Global Journal of Mathematics and Statistics
ISSN: 3078-8927
DOI: 10.61424/gjms
Journal Homepage: www.bluemarkpublishers.com/index.php/GJMS

GJMS
BLUEMARK PUBLISHERS

| RESEARCH ARTICLE

A Performance Comparison of Machine Learning Models for Robotic Navigation Using Imbalanced and SMOTE-Enhanced Data

Samiul Islam[1] ✉ Md. Mynuddin[2], Sharmin Sultana[3], Somaresh Kumar Mondal[4], Md. Abul Hossain[5], Gowranga Kumar Paul[6], Md. Ayub Ali[7] and Md. Ayub Ali[8]

[12456]Department of Statistics, Mawlana Bhashani Science and Technology University, Santosh, Tangail-1902, Bangladesh

[38]Department of Statistics, University of Rajshahi, Rajshahi-6205, Bangladesh

[7]Department of Statistics, University of Barisal, Kornokathi, Barishal-8254, Bangladesh

Corresponding Author: Samiul Islam, **E-mail**: samiul.islam@mbstu.ac.bd

| ABSTRACT

Robotic navigation systems rely heavily on accurate classification of sensor inputs to make real-time movement decisions. However, class imbalance—where certain navigation commands are underrepresented—can severely degrade the performance of machine learning models, particularly in safety-critical scenarios. This study presents a comprehensive comparative analysis of six supervised learning classifiers—Decision Tree (DT), Random Forest (RF), k-Nearest Neighbors (KNN), Ranger, Support Vector Machine (SVM), and XGBoost—applied to the UCI Wall-Following Robot Navigation dataset. The objective is to evaluate classifier performance across both the original imbalanced dataset and a SMOTE-balanced version, and to understand the impact of data balancing on classification accuracy. A range of performance metrics was employed, including Accuracy, F1 Score, Precision, Recall, Specificity and Area Under the Curve (AUC). Results show that ensemble-based classifiers, particularly XGBoost and Ranger, significantly outperform traditional models under both data conditions, achieving near-perfect performance with F1-scores above 0.995 and AUC values of 1.000. In contrast, KNN and SVM showed limited robustness to class imbalance, with substantial drops in F1 and AUC when exposed to synthetic samples generated by SMOTE. The study highlights that while SMOTE improves minority class recall across most models, it may introduce noise and overlapping classes, especially detrimental to distance- and margin-based classifiers. Overall, the findings emphasize the effectiveness and resilience of ensemble models in managing imbalanced, multi-class robotics data. The study concludes that both classifier selection and data preprocessing are critical for developing reliable, real-time robot navigation systems. Recommendations include using XGBoost or Ranger in such applications and applying SMOTE selectively based on model characteristics.

| KEYWORDS

Machine Learning Classifiers, UCI Robotics Dataset, Class Imbalance, SMOTE Technique for Data Balancing, Supervised Learning, Ensemble Learning, Multiclass Classification, Classifier Performance Metrics

| ARTICLE INFORMATION

ACCEPTED: 11 April 2025 **PUBLISHED:** 25 June 2025 **DOI:** 10.61424/gjms.v2.i1.308

1. Introduction

In the realm of autonomous systems, robotic navigation is an essential capability that enables robots to navigate and engage with their surroundings intelligently and securely. Nowadays, fully autonomous mobile robots are

increasingly used in many fields for complicated tasks like navigating and interacting with their surroundings (Hammad et al., 2019). Robotic navigation is the mechanism by which a robot autonomously ascertains its location within an environment, senses surrounding features, and strategies and executes movement towards a designated objective while avoiding barriers and adapting to changing circumstances (Bongard, 2008). Navigation is an important part of controlling a mobile robot which includes unique actions like wall-following, when the robot stays a safe distance away from a wall or object while moving along it and quite hard to automatically manage and guide a mobile robot, especially when it has to cope with unpredictable situations and settings that change quickly (Y. L. Chen et al., 2013). To drive a robot well, it is necessary to make the proper choices about altering course based on data from sensors on board (Grandini et al., 2020).

Advancements in artificial intelligence (AI) and machine learning (ML) allow robots to comprehend their environment and make real-time choices based on sensory input (Chen et al., 2015). ML has emerged as a powerful approach for developing intelligent controllers for mobile robots, allowing them to learn complex control strategies directly from data (Y. L. Chen et al., 2013). Consequently, Machine learning (ML) has become a significant technique for improving navigation performance as artificial intelligence (AI) has advanced quickly because ML allows robots to learn complicated patterns from sensory input and make judgements in real time in changing situations (Alotaibi et al., 2024). Challenges in pattern recognition may be used to describe robot navigation tasks, such as wall-following (van Engelen & Hoos, 2020). In this model, sensor data is used as input, and the required robot movement or direction is shown as output. Robotic systems' localization, route planning, and obstacle avoidance skills have been greatly enhanced by supervised learning approaches in machine learning (Kiran et al., 2022). According to Hammad et al., (2019), ML models are specifically employed to predict the appropriate direction for a wall-following robot based on its sensor inputs. One of the primary issues in robotic navigation is addressing unbalanced datasets, where some navigation states (e.g., obstacle vs non-obstacle) are markedly under-represented.

However, for ML-based robotic navigation, class imbalance in the collected training data is a big problem. Navigation datasets are essential resources comprising sensor data and environmental annotations that facilitate the development and evaluation of autonomous systems for tasks such as path planning, obstacle avoidance, and environment mapping, and typically include multimodal inputs such as RGB images, depth maps, LiDAR scans, inertial measurements, and ground truth positional information (Cadena et al., 2016). These datasets enable ML models and robotic agents to perceive and interpret complex environments, improving navigation performance in indoor and outdoor scenarios (Geiger et al., 2012). Navigation datasets are typically biased towards the majority class, which usually means safe and normal movement, and the contrary, minority classes, such as unexpected stops, collisions, or edge-case manoeuvres, are not well represented (Buda et al., 2018; Japkowicz & Stephen, 2002). This mismatch might lead to biased models that do not do as well at making safety-critical predictions, which would significantly impact navigation safety in the real world. Several data-level resampling techniques have been investigated as potential solutions to this problem; among them, SMOTE (Synthetic Minority Over-sampling Technique) stands out. By analyzing each minority class sample and inserting synthetic instances along the line segments that connect any or all of the k closest neighbor's in that class, SMOTE functions in feature space instead of data space (Chawla et al., 2002). To achieve this goal of a fairer model training and better class balance, it employs a k-nearest neighbor's strategy to generate synthetic instances for the minority class. Subsequent studies have demonstrated that SMOTE can significantly improve classification performance, especially when combined with ensemble models or deep learning (Fernández et al., 2018; Haixiang et al., 2017). In robotics, the use of SMOTE-enhanced data can result in safer navigation systems by ensuring that rare but important classes, such as obstacle proximity or emergency braking, are accurately recognized and appropriately responded to.

Automatic mobile robots are becoming important in complicated contexts, including nuclear power plants, oil refineries, and search and rescue. Mission success and safety depend on these robots' strong and precise navigation (Hammad et al., 2019). Real-world surroundings are unstable and dynamic, making traditional navigation approaches difficult. Robots' awareness and knowledge of their environment might restrict their guiding (Oscar et al., 2019). Machine learning (ML) may help robots learn and adapt to these navigation issues by learning from experience or data. Machine learning improves computational systems' performance on a given job using data, which is useful when explicit programming for all situations is impractical. This method has worked in robotics,

where ML can create precise controllers (Hammad et al., 2019). Classification-based learning, a major machine learning field, has been effective in many areas and may be used for robot navigation (Y. L. Chen et al., 2013).

Mobile robotics tasks like wall-following navigation need a robot to stay a certain distance and orientation from a wall (Y. L. Chen et al., 2013). This is generally a pattern recognition problem: the system must decide the robot's control action or direction based on sensory information from its surroundings (Hammad et al., 2019). Several research have employed control and machine learning technologies to address this difficulty (Aguas et al., 2019). A data-driven fuzzy controller is used in a hexapod robot design, while others use probabilistic neural networks. Freire et al. examined 24 ultrasonic sensors on a robot and recorded wall-following direction, demonstrating the dataset's non-linear separability (Hammad et al., 2019). This effort produced an open-source dataset containing 24 sensor readings and robot orientation (Y. L. Chen et al., 2013).

A highly cited open-source wall-following robot navigation dataset is available from the UCI Machine Learning Repository. This dataset was obtained using a SCITOS G5 mobile robot for research and industry. The SCITOS G5 has 24 circular ultrasonic sensors around its waist, numbered clockwise from the front. For four rounds, the robot recorded sensor readings and robot orientation while following the walls of a confined chamber. The navigation assignment has four directions: forward, mild right-turn, sharp right-turn, and minor left-turn. The entire 24-sensor dataset and reduced forms with 4 and 2 aggregated sensor inputs per record are available (Y. L. Chen et al., 2013).

This dataset has helped several control model research endeavor's. These projects have used machine learning methods. MLP and Elman Recurrent Networks were suggested for the 24-sensor dataset (Y. L. Chen et al., 2013). This dataset has also been studied utilizing classification-based learning with Particle Swarm Optimization (PSO) for parameter adjustment. Fuzzy-PD controllers, adjusted by PSO and proven experimentally on a DaNI 2.0 robot utilizing ultrasonic sensors, are another wall-following control strategy that focusses on controller design rather than UCI dataset categorization (Somiya Rani et al., 2020). Comparative studies are necessary due to the variety of machine learning algorithms and the difficulty of choosing the best one. The "No Free Lunch Theorem" states that no algorithm solves all problems optimally. Thus, comparing algorithms on tasks like robot navigation reveals their strengths and flaws (Y. L. Chen et al., 2013).

In recent research, several machine learning and deep learning algorithms were compared to assess their efficacy for sensor fusion. This research analyzed many models, such as Decision Tree (DT), Gradient Boost Classifier (GBC), Random Forest Classifier (RFC), Linear Discriminant Analysis (LDA), Support Vector Machine (SVM), K-Nearest Neighbor's (KNN), and Gaussian Naïve Bayes (GNB). They also compared DFNN, GRU, and LSTM performance (Y. L. Chen et al., 2013). Automatic control and guidance were examined using machine learning methods on "wall-following robot navigation data" and classifiers, with performance measured by Precision, Absolute Mean Error (AME), Mean Squared Error (MSE), Relative Absolute Error (RAE), and Root Relative Squared Error (RRSE) (Ghandibidgoli et al., 2022). This UCI dataset has also been used to compare machine learning techniques for wall-following robot control (ALPAYDIN E, 2020). This research compared machine learning and deep learning models on this topic (Hammad et al., 2019).

This project used Decision Tree (DT), Random Forest Classifier (RFC/RF), K-Nearest Neighbors (KNN), Ranger, Support Vector Machine (SVM), and XGBoost, a suite of widely recognized and effective classifiers for pattern recognition problems like robot navigation command prediction (Ghandibidgoli et al., 2022). Support Vector Machines (SVM) are good at classifying and reducing errors. Random Forests (RF) are a popular machine learning approach, and Ranger provides a fast implementation for high-dimensional data. Classic classification techniques include Decision Trees (DT) and K-Nearest Neighbor (KNN). XGBoost, a gradient boosting technique, excels in machine learning applications like intrusion detection (ALPAYDIN E, 2020).

Effective classifier performance assessment in robot navigation requires a thorough set of measures. Accuracy is important, but F1 score, sensitivity (recall), specificity, precision, AUC, and Kappa are essential for class imbalance datasets. These measures provide a deeper knowledge of a classifier's performance across classes and capacity to distinguish positive and negative occurrences (Hammad et al., 2019). Evaluating classifier performance needs solid methods and metrics. Standard techniques examine generalization ability utilizing different training and test datasets. Monte-Carlo cross-validation ensures findings dependability (Y. L. Chen et al., 2013). Accuracy, Precision,

Recall (TP-rate), FP-rate, MSE, MAE, RAE, RRSE, Youden's index, positive likelihood, negative likelihood, DOR, and AUC ROC are used to quantify performance. Multiple measures provide a more full view of a classifier's efficacy than accuracy (Ghandibidgoli et al., 2022). Continued development and comparison of machine learning algorithms on this basic robotics navigation challenge is crucial. Exploring a variety of algorithms and thoroughly evaluating their performance using standardized criteria helps determine their applicability and limits in robot control situations (Hammad et al., 2019).

Previous studies have applied machine learning to the UCI Wall-Following dataset and compared some algorithms, but a detailed comparative analysis of DT, RF, KNN, SVM, Ranger, and XGBoost's strengths and weaknesses under balanced and imbalanced data is needed. Some literature assessments say present work may not adequately highlight ML algorithm strengths and weaknesses. Using a comprehensive suite of evaluation metrics, this project aims to determine whether these models are suitable and effective for predicting mobile robot navigation actions based on sensor data, addressing the task's and dataset's challenges (Choi et al., 2020).

The primary objective of this study is to conduct a comparative investigation of various machine learning algorithms applied to a robotic wall-following navigation task. This research focuses on evaluating the performance of several widely used machine learning classifiers in predicting navigation commands based on sensor input data. Specifically, the study examines algorithms such as Decision Tree (DT), Random Forest (RF), K-Nearest Neighbor (KNN), Ranger (a high-performance Random Forest variant), Support Vector Machine (SVM), and XGBoost. To ensure a robust and fair comparison, the performance of these models will be assessed using a comprehensive set of classification metrics, including accuracy, F1 score, sensitivity, specificity, precision, AUC, and Kappa. These metrics are especially important in contexts where class imbalance is present, which can otherwise skew model evaluation. Through this analysis, the research aims to offer meaningful insights into the comparative strengths and limitations of each algorithm in sensor-driven robotic navigation. The findings are intended to inform the development and selection of effective ML-based solutions for real-world autonomous navigation tasks.

