DAFTAR PUSTAKA

Ahmad, A. (n.d.). *Mengenal Artificial Intelligence, Machine Learning, Neural Network, dan Deep Learning*. www.teknoindonesia.com

A.Hidayat, D. Makhsun, and M. Si, "ANALISIS CITRA DAUN BERDASARKAN FITUR LOCAL BINARY PATTERN DAN FITUR CANNY EDGE DETECTION MENGGUNAKAN METODE KLASIFIKASI K-NEAREST NEIGHBOR (K-NN)," J. Esit,pp. 1– 10,2018.

Akhtar, Kashif dkk. (2017). Pattern Recognition and Classification of Indian Herbal Leaves. RIET-IJSET International Journal of Science Engineering and Technology. 6.

Alvansga, E., 2019. Pengenalan Tekstur Menggunakan Metode GLCM Serta Modul Nirkabel. *Comput. J*, pp.70-75.

Amalia, Tuti. 2022. "Klasifikasi Mutu Cabai Merah Besar (*Capsicuum Annuum* L.) Berbasis Video Processing. *Skripsi*. Universitas Hasanuddin. Makassar.

A. M. Raghukumar and G. Narayanan, "Comparison Of Machine Learning Algorithms For Detection Of Medicinal Plants," 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC).

Begue, Adams dkk. (2017). Automatic Recognition of Medicinal Plants using Machine Learning Techniques. International Journal of Advanced Computer Science and Applications. 8. 10.14569/ IJACSA.2017.080424.

Budi Putranto, Benedictus Yoga, et al. "Segmentasi Warna Citra Dengan Deteksi Warna Hsv Untuk Mendeteksi Objek." *Informatika: Jurnal Teknologi Komputer dan Informatika*, vol. 6, no. 2, 2010, doi:10.21460/inf.2010.62.81.

Budi Prawira, A., & Widiastiwi, Y. (2021). Penerapan Metode Gray Level Co-Occurance Matrix dan Algoritma *Support Vector Machine* Pada Klasifikasi Tanaman Bidara Berdasarkan Tekstur Daun. In *Seminar Nasional Mahasiswa Ilmu Komputer dan Aplikasinya (SENAMIKA) Jakarta-Indonesia*.

Darnita, Y., Toyib, R., & Kurniawan, Y. (2020). PENERAPAN METODE K-MEANS CLUSTERING PADA APLIKASI ANDROID PADA TANAMAN

OBAT HERBAL. In *Jurnal Pseudocode* (Vol. 2). www.ejournal.unib.ac.id/index.php/pseudocode

D. P. Adriani and H. Syahputra, "Klasifikasi Tanaman Obat-Obatan Berdasarkan Citra Daun Dengan Menggunakan Jaringan Syaraf Tiruan," Karismatika Vol. 6 No. 3 Desember 2020.

F. Shofrotun, dkk. (2018). Identifikasi Tumbuhan Obat Herbal Berdasarkan Citra Daun Menggunakan Algoritma Gray Level Co-occurence Matrix dan K-Nearest Neighbor. Jurnal Teknologi dan Sistem Komputer. 6. 51. 10.14710/jtsiskom.6.2.2018.51-56.

Feta, N., Ginanjar, A., 2019. KOMPARASI FUNGSI KERNEL METODE *SUPPORT VECTOR MACHINE* UNTUK PEMODELAN KLASIFIKASI TERHADAP PENYAKIT TANAMAN KEDELAI 1, 33–39.

Grandini, M., Bagli, E., & Visani, G. (2020). *Metrics for Multi-Class Classification: An Overview*. http://arxiv.org/abs/2008.05756

Harefa, D., Nias Selatan, S., Kunci, K., & Tanaman Obat Keluarga, P. (2020). Pemanfaatan Hasil Tanaman Sebagai Tanaman Obat Keluarga (TOGA). *Indonesian Journal Of Civil Society*, *2*(2), 28–36. https://doi.org/10.35970/madani.v1i1.233

Hasan Robbani, I., Trisnawati, E., Noviyanti, R., Rivaldi, A., Puji Cahyani, F., & Utaminingrum, F. (2016). APLIKASI MOBILE SCOTECT: APLIKASI DETEKSI WARNA TANAH DENGAN TEKNOLOGI CITRA DIGITAL PADA ANDROID. In *Jurnal Teknologi Informasi dan Ilmu Komputer (JTIIK)* (Vol. 3, Issue 1).

Isman, Andani Ahmad, and Abdul Latief. 2021. “Perbandingan Metode KNN Dan LBPH Pada Klasifikasi Daun Herbal | Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi),” July. https://jurnal.iaii.or.id/index.php/RESTI/article/view/3006.

Jamaluddin, Nur Latifah, dkk. 2019. “Detection and Distance Estimation against Motorcycles as Navigation Aids for Visually-impaired People”. *12th International Conference on Information & Communication Technology and System (ICTS)* (pp. 224-228). Diakses pada tanggal 15 Oktober 2022

Open Computer Vision Module. 2021. “Interactive *Foreground* Extraction Using *Grabcut* Algorithm Open CV.” Open CV. https://docs.opencv.org/3.4/d8/d83/tutorial_py_*Grabcut*.html.

Prawira, A.B., Jayanta, J. and Widiastiwi, Y., 2021. Penerapan Metode Gray Level Co-Occurance Matrix dan Algoritma *Support Vector Machine* pada Klasifikasi Tanaman Bidara berdasarkan Tekstur Daun. *Senamika*, *2*(1), pp.569-578.

Praseptiyana, W.I., Widodo, A.W. and Rahman, M.A., 2019. Pemanfaatan Ciri Gray Level Co-occurrence Matrix (GLCM) Untuk Deteksi Melasma Pada Citra Wajah. *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer e-ISSN*, *2548*, p.964X.

Puspitasari, A., Eka Ratnawati, D., Wahyu Widodo, A., 2018. “Klasifikasi Penyakit Gigi Dan Mulut Menggunakan Metode *Support Vector Machine*”. Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer.

R. A. Saktiawan and A. Atmiasri, “PEMANFAATAN TANAMAN TOGA BAGI KESEHATAN KELUARGA DAN MASYARAKAT,” *abadi*, vol. 1, no. 2, pp. 57–64, Nov. 2017, doi: 10.36456/abadimas.v1.i2.a960.

Septiarini, A. and Wardoyo, R., 2015. Kompleksitas Algoritma GLCM untuk Ekstraksi Ciri Tekstur pada Penyakit Glaucoma. In *Prosiding Seminar Teknik Informatika dan Sistem Informasi. Bandung*

Shin, You-Eun, and Woong-Jin Han. 2021. “Online Finger Circumference Measurement System Using Semantic Segmentation with Transfer Learning.” *The Journal of Korean Institute of Information Technology* 19 (12): 105–13. https://doi.org/10.14801/jkiit.2021.19.12.105.

Suyanto. 2019. DATA MINING UNTUK KLASIFIKASI DAN KLASTERISASI DATA. Bandung: INFORMATIKA.

S. F. Alamsyah, “Implementasi Deep Learning Untuk KlasifikasiTanaman Toga Berdasarkan Ciri Daun Berbasis Android,” Ubiquitous: Computers And Its Applications Journal Volume 2, Nomor 2, Desember 2019, 113-122.

Taslinda. 2022. “Sistem Deteksi Hambatan pada Autonomous Driving Menggunakan Metode Single Shot Multibox Detector (SSD)”. *Skripsi*. Universitas Hasanuddin. Makassar.

