PERBAIKAN AKURASI *RANDOM FOREST* DENGAN *ANOVA* DAN *SMOTE* PADA KLASIFIKASI DATA STUNTING

SKRIPSI

Diajukan Oleh :
Ari Ahmad Dhani
2011102441090

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

PERBAIKAN AKURASI *RANDOM FOREST* DENGAN *ANOVA* DAN *SMOTE* PADA KLASIFIKASI DATA STUNTING

SKRIPSIs

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

Diajukan Oleh :
Ari Ahmad Dhani
2011102441090

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

LEMBAR PERSETUJUAN

PERBAIKAN AKURASI RANDOM FOREST DENGAN ANOVA DAN SMOTE PADA KLASIFIKASI DATA STUNTING

SKRIPSI

Diajukan oleh:

Ari Ahmad Dhani
2011102441090

Disetujui untuk diujikan Pada tanggal 26 Juni 2024

Pembimbing

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

Mengetahui,
Koordinator Skripsi

Abdul Rahim, S.Kom., M.Cs.
NIDN. 1115039601

LEMBAR PENGESAHAN

PERBAIKAN AKURASI RANDOM FOREST DENGAN ANOVA DAN SMOTE PADA KLASIFIKASI DATA STUNTING

SKRIPSI

Diajukan oleh:

Ari Ahmad Dhani
20111102441090

Diseminarkan dan Diujikan
Pada tanggal 24 Juli 2024

Penguji I	Penguji II
Wawan Joko Pranoto, S.Kom, M.Ti	Tachfirul Azhima Yoga Siswa, S.Kom,M.Kom
NIDN. 1102057701	NIDN. 1118038805

Mengetahui,

Ketua

Program Studi Teknik Informatika

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

PERNYATAAN KEASLIAN PENELITIAN

Saya yang bertanda tangan di bawah ini:

Nama	: Ari Ahmad Dhani
NIM	: 2011102441090
Program Studi	: Teknik Informatika
Judul Penelitian	: Perbaikan Akurasi *Random Forest* Dengan *Anova* Dan *Smote* pada Klasifikasi Data Stunting

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

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 sayaini

Samarinda, 11 Juli 2021
Yang membuat pernyataan

Ari Ahmad Dhani
NIM: 2011102441090

ABSTRAK

Stunting terus menjadi isu kesehatan masyarakat yang kritis di Indonesia, khususnya di Kota Samarinda yang mencatat prevalensi sebesar 25,3% pada tahun 2022, menjadi yang tertinggi kedua di Provinsi Kalimantan Timur. Di tengah prioritas nasional untuk riset 2020-2024, penggunaan data mining untuk klasifikasi stunting memperlihatkan potensi yang signifikan namun tetap menghadapi tantangan dalam menangani data berdimensi tinggi dan ketidakseimbangan kelas. Penelitian ini bertujuan untuk meningkatkan akurasi klasifikasi stunting menggunakan metode *Random Forest* yang diintegrasikan dengan seleksi fitur *ANOVA* dan teknik *SMOTE* untuk menyeimbangkan kelas. Data yang digunakan dalam penelitian ini bersumber dari Dinas Kesehatan Kota Samarinda, meliputi 26 puskesmas dengan 21 atribut dan total 150.466 record. Teknik validasi yang dipakai adalah cross-validation k-fold=10. Hasil menunjukkan peningkatan akurasi dari 98,83% menjadi 99,77% naik sebesar 0,94% setelah penerapan seleksi fitur *ANOVA*. Fitur ZS TB/U, ZS BB/U, dan BB/U diidentifikasi sebagai yang paling berpengaruh. Peningkatan ini menunjukkan efektivitas integrasi metode dalam mengatasi masalah stunting pada dataset yang kompleks dan tidak seimbang, ini diharapkan dapat mendukung kebijakan dan intervensi kesehatan lebih lanjut di kawasan tersebut.

Kata kunci: Klasifikasi, *Random Forest, ANOVA, SMOTE, High Dimensional*

ABSTRACT

Stunting continues to be a critical public health issue in Indonesia, particularly in Samarinda City, which recorded a prevalence of 25.3% in 2022, the second highest in East Kalimantan Province. Amidst the national research priorities for 2020-2024, the use of data mining for stunting classification shows significant potential but still faces challenges in handling high-dimensional data and class imbalance. This study aims to improve stunting classification accuracy using the Random Forest method integrated with ANOVA feature selection and SMOTE technique for class balancing. The data used in this study were sourced from the Samarinda City Health Office, encompassing 26 health centers with 21 attributes and a total of 150,466 records. The validation technique used is 10-fold cross-validation. The results show an accuracy increase from 98.83% to 99.77%, an increase of 0.94%, after applying ANOVA feature selection. The features ZS TB/U, ZS BB/U, and BB/U were identified as the most influential. This increase demonstrates the effectiveness of the method integration in addressing the stunting problem in complex and imbalanced datasets and is expected to support further health policies and interventions in the area.

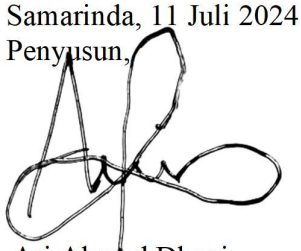
Keywords: Classification, Random Forest, ANOVA, SMOTE, High Dimensional

PRAKATA

Alhamdulillah, puji syukur kehadirat Allah SWT yang telah melimpahkan Rahmat dan Karunia-Nya, sehingga peneliti dapat menyelesaikan skripsi dengan judul "Perbaikan Akurasi *RANDOM FOREST* Dengan *ANOVA* Dan *SMOTE* Pada Klasifikasi Data Stunting". Dalam penyusunan skripsi ini, peneliti banyak mendapatkan bantuan dari beberapa pihak. Oleh karena itu, pada kesempatan ini peneliti ingin mengucapkan terima kasih kepada:

1. Bapak Sukari, Ibu Rumini, Kakak, dan Adik tercinta yang selalu memberikan doa serta dukungan kepada penulis.
2. Dr. Muhammad Musiyam, M.T, selaku Rektor Universitas Muhammadiyah Kalimantan Timur.
3. Bapak Arbansyah, S.Kom., M.TI selaku ketua Program Studi S1 Teknik Informatika.
4. Bapak Taghfirul Azhima Yoga Siswa, S.Kom, M.Kom selaku Dosen pembimbing yang telah memberikan wawasan dan arahan dalam pengerjaan skripsi ini.
5. Bapak Wawan Joko Pranoto, S.Kom, M.Ti selaku Dosen Penguji yang telah memberikan masukan dan arahan dalam revisi skripsi ini.
6. Seluruh Bapak dan Ibu Dosen Program Studi Teknik Informatika Universitas Muhammadiyah Kalimantan Timur dan 1 staff tendik yang penulis banggakan dan hormati.
7. Perpustakaan Daerah, Kota Samarinda, dan Perpustakaan Universitas Muhammadiyah Kalimantan Timur.
8. Tim RTA yang selalu siap membantu dalam pengerjaan skripsi ini
9. Tim GC Elite yaitu Renal, Bulan,Vito ttp semangat dan semoga dipermudah dalam meraih gelar S.Kom.
10. Pemilik Nim 2011102431071 terima kasih telah menjadi salah satu sosok penting baik dialam perkuliahan sampai ditahap ini, Telah berkontribusi banyak dalam penulisan skripsi ini, meluangkan waktu, pikiran, dan pengetahuan. Terima kasih karena sudah menjadi bagian dalam menuju gelar S.Kom semoga juga dipermudah dalam menuju gelar S.M dan semoga bisa menjadi Pribadi yang lebih baik lagi.

Samarinda, 11 Juli 2024
Penyusun,



Ari Ahmad Dhani
NIM: 2011102441090

DAFTAR ISI

Halaman

DAFTAR TABEL

Tabel Halaman

DAFTAR GAMBAR

Gambar Halaman

DAFTAR LAMPIRAN

Lampiran

BAB I

PENDAHULUAN

1.1 Latar Belakang

Stunting merupakan sebuah penyakit yang saat ini dianggap sebagai permasalahan serius dan menjadi fokus perhatian pemerintah sebagai prioritas riset nasional tahun 2020 – 2024. Berdasarkan data dari kementrian kesehatan pada tahun 2022, prevalensi stunting pada anak indonesia dengan usia dibawah lima tahun masih tergolong tinggi yakni sebesar 21,6% (Cindy, 2023). Dengan artian, penyakit stunting setidaknya mempengaruhi 22 anak dari 100 anak di bawah usia lima tahun. Prevalensi stunting di indonesia juga tergolong tinggi jika dibandingkan dengan negara negara yang ada di kawasan ASEAN. Sebagai salah satu kota penyangga IKN, Samarinda memiliki prevalensi stunting tertinggi kedua setelah kabupaten Kutai Kartanegara di provinsi Kalimantan Timur tahun 2022 dengan besar persentase 25,3% (Cindy, 2023). Stunting pada anak balita dapat menyebabkan berbagai masalah kesehatan jangka panjang dan menghambat perkembangan kognitif pada anak sehingga menyebabkan prestasi belajar yang lebih rendah jika dibandingkan dengan anak yang tidak stunting (Pratiwi et al., 2021). Oleh karena itu, penting untuk mengembangkan strategi penggunaan data mining dalam mengklasifikasi status stunting pada anak di bawah lima tahun. Metode ini diharapkan dapat menghasilkan informasi dan pengetahuan yang berguna sebagai dasar dalam pengambilan keputusan dan pertimbangan untuk mendeteksi stunting pada anak.

Berdasarkan penelitian yang sudah pernah dilakukan dalam bidang data mining dalam klasifikasi status stunting telah dilakukan dengan beberapa pendekatan, seperti algoritma *Random Forest, Support vector machine, KNN, Neural Network, Naive Bayes, classification tree* dan lain lain. Penelitian yang dilakukan oleh Apriyani & Kurniati (2020) dilakukan pengujian terhadap metode algoritma *Naive Bayes dan Support vector machine* (SVM*)* hasil penelitiaan menunjukan akurasi yang terbilang cukup baik dengan persentase akurasi sebesar 90%. Namun pada penelitian terkait masih menggunakan data berdimensi rendah (low – dimensional) atau data yang memiliki atribut sedikit untuk digunakan sebagai analisis. Data berdimensi rendah juga memiliki kelemahan seperti overfitting, dan sulit diinterpretasikan (Hakimah et al., 2022). Berdasarkan penelitian yang dilakukan Gebeye *et al.,*(2023) Model *Random Forest* (RF), *Support Vector Machine* (SVM), *Logistic Regression* (LR), *Neural Network* (NN*), dan Naive Bayes* (NB) mengalami penurunan performa saat berhadapan dengan data berdimensi tinggi, yang mempengaruhi akurasi metode klasifikasi tersebut. Hasil akurasi yang diperoleh menunjukkan bahwa Support Vector Machine (SVM) mencapai akurasi sebesar 71.03%, Random Forest (RF) mencapai 72.41%, Neural Network (NN) mencapai 71.03%, dan Naive Bayes (NB) mencapai 67.93%. Temuan ini menyoroti tantangan yang dihadapi oleh model klasifikasi saat beroperasi pada data dengan dimensi tinggi, yang dapat mengurangi efektivitas dan akurasi dari metode yang digunakan.

Data berdimensi tinggi merupakan sebuah data yang memiliki banyak atribut atau fitur yang digunakan dalam menganalisis. Sebagai contoh, bila suatu data yang memiliki atribut berjumlah puluhan bahkan ratusan atribut, maka data tersebut dapat dikategorikan sebagai data berdimensi tinggi. Data berdimensi tinggi memiliki beberapa tantangan, termasuk peningkatan kompleksitas model, risiko overfitting, dan kesulitan dalam visualisasi data (Hakimah et al., 2022). Model-model seperti *Random Forest, Support Vector Machine* , *Logistic Regression*, *Neural Network*, dan *Naive Bayes* dapat mengalami penurunan performa ketika berhadapan dengan data berdimensi tinggi (Gebeye et al., 2023), sehingga mempengaruhi akurasi klasifikasi yang didapatkan. Oleh karena itu, strategi seperti reduksi dimensi dan pemilihan fitur diperlukan untuk mengatasi masalah ini. Pemilihan fitur memungkinkan model untuk fokus pada fitur-fitur yang paling relevan dengan masalah yang dihadapi.

Algoritma klasifikasi pada penelitian ini akan menggunakan *Random Forest* (RF), hal ini berdasarkan penelitian yang dillakukan oleh (Chilyabanyama et al., 2022; Gebeye et al., 2023; Hemo, and Rayhan, 2021; Talukder and Ahammed, 2020) dimana *RF* memiliki performa terbaik jika dibandingkan dengan algoritma *Support Vector Machine, Logistic Regression, Neural Network, Naïve Bayes, linear discriminant analysis* (LDA), *k-nearest neighbors* (k-NN), *eXtreme Gradient Boosting* (Xg boost) dan *Classification and Regression Tree*s (CART) terkait klaisifikasi data stunting. Penggunaan seleksi fitur *Analysis of variances* (ANOVA) juga akan digunakan untuk menemukan atribut-atribut yang paling relevan dalam klasifikasi. Penelitian Nugroho *et al.,*(2022) metode *ANOVA* terbukti bisa meningkatkan akurasi dari metode klasfikasi KNN dan *decision tree* (DT) dimana sebelum penerapan akurasi yang didapat sebesar 67% untuk KNN dan 57% untuk DT kemudian setelah dilakukan penerapan metode *ANOVA* di dapatkan peningkatan akurasi yang tinggi dimana akurasi KNN menjadi 87% dan DT menjadi 96% pada data stunting. Kemudian dalam penelitian yang dilakukan oleh Hasan (2020) metode *ANOVA* dikombinasikan dengan metode klasifikasi *RF* maka didapatkan akurasi yang cukup tinggi sebesar 91%. Selain itu pada penelitian lain yang melakukan komparasi penggunaa seleksi fitur *ANOVA* terhadap algoritma klasifikasi mendapatkan kenaikan akurasi 0.8% menjadi 94.1% untuk Random Forest dengan tingkat akurasi paling tinggi dibandingkan dengan algoritma klasifikasi lainnya (Yoga Siswa, 2023). Dalam beberapa penelitian terkait stunting juga masih ditemukan ketidakseimbangan kelas seperti penelitian yang dilakukan oleh Sutarmi *et al.* (2023).

Ketidakseimbangan kelas terjadi ketika jumlah sampel dari satu kelas secara signifikan lebih banyak atau lebih sedikit dibandingkan dengan kelas lainnya (Luo et al., 2019). Dalam konteks stunting, hal ini bisa berarti bahwa jumlah anak yang tidak mengalami stunting jauh lebih banyak daripada yang mengalami stunting, atau sebaliknya. Masalah ini dapat menyebabkan bias pada model yang cenderung mengklasifikasi kelas mayoritas dengan lebih baik dibandingkan kelas minoritas. Untuk mengatasi masalah ketidakseimbangan kelas dan memastikan bahwa model dapat memprediksi kedua kelas (stunting dan tidak stunting) dengan akurat, bisa menggunakan teknik *oversampling* dan *undersampling* untuk menyeimbangkan kelas, oleh karena itu penelitian ini juga akan menggunakan teknik *Synthetic Minority Over-sampling Technique* (SMOTE).

Berdasarkan pada penelitian yang sudah pernah dilakukan Santoso et al., (2019) metode *SMOTE* terbukti mampu meningkatkan akurasi pada algoritma klasifikasi dimana Naive Bayes mencapai akurasi sebesar 62.4% dengan data asli dan meningkat menjadi 72.6% setelah menggunakan *SMOTE*. *Support Vector Machine* menunjukkan peningkatan dari 50.3% menjadi 77.0%, dan *Random Forest* meningkat dari 48.6% menjadi 80.8%. Hasil ini membuktikan bahwa penggunaan metode *SMOTE* untuk menangani ketidakseimbangan kelas mampu memberikan kenaikan akurasi yang cukup baik terutama jika dikombinasikan dengan algoritma *Random Forest*.

