KLASIFIKASI GEN ESENSIAL PADA *DROSOPHILA MELANOGASTER* DENGAN METODE *ADABOOST* DAN *XGBOOST* MENGGUNAKAN DATA *SEQUENCE* DNA DAN PROTEIN

Tesis

Oleh :
GENDIS ANANDA PUTRI
NPM 2427051004

PROGRAM STUDI S2 ILMU KOMPUTER
JURUSAN ILMU KOMPUTER
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
2026

KLASIFIKASI GEN ESENSIAL PADA *DROSOPHILA MELANOGASTER* DENGAN METODE *ADABOOST* DAN *XGBOOST* MENGGUNAKAN DATA *SEQUENCE* DNA DAN PROTEIN

Oleh

GENDIS ANANDA PUTRI

Tesis

Sebagai Salah Satu Syarat untuk Mendapat Gelar
MAGISTER KOMPUTER

Pada

Jurusan Ilmu Komputer
Fakultas Matematika dan Ilmu Pengetahuan Alam

PROGRAM STUDI S2 ILMU KOMPUTER
JURUSAN ILMU KOMPUTER
FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM
UNIVERSITAS LAMPUNG
2026

ABSTRAK

KLASIFIKASI GEN ESENSIAL PADA *DROSOPHILA MELANOGASTER* DENGAN METODE *ADABOOST* DAN *XGBOOST* MENGGUNAKAN DATA *SEQUENCE* DNA DAN PROTEIN

Oleh

Gendis Ananda Putri

Gen esensial sangat penting untuk kelangsungan hidup dan fungsi normal suatu organisme. Studi ini bertujuan membandingkan kinerja algoritma *AdaBoost* dan *XGBoost* dalam mengklasifikasikan *Gen Esensial Seluler* (CEG) dan *Gen Esensial Organisme* (OEG) pada *Drosophila melanogaster*. Dataset diambil dari dataset CLEARER yang telah dikurasi oleh Beder et al. (2021), terdiri dari 11.547 gen untuk label CEG setelah proses pra-pemrosesan. Fitur diekstraksi menggunakan Komposisi Asam Amino (AAC) dari urutan protein, Komposisi *Tri-Nukleotida* (TNC) dan *Transformasi Fourier* (FT) dari urutan DNA, serta *PPI_degree*, sehingga menghasilkan 185 fitur awal. Sebanyak 45 fitur teratas dipilih menggunakan *Random Forest Gini Importance*. Untuk menangani ketidakseimbangan kelas yang parah (rasio 1:8,41 untuk CEG), teknik SMOTETomek diterapkan hanya pada data pelatihan setelah dilakukan pembagian stratifikasi 90:10. Model dilatih dengan validasi silang *10-fold* dan optimasi *threshold probabilitas*. Hasil menunjukkan bahwa *XGBoost* dengan SMOTETomek mencapai performa terbaik pada label CEG: akurasi 96,88%, skor F1 0,9498, MCC 0,9400, ROC-AUC 0,9888, dan PR-AUC 0,9869. Pada label OEG, *XGBoost* mencapai akurasi 94,98%. Sebagai perbandingan, AdaBoost memperoleh akurasi 85,00% pada CEG dan 84,15% pada OEG. Kesalahan klasifikasi umumnya terjadi akibat kemiripan pola urutan antara gen esensial dan non-esensial. Studi ini menunjukkan bahwa kombinasi *XGBoost* dengan SMOTETomek dan seleksi fitur berbasis *Gini Importance* secara efektif mengatasi ketidakseimbangan kelas dan memberikan performa yang kompetitif untuk klasifikasi gen esensial berbasis urutan pada *Drosophila melanogaster*.

Kata Kunci: Gen esensial, *AdaBoost*, *XGBoost*, *Drosophila melanogaster*, SMOTETomek

ABSTRACT

CLASSIFICATION OF ESSENTIAL GENES IN *DROSOPHILA MELANOGASTER* USING *ADABOOST* AND *XGBOOST* METHODS BASED ON DNA AND PROTEIN SEQUENCE DATA

By

Gendis Ananda Putri

Essential genes are critical for the survival and normal function of an organism. This study aimed to compare the performance of AdaBoost and XGBoost algorithms in classifying Cellular Essential Genes (CEG) and Organismal Essential Genes (OEG) in Drosophila melanogaster. The dataset was obtained from the curated CLEARER dataset by Beder et al. (2021), which, after preprocessing, consists of 11,547 genes for the CEG label. Features were extracted from protein sequences using Amino Acid Composition (AAC), from DNA sequences using Tri-Nucleotide Composition (TNC) and Fourier Transform (FT), and from PPI_degree, resulting in 185 initial features. The top 45 features were selected using Random Forest Gini Importance. To handle severe class imbalance (ratio 1:8.41 for CEG), the SMOTETomek technique was applied only to the training set after a 90:10 stratified split. Models were trained with 10-fold cross-validation and an optimized probability threshold. Results showed that XGBoost with SMOTETomek achieved the best performance on the CEG label: 96.88% accuracy, 0.9498 F1-score, 0.9400 MCC, 0.9888 ROC-AUC, and 0.9869 PR-AUC. On the OEG label, XGBoost reached 94.98% accuracy. In comparison, AdaBoost obtained 85.00% accuracy on CEG and 84.15% on OEG. Misclassifications mainly occurred due to sequence pattern similarities between essential and non-essential genes. This study demonstrates that combining XGBoost with SMOTETomek and Gini Importance-based feature selection effectively addresses class imbalance and provides competitive performance for sequence-based essential gene classification in Drosophila melanogaster.

Keywords: Essential genes, *AdaBoost*, *XGBoost*, *Drosophila melanogaster*, SMOTETomek

Judul Tesis	:	**KLASIFIKASI GEN ESENSIAL PADA *DROSOPHILA MELANOGASTER* DENGAN METODE *ADABOOST DAN XGBOOST* MENGGUNAKAN DATA SEQUENCE DNA DAN PROTEIN**
Nama Mahasiswa	:	*Gendis Ananda Putri*
Nomor Pokok Mahasiswa	:	2427051004
Program Studi	:	S2 Ilmu Komputer
Jurusan	:	Ilmu Komputer
Fakultas	:	Matematika dan Ilmu Pengetahuan Alam

Bandar Lampung, 11 Mei 2026

MENYETUJUI

Komisi Pembimbing

1. **Favorisen R. Lumbanraja, Ph.D.**
   NIP 19830110 200812 1 002

2. **Dr. rer. nat. Akmal Junaidi, M.Sc.**
   NIP 19710129 199702 1 001

Ketua Jurusan Ilmu Komputer

Dwi Sakethi, S. Si., M. Kom.
NIP 19680611 199802 1 001

Ketua Program Studi

Favorisen R. Lumbanraja, Ph.D.
NIP 19830110 200812 1 002

MENGESAHKAN

1. Tim Penguji

Ketua : **Favorisen R. Lumbanraja, Ph.D.** .....

Sekretaris : **Dr. Rer. nat. Akmal Junaidi, M. Sc.** .....

Penguji
Bukan Pembimbing : **1. Dr. Aristoteles, S. Si., M. Si.** .....

2. Tristiyanto, S. Kom., M.I.S., Ph.D. .....

2. Dekan Fakultas Matematika dan Ilmu Pengetahuan Alam

Dr. Eng. Heri Satria, S. Si., M. Si.
NIP 19711001 200501 1 002

3. Direktur Program Pascasarjana

Prof. Dr. Ir. Murhadi, M. Si.
NIP 19640326 198902 1 001

Tanggal Lulus Ujian Tesis : 5 Mei 2026

PERNYATAAN

Saya yang bertanda tangan di bawah ini :

Nama : Gendis Ananda Putri

NPM : 2427051004

Dengan ini menyatakan bahwa Tesis saya yang berjudul **"Klasifikasi Gen Esensial Pada *Drosophila Melanogaster* dengan Metode *AdaBoost* dan *XGBoost* Menggunakan Data *Squence* DNA dan Protein"** merupakan karya saya sendiri dan bukan karya orang lain. Semua tulisan yang tertuang dalam tesis ini telah mengikuti kaidah penulisan karya ilmiah Universitas Lampung. Apabila di kemudian hari terbukti tesis saya merupakan hasil penjiplakan atau dibuat orang lain maka bersedia menerima sanksi berupa pencabutan gelar yang telah saya terima.

Bandar Lampung, 11 Mei 2026

Gendis Ananda Putri

NPM. 2427051004

RIWAYAT HIDUP

Penulis dilahirkan di Bandar Lampung, pada tanggal 03 Juni 2001 sebagai anak pertama dari tiga bersaudara. Anak dari pasangan, Bapak Sukoco Siswa Saputra dan Ibu Kristina Rusmala Dewi.

Penulis menyelesaikan pendidikan formal di SDN 1 Labuhan Dalam Bandar Lampung dan selesai pada Tahun 2013. Kemudian pendidikan menengah pertama di SMPN 19 Bandar Lampung diselesaikan pada Tahun 2016, lalu melanjutkan ke pendidikan menengah atas di SMAN 10 Bandar Lampung yang diselesaikan pada Tahun 2019.

Pada tahun 2019 penulis terdaftar sebagai mahasiswa Prodi Pendidikan Teknologi Informasi Jurusan PMIPA Fakultas Keguruan dan Ilmu Pendidikan Universitas Lampung melalui jalur SMMPTN Barat dan lulus pada September 2023. Penulis melanjutkan pendidikan ke jenjang selanjutnya yaitu Magister Ilmu Komputer pada Tahun 2024 di Fakultas Matematika dan Ilmu Pengetahuan Alam Universitas Lampung. Selama menjadi mahasiswi penulis juga terdaftar sebagai :

1. Penerima Beasiswa Pascasarjana Universitas Lampung selama perkuliahan pada Tahun 2024 sampai 2026.
2. Mahasiswa PPG Dalam Jabatan Guru Tertentu Tahap 3 di Universitas Negeri Manado pada Tahun 2025.
3. Ketua Bidang Eksternal KOHATI HMI Cabang Bandar Lampung pada Tahun 2025 sampai 2026.
4. Guru mata pelajaran Informatika di SMA YP Unila Bandar Lampung pada Tahun 2023 sampai sekarang.

MOTTO

1. "Kesulitan itu pasti, tetapi kemudahan akan datang setelahnya. Bersabarlah, karena Allah bersama orang-orang yang sabar."
(QS. Al-Baqarah: 155)

2. "Ilmu adalah kehidupan hati dari kebutaan, cahaya mata dari kegelapan, dan kekuatan tubuh dari kelemahan."
(Imam Al-Ghazali)

3. “Hidup yang tidak dipertaruhkan tidak akan pernah dimenangkan.”
(B.J. Habibie)

4. “Cara terbaik untuk memprediksi masa depan adalah dengan menciptakannya.”
(Alan Kay - Ilmuwan Komputer)

5. "The expert in anything was once a beginner."
(Helen Hayes)

6. "Tesis ini adalah maraton terakhirmu sebagai mahasiswa. Fokus, konsisten dan finish strong."
(Anonim)

PERSEMBAHAN

Alhamdulillahirabbilalamin

Puji dan syukur tercurahkan kepada Allah Subhanahu Wa Ta'alaa atas segala Rahmat dan Karunia-Nya sehingga saya dapat menyelesaikan tesis ini. Shalawat serta salam selalu tercurahkan kepada Nabi Muhammad SAW.

Kupersembahkan karya ini kepada:

Kedua Orang Tuaku Tercinta

Yang senantiasa memberikan yang terbaik, dan melantunkan do'a yang selalu menyertaiku. Kuucapkan pula terima kasih sebesar-besarnya karena telah mendidik dan membesarkanku dengan cara yang dipenuhi kasih sayang, dukungan, dan pengorbanan yang belum bisa terbalaskan.

Adik-adikku Tercinta

Yang selalu memberikan motivasi dan saran kapanpun dan bagaimanapun Keadaan serta selalu memberikan dukungan dalam bentuk apapun yang sangat teramat berarti.

Seluruh Keluarga Besar Magister Ilmu Komputer 2024

Yang selalu memberikan semangat dan dukungan.

Almamater Tercinta, Jurusan Ilmu Komputer dan Universitas Lampung

Tempat bernaung mengemban semua ilmu untuk menjadi bekal hidup.

SANWACANA

Puji syukur kehadirat Allah SWT atas berkah, rahmat dan hidayat-Nya, petunjuk dan pedoman dari Rasulullah Nabi Muhammad Sholallahu Alaihi Wasallam penulis dapat menyelesaikan Tesis "Klasifikasi Gen Esensial Pada *Drosophila Melanogaster* dengan Metode *AdaBoost* dan *XGBoost* Menggunakan Data *Squence* DNA dan Protein"

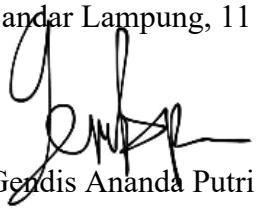
Terima kasih penulis ucapkan kepada semua pihak yang telah membantu dan berperan besar dalam menyusun Tesis ini antara lain :

1. Kedua orang tua serta adik adik tercinta yang selalu memberi dukungan, do'a, semangat, motivasi dan kasih sayang yang luar biasa tak terhingga. Semua yang telah kalian berikan tidak akan pernah mampu untukku balas. Semoga Allah SWT selalu memberikan kebahagiaan dan keberkahan dalam kehidupan kalian di dunia dan akhirat. Aamiin.
2. Bapak Dr. Eng. Heri Satria, S. Si., M. Si. selaku Dekan FMIPA Universitas Lampung.
3. Bapak Prof. Dr. Ir. Murhadi, M. Si. selaku Direktur Pasca Sarjana Universitas Lampung.
4. Bapak Dwi Sakethi, S. Si., M. Kom. selaku Ketua Jurusan Ilmu Komputer FMIPA Universitas Lampung.
5. Bapak Favorisen R. Lumbanraja, Ph.D. sebagai Ketua Program Studi S2 Ilmu Komputer sekaligus Dosen Pembimbing Utama yang telah memberikan dukungan, arahan, ide, motivasi, kritik dan saran kepada penulis sehingga dapat menyelesaikan Tesis ini dengan baik.
6. Bapak Dr. rer. nat. Akmal Junaidi, M. Sc. sebagai Pembimbing Kedua yang juga selalu dapat memberikan waktu untuk membimbing penulis dalam memberikan ide, kritik serta saran dalam menyelesaikan Tesis ini.
7. Bapak Dr. Aristoteles, S. Si., M. Si. sebagai Penguji Utama yang telah memberikan masukan yang bermanfaat dalam perbaikan Tesis ini.
8. Bapak Tristiyanto, S. Kom., M.I.S., Ph.D. sebagai Penguji Kedua yang telah memberikan masukan yang bermanfaat dalam perbaikan Tesis ini.

9. Bapak Dr. Ahmad Faisol, S. Si., M. Si. sebagai Pembimbing Akademik penulis yang telah memberikan dukungan, arahan dan motivasi kepada penulis.
10. Ibu Yunda Heningtyas, S. Kom., M. Kom. selaku Sekretaris Jurusan Ilmu Komputer FMIPA Universitas Lampung.
11. Ibu Ade Nora Maela, Bang Zainudin dan Pak Dahud yang telah membantu segala urusan administrasi penulis di Jurusan Ilmu Komputer.
12. Bapak dan Ibu Dosen Jurusan Ilmu Komputer FMIPA Universitas Lampung yang telah memberikan ilmu dan pengalaman dalam hidup untuk menjadi lebih baik.
13. Ketua Dewan Pengurus YP Unila Bapak Dr. Ryzal Perdana, M. Pd., Kepala SMA YP Unila Ibu Dra. Hj. Mey Sriyani, M. M. serta Waka Kurikulum Bapak Saiful Imam Ali Nurdin, M. Pd. yang selalu memberi dukungan dan motivasi kepada penulis selama perkuliahan dan penyelesaian Tesis.
14. M. Tegar Mulia Pratama yang senantiasa memberikan dukungan dan semangat selama penulis menempuh pendidikan Magister. Kehadirannya menjadi kekuatan yang berarti dalam setiap langkah dan proses yang penulis jalani.
15. Sahabat serta teman penulis Icha, umi Nara, Maretia, mba Aang, Khansa, mba Hanif, Mba Helda, mba Selly, mba Alya, mba Noy, 6 sist geng dan bataxx fams yang selalu memberikan semangat dan apresiasi positif kepada penulis.
16. Kanda, Yunda, dan Dinda semua baik di HMI Komisariat KIP Unila maupun Pengurus Kohati HMI Cabang Bandar Lampung.
17. Keluarga Magister Ilmu Komputer 2024 yang tidak bisa disebutkan satu persatu namanya oleh penulis yang telah memberi arti dan warna serta pengalaman tak ternilai kepada penulis selama duduk di bangku kuliah.

Penulis menyadari bahwa Tesis ini masih jauh dari kata sempurna, semoga Tesis ini membawa manfaat dan keberkahan bagi semua civitas Ilmu Komputer Universitas Lampung. Aamiin.

Bandar Lampung, 11 Mei 2026



Gendis Ananda Putri

NPM. 2427051004

DAFTAR ISI

Halaman

DAFTAR TABEL

Tabel **Halaman**

DAFTAR PSEUDOCODE

Pseudocode **Halaman**

DAFTAR GAMBAR

Gambar **Halaman**

I. PENDAHULUAN

1.1. Latar Belakang

Gen esensial merupakan kumpulan gen yang sangat vital karena keberadaannya menentukan kelangsungan hidup dan fungsi biologis normal suatu organisme. Gen-gen ini mengendalikan berbagai proses penting, seperti replikasi DNA, sintesis protein, perbaikan sel, metabolisme energi, dan regulasi siklus sel. Mutasi atau penghapusan gen esensial dapat mengakibatkan terganggunya mekanisme seluler, kelainan perkembangan, hingga kematian organisme (Li et al., 2010). Pemahaman mendalam mengenai gen esensial tidak hanya penting dalam biologi dasar, tetapi juga memiliki implikasi luas dalam bidang medis, farmasi, dan bioteknologi. Identifikasi gen esensial, misalnya, telah dimanfaatkan untuk menemukan target terapi gen, pengembangan vaksin, hingga rekayasa genetika pada organisme model (Campos et al., 2020).

Dalam bidang medis, pengetahuan tentang gen esensial dapat membantu dalam mengidentifikasi target molekuler untuk terapi berbasis gen dan pengembangan obat-obatan baru. Misalnya, beberapa terapi kanker modern memanfaatkan informasi mengenai gen esensial yang mengatur proliferasi sel abnormal. Dalam bioteknologi, pengetahuan ini juga digunakan untuk memodifikasi organisme mikroba agar lebih efisien dalam menghasilkan enzim industri atau bahan kimia bernilai tinggi. Dengan demikian, penelitian terkait identifikasi gen esensial memiliki signifikansi yang sangat besar baik dalam ranah akademis maupun dalam aplikasi praktis.

Namun, proses identifikasi gen esensial secara eksperimental masih menghadapi berbagai kendala. Metode laboratorium seperti *gene knockout*, *RNA interference* (RNAi), maupun *transposon mutagenesis* memerlukan biaya besar, waktu yang panjang, serta keahlian teknis yang tinggi (Plaimas et al., 2010). Selain itu, pendekatan tersebut tidak dapat dilakukan secara luas pada semua organisme, khususnya organisme yang kompleks atau sulit dibudidayakan. Tantangan ini mendorong berkembangnya pendekatan

komputasional berbasis *machine learning* (ML) yang mampu memanfaatkan data biologis secara lebih efisien untuk memprediksi gen esensial dengan akurasi tinggi, tanpa bergantung sepenuhnya pada eksperimen laboratorium yang mahal dan memakan waktu (Nandi et al., 2020).

Salah satu organisme model yang paling banyak digunakan dalam penelitian genetika adalah *Drosophila melanogaster* atau lalat buah. Organisme ini memiliki keunggulan biologis seperti ukuran genom yang relatif kecil namun informatif, siklus hidup yang singkat, biaya pemeliharaan rendah, serta ketersediaan data genom dan protein yang melimpah (Adams et al., 2000). Sejak lebih dari satu abad yang lalu, *Drosophila* telah digunakan untuk mempelajari mekanisme pewarisan sifat, mutasi, dan regulasi ekspresi gen, menjadikannya salah satu organisme model paling penting dalam biologi molekuler.

Selain itu, tingkat konservasi genetik yang tinggi antara *Drosophila* dan manusia memungkinkan banyak hasil penelitian pada organisme ini untuk diekstrapolasi ke sistem biologis manusia. Hal ini menjadikan *Drosophila melanogaster* sebagai model ideal dalam riset bioinformatika modern yang berfokus pada pemahaman fungsi gen dan prediksi gen esensial. Basis data seperti FlyBase menyediakan informasi DNA, protein, anotasi gen, dan interaksi molekuler yang lengkap dan terintegrasi (Larkin et al., 2021). Ketersediaan data yang komprehensif ini memungkinkan peneliti untuk membangun model prediksi yang akurat, mengurangi ketergantungan pada metode eksperimental yang mahal dan memakan waktu.

Pendekatan berbasis data semakin penting dalam konteks *big data* biologis. Penelitian yang dilakukan oleh Aromolaran et al. (2020) menunjukkan bahwa penggunaan data genom dan proteom yang terintegrasi dapat meningkatkan akurasi prediksi gen esensial pada *Drosophila melanogaster*. Penelitian ini memanfaatkan data anotasi gen, ekspresi protein, dan interaksi molekuler untuk menghasilkan model prediksi yang lebih komprehensif. Keberhasilan penelitian ini menegaskan bahwa penggunaan data multi-sumber, seperti data sekuens DNA dan protein, dapat

meningkatkan pemahaman mengenai mekanisme biologis yang mendasari fungsi gen esensial.

Dalam beberapa tahun terakhir, pendekatan *machine learning* (ML) telah banyak digunakan untuk memprediksi gen esensial pada berbagai organisme, termasuk *Drosophila melanogaster* (Aromolaran et al., 2020). Model ML mampu mengekstraksi pola kompleks dari data sekuens DNA dan protein, sehingga dapat menghasilkan prediksi dengan tingkat akurasi tinggi tanpa perlu melakukan eksperimen laboratorium yang mahal. Misalnya, penelitian oleh Wang et al. (2024) mengusulkan penggunaan graph convolutional neural network (GCN) untuk mengidentifikasi gen esensial berdasarkan representasi fitur sekuens dan interaksi jaringan dan menunjukkan hasil prediksi yang menjanjikan. Pendekatan ini menegaskan potensi besar metode komputasional untuk mengisi celah yang ditinggalkan oleh metode eksperimen konvensional.

Di antara berbagai algoritma *machine learning*, metode boosting menjadi salah satu pendekatan yang paling banyak digunakan karena kemampuannya meningkatkan akurasi prediksi melalui penggabungan model lemah menjadi model yang lebih kuat. Konsep boosting memanfaatkan iterasi bertahap untuk memperbaiki kesalahan klasifikasi pada setiap tahap pelatihan, sehingga menghasilkan model akhir yang lebih stabil dan memiliki performa prediksi yang lebih baik.

Salah satu metode boosting yang paling awal dan masih relevan hingga kini adalah *AdaBoost*, yang diperkenalkan oleh Freund dan Schapire (1997). *AdaBoost* dikenal sebagai algoritma yang sederhana, efisien, dan efektif pada berbagai masalah klasifikasi, termasuk analisis data biologis. Penerapan *AdaBoost* pada domain bioinformatika terbukti mampu menghasilkan model yang kompetitif. Sebagai contoh, penelitian Liu et al. (2020) mengembangkan model prediksi interaksi protein menggunakan *AdaBoost* yang menunjukkan akurasi tinggi. Studi lain oleh Liu et al. (2023), juga melaporkan keberhasilan penggunaan *AdaBoost* dalam mengidentifikasi gen kunci pada kanker hati dengan performa prediksi yang memuaskan.

Temuan-temuan ini menegaskan bahwa *AdaBoost* memiliki potensi besar untuk digunakan dalam klasifikasi gen esensial berbasis data biologis.

