OPTIMASI METODE BMR DAN *SIMULATED ANNEALING* UNTUK ALGORITMA SVM DALAM MENGATASI *IMBALANCED* DAN *HIGH DIMENSIONAL* DATA STUNTING

SKRIPSI

Diajukan oleh:
Mukminatul Munawaroh
2011102441064

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS MUHAMMADIYAH KALIMANTAN TIMUR
JULI 2024

OPTIMASI METODE BMR DAN *SIMULATED ANNEALING* UNTUK ALGORITMA SVM DALAM MENGATASI *IMBALANCED* DAN *HIGH DIMENSIONAL* DATA STUNTING

SKRIPSI

Diajukan Sebagai Salah Satu Persyaratan
Untuk Memperoleh Gelar Sarjana Komputer Fakultas Sains dan Teknologi
Universitas Muhammadiyah Kalimantan Timur

Diajukan oleh:
Mukminatul Munawaroh
2011102441064

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS SAINS DAN TEKNOLOGI
UNIVERSITAS MUHAMMADIYAH KALIMANTAN TIMUR
JULI 2024

LEMBAR PERSETUJUAN

OPTIMASI METODE BMR DAN *SIMULATED ANNEALING* UNTUK ALGORITMA SVM DALAM MENGATASI *IMBALANCED* DAN *HIGH DIMENSIONAL* DATA STUNTING

SKRIPSI

Diajukan oleh:

Mukminatul Munawaroh
2011102441064

Disetujui untuk diujikan
Pada tanggal 28 Juni 2024

Pembimbing

Taghfirul Azhima Yoga Siswa, M. Kom
NIDN. 1118038805

Mengetahui,
Koordinator Skripsi

Abdul Rahim, S.Kom., M.Cs
NIDN. 0009047901

LEMBAR PENGESAHAN

OPTIMASI METODE BMR DAN *SIMULATED ANNEALING* UNTUK ALGORITMA SVM DALAM MENGATASI *IMBALANCED* DAN *HIGH DIMENSIONAL* DATA STUNTING

SKRIPSI

Diajukan Oleh:

Mukminatul Munawaroh
2011102441064

Diseminarkan dan Diujikan
Pada tanggal 9 Juli 2024

Penguji I	Penguji II
Wawan Joko Pranoto, M.TI	**Taghfirul Azhima Yoga Siswa, M. Kom**
NIDN. 1102057701	**NIDN. 1118038805**

Mengetahui,
Ketua
Program Studi Teknik Informatika

Arbansyah, S.Kom., M.TI
NIDN. 1118019

PERNYATAAN KEASLIAN PENELITIAN

Saya yang bertanda tangan di bawah ini:

Nama	: Mukminatul Munawaroh
NIM	: 2011102441064
Program Studi	: Teknik Informatika
Judul Penelitian	: Optimasi Metode BMR dan *Simulated Annealing* untuk Algoritma SVM Dalam Mengatasi *Imbalanced* dan *High Dimensional* Data Stunting

Menyatakan bahwa **skripsi** yang saya tulis ini benar-benar hasil karya saya sendiri, dan bukan merupakan hasil plagiasi/falsifikasi/fabrikasi baik sebagian atau seluruhnya.

Atas pernyataan ini, saya siap menanggung risiko atau sanksi yang dijatuhkan kepada saya apabila kemudian ditemukan adanya pelanggaran terhadap etika keilmuan dalam **skripsi** saya ini, atau klaim dari pihak lain terhadap keaslian karya saya ini

Samarinda, 11 Juli 2024
Yang membuat pernyataan

Mukminatul Munawaroh
NIM: 2011102441064

ABSTRAK

Kasus Stunting di kota Samarinda dari tahun 2022 masih mengalami peningkatan prevalensi sebesar 23,9%, penanganannya kasus ini masih kurang optimal karena tahun 2023 tingkat stunting masih tinggi. Tujuan penelitian ini meningkatkan akurasi klasifikasi atau prediksi stunting di Kota Samarinda menggunakan algoritma *Support Vector Machine* (SVM) yang akan di optimasi dengan metode *Simulated Annealing* (SA) dan metode *Bundary Margin Relief* (BMR) sebagai seleksi fitur, guna mengetahui fitur yang berpengaruh dalam menentukan atau mengklasifikasikan kondisi stunting pada anak. Proses penelitian ini melibatkan pengumpulan data dari Dinas Kesehatan Kota Samarinda tahun 2023 yang melibatkan 26 puskesmas, 20 atibut dan total 150,465 baris data. Pembagian data uji dan data latih menggunakan *cross validation* k-fold 10. Penelitian ini menunjukkan bahwa BMR menghasilkan sembilan fitur penting yaitu ZS TB/U, ZS BB/U, ZS BB/TB, Tinggi Badan, Berat Badan, Naik Berat Badan, LiLA, BB/TB, dan BB/U. Terdapat fitur dominan yang sama dengan penelitian sebelumnya yaitu atribut Berat Badan/Umur (BB/U) dan Berat Badan/Tinggi Badan (BB/TB). Evaluasi kinerja di tampilkan dalam *confusion matrix*, rata-rata akurasi SVM dengan karnel RBF, *cost* =10, *gamma* = 5 menghasilkan akurasi 54.90%, SVM dengan *Simulated Annealing* 88.10%, sehingga kenaikan akurasi sebesar 33.20%. Peningkatan akurasi menunjukkan metode ini baik dalam mengatasi stunting dengan dataset besar dan ketidakseimbangan kelas.

Kata kunci: Klasifikasi, Stunting, SVM, BMR, *Simulated Annealing*

ABSTRACT

Stunting cases in Samarinda City have continued to increase, with a prevalence of 23.9% in 2022. The handling of these cases remains suboptimal, as the stunting rate was still high in 2023. The aim of this research is to improve the accuracy of stunting classification or prediction in Samarinda City using the Support Vector Machine (SVM) algorithm, optimized with the Simulated Annealing (SA) method and the Boundary Margin Relief (BMR) method for feature selection, in order to identify influential features in determining or classifying stunting conditions in children. This research involved collecting data from the Samarinda City Health Office in 2023, involving 26 health centers, 20 attributes, and a total of 150,465 rows of data. The division of test data and training data used 10-fold cross-validation. This study showed that BMR produced nine important features: ZS Height/Age, ZS Weight/Age, ZS Weight/Height, Height, Weight, Weight Gain, MUAC, Weight/Height, and Weight/Age. Dominant features consistent with previous research are the attributes Weight/Age (W/A) and Weight/Height (W/H). Performance evaluation is presented in a confusion matrix, showing that the average accuracy of SVM with RBF kernel, cost = 10, gamma = 5 was 54.90%, while SVM with Simulated Annealing achieved 88.10%, resulting in an accuracy improvement of 33.20%. The increase in accuracy indicates that this method is effective in addressing stunting with large datasets and class imbalance.

Keywords: *Classification, Stunting, SVM, BMR, Simulated Annealing*

PRAKATA

Dengan nama Allah Yang Maha Pengasih dan Penyayang, segala puji hanya bagi-Nya. Shalawat serta salam semoga tercurahkan kepada Rasulullah Muhammad SAW, yang telah membawa petunjuk serta rahmat bagi seluruh alam. Prakata ini dibuat sebagai ungkapan terima kasih kepada semua pihak yang telah turut serta memberikan dukungan, bimbingan, dan motivasi dalam penyelesaian skripsi ini.

Penelitian ini merupakan sebuah upaya untuk mengatasi permasalahan yang sangat penting, yaitu stunting, yang telah menjadi masalah serius di banyak wilayah termasuk Kota Samarinda. Fenomena stunting tidak hanya menyangkut aspek kesehatan, tetapi juga berdampak pada aspek ekonomi dan sosial masyarakat. Oleh karena itu, penulis merasa perlu untuk melakukan penelitian yang mendalam guna memberikan kontribusi nyata dalam penanganan stunting.

Dalam penelitian ini, penulis mengusulkan sebuah pendekatan yang inovatif dengan menggabungkan metode Boundary Margin Relief (BMR) dan Simulated Annealing ke dalam algoritma Support Vector Machine (SVM). Kombinasi ini diharapkan dapat mengatasi dua masalah utama yang sering dihadapi dalam analisis data stunting, yaitu ketidakseimbangan kelas (Imbalanced data) dan dimensi yang tinggi (High Dimensional data)

Penelitian ini tidak terlepas dari bimbingan dan arahan dari para pembimbing, serta dukungan dari keluarga, teman, dan rekan-rekan RTA. Terima kasih kepada Bapak Taghfirul Azhima Yoga Siswa, M. Kom selaku dosen pembimbing yang telah memberikan waktu, pengetahuan, dan saran yang sangat berharga dalam proses penelitian ini. Tak lupa, penulis juga ingin mengucapkan terima kasih kepada bapak Wawan Joko Pranoto, M.TI selaku penguji serta semua pihak yang telah memberikan dukungan moral, doa, serta motivasi dalam setiap langkah perjalanan penelitian ini.

Akhir kata, semoga skripsi ini dapat memberikan manfaat dan kontribusi yang signifikan serta menjadi bahan referensi yang berguna bagi peneliti dan praktisi di bidang data science dan kesehatan masyarakat.

Samarinda 11 Juli 2024

DAFTAR ISI

Halaman

DAFTAR TABEL

Tabel **Halaman**

DAFTAR GAMBAR

Gambar **Halaman**

DAFTAR LAMPIRAN

Lampiran **Halaman**

BAB I

PENDAHULUAN

1.1 Latar Belakang

Stunting adalah masalah penelitian nasional yang masih trending sebagai riset nasional tahun 2020 hingga 2024, Sumber *(Ministry of Research and Technology of the republic of Indonesia,* 2020). Prevalensi balita stunting di Kalimantan Timur tahun 2022 adalah 23,9%. Stunting menjadi masalah kesehatan yang mendesak, karena dampaknya yang berkelanjutan dapat mengganggu perkembangan kognitif, produktivitas, dan kualitas hidup anak di masa dewasa. Oleh karena itu, penggunaan teknologi informasi dan metode klasifikasi data mining menjadi penting dalam upaya untuk memberikan diagnosis dini dan intervensi yang tepat guna, sehingga dapat mencegah dampak yang lebih serius pada pertumbuhan dan perkembangan anak. Salah satu metode *machine learning* dalam klasifikasi atau prediksi stunting adalah SVM (*Khan*, 2021), karena dalam situasi di mana ada perbedaan kelas yang jelas dan ruang berdimensi besar, SVM cocok untuk masalah dengan banyak fitur sehingga performa metode tidak menurun drastis ketika jumlah fitur melebihi jumlah sampel. Pada penelitian terdahulu SVM+PSO menghasilkan akurasi 78% (*Andriyani et al.*, 2023). C4.5 menghasilkan akurasi 61,82% (*Yunus et al.*, 2023)(Ula et al., 2022)(*Anggriawan & Nugroho,* 2023) dan *Naïve Bayes* menghasilkan akurasi 75% dengan data yang berdimensi rendah (*Arumi et al*., 2023)(*V. Herliansyah et al*., 2021).

Adapun masalah *high dimension* berdasarkan penelitian dari (*Huang,* 2020*; Migoń,* 2021*; Rajabi,* 2023*; Shao,* 2021*; Wang et al.,* 2021*; Xdqj et al.,* 2020*; Yun et al.,* 2023) menurut penelitian tersebut, metode *Relief* adalah algotritma yang baik dalam menenangani data berdimensi tinggi. Menurut (*Raj & Mohanasundaram,* 2020) pemilihan fitur menggunakan algoritma *Relief* yaitu *MultiSURF* dapat memperkirakan atribut yang sesuai dari kumpulan data berdimensi tinggi namun banyaknya fitur berlebihan dapat menurunkan kinerja *MultiSURF.* Untuk memilih fitur yang berkualitas dari dataset yang besar, dalam penelitiannya menyarankan, untuk menggunakan algoritma pembobotan fitur inovatif yang disebut *Boundary Margin Relief* (BMR). Konsep utama BMR adalah memprediksi bobot fitur melalui pengukuran *hyperplane* lokal, yang umumnya dilakukan dalam teknik *I-Relief*. Bobot fitur dalam metode tersebut sangat efektif dalam mengeliminasi fitur yang berlebihan. Terbukti dapat meningkatkan akurasi dengan eksperimenya pada data *Leukemia* metode BMR menghasilkan akurasi sebesar 91,2%. Oleh karena itu, penulis ingin menerapkan metode tersebut dalam penelitian data stunting di Kota Samarinda guna membantu dalam proses seleksi fitur.

Selain itu, masalah ketidakseimbangan kelas juga menambah tantangan dalam klasifikasi karena dapat menyebabkan tumpang tindih antara label atau kelas (*Hussein et al.,* 2023) sehingga penelitian ini bertujuan untuk meningkatkan kinerja SVM dengan menangani data yang tidak seimbang, menggunakan seleksi fitur dengan metode *Boundary Margin Relief* (BMR) untuk data berdimensi tinggi, dan metode *Simulated Annealing* (SA) untuk meningkatkan akurasi model SVM. SA Terbukti dapat meningkatkan akurasi SVM, berdasarkan hasil eksperimen dari (*Mahareek et al*., 2021), dalam penelitiannya, menguji metode SVM (MLP karnel) dan SA dengan data *Portuguese course* menghasilkan akurasi sebesar 78.35% tanpa SA setelah di optimasi dengan SA menjadi 90.72%. sehingga penulis ingin menerapkan metode *Simulated Annealing* (SA) sebagai metode optimasi untuk meningkatkan akurasi SVM.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang ada, masalah penelitian ini dirumuskan sebagai berikut:

a. Apasaja fitur yang mempengaruhi kinerja BMR dan *Support Vector Machine* (SVM)?
b. Apakah Metode *Simulated Annealing* (SA) dan *Boundary Margin Relief* (BMR) apabila diterapkan terhadap algoritma klasifikasi *Support Vector Machine* mengalami perbaikan performa pada nilai akurasi model tersebut?

1.3 Tujuan Penelitian

Tujuan dari penelitian ini adalah sebagai berikut:

a. Mengetahui fitur yang mempengaruhi kinerja BMR dan *Support Vector Machine* (SVM)
b. Mengetahui seberapa meningkat akurasi dengan *confusion matrix* pada metode SVM apabila SA dan BMR diterapkan untuk menangani *High Dimensional* dan *Imbalanced* pada data Stunting Kota Samarinda

1.4 Manfaat Penelitian

Penelitian ini diharapkan dapat memberikan manfaat bagi beberapa pihak yaitu:

a. Penulis
   Hasil penelitian sebagai kontribusi berupa model optimasi klasifikasi baru dalam mengatasi masalah *High Dimensional* dan *Imbalanced* data Stunting Kota Samarinda
b. Pembaca
   Mengembangkan wawasan mengenai model klasifikasi yang dapat mengoptimalkan kinerja SVM dengan metode SA dan kombinasi metode seleksi fitur BMR untuk mengatasi *High Dimension* dan *Imbalanced* pada data stunting.

1.5 Batasan Masalah

Penulis membatasi lingkup penelitian sebagai berikut untuk memastikan bahwa topik yang dibahas tetap terfokus dan tidak terlalu meluas.

a. Data yang digunakan adalah data Stunting yang diambil dari Dinas Kesehatan Kota Samarinda tahun 2023.
b. Menggunakan karnel RBF dengan nilai *cost* = 10 dan *gamma* = 5
c. Menggunakan SMOTE untuk menangani data *imbalance* pada tahap *pre-processing* data
d. Menggunkan *Boundary Margin Relief* (BMR) sebagai seleksi untuk memilih fitur-fitur yang berpengaruh tinggi dalam klasifikasi model SVM.
e. Menggunakan *Simulated Annealing* (SA) untuk meningkatkan kinerja algoritma BMR dan SVM.
f. Menggunakan *Cross-Validation* k-*Fold* 10 sebagai pembagi data uji dan data latih

BAB II

METODOLOGI PENELITIAN

2.1 Objek Penelitian

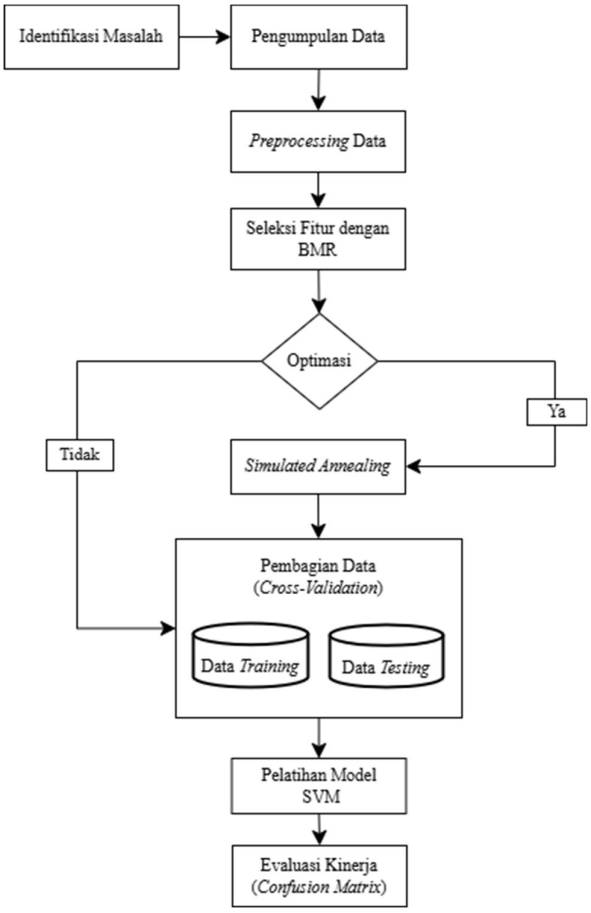
Data yang digunakan dalam penelitian ini adalah data stunting yang bersumber dari Dinas Kesehatan Jl. Milono No.1, Bugis, Kecamatan Samarinda Kota, Kota Samarinda, Kalimantan Timur 76112. Stunting adalah kondisi dimana pertumbuhan fisik seseorang, terutama anak-anak, terhambat sehingga tinggi badan tidak sesuai dengan usia mereka. Data diperoleh dari hasil pemeriksaan pasien setiap harinya baik yang mengidap penyakit stunting atau tidak. Dari 26 Puskesmas di Kota Samarinda dengan jumlah total ukur tinggi badan berdasarkan usia yaitu 15,285 jiwa seperti pada tabel berikut:

Tabel 2. 1 Sumber Data Dinas Kesehatan Kota Samarinda

No	Puskesmas	TB/U Jml Diukur
1	Palaran	351
2	Bantuas	254
3	Bukuan	560
4	Sidomulyo	442
5	Samarinda Kota	670
6	Sungai Kapih	31
7	Makroman	405
8	Sambutan	716
9	Kampung Baka	568
10	Mangkupalas	499
11	Harapan Baru	781
12	Trauma Center	977
13	Loa Bakung	1125
14	Karang Asam	630
15	Lok Bahu	456
16	Wonorejo	690
17	Pasundan	382
18	Air Putih	1628
19	Juanda	349
20	Segiri	778
21	Lempake	62
22	Sei Siring	382
23	Sempaja	379
24	Bengkuring	1080
25	Remaja	842
26	Temindung	248
	Total	15285

2.2 Prosedur Penelitian

Proses penelitian yang dilakukan secara terstruktur untuk merencanakan, melaksanakan, dan mengevaluasi suatu penelitian berdasarkan model yang digunakan, untuk memastikan bahwa penelitian berjalan lancar dan menghasilkan hasil yang dapat dipercaya. Alur penelitian ini tertera pada gamber diagram berikut:



Gambar 2. 1 Diagram Alur Penelitian

2.2.1 Identifikasi Masalah

Penanganan masalah klasifikasi data stunting dengan dimensi tinggi dan ketidakseimbangan kelas memerlukan pendekatan yang cermat. Metode SVM telah terbukti efektif dalam menangani masalah ini (J. R. Khan, p. 2021), tetapi untuk meningkatkan kinerjanya, penggunaan *Simulated Annealing* dan *Boundary Margin Relief* dalam seleksi fitur. *Simulated Annealing* digunakan untuk mencari konfigurasi parameter SVM yang optimal, sementara *Boundary Margin Relief* membantu mengidentifikasi fitur-fitur yang paling penting. Kombinasi ketiga metode ini diharapkan dapat meningkatkan akurasi klasifikasi data stunting dan memberikan dasar yang kuat untuk merancang metodologi penelitian yang efektif.

