

RIWAYAT HIDUP



Penulis bernama Bima Satria, lahir di Waru pada tanggal 19 Agustus 1999, adalah anak keenam dari almarhum Bapak Yusrani dan almarhumah Ibu Masraya. Penulis berkebangsaan Indonesia, Suku Bugis dan Banjar, serta beragama Islam. Penulis memulai pendidikan formalnya di TK 10 November Balikpapan pada tahun 2005, kemudian melanjutkan pendidikan di MI Darussalam Balikpapan pada tahun 2006. Pendidikan dasar Bima dimulai di SDN 001 PPU (2006-2009), dilanjutkan di SDN 020 Balikpapan Utara (2009-2012), dan diakhiri di SD Muhammadiyah 1 PPU (2012-2014). Selanjutnya, Penulis melanjutkan studinya di SMP Muhammadiyah 2 PPU dari tahun 2014 hingga 2017. Setelah menyelesaikan pendidikan SMP, Penulis melanjutkan studinya di SMK Muhammadiyah 1 PPU dan lulus pada tahun 2020. Saat ini, Penulis tercatat sebagai mahasiswa pada Fakultas Sains dan Teknologi, jurusan Teknik Informatika, di Universitas Muhammadiyah Kalimantan Timur, dimulai pada tahun 2020. Selama masa studi, Penulis memiliki pengalaman magang di SMK Muhammadiyah 1 PPU selama 3 bulan yang dilaksanakan pada semester tujuh. Demikian riwayat hidup ini disampaikan dengan harapan dapat memberikan gambaran yang jelas tentang latar belakang pendidikan dan pengalaman Penulis. Penulis berkomitmen untuk terus belajar dan berkontribusi dalam bidang Teknik Informatika, serta berusaha untuk mencapai prestasi akademis yang lebih tinggi demi masa depan yang lebih baik. Penulis juga mengucapkan terima kasih kepada semua pihak yang telah memberikan dukungan dan bimbingan selama perjalanan studinya, baik dari keluarga, teman, dosen, maupun institusi pendidikan. Semoga segala upaya dan dedikasi Penulis dapat bermanfaat bagi diri sendiri, keluarga, masyarakat, dan bangsa Indonesia.

LAMPIRAN

Lampiran 1 Surat Permohonan Pengambilan Data



UMKT
Program Studi
Teknik Informatika
Fakultas Sains dan Teknologi

Telp. 0541-748511 Fax. 0541-766832

Website <http://informatika.umkt.ac.id>

email: informatika@umkt.ac.id



بِسْمِ اللَّهِ الرَّحْمَنِ الرَّحِيمِ

Nomor : 003-009/FST.1/D.3/C/2024

Lampiran : -

Perihal : **Permohonan Pengambilan Data**

Kepada Yth.
Kepala Dinas Kesehatan Kota Samarinda
di -

Tempat

Assalamu'alaikum Warrahmatullahi Wabarrakatuh

Puji Syukur kepada Allah Subhanahu wa ta'ala yang senantiasa melimpahkan Rahmat-Nya kepada kita sekalian. Aamiin.

Sehubungan untuk memenuhi Tugas Akhir/Skripsi Tahun Akademik 2023/2024, maka dengan ini kami bermaksud untuk melakukan pengambilan data di Dinas Kesehatan Kota Samarinda. Adapun data yang diminta yaitu data penyakit stunting di Kota Samarinda tahun 2023, dengan nama mahasiswa sebagai berikut:

No	Nama	NIM	Program Studi
1	Ari Ahmad Dhani	2011102441090	Teknik Informatika
2	Bima Satria	2011102441102	Teknik Informatika
3	Lidya Sari	2011102441121	Teknik Informatika
4	Mukminatul Munawaroh	2011102441064	Teknik Informatika
5	Siti Muawwanah	2011102441153	Teknik Informatika

Demikian surat permohonan ini dibuat. Atas perhatiannya dan kerjasamanya kami mengucapkan terima kasih.

Wassalamu'alaikum Warrahmatullahi Wabarrakatuh

Samarinda, 4 Ramadhan 1445 H

15 Maret 2024 M



Program Studi S1 Teknik Informatika

[Signature]
Syah, S.Kom., M.TI
IDN. 1118019203

Lampiran 2 Tampilan Dataset Stunting Kota Samarinda Tahun 2023

No	Nama	J	BB Lahir	TB Lahir	Prov	Kab/Kota	Kec	Puskesmas	Desa/Kel	Posyandu	RT	RW	Usia Saat Ukur	Tanggal Pengukuran	Berat	Tinggi	LILA	BB/U	Zs BB/U	TB/U	Zs TB/U	BB/TB	Zs BB/TB	Naik Berat Badan	PMT Diterima (kg)	Jml. Vit	KPSP	KIA
1	Dimas	L	3.5	49	Kaltim	Samarinda	Sungai Pin	Remaja	Temindu	Pulau Ind			0 Tahun - 11 Bulan - 11 Hari	2023-01-02	9.1	74.5	0	Normal	-0.39	Normal	-0.39	Gizi Baik	-0.39	0	-		-	-
2	SITI	P	3	48	Kaltim	Samarinda	Sungai Pin	Remaja	Temindu	Pulau Ind	34		4 Tahun - 0 Bulan - 16 Hari	2023-01-02	12	94	0	Kurang	-2.25	Pendek	-2.25	Gizi Baik	-1.46	0	0.84		-	-
3	M AL	L	2.8	49	Kaltim	Samarinda	Sungai Pin	Remaja	Temindu	Pulau Ind			0 Tahun - 4 Bulan - 7 Hari	2023-01-02	8.1	69	0	Normal	-0.53	Normal	-0.53	Gizi Baik	-0.14	0	-		-	-
15072	Aftzah	P	2.5	45	Kaltim	Samarinda	Samarinda	Mangkup	Tenun Sa	Balo Neg	12		0 Tahun - 0 Bulan - 0 Hari	2023-12-09	2.5	45	0	Kurang	-2.03	Pendek	-2.63	Gizi Baik	-0.35	-	-		-	-
15073	M Arsyia	P	3	49	Kaltim	Samarinda	Samarinda	Juanda	Air Hitam	Mekar Se	32		0 Tahun - 0 Bulan - 0 Hari	2023-12-19	3	49	0	Normal	-1.56	Normal	-1.3	Gizi Baik	-1.04	-	-		-	-
15074	Muhammad	L	2.9	49	Kaltim	Samarinda	Samarinda	Sidomulyo	Sungai D	Ramania			0 Tahun - 0 Bulan - 0 Hari	2023-12-29	2.9	49	0	Kurang	-2.97	Pendek	-2.48	Gizi Baik	-1.37	-	-		-	-

Lampiran 3 Laporan Bimbingan

KARTU KENDALI BIMBINGAN LAPORAN KARYA ILMIAH

Nama Mahasiswa : Bima Satria
 NIM : 2011102441102
 Nama Dosen Pembimbing : Taghfirul Azhima Yoga Siswa, S.Kom, M.Kom
 Judul Penelitian : Optimasi Random Forest Dengan GA Dan RFE Pada High Dimensional Data Stunting

No	Tanggal	Uraian Pembimbingan	Paraf Dosen
1	5 Februari 2024	Pertemuan pertama, membahas tahapan-tahapan penelitian skripsi.	
2	7 Februari 2024	Pertemuan kedua, Membahas cara mencari artikel/paper rujukan penelitian dengan baik dan benar.	
3	16 Februari 2024	Pertemuan ketiga, Review Survey Paper untuk mencari gap permasalahan klasifikasi pada data mining, machine learning dan mencari topik penelitian yang masih relevan untuk diteliti peneliti.	
4	17 Februari 2024	Pertemuan keempat, Review Technical Paper dan Road Maps penelitian, untuk mencari algoritma dan metode yang di inginkan biar bisa mengatasi permasalahan kalsifikasi.	
5	22 Februari 2024	Pertemuan kelima, Mencari Paper/Artikel sesuai dengan rujukkan terkait objek penelitian dan pembahasan study literatur dan riset problem.	
6	6 Maret 2024	Pertemuan keenam, penentuan judul penelitian.	
7	13 Maret 2024	Pertemuan Ketujuh, Pembuatan Canvas penelitian untuk di submit ke prodi.	
8	5 April 2024	Pertemuan Kedelapan, Pengajuan Surat permohonan data untuk penelitian ke Dinas Kesehatan Samarinda.	
9	18 April 2024	Pertemuan Kesembilan, Revisi Proposal Bab 1, Bab 2 dan perbaikan format penulisan skripsi.	

10	24 April 2024	Pertemuan Kesepuluh, Revisi Proposal Bab 1, Bab 2 dan perbaikan format penulisan skripsi sebelum di submit ke simple.	
11	10 Mai 2024	Pertemuan kesebelas, bimbingan pembahasan penulisan Bab 3 dan Bab 4 beserta publikasi	
12	17 Mai 2024	Pertemuan kedua belas, Revisi naskah skripsi bab 3 sampai dengan bab 4	
13	22 Juni 2024	Pertemuan ketigabelas, konsultasi revisi publikasi jurnal bagian abstrak, pendahuluan, metodologi penelitian, hasil & pembahasan dan kesimpulan	

