

# **Prediksi Spesies Burung Dengan Random Forest**

## **Mata Kuliah Kecerdasan Buatan**

<sup>1</sup> Yosef Alfredo Khawarga, <sup>2</sup> Anisha dwi nur fadlilah

<sup>1</sup> [yosef.alfredo@student.ukdc.ac.id](mailto:yosef.alfredo@student.ukdc.ac.id)

<sup>2</sup> [anisha.fadlilah@student.ukdc.ac.id](mailto:anisha.fadlilah@student.ukdc.ac.id)

### *Abstrak*

*Random forest adalah sebuah algoritma supervised learning yang menggunakan metode pembelajaran ensemble untuk klasifikasi dan regresi. Random forest merupakan sebuah pemodelan dengan menggunakan teknik mengantongi dan bukan teknik meningkatkan. pohon-pohon di hutan acak berjalan secara paralel. Hal Ini beroperasi dengan membangun banyak pohon keputusan pada saat pelatihan dan keluaran kelas yang merupakan modus dari kelas (klasifikasi) atau prediksi rata-rata (regresi) dari setiap pohon. Pada laporan ini kami akan menjelaskan mengenai bagaimana membuat sebuah prediksi mengenai spesies burung menggunakan data performa dari spies burung yang ada prediksi ini kami buat menggunakan metode Random forest dan juga menggunakan confussion matrix. Hal pertama yang harus dilakukan yaitu mempunyai dataset dari jenis – jenis burung ,untuk datasetnya nanti akan kami sertakan pada hasil dan juga pembahasan .Hasil dari laporan ini menunjukan bahwa penggunaan metode random forest dan juga confussion matrix cukup baik untuk memprediksi spesies burung dengan berdasarkan ciri – ciri dan karakteristik masing- masing burung.*

*Kata Kunci: Random Forest, confussion matrix, pohon keputusan.*

### *Abstract*

*Random forest is a supervised learning algorithm that uses ensemble learning methods for classification and regression. Random forest is a modelling using pocketing technique and not increasing technique. trees in random forest run in parallel. It operates by constructing multiple decision trees at the time of training and class output which is the mode of class (classification) or mean prediction (regression) of each tree. In this report, we will explain how to make a prediction about bird species using performance data from bird species. We made this prediction using the Random Forest method and also using a confusion matrix. The first thing to do is to have a dataset of bird species, For the dataset, we will include it in the results and discussion. The results of this report show that the use of the random forest method and the confusion matrix is good enough to predict bird species based on the characteristics and characteristics of each bird.*

*Keywords: Random Forest, confusion matrix, decision tree.*

## 1. Pendahuluan

Random forest atau yang biasa kita sebut dengan hutan acak adalah suatu algoritma yang digunakan pada klasifikasi data dalam jumlah yang besar. Klasifikasi random forest dilakukan melalui penggabungan pohon dengan melakukan training pada sampel data yang dimiliki. Penggunaan Random Forest ini umumnya digunakan pada riset pengambilan keputusan adalah sebuah diagram alir yang berbentuk seperti pohon yang memiliki sebuah *root node* yang digunakan untuk mengumpulkan data. Dalam ekosistem alam tentu kita melihat adanya berbagai macam binatang salah satunya burung. Burung termasuk binatang unggas yang dapat bertelur, burung juga memiliki berbagai macam spesies. Spesies burung ini lah yang akan kami riset sebagai bahan pertimbangan apakah spesies burung ini sama dengan satu sama yang lain, dan apa perbedaan spesies satu dengan spesies yang lain. Riset ini tentu memerlukan data dari berbagai spesies burung, dan riset yang akan kami uji ini menggunakan metode random forest sebagai metode pengujinya.

Pengklasifikasian spesies burung pada laporan ini tidak selesai hanya dengan Teknik random forest karena ada beberapa dari spesies burung yang ciri -cirinya sangat sama dan sulit dibedakan dengan Teknik random forest oleh karena akan dilanjutkan dengan menggunakan confusion matrix atau matrix konfusi. Matriks Confusion adalah matriks  $N \times N$  yang digunakan untuk mengevaluasi kinerja model klasifikasi, di mana  $N$  adalah jumlah kelas target. Matriks tersebut membandingkan nilai target aktual dengan yang diprediksi oleh model pembelajaran mesin. Ini memberi kita pandangan holistik tentang seberapa baik kinerja model klasifikasi kita dan jenis kesalahan apa yang dibuatnya. Untuk masalah klasifikasi biner, kita akan memiliki matriks  $2 \times 2$  seperti yang ditunjukkan di bawah ini dengan 4 nilai yaitu true positive (TP), False Positif (FP), true Negative (TN), false negative (FN).

Decision tree atau pohon keputusan adalah alat pendukung dengan struktur seperti pohon yang memodelkan kemungkinan hasil, biaya sumber daya, utilitas, dan kemungkinan konsekuensi. Disebut decision tree atau pohon keputusan karena pilihannya bercabang, membentuk struktur yang terlihat seperti pohon. Pohon keputusan menyediakan cara untuk menyajikan algoritma dengan pernyataan kontrol bersyarat. Mereka termasuk cabang yang mewakili langkah-langkah pengambilan keputusan yang dapat mengarah pada hasil yang menguntungkan.

Diagram alir struktur mencakup node internal yang mewakili tes atau atribut pada setiap tahap. Setiap cabang mewakili hasil untuk atribut, sedangkan jalur dari daun ke akar mewakili aturan untuk klasifikasi. Decision tree merupakan salah satu bentuk algoritma pembelajaran terbaik berdasarkan berbagai metode pembelajaran. Mereka meningkatkan model prediktif dengan akurasi, memudahkan dalam interpretasi, dan akurasi. Alat ini juga efektif dalam menyesuaikan hubungan non-linier karena mampu memecahkan tantangan penyesuaian data, seperti regresi dan klasifikasi. Pohon keputusan bekerja paling baik ketika Anda mengikuti aturan diagram alur dasar:

- Persegi panjang atau bujur sangkar: Gambar awal dari tempat Anda menulis pertanyaan.
- Garis: Mewakili cabang-cabang pohon. Ini semua adalah kemungkinan tindakan.
- Lingkaran: Menandakan hasil yang tidak pasti bahwa Anda akan membutuhkan cabang tambahan untuk diklarifikasi.
- Segitiga: Berikan jawaban yang jelas dan final. Mereka juga disebut “daun.”

