CHAPTER 4
ANALYSIS AND DESIGN

4.1 Analysis

4.1.1 Collecting data

This student data collection stage was made with dummy data or I made it myself. The dataset needed in this study is the IP value (Semester Index) 1 to 8 and the student's graduation status. This dummy data is saved in CSV format. After creating dummy data, then classifying the data with the two algorithms. Classify this data using python language and create from scratch.

4.1.2 Make Classification with the KNN Algorithm

a. Euclidean Distance

$$d = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

p,q = Two points in the Euclidean room

q_i,p_i = Euclidean vector, starting from space origin (starting point)

n = space-n

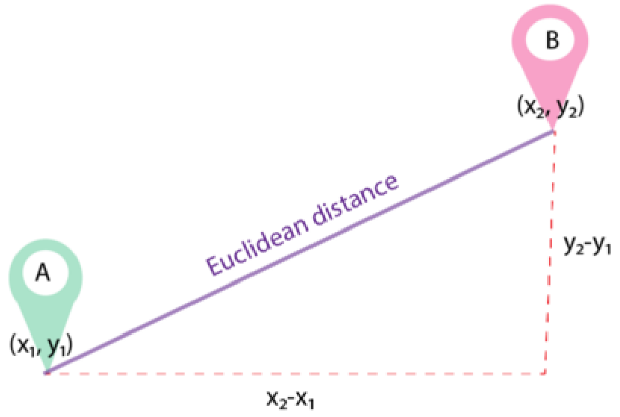


Illustration 1: Image Euclidean Distance [12]

Sample 1 Euclidean distance:

Instance 1: 3.1, 3.06, 3.0, 3.23, 2.79, 3.0, 2.41, 3.0, 'TEPAT'

Instance 2: 3.07, 2.85, 3.2, 3.0, 3.21, 2.47, 2.55, 2.75, 'TEPAT'

Euclidean Distance:

$$\sqrt{(3{,}1-3{,}07)^2+(3{,}06-2{,}85)^2+(3{,}0-3{,}2)^2+(3{,}23-3{,}0)^2+(2{,}79-3{,}21)^2+(3{,}0-2{,}47)^2}$$

$$\sqrt{+(2{,}41-2{,}55)^2+(3-2{,}75)^2}$$

= 0.6773

From the sample above, it shows that instance 1 and instance 2 will be calculated using the euclidean distance formula to find the distance between 2 or more points in a certain dimensional space.

b. Sort from Euclidean Distance

After getting the results of the euclidean distance from instance 1 and instance 2, the next step is to sort the Euclidean distance values from the smallest to the largest values. After sorted, retrieve the smallest data based on k values. If the value of k = 5 then the 5 smallest neighbors will be chosen.

c. Get the best k score from the prediction

To get the best k value, the technique used is the brute force search technique where k is the best by comparing the k values that appear. The following is the result of the k value and its accuracy.

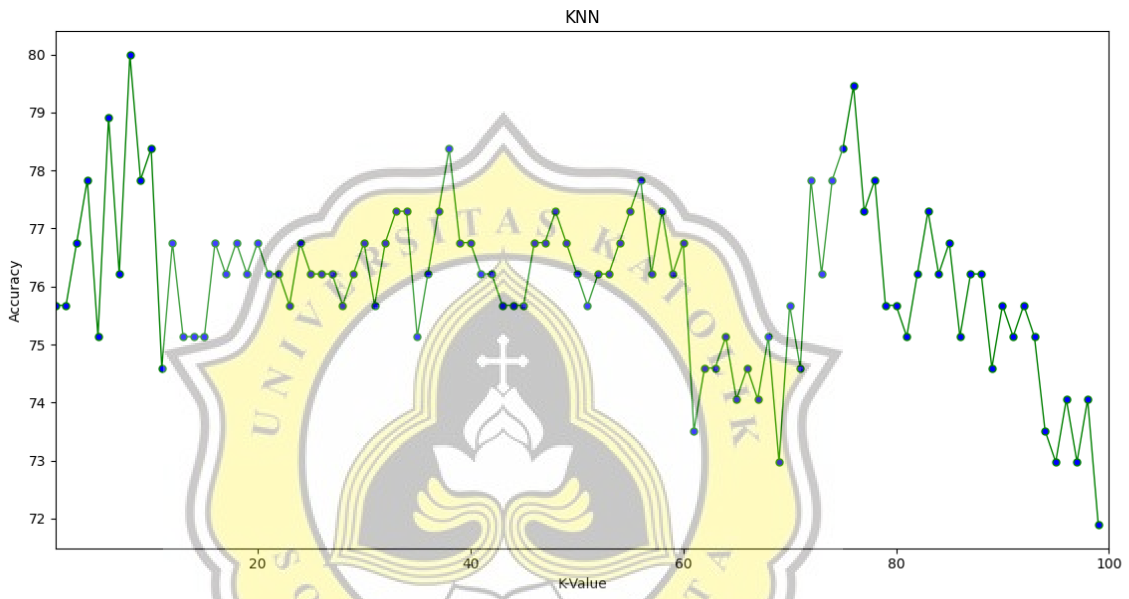


Illustration 2:Figure K value results with data split as 50%

Based on illustration 2 shows that the best k value with a data split of 50% (training 50% and testing 50%) is the value of k = 8 with an accuracy of 80%.

