COMPARATIVE STUDY OF MACHINE LEARNING AND ENSEMBLE LEARNING APPROACH ON TOOL WEAR CLASSIFICATION

Muhammet Ali Aykanat[a,b] , Rifat Kurban[c]

a, Department of Electrical and Computer Engineering, Abdullah Gul University, Kayseri, 38080, Turkey, muhammetali.aykanat@agu.edu.tr
b, Dener Group, Kayseri/Turkey, muhammet.aykanat@dener.com
c, Department of Computer Engineering, Abdullah Gul University, Kayseri, 38080, Turkey, rifatkurban@agu.edu.tr

Abstract

This study investigates the application of machine learning algorithms for predicting tool wear in machining operations, aiming to enhance production efficiency and reduce costs associated with tool maintenance. We implemented five distinct algorithms: K-Nearest Neighbors (KNN), Decision Trees, Random Forests, LightGBM, and XGBoost. The results reveal that these models can accurately classify tool conditions as "worn" or "unworn," with LightGBM and XGBoost showing solid performance. Notably, an ensemble approach using a soft voting classifier combining KNN, Random Forest, and LightGBM achieved an accuracy of 0.9968 and a ROC AUC of 0.9998. This research underscores the potential of machine learning to transform traditional tool management practices, enabling proactive maintenance strategies that can significantly improve machining efficiency and product quality. Future work may explore integrating real-time data for further enhancements in predictive accuracy.

Keywords: tool wear prediction, machine learning, ensemble learning.

1 Introduction

Tool wear is a critical issue in machining operations as it directly impacts the workpiece's quality and the machining process's efficiency [1]. The gradual loss of material from the cutting tool due to friction and other factors not only affects the tool itself but also leads to changes in the machined surface and the overall performance of the machine tool Understanding and effectively managing tool wear is essential to maintaining production quality, reducing production time, and minimizing economic losses associated with tool replacement and poor workpiece quality [2].

Researchers have explored various traditional methods and technologies to address tool wear problems without resorting to machine learning or artificial intelligence. One approach involves using sensor fusion strategies to monitor cutting tool wear [2]. By integrating data from different sensors that capture information on tool conditions during machining processes, operators can make informed decisions regarding tool replacement and maintenance to ensure consistent workpiece quality and production efficiency. Additionally, the application of Ti/AlTiN multilayer coatings on cutting tools has been investigated to mitigate the crater wear process and improve the tribological properties of the tools [3]. These coating technologies offer a preventive measure against wear, enhancing the durability and performance of cutting tools in machining operations. On the other hand, leveraging machine learning techniques for tool wear classification has shown promising results in enhancing the accuracy and efficiency of wear monitoring systems. Studies have demonstrated using support vector machine (SVM) algorithms coupled with time and frequency domain analysis to correlate sound signals generated during cutting processes with tool wear conditions [4]. Training machine learning models on these acoustic signatures makes it possible to classify tool wear states in real time, enabling proactive maintenance and replacement strategies to be implemented.

Furthermore, the integration of machine learning classification models, such as convolutional neural networks (CNNs), has been explored for online tool wear classification during machining processes [5]. By utilizing real-time cutting force measurements and CNN approaches, researchers have achieved significant accuracy rates in classifying tool wear states, enabling timely identification and mitigation strategies to be deployed [5]. Additionally, the use of pre-trained CNNs for vision-based tool wear classification has been investigated, highlighting the importance of timely identification and classification of wear conditions to guide tool replacement decisions and minimize wear-related issues [6].

In conclusion, the problem of tool wear in machine tools is a multifaceted issue that requires a comprehensive approach for effective management. While traditional methods like sensor fusion and coating technologies offer preventive measures against wear, the use of machine learning and artificial intelligence techniques provides advanced capabilities for real-time wear monitoring and classification. By combining these approaches, manufacturers can optimize tool usage, enhance production efficiency, and ensure consistent quality in machining operations.