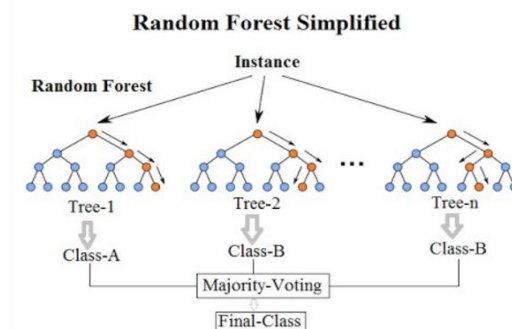
Dalam kasus klasifikasi ataupun prediksi suatu topic juga dilakukan oleh beberapa peneliti lain yang membawakannya dengan judul berbeda akan tetapi menggunakan algoritma yang sama yaitu algoritma random forest seperti penelitian yang dilakukan oleh Nur Fajri Azhar [1] dengan judul Memprediksi Waktu Memperbaiki Bug dari Laporan Bug Menggunakan Klasifikasi Random Forest. Dan penelitian selanjutnya juga dilakukan oleh Aji Prima Jaya [2] dengan judul Random Forest Algorithm for Prediction of Precipitation. Dan referensi terakhir yaitu penelitian yang dilakukan oleh Yoga Religia [3] dengan judul Analisis Perbandingan Algoritma Optimasi pada Random Forest untuk Klasifikasi Data Bank Marketing. Berdasarkan hasil kesimpulan dari ketiga peneliti diatas algoritma random forest cukup baik dan mampu menghasilkan error yang relatif rendah, performa yang baik dalam klasifikasi, dapat mengatasi data pelatihan dalam jumlah besar secara efisien, serta metode yang efektif untuk mengestimasi missing data.

## 2. Tinjauan Pustaka

### 2.1 Random Forest

Random forest adalah teknik atau metode suatu klasifikasi yang sering digunakan dan cukup populer di mata manusia. Random forest adalah struktur flowchart yang menyerupai pohon yang bercabang[4]. Random forest merupakan kombinasi dari masing - masing pohon yang baik kemudian dikombinasikan ke dalam satu model yang bergantung pada sebuah nilai vector random dengan distribusi yang sama pada semua pohon yang masing masing decision tree memiliki kedalaman yang maksimal. Untuk melakukan random forest yang menghasilkan variable importance, disarankan untuk menggunakan banyak pohon, misalnya seribu pohon atau lebih. Jika peubah penjelas yang dianalisis sangat banyak, nilai tersebut sebaiknya lebih besar agar variable importance yang dihasilkan semakin stabil.

Random Forest mengurangi varians dari sejumlah besar model "kompleks" dengan bias rendah. Kita bisa melihat elemen komposisi bukan model "lemah" tetapi model terlalu kompleks. Jika Anda membaca tentang algoritme, pohon di bawahnya ditanam "agak" sebesar "mungkin". Pohon yang mendasarinya adalah model paralel independen. Dan pemilihan variabel acak tambahan diperkenalkan ke dalamnya untuk membuatnya lebih mandiri, yang membuatnya berkinerja lebih baik daripada pengantongan biasa dan memberi nama "acak". Sementara meningkatkan mengurangi bias sejumlah besar model "kecil" dengan varian rendah. Mereka adalah model "lemah" seperti yang Anda kutip. Elemen-elemen yang mendasarinya entah bagaimana seperti model iteratif "rantai" atau "bersarang" tentang bias dari setiap level. Jadi mereka bukan model paralel independen tetapi masing-masing model dibangun berdasarkan semua model kecil sebelumnya dengan pembobotan. Itulah yang disebut "meningkatkan" dari satu per satu.



Gambar 1. Random Forest.

## 2.2 Confussion Matrix

Matriks Confusion adalah matriks  $N \times N$  yang digunakan untuk mengevaluasi kinerja model klasifikasi, di mana  $N$  adalah jumlah kelas target. Matriks tersebut membandingkan nilai target aktual dengan yang diprediksi oleh model pembelajaran mesin. Ini memberi kita pandangan holistik tentang seberapa baik kinerja model klasifikasi kita dan jenis kesalahan apa yang dibuatnya. Untuk masalah klasifikasi biner, kita akan memiliki matriks  $2 \times 2$  seperti yang ditunjukkan di bawah ini dengan 4 nilai[5]:

		ACTUAL VALUES	
		POSITIVE	NEGATIVE
PREDICTED VALUES	POSITIVE	TP	FP
	NEGATIVE	FN	TN

Gambar 2. Confusion matrix.

Keterangan:

- Variabel target memiliki dua nilai: Positif atau Negatif
- Kolom mewakili nilai aktual dari variabel target
- Baris mewakili nilai prediksi dari variabel target

Memahami yaitu true positive (TP), False Positif (FP), true negative (TN), false negative (FN) dalam Confusion matrix.

### 1. True Positive (TP)

- Nilai prediksi cocok dengan nilai sebenarnya.
- Nilai aktualnya positif dan model memprediksi nilai positif.

2. True Negative (TN)

- Nilai prediksi cocok dengan nilai sebenarnya.
- Nilai sebenarnya negatif dan model memprediksi nilai negative.

3. False Positive (FP) – Kesalahan tipe 1

- Nilai prediksi salah diprediksi.
- Nilai sebenarnya negatif tetapi model memprediksi nilai positif.
- Juga dikenal sebagai kesalahan Tipe 1.

4. False Negative (FN) – Kesalahan tipe 2

- Nilai prediksi salah diprediksi.
- Nilai sebenarnya positif tetapi model memprediksi nilai negative.
- Juga dikenal sebagai kesalahan Tipe 2.

Contoh perhitungan Dari confusion Matrix

Misalkan kita memiliki dataset klasifikasi dengan 1000 titik data. Kami memasang pengklasifikasi di atasnya dan mendapatkan Confusion matrix di bawah ini:

		ACTUAL VALUES	
		POSITIVE	NEGATIVE
PREDICTED VALUES	POSITIVE	560	60
	NEGATIVE	50	330

Gambar 3. Perhitungan Confusion Matrix.

