INTERNATIONAL STUDIES AND EVALUATIONS IN

COMPUTER ENGINEERING

December 2024

EDITOR

PROF. DR. SELAHATTİN BARDAK

SERÜVEN YAYINEVİ

Genel Yayın Yönetmeni / Editor in Chief • C. Cansın Selin Temana

Kapak & İç Tasarım / Cover & Interior Design • Serüven Yayınevi

Birinci Basım / First Edition • © Aralık 2024

ISBN • 978-625-5955-69-2

Serüven Yayınevi / Serüven Publishing
Türkiye Adres / Turkey Address: Kızılay Mah. Fevzi Çakmak 1. Sokak
Ümit Apt No: 22/A Çankaya/ANKARA
Telefon / Phone: 05437675765
web: www.seruvenyayinevi.com
e-mail: seruvenyayinevi@gmail.com

Baskı & Cilt / Printing & Volume
Sertifika / Certificate No: 47083

INTERNATIONAL STUDIES AND EVALUATIONS IN

COMPUTER ENGINEERING

EDITOR

PROF. DR. SELAHATTİN BARDAK

CONTENTS

CHAPTER 1

IMPROVING THE CLASSIFICATION SUCCESS IN DATASETS WITH CLASS IMBALANCE USING THE SMOTE TECHNIQUE: AN EXAMPLE OF KIDNEY STONE DETECTION

Ebru Efeoğlu[a, 1]

1 Ebru Efeoğlu[a, *]
[a] Department of Software Engineering, Faculty of Engineering, Kütahya Dumlupınar University, Kütahya, Turkey
*email: ebru.efeoglu@dpu.edu.tr
ORCİD: https://orcid.org/0000-0001-5444-6647

1. Introduction

If the number of data in the classes is not equal in the data set to be used in classification, the data set is considered to be an imbalanced data set. The imbalanced data set problem is frequently encountered in classification. For example, when people are classified as sick or healthy for the detection of a rare disease, the number of people in the sick class is greater than the number of people in the non-sick class. This is because the disease is rare. This situation creates the imbalanced data set problem. When data is classified in imbalanced data, a high accuracy does not indicate that the algorithm is successful. In a data set with laboratory results of 100 people, let's say 95 people are in the non-sick class and 5 people are in the patient class. In this case, even if the algorithm correctly predicts the non-sick people and does not make any correct predictions from the patient class, the accuracy of the algorithm will be 95%. However, the algorithm could not detect any patients. Therefore, it is necessary to solve this problem in imbalanced data sets. There are different techniques to solve this problem. In this section, the problem of imbalanced data was solved using SMOTE (Synthetic Minority Over Sampling Technique). In the study, a freely available data set from the the open source Kaggle data repository was used as the data set for kidney stone detection.

Chronic kidney disease is a worldwide health problem. The prevalence of kidney stone risk factors in the adult population was investigated (Khalili et al., 2021). Nutritional and lifestyle risk factors associated with kidney stone formation in men and women were examined (Ferraro et al., 2017; Dai et al., 2013). A large part of the world's population is affected by chronic kidney disease. It is possible to stop the progression of this disease with early diagnosis and treatment. Developments in the field of informatics are also frequently and effectively used in the health sector. The application of machine learning (ML) to the field of medicine has provided great convenience to healthcare professionals and patients. Automatic kidney stone detection using coronal computed tomography (CT) images (Manoj et al., 2022; Yildirim et al., 2021)

and deep learning model for volumetric segmentation (Elton et al., 2022) and Kronecker curves in deep learning technique were applied (Patro et al., 2023). Detection of stones from CT images with ML and deep learning (Kolli et al., 2022), segmentation and measurement were performed (Babajide et al., 2022; Caglayan et al., 2022; Joseph & Apena, 2021). X-ray images of kidneys were classified with ML and deep learning algorithms (Aksakallı et al., 2021). A methodology consisting of preprocessing, feature extraction and classification stages was proposed for the detection of kidney stones from ultrasonic images (Manjunatha et al., 2024). Apart from these, transfer learning (Vishmitha et al., 2022) and discrete wavelet transform were used in the detection of kidney stones (Tahir & Abdulrahman, 2023).

Although the use of CT in the detection of kidney stones is effective, this method is costly and patients are exposed to radiation. Moreover, the waiting times for the radiology result report are long. Therefore, the laboratory results of patients registered in hospitals can be analyzed with ML techniques and presented to the service of physicians and managers. ML methods and laboratory results were used in the detection of kidney failure (Ventrella et al., 2021). 5 different ML methods were compared for the detection of kidney stones and a framework was proposed (Ventrella et al., 2021; Yang et al., 2021). Kidney stones were detected based on laboratory test results using ML algorithms (Alghamdi & Amoudi, 2024) . Kidney stone detection was performed using the physical properties of crystalline and non-crystalline urine samples. ML and deep learning algorithms were used for detection. The highest accuracy (90%) was achieved with deep learning (Gulhane et al., 2024). A multi-sensor urine test and data collection platform was developed for the risk assessment of kidney stone formation using the logistic regression model (Chung et al., 2020). The success of XGBoost and logistic regression models to predict stone composition using stone composition and electronic health record data with 24-h urine test was evaluated and accuracy rates between 0.64 and 0.56 were obtained (Abraham et al., 2022).

This study used routine laboratory test results and popular ML techniques to detect kidney stones early. The dataset imbalance problem was solved with the SMOOTE method and the performance of the algorithms was increased.

The rest of this chapter is organized as follows. Section 2 describes the solution methods for the classification problem in imbalanced data. Section 3 provides information about the ML techniques used. Section 4 includes performance analysis methods and performance measurements, Section 5 includes implementation, and Section 6 includes discussion and conclusion.

2. Solution methods for the classification problem in imbalanced data.

2.1. Undersampling

In this method, class balance is achieved by reducing the data of the majority class. The Undersampling technique is shown schematically in Figure 1. If the reduction is done randomly, it is called Random Undersampling. If the reduction is done using statistical information, it is called Informed Undersampling. However, it has been suggested that this technique may also result in the omission of useful information regarding classification (She et al., 2009).

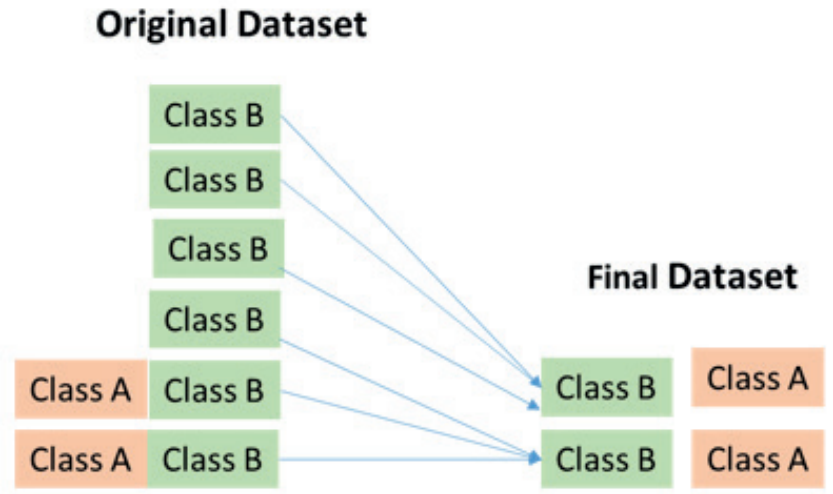


Figure 1. *Undersampling technique*

2.2. Oversampling

In this method, class balance is achieved by increasing the data of the minority class. The oversampling technique is shown schematically in Figure 2.

Random oversampling is the name given to randomly selecting and multiplying data from the minority class. It has been suggested that this method may lead to overfitting (Barista et al., 2004).

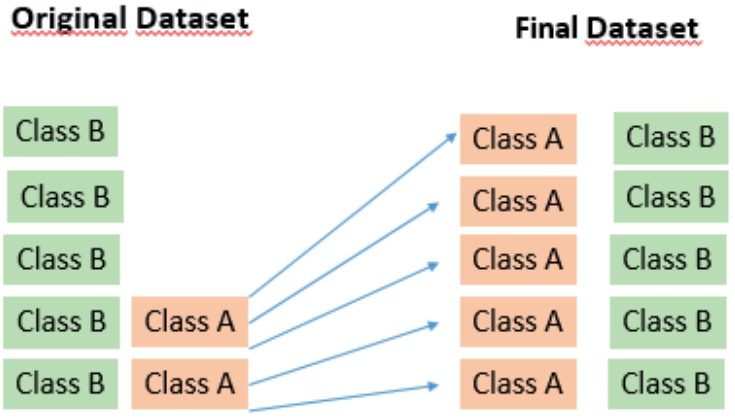


Figure 2. *Oversampling technique*

2.3. SMOTE technique

There are different techniques used to increase the quality of the data used for high model accuracy in classification models. Examples of these techniques include removing outliers, handling repeated values, and normalization. In addition, a balanced class distribution is required for high model accuracy. SMOTE, a statistical technique, is used to solve the problem of class imbalance in data sets with unbalanced class distribution. Using this technique, the number of data of classes with low data counts in the data set is increased and equalized to the number of data of other classes (Chawla, et al., 2002). The data augmentation process is done based on the k nearest neighbors of the data. In other words, artificial data is generated based on the k neighbors of the data. This technique can also be used to increase model accuracy in cases where the number of data is low.

3. Classification algorithms

ML method produces an output to make a prediction. If this output is categorical, the process is called classification, if it is numerical, it is called regression. The process of obtaining a model with a method from a training data and using this model in prediction is called classification. In classification, the experience obtained from the data is used and the class of the unknown data is predicted.

In this section, preliminary information will be given about the most popular and successful classification algorithms. In addition, the performances of these algorithms will be compared using the kidney stone data set.

3.1.Random Forest

It is one of the popular decision tree algorithms. This algorithm is often preferred in solving classification and regression problemsThe algorithm uses Random Feature Subset Selection to create a random Random Forest. Along with this, it includes Decision Tree Construction, Ensemble of Decision Trees and Out-of-Bag Evaluation steps to create a Random Forest. (Rajendiran, 2024).

3.2. K-Nearest Neighbor Algorithm (KNN)

Generally, in classification, there is a sample set with known classes. The class of a new sample is determined by using this sample set. In this method, the distances of the new sample to the observations in the sample set are taken into account. There are many metrics determined as distance criteria. Euclidean distance is commonly preferred. The K value represents the number of nearest neighbors to a given point.

3.3.Support vector machines (SVM)

It is based on Vapnik-Chervonenkis theory. It is a statistical learning algorithm. It has strong foundations. It separates two classes in the data set according to the boundary elements. The main purpose of the algorithm is to separate the classes with a line or plane.

3.4. Rep Tree

It uses a combination of incremental pruning and error reduction techniques to create a decision tree. The algorithm aims to reduce overfitting and improve the performance of the model by pruning. In the process steps of the algorithm, there is a Repetitive Binary Splitting operation in the Forest. In addition, a Pruning operation is applied to reduce the error. (Rajendiran, 2024).

4. Performance analysis methods and performance metrics

Performance analysis is performed to compare the success of classification algorithms. There are many analysis methods. The most frequently preferred cross-validation method was used in the chapter. This method allows researchers to test all the examples in the data set. The cross-validation method is known as k-fold cross-validation and the number k is usually taken as 10, as in this chapter. Taking the number k as 10 means dividing the data set into 10 equal parts. 1 of these parts is used for testing, while the other parts are used for training. This process is continued until all the examples in the data set are used as test data. In performance analysis, comparisons are made with various criteria. These criteria help understand which algorithm is more successful in classification. The values of the criteria vary between 0 and 1. It is accepted that the classification success increases as the criterion approaches 1. A criterion value of 1 indicates a perfect classification. The most basic performance criterion is the confusion matrix. A schematic representation of the confusion matrix is given in Figure 3. Other criteria can be calculated using this matrix. There are 4 different situations in the matrix. These situations occur as a result of the classification. It is possible to examine these situations in two parts. The first part is the cases where the targeted class of the sample and the class detected by the algorithm are the same. These cases are True Positive (TP) and False Positive (FP). The other part is the cases where the targeted class of the sample and the class detected by the algorithm are not the same. These are True Negative (TN) and False Negative (FN). The performance criteria used in the study can be calculated using the equations given below.

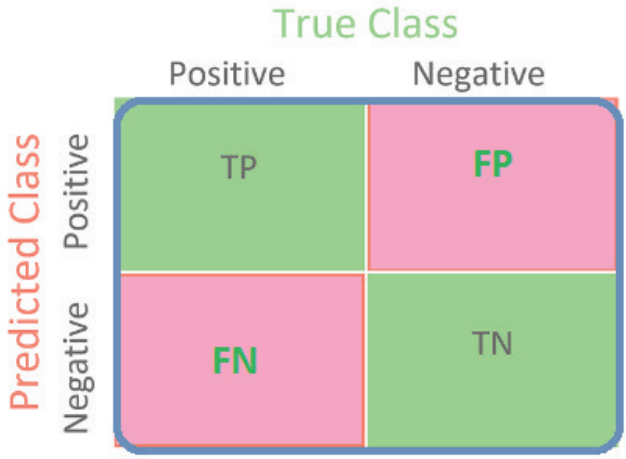


Figure 3. *Confusion matrix*

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FN+FP} \quad (1)$$

$$YPR = \frac{FP}{FP+TN} \quad (2)$$

$$\text{Precision} = \frac{TP}{TP+FP} \quad (3)$$

$$TPR = \frac{TP}{TP+FN} \quad (4)$$

$$\text{Recall} = \frac{TP}{TP+FN} \quad (5)$$

$$F-\text{score} = 2x\frac{\text{Precision x Recall}}{(\text{Precision} + \text{Recall})} \quad (6)$$

The Kappa value is a statistic that measures the agreement between two observers and takes values between -1 and +1. +1 indicates high agreement.

5. Uygulama

5.1. Dataset

The dataset used in the chapter was collected from the open source Kaggle data repository. The dataset contains laboratory test results of 89 patients (Kaggle stone dataset). This data includes 7 different parameters: gravity, ph, osmo, cond, urea, calc and Risk of Stone, which represents the risk of developing kidney stones. The risk of kidney stones was estimated based on these parameters in the chapter.

Figure 4 shows the class data numbers before and after the SMOTE process. With the SMOTE process, the data number of class A was increased from 55 to 96 and the data number of class B was increased from 34 to 97. Class A represents patients without risk of kidney stones, and class B represents patients with risk of kidney stones.

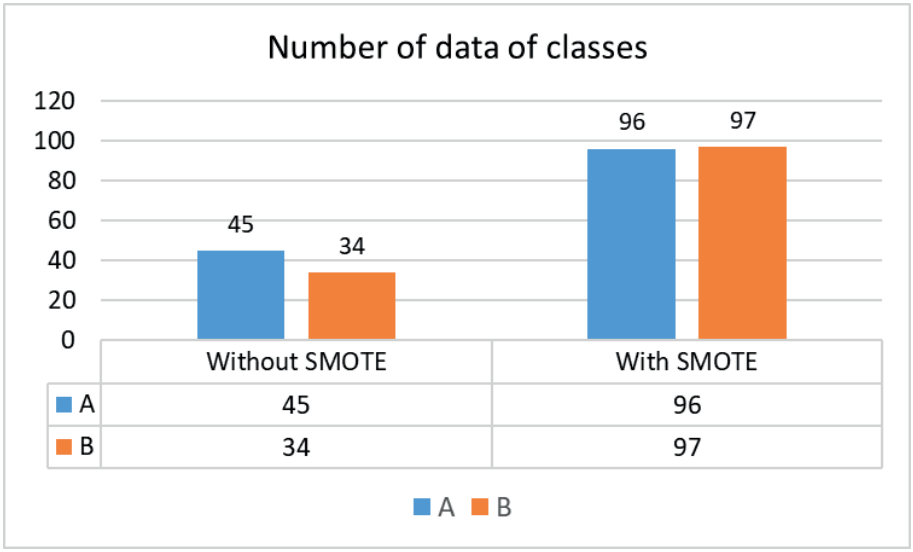


Figure 4. *Number of data classes*

6. Result

In the chapter, four different ML methods were used for kidney stone risk prediction. Performance analysis was performed. The performances of the algorithms were compared with 10-fold cross-validation. Confusion matrix, recall, precision, F-score, ROC and Kappa statistical value were used in the comparison. After using the SMOTE technique to increase the performances of the algorithms, the performances of the algorithms were compared using these metrics. The flow chart of the study is given in Figure 5.

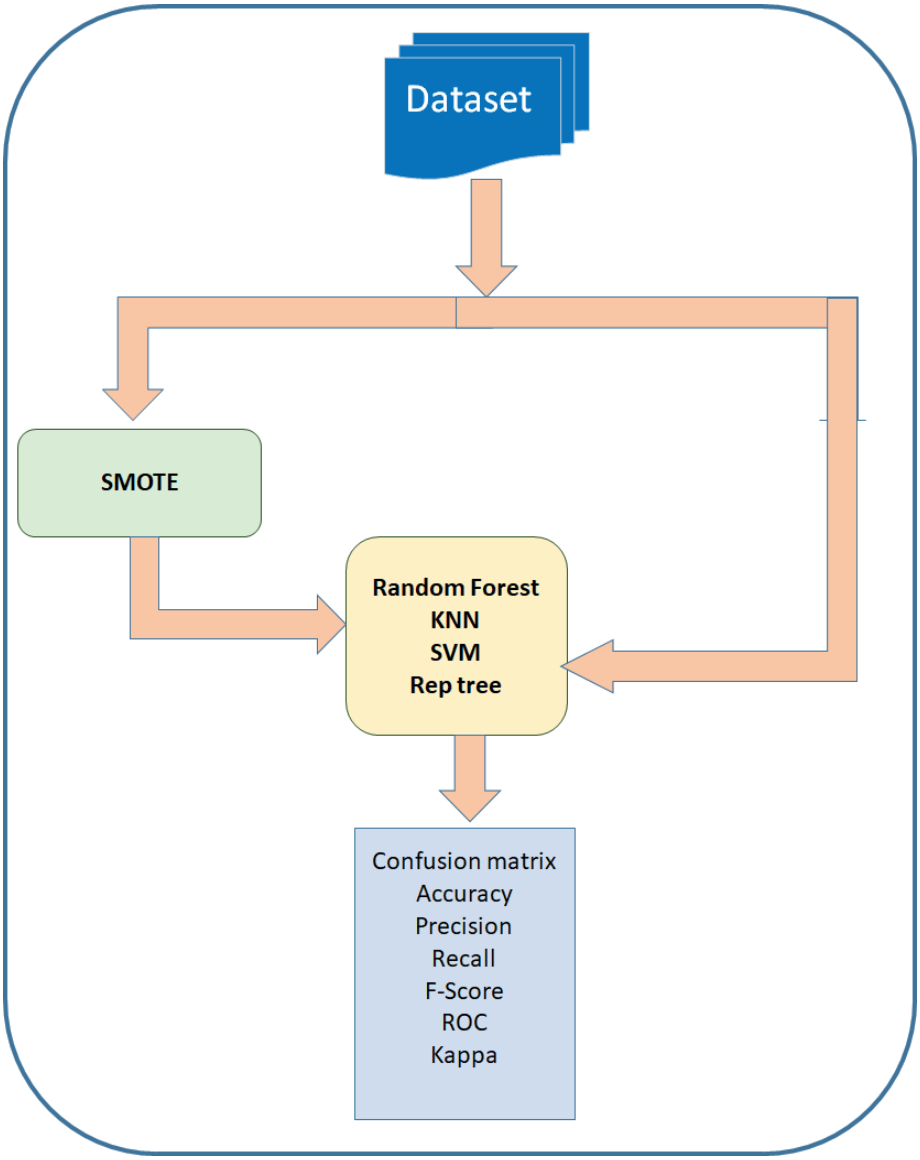


Figure 5. *Flowchart of the study*

Classification was done with and without SMOTE and these metrics were calculated for each classification algorithm. The calculated metrics are given in Table 1 and Table 2.

Table1. Performance metrics without SMOTE

Algorithm	Without SMOTE				
	Precision	Recall	F-Score	ROC	Kappa
Random Forest	0.75	0.75	0.75	0.80	0.50
KNN	0.47	0.74	0.74	0.74	0.48
SVM	0.75	0.73	0.72	0.70	0.43
Rep Tree	0.70	0.70	0.70	0.70	0.39

Table 2. Performance metrics with SMOTE

Algorithm	With SMOTE				
	Precision	Recall	F-Score	ROC	Kappa
Random Forest	0.89	0.89	0.89	0.95	0.78
KNN	0.88	0.88	0.88	0.89	0.77
SVM	0.78	0.77	0.77	0.77	0.54
Rep Tree	0.80	0.80	0.80	0.86	0.61

According to Table 1, the highest precision, recall and F-score were calculated as 0.75 in the classification performed without the SMOTE process. The highest ROC and Kappa values were obtained as 0.80 and 0.5, respectively. When the results were examined on an algorithmic basis, it was seen that the highest metric values were obtained when the Random Forest algorithm was used. When Table 2 was examined, Precision increased from 0.75 to 0.89 with the SMOTE process. Roc and kappa reached 0.95 and 0.78, respectively. With the SMOTE process, the highest metrics were achieved again with the Random Forest algorithm.

Another metric that measures the success of algorithms is confusion matrices. Confusion matrix is calculated separately for each algorithm. Random forest confusion matrix is given in Figure 6, KNN confusion matrix is given in Figure 7, SVM confusion matrix is given in Figure 8 and Rep tree confusion matrix is given in Figure 9. In the figure, pink color represents the number of patients that the algorithm correctly classified. Green color is the number of patients that the algorithm incorrectly predicted.

Figure 6. Randon forest confusion matrix a) without SMOTE b) with SMOTE

Actual Class		Predicted Class A	Predicted Class B
	A	34	11
	B	9	25

a)

Actual Class		Predicted Class A	Predicted Class B
	A	85	11
	B	11	86

b)

Figure 7. KNN confusion matrix a) without SMOTE b) with SMOTE

Actual Class		Predicted Class A	Predicted Class B
	A	41	4
	B	17	17

a)

Actual Class		Predicted Class A	Predicted Class B
	A	82	14
	B	30	67

b)

Figure 8. SVM confusion matrix a) without SMOTE b) with SMOTE

Actual Class		Predicted Class A	Predicted Class B
	A	35	10
	B	13	21

a)

Actual Class		Predicted Class A	Predicted Class B
	A	78	18
	B	19	78

b)

Figure 9. SVM confusion matrix a) without SMOTE b) with SMOTE

Accordingly, when the confusion matrix in Figure 6 is examined, it is seen that the Randon forest algorithm correctly predicted the kidney stone risk of 38 patients out of 45 patients without kidney stone risk (Class A) and incorrectly predicted the kidney stone risk of 7 patients. It correctly predicted 22 patients out of 34 patients with kidney stone risk (Class B). A total of 60 patients were correctly predicted. The number of patients correctly classified of the algorithm increased to 172 after the SMOTE process was applied.

Accordingly, when the confusion matrix in Figure 7 is examined, it is seen that the KNN algorithm correctly predicted the kidney stone risk of 34 patients out of 45 patients without kidney stone risk (Class A) and incorrectly predicted the kidney stone risk of 11 patients. It correctly predicted 25 patients out of 34 patients with kidney stone risk (Class B). A total of 59 patients were correctly predicted. The number of patients correctly classified of the algorithm increased to 171 after the SMOTE process was applied.

Figures shows that the number of patients correctly predicted of the algorithms increased after the SMOTE process.

The Rms value and accuracy rates calculated to determine the error rates in the predictions of the algorithms are given in Figure 10 and Figure 11, respectively.