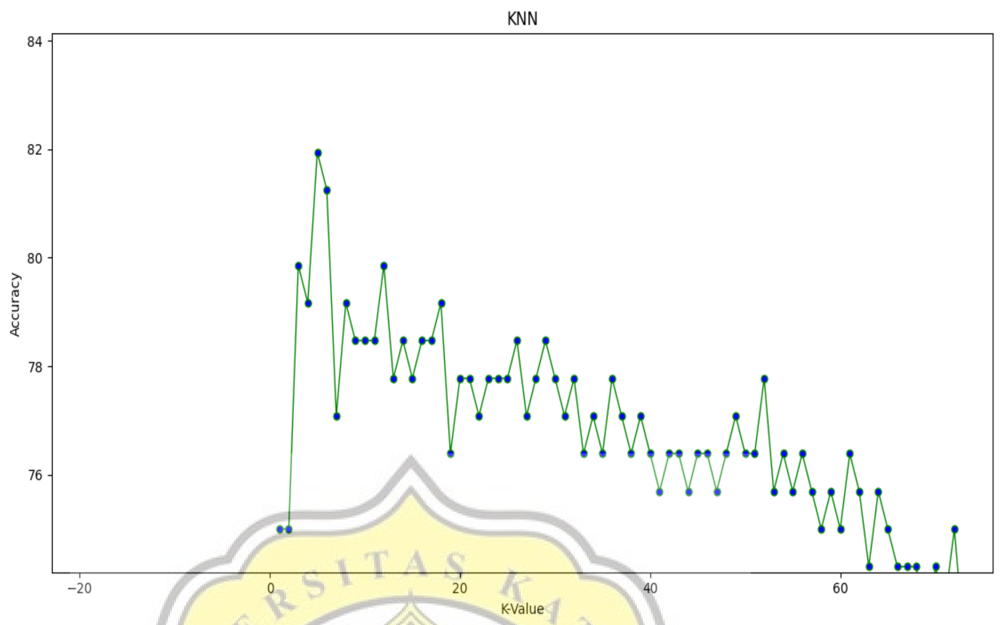


Illustration 3:Figure K value results with data split as 60%

Based on illustration 3, it can be seen that the best k value with 60% data split (60% training and 40% testing) is the value of k = 5 with an accuracy of 81.94%.

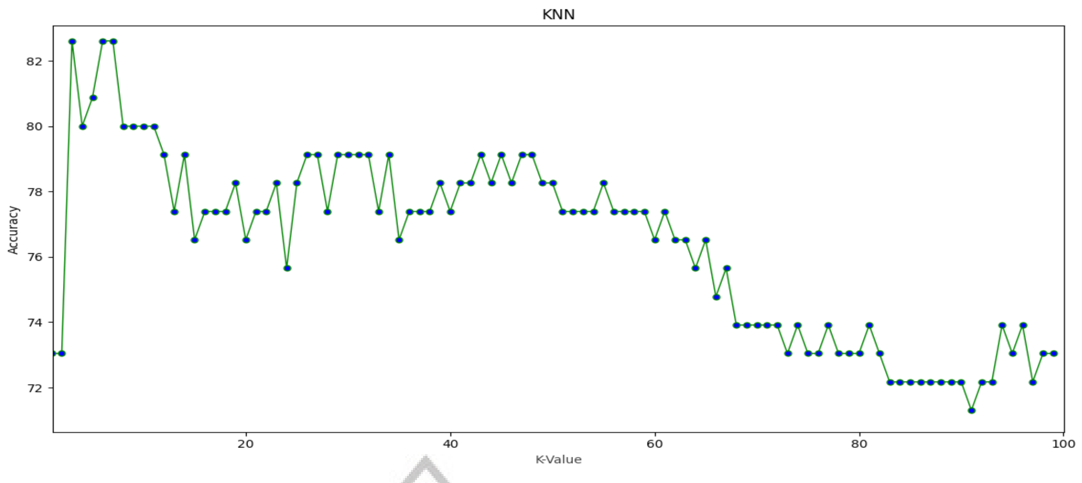


Illustration 4:Figure K value results with data split as 70%

Based on illustration 4, it can be seen that the best k value with 70% data split (70% training and 30% testing) is the value of k = 3, 6, 7 with an accuracy of 82.60%.

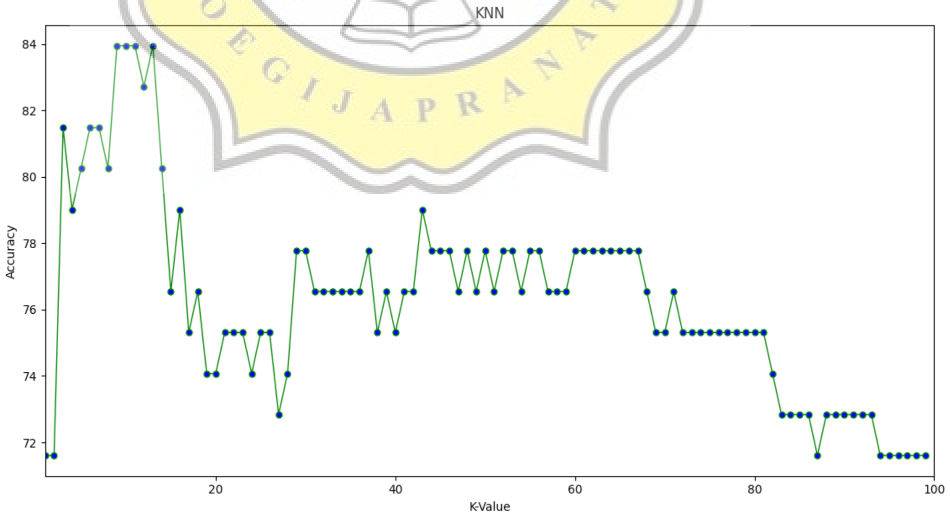


Illustration 5:Figure K value results with data split as 80%

Based on illustration 5, it can be seen that the best k value with 80% data split (80% training and 20% testing) is the value of k = 9, 10, 11, 13 with an accuracy of 83.95%.

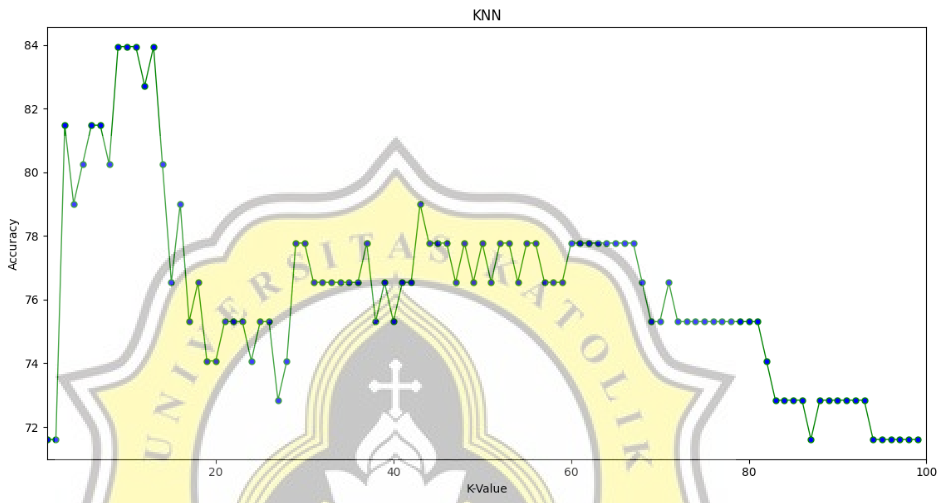


Illustration 6:Figure K value results with data split as 90%

Based on illustration 6, it can be seen that the best k value with 90% data split (90% training and 10% testing) is the value of k = 10, 12 with an accuracy of 86.84%.

