

# Assembling methods and regularization

Zeham Management Technologies BootCamp  
by SDAIA

August 5th, 2024



**SDAIA**  
الهيئة السعودية للبيانات  
والذكاء الاصطناعي  
Saudi Data & AI Authority

# Ensemble

Let's start together...



## Agenda

Intro to Ensemble

Bagging

Boosting

Stacking

L1 and L2 Regularization

Enhance Model Generalization and Robustness

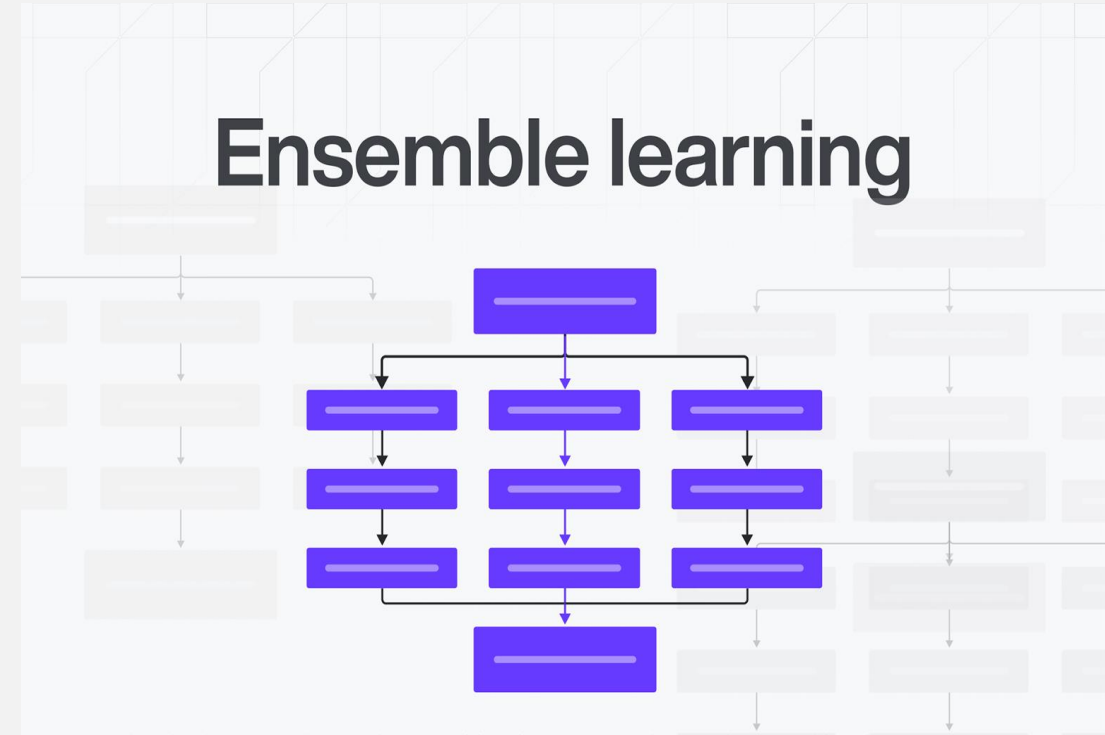
Oversampling and Undersampling



# Introduction to Ensemble

# ▶ What is Ensemble Learning?

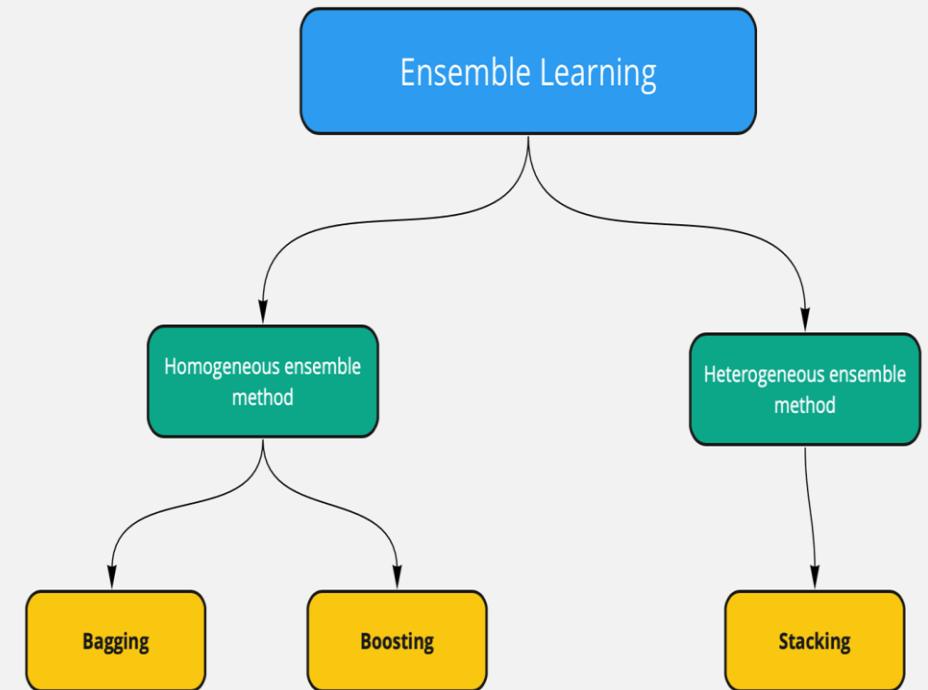
Ensemble Learning is an approach that enhances prediction accuracy by combining the strengths of multiple models. This method creates a more robust framework compared to individual models, as it helps to minimize prediction errors by reducing variance.



# ▶ What is Ensemble Learning?

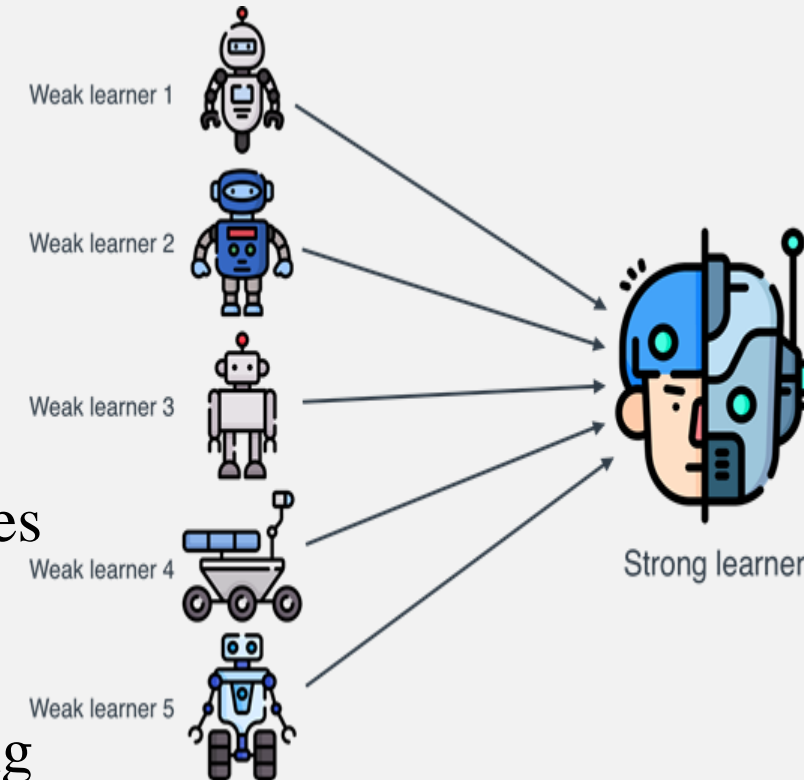
The key to successful Ensemble Learning lies in leveraging the diverse strengths of its contributing models. An ensemble is more effective when its constituent models offer varying performances on the same dataset, as this statistical diversity allows the ensemble to capture complementary information.

.



# Reason for Using Ensemble Learning

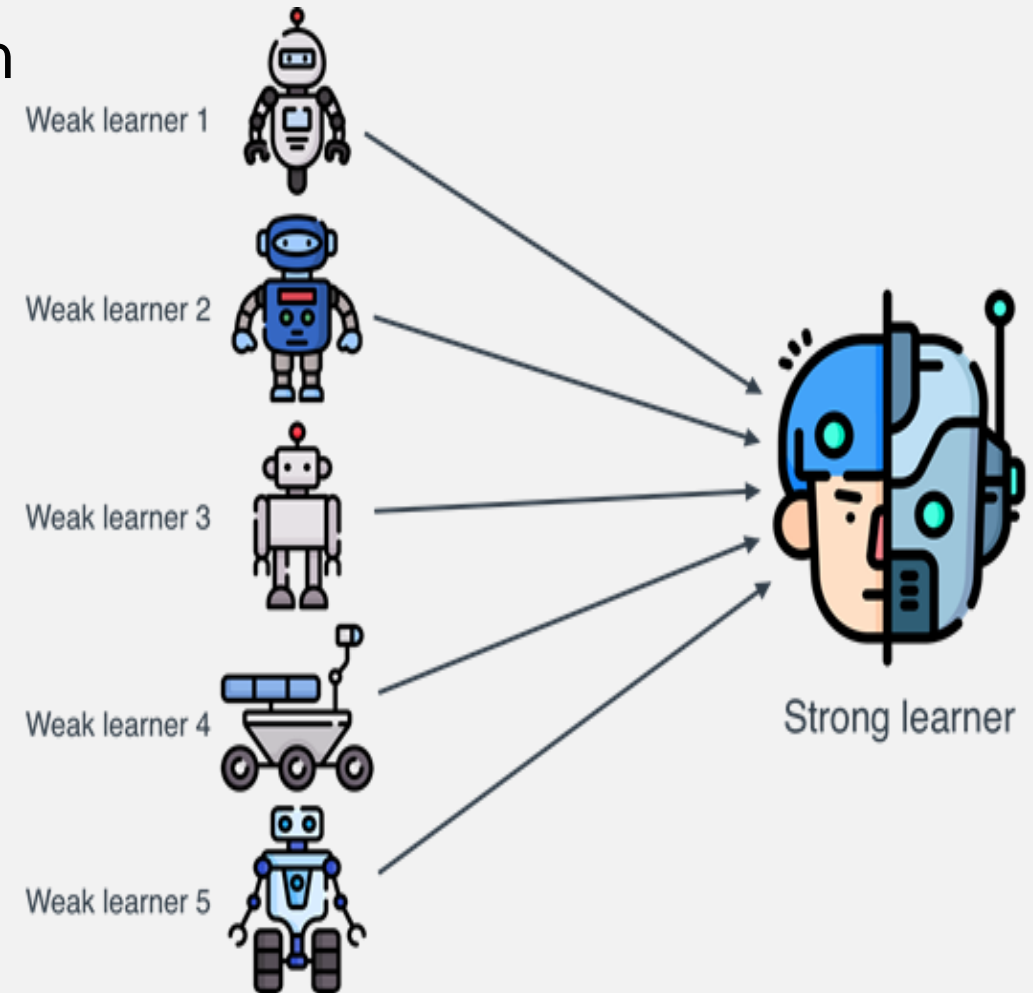
- The primary motivation for employing an ensemble learning model is to enhance classification performance through information fusion. By utilizing models trained on different data distributions but pertaining to the same set of classes, we can achieve more reliable and robust predictions.
- For instance, consider a scenario where we have one classifier trained to differentiate between handwritten digits using samples from high-resolution scans. Another classifier is trained using lower-resolution images from mobile phone captures. By combining the predictions from both classifiers when evaluating a new sample, we can achieve a more robust and unbiased decision.



# ▶ Ensemble methods

We will describe the most popular methods in Ensemble learning:

- Bagging
- Boosting
- Stacking



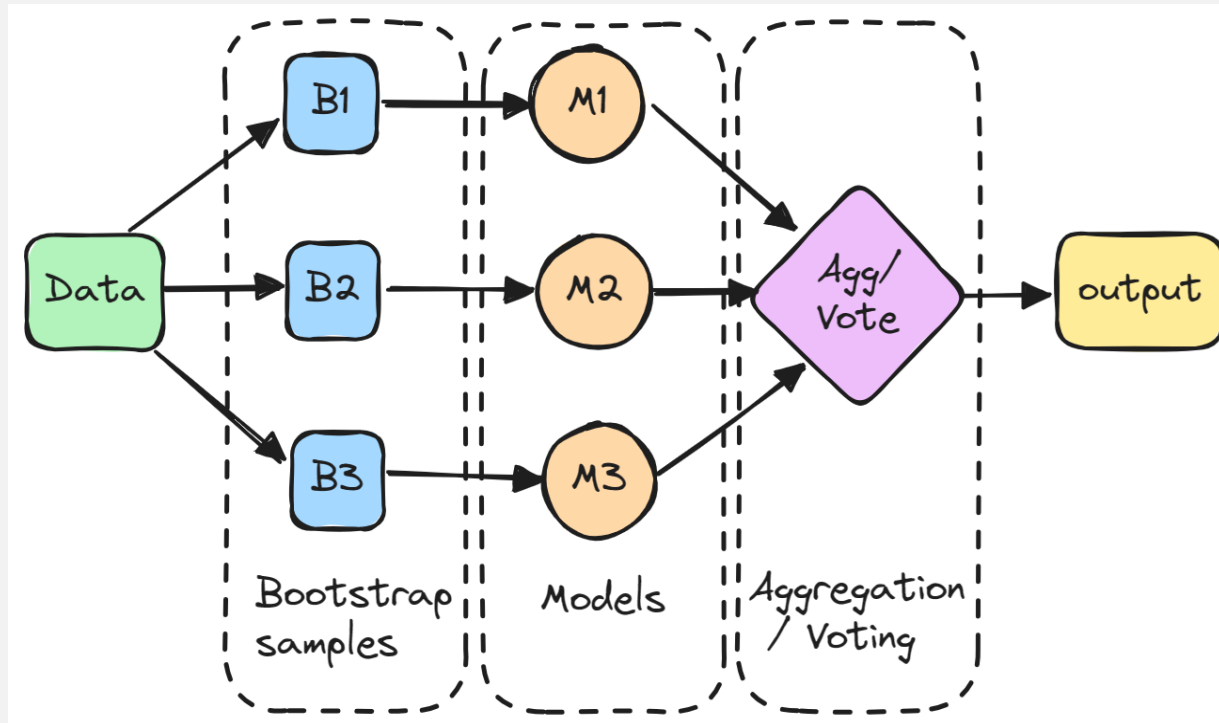


# Bagging



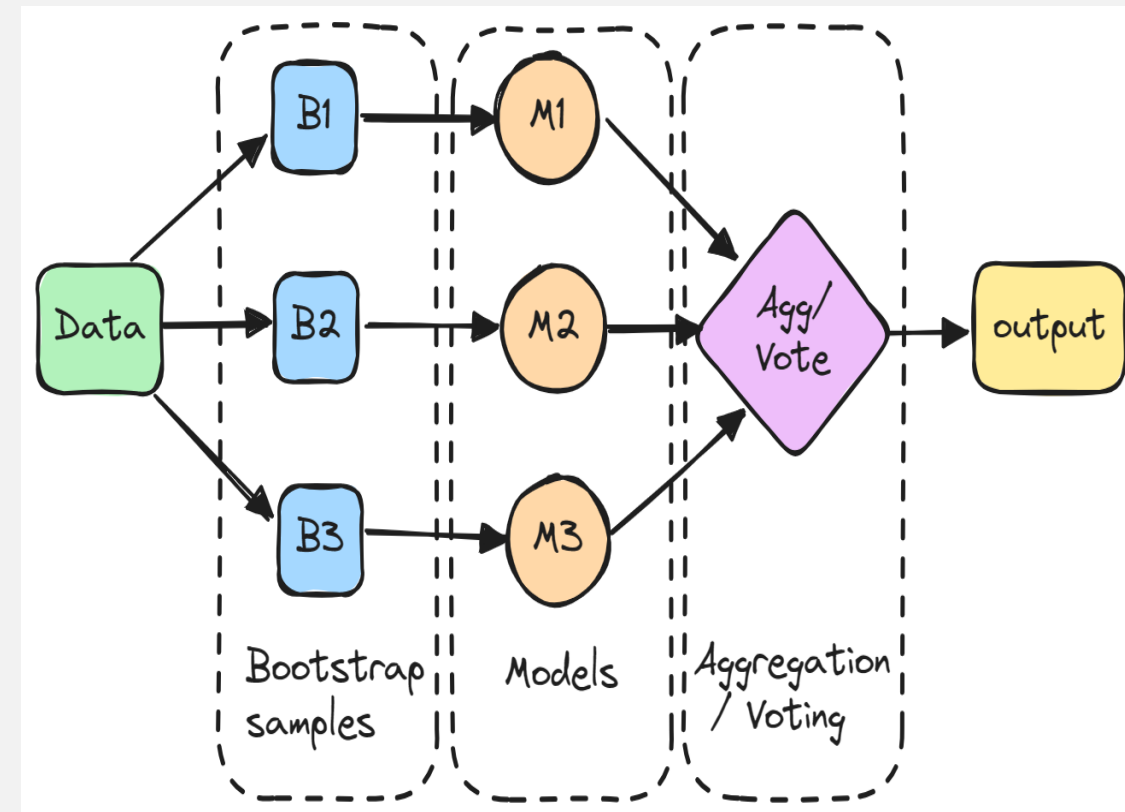
# What is Bagging ?