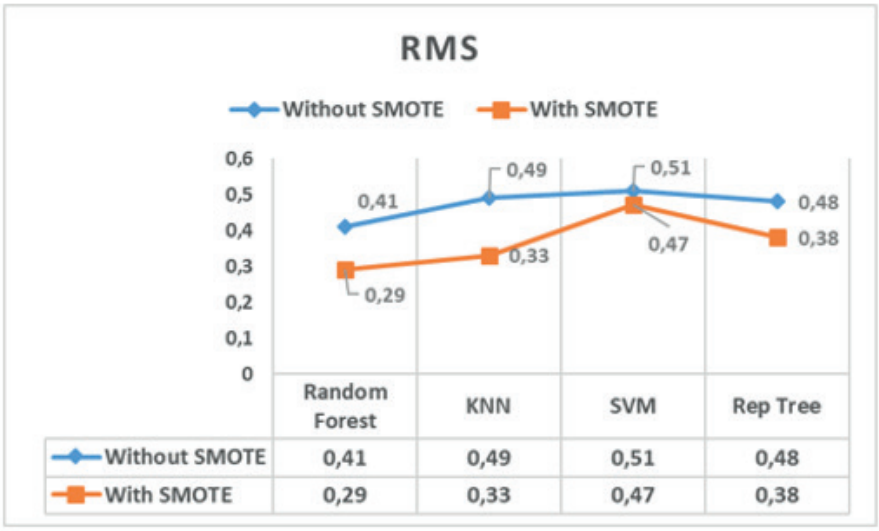


Figure 10. Accuracy of algorithms

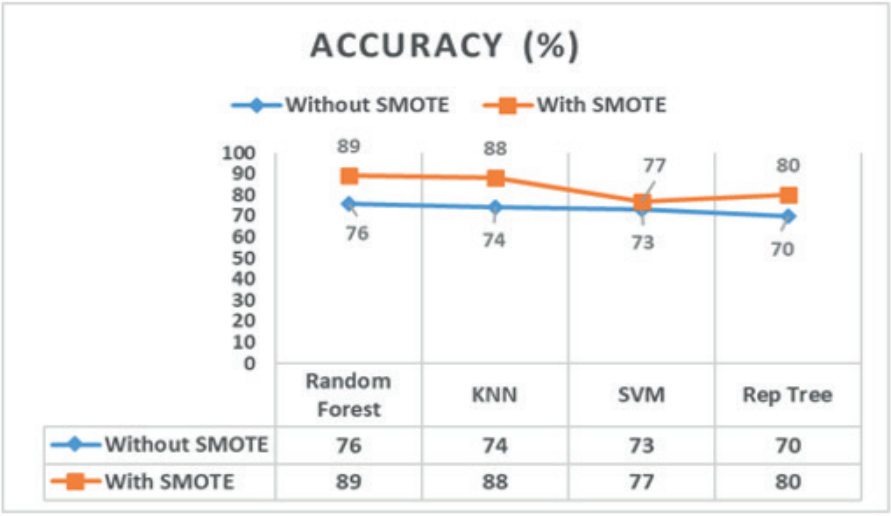


Figure 11. RMS of algorithms

Random forest algorithm is the algorithm that predicts with the least error rate (0.41). This error rate decreased to 0.29 with SMOTE. The highest value of accuracy (76%) was obtained when Random forest algorithm was used. This rate increased to 89% with SMOTE.

7. Discussion and Conclusion

In this chapter, the risk of developing kidney stones was estimated from the laboratory results of 89 patients. Patients were divided into two classes as at risk of developing kidney stones and not at risk of developing kidney stones.

The algorithms showed low performance due to the classes not having the same number of data. SMOTE was applied to increase the performance of the algorithm. Thus, both the class imbalance problem was solved and the number of data was increased. A significant increase was observed in all performance metrics calculated for all algorithms with SMOTE. The number of patients correctly predicted by the algorithms increased. The Random forest algorithm showed the best performance among the algorithms. Without SMOTE, Precision, Recall and F-score were calculated as 0.75. With SMOTE, Precision, Recall and F-score increased to 0.89. ROC increased to 0.95 and Kappa value increased to 0.78. At the same time, SMOTE provided a decrease in the RMS value of the algorithms and an increase in accuracy. With SMOTE, the rms value of random forest is 0.29 and the accuracy is 89%.

References

Abraham, A., Kavoussi, N. L., Sui, W., Bejan, C., Capra, J. A., & Hsi, R. (2022). Machine learning prediction of kidney stone composition using electronic health record-derived features. *Journal of Endourology*, *36*(2), 243-250.

Aksakallı, I., Kaçdıoğlu, S., & Hanay, Y. S. (2021). Kidney x-ray images classification using machine learning and deep learning methods. *Balkan Journal of Electrical and Computer Engineering*, *9*(2), 144-151.

Alghamdi, H., & Amoudi, G. (2024). Using Machine Learning for Non-Invasive Detection of Kidney Stones Based on Laboratory Test Results: A Case Study from a Saudi Arabian Hospital. *Diagnostics*, *14*(13), 1343.

Babajide, R., Lembrikova, K., Ziemba, J., Ding, J., Li, Y., Fermin, A. S., Fan, Y., & Tasian, G. E. (2022). Automated machine learning segmentation and measurement of urinary stones on CT scan. *Urology*, *169*, 41-46.

Batista, G. E., Prati, R. C., & Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD explorations newsletter*, 6(1), 20-29.

Caglayan, A., Horsanali, M. O., Kocadurdu, K., Ismailoglu, E., & Guneyli, S. (2022). Deep learning model-assisted detection of kidney stones on computed tomography. *International braz j urol*, *48*(5), 830-839.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: synthetic minority over-sampling technique. *Journal of artificial intelligence research*, 16, 321-357.

Chung, W.-Y., Falah Ramezani, R., A. Silverio, A., & F. Tsai, V. (2020). Development of a Portable Multi-Sensor Urine Test and Data Collection Platform for Risk Assessment of Kidney Stone Formation. *Electronics*, *9*(12), 2180.

Dai, M., Zhao, A., Liu, A., You, L., & Wang, P. (2013). Dietary factors and risk of kidney stone: a case–control study in southern China. *Journal of Renal Nutrition*, *23*(2), e21-e28.

Elton, D. C., Turkbey, E. B., Pickhardt, P. J., & Summers, R. M. (2022). A deep learning system for automated kidney stone detection and volumetric segmentation on noncontrast CT scans. *Medical Physics*, *49*(4), 2545-2554.

Ferraro, P. M., Taylor, E. N., Gambaro, G., & Curhan, G. C. (2017). Dietary and lifestyle risk factors associated with incident kidney stones in men and women. *The Journal of urology*, *198*(4), 858-863.

Gulhane, M., Kumar, S., Choudhary, S., Rakesh, N., Zhu, Y., Kaur, M., Tandon, C., & Gadekallu, T. R. (2024). Integrative approach for efficient detection of kidney

stones based on improved deep neural network architecture. *SLAS technology*, *29*(4), 100159.

He, H., & Garcia, E. A. 2009. Learning from imbalanced data. IEEE Transactions on knowledge and data engineering, 21(9), 1263-1284

Joseph, O., & Apena, W. O. (2021). Development of segmentation and classification algorithms for computed tomography images of human kidney stone. *Journal of Electronic Research and Application*, *5*(5), 1-10.

Kaggle Stone dataset. https://www.kaggle.com/datasets/harshghadiya/kidneystone/data

Khalili, P., Jamali, Z., Sadeghi, T., Esmaeili-Nadimi, A., Mohamadi, M., Moghadam-Ahmadi, A., Ayoobi, F., & Nazari, A. (2021). Risk factors of kidney stone disease: a cross-sectional study in the southeast of Iran. *BMC urology*, *21*, 1-8.

Kolli, C. S., Raghunath, M. P., Meenakshi, S., Maheswari, K., Britto, C. F., & Kushwaha, S. (2022). Efficient development of supervised learning algorithm for kidney stone prediction. 2022 International Conference on Inventive Computation Technologies (ICICT),

Manjunatha, D., Vishwakarma, V., Mishra, A., Ravindran, R. E., & Kumari, K. (2024). Kidney Stone Detection Using Ultrasonographic Images by Support Vector Machine Classification. *Nanotechnology Perceptions*, 93–106-193–106.

Manoj, B., Mohan, N., & Kumar, S. (2022). Automated detection of kidney stone using deep learning models. 2022 2nd International conference on intelligent technologies (CONIT),

M. Rajendiran, M. (2024). Predictive Analysis of Kidney Stone Formation Using Machine Learning Approaches, *African Journal of Biological Sciences*,6(7) (2024) 4149-4158

Patro, K. K., Allam, J. P., Neelapu, B. C., Tadeusiewicz, R., Acharya, U. R., Hammad, M., Yildirim, O., & Pławiak, P. (2023). Application of Kronecker convolutions in deep learning technique for automated detection of kidney stones with coronal CT images. *Information Sciences*, *640*, 119005.

Tahir, F. S., & Abdulrahman, A. A. (2023). Kidney stones detection based on deep learning and discrete wavelet transform. *Indonesian Journal of Electrical Engineering and Computer Science*, *31*(3), 1829.

Ventrella, P., Delgrossi, G., Ferrario, G., Righetti, M., & Masseroli, M. (2021). Supervised machine learning for the assessment of chronic kidney disease advancement. *Computer methods and programs in biomedicine*, *209*, 106329.

Vishmitha, D., Yoshika, K., Sivalakshmi, P., Chowdary, V., Shanthi, K., & Yamini, M. (2022). Kidney stone detection using deep learning and transfer learning. 2022 4th International Conference on Inventive Research in Computing

Applications (ICIRCA),

Yang, Y., Li, Y., Chen, R., Zheng, J., Cai, Y., & Fortino, G. (2021). Risk prediction of renal failure for chronic disease population based on electronic health record big data. *Big Data Research*, *25*, 100234.

Yildirim, K., Bozdag, P. G., Talo, M., Yildirim, O., Karabatak, M., & Acharya, U. R. (2021). Deep learning model for automated kidney stone detection using coronal CT images. *Computers in biology and medicine*, *135*, 104569.

CHAPTER 2

BOOSTING ALGORITHMS COMPARISON FOR PREDICTION OF ELECTRICAL RESISTIVITY OF COPPER-ZINC THIN FILMS BASED ON EXPERIMENTAL DATA

Hasan GÜLER[1] ,
Rasim ÖZDEMİR[2]

1 Assist. Prof. Dr., Kilis 7 Aralik University, Vocational High School of Technical Sciences, Department of Computer Technologies, Kilis, Turkey. ORCID: 0000-0001-5312-171X
2 Assoc. Prof. Dr., Kilis 7 Aralik University, Vocational High School of Technical Sciences, Department of Electrics, Kilis, Turkey. ORCID: 0000-0003-1439-0444

1. INTRODUCTION

Copper and zinc alloys (Cu-Zn), also known as brass, characteristically have good electrical conductivity and thermal properties. It is broadly used in electrical and electronic products (lighting, heating, etc. systems), and in the manufacturing industry due to its adhesion to steel and plastic, its suitability for decorative use, and its resistance to corrosion (Juškėnas et al. 2007). The physical properties of materials are directly related to their structure and composition. Producing alloys by adding different elements to the materials can lead to significant changes in their physical properties. Material characteristics such as softness, flexibility, malleability, hardness, castability, corrosion resistance, thermal and electrical conductivity, and electrical resistivity are required to be at certain values according to the industrial field where to be used. Especially electrical resistivity is an important parameter affecting the performance of materials used in electrical circuits and devices. Zinc and its alloys are known for their high corrosion resistance. Studies on the electrical resistivity of new alloys to be produced by adding new elements to these alloys can make important contributions to the literature and improve applications in the field of materials science as well. The computation of the electrical resistivity of materials is based on a formula that depends on the cross-sectional area, length, and type of material. Material production conditions are not included in the resistivity computation. In the electrodeposition method, the electrolyte pH value, environment temperature, amount of material added to the electrolyte, and corrosion resistance of the material have important effects on Cu-Zn alloy thin films produced. The analysis of the characteristics (resistivity, material content, grain size, etc.) of alloy thin films is conducted with various measuring instruments. The elemental analysis and grain size of the material can be measured by X-ray diffraction spectrometry (XRD), the distribution of the amount of material in the film can be measured by Energy Dispersive X-Ray Spectroscopy (EDX) analysis, and the resistivity changing with temperature can be measured with the four-point contact method and a cryostat. In experimental applications, material properties depend on various complex

factors such as manufacturing conditions and environmental variables. The solution of these nonlinear problems by deterministic approaches involves difficulties in both modeling and solution process due to the nature of the problem. However, there are many heuristics developed for solving such nonlinear problems (Keskintürk and Serap Şahin 2009) . Among these heuristics, machine learning (ML) is commonly used as a method that models (infer) a given problem based on the data obtained from the problem. With ML, classification, regression, and clustering tasks can be performed.

In the literature, there are few studies on the prediction of thin film properties with ML, but there is no study on the prediction of Cu-Zn thin film electrical resistivity with boosting methods, especially from current ML ensemble models. In this context, the study aims to predict the resistivity of Cu-Zn alloy thin films with boosting algorithms using real experimental data.

2. PREVIOUS STUDIES

Thin films are widely used in many fields such as electronics, optics, and protective coatings. However, the number of studies that use ML techniques to predict thin film properties is limited. This situation increases the importance and potential of research in this field. Studies emphasizing the effectiveness of ML in predicting thin film properties such as thin film thickness, mechanical properties, electrical properties have used various ML algorithms such as convolutional neural networks (CNNs), artificial neural networks, and random forests. Park, Lim, and Kim (2020) proposed a method using a modified VGG-16 network to measure semiconductor thin film thickness by learning the reflectance spectrum of four-layer thin films with respect to thickness and wavelength. Karade et al.(2023) determined important fabrication parameters such as i-ZnO thickness, Zn/Sn composition ratio and sulfo-cellenization temperature using ML to improve the efficiency of $Cu_2ZnSn(S, Se)_4$ (CZTSSe) thin film solar cells. With data collected from more than 25,000 data points, they obtained an adjusted R^2 approximately 0.96. Yoon et al.(2023) used a dataset of 625 samples produced under various conditions and achieved a prediction accuracy of 99.9%. They found that the XGBoost algorithm significantly outperformed other models,

including linear regression and random forests, in terms of accuracy and inference time. In their study, they emphasize the effectiveness of XGBoost in understanding complex relationships in thin film processing data. Nayak et al.(2023) studied the relationship between microhardness and abrasive wear in thin films and found that the specific wear rate of the coating decreases as the microhardness increases. They also developed a ML model that confirms the effect of microhardness and highlights its relationship with wear behavior. Wang et al.(2023) developed a three-dimensional manifold relating initial solution concentration, thin film coating thickness and monodisperse bulk molecular weight using curve-fitting ML. The model used the polystyrene bulk molecular weight and desired thin film thickness as input and an accurate prediction of the initial solution concentration required to form a coating of the desired thickness as output. Tseng et al. (2024) also used ML to obtain an optimal processing pulsed frequency in reactive pulsed DC sputtering of aluminum nitride films and compared its algorithms such as Random Forest, Categorical Boosting and Histogram Gradient Boosting in the analysis of process data. Ozdemir et al.(2024) used an experimental and simulation-based approach to evaluate the yield stress of thin film materials relevant to the integrated circuit (IC) industry. An artificial neural network (ANN) trained on synthetic data generated using finite element simulations predicted thin film yield stress from nano-indentation measurements. Nozawa et al.(2024) trained multichannel convolutional neural networks (ESA) models from electron backscatter diffraction (EBSD) data such as band contrasts, grain boundaries, and reverse polarity shapes to obtain highly accurate prediction of electrical properties of polycrystalline semiconductor thin films. They also studied how the ESA model learns the relationship between crystallinity, grain boundaries, crystallographic orientation, and carrier mobility by polarizing specific EBSD data and controlling estimated changes in carrier mobility. Abdullah et al. (2024) used Gradient Boosting and Adaptive Boosting methods to predict the unconfined compressive strength of geopolymer stabilized clay soil. Using a data set of 270 samples, they found R^2=0.98 for GB and R^2=0.975 for AdaBoost. Both methods showed satisfactory performance and high reliability.

3. MACHINE LEARNING BOOSTING ALGORITHMS

There are ML methods in the literature like support vector machines, logistic regression, naive bayes, k-nearest neighbor, random forest, and decision tree, etc. Boosting is the most widely used ML method because of its superior performance, scalability, and efficiency in handling complex data. Boosting is a powerful ensemble learning method in ML that aims to improve the predictive performance of models by additive combining multiple weak learners into a stronger model. Each weak learner is trained to correct the errors of its predecessor, resulting in higher accuracy of the overall model (Ustimenko and Prokhorenkova 2020; Guo and Zhang 2024; Kurugama et al. 2024) . In boosting models, weak learners are decision trees and are trained sequentially. Each new model focuses on the errors of the previous models and updates its weights accordingly. Commonly used boosting methods are: Gradient Boosting (GB), Hist Gradient Boosting (HGB), Adaptive Boosting (AdaBoost), Extreme Gradient Boosting (XGBoost), Light Gradient-Boosting Machine (LGBM) and Categorical Boosting (CatBoost). These methods can be used for both classification and continuous variable estimation (regressor) tasks.

3.1 Gradient Boosting

GB is an ensemble learning method that iteratively minimizes a differentiable loss function by sequentially adding decision trees. At each iteration, a new tree is trained to correct the model's errors, and the predictions of these trees are integrated into the overall model, scaled by a learning rate. This process continues until the prediction accuracy of the model improves (Tuychiev 2023; Masui 2024). The mathematical representation of GB model training can be summarized as follows (Friedman 2001):

- **Initialization:** Initial model $F_0(x)$ that begins with a simple estimate (average of target values) can be expressed as follows:
  $F_0(x) = \arg\min_c \sum_{i=1}^{n} L(y_i, c)$ where $L(y_i, c)$ represents loss function, y_i target and c a constant value. If loss function is mean square error (MSE), this can be simplified as follows:

$F_0(x) = \frac{1}{n}\sum_{i=1}^{n} y_i$

- At each iteration m,
  - a new weak learner $h_m(x)$ is added to the model to minimize the loss function.

  $F_m(x) = F_{m-1}(x) + \eta h_m(x)$ where η is the learning rate.
  - fit the new weak learner to the negative gradient of the loss function according to the predictions of the current model:

    $r_i^{(m)} = -\left[\frac{\partial L(y_i, F_{m-1}(x_i))}{\partial F_{m-1}(x_i)}\right]$
  - fits weak learner $h_m(x)$ to the negative gradient (residuals):

    $h_m(x) = arg\ \min_h \sum_{i=1}^{n} \left(r_{im} - h(x_i)\right)^2$
  - The model scaled by the learning rate is updated by adding new weak learners:

    $F_m(x) = F_{m-1}(x) + \eta h_m(x)$

3.2 Histogram Gradient Boosting

HGB is a variation of GB that uses histogram-based techniques to improve efficiency on large datasets. By dividing continuous input variables into a finite number of histogram bins, data complexity is reduced, and decision tree training is accelerated. This results in faster computation than standard GB and performs well on data with missing data as it can handle missing data naturally. HGB can be expressed mathematically as (Guryanov 2019) :

- **Initialization**: Initial model$F_0(x)$ begins with a simple estimate (average of target values) can be expressed as follows:

  $F_0(x) = \arg\min_c \sum_{i=1}^{n} L(y_i, c)$ where$L(y_i, c)$ represents loss function, y_i target and c a constant value. If loss function is mean square error (MSE), this can be simplified as follows:

  $F_0(x) = \frac{1}{n}\sum_{i=1}^{n} y_i$
- Continuous input features are split into k bins (histogram bins). For feature(x_j) , split can be expressed as follows:

  $b_j(x_i) = \text{binning function}(x_{ij})$ where $b_j(x_i)$ i represents the bin index for the first sample and j represents the bin index for the first feature. This reduces the number of unique values and speeds up the calculation of optimal split points.

- At each iteration (m=1, 2 ... M):
  - Gradients (pseudo residuals) of the loss function are calculated according to the estimates:
    $r_{im} = y_i - F_{m-1}(x_i)$
    For MSE loss:
    $r_{im} = -\left[\frac{\partial L(y_i, F_{m-1}(x_i))}{\partial F_{m-1}(x_i)}\right] = r_{im} = y_i - F_{m-1}(x_i)$
  - For each j feature, histograms of gradients and their corresponding values are generated:
    $H_{j,k} = \sum_{i \in \text{bin } k} r_{im}$ $C_{j,k} = \sum_{i \in \text{bin } k} 1$ where $H_{j,k}$ is the sum of the gradients for the histogram bin k of the feature$j$ and $C_{j,k}$, k is the number of samples in the histogram bin.
  - For each feature$j$, it finds the best split point by evaluating the potential split based on histogram bins. Split is chosen to minimize the loss function. The optimal split point for histogram bin k is found as follows:
    $\text{BestSplit}(j) = \arg\min_k \left[\frac{H_{j,k}^2}{C_{j,k}}\right]$ This splitting minimizes the weighted sum of the gradients in the left and right child nodes.
  - Once the best split is found for all features, the tree is created, and the model is updated:
    $F_m(x) = F_{m-1}(x) + \eta h_m(x)$ where $h_m(x)$ is the new weak learner (tree) generated using the best split and η is the learning rate.

3.3 Adaptive Boosting

AdaBoost is an ensemble learning algorithm designed to improve the prediction accuracy of its models. The algorithm applies weak learners to the training data by increasing the weight of incorrectly predicted instances and decreasing the weight of correctly predicted instances at each iteration. This weight adjustment helps subsequent models to learn more difficult examples. The final prediction is calculated as a weighted sum of the predictions from all weak learners. The mathematical representation of GB model training can be summarized as (Hastie, Tibshirani, and Friedman 2009) :

- **Initialization**: The algorithm starts by initially assigning equal weights to all training examples:

$w_i^{(1)} = \frac{1}{n}$ for $i = 1,2, \dots, n$ where n represents the number of training samples.

- $from\ m = 1\ to\ n$:
  - A weak learner $h_m(x)$ is trained on the weighted training data.
  - The weighted error of the weaker learner is calculated: $e_m = \frac{\sum_{i=1}^{N} w_i^{(m)} |y_i - h_m(x_i)|}{\sum_{i=1}^{N} w_i^{(m)}}$ where($|y_i - h_m(x_i)|$) is the difference between the actual value and the estimate.
  - The weight of the weaker learner is calculated based on its error: $\alpha_m = \frac{1}{2} \ln\left(\frac{1-e_m}{e_m}\right)$
  - The weights of the training examples are updated: $w_i^{(m+1)} = w_i^{(m)} \cdot \exp(\alpha_m \cdot |y_i - h_m(x_i)|)$
  - Weights are normalized so that their sum is 1: $w_i^{(m+1)} = \frac{w_i^{(m+1)}}{\sum_{i=1}^{N} w_i^{(m+1)}}$

3.4 Extreme Gradient Boosting

XGBoost is a GB algorithm that provides superior performance and efficiency on large datasets and complex prediction tasks. XGBoost uses regularization functions to prevent overfitting and applies advanced tree pruning techniques. It is fast thanks to its parallel computing capability. These features make XGBoost a preferred choice for tasks that require high accuracy and speed. XGBoost model training can be mathematically summarized as follows (Analytics Vidhya 2018; GeeksforGeeks 2023) :

- **Initialization**: Starts with an initial estimate, which is usually an average of the target values: $F_0(x) = \frac{1}{n} \sum_{i=1}^{n} y_i$
- XGBoost minimizes a regularized objective function: $\mathcal{L}(\theta) = \sum_{i=1}^{n} l(y_i, \hat{y_i}) + \sum_{k=1}^{K} \Omega(f_k)$ here:
  - l is the loss function (e.g., squared error for regression).
  - $\hat{y_i} i$ is the prediction for the fifth sample.
  - $\Omega(f_k) k$ is a regularization term for the ninth tree.
- The term regularization helps prevent over-compliance and includes the number of leaves on the tree$T$ and the norm of leaf weightsL_2 :

$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$ where γ and λ are regularization parameters and w_j is the weight of the leaf j .

- At each iteration (t=1, 2, ... T),
  - A new treef_t is added to the model to minimize the objective function:
    $F_t(x) = F_{t-1}(x) + f_t(x)$
  - calculate the gradientg_i and Hessianh_i for each sample$i$:
    $g_i = \frac{\partial l(y_i, \hat{y_i})}{\partial \hat{y_i}}$ $h_i = \frac{\partial^2 l(y_i, \hat{y_i})}{\partial \hat{y_i^2}}$
  - To build the tree, XGBoost uses an algorithm to find the best split by maximizing the gain in the loss function. The gain from the split of left and right nodes $Gain$ is as follows:
    $Gain = \frac{1}{2}\left[\frac{(\sum_{i\in \text{left}} g_i)^2}{\sum_{i\in \text{left}} h_i + \lambda} + \frac{(\sum_{i\in \text{right}} g_i)^2}{\sum_{i\in \text{right}} h_i + \lambda} - \frac{(\sum_{i\in \text{all}} g_i)^2}{\sum_{i\in \text{all}} h_i + \lambda}\right] - \gamma$
  - Once the optimum separations have been determined, the weight of each leaf is calculated as w_j :
    $w_j = -\frac{\sum_{i\in j} g_i}{\sum_{i\in j} h_i + \lambda}$
  - the model is updated by adding a new tree:
    $F_t(x) = F_{t-1}(x) + \eta f_t(x)$ where η is the learning rate.

