T.C.

BAHCESEHIR UNIVERSITY

GRADUATE SCHOOL

THE DEPARTMENT OF ARTIFICIAL INTELLIGENCE

A NOVEL ENSEMBLE FRAMEWORK FOR XAI-BASED FEATURE SELECTION IN MACHINE LEARNING MODELS

MASTER'S THESIS

HALİL İBRAHİM DEMİREL

ISTANBUL 2024

T.C.

BAHCESEHIR UNIVERSITY

GRADUATE SCHOOL

THE DEPARTMENT OF ARTIFICIAL INTELLIGENCE

A NOVEL ENSEMBLE FRAMEWORK FOR XAI-BASED FEATURE SELECTION IN MACHINE LEARNING MODELS

MASTER'S THESIS

THESIS ADVISOR

PROF. DR. SÜREYYA AKYÜZ

ISTANBUL 2024

T.C.

BAHCESEHIR UNIVERSITY

GRADUATE SCHOOL

MASTER THESIS APPROVAL FORM

Program Name:	Artificial Intelligence
Student's Name and Surname:	Halil İbrahim Demirel
Name Of The Thesis:	A Novel Ensemble Framework for XAI-Based Feature Selection in Machine Learning Models
Thesis Defense Date:	02.09.2024

This thesis has been approved by the Graduate School which has fulfilled the necessary conditions as Master thesis.

Assoc. Prof. Yücel Batu SALMAN

Institute Director

This thesis was read by us, quality and content as a Master's thesis has been seen and accepted as sufficient.

	Title/Name	**Institution**	**Signature**
Thesis Advisor's	Prof. Dr. Süreyya Akyüz	Bahçeşehir University	
Member's	Assist. Prof. Tarkan Aydın	Bahçeşehir University	
Member's	Assoc. Prof. Burcu Tunga	İstanbul Technical University	

I hereby declare that all information in this document has been obtained and presented in accordance with academic rules and ethical conduct. I also declare that, as required by these rules and conduct, I have fully cited and referenced all material and results that are not original to this work.

Name, Surname : Halil İbrahim DEMİREL

Signature:

ABSTRACT

A NOVEL ENSEMBLE FRAMEWORK FOR XAI-BASED FEATURE SELECTION IN MACHINE LEARNING MODELS

Halil İbrahim Demirel
Master's Program in Artificial Intelligence
Supervisor: Prof. Dr. Süreyya Akyüz

September 2024, 64 pages

To improve the effectiveness and readability of machine learning (ML) models, this study explores the impact of feature selection on model performance. It investigates how models like Random Forests and Gradient Boosting Machines can be integrated with Explainable Artificial Intelligence (XAI) techniques using SHAP. Various feature selection methods are examined using data from sources like SpamBase, Breast Cancer Wisconsin, and EEG Creativity. The new hybrid feature selection method enhances model accuracy and interpretability without sacrificing precision. The results demonstrate that the combination approach provides a comprehensive solution for achieving precision while enhancing model interpretability. The approach outlined in this study advances the development of reliable and effective machine learning models, particularly in handling complex models and large datasets. Additionally, tests conducted using EEG data validate the efficiency of the proposed techniques in data fields such as biomedical data analysis. This thesis marks a significant advancement in the theoretical foundations and practical applications of machine learning. The practicality and scalability of the proposed approaches offer valuable assets to the machine-learning community. The findings of this study could influence crucial decision-making processes where clear and effective models are essential.

Key Words: Feature Selection, Machine Learning, Explainable Artificial Intelligence, SHAP

ÖZ

MAKİNE ÖĞRENME MODELLERİNDE XAI TABANLI ÖZELLİK SEÇİMİ İÇİN YENİ BİR TOPLULUK ÇERÇEVESİ

Halil İbrahim Demirel
Yapay Zeka Yüksek Lisans Programı
Tez Danışmanı: Prof. Dr. Süreyya Akyüz

Eylül 2024, 64 sayfa

Makine öğrenimi (ML) modellerinin etkinliğini ve anlaşılabilirliğini artırmak amacıyla bu çalışma, özellik seçiminin model performansı üzerindeki etkisini araştırmaktadır. Random Forest ve Gradient Boosting Machines gibi modellerin, SHAP kullanılarak Açıklanabilir Yapay Zeka (XAI) teknikleri ile nasıl entegre edilebileceğini incelemektedir. SpamBase, Wisconsin Meme Kanseri ve EEG Yaratıcılık gibi veri kaynaklarından elde edilen veriler kullanılarak çeşitli özellik seçimi yöntemleri değerlendirilmiştir. Yeni hibrit özellik seçimi yöntemi, model doğruluğunu ve açıklanabilirliğini artırırken, kesinlikten ödün vermeden kapsamlı bir çözüm sunmaktadır. Bu çalışmada ortaya konan yaklaşım, karmaşık modellerin ve büyük veri setlerinin işlenmesinde güvenilir ve etkili makine öğrenimi modellerinin geliştirilmesine katkı sağlamaktadır. Ayrıca, EEG verileriyle yapılan testler, biyomedikal veri analizi gibi alanlarda önerilen tekniklerin verimliliğini doğrulamaktadır. Bu tez, makine öğreniminin teorik temelleri ve pratik uygulamalarında önemli bir ilerleme kaydetmektedir. Önerilen yaklaşımların uygulanabilirliği ve ölçeklenebilirliği, makine öğrenimi topluluğuna değerli katkılar sunmaktadır. Çalışmanın bulguları, net ve etkili modellerin hayati önem taşıdığı karar verme süreçlerini etkileyebilir.

Anahtar Kelimeler: Özellik Seçimi, Makine Öğrenmesi, Açıklanabilir Yapay Zeka, SHAP

To My Wife

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to my supervisor, Prof. Dr. Süreyya AKYÜZ, for her guidance, valuable advice, criticism, encouragement, and profound insight throughout this research.

I also wish to extend my heartfelt thanks to my wife, Buket DEMİREL, and my sons, Ömer Asaf DEMİREL and Kadir Uras DEMİREL, for their unwavering support throughout my life. Without their understanding and continuous encouragement, aspiring to this level of education and completing this study would not have been possible.

TABLE OF CONTENTS

LIST OF TABLES

TABLES

LIST OF FIGURES

FIGURES

Chapter 1

Introduction

Feature selection in machine learning can be likened to searching for relevant books on a specific topic in a vast library. This procedure entails determining the factors that significantly affect a model's capacity for prediction. While traditional approaches are helpful, they may overlook the nuances of how features interact and influence the model's outcomes. In this scenario, Explainable Artificial Intelligence (XAI) aims to show how model predictions are made. This study delves into a method that boosts feature selection by combining XAI techniques with a particular focus on SHAP (Shapley Additive Explanations).

Machine learning has made advancements in marketing, finance, and healthcare. However, there is still some ambiguity about how these models work in situations that involve crucial decision-making. Feature selection is crucial in refining models and improving their clarity. All standard techniques have advantages and limitations, including filter-based methods, wrapper approaches, and embedded strategies. Although filters can capture interactions between features at times, they may not always do so because they assess the importance of features separately from any learning approach.

When considering methods like wrapper and embedded techniques for selecting features in learning algorithms, we also consider how well the algorithm performs a view expressed (Kohavi & John, 1997; Guyon & Elisseeff, 2003). They highlighted that these methods may require many resources and be prone to overfitting issues. XAI, like the SHAP technique developed (Lundberg & Lee, 2017), provides a view on selecting features to make better-informed decisions regarding how features impact model predictions. The study integrates SHAP with various tools, like Gradient Boosting, Random Forests, LighGBM, XGBoost, and Decision Trees, to identify crucial features. The goal is to shorten the time required to train machine learning models while maintaining accuracy and precision.

1.1 Statement of the Problem

The techniques used for feature selection play an important role in determining the relevant features required for tasks involving analysis and prediction. Traditional approaches, such as wrapper and embedding methods, evaluate features based on their performance with learning algorithms. It can require significant resources and lead to overfitting issues. Popular algorithms such as Gradient Boosting, Random Forest, LightGBM, and XGBoost use embedded techniques to identify features during selection. These methods can be complex and challenging regarding the power required for analysis. Moreover, these strategies may not fully consider how different features interact, making it more difficult to understand the model's functioning.

1.2 Goal of the Research

This study aims to establish an accurate method for selecting features in models using XGBoost, LightGBM, Decision Tree, Random Forest, Gradient Boosting, and the SHAP technique. SHAP makes the feature selection process more informed and efficient by evaluating how features affect model predictions. Specifically, this research aims to deepen the understanding of how features interact in models, thereby increasing the efficiency and interpretability of these models and improving running times to reduce resource usage.

1.3 Importance of the Research

This work is focused on improving the precision and comprehensibility of machine learning models to facilitate the development of understandable models for critical decision-making procedures. Implementing SHAP enables a deeper insight into the distinct and combined impacts of features within the model, ultimately enhancing the internal clarity of how the model functions (Lundberg & Lee, 2017). As a result of this improvement in transparency and understanding, the model's predictive capacity is preserved while decreasing running times and streamlining the feature selection process.

Chapter 2

Literature Review

The field of feature selection in literature is extensive, covering methods such as filter and wrapper techniques and embedded approaches. Filters like information and correlation coefficients evaluate features independently of the model. On the other hand, wrappers such as feature elimination rely on the model performance to assess feature subsets but can be computationally intensive (Guyon & Elisseeff, 2003). Embedded methods, like LASSO (Least Absolute Shrinkage and Selection Operator) and decision trees, incorporate feature selection into the model training process (Tibshirani, 1996; Breiman, 2001).

2.1 Filter Methods

Filtering methods are commonly used before applying machine learning techniques to streamline the process. These methods are efficient as they evaluate feature relevance independently of any specific learning algorithm. Some known filtering methods include information, correlation coefficients, and chi-square tests. Mutual information gauges how much insight is gained about one variable from another, making it a valuable tool for selecting features. However, it does not account for interactions between features. Correlation coefficients evaluate the linear dependency between features and the target variable. While easy to implement, they may overlook linear relationships. Chi-square tests are suitable for data. Assess the independence between features and the target variable. Despite their effectiveness, filter methods often fail to capture feature interactions for many machine-learning tasks.

Filtering methods offer benefits by not necessitating model creation, thereby conserving resources. They prove handy in early feature selection stages to swiftly weed out features. Nonetheless, their drawback lies in overlooking interactions among features and the predictive model. For example, while a feature may seem insignificant, it could provide insights when combined with another feature.

In situations where feature interactions are significant, the effectiveness of filter methods is reduced.

2.2 Wrapper Methods

Wrapping techniques evaluate the quality of feature subsets through training. They are evaluating a machine learning model with combinations of features. Some typical examples of wrapper methods are forward feature selection, recursive feature elimination, and backward feature elimination. These approaches can capture feature interactions, often making them more effective than filter methods. However, they are computationally intensive as they require training models. Forward selection begins with no features. In forward selection, the most important feature is added at each step. All features are initially included in backward elimination, and the least significant one is removed at each stage. A model is iteratively trained through recursive feature removal, gradually eliminating less essential features.

While wrapper methods can yield high-quality feature subsets tailored to models by considering feature interactions, their computational demands may render them impractical for datasets or complex models (Kohavi & John, 1997; Guyon *et al*., 2002).

Wrapper methods excel at capturing feature interactions by evaluating predictive model performance to provide a range of characteristics for a specific model. Nevertheless, their high computational cost presents a drawback. The exhaustive evaluation of all feature subsets becomes unmanageable for datasets. Moreover, wrapper methods are susceptible to overfitting when dealing with features and observations.

To overcome this limitation, researchers should apply validation techniques to ensure that the selected features generalize well to a broader context.

2.3 Embedded Techniques

These methods perform feature selection during the model training process. Examples include decision trees, LASSO, and Ridge Regression. Decision trees inherently select features by making the split at each node. LASSO introduces an L1 penalty to shrink a few coefficients to zero, allowing efficient feature selection. Ridge

Regression uses an L2 penalty, which can also select features but often results in zero coefficients. These methods are efficient computationally. Can capture feature interactions. However, they are specific to the model. It may not generalize well to models. For example, the significance of features in a decision tree may not apply to a regression model (Tibshirani, 1996; Zou & Hastie, 2005).

The benefit of embedded methods is that they combine feature selection with model training, reducing complexity compared to wrapper methods. Decision trees naturally choose features by finding splits that offer information gain. Similarly, LASSO combines selection and regularization, leading to a more understandable model. Nonetheless, the model-specific nature of embedded techniques can pose limitations.

The features chosen might work best for the model during training but not effectively with other models. This limited adaptability could constrain the versatility of these approaches in modeling situations.

2.4 Tree-Based Machine Learning Models

Detailed information about the Tree-Based Machine Learning Models used in the study is provided below.

2.4.1 Gradient Boosting. Gradient Boosting is a strong machine-learning technique widely used for regression and classification tasks. It utilizes a gradient descent optimization approach to minimize a loss function by sequentially training models that correct the errors made by previous models (Friedman, 2001). As known as Gradient Boost Machines (GBMs), this method is highly respected for its capability to improve model accuracy by dealing with remaining errors step-by-step (Natekin & Knoll, 2013).

A simple model (often started with a significant bias) is the foundation of a gradient-boosting ensemble. New models are added to the ensemble, correcting the preceding errors and gradually improving the process.

At each stage, the residuals (errors) are calculated based on the model's current predictions, and a new base learner is trained on these residuals. Thus, the new learner

is included in the ensemble, and the procedure continues until the model achieves the required performance or the specified iterations are completed.

One of Gradient Boosting's main advantages is its capacity to manage massive datasets and intricate variable connections. According to Chen and Guestrin (2016), Gradient Boosting was among the most highly regarded machine learning models, using a variety of benchmark datasets. Unlike many other algorithms, Gradient Boosting can handle category variables directly without converting them into numerical values. Compared to other boosting techniques, it is less prone to overfitting because of its ability and efficiency in modeling non-linear connections (Chen & Guestrin, 2016).

Gradient Boost can also work with convolution neural networks and other complex machine learning models besides simple decision trees (Walach & Wolf, 2016; Badirli *et al*., 2020).

Despite its advantages, gradient boosting has associated challenges. Training the model using datasets can consume significant time and resources (Chen & Guestrin, 2016). The model's sequential nature makes parallelizing challenging, leading to longer running times. For results, precise adjustments to hyperparameters, such as the number of base models and the learning rate, are necessary when employing Gradient Boosting.

Because of their intricacy, understanding gradient-boosted models can be challenging (Friedman, 2001). To some extent, this difficulty can be mitigated using shrinkage techniques suggested by Zeiler (2012) and modifying the learning rate. These adjustments help control the learning process and improve the model's resistance to overfitting problems.

Gradient Boost has demonstrated efficacy in various domains, including disease risk assessment (Ma *et al*., 2022), credit risk evaluation (Chang *et al*., 2018), mobility pattern prediction (Semjanisk & Gautama, 2015), and money laundering prevention (Valssallo *et al*., 2021). Its adaptability is demonstrated in several domains, including regions (Georganos *et al*., 2018), medical diagnosis (Liu *et al*., 2020), and visual identification skills (Zhang *et al*., 2013).

2.4.2 Random Forest. This technique, introduced by Ho (1995), is a recognized supervised ensemble learning model widely utilized for regression and

classification in machine learning. Breiman (2001) refined the concept, making it a widely adopted implementation of the bagging ensemble method. Random Forest (RF) stands out from bagging techniques by utilizing decision trees as its fundamental building blocks rather than any arbitrary base model type. The model constructs decision trees from different random subsets of the data. A bootstrap sample trains each tree, and features are randomly chosen at each split (Ho, 1995; Breiman, 2001).

Among Random Forest's features is its ability to process features independently without requiring every feature to be present in every bagged sample. This method of segmenting the features increases the diversity of the trees in the forest. It also decreases variance, which can enhance the model's generalization capacity, though at the cost of potentially increasing bias (Biau & Scornet, 2016). Consolidating the outcomes from each tree using majority voting for classification tasks and averaging for regression tasks determines the final output produced by the RF model (Breiman, 2001).

Random Forests have a trait in their ability to make decisions clearly and effectively by assessing feature importance and deriving rules, as highlighted by Bénard *et al*. (2021). This transparency aids users in comprehending the significance of features in predicting outcomes. It empowers them to streamline feature selection and decrease complexity (Alam *et al*., 2019). Moreover, Random Forests are well suited for handling data sets due to their processing capabilities.

