Bootstrapped dataset, random subset of features at each split, reduces bias and variance.
    - **Use Case:** Loan approval, medical diagnosis, feature importance ranking.
    - **Cons:**
        - Less interpretable or black box making it hard to explain individual prediction.
        - Computationally heavy.
        - Slow prediction.

5. **Naive Bayes**
A probabilistic algorithm, which assumes all features to be independent of each other. It calculates the impact of each featu re on the likelihood of a class using probability then it multiplies all feature-wise probability to get an aggregate probability for each class, the class with highest probability score gets selected.
Formula:

$$P(C|X) = \frac{P(X|C) \cdot P(C)}{P(X)}$$

    - C= class label
    - X= {xa, x2, x3, …… , xn) features

Types:
    - Gaussian Naive Bayes: continuous features assumed to follow a normal distribution.
    - Multinomial Naïve Bayes: discrete features, often used for text classification (word counts).
    - Bernoulli Naive Byas: binary features (0/1), e.g., presence/absence of words.

Key Points:
    - **Purpose:** Quick, probabilistic classification; good for high-dimensional data.
    - **Output:** Class with highest posterior probability.
    - **Decision Rule:** Highest probability gets selected.
    - **Key Features:** Works well for both continuous and categorical data, assume conditional independence, can handle missing features.
    - **Use Case:** Email spam detection, medical diagnosis, spam analysis.
    - **Cons:**
        - Not suitable for small datasets.
        - Fails if features are correlated.
        - Zero probability problem when feature value never occurs in training for a class.

6. **SVM**
Support Vector Machine finds the optimal boundary separating classes in a dataset. It does this by using the closest data poi nts from each class, called support vectors, as references. These support vectors lie on the margin — the widest possible gap between the classes. For non-linear datasets, SVM projects the data into a higher-dimensional space using a kernel trick and then draws a separating hyperplane there.
Types:
    - Linear SVM: Finds straight line or hyperplane for separation.
    - Non-linear SVM: Uses kernel trick to map data into higher dimension space
        - Polynomial Kernel
        - RBF/Gaussian Kernel
        - Sigmoid Kernel

Formula:
    - Linear SVM: w*x + b = 0
        - w = weight vector
        - b = bias
    - Non-linear SVM:
        - Uses kernel function K(Xi,Xj) to compute dot products in higher-dimensional space.
        - Optimize same objective in transformed space

Key Points:

- **Purpose:** Classification (binary/multi-class) and regression (SVR).
- **Output:** Class label (or regression value).
- **Decision Rule:** Side of hyperplane (or transformed space) determines class.
- **Key Features:**
  - Maximizes margin for better generalization
  - Can handle high-dimensional data
  - Kernel trick allows non-linear separation
  - Sensitive to feature scaling → requires normalization
- **Use Cases:**
  - Text categorization (spam detection)
  - Image classification
  - Medical diagnosis
- **Cons**
  - Computationally intensive: slow for very large datasets.
  - Not easily interpretable: hard to explain the hyperplane in original feature space.
  - Sensitive to parameter tuning: kernel choice, C, gamma.
  - Needs scaled/normalized features: otherwise, features with large magnitude dominate the distance calculations.

Key Points:

Distance Metrics

- Euclidean:
- Manhattan:
- Mikowski:

Splitting Criteria

- Gini:
- Entropy:
- MSE:

Kernel Tricks

- Polynomial:
- RBF/Gaussian:
- Sigmoid: