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

Universitas Muhammadiyah Kalimantan Timur
Berkarakter | Berwawasan | Berkemajuan
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 الرَّحْمَٰنِ 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

Rosnansyah, S.Kom., M.TI
NIDN. 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	O	-		-	-
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	O	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	O	-		-	-
:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:	:
15072	Afizah	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 Arsya	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	-	-		-	-

Lamprian 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 Algortima *Random Forest + RFE + GA*

```python
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: 150474
Total data setelah penghapusan duplikat: 34201
```

```python
dataset = pd.read_csv('dataset_new.csv')
dataset.info()
dataset = data.drop('Jml 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: 34201 entries, 0 to 34200
Data columns (total 14 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Nama                34200 non-null  object
 1   JK                  34201 non-null  object
 2   Berat               34201 non-null  object
 3   Tinggi              33633 non-null  object
 4   LiLA                28598 non-null  float64
 5   BB/U                34201 non-null  object
 6   ZS BB/U             34201 non-null  object
 7   TB/U                31561 non-null  object
 8   ZS TB/U             34201 non-null  object
 9   BB/TB               31574 non-null  object
 10  ZS BB/TB            34201 non-null  object
 11  Naik Berat Badan    34201 non-null  object
 12  Jml Vit A           12663 non-null  float64
 13  Tanggal Pengukuran  34201 non-null  object
dtypes: float64(2), object(12)
memory usage: 3.7+ MB
```

```python
dataset = pd.read_csv('dataset_tanpa_duplikat.csv')
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 34201 entries, 0 to 34200
Data columns (total 13 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Nama                34200 non-null  object
 1   JK                  34201 non-null  object
 2   Berat               34201 non-null  object
 3   Tinggi              33633 non-null  object
 4   LiLA                28598 non-null  float64
 5   BB/U                34201 non-null  object
 6   ZS BB/U             34201 non-null  object
 7   TB/U                31561 non-null  object
 8   ZS TB/U             34201 non-null  object
 9   BB/TB               31574 non-null  object
 10  ZS BB/TB            34201 non-null  object
 11  Naik Berat Badan    34201 non-null  object
 12  Tanggal Pengukuran  34201 non-null  object
dtypes: float64(1), object(12)
memory usage: 3.4+ MB
```

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

```
TB/U
Normal          26943
Pendek           3350
Sangat Pendek    1125
Tinggi            143
Name: count, dtype: int64
```

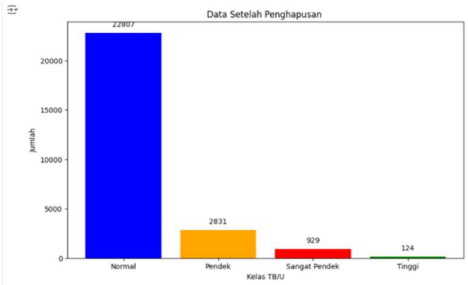
```python
import pandas as pd

# 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_isi = data.dropna()

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

# Menyimpan hasilnya ke file CSV baru (jika diperlukan)
data_tanpa_isi.to_csv('dataset_tanpa_nilai.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: 34201
Jumlah data setelah penghapusan: 26691
```

```python
data_tanpa_isi['TB/U'].value_counts()
```

```
TB/U
Normal          22807
Pendek           2831
Sangat Pendek     929
Tinggi            124
Name: count, dtype: int64
```

```python
import pandas as pd
import matplotlib.pyplot as plt

# Contoh data untuk TB/U
data_tanpa_isi = pd.DataFrame({
    'TB/U': ['Normal'] * 22807 + ['Pendek'] * 2831 + ['Sangat Pendek'] * 929 + ['Tinggi'] * 124
})

# Hitung value counts
value_counts = data_tanpa_isi['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 + 500, int(yval), ha='center', va='bottom')

# Menambahkan judul dan label sumbu
plt.title('Data Setelah Penghapusan ')
plt.xlabel('Kelas 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())
```

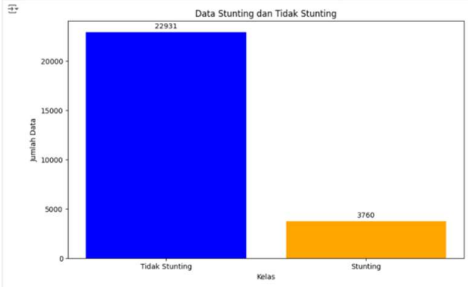
```
                Nama  JK Berat  Tinggi  LiLA              BB/U  ZS BB/U  \
0          A ALVIN    L   18.6   104.5  18.0       Risiko Lebih     1.19
1          A FADLAN   L   11.6    83.0   0.0  Berat Badan Normal    -0.08
2  A FARIS WICAKSONO  L    9.7    78.0  16.0  Berat Badan Normal    -1.75
3          A FATHAN   L     15   107.0  17.0  Berat Badan Normal    -1.08
4          A FAUJAN   L     14   100.0   0.0  Berat Badan Normal    -0.85

          TB/U  ZS TB/U              BB/TB  ZS BB/TB Naik Berat Badan  \
0  Tidak Stunting     0.41  Risiko Gizi Lebih      1.41                N
1  Tidak Stunting    -0.97          Gizi Baik      0.59                T
2       Stunting    -2.84          Gizi Baik     -0.47                O
3  Tidak Stunting     0.14          Gizi Baik     -1.03                T
4  Tidak Stunting    -0.16          Gizi Baik     -1.14                O

  Tanggal Pengukuran
0         2023-12-16
1         2023-07-10
2         2023-10-06
3         2023-10-23
4         2023-02-07
```

```
import pandas as pd
import numpy as np
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()
```

```
Kelas
Tidak Stunting    22931
Stunting           3760
Name: count, dtype: int64
```

```
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())
```

```
Kolom dalam DataFrame: Index(['JK', 'Berat', 'Tinggi', 'LiLA', 'BB/U', 'ZS BB/U', 'Kelas', 'ZS TB/U',
       'BB/TB', '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  \
0   0   18.6   104.5  18.0     2     1.19     0.41                 1      5
1   0   11.6    83.0   0.0     0    -0.08    -0.97                 3      0
2   0    9.7    78.0  16.0     0    -1.75    -2.84                 2      0
3   0     15   107.0  17.0     0    -1.08     0.14                 3      0
4   0     14   100.0   0.0     0    -0.85    -0.16                 2      0

   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	0.0	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 × 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	0.0	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 × 11 columns

RANDOM FOREST

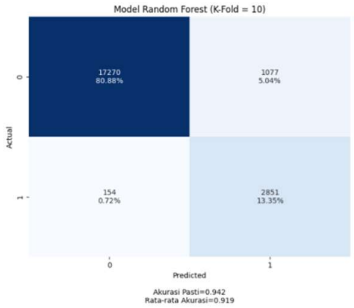
```
# Mengimpor pustaka yang diperlukan
import pandas as pd
import numpy as np
from sklearn.model_selection import cross_val_predict, KFold, train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.preprocessing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

# Fungsi untuk membuat plot matriks kebingungan
def buat_matriks_kebingungan(cf, nama_grup=None, kategori='auto', jumlah=True, persen=True, barwarna=True, xyticks=True, xyplotlabels=True, stat_ringkasan=True, ukuran_gambar=None, cmap='Blues', judul=None, akurasi_pasti=None, rata2_akurasi=None):
    kosong = ['' for i in range(cf.size)]
    if nama_grup and len(nama_grup) == cf.size:
        label_grup = ["{}\n".format(value) for value in nama_grup]
    else:
        label_grup = kosong
    if jumlah:
        jumlah_grup = ["{0:0.0f}\n".format(value) for value in cf.flatten()]
    else:
        jumlah_grup = kosong
    if persen:
        persentase_grup = ["{0:.2%}".format(value) for value in cf.flatten() / np.sum(cf)]
    else:
        persentase_grup = kosong
    label_kotak = [f"{v1}{v2}{v3}".strip() for v1, v2, v3 in zip(label_grup, jumlah_grup, persentase_grup)]
    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 += "\n\nAkurasi Pasti={:0.3f}".format(akurasi_pasti)
        if rata2_akurasi is not None:
            teks_stat += "\nRata-rata Akurasi={:0.3f}".format(rata2_akurasi)
    if ukuran_gambar is None:
        ukuran_gambar = plt.rcParams.get('figure.figsize')
    if not xyticks:
        kategori = False
    plt.figure(figsize=ukuran_gambar)
    sns.heatmap(cf, annot=label_kotak, fmt="", cmap=cmap, cbar=barwarna, xticklabels=kategori, yticklabels=kategori)
    if xyplotlabels:
        plt.ylabel('Actual')
        plt.xlabel('Predicted' + teks_stat)
```

```
    else:
        plt.xlabel(teks_stat)
    if judul:
        plt.title(judul)
    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 bermasalah di kolom 'JK'
data['JK'] = data['JK'].str.strip().replace({'L\xa0': 'L', 'P\xa0': 'P'})

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

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

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

# Memisahkan 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=0.2, random_state=42)

# Merekam hasil K-Fold 10 kali
kfolds = 10
repeats = 10
akurasi_list = []

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

```
# Melakukan cross-validation dengan variasi
for i in range(repeats):
    print(f"Rekaman {i+1}")

    # Mendefinisikan classifier Random Forest dengan parameter yang sedikit berbeda untuk setiap iterasi
    rf_classifier = RandomForestClassifier(
        n_estimators=10 + i,  # Mengubah jumlah estimators
        criterion='gini',
        max_depth=1 + i % 3,  # Mengubah kedalaman maksimal pohon
        min_samples_split=100 + i,  # Mengubah jumlah minimal sampel untuk split
        min_samples_leaf=50 - i * 5,  # Mengubah jumlah minimal sampel per daun
        max_features=0.1 + (i % 5) * 0.01,  # Mengubah jumlah fitur yang dipertimbangkan untuk split
        class_weight='balanced',
        random_state=42 + i  # Mengubah random state
    )

    y_pred = cross_val_predict(rf_classifier, x_train, 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(f"Akurasi: {akurasi:.3f}")
    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(f"Rata-rata Akurasi: {rata2_akurasi:.3f}")
print(f"Akurasi Pasti (Iterasi Terakhir): {akurasi_terakhir:.3f}")

# Menggambar plot matriks kebingungan untuk iterasi terakhir dengan rata-rata akurasi dan akurasi pasti
buat_matriks_kebingungan(cf_matrix, ukuran_gambar=(8, 6), berwarna=False, judul="Model 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      3005

      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

Tidak Stunting       0.93      0.90      0.92     18347
      Stunting       0.50      0.60      0.55      3005

      accuracy                           0.86     21352
     macro avg       0.72      0.75      0.73     21352
  weighted avg       0.87      0.86      0.87     21352

Rekaman 3
Akurasi: 0.996
              precision    recall  f1-score   support

Tidak Stunting       1.00      1.00      1.00     18347
      Stunting       0.98      1.00      0.99      3005

      accuracy                           1.00     21352
     macro avg       0.99      1.00      0.99     21352
  weighted avg       1.00      1.00      1.00     21352

Rekaman 4
Akurasi: 0.894
              precision    recall  f1-score   support

Tidak Stunting       0.97      0.90      0.94     18347
      Stunting       0.59      0.85      0.69      3005

      accuracy                           0.89     21352
     macro avg       0.78      0.88      0.82     21352
  weighted avg       0.92      0.89      0.90     21352

Rekaman 5
Akurasi: 0.968
              precision    recall  f1-score   support

Tidak Stunting       1.00      0.96      0.98     18347
      Stunting       0.82      0.99      0.90      3005

      accuracy                           0.97     21352
     macro avg       0.91      0.98      0.94     21352
  weighted avg       0.97      0.97      0.97     21352

Rekaman 6
Akurasi: 0.916
              precision    recall  f1-score   support

Tidak Stunting       0.98      0.93      0.95     18347
      Stunting       0.65      0.86      0.74      3005

      accuracy                           0.92     21352
     macro avg       0.82      0.89      0.85     21352
  weighted avg       0.93      0.92      0.92     21352

Rekaman 7
Akurasi: 0.942
              precision    recall  f1-score   support

Tidak Stunting       1.00      0.93      0.96     18347
      Stunting       0.71      0.99      0.83      3005

      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: 0.966
              precision    recall  f1-score   support

Tidak Stunting       0.99      0.97      0.98     18347
      Stunting       0.84      0.93      0.88      3005

      accuracy                           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

Tidak Stunting       1.00      0.99      1.00     18347
      Stunting       0.95      0.99      0.97      3005

      accuracy                           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

Tidak Stunting       0.99      0.94      0.97     18347
      Stunting       0.73      0.95      0.82      3005

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

Rata-rata Akurasi: 0.919
Akurasi Pasti (Iterasi Terakhir): 0.942
```



