Computational Strategies for Handling Imbalanced Data in Machine Learning

O. Olawale Awe, PhD.
VP-IASE
VP of Global Engagement, -LISA 2020 Global Network, USA

Agenda

- ➢ Introduction
- ➢ Understanding Data Imbalance
- ➢ Why does Data Imbalance matter?
- ➢ Approaches to address Data Imbalance
- ➢ Oversampling
- ➢ Undersampling
- ➢ Combining Resampling Techniques
- ➢ Algorithm-Level Techniques
- ➢ Evaluation Metrics
- ➢ Anomaly Detection
- ➢ Cost-Sensitive Learning
- ➢ Q & A

Definition of Machine Learning

Machine learning (ML) is a subset of artificial intelligence that empowers computers to learn and make predictions or decisions from data without explicit programming.

It involves the development of algorithms that iteratively analyze data, identify patterns, and make predictions/decisions.

These algorithms are trained using vast datasets to recognize relationships and trends, enabling systems to make predictions, classify information, or take actions based on new, unseen data.

ML has gained tremendous popularity in various fields.

Types of Machine Learning

Supervised learning: where models learn from labeled data to make predictions or classifications.

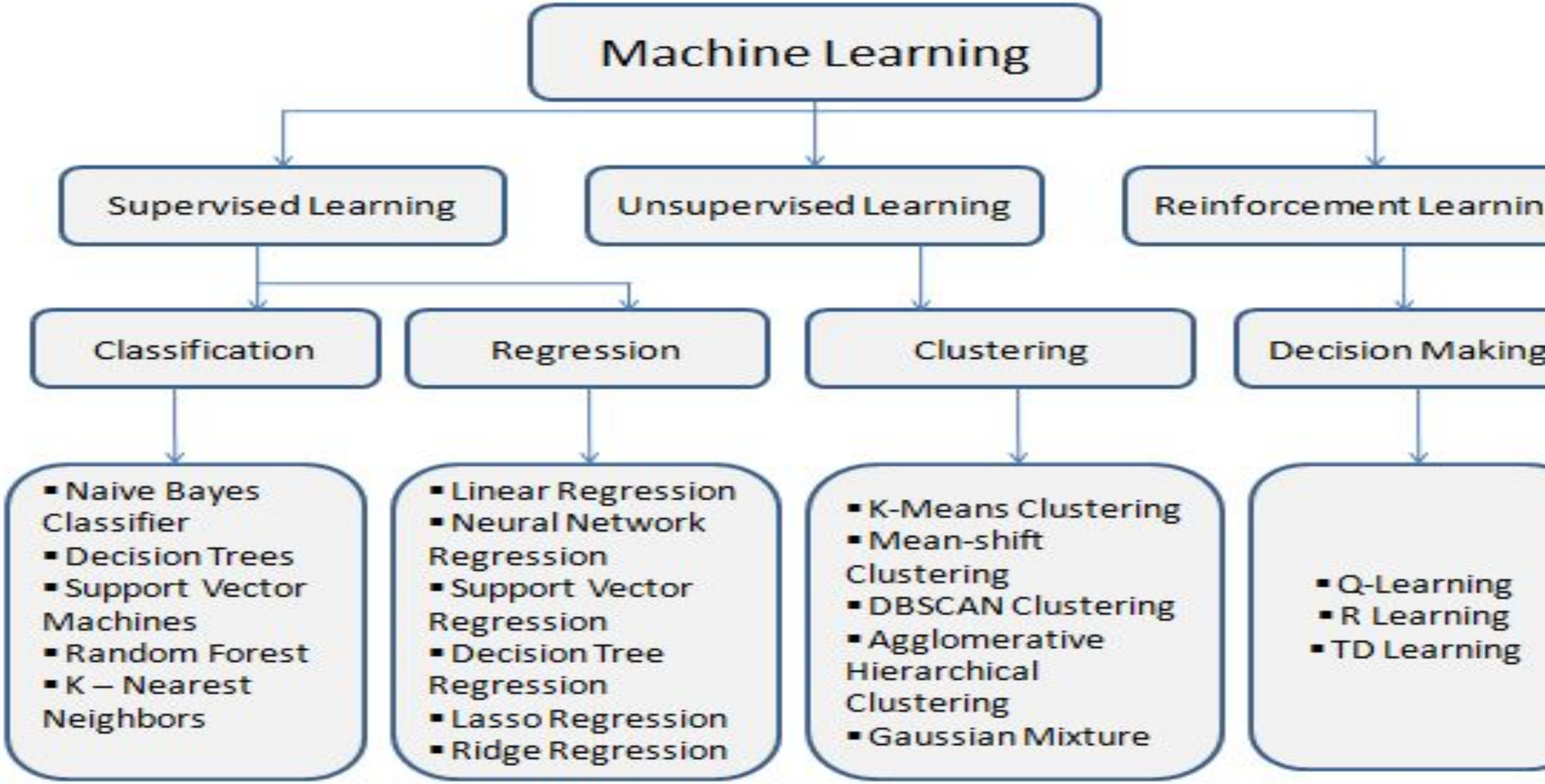
Unsupervised learning discovers hidden patterns within unlabeled data, useful in clustering and dimensionality reduction.

Reinforcement learning teaches agents to make decisions by trial and error, valuable in autonomous systems.

Semi-supervised learning combines labeled and unlabeled data to make predictions.

Transfer learning uses knowledge from one task to aid another.

Self-supervised learning generates labels from data itself.


Machine Learning
Supervised Learning
Unsupervised Learning
Reinforcement Learnin
Classification
Regression
Clustering
Decision Making
▪ Naive Bayes Classifier
▪ Decision Trees
▪ Support Vector Machines
▪ Random Forest
▪ K – Nearest Neighbors
▪ Linear Regression
▪ Neural Network Regression
▪ Support Vector Regression
▪ Decision Tree Regression
▪ Lasso Regression
▪ Ridge Regression
▪ K-Means Clustering
▪ Mean-shift Clustering
▪ DBSCAN Clustering
▪ Agglomerative Hierarchical Clustering
▪ Gaussian Mixture
▪ Q-Learning
▪ R Learning
▪ TD Learning

Data Imbalance

Data imbalance is a common and critical challenge in the field of machine learning.

It refers to a situation where the distribution of classes in a dataset is highly skewed, with one class significantly outnumbering the other(s).