Bagging, also known as bootstrap aggregating, is an ensemble method. It involves training multiple models independently on random subsets of the data and aggregating their predictions through voting or averaging.



# ▶ How it work ?

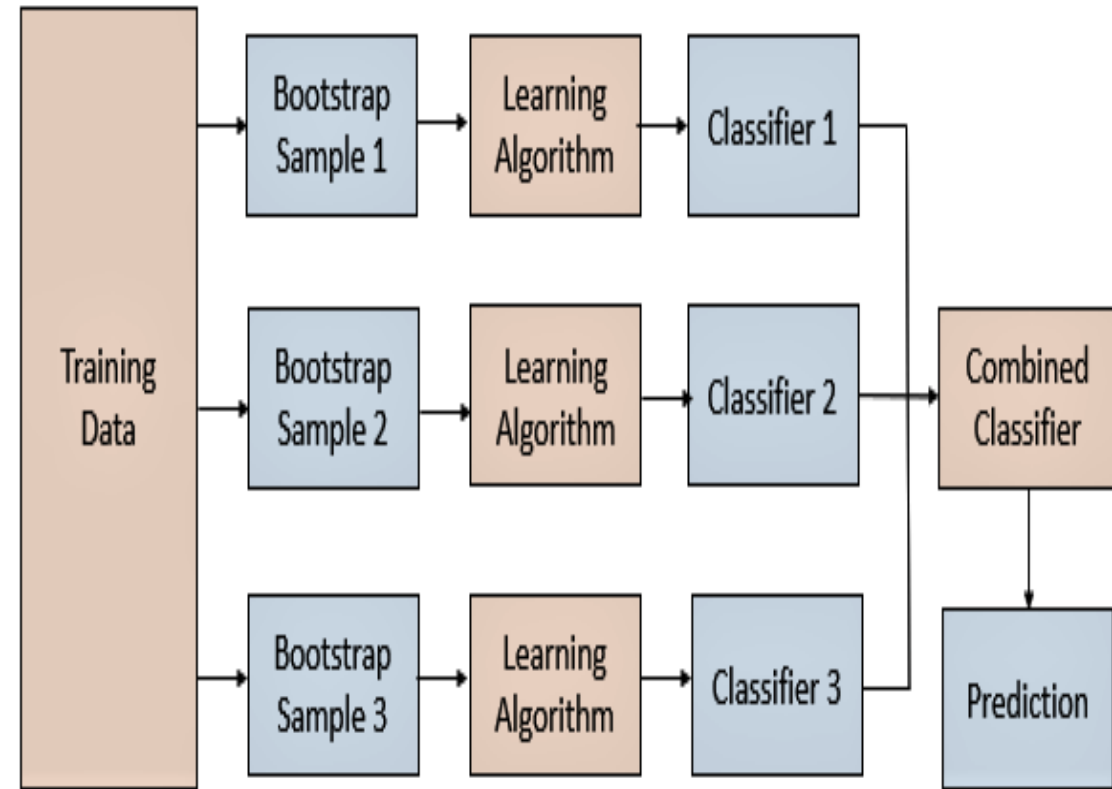
Each model is trained on a random subset of the data, which is sampled with replacement. This means that individual data points can be chosen more than once, and the random subset is called a bootstrap sample. By training models on different bootstrap samples, bagging reduces the variance of the individual models and avoids overfitting by exposing the models to different parts of the dataset.





## How it work ?

Each model is trained on a random subset of the data, which is sampled with replacement. This means that individual data points can be chosen more than once, and the random subset is called a bootstrap sample. By training models on different bootstrap samples, bagging reduces the variance of the individual models and avoids overfitting by exposing the models to different parts of the dataset.





# Advantages of Bagging

- **Reduces Overfitting:** Bagging can lower the chance of an overfit model, resulting in improved model accuracy on unseen data.
- **Decreases Model Variance:** Multiple models trained on different subsets of data average out their predictions, leading to lower variance than a single model.
- **Improves Stability:** Changes in the training dataset have less impact on bagged models, making the overall model more stable.



# Advantages of Bagging

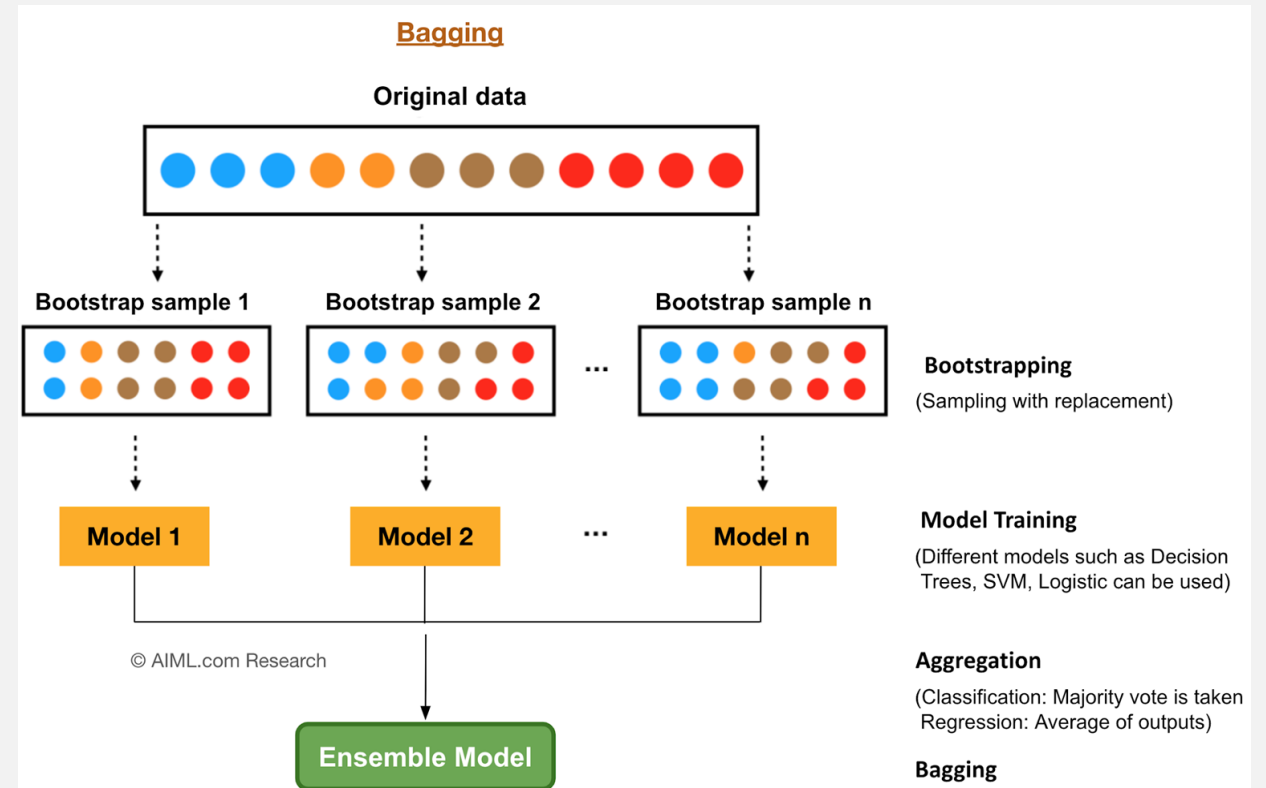
- **Handles High Variability:** Especially effective for algorithms like decision trees, which tend to have high variance.
- **Parallelizable Computation:** Each model in the ensemble can be trained independently, allowing for parallel processing and efficient use of computational resources.
- **Easy to Understand and Implement:** The concept behind bagging is straightforward and can be implemented without complex modifications to the learning algorithm.



# Advantages of Bagging

**Good with Noisy Data:** The averaging process helps in reducing the noise in the final prediction.

**Handles Imbalanced Data:** Bagging can help in scenarios where the dataset is imbalanced, improving the performance of the model in such situations.





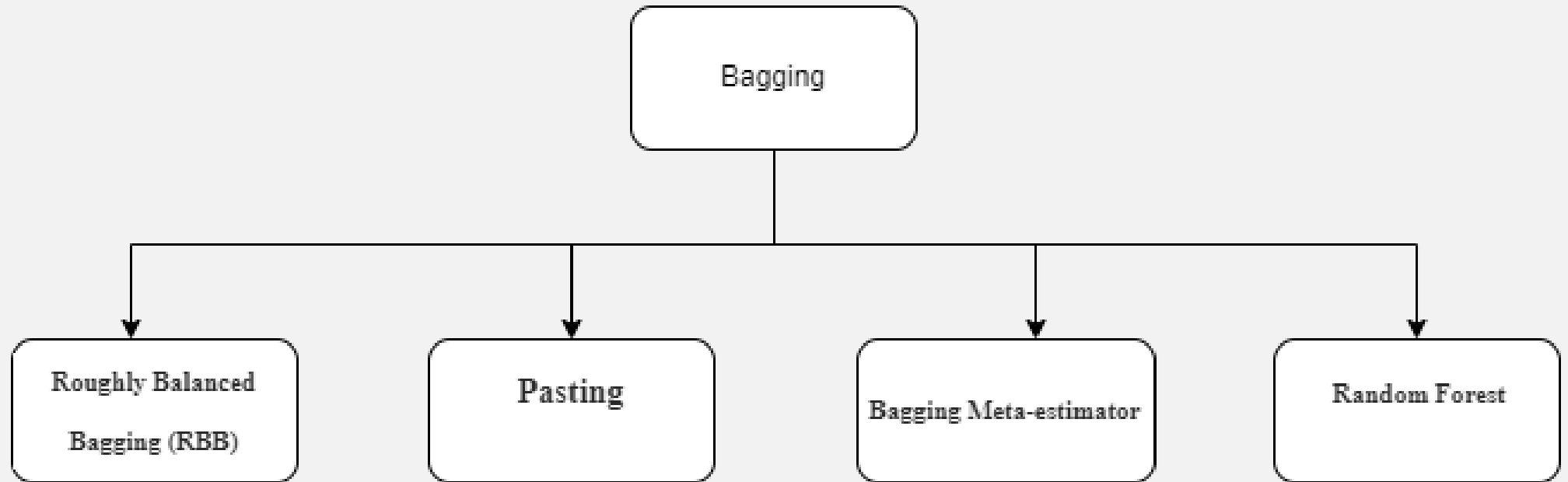
## Bagging Disadvantage

- One drawback of bagging is that it reduces the interpretability of a model. If the proper procedures are ignored, the resulting model can experience a lot of bias. While bagging is highly accurate, it can be computationally expensive, which may discourage its use in certain instances.