In this study, various machine learning algorithms are implemented to address the tool wear problem. By leveraging the capabilities of machine learning, it becomes possible to predict tool wear with higher accuracy and reliability compared to traditional methods. The algorithms used in this study include K-Nearest Neighbors (KNN), Decision Tree, Random Forest, LightGBM, and XGBoost, each known for their unique strengths in handling different aspects of data. These models are compared in terms of their predictive accuracy to identify the most effective approach for tool wear prediction. Additionally, ensemble learning techniques are employed to combine the strengths of multiple models, aiming to achieve more robust and reliable results. Ensemble learning, through methods like voting classifiers, enhances the overall performance by mitigating the weaknesses of individual models, thus providing a more comprehensive solution to the tool wear problem.

2 Material and Method

2.1 Dataset

The dataset, originating from the University of Michigan's System-level Manufacturing and Automation Research Testbed (SMART), 18 different machining experiments performed on wax blocks (2" x 2" x 1.5") with S shape using a CNC milling machine [7]. The general data from each of the 18 distinct experiments encompass the experiment number, the material used (wax), the feed rate, and the clamping pressure. Each experiment's outputs include the condition of the tool (unworn or worn) and whether the tool passed a visual inspection. Time series data were collected from the 18 experiments at a sampling rate of 100 ms and are individually documented in files named experiment_01.csv to experiment_18.csv. Each file contains measurements from the CNC machine's four motors (X, Y, Z axes, and spindle). These experiments varied tool conditions, feed rates, and clamping pressures to investigate their effects on machining performance. The aggregated dataset comprised 25,286 observations and 52 features, of which 12 were categorical, and 40 were numerical.

2.2 Proposed Method

The proposed method, given in Figure 1, leverages a machine learning and ensemble learning approach to solve the given problem. This methodology comprises three main steps: data preprocessing, model implementation, and ensemble approach.

Data preprocessing is a crucial step that involves handling outliers and missing values, encoding categorical variables, standardizing the features, and performing stratified data splitting. Outlier handling ensures that extreme values do not skew the model's performance while addressing missing values, which prevents the introduction of bias. Encoding categorical variables transforms them into a numerical format suitable for machine learning algorithms. Standardization ensures that the features have a mean of zero and a standard deviation of one, essential for the proper convergence of many machine learning algorithms. Stratified splitting ensures that the train and test sets have similar distributions of the target variable, maintaining the representativeness of the data.

Five different machine learning models are implemented to identify the best solution: K-Nearest Neighbors (KNN) [8], Decision Tree [9], Random Forest [10], LightGBM [11], and XGBoost [12]. Each base model undergoes hyperparameter optimization and is evaluated using 5-fold cross-validation on the training set to ensure robust performance and prevent overfitting. KNN is known for its simplicity and effectiveness in classification tasks [13]. Decision Trees provide interpretability by creating a tree-like structure of decisions [14]. Random Forest, an ensemble of Decision Trees, improves performance through averaging, which reduces variance and prevents overfitting [15]. LightGBM and XGBoost are gradient-boosting frameworks that build models sequentially, with each new model correcting errors made by the previous ones [11], [12]. These methods are compelling for large datasets and have been shown to achieve high predictive accuracy [16], [17].

The ensemble approach employs a voting classifier, evaluated on the test set. The voting classifier combines KNN, Random Forest, and LightGBM as voters. Ensemble methods are known to improve predictive performance by combining the strengths of multiple models [18]. This approach reduces the likelihood of overfitting and increases robustness and generalizability [19]. By aggregating the predictions of diverse models, the ensemble method can achieve higher accuracy and better generalization compared to individual models [20], [21].