mengetahui

Dosen Pembimbing

Ketua Program Studi




Taghfirul Azhima Yoga Siswa, S.Kom, M.Kom
NIDN.1118038805

Arbansyah, S.Kom, M.TI
NIDN.1118019203

Lampiran 4 Kode Kode Klasifikasi Algoritma *Random Forest* + *RFE* + *GA*

```
import pandas as pd

# Membaca data dari file CSV (ganti dengan lokasi file yang sesuai)
data = pd.read_csv('dataset_stunting_samarinda.csv', encoding='latin1', low_memory=False)

# Menghitung jumlah data sebelum penghapusan duplikat
total_data_sebelum = len(data)

# Mengurutkan data berdasarkan kolom "Nama" dan "Tanggal Pengukuran" secara menurun
data = data.sort_values(by=["Nama", "Tanggal Pengukuran"], ascending=True, false)

# Menghapus duplikat berdasarkan kolom "Nama" dan mempertahankan yang pertama (yang memiliki tanggal pengukuran terbaru)
data = data.drop_duplicates(subset='Nama', keep='first')

# Menghitung jumlah data setelah penghapusan duplikat
total_data_sesudah = len(data)

# Menyimpan data yang telah dihapus duplikatnya ke file CSV baru (jika diperlukan)
data.to_csv('dataset_new.csv', index=False)

print(f'Total data sebelum penghapusan duplikat: {total_data_sebelum}')
print(f'Total data setelah penghapusan duplikat: {total_data_sesudah}')

# Total data sebelum penghapusan duplikat: 19474
# Total data setelah penghapusan duplikat: 3420

[ ] dataset = pd.read_csv('dataset_new.csv')
dataset.info()
dataset = data.drop('D1 Vit A', axis=1)

# Menyimpan hasilnya ke file CSV baru (jika diperlukan)
dataset.to_csv('dataset_tanpa_duplikat.csv', index=False)

# <class 'pandas.core.frame.DataFrame'>
# RangeIndex: 3420 entries, 0 to 3420
# Data columns (total 14 columns):
# Column Non-Null Count Dtype
# ---
# 0 Nama 3420 non-null object
# 1 JK 3420 non-null object
# 2 Berat 3420 non-null object
# 3 Tinggi 3363 non-null object
# 4 LILA 2898 non-null float64
# 5 BB/U 3420 non-null object
# 6 CS BB/U 3420 non-null object
# 7 TB/U 3154 non-null object
# 8 CS TB/U 3420 non-null object
# 9 BB/TB 3174 non-null object
# 10 CS BB/TB 3420 non-null object
# 11 Berat Badan 3420 non-null object
# 12 D1 Vit A 1260 non-null float64
# 13 Tanggal Pengukuran 3420 non-null object
# dtypes: float64(1), object(12)
# memory usage: 3.7 MB

[ ] dataset = pd.read_csv('dataset_tanpa_duplikat.csv')
dataset.info()

# <class 'pandas.core.frame.DataFrame'>
# RangeIndex: 3420 entries, 0 to 3420
# Data columns (total 13 columns):
# Column Non-Null Count Dtype
# ---
# 0 Nama 3420 non-null object
# 1 JK 3420 non-null object
# 2 Berat 3420 non-null object
# 3 Tinggi 3363 non-null object
# 4 LILA 2898 non-null float64
# 5 BB/U 3420 non-null object
# 6 CS BB/U 3420 non-null object
# 7 TB/U 3154 non-null object
# 8 CS TB/U 3420 non-null object
# 9 BB/TB 3174 non-null object
# 10 CS BB/TB 3420 non-null object
# 11 Berat Badan 3420 non-null object
# 12 Tanggal Pengukuran 3420 non-null object
# dtypes: float64(1), object(12)
# memory usage: 3.4 MB

dataset['TB/U'].value_counts()

TB/U
Normal      2843
Pendek      339
Sangat Pendek  115
Tinggi       143
Name: count, dtype: int64

import pandas as pd

# Membaca data dari file CSV atau sumber data lainnya
data = pd.read_csv('dataset_tanpa_duplikat.csv')

# Menghitung jumlah data sebelum penghapusan
jumlah_data_sebelum = len(data)

# Menghapus baris yang memiliki setidaknya satu nilai yang hilang
data_tanpa_nan = data.dropna()

# Menghitung jumlah data setelah penghapusan
jumlah_data_sesudah = len(data_tanpa_nan)

# Menyimpan hasilnya ke file CSV baru (jika diperlukan)
data_tanpa_nan.to_csv('dataset_tanpa_nan.csv', index=False)

# Mencetak jumlah data sebelum dan setelah penghapusan
print(f'Jumlah data sebelum penghapusan: {jumlah_data_sebelum}')
print(f'Jumlah data setelah penghapusan: {jumlah_data_sesudah}')

# Jumlah data sebelum penghapusan: 3420
# Jumlah data setelah penghapusan: 2691

[ ] data_tanpa_nan['TB/U'].value_counts()

TB/U
Normal      2287
Pendek      231
Sangat Pendek  99
Tinggi       124
Name: count, dtype: int64

import pandas as pd
import matplotlib.pyplot as plt

# Contoh data untuk TB/U
data_tanpa_nan = pd.DataFrame({
    'TB/U': ['Normal'] * 2287 + ['Pendek'] * 231 + ['Sangat Pendek'] * 99 + ['Tinggi'] * 124
})

# Hitung value counts
value_counts = data_tanpa_nan['TB/U'].value_counts()

# Buat DataFrame dari value counts
df_value_counts = value_counts.reset_index()
df_value_counts.columns = ['TB/U', 'count']

# Plot bar chart
plt.figure(figsize=(10, 6))
bars = plt.bar(df_value_counts['TB/U'], df_value_counts['count'], color=['blue', 'orange', 'red', 'green'])

# Tambahkan label di atas setiap bar
for bar in bars:
    yval = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2, yval + 5, int(yval), ha='center', va='bottom')

# Menambahkan judul dan label sumbu
plt.title('Data Setelah Penghapusan')
plt.xlabel('Kategori TB/U')
plt.ylabel('Jumlah')

# Tampilkan plot
plt.show()
```



```

import pandas as pd

# Membaca dataset
data_tanpa_isi = pd.read_csv('dataset_tanpa_nilai.csv')

# Membuat kamus penggantian nilai
penggantian = {
    'Normal': 'Tidak Stunting',
    'Tinggi': 'Tidak Stunting',
    'Pendek': 'Stunting',
    'Sangat Pendek': 'Stunting'
}

# Mengganti nilai dalam kolom 'TB/U' sesuai dengan kamus penggantian
if 'TB/U' in data_tanpa_isi.columns:
    data_tanpa_isi['TB/U'] = data_tanpa_isi['TB/U'].replace(penggantian)

# Menyimpan hasilnya ke file CSV baru
data_tanpa_isi.to_csv('dataset_stunting_samarinda_modif.csv', index=False)

# Menampilkan DataFrame hasil
print(data_tanpa_isi.head())

```

```

import pandas as pd

stunting = pd.read_csv('dataset_stunting_samarinda_modif.csv')
stunting = stunting.rename(columns={'TB/U': 'Kelas'})
stunting = stunting.drop('nama', axis=1)
stunting = stunting.drop('Tanggal Pengukuran', axis=1)
stunting

# Menyimpan hasilnya ke file CSV baru
stunting.to_csv('dataset_stunting_samarinda_terbaru.csv', index=False)
# class_data = stunting['TB/U']

stunting['Kelas'].value_counts()

```

```

import pandas as pd
import matplotlib.pyplot as plt
import os

# Contoh data untuk kelas
data = {
    'Kelas': ['Tidak Stunting'] * 22931 + ['Stunting'] * 3760
}
stunting = pd.DataFrame(data)

# Hitung value counts untuk kolom 'Kelas'
value_counts = stunting['Kelas'].value_counts()

# Buat DataFrame dari value counts
df_value_counts = value_counts.reset_index()
df_value_counts.columns = ['Kelas', 'count']

# Plot bar chart
plt.figure(figsize=(10, 6))
bars = plt.bar(df_value_counts['Kelas'], df_value_counts['count'], color=['blue', 'orange'])

# Tambahkan label di atas setiap bar
for bar in bars:
    yval = bar.get_height()
    plt.text(bar.get_x() + bar.get_width()/2, yval + 200, int(yval), ha='center', va='bottom')

# Menambahkan judul dan label sumbu
plt.title('Data Stunting dan Tidak Stunting')
plt.xlabel('Kelas')
plt.ylabel('Jumlah Data')