d. Result Accuracy

After making the steps for the classification method by experimenting between training data and test Instance data. Next, make predictions and calculate the accuracy of these predictions. The prediction here replaces the testInstance data into testing data prepared from the number of split data. To calculate the accuracy, it is obtained by calculating the number of correct predictions based on the passing status with the tested data. The number of correct predictions is divided by the number of testing data then multiplied by 100. In this stage of calculating accuracy, it is based on the number of split data. Table 1 shows the best accuracy, precision, recall and f1 score results from the KNN algorithm of the split data of 50%, 60%, 70% and 80%,90%.

tp=predictions ‘TEPAT’, actual ‘TEPAT’

tn=predictions ‘TERLAMBAT’, actual ‘TERLAMBAT’

fp=predictions ‘TEPAT’, actual ‘TERLAMBAT’

fn=predictions ‘TERLAMBAT’, actual ‘TEPAT’

Accuracy = ((tp + tn) * 100) / float(tp + tn + fn + fp)

Precision = (tp * 100) / float(tp + fp)

Recall = (tp * 100)/ float(tp + fn)

F1 Score= (2*precision*recall)/ (precision + recall)

Table 1: The best accuracy, precision, recall and f1 score results from the KNN algorithm

	Split 50%	Split 60%	Split 70%	Split 80%	Split 90%
Accuracy	80.0%	81.94%	82.60 %	83.95%	86.84%
Precision	78.10%	83.91%	85.29%	86.0%	89.29%
Recall	94.0%	89.41%	91.18%	91.84%	92.59%
F1 Score	83.04%	84.92%	86.11%	87.38%	90.91%

e. Result Accuracy, Prescision, Recall and F1 Score with Add New Dataset

After using 372 data, we can add 300 or more data by calculating the accuracy, precision and recall. Adding this data will stop if we input the word no. The following are the results of the precision and recall accuracy of adding 300 data.

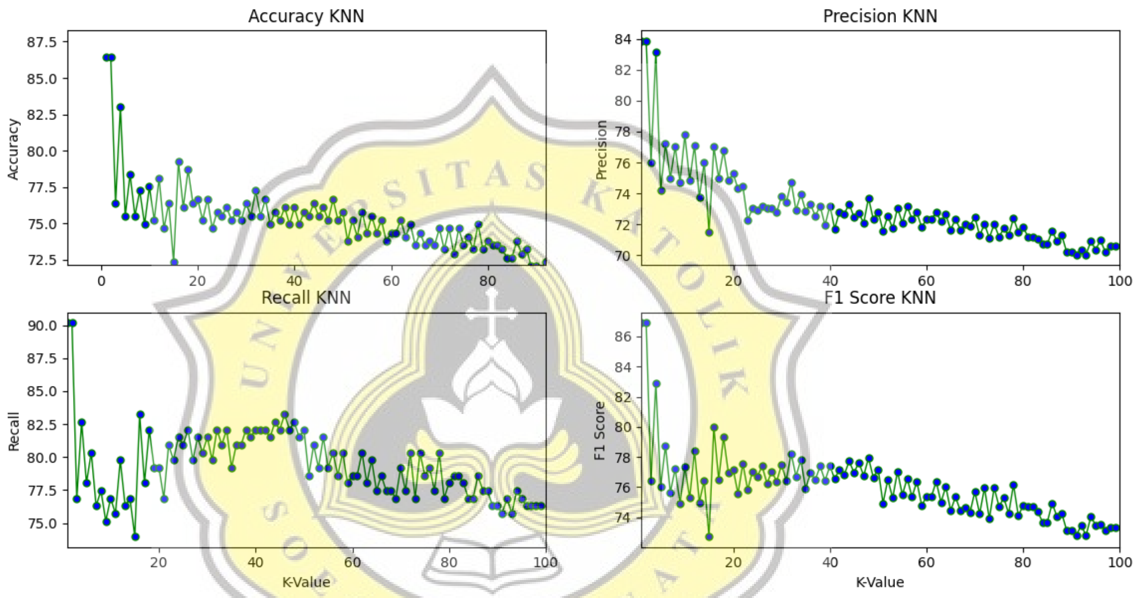


Illustration 7: Results of Accuracy, Precision, Recall and F1 Score with New Dataset with data split as 50%

Based on illustration 7, it illustrates the results of accuracy, precision, recall and F1 score based on a 50% split of data from the previous data set 372 data were added with new data sets totaling 300 data. The best accuracy results were 86.46% with k values of 1 and 2. The scores for precision, recall, and f1 score were 83.87%, 90.17%, and 86.91%.

