

# Machine Learning Pipeline for Detecting PCR-Induced Chimeric Reads

MitoChime: Organellar Chimera Detection from Per-Read Features

Duran, Lin, Pailden

University of the Philippines Visayas  
Philippine Genome Center Visayas

December 8, 2025

# Outline

- 1 Objectives
- 2 Scope and Limitations
- 3 Methodology

# General Objective

- Develop and evaluate a machine-learning pipeline (MitoChime) to detect PCR-induced chimeric reads in *S. lemur* mitochondrial sequencing data to improve downstream assembly quality.

# Specific Objectives

- 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.

# Scope of the Study

- Focuses on PCR-induced chimeric reads in *Sardinella lemuru* mitochondrial sequencing data to:
  - 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
  - to align the analysis with relevant *S. lemuru* sequencing projects at PGC Visayas
  - 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
  - to develop a tool that directly supports local studies on *S. lemuru* population structure and fisheries management produce tools applicable to local population and fisheries studies

# Scope of the Study

- Uses wgsim-based simulations and selected empirical mitochondrial datasets
- Analysis targets low-dimensional alignment and sequence features (k-mers, GC content, clipping, split alignments) to maintain interpretability and computational accessibility
- Long-read platforms (Nanopore, PacBio) and other taxa are not included

# Key Exclusions

- Naturally occurring chimeras
- NUMTs
- Large-scale nuclear genome rearrangements
- High-dimensional deep learning embeddings

# Other Limitations

- No simulations with variable sequencing error rates
- No testing of alternative parameter settings (k-mer length, microhomology windows)
- Reliance on supervised machine learning may limit detection of novel/unknown chimeric patterns



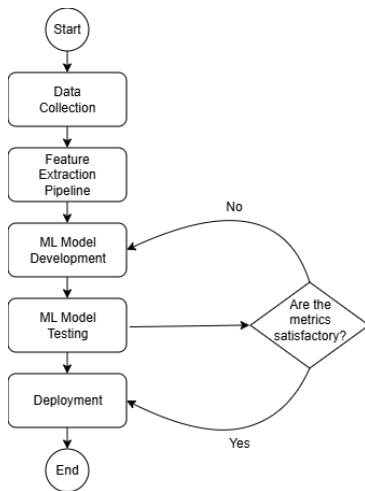


Figure: Process Diagram of the Special Project

The *S. lemur* mitochondrial reference genome (NCBI: NC\_039553.1) was downloaded in FASTA format and used as the basis for generating simulated reads.

- A Python script was used to generate the reads.
- Clean reads were produced with wgsim from the reference genome.
- A chimeric reference was created by creating a custom script to combine non-adjacent segments with microhomology
- Chimeric reads were simulated with wgsim.
- All reads were mapped with minimap2 to extract alignment information.
- SAM/BAM files were converted, sorted, and indexed with samtools.

- Final dataset: 40k reads, roughly balanced between clean and chimeric (19,984 clean reads and 20,000 chimeric).
- Some of the clean reads failed to align due to the set error rate.