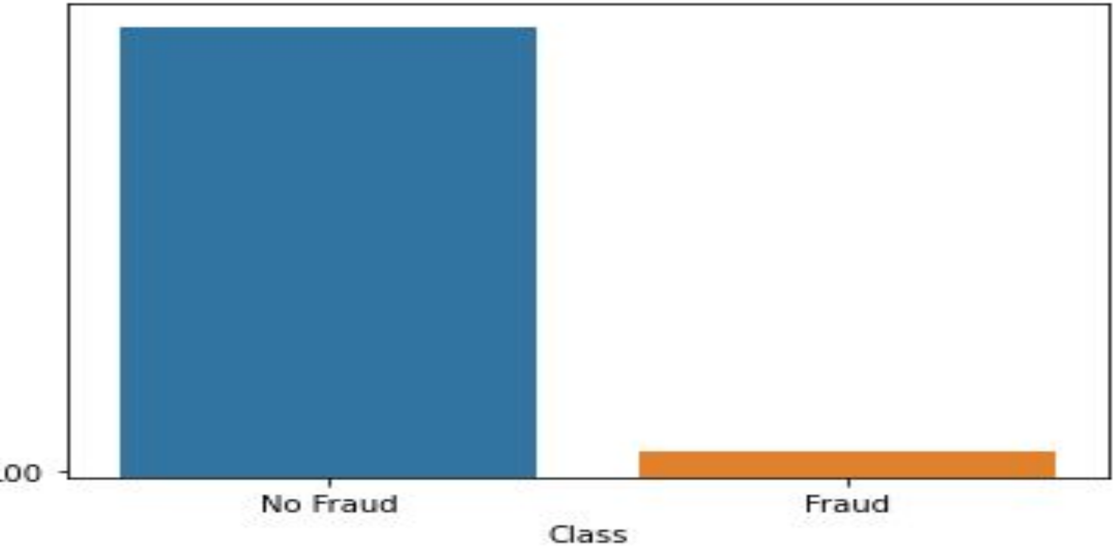
This issue can manifest in various real-world scenarios, such as fraud detection, medical diagnosis, text classification, and image recognition.

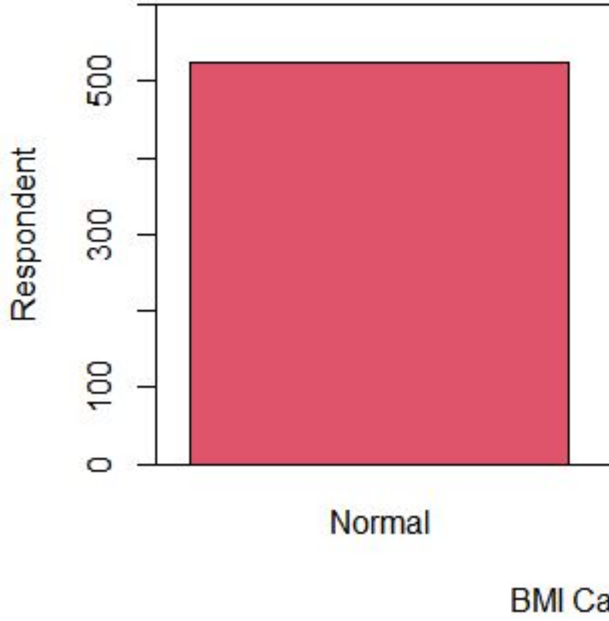
The consequences of data imbalance are substantial and can have a profound impact on the performance of machine learning models.

Machine learning algorithms are typically designed to optimize overall accuracy, which means they tend to favor the majority class.

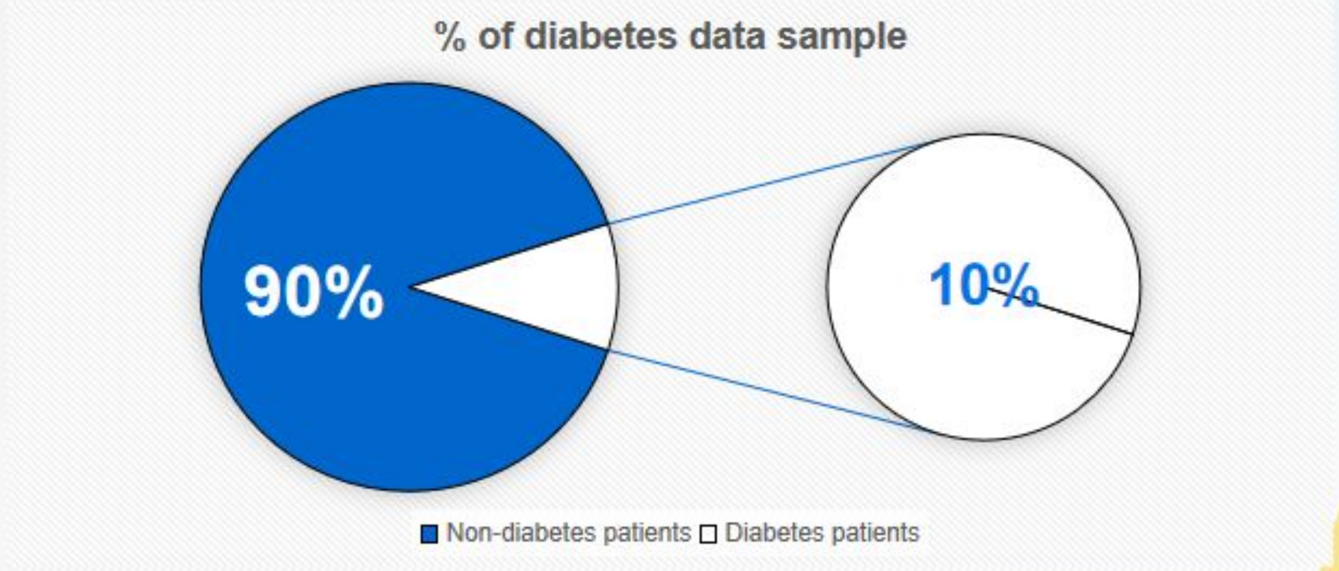
As a result, the minority class is often underrepresented in the model's learning process, leading to skewed predictions and poor generalization to the minority class.

Imbalanced Data Examples





Example of Class Imbalance in Diabetes data



In practical terms, models trained on imbalanced datasets may exhibit a high accuracy rate, but they are often ineffective at identifying and correctly classifying instances of the minority class.