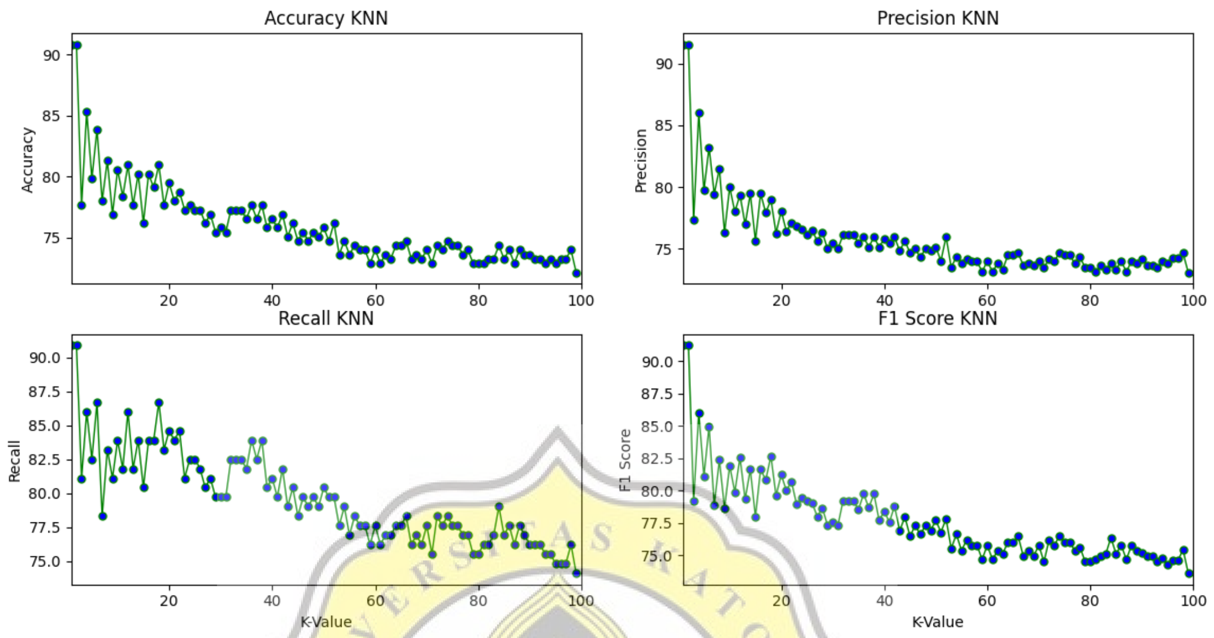


Illustration 8: Results of Accuracy, Precision, Recall and F1 Score with New Dataset with data split as 60%

Based on illustration 8, it illustrates the results of accuracy, precision, recall and F1 score based on a data split of 60% from the previous data set 372 data added with new data sets totaling 300 data. The best accuracy results are 90.84% with k values of 1 and 2. The scores for precision, recall, and f1 score are 91.55%, 90.91%, and 91.23%.

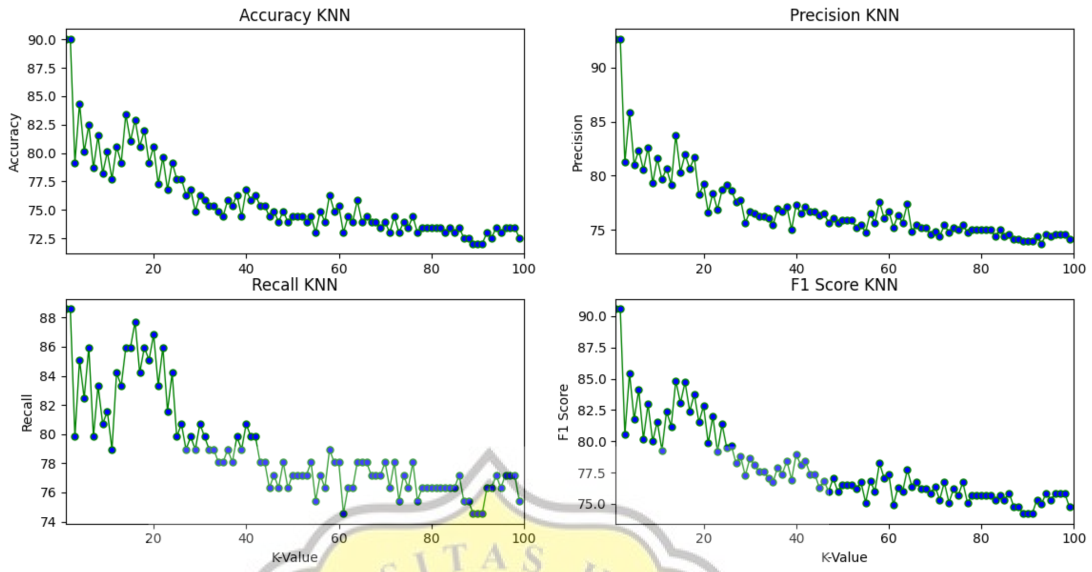


Illustration 9: Results of Accuracy, Precision, Recall and F1 Score with New Dataset with data split as 70%

Based on illustration 9, it illustrates the results of accuracy, precision, recall and F1 score based on a data split of 70% from the previous data set 372 data were added with new data sets totaling 300 data. The best accuracy results are 90.05% with k values of 1 and 2. The scores for precision, recall, and f1 score are 92.66%, 88.60%, and 90.58%.

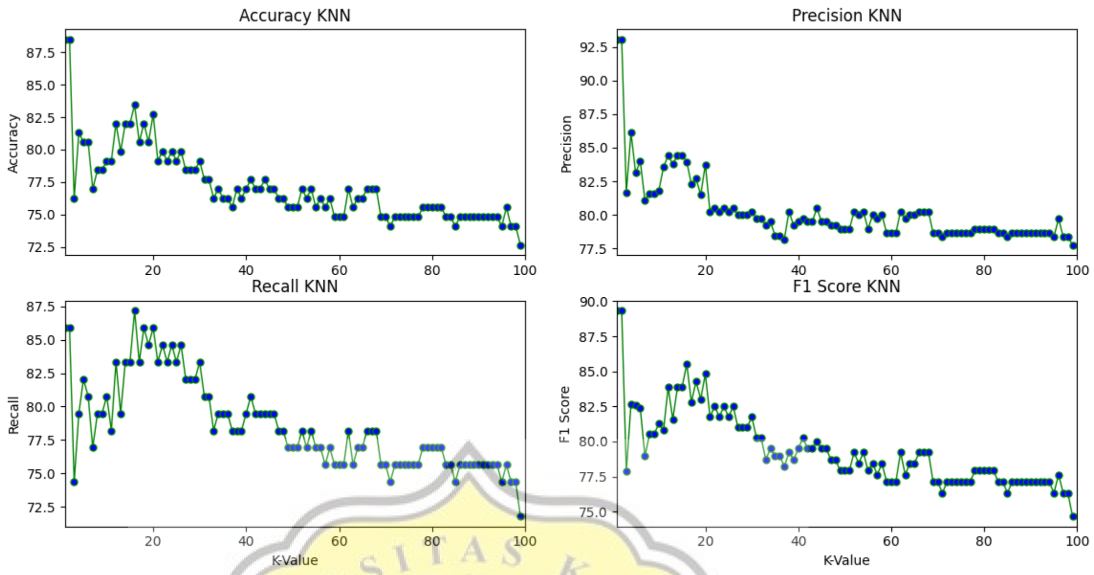


Illustration 10: Results of Accuracy, Precision, Recall and F1 Score with New Dataset with data split as 80%

Based on illustration 10, it illustrates the results of accuracy, precision, recall and F1 score based on a data split of 80% from the previous data set 372 data were added with new data sets totaling 300 data. The best accuracy results are 88.49% with k values of 1 and 2. The scores for precision, recall, and f1 score are 93.06%, 87.18%, and 89.33%. The best k value in recall for illustration 10 is the k value of 16.

