



## DAY 8

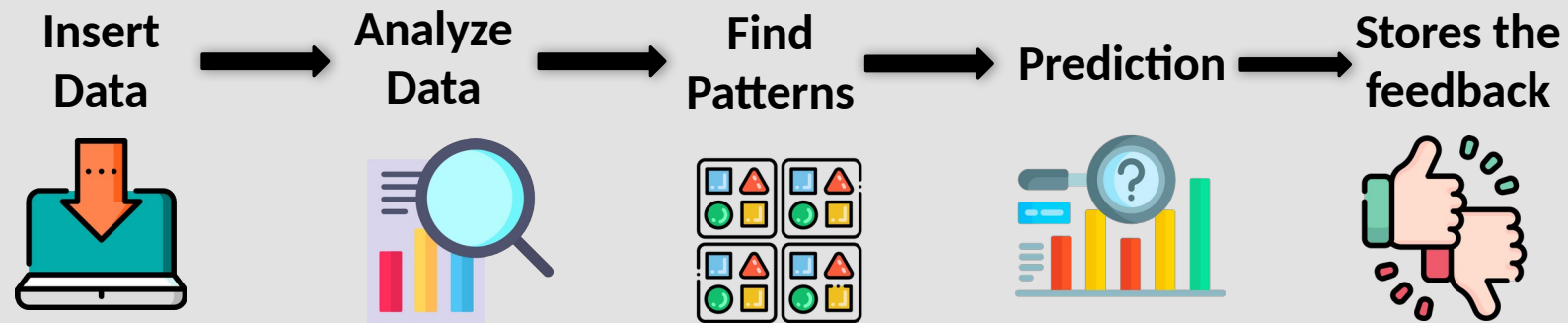
---

Instructor: Balu Mohandas Menon

---

Christian B. Wiberg  
Philip Jess Teining

# HOW DOES ML WORK?



<https://data-flair.training/blogs/machine-learning-tutorial/>

# TYPES OF ML



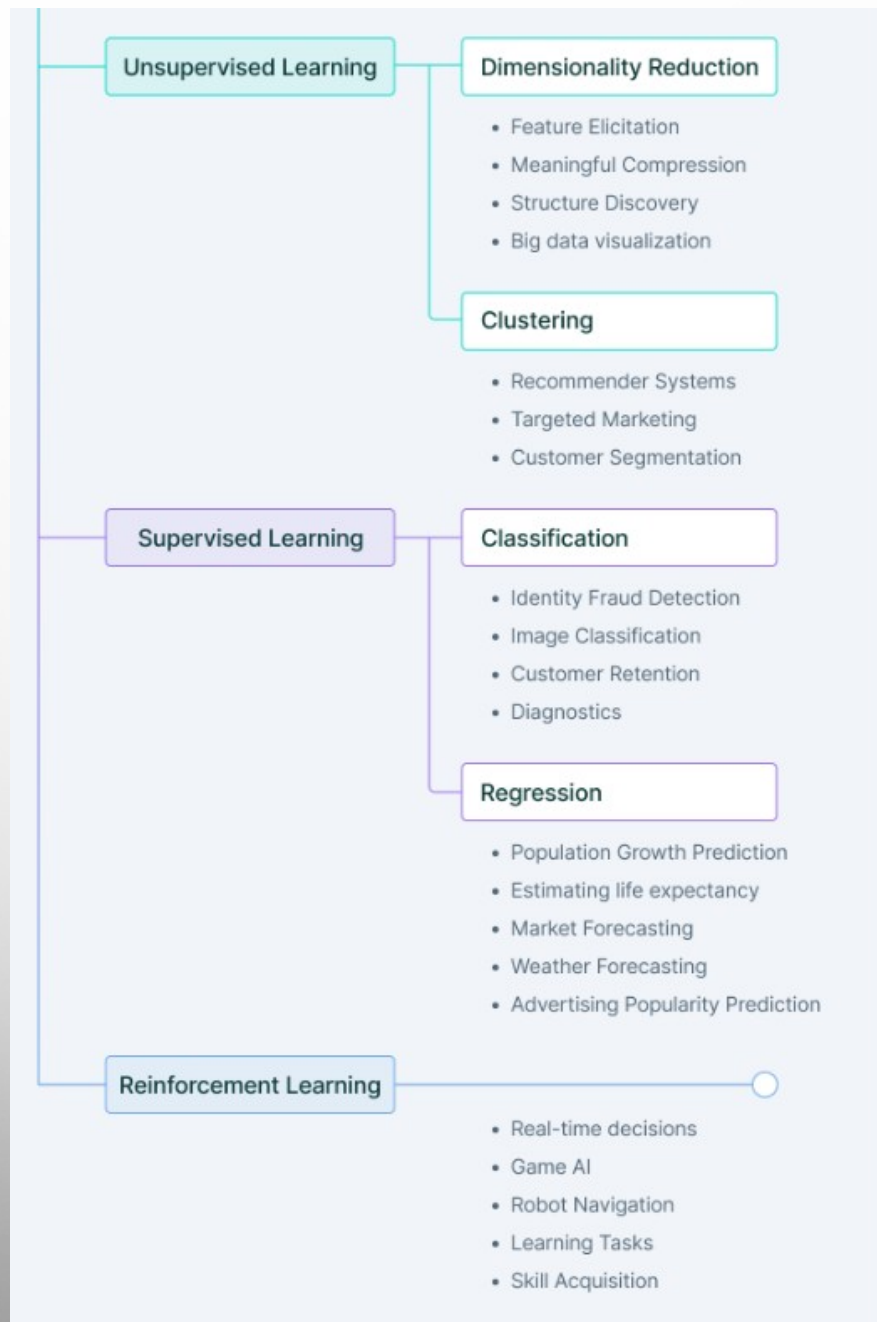
Supervised  
learning



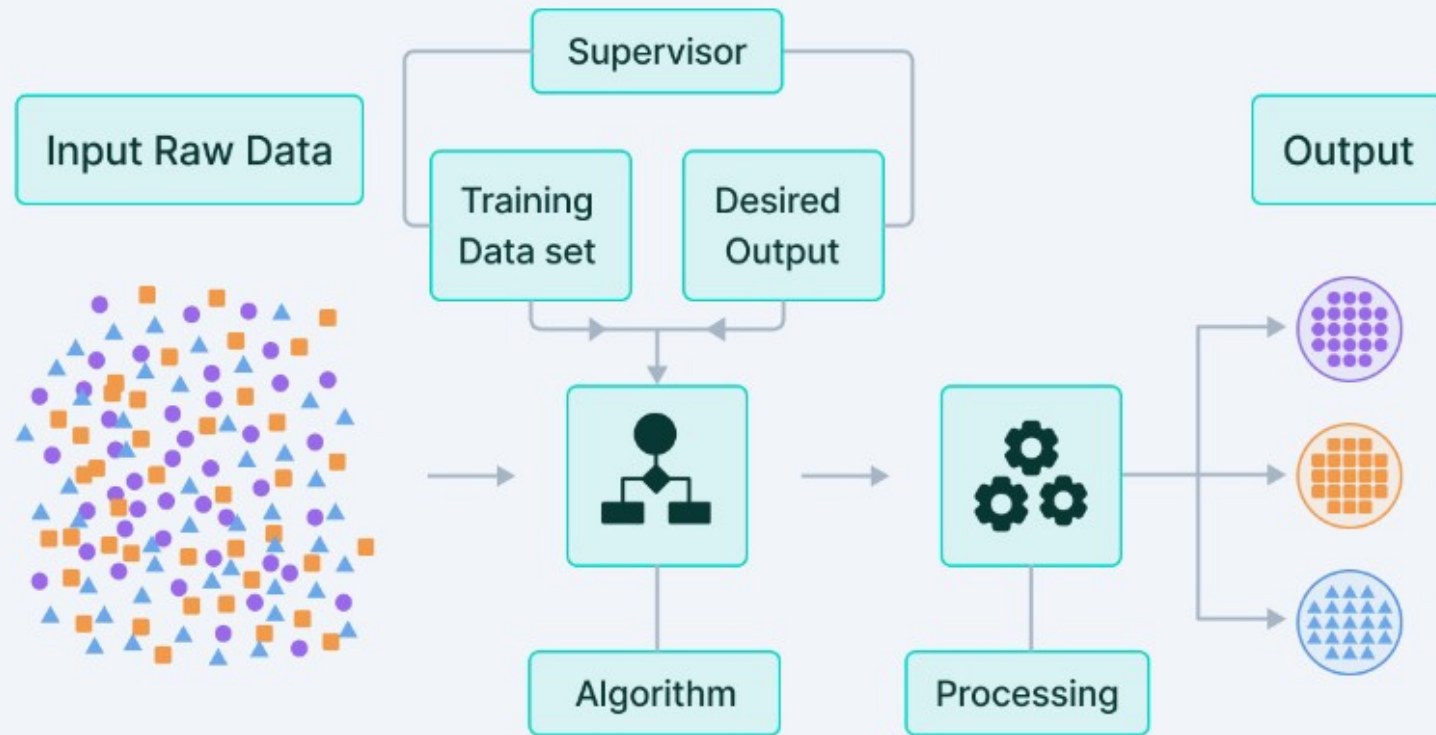
Unsupervised  
learning



Reinforcement  
learning



# Supervised Learning



## SUPERVISED MACHINE LEARNING METHODS



**Classification** : Classification refers to taking an input and assigning it to a predefined category or class based on its characteristics.



Output typically consists of categories, ex: whether it is going to rain today or not.

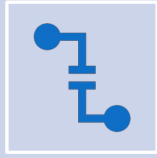


**Regression** : predicted output values are real numbers.



Predicting the price of a house or the trend in the stock price at a given time, etc.

## UNSUPERVISED MACHINE LEARNING METHODS



**Clustering** : Clustering is the type of Unsupervised Learning where we find hidden patterns in the data based on their similarities or difference.