Nilai yang berbeda dari matriks Confusion adalah sebagai berikut:

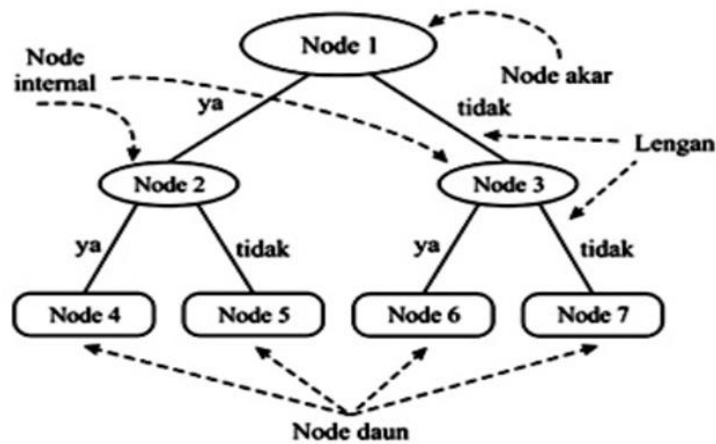
- True Positive (TP) = 560; artinya 560 titik data kelas positif diklasifikasikan dengan benar oleh model.

- True Negative (TN) = 330; artinya 330 titik data kelas negatif diklasifikasikan dengan benar oleh model.
- False Positive (FP) = 60; artinya 60 poin data kelas negatif salah diklasifikasikan sebagai milik kelas positif oleh model.
- False Negatif (FN) = 50; artinya 50 poin data kelas positif salah diklasifikasikan sebagai milik kelas negatif oleh model.

### 2.3 Pohon Keputusan (Decision Tree)

Decision tree atau pohon keputusan adalah alat pendukung dengan struktur seperti pohon yang memodelkan kemungkinan hasil, biaya sumber daya, utilitas, dan kemungkinan konsekuensi. Disebut decision tree atau pohon keputusan karena pilihannya bercabang, membentuk struktur yang terlihat seperti pohon[6]. Pohon keputusan menyediakan cara untuk menyajikan algoritma dengan pernyataan kontrol bersyarat. Mereka termasuk cabang yang mewakili langkah-langkah pengambilan keputusan yang dapat mengarah pada hasil yang menguntungkan. Diagram alir struktur mencakup node internal yang mewakili tes atau atribut pada setiap tahap. Setiap cabang mewakili hasil untuk atribut, sedangkan jalur dari daun ke akar mewakili aturan untuk klasifikasi. Decision tree merupakan salah satu bentuk algoritma pembelajaran terbaik berdasarkan berbagai metode pembelajaran. Mereka meningkatkan model prediktif dengan akurasi, memudahkan dalam interpretasi, dan akurasi. Alat ini juga efektif dalam menyesuaikan hubungan non-linier karena mampu memecahkan tantangan penyesuaian data, seperti regresi dan klasifikasi. Pohon keputusan bekerja paling baik ketika Anda mengikuti aturan diagram alur dasar:

- Persegi panjang atau bujur sangkar: Gambar awal dari tempat Anda menulis pertanyaan.
- Garis: Mewakili cabang-cabang pohon. Ini semua adalah kemungkinan tindakan.
- Lingkaran: Menandakan hasil yang tidak pasti bahwa Anda akan membutuhkan cabang tambahan untuk diklarifikasi.
- Segitiga: Berikan jawaban yang jelas dan final. Mereka juga disebut “daun.”



Gambar 4. Pohon Keputusan.

### 3. Metode Penelitian

Pada laporan ini, kami akan membuat sebuah prediksi tentang spesies burung berdasarkan data Spesies burung dari berbagai negara menggunakan metode Random Forest. Laporan riset ini berdasarkan coding yang kami kerjakan di google colab dan dapat diakses melalui link berikut:

<https://colab.research.google.com/drive/1E5Pk65tESeLBy3Vq7cQAYRSdqKhFWDv1?usp=s>  
 haring. Untuk dataset mentah yang kami gunakan, kami akses dari link berikut:  
<http://www.vision.caltech.edu/visipedia/CUB-200-2011.html> . File yang digunakan ada dalam folder Bernama 'CUB\_200\_2011'. Penjelasan mengenai hal tersebut ada di google colab.



Gambar 5. Spesies Burung.



Laporan prediksi Spesies Burung dengan menggunakan algoritma Random Forest menggunakan metode penelitian berjumlah 4 tahapan. Tahapan tersebut adalah: (1) pengumpulan data; (2) tahap pengolahan data; (3) tahap implementasi random forest; dan (4) tahap Hasil analisis. Metode penelitian yang dilakukan pada penelitian ini dapat dilihat pada Gambar 6.



Gambar 6. Flochart Metode Penelitian.

### 3.1 Pengumpulan Data

Data dari penelitian ini diambil di <http://www.vision.caltech.edu/visipedia/CUB-200-2011.html>. Data tersebut berasal sebuah buku yang berjudul Python Artificial Intelligence Projects for Beginners. Dimana isi dari buku tersebut memnuat 5 bab dan pada laporan ini membahas topic pada buku di bab 2.

### 3.2. Pengolahan Data

Pada tahap pengolahan data dilakukan pemilihan file dari sebuah dataset yang sudah disediakan dan yang sudah di download sebelumnya dalam buku, dikarenakan dalam folder yang sudah kita download tersebut banyak sekali filenya jadi perlu dilakukan pemilihan file yang diperlukan yang kemudian setelah mengetahui file apa saja yang dibutuhkan untuk melakukan prediksi spesies burung ,setelah itu dapat mengupload file tersebut ke google colab dan langsung mengeksekusi source yang sudah disediakan juga .

### 3.3 Implementasi Random Forest

Random Forest (RF) adalah algoritma yang menggunakan metode pemisahan biner rekursif untuk mencapai node akhir dalam struktur pohon berdasarkan pada pohon klasifikasi dan regresi. RF menghasilkan banyak pohon independen dengan subset yang dipilih secara acak melalui bootstrap dari sampel pelatihan dan dari variabel input disetiap node.

Random Forest melakukan klasifikasi dengan cara mengadopsi pendekatan ansambel dari berbagai pohon melalui kemunculan mayoritas untuk mencapai keputusan akhir . Set data pelatihan pada algoritma RF diformulasikan sebagai  $S = \{(x_i, y_j), i = 1, 2, \dots, N; j = 1, 2, \dots, M\}$ , dimana  $x$  adalah sampel dan  $y$  adalah variabel fitur  $S$ .  $N$  adalah jumlah sampel pelatihan, dan ada variabel fitur  $M$  di setiap sampel.

