MitoChime: A Machine Learning Pipeline for Detecting PCR-Induced Chimeras in Mitochondrial Illumina Reads

A Special Project Proposal
Presented to
the Faculty of the Division of Physical Sciences and Mathematics
College of Arts and Sciences
University of the Philippines Visayas
Miagao, Iloilo

In Partial Fulfillment
of the Requirements for the Degree of
Bachelor of Science in Computer Science

by

Duranne Duran
Yvonne Lin
Daniella Pailden

Adviser
Francis D. Dimzon, Ph.D.

January 3, 2026

Abstract

Next-generation sequencing (NGS) platforms have advanced research but remain susceptible to artifacts such as PCR-induced chimeras that compromise mitochondrial genome assembly. These artificial hybrid sequences are problematic for small, circular, and repetitive mitochondrial genomes, where they can generate fragmented contigs and false junctions. Existing detection tools, such as UCHIME, are optimized for amplicon-based microbial community analysis and depend on reference databases or abundance assumptions unsuitable for organellar assembly. To address this gap, this study presents MitoChime, a machine learning pipeline for detecting PCR-induced chimeric reads in *Sardinella lemuru* Illumina paired-end data without relying on external reference databases.

Using simulated datasets containing clean and chimeric reads, a feature set was extracted, combining alignment-based metrics (e.g., supplementary alignments, soft-clipping) with sequence-derived statistics (e.g., k-mer composition, microhomology). A comparative evaluation of supervised learning models identified tree-based ensembles CatBoost and Gradient Boosting as top performers, achieving an F1-score of 0.77 and an ROC-AUC of 0.84 on held-out test data. Feature importance analysis highlighted soft-clipping and k-mer compositional shifts as the strongest predictors of chimerism, whereas microhomology contributed minimally. Integrating MitoChime as a pre-assembly step can aid in streamlining mitochondrial reconstruction pipelines.

Keywords: Chimera detection, Mitochondrial genome, Assembly, Machine learning

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

The rapid advancement of next-generation sequencing (NGS) technologies has transformed genomic research by enabling high-throughput and cost-effective DNA analysis (Metzker, 2010). Among current platforms, Illumina sequencing remains the most widely adopted, capable of producing millions of short reads that can be assembled into reference genomes or analyzed for genetic variation (Bentley et al., 2008; Glenn, 2011). Despite its high base-calling accuracy, Illumina sequencing is prone to artifacts introduced during library preparation, particularly polymerase chain reaction (PCR)-induced chimeras, which are artificial hybrid sequences that do not exist in the true genome (Judo, Wedel, & Wilson, 1998).

PCR chimeras form when incomplete extension products from one template anneal to an unrelated DNA fragment and are extended, creating recombinant reads (Qiu et al., 2001). In mitochondrial genome assembly, such artifacts are especially problematic because the mitochondrial genome is small, circular, and often repetitive (Boore, 1999; Cameron, 2014). Even a small number of chimeric or misjoined reads can reduce assembly contiguity and introduce false junctions during organelle genome reconstruction (Dierckxsens, Mardulyn, & Smits, 2017; Hahn, Bachmann, & Chevreux, 2013; Jin et al., 2020). Existing assembly tools such as GetOrganelle and MITObim assume that input reads are largely free of such artifacts (Hahn et al., 2013; Jin et al., 2020). Consequently, undetected chimeras may produce fragmented assemblies or misidentified organellar boundaries. To ensure accurate reconstruction of mitochondrial genomes, a reliable method for detecting PCR-induced chimeras before assembly is essential.

This study focuses on mitochondrial sequencing data from the genus *Sardinella*, a group of small pelagic fishes widely distributed in Philippine waters. Among them, *Sardinella lemuru* (Bali sardinella) is one of the country's most abundant and economically important species, providing protein and livelihood to coastal communities (Labrador, Agmata, Palermo, Ravago-Gotanco, & Pante, 2021; Willette, Bognot, Mutia, & Santos, 2011). Accurate mitochondrial assemblies are critical for understanding its population genetics, stock structure, and evolutionary history. However, assembly pipelines often encounter errors or fail to complete due to undetected chimeric reads. To address this gap, this research introduces MitoChime, a machine learning pipeline designed to detect PCR-induced chimeric reads using both alignment-based and sequence-derived statistical features. The tool aims to provide bioinformatics laboratories, particularly the Philippine Genome Center Visayas (PGC Visayas), with an efficient solution for improving mitochondrial genome reconstruction.

1.2 Problem Statement

Chimeric reads can distort assembly graphs and cause misassemblies, with particularly severe effects in mitochondrial genomes (Boore, 1999; Cameron, 2014). Existing assembly pipelines such as GetOrganelle, MITObim, and NOVOPlasty assume that sequencing reads are free of such artifacts (Dierckxsens et al., 2017; Hahn et al., 2013; Jin et al., 2020). At PGC Visayas, several mitochondrial assemblies have failed or yielded incomplete contigs despite sufficient coverage, suggesting that undetected chimeric reads compromise assembly reliability. Meanwhile, existing chimera detection tools such as UCHIME and VSEARCH were developed primarily for amplicon-based community analysis and rely heavily on reference or taxonomic comparisons (Edgar, Haas, Clemente, Quince, & Knight, 2011; Rognes, Flouri, Nichols, Quince, & Mahé, 2016). These approaches are unsuitable for single-species organellar data, where complete reference genomes are often unavailable.

1.3 Research Objectives

1.3.1 General Objective

This study aims to develop and evaluate a machine learning-based pipeline (MitoChime) that detects PCR-induced chimeric reads in *Sardinella lemuru* mitochondrial sequencing data in order to improve the quality and reliability of downstream mitochondrial genome assemblies.

1.3.2 Specific Objectives

Specifically, the study aims to:

1. construct simulated *Sardinella lemuru* Illumina paired-end datasets containing both clean and PCR-induced chimeric reads,
2. extract alignment-based and sequence-based features such as k-mer composition, junction complexity, and split-alignment counts from both clean and chimeric reads,
3. train, validate, and compare supervised machine learning models for classifying reads as clean or chimeric,
4. determine feature importance and identify indicators of PCR-induced chimerism,
5. integrate the optimized classifier into a modular and interpretable pipeline deployable on standard computing environments at PGC Visayas.

1.4 Scope and Limitations of the Research

This study focuses solely on PCR-induced chimeric reads in *Sardinella lemuru* mitochondrial sequencing data, with the species choice guided by four considerations: (1) to limit interspecific variation in mitochondrial genome size, GC content, and repetitive regions so that differences in read patterns can be attributed more directly to PCR-induced chimerism, (2) to align the analysis with relevant *S. lemuru* sequencing projects at PGC Visayas, (3) to take advantage of the availability of *S. lemuru* mitochondrial assemblies and raw datasets in public repositories such as the National Center for Biotechnology Information (NCBI), which facilitates reference selection and benchmarking, and (4) to develop a tool that directly supports local studies on *S. lemuru* population structure and fisheries management.

The study emphasizes `wgsim`-based simulations and selected empirical mitochondrial datasets from *S. lemuru*. It excludes naturally occurring chimeras, nuclear mitochondrial pseudogenes (NUMTs), and large-scale assembly rearrangements in nuclear genomes. Feature extraction is restricted to low-dimensional alignment and sequence statistics, such as k-mer frequency profiles, GC content, soft and hard clipping metrics, and split-alignment counts rather than high-dimensional deep learning embeddings. This design keeps model behaviour interpretable and ensures that the pipeline can be run on standard workstations at PGC Visayas. Testing on long-read platforms (e.g., Nanopore, PacBio) and other taxa is outside the scope of this project.

Other limitations in this study include the following: simulations with varying error rates were not performed, so the effect of different sequencing errors on model performance remains unexplored; alternative parameter settings, including k-mer lengths and microhomology window sizes, were not systematically tested, which could affect the sensitivity of both k-mer and microhomology feature detection; and the machine learning models rely on supervised training with labeled examples, which may limit their ability to detect novel or unexpected chimeric patterns.

1.5 Significance of the Research

This research provides both methodological and practical contributions to mitochondrial genomics and bioinformatics. First, MitoChime detects PCR-induced chimeric reads prior to genome assembly, with the goal of improving the contiguity and correctness of *Sardinella lemuru* mitochondrial assemblies. Second, it replaces informal manual curation with a documented workflow, improving automation and reproducibility. Third, the pipeline is designed to run on computing infrastructures commonly available in regional laboratories, enabling routine use at facilities such as PGC Visayas. Finally, more reliable mitochondrial assemblies for *S. lemuru* provide a stronger basis for downstream applications in the field of fisheries and genomics.

Chapter 2

Review of Related Literature

This chapter presents an overview of the literature relevant to the study. It discusses the biological and computational foundations underlying mitochondrial genome analysis and assembly, as well as existing tools, algorithms, and techniques related to chimera detection and genome quality assessment. The chapter aims to highlight the strengths, limitations, and research gaps in current approaches that motivate the development of the present study.

2.1 The Mitochondrial Genome

Mitochondrial genome (mtDNA) is a small, typically circular molecule found in most eukaryotes. It encodes essential genes involved in oxidative phosphorylation and energy metabolism. Because of its conserved structure, mtDNA has become a valuable genetic marker for studies in population genetics and phylogenetics (Anderson et al., 1981; Boore, 1999). In animal species, the mitochondrial genome ranges from 15–20 kilobase and contains 13 protein-coding genes, 22 tRNAs, and two rRNAs arranged compactly without introns (Gray, 2012). In comparison to nuclear DNA, the ratio of the number of copies of mtDNA is higher and has simple organization which make it particularly suitable for genome sequencing and assembly studies (Dierckxsens et al., 2017).

2.1.1 Mitochondrial Genome Assembly

Mitochondrial genome assembly refers to the reconstruction of the complete mitochondrial DNA (mtDNA) sequence from raw or fragmented sequencing reads. It is conducted to obtain high-quality, continuous representations of the mitochondrial genome that can be used for a wide range of analyses, including species identification, phylogenetic reconstruction, evolutionary studies, and investigations of mitochondrial diseases. Because mtDNA evolves rapidly, its assembled sequence provides valuable insights into population structure, lineage divergence, and adaptive evolution across taxa (Boore, 1999). Compared to nuclear genome assembly, assembling the mitochondrial genome is often considered more straightforward but still encounters technical challenges such as the formation of chimeric reads. Commonly used tools for mitogenome assembly such as GetOrganelle and MITObim operate under the assumption of organelle genome circularity, and are vulnerable when chimeric reads disrupt this circular structure, resulting in assembly errors (Hahn et al., 2013; Jin et al., 2020).

2.2 PCR Amplification and Chimera Formation

PCR plays an important role in NGS library preparation, as it amplifies target DNA fragments for downstream analysis. However as previously mentioned, the amplification process can also introduce chimeric reads which compromises the quality of the input reads supplied to sequencing or assembly workflows. Chimeras typically arise when incomplete extension occurs during a PCR cycle. This causes the DNA polymerase to switch from one template to another and generate hybrid recombinant molecules (Judo et al., 1998). Artificial chimeras are produced through such amplification errors, whereas biological chimeras occur naturally through genomic rearrangements or transcriptional events.

In the context of amplicon-based sequencing, the presence of chimeras can inflate estimates of genetic or microbial diversity and may cause misassemblies during genome reconstruction. Qin et al. (2023) has reported that chimeric sequences may account for more than 10% of raw reads in amplicon datasets. This artifact tends to be most prominent among rare operational taxonomic units (OTUs) or singletons, which are sometimes misinterpreted as novel diversity, further causing the complication of microbial diversity analyses (Gonzalez, Zimmermann, & Saiz-Jimenez, 2004). As such, determining and minimizing PCR-induced chimera formation is vital for improving the quality of mitochondrial genome assemblies, and ensuring the reliability of amplicon sequencing data.

2.3 Existing Traditional Approaches for Chimera Detection

Several computational tools have been developed to identify chimeric sequences in NGS datasets. These tools generally fall into two categories: reference-based and de novo approaches. Reference-based chimera detection, also known as database-dependent detection, is one of the earliest and most widely used computational strategies for identifying chimeric sequences in amplicon-based community studies. These methods rely on the comparison of each query sequence against a curated, high-quality database of known, non-chimeric reference sequences (Edgar et al., 2011).

On the other hand, the de novo chimera detection, also referred to as reference-free detection, represents an alternative computational paradigm that identifies chimeric sequences without reliance on external reference databases. This method infer chimeras based on internal relationships among the sequences present within the dataset itself, making it particularly advantageous in studies of under explored or taxonomically diverse communities where comprehensive reference databases are unavailable or incomplete (Edgar, 2016; Edgar et al., 2011). The underlying assumption on this method is that during PCR, true biological sequences are generally more abundant as they are amplified early and dominate the read pool, whereas chimeric sequences appear later and are generally less abundant. The de novo approach leverage this abundance hierarchy, treating the most abundant sequences as supposed parents and testing whether less abundant sequences can be reconstructed as mosaics of these templates. Compositional and structural similarity are also evaluated to check whether different regions of a candidate sequence correspond to distinct high-abundance sequences.

In practice, many modern bioinformatics pipelines combine both paradigms sequentially: an initial de novo step identifies dataset-specific chimeras, followed by a reference-based pass that removes remaining artifacts relative to established databases (Edgar, 2016). These two methods of detection form the foundation of tools such as UCHIME and later UCHIME2.