Penelitian ini akan menggunakan algoritma Random Forest yang sering kali menunjukkan performa lebih baik dibandingkan dengan metode lain dalam mengklasifikasi stunting, terutama ketika dikombinasikan dengan teknik pemilihan fitur dan penanganan ketidakseimbangan kelas. Namun, sebagian besar penelitian sebelumnya menggunakan data dengan dimensi yang berbeda dan teknik pemilihan fitur yang beragam. Kombinasi Random Forest dengan seleksi fitur ANOVA dan teknik SMOTE secara bersamaan bertujuan untuk meningkatkan akurasi klasifikasi stunting dengan menangani tantangan data berdimensi tinggi dan ketidakseimbangan kelas secara lebih efektif. Dengan demikian, penelitian ini diharapkan dapat memberikan kontribusi baru dalam pemanfaatan teknik data mining untuk klasifikasi stunting pada anak balita, serta memberikan dasar yang lebih kuat untuk pengambilan keputusan dan intervensi yang lebih tepat dalam mengatasi masalah stunting.

1.2 Rumusan Masalah

Berdasarkan latar belakang yang telah dijabarkan, maka didapatkan rumusan masalah pada penelitian ini sebagai berikut:

a) Fitur-fitur apa saja yang memberikan pengaruh signifikan pada algoritma *Random Forest* dengan dataset Stunting Kota Samarinda menggunakan seleksi fitur *ANOVA*?
b) Seberapa meningkat akurasi yang didapat algoritma *Random Forest* dalam mengklasifikasi data stunting Kota Samarinda dengan menggunakan seleksi fitur *ANOVA* dan metode *oversampling SMOTE*?

1.3 Tujuan Penelitian

Berikut adalah tujuan dari penelitian ini:

a) Mengidentifikasi atribut apa saja yang berpengaruh terhadap akurasi model *Random Forest* dalam Analisis Data Stunting di Kota Samarinda
b) Mengimplementasi dan mengevaluasi algoritma *Random Forest* untuk klasifikasi penyakit Stunting di Kota Samarinda, dengan penerapan seleksi fitur *ANOVA* dan metode *balancing SMOTE.* Kinerja model dievaluasi *menggunakan K-Fold Cross Validation* dan *confusion matrix*, untuk mengukur akurasi guna memastikan efektivitas model dalam klasifikasi Stunting.

1.4 Manfaat Penelitian

Diharapkan penelitian ini dapat memberikan manfaat dan pengetahuan kepada berbagai pihak, khususnya:

a) Penulis
   Untuk menambah pengetahuan dan wawasan penulis dalam mengimplementasikan metode klasifikasi dengan algoritma *Random Forest* dengan seleksi fitur *ANOVA* untuk mengingkatkan akurasi menggunakan algoritma *Random Forest* pada dataset stunting Kota Samarinda.
b) Pembaca
   Penelitian ini bermanfaat bagi pembaca yang menekuni bidang data mining, membantu mengasah kemampuan analisis data, penggunaan *Random Forest,* serta implementasi *ANOVA* dan *SMOTE* pada dataset stunting di Samarinda. Hasilnya juga dapat dijadikan referensi untuk penelitian di masa mendatang.

1.5 Batasan Masalah

Agar masalah yang dibahas tidak menjadi lebih luas lagi, maka penulis membatasi masalah penelitian sebagai berikut:

a) Data yang digunakan adalah data penyakit stunting kota samarinda pada tahun 2023.
b) Melakukan Penerapan seleksi fitur *ANOVA* pada algoritma *Random Forest* untuk memilih fitur pada data stunting berdimensi tinggi.
c) Melakukan penerapan *oversampling SMOTE* pada algoritma *Random Forest* untuk menyeimbangkan kelas pada data stunting.
d) Penelitian ini akan menggunakan atribut yang telah di seleksi dari dataset, yaitu *(i)* Nama, *(ii)* JK, *(iii)* Berat, *(iv)* Tinggi, *(v)* LiLA, *(vi)* BB/U, *(vii)* ZS BB/U, *(viii)* TB/U, *(ix)* ZS TB/U, *(x)* BB/TB, *(xii)* ZS BB/TB, *(xiii)* Naik Berat Badan, *(xiv)* Jml Vit A, *(xv)* Tanggal Pengukuran.

BAB II

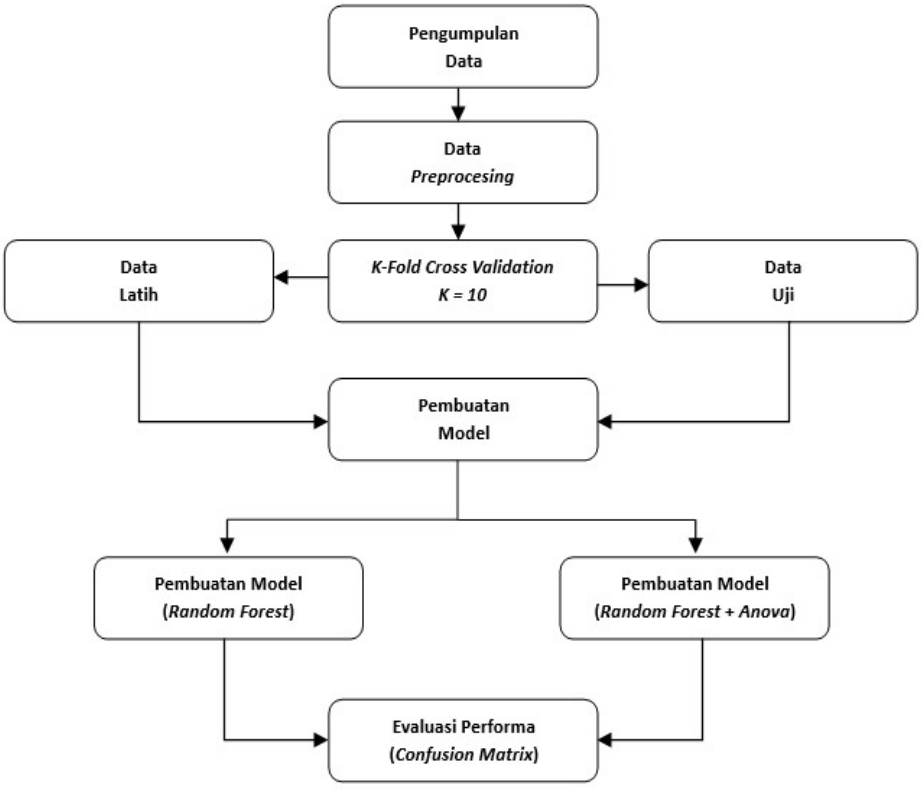
METODOLOGI PENELITIAN

2.1 Objek Penelitian

Stunting merupakan masalah kesehatan yang terjadi pada anak anak. Di kota Samarinda kasus Stunting terbilang cukup tinggi yaitu sebesar 25,3% (Cindy Mutia Annur, 2023). Oleh karena itu, Objek yang digunakan dalam penelitian ini adalah data Stunting pada tahun 2023 yang didapat dari 26 puskesmas di lingkup Dinas kesehatan Kota Samarinda. Penelitian dilakukan di Dinas Kesehatan Kota Samarinda yang berlokasi di Jl. Milono No.1, 75122, Bugis, Kec. Samarinda Kota, Kota Samarinda, Kalimantan Timur 76112.

2.2 Teknik Analisis Data

Penelitian ini bertujuan untuk mengembangkan sebuah metodologi yang terstruktur dan terorganisir dengan baik untuk memastikan bahwa proses penelitian dilaksanakan secara konsisten dan teratur. Teknik analisis data yang akan digunakan telah dirancang untuk mengikuti alur kerja yang sistematis, yang akan diuraikan dalam alur penelitian berikut :



Gambar 2. 1 Alur Penelitian

2.2.1 Pengumpulan data

Tahapan awal pada penelitian ini adalah melakukan perhaman pada data, dimulai dari pengumpulan data lalu dilanjutkan dengan proses data preparation atau persiapan data kasus Stunting. Data yang didapat memiliki 23 kolom dengan 22 atribut dan 1 atribut sebagai label (TB/U). Dapat dilihat pada Tabel 2.1 adalah data yang didapat dari Dinas kesehatan Kota Samarinda.

Tabel 2.1 Atribut Data Dinas Kesehatan Kota Samarinda

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

2.2.2 Data *Pre-Procesing*

Pada Tahapan ini akan memiliki 4 tahapan, yaitu data *selection*, data *cleaning*, data *Transformation* dan data *Balancing*. Data yang digunakan berasal dari data Dinas Kesehatan Kota Samarinda dengan proses persiapan sebagai berikut :

a) Data *Selection*

Pada tahapan ini, akan dilakukan pengambilan data dengan memililih fitur atau atribut yang relevan untuk digunakan pada penelitian sedangkan fitur yang dianggap tidak relevan akan dihilangkan, seperti pada tabel 2.2 menampilkan data awal yang didapatkan dari Dinas Kesehatan Kota Samarinda memiliki 23 kolom, lalu didapatkan 11 kolom yang dianggap tidak relevan untuk digunakan dalam mengklasifikasikan stunting pada anak, setelah dilakukan proses seleksi maka didapatkan 13 atribut yang dijadikan fitur dan 1 atribut (TB/U) sebagai target atau kelas. Dapat dilihat pada Tabel 2.2 adalah data yang telah diseleksi.

Tabel 2.2 Data *Selection*

No	Atribut	Keterangan	Tipe Data
1	Nama	Nama	*String*
2	JK	Jenis Kelamin	*String*
3	Berat	Berat Badan	*Integer*
4	Tinggi	Tinggi Badan	*Integer*
5	LiLA	Lingkar Lengan Atas	*Integer*
6	BB/U	Berat Badan Menurut Umur	*numeric*
7	ZS BB/U	*Z Score* Berat Badan menurut Umur	*numeric*
8	Tanggal Pengukuran	Tanggal Pengukuran	*Integer*
9	ZS TB/U	*Z Score* Tinggi Badan menurut Umur	*numeric*
10	BB/TB	Berat Badan menurut Tinggi Badan	*numeric*
11	ZS BB/TB	*Z Score* Berat Badan menurut Tinggi Badan	*numeric*
12	Naik Berat B	Naik Berat Badan	*String*
13	Jml Vit A	Jumalah Vitamin A	*integer*
14	TB/U	Tinggi Badan menurut Umur	*numeric*

b) Data *Cleaning*

Pada Tahapan ini data yang digunakan akan dilakukan proses pembersihan dengan menghapus data yang memiliki nilai #N/A (*no value is available*) atau data yang tidak memiliki nilai serta yang kembar atau terduplikasi. Proses ini akan menggunakan bahasa pemrograman *Python* serta library Pandas, dengan fungsi ***dropna()*** untuk menghapus baris yang memiliki nilai hilang dan fungsi ***drop_duplicates()*** untuk menghapus baris yang merupakan duplikat.

```
missing_values = data.isna().sum()

# Menampilkan jumlah missing values untuk setiap kolom
print("Jumlah missing values untuk setiap kolom:")
print(missing_values)
```

Gambar 2. 2 Kode Menampilkan Nilai Kosong

Berikut adalah tabel 2.3 yang berisi tentang tiap parameter yang digunakan dalam kode tersebut beserta penjelasan fungsi tiap parameter.

Tabel 2.3 Parameter Kode Menampilkan Nilai Kosong

Parameter	Keterangan
data	Ini adalah variabel yang merujuk pada *DataFrame pandas. DataFrame* adalah struktur data dua dimensi yang bisa menyimpan data dalam bentuk tabel dengan baris dan kolom.
isna()	Fungsi *isna()* dari pandas digunakan untuk mendeteksi nilai-nilai yang hilang dalam *DataFrame*. Fungsi ini mengembalikan *DataFrame* baru di mana setiap *entri* adalah *True* jika *entri* asli adalah nilai yang hilang (NaN, None, atau NaT), dan *Fals*e jika tidak.
sum()	Metode *sum()* digunakan di sini setelah isna() untuk menghitung jumlah True di setiap kolom dari DataFrame yang dihasilkan oleh *isna()*. Ini efektif menghitung jumlah nilai yang hilang dalam setiap kolom dari DataFrame asli data.

```python
# 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')

#Menghapus Nilai Kosong Tiap Kolom
data.dropna(inplace=True)

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

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

Gambar 2.3 Kode Menghapus Data Terduplikasi Dan Data Kosong

Berikut adalah tabel 2.4 yang berisi tentang tiap parameter yang digunakan dalam kode tersebut beserta penjelasan fungsi tiap parameter.

Tabel 2.4 Parameter Kode Menghapus Data Terduplikasi Dan Data Kosong

Fungsi		Parameter	Keterangan
len()	:	data	data adalah *DataFrame* yang sedang diproses. Fungsi *len*() di sini digunakan untuk menghitung jumlah baris dalam DataFrame.
sort_values()	:	*by*	Parameter by menerima list dari nama-nama kolom yang akan dijadikan dasar pengurutan. Di sini, pengurutan dilakukan berdasarkan kolom "Nama" dan "Tanggal Pengukuran".
		ascending	Parameter *ascending* menerima list dari nilai *boolean* yang menentukan urutan pengurutan untuk setiap kolom yang ditentukan di *by*. Nilai *True* untuk "Nama" berarti pengurutan secara *ascending* (A-Z), dan *False* untuk "Tanggal Pengukuran" berarti pengurutan secara *descending* (tanggal terbaru ke terlama).
drop_duplicates()	:	*subset*	Parameter *subset* menentukan kolom yang akan digunakan untuk identifikasi duplikat. Di sini, 'Nama' adalah kolom yang digunakan, yang berarti duplikat akan dicari berdasarkan nama.
		keep	Parameter *keep* menentukan duplikat mana yang akan dipertahankan. *first* menunjukkan bahwa hanya baris pertama dari setiap duplikat yang akan dipertahankan, yang di kasus ini adalah entri dengan 'Tanggal Pengukuran' terbaru karena pengurutan sebelumnya.
print()	:	*f-string*	*f-string* digunakan untuk memformat string, menyisipkan nilai dari variabel langsung ke dalam *string* yang dicetak. total_data_sebelum dan total_data_sesudah adalah variabel yang nilai-nya disisipkan dan dicetak.

c) Data *Transformation*

Pada tahapan ini dilakukan dengan proses konversi nilai-nilai atribut yang kategorikal menjadi bentuk numerik, tahapan ini perlu dilakukan karena dalam penggunaan library sklearn, hanya bisa menerima atribut dengan tipe numerik. Beberapa atribut yang ditransformasi meliputi jenis kelamin, naik berat badan, berat badan menurut umur, dan berat badan menurut tinggi badan. Proses ini akan menggunakan *library scikit-learn* dengan fungsi *LabelEncoder*. *Label Encoder* tersebut digunakan untuk mengubah teks atau data kategorikal menjadi data numerik dalam satu kolom data secara otomatis (M. M. Mafa'atiih, 2020)

```
from sklearn.preprocessing import LabelEncoder

# Buat instance LabelEncoder
encoder = LabelEncoder()

# Kolom yang perlu di-encode
columns_to_encode = ['JK', 'BB/U', 'BB/TB', 'Naik Berat Badan']

# Loop melalui setiap kolom yang perlu di-encode dan terapkan LabelEncoder
for column in columns_to_encode:
    data[column] = encoder.fit_transform(data[column])
```

Gambar 2.4 Kode Tranformasi Data Pada Kolom Atribut

```
# Ubah TB/U
data['TB/U'] = data['TB/U'].replace({'Normal': 0, 'Tinggi': 0, 'Pendek': 1, 'Sangat Pendek': 1})
```

Gambar 2.5 Kode Tranformasi Data Pada Kolom Label

Berikut adalah tabel 2.5 yang berisi tentang tiap parameter yang digunakan dalam kode tersebut beserta penjelasan fungsi tiap parameter.

Tabel 2.5 Parameter Kode Tranformasi

Fungsi	Parameter	Keterangan
LabelEncoder()	-	*LabelEncoder* adalah kelas dari *sklearn.preprocessing* yang digunakan untuk mengkonversi label kategori menjadi label numerik.
fit_transform()	*data[column]*	Metode *fit_transform()* digunakan untuk menyesuaikan encoder dan mengubah label menjadi bentuk *numerik*. Parameter *data[column]* merupakan kolom spesifik dari *DataFrame* data yang sedang diolah.
replace()	*{'Normal': 0, 'Tinggi': 0, 'Pendek': 1, 'Sangat Pendek': 1}*	Metode *replace()* digunakan untuk mengganti nilai-nilai dalam sebuah Series atau *DataFrame*. Parameter ini merupakan *dictionary* yang menentukan nilai lama dan baru; dalam hal ini, 'Normal' dan 'Tinggi' diganti dengan 0, 'Pendek' dan 'Sangat Pendek' diganti dengan 1.

d) Data *Balancing*

Pada tahapan terakhir preprocessing, ketidakseimbangan kelas akan terjadi ketika jumlah atribut mengalami ketidakseimbangan antara kelas mayoritas dan kelas minoritas. Sehingga Perlu melakukan data balancing menggunakan modul *python imblearn.over_sampling* dengan mengimpor fungsi *SMOTE* (*sampling strategy*) untuk melakukan *oversampling*.

