KLASIFIKASI TANAMAN *Begonia brevirimosa* DAN *Begonia lugrae* MENGGUNAKAN METODE FRAKTAL DAN *DECISION TREE*

CLASSIFICATION OF Begonia brevirimosa AND Begonia lugrae PLANT USING FRACTAL METHOD AND DECISION TREE

TUGAS AKHIR

Disusun dalam rangka memenuhi salah satu persyaratan untuk menyelesaikan
Program Studi S1 Teknik Telekomunikasi

Disusun oleh:

OKTA KHIFDIL LISANAH

1101174498

FAKULTAS TEKNIK ELEKTRO

S1 TEKNIK TELEKOMUNIKASI

BANDUNG

2023

	UNIVERSITAS TELKOM	No. Dokumen	
	Jl. Telekomunikasi No. 1 Ters. Buah Batu Bandung 40257	No. Revisi	
	FORMULIR LEMBAR PENGESAHAN TUGAS AKHIR	Berlaku efektif	

LEMBAR PENGESAHAN

TUGAS AKHIR

KLASIFIKASI TANAMAN *Begonia Bririmosa* DAN *Begonia Lugrae* MENGGUNAKAN METODE FRAKTAL DAN *DECISION TREE*

CLASSIFICATION OF Begonia Brevirimosa AND Begonia Lugrae PLANT USING FRACTAL METHOD AND DECISION TREE

Telah disetujui dan disahkan sebagai Buku Tugas Akhir

Program Studi S1 Teknik Telekomunikasi

Fakultas Teknik Elektro

Universitas Telkom

Bandung

Disusun oleh

OKTA KHIFDIL LISANAH

1101174498

Bandung, 29 MEI 2023

Menyetujui,

Pembimbing I	Pembimbing II
Dr. Ir. Jangkung Raharjo, M.T.	Dr. Sutomo, S.Hut., M.Sc.
NIP. 91660051-1	NIP. 198103312005021001

LEMBAR PERNYATAAN ORISINALITAS

Nama : Okta Khifdil Lisanah
NIM : 1101174498
Alamat : Dk. Waru Pagojengan Rt 01 Rw 04, Kecamatan Paguyangan, Kabupaten Brebes
No. Telepon : 082138528376
Email : oktalisa0898@gmail.com

Menyatakan bahwa Tugas Akhir ini merupakan karya orisinal saya sendiri dibawah bimbingan Pembimbing I dan Pembimbing II, dengan judul:

KLASIFIKASI *Begonia brevirimosa* DAN *Begonia lugrae* MENGGUNAKAN METODE FRAKTAL DAN *DECISION TREE*

CLASSIFICATION OF Begonia brevirimosa AND Begonia lugrae PLANT USING FRACTAL METHOD AND DECISION TREE

Atas pernyataan ini, saya siap menanggung risiko/sanksi yang dijatuhkan kepada saya apabila kemudian ditemukan adanya pelanggaran terhadap kejujuran akademik atau etika keilmuan dalam karya ini, atau ditemukan bukti yang menunjukkan ketidakaslian karya ini.

Bandung, 31 Januari 2023

Okta Khifdil Lisanah
1101174498

ABSTRAK

Indonesia merupakan rumah bagi beragam flora dan fauna. Begonia merupakan salah satu jenis flora yang terdapat di Indonesia. Begonia adalah bunga dengan banyak spesies; ada lebih dari 1700 di dunia dan jumlahnya terus bertambah. Begonia, di sisi lain, merupakan tanaman langka, sehingga perlu dikembangkan sistem yang dapat mengidentifikasi jenis Begonia dengan cepat dan andal. Setiap spesies Begonia memiliki pola daun yang unik. Hanya ada 2 jenis Begonia yang digunakan dalam riset ini. Hal tersebut disebabkan pada saat pengambilan data foto di lapangan, hanya terdapat 2 jenis begonia yang "ready" atau tidak dalam fase dorman (tidur) serta jumlah spesimen (individual) yang tersedia mencukupi untuk diambil citranya secara re presentatif sebagai sampel uji berikutnya.

Pada penelitian ini, peneliti menerapkan algoritma *decision tree* untuk mengkategorikan jenis Begonia dengan pola daun yang diperoleh dengan metode *box counting*. Dengan mengukur tingkat kesamaan diri yang diamati pada skala yang berbeda, analisis fraktal digunakan untuk mengekstraksi karakteristik dari struktur tanaman seperti bentuk daun dan bentuk kelopak. Atribut ini kemudian digunakan untuk membangun pohon keputusan untuk klasifikasi tanaman fisik.

Metode fraktal dan *decision tree* memiliki presentase akurasi uji paling tinggi pada *resize* citra dengan ukuran 512 x 512 piksel yaitu sebesar 100%. Sedangkan akurasi uji paling rendah diperoleh pada resize citra memiliki ukuran 256 x 256 piksel yaitu sebesar 97,3%. Hasil penelitian menunjukkan bahwa metode fraktal dan *decision tree* memperoleh akurasi yang lebih tinggi dan mampu mengidentifikasi spesies tanaman yang sebelumnya belum ditemukan. Metode ini memiliki potensi untuk mengubah cara kita mengidentifikasi dan mempelajari spesies tumbuhan dengan memungkinkan kita mengekstrak informasi yang relevan dari struktur kompleks yang akan sulit diukur dengan menggunakan metode standar. Pendekatan ini dapat digunakan dalam berbagai aplikasi, termasuk konservasi tanaman, pertanian, dan penelitian keanekaragaman hayati.

Kata Kunci: *Begonia, Decision Tree, Box Counting*

ABSTRACT

Indonesia is home to a variety of flora and fauna. Begonia is a type of flora found in Indonesia. Begonia is a flower of many species; there are more than 1700 in the world and the number is growing. Begonias, on the other hand, are rare plants, so it is necessary to develop a system that can identify the Begonia species quickly and reliably. Each Begonia species has a unique leaf pattern. There are only 2 types of Begonia used in this research. This is because at the time of taking photo data in the field, there are only 2 types of begonias that are "ready" or not in the dormant phase (sleeping) and the number of specimens (individual) available is sufficient to be taken re-representative images as the next test sample.

In this study, researchers will create a classifier using a decision tree algorithm to categorize Begonia types with leaf patterns obtained by the box counting method. By measuring the degree of self-similarity observed at different scales, fractal analysis is used to extract characteristics from plant structures such as leaf shape and petal shape. These attributes are then used to construct decision trees for physical plant classification.

The fractal and decision tree methods have the highest percentage of test accuracy on resized images with a size of 512 x 512 pixels, which is 100%. While the lowest test accuracy is obtained on resized images having a size of 256 x 256 pixels which is equal to 97.3%.. The results showed that fractals and decision trees obtained higher accuracy and were able to identify plant species that had not previously been found. This method has the potential to change the way we identify and study plant species by enabling us to extract relevant information from complex structures that would be difficult to measure using standard methods. This approach can be used in a variety of applications, including crop conservation, agriculture, and biodiversity research.

***Keywords**: Begonia, Decision Tree, Box Counting*

KATA PENGANTAR

Puji syukur penulis ucapkan ke hadirat Tuhan Yang Maha Esa, atas berkat dan kasih karunia-Nya, penulis mampu menyelesaikan Tugas Akhir ini dengan judul KLASIFIKASI *Begonia brevirimosa* DAN *Begonia lugrae* MENGGUNAKAN METODE FRAKTAL DAN *DECISION TREE*. Penulis menyusun Tugas Akhir ini sebagai syarat guna menyelesaikan pendidikan sarjana Program Studi S1 Teknik Telekomunikasi pada Fakultas Teknik Elektro, Universitas Telkom.

Penulis menyadari bahwa Tugas Akhir ini jauh dari kesempurnaan. Hal tersebut didasari karena keterbatasan dan kekurangan penulis. Oleh karena itu, penulis sangat berharap mendapatkan kritik dan saran dari berbagai pihak agar Tugas Akhir ini dapat maksimal. Penulis juga berharap agar Tugas Akhir ini memiliki manfaat untuk para pembaca sehingga dapat dikembangkan lebih lanjut untuk kemajuan ilmu pengetahuan di institusi Telkom *University*, kota Bandung, negara Indonesia, dan di seluruh dunia.

Bandung, 31 Januari 2023

Okta Khifdil Lisanah

1101174498

UCAPAN TERIMA KASIH

Tugas Akhir telah disusun dengan segala upaya, bantuan, serta dukungan dari berbagai pihak. Dengan ini, penulis ingin menyampaikan rasa syukur dan terima kasih yang terdalam kepada:

1. Allah S.W.T yang telah memberikan rahmat, karunia serta nikmat yang tiada hentinya kepada penulis di sepanjang waktu.
2. Nabi Muhammad SAW yang selalu menjadi inspirasi bagi penulis dalam menjalani kehidupan.
3. Terimakasih kepada kedua orang tua penulis, abah Muh Ruchi dan mamah Siti Fadilah yang telah mendoakan serta beri dukungan, semangat serta motivasi kepada penulis sehingga tugas akhir ini dapat diselesaikan.
4. Pembimbing 1 Bapak Dr. Ir. Jangkung Raharjo, M.T. terimakasih telah memberikan ilmu, masukan dan arahan kepada penulis dalam penyusunan Tugas Akhir ini.
5. Bapak Dr. Sutomo, S.Hut., M.Sc. Selaku pembimbing 2 yang juga telah memberikan masukan serta arahan kepada penulis dalam menyelesaikan Tugas Akhir ini.
6. Bapak Dr. Gelar Budiman, S.T., M.T. Selaku dosen wali TT-41-08 yang sudah memberikan motivasi serta arahan selama penulis mengikuti perkuliahan.
7. Teman-teman terbaik penulis, Sherin, Desinta, Manda, Culis, Hani dan Anggun yang sudah menjadi teman sekaligus penyemangat bagi penulis.
8. Teman seperjuangan TT-41-08 yang senantiasa menemani penulis hingga memperoleh gelar Sarjana Teknik.
9. Terimakasih kepada Candra calon penulis yang telah support dari awal hingga akhir.
10. Kepada dosen serta civitas akademika Universitas Telkom. Terimakasih atas ilmu, wawasan dan pengalaman yang sudah diberikan kepada penulis selama menimba ilmu di Universitas Telkom.

DAFTAR ISI

LEMBAR PENGESAHAN ..... **i**
LEMBAR PERNYATAAN ORISINALITAS ..... **ii**
ABSTRAK ..... **iiii**
ABSTRACT ..... **iv**
KATA PENGANTAR ..... **v**
UCAPAN TERIMA KASIH ..... **vi**
DAFTAR ISI ..... **vii**
DAFTAR GAMBAR ..... **ix**
DAFTAR TABEL ..... **x**
DAFTAR SINGKATAN ..... **xi**
BAB I PENDAHULUAN **1**
1.1 Latar Belakang **1**
1.2 Rumusan Masalah **3**
1.3 Tujuan dan Manfaat **3**
1.4 Batasan Masalah **4**
1.5 Metode Penelitian **4**
BAB II TINJAUAN PUSTAKA **6**
2.1 Tanaman *Begonia* **6**
2.1.1 *Begonia brevirimosa* **6**
2.1.2 *Begonia lugrae* **6**
2.2 Metode Fraktal **6**
2.3 *Decision Tree* **8**
2.4 *K-Fold Cross Validation* **9**
2.5 *Confusion Matrix* **10**
BAB III PERANCANGAN SISTEM **13**
3.1 Desain Sistem **13**
3.2 Pengumpulan Data **14**
3.3 *Preprocessing Citra* **15**
3.4 Ekstraksi Ciri dengan *Box Counting* **15**

3.5 Pembagian Data 17
3.6 Pemilihan model dengan *K-Fold Cross Validation* 18
3.7 Permodelan *Decision Tree* 20
3.8 Pengujian Performansi 21
3.8.1 *Confusion Matrix* 21
3.8.2 Akurasi 22
3.8.3 *Precision* 22
3.8.4 *Recall* 22
3.8.5 *F1 Score* 22
BAB IV HASIL DAN PEMBAHASAN 23
4.1 Pengujian sistem 23
4.2 Spesifikasi Perangkat Keras 23
4.3 Spesifikasi Perangkat Lunak 23
4.4 Skenario Pengujian Sistem 23
4.5 Parameter Data Citra 38
4.6 Skenario Pelatihan Sistem Pertama 39
4.7 Analisis Skenario Pelatihan Sistem Pertama 39
4.8 Skenario Pengujian Sistem Kedua 42
4.9 Analisis Sistem Kedua 42
4.10 Hasil Cofusion Matrik dengan Percobaan Beberapa Ukuran Citra yang Berbeda 48
4.11 Pengujian Nilai Fraktal pada *F1 Score* Terhadap *Precision* dan *Recall*…..... 49
BAB V KESIMPULAN DAN SARAN 51
5.1. Kesimpulan 51
5.2. Saran 52
DAFTAR PUSTAKA 53
LAMPIRAN 55

DAFTAR GAMBAR

Gambar 2. 1 Contoh Pemrosesan *Box Counting* setiap iterasi 8
Gambar 2. 2 Pembagian Data pada *K-Fold* 10
Gambar 2. 3 Contoh *Confusion Matrix* 11
Gambar 3. 1 Desain Sistem 13
Gambar 3. 2 Diagram Alir Ekstraksi Ciri dengan *Box Counting* 16
Gambar 3. 3 Diagram Alir Pembagian Data..... 18
Gambar 3. 4 Diagram Alir Pemilihan Model 19
Gambar 4.1 Contoh pohon keputusan hasil perhitungan 36
Gambar 4. 2 Training Size Begonia *brevirimosa* 512 x 512 *pixels*..... 39
Gambar 4. 3 Testing Size Begonia *brevirimosa* 512 x 512 *pixels* 42
Gambar 4. 4 *Decision tree* Skenario Pertama..... 45
Gambar 4. 5 *Decision tree* Begonia *brevirimosa* 45
Gambar 4. 6 Iterasi Proses Fraktal Dimensi 2 47
Gambar 4. 7 Grafik Pengujian Ukuran *Resize* Citra Terhadap Poin Akurasi Latih dan Poin Akurasi Uji 48
Gambar 4. 8 Grafik Pengujian Nilai Dimensi Fraktal pada *F1 Score* dan *Recall* 49

DAFTAR TABEL

Tabel 3. 1 Sampel Data Tiap Kelas 14
Tabel 4.1 Tampilan contoh pengolahan citra meresize dan merubah citra rgb menjadi grayscale 24
Tabel 4.2 Tampilan contoh pengolahan citra merubah citra grayscale menjadi citra biner dan operasi komplemen 24
Tabel 4.3 Tampilan contoh penerapan box counting 24
Tabel 4.4 Contoh hasil perhitungan ekstraksi ciri fraktal 27
Tabel 4.5 Tampilan contoh hasil ekstraksi terhadap data latih 29
Tabel 4.6 Contoh data latih ciri hasil ekstraksi 32
Tabel 4.7 Nilai parameter Empty Boxes yang sudah diurutkan 32
Tabel 4.8 Pemecahan parameter ‘Empty Boxes’ di node akar 33
Tabel 4.9 Nilai parameter Full Boxes yang sudah diurutkan 33
Tabel 4.10 Pemecahan parameter ‘Full Boxes’ di node akar 34
Tabel 4.11 Nilai parameter Fractal Dimension yang sudah diurutkan 34
Tabel 4.12 Pemecahan parameter ‘Fractal Dimension’ di node akar 35
Tabel 4.13 Hasil perhitungan entropy dan gain untuk node akar 35
Tabel 4.14 Jumlah Data Latih dan Data Uji 38
Tabel 4.15 Pengujian Ukuran Resize Citra Terhadap Akurasi dan Waktu Komputasi 48
Tabel 4.16 Pengujian Nilai Dimensi Fraktal pada *F1 Score* Terhadap *Precision* dan *Recall* 49

DAFTAR SINGKATAN

RGB	:	*Red Green Blue*
TP	:	*True Positif*
TN	:	*True Negatif*
FP	:	*False Positif*
FN	:	*False Negatif*

BAB I
PENDAHULUAN

1.1 Latar Belakang

Indonesia merupakan negara yang kaya jenis flora dan faunanya. Salah satu jenis flora di Indonesia adalah dari marga *Begonia*. *Begonia* merupakan tanaman bunga yang memiliki spesies yang sangat besar, lebih dari 1700 spesies tanaman ini sudah ditemukan dan terus bertambah seiring berjalannya waktu [1]. Jenis *Begonia* yang banyak dijumpai di Indonesia di antaranya adalah jenis *Begonia brevirimosa* dan jenis *Begonia lugrae.* Di sisi lain, habitat alam *Begonia* semakin tergerus sehingga sebagian jenis *Begonia* menjadi langka [2]. Begonia yang digunakan dalam riset ini memang hanya 2 jenis. Hal ini dikarenakan pada saat pengambilan data foto ke lapangan, hanya 2 jenis begonia ini saja yang "ready" artinya tidak sedang dalam fase dorman (tidur) dan memiliki jumlah spesimen (individual) yang cukup untuk dapat diambil citranya yang re presentatif sebagai sampel untuk uji selanjutnya. Dengan semakin berkurangnya habitat alam dari *Begonia* itu sendiri menyebabkan perlunya dibentuk sistem yang mampu mengklasifikasikan tanaman *Begonia* agar dapat dikonservasi lebih cepat. Proses klasifikasi tersebut dapat dilakukan dengan algoritma *machine learning* salah satunya dengan menggunakan *decision tree*. *Begonia* yang memiliki beragam karakteristik [3] dapat diklasifikasikan dengan *decision tree*. Namun, performa algoritma *machine learning* ini sangat tergantung dari ciri yang digunakan sebagai masukan algoritma tersebut. Klasifikasi dengan dapat dilakukan dengan memanfaatkan ciri bentuk dan tekstur yang merupakan ciri utama dari identifikasi tanaman [4]. *Begonia* yang memiliki karakteristik daun yang berbeda-beda [5] dapat diklasifikasikan dengan menggunakan ciri dari bentuk daun *Begonia* tersebut. Ciri tersebut dapat diperoleh dengan metode ekstraksi ciri fraktal. Metode tersebut merupakan metode yang memanfaatkan bentuk geometri dari citra daun yang dimasukkan sehingga diperoleh ciri yang mewakili bentuk dan tekstur *Begonia*.

Beberapa penelitian telah dilakukan terkait klasifikasi tanaman salah satunya membahas mengenai identifikasi bunga dengan metode MKL-SVM [6]. Dalam penelitian tersebut ciri yang digunakan merupakan bentuk bunga tanaman itu sendiri dengan hasil akurasi sebesar 95%. Karena *Begonia* merupakan tanaman hias

yang keunikannya terletak pada daunnya, maka penggunaan daun dirasa lebih cocok dibandingkan bunga sebagai citra masukan untuk diterapkan. Penelitian lain dengan menggunakan daun sebagai ciri masukan pada berbagai algoritma *machine learning* menunjukkan hasil yang lebih baik [7]. Hasil klasifikasi menunjukkan hasil yang sangat baik khususnya pada algoritma *decision tree* yang menghasilkan akurasi hingga 100%. Namun, penelitian ini menggunakan ciri yang cukup banyak kontur, tekstur dan juga ciri warna pada daun. Penelitian ini juga tidak membahas mengenai klasifikasi tanaman *Begonia*. Penelitian mengenai pemanfaatan ekstraksi ciri fraktal telah dilakukan salah satunya untuk mengklasifikasikan darah putih dan darah merah [8]. Penelitian tersebut menunjukkan hasil yang cukup baik dimana akurasi yang dihasilkan adalah sebesar 100%. Hal ini menunjukkan bahwa fraktal mampu memberikan ciri yang memungkinkan untuk melakukan klasifikasi darah dengan mengambil bentuk geometri darah tersebut menggunakan metode *box counting*. Akan tetapi, penelitian ini hanya menggunakan *threshold* sebagai pembeda antara kelas darah putih dan darah merah. Penggunaan *threshold* tersebut memiliki kelemahan karena ciri yang digunakan untuk klasifikasi tanaman tidak linear sehingga diperlukan metode tambahan yang dapat melakukan klasifikasi dengan ciri yang tidak linear salah satunya dengan menggunakan *machine learning*. Penelitian lain dilakukan untuk menerapkan metode *fractal box counting* untuk pengenalan iris mata [9]. Penelitian ini sama seperti penelitian sebelumnya yang memanfaatkan *threshold* secara langsung untuk melakukan klasifikasi hanya jumlah kelas yang dibahas pada penelitian ini lebih banyak daripada penelitian sebelumnya. Hal ini menunjukkan bahwa metode fraktal *box counting* memiliki kapabilitas dalam mengekstraksi ciri untuk kelas yang banyak. Saat penelitian sebelumnya menggunakan metode KNN, pada klasifikasi KNN menghasilkan akurasi sebesar 100%. Berdasarkan hal tersebut, maka penulis terinspirasi untuk menggunakan algoritma decision tree pada tanaman begonia.