2.2.2 Pengumpulan Data

Teknik pengumpulan data berupa data sekunder, yaitu penggunaan data yang sudah ada dan dikumpulkan oleh Dinas Kesehatan Kota Samarinda dari lembaga lain untuk tujuan tertentu. Data sekunder dapat berupa data dari *survey*, laporan, basis data, rekaman administratif, atau sumber data lainnya yang tersedia untuk umum. Data yang diperoleh terdiri dari 20 kolom dengan total 150,465 *record*. Rincian data yang diperoleh adalah sebagai berikut:

Tabel 2. 2 Informasi Atribut

No	Atribut	Tipe Data	Keterangan
1	Nik	*String*	Nomor Induk Kependudukan
2	Nama	*String*	Nama
3	JK	*String*	Jenis Kelamin
4	Tel Lahir	*Date*	Tanggal Lahir
5	Nama Ortu	*String*	Nama Orang Tua
6	Provinsi	*String*	Provinsi
7	Kab/Kota	*String*	Kabupaten atau Kota
8	Kec	*String*	Kecamatan
9	Puskesmas	*String*	Lokasi Puskesmas
10	Posyandu	*String*	Lokasi Posyandu
11	Total Pengukuran	*Integer*	Total Pengukuran
12	Tanggal Pengukuran	*Integer*	Tanggal Pengukuran
13	Berat	*Integer*	Berat Badan
14	Tinggi	*Integer*	Tinggi Badan
15	BB/U	*Numeric*	Berat Badan Menurut Umur
16	ZS BB/U	*Numeric*	Z Score Berat Badan menurut Umur
17	TB/U	*Numeric*	Tinggi Badan menurut Umur
18	ZS TB/U	*Numeric*	Z Score Tinggi Badan menurut Umur
19	BB/TB	*Numeric*	Berat Badan menurut Tinggi Badan
20	ZS BB/TB	*Numeric*	Z Score Berat Badan menurut Tinggi Badan

Pada Tabel 2.2 fitur akan di seleksi lagi dan menyisakan 14 fitur yaitu Nama, JK, Berat, Tinggi, LiLA, BB/U, ZS BB/U, TB/U, ZS TB/U, BB/TB, ZS BB/TB, Naik Berat Badan, Jml Vit A, dan Tanggal Pengukuran.

Proses pengolahan data menggunakan *Python*, dengan memanfaatkan pustaka *Pandas*, yang merupakan salah satu pustaka paling populer untuk manipulasi dan analisis data yang di mulai dengan mengimpor data dari file CSV yang berjudul *'stunting.csv'*.

```
import pandas as pd
data = pd.read_csv('stunting.csv')
data.head()
```

Gambar 2. 2 Membaca Informasi Data di *Google Colab Python*

Pada Gambar 2.2, perintah tersebut adalah perintah untuk membaca file *stunting.csv* dan menyimpannya ke dalam sebuah *Dataframe* yang dinamakan data. Fungsi *head()* digunakan untuk menampilkan lima baris pertama dari *Dataframe*, memberikan gambaran awal tentang struktur dan isi data yang telah dibaca. Berikut tabel parameter dan keterangannya

Tabel 2. 3 Parameter Informasi Data Srunting

Parameter	Keterangan
pd	Alias untuk modul *Pandas* yang menyediakan fungsi untuk membaca, menulis, dan memanipulasi data.
stunting.csv	Nama file CSV yang akan dibaca
data	Nama variabel yang digunakan untuk menyimpan *Dataframe* yang berisi data dari file CSV.

head()	Fungsi yang digunakan untuk menampilkan lima baris pertama dataset dari *Dataframe*.

2.2.3 Data Preprocessing

Pada tahapan ini adalah serangkaian langkah yang dilakukan untuk membersihkan, menyiapkan, dan mengatur data mentah menjadi format yang sesuai untuk analisis lebih lanjut. Tahapan ini penting untuk memastikan data yang digunakan dalam analisis memiliki kualitas yang baik dan siap digunakan. Berikut adalah beberapa tahapannya:

A. Pembersihan Data

Pada dataset stunting di Kota Samarinda, dilakukan pembersihan data yang mencakup penanganan data yang tidak wajar, misalnya tinggi badan atau berat badan yang *ekstrem*. Penghapusan data yang memiliki nilai #N/A (nilai tidak tersedia) atau tidak memiliki nilai sama sekali, serta penghapusan data yang terduplikasi. Proses ini penting untuk memastikan bahwa data yang digunakan dalam analisis adalah *valid* dan akurat.

```
import pandas as pd

# Membaca data dari file CSV (ganti dengan lokasi file yang sesuai)
data = pd.read_csv('stunting.csv')

# 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
data.to_csv('dataset_remove_duplikat.csv', index=False)

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

Gambar 2. 3 Proses Penghapusan Data yang Terduplikat

Sebanyak 150465 *record* ditemukan pada Gambar 2.3 sebagai data terduplikasi dan dihapus dari dataset. Data yang terduplikasi dapat menyebabkan kesalahan dalam analisis karena informasi yang sama muncul lebih dari sekali.

```
dataset = pd.read_csv('dataset_remove_duplikat.csv')
dataset.info()
dataset = data.drop('Jml Vit A', axis=1)

# Menyimpan hasilnya ke file CSV baru
dataset.to_csv('dataset_without_vitA.csv', index=False)
```

Gambar 2. 4 Proses Penghapusan Atribut 'Jml Vit A'

Pada Gambar 2.4 proses penghapusan atribut *'*Jml Vit A*'* karena terdapat banyak data yang kosong atau *error 'NaN'*. Selanjutnya penghapusan data *missing* pada file *'dataset_without_vitA.csv'*

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

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

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

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

# Menyimpan hasilnya ke file CSV baru
data_tanpa_missing.to_csv('nama_file_tanpa_missing.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}")
```

Gambar 2. 5 Proses Penghapusan Data *Missing*

Terdapat 34,199 *record* yang memiliki nilai #N/A, yang juga dihapus karena tidak memberikan informasi yang berguna untuk analisis terlihat pada Gambar 2.5. Data yang memiliki nilai #N/A berarti informasi tersebut tidak tersedia atau hilang, sehingga tidak dapat digunakan untuk mendapatkan hasil analisis yang tepat. Setelah melalui proses pembersihan yang teliti ini, jumlah total data yang tersisa dan siap digunakan dalam analisis stunting di Kota Samarinda adalah 18,396 *record*. Dengan dataset yang sudah dibersihkan, analisis yang dilakukan akan lebih dapat efisien dan memberikan wawasan yang lebih akurat mengenai kondisi stunting di Kota Samarinda.

B. Transformasi Data

Langkah pertama adalah mengubah nilai-nilai atribut kategorikal menjadi bentuk numerik karena *library sklearn*, yang akan digunakan dalam analisis, hanya menerima atribut dengan nilai *numerik.* Misalnya, atribut seperti jenis kelamin, kenaikan berat badan, berat badan menurut umur, dan berat badan menurut tinggi badan harus diubah dari kategori teks menjadi angka. Selain itu, proses standarisasi atau normalisasi dilakukan pada data *numerik* seperti usia anak-anak, tinggi badan, dan berat badan. Standarisasi ini penting untuk memastikan bahwa semua data berada pada skala yang konsisten, sehingga memudahkan pembacaan dan analisis yang akurat. Transformasi variabel kategorikal menjadi numerik dilakukan dengan menggunakan fungsi *LabelEncoder* dari *library scikit-learn. LabelEncoder* adalah fungsi yang mengubah variabel kategorikal, seperti jenis kelamin dan berat badan berdasarkan umur, menjadi angka yang dapat dimengerti oleh algoritma *machine learning.*

```
# Proses Transformasi Data
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OrdinalEncoder
ordinal = OrdinalEncoder()
labelencoder = LabelEncoder()

df_transform = stunting[['JK','Berat','Tinggi','LiLA','BB/U','ZS BB/U',
                         'ZS TB/U','Naik Berat Badan','BB/TB', 'ZS BB/TB','Kelas']]

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'])

df = stunting[['JK','Berat','Tinggi','LiLA','BB/U','ZS BB/U','ZS TB/U','Naik Berat Badan','BB/TB','ZS BB/TB','Kelas']]
```

Gambar 2. 6 Proses Transformasi Data di *Python*

Pada Gambar 2.6, perintah untuk melakukan transformasi data dengan mengubah kolom kategori dalam *Dataframe* *'*stunting*'* menjadi format numerik menggunakan *'LabelEncoder'*. Kolom yang diproses adalah *'*JK*'*, *'*BB/U*'*, *'*BB/TB*'*, dan *'*Naik Berat Badan*'*. Data yang telah diubah disimpan kembali dalam *Dataframe* *'*stunting*'*, dengan kolom lainnya tetap ada. Hasilnya adalah *Dataframe* *'*df*'* yang siap digunakan untuk analisis lebih lanjut. Berikut tabel parameter dan keterangannya.

Tabel 2. 4 Parameter Proses Transformasi

Parameter	Keterangan
from sklearn.preprocessing import OrdinalEncoder	Mengimpor modul *OrdinalEncoder* dari pustaka *sklearn.preprocessing*, yang digunakan untuk mengkodekan fitur kategori menjadi bilangan bulat.
ordinal OrdinalEncoder()	Membuat objek *OrdinalEncoder* yang akan digunakan untuk mentransformasikan fitur-fitur ordinal dalam *Dataframe*.
LabelEncoder LabelEncoder()	Membuat objek *LabelEncoder* yang akan digunakan untuk mentransformasikan fitur-fitur kategori dalam *Dataframe*.
df_transform	*Dataframe* yang berisi subset kolom-kolom yang akan digunakan untuk proses transformasi.
stunting['JK'] LabelEncoder.fit_transform(stunting['JK'])	Menggunakan *LabelEncoder* untuk mengubah nilai kategori dalam kolom *'*JK*'* menjadi nilai numerik.
stunting['BB/U'] LabelEncoder.fit_transform(stunting['BB/U'])	Menggunakan *LabelEncoder* untuk mengubah nilai kategori dalam kolom *'*BB/U*'* menjadi nilai numerik.
stunting['BB/TB'] LabelEncoder.fit_transform(stunting['BB/TB'])	Menggunakan *LabelEncoder* untuk mengubah nilai kategori dalam kolom *'*BB/TB*'* menjadi nilai numerik.
stunting['Naik Berat Badan'] LabelEncoder.fit_transform(stunting['Naik Berat Badan'])	Menggunakan *LabelEncoder* untuk mengubah nilai kategori dalam kolom *'*Naik Berat Badan*'* menjadi nilai numerik.
df	*Dataframe* yang berisi kolom-kolom hasil transformasi yang telah dilakukan.

C. Data *Balanced*

Teknik-teknik untuk mengatasi masalah saat satu atau beberapa kelas dalam dataset memiliki jumlah sampel yang jauh lebih sedikit dibandingkan kelas lainnya sangat penting dalam pembelajaran mesin. Ketidakseimbangan kelas ini dapat menyebabkan model menjadi cenderung tidak adil, di mana model lebih cenderung memprediksi kelas dengan lebih banyak sampel dan mengabaikan kelas dengan lebih sedikit sampel (*Anggrawan et al*., 2023). Akibatnya, performa model secara keseluruhan menurun, terutama dalam hal prediksi kelas yang kurang terwakili. Salah satu teknik yang efektif untuk menangani masalah ini adalah SMOTE (*Synthetic Minority Over-sampling Technique*), yang bekerja dengan menghasilkan sampel sintetis untuk kelas dengan lebih sedikit sampel, sehingga meningkatkan jumlah dan keberagaman kelas tersebut dalam dataset stunting.

```
X1 = df.drop(['Kelas'],axis=1)
Y1 = df['Kelas']
from imblearn.over_sampling import SMOTE
sm = SMOTE(random_state=42, sampling_strategy=1)
X_imb, y_imb = sm.fit_resample(X1, Y1)
```

Gambar 2. 7 Proses Penanganan Ketidakseimbanagan Kelas

Implementasi pada Gambar 2.7 bertujuan untuk mengatasi masalah ketidakseimbangan kelas dengan menambahkan sampel sintetis pada kelas minoritas, sehingga meningkatkan performa model dalam memprediksi kelas yang kurang terwakili. Berikut adalah tabel parameter dan penjelasan dari setiap langkah yang dilakukan dalam proses tersebut:

Tabel 2. 5 Parameter Proses Ketidakseimbanagan Kelas

Parameter	Penjelasan
X1 = df.drop(['Kelas'], axis=1)	Membuat *Dataframe* X1 yang berisi semua kolom dari df kecuali kolom *'*Kelas*'*.
Y1 = df['Kelas']	Membuat seri Y1 yang berisi kolom *'*Kelas*'* dari *Dataframe* df.
from imblearn.over_sampling import SMOTE	Mengimpor fungsi SMOTE dari *library imblearn* untuk *oversampling* kelas minoritas.
sm = SMOTE(random_state=42, sampling_strategy=1)	Membuat objek SMOTE dengan seed random 42 dan strategi *sampling* 1 (kelas seimbang 1:1).
X_imb, y_imb = sm.fit_resample(X1, Y1)	Menerapkan SMOTE pada data X1 dan Y1, menghasilkan data baru *X_imb dan y_imb* yang seimbang.

Pada Tabel 2.5 kolom *'Kelas'* dihapus dari *Dataframe* untuk membuat *Dataframe* fitur *'X1'*, sementara kolom 'Kelas' tersebut disimpan sebagai label *'*Y1*'*. Selanjutnya, modul SMOTE diimpor dari *library 'imblearn.over_sampling',* dan objek SMOTE dibuat dengan menetapkan seed acak *'random_state=42'* dan strategi sampling *'sampling_strategy=1',* yang berarti kelas dengan sampel lebih sedikit dan kelas dengan sampel lebih banyak akan seimbang setelah *resampling*. Kemudian, metode *'fit_resample'* dari objek SMOTE diterapkan pada data fitur dan label, menghasilkan dataset

yang baru *('X_imb' dan 'y_imb')* yang telah *diresample* sehingga kelas dengan sampel lebih sedikit dan kelas dengan sampel lebih banyak menjadi seimbang.

2.2.4 Seleksi Fitur Menggunakan BMR

Seleksi fitur adalah langkah penting dalam pemrosesan data yang bertujuan untuk meningkatkan performa model prediktif dengan mengurangi jumlah fitur yang tidak relevan atau kurang signifikan. Salah satu metode yang dapat digunakan untuk seleksi fitur adalah *Boundary Margin Relief*. *Boundary Margin Relief* adalah teknik yang mengukur kepentingan setiap fitur berdasarkan margin batas antara kelas-kelas yang berbeda dalam dataset. Teknik ini membantu dalam mengidentifikasi fitur-fitur yang paling berpengaruh dalam klasifikasi, dengan mempertimbangkan jarak antara sampel dari kelas yang sama dan kelas yang berbeda. Berikut ini adalah implementasi dari metode *Boundary Margin Relief* yang digunakan untuk melakukan seleksi fitur pada dataset:

```
def boundary_margin_relief(X, y, feature_names):
    weights = np.zeros(X.shape[1])
    for i in range(len(X)):
        # Menghitung jarak ke semua sampel lain
        distances = np.linalg.norm(X - X[i], axis=1)
        # Mengabaikan jarak ke dirinya sendiri dengan mengatur ke nilai besar
        distances[i] = np.inf
        # Mencari tetangga terdekat dalam kelas yang sama (hit) dan kelas berbeda (miss)
        same_class_indices = np.where(y == y[i])[0]
        diff_class_indices = np.where(y != y[i])[0]
        nearest_hit = same_class_indices[np.argmin(distances[same_class_indices])]
        nearest_miss = diff_class_indices[np.argmin(distances[diff_class_indices])]
        weights += np.abs(X[i] - X[nearest_miss]) - np.abs(X[i] - X[nearest_hit])
    return weights

# Menggunakan fungsi yang telah dimodifikasi
feature_names = X.columns.tolist()
weights = boundary_margin_relief(X_scaled, y, feature_names)

# Mengurutkan fitur berdasarkan bobot
sorted_indices = np.argsort(weights)[::-1]
sorted_weights = weights[sorted_indices]
sorted_feature_names = [feature_names[i] for i in sorted_indices]
```

```
# Menampilkan fitur terbaik beserta bobotnya
for feature, weight in zip(sorted_feature_names[:n_features_to_display], sorted_weights[:n_features_to_display]):
    print(f"Feature: {feature}, Bobot: {weight}")
```

Gambar 2. 8 Penerapan *Boundary Margin Relief* (BMR) Untuk Bobot Fitur

Pada Gambar 2.8 menunjukkan penggunaan *Boundary Margin Relief* untuk menyeleksi fitur penting dalam dataset. Prosesnya dimulai dengan mengimpor *library* seperti *NumPy, Pandas, Matplotlib,* dan *StandardScaler* dari *scikit-learn* untuk *preprocessing* data. Dataset dibaca, dan kolom target *('Kelas')* dipisahkan. Nilai yang hilang diisi dengan rata-rata kolom, dan data distandarisasi menggunakan *StandardScaler.* Fungsi *boundary_margin_relief* menghitung bobot fitur berdasarkan jarak *Euclidean* antara sampel dari kelas yang sama dan berbeda. Jarak ke sampel sendiri diatur ke nilai tak hingga untuk mengabaikannya. Indeks tetangga terdekat dalam kelas yang sama *(nearest_hit)* dan kelas yang berbeda *(nearest_miss)* ditentukan, dan bobot fitur diperbarui dengan selisih jarak tersebut. Setelah bobot dihitung, fitur diurutkan berdasarkan bobot, dan visualisasi dibuat untuk menunjukkan fitur terpenting beserta bobotnya. Informasi ini membantu memahami fitur yang paling berpengaruh dalam klasifikasi.