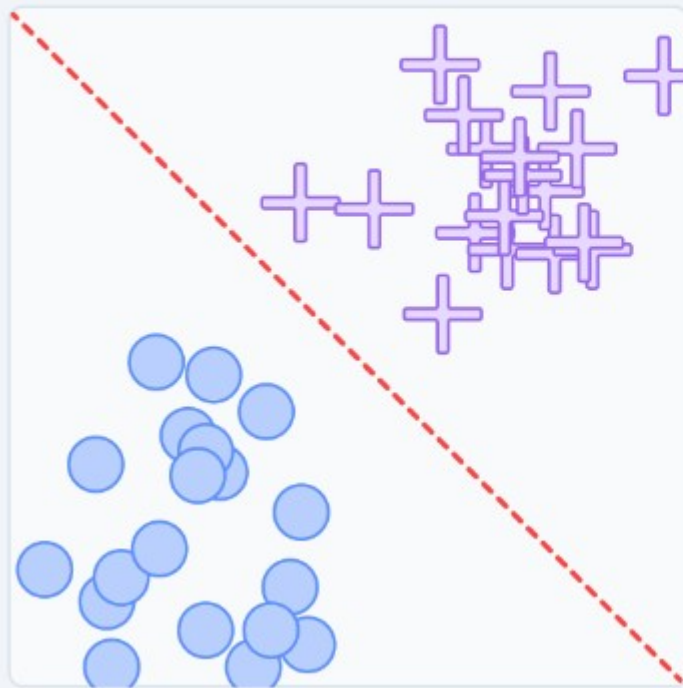
**Association** : Association is the kind of Unsupervised Learning where we can find the relationship of one data item to another data item.



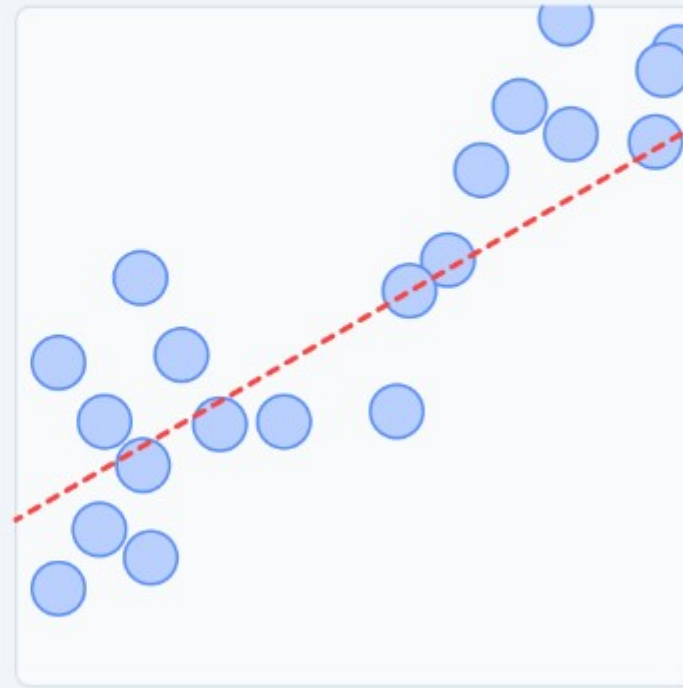
e.g., understanding consumers' habits regarding our products can help us develop better cross-selling strategies.



**Classification**



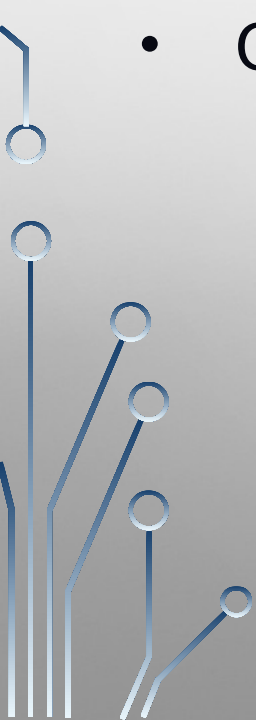
**Regression**



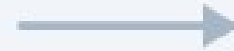




## SUPERVISED MACHINE LEARNING APPLICATIONS

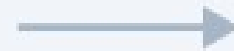
- Predictive analytics (house prices, stock exchange prices, etc.)
    - Text recognition
    - Spam detection
    - Customer sentiment analysis
    - Object detection (e.g. face detection)
- 

**Supervised Learning**



**Labeled Data**

**Unsupervised Learning**



**Unlabeled Data**



***Supervised Learning*** learns from the training dataset by iteratively making predictions on the data and adjusting for the correct answer.



Supervised techniques deal with labeled data where the output data patterns are known to the system.



***Unsupervised Learning*** models work on their own to discover the inherent structure of unlabeled data.



The unsupervised learning algorithm works with unlabeled data, in which the output is based solely on the collection of perceptions.