Berdasarkan permasalahan tersebut, serta diperlukannya sistem yang dapat mengklasifikasikan spesies *Begonia* khususnya jenis *brevirimosa* dan *lugrae* yang banyak dijumpai di Indonesia. Penulis mengambil judul "**KLASIFIKASI TANAMAN *Begonia brevirimosa* DAN *Begonia lugrae* MENGGUNAKAN METODE FRAKTAL DAN *DECISION TREE*"** yang akan dibuat dengan

menggunakan bahasa pemrograman Matlab. Penulis memanfaatkan daun yang menjadi fitur utama dalam klasifikasi tanaman dengan metode ekstraksi ciri *fractal box counting* yang sudah diujikan pada permasalahan lain namun belum diujikan untuk kasus klasifikasi tanaman. Penulis juga memanfaatkan algoritma *decision tree* untuk mengklasifikasikan *Begonia*. Algoritma ini dipilih karena algoritma ini memiliki performansi terbaik dalam mengklasifikasikan tanaman dengan citra input berupa daun pada penelitian yang telah dibahas sebelummnya. Selain itu, ekstraksi ciri dengan *fractal box counting* yang pada penelitian sebelumnya mampu melakukan klasifikasi hanya dengan *threshold* dapat membantu menutupi kelemahan *decision tree* yang lemah terhadap pemrosesan data yang bersifat numerik. Dengan semakin bertambahnya kerusakan habitat, populasi tumbuhan termasuk begonia semakin terancam. Kebun Raya Bali LIPI berperan penting dalam upaya konservasi jenis tumbuhan dan diantaranya adalah begonia. Upaya konservasi dimulai dengan melakukan identifikasi atau pengenalan jenis terlebih dahulu. Dengan semakin berkurangnya SDM di bidang taksonomi di Kebub Raya Bali LIPI maka kemampuan identifikasi tumbuhan juga berkurang. Untuk itu diperlukan metode lain yang berguna untuk membantu proses identifikasi tumbuhan dengan cepat. Dengan demikian bantuan teknologi dalam hal ini sangat diperlukan.

1.2 Rumusan Masalah

Berdasar pada latar belakang di atas, rumusan masalah yang akan dibahas pada penelitian ini adalah:

1. Bagaimana hasil ekstraksi ciri *Begonia brevirimosa* dan *Begonia lugrae* dengan metode fraktal?
2. Bagaimana performansi model *decision tree* yang dibuat dalam mengklasifikasikan *Begonia brevirimosa* dan *Begonia lugrae*?

1.3 Tujuan dan Manfaat

Adapun tujuan dan manfaat dari penelitian tugas akhir ini diantaranya:

1. Mengekstraksi ciri *Begonia brevirimosa* dan *Begonia lugrae*.
2. Mengklasifikasikan *Begonia brevirimosa* dan *Begonia lugrae* menggunakan *decision tree*.

1.4 Batasan Masalah

Batasan masalah dari penelitian ini adalah:

1. Bahasa pemrograman yang digunakan adalah *Matlab*.
2. Data yang digunakan adalah gambar daun *Begonia*.
3. Kelas *Begonia* yang diklasifikasikan adalah *Begonia brevirimosa* dan *Begonia lugrae*.
4. Algoritma untuk ekstraksi ciri yang digunakan adalah *Fractal Box Counting*, dan Algoritma yang digunakan untuk klasifikasi adalah *decision tree*.
5. Matriks performansi yang digunakan adalah akurasi, *f1-score*, *precision*, *recall*, serta *confusion matrix*.

1.5 Metode Penelitian

Adapun berbagai langkah yang dilakukan penulis dalam rangka mencapai tujuan penelitian adalah sebagai berikut:

1. Studi literatur

   Studi literatur dimaksudkan sebagai sarana untuk menemukan referensi yang berkaitan dengan topik penelitian yang penulis ambil agar tujuan penelitian dapat tercapai. Literatur dikumpulkan dari berbagai sumber seperti jurnal, dan buku.

2. Perancangan Sistem

   Setelah penulis mendapatkan informasi yang relevan untuk menyelesaikan permasalahan pada bagian latar belakang yang telah disebutkan sebelumnya, penulis merancang sistem yang sesuai sebagai solusi permasalahan tersebut. Sistem tersebut menjabarkan kebutuhan data, metode, serta analisis yang akan dilakukan.

3. Pengumpulan Data

   Proses berikutnya adalah pengumpulan data. Penulis mengambil data yang sesuai dengan kebutuhan sistem yang penulis rancang sebelumnya.

4. Mengimplementasikan sistem yang dirancang

   Setelah data yang dibutuhkan terkumpul, penulis mengimplementasikan sistem yang penulis sudah rancang sebelumnya.

5. Menguji performa sistem yang dikembangkan

   Sistem yang telah penulis implementasikan kemudian diuji performanya untuk melihat keberhasilan sistem dalam mengklasifikasi tanaman.

6. Analisa dan Kesimpulan

   Penulis kemudian melakukan analisis terhadap hasil implementasi dan performa dari sistem. Setelah analisa diperoleh, penulis kemudian membuat kesimpulan akhir dari penelitian yang penulis lakukan.

BAB II
TINJAUAN PUSTAKA

2.1 Tanaman *Begonia*

Tanaman *Begonia* adalah ragam tanaman hias yang memiliki beragam jenis dan tersebar diseluruh dunia. *Begonia* dapat dikelompokkan secara umum menjadi dua kategori, *Begonia* eksotik dan *Begonia* alam. *Begonia* alam merupakan *Begonia* yang ditemukan dari hasil penelusuran di hutan Indonesia, sedangkan *Begonia* eksotik merupakan *Begonia* yang bukan berasal dari alam namun merupakan hasil perkawinan silang menggunakan berbagai teknologi. *Begonia* memiliki bentuk daun yang dapat dikelompokkan menjadi beberapa bentuk daun yaitu Membundar (*orbicular*), Membundar telur (*ovate*), Elliptik/menjorong (*elliptic*), Lonjong (*oblong*), menjari (*palmate*), terompet, spiral, berumbai-rumbai [1][2]. *Begonia* memiliki banyak jenis, beberapa diantaranya yaitu jenis *Begonia brevirimosa* dan jenis *Begonia lugrae.*

2.1.1. *Begonia brevirimosa*

Begonia ini memiliki dua sub-spesies yaitu *B. brevirimosa Irmsch. susp. Brevirimosa*, dan *B. brevirimosa subsp. Exotica Tebbitt*. *B. brevirimosa Irmsch. susp. Brevirimosa* memiliki ciri daun yang berbentuk bundar telur(ovate-elliptic) yang memiliki panjang sebesar 14-16 cm dan lebar 7-8 cm, tangkai jenis ini hanya sepanjang 3 cm, warna dasar daunnya adalah hijau loreng dengan spot bertotol merah diantara tulang daun. Sedangkan *B. brevirimosa subsp. Exotica Tebbitt*, memiliki perbedaan pada warna dasar daunnya yang berwarna ungu atau merah muda dengan tonjolan dengan corak loreng hijau-garis merah [3].

2.1.2. *Begonia lugrae*

Begonia lugrae merupakan jenis Begonia yang haya ditemukan di Pulau Bali. Nama *lugrae* digunakan sebagai penghormatan untuk kepala Kebun Raya Bali saat itu I Nyoman Lugrayasa. Begonia ini memiliki ciri stipula yang asimetris dengan perpanjangan berdaging, tepi daun subentire (tidak bergelombang), lamina daun yang lebih besar (14–16,5 × 12–15 cm) , dan tangkainya lebih pendek dari tangkai daun [1].

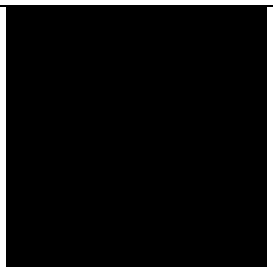
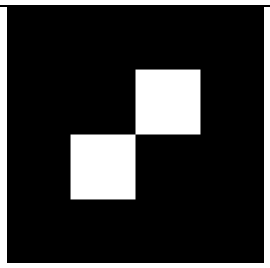
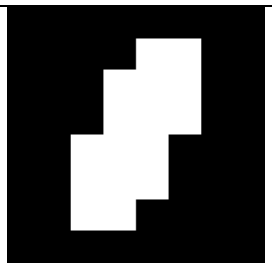
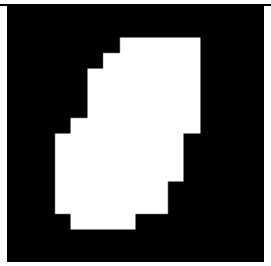
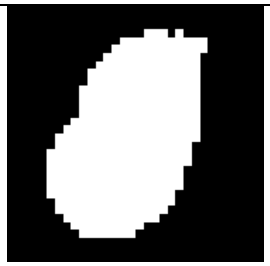
2.2 Metode Fraktal

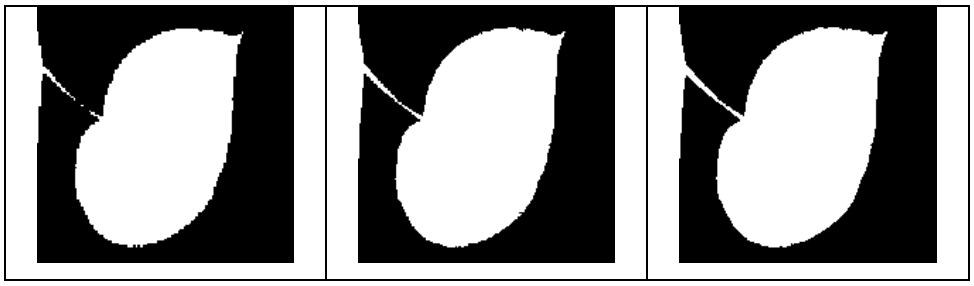
Fraktal adalah teknik pemisahan yang dilakukan dengan cara tertentu.

Mengurai gambar masukan menjadi serangkaian gambar dalam format digital 0 dan 1 yang nilainya ditentukan berdasarkan jumlah area menggunakan pengukuran. Fraktal untuk memperkirakan dan menghitung kompleksitas pola dalam suatu gambar [10].

Fraktal memiliki makna pecahan atau ketidakteraturan. Ini merupakan metode yang dapat digunakan untuk memodelkan representasi dari citra fenomena alam, objek alam, dan lain-lainnya. Fenomena-fenomena tersebut dapat dirangkum menjadi suatu ciri dengan menggunakan dimensi fractal. Salah satu metode untuk mencari dimensi fraktal yaitu dengan metode *box counting*. Metode ini memiliki kemampuan untuk mengestimasi kompleksitas dari suatu ciri spasial. Metode ini menutupi setiap ciri dengan sebuah *box* [11].

Box counting bekerja dengan membagi citra ke dalam beberapa kotak atau grid kemudian menghitung jumlah piksel yang berisi objek didalam kotak tersebut dan menghitung dimensi fraktalnya. Contoh pemrosesan yang terjadi pada algoritma ini dapat dilihat pada gambar 2.1.

iterasi 1	iterasi 2	iterasi 3
		
iterasi 4	iterasi 5	iterasi 6
		
iterasi 7	iterasi 8	iterasi 9



Gambar 2. 1 Contoh Pemrosesan *Box Counting* setiap iterasi

Persamaan untuk menghitung *box counting* dapat dilihat pada persamaan 2.1 [12].

$$D(s) = \frac{\log 2\, N(r)}{\log 2 \frac{1}{r}} \qquad (2.2)$$

Keterangan:

r : Ukuran kotak.

N (r) : Jumlah kotak yang berukuran s.

D (r) : Dimensi Fraktal dengan kotak berukuran s yang dihitung dengan metode *Box-Counting*.

2.3 *Decision Tree*

Decision tree adalah salah satu algoritma *machine learning* yang dapat digunakan sebagai sarana untuk mengklasifikasi. *Decision tree* bekerja dengan membentuk pohon keputusan berdasarkan data dan label yang diberikan. *Decision tree* membagi setiap nilai dalam setiap *attribute* pada data yang diberikan ke dalam cabang-cabang hingga diperoleh keputusan akhir dari pohon keputusan yang dibentuk.

Decision tree memiliki kelebihan yaitu sederhana untuk dipahami, *decision tree* juga memiliki kemampuan yang baik dalam menyeleksi fitur yang memiliki bias, dan mudah untuk ditafsirkan. Di sisi lain, *decision tree* juga memiliki kekurangan dimana *decision tree* sangat bergantung dari jumlah data, semakin banyak data akan semakin membuat decision tree kompleks. *Decision tree* juga tidak efektif untuk kasus dengan variable numerik [13].

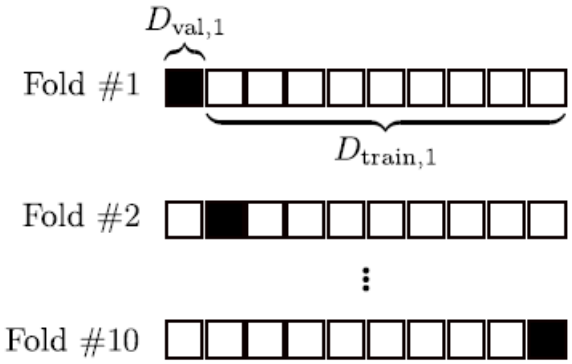
Decision tree telah banyak dikembangkan hingga menjadi beberapa jenis

seperti CART *(Classification and Regresstion Tree)*, ID3 (Iterative DIchotomiser 3(, C4.5, C5.0, CHAID (*Chi-square automatic intersection detector*), dan MARS (*multi adaptive regression splines*). CART menggunakan gini index sebagai *matrix* dalam pembagian akar dan cabangnya. CART dapat digunakan untuk kasus klasifikasi maupun regresi. ID3 mengggunakan entropi dan information gain sebagai *matrix* dalam pembagian akar dan cabangnya. ID3 mengharuskan data yang digunakan berbentuk kategorikal sehingga data dalam bentuk numerik harus diklasifikasikan dulu dalam bentuk kategorikal agar dapat diproses pada tipe *decision tree* ini. C4.5 merupakan versi peningkatan dari algoritma ID3 yang memberikan kemampuan untuk menangani data kategorikal dan numerik dengan baik. C5.0 merupakan peningkatan dari algoritma C4.5 dengan menambahkan kemampuan untuk memproses data yang kosong [13].

2.4 *K-Fold Cross Validation*

K-Fold Cross Validation, metode statistik yang berguna untuk mengestimasi performa model *machine learning* dalam melakukan pembelajaran. Metode ini biasa digunakan pada *machine learning* untuk membandingkan dan memilih model untuk memprediksi data yang telah diberikan karena metode ini mudah dipahami, diterapkan dan menghasilkan estimasi performa yang pada umumnya memiliki bias yang relatif lebih rendah dari metode lainnya. Metode ini bekerja dengan mengambil sampel secara berulang pada data untuk mengevaluasi pembelajaran model *machine learning* pada jumlah data yang terbatas [14].

Pada metode ini, terdapat parameter tunggal yang disebut K yang merujuk pada jumlah *Fold* atau grup dan dibagi berdasarkan sampel tertentu. Contoh pembagian data dalam beberapa grup dengan metode ini dapat dilihat pada gambar berikut ini.



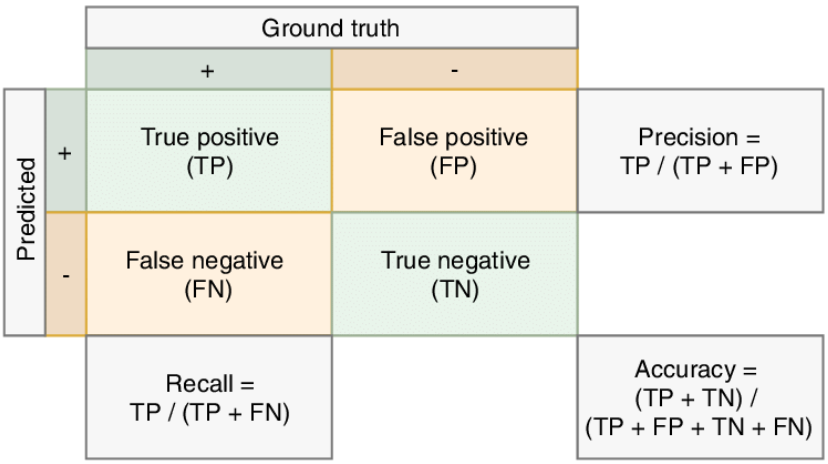
Gambar 2. 2 Pembagian Data pada *K-Fold* [17]

Dengan pembagian data tersebut, *k-fold cross validation* dapat mengestimasi performa model dalam memperkirakan data yang tidak digunakan dalam proses pelatihan. Prosedur metode ini adalah sebagai berikut:

- Acak data secara random.
- Pisahkan data kedalam beberapa grup.
- Untuk setiap grup, ambil satu grup sebagai data validasi dan grup lainnnya sebagai data latih.
- Lakukan pelatihan model *machine learning* pada data latih dan validasi performanya menggunakan data validasi.
- Rangkum hasil pengujian tersebut hingga diperoleh performa dari model machine learning yang digunakan.

2.5 *Confusion Matrix*

Confusion Matrix merupakan metode yang digunakan sebagai cara untuk melakukan evaluasi terhadap proses klasifikasi. Metode ini akan membandingkan hasil prediksi *classifier* dengan *ground truth* [15]. *Confusion matrix* memberikan rangkuman performa dari model *machine learning* dengan membentuk matriks dengan dimensi sesuai dengan jumlah kelas yang digunakan. *Confusion matrix* tidak hanya menyajikan akurasi sebuah model namun juga menunjukkan tipe kesalahan yang dihasilkan model tersebut. Berikut ini adalah gambar contoh *confusion matrix*.



Gambar 2. 3 Contoh *Confusion Matrix*

Matriks pada gambar 2.3. merupakan *confusion matrix* untuk dua kelas. Terdapat beberapa komponen pada diagonal matriks tersebut yaitu *true positive* (TP), *true negative* (TN), *false positive* (FP), dan *false negative* (FN). TP merupakan data dengan kelas positif yang berhasil diprediksi model. TN merupakan data dengan kelas negative yang berhasil diprediksi model. FP atau *Type I Error* merupakan data negatif yang diprediksi positif oleh model, FN atau *Type II Error* merupakan data positif yang diprediksi negatif oleh model. Komponen-komponen tersebut dapat menunjukkan performa model tersebut serta dapat digunakan untuk menghitung matrik lainnya seperti akurasi, *precision*, *recall*, dan *f1-score*.

Akurasi menunjukkan rasio prediksi model dari seluruh data yang diberikan. Akurasi merupakan matrik awal yang baik untuk mengukur performa suatu model. Namun, akurasi memiliki kelemahan yaitu akurasi menyembunyikan detail yang dibutuhkan untuk memahami performa model dengan lebih baik. Sebagai contoh apabila data yang dimiliki memiliki banyak kelas dengan akurasi yang tinggi. Pada kasus tersebut kita tidak dapat mengetahui kemampuan model tersebut pada tiap kelasnya, model bisa saja mengalami performa buruk pada kelas-kelas tertentu. Pada kasus kelas dengan jumlah data yang tidak seimbang tiap kelas, akurasi juga dapat mengabaikan kelas yang menjadi minoritas [15]. Persamaan akurasi dapat dilihat pada persamaan 2.2.

$$\text{Akurasi} = \frac{TP+TN}{TP+FP+FN+TN} \tag{2.2}$$

Precision menunjukkan rasio prediksi positif yang benar dari keseluruhan prediksi positif yang diprediksi model. Semakin besar nilai *precision* maka semakin rendah *rate false* positifnya. Hal tersebut menunjukkan semakin akurat prediksi positif yang diprediksi model tersebut. Persamaan *precision* dapat dilihat pada persamaan 2.3.

$$Precision = \frac{TP}{TP+FP} \quad (2.3)$$

Recall menunjukkan rasio prediksi kelas positif terhadap keseluruhan kelas positif pada model. Semakin besar recall maka model akan memiliki kemampuan yang makin baik dalam menghindari kelas negatif. Persamaan *recall* dapat dilihat pada persamaan 2.4.

$$Recall = \frac{TP}{TP+FN} \quad (2.4)$$

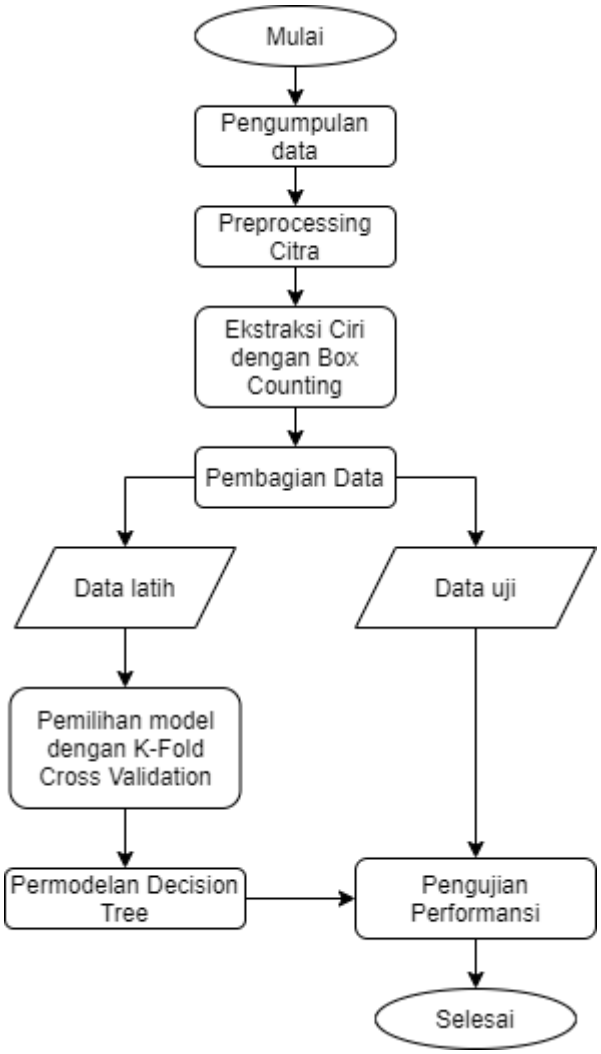
F1-Score menunjukkan keseimbangan antara *precision* dan juga *recall*. Persamaan *F1-score* dapat dilihat pada persamaan 2.5.

$$F1\ Score = \frac{2\ x\ Precision\ x\ Recall}{Precision+Recall} \quad (2.5)$$

BAB III
PERANCANGAN SISTEM

3.1 Desain Sistem

Rancangan sistem dibuat agar dapat melakukan klasifikasi terhadap *Begonia brevirimosa* dan *Begonia lugrae* dengan memanfaatkan algoritma *fractal box counting* dan *decision tree*. Perancangan sistem dapat dilihat pada gambar berikut ini.