Tabel 2. 6 Parameter BMR Untuk Menentukan Bobot Fitur

Parameter	Keterangan
X.fillna(X.mean(), inplace=True)	Mengisi nilai yang hilang dalam X dengan rata-rata dari masing-masing kolom.
scaler = StandardScaler()	Membuat *instance StandardScaler* untuk standarisasi data.
X_scaled = scaler.fit_transform(X)	Menstandarisasi fitur data sehingga memiliki skala yang sama.
boundary_margin_relief(X, y, feature_names)	Fungsi untuk menghitung bobot fitur menggunakan algoritma *Boundary Margin Relief.*
np.linalg.norm(X - X[i], axis=1)	Menghitung jarak *Euclidean* antara sampel i dengan semua sampel lainnya.
distances[i] = np.inf	Mengatur jarak ke sampel sendiri menjadi nilai tak terhingga untuk mengabaikannya.
same_class_indices = np.where(y == y[i])[0]	Mendapatkan indeks sampel dengan kelas yang sama.
diff_class_indices = np.where(y != y[i])[0]	Mendapatkan indeks sampel dengan kelas yang berbeda.
nearest_hit	Tetangga terdekat dalam kelas yang sama.
nearest_miss	Tetangga terdekat dalam kelas yang berbeda.
weights += np.abs(X[i] - X[nearest_miss]) - np.abs(X[i] - X[nearest_hit])	Memperbarui bobot fitur berdasarkan selisih jarak ke *nearest_miss* dan *nearest_hit.*
np.argsort(weights)[::-1]	Mengurutkan indeks fitur berdasarkan bobot dalam urutan menurun.

Selanjutnya, proses mengurangi kompleksitas model dan mempercepat proses pelatihan tanpa kehilangan informasi penting yang dapat mempengaruhi performa model. Fungsi ini dapat digunakan untuk fokus pada fitur-fitur yang paling relevan. Berikut implementasi dari fungsi tersebut:

```
# Fungsi boundary margin relief
def boundary_margin_relief(X_train, y_train):
    n_features = X_train.shape[1]
    features_to_remove = np.random.choice(n_features, n_features // 2, replace=False)
    return features_to_remove
```

Gambar 2. 9 Penerapan *Boundary Margin Relief* (BMR) Untuk Menghapus Setengah Fitur

Fungsi *boundary_margin_relief* bertujuan untuk menghapus setengah dari fitur yang ada dalam data *X_train* secara acak. Fungsi ini menghitung jumlah fitur dalam *X_train (n_features*), kemudian secara acak memilih indeks dari setengah fitur tersebut untuk dihapus dan mengembalikan *array* dari indeks fitur yang dipilih *(features_to_remove).*

2.2.5 Optimasi dengan Simulated Annealing

Simulated Annealing akan digunakan untuk mengoptimasi pemilihan fitur dalam proses pembelajaran mesin. Tujuan utamanya adalah untuk menemukan kumpulan fitur yang memberikan kinerja terbaik pada model SVM. *Simulated Annealing* adalah sebuah metode yang terinspirasi dari proses annealing dalam fisika, untuk mencari solusi optimal dalam ruang fitur dengan mengizinkan beberapa solusi yang lebih buruk diterima pada awalnya untuk menghindari terjebak di dalam daerah yang tidak optimal. Fungsi ini mengoptimalkan pemilihan fitur dengan mengevaluasi performa model pada data uji, mengubah suhu (T) dan faktor penurunan suhu (*alpha*) secara *iteratif*, dan memperbarui solusi terbaik berdasarkan skor akurasi.

```
# Simulated Annealing untuk optimasi
def simulated_annealing(X_train, y_train, X_test, y_test, svm, T=1.0, alpha=0.95, max_iter=100):
    best_score = 0.0
    best_features = None
    features_to_remove = []

    for _ in range(max_iter):
        # Pilih fitur yang akan dihapus
        new_features_to_remove = boundary_margin_relief(X_train, y_train)

        # Evaluasi model dengan fitur yang dipilih
        svm_copy = copy.deepcopy(svm)
        svm_copy.fit(X_train[:, [i for i in range(X_train.shape[1]) if i not in new_features_to_remove]], y_train)
        y_pred = svm_copy.predict(X_test[:, [i for i in range(X_test.shape[1]) if i not in new_features_to_remove]])
        score = accuracy_score(y_test, y_pred)

        # Simulated Annealing
        if score > best_score or np.exp((score - best_score) / T) > random.random():
            best_score = score
            best_features = new_features_to_remove
            features_to_remove = new_features_to_remove

        T *= alpha

    return best_score, features_to_remove

# Panggil fungsi simulated annealing untuk optimasi
best_score, selected_features = simulated_annealing(X_train_scaled, y_train, X_test_scaled, y_test, svm)
```

Gambar 2. 10 Penerapan *Simulated Annealing* (SA)

Pada Gambar 2.10 menunjukkan fungsi *simulated_annealing* yang mengoptimalkan pemilihan fitur untuk model SVM menggunakan algoritma *simulated annealing*. Fungsi ini menerima data pelatihan dan pengujian, model SVM, serta parameter temperatur awal, faktor pendinginan, dan jumlah iterasi maksimum. Dalam setiap *iterasi*, fitur yang akan dihapus dipilih dengan *boundary_margin_relief*, lalu model SVM dilatih dan diuji untuk menghitung akurasi. Jika akurasi lebih baik atau diterima berdasarkan probabilitas tertentu, fitur tersebut diterima dan skor diperbarui. Temperatur dikurangi setiap *iterasi* dengan mengalikan T dengan *alpha*. Fungsi ini mengembalikan skor terbaik dan fitur yang dipilih.

Tabel 2. 7 Parameter *Simulated Annealing*

Parameter	Keterangan
X_test , y_test	*X_test Array* fitur dari data pengujian yang digunakan untuk menguji model klasifikasi. *y_test Array* label atau kelas target dari data pengujian yang digunakan untuk mengevaluasi performa model klasifikasi

T	Temperatur awal dalam algoritma *Simulated Annealing*. Nilai *defaultnya* adalah 1.0.
alpha	Faktor penurunan temperatur dalam setiap *iterasi* algoritma *Simulated Annealing*. Nilai *defaultnya* adalah 0.95
max_iter	Jumlah iterasi maksimum yang akan dilakukan dalam algoritma *Simulated Annealing*. Nilai *defaultnya* adalah 100.
best_Score	Variabel yang menyimpan nilai akurasi terbaik yang dicapai setelah optimasi.
best_features	Variabel yang menyimpan fitur-fitur terbaik yang dipilih setelah optimasi.
features_to_remove	Variabel yang menyimpan *Array* indeks-indeks fitur yang akan dihapus dari data pelatihan setelah proses optimasi.

2.2.6 Pembagian Data dengan *Cross Validation*

Data stunting akan dibagi menjadi data pelatihan (*training*) dan data uji (*testing*) menggunakan metode *cross-validation*. Pembagian data dengan *cross-validation* yaitu pembagian dataset menjadi subset yang disebut (*folds*), di mana model dilatih pada beberapa *fold* dan diuji pada *fold* lainnya secara bergantian (*R. R. R. Arisandi*, 2022.).

```
import numpy as np
from sklearn.model_selection import StratifiedKFold
from sklearn.preprocessing import StandardScaler
from sklearn.datasets import make_classification

# Generate data
X, y = make_classification(n_samples=1000, n_features=10, n_classes=2, random_state=42)

# Preprocessing data
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Inisialisasi Stratified K-Fold Cross-Validation dengan 10 lipatan
kfold = StratifiedKFold(n_splits=10, shuffle=True, random_state=42)

# Loop melalui setiap lipatan cross-validation
for fold, (train_index, test_index) in enumerate(kfold.split(X_scaled, y), 1):
    X_train, X_test = X_scaled[train_index], X_scaled[test_index]
    y_train, y_test = y[train_index], y[test_index]
```

Gambar 2. 11 Penerapan *Cross Validation* di *Python*

Proses pembagian data pada Gambar 2.11 dimulai dengan mengimpor pustaka yang diperlukan, yaitu *numpy* dan komponen dari *sklearn* untuk penanganan data dan model, termasuk *StratifiedKFold* untuk validasi silang, serta *StandardScaler* untuk normalisasi data. Data sintetik dikembangkan menggunakan *make_classification* dengan 1000 sampel dan 10 fitur, kemudian dilakukan *preprocessing* dengan *StandardScaler* untuk menyamakan skala fitur-fitur tersebut. Setelah itu, *Stratified K-Fold Cross-Validation* dengan 10 *fold* diinisialisasi, di mana data akan diacak dan dibagi menjadi 10 bagian, memastikan distribusi kelas yang seimbang pada setiap *fold*. Proses ini dilakukan dalam *loop,* di mana setiap *iterasi* mengambil kumpulan data pelatihan dan pengujian berdasarkan indeks yang dihasilkan oleh *StratifiedKFold,* memisahkan data untuk pelatihan dan pengujian dengan cara yang menjaga proporsi kelas di setiap *fold*.

Tabel 2. 8 Parameter *Cross Validation*

Parameter	Keterangan
n_samples=1000	Menentukan jumlah sampel dalam dataset yang dihasilkan.
n_features=10	Menentukan jumlah fitur (atribut) dalam dataset yang dihasilkan.
n_classes=2	Menentukan jumlah kelas dalam dataset, yaitu biner (2 kelas).
random_state=42	Menentukan *seed* untuk pengacakan, yang memastikan hasil yang konsisten dan dapat direproduksi.
n_splits=10	Menentukan jumlah *fold* dalam Stratified *K-Fold cross-validation.*
shuffle=True	Menentukan apakah data harus diacak sebelum dibagi menjadi *fold*.
scaler = StandardScaler()	Inisialisasi objek untuk menstandarisasi fitur data.
X_scaled = scaler.fit_transform(X)	Pengaplikasian standar skala pada data X, mengubah fitur menjadi skala standar (*mean 0, variance 1*).
train_index	Indeks dari sampel yang digunakan untuk pelatihan pada *fold* tertentu.
test_index	Indeks dari sampel yang digunakan untuk pengujian pada *fold* tertentu.
fold	Nomor *fold* yang sedang diproses dalam *loop*, dimulai dari 1.

2.2.7 Pelatihan Model SVM

Support Vector Machine (SVM) adalah algoritma pembelajaran yang digunakan untuk tugas-tugas klasifikasi dan regresi. Tujuan utama dari SVM adalah untuk menemukan *hyperplane* terbaik yang memisahkan dua kelas dalam ruang fitur sedemikian rupa sehingga jarak antara *hyperplane* dan *instance* terdekat dari masing-masing kelas (yang disebut sebagai *support vectors*) sebesar mungkin. Berikut persamaan yang dapat digunakan untuk mendapatkan hyperplane pada SVM.

$$(w \,.\, x_i) + b = 0 \tag{2.1}$$

Data *xi* yang termasuk dalam kelas -1 dapat dirumuskan sebagai berikut:

$$(w \,.\, x_i) + b < 1,\ y_i = -1 \tag{2.2}$$

Data *xi* yang termasuk dalam kelas +1 dapat dirumuskan sebagai berikut:

$$(w \,.\, x_i) + b > 1,\ y_i = 1 \tag{2.3}$$

Keterangan:

x_i = Data ke-i

w = Nilai bobot support vector yang tegak lurus dengan *hyperplane* (*weight vector*)

b = Nilai bias (merujuk pada parameter yang menentukan posisi hyperplane terhadap titik asal (origin) dalam ruang fitur)

y_i = Kelas/label data ke-i

Dalam proses klasifikasi dengan SVM, seringkali terjadi kondisi di mana kernel linear tidak bekerja dengan baik, yang menyebabkan hasil klasifikasi data yang buruk. Kondisi ini dapat diatasi dengan menggunakan kernel *Radial Basis Function* (RBF) (*Al-Mejibli et al*., 2020), yang dirumuskan sebagai berikut:

$$K(x_1, x_2) = \exp(-\gamma \parallel x_1 - x_2 \parallel^2) \quad (2.4)$$

Keterangan:

$K(x_1, x_2)$ = fungsi kernel RBF yang mengukur kesamaan antara dua sampel x_1 dan x_2

$\parallel x_1 - x_2 \parallel$ = Jarak Euclidean kuadrat antara dua vektor fitur x_1 dan x_2

γ (gamma) = Parameter yang menentukan "lebar" fungsi RBF

exp = Fungsi eksponensial

Pelatihan model SVM akan menggunakan karnel RBF, cost = 10, gamma = 5 untuk membuktikan teori dari (Rahmi et al., 2022). Model SVM akan di latih menggunakan data *training* setelah penanganan ketidakseimbangan kelas, *high dimension* dengan BMR dan optimasi dengan *Simulated Annealing.*

```
from sklearn.svm import SVC

# Parameter SVM
C = 10
gamma = 5

# Model SVM
svm = SVC(kernel='rbf', C=C, gamma=gamma)  # kernel 'rbf'
svm.fit(X_train, y_train)

y_pred_svm = svm.predict(X_test)
accuracy_svm = accuracy_score(y_test, y_pred_svm) * 100  # Menjadi persen
accuracies_svm.append(accuracy_svm)


# Model SVM dengan SA
accuracy_optimized, best_features = simulated_annealing(X_train, y_train, X_test, y_test)
accuracy_optimized *= 100  # Menjadi persen
accuracies_optimized.append(accuracy_optimized)

svm_optimized = SVC(kernel='rbf', C=C, gamma=gamma)  # kernel 'rbf'
svm_optimized.fit(X_train[:, [i for i in range(X_train.shape[1]) if i not in best_features]], y_train)
y_pred_optimized = svm_optimized.predict(X_test[:, [i for i in range(X_test.shape[1]) if i not in best_features]])
```

Gambar 2. 12 Penerapan Model SVM

Pada Gambar 2.12, terdapat dua proses pemodelan, pertama menggunakan *Support Vector Machine* (SVM) dengan *kernel* RBF untuk membuat, melatih, dan mengevaluasi model. Dengan menggunakan *library Scikit-Learn,* model SVM (SVC) didefinisikan dengan *kernel 'rbf'*. Setelah dilatih dengan data pelatihan (*X_train dan y_train),* model digunakan untuk memprediksi label dari data uji *(X_test)* dan kemudian menghitung akurasinya menggunakan *accuracy_score().* Akurasi hasilnya disimpan dalam list *accuracies_svm.*

Kedua, proses optimasi dilakukan dengan menggunakan *Simulated Annealing* (SA) untuk menentukan fitur terbaik yang akan digunakan dalam model SVM kedua *(svm_optimized).* Setelah dilatih dengan fitur terpilih, model tersebut digunakan untuk membuat prediksi terhadap *X_test,* dan hasilnya dievaluasi untuk mendapatkan akurasi yang dioptimalkan *(accuracy_optimized),* yang juga dimasukkan ke dalam list *accuracies_optimized.*

Tabel 2. 9 Parameter Model SVM

Parameter	Keterangan
SVC	Kelas dari *Scikit-Learn* yang digunakan untuk membuat model *Support Vector Machine*.
kernel='rbf'	Parameter yang menentukan jenis *kernel* yang digunakan oleh model SVM. *'rbf'* adalah *Radial Basis Function*, yang umum digunakan untuk masalah *non-linear.*
X_train, y_train	*X_train* adalah data latih yang digunakan untuk melatih model SVM, sedangkan *y_train* adalah label dari data latih tersebut.
X_test	*X_test* adalah data uji yang digunakan untuk melakukan evaluasi atau prediksi menggunakan model SVM yang sudah dilatih.
accuracy_score(y_test, y_pred_svm)	Fungsi yang digunakan untuk menghitung akurasi prediksi dengan membandingkan label sebenarnya *(y_test)* dengan label yang diprediksi *(y_pred_svm).*
simulated_annealing(X_train, y_train, X_test, y_test)	*simulated_annealing* adalah fungsi yang dioptimalkan untuk memperbaiki model SVM dengan cara *Simulated Annealing.* Mengembalikan akurasi terbaik dan fitur terbaik yang dipilih.

2.2.8 Evaluasi Kinerja Algoritma

Kinerja model SVM yang dioptimalkan akan dievaluasi menggunakan *Confusion Matrix. Confusion Matrix* adalah sebuah tabel yang digunakan dalam evaluasi klasifikasi untuk menampilkan performa model dengan membandingkan hasil prediksi dengan nilai sebenarnya dari data yang diuji.

```
# Confusion matrix
from sklearn.metrics import confusion_matrix

# Menghitung confusion matrix SVM
confusion_matrices_svm.append(confusion_matrix(y_test, y_pred_svm))

# Menghitung confusion matrix SVM dengan SA
confusion_matrices_optimized.append(confusion_matrix(y_test, y_pred_optimized))
```

Gambar 2. 13 Evaluasi Kinerja Model dengan *Confusion Matrix*

Pada Gambar 2.13 proses *confusion matrix* untuk prediksi model SVM standar dihitung dengan membandingkan label sebenarnya *(y_test)* dengan label yang diprediksi oleh model *(y_pred_svm),* dan hasilnya ditambahkan ke daftar confusion_matrices_svm. Selanjutnya, *confusion matrix* untuk model SVM yang dioptimalkan dihitung dengan membandingkan *y_test* dengan prediksi dari model yang dioptimalkan *(y_pred_optimized),* dan hasilnya disimpan dalam daftar *confusion_matrices_optimized*. Confusion Matrix ini memberikan gambaran tentang performa model dalam hal prediksi benar, salah positif, salah negatif, dan salah klasifikasi.