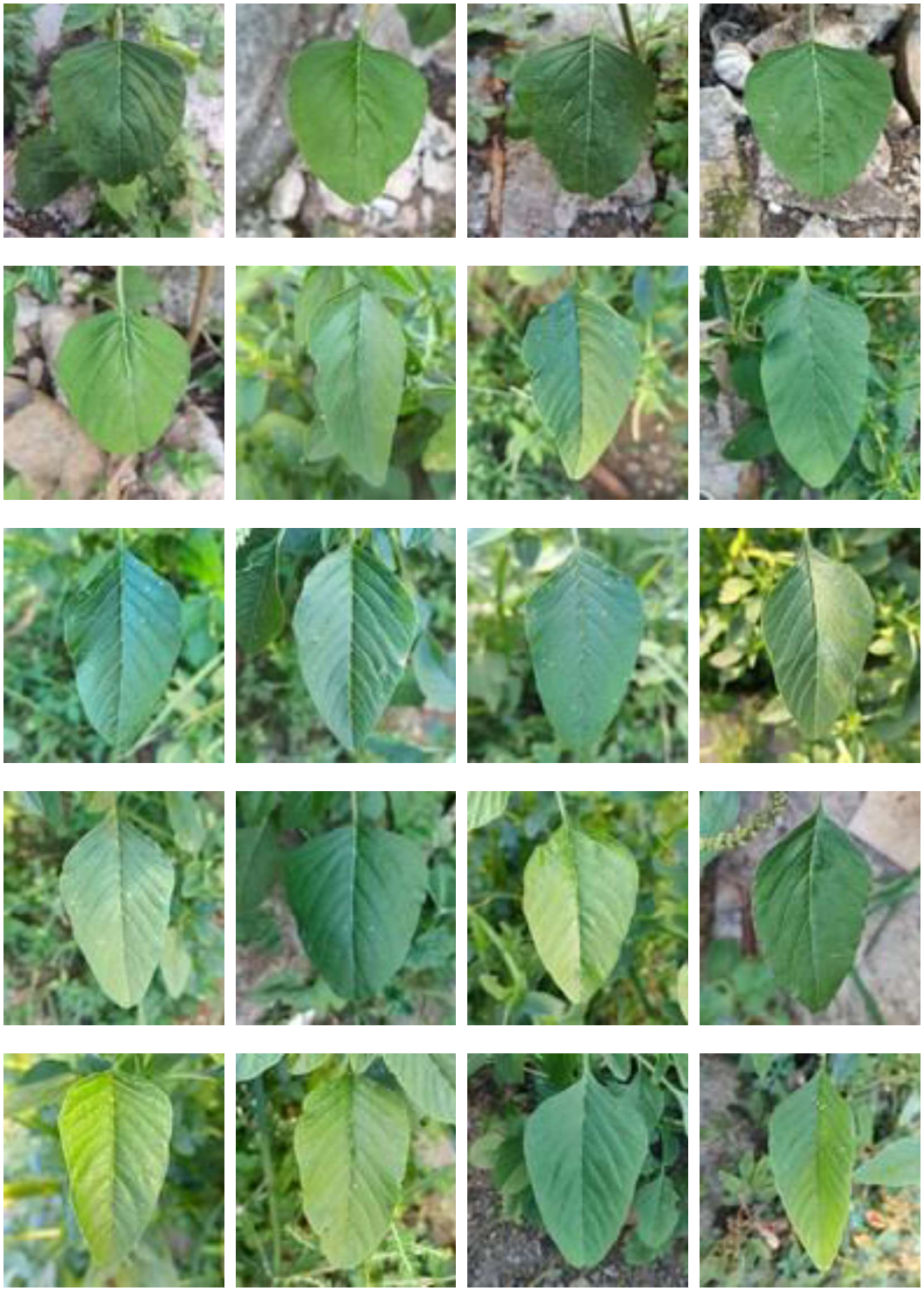
Z. Imaduddin and H. A. Tawakal, "Aplikasi Mobile Untuk DeteksiDan Klasifikasi Daun Secara Real Time", Jurnal Teknologi Terpadu Vol. 1, No. 1, Juli, 2015.