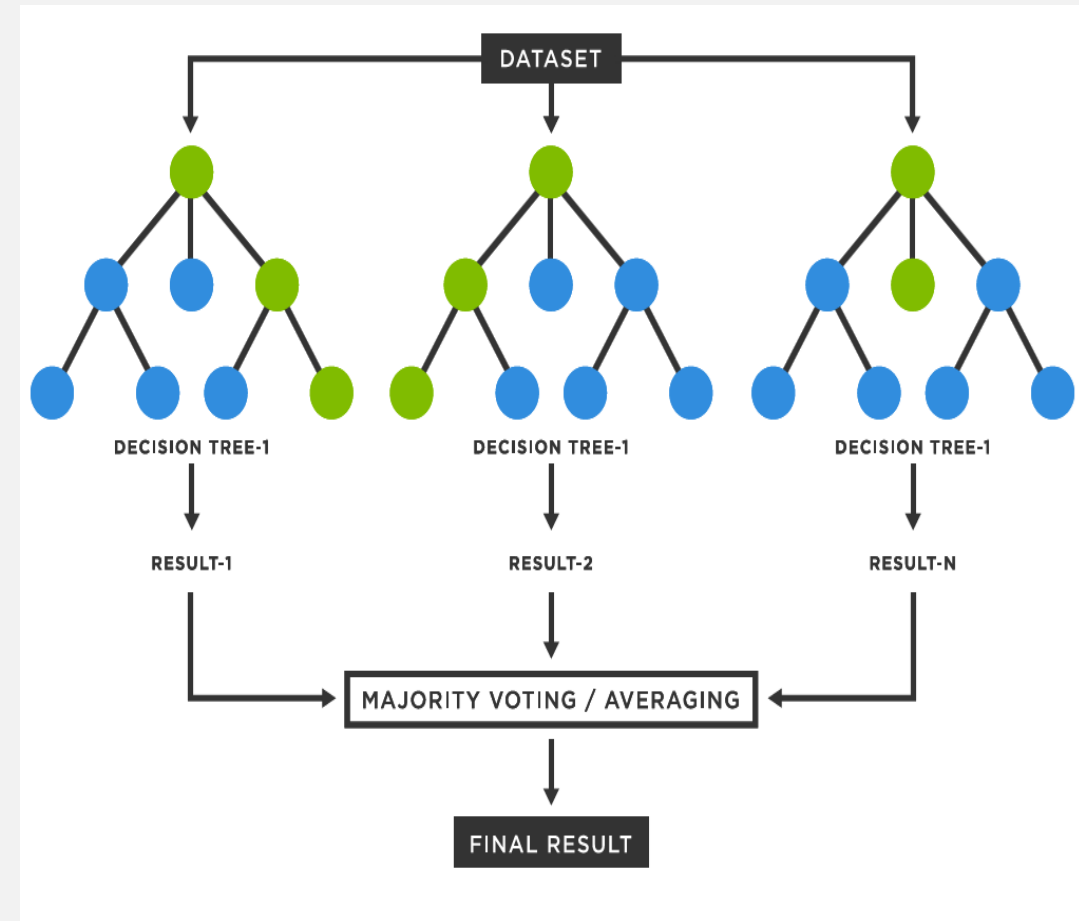
# Bagging Algorithms





# Random Forest

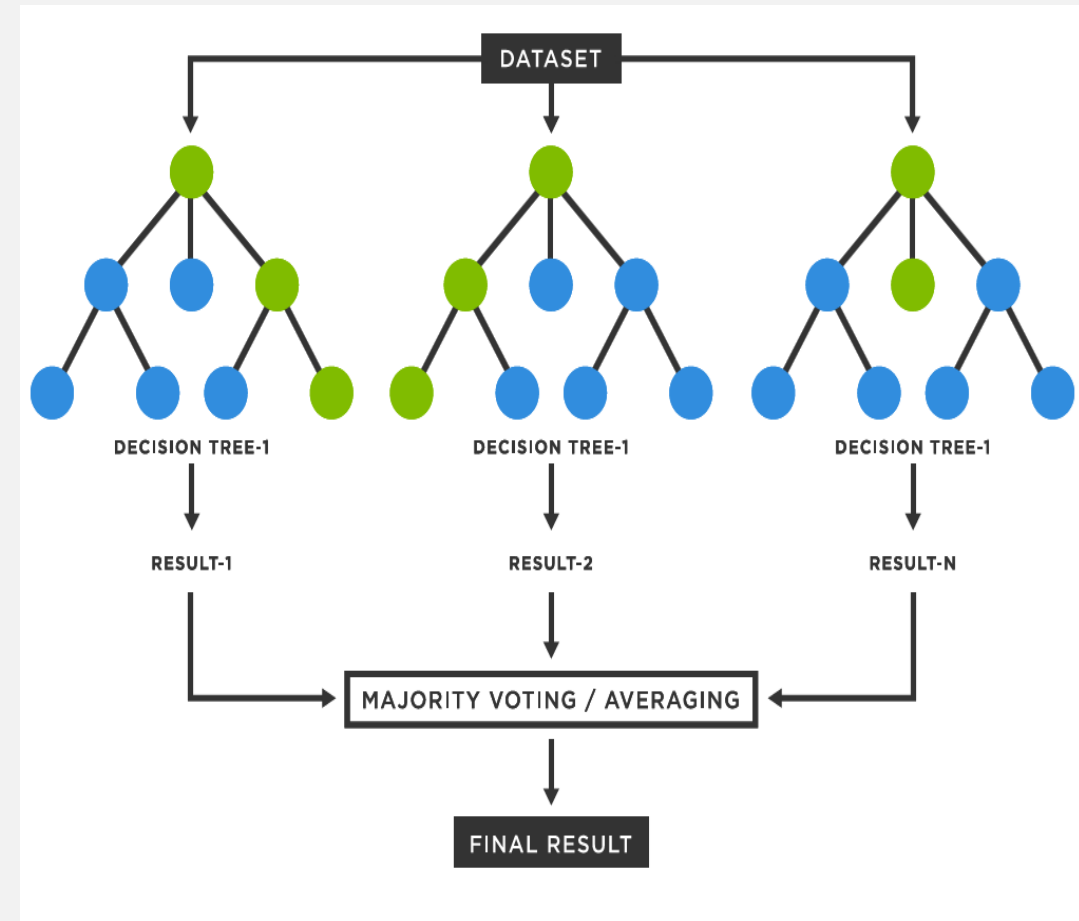
- The Random Forest algorithm is a popular bagging ensemble method known for its simplicity and strong predictive capabilities. It builds multiple decision trees during training and combines their predictions for unseen data.
- Each tree is trained on a random sample of the data, and when splitting nodes, only a random subset of features is considered, adding an extra layer of randomness to the ensemble.
- The final prediction is obtained by averaging the individual tree predictions for regression problems or using majority voting for classification problems.





# Random Forest: Strengths and Limitations

- Strengths:
  - - High Accuracy: Random forests often deliver highly accurate predictions with minimal tuning of parameters.
  - - Feature Importance: They can provide insights into the importance of different features in prediction.
- Limitations:
  - - Computational Complexity: Random forests can be computationally intensive and slow with large datasets.
  - - Less Interpretability: While an individual tree is interpretable, a forest may not be as interpretable.





# Let's Practice

## **Notebook :**

3- Advanced Machine Learning/2- assembling methods and regularization  
/LAB/Random Forest Classifier Tutorial.IPYNB

## **Dataset:**

3- Advanced Machine Learning/2- assembling methods and regularization  
/LAB/Datasets/car\_evaluation.csv



# Bagging Meta-estimator

- Bagging Meta-estimator is an ensemble method that trains base classifiers on random subsets of the dataset and aggregates their predictions. This reduces variance by introducing randomization and creating an ensemble.
- Multiple random sub-samples of the dataset are created (with replacement), and a classifier or regressor is trained on each. For unseen data, each model predicts, and the final output is determined by majority voting for classification or averaging for regression.



# Bagging Meta-estimator: Strengths and Limitations

## Strengths:

1. Reduction in Variance: By using multiple models, variance is significantly reduced, resulting in more stable predictions.
2. Robustness to Overfitting: Bagging helps reduce overfitting, especially when using models like decision trees.
3. Parallel Computations: Training models can be done in parallel, improving computational efficiency.

## Limitations:

1. Model Simplicity: The final ensemble model, while accurate, may lack the interpretability of simpler models.
2. Computationally Intensive: Depending on the base estimator and dataset size, bagging can be computationally demanding.



# Pasting

- Pasting is an ensemble method that is similar to Bagging, but with a different sampling technique. While Bagging uses bootstrapping (sampling with replacement), Pasting samples without replacement, giving a unique twist to ensemble learning.
- In Pasting, multiple training sets are created by sampling uniformly and without replacement. Predictive models are then trained on each of these sets. The final prediction is made by either taking the majority vote for classification tasks or averaging for regression, using the predictions of all the individual models.



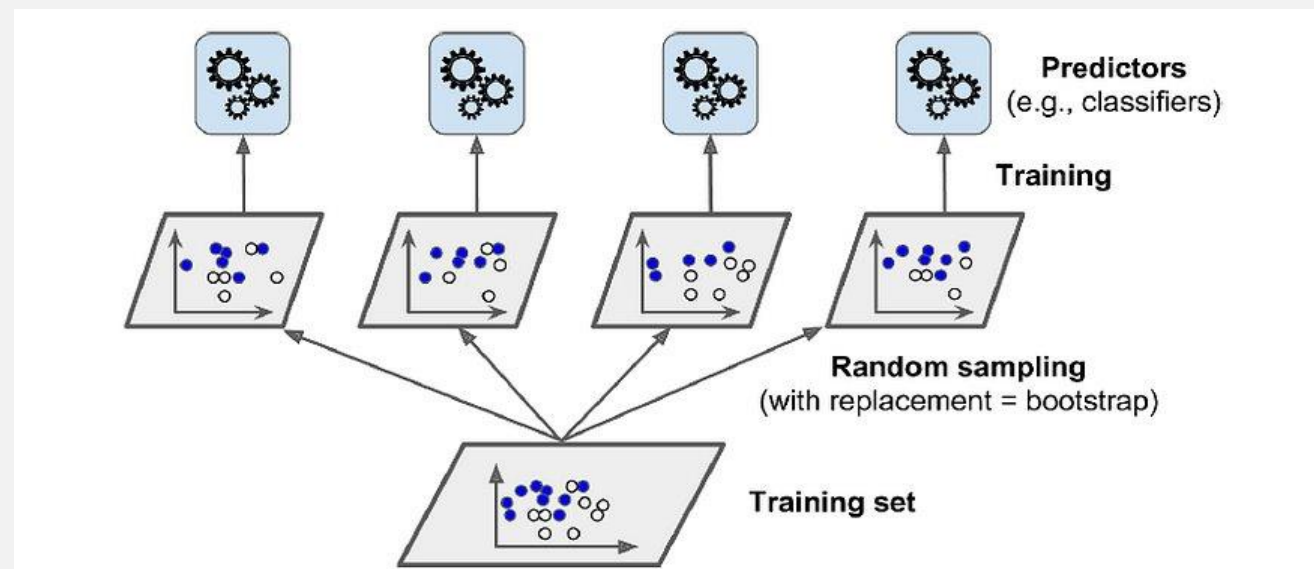
# Pasting: Strength and Limitations

- Strengths:
  1. Variance Reduction: Pasting, like bagging, reduces variance without increasing bias
  2. Parallelization: Models can be trained in parallel, improving computational efficiency
- Limitations:
  1. Model Complexity: The ensemble model may not be as interpretable as simpler models
  2. Memory Usage: Storing numerous models can become computationally expensive with large datasets



# Roughly Balanced Bagging (RBB)

- Roughly Balanced Bagging (RBB) is a technique designed to address imbalanced data. It aims to create balanced or nearly balanced class distributions in each bootstrap sample (bag) while maintaining diversity among the bags. In RBB, each bag is created by first sampling from the minority class and then from the majority class. The classifiers are trained on these bags, and the predictions are combined using majority voting.





# Roughly Balanced Bagging (RBB): Strength and Limitations

- Strengths:
  - Imbalanced Data Handling: Specially designed to handle imbalanced datasets.
  - Classifier Diversity: Maintains diversity among classifiers.
- Limitations:
  - Data Requirement: May necessitate substantial data for each class.
  - Parameter Tuning: The balance degree in each bag may require tuning.



# Let's Practice

## **Notebook :**

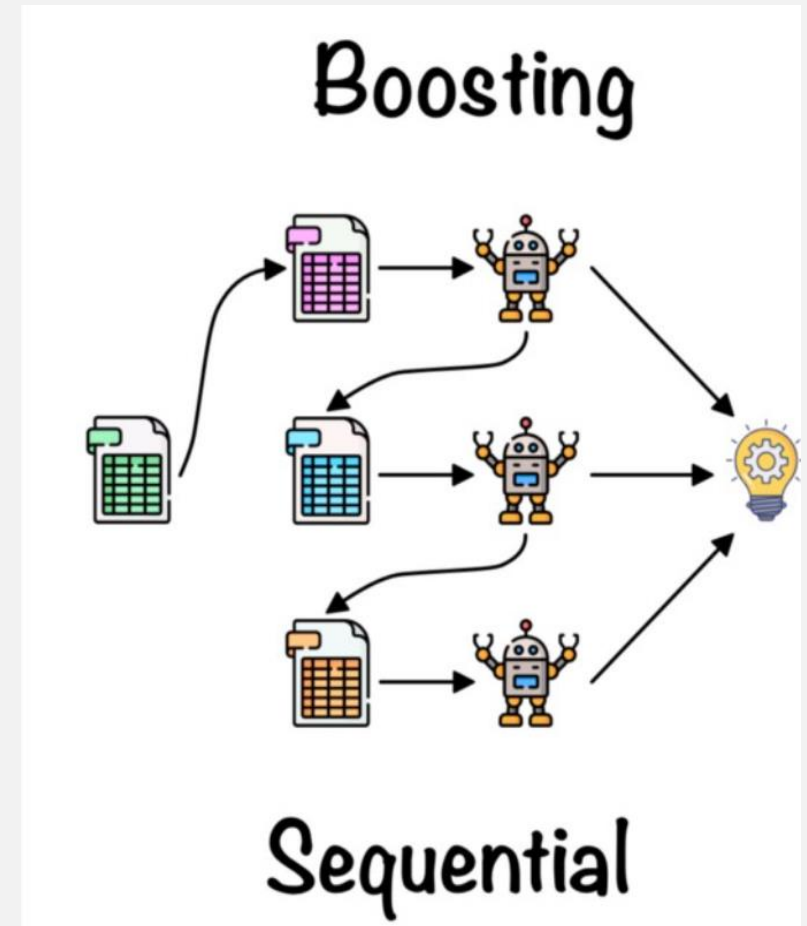
3- Advanced Machine Learning/2- assembling methods and regularization/LAB/Bagging\_Tutorial.ipynb

# Boosting



# What is Boosting?

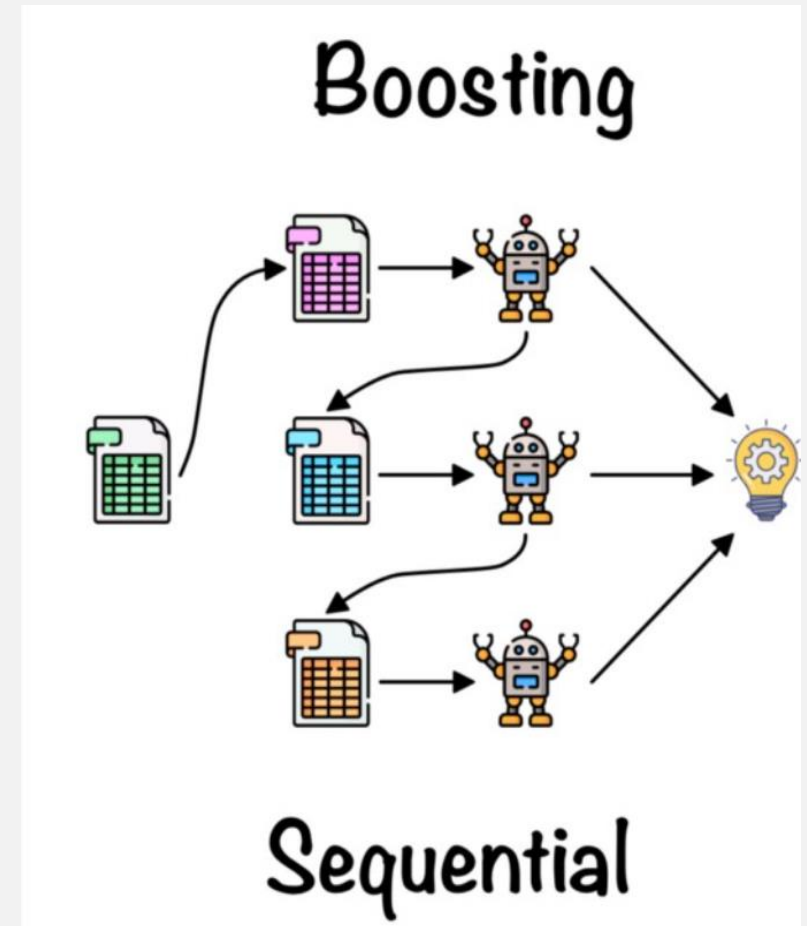
- Boosting is an ensemble learning method that combines a set of weak learners to create a strong learner, minimizing training errors. Boosting algorithms can enhance the predictive power of your data mining endeavors.





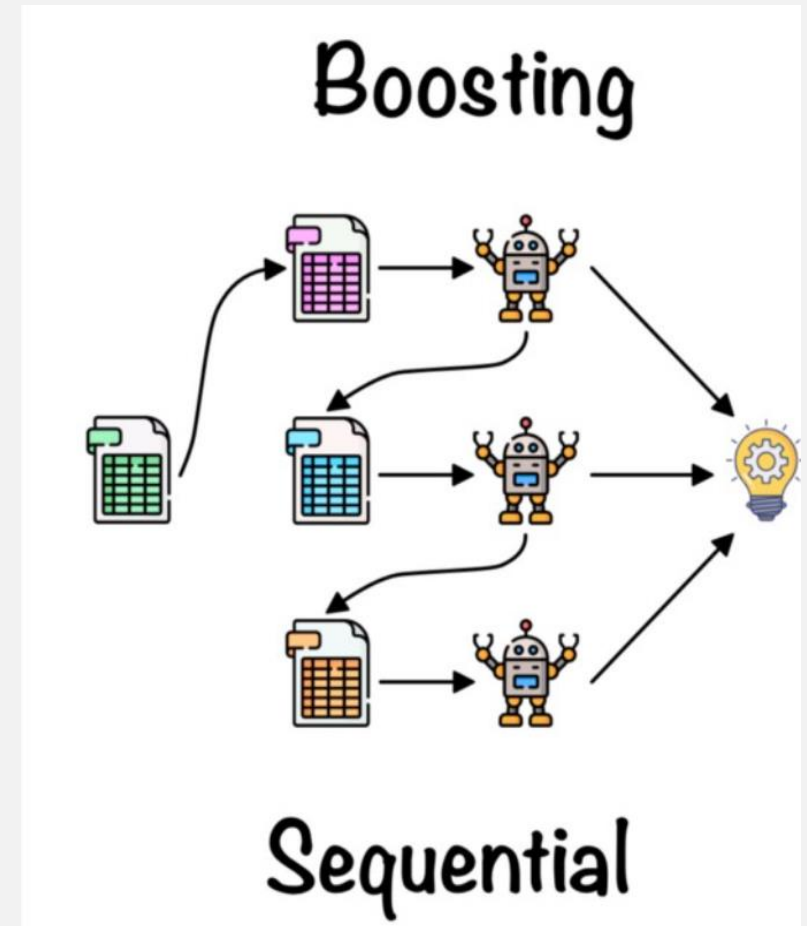
# How it works?