Hence, the study addresses the following research questions:

RQ1: How does class imbalance affect the performance of machine learning classifiers in robotic navigation?

RQ2: In the presence of unbalanced data, which classifiers do better than those with balanced data?

RQ3: Does SMOTE significantly enhance classification performance, particularly for minority classes?

RQ4: Which models are most suitable for reliable robot navigation in real-time?

Subsequent research sections are structured as follows: Section 2 encompasses the materials and techniques, including the dataset description, machine learning algorithms, and a summary of algorithm assessment criteria. Section 3 establishes the findings from the study of the sample dataset. Finally, section 4 summarizes the concluding remarks and outlines directions for future research.

2. Materials and Methodology

2.1 Description of Data Set

The Wall-Following Robot Navigation Dataset was gathered from the SCITOS G5 mobile robot as it carried out wall following tasks in an indoor controlled scenario (Freire et al., 2009). The robot made four complete circuits around the room during this process, using the 24 ultrasonic sensors that encircle its midsection in a circular arrangement, numbered from the front moving clockwise (Chen et al. These sensors provided raw proximity information representing a snapshot of the robot's environment per time step. Each instance of data corresponds to a vector of 24 sensor readings as well as a navigational decision, that is, a Move-forward, Slight-right-turn, Sharp-right-turn, or Slight-left-turn movement command (Hammad et al., 2019). The dataset contains 5,455 annotated samples, and has the characteristic of an intrinsic class imbalance due to under-representation of certain navigational commands—most particularly those associated to sharp turns (Ghandibidgoli & Mokhtari, 2022). To support varied research and modeling needs, the dataset is available in different formats, including the full 24-sensor configuration and simplified versions using only 4 or 2 aggregated sensor inputs (Chen et al., 2013). This dataset is widely recognized and publicly available through the **UCI Machine Learning Repository**, making it a valuable benchmark for evaluating machine learning models in robotic perception and navigation tasks.

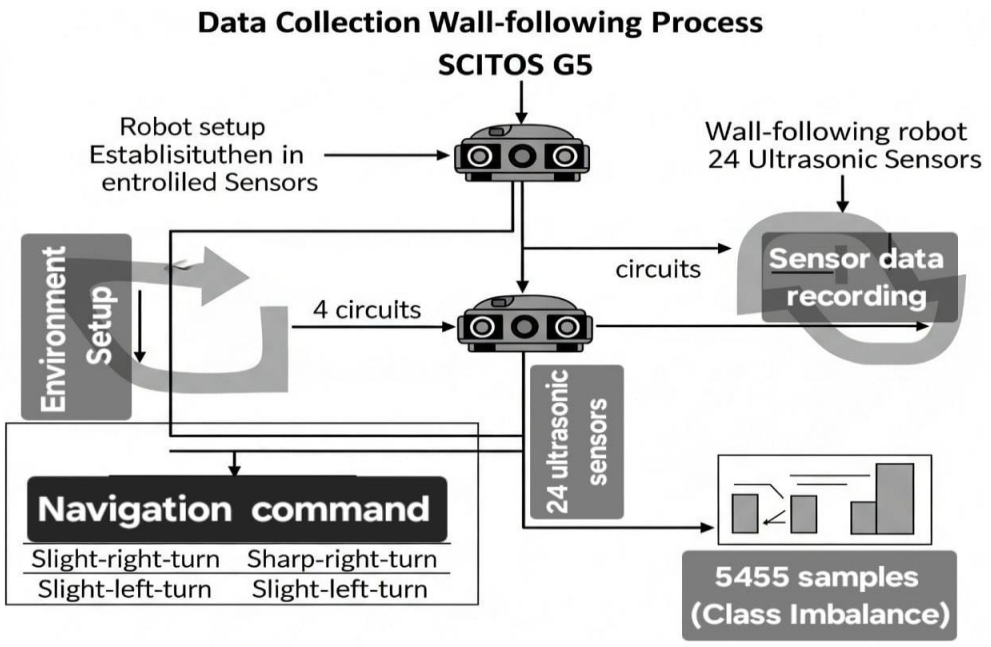

Figure 1: Wall-Following Robot Data Collection Steps

Dependent Variable:

In the dataset-3, the response variable was Class (Direction of Robot Movement). Class had four levels. They are: Move-Forward, Slight-Right-Turn, Sharp-Right-Turn, and Slight-Left-Turn

Independent Variables:

There were 24 Explanatory variables in dataset. They were:

- US1: ultrasound sensor at the front of the robot (reference angle:180°) - (numeric: real)
- US2: ultrasound reading (reference angle:-165°) - (numeric: real)
- US3: ultrasound reading (reference angle:-150°) - (numeric: real)
- US4: ultrasound reading (reference angle:-135°) - (numeric: real)
- US5: ultrasound reading (reference angle:-120°) - (numeric: real)
- US6: ultrasound reading (reference angle:-105°) - (numeric: real)
- US7: ultrasound reading (reference angle:-90°) - (numeric: real)
- US8: ultrasound reading (reference angle: -75°) - (numeric: real)
- US9: ultrasound reading (reference angle:-60°) - (numeric: real)
- US10: ultrasound reading (reference
- US13: reading of ultrasound sensor situated at the back of the robot (reference angle: 0°) - (numeric: real)
- US14: ultrasound reading (reference angle:15°) - (numeric: real)
- US15: ultrasound reading (reference angle:30°) - (numeric: real)
- US16: ultrasound reading (reference angle:45°) - (numeric: real)
- US17: ultrasound reading (reference angle:60°) - (numeric: real)
- US18: ultrasound reading (reference angle:75°) - (numeric: real)
- US19: ultrasound reading (reference angle:90°) - (numeric: real)
- US20: ultrasound reading (reference angle:105°) - (numeric: real)
- US21: ultrasound reading (reference angle:120°) - (numeric: real)
- US22: ultrasound reading (reference angle:135°) -

angle:-45°) - (numeric: real)

- US11: ultrasound reading (reference angle:-30°) - (numeric: real)
- US12: ultrasound reading (reference angle:-15°) - (numeric: real)

(numeric: real)

- US23: ultrasound reading (reference angle:150°) - (numeric: real)
- US24: ultrasound reading (reference angle: 165°) - (numeric: real)

2.2 Data Pre-processing

Data preprocessing is an important step in the classification framework that guarantees the high accuracy of the findings. The two tasks in this phase are data transformation and cleaning. Duplicate values were removed and outliers were truncated. There is no missing value in the data. To prepare the dataset for classification, several preprocessing steps were applied which are shown in the following figure-

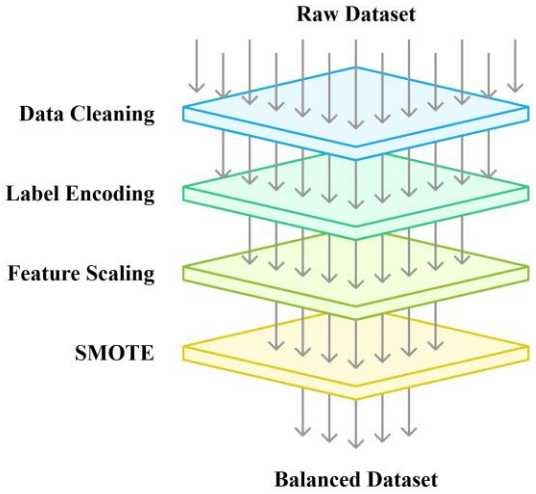

Figure 2: Data Pre-processing

2.3 Feature Selection

Feature selection is a critical step in machine learning that involves identifying the most relevant features from a larger set to improve model performance, reduce computational complexity, and minimize the risk of overfitting. This process enhances both the efficiency and interpretability of predictive models. Various approaches are employed for feature selection, including filters, wrappers, and embedded techniques, which evaluate feature importance based on statistical relevance, model performance, or their role during the training phase. This study employed three feature selection methods: the Boruta algorithm and regularized random forest.

Boruta algorithm

Boruta, developed by Kursa and Rudnicki in 2010, is a powerful feature selection method based on random forests. It is designed to fairly evaluate all features by comparing their importance to that of randomly shuffled versions (shadow features), helping to identify both strongly and weakly relevant variables. One of Boruta's key advantages is its ability to adapt to changes in feature importance and account for interactions between features. It is an easy-to-use R package, making it accessible for researchers and practitioners looking for a reliable and interpretable feature selection approach (Kursa & Rudnicki, 2010).

Regularized Random Forest

The regularized random forest provides a method for simplifying feature selection in random forests. Instead of traditional feature selection, this method decreases regression coefficients towards zero. Regularized random forests utilize tree models, making them beneficial in scenarios where the number of predictors significantly exceeds the amount of data. The R package named RRF4 implements the regularized random forest algorithm. Improved under the title Guided Regularized Random Forest (GRRF), this version selects features based on

relevance ratings sourced from a standard random forest. GRRF is noted for its computational efficiency, as it can identify small subsets of features while maintaining comparable accuracy to other methods. Regularization is less commonly applied in these techniques since decision trees and random forests inherently incorporate built-in regularization (Izquierdo-Verdiguier & Zurita-Milla, 2020).

2.4 Machine Learning Models and Evaluation

This study used both traditional supervised machine learning algorithms and ensemble learning methods to model and test robotic wall-following navigation tasks. We used supervised learning, which uses labeled datasets, to train models to guess navigation choices based on data from sensors (ALPAYDIN E, 2020). We used algorithms like Decision Trees, Support Vector Machines (SVM), Logistic Regression, k-Nearest Neighbors (KNN), and Ranger to solve classification problems (Kotsiantis SB et al., 2007; Wright & Ziegler, 2017). We also used ensemble learning methods like Bagging (Random Forest) and Boosting (XGBoost) to make predictions more accurate and reliable (Breiman, 2001; T. Chen & Guestrin, 2016). These ensemble methods use more than one model to lower bias and variance, which improves performance in the end. We used a number of performance metrics to evaluate the models, including Accuracy, F1 Score, Precision, Recall (Sensitivity), Specificity, and AUC (Area Under the Curve) (Dietterich, 2000; Powers & Ailab, 2020). These metrics helped us get a full picture of how well the model worked, especially when there was class imbalance. Using graphs like ROC and AUC curves to compare algorithms helped find the best ones for making robotic navigation work better (Ž Vujović, 2021).

Table 1: Description of Different ML Algorithms

Algorithm	Type	Description
Decision Tree (DT) (Quinlan, 1986; Safavian & Landgrebe, 1991)	**Supervised**	• A non-parametric algorithm that splits the dataset into subsets using feature values and constructs a tree-like model of decisions. • Simple to understand and visualize, handles both numerical and categorical data, requires little data preprocessing. • Prone to overfitting, especially on small datasets; small changes in data can drastically change the structure of the tree.
Random Forest (RF) (Breiman, 2001)	**Ensemble Learning**	• An ensemble of decision trees built on random subsets of features and samples. Uses majority voting for classification. • Reduces overfitting compared to DT, works well with high-dimensional data, robust to noise and outliers, handles missing values. • Computationally expensive; less interpretable than a single decision tree.
Ranger (Wright & Ziegler, 2017)	**Ensemble Learning**	• A high-speed C++ implementation of Random Forests optimized for high-dimensional datasets, often used in biomedical and ecological applications. • Highly scalable, supports classification, regression, and survival analysis, allows multithreading and fast execution. • Limited interpretability, and less widely supported in standard ML platforms like Scikit-learn.
K-Nearest Neighbors (KNN) (Altman, 1992; Cover & Hart, 1967)	**Supervised**	• Classifies a data point based on the majority class of its k nearest neighbors using a distance metric (e.g., Euclidean distance). • Intuitive, no model training required, effective with well-separated data, supports multi-class classification. • Slow during prediction on large datasets, sensitive to irrelevant features and the curse of dimensionality, requires feature scaling.

Support Vector Machine (SVM) (Cortes et al., 1995)	**Supervised**	• Constructs an optimal hyperplane that maximizes the margin between data points of different classes. Can use kernel functions for non-linear boundaries. • Works well with high-dimensional data, effective when the margin between classes is clear, robust to overfitting in many cases. • Requires careful parameter tuning (C, kernel, gamma), high memory consumption, difficult to scale to large datasets.
XGBoost (T. Chen & Guestrin, 2016)	**Ensemble Learning**	• Gradient boosting framework that builds additive models in a forward stage-wise fashion; focuses on optimization and regularization. • High prediction accuracy, handles missing data, supports regularization (L1 & L2), scalable to large datasets. • Requires careful hyperparameter tuning, less interpretable than simple models, and computationally intensive.

2.5 Model Assessment

Assessing a machine learning model entails evaluating its performance to determine how well it achieves its goals. Once the training phase is complete, it is crucial to analyze the model using various metrics and methods to verify that it generalizes effectively to new data and meets the application's specific needs (Kohavi Ron, 1995). The evaluation approach depends on the problem type and may include metrics like Confusion Matrix (Table 2) with macro method for multi-level(Grandini et al., 2020). and thorough, rigorous cross-validation techniques, alongside other techniques such as the ROC curve.