In fields like image processing, Random Forest models are frequently utilized for tasks like object classification, boundary detection, and result prediction. They are highly beneficial in sensing applications for handling vast amounts of image data efficiently, according to Belgiu and Drăguţ's (2016) findings in a research paper. In the healthcare sector, within the medical diagnosis and brain imaging for conditions such as Alzheimer's disease, as Sarica et al. (2017) indicated in their study, Random Forest models demonstrate significant potential. For example, Khalilia *et al*. (2011) showed that Random Forests performed better than models like support vector machines in forecasting disease likelihood using patients' medical backgrounds.

Random Forests have also been crucial in addressing complex and imbalanced problems. Advancements in this field involve the creation of decision forests that consider costs (Siers & Islam, 2015), methods for selecting features based on

categories (Wu *et al*., 2014), and classifiers that assign weights to classes (Zhu *et al*., 2018).

2.4.3 XGBoost. XGBoost is a version of Gradient Boosting that integrates sophisticated regularization methods like Lasso (L1) and Ridge (L2) to improve the model's generalization ability. Chen and Guestrin highlighted this approach in 2016. Compared to the Gradient Boosting technique, XGBoost demonstrates enhanced computational efficiency thanks to its capability for parallel processing across clusters, leading to a significant reduction in running time (Chen et al., 2015).

The XGBoost algorithm utilizes Taylor series approximation to modify the objective function into a format with which traditional optimization methods can work more effectively. At each iteration, the model aims to improve loss minimization. The log loss objective function is used by the XGBoost algorithm in assignments involving categorization scenarios. Its remarkable scalability and accuracy have helped it advance to a place in machine learning competitions, especially on websites such as Kaggle (Bojer & Meldgaard, 2021). The algorithm’s ability to efficiently handle distributed computing and parallel processing makes it ideal for addressing large datasets and challenging problems.

In addition to being computationally efficient at its core, XGBoost offers a variety of regularization options, including Ridge (L2) and Lasso (L1) regularization, to help prevent overfitting problems. However, like other ensemble learning methods, the choice of parameters significantly impacts XGBoost's effectiveness, necessitating careful changes for the best outcomes. XGBoost preserves interpretability despite its ensemble structure and specially created trees by providing essential feature importance metrics that support feature selection and understanding of data linkages (Banga *et al*., 2021).

XGBoost has proven successful in a variety of industries. For example, projecting sales statistics (Dairu & Shilong, 2021), anticipating stock market trends (Naik & Mohan, 2021), and predicting energy consumption (Wang *et al*., 2017). This model has succeeded in various domains, including real-time accident detection (Parsa *et al*., 2020) and identifying different types of rocks (Zhang & Zhan, 2017). It excels at multiclass classification problems. Furthermore, XGBoost has been applied in

forecasting customer turnover (Tang, 2020), assessing credit risk (Liu *et al*., 2022; Li *et al*., 2022), and aiding in diagnosis (Li & Zhang, 2020; Ogunleya & Wang, 2019).

Effectively addressing class imbalances, XGBoost uses a variety of specified hyperparameters and weight adjustments in the loss function (Wang *et al*., 2020). XGBoost has been applied to medical problems with class distribution. It has been used, for instance, to detect uncommon adverse medication reactions (Létinier *et al.*, 2021), diagnose diseases (Nichols *et al*., 2019), and manage a variety of datasets with class imbalance issues (Mishra *et al*., 2021; Le *et al*., 2022).

2.4.4 LightGBM. Ke et al. (2017) created LightGBM using tree-based learning methods to design a Gradient Boosting Machine (GBM). This model is favored for its speed and efficiency due to its name's implication of being lightweight. LightGBM is particularly useful for processing amounts of data thanks to its memory-friendly structure and capacity to manage extensive datasets; however, it may tend to overfit when dealing with smaller datasets due to its sensitivity. Determining the amount of data for a specific scenario is a complex task that does not have a clear cutoff point when effectively handling large datasets for accuracy and efficiency.

LightGBM builds its tree-based models step-by-step by adding a tree to the current ensemble to minimize the residual error from the current model iteration. Gradient-based optimization approaches train the tree on the loss function's negative gradient.

An essential aspect of LightGBM is its adoption of a histogram-based method for decision tree construction, making it particularly effective in handling extensive datasets and numerous features. Ke *et al*. (2017) introduced the Gradient-Based One Side Sampling (GOSS) method. GOSS selects a random portion of the data rather than using the complete dataset for each split in the tree construction process. When handling features, this strategy aids in preventing overfitting and enhancing computational efficiency.

The method of tree growth in LightGBM is significant compared to how other tree-based algorithms typically progress in their growth strategies. It differs from the level-based growth approach by employing a leaf-based method that utilizes histogram-based techniques to identify the optimal split point within a leaf. While this

technique does pose a risk of overfitting, it also enables accurate splits and enhances model performance, particularly in situations involving numerous trees. LightGBMs can be challenging due to the need to adjust hyperparameters carefully to achieve classification and regression analysis results.

In tasks and applications, including regression analysis and classification, LightGBMs are successful. It has been applied in tasks like natural language processing (Qin *et al*., 2021; Quinto, 2020), computer vision (Zeng *et al*., 2019; Sun *et al*., 2022) well as diverse fields, like finance (Wang *et al*., 2022; Sun *et al*., 2020) healthcare (Ghourabi, 2022) and marketing (Liang *et al*., 2019).

Machine learning models such as LightGBM, which are flexible in applying different goal functions and giving different weights to different classes within the data set, excel at handling datasets and the associated challenges. Because of this specific property, LightGBM is a popular option for various data distribution tasks, including fraud detection (Hu *et al*., 2019) and related applications (Huang, 2020).

2.5 Explainable Artificial Intelligence (XAI) and SHAP

Explainable Artificial Intelligence (XAI) methods allow us to understand how individual attributes affect model predictions. Using SHAP (SHapley Additive exPlanations), the importance of each attribute is assessed, including how it affects feature combination forecasts. These SHAP values offer information about the essential characteristics of the game theory guidelines (Lundberg & Lee, 2017). The SHAP framework ensures that the total contributions of features correspond to the model prediction, explaining how the model operates by exploring all possible combinations of features. SHAP reveals interactions among them and delivers a comprehensive evaluation of feature significance, established by Shapley (1953) and further discussed by Strumbelj and Kononenko (2014).

The power of SHAP lies in its ability to deliver insights at the level of model predictions, providing a detailed understanding of what each feature contributes to the predictions and highlighting the importance of features across the entire dataset when these local explanations are combined. SHAP is particularly useful for feature selection in machine learning models. Furthermore, Shapley's values derived from

cooperative game theory ensure a fair and reliable assignment of feature importance, enhancing the credibility and interpretability of the model's explanations.

2.6 Ensemble Approaches

Ensemble methods combine models or techniques to improve performance and robustness in feature selection tasks by integrating different approaches to generate a comprehensive ranking of feature importance for better decision-making processes.

Recent studies have explored ensemble approaches. Demir et al. (2021) proposed a hybrid model applied to birthday tweets, combining multiple feature selection methods and optimizing with ensemble pruning, leading to superior prediction accuracy compared to existing studies. Another study tested a mathematical model for pruning convolutional neural network ensembles using second-order conic optimization on different datasets, achieving promising accuracy and complexity reduction (Güldoğuş et al., 2023). In a recent study, the second-order cone programming algorithm for feature selection using second-order conic programming demonstrated its effectiveness on synthetic datasets by identifying key features and improving classification accuracy (Güldoğuş and Özöğür-Akyüz, 2024).

For instance, blending SHAP with Gradient Boost Machines and Random Forests can address biases in methodologies. Provide more reliable and interpretable rankings of feature importance. The ensemble framework proposed in this article seeks to leverage the capabilities of SHAP to create a feature selection procedure. By integrating XAI methods, this framework offers an insight into feature importance, thereby enhancing both the interpretability and accuracy of machine learning models (Dietterich, 2000; Rokach, 2010).

Various strategies are combined to increase the approach's dependability in selecting characteristics. Tree-based approaches prefer specific features, whereas SHAP offers a more balanced perspective by considering feature interactions from several perspectives. This ensemble method aggregates results from many methods to reduce biases in feature importance determination. This broad viewpoint benefits datasets with various feature selection strategies that may reveal different data features.

Chapter 3

Methodology

This section outlines the proposed approach in this study to develop and evaluate a robust feature selection system using various Explainable Artificial Intelligence (XAI) techniques and machine learning models. Phases include data preparation, model adjustments, feature importance determination using SHAP, and feature merging of critical features from several models. Each step enhances the models' precision and understandability, producing instructive outcomes.

3.1 Research Design

This study uses XAI methodologies and a variety of Machine Learning (ML) models. This research aims to improve the interpretability and predictive capability of models like Gradient Boosting, Random Forest, LightGBM, and XGBoost. The pre-processing of the data, model training, feature importance assessment using SHAP, a union of chosen features, and retraining of models based on these chosen features are all included in the research design.

3.2 Dataset

Three different datasets were used in this thesis study. Table 1 summarizes the datasets employed in this study, highlighting the number of features and instances for each dataset. These datasets were selected to explore various applications of machine learning in neuroscience, biomedical diagnostics, and text classification.

Table 1

Overview of Datasets Used in the Present Study

Data	Features	Instances
EEG Creativity Data (Ali *et al*, 2019)	130	10429
Breast Cancer Wisconsin (Diagnostic)	30	569
Spambase	57	4601

Dataset 1: Assessing the Effects of Creativity Training on the Default Mode Network and Attention

Ali *et al.* (2019) investigate the impact of creativity training on brain activity, specifically focusing on the Default Mode Network (DMN) and attention mechanisms. The research utilized EEG data collected from participants before and after a nine-week creativity training program. The dataset derived from this study includes EEG recordings, Power Spectral Density (PSD) measurements, and Event-Related Potentials (ERP) data, providing insights into neural changes associated with enhanced creativity.

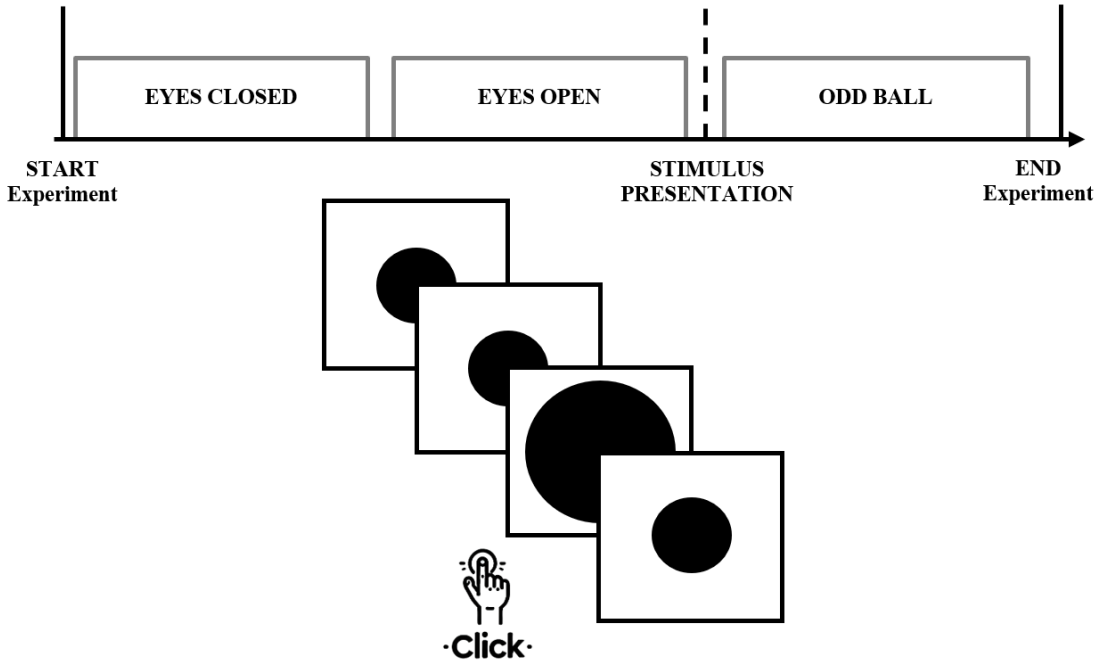

Figure 1. Experimental Paradigm and Oddball Test (Ali *et al*., 2019).

Dataset 2: Breast Cancer Wisconsin (Diagnostic)

The Breast Cancer Wisconsin (Diagnostic) dataset, created by the University of Wisconsin and available in the UCI Machine Learning Repository, is widely used in medical machine learning studies. Its primary purpose is to aid in distinguishing between malignant breast tumors using fine needle aspirate (FNA)samples from breast lumps for binary classification tasks. Each record in the dataset is described by 30 attributes derived from digitalized images of the FNA samples that portray different traits of the cell nuclei seen in the images.

Dataset 3: Spambase - UCI Machine Learning Repository

The Spambase dataset is available in the UCI Machine Learning Repository. It aims to classify emails as either spam or non-spam (referred to as ham). It comprises 4601 cases and 57 attributes encompassing textual and structural traits found in emails, such as word frequencies and character counts. These include patterns, like capital letter sequences, typically associated with spam messages.

3.3 Data Pre-Processing

Preparing the dataset for machine learning modeling requires an essential initial step, data pre-processing. When approaching this stage, relevant actions include addressing missing data, normalizing or standardizing features, and encoding variables if necessary. Subsequently, the dataset is split into training and test sets to ensure a model evaluation. This particular phase significantly impacts the effectiveness of the model. It establishes a strong foundation for subsequent model training.

3.4 The t-statistic

The t-statistic indicates the magnitude of an effect for a particular feature and, along with the p-value, helps determine the statistical significance of that effect. A high t-statistic (in absolute terms) and a low p-value suggest that the feature is crucial for the model. The p-value is used in hypothesis testing to understand whether the observed data could have come from a particular distribution. Generally, if the p-value

is below 0.05, the result is considered statistically significant. This analysis was conducted specifically for the EEG dataset.

3.5 Training Machine Learning Models

This research analyses models using XGBoost and popular machine-learning algorithms like LightGBM and Random Forest. The training data undergoes preprocessing before being used to train each model. After the training phase, the models are evaluated using precision, accuracy, recall, F1 score, and AUC-ROC to establish a baseline performance level. This evaluation is crucial before delving into SHAP analysis and selecting features, as it provides a benchmark for measuring the effects of feature selection.

3.6 Application of SHAP and Selection of Top Ten Features

Using SHAP and selecting the ten features are essential components of this process. After the models have been trained and fine-tuned, the next step involves applying SHAP to each model to evaluate the contribution of each feature to the model's predictions. The SHAP values are then computed for all features to rank them based on their significance in influencing the model's predictions. SHAP is a tool that offers clear insights into how models behave and excels at capturing interactions between different features in complex models. Following this analysis process across ML models, namely XGBoost, LightGBM Random Forest, and Gradient Boosting, helps identify and prioritize each model's top 10 most crucial features. The selection process focuses on keeping the significant characteristics while making the models more straightforward for better understanding and reducing complexity without sacrificing predictive accuracy.

3.7 Taking the Union of Important Features

After identifying the ten crucial features in each model and performing a union operation to merge them, all potentially beneficial features are included without any omissions. This process combines viewpoints from different models to generate an extensive and robust set of features that capitalize on each model's strengths. This

critical step enhances the final model's ability and underscores the importance of feature selection in boosting overall performance and reliability.

3.8 Retraining and Prediction with the Union Features

Once the key features are pinpointed collectively, machine learning models are enhanced by homing in selected features. The XGBoost, LightGBM, Random Forest Gradient Boosting, and Decision Tree models undergo retraining using features. Subsequently, the updated models are used to analyze the test data. A comparison is then drawn between the performances of these models.

3.9 Proposed Framework: SHAP-based feature selection and ensemble model training algorithm

To further understand the process outlined in this study, the algorithm's structure is detailed step by step. Each step is designed to clarify the methodology and provide a comprehensive understanding of the underlying logic. Additionally, the flowchart is provided in Figure 2.

Step 1: Data Collection and Pre-Processing

Along with the selection of the data to be worked on, the first step in preparing it for machine learning modeling is data preprocessing. The effectiveness of the model is greatly affected by this particular stage.

Step 2: Train Tree-Based Models

In the second step, tree-based ML models are trained on the dataset. These models are selected because of their inherent ability to handle large datasets, manage feature interactions, and provide robust predictions. During the training phase, the models learn between the input features and the target relationships, establishing a foundation for feature importance extraction.

Step 3: Obtain Feature Importance from Step 2

After training the tree-based models, extracting their feature importance scores is the next step. Tree-based models naturally provide feature importance as part of their structure, as they split the data based on the most informative features.

This step identifies the top 10 features that most significantly influence the model's predictions by analyzing how often and effectively each feature is used to make splits in the decision trees.