- In boosting, a random sample of data is selected, fitted with a model, and then trained sequentially. Each model tries to compensate for the weaknesses of its predecessor. With each iteration, the weak rules from each individual classifier are combined to form one strong prediction rule.



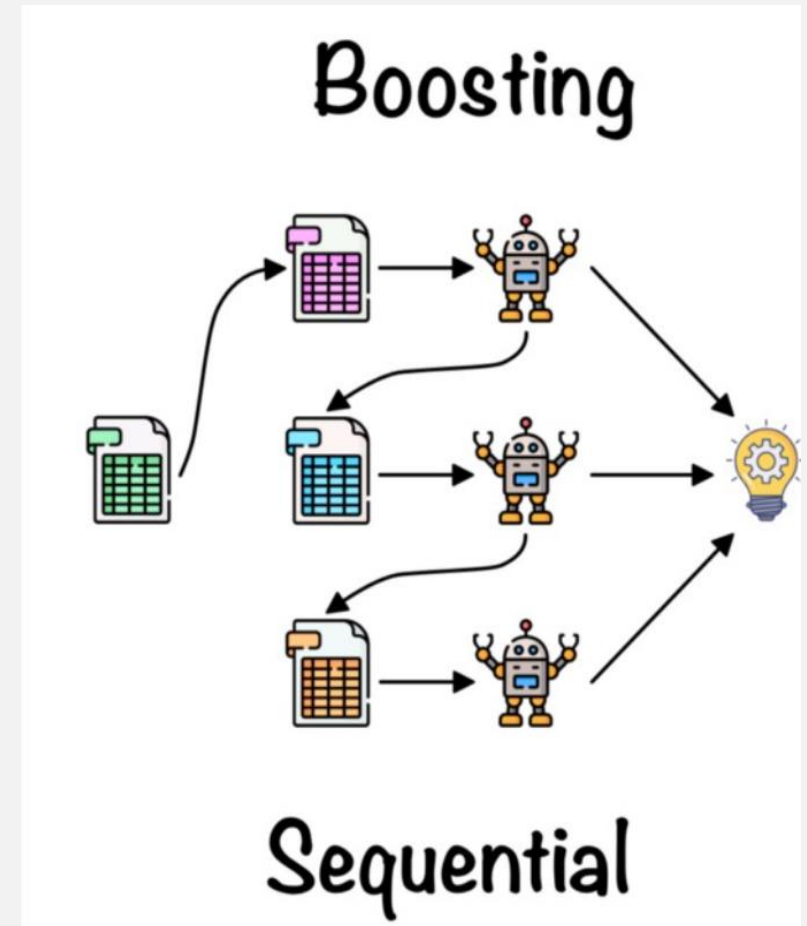
# Advantages

- **Ease of Implementation:** Boosting can be used with several hyper-parameter tuning options to improve fitting. No data preprocessing is required, and boosting algorithms like have built-in routines to handle missing data.
- **Reduction of bias:** Boosting algorithms combine multiple weak learners in a sequential method, iteratively improving upon observations. This approach can help to reduce high bias, commonly seen in shallow decision trees and logistic regression models.
- **Computational Efficiency:** Since boosting algorithms only select features that increase its predictive power during training, it can help to reduce dimensionality as well as increase computational efficiency.



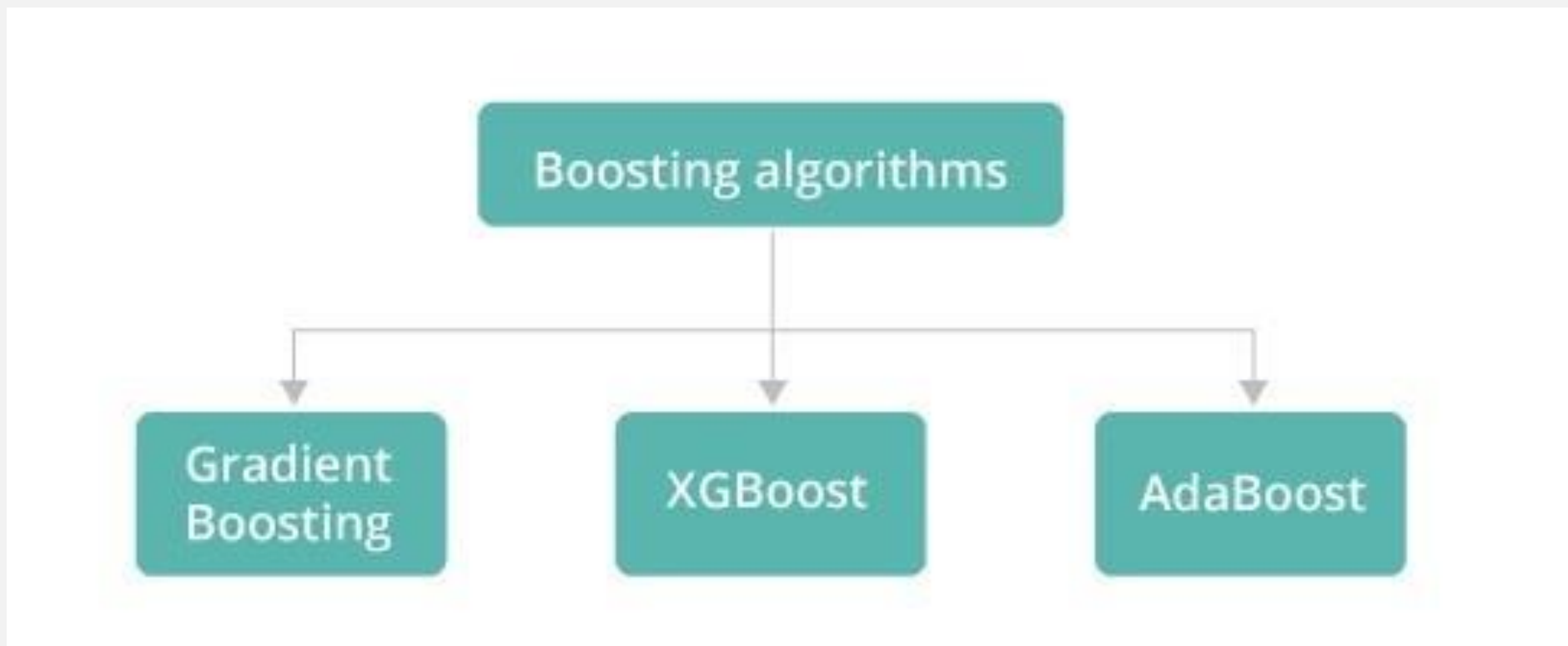
# Disadvantages

- **One disadvantage** of boosting is that it is sensitive to outliers since every classifier is obliged to fix the errors in the predecessors. Thus, the method is too dependent on outliers.
- **Another disadvantage** is that the method is almost impossible to scale up. This is because every estimator bases its correctness on the previous predictors, thus making the procedure difficult to streamline.
- **Overfitting:** There is some debate in the research community about whether boosting reduces or exacerbates overfitting. We categorize it as a challenge because, when overfitting occurs, the model's predictions fail to generalize to new datasets.
- **Intense Computation:** Boosting involves sequential training, which is difficult to scale up because each estimator builds on its predecessors. This makes boosting models computationally expensive. While XGBoost addresses some scalability issues, boosting algorithms can be slower to train compared to bagging due to the large number of parameters that influence model behavior.



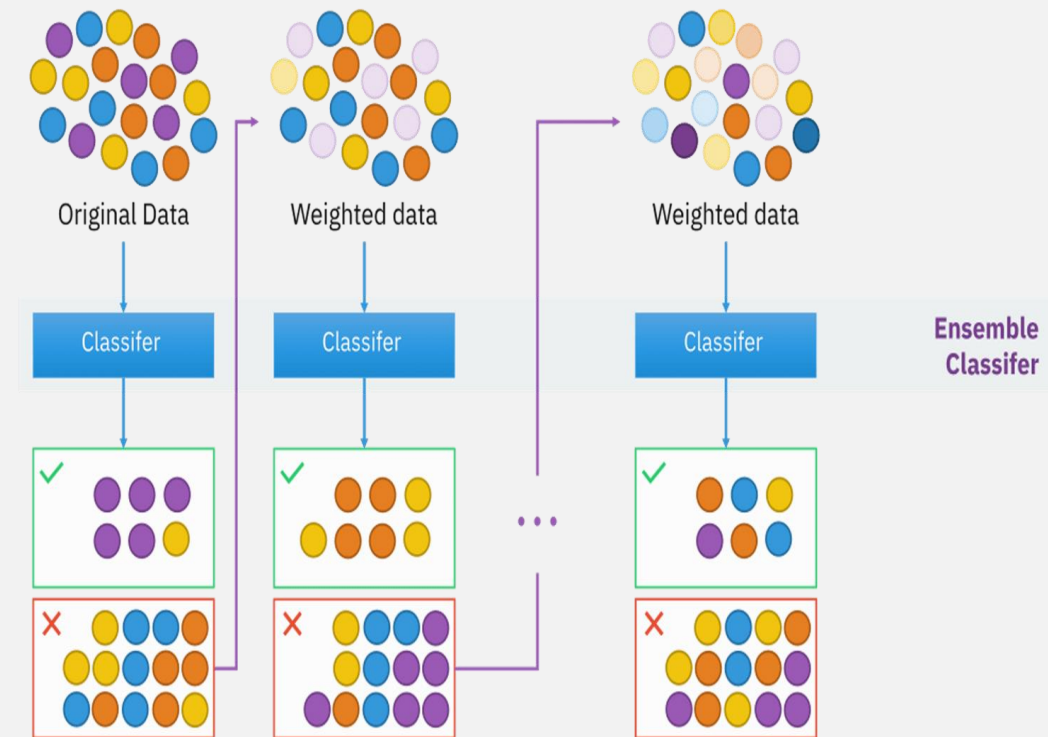


# ▶ Boosting Algorithms



# AdaBoost

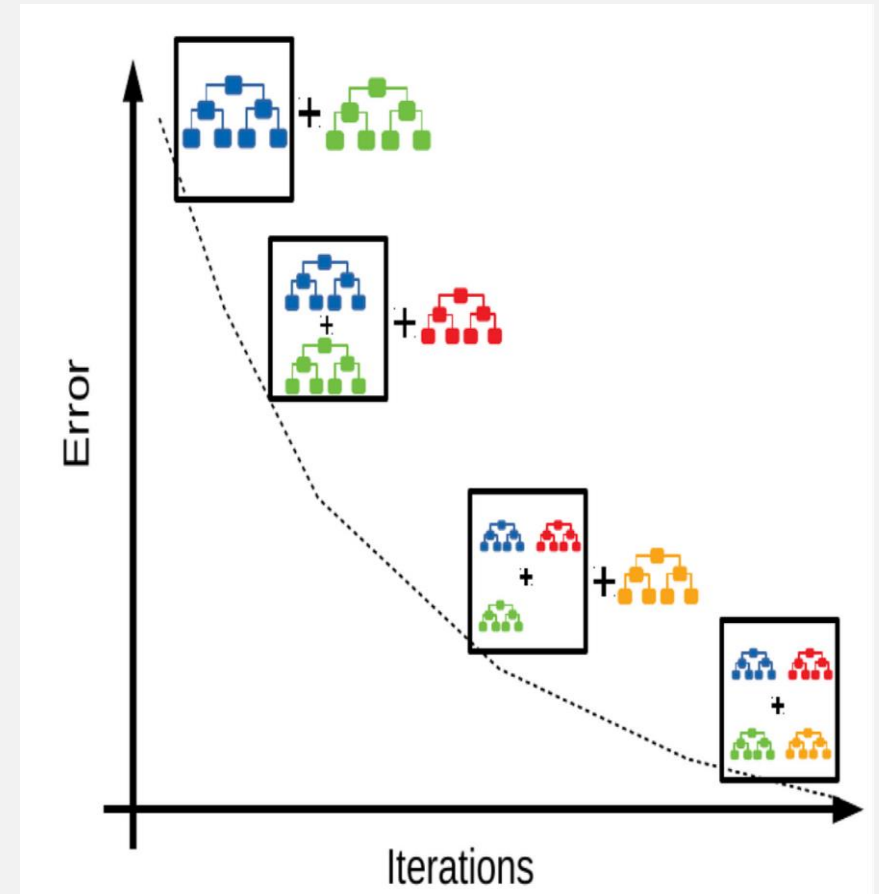
- Adaptive Boosting (AdaBoost), created by Yoav Freund and Robert Schapire, iteratively improves model accuracy by focusing on misclassified data points. In each iteration, it adjusts the weights of these instances, emphasizing them more in subsequent rounds. This sequential process continues until a robust final predictor is produced, effectively reducing bias and variance. AdaBoost's strength lies in its ability to sequentially correct errors, enhancing overall performance in both classification and regression tasks.





# Gradient Boosting

- **Gradient Boosting**, developed by Jerome H. Friedman based on Leo Breiman's work, enhances an ensemble by sequentially adding predictors, each one addressing the errors of its predecessor. Unlike AdaBoost, which adjusts the weights of data points, Gradient Boosting focuses on training new models on the residual errors from the previous model. The term "gradient boosting" stems from its use of the gradient descent algorithm in conjunction with the boosting technique.



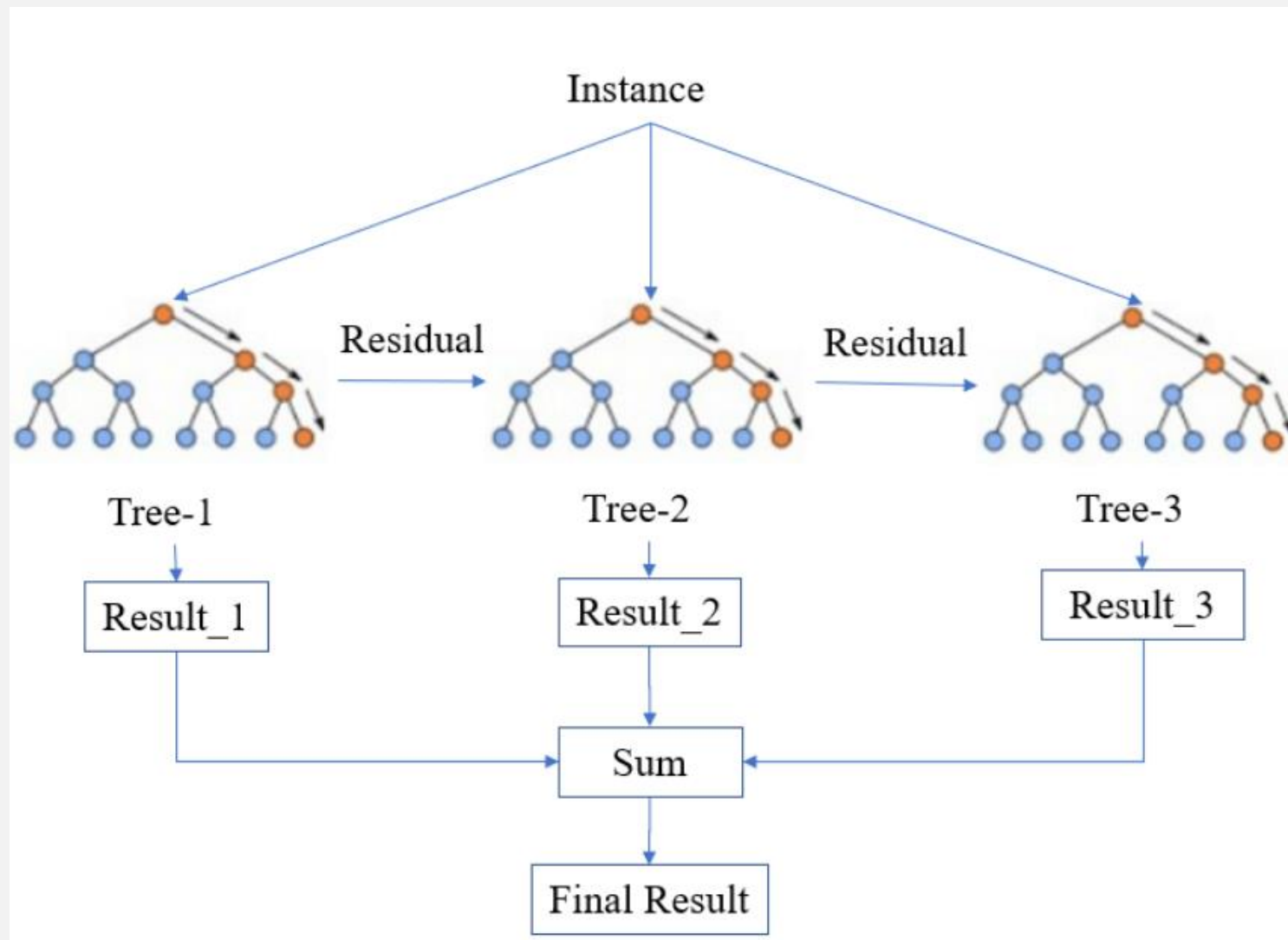


**XGBoost**, short for **Extreme Gradient Boosting**, is a scalable and distributed machine learning library designed for gradient-boosted decision trees (GBDT). It supports parallel tree boosting and is a top choice for tasks involving regression, classification, and ranking.