### 3.4 Hasil Analisa

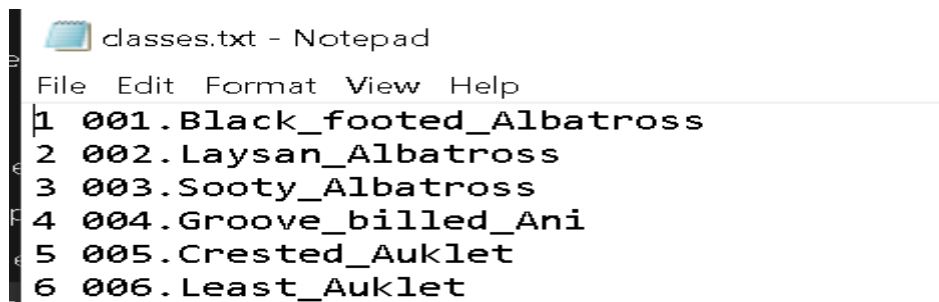
Hasil Analisa ini nantinya akan digunakan untuk memberikan suatu kesimpulan pada hasil akhir setelah uji coba melakukan prediksi spesies burung.

## 4. Hasil dan Pembahasan

Sebelum memulai penjelasan coding dan pembahasannya, pembaca diminta untuk mendownload dataset Jadi untuk memprediksi spesies burung dengan random forest ini kita menggunakan dataset yang berisi sekitar 12.000 foto burung dari 200 spesies berbeda. Dan berikut beberapa contoh gambar spesies dalam kumpulan data yang akan digunakan dalam project ini pada link berikut: <http://www.vision.caltech.edu/visipedia/CUB-200-2011.html>. File yang akan digunakan ada dalam folder bernama “[CUB-200-2011](http://www.vision.caltech.edu/visipedia/CUB-200-2011.html)”. Tujuan dari project ini adalah kami memprediksi spesies burung dalam jumlah yang banyak dengan menggunakan random forest dan juga confusion matrix. Akan tetapi tidak semua yang ada dalam folder yang sudah disebutkan diatas akan dipakai ,berikut beberapa file name yang akan digunakan dalam prediksi spesies burung:

### A. Prediksi Spesies Burung Dengan Random Forest Menggunakan Python.

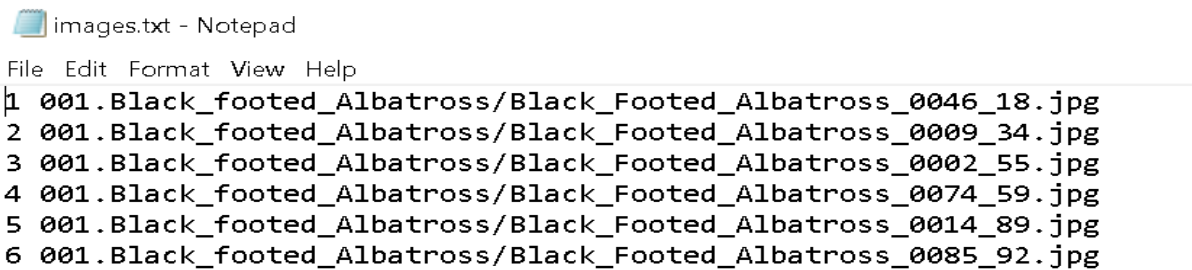
1. File classes.txt menunjukkan ID kelas dengan nama spesies burung.



```
classes.txt - Notepad
File Edit Format View Help
1 001.Black_footed_Albatross
2 002.Laysan_Albatross
3 003.Sooty_Albatross
4 004.Groove_billed_Ani
5 005.Crested_Auklet
6 006.Least_Auklet
```

Gambar 7. Classes.txt.

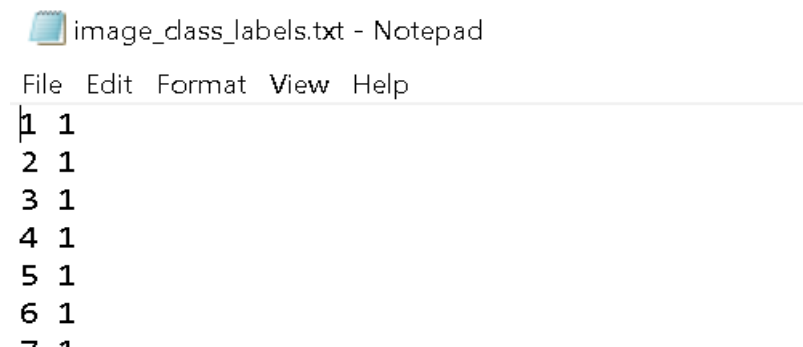
2. File image.txt menunjukkan ID gambar dan nama file.



```
images.txt - Notepad
File Edit Format View Help
1 001.Black_footed_Albatross/Black_Footed_Albatross_0046_18.jpg
2 001.Black_footed_Albatross/Black_Footed_Albatross_0009_34.jpg
3 001.Black_footed_Albatross/Black_Footed_Albatross_0002_55.jpg
4 001.Black_footed_Albatross/Black_Footed_Albatross_0074_59.jpg
5 001.Black_footed_Albatross/Black_Footed_Albatross_0014_89.jpg
6 001.Black_footed_Albatross/Black_Footed_Albatross_0085_92.jpg
```

Gambar 8. Image.txt.

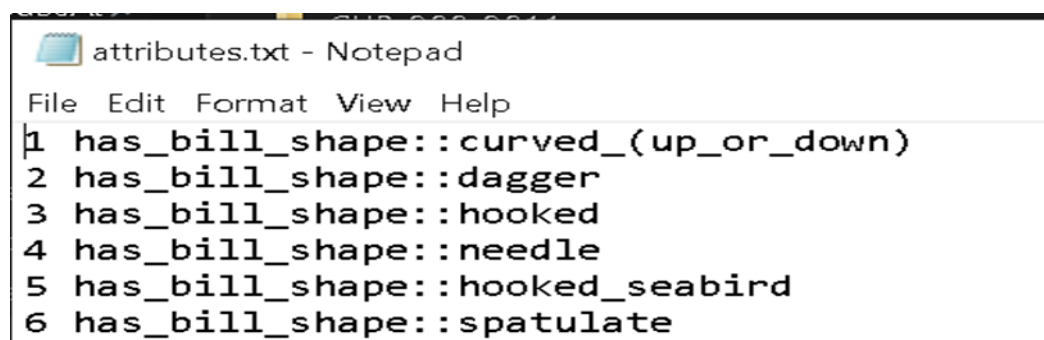
3. File image class\_labels.txt , yang menghubungkan ID kelas dengan ID gambar.



```
image_class_labels.txt - Notepad
File Edit Format View Help
1 1
2 1
3 1
4 1
5 1
6 1
7 1
```

Gambar 9. Image class label.txt.

