Text Classification is the process of automatically assigning predefined categories or labels to text documents based on their content.

What is it?

It is a supervised machine learning task used to organize, filter, and understand large collections of text by mapping them to meaningful classes like *spam/ham, sports, politics, sentiment,* etc.

Why is it used?

- To manage the growing quantity of digital text.
- To automate tasks like document sorting, filtering, sentiment detection.
- To support search engines, recommendation systems, and content organization.

Concept

Text is converted into numerical representations (features). Machine learning algorithms then learn patterns from labeled examples and use those patterns to classify new, unseen documents.

Working / Steps

1. **Data Collection:** Gather labeled documents.
2. **Preprocessing:** Tokenization, stop-word removal, stemming/lemmatization.
3. **Feature Extraction:** Convert text into BoW, TF-IDF, or embeddings.
4. **Model Training:** algorithms like Naïve Bayes / KNN learn class patterns.
5. **Classification:** The model predicts the class of new documents.
6. **Evaluation:** Using accuracy, precision, recall, F1.

Advantages

- Helps manage large-scale text data.
- Automates manual document sorting.
- Works well with simple ML models.
- Improves IR tasks like filtering, indexing, and ranking.

Disadvantages

- Requires high-quality labeled training data.
- Sensitive to noisy text (spelling errors, mixed languages).
- Bias in training data affects results.

Applications

- Email spam detection
- News article tagging
- Sentiment analysis
- Customer feedback classification
- Content moderation

2. Naïve Bayes Model

Definition

Naïve Bayes is a probabilistic classifier based on Bayes' Theorem and the assumption that all features (words) are conditionally independent given the class.

What is it?

A simple and efficient machine learning model widely used for text classification tasks like spam filtering and sentiment analysis.

Why is it used?

- Fast training and prediction
- Works well on high-dimensional text data
- Performs surprisingly well even with strong independence assumptions

Concept

Naïve Bayes calculates the probability that a document belongs to each class.
The class with the highest probability is chosen.

Minimal Math

Bayes' Theorem:

$$P(C \mid D) = \frac{P(D \mid C) \cdot P(C)}{P(D)}$$

Since P(D) is the same for all classes, the classifier compares:

$$P(D \mid C) \cdot P(C)$$

Working / Steps

1. **Compute Prior Probability:**

$$P(C) = \frac{\text{Documents in class}}{\text{Total documents}}$$

2. **Compute Likelihood:** Estimate probability of each word appearing in each class.
3. **Combine Probabilities:** Multiply word probabilities (with log to avoid underflow).
4. **Choose Class:** Select class with maximum posterior probability.

Advantages

- Very fast for large datasets
- Works well with sparse, high-dimensional data
- Requires little training data
- Simple, interpretable, and effective

Disadvantages

- Independence assumption is often unrealistic
- Zero probability problem (handled using Laplace smoothing)
- Not ideal for complex relationships between features

Applications

- Spam classification
- SMS categorization
- Sentiment analysis
- Topic classification
- Email filtering

3. K-Nearest Neighbor (KNN)

Definition

KNN is a non-parametric classification method that assigns a class to a new document based on the classes of its K most similar neighbors.

What is it?

An instance-based learning algorithm that stores all training examples and uses distance/similarity measures to classify new data.

Why is it used?

- Easy to understand and implement
- No training phase (lazy learning)
- Works well for document similarity tasks

Concept

Documents are represented as vectors (e.g., TF-IDF). KNN compares the new document to all stored documents, finds the closest K neighbors, and assigns the majority class among them.

Working / Steps

1. **Convert Text to Vectors:** Using TF-IDF or BoW.
2. **Choose K:** Often 3 or 5.
3. **Compute Similarity/Distance:**
    - Cosine similarity (common for text)
4. **Find K Nearest Neighbors:** Select closest documents.
5. **Vote for Majority Class:** Predict the class appearing most frequently among neighbors.

Minimal Math

Cosine Similarity:

$$\cos(\theta) = \frac{A \cdot B}{||\ A\ ||\ \ ||\ B\ ||}$$

Advantages

- Simple and intuitive
- No training time
- Works well with multi-class problems
- Naturally handles non-linear boundaries

Disadvantages

- Slow prediction for large datasets
- Requires storing entire dataset (high memory)
- Performance decreases with very high-dimensional data (curse of dimensionality)

Applications

- Document similarity search
- Query-by-example retrieval
- Recommendation systems
- Categorizing short texts
- News topic classification

1. Spam Filtering

Definition

Spam filtering is the process of automatically detecting and separating unwanted or harmful emails/messages (spam) from legitimate ones (ham).