2.3.1 UCHIME

UCHIME is one of the most widely used tools for detecting chimeric sequences in amplicon-based studies and remains a standard quality-control step in microbial community analysis. Its core strategy is to test whether a query sequence (Q) can be explained as a mosaic of two parent sequences, (A and B), and to score this relationship using a structured alignment model (Edgar et al., 2011).

In reference mode, UCHIME divides the query into several segments and maps them against a curated database of non-chimeric sequences. Candidate parents are identified, and a three-way alignment is constructed. The algorithm assigns "Yes" votes when different segments of the query match different parents and "No" votes when the alignment contradicts a chimeric pattern. The final score reflects the balance of these votes. In de novo mode, UCHIME operationalizes the abundance-skew principle described earlier: high-abundance sequences are treated as candidate parents, and lower-abundance sequences are evaluated as potential mosaics. This makes the method especially useful when no reliable reference database exists.

Although UCHIME is highly sensitive, it faces key constraints. Chimeras formed from parents with very low divergence (below 0.8%) are difficult to detect because they are nearly indistinguishable from sequencing errors. Accuracy in reference mode depends strongly on database completeness, while de novo detection assumes that true parents are both present and sufficiently more abundant, such conditions are not always met.

2.3.2 UCHIME2

UCHIME2 extends the original algorithm with refinements tailored for high-resolution sequencing data. One of its major contributions is a re-evaluation of benchmarking practices. Edgar (2016) demonstrated that earlier accuracy estimates for chimera detection were overly optimistic because they relied on unrealistic scenarios where all true parent sequences were assumed to be present. Using the more rigorous CHSIMA benchmark, UCHIME2 showed the prevalence of "fake models" or real biological sequences that can be perfectly reconstructed as apparent chimeras of other sequences, which suggests that perfect chimera detection is theoretically unattainable. UCHIME2 also introduces several preset modes (e.g., denoised, balanced, sensitive, specific, high-confidence) designed to tune sensitivity and specificity depending on dataset characteristics. These modes allow users to adjust the algorithm to the expected noise level or analytical goals.

Despite these improvements, UCHIME2 must be applied with caution. The website manual explicitly advises against using UCHIME2 as a standalone chimera-filtering step in OTU clustering or denoising workflows because doing so can inflate both false positives and false negatives (Edgar, n.d.).

2.3.3 CATch

As previously mentioned, UCHIME (Edgar et al., 2011) relied on alignment-based sequences in amplicon data. However, researchers soon observed that different algorithms often produced inconsistent predictions. A sequence might be identified as chimeric by one tool but classified as non-chimeric by another, resulting in unreliable filtering outcomes across studies.

To address these inconsistencies, Mysara, Saeys, Leys, Raes, and Monsieurs (2015) developed the Classifier for Amplicon Tool Chimeras (CATCh), which represents the first ensemble machine learning system designed for chimera detection in 16S rRNA amplicon sequencing. Rather than depending on a single detection strategy, CATCh integrates the outputs of several established tools, including UCHIME, ChimeraSlayer, DECIPHER, Pintail, and Perseus. The individual scores and binary decisions generated by these tools are used as input features for a supervised learning model. The algorithm employs a Support Vector Machine (SVM) with a Pearson VII Universal Kernel (PUK) to determine optimal weightings among the input features and to assign each sequence a probability of being chimeric.

Benchmarking in both reference-based and de novo modes demonstrated significant performance improvements. CATCh achieved sensitivities of approximately 85 percent in reference-based mode and 92 percent in de novo mode, with corresponding specificities of approximately 96 percent and 95 percent. These results indicate that CATCh detected 7 to 12 percent more chimeras than any individual algorithm while maintaining high precision.

2.3.4 ChimPipe

Among the available tools for chimera detection, ChimPipe is a pipeline developed to identify chimeric sequences such as biological chimeras. It uses both discordant paired-end reads and split-read alignments to improve the accuracy and sensitivity of detecting biological chimeras (Rodriguez-Martin et al., 2017). By combining these two sources of information, ChimPipe achieves better precision than methods that depend on a single type of indicator.

The pipeline works with many eukaryotic species that have available genome and annotation data (Rodriguez-Martin et al., 2017). It can also predict multiple isoforms for each gene pair and identify breakpoint coordinates that are useful for reconstructing and verifying chimeric transcripts. Tests using both simulated and real datasets have shown that ChimPipe maintains high accuracy and reliable performance.

ChimPipe lets users adjust parameters to fit different sequencing protocols or organism characteristics. Experimental results have confirmed that many chimeric transcripts detected by the tool correspond to functional fusion proteins, demonstrating its utility for understanding chimera biology and its potential applications in disease research (Rodriguez-Martin et al., 2017).

2.4 Machine Learning Approaches for Chimera and Sequence Quality Detection

Traditional chimera detection tools rely primarily on heuristic or alignment-based rules. Recent advances in machine learning (ML) have demonstrated that models trained on sequence-derived features can effectively capture compositional and structural patterns in biological sequences. Although most existing ML systems such as those used for antibiotic resistance prediction, taxonomic classification, or viral identification are not specifically designed for chimera detection, they highlight how data-driven models can outperform similarity-based heuristics by learning intrinsic sequence signatures. In principle, ML frameworks can integrate indicators such as k-mer frequencies, GC-content variation and split-alignment metrics to identify subtle anomalies that may indicate a chimeric origin (Arango et al., 2018; Liang, Bible, Liu, Zou, & Wei, 2020; Ren et al., 2020).

2.4.1 Feature-Based Representations of Genomic Sequences

Feature extraction converts DNA sequences into numerical representations suitable for machine learning models. One approach is k-mer frequency analysis, which counts short nucleotide sequences within a read (Vervier, Mahé, Tournoud, Veyrieras, & Vert, 2015). High-frequency k-mers, including simple repeats such as “AAAAAA,” can highlight repetitive or unusual regions that may occur near chimeric junctions. Comparing k-mer patterns across adjacent parts of a read can help identify such regions, while GC content provides an additional descriptor of local sequence composition (Ren et al., 2020).

Alignment-derived features further inform junction detection. Long-read tools such as Sniffles (Sedlazeck et al., 2018) use split alignments to locate breakpoints across extended sequences, whereas short-read aligners like Minimap2 (Li, 2018) report supplementary and secondary alignments that indicate local discontinuities. Split alignments, where parts of a read map to different regions, can reveal template-switching events. These features complement k-mer profiles and enhance detection of potentially chimeric reads, even in datasets with incomplete references.

Microhomology, or short sequences shared between adjacent segments, is another biologically meaningful feature. Short microhomologies, typically 3–20 bp, are involved in template switching both in cellular repair pathways and during PCR, where they act as signatures of chimera formation (Peccoud et al., 2018; Sfeir & Symington, 2015). In PCR-induced chimeras, short identical sequences at junctions provide a clear signature of chimerism. Measuring the longest exact overlap at each breakpoint complements k-mer and alignment features and helps identify reads that are potentially chimeric.

2.5 Synthesis of Chimera Detection Approaches

To provide an integrated overview of the literature discussed in this chapter, Table 2.1 summarizes the major chimera detection studies, their methodological approaches, and their known limitations.

Table 2.1: Comparison of Chimera Detection Approaches and Tools

Method / Tool	**Core Approach**	**Key Limitations**
Reference-based Detection	Compares each query sequence against curated databases of verified, non-chimeric sequences; evaluates segment similarity to identify mosaic patterns.	Accuracy depends on database completeness; performs poorly for novel taxa or missing parents; limited sensitivity for low-divergence chimeras.
De novo Detection	Identifies chimeras using only internal dataset structure; leverages abundance hierarchy and compositional similarity to infer whether low-abundance sequences can be reconstructed from abundant parents.	Assumes true sequences are more abundant; fails when amplification bias distorts abundances; struggles when parental sequences are similarly abundant or highly similar.
UCHIME	Alignment-based model that partitions the query into segments, identifies parent candidates, and computes a chimera score via a three-way alignment; supports reference and de novo modes.	Reduced accuracy for very closely related parents (<0.8% divergence); sensitive to incomplete databases; de novo mode fails if parents are absent or not sufficiently more abundant.
UCHIME2	Updated UCHIME with improved benchmarking (CHSIMA) and multiple sensitivity/specificity presets; better handles incomplete references and dataset variability.	"Fake models" limit theoretical accuracy; genuine variants may mimic chimeras; not recommended as a standalone step in OTU or denoising pipelines due to increased false positives/negatives.
CATCh	First ensemble ML model for 16S chimera detection; integrates outputs of UCHIME, ChimeraSlayer, DECIPHER, Pintail, and Perseus using an SVM to boost overall prediction accuracy.	Performance constrained by underlying tools; ML model cannot capture features not present in component algorithms; may misclassify in highly novel or low-coverage datasets.
ChimPipe	Pipeline for detecting biological chimeras in RNA-seq using discordant paired-end reads and split-read alignments; identifies isoforms and breakpoint coordinates.	Requires high-quality genome and annotation; tailored to RNA-seq rather than amplicons; computationally intensive; limited to organisms with available reference genomes.

Across existing studies, no single approach reliably detects all forms of chimeric sequences, and the reviewed literature consistently shows that chimeras remain a persistent challenge in genomics and bioinformatics. Although the surveyed tools are not designed specifically for organelle genome assembly, they provide valuable insights into which methodological strategies are effective and where current approaches fall short. These limitations collectively define a clear research gap: the need for a specialized, feature-driven detection framework tailored to PCR-induced mitochondrial chimeras. Addressing this gap aligns with the research objective outlined in Section 1.3, which is to develop and evaluate a machine learning-based pipeline (MitoChime) that improves the quality of downstream mitochondrial genome assembly. In support of this aim, the subsequent chapters describe the design, implementation, and evaluation of the proposed tool.

Chapter 3

Research Methodology

This chapter outlines the steps involved in completing the study, including data gathering, generating simulated mitochondrial Illumina reads, preprocessing and indexing the data, developing a feature extraction pipeline to obtain read-level features, applying machine learning algorithms for chimera detection, implementing feature selection methods, and validating and comparing model performance.

3.1 Research Activities

As illustrated in Figure 3.1, this study carried out a sequence of procedures to detect PCR-induced chimeric reads in mitochondrial genomes. The process began with collecting a mitochondrial reference sequence of *Sardinella lemuru* from the National Center for Biotechnology Information (NCBI) database, which was used as a reference for generating simulated clean and chimeric reads. These reads were subsequently indexed and mapped. The resulting collections then passed through a feature extraction pipeline that computed k-mer profiles, supplementary alignment (SA) features, and microhomology information to prepare the data for model construction. The machine learning models were trained using the processed input, evaluated using cross-validation and held-out testing, tuned for improved performance, and then subjected to feature importance and feature selection analyses before final validation.

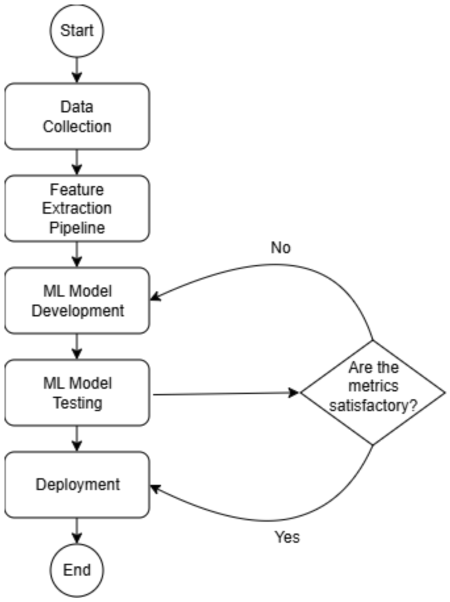


Figure 3.1: Process diagram of the study workflow.

3.1.1 Data Collection

The mitochondrial genome reference sequence of *S. lemuru* was obtained from the NCBI database (accession number NC_039553.1) in FASTA format and was used to generate simulated reads.

This step was scheduled to begin in the first week of November 2025 and expected to be completed by the end of that week, with a total duration of approximately one (1) week.

Data Preprocessing

All steps in the simulation and preprocessing pipeline were executed using a custom script in Python (Version 3.11). The script runs each stage, including read simulation, reference indexing, mapping, and alignment processing, in a fixed sequence.

wgsim (Version 1.13) was used to simulate 10,000 paired-end fragments, producing 20,000 reads (10,000 forward and 10,000 reverse) from the original reference (original_reference.fasta) and designated as clean reads. The tool was selected because it provides fast generation of Illumina-like reads with controllable error rates, using the following command:

```
wgsim -1 150 -2 150 -r 0 -R 0 -X 0 -e 0.05 -N 10000 \
        original_reference.fasta ref1.fastq ref2.fastq
```

Chimeric sequences were then generated from the same reference FASTA file using a separate Python script. Two non-adjacent segments were randomly selected such that their midpoint distances fell within specified minimum and maximum thresholds. The script attempted to retain microhomology to mimic PCR-induced template switching. The resulting chimeras were written to chimera_reference.fasta and processed with wgsim to simulate 10,000 paired-end fragments, generating 20,000 chimeric reads (10,000 forward reads in chimeric1.fastq and 10,000 reverse reads in chimeric2.fastq) using the same command format as above.