RANDOM FOREST + RFE

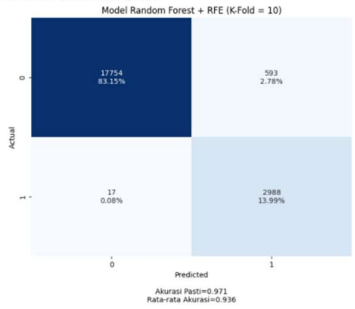
```
# Mengimpor pustaka yang diperlukan
import pandas as pd
import numpy as np
from sklearn.model_selection import cross_val_predict, KFold, train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, classification_report
from sklearn.preprocessing 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_grup=None, kategori='auto', jumlah=True, persen=True, barwarna=True, xyticks=True, xyplotlabels=True, stat_ringkasan=True, ukuran_gambar=None, cmap='Blues', judul=None, akurasi_pasti=None, rata2_akurasi=None):
    kosong = ['' for i in range(cf.size)]
    if nama_grup and len(nama_grup) == cf.size:
        label_grup = ["{}\n".format(value) for value in nama_grup]
    else:
        label_grup = kosong
    if jumlah:
        jumlah_grup = ["{0:0.0f}\n".format(value) for value in cf.flatten()]
    else:
        jumlah_grup = kosong
    if persen:
        persentase_grup = ["{0:.2%}".format(value) for value in cf.flatten() / np.sum(cf)]
    else:
        persentase_grup = kosong
    label_kotak = [f"{v1}{v2}{v3}".strip() for v1, v2, v3 in zip(label_grup, jumlah_grup, persentase_grup)]
    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 += "\n\nAkurasi Pasti={:0.3f}".format(akurasi_pasti)
        if rata2_akurasi is not None:
            teks_stat += "\nRata-rata Akurasi={:0.3f}".format(rata2_akurasi)
    if ukuran_gambar is None:
        ukuran_gambar = plt.rcParams.get('figure.figsize')
    if not xyticks:
        kategori = False
    plt.figure(figsize=ukuran_gambar)
    sns.heatmap(cf, annot=label_kotak, fmt="", cmap=cmap, cbar=barwarna, xticklabels=kategori, yticklabels=kategori)
    if xyplotlabels:
        plt.ylabel('Actual')
        plt.xlabel('Predicted' + teks_stat)
    else:
        plt.xlabel(teks_stat)
    if judul:
        plt.title(judul)
    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 bermasalah di kolom 'JK'
data['JK'] = data['JK'].str.strip().replace({'L\x00': 'L', 'P\x00': 'P'})

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

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

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

# Memisahkan 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=0.2, 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)

# Merekam hasil K-Fold 10 kali
kfolds = 10
repeats = 10
akurasi_list = []

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

```python
# Melakukan cross-validation dengan variasi
for i in range(repeats):
    print(f"Rekaman {i+1}")

    # Mendefinisikan classifier Random Forest dengan parameter yang sedikit berbeda untuk setiap iterasi
    rf_classifier = RandomForestClassifier(
        n_estimators=10 + i,  # Mengubah jumlah estimators
        criterion='gini',
        max_depth=1 + i % 3,  # Mengubah kedalaman maksimal pohon
        min_samples_split=100 + i,  # Mengubah jumlah minimal sampel untuk split
        min_samples_leaf=50 - i * 5,  # Mengubah jumlah minimal sampel per daun
        max_features=0.1 + (i % 5) * 0.02,  # Mengubah jumlah fitur yang dipertimbangkan untuk split
        class_weight='balanced',
        random_state=42 + i  # Mengubah random state
    )

    y_pred = cross_val_predict(rf_classifier, x_train_rfe, 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(f"Akurasi: {akurasi:.3f}")
    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(f"Rata-rata Akurasi: {rata2_akurasi:.3f}")
print(f"Akurasi Pasti (Iterasi Terakhir): {akurasi_terakhir:.3f}")