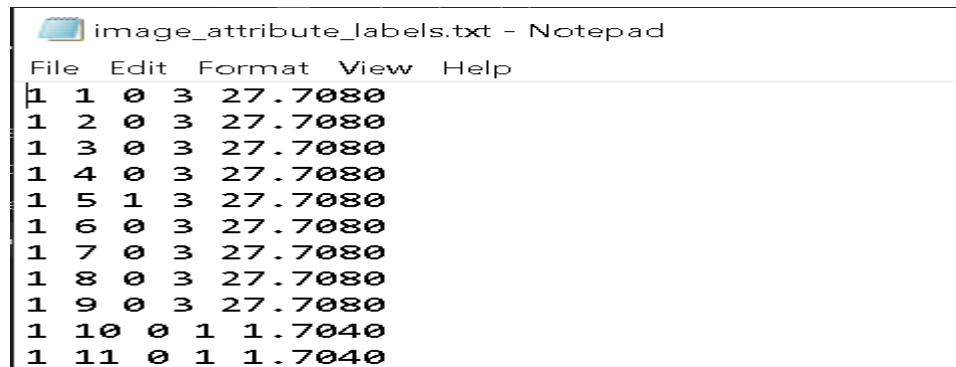
4. File attributes.txt memberikan nama setiap atribut



```
attributes.txt - Notepad
File Edit Format View Help
1 has_bill_shape::curved_(up_or_down)
2 has_bill_shape::dagger
3 has_bill_shape::hooked
4 has_bill_shape::needle
5 has_bill_shape::hooked_seabird
6 has_bill_shape::spatulate
```

Gambar 10. Attributes.txt.

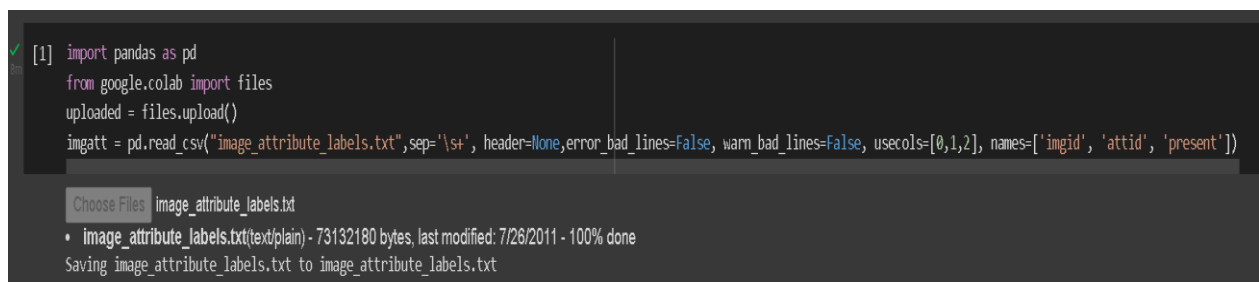
## 5. file image\_attribute\_labels.txt



1	1	0	3	27.7080
1	2	0	3	27.7080
1	3	0	3	27.7080
1	4	0	3	27.7080
1	5	1	3	27.7080
1	6	0	3	27.7080
1	7	0	3	27.7080
1	8	0	3	27.7080
1	9	0	3	27.7080
1	10	0	1	1.7040
1	11	0	1	1.7040

Gambar 11. image attributes labels.txt.

Setelah mengetahui file yang dibutuhkan dalam prediksi spesies burung dengan random forest selanjutnya kita akan mulai mengeksekusi code dengan menggunakan python. Pertama yang perlu dilakukan adalah:



```
[1] import pandas as pd
from google.colab import files
uploaded = files.upload()
imgatt = pd.read_csv("image_attribute_labels.txt", sep='\t', header=None, error_bad_lines=False, warn_bad_lines=False, usecols=[0,1,2], names=['imgid', 'attid', 'present'])
```

Choose Files image\_attribute\_labels.txt


- image\_attribute\_labels.txt(text/plain) • 73132180 bytes, last modified: 7/26/2011 • 100% done

Saving image\_attribute\_labels.txt to image\_attribute\_labels.txt

Jadi yang pertama kita perlu mengimport file csvnya yang sebelumnya sudah kita download, akan tetapi ada beberapa yang perlu diperhatikan:


1. Pemisahan ruang untuk semua nilai
2. Tidak ada kolom atau baris header
3. Abaikan pesan atau peringatan, `error_bad_lines=False` dan `warn_bad_lines=False`
4. Gunakan kolom 0, 1 dan 2 yang memiliki ID gambar, ID atribut, dan nilai sekarang atau tidak sekarang.

Berikut adalah tampilan hasil di bagian atas kumpulan data itu:

0s  `imgatt.head()`

	imgid	attid	present
0	1	1	0
1	1	2	0
2	1	3	0
3	1	4	0
4	1	5	1

ID gambar nomor 1 tidak memiliki atribut 1, 2, 3, atau 4, tetapi memiliki atribut 5. Untuk mengetahui berapa banyak baris dan kolom yang kita miliki dapat menggunakan code dibawah ini:

0s  `imgatt.shape`

`(3677856, 3)`

Ini memiliki 3,7 juta baris dan tiga kolom. Ini bukan formula sebenarnya yang kita butuhkan akan tetapi yang diperlukan adalah atribut menjadi kolom, bukan baris. Oleh karena itu, kita harus menggunakan pivot, seperti Excel memiliki metode pivot:

2s  `imgatt2 = imgatt.pivot(index='imgid', columns='attid', values='present')`

1. Putar ID gambar dan buat satu baris untuk setiap ID gambar. Hanya akan ada satu baris untuk gambar nomor satu.
2. Ubah atribut menjadi kolom yang berbeda, dan nilainya akan menjadi satu atau dua.