To fully understand XGBoost, it's essential to first comprehend the foundational machine learning concepts and algorithms it builds upon: **supervised machine learning**, decision trees, ensemble learning, and gradient boosting.

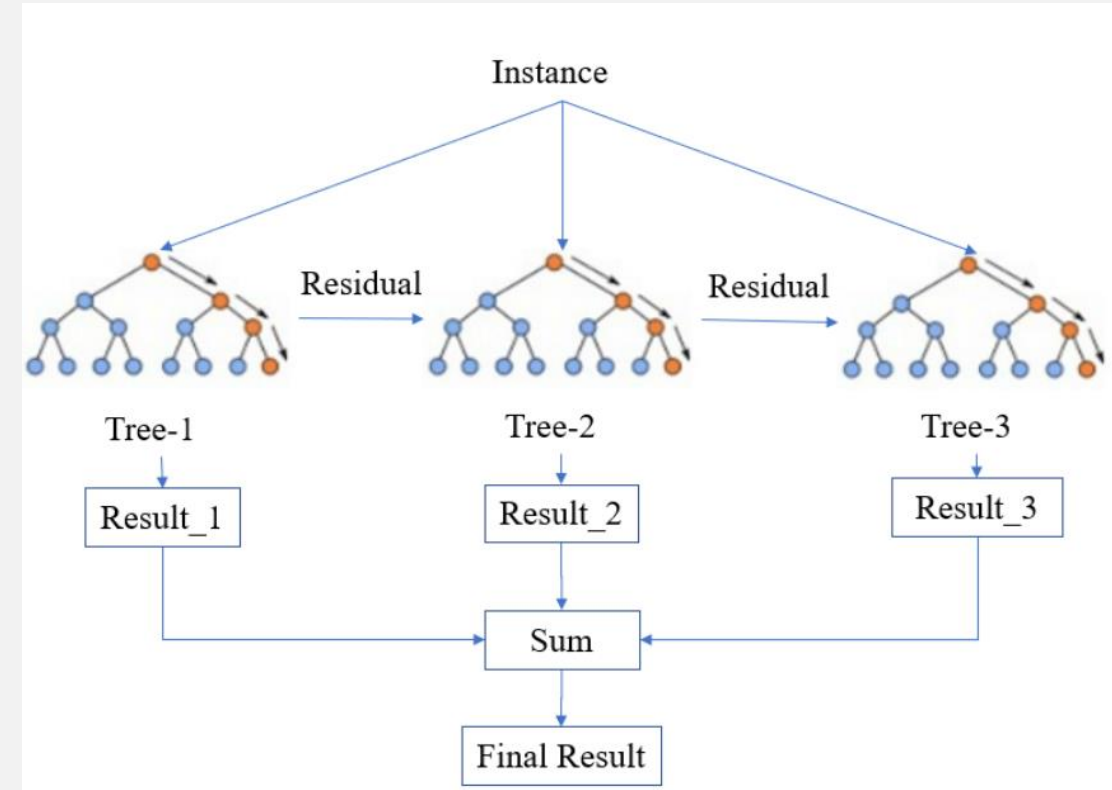
Supervised machine learning employs algorithms to train a model that identifies patterns within a dataset containing labeled features. The trained model is then utilized to predict labels for the features of a new dataset.

# XGBoost



# XGBoost

- **Extreme Gradient Boosting (XGBoost)** is an optimized gradient boosting implementation designed for speed and scalability.
- It leverages multiple CPU cores for parallel processing during training, enhancing computational efficiency.
- XGBoost advanced regularization techniques prevent overfitting, making it suitable for large datasets and complex models.
- Its ability to handle sparse data and deliver accurate results has made it a popular choice in competitive data science.





# Why XGBoost

XGBoost is favored for two main reasons: its execution speed and its model performance.

Execution speed is vital for handling large datasets efficiently. XGBoost imposes no limitations on dataset size, allowing you to work with datasets that might be unmanageable with other algorithms.

Model performance is equally important as it enables the creation of superior models. Comparisons between XGBoost and other algorithms, such as random forest (RF), gradient boosting machines (GBM), and gradient boosting decision trees (GBDT), consistently demonstrate that XGBoost excels in both execution speed and model performance.



Is XGBoost a classification or regression algorithm?

XGBoost is both a classification and regression algorithm. It's designed to handle problems where you can train a model using a set of data to create either a classifier for categorizing new data or a regressor for predicting continuous values.





# Let's Practice

## **Notebook :**

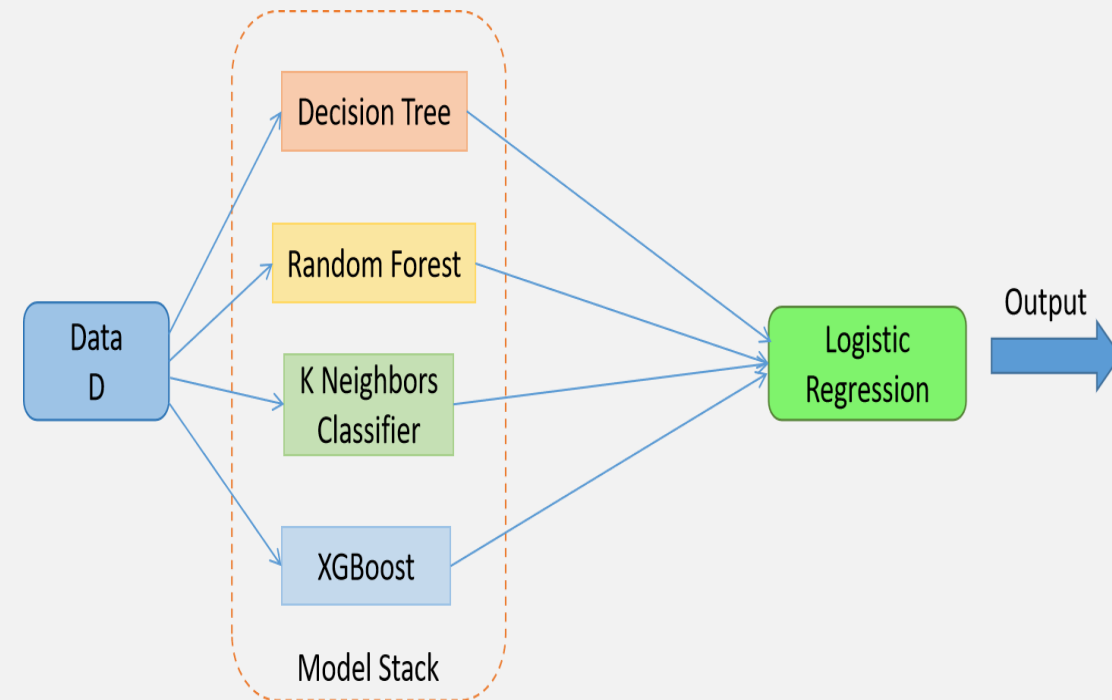
3- Advanced Machine Learning/2- assembling methods and regularization  
/LAB/Boosting\_Tutorial.ipynb

# Stacking



# What is stacking

- Stacking is a widely-used ensemble technique in machine learning, where multiple weak learners are combined in parallel. By integrating these learners with meta-learners, more accurate predictions can be achieved.
- Also referred to as stacked generalization, stacking is an advanced version of the Model Averaging Ensemble method. In this approach, all sub-models contribute based on their performance weights to form a new, more predictive model. The term "stacking" comes from the fact that this new model is layered on top of the existing ones.



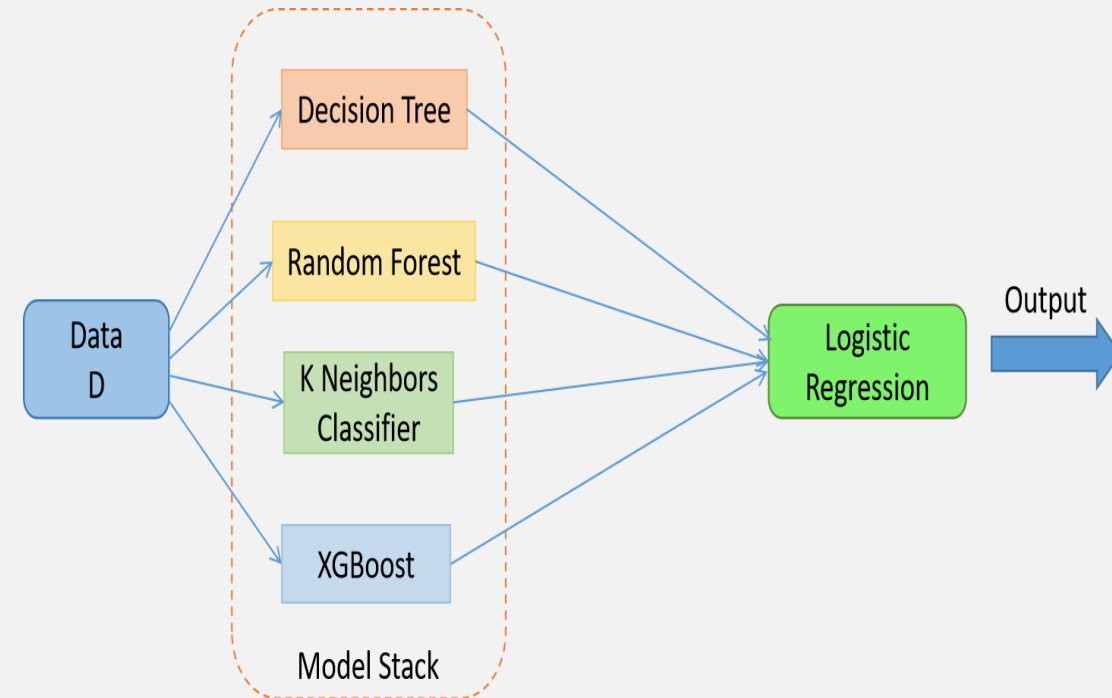
# How it works

- **Training Base Learners:**

- Multiple base learners (models) are trained on the same dataset. These can be different algorithms (e.g., decision trees, logistic regression, SVMs) or the same algorithm with different hyperparameters.
- Each base learner is trained independently on the training data.

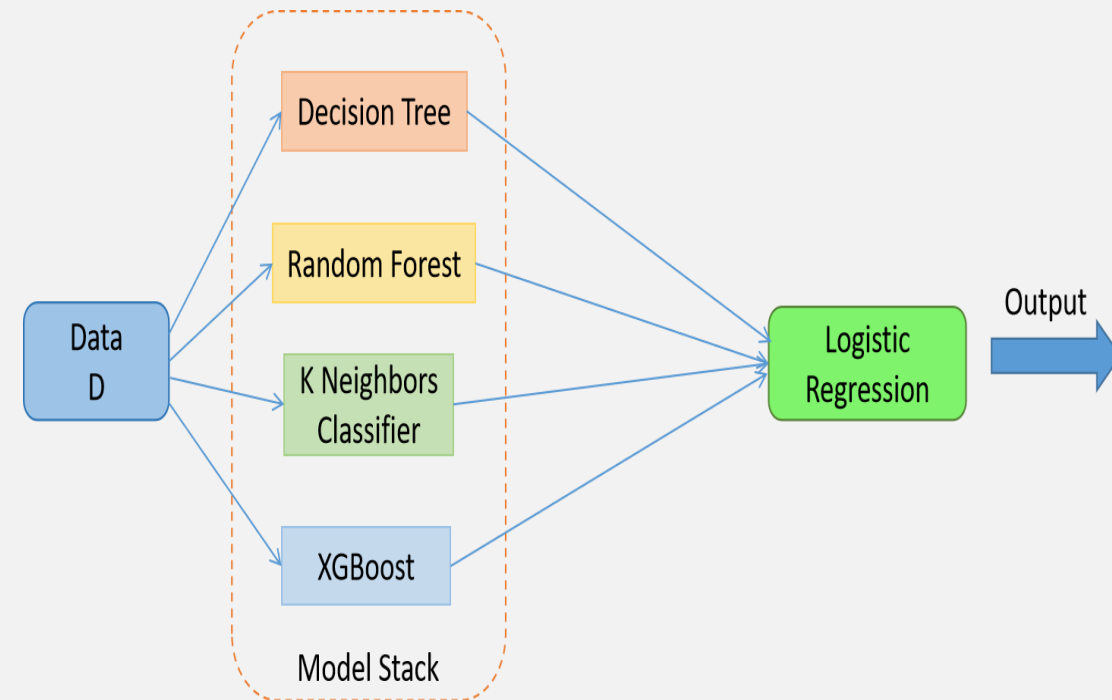
- **Generating Base Learner Predictions:**

- Once trained, each base learner makes predictions on a validation set or the training data using cross-validation.
- The predictions from all base learners form a new dataset, where each column corresponds to the predictions of a specific base learner.



# How it works

- **Training the Meta-Learner:**
  - A meta-learner (or meta-model) is trained using the predictions from the base learners as input features. The target variable for the meta-learner is the same as the original target variable.
  - The meta-learner learns to combine the predictions from the base learners in a way that improves overall performance.
- **Making Final Predictions:**
  - To make final predictions on new, unseen data, the base learners first generate their predictions.
  - These predictions are then fed into the meta-learner, which produces the final prediction.





# Benefits and Considerations of Stacking

## Benefits:

1. **Improved Accuracy:** Stacking combines multiple models, resulting in better predictive performance compared to using individual models.
1. **Flexibility:** Stacking allows for the combination of different types of models, creating a diverse set of base learners.
1. **Robustness:** Stacking reduces the risk of overfitting associated with individual models by averaging out their predictions."

## Considerations:

1. **Computational Complexity:** Training multiple models and a meta-learner can be computationally intensive.
2. **Overfitting:** There's a risk of overfitting if the meta-learner is too complex or if there isn't enough data to train both base learners and the meta-learner.