```



```
[ ] import pandas as pd
from sklearn.preprocessing import LabelEncoder, OrdinalEncoder

# Membaca dataset
stunting = pd.read_csv('dataset_stunting_samarinda_terbaru.csv')

# Menampilkan kolom yang ada dalam DataFrame
print('Kolom dalam DataFrame:', stunting.columns)

# Membuat instance untuk encoder
ordinal = OrdinalEncoder()
labelencoder = LabelEncoder()

# Daftar kolom yang dibutuhkan
columns_needed = ['JK', 'Berat', 'Tinggi', 'LILA', 'BB/U', 'ZS BB/U', 'ZS TB/U', 'Naik Berat Badan', 'BB/TB', 'ZS BB/TB', 'Kelas']

# Menyimpan subset data yang diperlukan
df_transform = stunting[columns_needed]

# Transformasi data kategorikal menjadi numerik
stunting['JK'] = labelencoder.fit_transform(stunting['JK'])
stunting['BB/U'] = labelencoder.fit_transform(stunting['BB/U'])
stunting['BB/TB'] = labelencoder.fit_transform(stunting['BB/TB'])
stunting['Naik Berat Badan'] = labelencoder.fit_transform(stunting['Naik Berat Badan'])

# Membuat DataFrame hasil transformasi
df = stunting[columns_needed]

# Menampilkan DataFrame hasil transformasi
print(df.head())
```

```
JK Kolom dalam DataFrame: Index(['JK', 'Berat', 'Tinggi', 'LILA', 'BB/U', 'ZS BB/U', 'ZS TB/U', 'Naik Berat Badan', 'BB/TB', 'ZS BB/TB', 'Kelas', 'ZS TB/U', 'ZS BB/TB', 'Naik Berat Badan'],
dtype='object')
JK Berat Tinggi LILA BB/U ZS BB/U ZS TB/U Naik Berat Badan BB/TB ZS BB/TB Kelas
0 0 18.6 104.5 18.0 2 1.19 0.41 1 5 1.41 Tidak Stunting
1 0 11.6 83.0 8.0 0 -0.08 -0.97 3 0 0.59 Tidak Stunting
2 0 9.7 78.0 16.0 0 -1.75 -2.84 2 0 -0.47 Stunting
3 0 15 107.0 17.0 0 -1.08 0.14 3 0 -1.83 Tidak Stunting
4 0 14 100.0 0.0 0 -0.85 -0.16 2 0 -1.14 Tidak Stunting
ZS BB/TB Kelas
0 1.41 Tidak Stunting
1 0.59 Tidak Stunting
2 -0.47 Stunting
3 -1.83 Tidak Stunting
4 -1.14 Tidak Stunting
```

```
[ ] df_transform
JK Berat Tinggi LILA BB/U ZS BB/U ZS TB/U Naik Berat Badan BB/TB ZS BB/TB Kelas
0 L 18.6 104.5 18.0 Risiko Lebih 1.19 0.41 N Risiko Gizi Lebih 1.41 Tidak Stunting
1 L 11.6 83.0 8.0 Berat Badan Normal -0.08 -0.97 T Gizi Baik 0.59 Tidak Stunting
2 L 9.7 78.0 16.0 Berat Badan Normal -1.75 -2.84 O Gizi Baik -0.47 Stunting
3 L 15 107.0 17.0 Berat Badan Normal -1.08 0.14 T Gizi Baik -1.83 Tidak Stunting
4 L 14 100.0 0.0 Berat Badan Normal -0.85 -0.16 O Gizi Baik -1.14 Tidak Stunting
26686 L 14.5 102.0 0.0 Berat Badan Normal -1.71 -1.62 T Gizi Baik -1.15 Tidak Stunting
26687 L 29.7 116.0 0.0 Risiko Lebih 4.16 2.18 N Obesitas 3.74 Tidak Stunting
26688 P 15.7 94.5 0.0 Risiko Lebih 1.78 1.38 O Risiko Gizi Lebih 1.43 Tidak Stunting
26689 L 17.3 107.0 0.0 Berat Badan Normal -0.14 -0.13 O Gizi Baik -0.13 Tidak Stunting
26690 L 15.6 102.0 0.0 Berat Badan Normal -0.59 -0.69 N Gizi Baik -0.25 Tidak Stunting
26691 rows x 11 columns
```

```
[ ] df
JK Berat Tinggi LILA BB/U ZS BB/U ZS TB/U Naik Berat Badan BB/TB ZS BB/TB Kelas
0 0 18.6 104.5 18.0 2 1.19 0.41 1 5 1.41 Tidak Stunting
1 0 11.6 83.0 8.0 0 -0.08 -0.97 3 0 0.59 Tidak Stunting
2 0 9.7 78.0 16.0 0 -1.75 -2.84 2 0 -0.47 Stunting
3 0 15 107.0 17.0 0 -1.08 0.14 3 0 -1.83 Tidak Stunting
4 0 14 100.0 0.0 0 -0.85 -0.16 2 0 -1.14 Tidak Stunting
26686 0 14.5 102.0 0.0 0 -1.71 -1.62 3 0 -1.15 Tidak Stunting
26687 0 29.7 116.0 0.0 2 4.16 2.18 1 4 3.74 Tidak Stunting
26688 1 15.7 94.5 0.0 2 1.78 1.38 2 5 1.43 Tidak Stunting
26689 0 17.3 107.0 0.0 0 -0.14 -0.13 2 0 -0.13 Tidak Stunting
26690 0 15.6 102.0 0.0 0 -0.59 -0.69 1 0 -0.25 Tidak Stunting
26691 rows x 11 columns
```