```
x = data.drop(['TB/U'],axis=1)
y = data['TB/U']
from imblearn.over_sampling import SMOTE
sm = SMOTE(random_state=42, sampling_strategy=1)
X_imb, y_imb = sm.fit_resample(x, y)
```

Gambar 2.6 Kode Data *Balancing SMOTE*

Berikut adalah tabel 2.6 yang berisi tentang tiap parameter yang digunakan dalam kode tersebut beserta penjelasan fungsi tiap parameter.

Tabel 2.6 Parameter Kode Data *Balancing SMOTE*

Fungsi	Parameter	Keterangan
drop()	*['TB/U']*	Parameter ini menentukan nama kolom atau daftar kolom yang akan dihapus dari *DataFrame* data. Di sini, kolom 'TB/U' akan dihapus.
	axis=1	*Axis* 1 mereferensikan kolom. Dengan *axis*=1, operasi *drop*() akan menghapus kolom yang ditentukan.
SMOTE	*random_state=42*	Memastikan bahwa hasil *oversampling* konsisten di berbagai eksekusi dengan menyediakan *seed* untuk generator bilangan acak. 42 adalah nilai yang sering digunakan dalam contoh untuk tujuan *reproducibilitas*.
	sampling_strategy=1	Menentukan strategi *sampling* yang diinginkan. Nilai 1 mengindikasikan bahwa kelas minoritas akan di*resample* hingga jumlahnya sama dengan kelas mayoritas.
fit_resample()	*x, y*	Metode *fit_resample()* digunakan untuk mengajarkan model *SMOTE* pada data input (x) dan target (y), kemudian mengembalikan dataset yang telah di-resample. x adalah *DataFrame* yang dihasilkan setelah menghapus kolom 'TB/U', dan y adalah nilai target yang berasal dari kolom 'TB/U'.

2.2.3 Pembagian data

Sebelum memulai proses pemodelan, dataset yang akan digunakan harus dibagi menjadi dua bagian utama, yaitu data latih dan data uji. Data latih berfungsi sebagai dasar untuk pembangunan model, sementara data uji digunakan untuk mengevaluasi kinerja model yang telah dibuat. Dalam penelitian ini, akan menggunakan teknik *K-Fold Cross Validation*, Hal ini dapat membantu mengidentifikasi

apakah model memiliki tingkat akurasi yang baik, serta membantu menghindari *overfitting* atau *underfitting* (Hafid, 2023). *K-Fold Cross Validation* akan diimplementasikan melalui *library* sklearn.model_selection dengan fungsi cross_val_score pada *Python*. Teknik ini melibatkan pembagian data menjadi 10 segmen atau kelompok, sesuai dengan parameter Cv yang ditetapkan sebesar 10, yang berarti data akan dibagi menjadi 10 bagian. Setiap segmen akan secara bergiliran berperan sebagai data latih dan data uji. Pada Penelitian ini menggunakan nilai k sebesar 10, yang menandakan bahwa eksperimen akan dijalankan sebanyak 10 kali. Setelah melakukan 10 kali percobaan, maka nilai rata-rata dari semua hasil pengujian akan diambil sebagai indikator untuk mendapatkan hasil yang lebih presisi dan akurat.

Tabel 2.7 *K-fold Cross Validation*

1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10
1	2	3	4	5	6	7	8	9	10

Pada Tabel 2.7, skema 10-fold CV dijelaskan, di mana setiap baris mewakili satu iterasi. Blok berwarna merah menunjukkan data yang digunakan sebagai data uji dalam iterasi tersebut, sedangkan blok berwarna putih menunjukkan data latih. Contoh visualisasinya adalah sebagai berikut, Pada Iterasi 1, Fold 1 digunakan sebagai data uji dan Fold 2 hingga Fold 10 sebagai data latih, pada Iterasi 2, Fold 2 sebagai data uji dan Fold 1 serta Fold 3 hingga Fold 10 sebagai data latih. Demikian seterusnya hingga Iterasi 10, di mana Fold 10 digunakan sebagai data uji dan Fold 1 hingga Fold 9 sebagai data latih. Teknik ini memastikan bahwa setiap fold digunakan sebagai data uji sekali, dan data latih pada iterasi lainnya, untuk memberikan evaluasi model yang lebih akurat dan menghindari overfitting.

2.2.4 Pembuatan Model

Berikut ini merupakan gambaran dan penjelasan dari model klasifikasi *Random Forest* yang digunakan pada penelitian in :

a) Dalam fase ini, dataset akan dilakukan pembagian menjadi dua segmen utama: data latih dan data uji. Data latih berperan sebagai pondasi untuk proses pembentukan model, sedangkan data uji digunakan sebagai alat untuk mengevaluasi efektivitas model yang sudah dibangun. Penelitian ini menggunakan metode *K-Fold Cross Validation*, sebuah teknik yang diaktualisasikan melalui penggunaan *library* sklearn.model_selection dan fungsi *cross_val_score* pada bahasa pemrograman *Python*. Dengan parameter Cv yang ditetapkan sebesar 10, ini mengindikasikan bahwa dataset akan terbagi menjadi 10 bagian atau kelompok yang sama. adapun model yang dipilih untuk melakukan klasifikasi Stunting di Kota Samarinda adalah model *Random Forest*. Berikut adalah penjelasan mengenai mekanisme kerja dari metode tersebut:

$$l(\mathrm{y}) = argmax_c(\sum_{n=1}^{N} I_{hn}(\mathrm{y}) = C) \quad (2.1)$$

a. Langkah Pertama sampel dataset tersebut dimanfaatkan untuk membentuk pohon keputusan atau *tree* ke-i (i = 1, 2, ...k). Dalam tahap pembuatan tree ini, digunakan metode *Classification and Regression Tree* (CART), di mana gini impurity diaplikasikan sebagai kriteria untuk menentukan pembagian pada setiap node dalam tree (Yoga Siswa, 2023) .

```
#Kode Inisialisasi Random Forets
from sklearn.ensemble import RandomForestClassifier
clf = RandomForestClassifier(max_depth=1, random_state=42)
```

Gambar 2.7 Kode Inisialisasi Model *Random Forest*

b. Langkah kedua melakukan pembagian data menggunakan *K-Fold Cross Validation* dengan K = 10.

```
# Kode kfold Cross Validation
from sklearn.model_selection import cross_val_score, StratifiedKFold
cv = StratifiedKFold(n_splits=10, shuffle=True, random_state=42)
```

Gambar 2.8 Kode Pembagian Data Latih dan Data Uji

Berikut adalah tabel 2.8 yang berisi tentang tiap parameter yang digunakan dalam kode tersebut beserta penjelasan fungsi tiap parameter:

Tabel 2.8 Parameter Pembagian Data Latih dan Data Uji

Parameter	Keterangan
n_splits	Jumlah lipatan (fold) yang akan dibuat. Dalam kasus ini, nilai n_splits = 10 berarti akan memiliki 10 lipatan untuk cross-validation.
shuffle	Menentukan apakah data akan diacak sebelum dibagi menjadi lipatan. Jika shufffle = True, data akan diacak sebelum dibagi, sehingga setiap lipatan akan memiliki sampel yang berbeda-beda. Jika shufffle = false, data akan dibagi berdasarkan urutan aslinya.
random_state	Menentukan seed untuk generator nomor acak yang digunakan saat mengacak data. Penggunaan nilai random_state yang sama akan menghasilkan pembagian yang sama saat melakukan pengacakan.

c. Langkah ketiga dalam pemodelan algoritma Random Forest melibatkan penggunaan *library scikit-learn* untuk mengimplementasikan model. Pada tahap ini, akan menggunakan metode *k-fold cross-validation* untuk membagi dataset menjadi data latih dan uji, serta untuk mengukur performa model secara objektif. Pertama, membuat model *Random Forest* menggunakan *RandomForestClassifier* dari *scikit-learn*. Selanjutnya, melakukan perulangan melalui setiap *fold* yang dihasilkan oleh *k-fold cross-validation*. Pada setiap perulangan, dataset dibagi menjadi data latih dan uji , dan model *Random Forest* dilatih menggunakan

data latih. Setelah dilatih, model tersebut digunakan untuk melakukan prediksi pada data uji. Skor evaluasi akurasi dihitung menggunakan metrik yang sesuai dari library *sklearn.metrics* dan disimpan untuk setiap *fold*. Selain itu, matriks kebingungan (confusion matrix) dari setiap *fold* juga dihitung menggunakan fungsi *confusion_matrix* dari *scikit-learn*. Hasil prediksi dan skor evaluasi ini kemudian disimpan untuk digunakan dalam analisis lebih lanjut tentang performa model pada dataset yang diberikan.

```
from sklearn.model_selection import cross_val_score, StratifiedKFold
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score
import numpy as np

# Perform cross-validation on the Random Forest classifier with K=10
clf = RandomForestClassifier(max_depth=1, random_state=42)

# Kode kfold Cross Validation
cv = StratifiedKFold(n_splits=10, shuffle=True, random_state=42)

# Lists to store evaluation metrics for each fold
conf_matrices = []
accuracies = []
precisions = []
recalls = []
f1_scores = []

for train_index, test_index in cv.split(X_imb, y_imb):
    X_train, X_test = X_imb.iloc[train_index], X_imb.iloc[test_index]
    y_train, y_test = y_imb.iloc[train_index], y_imb.iloc[test_index]

    # Train the model
    clf.fit(X_train, y_train)

    # Predict on the test set
    y_pred = clf.predict(X_test)

    # Calculate evaluation metrics
    conf_matrices.append(confusion_matrix(y_test, y_pred))
    accuracies.append(accuracy_score(y_test, y_pred))
```

Gambar 2.9 Kode Permodelan Algoritma *Random Forest*

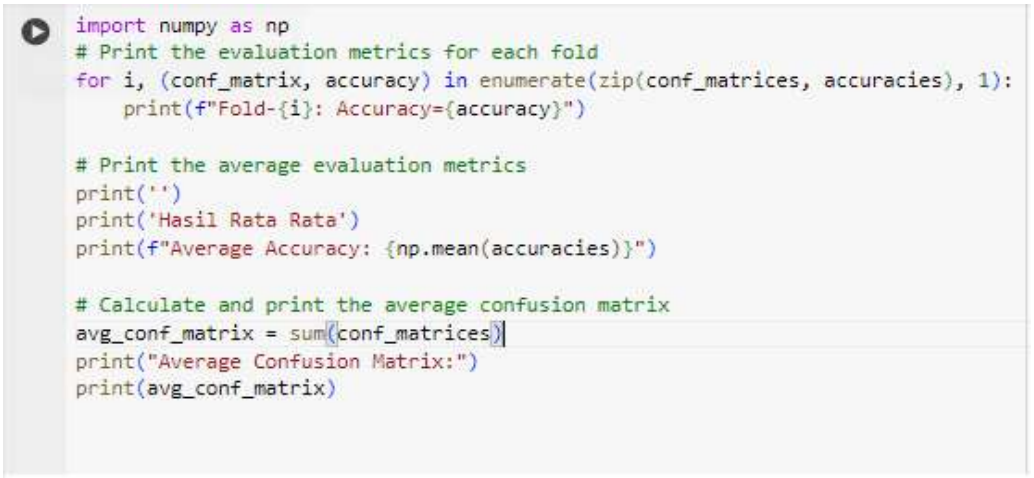
Berikut adalah tabel 2.9 yang berisi tentang tiap parameter yang digunakan dalam kode tersebut beserta penjelasan fungsi tiap parameter:

Tabel 2.9 Parameter Permodelan Algoritma *Random Forest*

Parameter	Keterangan
clf	Model *Random Forest* yang digunakan untuk melakukan klasifikasi.
train_index	Indeks data untuk data latih.
test_index	Indeks data untuk data uji.
accuracies	List yang akan menyimpan skor akurasi dari setiap *fold*.
confusion_matrices	List yang akan menyimpan *confusion matrix* dari setiap *fold*.
X_train	Subset atribut untuk training.
X_test	Subset atribut untuk testing.
y_train	Subset label untuk training..
y_test	Subset label untuk testing.
y_pred	Hasil prediksi dari subset test.

d. Langkah keempat Setelah proses *K-fold cross-validation* selesai dilakukan dan model *Random Forest* telah diuji pada setiap *fold*, skor evaluasi seperti akurasi dari setiap *fold* dapat ditampilkan. Dalam pengulangan menggunakan *enumerate* dan *zip*, akurasi dari masing-masing *fold* diproses dan ditsmpilkan. Selain itu, rata-rata akurasi dari semua *fold* juga dihitung dengan menggunakan fungsi *np.mean()* dari *NumPy*. Hasil rata-rata skor evaluasi

akurasi dari seluruh *fold* juga dicetak. Terakhir, rata-rata dari *confusion matrix* dari semua *fold* dihitung menggunakan fungsi *np.mean()* dan ditampilkan. *Confusion matrix*) rata-rata ini memberikan gambaran tentang seberapa baik model dapat mengklasifikasikan setiap kelas pada data uji.



```
import numpy as np
# Print the evaluation metrics for each fold
for i, (conf_matrix, accuracy) in enumerate(zip(conf_matrices, accuracies), 1):
    print(f"Fold-{i}: Accuracy={accuracy}")

# Print the average evaluation metrics
print('')
print('Hasil Rata Rata')
print(f"Average Accuracy: {np.mean(accuracies)}")

# Calculate and print the average confusion matrix
avg_conf_matrix = sum(conf_matrices)
print("Average Confusion Matrix:")
print(avg_conf_matrix)
```

Gambar 2. 10 Kode Hasil akurasi dari setiap *Fold*

Berikut adalah tabel 2.10 yang berisi tentang tiap parameter yang digunakan dalam kode tersebut beserta penjelasan fungsi tiap parameter

Tabel 2. 10 Parameter Hasil Skor Dari Setiap *Fold*

Parameter	Keterangan
np	Alias untuk *numpy*, sebuah library *Python* yang digunakan untuk melakukan operasi numerik pada *array*, matriks, dan struktur data numerik lainnya.
enumerate	Fungsi *Python* yang digunakan untuk mengambil elemen dari sebuah iterable (seperti *list, tuple,* atau *string*) bersamaan dengan indeksnya dalam bentuk *tuple*.
zip	Fungsi *Python* yang digunakan untuk menggabungkan elemen-elemen yang bersesuaian dari dua atau lebih *iterable* (*list, tuple,* atau *array*) menjadi *tuple-tuple* baru
accuracies	List yang menyimpan nilai akurasi untuk setiap *fold* dalam *cross-validation*
mean	Fungsi numpy yang digunakan untuk menghitung rata-rata dari sebuah array atau list
axis	Parameter yang digunakan dalam fungsi *np.mean* untuk menentukan sumbu (axis) mana yang akan dihitung rata-ratanya
avg_cm	Nilai rata-rata dari *confusion matrix* untuk semua *fold* dalam *cross-validation*

b) Tahap seleksi fitur menggunak metode *ANOVA* dengan cara menentukan atribut yang memiliki bobot atau pengaruh signifikan terhadap variabel yang akan diklasifikasi. Nilai F yang lebih besar menunjukkan bahwa ada perbedaan yang signifikan antara kelompok, karena menunjukkan bahwa

variansi antar kelompok lebih besar dibandingkan dengan variansi dalam kelompok. Sebaliknya, nilai F yang rendah menunjukkan bahwa perbedaan antar kelompok tidak signifikan, menunjukkan bahwa kelompok-kelompok tersebut memiliki variansi yang serupa. Berikut tahapan kerja dari seleksi fitur *Analysis of Variance* (ANOVA) adalah sebagai berikut :

a. Semua fitur dipilih pada dataset Stunting
b. Fungsi fitur target dari scikit-learn dihitung menggunakan *ANOVA F-Score* untuk setiap fitur.Berikut adalah rumus untuk menghitung *ANOVA*:

$$F = \frac{variance\ between\ groups}{variance\ within\ groups} \tag{2.2}$$

$$Variance\ between\ groups = \frac{\sum_i^n n_i\,(\bar{Y}_{i\,-}\,\bar{Y})^2}{(k-1)} \tag{2.3}$$

$$Variance\ within\ groups = = \frac{\sum_{i=1}^{k}\sum_{j=1}^{n_i}(\bar{Y}_{ij\,-}\,\bar{Y})^2}{(n-k)} \tag{2.4}$$

Keterangan :

n_i = Jumlah sampel di grup ke-i
$\bar{Y}_i$ = Rata-rata sampel di grup ke-i
$\bar{Y}$ = Rata-rata total sampe
k = Jumlah grup
n = Jumlah total sampel

c. Hasil dari pengujian akan digunakan dalam pemilihan fitur yang mempunyai pengaruh tinggi dan varian terendah dipilih dalam percobaan ini lalu akan diuji dengan SelectKbeast(); k mewakili jumlah fitur yang digunakan untuk dataset akhir
d. Jumlah fitur dengan pengaruh tertinggi akan digunakan untuk membuat berbagai subset fitur.