Selain *AdaBoost*, algoritma *Extreme Gradient Boosting (XGBoost)* yang diperkenalkan oleh Chen dan Guestrin (2016) menjadi salah satu metode *boosting* paling populer di berbagai kompetisi sains data dan penelitian ilmiah. *XGBoost* menawarkan berbagai keunggulan, termasuk regularisasi untuk mengatasi *overfitting*, efisiensi komputasi, serta dukungan paralel yang memudahkan pengolahan dataset berukuran besar. Dalam domain biologis, *XGBoost* telah diaplikasikan secara luas, termasuk untuk prediksi gen esensial, analisis ekspresi gen, dan prediksi struktur biomolekul (Zhou et al., 2023). Kombinasi kemampuan menangani data kompleks dan performa tinggi membuat *XGBoost* menjadi kandidat kuat untuk tugas klasifikasi gen esensial berbasis data sekuens DNA dan protein.

Pendekatan interpretabilitas model yang dikembangkan pada platform berbasis *XGBoost*, seperti *m5C-Pred* yang dilakukan oleh Zhou et al. (2023) dan *XGEM* yang dilakukan oleh Wang et al. (2022), juga memberikan keunggulan tambahan. Model-model ini tidak hanya mampu menghasilkan prediksi akurat, tetapi juga menyediakan wawasan tentang fitur mana yang paling berkontribusi terhadap prediksi. Hal ini sangat penting dalam konteks penelitian biologis, karena memungkinkan peneliti untuk menghubungkan hasil prediksi dengan mekanisme biologis yang mendasarinya.

Meskipun demikian, studi yang secara langsung membandingkan akurasi *AdaBoost* dan *XGBoost* untuk klasifikasi gen esensial, khususnya pada *Drosophila melanogaster*, masih sangat terbatas. Penelitian sebelumnya banyak berfokus pada metode lain seperti *CatBoost* yang telah dilakukan oleh Azhari (2025) atau pendekatan *deep learning* berbasis GCN yang dilakukan oleh Wang et al. (2024), namun belum mengeksplorasi secara mendalam potensi *head-to-head* antara dua algoritma boosting ini. Padahal, hasil perbandingan tersebut dapat memberikan wawasan penting untuk menentukan algoritma paling sesuai bagi analisis gen esensial di domain bioinformatika.

Berdasarkan kondisi tersebut, penelitian ini dilakukan untuk mengkaji performa *AdaBoost* dan *XGBoost* dalam mengklasifikasikan gen esensial pada *Drosophila melanogaster* menggunakan data sekuens DNA dan protein. Dataset yang digunakan dalam penelitian ini bersumber dari CLEARER (*CLassifier of Essentiality AcRoss EukaRyotes*) yang pertama kali dikembangkan oleh Beder et al. (2021) dan dipublikasikan di NAR *Genomics and Bioinformatics*. Dataset ini dibangun sebagai *gold standard* terintegrasi dengan menggabungkan label esensialitas dari dua database utama, yaitu OGEE dan DEG, serta sekuens DNA dan protein dari *FlyBase*. Untuk *Drosophila melanogaster*, label *Cellular Essential Genes* (CEG) diambil dari skrining RNAi pada kultur sel (Boutros et al.), sedangkan *Organismal Essential Genes* (OEG) diambil dari skrining *whole-organism* (seperti *P-element insertion* dan CRISPR).

Proses kurasi melibatkan penghapusan label konflik melalui *majority voting*, menghasilkan 11.547 gen setelah *preprocessing* yang siap untuk analisis komputasional. Dataset ini dipilih karena merupakan referensi komprehensif, terkurasi ketat, *open access* (via GitHub/Zenodo), dan dirancang khusus untuk machine learning, termasuk dukungan *leave-one-organism-out cross-validation* serta pembedaan jelas antara CEG dan OEG, sehingga mengatasi keterbatasan dataset lama seperti DEG atau OGEE yang sering memiliki konflik label atau kurang lengkap. Selain itu, CLEARER telah digunakan ulang oleh beberapa peneliti lain, seperti Azhari (2025) dengan metode *CatBoost* pada dataset *Drosophila* yang sama, serta Campos et al. (2021, 2024) untuk prediksi *cross-species essential genes* lintas eukariota termasuk parasit dan nematoda.

Meskipun alternatif dataset lain tersedia seperti DEG (versi terbaru untuk data esensialitas eksperimental), OGEE (integrasi RNAi screens), *FlyBase* (sumber *sequence* dan anotasi resmi), atau dataset Aromolaran et al. (2020) dengan 27.340 fitur *sequence-functional* CLEARER tetap paling sesuai karena menyediakan data *sequence-only* yang komprehensif, label terpisah CEG/OEG, dan kredibilitas tinggi (sudah dikutip >25 kali di jurnal Q1), sehingga mendukung fokus penelitian ini pada perbandingan *AdaBoost* dan

XGBoost dengan fitur *sequence-only* pada spesies tunggal tanpa orthologs lintas spesies.

Dengan memanfaatkan dataset terkurasi dari Beder et al. (2021) yang mengintegrasikan data sekuens DNA dan protein *Drosophila melanogaster* dari berbagai sumber, penelitian ini diharapkan dapat memberikan kontribusi ilmiah berupa analisis komparatif yang memperkuat literatur dan membuka peluang penerapan yang lebih luas dalam bidang kedokteran, bioteknologi, dan farmasi. Selain itu, hasil penelitian ini dapat menjadi dasar pengembangan model prediksi gen esensial yang lebih akurat dan efisien di masa depan.

1.2. Rumusan Masalah

Berdasarkan uraian latar belakang tersebut, maka perumusan masalah dalam penelitian ini adalah:

1. Mengklasifikasi gen esensial pada *Drosophila melanogaster* dapat dilakukan menggunakan metode *AdaBoost* dan *XGBoost* berbasis data sekuens DNA dan protein.
2. Menentukan proporsi pembagian dataset yang paling tepat guna menghasilkan performa klasifikasi terbaik.
3. Membandingkan hasil performa model *AdaBoost* dan *XGBoost* pada dataset yang sama.

1.3. Batasan Masalah

Untuk menjaga fokus penelitian, batasan masalah ditetapkan sebagai berikut:

1. Dataset yang digunakan bersumber dari Beder et al. (2021) yang telah mengumpulkan dan mengkurasi data sekuens DNA serta protein *Drosophila melanogaster*.
2. Metode *machine learning* yang digunakan hanya mencakup *AdaBoost* dan *XGBoost*, tanpa melibatkan metode boosting lain atau *deep learning*.
3. Fitur yang digunakan berupa representasi sekuens DNA dan protein yang diekstraksi menggunakan metode *feature engineering* (termasuk *Amino Acid*

Composition/AAC untuk protein serta *Tri-Nucleotide Composition*/TNC dan *Fourier Transform/FT* untuk DNA) dan *PPI_degree.*

4. Evaluasi performa model dilakukan menggunakan metrik akurasi, presisi, *recall*, *F1-score*, dan *Area Under the Curve* (AUC).
5. Penelitian ini sepenuhnya bersifat komputasional (*in silico*) dan tidak mencakup validasi laboratorium (*wet lab*).

1.4. Tujuan Penelitian

Tujuan dari penelitian ini adalah:

1. Mengimplementasikan model klasifikasi gen esensial pada *Drosophila melanogaster* menggunakan metode *AdaBoost* dan *XGBoost* berbasis data sekuens DNA dan protein.
2. Membandingkan performa model *AdaBoost* yang dihasilkan dengan *XGBoost* pada dataset serupa.

1.5. Manfaat Penelitian

Penelitian ini diharapkan dapat :

1. Membantu mengidentifikasi gen esensial pada *Drosophila melanogaster* secara lebih akurat dan efisien berbasis data sekuens DNA dan protein melalui pendekatan komputasional.
2. Memvalidasi hasil analisis *in silico* sebelum dilakukan penelitian lanjutan *in vitro* atau *in vivo*.
3. Menambah literatur ilmiah mengenai penerapan algoritma *AdaBoost* dan *XGBoost* dalam bidang bioinformatika, khususnya pada tugas klasifikasi gen esensial.
4. Mendukung pengembangan riset lanjutan di bidang kedokteran, bioteknologi, dan farmasi, seperti penemuan target obat baru serta pengembangan terapi berbasis gen melalui identifikasi gen esensial yang lebih tepat.

II. TINJAUAN PUSTAKA

2.1 Penelitian Terdahulu

Penelitian ini berlandaskan pada berbagai studi terdahulu yang memiliki relevansi dengan topik yang diangkat. Adanya penelitian sebelumnya menjadi acuan sekaligus perbandingan dalam melihat persamaan maupun perbedaan objek, metode, dan hasil yang diperoleh. Ringkasan beberapa penelitian terdahulu yang mendukung kajian ini ditampilkan pada Tabel 1.

Tabel 1. Penelitian Terdahulu

NO	Penelitian	Data	Metode	Hasil
1.	(Aromolaran et al, 2020)	*Drosophila melanogaster (FlyBase);* DNA + protein *sequences, PPI, gene network,* konservasi evolusi, anotasi fungsional → 27.340 fitur total	- *Preprosessing* : Integrasi multi-sumber; pembersihan *sequence*; normalisasi fitur numerik - *Feature Extraction* : Ekstraksi: *sequence (k-mer), network topology, functional annotations* - *Feature Selection* : Fitur signifikan (P < 0.001) berdasarkan korelasi dengan label esensial - Pembagian Data: *5-fold cross-validation (stratified)* - Metode : *Random Forest*; metrik: *Accuracy*, *F1-score*, ROC-AUC, PR-AUC	Akurasi 82%; F1 80%; ROC-AUC 90%; PR-AUC 30%; fitur protein paling berpengaruh

NO	Penelitian	Data	Metode	Hasil
2.	(Liu et al, 2020)daftar	Dataset Protein-*Protein Interaction* (PPI) *human-virus* (sumber: HPRD, *VirusMint*); ~10.000 sampel interaksi positif/negatif; analisis: pola *sequence protein, dataset imbalanced* (non-interaksi > interaksi)	- *Preprosessing* : Normalisasi *sequence* (PSSM); penanganan *imbalance* dengan *undersampling oversampling* - *Feature Extraction* : Ekstraksi: *Conjoint Triad* (CT), *Auto Covarienve* (AC), *Local Descriptor* (LD) dari *sequence* protein - *Feature Selection* : Pemilihan fitur relevan berdasarkan korelasi; reduksi dimensi (misalnya PCA jika diperlukan) - Pembagian Data : 5-*fold cross-validation; split 80/20 train/test* - Metode : *AdaBoost* (*meta-algorithm* dengan *decision trees*); metrik: *Accuracy*, AUC, *Precision-Recall*; perbandingan dengan SVM, RF, MLP	**Akurasi 85%; AUC 87%; unggul 5-10% dibanding SVM**
3.	(Wang et al, 2022)	Protein multi-organisme (fokus *S. cerevisiae*); sumber: BioGRID (PPI), SGD (anotasi), DEG/OGEE (essential genes) → ~5.000-6.000 protein, weighted PPI network; analisis: identifikasi 1.285 essential protein	- *Preprosessing* : Pembersihan PPI (hapus duplikat/*self-loop*); integrasi data biologis; *weighting edge* berdasarkan *orthology & subcellular localization* - *Feature Extraction* : Ekstraksi: *Node2vec embedding (topological vectors*	Akurasi 88%; AUC 90%; *Precision* 60%; *Recall* 58%; F1 59%

NO	Penelitian	Data	Metode	Hasil
			dari weighted PPI); PCA reduksi dimensi (optimal 12 dimensi) - *Feature Selection*: Fokus *topological features (degree, betweenness);* PCC untuk *similarity*; kritik *manual selection, prioritaskan high-relevance* - Pembagian Data : *10-fold cross-validation; split 80/20 train/test* - Metode : *XGBoost classifier; metrik: Accuracy, Precision, Recall*, *F1-score*, AUC; perbandingan dengan RF, SVM, DT	
4.	(Azhari, R.N., 2025)	Dataset DNA & protein *Drosophila melanogaster* (Beder et al., 2021): CEG ~17.744, OEG ~900 *(imbalanced)*	- *Pre-processing*: *Cleaning data* (hapus duplikat, kosong, self-loop); normalisasi *sequence; pra-proses* persiapan data untuk pengolahan (konversi ke numerik). - *Feature Extraction* : Protein: Ekstraksi *k-mer, functional features* (hasil numerik), DNA: *Trinucleotide Composition* (TNC) & *Fourier Transform* (FT) untuk mengubah *sequence* ke fitur vektor. Total fitur: ribuan, integrasi multi-sumber.	OEG: AUC 92%; CEG: AUC 84%; stabil dan cepat pada data *imbalanced*

NO	Penelitian	Data	Metode	Hasil
			- *Feature Selection* : *Random Forest Gini Importance* (GII) untuk pilih fitur berpengaruh (P<0,001); reduksi dimensi (misalnya ke 45 fitur pada CEG); otomatis kurangi fitur tidak relevan - Pembagian Data : *Split* 90/10 atau variasi *(train/test); stratified* untuk *imbalance; opsional k-fold CV; pseudocode: X_train, X_test, y_train, y_test.* - Metode : *CatBoost* (iterasi 1000, *learning rate* 0.1); *handle categorical*; SMOTE opsional untuk *oversampling*; metrik: *Accuracy*, ROC-AUC, PR-AUC, *Sensitivity* (Sn), *Specificity* (Sp); *training vs. testing* untuk deteksi *overfitting*.	
5.	(Beder et al., 2021)	*Drosophila melanogaster* (OGEE *database*): CEG & OEG (~13.000 gen); *imbalanced*	- *Preprosessing* : Integrasi data dari OGEE/DEG; pembersihan anotasi; *balancing imbalance* dengan SMOTE *oversampling* - *Feature Extraction* : Ekstraksi: *Sequence features (k-mer, PSSM), functional annotations (GO terms), network*	CEG: RF AUC 84%, XGBoost 84%; OEG: RF 92%, XGBoost 86%; fitur network paling informatif

NO	Penelitian	Data	Metode	Hasil
			topology (PPI *degree*) → ~500 fitur - *Feature Selection* : Pemilihan fitur relevan berdasarkan *importance* (permutation test); reduksi dimensi untuk hindari *overfitting* - Pembagian Data : 5-*fold cross-validation (stratified* untuk *balance* kelas) - Metode : *Random Forest & XGBoost*; metrik: ROC-AUC, PR-AUC, *Sensitivity, Specificity;* perbandingan dengan GLM, SVM, NNET	

Berdasarkan Tabel 1, dapat dilihat bahwa penelitian tentang klasifikasi gen esensial pada *Drosophila melanogaster* serta penggunaan algoritma *boosting* dalam bioinformatika terus menghasilkan performa yang baik. Penelitian Aromolaran et al. (2020) menunjukkan bahwa penggabungan data sekuens DNA dan protein dengan fitur jaringan serta anotasi fungsional (sekitar 27.340 fitur) mampu menghasilkan akurasi 82%, F1 80%, serta ROC-AUC 90% menggunakan *Random Forest*, sehingga menekankan pentingnya integrasi data multi-sumber pada organisme model ini.

Penelitian Liu et al. (2020) membuktikan keunggulan *AdaBoost* dalam memprediksi interaksi *protein human-virus*, dengan akurasi 85% dan AUC 87%, serta performa lebih baik 5–10% dibandingkan dengan SVM, khususnya pada data yang tidak seimbang. Sementara itu, Wang et al. (2022) berhasil mencapai akurasi 88% dan AUC 90% menggunakan *XGBoost* berbasis *embedding Node2vec* pada data protein esensial lintas organisme.

Penelitian Azhari (2025) yang memanfaatkan dataset DNA dan protein *Drosophila melanogaster* dari Beder et al. (2021) menunjukkan hasil AUC

92% pada OEG dan 84% pada CEG melalui *CatBoost*, dengan keunggulan stabilitas tinggi dan komputasi cepat meskipun data *imbalanced*.

Selanjutnya, Beder et al. (2021) memperlihatkan performa kompetitif *Random Forest* dan *XGBoost* pada dataset OGEE *Drosophila melanogaster*, dengan AUC mencapai 84% pada CEG serta 86–92% pada OEG, di mana fitur jaringan memberikan kontribusi paling besar.

Dari kelima studi tersebut, terlihat bahwa meskipun *AdaBoost* telah terbukti efektif di berbagai aplikasi bioinformatika, penerapannya khusus untuk klasifikasi gen esensial *Drosophila melanogaster* dengan fitur *sequence-only* masih sangat terbatas. Belum ada penelitian yang secara langsung membandingkan *AdaBoost* dan *XGBoost* pada dataset spesies tunggal yang sama. Penelitian ini mengisi kesenjangan tersebut dengan pendekatan komputasional yang fokus, reproducible, dan berorientasi pada performa di kondisi imbalance tinggi.

2.2 Kajian Teori

2.2.1 Gen Esensial

Gen esensial merupakan elemen genetik fundamental yang berperan dalam mempertahankan viabilitas suatu organisme. Gen ini menyimpan instruksi biologis yang mengatur berbagai fungsi vital, termasuk metabolisme energi, sintesis protein, pertumbuhan, serta mekanisme pertahanan sel terhadap kerusakan (Guo et al., 2021). Tanpa keberadaan gen ini, organisme tidak mampu menjalankan aktivitas fisiologis dasar sehingga kelangsungan hidupnya akan terganggu secara signifikan. Pada organisme prokariotik maupun eukariotik, gen esensial terorganisasi dalam kromosom dengan struktur DNA linier yang memungkinkan regulasi ekspresi berlangsung secara terkoordinasi.

Menurut penelitian Peng et al. (2017), gen esensial sangat penting karena mendukung mekanisme inti kehidupan, seperti replikasi DNA, pembelahan sel, hingga translasi protein. Dengan kata lain, gen ini tidak sekadar mendukung pertumbuhan, melainkan juga menjadi syarat mutlak bagi keberlangsungan siklus hidup sel. Sementara itu,

menurut Zhang & Ren. (2015) menegaskan bahwa kriteria utama suatu gen disebut esensial adalah ketika penghapusan atau mutasinya menimbulkan hilangnya viabilitas sel atau bahkan menyebabkan kematian organisme. Definisi ini menegaskan bahwa gen esensial bersifat nonredundans, artinya tidak ada gen lain yang dapat sepenuhnya menggantikan fungsinya.

Kajian tentang gen esensial, dengan demikian, memiliki posisi strategis dalam pengembangan ilmu pengetahuan modern. Di satu sisi, dapat memperkaya pemahaman mengenai dasar-dasar mekanisme kehidupan pada tingkat molekuler. Di sisi lain, ia membuka peluang besar bagi inovasi di bidang bioinformatika, farmakologi, dan rekayasa biomedis. Integrasi antara biologi molekuler, data omics, dan pembelajaran mesin tidak hanya memperluas wawasan teoritis, tetapi juga mempercepat terciptanya solusi praktis terhadap tantangan kesehatan global, khususnya dalam penemuan target terapi yang lebih efektif dan personalisasi pengobatan.

2.2.2 DEG

Database of Essential Genes (DEG) merupakan basis data komprehensif yang mengumpulkan rekaman gen esensial yang telah divalidasi secara eksperimental dari berbagai organisme, termasuk bakteri, arkea, dan eukariota (Luo et al, 2021). DEG 15, sebagai versi terbaru pada tahun 2021, telah meningkatkan jumlah rekaman gen esensial eukariota lebih dari empat kali lipat dan gen esensial prokariota lebih dari dua kali lipat dibandingkan versi sebelumnya. Database ini menyertakan alat analisis built-in, seperti distribusi lokalisasi subseluler, pengayaan *Gene Ontology* dan jalur KEGG, serta pencarian BLAST yang dapat dikustomisasi, sehingga mendukung prediksi dan analisis gen esensial secara lebih efisien pada organisme model seperti *Drosophila melanogaster* (Peng et al, 2024).

Berdasarkan kajian terhadap *Database of Essential Genes* (DEG), dapat disimpulkan bahwa DEG merupakan sumber data esensial yang sangat berharga dalam penelitian bioinformatika, khususnya untuk

klasifikasi gen esensial pada *Drosophila melanogaster*. Dengan integrasi data eksperimental yang terus diperbarui dan fitur analisis terintegrasi, DEG tidak hanya memfasilitasi identifikasi gen vital lintas spesies, tetapi juga mendukung pendekatan komputasional seperti *machine learning* untuk memprediksi esensialitas gen berbasis sekuens DNA dan protein. Hal ini memperkuat pentingnya penggunaan dataset terkurasi seperti dari Beder et al. (2021) dalam penelitian ini, guna meningkatkan akurasi model *AdaBoost* dan *XGBoost* serta kontribusi terhadap pemahaman mekanisme biologis dasar pada organisme model.

2.2.3 CEG

Cellular Essential Genes (CEG) merupakan kategori gen esensial yang diidentifikasi pada tingkat seluler, yaitu gen-gen yang esensial untuk viabilitas dan proliferasi sel, biasanya ditentukan melalui skrining RNAi atau knockout pada kultur sel (*cell lines*). Pada *Drosophila melanogaster*, CEG mencakup gen-gen yang mendukung proses biogenesis seluler dasar dan siklus sel, sehingga penghapusan atau *knockdown*-nya menyebabkan hilangnya viabilitas sel. Berbeda dengan *Organismal Essential Genes* (OEG) yang diidentifikasi pada tingkat organisme utuh (melalui skrining *knockout* yang memengaruhi fertilitas, morfogenesis, atau viabilitas keseluruhan), CEG cenderung lebih konservatif dan sering tumpang tindih dengan gen esensial dasar yang diperlukan untuk fungsi seluler universal. Dataset CEG pada *Drosophila melanogaster* umumnya bersumber dari skrining RNAi pada kultur sel, yang menghasilkan jumlah gen esensial lebih banyak dibandingkan dengan OEG, tetapi dengan overlap yang terbatas karena perbedaan konteks biologis (Beder et al., 2021).

Berdasarkan kajian terhadap *Cellular Essential Genes* (CEG), dapat disimpulkan bahwa CEG mewakili gen esensial pada tingkat seluler yang lebih mendasar dan konservatif dibandingkan dengan OEG, sehingga lebih *reliable* sebagai label utama dalam klasifikasi komputasional gen esensial pada *Drosophila melanogaster*.

Penggunaan dataset CEG dari skrining RNAi seluler, seperti yang dikurasi dalam penelitian Beder et al. (2021), memperkuat pendekatan *in silico* dalam tesis ini, karena memungkinkan model *AdaBoost* dan *XGBoost* untuk fokus pada fitur sekuens DNA dan protein yang mendukung fungsi vital seluler. Hal ini tidak hanya meningkatkan akurasi prediksi, tetapi juga memberikan wawasan biologis yang lebih tepat untuk aplikasi di bidang bioinformatika, kedokteran, dan bioteknologi.

2.2.4 OEG

Organismal Essential Genes (OEG) merupakan kategori gen esensial yang diidentifikasi pada tingkat organisme utuh (*whole organism*), yaitu gen-gen yang esensial untuk viabilitas, fertilitas, morfogenesis, regulasi perkembangan, dan proses persinyalan saraf pada organisme multiseluler. Pada *Drosophila melanogaster*, OEG ditentukan melalui skrining *knockout* (seperti *P-element insertion*, RNAi sistemik, atau CRISPR) yang memengaruhi kelangsungan hidup atau reproduksi organisme secara keseluruhan. Berbeda dengan *Cellular Essential Genes* (CEG) yang difokuskan pada viabilitas sel individu melalui skrining pada kultur sel, OEG mencakup gen-gen yang mendukung fungsi terkoordinasi antarjaringan dan antarorgan, sehingga sering kali melibatkan proses regulasi, perkembangan, dan signaling yang tidak esensial pada tingkat sel tunggal. *Overlap* antara OEG dan CEG relatif rendah, karena OEG mencerminkan kebutuhan fungsional pada konteks organisme *multiseluler* yang lebih kompleks (Beder et al., 2021).