3.5 Light Gradient Boosting Machine,

LightGBM is a GB algorithm developed by Microsoft. Unlike traditional methods, LightGBM uses innovative approaches such as leaf-based tree growth strategy, histogram-based learning, and Gradient-Based One-Side Sampling (GOSS). These approaches result in deeper trees, faster training time, reduced memory usage and increased model accuracy. LightGBM also reduces data dimensionality with Exclusive Feature Bundling (EFB), automatically processes categorical features and is compatible with distributed computing environments. The mathematical representation of the LightGBM model training is (Ke et al. 2017):

- **Initialization:** Starts with an initial estimate, which is usually an average of the target values:
  $F_0(x) = \frac{1}{n}\sum_{i=1}^{n} y_i$
- XGBoost minimizes a regularized objective function:
  $\mathcal{L}(\theta) = \sum_{i=1}^{n} l(y_i, \hat{y_i}) + \sum_{k=1}^{K} \Omega(f_k)$ here:
  - l is the loss function (e.g., squared error for regression).

- $\widehat{y_i}i$ is the prediction for the fifth sample.
- $\Omega(f_k)k$ is a regularization term for the ninth tree.

- Continuous input features are split into k bins (histogram bins). For feature (x_j) , the split can be expressed as follows:
  $b_j(x_i) = \text{binning function}(x_{ij})$ where $b_j(x_i)$ i represents the bin index for the first sample and j represents the bin index for the first feature. This reduces the number of unique values and speeds up the calculation of optimal split points.
- At each iteration (t=1, 2, ..., T):
  - calculate the gradientg_i and Hessianh_i for each sample$i$:
    $g_i = \left[\frac{\partial L(y_i, F_{t-1}(x_i))}{\partial F_{t-1}(x_i)}\right]$ $\quad$ $h_i = \left[\frac{\partial^2 L(y_i, F_{t-1}(x_i))}{\partial F_{t-1}(x_i)^2}\right]$
  - For each j feature, histograms of gradients and their corresponding values are generated:
    $H_{j,k} = \sum_{i \in \text{bin } k} g_i$, $S_{j,k} = \sum_{i \in \text{bin } k} h_i$ where $H_{j,k}$ is the sum of the gradients for the histogram bin k of the feature j and $S_{j,k}, k$ is the number of samples in the histogram bin.
  - LightGBM uses a leaf-based growth strategy. For each feature j, it finds the best split point by evaluating potential split based on histogram bins.
    $G_{split} = \frac{(H_{j,left})^2}{S_{j,left}} + \frac{(H_{j,right})^2}{S_{j,right}} - \frac{(H_j)^2}{S_j}$ where G_{split} represents the gain from the split.
  - Once the optimum split has been determined, the weight of each leaf w_j is calculated:
    $w_j = -\frac{H_j}{S_j + \lambda}$
  - The model is updated by adding the new tree:
    $F_t(x) = F_{t-1}(x) + \eta f_t(x)$ where η is the learning rate.

3.6 Categorical Boosting

CatBoost is an open-source GB algorithm developed by Yandex, specifically designed for datasets containing categorical variables. It uses unique techniques such as sequential boosting and symmetric decision trees to reduce overfitting and improve computational efficiency. CatBoost's ability to automatically handle categorical variables eliminates the need for pre-processing, making it easy to use. Its high performance, scalability, and minimal parameter tuning requirements make it a favorite in a variety of

fields. The mathematical representation of CatBoost model training is as follows (Prokhorenkova et al. 2018) :

- **Initialization:** The model starts with an initial estimate, which is usually the average of the target values:
  $F_0(x) = \arg\min_c \sum_{i=1}^{n} L(y_i, c)$ where $L(y_i, c)$ represents the loss function, y_i the target and c a constant value.
- For MSE loss function, this is simplified as follows:
  $F_0(x) = \frac{1}{n}\sum_{i=1}^{n} y_i$
- At each iteration (t=1, 2, ..., T):
  - calculate the gradientg_i and Hessianh_i for each sample$i$:
    $g_i = \left[\frac{\partial L(y_i, F_{t-1}(x_i))}{\partial F_{t-1}(x_i)}\right]$ $\quad$ $h_i = \left[\frac{\partial^2 L(y_i, F_{t-1}(x_i))}{\partial F_{t-1}(x_i)^2}\right]$
  - Using a combination of One-Hot coding and mean coding, categorical characteristics are converted into numerical values:
    $\text{NewFeature}_{ij} = \text{Transform}\left(\text{CategoricalFeature}_j\right)$
  - It uses permutations of data to ensure that the model is built in an ordered fashion and to reduce overfitting:
    $\{(x_{i1}, y_{i1}), (x_{i2}, y_{i2}), \ldots, (x_{in}, y_{in})\}$
  - For each feature, it finds the best split by evaluating potential split using gradients and Hessians:
    $Gain(j,k) = \frac{1}{2}\left[\frac{(\sum_{i\in\text{left}} g_i)^2}{\sum_{i\in\text{left}} h_i+\lambda} + \frac{(\sum_{i\in\text{right}} g_i)^2}{\sum_{i\in\text{right}} h_i+\lambda} - \frac{(\sum_{i\in\text{all}} g_i)^2}{\sum_{i\in\text{all}} h_i+\lambda}\right] - \gamma$
  - Once the best splits are identified, the weight for each leaf is calculated.
    $w_j = -\frac{\sum_{i\in j} g_i}{\sum_{i\in j} h_i+\lambda}$
  - Updates the model by adding the new tree:
    $F_t(x) = F_{t-1}(x) + \eta f_t(x)$ where η is the learning rate.
  - Regularization L_2 is the norm:
    $L_2 = (\lambda \sum_{j=1}^{T} w_j^2)$

4. MATERIALS AND METHODS

4.1 Experimental Study

Cu-Zn alloy thin films were fabricated as coatings on aluminum substrates using electrochemical deposition system. Various experiments were performed to investigate the structural and surface characterization of thin

films. Cyclic voltammetry (CV) experiments, scanning electron microscopy (SEM) images, X-ray diffraction spectrometry (XRD) measurements were performed to determine the deposition potential using a potentiostat-galvanostat. The distribution of the amount of material in the film was analyzed by Energy Dispersive X-Ray Spectroscopy (EDX) analysis. Electrical resistivity was measured using the four-point contact method. Electrical resistivity measurement is performed using different methods depending on the resistance value. Especially when measuring low resistance values such as resistivity, device errors in the measuring system, resistance of the connecting conductors and contact resistances can affect the measurement result and lead to erroneous values. Two-point contact and four-point contact methods have been developed to minimize these errors. These methods, which are frequently used in the measurement of the electrical properties of thin metal films and semiconductor materials, are based on the measurement of the current flow through the sample and the potential difference in a specific region. The four-point contact method is widely preferred because it provides more precise measurements. The Van Der Pauw method for measuring resistivity is one of these methods (Wikipedia 2024) . In the Van Der Pauw method, resistivity measurements of free-form planar plate-shaped materials can be made. The measurement is performed by ohmic contacts at four points on the edge of the sample.

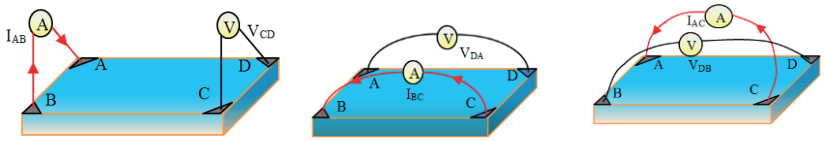


Figure 1. Van Der Pauw Method measurement patterns

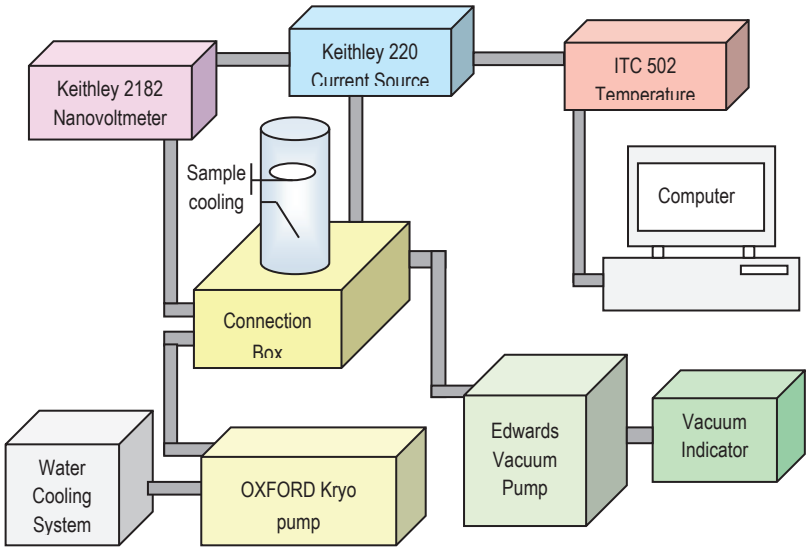


Figure 2. Computer controlled closed circuit Helium cooled four-point resistivity measurement setup

To calculate the resistivity in four-point contact measurement by the van der Pauw method, electrodes are brought into contact with the sample surface at points A, B, C and D as shown in Figure 1. The basis of these methods is the measurement of the electric potential difference in a specific region of the sample as current flows through the sample to be measured. The measured current, voltage (I, V) values are used to calculate the electrical properties. The resistivity measurement process, the temperature varied between 100 K and 350 K with the help of a cryostat and is conducted under computer control using the Labview program for fully automation (Figure 2). Data was recorded at each increase of ten °C. The measurement process of one sample takes approximately 6 hours. The production conditions of the thin films produced by the electrochemical deposition and the experimental data of the bath content of 5 thin films prepared are given in Table 1. As seen in Table 1, copper sulfate ($CuSO_45H_2O$) between 0.06-0.1 $mol*lt^{-1}$, zinc sulfate ($ZnSO_47H_2O$) between 0.2 $mol*lt^{-1}$, sodium citrate (Na(3)C(6)O) between 0.5 $mol*lt^{-1}$ sodium citrate ($Na_3C_6H_5O_7$), pH value 5.8, film production time 60 minutes, production current between anode and cathode 60 mA, electrolyte temperature between 20-50 ℃.

Table 1. Bath contents and experimental data of CuZn thin film alloys

SN	Materials in the electrolyte			pH	Current (mA)	Time Dk.	Electrolyte Temperature (°C)
	$CuSO_45H_2O$ (Mol/lt)	$ZnSO_47H_2O$ (Mol/lt)	$Na_3C_6H_5O_7$ (Mol/lt)				
$Zn_{17}Cu_{83}$	0,06	0,2	0,5	5,8	60	60	20
$Zn_{36}Cu_{64}$	0,08	0,2	0,5	5,8	60	60	20
$Zn_{48}Cu_{52}$	0,10	0,2	0,5	5,8	60	60	20
$Zn_{50}Cu_{50}$	0,06	0,2	0,5	5,8	60	60	30
Zn79Cu21	0,06	0,2	0,5	5,8	60	60	50

4.2 Resistivity measurement data:

The electrical resistivity measurement results of five Cu-Zn thin film alloys are given in Figure 3. That shows the electrical resistivity change according to the sequence for experiments realized. The electrical resistivity change depending on the environment temperature changed during the measurement is in Figure 4 for all samples. The temperature varied between 100 K (Kelvin) and 350 K.

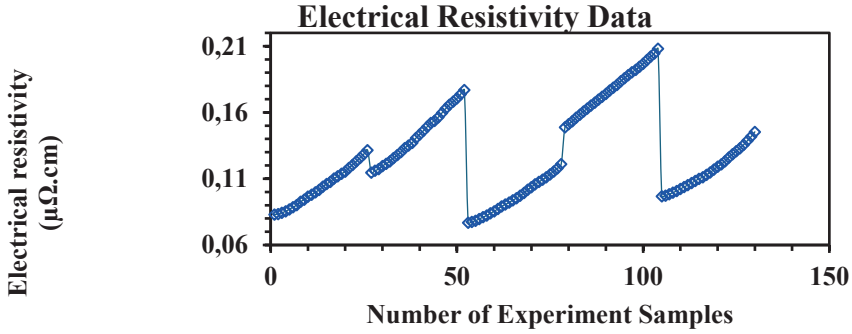


Figure 3. Electrical resistivity changes according to experimental sequence

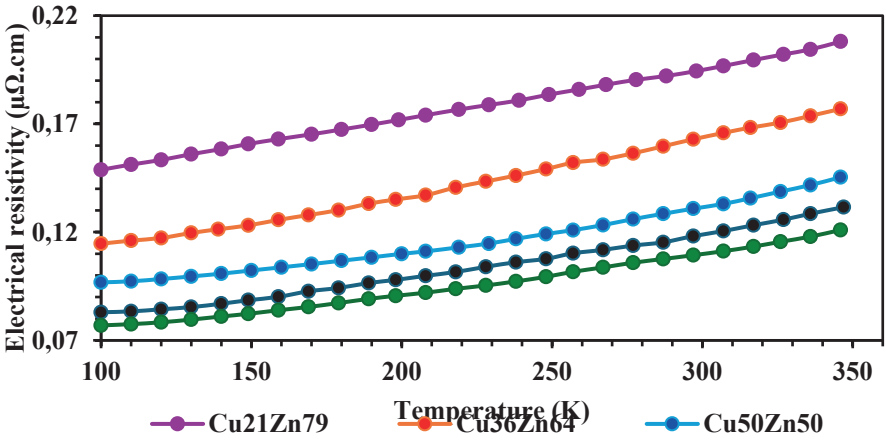


Figure 4. Electrical resistivity changes according to environment temperature during the measurement

4.3 XRD Measurements

The crystal structures of Cu-Zn alloy films were determined from XRD analysis data. With this data, the grain size was calculated using the Debye-Scherrer formula. Scherrer equation $L = 0.9\lambda(B.Cos\theta)^{-1}$ was used to estimate the average grain size in the direction normal to the film plane where L is the grain size, $2^{*}\theta$ is the diffraction peak, B is the half height width and λ is the wavelength (Leontyev et al. 2018; Ozdemir and Karahan 2023) .

4.4 Dataset Preparation

A dataset was created from our experimental data for the prediction of electrical resistivity with ML boosting methods. This dataset includes various features related to the production of Cu-Zn alloy thin film. These include electrochemical deposition conditions, bath component and thin film composition ratios, grain size and temperature varying resistivity values of thin films. This data set consists of 130 samples with eight input features affecting resistivity and resistivity value as output.

Table2. Statistical representation of the dataset

	T	%ECu	%EZn	%FCu	%FZn	Temperature	Volt	Grain Size	Resistivity
Average	223.25	26.24	73.76	35.48	64.52	28.00	3.11	80.30	0.13
Standard Deviation	74.03	4.14	4.14	14.34	14.34	11.71	0.21	14.64	0.03
The smallest	100.00	23.10	66.70	17.30	47.70	20.00	2.73	66.09	0.08
25%	159.00	23.10	71.40	21.40	50.10	20.00	3.02	69.67	0.10
50%	223.50	23.10	76.90	36.50	63.50	20.00	3.24	69.72	0.12
75%	287.00	28.60	76.90	49.90	78.60	30.00	3.27	95.91	0.15
Highest	347.00	33.30	76.90	52.30	82.70	50.00	3.29	100.10	0.21

The data set contains various data related to thin film production and resistivity measurement. The input characteristics are: 1) Cu content in the electrolyte (ECu %), 2) Zn content in the electrolyte (EZn %), 3) Cu content in the thin film (FCu %), 4) Zn content in the thin film (FZn %), 5) grain size (nm) (G), calculated by the Debye- Scherrer formula, 6) application voltage (Vuyg), 7) electrolyte temperature (St °C) and 8) ambient temperature (T Kelvin) during the measurement. The output variable is the electrical resistivity (μΩ.cm) (q). Table 2 shows the statistical characteristics of the data set.

In training ML models, variability in the data set can affect the convergence time and prediction accuracy of the model (Zhou and McArdle 2015; Jang, Jeong, and Cho 2021) . To reduce this effect, standardization (known as z-score) is applied to the dataset (Al-Mekhlafi, Klawitter, and Klawonn 2024) . By adjusting the data to have zero mean and one standard deviation, standardization eliminates biases that can be caused by different scales and allows the model to learn more efficiently. The formula for standardization is as follows: $z = \frac{(X-\mu)}{\sigma}$ z, x, μ, σ represents the standardized, original, mean, and standard deviation, respectively.

5. RESULTS AND DISCUSSION

5.1 Model Education

In this study, the grid search method is used to determine the hyperparameters of the ML boosting methods. This method systematically adjusts hyperparameters to optimize model performance (Wu et al., 2019). The overfitting problem can cause the model to memorize the training data and perform poorly on new data. To avoid this problem, 4-fold cross-validation was applied for the training data (60% of the dataset) (Han, Marimuthu, and Lee 2022) . These operations were performed in Python using boosting regressor packages (CatBoost 2024; LightGBM 2024; scikit-learn 2024; XGBoost 2024) .

5.2 ML Boosting Methods Analysis

The boosting regressor models with the best parameters obtained by 4-fold cross-validation with the grid search method were identified. The performance metrics of these models for training and test data are given in Table 3.

Table 3. ML Model performances

	Education		**Test**	
Boosting Models	**MAPE**	**R2**	**MAPE**	**R2**
Gradient Boosting	0.0045	0.9994	0.0195	0.9924
Hist Gradient Boosting	0.0174	0.9938	0.0296	0.9806
AdaBoost	0.0369	0.9778	0.0491	0.9508
XGBoost	**0.0053**	**0.9994**	**0.0126**	**0.9962**
LGBM	0.027	0.9865	0.0343	0.9738
CatBoost	**0.0017**	**0.9999**	**0.0062**	**0.999**

As seen in Table 3, CatBoost and XGBoost regressor models are the best forecasting models. The metric values of the test data particularly indicate that the models can generalize and make reliable predictions on new data sets. Scatter (Figures 5, 6), line (Figures 7, 8) and comparison (Figures 9, 10) plots of the electrical resistivity training and test data are given for the predicted values by the XGBoost and CatBoost models and actual values.

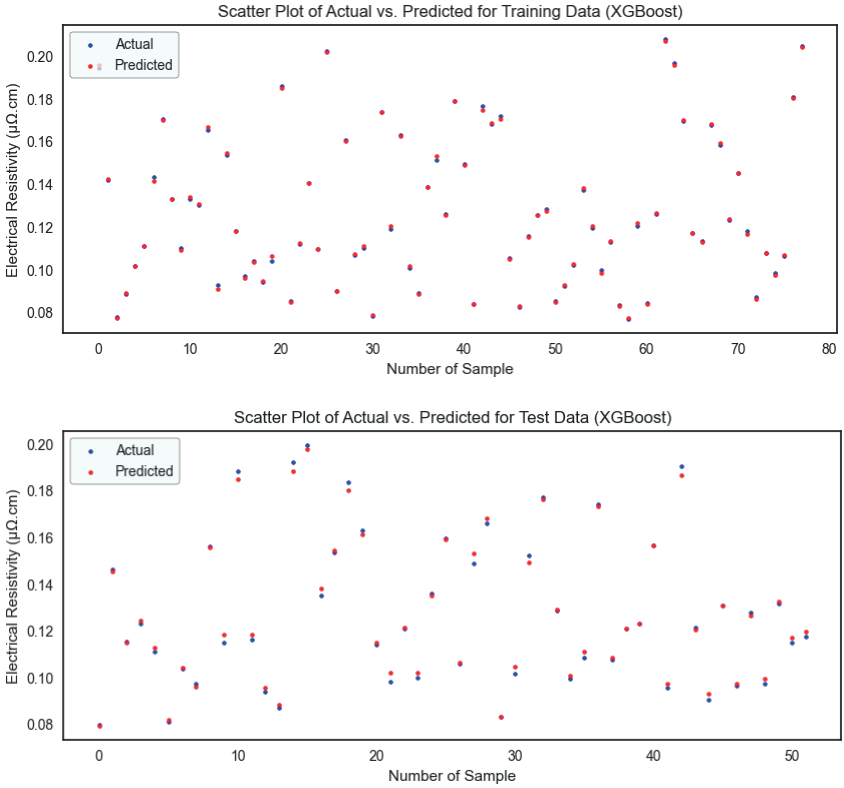
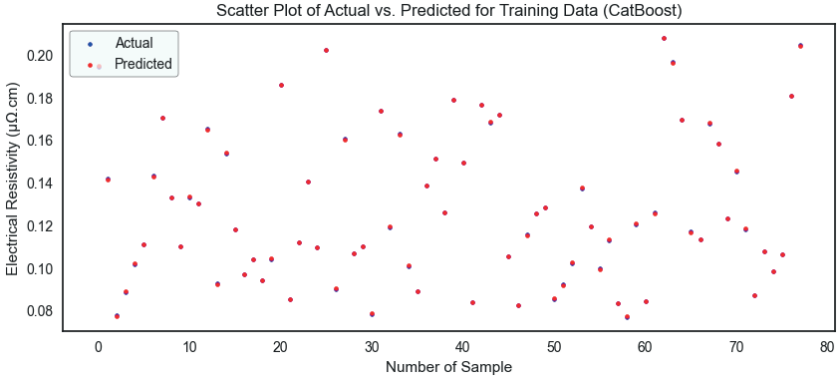


Figure 5. Scatter plot of actual and XGBoost model predictions of electrical resistivity training and test data



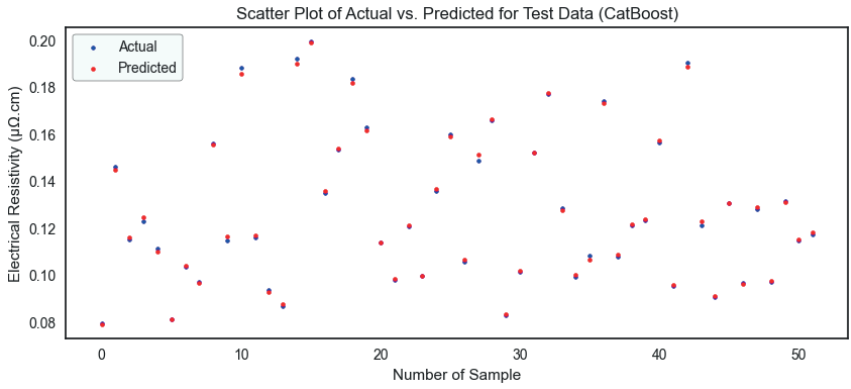


Figure 6. Scatter plot of actual and CatBoost model predictions of electrical resistivity training and test data

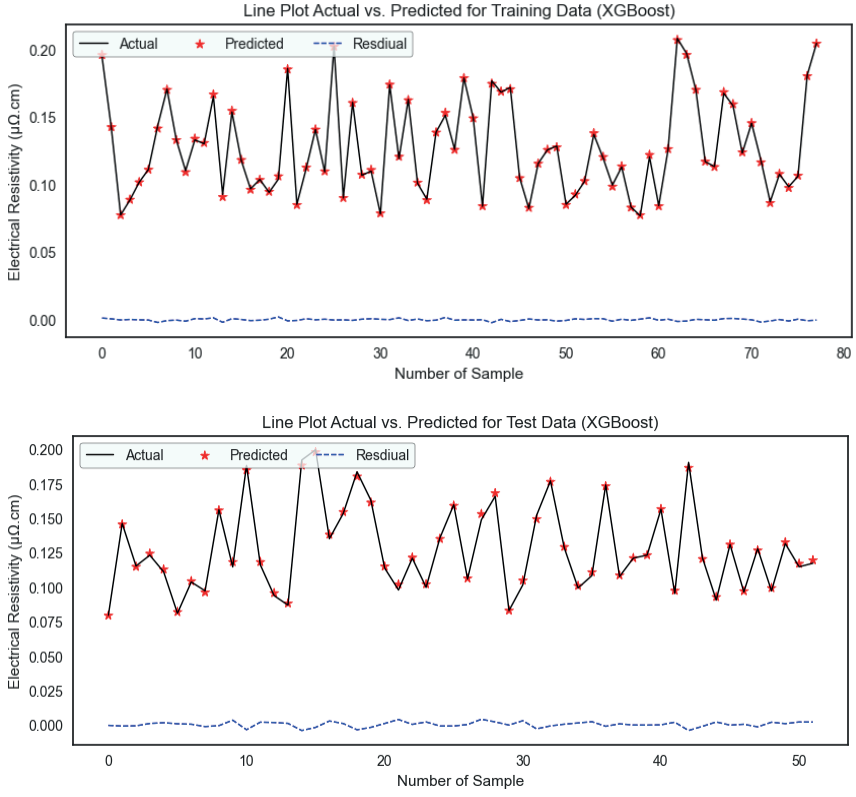


Figure 7. Line plot of actual and XGBoost model predictions of electrical resistivity training and test data