In applications like fraud detection or medical diagnosis, this could result in undetected fraudulent transactions or missed critical diagnoses.

To address the challenges posed by data imbalance, various techniques and strategies have been developed.

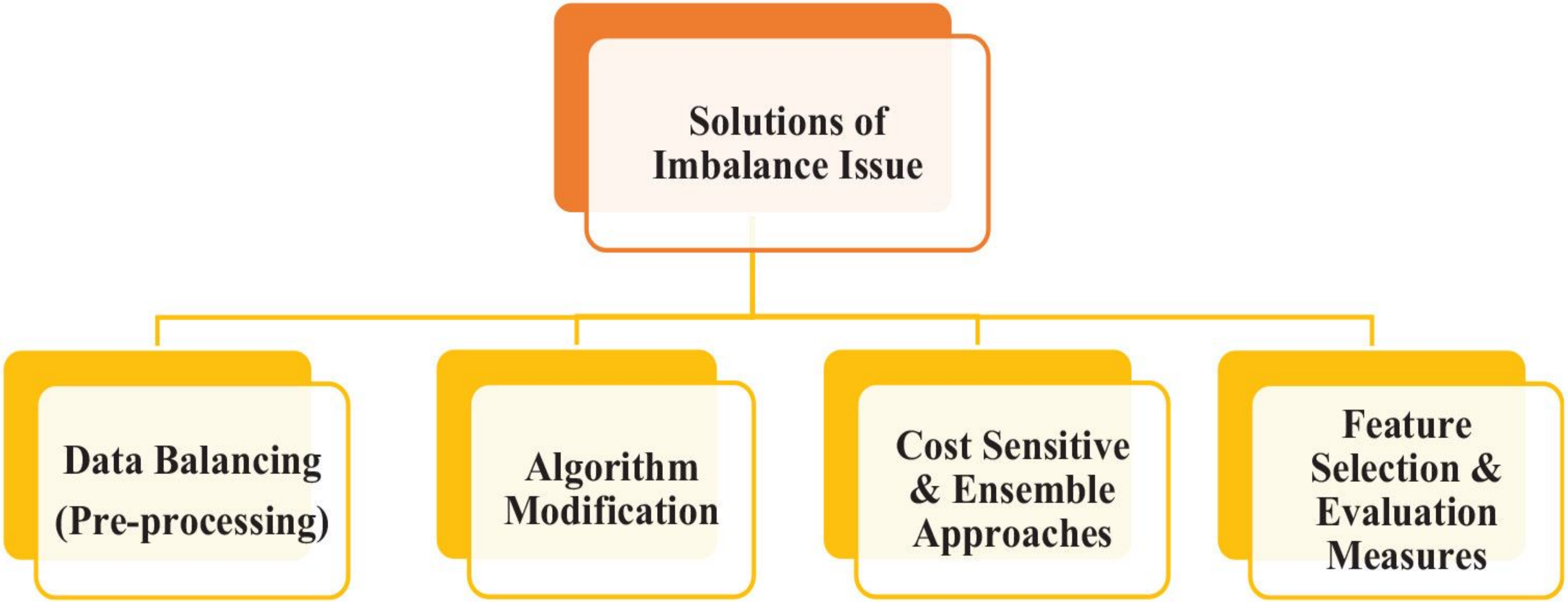
These methods aim to rebalance the dataset, adjust the model's learning process, or use specialized evaluation metrics that better reflect the performance on imbalanced data.

The selection of the most appropriate approach depends on the specific problem, the dataset, and the desired outcome.

In this talk, we will explore different techniques for handling data imbalance in machine learning and discuss when and how to use them effectively.

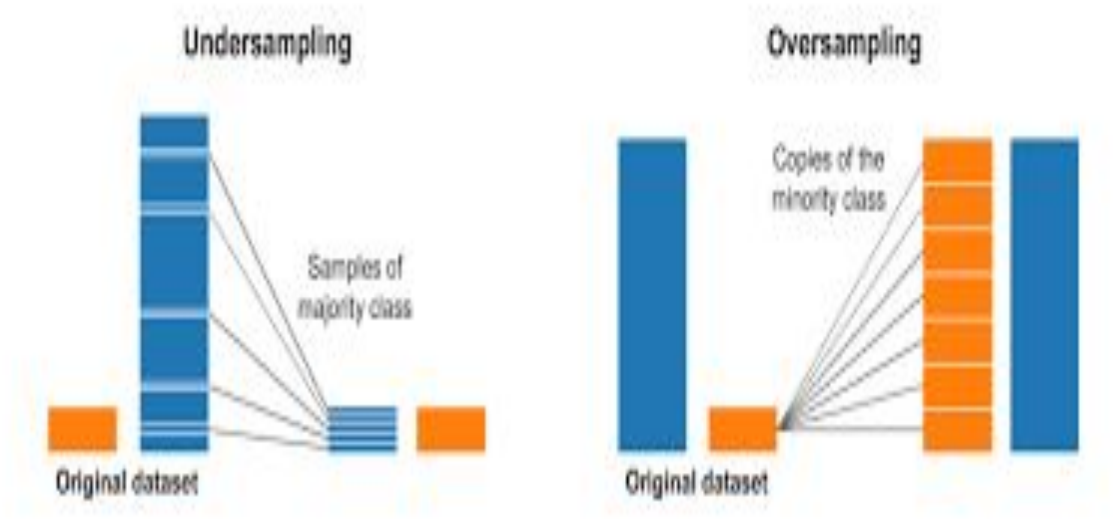
Consequences of Imbalance Data

- **Bias:** Imbalanced data can lead to model bias, where the model becomes overly influenced by the majority class. It may struggle to make accurate predictions for the minority class.
- **High Accuracy, Low Performance:** A model trained on imbalanced data may appear to have high accuracy but may perform poorly on minority classes, which are often the ones of greater interest.
- **Missed Insights:** Data imbalance can result in the loss of important insights and patterns present in the minority class, leading to missed opportunities or critical errors.
- Missclassifying an example of fraud or disease can be very costly!