Sekarang kita dapat melihat bahwa setiap ID gambar hanya satu baris dan setiap atribut adalah kolomnya sendiri:

imgatt2.head()

attid	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	...	273	274	275
imgid	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	
2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	...	0	0	0
3	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0	...	0	0	0
4	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	...	0	0	0
5	0	0	0	0	1	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	...	0	0	0	

5 rows x 312 columns

Setelah itu kita masukkan data ini ke dalam random forest. Pada contoh sebelumnya, kita memiliki 312 kolom dan 312 atribut, yang pada akhirnya adalah sekitar 12.000 gambar atau 12.000 contoh berbeda dari burung-burung:

```
imgatt2.shape
```

```
(11788, 312)
```

Sekarang, kita perlu memuat jawaban, seperti apakah itu burung dan termasuk dalam spesies apa. Sejak ini file image class label, pemisahannya adalah spasi. Tidak ada baris header dan keduanya kolom adalah imgid dan label. Kita akan menggunakan `set_index('imgid')` yang memiliki hasil yang sama dengan yang dihasilkan oleh `imgatt2.head()`, di mana baris diidentifikasi oleh ID gambar:

```
from google.colab import files
uploaded = files.upload()
imglabels = pd.read_csv("image_class_labels.txt", sep=' ', header=None, names=['imgid', 'label'])

imglabels = imglabels.set_index('imgid')
```

Choose Files image\_class\_labels.txt

- image\_class\_labels.txt(text/plain) - 100487 bytes, last modified: 7/26/2011 - 100% done


Saving image\_class\_labels.txt to image\_class\_labels.txt

Dan hasilnya seperti Berikut:

```
imglabels.head()
```

imgid	label
1	1
2	1
3	1
4	1
5	1

Kolom imgid memiliki 1, 2, 3, 4, dan 5, Semua diberi label 1

✓ 0s  `imglabels.shape`  
`(11788, 1)`

Seperti yang terlihat, ada sekitar 12.000 baris, yang sempurna.

Ini adalah nomor yang sama dengan data atribut. Kita akan menggunakan join. Dalam join, kita akan menggunakan indeks pada image ID untuk menggabungkan dua data frame. Secara efektif, apa yang akan kita dapatkan adalah bahwa labelnya menempel pada kolom terakhir. Sekarang kita akan mengacak dan kemudian memisahkan atribut. Dengan kata lain, kita ingin lepaskan label dari label. Jadi, inilah atributnya, dengan 312 kolom pertama dan kolom terakhir menjadi label

```
[10] df = imgatt2.join(imglabels)
df = df.sample(frac=1)
```

```
[11] df_att = df.iloc[:, :312]
df_label = df.iloc[:, 312:]
```

```
df_att.head()
```

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	...	273	274	275	
imgid																																													
3058	0	1	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	0	...	0	0	0		
9422	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1	0	0	0	0	0	1	0	0	0	...	0	0	1			
10994	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0	...	0	0	0				
8665	0	0	0	0	0	0	1	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	...	0	0	0				
2245	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	1	0	...	0	0	1			

5 rows x 312 columns

Setelah mengacak, kita memiliki baris pertama sebagai gambar 6385, baris kedua sebagai gambar 198, dan seterusnya maju. Atribut dalam data label sesuai. Di baris pertama, itu gambar 6385, yang merupakan nomor 10. Anda tidak akan tahu burung mana itu, tetapi jenisnya, dan ini adalah atributnya. Tapi akhirnya dalam bentuk yang tepat. Kita perlu melakukan split tes pelatihan. Ada 12.000 baris, jadi mari kita ambil 8.000 pertama dan sebut mereka pelatihan, dan sisanya panggilan dari mereka menguji (4.000). Kami akan mendapatkan jawaban menggunakan RandomForestClassifier.

```
[13] df_train_att = df_att[:8000]
     df_train_label = df_label[:8000]
     df_test_att = df_att[8000:]
     df_test_label = df_label[8000:]

     df_train_label = df_train_label['label']
     df_test_label = df_test_label['label']

from sklearn.ensemble import RandomForestClassifier
clf = RandomForestClassifier(max_features=50, random_state=0, n_estimators=100)
```

Misalnya, jika kita mengatakan sesuatu seperti, lihat dua atribut, itu mungkin tidak cukup untuk benar-benar mencari tahu burung mana itu. Beberapa burung unik, jadi Anda mungkin membutuhkan lebih banyak lagi atribut. Kemudian jika kita mengatakan `max_features = 50` dan jumlah estimator menunjukkan jumlah pohon yang dibuat. Kecocokan benar-benar membangunkannya.

```
clf.fit(df_train_att, df_train_label)

RandomForestClassifier(max_features=50, random_state=0)
```

Mari kita prediksi beberapa kasus. Mari kita gunakan atribut dari lima baris pertama dari set pelatihan, yang akan memprediksi spesies 53, 161, 187, 148, dan 40. Setelah pengujian, kami mendapatkan akurasi 44%:

```
[16] print(clf.predict(df_train_att.head()))

[ 53 161 187 148 40]

clf.score(df_test_att, df_test_label)

0.4461457233368532
```

## B. Membuat Confusion Matrix Untuk Data

Jai selanjutnya yaitu membuat matyrix konfusi untuk melihat burung mana yang dibingungkan oleh kumpulan data. Itu Fungsi `confusion_matrix` dari scikit-learn akan menghasilkan matriks, tetapi ukurannya cukup besar

matriks:



```
[18]: from sklearn.metrics import confusion_matrix
      pred_labels = clf.predict(df_test_att)
      cm = confusion_matrix(df_test_label, pred_labels)

cm
array([[ 4,  2,  2, ...,  0,  0,  0],
       [ 0, 13,  0, ...,  0,  0,  0],
       [ 1,  1, 10, ...,  0,  0,  0],
       ...,
       [ 0,  0,  0, ...,  3,  0,  0],
       [ 0,  0,  0, ...,  0, 10,  0],
       [ 0,  0,  0, ...,  0,  0, 16]])
```

Dua ratus kali dua ratus tidak mudah dipahami dalam bentuk angka seperti ini. Berikut beberapa kode dari dokumentasi scikit-learn yang memungkinkan kita untuk memplot matriks dan warna dalam matriks:

```
import matplotlib.pyplot as plt
import itertools
def plot_confusion_matrix(cm, classes, normalize=False, title='Confusion matrix', cmap=plt.cm.Blues):

    if normalize:
        cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
        print("Normalisasi confusion matrix")
    else:
        print("confusion matrix, tanpa normalisasi")

    print(cm)

    plt.imshow(cm, interpolation='nearest', cmap=cmap)
    plt.title(title)
    #plt.colorbar()
    tick_marks = np.arange(len(classes))
    plt.xticks(tick_marks, classes, rotation=99)
    plt.yticks(tick_marks, classes)

    fmt = '.2f' if normalize else 'd'
    thresh = cm.max() / 2.

    plt.tight_layout()
    plt.ylabel('label benar')
    plt.xlabel('label prediksi')
```

