

TUGAS AKHIR - EF4801

PEMBUATAN DATA SINTETIK PADA DATA HISTORI PASIEN MENGGUNAKAN METODE BERBASIS GAN UNTUK DETEKSI RISIKO KOMPLIKASI TERAPI CAPD

MUHAMMAD NAUFAL BAIHAQI

NRP 5025211103

Dosen Pembimbing I

Dini Adni Navastara, S.Kom., M.Sc.

NIP 198510172015042001

Dosen Pembimbing II

Prof. Dr. Eng. Chistine Fatichah, S.Kom., M.Kom.

NIP 197512202001122002

Program Studi S1 Teknik Informatika

Departemen Teknik Informatika

Fakultas Teknologi Elektro dan Informatika Cerdas

Institut Teknologi Sepuluh Nopember

Surabaya

2025



TUGAS AKHIR - EF4801

PEMBUATAN DATA SINTETIK PADA DATA HISTORI PASIEN MENGGUNAKAN METODE BERBASIS GAN UNTUK DETEKSI RISIKO KOMPLIKASI TERAPI CAPD

MUHAMMAD NAUFAL BAIHAQI

NRP 5025211103

Dosen Pembimbing I

Dini Adni Navastara, S.Kom., M.Sc.

NIP 198510172015042001

Dosen Pembimbing II

Prof. Dr. Eng. Chastine Faticah, S.Kom., M.Kom.

NIP 197512202001122002

Program Studi S1 Teknik Informatika

Departemen Teknik Informatika

Fakultas Teknologi Elektro dan Informatika Cerdas

Institut Teknologi Sepuluh Nopember

Surabaya

2025



FINAL PROJECT - EF4801

**SYNTHETIC DATA GENERATION ON PATIENT HISTORY
DATA USING GAN-BASED METHOD FOR CAPD
THERAPY COMPLICATION RISK DETECTION**

MUHAMMAD NAUFAL BAIHAQI

NRP 5025211103

Advisor I

Dini Adni Navastara, S.Kom., M.Sc.

NIP 198510172015042001

Advisor II

Prof. Dr. Eng. Chastine Faticahah, S.Kom., M.Kom.

NIP 197512202001122002

Study Program Bachelor of Informatics

Department of Informatics

Faculty of Intelligent Electrical and Informatics Technology

Institut Teknologi Sepuluh Nopember

Surabaya

2025

LEMBAR PENGESAHAN

PEMBUATAN DATA SINTETIK PADA DATA HISTORI PASIEN MENGGUNAKAN
METODE BERBASIS GAN UNTUK DETEKSI RISIKO KOMPLIKASI TERAPI CAPD

TUGAS AKHIR

Diajukan untuk memenuhi salah satu syarat
memperoleh gelar Sarjana Komputer pada
Program Studi S-1 Teknik Informatika
Departemen Teknik Informatika
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember

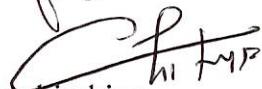
Oleh : MUHAMMAD NAUFAL BAIHAQI

NRP. 5025211103

Disetujui oleh Tim Penguji Penelitian :

1. Dini Adni Navastara, S.Kom., M.Sc.
2. Prof. Dr. Eng. Chastine Faticahah, S.Kom., M.Kom.
3. Prof. Dr. Eng. Nanik Suciati, S.Kom., M.Kom.
4. Dr.Eng. Radityo Anggoro, S.Kom., M.Sc.


Pembimbing


Ko-pembimbing


Penguj


Penguj

SURABAYA

Juli, 2025

(halaman ini sengaja dikosongkan)

APPROVAL SHEET

SYNTHETIC DATA GENERATION ON PATIENT HISTORY DATA USING GAN-BASED METHOD FOR CAPD THERAPY COMPLICATION RISK DETECTION

FINAL PROJECT

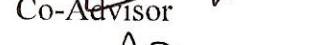
Submitted to fulfill one of the requirements
for obtaining a degree Bachelor of Computer Science at
Undergraduate Study Program of Infomatics
Department of Infomatics
Faculty of Intelligent Electrical and Informatics Technology
Institut Teknologi Sepuluh Nopember

By : MUHAMMAD NAUFAL BAIHAQI

NRP. 5025211103

Approved by Final Project Examiner Team:

1. Dini Adni Navastara, S.Kom., M.Sc.
2. Prof. Dr. Eng. Chastine Faticahah, S.Kom., M.Kom.
3. Prof. Dr. Eng. Nanik Suciati, S.Kom., M.Kom.
4. Dr. Eng. Radityo Anggoro, S.Kom., M.Sc.


Advisor

Co-Advisor

Examiner

Examiner

SURABAYA

July, 2025

(halaman ini sengaja dikosongkan)

PERNYATAAN ORISINALITAS

Yang bertanda tangan di bawah ini:

Nama mahasiswa / NRP : Muhammad Naufal Baihaqi / 5025211103
Program studi : Teknik Informatika
Pembimbing / NIP : Dini Adni Navastara, S.Kom., M.Sc. / 19851017201504
2001
Ko-Pembimbing / NIP : Prof. Dr. Eng. Chastine Fatichah, S.Kom., M.Kom.
/ 197512202001122002

dengan ini menyatakan bahwa Tugas Akhir dengan judul “PEMBUATAN DATA SINTETIK PADA DATA HISTORI PASIEN MENGGUNAKAN METODE BERBASIS GAN UNTUK DETEKSI RISIKO KOMPLIKASI TERAPI CAPD” adalah hasil karya sendiri, bersifat orisinal, dan ditulis dengan mengikuti kaidah penulisan ilmiah.

Bilamana di kemudian hari ditemukan ketidaksesuaian dengan pernyataan ini, maka saya bersedia menerima sanksi sesuai dengan ketentuan yang berlaku di Institut Teknologi Sepuluh Nopember.

Surabaya, 11 Juli 2025

Mengetahui
Dosen Pembimbing

Mahasiswa

Dini Adni Navastara, S.Kom., M.Sc.
NIP. 198510172015042001

Muhammad Naufal Baihaqi
NRP. 5025211103

Dosen Ko-pembimbing



Prof. Dr. Eng. Chastine Fatichah,
S.Kom., M.Kom.
NIP. 197512202001122002

(halaman ini sengaja dikosongkan)

STATEMENT OF ORIGINALITY

The undersigned below:

Name of student / NRP : Muhammad Naufal Baihaqi / 5025211103
Department : S-1 Teknik Informatika
Advisor / NIP : Dini Adni Navastara, S.Kom., M.Sc.
/ 198510172015042001
Co-Advisor / NIP : Prof. Dr. Eng. Chastine Faticahah, S.Kom., M.Kom.
/ 197512202001122002

Hereby declare that Final Project with the title of "SYNTHETIC DATA GENERATION ON PATIENT HISTORY DATA USING GAN-BASED METHOD FOR CAPD THERAPY COMPLICATION RISK DETECTION" is the result of my own work, is original, and is written by following the rules of scientific writing.

If in the future there is a discrepancy with this statement, then I am willing to accept sanctions in accordance with the provisions that apply at Institut Teknologi Sepuluh Nopember.

Surabaya, 11 Juli 2025

Acknowledge
Advisor

Student


Dini Adni Navastara, S.Kom., M.Sc.
NIP. 198510172015042001


Muhammad Naufal Baihaqi
NRP. 5025211103

Co-Advisor



Prof. Dr. Eng. Chastine Faticahah,
S.Kom., M.Kom.
NIP. 197512202001122002

(halaman ini sengaja dikosongkan)

ABSTRAK

PEMBUATAN DATA SINTETIK PADA DATA HISTORI PASIEN MENGGUNAKAN METODE BERBASIS GAN UNTUK DETEKSI RISIKO KOMPLIKASI TERAPI CAPD

Nama Mahasiswa / NRP : Muhammad Naufal Baihaqi / 5025211103
Departemen : Teknik Informatika FTEIC - ITS
Dosen Pembimbing : Dini Adni Navastara, S.Kom., M.Sc., Prof. Dr. Eng. Chastine Fatichah, S.Kom., M.Kom.

Abstrak

Continuous Ambulatory Peritoneal Dialysis (CAPD) adalah alternatif terapi pengganti ginjal yang fleksibel untuk pasien penyakit ginjal tahap akhir, namun memiliki risiko tinggi terhadap komplikasi infeksi. Penelitian ini mengembangkan model klasifikasi multimodal untuk deteksi risiko komplikasi pada pasien CAPD. Pendekatan ini mengatasi masalah ketidakseimbangan dataset dan keterbatasan data melalui synthetic data generation berbasis Generative Adversarial Network (GAN). Model multimodal ini menggabungkan data citra effluent dialysate yang diekstraksi menggunakan model pre-trained (Swin Transformer, CaiT, CoAtNet, YOLOv9) dengan data klinis pasien menggunakan metode early fusion. Untuk mengatasi ketidakseimbangan data, empat model synthetic data generation berbasis GAN, yaitu CTGAN, CTABGAN, CTABGAN+, dan CopulaGAN, diterapkan untuk menghasilkan data sintetik. Sembilan algoritma machine learning digunakan untuk proses klasifikasi, yaitu logistic regression, *k-nearest neighbors* (KNN), support vector machine (SVM), decision tree, random forest, AdaBoost, eXtreme Gradient Boosting (XGBoost), naïve Bayes (NB), dan Extra Trees. Hasil terbaik terdapat pada model machine learning Extra Trees. Model Extra Trees ini mencapai recall 0,9355, precision 1,0000, dan F1-score 0,9667 pada skenario penggunaan PCA 85% variance, CaiT sebagai feature extractor, standard scaler, dan CTABGAN sebagai model pembuat data sintetik. Pengembangan model deteksi ini berfokus pada meminimalkan false negative dan mengoptimalkan metrik F1-score serta precision, mengingat pentingnya deteksi kasus abnormal secara akurat dalam konteks medis. Hasil penelitian ini akan diintegrasikan ke dalam aplikasi SahabatCAPD sebagai sistem pemantauan mandiri pasien selama menjalani terapi CAPD untuk mendukung keputusan medis yang lebih cepat dan tepat dalam perawatan pasien CAPD.

Kata kunci: CAPD, Klasifikasi Multimodal, Data Sintetik, GAN, Effluent Dialysate

(halaman ini sengaja dikosongkan)

ABSTRACT

SYNTHETIC DATA GENERATION ON PATIENT HISTORY DATA USING GAN-BASED METHOD FOR CAPD THERAPY COMPLICATION RISK DETECTION

Student Name / NRP	: Muhammad Naufal Baihaqi / 5025211103
Department	: Teknik Informatika FTEIC - ITS
Advisor	: Dini Adni Navastara, S.Kom., M.Sc., Prof. Dr. Eng. Chastine Faticah, S.Kom., M.Kom.

Abstract

Continuous Ambulatory Peritoneal Dialysis (CAPD) is a flexible alternative renal replacement therapy for patients with end-stage renal disease, but has a high risk of infectious complications. This study develops a multimodal classification model for complication risk detection in CAPD patients. This approach overcomes the problem of dataset imbalance and data limitation through synthetic data generation based on Generative Adversarial Network (GAN). The multimodal model combines effluent dialysate image data extracted using pre-trained models (Swin Transformer, CaiT, CoAtNet, YOLOv9) with patient clinical data using early fusion method. To overcome data imbalance, four GAN-based synthetic data generation models, namely CTGAN, CTABGAN, CTABGAN+, and CopulaGAN, were applied to generate synthetic data. Nine machine learning algorithms were used for the classification process, namely logistic regression, k-nearest neighbors (KNN), support vector machine (SVM), decision tree, random forest, AdaBoost, eXtreme Gradient Boosting (XGBoost), naïve Bayes (NB), and Extra Trees. The best results are found in the Extra Trees machine learning model. This Extra Trees model achieved recall 0.9355, precision 1.0000, and F1-score 0.9667 in the scenario of using PCA 85% variance, CaiT as feature extractor, standard scaler, and CTABGAN as synthetic data generation model. The development of this detection model focuses on minimizing false negatives and optimizing F1-score and precision metrics, given the importance of accurate detection of abnormal cases in a medical context. The results of this study will be integrated into the SahabatCAPD application as a patient self-monitoring system during CAPD therapy to support faster and more informed medical decisions in CAPD patient care.

Keywords: CAPD, Multimodal Classification, Synthetic Data, GAN, Effluent Dialysate

(halaman ini sengaja dikosongkan)

KATA PENGANTAR

Alhamdulillah, puji syukur penulis panjatkan kepada Allah Swt. atas segala rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan tugas akhir yang berjudul “Pembuatan Data Sintetik pada Data Histori Pasien Menggunakan Metode Berbasis GAN untuk Deteksi Risiko Komplikasi Terapi CAPD” dengan lancar. Terselesaiannya Tugas Akhir ini tidak lepas dari dukungan, bantuan, dan doa dari berbagai pihak, baik secara langsung maupun tidak langsung. Oleh karena itu, penulis menyampaikan terima kasih dan penghargaan yang sebesar-besarnya kepada:

1. Orang tua penulis, Bapak Alm. Bakori dan Ibu Meri Mariam, serta kedua kakak penulis, Luthfiana Azmi dan Ayu Faridah, atas segala doa, didikan, semangat, dan dukungan yang tak pernah henti.
2. Ibu Dini Adni Navastara, S.Kom., M.Sc. dan Ibu Prof. Dr. Eng. Chistine Faticah, S.Kom., M.Kom., selaku dosen pembimbing, atas arahan, masukan, dan bimbingan yang sangat berarti selama proses penyusunan Tugas Akhir ini.
3. Bapak/Ibu Pengudi Sidang yang telah memberikan masukan, kritik, dan saran konstruktif demi perbaikan karya ini.
4. Seluruh dosen dan staf Departemen Informatika ITS yang telah membagikan ilmu dan membantu kelancaran studi selama ini.
5. Teman-teman seperjuangan di Informatika ITS, terkhusus remtil (Adam, Akmal, Alfa, Alfan, Andika, Aryan, Cakno, Dafian, Daud, Daris, Dewangga, Dimas, Ihsan, Key, Rifqi, Syukra, Vino, Yoel, dan Zeon) atas dukungan, semangat, dan diskusi-diskusi yang membangun selama masa perkuliahan.
6. Nadya Saraswati Putri dan Wilman Nugraha, atas dukungan dan bantuan dari berbagai aspek, serta ketersediaannya mendengarkan cerita dan keluh kesah penulis selama proses penggerjaan Tugas Akhir ini.
7. Segenap keluarga besar Flexoo Software House, terkhusus Rule Lulu Damara, Agym Kamil Ramadhan, dan Abimanyu Danendra, yang telah memberikan salah satu pengalaman paling mengesankan bagi penulis selama menjalani perkuliahan.
8. Seluruh pihak yang telah membantu penulis dalam menyelesaikan Tugas Akhir ini yang tidak dapat disebutkan penulis satu persatu.

Penulis menyadari bahwa Tugas Akhir ini masih memiliki kekurangan, baik dari sisi penulisan maupun substansi, karena keterbatasan pengetahuan dan pengalaman. Semoga kekurangan tersebut menjadi titik awal untuk pembelajaran dan pengembangan keilmuan lebih lanjut. Akhir kata, semoga hasil penelitian ini dapat bermanfaat bagi perkembangan ilmu pengetahuan dan teknologi di masa yang akan datang.

Hormat saya,
Muhammad Naufal Baihaqi

(halaman ini sengaja dikosongkan)

DAFTAR ISI

LEMBAR PENGESAHAN	i
APPROVAL SHEET	iii
PERNYATAAN ORISINALITAS	v
STATEMENT OF ORIGINALITY	vii
ABSTRAK	ix
ABSTRACT	xi
KATA PENGANTAR	xiii
DAFTAR ISI	xv
DAFTAR GAMBAR	xix
DAFTAR TABEL	xxi
DAFTAR KODE SEMU	xxiii
DAFTAR PERSAMAAN	xxv
BAB 1 PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	2
1.3 Batasan Masalah	2
1.4 Tujuan	3
1.5 Manfaat	3
BAB 2 TINJAUAN PUSTAKA	5
2.1 Hasil Penelitian Terdahulu	5
2.2 Dasar Teori	6
2.2.1 <i>Continuous Ambulatory Peritoneal Dialysis (CAPD)</i>	6
2.2.2 <i>Effluent Dialysate</i>	6
2.2.3 <i>Pre-trained Model</i>	8
2.2.4 <i>Generative Adversarial Network</i>	11
2.2.5 <i>Multimodal Fusion</i>	14
2.2.6 <i>Ensemble Learning</i>	14
2.2.7 <i>Evaluasi Performa</i>	14
BAB 3 METODOLOGI	17
3.1 Metode yang digunakan	17
3.2 <i>Dataset</i> yang Digunakan	17
3.3 Perancangan Sistem	20

3.3.1	<i>Data Preprocessing</i> pada Data Klinis	20
3.3.2	<i>Image Preprocessing</i> pada Cairan <i>Effluent Dialysate</i>	21
3.3.3	<i>Feature Extraction</i>	21
3.3.4	<i>Early Fusion</i>	22
3.3.5	<i>Data Splitting</i>	22
3.3.6	<i>Dimensionality Reduction</i>	22
3.3.7	<i>Synthetic Data Generation</i>	23
3.3.8	<i>Normalisasi</i>	23
3.3.9	<i>Model Building</i>	23
3.3.10	<i>Model Evaluation</i>	24
3.4	Implementasi	24
3.4.1	Implementasi <i>Exploratory Data Analysis</i>	24
3.4.2	Implementasi <i>Data Preprocessing</i>	25
3.4.3	Implementasi <i>Image Preprocessing</i>	26
3.4.4	Implementasi <i>Feature Extraction</i>	27
3.4.5	Implementasi <i>Early Fusion</i>	28
3.4.6	Implementasi <i>Data Splitting</i>	28
3.4.7	Implementasi <i>Dimensionality Reduction</i>	29
3.4.8	Implementasi <i>Synthetic Data Generation</i>	30
3.4.9	Implementasi <i>Normalization</i>	31
3.4.10	Implementasi <i>Model Building</i>	32
3.4.11	Implementasi <i>Model Evaluation</i>	35
BAB 4	Hasil dan Pembahasan.....	37
4.1	Skenario Uji Coba	37
4.1.1	Uji Coba <i>Feature Extractor</i>	38
4.1.2	Uji Coba Normalisasi	47
4.1.3	Uji Coba Model Sintetik Data	53
4.1.4	Uji Coba <i>Classifiers</i>	63
4.2	Pembahasan	67
4.2.1	Pembahasan Uji Coba <i>Feature Extractor</i>	68
4.2.2	Pembahasan Uji Coba Normalisasi	69
4.2.3	Pembahasan Uji Coba Model Sintetik Data	70
4.2.4	Pembahasan Uji Coba <i>Classifier</i>	71
BAB 5	Kesimpulan dan Saran.....	75

5.1	Kesimpulan.....	75
5.2	Saran	75
	DAFTAR PUSTAKA	77
	LAMPIRAN.....	83
	BIODATA PENULIS	89

(halaman ini sengaja dikosongkan)

DAFTAR GAMBAR

Gambar 2.1 Alur Proses CAPD.....	6
Gambar 2.2 Arsitektur PGI pada YOLOv9 (Yaseen, 2024)	8
Gambar 2.3 Arsitektur GELAN pada YOLOv9 (Yaseen, 2024)	9
Gambar 2.4 Overview Arsitektur CaiT (Wang et al., 2023)	9
Gambar 2.5 Overview Arsitektur Swin Transformer (Liu et al., 2021).....	10
Gambar 2.6 Overview Arsitektur CoAtNet (Dai et al., 2021)	10
Gambar 2.7 Arsitektur GAN (Baowaly, Liu, et al., 2019)	11
Gambar 2.8 Arsitektur CTGAN (Skoularidou & Cuesta-infante, 2019)	12
Gambar 2.9 Arsitektur CTAB-GAN (Zhao et al., 2017).....	12
Gambar 2.10 Arsitektur CTABGAN+ (Zhao, Kunar, et al., 2023).....	13
Gambar 2.11 Arsitektur CopulaGAN.....	13
Gambar 3.1 Sampel Citra Effluent Dialysate.....	18
Gambar 3.2 Buku Catatan Monitoring Penggantian CAPD.....	18
Gambar 3.3 Diagram Alir Proses Penelitian	20
Gambar 3. 4 Alur Data Preprocessing pada Data Klinis .. Error! Bookmark not defined.	
Gambar 4.1 Evaluasi PCA.....	37
Gambar 4. 2 Grafik Nilai Evaluasi Uji Coba Feature Extractor	68
Gambar 4. 3 Grafik Nilai Evaluasi Uji Coba Normalisasi	69
Gambar 4. 4 Grafik Nilai Evaluasi Uji Coba Model Sintetik Data	70
Gambar 4. 5 Grafik Nilai Evaluasi <i>Precision Classifiers</i>	71
Gambar 4. 6 Grafik Nilai Evaluasi <i>Recall Classifiers</i>	72
Gambar 4. 7 Grafik Nilai Evaluasi <i>F1-score Classifiers</i>	72
Gambar 4. 8 Confusion Metrics	72
Gambar 4. 9 Data Uji False Negative.....	73

(halaman ini sengaja dikosongkan)

DAFTAR TABEL

Tabel 2.1 Jenis-Jenis Effluent Dialysate Beserta Pertimbangan Diagnostik	7
Tabel 2.2 Confusion Matrix	15
Tabel 3.1 Fitur-Fitur pada Data Klinis Pasien CAPD	19
Tabel 4.1 Skenario Uji Coba	37
Tabel 4.2 Proporsi <i>Dataset Uji Coba Feature Extractor</i>	38
Tabel 4.3 Evaluasi <i>Cross validation</i> CaiT	38
Tabel 4.4 Evaluasi <i>Modelling</i> CaiT.....	39
Tabel 4.5 Evaluasi <i>Hyperparameter tuning</i> CaiT	40
Tabel 4.6 Evaluasi <i>Cross validation</i> CoAtNet.....	41
Tabel 4.7 Evaluasi <i>Modelling</i> CoAtNet	41
Tabel 4.8 Evaluasi <i>Hyperparameter tuning</i> CoAtNet.....	42
Tabel 4.9 Evaluasi <i>Cross validation</i> Swin Transformer	43
Tabel 4.10 Evaluasi <i>Modelling</i> Swin Transformer	43
Tabel 4.11 Evaluasi <i>Hyperparameter tuning</i> Swin Transformer.....	44
Tabel 4.12 Evaluasi <i>Cross validation</i> YOLOv9	45
Tabel 4.13 Hasil Evaluasi <i>Modelling</i> YOLOv9	45
Tabel 4.14 Hasil Evaluasi <i>Hyperparameter tuning</i> YOLOv9.....	46
Tabel 4.15 Proporsi <i>Dataset Uji Coba Normalisasi</i>	47
Tabel 4.16 Hasil Evaluasi <i>Cross validation</i> Tanpa Normalisasi.....	47
Tabel 4.17 Hasil Evaluasi <i>Modelling</i> Tanpa Normalisasi.....	48
Tabel 4.18 Hasil Evaluasi <i>Hyperparameter tuning</i> Tanpa Normalisasi	49
Tabel 4.19 Hasil Evaluasi <i>Cross validation</i> Robust Scaler.....	49
Tabel 4.20 Hasil Evaluasi <i>Modelling</i> Robust Scaler.....	50
Tabel 4.21 Hasil Evaluasi <i>Hyperparameter tuning</i> Robust Scaler	50
Tabel 4.22 Hasil Evaluasi <i>Cross validation</i> Standard Scaler.....	51
Tabel 4.23 Hasil Evaluasi <i>Modelling</i> Standard Scaler	52
Tabel 4.24 Hasil Evaluasi <i>Hyperparameter tuning</i> Standard Scaler	52
Tabel 4.25 Proporsi <i>Dataset Uji Coba Model Sintetik Data</i>	54
Tabel 4.26 Hasil Evaluasi <i>Cross validation</i> CTGAN	54
Tabel 4.27 Hasil Evaluasi <i>Modelling</i> CTGAN	55
Tabel 4.28 Hasil Evaluasi <i>Hyperparameter tuning</i> CTGAN	55
Tabel 4.29 Hasil Evaluasi <i>Cross validation</i> CTABGAN.....	56
Tabel 4.30 Hasil Evaluasi <i>Modelling</i> CTABGAN.....	57
Tabel 4.31 Hasil Evaluasi <i>Hyperparameter tuning</i> CTABGAN	57
Tabel 4.32 Hasil Evaluasi <i>Cross validation</i> CTABGAN+	59
Tabel 4.33 Hasil Evaluasi <i>Modelling</i> CTABGAN+.....	59
Tabel 4.34 Hasil Evaluasi <i>Hyperparameter tuning</i> CTABGAN+.....	60
Tabel 4.35 Hasil Evaluasi <i>Cross validation</i> CopulaGAN	61
Tabel 4.36 Hasil Evaluasi <i>Modelling</i> CopulaGAN	61
Tabel 4.37 Hasil Evaluasi <i>Hyperparameter tuning</i> CopulaGAN	62
Tabel 4.38 Proporsi <i>Dataset Uji Coba Classifiers</i>	63
Tabel 4.39 Hasil Evaluasi <i>Cross validation</i> Single Learning	64
Tabel 4.40 Hasil Evaluasi <i>Modelling</i> Single Learning	64

Tabel 4.41 Hasil Evaluasi <i>Hyperparameter tuning</i> Single Learning	65
Tabel 4.42 Hasil Evaluasi <i>Cross validation</i> Ensemble Learning	66
Tabel 4.43 Hasil Evaluasi <i>Modelling</i> Ensemble Learning	66
Tabel 4.44 Hasil Evaluasi <i>Hyperparameter tuning</i> Ensemble Learning	67
Tabel 4.45 Ringkasan Nilai Evaluasi Uji Coba Feature Extractor	68
Tabel 4.46 Ringkasan Nilai Evaluasi Uji Coba Normalisasi	69
Tabel 4.47 Ringkasan Nilai Evaluasi Uji Coba Model Sintetik Data	70

DAFTAR KODE SEMU

Kode Semu 3.1 Pseudocode Exploratory Data Analysis	25
Kode Semu 3.2 Pseudocode Data Preprocessing	26
Kode Semu 3.3 Pseudocode Image Preprocessing	27
Kode Semu 3.4 Pseudocode Feature Extraction	27
Kode Semu 3.5 Pseudocode Early Fusion	28
Kode Semu 3.6 Pseudocode Data Splitting	29
Kode Semu 3.7 Pseudocode Dimensionality Reduction.....	29
Kode Semu 3.8 Pseudocode Synthetic Data Generation	30
Kode Semu 3.9 Pseudocode Normalization.....	31
Kode Semu 3.10 Pseudocode Early Modelling	33
Kode Semu 3.11 Pseudocode Cross validation.....	34
Kode Semu 3.12 Pseudocode Hyperparameter tuning.....	35

(halaman ini sengaja dikosongkan)

DAFTAR PERSAMAAN

Persamaan 2.1	13
Persamaan 2.2	15
Persamaan 2.3	15
Persamaan 2.4	15
Persamaan 2.5	16

(halaman ini sengaja dikosongkan)

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Penyakit ginjal tahap akhir (PGTA) telah menjadi salah satu masalah kesehatan utama di seluruh dunia (Zhao, Yan, et al., 2023). Dalam kasus global dari tahun 1990 hingga 2016, penderita gagal ginjal kronis (GGK) meningkat sebesar 89% dan angka kematian akibat GGK meningkat sampai 98% (Supriyadi et al., 2020). Dalam kondisi seperti ini, terapi pengganti ginjal (TPG) sangat dibutuhkan untuk membantu fungsi ginjal pasien PGTA. Ada tiga jenis TPG yang umumnya digunakan, yaitu hemodialisis, dialisis peritoneal (PD), dan transplantasi ginjal (Anggraini & Fadila, 2022). Hemodialisis adalah prosedur yang menggunakan mesin untuk membersihkan darah dari limbah saat ginjal tidak berfungsi. Dialisis peritoneal adalah prosedur yang menggunakan rongga perut untuk menyaring darah. Lalu, transplantasi ginjal adalah penempatan ginjal sehat dari donor untuk menggantikan ginjal yang rusak. Dari ketiga prosedur tersebut, hemodialisis memiliki efektivitas yang tinggi dengan biaya di lebih rendah daripada transplantasi ginjal. Akan tetapi, tidak semua pasien PGTA di Indonesia bisa mendapatkan layanan hemodialisis. Hal tersebut disebabkan oleh minimnya fasilitas di rumah sakit dan kurangnya tenaga medis. Oleh karena itu, digunakan terapi lain yang dapat dijangkau oleh lebih banyak pasien PGTA.

Continuous Ambulatory Peritoneal Dialysis (CAPD) adalah salah satu jenis terapi PD yang menjadi alternatif perawatan yang lebih fleksibel karena dapat dilakukan oleh pasien sendiri di rumah tanpa memerlukan peralatan kompleks atau perjalanan ke pusat layanan kesehatan (Dewi & Kandarini, 2020). Meskipun demikian, CAPD memiliki risiko tinggi terhadap infeksi, terutama jika prosedur penggantian cairan tidak dilakukan dengan benar (Mihalache et al., 2018). Penyebabnya adalah kelalaian pasien dan pengoperasian yang tidak sesuai standar yang menyebabkan masuknya bakteri melalui udara dari antarmuka tabung dialisis peritoneal (Zhao, Yan, et al., 2023). Dengan kondisi tersebut, perlu dibuat sebuah teknologi deteksi dini abnormalitas pasien PGTA selama menjalani terapi CAPD.

Dalam upaya mengembangkan teknologi deteksi risiko komplikasi pada CAPD, diperlukan model prediktif yang dapat memanfaatkan berbagai sumber data, seperti citra *effluent dialysate* dan hasil pemantauan klinis. Konsep penggabungan data dari berbagai modalitas ini dikenal sebagai multimodal. Beberapa penelitian sebelumnya telah menunjukkan efektivitas metode ini. Sebagai contoh, penelitian yang menggabungkan data piksel volumetrik dan data klinis untuk mendeteksi risiko pulmonary embolism menghasilkan nilai AUC sebesar 0,96 dan *recall* 94% (Cahan et al., 2023). Pada penelitian lain dimanfaatkan data multimodal seperti ekspresi gen dan data klinis untuk memprediksi prognosis kanker payudara, dengan AUC 0,93 dan *recall* 74,7% (Arya & Saha, 2022). Penelitian lainnya juga berhasil mengembangkan sistem klasifikasi kanker paru-paru dengan menggabungkan data CT scan dan data klinis, mencapai akurasi 93,6%, sensitivitas 91,9%, spesifitas 95,6%, dan AUC 97,1% (Bai & Tang, 2022).

Dalam konteks CAPD, telah dilaksanakan berbagai penelitian. penelitian oleh Navastara berhasil mendeteksi risiko komplikasi menggunakan citra *effluent dialysate* dan menghasilkan *recall* 80% dengan arsitektur InceptionResNetV2 (Navastara et al., 2023). Penelitian lainnya oleh Jalil (2023) mengembangkan sistem pemantauan bernama SahabatCAPD yang mencatat volume dan waktu penggantian cairan. Penelitian sebelumnya juga mengidentifikasi keterbatasan karena pencatatan kondisi pasien masih dilakukan secara manual sehingga data

yang tersedia belum lengkap (Navastara et al., 2023). Penelitian terbaru oleh Jayanti (2024) telah mencoba menggabungkan beberapa *dataset*, seperti citra *effluent dialysate* dari penelitian Sari (2023), data pemantauan klinis dari Jalil (2023), dan data kondisi fisik seperti tekanan darah dan berat badan. Meskipun integrasi beberapa sumber data telah dilakukan, data yang dipakai dalam proses penelitian sebelumnya mengindikasikan ketidakseimbangan karena data abnormal hanya 10% dari keseluruhan *dataset*. *Imbalanced datasets* dalam bidang medis, seperti pada pengklasifikasian kanker kulit, dapat menyebabkan model memberikan prediksi yang bias terhadap kelas mayoritas sehingga menurunkan akurasi deteksi kelas minoritas (Alam et al., 2022). Ketidakseimbangan ini juga menghambat kinerja model dalam aplikasi medis yang memerlukan deteksi kasus langka atau abnormal (Jiang et al., 2021).

Dengan kondisi tersebut, penelitian ini akan menjadi bentuk pengembangan dari penelitian-penelitian yang sudah dilakukan sebelumnya terkait dengan deteksi risiko komplikasi CAPD. Pada penelitian ini dilakukan penambahan data sintetis dari data primer yang tersedia menggunakan beberapa model generatif berbasis *Generative Adversarial Network* (GAN) untuk mengatasi permasalahan jumlah data yang masih terbatas dan ketidakseimbangan *dataset*. Penelitian ini juga akan menggunakan model deteksi risiko komplikasi CAPD berbasis klasifikasi multimodal. Kelas target akan dibagi menjadi normal dan abnormal. Kemudian, akan diterapkan pula beberapa skenario. Evaluasi akan dilakukan dengan mengoptimalkan *recall*, *precision* dan *F1-score*. Akan tetapi, fokus utamanya adalah meminimalkan *false negative* untuk meminimalisasi suatu komplikasi yang terlewat. Model terbaik nantinya akan diintegrasikan ke dalam sistem SahabatCAPD sebagai aplikasi *monitoring* pasien PGTA saat menjalani terapi CAPD. Hal tersebut bertujuan untuk mendekripsi risiko abnormalitas terapi CAPD secara otomatis dengan memanfaatkan data laporan pasien.

1.2 Rumusan Masalah

Berdasarkan latar belakang di atas, fokus permasalahan dalam penelitian ini adalah

1. Bagaimana membangun model dengan pendekatan multimodal menggunakan *dataset* citra *effluent dialysate* dan tabel pemantauan pasien untuk mendekripsi abnormalitas komplikasi pasien saat terapi CAPD?
2. Bagaimana cara mengatasi keterbatasan jumlah data pada penelitian sebelumnya dalam pengembangan model deteksi abnormalitas komplikasi CAPD menggunakan metode berbasis GAN?
3. Bagaimana mengevaluasi performa model deteksi abnormalitas komplikasi CAPD yang menggunakan pendekatan multimodal dengan penambahan data sintetis?

1.3 Batasan Masalah

Adapun batasan dalam pelaksanaan Penelitian ini adalah

1. *Dataset* yang digunakan berupa citra *effluent dialysate* dan data pemantauan pasien dengan penyakit ginjal tahap akhir (PGTA) yang melakukan terapi CAPD. dikumpulkan secara langsung oleh penulis dari pasien CAPD lewat aplikasi SahabatCAPD dengan perizinan oleh KEPK Rumah Sakit Universitas Airlangga dan Rumah Sakit Umum Dr. Sardjito.

2. *Tools* yang digunakan untuk membangun model dengan pendekatan multimodal sebagai deteksi risiko komplikasi CAPD adalah Jupyter Notebook menggunakan platform Google Colab, PC Teknik Informatika, dan bahasa pemrograman Python.
3. Penilaian performa metode klasifikasi yang digunakan adalah menggunakan nilai *recall*, *precision*, dan *F1-score*.

1.4 Tujuan

Tujuan yang ingin dicapai dari penggerjaan Penelitian ini adalah

1. Membangun model deteksi abnormalitas komplikasi pasien saat menjalani terapi CAPD dengan pendekatan multimodal yang menggabungkan *dataset* citra *effluent dialysate* dan tabel pemantauan pasien yang sedang menjalani terapi CAPD.
2. Mengatasi keterbatasan jumlah data pada penelitian sebelumnya dengan menggunakan metode *synthetic data generation* berbasis GAN untuk pengembangan model deteksi abnormalitas komplikasi CAPD.
3. Mengevaluasi performa model deteksi abnormalitas komplikasi CAPD yang menggunakan pendekatan multimodal dengan penambahan data sintetis sehingga mendapatkan model dengan performa terbaik, dapat meminimalisasi *false negative*.

1.5 Manfaat

Pengerjaan Penelitian ini menghasilkan model yang optimal dengan metode multimodal. Model ini diharapkan dapat mendukung sistem deteksi dini risiko komplikasi pada pasien CAPD serta membantu pemantauan kondisi pasien secara lebih efektif pada sistem SahabatCAPD. Dengan adanya sistem ini, keterlambatan dalam penanganan komplikasi CAPD dapat diminimalisasi, serta dapat memperluas akses layanan kesehatan yang berkualitas bagi pasien dengan penyakit ginjal tahap akhir (PGTA) di Indonesia.

(halaman ini sengaja dikosongkan)

BAB 2 TINJAUAN PUSTAKA

2.1 Hasil Penelitian Terdahulu

Aplikasi teknologi *machine learning* (ML) di dunia medis khususnya pada sistem deteksi dini suatu penyakit atau komplikasi sedang marak digunakan. Teknologi ML sudah digunakan untuk memprediksi berbagai jenis penyakit, seperti kanker, diabetes, dan kondisi kardiovaskular melalui diagnosis dini (Siddiq, 2022). Aplikasi lain dari ML yang paling menarik adalah dalam mendeteksi penyakit arteri koroner (CAD) menggunakan pencitraan angiografi. Sebuah studi oleh (Information & Sciences, 2023) menunjukkan bahwa teknik *deep learning* dapat mencapai akurasi 90,6% dalam mengenali CAD dari gambar angiografi, melampaui algoritme ML tradisional dan kemampuan diagnostik manusia. Selain itu, integrasi ML melalui model *Modified Early Warning Score* (MEWS) telah berhasil memprediksi penurunan kondisi pasien yang berisiko mengalami eskalasi klinis enam jam sebelum peristiwa tersebut terjadi, sehingga memfasilitasi respons klinis yang cepat (Kia et al., 2020).

Pada penelitian lain yang berhubungan dengan penyakit langka, Uveitis, dilakukan *synthetic data generation* menggunakan metode medWGAN. Lebih dari 55% data sintetik yang dihasilkan dinilai baik atau sangat baik oleh ahli medis. Selain itu, model ML yang menggunakan data sintetik tersebut berhasil mencapai akurasi lebih dari 80% dalam memprediksi etiologi Uveitis (Sliman et al., 2023). Pada penelitian terkait pembuatan data sintetik dari *electronic health records* (EHR) yang ada, digunakan model medGAN dan dua variasinya, yaitu medWGAN dan medBGAN. Dari ketiga model tersebut, medBGAN memiliki kinerja terbaik (Baowaly, Lin, et al., 2019). Untuk tugas pemodelan prediktif, didapatkan bahwa medBGAN menunjukkan hasil yang sangat baik dan mengungguli medGAN dalam semua uji yang dipakai (Baowaly, Liu, et al., 2019).

Pada terapi *peritoneal dialysis* (PD), penggunaan *machine learning* dan *artificial intelligence* (AI) telah menunjukkan manfaat dalam meningkatkan prediksi komplikasi dan infeksi. Penelitian yang dilakukan oleh Bai dan Tang memberikan gambaran umum tentang penerapan AI/ML dalam PD, dengan studi yang mengembangkan alat prediksi *prolonged length of stay* (pLOS) pada pasien PD menggunakan kombinasi ML dan regresi logistik (Bai & Tang, 2022). Pada penelitian lain yang berasal dari Korea Selatan ditemukan bahwa *deep neural network* (DNN) lebih baik dalam memprediksi risiko kematian pasien PD dibandingkan regresi logistik (Noh et al., 2020). Selain itu, penelitian (J. Wu et al., 2020) juga menunjukkan keefektifan ML dalam memprediksi lama rawat inap yang berkepanjangan pada pasien PD dengan menggunakan algoritma seperti *classification and regression tree* (CART), RF, dan *Gradient Boosting Decision Trees* (GBDT).