### Supervised learning

Input data is labeled

Has a feedback mechanism

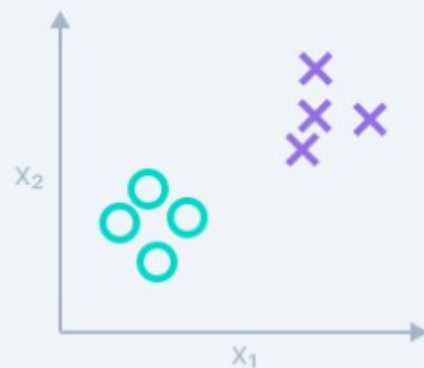
Data is classified based on the training dataset

Divided into Regression & Classification

Used for prediction

Algorithms include: decision trees, logistic regressions, support vector machine

A known number of classes



### Unsupervised learning

Input data is unlabeled

Has no feedback mechanism

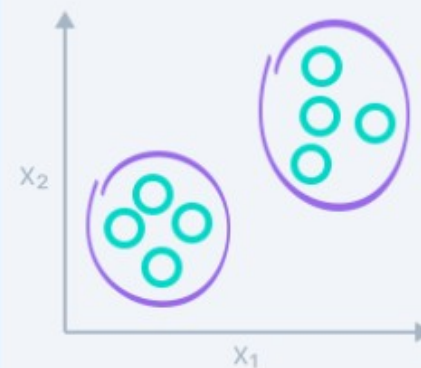
Assigns properties of given data to classify it

Divided into Clustering & Association

Used for analysis

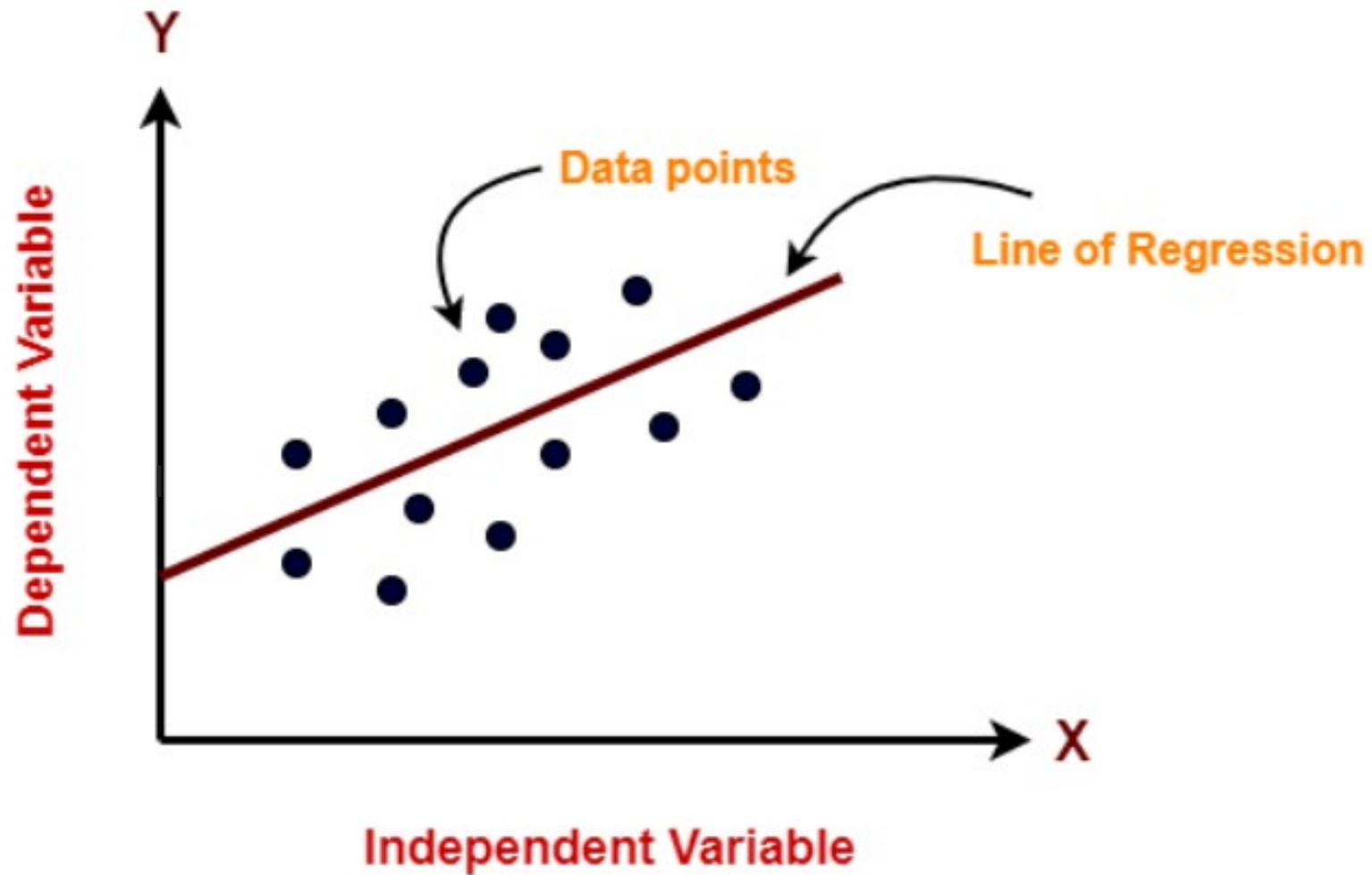
Algorithms include: k-means clustering, hierarchical clustering, apriori algorithm

