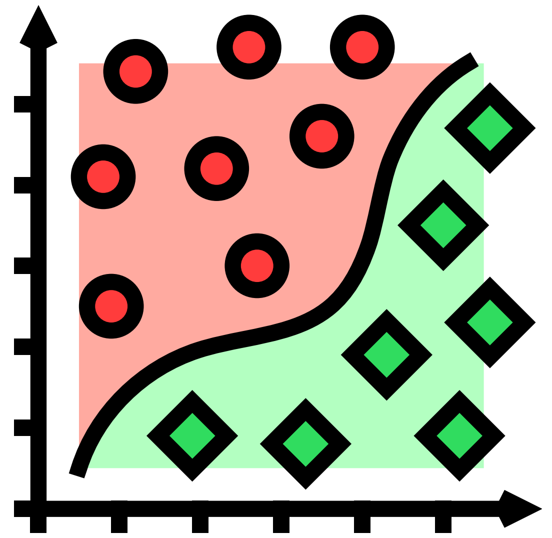
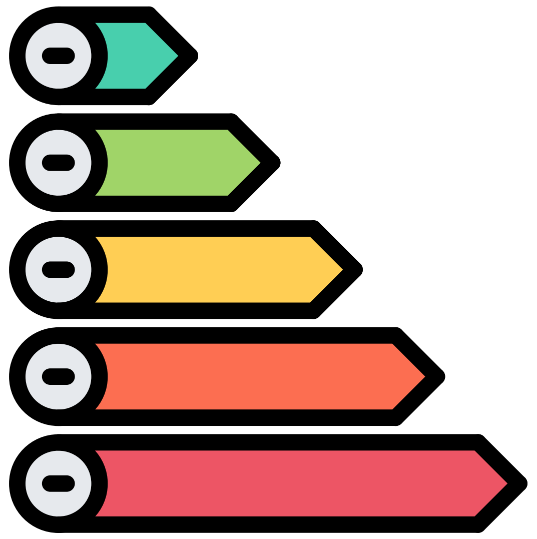
KAZAKH-BRITISH TECHNICAL UNIVERSITY

School of IT and Engineering





MACHINE LEARNING

LECTURE #3

Number of credits: 3 (2/0/1)
Course code – CSCI3234
MS Teams team code – **y4ntwlz**

Adilet Yerkin, MS in Engineering, Senior Lecturer
a.yerkin@kbtu.kz

Learning methods

Learning methods are approaches that allow computers to automatically learn patterns from data. They form the foundation for predictive modeling, pattern discovery, and decision-making systems.

Definitions of ML

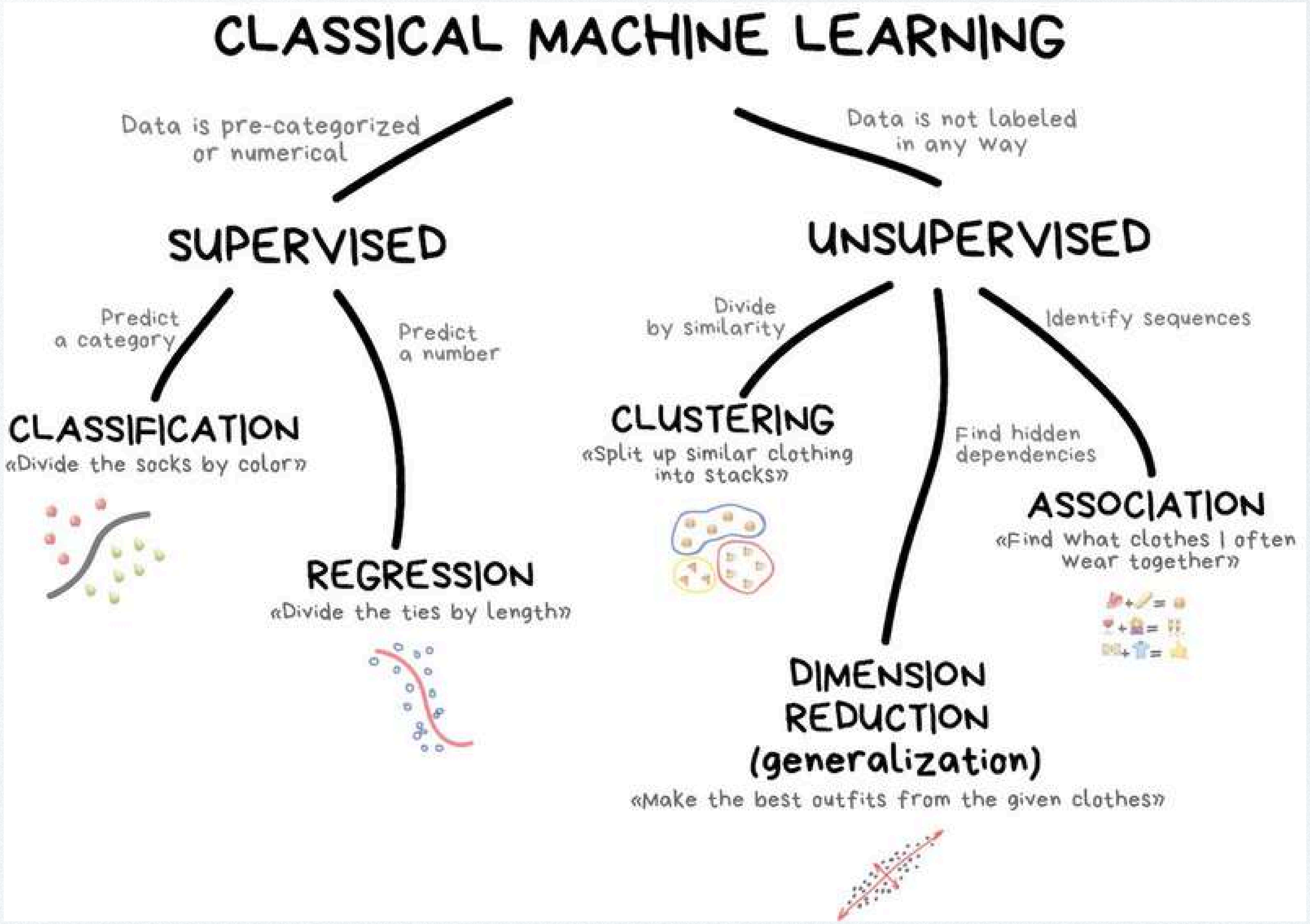
Machine learning is a field of artificial intelligence (AI) that focuses on the development of algorithms and statistical models that enable computer systems to improve their performance on a specific task through learning from data, without being explicitly programmed.

The field of study that gives computers the ability to learn without being explicitly programmed.

the ability to improve the performance of a task by being exposed to data

A field of AI that uses statistical techniques and algorithms to enable computer systems to recognize patterns in data, make predictions, classify information, and automate decision-making based on the learned patterns and insights.