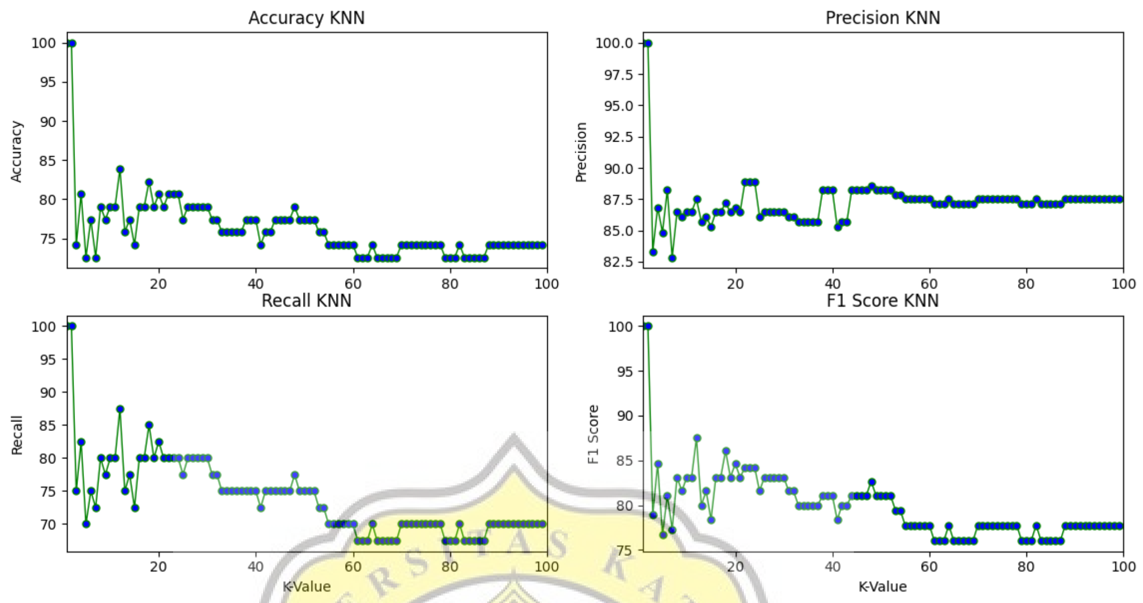


Illustration 11: Results of Accuracy, Precision, Recall and F1 Score with New Dataset with data split as 90%

Based on illustration 11, it illustrates the results of accuracy, precision, recall and F1 score based on a data split of 90% from the previous data set 372 data were added with new data sets totaling 300 data. The best accuracy results are 100% with k values of 1 and 2. The scores for precision, recall, and f1 score are 100%, 100%, and 100%. Based on the results of the precision accuracy, recall and f1 score of the 372 dataset and adding a new dataset of 300, it is concluded that the increase can affect the accuracy results. In this case, the accuracy with the new dataset is increased compared to the accuracy with a dataset of 372

4.1.3 Make Classification with the Random Forest Algorithm

The Random Forest algorithm is an algorithm that aims to create a forest with a number of trees. The more trees in the forest, the stronger the shape of the forest will be. The higher the number of trees in the forest, the higher the yield. Illustration 8 is a picture of the random forest algorithm.

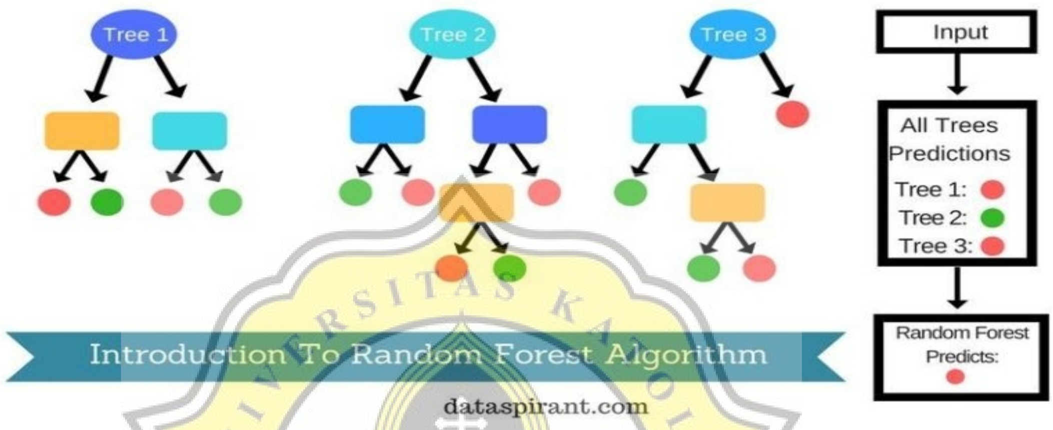


Illustration 12: Random Forest Classifier[13]

a. Gini Index

Gini index is a method of measuring or calculating the results of the level of equalization in a data. If the Gini Index is equal to 0 then the level of equalization is perfect. If the Gini Index is equal to 1, it means that there is a level of inequality that occurs

Gini= (1-Score) * (size/n_point)

size = random group length

n_point = sum of group and groups randomly

score = the score of each class

Table 2: Gini Index

Numbers of Group (Size)	Numbers of Groups	N_Point	Score	Gini Index
60	63	123	0.53	0.49
112	11	123	0.60	0.48
40	83	123	0.51	0.49
91	57	148	0.51	0.48
128	20	148	0.51	0.46
62	86	148	0.52	0.49

b. Best Split

After getting the value from the Gini index, the next step is to find the best value for the split data from the Gini index. Table 4.9 is the result of the Best Split Point. At this stage, looking for the best split is obtained from the smallest or lowest Gini index value.