A unknown number of classes

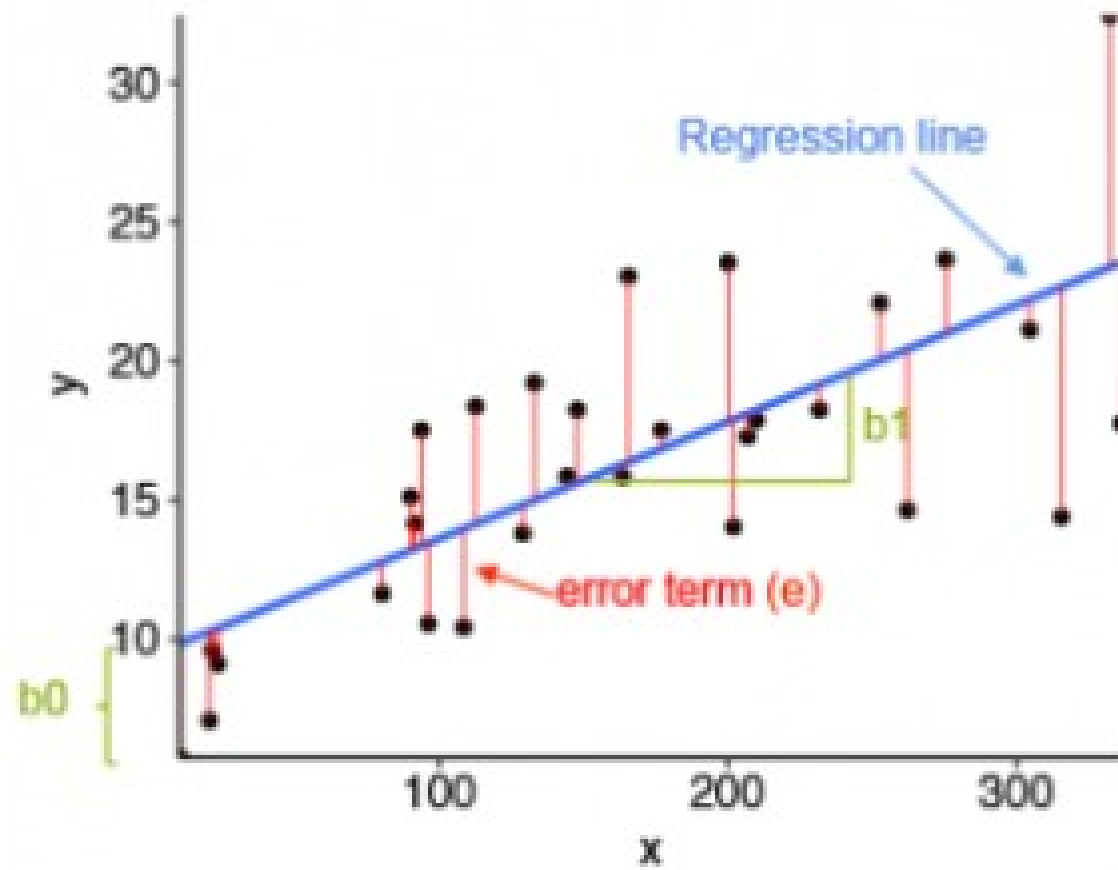


# REGRESSION TECHNIQUES

- Regression is a technique for investigating the relationship between independent variables or features and a dependent variable or outcome. It's used as a method for predictive modelling in **machine learning**, in which an algorithm is used to predict continuous outcomes. Regression algorithms include:
  - 1. Linear Regression
  - 2. Decision Tree
  - 3. Support Vector Regression
  - 4. Lasso Regression
  - 5. Random Forest



In the figure above, on X-axis is the independent variable and on Y-axis is the output. The regression line is the best fit line for a model. And our main objective in this algorithm is to find this best fit line.