```
from sklearn.feature_selection import SelectKBest, f_classif

X = data

# Menggunakan SelectKBest dengan ANOVA untuk memilih fitur terbaik
selector = SelectKBest(score_func=f_classif)
X_selected = selector.fit_transform(X, y)

# Mengambil nilai F dan nama fitur dari SelectKBest
f_values = selector.scores_
features = X.columns

# Membuat DataFrame dari fitur dan nilai F
import pandas as pd
df_f = pd.DataFrame({'Feature': features, 'F_Value': f_values})

# Sorting DataFrame berdasarkan nilai F secara ascending
df_sorted = df_f.sort_values('F_Value', ascending=False)

# Membuat diagram bar untuk menampilkan nilai F dari semua fitur
plt.figure(figsize=(12, 6))
plt.bar(df_sorted['Feature'], df_sorted['F_Value'], color='skyblue')
plt.xlabel('Fitur')
plt.ylabel('Nilai F')
plt.title('Perbandingan Nilai F dari Semua Fitur (Ascending)')
plt.xticks(rotation=90)
plt.show()
```

Gambar 2. 11 Kode Seleksi Fitur *ANOVA*

Berikut adalah tabel 2.11 yang berisi tentang tiap parameter yang digunakan dalam kode tersebut beserta penjelasan fungsi tiap parameter

Tabel 2.11 Parameter Seleksi Fitur *ANOVA*

Fungsi	Parameter	Keterangan
SelectKBest	*score_func=f_classif*	*SelectKBest* adalah metode untuk seleksi fitur yang memilih K fitur terbaik berdasarkan nilai statistik tertentu. *score_func = f_classif* menunjukkan penggunaan *F-test ANOVA* untuk tugas klasifikasi, yang mengukur pengaruh tiap fitur terhadap *variabilitas* antara kelas.
fit_transform()	*X, y*	Metode *fit_transform()* digunakan untuk mengajarkan *selector* pada data X (fitur) dan y (target) dan mengubah X menjadi subset fitur yang terpilih berdasarkan skor tertinggi.
DataFrame()	*{'Feature': features, 'F_Value': f_values}*	Membuat *DataFrame* dari dua *arrays,* features yang berisi nama fitur dari X dan *f_values* yang berisi nilai F dari setiap fitur setelah dihitung oleh *SelectKBest.*
sort_values()	*'F_Value', ascending=False*	Mengurutkan DataFrame berdasarkan kolom *F_Value* secara descending untuk melihat fitur dengan pengaruh paling tinggi terlebih dahulu.

2.2.5 Evaluasi

Ditahapan evaluasi akan dilakukan pengukuran akurasi dari algoritma yang digunakan dengan kualitas data training serta akan diuji dengan menggunakan teknik C*onfusion Matrix.*

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \times 100\% \quad (2.5)$$

Keterangan :

TP (True Positive) : Jumlah data point berlabel yes yang nilainya diidentifikasi benar.
TP (True Negative) : Jumlah data point berlabel no yang nilainya diidentifikasi salah.
FP (False Positive) : Jumlah data point berlabel yes yang nilai sebenarnya diidentifikasi salah
FN (False Negative) : Jumlah data point berlabel no yang nilai sebenarnya teridentifikasi benar

BAB III

HASIL DAN PEMBAHASAN

3.1 Hasil Penelitian

Berdasarkan hasil penelitian yang telah dilakukan, tujuan utama adalah untuk mengevaluasi kinerja algoritma Random Forest dengan tambahan metode *SMOTE* dan *ANOVA* dalam klasifikasi penyakit Stunting di Kota Samarinda, dengan menggunakan metrik utama seperti akurasi. Akurasi memberikan wawasan mengenai efektivitas algoritma dalam mengklasifikasi Stunting. Dataset yang didapat memiliki 23 kolom dengan 22 atribut dan 1 atribut sebagai label (TB/U).

3.1.1 Data *Selection*

Pada Tabel 3.1 merupakan atribut yang telah dipilih secara manual terdapat 13 Atribut yang terpilih dan 1 atribut TB/U sebagai target atau kelas

Tabel 3. 1 Data *Selection*

	Nama	JK	Berat	Tinggi	LiLA	BB/U	ZS BB/U	Tanggal Pengkuran	ZS TB/U	BB/TB	ZS BB/TB	Naik Berat Badan	Jml Vit A	TB/U
1	DIMAS ADITYA	L	9.01		0	Kurang	-0.39	2023-01-02	-0.21	Gizi Baik	-0.39	O		Normal
2	SITI AISYAH	P	12	94	0	Berat Badan	-2.25	2023-01-02	-2.09	Gizi Baik	-1.46	O		Pendek
3	M AL FATIH ALMAHIRA	L	8.01	69	0	Normal Berat Badan	-0.53	2023-01-02	-0.65	Gizi Baik	-0.14	O		Normal
4	AKIRA AKBAR GIUNIA	P	6.03		0	Normal	-0.31 0.09097	2023-01-02	0.42	Gizi Baik	-0.74	O		Normal
5	QAMELA	P	10.06		0	Risiko Lebih	2222	2023-01-02	0.23	Gizi Lebih	0.09	O		Normal
.....	.....	.....	.....	.....	...	.....	.....	.....	.....	.....	.....	.....	....	.....
150462	ADNAN IRAGUSTI SIENA AL	L	3	50		Normal Berat Badan	-0.73	2023-12-30	0.06	Gizi Baik	-1.19	-		Normal
150463	RAISHA AFIZAH	P	13		0	Normal	-1.22	2023-12-13	-1.11	Gizi Baik	-0.85	O		Normal
150464	KHAIRINA M ARSYA	P	2.05	45		Kurang Berat Badan	-2.03	2023-12-09	-2.63	Gizi Baik	-0.35	-		Normal
150465	KHOLIF MUHAMMAD	P	3	49		Normal	-1.56	2023-12-19	-1.3	Gizi Baik	-1.04	-		Pendek
150466	IQBAL	L	2.09	49		Kurang	-2.97	2023-12-29	-2.48	Gizi Baik	-1.37	-		Normal

3.1.2 Data *Cleaning*

Pada Tabel 3.2 Merupakan tampilan dataset sebelum dilakukan pembersihan data untuk menghapus baris yang memiliki nilai hilang dan menghapus baris yang terduplikat pada dataset stunting dengan jumlah 150466 record. Proses ini memastikan data bersih dan siap untuk analisis lebih lanjut.

Tabel 3. 2 Dataset sebelum dibersihkan

	Nama	JK	Berat	Tinggi	LiLA	BB/U	ZS BB/U	Tanggal Pengkuran	ZS TB/U	BB/TB	ZS BB/TB	Naik Berat Badan	Jml Vit A	TB/U
1	DIMAS ADITYA	L	9.01		0	Kurang	-0.39	2023-01-02	-0.21	Gizi Baik	-0.39	O		Normal
2	SITI AISYAH	P	12	94	0	Berat Badan	-2.25	2023-01-02	-2.09	Gizi Baik	-1.46	O		Pendek
3	M AL FATIH ALMAHIRA	L	8.01	69	0	Normal Berat Badan	-0.53	2023-01-02	-0.65	Gizi Baik	-0.14	O		Normal
4	AKIRA AKBAR GIUNIA	P	6.03		0	Normal	-0.31 0.09097	2023-01-02	0.42	Gizi Baik	-0.74	O		Normal
5	QAMELA	P	10.06		0	Risiko Lebih	2222	2023-01-02	0.23	Gizi Lebih	0.09	O		Normal
.....	.....	.....	.....	.....	...	.....	.....	.....	.....	.....	.....	.....	....	.....
150462	ADNAN IRAGUSTI SIENA AL	L	3	50		Normal Berat Badan	-0.73	2023-12-30	0.06	Gizi Baik	-1.19	-		Normal
150463	RAISHA AFIZAH	P	13		0	Normal	-1.22	2023-12-13	-1.11	Gizi Baik	-0.85	O		Normal
150464	KHAIRINA M ARSYA	P	2.05	45		Kurang Berat Badan	-2.03	2023-12-09	-2.63	Gizi Baik	-0.35	-		Normal
150465	KHOLIF MUHAMMAD	P	3	49		Normal	-1.56	2023-12-19	-1.3	Gizi Baik	-1.04	-		Pendek
150466	IQBAL	L	2.09	49		Kurang	-2.97	2023-12-29	-2.48	Gizi Baik	-1.37	-		Normal

Pada tampilan data sebelum dilakukan proes pembersihan terdapat banyak data yang memiliki kesamaan nama dikarenakan pemeriksaan stunting selama periode tahun 2023 terjadi beberapa kali namun untuk penelitian ini hanya akan menggunakan tanggal pemeriksaan terbaru sehingga perlu untuk melakukan penghapusan data untuk memililih tanggal pemeriksaan terbaru setelah proses penghapusan data yang terduplikasi dilanjutkan dengan penghapusan kolom yang memuat nilai N/A (no value is available) atau data yang tidak memiliki nilai.

```
Jumlah missing values untuk setiap kolom:
Nama                    6
JK                      0
Berat                 558
Tinggi              51222
LiLA                29075
BB/U                    0
ZS BB/U               159
TB/U                12871
ZS TB/U             12264
BB/TB               12812
ZS BB/TB            12200
Naik Berat Badan       57
Jml Vit A           95334
Tanggal Pengukuran      0
dtype: int64
```

Gambar 3. 1 Jumlah Nilai Kosong Tiap Kolom Sebelum Pembersihan

Tabel 3. 3 Dataset setelah dibersihkan

	Nama	JK	Berat	Tinggi	LiLA	BB/U	ZS BB/U	Tanggal Pengkuran	ZS TB/U	BB/TB	ZS BB/TB	Naik Berat Badan	Jml Vit A	TB/U
1	A FARIS WICAKSONO	L	9.07	78.0	16	Berat Badan Normal	-1.75	2023-10-06	-2.84	Gizi Baik	-0.47	O	1	Pendek
2	A FATHAN	L	15.00	107.0	17	Berat Badan Normal	-1.08	2023-10-23	0.14	Gizi Baik	-1.83	T	1	Normal
3	A FAUJAN	L	14.00	100.0	0	Berat Badan Normal	-0.85	2023-02-07	-0.16	Gizi Baik	-1.14	O	1	Normal
4	A MISHAEL	P	15.00	103.0	0	Berat Badan Normal	0.06	2023-02-09	1.05	Gizi Baik	-0.80	O	1	Normal
5	A ZHAFIR KAMIL	L	14.00	102.0	0	Berat Badan Normal	-1.91	2023-10-14	-1.49	Gizi Baik	-1.59	O	1	Normal
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
8055	yelmi	L	17.04	111.0	0	Berat Badan Normal	-0.04	2023-02-27	0.06	Gizi Baik	-0.93	N	1	Normal
8056	yumna	P	13.07	100.0	14	Berat Badan Normal	-1.07	2023-10-14	-0.45	Gizi Baik	-1.18	O	1	Normal
8057	ziaan aqilla karno	L	14.05	102.0	0	Berat Badan Normal	-1.71	2023-10-21	-1.62	Gizi Baik	-1.15	T	1	Normal
8058	zubair	L	17.03	107.0	0	Berat Badan Normal	-0.14	2023-02-20	-0.13	Gizi Baik	-0.13	O	1	Normal
8059	zulkifli abdi	L	15.06	102.0	0.0	Berat Badan Normal	-0.59	2023-10-03	-0.69	Gizi Baik	-0.25	N	1	Normal

Setelah dilakukan proses pembersihan dapat terlihat pada tabel 3.3 jumah data yang memiliki nilai kosong sudah tidak ada dan data yang tersisa setelah proses pembersihan data berjumlah 8059 record .