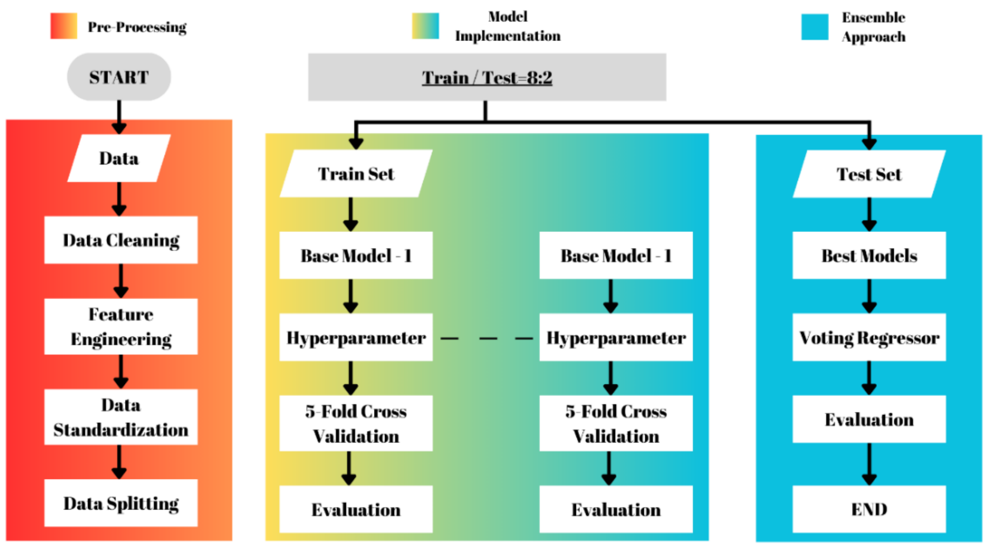

Figure 1: An architecture of proposed method.

3 Experiments and Results

3.1.1 Data Preprocessing

The dataset comprised 18 experimental CSV files and one training file containing tool status labels categorized as "worn" or "unworn". The initial step involved merging the 18 experimental files into a single dataset. This merged dataset included features from the experimental files along with additional columns for exp_no, feedrate, clamp_pressure, and tool_condition extracted from the training file.

The aggregated dataset consisted of 25,286 observations and 52 features. Among these features, 12 were categorical, and 40 were numerical.

Outliers were detected in 27 features and addressed using the Interquartile Range (IQR) method to ensure a more robust dataset for analysis.

To prepare the dataset for machine learning algorithms, we meticulously applied label encoding to the tool_condition feature. This process converted the categorical labels "worn" and "unworn" into numerical values, ensuring the accuracy of the data. One-hot encoding was then applied to the other categorical features to avoid any ordinal relationships being implied by the model.

After implementing the encoding, the shape of the dataset was transformed to (25,286, 61), reflecting the addition of new columns from the one-hot encoding process.

To standardize the dataset, Min-Max scaling (1) was applied to all features, bringing them into the range [0, 1]. The exp_no feature was subsequently dropped to prevent potential issues with high correlation.

$$X' = \frac{X - X_{min}}{X_{max} - X_{min}} \qquad (1)$$

These preprocessing steps resulted in a clean, normalized, and well-structured dataset ready for subsequent machine learning model development and analysis.

3.1.2 Base Model

Before the training phase, the dataset was stratified and split into training and testing sets with an 80-20 ratio, ensuring that both sets' class distribution of the tool_condition labels was preserved. A fixed random state was used to ensure the reproducibility of the results.

Five different machine learning models, KNN, DT, RF, LightGBM, and XGBoost, respectively, were implemented to predict the tool condition of the dataset.

To assess the models' performance and their ability to generalize to unseen data, a 5-fold cross-validation was conducted on the training set. This strategy ensured that each model was trained and validated on different portions of the data, providing a solid evaluation of the model's effectiveness. The results of this evaluation are presented in Table 1.

Model	Accuracy	F1_Score	ROC_AUC
KNN	0.9063	0.9111	0.9675
Decision Tree	0.988	0.9886	0.988
Random Forest	0.994	0.9943	0.9999
LightGBM	0.9952	0.9954	0.9998
XGBoost	0.9953	0.9955	0.9999

Table 1: Base model train phase results.

3.1.3 Hyperparameter Optimization

The same split data and model were used to implement hyperparameter optimization. A 5-fold cross-validation was performed during the training phase to evaluate the models. Hyperparameter optimization was then conducted using the following ranges in Table 2.