Step 4: Apply SHAP on the Trained Models in Step 2

Once feature importance is obtained, SHAP (SHapley Additive exPlanations) is applied to the trained models. SHAP provides a more profound understanding by explaining the contribution of each feature to individual predictions. Unlike traditional feature importance, which gives a general sense of importance, SHAP values show how much each feature influences the model's output for specific cases, thus offering a more granular view.

Step 5: Take the Union of Selected Features from Step 3 & Step 4

The most important features identified from the feature importance scores (Step 2) and the SHAP analysis (Step 3) are combined. The union of these selected features ensures that all relevant features are considered for the final model, integrating the strengths of both traditional feature importance and SHAP's interpretability. This union process helps form a comprehensive set of features crucial for the model's predictive power.

Step 6: Train and Predict with the Features in Step 5

Finally, the models are retrained using the selected features from the union process and then used to make predictions. Training the models with this refined set of features can enhance their performance by focusing on the most critical parameters, leading to better generalization and potentially higher prediction accuracy. This step validates the feature selection process's effectiveness by analyzing the new model's success with the original one. Additionally, the pseudocode for the proposed SHAP-based feature selection and ensemble model training algorithm is provided below in Algorithm 1.

Algorithm 1: Proposed Framework: SHAP-based feature selection and ensemble model training algorithm

Input: d: dataset, M: number of Tree-Based ML Models

Output: m: Best performing model, F: Corresponding best feature set

1: Collect and preprocess the data d for machine learning modeling
2: **for i** ← **1** to M **do**
3: Train tree-based model i on the preprocessed data d
4: Save the trained model i
5: **end for**
6: **for each** trained model i **do**
7: Extract feature importance scores from model i
8: Select the top 10 features based on importance scores
9: Save the selected features
10: **end for**
11: **for each** trained model i **do**
12: Apply SHAP analysis on model i to obtain SHAP values
13: Identify important features based on SHAP values
14: Save the important features
15: **end for**
16: Combine selected features from both feature importance and SHAP analysis
17: Create a union set of features F
18: Retrain models using the selected features F
19: Evaluate model performance on a validation set
20: Select the best-performing model m based on performance metrics
21: Save the best model m and corresponding feature set F

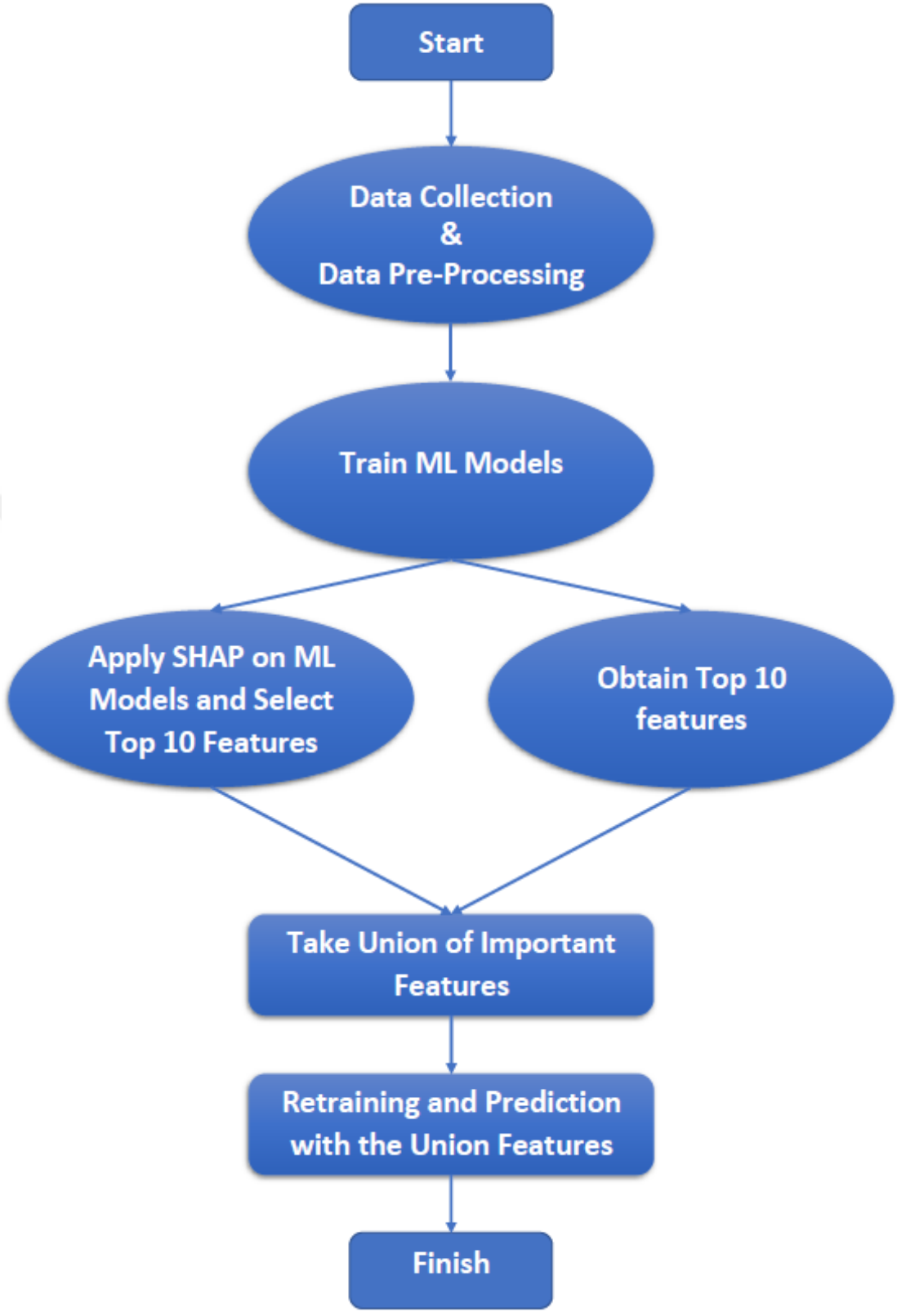

Figure 2. Flowchart of the Proposed Framework.

.

Chapter 4

Findings

This section focuses on analyzing the performance of machine learning models when applied to datasets such as the EEG Creativity dataset, Breast Cancer Wisconsin (Diagnostic), and Spambase. Different feature selection methods are employed to identify the best-performing model and approach in terms of accuracy, computational efficiency, and overall predictive capability. The models under consideration include LightGBM, XGBoost, Random Forest, Gradient Boosting, and Decision Tree. In evaluating the performance of each model and understanding the impact of feature selection methods across various datasets, key metrics such as accuracy, precision, recall, F1 score, receiver operating characteristic area under the curve (ROC AUC), and running time are utilized.

Notably, when measuring running time, only the training time of the model is considered in the without feature selection step, while the hybrid model, model feature, and SHAP feature steps account for the time taken to compute model feature importance and the time required to generate SHAP values.

4.1 Performance Results of ML Models on EEG Creativity Data without Feature Selection (130 Features)

In this dataset, many p-values are very small, indicating that many features have a highly significant contribution to the model (Figure 3). Therefore, data with small p-values should be given the most attention during the research or modeling process, as these features have a statistically strong contribution to the model. These features should be retained, while irrelevant features should be removed. This is because non-significant features (with high p-values) can unnecessarily occupy space in the model, increasing the risk of overfitting.

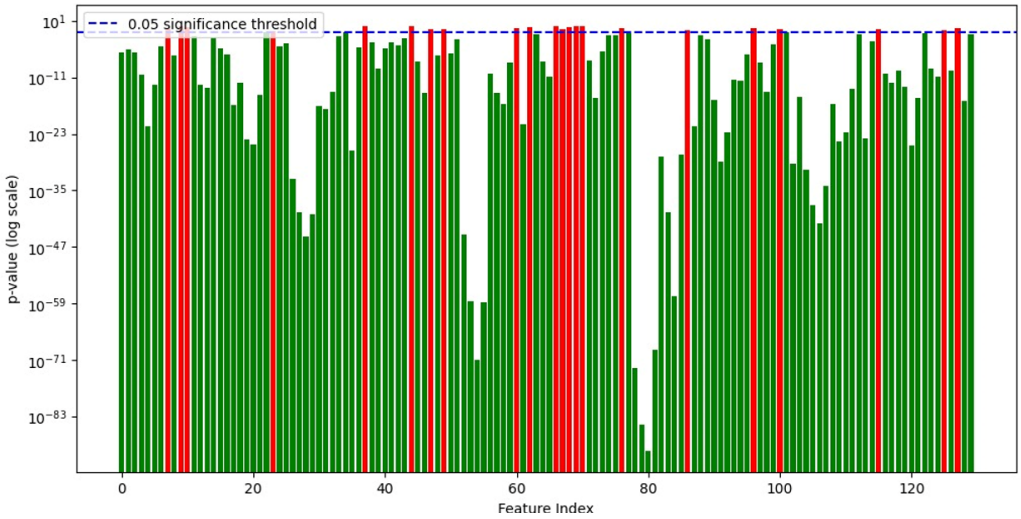

Figure 3. p-Values of Features with Significance Highlighted for EEG data.

Table 2 and Table 3 display the performance metrics of ML models trained on the EEG Creativity training and test dataset without undergoing any previous feature selection, respectively. Based on the test results presented in Table 3, the following evaluations have been made. LightGBM, XGBoost, Random Forest, Gradient Boosting, and Decision Tree are among the models that were assessed. AUC, F1 score, ROC AUC, accuracy, precision, recall, training duration, and other critical performance measures were all measured for each model, and the results were collated for easy comparison. Based on the data presented in Table 3, LightGBM, Random Forest, and XGBoost exhibit very similar performance across most metrics. Both LightGBM and Random Forest achieve an accuracy of 0.998, precision of 0.998, recall between 0.997 and 0.998, an F1 score of 0.998, and a perfect ROC AUC of 1.0. XGBoost also performs at a comparable level, with an accuracy of 0.997, precision of 0.998, recall of 0.997, and an F1 score of 0.997. Additionally, XGBoost has a faster running time (1.698 seconds) compared to LightGBM (2.173 seconds). Given these minimal differences, LightGBM may be slightly preferred due to its marginally higher accuracy (0.998) compared to XGBoost. In applications where high accuracy is critical, this minor advantage in accuracy could make LightGBM the optimal choice. However, in scenarios where speed is a priority, XGBoost would be a more suitable option due to its shorter running time. Thus, while LightGBM might be favored for applications that require the highest possible accuracy, XGBoost remains a strong alternative for use cases that demand faster model deployment.

Table 2

Training Results for EEG Data without Feature Selection: Performance Metrics

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Random Forest	0.999	0.999	0.999	0.999	1.000	7.449
Light GBM	0.999	0.999	0.999	0.999	1.000	2.173
XGBoost	0.999	0.999	0.999	0.999	1.000	1.698
Gradient Boosting	0.990	0.990	0.989	0.990	1.000	32.4
Decision Tree	0.983	0.983	0.982	0.982	0.999	2.234

Table 3

Test Results for EEG Data without Feature Selection: Performance Metrics

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Random Forest	0.998	0.998	0.998	0.998	1.000	7.449
Light GBM	0.998	0.998	0.997	0.998	1.000	2.173
XGBoost	0.997	0.998	0.997	0.997	1.000	1.698
Gradient Boosting	0.983	0.984	0.982	0.983	0.998	32.4
Decision Tree	0.946	0.946	0.943	0.944	0.943	2.234

The XGBoost model also did well and showed similar results to LightGBM in performance metrics like accuracy 0.997, precision 0.998, recall 0.997, and F1 score 0.997. It also had a ROC AUC of 1.0. Additionally, the XGBoost model was faster during training than LightGBM as it only took 1.698 seconds to train, which makes it the quickest model for this assessment. Given its performance and quick running time,

the XGBoost model would be an excellent choice for situations requiring high accuracy and fast training speeds.

Random Forest also delivered strong results, with an accuracy of 0.998, precision and recall at 0.998, and an F1 score of 0.998. The model achieved a perfect ROC AUC of 1.0, similar to the other top-performing models. However, its running time was significantly longer at 7.449 seconds, indicating a trade-off between model complexity and computational efficiency.

Gradient Boosting exhibited slightly lower performance metrics than the top three models, with an accuracy of 0.983, precision of 0.984, recall of 0.983, and an F1 score of 0.983. The ROC AUC was nearly perfect at 0.998, demonstrating strong classification capability. However, the running time for Gradient Boosting was the highest among the models, at 32.4 seconds, which may be a limiting factor for time-sensitive applications.

While achieving decent performance metrics, the decision tree model lagged behind the other models with an accuracy of 0.946, precision of 0.946, recall of 0.943, and an F1 score of 0.944. The ROC AUC was 0.943, which, while respectable, indicates that the model is less effective at distinguishing between classes than the others. However, its running time was the shortest after XGBoost and LightGBM, at 2.234 seconds, making it suitable for quick, less complex tasks.

The results highlight several important considerations when selecting machine learning models for EEG data analysis, particularly in creativity research. LightGBM and XGBoost emerged as the top-performing models, balancing high classification accuracy and efficiency in running time. These models are particularly suited for applications with critical precision and speed, such as real-time cognitive state monitoring or adaptive learning systems.

While highly accurate, random forests require more computational resources, which may limit their use in scenarios with constrained processing power or time-sensitive requirements. Gradient Boosting, although accurate, demonstrated the longest running time, suggesting it may be more appropriate for applications where model accuracy is prioritized over speed.

Decision Tree, while the simplest model, provides a baseline for comparison. Its lower performance metrics and shorter running time make it less suitable for complex

tasks but potentially valuable for scenarios where interpretability and quick model deployment are more critical than raw predictive power.

Overall, this analysis emphasizes how critical assessing a machine learning model's computational effectiveness and predictive accuracy is, especially when working with massive and complicated datasets like EEG recordings. The model selection should be based on the application's requirements, weighing interpretability, accuracy, and training duration to get the best results.

4.2 Performance Results of ML Models in the Proposed Framework on EEG Creativity Data (24 Hybrid Features)

Tables 4 and 5 present the performance values of machine learning models trained using a proposed framework with 24 hybrid features selected through feature selection methods on the EEG Creativity training and test datasets. The following evaluations have been made based on the test results shown in Table 5. LightGBM, XGBoost, Random Forest, Gradient Boosting, and Decision Tree are among the models that were assessed. The performance indicators were used to evaluate each model's performance, including accuracy, precision, recall, F1 score, ROC AUC, and running time.

LightGBM was the best-performing model with an accuracy of 0.997, precision of 0.997, recall of 0.996, and an F1 score of 0.997; additionally, the model's flawless ROC AUC score of 1.0 demonstrated its remarkable capacity for class distinction. With a comparatively short running time of 3.841 seconds, LightGBM is a viable and effective choice for applications that demand rapid deployment and excellent precision.

With a 0.996 F1 score, 0.996 accuracy, 0.996 precision, and 0.996 recall, XGBoost likewise demonstrated remarkable performance. Similar to LightGBM, XGBoost also attained a flawless 1.0 ROC AUC score. With a running time of 10.796 seconds, XGBoost offers relatively good speed, though it is not the fastest model. The Decision Tree, with a running time of 1.304 seconds, is the quickest. However, XGBoost is still suitable for applications where fast training is important, even though its running time is not among the best.

Table 4

Performance Results of the Proposed Framework for the EEG Training Data (Hybrid Features: 24)

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Light GBM	0.999	0.999	0.999	0.999	1.000	3.841
Random Forest	0.999	0.999	0.998	0.999	1.000	69.620
XGBoost	0.998	0.998	0.998	0.998	1.000	10.796
Decision Tree	0.986	0.986	0.985	0.985	1.000	1.304
Gradient Boosting	0.985	0.986	0.984	0.985	0.999	8.118

Table 5

Performance Results of the Proposed Framework for the Test Data (Hybrid Features: 24)

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Light GBM	0.997	0.997	0.996	0.997	1.000	3.841
XGBoost	0.996	0.996	0.996	0.996	1.000	10.796
Random Forest	0.995	0.995	0.995	0.995	1.000	69.620
Gradient Boosting	0.974	0.975	0.971	0.973	0.997	8.118
Decision Tree	0.956	0.956	0.953	0.954	0.953	1.304

Random Forest demonstrated strong performance, with an accuracy of 0.995, precision and recall at 0.995, and an F1 score of 0.995. The model also achieved a perfect ROC AUC of 1.0. However, the running time for Random Forest was

significantly longer at 69.620 seconds, indicating that while the model is highly accurate, it is less efficient regarding computational resources than LightGBM and XGBoost.