# DECISION TREE



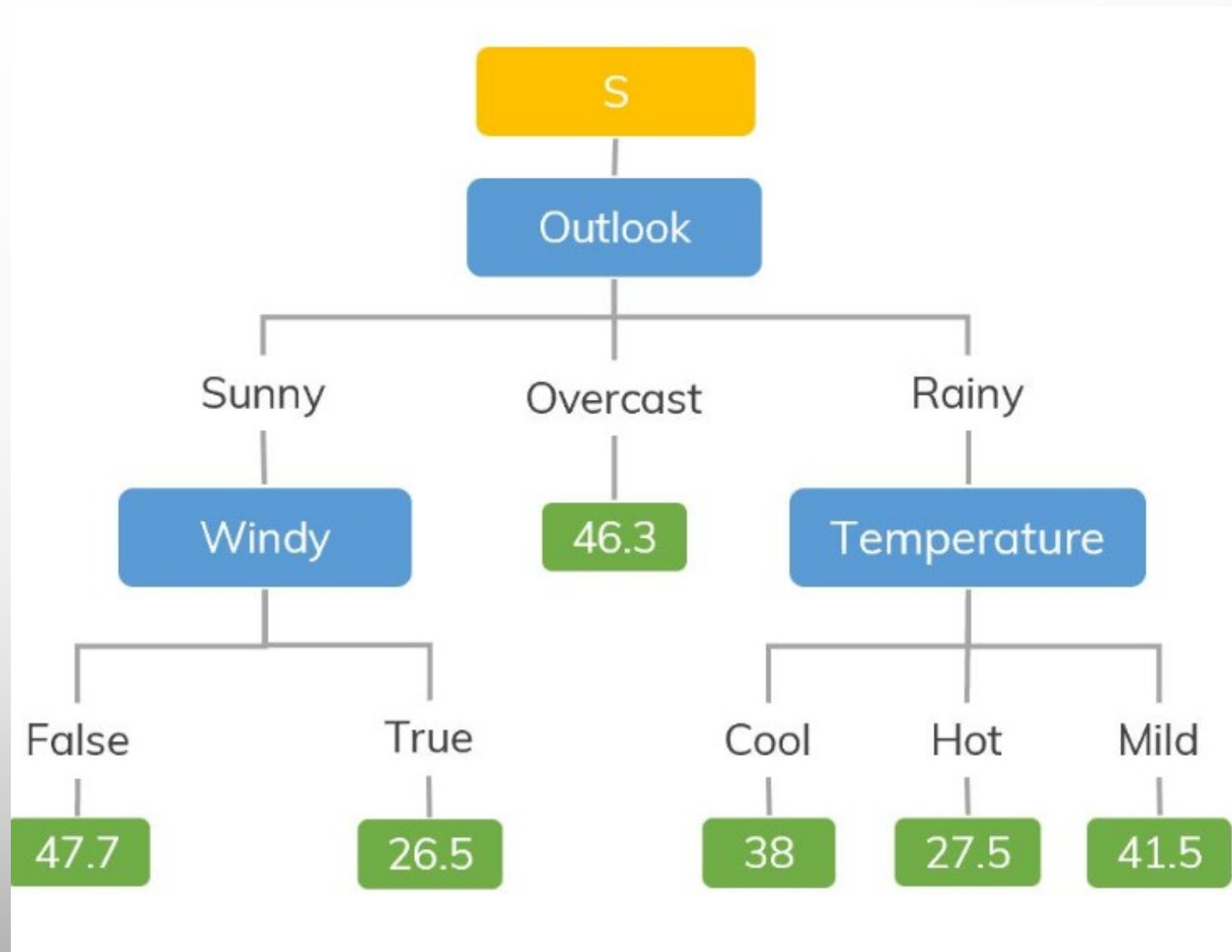
The decision tree models can be applied to all those data which contains numerical features and categorical features.



Decision trees are good at capturing non-linear interaction between the features and the target variable.



Decision trees somewhat match human-level thinking so it's very intuitive to understand the data.



For example, if we are classifying how many hours a kid plays in particular weather then the decision tree looks like somewhat this

## OVERVIEW OF BIAS AND VARIANCE

- The goal of any supervised machine learning algorithm is to best estimate the mapping function ( $f$ ) for the output variable ( $Y$ ) given the input data ( $X$ ). The mapping function is often called the target function because it is the function that a given supervised machine learning algorithm aims to approximate.
- The prediction error for any machine learning algorithm can be broken down into three parts:
  - Bias Error
  - Variance Error
  - Irreducible Error

## BIAS ERROR

**Low Bias:** Suggests less assumptions about the form of the target function.

**High-Bias:** Suggests more assumptions about the form of the target function

Examples of **low-bias** machine learning algorithms include: Decision Trees, k-Nearest Neighbors and Support Vector Machines.

Examples of **high-bias** machine learning algorithms include: Linear Regression, Linear Discriminant Analysis and Logistic Regression.

## VARIANCE ERROR

**Low Variance:** Suggests small changes to the estimate of the target function with changes to the training dataset.

**High Variance:** Suggests large changes to the estimate of the target function with changes to the training dataset.

Examples of **low-variance** machine learning algorithms include: Linear Regression, Linear Discriminant Analysis and Logistic Regression.

Examples of **high-variance** machine learning algorithms include: Decision Trees, k-Nearest Neighbors and Support Vector Machines.

# BIAS-VARIANCE TRADE-OFF

- The goal of any supervised machine learning algorithm is to achieve low bias and low variance. In turn the algorithm should achieve good prediction performance.

# EXAMPLE WITH A DATASET



Suppose we have a dataset of students' exam scores, and we now include two additional feature variables: the number of hours they studied and the amount of sleep they got the night before the exam. Our goal is to build a model that predicts exam scores based on these two features.



**Bias:** If we use a linear regression model with only the number of study hours as a feature, the model might have high bias. It assumes a linear relationship between study hours and exam scores while ignoring the effect of sleep. This oversimplified assumption restricts the model's understanding and may lead to biased predictions. The model fails to capture the influence of sleep on exam performance and overlooks any non-linear relationship between the features and the target variable.



**Variance:** On the other hand, if we use a highly complex model, such as a deep neural network, that considers both study hours and sleep as features, it may have low bias but high variance. The model can capture intricate relationships and account for the effect of sleep on exam scores. However, if the dataset is limited or noisy, the model may try to fit the noise or specific examples in the training set too closely, resulting in overfitting. It becomes overly sensitive to the variations and idiosyncrasies in the training data, making it less generalizable to new students' exam scores.





**Bias-Variance Tradeoff:** To strike a balance, we can use a multiple linear regression model that considers both study hours and sleep as features. This model can capture the relationships between multiple features and exam scores without being overly complex. It finds the linear combination of the features that best fits the data while still allowing for some flexibility. By including both study hours and sleep as features, the model can account for the influences of both factors and make more accurate predictions.



