JURNAL TEKNOLOGI DAN OPEN SOURCE
Vol. 6, No. 1, June 2023, pp. 31~40
e-ISSN: 2622-1659, accredited Four Grade by Kemenristekdikti, Decree No: 36/E/KPT/2019
DOI: 10.36378/jtos.v6i1.2883

Synthetic Minority Oversampling Technique for Efforts to Improve Imbalanced Data in Classification of Lettuce Plant Diseases

Nurliana Nasution[1], Feldiansyah [2],Ahmad Zamsuri [3], Mhd Arief Hasan[4]
[1,2,3,4]Program Studi Teknik Informatika, Universitas Lancang Kuning, Riau, Indonesia

Article Info

Article history:

Received 02 07, 2023
Revised 02 ,23 2023
Accepted 05 15, 2023

Keywords:

Classification
Disease
Machine Learning
Lettuce
Smote

ABSTRACT

In this study we classified lettuce plant diseases. These plant diseases are available in the form of images that have been converted in .csv format to be classified. These plant diseases are available in the form of images that have been converted in .csv format to be classified. Image These plant diseases have been divided into several classes or categories. Then we determine the features of each row and column of the dataset. Each line in the CSV file represents one image, and each column represents one feature Each line in the CSV file represents one image, and each column represents one feature. Then a label is made for each line in the CSV file, namely the class or category where the images are grouped. Thus, so that we get datasets that are ready to be processed with machine learning. However, in processing the dataset, we get imbalanced data. So we added the Synthetic Minority Over-sampling Technique (SMOTE) method to overcome the imbalance that occurs. So that the data can be classified using several algorithms to find the best accuracy.

Corresponding Author:

Nurliana Nasution
Program Studi Teknik Informatika
Universitas Lancang Kuning
Riau,Indonesia
Email: nurliananst@unilak.ac.id

1. Introduction

Lettuce or sla leaf (Lactuca sativa) is a vegetable plant that is commonly grown in temperate and tropical regions. The main use is as a salad. Lettuce is used in a variety of dishes, including soups, sandwiches and it can even be grilled. Celtuce (asparagus lettuce) is one type produced from the stems, which can be eaten raw or cooked. Through thousands of years of human use, it has acquired religious and therapeutic value in addition to its primary use as a vegetable. Pops of lettuce come in many shapes and textures, from dense heads of iceberg lettuce to squiggly, scalloped, frilly, or wrinkled leaves. The root system of lettuce plants consists of a main taproot and smaller tiller roots. Long, slender taproots and simple secondary roots occur in several varieties, mainly found in the United States and Western Europe. The Asian variant has a longer taproot and a more extensive secondary system.

Planting lettuce is not always successful without obstacles. Even though it looks easy at first glance, in reality it still often encounters obstacles.Apart from pests, what often happens is the disease itself.As for some lettuce plant diseases including: Soft Rot, This disease is caused by the bacterium Erwinia Carotovora. The attack starts from the edge of the leaf, then the leaves will turn brown and eventually wither.These bacteria are dangerous regardless of the location. Stem Rot. If lettuce plants are infected with this disease, the

most obvious thing to look at is the leaf stems that feel soft, different from usual, and slimy too. Generally, the attacking bacteria is a type of fungus called Rhizoctonia Solani. Leaf base rot, Felicularia Filamentosa is the bacterium that causes the disease. Which is generally attacked is the base of the leaf, when the harvest season arrives.

Classification is the process of predicting the class of an object based on its attributes. In machine learning, classification is often used to determine whether an item belongs to a certain class or not[1], [2]. For example, classification is used to determine whether an email is spam or not spam, or to determine plant types based on features such as leaf color and height. Classification is part of the supervised learning problem, where machine learning algorithms are provided with training data that have class labels to learn how to make accurate predictions[3], [4]. Once the algorithm is trained, it can apply that knowledge to predict classes from new, unlabeled data. Various machine learning algorithms, such as Decision Tree, Random Forest, Support Vector Machines (SVM), and k-Nearest Neighbors (KNN), can be used to solve classification problems. The choice of algorithm depends on the characteristics of the data and the specific problem to be solved[5].