# Let's Practice

## **Notebook :**

3- Advanced Machine Learning/2- assembling methods and regularization  
/LAB/Stacking\_Tutorial.ipynb

# Regularization Techniques



**SDAIA**  
الهيئة السعودية للبيانات  
والذكاء الاصطناعي  
Saudi Data & AI Authority

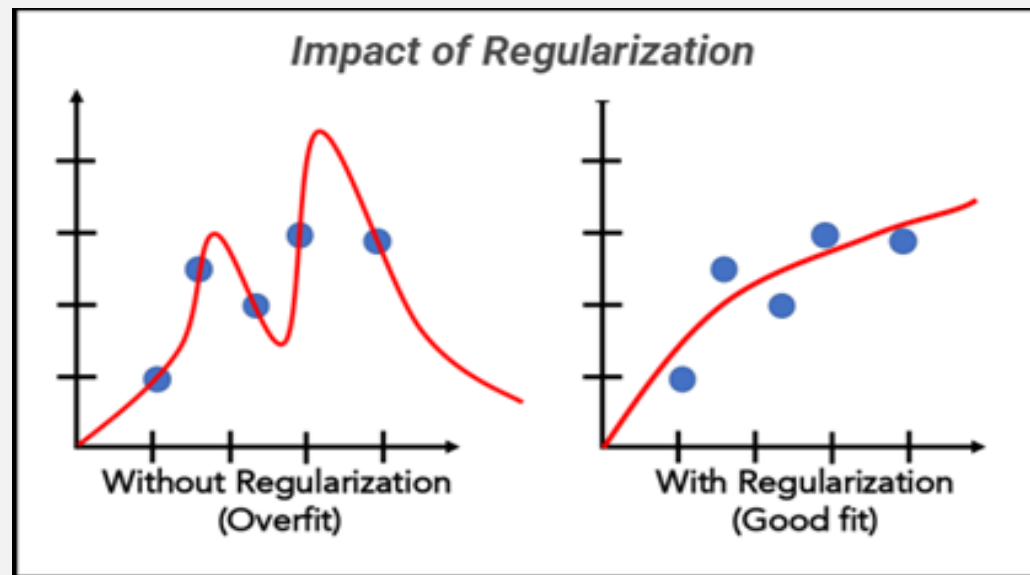


# L1 and L2 Regularization

---

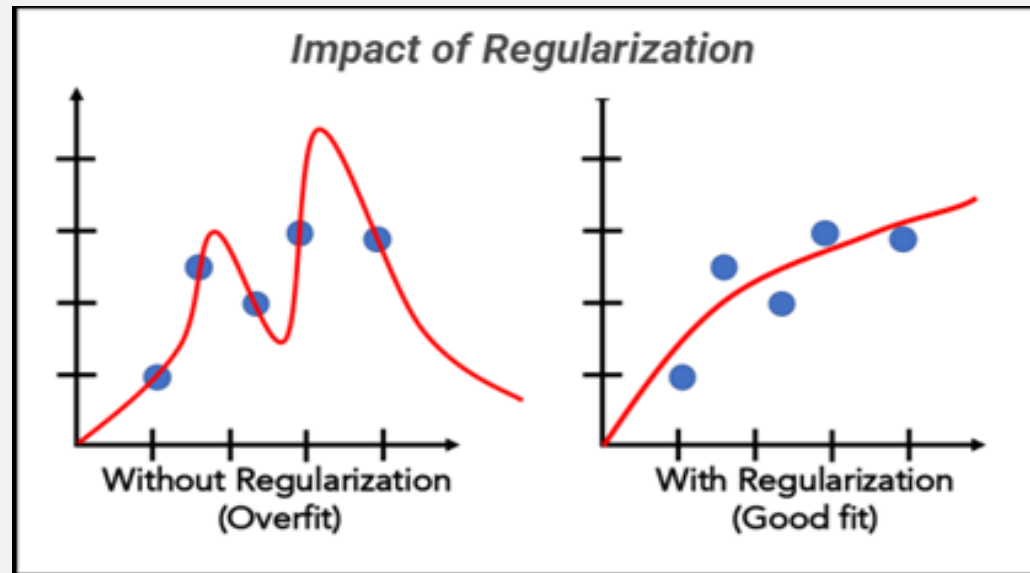
# ► Understanding Regularization in Machine Learning

- Regularization helps to prevent overfitting in machine learning models.
- Overfitting happens when a model learns the noise in the training data instead of the actual pattern.
- Regularization adds a penalty to the model's complexity, discouraging it from fitting the noise.



## ► What is Overfitting (Recap)

- Overfitting: When a model performs well on training data but poorly on new, unseen data.
- This occurs because the model learns too many details and noise from the training data.
- Regularization helps to reduce overfitting by simplifying the model.



## ► L1 and L2 Regularization

- Two common types: L1 (Lasso) and L2 (Ridge) Regularization.
- Both add a penalty to the loss function to discourage complex models.
- The main difference is how they add this penalty.

L1 Regularization

$$\text{Cost} = \sum_{i=0}^N (y_i - \sum_{j=0}^M x_{ij} W_j)^2 + \lambda \sum_{j=0}^M |W_j|$$

L2 Regularization

$$\text{Cost} = \underbrace{\sum_{i=0}^N (y_i - \sum_{j=0}^M x_{ij} W_j)^2}_{\text{Loss function}} + \lambda \underbrace{\sum_{j=0}^M W_j^2}_{\text{Regularization Term}}$$

## ► L1 Regularization (lasso)

- Adds the absolute value of the coefficients to the loss function.
- Promotes sparsity, which means it can shrink some coefficients to zero.
- Useful for feature selection by eliminating less important features.

L1 Regularization

$$\text{Cost} = \sum_{i=0}^N (y_i - \sum_{j=0}^M x_{ij} W_j)^2 + \lambda \sum_{j=0}^M |W_j|$$

## ► L1 Regularization (lasso) formula

- $\lambda$ : Regularization parameter that controls the strength of the penalty.
- $|w_i|$ : Absolute value of the coefficients.

### **L1 Regularization**

$$\begin{array}{l} \text{Modified loss} \\ \text{function} \end{array} = \text{Loss function} + \lambda \sum_{i=1}^n |w_i|$$

## ► L2 Regularization (Ridge)

- Adds the square of the coefficients to the loss function.
- Promotes smaller, non-zero coefficients, thus discouraging large coefficients.
- Useful for when you want to keep all features but reduce their impact.

L2 Regularization

$$\text{Cost} = \sum_{i=0}^N (y_i - \sum_{j=0}^M x_{ij} W_j)^2 + \lambda \sum_{j=0}^M W_j^2$$

## ▶ L2 Regularization (Ridge) Formula

- $\lambda$ : Regularization parameter that controls the strength of the penalty.
- $W_i^2$ : Square of the coefficients.

### L2 Regularization

$$\begin{array}{l} \text{Modified loss} \\ \text{function} \end{array} = \text{Loss function} + \lambda \sum_{i=1}^n W_i^2$$





## Comparing L1 and L2 Regularization

| L1 Regularization  | L2 Regularization   |
|--|---|
| Can zero out some coefficients, leading to simpler models. | Reduces the impact of less important features but keeps them. |
| Useful for feature selection.                              | Helps to stabilize models.                                    |



## Choose the Right Regularization

| L1 Regularization                                  | L2 Regularization   |
|--|---|
| When you need a simpler model with fewer features. | When you want to keep all features but reduce their effect. |
| When feature selection is important.               | When you need a more stable model.                          |



## Conclusion

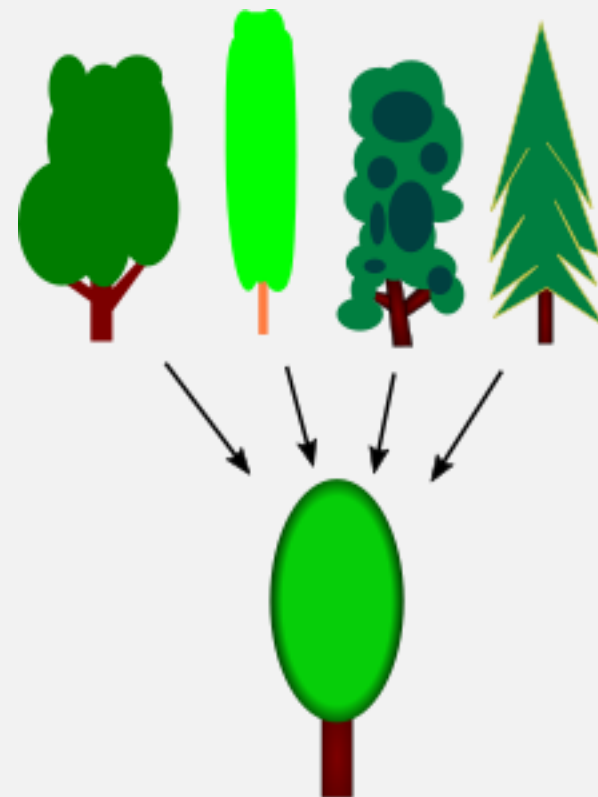
- Regularization is essential for preventing overfitting in machine learning models.
- L1 and L2 Regularization are two common techniques with different benefits.
- Choosing the right regularization depends on the specific needs of your model.

# Enhance Model Generalization and Robustness

---

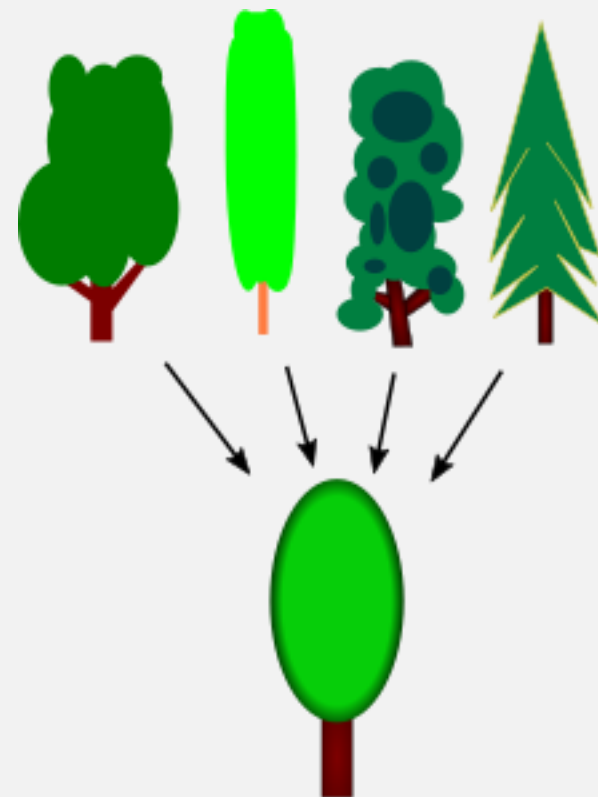
## ▶ Enhancing Model Generalization and Robustness

- Focus on making machine learning models more accurate and reliable.
- Generalization: How well a model performs on new, unseen data.
- Robustness: How well a model handles various types of data, including noisy or corrupted data.



# ► Importance of Generalization and Robustness

- Models must work well on real-world data, not just training data.
- Poor generalization leads to overfitting and inaccurate predictions.
- Lack of robustness makes models sensitive to noise and errors in data.





# Techniques to improve Generalization

- Cross-validation:
  - ❖ Split data into training and testing sets multiple times.
  - ❖ Ensure the model performs well across all splits.
- Regularization:
  - ❖ Add penalties for complexity (L1, L2 regularization).
  - ❖ Discourage the model from learning noise in the data.
- Dropout:
  - ❖ Randomly drop neurons during training.
  - ❖ Prevent the model from relying too heavily on specific paths.



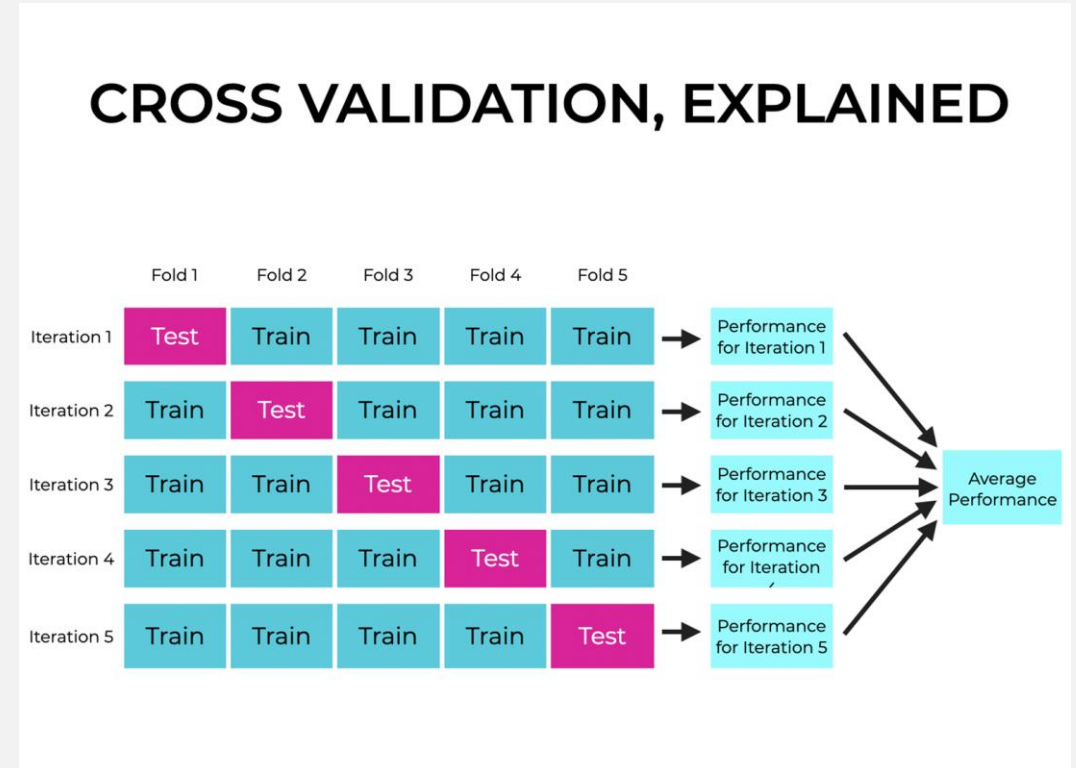
# Techniques to improve Robustness

- Data Augmentation:
  - ❖ Create new training samples by modifying existing data.
  - ❖ Helps the model learn to handle variations.
- Noise Injection:
  - ❖ Add random noise to training data.
  - ❖ Train the model to handle noisy inputs.
- Adversarial Training:
  - ❖ Use slightly altered inputs that could confuse the model.
  - ❖ Teach the model to be resilient against such alterations.



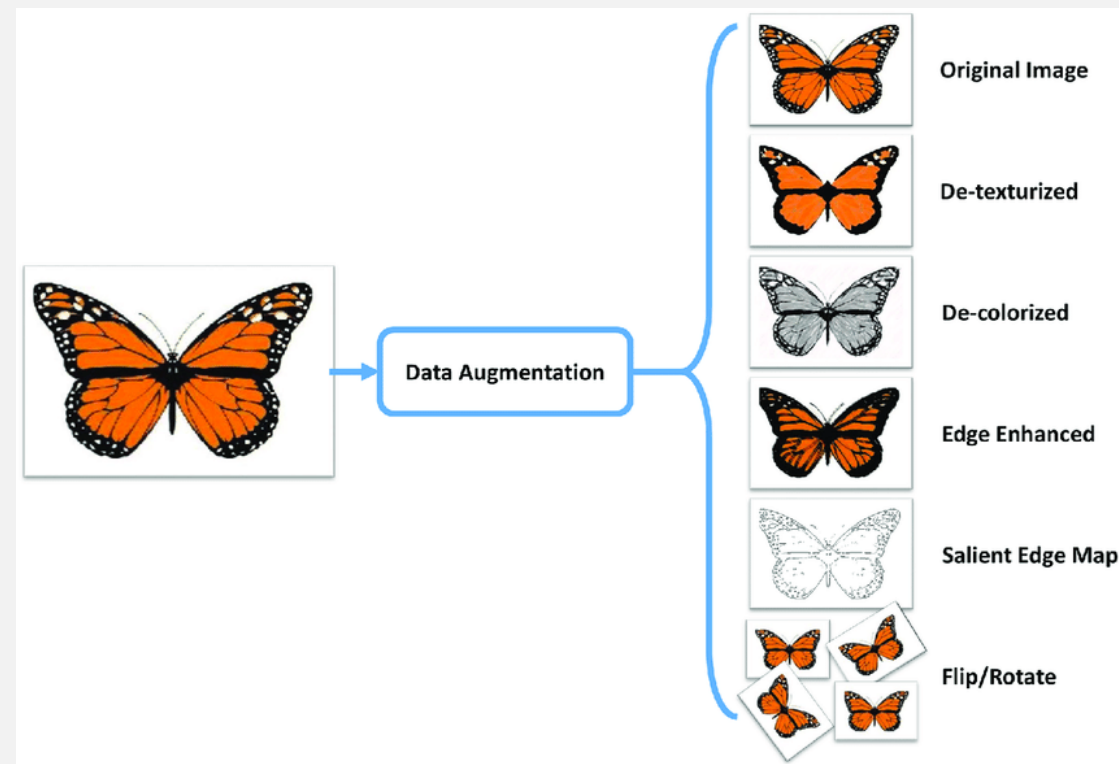
# Cross-validation

- Split the dataset into several parts.
- Train the model on different combinations of these parts.
- Test the model on the remaining parts.
- Helps ensure the model performs well on different data splits.