Kita akan membutuhkan nama sebenarnya dari burung pada matriks sehingga kita mengetahui spesies yang sedang bingung satu sama lain. Jadi, mari kita muat file kelas:

```
from google.colab import files
uploaded = files.upload()
birds = pd.read_csv("classes.txt", sep='\s+', header=None, usecols=[1], names=['birdname'])
birds = birds['birdname']
birds
```

Choose Files classes.txt

- classes.txt(text/plain) - 4824 bytes, last modified: 7/26/2011 - 100% done

Saving classes.txt to classes.txt

```
0      001.Black_footed_Albatross
1      002.Laysan_Albatross
2      003.Sooty_Albatross
3      004.Groove_billed_Ani
4      005.Crested_Auklet
...
195     196.House_Wren
196     197.Marsh_Wren
197     198.Rock_Wren
198     199.Winter_Wren
199     200.Common_Yellowthroat
Name: birdname, Length: 200, dtype: object
```

Gambarkan matriksnya. Ini adalah matriks kebingungan untuk dataset ini:

```
import numpy as np
np.set_printoptions(precision=2)
plt.figure(figsize=(60,60), dpi=300)
plot_confusion_matrix(cm, classes=birds, normalize=True)
plt.show()
```


Normalisasi confusion matrix  
<Figure size 18000x18000 with 0 Axes>

Karena nama burung diurutkan, kuadrat kebingungannya lebih kecil. Mari kita bandingkan ini dengan pohon keputusan sederhana:

```
from sklearn import tree
clftree = tree.DecisionTreeClassifier()
clftree.fit(df_train_att, df_train_label)
clftree.score(df_test_att, df_test_label)
```

0.26583949313621963

Di sini, akurasi adalah 27%, lebih rendah dari akurasi 44% sebelumnya. Oleh karena itu, pohon keputusan lebih buruk. Jika kita menggunakan Support Vector Machine (SVM), yang merupakan neural pendekatan jaringan, outputnya adalah 29%:

```
✓ 43s  from sklearn import svm
clfsvm = svm.SVC()
clfsvm.fit(df_train_att, df_train_label)
clfsvm.score(df_test_att, df_test_label)

0.46858500527983105
```

Hutan acak masih lebih baik.


Mari kita lakukan validasi silang untuk memastikan bahwa kita membagi tes pelatihan dengan cara yang berbeda. Outputnya masih 45% untuk hutan acak, 26% untuk pohon keputusan kami, dan 47% untuk SVM, seperti yang ditunjukkan pada tangkapan layar berikut:

```
✓ 40s [25] from sklearn.model_selection import cross_val_score
scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
print("akurat: %0.2f (+/- %0.2f)" % (scores.mean(), scores.std() * 2))

akurat: 0.45 (+/- 0.03)

✓ 2s [26] scorestree = cross_val_score(clftree, df_train_att, df_train_label, cv=5)
print("akurat: %0.2f (+/- %0.2f)" % (scorestree.mean(), scorestree.std() * 2))

akurat: 0.26 (+/- 0.03)

✓ 1m  scoressvm = cross_val_score(clfsvm, df_train_att, df_train_label, cv=5)
print("akurat: %0.2f (+/- %0.2f)" % (scoressvm.mean(), scoressvm.std() * 2))

akurat: 0.47 (+/- 0.02)
```

Hasil terbaik tercermin melalui hutan acak karena kami memiliki beberapa opsi dan pertanyaan dengan hutan acak. Misalnya, berapa banyak pertanyaan berbeda yang dapat diajukan setiap pohon? Berapa banyak atributnya? lihat, dan ada berapa pohon? Nah, ada banyak parameter yang harus dilihat, jadi mari kita buat lingkaran dan coba semuanya:

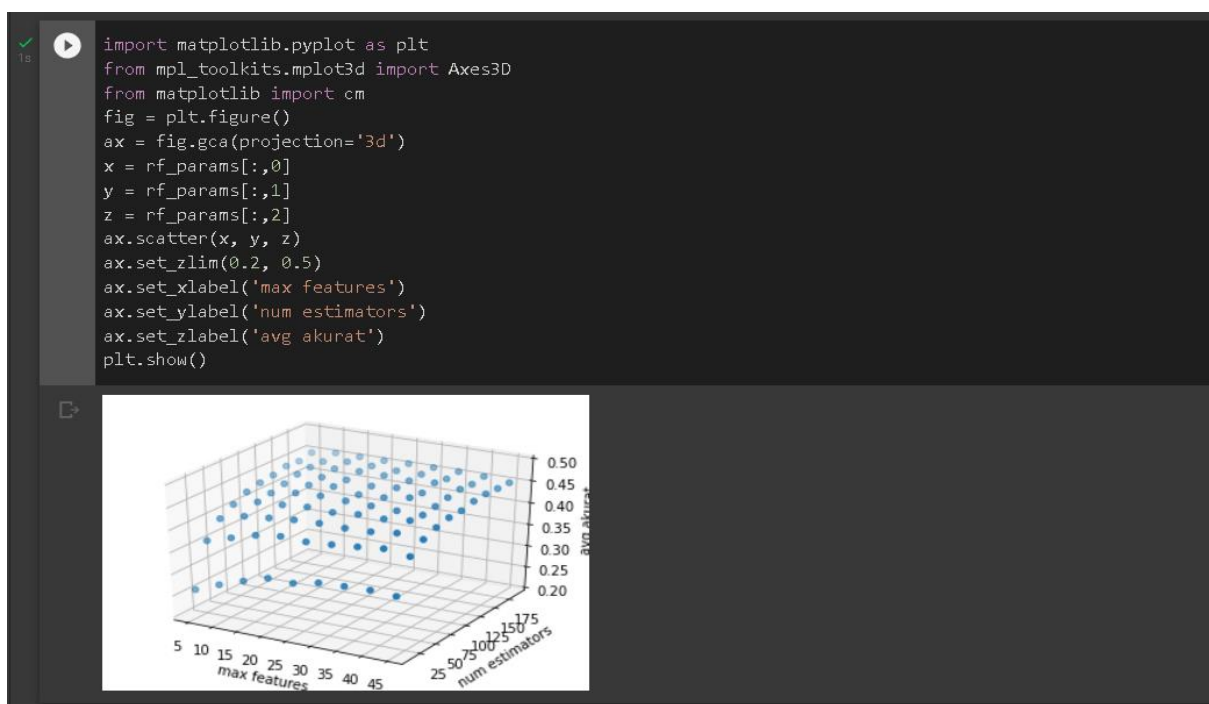
```

rf_params = np.empty((len(max_features_opts)*len(n_estimators_opts),4), float)
i = 0
for max_features in max_features_opts:
    for n_estimators in n_estimators_opts:
        clf = RandomForestClassifier(max_features=max_features, n_estimators=n_estimators)
        scores = cross_val_score(clf, df_train_att, df_train_label, cv=5)
        rf_params[i,0] = max_features
        rf_params[i,1] = n_estimators
        rf_params[i,2] = scores.mean()
        rf_params[i,3] = scores.std() * 2
        i += 1
    print("max feature: %d, num estimators: %d, akurat: %0.2f (+/- %0.2f)" % (max_features, n_estimators, scores.mean(), scores.std() * 2 ))

max feature: 5, num estimators: 10, akurat: 0.26 (+/- 0.01)
max feature: 5, num estimators: 30, akurat: 0.36 (+/- 0.02)
max feature: 5, num estimators: 50, akurat: 0.39 (+/- 0.01)
max feature: 5, num estimators: 70, akurat: 0.41 (+/- 0.02)
max feature: 5, num estimators: 90, akurat: 0.43 (+/- 0.02)
max feature: 5, num estimators: 110, akurat: 0.43 (+/- 0.02)
max feature: 5, num estimators: 130, akurat: 0.44 (+/- 0.02)
max feature: 5, num estimators: 150, akurat: 0.44 (+/- 0.02)
max feature: 5, num estimators: 170, akurat: 0.44 (+/- 0.02)
max feature: 5, num estimators: 190, akurat: 0.45 (+/- 0.02)
max feature: 10, num estimators: 10, akurat: 0.28 (+/- 0.02)
max feature: 10, num estimators: 30, akurat: 0.37 (+/- 0.02)
max feature: 10, num estimators: 50, akurat: 0.41 (+/- 0.01)
max feature: 10, num estimators: 70, akurat: 0.43 (+/- 0.01)
max feature: 10, num estimators: 90, akurat: 0.43 (+/- 0.02)
max feature: 10, num estimators: 110, akurat: 0.44 (+/- 0.02)
max feature: 10, num estimators: 130, akurat: 0.45 (+/- 0.02)
max feature: 10, num estimators: 150, akurat: 0.45 (+/- 0.02)

```

Ini semua akurasi, tetapi akan lebih baik untuk memvisualisasikannya dalam grafik, seperti yang ditunjukkan di sini:



Kita dapat melihat bahwa peningkatan jumlah pohon menghasilkan hasil yang lebih baik. Juga, meningkat jumlah fitur menghasilkan hasil yang lebih baik jika Anda dapat melihat lebih banyak fitur, tetapi pada akhirnya, jika Anda memiliki sekitar 20 hingga 30 fitur dan Anda memiliki sekitar 75 hingga 100 pohon, itu sekitar sebegus Anda akan mendapatkan akurasi 45%.

## **5. Kesimpulan Dan Saran**

### **Kesimpulan:**

Berdasarkan hasil ujicoba prediksi spesies burung dengan Random Forest diatas dapat disimpulkan bahwa:

1. Hasil dari laporan ini menunjukkan bahwa penggunaan metode random forest dan juga confusion matrix cukup baik untuk memprediksi spesies burung dengan berdasarkan ciri – ciri dan karakteristik masing- masing burung.
2. Dengan menggunakan confusion matrix data spesies burung yang sebelumnya menjadi kebingungan pada teknik random forest dikarenakan terdapat beberapa spesies burung yang sulit dibedakan dapat terselesaikan dengan hasil akurasi hampir mendekati benar.
3. Mampu menghasilkan error yang relatif rendah, performa yang baik dalam klasifikasi, dapat mengatasi data pelatihan dalam jumlah besar secara efisien, serta metode yang efektif untuk memprediksi spesies burung.

### **Saran:**

Untuk Uji coba selanjutnya disarankan dapat menggunakan algoritma lain selain random forest misalnya menggunakan CNN (Convolutional Neural Network) karena algoritma ini dapat membaca citra gambar dan warna yang lebih baik dan jelas ,jika random forest dalam memprediksi spesies burung perlu menggunakan karakteristik yang detail agar mengetahui termasuk dalam jenis apakah burung tersebut , Algoritma ini juga digunakan terutama dengan data visual, seperti klasifikasi gambar.

## Daftar Pustaka

- [1] N. F. Azhar and S. Rochimah, “Memprediksi Waktu Memperbaiki Bug dari Laporan Bug Menggunakan Klasifikasi Random Forest,” *J. Sist. dan Inform.*, vol. 11, no. 1, pp. 156–164, 2016.
- [2] A. Primajaya and B. N. Sari, “Random Forest Algorithm for Prediction of Precipitation,” *Indones. J. Artif. Intell. Data Min.*, vol. 1, no. 1, p. 27, 2018, doi: 10.24014/ijaidm.v1i1.4903.
- [3] R. Sistem, “JURNAL RESTI Analisis Perbandingan Algoritma Optimasi pada Random Forest untuk,” vol. 1, no. 10, pp. 187–192, 2021.
- [4] A. Yanuar, “Random Forest – Universitas Gadjah Mada Menara Ilmu Machine Learning,” *Universitas Gajah Mada*. 2018, [Online]. Available: <https://machinelearning.mipa.ugm.ac.id/2018/07/28/random-forest/>.
- [5] Aniruddha Bhandari, “Confusion Matrix for Machine Learning,” *Analytics Vidhya*. 2020, [Online]. Available: <https://www.analyticsvidhya.com/blog/2020/04/confusion-matrix-machine-learning/>.
- [6] “Decision Tree\_ Pengertian, Cara Buat, Kelebihan dan Kekurangannya.” .