Solutions of Imbalance Issue
Data Balancing (Pre-processing)
Algorithm Modification
Cost Sensitive & Ensemble Approaches
Feature Selection & Evaluation Measures

Approaches to Address Data Imbalance

1. **Data-Level Methods:**
- **Oversampling**
- **Undersampling**
- **Combined Resampling**



2. Algorithm-Level Techniques:

- Make the model more robust to class imbalance without changing the distribution of the training data.
- Some machine learning algorithms inherently handle imbalanced datasets better than others. For example, tree-based models like Random Forest and ensemble methods like AdaBoost often perform well with small imbalanced data.
- Building ensembles of multiple models can enhance predictive performance on imbalanced datasets.
- Anomaly Detection: In cases of extreme imbalance, treating the minority class as anomalies and using anomaly detection techniques, such as One-Class SVM or Isolation Forest, can be effective.

3. Cost-Sensitive Learning:

- Assigning different misclassification costs to classes can encourage the model to focus on the minority class. This is particularly useful when the misclassification costs are not equal, and the consequences of errors vary between classes.

4. Evaluation Metrics:

- When working with imbalanced datasets, it's important to use appropriate evaluation metrics.
- Common metrics include:
  - **Precision:** The proportion of true positive predictions among all positive predictions.
  - **Recall (Sensitivity):** The proportion of true positive predictions among all actual positives.
  - **F1-Score:** The harmonic mean of precision and recall.
  - **AUC-ROC (Area Under the Receiver Operating Characteristic Curve):** Measures the trade-off between true positive rate and false positive rate.
  - **Matthew's Correlation Coefficient (MCC)**

5. Hybrid Methods:

- Combining multiple techniques and approaches can provide robust solutions for handling data imbalance. For instance, combining resampling with algorithm-level adjustments and cost-sensitive learning can yield improved results.

The choice of which approach to use depends on the specific problem, the characteristics of the dataset, and the goals of the machine learning task.

It is often necessary to experiment with different techniques and evaluate their impact on model performance using appropriate evaluation metrics to determine the most effective approach for addressing data imbalance.

1. Resampling Techniques for Handling Data Imbalance

Resampling techniques are a common set of strategies used to address data imbalance in machine learning.

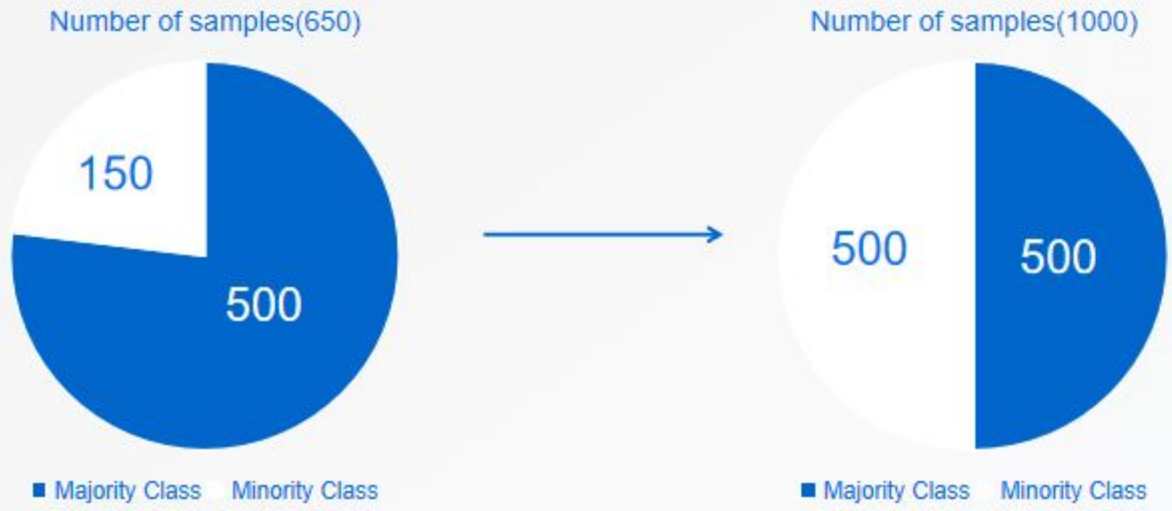
These techniques involve modifying the dataset by either increasing the number of minority class samples (oversampling) or reducing the number of majority class samples (undersampling). Here are some key resampling techniques:

1. **Oversampling:**
   - **Random Oversampling:** In this method, random instances from the minority class are duplicated until a more balanced distribution is achieved. While this can balance the class distribution, it may lead to overfitting.
   - **SMOTE (Synthetic Minority Over-sampling Technique):** SMOTE generates synthetic instances for the minority class by interpolating between neighboring instances. This approach creates new, realistic data points and helps prevent overfitting compared to random oversampling.

Oversampling Techniques

- ADASYN
- Random oversampler
- SMOTE
- SMOTEN
- Borderline-SMOTE
- Kmeans-SMOTE
- svm-SMOTE

Oversampling



Advantages of Oversampling:

- It mitigates class imbalance, reducing the model's bias towards the majority class.
- It can improve the model's ability to correctly classify instances from the minority class.
- Oversampling is relatively easy to implement and can be combined with other techniques for further enhancement.

Considerations and Potential Challenges:

- **Overfitting:** Random oversampling may lead to overfitting if it significantly increases the number of minority class instances.
- **Increased Computational Load:** Generating synthetic instances in SMOTE can increase the computational load, especially for large datasets.
- **Evaluation Metrics:** After oversampling, it's essential to use appropriate evaluation metrics like precision, recall, and F1-score, as accuracy alone may not provide a complete picture of model performance.

When to Use Oversampling:

Oversampling is beneficial when you want to balance the class distribution in a dataset with an imbalanced distribution and when there's a concern that the minority class may be underrepresented during model training.

It is commonly used in tasks such as fraud detection, medical diagnosis, and other applications where the cost of false negatives is high.