In this study we classified lettuce plant diseases. These plant diseases are available in the form of images that have been converted in .csv format to be classified. These plant diseases are available in the form of images that have been converted in .csv format to be classified. Image These plant diseases have been divided into several classes or categories. Then we determine the features of each row and column of the dataset. Each line in the CSV file represents one image, and each column represents one feature Each line in the CSV file represents one image, and each column represents one feature[6]. Then a label is made for each line in the CSV file, namely the class or category where the images are grouped. Thus, so that we get datasets that are ready to be processed with machine learning[7]. However, in processing the dataset, we get imbalanced data. So we added the Synthetic Minority Over-sampling Technique (SMOTE) method to overcome the imbalance that occurs. So that the data can be classified using several algorithms to find the best accuracy.

2. Research Method

In this study, we have obtained image data of lettuce plant diseases that have been converted in csv form. Our dataset is obtained from the Kaggle website. The following stages of this research can be explained in the following flowchart:

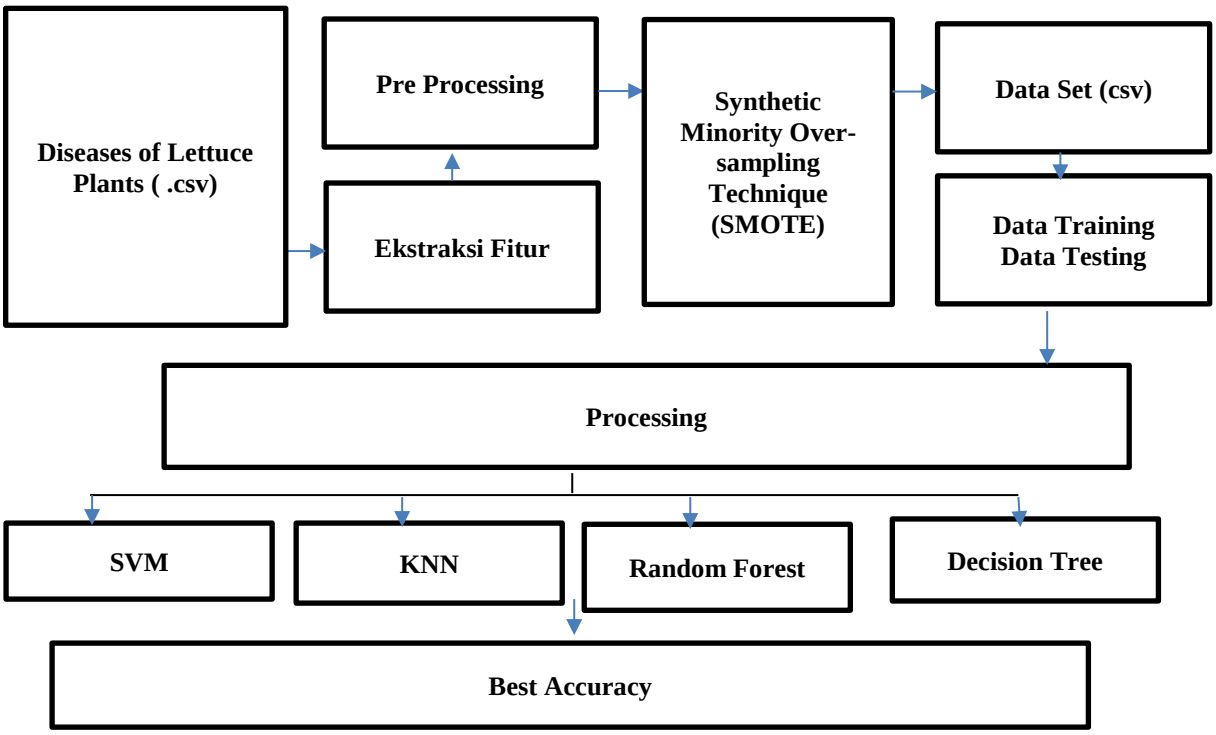


Figure 1. Research Method

2.1 Plant Disease Data

We take the dataset available on Kaggle for processing. The data set that we found is in the form of pictures of plant diseases. The types of plant diseases are as follows: Stem Rot, Leaf Rot, Leaf Caterpillars and Good Lettuce.