The multiple linear regression model strikes a balance between oversimplification (high bias) and overfitting (high variance). It captures the relationships between multiple features and the target variable, accounting for both the main effects of the features and any interactions between them.



By understanding the bias-variance tradeoff in the context of multiple feature variables, students can appreciate the need to find an optimal balance in model complexity, considering the relationships among multiple features while avoiding oversimplification or overfitting.

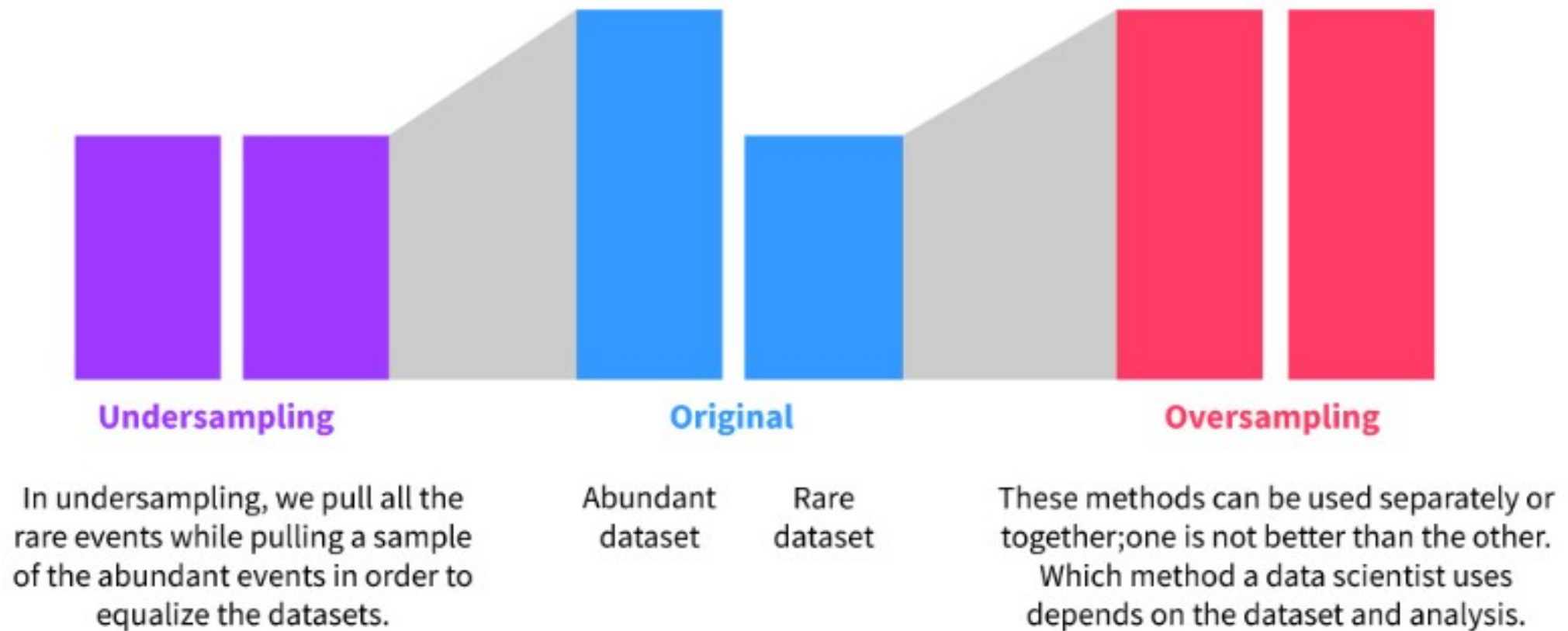
# BIAS-VARIANCE TRADE-OFF

- Below are two examples of configuring the bias-variance trade-off for specific algorithms:
  - The k-nearest neighbors algorithm has low bias and high variance, but the trade-off can be changed by increasing the value of  $k$  which increases the number of neighbors that contribute to the prediction and in turn increases the bias of the model.
  - The support vector machine algorithm has low bias and high variance, but the trade-off can be changed by increasing the  $C$  parameter that influences the number of violations of the margin allowed in the training data which increases the bias but decreases the variance.