Gambar 3. 1 Desain Sistem

Sistem dimulai dengan mengumpulkan data citra *Begonia*. Data citra tersebut kemudian melalui proses *preprocessing* agar data yang dimiliki dapat digunakan dalam sistem. Data yang telah dipreprocessing kemudian diekstrasi cirinya dengan menggunakan metode *fractal box counting*. Ciri yang diperoleh dari *fractal box*

counting kemudian dibagi kedalam dua bagian yaitu data latih dan data uji. Data latih dimaksudkan untuk proses pembentukan model, sedangkan data uji dimaksudkan sebagai proses untuk mengukur performansi model yang telah dibentuk. Proses pembentukan model terbaik dilakukan menggunakan *K-Fold Cross Validation*. Model terbaik yang telah diperoleh kemudian diukur performansinya pada data uji yang telah disiapkan. Detail mengenai proses pada sistem yang telah dirancang ini dapat dilihat pada bagian berikutnya.

3.2 Pengumpulan Data

Pengumpulan data pada penelitian ini dilakukan dengan mengambil gambar daun dari tanaman *Begonia*. Gambar daun diambil pada setiap kelas atau jenis tanaman *Begonia*, sampel data yang diambil dapat dilihat pada tabel 3.1 dibawah ini:

Tabel 3.1 Sampel Data Tiap Kelas

Kelas	Contoh Gambar
Begonia brivirimosa	
Begonia lugrae	

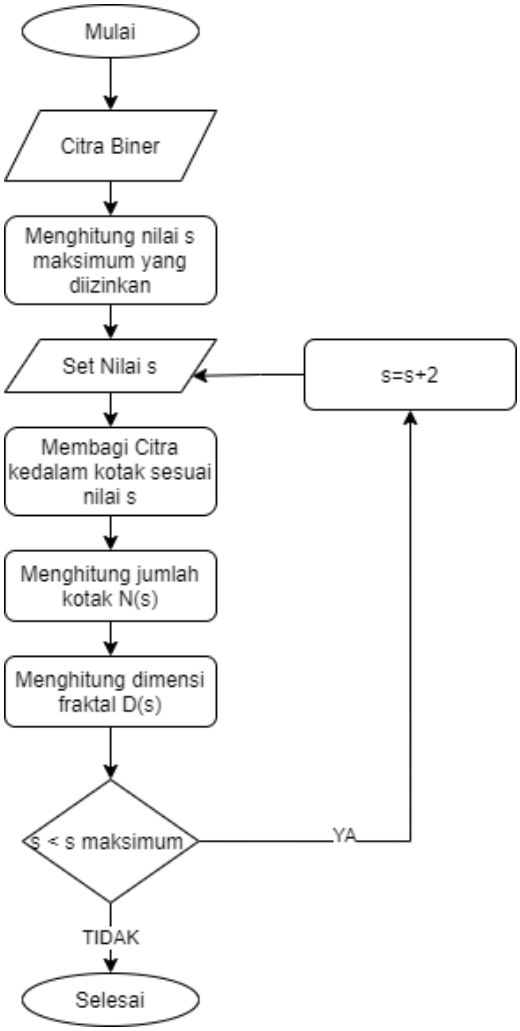
Data yang telah dikumpulkan memiliki ukuran yang bervariatif sehingga perlu dilakukan *pre-processing* agar data citra tersebut memiliki ukuran yang sama. Proses *pre-processing* citra dibahas pada bagian berikutnya.

3.3 *Preprocessing* Citra

Pada tahap ini, citra yang telah dikumpulkan kemudian di *pre-processing* agar dapat digunakan dalam sistem. *Pre-processing* yang dilakukan meliputi konversi citra RGB menjadi citra biner. Citra biner tersebut kemudian di resize ukurannya menjadi beberapa ukuran 120 x 120 piksel, 320 x 320 piksel, dan 640 x 640 piksel. Setelah citra dikonversi kedalam bentuk citra biner dan memiliki ukuran yang sama, maka data citra dapat dilanjutkan untuk diekstraksi cirinya menggunakan metode *fractal box counting*.

3.4 Ekstraksi Ciri dengan *Box Counting*

Ekstraksi ciri dengan *box counting* dilakukan untuk mendapatkan vektor yang akan digunakan sebagai masukan pada algoritma *decision tree*. Alur dari proses ekstraksi ciri ini dapat dilihat pada gambar 3.2 dibawah ini.



Gambar 3. 2 Diagram alir Ekstraksi Ciri dengan *Box Counting*

Citra biner diambil kemudian dilihat ukurannya sebagai acuan dalam menentukan nilai s maksimum yang diizinkan dalam percobaan ini. Proses ekstraksi ciri dimulai dengan s = 1 kemudian citra tersebut dibagi kedalam kotak-kotak (*grid*) dengan ukuran 2^s x2^s. Kemudian, hitung jumlah kotak N(s) yang melingkupi objek daun *Begonia*. Setelah itu hitung dimensi fraktal D(s). Lakukan perulangan hingga nilai s mencapai batas nilai s maksimum yang diizinkan.

Secara lebih mendetail langkah langkah ekstraksi ciri dengan *box counting* adalah sebagai berikut:

1. Mengambil citra biner sebagai input.
2. Menghitung nilai s maksimum dengan ukuran citra biner (m x m piksel) menggunakan persamaan 3.1 berikut:

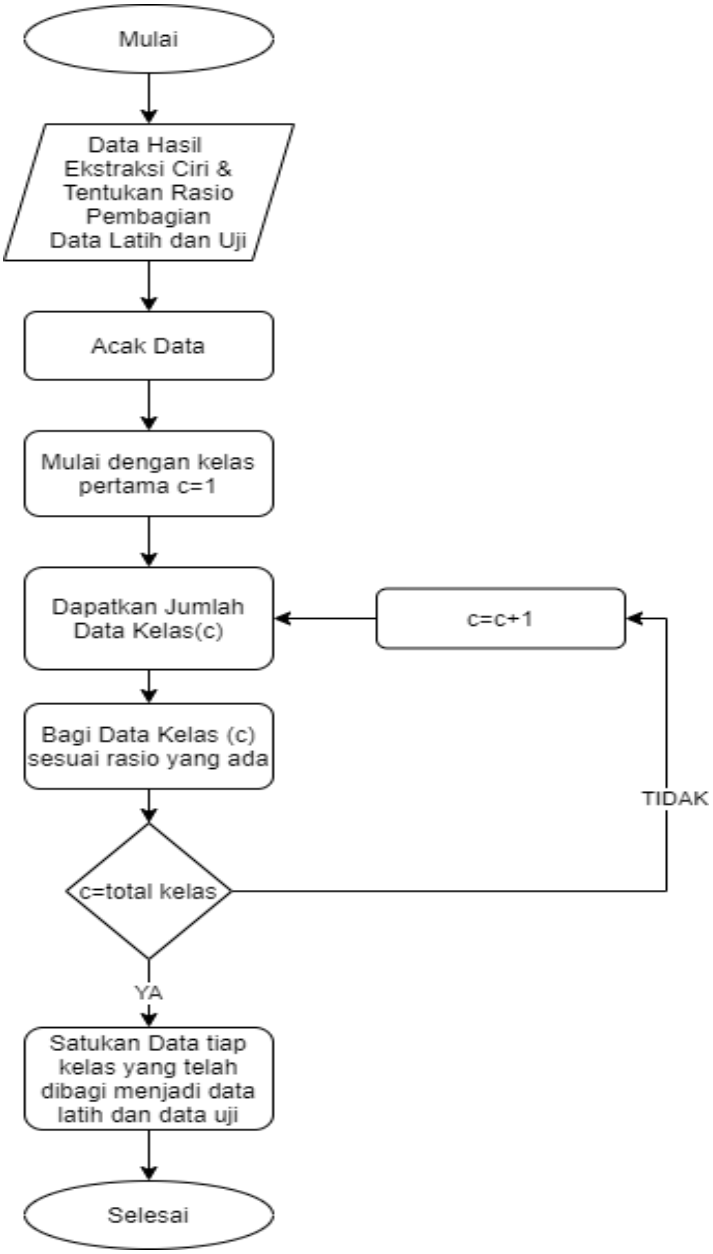
$$s_{maks} = \log_2 m \quad (3.1)$$

3. Mulai dengan s=1
4. Citra dipetakan dalam kotak-kotak dengan ukuran 2^s x2^s.
5. Hitung banyaknya kotak N(s) yang melingkupi suatu objek.
6. Menghitung dimensi fractal D(s) dengan persamaan (2.1)
7. Mengecek nilai s kurang dari nilai s_{maks} atau tidak. Jika s kurang dari s_{maks} maka tambahkan nilai s dengan 1 dan lakukan kembali langkah 4. Jika nilai s_{maks} tidak kurang dari maka ekstraksi ciri telah selesai dilakukan.

Setelah ciri pada data tersebut berhasil diekstraksi, data tersebut kemudian dapat dibagi untuk keperluan permodelan *decision tree*.

3.5 Pembagian Data

Pembagian data dilakukan dengan metode *stratified shuffle split*. Metode ini memungkinkan pembagian data memiliki rasio yang sama setiap kelasnya sehingga pengukuran performansi dan permodelan yang dilakukan lebih baik. Proses pembagian data dengan metode *stratified shuffle split* dapat dilihat pada gambar 3.3. Pembagian data diawali dengan mengambil keseluruhan data ekstraksi ciri sebelumnya dan menentukan nilai rasio antara data latih dan data uji. Data tersebut kemudian diacak. Setelah diacak data dikelompokkan berdasarkan kelasnya, kemudian dimulai dari kelas pertama data dibagi dengan rasio yang telah ditentukan. Langkah ini kemudian dilanjutkan hingga kelas terakhir. Setelah diperoleh data latih dan uji pada setiap kelas, data latih tersebut kemudian disatukan menjadi data latih, sementara itu data uji setiap kelas juga disatukan menjadi data uji.



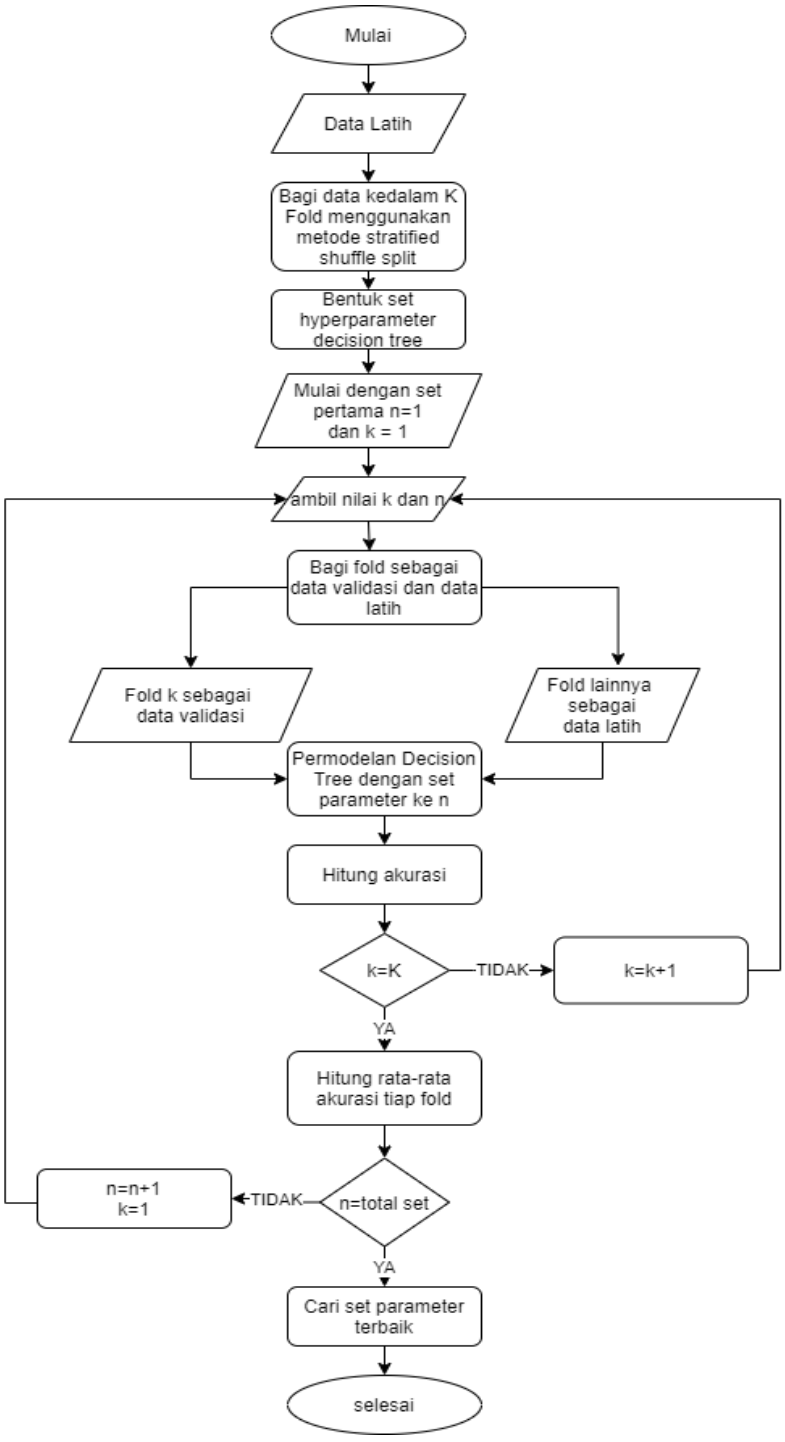
Gambar 3. 3 Diagram alir pembagian data

Setelah data disatukan, data tersebut kemudian dapat digunakan sebagai pembentukan model dan pengujian model pada tahapan berikutnya. Proses pembentukan model dimulai dengan pemilihan model terbaik yang dibahas pada bagian berikutnya.

3.6 Pemilihan model dengan *K-Fold Cross Validation*

Pemilihan model dengan *K-Fold Cross Validation* bertujuan agar model yang dibentuk memiliki performansi yang paling optimal. Pemilihan model dilakukan

dengan mengganti *hyper parameter* pada *decision tree* hingga diperoleh parameter yang menghasilkan *model decision* tree terbaik. Proses pemilihan parameter terbaik ini dilakukan dengan *K-Fold Cross Validation* dengan alur yang dapat dilihat pada gambar 3.4.



Gambar 3. 4 Diagram alir pemilihan model

Data yang dipakai pada percobaan ini merupakan data latih yang kemudian akan dibagi ke dalam *K Fold*. Pembagian data ke dalam *K Fold* dilakukan dengan metode *stratified shuffle split* agar rasio kelas yang ada dalam setiap fold sama hingga menghasilkan hasil yang lebih baik. Setelah data tersebut dibagi kedalam *K fold*, kemudian akan dipilih satu set *hyper parameter* yang ingin diuji menggunakan *K-Fold*. Set itu kemudian digunakan untuk melatih *decision tree* dengan data validasi dari salah satu *fold* kemudian data latih adalah gabungan dari fold lainnya. Percobaan *fold* pertama kemudian dihitung akurasinya hingga diperoleh akurasi Fold pada percobaan dengan *fold* pertama. Setelah percobaan *fold* pertama selesai, dilakukan percobaan dengan menggunakan *fold* kedua sebagai data validasi dan *fold* lain sebagai data latih. Hal tersebut dilakukan secara berulang sampai diperoleh akurasi dari *K fold* tersebut. Setelah diperoleh akurasi dari keseluruhan *fold*. Akurasi tersebut kemudian dirata-ratakan untuk memperoleh performansi dari model *decision tree* dengan set *hyper parameter* pertama. Berikutnya dilakukan percobaan dengan set *hyperparameter* kedua dan seterusnya hingga keseluruhan set *hyperparameter* yang dibuat diuji dengan proses *K-Fold Cross Validation*. Dari keseluruhan set tersebut dipilih satu set dengan akurasi terbaik.

3.7 Permodelan *Decision Tree*

Set *hyperparameter* terbaik yang sudah diperoleh melalui *K-Fold Cross Validation* kemudian digunakan untuk melatih *Decision Tree* dengan keseluruhan data latih yang dimiliki.

Langkah-langkah algoritma *decision tree* adalah sebagai berikut:

1. Tentukan kolom yang digunakan sebagai target dan kolom yang digunakan sebagai attribut.
2. Hitung entropi dari kelas tersebut

   $$E(S) = -\sum_i^C p_i \log_2 p_i \quad (3.2)$$

   Keterangan:

   $E(S)$ = Entropi untuk semesta S

   p_i = proporsi kelas i terhadap seluruh data

   C = total kelas
3. Hitung entropi setiap atribut dan *information gain* dari setiap atribut. Atribut dengan nilai *information gain* terbesar akan digunakan untuk membentuk akar.

$$Gain(S, a) = E(S) - \sum_{v \in T} \frac{|S_v|}{|S|} E(S_v) \quad (3.3)$$

Keterangan:

$Gain(S, a)$ = Information gain attribute a

$E(S_v)$ = Entropy attribut a dengan nilai v

S_v = Semesta attribute a dengan nilai v

S = Semesta keseluruhan data

4. Setelah akar terbentuk, hitung kembali entropi dan *information gain* setiap atribut pada akar tersebut dan pilih atribut dengan *information gain* terbesar sebagai cabang dari akar yang dibuat. Lakukan hal ini hingga keseluruhan data terhitung.

3.8 Pengujian Performansi

Setelah diperoleh model *decision tree* dari data latih yang diberikan, maka performansi model tersebut dapat diuji dengan beberapa nilai seperti *confusion matrix*, akurasi, *precision*, *recall*, dan *f1-score*.

3.8.1 *Confusion Matrix*

Confusion matrix bertujuan untuk membandingkan hasil klasifikasi dari model *decision tree* dengan kelas sebenarnya. *Confusion matrix* akan memberikan nilai *True Positif (TP)*, *True Negatif* (TN), *False Positif* (FP), dan *False Negatif* (FN). True Positif (TP) mengindikasikan jumlah data *Begonia brevirimosa* dan *Begonia lugrae* yang diprediksi benar kelas *Begonia brevirimosa* dan *Begonia lugrae*. True Negatif (TN) mengindikasikan jumlah data yang benar diprediksi bukan *Begonia brevirimosa* dan *Begonia lugrae*. *False* Positif (FP) mengindikasikan mengindikasikan jumlah data yang diprediksi *Begonia brevirimosa* dan *Begonia lugrae* namun sebenarnya bukan *Begonia brevirimosa* dan *Begonia lugrae*. *False* Negatif (FN) mengindikasikan data yang diprediksi bukan *Begonia brevirimosa* dan *Begonia lugrae* namun sebenarnya adalah *Begonia brevirimosa* dan *Begonia lugrae*. Nilai TP, TN, FP, dan FN dapat digunakan untuk perhitungan akurasi, *precision*, *recall*, dan *f1-score*.