2.2 Feature Extraction

After getting the plant disease data, we do feature extraction with image processing. The features that we get in the picture include (XTL, YTL, XBR and YBR). XTL, YTL, XBR and YBR are techniques in image processing to retrieve features from each image based on the X and Y axis angles.

2.3 Pre Processing

In the Pre Processing process we read the data set using python. We use the Matplotlib Library. Where matplotlib is the most popular python library for visualizing data that is more interesting and easy to understand so that matplotlib will feel more natural to learn. From the results of Pre Processing data processing, the following description is obtained.

Table 1. Dataset Reading

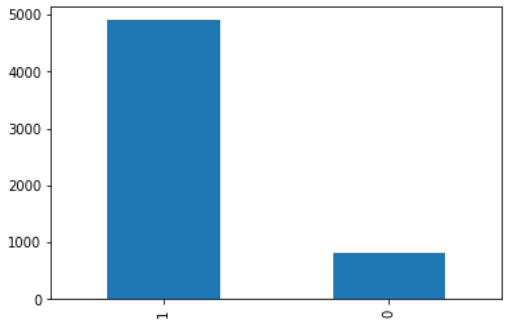
RangeIndex: 5923 entries, 0 to 5922
Data columns (total 7 columns):

#	Column	Non-Null Count	Dtype
0	Image	5923 non-null	Object
1	Grow	5923 non-null	int64
2	Disease	5923 non-null	int64
3	disease-grow	5923 non-null	Object
4	Area	5707 non-null	float64
5	Points	5707 non-null	Object
6	Original	2951 non-null	Object

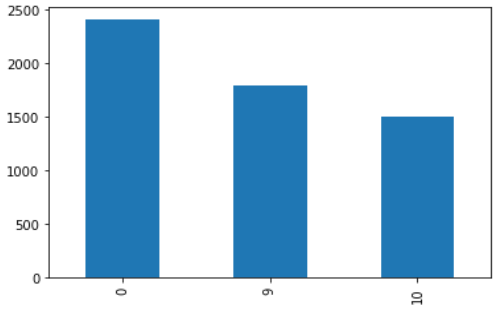
dtypes: float64(1), int64(2), object(4)
memory usage: 324.0+ KB

From table 1 it can be explained that it consists of 8 columns including image, grow, disease, disease-grow, area, points and original. With a total of 5923 data lines.

Then in the Pre Processing process we found null data so we deleted null and nan data so that the data results were 5707 data and 4 columns. Then we try to plot the data from each column so that we get the results of the data plot as follows:



Grow



Disease

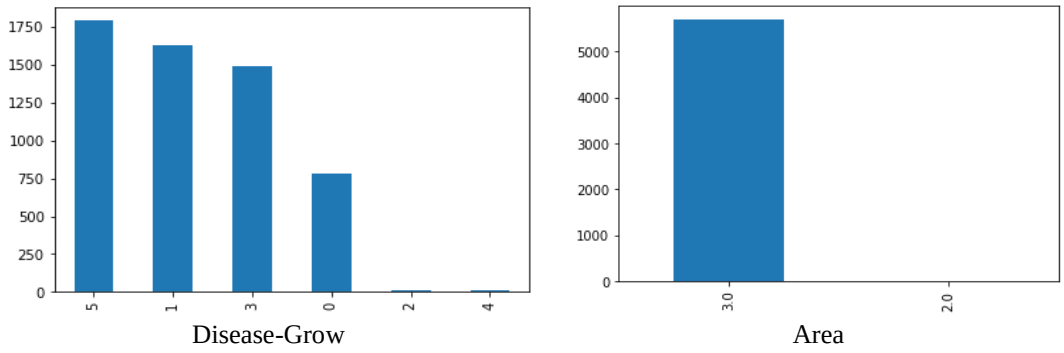


Figure 2. Plot Data

After plotting the data, there are 4 dominant features including Grow, Disease, Disease-Grow and Area. However, after plotting the data, there is an imbalance in the data.