Table 2: Confusion Matrix Table

		Predictive Class	
		Positive (P)	Negative (N)
Actual Class	Positive (P)	True Positive (TP)	False Negative (FN)
	Negative (N)	False Positive (FP)	True Negative (TN)

Table 3: Performance Evaluation Metrics for Classification Models

Metric	Formula	Description
Accuracy (Kohavi Ron, 1995; Powers & Ailab, 2020)	For two class $\frac{\text{TP+TN}}{\text{TP+TN+FN+FP}}$ For Multi-Class $\frac{\sum_{i=1}^{n} TPi}{\sum_{i=1}^{n}(TPi+FPi)}$	Measures the proportion of correctly classified instances among the total instances.
Sensitivity (Recall or TPR) (Sokolova & Lapalme, 2009)	$\frac{TP}{TP+FN}$	Measures the model's ability to identify positive instances correctly for each level. In case of four level, we get four sensitivity score then we average the total four score.
Specificity (TNR) (Fawcett, 2006; Sokolova & Lapalme, 2009)	$\frac{\text{TN}}{\text{TN}+\text{FP}}$	Measures the model's ability to identify negative instances correctly. In case of four level, we get four specificity score then we average the total four scores.
Precision (PPV) (Powers & Ailab, 2020; Sokolova & Lapalme, 2009)	$\frac{\text{TP}}{\text{TP}+\text{FP}}$	Measures the proportion of predicted positive cases that are actually positive. In case of four level, we get four precision score then we average the total four score.
F1 Score (Sokolova & Lapalme, 2009)	$\frac{2 * \text{Recall} * \text{Precision}}{\text{Recall} + \text{Precision}}$	Harmonic mean of precision and recall; useful for imbalanced class distributions.
ROC Curve and AUC Score (Grandini et al., 2020)	$\frac{\text{Area under Curve}}{\text{Total Area}}$	This is a two-dimensional plot of sensitivity and Specificity. The multilevel ROC AUC assesses model performance by computing class-specific AUCs, then averaging. Macro-average averages individual class AUCs, while micro-average aggregates all classes' TP/FP rates into one ROC curve

To compute these metrics, we obtain the predicted labels (or scores) for a set of test data. We then use this information to identify the true positives, true negatives, false positives, and false negatives of the model. With these values, we can calculate sensitivity, specificity, precision, recall, and the F1 score. Notably, these metrics rely on the classification threshold and help assess the trade-off between false positives and false negatives. Altogether, these metrics thoroughly evaluate a classification model's performance, focusing on correctness, completeness, and precision.

3. Results of the Analysis of Example Data Set

This section presents the analysis results of the example dataset discussed in the previous section. R programming language (R-version 4.2.2) with RStudio was used for data preprocessing, frequency distribution, association, and was applied for 14 machine learning algorithms.

3.1 *Frequency Distribution and Association Analysis*

The following figure-3 illustrates the class distribution in the robot navigation dataset used for training and evaluation. The dataset consists of four distinct navigational commands: *Move-Forward, Sharp-Right-Turn, Slight-Right-Turn,* and *Slight-Left-Turn*. Move-Forward and Sharp-Right-Turn dominate the dataset with 2205 (40.4%) and 2097 (38.4%) instances, respectively. Slight-Right-Turn and Slight-Left-Turn are significantly underrepresented, accounting for only 15.1% and 6.0% of the total data. This imbalance highlights the need for data balancing techniques like SMOTE, as used in the study, to ensure minority classes are adequately learned by classifiers. Without addressing this imbalance, models tend to bias toward majority classes, reducing recall and precision for the rarer yet critical commands in robotic navigation.

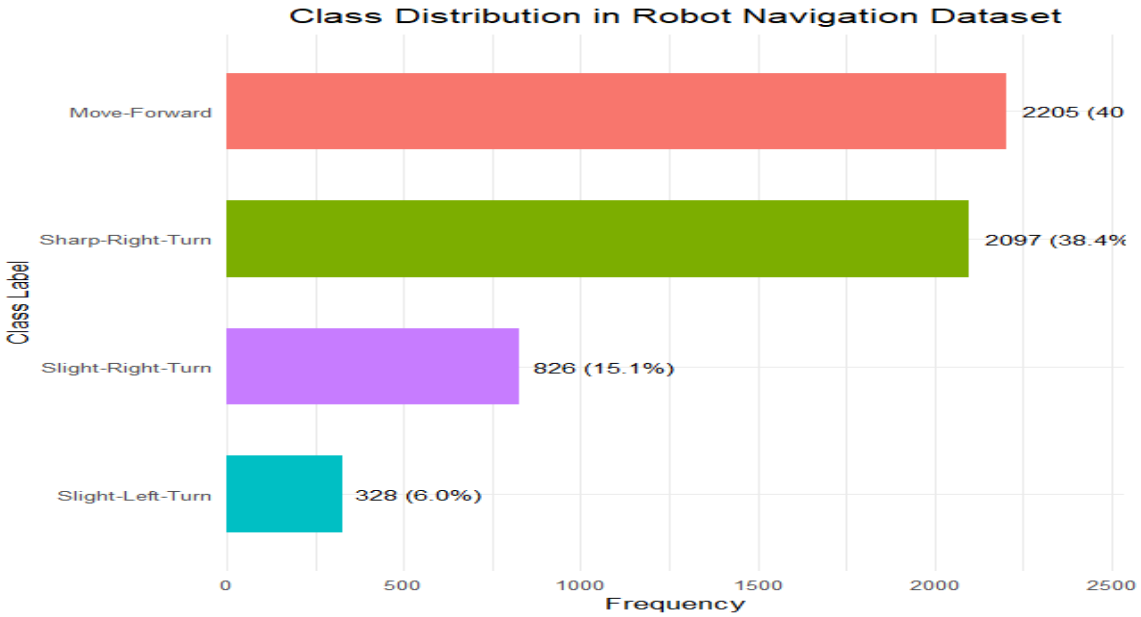

Figure 3: Class Distribution in Robot Navigation Dataset

The figure 4 illustrates the impact of SMOTE on class distribution. Initially, the dataset was highly imbalanced, with the "Slight-Left-Turn" and "Slight-Right-Turn" classes under-represented (shown by red bars). After applying SMOTE, all classes were balanced to approximately 1800 samples each (green bars), ensuring equal representation. This preprocessing step was essential to reduce model bias and enable fair performance comparison across classifiers, aligning with the project's goal of evaluating models under both imbalanced and balanced conditions.

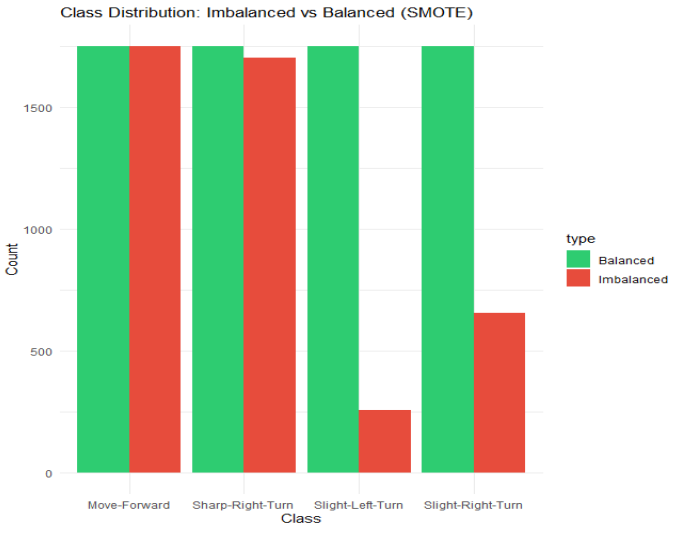

Figure 4: Class Distribution Before and After SMOTE Balancing

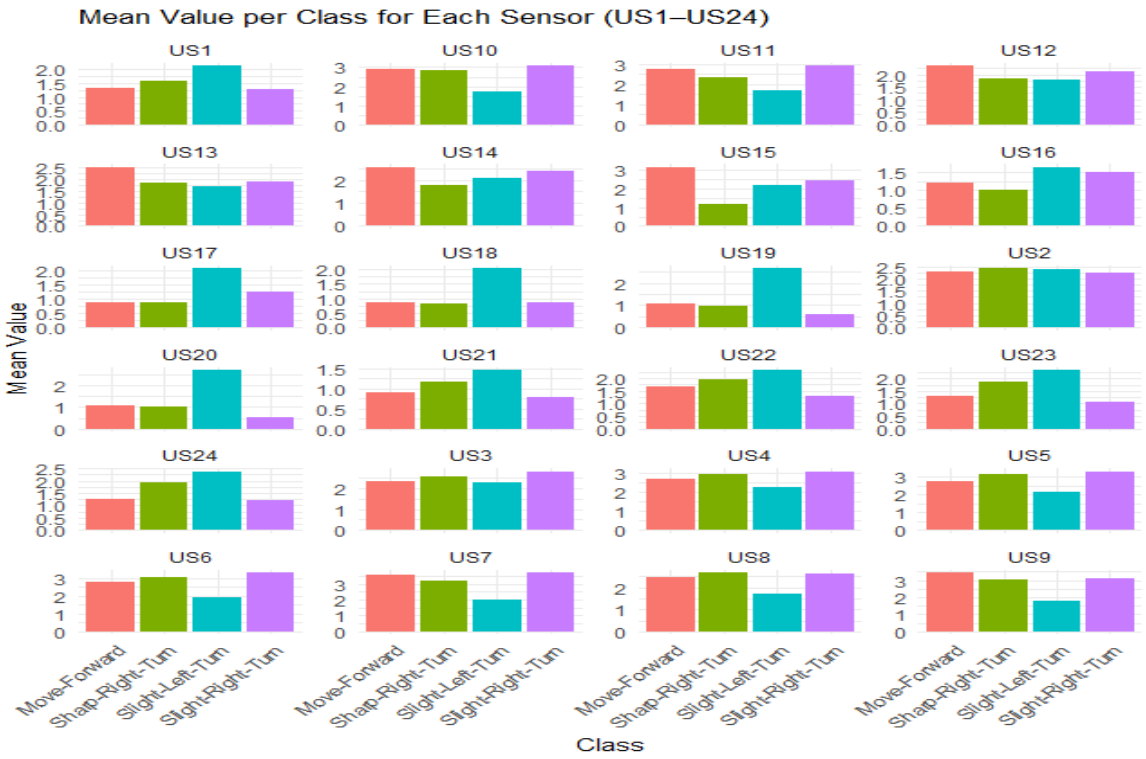

Figure 5: Mean Value per Class (Bar Chart) for each Sensor

Figure-5 illustrates distinct class-specific patterns across various sensors, revealing important insights for classification. Several sensors—such as US17, US18, and US19—exhibit noticeable differences in mean values between movement classes, with the *Sharp-Right-Turn* class (depicted in light blue) showing particularly high readings. These sensors are likely to be highly informative features for classification models. Additionally, sensors like US1, US2, and US24 demonstrate good separation in mean values across all four classes, which enhances class discriminability—especially beneficial for tree-based models like Decision Trees, Random Forests, and XGBoost that exploit such feature splits effectively. In contrast, sensors such as US12, US13, and US14 show very similar mean values across all classes, indicating low discriminative power. These may be considered redundant or less informative, and applying feature selection techniques could reduce noise for models that are sensitive to irrelevant features, such as SVM and k-NN. Furthermore, the figure underscores the potential impact of class imbalance. Sensors like US6 and US7, which show only subtle differences between classes, may lead to misclassifications in less robust models like k-NN. To mitigate this, techniques such as SMOTE or class weighting may be necessary to enhance model performance under imbalanced conditions.

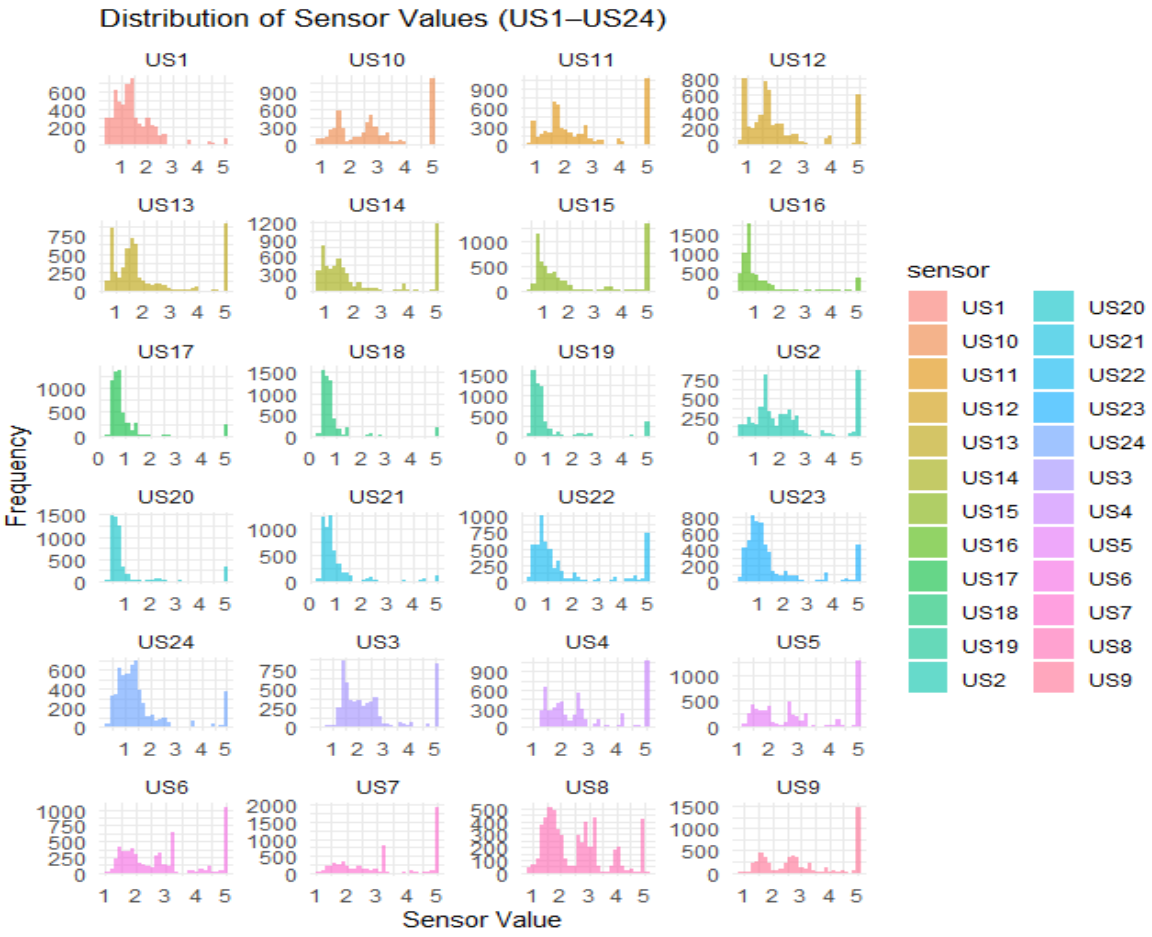

Figure 6: Distribution of Sensor Values

Figure 6 illustrates key distributional characteristics of sensor readings that have important implications for model performance and feature engineering. Many sensors, particularly US15–US21, exhibit strong right-skewed distributions with values concentrated near zero. This skewness can impact the performance of models like k-NN and SVM, which are sensitive to scale and distribution, making transformations such as log-scaling potentially beneficial. In contrast, sensors like US4, US5, US7, and US8 show multimodal distributions, suggesting they capture multiple distinct environmental patterns. These multimodal features are likely to be highly informative for models such as Random Forest, XGBoost, and Decision Trees, which can effectively leverage such complexity. Additionally, sensors like US4, US7, and US8 show high variance and span a broad value range (approximately 1 to 5), indicating their ability to reflect diverse environmental conditions—a valuable trait, particularly when dealing with imbalanced datasets where capturing variability is crucial. Finally, the figure highlights that certain features, such as US20 and US21, show high frequency in narrow value ranges, potentially dominating model input. This reinforces the need for appropriate feature scaling and outlier handling to ensure balanced and unbiased model training, especially in tree-based algorithms.