LAMPIRAN

Lampiran 1. Beberapa Contoh Dataset Primer

Kelas 1 : Tanaman Obat Anting-Anting

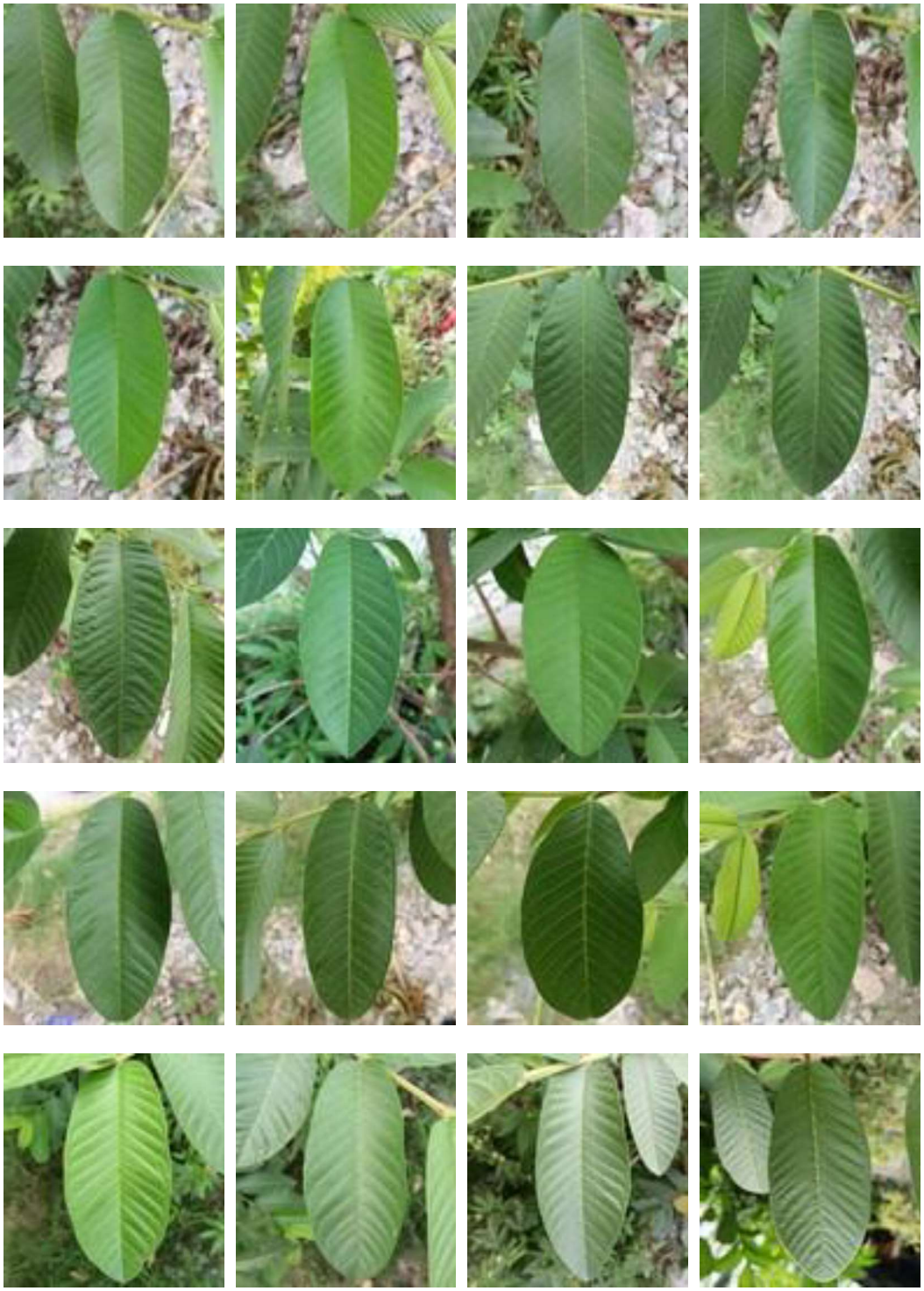
Kelas 2 : Tanaman Obat Bayam Duri



Kelas 3 : Tanaman Obat Bidara

Kelas 4 :Tanaman Obat Daun Ungu

Kelas 5 : Tanaman Obat Jambu Biji

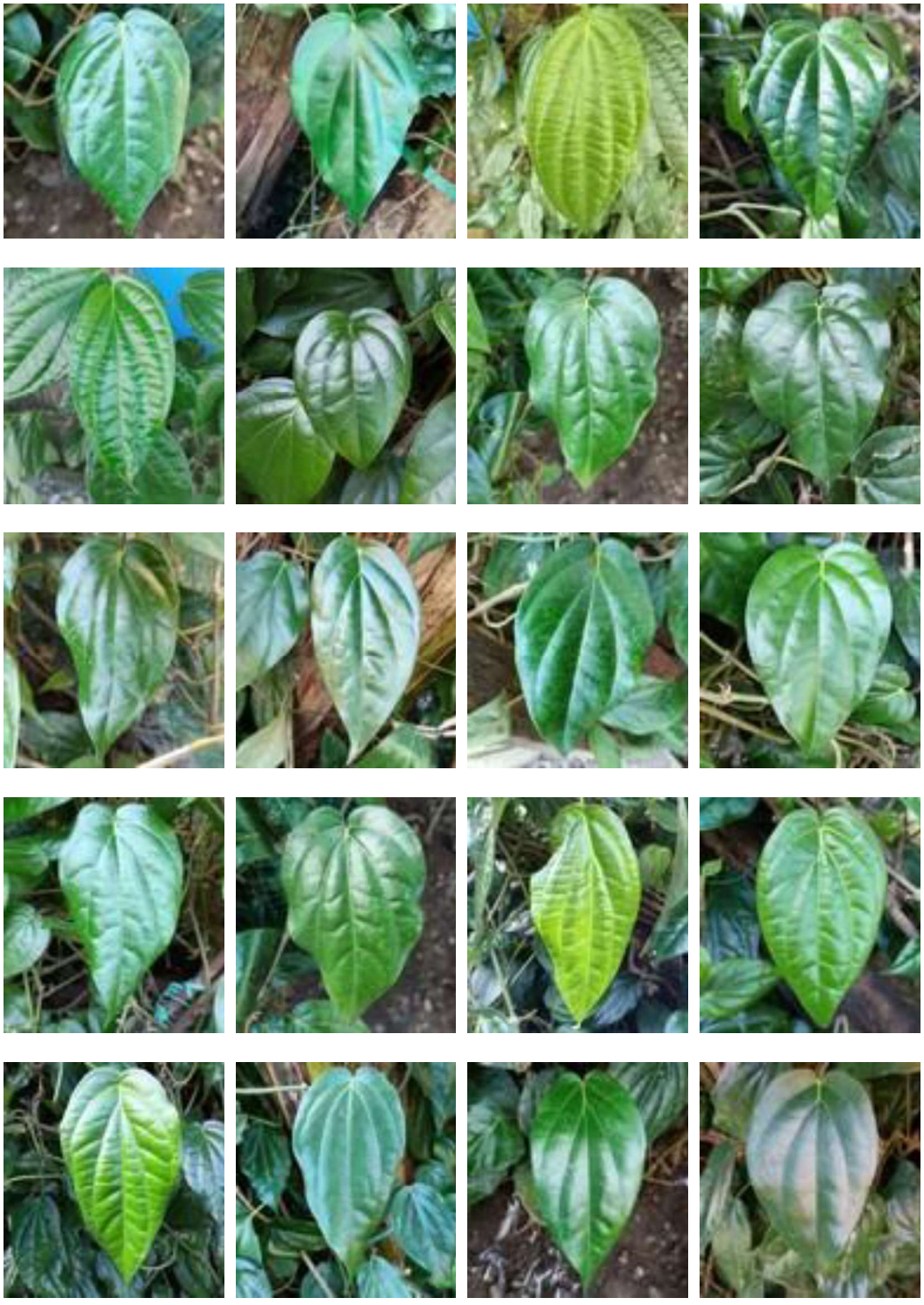


Kelas 6 : Tanaman Obat Kirinyuh

Kelas 7 : Tanaman Obat Miana

Kelas 8 : Tanaman Obat Sidaguri

Kelas 9 : Tanaman Obat Sirih



Kelas 10 : Tanaman Obat Sirsak

Lampiran 2. Source Code Preprocessing

```
#menggunakan library
import cv2
import os
import numpy as np
import matplotlib.pyplot as plt

#Augmentasi
def flip_image(image,dir):
    image = cv2.flip(image, dir)
    cv2.imwrite(path + "/flip-" + file, image)

#Background Substraction
def init_mask(h, w):
    mask = np.ones((h, w), np.uint8) * cv2.GC_PR_BGD
    mask[h//4:3*h//4, w//4:3*w//4] = cv2.GC_PR_FGD
    mask[2*h//5:3*h//5, 2*w//5:3*w//5] = cv2.GC_FGD
    return mask

def preprocess_background(path):
    original_image = cv2.imread(path)
    original_image = cv2.cvtColor(original_image, cv2.COLOR_BGR2RGB)
    original_image = cv2.resize(original_image , (500,500))
    image = original_image
    h, w = image.shape[:2]
    mask = init_mask(h, w)
    contours, _ = cv2.findContours(mask, cv2.RETR_EXTERNAL, cv2.CHAIN_APPROX_NONE)
    backgroundModel = np.zeros((1, 65), np.float64)
    foregroundModel = np.zeros((1, 65), np.float64)
    rectangle = (100, 10, 300, 480)
    cv2.grabCut(image, mask, rectangle,
                backgroundModel, foregroundModel,
                10, cv2.GC_INIT_WITH_RECT)
    mask2 = (mask==2) | (mask==0)
    image[mask2] = 255
    f = plt.figure()
    f.add_subplot(1,2, 1)
    plt.title(file)
    plt.imshow(image)
    image = cv2.cvtColor(image, cv2.COLOR_BGR2RGB)
    return image
```

Lampiran 3. Source Code Ekstraksi Fitur

```
# Membuat fungsi untuk dataset
def create_dataset():
    #Membuat perulangan untuk kolom feature glcm
    glcm_feature = ['correlation', 'homogeneity','dissimilarity','contrast','ASM','energy']
    angle = ['0','45','90','135']
    # Membuat variabel kolom untuk dataset
    names =
['class',\ 'area','perimeter','physiological_length','physiological_width','aspect_ratio','rectangularity','c
ircularity','eccentricity','metric','hue','saturation','value']
    #Pemanggilan perulangan kolom glcm
    for i in glcm_feature:
        for j in angle:
            names.append(i + ' ' + j)
# Membuat dataframe berdasarkan nama kolom yang dibuat
    df = pd.DataFrame([], columns=names)
    for i in range(len(results)):
        img = cv2.cvtColor(results[i], cv2.COLOR_BGR2RGB)
        grayscale = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        blur = cv2.GaussianBlur(grayscale, (25,25),0)
        ret, img1 =
cv2.threshold(grayscale,0,255,cv2.THRESH_BINARY_INV+cv2.THRESH_OTSU)
        kernel = np.ones((50,50),np.uint8)
        closing = cv2.morphologyEx(img1, cv2.MORPH_CLOSE, kernel)
        b, g, r=cv2.split(img)
        rgba = [b,g,r,img1]
        dst = cv2.merge(rgba, 4)
        contours, hierarchy = cv2.findContours(img1,
cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)
        select = max(contours, key=cv2.contourArea)
        x,y,w,h = cv2.boundingRect(select)
        png = dst[y:y+h,x:x+w]
        gray = cv2.cvtColor(png, cv2.COLOR_BGR2GRAY)
#shape
        area = cv2.contourArea(select)
        perimeter = cv2.arcLength(select,True)
        aspect_ratio = float(w)/h
        rectangularity = w*h/area
        circularity = ((perimeter)**2)/area
#shape eccentricity
        dimension = png.shape
        height = png.shape[0]
        width = png.shape[1]
        mayor = max(height,width)
        minor = min(height,width)
        eccentricity = math.sqrt(1-((minor*minor)/(mayor*mayor)))
```

```
        eccentricity = math.sqrt(1-((minor*minor)/(mayor*mayor)))
#shape metric
        height1=png.shape[0]
        width1=png.shape[1]
        edge = cv2.Canny(img,100,200)
        k=0
        keliling=0
        while k<height1:
            l=0
            while l<width1:
                if edge[k,l]==255:
                    keliling=keliling+1
                l=l+1
            k=k+1
        k=0
        luas = 0
        while k<height1:
            l=0
            while l<width1:
                if img1[k,l]==255:
                    luas=luas+1
                l=l+1
            k=k+1
        metric = (4*math.pi*luas)/(keliling*keliling)
#hsv color
        hsv = cv2.cvtColor(png, cv2.COLOR_BGR2HSV)
        height=png.shape[0]
        width=png.shape[1]
        H=hsv[:,:,0]
        S=hsv[:,:,1]
        V=hsv[:,:,2]
        hue = np.reshape(H,(1,height*width))
        mode_h = stats.mode(hue[0])
        if int(mode_h[0])==0:
            mode_hue = np.mean(H)
        else:
            mode_hue = int(mode_h[0])
        mean_s = np.mean(S)
        mean_v = np.mean(V)
#glcm
        distance = [5]
        angles = [0,np.pi/4,np.pi/2,3*np.pi/4
```

```
        levels = 256
        symetric = True
        normed = True

        glcm = greycomatrix(gray, distance, angles, levels, symetric, normed)

        # Mengubah dari nama folder menjadi angka
        folder = classes[i]
        if folder == 'Bidara':
            fold = 1
        elif folder == 'Jambu':
            fold = 2
        elif folder == 'Miana':
            fold = 3
        elif folder == 'Sirih':
            fold = 4
        elif folder == 'Anting-anting':
            fold = 5
        elif folder == 'Bayam Duri':
            fold = 6
        else:
            fold = 7

        # Membuat dataset berdasarkan variabel kolom
        glcm_props = [propery for name in glcm_feature for propery in greycoprops(glcm,name)[0]]
        vector = [fold] +
[area,perimeter,w,h,aspect_ratio,rectangularity,circularity,eccentricity,metric,mode_hue,mean_s,me
an_v] + glcm_props
        df_temp = pd.DataFrame([vector],columns=names)
        df = df.append(df_temp)

    return df
```