Table 3: Result Best Split

Index	Value	Gini
7	0	0.0
7	1.5	0.0
2	3.21	0.35
1	3.3	0.06
4	2.61	0.19
2	3.02	0.16
3	3.36	0.20
6	1.29	0.0
0	2.43	0.06
2	2.96	0.11
6	2.73	0.0
0	2.88	0.0
4	3.14	0.0

c. Build Tree / Decision Tree

After getting the best value from the split data, the next step is to branch the tree. In this step, make a terminal or tree branch and create a node or leaf. This step is repeated continuously until the condition of the leaf or node is too small to displit. From this stage, you will check whether there is a split or not.

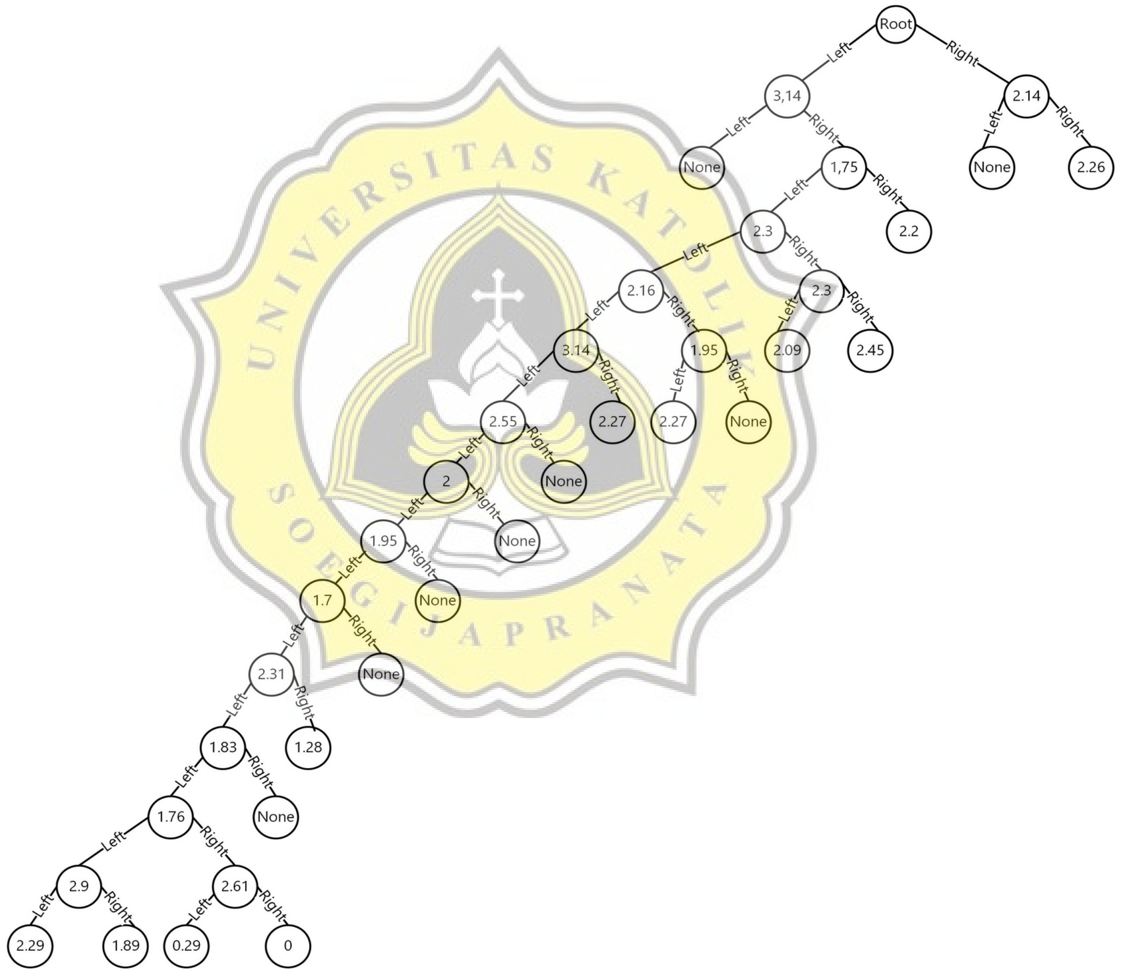


Illustration 13: Decision Tree

Illustration 13 explains that the result of the decision tree formation.If it is not there, it will create a new terminal that will connect the left and right node values. If the depth level is greater than the maximum depth, the left node will enter the left terminal and the right node will enter the right terminal. If the left length is less than the minimum size specified then the left node equals the left terminal. If not then the left node then the left node will look for the best split data value and the depth increases by 1. Likewise with the right child process. If the right length is less than the minimum size then the right node will be the right terminal. If not then the right node will look for the best split value from the right and n_features and the right node split will increase the depth level to 1.

d. Create a Random Subsample

At this stage the authors made a random subsample which aims to predict the results of student graduation status. At this stage the sample will be taken from the length of the dataset and then multiplied by the ratio. If the sample length is less than n_sample then the sample will print the dataset randomly.

e. Calculate Accuracy

To calculate the accuracy, it is done by adding up the scores that arise from the number of n_fold values then divided by the number of n_fold. The calculation is based on the results of prediction decisions and bagging predictions. After getting accurate results from various trees, the next step is to determine the best value for each split by selecting the highest accuracy value from the other accuracy values.

Table 4:Result Accuracy Random Forest from tree

	Split 50%	Split 60%	Split 70%	Split 80%	Split 90%
Tree = 1	68.38%	65.68%	64.87%	66.22%	64.87%
Tree = 5	73.24%	74.60%	71.62%	73.78%	72.16%
Tree = 10	72.97%	72.70%	75.68%	73.24%	76.49%

Based on table 4, it shows that the results of the accuracy value of each tree that arise from the random forest algorithm are based on a data split of 50%, 60%, 70%, 80%, 90%. The accuracy value here is taken from the average scores that appear in each tree. Each tree has 5 scores taken from the k folds cross validation model based on the formed folds.