Next, a `minimap2` index of the reference genome was created using:

```
minimap2 -d ref.mmi original_reference.fasta
```

Minimap2 (Version 2.28) was used to map simulated clean and chimeric reads to the original reference. An index (`ref.mmi`) was first generated to enable efficient alignment, and mapping produced the alignment features used as input for the machine learning model. The reads were mapped using the following commands:

```
minimap2 -ax sr -t 8 ref.mmi ref1.fastq ref2.fastq > clean.sam
```

```
minimap2 -ax sr -t 8 ref.mmi \
  chimeric1.fastq chimeric2.fastq > chimeric.sam
```

The resulting clean and chimeric SAM files contain the alignment positions of each read relative to the original reference genome. These files were then converted to BAM format, sorted, and indexed using `samtools` (Version 1.20):

```
samtools view -bS clean.sam -o clean.bam
samtools view -bS chimeric.sam -o chimeric.bam

samtools sort clean.bam -o clean.sorted.bam
samtools index clean.sorted.bam

samtools sort chimeric.bam -o chimeric.sorted.bam
samtools index chimeric.sorted.bam
```

The total number of simulated reads was expected to be 40,000. The final collection of reads contained 19,984 clean reads and 20,000 chimeric reads (39,984 entries in total), providing a roughly balanced distribution between the two classes. After alignment with `minimap2`, only 19,984 clean reads remained because unmapped reads were not included in the BAM file. Some sequences failed to align due to the error rate defined during `wgsim` simulation, which produced mismatches that caused certain reads to fall below the aligner's matching threshold.

This whole process was scheduled to start in the second week of November 2025 and was expected to be completed by the last week of November 2025, with a total duration of approximately three (3) weeks.

3.1.2 Feature Extraction Pipeline

This stage directly followed the alignment phase, utilizing the resulting BAM files (specifically `chimeric.sorted.bam` and `clean.sorted.bam`). A custom Python script was created to efficiently process each primary-mapped read to extract the necessary set of features, which were then compiled into a structured feature matrix in TSV format. The pipeline's core functionality relied on the `Pysam` (Version 0.22) library for parsing BAM structures and `NumPy` (Version 1.26) for array operations and computations. To ensure correctness and adherence to best practices, bioinformatics experts at PGC Visayas were consulted to validate the pipeline design, feature extraction logic, and overall data integrity.

This stage of the study was scheduled to begin in the last week of November 2025 and conclude by the first week of December 2025, with an estimated total duration of approximately two (2) weeks.

The pipeline focused on three feature families that collectively capture biological signatures associated with PCR-induced chimeras: (1) supplementary alignment (SA) and alignment-structure metrics, (2) k-mer composition difference, and (3) microhomology around putative junctions. Additional alignment quality indicators such as mapping quality were also included.

Supplementary Alignment and Alignment-Structure Features

Split-alignment information was derived from the SA tag embedded in each primary read of the BAM file. This tag is typically associated with reads that map to multiple genomic locations, suggesting a chimeric structure. To extract this information, the script first checked whether the read carried an `SA:Z` tag. If present, the tag string was parsed using the function `parse_sa_tag`, yielding metadata for each alignment containing the reference name, mapped position, strand, mapping quality, and number of mismatches.

After parsing, the function `sa_feature_stats` was applied to establish the fundamental split indicators, `has_sa` and `sa_count`. Along with these initial counts, the function aggregated metrics related to the structure and reliability of the split alignments, including the number of alignment segments, strand consistency, minimum, maximum, and mean distance between split segments, and summary statistics of mapping quality and mismatch counts across segments.

K-mer Composition Difference

Comparing k-mer frequency profiles between the left and right halves of a read allows for the detection of abrupt compositional shifts, independent of alignment information.

The script implemented this by inferring a likely junction breakpoint using the function `infer_breakpoints`, prioritizing the boundaries defined by soft-clipping operations. If no clipping was present, the midpoint of the alignment or the read length was used as a fallback. The read sequence was then divided into left and right segments at this inferred breakpoint, and k-mer frequency profiles ($k = 6$) were generated for both halves, ignoring any k-mers containing ambiguous `N` bases. The resulting k-mer frequency vectors were normalised and compared using the functions `cosine_difference` and `js_divergence` to quantify compositional discontinuity across the inferred breakpoint.

Microhomology

The process of extracting the microhomology feature also started by using `infer_breakpoints` to identify a candidate junction. Once a breakpoint was established, the script scanned a ±40 base-pair window surrounding the breakpoint and applied the function `longest_suffix_prefix_overlap` to identify the longest exact suffix–prefix overlap between the left and right read segments. This overlap, representing consecutive bases shared at the junction, was recorded as `microhomology_length` in the dataset. The 40 base-pair window was chosen to ensure that short shared sequences at or near the breakpoint were captured without including distant sequences that are unlikely to be mechanistically related.

Additionally, the GC content of the overlapping sequence was calculated using the function `gc_content`, which counts guanine (G) and cytosine (C) bases within the detected microhomology and divides by the total length, yielding a proportion between 0 and 1 that was stored under the `microhomology_gc` attribute. Microhomology was quantified using a 3–20 bp window, consistent with values reported in prior research on PCR-induced chimeras. A k-mer length of 6 was used to capture patterns within the 40 bp window surrounding each breakpoint, providing sufficient resolution to detect informative sequence shifts.

3.1.3 Machine Learning Model Development

After feature extraction, the per-read feature matrices for clean and chimeric reads were merged into a single dataset. Each row corresponded to one paired-end read, and columns encoded alignment-structure features (e.g., supplementary alignment count and spacing between segments), CIGAR-derived soft-clipping statistics (e.g., left and right soft-clipped length, total clipped bases), k-mer composition discontinuity between read segments, microhomology descriptors near candidate junctions, and alignment quality (e.g., mapping quality). The resulting feature set comprised 23 numeric features and was restricted to quantities that can be computed from standard BAM/FASTQ files in typical mitochondrial sequencing workflows.

The labelled dataset was randomly partitioned into training (80%) and test

(20%) subsets using stratified sampling to preserve the 1:1 ratio of clean to chimeric reads. Model development and evaluation were implemented in Python (Version 3.11) using the `scikit-learn`, `xgboost`, `lightgbm`, and `catboost` libraries. A broad panel of classification algorithms was then benchmarked on the training data to obtain a fair comparison of different model families under identical feature conditions. The panel included a trivial dummy classifier, L_2-regularized logistic regression, a calibrated linear support vector machine (SVM), k-nearest neighbours, Gaussian Naïve Bayes, decision-tree ensembles (Random Forest, Extremely Randomized Trees, and Bagging with decision trees), gradient boosting methods (Gradient Boosting, XGBoost, LightGBM, and CatBoost), and a shallow multilayer perceptron (MLP).

For each model, five-fold stratified cross-validation was performed on the training set. In every fold, four-fifths of the data were used for fitting and the remaining one-fifth for validation. Mean cross-validation accuracy, precision, recall, F1-score for the chimeric class, and area under the receiver operating characteristic curve (ROC–AUC) were computed to summarize performance and rank candidate methods. This baseline screen allowed comparison of linear, probabilistic, neural, and ensemble-based approaches and identified tree-based ensemble and boosting models as consistently strong performers relative to simpler baselines.

3.1.4 Model Benchmarking, Hyperparameter Optimization, and Evaluation

Model selection and refinement proceeded in two stages. First, the cross-validation results from the broad panel were used to identify a subset of competitive models for more detailed optimization. Specifically, ten model families were carried forward: L_2-regularized logistic regression, calibrated linear SVM, Random Forest, ExtraTrees, Gradient Boosting, XGBoost, LightGBM, CatBoost, Bagging with decision trees, and a shallow MLP. This subset spans both linear and nonlinear decision boundaries, but emphasizes ensemble and boosting methods, which showed superior F1 and ROC–AUC in the initial benchmark.

Second, hyperparameter optimization was conducted for each of the ten selected models using randomized search with five-fold stratified cross-validation (`RandomizedSearchCV`). For tree-based ensembles, the search space included the number of trees, maximum depth, minimum samples per split and per leaf, and the fraction of features considered at each split. For boosting methods, key hyperparameters such as the number of boosting iterations, learning rate, tree depth, subsampling rate, and column subsampling rate were tuned. For the MLP, the number and size of hidden layers, learning rate, and L_2-regularization strength were varied. In all cases, the primary optimisation criterion was the F1-score of the chimeric class, averaged across folds.

For each model family, the hyperparameter configuration with the highest mean cross-validation F1-score was selected as the best-tuned estimator. These tuned models were then refitted on the full training set and evaluated once on the held-out test set to obtain unbiased estimates of performance. Test-set metrics included accuracy, precision, recall, F1-score for the chimeric class, and ROC–AUC. Confusion matrices and ROC curves were generated for the top-performing models to characterise common error modes, such as false negatives (missed chimeric reads) and false positives (clean reads incorrectly labelled as chimeric). The final model or small set of models for downstream interpretation was chosen based on a combination of test-set F1-score and ROC–AUC.

3.1.5 Feature Importance, Feature Selection, and Interpretation

To relate model decisions to biologically meaningful signals, feature-importance analyses were performed on the best-performing tree-based models. Two complementary approaches were used. First, built-in importance measures from ensemble methods (e.g., split-based importances in Random Forest and Gradient Boosting) were examined to obtain an initial ranking of features based on their contribution to reducing impurity. Second, model-agnostic permutation importance was computed on the test set by repeatedly permuting each feature column while keeping all others fixed and measuring the resulting decrease in F1-score. Features whose permutation led to a larger performance drop were interpreted as more influential for chimera detection.

For interpretability, individual features were grouped into conceptual families: (i) supplementary alignment and alignment-structure features (e.g., SA count, spacing between alignment segments, strand consistency), (ii) soft-clipping features (e.g., left and right soft-clipped length, total clipped bases, inferred breakpoint position), (iii) k-mer composition discontinuity features (e.g., cosine distance and Jensen–Shannon divergence between k-mer profiles of read segments), (iv) microhomology descriptors (e.g., microhomology length and local GC content around putative breakpoints), and (v) other alignment quality features (e.g., mapping quality). This analysis provided a basis for interpreting the trained models in terms of known mechanisms of PCR-induced template switching and for identifying which alignment-based and sequence-derived cues are most informative for distinguishing chimeric from clean mitochondrial reads.

Building on these importance results, an explicit feature selection step was implemented using CatBoost as the reference model, since it was among the top-performing classifiers. Permutation importance scores were re-estimated for CatBoost on the held-out test set using the F1-score of the chimeric class as the scoring function. Negative importance scores, which indicate that permuting a feature did not reliably harm performance, were set to zero and interpreted as noise. The remaining non-negative importances were sorted in descending order and converted into a cumulative importance curve by expressing each feature's importance as a fraction of the total positive importance.

A compact feature subset was then defined by selecting the smallest number of features whose cumulative importance reached at least 95% of the total positive importance. This procedure yielded a reduced set of four strongly predictive variables dominated by soft-clipping and k-mer divergence metrics (for example, total clipped bases and k-mer divergence between read halves).

To quantify the impact of this reduction, CatBoost was retrained using only the selected feature subset, with the same tuned hyperparameters as the full 23-feature model, and evaluated on the held-out test set. Performance of the reduced model was then compared to that of the full model in terms of F1-score and ROC–AUC to assess whether dimensionality could be reduced without appreciable loss in predictive accuracy.

In addition, an ablation experiment was performed to specifically evaluate the contribution of explicit microhomology features. The microhomology variables (`microhomology_length` and `microhomology_gc`) were removed from the full feature set to obtain a 21-feature configuration. CatBoost was refitted on this microhomology-ablated feature set, using the same tuned hyperparameters, and evaluated on the held-out test set. Comparing the full, reduced-subset, and microhomology-ablated variants allowed the study to quantify both the degree of redundancy among features and the practical contribution of microhomology to classification accuracy.

Taken together, the feature importance and feature selection analyses provided a more parsimonious model variant and a clearer interpretation of which alignment-based and sequence-derived signals are most informative for detecting PCR-induced chimeras.

3.1.6 Validation and Testing

Validation involved both internal and external evaluations. Internal validation was achieved through five-fold stratified cross-validation on the training data to verify model generalization and reduce variance due to random sampling. External testing was performed on the 20% hold-out dataset from the simulated reads, providing an unbiased assessment of model generalization. Feature extraction and preprocessing were applied consistently across all splits.

Comparative evaluation was performed across all candidate algorithms and CatBoost feature-set variants to determine which models demonstrated the highest predictive performance and computational efficiency under identical data conditions. Their metrics were compared to identify which algorithms and feature configurations were most suitable for further refinement and potential integration into downstream mitochondrial assembly workflows.

3.1.7 Documentation

Comprehensive documentation was maintained throughout the study to ensure transparency and reproducibility. All stages of the research, including data gathering, preprocessing, feature extraction, model training, feature selection, and validation, were systematically recorded in a `README` file in the GitHub repository. For each analytical step, the corresponding parameters, software versions, and command line scripts were documented to enable exact replication of results.

The repository structure followed standard research data management practices, with clear directories for datasets and scripts. Computational environments were standardised using Conda, with an environment file (`environment.yml`) specifying dependencies and package versions to maintain consistency across systems.

For manuscript preparation and supplementary materials, Overleaf (LaTeX) was used to produce publication-quality formatting and consistent referencing.

3.2 Calendar of Activities

Table 3.1 presents the project timeline in the form of a Gantt chart, where each bullet point corresponds to approximately one week of planned activity.

Table 3.1: Timetable of activities.