From the graph above, it can be seen that there is an imbalance in the existing data. Then we try to do smooth data. In real data, there are many situations where the number of instances in one class is far less than the number of instances in another class. This situation is referred to as the problem of unbalanced datasets (imbalance class). As a result, classification performance usually decreases in some data mining applications. In this study, it was identified that the lettuce plant disease dataset used had a very large class imbalance problem where instances that had high rating values, much less than instances with small and medium rating values. So that an over-sampling method is needed to overcome the class imbalance problem. The method that can be used is the Synthetic Minority Over-sampling Technique (SMOTE). Synthetic Minority Oversampling Technique(SMOTE) is one of the derivatives of oversampling. SMOTE was first introduced by Nithes V. Chawla. This approach works by creating a replication of minority data. This replication is known as synthetic data[8]–[10].

The SMOTE method increases the amount of minor class data to be equivalent to the major class by generating artificial data. The artificial or synthesized data is made based on the k-nearest neighbors (KNN). The number of k-nearest neighbors is determined by considering the ease of implementation[11]. Generating artificial data that is numerically different from categorical. Numerical data is measured by its proximity to the Euclidean distance while categorical data is simpler, namely by the mode value[12]–[16]. Calculation of the distance between examples of minor classes whose variables are on a categorical scale is done by using the Value Difference Metric (VDM) formula, namely:

$$\Delta(X,Y) = w_x w_y \sum_{i=1}^{N} \delta(x_i, y_i)^r \qquad (1)$$

Where :

$\Delta(X,Y)$: The distance between X and Y observations
$w_x w_y$: observed weight (negligible)
N : many explanatory variables
R : value of 1 (manhattan distance) or 2 (euclidian) distance
$\delta(x_i, y_i)$: distance between categories by formula:

$$\delta(V_1, V_2) = \sum_{i=1}^{n} \left| \frac{C_{1i}}{C_1} - \frac{C_{2i}}{C_2} \right|^k \qquad (2)$$

Where :

$\delta(V_1, V_2)$: Distance between values V_1 and V_2
C_{1i} : The Number of V_1 belonging to class i
C_{2i} : The Number of V_2 belonging to class i
I : The Number of ; i = 1,2,…….,m
C_1 : the number of 1's occurs
C_2 : the number of values 2 occurs
N : many categories
K : constant (usually 1)

Artificial data generation procedure for:

1) Numerical Data
   a) Calculate the difference between the main vector and its nearest neighbor.
   b) Multiply the difference by a random number between 0 and 1.
   c) Add these differences into the main values of the original main vector to get a new main vector.
2) Categorical Data
   a) Select the majority between the main vector under consideration and its k-nearest neighbors for the nominal value. If the same value occurs, choose randomly.
   b) Make the value the data instance of the newly created class.

As for the implementation in Python, we use the following script:

```
fitur = df1[['area', 'disease', 'disease-grow']].values
target = df1['grow']
from imblearn.over_sampling import SMOTE
oversample = SMOTE()
fitur, target = oversample.fit_resample(fitur, target)
target.value counts().plot(kind='bar')
<matplotlib.axes. subplots.AxesSubplot at 0x7fb7222514c0>
```

From the above script produces a data plot display as follows:

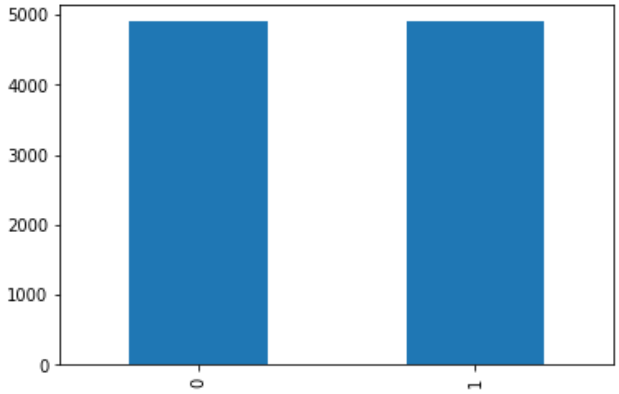


Figure 3. Balance Data

In Figure 3 it is explained that the Synthetic Minority Over-sampling Technique (SMOTE) method has been able to overcome the problem of data imbalance. The Smote method duplicates existing data to adjust the imbalances that occur so that the data obtained can later be tested and trained for classification processing.