RANDOM FOREST

```
# Mengimport pustaka yang diperlukan
import pandas as pd
import numpy as np
from sklearn.model_selection import cross_val_predict, KFold, train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.preprocessing import MinMaxScaler
import matplotlib.pyplot as plt
import seaborn as sns

# Fungsi untuk membuat plot matrix kebingungan
def mat_matriks_kebingungan(cf, nama_grup=None, kategori='auto', jumlah=True, persen=True, barwarna=True, yticks=True, yjlotlabel=True, stat_inpgasan=True, ukuran_gambar=None, cmap='blue', judul=None, akurasi_pasti=None, rata_akurasi=None):
    kosong = ['' for i in range(cf.size)]
    if nama_grup and len(nama_grup) == cf.size:
        label_grup = ['(v1)'+(v2)'+(v3)'+(v4)'+(v5)'+(v6)'+(v7)'+(v8)'+(v9)'+(v10)'+(v11)'+(v12)'+(v13)'+(v14)'+(v15)'+(v16)'+(v17)'+(v18)'+(v19)'+(v20)'+(v21)'+(v22)'+(v23)'+(v24)'+(v25)'+(v26)'+(v27)'+(v28)'+(v29)'+(v30)'+(v31)'+(v32)'+(v33)'+(v34)'+(v35)'+(v36)'+(v37)'+(v38)'+(v39)'+(v40)'+(v41)'+(v42)'+(v43)'+(v44)'+(v45)'+(v46)'+(v47)'+(v48)'+(v49)'+(v50)'+(v51)'+(v52)'+(v53)'+(v54)'+(v55)'+(v56)'+(v57)'+(v58)'+(v59)'+(v60)'+(v61)'+(v62)'+(v63)'+(v64)'+(v65)'+(v66)'+(v67)'+(v68)'+(v69)'+(v70)'+(v71)'+(v72)'+(v73)'+(v74)'+(v75)'+(v76)'+(v77)'+(v78)'+(v79)'+(v80)'+(v81)'+(v82)'+(v83)'+(v84)'+(v85)'+(v86)'+(v87)'+(v88)'+(v89)'+(v90)'+(v91)'+(v92)'+(v93)'+(v94)'+(v95)'+(v96)'+(v97)'+(v98)'+(v99)'+(v100)'+(v101)'+(v102)'+(v103)'+(v104)'+(v105)'+(v106)'+(v107)'+(v108)'+(v109)'+(v110)'+(v111)'+(v112)'+(v113)'+(v114)'+(v115)'+(v116)'+(v117)'+(v118)'+(v119)'+(v120)'+(v121)'+(v122)'+(v123)'+(v124)'+(v125)'+(v126)'+(v127)'+(v128)'+(v129)'+(v130)'+(v131)'+(v132)'+(v133)'+(v134)'+(v135)'+(v136)'+(v137)'+(v138)'+(v139)'+(v140)'+(v141)'+(v142)'+(v143)'+(v144)'+(v145)'+(v146)'+(v147)'+(v148)'+(v149)'+(v150)'+(v151)'+(v152)'+(v153)'+(v154)'+(v155)'+(v156)'+(v157)'+(v158)'+(v159)'+(v160)'+(v161)'+(v162)'+(v163)'+(v164)'+(v165)'+(v166)'+(v167)'+(v168)'+(v169)'+(v170)'+(v171)'+(v172)'+(v173)'+(v174)'+(v175)'+(v176)'+(v177)'+(v178)'+(v179)'+(v180)'+(v181)'+(v182)'+(v183)'+(v184)'+(v185)'+(v186)'+(v187)'+(v188)'+(v189)'+(v190)'+(v191)'+(v192)'+(v193)'+(v194)'+(v195)'+(v196)'+(v197)'+(v198)'+(v199)'+(v200)'+(v201)'+(v202)'+(v203)'+(v204)'+(v205)'+(v206)'+(v207)'+(v208)'+(v209)'+(v210)'+(v211)'+(v212)'+(v213)'+(v214)'+(v215)'+(v216)'+(v217)'+(v218)'+(v219)'+(v220)'+(v221)'+(v222)'+(v223)'+(v224)'+(v225)'+(v226)'+(v227)'+(v228)'+(v229)'+(v230)'+(v231)'+(v232)'+(v233)'+(v234)'+(v235)'+(v236)'+(v237)'+(v238)'+(v239)'+(v240)'+(v241)'+(v242)'+(v243)'+(v244)'+(v245)'+(v246)'+(v247)'+(v248)'+(v249)'+(v250)'+(v251)'+(v252)'+(v253)'+(v254)'+(v255)'+(v256)'+(v257)'+(v258)'+(v259)'+(v260)'+(v261)'+(v262)'+(v263)'+(v264)'+(v265)'+(v266)'+(v267)'+(v268)'+(v269)'+(v270)'+(v271)'+(v272)'+(v273)'+(v274)'+(v275)'+(v276)'+(v277)'+(v278)'+(v279)'+(v280)'+(v281)'+(v282)'+(v283)'+(v284)'+(v285)'+(v286)'+(v287)'+(v288)'+(v289)'+(v290)'+(v291)'+(v292)'+(v293)'+(v294)'+(v295)'+(v296)'+(v297)'+(v298)'+(v299)'+(v300)'+(v301)'+(v302)'+(v303)'+(v304)'+(v305)'+(v306)'+(v307)'+(v308)'+(v309)'+(v310)'+(v311)'+(v312)'+(v313)'+(v314)'+(v315)'+(v316)'+(v317)'+(v318)'+(v319)'+(v320)'+(v321)'+(v322)'+(v323)'+(v324)'+(v325)'+(v326)'+(v327)'+(v328)'+(v329)'+(v330)'+(v331)'+(v332)'+(v333)'+(v334)'+(v335)'+(v336)'+(v337)'+(v338)'+(v339)'+(v340)'+(v341)'+(v342)'+(v343)'+(v344)'+(v345)'+(v346)'+(v347)'+(v348)'+(v349)'+(v350)'+(v351)'+(v352)'+(v353)'+(v354)'+(v355)'+(v356)'+(v357)'+(v358)'+(v359)'+(v360)'+(v361)'+(v362)'+(v363)'+(v364)'+(v365)'+(v366)'+(v367)'+(v368)'+(v369)'+(v370)'+(v371)'+(v372)'+(v373)'+(v374)'+(v375)'+(v376)'+(v377)'+(v378)'+(v379)'+(v380)'+(v381)'+(v382)'+(v383)'+(v384)'+(v385)'+(v386)'+(v387)'+(v388)'+(v389)'+(v390)'+(v391)'+(v392)'+(v393)'+(v394)'+(v395)'+(v396)'+(v397)'+(v398)'+(v399)'+(v400)'+(v401)'+(v402)'+(v403)'+(v404)'+(v405)'+(v406)'+(v407)'+(v408)'+(v409)'+(v410)'+(v411)'+(v412)'+(v413)'+(v414)'+(v415)'+(v416)'+(v417)'+(v418)'+(v419)'+(v420)'+(v421)'+(v422)'+(v423)'+(v424)'+(v425)'+(v426)'+(v427)'+(v428)'+(v429)'+(v430)'+(v431)'+(v432)'+(v433)'+(v434)'+(v435)'+(v436)'+(v437)'+(v438)'+(v439)'+(v440)'+(v441)'+(v442)'+(v443)'+(v444)'+(v445)'+(v446)'+(v447)'+(v448)'+(v449)'+(v450)'+(v451)'+(v452)'+(v453)'+(v454)'+(v455)'+(v456)'+(v457)'+(v458)'+(v459)'+(v460)'+(v461)'+(v462)'+(v463)'+(v464)'+(v465)'+(v466)'+(v467)'+(v468)'+(v469)'+(v470)'+(v471)'+(v472)'+(v473)'+(v474)'+(v475)'+(v476)'+(v477)'+(v478)'+(v479)'+(v480)'+(v481)'+(v482)'+(v483)'+(v484)'+(v485)'+(v486)'+(v487)'+(v488)'+(v489)'+(v490)'+(v491)'+(v492)'+(v493)'+(v494)'+(v495)'+(v496)'+(v497)'+(v498)'+(v499)'+(v500)'+(v501)'+(v502)'+(v503)'+(v504)'+(v505)'+(v506)'+(v507)'+(v508)'+(v509)'+(v510)'+(v511)'+(v512)'+(v513)'+(v514)'+(v515)'+(v516)'+(v517)'+(v518)'+(v519)'+(v520)'+(v521)'+(v522)'+(v523)'+(v524)'+(v525)'+(v526)'+(v527)'+(v528)'+(v529)'+(v530)'+(v531)'+(v532)'+(v533)'+(v534)'+(v535)'+(v536)'+(v537)'+(v538)'+(v539)'+(v540)'+(v541)'+(v542)'+(v543)'+(v544)'+(v545)'+(v546)'+(v547)'+(v548)'+(v549)'+(v550)'+(v551)'+(v552)'+(v553)'+(v554)'+(v555)'+(v556)'+(v557)'+(v558)'+(v559)'+(v560)'+(v561)'+(v562)'+(v563)'+(v564)'+(v565)'+(v566)'+(v567)'+(v568)'+(v569)'+(v570)'+(v571)'+(v572)'+(v573)'+(v574)'+(v575)'+(v576)'+(v577)'+(v578)'+(v579)'+(v580)'+(v581)'+(v582)'+(v583)'+(v584)'+(v585)'+(v586)'+(v587)'+(v588)'+(v589)'+(v590)'+(v591)'+(v592)'+(v593)'+(v594)'+(v595)'+(v596)'+(v597)'+(v598)'+(v599)'+(v600)'+(v601)'+(v602)'+(v603)'+(v604)'+(v605)'+(v606)'+(v607)'+(v608)'+(v609)'+(v610)'+(v611)'+(v612)'+(v613)'+(v614)'+(v615)'+(v616)'+(v617)'+(v618)'+(v619)'+(v620)'+(v621)'+(v622)'+(v623)'+(v624)'+(v625)'+(v626)'+(v627)'+(v628)'+(v629)'+(v630)'+(v631)'+(v632)'+(v633)'+(v634)'+(v635)'+(v636)'+(v637)'+(v638)'+(v639)'+(v640)'+(v641)'+(v642)'+(v643)'+(v644)'+(v645)'+(v646)'+(v647)'+(v648)'+(v649)'+(v650)'+(v651)'+(v652)'+(v653)'+(v654)'+(v655)'+(v656)'+(v657)'+(v658)'+(v659)'+(v660)'+(v661)'+(v662)'+(v663)'+(v664)'+(v665)'+(v666)'+(v667)'+(v668)'+(v669)'+(v670)'+(v671)'+(v672)'+(v673)'+(v674)'+(v675)'+(v676)'+(v677)'+(v678)'+(v679)'+(v680)'+(v681)'+(v682)'+(v683)'+(v684)'+(v685)'+(v686)'+(v687)'+(v688)'+(v689)'+(v690)'+(v691)'+(v692)'+(v693)'+(v694)'+(v695)'+(v696)'+(v697)'+(v698)'+(v699)'+(v700)'+(v701)'+(v702)'+(v703)'+(v704)'+(v705)'+(v706)'+(v707)'+(v708)'+(v709)'+(v710)'+(v711)'+(v712)'+(v713)'+(v714)'+(v715)'+(v716)'+(v717)'+(v718)'+(v719)'+(v720)'+(v721)'+(v722)'+(v723)'+(v724)'+(v725)'+(v726)'+(v727)'+(v728)'+(v729)'+(v730)'+(v731)'+(v732)'+(v733)'+(v734)'+(v735)'+(v736)'+(v737)'+(v738)'+(v739)'+(v740)'+(v741)'+(v742)'+(v743)'+(v744)'+(v745)'+(v746)'+(v747)'+(v748)'+(v749)'+(v750)'+(v751)'+(v752)'+(v753)'+(v754)'+(v755)'+(v756)'+(v757)'+(v758)'+(v759)'+(v760)'+(v761)'+(v762)'+(v763)'+(v764)'+(v765)'+(v766)'+(v767)'+(v768)'+(v769)'+(v770)'+(v771)'+(v772)'+(v773)'+(v774)'+(v775)'+(v776)'+(v777)'+(v778)'+(v779)'+(v780)'+(v781)'+(v782)'+(v783)'+(v784)'+(v785)'+(v786)'+(v787)'+(v788)'+(v789)'+(v790)'+(v791)'+(v792)'+(v793)'+(v794)'+(v795)'+(v796)'+(v797)'+(v798)'+(v799)'+(v800)'+(v801)'+(v802)'+(v803)'+(v804)'+(v805)'+(v806)'+(v807)'+(v808)'+(v809)'+(v810)'+(v811)'+(v812)'+(v813)'+(v814)'+(v815)'+(v816)'+(v817)'+(v818)'+(v819)'+(v820)'+(v821)'+(v822)'+(v823)'+(v824)'+(v825)'+(v826)'+(v827)'+(v828)'+(v829)'+(v830)'+(v831)'+(v832)'+(v833)'+(v834)'+(v835)'+(v836)'+(v837)'+(v838)'+(v839)'+(v840)'+(v841)'+(v842)'+(v843)'+(v844)'+(v845)'+(v846)'+(v847)'+(v848)'+(v849)'+(v850)'+(v851)'+(v852)'+(v853)'+(v854)'+(v855)'+(v856)'+(v857)'+(v858)'+(v859)'+(v860)'+(v861)'+(v862)'+(v863)'+(v864)'+(v865)'+(v866)'+(v867)'+(v868)'+(v869)'+(v870)'+(v871)'+(v872)'+(v873)'+(v874)'+(v875)'+(v876)'+(v877)'+(v878)'+(v879)'+(v880)'+(v881)'+(v882)'+(v883)'+(v884)'+(v885)'+(v886)'+(v887)'+(v888)'+(v889)'+(v890)'+(v891)'+(v892)'+(v893)'+(v894)'+(v895)'+(v896)'+(v897)'+(v898)'+(v899)'+(v900)'+(v901)'+(v902)'+(v903)'+(v904)'+(v905)'+(v906)'+(v907)'+(v908)'+(v909)'+(v910)'+(v911)'+(v912)'+(v913)'+(v914)'+(v915)'+(v916)'+(v917)'+(v918)'+(v919)'+(v920)'+(v921)'+(v922)'+(v923)'+(v924)'+(v925)'+(v926)'+(v927)'+(v928)'+(v929)'+(v930)'+(v931)'+(v932)'+(v933)'+(v934)'+(v935)'+(v936)'+(v937)'+(v938)'+(v939)'+(v940)'+(v941)'+(v942)'+(v943)'+(v944)'+(v945)'+(v946)'+(v947)'+(v948)'+(v949)'+(v950)'+(v951)'+(v952)'+(v953)'+(v954)'+(v955)'+(v956)'+(v957)'+(v958)'+(v959)'+(v960)'+(v961)'+(v962)'+(v963)'+(v964)'+(v965)'+(v966)'+(v967)'+(v968)'+(v969)'+(v970)'+(v971)'+(v972)'+(v973)'+(v974)'+(v975)'+(v976)'+(v977)'+(v978)'+(v979)'+(v980)'+(v981)'+(v982)'+(v983)'+(v984)'+(v985)'+(v986)'+(v987)'+(v988)'+(v989)'+(v990)'+(v991)'+(v992)'+(v993)'+(v994)'+(v995)'+(v996)'+(v997)'+(v998)'+(v999)'+(v1000)'+(v1001)'+(v1002)'+(v1003)'+(v1004)'+(v1005)'+(v1006)'+(v1007)'+(v1008)'+(v1009)'+(v1010)'+(v1011)'+(v1012)'+(v1013)'+(v1014)'+(v1015)'+(v1016)'+(v1017)'+(v1018)'+(v1019)'+(v1020)'+(v1021)'+(v1022)'+(v1023)'+(v1024)'+(v1025)'+(v1026)'+(v1027)'+(v1028)'+(v1029)'+(v1030)'+(v1031)'+(v1032)'+(v1033)'+(v1034)'+(v1035)'+(v1036)'+(v1037)'+(v1038)'+(v1039)'+(v1040)'+(v1041)'+(v1042)'+(v1043)'+(v1044)'+(v1045)'+(v1046)'+(v1047)'+(v1048)'+(v1049)'+(v1050)'+(v1051)'+(v1052)'+(v1053)'+(v1054)'+(v1055)'+(v1056)'+(v1057)'+(v1058)'+(v1059)'+(v1060)'+(v1061)'+(v1062)'+(v1063)'+(v1064)'+(v1065)'+(v1066)'+(v1067)'+(v1068)'+(v1069)'+(v1070)'+(v1071)'+(v1072)'+(v1073)'+(v1074)'+(v1075)'+(v1076)'+(v1077)'+(v1078)'+(v1079)'+(v1080)'+(v1081)'+(v1082)'+(v1083)'+(v1084)'+(v1085)'+(v1086)'+(v1087)'+(v1088)'+(v1089)'+(v1090)'+(v1091)'+(v1092)'+(v1093)'+(v1094)'+(v1095)'+(v1096)'+(v1097)'+(v1098)'+(v1099)'+(v1100)'+(v1101)'+(v1102)'+(v1103)'+(v1104)'+(v1105)'+(v1106)'+(v1107)'+(v1108)'+(v1109)'+(v1110)'+(v1111)'+(v1112)'+(v1113)'+(v1114)'+(v1115)'+(v1116)'+(v1117)'+(v1118)'+(v1119)'+(v1120)'+(v1121)'+(v1122)'+(v1123)'+(v1124)'+(v1125)'+(v1126)'+(v1127)'+(v1128)'+(v1129)'+(v1130)'+(v1131)'+(v1132)'+(v1133)'+(v1134)'+(v1135)'+(v1136)'+(v1137)'+(v1138)'+(v1139)'+(v1140)'+(v1141)'+(v1142)'+(v1143)'+(v1144)'+(v1145)'+(v1146)'+(v1147)'+(v1148)'+(v1149)'+(v1150)'+(v1151)'+(v1152)'+(v1153)'+(v1154)'+(v1155)'+(v1156)'+(v1157)'+(v1158)'+(v1159)'+(v1160)'+(v1161)'+(v1162)'+(v1163)'+(v1164)'+(v1165)'+(v1166)'+(v1167)'+(v1168)'+(v1169)'+(v1170)'+(v1171)'+(v1172)'+(v1173)'+(v1174)'+(v1175)'+(v1176)'+(v1177)'+(v1178)'+(v1179)'+(v1180)'+(v1181)'+(v1182)'+(v1183)'+(v1184)'+(v1185)'+(v1186)'+(v1187)'+(v1188)'+(v1189)'+(v1190)'+(v1191)'+(v1192)'+(v1193)'+(v1194)'+(v1195)'+(v1196)'+(v1197)'+(v1198)'+(v1199)'+(v1200)'+(v1201)'+(v1202)'+(v1203)'+(v1204)'+(v1205)'+(v1206)'+(v1207)'+(v1208)'+(v1209)'+(v1210)'+(v1211)'+(v1212)'+(v1213)'+(v1214)'+(v1215)'+(v1216)'+(v1217)'+(v1218)'+(v1219)'+(v1220)'+(v1221)'+(v1222)'+(v1223)'+(v1224)'+(v1225)'+(v1226)'+(v1227)'+(v1228)'+(v1229)'+(v1230)'+(v1231)'+(v1232)'+(v1233)'+(v1234)'+(v1235)'+(v1236)'+(v1237)'+(v1238)'+(v1239)'+(v1240)'+(v1241)'+(v1242)'+(v1243)'+(v1244)'+(v1245)'+(v1246)'+(v1247)'+(v1248)'+(v1249)'+(v1250)'+(v1251)'+(v1252)'+(v1253)'+(v1254)'+(v1255)'+(v1256)'+(v1257)'+(v1258)'+(v1259)'+(v1260)'+(v1261)'+(v1262)'+(v1263)'+(v1264)'+(v1265)'+(v1266)'+(v1267)'+(v1268)'+(v1269)'+(v1270)'+(v1271)'+(v1272)'+(v1273)'+(v1274)'+(v1275)'+(v1276)'+(v1277)'+(v1278)'+(v1279)'+(v1280)'+(v1281)'+(v1282)'+(v1283)'+(v1284)'+(v1285)'+(v1286)'+(v1287)'+(v1288)'+(v1289)'+(v1290)'+(v1291)'+(v1292)'+(v1293)'+(v1294)'+(v1295)'+(v1296)'+(v1297)'+(v1298)'+(v1299)'+(v1300)'+(v1301)'+(v1302)'+(v1303)'+(v1304)'+(v1305)'+(v1306)'+(v1307)'+(v1308)'+(v1309)'+(v1310)'+(v1311)'+(v1312)'+(v1313)'+(v1314)'+(v1315)'+(v1316)'+(v1317)'+(v1318)'+(v1319)'+(v1320)'+(v1321)'+(v1322)'+(v1323)'+(v1324)'+(v1325)'+(v1326)'+(v1327)'+(v1328)'+(v1329)'+(v1330)'+(v1331)'+(v1332)'+(v1333)'+(v1334)'+(v1335)'+(v1336)'+(v1337)'+(v1338)'+(v1339)'+(v1340)'+(v1341)'+(v1342)'+(v1343)'+(v1344)'+(v1345)'+(v1346)'+(v1347)'+(v1348)'+(v1349)'+(v1350)'+(v1351)'+(v1352)'+(v1353)'+(v1354)'+(v1355)'+(v1356)'+(v1357)'+(v1358)'+(v1359)'+(v1360)'+(v1361)'+(v1362)'+(v1363)'+(v1364)'+(v1365)'+(v1366)'+(v1367)'+(v1368)'+(v1369)'+(v1370)'+(v1371)'+(v1372)'+(v1373)'+(v1374)'+(v1375)'+(v1376)'+(v1377)'+(v1378)'+(v1379)'+(v1380)'+(v1381)'+(v1382)'+(v1383)'+(v1384)'+(v1385)'+(v1386)'+(v1387)'+(v1388)'+(v1389)'+(v1390)'+(v1391)'+(v1392)'+(v1393)'+(v1394)'+(v1395)'+(v1396)'+(v1397)'+(v1398)'+(v1399)'+(v1400)'+(v1401)'+(v1402)'+(v1403)'+(v1404)'+(v1405)'+(v1406)'+(v1407)'+(v1408)'+(v1409)'+(v1410)'+(v1411)'+(v1412)'+(v1413)'+(v1414)'+(v1415)'+(v1416)'+(v1417)'+(v1418)'+(v1419)'+(v1420)'+(v1421)'+(v1422)'+(v1423)'+(v1424)'+(v1425)'+(v1426)'+(v1427)'+(v1428)'+(v1429)'+(v1430)'+(v1431)'+(v1432)'+(v1433)'+(
```

```

else:
    plt.xlabel('tela_stet')
    plt.ylabel('jumlah')
    plt.title('jumlah')
    plt.show()

# Membaca data yang telah diproses
data = pd.read_csv('dataset_stunting_samarinda_modif.csv')

# Mengganti nama kolom 'TB(U)' menjadi 'kelas'
data = data.rename(columns={'TB(U)': 'kelas'})

# Menghapus kolom non-numerik dan tidak relevan
data = data.drop(['nama', 'tanggal Pengukuran'], axis=1)

# Mengganti nilai bernilai di kolom 'JK'
data['JK'] = data['JK'].str.strip().replace({'L': 'L', 'P': 'P'})

# Menangani nilai numerik bernilai
data['berat'] = data['berat'].str.replace(',','').astype(float)

# Mengonversi kolom kategorikal menjadi numerik menggunakan one-hot encoding
kolom_kategorikal = ['JK', 'TB(U)', 'BB(U)', 'jenis Berat badan']
data = pd.get_dummies(data, columns=kolom_kategorikal, drop_first=True)

# Memastikan tidak ada kehilangan data setelah konversi
print("Bentuk dataset setelah konversi: (data.shape)")

# Menyiapkan fitur dan variabel target
X = data.drop('kelas', axis=1)
y = data['kelas'].map({'Tidak Stunting': 0, 'Stunting': 1})

# Menormalisasi fitur
scaler = MinMaxScaler()
X_normalized = scaler.fit_transform(X)

# Membagi data menjadi training dan testing set
X_train, X_test, y_train, y_test = train_test_split(X_normalized, y, test_size=0.3, random_state=42)

# Melakukan k-fold 10 kali
kfold = 10
repeats = 10
akurasi_list = []

# Melakukan k-fold untuk cross-validation
kf = KFold(n_splits=kfold, shuffle=True, random_state=1)
fold = 1

