1. Introduction & Need of Classification

Classification is a supervised learning method that assigns data to predefined categories by learning patterns from labeled training examples.

It predicts discrete class labels rather than numeric values.

The model studies relationships between input features and class labels and uses this knowledge to classify new, unseen data.

It is essential in domains where outcomes must fall into specific categories.

Need of Classification (Expanded Points)

1. **Automates decision-making**
   - Helps systems take quick and consistent decisions without human intervention (e.g., spam email filtering).
2. **Identifies patterns in large datasets**
   - Finds meaningful trends and hidden structures that humans cannot easily detect.
3. **Supports risk prediction**
   - Used by banks, insurance, and security systems to predict fraud, risk level, or customer churn.
4. **Reduces human effort**
   - Replaces manual sorting and categorization tasks with automated, accurate models.
5. **Improves speed and accuracy**
   - Classification systems process data quickly and often perform better than manual classification.
6. **Provides category-based insights**
   - Converts raw data into actionable categories useful for analysis and decision-making.

Applications

- Spam detection, sentiment classification, fraud detection, disease diagnosis, image & object recognition.

2. Types of Classification

A. Binary Classification

Binary classification predicts one of two possible classes such as yes/no, true/false, or 0/1.

Expanded Points

1. **Two output labels**
   - The target variable only contains two categories, making the problem straightforward and easy to interpret.
2. **Uses simple and effective algorithms**
   - Models like Logistic Regression, SVM, and Decision Trees perform well for two-class separation.
3. **Most commonly used in real-world problems**
   - Many everyday tasks—spam filtering, disease detection—naturally fit a two-class structure.
4. **Suitable for yes/no decision-based problems**
   - Useful when the objective is to decide between two mutually exclusive outcomes.
5. **Evaluation uses common metrics**
   - Accuracy, precision, recall, and F1-score are widely used to analyze binary classifier performance.

Advantages

- **Simple and fast** – Easy to train and test.
- **Easy interpretation** – Relationships between input and output are more understandable.
- **Requires less data** – Works well even with small datasets.

Disadvantages

- **Limited to two categories** – Cannot handle multi-class situations.
- **Risk of oversimplification** – Some problems require more than two classes for correct modeling.

Applications

Spam vs. Not spam, Fraud vs. Not fraud, Healthy vs. Diseased.

B. Multiclass Classification

Multiclass classification predicts one label from three or more possible categories.

Expanded Points

1. **Output contains 3 or more labels**
   - Suitable for problems where data naturally belongs to multiple categories (e.g., digit classification 0–9).
2. **Uses OvR, OvO, Softmax methods**
   - Techniques like “One-vs-Rest” or “One-vs-One” decompose a multiclass problem into simpler binary ones.
3. **More complex than binary classification**
   - Requires multiple decision boundaries and more training time.
4. **Needs more data and computation**
   - More classes require more examples to correctly understand each category.
5. **Uses specialized performance metrics**
   - Macro/micro accuracy and confusion matrices are used due to uneven class distribution.

Advantages

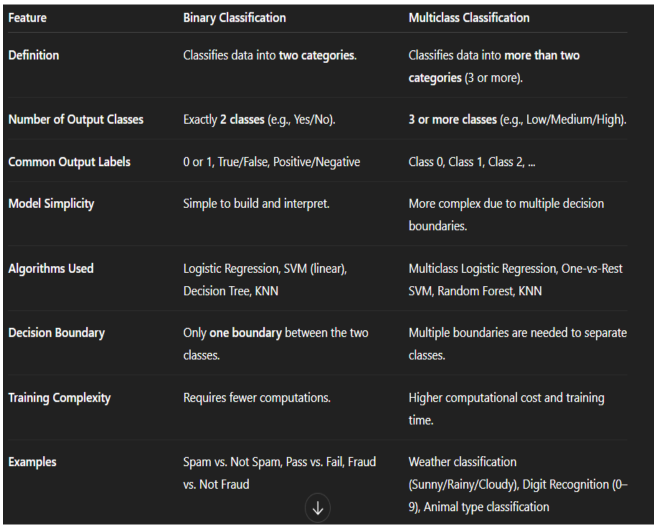
- **Handles real-world multiple-category tasks** – Many domains naturally require multi-class outputs.
- **Provides detailed predictions** – Offers more meaningful classification results.

Disadvantages

- **Higher complexity** – Model training and tuning becomes more difficult.
- **Greater chance of misclassification** – More classes increase confusion.