2.4 Training Data and Testing Data

Training data and test data are separate parts of the dataset used in machine learning classification. The training data is used as a source for training algorithms, creating models and determining optimal parameters. In the training data, there are data entries that have known class labels, so that the algorithm can understand the relationship between attributes and class labels. Test data, on the other hand, is used to assess the performance of the model created through training. In the test data, the data entry has no known class label, so the algorithm must make class label predictions for each data entry. This prediction is then compared with the actual class label to determine the accuracy of the algorithm. Separation of training data and test data is very important to ensure that the algorithm does not study the test data and does not experience overfitting. Overfitting occurs when the algorithm learns too much about patterns in the training data and is unable to apply this knowledge to new data. Therefore, test data is used to validate the performance of the algorithm and ensure that it is capable of working properly on new data. In this case we took 7353 data for training data, and 2451 data for testing data.

2.5 Classification

After we got the dataset from this research, we then classified it using machine learning. We try to do a classification comparison using several methods including :

2.5.1. Support Vector Machine (SVM)

Support Vector Machine (SVM) is a method in supervised learning which is usually used for classification (such as Support Vector Classification) and regression (Support Vector Regression). In classification modeling, SVM has a more mature and clearer concept mathematically compared to other classification techniques[17], [18]. SVM can also solve classification and regression problems with linear and non-linear. The syntax in Python is as follows:

```
from sklearn.svm import SVC
clf = SVC()
clf.fit(x_train, y_train)
SVC()
y pred = clf.predict(x test)
accuracy score(y test, y pred)
```

2.5.2. K-Nearest Neighbor *(KNN)*

The KNN algorithm is a classification algorithm that works by taking a number of K closest data (neighbors) as a reference for determining the class of new data. This algorithm classifies data based on similarity or similarity or proximity to other data. In K-Nearest Neighbor, data points that are close together are called "neighbors" or "neighbors". The working principle of K-Nearest Neighbor (KNN) is to find the shortest distance between the data to be evaluated and the k closest neighbors in the training data. Where k is the number of nearest neighbors. In determining the value or class of k , we should use an odd number, because if not, there is a possibility that we will not get an answer. Determining the value of k is considered based on the amount of existing data and the size of the dimensions formed by the data. The more data there is, the lower the number k chosen should be. However, the larger the size of the data dimension, the higher the number k chosen should be. To find the distance between points in class k, it is usually calculated using the Euclidean distance[19]–[21]. The Euclidean distance is a formula for finding the distance between 2 points in two-dimensional space.

$$d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

Basically, the formula used in KNN is a distance formula such as Euclidean Distance, Manhattan Distance, or Minkowski Distance. This distance formula is used to calculate the distance between the new data and each data in the dataset.

3.5.3. Random Forest

Random Forest is one of the machine learning methods in the Ensemble algorithm group. This algorithm combines the results of several decision trees to make predictions. Each tree makes its own prediction and the results of all trees are combined, usually through averaging or voting. Random Forest is commonly used in solving classification and regression problems. For individual decision tree calculation in Random Forest, the CART (Classification and Regression Tree) or ID3 (Iterative Dichotomiser 3) algorithm is usually used. This algorithm uses several metrics such as Gini Impurity or Entropy to determine the best feature for dividing the data[22]–[24].

3.5.4. Decision Tree

Decision Tree is a method in machine learning that aims to make predictions on a target. It works by building a branching tree that represents the features of the data. Decision Tree is easy to understand because of its representation in the form of a branching tree. This algorithm is also efficient in processing data and can handle data with numeric and categorical features. However, the method has a weakness in terms of easily causing overfitting and tends to have a bias towards the feature with the largest amount of data. Therefore, Decision Tree is often combined with other algorithms in ensemble learning, such as Random Forest, to improve prediction results. The Decision Tree algorithm uses techniques such as Information Gain, Gini Index, and Chi-Square to determine which feature to use at each internal node in dividing the data[25]–[27].