Gradient Boosting demonstrated solid performance with an accuracy of 0.974, precision of 0.975, recall of 0.971, and F1 score of 0.973, although it exhibited marginally inferior performance metrics compared to the top three models. The classification abilities were strong, with a ROC AUC of 0.997, which was almost perfect. Although Gradient Boosting has a relatively long running time of 8.118 seconds compared to some other models, it is not the longest. Random Forest, with a running time of 69.620 seconds, takes significantly longer. Therefore, while Gradient Boosting may not be ideal for time-sensitive applications, it is not entirely unsuitable.

As expected, the decision tree showed the lowest performance among the models evaluated, with an accuracy of 0.956, precision of 0.956, recall of 0.953, and an F1 score of 0.954. The ROC AUC was 0.953, indicating a lower capability to distinguish between classes than the others. However, the running time was the shortest at 1.304 seconds, positioning the Decision Tree as a viable option for applications where interpretability and quick deployment are prioritized, overachieving the highest predictive accuracy.

The outcomes demonstrate how well the suggested approach works to optimize model performance by selecting a set of 24 hybrid characteristics. The models that performed best were LightGBM and XGBoost, which combined good accuracy with quick running times. These models might work especially well when precision and speed are critical, such as in real-time applications.

While still highly accurate, Random Forest demonstrated longer running times, which might limit its application in time-sensitive contexts. However, it remains a strong candidate for tasks where accuracy is paramount and running time is less of a concern.

Gradient Boosting showed strong classification capabilities but at the cost of longer running times, making it more suitable for applications where model accuracy is prioritized over training efficiency.

While the Decision Tree was the least accurate, it provided the fastest running time. It is ideal for rapid prototyping or applications where model simplicity and quick deployment are more important than achieving the highest accuracy.

Overall analysis underscores the importance of feature selection in enhancing model performance and efficiency. By carefully selecting a set of hybrid features, the proposed framework allows for the development of models that are not only accurate but also computationally efficient, making them suitable for a broad range of applications.

4.3 Performance Results of ML Models on EEG Creativity Data Using Selected Important Features (10 Model Features)

Tables 6 and 7 present the performance metrics of various ML models trained using a selected set of 10 features determined by feature relevance on the EEG Creativity training and test datasets, respectively. The following evaluations are based on the test results shown in Table 7. LightGBM, Random Forest, XGBoost, Gradient Boosting, and Decision Tree are among the models that were assessed. The performance indicators were used to evaluate each model's performance, including accuracy, precision, recall, F1 score, ROC AUC, and running time.

With an accuracy of 0.990, precision of 0.989, recall of 0.990, and an F1 score of 0.990, LightGBM is the best-performing model. Additionally, the model's nearly flawless ROC AUC score of 0.999 demonstrated its remarkable capacity for class differentiation. With a training duration of only 0.239 seconds, LightGBM is an effective and potent choice for applications that demand rapid deployment and great accuracy.

Random Forest performed admirably with an accuracy of 0.973, precision of 0.971, recall of 0.973, and an F1 score of 0.972. The model demonstrated strong classification abilities with a high ROC AUC of 0.995. In contrast to LightGBM and XGBoost, Random Forest required a substantially extended training period (3.101 seconds), suggesting that the model is more computationally intensive even with its excellent accuracy.

With an accuracy of 0.973, precision of 0.972, recall of 0.972, and F1 score of 0.972, XGBoost outperformed Random Forest. Despite being somewhat lower at 0.997, the ROC AUC still shows high classification capability. With a running time of 0.252 seconds, XGBoost was able to outperform Random Forest and be a more time-efficient solution overall.

Gradient Boosting, while showing good performance metrics, had slightly lower values than the top three models, with an accuracy of 0.939, precision of 0.939, recall of 0.937, and an F1 score of 0.938. The ROC AUC was also lower at 0.983. Additionally, the running time for Gradient Boosting was the highest among the models at 3.206 seconds, which may limit its applicability in time-sensitive scenarios.

Table 6

Performance Results of ML Models Feature Importance for Training Data (Model Features: 10)

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Light GBM	0.997	0.997	0.997	0.997	1.000	0.239
XGBoost	0.992	0.992	0.992	0.992	1.000	0.252
Random Forest	0.990	0.989	0.989	0.989	1.000	3.101
Decision Tree	0.979	0.979	0.979	0.979	1.000	0.164
Gradient Boosting	0.956	0.955	0.954	0.954	0.993	3.206

Table 7

Performance Results of ML Models Feature Importance for the Test Data (Model Features: 10)

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Light GBM	0.990	0.989	0.990	0.990	0.999	0.239
XGBoost	0.973	0.972	0.972	0.972	0.997	0.252
Random Forest	0.973	0.971	0.973	0.972	0.995	3.101
Gradient Boosting	0.939	0.939	0.937	0.938	0.983	3.206
Decision Tree	0.926	0.925	0.924	0.925	0.924	0.164

As expected, the decision tree showed the lowest performance among the models evaluated, with an accuracy of 0.926, precision of 0.925, recall of 0.924, and an F1

score of 0.925. The ROC AUC was 0.924, indicating a lower capability to distinguish between classes than the other models. However, the running time was the shortest at 0.164 seconds, positioning the Decision Tree as a viable option for applications where interpretability and quick deployment are prioritized, overachieving the highest predictive accuracy.

The analysis shows the impact of focusing on a small, highly relevant feature set on the performance of machine learning models. LightGBM and XGBoost emerged as the most effective models, balancing high accuracy and efficient running times. These models are particularly well-suited for real-time applications or scenarios where both speed and accuracy are crucial.

While still highly accurate, Random Forest demonstrated longer running times, which might limit its application in time-sensitive contexts. However, it remains a strong candidate for tasks where accuracy is paramount and running time is less of a concern.

Gradient Boosting showed good classification capabilities but at the cost of longer running times, making it more suitable for applications where model accuracy is prioritized over training efficiency.

While the Decision Tree was the least accurate, it provided the fastest running time, making it ideal for rapid prototyping or applications where model simplicity and quick deployment are more critical than reaching maximizing accuracy.

Feature selection is vital for improving model performance and efficiency. By carefully selecting key features, the models sustained high classification accuracy and increased computational efficiency, making them versatile for various applications.

4.4 Performance Results of ML Models Using SHAP-Based Feature Selection on EEG Creativity Data (10 SHAP Features)

Tables 8 and 9 present the evaluation metrics of various ML models trained on the EEG Creativity training and test datasets. These models were developed using ten features selected based on SHAP (SHapley Additive exPlanations). The models assessed include LightGBM, XGBoost, Random Forest, Gradient Boosting, and Decision Tree. Each model's performance was evaluated using key metrics, including

accuracy, precision, recall, F1 score, ROC AUC, and training time. The following evaluations are based on the test results shown in Table 9.

The LightGBM model shows remarkable performance, achieving an accuracy, precision, recall, and F1 score of 0.983. The model also attained an exceptionally high ROC AUC score of 0.998, demonstrating its robust ability to distinguish across different groups. With a training duration of only 3.602 seconds, LightGBM is an effective and strong option for applications that demand high accuracy and fast deployment.

Table 8

Performance Results of SHAP for Training Data (SHAP Features: 10)

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Light GBM	0.995	0.995	0.995	0.995	1.000	3.602
XGBoost	0.992	0.991	0.991	0.991	1.000	5.783
Random Forest	0.989	0.989	0.989	0.989	0.999	62.298
Decision Tree	0.981	0.981	0.980	0.981	0.997	0.926
Gradient Boosting	0.955	0.954	0.954	0.954	0.991	3.349

Table 9

Performance Results of SHAP for Test Data (SHAP Features: 10)

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Light GBM	0.983	0.983	0.982	0.983	0.998	3.602
XGBoost	0.972	0.972	0.971	0.971	0.996	5.783
Random Forest	0.970	0.969	0.970	0.970	0.996	62.298
Gradient Boosting	0.943	0.943	0.941	0.942	0.983	3.349
Decision Tree	0.932	0.931	0.929	0.93	0.929	0.926

XGBoost achieved strong performance, with accuracy, precision, recall, and an F1 score equal to 0.972. The model demonstrated strong classification capabilities with a high ROC AUC score of 0.996. The training period of XGBoost was marginally more extended than that of LightGBM, taking 5.783 seconds. However, it remained highly efficient, rendering it ideal for real-time applications.

The Random Forest model exhibited excellent performance, achieving an accuracy of 0.97, precision of 0.969, recall of 0.97, and an F1 score of 0.97. ROC AUC was exceptionally high, measuring 0.996, indicating the model's strong ability to classify data accurately. Nevertheless, the duration of the training process was extended to 62.298 seconds, potentially constraining its use in time-critical situations where computational efficiency is of the utmost importance.

Gradient Boosting performed slightly worse than the leading models, achieving an accuracy of 0.943, precision of 0.943, recall of 0.941, and an F1 score of 0.942. The ROC AUC value of 0.983 indicates a satisfactory level of classification performance. Although its results show lower performance compared to other boosting models, its training time is reasonable and makes it suitable for applications requiring quick model deployment.

Decision Tree showed the lowest performance among the models evaluated, with an accuracy of 0.932, precision of 0.931, recall of 0.929, and an F1 score of 0.93. The ROC AUC was 0.929, indicating the model's lower capability to distinguish between classes than the others. However, the Decision Tree had the shortest running time at 0.926 seconds, making it a viable option for situations where interpretability and fast training are prioritized overachieving the highest predictive accuracy.

The results showed the effectiveness of SHAP-based feature selection in optimizing model success. LightGBM and XGBoost emerged as the most effective models, balancing high accuracy and efficient running times. These models are particularly well-suited for real-time applications or scenarios where both speed and accuracy are crucial.

Random Forest demonstrated strong classification capabilities but required longer running time, making it less suitable for time-sensitive applications. However, it remains a solid choice for scenarios where accuracy is prioritized over training efficiency.

While less accurate, the decision tree provided the fastest running time, making it ideal for rapid prototyping or applications where model simplicity and quick deployment are more important than achieving the highest accuracy.

Overall, the analysis shows the importance of feature selection on model performance and efficiency. By carefully selecting relevant features using SHAP, models can maintain strong classification power while improving computational efficiency.

4.5 Comparative Analysis of the Studies for Test Data

The comparative analysis presented in Table 10 evaluates the performance of various machine learning models on the EEG Creativity dataset using different feature selection methods. It identifies the best-performing model in each scenario and determines the overall most effective approach.

Table 10

Comparative Analysis of Performance Results for Test Data

Feature Selection Type	Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Without Feature Selection (F:130)	Random Forest	0.998	0.998	0.998	0.998	1.000	7.449
Hybrid Features (F:24)	Light GBM	0.997	0.997	0.996	0.997	1.000	3.841
Model Features (F:10)	Light GBM	0.990	0.989	0.990	0.990	0.999	0.239
SHAP Features (F:10)	Light GBM	0.983	0.983	0.982	0.983	0.998	3.602

Although Random Forest performed quite well, when compared to LightGBM's reasonable training time, LightGBM delivers better results when both factors are considered. Using 24 features with the Hybrid Feature Selection Framework, LightGBM delivered nearly perfect classification results while maintaining efficient running times. This combination emerged as the most effective and balanced solution.

Regardless of the feature selection method, LightGBM consistently outperformed other models in this dataset and for the tasks evaluated. While high accuracy was achieved on the training data, there was a slight decrease in performance on the test data. Specifically, with the SHAP-based feature selection, LightGBM's accuracy dropped from 0.995 in the training set to 0.983 in the test set. This minor decline highlights an important aspect of the model's generalization capability.

In conclusion, the combination of the Hybrid Feature Selection Framework and LightGBM provides a strong balance of accuracy, computational efficiency, and classification power. The small performance difference between the training and test datasets is a significant factor to consider, as it reflects the model's ability to generalize beyond the training data.

4.6 Feature Importance Scores for Model Performance

The graph in Figure 4 shows the importance of different features in Decision Tree, Gradient Boosting, LightGBM, Random Forest, and XGBoost models on the EEG dataset. The scores related to the importance of features in each model help to understand which features the model relies on more heavily when making decisions.

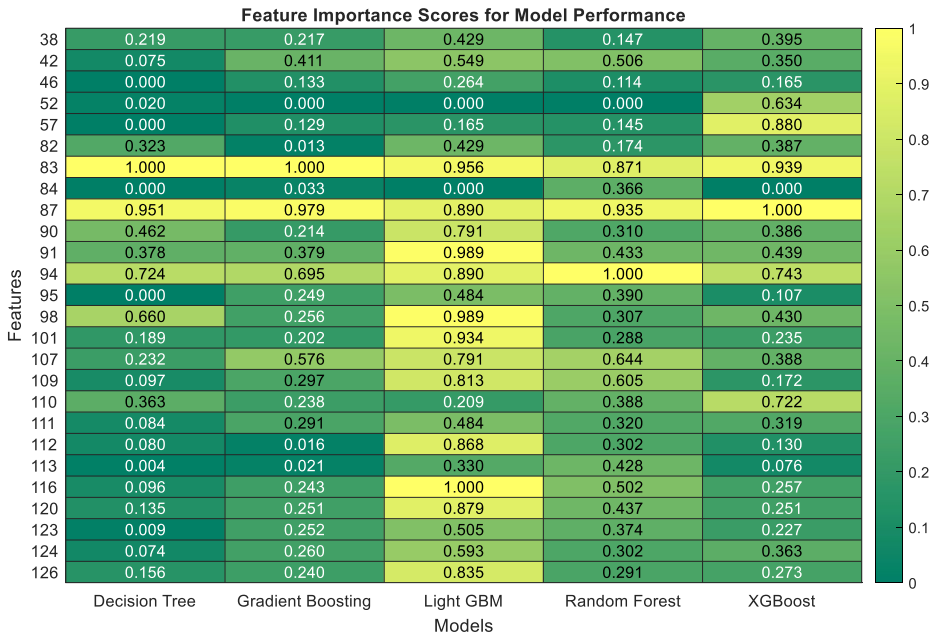

Figure 4. Feature Importance Scores for Model Performance on the EEG Dataset.

When examining the overall distribution of feature importance, it is observed that XGBoost and LightGBM models have high importance scores for many features. Specifically, "feature 83" and "feature 87" have high importance across almost all models. This suggests that these two features play a critical role in enabling the models to perform accurate classification.

On the other hand, when comparing models, it is evident that Gradient Boosting and XGBoost models provide more balanced and higher scores across most features. These models generally use more advanced optimization techniques and adapt better to the data. In contrast, models such as Decision Tree and Random Forest tend to focus more on specific features (e.g., "feature 83").

4.7 SHAP Values for Model Performance

SHAP values enhance the explainability of the decisions made by models. For each feature, the SHAP value explains how much that feature contributes to the model's output. In Figure 5, the effect of SHAP values on model performance is visually presented.

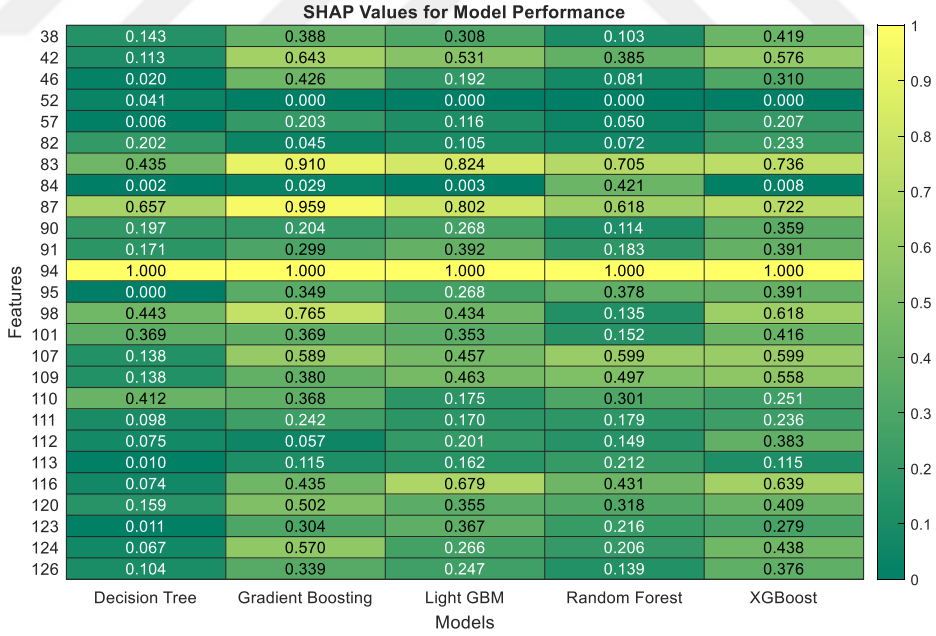

Figure 5. SHAP Values for Model Performance on the EEG Dataset.

