Indonesian Journal of Artificial Intelligence and Data Mining (IJAIDM)
Vol. 9, No. 1, March 2026, pp. 114 – 129
p-ISSN: 2614-3372 | e-ISSN: 2614-6150

Application of the Categorical Boosting Model in Classifying Diseases of Tomato Leaves

[1*]Fitria Rahmah, [2]Selvi Annisa, [3]Dewi Anggraini
[1,2,3]Department of Statistics, Lambung Mangkurat University, Indonesia
Email: [1]fitriarahmahh@gmail.com, [2]selvi.annisa@ulm.ac.id, [3]dewi.anggraini@ulm.ac.id

Article Info

Article history:
Received Dec 12th, 2025
Revised Feb 16th, 2026
Accepted Mar 14th, 2026

Keyword:
Categorical Boosting
Early Blight
Late Blight
Tomato Leaf
VGG16 Architecture

ABSTRACT

Tomatoes are a strategic horticultural commodity whose productivity is often hampered by leaf diseases, particularly early blight and late blight. Manual identification through visual inspection is often inaccurate due to the similarity of symptoms between diseases. This study aims to improve the performance of tomato leaf disease classification using machine learning by overcoming the limitations of previous research by Ningsih et al., which focused solely on disease classes and did not include healthy leaf samples, thereby risking the model failing to recognize normal plant conditions. The proposed methodology integrates the VGG16 architecture as a feature extractor with the Categorical Boosting (CatBoost) algorithm as a classifier. The dataset sourced from Kaggle was cleaned and resized to 224x224 pixels, resulting in 3,285 images. The experimental results show that integrating VGG16 with CatBoost achieves good performance. The accuracy score achieved is 93.1%, while the F1 scores achieved are 90.2% (healthy leaves), 90.3% (early blight), and 98.6% (late blight). Compared to the research by Ningsih et al., this approach not only expands the scope of classification by including the healthy leaf class, but also shows better accuracy in identifying the health conditions of tomato plants.

Corresponding Author:
Fitria Rahmah,
Department of Statistics,
Lambung Mangkurat University,
Ahmad Yani Km 35.5, South Loktabat, South Banjarbaru, Banjarbaru, South Kalimantan, Indonesia.
Email: fitriarahmahh@gmail.com

DOI: http://dx.doi.org/10.24014/ijaidm.v9i1.38869

1. INTRODUCTION

Tomatoes (Lycopersicon Esculentum) are one of the species of flowering plants (Angiospermae). Tomatoes are one of the plants that have high nutritional benefits and are a good source of vitamin C. Tomatoes are widely consumed in various countries, including Indonesia, where they can be used as a basic ingredient in food preparation. According to the Food and Agriculture Organization of the United Nations Statistics (FAOSTAT) in 2023, the highest ranking country in Asia in tomato production is the People's Republic of China, with a total of 70,119,693.64 tons. Meanwhile, tomato production in Indonesia ranks 6th in Asia with a total production of 1,143,787.67 tons.

Along with this development, in order to improve the quality of tomato production, tools and requirements that support tomato plant growth are needed, such as the use of quality seeds, good and stable planting media, an adequate irrigation system, the use of appropriate pesticides, the use of good fertilizers, and post-harvest improvements [1]. The quality of tomato production can also be seen in the health of the growing tomato plants, specifically in the fruit, stems, and leaves. Tomato leaves are one part of the tomato plant that often shows symptoms of disease.

Figure 1a shows healthy tomato leaves with uniform green color without spots, wilting, or discoloration. In contrast, Figures 1b and 1c show leaves infected with early blight and late blight, caused by Alternaria solani and Phytophthora infestans, respectively. Late blight is characterized by spreading wet spots

that turn grayish brown, while early blight appears as dry brown circular or oval spots [2]. Both cause leaf drop, which can potentially spread to tomato fruits, reducing the quality of the tomatoes to be harvested.

Figure 1. Type of Tomato Leaf Diseases

Identifying the type of disease on tomato leaves is difficult to do through visual observation alone. Therefore, a machine learning-based image classification method is needed to help detect diseases accurately. Among various machine learning strategies, ensemble learning has gained significant attention due to its ability to improve predictive performance compared to single models. One effective ensemble approach is boosting, which sequentially combines multiple prediction models to reduce bias and enhance overall accuracy. A well-known boosting technique is Categorical Boosting (CatBoost), which offers advantages such as reducing overfitting and effectively handling high-dimensional feature data, enabling the model to achieve high prediction accuracy [3].

Research on image classification has been widely conducted using various methodological approaches and model architectures. The Graph Cut Segmentation method has been applied to potato leaves [4], while other studies have combined EfficientNet-B2-based feature extraction with the CatBoost model for peanut disease classification [5]. In addition, a transfer learning approach using the VGG-16 architecture has demonstrated superior performance compared to other architectures in tomato leaf disease classification [6]. However, these studies have not specifically explored the integration of VGG-16 as a feature extractor with CatBoost as a classifier for tomato leaf disease classification that includes healthy leaf classes .

In a different domain, a CNN–CatBoost combination has been utilized to detect vehicle license plates using camera sensors, and the proposed model outperformed several comparative approaches [7]. Similarly, a CNN–CatBoost framework has been applied for kidney disease classification, achieving an accuracy of 98.37% [8]. However, in the context of tomato leaf disease classification, a web-based Convolutional Neural Network (CNN) system has been developed, focusing only on early blight and late blight [9]. Although the system achieved good performance, it did not include healthy leaves during training. This limitation may lead to classification bias, where healthy tomato leaves are misclassified as diseased, since the model was not trained to recognize healthy leaf characteristics.

This issue is crucial in the context of applying plant disease classification systems in the real world. In agricultural practice, most of the leaves observed by farmers are not always infected with disease. Therefore, a classification system that only distinguishes between two types of disease without considering healthy leaves can reduce the reliability of the system, increase diagnostic errors, and potentially cause losses due to unnecessary disease control measures.

To address this research gap, this study included healthy leaves as part of the classification process. This study proposes the use of VGG-16-based transfer learning as a feature extractor capable of capturing the visual characteristics of leaves in depth, then combined with CatBoost as a classification method that excels in handling high-dimensional features and nonlinear relationships between features With this approach, it is expected that the classification system will not only be able to distinguish between early blight and late blight, but also accurately recognize healthy leaves, thereby improving the reliability, accuracy, and relevance of the system in real-world implementation in the field of precision agriculture.

Based on the description above, this study will analyze the classification of tomato leaves (healthy, early blight-exposed, and late blight-exposed) using the CatBoost model. The aim is to assess the CatBoost model's ability to handle image data by examining its accuracy.

2. RESEARCH METHOD

The research workflow is shown in Figure 2.

2.1 Theoretical Base

2.1.1 Preprocessing Data

Data preprocessing is the initial stage in developing machine learning models, including in the context of detecting plant diseases such as early blight and late blight on tomato leaves. Data preprocessing in this study comprises several steps: data cleaning, resizing or altering image pixels, and extracting image features.