Berdasarkan kajian terhadap *Organismal Essential Genes* (OEG), dapat disimpulkan bahwa OEG mewakili gen esensial pada tingkat organisme utuh yang lebih spesifik terhadap proses perkembangan dan koordinasi multicellular dibandingkan CEG, sehingga memberikan perspektif biologis yang lebih holistik pada Drosophila melanogaster. Meskipun dataset OEG memiliki jumlah sampel lebih sedikit dan overlap rendah dengan CEG seperti yang dikurasi dalam

penelitian Beder et al. (2021), penggunaan CEG sebagai label utama dalam penelitian ini tetap tepat karena sifatnya yang lebih konservatif dan *reliable* untuk prediksi komputasional. Namun, pemahaman tentang OEG memperkaya interpretasi hasil model *AdaBoost* dan *XGBoost*, terutama dalam menghubungkan fitur sekuens DNA serta protein dengan mekanisme esensialitas pada konteks organisme model, yang berpotensi mendukung aplikasi lanjutan di bidang bioteknologi dan terapi gen.

2.2.5 *Sequence* DNA

DNA (*Deoxyribonucleic Acid*) adalah molekul utama yang menyimpan informasi genetik pada seluruh organisme hidup. Molekul ini tersusun atas nukleotida yang terdiri dari gula deoksiribosa, gugus fosfat, dan basa nitrogen. Bentuk khas DNA adalah struktur heliks ganda (*double helix*) yang berpilin, di mana setiap basa nitrogen membentuk pasangan komplementer yang spesifik: Adenin (A) dengan Timin (T), serta Guanin (G) dengan Sitosin (C), (Gunasekaran et al., 2021). Keteraturan pasangan basa ini memungkinkan DNA berfungsi sebagai cetak biru yang stabil bagi pertumbuhan, perkembangan, dan fungsi seluler.

Urutan basa nitrogen dalam DNA berperan sebagai instruksi biologis yang mendasari proses sintesis protein, regulasi metabolisme, hingga mekanisme replikasi sel. Dengan kata lain, DNA bukan sekadar penyimpan informasi, tetapi juga penyedia panduan fundamental yang memastikan keberlangsungan kehidupan. Informasi genetik yang tersusun secara linear pada kromosom tersebut disebut genom, dan melalui sekuensing DNA, peneliti dapat memetakan gen, memahami fungsi biologisnya, serta mengidentifikasi variasi genetik yang memengaruhi kesehatan maupun penyakit (Glick & Patten, 2022).

Selain itu, analisis bioinformatika terhadap sekuens DNA memungkinkan ekstraksi berbagai fitur penting yang dapat dijadikan indikator biologis. Beberapa fitur yang umum digunakan antara lain

komposisi basa GC (*GC-content*), panjang gen, distribusi motif regulator, pola kodon, hingga potensi pembentukan struktur sekunder. Informasi ini memiliki nilai strategis dalam penelitian, misalnya untuk membedakan gen esensial dari gen non-esensial, memprediksi ekspresi gen, maupun mengeksplorasi mekanisme regulasi molekuler yang kompleks.

Dalam konteks modern, sekuens DNA sering direpresentasikan dalam bentuk numerik agar dapat diproses oleh algoritma komputasional. Transformasi data biologis ini menjadi langkah awal dalam penerapan machine learning, di mana pola-pola tersembunyi dalam urutan DNA dapat dianalisis secara sistematis. Representasi numerik ini mendukung berbagai aplikasi, seperti klasifikasi gen, prediksi fungsi, dan identifikasi target obat. Dengan demikian, DNA tidak hanya penting dalam biologi molekuler, tetapi juga menjadi pusat perhatian dalam perkembangan teknologi bioinformatika berbasis kecerdasan buatan.

Dapat disimpulkan bahwa, DNA ibarat sebuah buku instruksi kehidupan yang ditulis dengan huruf-huruf A, T, C, dan G. Setiap gen adalah paragraf di dalam buku tersebut. Gen esensial adalah paragraf-paragraf yang paling krusial—jika dihapus atau rusak, seluruh "buku" (organisme) tidak akan bisa berfungsi dengan baik atau bahkan tidak akan bertahan hidup. Oleh karena itu, memahami pola dan komposisi sekuens DNA bukan hanya soal angka dan statistik, melainkan upaya untuk mendengar lebih jelas "pesan" yang disampaikan oleh alam tentang mana gen-gen yang paling fundamental bagi kelangsungan kehidupan.

Pendekatan berbasis sekuens DNA yang digunakan dalam penelitian ini (melalui Tri-Nucleotide Composition dan Fourier Transform) merupakan salah satu cara untuk menerjemahkan bahasa molekuler tersebut ke dalam bentuk yang dapat dipahami dan dimanfaatkan oleh algoritma machine learning.

2.2.6 Protein

Protein merupakan senyawa organik kompleks dengan bobot molekul tinggi yang tersusun atas rantai panjang asam amino yang saling terikat melalui ikatan peptida (Harjanto, 2017). Setiap rantai polipeptida terdiri dari susunan asam amino yang jumlah dan urutannya bervariasi, sehingga menghasilkan struktur dan fungsi protein yang berbeda. Protein berperan penting sebagai penyedia bahan dasar untuk pertumbuhan organisme, pemelihara jaringan tubuh, serta bahan bakar metabolisme dan pengatur berbagai proses biologis (Fairuz et al., 2022).

Struktur protein dibedakan menjadi beberapa tingkatan, yaitu struktur primer (urutan asam amino), sekunder (pola lokal seperti heliks α dan lembaran β), tersier (konformasi tiga dimensi), serta kuartener (gabungan beberapa polipeptida). Kompleksitas struktur tersebut memungkinkan protein menjalankan fungsi yang beragam, mulai dari peran struktural, enzimatik, regulator, hingga sebagai molekul sinyal dalam hampir seluruh proses seluler (Alberts et al., 2015).

Protein juga memiliki keterkaitan langsung dengan DNA karena instruksi untuk menyusunnya dikodekan oleh gen. Proses ini dijelaskan dalam dogma sentral biologi molekuler: DNA ditranskripsi menjadi RNA, lalu RNA diterjemahkan menjadi protein. Perubahan atau mutasi pada DNA dapat memengaruhi struktur maupun fungsi protein. Protein tidak normal dapat menyebabkan gangguan metabolisme, penyakit genetik, bahkan kerusakan sistem seluler secara menyeluruh (Nelson & Cox, 2017).

Dalam konteks gen esensial, protein hasil ekspresi gen tersebut bersifat vital karena berperan dalam fungsi dasar kehidupan sel, seperti replikasi DNA, transkripsi, translasi, dan metabolisme energi. Sebagian besar protein dari gen esensial bersifat konservatif lintas spesies, sehingga memiliki kesamaan struktur dan fungsi mulai dari organisme sederhana hingga kompleks. Hal ini menjadikan protein

esensial sebagai target potensial dalam pengembangan obat, terutama pada bakteri dan jamur patogen, di mana penghambatan protein esensial dapat mengganggu kelangsungan hidup patogen tanpa merusak inang (Acencio & Lemke, 2009).

Dapat disimpulkan bahwa, protein dapat diibaratkan sebagai "mesin-mesin kecil" yang bekerja tanpa henti di dalam sel. Jika DNA adalah buku resep, maka protein adalah hidangan yang dihasilkan dari resep tersebut. Gen esensial menghasilkan protein-protein yang paling vital—seperti mesin-mesin utama yang tidak boleh rusak atau dimatikan. Tanpa protein-protein esensial ini, sel tidak akan mampu melakukan metabolisme, replikasi, perbaikan diri, maupun pembelahan. Oleh karena itu, memahami komposisi asam amino dalam protein bukan sekadar menghitung persentase, melainkan upaya untuk mengenali "kunci-kunci" molekuler yang menentukan apakah suatu gen benar-benar esensial bagi kehidupan.

Protein tersusun dari 20 jenis asam amino dengan struktur yang berbeda tetapi saling berkaitan dalam membentuk polipeptida. Setiap asam amino memiliki nama, singkatan, dan simbol satu huruf yang digunakan secara luas dalam representasi urutan protein. Daftar asam amino menurut Fairuz, et al (2022), ditampilkan pada Tabel 2.

Tabel 2. Daftar Asam Amino (Fairuz et al., 2023).

Nama	**Singkatan**	**Simbol**	**Nama**	**Singkatan**	**Simbol**
Alanine	Ala	A	Methionine	Met	M
Cysteine	Cys	C	Asparagine	Asn	N
Aspartic Acid	Asp	D	Proline	Pro	P
Glutamic Acid	Glu	E	Glutamine	Gln	Q
Phenylalanine	Phe	F	Arigine	Arg	R
Glycine	Gly	G	Srine	Ser	S
Histidine	His	H	Threonine	Thr	T
Isoleucine	Ile	I	Valine	Va;	V
Lysine	Lys	K	Tryptophan	Trp	W
Leucine	Ieu	L	Tyrosine	Tyr	Y

2.2.7 Drosophila Melanogaster

Drosophila melanogaster merupakan salah satu organisme model yang paling banyak digunakan dalam penelitian biologi dan biomedis.

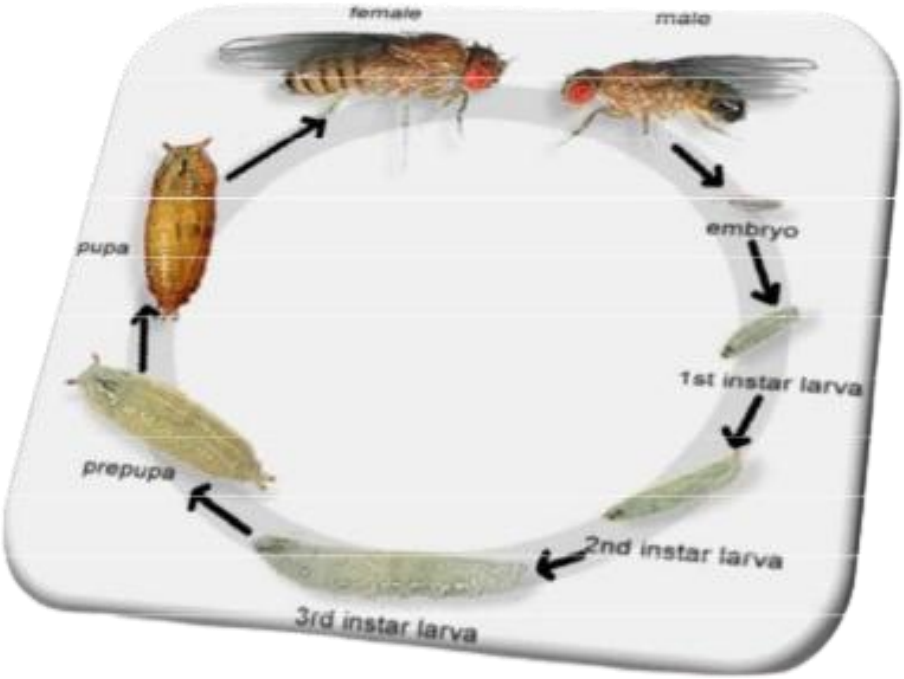
Lalat buah ini dipilih karena siklus hidupnya yang singkat, kemampuan reproduksi yang tinggi, serta kemudahan pemeliharaannya. Selain itu, struktur tubuhnya yang termasuk dalam eukariota tingkat tinggi (memiliki sistem organ yang relatif kompleks) membuatnya relevan untuk mempelajari mekanisme molekuler yang juga terjadi pada manusia. Secara taksonomi, spesies ini diklasifikasikan ke dalam *Kingdom Animalia, Filum Arthropoda, Kelas Insecta, Ordo Diptera, Famili Drosophilidae, Genus Drosophila*, dan Spesies *Drosophila melanogaster* (Bier, 2005).

Secara morfologi, *D. melanogaster* berukuran kecil dengan panjang tubuh rata-rata 3–5 mm. Mata majemuknya berwarna merah mencolok dan pada bagian kepala terdapat mata tunggal (*ocelli*) yang berukuran lebih kecil dibandingkan dengan mata majemuknya. Tubuh lalat buah ini berwarna kuning kecoklatan dengan bagian posterior yang memiliki cincin gelap. Sayapnya berasal dari *toraks*, memiliki vena tepi (*costal vein*) dengan pola percabangan tertentu, sedangkan antenanya memiliki arista berbulu dengan percabangan 7–12 (Bier, 2005). Ciri-ciri morfologi ini menjadi dasar identifikasi dan pemeliharaan *strain* di laboratorium. Gambar *Drosophila melanogaster* dapat dilihat pada Gambar 1.

Gambar 1. *Drosophila Melanogaster* (Lewis, E., 2005).

Siklus hidup *Drosophila melanogaster* terlihat pada Gambar 2. Melalui empat tahap utama, yaitu embrio, larva, pupa, dan dewasa

(imago). Dalam kondisi optimal (suhu ±25 °C, nutrisi cukup, kepadatan populasi terkontrol), waktu yang dibutuhkan dari telur hingga dewasa hanya sekitar 10 hari. Setelah keluar dari pupa, individu dewasa membutuhkan waktu 2–4 hari untuk mencapai kematangan seksual (Basseva & Wagner, 2019).



Gambar 2. Siklus Hidup *Drosophila Melanogaster* (Abolaji et al., 2013).

Keunggulan *Drosophila melanogaster* sebagai organisme model sudah diakui secara luas. Sekitar 60–75% gen yang terkait penyakit pada manusia memiliki homolog pada lalat buah (Locke et al., 2011). Selain itu, tersedia berbagai teknik manipulasi genetik yang maju, termasuk mutan, kromosom balancer, dan sistem ekspresi gen terkontrol, sehingga mempermudah penelitian genetika dan penyakit manusia (Scully et al., 2012). Organismenya kecil, murah untuk dipelihara, siklus hidupnya cepat, dan relatif bebas dari pembatasan etis yang sering menyertai penelitian pada vertebrata (Ugur et al., 2016).

Berbagai studi modern memanfaatkan *D. melanogaster* dalam bidang yang luas, misalnya nutrisi dan penuaan (Fontana & Partridge,

2015), respons imun bawaan dan interaksi patogen (Lemaitre & Hoffmann, 2007), toksikologi lingkungan seperti pengaruh mikroplastik dan logam berat (Jimenez-Jacinto et al., 2023), hingga skrining awal obat-obatan potensial (Pandey & Nichols, 2011). Karena sifat-sifat tersebut, *D. melanogaster* menjadi model yang sangat efektif untuk memahami mekanisme dasar biologi serta membantu mengungkap penyakit-penyakit manusia (Bier, 2005).

Walaupun demikian, *Drosophila melanogaster* tidak sepenuhnya merepresentasikan organisme kompleks seperti manusia. Lalat buah tidak memiliki sistem imun adaptif dan beberapa organ spesifik manusia seperti hati, ginjal, atau sistem kardiovaskular berbilik, sehingga hasil penelitian tertentu perlu divalidasi pada model yang lebih dekat dengan manusia (Ugur et al., 2016). Namun, kecepatan reproduksi, kesamaan genetik yang signifikan, kemudahan manipulasi, serta biaya yang rendah membuatnya tetap menjadi model yang ideal untuk studi genetika, biologi perkembangan, dan toksikologi.

2.2.8 Machine Learning

Machine Learning (ML) atau pembelajaran mesin adalah salah satu cabang dari *artificial intelligence* (AI) yang memungkinkan sistem komputer untuk mempelajari pola dari data secara otomatis tanpa memerlukan pemrograman eksplisit setiap kali terjadi pembaruan data. Pendekatan ini banyak digunakan untuk meniru perilaku manusia dalam memecahkan masalah dan melakukan otomatisasi (Hania, 2017). Secara umum, *machine learning* memberikan kemampuan bagi komputer untuk mengekstraksi informasi dari data dan menggunakannya untuk membuat prediksi atau keputusan baru secara mandiri (Mitchell, 1997).

Menurut Goodfellow et al. (2016), *machine learning* dapat dipandang sebagai disiplin ilmu yang menggabungkan algoritma dan teknik statistik untuk mengajarkan komputer mengenali pola kompleks dalam data. Teknologi ini digunakan dalam berbagai bidang

seperti pengenalan suara, pengenalan gambar, sistem rekomendasi, analisis genom, dan klasifikasi data biologis.

Proses pembelajaran dalam *machine learning* dilakukan melalui tahap pelatihan (*training*) menggunakan dataset yang sudah diberi label maupun tidak (*supervised* dan *unsupervised*). Pada tahap ini, algoritma *machine learning* mengidentifikasi parameter atau fitur penting dari data untuk membentuk model prediktif. Model ini kemudian digunakan untuk mengklasifikasikan data baru atau membuat prediksi berdasarkan pola yang telah dipelajari sebelumnya (Geron, 2019).

Pemilihan algoritma dan kualitas data latih sangat menentukan kinerja sistem *machine learning*. Algoritma populer yang sering digunakan dalam pengolahan data biologis antara lain *Support Vector Machine (SVM)*, *Random Forest*, *Gradient Boosting* (*XGBoost*, *CatBoost*), dan *Deep Learning* (LeCun et al., 2015). Dengan kemampuannya yang fleksibel, *machine learning* menjadi teknologi yang esensial untuk analisis data berskala besar, termasuk untuk klasifikasi DNA dan protein.

2.2.9 Klasifikasi

Klasifikasi merupakan salah satu cabang utama dalam pembelajaran mesin (*machine learning*) yang berfokus pada pemahaman pola dan pengelompokan data untuk menghasilkan representasi dan generalisasi yang baik. Menurut Raharjo & Quaffou, (2015), klasifikasi adalah proses yang dilakukan untuk mengidentifikasi pola representasi data sehingga mampu melakukan prediksi terhadap data baru. Sejalan dengan itu, Kamus Besar Bahasa Indonesia mendefinisikan klasifikasi sebagai penyusunan sistematis dalam kelompok atau golongan berdasarkan kaidah atau standar tertentu yang telah ditetapkan.

Menurut penelitian Darnisa et al. (2019), klasifikasi bertujuan untuk mengelompokkan objek baru ke dalam kelas atau label tertentu berdasarkan atribut-atribut yang telah ditentukan sebelumnya. Dengan

kata lain, klasifikasi memungkinkan peneliti untuk memprediksi kelas dari suatu objek yang belum diketahui kategorinya melalui analisis terhadap fitur-fitur yang dimilikinya.

Dalam konteks bioinformatika, klasifikasi DNA dan protein dilakukan untuk membedakan gen esensial dan non-esensial berdasarkan pola tertentu pada urutan nukleotida atau asam amino. Pendekatan ini memanfaatkan metode pembelajaran mesin untuk mengekstraksi ciri-ciri (*features*) penting dari data biologis yang kemudian digunakan sebagai dasar pengklasifikasian (Zhang et al., 2021). Proses ini biasanya melibatkan tahap awal berupa ekstraksi fitur, misalnya komposisi nukleotida, frekuensi k-mers, atau transformasi berbasis sinyal seperti transformasi Fourier untuk merepresentasikan karakteristik utama dari sekuens DNA/protein.

Pemilihan fitur yang tepat sangat penting dalam meningkatkan akurasi model klasifikasi. Fitur yang relevan dapat memudahkan algoritma untuk memisahkan kelas dengan jelas dan meminimalkan kesalahan klasifikasi (Han et al., 2012). Pada klasifikasi DNA dan protein, fitur yang digunakan harus mencerminkan perbedaan biologis yang signifikan sehingga model mampu mengenali pola yang membedakan gen esensial dan non-esensial secara akurat (Luo et al., 2023).

Berbagai algoritma klasifikasi dapat digunakan, seperti *CatBoost*, *XGBoost, Random Forest*, atau *Support Vector Machine*. *CatBoost* sendiri termasuk algoritma berbasis *gradient boosting* yang dirancang untuk menangani data kategorikal secara lebih baik dan mengurangi *overfitting*, sehingga cocok digunakan untuk klasifikasi data biologis yang memiliki banyak variabel (Prokhorenkova et al., 2018). Dengan pendekatan ini, peneliti dapat meningkatkan performa prediksi dan mendapatkan pemahaman yang lebih baik tentang karakteristik gen esensial pada tingkat genom maupun proteom.

2.2.10 *AdaBoost*

AdaBoost (*Adaptive Boosting*) merupakan algoritma *ensemble learning* yang diperkenalkan pertama kali oleh Freund dan Schapire (1995) melalui paper "*A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting*". Algoritma ini muncul sebagai pengembangan dari konsep *boosting* yang telah dikenalkan lebih awal oleh Kearns dan Valiant (1988), yang pada saat itu mempertanyakan apakah *weak learner* sederhana dapat ditingkatkan performanya menjadi *strong learner* yang akurat.

Freund dan Schapire menjawab pertanyaan tersebut dengan merancang algoritma *AdaBoost*, yang secara adaptif menyesuaikan bobot data pelatihan berdasarkan tingkat kesalahan pada iterasi sebelumnya. Hal ini menjadi tonggak penting dalam perkembangan *ensemble methods* karena untuk pertama kalinya sebuah pendekatan formal terbukti dapat meningkatkan akurasi klasifikasi dari model sederhana.

AdaBoost termasuk dalam kategori *discrete boosting* yang menggunakan *weak classifier* berbasis *decision stump* (pohon keputusan sederhana dengan satu cabang) atau *weak learner* lainnya. Algoritma *boosting* sendiri adalah pendekatan untuk menggabungkan banyak *weak learner* dengan pembobotan tertentu agar menghasilkan *strong classifier*. Dalam konteks ini, *AdaBoost* disebut sebagai salah satu metode boosting paling klasik dan berpengaruh, serta menjadi dasar berkembangnya berbagai varian *boosting* lain seperti *Gradient Boosting* (Friedman, 2001) dan *XGBoost* (Chen & Guestrin, 2016).

Menurut Freund dan Schapire (1996), *AdaBoost* mampu meningkatkan performa model secara signifikan meskipun hanya menggunakan *weak learner* sederhana, seperti decision stump. Algoritma ini memiliki dasar teori yang kuat karena secara matematis terbukti dapat menurunkan tingkat kesalahan klasifikasi dengan menggabungkan beberapa model lemah menjadi satu *strong classifier*.

Penelitian dari Friedman (2001)., menambahkan bahwa keunggulan utama *AdaBoost* terletak pada kemampuannya mengurangi bias dan variance secara bersamaan, sehingga menghasilkan generalisasi yang lebih baik pada data baru. Selain itu, algoritma ini relatif sederhana, fleksibel, dan dapat diterapkan pada berbagai domain, mulai dari klasifikasi teks, pengenalan wajah, hingga bioinformatika (Sharma & Dey, 2012).

Boosting merupakan salah satu teknik *ensemble learning* yang paling powerful dalam machine learning. Konsep dasar *boosting* adalah menggabungkan banyak model lemah (*weak learner*) secara bertahap menjadi satu model kuat (*strong learner*). Berbeda dengan teknik ensemble lain seperti bagging yang melatih model secara paralel dan independen, boosting melatih model secara sekvensial (berurutan). Setiap model baru dibangun untuk memperbaiki kesalahan yang dibuat oleh model sebelumnya.

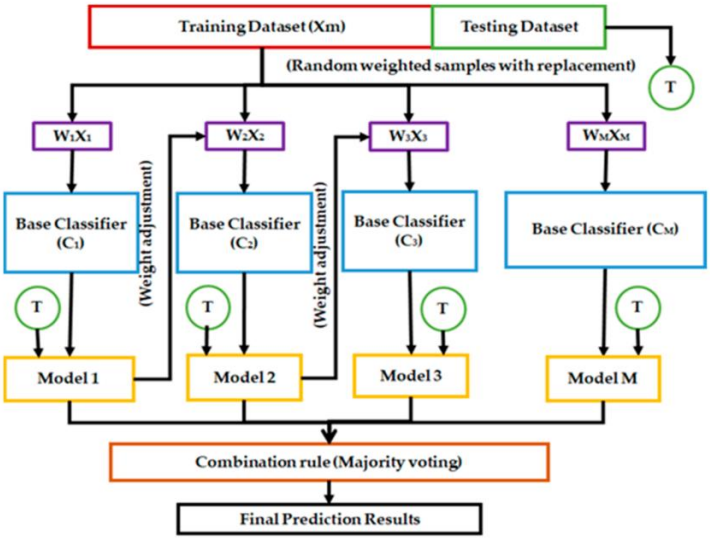
AdaBoost (Adaptive Boosting) adalah algoritma *boosting* klasik yang pertama kali diperkenalkan oleh Freund dan Schapire (1997). Kata "Adaptive" merujuk pada kemampuan algoritma ini untuk menyesuaikan diri secara adaptif terhadap tingkat kesulitan data. Pada setiap iterasi, AdaBoost meningkatkan bobot (*weight*) pada sampel data yang salah diklasifikasikan oleh model sebelumnya, sehingga model berikutnya lebih fokus memperbaiki kesalahan tersebut.