Similar to the feature importance score evaluation, "feature 83" and "feature 87" stand out as the features that contribute the most according to SHAP values. This

indicates that the model relies more heavily on these two features when producing results. Particularly, XGBoost and LightGBM models have produced higher scores according to SHAP values.

SHAP values are extremely useful for understanding how each feature impacts the model and which features the model relies on more to make decisions. For instance, "feature 94" has a high SHAP value across all models, indicating that this feature significantly affects model performance.

In conclusion, both feature importance scores and SHAP values suggest that "feature 83", "feature 87", and "feature 94" are critical features for the models. These features were found to contribute significantly to the classification accuracy of the models on the EEG dataset. Additionally, XGBoost and Gradient Boosting models generally produced higher and more balanced results in both feature importance and SHAP value evaluations, demonstrating better performance on the EEG dataset.

4.8 Performance Results of ML Models on Spambase Dataset without Feature Selection (57 Features)

Tables 11 and 12 present the performance metrics of several ML models trained on the Spambase dataset without applying feature selection. The evaluated models include Random Forest, XGBoost, LightGBM, Gradient Boosting, and Decision Tree. These models were assessed based on accuracy, precision, recall, F1 score, ROC AUC, and running time using the test results.

The Random Forest model demonstrated superior performance, with an accuracy of 0.954, precision of 0.953, recall of 0.952, and an F1 score of 0.952. The model achieved a high ROC AUC of 0.987, demonstrating its robust capability to distinguish between spam and non-spam emails accurately. The Random Forest model has a running time of 0.512 seconds, which is relatively low given its complex nature and good performance. Random Forest is a suitable option for this dataset as it balances accuracy and efficiency.

XGBoost demonstrated excellent performance, achieving an accuracy of 0.951, precision of 0.951, recall of 0.949, and an F1 score of 0.95. The model attained a ROC AUC of 0.987, equivalent to the Random Forest model. This suggests that the model can effectively differentiate between spam and non-spam emails. The primary benefit

of XGBoost is its rapid running time of 0.343 seconds, surpassing that of Random Forest. This attribute renders it particularly well-suited for applications prioritizing efficiency and swift training.

Table 11

Performance Results without Feature Selection (Features: 57) for Train Spambase Data

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Random Forest	0.986	0.986	0.985	0.986	0.999	0.512
XGBoost	0.983	0.983	0.982	0.982	0.999	0.343
Light GBM	0.982	0.982	0.981	0.982	0.998	0.26
Decision Tree	0.971	0.971	0.970	0.970	0.996	0.037
Gradient Boosting	0.956	0.955	0.952	0.954	0.987	1.032

Table 12

Performance Results without Feature Selection (Features: 57) for Test Spambase Data

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Random Forest	0.954	0.953	0.952	0.952	0.987	0.512
XGBoost	0.951	0.951	0.949	0.95	0.987	0.343
Light GBM	0.946	0.945	0.944	0.944	0.987	0.26
Gradient Boosting	0.945	0.945	0.941	0.943	0.983	1.032
Decision Tree	0.920	0.918	0.917	0.917	0.919	0.037

LightGBM demonstrated strong performance, achieving an accuracy of 0.946, precision of 0.945, recall of 0.944, and an F1 score of 0.944. The Receiver Operating

ROC AUC for LightGBM was 0.987, comparable to the performance of Random Forest and XGBoost. This suggests that LightGBM is highly proficient at accurately identifying the data. The LightGBM model has the relatively short running time of only 0.26 seconds among the examined models, making it very efficient for scenarios that demand fast model deployment.

Gradient Boosting achieved somewhat lesser performance than the top three models, with an accuracy of 0.945, precision of 0.945, recall of 0.941, and an F1 score of 0.943. The model's ROC AUC was 0.983, which, although still robust, is marginally lower than that of Random Forest, XGBoost, and LightGBM. Gradient Boosting's primary limitation is its lengthy training period of 1.032 seconds, which is the longest among all models in our study. Our extended duration may restrict its use in time-sensitive applications.

The Decision Tree model demonstrated the poorest performance compared to the other models, with an accuracy of 0.92, precision of 0.918, recall of 0.917, and an F1 score of 0.917. The ROC AUC value for the Decision Tree model was 0.919, much lower than the other models. This suggests that the Decision Tree model is less proficient in accurately classifying the dataset. Nevertheless, the running time of the model was remarkably rapid, taking only 0.037 seconds. This characteristic renders it appropriate for swift and uncomplicated applications, prioritizing speed over utmost precision.

The Random Forest algorithm demonstrated superior performance across various metrics, including accuracy, precision, recall, and F1 score. Additionally, it exhibited a robust ROC AUC value of 0.987. Despite having a slightly longer running time than XGBoost and LightGBM, this model is still considered the best for this dataset because of its higher classification capability.

Although LightGBM slightly trails behind Random Forest and XGBoost in certain performance metrics, it compensates with a faster running time of 0.26 seconds. This feature makes it the most optimal model regarding computational resources and speed, making it perfect for situations where rapid deployment is essential.

XGBoost provides a robust equilibrium between efficiency and running time, exhibiting a significantly quicker training duration than Random Forest while maintaining similar performance measures. This option optimizes time-critical tasks requiring precise measurements and swift model development.

Although the Decision Tree model was the least accurate of the other models, its remarkably quick running time makes it suitable for applications where speed is more important than precision.

After analyzing the Spam & Not Spam dataset without feature selection, it is evident that Random Forest is the top performance in all aspects, especially where accuracy and precision are paramount. LightGBM and XGBoost exhibit high performance, showcasing exceptional speed and efficiency, rendering them appropriate for real-time applications. Gradient Boosting is a highly efficient method, but it may not be the best choice for time-sensitive tasks because it takes longer to train. Although Decision Tree is less precise, it offers the advantage of the fastest running time, which makes it suitable for simpler and less complex tasks.

4.9 Performance Results of ML Models in the Proposed Framework on Spambase Dataset (17 Hybrid Features)

Tables 13 and 14 present the performance metrics of various ML models trained on the Spam & Not Spam dataset using a proposed framework that employs 17 hybrid features. The models evaluated on the test data include LightGBM, Random Forest, XGBoost, Gradient Boosting, and Decision Tree. These models were assessed using metrics such as accuracy, precision, recall, F1 score, ROC AUC, and training time.

LightGBM achieved a strong accuracy of 0.947, matching the performance of Random Forest. Additionally, it demonstrated high precision (0.946), recall (0.945), and an F1 score (0.946), making it a top contender in this evaluation. With an ROC AUC of 0.984, LightGBM showcased its robust capability to distinguish between spam and non-spam emails. While its training time of 2.849 seconds is reasonable, it is longer compared to models like XGBoost and Gradient Boosting. Despite not being the fastest for rapid computation tasks, it remains a strong contender in terms of accuracy.

Random Forest achieved comparable results to LightGBM in terms of accuracy (0.947) and other measures, including precision (0.948), recall (0.943), and F1 score (0.945). The model also attained a commendable ROC AUC of 0.982. Nevertheless, Random Forest necessitated a lengthier training duration of 41.097 seconds, which,

although not excessive, is nevertheless greater than that of LightGBM and XGBoost. Random Forest is a reliable option when there is a need for high accuracy, although it is somewhat less efficient in terms of running time.

XGBoost demonstrated robust performance, with an accuracy of 0.943, precision of 0.943, recall of 0.939, and an F1 score of 0.941. The Receiver Operating Characteristic Area under the Curve (ROC AUC) was 0.983, similar to the ROC AUC values of LightGBM and Random Forest. The running time of XGBoost was a mere 0.795 seconds, indicating its exceptional efficiency as a model. This makes it particularly well-suited for applications where training speed and good performance are crucial.

Gradient Boosting performed slightly worse than the top three models, with an accuracy of 0.934, precision of 0.934, recall of 0.930, and an F1 score of 0.932. The receiver operating characteristic area under the curve (ROC AUC) was 0.981, indicating a high level of performance. However, the model took 1.023 seconds to train, which is longer than most models except for Random Forest. This suggests that although Gradient Boosting is efficient, it may not be ideal for situations where training time is of utmost importance.

The Decision Tree model had the poorest performance metrics compared to the other models, with an accuracy of 0.907, precision of 0.905, recall of 0.903, and an F1 score of 0.904. The ROC AUC was 0.904, which was noticeably inferior to the other models. Nevertheless, the Decision Tree algorithm exhibited the quickest running time, clocking in at a mere 0.407 seconds. This characteristic renders it well-suited for jobs that prioritize swift training over attaining the utmost accuracy.

LightGBM is the top performer in this scenario, with superior accuracy, outstanding classification scores, and quick running time. This makes it the optimal selection for applications requiring accuracy and swiftness. Random Forest and LightGBM have comparable accuracy, but Random Forest has a longer running time. It is a robust option for tasks prioritizing accuracy above computing efficiency.

XGBoost provides a well-balanced approach, delivering impressive performance metrics and economical running time. This makes it particularly well-suited for applications that require fast model training and deployment.

Gradient Boosting exhibits superior performance but has a longer running time, rendering it more suitable for situations where precision precedes training speed.

The Decision Tree algorithm, albeit less precise, offers the advantage of the shortest training period. This makes it particularly suitable for situations that require quick prototyping or prioritize model simplicity and speed, overreaching the utmost accuracy.

Table 13

Performance Results of the Proposed Framework (Hybrid Features: 17) for Train Spambase Data

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Light GBM	0.982	0.982	0.981	0.981	0.997	2.849
Random Forest	0.982	0.982	0.981	0.981	0.997	41.097
XGBoost	0.979	0.980	0.978	0.979	0.997	0.795
Decision Tree	0.974	0.974	0.973	0.973	0.996	1.023
Gradient Boosting	0.948	0.947	0.944	0.945	0.987	0.407

Table 14

Performance Results of the Proposed Framework (Hybrid Features: 17) for Test Spambase Data

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Light GBM	0.947	0.946	0.945	0.946	0.984	2.849
Random Forest	0.947	0.948	0.943	0.945	0.982	41.097
XGBoost	0.943	0.943	0.939	0.941	0.983	0.795
Gradient Boosting	0.934	0.934	0.93	0.932	0.981	1.023
Decision Tree	0.907	0.905	0.903	0.904	0.904	0.407

Analyzing the Spam & Not Spam dataset with the hybrid feature selection framework shows that LightGBM is the most efficient model. It ensures high accuracy and is also efficient in training duration. Random Forest and XGBoost demonstrate strong performance, with Random Forest exhibiting exceptional accuracy and

XGBoost showcasing superior efficiency. Gradient Boosting, while offering moderate accuracy, may not be the most suitable choice for time-sensitive tasks due to its extended training period of 1.023 seconds. Given its lower accuracy compared to models like LightGBM and Random Forest, which also have shorter training times, Gradient Boosting may not be ideal for scenarios where both accuracy and speed are critical. On the other hand, the Decision Tree is the fastest model available, although it sacrifices some accuracy.

4.10 Performance Results of ML Models on Spambase Dataset Using Selected Important Features (10 Model Features)

Tables 15 and 16 present the performance values of various machine learning models trained on the Spam & Not Spam training and test datasets, using ten features identified as important by the models. The evaluated models include Gradient Boosting, Random Forest, LightGBM, Decision Tree, and XGBoost, with the test results based on Table 16. The key metrics assessed are accuracy, precision, recall, F1 score, ROC AUC, and running time.

Gradient Boosting achieved the highest overall performance with an accuracy of 0.926, precision of 0.927, recall of 0.921, and an F1 score of 0.924. The model also had a strong ROC AUC of 0.977, indicating its effectiveness in distinguishing between spam and non-spam emails. The running time for Gradient Boosting was 0.376 seconds, which, while not the fastest, is reasonable given its performance metrics. Gradient Boosting's balance of accuracy and reliability makes it a solid choice for this dataset.

Random Forest performed very well, with an accuracy of 0.922, precision of 0.922, recall of 0.917, and an F1 score of 0.919. The ROC AUC was 0.972, slightly lower than that of Gradient Boosting but still robust. Random Forest’s running time was 0.388 seconds, making it a competitive model for performance and training efficiency. This model is well-suited for tasks requiring high accuracy with a moderate running time.

LightGBM demonstrated good performance with an accuracy of 0.915, precision of 0.915, recall of 0.91, and an F1 score of 0.912. The ROC AUC was 0.968, indicating strong classification capability. LightGBM’s running time was notably low at 0.124

seconds, making it the most efficient among the top performers in terms of running time. LightGBM is a good choice for applications where quick training is essential to maintaining decent accuracy.

Table 15

Performance Results of ML Models Feature Importance (Model Features:10) for Train Spambase Data

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Random Forest	0.977	0.978	0.976	0.977	0.997	0.388
Decision Tree	0.971	0.971	0.969	0.970	0.996	0.018
Light GBM	0.968	0.969	0.965	0.967	0.996	0.124
Gradient Boosting	0.938	0.938	0.932	0.935	0.983	0.376
XGBoost	0.935	0.937	0.927	0.931	0.981	0.078

Table 16

Performance Results of ML Models Feature Importance (Model Features:10) for Test Spambase Data

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Gradient Boosting	0.926	0.927	0.921	0.924	0.977	0.376
Random Forest	0.922	0.922	0.917	0.919	0.972	0.388
Light GBM	0.915	0.915	0.91	0.912	0.968	0.124
Decision Tree	0.907	0.906	0.904	0.905	0.902	0.018
XGBoost	0.907	0.909	0.899	0.903	0.964	0.078

Decision Tree had the fastest running time at 0.018 seconds, but this speed came at the cost of lower accuracy, which was 0.907. Precision, recall, and F1 scores were also the lowest among the models, with values of 0.906, 0.904, and 0.905, respectively. The ROC AUC was 0.902, the lowest in this comparison, indicating that Decision Tree

may struggle more with correctly classifying the data than other models. However, due to its swift running time, it is well-suited for simple or urgent tasks where speed outweighs accuracy. XGBoost showed balanced performance, achieving an accuracy of 0.907, precision of 0.909, recall of 0.899, and an F1 score of 0.903. The ROC AUC was 0.964, which is lower than Gradient Boosting and Random Forest but still indicates good classification ability. XGBoost's running time was 0.078 seconds, the second fastest in this evaluation, making it an efficient choice for applications requiring rapid training and reasonable performance.

Gradient Boosting emerged as the best overall performer in this scenario, offering the highest accuracy and superior classification metrics. It is particularly suitable for tasks where accuracy is the primary concern, and slightly longer running times can be tolerated.

Random Forest provides high accuracy and balanced performance metrics, with a competitive running time, though not the fastest. It is ideal for applications where accuracy is essential, but running time can be moderately flexible.

LightGBM, while slightly behind Gradient Boosting and Random Forest in accuracy, offers the best training efficiency with a very low running time. It is suitable model for applications where rapid training and deployment are critical.

A Decision Tree with the fastest running time is ideal for scenarios requiring rapid prototyping or when the model's simplicity is more important than achieving the highest accuracy.

The Spam & Not Spam dataset analysis using model-selected important features highlights Gradient Boosting as the best overall accuracy and classification power performer. Random Forest and LightGBM also perform well, with LightGBM excelling in training efficiency. Decision Tree is the fastest model with lower accuracy and is best suited for more straightforward tasks. XGBoost offers a balanced approach, making it a versatile option for various applications.

4.11 Performance Results of ML Models Using SHAP-Based Feature Selection on Spambase Dataset (10 SHAP Features)

Tables 17 and 18 present the performance metrics of various machine learning models trained on the Spam & Not Spam training and test datasets using 10 features

selected based on SHAP (SHapley Additive exPlanations). The models evaluated on the test data, as shown in Table 18, include Random Forest, LightGBM, Gradient Boosting, XGBoost, and Decision Tree. These models were assessed using accuracy, precision, recall, F1 score, ROC AUC, and running time as key metrics.