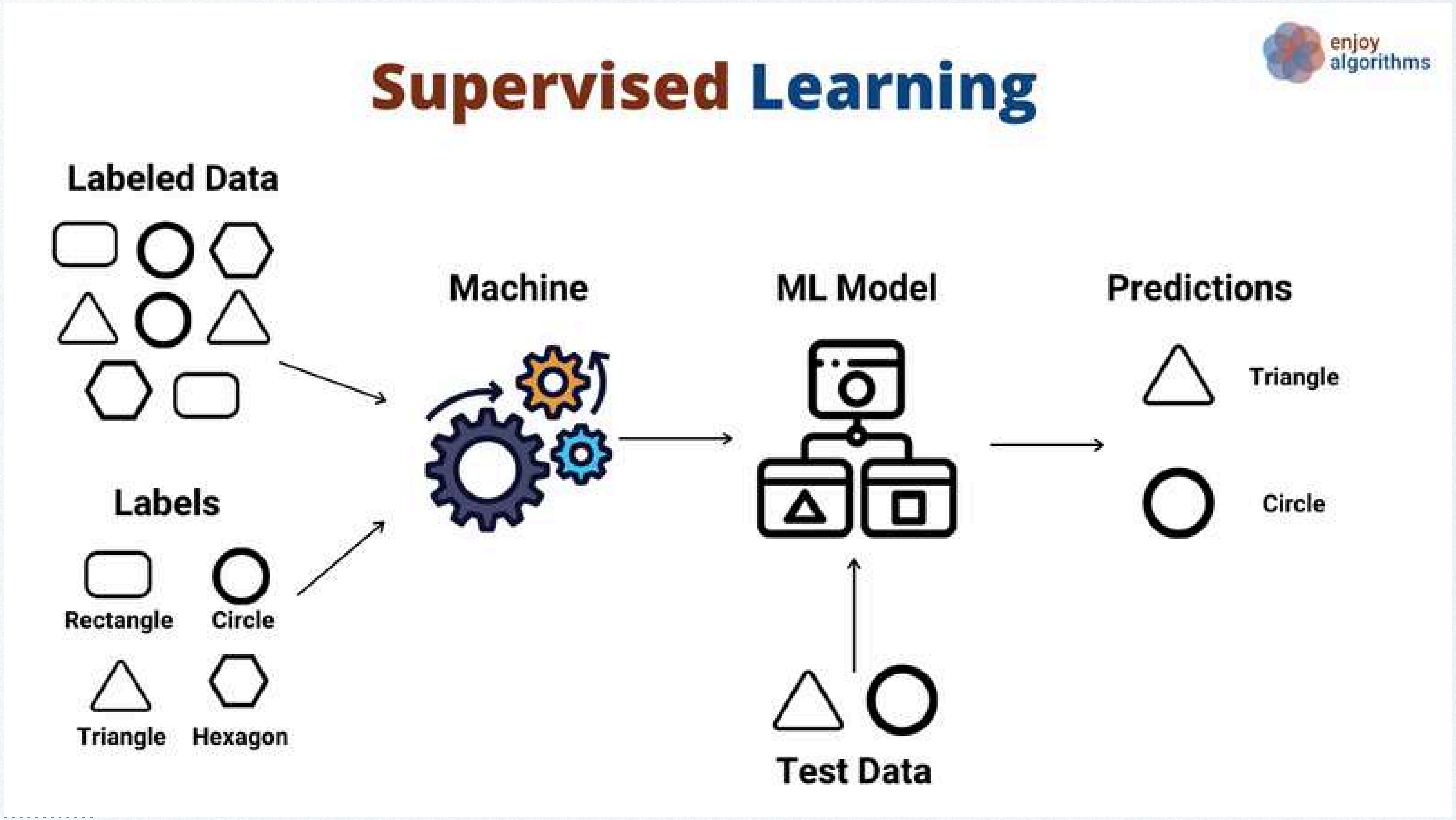
Types of ML


CLASSICAL MACHINE LEARNING
Data is pre-categorized or numerical
Data is not labeled in any way
SUPERVISED
UNSUPERVISED
Predict a category
Predict a number
Divide by similarity
Identify sequences
CLASSIFICATION
«Divide the socks by color»
REGRESSION
«Divide the ties by length»
CLUSTERING
«Split up similar clothing into stacks»
Find hidden dependencies
ASSOCIATION
«Find what clothes I often wear together»
DIMENSION REDUCTION (generalization)
«Make the best outfits from the given clothes»

Supervised Learning

In **supervised learning**, the model is trained on a labeled dataset, meaning each training example is paired with an output label.

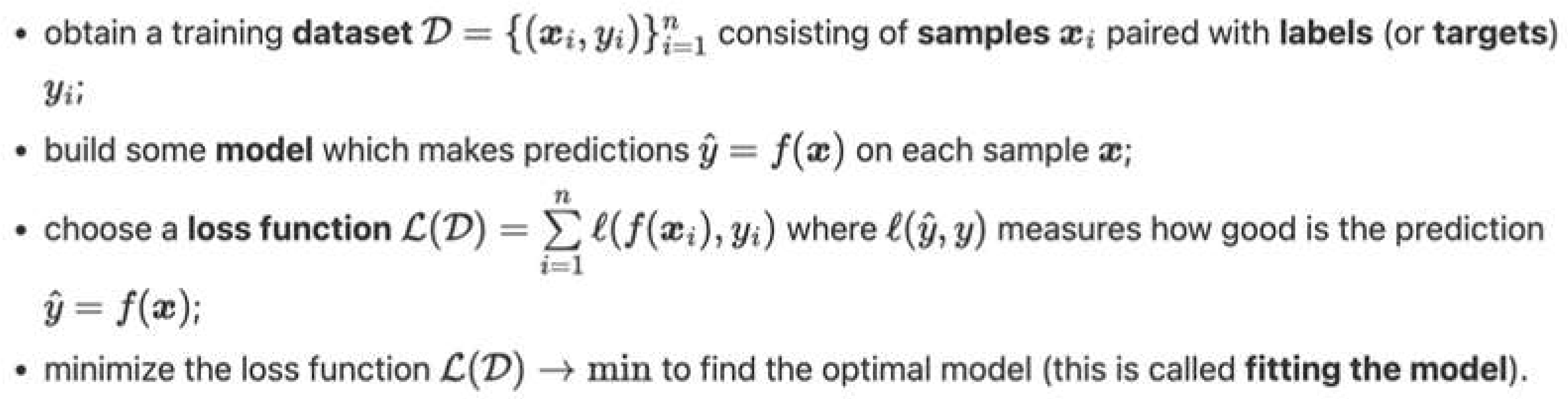
The goal is to learn a mapping from inputs to outputs so that the model can predict labels for unseen data.



the algorithm learns a mapping from input features (X) to output labels (Y) using a labeled dataset.

Supervised Learning

The general pipeline of supervised learning:



- obtain a training **dataset** $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$ consisting of **samples** $\boldsymbol{x}_i$ paired with **labels** (or **targets**) y_i;
- build some **model** which makes predictions $\hat{y} = f(\boldsymbol{x})$ on each sample $\boldsymbol{x}$;
- choose a **loss function** $\mathcal{L}(\mathcal{D}) = \sum\limits_{i=1}^{n} \ell(f(\boldsymbol{x}_i), y_i)$ where $\ell(\hat{y}, y)$ measures how good is the prediction $\hat{y} = f(\boldsymbol{x})$;
- minimize the loss function $\mathcal{L}(\mathcal{D}) \to \min$ to find the optimal model (this is called **fitting the model**).

A **loss function** (also called a cost function or error function) is a formula that calculates the difference between the predicted values (from the model) and the true values (from the training data).

Supervised Learning

In supervised learning training **dataset** $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^n$ consists of

- training **samples** $\boldsymbol{x}_i$ which are usually vectors from a multidimensional space $\boldsymbol{x}_i \in \mathbb{R}^d$;
- **targets** (or **labels**) $y_i \in \mathcal{Y}$ paired with samples $\boldsymbol{x}_i$.

The elements of a training sample

$$\boldsymbol{x}^\top = (x_1, \ldots, x_d)$$

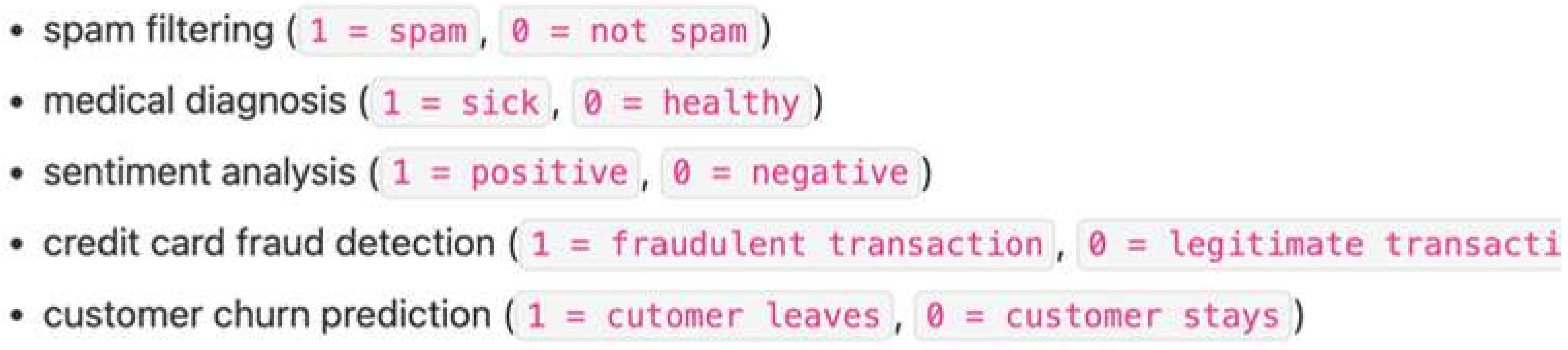
are also called **features**. Thus, if $\boldsymbol{x}_i \in \mathbb{R}^d$, then dataset $\mathcal{D}$ has d (numeric) features. All training samples