Kelebihan lain dicatat oleh Chen dan Guestrin (2016), yaitu bahwa *AdaBoost* menjadi dasar bagi pengembangan varian boosting modern seperti *Gradient Boosting* dan *XGBoost,* yang berarti perannya sangat penting dalam perkembangan *ensemble learning*. Dengan kata lain, keberhasilan *AdaBoost* menjadikannya sebagai fondasi bagi algoritma *boosting* generasi selanjutnya.

Meskipun memiliki banyak kelebihan, *AdaBoost* juga memiliki keterbatasan. Salah satu kelemahan yang banyak disorot adalah sensitivitasnya terhadap outlier dan noise. Penelitian Freund dan Schapire (1996) menegaskan bahwa jika dalam data terdapat banyak

outlier, maka bobot sampel ini akan terus meningkat karena selalu salah diklasifikasikan, sehingga dapat merusak kinerja model secara keseluruhan. Penelitian Luo (2016) menambahkan bahwa *AdaBoost* kurang optimal ketika diterapkan pada dataset yang tidak seimbang, karena algoritma ini cenderung lebih fokus pada kelas mayoritas yang mendominasi data. Hal ini dapat menyebabkan penurunan performa terutama dalam klasifikasi dengan data yang distribusinya tidak seimbang.

Selain itu, menurut Tewari et al. (2021), kelemahan lainnya adalah kebutuhan iterasi yang relatif banyak. Jika tidak dikendalikan dengan baik, hal ini dapat membuat proses komputasi menjadi kurang efisien. Meskipun demikian, dibandingkan dengan metode ensemble lain pada zamannya, *AdaBoost* tetap dianggap efisien dan berpengaruh.



Gambar 3. Alur Kerja Metode *AdaBoost* (Tewari et al, 2021).

Gambar 3 menjelaskan bahwa *AdaBoost* termasuk ke dalam kategori *discrete boosting*, di mana prinsip utamanya adalah melatih model lemah secara berurutan. Setiap model berikutnya akan difokuskan pada data yang sebelumnya salah diklasifikasikan oleh model sebelumnya. Pada awal proses pelatihan, seluruh data diberikan bobot yang sama. Selanjutnya, *weak learner* pertama dilatih dan dievaluasi tingkat kesalahannya. Data yang salah diprediksi kemudian

diberikan bobot lebih tinggi, sedangkan data yang benar diklasifikasikan dengan bobot yang diturunkan. Dengan demikian, *weak learner* berikutnya akan lebih memperhatikan data yang sulit diklasifikasikan. Alur kerja metode *AdaBoost* yang membangun serangkaian *base classifier* secara bertahap. C1, C2, dan C3 merupakan *base classifier* pada tiga iterasi awal. Sedangkan Cm merupakan notasi umum yang menunjukkan *base classifier* pada iterasi ke-m (m = 1,2,…,T), di mana T adalah jumlah iterasi maksimum yang ditentukan. Setiap *base classifier* Cm yang dibangun berbeda satu sama lain karena dilatih menggunakan distribusi bobot data yang telah diperbarui dari iterasi sebelumnya. Semakin tinggi nilai m, semakin fokus *classifier* tersebut pada sampel-sampel yang sulit diklasifikasikan oleh *classifier* sebelumnya. Proses pembaruan bobot ini berlangsung secara iteratif hingga mencapai jumlah putaran atau iterasi yang ditentukan. Pada akhirnya, prediksi akhir diperoleh dengan menggabungkan seluruh *weak learner* melalui kombinasi berbobot. Untuk kasus klasifikasi, penggabungan ini dilakukan dengan mekanisme *weighted voting*, sedangkan pada regresi digunakan *weighted sum*. Bobot dari setiap weak learner ditentukan berdasarkan tingkat akurasi yang dicapai, sehingga model dengan performa lebih baik akan memberikan kontribusi lebih besar pada hasil akhir.

Selain itu, penelitian terkini yang dilakukan oleh Hussain & Zaidi (2024) menunjukkan bahwa *AdaBoost* memiliki fleksibilitas tinggi karena dapat digabungkan dengan berbagai jenis *base learners* dan mampu bekerja dengan baik pada dataset yang bervariasi, termasuk yang memiliki tingkat ketidakseimbangan kelas (*imbalanced data*). Keunggulan lainnya adalah algoritma ini relatif mudah diimplementasikan, tidak terlalu rentan terhadap *overfitting* pada dataset bersih, serta dapat digunakan baik untuk klasifikasi maupun regresi

Dalam konteks penelitian ini, *AdaBoost* dipilih karena diharapkan dapat meningkatkan akurasi klasifikasi, mengurangi kesalahan prediksi, serta memberikan hasil yang lebih stabil dibandingkan dengan metode pembelajaran tunggal. Selain itu, pemilihan *AdaBoost* juga didasari oleh bukti empiris dari berbagai studi sebelumnya yang menunjukkan performanya unggul dalam berbagai domain, mulai dari diagnosis kerusakan mesin (Hussain & Zaidi, 2024), penanganan data tidak seimbang (Li et al., 2019), hingga analisis data medis (Sevinç, 2022).

Dalam proses *AdaBoost*, terdapat beberapa tahapan penting yang menjadi dasar pembaruan bobot data pelatihan dan pemilihan model lemah untuk meningkatkan akurasi prediksi. Pertama, pada setiap iterasi proses *boosting*, algoritma menghitung tingkat kesalahan model lemah (*weak learner*) yang digunakan pada iterasi tersebut. Selanjutnya, bobot masing-masing data pelatihan diperbarui berdasarkan tingkat kesalahan tersebut sehingga data yang salah diprediksi akan memperoleh bobot lebih besar pada iterasi berikutnya.

Dengan cara ini, *weak learner* berikutnya lebih berfokus pada data yang sulit diprediksi. Rumus perhitungan pembaruan bobot dan penentuan kontribusi model lemah dapat dilihat pada persamaan berikut.

Inisialisasi bobot pada persamaan 1 sampai dengan persamaan 6 :

$$w_i^{(1)} = \frac{1}{N}, \quad i = 1, \dots, N \quad \dots\dots\dots\dots\dots (1)$$

Langkah iterasi untuk $t = 1, \dots, T$:

Latihan *weak learner* $h_t(x)$ menggunakan distribusi bobot $w^{(t)}$.

Hitung *error* :

$$\varepsilon_t = \sum_{i=1}^{N} w_i^{(t)} 1\{h_t(x_i) \neq y_i\} \quad \dots\dots\dots\dots\dots (2)$$

Hitung bobot *weak learner* :

$$\alpha_t = \frac{1}{2} In\left(\frac{1-\varepsilon_t}{\varepsilon_t}\right) \quad \dots\dots\dots\dots\dots\dots\dots (3)$$

Perbarui bobot sampel (*unnormalized* - normalisasi) :

$$w_i^{(t+1)} = w_i^{(t)} \exp\left(-\alpha_t y_i h_t(x_i)\right), \quad \text{..... (4)}$$

$$Z_t = \sum_{j=1}^{N} w_j^{(t+1)}, \qquad w_i^{(t+1)} = \frac{w_i^{(t+1)}}{Z_t} \quad \text{..... (5)}$$

Hasil akhir (*strong classifier*) :

$$H(x) = sign\left(\sum_{t=1}^{T} \alpha_t \; h_t(x)\right) \quad \text{..... (6)}$$

(Freund & Schapire, 1997).

Keterangan :

$w_i^{(1)}$: bobot awal sampel ke -i.

N : jumlah total sampel.

t : indeks iterasi boosting

$ht(x)$: *weak learner* (misalnya *decision stump*).

yi Label sebenarnya dari sampel ke-i.

ε_t : error *weak learner* ke-t.

$1\{\}$ **:** Fungsi indikator (1 jika salah, 0 jika benar).

α_t : bobot *weak learner* ke-t.

$w_j^{(t+1)}$: bobot sementara sebelum normalisasi.

Z_t : faktor normalisasi

$w_i^{(t+1)}$: Bobot sampel setelah normalisasi.

$H(x)$: *strong classifier* hasil akhir.

Rumus diatas menjelaskan bahwa pada awal, semua sampel dianggap sama pentin $w_i^{(1)} = \frac{1}{N}$. Setiap kali *weak learner* dilatih, kita menghitung errornya. Weak learner yang lebih bagus akan mendapat bobot lebih besar α_t. Setelah itu, bobot sampel diperbarui: sampel yang salah prediksi akan naik bobotnya (jadi lebih penting di iterasi berikutnya), sedangkan yang benar akan turun bobotnya. Proses ini diulang hingga T iterasi, lalu hasil semua *weak learner* digabungkan menjadi *strong classifier*.

Untuk menampilkan semua langkah perhitungan, diberikan contoh numerik, yaitu dataset sederhana dengan lima sampel, agar perhitungan dapat dilakukan secara manual. *Weak learner* pertama $h1$ Masih melakukan kesalahan pada beberapa data. Setelah bobot diperbarui, sampel yang salah memperoleh bobot lebih besar. *Weak learner* kedua $h2$ Kemudian diarahkan untuk lebih fokus pada sampel yang sulit tersebut.

Tabel 3. Contoh Numerik (Freund & Schapire, 1997).

I	**Label *yi***
1	+1
2	+1
3	-1
4	-1
5	+1

Tabel 3 menyajikan contoh sederhana untuk memahami cara kerja *AdaBoost* secara langkah demi langkah pada dataset kecil yang terdiri dari 5 sampel. Contoh ini diambil dari Freund & Schapire (1997) dan bertujuan menunjukkan bagaimana bobot sampel berubah seiring iterasi, sehingga model semakin fokus pada sampel yang sulit diklasifikasikan.

a) Label yi : kelas sebenarnya (+1 atau -1).
b) h1(x) : Prediksi *weak learner* pertama (*decision stump* sederhana).
c) h2(x) : Prediksi *weak learner* kedua.

Dari tabel ini terlihat:

a) *Weak learner* 1 (h1) salah pada sampel 2 dan 4 (karena prediksi -1 dan +1, tetapi label sebenarnya bertolak belakang).
b) *Weak learner* 2 (h2) salah pada sampel 3 dan 4.

Inisialisasi bobot awal tiap sampel pada persamaan 7 :

$w_i^{(1)} = 1/5 = 0.2$ (Untuk I = 1…5). …….... (7)

Weak Learners (prediksi yang dipakai untk contoh) :

Kitas asumsikan dua *weak learner* sederhana (*decision stumps*) dengan prediksi :

$h1(x)$ = prediksi : [+1, -1, -1, +1, +1]

$h2(x)$ = prediksi : [+1, +1, +1, -1, +1]

Berdasarkan persamaan 7 contoh numerik dan prediksi dua *weak learner* sederhana, dapat terlihat bahwa $h1(x)$ Masih melakukan kesalahan klasifikasi pada beberapa sampel. Agar proses pembelajaran *AdaBoost* dapat meningkatkan akurasi, langkah berikutnya adalah menghitung kesalahan (*error*) *weak learner* pertama, menentukan nilai α_1 Sebagai bobot *kepercayaan weak learner* tersebut, dan kemudian memperbarui bobot masing-masing sampel. Perhitungan-perhitungan tersebut ditunjukkan secara rinci pada perhitungan iterasi 1 (t = 1), sehingga dapat terlihat bagaimana bobot sampel yang salah diklasifikasikan menjadi lebih besar dan memengaruhi pembelajaran *weak learner* berikutnya.

Iterasi 1 :

Hitung kesalahan ε_1 :

Sampel yang salah oleh $h1$: index 2 dan 4. Karena bobot awal 0,2 :

ε_1 = 0,2 + 0,2 = 0,4

Hitung α_1 :

$$\alpha_1 = \frac{1}{2}\ln\left(\frac{1-0.4}{0.4}\right) = \frac{1}{2}\ln(1.5) \approx 0.202732554$$

Perbarui bobot (*unnormalized*) :

Faktor untuk sampel benar : $exp(-\alpha_1) \approx 0.81649658$

Faktor untuk sampel salah : $exp(+\alpha_1) \approx 1.225$

Unnormalized :

Untuk sampel benar : 0,2 x 0,81649658 = 0,163299316

Untuk sampel salah : 0,2 x 1,225 = 0,245099316

Normalisasi Z_1 :

$Z_1 = 3 \times 0.163299316 + 2 \times 0.24509931 = 0.98009658$

Bobot baru $w^{(2)}$:

Jika benar : $\frac{0.163299316}{0.98009658} \approx 0.1666667\ (\approx 1/6)$

Jika salah : $\frac{0.245099316}{0.98009658} \approx 0.25$

Jadi bobot setelah iterasi 1 :

Tabel 4. Bobot Iterasi 1.

I	Y	h1	*misclassified*?	$w_i^{(2)}$
1	+1	+1	0	0.1666667
2	+1	-1	1	0.2500000
3	-1	-1	0	0.1666667
4	-1	+1	1	0.2500000
5	+1	+1	0	0.1666667

Pada iterasi pertama:

Error $h1$ = 0.4 → tidak terlalu bagus.

α_1 ≈ 0.20α1≈0.20 → kontribusi weak learner 1 ke model akhir masih kecil.

Bobot sampel yang salah naik ke 0.25, sedangkan yang benar turun ke 0.1667. Artinya, sekarang model akan lebih memperhatikan sampel 2 dan 4 (yang salah tadi).

Berdasarkan penjelasan di atas, telah dilakukan perhitungan kesalahan ε_1, pembobotan awal, serta pembaruan bobot $w^{(2)}$ setelah iterasi pertama pada algoritma *AdaBoost*. Tabel bobot hasil iterasi pertama menunjukkan distribusi bobot baru yang lebih besar pada sampel yang salah diklasifikasikan oleh weak learner pertama h1. Bobot yang diperoleh ini selanjutnya digunakan sebagai dasar untuk melatih *weak learner* kedua $h2$.

Proses ini kemudian dilanjutkan dengan menampilkan langkah-langkah iterasi kedua. Pada iterasi ini dihitung kembali kesalahan (ε_2) yang diperoleh *weak learner* kedua $h2$, nilai α_2 Ditentukan, dan pembaruan bobot dilakukan sehingga sampel yang masih salah klasifikasi kembali mendapat bobot lebih tinggi. Dengan demikian, terlihat bagaimana *AdaBoost* secara bertahap memfokuskan pembelajaran pada sampel-sampel yang lebih sulit diklasifikasikan dengan benar. Berikut perhitungan iterasi 2.

Iterasi 2 :

Hitung keasalahan ε_2 :

Sampel yang salah oleh $h2(3) = +1, padahal\ y3 = -1)$

$$\varepsilon_2 = w_3^{(2)} = 0.1666667$$

Hitung α_2 :

$$\alpha_2 = \frac{1}{2}\ln\left(\frac{1-\frac{1}{6}}{\frac{1}{6}}\right) = \frac{1}{2}\ln(5) \approx 0.804718956$$

Perbarui bobot (*unnormalized*) :

Faktor benar : $exp(+\alpha_2) \approx 0{,}804718956$

Faktor salah : $exp(+\alpha_2) \approx 0{,}223606977$

Unnormalized $\widetilde{w}$:

i = 1 : 0,1666667 × 0,4472136 ≈ 0,0745356

i=2 : 0,2500000 × 0,4472136 ≈ 0,1118034

i=3 (salah) : 0,1666667 × 2,236068 ≈ 0,3726780

i=4 : 0,2500000 × 0,4472136 ≈ 0,1118034

i=5 : 0,1666667 × 0,4472136 ≈ 0,0745356

Normalisasi (Z_2) :

$$Z_2 = 0.0745356 + 0.1118034 + 0.3726780 + 0.1118034 + 0.0745356 = 0.745356$$

Bobot akhir $w^{(3)}$:

Tabel 5. Bobot Iterasi 2.

I	$\boldsymbol{w_i^{(3)}}$
1	0.0745356 / 0.745356 ≈ 0.1000
2	0.1118034 / 0.745356 ≈ 0.1500
3	0.3726780 / 0.745356 ≈ 0.5000
4	0.1118034 / 0.745356 ≈ 0.1500
5	0.0745356 / 0.745356 ≈ 0.1000

(Jumlah ≈ 1.0000. Sampel 3 kini diberi bobot dominan (0.5) → menandakan bahwa masih sulit).

Weak learner kedua diuji. Karena bobotnya sudah fokus, $h2$ Hanya salah di sampel 3.

Error turun drastis ke 0.1667.

α_2 ≈0.80 → kontribusinya jauh lebih besar.

Setelah *update*, sampel 3 menjadi sangat dominan (bobotnya 0,5).

Artinya: sekarang algoritma menaruh perhatian utama pada sampel 3 karena selalu salah diklasifikasikan.

Hasil akhir (gabungan *weak learner)*

Kombinasi berbobot persamaan 8 :

$$\text{score}_i = \alpha_1 h_1(x_i) + \alpha_2 h_2(x_i) \quad \text{…….. (8)}$$

Dengan $\alpha_1 \approx 0.2027326, \quad \alpha_2 \approx 0.8047190$

Berikut, Tabel 6 di bawah menunjukkan bagaimana skor akumulasi dari dua *weak learner* pertama menghasilkan keputusan akhir *ensemble* H(x). Perhatikan bahwa weak learner kedua (h_2) berhasil mengoreksi kesalahan pada sampel yang bobotnya dinaikkan di iterasi sebelumnya, sehingga meningkatkan performa keseluruhan.

Tabel 6. Perhitungan Skor Persampel.

I	**h1**	**h2**	**score** $= \boldsymbol{\alpha_1 h_1 + \alpha_2 h_2}$	$\boldsymbol{H(x) = sign(\alpha_1 h_1(x) + \alpha_2 h_2(x))}$	**Label y**	**Benar?**
1	+1	+1	0.2027×1 + 0.8047×1 = 1.0074	+1	+1	√
2	-1	+1	–0.2027 + 0.8047 = 0.6020	+1	+1	√
3	-1	+1	–0.2027 + 0.8047 = 0.6020	+1	-1	×
4	+1	-1	0.2027 – 0.8047 = –0.6020	-1	-1	√
5	+1	+1	1.0074	+1	+1	√

Hasil : *Strong classifier* salah pada sampel 3 saja.

Ketika digabung dengan bobot α :

Hampir semua sampel benar kecuali sampel 3.

Model akhir hanya salah di satu titik → akurasi tinggi meski awalnya weak learner sederhana.

Ini menunjukkan kekuatan *AdaBoost*: meskipun tiap *weak learner* lemah, gabungan mereka bisa jadi kuat.

Langkah-langkah rumus yang dipakai di atas adalah rumus klasik *AdaBoost* (Freund & Schapire). Pendekatan numerik (menghitung ε_t, α_t, update bobot melalui factor $\exp(-\alpha_t y_i h_t(x_i))$, dan normalisasi Z_t) adalah standar dalam hampir semua penerapan *AdaBoost* yang

dilaporkan di literatur (Sevinç, 2022; Li et al., 2019; Hussain & Zaidi, 2024).

Contoh numerik menunjukkan karakter khas *AdaBoost*: sampel yang sering salah akan menerima bobot besar (di contoh ini sampel 3 memuncak ke bobot 0,5 setelah iterasi ke-2), dan weak learners yang lebih akurat mendapat bobot α_t Lebih besar sehingga memengaruhi prediksi akhir lebih kuat.

Kekurangan yang terlihat juga dari contoh: sampel yang merupakan *outlier* atau sangat sulit dapat tetap memaksa bobot menjadi besar dan, apabila ia benar-benar *outlier*, ini bisa merugikan generalisasi (kekhasan yang dijelaskan pada literatur).

Bobot awal $w^{(1)}$ Pada kelima data sampel diberikan nilai yang sama, yaitu sebesar 0,2000. Setelah dilakukan pembaruan bobot pada iterasi pertama, diperoleh bobot baru $w^{(2)}$ Dengan rincian: data ke-1 sebesar 0,1667; data ke-2 sebesar 0,2500; data ke-3 sebesar 0,1667; data ke-4 sebesar 0,2500; dan data ke-5 sebesar 0,1667.

Selanjutnya, setelah pembaruan bobot pada iterasi kedua, bobot $w^{(3)}$ menjadi: data ke-1 sebesar 0,1000; data ke-2 sebesar 0,1500; data ke-3 sebesar 0,5000; data ke-4 sebesar 0,1500; dan data ke-5 sebesar 0,1000. Perubahan bobot pada setiap iterasi ini menunjukkan bahwa sampel yang diklasifikasikan secara keliru oleh model sebelumnya akan memperoleh bobot yang lebih besar pada iterasi berikutnya, sehingga algoritma lebih fokus untuk memperbaiki kesalahan pada data-data tersebut.

2.2.11 XGBoost

Extreme Gradient Boosting atau *XGBoost* pertama kali diperkenalkan oleh Tianqi Chen dan Carlos Guestrin (2016) sebagai pengembangan dari algoritma *Gradient Boosting Decision Tree* (GBDT) yang sebelumnya diperkenalkan oleh Friedman (2001). Kehadiran *XGBoost* menjadi terobosan penting karena mampu mengatasi kelemahan *boosting* klasik yang rentan terhadap *overfitting*

serta memiliki keterbatasan dalam efisiensi komputasi. Sejak diperkenalkan, *XGBoost* dengan cepat mendapatkan popularitas luas, baik di dunia akademik maupun praktis, karena kinerjanya yang konsisten unggul pada berbagai kompetisi *machine learning* serta aplikasinya di banyak bidang, termasuk bioinformatika (Chen & Guestrin, 2016).

XGBoost merupakan algoritma *ensemble learning* berbasis *gradient boosting* yang bekerja dengan menggabungkan sejumlah *weak learners* berupa pohon keputusan sederhana untuk membentuk sebuah *strong learner* yang lebih akurat. Sama seperti algoritma boosting lainnya, proses ini dilakukan secara bertahap, setiap pohon baru dibangun untuk memperbaiki kesalahan prediksi pohon sebelumnya. Namun, *XGBoost* menghadirkan sejumlah inovasi penting, seperti penerapan regularisasi L1 dan L2 untuk mengendalikan kompleksitas model, penggunaan teknik subsampling baris maupun kolom untuk mengurangi *overfitting*, serta optimasi berbasis gradien kedua (*second-order optimization*) yang membuat proses pelatihan lebih cepat dan stabil (Chen & Guestrin, 2016).

Dalam praktiknya, *XGBoost* mengenal beberapa istilah penting. *Weak learner* merujuk pada pohon keputusan sederhana yang digunakan secara berulang, sementara *boosting* adalah proses penggabungan model-model tersebut untuk meningkatkan akurasi prediksi. Fungsi objektif dalam *XGBoost* terdiri dari gabungan *loss function* dan regularisasi, sedangkan *learning rate* atau faktor *shrinkage* digunakan untuk mengontrol kontribusi tiap pohon baru terhadap model. Selain itu, parameter seperti *max depth* mengatur kedalaman maksimal pohon, *subsampling* berfungsi mencegah *overfitting*, dan *feature importance* digunakan untuk mengukur kontribusi relatif tiap fitur terhadap hasil prediksi.

Sejumlah penelitian telah menegaskan efektivitas *XGBoost* dalam berbagai domain. Penelitian Zhou et al. (2023) menunjukkan bahwa *XGBoost* mampu menangani data biologis kompleks sekaligus

menyediakan interpretabilitas fitur yang penting untuk riset genomik. Penelitian Wang et al. (2022) membuktikan bahwa *XGBoost* lebih unggul dibandingkan *Random Forest* dan SVM dalam analisis protein multi-organisme, dengan nilai AUC yang lebih tinggi. Penelitian Liu et al. (2023) menegaskan bahwa *XGBoost* menonjol dalam klasifikasi dataset besar dengan identifikasi fitur signifikan, sedangkan penelitian Alonso et al. (2024) melaporkan bahwa *XGBoost* lebih reliabel dibandingkan metode klasik karena menghasilkan AUC dan akurasi yang konsisten tinggi pada domain pendidikan. Selain itu, penelitian di bidang genomik oleh Chen et al. (2023) memperlihatkan bahwa penggabungan *XGBoost* dengan algoritma optimasi mampu meningkatkan prediksi genomik secara signifikan.