1. Data Cleaning
   Data cleaning is an important part of preprocessing, including noise removal, handling missing data, and correcting inconsistencies. In this process, blurred image checking is assisted by the Laplacian Variance method, which is an effective method for determining the level of blurriness in an image based on its variance value [10].

2. Resize
   Resize is a process that changes the resolution or size of an image to be larger or smaller than the previous image size. In this study, images were resized from 256×256 to 224×224 pixels. This was done based on the feature extraction process which required images with 224×224 pixels as default input data [11].

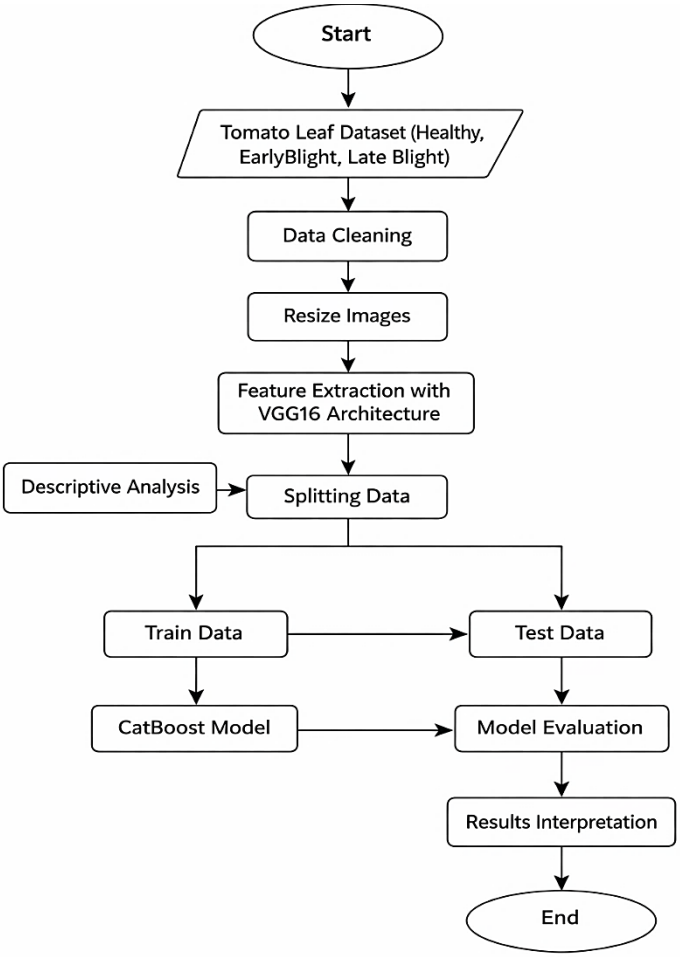

Figure 2. Research Workflow

2.1.2 Feature Extraction

Feature extraction is the process of extracting features from an image [12]. This study uses the VGG16 architecture, a convolutional neural network with multi-level convolutional layers and 3×3 filters, along with pooling to simplify data representation. The image is adjusted to a size of 224×224 pixels to match the pre-trained model and optimize accuracy [13]. The VGG16 architecture as image feature extraction can be done with the following processes.

1. Convolution Layer
   In the convolutional layer, the process extracts important features, such as edges and boundaries, from the input image. This process uses small boxes (filters) that move across the large image, examining small parts of the image one by one. The convolution operation is calculated using Equation 1.

$$o(i,j) = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} k(m,n) \times h(i-m, j-n) \quad (1)$$

Where:

$o(i,j)$	: Output value at position (i,j)
$k(m,n)$	: Filter element at position (m,n)
$h(i-m,j-n)$	: Element at position (i,j) that adjust the filter position (m,n)
M	: Number of rows in the filter
N	: Number of columns in the filter

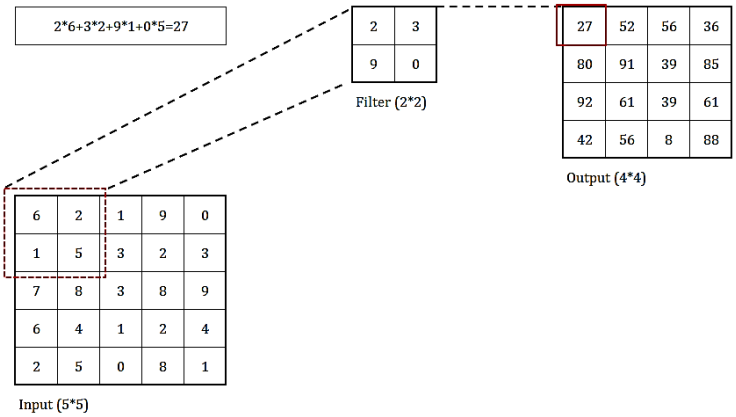

Figure 3. Convolution Operations

Figure 3 shows the convolution process on a 5×5 pixel image, where each filter calculates the output value by multiplying the image pixels and filter weights to extract features such as edges, textures, or specific shapes [14]. The VGG16 architecture generally uses the Rectified Linear Unit (ReLU) activation function to avoid negative output values [15]. The ReLU function is defined in Equation 2.

$$f(z) = \max(0, z)$$

$$f(z) = \begin{cases} z \text{ jika } z \geq 0 \\ 0 \text{ jika } z < 0 \end{cases} \quad (2)$$

Where z is the value produced by the convolution process with padding.

2. Pooling Layer

The VGG16 architecture uses max pooling, which is the process of taking the maximum value from the filter coverage area. The parameter used is stride, where a small stride value produces a larger output, while a large stride value produces a smaller output.

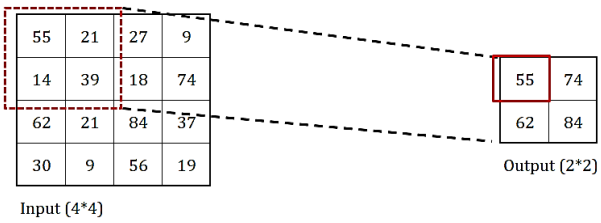

Figure 4. Max Pooling Process with Stride 2

If the image contains interference or noise, max pooling helps reduce it, thereby improving recognition quality. Max pooling on a 2×2 matrix can be calculated using Equation 3.

$$P_{i,j} = \max\left(C_{(0,0)}, C_{(0,1)}, C_{(1,0)}, C_{(1,1)}\right) \quad (3)$$

Where:

$P_{i,j}$	: Max pooling element at position (i,j)
$C_{(0,0)}$	: Input element at position (0,0)

2.1.3 Boosting Concepts

Boosting is a machine learning method that combines several simple models of the same type (such as Decision Trees) gradually and sequentially. Each new model is built to account for the errors of the previous model, so that the next model focuses more on data that was previously mispredicted. In this way, the resulting models complement each other and become "specialists" in difficult parts of the data. In addition, in boosting, each model does not have the same weight; its contribution is determined based on its performance. This approach allows the system to gradually improve accuracy by correcting the weaknesses of the previous model [16].

Some popular boosting algorithms include AdaBoost, Gradient Boosting, XGBoost, and LightGBM. These algorithms are widely used because they can produce high accuracy, especially with tabular data. Along with the development of these models, a boosting approach has emerged to address weaknesses in handling categorical features and potential overfitting during training. Categorical Boosting models have advantages such as avoiding overfitting. In addition, Categorical Boosting not only provides faster training speeds compared to other boosting algorithms, but also has the ability to manage data with many features that enable the model to produce high prediction accuracy [3].