# Data Preprocessing

```
NC_039553.1_3_540_8:0:0_6:0:0_ef2 163 NC_039553.1 3 60 150M = 391 538
TGGTGTAGCTTAAACAAGCATAAAGCTGAAGATGTTACGATGGGCGGTGATAAGCCACAGCAGCTGAAAGGTTTGGTCTGGCTTTATTATCAGCTTTACCGGAATTTACACACCGAGAGCTCCGCGGCGCGGTGAGGATGGCTCA
..... NM:i:8 ms:i:220
AS:i:220 nn:i:0 tp:A:P cm:i:8 s1:i:164 s2:i:0 de:f:0.0533 rl:i:0
NC_039553.1_4_430_13:0:0_11:0:0_243d 163 NC_039553.1 4 60 150M = 281 427
GGTGTAGCTTAAACAAGCATAAAGCTGAGATGATCCGCTGGGCGGTGATAAGCCGAGCAGGAGTGAAAGTTTGGTCCAGGCTTTATTATCAGCTTTACCCCAATTTACACATGCGAGCTCCGCGGCGCGGTGAGGATGGCTCAG
..... NM:i:13 ms:i:170
AS:i:170 nn:i:0 tp:A:P cm:i:9 s1:i:135 s2:i:0 de:f:0.0867 rl:i:0
NC_039553.1_5_495_6:0:0_11:0:0_1d49 163 NC_039553.1 5 60 150M = 346 491
GTGTAGCTTACACAAGCATAAAGCTGAAGATGTTAAGATGGGCGGTGATCAGCCCAAGCAGCTGAAAGGTTAGGTCCTGGCTTTATTATCAGGTTTCCCCCAATTTACACATGCGAGCTCCGCGGCGCGGTGAGGATGGCTCAGC
..... NM:i:6 ms:i:240
AS:i:240 nn:i:0 tp:A:P cm:i:12 s1:i:148 s2:i:0 de:f:0.04 rl:i:0
NC_039553.1_6_523_6:0:0_9:0:0_82c 163 NC_039553.1 6 60 150M = 374 518
TGTAGCTTAAACAAGCATAAAGCTGAAGATGTTAAGATGGGCGGTGATAAGCCCAAGCAGCTGCAAGGTTTGGTCTGGCTATATTACAGCTTTACCCCAATTTACACATGCGGCTCCGCGGCGCGGTGAGGATGGCTCAGCC
..... NM:i:6 ms:i:240
AS:i:240 nn:i:0 tp:A:P cm:i:10 s1:i:157 s2:i:0 de:f:0.04 rl:i:0
NC_039553.1_9_574_7:0:0_7:0:0_181b 163 NC_039553.1 9 60 150M = 425 566
AGCTTAAACAAGCATAAAGCTGAGATGTTAAGCTGGGCGGTGATAAGCCCAAGCAGCTGAAAGGTTTGGTCTGGCTTTATTATCAGCTTTACCGCAATTTACACATGCGAGCTCCGCGGCGCGGTGAGGCTGCCCTCCGCTCC
..... NM:i:7 ms:i:230
AS:i:230 nn:i:0 tp:A:P cm:i:12 s1:i:176 s2:i:0 de:f:0.0467 rl:i:0
NC_039553.1_10_391_9:0:0_8:0:0_256b 99 NC_039553.1 10 60 150M = 242 382
GCTTAAACAAGCATAAAGCTGAAGATGTTAAGATGGGCGGTGATAAGCCCAAGCAGGAGGTTTGGTCTGGCTTTATTATCAGCTTTACCCCAATTTAGACATGCGAGCTCCGCGGCGCGGTGATGCTGGCTCAGCTCCC
..... NM:i:9 ms:i:210
AS:i:210 nn:i:0 tp:A:P cm:i:15 s1:i:156 s2:i:0 de:f:0.06 rl:i:0
NC_039553.1_11_509_6:0:0_11:0:0_a19 99 NC_039553.1 11 60 150M = 360 499
CTTCAACAAGCATAAAGCTGAAGATGTTAAGATGGGCGGTGATAAGCCCAAGCAGCTGAAAGGTTAGGTCCTGGCTTTATTATGAGCTTTACCCCAATTTACACATGCGATCTCCGCGGCGCGGTGAGGATGCCCTCAGCTCCCG
..... NM:i:6 ms:i:242
AS:i:242 nn:i:0 tp:A:P cm:i:10 s1:i:150 s2:i:0 de:f:0.04 rl:i:0
NC_039553.1_12_427_9:0:0_9:0:0_157 163 NC_039553.1 12 60 150M = 278 416
TTAAACAAGCATAAAGCTGAAGATTTAGATGGGCGGTGATAAGCCCAAGCAGCTGAAAGTTTGGTCTGGCTTTATTATCAGCTTTACCCCAATTTACACATGCGAGCTCCGCGGCGCGGTGAGGATGCCCTCCGCTCCCGT
..... NM:i:9 ms:i:210
AS:i:210 nn:i:0 tp:A:P cm:i:8 s1:i:150 s2:i:0 de:f:0.06 rl:i:0
```