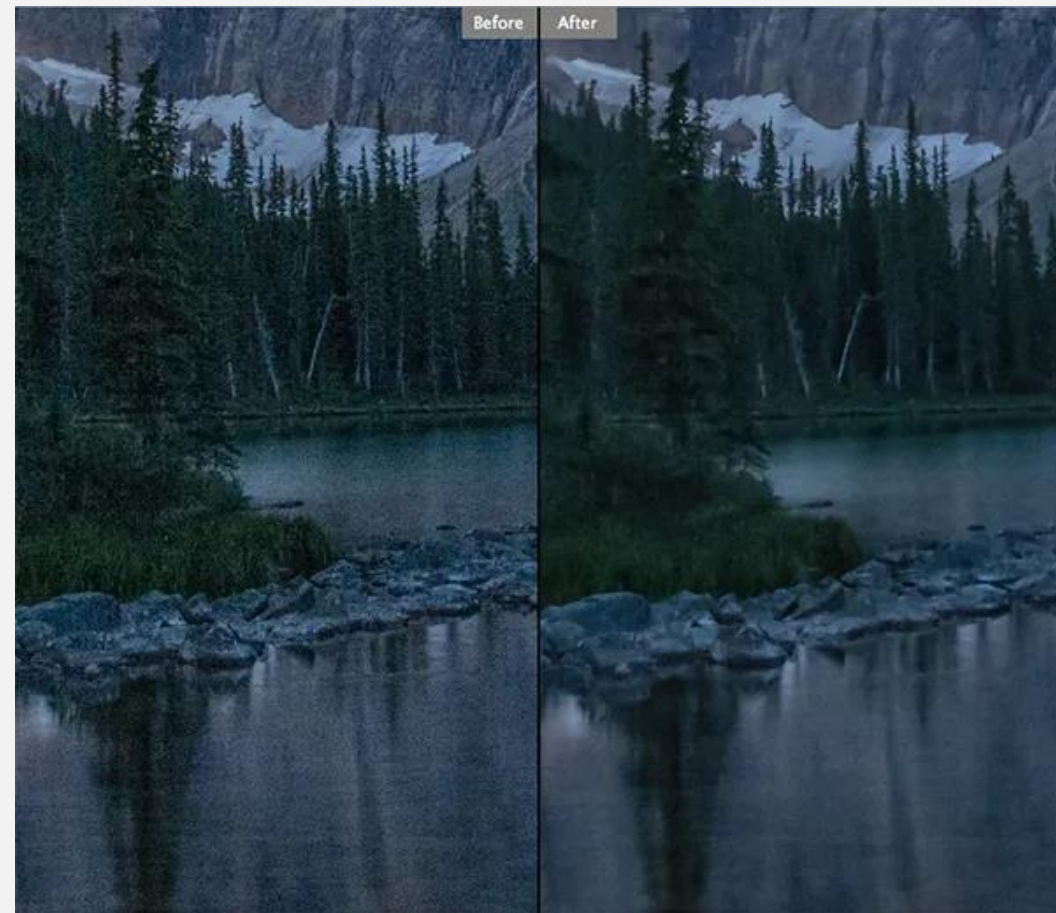
# ▶ Data Augmentation

- Generate new training data by altering existing data (e.g., rotating, flipping images).
- Helps the model learn to recognize data in different forms.
- Increases the diversity of the training dataset.



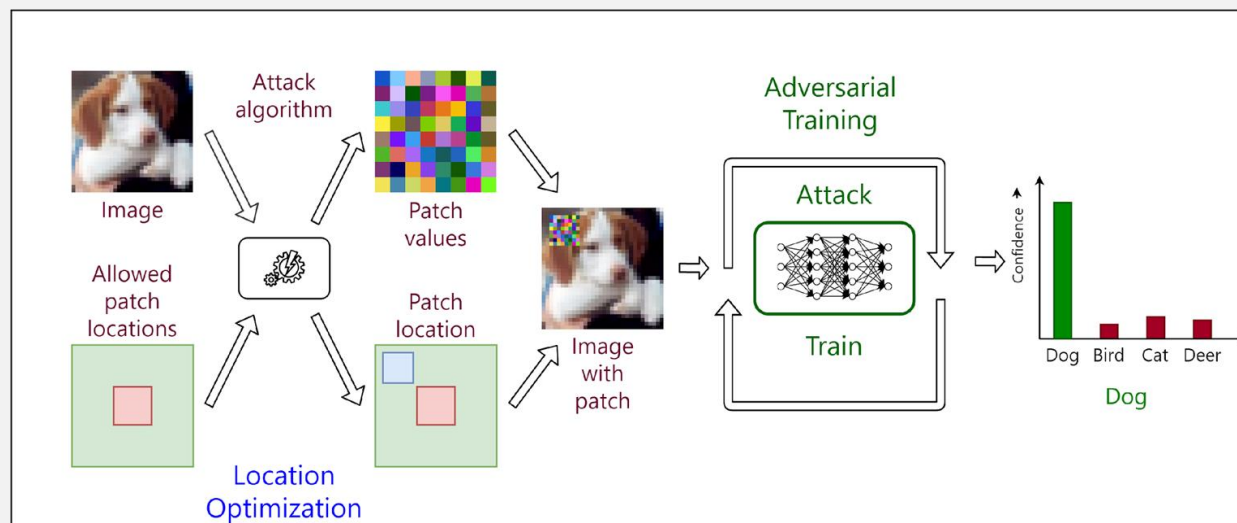
## ▶ Noise Injection

- Add random noise to the training data.
- Train the model to recognize patterns even with noisy inputs.
- Increases the model's robustness to imperfect data.



# Adversarial Training

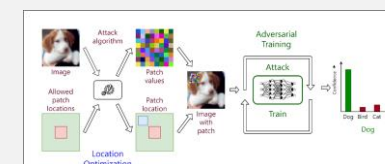
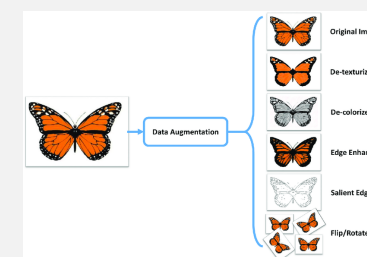
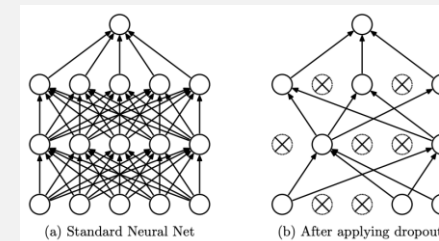
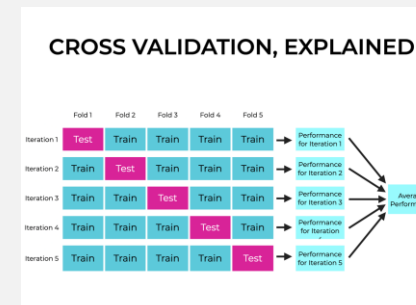
- Use inputs designed to trick the model.
- Teach the model to be resilient against small, intentional perturbations.
- Enhances the model's robustness against adversarial attacks.





# Putting it All Together

- Use a mix of these techniques to build models that generalize well and are robust.
- Experiment with different methods to see what works best for your data.
- Continuously evaluate and improve the model based on performance.





## Conclusion

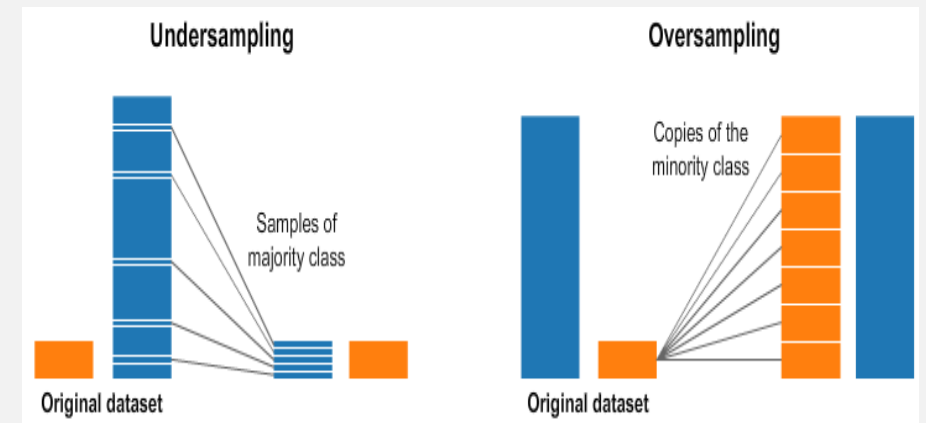
- Enhancing generalization and robustness is crucial for reliable machine learning models.
- Techniques like cross-validation, regularization, dropout, data augmentation, noise injection, and adversarial training are effective strategies.
- The goal is to create models that perform well on new data and are resilient to various types of data.

# Oversampling and Undersampling

---

# ► Understanding Oversampling and Undersampling

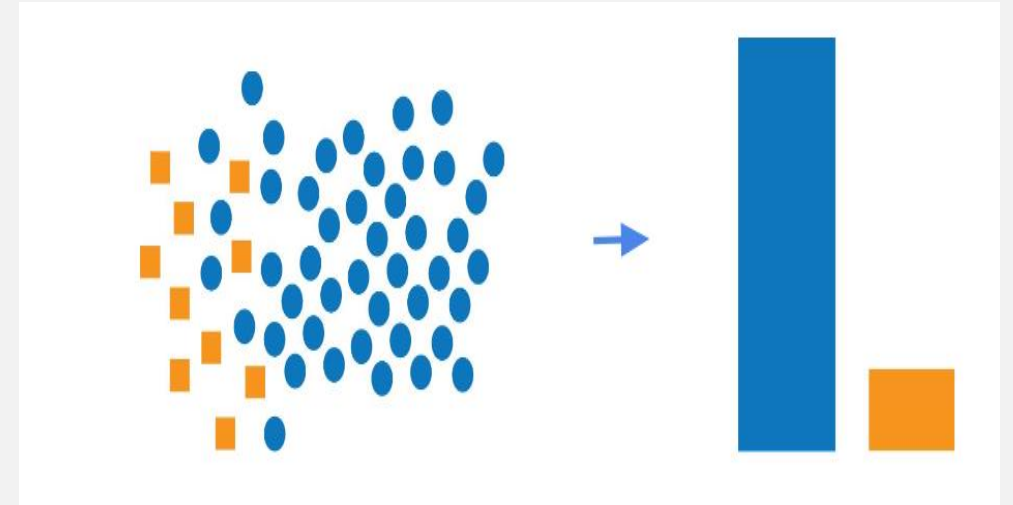
- Oversampling and undersampling are techniques to handle imbalanced datasets in machine learning.
- Imbalanced datasets have unequal class distributions.
- Oversampling and undersampling adjust the dataset to improve model performance.





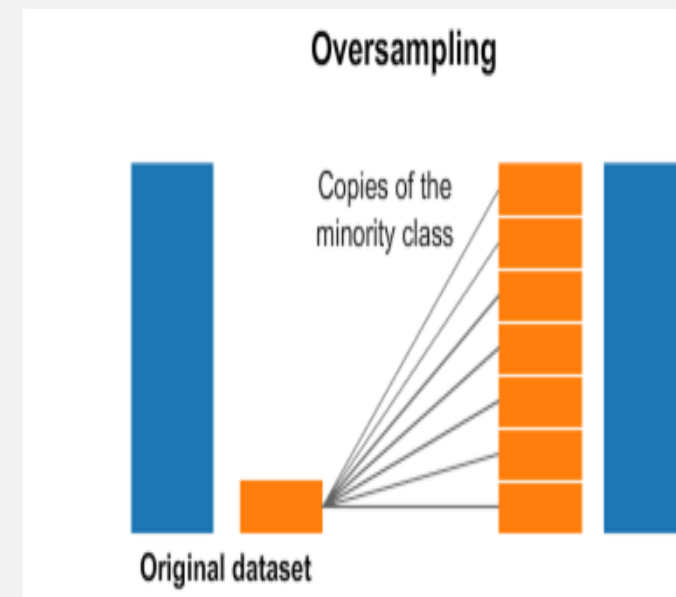
## ▶ Imbalanced Datasets

- Imbalance occurs when one class is significantly more prevalent than others.
- Can lead to biased models that favor the majority class.
- Oversampling and undersampling help balance these classes for better model training.



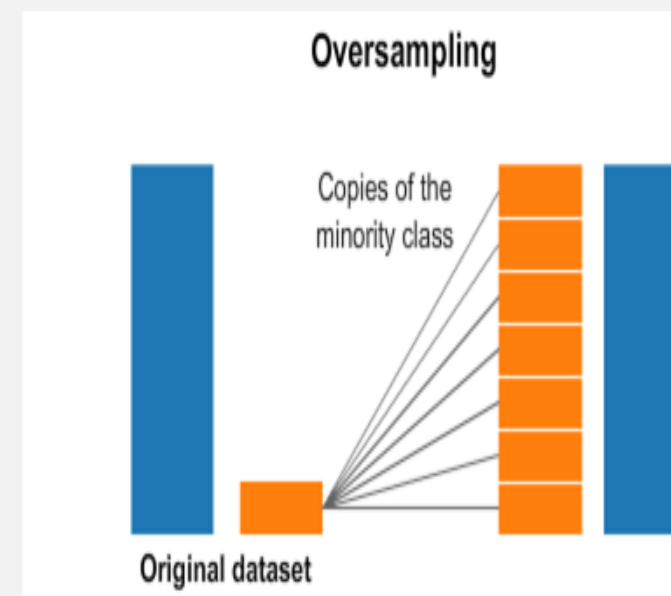
## ▶ Oversampling

- Increase the number of instances in the minority class.
- Helps the model learn from more examples of the minority class.
- Techniques include duplication, SMOTE (Synthetic Minority Over-sampling Technique), and ADASYN (Adaptive Synthetic Sampling).



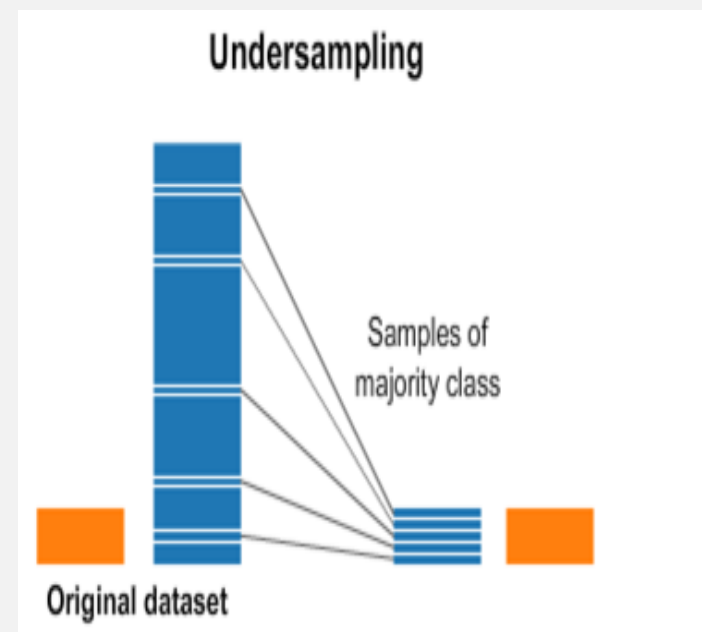
# ▶ Oversampling Techniques

- Duplicate:
  - ❖ Copy instances from the minority class to balance the dataset.
- SMOTE:
  - ❖ Generate synthetic samples by interpolating between existing minority class samples.
- ADASYN:
  - ❖ Focuses on generating samples near the existing minority class instances, emphasizing harder to learn cases.