2.1.4 Categorical Boosting

CatBoost is a boosting algorithm developed by Yandex and included in Gradient Boosting on Decision Tree (GBDT). This algorithm is effective in handling categorical data and works faster than other methods. CatBoost builds models sequentially with a focus on data that is difficult to classify, using parameters such as the number of iterations, learning rate, and depth to improve accuracy [17]. The CatBoost algorithm formula is shown in Equation 4 [18].

$$\hat{y} = \sum_{t=1}^{T} \alpha_t f_t(x) \tag{4}$$

Where:

$\hat{y}$: Prediction score
T : Number of iteration (tree)
α_t : Weight of the t-th tree

The CatBoost algorithm is effective for handling categorical features and runs faster than other algorithms. This algorithm improves prediction accuracy by combining weak models into a strong model [19]. The illustration of how GBDT works in CatBoost is as Figure 5.

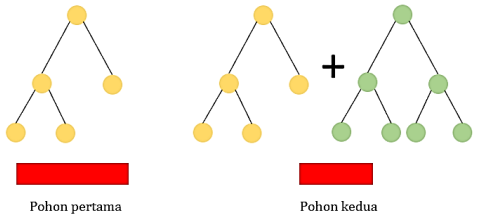

Figure 5. First and Second Tree Iterations in CatBoost

In the first iteration (Figure 5), the algorithm builds an initial tree to reduce the training error (residual error), as indicated by the red color. Next, in the second iteration, the algorithm builds an additional tree to correct the errors from the previous tree. This process is repeated until an optimal final model is formed, as shown in Figure 6.

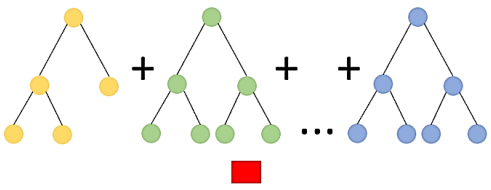

Figure 6. N-th Tree Iterations

This algorithm is used in ensemble models with iterative processes that are performed repeatedly. In the first iteration, the algorithm builds an initial tree to reduce the initial prediction error, which is generally still quite large. Instead of forming a large tree from the outset, this approach is more effective by adding trees gradually. In subsequent iterations, the algorithm continues to correct errors from previous iterations (Figure 7) [20].

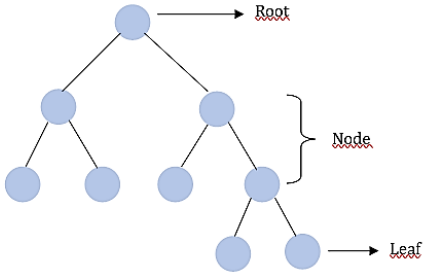

Figure 7. Parts of a Tree

The Categorical Boosting model uses a decision tree as its base model, consisting of three main parts: the root, nodes, and leaves. The root is the highest level that contains the most influential attributes [21]. Internal nodes are located in the middle of the tree with one incoming branch and several outgoing branches. Meanwhile, leaf nodes are endpoints without outgoing branches that show classification results or data labels [22].

2.1.5 Evaluating Model

The modeling results are evaluated based on processed training data. The model will produce predictions as category labels, such as "1 or 0" or "yes or no." To measure model performance, several evaluation techniques are used, including a confusion matrix, overall accuracy, and accuracy per class, to identify the level of error and accuracy of the model [23].

1. Confusion Matrix

   Confusion matrix is used to see the performance of the model from the evaluation analysis results [24]. The confusion matrix for multi-class data is as Table 1.

Table 1. Confusion Matrix

Actual	Prediction		
	Healthy	Early Blight	Late Blight
Healthy	a	b	c
Early Blight	d	e	f
Late Blight	g	h	i

   In multiclass classification, the true positive (TP), true negative (TN), false positive (FP), and false negative (FN) values are not obtained directly as in binary classification. These values are obtained by assuming one class as the "positive" class and the other classes as the "negative" class. For example, to assess the classification method's performance in class 1, class 1 is the 'positive' class, while classes 2 and 3 are the "negative" classes.

2. Accuracy

   Accuracy is a metric used when the target variable categories in the data are balanced. In addition, this metric is the simplest classification metric to use and is considered to be able to determine correct predictions with the total number of predictions found [25]. The following is formulation 5 of the accuracy metric.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{FP} + \text{FN} + \text{TN}} \times 100\% \quad (5)$$

2.2 Dataset

2.2.1 Healthy Leaves

Tomato leaves are an important indicator in assessing the overall health of a plant. Healthy leaves are generally dark green and shiny, indicating a high chlorophyll content, which plays an important role in photosynthesis. In addition to color, texture is also an indicator of plant condition; healthy leaves tend to be

thick, strong, and slightly rough, while thin and brittle leaves may indicate stress or health problems [27]. Healthy leaves should be free of spots, rust, powdery mildew, or damage from pests such as bites or holes.

2.2.2 Early Blight

Early blight is a leaf spot disease in tomatoes caused by the fungus Alternaria solani [28]. Early symptoms are characterized by round brown to black spots on the leaves that can merge and cause leaf rot, defoliation, and the falling of unripe fruit. Symptoms of early blight on tomato leaves are characterized by the appearance of small brown spots that later develop into round or irregular lesions with a distinctive concentric ring pattern. This disease can develop rapidly in favorable environmental conditions. The optimal temperature for early blight development is between 24-29°C with high relative humidity. The impact of early blight on tomato plants can be very detrimental. Early blight can cause a yield reduction of up to 79% if not properly controlled [29].

2.2.3 Late Blight

Late blight or leaf blight in tomato plants is caused by the fungus Phytophthora infestans, which is particularly destructive in humid environments and can cause total crop failure in a short period of time [30]. Phytophthora infestans has a complex life cycle and spreads rapidly through spores, infecting leaves, stems, and fruits. The main symptom is dark green or brown spots that develop into necrotic lesions with pale yellow edges [31].

3. RESULTS AND ANALYSIS

3.1 Preprocessing Data

Before performing descriptive analysis and classification of tomato leaf diseases using the Categorical Boosting model, the data was first cleaned through a preprocessing process. This stage included removing blurry images and adjusting image sizes to meet the VGG16 feature extraction requirements. Of the total 3300 images, 15 were detected as blurry using the Laplacian Variance method with a blur threshold of 100, comprising 3 Early Blight and 12 Late Blight images. Blurry images were removed to prevent classification errors and improve model performance. Some examples of blurry images are shown in the following illustration, can view Figure 8.

Figure 8. Example of Blurry Images

From 3,285 clearly detected image data, a ratio check was performed before resizing to maintain display consistency and avoid distortion. Since all images had a 1:1 aspect ratio, the resizing was performed directly from 256×256 to 224×224 using interpolation. This process maintained the aspect ratio because the initial and final sizes were square. Technically, resizing was performed using functions from image processing libraries such as OpenCV (cv2.resize) or Keras (ImageDataGenerator or load_img with the target_size parameter).