```

```

# Melakukan cross-validation dengan variasi
for i in range(repeats):
    print("Rekaman (%i):" % i)

    # Mendefinisikan classifier Random Forest dengan parameter yang sedikit berbeda untuk setiap iterasi
    rf_classifier = RandomForestClassifier(
        n_estimators=10, # Mengubah jumlah estimators
        criterion='gini',
        max_depth=1, # Mengubah kedalaman maksimal pohon
        min_samples_split=10, # Mengubah jumlah minimal sampel untuk split
        min_samples_leaf=1, # Mengubah jumlah minimal sampel per daun
        max_features=0.1, # (1/5) * 0.02, # Mengubah jumlah fitur yang dipertimbangkan untuk split
        class_weight='balanced',
        random_state=i, # Mengubah random state
    )

    y_pred = cross_val_predict(rf_classifier, X_train, y_train, cv=kfold)
    cf_matrix = confusion_matrix(y_train, y_pred)
    akurasi = np.trace(cf_matrix) / float(np.sum(cf_matrix))
    akurasi_list.append(akurasi)
    print("Akurasi: (akurasi: %f)" % akurasi)
    class_report = classification_report(y_train, y_pred, target_names=('Tidak Stunting', 'Stunting'))
    print(class_report)

# Menghitung akurasi rata-rata
rata2_akurasi = np.mean(akurasi_list)
akurasi_terakhir = akurasi_list[-1]
print("Rata-rata Akurasi: (rata2_akurasi: %f)" % rata2_akurasi)
print("Akurasi Terakhir (Iterasi Terakhir): (akurasi_terakhir: %f)" % akurasi_terakhir)

# Menggambar plot matriks kebingungan untuk iterasi terakhir dengan rata-rata akurasi dan akurasi pasti
buat_matriks_kebingungan(cf_matrix, akurasi_gambar=(f, 4), barname=False, judul="Matriks Random Forest (k-fold = 10)", akurasi_pasti=akurasi_terakhir, rata2_akurasi=rata2_akurasi)

```

Bentuk dataset setelah konversi: (26691, 19)

Rekaman 1
Akurasi: 0.709

	precision	recall	f1-score	support
Tidak Stunting	0.94	0.70	0.81	18347
Stunting	0.29	0.74	0.42	3095
accuracy			0.71	21352
macro avg	0.62	0.72	0.61	21352
weighted avg	0.85	0.71	0.75	21352