```
Jumlah missing values untuk setiap kolom:
Nama                    0
JK                      0
Berat                   0
Tinggi                  0
LiLA                    0
BB/U                    0
ZS BB/U                 0
TB/U                    0
ZS TB/U                 0
BB/TB                   0
ZS BB/TB                0
Naik Berat Badan        0
Jml Vit A               0
Tanggal Pengukuran      0
dtype: int64
```

Gambar 3. 2 Jumlah Nilai Kosong Tiap Kolom Setelah Pembersihan

3.1.3 Data *Transformation*

Beberapa atribut yang ditransformasi meliputi jenis kelamin, naik berat badan, berat badan menurut umur, dan berat badan menurut tinggi badan. Proses ini melibatkan teknik encoding untuk mengubah data kategori menjadi numerik dan penggantian nilai pada kolom label TB/U untuk mengasumsikan kelas menjadi dua kategori. Dengan transformasi ini, data menjadi lebih siap untuk digunakan dalam analisis dan pemodelan.

Tabel 3. 4 Dataset sebelum ditransformasi

	JK	**BB/U**	**BB/TB**	**Naik Berat Badan**	**TB/U**
1	L	Berat Badan Normal	Gizi Baik	O	Pendek
2	L	Berat Badan Normal	Gizi Baik	T	Normal
3	L	Berat Badan Normal	Gizi Baik	O	Normal
4	P	Berat Badan Normal	Gizi Baik	O	Normal
5	L	Berat Badan Normal	Gizi Baik	O	Normal
...	...	...	...	...	...
8055	L	Berat Badan Normal	Gizi Baik	N	Normal
8056	P	Berat Badan Normal	Gizi Baik	O	Normal
8057	L	Berat Badan Normal	Gizi Baik	T	Normal
8058	L	Berat Badan Normal	Gizi Baik	O	Normal
8059	L	Berat Badan Normal	Gizi Baik	N	Normal

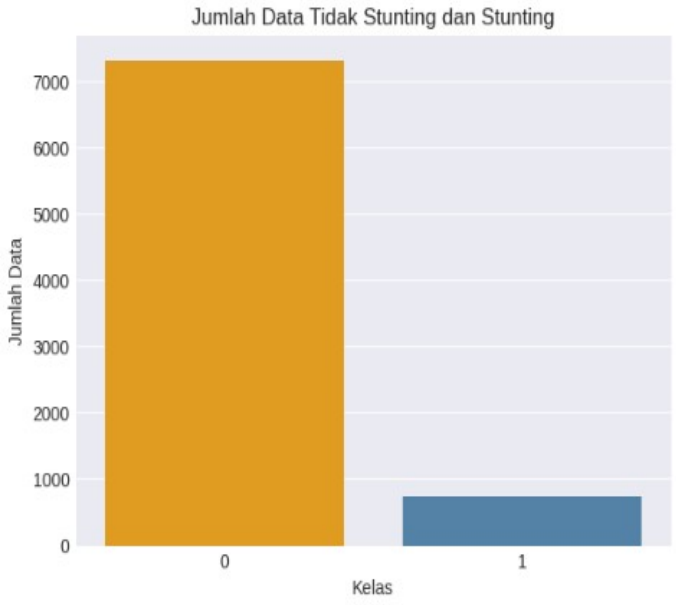
Tabel 3. 5 Dataset Setelah ditransformasi

	JK	BB/U	BB/TB	Naik Berat Badan	TB/U
1	0	0	0	2	1
2	0	0	0	3	0
3	0	0	0	2	0
4	1	0	0	2	0
5	0	0	0	2	0
...	...	…	….	...	….
8055	0	0	0	4	0
8056	1	0	0	2	0
8057	0	0	0	3	0
8058	0	0	0	2	0
8059	0	0	0	4	0

Pada tabel 3.5 tampilan data pada kolom atribut 'JK', 'BB/U', 'BB/TB', 'Naik Berat Badan' dan label 'TB/U' setelah dilakukan transformasi data dimana data yang sebelumny berupa String di ubah menjadi Integer untuk memudahkan proses klasifikasi.

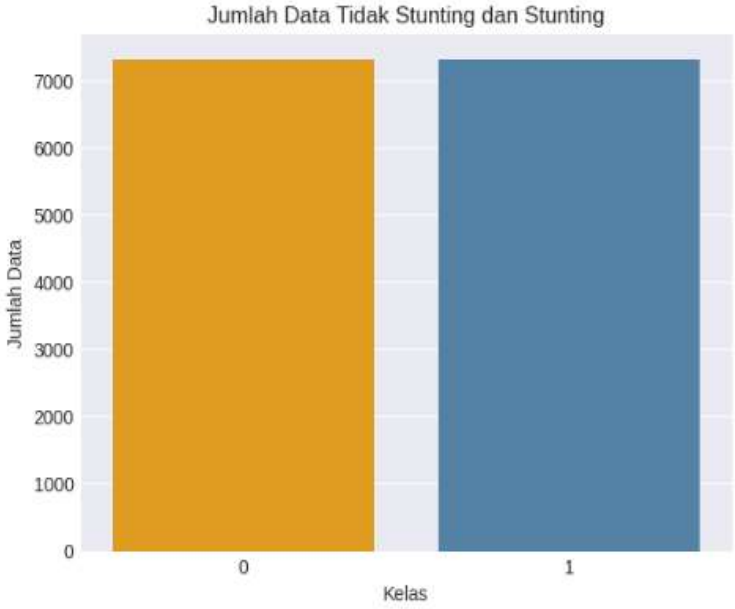
3.1.4 Data *Balancing*

Proses penyeimbangan data dilakukan dengan menggunakan teknik oversampling. Teknik ini bertujuan untuk menyamakan jumlah sampel antara kelas minoritas dan kelas mayoritas dalam dataset.



Gambar 3. 3 Jumlah Kelas Sebelum Data *Balancing*

Pada Gambar 3.3 terdapat perbedaan jumlal kelas dimana kategori tidak stunting (0) berjumlah 7317 data dan kategori stunting (1) berjumlah 742 data.



Gambar 3. 4 Jumlah Kelas Sesudah Data *Balancing*

Pada gambar 3.4 menunjukan perbandingan jumlah kelas yang sudah seimbang antara kelas mayoritas dan kelas minoritas setelah proses *SMOTE* dimana jumlah data kategori tidak stunting (0) berjumlah 7317 data dan kategori stunting (1) berjumlah 7317 data.

3.1.5 Permodelan *Random forest* Tanpa *ANOVA*

Pada tahap ini, akan menjalankan eksperimen pertama dengan melakukan pengujian model menggunakan *Python*. Pengujian ini melibatkan implementasi algoritma *Random Forest* tanpa penggunaan teknik *Analysis of variances* (ANOVA). Tujuan dari pengujian ini adalah untuk menilai kinerja dasar model *Random Forest* dalam mengklasifikasikan data.

Hasil pengujian pada tabel 3.6 menunjukkan bahwa model *Random Forest* tanpa seleksi fitur *ANOVA* pada setiap fold menunjukkan kinerja yang sangat baik dalam memprediksi kelas pada dataset Stunting Kota Samarinda. Dalam kesepuluh f*old* yang dievaluasi, model ini mencapai akurasi yang konsisten tinggi, dengan nilai rata-rata mencapai 98.83%. Tingkat akurasi yang tinggi ini menunjukkan kemampuan model untuk secara tepat mengklasifikasikan data, dengan tingkat kesalahan yang rendah.

Tabel 3. 6 Hasil Pengujian model *Random Forest*

Fold	Akurasi
1	97.95%
2	98.84%
3	98.91%
4	98.91%
5	98.56%
6	99.18%
7	99.38%
8	98.91%
9	99.18%
10	98.50%

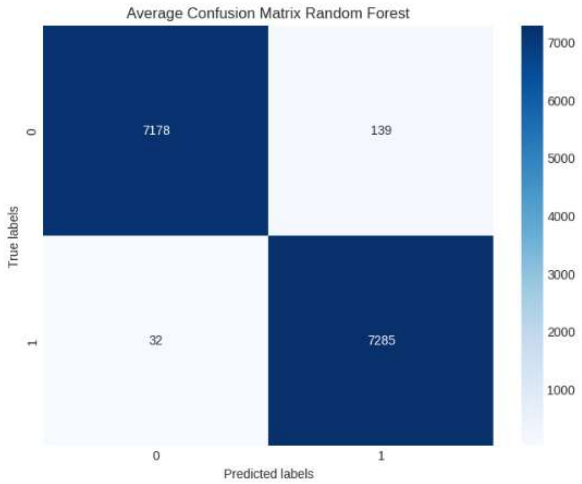
Tabel 3. 7 Rata Rata Akurasi Pengujian Model *Random Forest*

Hasil Rata Rata	
Akurasi	98.83%

Dari evaluasi model *Random Forest* tanpa *ANOVA*, menampilkan hasil yang cukup baik, dimana mendapatkan evaluasi di atas 90% pada tiap parameter. Hasil ini menunjukkan bahwa model *Random* Forest yang dikembangkan mampu melakukan klasifikasi dengan tingkat keakuratan yang sangat tinggi. Hal ini memberikan indikasi kuat bahwa fitur-fitur yang digunakan dalam model telah efektif dalam membedakan berbagai kelas dengan hasil yang diinginkan. Meskipun tanpa penerapan teknik *ANOVA* sebagai seleksi fitur, untuk membandingkan hasil evaluasi tersebut maka dilakukan perhitungan manual nilai *confusion matrix* untuk mengkonfirmasi keakuratan hasil evaluasi tersebut.

Tabel 3. 8 *Confusion Matrix*

	Predicted Positive (0)	*Predicted Negative* (1)
Actual Positive (0)	7178	139
Actual Negative(1)	32	7285



Gambar 3. 5 *Confusion Matrix Random Forest*

$$Accuracy = \frac{7178 + 7285}{7178 + 7285 + 139 + 32} = 0.9883 = 98.83\%$$

Setelah dilakukan proses perhitungan manual pada tiap parameter maka didapatkan hasil yang sama dengan perhitungan menggunakan *python*, hasil ini menunjukkan kesesuaian yang sangat baik dengan perhitungan manual, hal ini menunjukan bahwa model memiliki kemampuan yang baik dalam menerapkan data latih ke data uji dan mampu mengklasifikasikan data dengan akurat.

3.1.6 Permodelan *Random Forest* Dengan *ANOVA*

Pada tahap ini, akan menjalankan eksperimen kedua dengan melakukan pengujian model menggunakan *Python*. Pengujian ini melibatkan implementasi algoritma Random Forest dengan tambahan penggunaan teknik *Analysis of variances* (ANOVA). Tujuan dari pengujian ini adalah untuk menilai kinerja model *Random Forest* setelah penerapan *ANOVA* dalam mengklasifikasikan data.

a) Seleksi Fitur

Seleksi fitur dilakukan untuk merangkingkan atribut yang ada, mulai dari atribut yang memiliki pengaruh besar (mendapatkan nilai F tertinggi) terhadap hasil klasifikasi hingga atribut yang memiliki pengaruh kecil (mendapatkan nilai F terendah). Dengan melakukan seleksi fitur, maka dapat dipilih subset dari atribut-atribut tersebut yang paling berpengaruh dalam memprediksi atau mengklasifikasikan target. Seleksi fitur membantu mengurangi dimensi data, mempercepat proses pembelajaran, dan mencegah overfitting (Bengnga and Ishak, 2022). Proses seleksi fitur sangat penting dalam dataset stunting di Kota Samarinda, di mana terdapat beragam atribut yang mungkin memiliki kontribusi berbeda terhadap status stunting. Dengan mengidentifikasi atribut yang paling relevan, maka dapat meningkatkan akurasi model prediksi, mengurangi waktu komputasi, dan meningkatkan interpretabilitas model. Selain itu, seleksi fitur juga membantu dalam mengeliminasi atribut yang tidak relevan, sehingga model menjadi lebih efisien dan mudah dipahami. Dengan demikian, seleksi fitur tidak hanya meningkatkan kinerja model secara keseluruhan, tetapi juga memungkinkan penggunaan sumber daya yang lebih efektif dalam analisis data.

Tabel 3. 9 Hasil Perengkingan *ANOVA*

Atribut	Nilai F	Ranking
ZS TB/U	3091	1
ZS BB/U	975	2
BB/U	737	3
Tinggi	441	4
Berat	290	5
LiLA	32	6
ZS BB/TB	4	7
JK	8	8
BB/TB	4	9
Naik Berat Badan	3	10
Jml Vit A	NaN	11

Pada hasil perangkingan atribut yang dapat dilihat pada tabel 3.9 , maka ditentukan atribut yang akan digunakan adalah atibut dengan rangking 1 – 6 sebagai atribut dalam permodelan *Random Forest* karena memiliki nilai F yang tinggi sehingga atribut pada pemodelan ini hanya 6 yaitu ZS TB/U, ZS BB/U, BB/U, Tinggi, Berat, LiLA Selain itu, kolom 'Jml Vit A' perlu dihapus dari dataset karena library pandas membacanya sebagai nilai NaN, yang tidak dapat diproses dalam pemodelan.

b) Permodelan

Hasil pengujian pada tabel 3.10 menunjukkan bahwa model *Random Forest* dengan seleksi fitur *ANOVA* pada kesepuluh fold menunjukkan kinerja yang sangat baik dalam memprediksi kelas pada dataset Stunting Kota Samarinda. Model ini mencapai akurasi yang tinggi, dengan nilai rata-rata mencapai 99.78%. Tingkat akurasi yang tinggi ini menunjukkan kemampuan model untuk secara tepat mengklasifikasikan data, dengan tingkat kesalahan yang sangat rendah. Penggunaan seleksi fitur *ANOVA* dalam proses pemodelan telah membantu model dalam mengidentifikasi atribut-atribut yang paling berpengaruh dalam klasifikasi stunting, sehingga meningkatkan akurasi prediksinya.

Tabel 3. 10 Hasil Pengujian Model *Random Forest*

Fold	Akurasi
1	99.66%
2	99.80%
3	99.86%
4	99.93%
5	99.66%
6	99.80%
7	99.80%
8	99.52%
9	99.86%
10	99.86%

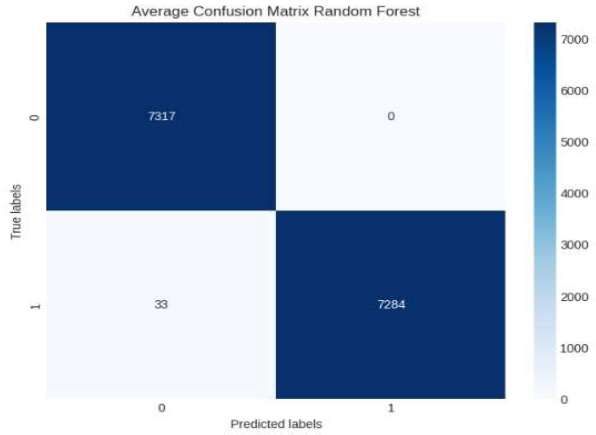
Tabel 3. 11 Rata Rata Akurasi pengujian model *Random Forest*

Hasil Rata Rata	
Akurasi	99.77%

Dari evaluasi model *Random Forest* dengan *ANOVA*, menunjukan hasil yang cukup baik dilihat dari kenaikan akurasi setelah penerapan metode seleksi fiitur. Hasil ini menunjukan bahwa model *Random Forest* mampu memaksimalkan hasil yang didapat setelah penambahan metode tersebut. Untuk membandingkan hasil evaluasi teersebut maka dilakukan perhitungan manual nilai confusion matrix untuk mengkonfirmasi keakuratan hasil tersbut.

Tabel 3. 12 *Confusion Matrix*

	Predicted Positive (0)	*Predicted Negative (1)*
Actual Positive (0)	7317	0
Actual Negative(1)	33	7284



Gambar 3. 6 *Confusion Matrix Random Forest*

$$Accuracy = \frac{7284 + 7317}{7284 + 7317 + 0 + 33} = 0.9977 = 99.77\%$$

Setelah dilakukan proses perhitungan manual pada tiap parameter, didapatkan hasil yang sama dengan perhitungan menggunakan *Python*. Hasil ini menunjukkan kesesuaian yang sangat baik dengan perhitungan manual, mengindikasikan bahwa model memiliki kemampuan yang kuat dalam menerapkan data latih ke data uji dan mampu mengklasifikasikan data dengan akurat. Hal ini menegaskan validitas metode yang digunakan dalam penelitian ini, serta memberikan kepercayaan tambahan bahwa algoritma yang diterapkan dapat diandalkan. Kesesuaian antara hasil perhitungan manual dan hasil dari implementasi *Python* juga menunjukkan bahwa tidak ada kesalahan signifikan dalam proses pemrograman atau implementasi algoritma, serta pemilihan fitur telah dilakukan dengan tepat.

3.1.7 Perbandingan Hasil

Dalam melakukan perbandingan antara dua model *Random Forest*, di mana satu menggunakan seleksi fitur *ANOVA* dan yang lainnya tidak, fokus pada perbedaan akurasi memberikan pandangan yang penting dalam mengevaluasi efektivitas seleksi fitur dalam meningkatkan kinerja model. Hasil yang signifikan dapat mengindikasikan bahwa seleksi fitur mampu mengidentifikasi subset fitur yang lebih

relevan, sehingga memungkinkan model untuk mempelajari pola yang lebih penting dalam data. Hal ini dapat menghasilkan prediksi yang lebih akurat dan efisien.

Tabel 3. 13 Perbandingan Hasil Akurasi Pengujian *Random Forest*

Fold	*Random Forest* (Tanpa *ANOVA*)	*Random Forest (*Dengan *ANOVA*)	Status
1	97.95%	99.66%	Naik
2	98.84%	99.80%	Naik
3	98.91%	99.86%	Naik
4	98.91%	99.93%	Naik
5	98.56%	99.66%	Naik
6	99.18%	99.80%	Naik
7	99.38%	99.80%	Naik
8	98.91%	99.52%	Naik
9	99.18%	99.86%	Naik
10	98.50%	99.86%	Naik

Tabel 3. 14 Perbandingan Hasil Akurasi Rata-Rata *Random Forest*

Random Forest (Tanpa *ANOVA*)	*Random Forest (*Dengan *ANOVA*)
98.83%	99.77%

Berdasarkan tabel perbandingan akurasi antara model *Random Forest* (RF) dengan dan tanpa seleksi fitur *ANOVA* pada tabel 3.13, terlihat bahwa model dengan seleksi fitur *ANOVA* cenderung memberikan akurasi yang lebih tinggi dalam mengklasifikasi. Dari 10 lipatan (folds) pada *cross validation*, model *RF* dengan seleksi fitur *ANOVA* konsisten menunjukkan peningkatan akurasi dibandingkan dengan model *RF* tanpa seleksi fitur *ANOVA*. Peningkatan akurasi yang dicapai sebesar 0.94% menunjukkan bahwa seleksi fitur *ANOVA* efektif dalam meningkatkan kinerja model Random Forest.

3.2 Pembahasan

Penelitian ini menggunakan data Stunting Kota Samarinda periode tahun 2023, data yang didapat akan melalalu beberapa tahapan pengolahan mulai dari data *selection*, data *cleaning*, data *integration* dan data *balancing*. Tahapan tersebut dilakukan agar data yang akan digunakan dalam klasifikasi memiliki kualitas terbaik karena data yang diperolah meiliki ketidakseimbangan kelas maka dilakukanlah teknik oversampling menggunakan *SMOTE* sedangkan untuk membagi datanya menjadi data latih dan data uji dilakukan dengan menggunakan *K-fold Crross Validation* dengan nilai K sebesar sepuluh.

a) Seleksi fitur *ANOVA* pada dataset Stunting Kota Samarinda mengidentifikasi fitur ZS TB/U sebagai yang paling berpengaruh terhadap hasil klasifikasi dengan nilai F sebesar 3091, diikuti oleh ZS BB/U dengan nilai F 975, dan BB/U dengan nilai F 737. Fitur-fitur ini memiliki pengaruh signifikan dalam membedakan berbagai kelas dalam dataset stunting. Sementara itu, fitur-fitur seperti Tinggi, Berat, dan LiLA juga menunjukkan pengaruh yang cukup besar, menduduki peringkat 4 hingga 6. Penelitian oleh K. M. Rajabi dkk. (2023) yang menggunakan metode *Relief* juga menunjukan

pentingnya fitur Tinggi, yang teridentifikasi sebagai salah satu dari dua fitur terbaik yang meningkatkan akurasi *k-nearest neighbor* (KNN*)* menjadi 98,16%. Selanjutnya, Yunus dkk (2023) menzggunakan metode *Backward Elimination* (BE) berhasil mengidentifikasi 2 fitur terbaik, yaitu Tinggi dan Berat. Fitur-fitur ini berkontribusi meningkatkan akurasi model *Naive Bayes* (NB*)* dari 53,50% menjadi 92,54%. Dengan demikian, penggunaan seleksi fitur *ANOVA* dalam studi ini tidak hanya mengonfirmasi keefektifan atribut Tinggi dan Berat yang telah diakui dalam penelitian lain, tetapi juga meningkatkan performa model *Random Forest* dalam klasifikasi stunting.

Tabel 3. 15 Persamaan Hasil Seleksi Fitur Dengan Penelitian Lain

Hasil Seleksi Fitur *ANOVA* (*RF – ANOVA*)	Penelitian K. M. Rajabi dkk. (2023) (KNN – *Relief)*	Penelitian Yunus dkk. (2023) (*NB – BE)*
ZS TB/U		
ZS BB/U		
BB/U		
Tinggi	√	√
Berat		√
LiLA		