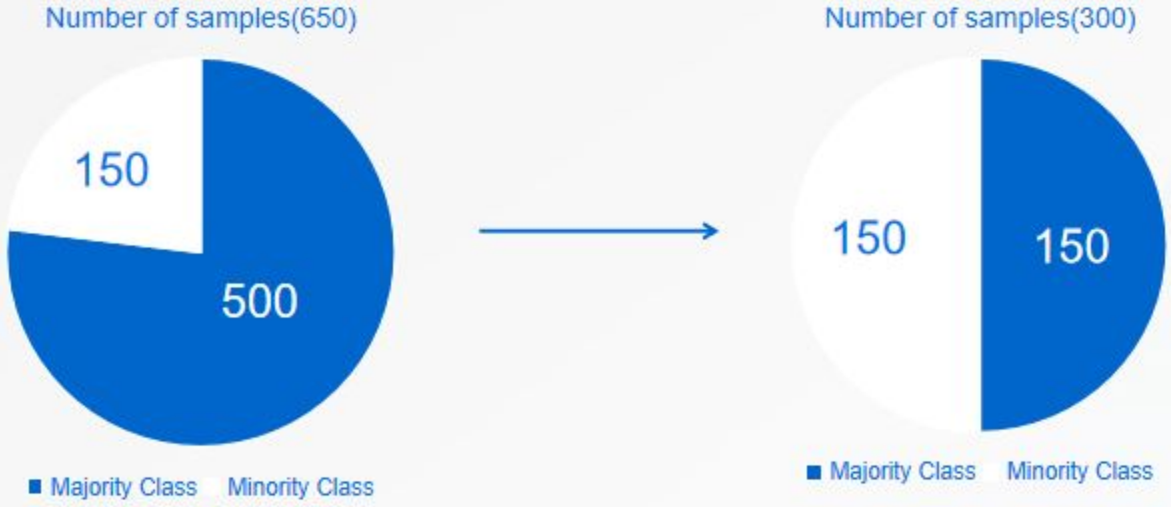
B. Undersampling

Undersampling is a resampling technique used to address data imbalance in machine learning.

It involves reducing the number of instances in the majority class, which is the class with more examples, to create a more balanced dataset.

The primary objective of undersampling is to prevent the model from being overwhelmed by the majority class and to ensure that the minority class is given equal importance in the learning process.

Undersampling



Methods of Undersampling:

1. **Random Undersampling:** In this method, a random subset of instances from the majority class is selected and retained, effectively reducing the number of majority class instances.

   While this technique can help balance the class distribution, it may result in the loss of potentially valuable information from the majority class.

2. **Tomek Links:** Tomek links are pairs of instances, one from the minority class and one from the majority class, that are each other's nearest neighbors.

   Removing the majority class instances in these pairs helps create a clearer separation between the two classes, potentially improving classification performance.

   This method is a selective undersampling technique.

Undersampling Techniques

- Condensed Nearest Neighbor
- Random undersampler
- Edited Nearest Neighbor
- Repeated Edited Nearest Neighbor
- TomekLinks
- NearMiss

Advantages of Undersampling:

- It mitigates class imbalance, reducing the dominance of the majority class in the learning process.
- Undersampling can make model training faster and more computationally efficient because the dataset is smaller.
- In some cases, it can lead to improved interpretability of the model.

Considerations and Potential Challenges:

- **Information Loss:** Random undersampling can lead to a significant loss of data and potentially valuable information from the majority class.
- **Reduced Model Generalization:** When the majority class is undersampled, the model may have less data to learn from and may not generalize well to new, unseen data.
- **Choice of Instances:** Care must be taken when selecting instances for removal. Random undersampling might inadvertently remove informative instances, and Tomek links may not always be straightforward to identify in complex datasets.

C. Combining Resampling Techniques

Combining resampling techniques is a powerful strategy for addressing data imbalance in machine learning.

It involves using a combination of oversampling and undersampling methods to create a balanced dataset.

By striking a balance between the two, this approach aims to mitigate the drawbacks of each technique while leveraging their advantages.

- Sometimes, using both oversampling and undersampling in combination can lead to a more balanced dataset. For instance, you can oversample the minority class and simultaneously undersample the majority class. The aim is to strike a balance between preserving information and mitigating class imbalance.

Methods of Combining Resampling Techniques

1. **SMOTEENN:** SMOTEENN combines SMOTE (Synthetic Minority Over-sampling Technique) with Edited Nearest Neighbors (ENN). Here's how it works:
   - SMOTE generates synthetic instances for the minority class.
   - ENN then identifies and removes noisy or misleading instances from both classes.
   - The result is a dataset that has been oversampled with synthetic instances while simultaneously reducing the number of majority class instances.
2. **SMOTETomek:** SMOTETomek combines SMOTE with Tomek links, The process is as follows:
   - SMOTE generates synthetic instances for the minority class.
   - Tomek links are identified, and majority class instances involved in Tomek links are removed.
   - The result is a dataset that combines oversampling of the minority class with the removal of noisy majority class instances.

Benefits of Combining Resampling Techniques

- **Balanced and Informative Dataset:** Combining oversampling and undersampling can result in a balanced dataset that retains relevant information from both the majority and minority classes. It strikes a balance between addressing class imbalance and reducing the risk of overfitting.
- **Reduced Risk of Overfitting:** While oversampling can potentially lead to overfitting, combining it with undersampling helps control this issue by removing some majority class instances.
- **Improved Model Generalization:** By providing the model with a dataset that better represents both classes, combining resampling techniques can improve the model's generalization to unseen data.
- **Enhanced Model Performance:** Models trained on balanced, informative datasets often demonstrate better performance, particularly when working with imbalanced data.

Considerations:

- The choice of combining resampling techniques should be guided by the characteristics of the dataset and the specific problem. It may not be the best approach for all situations.
- Depending on the problem, you can also experiment with different combinations of oversampling and undersampling techniques to find the most effective balance.
- Care must be taken when selecting and fine-tuning the specific resampling methods and their parameters to achieve the desired balance and model performance.

When to use Combining Resampling Techniques

Combining resampling techniques is particularly beneficial when you want to strike a balance between addressing data imbalance and reducing the risk of overfitting.

It is often used in scenarios where the choice between pure oversampling or undersampling is not clear-cut, and a combination of the two may offer the best results.

Ultimately, the choice of resampling techniques, whether combined or standalone, should be based on a thorough understanding of the dataset, problem requirements, and the goals of the machine learning task.

2. Algorithm-Level Techniques for Handling Data Imbalance

In addition to resampling techniques, algorithm-level methods are another set of strategies used to address data imbalance in machine learning.

These techniques involve adjusting the algorithms themselves to handle imbalanced datasets more effectively. Here are some key algorithm-level techniques:

1. **Class Weighting:**
   - Many machine learning algorithms allow you to assign different weights to different classes. By assigning higher weights to the minority class and lower weights to the majority class, you can instruct the model to pay more attention to the minority class during training. This is especially useful for algorithms like logistic regression and support vector machines.
2. **Cost-Sensitive Learning:**
   - Cost-sensitive learning is a more advanced version of class weighting. It involves explicitly assigning different misclassification costs to classes. Algorithms are then trained to minimize the overall cost, which can be skewed toward the minority class. This approach encourages the model to focus on correctly classifying the minority class, even if it results in more false positives in the majority class.