# Menggambar plot matriks kebingungan untuk iterasi terakhir dengan rata-rata akurasi dan akurasi pasti
buat_matriks_kebingungan(cf_matrix, ukuran_gambar=(8, 6), berwarna=False, judul="Model Random Forest - RFE (K-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 (Iterasi Terakhir): 0.971
```




```
[ ] # Instal DEAP
    !pip install deap
```

```
Collecting deap
  Downloading deap-1.4.1-cp310-cp310-manylinux_2_5_x86_64.manylinux1_x86_64.manylinux_2_17_x86_64.manylinux2014_x86_64.whl (135 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 135.4/135.4 kB 2.7 MB/s eta 0:00:00
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

```
# Mengimpor 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 kebingungan
def buat_matriks_kebingungan(cf, nama_grup=None, kategori='auto', jumlah=True, persen=True, berwarna=True, xyticks=True, xyplotlabels=True, stat_ringkasan=True, ukuran_gambar=None, cmap='Blues', judul=None, akurasi_pasti=None, rata2_akurasi=None):
    kosong = ['' for i in range(cf.size)]
    if nama_grup and len(nama_grup) == cf.size:
        label_grup = ["{}\n".format(value) for value in nama_grup]
    else:
        label_grup = kosong
    if jumlah:
        jumlah_grup = ["{0:0.0f}\n".format(value) for value in cf.flatten()]
    else:
        jumlah_grup = kosong
    if persen:
        persentase_grup = ["{0:.2%}".format(value) for value in cf.flatten() / np.sum(cf)]
    else:
        persentase_grup = kosong
    label_kotak = [f"{v1}{v2}{v3}".strip() for v1, v2, v3 in zip(label_grup, jumlah_grup, persentase_grup)]
    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 += "\n\nAkurasi Pasti={:0.3f}".format(akurasi_pasti)
        if rata2_akurasi is not None:
            teks_stat += "\nRata-rata Akurasi={:0.3f}".format(rata2_akurasi)
    if ukuran_gambar is None:
        ukuran_gambar = plt.rcParams.get('figure.figsize')
    if not xyticks:
        kategori = False
    plt.figure(figsize=ukuran_gambar)
    sns.heatmap(cf, annot=label_kotak, fmt="", cmap=cmap, cbar=berwarna, xticklabels=kategori, yticklabels=kategori)
    if xyplotlabels:
        plt.ylabel('Actual')
        plt.xlabel('Predicted' + teks_stat)
    else:
        plt.xlabel(teks_stat)
    if judul:
        plt.title(judul)
    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 bermasalah di kolom 'JK'
data['JK'] = data['JK'].str.strip().replace({'L\xa0': 'L', 'P\xa0': 'P'})

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

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

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

# Memisahkan 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=0.2, 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)

# 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)
```

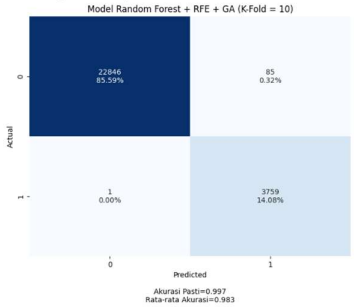
```python
# Fungsi evaluasi
def evalRF(individual):
    selected_features = [i for i, bit in enumerate(individual) if bit == 1]
    if len(selected_features) == 0:
        return 0,  # Hindari subset fitur kosong
    X_subset = X_normalized[:, selected_features]
    selector = RFE(rf, n_features_to_select=min(2, len(selected_features)), step=1)
    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.cxTwoPoint)
toolbox.register("mutate", tools.mutFlipBit, indpb=0.05)
toolbox.register("select", tools.selTournament, tournsize=3)
toolbox.register("evaluate", evalRF)

# Menjalankan GA dengan pengaturan yang lebih cepat
population = toolbox.population(n=10)  # Mengurangi ukuran populasi
ngen = 10  # Mengurangi jumlah generasi
cxpb = 0.5
mutpb = 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_normalized[:, selected_features]

# Merekam hasil K-Fold 10 kali
kfolds = 10
repeats = 10
akurasi_list = []

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

```python
# Melakukan cross-validation dengan variasi
for i in range(repeats):
    print(f"Rekaman {i+1}")