Lampiran 4. Source Code Klasifikasi M-SVM

```
# Library
# Untuk mengolah data
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
# Untuk mengimport SVM
from sklearn import svm
# Untuk digunakan pada SVM dengan parameter tuning
from sklearn.model_selection import GridSearchCV
# Standarisasi dengan metode StandardScaler
from sklearn.preprocessing import StandardScaler
from sklearn.decomposition import PCA
# Untuk memanggil metrik akurasi
from sklearn import metrics
# Untuk visualisasi data
import seaborn as sns
import matplotlib.pyplot as plt
from mlxtend.plotting import plot_decision_regions
# Menghitung nilai akurasi untuk model
from sklearn.metrics import accuracy_score
from sklearn.metrics import classification_report, confusion_matrix

# Memuat data train pada dataframe
df_train = pd.read_csv('Dataset10Kelas.csv')
df_train
# Menyimpan fitur atribut ke dalam variabel X_train
X = df_train.drop(labels = ['class'],axis = 1)
#menyimpan class (label) pada y_train
y = df_train['class']

# Melakukan pembagian data dengan train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.20, random_state=42)

y_test = y_test.tolist()
y_train = y_train.tolist()

# inisiasi StandardScaler
scaler = StandardScaler()
# Melakukan standarisasi data
X_train = scaler.fit_transform(X_train)
X_test = scaler.transform(X_test)
```

```
parameters = [{'kernel': ['rbf'],
               'gamma': [1e-4, 1e-3, 0.01, 0.1, 0.2, 0.5],
               'C': [1, 10, 100, 1000]},
              {'kernel': ['linear'], 'C': [1, 10, 100, 1000]}]

# Menggunakan Gridsearch dengan memanggil class SVC
model_param = GridSearchCV(svm.SVC(),parameters,cv=5)
#melakukan training pada objek dan label
model_param.fit(X_train, y_train)

# Menampilkan hasil dari model SVM berdasarkan Hyper Parameter Tuning
means = model_param.cv_results_['mean_test_score']
stds = model_param.cv_results_['std_test_score']
for mean, std, params in zip(means, stds, model_param.cv_results_['params']):

from sklearn.multiclass import OneVsOneClassifier
model_ovo = OneVsOneClassifier(svm.SVC(C=1000, gamma=0.01, kernel='rbf'))
model_ovo.fit(X_train, y_train)
y_pred2 = model_ovo.predict(X_test)
metrics.accuracy_score(y_test, y_pred2)

conf2= confusion_matrix(y_test, y_pred2)
#confusion matrix
conf_matrix = pd.DataFrame(conf2, ('1','2','3','4','5','6','7','8','9','10'), ('1','2','3','4','5','6','7','8','9','10'))
# Plot confusion matrix
plt.figure()
heatmap = sns.heatmap(conf_matrix, annot=True, annot_kws={'size': 14}, fmt='d', cmap='Blues')
plt.title('Confusion Matrix untuk  Model SVM\nOneVsOneClassifier\n', fontsize=18)
plt.ylabel('True label', fontsize=14)
plt.xlabel('Predicted label', fontsize=14)
plt.show()

from sklearn.multiclass import OneVsRestClassifier
model_ovr = OneVsRestClassifier(svm.SVC(C=1000, gamma=0.01, kernel='rbf',
probability=True))
model_ovr.fit(X_train, y_train)
y_pred3 = model_ovr.predict(X_test)
metrics.accuracy_score(y_test, y_pred3)
```

Lampiran 5. Source Code Android

```
class ViewImageActivity : AppCompatActivity() {

    private lateinit var binding: ActivityViewImageBinding
    private val viewModel: ClassifyViewModel by viewModels()
    private var image: File? = null
    private var imageCapture: ImageCapture? = null

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        binding = ActivityViewImageBinding.inflate(layoutInflater)
        setContentView(binding.root)
        observeData()
        binding.btnTakephoto.setOnClickListener {
            val i = Intent(this, DetectorActivity::class.java)
            startActivityForResult(i, DETECTOR_ACTIVITY_REQUEST_CODE)
        }

        binding.btnGallery.setOnClickListener {
            startGallery()
        }

        binding.btnSubmit.setOnClickListener {
            image?.let { img -> viewModel.getPrediction(img) }
            binding.btnSubmit.isInvisible = true
            binding.progressBar.isVisible = true
        }
    }

    private fun observeData() {
        viewModel.herbal.observe(this) {
            Log.d("RESULT", it.toString())
            showDialog(it)

            binding.btnSubmit.isInvisible = false
            binding.progressBar.isVisible = false
        }
    }
```

```
private fun showDialog(res: PredictionResponse) {
        val result = when (res.result) {
            "Bidara" -> resources.getStringArray(R.array.bidara)
            "Jambu" -> resources.getStringArray(R.array.jambu)
            "Miana" -> resources.getStringArray(R.array.miana)
            "Sirih" -> resources.getStringArray(R.array.sirih)
            "Anting-anting" -> resources.getStringArray(R.array.anting)
            "Bayam Duri" -> resources.getStringArray(R.array.bayamduri)
            "Kirinyuh" -> resources.getStringArray(R.array.kirinyuh)
            "Daun Ungu" -> resources.getStringArray(R.array.daunungu)
            "Sidaguri" -> resources.getStringArray(R.array.sidaguri)
            "Sirsak" -> resources.getStringArray(R.array.sirsak)
            else -> resources.getStringArray(R.array.jambu)
        }
        MaterialAlertDialogBuilder(this)
            .setTitle(res.result)
            .setMessage("Probabilitas : \n" +
                    "${res.probabilitas[0].name}, ${res.probabilitas[0].presentage}\n" +
                    "${res.probabilitas[1].name}, ${res.probabilitas[1].presentage}\n" +
                    "${res.probabilitas[2].name}, ${res.probabilitas[2].presentage}\n")
            .setPositiveButton("lihat detail") {_,_ ->
                val intent = Intent(this, ResultActivity::class.java)
                intent.putExtra("RESULT", result)
                startActivity(intent)}
            .create()
            .show()}
    override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {
        super.onActivityResult(requestCode, resultCode, data)
        if (requestCode == DETECTOR_ACTIVITY_REQUEST_CODE) {
            if (resultCode === RESULT_OK) {
                val isTake = data?.extras?.getBoolean(KEY_IS_TAKE) ?: false
                if (isTake) {
                    Toast.makeText(this, "LOAD", Toast.LENGTH_SHORT).show()
                    val imgString = getExternalFilesDir("/Pictures/bitmap_test.jpg") as File
                    image = imgString
                    Log.d("CAMERA", imgString.toString())
                    binding.imageview.setImageBitmap(
                        BitmapFactory.decodeFile(imgString?.path)
                    )
                }
            }
        }
    }
```

```
    fun uriToFile(selectedImg: Uri, context: Context): File {
        val contentResolver: ContentResolver = context.contentResolver
        val myFile = createCustomTempFile(context)
        val inputStream = contentResolver.openInputStream(selectedImg) as InputStream
        val outputStream: OutputStream = FileOutputStream(myFile)
        val buf = ByteArray(1024)
        var len: Int
        while (inputStream.read(buf).also { len = it } > 0) outputStream.write(buf, 0, len)
        outputStream.close()
        inputStream.close()
        return myFile
    }
    private val launcherIntentCamera = registerForActivityResult(
        ActivityResultContracts.StartActivityForResult()
    ) { result ->
        if (result.resultCode == RESULT_OK) {
            val selectedImg: Uri = result.data?.data as Uri
            val myFile = uriToFile(selectedImg, this)
            image = myFile
            binding.imageview.setImageURI(selectedImg)
        }
    }
    private val launcherIntentGallery = registerForActivityResult(
        ActivityResultContracts.StartActivityForResult()
    ) { result ->
        if (result.resultCode == RESULT_OK) {
            val selectedImg: Uri = result.data?.data as Uri
            val myFile = uriToFile(selectedImg, this)
            image = myFile
            binding.imageview.setImageURI(selectedImg)
        }}
    private fun startGallery() {
        val intent = Intent()
        intent.action = ACTION_GET_CONTENT
        intent.type = "image/*"
        val chooser = Intent.createChooser(intent, "Choose a Picture")
        launcherIntentGallery.launch(chooser)
    }
    companion object {
        const val DETECTOR_ACTIVITY_REQUEST_CODE = 1
    }
}
```