Model	**Hyperparameter**	**Range**
KNN	Number of neighbors	2 to 50
Decision Tree	Maximum depth	1 to 20
	Minimum sample split	2 to 30
Random Forest	Maximum depth	8 to 15
	Minimum sample split	15 to 20
	Number of estimators	200, 300
LightGBM	Learning Rate	0.01 to 0.1
	Number of estimators	300, 500
XGBoost	Learning rate	0.01 to 0.1
	Maximum depth	5 to 8
	Number of estimators	100, 200

Table 2: Models and their hyperparameter ranges.

The performance of each model was evaluated based on accuracy, F1-score, and ROC_AUC on the test set. The results of the best models after hyperparameter optimization with the train set are summarized in Table 3.

Model	***Accuracy***	***F1_Score***	***ROC_AUC***	***Best Parameters***
KNN	*0.9203*	*0.9242*	*0.9666*	*{'n_neighbors': 3}*
Decision Tree	*0.9873*	*0.988*	*0.9901*	*{'max_depth': 18, 'min_samples_split': 4}*
Random Forest	*0.9926*	*0.993*	*0.9998*	*{'max_depth': None, 'min_samples_split': 15, 'n_estimators': 200}*
LightGBM	*0.9968*	*0.9969*	*0.9999*	*{'learning_rate': 0.1, 'n_estimators': 500}*
XGBoost	*0.9958*	*0.996*	*0.9999*	*{'learning_rate': 0.1, 'max_depth': 8, 'n_estimators': 200}*

Table 3: Models and their hyperparameter results.

3.1.4 Model Evaluation

Accuracy measures how correct a model's predictions are overall. It is calculated as the ratio of correctly predicted instances to the total number of instances in the dataset. The formula for accuracy is:

$$Acc = \frac{TP+\mathrm{TN}}{All} \tag{2}$$

Accuracy is a valuable metric when the classes are balanced, as it provides a straightforward measure of how often the model is correct.

The F1-Score, which is the harmonic mean of precision and recall, serves as a metric that balances false positives and false negatives. It is particularly beneficial for imbalanced datasets because it takes into account both precision (the correctness of positive predictions) and recall (the capability to identify all positive cases). The formula for the F1-score is:

$$F1_{Score} = 2\,x\,\frac{\mathrm{Precision}\times\mathrm{Recall}}{\mathrm{Precision}+\mathrm{Recall}} \tag{3}$$

where

$$Precision = \frac{TP}{TP+\mathrm{FP}} \tag{4}$$

$$Recall = \frac{TP}{TP+\mathrm{FN}} \tag{5}$$

ROC AUC evaluates the area under the ROC curve, indicating the model's capacity to distinguish between positive and negative classes. The value ranges from 0 to 1, where higher values reflect superior performance.

$$ROC\_AUC = \int TPR \times d(FPR) \tag{6}$$

where TPR is true positive rate, FPR is false positive rate

Accuracy was selected as the primary evaluation metric for this study because the dataset is close to balanced, with 13,308 instances labelled as "worn" (52.63%) and 11,978 instances labelled as "unworn" (47.37%). In a balanced dataset, accuracy provides a clear and straightforward measure of model performance, as it equally considers the correct predictions of both classes. Additionally, since there is no significant class imbalance, the potential issues of overemphasizing either precision or recall (which the F1-score addresses) are minimized.

3.1.5 Accuracy Comparison of Models

Prediction results are obtained from the test set and classification report results are given in Table 4.

Model	Tool Condition	Accuracy	F1_Score	Support
KNN	Unworn	0.9306	0.9269	2396
	Worn		0.9339	2662
Decision Tree	Unworn	0.9871	0.9864	2396
	Worn		0.9878	2662
Random Forest	Unworn	0.9917	0.9912	2396
	Worn		0.9921	2662
LightGBM	Unworn	**0.9972**	**0.9971**	2396
	Worn		**0.9974**	2662
XGBoost	Unworn	0.9962	0.9960	2396
	Worn		0.9964	2662

Table 4: Models and their prediction results on test set.

Experiment results are given in Table 4 and Figure 1 and show that most of the models have good enough accuracy to handle tool wear classification. LightGBM and XGBoost are significantly accurate classifications compared to others.