$$\boldsymbol{x}_i, \quad i = 1, \ldots, n,$$

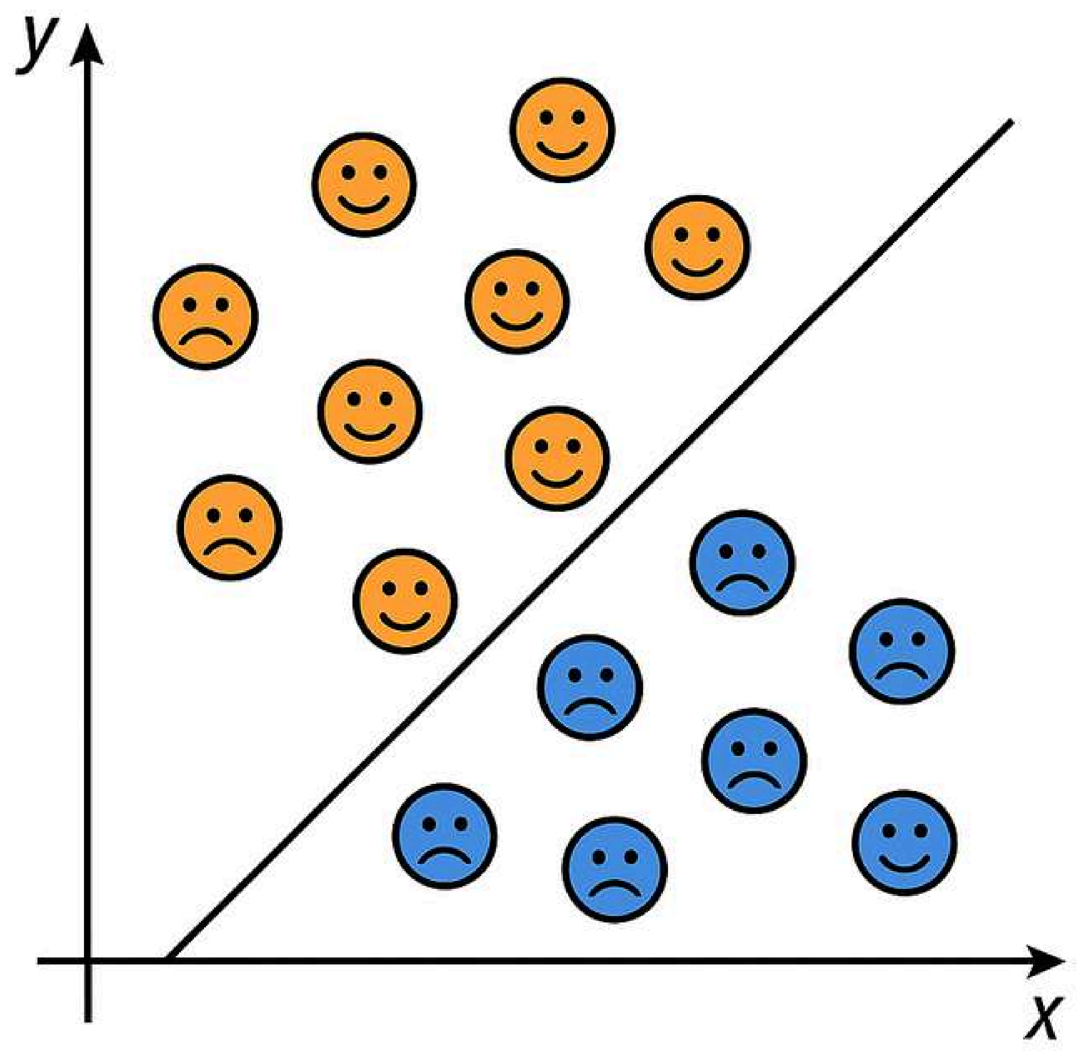
Supervised Learning

Binary classification

In binary classification problems there are only two classes, which are often called positive and negative.



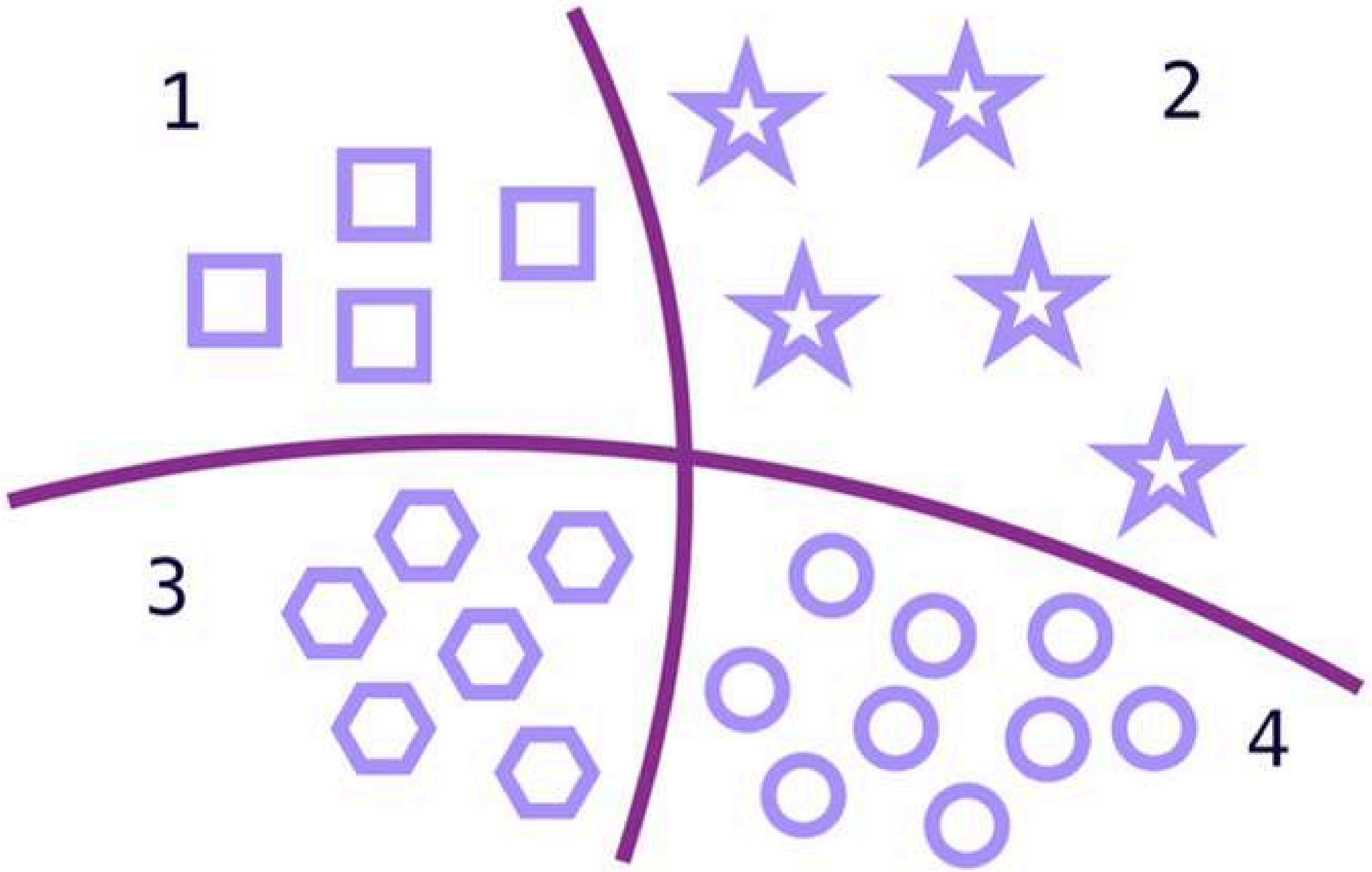
- spam filtering (`1 = spam`, `0 = not spam`)
- medical diagnosis (`1 = sick`, `0 = healthy`)
- sentiment analysis (`1 = positive`, `0 = negative`)
- credit card fraud detection (`1 = fraudulent transaction`, `0 = legitimate transacti`
- customer churn prediction (`1 = cutomer leaves`, `0 = customer stays`)



Multiclass classification

Multiclass classification is a type of supervised learning problem where the goal is to assign an input (data point) to one label out of three or more possible classes.

Handwritten digit recognition (MNIST):
Input = image of a digit (0–9).
Output = one of 10 classes.
Fruit classification:
Input = picture of fruit.
Output = {apple, banana, orange, grape, …}.
Language detection:
Input = a sentence.
Output = {English, Spanish, French, German, …}.



Supervised Learning - Regression

Regression is a supervised learning task where the output variable is continuous (numeric).

Linear Regression

Linear regression tries to fit a straight line through the data that best explains the relationship between input (X) and output (Y).

Supervised Learning - Regression

Simple Linear Regression:

One input feature (1D line).
Example: Predicting house price only from house size.

$$\hat{y} = w \cdot x + b$$

$\hat{y}$ = predicted value

x = input (feature)

w = slope (weight) — shows how much y changes when x increases

b = intercept (bias) — the value of y when $x = 0$

To find the best line, we measure the error between predicted values and actual values.
The most common choice is **Mean Squared Error (MSE)**

$$L = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

Supervised Learning - Regression

Multiple Linear Regression

Several input features.
Example: Predicting house price from size, location, number of rooms, etc.

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

$\hat{y}$ = predicted value

x = input (feature)

w = slope (weight) — shows how much y changes when x increases

b = intercept (bias) — the value of y when $x = 0$

Supervised Learning

Key Characteristics:

- Requires labeled data.
- Focuses on prediction.
- Divided into Regression and Classification.

Algorithms:

- Linear Regression
- Logistic Regression
- Decision Trees
- Support Vector Machines (SVM)
- k-Nearest Neighbors (k-NN)
- Neural Networks