3. Results and Discussion

After obtaining the training and testing data, we performed classification experiments with several methods. The goal of classification in Machine Learning is to predict a class or label for new data based on the known data. This is done by building a model that can understand the relationship between the features in the data and the target label. The model is then used to make new label predictions. We used several algorithms to understand and distinguish between classes in the data, so that we can make more accurate and relevant predictions. The algorithms in classification processing produced the following results..

3.1. Support Vector Machine (SVM)

The result of classification using the Support Vector Machine Algorithm was obtained. The SVM Algorithm finds the best separating line that separates the data class as well as possible. This algorithm is very useful for data that has many features and requires good separation between classes. We obtained the following results.

Table 2. Accuracy results using SVM

Precision	Recall	f1-score	support	
0	1.00	0.99	0.99	1230
1	0.99	1	0.99	1221
Accuracy		0.99	2451	
Macro avg	0.99	0.99	2451	
Weighted avg	0.99	0.99	2451	

Table 2 can explain Precision, which measures how well the model predicts positive classes by dividing the number of correct positive class predictions by the total number of positive class predictions. On the other hand, Recall measures how well the model finds all actual positive classes by dividing the number of correctly predicted positive classes by the total number of actual positive classes. The results are 1 and 0.99 for Recall. Meanwhile, F1-Score is the harmonic mean of Precision and Recall, giving equal weight to both metrics. F1-Score is a good metric to combine both aspects. The result is 0.99 and 1 for F1-Score. Support measures the number of data in each class. Support can help you understand how well the model predicts each class in your data. In this study, the support value was 0.99.

Then, we also calculated Macro avg and weighted avg. Macro-average calculates the average performance metric for each class without considering the amount of data in each class. It calculates the average of each performance metric (precision, recall, f1-score) for each class and produces one number for each metric. Weighted-average calculates the average performance metric for each class, taking into account the amount of data in each class. It calculates the weighted average of each performance metric for each class, giving more weight to classes with larger amounts of data. In this algorithm, the Macro and Weighted avg values are the same, which is 0.99. From the overall Support Vector Machine algorithm, we obtained an accuracy value of 0.99265605875153.

3.2 K-Nearest Neighbors (KNN)

Then we performed the same steps as in 3.1. However, this time we used the KNN algorithm. This algorithm works by finding the K nearest neighbors from each new data point and predicting the class of the new data point based on the majority class of the K nearest neighbors. In this research, we set the value of K=3. As a result, we obtained the following classification results.

Table 3. Accuracy results using KNN

Precision	Recall	f1-score	support	
0	1.00	0.99	0.99	1230
1	0.99	1	0.99	1221
Accuracy		0.99	2451	
Macro avg	0.99	0.99	2451	
Weighted avg	0.99	0.99	2451	

The results obtained using the KNN algorithm are the same as the SVM algorithm where the highest accuracy results are obtained with a value of 1.

3.3 Random Forest

We also classify using the Random Forest algorithm. Where this algorithm is based on the concept of ensemble, which combines several Decision Tree models into one Random Forest model. Where we get the accuracy results of 0.9840881272949816. Following are the results of the accuracy using the Random Forest algorithm.

Table 3. Accuracy results using Random Forest

Precision	Recall	f1-score	support	
0	1.00	0.99	0.99	1230
1	0.99	1	0.99	1221
Accuracy		0.99	2451	
Macro avg	0.99	0.99	2451	
Weighted avg	0.99	0.99	2451	

3.4 Decission Tree

Lastly we compare with the Decision Tree algorithm. Decision Tree for classification. The basic principle of the Decision Tree is to create a decision tree with nodes that determine the final class of data. Each node in the decision tree has attributes that are used to make decisions. From this algorithm we get an accuracy value of 1. The following is the final result using the Decision Tree algorithm.

Table 3. Accuracy results using Random Forest

Precision	Recall	f1-score	support	
0	1.00	0.99	0.99	1230
1	0.99	1	0.99	1221
Accuracy		0.99	2451	
Macro avg	0.99	0.99	2451	
Weighted avg	0.99	0.99	2451	

Of all the classification algorithms that we use, we try to represent them using graphs. The results can be seen in Figure 4.