Table 17

Performance Results of ML Models Feature Importance (SHAP Features: 10) for Train Spambase Data

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Random Forest	0.977	0.978	0.976	0.977	0.997	41.336
Decision Tree	0.971	0.971	0.969	0.970	0.997	0.407
XGBoost	0.968	0.970	0.965	0.967	0.996	0.602
Light GBM	0.967	0.968	0.963	0.966	0.995	2.734
Gradient Boosting	0.938	0.938	0.932	0.935	0.983	0.503

Table 18

Performance Results of ML Models Feature Importance (SHAP Features: 10) for Test Spambase Data

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Random Forest	0.932	0.931	0.928	0.93	0.974	41.336
Light GBM	0.926	0.926	0.921	0.924	0.974	2.734
Gradient Boosting	0.925	0.926	0.921	0.923	0.974	0.503
XGBoost	0.92	0.92	0.916	0.918	0.971	0.602
Decision Tree	0.913	0.911	0.91	0.911	0.907	0.407

Random Forest achieved the highest overall performance in this scenario, with an accuracy of 0.932, precision of 0.931, recall of 0.928, and an F1 score of 0.93. The model also had a strong ROC AUC of 0.974, indicating its effectiveness in

distinguishing between spam and non-spam emails. However, the running time for Random Forest was 41.366 seconds, which is higher than other models, especially LightGBM and XGBoost. Despite the longer running time, Random Forest's strong classification performance makes it the best model in this evaluation.

LightGBM showed competitive performance with an accuracy of 0.926, precision of 0.926, recall of 0.921, and an F1 score of 0.924. The ROC AUC was 0.974, matching Random Forest, indicating that LightGBM is also very capable in classifying the data. The standout feature of LightGBM in this scenario is its running time, which was notably low at 2.734 seconds. This makes LightGBM more efficient model regarding running time, especially when quick deployment is crucial.

Gradient Boosting performed well, with an accuracy of 0.925, precision of 0.926, recall of 0.921, and an F1 score of 0.923. The ROC AUC was 0.974, indicating powerful classification capabilities. However, the running time for Gradient Boosting was 0.503 seconds, which, while faster than Random Forest and XGBoost, is significantly longer than that of Decision Tree. Gradient Boosting is a strong contender when high classification accuracy is prioritized over training speed.

XGBoost demonstrated good performance with an accuracy of 0.920, precision of 0.920, recall of 0.916, and an F1 score of 0.918. The ROC AUC was 0.971, slightly lower than the top models but still substantial. XGBoost's running time was 0.602 seconds, making it an efficient model for scenarios where a balance between quick training and decent accuracy is required.

Decision Tree had the fastest running time at 0.407 seconds, but this speed came at the cost of lower accuracy, with a value of 0.913. Precision, recall, and F1 scores were also the lowest among the models, with values of 0.911, 0.91, and 0.911, respectively. The ROC AUC was 0.907, the lowest in this comparison, indicating that Decision Tree may struggle more with correctly classifying the data than the other models. However, its speedy running time makes it suitable for simple or time-critical tasks where speed is prioritized over accuracy.

Random Forest is the best overall performer in this scenario, offering the highest accuracy, good classification metrics, and a robust ROC AUC. However, it has a higher running time, making it suitable for applications where accuracy is prioritized over speed.

LightGBM is the most efficient model in this comparison. Its excellent balance of strong performance metrics and very low running time makes it ideal for applications where quick training and deployment are critical without a significant sacrifice in accuracy.

The Spam & Not Spam dataset analysis using SHAP-based feature selection reveals that Random Forest is the best overall performer in accuracy and classification metrics. However, it requires a longer running time. LightGBM excels in training efficiency, making it highly suitable for rapid deployment scenarios. Gradient Boosting offers moderate classification power while achieving it with a short running time. At the same time, Decision Tree provides the fastest training, making it ideal for more straightforward tasks where speed is paramount. XGBoost strikes a good balance, making it versatile for various applications.

4.12 Comparative Analysis of the Studies for Test Data

Table 19 presents performance metrics for various models with different feature selection approaches. Random Forest without feature selection (F: 57) achieves the highest accuracy (0.954) and a strong ROC AUC (0.987) with a low running time (0.512 seconds), showing efficiency even with all features. LightGBM with hybrid features (F: 17) maintains good accuracy (0.947) and a high ROC AUC (0.984), but with a moderate running time (2.849 seconds), making it suitable for balanced performance and efficiency.

Random Forest without feature selection achieved the highest accuracy (0.954) and ROC AUC (0.987) with a relatively low running time (0.512 seconds), showing it performs well even without feature reduction. However, when SHAP-based feature selection was applied, although Random Forest still achieved reasonable accuracy (0.932) and ROC AUC (0.974), the running time significantly increased to 41.336 seconds. This increase in running time is due to the computational time required to generate SHAP values, which is more time-consuming.

Gradient Boosting with model-selected features (F: 10) delivers the lowest accuracy (0.926) but offers a fast-running time (0.376 seconds), making it ideal for quick deployment in scenarios where accuracy can be slightly compromised.

Table 19

Comparative Analysis of Performance Results for Test Data

Feature Selection Type	Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Without Feature Selection (F:57)	Random Forest	0.954	0.953	0.952	0.952	0.987	0.512
Hybrid Features (F:17)	Light GBM	0.947	0.946	0.945	0.946	0.984	2.849
SHAP Features (F:10)	Random Forest	0.932	0.931	0.928	0.93	0.974	41.336
Model Features (F:10)	Gradient Boosting	0.926	0.927	0.921	0.924	0.977	0.376

4.13 Feature Importance Scores for Model Performance

The visual in Figure 6 shows the importance of features in different models (Decision Tree, Gradient Boosting, LightGBM, Random Forest, XGBoost) on the Spambase dataset. According to the analysis results, the feature "word freq remove" has very high importance scores for the Decision Tree, Gradient Boosting, LightGBM, Random Forest, and XGBoost models. This suggests that this feature plays a critical role in the models' decision-making process during the classification task on the Spambase dataset. It has been observed that the LightGBM and Random Forest models have high importance scores for many features. Moreover, the Decision Tree model gives more weight to a few specific features like "word freq remove" and "capital run length total," while other models tend to approach the distribution of feature importance more evenly. Features such as "word freq free," "word freq hp," and "capital run length average" are also highly important, particularly for the Gradient Boosting and LightGBM models. These features have been evaluated as contributing significantly to the decision-making process of the models, especially in spam detection.

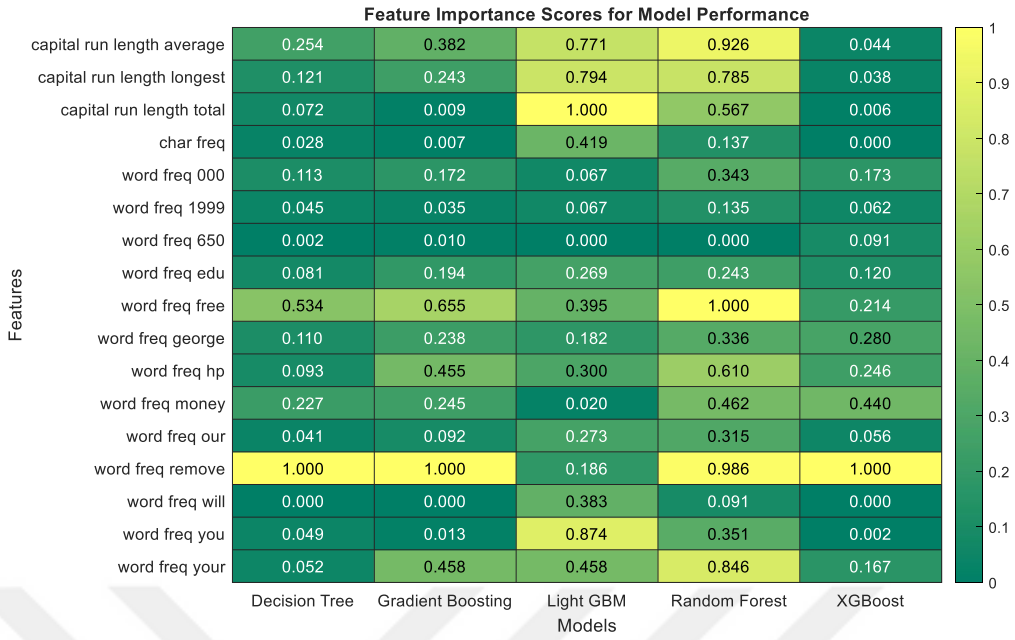

Figure 6. Feature Importance Scores for Model Performance on the Spambase Dataset.

4.14 SHAP Values for Model Performance

The SHAP impact assessment, which shows how much each feature influences the model's decision, is presented in Figure 7. The feature 'word freq remove' also stands out in SHAP values as one of the contributing features across most models, particularly in Random Forest and XGBoost. This demonstrates that the model uses this feature when making spam detection decisions. When examining SHAP values, the XGBoost and Random Forest models present higher and more balanced SHAP scores, indicating that these models provide more accurate and interpretable results. Features such as "word freq free," "word freq george," and "word freq hp" also show a significant impact in SHAP values. According to SHAP values, it is evident that Gradient Boosting and XGBoost models work more transparently. The influence of features like "word freq remove" is clearly visible in these charts, and these features have been evaluated as playing an important role in spam detection.

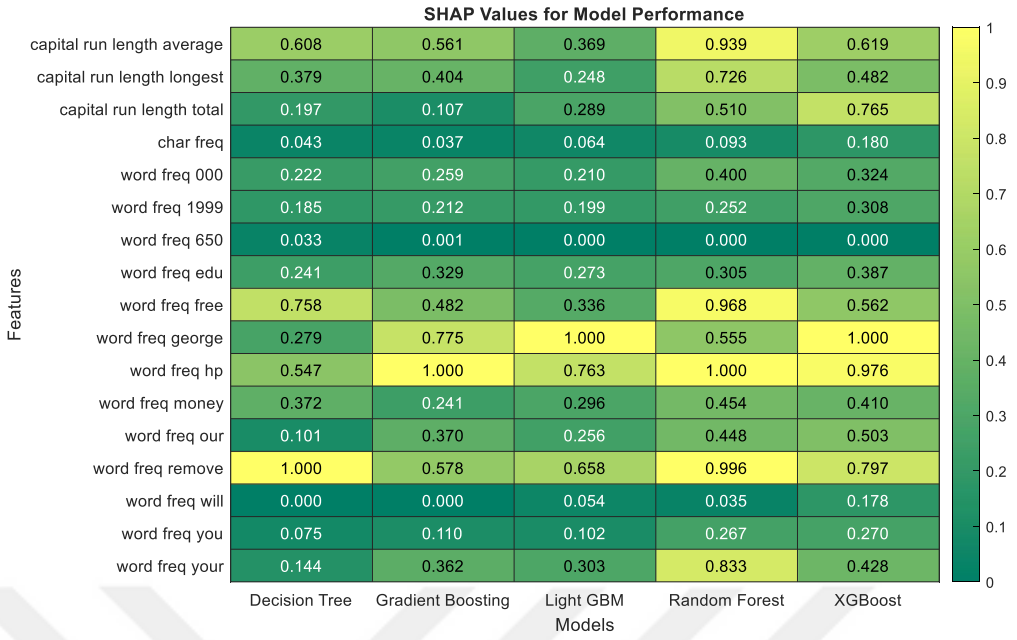

Figure 7. SHAP Values for Model Performance on the Spambase Dataset.

4.15 Performance Results of ML Models on Breast Cancer Dataset without Feature Selection (30 Features)

Tables 20 and 21 present the performance metrics of various ML models applied to the Breast Cancer training and test datasets without prior feature selection. The evaluated models include XGBoost, Random Forest, Gradient Boosting, LightGBM, and Decision Tree, based on the test results. Key performance indicators such as accuracy, precision, recall, F1 score, ROC AUC, and running time were assessed to comprehensively compare the models' capabilities.

XGBoost emerged as the top-performing model, achieving the highest accuracy of 0.971, with good precision (0.967) and recall (0.97). Its F1 score was 0.969, with an impressive ROC AUC of 0.994, indicating its excellent classification capability. Additionally, XGBoost demonstrated efficiency in training with a time of just 0.119 seconds, making it a robust choice for accurate and timely predictions.

Random Forest also performed well, with an accuracy of 0.965. Its precision and recall were 0.965, and the model maintained a solid F1 score of 0.962. However, it required a marginally longer running time of 0.219 seconds.

Table 20

Performance Results without Feature Selection for the Breast Cancer Training Data

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
XGBoost	0.991	0.990	0.991	0.991	0.997	0.119
Random Forest	0.989	0.990	0.988	0.989	0.997	0.219
Gradient Boosting	0.988	0.987	0.986	0.987	0.997	0.467
Light GBM	0.984	0.982	0.986	0.984	0.997	0.202
Decision Tree	0.984	0.983	0.984	0.983	0.996	0.009

Table 21

Performance Results without Feature Selection for the Breast Cancer Test Data

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
XGBoost	0.971	0.967	0.970	0.969	0.996	0.119
Random Forest	0.965	0.965	0.959	0.962	0.996	0.219
Gradient Boosting	0.953	0.953	0.946	0.949	0.995	0.467
Light GBM	0.947	0.942	0.945	0.944	0.994	0.202
Decision Tree	0.924	0.916	0.923	0.919	0.923	0.009

Gradient Boosting provided a balanced performance with an accuracy of 0.953, precision of 0.953, and recall of 0.946. While its ROC AUC was the highest at 0.995, indicating overall solid classification performance, the model required a longer running time of 0.467 seconds, which may limit its use in time-sensitive applications.

LightGBM showed decent performance, with an accuracy of 0.947, precision of 0.942, and recall of 0.945. It maintained a competitive F1 score of 0.944 and a ROC AUC of 0.994. It was also relatively efficient, with a running time of 0.202 seconds.

Decision Tree, while the fastest to train with a time of 0.009 seconds, had the lowest accuracy of 0.924. Although it showed reasonable precision (0.916) and recall (0.923), its F1 score (0.919) and ROC AUC (0.923) were lower compared to the other models, indicating that it may not be the best choice for complex classification tasks.

The analysis indicates that XGBoost is the most effective model for the Breast Cancer dataset when no feature selection is applied, balancing high accuracy, good classification metrics, and efficient running time. Random Forest is also a strong contender, particularly for applications. Gradient Boosting, while offering high classification power, requires more running time, making it less ideal for real-time applications. LightGBM and Decision Tree, while faster, may be more suited to scenarios where rapid deployment is prioritized over maximum accuracy.

4.16 Performance Results of ML Models in the Proposed Framework on Breast Cancer Dataset (21 Hybrid Features)

Tables 22 and 23 present the performance results of several machine learning models on the Breast Cancer training and test datasets, utilizing a feature selection method based on the proposed framework (Hybrid Features: 21). The evaluated models include Random Forest, Gradient Boosting, XGBoost, LightGBM, and Decision Tree. Performance is assessed using key metrics such as accuracy, precision, recall, F1 score, ROC AUC, and running time, as shown in Table 23.

XGBoost performed commendably, with an accuracy of 0.977 and a ROC AUC of 0.997. With precision of 0.975 and recall at 0.975, the model maintained a high F1 score of 0.975, and with the running time of 0.364 seconds, XGBoost offers a good balance of speed and performance. This makes XGBoost particularly suitable for scenarios requiring quick model deployment without significant sacrifices in performance.

Table 22

Performance Results of the Proposed Framework for the Training Data (Hybrid Features: 21)

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
XGBoost	0.993	0.992	0.992	0.992	0.998	0.364
Random Forest	0.991	0.991	0.990	0.991	0.998	0.434
Gradient Boosting	0.988	0.987	0.986	0.987	0.998	0.247
Light GBM	0.984	0.983	0.983	0.983	0.997	0.765
Decision Tree	0.975	0.972	0.978	0.974	0.997	0.008

Table 23

Performance Results of the Proposed Framework for the Test Data (Hybrid Features: 21)

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
XGBoost	0.977	0.975	0.975	0.975	0.997	0.364
Random Forest	0.971	0.970	0.967	0.968	0.997	0.434
Gradient Boosting	0.959	0.957	0.954	0.956	0.996	0.247
Light GBM	0.947	0.945	0.942	0.943	0.994	0.765
Decision Tree	0.918	0.908	0.925	0.914	0.925	0.008

Random Forest maintained its strong performance with an accuracy of 0.971 and a ROC AUC of 0.997. It also demonstrated robust precision and recall (at 0.970 and 0.967, respectively), with an F1 score of 0.968. The running time was relatively

efficient at 0.434 seconds, making Random Forest a good choice for accuracy and computational efficiency in this context.

Gradient Boosting achieved a slightly lower accuracy of 0.959 but performed well across other metrics, particularly in precision (0.957) and recall (0.954), resulting in a solid F1 score of 0.956. Its ROC AUC was high at 0.996, indicating reliable classification capability. The running time was faster than Random Forest, at 0.247 seconds, making it an efficient choice for applications requiring good performance with quicker training times.