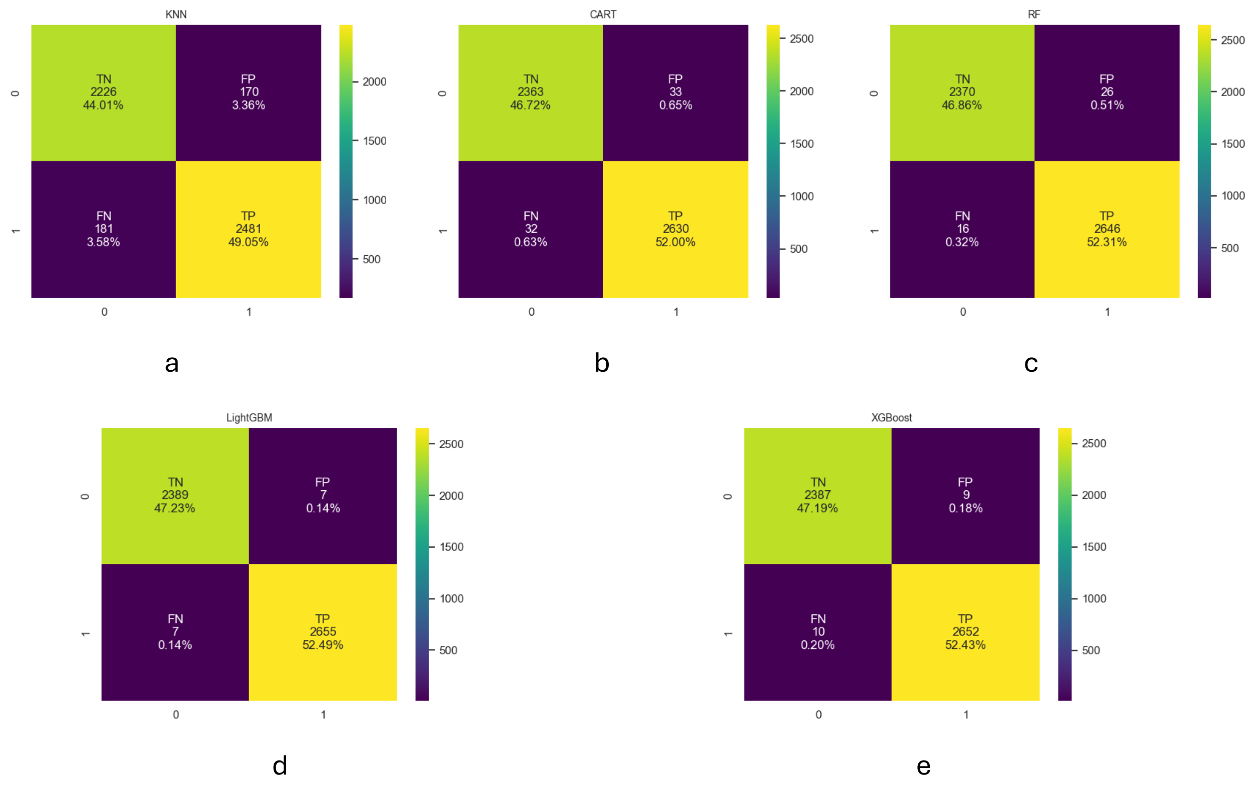

Figure 2: Confusion matrix of 5 models on test sets. KNN(a), DT(b), RF(c), LightGBM(d), XGBoost(e).

3.1.6 Ensemble Learning

In ensemble learning, a soft voting classifier is an advanced technique that merges the probabilistic outputs of several machine learning models to determine the final prediction. This classifier makes decisions based on the combined probabilities provided by all the contributing models. The soft voting classifier operates through the following steps:

Base model training: Multiple base classifiers, denoted as $C_1, C_2, \ldots, C_n$ are independently trained on the same dataset. These classifiers can be homogeneous (same algorithm) or heterogeneous (different algorithms)

Probability Prediction: For given input data x, each classifier C_i produces a predicted probability vector:

$$P_i = [p_{i1}, p_{i2}, \ldots, p_{ij}] \tag{7}$$

where p_{ij} is the predicted probability that belongs to classifier C_i and j is the total predicting class number.