Classification Comparison Results of Each Algorithm

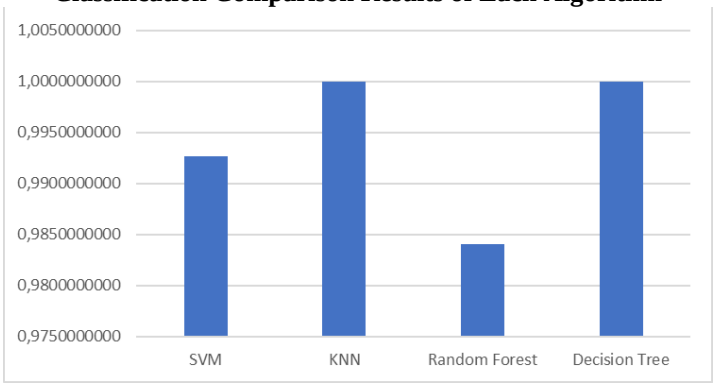


Figure 3. Comparison of Classification Results Using SVM, KNN, Random Forest and Decision Tree Methods

4. Conclusion

Overall, the Synthetic Minority Over-sampling Technique (SMOTE) method has been able to provide a solution in data imbalance. However, the results of the classification testing showed a accuracy value of 1 for both the KNN and Decision Tree methods. If the accuracy test result is 1 in the classification algorithm test, it indicates that the model correctly predicts all classes. This means that the model has very good performance and can be used to accurately predict classes. However, it should be noted that an accuracy score of 1 does not always guarantee that the model will perform well on new or unseen data. Therefore, further evaluation is necessary to ensure that the model has good generalization. The lowest accuracy was obtained in Random Forest with an accuracy value of 0.985.

Acknowledgement

This research is fully funded by the Research Funding from the Computer Science Faculty of Lancang Kuning University Pekanbaru Indonesia for the fiscal year 2022.

References

[1] A. E. Maxwell, T. A. Warner, and F. Fang, "Implementation of machine-learning classification in remote sensing: An applied review," *Int J Remote Sens*, vol. 39, no. 9, pp. 2784–2817, 2018, doi: 10.1080/01431161.2018.1433343.

[2] S. B. Kotsiantis, I. D. Zaharakis, and P. E. Pintelas, "Machine learning: A review of classification and combining techniques," *Artif Intell Rev*, vol. 26, no. 3, pp. 159–190, 2006, doi: 10.1007/s10462-007-9052-3.

[3] R. Konieczny and R. Idczak, "Mössbauer study of Fe-Re alloys prepared by mechanical alloying," *Hyperfine Interact*, vol. 237, no. 1, pp. 1–8, 2016, doi: 10.1007/s10751-016-1232-6.

[4] O. F.Y, A. J.E.T, A. O, H. J. O, O. O, and A. J, "Supervised Machine Learning Algorithms: Classification and Comparison," *International Journal of Computer Trends and Technology*, vol. 48, no. 3, pp. 128–138, 2017, doi: 10.14445/22312803/ijctt-v48p126.

[5] A. A. Soofi and A. Awan, "Classification Techniques in Machine Learning: Applications and Issues," *Journal of Basic & Applied Sciences*, vol. 13, pp. 459–465, 2017.

[6] H. Sunaryanto, M. A. Hasan, and G. Guntoro, "Classification Analysis of Unilak Informatics Engineering Students Using Support Vector Machine (SVM), Iterative Dichotomiser 3 (ID3), Random Forest and K-Nearest Neighbors (KNN)," *IT Journal Research and Development*, vol. 7, no. 1, pp. 36–42, Aug. 2022, doi: 10.25299/itjrd.2022.8912.

[7] N. Nasution, M. Rizal, D. Setiawan, and M. A. Hasan, "IoT Dalam Agrobisnis Studi Kasus : Tanaman Selada Dalam Green House," *It Journal Research and Development*, vol. 4, no. 2, pp. 86–93, 2019, doi: 10.25299/itjrd.2020.vol4(2).3357.