What is it?

A text classification task where machine learning or rule-based systems analyze message content to decide whether it is spam.

Why is it used?

- To protect users from phishing and harmful links
- To reduce inbox clutter
- To improve email deliverability

Concept

Emails are converted into features (words, frequency, metadata), and a classifier (Naïve Bayes, SVM, KNN, etc.) decides if the message is spam or not.

Working / Steps

1. **Data collection:** Collect spam and ham emails.
2. **Preprocessing:** Clean text, remove stop words, tokenize.
3. **Feature extraction:** Use TF-IDF, word frequency, or embeddings.
4. **Model training:** Train classifiers such as Naïve Bayes, SVM.
5. **Prediction:** Incoming email is scored as spam or ham.
6. **Evaluation:** Accuracy, precision, recall.

Advantages

- Prevents malware and phishing
- Saves storage and reduces inbox overload
- Highly accurate with ML models

Disadvantages

- False positives can hide important messages
- Requires updating with new spam patterns
- Attackers continuously change techniques

Applications

- Email spam detection
- SMS spam filtering
- Social media bot/spam detection

✅ **2. Support Vector Machine (SVM) Classifier**

Definition

SVM is a supervised machine learning algorithm that separates data into classes using the best possible boundary called a hyperplane.

What is it?

A high-performance classifier widely used in text classification because it handles high-dimensional data (like text) very well.

Why is it used?

- Works extremely well with large feature spaces such as TF-IDF vectors
- Provides strong accuracy even with small training data
- Handles linearly and non-linearly separable data

Concept

SVM tries to find a hyperplane that **maximizes the margin** between classes.
The points closest to the hyperplane are called **support vectors**.

Minimal Math

For a hyperplane:

$$w \cdot x + b = 0$$

SVM maximizes the margin:

$$\frac{2}{|| w ||}$$

Working / Steps

1. **Convert text into vectors:** (TF-IDF, embeddings).
2. **Train SVM:** Find hyperplane that best separates classes.
3. **Compute support vectors:** Identify key boundary points.

4. **Classification:** New documents are classified based on which side of the hyperplane they fall.
5. **Use kernel if data is non-linear.**

Advantages

- Very effective for text classification
- Works well with high-dimensional and sparse data
- Strong generalization performance

Disadvantages

- Slow training for very large datasets
- Hard to tune kernel parameters
- Not ideal for noisy datasets

Applications

- Spam filtering
- Sentiment analysis
- Document topic classification
- Face recognition

✅ 3. Vector Space Classification Using Hyperplanes

Definition

This method represents documents as vectors and separates them using geometric decision boundaries (hyperplanes).

What is it?

A classification approach where documents are located in a multi-dimensional space, and models use boundaries to classify them.

Why is it used?

- Works effectively with vector representations like TF-IDF
- Supports linear separability
- Forms the basis of SVM and perceptron classifiers

Concept

Each document becomes a point in a high-dimensional vector space.
A **hyperplane** is used to divide the space into class regions.

Hyperplane General Form

$$w_1x_1 + w_2x_2 + \cdots + b = 0$$

Working / Steps

1. **Convert text into vector representation.**
2. **Choose model:** linear classifiers (SVM, Perceptron).
3. **Find optimal hyperplane:** separates data with maximum margin.
4. **Classify:** Based on the side of the hyperplane where the document lies.

Advantages

- Works with large-dimensional text data
- Simple geometric interpretation
- Efficient for classification tasks

Disadvantages

- Only linear separation unless kernel functions are used
- Sensitive to outliers
- Requires proper feature scaling

Applications

- Document categorization
- Spam classification
- Sentiment analysis
- Query classification in search engines

✅ 4. Kernel Function (For SVM and Hyperplane Methods)

Definition

A kernel function is a mathematical function that transforms data into a higher-dimensional space to make classes separable.

What is it?

SVM uses kernels when data is **not linearly separable** in the original space.

Why is it used?

- To handle complex, curved, or non-linear decision boundaries

- To avoid explicitly computing high-dimensional transformation

Concept

Instead of mapping points to high-dimensional space manually, kernel functions compute similarities directly.

Common Kernels

1. **Linear Kernel:**

$$K(x, y) = x \cdot y$$

Used in text classification.