Tabel 2. 10 Parameter *Confusion Matrix*

Parameter	Keterangan
confusion_matrix	Fungsi dari *sklearn.metrics* yang digunakan untuk menghitung confusion matrix dari hasil prediksi model.
y_test	Label sebenarnya dari data uji yang digunakan sebagai referensi untuk menghitung akurasi prediksi model.
y_pred_svm	Label yang diprediksi oleh model SVM standar untuk data uji.
y_pred_optimized	Label yang diprediksi oleh model SVM yang dioptimalkan menggunakan Simulated Annealing (SA) untuk data uji.
confusion_matrices_svm	Daftar (list) yang menyimpan confusion matrix untuk setiap *fold* dari model SVM standar.
confusion_matrices_optimized	Daftar (list) yang menyimpan confusion matrix untuk setiap *fold* dari model SVM yang dioptimalkan dengan SA.

Confusion Matrix adalah alat penting untuk mengukur kinerja model. *Confusion matrix* menunjukkan jumlah prediksi yang benar dan salah dengan membandingkan hasil prediksi model dengan nilai sebenarnya. Berikut adalah tabel *confusion matrix*:

Tabel 2. 11 *Confusion Matrix*

Prediction	***True values***	
	True	***False***
True	TP *Correct result*	FP *Unexpected result*
False	FN *Missing result*	TN *Correct absence of result*

Dalam *confusion matrix*, terdapat beberapa istilah yang digunakan dalam kasus klasifikasi seperti yang dijelaskan oleh (*Anggrawan et al*., 2023).

Keterangan :

True Positive (TP) = Data positif pada confusion matrix yang terdeteksi dengan benar.

False Positive (FP) = Data negatif pada confusion matrix yang salah terdeteksi sebagai data positif.

False Negative (FN) = Data positif pada confusion matrix yang salah terdeteksi sebagai data negatif.

True Negative (TN) = Data negatif pada confusion matrix yang terdeteksi dengan benar

Evaluasi kinerja model yang akan dilakukan dengan mengukur nilai akurasi, berikut rumusnya

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (2.5)$$

Keterangan :

Accuracy = Mengukur sejauh mana model klasifikasi memberikan prediksi yang benar secara keseluruhan.

Performa model akan dibandingkan dengan metode SVM dengan metode SVM + BMR + *Simulated Annealing* untuk menilai efektivitas kombinasi model.

Berikut adalah kategori akurasi berdasarkan persentase:

Sangat Baik : 90% - 100%
Baik : 80% - 89%
Cukup : 70% - 79%
Kurang : 60% - 69%
Sangat Kurang : < 60%

BAB III

HASIL & PEMBAHASAN

3.1 Hasil Penelitian

Pada bagian ini akan menampilkan hasil dari setiap tahapan penelitian yang dilakukan. Tahapan tersebut meliputi preprocessing data, pembagian data, penerapan metode Simulated Annealing untuk optimasi parameter BMR pada algoritma SVM, serta evaluasi kinerja model yang telah dioptimasi. Hasil yang diperoleh dari setiap tahapan akan dijelaskan secara rinci untuk memberikan gambaran lengkap mengenai efektivitas metode yang digunakan dalam penelitian ini. Berikut adalah hasil dari tahapan-tahapan tersebut:

3.1.1 Hasil *Preprocessing* Data

Pada tahap ini ada bebrapa proses yang akan di tampilkan yaitu hasil data *cleaning*, trasformasi data dan penenganan ketidakseimbangan kelas.

A. Hasil Data *Cleaning*

Berikut adalah perbandingan data mentah sebelum dan sesudah melalui tahap *preprocessing*, menunjukkan peningkatan kualitas data setelah proses data *cleaning.*

Tabel 3. 1 Dataset Stunting yang Belum di Bersihkan

No	Nama	JK	Berat	Tinggi	LiLA	BB/U	ZS BB/U	TB/U	ZS TB/U	BB/TB	ZS BB/TB	Naik Berat Badan	Jml Vit A	Tanggal Pengukuran
0	DIMAS ADITYA	L	9.01	NaN	0.0	Berat Badan Normal	-0.390000	Normal	-0.21	Gizi Baik	-0.39	O	NaN	2023-01-02
1	SITI AISYAH	P	12.00	94.0	0.0	Kurang	-2.250000	Pendek	-2.09	Gizi Baik	-1.46	O	NaN	2023-01-02
2	M AL FATIH	L	8.01	69.0	0.0	Berat Badan Normal	-0.530000	Normal	-0.65	Gizi Baik	-0.14	O	NaN	2023-01-02
3	ALMAHIRA AKIRA AKBAR	P	6.03	NaN	0.0	Berat Badan Normal	-0.310000	Normal	0.42	Gizi Baik	-0.74	O	NaN	2023-01-02
4	GIUNIA QAMELA CALANDRA	P	10.06	NaN	0.0	Risiko Lebih	0.090972	Normal	0.23	Gizi Lebih	0.09	O	NaN	2023-01-02
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
150460	ADNAN IRAGUSTI	L	3.00	50.0	NaN	Berat Badan Normal	-0.730000	Normal	0.06	Gizi Baik	-1.19	-	NaN	2023-12-30
150461	SIENA AL RAISHA RAMADHANI	P	13.00	NaN	0.0	Berat Badan Normal	-1.220000	Normal	-1.11	Gizi Baik	-0.85	O	NaN	2023-12-13
150462	AFIZAH KHAIRINA	P	2.05	45.0	NaN	Kurang	-2.030000	Pendek	-2.63	Gizi Baik	-0.35	-	NaN	2023-12-09
150463	M ARSYA KHOLIF	P	3.00	49.0	NaN	Berat Badan Normal	-1.560000	Normal	-1.30	Gizi Baik	-1.04	-	NaN	2023-12-19
150464	MUHAMMAD IQBAL	L	2.09	49.0	NaN	Kurang	-2.970000	Pendek	-2.48	Gizi Baik	-1.37	-	NaN	2023-12-29

Pada Tabel 3.1 data berjumlah 150,465 baris didalamnya masih banyak data terduplikat, data *missing* dan data *error (NaN)*.

Tabel 3. 2 Dataset Stunting yang Sudah di Bersihkan

No	JK	Berat	Tinggi	LiLA	BB/U	ZS BB/U	Kelas	ZS TB/U	BB/TB	ZS BB/TB	Naik Berat Badan
0	L	11.06	83.0	0.0	Berat Badan Normal	-0.08	Tidak Stunting	-0.97	Gizi Baik	0.04	A
1	L	9.07	78.0	16.0	Berat Badan Normal	-1.75	Stunting	-2.84	Gizi Baik	-0.47	O
2	L	15.00	107.0	17.0	Berat Badan Normal	-1.08	Tidak Stunting	0.14	Gizi Baik	-1.83	T
3	L	14.00	100.0	0.0	Berat Badan Normal	-0.85	Tidak Stunting	-0.16	Gizi Baik	-1.14	O
4	L	9.03	78.0	0.0	Berat Badan Normal	-1.33	Tidak Stunting	-1.31	Gizi Baik	-0.99	N
...	...	...	...	...	...	...	...	...	...	...	...
18391	P	13.07	100.0	14.0	Berat Badan Normal	-1.07	Tidak Stunting	-0.45	Gizi Baik	-1.18	O
18392	P	12.08	95.0	16.0	Kurang	-2.36	Stunting	-2.78	Gizi Baik	-0.93	O
18393	L	14.05	102.0	0.0	Berat Badan Normal	-1.71	Tidak Stunting	-1.62	Gizi Baik	-1.15	T
18394	L	17.03	107.0	0.0	Berat Badan Normal	-0.14	Tidak Stunting	-0.13	Gizi Baik	-0.13	O
18395	L	15.06	102.0	0.0	Berat Badan Normal	-0.59	Tidak Stunting	-0.69	Gizi Baik	-0.25	N

Pada Tabel 3.2 dataset sudah melewati proses penghapusan duplikat dari 150,465 menjadi 34,199 *record* dan penghapusan data missing dari 34,199 menjadi 18,396 *record*.

B. Hasil Transformasion Data

Berikut adalah hasil transformasi data yang menunjukkan perubahan signifikan dalam pola dan struktur data, sehingga mempermudah analisis lebih lanjut.

Tabel 3. 3 Data Sebelum di Transformasi

No	JK	BB/U	Naik Berat Badan	BB/TB
0	L	Berat Badan Normal	A	Gizi Baik
1	L	Berat Badan Normal	O	Gizi Baik
2	L	Berat Badan Normal	T	Gizi Baik
3	L	Berat Badan Normal	O	Gizi Baik
4	L	Berat Badan Normal	N	Gizi Baik
...	...	...	...	...
18391	P	Berat Badan Normal	O	Gizi Baik
18392	P	Kurang	O	Gizi Baik
18393	L	Berat Badan Normal	T	Gizi Baik
18394	L	Berat Badan Normal	O	Gizi Baik
18395	L	Berat Badan Normal	N	Gizi Baik

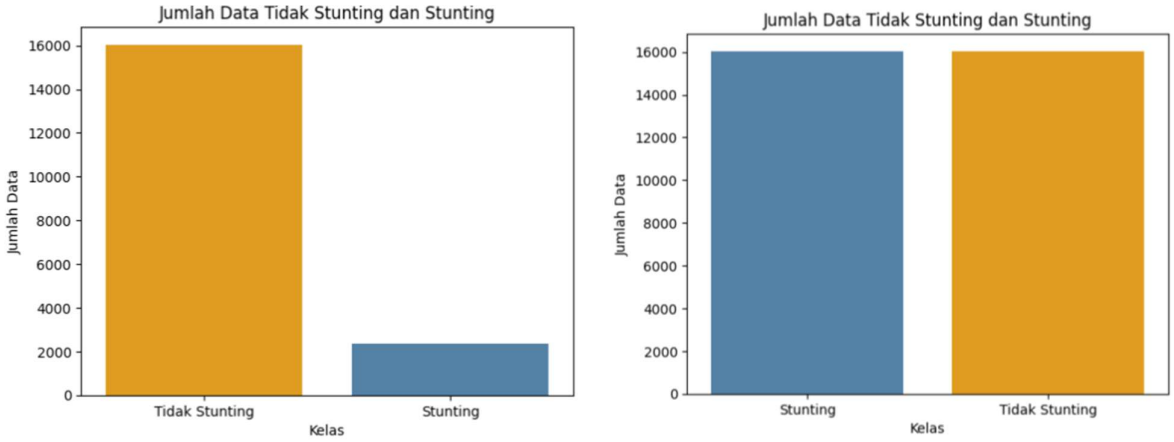
Pada Tabel 3.3 aribut yang akan di ubah adalah isi pada fitur JK, BB/U, Naik Berat Badan, dan BB/TB. Dalam Fitur JK ‘L’ akan di ubah menjadi 0 dan ‘P’ menjadi 1. Pada Fitur BB/U ‘Berat Badan Normal’ di ubah menjadi 0 dan berat badan yang ‘Kurang’ di ubah menjadi 1. Selanjutnay pada fitur Naik Berat Badan, setiap huruf akan di ubah menjadi numerik berdasarkan hasil *LabelEncoder*, seperti huruf A (pengganti nilai kosong atau -) menjadi 9, N (Normal) menjadi 10, O (Obesitas) menjadi 11 dan T (Tinggi) menjadi 12. Pada fitur BB/TB ‘Gizi Baik’ di ubah menjadi 0 dan ‘Gizi Buruk’ menjadi 1.

Tabel 3. 4 Hasil Transformasi

No	JK	BB/U	Naik Berat Badan	BB/TB
0	0	0	9	0
1	0	0	11	0
2	0	0	12	0
3	0	0	11	0
4	0	0	10	0
...	...	...	...	...
18391	1	0	11	0
18392	1	1	11	0
18393	0	0	12	0
18394	0	0	11	0
18395	0	0	10	0

C. Hasil Penanganan Ketidakseimbangan Kelas

Berikut adalah hasil penanganan ketidakseimbangan data yang menunjukkan kelas yang lebih merata

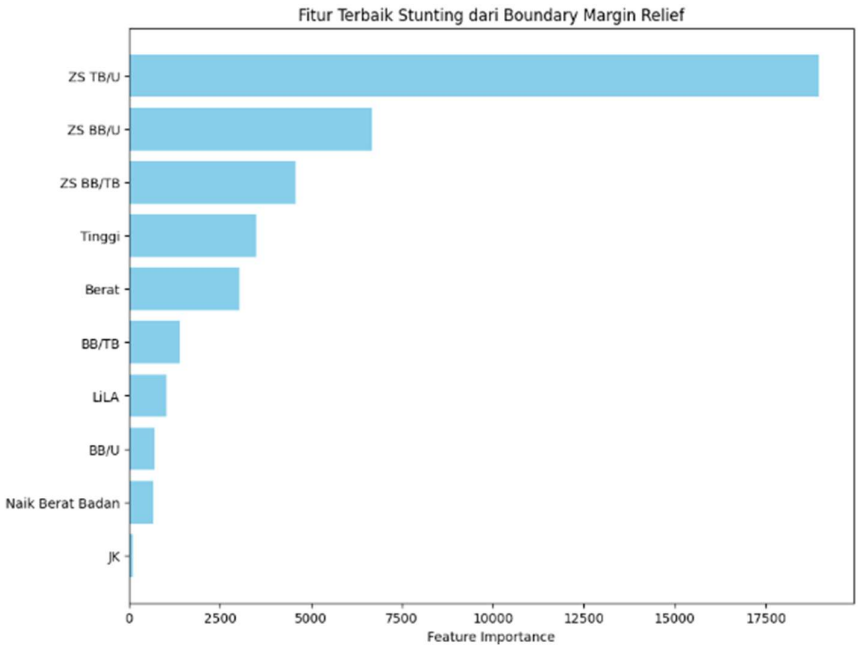


Gambar 3. 1 Hasil Penanganan Ketidakseimbangana Data

Pada Gambar 3.1, Grafik pertama menunjukkan distribusi awal data dengan ketidakseimbangan yang signifikan antara jumlah data "Tidak Stunting" sebanyak 16,046 baris data dan "Stunting" sebanyak 23,50 baris data, di mana data "Tidak Stunting" jauh lebih banyak dibandingkan data "Stunting". Grafik kedua memperlihatkan hasil setelah penanganan ketidakseimbangan data, di mana jumlah data antara "Stunting" dan "Tidak Stunting" telah disamakan, yaitu 16,046 baris data. Penanganan ketidakseimbangan data ini dilakukan untuk memastikan bahwa model pembelajaran tidak berat sebelah terhadap kelas dominan dan dapat memprediksi dengan lebih akurat untuk kedua kelas tersebut.

3.1.2 Hasil Seleksi Fitur dengan BMR

Seleksi fitur menggunakan *Boundary Margin Relief* menghasilkan bebrapa fitur terbaik yang memiliki nilai bobot tinggi. Terlihat pada gambar grafik di bawah ini:



Gambar 3. 2 Hasil Fitur Terbaik BMR

Berdasarkan hasil implementasi, fitur yang mempengaruhi kinerja BMR dan *Support Vector Machine* (SVM) terlihat pada tabel berikut berdasarkan nilai bobotnya:

Tabel 3. 5 Bobot Fitur

No	Fitur	Bobot	Peringkat
1	ZS TB/U	18964	1
2	ZS BB/U	6681	2
3	ZS BB/TB	4569	3
4	Tinggi	3507	4
5	Berat	3040	5
6	Naik Berat Badan	1398	6
7	LiLA	1026	7
8	BB/TB	700	8
9	BB/U	679	9
10	JK	112	10

Berdasarkan Tabel 3.5 fitur yang dipilih adalah ZS TB/U, ZS BB/U, ZS BB/TB, Tinggi, Berat, Naik Berat Badan, LiLA, BB/TB, dan BB/U.

3.1.3 Hasil Optimasi SVM dengan *Simulated Annealing*

Berikut adalah tabel untuk menampilkan hasil SVM dan optimasi SVM dengan *simulated annealing* berdasarkan 10 *fold:*

Tabel 3. 6 Perbandingan Akurasi SVM

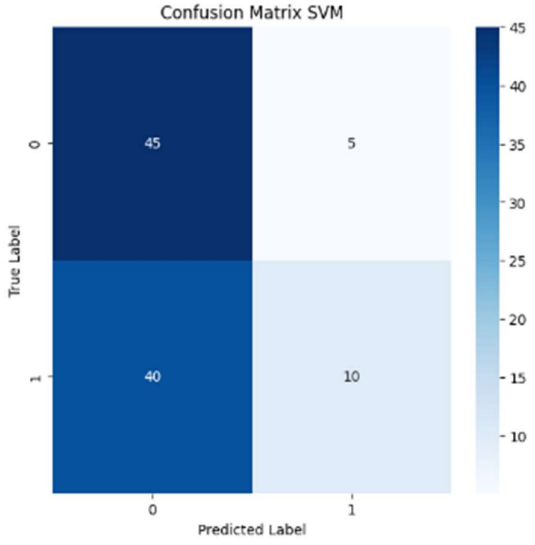
Fold	Akurasi SVM	Akurasi SVM Dengan SA	Kenaikan Akurasi
1	61.00%	86.00%	25.00%
2	54.00%	82.00%	28.00%
3	51.00%	87.00%	36.00%
4	58.00%	87.00%	29.00%
5	55.00%	91.00%	36.00%
6	54.00%	88.10%	34.00%
7	51.00%	90.00%	39.00%
8	50.00%	94.00%	44.00%
9	58.00%	84.00%	26.00%
10	57.00%	92.00%	35.00%
Rata-rata	**54.90%**	**88.10%**	**33.20%**

Rata-rata akurasi SVM adalah 54.90%, sedangkan rata-rata akurasi SVM dengan *simulated annealing* adalah 88.10%. Terdapat kenaikan rata-rata akurasi sebesar 33.20% ketika menggunakan *simulated annealing* untuk optimasi SVM serta karnel RBF dengan nilai cost = 10 dan gamma = 5.