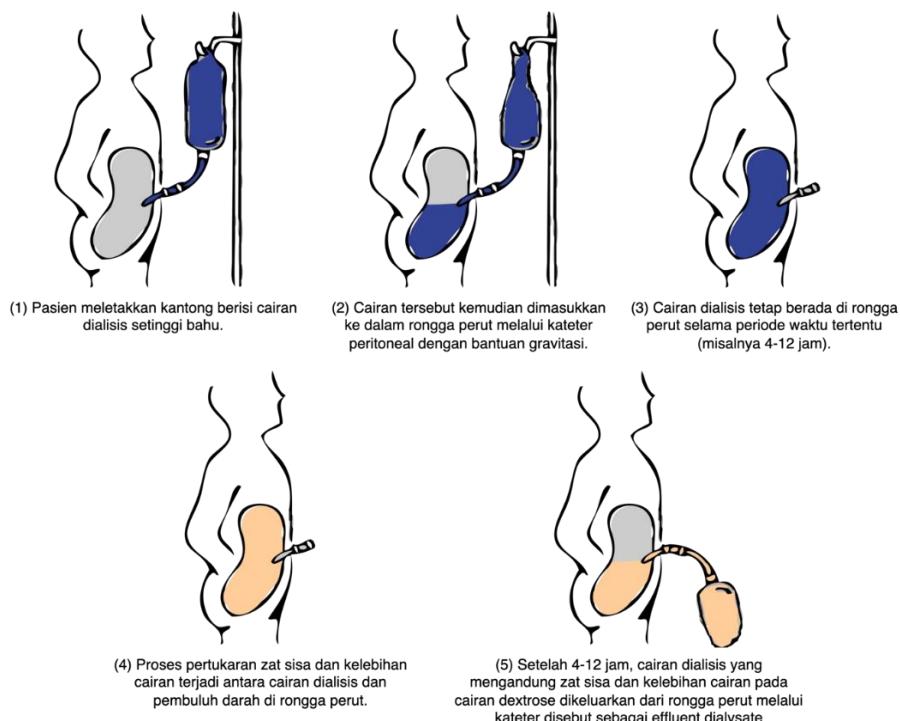
Studi terkait deteksi dini risiko komplikasi pada terapi CAPD telah dilakukan oleh Sari (2023), yang menggunakan dataset citra *effluent dialysate* dengan metode *deep learning* dan mencapai nilai evaluasi *recall* sebesar 80% (Navastara et al., 2023). Namun, penelitian ini menghadapi masalah sumber data tunggal. Kekurangan tersebut kemudian diatasi oleh Jayanti (2024) dengan menambahkan data klinis pasien sebagai indikator awal adanya risiko komplikasi pada terapi CAPD. Dengan pendekatan ini, model yang dikembangkan mampud mengurangi masalah ketidakseimbangan data kelas. Meskipun demikian, penelitian Jayanti masih memiliki keterbatasan pada perbandingan kelas labelnya yang tidak seimbang dengan kelas abnormal hanya 10% dari keseluruhan dataset. Keterbatasan ini berdampak pada

performa model dan kemampuannya dalam melakukan generalisasi. Oleh karena itu, penelitian ini berupaya mengatasi keterbatasan tersebut dengan melakukan *synthetic data generation* berdasarkan gabungan citra *effluent dialysate* dengan data klinis yang telah ditentukan. Dengan strategi ini, diharapkan dapat memperoleh *dataset* yang seimbang antar kelasnya sehingga performa model dapat meningkat dan lebih *robust*.

2.2 Dasar Teori

2.2.1 Continuous Ambulatory Peritoneal Dialysis (CAPD)

CAPD merupakan salah satu metode terapi pengganti ginjal berupa dialisis mandiri yang memanfaatkan membran rongga perut (*peritoneum*) sebagai penyaring dan cairan dialisat (*dextrose*) steril sebagai pengganti fungsi ginjal. Alur proses dialisis CAPD ditunjukkan pada **Gambar 2.1**.



Gambar 2.1 Alur Proses CAPD

Proses di atas dilakukan 3-5 kali setiap hari dengan waktu penggantian (*dwell time*) 4-12 jam sesuai rekomendasi dari dokter (Goldstein et al., 2013). Kemudian cairan *dextrose* dikeluarkan dari tubuh dan disebut sebagai *effluent dialysate* (cairan buangan). Cairan *Effluent dialysate* pasien CAPD dapat digunakan sebagai indikator awal terjadinya abnormalitas intra-peritoneal maupun inter-peritoneal (Dossin & Goffin, 2019).

2.2.2 Effluent Dialysate

Effluent dialysate merupakan cairan yang dihasilkan atau dikeluarkan dari proses terapi *peritoneal dialysis*. *Effluent dialysate*, atau disebut juga dengan cairan buangan, mengandung zat-zat limbah dan kelebihan cairan yang dikeluarkan dari tubuh pasien melalui membran *peritoneum* (Navastara et al., 2023). Dalam keadaan normal, *effluent dialysate* berwarna agak kuning dan transparan atau jernih. Kasus lain yang terjadi adalah ketika *effluent dialysate* berwarna keruh. Kasus tersebut mengindikasikan adanya infeksi bakteri *Corynebacterium amycolatum* pada membran *peritoneum*, atau disebut dengan *peritonitis*, sehingga menyebabkan jumlah leukosit pada sel *polymorphonuclear* (PMN) pada cairan *peritoneum*

sangat tinggi (Habeeb et al., 2023). Kasus lain yang sejenis juga disampaikan oleh Xiao tahun 2023, dimana *peritonitis* didiagnosis apabila cairan *effluent dialysate* berwarna keruh dan jumlah total sel darah putih pada cairan *peritoneum* lebih dari 100 sel/mm³ dengan PMN lebih dari 50% (Xiao et al., 2023). Dari kasus yang telah disebutkan, semakin mendukung Dossin dan Goffin bahwa kondisi warna pada *effluent dialysate* dapat menjadi indikator awal diagnosis komplikasi yang dapat menjadi input *deep learning*.

Pada penelitian ini, citra *effluent dialysate* hanya dapat diperoleh dengan melakukan pengumpulan primer karena pengumpulan sekunder hanya sedikit yang tersedia. Data primer citra *effluent dialysate* didapatkan dari aplikasi SahabatCAPD yang sudah bekerjasama dengan dua rumah sakit dan telah mendapatkan izin pemakaian data untuk penelitian. Pada **Tabel 2.1** dapat terlihat contoh jenis-jenis *effluent dialysate* berdasarkan sumber penelitian terkait.

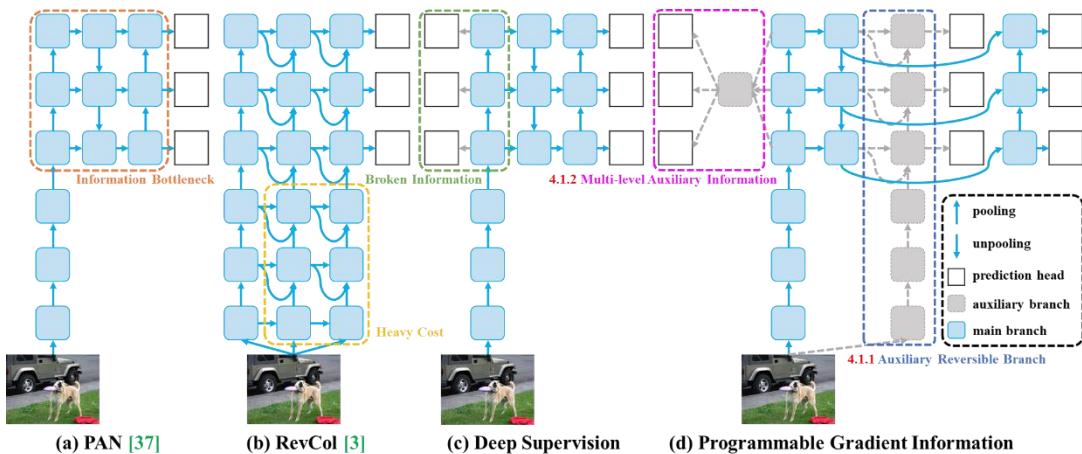
Tabel 2.1 Jenis-Jenis *Effluent Dialysate* Beserta Pertimbangan Diagnostik

Diagnosis	Citra <i>Effluent Dialysate</i>	
Normal		
Normal (PMN < 50%)		(Dewi & Kandarini, 2020)
Abnormal		
Infeksi bakteri		(Dewi & Kandarini, 2020; Dossin & Goffin, 2019)
Kolesistitis (adanya empedu dalam kantong cairan)		(Dossin & Goffin, 2019)
Retinopati diabetik		(Dossin & Goffin, 2019)
Kolesistitis (perforasi kantung empedu)		(Y. L. Wu et al., 2018)
Akumulasi trigliserida di rongga peritoneum		(Joubran et al., 2020)
Dialisat merah (darah)		(Dossin & Goffin, 2019)

2.2.3 Pre-trained Model

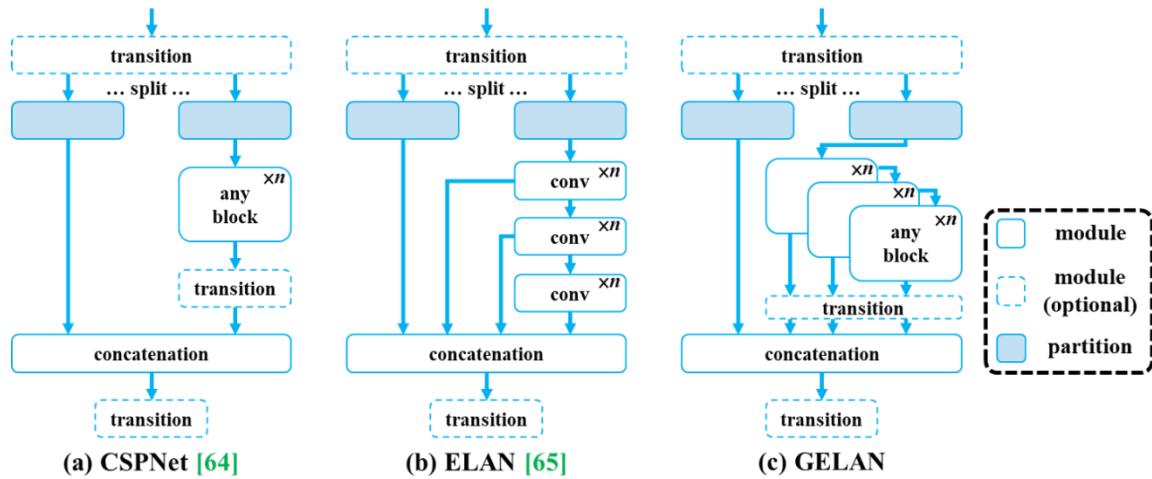
Pre-trained Model merupakan model-model *machine learning* yang telah dilakukan *training* sebelumnya. Model-model ini menjadi landasan *machine learning* modern dan *computer vision*, terutama dalam tugas-tugas seperti *object detection* dan *image classification*. Model-model ini awalnya dilatih pada *dataset* besar untuk mempelajari fitur dan representasinya secara umum. Kemudian model-model tersebut disesuaikan untuk tugas-tugas tertentu dengan *dataset* yang lebih kecil. Pendekatan ini secara signifikan mengurangi sumber daya komputasi dan waktu yang dibutuhkan untuk pelatihan, sekaligus meningkatkan kinerja pada tugas-tugas khusus karena transfer pengetahuan yang dipelajari dari model yang telah dilatih sebelumnya (Taneja, 2017). Pada penelitian ini, digunakan setidaknya empat model *pre-trained*, yaitu YOLOv9, Class-Attention in Image Transformers (CaiT), Swin Transformer, dan CoAtNet.

YOLOv9 adalah versi lanjutan dari YOLOv8 yang dirancang untuk meningkatkan akurasi dan efisiensi dalam deteksi objek. Dua inovasi utamanya adalah PGI (Programmable Gradient Information) untuk memperbaiki aliran gradien dan GELAN (Generalized Efficient Layer Aggregation Network) untuk ekstraksi fitur yang lebih baik.(Yaseen, 2024). Seluruh arsitektur PGI ditunjukkan pada **Gambar 2.2**.



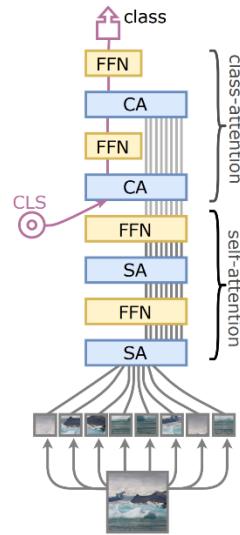
Gambar 2.2 Arsitektur PGI pada YOLOv9 (Yaseen, 2024)

Inovasi lain yang digunakan pada YOLOv9 adalah adanya arsitektur GELAN. Arsitektur tersebut menggabungkan dua arsitektur jaringan syaraf, yaitu Cross Stage Partial Network (CSPNet) dan Efficient Layer Aggregation Network (ELAN). CSPNet yang diintegrasikan dalam GELAN berfungsi untuk mengurangi redundansi komputasi sehingga untuk biaya komputasi inferensi dapat berkurang dan meningkatkan efisiensi pembelajaran dengan memisahkan aliran fitur menjadi dua bagian yang diproses secara parallel. Sementara itu, ELAN berkontribusi dalam mengagregasi fitur dari berbagai layer secara efisien, memungkinkan model untuk menangkap informasi multi-skala dengan lebih baik. Arsitektur GELAN juga memungkinkan YOLOv9 untuk mempertahankan informasi penting dari layer-layer awal sambil tetap memproses fitur kompleks di layer-layer yang lebih dalam. Kombinasi kedua komponen ini menghasilkan arsitektur yang tidak hanya efisien secara komputasi, tetapi juga mampu mempertahankan akurasi deteksi yang tinggi bahkan pada dataset yang kompleks. Arsitektur GELAN juga memungkinkan YOLOv9 untuk mempertahankan informasi penting dari layer-layer awal sambil tetap memproses fitur kompleks di layer-layer yang lebih dalam. Arsitektur keseluruhannya ditunjukkan pada **Gambar 2.3**.



Gambar 2.3 Arsitektur GELAN pada YOLOv9 (Yaseen, 2024)

Class-Attention in Image Transformers (CaiT) adalah inovasi pada model *Vision Transformer* (ViT) yang bertujuan meningkatkan performa dalam tugas klasifikasi gambar dengan memisahkan pemrosesan patch gambar dan interaksi dengan token CLS (*classification token*). Pada model ViT standar, token CLS dan *patch* gambar diproses bersama dalam satu tahap *self-attention* yang menyebabkan interaksi antara keduanya menjadi terikat (entangled) dan kurang optimal. Untuk mengatasi hal ini, CaiT memperkenalkan dua inovasi utama: *Class Attention* (CA) yang memisahkan pemrosesan CLS dan *patch* gambar, serta *LayerScale* untuk menangani stabilitas optimisasi pada model dengan kedalaman yang lebih besar. CaiT meningkatkan akurasi model pada tugas klasifikasi dengan mengurangi efek *bottleneck* dalam jaringan yang lebih dalam dan memungkinkan pengolahan yang lebih efisien. *Overview* arsitektur dari CaiT dapat dilihat pada **Gambar 2.4**.

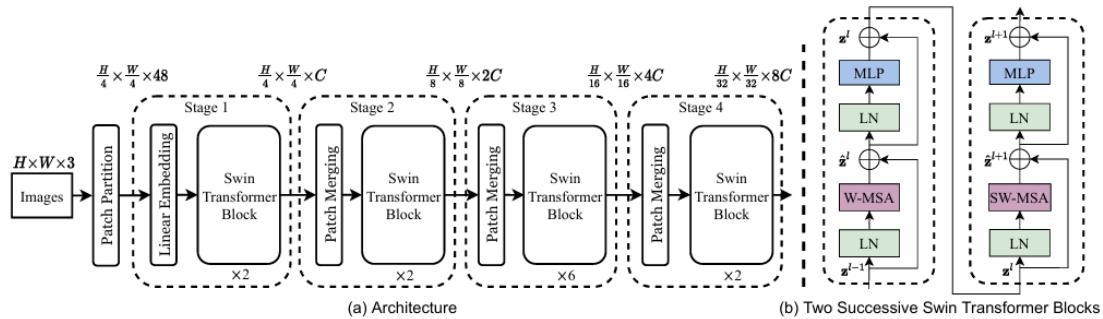


Gambar 2.4 Overview Arsitektur CaiT (Wang et al., 2023)

Dalam arsitektur CaiT, model ini mengadopsi prinsip *scaling depth* dengan menambah kedalaman *transformer* tanpa kehilangan stabilitas optimisasi. Setiap *transformer block* pada CaiT terdiri dari *self-attention* (SA) dan *feed-forward network* (FFN). Sebagai bagian dari modifikasi, *Class Attention* (CA) diterapkan dengan cara memisahkan pengolahan token CLS dan *patch* gambar. Token CLS diperkenalkan di tahap yang lebih akhir dalam jaringan dan

dihubungkan dengan *patch* gambar melalui lapisan perhatian khusus. Ini memisahkan tugas pemodelan gambar dan tugas klasifikasi yang memungkinkan model untuk lebih fokus pada informasi relevansi dalam token CLS. Selain itu, *LayerScale* digunakan untuk mengatasi masalah *bottleneck* dengan menstabilkan *residual connections* dalam *transformer block* yang lebih dalam.

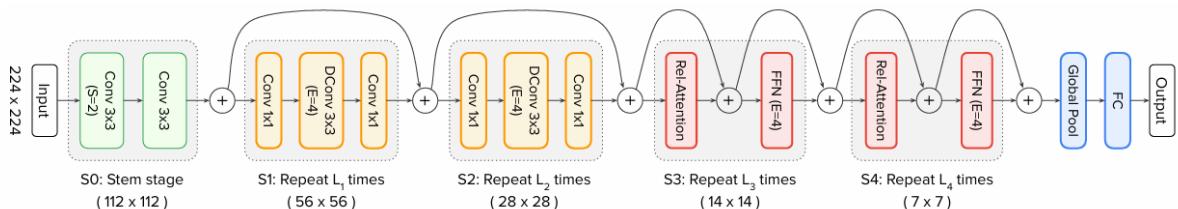
Swin Transformer adalah model berbasis *Transformer* yang dirancang khusus untuk tugas visi komputer dengan memanfaatkan teknik *self-attention* yang dimodifikasi. *Overview* arsitektur dari *Swin Transformer* dapat dilihat pada **Gambar 2.5**.



Gambar 2.5 Overview Arsitektur Swin Transformer (Liu et al., 2021)

Arsitektur *Swin Transformer* memulai pemrosesan gambar dengan membagi gambar RGB menjadi *patch-patch* kecil, kemudian melakukan *embedding* setiap *patch* ke dalam dimensi fitur yang lebih tinggi. Proses selanjutnya melibatkan beberapa *transformer blocks* yang menghitung *self-attention* pada jendela lokal (*window-based attention*), yang memungkinkan pengolahan informasi secara lebih efisien daripada *transformer* standar yang menggunakan perhatian global. Untuk menciptakan representasi hierarkis, jumlah token dikurangi secara bertahap menggunakan *patch merging layers* pada setiap tahap (*stage*), yang menurunkan resolusi gambar dan meningkatkan efisiensi komputasi. Dengan cara ini, *Swin Transformer* dapat menangani gambar dengan berbagai skala secara lebih efektif, sekaligus mempertahankan kemampuan model dalam menangkap hubungan global antar fitur gambar.

CoAtNet menggabungkan dua jenis pendekatan utama dalam deep learning, yaitu *Convolutional Neural Networks* (ConvNets) dan *Transformers*. Setiap pendekatan utamanya memiliki keunggulan sendiri. ConvNets memproses gambar dengan fokus pada bagian-bagian kecil (*patch* atau wilayah kecil gambar) untuk mengekstraksi fitur lokal seperti pola, tekstur, atau tepi. Lalu, hasil dari ConvNets (fitur-fitur lokal tadi) dijadikan input untuk *Transformers*. *Transformers* kemudian bekerja menghubungkan informasi antar fitur lokal tersebut untuk memahami hubungan antar bagian dari gambar yang lebih besar. *Overview* arsitektur dari CoAtNet dapat dilihat pada **Gambar 2.6**.

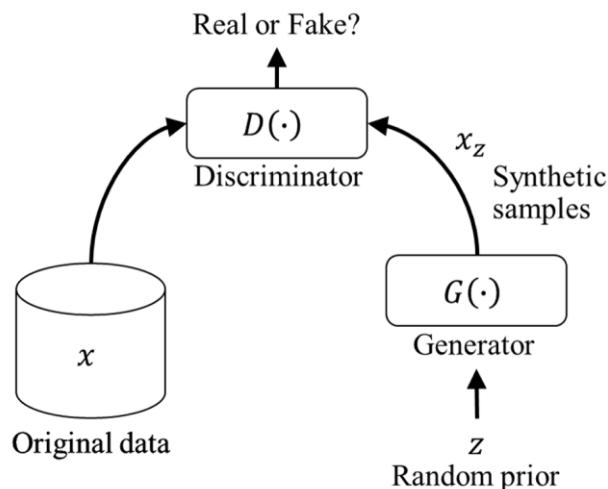


Gambar 2.6 Overview Arsitektur CoAtNet (Dai et al., 2021)

Arsitektur CoAtNet menggabungkan ConvNets dan *Transformers* untuk memproses gambar secara efisien. Dimulai dengan *Down-Sampling* di S0 hingga S2 menggunakan ConvNets, gambar secara bertahap mengalami reduksi dimensi dan ekstraksi fitur lokal. Pada S3 dan S4, model menggunakan 2D *Relative Attention* untuk menangkap hubungan global antar fitur, dengan penerapan *pre-norm* untuk stabilitas dan efisiensi. Seluruh blok ini mengadopsi *Pre-Activation*, di mana normalisasi dilakukan sebelum operasi utama untuk meningkatkan performa. Setelah fitur global ditangkap, gambar diteruskan ke *Global Pooling* dan *Fully Connected (FC) Layer*, menghasilkan output untuk tugas klasifikasi atau prediksi lainnya.

2.2.4 Generative Adversarial Network

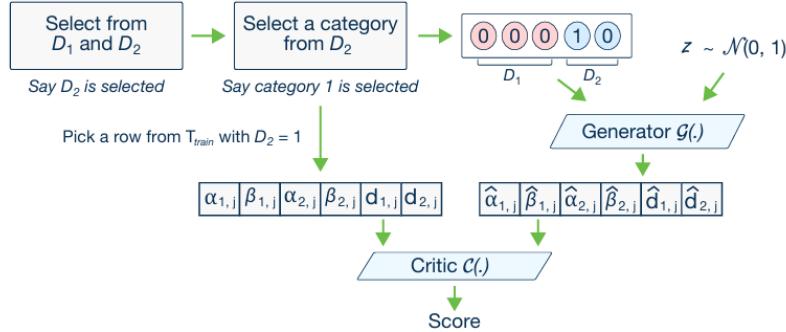
Generative Adversarial Network (GAN) adalah model pembelajaran yang melibatkan dua jaringan saraf, yaitu *generator* (G) dan *discriminator* (D). *Generator* bertugas menciptakan data sintetis yang menyerupai distribusi data asli, sementara *discriminator* bertugas membedakan apakah data tersebut berasal dari data asli atau palsu dari hasil dari *generator*. Tujuan *generator* adalah membuat data palsu yang semakin mirip dengan data asli sehingga mampu menipu *discriminator*, sedangkan tujuan *discriminator* adalah meningkatkan kemampuannya dalam membedakan data asli dan palsu (Baowaly, Lin, et al., 2019). Gambaran arsitektur dari GAN dapat dilihat pada **Gambar 2.7**.



Gambar 2.7 Arsitektur GAN (Baowaly, Liu, et al., 2019)

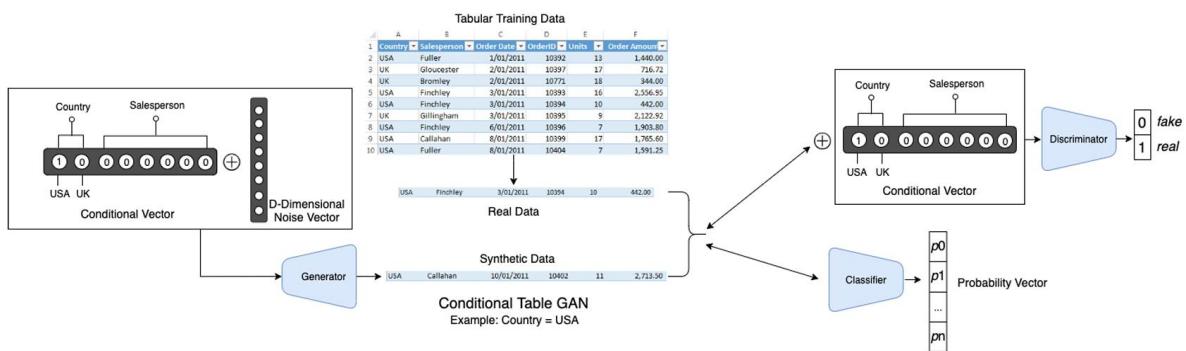
Conditional Tabular Generative Adversarial Network (CTGAN) adalah metode generatif berbasis GAN yang dirancang khusus untuk menghasilkan data tabular sintetik. Model ini dibuat untuk mengatasi tantangan yang dihadapi oleh GAN tradisional. Pada GAN biasa, generator akan menghasilkan data secara acak tanpa mempertimbangkan kondisi atau informasi khusus. Ini berarti, generator hanya belajar untuk menghasilkan data yang mengikuti distribusi keseluruhan dataset tanpa memperhatikan perbedaan antara kategori atau nilai spesifik dalam data. Oleh karena itu, CTGAN diperkenalkan dengan teknik baru seperti normalisasi spesifik mode, generator kondisional, dan pelatihan dengan sampling, untuk menghasilkan sampel yang lebih realistik dan berkualitas tinggi dari data tabular yang lebih kompleks (Skoularidou & Cuesta-infante, 2019). Keunggulan utama CTGAN terletak pada kemampuannya untuk mempertahankan korelasi statistik antar variabel dalam data asli, sehingga data sintetis yang dihasilkan tidak hanya realistik secara individual tetapi juga konsisten secara relasional. Selain itu, CTGAN mampu menangani ketidakseimbangan distribusi data dengan lebih baik melalui mekanisme conditional vector yang memungkinkan kontrol yang lebih presisi terhadap proses

generasi data. Hal ini menjadikan CTGAN sebagai solusi yang efektif untuk aplikasi yang memerlukan data sintetis berkualitas tinggi, terutama dalam konteks data augmentation dan privacy-preserving data sharing. Pada **Gambar 2.8** diketahui *generator* dari ctgan dapat membuat data sintetis berdasarkan kolom dan kategori yang ditentukan.



Gambar 2.8 Arsitektur CTGAN (Skoularidou & Cuesta-infante, 2019)

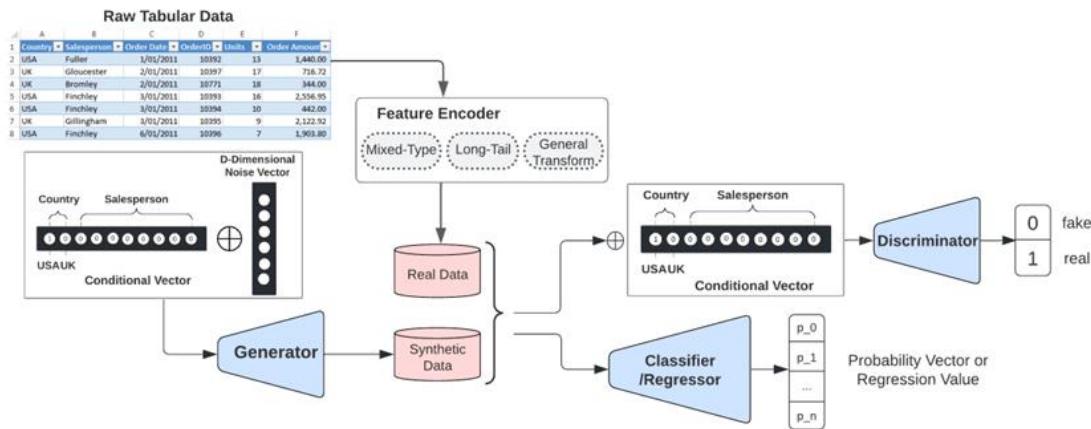
CTGAN kemudian dikembangkan menjadi CTAB-GAN. Model berbasis GAN tersebut dirancang untuk mengatasi beberapa keterbatasan yang ada pada CTGAN, terutama dalam menangani data tabular yang lebih kompleks yang mencakup campuran antara variabel kontinu dan kategorikal. CTAB-GAN memperkenalkan teknik tambahan seperti information loss dan classification loss, yang tidak ada pada CTGAN. Pada **Gambar 2.9**, arsitektur CTAB-GAN terdiri dari tiga komponen utama, yaitu *Generator*, *Discriminator*, dan *Classifier*. *Generator* menghasilkan data sintetis dengan menambahkan dua jenis *loss* ketika pelatihan, yaitu *information loss* yang memastikan data sintetis memiliki statistik yang mirip dengan data asli dan *Classification Loss* yang menjaga agar data sintetis tetap semantik dengan data asli. *Discriminator* berfungsi untuk membedakan antara data asli dan sintetis. Sementara itu, *Classifier* dilatih menggunakan data asli untuk memastikan data sintetis mempertahankan hubungan yang benar antar fitur dan kelas (Zhao et al., 2017).



Gambar 2.9 Arsitektur CTAB-GAN (Zhao et al., 2017)

CTAB-GAN+ merupakan pengembangan dari CTAB-GAN dengan beberapa pembaruan pada arsitekturnya. Modifikasi utama terletak pada penggantian komponen *Classifier* menjadi *Auxiliary Component* yang memiliki fleksibilitas sebagai *classifier* maupun *regressor*. Arsitektur ini juga mengalami perluasan kompleksitas loss function dari dua komponen menjadi tiga komponen, meliputi *information loss*, *downstream loss*, dan *generator loss*. Keunggulan lain dari CTAB-GAN+ terletak pada integrasi *differential privacy* melalui algoritma DP-SGD yang dikombinasikan dengan *sub-sampling strategy*. Fitur ini tidak ada pada CTAB-GAN dan memungkinkan pengendalian *privacy budget* dengan tetap

mempertahankan kualitas tinggi data sintetis. Seluruh pengembangan ini memberikan fleksibilitas dalam penanganan tugas regresi, berbeda dengan CTAB-GAN yang orientasinya lebih terbatas pada tugas klasifikasi (Zhao, Kunar, et al., 2023). Adapun arsitektur dari CTAB-GAN+ terlihat pada **Gambar 2.10**.

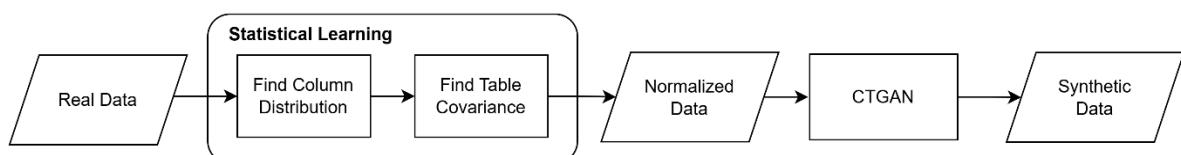


Gambar 2.10 Arsitektur CTABGAN+ (Zhao, Kunar, et al., 2023)

CopulaGAN merupakan jenis *synthesizer data* yang menggabungkan metode statistik dengan teknik *deep learning* berbasis GAN, yaitu *Conditional GAN* (CTGAN), dalam melatih model dan menghasilkan data sintetis. Pendekatan ini dilakukan dalam dua tahap utama, yaitu *statistical learning* dan *GAN-based learning*. Pada tahap pertama, *synthesizer* mempelajari distribusi marginal dari setiap kolom data asli menggunakan berbagai jenis distribusi probabilitas, seperti distribusi *beta*, *uniform*, *truncated Gaussian*, dan *exponential*. Setelah distribusi optimal untuk setiap kolom teridentifikasi melalui uji Kolmogorov-Smirnov, data kemudian ditransformasi menggunakan teknik Gaussian Copula dengan formula yang sesuai pada **Persamaan 2.1**.

$$Y = \Phi^{-1}(F_0(x_0)), \Phi^{-1}(F_1(x_1)), \dots, \Phi^{-1}(F_n(x_n)) \quad (2.1)$$

$F_i(x_i)$ pada persamaan tersebut adalah *cumulative distribution function* (CDF) distribusi marginal kolom i dan Φ^{-1} adalah *inverse CDF* distribusi Gaussian. Transformasi ini menghasilkan data dengan distribusi Gaussian standar (*mean* $\mu = 0$, deviasi standar $\sigma = 1$) dengan tetap mempertahankan struktur dependensi antar kolom melalui *covariance matrix* yang dihitung dari data hasil transformasi. Pada tahap kedua (*GAN-based Learning*), CTGAN melatih data yang telah dinormalisasi untuk menghasilkan data sintetis yang mampu mereplikasi baik karakteristik distribusi marginal maupun struktur korelasi dari data asli. Arsitektur CopulaGAN dapat dilihat pada **Gambar 2.11**.



Gambar 2.11 Arsitektur CopulaGAN

2.2.5 Multimodal Fusion

Pembelajaran mesin multimodal bertujuan untuk membangun model yang dapat secara efektif memproses dan menghubungkan informasi dari berbagai modalitas ini. Hal ini sangat penting karena setiap modalitas dapat memberikan wawasan unik yang, jika digabungkan, dapat meningkatkan pemahaman dan kinerja sistem pembelajaran mesin secara keseluruhan. Misalnya, dalam bidang kesehatan, pengintegrasian data citra medis dengan catatan kesehatan elektronik telah menunjukkan peningkatan yang signifikan dalam akurasi diagnostik untuk kondisi seperti penyakit Alzheimer, yang menunjukkan kegunaan teknik fusi multimodal (Huang et al., 2020). Demikian pula, di bidang radiologi, pembelajaran mendalam multimodal telah terbukti meningkatkan interpretasi gambar dengan memasukkan data klinis tambahan, sehingga meningkatkan kemampuan diagnostik (Heiliger et al., 2023).

Proses yang perlu digarisbawahi pada multimodal adalah kapan dan bagaimana multimodal tersebut digabungkan. Terdapat empat strategi dari penggabungan modalitas, yaitu *early fusion*, *intermediate fusion*, *late fusion*, dan *hybrid fusion*. Strategi pertama, *early fusion*, modalitas digabungkan menjadi representasi tunggal pada level input yang kemudian dilakukan *modelling*. Strategi kedua, *intermediate fusion*, setiap modalitas diproses secara terpisah ke dalam representasi yang lebih abstrak, lalu digabungkan untuk kemudian masuk ke proses *modelling*. Strategi ketiga, *late fusion*, setiap modalitas diproses hingga tahap *modelling* terlebih dahulu. Lalu, output dari masing-masing proess *modelling* nya digabungkan, mirip seperti *ensemble learning*. Strategi penggabungan yang terakhir, *hybrid fusion*, adalah kombinasi dari ketiga strategi sebelumnya,yaitu *early*, *intermediate*, dan *late fusion*.

2.2.6 Ensemble Learning

Pendekatan *ensemble learning* didasarkan pada prinsip bahwa penggabungan hasil prediksi dari beragam model dapat meningkatkan akurasi dan ketahanan dibandingkan dengan mengandalkan satu model. Ide dasarnya adalah bahwa model yang berbeda dapat menangkap aspek yang berbeda dari data, dan dengan mengintegrasikan output mereka, ensemble dapat mengurangi kelemahan model individu dan meningkatkan kekuatan mereka (Sagi & Rokach, 2018).

Ada berbagai strategi untuk menerapkan *ensemble leanring*, yaitu *bagging*, *boosting*, dan *stacking*. *Bagging*, atau agregasi bootstrap, melibatkan pelatihan beberapa model yang sama pada subset data yang berbeda. Hal ini membantu mengurangi varians dan mengurangi kemungkinan *overfitting* (Dietterich, 2000). Di sisi lain, *Boosting* berfokus pada pelatihan model secara berurutan, di mana setiap model baru mencoba untuk memperbaiki kesalahan yang dibuat oleh model sebelumnya, sehingga meningkatkan akurasi model secara keseluruhan (Gutierrez-espinoza et al., 2020). *Stacking* melibatkan pelatihan beberapa model dan kemudian menggunakan model lain untuk menggabungkan prediksi mereka, yang dapat menghasilkan peningkatan kinerja yang lebih besar (Bakasa & Viriri, 2021).

2.2.7 Evaluasi Performa

Confusion Matrix adalah salah satu jenis matriks yang memungkinkan developer dalam melakukan evaluasi dari model yang telah dibuat. *Confusion matrix* bertujuan untuk melihat dan menyimpulkan performa model dilihat dari beberapa perspektif yang berbeda. Matriks ini menyajikan informasi tentang prediksi yang benar dan salah yang dibuat oleh model klasifikasi dengan membandingkan hasil prediksi dengan label sebenarnya dari data uji. Melalui *confusion matrix*, dapat dihitung berbagai metrik evaluasi seperti akurasi, presisi, *recall*, dan *F1-score*

yang memberikan gambaran menyeluruh tentang kinerja model. *Confusion matrix* untuk klasifikasi biner ditunjukkan pada **Tabel 2.2**.

Tabel 2.2 *Confusion Matrix*

		Kelas Aktual	
		<i>True</i>	<i>False</i>
Kelas Prediksi	<i>Positive</i>	<i>True Positive (TP)</i>	<i>False Positive (FP)</i>
	<i>Negative</i>	<i>False Negative (FN)</i>	<i>True Negative (TN)</i>

True Positive (TP) dan *True Negative (TN)* merupakan jumlah kelas positif dan negatif yang diklasifikasikan dengan tepat atau dalam arti lain hasil prediksi sesuai dengan data aktual. Sedangkan, *False Positive (FP)* dan *False Negative (FN)* merupakan jumlah kelas positif dan negatif yang tidak diklasifikasikan dengan tepat. FP berarti hasil prediksi kelas positif yang bernilai negatif pada data aktual. Sementara itu, FN berarti hasil prediksi kelas negatif yang bernilai positif pada data aktual. Dari *confusion matrix*, dapat ditarik nilai metriks lain, di antaranya akurasi, presisi, *recall*, dan *F1-score*.

Metriks yang paling umum digunakan adalah akurasi karena hanya menghitung banyaknya prediksi yang benar dibagi dengan total jumlah sampel, seperti yang terlihat pada **Persamaan 2.1**. Namun untuk beberapa kasus, akurasi dinilai kurang cocok untuk dijadikan metriks utama yang menentukan perfoma model. Contohnya pada kasus data medis dimana seringkali terjadi ketidakseimbangan data antara diagnosa positif dan diagnosa negatif. Data dengan diagnosa negatif biasanya berjumlah jauh lebih sedikit.

$$\text{Accuracy} = \frac{(TP+TN)}{(TP+FP+TN+FN)} \quad (2.2)$$

Recall (juga disebut dengan sensitivitas) mengevaluasi banyaknya hasil prediksi positif yang benar terhadap banyaknya nilai yang seharusnya diklasifikasikan benar, seperti terlihat pada **Persamaan 2.2**. Metriks ini banyak digunakan pada data medis karena temuan nilai FN sangatlah krusial. Sebagai contoh, ketika seseorang yang seharusnya terdiagnosa positif suatu penyakit, namun model salah memprediksi menjadi negatif. Ke depannya hal tersebut akan berakibat fatal bagi penderita.

$$\text{Recall} = \frac{TP}{(TP+FN)} \quad (2.3)$$

$$\text{Precision} = \frac{TP}{(TP+FP)} \quad (2.4)$$

Presisi (juga disebut dengan spesifisitas) mengevaluasi banyaknya hasil prediksi positif yang benar terhadap banyaknya nilai yang diprediksi positif, seperti terlihat pada **Persamaan 2.3**. Metriks ini juga banyak digunakan pada data medis dan merupakan kebalikan dari *recall*.