Applications

Digit recognition, object detection, document classification.



Feature	Binary Classification	Multiclass Classification
Definition	Classifies data into **two categories.**	Classifies data into **more than two categories** (3 or more).
Number of Output Classes	Exactly **2 classes** (e.g., Yes/No).	**3 or more classes** (e.g., Low/Medium/High).
Common Output Labels	0 or 1, True/False, Positive/Negative	Class 0, Class 1, Class 2, ...
Model Simplicity	Simple to build and interpret.	More complex due to multiple decision boundaries.
Algorithms Used	Logistic Regression, SVM (linear), Decision Tree, KNN	Multiclass Logistic Regression, One-vs-Rest SVM, Random Forest, KNN
Decision Boundary	Only **one boundary** between the two classes.	Multiple boundaries are needed to separate classes.
Training Complexity	Requires fewer computations.	Higher computational cost and training time.
Examples	Spam vs. Not Spam, Pass vs. Fail, Fraud vs. Not Fraud	Weather classification (Sunny/Rainy/Cloudy), Digit Recognition (0–9), Animal type classification

4. Balanced vs. Imbalanced Classification Problems

A. Balanced Classification

A dataset is balanced when all classes have roughly equal number of samples.

Expanded Points

1. **Fair learning for all classes**
   - Each class influences the model equally during training.
2. **Avoids prediction bias**
   - Model does not favor any class due to equal representation.
3. **Accuracy remains meaningful**
   - Evaluation metrics correctly reflect model performance.
4. **Standard algorithms work well**
   - No special preprocessing is needed for balanced datasets.

Advantages

- Easy to train and tune.
- Produces unbiased and stable predictions.

Disadvantages

- Hard to find in real-world datasets where one class dominates.

Applications

General ML tasks like image classification, topic modeling.

B. Imbalanced Classification

A dataset is imbalanced when one class appears far more frequently than others.

Expanded Points

1. **Very common in real-world problems**
   - Fraud detection, rare disease detection, and intrusion detection often have <5% minority class samples.
2. **Model tends to predict majority class**
   - Reduces the ability to detect rare but important events.
3. **Accuracy becomes misleading**
   - Model may show 99% accuracy but still fail to detect minority cases.
4. **Requires special handling methods**
   - Oversampling, undersampling, SMOTE, and class weights are needed to balance training.
5. **Alternative evaluation metrics required**
   - Recall, F1-score, ROC-AUC are more meaningful than simple accuracy.

Advantages

- Represents real-world distributions accurately.
- Useful for detecting rare events.

Disadvantages

- Difficult to train accurate models.
- Minority class often misclassified.
- Risk of overfitting when oversampling.

Applications

Fraud detection, cancer diagnosis, anomaly detection.

1. Binary Classification

Binary classification is a supervised learning task where the goal is to assign each input instance into **one of two possible classes**, usually represented as **0/1**, **positive/negative**, or **yes/no**.

Key Characteristics

- Only two classes exist → the model has to learn a boundary between them.
- Works best when the problem involves a clear two-way decision.
- Used widely in fields like finance, healthcare, security, and NLP.

Examples

- Spam Email → Spam (1) / Not Spam (0)
- Medical Diagnosis → Disease (1) / No Disease (0)
- Credit Risk → Default (1) / Safe (0)

Binary classification is the simplest and most commonly used classification form.

2. Linear Classification Model

A linear classification model separates two classes using a **straight line (2D)** or a **hyperplane (multi-dimensional)**. It assumes that the classes can be separated by a linear boundary.

Detailed Points

1. Linear Decision Boundary

- The classifier predicts class based on a linear equation:

$$y = w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + b$$

- If $\mathbf{y \geq 0}$, one class is chosen; otherwise, the other class is selected.

2. Works for Linearly Separable Data

- When positive and negative points can be separated by a straight line.
- If classes overlap heavily, linear models may struggle.

3. Examples of Linear Classifiers

- **Logistic Regression** – Outputs probability using sigmoid.
- **Linear SVM** – Finds the line with maximum margin.
- **Perceptron** – Early neural network for binary classification.

4. Computationally Efficient

- Training time is low, suitable for large datasets.
- Model interpretation is easier because weights indicate influence of features.

5. Limitations

- Cannot model complex curved boundaries.
- May underperform when data is nonlinearly separable unless transformed.

3. Performance Evaluation of Binary Classifier

Binary classification performance is evaluated using the **confusion matrix**, from which many metrics are derived.