Rekaman 2
Akurasi: 0.860

	precision	recall	f1-score	support
--	-----------	--------	----------	---------

Rekaman 3
Akurasi: 0.996

	precision	recall	f1-score	support
Tidak Stunting	0.93	0.98	0.92	18347
Stunting	0.50	0.60	0.55	3095
accuracy			0.86	21352
macro avg	0.72	0.75	0.73	21352
weighted avg	0.87	0.86	0.87	21352

Rekaman 4
Akurasi: 0.894

	precision	recall	f1-score	support
Tidak Stunting	1.00	1.00	1.00	18347
Stunting	0.98	1.00	0.99	3095
accuracy			1.00	21352
macro avg	0.99	1.00	0.99	21352
weighted avg	1.00	1.00	1.00	21352

Rekaman 5
Akurasi: 0.968

	precision	recall	f1-score	support
Tidak Stunting	0.97	0.98	0.94	18347
Stunting	0.59	0.82	0.69	3095
accuracy			0.89	21352
macro avg	0.78	0.88	0.82	21352
weighted avg	0.92	0.89	0.90	21352

Rekaman 6
Akurasi: 0.956

	precision	recall	f1-score	support
Tidak Stunting	1.00	0.96	0.98	18347
Stunting	0.82	0.99	0.90	3095
accuracy			0.97	21352
macro avg	0.91	0.98	0.94	21352
weighted avg	0.97	0.97	0.97	21352

Rekaman 7
Akurasi: 0.942

	precision	recall	f1-score	support
Tidak Stunting	0.98	0.93	0.95	18347
Stunting	0.65	0.86	0.74	3095
accuracy			0.92	21352
macro avg	0.82	0.89	0.85	21352
weighted avg	0.93	0.92	0.92	21352

Rekaman 8
Akurasi: 0.942

	precision	recall	f1-score	support
Tidak Stunting	1.00	0.93	0.94	18347
Stunting	0.71	0.99	0.83	3095
accuracy			0.94	21352
macro avg	0.85	0.96	0.90	21352
weighted avg	0.96	0.94	0.95	21352

```

Rekaman 8
Akurasi: 0.965
precision    recall  f1-score   support

Tidek Stunting   0.99    0.97    0.98    18347
Stunting         0.84    0.93    0.88    3085

accuracy        0.91    0.95    0.97    21352
macro avg      0.91    0.95    0.93    21352
weighted avg   0.97    0.97    0.97    21352

Rekaman 9
Akurasi: 0.992
precision    recall  f1-score   support

Tidek Stunting   1.00    0.99    1.00    18347
Stunting         0.95    0.99    0.97    3085

accuracy        0.98    0.99    0.99    21352
macro avg      0.98    0.99    0.98    21352
weighted avg   0.99    0.99    0.99    21352

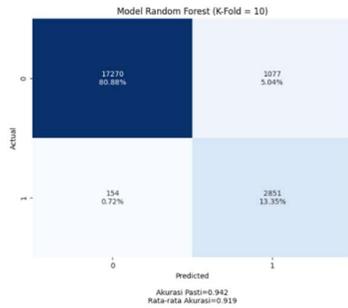
Rekaman 10
Akurasi: 0.942
precision    recall  f1-score   support

Tidek Stunting   0.99    0.94    0.97    18347
Stunting         0.73    0.95    0.82    3085

accuracy        0.86    0.95    0.89    21352
macro avg      0.86    0.95    0.89    21352
weighted avg   0.95    0.94    0.95    21352

Rate-rata Akurasi: 0.959
Akurasi Pasti (Iterasi Terakhir): 0.942

```



RANDOM FOREST + RFE

```

# Mengimport pustaka yang diperlukan
import pandas as pd
import numpy as np
from sklearn.model_selection import cross_val_predict, Kfold, train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.preprocessing import MinMaxScaler
from sklearn.feature_selection import RFE
import matplotlib.pyplot as plt
import seaborn as sns

# Fungsi untuk membuat plot matriks kebingungan
def buat_matriks_kebingungan(cf, nama_group=None, kategori='auto', jumlah=True, persentase=True, barwarna=True, xyticks=True, xyrotasi=True, stat_ringkasan=True, ukuran_gambar=None, cmap='Blues', judul=None, akurasi_pasti=None, rata_rata_akurasi=None):
    if nama_group and len(nama_group) == cf.size:
        label_group = []
        for value in nama_group:
            label_group.append(value)
    else:
        label_group = 'Kosong'
    if jumlah:
        jumlah_group = [(0,0,0)] * cf.size
    else:
        jumlah_group = 'Kosong'
    if persentase:
        persentase_group = [(0,0,0)] * cf.size
    else:
        persentase_group = 'Kosong'
    label_kotak = [(v1,v2,v3) for v1, v2, v3 in zip(label_group, jumlah_group, persentase_group)]
    label_kotak = np.asarray(label_kotak).reshape(cf.shape[0], cf.shape[1])
    teks_stat = []
    if stat_ringkasan:
        if akurasi_pasti is not None:
            teks_stat.append('Akurasi Pasti: (%.3f)' % akurasi_pasti)
        if rata_rata_akurasi is not None:
            teks_stat.append('Rate-rata Akurasi: (%.3f)' % rata_rata_akurasi)
    if ukuran_gambar is None:
        ukuran_gambar = plt.rcParams.get('figure.figsize')
    if not xyrotasi:
        kategori = False
    plt.figure(figsize=ukuran_gambar)
    cm = confusion_matrix(cf, cross_val_predict(rfc, data, y, cv=kfold, n_jobs=-1, random_state=42))
    if xyrotasi:
        plt.xlabel('Actual')
        plt.ylabel('Predicted')
    else:
        plt.xlabel('Actual')
        plt.ylabel('Predicted')
    if judul:
        plt.title(judul)
    if stat:
        plt.show()

```

```

plt.title('Judul')
plt.show()

# Membaca data yang telah diproses
data = pd.read_csv('dataset_stunting_samarinda_modif.csv')

# Mengganti nama kolom 'TB01' menjadi 'Kelas'
data = data.rename(columns={'TB01': 'Kelas'})

# Menghapus kolom non-numerik dan tidak relevan
data = data.drop(['Nama', 'Tempat Pengukuran'], axis=1)

# Mengganti nilai bernomor di kolom 'JK'
data['JK'] = data['JK'].str.strip().replace({'L': '1', 'P': '2'})

# Mengganti nilai numerik bernomor
data['Berat'] = data['Berat'].str.replace('.', '').astype(float)

# Mengonversi kolom kategorikal menjadi numerik menggunakan one-hot encoding
kolom_kategorikal = ['JK', 'TB01', 'BB78', 'Masa Berat Badan']
data = pd.get_dummies(data, columns=kolom_kategorikal, drop_first=True)

# Memastikan tidak ada nilai hilang data setelah konversi
print('Bentuk dataset setelah konversi: ', data.shape)

# Menisahkan fitur dan variabel target
X = data.drop('Kelas', axis=1)
y = data['Kelas'].map({'Tidek Stunting': 0, 'Stunting': 1})

# Menormalisasi fitur
scaler = MinMaxScaler()
X_normalized = scaler.fit_transform(X)

# Membagi data menjadi training dan testing set
x_train, x_test, y_train, y_test = train_test_split(X_normalized, y, test_size=0.2, random_state=42)

# Seleksi fitur menggunakan RFE
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rfe = RFE(estimator=rf, n_features_to_select=10)
x_train_rfe = rfe.fit_transform(x_train, y_train)
x_test_rfe = rfe.transform(x_test)

# Menekan hasil K-fold 10 kali
kfold = 10
repeats = 10
akurasi_list = []

# Menggunakan Kfold untuk cross-validation
kf = KFold(n_splits=kfold, shuffle=True, random_state=1)
fold = 1

```

```

# Melakukan cross-validation dengan variasi
for i in range(repeats):
    print("Rekaman (%i)" % i)

    # Menetukan classifier Random Forest dengan parameter yang sedikit berbeda untuk setiap iterasi
    rf_classifier = RandomForestClassifier(
        n_estimators=10, # Mengubah jumlah estimators
        criterion='gini',
        max_depth=1, # Mengubah kedalaman maksimal pohon
        n_jobs=-1, # Mengubah jumlah minimal sampel untuk split
        min_samples_leaf=1, # Mengubah jumlah minimal sampel per daun
        max_features=0.1, # Mengubah jumlah fitur yang dipertimbangkan untuk split
        class_weight='balanced',
        random_state=i # Mengubah random state
    )

    y_pred = cross_val_predict(rf_classifier, x_train_rf, y_train, cv=kfolds)
    cf_matrix = confusion_matrix(y_train, y_pred)
    akurasi = np.trace(cf_matrix) / float(np.sum(cf_matrix))
    akurasi_list.append(akurasi)
    print("Akurasi: (%i)" % akurasi)
    class_report = classification_report(y_train, y_pred, target_names=["Tidak Stunting", "Stunting"])
    print(class_report)

# Menghitung akurasi rata-rata
rata2_akurasi = np.mean(akurasi_list)
akurasi_terakhir = akurasi_list[-1]
print("Rata-rata akurasi (%i) (rata2_akurasi: %i)" % (rata2_akurasi, akurasi_terakhir))
print("Akurasi pasti (%i) (akurasi_terakhir: %i)" % (akurasi_terakhir, akurasi_terakhir))

# Menggambar plot matriks kebingungan untuk iterasi terakhir dengan rata-rata akurasi dan akurasi pasti
buat_matriks_kebingungan(cf_matrix, ukuran_sumbu=(5, 6), barwarna=False, judul="Model Random Forest - RFE (C-Fold = 10)", akurasi_pasti=akurasi_terakhir, rata2_akurasi=rata2_akurasi)

```

Bentuk dataset setelah konversi: (26691, 19)

Rekaman 1
Akurasi: 0.817

	precision	recall	f1-score	support
Tidak Stunting	0.94	0.84	0.89	18347
Stunting	0.41	0.66	0.50	3005
accuracy			0.82	21352
macro avg	0.67	0.75	0.69	21352
weighted avg	0.86	0.82	0.83	21352