Figure 9. Before and After Resizing Image

Figure 9 shows no significant difference between images with 256×256 and 224×224 pixels. Although they look the same visually, each pixel stores a numerical value that represents color intensity or RGB color composition. A matrix with a pixel size of 256×256 tends to store more information than a matrix with a pixel size of 224×224. However, the use of a 224×224 image matrix size is still required as a standard input for the

VGG16 architecture [32]. In clean image data, each image will be converted into Red, Green, and Blue (RGB) color channels. Each color channel can be seen in Figure 10.

Figure 10. Red, Green, and Blue color channels

Based on Figure 10, the leaf image has been converted into Red, Green, and Blue color channels. Each color channel is represented as a 224×224 two-dimensional matrix.

3.2 Descriptive Analysis

The descriptive analysis in this study provides information about the tomato leaf disease dataset, including the proportion of tomato leaf disease and the image pixel size.

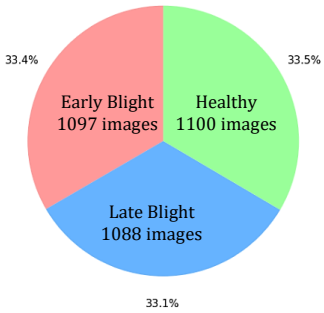

Figure 11. Proportion of Healthy Leaves , Early Blight, and Late Blight with Pie Chart

Figure 11 shows a comparison of three categories of tomato leaves. The healthy leaf class has 1,100 images (33.5%), early blight has 1,097 images (33.4%), and late blight has 1,088 images (33.1%). The balanced proportions between classes indicate that no data imbalance handling is required.

3.3 VGG16 Architecture as Feature Extraction

After the image is converted into a 224×224×3 array, with 224 as the pixel dimension and 3 as the color channels (Red, Green, Blue), the data is fed into the VGG16 input layer. The image is then processed by the first convolution layer, which uses 64 3×3 filters, producing an output with dimensions 224×224×64. Each filter has its own weights, which are initially initialized randomly and updated during the training process. The convolution operation in the first layer with 3×3 filters is described as Figure 12.

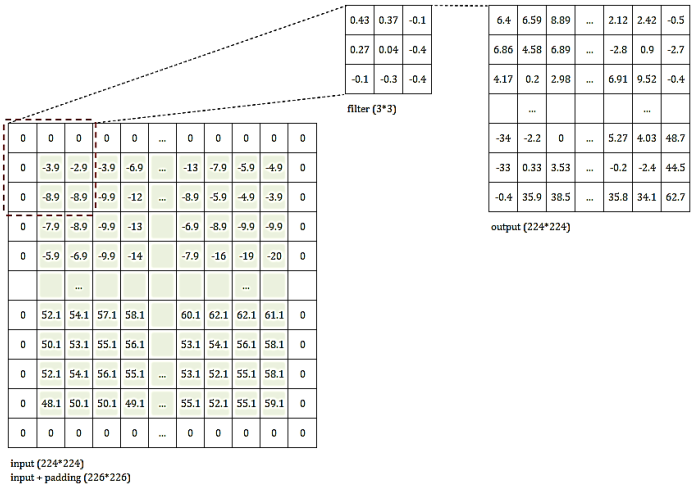

Figure 12. Operation on First Convolution Layer with Filter on Red Channel

In Figure 12, the calculation is performed by taking the first 3×3 matrix dimension and multiplying it by the 3×3 filter on the Red channel. Suppose that the 3×3 input matrix is marked in the figure with a 3x3 filter, then the calculation based on Equation 1 is as follows.

$$o_{R_{1,1}} = ((0 \times 0.43) + (0 \times 0.37) + (0 \times (-0.1)) + (0 \times 0.27) + ((-3.9) \times 0.04) + ((-2.9) \times (-0.4)) + (0 \times (-0.1)) + ((-8.9) \times (-0.3)) + ((-8.9) \times (-0.4))$$
$$= 0 + 0 + 0 + 0 - 0.156 + 0{,}116 + 0 + 2.67 + 3.56$$
$$= 6.4$$

The calculation was performed up to $o_{R_{224,224}}$

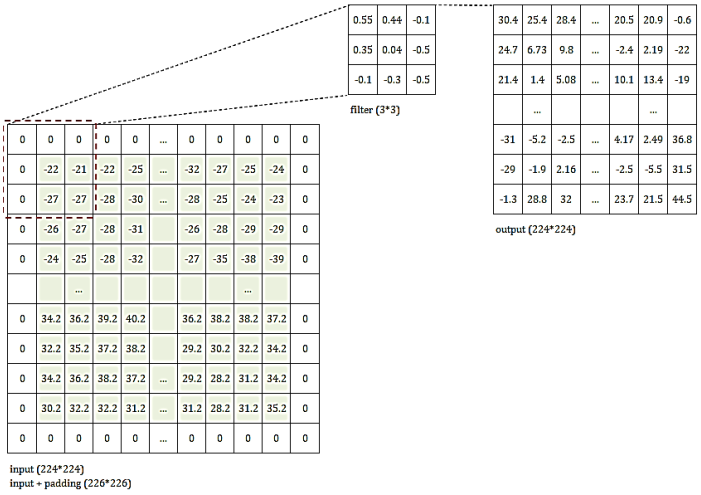

Figure 13. Operation on First Convolution Layer with Filter on Green Channel

In Figure 13, the calculation is performed by taking the first 3×3 matrix dimension and multiplying it by the 3×3 filter on the Green channel. Suppose that the calculation of the 3×3 input matrix marked in the figure with a 3x3 filter, then the calculation based on Equation 1 is as follows.

$$o_{G_{1,1}} = ((0 \times 0.55) + (0 \times 0.44) + (0 \times (-0.1)) + (0 \times 0.35) + ((-22) \times 0.04) + ((-21) \times (-0.5)) + (0 \times (-0.1)) + ((-27) \times (-0.3)) + ((-27) \times (-0.5))$$
$$= 0 + 0 + 0 + 0 - 0.88 + 10.5 + 0 + 8.1 + 13.5$$
$$= 30.4$$

The calculation was performed up to $o_{G_{224,224}}$

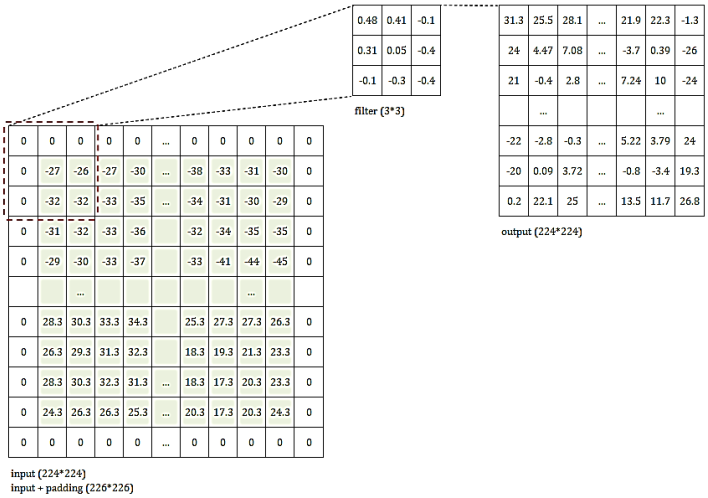

Figure 14. Operation on First Convolution Layer with Filter on Blue Channel

In Figure 14, the calculation is performed by taking the first 3×3 matrix dimension and multiplying it by the 3×3 filter on the Red channel. Suppose that the calculation of the 3×3 input matrix marked in the figure with a 3x3 filter, then the calculation based on Equation 1 is as follows.