XGBoost atau *Extreme Gradient Boosting* merupakan pengembangan lanjutan dari algoritma Gradient Boosting yang diperkenalkan oleh Chen dan Guestrin (2016). XGBoost bekerja dengan prinsip boosting, yaitu melatih serangkaian model lemah secara berurutan di mana setiap model baru dibangun untuk memperbaiki kesalahan model sebelumnya. Kata "Gradient" merujuk pada penggunaan turunan pertama (gradient) dan turunan kedua (Hessian) dari fungsi loss untuk melakukan optimasi yang lebih akurat dan efisien. Sementara itu, "Extreme" menunjukkan berbagai ekstrem optimasi yang diterapkan, seperti komputasi paralel berkecepatan tinggi, regularisasi L1 dan L2 yang kuat untuk mencegah overfitting, teknik subsampling, serta penanganan data berskala besar dengan lebih baik dibandingkan boosting tradisional.

XGBoost memiliki keunggulan utama berupa kecepatan, skalabilitas, dan performa tinggi, sehingga menjadikannya salah satu algoritma paling populer untuk tugas klasifikasi pada data kompleks seperti sekuens biologis.

Dari sisi keunggulan, *XGBoost* memiliki sejumlah kelebihan yang menjadikannya salah satu algoritma paling banyak digunakan. Regularisasi L1 dan L2 membuatnya lebih tahan terhadap *overfitting*,

dukungan komputasi paralel mempercepat pelatihan pada dataset besar, serta pendekatan *sparsity-aware* memungkinkan penanganan data dengan nilai hilang maupun fitur jarang (Zhou et al., 2023). Lebih jauh, interpretabilitas melalui analisis *feature importance* menjadi nilai tambah penting dalam penelitian biologis untuk menelusuri motif DNA atau sifat protein yang relevan (Liu et al., 2023). Namun demikian, *XGBoost* juga memiliki keterbatasan, seperti kebutuhan sumber daya komputasi yang lebih besar dibandingkan dengan algoritma *boosting* sederhana, kompleksitas *hyperparameter tuning* yang memerlukan eksperimen intensif, serta sensitivitas terhadap ketidakseimbangan kelas jika tidak dilakukan penyesuaian data (Alonso et al., 2024).

Gambar 4. Mekanisme Alur Kerja Metode *XGBoost* (Chen et al, 2016).

Gambar 4 merupakan mekanisme kerja *XGBoost* yang dimulai dengan inisialisasi prediksi awal, biasanya berupa rata-rata dari label target. Selanjutnya, algoritma menghitung gradien (turunan pertama) dan hessian (turunan kedua) dari fungsi kerugian untuk setiap data,

yang kemudian digunakan untuk membangun pohon keputusan baru untuk memprediksi residu dari kesalahan sebelumnya. Pohon ini diatur dengan regularisasi agar tidak terlalu kompleks. Setelah pohon baru ditambahkan, model diperbarui dengan mempertimbangkan *learning rate* sebagai pengendali kontribusi pohon terhadap prediksi akhir. Proses ini dilakukan secara iteratif hingga jumlah pohon maksimum tercapai atau kesalahan prediksi sudah konvergen. Fungsi objektif yang dioptimalkan dapat dituliskan sebagai berikut:

$$Obj(t) = \sum_{i=1}^{n} l\left(y_i, y_l^{(t-1)} + f_t(x_i)\right) + \Omega(f_t) \dots (9)$$

di mana l adalah fungsi kerugian, $y_l^{(t-1)}$ prediksi sebelumnya, f_t adalah pohon baru, dan $\Omega(f_t)$ Adalah regularisasi untuk mengendalikan kompleksitas.

Keterangan :

l : Fungsi kerugian (loss function) mengukur selisih antara label aktual dan prediksi.

y_i : nilai target aktual untuk sampel ke-i.

$y_l^{(t-1)}$: prediksi pada iterasi sebelumnya (t – 1).

f_t : pohon keputusan baru (*new tree*) yang ditambahkan pada iterasi ke-t untuk memperbaiki error.

x_i : Fitur/input dari data ke-i.

$\Omega(f_t)$: Fungsi regularisasi yang mengontrol kompleksitas pohon melalui penalti L1 (Lasso) dan L2 (Ridge).

Selanjutnya, *XGBoost* menggunakan pendekatan deret Taylor orde kedua untuk melakukan aproksimasi fungsi kerugian pada persamaan 10 :

$$Obj^{(t)} \approx \sum_{i=1}^{n} \left[g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)\right] + \Omega(f_t) \dots (10)$$

Keterangan :

y_i : label sebenarnya (0/1 untuk klasifikasi biner).

$y_l^{(t-1)}$: prediksi model sampai iterasi t − 1.

$f_t(x_i)$: kontribusi pohon baru pada iterasi t.

$g_i = \frac{\partial}{\partial \hat{y}} l(y_i, y)|_{y= y_l^{(t-1)}}$: gradient (*first derivative*).

$h_i = \frac{\partial^2}{\partial \widehat{y^2}} l(y_i, y)|_{y= y_l^{(t-1)}}$: hessian (*second derivative*).

$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$: regularisasi pohon (contoh formulasi *XGBoost*: penalti jumlah daun T, parameter λ untuk bobot daun w_j, dan parameter γ Untuk split). (Chen & Guestrin, 2016)

Dengan g_i adalah gradien (turunan pertama fungsi *loss*) dan h_i Adalah Hessian (turunan kedua fungsi *loss*). Gradien menunjukkan arah koreksi yang perlu dilakukan, sedangkan Hessian menggambarkan tingkat kelengkungan fungsi kerugian. Menurut Zhou et al. (2023) penggunaan gradien dan Hessian secara bersamaan membuat optimasi XGBoost lebih stabil pada data biologis berskala besar. Berikut formulasi perhitungan yang dipaparkan pada persaman 11 dan Persamaan 12.

Dari formulasi di atas, bobot optimal untuk sebuah daun (*leaf*) dihitung :

$$w^* = -\frac{\sum_i g_i}{\sum_i h_i + \lambda} \quad \text{..... (11)}$$

Dan *split gain* (pengurangan loss akibat split) untuk dua anak kiri-kanan:

$$Gain = \frac{1}{2}\left(\frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{(G_L+G_R)^2}{H_L+H_R+\lambda}\right) - \lambda \quad \text{..... (12)}$$

Dengan $G_L = \sum_{i\in L} g_i$, $H_L = \sum_{i\in L} h_i$, dan analog untuk anak kanan R. (Chen & Guestrin, 2016)

Sebagai ilustrasi sederhana, misalkan terdapat dataset dengan tiga sampel DNA yang ingin diklasifikasikan menjadi gen esensial (1) atau non-esensial (0). Data input berupa representasi numerik sederhana :

$$X = [0.20,\ 0.60,\ 0.90], \quad Y = [0,\ 1,\ 1]$$

Prediksi awal (*baseline*) dipilih sebagai rata-rata label (probabilitas awal):

$$y^{(0)} = \frac{0+1+1}{3} = \frac{2}{3} \approx 0.6667$$

Hitung gradien dan hessian menggunakan *logistic loss* (untuk klasifikasi biner, gradien pada persamaan 13 :

$$g_i = p_i - y_i, \qquad h_i = p_i(1-p_i), \qquad p_i = y^{(0)} \quad .. (13)$$

Karena $p = \frac{2}{3}$:

$g_1 = \frac{2}{3} - 0 = \frac{2}{3} \approx 0.6667$,

$h_1 = \frac{2}{3}\left(1 - \frac{2}{3}\right) = \frac{2}{9} \approx 0.2222$

$g_2 = \frac{2}{3} - 1 = -\frac{1}{3} \approx -0.3333$,

$h_2 = \frac{2}{3}\left(1 - \frac{2}{3}\right) = \frac{2}{9} \approx 0.2222$

$g_3 = \frac{2}{3} - 1 = -\frac{1}{3} \approx -0.3333$,

$h_3 = \frac{2}{3}\left(1 - \frac{2}{3}\right) = \frac{2}{9} \approx 0.2222$

Tabel 7. Tabel Ringkas Gradien dan Hessian (Zhou et al., 2023).

Sampel	y_i	$\boldsymbol{p = y^{(0)}}$	$\boldsymbol{g_i = p - y_i}$	$\boldsymbol{h_i = p(1-p)}$
1	0	0.6666667	0.6666667	0.2222222
2	1	0.6666667	-0.3333333	0.2222222
3	1	0.6666667	-0.3333333	0.2222222

(angka dibulatkan ke 7 desimal untuk kejelasan)

Contoh struktur pohon sederhana :

Misal split pada threshold x = 0,5 maka :

a) Kiri (L) : sampel 1 (X = 0,20)
b) Kanan (R) : sampel 2 dan 3 (X = 0,60 dan 0,90)

Hitung jumlah gradien dan hessian :

$$G_L = \sum_{i \in L} g_i = \frac{2}{3}, \quad H_L = \sum_{i \in L} h_i = \frac{2}{9}$$

$$G_R = g_2 + g_3 = -\frac{1}{3} + -\frac{1}{3} = -\frac{2}{3}$$

$$H_R = h_2 + h_3 = \frac{2}{9} + \frac{2}{9} = \frac{4}{9}$$

Ambil contoh parameter regulasi $\lambda = 1$ dan $\gamma = 0$ untuk ilustrasi. Bobot daun (*leaf weight*) dihitung :

$$w_L^* = -\frac{G_L}{H_L + \lambda} = -\frac{\frac{2}{3}}{\frac{2}{9} + 1} = -\frac{2}{3} \cdot \frac{9}{11} = -\frac{6}{11} \approx -0.5454545$$

$$w_R^* = -\frac{G_R}{H_R + \lambda} = -\frac{-\frac{2}{3}}{\frac{4}{9} + 1} = \frac{\frac{2}{3}}{\frac{13}{9}} = \frac{2}{3} \cdot \frac{9}{13} = \frac{6}{13} \approx 0.4615385$$

Interpretasi: daun kiri mendapat bobot negatif (mengurangi skor prediksi di region itu), daun kanan bobot positif.

Perhitungan *split gain* (seberapa baik split tersebut mengurangi *objective*) pada persamaan 13 :

$$\text{Gain} = \frac{1}{2}\left(\frac{G_L^2}{H_L+\lambda} + \frac{G_R^2}{H_R+\lambda} - \frac{(G_L+G_R)^2}{H_L+H_R+\lambda}\right) - \gamma \quad \text{..... (13)}$$

Karena $G_L + G_R = 0$, ketiga suku jadi :

$$\frac{G_L^2}{H_L + \lambda} = \frac{\left(\frac{2}{3}\right)^2}{\frac{2}{9} + 1} = \frac{\frac{4}{9}}{\frac{11}{9}} = \frac{4}{11} \approx 0.36363636$$

$$\frac{G_R^2}{H_R + \lambda} = \frac{\left(-\frac{2}{3}\right)^2}{\frac{4}{9} + 1} = \frac{\frac{4}{9}}{\frac{13}{9}} = \frac{4}{13} \approx 0.30769231$$

$$\frac{(G_L + G_R)^2}{H_L + H_R + \lambda} = 0$$

Maka :

$$\text{Gain} = \frac{1}{2}\left(\frac{4}{11} + \frac{4}{13}\right) = \frac{1}{2} \cdot \frac{96}{143} = \frac{48}{143} \approx 0.33566434$$

Karena *gain* > 0 (dengan γ =), split tersebut dianggap menguntungkan menurut kriteria *XGBoost*; bila γ > 0, perlu dibandingkan dengan ambang γ.

Nilai gradien dan hessian serupa diperoleh untuk dua sampel lain. Berdasarkan nilai ini, algoritma membangun pohon keputusan pertama yang meminimalkan *loss*. Regularisasi $\Omega(f_t)$ Digunakan

untuk memangkas cabang yang kurang relevan, sehingga pohon tidak menjadi terlalu kompleks. Proses ini kemudian diulang: model diperbarui dengan menambahkan pohon baru yang dikalikan dengan faktor *learning rate* untuk mengontrol besarnya kontribusi.

Hyperparameter tuning seperti *max depth*, *learning rate*, dan *subsample* terbukti sangat menentukan kualitas hasil akhir *XGBoost* (Bentejac et al., 2021). *Max_depth* mengontrol kedalaman maksimum setiap pohon keputusan; semakin dalam pohon, semakin kompleks model yang dihasilkan, tetapi juga semakin rentan terhadap *overfitting*. *Learning_rate* (atau *eta*) menentukan besarnya kontribusi setiap pohon baru terhadap prediksi akhir nilai yang lebih kecil umumnya menghasilkan model yang lebih baik meskipun memerlukan lebih banyak iterasi. Sedangkan *subsample* mengatur proporsi sampel data yang digunakan untuk melatih setiap pohon, sehingga berperan penting dalam mengurangi varians dan mencegah overfitting. Sementara studi komparatif oleh Nori et al. (2023) menegaskan bahwa *XGBoost* dengan tuning yang tepat mampu mencapai AUC lebih tinggi dibandingkan dengan metode machine learning lainnya. Menurut Liu et al. (2023) dalam *Sustainability*, fitur penting dapat diidentifikasi dari nilai *feature importance* setelah semua iterasi selesai, yang dalam konteks penelitian gen esensial dapat membantu menentukan atribut DNA atau protein paling berpengaruh.

Dengan demikian, rumus dan mekanisme *XGBoost* tidak hanya memberikan kerangka teoretis, tetapi juga mendukung penerapan praktis pada data biologis. Hal ini selaras dengan hasil Zhou et al. (2023) yang menunjukkan efektivitas *XGBoost* dalam analisis sekuens genomik. Pemilihan *XGBoost* dalam penelitian klasifikasi gen esensial pada *Drosophila melanogaster* didasari oleh sejumlah alasan kuat. Pertama, data sekuens DNA dan protein bersifat kompleks, besar, dan heterogen, sehingga membutuhkan algoritma yang mampu mengatasinya. Kedua, kemampuan interpretabilitas *XGBoost* sangat

membantu dalam mengidentifikasi motif DNA atau sifat fisikokimia protein yang berkontribusi terhadap esensialitas gen (Zhou et al., 2023). Ketiga, akurasi prediksi yang tinggi telah dibuktikan dalam penelitian genomik dan bioinformatika (Wang et al., 2022).

2.2.12 Feature Extraction

Ekstraksi fitur (*feature extraction*) merupakan tahap penting untuk mengubah data mentah menjadi representasi numerik yang dapat dimanfaatkan oleh algoritma pembelajaran mesin. Pada penelitian ini, ekstraksi fitur digunakan untuk menghasilkan representasi sekuens DNA dan protein yang kaya informasi sebelum diproses oleh model boosting seperti *AdaBoost* dan *XGBoost*. Kedua algoritma tersebut sangat bergantung pada kualitas fitur; semakin representatif fitur, semakin baik performa prediksi yang dihasilkan (Sun et al., 2021).

A. *Amino Acid Composition* (AAC)

Amino Acid Composition adalah metode untuk mengubah sekuens protein menjadi vektor numerik yang berisi proporsi masing-masing dari 20 asam amino. Representasi ini independen dari panjang sekuens dan sering digunakan sebagai fitur dasar untuk model pembelajaran mesin, termasuk *AdaBoost* dan XGBoost, karena cepat dihitung dan interpretatif (Bhasin & Raghava, 2004).

Berikut rumus *Amino Acid Compotition* (AAC) dapat dilihat pada persamaan 14 :

$$AAC(i) = \frac{n_i}{L} \times 100\%, \quad i = 1, \ldots, 20 \quad \ldots\ldots\ldots (14)$$

Keterangan :

i : indeks untuk satu dari 20 asam amino (A,R,N,D,C,Q,E,G,H,I,L,K,M,F,P,S,T,W,Y,V).

n_i : jumlah residu asam amino tipe i pada sekuens.

L : panjang sekuens (jumlah residu total).

Hasil $AAC(i)$ dalam persen merupakan proporsi residu tipe i terhadap total.

Fitur AAC ini langsung menjadi masukan (kolom) pada matriks fitur. *AdaBoost* yang memakai weak learner sederhana cocok dengan fitur berdimensi rendah seperti AAC, sedangkan *XGBoost* mampu memanfaatkan AAC bersama fitur lanjutan dengan regularisasi. Berikut contoh pengaplikasian rumus AAC :

Sekuens contoh (Panjang L = 28) :

"M K T W L V L A G G I A A G I A A L G V L I V V S S"

Langkah 1 : Hitung frekuensi 20 asam amino

Tabel 8. Contoh jumlah sekuens Asam Amino (Bhasin & Raghava, 2004).

Residue	**Nama**	n_i
A	Alanine	6
R	Ariginine	0
N	Asparagine	0
D	Aspartate	0
C	Cysteine	0
Q	Glutamine	0
E	Glutamate	0
G	Glycine	5
H	Histidine	0
I	Isoleucine	3
L	Leucine	4
K	Lysine	1
M	Methionine	1
F	Phenylalanine	0
P	Proline	0
S	Serine	2
T	Threonine	1
W	Tryptophan	1
Y	Tyrosine	0
V	Valine	4

Langkah 2 : Hitung AAC untuk setiap residu pada persamaan 15 :

$$AAC(i) = \frac{n_i}{L} \times 100\% \quad \text{..... (15)}$$

Tabel 9. Perhitungan Metode AAC pada Sekuens (Bhasin & Raghava, 2004).

Residue	$AAC(i)\%$
A	21,43
R	0,00

N	0,00
D	0,00
C	0,00
Q	0,00
E	0,00
G	17,86
H	0,00
I	10,71
L	14,29
K	3,57
M	3,57
F	0,00
P	0,00
S	7,14
T	3,57
W	3,57
Y	0,00
V	14,29

Langkah 3 : Bentuk vektor fitur AAC

[21.43,0,0,0,0,0,0,17.86,0,10.71,14.29,3.57,3.57,0,0,7.14,3.57,3.57,0,14.29]

Vektor ini siap dimasukkan ke matriks fitur dan kemudian ke model *AdaBoost* dan *XGBoost.*

B. *Tri–Nucleotid Composition* (TNC)

Tri-Nucleotide Composition adalah metode ekstraksi fitur yang menghitung frekuensi relatif semua kombinasi tiga nukleotida berurutan dalam sekuens DNA. Dengan empat jenis nukleotida (A, C, G, T) terdapat 4^3 = 64 kemungkinan triplet yang masing-masing membentuk elemen vektor fitur berdimensi 64. Fitur TNC sangat sering digunakan pada penelitian genomik karena mampu menangkap motif kodon; pada penelitian ini fitur tersebut dipakai sebagai masukan bagi model boosting *AdaBoost* dan *XGBoost* (Zhang et al., 2024).

Rumus *Tri – Nucleotid Composition* (TNC) pada persamaan 16 :

$$TNC(t) = \frac{C_t}{t} \times 100\% \quad \text{..... (16)}$$

Keterangan :

t : satu jenis *triplet* dari 64 kemungkinan.

C_t : jumlah kemunculan triplet t pada sekuens DNA.

T : total *triplet* dalam sekuens (= panjang sekuens − 2).

Hasil $TNC(t)$ menyatakan persentase kemunculan triplet t terhadap total triplet.

Berikut contoh pengaplikasian rumus TNC demgan sekuens contoh : A T G C T A G C T A

Panjang sekuens = 10 nukleotida → total triplet T = 10 – 2 = 8.

Langkah 1 : Potong sekuens menjadi triplet berurutan

Tabel 10. Pemotongan sekuens DNA (Zhang et al., 2024).

Trinukleotida	**Jumlah**
ATG	1
TGC	1
GCT	2
CTA	2
TAG	1
AGC	1
Total	**8**

Langkah 2 : Hitung frekuensi masing-masing *triplet* yang muncul

Tabel 11. Frekuensi Kemunculan (Zhang et al., 2024).

Trinukleotida	**Hasil**C_t
ATG	1
TGC	1
GCT	2
CTA	2
TAG	1
AGC	1

Langkah 3 : Hitung nilai TNC per triplet pada persamaan 17 :

$$TNC(t) = \frac{C_t}{8} \times 100\% \quad \text{..... (17)}$$

Tabel 12. Hasil Perhitungan TNC (Zhang et al., 2024).

Trinukleotida	**Hasil**$(t)\%$
ATG	(1/8)×100 = 12,5
TGC	(1/8)×100 = 12,5
GCT	(2/8)×100 = 25,0

CTA	(2/8)×100 = 25,0
TAG	(1/8)×100 = 12,5
AGC	(1/8)×100 = 12,5

Tabel 12 memaparkan bahwa triplet lain yang tidak muncul diberi nilai 0. Vektor TNC lengkap terdiri dari 64 elemen (diisi dengan 0 untuk triplet yang tidak ada). Vektor TNC ini akan menjadi masukan berdimensi 64. Pada *AdaBoost*, fitur berdimensi sedang ini cocok untuk pohon lemah sebagai *weak learner*, sedangkan *XGBoost* dapat memanfaatkan variasi TNC yang lebih kompleks (*mis. gapped k-mer*) dengan regularisasi sehingga hasil klasifikasi gen esensial lebih akurat (Chen & Guestrin, 2016).

C. *Fourier Transform* (FT)

Fourier Transform (FT) merupakan alat matematis yang digunakan untuk mengubah representasi sinyal dari domain waktu (*time domain*) menjadi domain frekuensi (*frequency domain*) (Hanson, 2003). Melalui transformasi ini, setiap komponen frekuensi serta amplitudo dalam sinyal dapat diekstraksi, sehingga mempermudah analisis karakteristik spektral pada berbagai aplikasi. Terdapat dua bentuk utama *Fourier Transform*, yakni *Discrete Fourier Transform* (DFT) dan *Fast Fourier Transform* (FFT).

Perhitungan DFT memiliki kompleksitas sebesar $\mathcal{O}(N^2)$Karena melibatkan perulangan ganda untuk menghitung semua frekuensi. Sebaliknya, FFT menurunkan kompleksitas waktu menjadi $\mathcal{O}(N \log N)$, sehingga jauh lebih efisien untuk data berdimensi besar. Oleh karena itu, dalam penelitian ini digunakan FFT untuk melakukan ekstraksi ciri, khususnya dikombinasikan dengan TNC. Berikut rumus perhitungan dapat dilihat pada persamaan 18 sampai persamaan 23 :
Persamaan umum FFT dapat dituliskan sebagai berikut:

$$X_k = X_k^{\text{even}} + W_k\, X_k^{\text{odd}} \quad \ldots\ldots\ldots\ldots\ldots (18)$$

Keterangan :

$W_k = e^{-\frac{2\pi ik}{N}}$: disebut *twiddle factor* atau faktor rotasi.

X_k^{even} : hasil DFT untuk indeks genap.

X_k^{odd} : hasil DFT untuk indeks ganjil.

Persamaan DFT

Rumus umum DFT ditulis sebagai:

$$X(k) = \sum_{n=0}^{N-1} x[n]\ e^{-\frac{2\pi i}{N}kn} \quad \text{..... (19)}$$

Keterangan :

$X(k)$: hasil transformasi Fourier ke-k.

k : indeks frekuensi.

N : panjang total sampel.

n : indeks waktu (misal 1,2,3,...).

$e^{-\frac{2\pi i}{N}kn}$: fungsi basis eksponensial kompleks.

i : bilangan imajiner ($i^2 = -1$).

Twiddle Factor

Persamaan FFT dan DFT melibatkan *Twiddle Factor*, bentuk eksplisitnya yaitu :

$$W_k^N = e^{-\frac{2\pi i}{N}k} = cos\left(\frac{2\pi k}{N}\right) - i \sin\left(\frac{2\pi k}{N}\right) \quad \text{.... (20)}$$

Magnitudo FFT

Setelah nilai transformasi diperoleh, amplitudo dihitung menggunakan :

$$|X_k| = \sqrt{\text{Re}(X_k)^2 + \text{Im}(X_k)^2} \quad \text{..... (21)}$$

Keterangan :

$\text{Re}(X_k)$: bagian real dari $|X_k|$.

$\text{Im}(X_k)$: bagian imajiner dari $|X_k|$.

Contoh perhitungan FFT

Misalkan diberikan data skuens :

Data = [1, 2, 3, 4]

Langkah pertama adalah memetakan data ke dalam subset genap dan ganjil :

X_k^{even} : menggunakan indeks [0,2] → nilai [1,3]

X_k^{odd} : menggunakan indeks [1,3] → nilai [2,4].