Jika *recall* ingin mengetahui seberapa sering model salah dalam memprediksi diagnosa yang seharusnya positif, presisi ingin mengetahui seberapa sering model salah dalam memprediksi diagnosa yang seharusnya negatif.

$$F1\text{-score} = \frac{(2 \times \text{precision} \times \text{recall})}{(\text{precision} + \text{recall})} \quad (2.5)$$

F1-score merepresentasikan keseimbangan antara nilai *recall* dan presisi, seperti terlihat pada **Persamaan 2.4**. Metriks ini dapat memberikan gambaran terkait performa model secara keseluruhan. Hal tersebut untuk memberikan gambaran yang lebih akurat tentang performa model pada kelas minoritas yang mungkin lebih penting secara klinis. *F1-score* biasanya lebih cocok digunakan ketika kelas target tidak seimbang (*imbalanced classes*), yaitu jumlah sampel positif dan negatif sangat tidak seimbang.

BAB 3 METODOLOGI

3.1 Metode yang digunakan

Penelitian ini menggunakan pendekatan multimodal dengan menggabungkan data klinis dan citra *effluent dialysate* untuk membuat sistem klasifikasi pasien CAPD setelah melakukan ganti cairan. Terdapat dua kelas klasifikasi, yaitu normal dan abnormal. Kedua kelas tersebut direpresentasikan dengan angka, seperti angka 0 untuk normal dan 1 untuk abnormal. Data klinis terlebih dahulu dianalisis distribusinya melalui *exploratory data analysis* (EDA). Lalu, proses dilanjutkan dengan *data preprocessing*, seperti penanganan nilai *null* dan *feature engineering*. Sementara itu, data citra mengalami tahap EDA sederhana untuk mengetahui distribusi ukuran dari setiap *image* nya. Pada tahap *image preprocessing*, setiap gambar mendapatkan setidaknya dua perlakuan, yaitu *bag detection* dan penyesuaian ukuran gambar. Setelah itu dilakukan ekstraksi fitur menggunakan empat *pre-trained model*, yaitu Swin Transformer, CaiT, CoAtNet, atau YOLOv9. Setelah fitur gambar berhasil diekstrak, fitur tersebut digabungkan dengan data klinis yang telah dilakukan *data preprocessing* menggunakan metode *early fusion*.

Data yang sudah digabungkan kemudian dilakukan proses *data splitting* menjadi data latih dan data uji menggunakan *stratified split* dengan rasio 75:25 untuk menjaga proporsi kelas. Selanjutnya, data latih dilakukan *dimensionality reduction* menggunakan *principal component analysis* (PCA) untuk mengurangi kompleksitas fitur dan meningkatkan efisiensi model. Untuk mengatasi ketidakseimbangan dan keterbatasan jumlah data, terutama pada kelas minoritas, data uji yang telah mengalami PCA dilakukan *synthetic data generation* menggunakan empat model berbasis GAN, yaitu CTGAN, CTABGAN, CTABGAN+, dan CopulaGAN. Setelah proses tersebut, data sintetik digabungkan dengan data asli sebelum dilakukan proses normalisasi. Data gabungan yang telah dinormalisasi tersebut menjadi input *training* model klasifikasi dengan beberapa algoritma *supervised learning*, seperti *Random Forest*, XGBoost, dan *Support Vector Machine*. Kinerja model kemudian dievaluasi menggunakan metrik seperti *precision*, *recall*, dan *F1-score* untuk menilai efektivitas klasifikasi terhadap data uji.

3.2 Dataset yang Digunakan

Terdapat dua bentuk data yang digunakan pada penelitian ini, yaitu data gambar dan tabular. *Dataset* gambar berisi kumpulan citra luaran terapi CAPD yang disebut *effluent dialysate*. *Dataset* citra *effluent dialysate* tersebut diambil secara langsung dari pasien PGTA yang menjalani terapi CAPD di Rumah Sakit Universitas Airlangga Surabaya dan Rumah Sakit Umum Pusat Dr. Sardjito Yogyakarta. Beberapa contoh citra *effluent dialysate* yang didapatkan dari penelitian sebelumnya terdiri dari kategori normal dan abnormal, ditunjukkan pada **Gambar 3.1**. *Dataset* diambil dari *database* aplikasi SahabatCAPD dengan persetujuan keamanan oleh kedua rumah sakit yang telah bekerja sama tersebut. Setiap citra *effluent dialysate* diambil menggunakan kamera ponsel masing-masing pasien yang menjalani terapi CAPD. Data klinis tabular diambil dari sumber yang sama dengan dataset citra dan memiliki label yang sama dengan citra, yaitu normal dan abnormal, serta saling berkorelasi. Data tersebut mencakup berbagai kondisi klinis pasien, seperti nadi, tekanan darah *systolic*, dan tekanan darah *diastolic*. Kombinasi data gambar dan tabular ini memungkinkan pengembangan model *machine learning* dengan konsep multimodal.



Normal

Abnormal

Gambar 3.1 Sampel Citra Effluent Dialysate ((Navastara et al., 2023)

Pada *dataset* tabular, *dataset* yang digunakan berupa data klinis *monitoring* mandiri. Sebelumnya, data klinis hasil monitoring mandiri dicatat pada logbook atau buku catatan *monitoring* penggantian CAPD pasien, ditunjukkan pada **Gambar 3.2** (Jalil, 2023).

CATATAN HARIAN CAPD																																																																		
SENIN					KAMIS																																																													
%	Masuk		Keluar		Balance		Kumulatif		Tanggal		%	Masuk		Keluar		Balance		Kumulatif		Tanggal																																														
	Volume	Jam	Volume	Jam	(+)	(-)	(+)	(-)	Waktu	Diurexis	Catatan		Volume	Jam	Volume	Jam	(+)	(-)	(+)	(-)	Waktu	Diurexis	Catatan																																											
											Tek. Darah											Tek. Darah																																												
											Berat Badan											Berat Badan																																												
											Exit Site											Exit Site																																												
SELASA					JUMAT						SABTU																																																							
%	Masuk		Keluar		Balance		Kumulatif		Tanggal		%	Masuk		Keluar		Balance		Kumulatif		Tanggal		%	Masuk		Keluar		Balance		Kumulatif		Tanggal																																			
	Volume	Jam	Volume	Jam	(+)	(-)	(+)	(-)	Waktu	Diurexis	Catatan		Volume	Jam	Volume	Jam	(+)	(-)	(+)	(-)	Waktu	Diurexis	Catatan		Volume	Jam	Volume	Jam	(+)	(-)	(+)	(-)	Waktu	Diurexis	Catatan																															
											Tek. Darah											Tek. Darah											Berat Badan											Berat Badan											Exit Site											Exit Site
RABU					MINGGU						SENIN																																																							
%	Masuk		Keluar		Balance		Kumulatif		Tanggal		%	Masuk		Keluar		Balance		Kumulatif		Tanggal		%	Masuk		Keluar		Balance		Kumulatif		Tanggal																																			
	Volume	Jam	Volume	Jam	(+)	(-)	(+)	(-)	Waktu	Diurexis	Catatan		Volume	Jam	Volume	Jam	(+)	(-)	(+)	(-)	Waktu	Diurexis	Catatan		Volume	Jam	Volume	Jam	(+)	(-)	(+)	(-)	Waktu	Diurexis	Catatan																															
											Tek. Darah											Tek. Darah											Berat Badan											Berat Badan											Exit Site											Exit Site

Gambar 3.2 Buku Catatan Monitoring Penggantian CAPD

Pada data klinis *monitoring* mandiri tersebut sekarang sudah diambil melalui aplikasi Sahabat CAPD. Dalam database aplikasi SahabatCAPD, terdapat tujuh tabel yang digunakan pada model penelitian ini, yaitu tabel cairan, pasien, penggantian, detail penggantian, resep, keluhan, dan detail keluhan. Ketujuh tabel tersebut lalu digabungkan dan dilakukan beberapa proses seperti penghapusan beberapa kolom dan baris yang tidak dibutuhkan sebelum masuk ke proses *data preprocessing*. Salah satu kolom yang mengalami penghapusan adalah kolom-kolom *id* dari setiap tabel karena dianggap hanya sebagai *identifier* dan tidak begitu dibutuhkan selama proses *modelling*. Setelah dilakukan proses *data preprocessing*, tabel data klinis yang digunakan mengalami perubahan yang signifikan baik jumlah baris maupun kolomnya. Pada

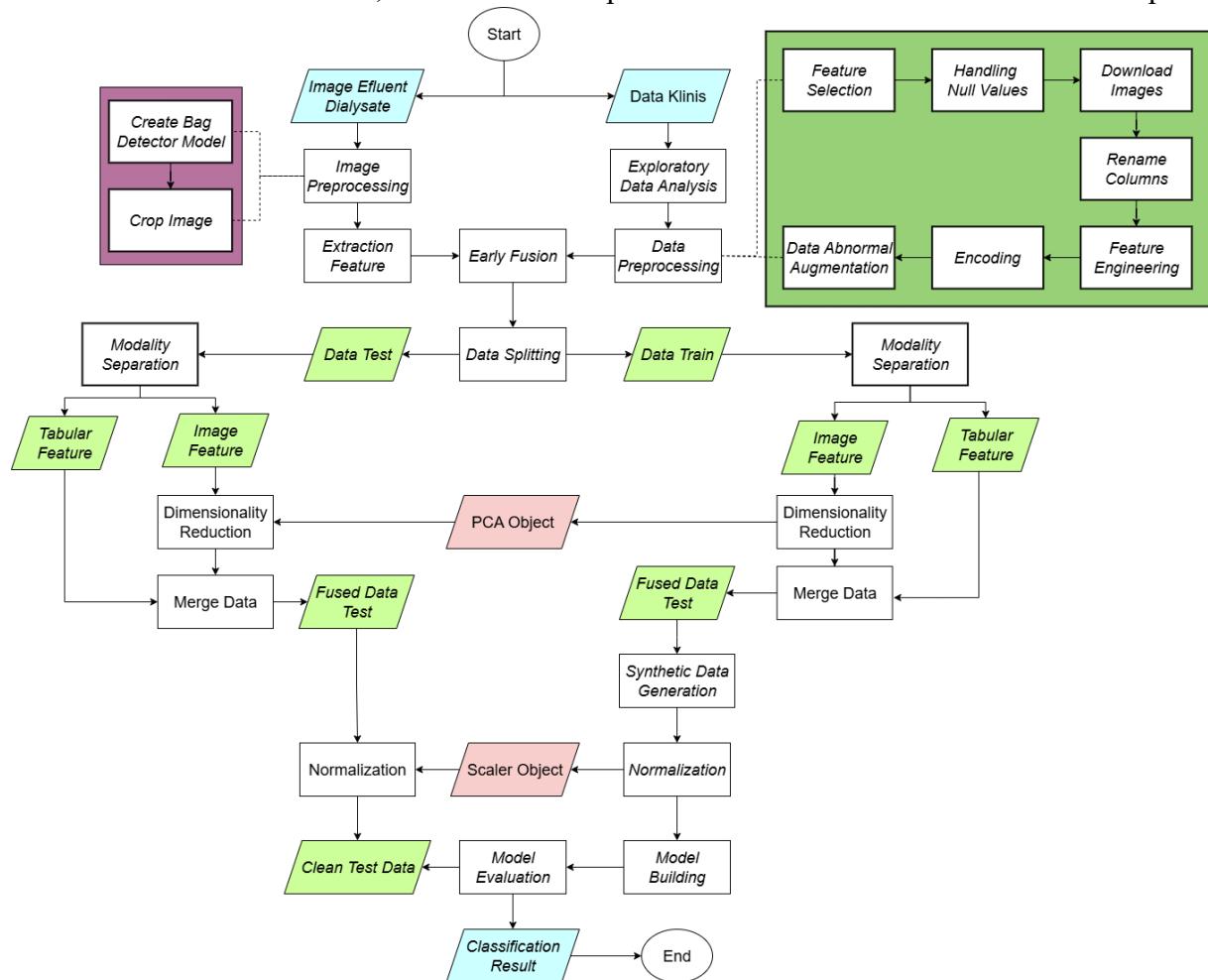
Tabel 3.1, dapat dilihat fitur-fitur dari data klinis setelah *data preprocessing*. Seluruh data tersebut, selanjutnya akan dilakukan penggabungan dengan fitur gambar yang telah diekstraksi menjadi *feature map* atau *vector*.

Tabel 3.1 Fitur-Fitur pada Data Klinis Pasien CAPD

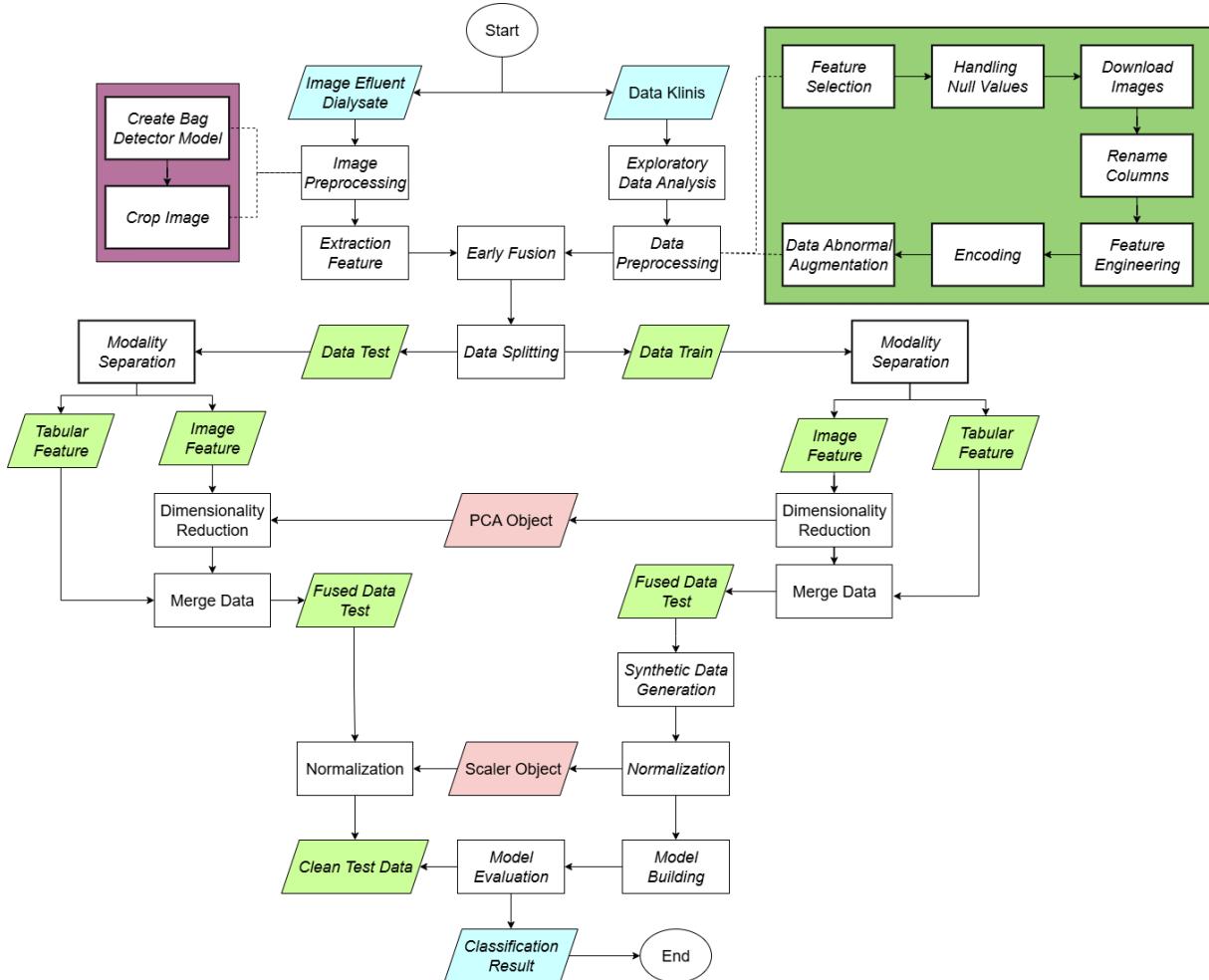
Nama Fitur	Jenis Data	Keterangan
Durasi drain menit	Numerikal	<ul style="list-style-type: none"> Selisih saat proses mengeluarkan cairan yang sudah ada di dalam dengan proses memasukkan cairan yang baru
Volume masuk	Numerikal	<ul style="list-style-type: none"> Volume Cairan yang dimasukkan pada setiap penggantian
Volume keluar	Numerikal	<ul style="list-style-type: none"> Volume cairan yang dikeluarkan pada setiap penggantian
Berat badan	Numerikal	<ul style="list-style-type: none"> Berat badan dari pasien
<i>systolic</i>	Numerikal	<ul style="list-style-type: none"> Tekanan darah saat jantung memompa darah keluar atau biasa disebut tekanan darah mm
<i>diastolic</i>	Numerikal	<ul style="list-style-type: none"> Tekanan darah saat jantung dalam keadaan istirahat atau biasa disebut tekanan darah Hg
nadi	Numerikal	<ul style="list-style-type: none"> Nilai nadi pasien
Jenis kelamin	Kategorikal	<ul style="list-style-type: none"> Jenis kelamin pasien, laki-laki atau perempuan
Jam masuk	Numerikal	<ul style="list-style-type: none"> Nilai jam saat pasien memasukkan cairan
Hari masuk	Kategorikal	<ul style="list-style-type: none"> Hari saat pasien memasukkan cairan
Dwell time menit	Numerikal	<ul style="list-style-type: none"> Durasi cairan berada di dalam tubuh pasien Didapatkan dengan mengurangi waktu keluar dan waktu masuk dalam format detik
Jam keluar	Kategorikal	<ul style="list-style-type: none"> Nilai jam saat pasien mengeluarkan cairan
Hari keluar	Kategorikal	<ul style="list-style-type: none"> Nilai jam saat pasien memasukkan cairan
Selisih volume	Numerikal	<ul style="list-style-type: none"> Selisih volume cairan Didapatkan dengan mengurangi volume masuk dan volume keluar.
Shift penggantian	Kategorikal	<ul style="list-style-type: none"> <i>Shift</i> penggantian cairan Terdapat empat <i>shift</i>, yaitu pagi, siang, sore, dan malam
Label	Kategorikal	<ul style="list-style-type: none"> Kondisi pasien Terdapat dua kondisi, normal (0) dan abnormal (1)

3.3 Perancangan Sistem

Penelitian ini menerapkan konsep *multimodal early fusion* dimana terdapat proses penggabungan fitur data klinis monitoring dan citra *effluent dialysate* yang telah dilakukan ekstraksi fitur, seperti terlihat pada



Gambar 3.3.



Gambar 3.3 Diagram Alir Proses Penelitian

Proses pembuatan data sintetik hanya terjadi pada data *train*. Hal tersebut bertujuan agar pada tahap evaluasi model hanya terdapat data asli yang menjadi data ujinya.

3.3.1 *Data Preprocessing* pada Data Klinis

Total data klinis yang berhasil dikumpulkan oleh peneliti adalah sejumlah 6.730 baris dengan komposisi 891 baris berupa data dengan label abnormal, 4.137 baris berupa data dengan label normal, dan 1.011 baris tidak memiliki label atau *null*. Data tersebut diambil dari database aplikasi SahabatCAPD yang merupakan penggabungan beberapa tabel, seperti tabel cairan, pasien, penggantian, detail_penggantian, resep, keluhan, dan detail_keluhan. Data hasil penggabungan beberapa tabel tersebut memiliki 49 kolom yang salah satu kolomnya adalah *link* untuk mengunduh cairan *effluent dialysate*. Setelah data digabungkan, masih perlu dilakukan *data preprocessing* sebelum dapat digunakan.

Proses pertama setelah data digabungkan adalah *feature selection*. Beberapa kolom tidak memiliki fungsi yang signifikan sehingga perlu di-*drop*, seperti kolom-kolom id dari setiap tabel dan kolom-kolom yang mayoritas *valuenya* berupa nilai *nul*. Beberapa kolom juga perlu tetap dijaga karena beberapa penyebab, seperti kolom-kolom yang memiliki kaitan dengan penelitian sebelumnya dan kolom-kolom yang biasa digunakan di dunia medis. Setelah dilakukan *feature selection* dan penghapusan baris yang memiliki nilai *null* pada kolom-kolom yang perlu dijaga, dataset tersisa 342 data normal dan 41 data abnormal. Lalu, dilakukan

pengunduhan citra *effluent dialysate* menggunakan *url* pada salah satu kolom yang ada di *dataset*. Setelah citra diunduh, dilakukan penghapusan kolom-kolom yang berisi *url* citra *effluent dialysate* karena sudah tidak terpakai.

Selanjutnya, beberapa kolom dilakukan *rename* untuk menyesuaikan nama kolom pada penelitian sebelumnya. Lalu, dilakukan proses *feature engineering* untuk memperbanyak jumlah kolom pada data klinis agar tidak berbeda secara signifikan ketika digabungkan dengan data gambar. Beberapa kolom hasil proses *feature engineering* tersebut antara lain *jam_masuk* yang diekstrak dari nilai *jam* pada kolom *waktu_masuk*, lalu ada kolom *hari masuk*, *dwell_time_menit*, *jam_keluar*, *hari_keluar*, *selisih_volume*, dan *shift_penggantian*. Setelah itu, dilakukan *encoding* untuk kolom-kolom kategorikal.

Terakhir, jumlah data abnormal setelah dilakukan *splitting* tidak mencukupi sebagai input dari salah satu metode GAN yang dipakai sehingga perlu dilakukan *data augmentation* sederhana. Beberapa metode yang dipakai untuk augmentasi data adalah *flip* gambar secara vertikal, penambahan gambar abnormal yang belum dipakai, dan operasi matematika sederhana pada beberapa kolom di data klinis. Setelah proses tersebut, *datasetnya* berjumlah 342 data normal dan 123 data abnormal. Seluruh data tambahan tersebut telah divalidasi oleh dokter di bidang terkait.

3.3.2 *Image Preprocessing* pada Cairan *Effluent Dialysate*

Dataset citra *effluent dialysate* disimpan dengan memisahkan citra normal dan abnormal di folder yang berbeda. Citra *effluent dialysate* yang tersedia memiliki *background* yang beraneka ragam dan terdapat beberapa objek lain, seperti kardus, lemari, dan bantal. Oleh karena itu, dilakukan deteksi *bag effluent dialysate*. Setelah dilakukan deteksi tersebut, citra kemudian di-*cropped* pada *bag effluent dialysate* saja agar citra dapat terfokus pada *bag effluent dialysate* sehingga meminimalisasi *noise*.

3.3.3 *Feature Extraction*

Sebelum citra *effluent dialysate* digabungkan dengan data klinis, perlu dilakukan *feature extraction* terlebih dahulu. Proses tersebut menggunakan empat *pre-trained model*, yaitu *Swin Transformer*, *Class-Attention in Image Transformers* (*CaiT*), *CoAtNet*, dan *YOLOv9*. Keempat pre-trained model tersebut dipilih karena perbedaan arsitektur dasarnya. *Swin Transformer* dan *CaiT* memakai transformer sebagai basis arsitekturnya. *CoAtNet* menggunakan kombinasi arsitektur transformer dan CNN. *YOLOv9* memakai CNN sebagai arsitektur dasarnya. Dengan begitu, penggunaan arsitektur transformer dan CNN pada kasus *feature extraction* dapat diketahui kefektifannya.

Keempat *pre-trained model* tersebut memiliki format ukuran input gambarnya masing-masing. Oleh karena itu, sebelum dilakukan *feature extraction*, terlebih dahulu diimplementasikan penyeragaman ukuran gambar sesuai dengan ketentuan input setiap *pre-trained model*. *Swin Transformer*, *CaiT*, dan *CoAtNet* memiliki ukuran input sebesar 224x224. Sedangkan, *YOLOv9* memiliki ukuran input sebesar 640x640. Setelah dilakukan *resize*, citra *effluent dialysate* disimpan di dalam *DataLoader*, fitur pendukung dari PyTorch untuk mempermudah proses mengelola dan memuat data ke model dalam bentuk *batch*. Selanjutnya, seluruh gambar yang disimpan di dalam *DataLoader* dilakukan *feature extraction* menggunakan keempat *pre-trained model*. Hasil dari proses *feature extraction* tersebut selanjutnya digabungkan dengan data klinis lewat proses *early fusion*.

3.3.4 Early Fusion

Proses *early fusion* merupakan tahapan krusial dalam penggabungan data multimodal. Hasil ekstraksi fitur citra *effluent dialysate* diintegrasikan dengan data klinis pasien CAPD pada tahap awal sebelum proses pemodelan dilakukan. Dalam pendekatan ini, vektor fitur yang dihasilkan dari proses *feature extraction* menggunakan keempat *pre-trained model* digabungkan secara langsung dengan data numerik dan kategorikal dari rekam medis pasien. Penggabungan ini dilakukan dengan cara menggabungkan vektor fitur dari citra dengan data klinis yang telah melewati proses *data preprocessing*. Dengan begitu, kedua jenis data bisa diperlakukan sebagai satu kesatuan input yang terintegrasi dalam proses pembelajaran model.

Keuntungan utama dari metode *early fusion* adalah kemampuannya untuk menangkap hubungan kompleks antara karakteristik visual dari citra *effluent dialysate* dan parameter klinis pasien sejak awal proses pemodelan. Dengan melakukan fusi di tahap awal, model dapat mempelajari korelasi dan interaksi antar modalitas data yang mungkin tidak terdeteksi jika kedua jenis data diproses secara terpisah. Hal ini memungkinkan adanya representasi yang lebih komprehensif dari kondisi pasien, yang menggabungkan informasi visual tentang kekeruhan cairan dialisis dengan indikator klinis. Pendekatan *early fusion* ini kemudian dilanjutkan ke tahap selanjutnya, yaitu *data splitting*.

3.3.5 Data Splitting

Setelah proses *early fusion* selesai dan data gabungan sudah tersedia, tahap selanjutnya adalah melakukan proses *data splitting*. Pembagian data ini dilakukan di awal sebelum proses pelatihan model klasifikasi agar dapat meminimalkan risiko *data leakage*, yaitu kondisi di mana informasi dari data uji secara tidak sengaja memengaruhi proses pelatihan dan evaluasi model. Selain itu, dengan memisahkan data uji di awal, tahapan-tahapan selanjutnya dipastikan hanya melibatkan data latih. Ini penting agar dapat mensimulasikan kondisi nyata setelah model digunakan, data yang menjadi input belum pernah diketahui sebelumnya.

Skema pembagian data dilakukan secara *stratified split* dengan rasio 75:25. Data latih berjumlah 75% dari keseluruhan data dan 25% sisanya sebagai data uji. Stratifikasi dilakukan berdasarkan label kelas (normal dan abnormal) agar proporsi kelas tetap seimbang di kedua subset data. Proses ini memastikan bahwa hasil evaluasi model mencerminkan kinerja yang lebih objektif dan tidak bias akibat paparan data uji selama pelatihan.

3.3.6 Dimensionality Reduction

Setelah proses early fusion selesai, jumlah fitur gabungan dari data hasil *feature extraction* dan data tabular klinis menjadi sangat banyak. Hal tersebut disebabkan oleh model *pre-trained* yang digunakan untuk ekstraksi fitur menghasilkan vektor berdimensi besar. Hal ini dapat menyebabkan *curse of dimensionality*, yaitu kondisi di mana data berdimensi tinggi justru memperburuk performa model karena meningkatnya kompleksitas, *overfitting*, dan waktu komputasi. Oleh karena itu, dilakukan dimensionality reduction untuk menyederhanakan representasi fitur tanpa kehilangan informasi penting. Proses *dimensionality reduction* dilakukan hanya pada data *feature extraction* dari gambar citra setelah dilakukan *splitting* data latih dan data uji. Pada data latih dan data uji, masing-masing akan dilakukan *dimensionality separation* terlebih dahulu untuk mengambil data *feature image*-nya yang nantinya akan digabungkan lagi setelah proses *dimensionality reduction*. Hal tersebut bertujuan untuk meminimalisasi kemungkinan terjadinya *data leakage*.

Metode *dimensionality reduction* yang digunakan adalah *principal component analysis* (PCA) karena kemampuannya dalam mereduksi dimensi dengan tetap mempertahankan variasi dari data asli. PCA diterapkan dengan dua skema, yaitu mempertahankan 90% dan 85% dari total varian data. Kedua skema ini digunakan untuk menguji pengaruh jumlah fitur terhadap akurasi dan stabilitas model klasifikasi. Objek PCA hasil *training* disimpan dan digunakan kembali untuk mentransformasi data uji, memastikan konsistensi dan mencegah terjadinya *data leakage* selama proses evaluasi.

3.3.7 Synthetic Data Generation

Setelah dimensi fitur direduksi, dilakukan proses *synthetic data generation* (SDG) untuk menambah jumlah data pada kelas minoritas (abnormal). Hal tersebut disebabkan oleh jumlah kelas abnormal yang tidak seimbang dibandingkan dengan jumlah kelas normal. Jumlah kelas abnormal berjumlah 26 persen dari total *dataset*. Ketidakseimbangan ini dapat menyebabkan model bias terhadap kelas mayoritas dan menurunkan kemampuan deteksi kasus abnormal. Untuk mengatasi masalah tersebut, digunakan empat model SDG berbasis GAN, yaitu CTGAN, CTABGAN, CTABGAN+, dan CopulaGAN.

Keempat model tersebut dipilih karena secara khusus dirancang untuk menangani data tabular dan memiliki pengembangan yang berkelanjutan. CTGAN menjadi dasar bagi dua model berikutnya, yaitu CTAB-GAN dan CTAB-GAN+ yang merupakan pengembangannya melalui modifikasi arsitektur dan penambahan fungsi loss. Sementara itu, CopulaGAN digunakan untuk mengeksplorasi pengaruh pendekatan copula dalam memodelkan dependensi antar fitur. Proses sintesis dilakukan pada data *fused* yang telah dilakukan PCA dan belum dilakukan normalisasi. Hal tersebut disebabkan karena model GAN membutuhkan data dalam bentuk asli agar distribusinya bisa ditiru dengan lebih optimal.

3.3.8 Normalisasi

Setelah data sintetik digabungkan dengan data latih yang asli, dilakukan proses normalisasi. Dalam konteks ini, normalisasi mengacu pada proses *data scaling* yang dilakukan pada data. Proses normalisasi fitur pada penelitian ini menggunakan dua skema, yaitu *Standard Scaler* dan *Robust Scaler*. Pemilihan kedua metode ini mempertimbangkan karakteristik umum data medis yang kerap mengandung outlier. *Robust Scaler* dipilih karena menggunakan median dan *interquartile range* dalam proses *scaling* sehingga lebih tahan terhadap pengaruh *outlier* dibandingkan metode seperti *Min-Max Scaling*. Sementara itu, *Standard Scaler* digunakan karena mampu mempertahankan distribusi standar data dengan mengubahnya menjadi distribusi normal standar ($\text{mean} = 0$, standar deviasi = 1).

Normalisasi baru dilakukan setelah proses SDG selesai, karena sebagian besar model generatif seperti GAN membutuhkan data dalam format aslinya. Apabila data sudah dinormalisasi sebelum SDG, model sulit mempelajari distribusi yang representatif dari data asli. Objek *scaler* disimpan setelah *fit* pada data train agar proses transformasi pada *data test* tetap konsisten dan tidak mengandung informasi yang seharusnya tidak diketahui sebelumnya.

3.3.9 Model Building

Model klasifikasi dalam penelitian ini dibangun menggunakan sembilan algoritma *machine learning*, yaitu *Logistic Regression*, *k-nearest neighbors* (KNN), *support vector machine* (SVM), *decision tree*, *random forest*, *AdaBoost*, *eXtreme Gradient Boosting* (XGBoost), *naïve Bayes* (NB), dan *Extra Trees*. Pemilihan algoritma tersebut didasarkan pada popularitasnya dalam berbagai studi skenario klasifikasi. Enam dari sembilan model telah digunakan pada

penelitian terdahulu, sementara tiga model tambahan, XGBoost, NB, dan *Extra Trees*, dilibatkan untuk memperluas cakupan evaluasi antara pendekatan *single learning* dan *ensemble learning*.

Proses pelatihan dilakukan pada data latih hasil penggabungan data asli dan sintetis yang telah melalui PCA dan normalisasi. Setiap model diuji melalui beberapa kombinasi pengujian, yaitu empat *pre-trained model*, empat model GAN, dua skema PCA, dua skema normalisasi, dan sembilan model *machine learning*. Penggunaan beberapa kombinasi pengujian tersebut memungkinkan analisis komparatif untuk menentukan model yang terbaik dalam mendekripsi kelas abnormal dan normal pada data gabungan cairan *effluent dialysate* yang telah dilakukan ekstraksi fitur dan data klinis pasien CAPD.

3.3.10 Model Evaluation

Proses evaluasi model yang dihasilkan akan difokuskan pada nilai *false negative* dari model tersebut. *False negative* pada kasus ini terjadi ketika terdapat pasien yang diprediksi tidak mengalami komplikasi, tetapi secara aktual pasien tersebut seharusnya dikategorikan sebagai pasien yang mengalami komplikasi. Oleh karena itu, nilai *false negative* perlu diperhatikan agar tidak banyak pasien yang mengalami telat perawatan akibat hasil prediksinya menyebutkan pasien tersebut tidak mengalami komplikasi. Selain itu, diperlukan pula keseimbangan nilai *false negative* dan *false positive*. Hal ini dibutuhkan karena pada *dataset* medis umumnya terdapat ketidakseimbangan jumlah kasus positif dan negatif, termasuk *dataset effluent dialysate* dan data klinis pemantauan mandiri pasien CAPD penelitian ini. Dengan kedua fokus tersebut, dipilihlah tiga *metrics evaluation*, yaitu *recall* yang berfokus pada nilai *false negative*, *precision* yang mengukur ketepatan prediksi positif dan *F1-score* yang berfokus pada keseimbangan nilai *false negative* dan *false positive*.

3.4 Implementasi

Subbab ini membahas terkait implementasi dari proses pelatihan data gabungan antara data citra *effluent dialysate* dengan data klinis pasien CAPD.

3.4.1 Implementasi Exploratory Data Analysis

Sebelum dilakukan proses pemodelan, salah satu tahap awal terpenting adalah tahap eksplorasi data atau *exploratory data analysis* (EDA). EDA bertujuan untuk memahami struktur, karakteristik, serta potensi masalah pada *dataset* yang digunakan, seperti keberadaan nilai kosong (*null*), distribusi fitur, dan *outlier*. Dengan EDA, peneliti dapat memperoleh wawasan awal mengenai pola data dan menentukan strategi *preprocessing* yang tepat. Proses EDA dilakukan melalui pengecekan nilai null, analisis statistik deskriptif, visualisasi distribusi fitur menggunakan histplot dengan kernel density estimation (KDE), serta identifikasi outlier menggunakan boxplot, seperti terlihat pada **Kode Semu 3.1**.

```
01 # INPUT: Data Tabular Normal[4137,49] dan Abnormal[891,49]
02 # OUTPUT: Insight Statistik Berupa Tabel atau Grafik
03
04 FUNCTION analyze_medical_data(normal_df, abnormal_df):
05     stats ← Summary(normal_df, abnormal_df)
06     merged ← CONCAT(normal_df, abnormal_df)
07     PLOT_PAIR(merged, normal_df, abnormal_df)
08
09     cols ← ["durasi_masuk", "volume_masuk", "volume_keluar", "berat_badan",
10             "tekanan_darah_mm", "tekanan_darah_hg", "nadi", "volume"]
11
12     FOR dataset IN [normal_df, abnormal_df, merged]:
```

```

13     FOR col IN cols:
14         PLOT(dataset[col], "histogram")
15         PLOT(dataset[col], "boxplot")
16
17     return stats, merged
18
19 BEGIN
20     stats, merged_data ← analyze_medical_data(normal_df[4137,49], abnormal_df[891,49])
21     return stats, merged_data
22 END

```

Kode Semu 3.1 Pseudocode Exploratory Data Analysis

Proses pengecekan nilai *null* dan nilai statistika deskriptif terjadi pada baris kedua di pseudocode tersebut. Lalu, dibuat data gabungan antara normal dan abnormal agar lebih jelas dalam mengetahui perbandingan distribusinya. Pengecekan distribusi terjadi pada baris ke sembilang hingga empat belas. Pengecekan tersebut menggunakan media grafik dengan jenis histogram untuk mengetahui pola distribusi data dan grafik boxplot untuk mengetahui keberadaan *outlier* pada data.

3.4.2 Implementasi *Data Preprocessing*

Setelah proses penggabungan dan eksplorasi awal data klinis, dilakukan tahap data preprocessing untuk menyiapkan data sebelum proses modeling. Tahapan ini mencakup seleksi fitur, penanganan data *null*, *feature engineering*, *encoding* data kategorikal, serta augmentasi sederhana untuk menangani permasalahan input yang terlalu sedikit pada metode CTABGAN+. Seluruh proses tersebut dapat dilihat pada **Kode Semu 3.2**.

```

01 # INPUT: Data Tabular Normal[4137,49] dan Abnormal[891,49]
02 # OUTPUT: Final Tabular Normal[342,15] dan Abnormal[41,15]
03
04 FUNCTION process_data(normal_df, abnormal_df):
05     FOR df IN [normal_df, abnormal_df]:
06         df ← df.drop(['id', 'foto_cairan', 'foto_cairan_url', 'kondisi'])
07         df['volume_masuk'] ← replace(df['volume_masuk'], 1.0 → 1000,0)
08         df['volume_keluar'] ← replace(df['volume_keluar'], 1.0 → 1000,0)
09         df[['waktu_keluar', 'waktu_masuk', 'waktu_penggantian']] ← to_datetime(df[time_cols])
10
11         df['jam_masuk'] ← extract_hour(df['waktu_masuk'])
12         df['hari_masuk'] ← extract_day(df['waktu_masuk'])
13         df['jam_keluar'] ← extract_hour(df['waktu_keluar'])
14         df['hari_keluar'] ← extract_day(df['waktu_keluar'])
15         df['dwell_time_menit'] ← time_diff(df['waktu_keluar'], df['waktu_masuk'])
16         df['selisih_volume'] ← df['volume_keluar'] - df['volume_masuk']
17
18     FOR row IN df:
19         hour ← extract_hour(row['waktu_penggantian'])
20         IF 5 ≤ hour < 11 → row['shift_penggantian'] ← "pagi"
21         ELIF 11 ≤ hour < 15 → row['shift_penggantian'] ← "siang"
22         ELIF 15 ≤ hour < 19 → row['shift_penggantian'] ← "sore"
23         ELSE → row['shift_penggantian'] ← "malam"
24
25     df ← df.drop(['waktu_masuk', 'waktu_keluar', 'waktu_penggantian', 'volume'])
26     df←df.rename({'tekanan_darah_mm': 'systolic', 'tekanan_darah_hg': 'diastolic',
27                   'durasi_masuk': 'durasi_drain_menit'})
28     encoder_gender ← LabelEncoder()
29     encoder_shift ← LabelEncoder()
30     normal_df['jenis_kelamin'] ← encoder_gender.fit_transform(normal_df['jenis_kelamin'])
31     normal_df['shift_penggantian'] ←encoder_shift.fit_transform(normal_df['shift_penggantian'])

```

```

32     abnormal_df['jenis_kelamin'] ← encoder_gender.transform(abnormal_df['jenis_kelamin'])
33     abnormal_df['shift_penggantian'] ← encoder_shift.transform(abnormal_df['shift_penggantian'])
34     return normal_df, abnormal_df
35
36 BEGIN
37     final_normal, final_abnormal ← process_data(normal_df[4137,49], abnormal_df[891,49])
38     return final_normal[342,15], final_abnormal[41,15]
39 END

```

Kode Semu 3.2 Pseudocode Data Preprocessing

Setelah dilakukan penghapusan kolom-kolom yang tidak lagi dibutuhkan, kolom-kolom yang tersisa kemudian dimanipulasi. Proses manipulasi tersebut meliputi pengubahan nilai data yang terindikasi salah input pada kode baris keempat dan kelima. Lalu, seluruh kolom yang berkaitan dengan waktu diubah format nya menjadi *datetime* agar bisa dilakukan ekstraksi. Pada baris kedelapan hingga kedua puluh dilakukan proses *feature engineering*. Lalu pada baris ke-25 hingga 31, dilakukan proses *encoding* menggunakan *label encoder* karena jumlah kategori yang sedikit dan jumlah kolom yang sudah berjumlah lebih dari seratus kolom. Seluruh proses tersebut diperuntukkan baik untuk data normal maupun data abnormal.