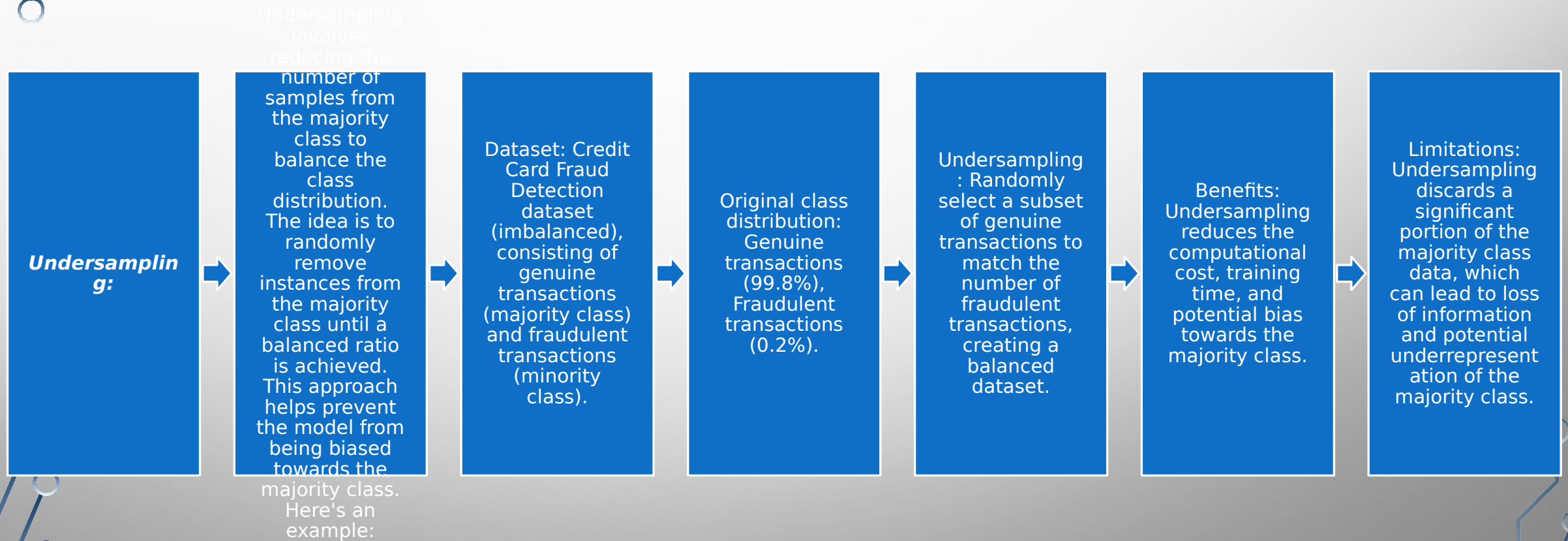
# BIAS-VARIANCE TRADE-OFF

- There is no escaping the relationship between bias and variance in machine learning.
  - Increasing the bias will decrease the variance.
  - Increasing the variance will decrease the bias

# UNDERSAMPLING VS. OVERSAMPLING FOR IMBALANCED DATASETS



# THE CREDIT CARD FRAUD DETECTION DATASET.



## **Oversampling :**

Oversampling involves increasing the number of samples in the minority class to balance the class distribution. This technique replicates or synthesizes new instances of the minority class to match the majority class. The goal is to provide more training examples for the minority class, enabling the model to learn better representation. Here's an example:

Dataset: Credit Card Fraud Detection dataset (imbalanced), consisting of genuine transactions (majority class) and fraudulent transactions (minority class).

Original class distribution: Genuine transactions (99.8%), Fraudulent transactions (0.2%).

Oversampling: Replicate or synthesize new instances of the fraudulent transactions to match the number of genuine transactions, creating a balanced dataset.

Benefits: Oversampling provides more data for the minority class, enabling better learning and capturing of its characteristics.

Limitations: Oversampling may lead to overfitting, as it introduces duplicated or synthetic samples that can amplify the minority class patterns, including noise and outliers



# TECHNIQUES TO CONTROL

- **Class Weighting:**

- Adjust the class weights during model training to give more importance to the minority class. This way, the model will penalize misclassifications of the minority class more heavily, promoting better classification performance on the underrepresented class.

- **Ensemble Methods:**

- Utilize ensemble methods like Random Forests or Gradient Boosting, which naturally handle class imbalances to some extent. These methods can assign more importance to the minority class and mitigate the impact of class imbalance.

- **Anomaly Detection:**

- Treat the problem as an anomaly detection task rather than a binary classification problem. Consider using techniques like One-Class SVM or Isolation Forests that are specifically designed for identifying anomalous instances.

- **Feature Engineering:**

- Analyze the dataset and identify relevant features that may contribute more significantly to fraud detection. Feature engineering can help in improving the discrimination power between genuine and fraudulent transactions.



1. **Mean Absolute Error (MAE):** When we subtract the predicted values from the actual values, obtaining the errors, sum the absolute values of those errors and get their mean. This metric gives a notion of the overall error for each prediction of the model, the smaller (closer to 0) the better.

$$mae = \left(\frac{1}{n}\right) \sum_{i=1}^n |Actual - Predicted|$$

**Note:** You may also encounter the  $y$  and  $\hat{y}$  notation in the equations. The  $y$  refers to the actual values and the  $\hat{y}$  to the predicted values.

2. **Mean Squared Error (MSE):** It is similar to the MAE metric, but it squares the absolute values of the errors. Also, as with MAE, the smaller, or closer to 0, the better. The MSE value is squared so as to make large errors even larger. One thing to pay close attention to, it that it is usually a hard metric to interpret due to the size of its values and of the fact that they aren't in the same scale of the data.

$$mse = \sum_{i=1}^D (Actual - Predicted)^2$$

3. **Root Mean Squared Error (RMSE):** Tries to solve the interpretation problem raised with the MSE by getting the square root of its final value, so as to scale it back to the same units of the data. It is easier to interpret and good when we need to display or show the actual value of the data with the error. It shows how much the data may vary, so, if we have an RMSE of 4.35, our model can make an error either because it added 4.35 to the actual value, or needed 4.35 to get to the actual value. The closer to 0, the better as well.

$$rmse = \sqrt{\sum_{i=1}^D (Actual - Predicted)^2}$$