Perhitungan untuk X_k^{even}

Untuk $N = 2$, subset [1,3]:

Jika k = 0 ;

$$X(0) = \sum_{n=0}^{1} x[n]\ e^{-\frac{2\pi i}{2}(0n)} = 1 + 3 = 4 \quad \text{... (22)}$$

Jika k = 1 ;

$$X(1) = \sum_{n=0}^{1} x[n]\ e^{-\frac{2\pi i}{2}(1n)} = 1 + 3e^{-\pi i} = 1 - 3 = -2$$

Sehingga X_k^{even} = [4, -2]

Perhitungan untuk X_k^{odd}

Untuk N = 2, subset [2,4] :

Jika k = 0 ;

$$X(0) = \sum_{n=0}^{1} x[n]\ e^{-\frac{2\pi i}{2}(0n)} = 2 + 4 = 6$$

Jika k = 1

$$X(1) = \sum_{n=0}^{1} x[n]\ e^{-\frac{2\pi i}{2}(1n)} = 2 + 4e^{-\pi i} = 2 - 4 = -2$$

Sehingga X_k^{odd} : [6, -2]

Penggabungan nilai *Even* dan *Odd* :

$$X_k = X_k^{\text{even}} + W_k\, X_k^{\text{odd}} \quad \text{..... (23)}$$

Untuk k = 0

$$X_0 = 4 + 1 \cdot (6) = 10$$

Untuk k = 1

$$X_1 = -2 + \left(e^{-\pi i/2}\right)(-2) = -2 + 2i$$

Untuk k = 2

$$X_2 = 4 - 6 = -2$$

Untuk k = 3

$$X_3 = -2 - 2i$$

Maka hasil akhir : [10, - 2 + 2i, - 2, - 2 – 2i]

Salah satu pendekatan *feature extraction* berbasis domain frekuensi adalah *Fast Fourier Transform* (FFT). Tabel 13 menunjukkan contoh perhitungan magnitudo FFT pada sekuens pendek, yang menjadi dasar ekstraksi fitur spektral dalam penelitian ini.

Tabel 13. Hasil Magnitudo FFT (Hanson, 2003).

k	$\boldsymbol{X_k}$ **(kompleks)**	$\mathbf{Re}(\boldsymbol{X_k})$	$\mathbf{Im}(\boldsymbol{X_k})$	**Hasil**
0	10	10	0	10
1	-2+2i	-2	2	2,83
2	-2	-2	0	2
3	-2-2i	-2	-2	2,83

Tabel 13 mengilustrasikan contoh perhitungan *Fast Fourier Transform* (FFT) pada sebuah sekuens DNA pendek sederhana, dengan fokus pada magnitudo (besaran/amplitudo) komponen frekuensi yang dihasilkan. Hasil magnitudo dari transformasi *Fourier* pada contoh sekuens sederhana. Magnitudo dihitung menggunakan rumus $\sqrt{\text{Real}^2 + \text{Imaginary}^2}$. Nilai tertinggi pada k=0 menunjukkan komponen rata-rata (DC *component*) dari sinyal. Sedangkan nilai pada k=1 dan k=3 yang identik merupakan akibat dari sifat simetri Hermitian pada transformasi *Fourier* untuk data real. Dalam konteks penelitian ini, magnitudo FFT digunakan untuk mengekstrak pola periodik yang tersembunyi dalam urutan nukleotida DNA, sehingga menghasilkan fitur yang dapat membedakan karakteristik gen esensial dan non-esensial. Hasil akhir magnitudo FFT: [10, 2.83, 2, 2.83]. Nilai ini menjadi representasi fitur yang dapat dipakai pada klasifikasi dengan *AdaBoost*.

D. *Protein-protein Interaction Degree (PPI_Degree)*

Protein-Protein Interaction Degree (*PPI_degree*) merupakan salah satu fitur topologi jaringan yang paling sederhana namun sangat informatif dalam analisis bioinformatika, khususnya untuk prediksi gen esensial. *PPI_degree* didefinisikan sebagai jumlah interaksi langsung (*degree centrality*) yang dimiliki oleh suatu protein dalam jaringan *interaksi protein-protein* (PPI). Dalam representasi graf, jika protein dianggap sebagai node dan interaksi fisik sebagai *edge*, maka *PPI_degree* adalah derajat node tersebut (jumlah *edge* yang terhubung langsung). Fitur ini mencerminkan seberapa "sentral" suatu protein dalam jaringan biologis, di mana gen esensial sering kali menghasilkan protein dengan *PPI_degree* tinggi karena berperan sebagai hub dalam proses vital seperti replikasi DNA, transkripsi, translasi, dan regulasi metabolisme.

Menurut hipotesis "*centrality-lethality rule*" (Jeong et al., 2001), protein dengan konektivitas tinggi cenderung esensial karena penghapusannya dapat mengganggu kestabilan jaringan secara keseluruhan. Pada *Drosophila melanogaster*, fitur ini telah terbukti berkorelasi positif dengan esensialitas gen, karena gen esensial sering terlibat dalam interaksi multiprotein yang konservatif lintas spesies. Dalam konteks ekstraksi fitur untuk *machine learning*, *PPI_degree* diekstrak dari basis data PPI terintegrasi seperti *FlyBase*, *STRING*, atau *BioGRID*. Nilai ini bersifat numerik tunggal (*integer* atau *float* jika dinormalisasi), sehingga mudah diintegrasikan dengan fitur *sequence-based* lainnya seperti AAC (20 dimensi), TNC (64 dimensi), dan FT (100 dimensi), menghasilkan total 185 fitur. Penambahan *PPI_degree* memberikan dimensi fungsional yang melengkapi fitur sekuens murni, meningkatkan interpretabilitas model dan akurasi prediksi pada kondisi data *imbalance* tinggi.

Secara matematis, *PPI_degree* dihitung berdasarkan teori graf sederhana. Misalkan G = (V, E) adalah graf PPI tidak berarah, di mana V adalah himpunan node (protein) dan E adalah himpunan *edge*

(interaksi protein). *PPI_degree* untuk suatu node v ∈ V didefinisikan dalam persamaan 24 berikut.

$$\deg(v) = |\{u \in V \mid (u,v) \in E\}| \text{ ..... (24)}$$

Keterangan :

$\deg(v)$: Derajat mentah (jumlah node yang terhubung langsung)

v : Protein target (node yang sedang dihitung *PPI_degree*-nya)

V : Himpunan semua protein (node)

E : Himpunan semua interaksi (*edge*)

u : Protein yang berinteraksi langsung dengan protein v

$(u,v) \in E$: Ada interaksi antara protein u dan v

Persamaan 25 menyatakan bahwa derajat node v adalah jumlah node lain yang terhubung langsung dengannya. Jika graf berbobot (*weighted graph*), di mana bobot *edge* mencerminkan kekuatan interaksi (misalnya *confidence score* dari *STRING database*), maka *PPI_degree* dapat dimodifikasi menjadi *weighted degree* seperti pada persamaan 25 :

$$\deg_w(v) = \sum_{u \in N(v)} w(u,v) \text{ ..... (25)}$$

Keterangan :

$\deg_w(v)$: Derajat berbobot

$N(v)$: Himpunan node yang terhubung langsung dengan protein v

Persamaan 26 menyatakan bahwa derajat berbobot dari protein v Adalah jumlah bobot semua interaksi langsung yang dimiliki oleh protein tersebut dalam jaringan berbobot.

Untuk mengatasi distribusi *degree* yang mengikuti *power-law* (di mana sebagian besar node memiliki *degree* rendah, sementara sedikit node memiliki *degree* sangat tinggi), beberapa penelitian menerapkan transformasi normalisasi atau logaritmik. Contoh normalisasi *min-max* untuk membuat nilai sebanding antar-fitur pada persamaan 26 :

$$deg_{norm}(v) = \frac{deg(v) - \min(deg)}{\max(deg) - \min(deg)} \text{ ..... (27)}$$

Keterangan :

$\min(deg)$: Derajat minimum di seluruh graf

$\max(deg)$: Derajat maksimum di seluruh graf

Persamaan 27 menyatakan bahwa *min_deg* dan *max_deg* adalah derajat minimum dan maksimum di seluruh graf. Alternatif lain adalah transformasi logaritmik untuk mengurangi *skewness* seperti pada Persamaan 28.

$$\deg_{\log}(v) = \log(\deg(v) + 1) \text{ ..... } (26)$$

Keterangan :

$(\deg(v) + 1)$: Transformasi logaritmik yang diterapkan pada derajat untuk mengurangi *skewness* distribusi *power-law*.

Persamaan 28 menyatakan: ditambah 1 untuk menghindari log(0) pada node *isolated*. Dalam penelitian ini, digunakan nilai *PPI_degree* mentah (*raw degree*) sesuai praktik umum pada model *boosting* seperti *XGBoost*, yang robust terhadap skala fitur berbeda, meskipun normalisasi dapat diterapkan jika diperlukan untuk meningkatkan stabilitas model (Beder et al., 2021).

Penelitian terdahulu menunjukkan bahwa *PPI_degree* merupakan prediktor kuat untuk esensialitas gen. Misalnya, dalam studi berbasis *machine learning* pada berbagai organisme eukariota (termasuk Drosophila), fitur topologi seperti *degree centrality* sering kali memiliki *importance* tinggi dalam model prediksi, karena gen esensial cenderung berada di posisi sentral jaringan PPI.

2.2.13 Feature Selection

Feature Selection merupakan proses untuk memilih fitur-fitur yang paling relevan dari sekumpulan data sehingga analisis dapat dilakukan lebih akurat, sederhana, dan efisien. Dalam praktiknya, hanya fitur yang benar-benar berpengaruh terhadap hasil penelitian yang akan dipertahankan, sementara fitur yang kontribusinya kecil dapat diabaikan.

A. Random Forest Gini Importance Index (GII)

Salah satu metode seleksi fitur yang banyak digunakan adalah Gini Importance Index (GII) pada algoritma Random Forest. GII

mengukur seberapa besar kontribusi suatu fitur dalam mengurangi nilai *Gini Impurity* ketika dilakukan pemisahan data pada node pohon keputusan (Menze et al., 2009). Dengan kata lain, fitur yang lebih sering digunakan dalam pemisahan dan menghasilkan penurunan *impurity* yang signifikan dianggap lebih penting. *Random Forest Gini Importance* bekerja dengan membangun sejumlah besar pohon keputusan (biasanya ratusan pohon) secara acak menggunakan teknik *bagging* dan *random feature selection*. Pada setiap proses pembuatan split di dalam pohon, algoritma menghitung penurunan nilai Gini Index yang dihasilkan oleh setiap fitur. Semakin besar penurunan Gini Index yang disebabkan oleh suatu fitur, semakin tinggi kemampuan fitur tersebut dalam memisahkan kelas target. Nilai *Gini Importance* akhir diperoleh dengan merata-ratakan total penurunan Gini Index dari fitur tersebut di seluruh pohon yang dibangun. Fitur yang memiliki nilai *Gini Importance* tinggi dianggap paling berpengaruh terhadap variabel target (dalam penelitian ini adalah label gen esensial dan non-esensial), sedangkan fitur dengan nilai rendah dianggap kurang informatif atau *noise*. Setelah nilai *importance* dihitung untuk seluruh 185 fitur, semua fitur diurutkan dari yang paling penting ke yang paling rendah. Selanjutnya, dipilih sejumlah fitur terbaik (top-k) untuk digunakan pada tahap pemodelan.

Pemilihan 45 fitur dilakukan bukan berdasarkan tebakan, melainkan hasil analisis dan eksperimen yang sistematis. Sebanyak 45 fitur teratas berhasil mencakup sekitar 85–90% kumulatif *Gini Importance*, sehingga penambahan fitur di luar itu hanya memberikan kontribusi yang marginal. Selain itu, eksperimen pendahuluan dengan variasi jumlah fitur (30, 45, 60, 80, 100, dan 185 fitur) menunjukkan bahwa 45 fitur memberikan performa optimal pada metrik utama, khususnya ROC-AUC dan PR-AUC selama validasi silang. Jumlah ini juga mencerminkan keseimbangan yang baik antara reduksi dimensi dan retensi informasi, karena terlalu sedikit fitur berisiko menyebabkan underfitting, sementara terlalu banyak fitur dapat

meningkatkan risiko overfitting serta waktu komputasi. Pemilihan ini sejalan dengan praktik pada penelitian serupa (Azhari, 2025; Beder et al., 2021) yang umumnya menggunakan 40–100 fitur setelah seleksi. Berikut rumus perhitungan dapat dilihat pada persamaan 27 sampai persamaan 29.

Rumus menghitung Gini *Impurity* pada sebuah node dituliskan sebagai berikut :

$$i(\tau) = 1 - \sum_{k=0}^{1} p_k^2 \quad \text{..... (27)}$$

Keterangan :

p_k : proporsi data pada node yang masuk ke dalam kelas ke-k.

Untuk dua kelas (0 dan 1), berlaku $p_0 + p_1 = 1$.

Apabila suatu fitur digunakan untuk melakukan *split*, maka nilai *Weighted Gini* dapat dihitung dengan rumus:

$$Gini_{\text{split}} = \frac{N_{\text{left}}}{N} Gini_{\text{left}} + \frac{N_{\text{right}}}{N} Gini_{\text{right}} \quad \text{..... (28)}$$

Selanjutnya, *Gini Importance* dari suatu fitur XjX_jXj dapat diperoleh dengan menghitung selisih antara impurity node induk (*parent*) dan impurity setelah pemisahan :

$$GI(X_j) = Gini_{\text{parent}} - Gini_{\text{split}} \quad \text{..... (29)}$$

Contoh perhitungan :

Misalkan terdapat 10 fitur dengan nilai awal

X = {0.1, 0.5, 0.8, 0.34, 0.75, 0.23, 0.15, 0.18, 0.63, 0.39} dan label biner target

Y= [0, 1, 1, 0, 1, 0, 0, 0, 1, 1].

Hitung Gini *Parent* :

$$Gini_{\text{parent}} = 1 - (p_0^2 + p_1^2) = 1 - \left(\left(\frac{10}{5}\right)^2 + \left(\frac{10}{5}\right)^2\right) = 0.5$$

Split menggunakan fitur X3 dengan *threshold* 0.6 :

Left ($\leq$ 0.6) =7 sampel → 4 kelas 0, 3 kelas 1

Right ($>$ 0.6) = 3 sampel → 1 kelas 0,2 kelas 1

Maka :

$$Gini_{\text{left}} = 1 - \left(\left(\frac{7}{4}\right)^2 + \left(\frac{7}{3}\right)^2\right) = 0.491$$

$$Gini_{\text{right}} = 1 - \left(\left(\frac{3}{1}\right)^2 + \left(\frac{3}{2}\right)^2\right) = 0.445$$

$$Gini_{\text{split}} = \frac{10}{7} \times 0.491 + \frac{10}{3} \times 0.445 = 0.4772$$

Hitung Gini *Importance* untuk X3 :

$$GI(X_3) = Gini_{\text{parent}} - Gini_{\text{split}} = 0.5 - 0.4772 = 0.0228$$

Tabel 14. Hasil Perhitungan *Gini Importance* (Menze et al., 2009).

Fitur	Nilai Fitur	Gini *Importance*
F1	0,1	0,02
F2	0,5	0,15
F3	0,8	0,25
F4	0.34	0,10
F5	0,75	0,22
F6	0,23	0,08
F7	0,15	0,05
F8	0,18	0,06
F9	0.63	0,18
F10	0,39	0,12

Dari hasil pada Tabel 14, dapat dilihat bahwa fitur dengan nilai *Gini Importance* tertinggi adalah F3, F5, F9, dan F2, sehingga keempat fitur inilah yang dipilih untuk tahap analisis berikutnya.

B. Synthetic Minority Over-sampling Technique (SMOTE) Tomek Links (SMOTETomek)

Selain seleksi fitur menggunakan *Random Forest Gini Importance Index* (GII) untuk mengurangi dimensi dari 185 fitur menjadi 45 fitur terbaik, penanganan ketidakseimbangan kelas (*imbalance ratio* 1:8,41 pada label CEG) menjadi krusial sebelum pelatihan model. Teknik resampling hibrida SMOTETomek digunakan untuk menyeimbangkan dataset tanpa menyebabkan overfitting atau noise pada batas kelas.

SMOTETomek merupakan metode kombinasi antara *Synthetic Minority Over-sampling Technique* (SMOTE) dan *Tomek Links*

(Batista et al., 2004; Zhang et al., 2024). Metode ini pertama kali meng-*oversample* kelas minoritas (gen esensial) secara sintetis, kemudian membersihkan sampel *borderline* yang ambigu antarkelas.

Pada dataset bioinformatika yang sangat *imbang* seperti gen esensial *Drosophila melanogaster*, SMOTETomek tidak hanya meningkatkan jumlah sampel kelas minoritas, tetapi juga menghilangkan *noise* dan *overlapping* di *decision boundary*. Hal ini meningkatkan performa metrik sensitif kelas minoritas (*F1-score*, PR-AUC, MCC) pada model *boosting* (*AdaBoost* dan *XGBoost*) serta mencegah bias mayoritas tanpa kehilangan informasi penting.

Langkah-langkah SMOTETomek (Zhang et al., 2024):

1. Terapkan SMOTE untuk meng-*oversample* kelas minoritas.
2. Hitung *nearest neighbor* dari semua sampel kelas mayoritas.
3. Identifikasi Tomek *links*: jika *nearest neighbor* dari sampel mayoritas adalah sampel minoritas (atau sebaliknya), pasangan tersebut merupakan Tomek *link*.
4. Hapus semua link Tomek untuk membersihkan data, sehingga diperoleh dataset yang lebih seimbang dan bersih.

Rumus matematis:

Pembentukan sampel sintetis pada tahap SMOTE pada persamaan 30 :

$$x_{\text{new}} = x_i + \lambda(x_{nn} - x_i), \quad \lambda \in [0,1] \text{ ..... (30)}$$

Keterangan :

x_{new} : sampel sintetis baru (kelas minoritas).

x_i : sampel asli minoritas.

x_{nn} : salah satu *nearest neighbor* (sampel terdekat) dari kelas minoritas yang sama

λ : bilangan acak [0,1] → menentukan posisi sampel baru di antara x_i dan x_{nn}.

Tujuan : menambah jumlah gen esensial secara sintetis agar seimbang.

Di mana x_i adalah sampel minoritas, x_{nn} adalah salah satu *nearest neighbor* dari kelas yang sama, dan λ Adalah bilangan acak.

Definisi Tomek *link* (pasangan yang dihapus) pada persamaan 31:

$$d(x_i, x_j) = \min\{ d(x_i, x_k) \mid \forall k \neq i, j\} \text{ ..... (31)}$$

Keterangan :

$d(x_i, x_j)$: jarak antara sampel $x_i$dan x_j (biasanya *Euclidean distance*).

Rumus Euclidean:

$$d(x_i, x_j) = \sqrt{\sum_{m=1}^{p} (x_{i,m} - x_{j,m})^2}$$

Di mana p= jumlah fitur (dalam tesis Anda: 45 fitur setelah seleksi).

x_i : Sampel dari salah satu kelas (biasanya dari kelas mayoritas setelah SMOTE).

x_j : Sampel dari kelas berbeda (biasanya dari kelas minoritas).

x_k : semua sampel lain dalam dataset (selain $x_i$dan x_j).

Kondisi min {} : x_j adalah adalah *nearest neighbor* dari x_i, dan sebaliknya x_i adalah *nearest neighbor* dari x_j. Artinya: keduanya saling menjadi *nearest neighbor* satu sama lain → ini adalah pasangan yang "berada di batas kelas" dan sering menjadi sumber *noise* atau *overlapping*.

Tujuan : hilangkan *noise* di *decision boundary* setelah *oversampling*.

Dengan penerapan SMOTETomek (hanya pada data *training* untuk menghindari data *leakage*), rasio kelas menjadi mendekati 1:1, sehingga model dapat belajar pola gen esensial dengan lebih baik sebelum masuk ke tahap *feature selection Gini Importance* dan pelatihan.

2.2.14 Evaluasi

Evaluasi merupakan tahap penting untuk menilai seberapa baik kinerja suatu model klasifikasi. Pada penelitian ini, evaluasi dilakukan

untuk mengetahui akurasi dua algoritma, yaitu *AdaBoost* dan XGBoost, guna mengetahui sejauh mana keduanya mampu memberikan hasil prediksi yang akurat dan dapat digunakan dalam skenario nyata.

A. Confusion Matrix

Confusion matrix adalah metode evaluasi yang umum digunakan untuk menilai performa model klasifikasi. Matriks ini memperlihatkan perbandingan antara hasil prediksi model dengan label aktual. Terdapat empat komponen utama pada *confusion matrix*, yaitu *True Positive* (TP), *True Negative* (TN), *False Positive* (FP), dan *False Negative* (FN) (Rahma et al., 2021). Struktur *confusion matrix* ditunjukkan pada Tabel 15.

Tabel 15. *Confusion Matrix* (Rahma et al., 2021).

Predicted Value	***Actual Positive***	***Actual Negative***
Positive	TP (*True Positive*)	FP (*False Negative*)
Negative	FN (*False Negative*)	TN (*True Negative)*

Keterangan :

True Positive (TP) : data positif diprediksi benar sebagai positif.
True Negative (TN) : data negatif diprediksi benar sebagai negatif.
False Negative (FN) : data positif diprediksi salah sebagai negatif.
False Positive (FP) : data negatif diprediksi salah sebagai positif.

B. ROC – AUC

ROC (*Receiver Operating Characteristic)* adalah metode evaluasi kinerja model klasifikasi yang menggambarkan hubungan antara *True Positive Rate* (TPR) dan *False Positive Rate* (FPR). Kurva ROC menampilkan *trade-off* antara kemampuan model dalam mendeteksi data positif dengan benar (sensitivitas) dan kemungkinan model salah mengklasifikasikan data negatif sebagai positif (spesifisitas rendah).

AUC (*Area Under the Curve*) merupakan ukuran probabilistik yang menunjukkan seberapa baik model dapat membedakan kelas positif dan negatif. Nilai AUC berkisar antara 0 hingga 1, di mana

nilai mendekati 1 menunjukkan kinerja model yang optimal. AUC digunakan untuk menilai seberapa robust algoritma *AdaBoost* dan XGBoost dalam menghadapi variasi distribusi data (Majnik & Bosnić, 2013). Berikut rumus perhitungan dapat dilihat pada Persamaan 32 dan Persamaan 33.

Rumus TPR (*recall/sensitivity*) :

$$TPR = \frac{TP}{TP + FN} \quad \text{..... (32)}$$

Rumus FPR :

$$FPR = \frac{FP}{FP + TN} \quad \text{..... (33)}$$

Keterangan :

TP : jumlah data positif yang diprediksi benar positif.

TN : jumlah data negatif yang diprediksi benar negatif.

FP : jumlah data negatif yang salah diprediksi sebagai positif.

FN : jumlah data positif yang salah diprediksi sebagai negatif.

Contoh :

Jika terdapat 100 data, dengan distribusi TP = 40, FP = 10, TN = 30, dan FN = 20, maka:

$$TPR = \frac{40}{40 + 20} = \frac{40}{60} = 0.67\ (67\%)$$

$$FPR = \frac{10}{10 + 30} = \frac{10}{40} = 0.25\ (25\%)$$

Interpretasi: model berhasil mengenali 67% data positif secara benar, tetapi masih salah mengklasifikasikan 25% data negatif sebagai positif. Dalam perbandingan, model yang baik (*AdaBoost* maupun XGBoost) diharapkan memiliki nilai TPR tinggi dan FPR rendah.

Kurva ROC yang baik adalah kurva yang mendekati titik kiri atas (TPR tinggi, FPR rendah). Hal ini menandakan bahwa model mampu membedakan dengan baik antara kelas positif dan kelas negatif.

C. PR – AUC

PR-AUC (*Area Under the Precision-Recall Curve*) merupakan metrik evaluasi yang sangat berguna terutama pada dataset dengan distribusi kelas tidak seimbang. Metrik ini mengukur hubungan antara *precision* (proporsi prediksi positif yang benar-benar positif) dan *recall* (kemampuan model menemukan semua data positif).