3.4.3 Implementasi Image Preprocessing

Citra *effluent dialysate* disimpan kedalam dua folder yang berbeda, folder normal dan abnormal. Pada tahap *image preprocessing*, dilakukan pemfokusan pada citra *effluent dialysate* karena memiliki *background* yang beragam dan objek lain yang tidak relevan. Gambar difokuskan pada area *bag effluent dialysate*-nya saja. Sebelum melakukan hal tersebut, terlebih dahulu dibuat model *bag detector* menggunakan YOLOv9 dengan memanfaatkan roboflow untuk melakukan anotasi gambar. Proses tersebut dapat dilihat pada **Kode Semu 3.3**.

```

01 # INPUT: Citra Effluent Dialysate Normal dan Abnormal
02 # OUTPUT: Cropped Citra Effluent Dialysate Normal dan Abnormal
03
04 FUNCTION setup_and_train():
05     rf ← Roboflow(api_key="###")
06     dataset ← rf.workspace("###").project("effluent-dialysate-annotation")
07         .version(3).download("yolov9")
08     model ← YOLO("yolov9c.pt")
09     results ← model.train(data="data.yaml", epochs=100, imgsz=640)
10     RETURN results
11
12 FUNCTION crop_images(model_path, input_folder, save_folder, conf_threshold):
13     model ← YOLO(model_path)
14     create_directory(save_folder)
15     results ← model.predict(input_folder, conf_threshold)
16
17     FOR result IN results:
18         IF result.bboxes.empty → CONTINUE
19         best_idx ← find_best_detection(result.bboxes.conf, result.bboxes.xyxy)
20         xmin, ymin, xmax, ymax ← result.bboxes.xyxy[best_idx]
21         img ← load_image(result.path)
22         cropped_img ← img[ymin:ymax, xmin:xmax]
23         save_image(cropped_img, save_folder)
24
25 BEGIN
26     setup_and_train()
27     crop_images("best.pt", "normal", "cropped/normal", 0,45)
28     crop_images("best.pt", "abnormal", "cropped/abnormal", 0,45)
29 END

```

Kode Semu 3.3 Pseudocode Image Preprocessing

Pada tujuh baris pertama, *pseudocode* tersebut berfokus pada pembuatan model *bag detector* untuk mendeteksi keberadaan *bag effluent dialysate* pada gambar. Proses pembuatan model *bag detector* tersebut menggunakan *dataset* yang telah dianotasi sebelumnya di Roboflow. Model yang digunakan adalah YOLOv9 karena salah satu fungsinya yang berfokus pada *object detection*. Setelah proses *training* model *bag detector* selesai, model tersebut selanjutnya diimplementasikan pada seluruh *dataset* citra *effluent dialysate*. Di baris selanjutnya, kode berfokus pada *cropping* data gambar agar dapat fokus pada *bag effluent dialysate* nya saja. Gambar-gambar yang telah melewati proses *cropping* tersebut akan dipakai untuk proses berikutnya. Terdapat dua *library* yang perlu di-*install*, yaitu Roboflow dan Ultralytics.

3.4.4 Implementasi Feature Extraction

Sebelum dilakukan penggabungan antara data citra dan data klinis, citra effluent dialysate perlu diekstraksi terlebih dahulu menjadi vektor fitur. Proses tersebut dapat dilihat pada **Kode Semu 3.4**.

```

01 # INPUT: Cropped Citra Effluent Dialysate Normal dan Abnormal
02 # OUTPUT: Representasi Numerik Citra
03
04 FUNCTION extract_features(normal_path, abnormal_path, output_folder):
05     models ← load_models(["swin", "cait", "coatnet", "yolov9"])
06     image_paths, labels ← collect_images(normal_path, abnormal_path)
07
08     features_dict ← {}
09     FOR model_name, model IN models:
10         size ← 640 IF model_name = "yolov9" ELSE 224
11         transform ← compose([resize(size), to_tensor(), normalize()])
12         dataloader ← create_dataloader(image_paths, labels, transform, batch_size=16)
13
14         features ← []
15         all_labels ← []
16         FOR batch_images, batch_labels IN dataloader:
17             output ← model(batch_images)
18             IF output.dim() > 2 → output ← output.flatten()
19             features.append(output)
20             all_labels.append(batch_labels)
21
22         features_dict[model_name] ← {"features": concatenate(features), "labels":
23                                     concatenate(all_labels)}
24     FOR model_name, data IN features_dict:
25         save_csv(output_folder + model_name + "_features.csv", data)
26
27     RETURN features_dict
28
29 BEGIN
30     extracted_features ← extract_features("/normal", "/abnormal", "/features")
31 END

```

Kode Semu 3.4 Pseudocode Feature Extraction

Pada tiga baris pertama, terjadi dua proses, yaitu pemuatan model *feature extractor* dan *path* dari setiap data gambar. Lalu untuk setiap modelnya, gambar-gambar yang telah ditransformasi sesuai dengan ketentuan setiap model disimpan di dalam *dataloader*, suatu komponen PyTorch yang dapat mempermudah pengelolaan data untuk diproses secara acak atau dibagi dalam bentuk *batch* secara otomatis sehingga proses *feature extraction* dapat lebih efisien. Pada baris tiga belas hingga dua puluh dilakukan *feature extraction* pada masing-masing model, baris empat belas. Dimensi output diatur tidak lebih dari dua dimensi agar

masing-masing gambar hanya direpresentasikan oleh satu baris saja dengan beberapa kolom hasil *feature extraction*. Hasil akhir dari proses tersebut adalah *feature numeric* dengan dimensi yang berbeda untuk setiap model *feature extractor*. Untuk model CaiT, dimensi data hasil *feature extraction* nya adalah 465 x 384, CoAtNet sebesar 465 x 37.632, Swin Transformer sebesar 465 x 768, YOLOv9 sebesar 465 x 204.800, Lalu, hasil *feature extraction* tiap model yang berisi gabungan *image* normal dan abnormal disimpan ke dalam file csv.

3.4.5 Implementasi *Early Fusion*

Integrasi data klinis dan citra dilakukan melalui pendekatan *early fusion*. Data yang digabungkan adalah data klinis yang telah melalui proses *data pre-processing* dan hasil ekstraksi fitur citra *effluent dialysate*. Proses tersebut dapat dilihat pada **Kode Semu 3.5**.

```

01 # INPUT: Representasi Numerik Citra dan Final Tabular (dim_normal=342x15, dim_abnormal=41x15)
02 # OUTPUT: Data Fused(dim_cait=465x399, dim_coatnet=465x37.647, dim_swin=465x783,
03                      dim_yolov9=465x204.815)
04 FUNCTION prepare_tabular_data(final_normal, final_abnormal):
05     final_normal["label"] ← 0
06     final_abnormal["label"] ← 1
07     tabular_combined ← concat([final_normal, final_abnormal])
08     tabular_features ← remove_columns(tabular_combined, ["label"])
09     RETURN tabular_features
10
11 FUNCTION fuse_features(image_features, tabular_features):
12     image_only ← remove_columns(image_features, ["label"])
13     fused_dataset ← concat([image_only, tabular_features], axis=1)
14     fused_dataset["label"] ← image_features["label"]
15     RETURN fused_dataset
16
17 FUNCTION feature_fusion(final_normal, final_abnormal, image_features_dict, output_path):
18     tabular_features ← prepare_tabular_data(final_normal, final_abnormal)
19     FOR model_name, image_features IN image_features_dict:
20         fused_dataset ← fuse_features(image_features, tabular_features)
21         save_csv(fused_dataset, output_path + model_name + "_fused.csv")
22 BEGIN
23     image_features ← {"swin": swin_features, "cait": cait_features,
24                       "coatnet": coatnet_features, "yolov9": yolov9_features}
25     feature_fusion(final_normal, final_abnormal, image_features, "/dataset/")
26 END

```

Kode Semu 3.5 Pseudocode Early Fusion

Data *feature extraction* yang sebelumnya telah dibuat sudah diurutkan dengan urutan label normal terlebih dahulu dilanjut label abnormal sesuai dengan urutan pada data tabularnya. Walaupun urutannya sudah sama, data tabular yang ada masih terpisah antara tabular normal dan abnormal. Pada enam baris pertama, data tabular normal dan abnormal digabung terlebih dahulu secara vertikal. Setelah data tabular dan data fitur hasil ekstraksi memiliki format urutan yang sama, kedua data tersebut pada masing-masing model *feature extractor* dilakukan *fusion*. Proses tersebut terjadi pada fungsi *fuse_features* dan *feature_fusion*.

3.4.6 Implementasi *Data Splitting*

Setelah proses *early fusion* menghasilkan empat dataset gabungan dari masing-masing model ekstraksi fitur, dilakukan proses *data splitting* untuk memisahkan data latih dan data uji. Pembagian ini dilakukan di awal untuk meminimalisasi potensi terjadinya *data leakage* dan menjaga objektivitas evaluasi model. Proses tersebut dapat dilihat pada **Kode Semu 3.6**.

```

01 # INPUT: Data Fused
02 # OUTPUT: Data Latih(rows=348) dan Data Uji(rows=117)
03

```

```

04 FUNCTION split_datasets(fused_datasets):
05     split_data ← {}
06
07     FOR model_name, dataframe IN fused_datasets:
08         X ← remove_columns(dataframe, ["label"])
09         y ← dataframe["label"]
10
11         X_train, X_test, y_train, y_test ← train_test_split(X, y, test_size=0.25, stratify=y,
12                                         random_state=42)
13         split_data[model_name] ← {"X_train": X_train, "X_test": X_test, "y_train": y_train,
14                                     "y_test": y_test}
15     RETURN split_data
16
17 BEGIN
18     fused_datasets ← {"swin": fused_swin, "cait": fused_cait, "coatnet": fused_coatnet,
19                       "yolov9": fused_yolov9}
20     split_real_data ← split_datasets(fused_datasets)
21 END

```

Kode Semu 3.6 Pseudocode Data Splitting

Data *fusion* pada masing-masing model dilakukan *data splitting* ke dalam dua bagian, yaitu data uji latih dan data uji, dengan proporsi 75:25. Pada baris kedelapan dapat terlihat proses *data splitting* menggunakan *stratify* untuk memastikan pembagian data tetap menyeimbangkan proporsi kelasnya.

3.4.7 Implementasi Dimensionality Reduction

Proses *dimensionality reduction* dilakukan hanya pada kolom-kolom hasil ekstraksi fitur pada masing-masing *data fused*. Proses tersebut menggunakan metode PCA dengan dua skema, yaitu *variance* 90 persen dan 85 persen. Proses tersebut dapat dilihat pada **Kode Semu 3.7**.

```

01 # INPUT: Data Latih(col_cait=399, col_coatnet=37.647, col_swin=783, col_yolov9=204.815)
02 # OUTPUT: Objek PCA dan Transformed Data Latih(dim_cait=348x82, dim_coatnet=348x252,
03                                         dim_swin=348x96, dim_yolov9=348x220)
04 FUNCTION apply_pca_reduction(split_real_data):
05     pca_results ← {}
06     FOR model_name, data IN split_real_data:
07         X_train ← data['X_train']
08         y_train ← data['y_train']
09         X_train_feat ← X_train[:, :-16]
10         X_train_tabular ← X_train[:, -16:]
11
12         pca ← PCA(n_components=0.85, random_state=42)
13
14         X_train_pca ← pca.fit_transform(X_train_feat)
15         X_train_combined ← concatenate([X_train_pca, X_train_tabular], axis=1)
16
17         pca_results[model_name] ← {'pca': pca, 'X_train': X_train_combined, 'y_train': y_train}
18
19         PRINT model_name + "-Variance: " + format_percentage(sum(pca.explained_variance_ratio))
20     RETURN pca_results
21
22 BEGIN
23     pca_results ← apply_pca_reduction(split_real_data)
24 END

```

Kode Semu 3.7 Pseudocode Dimensionality Reduction

Proses PCA hanya dilakukan pada fitur hasil ekstraksi gambar saja. Hal tersebut terlihat pada baris keenam dan ketujuh yang membagi datanya ke dalam dua bagian, *X_train_feat* sebagai fitur hasil ekstraksi dan *X_train_tabular* sebagai fitur dari data tabular. Pada baris

kesebelas juga terlihat proses PCA hanya dilakukan pada X_train_feat nya saja. Nilai n_components diset bernilai 0.85 pada baris ke sembilan menandakan proses PCA akan mengambil kolom-kolom yang menggambarkan 85% variasi dari keseluruhan datanya. X_train_feat yang telah dilakukan PCA digabungkan kembali dengan X_train_tabular. Setiap objek PCA disimpan untuk digunakan kembali pada *preprocessing* data uji.

3.4.8 Implementasi *Synthetic Data Generation*

Synthetic data generation dilakukan untuk menyeimbangkan distribusi kelas karena jumlah data abnormal hanya 26 persen dari total data. Proses tersebut menggunakan empat model berbasis GAN, yaitu CTGAN, CTABGAN, CTABGAN+, dan CopulaGAN. Proses ini dilakukan pada data latih setelah dilakukan proses PCA dan belum dilakukan proses normalisasi. Proses tersebut dapat dilihat pada **Kode Semu 3.8**.

```

01 # INPUT: Transformed Data Latih(num_rows_normal=256, num_rows_abnormal=92)
02 # OUTPUT: Data Gabungan Sintetik dan Asli(num_rows_normal=256, num_rows_abnormal=256)
03
04 FUNCTION generate_synthetic_data(pca_results, gan_variants, categorical_columns):
05     all_synthetic_data ← {}
06     all_combined_data ← {}
07
08     FOR gan_type IN gan_variants:
09         synthetic_data ← {}
10         combined_data ← {}
11
12         FOR model_name, model_data IN pca_results:
13             X_train ← model_data['X_train']
14             y_train ← model_data['y_train']
15             train_df ← concatenate([X_train, y_train], axis=1)
16
17             minority_data ← filter(train_df, label == 1)
18             majority_size ← count(filter(train_df, label == 0))
19
20             synthesizer ← initialize_gan(gan_type, categorical_columns)
21             synthesizer.fit(minority_data)
22             synthetic_samples ← synthesizer.sample(majority_size)
23
24             save_csv(synthetic_samples, "path.csv")
25
26             combined_df ← concatenate([train_df, align_columns(synthetic_samples,
27                                         train_df.columns)])
28             save_csv(combined_df, "path_combined.csv")
29             synthetic_data[model_name] ← synthetic_samples
30             combined_data[model_name] ← combined_df
31
32             all_synthetic_data[gan_type] ← synthetic_data
33             all_combined_data[gan_type] ← combined_data
34
35     RETURN all_synthetic_data, all_combined_data
36
37 BEGIN
38     gan_variants ← ["CTGAN", "CopulaGAN", "CTAB-GAN"]
39     categorical_columns ← ["jenis_kelamin", "shift_penggantian"]
40     synthetic_data, combined_data←generate_synthetic_data(pca_results,gan_variants,
41                                         categorical_columns)
42 END

```

Kode Semu 3.8 Pseudocode Synthetic Data Generation

Dua perulangan pada *pseudocode* tersebut berarti proses sintetik data dilakukan pada keempat variasi GAN. Pada masing-masing jenis GAN nya, dilakukan pembuatan sintetik data di setiap model *feature extractor*. Pada baris ke-14 hingga 18, dilakukan pelatihan model GAN hanya terhadap data minoritas atau data abnormal. Lalu pada baris ke-19, dibuat sintetik data sebanyak jumlah data minoritas berdasarkan hasil pelatihan model GAN sebelumnya. Proses selanjutnya adalah penggabungan data sintetik dan data asli. Baik data gabungan maupun data sintetik, disimpan ke dalam file csv agar lebih mudah ketika akan dilakukan proses berikutnya.

3.4.9 Implementasi Normalization

Setelah data sintetik digabung dengan data latih asli, dilakukan normalisasi fitur untuk menyamakan skala data. Terdapat dua metode normalisasi yang digunakan secara bergantian, yaitu *robust scaler* dan *standard scaler*. Normalisasi dilakukan pada gabungan data sintetik dan data asli. Proses tersebut dapat dilihat pada **Kode Semu 3.9**.

```

01 # INPUT: Data Gabungan Sintetik dan Asli(num_rows_normal=256, num_rows_abnormal=256)
02 # OUTPUT: Data Gabungan yang Sudah Dinormalisasi(num_rows_normal=256, num_rows_abnormal=256)
03
04 FUNCTION load_and_scale_data(dir_dataset, gan_methods, models, categorical_columns):
05     combined_data ← {}
06     scaled_combined_data ← {}
07     scalers ← {}
08
09     FOR gan IN gan_methods:
10         combined_data[gan] ← {}
11         scaled_combined_data[gan] ← {}
12         scalers[gan] ← {}
13
14     FOR model IN models:
15         path ← dir_dataset + "/" + gan + "/" + model + "_combined.csv"
16         df ← read_csv(path)
17         combined_data[gan][model] ← df
18
19         numerical_columns ← get_columns(df) - categorical_columns - ['label']
20         df_num ← select_columns(df, numerical_columns)
21         df_cat ← select_columns(df, categorical_columns + ['label'])
22
23         scaler ← StandardScaler()
24         df_scaled_values ← scaler.fit_transform(df_num)
25         df_scaled ← create_dataframe(df_scaled_values, columns=numerical_columns,
26                                     index=df.index)
27         df_scaled[categorical_columns + ['label']] ← df_cat
28
29         scaled_combined_data[gan][model] ← df_scaled
30         scalers[gan][model] ← scaler
31
32     RETURN scaled_combined_data, scalers
33
34 BEGIN
35     gan_methods ← ['ctgan', 'ctabgan', 'ctabganplus', 'copulagan']
36     models ← ['swin', 'cait', 'coatnet', 'yolov9']
37     categorical_columns ← ['jenis_kelamin', 'hari_masuk', 'hari_keluar', 'shift_penggantian']
38     scaled_data, scalers ← load_and_scale_data(dir_dataset, gan_methods, models,
39                                              categorical_columns)
40 END

```

Kode Semu 3.9 *Pseudocode Normalization*

Sama seperti proses sintetik data sebelumnya, dua perulangan berarti proses normalisasi dilakukan pada masing-masing model *feature extractor* di setiap model GAN yang ada. Proses normalisasi hanya diimplementasikan pada kolom-kolom yang bertipe numerikal, seperti yang terlihat pada baris ke-16 hingga 23. Selain menyimpan data yang sudah dinormalisasi, setiap objek normalisasinya juga disimpan agar dapat digunakan pada *preprocessing* data uji. Hal tersebut terlihat pada baris ke-26 dan 27.

3.4.10 Implementasi Model Building

Model klasifikasi dibangun menggunakan sembilan algoritma machine learning, yaitu *Logistic Regression*, *KNN*, *SVM*, *decision tree*, *random forest*, *AdaBoost*, *XGBoost*, *naïve Bayes*, dan *Extra Trees*. Pelatihan dilakukan pada data latih hasil early fusion yang telah melalui PCA dan normalisasi. Proses ini memiliki tiga bagian, yaitu *modelling* biasa dengan menggunakan *data train* dan *data test* yang sudah dipisah sebelumnya, lalu bagian kedua adalah proses *cross validation*, bagian terakhir adalah proses *hyperparameter tuning*. Bagian pertama dapat dilihat pada **Kode Semu 3.10**.

```

01 # INPUT: Data Gabungan yang Sudah Dinormalisasi(num_rows_normal=256, num_rows_abnormal=256)
02 # OUTPUT: Model Early Modelling dan Evaluasi Model Early Modelling
03
04 FUNCTION setup_classifiers():
05     classifiers ← {
06         "LogisticRegression": LogisticRegression(
07             C=0.01,penalty='l2',solver='liblinear',max_iter=1000,random_state=42),
08         "KNN": KNeighborsClassifier(n_neighbors=5,weights='uniform',metric='minkowski'),
09         "SVM": SVC(C=0.5,kernel='rbf',probability=True,random_state=42,gamma='scale'),
10         "DecisionTree": DecisionTreeClassifier(max_depth=5,min_samples_split=4,random_state=42),
11         "RandomForest": RandomForestClassifier(
12             n_estimators=100,max_depth=4,min_samples_split=5,random_state=42),
13         "AdaBoost": AdaBoostClassifier(n_estimators=50,learning_rate=1.0,random_state=42),
14         "XGBoost": XGBClassifier(
15             max_depth=3,n_estimators=100,learning_rate=0.01,subsample=0.8,colsample_bytree=0.8,
16             reg_lambda=1,reg_alpha=0.5,eval_metric="logloss",use_label_encoder=False,random_state=42
17         ),
18         "NaiveBayes": GaussianNB(),
19         "ExtraTrees": ExtraTreesClassifier(
20             n_estimators=100,max_depth=None,min_samples_split=2,random_state=42)
21     RETURN classifiers
22
23 FUNCTION evaluate_models(gan_methods, models, scaled_combined_data, split_real_data,
24                           pca_results, scalers, categorical_columns, classifiers):
25     results ← []
26     FOR gan IN gan_methods:
27         FOR model_name IN models:
28             X_train ← scaled_combined_data[gan][model_name].drop(['label']).values
29             y_train ← scaled_combined_data[gan][model_name]['label'].values
30
31             X_test_feat ← split_real_data[model_name]["X_test"].iloc[:, :-16]
32             X_test_tabular ← split_real_data[model_name]["X_test"].iloc[:, -16:]
33             y_test ← split_real_data[model_name]["y_test"]
34
35             pca ← pca_results[model_name]['pca']
36             X_test_pca ← pca.transform(X_test_feat)
37             X_test_combined ← concatenate([X_test_pca, X_test_tabular])
38
39             scaler ← scalers[gan][model_name]
```

```

40         X_test_scaled ← scale_numerical_data(X_test_combined, scaler, categorical_columns)
41
42     FOR clf_name, clf IN classifiers:
43         clf.fit(X_train, y_train)
44
45         y_train_pred ← clf.predict(X_train)
46         y_test_pred ← clf.predict(X_test_scaled)
47
48         results.append({ 'GAN': gan, 'Model': model_name, 'Classifier': clf_name,
49                         'Train_Recall': recall_score(y_train, y_train_pred),
50                         'Train_F1': f1_score(y_train, y_train_pred),
51                         'Test_Recall': recall_score(y_test, y_test_pred),
52                         'Test_F1': f1_score(y_test, y_test_pred)
53                     })
54
55     RETURN create_dataframe(results)
56
57 BEGIN
58     classifiers ← setup_classifiers()
59     results_df ← evaluate_models(gan_methods, models, scaled_combined_data, split_real_data,
60                                   pca_results, scalers, categorical_columns, classifiers)
61 END

```

Kode Semu 3.10 Pseudocode Early Modelling

Delapan baris pertama *pseudocode* tersebut berisi seluruh *classifiers* yang digunakan beserta parameternya yang diinisiasi secara acak. Sebelum melakukan proses evaluasi, data uji perlu disamakan terlebih dahulu polanya seperti data latih. Oleh karena itu, pada baris ke-32 hingga 27 berisi *preprocessing* data uji menggunakan objek PCA dan normalisasi yang digunakan saat melakukan *preprocessing* data latih.

Bagian kedua, *cross validation*, menggunakan *fold* sejumlah lima. Prosesnya menggunakan data latih saja. Adapun prosesnya dapat dilihat pada **Kode Semu 3.11**.

```

01 # INPUT: Data Gabungan yang Sudah Dinormalisasi(num_rows_normal=256, num_rows_abnormal=256)
02 # OUTPUT: Model Cross validation dan Evaluasi Model Cross validation
03
04 FUNCTION perform_cross_validation(gan_methods, models, scaled_combined_data, classifiers):
05     results ← []
06
07     FOR gan IN gan_methods:
08         FOR model_name IN models:
09             X_all ← scaled_combined_data[gan][model_name].drop(['label']).values
10             y_all ← scaled_combined_data[gan][model_name]['label'].values
11
12             FOR clf_name, clf IN classifiers:
13                 skf ← StratifiedKFold(n_splits=5, shuffle=True, random_state=42)
14                 recalls, f1s, precisions ← [], [], []
15
16                 FOR train_idx, test_idx IN skf.split(X_all, y_all):
17                     X_train_cv ← X_all[train_idx]
18                     X_test_cv ← X_all[test_idx]
19                     y_train_cv ← y_all[train_idx]
20                     y_test_cv ← y_all[test_idx]
21
22                     clf.fit(X_train_cv, y_train_cv)
23                     y_pred_cv ← clf.predict(X_test_cv)
24
25                     recalls.append(recall_score(y_test_cv, y_pred_cv))
26                     f1s.append(f1_score(y_test_cv, y_pred_cv))
27                     precisions.append(precision_score(y_test_cv, y_pred_cv))
28
29             results.append({ 'GAN': gan, 'Model': model_name, 'Classifier': clf_name,
30                             'Precision_Mean': round(mean(precisions), 4),

```

```

31             'Recall_Mean': round(mean(recalls), 4),
32             'F1_Mean': round(mean(f1s), 4) })
33
34     RETURN create_dataframe(results)
35 BEGIN
36     cv_results ← perform_cross_validation(gan_methods, models, scaled_combined_data,
37 END

```

Kode Semu 3.11 Pseudocode Cross validation

Perbedaan dengan proses *modelling* sebelumnya adalah penggunaan *StratifiedKFold* untuk membagi data kedalam lima bagian atau *fold*. Proses tersebut terlihat pada baris ke-10, Lalu, satu dari lima bagian tersebut dijadikan data uji. Itu berarti keseluruhan proses *cross validation* hanya menggunakan data latih saja yang dibagi menjadi lima bagian.

Pada bagian *hyperparameter tuning* digunakan metode *GridSearchCV*. Adapun prosesnya dapat dilihat pada **Kode Semu 3.12**.

```

01 # INPUT: Data Gabungan yang Sudah Dinormalisasi(num_rows_normal=256, num_rows_abnormal=256)
02 # OUTPUT: Model Hyperparameter tuning dan Evaluasi Model Hyperparameter tuning
03
04 FUNCTION setup_parameter_grids():
05     param_grids = {
06         "LogisticRegression": {
07             "C": [0.001, 0.01, 0.1, 1, 10], "penalty": ['l1', 'l2'], "solver": ['liblinear']},
08             "KNN": { "n_neighbors": [3, 5, 7], "weights": ['uniform', 'distance'],
09                 "metric": ['euclidean', 'manhattan']},
10             "SVM": {"C": [0.1, 1, 10], "kernel": ['linear', 'rbf'], "gamma": ['scale', 'auto']},
11             "DecisionTree": {"max_depth": [None, 5, 10], "min_samples_split": [2, 5, 10],
12                 "criterion": ['gini', 'entropy']},
13             "RandomForest": {"n_estimators": [100, 200], "max_depth": [None, 5, 10],
14                 "min_samples_split": [2, 5], "criterion": ['gini', 'entropy']},
15             "AdaBoost": {"n_estimators": [50, 100], "learning_rate": [0.01, 0.1, 1.0]},
16             "XGBoost": {"n_estimators": [100, 200], "max_depth": [3, 5], "learning_rate": [0.01, 0.1],
17                 "subsample": [0.8, 1.0], "colsample_bytree": [0.8, 1.0], "reg_lambda": [0.1],
18                 "reg_alpha": [0.0, 0.5]},
19             "NaiveBayes": {"var_smoothing": [1e-9, 1e-8, 1e-7]},
20             "ExtraTrees": {"n_estimators": [100, 200], "max_depth": [None, 5, 10],
21                 "min_samples_split": [2, 5],
22                 "criterion": ['gini', 'entropy']}}
23     RETURN param_grids
24 FUNCTION tune_and_evaluate_models(gan_methods, models, scaled_combined_data, split_real_data,
25                                     pca_results, scalers, categorical_columns, classifiers, param_grids):
26     final_results ← []
27     best_models ← []
28
29     FOR gan IN gan_methods:
30         FOR model_name IN models:
31             X_train ← scaled_combined_data[gan][model_name].drop(['label']).values
32             y_train ← scaled_combined_data[gan][model_name]['label'].values
33             X_test_feat ← split_real_data[model_name]["X_test"].iloc[:, :-16]
34             X_test_tabular ← split_real_data[model_name]["X_test"].iloc[:, -16:]
35             y_test ← split_real_data[model_name]["y_test"]
36             pca ← pca_results[model_name]['pca']
37             X_test_pca ← pca.transform(X_test_feat)
38             X_test_combined ← concatenate([X_test_pca, X_test_tabular])
39             scaler ← scalers[gan][model_name]
40             X_test_scaled ← scale_numerical_data(X_test_combined, scaler, categorical_columns)

```

```

42         FOR clf_name, clf IN classifiers:
43             grid_search ← GridSearchCV(clf, param_grids[clf_name], scoring='f1', cv=5)
44             grid_search.fit(X_train, y_train)
45
46             best_model ← grid_search.best_estimator_
47             y_pred ← best_model.predict(X_test_scaled)
48
49             final_results.append({'GAN': gan, 'Model': model_name, 'Classifier': clf_name,
50                 'Precision_Final': precision_score(y_test, y_pred),
51                 'Recall_Final': recall_score(y_test, y_pred),
52                 'F1_Final': f1_score(y_test, y_pred),
53                 'Best_Params': grid_search.best_params_})
54             best_models.append({'GAN': gan, 'Model': model_name, 'Classifier': clf_name,
55                 'Model': best_model})
56
57     RETURN create_dataframe(final_results), best_models
58 BEGIN
59     param_grids ← setup_parameter_grids()
60     final_results, best_models ← tune_and_evaluate_models(gan_methods, models,
61                                         scaled_combined_data, split_real_data, pca_results, scalers,
62                                         categorical_columns, classifiers, param_grids)
63 END

```

Kode Semu 3.12 Pseudocode Hyperparameter tuning

Proses *hyperparameter tuning* menggunakan *classifier* yang sama seperti bagian *modelling* yang pertama. Hanya saja, perbedaannya adalah penggunaan *list* parameter di setiap *classifier* nya untuk dicari tahu kombinasi parameter terbaik di seiap *classifier* nya. Pada baris ke-46, proses prediksi masing-masing model *classifier* sudah menggunakan kombinasi parameter terbaiknya. Proses pengujian pun sama seperti bagian pertama *modelling*, yaitu menggunakan data uji asli yang telah dilakukan *preprocessing* menggunakan objek PCA dan normalisasi yang disimpan saat *preprocessing* data latih.

3.4.11 Implementasi Model Evaluation

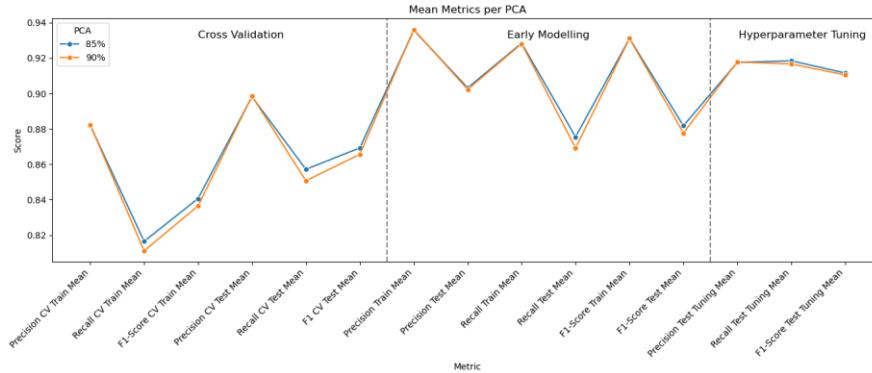
Evaluasi model dilakukan menggunakan tiga metrik, yaitu *recall*, *precision*, dan *F1-score*. Proses evaluasi dilakukan pada ketiga bagian *modelling* dengan membandingkan dengan data uji yang memiliki 86 data normal dan 31 data abnormal. Hasil evaluasi dari ketiga bagian *modelling* tersebut, akan menjadi fokus analisis setiap skenario yang dilakukan pada penelitian ini.

(halaman ini sengaja dikosongkan)

BAB 4 Hasil dan Pembahasan

4.1 Skenario Uji Coba

Pada penelitian ini, digunakan data gabungan dari citra *effluent dialysate* yang telah diekstrak fiturnya dan data klinis tabular. Fitur ekstraksi tersebut kemudian dilakukan *dimensionality reduction* menggunakan PCA dengan dua opsi variansi, yaitu 85 dan 90 persen. Dari kedua opsi tersebut, penggunaan variasi 85 persen relatif lebih baik untuk setiap *metrics*, seperti pada **Gambar 4.1**. Oleh karena itu, PCA dengan variasi 85 persen akan digunakan untuk semua skenario uji coba penelitian ini.



Gambar 4.1 Evaluasi PCA

Selanjutnya, penelitian ini menggunakan tiga jenis skenario umum. Hasil terbaik pada setiap skenarionya akan memengaruhi penggunaan parameter di skenario selanjutnya. Adapun keterangan skenario pada penelitian ini terlihat pada **Tabel 4.1**.

Tabel 4.1 Skenario Uji Coba

Jenis Skenario	Feature Extractor Model	Normalisasi	Jenis Data	Classifiers
Feature Extractor	CaiT	Tidak	Data Asli	Single dan Ensemble Learning
	CoAtNet			
	Swin Transformer			
	YOLOv9			
Normalisasi	CaiT	Tidak	Data Asli	Single dan Ensemble Learning
		Robust Scaler		
		Standard Scaler		
Model Sintetik Data	CaiT	Standard Scaler	Data Asli + Sintetik CTGAN	Single dan Ensemble Learning
			Data Asli + Sintetik CTABGAN	
			Data Asli + Sintetik CTABGAN+	
			Data Asli + Sintetik CopulaGAN	
Classifiers	CaiT	Standard Scaler	Data Asli + Sintetik CTABGAN	Single Learning Ensemble Learning

4.1.1 Uji Coba Feature Extractor

Pada model klasifikasi data gabungan antara citra *effluent dialysate* dengan data klinis, skenario uji coba yang pertama dilakukan adalah penggunaan *pre-trained model* untuk *feature extractor*. Dari keempat *pre-trained model* yang digunakan, akan dipilih *pre-trained model* terbaik untuk dipakai di skenario selanjutnya. Seluruh uji coba *feature extractor* ini hanya menggunakan data asli dan tidak mendapatkan proses normalisasi. Adapun jumlah data pada seluruh uji coba *feature extractor* ini terlihat pada **Tabel 4.2**.

Tabel 4.2 Proporsi Dataset Uji Coba Feature Extractor

Kelas	Jumlah Data		Jumlah
	Train	Test	
Normal	256	86	342
Abnormal	92	31	123
Total	348	117	465

4.1.1.1 Uji Coba Model CaiT

Pada uji coba menggunakan CaiT sebagai model *feature extractor*, dilakukan *cross validation* terlebih dahulu untuk mengetahui konsistensi evaluasi menggunakan data latih yang kemudian di uji menggunakan data uji di setiap *fold* nya. Hasil dari metode *cross validation* dapat terlihat pada **Tabel 4.3**.

Tabel 4.3 Evaluasi Cross validation CaiT

Model	Recall Test Mean	Precision Test Mean	F1-score Test Mean
AdaBoost	0,8839	0,9462	0,9131
Decision Tree	0,8000	0,8703	0,8326
Extra Trees	0,9161	0,9931	0,9529
KNN	0,8839	0,8204	0,8497
Logistic Regression	0,9290	0,9664	0,9473
Naïve Bayes	0,9032	0,9277	0,9152
Random Forest	0,8774	0,9659	0,9184
SVM	0,6968	0,7898	0,7399
XGBoost	0,7871	0,9349	0,8523

Berdasarkan tersebut, model *Extra Trees* menunjukkan performa terbaik secara keseluruhan dengan nilai *precision* tertinggi sebesar 0,9931 dan *F1-score* tertinggi sebesar 0,9529. Sementara itu, *Logistic Regression* unggul dalam evaluasi *recall* dengan nilai sebesar 0,9290, Naïve Bayes, AdaBoost, dan Random Forest juga menunjukkan performa kompetitif dalam hal *F1-score* dengan besaran nilai di atas 0,91. Di sisi lain, SVM menjadi model dengan performa terendah di semua metrik dengan rata-rata nilai *recall* hanya 0,6968, rata-rata nilai *precision* hanya 0,7898, dan rata-rata nilai *F1-score* hanya 0,7399. Secara keseluruhan, rata-rata nilai *recall mean*, *precision mean*, dan *F1-score mean* setiap model pada metode *cross validation* CaiT berturut-turut adalah 0,8638, 0,9517, dan 0,9043.

Selanjutnya dilakukan evaluasi menggunakan data uji dengan menggunakan parameter acak di tiap model *classifier*-nya. Hasil model evaluasi pada data uji tersebut dapat terlihat pada **Tabel 4.4**.

Tabel 4.4 Evaluasi *Modelling* CaiT

Model	Recall	Precision	F1-score
AdaBoost	0,8387	0,9610	0,8966
<i>Decision Tree</i>	0,8387	0,8387	0,8387
<i>Extra Trees</i>	0,9355	1,0000	0,9667
KNN	0,9355	0,8529	0,8923
<i>Logistic Regression</i>	0,9355	0,9667	0,9508
Naïve Bayes	0,9355	0,9355	0,9355
<i>Random Forest</i>	0,9032	0,9655	0,9333
SVM	0,7097	0,8148	0,7586
XGBoost	0,7419	0,9583	0,8364

Berdasarkan **Tabel 4.4**, terdapat empat model yang mencapai nilai *recall* sebesar 0,9355, yaitu Extra Trees, KNN, Logistic Regression, dan Naïve Bayes, yang seluruhnya berada di atas rata-rata *recall* pada saat proses *cross validation*. Untuk *precision*, nilai tertinggi dicapai oleh Extra Trees sebesar 1,0000, diikuti oleh Logistic Regression di posisi kedua dengan nilai sebesar 0,9667, Random Forest di posisi ketiga dengan nilai sebesar 0,9655, dan AdaBoost di posisi keempat dengan nilai sebesar 0,9610, sedangkan *precision* terendah dicatat oleh Decision Tree dengan nilai sebesar 0,8387. *F1-score* tertinggi juga dimiliki oleh Extra Trees sebesar 0,9667, diikuti oleh Logistic Regression di posisi kedua dengan nilai sebesar 0,9508 dan Naïve Bayes di posisi ketiga dengan nilai sebesar 0,9355, sedangkan nilai terendah tercatat pada SVM sebesar 0,7586. Model SVM menjadi satu-satunya model yang berada di bawah nilai rata-rata pada ketiga metrik evaluasi, yaitu *recall* 0,7097, *precision* 0,8148, dan *F1-score* 0,7586, jika dibandingkan dengan nilai rata-rata masing-masing sebesar 0,8638, 0,9217, dan 0,8899. Seluruh model dengan nilai *precision* di atas 0,9500 memiliki nilai *recall* bervariasi antara 0,7419 hingga 0,9355. Hal tersebut menunjukkan bahwa tingginya *precision* tidak selalu diikuti oleh *recall* yang setara. Jika dilihat dari sebaran performa antar metrik, Extra Trees merupakan satu-satunya model yang mencapai nilai maksimal *precision* (1,0000) dengan *recall* dan *F1-score* yang tetap tinggi, sementara model seperti XGBoost memiliki *precision* cukup tinggi (0,9583) namun *recall* rendah (0,7419), yang menyebabkan nilai *F1-score*-nya tidak termasuk tiga teratas. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* setiap model pada evaluasi data uji CaiT berturut-turut adalah 0,8638, 0,9217, dan 0,8899.