Tabel 5: Result Accuracy, Precision, Recall, F1 Score

	Split 50%	Split 60%	Split 70%	Split 80%	Split 90%
Accuracy	79.73%	67.57%	75.68%	74.32%	70.27%
Precision	86.11%	68.0%	75.0%	72.73%	70.45%
Recall	75.61%	80.95%	85.71%	82.05%	77.5%
F1 Score	80.52%	73.91%	80.0%	77.11%	73.81%

Based on table 5, it shows that the results of accuracy, precision, recall and f1 score are from the last scores in the last tree. For accuracy, precision, recall and f1 score values were tested using split data of 50%, 60%, 70%, 80%, 90%. From these results will be compared with the results of the KNN algorithm.

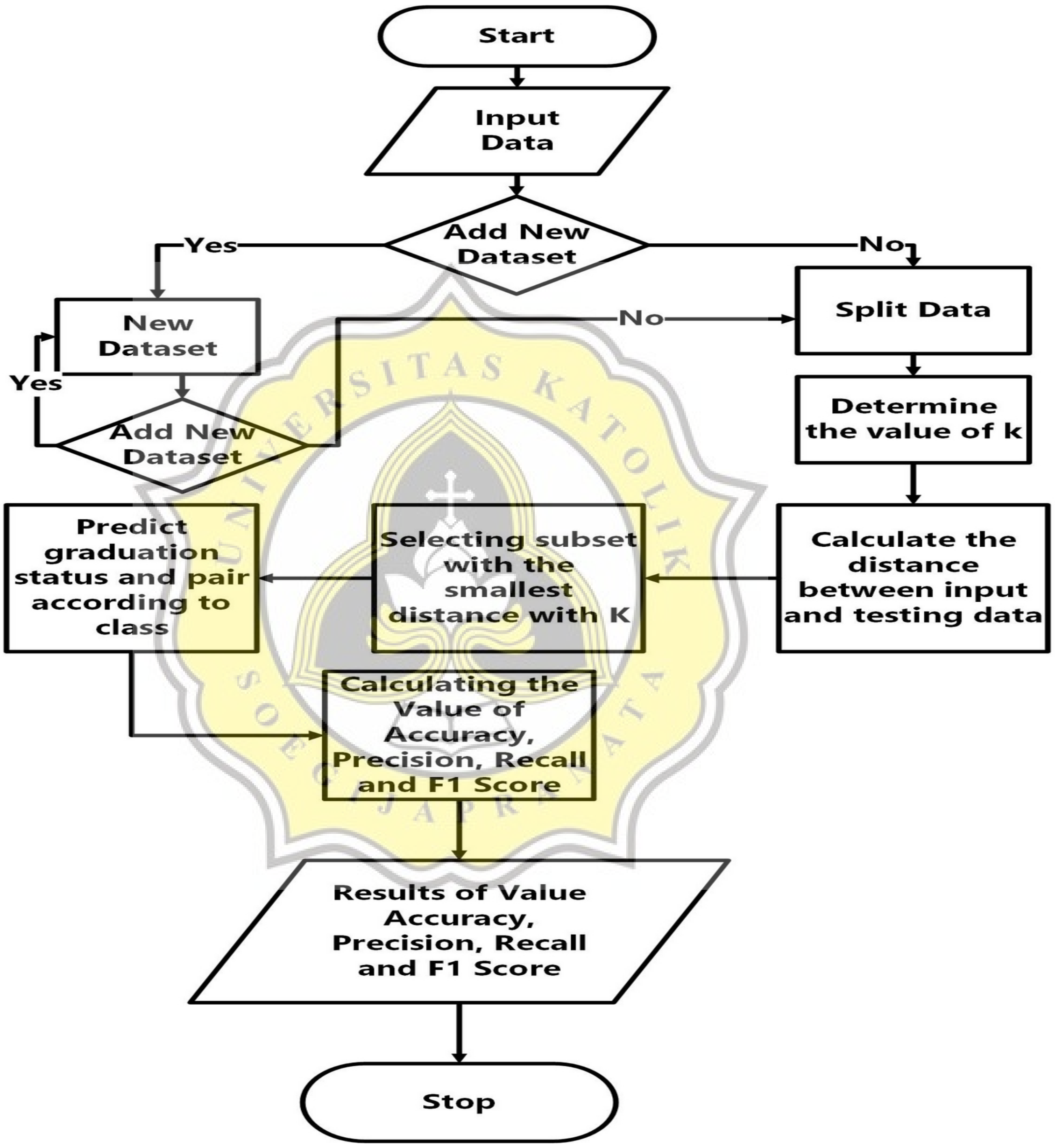
Tabel 6: Result Accuracy, Precision, Recall, F1 Score with Add New Dataset

	Split 50%	Split 60%	Split 70%	Split 80%	Split 90%
Accuracy	80.60%	86.57%	88.81%	92.54%	83.58%
Precision	78.05%	88.61%	92.42%	98.28%	76.92%
Recall	88.89%	88.61%	85.92%	86.36%	93.75%
F1 Score	83.17%	88.61%	89.05%	91.94%	84.51%

Based on table 6, it can be seen that the results of accuracy, precision, recall and f1 score come from the last score in the last tree based on the amount of initial data as much as 372 data added with a new dataset of 300 data. For accuracy, precision, recall and f1 values were tested using split data of 50%, 60%, 70%, 80%, 90%. From these results will be compared with the results of the KNN algorithm based on the amount of data added.

4.2 Desain

4.2.1 Flowchart KNN Algorithm



Start
Input Data
Add New Dataset
Yes
No
New Dataset
Yes
Add New Dataset
No
Split Data
Determine the value of k
Calculate the distance between input and testing data
Selecting subset with the smallest distance with K
Predict graduation status and pair according to class
Calculating the Value of Accuracy, Precision, Recall and F1 Score
Results of Value Accuracy, Precision, Recall and F1 Score
Stop