Activities (2025)	**Nov**	**Dec**	**Jan**	**Feb**	**Mar**	**Apr**	**May**
Data Collection and Simulation	● ● ●●						
Feature Extraction Pipeline	●	●					
Machine Learning Development		●	●●	● ● ●●	● ● ●●	●●	
Testing and Validation						●●	● ● ●●
Documentation	● ● ●●	● ● ●●	● ● ●●	● ● ●●	● ● ●●	● ● ●●	● ● ●●

Chapter 4

Results and Discussion

This chapter presents the performance of the proposed feature set and machine learning models for detecting PCR-induced chimeric reads in simulated mitochondrial Illumina data. The behaviour of the extracted features is first examined through descriptive and correlation analyses, followed by a comparison of baseline and tuned classifiers. The chapter then examines model performance in detail and investigates the contribution of individual features and feature families, including the impact of feature selection on classification performance.

The final dataset contained 31,986 reads for training and 7,997 reads for testing, with classes balanced (approximately 4,000 clean and 4,000 chimeric reads in the test split).

4.1 Descriptive Analysis of Features

4.1.1 Summary Statistics Per Class

Summary statistics were computed separately for clean reads (class 0) and chimeric reads (class 1) to characterize the distributional behavior of the features. For each feature, the mean, standard deviation, median, first and third quartiles (Q1, Q3), interquartile range (IQR), minimum, maximum, and sample size (n) were calculated.

Only a subset of the features is summarized in the main text to highlight key trends, and not all summary statistics columns are shown for brevity. The complete set of per-class summary statistics for all features is provided in Appendix A (Table A.1).

Alignment and Supplementary Alignment Features

Features related to supplementary alignments show strong separation between classes. Chimeric reads frequently exhibit supplementary alignments, reflected by higher values of `has_sa`, `sa_count`, and `num_segments`, whereas clean reads consistently show a single alignment segment with no supplementary mappings. Table 4.1 shows that `has_sa` is present in chimeric reads (mean = 0.406) but absent in clean reads (mean = 0.000), while `num_segments` increases from a constant value of 1.000 in clean reads to a mean of 1.406 in chimeric reads. These patterns align with the expected structure of chimeric reads and indicate that alignment-based features are highly informative.

Clipping-Based Features

Clipping-related features, including softclip_left, softclip_right, and total_clipped_bases, display higher values and broader distributions in chimeric reads. In chimeric reads, total_clipped_bases reaches 25.44 on average, with a median of 19.0 and an IQR of 48.0, while softclip_left and softclip_right have averages of 12.55 and 12.90, medians of 0.0, and IQRs of 19.0. Clean reads maintain values near zero across all these metrics. These patterns indicate substantial clipping and increased variability in chimeric reads, reflecting junction-like alignment fragmentation, whereas clean reads remain unaltered.

K-mer Distribution Features

K-mer–based features, including kmer_js_divergence and kmer_cosine_diff, show only minor differences between clean and chimeric reads. In chimeric reads, kmer_js_divergence has a mean of 0.974 with a median of 0.986, and kmer_cosine_diff has a mean of 0.974 with a median of 0.986. Clean reads show similar values, with kmer_js_divergence at 0.976 with a median of 0.986, and kmer_cosine_diff at 0.976 with a median of 0.986. The close similarity of the means, medians, and overall ranges of values indicates that these features alone provide limited ability to distinguish clean from chimeric reads.

Microhomology Features

Microhomology-related features, including microhomology_length and microhomology_gc, exhibit nearly identical summary statistics between clean and chimeric reads. Most reads in both classes have short or zero-length microhomologies. Table 4.1 shows that `microhomology_gc` has a mean of 0.172 and a median of 0.0 in both clean and chimeric reads, while `microhomology_length` averages 0.458 with a median of 0.0 in chimeric reads and 0.462 with a median of 0.0 in clean reads. These values indicate that microhomology features alone provide limited discriminatory power and are more appropriately considered as supporting evidence.

Overall, the summary statistics indicate that alignment-based and clipping-based features provide the strongest class separation, k-mer features contribute limited but complementary signal, and microhomology features exhibit minimal discriminative power on their own. These observations motivate the combined multi-feature approach used in subsequent modeling and evaluation.

Table 4.1: Summary statistics of selected key features by class.

Feature	Class	Mean	Std	Median	IQR
has_sa	chimeric	0.406	0.491	0.0	1.0
has_sa	clean	0.000	0.000	0.0	0.0
num_segments	chimeric	1.406	0.491	1.0	1.0
num_segments	clean	1.000	0.000	1.0	0.0
softclip_left	chimeric	12.55	21.90	0.0	19.0
softclip_left	clean	0.23	1.54	0.0	0.0
softclip_right	chimeric	12.90	22.12	0.0	19.0
softclip_right	clean	0.21	1.51	0.0	0.0
total_clipped_bases	chimeric	25.44	25.48	19.0	48.0
total_clipped_bases	clean	0.44	2.16	0.0	0.0
kmer_js_divergence	chimeric	0.974	0.025	0.986	0.043
kmer_js_divergence	clean	0.976	0.025	0.986	0.040
kmer_cosine_diff	chimeric	0.974	0.026	0.986	0.042
kmer_cosine_diff	clean	0.976	0.025	0.986	0.041
microhomology_length	chimeric	0.458	0.755	0.0	1.0
microhomology_length	clean	0.462	0.758	0.0	1.0
microhomology_gc	chimeric	0.172	0.361	0.0	0.0
microhomology_gc	clean	0.172	0.361	0.0	0.0

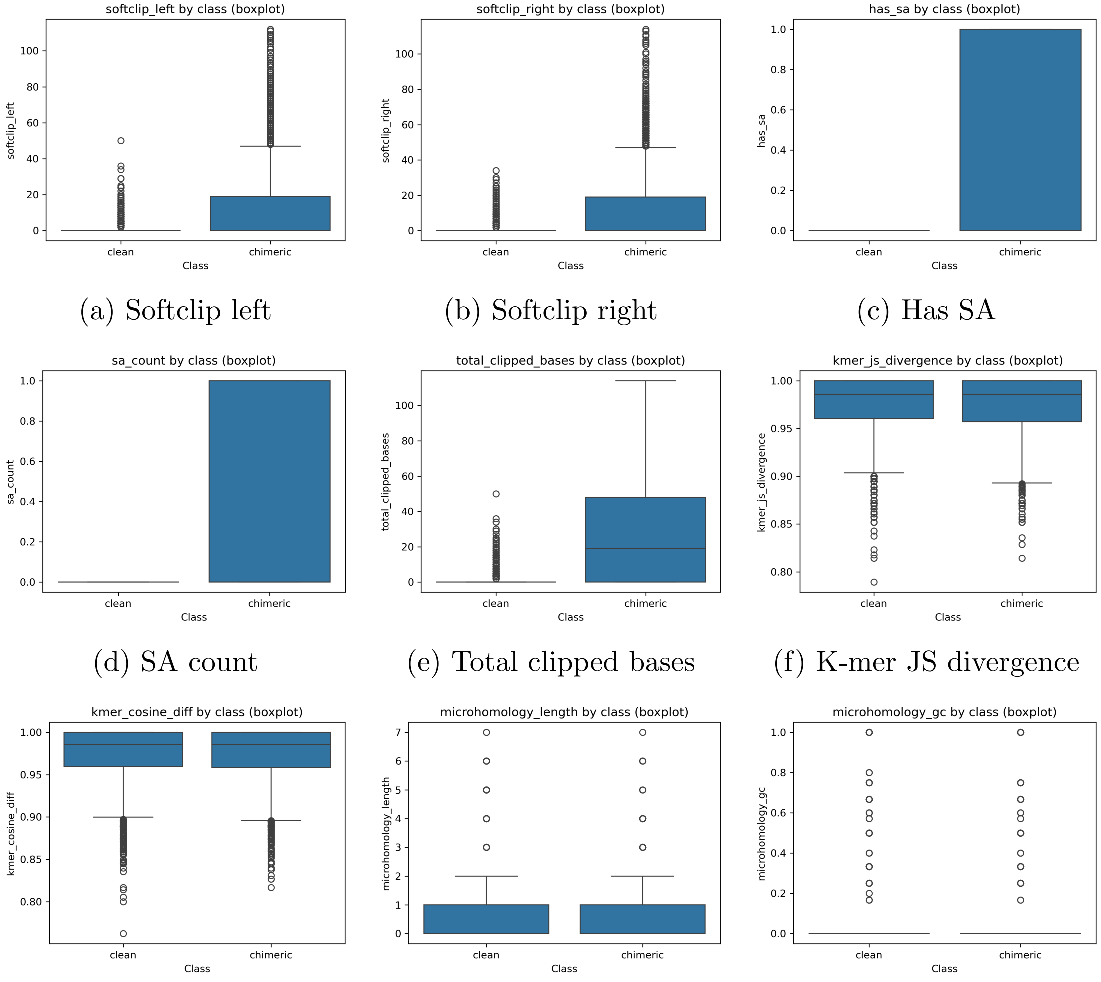
Boxplots were generated for each feature, with the x-axis representing the class (clean reads and chimeric reads) and the y-axis representing the feature value. Figure 4.1 presents a panel of selected key features, while boxplots for all numeric features are provided in Appendix B.

For clipping-related features (`softclip_left`, `softclip_right`, and `total_clipped_bases`), chimeric reads exhibit higher medians and longer upper whiskers than clean reads, indicating increased variability and the presence of split alignments.

Supplementary alignment features (`has_sa` and `sa_count`), show that clean reads are largely zero, whereas chimeric reads display a wider distribution, reflecting frequent supplementary alignments.

K-mer metrics (`kmer_js_divergence` and `kmer_cosine_diff`) show a slight upward shift for chimeric reads, but substantial overlap with clean reads indicates low discriminative power.

Microhomology features (`microhomology_length` and `microhomology_gc`) have nearly overlapping distributions for both classes, consistent with their low standalone predictive importance.



(a) Softclip left (b) Softclip right (c) Has SA

(d) SA count (e) Total clipped bases (f) K-mer JS divergence

(g) K-mer cosine difference (h) Microhomology length (i) Microhomology GC

Figure 4.1: Boxplots of selected features for clean and chimeric reads.

4.1.2 Correlation Analysis of Extracted Features

A feature correlation heatmap (Figure 4.2) was generated to examine relationships among the extracted variables and to identify patterns of redundancy and independence within the feature set. The analysis shows that alignment-related and clipping-related features form a strongly correlated cluster, including indicators of supplementary alignments, alignment segment counts, positional differences, and soft-clipping measures. These features capture related aspects of alignment fragmentation, which is a known characteristic of chimeric reads, and several show moderate correlations with the class label, supporting their relevance for distinguishing chimeric from clean reads. In contrast, general read-quality and alignment-quality metrics, such as read length, base quality, and mapping quality, exhibit weak correlations with most split-alignment features, indicating that they provide distinct information rather than overlapping with alignment-derived signals. Sequence-based features display a similar pattern of independence, as k-mer divergence metrics show weak correlations with other feature groups, while microhomology features exhibit generally low correlations with both alignment-based and k-mer-based features. Overall, the correlation structure highlights intentional redundancy within alignment-derived features and clear separation between feature families, supporting the use of features that capture different aspects of chimeric read characteristics to improve chimera classification.

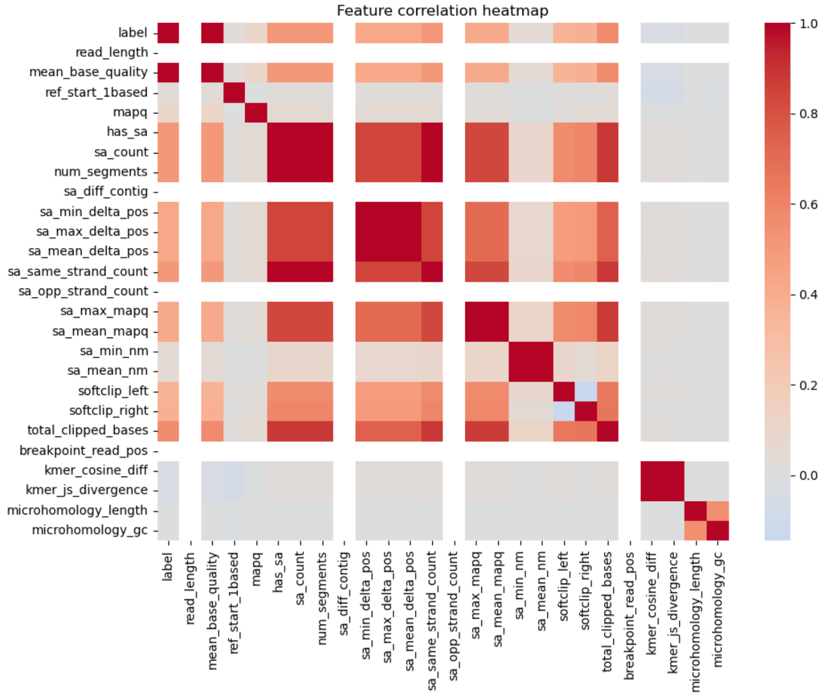


Figure 4.2: Feature correlation heatmap showing relationships among alignment-derived and sequence-derived features.

4.2 Baseline Classification Performance

Table 4.2 summarises the performance of eleven classifiers trained on the engineered feature set using five-fold cross-validation and evaluated on the held-out test set. All models were optimised using default hyperparameters, without dedicated tuning.