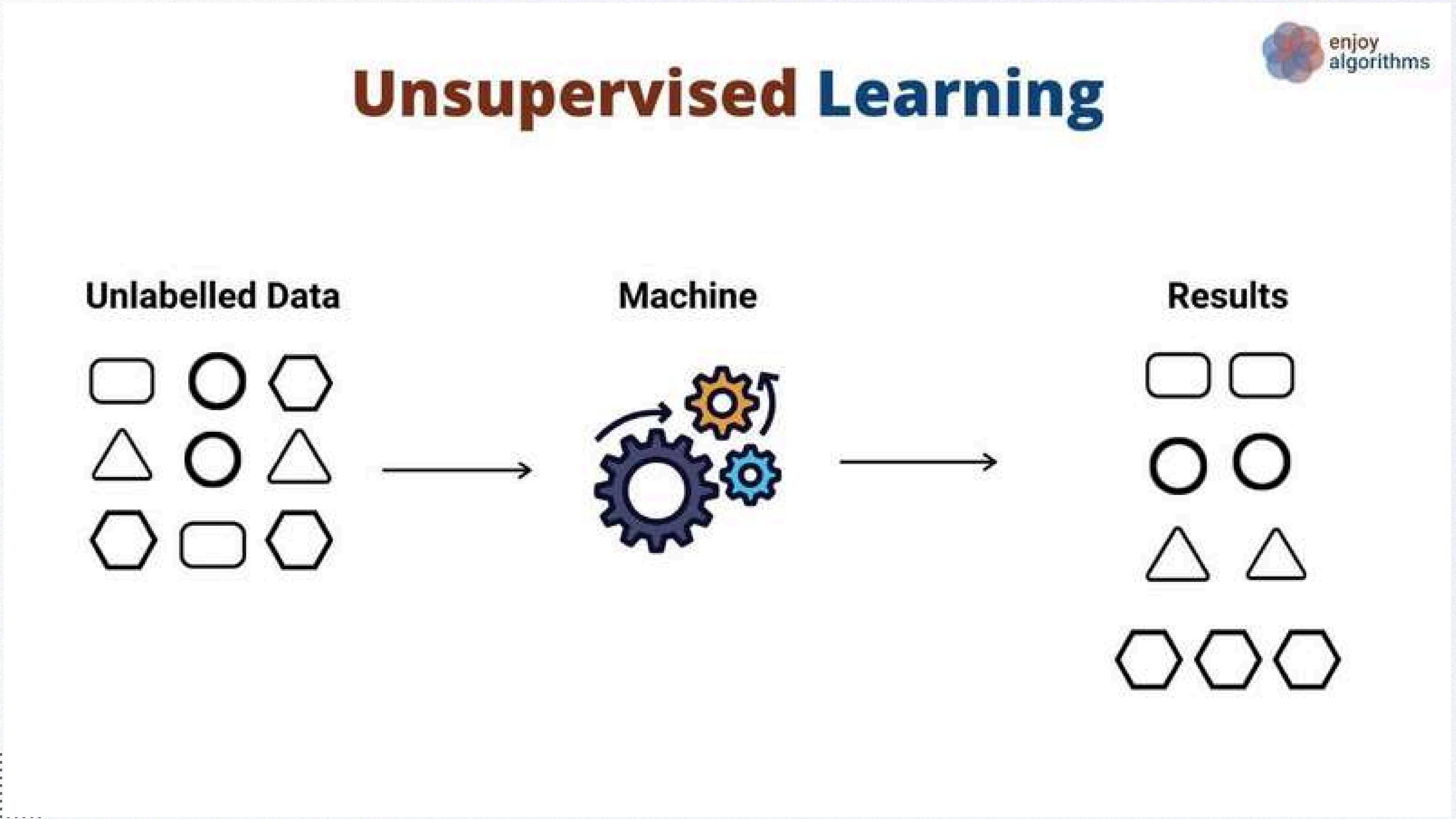
Supervised Learning

Applications:

- Credit scoring (predict default or not)
- Spam detection (spam vs. not spam)
- Medical diagnosis (disease classification)
- Price prediction (house, stock, insurance)

Unsupervised learning

Unsupervised learning involves training a model on data that does not have labeled responses. **The goal is** to discover underlying patterns or structures in the data, such as grouping similar items together.



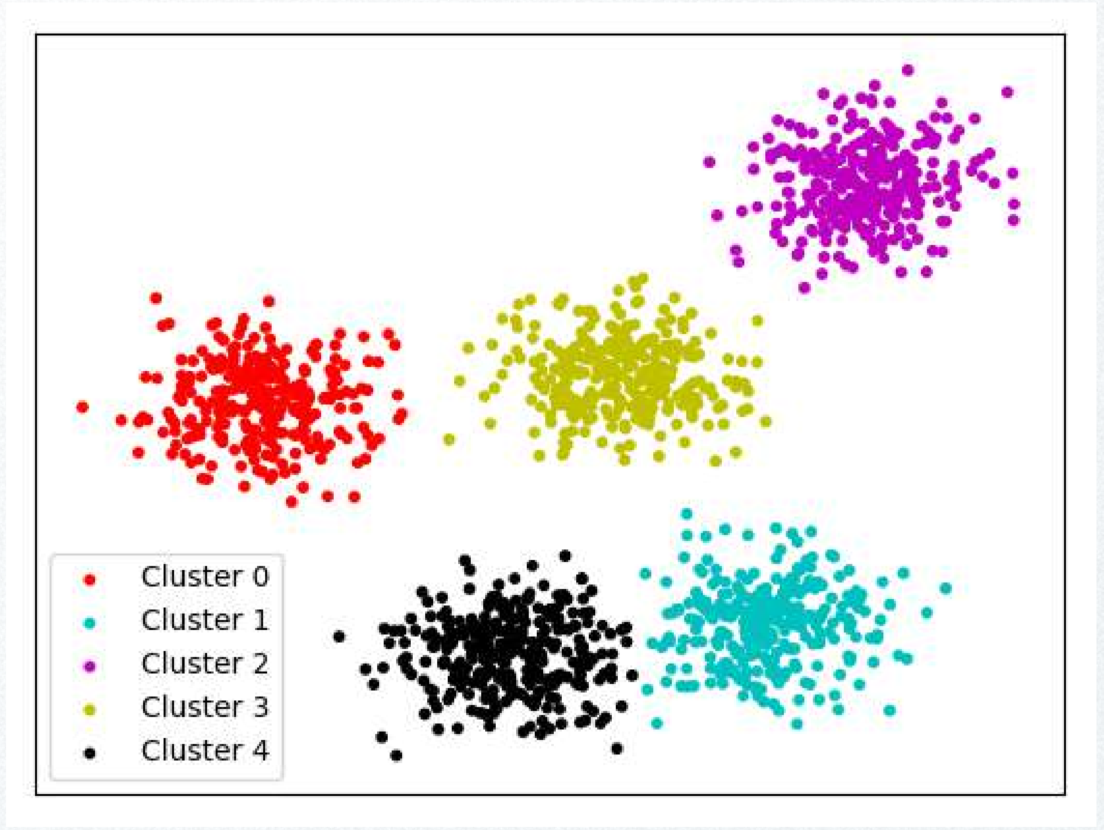
Unsupervised learning deals with unlabeled data. The algorithm tries to discover hidden patterns, structures, or groupings in the dataset without knowing the “right answer.”