3. Algorithm Selection:

- Some machine learning algorithms are naturally better suited to handling imbalanced data. For example, tree-based models like Random Forest and ensemble methods like AdaBoost are often robust choices for imbalanced datasets. These models can handle class imbalance by design, as they partition data into subsets based on class and make decisions independently in each subset.

4. Threshold Adjustment:

- By modifying the classification threshold, you can control the trade-off between precision and recall in a binary classification problem. Lowering the threshold can increase recall, making the model more sensitive to the minority class, albeit at the cost of precision.

5. Anomaly Detection:

- For extreme class imbalance, you can treat the minority class as anomalies and employ specialized anomaly detection techniques. Methods like One-Class SVM, Isolation Forest, and Local Outlier Factor are designed to identify rare and unusual instances, making them well-suited for this scenario.

When to Use Algorithm-Level Techniques:

Algorithm-level techniques are particularly beneficial when you want to work with imbalanced data without modifying the dataset itself. You may choose these techniques when:

- Resampling techniques are not suitable due to dataset size limitations or concerns about information loss.
- You want to retain the original data distribution but need the model to better adapt to the imbalanced nature of the data.
- You prefer algorithms that inherently handle class imbalance without additional preprocessing steps.

These techniques can be used in conjunction with other approaches, such as resampling or combining resampling methods, to further enhance model performance on imbalanced datasets.

The choice of which algorithm-level technique to use depends on the specific problem, the machine learning algorithm in use, and the desired model behavior.

Experimentation and thorough evaluation using appropriate metrics are essential to determine the most effective approach for addressing data imbalance in your specific scenario.

Algorithms that can Handle Imbalanced Data

Several ML algorithms can handle imbalanced data well. The choice often depends on the specific characteristics of your dataset and the problem you're trying to solve. They include:

Random Forest (RF) and Decision Trees:

- They can handle imbalanced data because of their inherent ability to find decision boundaries that separate classes well, especially when combined with techniques like class weights or cost-sensitive learning.

Support Vector Machines (SVM):

- SVMs with appropriate kernel functions and adjusted class weights can work effectively with imbalanced data by focusing on support vectors and maximizing the margin between classes.

Neural Networks:

- Techniques like deep learning can handle imbalanced data, especially when using architectures with specific adjustments like class weighting, oversampling within the network, or focal loss functions.

Ensemble Methods:

- Ensemble methods like AdaBoost, XGBoost, LightGBM, or CatBoost, Bagging, or Stacking with base learners that handle imbalanced data well can be quite effective. Combining multiple models can often improve performance on imbalanced datasets.

Naive Bayes:

- Despite its simplicity, Naive Bayes can perform reasonably well with imbalanced data, especially when the class imbalance is not extreme.

3. Evaluation Metrics for Handling Data Imbalance

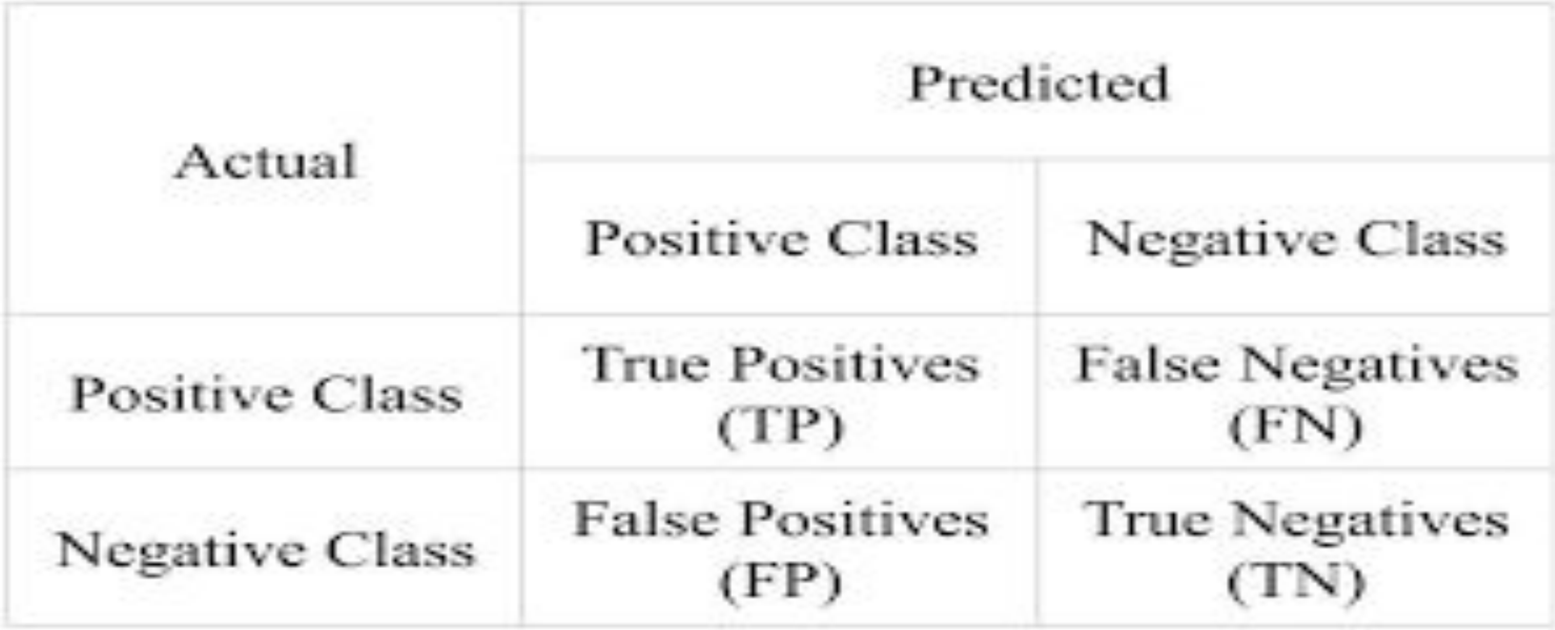
When dealing with imbalanced datasets in machine learning, using appropriate evaluation metrics is crucial.

Traditional metrics like accuracy may not provide an accurate assessment of a model's performance, as they can be misleading when one class significantly outnumbers the other(s).

Instead, it's important to focus on metrics that provide insights into how well a model is performing, particularly with respect to the minority class.