$$o_{B_{1,1}} = ((0 \times 0.48) + (0 \times 0.41) + \big(0 \times (-0.1)\big) + (0 \times 0.31) + \big((-27) \times 0.05\big) + \big((-26) \times (-0.4)\big) + \big(0 \times (-0.1)\big) + \big((-32) \times (-0.3)\big) + \big((-32) \times (-0.4)\big)$$
$$= 0 + 0 + 0 + 0 - 1.35 + 10.4 + 0 + 9.6 + 12.8$$
$$= 31.3$$

The calculation was performed up to $o_{B_{224,224}}$

In Figure 14, the calculation is performed by taking the first 3×3 matrix dimension and multiplying it by the 3×3 filter on the Red channel. Suppose that the calculation of the 3×3 input matrix marked in the figure with a 3x3 filter, then the calculation based on Equation 1 is as follows.

error = 0.73429835

R

6.4	6.59	8.89	...	2.12	2.42	-0.5
6.86	4.58	6.89	...	-2.8	0.9	-2.7
4.17	0.2	2.98	...	6.91	9.52	-0.4
	...				...	
-34	-2.2	0	...	5.27	4.03	48.7
-33	0.33	3.53	...	-0.2	-2.4	44.5
-0.4	35.9	38.5	...	35.8	34.1	62.7

+

G

30.4	25.4	28.4	...	20.5	20.9	-0.61
24.7	6.73	9.8	...	-2.43	2.19	-22
21.4	1.4	5.08	...	10.1	13.4	-19.2
	...				...	
-31.4	-5.25	-2.54	...	4.17	2.49	36.8
-29.3	-1.88	2.16	...	-2.46	-5.53	31.5
-1.27	28.8	32	...	23.7	21.5	44.5

+

B

31.3	25.5	28.1	...	21.9	22.3	-1.33
24	4.47	7.08	...	-3.73	0.39	-25.9
21	-0.35	2.8	...	7.24	10	-23.5
	...				...	
-21.7	-2.79	-0.3	...	5.22	3.79	24
-19.9	0.09	3.72	...	-0.82	-3.4	19.3
0.2	22.1	25	...	13.5	11.7	26.8

Figure 15. ReLu on First Layer

As can be seen in Figure 15, the pixel values marked with red lines in each color channel will be summed. Next, Equation 2 is used to compute the first-layer output as follows.
Suppose the calculation in $z_{1,1}$

$$f(z) = \max(0; z)$$
$$= \max\big(0; (6.4 + 30.4 + 31.3 + 0.73)\big)$$
$$= \max(0\ ; 68.8)$$
$$= 68.8$$

The addition operation and application of the ReLU activation function are performed up to the value $z_{224,224}$. Thus, the first output layer, as shown in Figure 16, is obtained from the calculations.

68.8	58.2	66.1	...	45.3	46.4	0
56.2	16.5	24.5	...	0	4.21	0
47.2	1.97	11.6	...	25	33.7	0
	...				...	
0	0	0	...	15.4	11	110
0	0	10.2	...	0	0	96
0	87.5	96.2	...	73.8	68.1	135

Figure 16. Output on the First Layer

After passing the first convolution layer, the image will be processed in the second convolution layer using 64 filters, so that the output dimensions remain the same as the first layer. Then, dimension reduction is performed using a max pooling layer with a 2×2 filter, which produces an output size of 112×112×64. The matrix in the max pooling layer of the first layer is as Figure 17.

Max pooling is performed by taking the highest value in each 2×2 submatrix. Based on equation 3, $P_{1,1}$ can be calculated as follows.

$$P_{1,1} = \max(68.8, 58.2, 56.2, 16.5) = 68.8$$

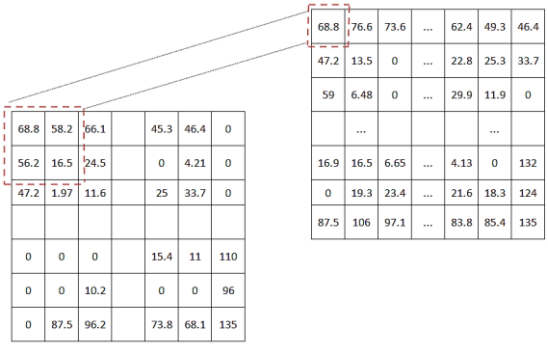

Figure 17. Max Pooling on the First Layer

After passing through the first max pooling layer, the image is processed by two convolution layers with 128 filters, producing an output size of 112×112×128. The process continues with a second max pooling layer (2×2), reducing the size to 56×56×128. Next, the image is processed by three convolutional layers with 256 filters, producing an output of 56×56×256. This size is then reduced through a third max pooling to 28×28×256. The next three convolutional layers use 512 filters and produce an output of 28×28×512. A fourth max pooling reduces the dimensions again to 14×14×512.

Then, the last three convolutional layers (each with 512 filters) produce a fixed output of 14×14×512. Finally, the fifth max pooling produces an output of 7×7×512, which is then flattened into a 1-dimensional vector for the classification stage, as shown in Table 2.

Table 2. Data Tabulation after Using VGG16 Architecture on Image Data

Feature_1	Feature _2	Feature _3	...	Feature _25086	Feature _25087	Feature _25088
0	0	0	...	0	0.9726009	0

Based on the steps in the VGG16 architecture, there are 18 hidden layers consisting of 13 hidden layers in the Convolution Layer and 5 hidden layers in the Maxpooling Layer, as shown in Appendix 5.

3.4 Training Categorical Boosting Model

The Categorical Boosting model trains the data extracted from VGG16 by gradually building a series of trees, where each new tree learns from the errors of the previous tree. In this process, the tree shown is the tree in the first iteration. The image of the tree in the first iteration is as Figure 18.