Lampiran 6. Contoh Dataset Hasil Ekstraksi Fitur pada Kelas 1

class	area	perimeter	physiological_length	physiological_width	aspect_ratio	rectangularity	circularity	eccentricity
1	80600	1423,047894	291	417	0,697841727	1,505545906	25,12487978	0,716251998
1	60297,5	1219,952442	266	354	0,751412429	1,561656785	24,68234937	0,659832828
1	47739	3514,075575	239	395	0,605063291	1,977523618	258,6716762	0,796177376

metric	hue	saturation	value	correlation 0	correlation 45	correlation 90	correlation 135	homogeneity 0
0,112800899	44,2798009	57,26144033	210,7256463	0,865226179	0,879640077	0,912960404	0,874271329	0,359521448
0,032584644	37,73961387	57,62006712	209,0812519	0,86778389	0,870994996	0,900426376	0,873174895	0,388723092
0,054857413	29,9454584	46,788454	234,4634606	0,801494872	0,826800833	0,888906011	0,816864078	0,423188712

homogeneity 45	homogeneity 90	homogeneity 135	dissimilarity 0	dissimilarity 45	dissimilarity 90	dissimilarity 135	contrast 0	contrast 45
0,375899225	0,409597403	0,365597311	11,86416461	10,49491694	7,994761952	11,05095713	603,0811155	535,9878428
0,386094385	0,397344998	0,380998388	11,5748317	11,23923664	9,80885236	11,58157034	605,3467649	587,0386914
0,429046438	0,45612304	0,427687086	10,98067727	10,07405997	7,457247076	10,07025086	427,8223088	373,0392447

contrast 90	contrast 135	ASM 0	ASM 45	ASM 90	ASM 135	energy 0	energy 45	energy 90	energy 135
389,8278451	559,8960019	0,092441476	0,090431713	0,096049827	0,09011495	0,304041898	0,300718661	0,309919065	0,300191522
456,7427774	577,1190403	0,10455169	0,102141982	0,109252006	0,10154653	0,323344538	0,319596593	0,330532912	0,318663663
239,0862354	394,4411928	0,142856897	0,140378664	0,148077915	0,141044957	0,377964148	0,374671409	0,384808933	0,375559525

entropy 0	entropy 45	entropy 90	entropy 135
1,582703425	1,547705673	1,338909612	1,519169664
1,154359724	1,132043873	1,088521288	1,144885795
1,421398071	1,408225595	1,277861578	1,389649596

Lampiran 7. Contoh Dataset Hasil Seleksi Fitur pada 10 Kelas

1. Fitur Bentuk

class	physiological_width	aspect_ratio	rectangularity	eccentricity	class	physiological_width	aspect_ratio	rectangularity	eccentricity
1	417	0,697841727	1,505545906	0,716251998	6	425	0,583529412	1,684136521	0,812092006
1	354	0,751412429	1,561656785	0,659832828	6	400	0,75	1,851494696	0,661437828
1	395	0,605063291	1,977523618	0,796177376	6	396	0,757575758	1,838511239	0,652747249
1	318	0,786163522	1,400473871	0,618018541	6	461	0,598698482	1,956348261	0,800974487
1	379	0,741424802	1,770336201	0,671035962	6	380	0,781578947	1,683899557	0,623806339
2	433	0,591224018	1,566401944	0,806507384	7	457	0,584245077	1,501578258	0,811577286
2	467	0,565310493	1,591480298	0,824878201	7	468	0,626068376	1,478976007	0,779768163
2	433	0,579676674	1,453379603	0,814846583	7	476	0,556722689	1,575096149	0,830698409
2	445	0,543820225	1,625460363	0,839201742	7	450	0,64	1,400702513	0,768374908
2	445	0,557303371	1,597776201	0,83030895	7	434	0,578341014	1,450065559	0,815795116
3	343	0,874635569	1,257124008	0,484781004	8	476	0,432773109	1,496501255	0,901502876
3	309	0,928802589	1,278663706	0,370574892	8	473	0,539112051	1,536115233	0,842234051
3	270	1,051851852	1,279983975	0,310099139	8	457	0,612691466	1,536503362	0,790322192
3	292	0,996575342	1,32344833	0,082689702	8	456	0,583333333	1,505314693	0,812232862
3	251	1,135458167	1,299455041	0,473671215	8	468	0,446581197	1,683380805	0,894743111
4	430	0,481395349	1,492842707	0,876503576	9	469	0,611940299	2,112745252	0,790903958
4	452	0,511061947	1,556169937	0,859543883	9	480	0,489583333	2,14963601	0,871956513
4	366	0,579234973	1,635736948	0,815160626	9	468	0,626068376	1,537204258	0,779768163
4	444	0,459459459	1,582320828	0,888198742	9	451	0,66518847	1,602586881	0,746675498
4	402	0,487562189	1,947741824	0,873088261	9	475	0,564210526	2,635992794	0,82563096
5	471	0,435244161	1,276608404	0,900312457	10	476	0,43487395	1,430892892	0,900491337
5	479	0,467640919	1,362419687	0,883918532	10	475	0,492631579	1,386748844	0,870237972
5	478	0,546025105	1,344454682	0,837768814	10	454	0,475770925	1,327772964	0,879569228
5	480	0,472916667	1,31049745	0,881107159	10	454	0,449339207	1,465941737	0,893361224
5	478	0,479079498	1,283101142	0,877771516	10	468	0,521367521	1,461807289	0,853332238