The dummy baseline, which always predicts the same class regardless of the input features, achieved an accuracy of 0.50 and test F1-score of 0.67. This reflects the balanced class distribution and provides a lower bound for meaningful performance.

Across other models, test F1-scores clustered in a narrow band between approximately 0.74 and 0.77 and ROC–AUC values between 0.82 and 0.84. Gradient boosting, CatBoost, LightGBM, XGBoost, bagging trees, random forest, and multilayer perceptron (MLP) all produced very similar scores, with CatBoost and gradient boosting slightly ahead (test F1 $\approx$ 0.77, ROC–AUC $\approx$ 0.84). Linear models (logistic regression and calibrated linear SVM) performed only marginally worse (test F1 $\approx$ 0.74), while Gaussian Naive Bayes lagged behind with substantially lower F1 ($\approx$ 0.65) despite very high precision for the chimeric class.

Table 4.2: Performance of baseline classifiers on the held-out test set.

model	test_accuracy	test_precision	test_recall	test_f1	test_roc_auc
dummy_baseline	0.500000	0.500000	1.000000	0.667000	0.500000
logreg_l2	0.789000	0.945000	0.614000	0.744000	0.821000
linear_svm_calibrated	0.789000	0.945000	0.614000	0.744000	0.820000
random_forest	0.788000	0.894000	0.654000	0.755000	0.834000
extra_trees	0.788000	0.901000	0.647000	0.753000	0.824000
gradient_boosting	0.802000	0.936000	0.648000	0.766000	0.840000
xgboost	0.800000	0.929000	0.650000	0.765000	0.839000
lightgbm	0.799000	0.926000	0.650000	0.764000	0.838000
catboost	0.803000	0.936000	0.650000	0.767000	0.839000
knn	0.782000	0.892000	0.642000	0.747000	0.815000
gaussian_nb	0.741000	0.996000	0.483000	0.651000	0.819000
bagging_trees	0.792000	0.900000	0.657000	0.760000	0.837000
mlp	0.789000	0.931000	0.625000	0.748000	0.819000

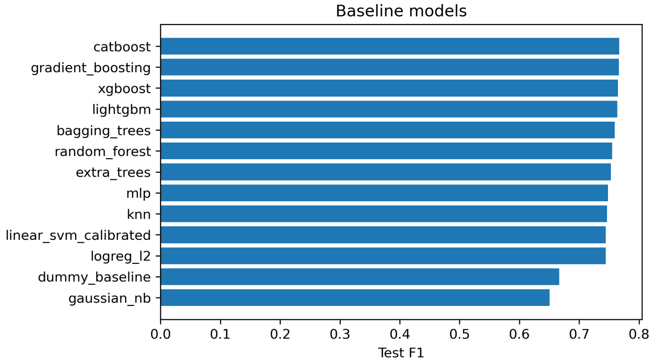


Figure 4.3: Test F1 of all baseline classifiers, showing that no single model clearly dominates and several achieve comparable performance.

4.3 Effect of Hyperparameter Tuning

To assess whether performance could be improved further, ten model families underwent randomised hyperparameter search. The tuned metrics are summarised in Table 4.3. Overall, tuning yielded modest but consistent gains for tree-based ensembles and boosting methods, while leaving linear models essentially unchanged or slightly worse.

CatBoost, gradient boosting, LightGBM, XGBoost, random forest, bagging trees, and MLP all experienced small increases in test F1 (typically ΔF1 ≈ 0.002–0.009) and ROC–AUC (up to ΔAUC ≈ 0.008). After tuning, CatBoost remained the best performer with test accuracy 0.80, precision 0.92, recall 0.66, F1-score 0.77, and ROC–AUC 0.84. Gradient boosting achieved almost identical performance (F1 0.77, AUC 0.84). Random forest and bagging trees also improved to F1 scores around 0.76 with AUC ≈ 0.84.

Table 4.3: Performance of tuned classifiers on the held-out test set.

model	test_accuracy	test_precision	test_recall	test_f1	test_roc_auc
logreg_l2_tuned	0.788000	0.946000	0.612000	0.743000	0.818000
linear_svm_calibrated_tuned	0.788000	0.944000	0.612000	0.743000	0.818000
random_forest_tuned	0.797000	0.915000	0.655000	0.763000	0.842000
extra_trees_tuned	0.794000	0.910000	0.652000	0.760000	0.837000
gradient_boosting_tuned	0.802000	0.928000	0.654000	0.767000	0.843000
xgboost_tuned	0.799000	0.922000	0.653000	0.765000	0.839000
lightgbm_tuned	0.801000	0.930000	0.651000	0.766000	0.842000
catboost_tuned	0.802000	0.924000	0.658000	0.769000	0.844000
bagging_trees_tuned	0.798000	0.922000	0.650000	0.763000	0.842000
mlp_tuned	0.790000	0.934000	0.625000	0.749000	0.821000

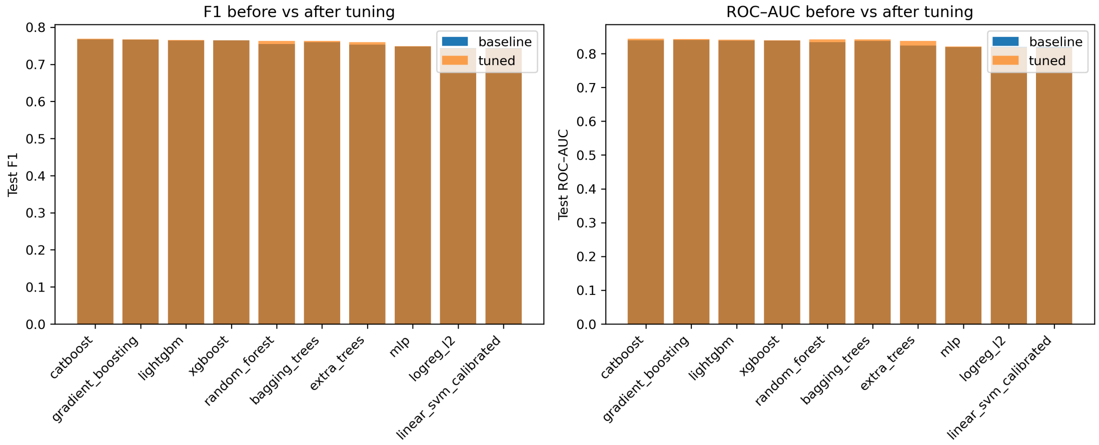


Figure 4.4: Comparison of test F1 (left) and ROC–AUC (right) for baseline and tuned models.

Because improvements are small and within cross-validation variability, tuning was interpreted as stabilising and slightly refining the models rather than completely altering their behaviour or their relative ranking.

4.4 Detailed Evaluation of Representative Models

For interpretability and diversity, four tuned models were selected for deeper analysis: CatBoost (best-performing boosted tree), scikit-learn gradient boosting (canonical gradient-boosting implementation), random forest (non-boosted ensemble baseline), and L_2-regularised logistic regression (linear baseline). All models were trained on the engineered feature set and evaluated on the same held-out test data.

4.4.1 Confusion Matrices and Error Patterns

Classification reports and confusion matrices for the four models reveal consistent patterns. CatBoost and gradient boosting both reached overall accuracy of approximately 0.80 with similar macro-averaged F1 scores (~ 0.80). For CatBoost, precision and recall for clean reads were 0.73 and 0.95, respectively, while for chimeric reads they were 0.92 and 0.66 (F1 = 0.77). Gradient boosting showed nearly identical trade-offs.

Random forest attained slightly lower accuracy (0.80) and chimeric F1 (0.76), whereas logistic regression achieved the lowest accuracy among the four (0.79) and chimeric F1 (0.74), although it provided the highest chimeric precision (0.95) at the cost of lower recall (0.61).

Across all models, errors were asymmetric. False negatives (chimeric reads predicted as clean) were more frequent than false positives. For example, CatBoost misclassified 1,369 chimeric reads as clean but only 215 clean reads as chimeric. This pattern indicates that the models are conservative and prioritise avoiding false chimera calls at the expense of missing some true chimeras. Consultation with PGC Visayas indicated that this conservative behavior is generally acceptable, though further evaluation and testing will be required to assess its impact on downstream analyses.

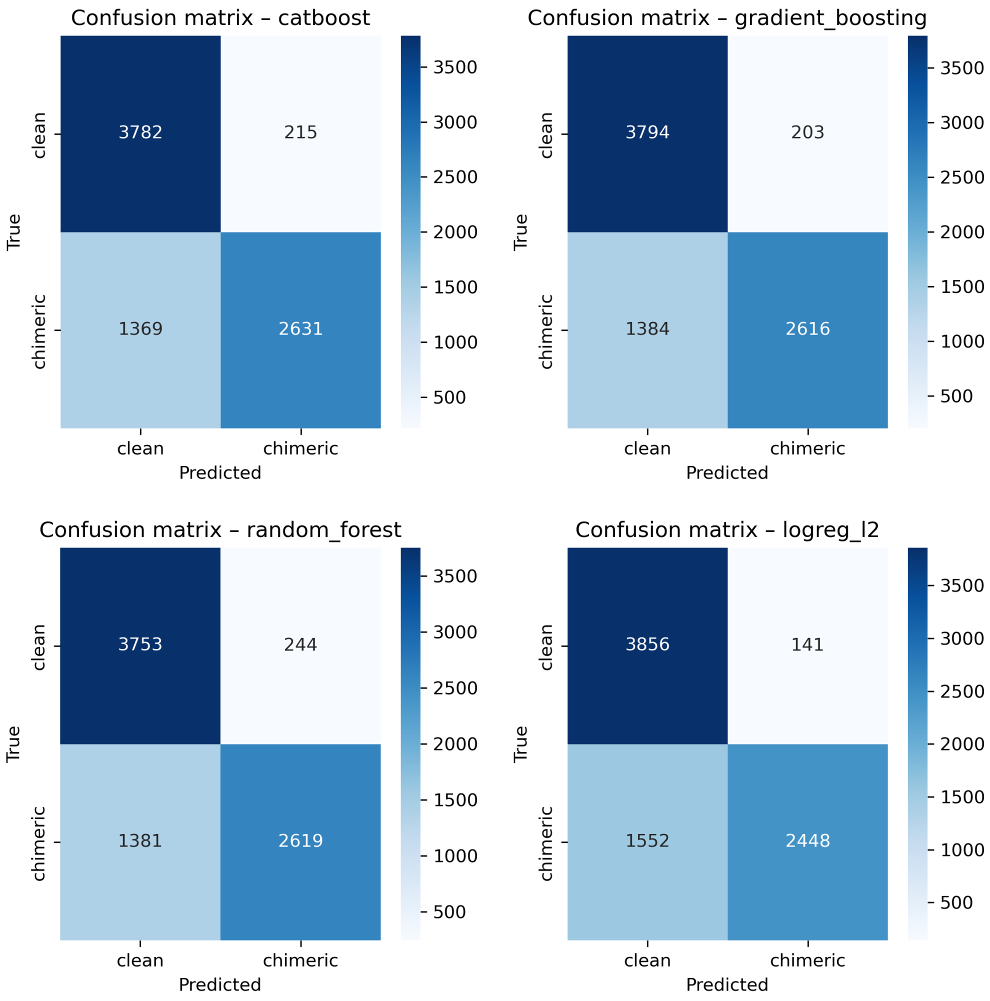


Figure 4.5: Confusion matrices for the four representative models on the held-out test set.

4.4.2 ROC and Precision–Recall Curves

Receiver operating characteristic (ROC) and precision–recall (PR) curves as shown in Figure 4.6 further support the similarity among the top models. The three tree-based ensembles (CatBoost, gradient boosting, random forest) achieved ROC–AUC values of approximately 0.84 and average precision (AP) around 0.88.

Logistic regression performed slightly worse (AUC ≈ 0.82, AP ≈ 0.87) but still substantially better than the dummy baseline.

The PR curves show that precision remains above 0.9 across a broad range of recall values (up to roughly 0.5–0.6), after which precision gradually declines. This behaviour indicates that the models can assign very high confidence to a subset of chimeric reads, while more ambiguous reads can only be recovered by accepting lower precision.

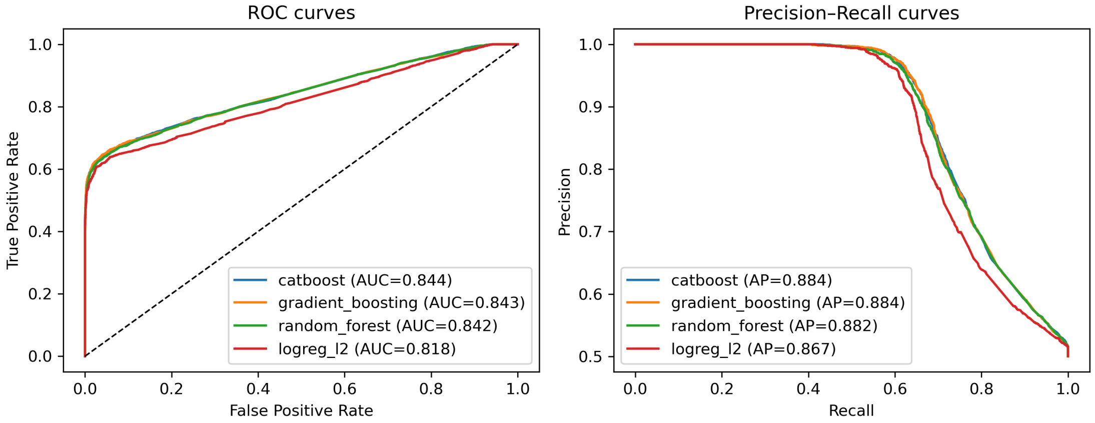


Figure 4.6: ROC (left) and precision–recall (right) curves for the four representative models on the held-out test set.