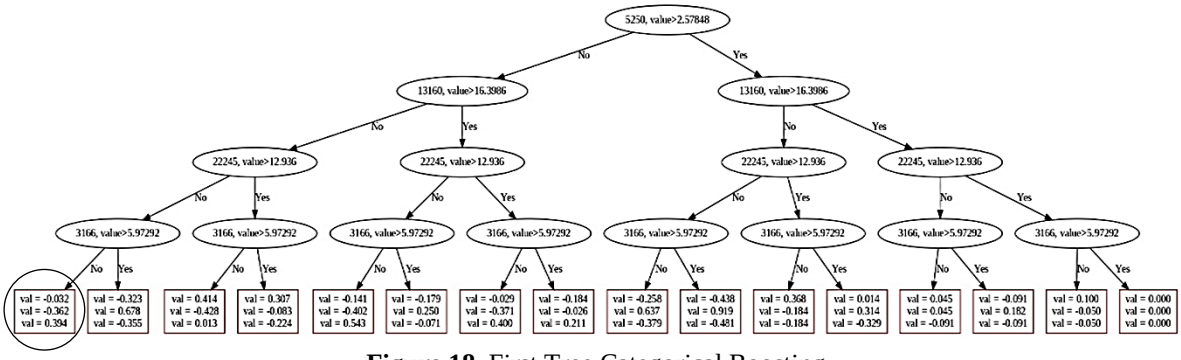

Figure 18. First Tree Categorical Boosting

The decision tree in the Categorical Boosting model consists of three main components: root, node, and leaf. The root is the top part that contains the most influential features in the classification process. In the first tree, feature 5250 is placed at the root with a threshold value of 2.57848, which is determined through a quantization process using the GreedyLogSum method according to the official CatBoost documentation.

The model divides the data based on feature values against the threshold. If the feature value is ≤ 2.57848, the data is directed to the left branch (No); if it is higher, it is directed to the right branch (Yes). This process continues with the next features, such as feature 13160, until it reaches the tree depth determined by the model parameters. In the leaf section, the model generates class predictions based on the val value, which represents three classes: Early Blight, Healthy, and Late Blight. A positive val value indicates support for a

class, while a negative value indicates rejection. For example, in one leaf, the val values = -0.032, -0.362, and 0.394, indicating a prediction of Late Blight for the image.

The tree formation process continues until the 600th iteration, which is the optimal number based on the highest accuracy from the parameter tuning results. Tuning is done by exploring three parameter combinations, and the best configuration is obtained in the CatBoost model with 600 iterations, a learning rate of 0.5, and a depth of 4. Figure 18 shows the tree at the 600th iteration, where the val values at the leaf are generally small or even zero. This indicates that classification errors in the training data have been minimized by the previous trees, and that the model has approached optimal results, with the last tree providing final adjustments to the predictions.

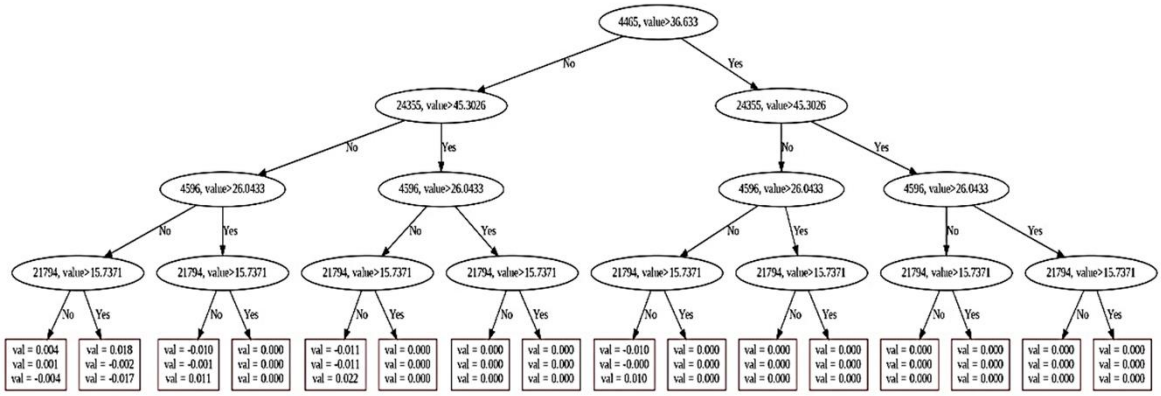

Figure 19. Last Tree Categorical Boosting

3.5 Model Evaluation on Test Data

3.5.1 Confusion Matrix

After the model was trained, the next step is to evaluate the results of the tomato leaf category classification using the Confusion Matrix shown in Table 3.

Table 3. Confusion Matrix of Healthy, Early Bligh, and Late Blight Tomato Leaves

		Predict		
		Early Blight	Late Blight	Healthy
Actual	Early Blight	191	2	17
	Late Blight	0	221	1
	Healthy	22	3	200

The Confusion Matrix results show that Late Blight was most often predicted correctly, while Early Blight was most often predicted incorrectly. However, to evaluate the results, it is necessary to compare them using several calculation metrics.

3.5.2 Accuracy

Based on the confusion matrix, the number of classes correctly predicted is 191+221+200=612 out of 657 test data. Thus, the accuracy obtained is:

$$\text{Accuracy} = \frac{612}{657} \times 100\% = 93{,}1\% = 0.931$$

Overall, the model correctly classified 93.1% of the test data (612 out of 657 images). A total of 6.9% (45 images) failed to be classified correctly.

3.5.3 Precision

Precision indicates the model's ability to correctly predict a class relative to all data predicted as that class. The precision calculation for each class is as follows:

$$\text{Precision}_{\text{Healthy}} = \frac{200}{218} \times 100\% = 91{,}7\% = 0.917$$

$$\text{Precision}_{\text{Early Blight}} = \frac{191}{213} \times 100\% = 89{,}7\% = 0.897$$

$$\text{Precision}_{\text{Late Blight}} = \frac{221}{226} \times 100\% = 97{,}8\% = 0.978$$

Based on these results, the Late Blight class had the highest precision value of 97.8%, indicating that most of the data predicted as Late Blight was correct. This value was higher than the Early Blight class (89.7%) and the Healthy class (91.7%).

3.5.4 Recall

The recall metric is also used to measure the model's ability to correctly identify all actual data in a class. Recall is calculated based on the Confusion Matrix using the following:

$$\text{Recall}_{\text{Healthy}} = \frac{200}{225} \times 100\% = 88.8\% = 0.888$$

$$\text{Recall}_{\text{Early Blight}} = \frac{191}{210} \times 100\% = 90.9\% = 0.909$$

$$\text{Recall}_{\text{Late Blight}} = \frac{221}{225} \times 100\% = 99.5\% = 0.995$$

The results show that the Late Blight class again achieved the highest recall of 99.5%, indicating that the model successfully recognized almost all Late Blight data. This recall value is higher than the Early Blight class at 90.9% and the Healthy class at 88.8%.

3.5.5 F1-Score

Based on the precision and recall values obtained, the F1-Score is calculated as the harmonic mean, balancing precision and recall. The F1-Score calculation results for each class are as follows:

$$\text{F1} - \text{Score}_{\text{Healthy}} = 2 \times \frac{0.917 \times 0.888}{0.917 + 0.888} = 90.2\% = 0.902$$

$$\text{F1} - \text{Score}_{\text{Early Blight}} = 2 \times \frac{0.897 \times 0.909}{0.897 + 0.909} = 90.3\% = 0.903$$

$$\text{F1} - \text{Score}_{\text{Late Blight}} = 2 \times \frac{0.978 \times 0.995}{0.978 + 0.995} = 98.6\% = 0.986$$

These results show that the model performs very well at classifying all tomato leaf disease classes, with F1-scores above 90% for each class. The Late Blight class has the highest F1-Score, indicating that the model is very consistent and reliable in accurately recognizing and predicting this class. The final classification report is presented in Table 4.