Untuk memaksimalkan nilai evaluasi pada *modelling*, dilakukan *hyperparameter tuning* pada pelatihan data latih. Parameter terbaik akan dipakai untuk mengevaluasi data uji. Berdasarkan hasil evaluasi setelah *hyperparameter tuning* pada **Tabel 4.5**, terdapat peningkatan pada beberapa metrik. Untuk *recall*, model yang mengalami peningkatan adalah AdaBoost, KNN, *Logistic Regression*, SVM, dan XGBoost, dengan nilai *recall* tertinggi mencapai 0,9677 pada KNN, *Logistic Regression*, dan SVM. Pada *precision*, hanya ada dua model yang tidak mengalami peningkatan, yaitu AdaBoost dan Naïve Bayes. Sedangkan untuk *F1-score*, terdapat dua model yang tidak mengalami peningkatan, yaitu Naïve Bayes dan *Extra Trees*. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* setiap model pada evaluasi data uji menggunakan parameter yang sudah di-*tuning* pada CaiT berturut-turut adalah 0,9247, 0,9567, dan 0,9398.

Tabel 4.5 Evaluasi Hyperparameter tuning CaiT

Model	Recall	Precision	F1-score	Best-param
AdaBoost	0,9032	0,9333	0,9180	{'learning_rate': 0.1, 'n_estimators': 100}
Decision Tree	0,8387	0,8966	0,8667	{'criterion': 'gini', 'max_depth': None, 'min_samples_split': 4}
Extra Trees	0,9355	1,0000	0,9667	{'criterion': 'entropy', 'max_depth': None, 'min_samples_split': 2, 'n_estimators': 200}
KNN	0,9677	0,9091	0,9375	{'metric': 'manhattan', 'n_neighbors': 3, 'weights': 'distance'}
Logistic Regression	0,9677	0,9677	0,9677	{'C': 0.5, 'penalty': 'l2', 'solver': 'liblinear'}
Naïve Bayes	0,9355	0,9355	0,9355	{'var_smoothing': 1e-09}
Random Forest	0,9032	1,0000	0,9492	{'criterion': 'entropy', 'max_depth': None, 'min_samples_split': 2, 'n_estimators': 200}
SVM	0,9677	0,9677	0,9677	{'C': 0.1, 'gamma': 'scale', 'kernel': 'linear'}
XGBoost	0,9032	1,0000	0,9492	{'colsample_bytree': 1.0, 'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 100, 'reg_alpha': 0.5, 'reg_lambda': 1, 'subsample': 1.0}

Dari ketiga metode pelatihan, *cross validation*, *modelling*, dan *hyperparameter tuning*, didapatkan hasil yang terus meningkat. Hal tersebut diketahui dengan adanya peningkatan di seluruh nilai rata-rata evaluasinya. Secara keseluruhan dari ketiga metode pelatihan tersebut, didapatkan rata-rata *recall*, *precision*, dan *F1-score* terakhir berturut-turut adalah 0,8841, 0,9434, dan 0,9113.

4.1.1.2 Uji Coba Model CoAtNet

Pada uji coba menggunakan CaAtNet sebagai model *feature extractor*, dilakukan *cross validation* terlebih dahulu untuk mengetahui konsistensi evaluasi menggunakan data training. Berdasarkan **Tabel 4.6**, Naïve Bayes memiliki nilai rata-rata *recall* yang terbaik dengan skor 0,9806, unggul hampir 0,06 dibanding AdaBoost di posisi kedua rata-rata *recall* terbaik. Untuk *precision*, *Extra Trees* memiliki nilai rata-rata sempurna. Hal ini menunjukkan model *classifier* tersebut berhasil mendekripsi keabnormalan pada pasien dengan sangat baik. Akan tetapi, Naïve Bayes yang sebelumnya memiliki nilai rata-rata *recall* terbaik tidak berhasil mendapatkan nilai rata-rata *precision* di atas 0,4. Ketimpangan tersebut menyebabkan Naïve Bayes memiliki nilai rata-rata *F1-score* hanya 0,5051. Dalam hal rata-rata nilai *F1-score*, AdaBoost memiliki nilai rata-rata *F1-score* terbaik dengan nilai sebesar 0,9564. Secara keseluruhan, rata-rata nilai *recall* mean, *precision* mean, dan *F1-score* mean setiap model pada metode *cross validation* CoAtNet berturut-turut adalah 0,7928, 0,9140, dan 0,7964.

Tabel 4.6 Evaluasi *Cross Validation* CoAtNet

Model	Recall Test Mean	Precision Test Mean	F1-score Test Mean
AdaBoost	0,9226	0,9933	0,9564
<i>Decision Tree</i>	0,8839	0,9264	0,9044
<i>Extra Trees</i>	0,8645	1,0000	0,9272
KNN	0,8839	0,7967	0,8378
<i>Logistic Regression</i>	0,9097	0,9735	0,9397
Naïve Bayes	0,9806	0,3407	0,5051
<i>Random Forest</i>	0,7935	0,9929	0,8809
SVM	0,6968	0,7957	0,7424
XGBoost	0,8645	0,9729	0,9150

Selanjutnya dilakukan evaluasi menggunakan data uji dengan menggunakan parameter acak di tiap model *classifier*-nya. Hasil model evaluasi pada data uji tersebut dapat terlihat pada **Tabel 4.7**.

Tabel 4.7 Evaluasi *Modelling* CoAtNet

Model	Recall	Precision	F1-score
AdaBoost	0,9033	1,0000	0,9492
<i>Decision Tree</i>	0,9033	0,9333	0,9180
<i>Extra Trees</i>	0,9355	1,0000	0,9667
KNN	0,8710	0,7941	0,8308
<i>Logistic Regression</i>	0,9355	0,9667	0,9508
Naïve Bayes	0,9677	0,3333	0,4959
<i>Random Forest</i>	0,9032	1,0000	0,9492
SVM	0,7097	0,8148	0,7586
XGBoost	0,8710	0,9643	0,9153

Berdasarkan **Tabel 4.7**, model dengan nilai *recall* tertinggi adalah Naïve Bayes dengan nilai sebesar 0,9677. Hal tersebut berbeda dengan hasil *cross validation* sebelumnya yang berada di tiga peringkat terbawah. Namun, meskipun nilai *recall*-nya tinggi, *precision* Naïve Bayes justru sangat rendah yaitu hanya sebesar 0,3333. Hal tersebut menjadikannya model dengan *precision* terendah dan satu-satunya model yang *precision*-nya berada di bawah 0,8000. Selisih *precision* antara *cross validation* dan data uji untuk Naïve Bayes tercatat mengalami penurunan sebesar 60,48%. Hal ini berdampak langsung pada nilai *F1-score*-nya yang hanya sebesar 0,4959, menjadi nilai *F1-score* terendah dan juga mengalami penurunan sebesar 35,96% dibandingkan dengan hasil validasi sebelumnya. Di sisi lain, model Extra Trees kembali menunjukkan performa paling stabil dan tinggi dengan nilai *recall* 0,9355, *precision* 1,0000, dan *F1-score* 0,9667, tanpa mengalami penurunan dari hasil sebelumnya. Logistic Regression dan Random Forest juga mencatat performa yang konsisten tinggi, dengan *precision* di atas 0,9600 dan *F1-score* di atas 0,9490. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* setiap model pada evaluasi data uji CoAtNet berturut-turut adalah 0,8889, 0,8674, dan 0,8594.

Untuk memaksimalkan nilai evaluasi pada *modelling*, dilakukan *hyperparameter tuning* pada pelatihan data latih. Parameter terbaik akan dipakai untuk mengevaluasi data uji. Hasil dari evaluasi tersebut terlihat pada **Tabel 4.8**.

Tabel 4.8 Evaluasi *Hyperparameter tuning* CoAtNet

Model	Recall	Precision	F1-score	Best-param
AdaBoost	0,9355	1,0000	0,9667	<code>{'learning_rate': 0.1, 'n_estimators': 100}</code>
Decision Tree	0,9032	0,9333	0,9180	<code>{'criterion': 'gini', 'max_depth': None, 'min_samples_split': 4}</code>
Extra Trees	0,9355	1,0000	0,9667	<code>{'criterion': 'entropy', 'max_depth': None, 'min_samples_split': 2, 'n_estimators': 200}</code>
KNN	0,9032	0,8750	0,8889	<code>{'metric': 'euclidean', 'n_neighbors': 3, 'weights': 'uniform'}</code>
Logistic Regression	0,9355	1,0000	0,9667	<code>{'C': 10, 'penalty': 'l2', 'solver': 'liblinear'}</code>
Naïve Bayes	0,9677	0,3333	0,4959	<code>{'var_smoothing': 1e-09}</code>
Random Forest	0,9355	1,0000	0,9667	<code>{'criterion': 'gini', 'max_depth': None, 'min_samples_split': 5, 'n_estimators': 200}</code>
SVM	0,9355	0,9667	0,9508	<code>{'C': 0.1, 'gamma': 'scale', 'kernel': 'linear'}</code>
XGBoost	0,9355	1,0000	0,9667	<code>{'colsample_bytree': 1.0, 'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 100, 'reg_alpha': 0.5, 'reg_lambda': 0, 'subsample': 0.8}</code>

Berdasarkan hasil evaluasi setelah *hyperparameter tuning* tersebut, terdapat peningkatan pada beberapa metrik evaluasi. Untuk *recall*, beberapa model yang mengalami peningkatan adalah AdaBoost, KNN, Random Forest, SVM, dan XGBoost, dengan nilai *recall* tertinggi mencapai 0,967742 pada Naïve Bayes. Pada *precision*, hanya terdapat empat model yang mengalami peningkatan, yaitu KNN, Logistic Regression, SVM, dan XGBoost. Terdapat tiga model yang konsisten mendapatkan nilai *precision* tertinggi mencapai 1, yaitu AdaBoost, Extra Trees, dan Logistic Regression. Untuk *F1-score*, hanya tiga model yang tidak mengalami peningkatan, yaitu Decision Tree, Extra Trees, dan Naïve. Sedangkan model lain, seperti AdaBoost, Logistic Regression, dan XGBoost menunjukkan peningkatan dengan nilai tertinggi 0,9667. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* setiap model pada evaluasi data uji menggunakan parameter yang sudah di-tuning berturut-turut adalah 0,9319, 0,9010, dan 0,8986.

Dari ketiga metode pelatihan, *cross validation*, *modelling*, dan *hyperparameter tuning*, didapatkan hasil yang terus meningkat seperti halnya yang terjadi pada CaiT. Hal tersebut diketahui dengan adanya peningkatan di seluruh nilai rata-rata evaluasinya. Secara keseluruhan

dari ketiga metode pelatihan tersebut, didapatkan rata-rata *recall*, *precision*, dan *F1-score* terakhir berturut-turut adalah 0,8712, 0,8941, dan 0,8515.

4.1.1.3 Uji Coba Model Swin Transformer

Pada uji coba menggunakan Swin Transformer sebagai model *feature extractor*, dilakukan *cross validation* terlebih dahulu untuk mengetahui konsistensi evaluasi menggunakan data training. Hasil dari metode *cross validation* dapat terlihat pada **Tabel 4.9**.

Tabel 4.9 Evaluasi *Cross Validation* Swin Transformer

Model	Recall Test Mean	Precision Test Mean	F1-score Test Mean
AdaBoost	0,8710	0,9006	0,8852
Decision Tree	0,8258	0,9433	0,8800
Extra Trees	0,8968	1,0000	0,9454
KNN	0,8516	0,9502	0,8981
Logistic Regression	0,8452	1,0000	0,9159
Naïve Bayes	0,9032	0,9602	0,9300
Random Forest	0,8968	0,9669	0,9294
SVM	0,7935	1,0000	0,8838
XGBoost	0,8452	0,9428	0,8912

Berdasarkan tabel 4.9, Naïve Bayes memiliki nilai rata-rata *recall* yang terbaik dengan skor 0,9032. Tidak seperti model *feature extractor* sebelumnya, nilai rata-rata *recall* pada model Swin Transformer hanya memiliki satu *classifier* yang memiliki nilai di atas 0,9. Untuk *precision*, terdapat tiga model yang memiliki nilai rata-rata *precision* sempurna, yaitu *Extra Trees*, *Logistic Regression*, dan *SVM*. Hal ini menunjukkan bahwa ketiga model *classifier* tersebut berhasil mendekripsi keabnormalan pada pasien dengan sangat baik. Dalam hal rata-rata nilai *F1-score*, *Extra Trees* memiliki nilai rata-rata *F1-score* terbaik dengan nilai sebesar 0,9454. Secara keseluruhan, rata-rata nilai *recall* mean, *precision* mean, dan *F1-score* mean setiap model pada metode *cross validation* Swin Transformer berturut-turut adalah 0,8588, 0,9627, dan 0,9066.

Selanjutnya dilakukan evaluasi menggunakan data uji dengan menggunakan parameter acak di tiap model *classifier*-nya. Hasil model evaluasi pada data uji tersebut dapat terlihat pada **Tabel 4.10**,

Tabel 4.10 Evaluasi *Modelling* Swin Transformer

Model	Recall	Precision	F1-score
AdaBoost	0,8387	0,9286	0,8814
Decision Tree	0,8387	0,9630	0,8966
Extra Trees	0,9355	1,0000	0,9667
KNN	0,9355	0,8529	0,8923
Logistic Regression	0,9032	0,9333	0,9180
Naïve Bayes	0,9032	0,9655	0,9333
Random Forest	0,9032	0,9655	0,9333
SVM	0,7097	0,8148	0,7586
XGBoost	0,8710	0,9310	0,9000

Berdasarkan hasil yang tertera pada Tabel 4.10, Hanya AdaBoost dan SVM yang mengalami penurunan dibanding saat pelatihan menggunakan *cross validation*. Untuk *precision*, seluruh model mengalami peningkatan dan *Extra Trees* memiliki nilai tertinggi. AdaBoost menjadi satu-satunya model yang mengalami penurunan nilai *F1-score*. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada evaluasi data uji dengan Swin Transformer berturut-turut adalah 0,8710, 0,9283, dan 0,8978.

Untuk memaksimalkan nilai evaluasi pada *modelling*, dilakukan *hyperparameter tuning* pada pelatihan data latih. Parameter terbaik akan dipakai untuk mengevaluasi data uji. Hasil dari evaluasi tersebut terlihat pada **Tabel 4.11**.

Tabel 4.11 Evaluasi *Hyperparameter tuning* Swin Transformer

Model	Recall	Precision	F1-score	Best-param
AdaBoost	0,8710	0,9310	0,9000	{'learning_rate': 1.0, 'n_estimators': 100}
Decision Tree	0,8065	0,8929	0,8475	{'criterion': 'gini', 'max_depth': None, 'min_samples_split': 2}
Extra Trees	0,9032	0,9655	0,9333	{'criterion': 'gini', 'max_depth': 10, 'min_samples_split': 5, 'n_estimators': 100}
KNN	0,9355	0,9062	0,9206	{'metric': 'manhattan', 'n_neighbors': 3, 'weights': 'distance'}
Logistic Regression	0,9355	0,8788	0,9063	{'C': 0.5, 'penalty': 'l1', 'solver': 'liblinear'}
Naïve Bayes	0,9032	0,9655	0,9333	{'var_smoothing': 1e-09}
Random Forest	0,9355	0,9063	0,9206	{'criterion': 'entropy', 'max_depth': None, 'min_samples_split': 5, 'n_estimators': 100}
SVM	0,9355	0,8788	0,9063	{'C': 0.1, 'gamma': 'scale', 'kernel': 'linear'}
XGBoost	0,9355	0,9355	0,9355	{'colsample_bytree': 0.8, 'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 200, 'reg_alpha': 0.5, 'reg_lambda': 1, 'subsample': 0.8}

Berdasarkan hasil evaluasi setelah *hyperparameter tuning*, terdapat peningkatan pada beberapa metrik evaluasi. Untuk *recall*, model yang mengalami peningkatan adalah AdaBoost, *Logistic Regression*, *Random Forest*, SVM, dan XGBoost. Nilai *recall* tertinggi adalah 0,9355. Pada *precision*, hanya tempat model yang mengalami peningkatan, yaitu AdaBoost, KNN, SVM, dan XGBoost. Hanya keempat model tersebut juga yang mengalami pengikatan nilai *F1-score*. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* setiap model pada evaluasi data uji menggunakan parameter yang sudah di-tuning berturut-turut adalah 0,9068, 0,9178, dan 0,9115.

Dari ketiga metode pelatihan, *cross validation*, *modelling*, dan *hyperparameter tuning*, terdapat satu metrik evaluasi yang tidak terus meningkat. Pada metrik *precision*, terjadi penurunan nilai setelah parameter dilakukan *tuning*. Hasil ini berbeda dengan kedua *model feature extractor* sebelumnya yang terus mengalami peningkatan di setiap metrik evaluasinya. Secara keseluruhan dari ketiga metode pelatihan tersebut, didapatkan rata-rata *recall*, *precision*, dan *F1-score* terakhir berturut-turut adalah 0,8789, 0,9363, dan 0,9053.

4.1.1.4 Uji Coba Model YOLOv9

Pada uji coba menggunakan YOLOv9 sebagai model *feature extractor*, dilakukan *cross validation* terlebih dahulu untuk mengetahui konsistensi evaluasi menggunakan data training. Hasil dari metode *cross validation* dapat terlihat pada **Tabel 4.12**.

Tabel 4.12 Evaluasi *Cross validation* YOLOv9

Model	Recall Test Mean	Precision Test Mean	F1-score Test Mean
AdaBoost	0,8452	0,9519	0,8949
Decision Tree	0,8710	0,9062	0,8861
Extra Trees	0,8065	1,0000	0,8918
KNN	0,9097	0,8262	0,8647
Logistic Regression	0,9097	0,9530	0,9304
Naïve Bayes	0,9548	0,6880	0,7940
Random Forest	0,8065	1,0000	0,8904
SVM	0,6968	0,7957	0,7424
XGBoost	0,8645	0,9586	0,9088

Berdasarkan tabel 4.12, Naïve Bayes memiliki nilai rata-rata *recall* yang terbaik dengan skor 0,9548. Untuk *precision*, terdapat dua model yang memiliki nilai rata-rata *precision* sempurna, yaitu *Extra Trees* dan *Random Forest*. Hal ini menunjukkan bahwa kedua model *classifier* tersebut berhasil mendekripsi keabnormalan pada pasien dengan sangat baik. Dalam hal rata-rata nilai *F1-score*, *Logistic Regression* memiliki nilai rata-rata *F1-score* terbaik dengan nilai sebesar 0,9304. Secara keseluruhan, rata-rata nilai *recall mean*, *precision mean*, dan *F1-score mean* setiap model pada metode *cross validation* YOLOv9 berturut-turut adalah 0,8423, 0,8697, dan 0,8247.

Selanjutnya dilakukan evaluasi menggunakan data uji dengan menggunakan parameter acak di tiap model *classifier*-nya. Hasil model evaluasi pada data uji tersebut dapat terlihat pada Tabel 4.13.

Tabel 4.13 Hasil Evaluasi *Modelling* YOLOv9

Model	Recall	Precision	F1-score
AdaBoost	0,8710	0,9000	0,8853
Decision Tree	0,9032	0,9333	0,9180
Extra Trees	0,8710	1.0000	0,9310
KNN	0,9355	0,8529	0,8923
Logistic Regression	0,9677	0,9677	0,9677
Naïve Bayes	0,9677	0,6250	0,7595
Random Forest	0,8065	1.0000	0,8929
SVM	0,7097	0,8148	0,7586
XGBoost	0,8710	0,9643	0,9153

Berdasarkan Tabel 4.13, hanya AdaBoost dan SVM yang mengalami penurunan. Dibanding saat pelatihan menggunakan *cross validation*. Dalam hal *precision*, terdapat dua model yang mengalami penurunan dibanding saat *cross validation*, yaitu AdaBoost dan Naïve Bayes. Model Naïve Bayes menjadi satu-satunya model yang mengalami penurunan nilai *F1-score*. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* setiap model pada evaluasi data uji YOLOv9 berturut-turut adalah 0,8781, 0,8954, dan 0,8801.

Untuk memaksimalkan nilai evaluasi pada *modelling*, dilakukan *hyperparameter tuning* pada pelatihan data latih. Parameter terbaik akan dipakai untuk mengevaluasi data uji. Hasil dari evaluasi tersebut terlihat pada **Tabel 4.14**.

Tabel 4.14 Hasil Evaluasi *Hyperparameter tuning* YOLOv9

Model	Recall	Precision	F1-score	Best-param
AdaBoost	0,8710	0,9000	0,8853	{'learning_rate': 1.0, 'n_estimators': 50}
Decision Tree	0,8710	0,9310	0,9000	{'criterion': 'gini', 'max_depth': None, 'min_samples_split': 2}
Extra Trees	0,9032	1,0000	0,9492	{'criterion': 'entropy', 'max_depth': None, 'min_samples_split': 2, 'n_estimators': 100}
KNN	0,9677	0,8824	0,9231	{'metric': 'euclidean', 'n_neighbors': 3, 'weights': 'distance'}
Logistic Regression	0,9355	0,8529	0,8923	{'C': 0.1, 'penalty': 'l1', 'solver': 'liblinear'}
Naïve Bayes	0,9677	0,6383	0,7692	{'var_smoothing': 1e-08}
Random Forest	0,8065	1,0000	0,8929	{'criterion': 'gini', 'max_depth': 4, 'min_samples_split': 5, 'n_estimators': 100}
SVM	0,9677	0,8824	0,9231	{'C': 0.1, 'gamma': 'scale', 'kernel': 'linear'}
XGBoost	0,9032	0,9655	0,9333	{'colsample_bytree': 0.8, 'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 100, 'reg_alpha': 0.5, 'reg_lambda': 1, 'subsample': 1.0}

Berdasarkan hasil evaluasi setelah *hyperparameter tuning*, terdapat peningkatan pada beberapa metrik evaluasi. Untuk *recall*, model yang mengalami peningkatan adalah *Extra Trees*, KNN, SVM, dan XGBoost. Pada *precision*, hanya empat model yang mengalami peningkatan, yaitu KNN, Naïve Bayes, SVM, dan XGBoost. Untuk *F1-score*, terdapat beberapa model yang mengalami peningkatan, *Extra Trees*, KNN, Naïve Bayes, SVM, dan XGBoost. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* setiap model pada evaluasi data uji menggunakan parameter yang sudah di-tuning berturut-turut adalah 0,9104, 0,8947, dan 0,8965.

Dari ketiga metode pelatihan, *cross validation*, *modelling*, dan *hyperparameter tuning*, terdapat satu metrik evaluasi yang tidak terus meningkat seperti halnya yang terjadi pada Swin Transformer. Pada metrik *precision*, terjadi penurunan nilai setelah parameter dilakukan *tuning*. Secara keseluruhan dari ketiga metode pelatihan tersebut, didapatkan rata-rata *recall*, *precision*,

dan *F1-score* terakhir berturut-turut adalah 0,8769, 0,8866, dan 0,8671. Dengan nilai tersebut, YOLOv9 menempati posisi terakhir dengan total rata-rata evaluasi terkecil di antara ketiga model *feature extractor* lainnya. Sedangkan untuk posisi pertama ditempati oleh CaiT. Lalu, Model *feature extractor* terbaik akan digunakan pada uji coba berikutnya.

4.1.2 Uji Coba Normalisasi

Uji coba kedua pada model klasifikasi data gabungan antara citra *effluent dialysate* dengan data klinis adalah perlakuan normalisasi pada data. Terdapat tiga skema dalam uji coba ini, yaitu penggunaan *standard scaler*, *robust scaler*, dan tanpa normalisasi. Dari ketiga skema tersebut, akan dipilih skema normalisasi terbaik untuk dipakai di skenario selanjutnya. Seluruh uji coba normalisasi ini menggunakan *pre-trained model* terbaik yang didapatkan pada uji coba sebelumnya, yaitu CaiT, dan hanya menggunakan data asli. Adapun jumlah data pada seluruh uji coba *feature extractor* ini terlihat pada **Tabel 4.15**.

Tabel 4.15 Proporsi Dataset Uji Coba Normalisasi

Kelas	Jumlah Data		Jumlah
	Train	Test	
Normal	256	86	342
Abnormal	92	31	123
Total	348	117	465

4.1.2.1 Uji Coba Tanpa Normalisasi

Proses pelatihan juga menggunakan metode *cross validation* pada data training dengan membagi data latih menjadi lima *fold*. Adapun hasil dari metode *cross validation* tanpa menggunakan metode normalisasi terlihat pada **Tabel 4.16**.

Tabel 4.16 Hasil Evaluasi *Cross validation* Tanpa Normalisasi

Model	Recall Test Mean	Precision Test Mean	F1-score Test Mean
AdaBoost	0,8839	0,9462	0,9131
Decision Tree	0,8000	0,8703	0,8326
Extra Trees	0,9161	0,9931	0,9529
KNN	0,8839	0,8204	0,8497
Logistic Regression	0,9290	0,9664	0,9473
Naïve Bayes	0,9032	0,9277	0,9152
Random Forest	0,8774	0,9659	0,9184
SVM	0,6968	0,7898	0,7399
XGBoost	0,7871	0,9349	0,8523

Berdasarkan tabel 4.16, *Logistic Regression* memiliki nilai rata-rata *recall* yang terbaik dengan skor 0,9290. Untuk *precision*, *Extra Trees* memiliki nilai rata-rata tertinggi dengan nilai sebesar 0,9931. *Extra Trees* juga unggul pada matriks *F1-score* dengan nilai sebesar 0,9529. Di sisi lain, model SVM selalu memiliki nilai terendah dengan tidak satupun metriks evaluasi yang mencapai 0,8. Nilai-nilai metriks evaluasi tersebut adalah 0,6968 untuk rata-rata *recall*, 0,7898 untuk rata-rata *precision*, dan 0,7399 untuk rata-rata *F1-score*. Secara keseluruhan, rata-rata nilai *recall mean*, *precision mean*, dan *F1-score mean* setiap model pada metode *cross validation* tanpa normalisasi berturut-turut adalah 0,8530, 0,9127, dan 0,8802.

Selanjutnya dilakukan evaluasi menggunakan data uji dengan menggunakan parameter acak di tiap model *classifier*-nya. Hasil model evaluasi pada data uji tersebut dapat terlihat pada Tabel 4.17.

Tabel 4.17 Hasil Evaluasi *Modelling* Tanpa Normalisasi

Model	Recall	Precision	F1-score
AdaBoost	0,8387	0,9630	0,8966
<i>Decision Tree</i>	0,8387	0,8387	0,8387
<i>Extra Trees</i>	0,9355	1,0000	0,9667
KNN	0,9355	0,8529	0,8923
<i>Logistic Regression</i>	0,9355	0,9667	0,9508
Naïve Bayes	0,9355	0,9355	0,9355
<i>Random Forest</i>	0,9032	0,9655	0,9333
SVM	0,7097	0,8148	0,7586
XGBoost	0,7419	0,9583	0,8364

Pada evaluasi menggunakan data uji, seluruh model dilatih terlebih dahulu menggunakan keseluruhan data latih tanpa proses pembagian ke dalam *fold* seperti pada *cross validation*. Berdasarkan hasil evaluasi terhadap metrik *recall*, terdapat lima dari sembilan model yang menunjukkan peningkatan dibandingkan hasil *cross validation* sebelumnya, yaitu Extra Trees, KNN, Logistic Regression, Naïve Bayes, dan Random Forest, di mana kelima model tersebut memiliki nilai *recall* sebesar 0,9355, kecuali Random Forest yang mencatat 0,9032. Untuk *precision*, hampir seluruh model mengalami peningkatan kecuali Decision Tree yang stagnan di angka 0,8387. Dari sisi *F1-score*, yang merupakan metrik gabungan dari *recall* dan *precision*, terdapat tiga model yang tidak mengalami kenaikan dibandingkan dengan hasil *cross validation*, yakni AdaBoost, Decision Tree, dan XGBoost dengan nilai secara urut adalah 0,8966, 0,8387, dan 0,8364. Nilai *F1-score* tertinggi dicapai oleh Extra Trees sebesar 0,9667, diikuti oleh Logistic Regression di posisi kedua dengan nilai sebesar 0,9508 dan Naïve Bayes di posisi ketiga dengan nilai sebesar 0,9355. Model dengan performa terendah di semua metrik tetap konsisten dipegang oleh SVM, dengan *recall* 0,7097, *precision* 0,8148, dan *F1-score* 0,7586. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada evaluasi data uji dengan tanpa proses normalisasi berturut-turut adalah 0,8638, 0,9217, dan 0,8899.

Untuk memaksimalkan nilai evaluasi pada *modelling*, dilakukan *hyperparameter tuning* pada pelatihan data latih. Parameter terbaik akan dipakai untuk mengevaluasi data uji. Hasil dari evaluasi tersebut terlihat pada Tabel 4.18. Setelah dilakukan *hyperparameter tuning*, terjadi peningkatan nilai *recall* pada lima *model classifier*, yaitu *Decision Tree*, *Extra Trees*, Naïve Bayes, dan *Random Forest*. Sedikit berbeda dengan *recall*, hanya tiga *model classifier* yang tidak mengalami peningkatan evaluasi *precision*. Ketiga model tersebut adalah AdaBoost, *Extra Trees*, dan Naïve Bayes. Untuk evaluasi menggunakan *F1-score*, terdapat dua *model classifier* yang tidak mengalami peningkatan, yaitu *Extra Trees* dan Naïve Bayes. Dari ketiga metrik evaluasi, hanya satu *model classifier* yang memiliki nilai di bawah 0,9000. Secara keseluruhan, rata-rata *recall*, *precision*, dan *F1-score* pada evaluasi data uji setelah *hyperparameter tuning* berturut-turut adalah 0,9247, 0,9567, dan 0,9398.

Tabel 4.18 Hasil Evaluasi *Hyperparameter tuning* Tanpa Normalisasi

Model	Recall	Precision	F1-score	Best-param
AdaBoost	0,9032	0,9333	0,9180	{'learning_rate': 0.1, 'n_estimators': 100}
Decision Tree	0,8387	0,8966	0,8667	{'criterion': 'gini', 'max_depth': None, 'min_samples_split': 4}
Extra Trees	0,9355	1,0000	0,9667	{'criterion': 'entropy', 'max_depth': None, 'min_samples_split': 2, 'n_estimators': 200}
KNN	0,9677	0,9091	0,9375	{'metric': 'manhattan', 'n_neighbors': 3, 'weights': 'distance'}
Logistic Regression	0,9677	0,9677	0,9677	{'C': 0.5, 'penalty': 'l2', 'solver': 'liblinear'}
Naïve Bayes	0,9355	0,9355	0,9355	{'var_smoothing': 1e-09}
Random Forest	0,9032	1,0000	0,9492	{'criterion': 'entropy', 'max_depth': None, 'min_samples_split': 2, 'n_estimators': 200}
SVM	0,9677	0,9677	0,9677	{'C': 0.1, 'gamma': 'scale', 'kernel': 'linear'}
XGBoost	0,9032	1,0000	0,9492	{'colsample_bytree': 1.0, 'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 100, 'reg_alpha': 0.5, 'reg_lambda': 1, 'subsample': 1.0}

Dari ketiga metode pelatihan, *cross validation*, *modelling*, dan *hyperparameter tuning*, didapatkan hasil yang terus meningkat. Hal tersebut diketahui dengan adanya peningkatan di seluruh nilai rata-rata evaluasinya. Secara keseluruhan dari ketiga metode pelatihan tersebut, didapatkan rata-rata *recall*, *precision*, dan *F1-score* terakhir berturut-turut adalah 0,8805, 0,9304, dan 0,9033.

4.1.2.2 Uji Coba *Robust Scaler*

Proses pelatihan menggunakan metode *cross validation*. Hasil dari metode *cross validation* menggunakan *Robust Scaler* sebagai metode normalisasinya terlihat pada **Tabel 4.19**.

Tabel 4.19 Hasil Evaluasi *Cross Validation Robust Scaler*

Model	Recall Test Mean	Precision Test Mean	F1-score Test Mean
AdaBoost	0,8839	0,9462	0,9131
Decision Tree	0,8000	0,8636	0,8297
Extra Trees	0,9161	0,9931	0,9529
KNN	0,8323	0,8608	0,8458
Logistic Regression	0,5677	1,0000	0,7221
Naïve Bayes	0,9032	0,9277	0,9152
Random Forest	0,8774	0,9659	0,9184
SVM	0,7613	1,0000	0,8628
XGBoost	0,7871	0,9349	0,8523

Berdasarkan tabel 4.19, *Extra Trees* memiliki nilai rata-rata *recall* yang terbaik dengan skor 0,9161. Sementara itu, *Logistic Regression* memiliki nilai rata-rata terendah, hanya 0,5677. Dalam hal *precision*, *Logistic Regression* memiliki nilai sempurna bersama dengan model SVM. Dari sembilan model *classifier*, hanya dua model yang tidak mencapai nilai 0,9. *Extra Trees* kembali unggul pada matriks *F1-score* dengan nilai sebesar 0,9529. Secara keseluruhan, rata-rata nilai *recall mean*, *precision mean*, dan *F1-score mean* setiap model pada metode *cross validation* Robust Scaler berturut-turut adalah 0,8143, 0,9436, dan 0,8680,

Selanjutnya dilakukan evaluasi menggunakan data uji dengan menggunakan parameter acak di tiap model *classifier*-nya. Hasil model evaluasi pada data uji tersebut dapat terlihat pada Tabel 4.20,

Tabel 4.20 Hasil Evaluasi *Modelling Robust Scaler*

Model	Recall	Precision	F1-score
AdaBoost	0,8387	0,9630	0,8966
<i>Decision Tree</i>	0,8387	0,8387	0,8387
<i>Extra Trees</i>	0,9355	1,0000	0,9667
KNN	0,8710	0,9000	0,8853
<i>Logistic Regression</i>	0,7097	1,0000	0,8302
Naïve Bayes	0,9355	0,9355	0,9355
<i>Random Forest</i>	0,9032	0,9655	0,9333
SVM	0,8387	1,0000	0,9123
XGBoost	0,7420	0,9583	0,8364

Pada evaluasi menggunakan data uji, terdapat tiga *model classifier* yang tidak mengalami kenaikan nilai *recall*, yaitu AdaBoost, *Decision Tree*, dan KNN. Dalam hal *precision*, hanya *Decision Tree* yang tidak mengalami kenaikan nilai. Terdapat empat *model classifier* yang tidak mengalami peningkatan nilai *F1-score*, yaitu AdaBoost, *Decision Tree*, KNN, dan XGBoost. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada evaluasi data uji menggunakan Robust Scaler berturut-turut adalah 0,8459, 0,9512, dan 0,8928.

Untuk memaksimalkan nilai evaluasi pada *modelling*, dilakukan *hyperparameter tuning* pada pelatihan data latih. Parameter terbaik akan dipakai untuk mengevaluasi data uji. Hasil dari evaluasi tersebut terlihat pada **Tabel 4.21**.

Tabel 4.21 Hasil Evaluasi *Hyperparameter tuning Robust Scaler*

Model	Recall	Precision	F1-score	Best-param
AdaBoost	0,9032	0,9333	0,9180	{'learning_rate': 0.1, 'n_estimators': 100}
<i>Decision Tree</i>	0,8387	0,8966	0,8667	{'criterion': 'gini', 'max_depth': None, 'min_samples_split': 4}
<i>Extra Trees</i>	0,9355	1,0000	0,9667	{'criterion': 'entropy', 'max_depth': None, 'min_samples_split': 2, 'n_estimators': 200}
KNN	0,9355	0,9355	0,9355	{'metric': 'manhattan', 'n_neighbors': 7, 'weights': 'uniform'}

Model	Recall	Precision	F1-score	Best-param
<i>Logistic Regression</i>	0,9355	0,9667	0,9508	{'C': 10, 'penalty': 'l1', 'solver': 'liblinear'}
Naïve Bayes	0,9355	0,9355	0,9355	{'var_smoothing': 1e-09}
<i>Random Forest</i>	0,9032	1,0000	0,9492	{'criterion': 'entropy', 'max_depth': None, 'min_samples_split': 2, 'n_estimators': 200}
SVM	0,9355	1,0000	0,9667	{'C': 5, 'gamma': 'scale', 'kernel': 'rbf'}
XGBoost	0,9032	1,0000	0,9492	{'colsample_bytree': 1.0, 'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 100, 'reg_alpha': 0.5, 'reg_lambda': 1, 'subsample': 1.0}

Setelah dilakukan *hyperparameter tuning*, terjadi peningkatan pada nilai *recall*, *precision*, dan *F1-score* di beberapa model. Empat dari sembilan model tidak mengalami peningkatan nilai *recall*. Lima dari sembilan model tidak mengalami pengikatan nilai *precision*. Hanya dua dari sembilan model yang tidak mengalami peningkatan nilai *F1-score*. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada evaluasi data uji setelah *hyperparameter tuning* berturut-turut adalah 0,9140, 0,9631, dan 0,9376. Nilai rata-rata tersebut menunjukkan peningkatan daripada saat tidak dilakukan *hyperparameter tuning*.

Dari ketiga metode pelatihan, *cross validation*, *modelling*, dan *hyperparameter tuning*, didapatkan hasil yang terus meningkat. Hal tersebut diketahui dengan adanya peningkatan di seluruh nilai rata-rata evaluasinya. Secara keseluruhan dari ketiga metode pelatihan tersebut, didapatkan rata-rata *recall*, *precision*, dan *F1-score* terakhir berturut-turut adalah 0,8581, 0,9526, dan 0,8995.

4.1.2.3 Uji Coba Standard Scaler

Proses pelatihan juga menggunakan metode *cross validation* pada data training dengan membagi data latih menjadi lima *fold*. Adapun hasil dari metode *cross validation* menggunakan *Standard Scaler* sebagai metode normalisasinya terlihat pada **Tabel 4.22**.

Tabel 4.22 Hasil Evaluasi *Cross Validation Standard Scaler*

Model	Recall Test Mean	Precision Test Mean	F1-score Test Mean
AdaBoost	0,8839	0,9462	0,9131
<i>Decision Tree</i>	0,8000	0,8703	0,8326
<i>Extra Trees</i>	0,9161	0,9931	0,9529
KNN	0,8903	0,9268	0,9080
<i>Logistic Regression</i>	0,8516	1,0000	0,9194
Naïve Bayes	0,9032	0,9277	0,9152
<i>Random Forest</i>	0,8839	0,9662	0,9221
SVM	0,8581	1,0000	0,9230
XGBoost	0,7871	0,9349	0,8523

Berdasarkan tabel 4.22, *Extra Trees* memiliki nilai rata-rata *recall* yang terbaik dengan skor 0,9161. Sementara itu, XGBoost memiliki nilai rata-rata terendah, hanya 0,7871. Dalam hal *precision*, *Logistic Regression* dan SVM memiliki nilai sempurna. Dari sembilan model *classifier*, hanya satu model yang tidak mencapai nilai 0,9. Hal tersebut menunjukkan hampir model dapat meminimalisasi nilai *false positive* nya. *Extra Trees* kembali unggul pada matriks *F1-score* dengan nilai sebesar 0,9529. Secara keseluruhan, rata-rata nilai *recall* mean, *precision* mean, dan *F1-score* mean setiap model pada metode *cross validation* Robust Scaler berturut-turut adalah 0,8638, 0,9517, dan 0,9043.

Selanjutnya dilakukan evaluasi menggunakan data uji dengan menggunakan parameter acak di tiap model *classifier*-nya. Hasil model evaluasi pada data uji tersebut dapat terlihat pada Tabel 4.23.

Tabel 4.23 Hasil Evaluasi *Modelling Standard Scaler*

Model	Recall	Precision	F1-score
AdaBoost	0,8387	0,9630	0,8966
<i>Decision Tree</i>	0,8387	0,8387	0,8387
<i>Extra Trees</i>	0,9355	1,0000	0,9667
KNN	0,9032	0,9655	0,9333
<i>Logistic Regression</i>	0,8710	1,0000	0,9310
Naïve Bayes	0,9355	0,9355	0,9355
<i>Random Forest</i>	0,9032	0,9655	0,9333
SVM	0,9032	1,0000	0,9492
XGBoost	0,7420	0,9583	0,8364

Pada evaluasi menggunakan data uji, terdapat tiga *model classifier* yang tidak mengalami kenaikan nilai *recall*, yaitu AdaBoost, *Decision Tree*, dan XGBoost. Dalam hal *precision*, hanya *Decision Tree* yang tidak mengalami kenaikan nilai. Tiga model yang tidak mengalami peningkatan nilai *recall* kembali tidak mengalami kenaikan nilai pada evaluasi *F1-score*. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada evaluasi data uji menggunakan Robust Scaler berturut-turut adalah 0,8746, 0,9585, dan 0,9134.