Here are some essential evaluation metrics for handling data imbalance:

Confusion Matrix



Actual	Predicted	
	Positive Class	Negative Class
Positive Class	True Positives (TP)	False Negatives (FN)
Negative Class	False Positives (FP)	True Negatives (TN)

1. **Precision:**
   - Precision measures the proportion of true positive predictions (correctly identified minority class instances) among all positive predictions (instances classified as the minority class). High precision indicates a low rate of false positives.
   - Formula: Precision = True Positives / (True Positives + False Positives)
2. **Recall (Sensitivity):**
   - Recall measures the proportion of true positive predictions among all actual positive instances (all actual minority class instances). High recall indicates a low rate of false negatives.
   - Formula: Recall = True Positives / (True Positives + False Negatives)
3. **F1-Score:**
   - The F1-score is the harmonic mean of precision and recall. It provides a balance between these two metrics. High F1-scores indicate a model that performs well on both precision and recall.
   - Formula: F1-Score = 2 * (Precision * Recall) / (Precision + Recall)

4. AUC-ROC (Area Under the Receiver Operating Characteristic Curve):

- AUC-ROC measures the trade-off between the true positive rate (sensitivity) and the false positive rate as the classification threshold varies. It provides a comprehensive view of a model's ability to distinguish between classes.

5. AUC-PR (Area Under the Precision-Recall Curve):

- AUC-PR focuses on the precision-recall trade-off, which is often more informative for imbalanced datasets. It quantifies the area under the precision-recall curve.

6. Confusion Matrix:

- The confusion matrix provides a detailed breakdown of the model's predictions. It includes the number of true positives, true negatives, false positives, and false negatives.
- The confusion matrix is used to derive metrics like precision, recall, and accuracy.

7. Specificity:

- Specificity measures the proportion of true negative predictions (correctly identified majority class instances) among all actual negative instances (all actual majority class instances). High specificity indicates a low rate of false positives.
- Formula: Specificity = True Negatives / (True Negatives + False Positives)

8. Balanced Accuracy:

- Balanced accuracy calculates the average of sensitivity (recall) and specificity, providing an overall assessment of a model's performance on both classes.
- Formula: Balanced Accuracy = (Sensitivity + Specificity) / 2

When working with imbalanced datasets, it is advisable to prioritize precision, recall, F1-score, and area under the precision-recall curve, as these metrics provide more insights into a model's performance, especially for the minority class.

The choice of metrics should align with the specific problems objectives, considering the relative importance of false positives and false negatives.

Additionally, these metrics should be used in conjunction with appropriate resampling or algorithm-level techniques to optimize model performance in imbalanced scenarios.

4. Anomaly Detection

Anomaly detection is a specialized technique for handling data imbalance in machine learning, particularly when one class (the anomaly or rare event) is vastly outnumbered by the other class (normal or majority class).

Instead of trying to create a balanced dataset, anomaly detection treats the minority class as the anomaly and focuses on identifying rare or unusual instances within the data.

Key Concepts in Anomaly Detection

1. **Anomalies or Outliers:** Anomalies are data points that deviate significantly from the majority of the data. In imbalanced datasets, the minority class is often treated as the anomaly.
2. **Detection Methods:** Anomaly detection methods are designed to identify these rare and unusual instances. They focus on patterns that differ from the majority of the data.
3. **Unsupervised Learning:** Anomaly detection is often performed through unsupervised learning, where the algorithm learns to identify anomalies without the need for labeled data.

Common Anomaly Detection Techniques

One-Class SVM (Support Vector Machine):

- One-Class SVM is a popular method for anomaly detection. It defines a hypersphere (or hyperplane in higher dimensions) that contains most of the data points. Instances that fall outside this hypersphere are considered anomalies.

Isolation Forest:

- Isolation Forest is a tree-based method that isolates anomalies by creating a random forest of decision trees. Anomalies are expected to require fewer splits to be isolated.

Local Outlier Factor (LOF):

- LOF is a density-based method that calculates the local density of instances and compares it to the density of their neighbors. Anomalies have much lower local densities than their neighbors.

Autoencoders:

- Autoencoders are neural network architectures used for unsupervised feature learning. They can be trained to reconstruct normal data. Instances that cannot be accurately reconstructed are considered anomalies.

K-Means Clustering:

- K-Means clustering can be used to identify anomalies by looking for data points that are distant from the cluster centers. These distant points are potential anomalies.

Benefits of Anomaly Detection:

- Anomaly detection is well-suited for scenarios where the class imbalance is extreme, and oversampling or undersampling may not be practical.
- It can uncover rare events, fraud, or abnormal behavior that may be critical in applications like cybersecurity, fraud detection, network monitoring, and quality control.
- Anomaly detection methods often generalize well to new, unseen anomalies, as they are not tailored to specific imbalanced datasets.

Considerations:

- Proper preprocessing and feature engineering are essential for effective anomaly detection, as the quality of features can significantly impact the results.
- The choice of the appropriate anomaly detection method depends on the specific problem, the nature of the data, and the characteristics of the anomalies.
- Anomaly detection methods might require parameter tuning and careful evaluation to achieve the desired balance between false positives and false negatives.

Anomaly detection is a powerful approach for handling data imbalance when one class is an extremely rare occurrence.

It can provide valuable insights into rare events and abnormal behavior without the need for balancing the dataset, making it a valuable tool in various real-world applications.

5. Cost-Sensitive Learning In Handling Data Imbalance

Cost-sensitive learning is a technique used to address data imbalance in machine learning by assigning different misclassification costs to different classes.

It aims to account for the unequal costs associated with making errors in imbalanced datasets, where the consequences of misclassifying instances from different classes may vary significantly.

Cost-sensitive learning helps models prioritize the minority class, which is often of greater interest, and minimize the impact of misclassifying those instances.

Cost-Sensitive Learning Techniques

1. **Cost Matrices:** Cost-sensitive learning often involves defining a cost matrix, where each element represents the cost of misclassifying one class as another. This matrix is used to adjust the loss function during model training.
2. **Cost-Sensitive Algorithms:** Some machine learning algorithms and libraries provide built-in support for cost-sensitive learning. They allow you to directly specify the misclassification costs during model training. Examples include cost-sensitive decision trees and cost-sensitive support vector machines.
3. **Customized Loss Functions:** In some cases, you can define custom loss functions that incorporate the misclassification costs. These loss functions penalize errors differently for each class, making the model more sensitive to the minority class.