3.1 Confusion Matrix (Detailed)

A confusion matrix compares **actual** vs **predicted** values, giving insight into all types of errors.

	Predicted Positive	**Predicted Negative**
Actual Positive	True Positive (TP)	False Negative (FN)
Actual Negative	False Positive (FP)	True Negative (TN)

Meaning of Each Term

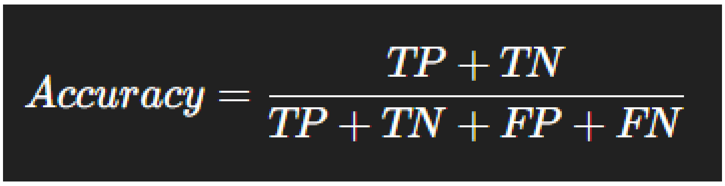
- **TP (True Positive):** Model correctly predicts positive cases.
- **TN (True Negative):** Model correctly predicts negative cases.
- **FP (False Positive):** Model incorrectly predicts positive → *Type I error*.
- **FN (False Negative):** Model failed to detect positive → *Type II error*.

Importance

- Gives complete picture of classification errors.
- Helps choose suitable metrics depending on domain (e.g., recall for medical diagnosis).

3.2 Accuracy

Accuracy is the proportion of all correct predictions.



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

Interpretation

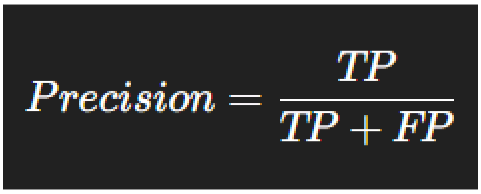
- High accuracy means overall correct classification.
- Easy to compute and widely used.

Limitation

- Misleading when dataset is **imbalanced**. Example: If 95% samples are negative, predicting all as negative gives 95% accuracy but is useless.

3.3 Precision

Precision tells how many predicted positives were truly positive.



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

Interpretation

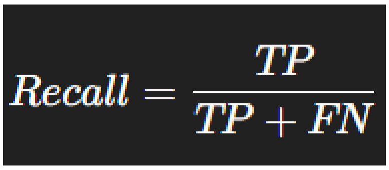
- High precision → very few false alarms.
- Ensures that positive predictions are reliable.

Useful When

- **False positives are costly.** Example: Predicting a person has cancer when they don't → unnecessary stress/tests.

3.4 Recall (Sensitivity)

Recall measures how many actual positives were correctly identified.



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

Interpretation

- High recall → model rarely misses positive cases.

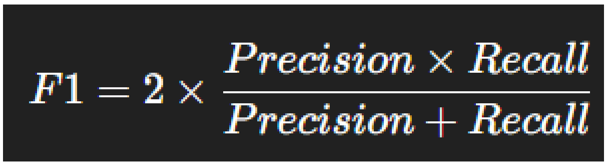
Useful When

- **False negatives are dangerous.** Example: Missing a fraud or cancer case can have serious outcomes.

3.5 F-Measure / F1-Score

Definition

F1-score is the harmonic mean of precision and recall.



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

Interpretation

- Balances the importance of precision and recall.
- Helpful when classes are imbalanced.
- A single metric that combines both correctness and completeness.

Why Harmonic Mean?

- Penalizes extreme values:
  If either precision or recall is low → F1 is low.

1. Multiclass Classification

Multiclass classification is a supervised learning task where the goal is to assign an input instance to **one class out of three or more possible categories**.

Examples

- Handwritten digit recognition (0–9)
- Classifying animals (cat, dog, horse)
- Sentiment categories (negative, neutral, positive)

Key Features

- Output variable contains **3+ categories**.
- Many binary classifiers are extended to handle multiple classes.
- Requires more computation and careful evaluation compared to binary classification.

2. Multiclass Classification Techniques

A. One-vs-All (OvA) / One-vs-Rest (OvR)

OvA creates **one classifier per class**, treating that class as **positive** and all other classes as **negative**.

Detailed Points

1. **Number of classifiers = number of classes (K)**
   - For 4 classes → 4 classifiers trained.
2. **Training approach**
   - Example: For class "Cat", label Cat = 1, all other classes = 0.
   - Repeat for Dog, Horse, etc.
3. **Prediction**
   - Input is passed to all classifiers.
   - The classifier giving the **highest confidence score** determines the class.
4. **Advantages**
   - Simple and easy to implement.
   - Efficient when number of classes is large.
5. **Disadvantages**
   - Performance may drop if classes are highly overlapping.
   - One classifier must separate one class from all others → harder boundary.