3.1.4 Hasil Evaluasi Performa dengan *Confusion Matrix*

Menggunakan *confusion matrix* untuk mengevaluasi kinerja model klasifikasi yang dikembangkan. *Confusion matrix* adalah tabel yang memperlihatkan perbandingan antara prediksi yang benar dan salah yang dilakukan oleh model terhadap data yang telah diketahui kelasnya. Dari *confusion matrix* ini, dapat dilihat seberapa baik model dalam mengklasifikasikan data ke dalam kategori yang benar, serta mengidentifikasi jenis-jenis kesalahan prediksi yang dilakukan oleh model tersebut.

a. Model SVM



Gambar 3. 3 Hasil *Confusion Matrix* Model SVM

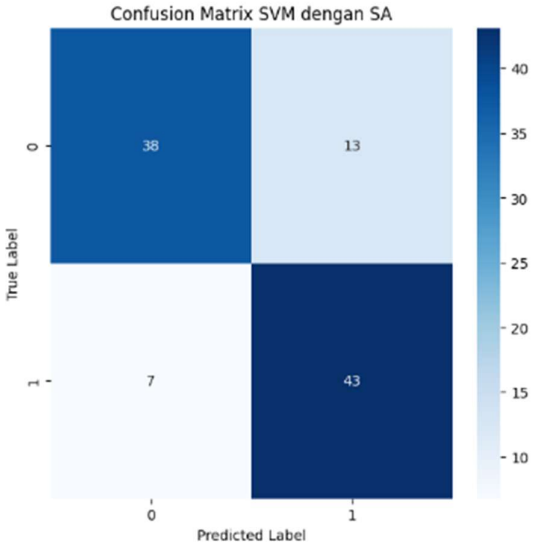
Untuk menghitung akurasi secara manual dari *confusion matrix* dapat menggunakan rumus akurasi sebagai berikut:

Tabel 3. 7 Perhitungan Rata-rata *Convusion Matriks* SVM

True Label		*Predicted Label*	
TN	45	*FP*	5
FN	40	*TP*	10

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} = \frac{10 + 45}{10 + 45 + 5 + 10} = \mathbf{54.90\%}$$

b. Model SVM dan SA



Gambar 3. 4 Hasil *Confusion Matrix* Model SVM dengan SA

Untuk menghitung akurasi secara manual dari *confusion matrix* dapat menggunakan rumus akurasi sebagai berikut:

Tabel 3. 8 Perhitungan Rata-rata *Convusion Matriks* SVM dengan SA

True Label		*Predicted Label*	
TN	38	*FP*	13
FN	7	*TP*	43

$$Accuracy = \frac{TP + TN}{TP + FN + TN + FP} = \frac{43 + 38}{43 + 7 + 38 + 13} = \mathbf{88.10\%}$$

3.2 Pembahasan

3.2.1 Fitur yang Mempengaruhi Kinerja BMR dan SVM

Berdasarkan hasil yang terlihat pada Tabel 3.6, maka ditentukan bahwa atribut yang memiliki rangking 1-9 yaitu ZS TB/U, ZS BB/U, ZS BB/TB, Tinggi, Berat, Naik Berat Badan, LiLA, BB/TB, dan BB/U digunakan sebagi atribut dalam pemodelan SVM karena memiliki nilai bobot yang tinggi. Sedangkan pada penelitian yang di lakukan oleh (*Azhima et al.,* 2021) menggunakan algoritma SVM dan *Chi-Square* menghasilkan 6 fitur yakni ZS TB/U, BB/U, ZS BB/U, Tinggi, LiLA dan Berat. Pada penelitian (*Fadellia Azzahra et al*., 2024) menggunakan *Random Forest* dan PSO, fitur yang paling berpengaruh yaitu BB/TB. Penelitian lainnya oleh (*Gina Purnama Insany et al*., 2023) menggunakan metode KNN dan ANN menghasilkan 3 atribut yaitu Berat Badan/Umur, Berat Badan/Tinggi Badan, dan Tinggi Badan/Umur. Sehingga persamaan fitur yang dominan dari penelitian tersebut adalah atribut Berat Badan/Umur (BB/U) dan Berat Badan/Tinggi Badan (BB/TB).

Tabel 3. 9 Perbandingan Fitur

Penelitian	Data	Metode	Jumlah Fitur	Fitur	Persamaan
Penelitian ini	Stunting	SVM+BMR+SA	9	ZS TB/U, ZS BB/U, ZS BB/TB, Tinggi, Berat, Naik Berat Badan, LiLA, BB/TB, dan BB/U	BB/TB
(*Azhima et al.,* 2021)	Stunting	SVM+*Chi-Square*	6	ZS TB/U, BB/U, ZS BB/U, Tinggi, LiLA dan Berat	BB/U
(*Fadellia Azzahra et al*., 2024)	Stunting	*Random Forest* dan PSO	1	BB/TB	BB/TB
(*Gina Purnama Insany et al*., 2023)	Stunting	KNN dan ANN	3	Berat Badan/Umur, Berat Badan/Tinggi Badan, dan Tinggi Badan/Umur.	Berat Badan/Umur, Berat Badan/Tinggi Badan

3.2.2 Performa Algoritma

Perbandingan performa akurasi dengan penelitian *(Fadellia Azzahra et al.*, 2024) menghasilkan akurasi 78.33% menggunakan metode *Random Forest* dan *Cross Validation* tanpa optimasi atupun seleksi fitur. Penelitian (Azhima et al., 2021) menggunakan data stunting SVM dengan Chi-Square dan SMOTE untuk *High Dimension* Data Stunting dengan *Cross Validation* k-fold=10 menghasilkan akurasi 96,6% namun tanpa metode optimasi. Penelitian lainnya dilakukan oleh (*Rahmi et al*., 2022) menggunakan data stunting, metode SVM dan karnel RBF untuk data stunting dengan nilai cost = 10 dan gamma = 5 menghasilkan akurasi 100% namun, saat di terapkan dalam penelitian ini, karnel RBF dengan nilai *cost* = 10 dan *gamma* 5 menghasilkan rata-rata akurasi 54.90% mengalami penurunan. Sedangkan saat penambahan seleksi fitur BMR dan optimasi *Simulated Annealing* akurasinya mengalami peningkatan sebanyak 33.20% sehingga akurasinya menjadi 88.10%. dengan demikian implementasi seleksi fitur menggunakan BMR dan optimasi menggunakan *Simulated Annealing* meningkatkan akurasi klasifikasi atau prediksi dari metode SVM dengan kernel RBF pada data stunting.

Tabel 3. 10 Perbandingan Akurasi

Penelitian	Data	Metode	Akurasi
(Fadellia Azzahra et al., 2024)	Stunting	*Random Forest* dan *Cross Validation*	78.33%
(Azhima et al., 2021)	Stunting	SVM + Chi-Square + SMOTE + *Cross Validation* k-fold=10	96.6%
(*Rahmi et al*., 2022)	Stunting	SVM dan karnel RBF	100%
Penelitian ini	Stunting	SVM dan karnel RBF	54.90%
		SVM+BMR+SA	88.10%

BAB IV

PENUTUP

4.1 Kesimpulan

Berikut kesimpulan penelitian berdasarkan hasil dan rumusan masalah yang di cantumkan:

1. Fitur yang memiliki pengaruh dominan terhadap stunting dari metode seleksi fitur *Boundary Margin Relief* (BMR), yaitu ZS TB/U, ZS BB/U, ZS BB/TB, tinggi badan, berat badan, naik berat badan, LiLA, BB/TB, dan BB/U. Fitur dominannya yaitu Berat Badan/Umur (BB/U) dan Berat Badan/Tinggi Badan (BB/TB).
2. Hasil akurasi SVM karnel RBF dengan metode optimasi *Simulated Annealing* dan seleksi fitur *Boundary Margin Relief* sebesar 54.90% tanpa optimasi dan sebesar 88.10% setelah di optimasi dengan SA, sehingga menunjukkan kenaikan akurasi sebesar 33.20%

4.2 Saran

Berikut beberapa saran berdasarkan hasil penelitian ini:

1. Penelitian selnjutnya disarankan untuk penyesuaian nilai pada parameter C (Cost) dan *gamma* karnel RBF, agar akurasi yang di dapat benar-benar akurat. Karena pada penelitian ini saat menambahkan nilai *gamma* dan nilai C pada karnel RBF akurasi SVM tanpa optimasi SA menurun sangat jauh, sehingga memerlukan penyesuaiaan lebih lanjut. Selain itu untuk penelitian lanjutan, dapat mencoba karnel SVM lainnya seperti *linear, sigmoid* atau *polynominal* untuk membandingkan tingkat kinerja karnel yang lebih baik.
2. Disarankan untuk melakukan penelitian lanjutan dengan mempertimbangkan lebih banyak fitur dan mencoba metode klasifikasi yang berbeda seperti *Random Forest* guna mendapatkan hasil yang lebih akurat dalam mendiagnosis stunting pada anak.

DAFTAR RUJUKAN

Al-Mejibli, I. S., Alwan, J. K., & Abd, D. H. (2020). The effect of gamma value on support vector machine performance with different kernels. *International Journal of Electrical and Computer Engineering*, *10*(5), 5497–5506. https://doi.org/10.11591/IJECE.V10I5.PP5497-5506

Andriyani, S. Y., Lydia, M. S., & Efendi, S. (2023). Optimization of Support Vector Machine Algorithm Using Stunting Data Classification. *Prisma Sains : Jurnal Pengkajian Ilmu Dan Pembelajaran Matematika Dan IPA IKIP Mataram*, *11*(1), 164. https://doi.org/10.33394/j-ps.v11i1.6619

Anggrawan, A., Hairani, H., & Satria, C. (2023). *Improving SVM Classification Performance on Unbalanced Student Graduation Time Data Using SMOTE*. *13*(2). https://doi.org/10.18178/ijiet.2023.13.2.1806

Anggriawan, R., & Nugroho, H. W. (2023). Komparasi Algoritma C4.5 dan Naive Bayes Dalam Prediksi Penderita Penyakit Gagal Jantung. *Jurnal SIMADA (Sistem Informasi Dan Manajemen Basis Data)*, *6*(1), 82–91. https://doi.org/10.30873/simada.v6i1.3425

Arumi, E. R., Sumarno Adi Subrata, & Anisa Rahmawati. (2023). Implementation of Naïve bayes Method for Predictor Prevalence Level for Malnutrition Toddlers in Magelang City. *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, *7*(2), 201–207. https://doi.org/10.29207/resti.v7i2.4438

Azhima, T., Siswa, Y., & Azmi, N. (2021). *Optimasi Algoritma SVM dengan Chi-Square dan SMOTE untuk High Dimension Data Stunting Kota Samarinda*. *xx*(200), 1–10.

Chennuru, V. K., & Timmappareddy, S. R. (2022). Simulated annealing based undersampling (SAUS): a hybrid multi-objective optimization method to tackle class imbalance. *Applied Intelligence*, *52*(2), 2092–2110. https://doi.org/10.1007/s10489-021-02369-4

Fadellia Azzahra, Suarna, N., & Arie Wijaya, Y. (2024). Penerapan Algoritma Random Forest Dan Cross Validation Untuk Prediksi Data Stunting. *Kopertip : Jurnal Ilmiah Manajemen Informatika Dan Komputer*, *8*(1), 1–6. https://doi.org/10.32485/kopertip.v8i1.238

Gina Purnama Insany, Indra Yustiana, & Sri Rahmawati. (2023). Penerapan KNN dan ANN pada klasifikasi status gizi balita berdasarkan indeks antropometri. *Jurnal CoSciTech (Computer Science and Information Technology)*, *4*(2), 385–393. https://doi.org/10.37859/coscitech.v4i2.5079

Huang, P. (2020). Recognition of common non-normal walking actions based on Relief-F feature selection and Relief-Bagging-SVM. *Sensors (Switzerland)*, *20*(5). https://doi.org/10.3390/s20051447

Hussein, H. I., Anwar, S. A., & Ahmad, M. I. (2023). *Imbalanced Data Classification Using SVM Based on Improved Simulated Annealing Featuring Synthetic Data Generation and Reduction*. https://doi.org/10.32604/cmc.2023.036025

Khan, J. R. (2021). Model and variable selection using machine learning methods with applications to childhood stunting in Bangladesh. *Informatics for Health and Social Care*, *46*(4), 425–442. https://doi.org/10.1080/17538157.2021.1904938

Mahareek, E. A., Desuky, A. S., & El-Zhni, H. A. (2021). Simulated annealing for svm parameters optimization in student's performance prediction. *Bulletin of Electrical Engineering and Informatics*, *10*(3), 1211–1219. https://doi.org/10.11591/eei.v10i3.2855

Migoń, P. (2021). Disentangling polygenetic relief of low mountains at the margin of inland glaciation – Upper Nysa Szalona drainage basin, Sudetes, Central Europe. *Catena*, *204*. https://doi.org/10.1016/j.catena.2021.105383

Ministry of Research and Technology of the republic of Indonesia. (2020). *Prioritas riset nasional*. *July*, 1–23.

Noviardianto, G. E., Novel, M., & Legowo, M. B. (2019). Penggunaan Metode Simulated Annealing untuk Optimasi Penempatan Posisi Access Point pada Jaringan WI-FI. *JURNAL Al-AZHAR INDONESIA SERI SAINS DAN TEKNOLOGI*, *5*(1), 10. https://doi.org/10.36722/sst.v5i1.318

Rahman, F. (2020). Implementation of Simulated Annealing-Support Vector Machine on QSAR Study of Fusidic Acid Derivatives as Anti-Malarial Agent. In *6th International Conference on Interactive Digital Media, ICIDM 2020*. https://doi.org/10.1109/ICIDM51048.2020.9339632

Rahmi, I., Susanti, M., Yozza, H., & Wulandari, F. (2022). Classification of Stunting in Children Under Five Years in Padang City Using Support Vector Machine. *BAREKENG: Jurnal Ilmu Matematika Dan Terapan*, *16*(3), 771–778. https://doi.org/10.30598/barekengvol16iss3pp771-778

Raj, D. M. D., & Mohanasundaram, R. (2020). An Efficient Filter-Based Feature Selection Model to Identify Significant Features from High-Dimensional Microarray Data. *Arabian Journal for Science and Engineering*, *45*(4), 2619–2630. https://doi.org/10.1007/s13369-020-04380-2

Rajabi, K. M. (2023). *Penerapan Algoritma K-Nearest Neighbor (KNN) Dengan Fitur Relief-F Dalam Penentuan Status Stunting*. *3*, 3555–3568.

Shao, Y. (2021). Integrated SVM Method with AM-ReliefF Feature Selection for Mechanical Fault Diagnosis of High Voltage Circuit Breakers. *Zhongguo Dianji Gongcheng Xuebao/Proceedings of the Chinese Society of Electrical Engineering*, *41*(8), 2890–2900. https://doi.org/10.13334/j.0258-8013.pcsee.200979

Taghfirul Azhima Yoga Siswa, S. K. M. K. (2017). DATA MINING: MENGUPAS TUNTAS ANALISIS DATA DENGAN METODE KLASIFIKASI HINGGA DEPLOYMENT APLIKASI MENGGUNAKAN PYTHON. In *Jurnal Sains dan Seni ITS* (Vol. 6, Issue 1). http://repositorio.unan.edu.ni/2986/1/5624.pdf%0Ahttp://fiskal.kemenkeu.go.id/ejournal%0Ahttp://dx.doi.org/10.1016/j.cirp.2016.06.001%0Ahttp://dx.doi.org/10.1016/j.powtec.2016.12.055%0Ahttps://doi.org/10.1016/j.ijfatigue.2019.02.006%0Ahttps://doi.org/10.1

Ula, M., Ulva, A. F., Ali, M. A., & Rilasmi, Y. (2022). *APPLICATION OF MACHINE LEARNING IN DETERMINING THE CLASSIFICATION OF CHILDREN ' S NUTRITION WITH DECISION TREE PENERAPAN MACHINE LEARNING DALAM PENENTUAN KLASIFIKASI GIZI*. *3*(5), 1457–1465.

V. Herliansyah, R. Latuconsina, & A. Dinimaharawati. (2021). Prediksi Stunting Pada Balita Dengan Menggunakan Algoritma Klasifikasi Random Forest. *E-Proceeding of Engineering*, *8*(5), 6643–6649.

Wang, H., Wang, P., Deng, S., & Li, H. (2021). *Improved Relief Weight Feature Selection Algorithm Based on Relief and Mutual Information*.