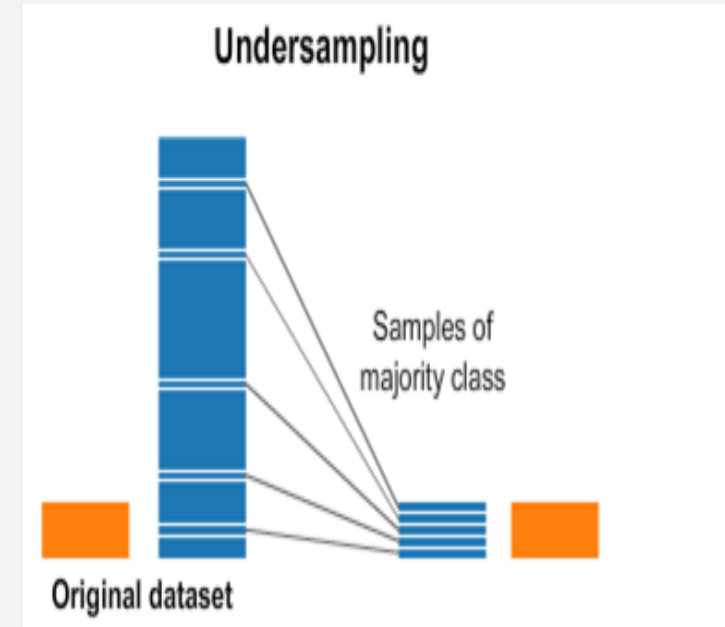
## Undersampling

- Decrease the number of instances in the majority class.
- Helps prevent the model from being biased towards the majority class.
- Techniques include random undersampling and cluster-based undersampling.



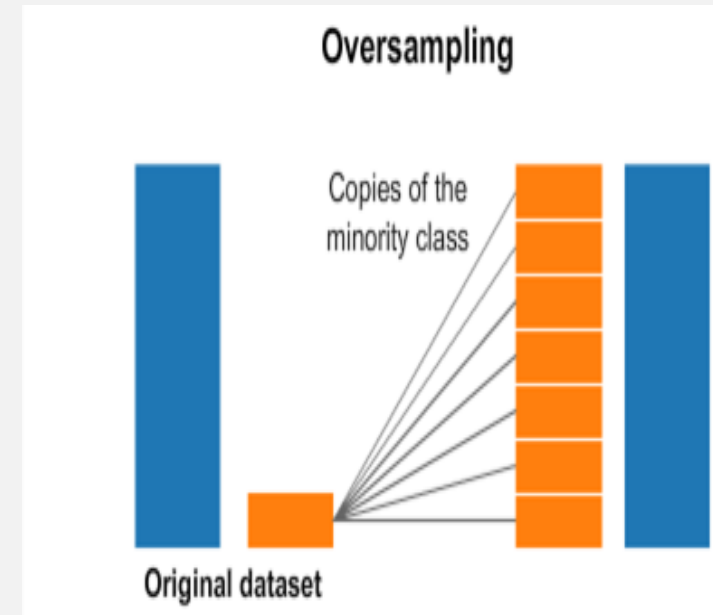
# ▶ Oversampling Techniques

- Random undersampling:
  - ❖ Randomly remove instances from the majority class.
- Cluster-based undersampling
  - ❖ Use clustering algorithms to identify clusters of majority class instances and then remove instances from these clusters.



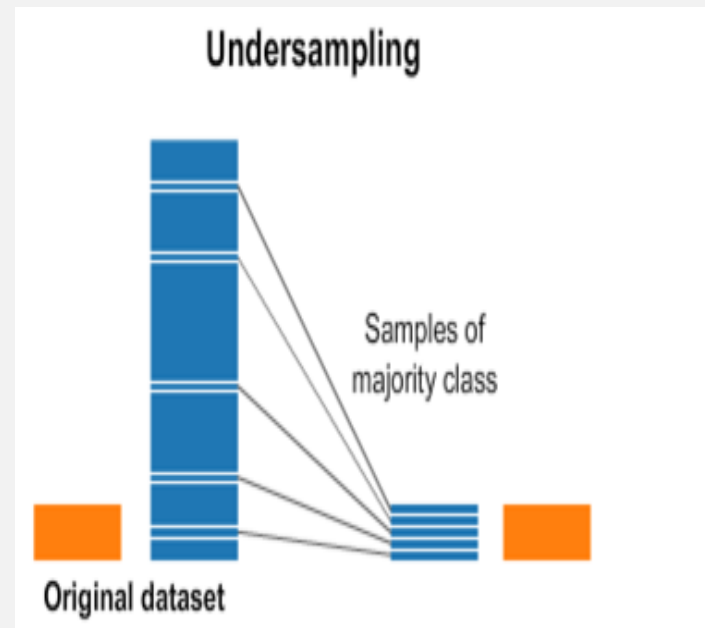
## ▶ Pros and Cons of Undersampling

- Pros:
  - ❖ Increases the amount of data for the minority class.
  - ❖ Helps in learning patterns from underrepresented classes.
- Cons:
  - ❖ May lead to overfitting if not carefully implemented.
  - ❖ Can increase computation time and resource usage.



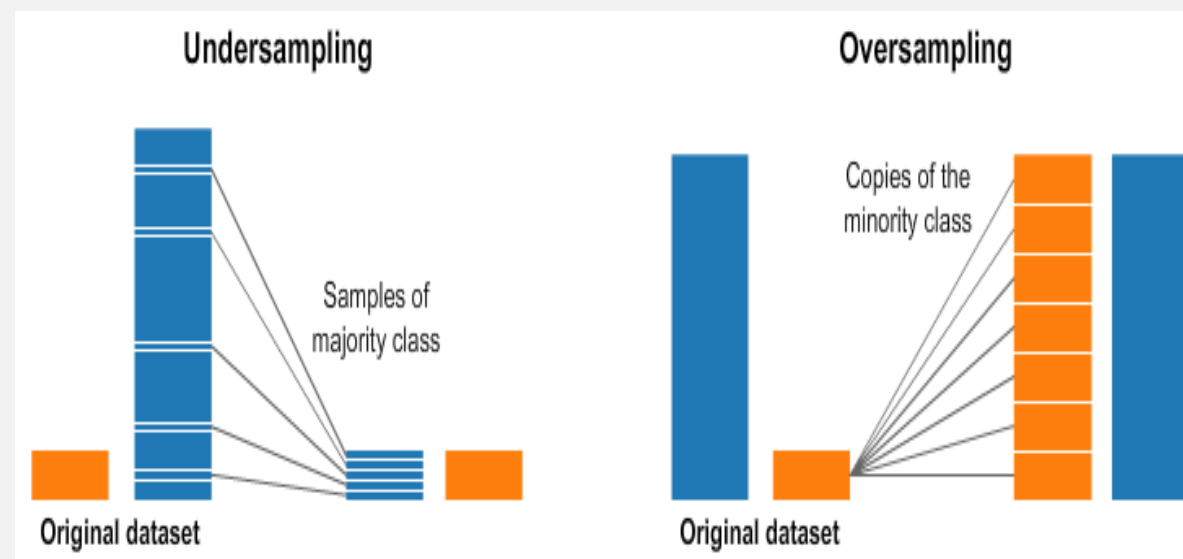
## ▶ Pros and Cons of Undersampling

- Pros:
  - ❖ Reduces the amount of data in the majority class.
  - ❖ Helps in balancing class distribution.
- Cons:
  - ❖ May discard useful information from the majority class.
  - ❖ Can lead to loss of important patterns.



# ▶▶ Choosing between Oversampling and Undersampling

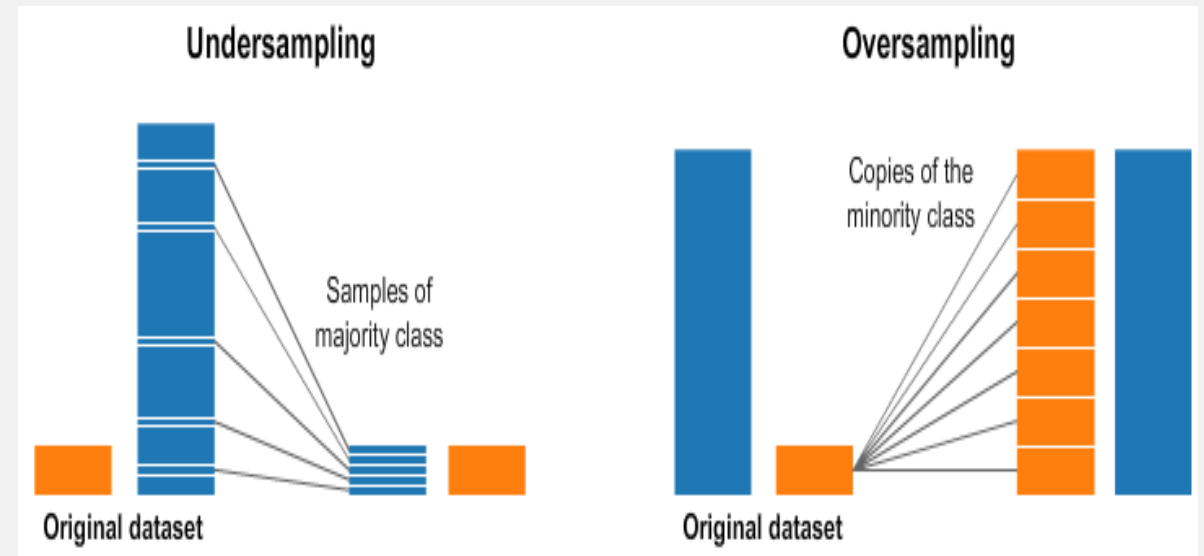
- Select based on dataset characteristics and problem requirements.
- Consider the impact on model performance and computational resources.
- Experiment with both techniques to determine which works best for your specific dataset.





## Best Practices

- Evaluate model performance before and after applying sampling techniques.
- Use cross-validation to assess the effectiveness of sampling.
- Combine with other techniques like regularization for enhanced model performance.





## Conclusion


- Oversampling and undersampling are effective strategies to handle imbalanced datasets in machine learning.
- Each technique has its advantages and challenges.
- Choose the method that best fits your dataset characteristics and modeling goals.

# Tutorial

**\*From: 8.A.c to: 8.A.d\***

<https://www.kaggle.com/code/kanncaa1/machine-learning-tutorial-for-beginners>

---



# References

- <https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.analyticsvidhya.com%2Fblog%2F2021%2F11%2Fstudy-of-regularization-techniques-of-linear-model-and-its-roles%2F&psig=AOvVaw2J9M1m3sLTSVBCGPnVOAOJ&ust=1720447155163000&source=images&cd=vfe&opi=89978449&ved=0CBEQjRxqFwoTCLjP16GLIYcDFQAAAAAdAAAAABAE>
- <https://www.google.com/url?sa=i&url=https%3A%2F%2Fmedium.com%2Fanalytics-vidhya%2FI1-vs-l2-regularization-which-is-better-d01068e6658c&psig=AOvVaw1BbtFVzG9nzaSNbljEx5TJ&ust=1720447444972000&source=images&cd=vfe&opi=89978449&ved=0CBEQjRxqFwoTCLCupa6MIYcDFQAAAAAdAAAAABAE>
- <https://www.google.com/url?sa=i&url=https%3A%2F%2Fmedium.com%2Fdata-science-365%2Fhow-to-apply-l1-and-l2-regularization-techniques-to-keras-models-da6249d8a469&psig=AOvVaw1S7nIm4VFp-28AtvoWfxSD&ust=1720447822074000&source=images&cd=vfe&opi=89978449&ved=0CBEQjRxqFwoTCjiI3OCNIYcDFQAAAAAdAAAAABAE>
- <https://www.google.com/url?sa=i&url=https%3A%2F%2Fen.wikipedia.org%2Fwiki%2FGeneralization&psig=AOvVaw04o7b7sMN-3UxGaSrR8d6b&ust=1720511013967000&source=images&cd=vfe&opi=89978449&ved=0CBEQjRxqFwoTCQjsxpz5IocDFQAAAAAdAAAAABAE>
- <https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.sharpsightlabs.com%2Fblog%2Fcross-validation-explained%2F&psig=AOvVaw1QCUyZwHQwhVEwCwjyAzfk&ust=1720511555477000&source=images&cd=vfe&opi=89978449&ved=0CBEQjRxqFwoTCNjju5L7locDFQAAAAAdAAAAABAK>
- <https://www.google.com/url?sa=i&url=https%3A%2F%2Fpaperswithcode.com%2Fmethod%2Fdropout&psig=AOvVaw3WOYbpY-ONX2uXXuPX6hoU&ust=1720511676995000&source=images&cd=vfe&opi=89978449&ved=0CBEQjRxqFwoTCKCC-cv7locDFQAAAAAdAAAAABAE>
- <https://www.google.com/url?sa=i&url=https%3A%2F%2Fmedium.com%2Fsecure-and-private-ai-writing-challenge%2Fdata-augmentation-increases-accuracy-of-your-model-but-how-aa1913468722&psig=AOvVaw1m7qkrxRVj8tqajn6F89Je&ust=1720511848966000&source=images&cd=vfe&opi=89978449&ved=0CBEQjRxqFwoTCMimsp78locDFQAAAAAdAAAAABAE>
- <https://www.google.com/url?sa=i&url=https%3A%2F%2Fdigital-photography-school.com%2Fhow-to-avoid-and-reduce-noise-in-your-images%2F&psig=AOvVaw3tynWoTA2pGtoo4f8DyOR1&ust=1720513779499000&source=images&cd=vfe&opi=89978449&ved=0CBEQjRxqFwoTCODn47aDI4cDFQAAAAAdAAAAABAg>
- [https://www.google.com/url?sa=i&url=https%3A%2F%2Fsukrutrao.github.io%2Fproject%2Fadversarial-patch-training%2F&psig=AOvVaw2SSS2OhfcES-rRSjPcivx\\_&ust=1720513920579000&source=images&cd=vfe&opi=89978449&ved=0CBEQjRxqFwoTCJj06\\_qDI4cDFQAAAAAdAAAAABAK](https://www.google.com/url?sa=i&url=https%3A%2F%2Fsukrutrao.github.io%2Fproject%2Fadversarial-patch-training%2F&psig=AOvVaw2SSS2OhfcES-rRSjPcivx_&ust=1720513920579000&source=images&cd=vfe&opi=89978449&ved=0CBEQjRxqFwoTCJj06_qDI4cDFQAAAAAdAAAAABAK)
- <https://www.google.com/url?sa=i&url=https%3A%2F%2Fmedium.com%2Fanalytics-vidhya%2Fundersampling-and-oversampling-an-old-and-a-new-approach-4f984a0e8392&psig=AOvVaw2o39A11N1fxsNbanLjDwZ-&ust=1720514388327000&source=images&cd=vfe&opi=89978449&ved=0CBEQjRxqFwoTCLiFm9mFI4cDFQAAAAAdAAAAABAE>
- <https://www.google.com/url?sa=i&url=https%3A%2F%2Fwww.linkedin.com%2Fpulse%2Fsome-tricks-handling-imbalanced-dataset-image-m-farhan-tandia&psig=AOvVaw3g1PB2pmc9kVQtp1MTGxbF&ust=1720514520755000&source=images&cd=vfe&opi=89978449&ved=0CBEQjRxqFwoTCLiBtZiGI4cDFQAAAAAdAAAAABAE>

# Thank you!



**SDAIA**  
الهيئة السعودية للبيانات  
والذكاء الاصطناعي  
Saudi Data & AI Authority