2. **Polynomial Kernel:**

$$K(x, y) = (x \cdot y + 1)^d$$

3. **RBF (Gaussian) Kernel:**

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

Working / Steps

1. **Check if data is linearly separable.**
2. **Apply suitable kernel (e.g., RBF for complex boundaries).**
3. **SVM computes similarity in high-dimensional space.**
4. **Optimal hyperplane is identified in transformed space.**
5. **New documents classified using kernel comparisons.**

Advantages

- Allows SVM to classify non-linear patterns
- Efficient for complex decision boundaries
- No need to compute high-dimensional mapping manually

Disadvantages

- Requires tuning kernel parameters
- Can be computationally expensive
- Risk of overfitting if kernel complexity is high

Applications

- Text classification (mostly linear kernel)
- Image recognition
- Bioinformatics
- Speech classification

Text Clustering – Introduction

Definition

Text clustering is an unsupervised learning technique used to group similar documents into clusters based on their content.

What is it?

It automatically discovers natural groups in a collection of documents **without predefined labels**.

Why is it used?

- To organize large text collections
- To discover hidden patterns or topics
- To support search engines, recommendation systems, topic modeling

Concept

Documents are converted into numerical vectors (TF-IDF, Bag-of-Words, embeddings).
Clustering algorithms then measure similarity between documents and group them into clusters where documents in the same cluster are more similar to each other than to those in other clusters.

✅ **2. Clustering vs Classification (Difference)**

Point	**Clustering**	**Classification**
Learning Type	Unsupervised	Supervised
Labels Needed?	No predetermined labels	Requires labeled training data
Goal	Discover hidden groups	Assign document to known class
Output	Clusters/Groups	Classes/Categories
Use Cases	Topic discovery, document grouping	Spam detection, sentiment analysis

Point	Clustering	Classification
Algorithms	K-Means, Hierarchical Clustering	SVM, Naïve Bayes, Decision Trees

Summary

- **Clustering = Find groups automatically.**
- **Classification = Learn from labeled examples.**

✅ **3. Partitioning Methods in Text Clustering**

Partitioning methods divide a document collection into **k distinct clusters** based on similarity.
They require choosing the number of clusters (k) beforehand.

3.1 K-Means Clustering (Most Important Partitioning Method)

Definition

K-Means is a partitioning algorithm that groups documents into **k clusters** by minimizing the distance between documents and cluster centers.

Concept

Each cluster has a **centroid** representing the average vector of documents in that cluster.

Working / Steps

1. **Choose k** = number of clusters.
2. **Initialize centroids** randomly or using k-means++.
3. **Assign documents** to the nearest centroid (using cosine similarity or Euclidean distance).
4. **Update centroids** by computing the mean of all documents in each cluster.
5. **Repeat** until assignments stop changing.

Minimal Math

Distance between document vector **d** and centroid **c**:

$$distance(d, c) = || \, d - c \, ||$$

Advantages

- Simple and easy to implement
- Fast and scalable for large document sets
- Works well when clusters are spherical and separated

Disadvantages

- Requires choosing k in advance
- Sensitive to initial centroids
- Fails with non-linear or overlapping clusters

Applications

- Document topic grouping
- News article clustering
- Search result grouping

3.2 K-Medoids (PAM – Partitioning Around Medoids)

Definition

K-Medoids is similar to K-Means but uses **actual documents (medoids)** instead of centroids as cluster centers.

Concept

The medoid is the real document with the least average distance to other documents in that cluster.

Working / Steps

1. Choose k medoids.
2. Assign each document to the nearest medoid.
3. Swap medoids with candidates to reduce cost.
4. Repeat until no improvement.

Advantages

- More robust to noise and outliers
- Distance-based: works with any similarity measure

Disadvantages

- Slower than K-Means
- Not suitable for extremely large datasets

Applications

- Clustering short texts
- Sentence clustering
- Customer review grouping

K-Means Clustering

Definition

K-Means is a **partitioning-based clustering algorithm** that divides documents/data points into **k clusters** by minimizing the distance between points and their cluster centroid.

What is it?

A fast, iterative method that forms clusters based on similarity.

Why is it used?

- Easy to implement
- Works well for large text collections
- Useful when clusters are well-separated

Concept

Each cluster has a **centroid** (mean vector).
Documents are assigned to the nearest centroid using cosine similarity or Euclidean distance.

Working Steps

1. Choose **k** clusters.
2. Initialize **k** centroids randomly.
3. Assign each document to the closest centroid.
4. Recalculate the centroid of each cluster.
5. Repeat steps 3–4 until cluster assignments stabilize.