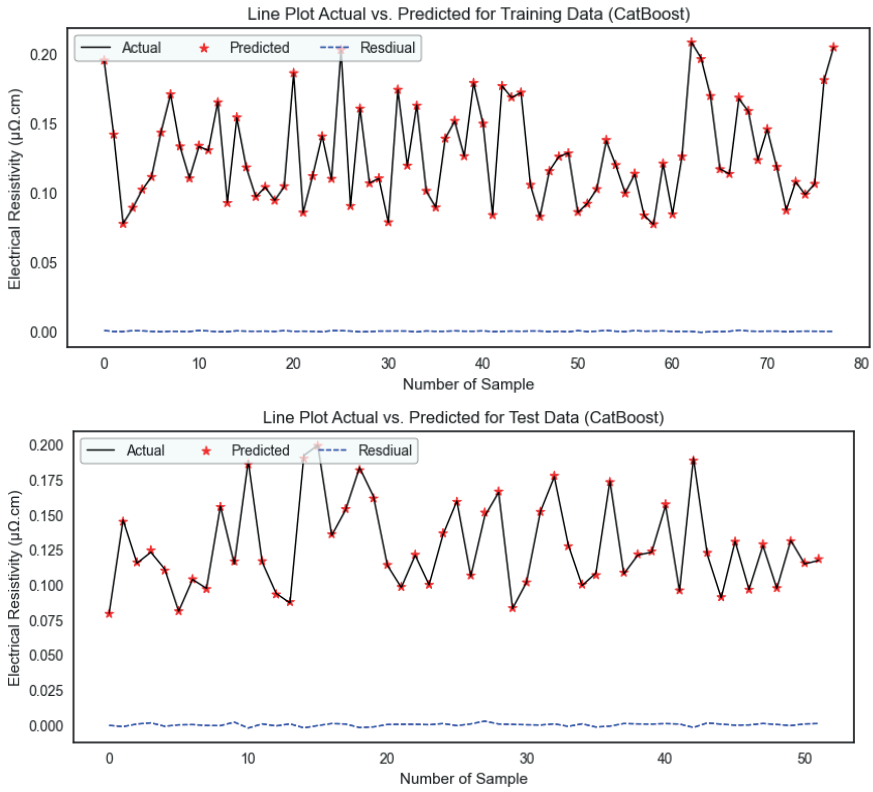
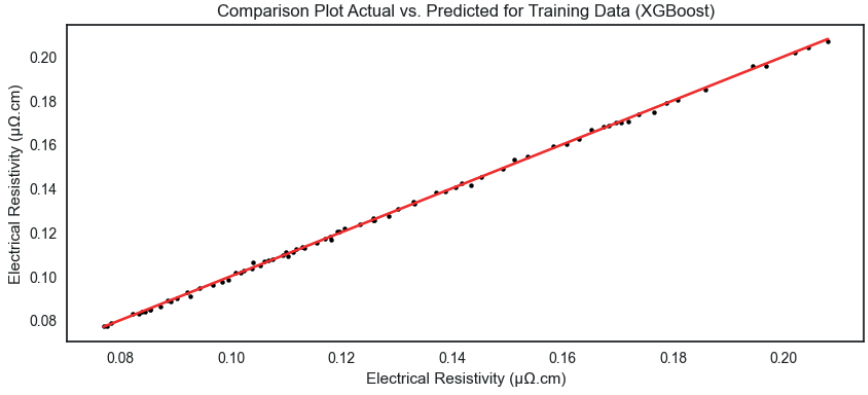


Figure 8. Line plot of actual and CatBoost model predictions of electrical resistivity training and test data



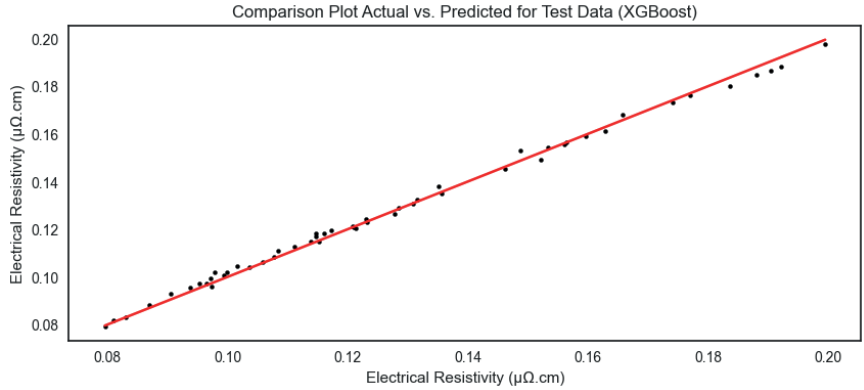


Figure 9. Comparison plot of actual and XGBoost model predictions of electrical resistivity training and test data

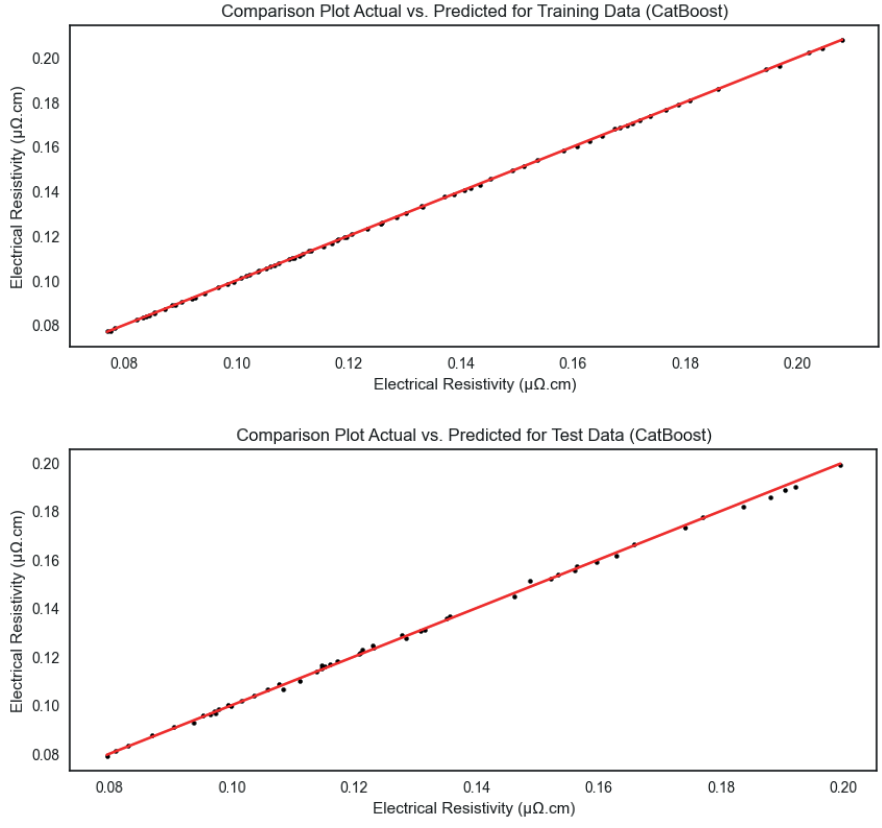


Figure 10. Comparison plot of actual and CatBoost model predictions of electrical resistivity training and test data

In this study, the electrical resistivity estimation performances of XGBoost and CatBoost models were analyzed, and it was observed that both models showed high accuracy rates. The XGBoost model achieved high accuracy values such as R^2=0.9994 and MAPE=0.0053 for training data and R^2=0.9962 and MAPE=0.0126 for test data. Similarly, the CatBoost model achieved high accuracy values of R^2=0.9999 and MAPE=0.0017 for training data and R^2=0.999 and MAPE=0.0062 for test data. The scatter plots presented in Figures 5 and 6 show that both models produce predictions close to the actual values. The line graphs presented in Figures 7 and 8 support that the models produce predictions remarkably close to the actual values for both training and test data. Finally, the comparison plots shown in Figures 9 and 10 prove that both models produce predictions close to the actual values and deviate little from the actual values.

6. CONCLUSION

This paper presents an evaluation from experimental data to compare ML boosting algorithms for determining the electrical resistivity of Cu-Zn alloy thin films depending on their fabrication conditions and structural properties. Six boosting algorithms were implemented and evaluated using our real experimental dataset for comparison. The best parameters of these six algorithms were identified by grid search and 4-fold cross-validation using 60% of the dataset. These algorithms were assessed with the rest of the dataset. Among these models, XGBoost (training R2=0.999, test R2=0.996) and CatBoost (training R2=0.999, test R2=0.996) performed the best for both training and testing. This showed that both models produced predictions close to the actual values and are reliable for electrical resistivity estimation. Boosting algorithms can provide solutions for complex problems such as production conditions and reverse design.

7. REFERENCES

Abdullah, Gamil M. S., Mahmood Ahmad, Muhammad Babur, Muhammad Usman Badshah, Ramez A. Al-Mansob, Yaser Gamil, and Muhammad Fawad. 2024. "Boosting-Based Ensemble Machine Learning Models for Predicting Unconfined Compressive Strength of Geopolymer Stabilized Clayey Soil". *Scientific Reports* 14(1):2323. doi: 10.1038/s41598-024-52825-7.

Al-Mekhlafi, Amani, Sandra Klawitter, and Frank Klawonn. 2024. "Standardization with Zlog Values Improves Exploratory Data Analysis and Machine Learning for Laboratory Data". *Journal of Laboratory Medicine* 48(5):215-22. doi: 10.1515/labmed-2024-0051.

Analytics Vidhya. 2018. "What Is XGBoost Algorithm?" *Analytics Vidhya*. Accessed December 19, 2024 (https://www.analyticsvidhya.com/blog/2018/09/an-end-to-end-guide-to-understand-the-math-behind-xgboost/).

CatBoost. 2024. "Tutorials". *CatBoost*. Accessed December 20, 2024 (https://catboost.ai/docs/en/concepts/en/concepts/tutorials).

Friedman, Jerome H. 2001. "Greedy Function Approximation: A Gradient Boosting Machine". *The Annals of Statistics* 29(5):1189-1232.

GeeksforGeeks. 2023. "XGBoost". *GeeksforGeeks*. Accessed December 19, 2024 (https://www.geeksforgeeks.org/xgboost/).

Guo, Siqi, and Boruo Zhang. 2024. "Revolutionizing the used car market: Predicting prices with XGBoost". *Applied and Computational Engineering* 48(1):173-80. doi: 10.54254/2755-2721/48/20241349.

Guryanov, Aleksei. 2019. "Histogram-Based Algorithm for Building Gradient Boosting Ensembles of Piecewise Linear Decision Trees". Pp. 39-50 in *Analysis of Images, Social Networks and Texts*. C. 11832, *Lecture Notes in Computer Science*, edited by W. M. P. Van Der Aalst, V. Batagelj, D. I. Ignatov, M. Khachay, V. Kuskova, A. Kutuzov, S. O. Kuznetsov, I. A. Lomazova, N. Loukachevitch, A. Napoli, P. M. Pardalos, M. Pelillo, A. V. Savchenko, and E. Tutubalina. Cham: Springer International Publishing.

Han, Giyeol, Karuppasamy Pandian Marimuthu, and Hyungyil Lee. 2022. "Evaluation of Thin Film Material Properties Using a Deep Nanoindentation and ANN". *Materials & Design* 221:111000. doi: 10.1016/j.matdes.2022.111000.

Hastie, Trevor, Robert Tibshirani, and J. H. Friedman. 2009. *The elements of statistical learning: data mining, inference, and prediction*. 2nd ed. New York, NY: Springer.

Jang, Youjin, Inbae Jeong, and Yong K. Cho. 2021. "Identifying Impact of Variables in Deep Learning Models on Bankruptcy Prediction of Construction Contractors". *Engineering, Construction and Architectural Management* 28(10):3282-98. doi: 10.1108/ECAM-06-2020-0386.

Juškėnas, R., V. Karpavičienė, V. Pakštas, A. Selskis, and V. Kapočius. 2007. "Electrochemical and XRD Studies of Cu-Zn Coatings Electrodeposited in Solution with d-Mannitol". *Journal of Electroanalytical Chemistry* 602(2):237-44. doi: 10.1016/j.jelechem.2007.01.004.

Karade, Vijay C., Santosh S. Sutar, Seung Wook Shin, Mahesh P. Suryawanshi, Jun Sung Jang, Kuldeep Singh Gour, Rajanish K. Kamat, Jae Ho Yun, Tukaram D. Dongale, and Jin Hyeok Kim. 2023. "Machine Learning Assisted Analysis, Prediction, and Fabrication of High-Efficiency CZTSSe Thin Film Solar Cells". *Advanced Functional Materials* 33(41):2303459. doi: 10.1002/adfm.202303459.

Ke, Guolin, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. "LightGBM: A Highly Efficient Gradient Boosting Decision Tree". in *Advances in Neural Information Processing Systems*. C. 30. Curran Associates, Inc.

Keskintürk, Timur and Serap Şahin. 2009. "Genetic Algorithm with Real Value Coding in Nonlinear Regression Analysis". *Istanbul Commerce University Journal of Social Sciences* 8(15):167-78.

Kurugama, Kumudu Madhawa, So Kazama, Yusuke Hiraga, and Chaminda Samarasuriya. 2024. "A Comparative Spatial Analysis of Flood Susceptibility Mapping Using Boosting Machine Learning Algorithms in Rathnapura, Sri Lanka". *Journal of Flood Risk Management* 17(2):e12980. doi: 10.1111/jfr3.12980.

Leontyev, Igor N., Alexandra B. Kuriganova, Mathieu Allix, Aydar Rakhmatullin, Pavel E. Timoshenko, Olga A. Maslova, Alexey S. Mikheykin, and Nina V. Smirnova. 2018. "On the Evaluation of the Average Crystalline Size and Surface Area of Platinum Catalyst Nanoparticles". *Physica Status Solidi (b)* 255(10):1800240. doi: 10.1002/pssb.201800240.

LightGBM. 2024. "Welcome to LightGBM's documentation! - LightGBM 4.5.0 documentation". *LightGBM*. Accessed December 20, 2024 (https://lightgbm.readthedocs.io/en/stable/).

Masui, Tomonori. 2024. "All You Need to Know about Gradient Boosting Algorithm - Part 1. Regression". *Medium*. Accessed December 18, 2024 (https://towardsdatascience.com/all-you-need-to-know-about-gradient-boosting-algorithm-part-1-regression-2520a34a502).

Nayak, Haridasa, K. Shanthala, M. K. Venkatesh, and L. Chirag. 2023. "Correlation of Microhardness with Abrasive Wear of Thin Films: A Machine Learning Approach". Pp. 1-5 in *2023 International Conference on the Confluence of Advancements in Robotics, Vision and Interdisciplinary Technology Management (IC-RVITM)*.

Nozawa, Koki, Takamitsu Ishiyama, Takashi Suemasu, and Kaoru Toko. 2024. "Prediction and Elucidation of Physical Properties of Polycrystalline Materials Using Multichannel Machine Learning of Electron Backscattering Diffraction". *Advanced Electronic Materials* 10(7):2300875. doi: 10.1002/aelm.202300875.

Ozdemir, Rasim, and Ismail Hakki Karahan. 2023. "Effect of Sodium Citrate as Complexing Agent on the Electrodeposited CuZn Alloys: Electrochemical, Morphology, Structure, and Electrical Resistivity Studies". *Protection of Metals and Physical Chemistry of Surfaces* 59(3):445-52. doi: 10.1134/S207020512370051X.

Ozdemir, Yusuf B., Oguzhan Orkut Okudur, Mario Gonzalez, and Clement Merckling. 2024. "Predictive Modeling of Thin Film Yield Stress Using Machine Learning: A Simulation-Based Approach". Pp. 1-6 in *2024 25th International Conference on Thermal, Mechanical and Multi-Physics Simulation and Experiments in Microelectronics and Microsystems (EuroSimE)*.

Park, Hyo-Bin, Kil-Taek Lim, and Kwang-Ju Kim. 2020. "Feasibility Study of Predicting Semiconductor Thin Film Thickness Based on 1D Convolutional Neural Network". pp. 1-4 in *2020 IEEE International Conference on Consumer Electronics - Asia (ICCE-Asia)*.

Prokhorenkova, Liudmila, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. 2018. "CatBoost: unbiased boosting with categorical features". in *Advances in Neural Information Processing Systems*. C. 31. Curran Associates, Inc.

scikit-learn. 2024. "Ensembles: Gradient Boosting, Random Forests, Bagging, Voting, Stacking". *Scikit-Learn*. Accessed December 20, 2024 (https://scikit-learn/stable/modules/ensemble.html).

Tseng, Xue-Li, Yu-Shin Chen, Hsuan-Fan Chen, Hsiao-Han Lo, Peter J Wang, Yu-Min Dai, Y. Fuh, and Tomi T. Li. 2024. "A Machine Learning Study to Obtain an Optimal Processing Pulsed Frequency on Reactive Pulsed DC Sputtering of Aluminum Nitride Films". *2024 Conference of Science and Technology for Integrated Circuits (CSTIC)* 1-5. doi: 10.1109/CSTIC61820.2024.10532082.

Tuychiev, Bex. 2023. "A Guide to The Gradient Boosting Algorithm". *Datacamp*. Accessed December 18, 2024 (https://www.datacamp.com/tutorial/guide-to-the-gradient-boosting-algorithm).

Ustimenko, Aleksei, and Liudmila Prokhorenkova. 2020. "SGLB: Stochastic Gradient Langevin Boosting".

Wang, Alexander, Samuel Chen, Matthew Chang, Evan Xie, Anthony Zhu, Adam Hansen, John Jerome, and Miriam Rafailovich. 2023. "Utilizing Machine Learning to Model Interdependency of Bulk Molecular Weight, Solution Concentration, and Thickness of Spin Coated Polystyrene Thin Films".

Wikipedia. 2024. "Van Der Pauw Method". *Wikipedia*.

XGBoost. 2024. "XGBoost Documentation - xgboost 2.1.1 documentation". *XGBoost*. Accessed December 20, 2024 (https://xgboost.readthedocs.io/en/stable/).

Yoon, Sung-Ho, Jun-Hyeok Jeon, Seung-Beom Cho, Edric John Cruz Nacpil, Il Jeon, Jae-Boong Choi, and Hyeongkeun Kim. 2023. "Extreme Gradient Boosting to Predict Atomic Layer Deposition for Platinum Nano-Film Coating". *Langmuir* 39(14):4984-92. doi: 10.1021/acs.langmuir.2c03465.

Zhou, Yan, and John J. McArdle. 2015. "Rationale and Applications of Survival Tree and Survival Ensemble Methods". *Psychometrika* 80(3):811-33. doi: 10.1007/s11336-014-9413-1.

CHAPTER 3

AN ENHANCED CRAYFISH OPTIMIZATION ALGORITHM LEVERAGING CHAOTIC MAPS AND LATIN HYPERCUBE SAMPLING

Hatem Dumlu[1]

Gürcan Yavuz[2]

Hasan Temurtaş[3]

1• ORCID: https://orcid.org/0000-0002-9056-4437

2• ORCID: https://orcid.org/0000-0002-2540-1930

3• ORCID: https://orcid.org/0000-0001-6738-3024

[3]*Department of Computer Engineering, Kutahya Dumlupinar University, 43100, Kutahya, Turkey.*

Corresponding Author: Hatem Dumlu, E-mail: hatem.dumlu@dpu.edu.tr; Phone: +90 546 208 9111

INTRODUCTION

Optimization algorithms play a critical role in today's rapidly advancing technology. Their application areas are remarkably extensive, enabling them to provide solutions to a wide range of problems encountered in various aspects of daily life. Optimization algorithms serve as a cornerstone in fields such as engineering (Dhiman, 2021; Houssein et al., 2022), medicine (Vommi & Battula, 2023), economics (Vasant, 2012; Yang et al., 2024), and many others, acting as a bridge to address numerous challenges. They have been, and continue to be, instrumental in providing effective solutions in these domains.

Optimization algorithms display considerable diversity, with population-based optimization algorithms representing a prominent category. Notable examples of these include Particle Swarm Optimization (PSO) (Kennedy & Eberhart, 1995), Artificial Bee Colony (ABC) (Yavuz et al., 2016), Differential Evolution (DE) (Dumlu & Yavuz, 2023), Gray Wolf Optimization (GWO) (Mirjalili et al., 2014) among others.

In recent years, the Crayfish Optimization Algorithm has been added to the family of population-based algorithms. Introduced by Jia et al., this algorithm is inspired by the behavior of crayfish in freshwater environments (Jia et al., 2023) .Emerging optimization algorithms have increasingly captured the attention of researchers, primarily due to the challenges faced by existing methods. Common issues include slow convergence, susceptibility to getting trapped in local optima, and the delicate balance between exploration and exploitation. To address these challenges, researchers are continually working on enhancements and developing new algorithms.

A common characteristic of population-based algorithms is the initialization step. In this step, the population size is defined, and, more importantly, the individuals within the population are generated. Typically, these individuals are determined randomly. However, if the randomly generated individuals are

concentrated in a specific region, the algorithm may face challenges in exploration. Conversely, if the population is too widely distributed across the search space, the algorithm may struggle with exploitation. This imbalance between exploration and exploitation highlights the critical importance of the initialization step. If the algorithm's exploration capability is overly emphasized, it may encounter difficulties in converging to the global optimum. On the other hand, if exploitation is overly dominant, the algorithm risks becoming trapped in local optima during the search for the global optimum.

To address this issue, researchers have introduced various improvements to the initialization step. For instance, Wang et al. proposed an adaptive Chimpanzee Optimization Algorithm (ChOA) that incorporates chaotic functions (Y. Wang et al., 2023). In their approach, the Tent Chaotic map was utilized during the initialization phase to generate better initial solutions, thereby enhancing the algorithm's convergence performance.

Tong et al. developed the Coyote Optimization Algorithm (COA) by integrating chaotic maps (Tong et al., 2022) .To address issues such as the algorithm's tendency to get stuck in local optima and slow convergence, they incorporated chaotic approximations into the equations governing the social behavior within the algorithm. This modification aimed to enhance the algorithm's overall performance and its ability to explore the search space more effectively.

Wang et al. proposed a chaotic version of the Cuckoo Search Optimization Algorithm (Cuckoo Search-CS) by incorporating chaotic maps (G.-G. Wang et al., 2016) . To enhance the algorithm's performance, they employed twelve different chaotic maps to dynamically adjust the step size of the cuckoos during the search process. This approach aimed to improve the algorithm's exploration and exploitation capabilities, leading to better overall optimization results.

MATERIAL AND METHODS

Original Crayfish Optimization Algorithm

Jia et al. (2023) introduced the Crayfish Optimization Algorithm (COA) (Jia et al., 2023), which draws inspiration from the natural behaviors of crayfish. Crayfish, resembling shrimp, are found in diverse freshwater habitats such as rivers, lakes, streams, and cave pools. Although there are over 600 species of crayfish, they all exhibit a common trait: the ability to dig burrows. These shelters serve various functions, including protecting the crayfish from predators, supporting foraging activities, providing spaces for incubation, and preserving moisture. This burrowing behavior forms the foundation of the algorithm's approach to search and optimization.