2. Fitur Warna

class	hue	value	class	hue	value
1	44,2798009	210,7256463	9	30,26930306	211,4105183
1	37,73961387	209,0812519	9	29,83942376	214,1935993
1	29,9454584	234,4634606	9	34,04935679	184,2485196
1	41,56221384	190,8920252	9	37,67766445	209,9420177
1	41,43157213	195,6373018	9	25,58982718	220,4584368
2	26,272382	197,76733	10	54,48936386	159,2850648
2	28,29772565	200,5058887	10	39,002861	175,9363113
2	29,20318725	191,8197326	10	60,9272006	133,0668645
2	43,21547962	197,2810103	10	28,90673318	158,6755528
2	37,90114172	218,2439743	10	35,26416912	179,6990157
3	40,69008746	164,6715063			
3	52,6515905	184,6532143			
3	52,91387585	178,3817553			
3	50,27332533	159,7033258			
3	44,44913679	150,8705109			
4	82,51458263	168,5174362			
4	88,80294411	166,6393326			
4	84,70582019	188,3740076			
4	90,57017311	204,4084415			
4	88,60064474	194,1628338			
5	54,73807674	175,2067733			
5	40,92653967	181,5190967			
5	51,53855464	177,1894788			
5	54,44130874	164,274844			
5	52,54641793	153,408708			
6	28,07091082	209,7944213			
6	27,26526667	195,1381333			
6	22,11917508	203,5787121			
6	33,13743752	199,448348			
6	29,89026227	184,2583555			
7	48,72622297	196,311935			
7	54,42875791	189,8724585			
7	55,04771682	198,9106707			
7	16,31783179	186,3746836			
7	8,202976114	164,1839554			
8	53,98274455	219,7243514			
8	38,68503088	184,3582805			
8	29,4928884	205,6039934			
8	32,05413204	214,6748615			
8	38,11263444	188,3135812			

3. Fitur Tekstur

cls	correlation 90	homogeneity 0	homogeneity 45	homogeneity 90	homogeneity 135	cls	correlation 90	homogeneity 0	homogeneity 45	homogeneity 90	homogeneity 135
1	0,912960404	0,359521448	0,375899225	0,409597403	0,365597311	6	0,908042975	0,454745268	0,45573342	0,480590681	0,456731263
1	0,900426376	0,388723092	0,386094385	0,397344998	0,380998388	6	0,891980417	0,451707867	0,447737614	0,463058062	0,449504736
1	0,888906011	0,423188712	0,429046438	0,45612304	0,427687086	6	0,879824306	0,448073581	0,445795467	0,458841657	0,446801828
1	0,882402041	0,304781912	0,300142393	0,31587648	0,297294368	6	0,931000393	0,498012264	0,496347817	0,513327237	0,492678483
1	0,882790026	0,35541737	0,365241463	0,373677239	0,346229444	6	0,887567562	0,41059866	0,404811743	0,418185662	0,408717428
2	0,868882435	0,36243875	0,360025438	0,376220587	0,363435979	7	0,772113496	0,324986786	0,321887045	0,332180413	0,319124989
2	0,919384686	0,383741524	0,389389466	0,402605319	0,385612319	7	0,836625358	0,333444876	0,329179162	0,342134901	0,328882925
2	0,890894197	0,369276218	0,362903985	0,376493228	0,36461649	7	0,829763454	0,369139898	0,363621732	0,376247093	0,363725678
2	0,907173769	0,389188529	0,388795583	0,410070078	0,393325397	7	0,820572607	0,298132112	0,287937661	0,299979809	0,290370198
2	0,869320919	0,387544663	0,390150392	0,404148366	0,389187419	7	0,912107919	0,327817175	0,317695228	0,334943851	0,3237478
3	0,903664839	0,341281473	0,336366753	0,354025005	0,330296813	8	0,900889919	0,358893368	0,358819274	0,388038512	0,367290136
3	0,863117008	0,247044858	0,236457524	0,248738039	0,239491822	8	0,903894864	0,366409949	0,361700609	0,384258299	0,368479357
3	0,830656917	0,236186824	0,225079851	0,234862779	0,224166659	8	0,884402248	0,363742418	0,36703149	0,392723019	0,367614069
3	0,827437144	0,245357183	0,236602987	0,244810421	0,234693036	8	0,890546058	0,352481193	0,355400587	0,378337026	0,354782515
3	0,782747413	0,212270299	0,199220966	0,206554064	0,198177632	8	0,900360451	0,395635895	0,388498094	0,419396145	0,403124623
4	0,834284155	0,276072524	0,274865876	0,293358482	0,274155704	9	0,878409231	0,320419966	0,32454289	0,339040311	0,318351045
4	0,88350062	0,323459511	0,321573094	0,338827432	0,324350745	9	0,871081839	0,325733576	0,325850613	0,348657176	0,328501256
4	0,772485025	0,311228828	0,309803677	0,326099834	0,307731096	9	0,918276396	0,307662729	0,308015397	0,320665344	0,302870876
4	0,786904885	0,33203276	0,331732548	0,351441729	0,331477647	9	0,877476477	0,29825743	0,292908223	0,308117931	0,291915473
4	0,794080838	0,29980113	0,30002713	0,329294476	0,301620588	9	0,817701521	0,365501499	0,36537622	0,381899012	0,365975089
5	0,871204719	0,221525765	0,221256344	0,241968462	0,22196359	10	0,941585545	0,41199447	0,416649817	0,448092003	0,41217693
5	0,885224564	0,270693095	0,271112227	0,290615979	0,271184448	10	0,942593626	0,322859394	0,324918986	0,346169932	0,324427084
5	0,917029058	0,319301114	0,303774943	0,318345923	0,309181158	10	0,894963859	0,239827684	0,237921976	0,256986514	0,237588342
5	0,911822654	0,251944715	0,251250604	0,272994875	0,254070099	10	0,929231544	0,363365461	0,366465165	0,39531129	0,365629559
5	0,904214296	0,234724928	0,234629901	0,254060281	0,234668201	10	0,943925998	0,384168389	0,391198543	0,412742128	0,38168843

class	dissimilarity 0	dissimilarity 45	dissimilarity 90	dissimilarity 135	class	dissimilarity 0	dissimilarity 45	dissimilarity 90	dissimilarity 135
1	11,86416461	10,49491694	7,994761952	11,05095713	6	9,651958364	9,128772244	7,230894777	9,274327324
1	11,5748317	11,23923664	9,80885236	11,58157034	6	17,27467797	17,03818591	13,90914768	16,90386978
1	10,98067727	10,07405997	7,457247076	10,07025086	6	16,98258004	16,71083839	14,80796249	16,53319774
1	15,19743294	15,31555849	13,21860703	15,27884366	6	12,4115872	11,95099916	9,969933893	12,69425763
1	19,57950939	17,49447894	15,34801226	20,43397353	6	17,5635274	17,82195374	15,63050954	17,10942379
2	17,37723471	17,36832612	14,60162639	15,46983572	7	21,0990696	21,36293741	20,66176958	22,58404888
2	13,99146776	11,33537963	10,70234816	14,01959628	7	16,99268459	17,24323619	16,54312652	17,62637961
2	11,70739218	11,38669158	10,09166139	11,31523268	7	16,20339367	17,40242061	16,2560109	16,77189266
2	16,44098042	15,74758475	12,28708678	14,54472265	7	17,42683942	19,05595907	17,95300406	18,65447167
2	12,61872659	11,48179436	10,58610704	12,35843463	7	15,38422127	16,41410413	13,96419915	15,35589869
3	7,80886495	7,797725823	7,35377712	8,320009168	8	13,27850245	13,0392998	9,814359038	11,73687909
3	16,21218068	16,93741528	15,41629608	16,49862712	8	16,0125074	16,39029384	12,68995308	14,87364826
3	19,3872428	19,80393394	19,13849322	20,43223684	8	14,35295803	13,92020187	10,79337863	13,23075631
3	23,03870103	23,23493757	22,55200738	24,12174555	8	14,64250185	13,99967912	10,70056516	13,22351888
3	28,26992032	29,39196335	28,64119241	29,73014249	8	17,76096657	18,30992431	13,94681038	16,64817073
4	29,39737509	28,93470015	25,28301222	28,81594163	9	16,0327088	14,32578745	13,11237685	16,59458186
4	27,60603806	27,57128107	23,5463649	25,64929791	9	18,08206522	18,07265634	13,46608735	16,13653643
4	28,63292944	28,05042765	26,59844248	28,90396568	9	19,20959461	17,13173398	14,73245417	19,21970081
4	25,33273349	25,54427273	22,7785743	24,81570455	9	14,78474952	14,97864895	12,71812407	14,59699045
4	32,62457347	31,25740683	26,31599239	31,82872697	9	18,06696818	17,79863926	15,08316926	17,18791417
5	19,92283439	19,3220088	16,92377264	19,22666113	10	11,23026042	10,32905778	6,639999179	9,976381815
5	19,90936216	19,06667943	16,37038502	18,8737799	10	12,95776603	12,28474107	9,205928351	12,11708668
5	12,42067371	12,6276084	10,6709031	11,73476826	10	24,35553375	23,75035639	20,31866081	23,81324948
5	19,40587462	18,34331311	15,3469418	18,42318273	10	15,8279946	15,02145556	9,898281584	14,53743333
5	20,63422744	19,87497421	16,74846054	19,38852321	10	11,79346815	10,42116559	7,73558935	11,54819504