4.5 Feature Importance

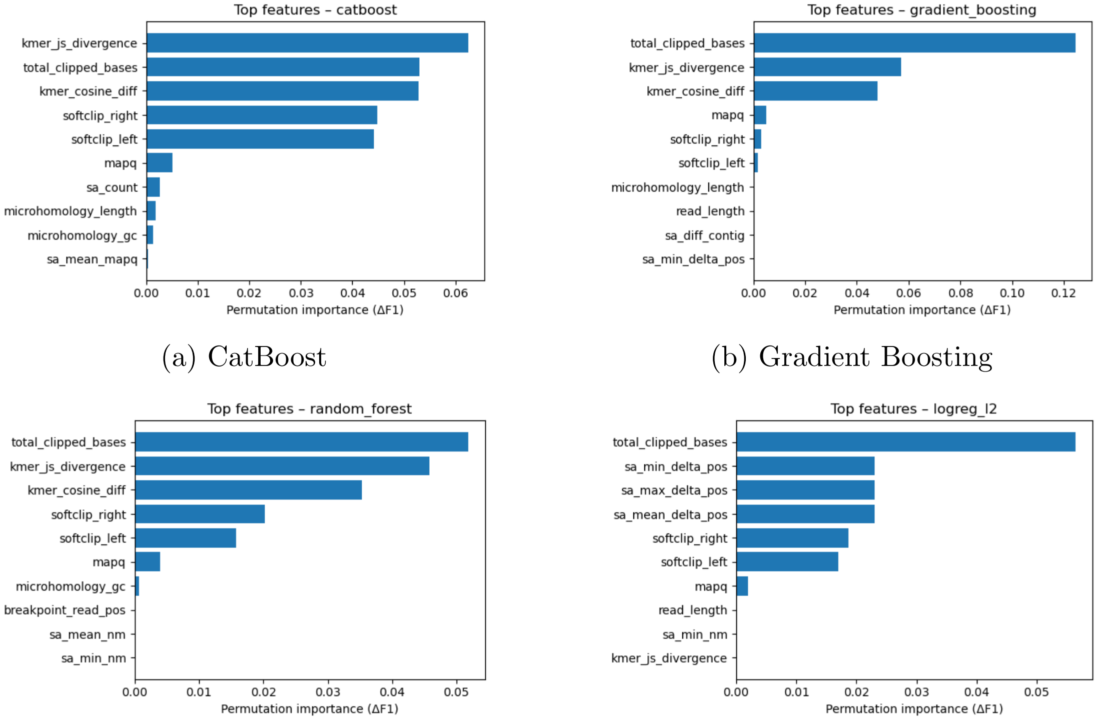
4.5.1 Permutation Importance of Individual Features

To understand how each classifier made predictions, feature importance was quantified using permutation importance. This analysis was applied to four representative models: CatBoost, Gradient Boosting, Random Forest, and L_2-regularized

Logistic Regression.

As shown in Figure 4.7, the total number of clipped bases consistently provides a strong predictive signal, particularly in Random Forest, Gradient Boosting, and L_2-regularized Logistic Regression. CatBoost differs by assigning the highest importance to k-mer divergence metrics such as `kmer_js_divergence`, which capture subtle sequence changes resulting from structural variants or PCR-induced chimeras. Soft-clipping features (`softclip_left` and `softclip_right`) provide more information around breakpoints, complementing these primary signals in all models except Gradient Boosting. L_2-regularized Logistic Regression relies more on alignment-based split-read metrics.

Overall, these results indicate that accurate detection of chimeric reads relies on both alignment-based signals and k-mer compositional information. Explicit microhomology features contribute minimally in this analysis, and combining both alignment-based and sequence-level features enhances model sensitivity and specificity.



(a) CatBoost

(b) Gradient Boosting

(c) Random Forest

(d) L_2-regularized Logistic Regression

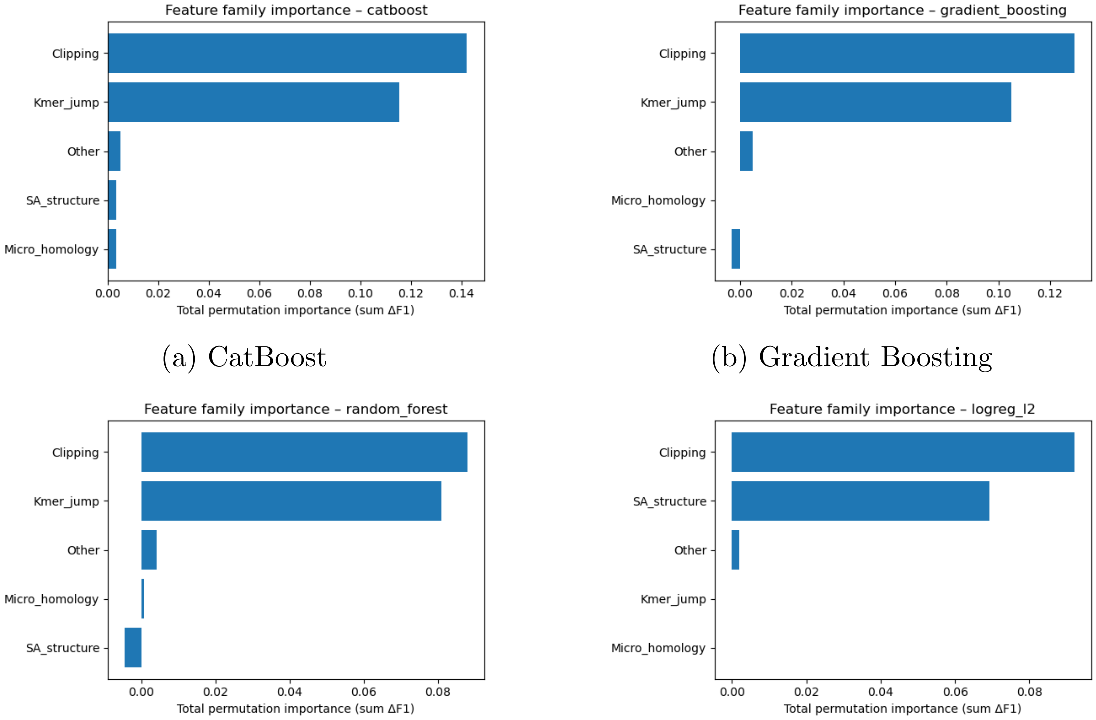
Figure 4.7: Permutation-based feature importance for four representative classifiers.

4.5.2 Feature Family Importance

To evaluate the contribution of broader signals, features were grouped into five families: SA_structure (supplementary alignment and segment metrics, e.g., `has_sa`, `sa_count`, `sa_min_delta_pos`, `sa_mean_nm`, etc.), Clipping (`softclip_left`, `softclip_right`, `total_clipped_bases`, `breakpoint_read_pos`), Kmer_jump (`kmer_cosine_diff`, `kmer_js_divergence`), Micro_homology (`microhomology_length`, `microhomology_gc`), and Other (e.g., `mapq`).

Aggregated analyses reveal consistent patterns across models. In CatBoost, the Clipping family has the largest cumulative contribution (0.14), followed by Kmer_jump (0.12), with Other features contributing minimally (0.005) and SA_structure (0.003) and Micro_homology (0.003) providing minimal predictive power. Gradient Boosting shows a similar trend, with Clipping (0.13) dominating, Kmer_jump (0.11) secondary, and the remaining families contributing negligibly. Random Forest integrates both Clipping (0.088) and Kmer_jump (0.08) effectively, while SA_structure, Micro_homology, and Other remain minor contributors. L_2-regularized Logistic Regression emphasizes Clipping (0.09) and SA_structure (0.07), with Kmer_jump and Micro_homology having minimal impact.

Both feature-level and aggregated analyses indicate that detection of chimeric reads in this dataset relies primarily on alignment irregularities (Clipping) and k-mer compositional shifts (Kmer_jump), which often arise from PCR-induced template switching events, while explicit microhomology features contribute minimally.



(a) CatBoost

(b) Gradient Boosting

(c) Random Forest

(d) L_2-regularized Logistic Regression

Figure 4.8: Aggregated feature family importance across four models.

4.6 Feature Selection

Feature selection was performed to identify the smallest subset reaching 95% cumulative importance. Three models were evaluated as references: the full model with all 23 features, a reduced model with the top-k features, and an ablation model excluding microhomology features, using a tuned CatBoost classifier to assess feature contributions and overall classification performance.

4.6.1 Cumulative Importance Curve

The cumulative importance curve was computed using the tuned CatBoost classifier. Figure 4.9 illustrates the contribution of features sorted by importance. The curve rises steeply for the first few features and then gradually plateaus, indicating that a small number of features capture most of the model's predictive power. A cumulative importance of 95% is reached at $k = 4$ features, which are `total_clipped_bases`, `kmer_js_divergence`, `kmer_cosine_diff`, and `softclip_left`.

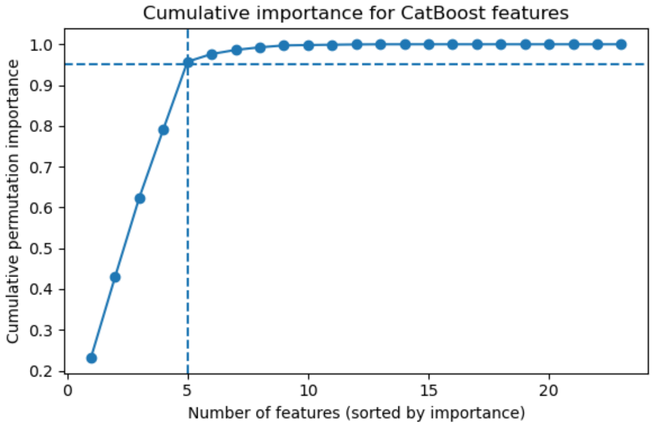


Figure 4.9: Cumulative importance curve of features sorted by importance.

4.6.2 Performance Comparison Across Feature Sets

Classification performance was compared across three feature sets using a tuned CatBoost classifier. The full model, incorporating all 23 engineered features, achieved an F1 score of 0.769 and a ROC–AUC of 0.844. A reduced model using only the top four features (`total_clipped_bases`, `kmer_js_divergence`, `kmer_cosine_diff`, and `softclip_left`) achieved nearly equivalent performance with an F1 of 0.767 and a ROC–AUC of 0.835. An ablation model excluding microhomology features (`microhomology_length` and `microhomology_gc`) also performed comparably, with an F1 of 0.768 and ROC–AUC of 0.845. These results indicate that clipping and k-mer features capture almost all predictive signal, while microhomology features are largely redundant in this dataset.

Table 4.4: Test set performance of three feature set variants using tuned CatBoost.

Variant	No. of Features	Test F1	ROC–AUC
Full CatBoost	23	0.769	0.844
Selected (top-4)	4	0.767	0.835
No microhomology	21	0.768	0.845

Figure 4.10 presents a bar chart comparing F1 and ROC–AUC across the three variants, with the x-axis showing the model variants and two bars per group representing the F1 and ROC–AUC values.

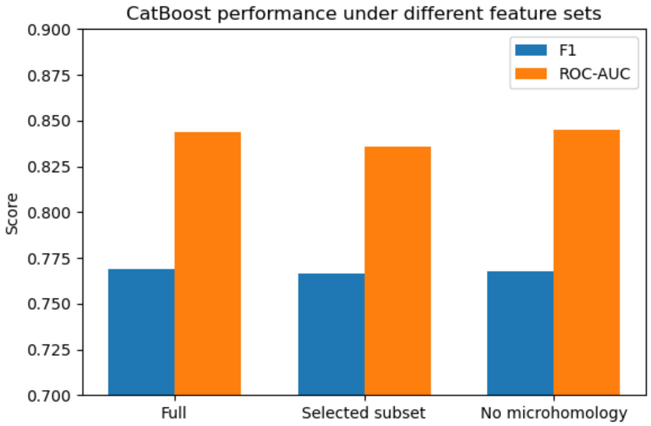


Figure 4.10: Comparison of F1 and ROC–AUC for the full, top-4 selected, and no-microhomology feature set variants.

4.6.3 Interpretation and Final Feature Set Choice

The full 23-feature model is retained as the primary configuration for the remainder of the study, while the four-feature subset serves as a lightweight alternative. Clipping features reflect alignment junctions and mapping disruptions typical of chimeric reads, and k-mer divergence captures changes in sequence composition across breakpoints. Microhomology features appear largely redundant, as their signal is either indirectly represented by clipping and k-mer features or not strongly expressed in the simulation dataset.

4.7 Summary of Findings

All evaluated machine learning models substantially outperformed the dummy baseline, demonstrating that the engineered feature set contains meaningful signals for detecting PCR-induced chimeric reads. Across classifiers, the best-performing models achieved test F1-scores of approximately 0.77 and ROC–AUC values around 0.84 on held-out simulated mitochondrial reads, indicating reliable discrimination between clean and chimeric sequences. Among the tested approaches, tree-based ensemble and boosting methods consistently showed the strongest and most stable performance. In particular, CatBoost and Gradient Boosting ranked among the top models across multiple evaluation metrics, both before and after hyperparameter tuning. These results suggest that non-linear ensemble methods are well suited to capturing the interaction between alignment-derived and sequence-derived features in this setting.