The formula for the soft-voting classifier final decision:

$$\hat{y} = \arg max \sum_{j=1}^{m} p_{ij} \tag{8}$$

where $p_{ij} = P_i(\mathrm{C} \mid \mathrm{x})$ is probability for each class C given an input x.

For a classifier task with m models and C classes, each model j outputs a probability distribution $P_i(\mathrm{c} \mid \mathrm{x})$ for each given class C. This approach effectively leverages the strengths and mitigates the weaknesses of individual models, leading to enhanced overall performance.

In this study, KNN, RF, and LightGBM models are utilized as constituent models for the soft voting classifier. KNN is a non-parametric method that classifies a sample by looking at the predominant class among its nearest neighbors. RF is an ensemble approach that utilizes a collection of decision trees to boost predictive accuracy and prevent overfitting by averaging the predictions from several trees. LightGBM is a gradient-boosting framework that utilizes tree-based algorithms, renowned for its efficiency and outstanding performance. The combined use of these diverse models in a soft voting classifier resulted in an exceptional performance, achieving an accuracy of 0.9968, an F1 score of 0.9970, and an ROC AUC of 0.9998, demonstrating the effectiveness of this ensemble approach.

4 Conclusion

This research highlights the ability of machine learning algorithms to accurately predict tool wear in machining operations. By utilizing aggregated dataset and implementing K-Nearest Neighbors, Decision Trees, Random Forests, LightGBM, and XGBoost, we achieved notable classification accuracy for tool conditions as either “unworn” or “worn”. While all models show over 90% accuracy, LightGBM outperforms all. With the proposed method, the ensemble method, particularly the soft voting classifier combining KNN, Random Forest, and LightGBM, yielded exceptional results with an accuracy of 0.9968 and ROC AUC of 0.9998.

These findings highlight the potential of machine learning to enhance tool monitoring, allowing manufacturers to implement proactive maintenance strategies. By improving prediction accuracy, companies can reduce costs associated with tool replacement and improve production efficiency.

Future research may focus on integrating real-time data with different types of materials and exploring additional algorithms to further enhance predictive capabilities with less features. Overall, this study provides a promising framework for leveraging advanced analytics in manufacturing to optimize operational performance.

Author Contributions: Authors contributed equally.

Data Availability: The underlying data repository is publicly available on kaggle [7].

Code Availability: Code is publicly available and link is provided in reference [22].

5 References

[1] J. Ni, X. Liu, Z. Meng, and Y. Cui, “Identification of Tool Wear Based on Infographics and a Double-Attention Network,” *Machines*, vol. 11, no. 10, 2023, doi: 10.3390/machines11100927.

[2] M. Mahardika, Z. Taha, D. Suharto, K. Mitsui, and H. Aoyama, “Sensor Fusion Strategy in the Monitoring of Cutting Tool Wear,” in *Fracture and Strength of Solids VI*, in Key Engineering Materials, vol. 306. Trans Tech Publications Ltd, 2006, pp. 727–732. doi: 10.4028/www.scientific.net/KEM.306-308.727.

[3] K. J. Kadhim, A. A. Abbas, and M. A. H. Hussein, “Effect Ti/AlTiN Multilayer Coating on the Crater Wear Process of Cutting Tool and Tribological Properties,” *Al-Khwarizmi Eng. J.*, vol. 13, no. 4, pp. 58–68, Dec. 2018, doi: 10.22153/kej.2017.07.005.

[4] A. Kothuru, S. P. Nooka, and R. Liu, “Cutting Process Monitoring System Using Audible Sound Signals and Machine Learning Techniques: An Application to End Milling,” in International Manufacturing Science and Engineering Conference, vol. Volume 3: Manufacturing Equipment and Systems. 2017, p. V003T04A050. doi: 10.1115/MSEC2017-3069.