When to Use Cost-Sensitive Learning:

Cost-sensitive learning is valuable when the consequences of misclassification are asymmetric and vary across classes.

It is particularly suitable for imbalanced datasets where one class is rare, and the cost of missing instances from that class is high.

It can be applied to various domains, including medical diagnosis, fraud detection, and quality control, where the impact of errors is not uniform across classes.

Cost-sensitive learning helps create models that are more practical and cost-effective in addressing real-world problems with data imbalance.

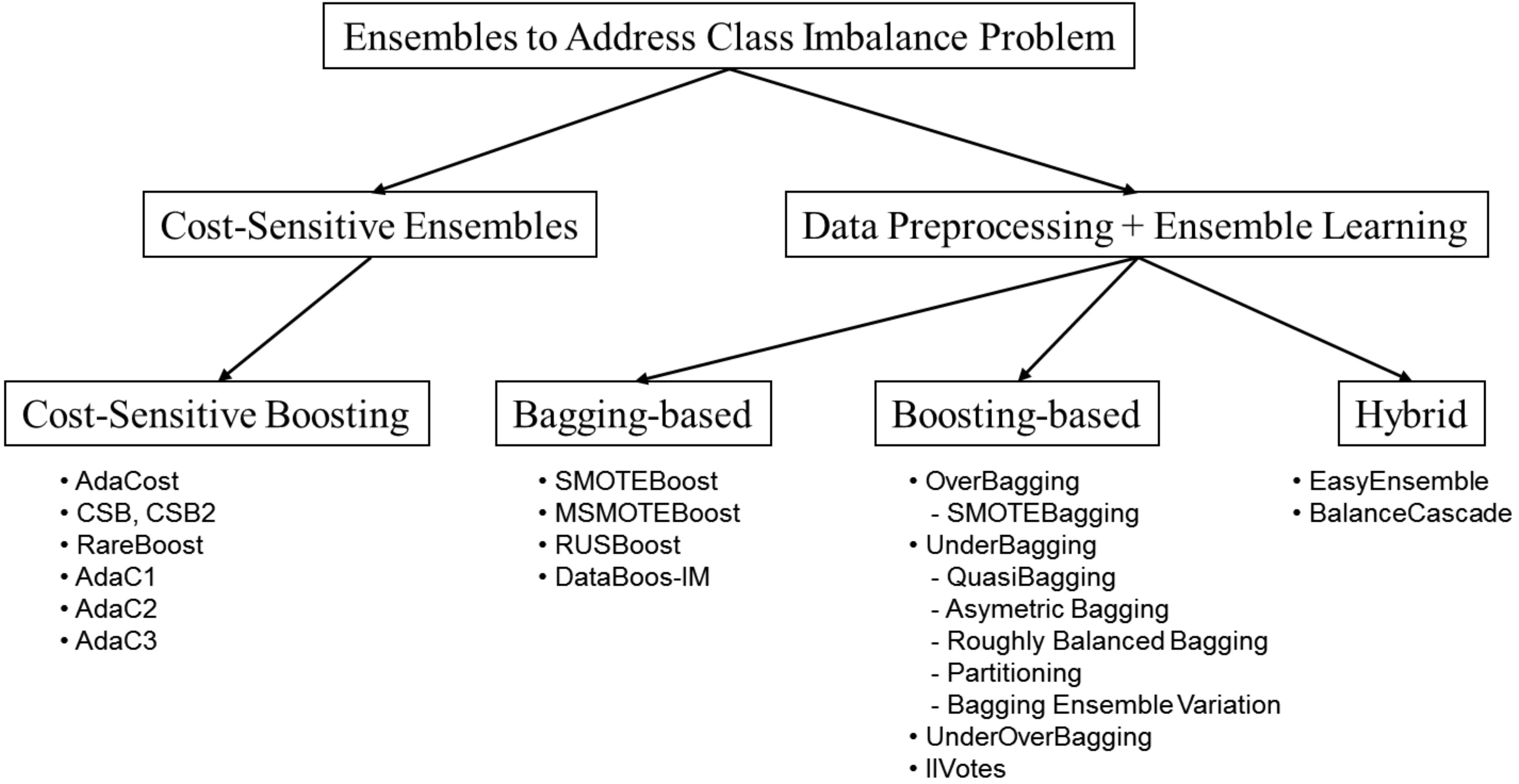
Ensemble-Based Methods- Galar et al. (2012)

Ensemble-based methods are based on a combination between ensemble learning algorithms and one of the previously discussed techniques, namely data and algorithmic approaches, or cost-sensitive learning solutions.

In the case of adding a data level approach to the ensemble learning algorithm, the new hybrid method usually preprocess the data before training each classifier.

On the other hand, cost-sensitive ensembles, instead of modifying the base classifier in order to accept costs in the learning process, guide the cost minimization procedure via the ensemble learning algorithm.

In this way, the modification of the base learner is avoided, but the major drawback, which is the costs definition, is still present.


Ensembles to Address Class Imbalance Problem
Cost-Sensitive Ensembles
Data Preprocessing + Ensemble Learning
Cost-Sensitive Boosting
Bagging-based
Boosting-based
Hybrid
• AdaCost
• CSB, CSB2
• RareBoost
• AdaC1
• AdaC2
• AdaC3
• SMOTEBoost
• MSMOTEBoost
• RUSBoost
• DataBoos-IM
• OverBagging
- SMOTEBagging
• UnderBagging
- QuasiBagging
- Asymetric Bagging
- Roughly Balanced Bagging
- Partitioning
- Bagging Ensemble Variation
• UnderOverBagging
• IIVotes
• EasyEnsemble
• BalanceCascade

Conclusion

Handling data imbalance in machine learning involves addressing skewed class distributions, where one class significantly outnumbers others.

Imbalanced datasets can lead to biased model predictions and poor performance on minority classes.

Various techniques are employed, including resampling (oversampling and undersampling), algorithm-level adjustments, cost-sensitive learning, and anomaly detection.

Resampling methods aim to balance class proportions, while algorithm-level techniques and cost-sensitive learning modify algorithms to consider class imbalance. Ensemble methods combine predictions from several weak base learners,

Anomaly detection treats the minority class as anomalies.

Proper evaluation metrics like precision, recall, and F1-score are essential to assess model performance accurately,

Making the choice of technique is critical for effective handling of data imbalance.

Q
&
A