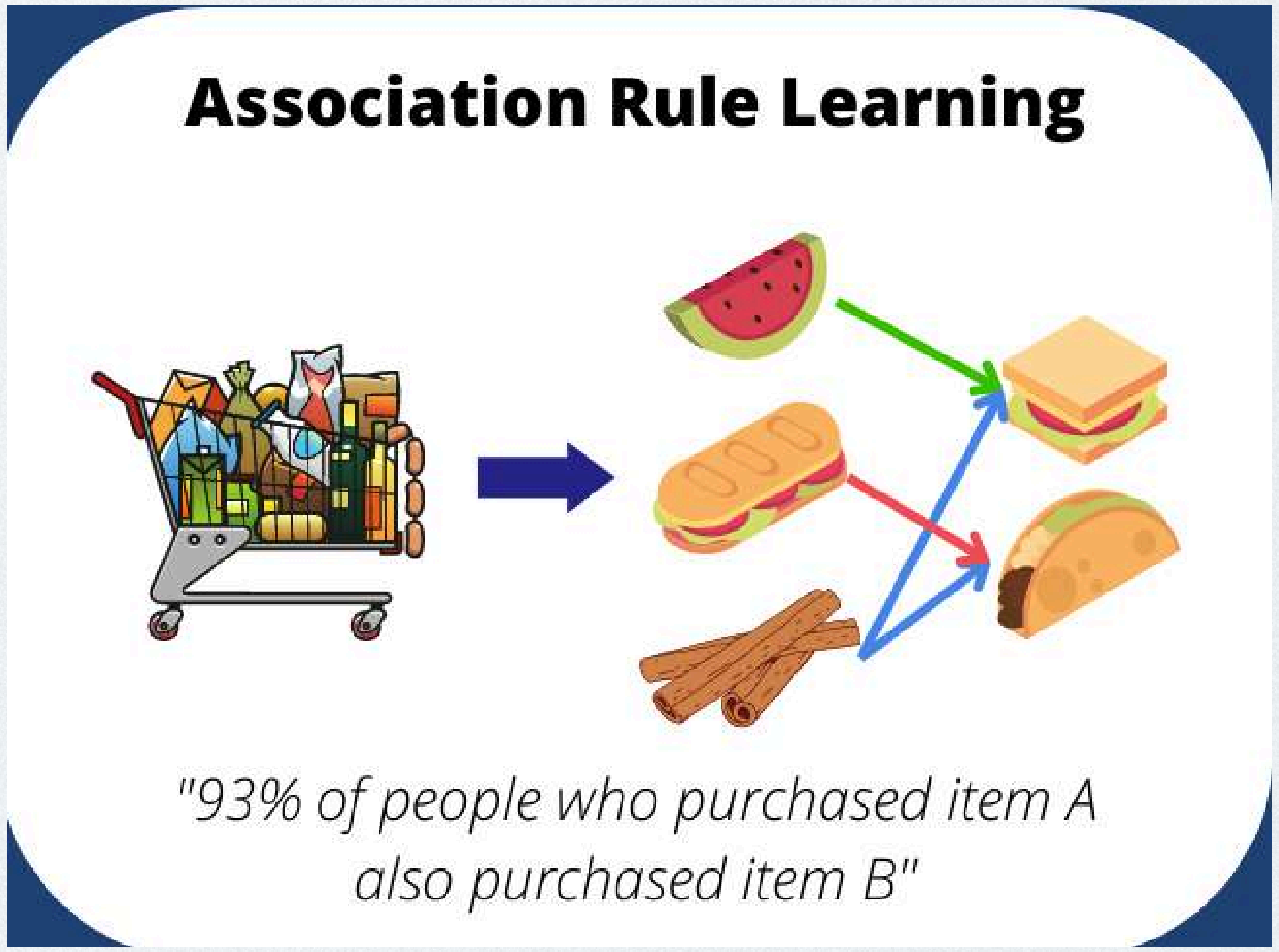
Unsupervised learning

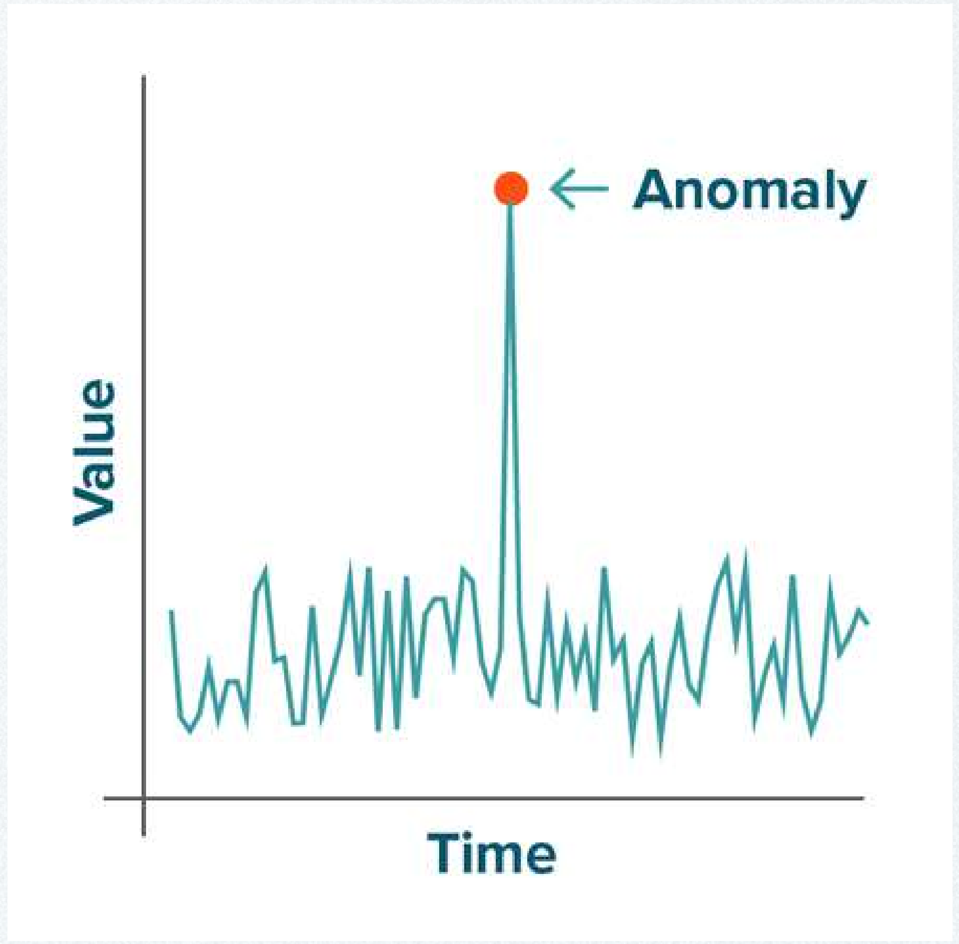
Market segmentation: Grouping customers into segments
Social network analysis: Identifying communities of friends.
Anomaly detection: Detecting unusual transactions that may be fraud.

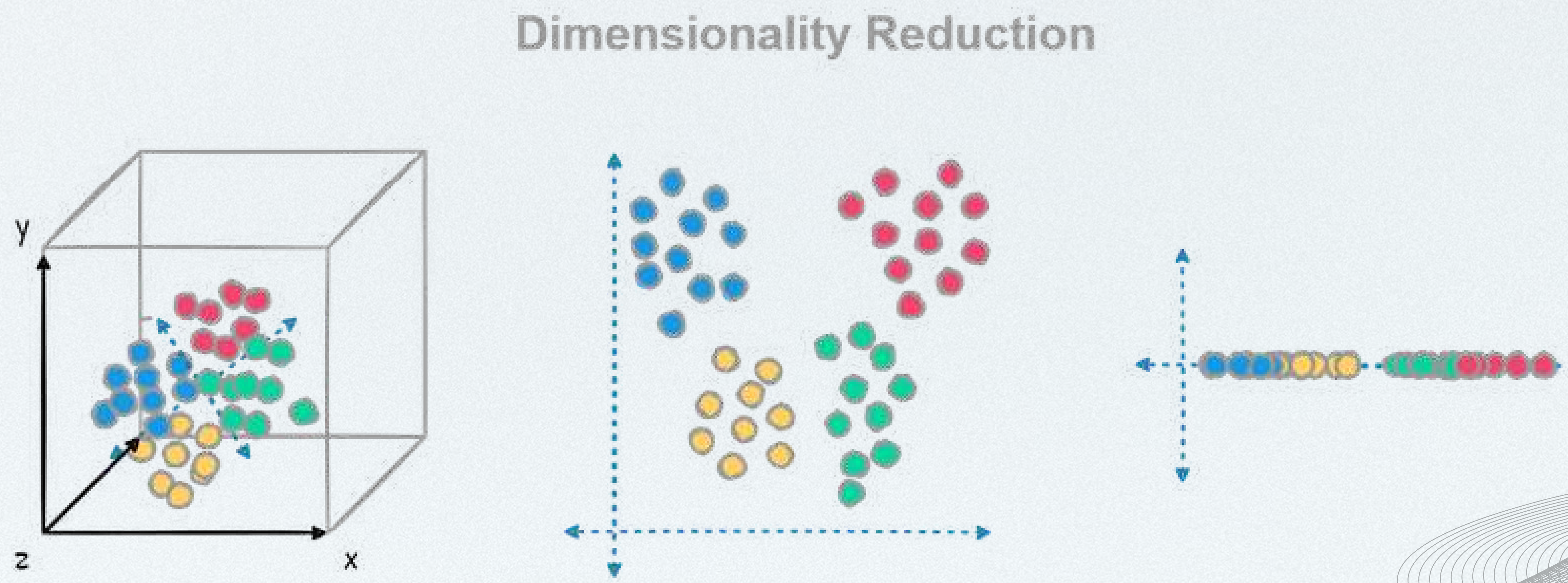
Clustering: k-Means, Hierarchical clustering, DBSCAN.
Dimensionality Reduction: Principal Component Analysis (PCA).
Association Rule Mining

Unsupervised learning




Association Rule Learning
"93% of people who purchased item A also purchased item B"





Real-World Applications

Banking & Finance

- Regression: Predict loan default probability.
- Classification: Fraud detection.
- Clustering: Segmenting customers by risk.

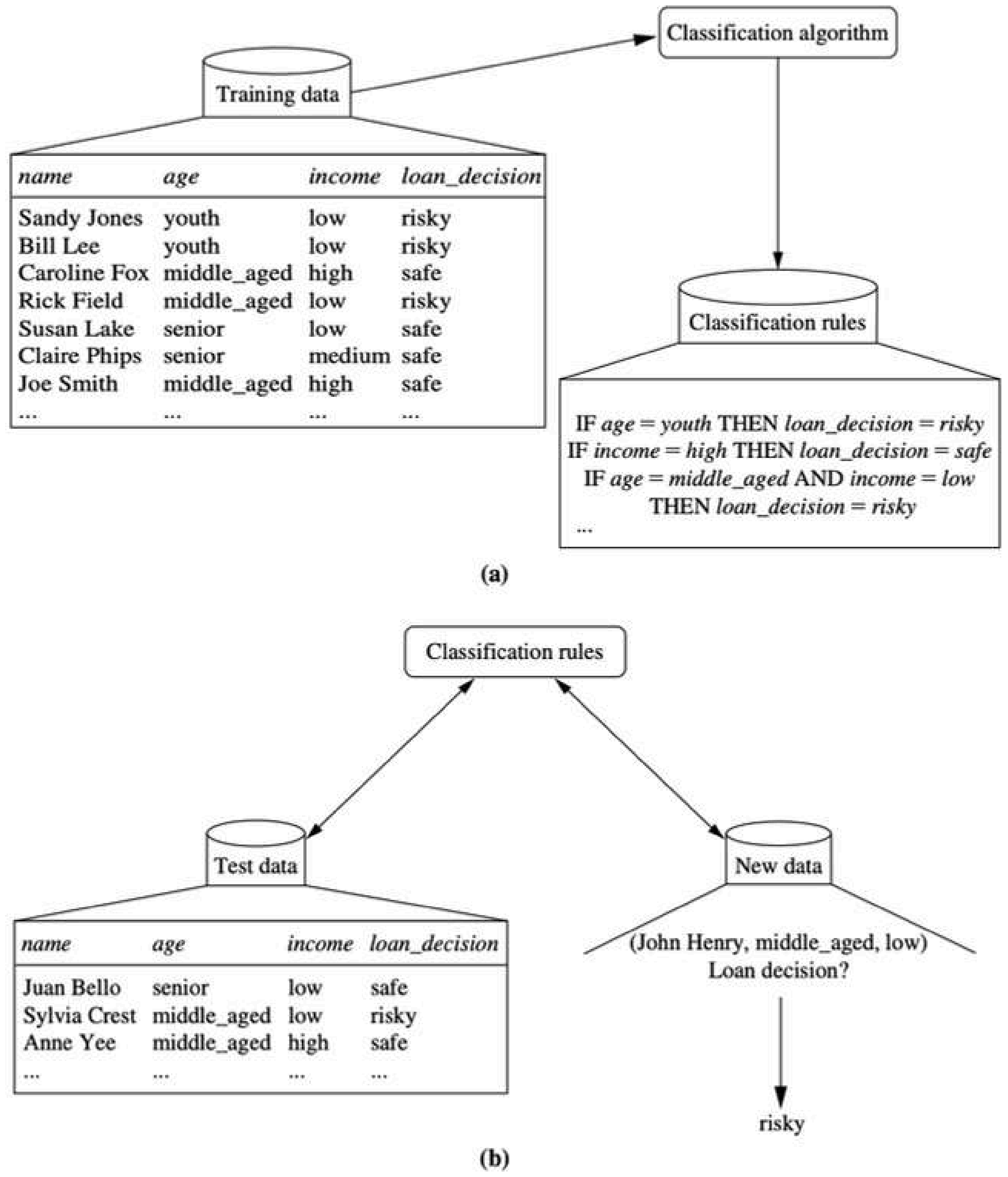
Healthcare

- Regression: Predict length of hospital stay.
- Classification: Diagnose disease from test results.
- Clustering: Group patients with similar symptoms.