Crayfish are highly sensitive to temperature, capable of surviving in environments ranging from 11 to 44 degrees Celsius. However, the ideal temperature range for their growth lies between 15 and 30 degrees, with 25 degrees Celsius being the optimal temperature for survival. Jia et al. introduced the Crayfish Optimization Algorithm, drawing inspiration from the crayfish's seasonal behavior, foraging habits, and competitive behaviors. These three behavioral patterns are used in the algorithm to simulate the processes of exploration and exploitation, organized into three distinct stages.

The population in COA is initialized according to Eqtn.1(Equation 1) presented below.

$$CF = [CF_1, CF_2, \cdots, CF_{PS}] = \begin{bmatrix} CF_{1,1} & \cdots & CF_{1,j} & \cdots & CF_{1,dim} \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ CF_{i,1} & \cdots & CF_{i,j} & \cdots & CF_{i,dim} \\ \vdots & \cdots & \vdots & \cdots & \vdots \\ CF_{N,1} & \cdots & CF_{N,j} & \cdots & CF_{N,dim} \end{bmatrix} \quad (1)$$

As shown in the equation above, each crayfish is represented as a 1×dim vector, where each column in the matrix corresponds to a potential solution to the problem. The value of each CF_i is constrained within the specified upper and lower bounds. During the initialization phase of the COA, a random

population of candidate solutions is generated. Here, CF represents the initial positions of the population, PS denotes the population size, and dmn refers to the dimensionality of the population. $CF_{i,j}$ indicates the position of individual i in dimension j. The value of $CF_{i,j}$ is calculated using Eqtn.2.

$$CF_{i,j} = lb_j + (ub_j - lb_j) \times random \quad (2)$$

In the COA algorithm, lb_j corresponds to the lower bound of dimension lb_j, ub_j denotes the upper bound of dimension j, and $random$ represents a random value. Since crayfish are temperature-sensitive, changes in temperature within the COA algorithm lead to the crayfish transitioning through different behavioral stages. The temperature is assigned using Eqtn.3.

$$temperature = random \times 15 + 20 \quad (3)$$

In Eqtn.3, the $temperature$ value represents the ambient temperature in the crayfish's environment. When the temperature exceeds 30 degrees Celsius, the crayfish seeks a cooler location for its summer retreat. Once the temperature stabilizes within an optimal range, the crayfish engages in foraging behavior. Crayfish typically exhibit foraging behavior between 20 and 30 degrees Celsius, with the optimal temperature for foraging being 25 degrees. The mathematical model describing the crayfish's foraging and food intake is provided in Eqtn.4.

$$p = C_1 \times \left(\frac{1}{\sqrt{2\times\pi}\times\sigma)} \times \exp\left(-\frac{(temperature-\Omega)^2}{2\sigma^2}\right)\right) \quad (4)$$

Ω represents the optimal temperature for the crayfish. σ and C_1 are used to control the food intake of the crayfish at different temperatures.

In this context, Ω represents the optimal temperature for the crayfish. The parameters σ and C_1 are employed to regulate the crayfish's food intake at varying temperatures.

Summer vacation phase (Exploration)

Crayfish display various behaviors depending on temperature conditions. When the temperature rises above 30 degrees Celsius (temperature > 30), the crayfish transitions into the summer vacation phase, where it seeks a shaded area for shelter. The shaded area, CF_{Shade} is defined as shown in Eqtn.5.

$$CF_{Shade} = (CF_G + CF_L)/2 \tag{5}$$

CF_G is the best position obtained so far according to the iteration. CF_L is the best position of the current population. According to the COA algorithm, the competition of crayfish for shades is a random event. If $random < 0.5$, there are no crayfish to compete for the shades and the crayfish enters the shade directly. The mathematical model of this situation is given in Eqtn.6.

$$CF_{i,j}^{t+1} = CF_{i,j}^{t} + C_2 \times random \times (CF_{Shade} - CF_{i,j}^{t}) \tag{6}$$

In Eqtn.6, t denotes the current iteration number, while $t+1$ represents the iteration number of the next generation. C_2 is a decreasing function, and its formula is shown in Eqtn.7.

$$C_2 = 2 - t/T \tag{7}$$

Here, T represents the maximum iterations number. During the summer vacation phase, the crayfish's objective is to converge to the shaded area, which symbolizes the optimal solution. As the crayfish approaches the shade, it indicates that the COA algorithm is moving closer to the optimal solution, thereby enhancing the exploitation capability of the algorithm.

Competition phase (Exploitation)

The crayfish enters the competition phase when the temperature exceeds thirty degrees $temperature > 30$ and $rand \geq 0.5$. This scenario indicates that other crayfish are also vying for the same shade. Consequently, as modeled in Eqtn.8, the crayfish engage in a competition for access to the shade.

$$CF_{i,j}^{t+1} = CF_{i,j}^{t} - CF_{z,j}^{t} + CF_{Shade} \tag{8}$$

Here, z represents a randomly selected crayfish individual, as shown in Eqtn.9.

$$z = round(random \times (N - 1)) + 1 \tag{9}$$

During the competition phase, the crayfish engage in a contest with one another. Crayfish CF_i adjusts its position based on the location of another crayfish, CF_z. This adjustment helps expand the search range of the COA, thereby enhancing its exploration ability.

Foraging Phase (Exploitation)

If the temperature is below or equal to thirty degrees Celsius ($temp \leq 30$), the conditions are favorable for feeding. Under such circumstances, the crayfish will approach the food. Upon locating the food, it will evaluate its size. If the food is too big, the crayfish will break it into smaller pieces using its claws before consuming it. The food, denoted as CF_{food}, is described by

Eqtn.10, while the amount of the food, denoted as Q, is represented by Eqtn.11.

$$CF_{food} = CF_G \tag{10}$$

$$Q = C_3 \times random \times (fitness_i / fitness_{food}) \tag{11}$$

In Eqtn.11, C_3 represents the food factor, which indicates the largest possible food size and has a constant value of 3. $fitness_i$ denotes the fitness value of the i. crayfish, while $fitness_{food}$ represents the fitness value of the food location. The crayfish determines the size of the food based on the biggest available food. When $Q > (C3 + 1)/2$, the food is considered too big. In this case, the crayfish will tear (or split) the food using its first claw foot, as described in Eqtn.12.

$$CF_{food} = \exp(-\frac{1}{Q}) \times CF_{food} \tag{12}$$

As the food size decreases, the second and third claws of the crayfish will alternately position the food into its mouth. To simulate this behavior, a combination of sine and cosine functions is employed. This is represented in Eqtn.13, along with the foraging equation that describes the crayfish's foraging behavior.

$$CF_{i,j}^{t+1} = CF_{i,j}^{t} + CF_{food} \times p \times (\cos(2 \times \pi \times random) - \sin(2 \times \pi \times random)) \tag{13}$$

When $Q \leq (C3 + 1)/2$, the crayfish moves directly towards the food and consumes it. This behavior is represented by the formula in Eqtn.14.

$$CF_{i,j}^{t+1} = (CF_{i,j}^{t} - CF_{food}) \times p + p \times random \times CF_{i,j}^{t} \quad (14)$$

To summarize this stage, crayfish employ different feeding strategies based on the size Q of the food. CF_{food} represents the optimal solution. If the food size is appropriate for the crayfish to consume, it will approach and eat the food. However, if the food is too big, it indicates a significant difference between the crayfish's current position and the optimal solution. In such cases, the food will need to be reduced in size, bringing it closer to the optimum (optimal food size). This process helps bring the COA closer to the optimal solution, enhances the exploitation capability of the algorithm, and ensures good convergence. The pseudo code for COA is provided in Algorithm 1.

Algorithm 1: Pseudocode of the Crayfish Optimization Algorithm

Set the number of iterations (**T**), population (**PS**), and population dimension (**dmn**).

Generate an initial random population

Evaluate the fitness of the population to determine the best solution $\boldsymbol{CF_G}$ and the worst solution $\boldsymbol{CF_L}$

While t < T

Define the temperature using Eqtn.3.

If temperature > 30

Calculate the shade ($\boldsymbol{CF_{shade}}$) based on Eqtn.5

If random < 0.5

The crayfish enters the summer vacation phase, as Eqtn.6

Else

The crayfish competes for the shade using Eqtn.8

End-If

Else

The food intake (**p**) and its size (**Q**) are determined by Eqtn.4 and Eqtn.11

If Q > 2

The crayfish splits the food based on Eqtn.12

The crayfish search food based on Eqtn.13

Else

The crayfish search for food based on Eqtn.14

End-If

End-If

Update the fitness values for the best solution $\boldsymbol{CF_G}$ and the worst solution $\boldsymbol{CF_L}$

t = t + 1

End-While

Chaotic Maps

Chaotic maps are mathematical tools that help in understanding and modeling the behavior of dynamic systems over time. These systems are typically characterized by their dynamic nature, and chaotic maps aim to explain the unpredictable, chaotic behavior of these systems by analyzing their

changes over time. Similar to optimization algorithms, systems based on chaotic maps are highly sensitive to initial conditions; small differences in the initial state can lead to significant variations in the system's behavior over time. Therefore, chaotic maps are valuable for studying how initial conditions influence the long-term dynamics of a system.

In the context of optimization algorithms, chaotic maps can be applied during the initialization phase to strike a balance between exploration and exploitation, thereby enhancing the algorithm's performance. The population generated using chaotic maps carries forward the benefits of this initialization, improving the overall optimization process.

Chaotic maps are diverse and come with various mathematical formulations. In this study, ten different chaotic map methods were employed to create chaotic variants of the Crayfish Optimization Algorithm. These methods include Arnold's Cat Map, Bernoulli Shift Map, Chebyshev Map, Circle Map, Cubic Map, Duffing Map, Gauss Map (Mouse Map), Henon Map, ICMIC Map, and Ikeda Map. The chaotic maps utilized to generate the chaotic variants of the Crayfish Optimization Algorithm are listed in Table 1, with their respective formulas provided below.

Table 1 Types and formulas of chaotic maps used in experiments

	id	**name**	**formula**
1	ch1	Arnold"s Cat Map	Eqtn.15
2	ch2	Bernoulli Shift Map	Eqtn.16
3	ch3	Chebyshev Map	Eqtn.17
4	ch4	Circle Map	Eqtn.18
5	ch5	Cubic Map	Eqtn.19
6	ch6	Duffing Map	Eqtn.20
7	ch7	Gauss/Mouse Map	Eqtn.21
8	ch8	Henon Map	Eqtn.22

9	ch9	ICMIC Map	Eqtn.23
10	ch10	Ikeda Map	Eqtn.24

$$\begin{pmatrix} x\prime \\ y\prime \end{pmatrix} = \begin{pmatrix} 1 & 1 \\ 1 & 2 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} \quad mod(N) \tag{15}$$

$$x_{n+1} = \begin{cases} 1.75x_n - 0.5, & x_n > 0 \\ 1.75x_n + 0.5, & x_n \leq 0 \end{cases} \tag{16}$$

$$x_{n+1} = cos(\frac{n}{cosx_n}) \tag{17}$$

$$x_{n+1} = [\, x_n + b - \frac{a}{2\pi} sin(2\pi x_n)] \;\; mod(1) \tag{18}$$

$$x_{n+1} = R_c x_n - {x_n}^3 \tag{19}$$

$$x_{n+1} = y_n \tag{20}$$

$$y_{n+1} = -bx_n + ay_n - {y_n}^3$$

$$x_{n+1} = \begin{cases} 0, & x_n = 0 \\ \frac{1}{x_n} - \left\lfloor \frac{1}{x_n} \right\rfloor, & otherwise \end{cases} \tag{21}$$

$$x_{n+1} = 1 - A{x_n}^2 - y_n \tag{22}$$
$$y_{n+1} = bx_n$$

$$x_{n+1} = sin(\frac{a}{x_n}) \tag{23}$$

$$x_{n+1} = 1 + k(x_n cos\theta - y_n sin\theta) \tag{24}$$
$$y_{n+1} = k(x_n sin\theta + y_n cos\theta)$$

$$\theta = 0.4 - \frac{6}{1 + {x_n}^2 + {y_n}^2}$$

Latin Hyper Cube Sampling

Latin Hyper Cube Sampling (LHS) is a sampling method used in fields such as engineering, statistics and computer science (Hakimi, 2024). It is especially used in simulation and optimization problems. This method provides effective sampling of parameters in multidimensional spaces. This is the biggest distinction that separates this method from traditional methods. Because it covers all ranges of each variable. Since the initialization step is essential for the balance of exploration and exploitation in optimization algorithms, this method is the subject of this paper. In the initialization step of optimization algorithms, this method can be used to initialize the population (Z. Wang et al., 2024). The Latin Hyper Cube Sampling method is described in detail below.

Consider an LHS with two variables (k=2) and five samples (n=5). Five equal intervals are created for each variable and random values are selected from these intervals.

Latin Hyper Cube Sampling (LHS) is a sampling technique widely used in various fields, including engineering, statistics, and computer science (Hakimi, 2024). It is particularly beneficial for simulation and optimization problems. This method allows for efficient sampling of parameters in multidimensional spaces, which is its primary distinction from traditional

sampling methods. Unlike conventional methods, LHS ensures that all ranges of each variable are covered. Given that the initialization step is crucial for balancing exploration and exploitation in optimization algorithms, LHS is a key focus of this paper. In optimization algorithms, LHS can be utilized during the initialization phase to generate the population (Z. Wang et al., 2024). The detailed process of the Latin Hyper Cube Sampling method is outlined below.

Consider an LHS with two variables (k=2) and five samples (n=5). In this case, five equal intervals are created for each variable, and random values are selected from these intervals.

$$L_{1,j} = \frac{j-1}{5} + \frac{u_{1,j}}{5} \qquad (j = 1, \dots, 5) \tag{25}$$

$$L_{2,j} = \frac{j-1}{5} + \frac{u_{2,j}}{5} \qquad (j = 1, \dots, 5) \tag{26}$$

$u_{1,j}$ and $u_{2,j}$ are randomly selected values and the sampling values are as follows.

$L_{1,1}, L_{1,2}, L_{1,3}, L_{1,4}, L_{1,5}, L_{2,1}, L_{2,2}, L_{2,3}, L_{2,4}, L_{2,5}$

The values are organized by applying different permutations to each variable, ensuring that each interval is represented in the sample. The final arrangement of the variables is then as follows.

$R_{1,1}, R_{1,2}, R_{1,3}, R_{1,4}, R_{1,5}, R_{2,1}, R_{2,2}, R_{2,3}, R_{2,4}, R_{2,5}$

The resulting sampling points are expressed as $(R_{1,j}, R_{2,j})$ $j = 1, \dots, 5$

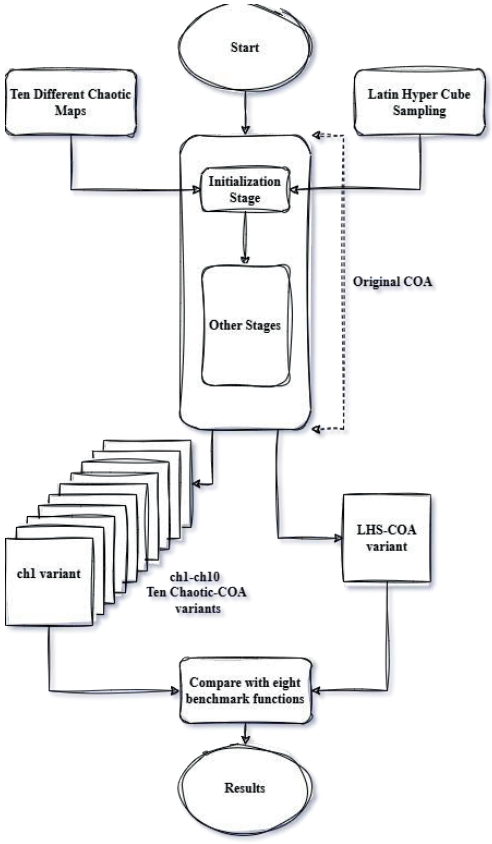
Chaotic-COA and Latin Hyper Cube Sampling-COA

The structure of chaotic maps can be leveraged to balance exploration and exploitation within the Crayfish Optimization Algorithm (COA). To achieve this, the population of COA was initialized using chaotic maps during the initialization phase. Ten different chaotic map methods were applied, resulting in ten distinct variants of COA. The performance of these variants

was then evaluated using eight different benchmark functions, with the results compared to those obtained from the original COA.

In a similar manner, Latin Hyper Cube Sampling (LHS) was employed to enhance the performance of the COA. The COA was initialized using LHS, resulting in the LHS-COA variant. This variant was then evaluated using eight different benchmark functions, with the results compared to those obtained from the original COA. A flowchart of the proposed COA variants is given in Figure 1. Also the pseudo-code for both the Chaotic COA and the LHS-COA is provided in Algorithm 2 and Algorithm 3, respectively. The benchmark functions used to compare all variants, including the original algorithm, are listed in Table 2.

Figure 1 : Flowchart of the producing COA variants



Algorithm 2: Pseudocode of the Chaotic-Crayfish Optimization Algorithm

Set the number of iterations (**T**), population (**PS**), and population dimension (**dmn**)
Generate an initial random population with chaotic maps
Evaluate the fitness of the population to determine the best solution $\boldsymbol{CF_G}$ and the worst solution $\boldsymbol{CF_L}$
While t < T
Define the temperature using Eqtn.3.
If temperature > 30
  Calculate the shade ($\boldsymbol{CF_{shade}}$) based on Eqtn.5
  If random < 0.5
    The crayfish enters the summer vacation phase, as Eqtn.6
  Else
    The crayfish competes for the shade using Eqtn.8
  End-If
Else
  The food intake (**p**) and its size (**Q**) are determined by Eqtn.4 and Eqtn.11
  If Q > 2
    The crayfish splits the food based on Eqtn.12
    The crayfish search food based on Eqtn.13
  Else
    The crayfish search for food based on Eqtn.14
  End-If
End-If
Update the fitness values for the best solution $\boldsymbol{CF_G}$ and the worst solution $\boldsymbol{CF_L}$
t = t + 1
End-While

Algorithm 3: Pseudocode of the LHS-Crayfish Optimization Algorithm

Set the number of iterations (**T**), population (**PS**), and population dimension (**dmn**)
Generate an initial random population with latin hyper cube sampling
Evaluate the fitness of the population to determine the best solution $\boldsymbol{CF_G}$ and the worst solution $\boldsymbol{CF_L}$
While t < T
Define the temperature using Eqtn.3.
If temperature > 30

Calculate the shade (CF_{shade}) based on Eqtn.5
If random < 0.5
The crayfish enters the summer vacation phase, as Eqtn.6
Else
The crayfish competes for the shade using Eqtn.8
End-If
Else
The food intake (**p**) and its size (**Q**) are determined by Eqtn.4 and Eqtn.11
If Q > 2
The crayfish splits the food based on Eqtn.12
The crayfish search food based on Eqtn.13
Else
The crayfish search for food based on Eqtn.14
End-If
End-If
Update the fitness values for the best solution CF_G and the worst solution CF_L
t = t + 1
End-While

Table 2 Benchmark functions

Id	Name
fnc1	QuarticWN Function
fnc2	Rosenbrock Function
fnc3	Schwefel 2.26 Function
fnc4	Penalized 1 Function
fnc5	Penalized 2 Function
fnc6	Levy Function
fnc7	Kowalik Function
fnc8	Perm Function d,β

EXPERIMENTS

In this section, the experimental results are presented. The Chaotic COA and LHS-COA represent enhanced versions of the Crayfish Optimization Algorithm. These improvements were achieved by introducing modifications to the initialization phase of the original COA. Each variant underwent ten independent runs, and the most successful outcomes were selected as the final results. The parameter settings employed in the experiments are detailed in Table 3, while the system specifications used during the experimental process are outlined in Table 4.

Table 3 Parameters in experiments

Parameter	**Value**
Population Size	30
Population Dimension	50
Iteration Number	1000
Independent Run Number	10

Table 4 System information of experiments

Property	**Value**
Operating System	Windows 10
Processor	Intel i5 13600K
Environment	Matlab 2022A

In the initial phase, the standard COA was augmented by incorporating a chaotic map during the initialization stage. This approach yielded ten distinct COA variants, each corresponding to a different chaotic

map. These variants, along with the original algorithm, were evaluated using eight benchmark functions commonly employed in optimization problem assessments, with a preference for more challenging functions. The outcomes were compared based on their optimal values. Relative to the original COA, the variants COA-ch1, COA-ch3, COA-ch4, COA-ch5, COA-ch6, COA-ch7, COA-ch8, and COA-ch10 outperformed in seven out of the eight functions and underperformed in only one. The COA-ch2 variant outperformed the original COA in six of the eight benchmark functions, underperformed in one, and tied in one functions. Similarly, the COA-ch9 variant achieved six wins and two losses. Detailed results are presented in Table 5, with the optimal values for each function highlighted in bold. Among the ten Chaotic COA variants, COA-ch5 stands out for its superior performance.

Table 5 Comparison of the results of COA and Chaotic-COA variants

FNC.	COA	COA-ch1	COA-ch2	COA-ch3	COA-ch4	COA-ch5	COA-ch6	COA-ch7	COA-ch8	COA-ch9	COA-ch10
Fnc1	8.622E-06	2.831E-07	**2.393E-07**	4.602E-07	5.674E-07	3.284E-07	8.942E-07	8.871E-07	2.812E-06	1.542E-06	1.330E-06
Fnc2	2.411E+01	2.382E+01	2.411E+01	2.362E+01	2.371E+01	2.390E+01	2.350E+01	2.341E+01	**2.321E+01**	2.377E+01	2.380E+01
Fnc3	2.177E+03	1.423E+03	1.791E+03	1.521E+03	1.882E+03	**1.292E+03**	1.422E+03	2.143E+03	1.860E+03	1.762E+03	1.382E+03
Fnc4	6.599E-10	3.302E-11	3.820E-11	4.942E-11	3.183E-11	**1.462E-12**	7.670E-11	5.585E-12	6.911E-11	6.721E-11	4.134E-11
Fnc5	7.301E-01	3.938E-01	5.333E-01	6.877E-01	4.541E-01	4.401E-01	**2.742E-01**	6.330E-01	5.391E-01	9.101E-01	6.495E-01
Fnc6	**7.302E-01**	5.940E+00	6.890E+00	1.211E+00	5.530E+00	3.003E+00	3.003E+00	6.041E+00	2.262E+00	7.960E+00	6.050E+00
Fnc7	1.892E-01	**2.301E-02**	**2.301E-02**	**2.301E-02**	**2.301E-02**	**2.301E-02**	**2.301E-02**	**2.301E-02**	**2.301E-02**	**2.301E-02**	**2.301E-02**
Fnc8	4.115E+80	3.282E+79	6.871E+79	2.831E+80	1.782E+79	1.033E+80	2.471E+80	1.871E+80	6.812E+78	**2.841E+78**	1.066E+80
Win		7.00	6.00	7.00	7.00	7.00	7.00	7.00	7.00	6.00	7.00
Lose		1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00	2.00	1.00
Draw		0	1	0	0	0	0	0	0	0	0

In the second stage, the original COA algorithm was created and run using Latin Hyper Cube Sampling while creating the population in the initialization step. The resulting variant is called LHS-COA. LHS-COA was compared with the original COA and COA-ch5(best-chaotic-variant). The results were compared with each other at the "best" values. Compared to the original COA, LHS-COA was victorious in seven out of eight functions and defeated in one. Compared to COA-ch5, the most successful Chaotic COA, LHS-COA won in five out of eight functions, lost in one and tied in two. In this case, the best COA variant was LHS-COA. The results are given in Table 6. The best values are in bold.

Table 6 COA, LHS-COA and COA-ch5 benchmark results

FNC	COA	LHS-COA	COA-ch5
Fnc1	8.622E-06	**8.132E-08**	3.284E-07
Fnc2	2.411E+01	**2.333E+01**	2.390E+01
Fnc3	2.177E+03	**1.292E+03**	**1.292E+03**
Fnc4	6.599E-10	1.751E-11	**1.462E-12**
Fnc5	7.301E-01	**1.940E-01**	4.401E-01
Fnc6	**7.302E-01**	2.980E+00	3.003E+00
Fnc7	1.892E-01	**2.301E-02**	**2.301E-02**
Fnc8	4.115E+80	**3.893E+79**	1.033E+80

RESULTS AND DISCUSSION

In this study, improved Crayfish Optimization Algorithms are proposed. These algorithms are made by modifying the initialization step of the original algorithm. In the first step, ten different variants were obtained by initializing

the population of COA using chaotic maps and the variants were tested with eight different benchmark functions to compare with the original algorithm. When evaluating the results, the "best" results were taken as a criterion. The most successful variant was COA-ch5.