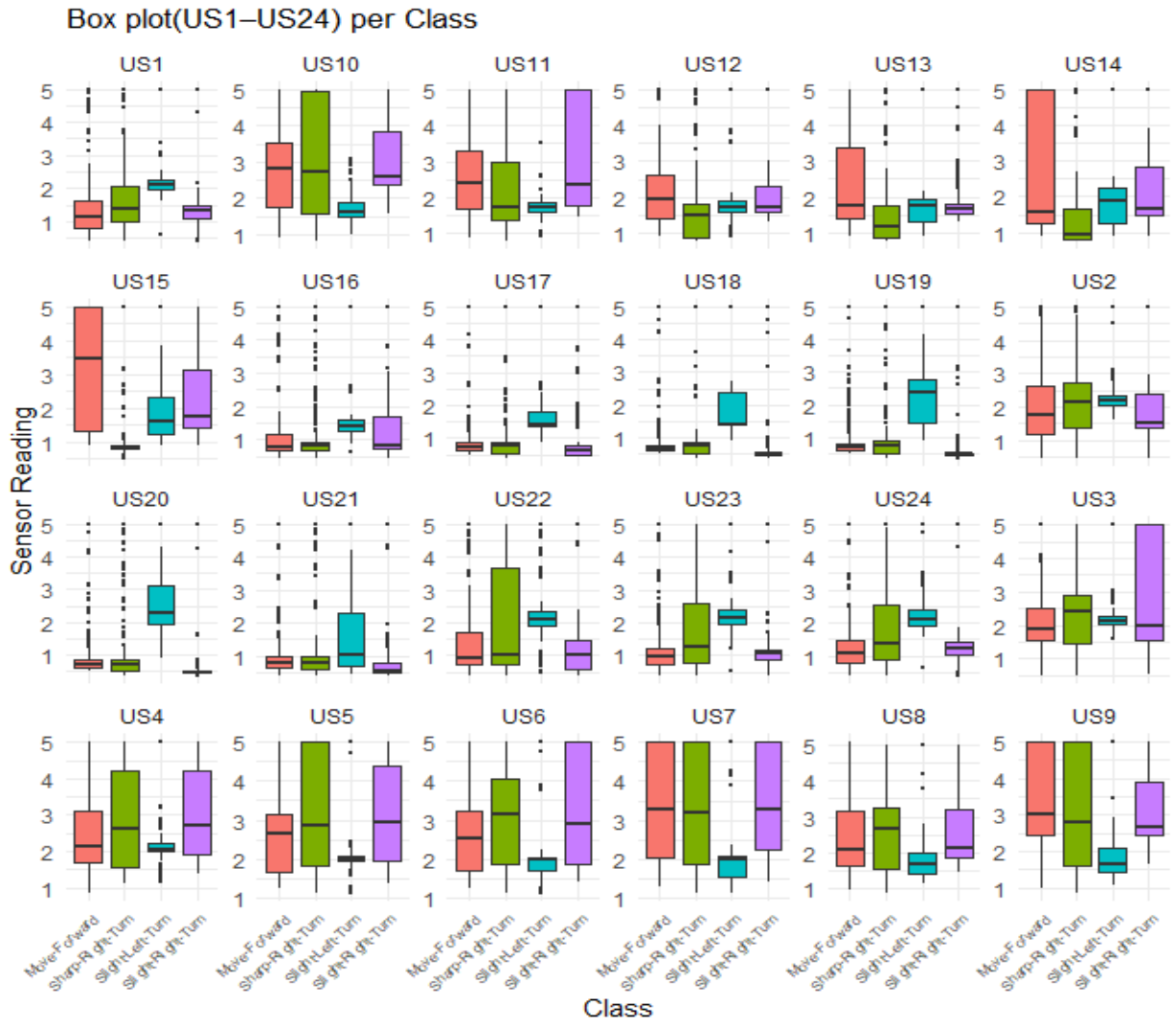

Figure 7: Box plot per Class

Figure 7 highlights distinctive sensor behaviors across movement classes based on median values and interquartile ranges (IQRs). Several sensors—such as US10, US11, US14, US22, US3, and US7—show strong class-wise separation in both median and IQR, indicating high discriminative power. These features are especially valuable for classification models like Random Forest and XGBoost, where feature importance rankings can guide model performance improvements. In contrast, sensors such as US19, US20, and US23 exhibit consistently narrow ranges across all classes, suggesting limited utility for class discrimination. Additionally, frequent outliers are observed across many sensors, reflecting rare or edge-case behaviors encountered during navigation. These insights emphasize the need for robust training strategies and potential outlier handling to reduce noise. Furthermore, sensors like US16, US17, and US21 demonstrate significant overlap in boxplot distributions across classes, implying lower classification value due to reduced class separability.

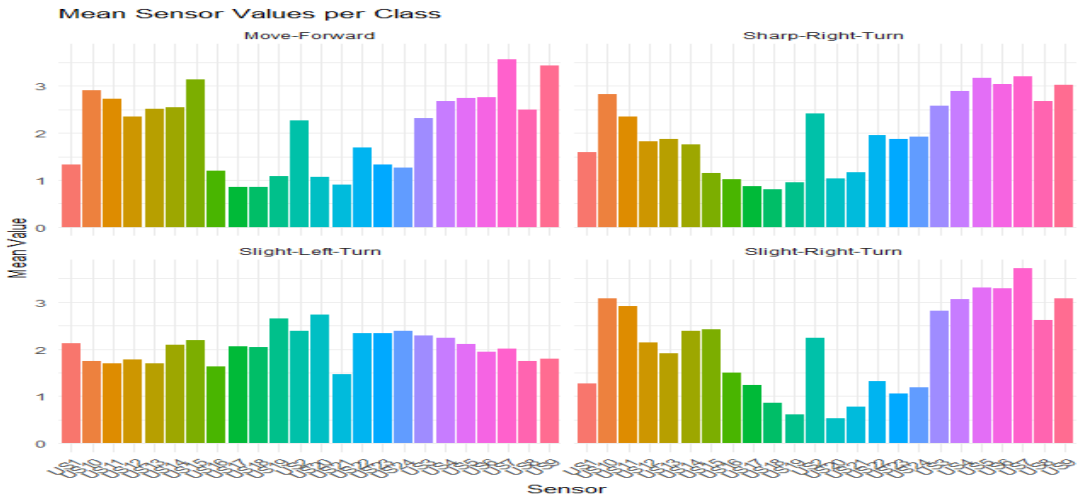

Figure 8: Mean Sensor Values per Class

The figure 8 illustrates distinct mean sensor profiles for each movement class, highlighting that the sensors effectively capture behavioral differences. For example, the *Move-Forward* class shows high readings on rear sensors (US22–US24), suggesting minimal obstruction behind. In contrast, the *Sharp-Right-Turn* class exhibits lower values on left-side sensors, likely due to proximity to walls. The *Slight-Left-Turn* and *Slight-Right-Turn* classes show elevated readings on the middle-right (US17–US21) and middle-left (US5–US9) sensors, respectively. These unique mean distributions across classes enhance class separability, which is particularly beneficial for models like Decision Trees, Random Forests, and XGBoost that leverage strong feature-class correlations. Sensors such as US2, US7, US8, US22, and US24 demonstrate significant variability across classes and are likely to be important features. Overall, the clear differentiation in sensor means supports the feasibility of multiclass classification, aligning well with the study's goal of comparing various model performances.

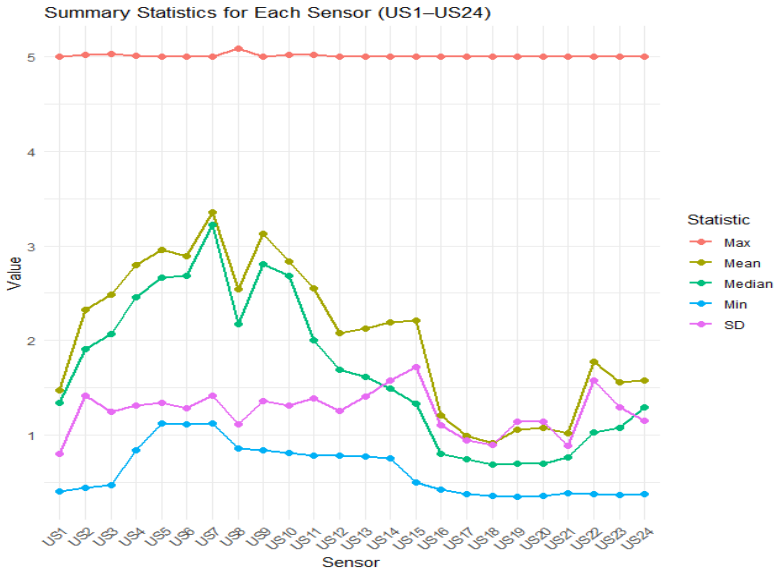

Figure 9: Summary Statistics Plot for each Sensor

	Sensor	Min	Max	Mean	Median	SD
1	US1	0.400	5.000	1.472	1.335	0.803
2	US2	0.437	5.025	2.327	1.905	1.410
3	US3	0.470	5.029	2.489	2.064	1.247
4	US4	0.833	5.017	2.797	2.458	1.309
5	US5	1.120	5.000	2.959	2.667	1.339
6	US6	1.114	5.005	2.893	2.683	1.283
7	US7	1.122	5.008	3.351	3.226	1.414
8	US8	0.859	5.087	2.540	2.172	1.112
9	US9	0.836	5.000	3.126	2.802	1.357
10	US10	0.810	5.022	2.832	2.679	1.308
11	US11	0.783	5.019	2.549	2.000	1.382
12	US12	0.778	5.000	2.078	1.689	1.249
13	US13	0.770	5.003	2.126	1.609	1.407
14	US14	0.756	5.000	2.190	1.493	1.577
15	US15	0.495	5.000	2.206	1.328	1.715
16	US16	0.424	5.000	1.202	0.803	1.099
17	US17	0.373	5.000	0.990	0.738	0.942
18	US18	0.354	5.000	0.910	0.685	0.890
19	US19	0.340	5.000	1.058	0.691	1.145
20	US20	0.355	5.000	1.076	0.693	1.141
21	US21	0.380	5.000	1.016	0.764	0.887
22	US22	0.370	5.000	1.778	1.030	1.572
23	US23	0.367	5.000	1.555	1.071	1.291
24	US24	0.377	5.000	1.579	1.289	1.150

Figure 10: Summary Statistics (in table) for each Sensor

From the figure 9 we see that Mean and median values vary significantly and the **standard deviation (SD)** is higher for sensors like **US6–US10 and US24**, implying they capture more dynamic variation. This variability in sensor behavior (especially SD and mean differences) suggests that **not all sensors contribute equally** to motion classification.

Figure 10 highlights patterns in sensor activity based on mean values and standard deviations. Sensors US7 (Mean: 3.35, SD: 1.414) and US6 (Mean: 2.89, SD: 1.283) stand out as the most active, exhibiting both high average readings and substantial variability. This suggests they are likely positioned in forward-facing directions, where environmental changes are more frequent. US15 shows the highest variability (SD: 1.715), indicating that it captures dynamic interactions with the surroundings. In contrast, sensors US18–US20 demonstrate the lowest mean values (ranging from 0.91 to 1.01) and low variability (SD < 1.2), suggesting they are monitoring more stable regions, possibly side-mounted or less informative areas. Additionally, the sensor value range shown in Figure 9 reveals that maximum readings across all sensors are close to 5, indicating consistent calibration or a capped measurement range. Minimum values vary from 0.34 (US19) to 1.12 (US6, US7), showing that some sensors frequently detect close obstacles—critical for accurate turning and navigation.

The correlation heat-map depicted the relationships between 24 variables (US1 to US24). Each cell in the map visualized the intensity of the correlation coefficient between the respective pairs of variables, ranging from -1 to 1. A positive correlation indicated a direct relationship, while a negative correlation suggests an inverse relationship. Notably, there are strong positive correlations between certain variables, such as US6 and US7, US8 and US9, and US10 and US11. These associations suggested potential interdependencies or shared trends between the corresponding aspects of the data we represented. Conversely, negative correlations were observed in pairs like US1 and US7, US1 and US8, and US2 and US11. These negative correlations may signify an opposing trend or a counteracting influence between the respective variables. Furthermore, some variables exhibited weak or near-zero correlations, indicating a lack of significant linear relationship. For instance, US2 and US21, US3 and US10, and US15 and US24 have correlation coefficients close to zero.