3.8.2 Akurasi

Akurasi menunjukkan jumlah prediksi dari *decision tree* yang benar dari keseluruhan prediksi yang dilakukan oleh *decision tree*. Akurasi dapat dihitung dengan menggunakan persamaan 3.4.

$$Akurasi = \frac{TP+TN}{TP+FP+TN+FN} \quad (3.4)$$

3.8.3 *Precision*

Precision menunjukkan performa *decision tree* dalam memprediksi kelas *Begonia brevirimosa* dan *Begonia lugrae* dibanding dengan seluruh prediksi *Begonia brevirimosa* dan *Begonia lugrae*. Persamaan *precision* dapat dilihat pada persamaan 3.5.

$$Presisi = \frac{TP}{TP+FP} \quad (3.5)$$

3.8.4 *Recall*

Recall menunjukkan performa *decision tree* dalam memprediksi kelas *Begonia brevirimosa* dan *Begonia lugrae* dibanding seluruh data *Begonia brevirimosa* dan *Begonia lugrae*. Persamaan recall dapat dilihat pada persamaan 3.6.

$$Recall = \frac{TP}{TP+FN} \quad (3.6)$$

3.8.5 *F1 Score*

F1 Score menujukkan rasio harmonik antara *precision* dan *recall*. Persamaan *F1 score* dapat dilihat pada rumus 3.7.

$$F1\ Score = \frac{2\ x\ Presisi\ \ x\ Recall}{Presisi+Recall} \quad (3.7)$$

BAB IV
PENGUJIAN DAN ANALISIS SISTEM

4.1 Pengujian Sistem

Uji sistem memiliki tujuan untuk mengidentifikasi kekuatan dan kelemahan sistem serta memastikan keberhasilan rancangan program apakah menghasilkan hasil yang diharapkan oleh pengguna dengan menguji efektivitas program. Dalam proses pengujian sistem, terdapat dua faktor yang mendukung keberhasilan sistem. Dua faktor tersebut adalah spesifikasi perangkat keras serta perangkat lunak.

4.2 Spesifikasi Perangkat Keras

Penggunaan perangkat keras berdasar pada kebutuhan yang diperlukan dalam menjalankan sistem, antara lain:

1. Laptop : Lenovo
2. *Processor* : Intel(R) Core(TM) i3-10110U CPU @ 2.10GHz (4 CPUs), ~2.6GHz
3. RAM : 4096MB RAM
4. *Camera* : NIKON COOLPIX S3700

4.3 Spesifikasi Perangkat Lunak

Perangkat lunak sebagai pendukung yang digunakan diantaranya:

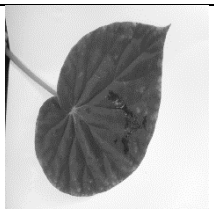
1. Sistem Operasi : Windows 11 Home Single Language 64-bit
2. Aplikasi Pemrograman : Matlab 2021a

4.4 Skenario Pengujian Sistem

Langkah pertama, penulis melakukan *pre-processing* pada jumlah perolehan data latih yaitu sebesar 100 citra dan total 50 citra pada setiap kelas, serta data uji sebanyak 37 citra pada setiap kelas dengan total 74 citra. Langkah ini dilakukan dengan merubah ukuran citra agar memiliki ukuran yang sama. *Resize* yang digunakan pada penelitian ini adalah 512 x 512 piksel, 128 x 128 piksel, 64 x 64 piksel, 32 x 32 piksel, hingga16 x 16 piksel. Setelah itu, langkah berikutnya adalah

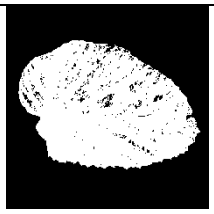
merubah citra rgb menjadi grayscale. Tampilan contoh pengolahan citra meresize dan merubah citra rgb menjadi grayscale ditunjukkan pada Tabel 4.1 berikut ini.

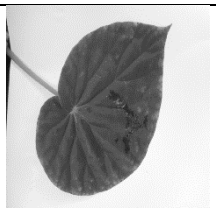
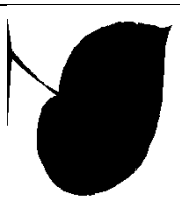
Tabel 4.1. Tampilan contoh pengolahan citra meresize dan merubah citra rgb menjadi grayscale.

No	Citra RGB	Hasil Resize	Citra Grayscale
1			
2			

Citra grayscale hasil pengolahan selanjutnya dikonversi menjadi citra biner. Pada proses ini dihasilkan objek daun dengan warna hitam dan background dengan warna putih. Karena dalam pengolahan citra, objek harus yang berwarna putih (terlabeli sebagai nilai piksel 1) dan background berwarna hitam (terlabeli sebagai nilai piksel 0), maka perlu dilakukan operasi komplemen. Tampilan contoh pengolahan citra merubah citra grayscale menjadi citra biner dan operasi komplemen ditunjukkan pada Tabel 4.2 berikut ini.

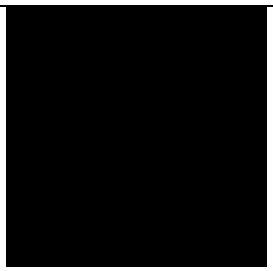
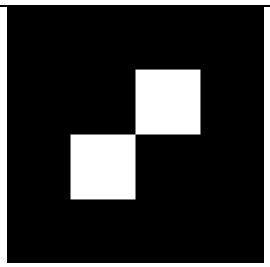
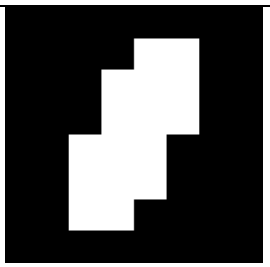
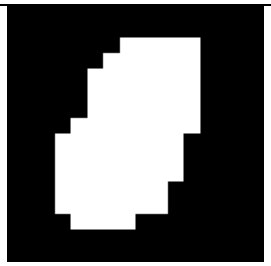
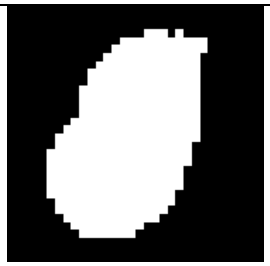
Tabel 4.2. Tampilan contoh pengolahan citra merubah citra grayscale menjadi citra biner dan operasi komplemen

No	Citra Grayscale	Citra Biner	Hasil Komplemen
1			

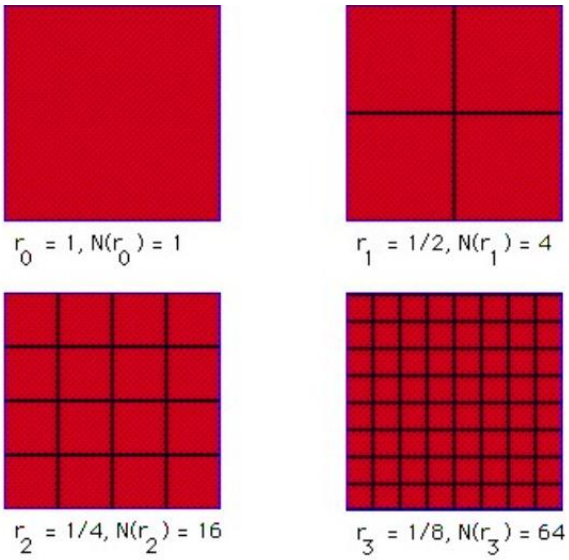
2			

Pada penelitian ini ekstraksi ciri dilakukan dengan mengekstrak ciri warna rgb dan ciri fraktal. Metode *Box Counting* dilakukan guna menentukan dimensi fraktal dengan nilai dimensi fraktal sebesar 2. Tampilan contoh penerapan *box counting* ditunjukkan pada Tabel 4.3.

Tabel 4.3. Tampilan contoh penerapan *box counting*

iterasi 1	iterasi 2	iterasi 3
		
iterasi 4	iterasi 5	iterasi 6
		
iterasi 7	iterasi 8	iterasi 9
		

Ekstraksi ciri fraktal dihitung dari citra hasil *box counting* berdasarkan tiga nilai yaitu *Empty Boxes*, *Full Boxes*, dan *Fractal Dimension. Empty Boxes* dihitung berdasarkan jumlah kotak penyusun objek berwarna hitam. *Full Boxes* dihitung berdasarkan jumlah kotak penyusun objek berwarna putih. Sedangkan *Fractal Dimension* dihitung menggunakan persamaan 2.2. Untuk panjang sisi r yang berbeda, N(r), yaitu jumlah terkecil kotak dengan panjang sisi r yang diperlukan untuk menutupi bentuk tersebut. Jika bentuknya 1-dimensi, seperti segmen garis, maka nilai N(r) = 1/r. Jika bentuknya 2-dimensi, seperti persegi satuan yang terisi, maka nilai N(r) = (1/r)^2. Jika bentuknya 3-dimensi, seperti kubus satuan yang terisi, nilainya akan menjadi N(r) = (1/r)^3.



Metode ini melibatkan pembagian objek menjadi sejumlah kotak persegi dengan ukuran yang bervariasi (r). Selanjutnya, jumlah kotak yang diperlukan untuk menutupi objek tersebut, di mana jika suatu garis dibagi menjadi bagian yang sama, setiap bagian akan memiliki rasio r = 1 / N. Dalam contoh ini, N(s) mewakili jumlah kotak dengan panjang sisi r yang digunakan untuk menutupi bidang. Sementara itu, r adalah ukuran kotak yang digunakan, N(r) adalah jumlah kotak dengan ukuran r yang diperlukan, dan D(r) adalah dimensi fraktal dengan kotak berukuran r. Dengan demikian, dimensi Box Counting dapat didefinisikan dalam sebuah persamaan. Adapun penurunan rumus dari persamaan 2.2 adalah sebagai berikut

$$\log(N) = \log(M^D)$$

$$\log(M) = D \log M$$

$$\frac{\log N}{\log M} = \text{D}$$

$$\frac{\log N(r)}{\log(\frac{1}{r})} = \text{D}$$

dengan D adalah dimensi fractal, N(r) adalah jumlah kotak secara keseluruhan dan r adalah satuan kotak terkecil. Sehingga contoh perhitungan sederhana dimensi fractal pada penelitian ini dapat dilihat pada persaman berikut

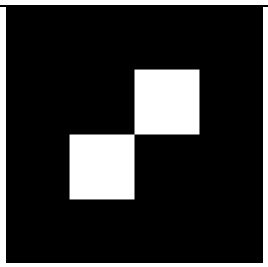
$$\text{D} = \frac{\log (N)}{\log\ (\frac{1}{r})}$$

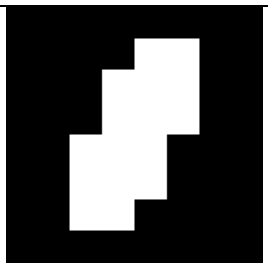
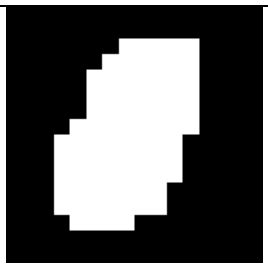
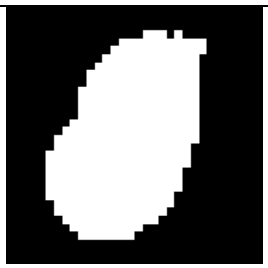
$$\text{D} = \frac{\log (16)}{\log (4)}$$

$$\text{D}= 2$$

dengan nilai dimensi fractal berkisar di nilai D=2. Untuk nilai lebih lengkap dapat dilihat pada table 4.4. Contoh hasil perhitungan ekstraksi ciri fraktal ditunjukkan pada Tabel 4.4 berikut ini [12].

Tabel 4.4. Contoh hasil perhitungan ekstraksi ciri fraktal

Iterasi	**Citra Biner**	***Empty Boxes***	***Full Boxes***	***Fractal Dimension***
1		4	0	0
2		14	2	1.8074

3		48	16	1.7925
4		174	82	1.8106
5		656	368	1.8351
6		2512	1584	1.8566
7		9778	6606	1.8755
8		38557	26979	1.892

9		153055	109089	1.9062

Sedangkan ciri warna rgb dihitung dari nilai rata-rata citra rgb juga berdasarkan tiga nilai yaitu *Red*, *Green*, dan *Blue* seperti pada contoh perhitungan berikut ini.

R = 255 G = 255 B = 0	R = 255 G = 102 B = 0	R = 0 G = 255 B = 0
R = 0 G = 255 B = 255	R = 204 G = 102 B = 204	R = 255 G = 255 B = 255
R = 0 G = 0 B = 0	R = 102 G = 255 B = 204	R = 153 G = 102 B = 51

Red = (255+255+0+0+204+255+0+102+153)/9 = 136

Green = (255+102+255+255+102+255+0+255+102)/9 = 175.6666

Blue = (0+0+0+255+204+255+0+204+51)/9 = 107.6666

Tampilan contoh hasil ekstraksi terhadap data latih ditunjukkan pada Tabel 4.5.

Tabel 4.5. Tampilan contoh hasil ekstraksi terhadap data latih

No	***Empty Boxes***	***Full Boxes***	***Fractal Dimen-sion***	***Red***	***Green***	***Blue***	**Kelas Actual**
1	160779	101365	1.8956	146.7044	81.8556	89.4250	brevirimosa
2	164330	97814	1.8989	137.8977	74.7586	83.4577	brevirimosa
3	164672	97472	1.9001	146.5208	80.0922	87.4669	brevirimosa
4	161720	100424	1.8964	144.4765	79.7288	87.4749	brevirimosa

5	158027	104117	1.8913	146.8494	80.6527	87.7446	brevirimosa
6	162284	99860	1.8988	162.9140	94.8704	100.3768	brevirimosa
7	169379	92765	1.9070	137.7690	73.9076	82.1830	brevirimosa
8	167881	94263	1.9048	140.5227	76.0547	83.5909	brevirimosa
9	162721	99423	1.8995	153.3609	86.0121	92.6786	brevirimosa
10	163864	98280	1.9015	156.5271	88.8242	95.3972	brevirimosa
11	175400	86744	1.9135	158.6490	88.1387	95.0488	brevirimosa
12	168325	93819	1.9045	137.6933	72.6892	80.6368	brevirimosa
13	165959	96185	1.9012	137.3345	73.1158	81.7973	brevirimosa
14	163191	98953	1.8978	132.3788	69.0893	77.3821	brevirimosa
15	157701	104443	1.8912	147.5180	79.0094	86.0894	brevirimosa
16	172821	89323	1.9199	84.4129	33.0508	36.2837	brevirimosa
17	175667	86477	1.9145	132.0579	64.5111	67.9936	brevirimosa
18	186163	75981	1.9317	184.3584	129.7763	131.2174	brevirimosa
19	189488	72656	1.9337	173.8731	112.9111	115.9785	brevirimosa
20	183956	78188	1.9243	105.1678	45.8682	48.6879	brevirimosa
21	186325	75819	1.9270	104.4593	43.8564	46.7402	brevirimosa
22	187332	74812	1.9321	97.1253	40.2949	43.2024	brevirimosa
23	190601	71543	1.9342	102.2940	44.1042	47.0430	brevirimosa
24	195156	66988	1.9395	174.1587	112.4776	114.3262	brevirimosa
25	195934	66210	1.9400	173.4862	109.6970	112.3471	brevirimosa
26	187698	74446	1.9321	97.0794	41.3772	44.4711	brevirimosa
27	186028	76116	1.9279	99.1279	41.8931	44.7421	brevirimosa
28	194612	67532	1.9383	97.5081	41.7754	44.4630	brevirimosa
29	193615	68529	1.9368	100.1897	43.0947	45.8358	brevirimosa
30	193859	68285	1.9381	96.6602	41.0416	44.4858	brevirimosa
31	193382	68762	1.9375	88.8993	36.2015	39.1451	brevirimosa
32	195213	66931	1.9409	91.9215	38.1187	41.3291	brevirimosa
33	196367	65777	1.9419	173.3355	116.8283	120.4646	brevirimosa
34	192653	69491	1.9357	166.2674	100.1903	100.8111	brevirimosa
35	212809	49335	1.9591	76.1053	30.0433	33.6408	brevirimosa
36	210505	51639	1.9567	76.4199	30.4265	34.6266	brevirimosa
37	211025	51119	1.9572	75.1860	29.6056	33.7735	brevirimosa
38	210832	51312	1.9568	74.7586	29.2433	33.4147	brevirimosa
39	210607	51537	1.9568	75.3294	29.4580	33.8158	brevirimosa
40	209941	52203	1.9560	76.4902	30.3720	34.3732	brevirimosa
41	208468	53676	1.9526	73.3698	28.5212	32.4866	brevirimosa
42	207240	54904	1.9532	74.9046	29.2317	33.1527	brevirimosa
43	207325	54819	1.9531	78.5807	31.1095	35.1462	brevirimosa
44	207333	54811	1.9532	79.9014	31.8969	35.7683	brevirimosa
45	207227	54917	1.9532	75.9365	29.9179	33.7606	brevirimosa
46	207290	54854	1.9523	79.8068	32.0699	36.3319	brevirimosa
47	207191	54953	1.9522	84.4292	34.6656	38.1977	brevirimosa

48	207308	54836	1.9522	83.1794	34.0801	38.1010	brevirimosa
49	207309	54835	1.9513	83.7431	34.3947	38.2243	brevirimosa
50	208100	54044	1.9522	85.4726	35.4006	39.1710	brevirimosa
51	153055	109089	1.9062	74.8418	97.7056	39.5763	lugrae
52	158124	104020	1.9035	70.4891	92.8357	36.8990	lugrae
53	150335	111809	1.8997	72.4769	94.8237	37.6446	lugrae
54	152706	109438	1.9020	69.7851	91.3138	37.1470	lugrae
55	142472	119672	1.8902	79.3642	98.7012	42.5911	lugrae
56	142305	119839	1.8921	79.4655	100.4208	42.6424	lugrae
57	135383	126761	1.8900	82.0652	106.3044	44.5368	lugrae
58	157987	104157	1.9060	69.4042	90.7330	35.9606	lugrae
59	169482	92662	1.9196	62.2446	82.2980	31.4095	lugrae
60	171970	90174	1.9148	62.3765	84.7403	27.3748	lugrae
61	170123	92021	1.9112	59.2093	80.9099	25.3363	lugrae
62	171492	90652	1.9226	54.5559	81.0674	21.8682	lugrae
63	171002	91142	1.9271	55.5553	79.7534	26.1309	lugrae
64	175095	87049	1.9263	58.2697	83.3005	27.2463	lugrae
65	175334	86810	1.9262	58.8501	84.1524	28.3755	lugrae
66	174116	88028	1.9250	62.8091	87.2247	29.5985	lugrae
67	169389	92755	1.9111	61.5333	85.5081	32.2206	lugrae
68	162966	99178	1.9042	55.8151	83.0062	25.9193	lugrae
69	162009	100135	1.9081	57.3698	84.6339	25.6195	lugrae
70	169831	92313	1.9187	56.0621	81.4063	23.6262	lugrae
71	170378	91766	1.9193	56.9074	82.6334	24.6166	lugrae
72	163319	98825	1.9080	54.6040	76.2246	24.9799	lugrae
73	153974	108170	1.8939	45.7628	61.0059	20.9567	lugrae
74	154927	107217	1.8903	50.1007	67.0387	23.4592	lugrae
75	120933	141211	1.8476	94.5821	119.4046	44.9112	lugrae
76	123027	139117	1.8606	106.9975	129.9807	57.5690	lugrae
77	127917	134227	1.8633	110.1884	132.8794	61.3325	lugrae
78	123613	138531	1.8546	107.2578	128.6224	60.1630	lugrae
79	122772	139372	1.8601	103.9630	125.7653	55.4996	lugrae
80	128576	133568	1.8723	109.0754	131.2388	59.6275	lugrae
81	125615	136529	1.8569	110.1068	132.2981	60.7072	lugrae
82	126428	135716	1.8674	107.5205	128.3393	58.9208	lugrae
83	133574	128570	1.8764	94.0715	113.9656	48.7177	lugrae
84	130196	131948	1.8685	105.0998	127.3897	58.3973	lugrae
85	128736	133408	1.8717	100.2590	120.4732	54.7892	lugrae
86	126366	135778	1.8678	104.5973	125.6230	57.9928	lugrae
87	128832	133312	1.8731	100.7886	121.1686	54.6450	lugrae
88	128786	133358	1.8695	105.5781	125.1013	59.7228	lugrae
89	125990	136154	1.8653	95.4984	116.2467	48.3098	lugrae
90	120321	141823	1.8558	106.1287	127.1441	59.0318	lugrae

91	131272	130872	1.8761	101.1356	121.4380	55.2187	lugrae
92	126135	136009	1.8648	104.8299	125.3071	55.5313	lugrae
93	117070	145074	1.8534	105.2057	125.9064	54.0763	lugrae
94	111886	150258	1.8478	111.6710	138.4933	58.7152	lugrae
95	119183	142961	1.8551	97.3301	119.1648	46.7398	lugrae
96	102475	159669	1.8322	107.3839	130.6325	54.5026	lugrae
97	125295	136849	1.8563	104.0499	127.8707	52.3179	lugrae
98	119384	142760	1.8498	106.4928	130.1394	54.0795	lugrae
99	122150	139994	1.8529	95.6992	116.8829	45.5231	lugrae
100	123832	138312	1.8564	110.3408	133.0661	55.6146	lugrae

Setelah tahap ekstraksi selesai, citra kemudian diklasifikasi menggunakan *decision tree* untuk menentukan kesesuaian data latih serta data uji. Perhitungan pembentukan *decision tree* ditunjukkan pada contoh berikut ini.