Figure: SAM File of Clean Reads

# Data Preprocessing

[illegible]

### Figure: SAM File of Chimeric Reads

# Dataset construction and split

- Simulated feature tables:
  - Clean reads (label 0)
  - PCR-induced chimeras (label 1)
- `build_datasets.py`:
  - Concatenate tables
  - Shuffle rows (avoid file-order artefacts)
- 80/20 **stratified** train-test split
- Test set held out and used **only once** at the end

# Validation strategy

- Layer 1: 80/20 stratified train–test split
- Layer 2: 5-fold stratified cross-validation on training set
  - Train on 4 folds, validate on 1
  - Rotate so each fold is validation once
- Layer 3: Final evaluation on held-out test set
- Hyperparameter tuning:
  - RandomizedSearchCV inside CV for top models
- Goal: stable estimates and **unbiased** test performance



# Model zoo and preprocessing pipeline

- **Baseline:** dummy majority-class classifier
- **Linear models:** logistic regression, calibrated linear SVM
- **Tree ensembles:**
  - Random Forest, Extra Trees
  - Gradient Boosting, XGBoost, LightGBM, CatBoost
- **Others:** bagging trees, k-NN, Gaussian NB, shallow MLP
- Common scikit-learn pipeline:
  - Median imputation (numeric missing values)
  - Standardisation (zero mean, unit variance)
- Ensures a **fair comparison** across models

# Effect of hyperparameter tuning

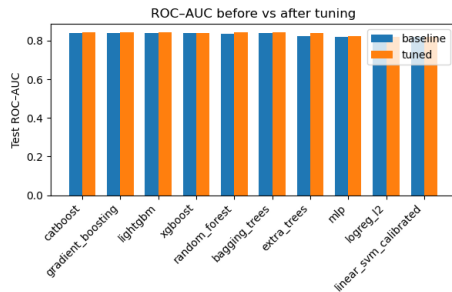
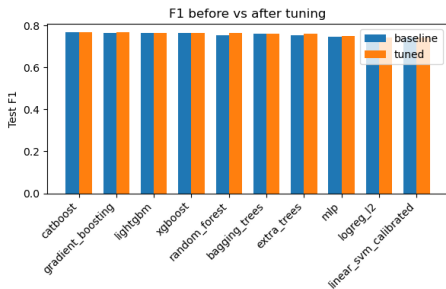
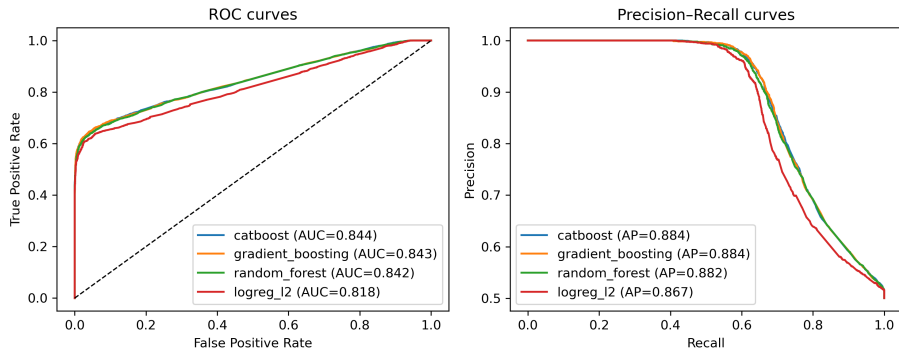


Figure: Test F1: baseline vs tuned.

Figure: Test ROC-AUC: baseline vs tuned.