    # Mendefinisikan classifier Random Forest dengan parameter yang sedikit berbeda untuk setiap iterasi
    rf_classifier = RandomForestClassifier(
        n_estimators=10 + i,  # Mengubah jumlah estimators
        criterion='gini',
        max_depth=1 + i % 3,  # Mengubah kedalaman maksimal pohon
        min_samples_split=100 + i,  # Mengubah jumlah minimal sampel untuk split
        min_samples_leaf=50 - i * 5,  # Mengubah jumlah minimal sampel per daun
        max_features=0.1 + (i % 5) * 0.02,  # Mengubah jumlah fitur yang dipertimbangkan untuk split
        class_weight='balanced',
        random_state=42 + i  # Mengubah random state
    )

    y_pred = cross_val_predict(rf_classifier, X_best, y, cv=kfolds)
    cf_matrix = confusion_matrix(y, y_pred)
    akurasi = np.trace(cf_matrix) / float(np.sum(cf_matrix))
    akurasi_list.append(akurasi)
    print(f"Akurasi: {akurasi:.3f}")
    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(f"Rata-rata Akurasi: {rata2_akurasi:.3f}")
print(f"Akurasi Pasti (Iterasi Terakhir): {akurasi_terakhir:.3f}")

# Menggambar plot matriks kebingungan untuk iterasi terakhir dengan rata-rata akurasi dan akurasi pasti
buat_matriks_kebingungan(cf_matrix, ukuran_gambar=(8, 6), berwarna=False, judul="Model Random Forest + RFE + 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/deap/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 '{0}' has already been created and it "
/usr/local/lib/python3.10/dist-packages/deap/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 '{0}' 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                           1.00     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     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     26691
     macro avg       0.78      0.93      0.83     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                           0.95     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      0.98      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       0.98      0.99      0.98      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
```

```
Rekaman 10
Akurasi: 0.997
              precision    recall  f1-score   support

Tidak Stunting       1.00      1.00      1.00     22931
      Stunting       0.98      1.00      0.99      3760

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

Rata-rata Akurasi: 0.983
Akurasi Pasti (Iterasi Terakhir): 0.997
```



SELEKSI FITUR RFE

```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import MinMaxScaler
from sklearn.feature_selection import RFE

# Membaca data yang telah diunggah
data_path = 'dataset_stunting_samarinda_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\xa0': 'L', 'P\xa0': '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 'Naik Berat Badan' menjadi numerik menggunakan label encoding
data['Naik Berat Badan'] = data['Naik Berat Badan'].map({'N': 0, 'O': 1, 'T': 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()

# Memisahkan 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_normalized_cleaned = scaler.fit_transform(X_cleaned)

# Seleksi fitur menggunakan RFE dengan jumlah fitur yang lebih banyak (10 fitur terbaik)
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rfe = RFE(estimator=rf, n_features_to_select=10)
rfe.fit(X_normalized_cleaned, y_cleaned)
```

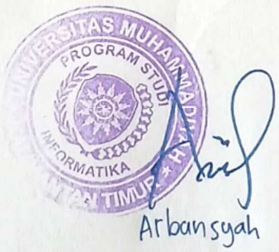
```python
# Mendapatkan nama-nama fitur yang dipilih
fitur_terpilih_10_cleaned = X_cleaned.columns[rfe.support_]

# Mendapatkan kepentingan fitur dari model Random Forest yang digunakan oleh RFE
importances_10 = rfe.estimator_.feature_importances_

# Membuat DataFrame untuk kepentingan fitur
feature_importances_10 = pd.DataFrame({
    'Fitur': fitur_terpilih_10_cleaned,
    'Kepentingan': importances_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     0.094062
1            Berat     0.031782
8          ZS BB/TB     0.030855
7             BB/TB     0.019662
2            Tinggi     0.015596
3              LILA     0.001236
9  Naik Berat Badan     0.000631
0                JK     0.000527
```

Lampiran 5 Turnitin Skripsi

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 Publication	5%
2	developers.google.com Internet Source	1%
3	repository.ub.ac.id Internet Source	1%
4	dspace.umkt.ac.id Internet Source	1%
5	satudata.pasuruankota.go.id Internet Source	<1%
6	api.repository.poltekesos.ac.id Internet Source	<1%
7	eprints.uad.ac.id Internet Source	<1%
8	jurnal.umt.ac.id Internet Source	