Tabel 4.6. Contoh data latih ciri hasil ekstraksi

No	***Empty Boxes***	***Full Boxes***	***Fractal Dimension***	**Kelas Actual**
1	160779	101365	1.8956	brevirimosa
2	164330	97814	1.8989	brevirimosa
3	153055	109089	1.9062	lugrae
4	158124	104020	1.9035	lugrae

Proses untuk node 1 (node akar)
Hitung entropy untuk node akar (semua data) terhadap komposisi kelas

$$E(semua) = -\big((p(brevirimosa|semua) \times \log_2 p(brevirimosa|semua)) + (p(lugrae|semua) \times \log_2 p(lugrae|semua))\big)$$

$$= -\left(\left(\left(\frac{2}{4}\right) \times \log_2 \left(\frac{2}{4}\right)\right) + \left(\left(\frac{2}{4}\right) \times \log_2 \left(\frac{2}{4}\right)\right)\right)$$

$$= 1$$

- **Pemecahan parameter '*Empty Boxes*' di node akar**

Urutkan nilai parameter *Empty Boxes*

Tabel 4.7. Nilai parameter *Empty Boxes* yang sudah diurutkan

No	***Empty Boxes***	**Kelas Actual**
1	153055	lugrae
2	158124	lugrae
3	160779	brevirimosa
4	164330	brevirimosa

Cari urutan data yang memiliki kelas berbeda

Pemecahan I (data ke-2 dan 3)

$$v = \frac{x_2 + x_3}{2}$$
$$= \frac{158124 + 160779}{2}$$
$$= \frac{318903}{2}$$
$$= 159451.5$$

Tabel 4.8. Pemecahan parameter '*Empty Boxes*' di node akar

Empty Boxes	**159451.5**	
	<	>=
brevirimosa	0	2
lugrae	2	0
Gain	1	

$$E(semua_{<159451.5}) = -((p(\text{brevirimosa}| < 159451.5) \times \log_2 p(\text{brevirimosa}| < 159451.5)) + (p(lugrae| < 159451.5) \times \log_2 p(lugrae| < 159451.5)))$$
$$= -((0/2) \times \log_2(0/2) + (2/2) \times \log_2(2/2))$$
$$= 0$$

$$E(semua_{\geq 159451.5}) = -((p(\text{brevirimosa}| \geq 159451.5) \times \log_2 p(\text{brevirimosa}| \geq 159451.5)) + (p(lugrae| \geq 159451.5) \times \log_2 p(lugrae| \geq 159451.5)))$$
$$= -((2/2) \times \log_2(2/2) + (0/2) \times \log_2(0/2))$$
$$= 0$$

$$G(semua, 159451.5) = E(semua) - ((p(< 159451.5|semua) \times E(semua_{<159451.5}))) + (p(\geq 159451.5|semua) \times E(semua_{\geq 159451.5})))$$
$$= 1 - (((2/4) \times (0)) + ((2/4) \times (0)))$$
$$= 1$$

- **Pemecahan parameter '*Full Boxes*' di node akar**

Urutkan nilai parameter *Full Boxes*

Tabel 4.9. Nilai parameter *Full Boxes* yang sudah diurutkan

No	***Full Boxes***	**Kelas Actual**
1	97814	brevirimosa
2	101365	brevirimosa

3	104020	lugrae
4	109089	lugrae

Cari urutan data yang memiliki kelas berbeda

Pemecahan I (data ke-2 dan 3)

$$v = \frac{x_2 + x_3}{2}$$
$$= \frac{158124 + 160779}{2}$$
$$= \frac{205385}{2}$$
$$= 102692.5$$

Tabel 4.10. Pemecahan parameter '*Full Boxes*' di node akar

Full Boxes	**102692.5**	
	<	>=
brevirimosa	2	0
lugrae	0	2
Gain	1	

$$E(semua_{<102692.5}) = -((p(\text{brevirimosa}| < 102692.5) \times \log_2 p(\text{brevirimosa}| < 102692.5)) + (p(lugrae| < 102692.5) \times \log_2 p(lugrae| < 102692.5)))$$
$$= -((2/2) \times \log_2(2/2) + (0/2) \times \log_2(0/2))$$
$$= 0$$

$$E(semua_{\geq 102692.5}) = -((p(\text{brevirimosa}| \geq 102692.5) \times \log_2 p(\text{brevirimosa}| \geq 102692.5)) + (p(lugrae| \geq 102692.5) \times \log_2 p(lugrae| \geq 102692.5)))$$
$$= -((0/2) \times \log_2(0/2) + (2/2) \times \log_2(2/2))$$
$$= 0$$

$$G(semua, 102692.5) = E(semua) - ((p(< 102692.5|semua) \times E(semua_{<102692.5}))) + (p(\geq 102692.5|semua) \times E(semua_{\geq 102692.5})))$$
$$= 1 - (((2/4) \times (0)) + ((2/4) \times (0)))$$
$$= 1$$

- **Pemecahan parameter '*Fractal Dimension*' di node akar**

Urutkan nilai parameter *Fractal Dimension*

Tabel 4.11. Nilai parameter *Fractal Dimension* yang sudah diurutkan

No	***Fractal Dimension***	**Kelas Actual**

1	1.8956	brevirimosa
2	1.8989	brevirimosa
3	1.9035	lugrae
4	1.9062	lugrae

Cari urutan data yang memiliki kelas berbeda

Pemecahan I (data ke-2 dan 3)

$$v = \frac{x_2 + x_3}{2}$$
$$= \frac{1.8989 + 1.9035}{2}$$
$$= \frac{3.8024}{2}$$
$$= 1.9012$$

Tabel 4.12. Pemecahan parameter '*Fractal Dimension*' di node akar

Fractal Dimension	**1.9012**	
	<	>=
brevirimosa	2	0
lugrae	0	2
Gain	1	

$$E(semua_{<1.9012}) = -((p(\text{brevirimosa}| < 1.9012) \times \log_2 p(\text{brevirimosa}| < 1.9012)) + (p(lugrae| < 1.9012) \times \log_2 p(lugrae| < 1.9012)))$$
$$= -((2/2) \times \log_2(2/2) + (0/2) \times \log_2(0/2))$$
$$= 0$$

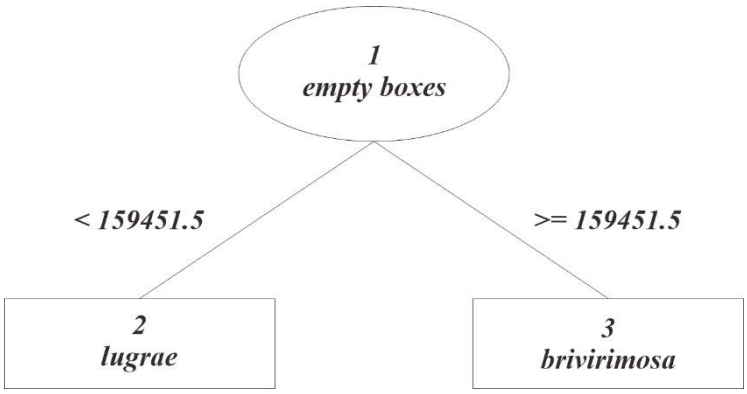
$$E(semua_{\geq 1.9012}) = -((p(\text{brevirimosa}| \geq 1.9012) \times \log_2 p(\text{brevirimosa}| \geq 1.9012)) + (p(lugrae| \geq 1.9012) \times \log_2 p(lugrae| \geq 1.9012)))$$
$$= -((0/2) \times \log_2(0/2) + (2/2) \times \log_2(2/2))$$
$$= 0$$

$$G(semua, 1.9012) = E(semua) - ((p(< 1.9012|semua) \times E(semua_{<1.9012})) + (p(\geq 1.9012|semua) \times E(semua_{\geq 1.9012}))$$
$$= 1 - (((2/4) \times (0)) + ((2/4) \times (0)))$$
$$= 1$$

Tabel 4.13. Hasil perhitungan entropy dan gain untuk node akar

Node			Jumlah	Brevirimosa	Lugrae	Entropy	Gain
1	Total		4	2	2		
2	*Empty Boxes*	< 159451.5 >= 159451.5	2 2	0 2	2 0	0 0	1
3	*Full Boxes*	< 102692.5 >= 102692.5	2 2	2 0	0 2	0 0	1
4	*Fractal Dimension*	< 1.9012 >= 1.9012	2 2	2 0	0 2	0 0	1

Nilai gain tertinggi akan dijadikan sebagai node akar, namun karena hasil yang didapat pada Tabel 4.13 menunjukkan bahwa parameter *empty boxes, full boxes,* dan *fractal dimension* memiliki gain yang sama, maka diambil salah satu parameter untuk dijadikan sebagai node akar. Pada contoh ini parameter *empty boxes* dijadikan sebagai node akar dan gambar pohon keputusannya ditunjukkan pada Gambar 4.1 berikut ini.



Gambar 4.1. Contoh pohon keputusan hasil perhitungan

Tahap terakhir dari pengujian ini yaitu mancari nilai terbaik dari capaian tingkat akurasi, *precision, recall, f1 score* dan waktu komputasi diantara parameter yang digunakan.

Metode fraktal adalah teknik klasifikasi tanaman yang didasarkan pada analisis fraktal dari gambar tanaman. Fraktal adalah sebuah objek matematika yang memiliki pola-pola yang sama pada berbagai skala ukuran, sehingga dapat digunakan untuk mengukur kompleksitas dan tekstur dari gambar tanaman. Metode fraktal dapat digunakan untuk mengklasifikasikan tanaman berdasarkan karakteristik fraktal yang terdapat pada gambar tanaman. Beberapa langkah yang penulis lakukan dalam mengklasifikasi tanaman antara lain:

Data gambar tanaman perlu dipersiapkan terlebih dahulu dengan mengambil citra gambar tanaman dari berbagai jenis tanaman yang berbeda. Citra gambar tanaman ini kemudian akan diolah untuk mendapatkan hasil yang dibutuhkan untuk klasifikasi. Pada langkah ini, gambar tanaman akan diolah dengan menggunakan algoritma fraktal dan *decision tree*. Data fraktal yang diperoleh dari langkah

sebelumnya kemudian akan diseleksi untuk mendapatkan fitur yang paling relevan dalam klasifikasi tanaman. Fitur yang dipilih dapat berupa nilai dimensi fraktal.

Pada penelitian ini, penulis menggunakan *decision tree* sebagai algoritma klasifikasi karena memiliki keunggulan dalam menyajikan aturan-aturan keputusan yang mudah dipahami dan diinterpretasikan oleh manusia. Dalam *decision tree*, setiap cabang dan simpul mewakili keputusan yang mudah dijelaskan secara logis. Di samping hal itu, *decision tree* dapat bekerja dengan baik pada data yang besar dan kompleks. Algoritma ini dapat memproses data dengan cepat karena struktur *decision tree* memungkinkan untuk melakukan pemrosesan secara parallel.

4.5 Parameter Data Citra

Tabel 4.14 Jumlah Data Latih dan Data Uji

	Jenis Tanaman	**Jumlah Gambar**
Data Latih	*Begonia brevirimosa*	50
	Begonia lugrae	50
Data Uji	*Begonia brevirimosa*	37
	Begonia lugrae	37

Pre-processing dataset dalam pengolahan citra untuk klasifikasi adalah tahap penting dalam mempersiapkan data sebelum dilakukan pengolahan lebih lanjut. Langkah-langkah *pre-processing* ini bertujuan untuk meningkatkan kualitas data citra, mengurangi kebisingan, menyesuaikan kontras, serta menghilangkan gangguan yang tidak relevan. Dengan melaksanakan *pre-processing* dengan baik, dataset menjadi lebih representatif, sehingga algoritma klasifikasi dapat bekerja lebih efisien dan menghasilkan prediksi yang lebih akurat. *Pre-processing* dataset melibatkan proses seperti membersihkan data citra, normalisasi, peningkatan kontras, filterisasi, pemotongan, dan lain sebagainya. Dengan melakukan *pre-processing* dataset secara teliti, kita dapat memastikan bahwa data yang digunakan untuk klasifikasi citra memiliki kualitas optimal, sehingga memaksimalkan potensi algoritma klasifikasi yang akan digunakan.

Manfaat dataset yang telah melalui tahap *pre-processing* sangat signifikan dalam pengolahan citra untuk klasifikasi. Dataset yang telah disiapkan dengan baik memungkinkan algoritma klasifikasi untuk mempelajari pola dan fitur penting secara lebih efektif. *Pre-processing* membantu mengurangi kebisingan atau gangguan yang dapat mempengaruhi performa algoritma klasifikasi. Selain itu, dengan menyesuaikan kontras dan melakukan normalisasi, dataset menjadi lebih seragam dan lebih mudah diinterpretasikan oleh algoritma. Hal ini membantu algoritma untuk mengenali perbedaan yang signifikan antara kategori atau kelas yang ingin diklasifikasikan. Dengan kata lain, *pre-processing* dataset merupakan langkah penting dalam memastikan data yang berkualitas tinggi, yang pada gilirannya meningkatkan akurasi dan keandalan hasil klasifikasi citra.

4.6 Skenario Pelatihan Sistem Pertama

Hasil Pelatihan pada skenario pertama dapat dilihat pada Gambar berikut

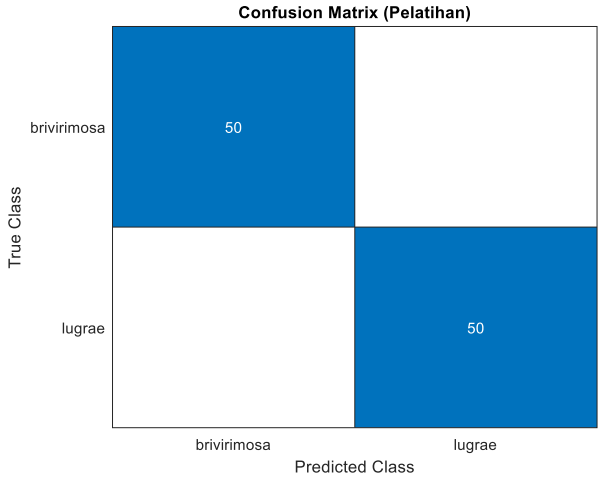
Accuracy = *100 %*

Precision = *100%*

Recall = *100%*

F1 score = *100%*

Waktu Komputasi 115,29 detik.



Gambar 4.2 *Traning Size Begonia Brevirimosa 512 x 512 pixels*

4.7 Analisis Skenario Pelatihan Sistem Pertama

Dalam rumus tersebut, "Jumlah prediksi yang benar" mengacu pada jumlah prediksi yang sesuai dengan label kelas yang sebenarnya, sedangkan "Jumlah total data" adalah total jumlah data yang dievaluasi. Misalnya, jika kita memiliki 50 data dan model memprediksi dengan benar 49 dari data tersebut, maka rumus akurasi akan menjadi:

$$Akurasi = \frac{TP + TN}{TP + FP + TN + FN} = \frac{50 + 50}{50 + 0 + 50 + 0} = \frac{100}{100} = 100\%$$

Akurasi memberikan persentase sejauh mana model melakukan prediksi yang benar terhadap data yang dievaluasi. Semakin tinggi nilai akurasi, semakin baik kinerja model dalam melakukan prediksi. Namun, penting juga untuk memperhatikan bahwa akurasi dapat memberikan hasil yang bias jika ada ketidakseimbangan dalam jumlah kelas dalam data.

Ada beberapa hal yang bisa memberikan pengaruh pada nilai akurasi yang tinggi dalam evaluasi model atau algoritma. Beberapa hal yang bisa berpengaruh pada nilai akurasi diantaranya (1) pembagian data menjadi set pelatihan dan set pengujian dapat membantu menghindari overfitting dan memberikan estimasi yang lebih baik tentang akurasi model. (2) Penanganan ketidakseimbangan kelas: Jika data memiliki ketidakseimbangan antara jumlah sampel di setiap kelas, hal ini dapat mempengaruhi akurasi. Penerapan teknik penanganan ketidakseimbangan kelas seperti oversampling atau undersampling dapat membantu meningkatkan akurasi dengan mengatasi bias yang dihasilkan oleh ketidakseimbangan tersebut

$$Presisi = \frac{TP}{TP + FP} = \frac{50}{50 + 0} = \frac{50}{50} = 100\%$$

Beberapa faktor yang dapat mempengaruhi *precision* dalam metode klasifikasi adalah:

1. Kesalahan *False Positive*: Jumlah *False Positive* yang tinggi akan mengurangi *precision*. False Positive terjadi ketika model salah mengklasifikasikan contoh negatif sebagai positif. Semakin rendah tingkat kesalahan *False Positive*, semakin tinggi *precision*.
2. Jumlah True Positive: Semakin banyak *True Positive*, yaitu jumlah contoh yang secara benar diklasifikasikan sebagai positif oleh model, semakin tinggi *precision*. Jika model dapat mengenali dan mengklasifikasikan lebih banyak contoh positif dengan benar, *precision* akan meningkat.
3. Jumlah *False Negative*: Meskipun *False Negative* tidak langsung mempengaruhi *precision*, namun dapat berdampak pada *precision* secara tidak langsung melalui *False Positive*. Jika model gagal mengenali contoh positif dan mengklasifikasikannya sebagai negatif (*False Negative*), hal ini dapat menyebabkan peningkatan kesalahan False Positive, yang akan menurunkan *precision*.

$$Recall = \frac{TP}{TP + FN} = \frac{50}{50 + 0} = \frac{50}{50} = 100\%$$

Adapun faktor yang mempengaruhi recall dalam klasifikasi diantaranya:

1. Kesalahan *False Negative*: *Recall* terpengaruh secara negatif oleh jumlah *False Negative* yang tinggi. *False Negative* terjadi ketika model gagal mengklasifikasikan contoh positif sebagai positif, dan secara keliru mengklasifikasikannya sebagai negatif. Semakin rendah tingkat kesalahan *False Negative*, semakin tinggi *recall*.
2. Jumlah *True Positive*: *Recall* meningkat seiring dengan peningkatan jumlah *True Positive*, yaitu jumlah contoh positif yang secara benar diklasifikasikan sebagai positif oleh model. Jika model dapat mengenali lebih banyak contoh positif dengan benar, *recall* akan meningkat.
3. Jumlah *False Positive*: Meskipun *False Positive* tidak secara langsung mempengaruhi *recall*, namun dapat berdampak secara tidak langsung melalui *False Negative*. Jika model mengklasifikasikan contoh negatif sebagai positif dengan kesalahan (*False Positive*), ini dapat mengakibatkan penurunan jumlah *False Negative* dan peningkatan *recall.*

$$F1\ Score = \frac{2\ x\ Presisi\ x\ Recall}{Presisi + Recall} = \frac{2\ x\ 100\%\ x\ 100\%}{100\% + 100\%} = \frac{200\%}{200\%} = 100\%$$

Nilai *F1 Score* secara langsung didapatkan dari nilai *precision* dan *recall*. Ketidakseimbangan jumlah contoh antara kelas positif dan negatif juga dapat mempengaruhi *recall*. Jika jumlah contoh positif jauh lebih besar daripada negatif, model cenderung menghasilkan lebih banyak *False Negative*, yang dapat menurunkan *recall.* Dalam situasi ketidakseimbangan kelas, perlu mempertimbangkan metrik evaluasi lain seperti *F1-score* yang memadukan *recall* dan *precision.*

4.8 Skenario Pengujian Sistem Kedua

Hasil Pengujian dari skenario pertama dapat dilihat pada gambar berikut.

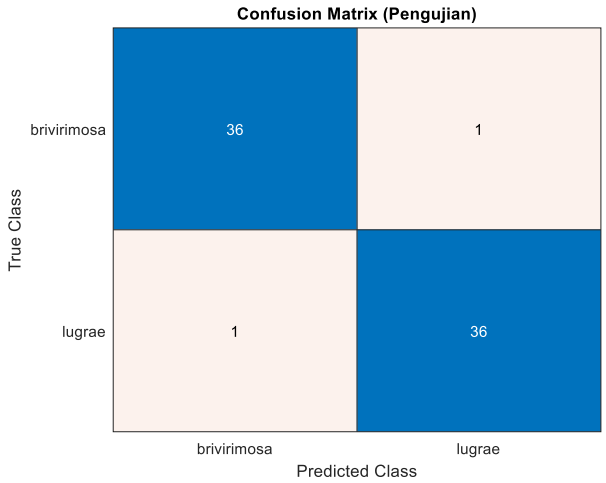
Accuracy = 97.2973%

precision = 97.2973%

recall = 97.2973%

f1 score = 97.2973%

Waktu Komputasi 50,16 detik.