class	contrast 0	contrast 45	contrast 90	contrast 135	class	contrast 0	contrast 45	contrast 90	contrast 135
1	603,0811155	535,9878428	389,8278451	559,8960019	6	633,2699879	584,4508197	444,2482911	614,7809081
1	605,3467649	587,0386914	456,7427774	577,1190403	6	1419,161983	1409,536275	1045,670312	1358,616059
1	427,8223088	373,0392447	239,0862354	394,4411928	6	1019,262378	989,2996587	811,0118329	980,7081495
1	853,6618149	873,1862022	673,1123067	846,7975636	6	948,1435512	872,1708151	663,6273996	965,542919
1	1494,149641	1372,835456	980,2867528	1500,302161	6	1291,976451	1314,589963	1103,497912	1267,168969
2	1114,222224	1137,921865	809,9113153	885,3610556	7	1190,691282	1188,43884	1102,725059	1272,567279
2	726,3371061	508,5205433	483,125164	737,574406	7	946,5344403	914,0966845	874,1038928	1003,123546
2	766,2788637	713,9221615	561,348559	712,9683758	7	814,1245637	889,7822667	786,7831431	838,6337343
2	1059,461167	985,6400751	653,8437547	877,6920768	7	907,9215548	1016,417104	911,6592463	989,464015
2	579,3796458	479,1842404	421,2717742	580,4941173	7	1076,243762	1081,907222	827,7156827	1043,605687
3	539,0117508	549,568315	531,2393984	601,2491728	8	880,7589155	860,9894802	577,6824769	768,538251
3	842,0484175	928,7995945	821,618593	905,7653594	8	1301,800211	1246,261716	832,2022541	1191,152227
3	1168,232749	1199,139541	1150,258956	1303,295502	8	888,2728785	867,6087037	589,0636378	791,9050533
3	1504,458928	1517,670502	1450,325682	1662,455974	8	728,9686597	708,6650341	453,5675275	623,1955347
3	2001,466221	2158,22868	2057,811239	2173,161842	8	1753,934735	1698,483347	1164,537466	1648,471005
4	2549,908197	2409,100985	1883,114237	2419,209892	9	827,730073	701,0103423	581,2033446	824,2300999
4	2506,306308	2337,154569	1795,999584	2216,224856	9	1000,595254	995,5154334	580,54357	797,9921878
4	2132,061482	2064,44078	1864,028145	2136,173276	9	1228,908253	959,113374	718,9268091	1224,708724
4	1789,049946	1847,991455	1486,252311	1668,174523	9	728,3605923	699,9313819	540,7578251	691,2273339
4	2879,05681	2651,873037	1890,050224	2723,422739	9	1161,200224	1075,634176	824,538536	1058,905874
5	1157,024257	1078,312644	782,9955407	1062,84478	10	1241,677074	1148,780986	641,9790968	1028,357007
5	1227,723587	1094,152536	797,2263035	1121,347904	10	856,7569662	796,7469399	468,0267867	740,4064248
5	832,8949317	824,8183027	591,3027549	721,049812	10	1989,639184	1901,4713	1306,665522	1858,635996
5	1367,852628	1221,643404	800,6261442	1214,614274	10	1596,700197	1494,637833	744,250393	1399,636122
5	1534,306579	1413,05112	920,0001939	1323,987079	10	1021,676367	827,5261225	510,1431771	973,5627604

class	ASM 0	ASM 45	ASM 90	ASM 135	class	ASM 0	ASM 45	ASM 90	ASM 135
1	0,092441476	0,090431713	0,096049827	0,09011495	6	0,136464429	0,135355832	0,145671364	0,135058981
1	0,10455169	0,102141982	0,109252006	0,10154653	6	0,173256686	0,169509842	0,178089122	0,17026127
1	0,142856897	0,140378664	0,148077915	0,141044957	6	0,177794148	0,175168379	0,184494274	0,175990738
1	0,059822165	0,056757431	0,063475736	0,056682588	6	0,202515183	0,201379573	0,213407552	0,200750009
1	0,103327792	0,100956798	0,11047772	0,100250018	6	0,141288053	0,137263736	0,144248677	0,138193746
2	0,103641512	0,101983814	0,111328654	0,1026469	7	0,088040658	0,086392152	0,093178919	0,086070079
2	0,109101184	0,108183991	0,115475717	0,108149805	7	0,084430538	0,083428204	0,09042129	0,082617685
2	0,078251062	0,076848537	0,085578896	0,07730831	7	0,109529961	0,108259832	0,116267297	0,107905116
2	0,120143637	0,119500628	0,128546266	0,118786363	7	0,065009704	0,062505494	0,06819847	0,062621666
2	0,113754574	0,113783969	0,122064487	0,112321081	7	0,07491663	0,073283425	0,080871239	0,073241967
3	0,03247278	0,030213767	0,032534683	0,029851458	8	0,082500675	0,081494193	0,093141788	0,082875853
3	0,034085214	0,031119376	0,034771252	0,03119275	8	0,094329911	0,094686153	0,105793385	0,09416474
3	0,034605398	0,03100054	0,034276406	0,030774197	8	0,094603464	0,093976974	0,103077331	0,093491437
3	0,04406713	0,040581831	0,044348648	0,03997081	8	0,088136313	0,086899485	0,096169482	0,087233911
3	0,033300091	0,02867985	0,031470488	0,028815612	8	0,120012348	0,119342195	0,134973474	0,120800386
4	0,06285077	0,06199916	0,070634943	0,061890168	9	0,073650002	0,073773834	0,080755915	0,073917016
4	0,085198601	0,084783035	0,092701732	0,084781662	9	0,080129176	0,080639835	0,088646108	0,079715309
4	0,084413909	0,083262922	0,092850649	0,082581911	9	0,069538141	0,069450276	0,074131476	0,06768787
4	0,094773118	0,094776606	0,106361381	0,094273691	9	0,055837419	0,05562274	0,059610787	0,054799355
4	0,077807099	0,07784271	0,093884771	0,078925166	9	0,107261645	0,107665117	0,115976596	0,106636155
5	0,03074826	0,030318787	0,037499213	0,030656387	10	0,059084009	0,059347584	0,069725886	0,059791289
5	0,051116626	0,051032522	0,059747286	0,051037099	10	0,056392627	0,055676371	0,062711194	0,055591628
5	0,050173121	0,049133261	0,055462389	0,049428928	10	0,040157898	0,039042324	0,046337225	0,039338861
5	0,039009608	0,038805107	0,046913389	0,039154366	10	0,072092022	0,071520085	0,085467085	0,07216698
5	0,033096666	0,033238288	0,040415188	0,033187895	10	0,078307194	0,07988455	0,088270884	0,077569096