In the second stage, the LHS-COA variant was obtained by initializing the population of COA using Latin Hyper Cube Sampling. In order to compare this variant with COA-ch5 and the original COA, the variants were tested with eight different benchmark functions. Likewise, the results were evaluated by taking the “best” results as a criterion.

Considering all the results, the eight chaotic variants victorious in seven of the eight functions compared to the original COA. Two other chaotic variants victorious in six out of eight functions compared to the original COA. Another variant, LHS-COA, outperformed the original COA in seven out of eight functions. Comparing the best variants, LHS-COA and COA-ch5, it is noticeable that LHS-COA wins in five out of eight functions. According to all results LHS-COA appears the best variant.

In future studies, these improved variants can be used for feature selection. Also, different strategies can be added to COA and compared with these variants.

REFERENCES

Dhiman, G. (2021). ESA: a hybrid bio-inspired metaheuristic optimization approach for engineering problems. *Engineering with Computers*, *37*, 323–353.

Dumlu, H., & Yavuz, G. (2023). CEC 2019 Problemleri İçin Bir Diferansiyel Gelişim Algoritması. *Mühendislik Bilimleri ve Araştırmaları Dergisi*, *5*(2), 304–311. https://doi.org/10.46387/bjesr.1311593

Hakimi, F. (2024). *Robust estimation with latin hypercube sampling: a central limit theorem for Z-estimators*.

Houssein, E. H., Rezk, H., Fathy, A., Mahdy, M. A., & Nassef, A. M. (2022). A modified adaptive guided differential evolution algorithm applied to engineering applications. *Engineering Applications of Artificial Intelligence*, *113*, 104920.

Jia, H., Rao, H., Wen, C., & Mirjalili, S. (2023). Crayfish optimization algorithm. *Artificial Intelligence Review*, *56*(2), 1919–1979. https://doi.org/10.1007/s10462-023-10567-4

Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *Proceedings of ICNN'95 - International Conference on Neural Networks*, *4*, 1942–1948 vol.4. https://doi.org/10.1109/ICNN.1995.488968

Mirjalili, S., Mirjalili, S. M., & Lewis, A. (2014). Grey Wolf Optimizer. *Advances in Engineering Software*, *69*, 46–61. https://doi.org/https://doi.org/10.1016/j.advengsoft.2013.12.007

Tong, H., Zhu, Y., Pierezan, J., Xu, Y., & Coelho, L. dos S. (2022). Chaotic Coyote Optimization Algorithm. *Journal of Ambient Intelligence and Humanized Computing*, *13*(5), 2807–2827. https://doi.org/10.1007/s12652-021-03234-5

Vasant, P. M. (2012). *Meta-heuristics optimization algorithms in engineering, business, economics, and finance*. IGI Global.

Vommi, A. M., & Battula, T. K. (2023). A binary Bi-phase mutation-based hybrid Equilibrium Optimizer for feature selection in medical datasets classification.

Computers and Electrical Engineering, *105*, 108553. https://doi.org/https://doi.org/10.1016/j.compeleceng.2022.108553

Wang, G.-G., Deb, S., Gandomi, A. H., Zhang, Z., & Alavi, A. H. (2016). Chaotic cuckoo search. *Soft Computing*, *20*(9), 3349–3362. https://doi.org/10.1007/s00500-015-1726-1

Wang, Y., Liu, H., Ding, G., & Tu, L. (2023). Adaptive chimp optimization algorithm with chaotic map for global numerical optimization problems. *The Journal of Supercomputing*, *79*(6), 6507–6537. https://doi.org/10.1007/s11227-022-04886-6

Wang, Z., Zhao, D., Heidari, A. A., Chen, Y., Chen, H., & Liang, G. (2024). Improved Latin hypercube sampling initialization-based whale optimization algorithm for COVID-19 X-ray multi-threshold image segmentation. *Scientific Reports*, *14*(1), 13239. https://doi.org/10.1038/s41598-024-63739-9

Yang, P. B., Chan, Y. J., Yazdi, S. K., & Lim, J. W. (2024). Optimisation and economic analysis of industrial-scale anaerobic co-digestion (ACoD) of palm oil mill effluent (POME) and decanter cake (DC) using machine learning models: A comparative study of Gradient Boosting Machines (GBM), K-nearest neighbours (KNN), and random forest (RF). *Journal of Water Process Engineering*, *58*, 104752. https://doi.org/https://doi.org/10.1016/j.jwpe.2023.104752

Yavuz, G., Aydin, D., & Stützle, T. (2016). Self-adaptive search equation-based artificial bee colony algorithm on the CEC 2014 benchmark functions. *2016 IEEE Congress on Evolutionary Computation (CEC)*, 1173–1180. https://doi.org/10.1109/CEC.2016.7743920

CHAPTER 4

STELLAR CLASSIFICATION WITH MACHINE LEARNING ALGORITHMS GRAPH-BASED FEATURE SELECTION APROACH

Mehmet Bilge Han TAŞ[1]
Eyyüp YILDIZ[2]

1 Res. Asst., Erzincan Binali Yildirim University, Faculty of Engineering and Architecture, Department of Computer Engineering, Erzincan, Turkey, ORCID: 0000-0001-6135-1849 bilgehantas@erzincan.edu.tr
2 Asst. Prof. Dr., Erzincan Binali Yildirim University, Faculty of Engineering and Architecture, Department of Computer Engineering, Erzincan, Turkey, ORCID: 0000-0002-7051-3368 eyyup.yildiz@erzincan.edu.tr

1.INTRODUCTİON

People have been trying to make sense of the environment since the past. He feels the need to research by wondering what he sees or cannot see. For this reason, he wants to learn what is going on around the world, starting from our own environment. With the developing technology, the distance and clarity that telescopes can see have increased. We can observe galaxies, stars and planets including the Earth. There is a lot of work and an accumulated data warehouse on this subject. Many data sets are published online. In the study, SDSS17, a data set that is continuously expanded by Sloan Digital Sky Surveys (SDSS), was used (Accetta et al., 2022). The dataset includes a wide variety of features and classifications of stars, galaxies and quasars. Quasars are radio sources very similar to stars. Since they are radio sources, they can be detected by radio waves. Because they are so far away from Earth, the radio waves they create take a long time to arrive. It is difficult to detect because it is very similar to stars due to its structure, and also because it is very far from the earth (Viquar et al., 2019).

With the increase in the processing capacity of computers, classification processes can be done in many areas. It has a very serious potential for distinguishing and classifying celestial bodies, which can also be used in the field of astronomy. It has come to play a very important role in the discovery of the universe (Hinners, Tat, & Thorp, 2018). Many different objects that cannot be distinguished with the naked eye can be classified using machine learning methods and even the movements of celestial bodies can be predicted. Although the use of machine learning techniques in astronomy is not very common, its use will reveal great efficiency. One of the first studies on this subject, Bailey et al. In his studies, he worked on object classification in synthetic supernovas (Bailey et al., 2007). There are also studies on the methods of using machine learning algorithms and data mining in astronomy (Ball & Brunner, 2010). There are studies on transit shapes (Armstrong, Pollacco, & Santerne, 2016) and on transit shape signals with transit metrics using Kepler data (Thompson et al., 2015). In recent years, more studies have been done on the increasing classification and prediction. A machine learning approach has been made on the prediction of Cepheid stars. If there is a labeled classification as a result of the experiments, it is stated that a good estimate is made (Vilalta, Gupta, Macri, & Computing, 2013). It is mentioned that an effective algorithm has been developed with Random forest with high classification accuracy for stars, galaxies and QSOs (Bai, Liu, Wang, & Yang, 2018).

Although feature selection studies are not very common in the field of astronomy, the important thing is how this process is done. Studies on this subject are carried out with many different techniques. Although the studies vary, it is basically trying to get better results by choosing the best features that are wanted to be done and correlated with the classes. As examples of studies,

feature selection with supervised aproach and unsupervised aproach (Roffo et al., 2020), hyper graph-theoretic approaches in very high-dimensional data (Zhang & Hancock, 2011), feature selection using PageRank in multi-label data (Hashemi, Dowlatshahi, & Nezamabadi-Pour, 2020), feature selection by making boundary region on audio data (Yasmin, Das, Nayak, Pelusi, & Ding, 2020). Many studies such as making selection have been done.

The aim of the study is to select the most suitable features by using graph structures and accordingly to make the best classification process using various machine learning methods. In the study;

• The received data set is first analyzed using various visual materials. Then, a result is obtained in the light of machine learning methods without selecting a feature.

• Afterwards, a graph-based feature is extracted to be able to select the feature. For this, the weighted graph of the 17 features in total is calculated one by one and the features are combined. The features are added to the classes in accordance with their weights, with classes (Galaxy, QSO, Star) in the center. The correlation matrix is based on when selecting the weights.

• The feature with the most weight approaches the center, that is, the classes. This makes it easy for us to choose a feature. Features that move away from the center are removed from the dataset. Because the farther it is from the center, the less effect it has on getting results. Then the machine learning methods are applied again and the necessary optimization processes are performed and new results are obtained and compared.

In this study, graph-based feature selection was made using SDSS17 and classification was done with machine learning methods. This paper is organized as follows. In Section 2, the machine learning algorithms used and the methods used are given. The experimental results of the proposed study are given in Section 3. In Section 4, the experimental results are discussed and interpreted. In Section 5, the summary of the study and the general results are explained.

2. MATERIAL AND METHOD

In this section, the general framework of the transactions to be made is presented. Later in the section, dataset, feature selection and artificial intelligence methods used are given. The flow chart of the study is given in Figure 1.

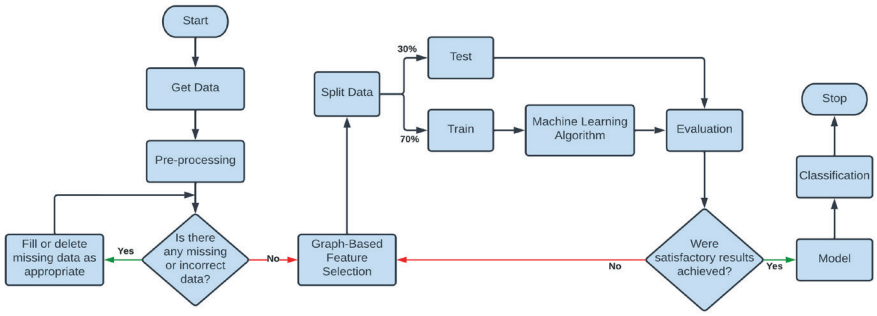


Figure 1. *Flow chart of the study.*

First, the data set is obtained. Then the obtained data is subjected to a preprocessing. If there are any deficiencies in the data, these missing data are filled in with generally accepted methods. This process was not performed because there was no missing data in the data used in this study. Then, graph-based feature selection is made. The dataset is splitted. Experimental results are obtained by applying machine learning techniques. If the results are satisfactory, the model is created. If the results are far below the desired, feature selection is made again, the parameters used in machine learning are optimized and the final model is created. This model, which has been created, performs the classification process and gives the results of the relevant class of the features.

2.1 Dataset

Sloan Digital Sky Surveys (SDSS) have been observing the sky for many years as part of their projects. They have published a huge amount of data, together with their latest work and data collected as a result of this observation The version of the dataset used in the study was taken from the kaggle website (Kaggle). The data consists of 100000 space observations, which means as many rows of data. Each observation consists of 17 features and a class column. The numerical distributions of the classes are given in Figure 2.

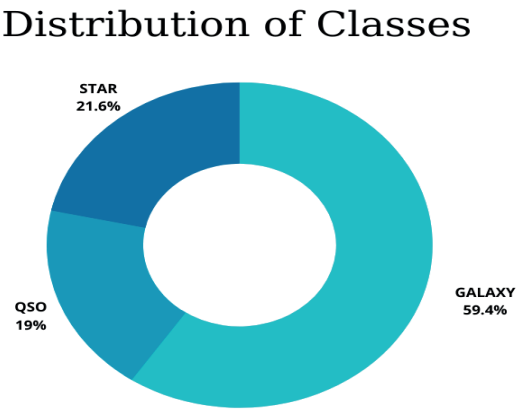


Figure 2. *Distribution of classes*

2.2 Preprocessing and Feature Selection

As data pre-processing, any filling process was not performed as there was no missing data. However, the rerun_ID column was removed, which corrupted the distribution and properties of the data. It is common practice to standardize data when values are in very different ranges (Zhu et al., 2021). StandardScaler was used for the normalization process (Formula 1).

$$Z = \frac{X-\mu}{\sigma} \qquad (1)$$

μ is the sample of the training sample, σ is the standard deviation of the training sample, X is the data to be processed.

Then, based on the values in the correlation matrix, features are added with classes in the center. As the splices increase, nodes with proportionally higher weights are expected to get closer to the center. The node closer to the

center becomes class-determining as a correlation.

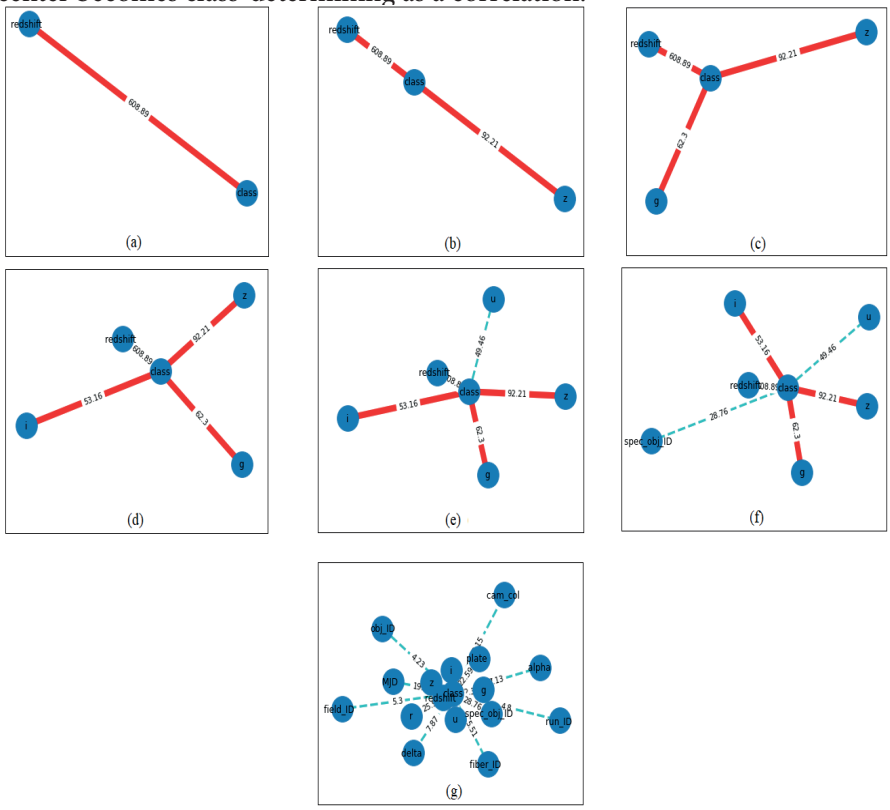


Figure 3. *Graph-based feature selection*

As shown in Figure 3, the first feature is added with classes in the center (a). As the features are added, it is seen that they approach or move away from the center depending on their weight (a-f). Since all data has been added, the ones closest to the center are the features (g) that should be selected. Although "alpha" and "delta" seem to have a determining structure when an estimated selection is made, it is seen that they are out of the field in the light of this approach. When considered as Graph-based, columns and features that should be used very clearly are separated.

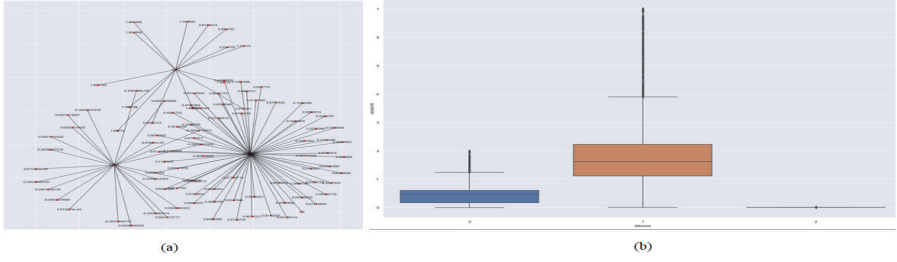


Figure 4. *Feature with highest affinity*

According to the graph-based analysis, it was seen that the feature that is closest to the center, that is, the feature that will have the highest effect on the classification, is "redshift". In Figure 4, the grapf distribution of Redshift (a) and the value ranges of the classes (b) are shown. The sample space was created at the rate of 1/1000. Because it is a very large data set, very long processing times are required. The aim here is to have an idea about distributions. Therefore, keeping the sample space small in visualization will not make a difference.

2.3 Logistic Regression (LR)

Logistic regression is used to find the odds ratio of more than one variable. Its main purpose of use is to analyze all variables and give a simple output (Sperandei, 2014). It is widely used because it gives good results in small data as well as large data. Because it has a simple calculation logic and at the same time it is preferred with its fast working structure. It has significant value in comparison with other machine learning methods. It can also be used for classification by revealing an estimation result such as a regression curve for class inferences to be made. It has been used in a wide variety of fields as well as in this study.

2.4 Decision Tree (DT)

Decision tree is a machine learning algorithm that belongs to supervised learning algorithms. It is used to solve classification problems (Safavian, Landgrebe, & cybernetics, 1991). In this study, a decision tree classifier is used to estimate the dependent variable based on some derived decision rules from previous data (training and testing phases). It is represented as nodes and nodes where root nodes are used to classify the properties of the instances. Leaf nodes (nodes without children) represent decisions or classifications. Evaluating the highest gain (most homogeneous branches) among all other features at each stage is the basic choice of a decision tree at each node. The performance of the decision tree is evaluated using a confusion matrix (Qawqzeh et al., 2019). Mathematically Entropy for multiple attributes is represented as:

$$E(T,X) = \sum_{c \in X} P(c)E(c) \quad (2)$$

2.5 Random Forest (RF)

Random forest classifier was used as the classification method. The random forest classifier consists of a combination of tree classifiers, in which each classifier is constructed using a random vector sampled independently of the input vector, and each tree gives a unit vote for the most popular class to classify an input vector, i.e. counts (L. J. U. o. C. B. Breiman, CA, USA, 1999). The design of a decision tree requires the selection of a feature selection measure and pruning method. There are many approaches to the selection of features used for decision tree induction, and most approaches directly add a measure

of quality to the attribute. The most frequently used attribute choices in decision tree induction are the information gain ratio criterion (Quinlan, 2014) and the Gini Index (L. Breiman, Friedman, Olshen, & Stone, 1984). The random forest classifier uses the Gini Index as an attribute selection measure that measures the impurity of attributes by classes. In short, the random forest classifier travels through the forests, making a progression towards the branches. When it reaches the end of the branches, it makes a vote. Here, n denotes the number of trees to be visited. For example, if we choose n as 5, the result is drawn from among 5 trees and a vote is made. As a result of voting, the most voted class or prediction result is obtained. In this way, a successful prediction or result opportunity is caught.

2.6 Naive Bayes (NB)

Naive Bayes assigns the most probable value in a sample space for feature extraction. The properties in the sample are treated as being independent of the given class and can be made very simple. Although it does not work very well in theory, it is seen that it gives better results than many classifiers in practice (Rish, 2001). It is based on a simple mathematical calculation and is as follows;

$$P(c|x) = \frac{P(c|x)P(c)}{P(x)} \quad (3)$$

The class's prediction probability is P(c|x). The class's prior probability is P(c). The probability of the class estimator is P(x|c), which is the probability. The estimator's prior probability is P(x). If a new sample is encountered, the class with the highest probability is found by considering the probability values calculated in finding the membership probability of this sample (Frank & Bouckaert, 2006). No estimation is made for a data in the test set if there is no counterpart in the training set. A straightforward yet effective approach for predictive modeling is the Naive Bayes method. Even with few data, it has great predictive power. Because of these beneficial and useful characteristics, it is a classifier that is preferred and used in many fields. A modest yet effective approach for predictive modeling is the Naive Bayes method. Even with few data, it has great predictive power. Because of these beneficial and useful characteristics, it is a classifier that is preferred and used in many fields (ŞAHİNASLAN, DALYAN, & ŞAHİNASLAN, 2022).

2.7 K-Nearest Neighbor (KNN)

K-nearest neighbor (K-NN) is a widely used classifier in classification (Wu, Ianakiev, & Govindaraju, 2002). It basically develops an estimate by interacting with neighbors around a certain diameter in the dataset. The larger the diameter, the more likely the features will be lost. Therefore, it is very important to use it at the optimum level when determining the number of neighbors.

K-Nearest Neighbor (KNN) classifier; The KNN algorithm is a widely used method in data mining. K-Nearest Neighbors (kNN) is a simple but effective non-parametric classification method in many situations. To classify a t data record, the k nearest neighbors are taken and this creates a neighborhood of t. Majority voting among neighborhood data records is often used to decide classification for t, with or without distance-based weighting (Guo, Wang, Bell, Bi, & Greer, 2003). In the study, KNN was used so that it can be used in comparison, even though it has a high time cost.

2.8 XGBoost (XGB)

Xgboost (Chen et al., 2015), developed by Chen et al., is an efficient and scalable implementation of the gradient boosting framework (Friedman J, 2000; Friedman, 2001). It has a tree learning-based structure. It has a linear solve function and it tries to increase low values in particular and it has a structure that allows this within the tree. It is a widely used and functional method with uses such as classification and regression. Since the packages applied are in an extensible state, it is possible to make an application for every problem. It has been used a lot in recent years because it generally shows high performance in machine learning algorithms. Therefore, this classifier is also used to compare and get results.

2.9 Evaluation Metrics

In order to compare the results of the study, some evaluation metrics are needed. In this way, the accuracy of the study and the superiority of machine learning methods to each other will be seen. Experimental results will reveal the final best performances within this framework. The metrics used in the study are as follows;

$$Accuracy = \frac{TP + TN}{TP+FP+TN+FN} \quad (4)$$

$$F1 - score = \frac{2 * Precision * Recall}{Precision + Recall} \quad (5)$$

$$Precision = \frac{TP}{TP+FP} \quad (6)$$

$$Recall = \frac{TP}{TP+TN} \quad (7)$$

Accuracy (4) is the quotient of correctly predicted results and all results, which actually means overall performance. F1-score (5) expresses the harmonic mean between precision and recall values. Precision (6) gives the ratio of true positive results to other positive results. Recall (7) value is the ratio of true positive values to true negative and true positive values (Hossin, Sulaiman, & process, 2015). Confusion Matrix must be created to calculate all these metrics.

3. EXPERIMENTS AND RESULTS

Experimental results are given in this section. While giving the results, feature selection and non-optimized machine learning techniques were applied first. Then, the results were optimized by making graph-based feature selection. While performing these operations, attributes were selected according to the graph usage evaluation. Processes were applied to make the data set ready for use first. Then, considering the correlative structure, the results of this structure were applied again on the graphs. In this way, while keeping the features that will increase the classification quality, low weights, that is, low similarities, are removed from the data set. The results of all these processes are demonstrated with performance metrics such as confusion matrix, ROC curve, accuracy. In these results, the scope of the evaluation metrics was kept wide.

Figure 5.*Confusion matrices obtained without feature selection*

As given in Figure 5, confusion matrices are given for each machine learning method. In this way, accuracy, precision, recall and f1-score values have become measurable. True False values are shown in darker colors compared to the number they contain so that they can be seen more clearly. Without any feature selection in this matrix, only the column that disrupts the algorithm structure was removed from the data set. Other than that, no action was taken. In addition, each process was carefully applied in the results without feature selection.