Wang, W. (2023). Hybrid Simulated Annealing Particle Swarm Optimization Support Vector Machine Based Temperature-Pressure Error Compensation Approach for TDLAS Gas Detection. *Combustion Science and Technology*. https://doi.org/10.1080/00102202.2023.2202320

Xdqj, D., Krx, K. L., Lqzhq, E., Dgguhvv, P., Frp, T. T., Dgguhvv, P., Kl, K., & Hgx, V. (2020). *Support Vector Machine Classification Algorithm Based on Relief-F Feature Weig*. 547–553. https://doi.org/10.1109/ICCEA50009.2020.00121

Yun, F., Dong, H., Liang, C., Weimin, T., & Chao, T. (2023). Feature Selection of XLPE Cable Condition Diagnosis Based on PSO-SVM. *Arabian Journal for Science …*. https://doi.org/10.1007/s13369-022-07175-9

Yunus, M., Biddinika, M. K., & Fadlil, A. (2023). Classification of Stunting in Children Using the C4.5 Algorithm. *Jurnal Online Informatika*, *8*(1), 99–106. https://doi.org/10.15575/join.v8i1.1062

LAMPIRAN

Lampiran 1 Surat Minta Data ke DINKES Kota Samarinda

UMKT
Program Studi
Teknik Informatika
Fakultas Sains dan Teknologi
Universitas Muhammadiyah Kalimantan Timur
Berkarakter | Berwawasan | Berkemajuan

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

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

Kampus 1 : Jl. Ir. H. Juanda, No.15, Samarinda
Kampus 2 : Jl. Pelita, Pesona Mahakam, Samarinda

Lampiran 2 Sampel Dataset Stunting Kota Samarinda

No	Nama	JK	BB Lahir	TB Lahir	Prov	Kab/Kota	Kec	Pukesmas	Desa/Kel	Posyandu	RT	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 A
1	DIMAS ADITYA	L	3.5	49	KALIMANTAN TIMUR	SAMARINDA	SUNGAI PINANG	REMAJA	TEMINDUNG PERMAI	PULAU INDAH		0 Tahun - 11 Bulan - 11 Hari	2023-01-02	9.1	74.5	0	Berat Badan Normal	-0.39	Normal	-0.21	Gizi Baik	-0.39	O	-	
2	SITI AISYAH	P	3	48	KALIMANTAN TIMUR	SAMARINDA	SUNGAI PINANG	REMAJA	TEMINDUNG PERMAI	PULAU INDAH	34	4 Tahun - 0 Bulan - 16 Hari	2023-01-02	12	94	0	Kurang	-2.25	Pendek	-2.09	Gizi Baik	-1.46	O	0.84	
3	M AL FATIH	L	2.8	49	KALIMANTAN TIMUR	SAMARINDA	SUNGAI PINANG	REMAJA	TEMINDUNG PERMAI	PULAU INDAH		0 Tahun - 7 Bulan - 26 Hari	2023-01-02	8.1	69	0	Berat Badan Normal	-0.53	Normal	-0.65	Gizi Baik	-0.14	O	-	
4	ALMAHIRA AKIRA AKBAR	P	3.98	50647201	KALIMANTAN TIMUR	SAMARINDA	SUNGAI PINANG	REMAJA	TEMINDUNG PERMAI	PULAU INDAH		0 Tahun - 4 Bulan - 7 Hari	2023-01-02	6.3	63.5	0	Berat Badan Normal	-0.31	Normal	0.42	Gizi Baik	-0.74	O	-	
5	GIUNIA QAMELA CALANDRA	P	3.5	50	KALIMANTAN TIMUR	SAMARINDA	SUNGAI PINANG	REMAJA	TEMINDUNG PERMAI	PULAU INDAH	34	0 Tahun - 10 Bulan - 10 Hari	2023-01-02	10.6	72.5	0	Risiko Lebih	1.71	Normal	0.23	Gizi Lebih	2.14	O	-	
6	MUHAMMAD GAVRIEL IZILAN NUGRAHA	L	3.5	49.5	KALIMANTAN TIMUR	SAMARINDA	SUNGAI PINANG	REMAJA	TEMINDUNG PERMAI	PULAU INDAH	34	2 Tahun - 11 Bulan - 16 Hari	2023-01-02	17	95	0	Risiko Lebih	1.44	Normal	-0.21	Gizi Lebih	2.28	O	-	
7	HADRAL HUSEIN	L	3.2	49.5	KALIMANTAN TIMUR	SAMARINDA	SUNGAI PINANG	REMAJA	TEMINDUNG PERMAI	PULAU INDAH	34	3 Tahun - 0 Bulan - 29 Hari	2023-01-02	15.8	96	0	Berat Badan Normal	0.7	Normal	-0.19	Risiko Gizi Lebih	1.21	O	-	
8	ASYIFA FATIAYA M	P	3.2	49	KALIMANTAN TIMUR	SAMARINDA	SUNGAI PINANG	REMAJA	TEMINDUNG PERMAI	PULAU INDAH		2 Tahun - 3 Bulan - 29 Hari	2023-01-02	15	88	0	Risiko Lebih	1.54	Normal	-0.31	Gizi Lebih	2.28	O	-	
…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…	…
150458	HADZKYA NUMA ADISYA	P	2.9	48	KALIMANTAN TIMUR	SAMARINDA	SAMARINDA UTARA	LEMPAKE	LEMPAKE	MATAHARI		1 Tahun - 11 Bulan - 26 Hari	2023-12-05	8.4	-	0	Kurang	-2.62		-		-	O	-	
150459	ADIVA FARHANAH AL AMIN	P	2.9	50	KALIMANTAN TIMUR	SAMARINDA	SAMARINDA UTARA	LEMPAKE	LEMPAKE	MATAHARI		0 Tahun - 0 Bulan - 0 Hari	2023-12-15	2.9	50		Berat Badan Normal	-0.75	Normal	0.46	Gizi Baik	-1.64	-	-	
150460	ALFARIZKI HAMIZAN	L	3	50	KALIMANTAN TIMUR	SAMARINDA	SAMARINDA UTARA	LEMPAKE	LEMPAKE	MATAHARI		0 Tahun - 0 Bulan - 0 Hari	2023-12-28	3	50		Berat Badan Normal	-0.73	Normal	0.06	Gizi Baik	-1.19	-	-	
150461	ADNAN IRAGUSTI	L	3	50	KALIMANTAN TIMUR	SAMARINDA	SAMARINDA UTARA	LEMPAKE	LEMPAKE	SRI REJEKI		0 Tahun - 0 Bulan - 0 Hari	2023-12-30	3	50		Berat Badan Normal	-0.73	Normal	0.06	Gizi Baik	-1.19	-	-	
150462	SIENA AL RAISHA RAMADHANI	P	3	52	KALIMANTAN TIMUR	SAMARINDA	SUNGAI PINANG	REMAJA	BANDARA	LESTARI	24	3 Tahun - 7 Bulan - 19 Hari	2023-12-13	13	95.5	0	Berat Badan Normal	-1.22	Normal	-1.11	Gizi Baik	-0.85	O	-	
150463	AFIZAH KHAIRINA	P	2.5	45	KALIMANTAN TIMUR	SAMARINDA	SAMARINDA SEBERANG	MANGKUPALAS	TENUN SAMARINDA	BALO NEGARA	12	0 Tahun - 0 Bulan - 0 Hari	2023-12-09	2.5	45		Kurang	-2.03	Pendek	-2.63	Gizi Baik	-0.35	-	-	
150464	M ARSYA KHOLIF	P	3	49	KALIMANTAN TIMUR	SAMARINDA	SAMARINDA ULU	JUANDA	AIR HITAM	MEKAR SEJAHTERA	32	0 Tahun - 0 Bulan - 0 Hari	2023-12-19	3	49		Berat Badan Normal	-1.56	Normal	-1.3	Gizi Baik	-1.04	-	-	
150465	MUHAMMAD IQBAL	L	2.9	49	KALIMANTAN TIMUR	SAMARINDA	SAMARINDA ILIR	SIDOMULYO	SUNGAI DAMA	RAMANIA	0	0 Tahun - 0 Bulan - 0 Hari	2023-12-29	2.9	49		Kurang	-2.97	Pendek	-2.48	Gizi Baik	-1.37	-	-	

Lampiran 3 Program *Python*

Library Python

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import cross_val_predict
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import cross_val_score
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OrdinalEncoder
from sklearn.preprocessing import MinMaxScaler
```

Membaca informasi data

```
import pandas as pd
data = pd.read_csv('stunting.csv')
data.head()
```

	Nama	JK	Berat	Tinggi	LiLA	BB/U	ZS BB/U	TB/U	ZS TB/U	BB/TB	ZS BB/TB	Naik Berat Badan	Jml Vit A	Tanggal Pengukuran
0	DIMAS ADITYA	L	9.01	NaN	0.0	Berat Badan Normal	-0.390000	Normal	-0.21	Gizi Baik	-0.39	O	NaN	2023-01-02
1	SITI AISYAH	P	12.00	94.0	0.0	Kurang	-2.250000	Pendek	-2.09	Gizi Baik	-1.46	O	NaN	2023-01-02
2	M AL FATIH	L	8.01	69.0	0.0	Berat Badan Normal	-0.530000	Normal	-0.65	Gizi Baik	-0.14	O	NaN	2023-01-02
3	ALMAHIRA AKIRA AKBAR	P	6.03	NaN	0.0	Berat Badan Normal	-0.310000	Normal	0.42	Gizi Baik	-0.74	O	NaN	2023-01-02
4	GIUNIA QAMELA CALANDRA	P	10.06	NaN	0.0	Risiko Lebih	0.090972	Normal	0.23	Gizi Lebih	0.09	O	NaN	2023-01-02

Melihat data dengan *spit* 5 *record*

```
data
```

	Nama	JK	Berat	Tinggi	LiLA	BB/U	ZS BB/U	TB/U	ZS TB/U	BB/TB	ZS BB/TB	Naik Berat Badan	Jml Vit A	Tanggal Pengukuran
0	DIMAS ADITYA	L	9.01	NaN	0.0	Berat Badan Normal	-0.390000	Normal	-0.21	Gizi Baik	-0.39	O	NaN	2023-01-02
1	SITI AISYAH	P	12.00	94.0	0.0	Kurang	-2.250000	Pendek	-2.09	Gizi Baik	-1.46	O	NaN	2023-01-02
2	M AL FATIH	L	8.01	69.0	0.0	Berat Badan Normal	-0.530000	Normal	-0.65	Gizi Baik	-0.14	O	NaN	2023-01-02
3	ALMAHIRA AKIRA AKBAR	P	6.03	NaN	0.0	Berat Badan Normal	-0.310000	Normal	0.42	Gizi Baik	-0.74	O	NaN	2023-01-02
4	GIUNIA QAMELA CALANDRA	P	10.06	NaN	0.0	Risiko Lebih	0.090972	Normal	0.23	Gizi Lebih	0.09	O	NaN	2023-01-02
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
150460	ADNAN IRAGUSTI	L	3.00	50.0	NaN	Berat Badan Normal	-0.730000	Normal	0.06	Gizi Baik	-1.19	-	NaN	2023-12-30
150461	SIENA AL RAISHA RAMADHANI	P	13.00	NaN	0.0	Berat Badan Normal	-1.220000	Normal	-1.11	Gizi Baik	-0.85	O	NaN	2023-12-13
150462	AFIZAH KHAIRINA	P	2.05	45.0	NaN	Kurang	-2.030000	Pendek	-2.63	Gizi Baik	-0.35	-	NaN	2023-12-09
150463	M ARSYA KHOLIF	P	3.00	49.0	NaN	Berat Badan Normal	-1.560000	Normal	-1.30	Gizi Baik	-1.04	-	NaN	2023-12-19
150464	MUHAMMAD IQBAL	L	2.09	49.0	NaN	Kurang	-2.970000	Pendek	-2.48	Gizi Baik	-1.37	-	NaN	2023-12-29

150465 rows × 14 columns

Membaca informasi *type* data

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150465 entries, 0 to 150464
Data columns (total 14 columns):
 #   Column              Non-Null Count   Dtype
---  ------              --------------   -----
 0   Nama                150459 non-null  object
 1   JK                  150465 non-null  object
 2   Berat               149907 non-null  float64
 3   Tinggi              99243 non-null   float64
 4   LiLA                121390 non-null  float64
 5   BB/U                150465 non-null  object
 6   ZS BB/U             150306 non-null  float64
 7   TB/U                137594 non-null  object
 8   ZS TB/U             138201 non-null  float64
 9   BB/TB               137653 non-null  object
 10  ZS BB/TB            138265 non-null  float64
 11  Naik Berat Badan    150408 non-null  object
 12  Jml Vit A           55131 non-null   float64
 13  Tanggal Pengukuran  150465 non-null  object
dtypes: float64(7), object(7)
memory usage: 16.1+ MB
```

Tahap preprocessing data menghapus data duplikat

```
import pandas as pd

# Membaca data dari file CSV (ganti dengan lokasi file yang sesuai)
data = pd.read_csv('stunting.csv')

# 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
data.to_csv('dataset_remove_duplikat.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: 150465
Total data setelah penghapusan duplikat: 34199
```

```
dataset = pd.read_csv('dataset_remove_duplikat.csv')
dataset.info()
dataset = data.drop('Jml Vit A', axis=1)

# Menyimpan hasilnya ke file CSV baru
dataset.to_csv('dataset_tanpa_vitA.csv', index=False)
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 34199 entries, 0 to 34198
Data columns (total 14 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Nama                34198 non-null  object
 1   JK                  34199 non-null  object
 2   Berat               34043 non-null  float64
 3   Tinggi              22314 non-null  float64
 4   LiLA                28576 non-null  float64
 5   BB/U                34199 non-null  object
 6   ZS BB/U             34160 non-null  float64
 7   TB/U                31559 non-null  object
 8   ZS TB/U             31737 non-null  float64
 9   BB/TB               31572 non-null  object
 10  ZS BB/TB            31746 non-null  float64
 11  Naik Berat Badan    34199 non-null  object
 12  Jml Vit A           12659 non-null  float64
 13  Tanggal Pengukuran  34199 non-null  object
dtypes: float64(7), object(7)
memory usage: 3.7+ MB
```

Membaca dataset_tanpa_vita.csv

```
dataset = pd.read_csv('dataset_tanpa_vitA.csv')
dataset.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 34199 entries, 0 to 34198
Data columns (total 13 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   Nama                34198 non-null  object
 1   JK                  34199 non-null  object
 2   Berat               34043 non-null  float64
 3   Tinggi              22314 non-null  float64
 4   LiLA                28576 non-null  float64
 5   BB/U                34199 non-null  object
 6   ZS BB/U             34160 non-null  float64
 7   TB/U                31559 non-null  object
 8   ZS TB/U             31737 non-null  float64
 9   BB/TB               31572 non-null  object
 10  ZS BB/TB            31746 non-null  float64
 11  Naik Berat Badan    34199 non-null  object
 12  Tanggal Pengukuran  34199 non-null  object
dtypes: float64(6), object(7)
memory usage: 3.4+ MB
```

```
dataset
```

	Nama	JK	Berat	Tinggi	LiLA	BB/U	ZS BB/U	TB/U	ZS TB/U	BB/TB	ZS BB/TB	Naik Berat Badan	Tanggal Pengukuran
0	A ALVIN	L	18.06	NaN	18.0	Risiko Lebih	1.190000	Normal	0.41	Risiko Gizi Lebih	0.07	N	2023-12-16
1	A FADLAN	L	11.06	83.0	0.0	Berat Badan Normal	-0.080000	Normal	-0.97	Gizi Baik	0.04	-	2023-07-10
2	A FARIS WICAKSONO	L	9.07	78.0	16.0	Berat Badan Normal	-1.750000	Pendek	-2.84	Gizi Baik	-0.47	O	2023-10-06
3	A FATHAN	L	15.00	107.0	17.0	Berat Badan Normal	-1.080000	Normal	0.14	Gizi Baik	-1.83	T	2023-10-23
4	A FAUJAN	L	14.00	100.0	0.0	Berat Badan Normal	-0.850000	Normal	-0.16	Gizi Baik	-1.14	O	2023-02-07
...	...	...	...	...	...	...	...	...	...	...	...	...	...
34194	zihan sonia bella	P	15.07	NaN	0.0	Risiko Lebih	0.095833	Normal	1.38	Risiko Gizi Lebih	0.07	O	2023-02-06
34195	zubair	L	17.03	107.0	0.0	Berat Badan Normal	-0.140000	Normal	-0.13	Gizi Baik	-0.13	O	2023-02-20
34196	zulaikha mina chandra	P	14.00	104.0	NaN	Berat Badan Normal	-1.430000	Normal	-0.42	Gizi Baik	-1.81	N	2023-02-05
34197	zulkifli abdi	L	15.06	102.0	0.0	Berat Badan Normal	-0.590000	Normal	-0.69	Gizi Baik	-0.25	N	2023-10-03
34198	NaN	L	8.00	71.0	NaN	Berat Badan Normal	-1.570000	Normal	-1.40	Gizi Baik	-1.18	T	2023-12-18

34199 rows × 13 columns

Menghitung jumlah kategori dalam atribut TB/U

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

```
TB/U
Normal          26941
Pendek           3350
Sangat Pendek    1125
Tinggi            143
Name: count, dtype: int64
```

Penghapusan data *mising*

```
import pandas as pd

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

# Mengganti nilai '-' pada kolom 'Naik Berat Badan' dengan 'A'
data['Naik Berat Badan'].replace('-', 'A', inplace=True)
data
```

	Nama	JK	Berat	Tinggi	LiLA	BB/U	ZS BB/U	TB/U	ZS TB/U	BB/TB	ZS BB/TB	Naik Berat Badan	Tanggal Pengukuran
0	A ALVIN	L	18.06	NaN	18.0	Risiko Lebih	1.190000	Normal	0.41	Risiko Gizi Lebih	0.07	N	2023-12-16
1	A FADLAN	L	11.06	83.0	0.0	Berat Badan Normal	-0.080000	Normal	-0.97	Gizi Baik	0.04	A	2023-07-10
2	A FARIS WICAKSONO	L	9.07	78.0	16.0	Berat Badan Normal	-1.750000	Pendek	-2.84	Gizi Baik	-0.47	O	2023-10-06
3	A FATHAN	L	15.00	107.0	17.0	Berat Badan Normal	-1.080000	Normal	0.14	Gizi Baik	-1.83	T	2023-10-23
4	A FAUJAN	L	14.00	100.0	0.0	Berat Badan Normal	-0.850000	Normal	-0.16	Gizi Baik	-1.14	O	2023-02-07
...	...	...	...	...	...	...	...	...	...	...	...	...	...
34194	zihan sonia bella	P	15.07	NaN	0.0	Risiko Lebih	0.095833	Normal	1.38	Risiko Gizi Lebih	0.07	O	2023-02-06
34195	zubair	L	17.03	107.0	0.0	Berat Badan Normal	-0.140000	Normal	-0.13	Gizi Baik	-0.13	O	2023-02-20
34196	zulaikha mina chandra	P	14.00	104.0	NaN	Berat Badan Normal	-1.430000	Normal	-0.42	Gizi Baik	-1.81	N	2023-02-05
34197	zulkifli abdi	L	15.06	102.0	0.0	Berat Badan Normal	-0.590000	Normal	-0.69	Gizi Baik	-0.25	N	2023-10-03
34198	NaN	L	8.00	71.0	NaN	Berat Badan Normal	-1.570000	Normal	-1.40	Gizi Baik	-1.18	T	2023-12-18

34199 rows × 13 columns

Menghitung jumlah data sebelum dan sesudah penghapusan

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

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

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

# Menyimpan hasilnya ke file CSV baru
data_tanpa_missing.to_csv('nama_file_tanpa_missing.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: 34199
Jumlah data setelah penghapusan: 18396
```

Membaca file data tanpa *mising*