Untuk memaksimalkan nilai evaluasi pada *modelling*, dilakukan *hyperparameter tuning* pada pelatihan data latih. Parameter terbaik akan dipakai untuk mengevaluasi data uji. Hasil dari evaluasi tersebut terlihat pada **Tabel 4.24**.

Tabel 4.24 Hasil Evaluasi *Hyperparameter Tuning Standard Scaler*

Model	Recall	Precision	F1-score	Best-param
AdaBoost	0,9032	0,9333	0,9180	{'learning_rate': 0,1, 'n_estimators': 100}
<i>Decision Tree</i>	0,8387	0,8966	0,8667	{'criterion': 'gini', 'max_depth': None, 'min_samples_split': 4}
<i>Extra Trees</i>	0,9355	1,0000	0,9667	{'criterion': 'entropy', 'max_depth': None, 'min_samples_split': 2, 'n_estimators': 200}
KNN	0,9355	0,9355	0,9355	{'metric': 'manhattan', 'n_neighbors': 7, 'weights': 'distance'}

Model	Recall	Precision	F1-score	Best-param
<i>Logistic Regression</i>	0,9677	0,9677	0,9677	{'C': 0.1, 'penalty': 'l2', 'solver': 'liblinear'}
Naïve Bayes	0,9355	0,9355	0,9355	{'var_smoothing': 1e-09}
<i>Random Forest</i>	0,9032	1,0000	0,9492	{'criterion': 'entropy', 'max_depth': None, 'min_samples_split': 2, 'n_estimators': 200}
SVM	0,9032	1,0000	0,9492	{'C': 5, 'gamma': 'auto', 'kernel': 'rbf'}
XGBoost	0,9032	1,0000	0,9492	{'colsample_bytree': 1.0, 'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 100, 'reg_alpha': 0.5, 'reg_lambda': 1, 'subsample': 1.0}

Setelah dilakukan *hyperparameter tuning*, terjadi peningkatan pada nilai *recall*, *precision*, dan *F1-score* di beberapa model. Empat dari sembilan model mengalami peningkatan nilai *recall*, yaitu AdaBoost, KNN, *Logistic Regression*, dan XGBoost. Lalu, tiga dari sembilan *model classifier* mengalami peningkatan nilai *precision*, yaitu *Decision Tree*, *Random Forest*, dan XGBoost. Berbeda dengan *recall* dan *precision*, mayoritas *model classifier* mengalami peningkatan nilai pada evaluasi *F1-score*. Tiga *model classifier* yang tidak mengalami peningkatan nilai *F1-score* adalah *Extra Trees*, Naïve Bayes, dan SVM. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada evaluasi data uji setelah *hyperparameter tuning* berturut-turut adalah 0,9140, 0,9632, dan 0,9375. Nilai rata-rata tersebut menunjukkan peningkatan daripada saat tidak dilakukan *hyperparameter tuning*.

Dari ketiga metode pelatihan, *cross validation*, *modelling*, dan *hyperparameter tuning*, didapatkan hasil yang terus meningkat. Hal tersebut diketahui dengan adanya peningkatan di seluruh nilai rata-rata evaluasinya. Secara keseluruhan dari ketiga metode pelatihan tersebut, didapatkan rata-rata *recall*, *precision*, dan *F1-score* terakhir berturut-turut adalah 0,8841, 0,9578, dan 0,9184. Dengan nilai tersebut, *Standard Scaler* menempati posisi pertama dengan total rata-rata evaluasi terbaik di antara metode normalisasi lainnya. Metode *Standard Scaler* akan digunakan pada uji coba berikutnya untuk proses normalisasi.

4.1.3 Uji Coba Model Sintetik Data

Uji coba selanjutnya pada model klasifikasi data gabungan antara citra effluent dialysate dengan data klinis adalah penggunaan model berbasis GAN untuk membuat data sintetik. Terdapat empat model sintetik data dalam uji coba ini, yaitu CTGAN, CTABGAN, CTABGAN+, dan CopulaGAN. Dari keempat model tersebut, akan dipilih satu model dengan performa terbaik berdasarkan evaluasi klasifikasi untuk digunakan pada tahap *deployment* akhir. Seluruh uji coba penggunaan model sintetik data ini dilakukan menggunakan *pre-trained model* dan metode normalisasi terbaik yang telah diperoleh dari uji coba sebelumnya. Data yang digunakan dalam uji coba ini merupakan gabungan dari data asli dan data sintetik yang dihasilkan masing-masing model, dengan tujuan untuk memperluas variasi data latih dan meningkatkan performa generalisasi model klasifikasi. Adapun jumlah data yang digunakan pada masing-masing skenario uji coba model sintetik ditampilkan secara rinci pada **Tabel 4.25**, yang memperlihatkan komposisi data asli dan sintetik yang digunakan untuk pelatihan model klasifikasi akhir.

Tabel 4.25 Proporsi Dataset Uji Coba Model Sintetik Data

Kelas	Jumlah Data			Jumlah	
	Train		Test		
	Asli	Sintetik			
Normal	256	0	86	342	
Abnormal	92	164	31	287	
Total	348	164	117	629	

4.1.3.1 Uji Coba Model CTGAN

Proses pelatihan juga menggunakan metode *cross validation* pada data training dengan membagi data latih menjadi lima *fold*. Adapun hasil dari metode *cross validation* menggunakan CTGAN sebagai model yang digunakan untuk membuat data sintetik terlihat pada **Tabel 4.26**.

Tabel 4.26 Hasil Evaluasi *Cross validation* CTGAN

Model	Recall Test Mean	Precision Test Mean	F1-score Test Mean
AdaBoost	0,9742	0,9686	0,9713
Decision Tree	0,9395	0,9438	0,9409
Extra Trees	0,9741	0,9856	0,9797
KNN	0,9540	0,9630	0,9581
Logistic Regression	0,9483	0,9762	0,9619
Naïve Bayes	0,8279	0,9835	0,8976
Random Forest	0,9799	0,9666	0,9730
SVM	0,9684	0,9768	0,9725
XGBoost	0,9742	0,9477	0,9605

Berdasarkan **Tabel 4.26**, *Random Forest* memiliki nilai rata-rata *recall* yang terbaik dengan skor 0,9799. Bukan hanya itu, terdapat tiga model lain yang cukup kompetitif dengan *Random Forest* yang memiliki nilai rata-rata *recall* lebih dari 0,9700, yaitu AdaBoost, *Extra Trees*, dan XGBoost. Dalam hal *precision*, *Extra Trees* memiliki nilai rata-rata *recall* tertinggi, yaitu 0,9856. Seluruh model juga sangat baik dalam meminimalisasi *false positive*, hal tersebut dibuktikan dengan tidak adanya model yang memiliki nilai rata-rata *precision* di bawah 0,9. *Extra Trees* kembali unggul pada matriks *F1-score* dengan nilai sebesar 0,9797. Terdapat tiga model yang memiliki nilai rata-rata *F1-score* di atas 0,9700, yaitu AdaBoost, *Random Forest*, dan SVM. Secara keseluruhan, rata-rata nilai *recall mean*, *precision mean*, dan *F1-score mean* setiap model pada metode *cross validation* Robust Scaler berturut-turut adalah 0,9489, 0,9680, dan 0,9573.

Selanjutnya dilakukan evaluasi menggunakan data uji dengan menggunakan parameter acak di tiap model *classifier*-nya. Hasil model evaluasi pada data uji tersebut dapat terlihat pada **Tabel 4.27**. Pada evaluasi menggunakan data uji, terdapat empat model dengan performa terbaik pada *recall*, yaitu *Extra Trees*, KNN, *Random Forest*, dan XGBoost. Akan tetapi, model Naïve Bayes memiliki performa *recall* yang memiliki perbedaan signifikan, hanya 0,2903 tidak mencapai angka 0,3000, sedangkan model lain memiliki *recall* di atas 0,8700. Di sisi lain, Naïve Bayes memiliki nilai *precision* terbaik dibanding yang lainnya. Hal tersebut mengindikasikan model Naïve Bayes efektif untuk meminimalisasi *false positive* sehingga

tidak ada pasien normal yang masuk kategori abnormal. Model *Extra Trees* juga tidak kalah bagus dalam hal *precision* dengan mendapatkan nilai 0,9667, satu peringkat di bawah Naïve Bayes. Bukan hanya itu, *Extra Trees* kembali memiliki nilai evaluasi terbaik, kali ini untuk metrik *F1-score*. Naïve Bayes kembali memiliki nilai metrik terendah disebabkan oleh nilai *recall* sebelumnya yang tidak mencapai 0,3000, Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada evaluasi data uji menggunakan CTGAN berturut-turut adalah 0,8423, 0,8906, dan 0,8435.

Tabel 4.27 Hasil Evaluasi *Modelling* CTGAN

Model	Recall	Precision	F1-score
AdaBoost	0,8710	0,7941	0,8308
Decision Tree	0,8710	0,8182	0,8438
Extra Trees	0,9355	0,9667	0,9508
KNN	0,9355	0,9355	0,9355
Logistic Regression	0,9032	0,9333	0,9180
Naïve Bayes	0,2903	1,0000	0,4500
Random Forest	0,9355	0,8286	0,8788
SVM	0,9032	0,9333	0,9180
XGBoost	0,9355	0,8056	0,8657

Untuk memaksimalkan nilai evaluasi pada *modelling*, dilakukan *hyperparameter tuning* pada pelatihan data latih. Parameter terbaik akan dipakai untuk mengevaluasi data uji. Hasil dari evaluasi tersebut terlihat pada **Tabel 4.28**.

Tabel 4.28 Hasil Evaluasi *Hyperparameter tuning* CTGAN

Model	Recall	Precision	F1-score	Best-param
AdaBoost	0,9355	0,7838	0,8529	{'learning_rate': 0.1, 'n_estimators': 50}
Decision Tree	0,8710	0,8182	0,8438	{'criterion': 'gini', 'max_depth': 5, 'min_samples_split': 2}
Extra Trees	0,8710	0,9000	0,8852	{'criterion': 'gini', 'max_depth': 5, 'min_samples_split': 2, 'n_estimators': 200}
KNN	0,9355	0,9355	0,9355	{'metric': 'manhattan', 'n_neighbors': 7, 'weights': 'uniform'}
Logistic Regression	0,9355	0,8056	0,8657	{'C': 0.01, 'penalty': 'l1', 'solver': 'liblinear'}
Naïve Bayes	0,2903	1,0000	0,4500	{'var_smoothing': 1e-09}
Random Forest	0,9355	0,8788	0,9063	{'criterion': 'gini', 'max_depth': 4, 'min_samples_split': 2, 'n_estimators': 200}
SVM	0,9032	0,9655	0,9333	{'C': 1, 'gamma': 'auto', 'kernel': 'rbf'}
XGBoost	0,9355	0,8286	0,8788	{'colsample_bytree': 0.8, 'learning_rate': 0.01, 'max_depth': 5, 'n_estimators': 200, 'reg_alpha': 0, 'reg_lambda': 1, 'subsample': 0.8}

Setelah dilakukan *hyperparameter tuning* dan didapatkan parameter terbaik, terdapat perubahan nilai *recall*, *precision*, dan *F1-score* di beberapa model. Untuk *recall*, AdaBoost dan *Logistic Regression* mengalami peningkatan dari proses *modelling* sebelumnya dengan penambahan masing-masing bernilai 0,0625 dan 0,0323. Akan tetapi, *Extra Trees* mengalami penurunan sebesar 0,0625. Dalam hal *precision*, terdapat tiga model yang mengalami penurunan, yaitu AdaBoost, *Extra Trees*, *Logistic Regression*, dengan total penurunan sebesar 0,2047. Sedangkan, model yang mengalami peningkatan nilai *precision* berjumlah tiga juga, yaitu *Random Forest*, SVM, dan XGBoost, dengan total penambahan nilai sebesar 0,1054. Dalam hal *F1-score*, terdapat dua model yang mengalami penurunan, yaitu *Extra Trees Logistic Regression*, dengan total penurunan sebesar 0,1179. Sementara itu, model-model yang mengalami peningkatan nilai berjumlah empat model, yaitu AdaBoost, *Random Forest*, SVM, dan XGBoost, dengan total peningkatan nilai sebesar 0,0780. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada evaluasi data uji menggunakan *hyperparameter tuning* pada CTGAN berturut-turut adalah 0,8459, 0,8795, dan 0,8391.

Dari ketiga metode pelatihan, *cross validation*, *modelling*, dan *hyperparameter tuning*, didapatkan hasil yang terus meningkat. Hal tersebut diketahui dengan adanya peningkatan di seluruh nilai rata-rata evaluasinya. Secara keseluruhan dari ketiga metode pelatihan tersebut, didapatkan rata-rata *recall*, *precision*, dan *F1-score* terakhir berturut-turut adalah 0,8790, 0,9127, dan 0,8800,

4.1.3.2 Uji Coba Model CTABGAN

Proses pelatihan juga menggunakan metode *cross validation* pada data training dengan membagi data latih menjadi lima *fold*. Adapun hasil dari metode *cross validation* menggunakan CTABGAN sebagai model yang digunakan untuk membuat data sintetik terlihat pada **Tabel 4.4**.

Tabel 4.29 Hasil Evaluasi *Cross validation* CTABGAN

<i>Model</i>	<i>Recall Test Mean</i>	<i>Precision Test Mean</i>	<i>F1-score Test Mean</i>
AdaBoost	0,9610	0,9764	0,9686
<i>Decision Tree</i>	0,9492	0,9346	0,9411
<i>Extra Trees</i>	0,9689	0,9767	0,9727
KNN	0,9845	0,8854	0,9319
<i>Logistic Regression</i>	0,9689	0,9509	0,9595
Naïve Bayes	0,9183	0,9832	0,9493
<i>Random Forest</i>	0,9572	0,9727	0,9646
SVM	0,9533	0,9687	0,9606
XGBoost	0,9492	0,9660	0,9563

Berdasarkan **Tabel 4.29**, KNN memiliki nilai rata-rata *recall* yang terbaik dengan skor 0,9845. Seluruh model juga cukup baik dalam meminimalisasi *false negative*. Hal ini didukung oleh tidak adanya model yang memiliki nilai rata-rata *recall* di bawah 0,9. Dalam hal *precision*, Naïve Bayes memiliki nilai rata-rata *recall* tertinggi, yaitu 0,9832. Di sisi lain, hanya KNN yang memiliki nilai rata-rata *precision* di bawah 0,9. *Extra Trees* kembali unggul pada matriks *F1-score* dengan nilai sebesar 0,9727. Secara keseluruhan, rata-rata nilai *recall mean*, *precision*

mean, dan *F1-score* mean setiap model pada metode *cross validation* Robust Scaler berturut-turut adalah 0,9567, 0,9572, dan 0,9561.

Selanjutnya dilakukan evaluasi menggunakan data uji dengan menggunakan parameter acak di tiap model *classifier*-nya. Hasil model evaluasi pada data uji tersebut dapat terlihat pada **Tabel 4.30**.

Tabel 4.30 Hasil Evaluasi *Modelling* CTABGAN

Model	Recall	Precision	F1-score
AdaBoost	0,8387	0,8125	0,8254
Decision Tree	0,8710	0,9000	0,8852
Extra Trees	0,9355	1,0000	0,9667
KNN	0,9355	0,8529	0,8923
Logistic Regression	0,9355	0,9667	0,9508
Naïve Bayes	0,8387	0,9630	0,8966
Random Forest	0,9355	0,9355	0,9355
SVM	0,9355	0,9063	0,9206
XGBoost	0,9032	0,9333	0,9180

Pada evaluasi menggunakan data uji, terdapat lima model dengan performa terbaik pada *recall*, yaitu *Extra Trees*, KNN, *Logistic Regression*, *Random Forest*, dan SVM, dengan nilai 0,9355. Berbeda dengan CTGAN, tidak ada model yang memiliki nilai *recall* di bawah 0,8000, Dalam hal *precision*, *Extra Trees* memiliki nilai tertinggi diikuti oleh *Logistic Regression* dan Naïve Bayes di posisi kedua dan ketiga. Pada metrik ini, hanya dua model yang memiliki nilai di bawah 0,9000, yaitu AdaBoost dan KNN. Dalam hal *F1-score*, *Extra Trees* menempati posisi pertama. Hal ini mengindikasikan *Extra Trees* menjadi model *classifier* terbaik pada percobaan CTABGAN. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada evaluasi data uji menggunakan CTGAN berturut-turut adalah 0,9032, 0,9189, dan 0,9101.

Untuk memaksimalkan nilai evaluasi pada *modelling*, dilakukan *hyperparameter tuning* pada pelatihan data latih. Parameter terbaik akan dipakai untuk mengevaluasi data uji. Hasil dari evaluasi tersebut terlihat pada **Tabel 4.31**.

Tabel 4.31 Hasil Evaluasi *Hyperparameter tuning* CTABGAN

Model	Recall	Precision	F1-score	Best-param
AdaBoost	0,8387	0,8966	0,8667	{'learning_rate': 1.0, 'n_estimators': 100}
Decision Tree	0,8065	0,9259	0,8621	{'criterion': 'entropy', 'max_depth': None, 'min_samples_split': 2}
Extra Trees	0,9355	0,9355	0,9355	{'criterion': 'entropy', 'max_depth': 10, 'min_samples_split': 2, 'n_estimators': 200}
KNN	0,9355	0,8286	0,8788	{'metric': 'manhattan', 'n_neighbors': 7, 'weights': 'uniform'}
Logistic Regression	0,9355	0,9667	0,9508	{'C': 0.1, 'penalty': 'l2', 'solver': 'liblinear'}
Naïve Bayes	0,8387	0,9630	0,8966	{'var_smoothing': 1e-09}

Model	Recall	Precision	F1-score	Best-param
<i>Random Forest</i>	0,9032	0,9032	0,9032	{'criterion': 'entropy', 'max_depth': 4, 'min_samples_split': 5, 'n_estimators': 200}
SVM	0,9032	0,9655	0,9333	{'C': 10, 'gamma': 'scale', 'kernel': 'rbf'}
XGBoost	0,9032	0,9032	0,9032	{'colsample_bytree': 0.8, 'learning_rate': 0.01, 'max_depth': 5, 'n_estimators': 200, 'reg_alpha': 0.5, 'reg_lambda': 1, 'subsample': 0.8}

Setelah dilakukan *hyperparameter tuning* dan didapatkan parameter terbaik, terdapat perubahan nilai *recall*, *precision*, dan *F1-score* di beberapa model. Untuk *recall*, tidak ada model yang mengalami peningkatan. Hal ini mengindikasikan parameter terbaik pada *hyperparameter tuning* menggunakan lima *fold* tidak cukup efektif untuk meningkatkan performa *recall*. Di sisi lain, *Decision Tree*, *Random Forest*, dan *SVM* justru mengalami penurunan nilai *recall* dengan total sebesar 0,1291. Sedangkan dalam hal *precision*, terdapat tiga model yang mengalami peningkatan nilai, yaitu AdaBoost, *Decision Tree*, dan *SVM*, dengan total nambahnilai sebesar 0,1692. Akan tetapi, terdapat empat model yang mengalami penurunan nilai *precision*, yaitu *Extra Trees*, *KNN*, *Random Forest*, dan *XGBoost*, dengan total penurunan nilai sebesar 0,1512. Dalam hal *F1-score*, hanya dua model yang mengalami peningkatan nilai, yaitu AdaBoost dan *SVM*, dengan total penambahan nilai sebesar 0,0540. Di sisi lain, terdapat lima model yang mengalami penurunan, yaitu *Decision Tree*, *Extra Trees*, *KNN*, *Random Forest*, dan *XGBoost*, dengan total penurunan sebesar 0,1149. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada evaluasi data uji menggunakan *hyperparameter tuning* pada CTGAN berturut-turut adalah 0,8889, 0,9206, dan 0,9034.

Dari ketiga metode pelatihan, *cross validation*, *modelling*, dan *hyperparameter tuning*, didapatkan hasil yang terus meningkat. Hal tersebut diketahui dengan adanya peningkatan di seluruh nilai rata-rata evaluasinya. Secara keseluruhan dari ketiga metode pelatihan tersebut, didapatkan rata-rata *recall*, *precision*, dan *F1-score* terakhir berturut-turut adalah 0,9163, 0,9322, dan 0,9232.

4.1.3.3 Uji Coba Model CTABGAN+

Proses pelatihan juga menggunakan metode *cross validation* pada data training dengan membagi data latih menjadi lima *fold*. Adapun hasil dari metode *cross validation* menggunakan CTABGAN+ sebagai model yang digunakan untuk membuat data sintetik terlihat pada **Tabel 4.32**. Berdasarkan tersebut, *KNN* memiliki nilai rata-rata *recall* yang terbaik dengan skor 0,9922. Akan tetapi, model tersebut memiliki nilai rata-rata *precision* terendah, yaitu 0,7921. Dalam hal *precision*, Naïve Bayes dan *Random Forest* menjadi dua model yang paling unggul dengan nilai rata-rata *precision* berturut-turut adalah 0,9827 dan 0,9804. *Extra Trees* kembali unggul pada matriks *F1-score* dengan nilai sebesar 0,9745 bersama dengan *Random Forest* yang memiliki nilai yang sama. Secara keseluruhan, rata-rata nilai *recall mean*, *precision mean*, dan *F1-score mean* setiap model pada metode *cross validation* Robust Scaler berturut-turut adalah 0,9545, 0,9453, dan 0,9480.

Tabel 4.32 Hasil Evaluasi *Cross validation* CTABGAN+

Model	Recall Test Mean	Precision Test Mean	F1-score Test Mean
AdaBoost	0,9766	0,9591	0,9675
<i>Decision Tree</i>	0,9454	0,9246	0,9344
<i>Extra Trees</i>	0,9689	0,9804	0,9745
KNN	0,9922	0,7921	0,8807
<i>Logistic Regression</i>	0,9689	0,9582	0,9632
Naïve Bayes	0,8948	0,9827	0,9365
<i>Random Forest</i>	0,9689	0,9804	0,9745
SVM	0,9298	0,9720	0,9501
XGBoost	0,9454	0,9578	0,9509

Selanjutnya dilakukan evaluasi menggunakan data uji dengan menggunakan parameter acak di tiap model *classifier*-nya. Hasil model evaluasi pada data uji tersebut dapat terlihat pada **Tabel 4.33**.

Tabel 4.33 Hasil Evaluasi *Modelling* CTABGAN+

Model	Recall	Precision	F1-score
AdaBoost	0,8710	0,9643	0,9153
<i>Decision Tree</i>	0,9032	0,8235	0,8615
<i>Extra Trees</i>	0,9032	1,0000	0,9492
KNN	0,9032	0,7000	0,7887
<i>Logistic Regression</i>	0,9032	0,9655	0,9333
Naïve Bayes	0,7097	0,9565	0,8148
<i>Random Forest</i>	0,9032	0,9655	0,9333
SVM	0,9355	0,9667	0,9508
XGBoost	0,9355	0,8788	0,9063

Pada evaluasi menggunakan data uji, terdapat dua model dengan performa terbaik pada *recall*, yaitu SVM dan XGBoost, dengan nilai 0,9355. Berbeda dengan CTABGAN, terdapat satu model yang memiliki nilai *recall* di bawah 0,8000, yaitu Naïve Bayes. Dalam hal *precision*, *Extra Trees* memiliki nilai tertinggi dengan nilai 1,0000 diikuti oleh SVM, *Random Forest*, dan AdaBoost di posisi kedua, ketiga, dan keempat. Pada metrik ini, terdapat satu model yang memiliki nilai di bawah 0,8000, yaitu KNN. Dalam hal *F1-score*, SVM menempati posisi pertama. Hal ini mengindikasikan SVM menjadi model *classifier* terbaik pada percobaan CTABGAN+ karena konsisten berada di peringkat terbaik. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada evaluasi data uji menggunakan CTGAN berturut-turut adalah 0,8853, 0,9134, dan 0,8948.

Untuk memaksimalkan nilai evaluasi pada *modelling*, dilakukan *hyperparameter tuning* pada pelatihan data latih. Proses *hyperparameter tuning* menggunakan metode GridSearchCV dengan paramater-parameter di setiap model klasifikasi yang telah diatur beberapa kombinasi untuk ditemukan parameter yang terbaik. Parameter terbaik tersebut selanjutnya akan dipakai untuk mengevaluasi data uji. Hasil dari evaluasi tersebut terlihat pada **Tabel 4.34**.

Tabel 4.34 Hasil Evaluasi *Hyperparameter tuning* CTABGAN+

Model	Recall	Precision	F1-score	Best-param
AdaBoost	0,9032	0,8000	0,8485	{'learning_rate': 0.1, 'n_estimators': 50}
Decision Tree	0,9355	0,8056	0,8657	{'criterion': 'entropy', 'max_depth': None, 'min_samples_split': 2}
Extra Trees	0,9355	0,9667	0,9508	{'criterion': 'gini', 'max_depth': 10, 'min_samples_split': 5, 'n_estimators': 200}
KNN	0,9355	0,7838	0,8529	{'metric': 'manhattan', 'n_neighbors': 7, 'weights': 'uniform'}
Logistic Regression	0,9032	0,9333	0,9180	{'C': 0.5, 'penalty': 'l1', 'solver': 'liblinear'}
Naïve Bayes	0,7097	0,9565	0,8148	{'var_smoothing': 1e-09}
Random Forest	0,9032	0,9655	0,9333	{'criterion': 'gini', 'max_depth': 4, 'min_samples_split': 5, 'n_estimators': 200}
SVM	0,9355	0,9355	0,9355	{'C': 0.1, 'gamma': 'scale', 'kernel': 'linear'}
XGBoost	0,9355	0,8529	0,8923	{'colsample_bytree': 1.0, 'learning_rate': 0.01, 'max_depth': 3, 'n_estimators': 100, 'reg_alpha': 0, 'reg_lambda': 0, 'subsample': 0.8}

Setelah dilakukan *hyperparameter tuning* dan didapatkan parameter terbaik, terdapat perubahan nilai *recall*, *precision*, dan *F1-score* di beberapa model. Untuk *recall*, tidak ada model yang mengalami penurunan. Hal ini mengindikasikan bahwa parameter terbaik pada *hyperparameter tuning* menggunakan lima fold cukup efektif untuk meningkatkan performa *recall*. Di sisi lain, terdapat empat model yang mengalami peningkatan nilai *recall*, yaitu AdaBoost, *Decision Tree*, *Extra Trees*, dan KNN, dengan total peningkatan sebesar 0,1291. Sedangkan dalam hal *precision*, hanya satu model yang mengalami peningkatan nilai, yaitu KNN, dengan total nambahnilai sebesar 0,0838. Akan tetapi, terdapat enam model yang mengalami penurunan nilai *precision*, yaitu AdaBoost, *Decision Tree*, *Extra Trees*, *Logistic Regression*, SVM dan XGBoost, dengan total penurunan nilai sebesar 0,3048. Dalam hal *F1-score*, terdapat tiga model yang mengalami peningkatan nilai, yaitu *Decision Tree*, *Extra Trees*, dan KNN, dengan total penambahan nilai sebesar 0,0700, Di sisi lain, terdapat empat model yang mengalami penurunan, yaitu AdaBoost, *Logistic Regression*, SVM, dan XGBoost, dengan total penurunan sebesar 0,1114. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada evaluasi data uji menggunakan *hyperparameter tuning* pada CTGAN berturut-turut adalah 0,8996, 0,8889, dan 0,8902.

Dari ketiga metode pelatihan, *cross validation*, *modelling*, dan *hyperparameter tuning*, didapatkan hasil yang terus meningkat. Hal tersebut diketahui dengan adanya peningkatan di seluruh nilai rata-rata evaluasinya. Secara keseluruhan dari ketiga metode pelatihan tersebut,

didapatkan rata-rata *recall*, *precision*, dan *F1-score* terakhir berturut-turut adalah 0,9131, 0,9159, dan 0,9110,

4.1.3.4 Uji Coba Model CopulaGAN

Proses pelatihan juga menggunakan metode *cross validation* pada data training dengan membagi data latih menjadi lima *fold*. Adapun hasil dari metode *cross validation* menggunakan CopulaGAN sebagai model yang digunakan untuk membuat data sintetik terlihat pada **Tabel 4.35**.

Tabel 4.35 Hasil Evaluasi *Cross validation* CopulaGAN

<i>Model</i>	<i>Recall Test Mean</i>	<i>Precision Test Mean</i>	<i>F1-score Test Mean</i>
AdaBoost	0,9650	0,9762	0,9705
<i>Decision Tree</i>	0,9493	0,9175	0,9327
<i>Extra Trees</i>	0,9492	0,9841	0,9662
KNN	0,9649	0,9649	0,9649
<i>Logistic Regression</i>	0,9298	0,9797	0,9539
Naïve Bayes	0,6998	0,9949	0,8191
<i>Random Forest</i>	0,9493	0,9801	0,9642
SVM	0,9610	0,9809	0,9704
XGBoost	0,9611	0,9624	0,9610

Berdasarkan tersebut, terdapat empat model yang cukup kompetitif dalam meminimalisasi nilai *false negative*, yaitu AdaBoost, KNN, XGBoost, dan SVM. Keempat model tersebut memiliki nilai rata-rata *recall* yang terbaik dengan skornya berturut-turut adalah 0,9650, 0,9649, 0,9611, dan 0,9610, Di sisi lain, Naïve Bayes memiliki nilai rata-rata *recall* terendah dengan nilai hanya 0,6998. Akan tetapi, Naïve Bayes menjadi model yang paling unggul dalam hal *precision* dengan skor rata-rata sebesar 0,9949. Dalam hal *F1-score*, AdaBoost memiliki nilai rata-rata *F1-score* tertinggi dengan skor sebesar 0,9705. Secara keseluruhan, rata-rata nilai *recall mean*, *precision mean*, dan *F1-score mean* setiap model pada metode *cross validation* Robust Scaler berturut-turut adalah 0,9255, 0,9712, dan 0,9448.

Selanjutnya dilakukan evaluasi menggunakan data uji dengan menggunakan parameter acak di tiap model *classifier*-nya. Hasil model evaluasi pada data uji tersebut dapat terlihat pada **Tabel 4.36**.

Tabel 4.36 Hasil Evaluasi *Modelling* CopulaGAN

<i>Model</i>	<i>Recall</i>	<i>Precision</i>	<i>F1-score</i>
AdaBoost	0,8710	0,9310	0,9000
<i>Decision Tree</i>	0,8710	0,8182	0,8438
<i>Extra Trees</i>	0,9032	1,0000	0,9492
KNN	0,9355	0,9063	0,9206
<i>Logistic Regression</i>	0,8710	1,0000	0,9310
Naïve Bayes	0,0968	1,0000	0,1765
<i>Random Forest</i>	0,8710	0,9643	0,9153
SVM	0,9032	1,0000	0,9492
XGBoost	0,8710	0,9310	0,9000

Pada evaluasi menggunakan data uji, model KNN memiliki nilai *recall* terbaik dengan nilai 0,9355. Akan tetapi, Naïve Bayes tidak mampu mendapatkan performa *recall* di atas 0,1000, Hal ini mengindikasikan bahwa model tersebut tidak dapat meminimalisasi *false negatif* pada penggunaan data sintetik menggunakan CopulaGAN. Dalam hal *precision*, terdapat empat model yang memiliki nilai sempurna, yaitu Extra Tree, *Logistic Regression*, Naïve Bayes, dan SVM. Hal ini menunjukkan bahwa model lebih efektif dalam menghindari kesalahan pemberian label abnormal dibandingkan dengan normal. Pada metrik ini, hanya terdapat satu model yang memiliki nilai di bawah 0,8000, yaitu *Decision Tree*. Dalam hal *F1-score*, *Extra Trees* dan SVM memiliki nilai terbaik dengan nilai sebesar 0,9492. Sedangkan Naïve Bayes, memiliki nilai *F1-score* tidak sampai 0,2000, Hal ini disebabkan oleh nilai *recall* sebelumnya yang tidak mencapai 0,1. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada evaluasi data uji menggunakan CTGAN berturut-turut adalah 0,7993, 0,9501, dan 0,8317.

Untuk memaksimalkan nilai evaluasi pada *modelling*, dilakukan *hyperparameter tuning* pada pelatihan data latih. Parameter terbaik akan dipakai untuk mengevaluasi data uji. Hasil dari evaluasi tersebut terlihat pada **Tabel 4.37**.

Tabel 4.37 Hasil Evaluasi *Hyperparameter tuning* CopulaGAN

Model	Recall	Precision	F1-score	Best-param
AdaBoost	0,9032	0,9655	0,9333	{'learning_rate': 1.0, 'n_estimators': 100}
<i>Decision Tree</i>	0,8710	0,9310	0,9000	{'criterion': 'entropy', 'max_depth': None, 'min_samples_split': 4}
<i>Extra Trees</i>	0,9032	1,0000	0,9492	{'criterion': 'entropy', 'max_depth': None, 'min_samples_split': 2, 'n_estimators': 100}
KNN	0,8710	0,9000	0,8852	{'metric': 'euclidean', 'n_neighbors': 3, 'weights': 'uniform'}
<i>Logistic Regression</i>	0,9355	0,9667	0,9508	{'C': 1, 'penalty': 'l1', 'solver': 'liblinear'}
Naïve Bayes	0,0968	1,0000	0,1765	{'var_smoothing': 1e-09}
<i>Random Forest</i>	0,8710	0,9643	0,9153	{'criterion': 'gini', 'max_depth': 4, 'min_samples_split': 5, 'n_estimators': 100}
SVM	0,9355	0,9667	0,9508	{'C': 0.1, 'gamma': 'scale', 'kernel': 'linear'}
XGBoost	0,9355	1,0000	0,9667	{'colsample_bytree': 0.8, 'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 200, 'reg_alpha': 0, 'reg_lambda': 1, 'subsample': 1.0}

Setelah dilakukan *hyperparameter tuning* dan didapatkan parameter terbaik, terdapat perubahan nilai *recall*, *precision*, dan *F1-score* di beberapa model. Untuk *recall*, hanya satu model yang mengalami penurunan, yaitu KNN, dengan nilai penurunan sebesar 0,0645. Di sisi lain, terdapat empat model yang mengalami peningkatan nilai, yaitu AdaBoost, *Logistic Regression*, SVM, XGBoost, dengan total peningkatan sebesar 0,1935. Hal ini mengindikasikan bahwa parameter terbaik pada *hyperparameter tuning* menggunakan lima fold cukup efektif untuk meningkatkan performa *recall*. Sedangkan dalam hal *precision*,

terdapat tiga model yang mengalami peningkatan nilai, yaitu AdaBoost, *Decision Tree*, dan XGBoost, dengan total nambahnilai sebesar 0,2163. Akan tetapi, terdapat tiga model yang mengalami penurunan nilai *precision*, yaitu KNN, *Logistic Regression*, dan SVM, dengan total penurunan nilai sebesar 0,0729. Dalam hal *F1-score*, hanya terdapat satu model saja yang mengalami penurunan nilai, yaitu KNN, dengan nilai penurunan sebesar 0,0354. Di sisi lain, terdapat lima model yang mengalami peningkatan nilai, yaitu AdaBoost, *Decision Tree*, *Logistic Regression*, SVM, dan XGBoost, dengan total penambahan nilai sebesar 0,1776. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada evaluasi data uji menggunakan *hyperparameter tuning* pada CTGAN berturut-turut adalah 0,8136, 0,9660, dan 0,8475.

Dari ketiga metode pelatihan, *cross validation*, *modelling*, dan *hyperparameter tuning*, didapatkan hasil yang terus meningkat. Hal tersebut diketahui dengan adanya peningkatan di seluruh nilai rata-rata evaluasinya. Secara keseluruhan dari ketiga metode pelatihan tersebut, didapatkan rata-rata *recall*, *precision*, dan *F1-score* terakhir berturut-turut adalah 0,8461, 0,9624, dan 0,8747. Dengan nilai tersebut, penggunaan data sintetik yang dihasilkan dari model CopulaGAN menempati posisi ketiga terbaik di antara metode sintetik lainnya. Metode pembuatan data sintetik dengan nilai rata-rata tertinggi dicapai oleh CTABGAN. Metode tersebut akan digunakan pada uji coba berikutnya untuk proses uji coba *classifiers*.

4.1.4 Uji Coba *Classifiers*

Uji coba terakhir pada model klasifikasi data gabungan antara citra *effluent dialysate* dengan data klinis adalah penggunaan jenis model *classifier*. Terdapat dua jenis yang akan dibandingkan, yaitu *single learning* dan *ensemble leanring*. Dari kedua model tersebut, akan dipilih model *classifier* terbaik untuk dipakai pada proses *deployment* jika diperlukan. Seluruh uji coba penggunaan jenis model *classifier* ini memakai *pre-trained model*, metode normaliasi, dan model GAN terbaik yang didapatkan pada uji coba sebelumnya. Data yang digunakan berupa gabungan dari data asli dan data sintetik yang dihasilkan menggunakan CTABGAN. Adapun jumlah data yang dipakai untuk seluruh uji coba model *classifier* ini terlihat pada **Tabel 4.38**.

Tabel 4.38 Proporsi Dataset Uji Coba *Classifiers*

Kelas	Jumlah Data			Jumlah	
	Train		Test		
	Asli	Sintetik			
Normal	256	0	86	342	
Abnormal	92	164	31	287	
Total	348	164	117	629	

4.1.4.1 Uji Coba Model *Single Learning*

Proses pelatihan pada model *single learning* juga menggunakan metode *cross validation* pada data training dengan membagi data latih menjadi lima *fold*. Pada setiap *fold*, terdapat pengujian pada data *test*. Nilai rata-rata dari masing-masing metrik evaluasi dari pengujian data uji tersebut yang digunakan sebagai pembanding dengan uji coba menggunakan metode lain. Adapun hasil dari metode *cross validation* menggunakan *single learning* sebagai model *classifier* terlihat pada **Tabel 4.39**.

Tabel 4.39 Hasil Evaluasi *Cross validation Single Learning*

Model	Recall Test Mean	Precision Test Mean	F1-score Test Mean
<i>Decision Tree</i>	0,9492	0,9346	0,9411
KNN	0,9845	0,8854	0,9319
<i>Logistic Regression</i>	0,9689	0,9509	0,9595
Naïve Bayes	0,9183	0,9832	0,9493
SVM	0,9533	0,9687	0,9606

Berdasarkan tabel 4.39, seluruh model *single learning* memiliki nilai rata-rata *recall* lebih dari 0,9. KNN menjadi model *single learning* yang memiliki nilai rata-rata *recall* tertinggi dengan nilai sebesar 0,9845. Dalam hal *precision*, Naïve Bayes memiliki nilai rata-rata tertinggi dengan skor 0,9832. Di sisi lain, KNN menjadi satu-satunya model yang tidak berhasil mencapai 0,9 pada rata-rata *precision*. Seperti *recall*, seluruh model *single learning* pada metrik *F1-score* memiliki nilai rata-rata lebih dari 0,9. SVM menjadi model *single learning* yang memiliki nilai rata-rata *F1-score* tertinggi dengan nilai sebesar 0,9606. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* untuk evaluasi setiap model pada metode *cross validation single learning* berturut-turut adalah 0,9548, 0,9446, dan 0,9485.

Selanjutnya dilakukan evaluasi menggunakan data uji dengan menggunakan parameter acak di tiap model *classifier*-nya. Hasil model evaluasi pada data uji tersebut dapat terlihat pada **Tabel 4.40**,

Tabel 4.40 Hasil Evaluasi *Modelling Single Learning*

Model	Recall	Precision	F1-score
<i>Decision Tree</i>	0,8710	0,9000	0,8852
KNN	0,9355	0,8529	0,8923
<i>Logistic Regression</i>	0,9355	0,9667	0,9508
Naïve Bayes	0,8387	0,9630	0,8966
SVM	0,9355	0,9063	0,9206

Pada evaluasi menggunakan data uji, terdapat tiga model dengan nilai rata-rata *recall* terbaik, yaitu KNN, *Logistic Regression*, dan SVM, dengan skor 0,9355. Dalam hal *precision*, *Logistic Regression* dan Naïve Bayes menjadi dua model dengan nilai rata-rata tertinggi dengan skor berturut-turut adalah 0,9667 dan 0,9630, Dalam hal *F1-score*, *Logistic Regression* memiliki nilai rata-rata terbaik dengan skor sebesar 0,9508. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada evaluasi data uji menggunakan *single learning* sebagai model *classifier*-nya berturut-turut adalah 0,9032, 0,9178, dan 0,9091.