Table 2. *Without feature selection and optimization*

ML Algorithm	Accuracy	Precision	Recall	F1-score	Weighted ROC
Logistic Regression	95.73%	95.72%	95.73%	95.69%	98.43%
Decision Tree	96.49%	96.5%	96.49%	96.5%	96.68%

Random Forest	97.85%	97.84%	97.85%	97.84%	99.21%
Naive Bayes	53.01%	57.4%	53.01%	53.14%	68.32%
K-nearest Neighbor	89.77%	90.08%	89.77%	89.61%	95.21%
XGBoost	**97.91%**	**97.89%**	**97.91%**	**97.89%**	**99.22%**

In Table 2, all machine learning algorithms were measured using the evaluation metrics given in section 2.9. In this way, the results are given clearly. The fractional parts are given as two digits because some methods are very close to each other in terms of success. Even very small numbers are included to make it easy to distinguish.

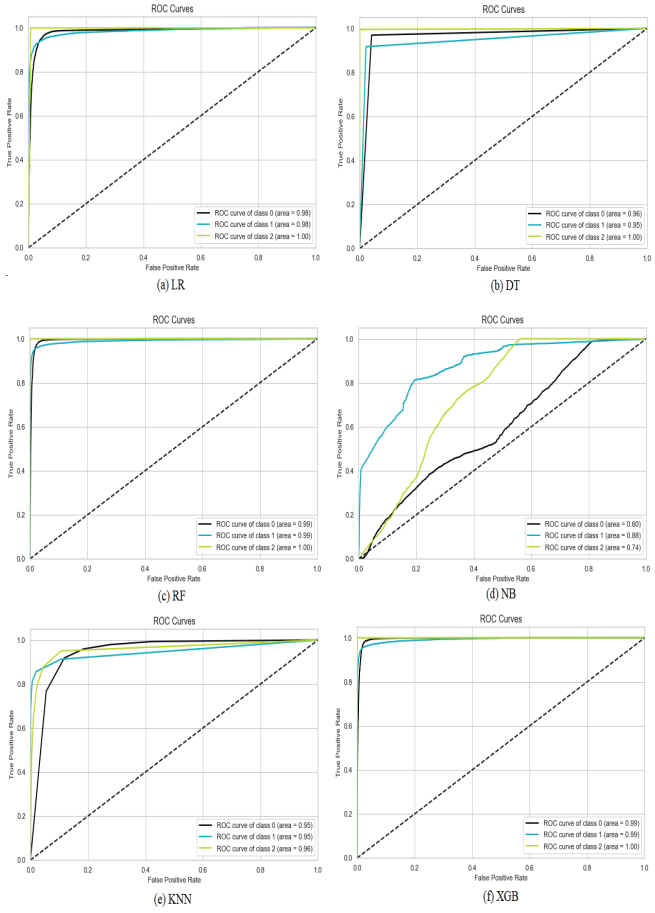


Figure 6. *ROC Curves obtained without feature selection*

It is very important to see the quality of the models created with the ROC Curve. For this reason, it is an important method for displaying multiple classes and comparing how well the working model gives results. These curves are evaluated as good as they get closer to 1 and as bad as they get closer to 0. From the results of applying machine learning techniques obtained from the data set without ROC Curve feature selection given in Figure 6. It is important for us to see the distribution of the classes separately for each class. In addition, ROC scores were evaluated in proportion to the number of classes while being processed into the tables. For example, since the GALAXY class constitutes 59.4% of all classes, it has a higher effect on the weighted value. Although the STAR classes with 21.6% and QSO classes with 19% are close to each other, their weights, namely their effects, have become less than the GALAXY class.

The experimental results given in Figure 7-8 and Table 3 are the results obtained after the graph-based feature selection. It is important to better understand the differences if they are shown as old and new when evaluating their performance. The use of different performance metrics to evaluate the results is important for the quality of the benchmark. Since there is not a clear criterion due to the lack of studies with Stellar classification, it would be more accurate to compare with old values. Relevant comparisons and evaluations are made in section 4.

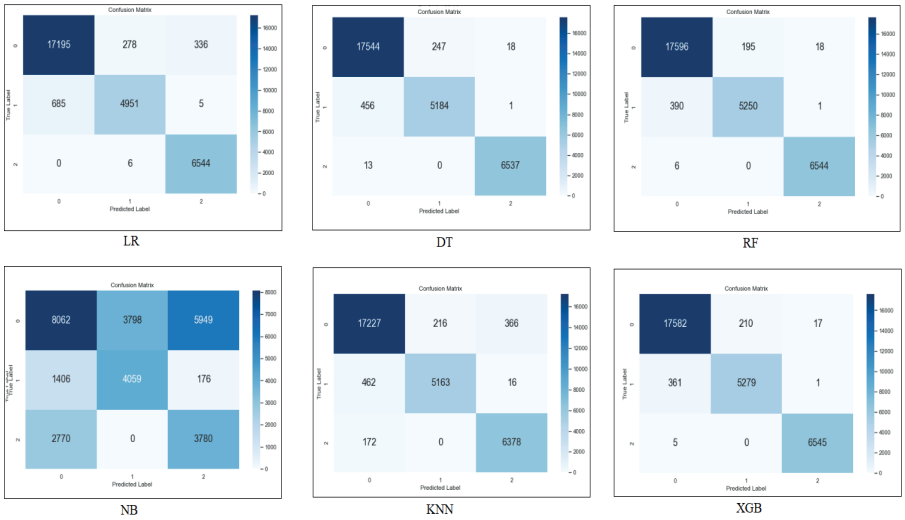


Figure 7. *Confusion matrices obtained with graph-based feature selection and optimization*

Table 3. *After feature selection and optimization*

ML Algorithm	Accuracy	Precision	Recall	F1-score	Weighted ROC
Logistic Regression	95.63%	95.62%	95.63%	95.59%	98.27%

Decision Tree	97.55%	97.54%	97.55%	97.53%	99.21%
Random Forest	97.97%	97.96%	97.97%	97.95%	99.21%
Naive Bayes	53%	57.15%	53%	53.2%	67.53%
K-nearest Neighbor	95.89%	95.89%	95.89%	95.88%	98.21%
XGBoost	**98.02%**	**98%**	**98.02%**	**98%**	**99.81%**

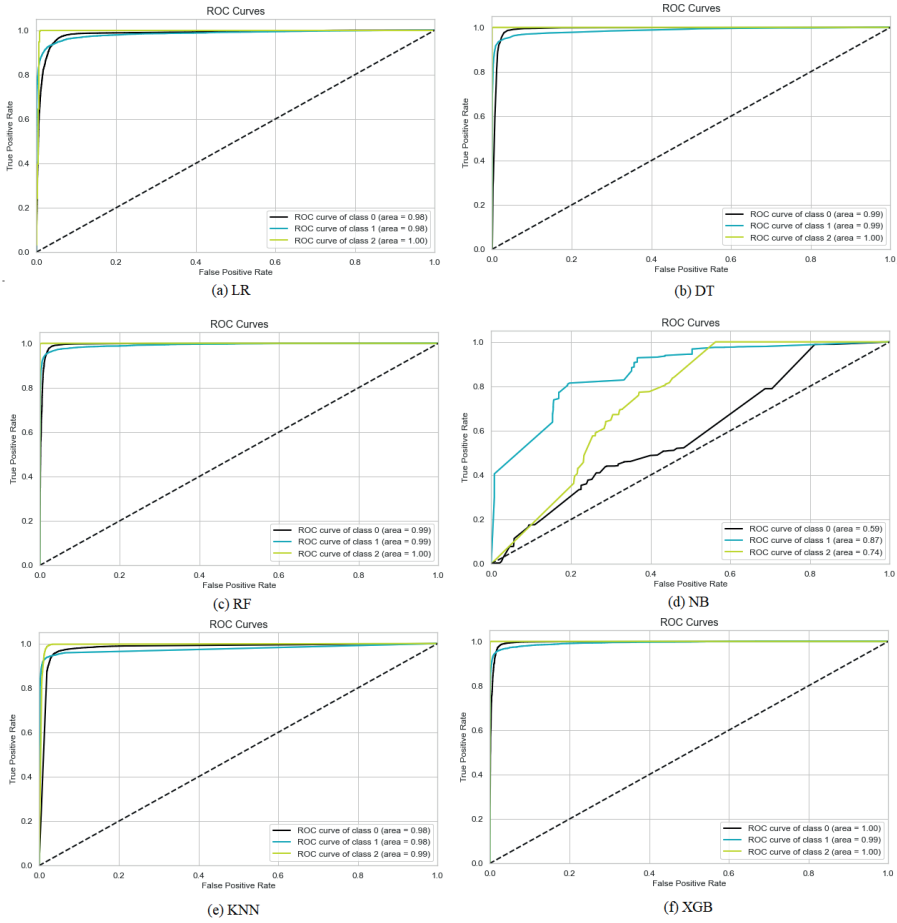


Figure 8.*ROC Curves obtained with graph-based feature selection and optimization*

4. DISCUSSION

Experimental results were obtained with machine learning methods using the data set. Looking at the results, the results without feature selection and optimization were not so bad, but lower than graph-based feature selection. Looking at the results without feature selection in the proposed method, the ini-

tial values for XGBoost are accuracy 97.91%, precision 97.89%,recall 97.91%, F1-score 97.89%, weigted ROC score 99.22% by graph-based feature selection, accuracy 98.02%, precision 98%, recall, 98.02%, F1-score 98%, weigted ROC score 99.81%. The highest machine learning technique is XGBoost. However, Random Forest was also very high. The ROC Curves are also shown as proof of improvement. results close to 1 confirm the application of the model and the quality of the results obtained. In this study, machine learning methods were compared and presented with various metrics.

5. SUMMARY and CONCLUSİON

A data set containing 100,000 pieces of data was used. The study dataset is a large-scale dataset from sky observations referred to as SDSS R17 and made available to everyone. In the study; Logistic Regression, Decision Tree Classifier, Random Forest Classifier, Naive Bayes, K-nearest Neighbor, XGBoost were used. As Evaluation metrics; accuracy, precision, recall, F1-score and ROC score were used. Thanks to these metrics, it can be understood how well the model is working. Pre-processing was done to create the model. Graph-based feature selection has been made for feature selection. While making this approach, nodes and edges were created considering the correlative structure. The model was created considering that the data weighted nodes and edges with classes in the center approach the center and this will affect the result of the classes in this center. Then, machine learning techniques were applied with these selected features. It is seen that better results are obtained in the models created than in the past. Classification has been done and satisfactory results have been obtained.

A computer aided identification can be made with Stellar classification. In this way, it will be beneficial to distinguish celestial objects that cannot be seen or distinguished with the naked eye. With these methods, it is possible to obtain more convenient, faster and high-accuracy practical results with computer classification of different celestial bodies. By working with other data sets, classification operations can be done and a preferable model has been put forward because it makes the operations short and with high accuracy. Since it is successful in multi-class classification, it is a model that can be used as an applicable method for almost every data set.

REFERENCES

Accetta, K., Aerts, C., Aguirre, V. S., Ahumada, R., Ajgaonkar, N., Ak, N. F., . . . Anders, F. J. T. A. J. S. S. (2022). The Seventeenth Data Release of the Sloan Digital Sky Surveys: Complete Release of MaNGA, MaStar, and APOGEE-2 Data. *259*(2), 35.

Armstrong, D. J., Pollacco, D., & Santerne, A. J. M. N. o. t. R. A. S. (2016). Transit shapes and self organising maps as a tool for ranking planetary candidates: Application to kepler and k2. stw2881.

Bai, Y., Liu, J., Wang, S., & Yang, F. J. T. A. J. (2018). Machine Learning Applied to Star–Galaxy–QSO Classification and Stellar Effective Temperature Regression. *157*(1), 9.

Bailey, S., Aragon, C., Romano, R., Thomas, R. C., Weaver, B. A., & Wong, D. J. T. A. J. (2007). How to find more supernovae with less work: Object classification techniques for difference imaging. *665*(2), 1246.

Ball, N. M., & Brunner, R. J. J. I. J. o. M. P. D. (2010). Data mining and machine learning in astronomy. *19*(07), 1049-1106.

Breiman, L., Friedman, J., Olshen, R., & Stone, C. J. B. R., Florida. (1984). Classification and regression trees–crc press.

Breiman, L. J. U. o. C. B., CA, USA. (1999). Random Forests; UC Berkeley TR567.

Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., & Chen, K. J. R. p. v.-. (2015). Xgboost: extreme gradient boosting. *1*(4), 1-4.

Frank, E., & Bouckaert, R. R. (2006). *Naive bayes for text classification with unbalanced classes.* Paper presented at the European Conference on Principles of Data Mining and Knowledge Discovery.

Friedman J, H. T., Tibshirani R, et al. . (2000). "Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors).". *The annals of statistics, 28(2)*, 337–407.

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

Guo, G., Wang, H., Bell, D., Bi, Y., & Greer, K. J. L. N. i. C. S. (2003). Ontologies, Databases, and Applications of Semantics (ODBASE) 2003 International Conference-Data Semantics and Metadata-KNN Model-Based Approach in Classification. *2888*, 986-996.

Hashemi, A., Dowlatshahi, M. B., & Nezamabadi-Pour, H. J. E. S. w. A. (2020). MGFS: A multi-label graph-based feature selection algorithm via PageRank centrality. *142*, 113024.

Hinners, T. A., Tat, K., & Thorp, R. J. T. A. J. (2018). Machine learning techniques for stellar light curve classification. *156*(1), 7.

Hossin, M., Sulaiman, M. N. J. I. j. o. d. m., & process, k. m. (2015). A review on evaluation metrics for data classification evaluations. *5*(2), 1.

Kaggle. Stellar Classification Dataset - SDSS17,2022. https://www.kaggle.com/datasets/fedesoriano/stellar-classification-dataset-sdss17 (Last Access: 27.06.2022).

Qawqzeh, Y. K., Otoom, M. M., Al-Fayez, F., Almarashdeh, I., Alsmadi, M., & Jaradat, G. J. I. (2019). A proposed decision tree classifier for atherosclerosis prediction and classification. *19*(12), 197.

Quinlan, J. R. (2014). *C4. 5: programs for machine learning*: Elsevier.

Rish, I. (2001). *An empirical study of the naive Bayes classifier.* Paper presented at the IJCAI 2001 workshop on empirical methods in artificial intelligence.

Roffo, G., Melzi, S., Castellani, U., Vinciarelli, A., Cristani, M. J. I. T. o. P. A., & Intelligence, M. (2020). Infinite feature selection: a graph-based feature filtering approach. *43*(12), 4396-4410.

Safavian, S. R., Landgrebe, D. J. I. t. o. s., man,, & cybernetics. (1991). A survey of decision tree classifier methodology. *21*(3), 660-674.

Sperandei, S. J. B. m. (2014). Understanding logistic regression analysis. *24*(1), 12-18.

ŞAHİNASLAN, Ö., DALYAN, H., & ŞAHİNASLAN, E. J. B. T. D. (2022). Naive Bayes Sınıflandırıcısı Kullanılarak YouTube Verileri Üzerinden Çok Dilli Duygu Analizi. *15*(2), 221-229.

Thompson, S. E., Mullally, F., Coughlin, J., Christiansen, J. L., Henze, C. E., Haas, M. R., & Burke, C. J. J. T. A. J. (2015). A machine learning technique to identify transit shaped signals. *812*(1), 46.

Vilalta, R., Gupta, K. D., Macri, L. J. A., & Computing. (2013). A machine learning approach to Cepheid variable star classification using data alignment and maximum likelihood. *2*, 46-53.

Viquar, M., Basak, S., Dasgupta, A., Agrawal, S., Saha, S. J. E. T. i. D. M., & Security, I. (2019). Machine learning in astronomy: A case study in quasar-star classification. 827-836.

Wu, Y., Ianakiev, K., & Govindaraju, V. J. P. r. (2002). Improved k-nearest neighbor classification. *35*(10), 2311-2318.

Yasmin, G., Das, A. K., Nayak, J., Pelusi, D., & Ding, W. J. E. S. w. A. (2020). Graph based feature selection investigating boundary region of rough set for language identification. *158*, 113575.

Zhang, Z., & Hancock, E. R. (2011). *A graph-based approach to feature selection.* Paper presented at the International workshop on graph-based representations in pattern recognition.

Zhu, X., Wan, Z., Tsang, D. C., He, M., Hou, D., Su, Z., & Shang, J. J. C. E. J. (2021). Machine learning for the selection of carbon-based materials for tetracycline and sulfamethoxazole adsorption. *406*, 126782.

CHAPTER 5

ARTIFICIAL INTELLIGENCE IN AGRICULTURE: THE ROLE OF COMPUTER VISION

Zehra YÜCEL[1]

1 Title: Lecturer Dr.
Institution: Necmettin Erbakan University
ORCID ID: 0000000228639119
Email: zehra.krhn@gmail.com

1. **Digital Transformation in Agriculture**

The global population is increasing, and as a result, the demand for food is also rising. The Food and Agriculture Organization (FAO) predicts that food demand will increase by nearly 70% by 2050. Along with population growth, the need for housing is also rising, and as more housing is constructed, the amount of arable land is decreasing (Alexandratos, N., & Bruinsma, J., 2012). The agricultural industry is become strategically significant in this regard. Growing interest is being paid to the necessity of expanding production activities and figuring out how to raise yields. Agriculture is undergoing a digital transformation to increase product productivity and quality and to anticipate certain potential outcomes.

Industry 4.0 can be considered as a specialized field under the concept of 'digital transformation in agriculture.' The impact of digitalization on agriculture is presented under several headings in Figure 1.

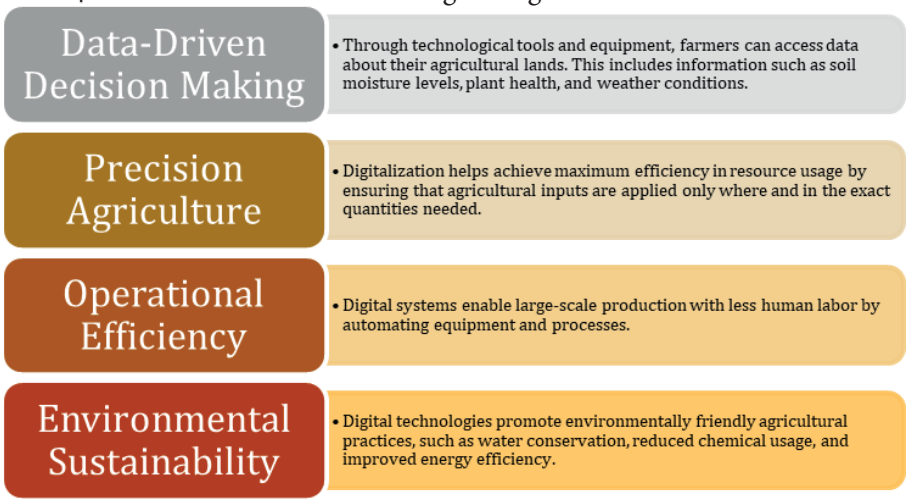


***Figure 1**. The impact of digitalization on agricultural practices*

Through digitalization in agriculture, production costs can also be reduced. By monitoring factors such as irrigation and harvest times, climate conditions, and soil health, resources can be used more effectively. This enables eco-friendly and sustainable production (Ghazal et al., 2024).

Although the transition to digitalization in agriculture is slower compared to other sectors, it remains an important area. A study conducted in the UK in 2015 saw an investment of 320 million dollars to support Agriculture 4.0 applications, which led to a 15% increase in wheat production. Similarly, in the United States, a 20% reduction in water usage for almond production was achieved. To determine water requirements, NASA launched an observation satellite to measure soil moisture, which transmits information on drought,

floods, and climate change every three days.

An agricultural equipment manufacturer in the US managed to reduce fuel costs by about 40% in fertilization and pesticide application by adding GPS sensors to their machines. The Netherlands, a leading exporter of advanced agricultural technologies, has successfully improved production efficiency and reduced costs through applications such as irrigation systems, renewable energy systems, big data analysis, and smart farming software.

When discussing the topic of digital transformation, the concept of Artificial Intelligence (AI), which is currently a popular subject, is doubtless the first to come to mind.AI can be defined as the process of enabling machines to develop cognitive abilities similar to those of humans. It supports real-time monitoring and regulation of processes and automates tasks related to the digitalization of agriculture, particularly in precision farming. The aim of automating processes with AI is to maximize efficiency and minimize costs (Liu, 2020; Talaviya et al., 2020; Ben Ayed, R., & Hanana, M., 2021; Sharma et al., 2022).

The increasing global population, along with limited arable land, indicates the need for AI in agriculture (Mohr, S., & Kühl, R., 2021; Sharma, 2021). In addition, the agricultural sector requires intensive labor for tasks such as smart pesticide application, irrigation, and fertilization systems. AI has the potential to greatly diminish the demand for human labor.

AI in agriculture is shaped by technologies such as big data, the Internet of Things (IoT), and computer vision. Computer vision has a wide range of applications in agriculture. It is utilized, for instance, in fields like soil analysis, weed identification, disease diagnosis, agricultural yield calculation, and crop condition assessment. Computer vision automates processes by analyzing images.

The application of computer vision in agriculture holds significant importance in terms of sustainability, food security, and increased productivity.

2. **Computer Vision Technologies**

The use of AI in agriculture can be defined as an area that encompasses many disciplines and technologies. As technology continues to develop, the use and advancement of AI in agriculture will accelerate. The field of computer vision holds great potential for increasing efficiency in agriculture (Rehman et al., 2019; Patrício, 2018). The general flow of the application of computer vision in agriculture is shown in Figure 2. In this process, image acquisition, image processing, classification or detection, and decision-making for the appropriate action are defined as key steps. For each step, it is crucial to determine the appropriate actions and algorithms based on the problem's characteristics and the desired objectives. The most important aspect is analyzing the problem well. Each problem is

unique, and the workflow applied will vary accordingly; there is no universal method. Since existing technologies and methods are implemented with an experimental platform, the data and results obtained will differ due to natural factors (Huang et al., 2017).

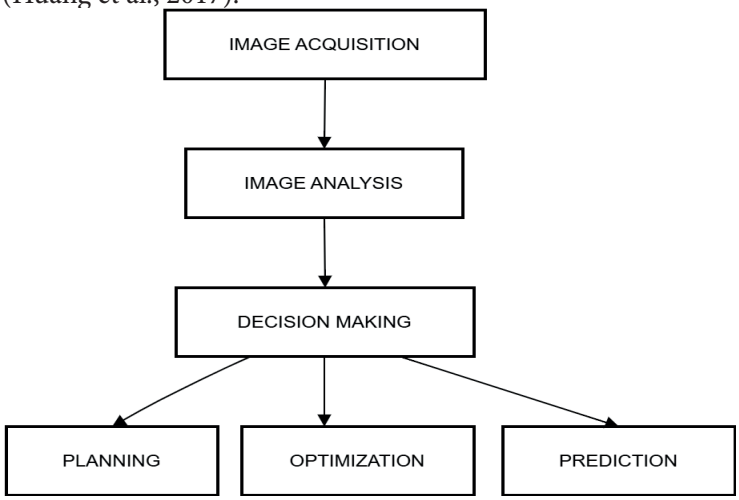


Figure 2. *AI-based agricultural cycle: computer vision steps*

2.1 Image Acquisition

To obtain images of agricultural land, drones, satellite images, or ground detection robots are used. These technologies enable soil condition analysis, yield forecasting, and more efficient crop planning. They also provide the ability for real-time monitoring. The techniques for image acquisition may include RGB, multispectral, hyperspectral, and thermal imaging. The most important factor here is obtaining images suitable for the purpose. The appropriate technique is chosen based on the processing to be carried out.

✓ RGB Imaging: This includes the red, green, and blue bands of the electromagnetic spectrum. It is a commonly used imaging method, and the electromagnetic spectrum ranges from 400 to 700 nm (Liu et al., 2022).

✓ Multispectral Imaging**:** This includes the RGB channels along with near and far infrared and some additional spectral bands of the electromagnetic spectrum, usually fewer than 10 bands (Ghazal, S., Munir, A., & Qureshi, W. S., 2024).

✓ Hyperspectral Imaging: This includes hundreds of bands of the electromagnetic spectrum, with a spectral resolution of 10-20 nm and continuous narrow bands. It has high spatial and temporal resolution, which means more information is included. However, the excessive memory consumption for large amounts of data can be a disadvantage (Ram et al., 2024).

✓ Thermal Imaging: This captures the infrared radiation emitted by an

object to generate its thermal image. Since the infrared radiation of plants that require water differs from those that do not, their thermal images are distinct (Ghazal, S., Munir, A., & Qureshi, W. S., 2024).