b) Penggunaan model Random Forest yang diintegrasikan dengan teknik oversampling SMOTE berhasil mencapai akurasi rata-rata 98.83%, tanpa menggunakan seleksi fitur. Teknik SMOTE diimplementasikan untuk mengatasi ketidakseimbangan kelas yang signifikan dalam dataset stunting, yang memungkinkan model untuk belajar dari distribusi kelas yang lebih seimbang. Kemudian, dengan menambahkan seleksi fitur menggunakan metode ANOVA terjadi peningkatan akurasi yang signifikan hingga mencapai 99.77%. Peningkatan sebesar 0.94% ini menunjukkan bahwa integrasi seleksi fitur ANOVA dapat efektif meningkatkan efisiensi dan efektivitas model Random Forest dalam klasifikasi stunting, begitupun pada penelitian terkait stunting yang dilakukan oleh Nugroho et al., (2022) yang menggkombinasikan ANOVA dengan algorima KNN dan Decision tree dimana ANOVA terbukti mampu menaikan akurasi sebesar 20% untuk KNN dan 39% untuk Decision Tree dan pada penelitan Mogarampalli et al., (2022) yang menggunakan algoritma kalsifikasi seperti SVM, KNN, Random Forest, Naive Bayes, logistic regression, Decision Tree terbutki mampu meraih akurasi diatas 90% ketika dikombinasikan dengan seleksi fitur ANOVA. Seleksi fitur membantu model mengidentifikasi dan memprioritaskan variabel yang paling berpengaruh, sehingga meningkatkan akurasi klasifikasi meskipun data tetap diolah dengan teknik SMOTE untuk memastikan keseimbangan antar kelas.

BAB IV

KESIMPULAN DAN SARAN

4.1 Kesimpulan

Berdasarkan penelitian yang telah dilakukan makan dapat ditarik beberapa kesimpulan sebagai berikut:

a) Hasil dari penerapan seleksi fitur dengan menggunakan *ANOVA* pada data stunting Kota Samarinda didapat 6 fitur terpilih yang memiliki pengaruh paling signifikan terhadap klasifikasi stunting yaitu ZS TB/U, ZS BB/U, BB/U, Tinggi, Berat, LiLA.
b) Dari Hasil pengujian penerapan metode oversampling *SMOTE* dan seleksi fitur *ANOVA* terbukti mampu meningkatkan akurasi sebesar 0.94% pada algoritma *Random Forest*, dimana sebelum diterapkan *ANOVA* akurasi yang didapatkan 98.83% dan setelah menerapkan seleksi fitur pada algoritma Random Forest akurasinya menjasi 99.77%. Hal ini membuktikan bahwa penerapan metode seleksi fitur *ANOVA* terhadap metode *Random Forest* terbukti mampu meningkatkan akurasi algoritma tersebut.

4.2 Saran

Berikut beberapa saran yang dapat diberikan berdasarkan hasil penelitian ini untuk penelitian selanjutnya dan pengembangan praktis terkait masalah stunting:

a) Penerapan pada Data Beragam, penelitian ini berhasil di Kota Samarinda dengan dataset spesifik. Untuk menggeneralisasi efektivitas model *Random Forest* dan teknik seleksi fitur *ANOVA*, disarankan untuk menerapkan metode yang sama pada dataset dari berbagai wilayah atau negara dengan kondisi sosioekonomi yang berbeda. Hal ini dapat membantu dalam memahami dinamika stunting di lingkungan yang beragam.
b) Eksplorasi Metode Seleksi Fitur Lainnya, meskipun *ANOVA* telah terbukti efektif dalam penelitian ini, penelitian mendatang bisa mempertimbangkan penggunaan metode seleksi fitur lain seperti *Chi-Square*, *atau Information Gain* untuk membandingkan efektivitas dalam meningkatkan kinerja model klasifikasi stunting.
c) Peningkatan Skala dan Frekuensi Pengambilan Data, meningkatkan frekuensi dan variasi pengambilan data dapat membantu dalam menghasilkan model yang lebih akurat dan dapat diandalkan. Data yang lebih komprehensif dari waktu ke waktu juga bisa memberikan insight lebih dalam mengenai tren dan pola stunting.
d) Fokus pada Implementasi di Lapangan. Mengingat tingkat akurasi tinggi dari model yang dikembangkan, ada potensi untuk mengimplementasikan sistem ini di fasilitas kesehatan atau program intervensi stunting. Pengembangan aplikasi mobile atau web untuk skrining cepat stunting dapat sangat bermanfaat untuk diterapkan di Dinas Kesehatan Kota Samarinda
e) Penggunaan Teknik Optimasi: Untuk lebih meningkatkan kinerja model, disarankan untuk menerapkan teknik optimasi seperti *Grid Search* atau *Random Search* untuk menemukan kombinasi parameter terbaik pada model *Random Forest*. Selain itu, teknik optimasi seperti *Bayesian Optimization* atau *Particle Swarm Optimization* (PSO) juga dapat digunakan untuk lebih meningkatkan akurasi dan efisiensi model.

DAFTAR RUJUKAN

Apriyani, H., Kurniati, K., 2020. Perbandingan Metode Naïve Bayes Dan Support Vector Machine Dalam Klasifikasi Penyakit Diabetes Melitus. J. Inf. Technol. Ampera 1, 133–143. https://doi.org/10.51519/journalita.volume1.isssue3.year2020.page133-143

Bengnga, A., Ishak, R., 2022. Implementasi Seleksi Fitur Klasifikasi Waktu Kelulusan Mahasiswa Menggunakan Correlation Matrix with Heatmap. Jambura J. Electr. Electron. Eng. 4, 169–174. https://doi.org/10.37905/jjeee.v4i2.14403

Chilyabanyama, O.N., Chilengi, R., Simuyandi, M., Chisenga, C.C., Chirwa, M., Hamusonde, K., Saroj, R.K., Iqbal, N.T., Ngaruye, I., Bosomprah, S., 2022. Performance of Machine Learning Classifiers in Classifying Stunting among Under-Five Children in Zambia. Children 9. https://doi.org/10.3390/children9071082

Cindy, M.A., 2023. Daftar Prevalensi Balita Stunting di Indonesia pada 2022. Katadata Media Netw. 1–11.

Cindy Mutia Annur, 2023. Calon Ibu Kota Baru, Bagaimana Angka Balita Stunting di Wilayah di Kalimantan Timur? Layanan konsumen & Kesehatan. Katadata.Co.Id 2023–2024.

Gebeye, L.G., Dessie, E.Y., Yimam, J.A., 2023. Predictors of micronutrient deficiency among children aged 6–23 months in Ethiopia: a machine learning approach. Front. Nutr. 10, 1–13. https://doi.org/10.3389/fnut.2023.1277048

Hafid, H., 2023. Penerapan K-Fold Cross Validation untuk Menganalisis Kinerja Algoritma K-Nearest Neighbor pada Data Kasus Covid-19 di Indonesia. J. Math. 6, 161–168.

Hakimah, M., Prabiantissa, C.N., Rozi, N.F., Yamani, L.N., Puspitasari, I., 2022. Determination of Relevant Feature Combinations for Detection Stunting Status of Toddlers. 2022 5th Int. Semin. Res. Inf. Technol. Intell. Syst. ISRITI 2022 324–329. https://doi.org/10.1109/ISRITI56927.2022.10053069

Hasan, K.A., Al Mehedi Hasan, M., 2020. Classification of Parkinson's Disease by Analyzing Multiple Vocal Features Sets. 2020 IEEE Reg. 10 Symp. TENSYMP 2020 758–761. https://doi.org/10.1109/TENSYMP50017.2020.9230842

Hemo, S.A., Rayhan, M.I., 2021. Classification tree and random forest model to predict under-five malnutrition in Bangladesh. Biometrics Biostat. Int. J. 10, 116–123. https://doi.org/10.15406/bbij.2021.10.00337

Kemal Musthafa Rajabi, Witanti, W., Rezki Yuniarti, 2023. Penerapan Algoritma K-Nearest Neighbor (KNN) Dengan Fitur Relief-F Dalam Penentuan Status Stunting. Innov. J. Soc. Sci. Res. 3, 3555–3568.

Luo, H., Pan, X., Wang, Q., Ye, S., Qian, Y., 2019. Logistic regression and random forest for effective imbalanced classification. Proc. - Int. Comput. Softw. Appl. Conf. 1, 916–917. https://doi.org/10.1109/COMPSAC.2019.00139

M. M. Mafa'atiih, 2020. MPLEMENTASI ARTIFICIAL INTELLIGENCE UNTUK MEMPREDIKSI HARGA SEWA AIRBNB MENGGUNAKAN METODE RANDOM FOREST DAN PENERAPAN WEB APPLICATION MENGGUNAKAN FLASK 83–99. https://doi.org/10.1007/978-3-030-29761-9_6

Mogarampalli, R., Chengalvala, G., Challapuram, S.S., Karri, N., Malik, J., Singh, A.K., 2022. Stroke Disease Classification with help of a ANOVA and Repeated StratiFiedKFold. 2022

IEEE North Karnataka Subsect. Flagsh. Int. Conf. NKCon 2022 1–5. https://doi.org/10.1109/NKCon56289.2022.10127074

Nugroho, A., Warnars, H.L.H.S., Gaol, F.L., Matsuo, T., 2022. Trend of Stunting Weight for Infants and Toddlers Using Decision Tree. IAENG Int. J. Appl. Math. 52.

Pratiwi, R., Sari, R.S., Ratnasari, F., 2021. Dampak status gizi pendek (stunting) terhadap prestasi belajar: A literature review. J. Ilm. Ilmu Keperawatan 12, 10–23.

Santoso, N., Wibowo, W., Himawati, H., 2019. Integration of synthetic minority oversampling technique for imbalanced class. Indones. J. Electr. Eng. Comput. Sci. 13, 102–108. https://doi.org/10.11591/ijeecs.v13.i1.pp102-108

Siswa, T.A.Y., 2023. Data Mining - Mengupas Tuntas Analisis Data Dengan Metode Klasifikasi Hingga Deployment Aplikasi Menggunakan Python.

Sutarmi, S., Warijan, W., Indrayana, T., B, D.P.P., Gunawan, I., 2023. Machine Learning Model For Stunting Prediction. J. Heal. Sains 4, 10–23. https://doi.org/10.46799/jhs.v4i9.1073

Talukder, A., Ahammed, B., 2020. Machine learning algorithms for predicting malnutrition among under-five children in Bangladesh. Nutrition 78, 110861. https://doi.org/10.1016/j.nut.2020.110861

Yoga Siswa, T.A., 2023. Komparasi Optimasi Chi-Square, CFS, Information Gain dan ANOVA dalam Evaluasi Peningkatan Akurasi Algoritma Klasifikasi Data Performa Akademik Mahasiswa. Inform. Mulawarman J. Ilm. Ilmu Komput. 18, 62. https://doi.org/10.30872/jim.v18i1.11330

Yunus, M., Muhammad Kunta Biddinika, Fadlil, A., 2023. Optimasi Algoritma Naïve Bayes Menggunakan Fitur Seleksi Backward Elimination untuk Klasifikasi Prevalensi Stunting. Decod. J. Pendidik. Teknol. Inf. 3, 278–285. https://doi.org/10.51454/decode.v3i2.188

LAMPIRAN

Lampiran 1 Tampilan Dataset Stunting Kota Samarinda

No	Nama	JK	BB Lahir	TB Lahir	Prov	Kab/Kota	Kec	Pukesmas	Desa/Kel	Posyandu	RT	RW	Tanggal Pengukuran	Berat	Tinggi	LiLA	BB/U	ZS BB/U	TB/U	ZS TB/U	BB/TB	ZS BB/TB	Naik Berat Badan	Jml Vit A
1 .	DIMAS ADITYA	L	3.5	49	KALIMANTAN TIMUR	SAMARINDA	SUNGAI PINANG	REMAJA	TEMINDUNG PERMAI	PULAU INDAH			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		2023-01-02	12	94	0	Kurang	-2.25	Pendek	-2.09	Gizi Baik	-1.46	O	
3 .	M AL FATIH	L	2.8	49	KALIMANTAN TIMUR	SAMARINDA	SUNGAI PINANG	REMAJA	TEMINDUNG PERMAI	PULAU INDAH			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			2023-01-02	6.3	63.5	0	Berat Badan Normal	-0.31	Normal	0.42	Gizi Baik	-0.74	O	
5 .	GIUNIA QAMELA	P	3.5	50	KALIMANTAN TIMUR	SAMARINDA	SUNGAI PINANG	REMAJA	TEMINDUNG PERMAI	PULAU INDAH	34		2023-01-02	10.6	72.5	0	Risiko Lebih	1.71	Normal	0.23	Gizi Lebih	2.14	O	
....	....	..	.....	.....	.....	.....	.....	.....	.....	....	....	...	.....	.....	.....	.....	.....	.....	.....	.....	....	.....	....	....
150462	ADNAN IRAGUSTI	L	3	50	KALIMANTAN TIMUR	SAMARINDA	SAMARINDA UTARA	LEMPAKE	LEMPAKE	SRI REJEKI			2023-12-30	3	50		Berat Badan Normal	-0.73	Normal	0.06	Gizi Baik	-1.19	-	
150463	SIENA AL RAISHA RAMADHANI	P	3	52	KALIMANTAN TIMUR	SAMARINDA	SUNGAI PINANG	REMAJA	BANDARA	LESTARI	24		2023-12-13	13	95.5	0	Berat Badan Normal	-1.22	Normal	-1.11	Gizi Baik	-0.85	O	
150464	AFIZAH KHAIRINA	P	2.5	45	KALIMANTAN TIMUR	SAMARINDA	SAMARINDA SEBERANG	MANGKUPALAS	TENUN SAMARINDA	BALO NEGARA	12		2023-12-09	2.5	45		Kurang	-2.03	Pendek	-2.63	Gizi Baik	-0.35	-	
150465	M ARSYA KHOLIF	P	3	49	KALIMANTAN TIMUR	SAMARINDA	SAMARINDA ULU	JUANDA	AIR HITAM	MEKAR SEJAHTERA	32		2023-12-19	3	49		Berat Badan Normal	-1.56	Normal	-1.3	Gizi Baik	-1.04	-	
150466	MUHAMMAD IQBAL	L	2.9	49	KALIMANTAN TIMUR	SAMARINDA	SAMARINDA ILIR	SIDOMULYO	SUNGAI DAMA	RAMANIA	0		2023-12-29	2.9	49		Kurang	-2.97	Pendek	-2.48	Gizi Baik	-1.37	-	

Lampiran 2 Surat Pengantar Pengambilan Data

UMKT
Program Studi
Teknik Informatika
Fakultas Sains dan Teknologi

Universitas Muhammadiyah Kalimantan Timur

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

...ogram Studi S1 Teknik Informatika

...nsyah, S.Kom., M.TI
...DN. 1118019203

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

Lampiran 3 Kartu Bimbingan

KARTU KENDALI BIMBINGAN LAPORAN KARYA ILMIAH

Nama Mahasiswa	:	Ari Ahmad Dhani
NIM	:	2011102441090
Nama Dosen Pembimbing	:	Taghfirul Azhima Yoga Siswa, S.Kom, M.Kom
Judul Penelitian	:	PERBAIKAN AKURASI *RANDOM FOREST* DENGAN *ANOVA* DAN *SMOTE* PADA KLASIFIKASI DATA STUNTING

No	Tanggal	Uraian Pembimbingan	Paraf Dosen
1	22 Januari 2024	Membahas tahapan penelitian skripsi	
2	28 Januari 2024	Mencari paper/artikel rujukan penelitian	
3	5 Februari 2024	Review survei paper untuk mencari permasalahan pada klasifikasi data minning dan mecari topik penelitian yg masih relevan untuk diteliti	
4	16 Februari 2024	Review technical paper dan road maps penelitian, untuk mencari algoritma dan metode untuk mengatasi permasalahan klasifikasi	
5	22 Februari 2024	Mencari paper/artikel rujukan yang sesuai dengan topik penelitian yang telah ditentukan	
6	26 Februari 2024	Penentuan judul penelitian	
7	13 Maret 2024	Pembuatan canvas penelitian untuk disubmit ke prodi	
8	15 Maret 2024	Pengajuan surat permohonan data untuk penelitian	
9	18 April 2024	Revisi proposal bab 1, bab 2 dan perbaikan penulisan format skripsi	

10	24 April 2024	Revisi proposal bab 1 dan bab 2, untuk penyelesaian akhir sebelum disubmit	
11	10 Mai 2024	Pembahasan Penulisam bab 3 dan 4 serta naskah publikasi	
12	17 Mai 2024	Revisi naskah skripsi bab 3 dan bab 4 serta naskah publikasi	
13	3 Juni 2024	Revisi naskah Publikasi Sebelum dipublish	

mengetahui

Dosen Pembimbing

(Taghfirul Azhima Yoga Siswa. S.kom, M.kom)
NIDN. 1118038805

Ketua Program Studi

(Arbansyah, S.kom., M.TI)
NIDN. 1118019203

RIWAYAT HIDUP

Ari Ahmad Dhani, atau biasa disapa dengan sebutan Ari, lahir di jawa tengah tepatny di daerah Demak pada 07 April 2002, Penulis merupakan anak ke-dua dari Bapak Sukari dan Ibu Rumini. Menempuh pendidikan di SDN 002 Loa Kulu tahun 2008 kemudia 2014, SMPN 1 Loa Kulu tahun 2014 – 2017, SMAN 1 Loa Kulu tahun 2017 – 2020. Kemudian penlis tercatat sebagai mahasiswa pada perguruan tinggi swasta Universitas Muhammadiyah Kalimantan Timur pada Fakultas Sains Dan Teknologi jurusan Teknik informatika pada tahun 2020. Saat menjadi mahasiswa penulis pernah melaksanakan perogram magang di Perusahaan pertambangan batu bara yaitu PT. Bukit Baiduri Energi Kutai Kartainegara selama 3 bulan yang dilaksanakan pada semester 7. Demikian deskripsi riwayat hidup yang penulis sampaikan jika terdapat kesalahan atau kekurangan mohon dimaafkan karena kesempurnaan hanya milik Sang Maha pencipta, maka penulis mengharapkan kritik dan saran mengenai skripsi ini.

JADWAL PENELITIAN

No	Jenis Penelitian	Bulan / 2024						
		Jan	Feb	Mar	Apr	Mei	Jun	Jul
	Tahap Pra Penelitian							
1.	Menentukan Judul Penelitian							
2.	Menyusun Rumusan Masalah							
3.	Mencari Data Pendukung							
4.	Menyusun Metode Penelitian							
5.	Menyusun Proposal Penelitian							
6.	Review Desk Simpel							
	Tahap Penelitian							
1.	Pengumpulan Data							
2.	Analisis Data							
3.	Pengujian algoritma *Random Forest*							
	Tahap Akhir Penelitian							
1.	Penyusunan Laporan							
2.	Evaluasi							
3.	Penyempurnaan Laporan							
4.	Seminar Hasil							

SELURUH KODE PYTHON