Minimal Math

$$\text{Centroid } c = \frac{1}{n}\sum_{i=1}^{n} \quad x_i$$

Advantages

- Very fast and scalable
- Simple and easy to understand
- Works well on large datasets

Disadvantages

- Must choose k in advance
- Sensitive to initial points
- Not suitable for non-spherical clusters
- Sensitive to outliers

Applications

- Document clustering
- News topic grouping
- Filtering search results (cluster-based browsing)
- Customer segmentation

2. Agglomerative Hierarchical Clustering (AHC)

Definition

Hierarchical clustering builds a **tree-like structure (dendrogram)** to group documents, starting with each document as its own cluster and merging them step by step.

What is it?

A bottom-up clustering approach that does not require specifying k in advance.

Why is it used?

- Reveals natural cluster structure
- Useful when the number of clusters is unknown

Concept

Documents are merged based on **similarity**.
Linkage criteria decide how to measure cluster distance:

- **Single-link** → nearest points
- **Complete-link** → farthest points
- **Average-link** → average similarity
- **Ward's method** → minimizes variance

Working Steps

1. Start with each document as a single cluster.
2. Compute pairwise similarity between all clusters.
3. Merge the closest two clusters.
4. Update distances and repeat until one cluster remains.
5. Cut the dendrogram at chosen height to get final clusters.

Minimal Math

Cluster distance example (single link):

$$D(C_i, C_j) = \min_{x \in C_i, y \in C_j} || \, x - y \, ||$$

Advantages

- No need to choose number of clusters initially
- Produces interpretable dendrogram
- Works well for small to medium datasets

Disadvantages

- Computationally expensive ($O(n^2)$)
- Cannot undo wrong merges
- Not suitable for very large datasets

Applications

- Topic discovery in text
- Gene expression analysis
- Grouping academic papers by similarity
- Creating taxonomy hierarchies

3. Expectation-Maximization (EM) Algorithm

Definition

EM is an iterative algorithm used to estimate parameters of probabilistic models when data contains hidden or missing variables.

What is it?

Used to fit mixture models (like Gaussian mixtures) to data.

Why is it used?

- Handles incomplete or uncertain cluster assignments
- More flexible than k-means (soft clustering)

Concept

EM alternates between:

- **E-Step (Expectation):** Compute probability that a data point belongs to each cluster.
- **M-Step (Maximization):** Update the model parameters (mean, variance, weights) to maximize likelihood.

Working Steps

1. Initialize parameters (means, variances, weights).
2. **E-Step:** Compute membership probabilities.
3. **M-Step:** Update parameters using those probabilities.
4. Compute new likelihood.
5. Repeat until convergence.

Minimal Math

E-step probability:

$$P(z_k \mid x_i) = \frac{w_k \cdot N(x_i \mid \mu_k, \Sigma_k)}{\sum_j \quad w_j N(x_i \mid \mu_j, \Sigma_j)}$$

Advantages

- Soft clustering (documents can belong partially to clusters)
- Good for complex cluster shapes
- Probabilistic and more accurate than k-means

Disadvantages

- Slow convergence
- Can get stuck in local minima
- Requires good initialization

Applications

- Document clustering in IR
- Speech recognition
- Medical diagnosis
- Missing data prediction

4. Mixture of Gaussians (Gaussian Mixture Model – GMM)

Definition

A probabilistic model that assumes the data is generated from a mixture of **multiple Gaussian (normal) distributions**.

What is it?

A soft-clustering model used to assign documents to clusters based on probability distribution.

Why is it used?

- Can model clusters of various shapes, sizes, and densities
- More flexible than k-means

Concept

Each cluster is represented by:

- Mean (center)
- Covariance (shape)
- Weight (importance)

EM algorithm is used for parameter estimation.

Working Steps

1. Assume number of Gaussians = k.
2. Initialize mean, covariance, and weights.
3. Use EM algorithm to update parameters.
4. Determine cluster membership probabilities.

Minimal Math

Total likelihood:

$$P(x) = \sum_{k=1}^{K} w_k N(x \mid \mu_k, \Sigma_k)$$

Advantages

- More accurate cluster boundaries
- Can capture elliptical cluster shapes
- Probabilistic and flexible

Disadvantages

- Computationally expensive
- Sensitive to initialization
- Assumes data follows Gaussian distribution

Applications

- Document topic modeling
- Clustering user behavior
- Image segmentation
- Pattern recognition