Rekaman 2
Akurasi: 0.849

	precision	recall	f1-score	support
Tidak Stunting	0.97	0.85	0.91	18347
Stunting	0.48	0.83	0.61	3005
accuracy			0.85	21352
macro avg	0.72	0.84	0.76	21352
weighted avg	0.90	0.85	0.86	21352

Rekaman 3
Akurasi: 0.995

	precision	recall	f1-score	support
Tidak Stunting	1.00	0.99	1.00	18347
Stunting	0.97	1.00	0.98	3005
accuracy			0.99	21352
macro avg	0.98	1.00	0.99	21352
weighted avg	0.99	0.99	0.99	21352

Rekaman 4
Akurasi: 0.864

	precision	recall	f1-score	support
Tidak Stunting	0.92	0.92	0.92	18347
Stunting	0.51	0.54	0.53	3005
accuracy			0.86	21352
macro avg	0.72	0.73	0.72	21352
weighted avg	0.87	0.86	0.86	21352

Rekaman 5
Akurasi: 0.985

	precision	recall	f1-score	support
Tidak Stunting	0.99	0.99	0.99	18347
Stunting	0.93	0.97	0.95	3005
accuracy			0.98	21352
macro avg	0.96	0.98	0.97	21352
weighted avg	0.99	0.98	0.98	21352

Rekaman 6
Akurasi: 0.935

	precision	recall	f1-score	support
Tidak Stunting	0.98	0.94	0.96	18347
Stunting	0.71	0.91	0.80	3005
accuracy			0.93	21352
macro avg	0.85	0.92	0.88	21352
weighted avg	0.95	0.93	0.94	21352

Rekaman 7
Akurasi: 1.000

	precision	recall	f1-score	support
Tidak Stunting	1.00	1.00	1.00	18347
Stunting	1.00	1.00	1.00	3005
accuracy			1.00	21352
macro avg	1.00	1.00	1.00	21352
weighted avg	1.00	1.00	1.00	21352

Rekaman 8
Akurasi: 0.950

	precision	recall	f1-score	support
Tidak Stunting	1.00	0.94	0.97	18347
Stunting	0.74	0.99	0.85	3005
accuracy			0.95	21352
macro avg	0.87	0.97	0.91	21352
weighted avg	0.96	0.95	0.95	21352

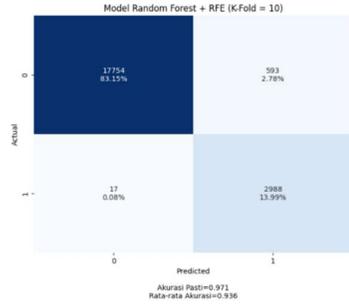
Rekaman 9
Akurasi: 0.996

	precision	recall	f1-score	support
Tidak Stunting	1.00	1.00	1.00	18347
Stunting	0.98	0.99	0.98	3005
accuracy			1.00	21352
macro avg	0.99	0.99	0.99	21352
weighted avg	1.00	1.00	1.00	21352

Rekaman 10
Akurasi: 0.971

	precision	recall	f1-score	support
Tidak Stunting	1.00	0.97	0.98	18347
Stunting	0.83	0.99	0.91	3005
accuracy			0.97	21352
macro avg	0.92	0.98	0.95	21352
weighted avg	0.98	0.97	0.97	21352

Rata-rata Akurasi: 0.936
Akurasi Pasti (Starsi Terastir): 0.971



```
[ ] # Instal DEAP
pip install deap

Collecting deap
  Downloading deap-1.4.1-cp310-cp310-manylinux_2_3_x86_64_manylinux_2_28_x86_64_manylinux2014_x86_64.whl (135 kb)
Requirement already satisfied: numpy in /usr/local/lib/python3.10/dist-packages (from deap) (1.25.2)
Installing collected packages: deap
Successfully installed deap-1.4.1
```

RF+RFE+GA

```
# Mengimport pustaka yang diperlukan
import pandas as pd
import numpy as np
from sklearn.model_selection import cross_val_predict, KFold, train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, classification_report, accuracy_score
from sklearn.preprocessing import MinMaxScaler
from sklearn.feature_selection import RFE
import matplotlib.pyplot as plt
import seaborn as sns
from deap import base, creator, tools, algorithms

# Fungsi untuk membuat plot matriks kelungupan
def buat_matriks_kelungupan(cf, nama_group=None, kategori='auto', jumlah=True, persen=True, barwarna=True, xticks=True, yplotlabel=True, stat_lingkasan=True, ukuran_gambar=None, cmap='Blues', judul=None, akurasi_pasti=None, rata_akurasi=None):
    kosong = [" " for i in range(cf.size)]
    if nama_group and len(nama_group) == cf.size:
        label_group = [{"\n".format(value) for value in nama_group}]
    else:
        label_group = kosong
    if jumlah:
        jumlah_group = [{"(0.0f)\n".format(value) for value in cf.flatten()}]
    else:
        jumlah_group = kosong
    if persen:
        persentase_group = [{"(0.20)\n".format(value) for value in cf.flatten()} / np.sum(cf)]
    else:
        persentase_group = kosong
    label_kotak = [{"(0.1)\n".format(i) for i, v1, v2, v3 in zip(label_group, jumlah_group, persentase_group)}]
    label_kotak = np.asarray(label_kotak).reshape(cf.shape[0], cf.shape[1])
    teks_stat = []
    if stat_lingkasan:
        if akurasi_pasti is not None:
            teks_stat += "\nAkurasi Pasti: (0.3f)".format(akurasi_pasti)
        if rata_akurasi is not None:
            teks_stat += "\nRata-rata Akurasi: (0.3f)".format(rata_akurasi)
        if ukuran_gambar is None:
            ukuran_gambar = plt.rcParams.get('figure.figsize')
    if not yplotlabel:
        kategori = False
    plt.figure(figsize=ukuran_gambar)
    sns.heatmap(cf, annot=label_kotak, fmt="", cmap=cmap, cbar=barwarna, xticklabels=kategori, yticklabels=kategori)
    if yplotlabel:
        plt.xlabel('Actual')
        plt.ylabel('Predicted' + teks_stat)
    else:
        plt.xlabel('Actual')
    if judul:
        plt.title(judul)
```

```
plt.title(judul)
plt.show()

# Mem baca data yang telah diproses
data = pd.read_csv('dataset_stunting_samarinda_modif.csv')
# Mengganti nama kolom 'B0' menjadi 'kelas'
data = data.rename(columns={'B0': 'kelas'})
# Menghapus kolom non-numerik dan tidak relevan
data = data.drop(['Name', 'Tanggal Pengukuran'], axis=1)
# Mengganti nilai bermasalah di kolom 'JK'
data['JK'] = data['JK'].str.strip().replace({'L': 'L', 'P': 'P'})
# Mengganti nilai numerik bermasalah
data['Berat'] = data['Berat'].str.replace(',', '.').astype(float)
# Mengonversi kolom kategorikal menjadi numerik menggunakan one-hot encoding
kolom_kategorikal = ['JK', 'B0', 'B7B', 'Tinggi Berat Badan']
data = pd.get_dummies(data, columns=kolom_kategorikal, drop_first=True)
# Memastikan tidak ada kehilangan data setelah konversi
print("Bentuk dataset setelah konversi: (data.shape)")
# Memisahkan fitur dan variabel target
X = data.drop('kelas', axis=1)
y = data['kelas'].map({'Stunting': 0, 'Tidak Stunting': 1})
# Menormalisasi fitur
scaler = MinMaxScaler()
X_normalized = scaler.fit_transform(X)
# Membagi data menjadi training dan testing set
X_train, X_test, y_train, y_test = train_test_split(X_normalized, y, test_size=0.2, random_state=42)
# Memilih fitur menggunakan RFE
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rfe = RFE(estimator=rf, n_features_to_select=10)
X_train_rfe = rfe.fit_transform(X_train, y_train)
X_test_rfe = rfe.transform(X_test)
# Konfigurasi GA
creator.create("FitnessMax", base.Fitness, weights=(1.0,))
creator.create("Individual", list, fitness=creator.FitnessMax)
toolbox = base.Toolbox()
toolbox.register("attr_bool", np.random.randint, 0, 2)
toolbox.register("individual", tools.initRepeat, creator.Individual, toolbox.attr_bool, n=len(X.columns))
toolbox.register("population", tools.initRepeat, list, toolbox.individual)
```

```

# Fungsi evaluasi
def evalf(individual):
    selected_features = [i for i, bit in enumerate(individual) if bit == 1]
    if len(selected_features) == 0:
        return 0, # Hindari subset fitur kosong
    X_subset = X_normalised[:, selected_features]
    selector = RF(rf, X_subset, cv=cv5) # Mengurangi ukuran CV fold
    X_selected = selector.fit_transform(X_subset, y)
    y_pred = cross_val_predict(rf, X_selected, y, cv=5) # Mengurangi Jumlah CV fold
    return np.mean(cross_val_predict(rf, X_selected, y, cv=5) == y), # Mengembalikan akurasi

toolbox.register('mate', tools.cxMPoint)
toolbox.register('mutate', tools.mutFlipBit, indpb=0.05)
toolbox.register('select', tools.selTournament, tournsize=3)
toolbox.register('evaluate', evalf)

# Menjalankan GA dengan pengaturan yang lebih cepat
population = toolbox.population(n=30) # Mengurangi ukuran populasi
ngen = 10 # Mengurangi jumlah generasi
cprob = 0.5
mprob = 0.3

result = algorithms.eaSimple(population, toolbox, cxpb, mutpb, ngen, stats=None, halloffame=None, verbose=False)

# Menggunakan fitur terbaik yang ditemukan oleh GA
best_individual = tools.selBest(population, 1)[0]
selected_features = [i for i, bit in enumerate(best_individual) if bit == 1]
X_best = X_normalised[:, selected_features]

# Menekan hasil k-fold 10 kali
kfold = 10
repeats = 10
akurasi_list = []

# Menggunakan k-fold untuk cross-validation
kf = KFold(n_splits=kfold, shuffle=True, random_state=1)
fold = 1