[5] G. Terrazas, G. Martínez-Arellano, P. Benardos, and S. Ratchev, “Online Tool Wear Classification during Dry Machining Using Real Time Cutting Force Measurements and a CNN Approach,” *J. Manuf. Mater. Process.*, vol. 2, no. 4, 2018, doi: 10.3390/jmmp2040072.

[6] A. S. Kumar, A. Agarwal, V. G. Jansari, K. A. Desai, C. Chattopadhyay, and L. Mears, “Vision-Based Tool Wear Classification During End-Milling of Inconel 718 Using a Pre-Trained Convolutional Neural Network,” in ASME International Mechanical Engineering Congress and Exposition, vol. Volume 3: Advanced Manufacturing. 2023, p. V003T03A016. doi: 10.1115/IMECE2023-113344.

[7] S. M. and A. R. T. (SMART), “CNC Mill Tool Wear.” [Online]. Available: https://www.kaggle.com/datasets/shasun/tool-wear-detection-in-cnc-mill

[8] T. Cover and P. Hart, “Nearest neighbor pattern classification,” *IEEE Trans. Inf. Theory*, vol. 13, no. 1, pp. 21–27, 1967, doi: 10.1109/TIT.1967.1053964.

[9] J. R. Quinlan, “Induction of decision trees,” *Mach. Learn.*, vol. 1, no. 1, pp. 81–106, 1986, doi: 10.1007/bf00116251.

[10] L. BREIMAN, “Random Forests,” *Mach. Learn.*, vol. 12343 LNCS, no. 45, pp. 5–32, 2001, doi: https://doi.org/10.1023/A:1010933404324.

[11] T. Y. Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., ... & Liu, “Lightgbm: A highly efficient gradient boosting decision tree,” *Adv. Neural Inf. Process. Syst.*, vol. 20, 2017, [Online]. Available: https://proceedings.neurips.cc/paper/2017/hash/6449f44a102fde848669bdd9eb6b76fa-Abstract.html

[12] T. Chen and C. Guestrin, “XGBoost: A scalable tree boosting system,” *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Min.*, vol. 13-17-Augu, pp. 785–794, 2016, doi: 10.1145/2939672.2939785.

[13] L. E. Peterson, “{K}-nearest neighbor,” *Scholarpedia*, vol. 4, no. 2, p. 1883, 2009, doi: 10.4249/scholarpedia.1883.

[14] S. R. Safavian and D. Landgrebe, “A survey of decision tree classifier methodology,” *IEEE Trans. Syst. Man. Cybern.*, vol. 21, no. 3, pp. 660–674, 1991, doi: 10.1109/21.97458.

[15] A. Liaw and M. Wiener, “Classification and Regression by RandomForest,” *Forest*, vol. 23, 2001.

[16] T. G. Dietterich, “Ensemble Methods in Machine Learning,” in *Multiple Classifier Systems*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2000, pp. 1–15.

[17] Z.-H. Zhou, *Ensemble Methods: Foundations and Algorithms*, vol. 14. 2012. doi: 10.1201/b12207.

[18] L. Rokach, “Ensemble-based classifiers,” *Artif. Intell. Rev.*, vol. 33, no. 1, pp. 1–39, 2010, doi: 10.1007/s10462-009-9124-7.

[19] R. Polikar, “Ensemble based systems in decision making,” *IEEE Circuits Syst. Mag.*, vol. 6, no. 3, pp. 21–45, 2006, doi: 10.1109/MCAS.2006.1688199.

[20] L. Kuncheva, “Combining Pattern Classifiers: Methods and Algorithms: Second Edition,” in *Combining Pattern Classifiers: Methods and Algorithms: Second Edition*, vol. 47, 2014. doi: 10.1002/0471660264.

[21] D. Opitz and R. Maclin, “Popular Ensemble Methods: An Empirical Study,” vol. 11, 1999, doi: 10.1613/jair.614.

[22] M. A. Aykanat; and R. Kurban, “CNC-Tool-Wear-Detection.” [Online]. Available: https://github.com/MAAykanat/CNC-Tool-Wear-Detection