E-commerce

- Regression: Predict sales revenue.
- Classification: Recommend “buy” or “don’t buy.”
- Clustering: Personalized customer groups.

Classification problems



name	*age*	*income*	*loan_decision*
Sandy Jones	youth	low	risky
Bill Lee	youth	low	risky
Caroline Fox	middle_aged	high	safe
Rick Field	middle_aged	low	risky
Susan Lake	senior	low	safe
Claire Phips	senior	medium	safe
Joe Smith	middle_aged	high	safe
...	...	...	...

IF *age* = *youth* THEN *loan_decision* = *risky*
IF *income* = *high* THEN *loan_decision* = *safe*
IF *age* = *middle_aged* AND *income* = *low*
THEN *loan_decision* = *risky*
...

name	*age*	*income*	*loan_decision*
Juan Bello	senior	low	safe
Sylvia Crest	middle_aged	low	risky
Anne Yee	middle_aged	high	safe
...	...	...	...

The data classification process:
(a) Learning: Training data are analyzed by a classification algorithm. Here, the class label attribute is loan decision, and the learned model or classifier is represented in the form of classification rules.

(b) Classification: Test data are used to estimate the accuracy of the classification rules. If the accuracy is considered acceptable, the rules can be applied to the classification of new data tuples.

Logistic Regression

Logistic Regression is one of the most fundamental algorithms in classification problems, especially binary classification (e.g., predicting “yes/no,” “spam/not spam,” “default/no default”).

Although it has "regression" in its name, it is used for classification, not regression. The term comes from the use of a linear model combined with a logistic (sigmoid) function to predict probabilities.

If we apply linear regression: $\hat{y} = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$

We might get predictions like −0.3 or 1.5, which don’t make sense for probabilities.
We need outputs between 0 and 1. This is why we apply a transformation — the **sigmoid function**.

The **sigmoid function** squashes any real number into the range (0, 1):

$$\sigma(z) = \frac{1}{1 + e^{-z}}$$

Where $z = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$

Logistic Regression

Linear regression models continuous outcomes. Logistic regression, a type of generalized linear model, extends this idea to categorical (binary) outcomes.

It estimates the probability p of an event (e.g., good credit rating) instead of predicting a numeric value.
If p > 0.5, the outcome is classified as YES; otherwise, NO.

Logistic regression is used when the dependent variable is binary (0 or 1), while input variables can be quantitative or categorical.

$$P(y=1|x) = \frac{1}{1+e^{-(\beta_0+\beta_1 x_1+\cdots+\beta_n x_n)}}$$

The logistic regression model estimates the probability that an observation belongs to class 1:

$$P(y=0|x) = 1 - P(y=1|x)$$

Support Vector Machines (SVM)

SVM aims to find a hyperplane that best separates data points of different classes.
A hyperplane is a line (in 2D), a plane (in 3D), or a higher-dimensional surface that divides the data space.

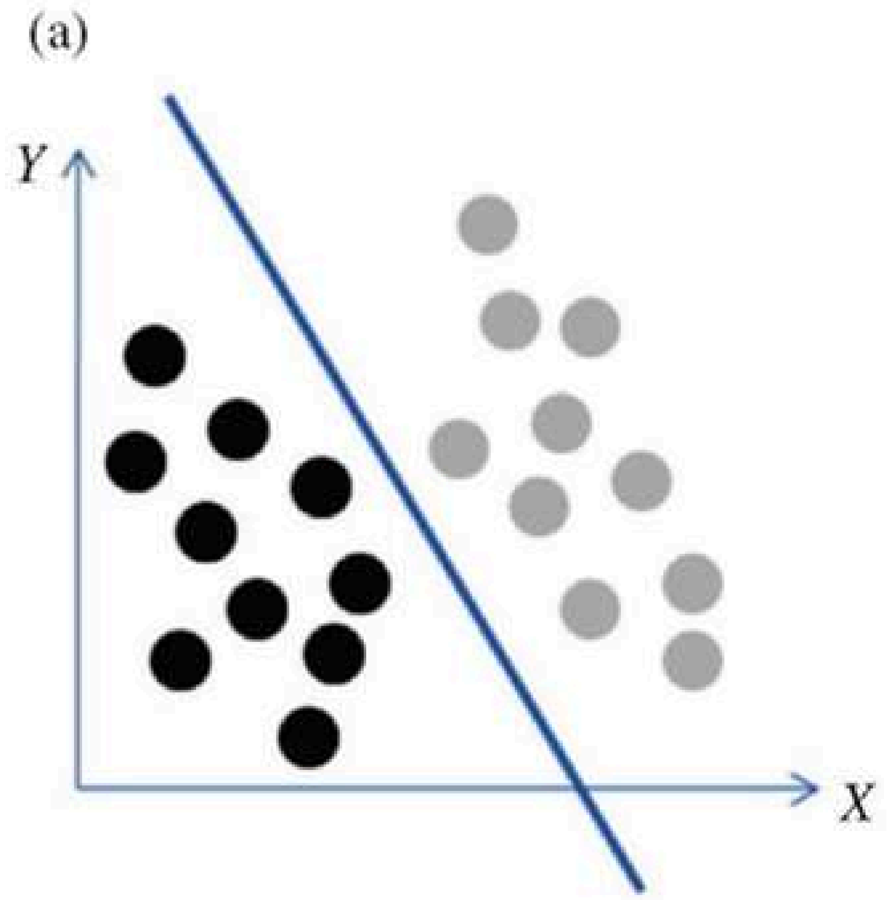
SVM are supervised learning algorithms used for classification tasks (+ regression*). They are based on the idea of finding the optimal boundary (hyperplane) that best separates different classes in the feature space.

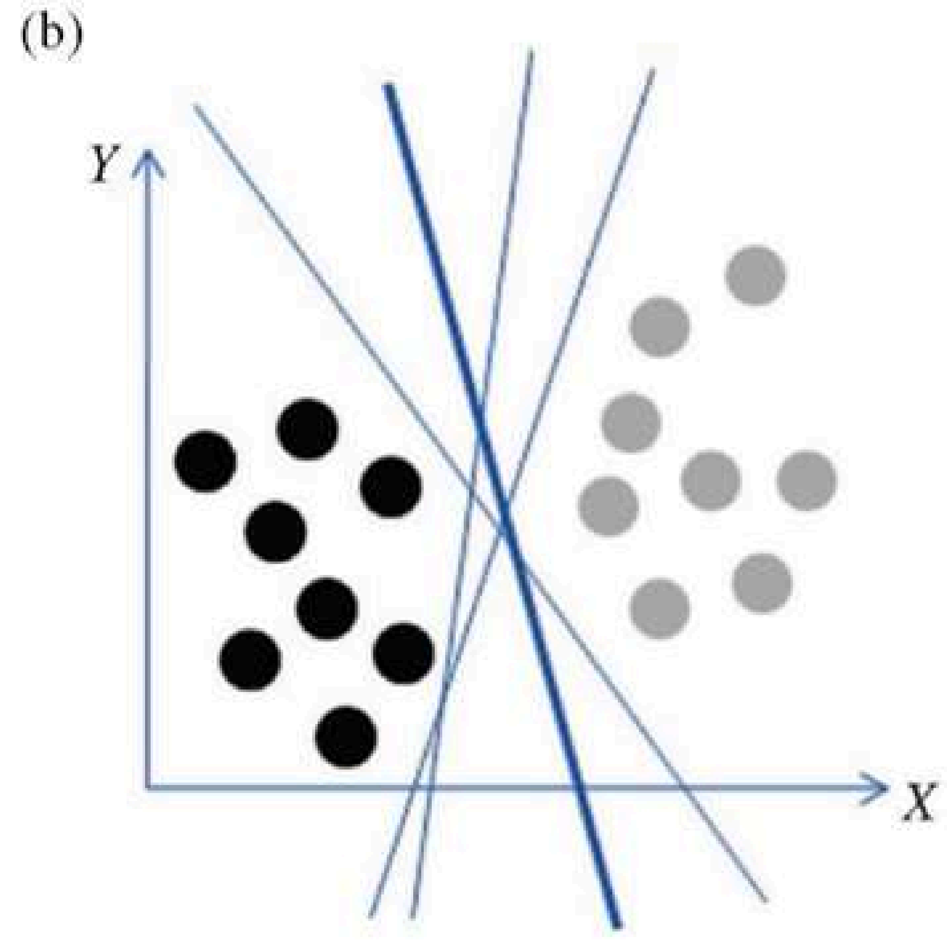
The margin is the distance between the separating hyperplane and the closest data points from either class.
The support vectors are those closest data points — they “support” or define the position of the hyperplane.