```
data_tanpa_missing
```

	Nama	JK	Berat	Tinggi	LiLA	BB/U	ZS BB/U	TB/U	ZS TB/U	BB/TB	ZS BB/TB	Naik Berat Badan	Tanggal Pengukuran
1	A FADLAN	L	11.06	83.0	0.0	Berat Badan Normal	-0.08	Normal	-0.97	Gizi Baik	0.04	A	2023-07-10
2	A FARIS WICAKSONO	L	9.07	78.0	16.0	Berat Badan Normal	-1.75	Pendek	-2.84	Gizi Baik	-0.47	O	2023-10-06
3	A FATHAN	L	15.00	107.0	17.0	Berat Badan Normal	-1.08	Normal	0.14	Gizi Baik	-1.83	T	2023-10-23
4	A FAUJAN	L	14.00	100.0	0.0	Berat Badan Normal	-0.85	Normal	-0.16	Gizi Baik	-1.14	O	2023-02-07
5	A GHAFIQI	L	9.03	78.0	0.0	Berat Badan Normal	-1.33	Normal	-1.31	Gizi Baik	-0.99	N	2023-12-21
...	...	...	...	...	...	...	...	...	...	...	...	...	...
34186	yumna	P	13.07	100.0	14.0	Berat Badan Normal	-1.07	Normal	-0.45	Gizi Baik	-1.18	O	2023-10-14
34188	zahratul	P	12.08	95.0	16.0	Kurang	-2.36	Pendek	-2.78	Gizi Baik	-0.93	O	2023-05-13
34191	ziaan aqilla karno	L	14.05	102.0	0.0	Berat Badan Normal	-1.71	Normal	-1.62	Gizi Baik	-1.15	T	2023-10-21
34195	zubair	L	17.03	107.0	0.0	Berat Badan Normal	-0.14	Normal	-0.13	Gizi Baik	-0.13	O	2023-02-20
34197	zulkifli abdi	L	15.06	102.0	0.0	Berat Badan Normal	-0.59	Normal	-0.69	Gizi Baik	-0.25	N	2023-10-03

18396 rows × 13 columns

Membaca nama_file_tanpa_missing.csv

```
bacadata = pd.read_csv('nama_file_tanpa_missing.csv')
bacadata
```

	Nama	JK	Berat	Tinggi	LiLA	BB/U	ZS BB/U	TB/U	ZS TB/U	BB/TB	ZS BB/TB	Naik Berat Badan	Tanggal Pengukuran
0	A FADLAN	L	11.06	83.0	0.0	Berat Badan Normal	-0.08	Normal	-0.97	Gizi Baik	0.04	A	2023-07-10
1	A FARIS WICAKSONO	L	9.07	78.0	16.0	Berat Badan Normal	-1.75	Pendek	-2.84	Gizi Baik	-0.47	O	2023-10-06
2	A FATHAN	L	15.00	107.0	17.0	Berat Badan Normal	-1.08	Normal	0.14	Gizi Baik	-1.83	T	2023-10-23
3	A FAUJAN	L	14.00	100.0	0.0	Berat Badan Normal	-0.85	Normal	-0.16	Gizi Baik	-1.14	O	2023-02-07
4	A GHAFIQI	L	9.03	78.0	0.0	Berat Badan Normal	-1.33	Normal	-1.31	Gizi Baik	-0.99	N	2023-12-21
...	...	...	...	...	...	...	...	...	...	...	...	...	...
18391	yumna	P	13.07	100.0	14.0	Berat Badan Normal	-1.07	Normal	-0.45	Gizi Baik	-1.18	O	2023-10-14
18392	zahratul	P	12.08	95.0	16.0	Kurang	-2.36	Pendek	-2.78	Gizi Baik	-0.93	O	2023-05-13
18393	ziaan aqilla karno	L	14.05	102.0	0.0	Berat Badan Normal	-1.71	Normal	-1.62	Gizi Baik	-1.15	T	2023-10-21
18394	zubair	L	17.03	107.0	0.0	Berat Badan Normal	-0.14	Normal	-0.13	Gizi Baik	-0.13	O	2023-02-20
18395	zulkifli abdi	L	15.06	102.0	0.0	Berat Badan Normal	-0.59	Normal	-0.69	Gizi Baik	-0.25	N	2023-10-03

18396 rows × 13 columns

Menghitung jumlah kategori atribut TB/U

```
data_tanpa_missing['TB/U'].value_counts()
```

```
TB/U
Normal          15951
Pendek           1739
Sangat Pendek     611
Tinggi             95
Name: count, dtype: int64
```

```
# 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
data_tanpa_missing['TB/U'] = data_tanpa_missing['TB/U'].replace(penggantian)

# Menyimpan hasilnya ke file CSV baru
data_tanpa_missing.to_csv('dataset_stunting_samarinda_modified.csv', index=False)
```

```
<ipython-input-118-9f19285c4c44>:10: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-
  data_tanpa_missing['TB/U'] = data_tanpa_missing['TB/U'].replace(penggantian)
```

```
import pandas as pd
import numpy as np
stunting = pd.read_csv('dataset_stunting_samarinda_modified.csv')
stunting = stunting.rename(columns={'TB/U': 'Kelas'})
stunting = stunting.drop('Nama', axis=1)
stunting = stunting.drop('Tanggal Pengukuran', axis=1)
stunting
```

	JK	Berat	Tinggi	LiLA	BB/U	ZS BB/U	Kelas	ZS TB/U	BB/TB	ZS BB/TB	Naik Berat Badan
0	L	11.06	83.0	0.0	Berat Badan Normal	-0.08	Tidak Stunting	-0.97	Gizi Baik	0.04	A
1	L	9.07	78.0	16.0	Berat Badan Normal	-1.75	Stunting	-2.84	Gizi Baik	-0.47	O
2	L	15.00	107.0	17.0	Berat Badan Normal	-1.08	Tidak Stunting	0.14	Gizi Baik	-1.83	T
3	L	14.00	100.0	0.0	Berat Badan Normal	-0.85	Tidak Stunting	-0.16	Gizi Baik	-1.14	O
4	L	9.03	78.0	0.0	Berat Badan Normal	-1.33	Tidak Stunting	-1.31	Gizi Baik	-0.99	N
...	...	...	...	...	...	...	...	...	...	...	...
18391	P	13.07	100.0	14.0	Berat Badan Normal	-1.07	Tidak Stunting	-0.45	Gizi Baik	-1.18	O
18392	P	12.08	95.0	16.0	Kurang	-2.36	Stunting	-2.78	Gizi Baik	-0.93	O
18393	L	14.05	102.0	0.0	Berat Badan Normal	-1.71	Tidak Stunting	-1.62	Gizi Baik	-1.15	T
18394	L	17.03	107.0	0.0	Berat Badan Normal	-0.14	Tidak Stunting	-0.13	Gizi Baik	-0.13	O
18395	L	15.06	102.0	0.0	Berat Badan Normal	-0.59	Tidak Stunting	-0.69	Gizi Baik	-0.25	N

18396 rows × 11 columns

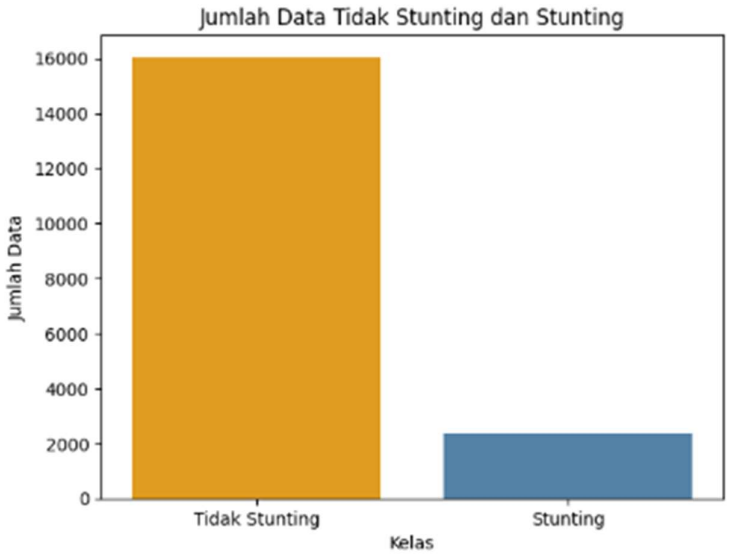
Menghiting kategori atribut Kelas

```
stunting['Kelas'].value_counts()
```

```
Kelas
Tidak Stunting    16046
Stunting           2350
Name: count, dtype: int64
```

```
class_data = stunting['Kelas'].value_counts()

print('Jumlah data kelas Stunting:', class_data[1])
print('Jumlah data kelas Tidak Stunting:', class_data[0])

# Create a bar plot
sns.barplot(x=class_data.index, y=class_data.values, palette=['orange', 'steelblue'])
plt.title("Jumlah Data Tidak Stunting dan Stunting")
plt.xlabel("Kelas")
plt.ylabel("Jumlah Data")
plt.show()
```

```
Jumlah data kelas Stunting: 2350
Jumlah data kelas Tidak Stunting: 16046
<ipython-input-50-965dae2bfd47>:7: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue`

  sns.barplot(x=class_data.index, y=class_data.values, palette=['orange', 'steelblue'])
```




```
# Proses Transformasi Data
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import OrdinalEncoder
ordinal = OrdinalEncoder()
labelencoder = LabelEncoder()

df_transform = stunting[['JK','Berat','Tinggi','LiLA','BB/U','ZS BB/U',
                         'ZS TB/U','Naik Berat Badan','BB/TB', 'ZS BB/TB','Kelas']]

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'])

df = stunting[['JK','Berat','Tinggi','LiLA','BB/U','ZS BB/U','ZS TB/U','Naik Berat Badan','BB/TB','ZS BB/TB','Kelas']]
```

Hasil data yang belum di transformasi

```
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	11.06	83.0	0.0	Berat Badan Normal	-0.08	-0.97	A	Gizi Baik	0.04	Tidak Stunting
1	L	9.07	78.0	16.0	Berat Badan Normal	-1.75	-2.84	O	Gizi Baik	-0.47	Stunting
2	L	15.00	107.0	17.0	Berat Badan Normal	-1.08	0.14	T	Gizi Baik	-1.83	Tidak Stunting
3	L	14.00	100.0	0.0	Berat Badan Normal	-0.85	-0.16	O	Gizi Baik	-1.14	Tidak Stunting
4	L	9.03	78.0	0.0	Berat Badan Normal	-1.33	-1.31	N	Gizi Baik	-0.99	Tidak Stunting
...	...	...	...	...	...	...	...	...	...	...	...
18391	P	13.07	100.0	14.0	Berat Badan Normal	-1.07	-0.45	O	Gizi Baik	-1.18	Tidak Stunting
18392	P	12.08	95.0	16.0	Kurang	-2.36	-2.78	O	Gizi Baik	-0.93	Stunting
18393	L	14.05	102.0	0.0	Berat Badan Normal	-1.71	-1.62	T	Gizi Baik	-1.15	Tidak Stunting
18394	L	17.03	107.0	0.0	Berat Badan Normal	-0.14	-0.13	O	Gizi Baik	-0.13	Tidak Stunting
18395	L	15.06	102.0	0.0	Berat Badan Normal	-0.59	-0.69	N	Gizi Baik	-0.25	Tidak Stunting

18396 rows × 11 columns

Hasil data yang sudah di transformasi

```
df
```

	JK	Berat	Tinggi	LiLA	BB/U	ZS BB/U	ZS TB/U	Naik Berat Badan	BB/TB	ZS BB/TB	Kelas
0	0	11.06	83.0	0.0	0	-0.08	-0.97	0	0	0.04	Tidak Stunting
1	0	9.07	78.0	16.0	0	-1.75	-2.84	11	0	-0.47	Stunting
2	0	15.00	107.0	17.0	0	-1.08	0.14	12	0	-1.83	Tidak Stunting
3	0	14.00	100.0	0.0	0	-0.85	-0.16	11	0	-1.14	Tidak Stunting
4	0	9.03	78.0	0.0	0	-1.33	-1.31	10	0	-0.99	Tidak Stunting
...	...	...	...	...	...	...	...	...	...	...	...
18391	1	13.07	100.0	14.0	0	-1.07	-0.45	11	0	-1.18	Tidak Stunting
18392	1	12.08	95.0	16.0	1	-2.36	-2.78	11	0	-0.93	Stunting
18393	0	14.05	102.0	0.0	0	-1.71	-1.62	12	0	-1.15	Tidak Stunting
18394	0	17.03	107.0	0.0	0	-0.14	-0.13	11	0	-0.13	Tidak Stunting
18395	0	15.06	102.0	0.0	0	-0.59	-0.69	10	0	-0.25	Tidak Stunting

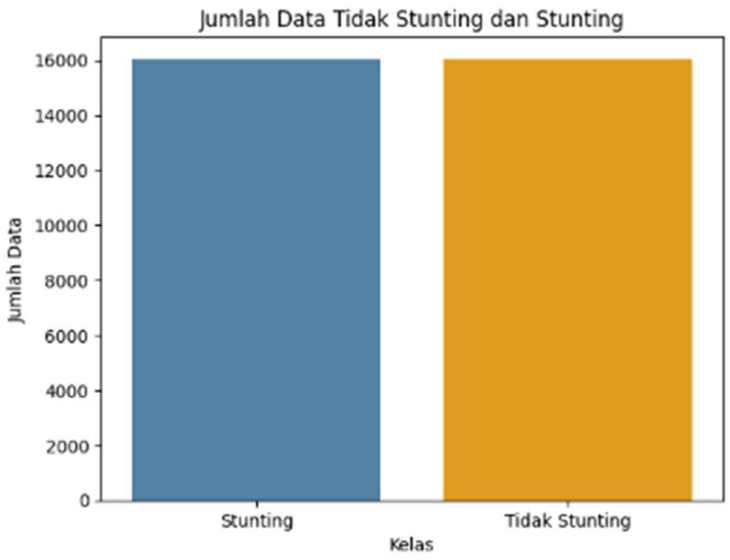
18396 rows × 11 columns

Penanganan ketidakseimbangan kelas

```
X1 = df.drop(['Kelas'],axis=1)
Y1 = df['Kelas']
from imblearn.over_sampling import SMOTE
sm = SMOTE(random_state=42, sampling_strategy=1)
X_imb, y_imb = sm.fit_resample(X1, Y1)
```

```
class_data = y_imb.value_counts()

print('Jumlah data kelas stunting:', class_data[1])
print('Jumlah data kelas tidak stunting:', class_data[0])

# Create a bar plot
sns.barplot(x=class_data[1::-1].index, y=class_data[1::-1].values, palette=['steelblue', 'orange'])
plt.title("Jumlah Data Tidak Stunting dan Stunting")
plt.xlabel("Kelas")
plt.ylabel("Jumlah Data")
plt.show()
```

```
Jumlah data kelas stunting: 16046
Jumlah data kelas tidak stunting: 16046
<ipython-input-56-19277e960ace>:7: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue`

  sns.barplot(x=class_data[1::-1].index, y=class_data[1::-1].values, palette=['steelblue', 'orange'])
```



```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()
X_normalized = scaler.fit_transform(X1)
```

```
X_normalized
```

```
array([[0.        , 0.35393754, 0.50666667, ..., 0.        , 0.        ,
        0.50333333],
       [0.        , 0.28946089, 0.44      , ..., 0.91666667, 0.        ,
        0.46083333],
       [0.        , 0.48159482, 0.82666667, ..., 1.        , 0.        ,
        0.3475    ],
       ...,
       [0.        , 0.45081451, 0.76      , ..., 1.        , 0.        ,
        0.40416667],
       [0.        , 0.54736747, 0.82666667, ..., 0.91666667, 0.        ,
        0.48916667],
       [0.        , 0.48353884, 0.76      , ..., 0.83333333, 0.        ,
        0.47916667]])
```

```
X1
```

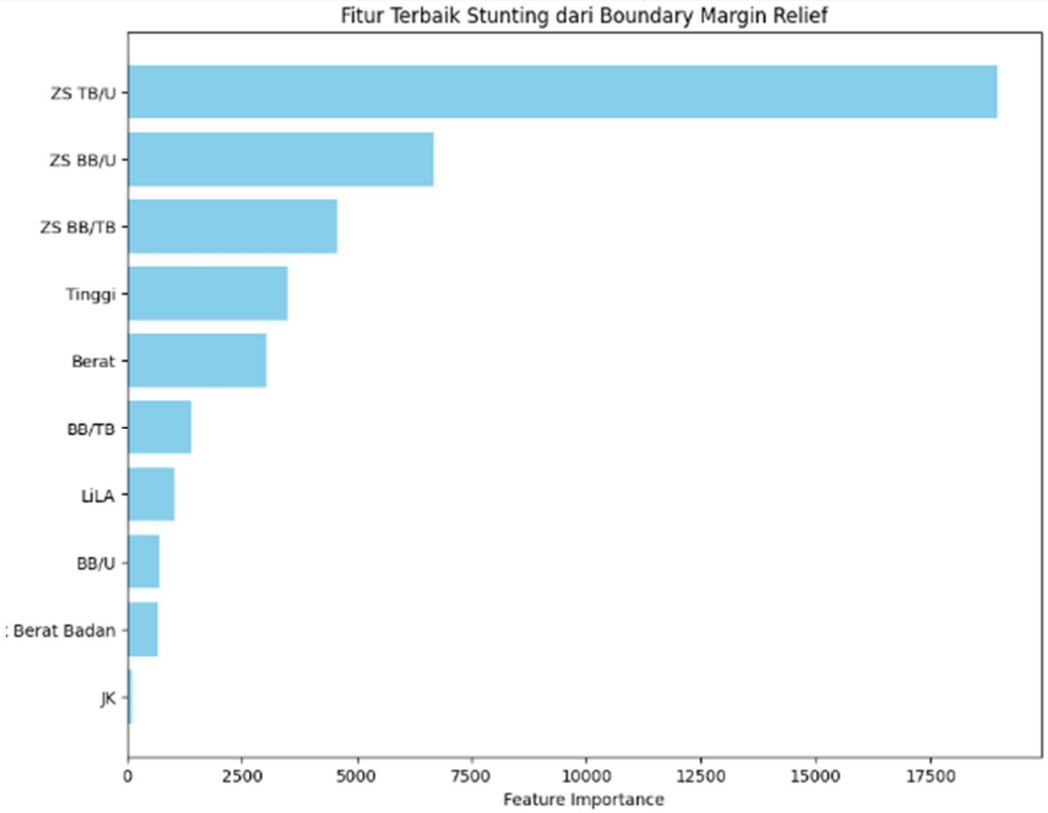
	JK	Berat	Tinggi	LiLA	BB/U	ZS BB/U	ZS TB/U	Naik Berat Badan	BB/TB	ZS BB/TB
0	0	11.06	83.0	0.0	0	-0.08	-0.97	9	0	0.04
1	0	9.07	78.0	16.0	0	-1.75	-2.84	11	0	-0.47
2	0	15.00	107.0	17.0	0	-1.08	0.14	12	0	-1.83
3	0	14.00	100.0	0.0	0	-0.85	-0.16	11	0	-1.14
4	0	9.03	78.0	0.0	0	-1.33	-1.31	10	0	-0.99
...	...	...	...	...	...	...	...	...	...	...
18391	1	13.07	100.0	14.0	0	-1.07	-0.45	11	0	-1.18
18392	1	12.08	95.0	16.0	1	-2.36	-2.78	11	0	-0.93
18393	0	14.05	102.0	0.0	0	-1.71	-1.62	12	0	-1.15
18394	0	17.03	107.0	0.0	0	-0.14	-0.13	11	0	-0.13
18395	0	15.06	102.0	0.0	0	-0.59	-0.69	10	0	-0.25