LightGBM had an accuracy of 0.947, with precision and recall at 0.945 and 0.942, respectively, leading to an F1 score of 0.943. The model's ROC AUC was 0.994, comparable to the other top-performing models. LightGBM was relatively efficient in training, with a time of 0.765 seconds, making it a reliable option for applications where both accuracy and efficiency are important.

While the Decision Tree was the fastest to train at just 0.008 seconds, it had the lowest accuracy of 0.918. Its precision (0.908) and recall (0.925) were reasonable, resulting in an F1 score of 0.914. However, its ROC AUC of 0.925 was lower than the other models, indicating that while Decision Tree is highly efficient, it may not be the best choice for tasks requiring high accuracy and strong classification capabilities.

The analysis shows that XGBoost and Random Forest are the most effective models when using the Proposed Framework, offering a balance of high accuracy, highly accurate classification metrics, and reasonable running times. Gradient Boosting is also a strong contender, especially when quick training is a priority. LightGBM provides consistent performance across all metrics, making it a versatile option. However, the Decision Tree is best suited for tasks where training speed is critical, but accuracy and classification power are less of a concern. Overall, the Proposed Framework effectively enhances model performance while optimizing training efficiency, making it a valuable approach for machine learning tasks on the Breast Cancer dataset.

4.17 Performance Results of ML Models on Breast Cancer Dataset Using Selected Important Features (10 Model Features)

Tables 24 and 25 showcase the performance of various machine learning models on the Breast Cancer training and test datasets, applying feature selection using the ML Models Feature Importance method, focusing on 10 key features. The models analyzed include Random Forest, LightGBM, XGBoost, Gradient Boosting, and Decision Tree. Their performance is evaluated based on accuracy, precision, recall, F1 score, ROC AUC, and running time, as detailed in Table 25.

Random Forest achieved the highest accuracy of 0.965, demonstrating good precision (0.965) and recall (0.959). The model's F1 score was also high at 0.962 and maintained a robust ROC AUC of 0.995. However, the running time was 0.214 seconds, which, while not the fastest, is acceptable considering the model's overall performance.

LightGBM followed closely with an accuracy of 0.953, precision and recall at 0.950, and an F1 score of 0.950. The model's ROC AUC was 0.995, indicating its reliability in classification tasks. LightGBM had a very efficient running time of 0.09 seconds, making it a strong candidate for applications where quick training is essential without compromising much on accuracy.

XGBoost performed well with an accuracy of 0.947, slightly lower than Random Forest and LightGBM. Its precision was 0.945, and recall was 0.942, resulting in an F1 score of 0.943. XGBoost also achieved a high ROC AUC of 0.995 and was the fastest to train among the higher-performing models, with a time of 0.054 seconds. This efficiency makes XGBoost particularly suitable for time-sensitive tasks requiring fast model deployment.

Gradient Boosting showed lower performance with an accuracy of 0.936, precision of 0.928, and recall of 0.936. The model's F1 score was 0.932, and it maintained a ROC AUC of 0.994. The running time was longer at 0.322 seconds, which might limit its applicability in scenarios requiring rapid model updates.

While Decision Tree had the shortest running time, at just 0.006 seconds, it delivered the lowest accuracy of 0.930. Its precision and recall were 0.923 and 0.928, respectively, with an F1 score of 0.925 and a ROC AUC of 0.928. While highly

efficient in training, Decision Tree's lower accuracy and classification power make it less suitable for tasks requiring high predictive performance.

Table 24

Performance Results of ML Models Feature Importance for the Training Data (Model Features: 10)

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Random Forest	0.988	0.987	0.986	0.987	0.998	0.214
Light GBM	0.986	0.984	0.986	0.985	0.998	0.090
XGBoost	0.986	0.985	0.985	0.985	0.998	0.054
Gradient Boosting	0.984	0.983	0.983	0.983	0.997	0.322
Decision Tree	0.984	0.983	0.984	0.983	0.997	0.006

Table 25

Performance Results of ML Models Feature Importance for the Test Data (Model Features: 10)

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Random Forest	0.965	0.965	0.959	0.962	0.995	0.214
Light GBM	0.953	0.950	0.950	0.950	0.995	0.090
XGBoost	0.947	0.945	0.942	0.943	0.995	0.054
Gradient Boosting	0.936	0.928	0.936	0.932	0.994	0.322
Decision Tree	0.930	0.923	0.928	0.925	0.928	0.006

The analysis indicates that Random Forest remains the top performer in accuracy and classification metrics, making it a reliable choice using the ML Models Feature Importance method with 10 features. LightGBM and XGBoost offer a strong balance between accuracy and training efficiency, with LightGBM being particularly efficient in running time. Gradient Boosting provides solid classification capability but with

higher training costs, while Decision Tree offers unparalleled training speed at the expense of predictive power. Overall, the ML Models Feature Importance method effectively enhances the performance of most models, particularly Random Forest and LightGBM, while optimizing training efficiency.

4.18 Performance Results of ML Models Using SHAP-Based Feature Selection on Breast Cancer Dataset (10 SHAP Features)

Tables 26 and 27 present the performance metrics of various ML models on the Breast Cancer training and test datasets, respectively. The models were evaluated using SHAP-based feature selection, focusing on the top 10 features identified by SHAP. The models analyzed include Random Forest, XGBoost, LightGBM, Gradient Boosting, and Decision Tree. Performance is assessed based on key parameters such as accuracy, precision, recall, F1 score, ROC AUC, and training duration, as detailed in the test results of Table 27.

Both Random Forest and XGBoost obtained a peak accuracy of 0.965. The Random Forest model achieved a precision and recall of 0.962, leading to an F1 score of 0.962 and an ROC AUC of 0.994. Nevertheless, the model's running time was 0.320 seconds, slightly longer than that of other models. Conversely, XGBoost exhibited consistent accuracy while achieving a slightly improved running time of 0.189 seconds, establishing itself as an efficient model that strikes a commendable equilibrium between accuracy and computing speed.

LightGBM's performance was excellent, achieving an accuracy of 0.959, precision of 0.955, recall of 0.958, and a high F1 score of 0.956. The model's ROC AUC was 0.994, which aligns with the performance of the best models. LightGBM demonstrated exceptional efficiency in running time, requiring only 0.341 seconds. This makes it a favorable choice for situations where training speed and optimal performance are paramount.

Gradient Boosting demonstrated a commendable level of accuracy, achieving a score of 0.947. Additionally, it exhibited a precision of 0.945 and a recall of 0.942. The model's F1 score was 0.943 and attained a ROC AUC of 0.994. Although Gradient Boosting exhibited strong performance, it had a rather lengthy running time of 0.234

seconds, which could be seen as a disadvantage in applications that require quick results.

The Decision Tree model, with a running time of only 0.007 seconds, exhibited the lowest accuracy among all the models, measuring at 0.936. The system's precision was 0.930, while the recall was 0.933, leading to an F1 score of 0.931. The ROC AUC was also the lowest at 0.933, suggesting that although the Decision Tree algorithm is highly efficient regarding running time, it may not be optimal for applications that demand high accuracy and strong classification capability.

Based on the investigation, Random Forest and XGBoost perform better when utilizing SHAP-based feature selection. These models exhibit the highest accuracy and effectively balance precision, recall, and training efficiency. LightGBM offers exceptional performance with slightly reduced accuracy, making it an efficient option. Gradient Boosting provides strong classification abilities while it requires longer training durations. Although Decision Trees are known for their efficiency, they may not be the most appropriate choice for applications that demand high accuracy and robust categorization capabilities. SHAP-based feature selection significantly improves the performance of several models, including Random Forest, XGBoost, and LightGBM, by striking a favorable balance between accuracy and computational efficiency.

Table 26

Performance Results of SHAP for Training Data (SHAP Features: 10)

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Random Forest	0.989	0.990	0.988	0.989	0.997	0.320
XGBoost	0.988	0.986	0.987	0.987	0.997	0.189
Light GBM	0.988	0.987	0.986	0.987	0.997	0.341
Gradient Boosting	0.984	0.983	0.983	0.983	0.996	0.234
Decision Tree	0.982	0.980	0.984	0.981	0.996	0.007

Table 27

Performance Results of SHAP for Test Data (SHAP Features: 10)

Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Random Forest	0.965	0.962	0.962	0.962	0.994	0.320
XGBoost	0.965	0.965	0.959	0.962	0.994	0.189
Light GBM	0.959	0.955	0.958	0.956	0.994	0.341
Gradient Boosting	0.947	0.945	0.942	0.943	0.994	0.234
Decision Tree	0.936	0.93	0.933	0.931	0.933	0.007

4.19 Comparative Analysis of the Studies for Test Data

Table 28 presents a comparative analysis of different feature selection methods applied to XGBoost and Random Forest models. The results highlight the trade-offs between accuracy, precision, recall, F1 score, ROC AUC, and running time for each method.

For XGBoost without feature selection (F: 30), the model achieves an accuracy of 0.971 and a precision of 0.967, alongside a ROC AUC of 0.994. Notably, the running time is the shortest at 0.119 seconds, making this approach highly efficient for applications where computational speed is critical. Although the accuracy is slightly lower than when using feature selection, the rapid execution makes this method ideal when feature reduction is not a priority, but time efficiency is.

On the other hand, Hybrid Features (F: 21) with XGBoost achieves the highest accuracy of 0.977, with precision, recall, and F1 score all at 0.975. This demonstrates that feature selection enhances the model's performance, albeit at the cost of a longer running time of 0.364 seconds. While this is still efficient, the increase in running time suggests that this approach is best suited for cases where maximizing accuracy and maintaining high ROC AUC (0.995) is more important than reducing computational costs.

When using Random Forest with Model Features (F: 10), the model delivers an accuracy of 0.965 and a precision of 0.965. While the running time of 0.214 seconds is longer than XGBoost without feature selection, it offers a solid balance between

accuracy and execution speed. This method may be preferable when a slightly lower accuracy is acceptable in exchange for faster training times compared to larger feature sets.

Finally, the SHAP-based feature selection (F: 10) with Random Forest results in similar performance metrics (accuracy: 0.965, precision: 0.962) as the model feature selection approach, but with a longer running time of 0.320 seconds. Although this method maintains good classification power (ROC AUC: 0.994), it is less efficient computationally, making it more suitable for applications where the interpretability benefits of SHAP outweigh the need for speed.

Based on Table 28, the Hybrid Features (F: 21) with XGBoost stands out as the top-performing method, achieving the highest accuracy of 0.977, along with strong precision, recall, and F1 scores, all at 0.975, and an excellent ROC AUC of 0.995. While the running time of 0.364 seconds is longer than XGBoost without feature selection, the increase in performance justifies the trade-off. This method demonstrates that a careful selection of 21 hybrid features can optimize both model accuracy and classification power, making it a robust choice for tasks where precision and slightly longer computation time are acceptable to ensure superior results.

Table 28

Comparative Analysis of Performance Results for Test Data

Feature Selection Type	Model	Accuracy	Precision	Recall	F1 Score	ROC AUC	Running Time (s)
Hybrid Features (F:21)	XGBoost	0.977	0.975	0.975	0.975	0.995	0.364
Without Feature Selection (F:30)	XGBoost	0.971	0.967	0.97	0.969	0.994	0.119
Model Features (F:10)	Random Forest	0.965	0.965	0.959	0.962	0.995	0.214
SHAP Features (F:10)	Random Forest	0.965	0.962	0.962	0.962	0.994	0.320

4.20 Feature Importance Scores for Model Performance

In the evaluation of the Breast Cancer dataset, the importance of features across five different models (Decision Tree, Gradient Boosting, LightGBM, Random Forest, and XGBoost) is shown in Figure 8. According to the overall distribution of the features, the "Concave points (mean)" feature stands out as the feature with the highest importance score for all models. It can be said that this feature plays a critical role in classification accuracy for all models. Gradient Boosting and XGBoost models have assigned higher importance scores to many features. Notably, features like "Concave points (mean)" and "Perimeter (worst)" also show high importance scores in LightGBM and Random Forest models. Features such as "Concave points (worst)" and "Radius (worst)" also have notable importance scores, particularly playing a more critical role in Random Forest. However, for XGBoost, the "Radius (worst)" feature shows a lower importance score than expected, suggesting that it plays a less critical role in XGBoost's decision-making process compared to other models.

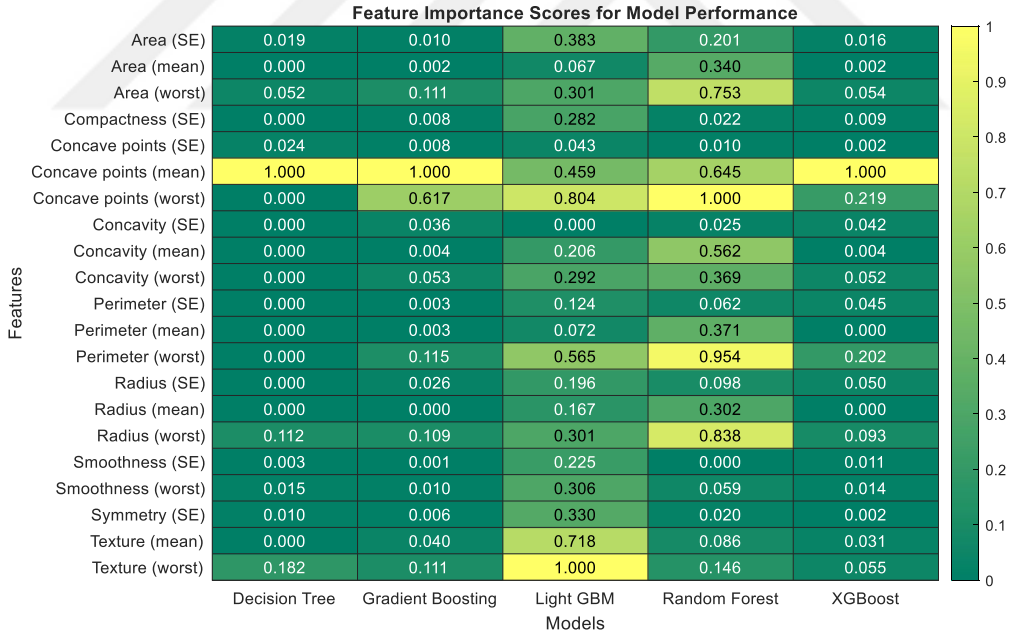

Figure 8. Feature Importance Scores for Model Performance on the Breast Cancer Dataset.

4.21 SHAP Values for Model Performance

The graph displaying SHAP values, which provide insights into the explainability and internal workings of the models, is presented in Figure 9. According to SHAP values, "Concave points (mean)" emerges as one of the most important features across all models, contributing significantly to model outcomes, especially in the XGBoost and Random Forest models with high SHAP values. SHAP values indicate that there is very little difference in explainability between the Gradient Boosting, LightGBM, and XGBoost models. However, the "Concave points (mean)" feature shows a very high SHAP value in the XGBoost model, indicating that the model provides more interpretable results. Features like "Perimeter (worst)", "Radius (worst)", and "Texture (worst)" are also shown to be important in SHAP values, contributing significantly to the models' decision-making processes in cancer detection.

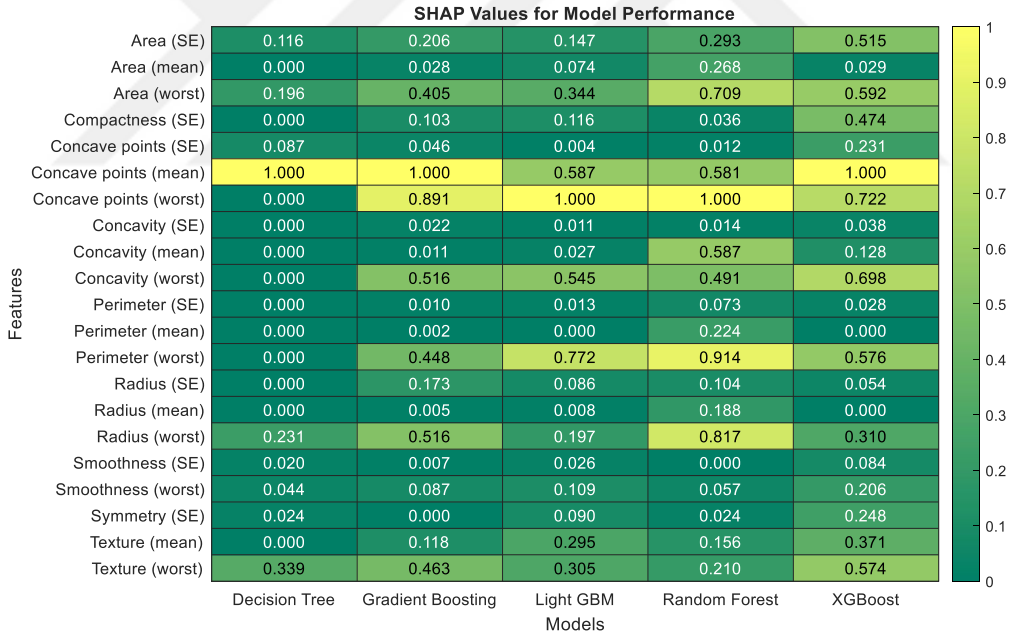

Figure 9. SHAP Values for Model Performance on the Breast Cancer Dataset.