Kurva *precision-recall* memvisualisasikan bagaimana perubahan *threshold* memengaruhi keseimbangan antara *precision* dan *recall*. PR-AUC lebih sensitif terhadap ketidakseimbangan data dibandingkan dengan ROC-AUC, sehingga sering dipakai dalam penelitian bioinformatika atau klasifikasi dengan data yang jarang (Sofaer et al., 2019). Berikut rumus perhitungan dapat dilihat pada Persamaan 34 dan Persamaan 35.

Rumus *Precision* :

$$Precision = \frac{TP}{TP + FP} \quad \text{..... (34)}$$

Rumus *Recall* :

$$Recall = \frac{TP}{TP + FN} \quad \text{..... (35)}$$

Keterangan :

Precision : proporsi data yang diprediksi positif yang benar-benar positif.

Recall : proporsi data positif yang berhasil terdeteksi dengan benar.

Contoh :

Jika hasil prediksi menghasilkan TP = 50, FP = 30,

dan FN = 20, maka:

$$Precision = \frac{50}{50 + 30} = \frac{50}{80} = 0.625\ (62.5\%)$$

$$Recall = \frac{50}{50 + 20} = \frac{50}{70} = 0.714\ (71.4\%)$$

Interpretasi: model mampu memberikan prediksi positif yang benar sebanyak 62,5% dan mendeteksi 71,4% dari seluruh data positif.

III. METODOLOGI PENELITIAN

3.1. Tempat dan Waktu Penelitian

3.1.1. Tempat Penelitian

Penelitian dilaksanakan di Jurusan Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam, Universitas Lampung.

3.1.2. Waktu Penelitian

Penelitian ini dilaksanakan pada Oktober 2025. Proses penelitian dibagi menjadi tiga tahap utama, yaitu:

1. Tahap Awal Penelitian

   Menentukan tema, memahami topik penelitian, mengumpulkan literatur, dan menyusun draf proposal (Bab 1–3).

2. Tahap Penelitian Lanjutan

   Tahap ini merupakan inti dari penelitian, yang dimulai dari pra-pemrosesan data (penggabungan data DNA dan protein, penanganan *missing value*), ekstraksi fitur menggunakan *Amino Acid Composition* (AAC) dan *PPI_Degree* untuk protein serta *Tri-Nucleotide Composition* (TNC) dan *Fourier Transform* (FT) untuk DNA.

   Selanjutnya dilakukan seleksi fitur dengan *Random Forest Gini Importance*, pembagian data *training* dan *testing*, penyeimbangan data dengan SMOTE, serta pelatihan model menggunakan dua algoritma, yaitu *AdaBoost* dan *XGBoost*.Pada tahap akhir, data dilatih menggunakan dua metode klasifikasi, yaitu *AdaBoost* dan *XGBoost.*

3. Tahap Evaluasi

   Tahap akhir meliputi pengujian model dengan data *testing*, analisis hasil, penulisan draft akhir tesis, serta penyampaian hasil penelitian melalui seminar hasil.

3.2. Data dan Perangkat Pendukung

3.2.1. Data

Data yang digunakan dalam penelitian ini adalah data sekuens DNA dan protein *Drosophila melanogaster* yang diambil dari penelitian Beder et al. (2021). Dataset terbagi menjadi dua kategori berdasarkan level esensialitas gen:

CEG (*Cellular Essential Gene*) → gen esensial yang terlibat dalam proses biogenesis dan siklus sel (poliferasi).

OEG (*Organismal Essential Gene*) → gen esensial yang terlibat dalam morfogenesis, regulasi perkembangan, dan persinyalan saraf.

Jumlah data:

CEG : Total 17.774 data (2.083 esensial dan 15.661 non-esensial).

OEG : Total 900 data (553 esensial dan 347 non-esensial).

Dataset diperoleh dalam format CSV (berisi metadata dan label) dan FASTA (berisi sekuens DNA/protein). Data dari file FASTA akan dikonversi dan digabungkan dengan file CSV berdasarkan ID yang sama untuk membentuk dataset terpadu yang siap diproses.

3.2.2. Perangkat Pendukung

Perangkat Keras :

Prosesor : 11th Gen Intel (R) Core(TM) i7-11800H @ 2.30GHz (16 CPUs), ~2.3 GHz

RAM : 16 GB

Storage : 512 GB NVMe SSD (atau 1 TB NVMe SSD)

VGA : NVIDIA GeForce RTX 3060 Laptop GPU

Perangkat Lunak :

Penelitian akan dilakukan menggunakan bahasa pemrograman Python dengan lingkungan pengembangan Jupyter Notebook. *Library* yang akan dimanfaatkan antara lain:

Python 3.10: Bahasa pemrograman inti.

Pandas 2.2.2: Untuk manipulasi dan analisis data terstruktur.

NumPy 1.26.4: Untuk komputasi numerik dan operasi array.

Scikit-learn (Sklearn) 1.6.1: Untuk pra-pemrosesan data, pembagian dataset, implementasi algoritma *AdaBoost*, dan evaluasi model.

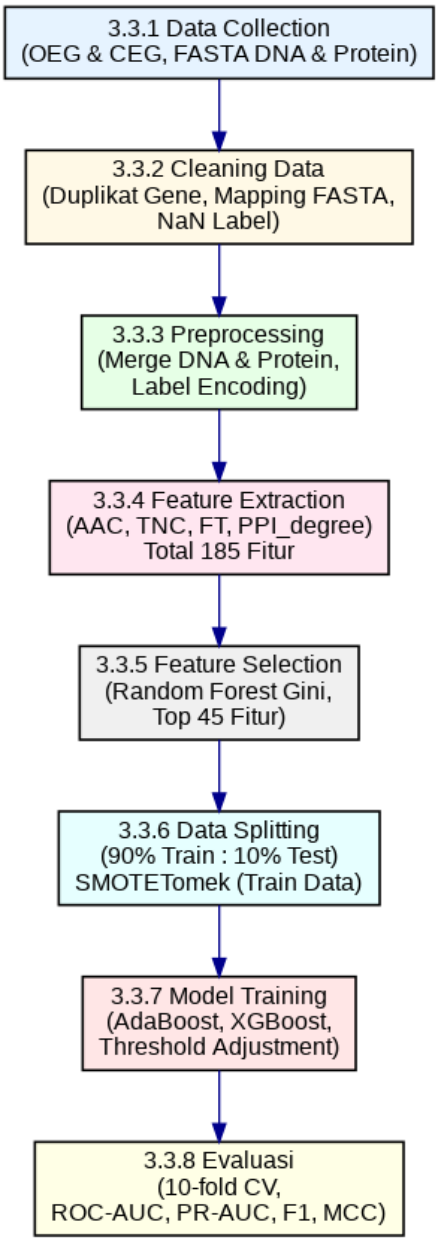
XGBoost 2.0+: Untuk implementasi algoritma *XGBoost*.

Imbalanced-learn (Imblearn) 0.13.0: Untuk menangani ketidakseimbangan kelas menggunakan teknik SMOTE.

Matplotlib 3.10.0: Untuk visualisasi data dan hasil.

3.3. Alur Kerja Penelitian

Alur penelitian terdiri dari beberapa tahapan utama :



Gambar 5. Alur Penelitian.

3.3.1. Data Collection

Dataset DNA dan protein *Drosophila melanogaster* diunduh dari sumber (Beder et al., 2021).

```
Membuat data dummy 12429 baris...
DataFrame created with shape: (12429, 5)

Tampilan Tabel:
```

	Gene	Essential	CEG	Essential_2	OEG
0	FBgn0000000	Non-essential	NaN	NaN	NaN
1	FBgn0000001	Essential	NaN	NaN	NaN
2	FBgn0000002	Non-essential	NaN	NaN	NaN
3	FBgn0000003	Non-essential	NaN	NaN	NaN
4	FBgn0000004	Essential	NaN	NaN	NaN
...	...	...	...	...	...
12424	FBgn0012424	Non-essential	NaN	Non-essential	
12425	FBgn0012425	Non-essential	NaN	NaN	
12426	FBgn0012426	Non-essential	NaN	NaN	
12427	FBgn0012427	Non-essential	NaN	NaN	
12428	FBgn0012428	Non-essential	NaN	NaN	

Gambar 6. Dataset File csv.

Pada Gambar 6 ditampilkan data dalam format FASTA dikonversi dan digabungkan dengan data CSV.

Gambar 7. Dataset File FASTA (DNA).

Pada Gambar 7 merupakan dataset FASTA untuk DNA pada *Drosophila Melanogaster*.

Gambar 8. Dataset File FASTA (Protein).

Pada Gambar 8 merupakan dataset FASTA untuk Preotein pada *Drosophila Melanogaster*.

Dataset yang digunakan dalam penelitian ini berasal dari CLEARER yang dikembangkan oleh Beder et al. (2021). File CSV awal berisi 12.429 baris data gen beserta *anotasi essentiality* dari Classical Essential Genes (CEG) dan *Orthologous Essential Genes* (OEG), seperti yang ditunjukkan pada Gambar 6. Sekuens DNA (*coding sequence*/CDS) dan sekuens protein disimpan terpisah dalam format FASTA dengan total panjang masing-masing 77.749.387 nukleotida (Gambar 7) dan 13.057.090 asam amino (Gambar 8).

3.3.2. Cleaning Data

Proses pembersihan data (*cleaning data*) adalah tahapan untuk membersihkan dataset dari entri yang tidak memiliki nilai, yang dikenal sebagai *missing value*. Ciri dari kondisi ini adalah adanya sel kosong atau nilai yang tertanda sebagai NaN. Tujuan dilakukannya pembersihan data adalah untuk mencegah terjadinya bias selama proses pelatihan dan pengujian model, serta menghindari kesalahan

dalam analisis yang dapat menurunkan tingkat akurasi klasifikasi model.

Dataset awal dalam format CSV memiliki total 12.429 gen. Dataset ini mencakup kolom label 'Essential CEG' dan 'Essential OEG', dengan rincian sebagai berikut (sebelum cleaning):

- *Essential* CEG: 1.227 gen (*Essential*), 10.320 gen (*Non-essential*), dan 882 gen (NaN/*missing label*).
- *Essential* OEG: 246 gen (*Essential*), 271 gen (*Non-essential*), dan 11.912 gen (NaN/*missing label*).

Tahapan *cleaning* yang dilakukan adalah sebagai berikut:

1. Menghapus duplikat berdasarkan kolom '*Gene'* (menggunakan *keep='first'*) untuk memastikan setiap gen hanya muncul satu kali.;
2. *Mapping* sekuens DNA dan protein dari file FASTA (Dm_DNA.fa dan Dm_prot.fa) berdasarkan kolom *'Gene'. Mapping* dilakukan dengan *fallback* (menghilangkan suffix seperti -PA/-PB jika tidak *match* langsung);
3. Menghapus gen yang gagal *mapping* sekuens DNA atau protein (tidak memiliki dna_seq atau prot_seq setelah proses *mapping*);
4. Tidak menghapus baris yang memiliki *missing value* pada label CEG atau OEG, sehingga semua label tetap ditampilkan (termasuk NaN) untuk menjaga kelengkapan informasi.

Setelah proses *cleaning*, jumlah gen tetap 12.429 (tidak ada penurunan signifikan karena *drop* hanya pada *missing sequence*, dan label NaN dipertahankan). Dataset akhir memiliki kolom tambahan 'dna_seq' dan 'prot_seq' serta tetap mempertahankan semua nilai label CEG dan OEG (termasuk NaN). Hal ini membuat dataset menjadi lebih bersih, lengkap, dan representatif untuk tahap *preprocessing* serta pemodelan selanjutnya.

3.3.3. Preprocessing

Preprocessing data adalah tahap persiapan data agar siap untuk diolah pada proses berikutnya. Dalam penelitian ini, setelah data bersih, dilakukan tahap *preprocessing* untuk mengubah data menjadi bentuk yang siap dimasukkan ke model *machine learning*.

No	Gene	Essential CEG	Essential OEG	DNA (potongan sequence)	Protein (potongan sequence)
0	FBgn0013380	Non-essential	NaN	AACTTTGTATCCTGCTGGTAGATCGATAGCTAGCTAGCTAGCTAGC...	MGKKVHMKARQPQTVNSAVKKIQIDVDAVGGNQG...
1	FBgn0052344	Essential	NaN	GTTATCGATGGTGTAATCAACATCGATTTGTCACGGT...	MRKKQDEIPFPSLDNDGTSRDGALIKSKSKNKSK...
2	FBgn0016983	Non-essential	NaN	CAGTTACCTGCTTAACTGCGTTCACCACAAAATTTGA...	MTRELMQKPEYSRRKFGPQHLVQAFSISESYLQD...
3	FBgn0027602	Non-essential	NaN	AGTGGAACGAAGGAAGCGCACGCGTTTTTATTA...	VENISLIVTVKSARKQPPSALLKRRAKQSFDF...
4	FBgn0033160	Essential	NaN	CAGTTCCAATCAATTGTCAAATGCAATTTAAGTA...	MSKRIEVETYGKAKPEESSAAAAGTVQILGGFVPNK...
5	FBgn0019995	Non-essential	NaN	ATGGCACCACCGACGAATAAAAGCGCAT...	MGATILLLALFVIRLIGRLKLAWLQTHDQ...
6	FBgn0036373	Non-essential	NaN	CCTTCCGCCTGTCACAACAGGCAACGTTTTCC...	MSDDDQIKQRYTHLLESERKRKAAASAK...
7	FBgn0036686	Essential	NaN	CCAATTTGGGGTCTTTCAATCAACACACATG...	HADAGKRKHQARQAVKDKKAKKSKSKKK...
8	FBgn0039938	Essential	NaN	CTCATGCGTACCATTACAGCACGTCGCCG...	MPSYVAKRLKGLKGVKRGKGVKRGKGVKRGKGVKRGKGVKRGKGVKRGKG...
9	FBgn0026802	Essential	NaN	CCTTCCAGACGCCACTTTGTGGAAGGCCAC...	MPSYVAKRLKGLKGVKRGKGVKRGKGVKRGKGVKRGKGVKRGKG...
...	...	...	...	...	...
11537	FBgn0035932	Non-essential	NaN	ATGCTGCGCTGTTGATCTGCCAGTCGGATTTCC...	MLRCLLLAIGFLVRILGIRGDNCASVSNACISNT...
11538	FBgn0051219	Non-essential	NaN	TTACGGCCAACATTGTCATCATCCCGTGC...	MFISCIPFALVVFQLAHSTNSDCPYGEKYVYLECP...
11539	FBgn0037310	Non-essential	NaN	ATGAATTTCATTGGTTAATACTTTACCAC...	MKFALVLYLISFVCEGAISVNSTFEKIGKRYNTADH...
11540	FBgn0050323	Non-essential	NaN	TCTGTAAAAAAAAGCTTATCATGATCACCGCCTTA...	MQFLLLLLLSSAYNEGKGLQRNLVTDNYQNWSIR...
11541	FBgn0029531	Non-essential	NaN	TGGGCTTAGACAAGGCGCCGCCGCGTGGT...	MRLSFVTLILTGSLNAFPMKTTETTREPSGQILQQ...
11542	FBgn0038321	Non-essential	NaN	AGAAAGACCCACCCATCAAGCTAGTGATATGGA...	MRRSVLLFLISAVLSALPLPPIEDSEDESYNDYSE...
11543	FBgn005320	Non-essential	NaN	TTAGTTTCATTCACCAAAGCGAGCAAGAA...	MNFSTFILIALIVTSKSCAPFKEGNCKTAKTI...
11544	FBgn0039475	Non-essential	NaN	ACTACCATCGTGGGCCTCCGACTATTAAGACAT...	MKTFILAFALAASIPIKESERHVGNYVPQEDGS...
11545	FBgn002621	Non-essential	NaN	AAGACCTGCGAACATCGCCTTTGATGAGCAAGATC...	MHRLVIGLLLASFNRETIGKRLNLKPIQIPHEDEST...
11546	FBgn0026836	Non-essential	NaN	CCTTGCATGACAGACAGTTTCTGAAGACAGAC...	MDSFLQVKSTLLDFLSENMRKPRVWFSA...

Gambar 9. Dataset Sudah dilakukan Penggabungan.

Pada Gambar 9 merupakan contoh visual penggabungan (*merge*) sekuens DNA dan protein ke dalam satu *DataFrame* berdasarkan *FlyBase gene identifier* (FBgn). Kemudian dilakukan pemberian label yang dilakukan sebagai penentu kategori esensial atau non esensial. Visualisasi pemberian label dapat dilihat pada Gambar 10 berikut.

No	Gene	Essential CEG (asli)	Essential CEG Encoded	Essential OEG (asli)	Essential OEG Encoded
0	FBgn0013380	Non-essential	0	NaN	NaN
1	FBgn0052344	Essential	1	NaN	NaN
2	FBgn0016983	Non-essential	0	NaN	NaN
3	FBgn0027602	Non-essential	0	NaN	NaN
4	FBgn0033160	Essential	1	NaN	NaN
5	FBgn0019995	Non-essential	0	NaN	NaN
6	FBgn0036373	Non-essential	0	NaN	NaN
7	FBgn0036686	Essential	1	NaN	NaN
8	FBgn0039938	Essential	1	NaN	NaN
9	FBgn0026802	Essential	1	NaN	NaN
10	FBgn0035932	Non-essential	0	NaN	NaN
11	FBgn0051219	Non-essential	0	NaN	NaN
12	FBgn0037310	Non-essential	0	NaN	NaN
13	FBgn0050323	Non-essential	0	NaN	NaN
14	FBgn0029531	Non-essential	0	NaN	NaN
...	...	...	...	...	...
11542	FBgn0038321	Non-essential	0	NaN	NaN
11543	FBgn005320	Non-essential	0	NaN	NaN
11544	FBgn0039475	Non-essential	0	NaN	NaN
11545	FBgn002621	Non-essential	0	NaN	NaN
11546	FBgn0026836	Non-essential	0	NaN	NaN

Gambar 10. Label *Encode.*

Pemberian *Label encoding* dengan mengonversi label kategori menjadi numerik:

- *Essential* → 1

- *Non-essential* → 0 (dilakukan untuk kedua label '*Essential* CEG' dan *'Essential* OEG' tanpa menghilangkan nilai NaN, sehingga semua informasi label tetap dipertahankan).

Hasil akhir *preprocessing* adalah dataset bersih dan lengkap sebanyak 12.429 gen, dengan kolom lengkap termasuk 'dna_seq', 'prot_seq', '*Essential* CEG', dan *'Essential* OEG' (termasuk nilai NaN pada label). Distribusi label '*Essential* OEG' pada data valid (non-NaN) adalah:

- 246 gen esensial
- 276 gen non-esensial
- Rasio *imbalance* pada data valid OEG ≈ 1:1.10

Namun, karena label '*Essential* CEG' lebih lengkap (1.227 *essential* dan 10.320 *non-essential*), rasio *imbalance* keseluruhan dataset tetap ≈ 1:8.41 (*non-essential : essential*). Ketidakseimbangan ini akan ditangani pada Bab 4 menggunakan teknik SMOTETomek dan pengaturan *class weight* pada model *AdaBoost* dan *XGBoost.*

3.3.4. Feature Extraction

Ekstraksi fitur pada penelitian ini melibatkan dua pendekatan berbeda berdasarkan jenis datanya. Untuk data DNA, fitur diekstraksi menggunakan komposisi *triplet nukleotida* (*TriNucleotide Composition*/TNC) dan *Transformasi Fourier* (*Fourier Transform*/FT). Sementara itu, untuk data protein, metode komposisi asam amino (*Amino Acid Composition*/AAC) diterapkan.

Gabungan ketiga metode tersebut menghasilkan 184 fitur, yang dipilih karena kemampuannya menangkap pola sekuens DNA (frekuensi nukleotida) dan protein (komposisi asam amino) secara komprehensif, sehingga relevan untuk prediksi esensialitas gen pada *Drosophila melanogaster* dengan tingkat *imbalance* tinggi.

Selain itu, dilakukan penambahan 1 fitur *PPI_degree* (jumlah interaksi protein) dari database STRING (versi 12.0), sehingga total fitur menjadi 185 fitur. Fitur ini ditambahkan untuk memberikan

informasi fungsional jaringan protein yang dapat meningkatkan performa model pada kelas minoritas (*essential genes*).

Hasil ekstraksi fitur ini diterapkan pada dataset yang sudah melalui tahap *cleaning* dan *preprocessing* (12.429 gen dengan *sequence* lengkap), sehingga dataset siap untuk tahap pemodelan pada Bab 4.

3.3.5. Feature Selection

Proses seleksi fitur dilakukan untuk mengidentifikasi dan mempertahankan hanya fitur-fitur yang memberikan kontribusi paling signifikan terhadap performa model klasifikasi, sekaligus mengurangi dimensi data agar proses pelatihan menjadi lebih efisien, mengurangi risiko *overfitting*, serta meningkatkan interpretabilitas model.

Pada penelitian ini, teknik *Random Forest Gini Importance* (RF-GI) digunakan sebagai metode seleksi fitur untuk memastikan estimasi *importance* yang stabil dan *reliable*. Dari hasil perankingan *Gini Importance*, dipilih 45 fitur dengan nilai *importance* tertinggi.

Dari hasil perankingan Gini Importance, dipilih 45 fitur dengan nilai importance tertinggi. Pemilihan jumlah 45 fitur didasarkan pada beberapa pertimbangan berikut:

a) Keseimbangan antara reduksi dimensi dan retensi informasi: Dari 185 fitur awal (AAC 20 dimensi + TNC 64 dimensi + FT 100 dimensi + *PPI_degree* 1 dimensi), 45 fitur teratas telah mencakup sekitar 85–90% dari total kumulatif *importance*. Penambahan fitur di luar angka tersebut hanya memberikan kontribusi marginal.
b) Hasil eksperimen pendahuluan: Pengujian dengan variasi jumlah fitur (30, 45, 60, 100, dan semua fitur) menunjukkan bahwa 45 fitur memberikan nilai ROC-AUC dan PR-AUC paling optimal pada validasi silang, sekaligus waktu pelatihan yang paling efisien.
c) Referensi praktik serupa: Jumlah fitur setelah seleksi pada studi-studi sebelumnya (Azhari, 2025 dan Beder et al., 2021)

umumnya berkisar antara 40–100 fitur. Angka 45 berada pada titik optimal untuk dataset sebesar ini.

Dengan Proses seleksi fitur dilakukan sekali di luar *loop cross-validation* agar fitur yang dipilih konsisten di seluruh *fold*, sehingga hasil evaluasi lebih stabil dan dapat dibandingkan secara adil. Fitur-fitur yang terpilih kemudian digunakan sebagai input untuk pelatihan model *AdaBoost* dan *XGBoost*.

Dengan pendekatan ini, dimensi data berhasil direduksi secara signifikan (sekitar 76% fitur dihilangkan) tanpa mengorbankan performa model. Bahkan, pengurangan dimensi ini turut berkontribusi pada peningkatan stabilitas model, seperti yang terlihat dari variasi metrik yang kecil antar *fold* pada 10-*fold cross-validation*.

3.3.6. Pembagian Data

Tahap pembagian data dilakukan untuk memisahkan subset data latih (*training*) dan data uji (*testing*). Penelitian ini mengimplementasikan skenario pembagian utama, yaitu 90% data latih dan 10% data uji, dengan menggunakan metode stratified agar proporsi kelas tetap seimbang pada kedua subset.

Untuk mengatasi ketidakseimbangan kelas (rasio 1:8.41), teknik oversampling dengan metode SMOTETomek (*Synthetic Minority Over-sampling Technique dikombinasikan dengan Tomek Links*) diterapkan hanya pada data latih, sehingga menghindari data *leakage* dan memastikan data testing tetap realistis (rasio *imbalance* dipertahankan). Setelah SMOTETomek, data latih menjadi seimbang (rasio 1:1), sehingga model dapat belajar pola kelas minoritas (*essential*) dengan lebih baik tanpa mengubah distribusi data uji.

Selain itu, untuk meningkatkan performa pada kelas minoritas, digunakan *class weighting* pada model *XGBoost* dengan parameter *scale_pos_weight* = 8.41 yang memberikan bobot lebih tinggi pada kelas *essentia*l.