Illustration 14: Flowchart Knn Algorithm

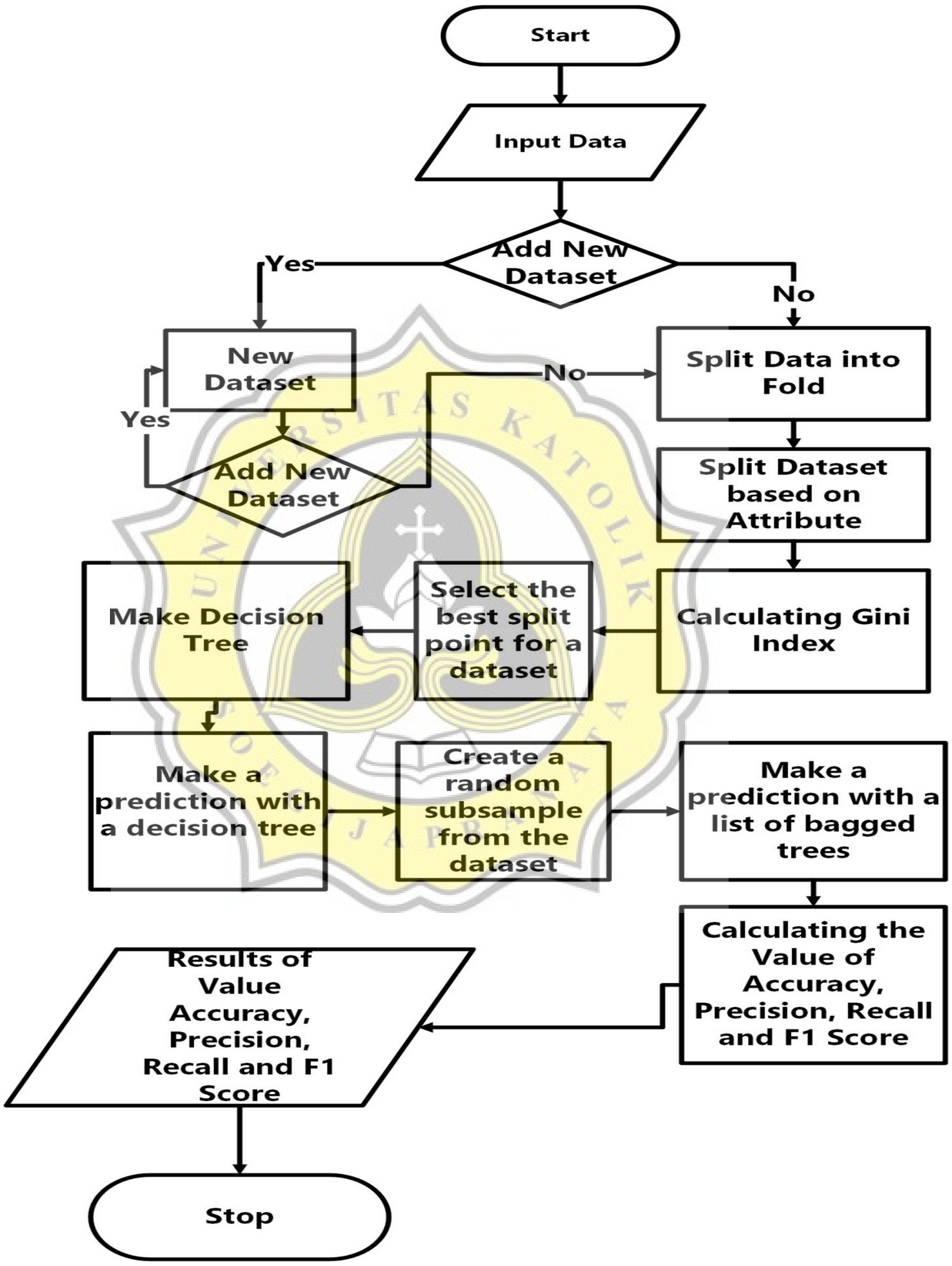
In illustration 14 is a program flow diagram showing the steps from the start of classifying data using the KNN algorithm. The data collection process is carried out with dummy data as data to be used in data classification. This dummy data will become a reference in data classification. After creating dummy data, the next step is to read and use the csv dataset. In the data input process we can add new datasets to classify. The add new dataset process will run continuously and will begin classifying if our input data is n which means no or stops.

Furthermore, the data will be used for data classification to see the extent of the comparison between the KNN algorithm and the random forest in data classification. The classification process in the KNN Algorithm uses the split dataset technique. The purpose of this technique is to divide the dataset into 2 parts, namely in the form of training data and test data.

To determine the amount of training data and testing data is determined by the amount of data that is divided. The amount of data splitting here starts with the 50%, 60%, 70%, 80% and 90% trials. So the number of splits shows the amount of training data and testing data. If the specified separation is 70%, the total training data is 70% and the test data is 30% of the total data. This applies to the number or other split amounts.

After dividing the data, the next step is to determine the value of k which uses the brute force search method, which searches for the optimum k value from the smallest k value to the specified k value. After that, calculate the distance between input and data testing with the Euclidean distance formula and sort the smallest distance with the k value determined from the Euclidean distance results. Next is to make predictions and pair according to class. Next is to calculate the value of accuracy, precision, recall and f1 score. The f1 score is the average of precision and recall.

4.2.1 Flowchart Random Forest Algorithm



Start
Input Data
Add New Dataset
Yes
No
New Dataset
Split Data into Fold
No
Yes
Add New Dataset
Split Dataset based on Attribute
Make Decision Tree
Select the best split point for a dataset
Calculating Gini Index
Make a prediction with a decision tree
Create a random subsample from the dataset
Make a prediction with a list of bagged trees
Calculating the Value of Accuracy, Precision, Recall and F1 Score
Results of Value Accuracy, Precision, Recall and F1 Score
Stop

Illustration 15: Flowchart Random Forest Algorithm

In illustration 15 shows that the data classification steps are using the Random Forest Algorithm. The process of inputting data and adding new data is the same as the method used by the KNN algorithm. Whereas for split data using k fold cross validation, where the distribution of data depends on the number of folds that exist. After that, divide the dataset based on attributes where the data is based on indexes and values such as the 1st semester cumulative grade point index to the 0th index.After dividing the data, the next step is to calculate the Gini index which aims to measure the probability of certain variables.

After getting the Gini index value, then we look for the best split point from the Gini index where the Gini index value = 0 then it is the best. Creating a decision tree for random forest determination. Making a decision tree here means making the terminal or leaves, roots, outgrowth of the tree that appears. After that, make a random sample as a basis for prediction. After making a random sample, the next step is to make predictions from the existing sample.

After that, calculate the value of accuracy, recall precision and f1 score. For the calculation of accuracy, precision, recall and f1 score are the same as the calculation of accuracy, recall and f1 score in the KNN algorithm. The difference is that the accuracy is obtained from the score that appears from the last tree. This random forest classification aims to create forest and decision trees from the formed forest.