Analysis of feature behaviour revealed clear differences in how effectively feature groups distinguished clean and chimeric reads. Alignment- and clipping-based features, such as soft-clipping measures and total clipped bases, showed strong separation between clean and chimeric reads and emerged as the most informative signals. K-mer divergence features provided additional but weaker separation, contributing complementary information beyond alignment irregularities. In contrast, microhomology features and several supplementary alignment (SA) structure metrics exhibited minimal class separation and contributed little to overall predictive performance.

Feature selection results further supported these observations. A reduced subset of four features, dominated by clipping-based and k-mer divergence metrics, achieved nearly identical performance to the full 23-feature model. Moreover, removing explicit microhomology features did not degrade performance and in some cases resulted in slightly improved metrics, suggesting that these features are largely redundant under the simulated conditions tested.

Overall, these findings suggest that alignment-based and k-mer–based features provide sufficient signal to detect PCR-induced chimeric reads in simulated mitochondrial data, supporting the use of a compact and interpretable machine learning approach as a pre-assembly chimera detection step.

Appendix A

Complete Per-Class Summary Statistics

Table A.1: Complete per-class summary statistics for all extracted features.

Feature	Class	Mean	Std	Median	Q1	Q3	IQR	Min	Max	n
breakpoint_read_pos	chimeric	75.000	0.000	75.000	75.000	75.000	0.000	75.000	75.000	20000
breakpoint_read_pos	clean	75.000	0.000	75.000	75.000	75.000	0.000	75.000	75.000	19983
has_sa	chimeric	0.406	0.491	0.000	0.000	1.000	1.000	0.000	1.000	20000
has_sa	clean	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	19983
kmer_cosine_diff	chimeric	0.974	0.026	0.986	0.958	1.000	0.042	0.817	1.000	20000
kmer_cosine_diff	clean	0.976	0.025	0.986	0.959	1.000	0.041	0.814	1.000	19983
kmer_js_divergence	chimeric	0.974	0.025	0.986	0.957	1.000	0.043	0.811	1.000	20000
kmer_js_divergence	clean	0.976	0.025	0.986	0.959	1.000	0.040	0.817	1.000	19983
mapq	chimeric	59.987	0.355	60.000	60.000	60.000	0.000	43.000	60.000	20000
mapq	clean	59.663	2.036	60.000	60.000	60.000	0.000	0.000	60.000	19983
mean_base_quality	chimeric	40.000	0.000	40.000	40.000	40.000	0.000	40.000	40.000	20000
mean_base_quality	clean	13.000	0.000	13.000	13.000	13.000	0.000	13.000	13.000	19983
microhomology_gc	chimeric	0.172	0.361	0.000	0.000	0.000	0.000	0.000	1.000	20000
microhomology_gc	clean	0.172	0.361	0.000	0.000	0.000	0.000	0.000	1.000	19983
microhomology_length	chimeric	0.458	0.755	0.000	0.000	1.000	1.000	0.000	5.000	20000
microhomology_length	clean	0.462	0.758	0.000	0.000	1.000	1.000	0.000	5.000	19983

Continued on next page

Feature	Class	Mean	Std	Median	Q1	Q3	IQR	Min	Max	n
num_segments	chimeric	1.406	0.491	1.000	1.000	2.000	1.000	1.000	2.000	20000
num_segments	clean	1.000	0.000	1.000	1.000	1.000	0.000	1.000	1.000	19983
read_length	chimeric	150.000	0.000	150.000	150.000	150.000	0.000	150.000	150.000	20000
read_length	clean	150.000	0.000	150.000	150.000	150.000	0.000	150.000	150.000	19983
ref_start_1based	chimeric	8428.635	4248.348	8433.000	5013.000	11786.250	6773.250	1.000	16521.000	20000
ref_start_1based	clean	8200.121	4626.918	8240.000	3639.000	11565.000	7926.000	1.000	16521.000	19983
sa_count	chimeric	0.406	0.491	0.000	0.000	1.000	1.000	0.000	1.000	20000
sa_count	clean	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	19983
sa_diff_contig	chimeric	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	20000
sa_diff_contig	clean	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	19983
sa_max_delta_pos	chimeric	1573.531	2364.996	0.000	0.000	2826.250	2826.250	0.000	16519.000	20000
sa_max_delta_pos	clean	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	19983
sa_max_mapq	chimeric	14.104	21.424	0.000	0.000	27.000	27.000	0.000	60.000	20000
sa_max_mapq	clean	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	19983
sa_mean_delta_pos	chimeric	1573.531	2364.996	0.000	0.000	2826.250	2826.250	0.000	16519.000	20000
sa_mean_delta_pos	clean	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	19983
sa_mean_mapq	chimeric	14.104	21.424	0.000	0.000	27.000	27.000	0.000	60.000	20000
sa_mean_mapq	clean	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	19983

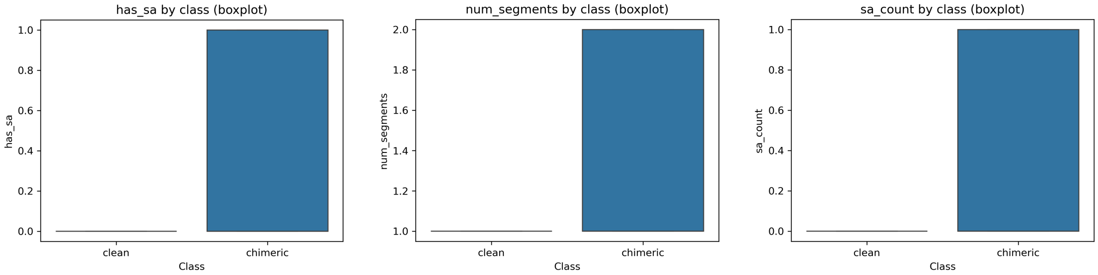
Continued on next page

Feature	Class	Mean	Std	Median	Q1	Q3	IQR	Min	Max	n
sa_mean_nm	chimeric	0.022	0.319	0.000	0.000	0.000	0.000	0.000	6.000	20000
sa_mean_nm	clean	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	19983
sa_min_delta_pos	chimeric	1573.531	2364.996	0.000	0.000	2826.250	2826.250	0.000	16519.000	20000
sa_min_delta_pos	clean	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	19983
sa_min_nm	chimeric	0.022	0.319	0.000	0.000	0.000	0.000	0.000	6.000	20000
sa_min_nm	clean	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	19983
sa_opp_strand_count	chimeric	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	20000
sa_opp_strand_count	clean	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	19983
sa_same_strand_count	chimeric	0.406	0.491	0.000	0.000	1.000	1.000	0.000	1.000	20000
sa_same_strand_count	clean	0.000	0.000	0.000	0.000	0.000	0.000	0.000	0.000	19983
softclip_left	chimeric	12.546	21.898	0.000	0.000	19.000	19.000	0.000	150.000	20000
softclip_left	clean	0.225	1.543	0.000	0.000	0.000	0.000	0.000	56.000	19983
softclip_right	chimeric	12.896	22.123	0.000	0.000	19.000	19.000	0.000	150.000	20000
softclip_right	clean	0.212	1.513	0.000	0.000	0.000	0.000	0.000	55.000	19983
total_clipped_bases	chimeric	25.442	25.481	19.000	0.000	48.000	48.000	0.000	150.000	20000
total_clipped_bases	clean	0.437	2.157	0.000	0.000	0.000	0.000	0.000	110.000	19983

Appendix B

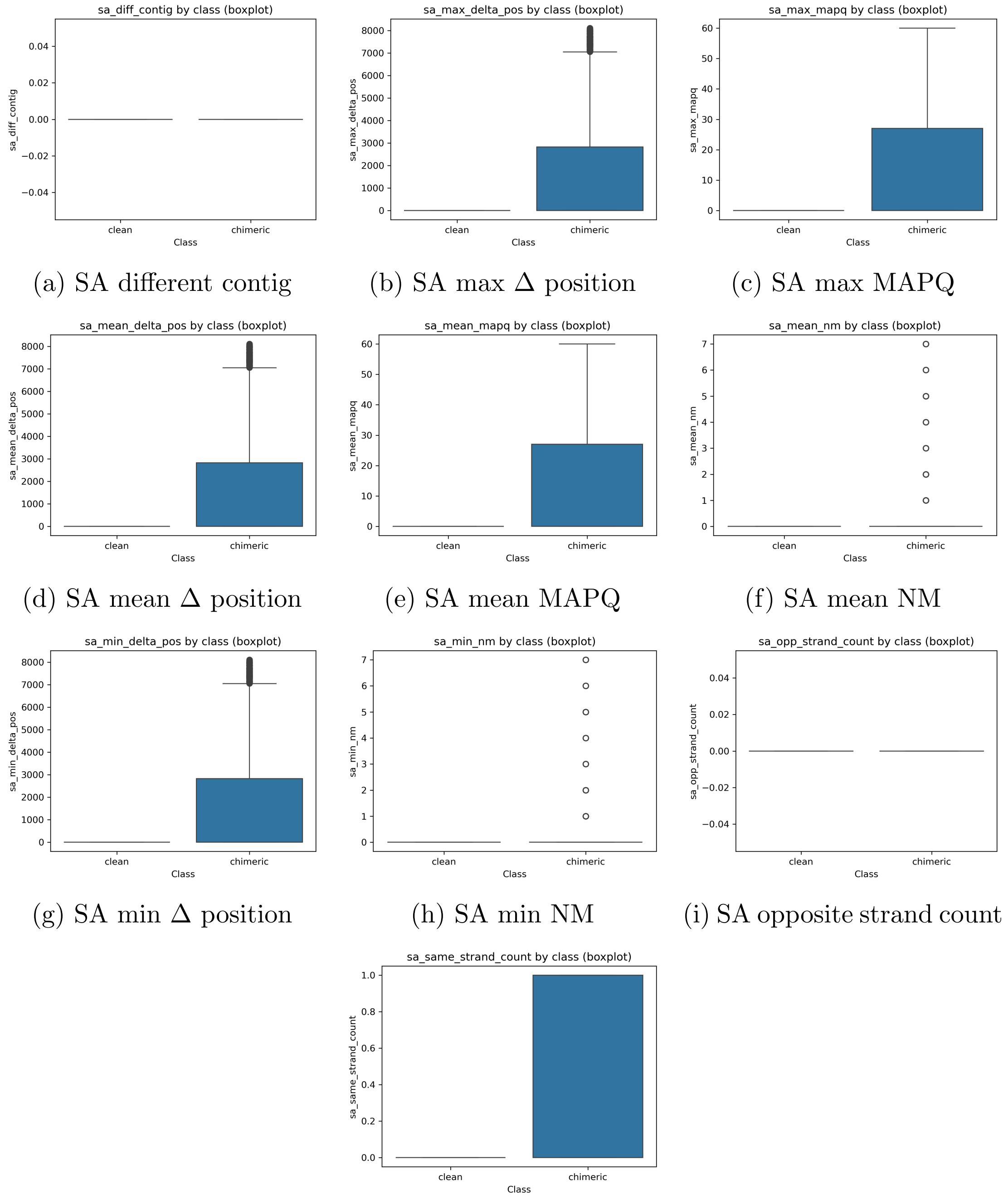
Boxplots for All Numeric Features by Feature Family

B.0.1 SA Structure (Supplementary Alignment and Segment Metrics)



(a) Has SA (b) Number of segments (c) SA count

Figure B.1: Boxplots of SA Structure features by class (1/2).



(a) SA different contig

(b) SA max Δ position

(c) SA max MAPQ

(d) SA mean Δ position

(e) SA mean MAPQ

(f) SA mean NM

(g) SA min Δ position

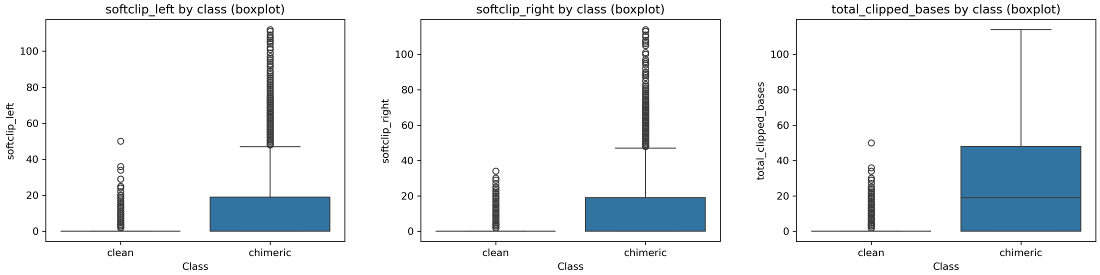
(h) SA min NM

(i) SA opposite strand count

(j) SA same strand count

Figure B.2: Boxplots of SA Structure features by class (2/2).