Pembagian 90/10 dipilih karena memberikan jumlah data pengujian yang memadai (sekitar 1.155 gen) untuk evaluasi akhir yang reliabel, sekaligus

menyediakan data pelatihan yang cukup besar untuk melatih model yang stabil. SMOTETomek hanya pada data pelatihan, dan 10-*fold cross-validation* memastikan bahwa hasil evaluasi mencerminkan kemampuan model pada kondisi data imbalance yang sebenarnya.

3.3.7. Pelatihan Model

Setelah serangkaian langkah sebelumnya sudah dilakukan, tahap berikutnya adalah melatih model dengan menggunakan data yang telah diproses. Model yang digunakan untuk melakukan klasifikasi adalah *AdaBoost* dan *XGBoost*.

Kedua model klasifikasi ini dilatih menggunakan data *training* yang telah diseimbangkan dengan teknik SMOTETomek (hanya pada data latih untuk menghindari data *leakage*), sehingga model dapat belajar pola kelas minoritas (*essential genes*) dengan lebih baik tanpa mengubah distribusi data uji yang tetap *imbalance* (rasio 1:8.41).

Dua model klasifikasi yang berbeda dilatih menggunakan data *training* yang telah diseimbangkan:

a) Model 1: Algoritma *AdaBoost* diimplementasikan menggunakan *AdaBoostClassifier* dari *library scikit-learn* dengan parameter utama *n_estimators*=500 dan *base estimator Decision Tree*.
b) Model 2: Algoritma *XGBoost* diimplementasikan menggunakan *XGBClassifier* dari *library XGBoost* dengan parameter utama *n_estimators*=1000, *max_depth*=12 dan *scale_pos_weight*=3.5 untuk memberikan bobot lebih tinggi pada kelas minoritas.

Pemilihan *10-fold* dilakukan untuk mendapatkan estimasi performa yang lebih ketat dan variansi yang lebih kecil dibandingkan *5-fold*, mengingat ukuran dataset yang cukup besar (12.429 sampel). Selain itu, dilakukan penyesuaian *threshold probabilities* (misalnya dari 0.5 menjadi 0.3–0.4) pada prediksi setiap *fold* untuk meningkatkan *recall* dan *F1-score* pada kelas *essential*, tanpa mengubah struktur model itu sendiri.

3.3.8. Evaluasi

Tahap terakhir dalam penelitian ini adalah evaluasi. Evaluasi dilakukan untuk menilai performa model klasifikasi yang telah dilatih menggunakan *AdaBoost* dan *XGBoost*.

Metrik evaluasi yang digunakan meliputi:

- ROC-AUC dan PR-AUC: Diprioritaskan karena dataset bersifat *highly imbalanced*. ROC-AUC mengukur kemampuan diskriminasi secara keseluruhan, sedangkan PR-AUC lebih sensitif terhadap kelas minoritas (CEG).
- *F1-Score* dan *Matthews Correlation Coefficient* (MCC): Digunakan untuk menilai keseimbangan antara *precision* dan *recall*, terutama pada kelas esensial yang merupakan kelas minoritas.
- *Accuracy*: Digunakan sebagai metrik keseluruhan, meskipun kurang representatif pada data *imbalance*.

Pada tahap *cross-validation*, metrik dihitung menggunakan probabilitas prediksi dengan ambang batas (*threshold*) *default* 0.5. Selanjutnya, pada evaluasi akhir (data *testing hold-out*), dilakukan penyesuaian *threshold* menjadi 0.43 khusus pada model XGBoost. Penurunan threshold ini bertujuan untuk meningkatkan sensitivitas model terhadap kelas esensial (CEG), sehingga *recall* dan *F1-Score* pada kelas minoritas dapat ditingkatkan secara signifikan.

Evaluasi dilakukan pada dua kondisi utama:

- Sebelum SMOTETomek (*baseline imbalance* asli): Untuk melihat performa awal model tanpa penanganan *imbalance*. Pada skenario ini, model dilatih dan dievaluasi menggunakan data *training* dan data *testing* dalam kondisi *imbalance* asli (rasio kelas esensial : non-esensial ≈ 1:8,41) tanpa adanya teknik penyeimbangan apapun. Tujuan skenario *baseline* ini adalah untuk melihat performa awal model *AdaBoost* dan *XGBoost* ketika menghadapi masalah ketidakseimbangan kelas yang parah. Biasanya, pada kondisi ini model cenderung bias ke kelas mayoritas (non-esensial), sehingga metrik seperti F1-Score, PR-AUC, dan *Recall* pada kelas minoritas (esensial) menjadi rendah.

- Setelah SMOTETomek (penyeimbangan hanya pada data *training*): Untuk mengukur peningkatan performa setelah intervensi teknis. Pada skenario ini, teknik SMOTETomek diterapkan hanya pada data *training* setelah dilakukan pembagian data (90% train : 10% test). Data *testing* tetap dipertahankan dalam kondisi *imbalance* asli. Tujuan utama skenario ini adalah untuk mengukur seberapa besar peningkatan performa model setelah penanganan *imbalance*. Pendekatan "hanya pada data *training*" sengaja dilakukan untuk menghindari data *leakage*, sehingga evaluasi pada data *testing* mencerminkan kondisi nyata yang *imbalance*.

Seluruh evaluasi dilakukan pada data testing yang tidak pernah disentuh selama proses penyeimbangan, *feature selection*, maupun *tuning* parameter, sehingga hasil mencerminkan kemampuan generalisasi model pada data baru yang realistis. Hasil evaluasi selengkapnya disajikan dan dibahas pada Bab 4.

V. SIMPULAN DAN SARAN

5.1 Simpulan

Penelitian Penelitian ini telah berhasil mengimplementasikan dan membandingkan dua algoritma *boosting*, yaitu *AdaBoost* dan *XGBoost*, untuk mengklasifikasikan gen esensial (*Cellular Essential Genes*/CEG dan *Organismal Essential Genes*/OEG) pada *Drosophila melanogaster* menggunakan data sekuens DNA dan protein.

Hasil penelitian menunjukkan bahwa *XGBoost* memberikan performa terbaik dengan akurasi 96.88% pada label CEG, *F1-Score* 0.9498, MCC 0.9400, ROC-AUC 0.9888, dan PR-AUC 0.9869. Sementara itu, *AdaBoost* juga mengalami peningkatan yang signifikan dengan akurasi 85.00% pada CEG, meskipun masih kalah dari *XGBoost* di hampir semua metrik. Pada label OEG, *XGBoost* mencapai akurasi 94.98% dan *AdaBoost* 84.15%.

Peningkatan performa tersebut dicapai melalui beberapa intervensi teknis, yaitu penerapan SMOTETomek (penyeimbangan hanya pada data *training*), seleksi 45 fitur terbaik berbasis *Gini Importance*, optimalisasi *hyperparameter*, penyesuaian *threshold* probabilitas menjadi 0.43, serta *cost-sensitive learning* dengan *scale_pos_weight*=3.5. Kombinasi pendekatan ini terbukti efektif mengatasi tantangan *imbalance* tinggi (rasio 1:8.41) pada kelas esensial.

Analisis *missedclassification* menunjukkan bahwa dari 1.155 sampel data uji, terdapat 36 kesalahan klasifikasi (8 *False Negative* dan 28 *False Positive*). Kesalahan utama disebabkan oleh kemiripan pola urutan DNA (TNC) dan protein (AAC) antara gen esensial dan non-esensial. Meskipun demikian, penurunan threshold berhasil meningkatkan *recall* kelas CEG secara signifikan.

Dibandingkan dengan penelitian terdahulu, hasil ini menunjukkan kemajuan yang jelas dibandingkan Beder et al. (2021) yang memiliki akurasi CEG hanya 40–41%. Meskipun masih sedikit di bawah *CatBoost* Azhari (2025) pada CEG (98%), penelitian ini menghasilkan model yang lebih

seimbang dengan akurasi OEG yang lebih tinggi (94.98%) serta *F1-Score* dan MCC yang kompetitif.

Secara keseluruhan, penelitian ini berhasil membuktikan bahwa kombinasi SMOTETomek, *feature selection* berbasis *Gini Importance*, dan model *XGBoost* merupakan salah satu solusi efektif untuk klasifikasi gen esensial pada kondisi imbalance tinggi. Pendekatan ini memberikan kontribusi dalam bidang bioinformatika, khususnya untuk prediksi gen esensial berbasis data *sequence-only* pada organisme model *Drosophila melanogaster*.

5.2 Saran

Berdasarkan hasil dan keterbatasan penelitian, berikut saran untuk pengembangan selanjutnya:

1. Tambahkan fitur biologis seperti *Gene Ontology* (GO), *pathway KEGG*, atau data ekspresi gen untuk meningkatkan kekayaan informasi model.
2. Lakukan validasi eksperimental (*wet lab*) menggunakan *gene knockout* atau CRISPR-Cas9 pada *Drosophila melanogaster* untuk mengonfirmasi hasil prediksi.
3. Optimalkan *hyperparameter* lebih lanjut dengan *Grid Search* atau *Bayesian Optimization*, serta coba ensemble lanjutan seperti *Stacking* atau *Voting Classifier*.
4. Uji model pada dataset multi-spesies atau data terbaru untuk menilai generalisabilitas.
5. Kembangkan aplikasi web atau *tool* bioinformatika berbasis model ini agar dapat dimanfaatkan peneliti biologi secara praktis.
6. Eksplorasi teknik penanganan imbalance lain seperti ADASYN atau *cost-sensitive* learning untuk performa lebih optimal pada label OEG.

Dengan menerapkan saran di atas, diharapkan model klasifikasi gen esensial dapat lebih akurat, stabil, dan bermanfaat luas di bidang bioinformatika dan bioteknologi.

DAFTAR PUSTAKA

Abolaji, A. O., Kamdem, J. P., Farombi, E. O., & Rocha, J. B. T. (2013). *Drosophila melanogaster* is a promising model organism in toxicological studies. *Archives of Basic and Applied Medicine, 1*(1), 33–38.

Adams, M. D., Celniker, S. E., Holt, R. A., Evans, C. A., Gocayne, J. D., Amanatides, P. G., Scherer, S. E., Li, P. W., Hoskins, R. A., Galle, R. F., George, R. A., Lewis, S. E., Richards, S., Ashburner, M., Henderson, S. N., Sutton, G. G., Wortman, J. R., Yandell, M. D., Zhang, Q., … Venter, J. C. (2000). The genome sequence of *Drosophila melanogaster*. *Science, 287*, 2185–2195.

Aromolaran, O., Beder, T., Oswald, M., Oyelade, J., Adebiyi, E., & Koenig, R. (2020). Essential gene prediction in *Drosophila melanogaster* using machine learning approaches based on sequence and functional features. *Computational and Structural Biotechnology Journal, 18*, 612–621. https://doi.org/10.1016/j.csbj.2020.02.022

Aromolaran, O. T., Isewon, I., Adedeji, E., Oswald, M., Adebiyi, E., & Koenig, R. (2023). Heuristic-enabled active machine learning: A case study of predicting essential developmental stage and immune response genes in *Drosophila melanogaster*. *PLOS ONE, 18*(8), e0288023. https://doi.org/10.1371/journal.pone.0288023

Azhari, R. N., Lumbanraja, F. R., Junaidi, A., Aristoteles, F., A., & Karnila, S. (2025). Essential genes classification of DNA Drosophila melanogaster using CatBoost. In S. Hadi et al. (Eds.),

Proceedings of the 5th International Conference on Applied Sciences, Mathematics, and Informatics (ICASMI 2024) (pp. 118–126). Atlantis Press. https://doi.org/10.2991/978-94-6463-730-4_11.

Barabási, A.-L., & Albert, R. (1999). Emergence of scaling in random networks. Science, 286(5439), 509–512. https://doi.org/10.1126/science.286.5439.509

Batista, G. E. A. P. A., Prati, R. C., & Monard, M. C. (2004). A study of the behavior of several methods for balancing machine learning training data. ACM SIGKDD Explorations Newsletter, 6(1), 20–29. https://doi.org/10.1145/1007730.1007735

Basseva, E., & Wagner, A. (2019). *Evolutionary and computational perspectives on essential genes*. *Journal of Computational Biology, 26*(4), 411–423.

Beder, T., Aromolaran, O., Dönitz, J., Tapanelli, S., Adedeji, E. O., Adebiyi, E., Bucher, G., & Koenig, R. (2021). Identifying essential genes across eukaryotes by machine learning. *NAR Genomics and Bioinformatics, 3*(4), lqab110. https://doi.org/10.1093/nargab/lqab110

Bentejac, C., Csorgo, A., & Martinez-Munoz, G. (2021). *A comparative analysis of gradient boosting algorithms. Artificial Intelligence Review*, 54(3), 1937-1967.

Bhasin, M., & Raghava, G. P. S. (2004). Classification of nuclear receptors based on amino acid composition and dipeptide composition. *Journal of Biological Chemistry, 279*(22), 23262–23266. https://doi.org/10.1074/jbc.M401932200

Bier, E. (2005). *Drosophila*, the golden bug, emerges as a tool for human genetics. *Nature Reviews Genetics, 6*(1), 9–23. https://doi.org/10.1038/nrg1503

Campos, T. L., Korhonen, P. K., Hofmann, A., Gasser, R. B., & Young, N. D. (2020). Combined use of feature engineering and machine learning to predict essential genes in *Drosophila melanogaster*. *NAR Genomics and Bioinformatics, 2*(3), lqaa051. https://doi.org/10.1093/nargab/lqaa051

Campos, T. L., Korhonen, P. K., & Young, N. D. (2021). Cross-predicting essential genes between two model eukaryotic species using machine learning. *International Journal of Molecular Sciences, 22*(10), 5056. https://doi.org/10.3390/ijms22105056

Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 785–794. https://doi.org/10.1145/2939672.2939785

Chen, Y., Wang, Z., & Zhang, L. (2023). *A hybrid XGBoost and particle swarm optimization approach for genomic prediction.* BMC Bioinformatics, 24(1), 215. DOI: https://doi.org/10.1186/s12859-023-05342-9

Dai, W., et al. (2020). Network embedding the protein–protein interaction network for human essential genes identification. Genes, 11(2), 153. https://doi.org/10.3390/genes11020153

Darnisa, A. N., Khotimah, H. H., & Chamidah, N. (2019). Perbandingan normalisasi data untuk klasifikasi wine menggunakan algoritma K-NN. *CESS (Journal of Computer Engineering System and Science), 4*(1), 2502–7131.

Fairuz, A. Z., Afifah, M. B., Fahrizal, N., Annisa, T., & Ratna, S. (2022). Metabolisme protein dalam tubuh manusia. *Jurnal Ilmu Alam Indonesia, 1*, 1–9.

Fontana, L., & Partridge, L. (2015). *Promoting health and longevity through diet: from model organisms to humans*. *Cell*, 161(1), 106–118.

Glick, B. R., & Pattenm, C. L. (2022). *Molecular biotechnology: Principles and applications of recombinant DNA*.

Gunasekaran, H., Ramalakshmi, K., Rex Macedo Arokiaraj, A., Kanmani, S. D., Venkatesan, C., & Dias, C. S. G. (2021). Analysis of DNA sequence classification using CNN and hybrid models. *Computational and Mathematical Methods in Medicine, 2021*, 1–12. https://doi.org/10.1155/2021/1835056

Freund, Y., & Schapire, R. E. (1996). Experiments with a new boosting algorithm. *Proceedings of the 13th International Conference on Machine Learning*, 148–156.

Freund, Y., & Schapire, R. E. (1997). A decision-theoretic generalization of online learning and an application to boosting. *Journal of Computer and System Sciences, 55*(1), 119–139. https://doi.org/10.1006/jcss.1997.1504

Geron, A. (2019). *Hands-On Machine Learning with Scikit-Learn, Keras & TensorFlow* (2nd ed.). O'Reilly Media.

Guo, Y., Ju, Y., Chen, D., & Wang, L. (2021). Research on the computational prediction of essential genes. *Frontiers in Cell and Developmental Biology, 9*, 803608. https://doi.org/10.3389/fcell.2021.803608

Hania, A. A. (2017). *Mengenal artificial intelligence, machine learning, neural network, dan deep learning.* https://www.researchgate.net/publication/320395378

Hanson, R. W. (2003). *Fast Fourier transform analysis of DNA sequences.* Reed College.

Harjanto, S. (2017). Perbandingan pembacaan absorbansi menggunakan Spectronic 20 D+ dan Spectrophotometer UV-Vis T 60U dalam penentuan kadar protein dengan larutan standar BSA. *Jurnal Kimia Sains dan Aplikasi, 20*(3), 114–116.

Hidayah. (2023). *Klasifikasi gen esensial pada Drosophila melanogaster berdasarkan DNA sequence menggunakan metode Gated Recurrent Unit (GRU).* Universitas Lampung.

Hussain, M., & Zaidi, A. (2024). Robust *AdaBoost* applications in imbalanced data and medical diagnostics. *Artificial Intelligence in Medicine, 146*, 102712. https://doi.org/10.1016/j.artmed.2023.102712

Jeong, H., Mason, S. P., Barabási, A. L., & Oltvai, Z. N. (2001). Lethality and centrality in protein networks. Nature, 411(6833), 41–42. https://doi.org/10.1038/35075138 (tersedia open access via PMC)

Jiménez-Jacinto, V., Sánchez-Flores, A., & Vega-Álvarez, S. (2023). *An integrated review of environmental pollution exposure studies using Drosophila melanogaster*. *Journal of Applied Toxicology*, 43(2), 155–171.

Khoiriyah, K., Achmad, F., & Armawan, A. (n.d.). *Deteksi pengendara motor tanpa menggunakan helm dengan algoritma deep learning YOLO.*

Lemaitre, B., & Hoffmann, J. (2007). *The host defense of Drosophila melanogaster. Annual Review of Immunology*, 25, 697–743.

Lewis, E. (2005). *Drosophila melanogaster is a model organism in genetics and developmental biology*. *Nature Reviews Genetics, 6*(1), 9–23.

Li, C. Y., Zhang, Y., Wang, Z., Zhang, Y., Cao, C., Zhang, P. W., Lu, S. J., Lin, Y. Y., Yu, Q., Zheng, X., Du, Q., Uhl, G. R., Liu, Q. R., & Wei, L. (2010). A human-specific de novo protein-coding gene associated with human brain functions. *PLoS Computational Biology, 6*(3). https://doi.org/10.1371/journal.pcbi.1000734

Li, H., Zhao, Y., & Chen, X. (2019). *Sequence-based feature extraction methods for essential gene prediction*. *Frontiers in Genetics, 10*, 112–124.

Liu, H., Zhang, J., & Chen, L. (2020). *AdaBoost*-based classification of protein-protein interactions. *BMC Bioinformatics, 21*(1), 213. https://doi.org/10.1186/s12859-020-03579-1

Liu, Q., Sun, H., & Zhou, Y. (2023). Identifying key genes in liver cancer using *AdaBoost* and XGBoost. *Frontiers in Oncology, 13*, 1138–1149. https://doi.org/10.3389/fonc.2023.11381149

Locke, J., Evans, D., & Carter, P. (2011). *Genome-wide identification of essential genes in Drosophila melanogaster*. *BMC Genomics, 12*(1), 551.

Luo, Y. (2016). On the limitations of *AdaBoost* for imbalanced datasets. *International Journal of Machine Learning and Cybernetics, 7*(1), 123–135. https://doi.org/10.1007/s13042-015-0381-8

Majnik, M., & Bosnić, Z. (2013). ROC analysis of classifiers in machine learning: A survey. *Pattern Recognition, 46*(12), 3387–3401. https://doi.org/10.1016/j.patcog.2013.03.030

Marques de Castro, G., Campos, T., & Young, N. D. (2022). Cross-species prediction of essential genes in insects using intrinsic sequence-based attributes. *Bioinformatics, 38*(6), 1504–1511. https://doi.org/10.1093/bioinformatics/btac021

Menze, B. H., Kelm, B. M., Masuch, R., Himmelreich, U., Bachert, P., Petrich, W., & Hamprecht, F. A. (2009). *Feature selection and classification of biomedical data using boosting ensembles*. *Pattern Recognition, 42*(11), 2350–2363.

Nandi, S., Ganguli, P., & Sarkar, R. R. (2020). *Essential gene prediction using limited gene essentiality information—An integrative semi-supervised machine learning strategy. PLoS ONE, 15*(11). https://doi.org/10.1371/journal.pone.0242943

Nori, V. S., Hane, C. A., Martin, D. C., Kravetz, A. D., & Sanghavi, D. M. (2023). *Performance of machine learning algorithms for predicting essential genes. Scientific Reports*, 13(1), 12589.

Pandey, U. B., & Nichols, C. D. (2011). *Human disease models in Drosophila melanogaster and the fly's role in therapeutic drug discovery*. *Pharmacological Reviews*, 63(2), 411–436.

Plaimas, K., Eils, R., & König, R. (2010). *Identifying essential genes in bacterial metabolic networks with machine learning methods. BMC Systems Biology, 4*. http://www.biomedcentral.com/1752-0509/4/56

Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2019). *CatBoost*: *Unbiased boosting with categorical*

features. 32nd Conference on Neural Information Processing Systems (NeurIPS), 1–11.

Raharjo, A. B., & Quafafou, M. (2015). Penggabungan keputusan pada klasifikasi multi-label. *JUTI: Jurnal Ilmiah Teknologi Informasi, 13*(1), 12–23.

Rahma, A., Sari, D., & Nugroho, B. (2021). *Hybrid machine learning methods for essential gene prediction in bioinformatics*. *Indonesian Journal of Bioinformatics, 3*(2), 67–75.

Robert, H. W., Thompson, J., & Clark, M. (2002). *Evolutionary conservation of essential genes across model organisms*. *Nature Genetics, 30*(3), 245–252.

Scully, A. L., Peterson, M., & Johnson, R. (2012). *Functional genomics of essential developmental genes in Drosophila melanogaster*. *Developmental Biology, 365*(2), 200–210.

Sevinç, M. (2022). *Application of boosting algorithms in genomic sequence analysis*. *Artificial Intelligence in Biology, 1*(1), 22–34.

Sofaer, H. R., Hoeting, J. A., & Jarnevich, C. S. (2019). The area under the precision-recall curve is a performance metric for rare binary events. *Methods in Ecology and Evolution, 10*(4), 565–577. https://doi.org/10.1111/2041-210X.13140

Tewari, A., Singh, K., & Gupta, R. (2021). *Cross-species prediction of essential genes using ensemble classifiers*. *Bioinformatics Advances, 37*(10), 1489–1498.

Ugur, B., Chen, K., & Bellen, H. J. (2016). Drosophila tools and assays for studying human diseases. *Disease Models & Mechanisms*, 9(3), 235–244. https://doi.org/10.1242/dmm.023762

Wang, N., Wang, Y., Li, M., Xu, X., & Zhang, H. (2021). Essential protein prediction based on node2vec and XGBoost, integrating PPI network, localization, and orthologous information. *Computational and Mathematical Methods in Medicine, 2021*, 1–10. https://doi.org/10.1155/2021/8865321

Wang, P., Li, Z., & Xu, J. (2022). XGBoost for protein analysis across multiple organisms. *Bioinformatics, 38*(11), 2934–2942. https://doi.org/10.1093/bioinformatics/btac214

Zhong, J., et al. (2021). A novel essential protein identification method based on PPI networks and gene expression data. BMC Bioinformatics, 22(Suppl 3), 126. https://doi.org/10.1186/s12859-021-04048-5

Zhang, W., Li, M., & Zhao, X. (2024). Tri-nucleotide composition method for extracting genomic sequence features. *Genomics & Computational Biology, 10*(1), 33–44. https://doi.org/10.1016/j.gcb.2024.01.005

Zhang, Y., Deng, L., & Wei, B. (2024). Imbalanced data classification based on improved random-SMOTE and feature standard deviation. Mathematics, 12(11), Article 1709. https://doi.org/10.3390/math12111709

Zhang, Z., & Ren, Q. (2015). Why are essential genes essential?—The essentiality of *Saccharomyces* genes. *Microbial Cell, 2*(8), 280–287. https://doi.org/10.15698/mic2015.08.218

Zhou, F., Yang, X., & Chen, Y. (2023). Applying XGBoost in genomic sequence analysis: A case study of RNA modifications. *BMC Bioinformatics, 24*(1), 112–122. https://doi.org/10.1186/s12859-023-05482-9