B. One-vs-One (OvO)

OvO builds a classifier for **every possible pair of classes**.

Detailed Points

1. **Number of classifiers = K(K−1)/2**
   - Example: For 4 classes → 6 classifiers.
2. **Training approach**
   - Each classifier trains on **only two classes at a time**.
   - Example: Cat vs Dog, Cat vs Horse, Dog vs Horse.
3. **Prediction**
   - Every classifier votes for one class.
   - The class with **maximum votes** is the final prediction.
4. **Advantages**
   - Each classifier handles a simpler 2-class problem.
   - Often gives higher accuracy than OvA.
5. **Disadvantages**
   - Requires many classifiers → computationally expensive for large K.
   - Prediction time increases because many models must vote.

3. Performance Evaluation for Multiclass Classification

Multiclass problems require more detailed evaluation because errors can occur **between any pair of classes**.

3.1 Multiclass Confusion Matrix

A multiclass confusion matrix is a **K × K** table where rows represent **actual classes** and columns represent **predicted classes**.

Structure Example (3 classes: A, B, C)

Actual \ Predicted	**A**	**B**	**C**
A	AA (TP of A)	AB	AC
B	BA	BB (TP of B)	BC
C	CA	CB	CC (TP of C)

Interpretation

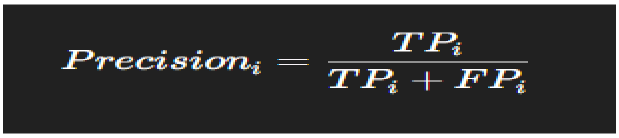
- Diagonal elements = Correct predictions (True Positives for each class).
- Off-diagonal elements = Misclassifications between classes.

Why Useful?

- Shows how often one class is confused with another.
- Very important when classes are imbalanced.

3.2 Per-Class Precision

Per-Class Precision measures how many predictions **for a class** were correct.



$$Precision_i = \frac{TP_i}{TP_i + FP_i}$$

- **TP$_i$** = Correct predictions of class i
- **FP$_i$** = Instances incorrectly predicted as class i

Interpretation

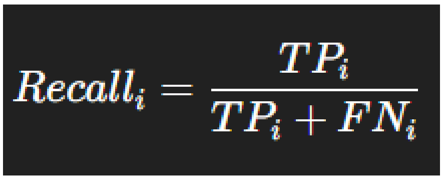
- High precision for class *i* means **few false positives** for that class.
- Indicates reliability of predictions for that specific class.

Use Cases

- Useful when **false positives are costly** (e.g., predicting a category incorrectly triggers an expensive action).

3.3 Per-Class Recall

Per-Class Recall measures how many **actual instances** of that class were correctly identified.



$$Recall_i = \frac{TP_i}{TP_i + FN_i}$$

- **TP$_i$** = Correct predictions of class i
- **FN$_i$** = Actual class i instances predicted as other classes

Interpretation

- High recall means class *i* is rarely missed.
- Good for situations where **false negatives are dangerous**.

Use Cases

- Medical diagnosis
- Defect detection
- Fraud detection

1. K-Nearest Neighbor (KNN)

KNN is a non-parametric, instance-based, lazy learning classification algorithm. A new sample is classified based on the majority class among its K nearest neighbors. Uses distance metrics such as Euclidean, Manhattan, or Minkowski.

How KNN Works

1. Choose a value for K.
2. Calculate distance of new data point from all training points.
3. Select the K closest points.
4. Perform majority voting.
5. Assign class with maximum votes.

Advantages

- Simple to understand and implement.
- Works well for non-linear decision boundaries.

Disadvantages

- Slow for large datasets.
- Sensitive to noisy features and scaling.

Applications

- Pattern recognition
- Recommendation systems
- Image classification

2. Support Vector Machine (SVM) Classification Algorithm

Support Vector Machine (SVM) is a **supervised machine learning algorithm** used for **binary and multi-class classification**.
Its main idea is to **find the best separating boundary (hyperplane)** that divides data points of different classes with the **maximum margin**.

1. Hyperplane

A hyperplane is a decision boundary that separates different classes.

In 2D → a line
In 3D → a plane
In nDimensions → a hyperplane

SVM tries to find the **optimal hyperplane**.