2.2 Image Analysis

This phase involves extracting relevant information from the obtained images based on the problem at hand. The goal here is to have the computer perform tasks that humans can easily distinguish when observing the image externally. With preprocessing steps, plant recognition involves extracting image features (such as color, shape, texture, etc.) relevant to the desired problem. This can be defined as extracting information from the image.

Image analysis is divided into three categories based on the extracted information (Patrício, D. I., & Rieder, R., 2018):

✓ Low-Level Image Analysis: This involves tasks like adjusting tonality and contrast, as well as noise reduction. Methods in low-level analysis include geometric transformations, Gaussian filtering, and so on.

✓ Mid-Level Image Analysis: This involves extracting distinguishing information to isolate or identify desired objects or regions within an image. Methods in mid-level analysis include Laplacian, Gabor filters, bounding box techniques, Canny edge detection, Haar features, etc.

✓ High-Level Image Analysis: This phase involves conducting analyses on the image to produce the desired outcome. High-level methods are dominated by deep learning networks, which are a key focus of AI (particularly deep learning in agricultural applications).

2.3. Deep Learning Models

In traditional learning approaches, the steps outlined in Figure 2 are performed sequentially, leading to a final decision. With the advent of deep learning models, these steps are carried out automatically through deep learning methods following the acquisition of the input image. Compared to traditional learning methods, deep learning generally provides significant improvements. The structure of deep learning can be described, in its simplest form, as being inspired by the structure of the human brain (Dhanya et al., 2022).

Deep learning autonomously extracts features (e.g., texture, color, shape, statistical properties) and performs the decision-making process to deliver relevant outputs to the user. The decision-making mechanism can be adjusted according to the user's requirements (e.g., detection, classification, segmentation), and deep learning generates the desired results accordingly.

The architecture of deep learning, which is based on artificial neural networks, consists of multiple hidden layers that facilitate the learning process. These layers process input images to perform tasks such as detecting weeds,

identifying diseases on leaves, or counting fruits and vegetables. Starting from a raw input image, the network extracts features and learns which of these features are most distinctive to produce the desired output.

The foundation of deep learning networks and one of the most widely used architectures is the Convolutional Neural Network (CNN). Its general structure, including CNN layers, is illustrated in Figure 3. For instance, the classification of an apricot image as high or low quality using a CNN network is demonstrated. A basic CNN architecture typically includes a convolutional layer, a pooling layer for dimensionality reduction, a flattening layer for processing two-dimensional data, followed by a fully connected layer, and finally, a classification layer (e.g., good, poor, etc.) to process the input image (Othman et al., 2023; Karim et al., 2024).

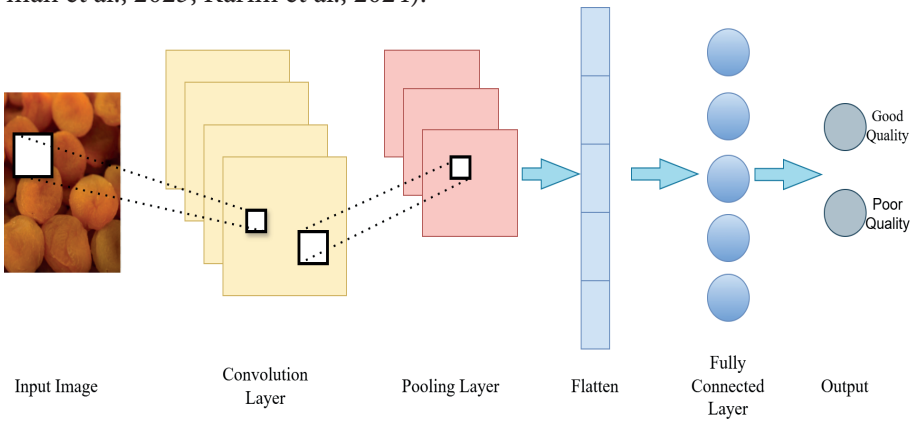


Figure 3. *General structure of a CNN*

3. Applications of Computer Vision

The use of advanced technologies in agriculture is inevitable in order to increase productivity. Artificial Intelligence (AI) in agriculture enables more informed decision-making regarding issues such as labor, water usage, and climate change by transforming agricultural processes. Especially with the increasing volume of data, access to big data makes decision-making easier. This technology is widely used in various areas of agriculture.

A summary of publications related to the use of computer vision in AI applications in agriculture is presented in Table 1. The tasks carried out are categorized into soil analysis, plant species identification and detection, disease diagnosis and detection, quality analysis, and yield prediction. Computer vision is utilized within these categories in agriculture.

In soil analysis, information extraction from images is used for classification tasks. Features such as soil texture, moisture, and pH value are considered. Studies in this group focus on various soil types (e.g., Red Soil, Alluvial

Soil, Forest Soil, Saline Soil, Regur Soil, Desert/Arid Soil, Sub-mountain Soil, Swamp/Peaty Soil, Laterite Soil, Snow-covered Areas). Additionally, studies using spectral databases such as the Land Use/Land Cover Area Frame Survey (LUCAS) (Tóth, G., Jones, A., & Montanarella, L., 2013) are also available. Unmanned Aerial Vehicles (UAVs) and Unmanned Ground Vehicles (UGVs), along with artificial satellites such as the European Sentinel-2 System, the RapidEye constellation, the GeoEye-1 system, and WorldView-3, as well as multispectral satellite data sources like Landsat 8, enable the collection of valuable data (Liu et al., 2020). A summary of studies related to soil analysis is presented in Table 1.

Table 1. *Soil Analysis Studies*

Topic	Techniques Used	References
Subsurface Soil Characterization	Gray Level Co-occurrence Matrix (GLCM)	Khomiak, O., Benndorf, J., & Verbeek, G., 2024
Prediction of Organic Matter Content in Soil	Image and spectral feature fusion for multi-CNN	Li et al., 2024
Soil Analysis in Western Azerbaijan	CNN, ANN, SVM, RF, and KNN	Azadnia et al., 2022
Soil Type Analysis from LUCAS Database	Deep Convolutional Neural Networks (DCNN)	Zhong et al., 2021
Spatial Variability of Soil Nitrogen Content	Partial Least Squares Regression (PLSR)	Pechanec et al., 2021
Prediction of Organic Carbon in Soil using Hyperspectral Satellite Data	Discrete Wavelet Transform (DWT) + Random Forest (RF), Support Vector Machine (SVM), and Backpropagation Neural Network (BPNN)	Meng et al., 2020

Plant species recognition and detection, along with weed detection, help identify harmful or unwanted plants. Disease diagnosis and detection are crucial for sustainable agriculture. Early disease detection enables corrective actions to improve crop productivity. Computer vision technologies automate these processes, providing fast and efficient management. Publicly accessible datasets, such as TurkeyPlantDataset, PlantDoc, and datasets for crops like strawberries and rice, are available for these tasks (Shafik et al., 2023; Singh et al., 2020; Nie et al., 2019; Ramesh, S., & Vydeki, D., 2020).

Some studies in the category of plant species recognition, detection, and disease diagnosis are summarized in Table 2.

Table 2. *Plant Species Recognition, Detection, and Disease Diagnosis Studies*

Topic	Techniques Used	References
Tomato Disease Image Recognition	Data-efficient image transformers (DeiT) + Exponential Moving Average (EMA)	Sun et al., 2025
Plant Disease Recognition	RGB, HSL, HSV, LAB, LUV, XYZ, YUV color spaces + CNN	Nain et al., 2024
Corn Plant Detection	YOLOv5	Lu et al., 2024
Rice Leaf Disease Detection	CNN	Kulkarni, P., & Shastri, S., 2024
Plant Image Classification (fruits, vegetables, flowers, trees, etc. and their leaves)	CNN	Batchuluun et al., 2022
Corn Leaf Disease Detection	Multi-axis Vision Transformer (MaxViT)	Pacal, I., 2024
Coffee Leaf Disease Recognition	CoffeeNet: CenterNet with spatial-channel attention	Nawaz et al., 2024

Seed quality analysis is fundamental to improving agricultural productivity. High-quality seeds lead to higher crop yields. When classifying seeds based on quality, features like color, shape, and texture are considered. Automating this process using computer vision is important.

Some studies related to seed quality analysis are summarized in Table 3.

Table 3. *Seed Quality Analysis Studies*

Topic	Techniques Used	References
Crack Detection in Corn Seeds	YOLOv8	Chen et al., 2024
Feature Extraction and Classification of Seed Images	Morphological, texture, and color features + RF	Loddo et al., 2023
Selection of High-Quality Pepper Seeds	Three color features (R, a*, brightness), width, length, reflected area, and single core density and weight + Multilayer Perceptron (MLP)	TU et al., 2018
Internal Quality Detection of Seeds	Terahertz Imaging Model (CS-IRE) = Compressed Sensing (CS) + Improved Real ESRGAN (IRE)	Jin-li et al., 2024
Internal Quality Detection of Sugar Beet and Fava Bean Seeds	X-ray Data Processing Approach (X-Robustifier)	Hamdy et al., 2024

Rice Seed Classification	CNN	Wibowo et al., 2024
Classification of Corn Seed Varieties	Conventional feature extraction + CNN + Artificial Neural Network (ANN), Cubic SVM, Quadratic SVM, K-Nearest Neighbor (KNN), Boosted Tree, Bagged Tree, Linear Discriminant Analysis (LDA)	Javanmardi et al., 2021

Yield prediction studies help farmers anticipate outcomes and make informed decisions or plan accordingly. Yield forecasting affects not only farmers but also government food trade planning. Some studies related to yield prediction are summarized in Table 4.

Table 4. *Yield Prediction Studies*

Topic	Techniques Used	References
Rice Yield Prediction	RaNN: Random Forest + Multilayer Feedforward Neural Network	Lingwal et al., 2024
Corn Grain Yield Prediction	High Spatial-Temporal Resolution Imaging + RF, Linear Regression (LR)	Killeen et al., 2024
Blueberry Maturity Stage Detection and Yield Prediction	YOLOv4	MacEachern et al., 2023
Apple Orchard Production Forecasting	Deep Simple Online Realtime (DeepSORT)	Villacrés et al., 2023
Wheat Yield Prediction	SVM, Deep Neural Network (DNN), Ridge Regression (RR), RF	Fei et al., 2023
Citrus Yield Prediction	Faster R-CNN, LSTM	Apolo-Apolo et al., 2020
Corn Yield Prediction	CNN, Recurrent Neural Network (RNN)	Sun et al., 2020
Chestnut Fruit Detection, Counting, and Yield Prediction	YOLOv4	Arakawa et al., 2024

4. Challenges and Future Prospects

The shift towards AI-driven technologies in agriculture is aimed at increasing production and sustainability. Computer vision plays a vital role in this transformation. By using image-based tools in agriculture, both cost and time can be saved in processes where decisions are made automatically. Most importantly, this contributes to increased productivity and sustainability in agriculture. However, several challenges need to be addressed for successful implementation.

• Image Quality: The first requirement for working with images and producing image-based results is that the image quality must be high. It is essential that the image production process is minimally affected by lighting condi-

tions and weather factors.

• Insufficient Standardized Datasets: The lack of large, standardized datasets negatively impacts progress in this field.

• Challenging Agricultural Conditions: Difficult conditions in agricultural fields can result in misleading image data, which may affect the final outcomes.

• Data Insufficiency: The lack of sufficient data can hinder the extraction of accurate information during training or feature extraction from images.

• Model generalization: A model that is trained on one crop should yield results that are comparable to those of other crops; this is known as model generalization.

• Insufficient Interdisciplinarity: The advancement of computer science and agriculture is impeded by the absence of interdisciplinarity.

• Costly Real-Time Systems: Real-time systems are expensive, and solving the issues mentioned will produce better outcomes.

The process of digital transformation in agriculture relies heavily on the physical, economic, legal, and infrastructural support provided by governments. At the same time, overcoming these challenges requires significant effort from farmers. Farmers must not remain distant from these technologies; instead, they should be equipped with the necessary skills through well-structured education and training programs, such as digital literacy courses, agricultural technology centers, and incentive schemes. Moreover, the collaboration between farmers and companies is a crucial driver for advancing digitalization in agriculture. For instance, initiatives like digital cooperatives or partnerships with technology providers can play a transformative role.

In the field of artificial intelligence (AI) in agriculture, computer vision holds important position and is expected to maintain its significance in the future. Therefore, addressing the challenges to maximize the potential of computer vision in agriculture is of utmost importance. Continued research and development in this area can bring us closer to solving critical future issues, such as hunger and food scarcity. A productive, sustainable, and food-secure agricultural model for the future can be realized through the effective integration of these technologies.

REFERENCES

Alexandratos, N., & Bruinsma, J. (2012). World agriculture towards 2030/2050: the 2012 revision.

Apolo-Apolo, O. E., Martínez-Guanter, J., Egea, G., Raja, P., & Pérez-Ruiz, M. (2020). Deep learning techniques for estimation of the yield and size of citrus fruits using a UAV. *European Journal of Agronomy*, *115*, 126030.

Arakawa, T., Tanaka, T. S., & Kamio, S. (2024). Detection of on-tree chestnut fruits using deep learning and RGB unmanned aerial vehicle imagery for estimation of yield and fruit load. *Agronomy Journal*, *116*(3), 973-981.

Azadnia, R., Jahanbakhshi, A., Rashidi, S., & Bazyar, P. (2022). Developing an automated monitoring system for fast and accurate prediction of soil texture using an image-based deep learning network and machine vision system. *Measurement*, *190*, 110669.

Barman, U., & Choudhury, R. D. (2020). Soil texture classification using multi class support vector machine. *Information processing in agriculture*, *7*(2), 318-332.

Batchuluun, G., Nam, S. H., & Park, K. R. (2022). Deep learning-based plant-image classification using a small training dataset. *Mathematics*, *10*(17), 3091.

Ben Ayed, R., & Hanana, M. (2021). Artificial intelligence to improve the food and agriculture sector. *Journal of Food Quality*, *2021*(1), 5584754.

Chen, J., Chen, J., Zhang, D., Sun, Y., & Nanehkaran, Y. A. (2020). Using deep transfer learning for image-based plant disease identification. *Computers and Electronics in Agriculture*, *173*, 105393.

Chen, S., Li, Y., Zhang, Y., Yang, Y., & Zhang, X. (2024). Soft X-ray image recognition and classification of maize seed cracks based on image enhancement and optimized YOLOv8 model. *Computers and Electronics in Agriculture*, *216*, 108475.

Fabiyi, S. D., Vu, H., Tachtatzis, C., Murray, P., Harle, D., Dao, T. K., ... & Marshall, S. (2020). Varietal classification of rice seeds using RGB and hyperspectral images. *IEEE Access*, *8*, 22493-22505.

Fei, S., Hassan, M. A., Xiao, Y., Su, X., Chen, Z., Cheng, Q., ... & Ma, Y. (2023). UAV-based multi-sensor data fusion and machine learning algorithm for yield prediction in wheat. *Precision agriculture*, *24*(1), 187-212.

Ghazal, Sumaira, Arslan Munir, and Waqar S. Qureshi. "Computer vision in smart agriculture and precision farming: Techniques and applications." *Artificial Intelligence in Agriculture* (2024).

Hamdy, S., Charrier, A., Corre, L. L., Rasti, P., & Rousseau, D. (2024). Toward robust and high-throughput detection of seed defects in X-ray images via deep learning. *Plant Methods*, *20*(1), 63.

Huang, X., Bi, J., Zhang, N., Ding, X., Li, F., & Hou, F. (2017). Application of computer vision technology in agriculture. *Agricultural Science & Technology*, *18*(11), 2158-2162.

Javanmardi, S., Ashtiani, S. H. M., Verbeek, F. J., & Martynenko, A. (2021). Computer-vision classification of corn seed varieties using deep convolutional neural network. *Journal of Stored Products Research*, *92*, 101800.

Jin-li, Y., Bin, L., Yang, A., Zhao-xiang, S., Xia, W., Aiguo, O., & Yan-de, L. (2024). A generalized model for seed internal quality detection based on terahertz imaging technology combined with image compressed sensing and improved-real ESRGAN. *Microchemical Journal*, 112410.

Khomiak, O., Benndorf, J., & Verbeek, G. (2024). Sub-Surface Soil Characterization Using Image Analysis: Material Recognition Using the Grey Level Co-Occurrence Matrix Applied to a Video-CPT-Cone. *Mining*, *4*(1), 91-105.

Killeen, P., Kiringa, I., Yeap, T., & Branco, P. (2024). Corn grain yield prediction using UAV-based high spatiotemporal resolution imagery, machine learning, and spatial cross-validation. *Remote Sensing*, *16*(4), 683.

Kulkarni, P., & Shastri, S. (2024). Rice leaf diseases detection using machine learning. *Journal of Scientific Research and Technology*, 17-22.

Li, H., Ju, W., Song, Y., Cao, Y., Yang, W., & Li, M. (2024). Soil organic matter content prediction based on two-branch convolutional neural network combining image and spectral features. *Computers and Electronics in Agriculture*, *217*, 108561.

Lingwal, S., Bhatia, K. K., & Singh, M. (2024). A novel machine learning approach for rice yield estimation. *Journal of Experimental & Theoretical Artificial Intelligence*, *36*(3), 337-356.

Liu, G., Tian, S., Mo, Y., Chen, R., & Zhao, Q. (2022). On the acquisition of high-quality digital images and extraction of effective color information for soil water content testing. *Sensors*, *22*(9), 3130.

Liu, S. Y. (2020). Artificial intelligence (AI) in agriculture. *IT professional*, *22*(3), 14-15.

Liu, Y., Ma, X., Shu, L., Hancke, G. P., & Abu-Mahfouz, A. M. (2020). From industry 4.0 to agriculture 4.0: Current status, enabling technologies, and research challenges. *IEEE transactions on industrial informatics*, *17*(6), 4322-4334.

Loddo, A., Di Ruberto, C., Vale, A. M. P. G., Ucchesu, M., Soares, J. M., & Bacchetta, G. (2023). An effective and friendly tool for seed image analysis. *The Visual Computer*, *39*(1), 335-352.

Lu, D., Ye, J., Wang, Y., & Yu, Z. (2023). Plant detection and counting: Enhancing precision agriculture in UAV and general scenes. *IEEE Access*.

Lu, C., Nnadozie, E., Camenzind, M. P., Hu, Y., & Yu, K. (2024). Maize plant detection using UAV-based RGB imaging and YOLOv5. *Frontiers in Plant Science*, *14*, 1274813.

MacEachern, C. B., Esau, T. J., Schumann, A. W., Hennessy, P. J., & Zaman, Q. U. (2023). Detection of fruit maturity stage and yield estimation in wild blueberry using deep learning convolutional neural networks. *Smart Agricultural Technology*, *3*, 100099.

Meng, X., Bao, Y., Liu, J., Liu, H., Zhang, X., Zhang, Y., ... & Kong, F. (2020). Regional

soil organic carbon prediction model based on a discrete wavelet analysis of hyperspectral satellite data. *International Journal of Applied Earth Observation and Geoinformation*, *89*, 102111.

Mohr, S., & Kühl, R. (2021). Acceptance of artificial intelligence in German agriculture: an application of the technology acceptance model and the theory of planned behavior. *Precision Agriculture*, *22*(6), 1816-1844.

Nain, S., Mittal, N., & Hanmandlu, M. (2024). CNN-based plant disease recognition using colour space models. *International Journal of Image and Data Fusion*, 1-14.

Nawaz, M., Nazir, T., Javed, A., Amin, S. T., Jeribi, F., & Tahir, A. (2024). CoffeeNet: A deep learning approach for coffee plant leaves diseases recognition. *Expert Systems with Applications*, *237*, 121481.

Nie, X., Wang, L., Ding, H., & Xu, M. (2019). Strawberry verticillium wilt detection network based on multi-task learning and attention. *IEEE access*, *7*, 170003-170011.

Pacal, I. (2024). Enhancing crop productivity and sustainability through disease identification in maize leaves: Exploiting a large dataset with an advanced vision transformer model. *Expert Systems with Applications*, *238*, 122099.

Patrício, D. I., & Rieder, R. (2018). Computer vision and artificial intelligence in precision agriculture for grain crops: A systematic review. *Computers and electronics in agriculture*, *153*, 69-81.

Pechanec, V., Mráz, A., Rozkošný, L., & Vyvlečka, P. (2021). Usage of airborne hyperspectral imaging data for identifying spatial variability of soil nitrogen content. *ISPRS International Journal of Geo-Information*, *10*(6), 355.

Ram, B. G., Oduor, P., Igathinathane, C., Howatt, K., & Sun, X. (2024). A systematic review of hyperspectral imaging in precision agriculture: Analysis of its current state and future prospects. *Computers and Electronics in Agriculture*, *222*, 109037.

Ramesh, S., & Vydeki, D. (2020). Recognition and classification of paddy leaf diseases using Optimized Deep Neural network with Jaya algorithm. *Information processing in agriculture*, *7*(2), 249-260.

Rehman, T. U., Mahmud, M. S., Chang, Y. K., Jin, J., & Shin, J. (2019). Current and future applications of statistical machine learning algorithms for agricultural machine vision systems. *Computers and electronics in agriculture*, *156*, 585-605.

Shafik, W., Tufail, A., Liyanage, C. D. S., & Apong, R. A. A. H. M. (2023). Using a novel convolutional neural network for plant pests detection and disease classification. *Journal of the Science of Food and Agriculture*, *103*(12), 5849-5861.

Sharma, A., Georgi, M., Tregubenko, M., Tselykh, A., & Tselykh, A. (2022). Enabling smart agriculture by implementing artificial intelligence and embedded sensing. *Computers & Industrial Engineering*, *165*, 107936.

Sharma, R. (2021, May). Artificial intelligence in agriculture: a review. In *2021 5th*

international conference on intelligent computing and control systems (ICICCS) (pp. 937-942). IEEE.

Singh, S., & Kasana, S. S. (2019). Estimation of soil properties from the EU spectral library using long short-term memory networks. *Geoderma Regional*, *18*, e00233.

Singh, D., Jain, N., Jain, P., Kayal, P., Kumawat, S., & Batra, N. (2020). PlantDoc: A dataset for visual plant disease detection. In *Proceedings of the 7th ACM IKDD CoDS and 25th COMAD* (pp. 249-253).

Sun, C., Li, Y., Song, Z., Liu, Q., Si, H., Yang, Y., & Cao, Q. (2025). Research on tomato disease image recognition method based on DeiT. *European Journal of Agronomy*, *162*, 127400.

Sun, J., Lai, Z., Di, L., Sun, Z., Tao, J., & Shen, Y. (2020). Multilevel deep learning network for county-level corn yield estimation in the us corn belt. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, *13*, 5048-5060.

Sethy, P. K., Barpanda, N. K., Rath, A. K., & Behera, S. K. (2020). Deep feature based rice leaf disease identification using support vector machine. *Computers and Electronics in Agriculture*, *175*, 105527.

Talaviya, T., Shah, D., Patel, N., Yagnik, H., & Shah, M. (2020). Implementation of artificial intelligence in agriculture for optimisation of irrigation and application of pesticides and herbicides. *Artificial Intelligence in Agriculture*, *4*, 58-73.

Tóth, G., Jones, A., & Montanarella, L. (2013). The LUCAS topsoil database and derived information on the regional variability of cropland topsoil properties in the European Union. *Environmental monitoring and assessment*, *185*, 7409-7425.

TU, K. L., LI, L. J., YANG, L. M., WANG, J. H., & Qun, S. U. N. (2018). Selection for high quality pepper seeds by machine vision and classifiers. *Journal of Integrative Agriculture*, *17*(9), 1999-2006.

Villacrés, J., Viscaino, M., Delpiano, J., Vougioukas, S., & Cheein, F. A. (2023). Apple orchard production estimation using deep learning strategies: A comparison of tracking-by-detection algorithms. *Computers and Electronics in Agriculture*, *204*, 107513.

Wibowo, T. T., Hidayat, S. S., & Suharjono, A. (2024, August). Dataset Amount and Augmentation Process Effect for CNN Rice Seed Classification. In *2024 11th International Conference on Information Technology, Computer, and Electrical Engineering (ICITACEE)* (pp. 48-53). IEEE.

Zhong, L., Guo, X., Xu, Z., & Ding, M. (2021). Soil properties: Their prediction and feature extraction from the LUCAS spectral library using deep convolutional neural networks. *Geoderma*, *402*, 115366.