Gambar 4.3 *Testing Size Begonia brevirimosa* 256 x 256 *pixels*

4.9 Analisis Sistem Kedua

Dalam rumus tersebut, “Jumlah prediksi yang benar” mengacu pada jumlah prediksi yang sesuai dengan label kelas yang sebenarnya, sedangkan “Jumlah total data” adalah total jumlah data yang dievaluasi. Misalnya, jika kita memiliki 37 data dan model memprediksi dengan benar 37 dari data tersebut, maka rumus *precision* akan menjadi

$$Akurasi = \frac{TP + TN}{TP + FP + TN + FN} = \frac{36 + 36}{36 + 1 + 36 + 1} = \frac{72}{74} = 97\%$$

Akurasi memberikan persentase sejauh mana model melakukan prediksi yang benar terhadap data yang dievaluasi. Semakin tinggi nilai akurasi, semakin baik kinerja model dalam melakukan prediksi. Namun, penting juga untuk

memperhatikan bahwa akurasi dapat memberikan hasil yang bias jika ada ketidakseimbangan dalam jumlah kelas dalam data.

Ada beberapa hal yang bisa berpengaruh pada nilai akurasi yang tinggi dalam evaluasi model atau algoritma. Beberapa hal yang dapat berpengaruh pada nilai akurasi antara lain (1) pembagian data menjadi set pelatihan dan set pengujian dapat membantu menghindari overfitting dan memberikan estimasi yang lebih baik tentang akurasi model. (2) Penanganan ketidakseimbangan kelas: Jika data memiliki ketidakseimbangan antara jumlah sampel di setiap kelas, hal ini dapat mempengaruhi akurasi. Penerapan teknik penanganan ketidakseimbangan kelas seperti oversampling atau undersampling dapat membantu meningkatkan akurasi dengan mengatasi bias yang dihasilkan oleh ketidakseimbangan tersebut

$$Presisi = \frac{TP}{TP + FP} = \frac{37}{37 + 0} = \frac{37}{37} = 100\%$$

Beberapa yang dapat mempengaruhi *precision* dalam metode klasifikasi adalah:

1. Jumlah *False Positive* berbanding terbalik dengan *precision*. *False Positive* terjadi 43egati model salah mengklasifikasikan contoh 43egative sebagai positif. Semakin rendah tingkat kesalahan *False Positive*, semakin tinggi *precision*.
2. Jumlah *True Positive* berbanding lurus dengan *precision*. Jika model dapat mengenali dan mengklasifikasikan lebih banyak contoh positif dengan benar, *precision* akan meningkat.

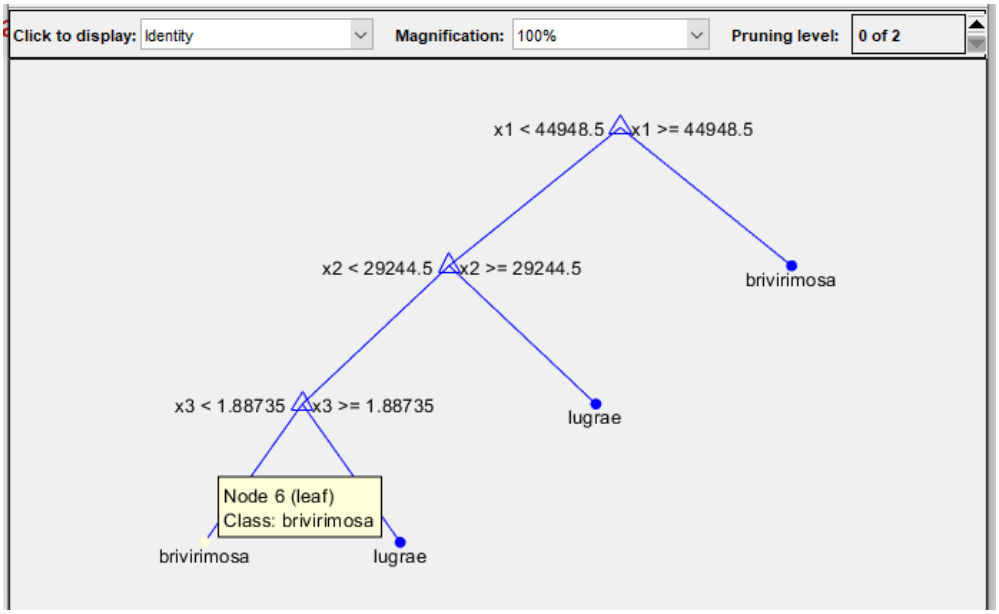
$$Recall = \frac{TP}{TP + FN} = \frac{37}{37 + 1} = \frac{37}{38} = 97\%$$

Preprocessing data seperti normalisasi, pengurangan dimensi, atau penghilangan *outlier* juga dapat mempengaruhi *recall*. Proses ini dapat membantu meningkatkan *recall* dengan meningkatkan kualitas data yang digunakan dalam pelatihan model

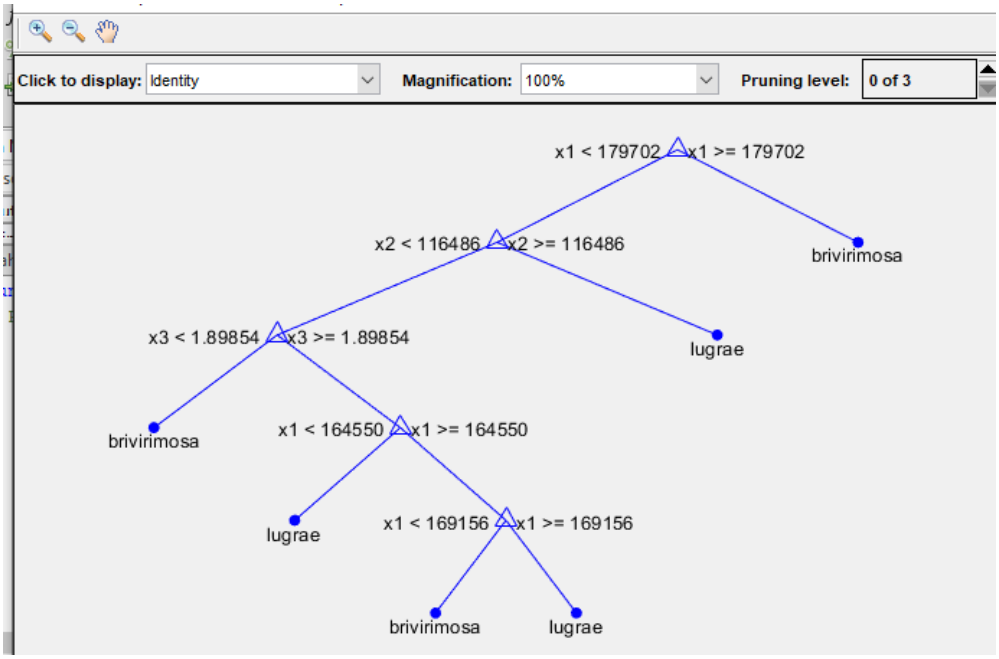
$$F1\ Score = \frac{2\ x\ Presisi\ x\ Recall}{Presisi + Recall} = \frac{2\ x\ 100\%\ x\ 97\%}{100\% + 97\%} = \frac{1{,}94}{1{,}97} = 98\%$$

Adapun faktor yang mempengaruhi recall dalam klasifikasi diantaranya:

1. Kesalahan *False Negative*: *Recall* terpengaruh secara 44egative oleh jumlah *False Negative* yang tinggi. False Negative terjadi 44egati model gagal mengklasifikasikan contoh positif sebagai positif, dan secara keliru mengklasifikasikannya sebagai 44egative. Semakin rendah tingkat kesalahan *False Negative*, semakin tinggi recall.
2. Jumlah *True Positive*: *Recall* meningkat seiring dengan peningkatan jumlah *True Positive*, yaitu jumlah contoh positif yang secara benar diklasifikasikan sebagai positif oleh model. Jika model dapat mengenali lebih banyak contoh positif dengan benar, *recall* akan meningkat.
3. Jumlah *False Positive*: Meskipun *False Positive* tidak secara langsung mempengaruhi *recall*, namun dapat berdampak secara tidak langsung melalui *False Negative*. Jika model mengklasifikasikan contoh 44egative sebagai positif dengan kesalahan (*False Positive*), ini dapat mengakibatkan penurunan jumlah.
4. Nilai *F1 Score* secara langsung didapatkan dari nilai *precision* dan *recall*. Ketidakseimbangan jumlah contoh antara kelas positif dan negatif juga dapat mempengaruhi *recall*. Jika jumlah contoh positif jauh lebih besar daripadnegatif, model cenderung menghasilkan lebih banyak *False Negative*, yang dapat menurunkan *recall.* Dalam situasi ketidakseimbangan kelas, perlu mempertimbangkan metrik evaluasi lain seperti *F1-score* yang memadukan *recall* dan *precision*.



Gambar 4.4 *Decision Tree* Skenario Pertama



Gambar 4.5 *Decision Tree* Begonia Brevirimosa

Decision tree adalah metode klasifikasi *dalam machine learning* yang digunakan untuk mengambil keputusan berdasarkan serangkaian pertanyaan (*decision*) terstruktur. Proses *decision tree* dalam melakukan klasifikasi dapat dijelaskan sebagai berikut:

1. Memilih variabel (atribut) yang paling penting

   Proses dimulai dengan memilih atribut yang paling penting untuk memisahkan data menjadi kelas yang berbeda. Atribut yang paling penting akan memiliki kemampuan untuk memisahkan data dengan cara yang paling efisien.

2. Membuat *node* keputusan

   Setelah atribut yang paling penting dipilih, *node* keputusan dibuat pada cabang pohon. *Node* keputusan berisi pertanyaan biner yang memisahkan data menjadi dua kelompok yang berbeda berdasarkan nilai atribut yang dipilih.

3. Memisahkan data

   Data dipisahkan berdasarkan nilai atribut pada *node* keputusan. Setiap cabang pohon mengandung data yang homogen. Artinya, data pada setiap cabang pohon harus memiliki kesamaan dalam kelas target atau atribut yang dipilih.

4. Melakukan rekursi

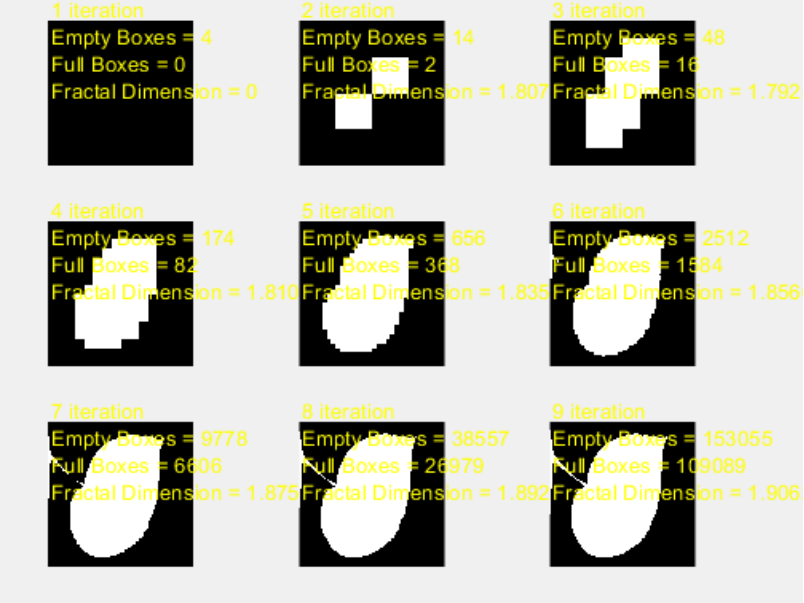
   Proses tersebut dilakukan secara rekursif pada setiap cabang pohon hingga mencapai titik di mana tidak ada lagi atribut yang dapat memisahkan data dengan lebih baik atau data telah terbagi menjadi kelas yang homogen.

5. Membuat *leaf node*

   Leaf node atau simpul daun dibuat saat tidak ada lagi atribut yang dapat memisahkan data atau ketika data telah terbagi menjadi kelas yang homogen. Setiap simpul daun mewakili kelas target.

6. Mengukur performa

   Setelah pohon keputusan selesai dibangun, performa model diukur dengan menggunakan data validasi. Data validasi digunakan untuk mengukur akurasi model dalam memprediksi kelas target.



Gambar 4.6 Iterasi Proses Fraktal Dimensi 2

dan merujuk pada metode penghitungan dimensi fraktal yang disebut "metode kotak" (*box-counting method*). Metode ini digunakan untuk mengukur dimensi fraktal dari suatu objek citra digital.

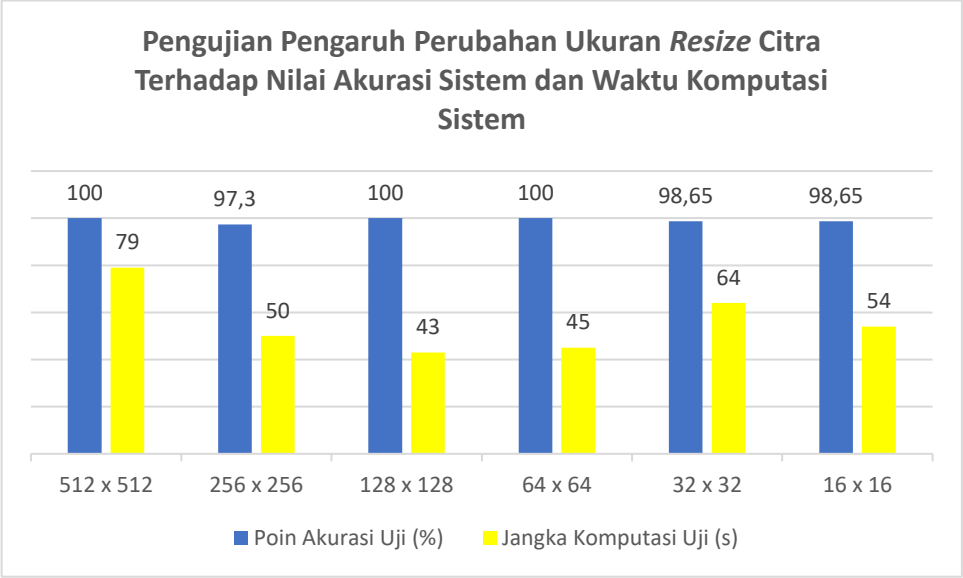
Metode kotak ini melibatkan pembagian objek yang akan diukur menjadi kotak-kotak kecil dengan ukuran tertentu. Kemudian, kotak-kotak ini ditempatkan pada objek dan dihitung berapa banyak kotak yang kosong (*empty boxes*) dan berapa banyak kotak yang terisi. Nilai-nilai ini kemudian digunakan untuk menghitung dimensi fraktal. Semakin kecil ukuran kotak, semakin banyak kotak yang dibutuhkan untuk mengisi seluruh objek. Ini berarti akan ada lebih banyak. Semakin banyak kotak yang terisi, semakin detail dan kompleks objek yang diukur. Ini berarti dimensi fraktal yang diukur semakin tinggi. Dalam metode kotak, dimensi fraktal didefinisikan sebagai logaritma negatif dari rasio antara jumlah kotak yang diperlukan untuk menutupi objek dan ukuran kotak. Dalam hal ini, empty boxes dan full *boxes* juga digunakan untuk menghitung rasio ini. Dengan demikian, hubungan antara dan dimensi fraktal adalah penting dalam metode kotak untuk mengukur dimensi fraktal dari objek.

4.10 Hasil *Confusion* Matrik Dengan Percobaan Beberapa Ukuran Citra Yang Berbeda

Pada bagian ini, peneliti melakukan perubahan dalam ukuran citra data latih dan uji. Beberapa ukuran citra yang diujikan antara lain 16 x 16, 32 x 32, 64 x 64, 128 x 128, 256 x 256, dan 512 x 512. Hasil dari pengujian ini bisa dilihat pada gambar dan tabel berikut.

Tabel 4.15 Pengujian Ukuran *Resize* Citra Terhadap Akurasi dan Waktu Komputasi

No	Skala *Resize* Citra	Poin Akurasi Latih (%)	Poin Akurasi Uji (%)	Jangka Komputasi Uji (s)
1.	512 x 512	100%	100%	79
2.	256 x 256	100%	97,30%	50
3.	128 x 128	100%	100%	43
4.	64 x 64	100%	100%	45
5.	32 x 32	100%	98,65%	64
6.	16 x 16	100%	98,65%	54



Gambar 4.7 Grafik Pengujian Ukuran *Resize* Citra Terhadap Poin Akurasi Latih dan Poin Akurasi Uji.

Pada penelitian ini, hasil pengujian pengaruh *resize* citra terhadap akurasi sistem dan jangka komputasi sistem menghasilkan poin akurasi *resize* citra paling rendah sebesar 256 x 256 piksel, serta menunjukkan poin akurasi pengujian sebesar 97,3% dengan waktu komputasi uji selama 50 detik. Pada tabel juga dapat dilihat

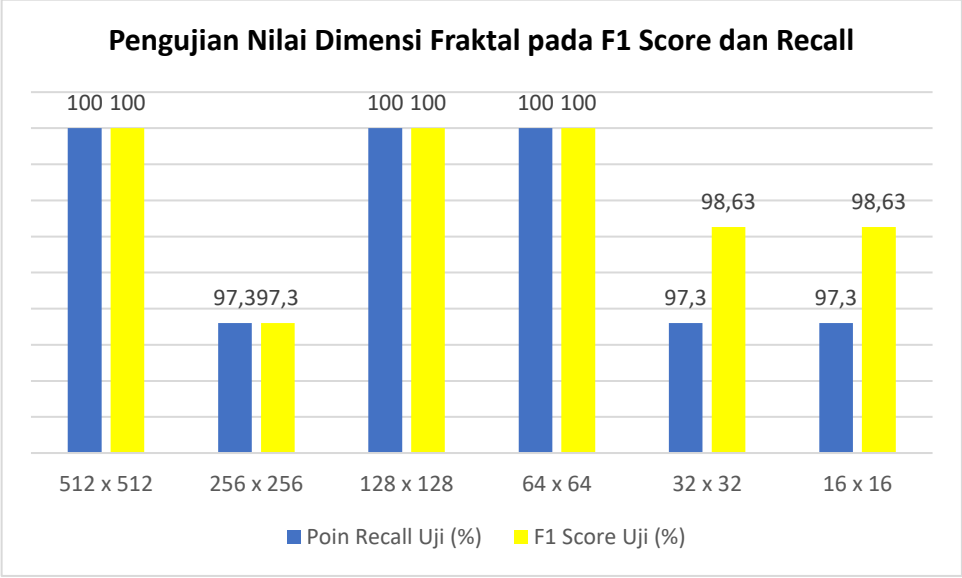
bahwa poin akurasi paling tinggi diperoleh pada uji *resize* citra dengan ukuran 512 x 512 piksel serta memberikan hasil akurasi pengujian sebesar 100% dengan waktu komputasi uji selama 79 detik. Oleh karena itu, ukuran terbaik pada *resize* citra yang digunakan dalam pengujian ini adalah 512 x 512.

4.11 Pengujian Nilai Dimensi Fraktal pada *F1 Score* Terhadap *Precision* dan *Recall*

Pada bagian ini, peneliti melakukan berbagai perubahan dalam nilai dimensi fraktal. Perubahan yang dilakukan pada skala *resize* citra yang penulis lakukan bervariasi dari ukuran 16 hingga 512. Hasil dari uji pada bagian ini bisa dilihat pada gambar dan tabel berikut.

Tabel 4.16 Pengujian dengan skala resize citra pada *F1 Score* Terhadap *Precision* dan *Recall*

No	Skala *Resize* Citra	Poin *Precision* Uji (%)	Poin *Recall* Uji (%)	*F1 Score* Uji (s)
1.	512 x 512	100%	100%	100%
2.	256 x 256	100%	97,30%	97,30%
3.	128 x 128	100%	100%	100%
4.	64 x 64	100%	100%	100%
5.	32 x 32	100%	97,30%	98,63%
6.	16 x 16	100%	97,30%	98,63%



Gambar 4.8 Grafik Pengujian Nilai Dimensi Fraktal pada *F1 Score* dan *Recall*

Setelah dilakukan percobaan, diperoleh perbedaan dimensi fraktal pada *precision* latih dan *recall* uji. Pengujian yang dilakukan memberikan hasil *recall* uji paling rendah pada ukuran citra yang telah di-*resize* hingga ukuran 256 x 256 dengan presentase *recall* uji 97,3%, serta presentase *F1 score* uji sebesar 97,3%. Sementara itu, hasil recall dan dan *F1 score* uji paling tinggi diperoleh pada ukuran *resize*-citra ke-1, 3, dan 4 yaitu ukuran 512 x 512, 128 x 128, dan 64 x 64. Nilai *presentase recall* pengujian pada penelitian tersebut sebesar 100% serta hasil presentase *F1 score* uji sebesar 100%. Sedangkan nilai persentasi dengan tinggi 100% didapatkan paling banyak di parameter *Recall*.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari hasil pengujian serta analisis yang telah dilakukan, dapat disimpulkan bahwa metode fraktal dengan algoritma *decision tree* dapat memberikan perbedaan dengan cukup maksimal. Metode ini memiliki presentase akurasi paling tinggi pada *resize* citra dengan ukuran 512 x 512 piksel. Presentase akurasi uji yang diperoleh yaitu sebesar 100% dengan jangka waktu komputasi uji selama 79 detik. Sedangkan, nilai akurasi paling rendah diperoleh pada *resize* citra memiliki ukuran 256 x 256 piksel dengan nilai akurasi uji sebesar 97, 3% serta jangka waktu komputasi uji selama 50 detik. Pada langkah berikutnya juga diperoleh perbedaan dimensi fraktal pada *precision* latih dan *recall* uji. Pengujian yang dilakukan memberikan hasil *recall* uji paling rendah pada ukuran citra yang telah di-*resize* hingga ukuran 256 x 256 dan paling tinggi pada ukuran *resize* citra 512 x 512, 128 x 128, dan 64 x 64. Pada bagian terendah menghasilkan presentase *recall* serta presentase *F1 score* uji sebesar 97,3%. Sementara itu, dari hasil paling tinggi diperoleh presentase recall uji dan presentase *F1 score* uji sebesar 100%.