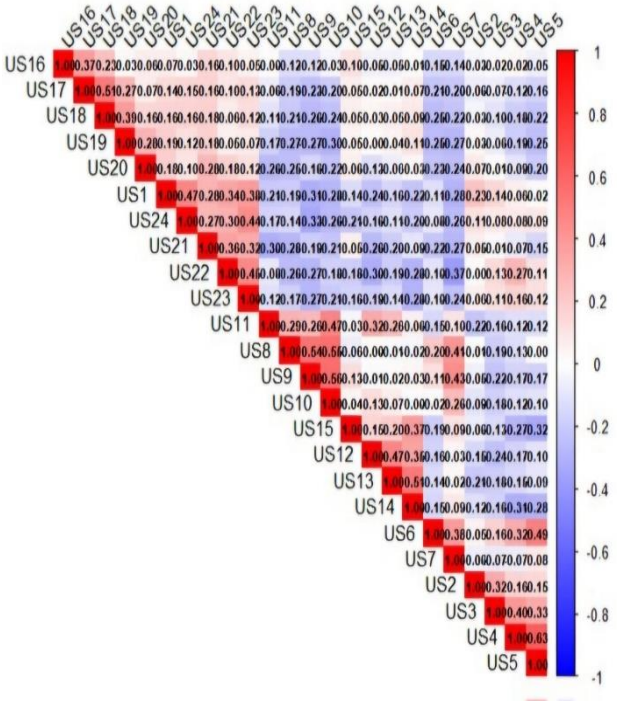

Figure 11: Correlation Matrix of Sensor Data

The following figure 12 illustrates the results of feature importance analysis conducted using Boruta, Regularized Random Forest, and step-wise multinomial logistic regression. The dataset comprised 24 explanatory variables, all of which demonstrated significant importance for modeling. Among them, sensors US15, US19, US20, US14, and US12 emerged as the top-ranked features, with US15 exhibiting the highest importance score (approximately 80), highlighting its strong role in differentiating between navigational classes. These sensors are likely critical for detecting spatial patterns during turning maneuvers or obstacle avoidance. In addition, sensors such as US13, US27, US11, and US24 were found to be moderately important, offering valuable supplementary information that can enhance model performance when used alongside the top-ranked features. Conversely, certain sensors—specifically US22, US1, and US7—as well as the shadow-based artificial features (shadowMin, shadowMean, shadowMax), were

deemed less important or were rejected, indicating their limited relevance or redundancy in improving predictive accuracy.

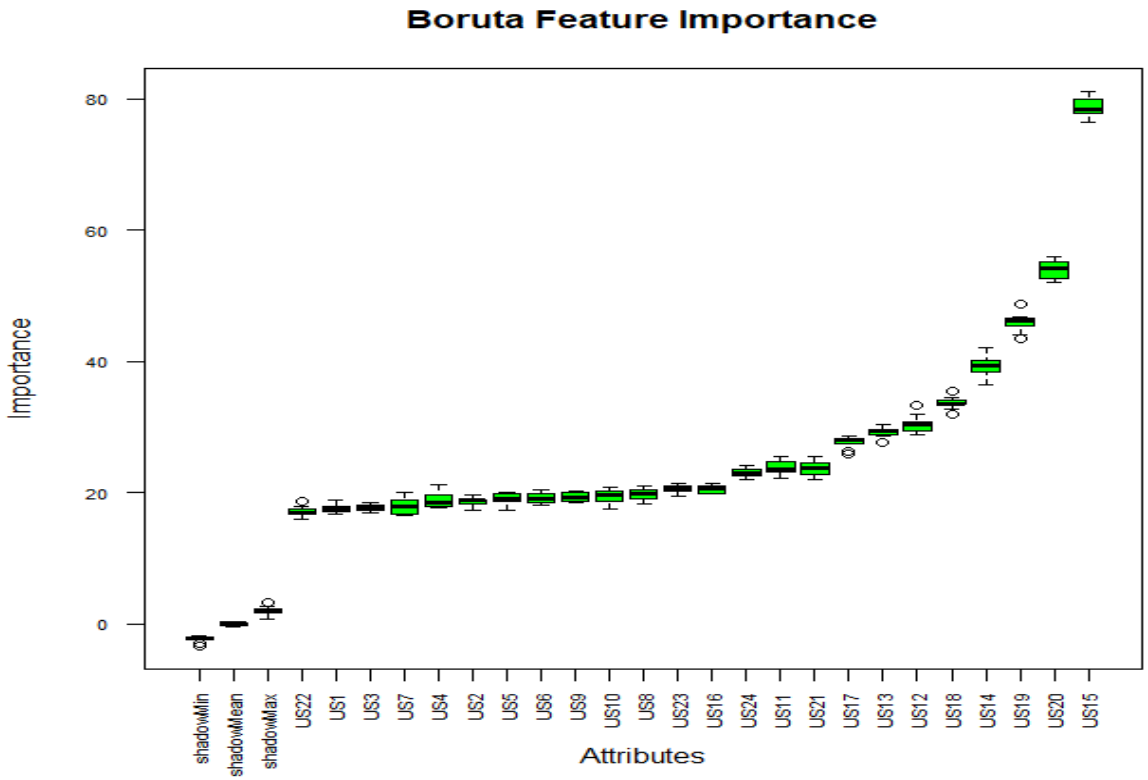

Figure 12: Boruta Feature Importance

3.2 Imbalance Data Analysis

This subsection details the performance of the selected machine learning classifiers when applied to the original, imbalanced Wall-Following Robot Navigation Data Set. As noted previously, this dataset, collected from a SCITOS G5 mobile robot using 24 ultrasound sensors, exhibits a non-uniform distribution across its four navigation classes: Move-forward, Slight-right-turn, Sharp-right-turn, and Slight-left-turn, with specific sample counts reflecting this imbalance.

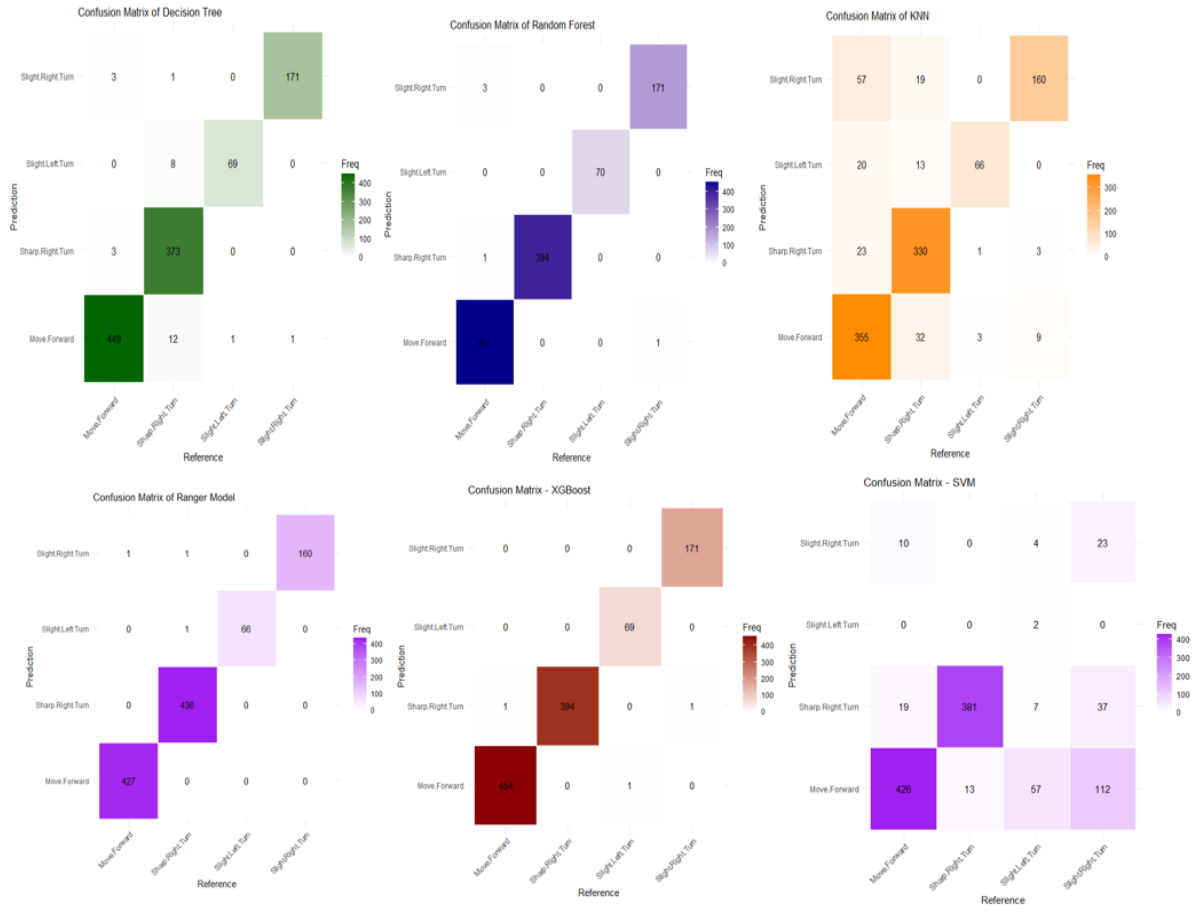

Figure 13: Confusion Matrices Comparing Classifier Performance on Movement Classification

Confusion matrices comparing the performance of multiple classifiers on the movement classification task are shown in Figure 13. The "Move.Forward" and "Sharp.Right.Turn" classes were correctly classified by the Decision Tree model with high true positives, but "Slight.Right.Turn" was noticeably misclassified, indicating some class overlap and sensor ambiguity. Random Forest performed almost flawlessly for three classes, but it had trouble with "Slight.Right.Turn," which was frequently mislabeled as "Move.Forward," suggesting a class imbalance that might need to be resampled or weighted. The K-Nearest Neighbours demonstrated sensitivity to overlapping features and imbalance by performing well on "Sharp.Right.Turn" and "Move.Forward," but they struggled to distinguish between "Slight.Right.Turn" and "Move.Forward." Despite unbalanced data, the Ranger model showed excellent generalisation with very high class-wise accuracy and few misclassifications, particularly in dominant classes like "Move.Forward" and "Sharp.Right.Turn." Due to feature overlap and class imbalance, the SVM classifier did well on the major classes but had trouble with "Slight.Left.Turn" and "Slight.Right.Turn." Last but not least, XGBoost demonstrated strong performance even in the face of data imbalance, nearly total diagonal dominance, and low misclassifications. All things considered, these matrices show that although models such as Random Forest, Ranger, and XGBoost attain high accuracy, it is still difficult to consistently distinguish between closely related turning classes, indicating the need for additional fine-tuning or data balancing.

Model	Accuracy	Precision	Recall	Specificity	F1_Score	AUC
Decision Tree	0.818	0.846	0.650	0.927	0.843	0.840
Random Forest	0.994	0.992	0.995	0.998	0.994	1.000
KNN	0.846	0.851	0.843	0.940	0.847	0.959
Ranger	0.994	0.994	0.991	0.998	0.992	1.000
SVM	0.880	0.881	0.868	0.954	0.874	0.980
XGBoost	0.994	0.992	0.992	0.998	0.992	1.000

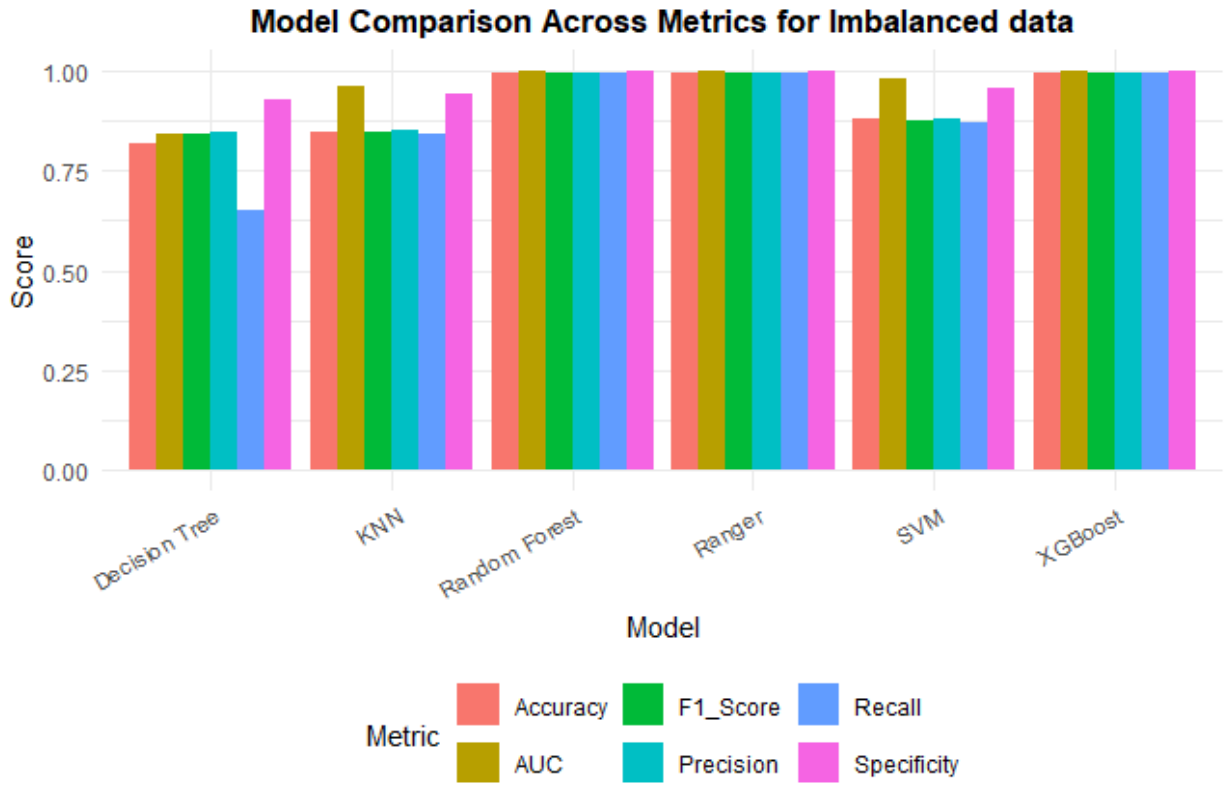

Figure 14: Comparative Evaluation of Classifier Performance Across Metrics

This figure 13 illustrates the overall performance of six classifiers based on key evaluation metrics: Accuracy, Precision, Recall, Specificity, F1-Score, and AUC. The top-performing models—Random Forest, Ranger, and XGBoost—achieved near-perfect scores across all metrics (AUC = 1.000), reflecting their superior ability to manage class imbalance and learn robust decision boundaries due to their ensemble architecture. SVM ranked as a moderate performer, showing strong specificity (0.954) and a high AUC (0.980), but a slightly lower recall (0.868), suggesting reduced sensitivity to minority classes. KNN also performed moderately, with acceptable scores but weaker recall and F1-score (around 0.84), highlighting limitations in handling overlapping features or imbalanced data. The Decision Tree model lagged behind, showing the lowest recall (0.650), which undermines its reliability in identifying less-represented classes, despite a respectable specificity of 0.927.