18396 rows × 10 columns

Seleksi fitur dengan BMR

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.preprocessing import StandardScaler

X_normalized = df.drop(['Kelas'],axis=1)

X = X_normalized
y = df['Kelas']

# Tangani nilai yang hilang jika ada
X.fillna(X.mean(), inplace=True)

# Standarisasi data
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

def boundary_margin_relief(X, y, feature_names):
    weights = np.zeros(X.shape[1])
    for i in range(len(X)):
        # Menghitung jarak ke semua sampel lain
        distances = np.linalg.norm(X - X[i], axis=1)
        # Mengabaikan jarak ke dirinya sendiri dengan mengatur ke nilai besar
        distances[i] = np.inf
        # Mencari tetangga terdekat dalam kelas yang sama (hit) dan kelas berbeda (miss)
        same_class_indices = np.where(y == y[i])[0]
        diff_class_indices = np.where(y != y[i])[0]
        nearest_hit = same_class_indices[np.argmin(distances[same_class_indices])]
        nearest_miss = diff_class_indices[np.argmin(distances[diff_class_indices])]
        weights += np.abs(X[i] - X[nearest_miss]) - np.abs(X[i] - X[nearest_hit])
    return weights
```

```python
# Menggunakan fungsi yang telah dimodifikasi
feature_names = X.columns.tolist()
weights = boundary_margin_relief(X_scaled, y, feature_names)

# Mengurutkan fitur berdasarkan bobot
sorted_indices = np.argsort(weights)[::-1]
sorted_weights = weights[sorted_indices]
sorted_feature_names = [feature_names[i] for i in sorted_indices]

# Visualisasi
n_features_to_display = len(sorted_feature_names)
plt.figure(figsize=(10, 8))
plt.barh(sorted_feature_names[:n_features_to_display], sorted_weights[:n_features_to_display], color='skyblue')
plt.xlabel('Feature Importance')
plt.title('Fitur Terbaik Stunting dari Boundary Margin Relief')
plt.gca().invert_yaxis()
plt.show()

# Menampilkan fitur terbaik beserta bobotnya
for feature, weight in zip(sorted_feature_names[:n_features_to_display], sorted_weights[:n_features_to_display]):
    print(f"Feature: {feature}, Bobot: {weight}")
```




```
Feature: ZS TB/U, Bobot: 18964.710443529853
Feature: ZS BB/U, Bobot: 6681.638090596008
Feature: ZS BB/TB, Bobot: 4569.992820843093
Feature: Tinggi, Bobot: 3507.848260560233
Feature: Berat, Bobot: 3040.1218697819754
Feature: BB/TB, Bobot: 1398.5830127211766
Feature: LiLA, Bobot: 1026.8887730802444
Feature: BB/U, Bobot: 700.7320680447172
Feature: Naik Berat Badan, Bobot: 679.8755275871911
Feature: JK, Bobot: 112.29386722922584
```

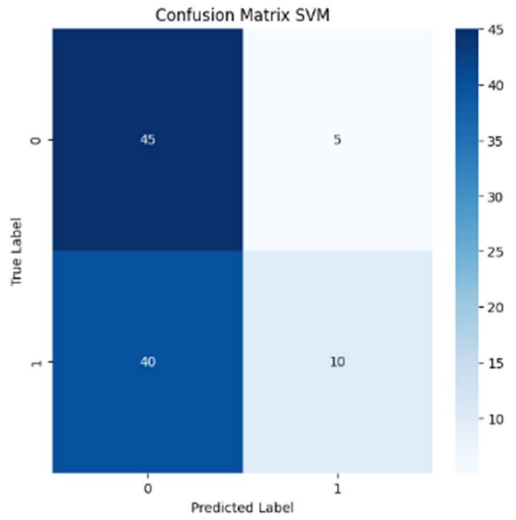
```python
# Fungsi boundary margin relief
def boundary_margin_relief(X_train, y_train):
    n_features = X_train.shape[1]
    features_to_remove = np.random.choice(n_features, n_features // 2, replace=False)
    return features_to_remove
```

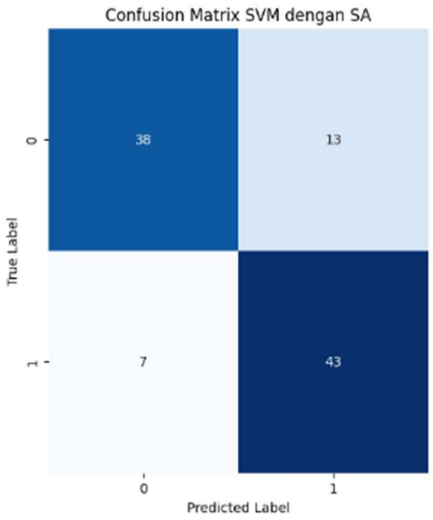
Model SVM dan SVM dengan SA

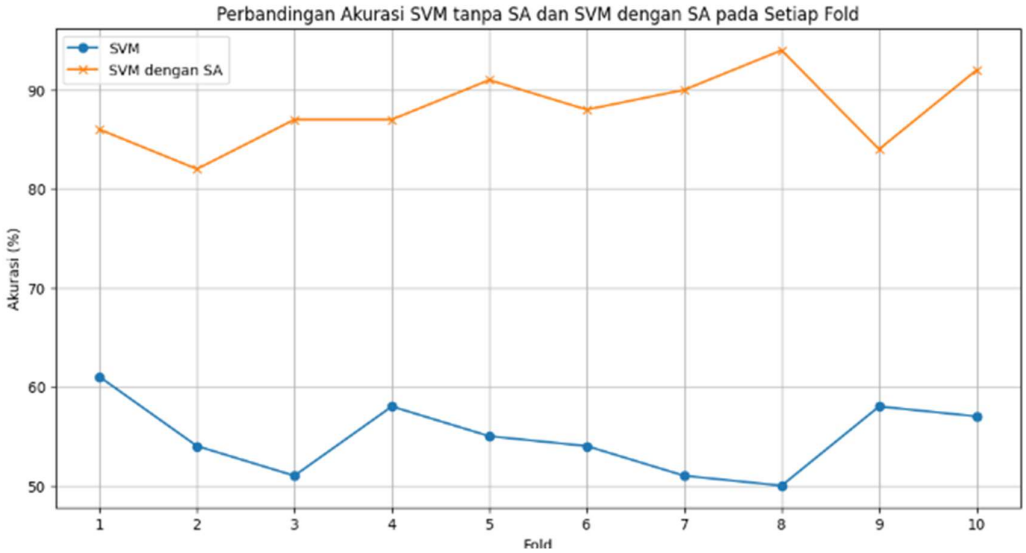
```
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, confusion_matrix
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np

# Variabel untuk menyimpan akurasi dan confusion matrix pada setiap lipatan
accuracies_svm = []
accuracies_optimized = []
confusion_matrices_svm = []
confusion_matrices_optimized = []
accuracy_improvements = []

# Parameter SVM
C = 10
gamma = 5

# Loop melalui setiap lipatan cross-validation
for fold, (train_index, test_index) in enumerate(kfold.split(X_scaled, y), 1):
    X_train, X_test = X_scaled[train_index], X_scaled[test_index]
    y_train, y_test = y[train_index], y[test_index]

    # Model SVM
    svm = SVC(kernel='rbf', C=C, gamma=gamma)  # kernel 'rbf'
    svm.fit(X_train, y_train)
    y_pred_svm = svm.predict(X_test)
    accuracy_svm = accuracy_score(y_test, y_pred_svm) * 100  # Menjadi persen
    accuracies_svm.append(accuracy_svm)
    confusion_matrices_svm.append(confusion_matrix(y_test, y_pred_svm))

    # Model SVM dengan SA
    accuracy_optimized, best_features = simulated_annealing(X_train, y_train, X_test, y_test)
    accuracy_optimized *= 100  # Menjadi persen
    accuracies_optimized.append(accuracy_optimized)
    svm_optimized = SVC(kernel='rbf', C=C, gamma=gamma)  # kernel 'rbf'
    svm_optimized.fit(X_train[:, [i for i in range(X_train.shape[1]) if i not in best_features]], y_train)
    y_pred_optimized = svm_optimized.predict(X_test[:, [i for i in range(X_test.shape[1]) if i not in best_features]])
    confusion_matrices_optimized.append(confusion_matrix(y_test, y_pred_optimized))
    # Perhitungan kenaikan akurasi
    accuracy_improvement = accuracy_optimized - accuracy_svm
    accuracy_improvements.append(accuracy_improvement)

    print(f"Fold {fold}")
    print("Akurasi SVM: {:.2f}%".format(accuracy_svm))
    print("Akurasi SVM dengan SA: {:.2f}%".format(accuracy_optimized))
    print("Kenaikan Akurasi: {:.2f}%".format(accuracy_improvement))
    print("-------------------------------------------------------")

# Menghitung rata-rata akurasi dan confusion matrix
mean_accuracy_svm = np.mean(accuracies_svm)
mean_accuracy_optimized = np.mean(accuracies_optimized)
mean_accuracy_improvement = np.mean(accuracy_improvements)

mean_confusion_matrix_svm = np.mean(confusion_matrices_svm, axis=0)
mean_confusion_matrix_optimized = np.mean(confusion_matrices_optimized, axis=0)

print("Rata-rata Akurasi SVM: {:.2f}%".format(mean_accuracy_svm))
print("Rata-rata Akurasi SVM dengan SA: {:.2f}%".format(mean_accuracy_optimized))
print("Rata-rata Kenaikan Akurasi: {:.2f}%".format(mean_accuracy_improvement))
print(" ")

# Plotting the confusion matrices with consistent color and layout
fig, ax = plt.subplots(1, 2, figsize=(14, 6))
cmap = 'Blues'

sns.heatmap(mean_confusion_matrix_svm, annot=True, fmt=".0f", cmap=cmap, ax=ax[0])
ax[0].set_title('Confusion Matrix SVM')
ax[0].set_xlabel('Predicted Label')
ax[0].set_ylabel('True Label')

sns.heatmap(mean_confusion_matrix_optimized, annot=True, fmt=".0f", cmap=cmap, ax=ax[1])
ax[1].set_title('Confusion Matrix SVM dengan SA')
ax[1].set_xlabel('Predicted Label')
ax[1].set_ylabel('True Label')

plt.show()
```

```python
# Plotting the accuracies (Line Plot)
folds = range(1, len(accuracies_svm) + 1)
plt.figure(figsize=(12, 6))
plt.plot(folds, accuracies_svm, marker='o', label='SVM')
plt.plot(folds, accuracies_optimized, marker='x', label='SVM dengan SA')
plt.title('Perbandingan Akurasi SVM tanpa SA dan SVM dengan SA pada Setiap Fold')
plt.xlabel('Fold')
plt.ylabel('Akurasi (%)')
plt.legend()
plt.grid(True)
plt.xticks(folds)
plt.show()
```

```
Fold 1
Akurasi SVM: 61.00%
Akurasi SVM dengan SA: 86.00%
Kenaikan Akurasi: 25.00%
--------------------------------------------------
Fold 2
Akurasi SVM: 54.00%
Akurasi SVM dengan SA: 82.00%
Kenaikan Akurasi: 28.00%
--------------------------------------------------
Fold 3
Akurasi SVM: 51.00%
Akurasi SVM dengan SA: 87.00%
Kenaikan Akurasi: 36.00%
--------------------------------------------------
Fold 4
Akurasi SVM: 58.00%
Akurasi SVM dengan SA: 87.00%
Kenaikan Akurasi: 29.00%
--------------------------------------------------
Fold 5
Akurasi SVM: 55.00%
Akurasi SVM dengan SA: 91.00%
Kenaikan Akurasi: 36.00%
--------------------------------------------------
Fold 6
Akurasi SVM: 54.00%
Akurasi SVM dengan SA: 88.00%
Kenaikan Akurasi: 34.00%
--------------------------------------------------
Fold 7
Akurasi SVM: 51.00%
Akurasi SVM dengan SA: 90.00%
Kenaikan Akurasi: 39.00%
--------------------------------------------------
Fold 8
Akurasi SVM: 50.00%
Akurasi SVM dengan SA: 94.00%
Kenaikan Akurasi: 44.00%
--------------------------------------------------
Fold 9
Akurasi SVM: 58.00%
Akurasi SVM dengan SA: 84.00%
Kenaikan Akurasi: 26.00%
--------------------------------------------------
Fold 10
Akurasi SVM: 57.00%
Akurasi SVM dengan SA: 92.00%
Kenaikan Akurasi: 35.00%
--------------------------------------------------
Rata-rata Akurasi SVM: 54.90%
Rata-rata Akurasi SVM dengan SA: 88.10%
Rata-rata Kenaikan Akurasi: 33.20%
```






Perbandingan Akurasi SVM tanpa SA dan SVM dengan SA pada Setiap Fold
SVM
SVM dengan SA
Akurasi (%)
90
80
70
60
50
1
2
3
4
5
6
7
8
9
10
Fold

Lampiran 4 Kartu Bimbingan

KARTU KENDALI BIMBINGAN LAPORAN KARYA ILMIAH

Nama Mahasiswa : Mukminatul Munawaroh

NIM : 201112441064

Nama Dosen Pembimbing : Taghfirul Azhima Yoga Siswa, M. Kom

Judul Penelitian : Optimasi Metode BMR dan *Simulated Annealing* untuk Algoritma SVM Dalam Mengatasi *Imbalanced* dan *High Dimensional* Data Stunting

No	Tanggal	Uraian Pembimbingan	Paraf Dosen
1	Jan 22, 2024	Pertemuan pertama pemahaman bidang data science, data mining, machine learning dan pembahasan tahapan penelitian.	
2	Jan 28, 2024	Pertemuan kedua mengumpulkan artikel atau paper sebagai rujukan penelitian	
3	Feb 5, 2024	Pertemuan ketiga melakukan survey studi literatur review paper untuk mencari permasalahan pada bidang klasifikasi data mining yang akan diteliti	
4	Feb 16, 2024	Pertemuan keempat melakukan review technical paper dan road maps penelitian, untuk mencari metode penyelesaian yang sesuai dengan permasalahan pada klasifikasi data mining	
5	Feb 22, 2024	Pertemuan kelima mencari paper atau artikel rujukan 5 tahun terakhir yang sesuai dengan topik penelitian yang telah ditentukan	
6	Feb 26, 2024	Pertemuan keenam penentuan judul penelitian berdasarkan hasil literatur review paper	
7	March 13, 2024	Pertemuan ketujuh membuat canvas penelitian berdasarkan hasil kesimpulan dari review paper, selanjutnya akan di submit ke prodi untuk memenuhi persyaratan skripsi	
8	March 15, 2024	Pertemuan kedelapan pengajuan surat permohonan data ke dinas kesehatan kota samarinda untuk penelitian	
9	April 22, 2024	Pertemuan kesembilan perbaikan atau revisi proposal bab 1 bagian tujuan penelitian dan batasan masalah, bab 2 perjelas bagian implementasi metode dan diagram alur penelitian, perbaikan penulisan berdasaarkan format penulisan skripsi bagian tabel, margin, spasi dan penomoran.	

10	April 24, 2024	Pertemuan kesepuluh revisi akhir bab 1, bab 2 disesuaikan dengan format penulisan skripsi sebelum di submit ke simpel	
11	Mei 19, 2024	Pertrmuan kesebelas penambahan metode untuk evaluasi kodingan	
12	Juni 13, 2024	Pertemuan kedua belas revisi bab 2 bagian kodingan dan bab 3 bagian hasil	
13	Juni 25, 2024	Pertemuan ketiga belas revisi bagian abstak, bagian rumus, dan penyesuaian dengan tempalte paper	
14	Juni 28, 2024	Pertemuan keempat belas revisi akhir bab 3 dan paper yang akan di publikasi	

Dosen Pembimbing

Taghfirul Azhima Yoga Siswa, M. Kom

Mengetahui
Ketua Program Studi

Arbansyah, S.Kom., M.TI

DAFTAR RIWAYAT HIDUP

Mukminatul Munawaroh adalah seorang mahasiswi semester 8 di Universitas Muhammadiyah Kalimantan Timur, Jurusan Teknik Informatika selaku penulis dari naskah skripsi ini. Penulis adalah anak ke 4 dari bapak Mahsun dan ibu Saptiah. Penulis memiliki latar belakang pendidikan yang stabil. Setelah menyelesaikan pendidikan dasar di SDN 011 Tenggarong Seberang, Penulis melanjutkan pendidikan menengah pertamanya di MTS NW II Tenggarong Seberang dan menamatkan SMA di SMA NEGERI 2 Tenggarong Seberang. Selama masa pendidikannya, Penulis mengasah kemampuan di bidang teknologi informasi, terutama dalam penggunaan *Microsoft Office, Wordpress*, *Figma, Canva*, serta bahasa pemrograman *frontend* (HTML, CSS, JS) dan *backend* (*Python, Java*). Pada tahun 2023, Penulis berhasil membangun sebuah website menggunakan *Wordpres*s dan *Django*. Penulis juga mengikuti Pelatihan Kerja Lapangan (PKL) di Dinas Perdagangan Kota Samarinda sebagai Admin *Survey* Data pada semeseter 7 tahun 2023. Dengan kemampuan dan pengalaman yang di dapat, Penulis berharap dapat terus berkembang dan berkontribusi dalam industri teknologi informasi dan sistem informasi. Dalam deskripsi riwayat hidup yang telah disampaikan, penulis dengan rendah hati meminta maaf jika terdapat kesalahan atau kekurangan, karena kesempurnaan hanya miliki oleh Allah SWT, dan penulis dengan tulus mengharapkan kritik serta saran yang membangun terkait skripsi ini.