B.0.2 Clipping-Based Features



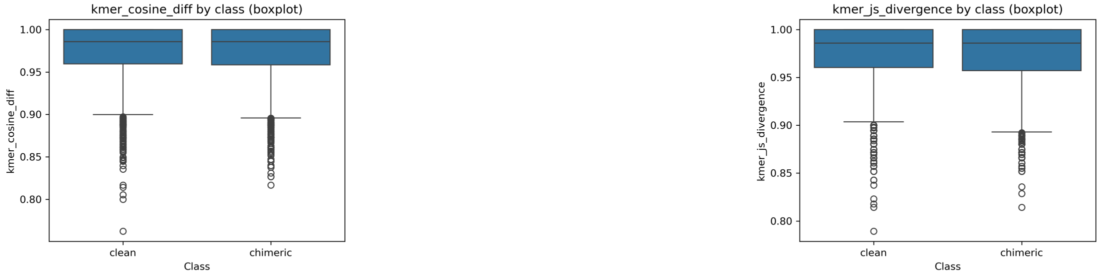
(a) Softclip left

(b) Softclip right

(c) Total clipped bases

Figure B.3: Boxplots of clipping-based features by class.

B.0.3 K-mer Features

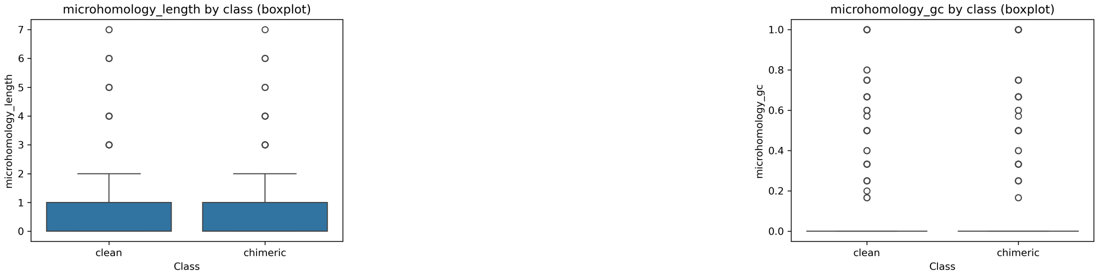


(a) K-mer cosine difference

(b) K-mer JS divergence

Figure B.4: Boxplots of k-mer features by class.

B.0.4 Microhomology Features

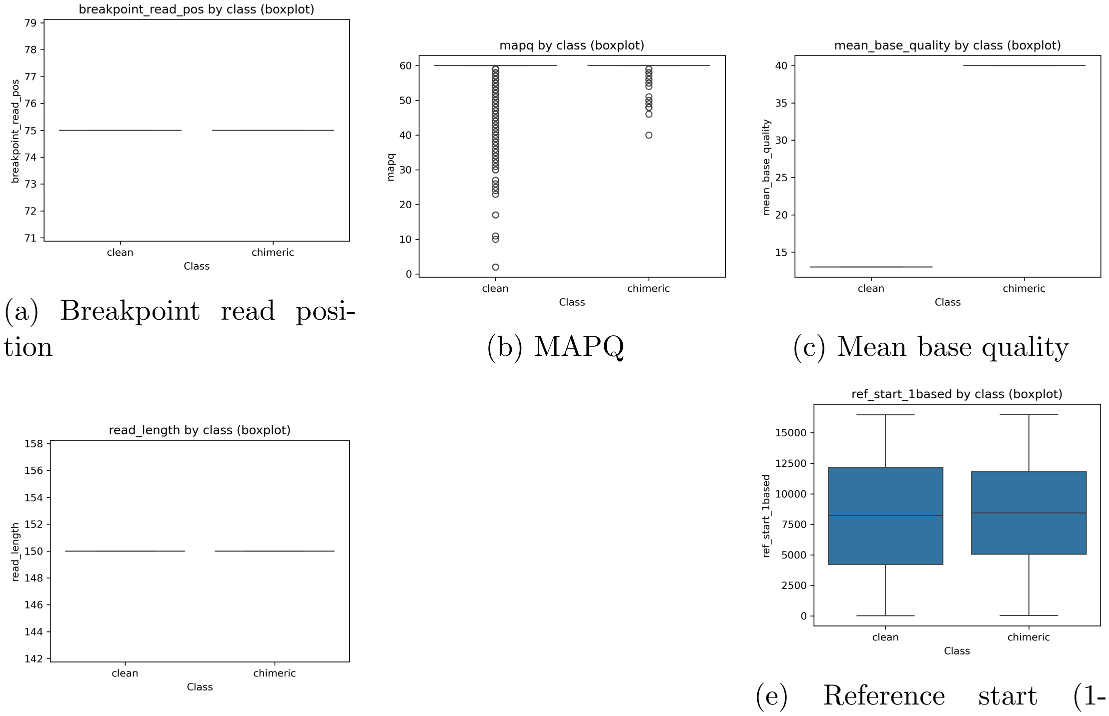


(a) Microhomology length

(b) Microhomology GC

Figure B.5: Boxplots of microhomology features by class.

B.0.5 Others



(a) Breakpoint read position

(b) MAPQ

(c) Mean base quality

(d) Read length

(e) Reference start (1-based)

Figure B.6: Boxplots of other numeric features by class.

References

Anderson, S., Bankier, A., Barrell, B., Bruijn, M., Coulson, A., Drouin, J., … Young, I. (1981, 04). Sequence and organization of the human mitochondrial genome. *Nature*, *290*, 457-465. doi: 10.1038/290457a0

Arango, G., Garner, E., Pruden, A., Heath, L., Vikesland, P., & Zhang, L. (2018, 02). Deeparg: A deep learning approach for predicting antibiotic resistance genes from metagenomic data. *Microbiome*, *6*. doi: 10.1186/s40168-018-0401-z

Bentley, D. R., Balasubramanian, S., Swerdlow, H. P., Smith, G. P., Milton, J., Brown, C. G., … Smith, A. J. (2008). Accurate whole human genome sequencing using reversible terminator chemistry. *Nature*, *456*(7218), 53–59. doi: 10.1038/nature07517

Boore, J. L. (1999). Animal mitochondrial genomes. *Nucleic Acids Research*, *27*(8), 1767–1780. doi: 10.1093/nar/27.8.1767

Cameron, S. L. (2014). Insect mitochondrial genomics: Implications for evolution and phylogeny. *Annual Review of Entomology*, *59*, 95–117. doi: 10.1146/annurev-ento-011613-162007

Dierckxsens, N., Mardulyn, P., & Smits, G. (2017). Novoplasty: de novo assembly of organelle genomes from whole genome data. *Nucleic Acids Research*, *45*(4), e18. doi: 10.1093/nar/gkw955

Edgar, R. C. (n.d.). *Uchime in practice.* Retrieved from https://www.drive5.com/usearch/manual7/uchime_practical.html

Edgar, R. C. (2016). Uchime2: improved chimera prediction for amplicon sequencing. *bioRxiv*. Retrieved from https://api.semanticscholar.org/CorpusID:88955007

Edgar, R. C., Haas, B. J., Clemente, J. C., Quince, C., & Knight, R. (2011). Uchime improves sensitivity and speed of chimera detection. *Bioinformatics*, *27*(16), 2194–2200. doi: 10.1093/bioinformatics/btr381

Glenn, T. C. (2011). Field guide to next-generation dna sequencers. *Molecular Ecology Resources*, *11*(5), 759–769. doi: 10.1111/j.1755-0998.2011.03024.x

Gonzalez, J. M., Zimmermann, J., & Saiz-Jimenez, C. (2004, 09). Evaluating putative chimeric sequences from pcr-amplified products. *Bioinformatics*, *21*(3), 333-337. Retrieved from https://doi.org/10.1093/bioinformatics/bti008 doi: 10.1093/bioinformatics/bti008

Gray, M. W. (2012). Mitochondrial evolution. *Cold Spring Harbor perspectives in biology*, *4*. Retrieved from https://doi.org/10.1101/cshperspect.a011403 doi: 10.1101/cshperspect.a011403

Hahn, C., Bachmann, L., & Chevreux, B. (2013). Reconstructing mitochondrial genomes directly from genomic next-generation sequencing reads—a baiting and iterative mapping approach. *Nucleic Acids Research*, *41*(13), e129. doi: 10.1093/nar/gkt371

Jin, J.-J., Yu, W.-B., Yang, J., Song, Y., dePamphilis, C. W., Yi, T.-S., & Li, D.-Z. (2020). Getorganelle: a fast and versatile toolkit for accurate de novo assembly of organelle genomes. *Genome Biology*, *21*(1), 241. doi: 10.1186/s13059-020-02154-5

Judo, M. S. B., Wedel, W. R., & Wilson, B. H. (1998). Stimulation and suppression of pcr-mediated recombination. *Nucleic Acids Research*, *26*(7), 1819–1825. doi: 10.1093/nar/26.7.1819

Labrador, K., Agmata, A., Palermo, J. D., Ravago-Gotanco, R., & Pante, M. J. (2021). Mitochondrial dna reveals genetically structured haplogroups of bali sardinella (sardinella lemuru) in philippine waters. *Regional Studies in Marine Science*, *41*, 101588. doi: 10.1016/j.rsma.2020.101588

Li, H. (2018, 05). Minimap2: pairwise alignment for nucleotide sequences. *Bioinformatics*, *34*(18), 3094-3100. Retrieved from https://doi.org/10.1093/bioinformatics/bty191 doi: 10.1093/bioinformatics/bty191

Liang, Q., Bible, P. W., Liu, Y., Zou, B., & Wei, L. (2020, 02). Deepmicrobes: taxonomic classification for metagenomics with deep learning. *NAR Genomics and Bioinformatics*, *2*(1), lqaa009. Retrieved from https://doi.org/10.1093/nargab/lqaa009 doi: 10.1093/nargab/lqaa009

Metzker, M. L. (2010). Sequencing technologies — the next generation. *Nature Reviews Genetics*, *11*(1), 31–46. doi: 10.1038/nrg2626

Mysara, M., Saeys, Y., Leys, N., Raes, J., & Monsieurs, P. (2015). Catch, an ensemble classifier for chimera detection in 16s rrna sequencing studies. *Applied and Environmental Microbiology*, *81*(5), 1573-1584. Retrieved from https://journals.asm.org/doi/abs/10.1128/aem.02896-14 doi: 10.1128/AEM.02896-14

Peccoud, J., Lequime, S., Moltini-Conclois, I., Giraud, I., Lambrechts, L., & Gilbert, C. (2018, 04). A survey of virus recombination uncovers canonical features of artificial chimeras generated during deep sequencing library preparation. *G3 Genes—Genomes—Genetics*, *8*(4), 1129-1138. Retrieved from https://doi.org/10.1534/g3.117.300468 doi: 10.1534/

g3.117.300468

Qin, Y., Wu, L., Zhang, Q., Wen, C., Nostrand, J. D. V., Ning, D., … Zhou, J. (2023). Effects of error, chimera, bias, and gc content on the accuracy of amplicon sequencing. *mSystems*, *8*(6), e01025-23. Retrieved from `https://journals.asm.org/doi/abs/10.1128/msystems.01025-23` doi: 10.1128/msystems.01025-23

Qiu, X., Wu, L., Huang, H., McDonel, P. E., Palumbo, A. V., Tiedje, J. M., & Zhou, J. (2001). Evaluation of pcr-generated chimeras, mutations, and heteroduplexes with 16s rrna gene-based cloning. *Applied and Environmental Microbiology*, *67*(2), 880–887. doi: 10.1128/AEM.67.2.880-887.2001

Ren, J., Song, K., Deng, C., Ahlgren, N., Fuhrman, J., Li, Y., … Sun, F. (2020, 01). Identifying viruses from metagenomic data using deep learning. *Quantitative Biology*, *8*. doi: 10.1007/s40484-019-0187-4

Rodriguez-Martin, B., Palumbo, E., Marco-Sola, S., Griebel, T., Ribeca, P., Alonso, G., … Djebali, S. (2017, 01). Chimpipe: Accurate detection of fusion genes and transcription-induced chimeras from rna-seq data. *BMC Genomics*, *18*. doi: 10.1186/s12864-016-3404-9

Rognes, T., Flouri, T., Nichols, B., Quince, C., & Mahé, F. (2016). Vsearch: a versatile open source tool for metagenomics. *PeerJ*, *4*, e2584. doi: 10.7717/peerj.2584

Sedlazeck, F., Rescheneder, P., Smolka, M., Fang, H., Nattestad, M., von Haeseler, A., & Schatz, M. (2018, 06). Accurate detection of complex structural variations using single-molecule sequencing. *Nature Methods*, *15*. doi: 10.1038/s41592-018-0001-7

Sfeir, A., & Symington, L. S. (2015). Microhomology-mediated end joining: A back-up survival mechanism or dedicated pathway? *Trends in Biochemical*

Sciences, *40*(11), 701-714. Retrieved from https://www.sciencedirect.com/science/article/pii/S0968000415001589 doi: https://doi.org/10.1016/j.tibs.2015.08.006

Vervier, K., Mahé, P., Tournoud, M., Veyrieras, J.-B., & Vert, J.-P. (2015, 11). Large-scale machine learning for metagenomics sequence classification. *Bioinformatics*, *32*(7), 1023-1032. Retrieved from https://doi.org/10.1093/bioinformatics/btv683 doi: 10.1093/bioinformatics/btv683

Willette, D., Bognot, E., Mutia, M. T., & Santos, M. (2011). *Biology and ecology of sardines in the philippines: A review* (Vol. 13; Tech. Rep. No. 1). NFRDI Technical Paper Series. Retrieved from https://nfrdi.da.gov.ph/tpjf/etc/Willette%20et%20al.%20Sardines%20Review.pdf