Table 4. Classification Report

Class	Precision	Recall	F1-Score
Healthy	0.917	0.888	0.902
Early Blight	0.897	0.909	0.903
Late Blight	0.978	0.995	0.986

Based on the results of the research, there was an error in the classification of the early blight class, which was predicted to be healthy leaves. This error was caused by the visual characteristics of early blight in the early stages, which were still mild, where the size of the spots was relatively small and the concentric circle pattern had not yet formed clearly. As a result, the visual features of the disease are difficult to distinguish from the characteristics of healthy leaves, limiting the model's ability to extract representative features. Additionally, the dominance of leaf areas that are still green causes the global image features to resemble healthy leaves more than disease characteristics. Conversely, healthy leaves predicted as early blight are leaves that exhibit a similar pattern due to aging (caused by protein/chlorophyll degradation), resulting in a brown color change [33], leading the model to predict that the leaves show indications of early blight. Leaf color changes can also occur due to nutrient deficiency, environmental stress, or minor mechanical damage. These conditions can visually resemble early blight symptoms, increasing the potential for misclassification, especially in cases

where healthy leaves are predicted as early blight. Meanwhile, the late blight class achieved the highest accuracy compared to other classes. This is due to the more distinct and consistent visual characteristics of late blight, such as irregular dark spots with a wet appearance and relatively large infected areas. Thus, the model more easily recognizes images, and the misclassification rate for the late blight class is lower.

4. CONCLUSION

The dataset consists of 3300 images of tomato leaves divided evenly into three categories: Healthy, Early Blight, and Late Blight (1100 images each). After preprocessing to remove 15 blurry images, 3285 clean images were obtained for further analysis. Descriptive analysis results show that the data distribution remains balanced, as visualized in the pie chart. The classification process using CatBoost was not performed directly on the raw images but rather via a feature-extraction stage using the VGG16 architecture. The VGG16 model converts 224×224-pixel images into simpler, more informative feature representations. The resulting features were used as input for the CatBoost model with classification labels (Healthy, Early Blight, Late Blight). The CatBoost model was built on a Decision Tree as the base model and automatically handled numerical features via gradient boosting to improve accuracy and reduce classification errors. The model was built with 600 trees, each correcting the errors of the previous tree. After training was complete, the model was tested using test data to evaluate performance based on accuracy, precision, recall, and F1-score.

The test results show that the CatBoost model achieves an overall accuracy of 93% on the test data. In addition, the per-class evaluation metric values show consistent and high performance, with recall values of 88.8%, 90.9%, and 99.5%, precision values of 91.7%, 89.7%, and 97.8%, and F1-scores of 90.2%, 90.3%, and 98.6% for the healthy tomato leaf, early blight, and late blight classes, respectively. These results show that the model achieves high accuracy and balanced performance in classifying the three tomato leaf categories. Compared with previous research by Ningsih et al. (2022), which reported 80% accuracy and 85% recall on the test data, the approach proposed in this study shows improved classification performance.

As a recommendation for further research, it is suggested to enrich the dataset by adding more data or applying data augmentation techniques to improve the model's generalization ability. In addition, the addition of disease classes or other conditions, such as “nutritional deficiencies”, can be considered to reduce the potential for classification errors due to visual symptom similarities. Further testing using field data with variations in lighting conditions, shooting angles, and more diverse backgrounds is also necessary to make the model more adaptive to real conditions in agricultural environments.

REFERENCES

[1] D. E. Kusumawati, L. E. Saputra e A. Amiroh, “Aplikasi Macam dan Dosis Pupuk Kandang Pada Tanaman Tomat (Lycopersicon Eskulentum),” *Agroradix,* vol. 5, nº 1, pp. 36-41, 2021.

[2] A. J. Rozaqi, A. Sunyoto e R. Arief, “Deteksi Penyakit pada Daun Kentang Menggunakan Pengolahan Citra dengan Metode Convolutional Neural Network,” *Creative Information Technology Journal*, vol. 8, nº 1, pp. 22-31, 2021.

[3] M. W. Ahmed, A. Sprigler, J. L. Emmert, R. N. Dilger, G. Chowdhary e M. Kamruzzaman, “Non-Destructive Detection of Pre-Incubated Chicken Egg Fertility Using Hyperspectral Imaging and Machine Learning,” *Jurnal Elsevier*, pp. 2772-3755, 2024.

[4] C. Hou, J. Zhuang, Y. Tang, Y. He, A. Miao, H. Huang e S. Luo, “Recognition of Early Blight and Late Blight Diseases on Potato Leaves Based on Graph Cut Segmentation,” *Journal of Agriculture and Food Research*, vol. 5, 2021.

[5] S. Rohman, E. Y. Puspaningrum e A. D. Rahajoe, “Perbandingan Model LightGBM dan CatBoost pada Klasifikasi Penyakit Daun Kacang Tanah Berbasis Ekstraksi Fitur EfficientNet-B2,” *Jurnal Ilmiah Ilmu Pendidikan*, vol. 9, nº 1, pp. 953-961, 2026.

[6] M. I. F. Rozi, N. O. Adiwijaya e D. I. Swasono, “Identification of VGG16, ResNet-50, and Inception-V3 Transfer Architecture Performance in Image Classification of Tomato Leaf Diseases,” *Jurnal Riset Rekayasa Elektro*, vol. 5, nº 2, pp. 145-154, 2023.

[7] S. Zhang, X. Lu e Z. Lu, “Improved CNN-based CatBoost model for license plate remote sensing image classification,” *Signal Processing*, vol. 213, 2023.

[8] N. Bhaskar, R. R. Borhade, S. Barekar, M. Bachute e V. Bairagi, “CNN-CatBoost Ensemble Deep Learning Model for Enhanced Disease Detection and Classification of Kidney Disease,” *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 34, nº 1, pp. 144-151, 2024.

[9] N. P. Ningsih, E. Suryadi, L. D. Bakti e B. Imran, “Klasifikasi Penyakit Early Blight dan Late Blight pada Tanaman Tomat Berdasarkan Citra Daun Menggunakan Metode CNN Berbasis Website,” *Jurnal Kecerdasan Buatan dan Teknologi Informasi (JKBTI)*, vol. 3, nº 1, pp. 27-35, 2022.

[10] C. W. So, E. L. H. Yuen, E. H. F. Leung e J. C. S. Pun, “Solar Image Quality Assessment: A Proof of Concept Using Variance of Laplacian Method and its Application to Optical Atmospheric Condition Monitoring,” *Publications of the Astronomical Society of the Pacific*, pp. 1-9, 2024.

[11] W. Hutamaputra, R. Y. Krisnabayu, M. Mawarni, F. A. Bachtiar e N. Yudistira, “Perbandingan Kinerja Convolutional Neural Network VGG16 dan ResNet34 pada Sistem Klasifikasi Sampah Botol,” Jurnal Teknologi dan Sistem Komputer, vol. 2, pp. 136-142, 2022.