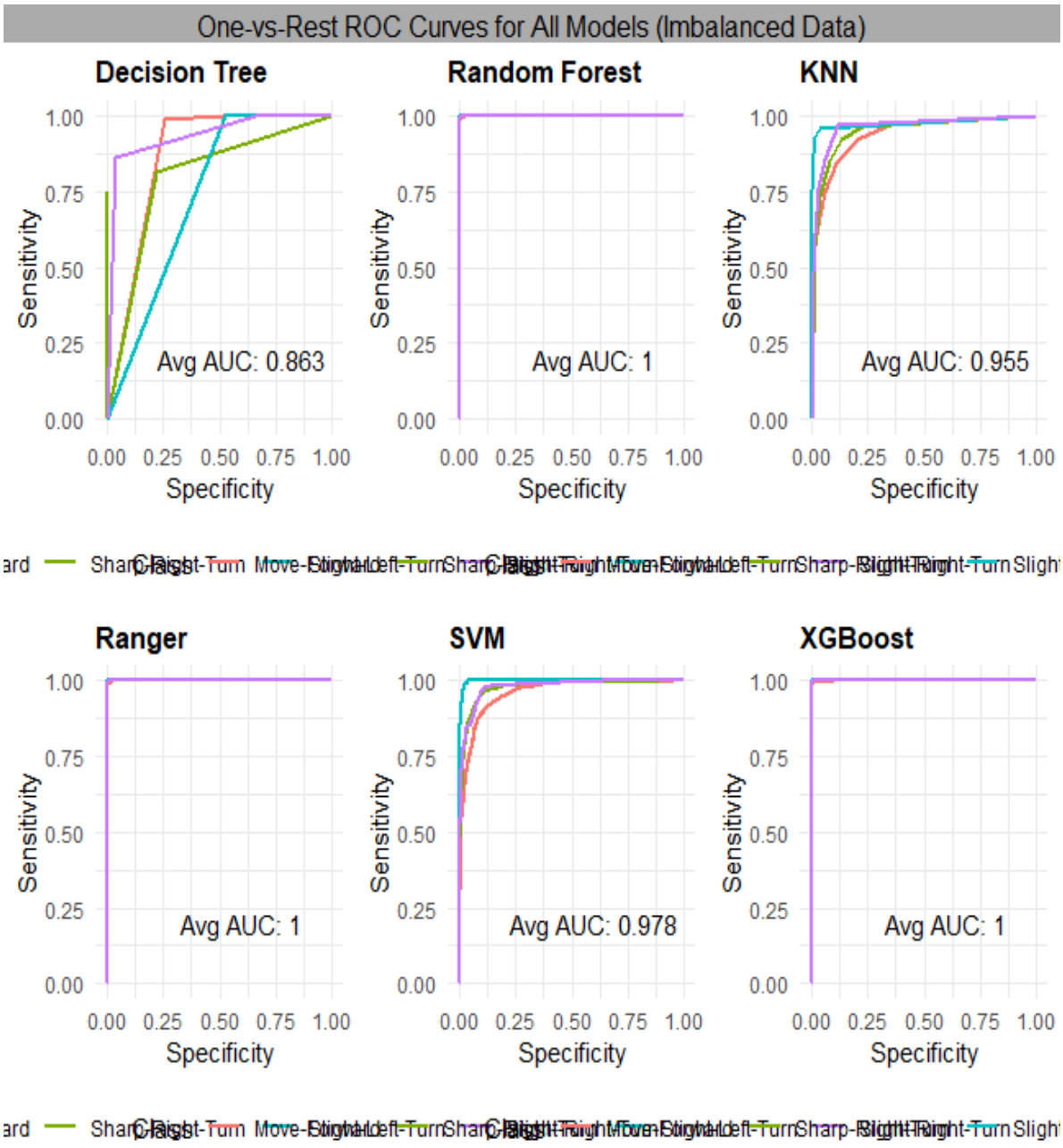

Figure 15: Multi-Class ROC Curves for All Models on Imbalanced Data

The figure 14 presents the multi-class Receiver Operating Characteristic (ROC) curves for each classifier, highlighting their ability to distinguish between classes under imbalanced data conditions. Random Forest, Ranger, and XGBoost exhibit nearly ideal ROC curves, with AUC values approaching 1.0 across all classes, indicating excellent discrimination and robustness. SVM also performs strongly, with high AUC scores but slightly less separation for some classes, reflecting moderate sensitivity to overlap and imbalance. KNN shows moderate class separation, with some flattening of the curves suggesting difficulty in distinguishing closely related classes. Decision Tree demonstrates the weakest ROC performance, with less steep curves and lower AUCs, confirming its challenges in effectively handling imbalanced, multi-class data.

3.3 Balance Data Analysis

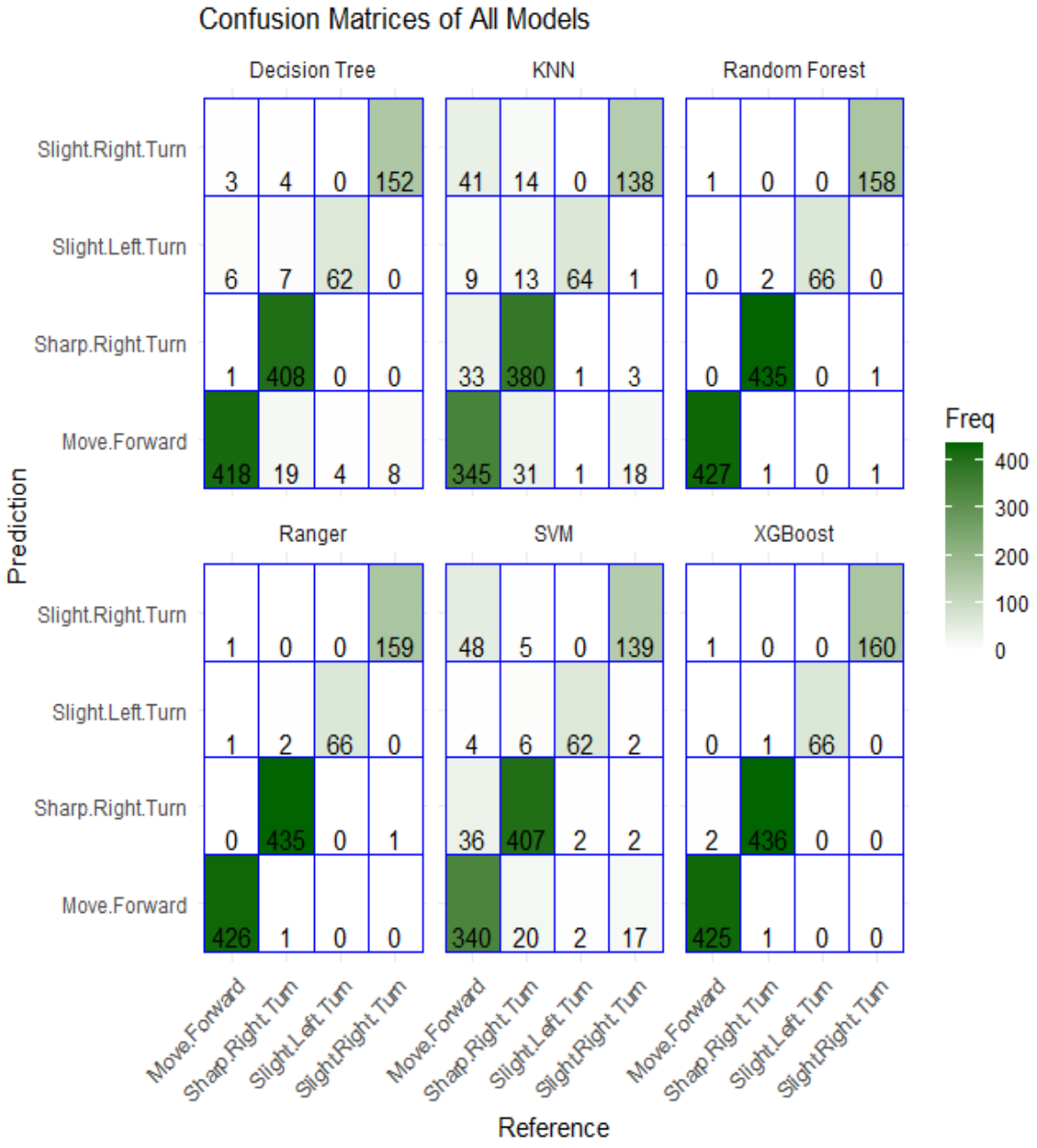

Figure 16: Confusion Matrix of all model for Balanced Data

Ranger and XGBoost show nearly perfect classification across all classes, with minimal misclassifications. This highlights their strength in handling SMOTE-balanced data. Decision Tree and KNN struggle especially with 'Slight.Right.Turn' and 'Move.Forward', often confusing between them—especially DT, which heavily misclassifies 'Slight.Right.Turn' as 'Move.Forward'. SVM performs the weakest, with noticeable confusion among all classes, particularly 'Move.Forward' and 'Slight.Right.Turn', suggesting it is not well-suited for this task even after SMOTE balancing.

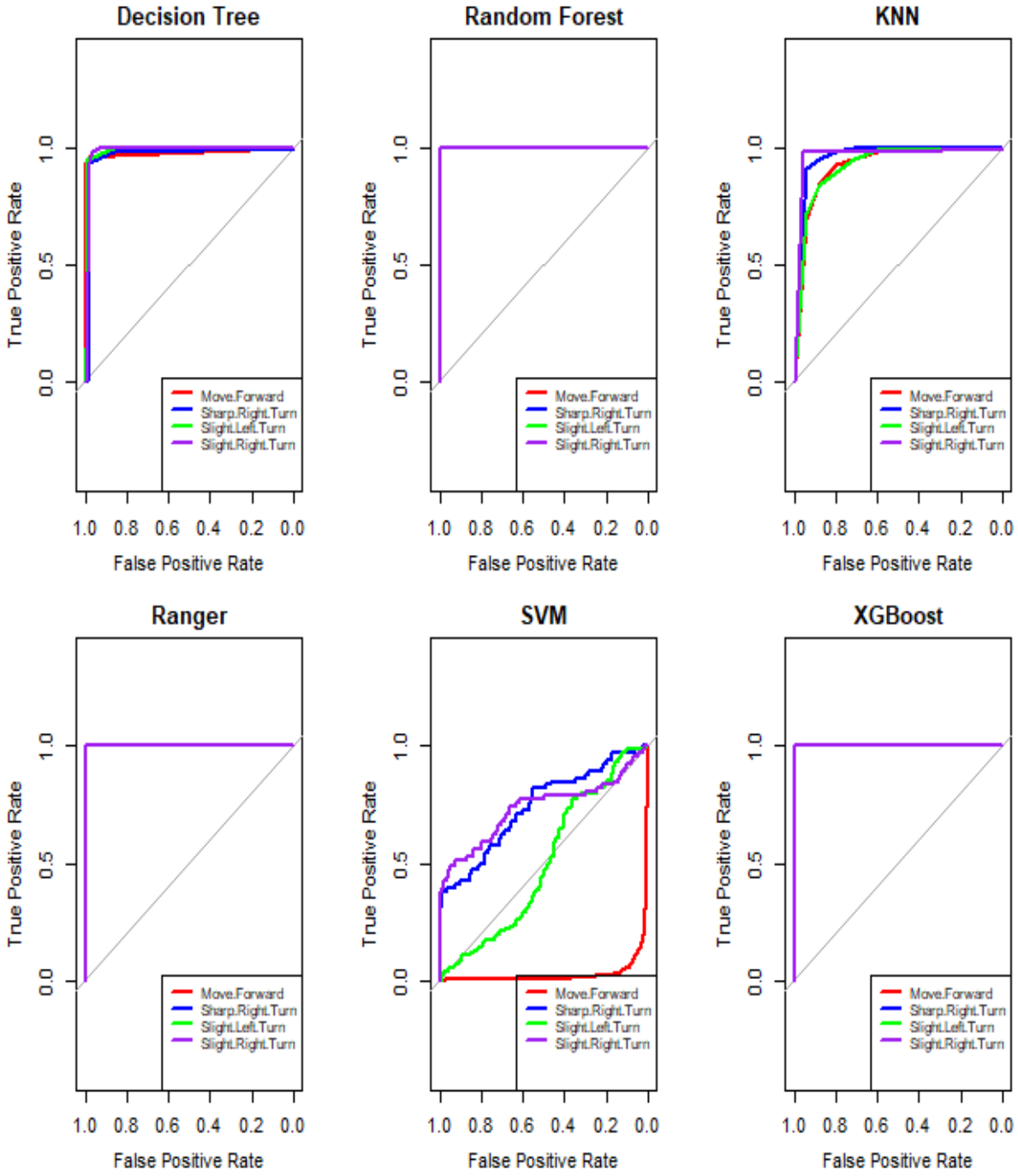

Figure 17: Multi-class ROC Curves of all models for Balanced Data

From the figure 17, Ranger and XGBoost again show ideal ROC curves, with all class curves nearly touching the top-left corner, confirming excellent true positive vs. false positive trade-off. SVM's ROC curves are inconsistent, especially for 'Move.Forward' and 'Slight.Right.Turn', which show curves close to the diagonal—indicating near-random performance for those classes. Random Forest, KNN, and Decision Tree perform reasonably, but with some deviation for certain classes compared to the top-performing models, suggesting room for improvement.