class	energy 0	energy 45	energy 90	energy 135	class	energy 0	energy 45	energy 90	energy 135
1	0,304041898	0,300718661	0,309919065	0,300191522	6	0,369410922	0,367907369	0,381669182	0,367503716
1	0,323344538	0,319596593	0,330532912	0,318663663	6	0,416241139	0,41171573	0,422006069	0,412627277
1	0,377964148	0,374671409	0,384808933	0,375559525	6	0,421656434	0,418531217	0,429527966	0,419512501
1	0,2445857	0,238238182	0,251943914	0,238081054	6	0,450016869	0,448753355	0,461960553	0,448051346
1	0,321446406	0,317736995	0,332381888	0,316622833	6	0,375883031	0,370491209	0,379800838	0,371744194
2	0,321934018	0,319349048	0,333659488	0,320385548	7	0,29671646	0,29392542	0,305252223	0,293377025
2	0,330304683	0,328913349	0,339817181	0,328861376	7	0,290569335	0,288839409	0,300701331	0,287432922
2	0,279733912	0,277215687	0,292538708	0,27804372	7	0,33095311	0,329028618	0,340979907	0,328489141
2	0,346617422	0,345688628	0,358533493	0,344653976	7	0,254970006	0,250010989	0,261148368	0,250243214
2	0,33727522	0,337318795	0,349377284	0,335143373	7	0,273709024	0,270709114	0,28437869	0,270632531
3	0,180202052	0,173821078	0,180373731	0,172775744	8	0,287229308	0,285471878	0,305191396	0,287881664
3	0,184621813	0,176406849	0,186470513	0,176614693	8	0,307131749	0,307711152	0,325258951	0,306862737
3	0,186025261	0,176069701	0,185138882	0,17542576	8	0,307576761	0,306556641	0,321056586	0,305763694
3	0,209921723	0,201449325	0,210591187	0,199927013	8	0,296877606	0,294787185	0,310112048	0,295353875
3	0,182483125	0,169351263	0,177399232	0,169751617	8	0,346427983	0,345459397	0,367387362	0,347563499
4	0,250700559	0,248996306	0,265772352	0,248777345	9	0,271385338	0,27161339	0,284175853	0,271876841
4	0,291887995	0,291175266	0,304469592	0,291172907	9	0,283070973	0,283971539	0,297734963	0,282338996
4	0,290540718	0,288553152	0,304714045	0,287370685	9	0,263700855	0,263534203	0,272270961	0,260168925
4	0,307852429	0,307858094	0,326130926	0,307040211	9	0,236299426	0,235844737	0,244153204	0,23409262
4	0,278939238	0,279003065	0,306406219	0,280936231	9	0,327508237	0,32812363	0,340553367	0,326551918
5	0,175351817	0,174122908	0,193647136	0,175089654	10	0,243072024	0,243613596	0,264056596	0,244522573
5	0,226089863	0,225903789	0,24443258	0,225913918	10	0,237471318	0,23595841	0,250422033	0,235778768
5	0,223993574	0,221660238	0,235504542	0,222326175	10	0,200394355	0,197591305	0,215260831	0,198340266
5	0,197508501	0,196990118	0,216594989	0,197874623	10	0,268499575	0,267432393	0,292347541	0,268639126
5	0,181924892	0,182313708	0,20103529	0,18217545	10	0,279834226	0,282638551	0,297104164	0,27851229

class	entropy 0	entropy 45	entropy 90	entropy 135	class	entropy 0	entropy 45	entropy 90	entropy 135
1	1,582703425	1,547705673	1,338909612	1,519169664	6	1,01767043	1,015934871	0,965843501	1,005645748
1	1,154359724	1,132043873	1,088521288	1,144885795	6	2,128335303	2,09749536	1,974195276	2,109682175
1	1,421398071	1,408225595	1,277861578	1,389649596	6	1,924614978	1,91586872	1,880330531	1,909592996
1	1,378941076	1,38379477	1,344248646	1,37872108	6	2,043376251	2,039927533	1,982896655	2,083278489
1	2,17709725	2,075762574	2,048725229	2,178858376	6	2,004249702	2,002981717	1,939500159	1,981273133
2	2,063496437	2,057250058	2,019724065	1,983173417	7	2,271708428	2,259654918	2,266739863	2,271911133
2	2,061841109	1,906675365	1,931102885	2,075328925	7	2,110359022	2,115304512	2,119339399	2,112577375
2	1,057224082	1,047740238	1,065973539	1,044830545	7	1,880841278	1,881966128	1,885958195	1,883190252
2	1,955288448	1,93438873	1,839686359	1,883339018	7	2,118033924	2,141127846	2,13285482	2,136161799
2	1,609860418	1,575488574	1,557586256	1,598636908	7	1,402004934	1,409034512	1,385209489	1,394358283
3	0,749074842	0,742776303	0,729831066	0,756423459	8	1,116968814	1,118916767	1,05458165	1,082373077
3	1,645305946	1,648415331	1,605673582	1,626954665	8	1,640948603	1,674904551	1,605464298	1,58797467
3	1,770865009	1,772036296	1,762061056	1,772195741	8	1,688256001	1,693572643	1,590010572	1,61164863
3	1,845320389	1,837936149	1,832945946	1,846758192	8	1,842423176	1,830237554	1,71610416	1,787649256
3	2,152806023	2,154217732	2,147324059	2,160670343	8	1,561948178	1,632970831	1,55886779	1,524512904
4	2,62327399	2,627579587	2,596052201	2,62273601	9	2,222911181	2,136968661	2,138436837	2,287425737
4	2,74825699	2,768663318	2,711318638	2,712104219	9	2,352140436	2,359089997	2,147931023	2,257396845
4	2,516415574	2,501727321	2,503149669	2,521591112	9	2,722685733	2,643398589	2,49905142	2,707014837
4	2,352578948	2,350793548	2,326868975	2,345101449	9	2,133084251	2,15067521	2,027142899	2,112643026
4	2,447440207	2,441938295	2,423460838	2,451051372	9	2,260991742	2,275414264	2,207370845	2,262378524
5	1,787940589	1,787230248	1,770787621	1,783127808	10	0,716437615	0,706025198	0,594985041	0,707744512
5	2,025305723	2,028123555	1,988086125	1,997441709	10	1,652545777	1,639359468	1,49859332	1,630822223
5	1,353496274	1,381937197	1,346422708	1,332151641	10	2,101106842	2,097518163	2,079590356	2,105714729
5	1,973420229	1,942795609	1,894904241	1,958918635	10	1,088370508	1,08841179	1,06050983	1,086296438
5	2,199393018	2,195887865	2,167489239	2,171245589	10	1,322776678	1,317178748	1,182024006	1,310596142

Lampiran 8. Lembar Perbaikan Skripsi

LEMBAR PERBAIKAN SKRIPSI

"KLASIFIKASI TANAMAN OBAT MENGGUNAKAN MULTICLASS SUPPORT VECTOR MACHINE BERBASIS ANDROID"

OLEH:

RIEKA ZALZABILLAH PUTRI
D121 17 1001

Skripsi ini telah dipertahankan pada Ujian Akhir Sarjana tanggal 10 Maret 2023.

Telah dilakukan perbaikan penulisan dan isi skripsi berdasarkan usulan dari penguji dan pembimbing skripsi.

Persetujuan perbaikan oleh tim penguji:

	Nama	Tanda Tangan
Ketua	Prof. Dr. Ir. Indrabayu, S.T., M.T., M.Bus.Sys., IPM., ASEAN.Eng.	
Sekretaris	A. Ais Prayogi Alimuddin, S.T., M.Eng.	
Anggota	Dr. Eng. Muhammad Niswar, S.T., M.IT.	
	Dr. Ir. Ingrid Nurtanio, M.T	

Persetujuan Perbaikan oleh pembimbing:

Pembimbing	Nama	Tanda Tangan
I	Prof. Dr. Ir. Indrabayu, S.T., M.T., M.Bus.Sys., IPM., ASEAN.Eng.	
II	A. Ais Prayogi Alimuddin, S.T., M.Eng.	