Hasil dari penelitian menggunakan metode *Decision tree* menujukkan hasil yang spesifik pada variabel data yang sudah diklasifikasikan. Metode *Decision tree* sering digunakan dalam *data mining* dan *machine learning* untuk mengklasifikasikan data, mengidentifikasi pola, dan membuat prediksi berdasarkan input yang diberikan. Sejalan dengan hal tersebut metode ini memudahkan penulis dalam mengklasifikasikan tanaman *begonia brevirimosca* maupun *begonia lugrae.* Hasil penelitian berdasarkan klasifikasi *decision tree,* menunjukkan perolehan yang yang baik dalam membedakan jenis tanaman begonia *brevirimosa* dan *lugrae* mendeteksi jenis tanaman dengan menggunakan berbagai parameter yang mempengaruhi nilai akurasi.

5.2 Saran

Berdasarkan pada pengujian yang telah dilakukan, agar mendapatkan hasil yang lebih baik sebaiknya menggunakan lebih banyak data latih dan data uji sehingga mendapatkan hasil yang lebih maksimal. Bagi penelitian selanjutnya, sebaiknya dilakukan perbandingan dengan jenis algoritma lainnya untuk pengembangan lebih lanjut dari sistem ini. Selanjutnya, untuk penelitian lainnya dapat juga dilakukan dengan membuat klasifikasi jenis tanaman lain dalam lingkup yang lebih luas.

DAFTAR PUSTAKA

[1] P. W. Moonlight *et al.*, “Dividing and conquering the fastest-growing genus: Towards a natural sectional classification of the mega-diverse genus begonia (Begoniaceae),” *Taxon*, vol. 67, no. 2, pp. 267–323, 2018, doi: 10.12705/672.3.

[2] Hartutiningsih-M. Siregar, S. Wahyuni, and I. M. Ardaka, “Karakterisasi Morfologi Daun Begonia Alam (Begoniaceae): Prospek Pengembangan Koleksi Tanaman Hias Daun di Kebun Raya Indonesia (Leaf morfological characterization of native Begonia (Begoniaceae): Development prospect of foliage ornamental plants collecti,” *J. Biol. Indones.*, vol. 14, no. 2, pp. 201–211, 2018.

[3] “(Cattle Wieght Estimation Based on Digital Image With Fractal,” vol. 8, no. 2, pp. 1385–1393, 2021.

[4] S. Widyarto, M. W. Sharif, M. Syafrullah, and G. A. Budaya, “Fractals study and its application,” *Int. Conf. Electr. Eng. Comput. Sci. Informatics*, pp. 200–204, 2019, doi: 10.23919/EECSI48112.2019.8977124.

[5] Y. G. Pamela and D. Juniati, “Klasifikasi Jenis Delphinidae (Lumba-Lumba) Dengan Dimensi Fraktal Menggunakan Metode Higuchi Dan Knn (K-Nearest Neighbor),” *MATHunesa J. Ilm. Mat.*, vol. 9, no. 1, pp. 204–211, 2021, doi: 10.26740/mathunesa.v9n1.p204-211.

[6] A. Zainet, R. Magdalena, and J. Raharjo, “Classification of Non-Proliferative Diabetic Retinopathy (Npdr) By Iris Images Using Fractal Method,” *Semin. Nas. Teknol. Inf. dan Komun.*, pp. 503–509, 2020.

[7] D. Juniati and A. E. Suwanda, “Klasifikasi Penyakit Mata Berdasarkan Citra Fundus Retina Menggunakan Dimensi Fraktal Box Counting Dan Fuzzy K-Means,” *Prox. J. Penelit. Mat. dan Pendidik. Mat.*, vol. 5, no. 1, pp. 10–18, 2022, doi: 10.30605/proximal.v5i1.1623.

[8] R. Zainet, “https://repository.telkomuniversity.ac.id/catalogue/2021.html,” 2021.

[9] M. Vishnu and R. Jaishanker, “Fractal-Thermodynamic System Analogy and Complexity of Plant Leaves,” *bioRxiv*, p. 2022.07.05.498782, 2022, [Online]. Available: http://biorxiv.org/content/early/2022/07/05/2022.07.05.498782.abstract.

[10] N. P. Nurpadilah *et al.*, “Identifikasi Pola Rugae Palatina Berdasarkan Metode Image Registration Dan Fractal Dengan Klasifikasi Decision Tree Pada Populasi Mahasiswa S1 Teknik Telekomunikasi Angkatan 2015 Universitas Telkom Palatal Rugae Pattern Identification With Image Registrat,” *e-Proceeding Eng.*, vol. 6, no. 1, pp. 734–740, 2019.

[11] R. Munir, “https://opac.perpusnas.go.id/DetailOpac.aspx?id=332204,” 2004.

[12] B. Kamiński, M. Jakubczyk, and P. Szufel, “A framework for sensitivity analysis of decision trees,” *Cent. Eur. J. Oper. Res.*, vol. 26, no. 1, pp. 135–159, 2018, doi: 10.1007/s10100-017-0479-6.

[13] B. Charbuty and A. Abdulazeez, “Classification Based on Decision Tree Algorithm for Machine Learning,” *J. Appl. Sci. Technol. Trends*, vol. 2, no. 01, pp. 20–28, 2021, doi: 10.38094/jastt20165.

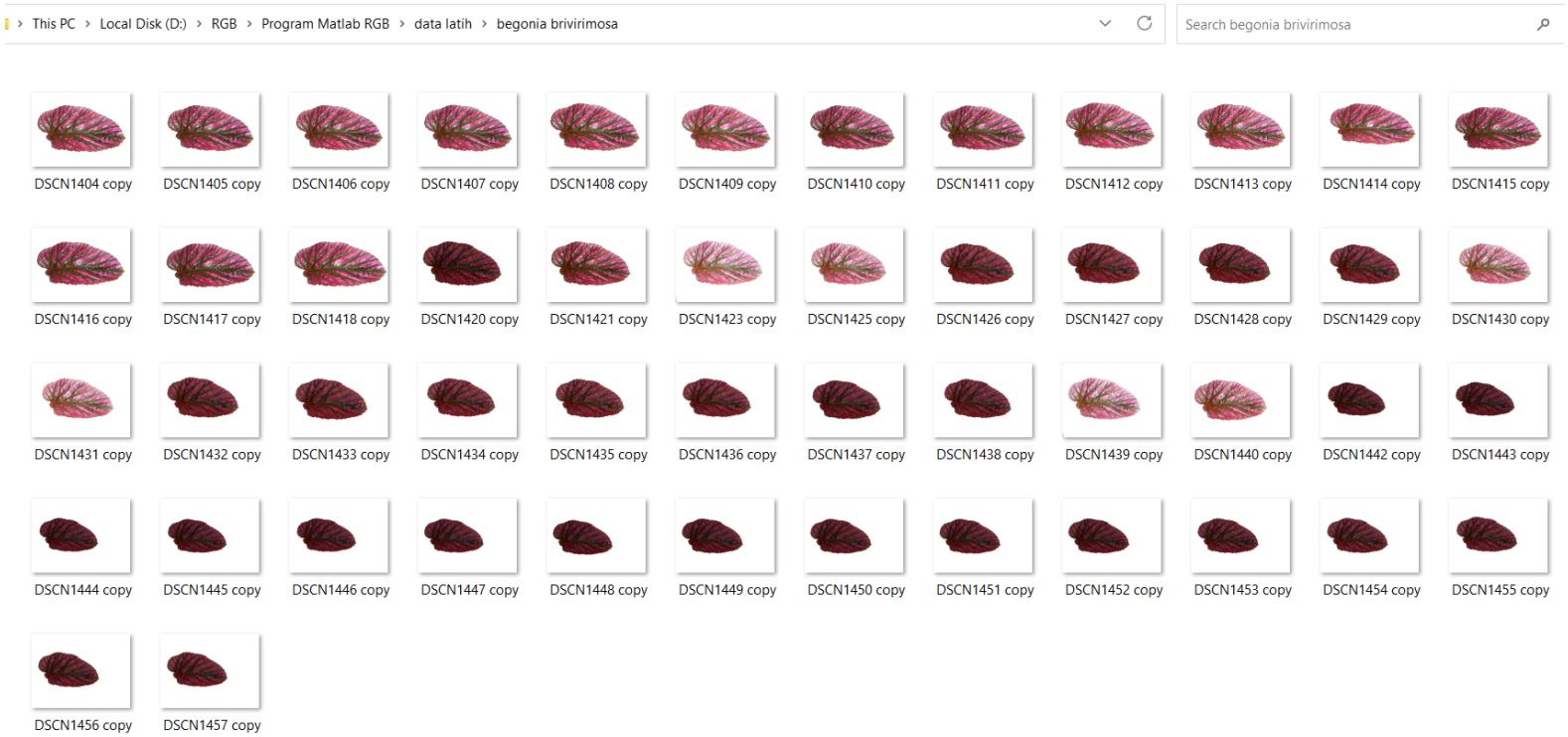
[14] N. Alifiah, “http://digilib.unila.ac.id/id/eprint/64426,” 2022.

[15] R. Cheng, “A survey: Comparison between Convolutional Neural Network and YOLO in image identification,” *J. Phys. Conf. Ser.*, vol. 1453, no. 1, 2020, doi: 10.1088/1742-6596/1453/1/012139.

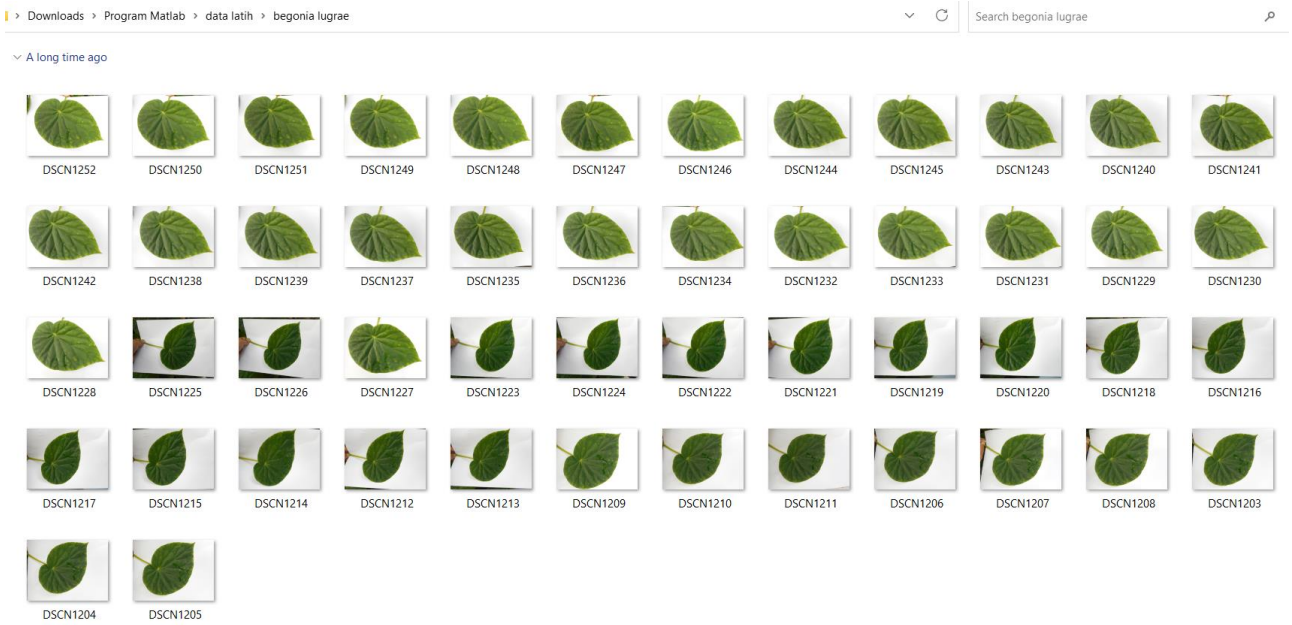
LAMPIRAN

LAMPIRAN 1: DATA LATIH

- Data latih *Begonia brevirimosa*

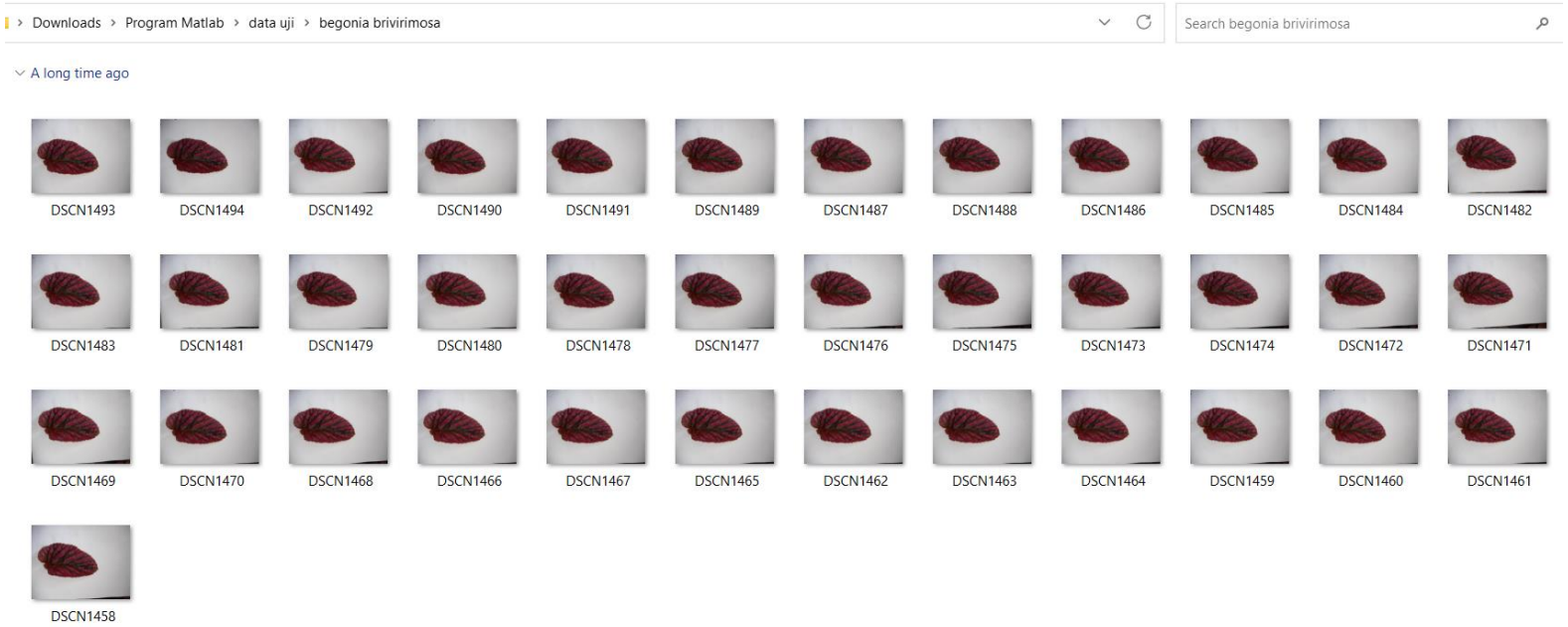


- Data latih *Begonia lugrae*

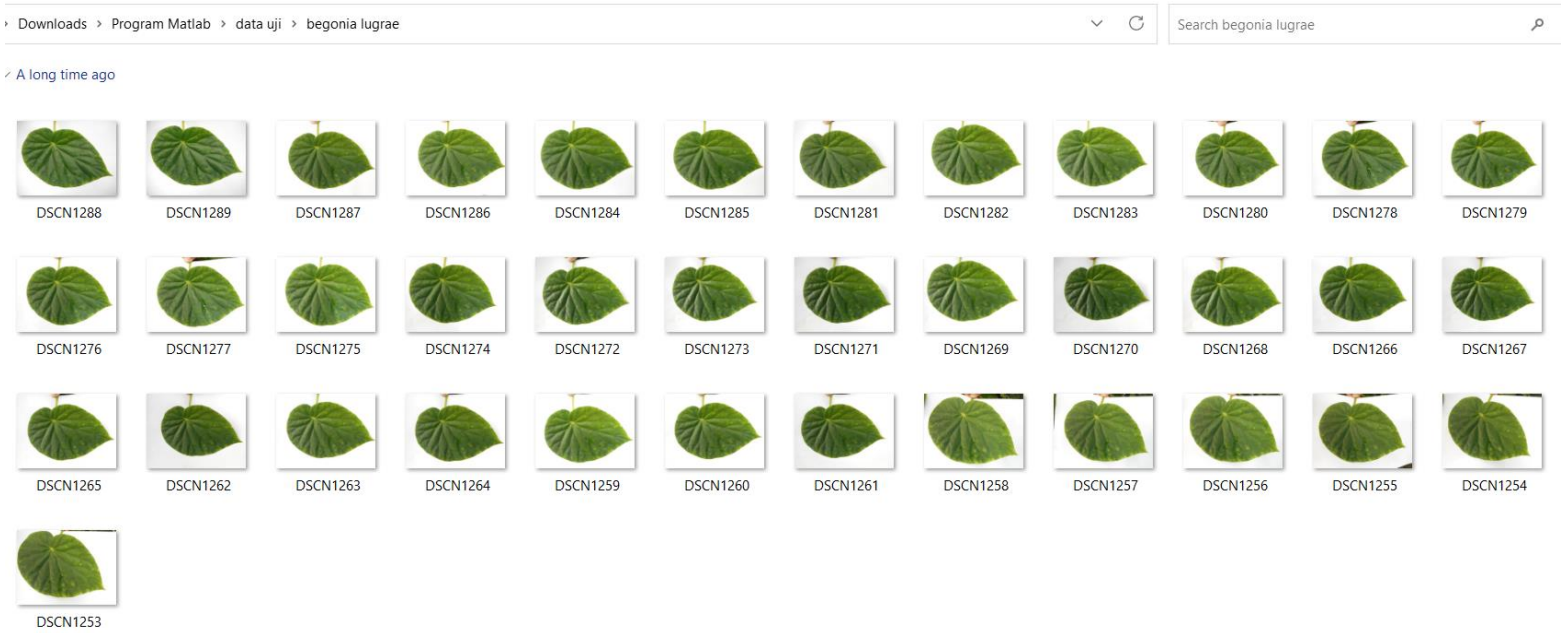


LAMPIRAN 2: DATA UJI

- Data Uji Begonia *brevirimosa*

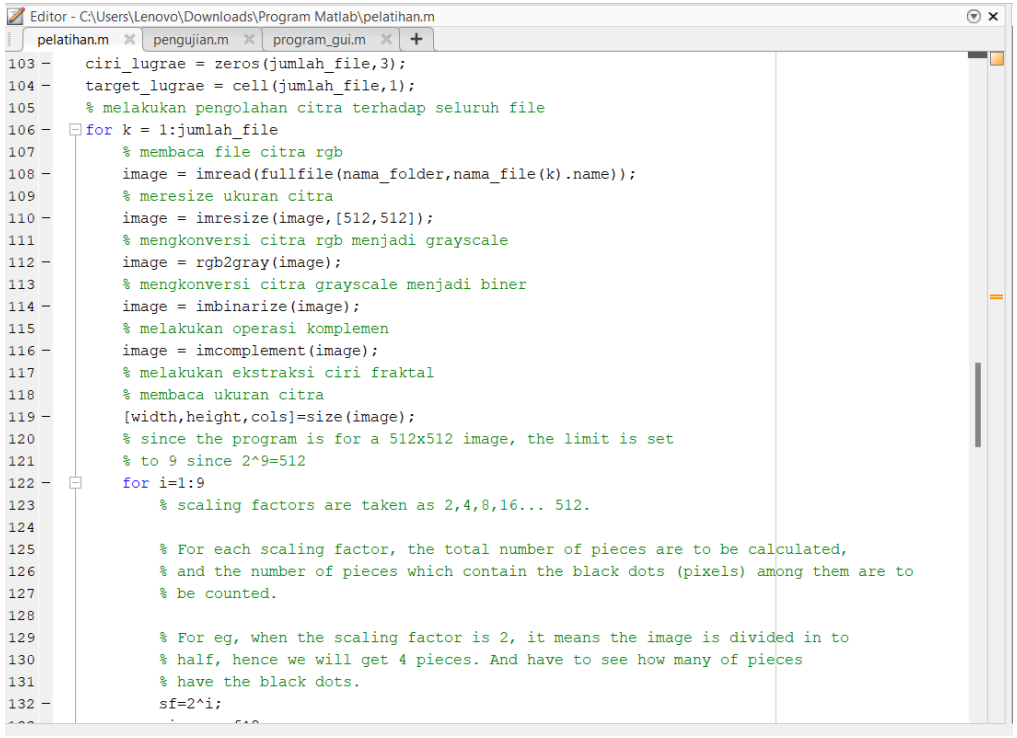


- Data Uji Begonia *lugrae*



LAMPIRAN 3: *Source Code*

- *Code* untuk metode fraktal




```
103 -   ciri_lugrae = zeros(jumlah_file,3);
104 -   target_lugrae = cell(jumlah_file,1);
105     % melakukan pengolahan citra terhadap seluruh file
106 -   for k = 1:jumlah_file
107         % membaca file citra rgb
108 -       image = imread(fullfile(nama_folder,nama_file(k).name));
109         % meresize ukuran citra
110 -       image = imresize(image,[512,512]);
111         % mengkonversi citra rgb menjadi grayscale
112 -       image = rgb2gray(image);
113         % mengkonversi citra grayscale menjadi biner
114 -       image = imbinarize(image);
115         % melakukan operasi komplemen
116 -       image = imcomplement(image);
117         % melakukan ekstraksi ciri fraktal
118         % membaca ukuran citra
119 -       [width,height,cols]=size(image);
120         % since the program is for a 512x512 image, the limit is set
121         % to 9 since 2^9=512
122 -       for i=1:9
123             % scaling factors are taken as 2,4,8,16... 512.
124
125             % For each scaling factor, the total number of pieces are to be calculated,
126             % and the number of pieces which contain the black dots (pixels) among them are to
127             % be counted.
128
129             % For eg, when the scaling factor is 2, it means the image is divided in to
130             % half, hence we will get 4 pieces. And have to see how many of pieces
131             % have the black dots.
132 -           sf=2^i;
```