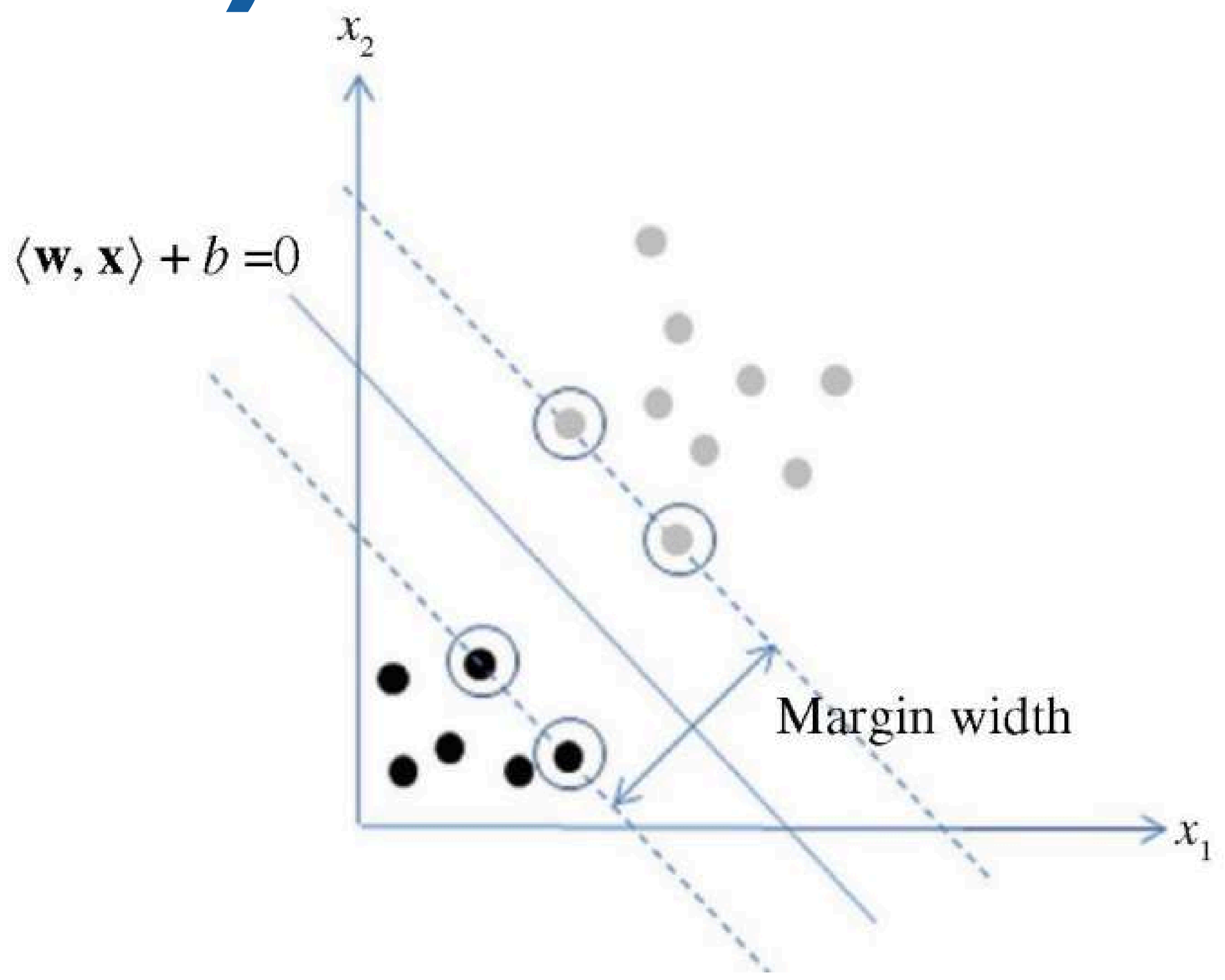
hyperplane $w \cdot x + b = 0$

***by the introduction of an alternative loss function that is modified to include a distance measure**

Support Vector Machines (SVM)

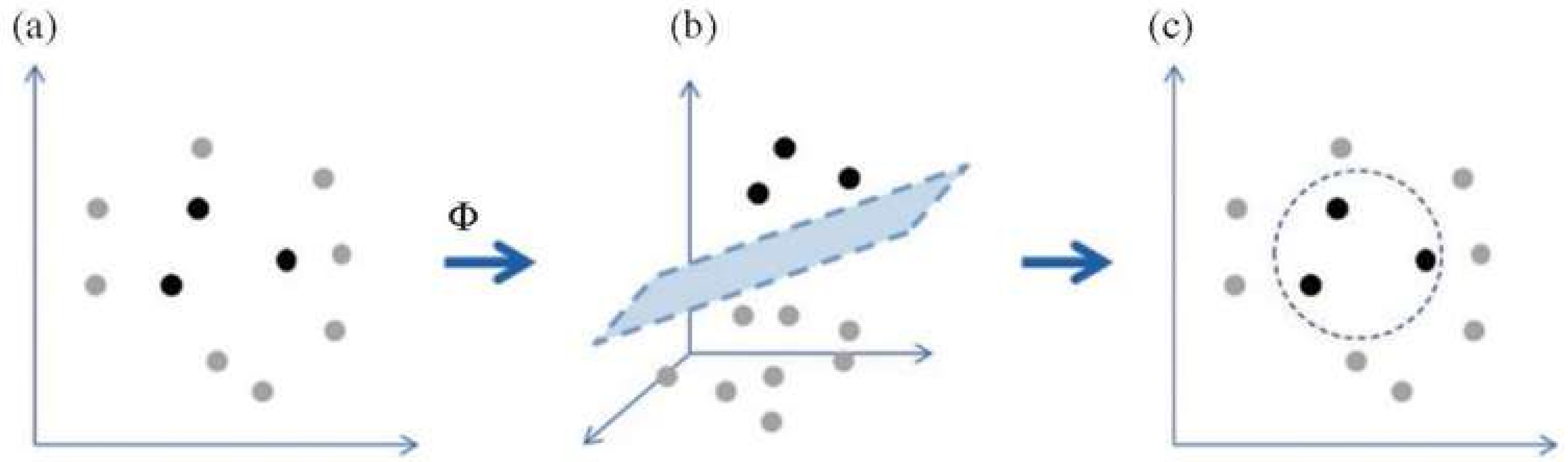






The main idea is that the decision boundary should be as far away as possible from the data points of both classes. There is only one that maximizes the margin (maximizes the distance between it and the nearest data point of each class). Intuitively, the margin is defined as the amount of space or separation between the two classes as defined by the hyperplane. Geometrically, the margin corresponds to the shortest distance between the closest data points to a point on the hyperplane.

Support Vector Machines (SVM)



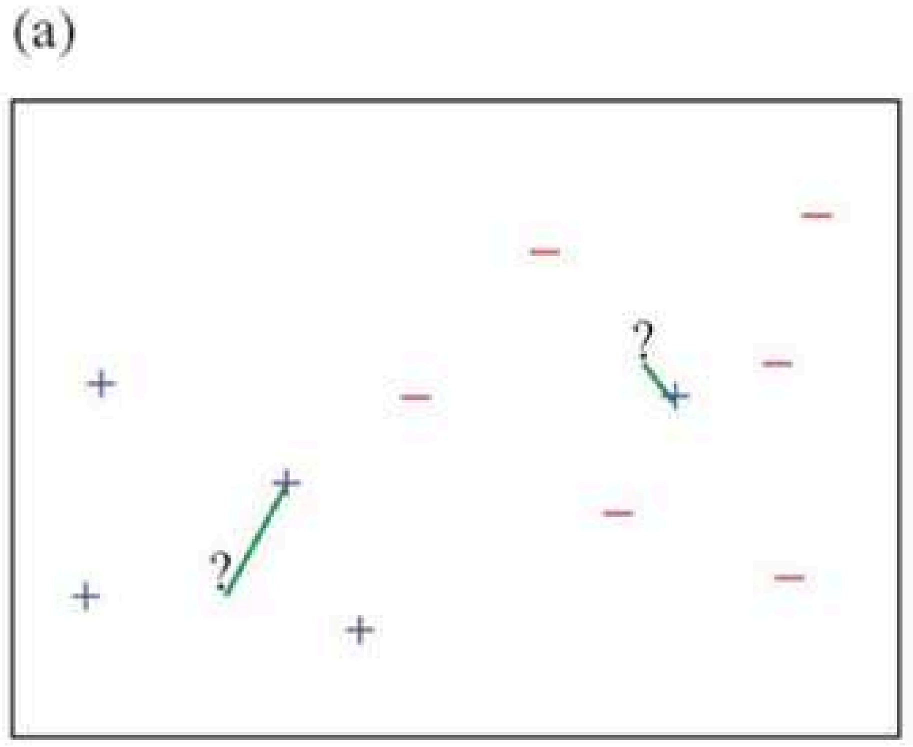
SVM performs nonlinear classification by kernel-based transformations.