Hyperplane equation:

w·x+b=0

Decision Boundary

If the optimal hyperplane found is:

w1x1+w2x2+b=0

Then the classifier decides:

- If w·x+b>0→ Class A
- If w·x+b<0 → Class B

2. Margin

The margin is the **distance between the hyperplane and the nearest data points** from each class.

SVM chooses the hyperplane with the **maximum margin**, making the classifier more robust.

3. Support Vectors

Support Vectors are the **critical data points** that lie closest to the decision boundary.
They influence the position and orientation of the hyperplane.

If these points are removed, the hyperplane will change.

4. Linearly Separable vs Non-Linear Data

- If data is linearly separable → SVM finds a straight line.
- If data is non-linear → SVM uses **Kernel Trick** to convert low-dimensional data into high-dimensional space to make it separable.

Common kernels:

- Linear Kernel
- Polynomial Kernel
- Radial Basis Function (RBF)
- Sigmoid Kernel

✅ Advantages of SVM

- Works well for high-dimensional data
- Effective when classes are separable
- Robust to outliers because margin maximization reduces overfitting
- Kernel trick helps classify complex data

✅ Applications

- Face recognition
- Text and email classification (spam/ham)
- Handwriting recognition
- Bioinformatics

3. Soft Margin SVM

Why Needed

- Real-world data is not perfectly linearly separable.
- Soft margin SVM allows some misclassification to reduce overfitting.

Concept

- Introduces slack variables (xi) to allow margin violations.
- Adds a penalty parameter C:
  - High C = less tolerance to misclassification (hard margin).
  - Low C = more tolerance (soft margin).

Benefits

- Handles noisy, overlapping data better.
- Achieves a balance between margin size and classification accuracy.

4. Kernel Functions in SVM

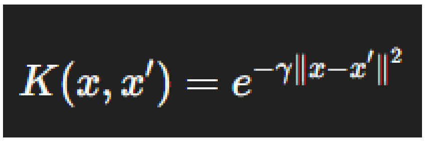
Support Vector Machines (SVM) work very well when data is **linearly separable**, meaning a single straight line or hyperplane can divide the classes.
However, most real-world data is **non-linear**, and a linear SVM cannot find a good boundary.

To solve this, SVM uses **Kernel Functions**, which allow the algorithm to operate in a **higher-dimensional feature space** without explicitly computing the transformation.
This trick is called the **Kernel Trick**.

Why Kernels Are Used

- Linear SVM cannot handle non-linear data.
- Kernel functions map data to higher-dimensional space without explicitly computing it (Kernel Trick).
- Enables SVM to find non-linear boundaries in original space.

4.1 Radial Basis Function (RBF) Kernel

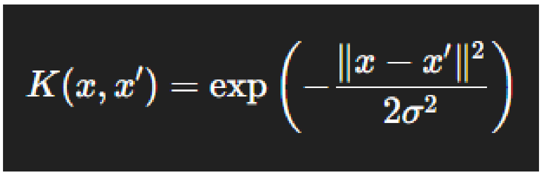


$$K(x, x') = e^{-\gamma \|x - x'\|^2}$$

Key Points

- Measures similarity based on distance.
- gamma controls flexibility of the decision boundary.
- Very effective for non-linear classification.

4.2 Gaussian Kernel

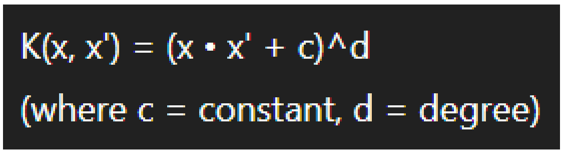


$$K(x, x') = \exp\left(-\frac{\|x - x'\|^2}{2\sigma^2}\right)$$

Key Points

- A special case of RBF.
- sigma controls the spread of the kernel.
- Useful for clustered data.

4.3 Polynomial Kernel



K(x, x') = (x • x' + c)^d

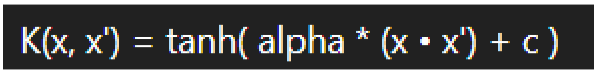
(where c = constant, d = degree)

Key Points

- Captures feature interactions.
- Allows curved decision boundaries.
- Good for datasets where relation between features is polynomial.

4.4 Sigmoid Kernel

Formula



K(x, x') = tanh(alpha * (x • x') + c)

Key Points

- Similar to activation function of neural networks.
- Works well for certain text and signal classification tasks.