[8] N. Thanh-Long, Tran-Minh, and L. Hong-Chuong, "A Back Propagation Neural Network Model with the Synthetic Minority Over-Sampling Technique for Construction Company Bankruptcy Prediction," *International Journal of Sustainable Construction Engineering and Technology*, vol. 13, no. 3, pp. 68–79, Oct. 2022, doi: 10.30880/ijscet.2022.13.03.007.

[9] F. Yang, K. Wang, L. Sun, M. Zhai, J. Song, and H. Wang, "A hybrid sampling algorithm combining synthetic minority over-sampling technique and edited nearest neighbor for missed abortion diagnosis," *BMC Med Inform Decis Mak*, vol. 22, no. 1, p. 344, Dec. 2022, doi: 10.1186/s12911-022-02075-2.

[10] O. Oluwaseyi *et al.*, "SYNTHETIC MINORITY OVER-SAMPLING TECHNIQUE AND RESAMPLE APPROACH FOR ANDROID MALWARE DETECTION USING TREE-BASED CLASSIFIERS Detection of Phishing URLs View project Remote Weapon Station View project SYNTHETIC MINORITY OVER-SAMPLING TECHNIQUE AND RESAMPLE APPROACH FOR ANDROID MALWARE DETECTION USING TREE-BASED CLASSIFIERS." [Online]. Available: https://www.researchgate.net/publication/365650520

[11] E. Erlin, Y. Desnelita, N. Nasution, L. Suryati, and F. Zoromi, "Dampak SMOTE terhadap Kinerja Random Forest Classifier berdasarkan Data Tidak seimbang," *MATRIK : Jurnal Manajemen, Teknik Informatika dan Rekayasa Komputer*, vol. 21, no. 3, pp. 677–690, Jul. 2022, doi: 10.30812/matrik.v21i3.1726.

[12] A. Fernández, S. García, F. Herrera, and N. v Chawla, "SMOTE for Learning from Imbalanced Data: Progress and Challenges, Marking the 15-year Anniversary," 2018.

[13] N. v Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic Minority Over-sampling Technique," 2002.

[14] H. Han, W.-Y. Wang, and B.-H. Mao, "Borderline-SMOTE: A New Over-Sampling Method in Imbalanced Data Sets Learning," 2005.

[15] L. Torgo, R. P. Ribeiro, B. Pfahringer, and P. Branco, "SMOTE for Regression."

[16] J. Wang, M. Xu, H. Wang, and J. Zhang, "Classification of Imbalanced Data by Using the SMOTE Algorithm and Locally Linear Embedding."

[17] P.-H. Chen, C.-J. Lin, and B. Schölkopf, "A Tutorial on ν-Support Vector Machines."

[18] T. Joachims, "SVMLight: Support Vector Machine," 2018. [Online]. Available: https://www.researchgate.net/publication/243763293

[19] S. Zhang, X. Li, M. Zong, X. Zhu, and D. Cheng, "Learning k for kNN Classification," *ACM Trans Intell Syst Technol*, vol. 8, no. 3, Jan. 2017, doi: 10.1145/2990508.

[20] M.-L. Zhang and Z.-H. Zhou, "Ml-knn: A Lazy Learning Approach to Multi-Label Learning."

[21] G. Guo, H. Wang, D. Bell, Y. Bi, and K. Greer, "KNN model-based approach in classification," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and*

Lecture Notes in Bioinformatics), vol. 2888, pp. 986–996, 2003, doi: 10.1007/978-3-540-39964-3_62.
[22] G. Biau and E. Scornet, "A Random Forest Guided Tour," Nov. 2015, [Online]. Available: http://arxiv.org/abs/1511.05741
[23] L. Breiman, "Random Forests," 2001.
[24] Y. Qi, "Random Forest for Bioinformatics."
[25] S. R. Safavian and D. Landgrebe, "A Survey of Decision Tree Classifier Methodology," 1990.
[26] S. Nowozin, C. Rother, S. Bagon, T. Sharp, B. Yao, and P. Kohli, "Decision Tree Fields."
[27] J. R. Quinlan, "Learning Decision Tree Classifiers," 1996.