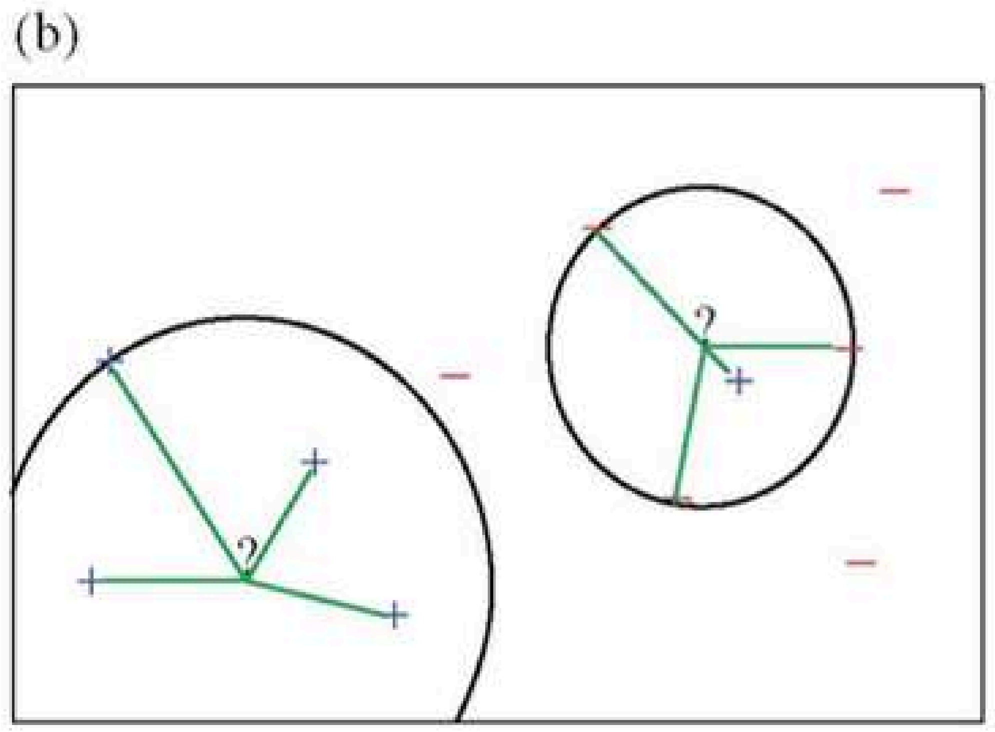
kNN: NEAREST NEIGHBOR CLASSIFIER

kNN is a non-parametric algorithm — it makes predictions based on the closest training examples in the feature space.

Unlike SVM's global classification model, k-nearest neighbor (kNN) classifier determines the decision boundary locally. For 1NN we assign each new sample to the class of its closest neighbor as it is represented in Figure below. Initially we have samples belonging to two classes (+ and -).
The new sample "?" should be labeled with the class of its closest neighbor. 1NN classifier is not very robust methodology.

The classification decision of each test sample relies on the class of a single training sample, which may be incorrectly labeled or atypical. For larger k, kNN will assign new sample to the majority class of its k closest neighbors where k is a parameter of the methodology.
An example for k = 4 is given in Figure below. kNN classifier for k > 1 is more robust. Larger k values help reduce the effects of noisy points within the training data set.





$$d(x_i, x_j) = \sqrt{\sum_{n=1}^{N} (x_{i,n} - x_{j,n})^2}$$

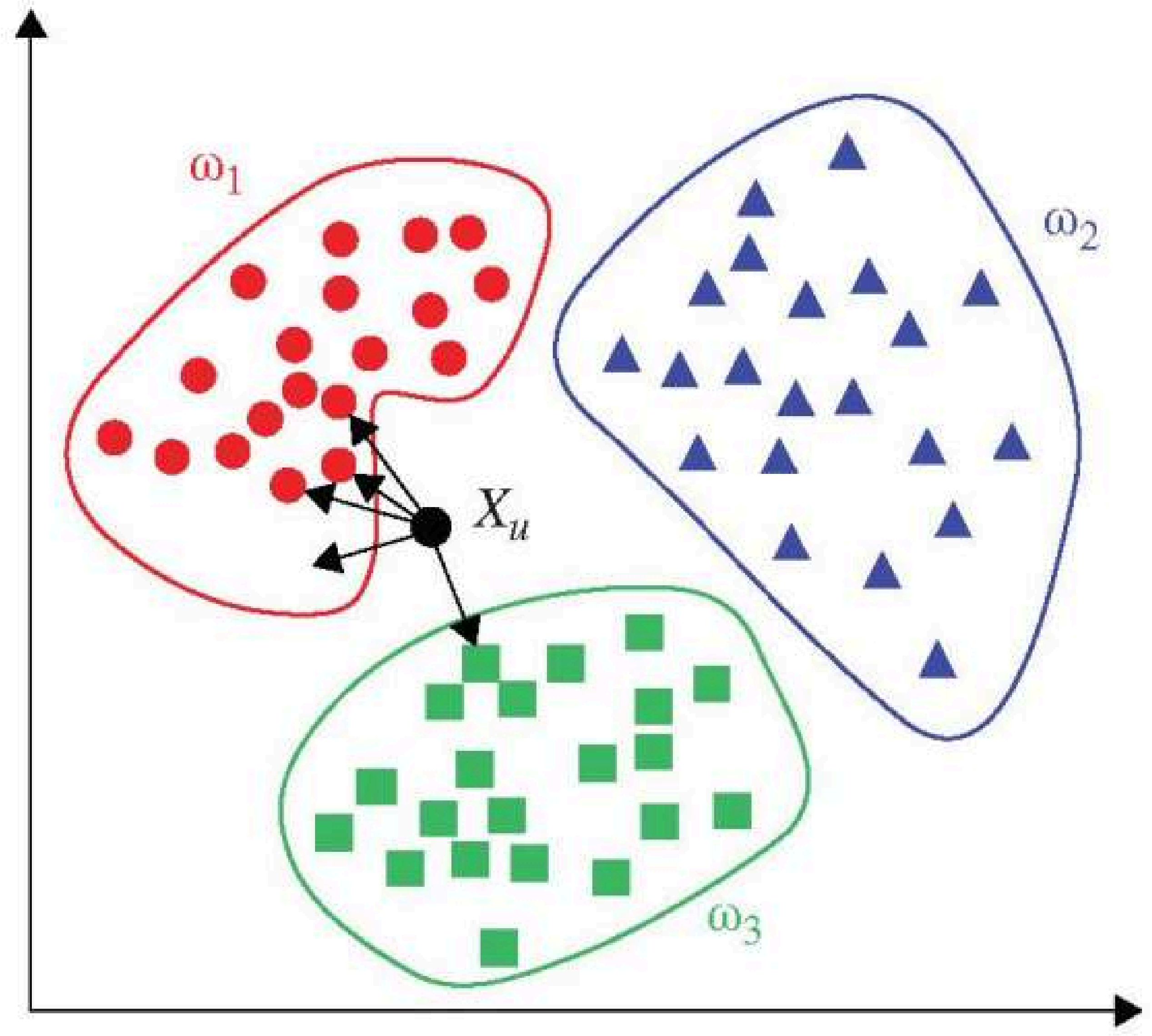
Calculate the distance between the test point and all training samples

kNN: NEAREST NEIGHBOR CLASSIFIER

In summary, kNN classifier only requires a parameter k, a set of labeled training samples, and a metric measure for determining distances in n-dimensional space. kNN classification process is usually based on the following steps:

- Determine parameter k-number of nearest neighbors.
- Calculate the distance between each testing sample and all the training samples.
- Sort the distance and determine nearest neighbors based on the k-th threshold.
- Determine the category (class) for each of the nearest neighbors.
- Use simple majority of the category of nearest neighbors as the prediction value of the testing sample classification.

"Similar data points are likely to have similar outcomes."



Decision Tree Classifier

A decision tree is a tree-structured model that recursively splits data according to feature conditions.

A Decision Tree Classifier is a supervised learning algorithm used for classification and regression tasks. It predicts the class (output) of an object by learning simple decision rules inferred from data features.

It **mimics human decision-making** — splitting data into smaller subsets based on feature tests (e.g., “Is age > 30?”), forming a tree structure.

```
            [Age > 30?]
            /          \
        Yes              No
    [Income > 50k?]        Class = B
     /        \
Class = A     Class = B
```

Root Node – represents the entire dataset.
Internal Nodes – represent tests on features.
Branches – represent outcomes of those tests.
Leaf Nodes – represent the final class label or output value.

Decision Tree Classifier

Entropy

Entropy measures uncertainty or impurity in the dataset.

$$H(S) = -\sum_{i=1}^{C} p_i \log_2(p_i)$$

Low entropy → dataset is pure (mostly one class)
High entropy → dataset is impure (mixed classes)

where

- p_i = proportion of samples belonging to class i
- C = number of classes
- If all examples belong to one class → Entropy = 0 (pure node).
- If evenly split between classes → Entropy = 1 (maximum impurity).

When building a Decision Tree, the goal is to split data such that each subset is purer (less mixed) than before.
To decide which feature to split on, we measure how much entropy decreases after the split.

Entropy gives us a way to measure how disordered a set is before and after a split.

Decision Tree Classifier

Information Gain

Information Gain (IG) measures how much “information” a feature gives about the class.

$$IG(S,A) = H(S) - \sum_{v \in Values(A)} \frac{|S_v|}{|S|} H(S_v)$$

where:

- S: the entire dataset
- A: the feature we're splitting on
- S_v: subset of S for which feature A takes value v
- $H(S)$: entropy before the split
- $H(S_v)$: entropy after the split
- $\frac{|S_v|}{|S|}$: weight (proportion of samples in subset)

The original dataset has some disorder (entropy).
When we split it by a feature, it breaks into smaller subsets.
If those subsets are purer than before (less mixed), then the feature gives us information about the class.
The higher the Information Gain, the better the feature for splitting.

Decision Tree Classifier

- Entropy measures how uncertain the dataset is.
- Information Gain measures how much that uncertainty is reduced when we split.
- The Decision Tree algorithm uses Information Gain to pick the best feature at each step.
- The process repeats recursively, forming a tree that best separates the classes.

Thank You!