Untuk memaksimalkan nilai evaluasi pada *modelling*, dilakukan *hyperparameter tuning* pada pelatihan data latih. Parameter terbaik akan dipakai untuk mengevaluasi data uji. Hasil dari evaluasi tersebut terlihat pada **Tabel 4.41**. Setelah dilakukan *hyperparameter tuning* dan didapatkan parameter terbaik, terdapat perubahan nilai *recall*, *precision*, dan *F1-score* di beberapa model. Untuk *recall*, tidak ada model yang mengalami peningkatan dibanding dengan pelatihan sebelumnya yang tidak menggunakan *hyperparameter tuning*. Bukan hanya itu, terdapat dua model yang mengalami penurunan, yaitu *Decision Tree* dan SVM. Kedua model tersebut mengalami penurunan masing-masing sebesar 0,0645 dan 0,0323. Dalam hal *precision*,

terdapat dua model yang mengalami peningkatan, yaitu *Decision Tree* dan *SVM*. Nilai peningkatan dari kedua model tersebut berturut-turut adalah 0,0259 dan 0,0592. Akan tetapi, terdapat model yang mengalami penurunan nilai rata-rata *precision*, yaitu *KNN*, dengan nilai penurunan sebesar 0,0243. Berbeda dengan *precision*, nilai rata-rata pada metrik *F1-score* mengalami dua penurunan dan satu peningkatan nilai. *Decision Tree* dan *KNN* adalah dua model yang mengalami penurunan. Satu-satunya model yang mengalami peningkatan adalah *SVM*. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada data uji menggunakan *single learning* sebagai model *classifier*-nya berturut-turut adalah 0,8839, 0,9299, dan 0,9043.

Tabel 4.41 Hasil Evaluasi *Hyperparameter tuning Single Learning*

Model	Recall	Precision	F1-score	Best-param
<i>Decision Tree</i>	0,8065	0,9259	0,8621	{'criterion': 'entropy', 'max_depth': None, 'min_samples_split': 2}
<i>KNN</i>	0,9355	0,8286	0,8788	{'metric': 'manhattan', 'n_neighbors': 7, 'weights': 'uniform'}
<i>Logistic Regression</i>	0,9355	0,9667	0,9508	{'C': 0.1, 'penalty': 'l2', 'solver': 'liblinear'}
<i>Naïve Bayes</i>	0,8387	0,9630	0,8966	{'var_smoothing': 1e-09}
<i>SVM</i>	0,9032	0,9655	0,9333	{'C': 10, 'gamma': 'scale', 'kernel': 'rbf'}

Setelah dilakukan *hyperparameter tuning* dan didapatkan parameter terbaik, terdapat perubahan nilai *recall*, *precision*, dan *F1-score* di beberapa model. Untuk *recall*, tidak ada model yang mengalami peningkatan dibanding dengan pelatihan sebelumnya yang tidak menggunakan *hyperparameter tuning*. Bukan hanya itu, terdapat dua model yang mengalami penurunan, yaitu *Decision Tree* dan *SVM*. Kedua model tersebut mengalami penurunan masing-masing sebesar 0,0645 dan 0,0323. Dalam hal *precision*, terdapat dua model yang mengalami peningkatan, yaitu *Decision Tree* dan *SVM*. Nilai peningkatan dari kedua model tersebut berturut-turut adalah 0,0259 dan 0,0592. Akan tetapi, terdapat model yang mengalami penurunan nilai rata-rata *precision*, yaitu *KNN*, dengan nilai penurunan sebesar 0,0243. Berbeda dengan *precision*, nilai rata-rata pada metrik *F1-score* mengalami dua penurunan dan satu peningkatan nilai. *Decision Tree* dan *KNN* adalah dua model yang mengalami penurunan. Satu-satunya model yang mengalami peningkatan adalah *SVM*. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada data uji menggunakan *single learning* sebagai model *classifier*-nya berturut-turut adalah 0,8839, 0,9299, dan 0,9043.

Dari ketiga metode pelatihan, *cross validation*, *modelling*, dan *hyperparameter tuning*, didapatkan hasil yang terus meningkat. Hal tersebut diketahui dengan adanya peningkatan di seluruh nilai rata-rata evaluasinya. Secara keseluruhan dari ketiga metode pelatihan tersebut, didapatkan rata-rata *recall*, *precision*, dan *F1-score* terakhir berturut-turut adalah 0,9140, 0,9308, dan 0,9206.

4.1.4.2 Uji Coba Model *Ensemble Learning*

Proses pelatihan juga menggunakan metode *cross validation* pada data training dengan membagi data latih menjadi lima *fold*. Adapun hasil dari metode *cross validation* menggunakan *ensemble learning* sebagai model *classifier* terlihat pada **Tabel 4.42**.

Tabel 4.42 Hasil Evaluasi *Cross validation Ensemble Learning*

Model	Recall Test Mean	Precision Test Mean	F1-score Test Mean
AdaBoost	0,9610	0,9764	0,9686
Extra Trees	0,9689	0,9767	0,9727
Random Forest	0,9572	0,9727	0,9646
XGBoost	0,9492	0,9660	0,9563

Berdasarkan **Tabel 4.42**, *Extra Trees* menjadi model *ensemble learning* yang memiliki nilai rata-rata *recall* tertinggi dengan nilai sebesar 0,9689. AdaBoost berada di peringkat kedua dengan nilai rata-rata *recall* sebesar 0,9610, hanya kurang 0,0079 dari nilai rata-rata *recall* *Extra Trees*. Dalam hal *precision*, seluruh model *ensemble learning* memiliki nilai rata-rata yang cukup kompetitif karena memiliki skor 0,96 dan 0,97. *Extra Trees* memiliki nilai rata-rata tertinggi dengan skor 0,9767. Dalam hal *F1-score*, *Extra Trees* memiliki nilai rata-rata tertinggi, yaitu 0,9727. Dari ketiga metriks evaluasi tersebut, seluruh model *ensemble learning* memiliki nilai di atas 0,95. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada evaluasi *cross validation* menggunakan *ensemble learning* sebagai model *classifier*-nya berturut-turut adalah 0,9591, 0,9730, dan 0,9656

Selanjutnya dilakukan evaluasi menggunakan data uji dengan menggunakan parameter acak di tiap model *classifier*-nya. Hasil model evaluasi pada data uji tersebut dapat terlihat pada **Tabel 4.43**.

Tabel 4.43 Hasil Evaluasi *Modelling Ensemble Learning*

Model	Recall	Precision	F1-score
AdaBoost	0,8387	0,8125	0,8254
Extra Trees	0,9355	1.0000	0,9667
Random Forest	0,9355	0,9355	0,9355
XGBoost	0,9032	0,9333	0,9180

Pada evaluasi menggunakan data uji, terdapat dua model dengan nilai rata-rata *recall* terbaik, yaitu *Extra Trees* dan *Random Forest*, dengan skor 0,9355. Dalam hal *precision*, *Extra Trees* memiliki nilai rata-rata sempurnya. *Random Forest* dan XGBoost menempati posisi kedua dan ketiga nilai rata-rata *precision* terbaik. Nilai rata-rata *precision* kedua model tersebut berturut-turut adalah 0,9355 dan 0,9333. Dalam hal *F1-score*, *Extra Trees* memiliki nilai rata-rata terbaik dengan skor sebesar 0,9667. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada evaluasi data uji menggunakan *single learning* sebagai model *classifier*-nya berturut-turut adalah 0,9032, 0,9203, 0,9114.

Untuk memaksimalkan nilai evaluasi pada *modelling*, dilakukan *hyperparameter tuning* pada pelatihan data latih. Proses *hyperparameter tuning* menggunakan metode GridSearchCV dengan paramater-parameter di setiap model klasifikasi yang telah diatur beberapa kombinasi untuk ditemukan parameter yang terbaik. Parameter terbaik tersebut selanjutnya akan dipakai untuk mengevaluasi data uji. Hasil dari evaluasi tersebut terlihat pada **Tabel 4.44**.

Tabel 4.44 Hasil Evaluasi *Hyperparameter tuning Ensemble Learning*

Model	Recall	Precision	F1-score	Best-param
AdaBoost	0,8387	0,8966	0,8667	{'learning_rate': 1.0, 'n_estimators': 100}
Extra Trees	0,9355	0,9355	0,9355	{'criterion': 'entropy', 'max_depth': 10, 'min_samples_split': 2, 'n_estimators': 200}
Random Forest	0,9032	0,9032	0,9032	{'criterion': 'entropy', 'max_depth': 4, 'min_samples_split': 5, 'n_estimators': 200}
SVM	0,9032	0,9032	0,9032	{'colsample_bytree': 0.8, 'learning_rate': 0.01, 'max_depth': 5, 'n_estimators': 200, 'reg_alpha': 0.5, 'reg_lambda': 1, 'subsample': 0.8}

Setelah dilakukan *hyperparameter tuning* dan didapatkan parameter terbaik, terdapat perubahan nilai *recall*, *precision*, dan *F1-score* di beberapa model. Untuk *recall*, tidak ada model yang mengalami peningkatan dibanding dengan pelatihan sebelumnya yang tidak menggunakan *hyperparameter tuning*. Bukan hanya itu, terdapat model yang mengalami penurunan, yaitu *Random Forest*. Model tersebut mengalami penurunan nilai sebesar 0,0323. Dalam hal *precision*, hanya satu model yang mengalami peningkatan, yaitu AdaBoost. Nilai peningkatan tersebut sebesar 0,0841. Akan tetapi, ketiga model lainnya mengalami penurunan. *Extra Trees* mengalami penurunan rata-rata *precision* terbesar, 0,0645. Sama halnya dengan *precision*, nilai rata-rata pada metrik *F1-score* mengalami tiga penurunan dan satu peningkatan nilai. AdaBoost menjadi satu-satunya model yang mengalami peningkatan. Secara keseluruhan, rata-rata nilai *recall*, *precision*, dan *F1-score* pada data uji menggunakan *single learning* sebagai model *classifier*-nya berturut-turut adalah 0,8952, 0,9096, dan 0,9022.

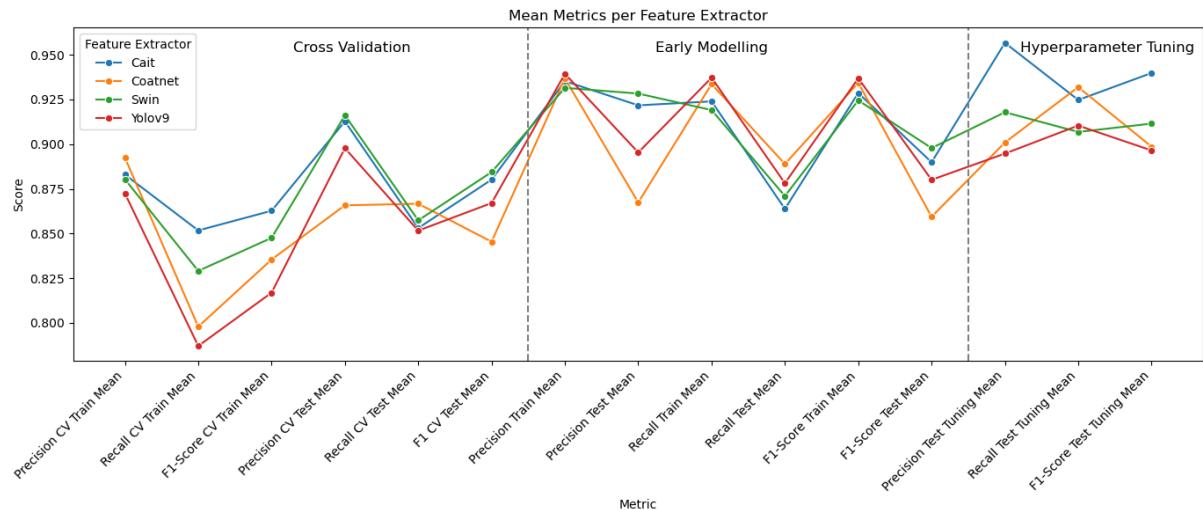
Dari ketiga metode pelatihan, *cross validation*, *modelling*, dan *hyperparameter tuning*, didapatkan hasil yang terus meningkat. Hal tersebut diketahui dengan adanya peningkatan di seluruh nilai rata-rata evaluasinya. Secara keseluruhan dari ketiga metode pelatihan tersebut, didapatkan rata-rata *recall*, *precision*, dan *F1-score* terakhir berturut-turut adalah 0,9192, 0,9343, dan 0,9264. Dengan nilai tersebut, penggunaan *ensemble learning* memiliki nilai rata-rata keseluruhan lebih tinggi daripada *single learning*.

4.2 Pembahasan

Bab ini membahas hasil evaluasi model dalam mendekripsi adanya potensi keadaan tidak normal pasien selama menjalani proses CAPD. Terdapat tiga tahap evaluasi, yaitu *cross validation*, *early modelling* (*modelling* dengan parameter acak tanpa *tuning*), dan *modelling* menggunakan parameter terbaik setelah dilakukan *hyperparameter tuning*. Proses *cross validation* digunakan untuk menguji stabilitas *training* pada data latih. Proses *early modelling* digunakan untuk acuan awal proses pelatihan menggunakan data latih, tanpa dibagi menjadi beberapa *fold*, dan pengujian menggunakan data uji. Proses terakhir, *hyperparameter tuning*, digunakan untuk mengoptimalkan proses *modelling* sebelumnya dengan mencari parameter *model classifier* terbaik.

4.2.1 Pembahasan Uji Coba Feature Extractor

Terdapat empat *pre-trained model* yang digunakan sebagai *feature extractor*, yaitu CaiT, CoAtNet, Swin Transformer, dan YOLOv9. Data yang telah mengandung hasil *feature extraction* dengan empat *pre-trained model* tersebut, dilanjutkan ke proses selanjutnya untuk klasifikasi kondisi pasien.



Gambar 4.2 Grafik Nilai Evaluasi Uji Coba Feature Extractor

Dari grafik yang terdapat pada **Gambar 4.2**, CaiT menjadi *model feature extractor* terbaik dengan empat kali posisi pertama, enam kali posisi kedua, empat kali posisi ketiga, dan hanya sekali memiliki nilai terendah. Pada tahap *cross validation*, CaiT dan Swin Transformer relatif mendominasi di peringkat pertama baik pada data latih maupun data uji. Hal tersebut mengindikasikan penggunaan *pre-trained model feature extractor* berbasis *transformer* memiliki kestabilan evaluasi yang lebih baik daripada *feature extractor* yang mengandung CNN.

Pada tahap *early modelling*, CaiT tidak sekalipun menempati peringkat pertama. Hal tersebut berbeda dengan tahap sebelumnya karena di tahap ini data latih tidak dibagi menjadi beberapa *fold* sehingga terdapat perbedaan distribusi data latih yang digunakan untuk setiap pelatihan. Setelah ditemukan parameter terbaik melalui tahap *hyperparameter tuning*, CaiT kembali dominan dengan dua kali peringkat pertama, yaitu pada nilai *precision* dan *F1-score*, dan peringkat kedua pada *recall*. Dengan fokus utama pada *recall*, CaiT menjadi model *feature extractor* terbaik dibanding dengan *feature extractor* lainnya. Adapun ringkasan hasil uji model *feature extractor* dapat terlihat pada **Tabel 4.45**.

Tabel 4.45 Ringkasan Nilai Evaluasi Uji Coba Feature Extractor

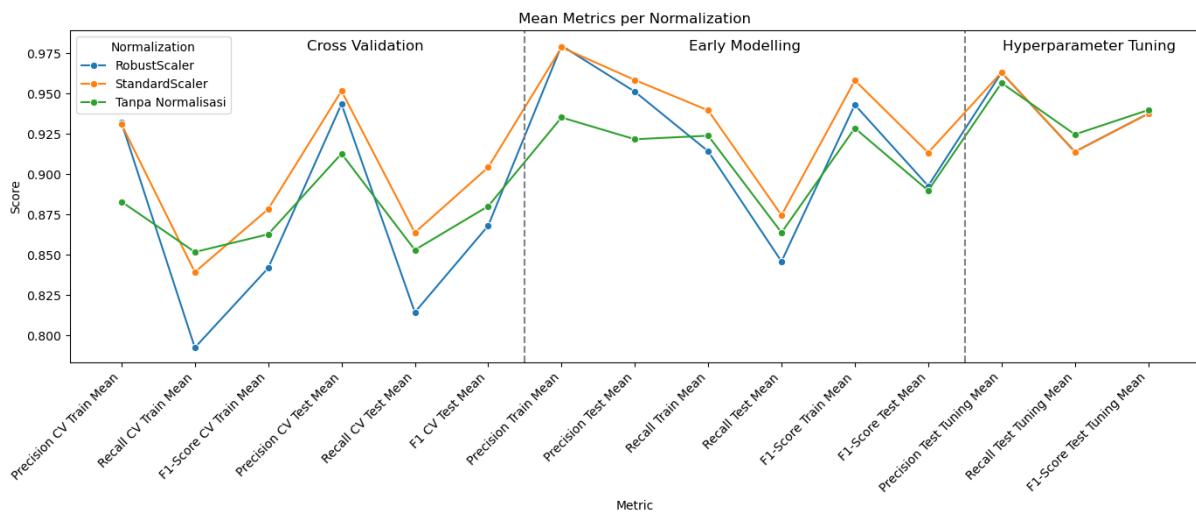
No.	Feature Extractor	Recall Mean	Precision Mean	F1-score Mean
1	CaiT	0,8841	0,9434	0,9113
2	Swin Transformer	0,8789	0,9363	0,9053
3	YOLOv9	0,8769	0,8866	0,8671
4	CoAtNet	0,8712	0,8941	0,8515

Pada tabel tersebut ditemukan juga dari keempat *model feature extractor* yang digunakan, penggunaan *feature extractor* berbasis *transformer* lebih baik dibandingkan dengan yang

berbasis CNN. Hal tersebut dapat disebabkan oleh kemampuan arsitektur *transformer* dalam menangkap relasi global antar piksel melalui mekanisme *self-attention*, berbeda dengan CNN yang cenderung fokus pada fitur lokal dalam *receptive field* terbatas. *Transformer* seperti CaiT mampu menyusun representasi yang lebih menyeluruh dari gambar, terutama pada data citra medis yang umumnya memiliki pola halus dan konteks menyeluruh sebagai penanda abnormalitas..

4.2.2 Pembahasan Uji Coba Normalisasi

Untuk proses normalisasi, penelitian ini menggunakan tiga skema, yaitu tanpa normalisasi, *standard scaler*, dan *robust scaler*. Data yang telah melewati proses normalisasi, dilanjutkan ke proses selanjutnya untuk klasifikasi kondisi pasien.



Gambar 4.3 Grafik Nilai Evaluasi Uji Coba Normalisasi

Dari grafik yang terdapat pada **Gambar 4.3**, *standard scaler* menjadi metode normalisasi terbaik dengan sebelas kali posisi pertama dan empat kali posisi kedua. Pada tahap *cross validation*, *standard scaler* terlihat cukup dominan terutama pada data uji. Pada tahap *early modelling*, *standard scaler* selalu menempati peringkat pertama. Hal tersebut mengindikasikan bahwa penggunaan *standard scaler* sebagai metode normalisasi relatif stabil. Setelah ditemukan parameter terbaik melalui tahap *hyperparameter tuning*, *standard scaler* mengalami peningkatan nilai *recall* dan *F1-score* pada data uji. Dengan fokus utama pada *recall*, *standard scaler* terlihat unggul dibanding dengan metode normalisasi lainnya. Adapun ringkasan hasil uji metode normalisasi dapat terlihat pada **Tabel 4.46**.

Tabel 4.46 Ringkasan Nilai Evaluasi Uji Coba Normalisasi

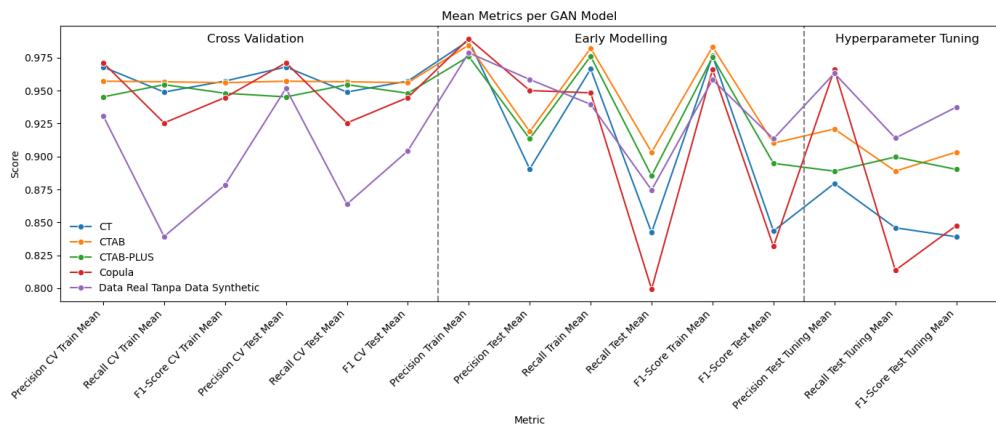
No.	Normalisasi	Recall Mean	Precision Mean	F1-score Mean
1	Standard Scaler	0,8841	0,9578	0,9184
2	Tanpa Normalisasi	0,8805	0,9304	0,9033
3	Robust Scaler	0,8581	0,9526	0,8995

Pada tabel tersebut diperjelas bahwa *standard scaler* merupakan metode normalisasi terbaik pada penelitian ini, khususnya dalam rata-rata *recall* yang menjadi fokus dari penelitian ini. Hal tersebut disebabkan oleh kemampuannya menyamakan skala distribusi antara fitur hasil ekstraksi citra dan data tabular klinis dengan mentransformasi data menjadi distribusi dengan

mean nol dan standar deviasi satu. Dengan pendekatan ini, pengaruh dari *outliers* tetap dipertahankan. Hal tersebut dapat menjadi pertanda abnormalitas pada data medis. Pendekatan ini berbeda dengan *robust scaler* yang meredam pengaruh *outliers* dengan penggunaan median atau IQR sebagai metode *scalling*-nya.

4.2.3 Pembahasan Uji Coba Model Sintetik Data

Pada pembuatan data sintetik digunakan empat jenis model sintetik berbasis GAN, yaitu CTGAN, CTABGAN, CTABGAN+, dan CopulaGAN. Pembuatan data sintetik hanya menggunakan kelas minoritas pada data latih agar dapat mengatasi *class imbalance* dan meningkatkan representasi kelas abnormal yang jumlahnya lebih sedikit dibandingkan kelas normal dalam *dataset*. Data sintetik yang telah dibuat lalu digabungkan dengan data asli sebelum dilakukan proses normalisasi dan *modelling*.



Gambar 4.4 Grafik Nilai Evaluasi Uji Coba Model Sintetik Data

Dari grafik yang terdapat pada **Gambar 4.4**, penggunaan tambahan data sintetik pada proses *cross validation* hampir selalu lebih baik daripada ketika hanya digunakan data asli saja. Hal tersebut mengindikasikan penambahan data sintetik menjadikan proses *modelling* lebih stabil dalam berbagai kondisi data latih dan uji. Pada tahap *early modelling*, data sintetik yang dihasilkan dari model CTABGAN cenderung lebih baik dari model lain atau dari tanpa data sintetik. Hal tersebut dapat dilihat dari frekuensi model CTABGAN memiliki nilai rata-rata terbaik di setiap metriknya. Setelah ditemukan parameter terbaik melalui tahap *hyperparameter tuning*, penggunaan data asli saja tanpa melibatkan data sintetik menjadi lebih baik daripada hampir seluruh keterlibatan data sintetik. Dengan fokus utama pada *recall*, penambahan data sintetik menggunakan model CTABGAN menjadi yang terbaik dibanding kombinasi data asli dan sintetik lainnya. Adapun ringkasan hasil uji metode pemakaian data sintetik dapat terlihat pada **Tabel 4.47**.

Tabel 4.47 Ringkasan Nilai Evaluasi Uji Coba Model Sintetik Data

No.	Model Sintetik Data	Recall Mean	Precision Mean	F1-score Mean
1	CTABGAN	0,9163	0,9322	0,9232
2	Tanpa Data Sintetik	0,8841	0,9578	0,9184
3	CTABGAN+	0,9131	0,9159	0,911
4	CopulaGAN	0,8461	0,9624	0,8747
5	CTGAN	0,8790	0,9127	0,8800

Pada tabel tersebut, ditunjukkan kalau penambahan data sintetik menggunakan CTABGAN dan CTABGAN+ dapat meningkatkan rata-rata *recall*, metrik yang menjadi fokus pada penelitian ini. CTABGAN adalah model penghasil data sintetik dengan nilai rata-rata *recall* terbaik dibanding dengan ketiga model sintetik lainnya. CTABGAN+ menjadi model pembuatan data sintetik terbaik kedua dalam hal *recall*. Hal tersebut disebabkan karena kedua model tersebut memiliki kualitas data sintetik terbaik dibanding dengan kedua model sintetik lainnya.

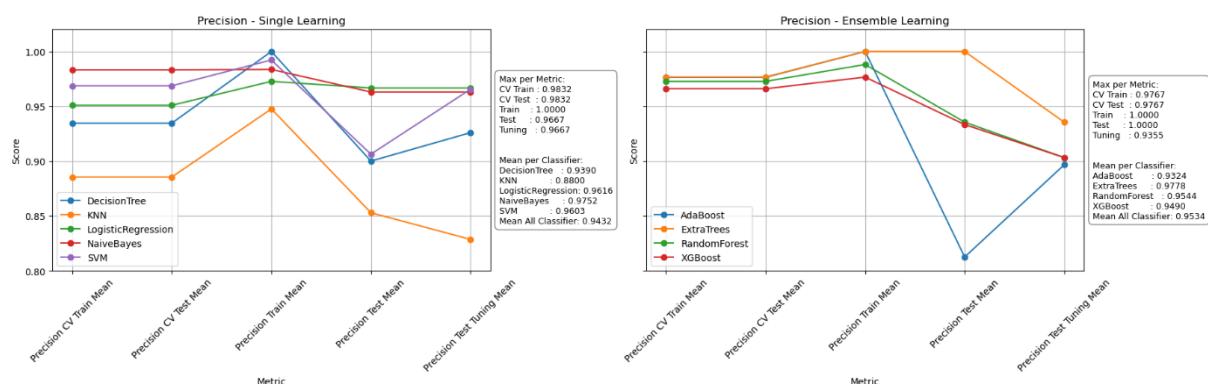
Tabel 4.48 Skor Kualitas Data Sintetik

No.	Model Sintetik Data	Rata-Rata Skor Kualitas Data Sintetik
1	CTABGAN	90.40%
2	CTABGAN+	90.25%
3	CTGAN	89.44%
4	CopulaGAN	86.36%

Penilaian skor kualitas data sintetik dinilai berdasarkan empat parameter utama, yaitu pemeriksaan validitas dasar setiap kolom, seperti keunikan *primary key*, verifikasi kesamaan struktur kolom antara data asli dan sintetik, evaluasi kesamaan distribusi statistik untuk setiap kolom tunggal, serta analisis kesamaan korelasi dan distribusi bivariat antar pasangan kolom untuk memastikan hubungan antar variabel tetap terjaga dalam data sintetik yang dihasilkan.

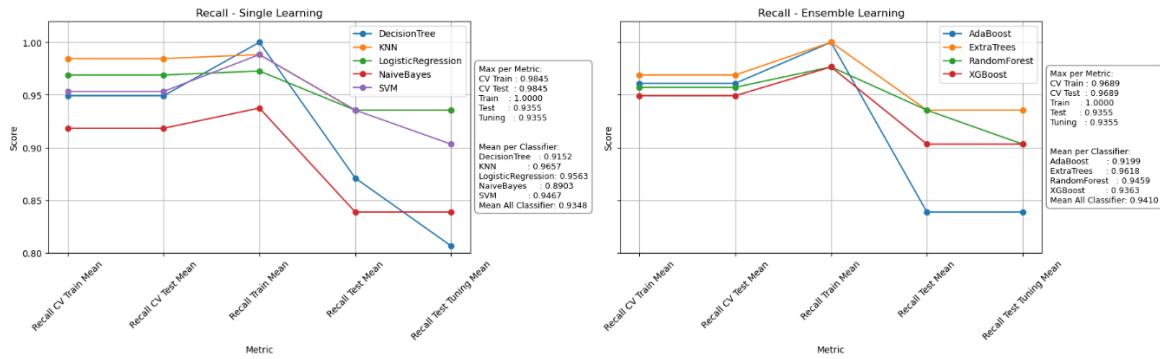
4.2.4 Pembahasan Uji Coba *Classifier*

Dengan menggunakan metode terbaik di semua uji coba sebelumnya, didapatkan hasil terbaik adalah kombinasi penggunaan CaiT sebagai *feature extractor model*, *standard scaler* sebagai metode normalisasi, dan CTABGAN sebagai model pembuatan sintetik data. Dengan kombinasi terbaik tersebut, dilakukan uji coba terakhir, yaitu perbandingan penggunaan *single learning* dan *ensemble learning* pada *classifier*-nya. Dari sembilan jenis model *classifier* yang digunakan, lima di antaranya termasuk ke dalam *single learning* dan empat di antaranya termasuk *ensemble learning*.



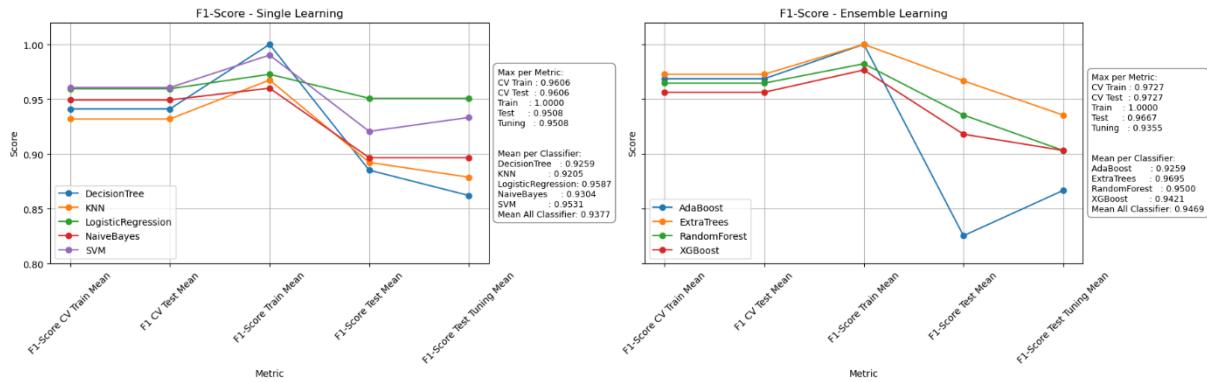
Gambar 4.5 Grafik Nilai Evaluasi *Precision Classifiers*

Dari grafik *precision* yang terdapat pada **Gambar 4.4**, model *single learning* unggul pada *cross validation* dan *modelling* menggunakan *hyperparameter tuning*. Untuk proses *modelling* menggunakan parameter awal, model *ensemble learning* memiliki nilai maksimum yang lebih tinggi dibandingkan dengan *single learning*. Secara keseluruhan, model *ensemble learning* memiliki nilai rata-rata *precision* yang lebih tinggi daripada model *single learning*.



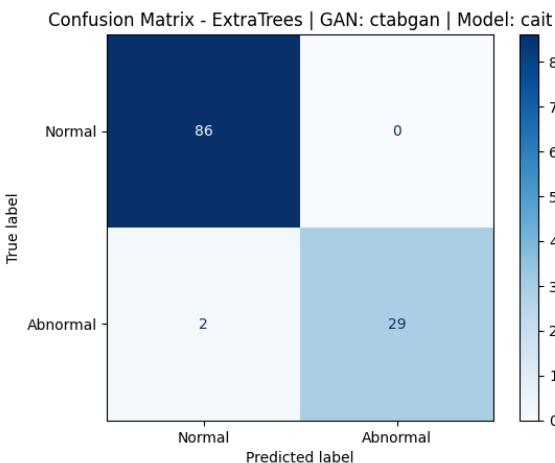
Gambar 4.6 Grafik Nilai Evaluasi *Recall Classifiers*

Dari grafik *recall* yang terdapat pada **Gambar 4.6**, model *single learning* unggul pada *cross validation*. Untuk proses lainnya, model *ensemble learning* memiliki nilai maksimum yang sama dengan *single learning*. Secara keseluruhan, model *ensemble learning* memiliki nilai rata-rata *recall* yang lebih tinggi daripada model *single learning*.



Gambar 4.7 Grafik Nilai Evaluasi *F1-score Classifiers*

Dari grafik *F1-score* yang terdapat pada **Gambar 4.7**, model *single learning* hanya unggul pada *modelling* menggunakan parameter awal. Untuk proses *modelling* lainnya, model *ensemble learning* memiliki nilai maksimum yang lebih tinggi dibandingkan dengan *single learning*. Secara keseluruhan, model *ensemble learning* memiliki nilai rata-rata *recall* yang lebih tinggi daripada model *single learning*.



Gambar 4.8 Confusion Matrics

Dari hasil rata-rata keseluruhan metrik evaluasi tersebut, model *ensemble learning* lebih baik untuk meminimalisasi baik *false negative* maupun *false positive*. Dari keempat model *ensemble learning*, *Extra Trees* menjadi model *classifier* di seluruh metrik evaluasi. Dari seluruh proses metrik evaluasi dan skema, *Extra Trees* memiliki nilai terbaik pada *modelling* menggunakan parameter awal dengan hanya mendapatkan dua *false negative* dan tidak ada *false positive*, terlihat pada **Gambar 4.8**. Hal tersebut dapat disebabkan karena pada model *classifier* tersebut terdapat pemilihan fitur dan penentuan ambang batas secara acak sehingga dapat meningkatkan diversitas *tree* dalam model. Randomisasi tinggi di setiap pohon memungkinkan model tidak terlalu bergantung pada fitur dominan sehingga lebih mampu menggeneralisasi ke data baru. Dalam *ensemble learning*, keberagaman ini memungkinkan menghasilkan prediksi yang lebih stabil dan akurat khususnya untuk data berdimensi tinggi seperti data yang digunakan pada penelitian ini.



durasi_drain_menit	volume_masuk	volume_keluar	berat_badan	systolic
12	2000	2000	79	146
diastolic	nadi	jenis_kelamin	jam_masuk	hari_masuk
82	64	0	5	3
dwell_time_menit	jam_keluar	hari_keluar	selisih_volume	shift_penggantian
306	10	3	0	1

durasi_drain_menit	volume_masuk	volume_keluar	berat_badan	systolic
15	1500	1764	50.745	131
diastolic	nadi	jenis_kelamin	jam_masuk	hari_masuk
87	67	0	22	3
dwell_time_menit	jam_keluar	hari_keluar	selisih_volume	shift_penggantian
450	5	4	264	0

Gambar 4.9 Data Uji *False Negative*

Citra yang terdapat pada **Gambar 4.9** merupakan data yang memiliki label sebenarnya berupa abnormal. Akan tetapi, model *Extra Trees* mendeteksi kedua kombinasi data tersebut sebagai normal. Dari segi warna citranya, kedua data citra tersebut tergolong ke dalam dua kategori abnormal yang berbeda. Citra bagian atas termasuk pada kondisi abnormal ringan, yaitu kondisi abnormal yang memiliki perbedaan tipis dengan kondisi normal. Sedangkan, citra di bawahnya termasuk pada kondisi abnormal berat. Pada data pertama, kesalahan prediksi dapat disebabkan oleh warna pada citra yang memiliki perbedaan tipis dengan kondisi normal walaupun dari segi data klinisnya terdapat kombinasi tekanan darah dan selisih volume nol yang dapat menjadi penanda awal abnormalitas bagi pasien. Sementara itu, pada data bagian bawah kesalahan pendektsian model dapat disebabkan oleh keberhasilan model membaca *background* gambar yang berwarna dan bercorak sehingga tidak langsung dikategorikan sebagai abnormal karena terdapat warna merah. Bukan hanya itu, pendektsian normal juga dapat disebabkan oleh kondisi klinis, seperti nilai nadi, durasi drain, dan selisih volume, yang masih berada di dalam rentang normal.

(halaman ini sengaja dikosongkan)

BAB 5 Kesimpulan dan Saran

5.1 Kesimpulan

Dari penggerjaan Penelitian yang telah dilakukan, mulai dari tahap perumusan masalah, perancangan sistem, implementasi, uji coba, dan evaluasi, didapatkan hasil atau kesimpulan sebagai berikut.

1. Pengembangan model klasifikasi multimodal pada data citra *effluent dialysate* dan data klinis tabular pasien CAPD menggunakan model *machine learning* AdaBoost, *Decision Tree*, *Extra Trees*, KNN, *Logistic Regression*, *Naïve Bayes*, *Random Forest*, SVM, dan XGBoost.
2. Proses penambahan data menggunakan empat model sintetik data berbasis GAN, yaitu CTGAN, CTABGAN, CTABGAN+, dan CopulaGAN. Data sintetik yang dihasilkan kemudian digabungkan dengan data asli agar jumlah data dapat lebih bervariasi.
3. Model deteksi risiko komplikasi CAPD menggunakan pendekatan multimodal dievaluasi menggunakan tiga metrik evaluasi, yaitu *recall*, *precision*, dan *F1-score*. Model terbaik untuk klasifikasi multimodal adalah *Extra Trees* dengan nilai *recall* sebesar 0,9355, *precision* sebesar 1,0000, dan *F1-score* sebesar 0,9667 pada skenario penggunaan PCA 85% variance, CaiT sebagai *feature extractor*, *standard scaler* sebagai metode normalisasi, CTABGAN sebagai model pembuatan sintetik data, dan *modelling* menggunakan parameter *default*.

5.2 Saran

Dari hasil evaluasi selama penggerjaan Penelitian ini, terdapat beberapa saran yang dapat dilakukan pada penelitian selanjutnya.

1. Model klasifikasi risiko komplikasi pasien CAPD selanjutnya dapat ditingkatkan dengan menambah variasi data citra *effluent dialysate* dan data klinis pasien abnormal yang berkorelasi untuk meningkatkan generalisasi model sintetik data.
2. Parameter hasil uji laboratorium dapat ditambahkan untuk penelitian selanjutnya, di antaranya hemoglobin, albumin, natrium, dan kalium.
3. Perlu adanya pengintegrasian model *bag detection* pada antarmuka agar pasien dapat lebih memerhatikan hasil gambar citra *effluent dialysate* yang dipotret.
4. Setelah variasi data diperbanyak, perlu ditambahkan *feature* yang berisikan perbandingan selisih volume cairan *effluent dialysate* ketika mengeluarkan cairan saat itu dengan selisih volume ketika mengeluarkan cairan saat pengeluaran sebelumnya.
5. Untuk meningkatkan performa model, perlu dilakukan uji coba menggunakan model berbasis *deep learning* pada klasifikasi multimodal.

Beberapa hal tersebut nantinya diharapkan dapat mempermudah dokter serta tenaga medis untuk menentukan tindakan yang tepat dalam pencegahan komplikasi pada pasien CAPD.