```

```

# Melakukan cross-validation dengan variasi
for i in range(repeats):
    print("Rekaman (%d)" % i)

    # Mendefinisikan Classifier Random Forest dengan parameter yang sedikit berbeda untuk setiap iterasi
    rf_classifier = RandomForestClassifier(
        n_estimators=10, # Mengubah jumlah estimators
        criterion='gini',
        max_depth=1, # Mengubah kedalaman maksimal pohon
        min_samples_split=1, # Mengubah jumlah minimal sampel untuk split
        min_samples_leaf=1, # Mengubah jumlah minimal sampel per daun
        max_features=1, # Mengubah jumlah fitur yang dipertimbangkan untuk split
        class_weight='balanced',
        random_state=42 # Mengubah random state
    )

    y_pred = cross_val_predict(rf_classifier, X_best, y, cv=kfold)
    cf_matrix = confusion_matrix(y, y_pred)
    akurasi = np.trace(cf_matrix) / float(np.sum(cf_matrix))
    akurasi_list.append(akurasi)
    print("Akurasi: (%d)" % akurasi)
    class_report = classification_report(y, y_pred, target_names=["Tidak Stunting", "Stunting"])
    print(class_report)

# Menghitung akurasi rata-rata
rata2_akurasi = np.mean(akurasi_list)
akurasi_terakhir = akurasi_list[-1]
print("Rata-rata Akurasi: (%d)" % rata2_akurasi)
print("Akurasi Pasti (Terakhir Iterasi): (%d)" % akurasi_terakhir)

# Menggambar plot matriks kesingungan untuk iterasi terakhir dengan rata-rata akurasi dan akurasi pasti
buat_matriks_kesingungan(cf_matrix, ukuran_gambar=(8, 6), barname=False, judul="Model Random Forest - RF - GA (k-fold = 10)", akurasi_pasti=akurasi_terakhir, rata2_akurasi=rata2_akurasi)

```

```

Bentuk dataset setelah konversi: (26691, 19)
/usr/local/lib/python3.10/dist-packages/deep/creator.py:185: RuntimeWarning: A class named 'fitnessMax' has already been created and it will be overwritten. Consider deleting previous creation of that class or rename it.
warnings.warn("A class named '%s' has already been created and it "
/usr/local/lib/python3.10/dist-packages/deep/creator.py:185: RuntimeWarning: A class named 'Individual' has already been created and it will be overwritten. Consider deleting previous creation of that class or rename it.
warnings.warn("A class named '%s' has already been created and it "
Rekaman 1
Akurasi: 0.996
precision    recall  f1-score   support

Tidak Stunting    1.00    1.00    1.00   22931
Stunting          0.98    1.00    0.99   3760

accuracy         0.99    1.00    0.99   26691
macro avg       0.99    1.00    0.99   26691
weighted avg    1.00    1.00    1.00   26691

Rekaman 2
Akurasi: 0.997
precision    recall  f1-score   support

Tidak Stunting    1.00    1.00    1.00   22931
Stunting          1.00    0.98    0.99   3760

accuracy         1.00    0.99    0.99   26691
macro avg       1.00    0.99    0.99   26691
weighted avg    1.00    1.00    1.00   26691

Rekaman 3
Akurasi: 0.893
precision    recall  f1-score   support

Tidak Stunting    1.00    0.88    0.93   22931
Stunting          0.57    0.98    0.72   3760

accuracy         0.89    0.89    0.89   26691
macro avg       0.78    0.93    0.85   26691
weighted avg    0.94    0.89    0.90   26691

Rekaman 4
Akurasi: 0.955
precision    recall  f1-score   support

Tidak Stunting    0.99    0.96    0.97   22931
Stunting          0.78    0.94    0.85   3760

accuracy         0.89    0.95    0.91   26691
macro avg       0.89    0.95    0.91   26691
weighted avg    0.96    0.95    0.96   26691

```

Rekaman 5				
Akurasi: 0.997				
	precision	recall	f1-score	support
Tidak Stunting	1.00	1.00	1.00	22931
Stunting	1.00	0.99	0.99	3760
accuracy			1.00	26691
macro avg	1.00	0.99	0.99	26691
weighted avg	1.00	1.00	1.00	26691

Rekaman 6				
Akurasi: 0.998				
	precision	recall	f1-score	support
Tidak Stunting	1.00	1.00	1.00	22931
Stunting	0.99	0.99	0.99	3760
accuracy			1.00	26691
macro avg	0.99	1.00	1.00	26691
weighted avg	1.00	1.00	1.00	26691

Rekaman 7				
Akurasi: 0.995				
	precision	recall	f1-score	support
Tidak Stunting	1.00	1.00	1.00	22931
Stunting	0.99	0.99	0.99	3760
accuracy			1.00	26691
macro avg	0.99	0.99	0.99	26691
weighted avg	1.00	1.00	1.00	26691

Rekaman 8				
Akurasi: 1.000				
	precision	recall	f1-score	support
Tidak Stunting	1.00	1.00	1.00	22931
Stunting	1.00	1.00	1.00	3760
accuracy			1.00	26691
macro avg	1.00	1.00	1.00	26691
weighted avg	1.00	1.00	1.00	26691

Rekaman 9				
Akurasi: 0.999				
	precision	recall	f1-score	support
Tidak Stunting	1.00	1.00	1.00	22931
Stunting	1.00	0.99	0.99	3760
accuracy			1.00	26691
macro avg	1.00	1.00	1.00	26691
weighted avg	1.00	1.00	1.00	26691



SELEKSI FITUR RFE

```

import pandas as pd
import numpy as np

from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing importMinMaxScaler
from sklearn.feature_selection import RFE

# Membaca data yang telah diunggah
data_path = 'dataset_stunting_tanarindu_terbaru.csv'
data = pd.read_csv(data_path)

# Mengganti nama kolom 'TB/U' menjadi 'Kelas'
data = data.rename(columns={'TB/U': 'Kelas'})

# Menangani nilai bermasalah di kolom 'JK'
data['JK'] = data['JK'].str.strip().replace({'L': 'L', 'P': 'P'})

# Menangani nilai numerik bermasalah
data['Berat'] = data['Berat'].str.replace(',', '.').astype(float)

# Mengonversi kolom 'JK' menjadi numerik menggunakan label encoding
data['JK'] = data['JK'].map({'L': 0, 'P': 1})

# Mengonversi kolom 'Hais Berat Badan' menjadi numerik menggunakan label encoding
data['Hais Berat Badan'] = data['Hais Berat Badan'].map({'0': 0, '1': 1, '2': 2})

# Mengonversi kolom 'BB/U' dan 'BB/TB' menjadi numerik menggunakan label encoding
data['BB/U'] = data['BB/U'].map({'Normal': 0, 'Kurang': 1, 'Risiko Lebih': 2, 'Sangat Kurang': 3})
data['BB/TB'] = data['BB/TB'].map({'Normal': 0, 'Gizi Buruk': 1, 'Gizi Kurang': 2, 'Gizi Lebih': 3, 'Obesitas': 4, 'Risiko Gizi Lebih': 5})

# Menghapus baris yang mengandung nilai NaN
data_cleaned = data.dropna()

# Menetapkan fitur dan variabel target
X_cleaned = data_cleaned.drop('Kelas', axis=1)
y_cleaned = data_cleaned['Kelas'].map({'Tidak Stunting': 0, 'Stunting': 1})

# Menormalisasi fitur
scaler =MinMaxScaler()
X_normalised_cleaned = scaler.fit_transform(X_cleaned)

# Seleksi fitur menggunakan RFE dengan jumlah fitur yang lebih banyak (10 fitur terbaik)
rfe = RFE(estimator=rf, n_features_to_select=10, random_state=42)
rfe.fit(X_normalised_cleaned, y_cleaned)

```

```
# Mendapatkan nama-nama fitur yang dipilih
fitur_terpilih_10_cleaned = x_cleaned.columns[rf.support_]

# Mendapatkan kepentingan fitur dari model Random Forest yang digunakan oleh RFE
importance_10 = rfe.estimator_.feature_importances_

# Membuat DataFrame untuk kepentingan fitur
feature_importances_10 = pd.DataFrame({
    'fitur': fitur_terpilih_10_cleaned,
    'kepentingan': importance_10
})

# Mengurutkan fitur berdasarkan kepentingannya
feature_importances_10_sorted = feature_importances_10.sort_values(by='kepentingan', ascending=False)
print(feature_importances_10_sorted)
```

	Fitur	Kepentingan
6	ZS TB/U	0.647155
5	ZS BB/U	0.157594
4	BB/U	0.094962
1	Berat	0.033782
8	ZS BB/TB	0.030855
7	BB/TB	0.013662
2	Tinggi	0.011596
3	LIJA	0.001236
9	Indeks Berat Badan	0.000811
0	JK	0.000527

SKRIPSI BIMA SATRIA

by Teknik Informatika Universitas Muhammadiyah Kalimantan Timur



Submission date: 19-Jul-2024 08:44AM (UTC+0800)

Submission ID: 2418913734

File name: skah_Skripsi_Bima_Satria_2011102441102_-_Copy_-_BIMA_SATRIA.docx (830.28K)

Word count: 9823

Character count: 60824

SKRIPSI BIMA SATRIA

ORIGINALITY REPORT

17%	12%	11%	4%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Ari Ahmad Dhani, Taghfirul Azhima Yoga Siswa, Wawan Joko Pranoto. "Perbaikan Akurasi Random Forest Dengan ANOVA Dan SMOTE Pada Klasifikasi Data Stunting", Teknika, 2024	5%
	Publication	
2	developers.google.com	1%
	Internet Source	
3	repository.ub.ac.id	1%
	Internet Source	
4	dspace.umkt.ac.id	1%
	Internet Source	
5	satudata.pasuruankota.go.id	<1%
	Internet Source	
6	api.repository.poltekesos.ac.id	<1%
	Internet Source	
7	eprints.uad.ac.id	<1%
	Internet Source	
8	jurnal.umt.ac.id	
	Internet Source	