```
import pandas as pd
import numpy as np

from sklearn.model_selection import train_test_split

import seaborn as sns
import matplotlib.pyplot as plt
plt.style.use('seaborn-darkgrid')
```

```
<ipython-input-1-56fe5db77cb2>:8: MatplotlibDeprecationWarning: The seaborn styles shipped by Matplotlib are deprecated since 3.6, as they no longer correspond to the styles shipped by seaborn. Howeve
  plt.style.use('seaborn-darkgrid')
```

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

```
data = pd.read_csv('datasetfixstunting.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)

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

data.head()
```

```
Total data sebelum penghapusan duplikat: 150465
Total data setelah penghapusan duplikat: 34199
```

	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
143850	A ALVIN	L	18.06	NaN	18.0	Risiko Lebih	1.19	Normal	0.41	Risiko Gizi Lebih	0.07	N	NaN	2023-12-16
88655	A FADLAN	L	11.06	83.0	0.0	Berat Badan Normal	-0.08	Normal	-0.97	Gizi Baik	0.04	-	NaN	2023-07-10
128459	A FARIS WICAKSONO	L	9.07	78.0	16.0	Berat Badan Normal	-1.75	Pendek	-2.84	Gizi Baik	-0.47	O	1.0	2023-10-06
119828	A FATHAN	L	15.00	107.0	17.0	Berat Badan Normal	-1.08	Normal	0.14	Gizi Baik	-1.83	T	1.0	2023-10-23
20569	A FAUJAN	L	14.00	100.0	0.0	Berat Badan Normal	-0.85	Normal	-0.16	Gizi Baik	-1.14	O	1.0	2023-02-07

```
missing_values = data.isna().sum()

# Menampilkan jumlah missing values untuk setiap kolom
print("Jumlah missing values untuk setiap kolom:")
print(missing_values)
```

```
Jumlah missing values untuk setiap kolom:
Nama                  1
JK                    0
Berat               156
Tinggi            11885
LiLA               5623
BB/U                  0
ZS BB/U              39
TB/U               2640
ZS TB/U            2462
BB/TB              2627
ZS BB/TB           2453
Naik Berat Badan      0
Jml Vit A         21540
Tanggal Pengukuran    0
dtype: int64
```

```
data.drop_duplicates(inplace=True)
data.dropna(inplace=True)

pd.set_option('display.max_rows', 10)  # Atur jumlah maksimum baris yang ditampilkan
data
```

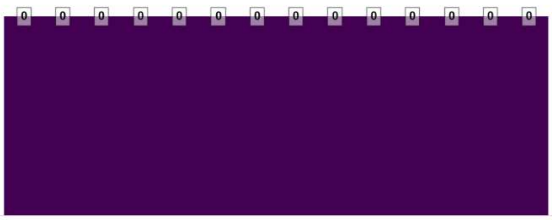
	Nama	JK	Berat	Tinggi	LiLA	BB/U	ZS BB/U	TB/U	ZS TB/U	BB/TB	ZS BB/TB	Naik Berat Badan	Jml Vit A	Tanggal Pengukuran
128459	A FARIS WICAKSONO	L	9.07	78.0	16.0	Berat Badan Normal	-1.75	Pendek	-2.840000	Gizi Baik	-0.47	O	1.0	2023-10-06
119828	A FATHAN	L	15.00	107.0	17.0	Berat Badan Normal	-1.08	Normal	0.140000	Gizi Baik	-1.83	T	1.0	2023-10-23
20569	A FAUJAN	L	14.00	100.0	0.0	Berat Badan Normal	-0.85	Normal	-0.160000	Gizi Baik	-1.14	O	1.0	2023-02-07
31702	A MISHAEL	P	15.00	103.0	0.0	Berat Badan Normal	0.06	Normal	1.050000	Gizi Baik	-0.80	O	1.0	2023-02-09
113633	A ZHAFIR KAMIL	L	14.00	102.0	0.0	Berat Badan Normal	-1.91	Normal	-1.490000	Gizi Baik	-1.59	O	1.0	2023-10-14
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
27525	yelmi	L	17.04	111.0	0.0	Berat Badan Normal	-0.04	Normal	0.060417	Gizi Baik	-0.93	N	1.0	2023-02-27
125185	yumna	P	13.07	100.0	14.0	Berat Badan Normal	-1.07	Normal	-0.450000	Gizi Baik	-1.18	O	1.0	2023-10-14
121521	ziaan aqilla karno	L	14.05	102.0	0.0	Berat Badan Normal	-1.71	Normal	-1.620000	Gizi Baik	-1.15	T	1.0	2023-10-21
22242	zubair	L	17.03	107.0	0.0	Berat Badan Normal	-0.14	Normal	-0.130000	Gizi Baik	-0.13	O	1.0	2023-02-20
112022	zulkifli abdi	L	15.06	102.0	0.0	Berat Badan Normal	-0.59	Normal	-0.690000	Gizi Baik	-0.25	N	1.0	2023-10-03

8059 rows × 14 columns

```
[ ] missing_values2 = data.isnull().sum()

    # Menampilkan jumlah missing values untuk setiap kolom
    print("Jumlah missing values untuk setiap kolom:")
    print(missing_values2)
```

```
Jumlah missing values untuk setiap kolom:
Nama                0
JK                  0
Berat               0
Tinggi              0
LiLA                0
                   ..
BB/TB               0
ZS BB/TB            0
Naik Berat Badan    0
Jml Vit A           0
Tanggal Pengukuran  0
Length: 14, dtype: int64
```

```
[ ] # Buat heatmap untuk menampilkan nilai-nilai yang hilang
    plt.figure(figsize=(10, 6))  # Atur ukuran plot
    sns.heatmap(data.isna(), yticklabels=False, cbar=False, cmap="viridis")

    # Hitung total nilai yang hilang tiap kolom
    missing_values_per_column = data.isna().sum()

    # Annotate dengan jumlah nilai yang hilang tiap kolom
    for i in range(len(missing_values_per_column)):
        plt.text(i + 0.5, -5, f'{missing_values_per_column[i]}', ha='center', va='center',
                 color='black', fontsize=12, fontweight='bold', bbox=dict(facecolor='white', alpha=0.5))

    plt.show()
```




```
[ ] # Menghapus Kolom Yang Tidak Digunakan
    data.drop(columns=["Nama", "Tanggal Pengukuran"], inplace=True)
```

```
[ ] pd.set_option('display.max_rows', 10)  # Atur jumlah maksimum baris yang ditampilkan

    # Menampilkan DataFrame
    data
```

	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
128459	L	9.07	78.0	16.0	Berat Badan Normal	-1.75	Pendek	-2.840000	Gizi Baik	-0.47	O	1.0
119828	L	15.00	107.0	17.0	Berat Badan Normal	-1.08	Normal	0.140000	Gizi Baik	-1.83	T	1.0
20569	L	14.00	100.0	0.0	Berat Badan Normal	-0.85	Normal	-0.160000	Gizi Baik	-1.14	O	1.0
31702	P	15.00	103.0	0.0	Berat Badan Normal	0.06	Normal	1.050000	Gizi Baik	-0.80	O	1.0
113633	L	14.00	102.0	0.0	Berat Badan Normal	-1.91	Normal	-1.490000	Gizi Baik	-1.59	O	1.0
...	...	...	...	...	...	...	...	...	...	...	...	...
27525	L	17.04	111.0	0.0	Berat Badan Normal	-0.04	Normal	0.060417	Gizi Baik	-0.93	N	1.0
125185	P	13.07	100.0	14.0	Berat Badan Normal	-1.07	Normal	-0.450000	Gizi Baik	-1.18	O	1.0
121521	L	14.05	102.0	0.0	Berat Badan Normal	-1.71	Normal	-1.620000	Gizi Baik	-1.15	T	1.0
22242	L	17.03	107.0	0.0	Berat Badan Normal	-0.14	Normal	-0.130000	Gizi Baik	-0.13	O	1.0
112022	L	15.06	102.0	0.0	Berat Badan Normal	-0.59	Normal	-0.690000	Gizi Baik	-0.25	N	1.0

8059 rows × 12 columns

```
[ ] from sklearn.preprocessing import LabelEncoder

    # Buat instance LabelEncoder
    encoder = LabelEncoder()

    # Kolom yang perlu di-encode
    columns_to_encode = ['JK', 'BB/U', 'BB/TB', 'Naik Berat Badan']

    # Loop melalui setiap kolom yang perlu di-encode dan terapkan LabelEncoder
    for column in columns_to_encode:
        data[column] = encoder.fit_transform(data[column])

    # Menampilkan hasilnya untuk memastikan encoding telah berhasil
    pd.set_option('display.max_rows', 10)  # Atur jumlah maksimum baris yang ditampilkan

    # Menampilkan DataFrame
    data
```

	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
128459	0	9.07	78.0	16.0	0	-1.75	Pendek	-2.840000	0	-0.47	2	1.0
119828	0	15.00	107.0	17.0	0	-1.08	Normal	0.140000	0	-1.83	3	1.0
20569	0	14.00	100.0	0.0	0	-0.85	Normal	-0.160000	0	-1.14	2	1.0
31702	1	15.00	103.0	0.0	0	0.06	Normal	1.050000	0	-0.80	2	1.0
113633	0	14.00	102.0	0.0	0	-1.91	Normal	-1.490000	0	-1.59	2	1.0
...	...	...	...	...	...	...	...	...	...	...	...	...
27525	0	17.04	111.0	0.0	0	-0.04	Normal	0.060417	0	-0.93	1	1.0
125185	1	13.07	100.0	14.0	0	-1.07	Normal	-0.450000	0	-1.18	2	1.0
121521	0	14.05	102.0	0.0	0	-1.71	Normal	-1.620000	0	-1.15	3	1.0
22242	0	17.03	107.0	0.0	0	-0.14	Normal	-0.130000	0	-0.13	2	1.0
112022	0	15.06	102.0	0.0	0	-0.59	Normal	-0.690000	0	-0.25	1	1.0

8059 rows × 12 columns

```
# Ubah TB/U
data['TB/U'] = data['TB/U'].replace({'Normal': 0, 'Tinggi': 0, 'Pendek': 1, 'Sangat Pendek': 1})
```