(halaman ini sengaja dikosongkan)

DAFTAR PUSTAKA

- Alam, T. M., Shaukat, K., Khan, W. A., Hameed, I. A., Almuqren, L. A., Raza, M. A., Aslam, M., & Luo, S. (2022). An Efficient Deep Learning-Based Skin Cancer Classifier for an Imbalanced Dataset. *Diagnostics*, 12(9). <https://doi.org/10.3390/diagnostics12092115>
- Anggraini, S., & Fadila, Z. (2022). Kualitas Hidup Pasien Gagal Ginjal Kronik Dengan Dialisis Di Asia Tenggara : a Systematic Review. *Hearty*, 11(1), 77. <https://doi.org/10.32832/hearty.v11i1.7947>
- Arya, N., & Saha, S. (2022). Multi-Modal Classification for Human Breast Cancer Prognosis Prediction: Proposal of Deep-Learning Based Stacked Ensemble Model. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 19(2), 1032–1041. <https://doi.org/10.1109/TCBB.2020.3018467>
- Bai, Q., & Tang, W. (2022). Artificial intelligence in peritoneal dialysis: general overview. *Renal Failure*, 44(1), 682–687. <https://doi.org/10.1080/0886022X.2022.2064304>
- Bakasa, W., & Viriri, S. (2021). *Stacked ensemble deep learning for pancreas cancer classification using extreme gradient boosting*.
- Baowaly, M. K., Lin, C. C., Liu, C. L., & Chen, K. T. (2019). Synthesizing electronic health records using improved generative adversarial networks. *Journal of the American Medical Informatics Association*, 26(3), 228–241. <https://doi.org/10.1093/jamia/ocy142>
- Baowaly, M. K., Liu, C. L., & Chen, K. T. (2019). Realistic data synthesis using enhanced generative adversarial networks. *Proceedings - IEEE 2nd International Conference on Artificial Intelligence and Knowledge Engineering, AIKE 2019*, 289–292. <https://doi.org/10.1109/AIKE.2019.00057>
- Cahan, N., Klang, E., Marom, E. M., Soffer, S., Barash, Y., Burshtein, E., Konen, E., & Greenspan, H. (2023). Multimodal fusion models for pulmonary embolism mortality prediction. *Scientific Reports*, 13(1), 1–15. <https://doi.org/10.1038/s41598-023-34303-8>
- Dai, Z., Liu, H., Le, Q. V., & Tan, M. (2021). *CoAtNet: Marrying Convolution and Attention for All Data Sizes*. <http://arxiv.org/abs/2106.04803>
- Dewi, I. G. A. A. I. N., & Kandarini, Y. (2020). Tatalaksana peritonitis bakteri *Staphylococcus Epidermidis* pada seorang pasien dengan continuous ambulatory peritoneal dialysis (CAPD). *Intisari Sains Medis*, 11(2), 504–510. <https://doi.org/10.15562/ism.v11i2.642>
- Dietterich, T. G. (2000). *Ensemble methods in machine learning*.
- Dossin, T., & Goffin, E. (2019). When the color of peritoneal dialysis effluent can be used as a diagnostic tool. *Seminars in Dialysis*, 32(1), 72–79. <https://doi.org/10.1111/sdi.12740>
- Goldstein, M., Carrillo, M., & Ghai, S. (2013). Continuous ambulatory peritoneal dialysis-a guide to imaging appearances and complications. *Insights into Imaging*, 4(1), 85–92. <https://doi.org/10.1007/s13244-012-0203-y>
- Gutierrez-espinoza, L., Abri, F., Namin, A. S., Jones, K. S., & Sears, D. R. W. (n.d.). *Fake Reviews Detection through Ensemble Learning*. 1–8.
- Habeeb, S. M., Yamin, H., Simkova, E., Awad, H. S., Alhammadi, E. A., Eid, L. A., Lone, R., & Bitzan, M. (2023). Relapsing and refractory peritoneal dialysis peritonitis caused by *Corynebacterium amycolatum*. *Pediatric Nephrology*, 38(5), 1687–1692.

<https://doi.org/10.1007/s00467-022-05801-0>

Heiliger, L., Sekuboyina, A., Menze, B., Egger, J., Kleesiek, J., Heiliger, L., Sekuboyina, A., Menze, B., Egger, J., & Kleesiek, J. (2023). *Beyond Medical Imaging - A Review of Multimodal Deep Learning in Radiology Beyond Medical Imaging: A Review of Multimodal Deep Learning in Radiology*.

Huang, S. C., Pareek, A., Zamanian, R., Banerjee, I., & Lungren, M. P. (2020). Multimodal fusion with deep neural networks for leveraging CT imaging and electronic health record: a case-study in pulmonary embolism detection. *Scientific Reports*, 10(1), 1–9. <https://doi.org/10.1038/s41598-020-78888-w>

Information, H., & Sciences, M. (2023). *Machine Learning Approaches for Detecting Coronary Artery Disease Using Angiography Imaging: A Scoping Review*. 244–248. <https://doi.org/10.3233/SHTI230474>

Jiang, Z., Yang, J., & Liu, Y. (2021). Imbalanced Learning with Oversampling based on Classification Contribution Degree. *Advanced Theory and Simulations*, 4(5), 1–10. <https://doi.org/10.1002/adts.202100031>

Joubran, N. I., Mansour, F. A., & El-Haddad, B. N. (2020). The Case | Turbid effluent dialysis fluid in a peritoneal dialysis patient. *Kidney International*, 97(5), 1065–1066. <https://doi.org/10.1016/j.kint.2019.11.004>

Kia, A., Timsina, P., Joshi, H. N., Klang, E., Gupta, R. R., Kohli-seth, R., Mazumdar, M., & Levin, M. A. (2020). *MEWS ++ : Enhancing the Prediction of Clinical Deterioration in Admitted Patients through a Machine Learning Model*.

Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., & Guo, B. (2021). *Swin Transformer: Hierarchical Vision Transformer using Shifted Windows*. <http://arxiv.org/abs/2103.14030>

Mihalache, O., Doran, H., Mustăăea, P., Bobircă, F., Georgescu, D., Bîrligea, A., Agache, A., & Pătraăcu, T. (2018). Surgical complications of peritoneal dialysis in children. *Revista Brasileira de Medicina*, 55(9), 681–692. <https://www.scopus.com/inward/record.uri?eid=2-s2.0-2542421004&partnerID=40&md5=70a773c2ceb422d4db25b22c135126d0>

Navastara, D. A., Indriati Eka Sari, F., Faticahah, C., Maroqi Abdul Jalil, M., Thaha, M., Dwi Suryantoro, S., Putri Sulistyaning, W., & Haryati, M. R. (2023). Abnormality Detection of Effluent Dialysate Images on Continuous Ambulatory Peritoneal Dialysis Using Deep Learning. *6th International Seminar on Research of Information Technology and Intelligent Systems, ISRITI 2023 - Proceeding*, 433–438. <https://doi.org/10.1109/ISRITI60336.2023.10467296>

Noh, J., Yoo, K. D., Bae, W., Lee, J. S., Kim, K., Cho, J. H., Lee, H., Kim, D. K., Lim, C. S., Kang, S. W., Kim, Y. L., Kim, Y. S., Kim, G., & Lee, J. P. (2020). Prediction of the Mortality Risk in Peritoneal Dialysis Patients using Machine Learning Models: A Nationwide Prospective Cohort in Korea. *Scientific Reports*, 10(1), 1–11. <https://doi.org/10.1038/s41598-020-64184-0>

Sagi, O., & Rokach, L. (2018). *Ensemble learning: A survey*. January, 1–18. <https://doi.org/10.1002/widm.1249>

Siddiq, M. (2022). *Multidisciplinary Sciences and Arts Use of Machine to predict patient*

- developing a disease or condition for early diagnose International Journal of Multidisciplinary Sciences and Arts. 01(01), 13–23.*
- Skoularidou, M., & Cuesta-infante, A. (2019). *Modeling Tabular Data using Conditional GAN*. *NeurIPS*.
- Sliman, H., Megdiche, I., Alajramy, L., Taweel, A., Yangui, S., Drira, A., & Lamine, E. (2023). MedGAN based synthetic dataset generation for Uveitis pathology. *Intelligent Systems with Applications*, 18(January), 200223. <https://doi.org/10.1016/j.iswa.2023.200223>
- Supriyadi, R., Roesli, R., Leong, G. B., Hilman, L. P., & Arini, F. C. (2020). *A Simple Tenckhoff Catheter Placement Technique for Continuous Ambulatory Peritoneal Dialysis (CAPD) Using the Bandung Method*. 2020.
- Taneja, N. (2017). *Comprehensive CAPD Intervention Approaches*. 24–28. <https://doi.org/10.17140/OTLOJ-SE-1-106>
- Wang, A., Chen, H., Lin, Z., Zhao, S., Han, J., & Ding, G. (2023). CAIT: *Triple-Win Compression towards High Accuracy, Fast Inference, and Favorable Transferability For ViTs*. <http://arxiv.org/abs/2309.15755>
- Wu, J., Kong, G., Lin, Y., Chu, H., Yang, C., Shi, Y., Wang, H., & Zhang, L. (2020). Development of a scoring tool for predicting prolonged length of hospital stay in peritoneal dialysis patients through data mining. *Annals of Translational Medicine*, 8(21), 1437–1437. <https://doi.org/10.21037/atm-20-1006>
- Wu, Y. L., Lin, Y. S., Hsueh, T. Y. R., Lo, W. C., Peng, K. C., & Kao, M. J. (2018). Green dialysate and gallbladder perforation in a peritoneal dialysis patients: A case report and literature review. *BMC Nephrology*, 19(1), 1–6. <https://doi.org/10.1186/s12882-018-0974-6>
- Xiao, A., Dilmaghani, S., & Bhuiyan, M. N. (2023). 62-Year-Old Man With Abdominal Pain and Cloudy Peritoneal Dialysate. *Mayo Clinic Proceedings*, 98(9), 1386–1391. <https://doi.org/10.1016/j.mayocp.2023.01.026>
- Yaseen, M. (2024). *What is YOLOv9: An In-Depth Exploration of the Internal Features of the Next-Generation Object Detector*. <http://arxiv.org/abs/2409.07813>
- Zhao, Z., Birke, R., & Chen, L. Y. (2017). CTAB-GAN : Effective Table Data Synthesizing. In *Proceedings of KDD* (Vol. 1, Issue 1). Association for Computing Machinery.
- Zhao, Z., Kunar, A., Birke, R., Van der Scheer, H., & Chen, L. Y. (2023). CTAB-GAN+: enhancing tabular data synthesis. *Frontiers in Big Data*, 6(1), 1–13. <https://doi.org/10.3389/fdata.2023.1296508>
- Zhao, Z., Yan, Q., Li, D., Li, G., Cai, J., Pan, S., Duan, J., Liu, D., & Liu, Z. (2023). Relationship between serum iPTH and peritonitis episodes in patients undergoing continuous ambulatory peritoneal dialysis. *Frontiers in Endocrinology*, 14(March), 1–14. <https://doi.org/10.3389/fendo.2023.1081543>
- Alam, T. M., Shaukat, K., Khan, W. A., Hameed, I. A., Almuqren, L. A., Raza, M. A., Aslam, M., & Luo, S. (2022). An Efficient Deep Learning-Based Skin Cancer Classifier for an Imbalanced Dataset. *Diagnostics*, 12(9). <https://doi.org/10.3390/diagnostics12092115>
- Anggraini, S., & Fadila, Z. (2022). Kualitas Hidup Pasien Gagal Ginjal Kronik Dengan Dialisis Di Asia Tenggara : a Systematic Review. *Hearty*, 11(1), 77.

<https://doi.org/10.32832/hearty.v11i1.7947>

- Arya, N., & Saha, S. (2022). Multi-Modal Classification for Human Breast Cancer Prognosis Prediction: Proposal of Deep-Learning Based Stacked Ensemble Model. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 19(2), 1032–1041. <https://doi.org/10.1109/TCBB.2020.3018467>
- Bai, Q., & Tang, W. (2022). Artificial intelligence in peritoneal dialysis: general overview. *Renal Failure*, 44(1), 682–687. <https://doi.org/10.1080/0886022X.2022.2064304>
- Bakasa, W., & Viriri, S. (2021). *Stacked ensemble deep learning for pancreas cancer classification using extreme gradient boosting*.
- Baowaly, M. K., Lin, C. C., Liu, C. L., & Chen, K. T. (2019). Synthesizing electronic health records using improved generative adversarial networks. *Journal of the American Medical Informatics Association*, 26(3), 228–241. <https://doi.org/10.1093/jamia/ocy142>
- Baowaly, M. K., Liu, C. L., & Chen, K. T. (2019). Realistic data synthesis using enhanced generative adversarial networks. *Proceedings - IEEE 2nd International Conference on Artificial Intelligence and Knowledge Engineering, AIKE 2019*, 289–292. <https://doi.org/10.1109/AIKE.2019.00057>
- Cahan, N., Klang, E., Marom, E. M., Soffer, S., Barash, Y., Burshtein, E., Konen, E., & Greenspan, H. (2023). Multimodal fusion models for pulmonary embolism mortality prediction. *Scientific Reports*, 13(1), 1–15. <https://doi.org/10.1038/s41598-023-34303-8>
- Dai, Z., Liu, H., Le, Q. V., & Tan, M. (2021). *CoAtNet: Marrying Convolution and Attention for All Data Sizes*. <http://arxiv.org/abs/2106.04803>
- Dewi, I. G. A. A. I. N., & Kandarini, Y. (2020). Tatalaksana peritonitis bakteri *Staphylococcus Epidermidis* pada seorang pasien dengan continuous ambulatory peritoneal dialysis (CAPD). *Intisari Sains Medis*, 11(2), 504–510. <https://doi.org/10.15562/ism.v11i2.642>
- Dietterich, T. G. (2000). *Ensemble methods in machine learning*.
- Dossin, T., & Goffin, E. (2019). When the color of peritoneal dialysis effluent can be used as a diagnostic tool. *Seminars in Dialysis*, 32(1), 72–79. <https://doi.org/10.1111/sdi.12740>
- Goldstein, M., Carrillo, M., & Ghai, S. (2013). Continuous ambulatory peritoneal dialysis-a guide to imaging appearances and complications. *Insights into Imaging*, 4(1), 85–92. <https://doi.org/10.1007/s13244-012-0203-y>
- Gutierrez-espinoza, L., Abri, F., Namin, A. S., Jones, K. S., & Sears, D. R. W. (n.d.). *Fake Reviews Detection through Ensemble Learning*. 1–8.
- Habeeb, S. M., Yamin, H., Simkova, E., Awad, H. S., Alhammadi, E. A., Eid, L. A., Lone, R., & Bitzan, M. (2023). Relapsing and refractory peritoneal dialysis peritonitis caused by *Corynebacterium amycolatum*. *Pediatric Nephrology*, 38(5), 1687–1692. <https://doi.org/10.1007/s00467-022-05801-0>
- Heiliger, L., Sekuboyina, A., Menze, B., Egger, J., Kleesiek, J., Heiliger, L., Sekuboyina, A., Menze, B., Egger, J., & Kleesiek, J. (2023). *Beyond Medical Imaging - A Review of Multimodal Deep Learning in Radiology Beyond Medical Imaging: A Review of Multimodal Deep Learning in Radiology*.
- Huang, S. C., Pareek, A., Zamanian, R., Banerjee, I., & Lungren, M. P. (2020). Multimodal fusion with deep neural networks for leveraging CT imaging and electronic health record:

- a case-study in pulmonary embolism detection. *Scientific Reports*, 10(1), 1–9. <https://doi.org/10.1038/s41598-020-78888-w>
- Information, H., & Sciences, M. (2023). *Machine Learning Approaches for Detecting Coronary Artery Disease Using Angiography Imaging: A Scoping Review*. 244–248. <https://doi.org/10.3233/SHTI230474>
- Jiang, Z., Yang, J., & Liu, Y. (2021). Imbalanced Learning with Oversampling based on Classification Contribution Degree. *Advanced Theory and Simulations*, 4(5), 1–10. <https://doi.org/10.1002/adts.202100031>
- Joubran, N. I., Mansour, F. A., & El-Haddad, B. N. (2020). The Case | Turbid effluent dialysis fluid in a peritoneal dialysis patient. *Kidney International*, 97(5), 1065–1066. <https://doi.org/10.1016/j.kint.2019.11.004>
- Kia, A., Timsina, P., Joshi, H. N., Klang, E., Gupta, R. R., Kohli-seth, R., Mazumdar, M., & Levin, M. A. (2020). *MEWS ++ : Enhancing the Prediction of Clinical Deterioration in Admitted Patients through a Machine Learning Model*.
- Liu, Z., Lin, Y., Cao, Y., Hu, H., Wei, Y., Zhang, Z., Lin, S., & Guo, B. (2021). *Swin Transformer: Hierarchical Vision Transformer using Shifted Windows*. <http://arxiv.org/abs/2103.14030>
- Mihalache, O., Doran, H., Mustățea, P., Bobircă, F., Georgescu, D., Bîrligăea, A., Agache, A., & Pătrașcu, T. (2018). Surgical complications of peritoneal dialysis in children. *Revista Brasileira de Medicina*, 55(9), 681–692. <https://www.scopus.com/inward/record.uri?eid=2-s2.0-2542421004&partnerID=40&md5=70a773c2ceb422d4db25b22c135126d0>
- Navastara, D. A., Indriati Eka Sari, F., Faticahah, C., Maroqi Abdul Jalil, M., Thaha, M., Dwi Suryantoro, S., Putri Sulistyaning, W., & Haryati, M. R. (2023). Abnormality Detection of Effluent Dialysate Images on Continuous Ambulatory Peritoneal Dialysis Using Deep Learning. *6th International Seminar on Research of Information Technology and Intelligent Systems, ISRITI 2023 - Proceeding*, 433–438. <https://doi.org/10.1109/ISRITI60336.2023.10467296>
- Noh, J., Yoo, K. D., Bae, W., Lee, J. S., Kim, K., Cho, J. H., Lee, H., Kim, D. K., Lim, C. S., Kang, S. W., Kim, Y. L., Kim, Y. S., Kim, G., & Lee, J. P. (2020). Prediction of the Mortality Risk in Peritoneal Dialysis Patients using Machine Learning Models: A Nationwide Prospective Cohort in Korea. *Scientific Reports*, 10(1), 1–11. <https://doi.org/10.1038/s41598-020-64184-0>
- Sagi, O., & Rokach, L. (2018). *Ensemble learning: A survey*. January, 1–18. <https://doi.org/10.1002/widm.1249>
- Siddiq, M. (2022). *Multidisciplinary Sciences and Arts Use of Machine to predict patient developing a disease or condition for early diagnose* *International Journal of Multidisciplinary Sciences and Arts*. 01(01), 13–23.
- Skoularidou, M., & Cuesta-infante, A. (2019). *Modeling Tabular Data using Conditional GAN*. *NeurIPS*.
- Sliman, H., Megdiche, I., Alajramy, L., Taweel, A., Yangui, S., Drira, A., & Lamine, E. (2023). MedWGAN based synthetic dataset generation for Uveitis pathology. *Intelligent Systems with Applications*, 18(January), 200223. <https://doi.org/10.1016/j.iswa.2023.200223>

- Supriyadi, R., Roesli, R., Leong, G. B., Hilman, L. P., & Arini, F. C. (2020). *A Simple Tenckhoff Catheter Placement Technique for Continuous Ambulatory Peritoneal Dialysis (CAPD) Using the Bandung Method*. 2020.
- Taneja, N. (2017). *Comprehensive CAPD Intervention Approaches*. 24–28. <https://doi.org/10.17140/OTLOJ-SE-1-106>
- Wang, A., Chen, H., Lin, Z., Zhao, S., Han, J., & Ding, G. (2023). CAIT: *Triple-Win Compression towards High Accuracy, Fast Inference, and Favorable Transferability For ViTs*. <http://arxiv.org/abs/2309.15755>
- Wu, J., Kong, G., Lin, Y., Chu, H., Yang, C., Shi, Y., Wang, H., & Zhang, L. (2020). Development of a scoring tool for predicting prolonged length of hospital stay in peritoneal dialysis patients through data mining. *Annals of Translational Medicine*, 8(21), 1437–1437. <https://doi.org/10.21037/atm-20-1006>
- Wu, Y. L., Lin, Y. S., Hsueh, T. Y. R., Lo, W. C., Peng, K. C., & Kao, M. J. (2018). Green dialysate and gallbladder perforation in a peritoneal dialysis patients: A case report and literature review. *BMC Nephrology*, 19(1), 1–6. <https://doi.org/10.1186/s12882-018-0974-6>
- Xiao, A., Dilmaghani, S., & Bhuiyan, M. N. (2023). 62-Year-Old Man With Abdominal Pain and Cloudy Peritoneal Dialysate. *Mayo Clinic Proceedings*, 98(9), 1386–1391. <https://doi.org/10.1016/j.mayocp.2023.01.026>
- Yaseen, M. (2024). *What is YOLOv9: An In-Depth Exploration of the Internal Features of the Next-Generation Object Detector*. <http://arxiv.org/abs/2409.07813>
- Zhao, Z., Birke, R., & Chen, L. Y. (2017). CTAB-GAN : Effective Table Data Synthesizing. In *Proceedings of KDD* (Vol. 1, Issue 1). Association for Computing Machinery.
- Zhao, Z., Kunar, A., Birke, R., Van der Scheer, H., & Chen, L. Y. (2023). CTAB-GAN+: enhancing tabular data synthesis. *Frontiers in Big Data*, 6(1), 1–13. <https://doi.org/10.3389/fdata.2023.1296508>
- Zhao, Z., Yan, Q., Li, D., Li, G., Cai, J., Pan, S., Duan, J., Liu, D., & Liu, Z. (2023). Relationship between serum iPTH and peritonitis episodes in patients undergoing continuous ambulatory peritoneal dialysis. *Frontiers in Endocrinology*, 14(March), 1–14. <https://doi.org/10.3389/fendo.2023.1081543>

LAMPIRAN

L 1. Nilai Kualitas Data Sintetik

gan	feature-extractor	data validity score	data structure score	overall diagnostic score
ctgan	swin	98.56%	100%	99.28%
ctgan	cait	98.47%	100%	99.24%
ctgan	coatnet	98.36%	100%	99.18%
ctgan	yolov9	98.86%	100%	99.43%
ctabgan	swin	96.60%	90.29%	93.45%
ctabgan	cait	96.89%	88.76%	92.83%
ctabgan	coatnet	97.26%	96.14%	96.70%
ctabgan	yolov9	96.79%	95.59%	96.19%
ctabgan+	swin	98.98%	90.29%	94.64%
ctabgan+	cait	98.82%	88.76%	93.79%
ctabgan+	coatnet	99.60%	96.14%	97.87%
ctabgan+	yolov9	99.53%	95.59%	97.56%
copulagan	swin	99.14%	100%	99.57%
copulagan	cait	98.46%	100%	99.23%
copulagan	coatnet	98.19%	100%	99.10%
copulagan	yolov9	97.87%	100%	98.94%

gan	feature-extractor	column shapes score	column pair trends score	overall quality score
ctgan	swin	70.01%	89.16%	79.59%
ctgan	cait	68.40%	87.87%	78.14%
ctgan	coatnet	68.22%	92.34%	80.28%
ctgan	yolov9	68.65%	92.18%	80.42%
ctabgan	swin	82.43%	86.70%	84.57%
ctabgan	cait	83.62%	87.10%	85.36%
ctabgan	coatnet	83.08%	91.87%	87.48%
ctabgan	yolov9	81.97%	91.27%	86.62%
ctabgan+	swin	80.12%	88.09%	84.11%
ctabgan+	cait	81.70%	87.37%	84.54%
ctabgan+	coatnet	77.50%	92.27%	84.89%
ctabgan+	yolov9	77.29%	91.97%	84.63%
copulagan	swin	61.43%	87.69%	74.56%
copulagan	cait	61.45%	85.99%	73.72%
copulagan	coatnet	56.44%	91.27%	73.86%
copulagan	yolov9	52.92%	90.85%	71.89%

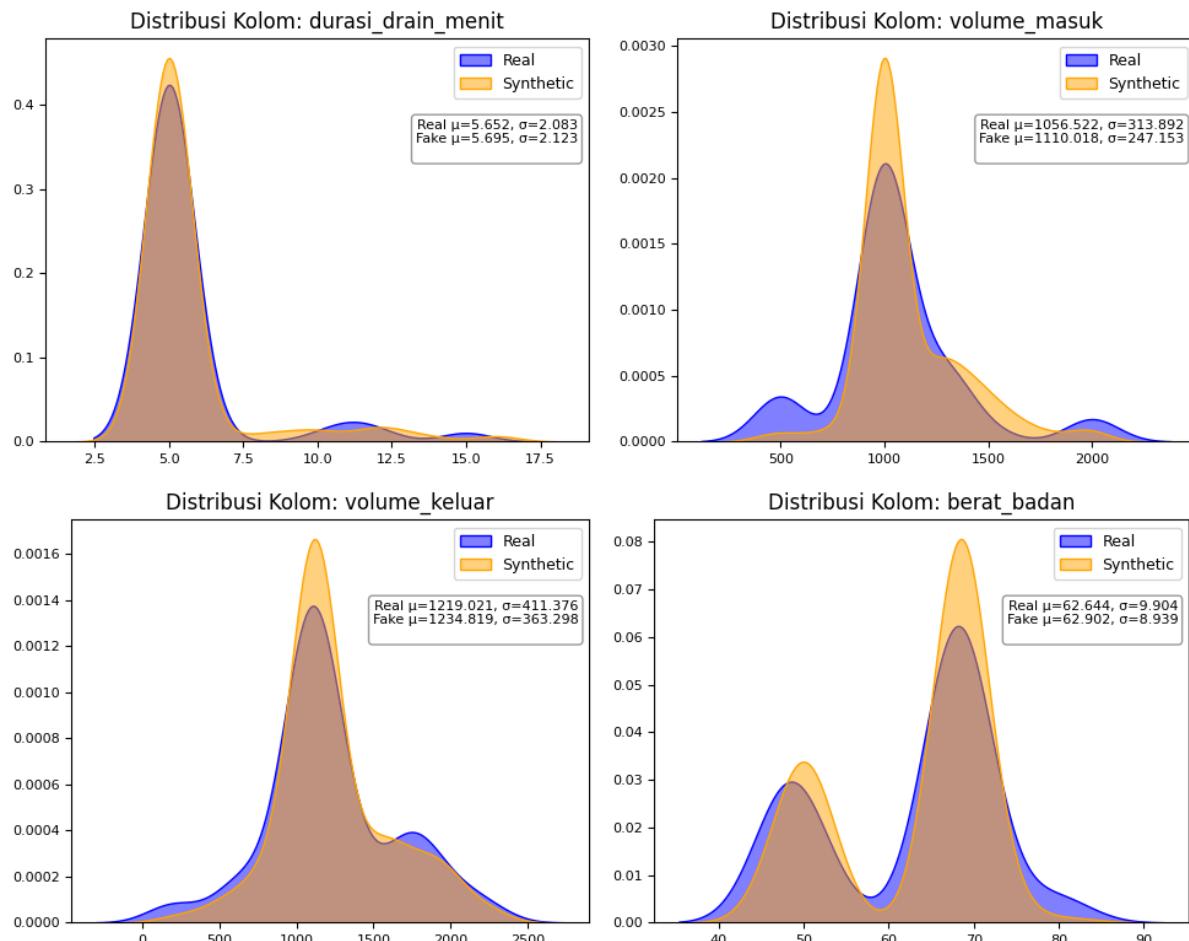
gan	feature-extractor	overall diagnostic score	overall quality score	overall score	overall gan score
ctgan	swin	99.28%	79.59%	89.43%	89.44%
ctgan	cait	99.24%	78.14%	88.69%	
ctgan	coatnet	99.18%	80.28%	89.73%	
ctgan	yolov9	99.43%	80.42%	89.92%	
ctabgan	swin	93.45%	84.57%	89.01%	90.40%
ctabgan	cait	92.83%	85.36%	89.09%	
ctabgan	coatnet	96.70%	87.48%	92.09%	
ctabgan	yolov9	96.19%	86.62%	91.41%	
ctabgan+	swin	94.64%	84.11%	89.37%	90.25%
ctabgan+	cait	93.79%	84.54%	89.16%	
ctabgan+	coatnet	97.87%	84.89%	91.38%	
ctabgan+	yolov9	97.56%	84.63%	91.10%	
copulagan	swin	99.57%	74.56%	87.07%	86.36%
copulagan	cait	99.23%	73.72%	86.48%	
copulagan	coatnet	99.10%	73.86%	86.48%	
copulagan	yolov9	98.94%	71.89%	85.41%	

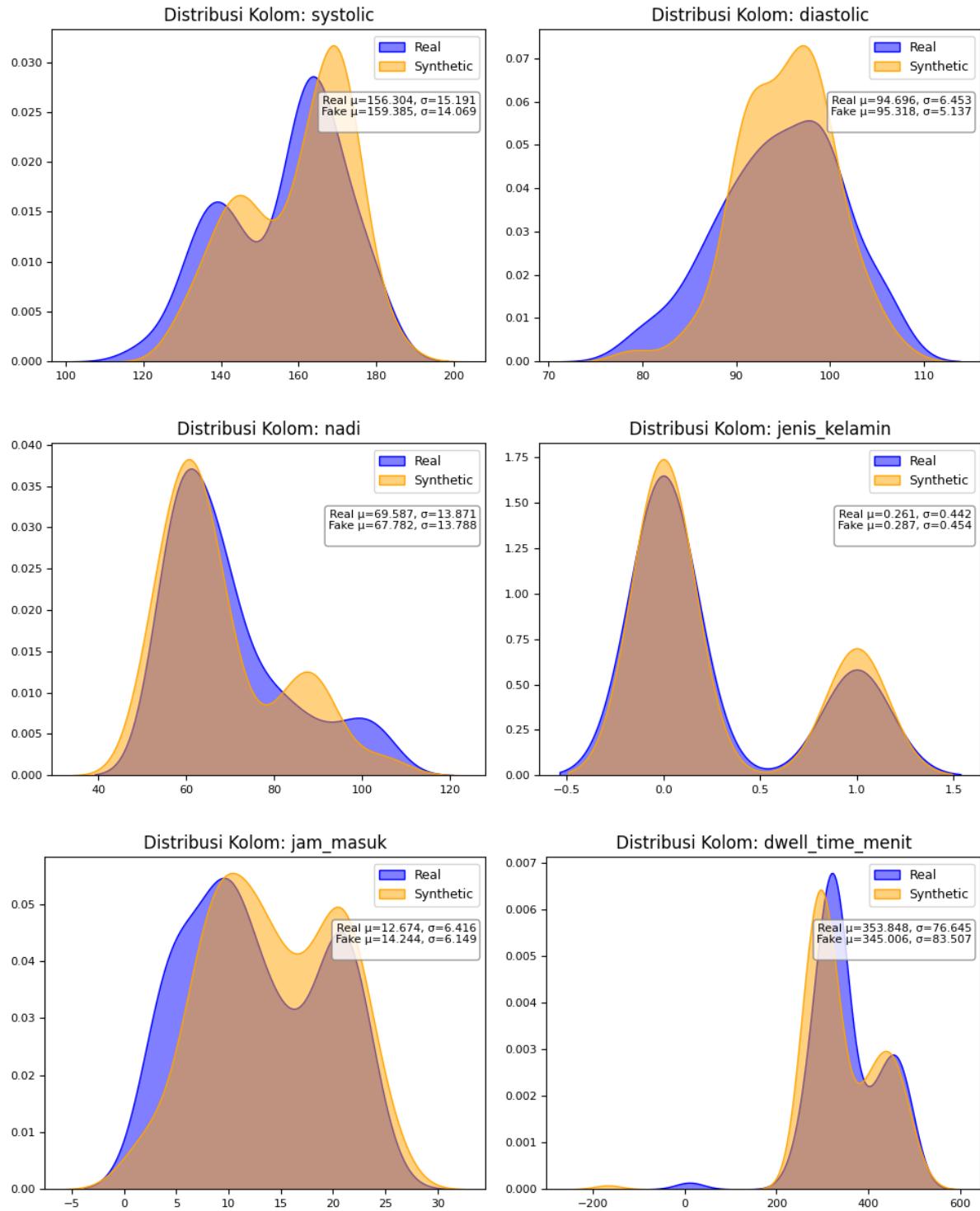
L 2. Contoh Data Sintetik untuk Data Tabular

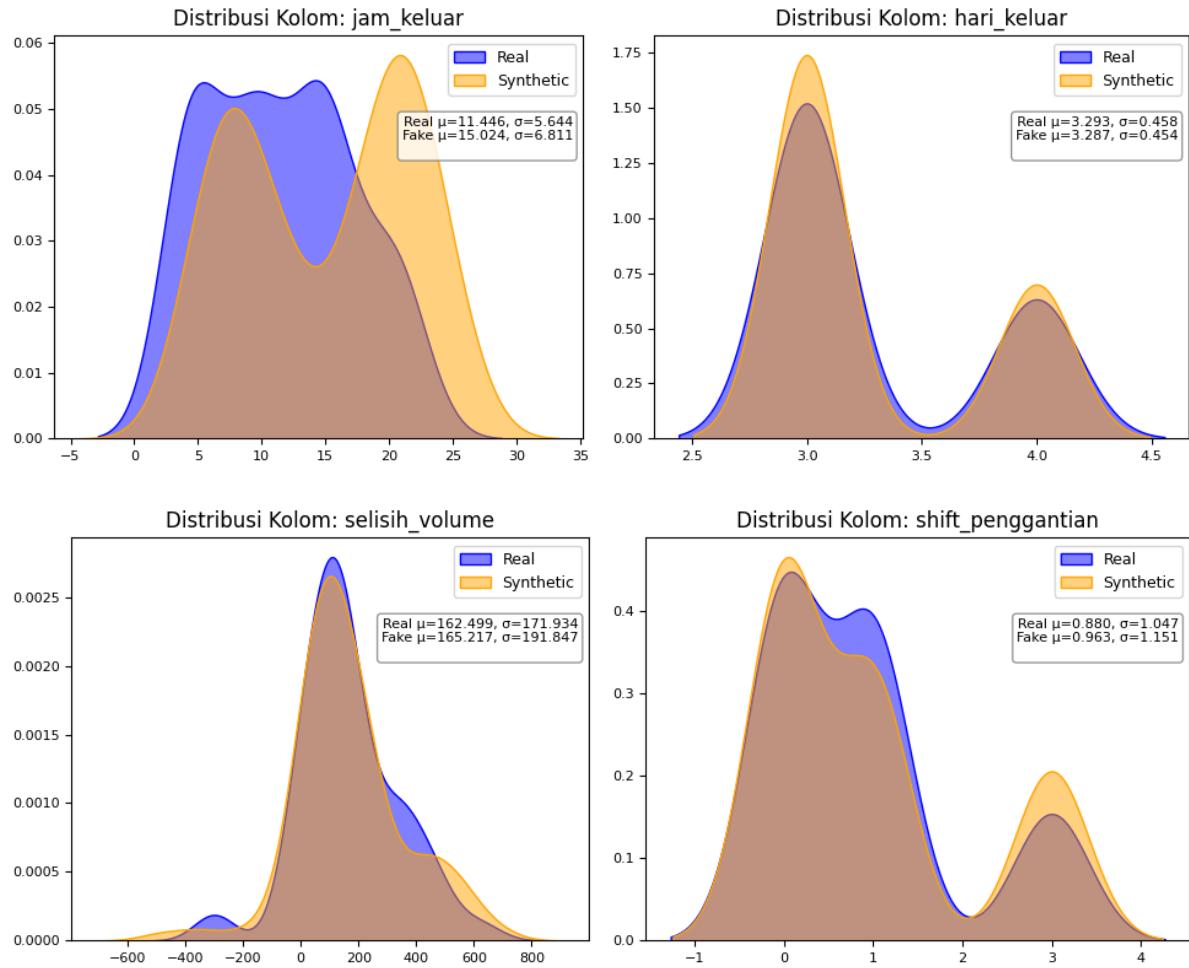
durasi_drain_menit	volume_masuk	volume_keluar	berat_badan	systolic
5	985.0000	726.6054	68.9070	173.4453517
diastolic	nadi	jenis_kelamin	jam_masuk	hari_masuk
99.1101	53.8694	1.0000	20	3
dwell_time_menit	jam_keluar	hari_keluar	selisih_volume	shift_penggantian
-167	8	4	74.73080344	3

durasi_drain_menit	volume_masuk	volume_keluar	berat_badan	systolic
5	1059.0000	1241.4244	69.3181	167.4972348
diastolic	nadi	jenis_kelamin	jam_masuk	hari_masuk
100.5968	59.5189	0.0000	20	3
dwell_time_menit	jam_keluar	hari_keluar	selisih_volume	shift_penggantian
389	6	4	231.4325882	0

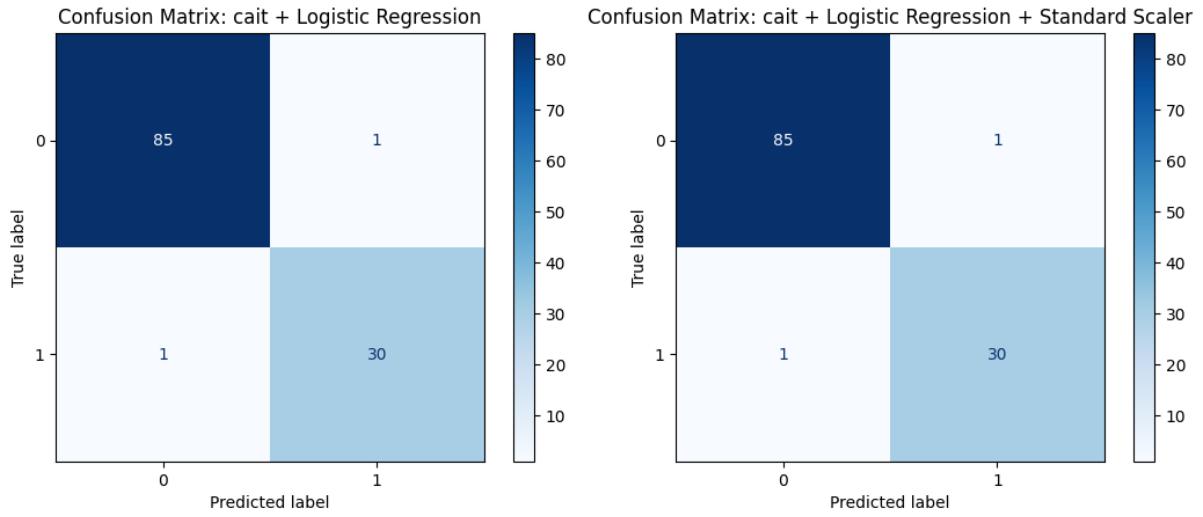
L 3. Perbandingan Distribusi Data Asli dan Data Sintetik







L 4. Confusion Matrix Hasil Terbaik pada Uji Coba *feature extractor* dan normalisasi



L 5. Hasil *Modelling* Penelitian Sebelumnya untuk Skema Hanya Data Citra

Model	Recall	Precision	Accuracy	F1-Score
MobileNetV3	0.8571	1	0.9714	0.9232
ConvNeXTv2	0.7143	1	0.9429	0.8333
MobileViTv2	0.8571	1	0.9714	0.9232
YOLOv8	0.8571	1	0.9714	0.9232

L 6. Hasil *Modelling* Penelitian Ini untuk Skema Hanya Data Citra

Model	Recall	Precision	Accuracy	F1-Score
CaiT	1	1	1	1
CoAtNet	0.7778	1	0.9394	0.875
Swin Transformer	1	1	1	1
YOLOv9	Belum memiliki kemampuan resmi untuk klasifikasi gambar			

(halaman ini sengaja dikosongkan)

BIODATA PENULIS



Penulis dilahirkan di Cirebon, 16 Agustus 2001, merupakan anak ketiga dari 3 bersaudara. Penulis telah menempuh pendidikan formal yaitu di TK Az-ziyadah Kota Cirebon, MI Salafiyah Kota Cirebon, MTSNU Kabupaten Cirebon, dan MAN 3 Kabupaten Cirebon. Setelah lulus dari MAN tahun 2020, Penulis mengikuti SBMPTN dan diterima pada tahun kedua di Departemen Teknik Informatika FTEIC - ITS pada tahun 2021 dan terdaftar dengan NRP 5025211103,

Di Departemen Teknik Informatika Penulis sempat aktif di beberapa kegiatan Seminar yang diselenggarakan oleh Departemen, Himpunan Mahasiswa Teknik Computer-Informatika (HMTC).