[12] Sutarno, R. F. Abdullah e R. Passarella, "Identifikasi Tanaman Buah Berdasarkan Fitur Bentuk, Warna dan Tekstur Daun Berbasis Pengolahan CItra dan Learning Vector Quantization (LVQ)," *Prosiding Annual Research Seminar*, Palembang, 2017.
[13] R. Adhyaksa e B. Purnama, "Application of VGG16 in Automated Detection of Bone Fractures in X-Ray Images," *Jurnal Rekayasa Sistem dan Teknologi Informasi*, vol. 1, pp. 118-129, 2025.
[14] B. Raharjo, Deep Learning dengan Python, Semarang: Yayasan Prima Agus Teknik, 2022.
[15] J. Bernal, K. Kushibar, D. S. Aswaf, S. Valverde, A. Oliver, R. Marti e X. Llado, "Deep Convolutional Neural Networks For Brain Image Analysis on Magnetic Resonance Imaging: A Review," *Artificial Intelligence in Medicine*, vol. 95, pp. 64-81, 2019.
[16] I. H. Witten, E. Frank, M. A. Hall e C. J. Pal, Data Mining: Practical Machine Learning Tools and Techniques, vol. Iv, *Cambridge: Elsevier*, 2017.
[17] S. B. Jabeur, C. Gharib, S. M. Wali e W. B. Arfi, "CatBoost Model and Artificial Intelligence Techniques for Corporate Failure Prediction," *Technological Forecasting and Social Change*, vol. 166, 2021.
[18] M. A. Harriz, N. V. Akbariani, H. Setiyowati e H. Santoso, "Classifying Village Fund in West Java Indonesia Using CatBoost Algorithm," *Jurnal Indonesia : Manajemen Informatika dan Komunikasi*, vol. 4, nº 2, pp. 691-697, 2023.
[19] N. D. Syandika e W. Yustanti, "Deteksi Anomali Terhadap Pembatalan Transaksi Pada Platform Tiktok Shop dengan Algoritma Categorical Boosting (Catboost)," *Journal of Informatics and Computer Science (JINACS)*, vol. 5, nº 2, pp. 149-156, 2023.
[20] S. Barua, D. Gavandi, P. Sangle, L. Shinde e J. Ramteke, "Swindle: Predicting the Probability of Loan Defaults using CatBoost Algorithm," *International Conference on Computing Methodologies and Communication (ICCMC)*, Erode, 2021.
[21] A. H. Nasrullah, "Implementasi Algoritma Decision Tree untuk Klasifikasi Produk Laris," *Jurnal Ilmiah Ilmu Komputer*, vol. 7, no 2, pp. 45-51, 2021.
[22] P. Kasih, "Pemodelan Data Mining Decision Tree Dengan Classification Error Untuk Seleksi Calon Anggota Tim Paduan Suara," *Innovation in Research of Informatics (INNOVATICS)*, vol. 1, nº 2, pp. 63-69, 2019.
[23] Fadlisyah e Muhathir, "Performance Evaluation Of Variations Boosting Algorithms For Classifying Formalin FIsh From Photos," *Journal of Informatics and Telecommunication Engineering*, vol. 6, nº 2, pp. 621-624, 2023.
[24] Q. Meidianingsih, D. E. Wardani, E. Salsabila, L. Nafisah e A. N. Mutia, "Perbandingan Performa Metode Berbasis Support Vector Machine untuk Penanganan Klasifikasi Multi Kelas Tidak Seimbang," *Statistika*, vol. 23, nº 1, pp. 8-15, 2023.
[25] E. Agustin, A. Eviyanti e N. L. Azizah, "Deteksi Penyakit Epilepsi Melalui Sinyal EEG Menggunakan Metode DWT dan Extreme Gradient Boosting," *Jurnal Media Informatika Budidarma*, vol. 7, nº 1, pp. 117-127, 2023.
[26] A. C. Müller e S. Guido, Introduction to Machine Learning with Python, *O'Reilly Media, Inc*., 2016.
[27] R. R. Shamshiri, F. Kalantari, K. C. Ting, K. R. Thorp, I. A. Hameed, C. Weltzien e Z. M. Shad, "Advances in Greenhouse Automation and Controlled Environment Agriculture: A Transition to Plant Factories and Urban Agriculture," *International Journal of Agricultural and Biological Engineering*, vol. 1, pp. 1-22, 2018.
[28] F. Roy, G. Biroli e C. Cammarota, "Numerical Implementation of Dynamical Mean Field Theory for Disordered Systems: Application to the Lotka-Volterra Model of Ecosystems," *Journal of Physics A: Mathematical and Theoretical*, vol. 52, nº 48, 2019.
[29] R. Adhikari, M. Agostini, N. A. Ky, T. Araki, M. Archidiacono, M. Bahr e K. Zuber, "A White Paper on keV Sterile Neutrino Dark Matter," *Journal of Cosmology and Astroparticle Physics*, 2017.
[30] D. A. Lage, W. A. Marouelli e A. C. Café-Filho, "Management of Powdery Mildew and Behaviour of Late Blight Under Different Irrigation Configurations in Organic Tomato," *Crop Protection*, vol. 125, 2019.
[31] W. E. Fry e S. B. Goodwin, Re-emergence of Potato and Tomato Late Blight in the United States, vol. 81, USA: *The American Phytopathological Society*, 1997.
[32] K. Simonyan e A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *ICLR*, pp. 1-14, 2015.
[33] Y. Yu, S. Wang, W. Guo, M. Geng, Y. Sun, W. Li, G. Yao, D. Zhang, H. Zhang e K. Hu, "Hydrogen Peroxide Promotes Tomato Leaf Senescence by Regulating Antioxidant System and Hydrogen Sulfide Metabolism*," Plants,* vol. 13, nº 4, p. 475, 2024.
[34] M. Azhari, Z. Situmorang e R. Rosnelly, "Perbandingan Akurasi, Recall, dan Presisi Klasifikasi pada Algoritma C4.5, Random Forest, SVM dan Naive Bayes," *Jurnal Media Informatika Budidarma*, vol. 5, nº 2, pp. 641-651, 2021.

BIBLIOGRAPHY OF AUTHORS

Fitria Rahmah was born in Tabalong, Indonesia, on December 9, 2002. The author's a student in Statistics at Universitas Lambung Mangkurat. The author's academic interests include data science, data mining, and big data analytics. The author's has experience as a teaching assistant for Calculus II and has participated in the SATRIA DATA Credit Earning Program. Author has also been actively involved in academic organizations and university committees.
Correspondence: fitriarahmahhh@gmail.com.

Selvi Annisa is a lecturer in the Department of Statistics, Faculty of Mathematics and Natural Sciences, Universitas Lambung Mangkurat, South Kalimantan, Indonesia. She obtained her Master's degree from IPB University in 2019. The author's research interests include machine learning, deep learning, and statistical modeling. Currently, the author is actively conducting research and data analysis focused on wetland environments

Dewi Anggraini, S.Si., M.App.Sci., Ph.D. is a lecturer in the Mathematics and Statistics Study Programs, FMIPA, Lambung Mangkurat University (ULM), since 2005. She teaches Introduction to Statistics, Experimental Design, Quality Control Statistics, Survival Analysis, Longitudinal Data Analysis, and Research Methods. Her research interests focus on maternal and child health databases, surveillance and monitoring systems, and longitudinal data modeling.