- *Code* untuk metode *decision tree*

Editor - D:\RGB\Program Matlab RGB\pelatihan.m

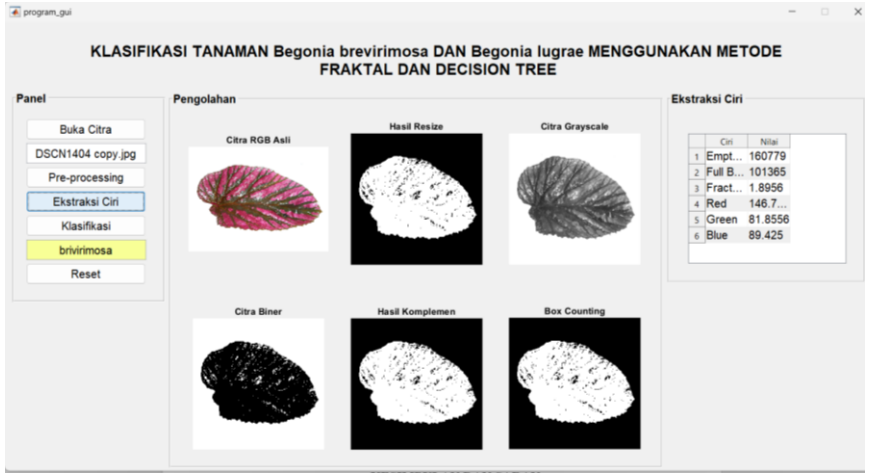
pelatihan.m | pengujian.m | program_gui.m

```
222     % melakukan klasifikasi menggunakan decision tree
223 -   rng(1); % For reproducibility
224 -   Mdl = fitctree(ciri,target);
225
226     % membaca kelas keluaran
227 -   kelas_keluaran = predict(Mdl,ciri);
228
229     % membentuk matriks confusion
230 -   figure
231 -   confusionchart(target,kelas_keluaran);
232 -   title('Confusion Matrix (Pelatihan)')
233 -   confMat = confusionmat(target,kelas_keluaran);
234
235     % menghitung akurasi dll
236 -   TP = confMat(1,1);
237 -   FN = confMat(1,2);
238 -   FP = confMat(2,1);
239 -   TN = confMat(2,2);
240 -   accuracy = (TN+TP)/(TN+FP+FN+TP)*100;
```

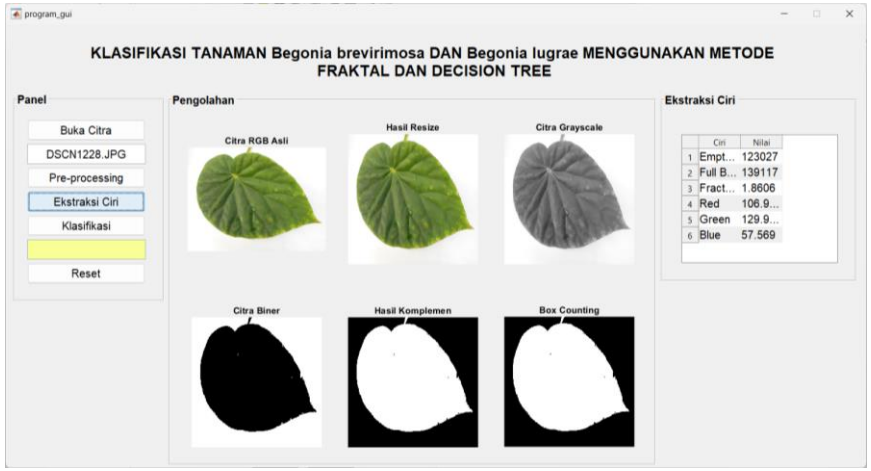
```
241 -   precision = TP/(TP+FP)*100;
242 -   recall = TP/(TP+FN)*100;
243 -   f1_score = 2*precision*recall/(precision+recall);
244
245     % menampilkan akurasi dll
246 -   disp('===== Pelatihan =====')
247 -   disp(['accuracy  = ',num2str(accuracy),'%'])
248 -   disp(['precision = ',num2str(precision),'%'])
249 -   disp(['recall    = ',num2str(recall),'%'])
250 -   disp(['f1 score  = ',num2str(f1_score),'%'])
251 -   disp('======================')
252
253     % menyimpan model decision tree
254 -   save Mdl Mdl
255 -   toc
```

GUI

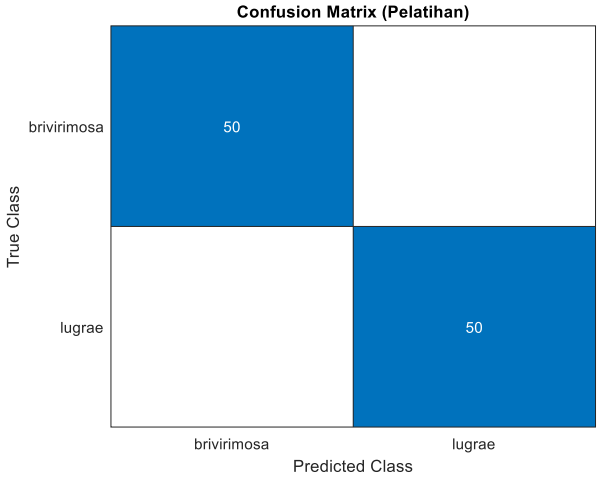
- Begonia *brevirimosa*



- Begonia *lugrae*



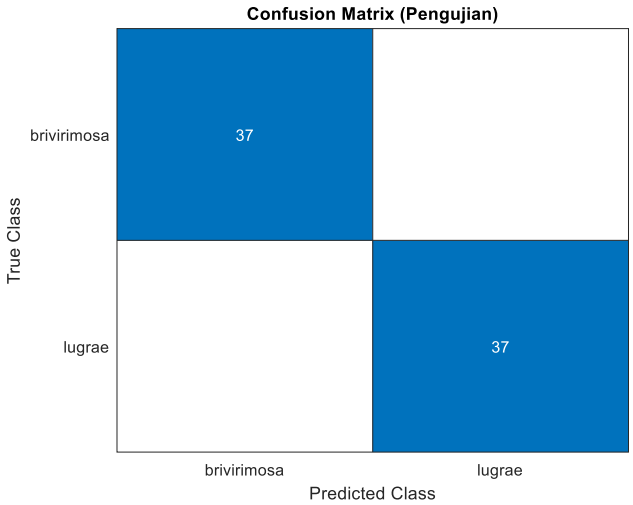
Hasil Simulasi Pelatihan ukuran resize citra 512 x 512



accuracy = 100%
precision = 100%
recall = 100%
f1 score = 100%

Jangka waktu uji komputasi adalah 115,290806 detik.

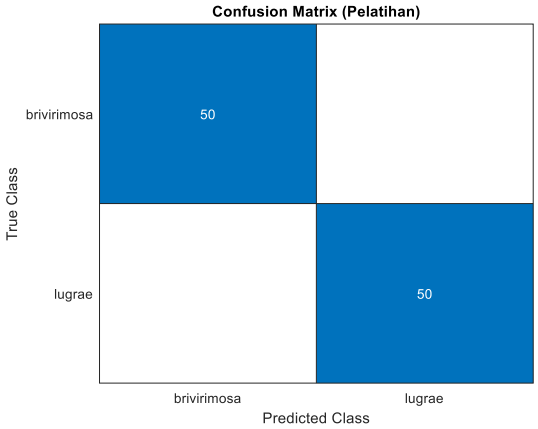
Hasil Simulasi Pengujian ukuran resize citra 512 x 512



accuracy = 100%
precision = 100%
recall = 100%
f1 score = 100%

Jangka waktu uji komputasi adalah 78,580597 detik.

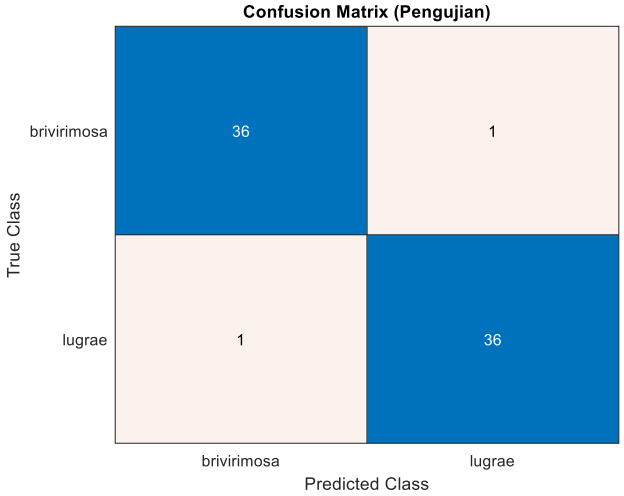
Hasil Simulasi Pelatihan ukuran resize citra 256 x 256



accuracy = 100%
precision = 100%
recall = 100%
f1 score = 100%

Jangka waktu uji komputasi adalah 66,924861 detik.

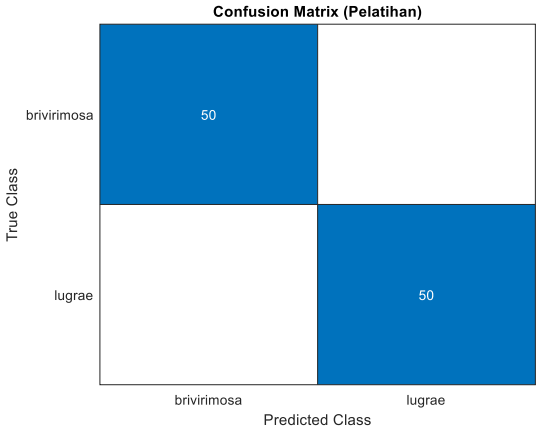
Hasil Simulasi Pengujian ukuran resize citra 256 x 256



accuracy = 97.2973%
precision = 97.2973%
recall = 97.2973%
f1 score = 97.2973%

Jangka waktu uji komputasi adalah 50,163126 detik.

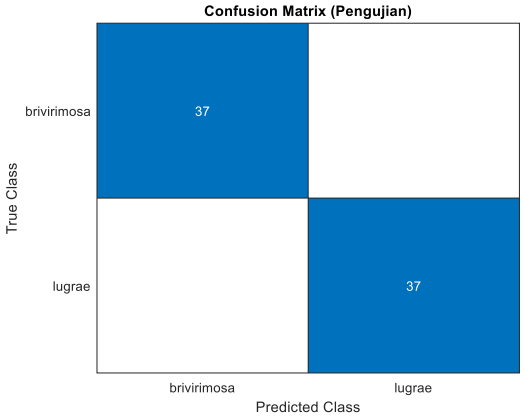
Hasil Simulasi Pelatihan ukuran resize citra 128 x 128



accuracy = 100%
precision = 100%
recall = 100%
f1 score = 100%

Jangka waktu uji komputasi adalah 64,536517 detik.

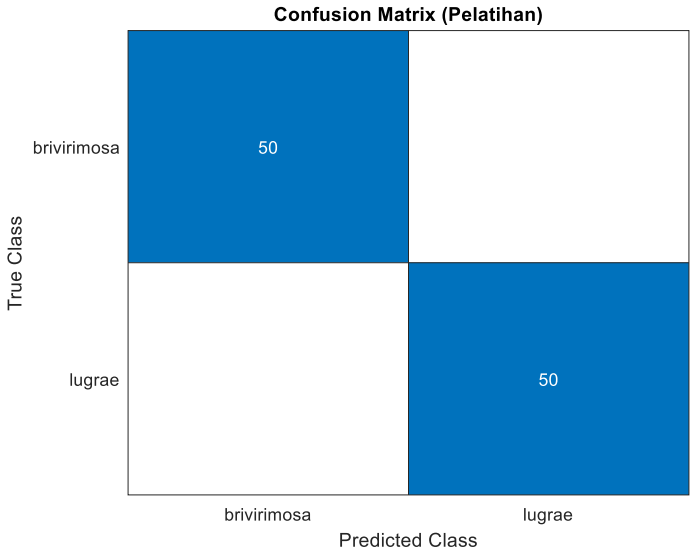
Hasil Simulasi Pengujian ukuran resize citra 128 x 128



accuracy = 100%
Precision = 100%
recall = 100%
f1 score = 100%

Jangka waktu uji komputasi adalah 43,017531 detik.

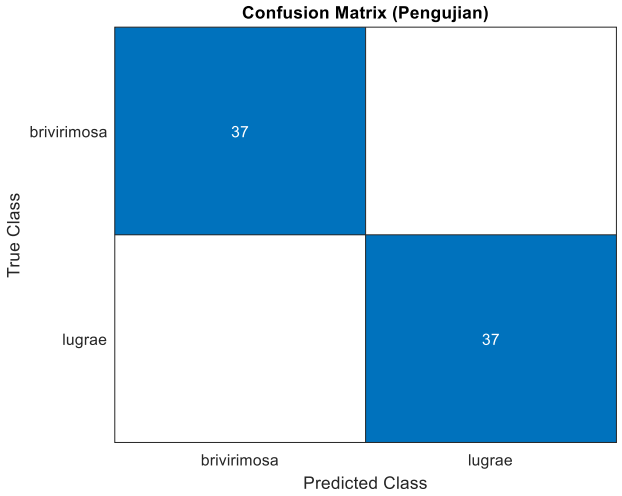
Hasil Simulasi Pelatihan ukuran resize citra 64 x 64



accuracy = 100%
precision = 100%
recall = 100%
f1 score = 100%

Jangka waktu uji komputasi adalah 67,843639 detik.

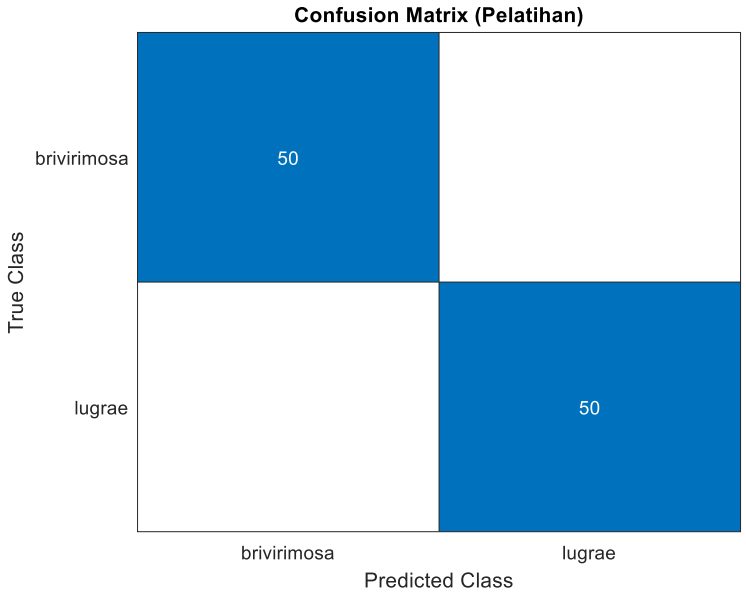
Hasil Simulasi Pengujian ukuran resize citra 64 x 64



accuracy = 100%
precision = 100%
recall = 100%
f1 score = 100%

Jangka waktu uji komputasi adalah 44,790816 detik.

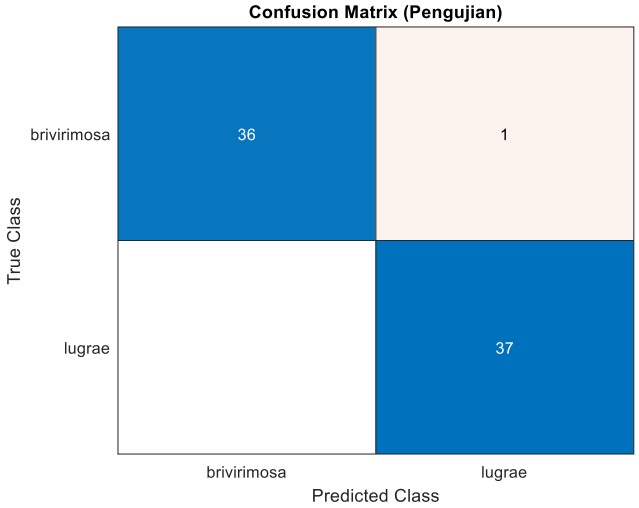
Hasil Simulasi Pelatihan ukuran resize citra 32 x 32



accuracy = 100%
precision = 100%
recall = 100%
f1 score = 100%

Jangka waktu uji komputasi adalah 70,424464 detik.

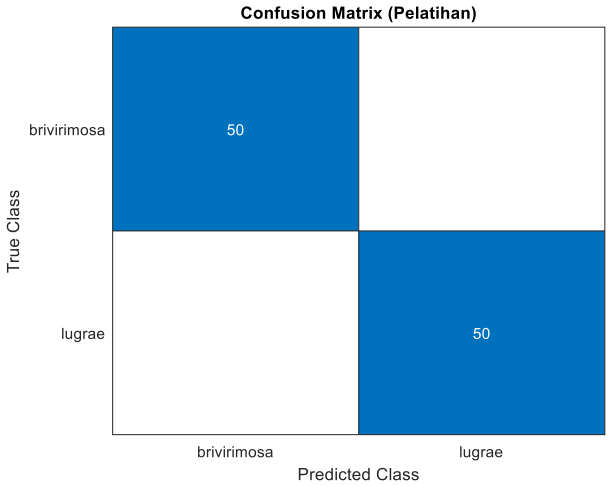
Hasil Simulasi Pengujian ukuran resize citra 32 x 32



accuracy = 98.6486%
precision = 100%
recall = 97.2973%
f1 score = 98.6301%

Jangka waktu uji komputasi adalah 63,518361 detik.

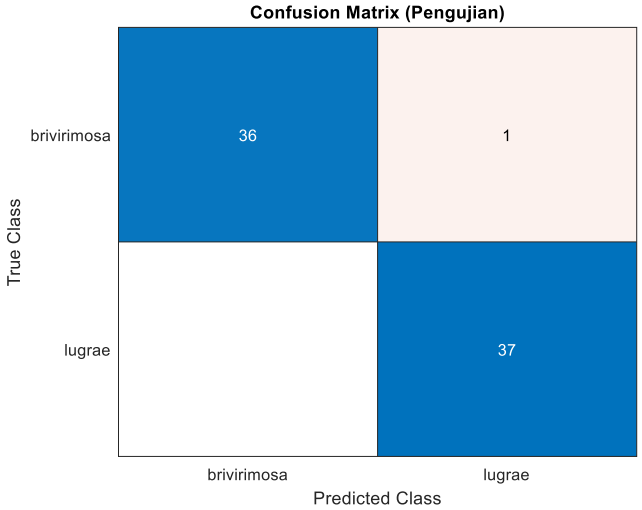
Hasil Simulasi Pelatihan ukuran resize citra 16 x 16



accuracy = 100%
precision = 100%
recall = 100%
f1 score = 100%

Jangka waktu uji komputasi adalah 70,338974 detik.

Hasil Simulasi Pengujian ukuran resize citra 16 x 16



accuracy = 98.6486%
precision = 100%
recall = 97.2973%
f1 score = 98.6301%

Jangka waktu uji komputasi adalah 54,066384 detik.