```
pd.set_option('display.max_rows', 10)  # Atur jumlah maksimum baris yang ditampilkan

data
```

	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
128459	0	9.07	78.0	16.0	0	-1.75	1	-2.840000	0	-0.47	2	1.0
119828	0	15.00	107.0	17.0	0	-1.08	0	0.140000	0	-1.83	3	1.0
20569	0	14.00	100.0	0.0	0	-0.85	0	-0.160000	0	-1.14	2	1.0
31702	1	15.00	103.0	0.0	0	0.06	0	1.050000	0	-0.80	2	1.0
113633	0	14.00	102.0	0.0	0	-1.91	0	-1.490000	0	-1.59	2	1.0
...	...	...	...	...	...	...	...	...	...	...	...	...
27525	0	17.04	111.0	0.0	0	-0.04	0	0.060417	0	-0.93	1	1.0
125185	1	13.07	100.0	14.0	0	-1.07	0	-0.450000	0	-1.18	2	1.0
121521	0	14.05	102.0	0.0	0	-1.71	0	-1.620000	0	-1.15	3	1.0
22242	0	17.03	107.0	0.0	0	-0.14	0	-0.130000	0	-0.13	2	1.0
112022	0	15.06	102.0	0.0	0	-0.59	0	-0.690000	0	-0.25	1	1.0

8059 rows × 12 columns

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 8059 entries, 128459 to 112022
Data columns (total 12 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   JK                8059 non-null   int64
 1   Berat             8059 non-null   float64
 2   Tinggi            8059 non-null   float64
 3   LiLA              8059 non-null   float64
 4   BB/U              8059 non-null   int64
 5   ZS BB/U           8059 non-null   float64
 6   TB/U              8059 non-null   int64
 7   ZS TB/U           8059 non-null   float64
 8   BB/TB             8059 non-null   int64
 9   ZS BB/TB          8059 non-null   float64
 10  Naik Berat Badan  8059 non-null   int64
 11  Jml Vit A         8059 non-null   float64
dtypes: float64(7), int64(5)
memory usage: 1.1 MB
```

```
x = data.drop(['TB/U'], axis=1)
y = data['TB/U']
```

```
pd.set_option('display.max_rows', 10)

data
```

	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
128459	0	9.07	78.0	16.0	0	-1.75	1	-2.840000	0	-0.47	2	1.0
119828	0	15.00	107.0	17.0	0	-1.08	0	0.140000	0	-1.83	3	1.0
20569	0	14.00	100.0	0.0	0	-0.85	0	-0.160000	0	-1.14	2	1.0
31702	1	15.00	103.0	0.0	0	0.06	0	1.050000	0	-0.80	2	1.0
113633	0	14.00	102.0	0.0	0	-1.91	0	-1.490000	0	-1.59	2	1.0
...	...	...	...	...	...	...	...	...	...	...	...	...
27525	0	17.04	111.0	0.0	0	-0.04	0	0.060417	0	-0.93	1	1.0
125185	1	13.07	100.0	14.0	0	-1.07	0	-0.450000	0	-1.18	2	1.0
121521	0	14.05	102.0	0.0	0	-1.71	0	-1.620000	0	-1.15	3	1.0
22242	0	17.03	107.0	0.0	0	-0.14	0	-0.130000	0	-0.13	2	1.0
112022	0	15.06	102.0	0.0	0	-0.59	0	-0.690000	0	-0.25	1	1.0

8059 rows × 12 columns

```
from sklearn.preprocessing import MinMaxScaler

scaler = MinMaxScaler()
x = scaler.fit_transform(x)
```

```
x
```

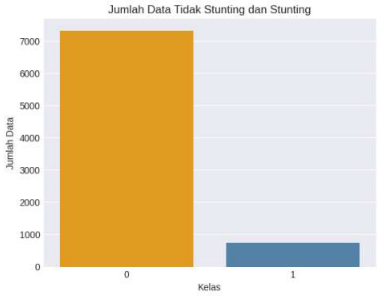
```
array([[0.        , 0.29523676, 0.4       , ..., 0.46083333, 0.66666667,
        0.        ],
       [0.        , 0.49491407, 0.81428571, ..., 0.3475    , 1.        ,
        0.        ],
       [0.        , 0.46124167, 0.71428571, ..., 0.405     , 0.66666667,
        0.        ],
       ...,
       [0.        , 0.46292529, 0.74285714, ..., 0.40416667, 1.        ,
        0.        ],
       [0.        , 0.56326903, 0.81428571, ..., 0.48916667, 0.66666667,
        0.        ],
```

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

```
TB/U
0    7317
1     742
Name: count, dtype: int64
```

```
[ ] class_data = data['TB/U'].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()
```

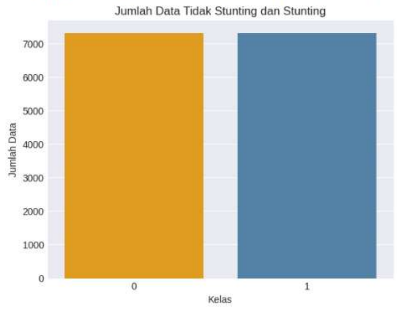
```
Jumlah data kelas Stunting: 742
Jumlah data kelas Tidak Stunting: 7317
<ipython-input-19-1da1770b606e>:7: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

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




```
[ ] x = data.drop(['TB/U'],axis=1)
    y = data['TB/U']
    from imblearn.over_sampling import SMOTE
    sm = SMOTE(random_state=42, sampling_strategy=1)
    X_imb, y_imb = sm.fit_resample(x, y)
```

```
[ ] 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.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()
```

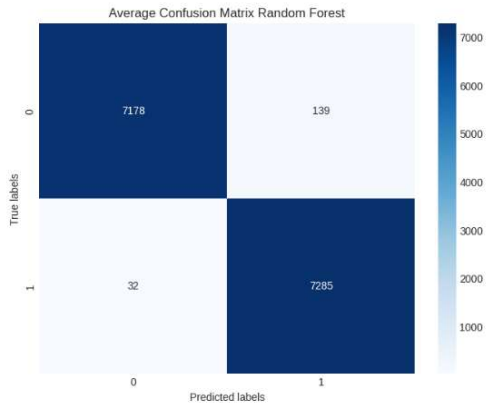
```
Jumlah data kelas stunting: 7317
Jumlah data kelas tidak stunting: 7317
<ipython-input-21-3c98e40829bf>:7: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

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



```python
from sklearn.model_selection import cross_val_score, StratifiedKFold
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score
import numpy as np

# Perform cross-validation on the Random Forest classifier with K=10
clf = RandomForestClassifier(max_depth=1, random_state=42)

# Kode kfold Cross Validation
cv = StratifiedKFold(n_splits=10, shuffle=True, random_state=42)

# Lists to store evaluation metrics for each fold
conf_matrices = []
accuracies = []
precisions = []
recalls = []
f1_scores = []

for train_index, test_index in cv.split(X_imb, y_imb):
    X_train, X_test = X_imb.iloc[train_index], X_imb.iloc[test_index]
    y_train, y_test = y_imb.iloc[train_index], y_imb.iloc[test_index]

    # Train the model
    clf.fit(X_train, y_train)

    # Predict on the test set
    y_pred = clf.predict(X_test)

    # Calculate evaluation metrics
    conf_matrices.append(confusion_matrix(y_test, y_pred))
    accuracies.append(accuracy_score(y_test, y_pred))

# Print the evaluation metrics for each fold
for i, (conf_matrix, accuracy) in enumerate(zip(conf_matrices, accuracies), 1):
    print(f"Fold-{i}: Accuracy={accuracy}")

# Print the average evaluation metrics
print('')
print('Hasil Rata Rata')
print(f"Average Accuracy: {np.mean(accuracies)}")

# Calculate and print the average confusion matrix
avg_conf_matrix = sum(conf_matrices)
print("Average Confusion Matrix:")
print(avg_conf_matrix)
```

```
Fold-1: Accuracy=0.979508196721311
Fold-2: Accuracy=0.9883879781420765
Fold-3: Accuracy=0.9890710382513661
Fold-4: Accuracy=0.9890710382513661
Fold-5: Accuracy=0.9856459330143541
Fold-6: Accuracy=0.9917976760082023
Fold-7: Accuracy=0.9938482570061518
Fold-8: Accuracy=0.9890635680109364
Fold-9: Accuracy=0.9917976760082023
Fold-10: Accuracy=0.9849624060150376

Hasil Rata Rata
Average Accuracy: 0.9883153767429006
Average Confusion Matrix:
[[7178  139]
 [  32 7285]]
```

```python
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

# Calculate the average confusion matrix
avg_conf_matrix = sum(conf_matrices)

# Create a heatmap
plt.figure(figsize=(8, 6))
sns.heatmap(avg_conf_matrix, annot=True, fmt=".0f", cmap="Blues")

plt.xlabel("Predicted labels")
plt.ylabel("True labels")
plt.title("Average Confusion Matrix Random Forest")
plt.show()
```

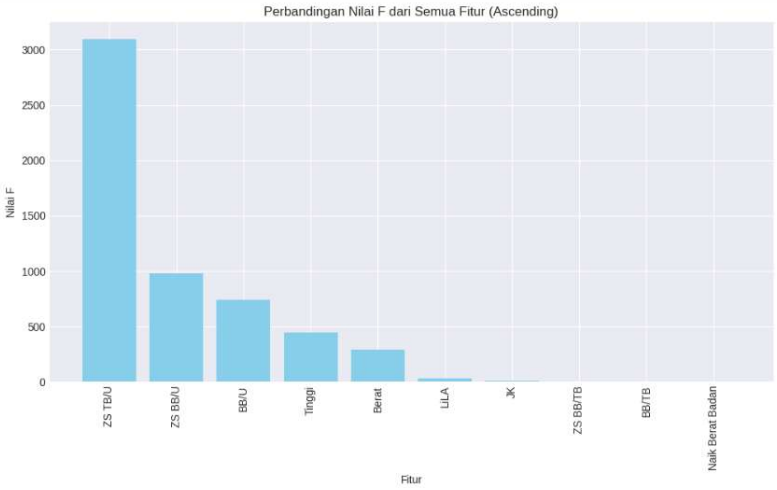


```
data
```

	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
128459	0	9.07	78.0	16.0	0	-1.75	1	-2.840000	0	-0.47	2	1.0
119828	0	15.00	107.0	17.0	0	-1.08	0	0.140000	0	-1.83	3	1.0
20569	0	14.00	100.0	0.0	0	-0.85	0	-0.160000	0	-1.14	2	1.0
31702	1	15.00	103.0	0.0	0	0.06	0	1.050000	0	-0.80	2	1.0
113633	0	14.00	102.0	0.0	0	-1.91	0	-1.490000	0	-1.59	2	1.0
...	...	...	...	...	...	...	...	...	...	...	...	...
27525	0	17.04	111.0	0.0	0	-0.04	0	0.060417	0	-0.93	1	1.0
125185	1	13.07	100.0	14.0	0	-1.07	0	-0.450000	0	-1.18	2	1.0
121521	0	14.05	102.0	0.0	0	-1.71	0	-1.620000	0	-1.15	3	1.0
22242	0	17.03	107.0	0.0	0	-0.14	0	-0.130000	0	-0.13	2	1.0
112022	0	15.06	102.0	0.0	0	-0.59	0	-0.690000	0	-0.25	1	1.0

8059 rows × 12 columns

```
[ ] from sklearn.feature_selection import SelectKBest, f_classif


    # Menggunakan SelectKBest dengan ANOVA untuk memilih fitur terbaik
    selector = SelectKBest(score_func=f_classif)
    X_selected = selector.fit_transform(x, y)

    # Mengambil nilai F dan nama fitur dari SelectKBest
    f_values = selector.scores_
    features = x.columns

    # Membuat DataFrame dari fitur dan nilai F
    import pandas as pd
    df_f = pd.DataFrame({'Feature': features, 'F_Value': f_values})

    # Sorting DataFrame berdasarkan nilai F secara ascending
    df_sorted = df_f.sort_values('F_Value', ascending=False)

    # Membuat diagram bar untuk menampilkan nilai F dari semua fitur
    plt.figure(figsize=(12, 6))
    plt.bar(df_sorted['Feature'], df_sorted['F_Value'], color='skyblue')
    plt.xlabel('Fitur')
    plt.ylabel('Nilai F')
    plt.title('Perbandingan Nilai F dari Semua Fitur (Ascending)')
    plt.xticks(rotation=90)
    plt.show()
```




```
[ ] # Mengatur lebar output konsol
    pd.set_option('display.max_rows', None)
    pd.set_option('display.max_columns', None)
    pd.set_option('display.width', None)

    # Print nilai F dari semua fitur
    print(df_sorted[['Feature', 'F_Value']])
```

```
            Feature      F_Value
6           ZS TB/U  3091.918974
5           ZS BB/U   975.781238
4              BB/U   737.297857
2            Tinggi   441.561417
1             Berat   290.194343
3              LiLA    32.357488
0                JK     8.574745
8           ZS BB/TB     4.964478
7              BB/TB     4.669220
9   Naik Berat Badan     3.936491
10         Jml Vit A          NaN
```

```
[ ] # Mengembalikan pengaturan lebar output konsol seperti semula
    pd.reset_option('display.max_rows')
    pd.reset_option('display.max_columns')
    pd.reset_option('display.width')
```

```
# Seleksi Fitur P-Value < 0,1
x = data.drop(['Naik Berat Badan','BB/TB','TB/U','Jml Vit A','ZS BB/TB','JK'],axis=1)
y = data['TB/U']
from imblearn.over_sampling import SMOTE
sm = SMOTE(random_state=42, sampling_strategy=1)
X_Anv, y_Anv = sm.fit_resample(x, y)
```

```
[ ] x
```

	Berat	Tinggi	LiLA	BB/U	ZS BB/U	ZS TB/U
128459	9.07	78.0	16.0	0	-1.75	-2.840000
119828	15.00	107.0	17.0	0	-1.08	0.140000
20569	14.00	100.0	0.0	0	-0.85	-0.160000
31702	15.00	103.0	0.0	0	0.06	1.050000
113633	14.00	102.0	0.0	0	-1.91	-1.490000
...	...	...	...	...	...	...
27525	17.04	111.0	0.0	0	-0.04	0.060417
125185	13.07	100.0	14.0	0	-1.07	-0.450000
121521	14.05	102.0	0.0	0	-1.71	-1.620000
22242	17.03	107.0	0.0	0	-0.14	-0.130000
112022	15.06	102.0	0.0	0	-0.59	-0.690000

8059 rows × 6 columns

```
[ ] from sklearn.model_selection import cross_val_score, StratifiedKFold
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import confusion_matrix, accuracy_score, precision_score, recall_score, f1_score
import numpy as np

# Perform cross-validation on the Random Forest classifier with K=10
clf = RandomForestClassifier(max_depth=1, random_state=42)

# Kode kfold Cross Validation
cv = StratifiedKFold(n_splits=10, shuffle=True, random_state=42)

# Lists to store evaluation metrics for each fold
conf_matrices = []
accuracies = []

for train_index, test_index in cv.split(X_Anv, y_Anv):
    X_train, X_test = X_Anv.iloc[train_index], X_Anv.iloc[test_index]
    y_train, y_test = y_Anv.iloc[train_index], y_Anv.iloc[test_index]

    # Train the model
    clf.fit(X_train, y_train)

    # Predict on the test set
    y_pred = clf.predict(X_test)

    # Calculate evaluation metrics
    conf_matrices.append(confusion_matrix(y_test, y_pred))
    accuracies.append(accuracy_score(y_test, y_pred))

# Print the evaluation metrics for each fold
for i, (conf_matrix, accuracy) in enumerate(zip(conf_matrices, accuracies), 1):
    print(f"Fold-{i}: Accuracy={accuracy}")

# Print the average evaluation metrics
print('')
print('Hasil Rata Rata')
print(f"Average Accuracy: {np.mean(accuracies)}")

# Calculate and print the average confusion matrix
avg_conf_matrix = sum(conf_matrices)
print("Average Confusion Matrix:")
print(avg_conf_matrix)
```

```
Fold-1: Accuracy=0.9965846994535519
Fold-2: Accuracy=0.9979508196721312
Fold-3: Accuracy=0.9986338797814208
Fold-4: Accuracy=0.9993169398907104
Fold-5: Accuracy=0.9965823650034177
Fold-6: Accuracy=0.9979494190020506
Fold-7: Accuracy=0.9979494190020506
Fold-8: Accuracy=0.9952153110047847
Fold-9: Accuracy=0.9986329460013671
Fold-10: Accuracy=0.9986329460013671

Hasil Rata Rata
Average Accuracy: 0.9977448744812852
Average Confusion Matrix:
[[7317    0]
 [  33 7284]]
```

```
[ ] import seaborn as sns
    import matplotlib.pyplot as plt
    import numpy as np

    # Calculate the average confusion matrix
    avg_conf_matrix = sum(conf_matrices)

    # Create a heatmap
    plt.figure(figsize=(8, 6))
    sns.heatmap(avg_conf_matrix, annot=True, fmt=".0f", cmap="Blues")

    plt.xlabel("Predicted labels")
    plt.ylabel("True labels")
    plt.title("Average Confusion Matrix Random Forest")
    plt.show()
```