In general, both feature importance and SHAP values indicate that features such as "Concave points (mean)", "Perimeter (worst)", and "Radius (worst)" play an important role in cancer detection. These features stand out with high importance and SHAP values across all models. Additionally, based on SHAP and feature importance

results, XGBoost and Random Forest models offer higher accuracy and better explainability on the Breast Cancer dataset. Particularly according to SHAP values, these models are observed to produce clearer and more robust results during the classification processes.

Chapter 5

Discussions and Conclusions

This study explored the integration of SHAP-based Explainable Artificial Intelligence (XAI) techniques into machine learning models, including Gradient Boosting Machines, XGBoost, Random Forest, LightGBM, and Decision Tree. It examined how this integration influences the models' performance and interpretability when feature selection methods are applied to datasets such as EEG data, Breast Cancer Wisconsin, and SpamBase.

The results indicate that incorporating ten critical features identified by each model within a hybrid feature selection framework improves the overall effectiveness of machine learning models without sacrificing predictive accuracy. By combining features from different models and incorporating diverse perspectives, this method ensures that no valuable features are missed. This approach leads to a more robust feature set, enhancing the model's ability to generalize more effectively.

The study demonstrated that applying the feature selection framework across all models maintains high accuracy while simplifying the complexity of the models. XGBoost and LightGBM, in particular, benefited from this feature selection strategy, maintaining strong predictive performance. SHAP-based feature selection improves model interpretability by providing insights into the importance of each feature, thus enhancing the transparency of the decision-making process. This effect was especially noticeable in the EEG dataset, which contains a large number of features. The approach proves to be more effective when applied to complex datasets with numerous features.

Additionally, applying this method to both EEG data and traditional datasets like Breast Cancer Wisconsin and SpamBase highlights its versatility and effectiveness across various research fields. In biological data analysis, maintaining high accuracy while simplifying models is crucial. Both predictive accuracy and interpretability are key factors in this sector.

The findings emphasize the importance of feature selection when developing models using ensemble methods. The integrated feature selection technique successfully balances accuracy, efficiency, and interpretability. This approach is a

valuable tool for enhancing the performance of machine learning models across different datasets.

This study contributes to machine learning research by presenting a reliable feature selection method with significant advantages. The proposed approach is particularly useful in disciplines that require clear explanations, such as biomedical and decision-making systems. Future research could explore expanding this framework to cover different data types and models, as well as examining its effectiveness in real-time machine learning systems. This study confirms that using SHAP-based feature selection with various feature combinations results in models that maintain high accuracy and interpretability, thus boosting confidence in their reliability for critical decision-making processes.

REFERENCES

Alam, M. Z., Rahman, M. S., & Rahman, M. S. (2019). A Random Forest based predictor for medical data classification using feature ranking. *Informatics in Medicine Unlocked, 15*, Article 100180.

Ali, 1. M. A., Özöğür-Akyüz, S. Duru, A. D., Almelek A. and Çalişkan, M. (2019). *Exploring Effects of Creativity Training on Default Mode Network and Attention*, 2019 11th International Conference on Electrical and Electronics Engineering (ELECO), Bursa, Turkey, 2019, pp. 954-958.

Badirli, S., Liu, X., Xing, Z., Bhowmik, A., Doan, K., & Keerthi, S. S. (2020). Gradient boosting neural networks: Grownet. *arXiv preprint* arXiv:2002.07971.

Banga, A., Ahuja, R., & Sharma, S. C. (2021). Performance analysis of regression algorithms and feature selection techniques to predict PM 2.5 in smart cities. *International Journal of Systems Assurance Engineering and Management,* 1–14.

Belgiu, M., & Drăguţ, L. (2016). Random forest in remote sensing: A review of applications and future directions. *ISPRS Journal of Photogrammetry and Remote Sensing, 114*, 24-31.

Bénard, C., Biau, G., Da Veiga, S., & Scornet, E. (2021). *Interpretable random forests via rule extraction*. In International conference on artificial intelligence and statistics (pp. 937–945). PMLR.

Biau, G., & Scornet, E. (2016). A random forest guided tour. *Test, 25(2)*, 197-227.

Bojer, C. S., & Meldgaard, J. P. (2021). Kaggle forecasting competitions: An overlooked learning opportunity. *International Journal of Forecasting 37(2)*, 5087-5093.

Breast Cancer Wisconsin (Diagnostic) - UCI Machine Learning Repository

Breiman, L. (2001). Random forests. Machine Learning, 45(1), 5-32.

Chang, Y.-C., Chang, K.-H., & Wu, G.-J. (2018). Application of eXtreme gradient boosting trees in the construction of credit risk assessment models for financial institutions. *Applied Soft Computing, 73*, 914–920.

Chen, T., & Guestrin, C. (2016). XGBoost: *A scalable tree boosting system*. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785-794.

Chen, T., He, T., Benesty, M., Khotilovich, V., & Tang, Y. (2015). XGBoost: Extreme Gradient Boosting. R package version 0.4-2, 1-4. https://github.com/dmlc/xgboost

Demir, C., Özöğür-Akyüz, S., & Göksel, İ. (2021). *Ensemble based feature selection with hybrid model*. In 2021 29th Signal Processing and Communications Applications Conference (SIU) (pp. 1-4). Istanbul, Turkey.

Dietterich, T. G. (2000). *Ensemble methods in machine learning*. In International workshop on multiple classifier systems (pp. 1-15). Springer, Berlin, Heidelberg.

Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics, 29*(5), 1189-1232.

Georganos, S., Grippa, T., Vanhuysse, S., Lennert, M., Shimoni, M., & Wolff, E. (2018). Very high-resolution object-based land use-land cover urban mapping using extreme gradient boosting. *IEEE Geoscience and Remote Sensing Letters, 15(4)*, 607-611.

Ghourabi, A. (2022). A security model based on LightGBM and transformer to protect healthcare systems from cyberattacks. *IEEE Access, 10*, 48890–48903.

Güldoğuş, B.Ç., & Özöğür-Akyüz, S. (2024). FSOCP: Feature selection via second-order cone programming. *Central European Journal of Operations Research*. https://doi.org/10.1007/s10100-023-00903-y

Güldoğuş, B.Ç., Abdullah, A.N., Ali, M.A., & Özöğür-Akyüz, S. (2023). Autoselection of the ensemble of convolutional neural networks with second-order cone programming. *arXiv Preprint*, arXiv:2302.05950.

Guyon, I., & Elisseeff, A. (2003). An introduction to variable and feature selection. *Journal of Machine Learning Research, 3*, 1157-1182.

Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2002). Gene selection for cancer classification using support vector machines. *Machine Learning, 46*(1), 389-422.

Ho, T. K. (1995). *Random decision forests*. In Proceedings of the 3rd International Conference on Document Analysis and Recognition (Vol. 1, pp. 278-282). IEEE.

Hu, X., Chen, H., & Zhang, R. (2019). Short paper: *Credit card fraud detection using LightGBM with asymmetric error control.* In 2019 second international conference on artificial intelligence for industries (AI4I) (pp. 91–94). IEEE.

Huang, K. (2020). An optimized lightgbm model for fraud detection. In *Journal of physics: Conference series*, Vol. 1651. IOP Publishing, Article 012111.

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., & Liu, T.-Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. *In Advances in Neural Information Processing Systems*, 3149-3157.

Khalilia, M., Chakraborty, S., & Popescu, M. (2011). Predicting disease risks from highly imbalanced data using random forest. *BMC Medical Informatics and Decision Making, 11*, 1–13.

Kohavi, R., & John, G. H. (1997). Wrappers for feature subset selection. *Artificial Intelligence, 97(1-2)*, 273-324.

Le, T.-T.-H., Oktian, Y. E., & Kim, H. (2022). XGBoost for imbalanced multiclass classification-based industrial internet of things intrusion detection systems. *Sustainability, 14(14)*, 8707.

Létinier, L., Jouganous, J., Benkebil, M., Bel-Létoile, A., Goehrs, C., Singier, A., Rouby, F., Lacroix, C., Miremont, G., Micallef, J., Salvo, F., Pariente, A. (2021). Artificial intelligence for unstructured healthcare data: application to coding of patient reporting of adverse drug reactions. *Clinical Pharmacology & Therapeutics, 110(2)*, 392–400.

Li, S., & Zhang, X. (2020). Research on orthopedic auxiliary classification and prediction model based on XGBoost algorithm. *Neural Computing and Applications, 32*, 1971–1979.

Li, Y., Stasinakis, C., & Yeo, W. M. (2022). A hybrid XGBoost-MLP model for credit risk assessment on digital supply chain finance. *Forecasting, 4(1)*, 184–207.

Liang, Y., Wu, J., Wang, W., Cao, Y., Zhong, B., Chen, Z., Li, Z. (2019). *Product marketing prediction based on XGboost and LightGBM algorithm*. In Proceedings of the 2nd international conference on artificial intelligence and pattern recognition (pp. 150–153).

Liu, J., Zhang, S., & Fan, H. (2022). A two-stage hybrid credit risk prediction model based on XGBoost and graph-based deep neural network. *Expert Systems with Applications, 195*, Article 116624.

Liu, Y., Meric, G., Havulinna, A. S., Teo, S. M., Ruuskanen, M., Sanders, J., et al. (2020). Early prediction of liver disease using conventional risk factors and gut microbiome-augmented gradient boosting. *MedRxiv*, 2020-06.

Lundberg, S. and Lee, S.-L. (2017) *A Unified Approach to Interpreting Model Predictions*. 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, 4-9 December 2017, 1-10.

Ma, H., Cao, J., Fang, Y., Zhang, W., Sheng, W., Zhang, S., et al. (2022). *Retrievalbased gradient boosting decision trees for disease risk assessment*. In Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining (pp. 3468–3476).

Mishra, A., Tiwari, A., & Kumar, A. (2021). XGBoost in Multiple Imbalanced Datasets: Techniques and Applications. *Expert Systems with Applications, 182*, 115221.

Mishra, M., Patnaik, B., Bansal, R. C., Naidoo, R., Naik, B., & Nayak, J. (2021). DTCDWT-SMOTE-XGBoost-based islanding detection for distributed generation systems: An approach of class-imbalanced issue. *IEEE Systems Journal, 16(2)*, 2008-2019.

Naik, K., & Mohan, B. (2021). Novel Stock Crisis Prediction Technique-A Study on Indian Stock Market. *IEEE Access, 9*, 86230-86242.

Natekin, A., & Knoll, A. (2013). Gradient boosting machines, a tutorial. *Frontiers in Neurorobotics, 7*, 21.

Nichols, J. A., Herbert Chan, H. W., & Baker, M. A. (2019). Machine learning: applications of artificial intelligence to imaging and diagnosis. *Biophysical Reviews, 11*, 111–118.

Ogunleye, A., & Wang, Q.-G. (2019). XGBoost model for chronic kidney disease diagnosis. IEEE/ACM *Transactions on Computational Biology and Bioinformatics, 17(6)*, 2131–2140.

Parsa, A. B., Movahedi, A., Taghipour, H., Derrible, S., & Mohammadian, A. K. (2020). Toward safer highways, application of XGBoost and SHAP for real-time accident detection and feature analysis. *Accident Analysis and Prevention, 136*, Article 105405.

Qin, X., Liu, J., Wang, Y., Liu, Y., Deng, K., Ma, Y., Zou, K., Li, L., Sun, X. (2021). Natural language processing was effective in assisting rapid title and abstract

screening when updating systematic reviews. *Journal of Clinical Epidemiology, 133*, 121-129.

Quinto, L. (2020). The Role of Gradient Boosting in Time Series Forecasting. *Journal of Time Series Analysis, 41(4)*, 493-512.

Quinto, B. (2020). Next-generation machine learning with spark: Covers XGBoost, LightGBM, Spark NLP, distributed deep learning with Keras, and more. Apress.

Rokach, L. (2010). Ensemble-based classifiers. *Artificial Intelligence Review, 33(1-2)*, 1-39.

Sarica, A., Cerasa, A., & Quattrone, A. (2017). Random forest algorithm for the classification of neuroimaging data in Alzheimer's disease: A systematic review. *Frontiers in Aging Neuroscience, 9*, 329.

Semanjski, I., & Gautama, S. (2015). Smart city mobility application—gradient boosting trees for mobility prediction and analysis based on crowdsourced data. *Sensors, 15(7)*, 15974–15987.

Shapley, L. S. (1953). A value for n-person games. In *Contributions to the Theory of Games (Vol. 2, pp. 307-317)*. Princeton University Press.

Dairu, X., Shilong, Z. (2021). *Machine learning model for sales forecasting by using XGBoost*. In 2021 IEEE international conference on consumer electronics and computer engineering (ICCECE) Guangzhou, China, pp. 480-483.

Siers, M. J., & Islam, M. Z. (2015). Software defect prediction using a cost sensitive decision forest and voting, and a potential solution to the class imbalance problem. *Information Systems, 51*, 62–71.

Spambase - UCI Machine Learning Repository

Strumbelj, E., & Kononenko, I. (2014). Explaining prediction models and individual predictions with feature contributions. *Knowledge and Information Systems, 41*(3), 647-665.

Sun, L., Liu, Y., & Sima, Y. (2020). A novel cryptocurrency price trend forecasting model based on LightGBM, *Finance Research Letters, 32*, 101084.

Sun, K., He, M., Xu, Y., Wu, Q., He, Z., Li, W., Liu, H., Pi, X. (2022). Multi-label classification of fundus images with graph convolutional network and LightGBM. *Computers in Biology and Medicine, 149*, 105909.

Tang, P. (2020). *Telecom customer churn prediction model combining k-means and xgboost algorithm*. 2020 5th International Conference on Mechanical, Control and Computer Engineering (ICMCCE), IEEE. pp. 1128–1131.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society: Series B (Methodological)*, 58(1), 267-288.

Vassallo, D., Vella, V., & Ellul, J. (2021). Application of gradient boosting algorithms for anti-money laundering in cryptocurrencies. *SN Computer Science, 2(3),* 1–15.

Walach, E., & Wolf, L. (2016). *Learning to count with CNN boosting*. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 5107-5115). IEEE.

Wang, D.-n., Li, L., & Zhao, D. (2022). Corporate finance risk prediction based on LightGBM. *Information Sciences, 602*, 259–268.

Wang, C., Deng, C., & Wang, S. (2020). Imbalance-XGBoost: leveraging weighted and focal losses for binary label-imbalanced classification with XGBoost. *Pattern Recognition Letters, 136,* 190–197.

Wang, Z., Shi, Y., Lyu, W., & Deng, H. (2017). Electricity Consumption Prediction Using XGBoost Based on Discrete Wavelet Transform. *DEStech Transactions on Computer Science and Engineering*.

Wu, Q., Ye, Y., Zhang, H., Ng, M. K., & Ho, S.-S. (2014). ForesTexter: An efficient random forest algorithm for imbalanced text categorization. *Knowledge-Based Systems, 67*, 105–116.

Zeiler, M. D. (2012). ADADELTA: An adaptive learning rate method. arXiv preprint arXiv:1212.5701.

Zeng, H., Yang, C., Zhang, H., Wu, Z., Zhang, J., Dai, G., Babiloni, F., Kong, W. (2019). A lightGBMbased EEG analysis method for driver mental states classification. *Computational Intelligence and Neuroscience, 2019*.

Zhang, L., & Zhan, C. (2017). *Machine learning in rock facies classification: An application of XGBoost*. In International geophysical conference, Qingdao, China, 17-20 April 2017 (pp. 1371–1374). Society of Exploration Geophysicists and Chinese Petroleum Society.

Zhang, K., Zhang, L., & Yang, M.-H. (2013). Real-time object tracking via online discriminative feature selection. *IEEE Transactions on Image Processing, 22(12)*, 4664–4677.

Zhu, M., Xia, J., Jin, X., Yan, M., Cai, G., Yan, J., et al. (2018). Class weights random forest algorithm for processing class imbalanced medical data. *IEEE Access,* 6, 4641–4652.

Zou, H., & Hastie, T. (2005). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology), 67(2),* 301-320.