Table 4: Performance Comparison of Classifiers on SMOTE-Balanced Dataset

Model	Accuracy	Precision	Recall	F1_Score	Specificity	AUC
Decision tree	0.960	0.939	0.962	0.949	0.986	0.983
Random forest	0.989	0.980	0.989	0.984	0.996	1.000
KNN	0.715	0.656	0.772	0.681	0.904	0.859
Ranger	0.989	0.979	0.988	0.983	0.996	1.000
SVM	0.808	0.763	0.811	0.780	0.930	0.950
XGBoost	0.996	0.994	0.996	0.996	0.999	1.000

The table summarizes the performance metrics of six classification models—Decision Tree, Random Forest, K-Nearest Neighbors (KNN), Ranger, Support Vector Machine (SVM), and XGBoost—on a SMOTE-balanced dataset addressing class imbalance in multi-class navigation tasks. Ensemble-based models, particularly XGBoost, Ranger, and Random Forest, demonstrate superior performance across all evaluation criteria, achieving accuracy, recall, and F1-scores above 0.98, along with perfect Area Under the Curve (AUC = 1.000). These results underscore the effectiveness of ensemble learning techniques in extracting class-discriminative features from balanced data. The Decision Tree model also exhibits notable improvement compared to its imbalanced-data counterpart, reaching an F1-score of 0.949 and AUC of 0.983. While SVM yields respectable performance (AUC = 0.950), it falls short in recall and F1-score, reflecting moderate sensitivity to class overlap. Conversely, KNN records the lowest metrics across all categories (e.g., accuracy = 0.715, F1-score = 0.681), suggesting limitations in capturing decision boundaries under complex feature distributions. Overall, the balanced data scenario reveals the clear advantage of ensemble approaches in achieving high-fidelity classification in navigation-based sensor data.

The following figure 18 illustrates the comparative performance of six classifiers on SMOTE-balanced navigation data across multiple evaluation metrics. Ranger and XGBoost clearly outperform all other models, achieving near-perfect scores in Accuracy, AUC, Precision, Recall, and F1-Score, highlighting their robustness and ability to generalize well across all classes. In contrast, SVM lags behind with notably lower AUC (~0.75), F1-Score, and Recall, suggesting challenges in handling overlapping classes despite balanced training—possibly due to kernel limitations. Decision Tree and KNN exhibit moderate performance, with reasonable Recall and Precision but lower AUC values than ensemble models, indicating difficulty in modeling complex decision boundaries.

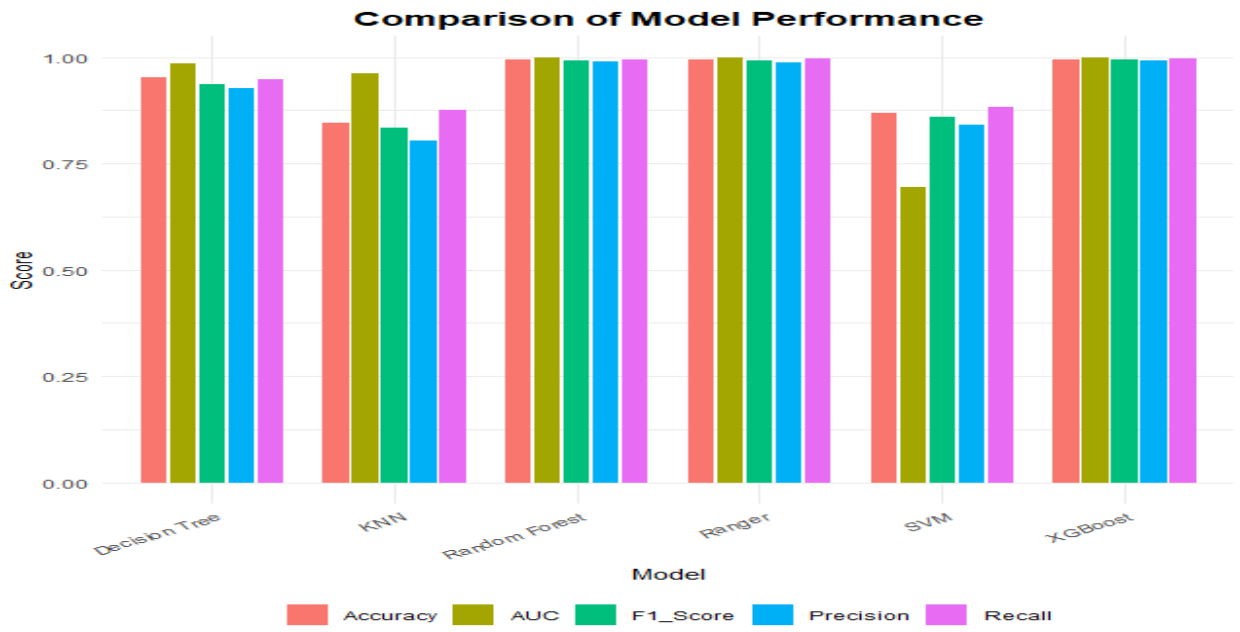

Figure 18: Comparison of ML classifier metrics for Balanced Data

Table 5: Summary Table for Model Performance Comparison: Balanced vs Imbalanced

For Balance data						
Model	**Accuracy**	**Precision**	**Recall**	**F1_Score**	**Specificity**	**AUC**
Decision tree	0.960	0.939	0.962	0.949	0.986	0.983
Random forest	0.989	0.980	0.989	0.984	0.996	1.000
KNN	0.715	0.656	0.772	0.681	0.904	0.859
Ranger	0.989	0.979	0.988	0.983	0.996	1.000
SVM	0.808	0.763	0.811	0.780	0.930	0.950
XGBoost	0.996	0.994	0.996	0.996	0.999	1.000
For Imbalance data						
Model	**Accuracy**	**Precision**	**Recall**	**F1_Score**	**Specificity**	**AUC**
Decision tree	0.987	0.982	0.985	0.983	0.995	0.997
Random forest	0.997	0.994	0.997	0.996	0.999	1.000
KNN	0.862	0.845	0.868	0.856	0.948	0.927
Ranger	0.997	0.994	0.997	0.996	0.999	1.000
SVM	0.889	0.889	0.884	0.886	0.957	0.979
XGBoost	0.998	0.996	0.999	0.997	0.999	1.000

The summary table comparing balanced and imbalanced datasets further reinforces these insights. For imbalanced data, XGBoost and Ranger demonstrated nearly perfect performance, with F1-scores and AUC values approaching 1.000, indicating a strong ability to classify accurately even when data distribution is skewed. Decision Tree and Random Forest also showed solid performance, though with slightly more variation. In contrast, KNN and SVM performed notably worse on the imbalanced dataset, especially in metrics such as recall, specificity, and F1-score. After applying SMOTE, these models saw significant improvements, underlining the importance of data balancing for their optimal use in robotic navigation contexts.

Top Performers	Weak Performers
• **XGBoost** consistently outperforms all other models across both imbalanced and SMOTE-balanced datasets. ○ Imbalanced: F1 = **0.997**, AUC = **1.000** ○ Balanced: F1 = **0.995**, AUC = **1.000** ○ Interpretation: XGBoost is robust to imbalance and benefits slightly from balancing. • **Random Forest** and **Ranger** also perform **exceptionally well**. ○ Both show near-perfect scores (F1 ≈ 0.996–0.984, AUC = 1.000), indicating strong generalization and insensitivity to class imbalance.	• **KNN** and **SVM** perform **notably worse**, especially in the **imbalanced** setting. ○ KNN (Imbalanced): F1 = **0.856**, AUC = **0.921** ○ KNN (Balanced): F1 = **0.681**, AUC = **0.859** ○ SVM (Balanced): AUC = **0.930** (decreased from **0.979** imbalanced), indicating instability with SMOTE

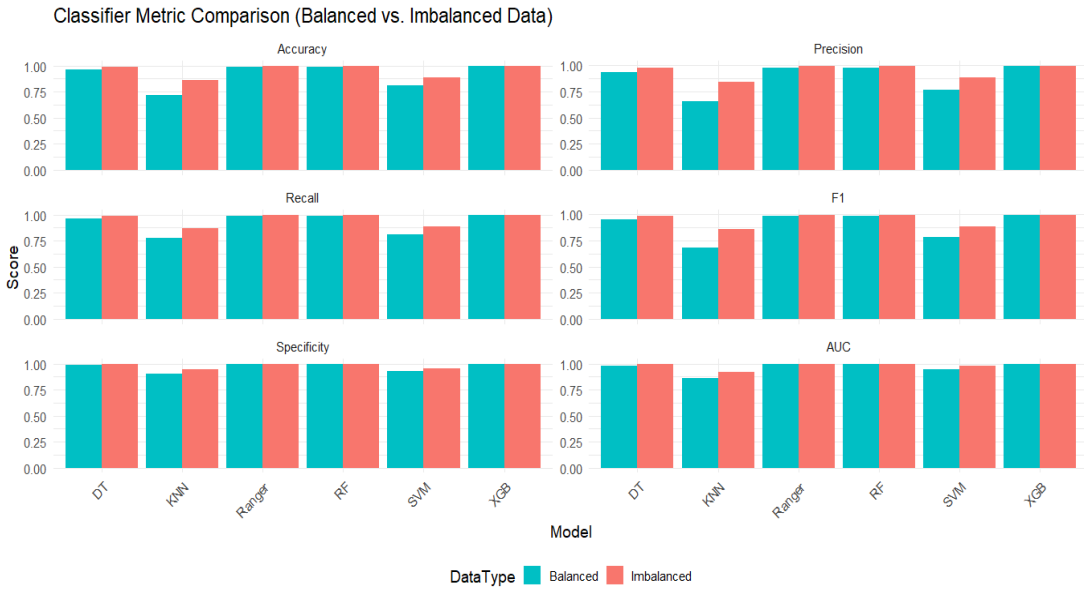

Figure 19: Classifier Metrics comparison (Bar Plot): Balanced vs Imbalanced Data

The metric comparison plots visually confirm that these trends. XGBoost, Ranger, and Random Forest maintained high and consistent scores across all metrics regardless of data balance. Decision Tree showed strong but slightly improved performance with SMOTE. KNN and SVM, however, were clearly more sensitive to class imbalance; both models exhibited substantial increases in all metrics after balancing. These results emphasize that while ensemble models are inherently resilient to class imbalance, simpler algorithms like KNN and SVM benefit considerably from resampling techniques such as SMOTE when using non-ensemble classifier.

4. Concluding Remarks

This research investigates machine learning classifiers in robotic navigation systems, particularly under the challenge of class-imbalanced datasets—a critical factor in autonomous decision-making environments. We observed that the dataset exhibited a high level of imbalance, with Move-Forward and Sharp-Right-Turn dominating the command distribution, while Slight-Left-Turn and Slight-Right-Turn were underrepresented. This imbalance led to reduced

classification accuracy for minority classes, even when overall accuracy appeared high (RQ1).The impact of imbalanced data on classifier performance is well-documented. (He & Garcia, 2009) highlight that traditional classifiers tend to be biased toward majority classes, leading to underperformance in minority class recognition. Techniques like SMOTE (Synthetic Minority Over-sampling Technique) have been introduced to mitigate these effects by generating synthetic samples for minority classes (Chawla et al., 2002). However, as Fernández et al. (2018) caution, the effectiveness of such oversampling varies with classifier architecture and dataset characteristics. Ensemble methods such as Random Forest and XGBoost have shown superior robustness in imbalanced scenarios by combining multiple weak learners (T. Chen & Guestrin, 2016; Y. L. Chen et al., 2013). In robotic navigation, where accurate and timely decisions are paramount, understanding classifier resilience to imbalance is crucial (Kavraki et al., 1996).

Initial evaluation of classifiers showed that ensemble-based models—such as Random Forest, Ranger, and XGBoost—achieved robust predictive outcomes, with XGBoost outperforming others with an F1 score of 0.997 and AUC of 1.00, even under data imbalance. This supports RQ2, as these models consistently performed better than traditional models (SVM, KNN) in handling imbalanced data. For example, KNN achieved lower F1 and AUC values, affirming its sensitivity to skewed distributions. SVM showed moderate resilience but failed to match the reliability of ensemble methods. These findings validate that not all classifiers perform equally under unbalanced conditions and highlight the importance of model selection. To address the imbalance, we applied SMOTE—a synthetic oversampling technique—to generate additional samples for minority classes. The application of SMOTE significantly boosted classification metrics across most models, especially ensemble ones. For example, Ranger's F1 score improved to 0.983 post-balancing, maintaining high sensitivity and precision. Similarly, XGBoost retained its top-tier performance, illustrating that SMOTE helped improve minority class recognition without sacrificing accuracy (RQ3). However, the performance of KNN deteriorated after balancing, likely due to its vulnerability to noise and class overlap introduced by synthetic samples. This mixed response underscores the model-specific effects of oversampling techniques and the need for critical evaluation before application.

In the context of real-time robotic navigation (RQ4), our findings emphasize the suitability of ensemble models—particularly XGBoost, Ranger, and Random Forest—for deployment. These models demonstrated consistent and balanced performance in precision, recall, and AUC, crucial for minimizing misclassification risks in dynamic and time-sensitive environments. Their high computational efficiency and capability to handle noisy and complex data make them reliable candidates for embedded AI systems in autonomous robotics. A complementary simulation study with increasing data volumes validated these results. As sample size grew, accuracy, F1 score, and kappa statistics steadily improved and plateaued above 99% for XGBoost, Ranger, and Random Forest, indicating strong generalization and resistance to overfitting. Conversely, KNN and SVM performance declined slightly due to bias-variance tradeoffs. This aligns with Domingos (2012) and reinforces ensemble methods' appropriateness for scalable robotic navigation applications.

In addition, we conducted a simulation study to validate our findings across increasing sample sizes. As data volume increased, the accuracy, F1 score, and kappa value steadily improved for XGBoost, Ranger, and Random Forest, stabilizing at over 99% accuracy. This result confirms that these models generalize well with larger training sets and are less prone to overfitting, unlike KNN and SVM, whose performance slightly declined due to bias-variance tradeoffs. These simulation results closely mirrored our main findings, further strengthening our model recommendations. Moreover, our study highlights the importance of class balancing strategies in robotic systems where decision precision is non-negotiable. While all machine learning algorithms facilitated faster processing of command prediction and decision-making—key aspects of real-time robotics—only a subset demonstrated reliable performance under class distribution challenges.

This research opens several future directions:

- Expanding model comparisons to include deep learning frameworks such as RNN, CNN, and LSTM to assess their performance under imbalance.

- Investigating advanced ensemble methods or hybrid models that integrate boosting and bagging to enhance navigation safety.
- Evaluating the impact of real-world sensor noise and uncertainty on model predictions in unstructured environments.
- Applying explainability techniques (e.g., SHAP, LIME) to increase transparency in high-performing ensemble classifiers.
- Developing robust, low-latency embedded systems for real-time deployment of these models in mobile robots.

Despite these advancements, a key limitation lies in the synthetic nature of SMOTE, which may not always replicate the complexity of real-world robotic movement patterns. Therefore, future studies should explore real-time data augmentation and generative models for more authentic balancing.

In conclusion, this study demonstrates that class imbalance significantly influences robotic navigation prediction performance (RQ1), with ensemble models outperforming traditional ones (RQ2). SMOTE offers an effective solution to balance data and boost minority class sensitivity (RQ3). Finally, XGBoost, Ranger, and Random Forest emerge as the most reliable models for real-time robotic applications, achieving strong generalization and robust metrics across conditions (RQ4).

References

[1] Aguas, X., Revelo, J., Herrera, M., Cuaycal, A., Camacho, O., Aguas, X., Revelo, J., Herrera, M., Cuaycal, A., & Camacho, O. (2019). Controlador PD-Difuso para seguimiento de pared de un robot móvil: validación experimental. *Revista Digital Novasinergia*, *2*(2), 49–57. https://doi.org/10.37135/UNACH.NS.001.04.05

[2] Alotaibi, A., Alatawi, H., Binnouh, A., Duwayriat, L., Alhmiedat, T., & Alia, O. M. (2024). Deep Learning-Based Vision Systems for Robot Semantic Navigation: An Experimental Study. *Technologies*, *12*(9), 157. https://doi.org/10.3390/TECHNOLOGIES12090157

[3] ALPAYDIN E. (2020). INTRODUCTION TO MACHINE LEARNING. *MIT Press*. https://scholar.google.com/scholar?hl=en&as_sdt=0%2C5&q=ALPAYDIN.+%282016%29.+INTRODUCTION+TO+MACHINE+LEARNING.+PHI+LEARNING.&btnG=

[4] Altman, N. S. (1992). An introduction to kernel and nearest-neighbor nonparametric regression. *American Statistician*, *46*(3), 175–185. https://doi.org/10.1080/00031305.1992.10475879

[5] Bongard, J. (2008). Probabilistic Robotics. Sebastian Thrun, Wolfram Burgard, and Dieter Fox. (2005, MIT Press.) 647 pages. *Artificial Life*, *14*(2), 227–229. https://doi.org/10.1162/ARTL.2008.14.2.227

[6] Breiman, L. (2001). Random forests. *Machine Learning*, *45*(1), 5–32. https://doi.org/10.1023/A:1010933404324/METRICS

[7] Buda, M., Maki, A., & Mazurowski, M. A. (2018). A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, *106*, 249–259. https://doi.org/10.1016/J.NEUNET.2018.07.011

[8] Cadena, C., Carlone, L., Carrillo, H., Latif, Y., Scaramuzza, D., Neira, J., Reid, I., & Leonard, J. J. (2016). Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, *32*(6), 1309–1332. https://doi.org/10.1109/TRO.2016.2624754

[9] Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic Minority Over-sampling Technique. *Journal of Artificial Intelligence Research*, *16*, 321–357. https://doi.org/10.1613/JAIR.953

[10] Chen C, Seff A, Kornhauser A, & Xiao J. (2015). Deepdriving: Learning affordance for direct perception in autonomous driving. *Proceedings of the IEEE International Conference on Computer Version*, 2722–2730. http://openaccess.thecvf.com/content_iccv_2015/html/Chen_DeepDriving_Learning_Affordance_ICCV_2015_paper.html

[11] Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, *13-17-August-2016*, 785–794. https://doi.org/10.1145/2939672.2939785/SUPPL_FILE/KDD2016_CHEN_BOOSTING_SYSTEM_01-ACM.MP4

[12] Chen, Y. L., Cheng, J., Lin, C., Wu, X., Ou, Y., & Xu, Y. (2013). Classification-based learning by particle swarm optimization for wall-following robot navigation. *Neurocomputing*, *113*, 27–35. https://doi.org/10.1016/J.NEUCOM.2012.12.037

[13] Choi, R. Y., Coyner, A. S., Kalpathy-Cramer, J., Chiang, M. F., & Peter Campbell, J. (2020). Introduction to machine learning, neural networks, and deep learning. *Translational Vision Science and Technology*, *9*(2). https://doi.org/10.1167/TVST.9.2.14

[14] Cortes, C., Vapnik, V., & Saitta, L. (1995). Support-vector networks. *Machine Learning 1995 20:3*, *20*(3), 273–297. https://doi.org/10.1007/BF00994018

[15] Cover, T. M., & Hart, P. E. (1967). Nearest Neighbor Pattern Classification. *IEEE Transactions on Information Theory*, *13*(1), 21–

27. https://doi.org/10.1109/TIT.1967.1053964
[16] Dietterich, T. G. (2000). Ensemble Methods in Machine Learning. *Lecture Notes in Computer Science (Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, *1857 LNCS*, 1–15. https://doi.org/10.1007/3-540-45014-9_1
[17] Domingos, P. (2012). A few useful things to know about machine learning. *Communications of the ACM*, *55*(10), 78–87. https://doi.org/10.1145/2347736.2347755
[18] Fawcett, T. (2006). An introduction to ROC analysis. *Pattern Recognition Letters*, *27*(8), 861–874. https://doi.org/10.1016/J.PATREC.2005.10.010
[19] Fernández, A., García, S., Galar, M., Prati, R. C., Krawczyk, B., & Herrera, F. (2018). Learning from Imbalanced Data Sets. *Learning from Imbalanced Data Sets*. https://doi.org/10.1007/978-3-319-98074-4
[20] Geiger, A., Lenz, P., & Urtasun, R. (2012). Are we ready for autonomous driving? the KITTI vision benchmark suite. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 3354–3361. https://doi.org/10.1109/CVPR.2012.6248074
[21] Ghandibidgoli, S., Data, H. M.-J. of A. and, & 2022, undefined. (2022). Automatic control and guidance of mobile robot using machine learning methods. *Journals.Shahroodut.Ac.IrS Ghandibidgoli, H MokhtariJournal of AI and Data Mining, 2022•journals.Shahroodut.Ac.Ir*, *10*(3), 385–400. https://doi.org/10.22044/jadm.2022.11278.2286
[22] Grandini, M., Bagli, E., & Visani, G. (2020). *Metrics for Multi-Class Classification: an Overview*. https://arxiv.org/pdf/2008.05756
[23] Haixiang, G., Yijing, L., Shang, J., Mingyun, G., Yuanyue, H., & Bing, G. (2017). Learning from class-imbalanced data: Review of methods and applications. *Expert Systems with Applications*, *73*, 220–239. https://doi.org/10.1016/J.ESWA.2016.12.035
[24] Hammad, I., El-Sankary, K., & Gu, J. (2019). A comparative study on machine learning algorithms for the control of a wall following robot. *IEEE International Conference on Robotics and Biomimetics, ROBIO 2019*, 2995–3000. https://doi.org/10.1109/ROBIO49542.2019.8961836
[25] He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, *21*(9), 1263–1284. https://doi.org/10.1109/TKDE.2008.239
[26] Izquierdo-Verdiguier, E., & Zurita-Milla, R. (2020). An evaluation of Guided Regularized Random Forest for classification and regression tasks in remote sensing. *International Journal of Applied Earth Observation and Geoinformation*, *88*, 102051. https://doi.org/10.1016/J.JAG.2020.102051
[27] Japkowicz, N., & Stephen, S. (2002). The class imbalance problem: A systematic study. *Intelligent Data Analysis*, *6*(5), 429–449. https://doi.org/10.3233/IDA-2002-6504
[28] Kavraki, L. E., Švestka, P., Latombe, J. C., & Overmars, M. H. (1996). Probabilistic roadmaps for path planning in high-dimensional configuration spaces. *IEEE Transactions on Robotics and Automation*, *12*(4), 566–580. https://doi.org/10.1109/70.508439
[29] Kiran, B. R., Sobh, I., Talpaert, V., Mannion, P., Sallab, A. A. A., Yogamani, S., & Perez, P. (2022). Deep Reinforcement Learning for Autonomous Driving: A Survey. *IEEE Transactions on Intelligent Transportation Systems*, *23*(6), 4909–4926. https://doi.org/10.1109/TITS.2021.3054625
[30] Kohavi Ron. (1995). A Study of Cross-Validation and Bootstrap for Accuracy Estimation and Model Selection. *Appears in the International Joint Conference on Arti Cial Intelligence (IJCAI)*, *14*(2), 1137–1145. https://www.researchgate.net/profile/Ron-Kohavi/publication/2352264_A_Study_of_Cross-Validation_and_Bootstrap_for_Accuracy_Estimation_and_Model_Selection/links/02e7e51bcc14c5e91c000000/A-Study-of-Cross-Validation-and-Bootstrap-for-Accuracy-Estimation-and-
[31] Kotsiantis SB, Zaharakis I, & P Pintelas. (2007). Supervised machine learning: A review of classification techniques. *Emerging Artificial Intelligence Applications in Computer Engineering*, *31*, 249–268. https://books.google.com/books?hl=en&lr=&id=vLiTXDHr_sYC&oi=fnd&pg=PA3&dq=Kotsiantis,+S.+B.+(2007).+Supervised+machine+learning:+A+review+of+classification+techniques.+Informatica,+31(3),+249–268.&ots=C_oyBz1Kko&sig=3Xw3t6pld9yP18XOpoVN8fjgCDk
[32] Kursa, M. B., & Rudnicki, W. R. (2010). Feature Selection with the Boruta Package. *Journal of Statistical Software*, *36*(11), 1–13. https://doi.org/10.18637/JSS.V036.I11
[33] Oscar Camacho., Aguas, Andres Cuaycal, Xavier, Jefferson revelo, & marco herrera. (2019). Controlador PD-Difuso para seguimiento de pared de un robot móvil: validación experimental. *Revista Digital Novasinergia*, *2*(2), 49–57. http://scielo.senescyt.gob.ec/scielo.php?pid=S2631-26542019000200049&script=sci_arttext
[34] Powers, D. M. W., & Ailab. (2020). *Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation*. https://arxiv.org/pdf/2010.16061
[35] Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning 1986 1:1*, *1*(1), 81–106. https://doi.org/10.1007/BF00116251
[36] Safavian, S. R., & Landgrebe, D. (1991). A Survey of Decision Tree Classifier Methodology. *IEEE Transactions on Systems, Man and Cybernetics*, *21*(3), 660–674. https://doi.org/10.1109/21.97458
[37] Sokolova, M., & Lapalme, G. (2009). A systematic analysis of performance measures for classification tasks. *Information*

Processing & Management, *45*(4), 427–437. https://doi.org/10.1016/J.IPM.2009.03.002

[38] Somiya Rani, Jain Amita, & Castilo Oscar. (2020). Research trends on fuzzy logic controller for mobile robot navigation: A scientometric study. *Journal of Automation Mobile Robotics and Intelligent Systems*, *14*. https://doi.org/10.14313/JAMRIS/1-2020/11

[39] van Engelen, J. E., & Hoos, H. H. (2020). A survey on semi-supervised learning. *Machine Learning*, *109*(2), 373–440. https://doi.org/10.1007/S10994-019-05855-6/FIGURES/5

[40] Wright, M. N., & Ziegler, A. (2017). ranger: A Fast Implementation of Random Forests for High Dimensional Data in C++ and R. *Journal of Statistical Software*, *77*(1), 1–17. https://doi.org/10.18637/JSS.V077.I01

[41] Ž Vujović. (2021). Classification model evaluation metrics. *International Journal of Advanced Computer Science and Applications*, *12*(6), 599–606. https://doi.org/10.14569/IJACSA.2021.0120670

[42] Grandini, M., Bagli, E., & Visani, G. (2020). Metrics for Multi-Class Classification: an Overview. 1–17. http://arxiv.org/abs/2008.05756