- Tuning done with RandomizedSearchCV on training set
- Small but consistent gains (F1, AUC  $\approx$  0.001–0.01)
- Top-ranked models remain the same (CatBoost, Gradient Boosting, LightGBM)

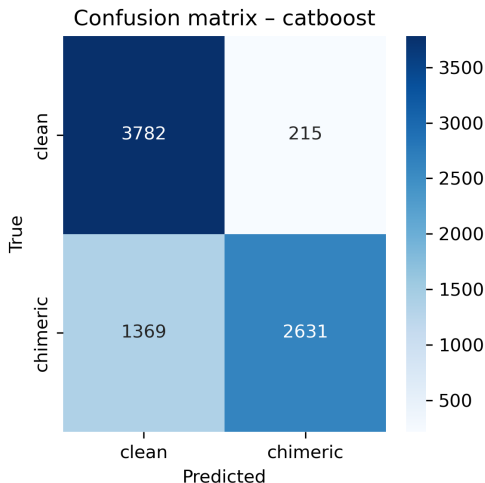
# ROC and precision–recall curves



**Figure:** ROC (left) and PR (right) curves for CatBoost, Gradient Boosting, Random Forest, and logistic regression.

- Ensembles: ROC–AUC  $\approx 0.84$ ; logreg:  $\approx 0.82$
- Average precision  $\approx 0.88$  for ensembles
- Precision  $> 0.9$  up to recall  $\approx 0.5$ – $0.6$

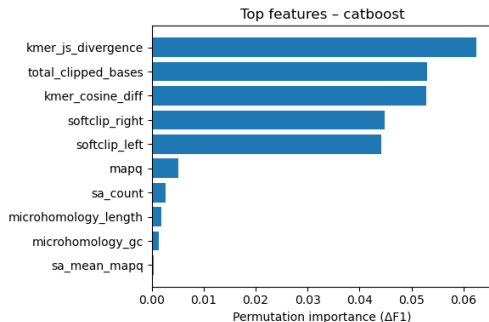
# Confusion matrix: CatBoost (test set)



- Clean reads:
  - Recall  $\approx 0.95$  (3782 / 3997)
- Chimeric reads:
  - Precision  $\approx 0.92$
  - Recall  $\approx 0.66$  (2631 / 4000)
- Behaviour at default threshold:
  - **Conservative chimera filter**
  - Protects clean reads, misses some subtle chimeras

Figure: Confusion matrix heatmap for CatBoost.

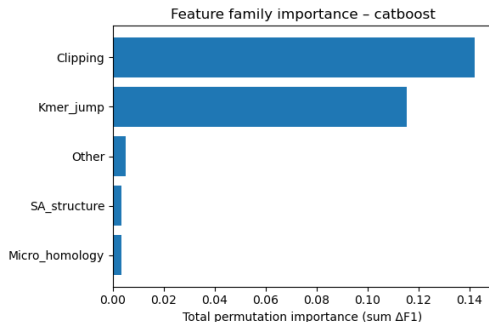
# Top features for CatBoost



**Figure:** Permutation importance (F1) for CatBoost.

- Strongest signals:
  - kmer\_js\_divergence
  - total\_clipped\_bases
  - kmer\_cosine\_diff
- Also important:
  - Left/right soft-clipping
  - Mapping quality (MAPQ)
  - SA count (supplementary alignments)
- Consistent with PCR chimera junctions

# Feature family importance



**Figure:** Aggregated feature families for CatBoost.

- Aggregated permutation importance:
  - **Clipping** features dominate
  - **K-mer jump** features also strong
- Smaller contributions:
  - SA structure
  - Micro-homology
  - Other alignment context
- Same pattern for Gradient Boosting and Random Forest

# ML component: summary and implications

- Per-read classifier for clean vs PCR chimeric reads
- Evaluation:
  - Stratified 80/20 split, 5-fold CV, held-out test set
- Best models: tree-based ensembles
  - CatBoost, Gradient Boosting, LightGBM
  - Test F1  $\approx 0.77$ , ROC-AUC  $\approx 0.84$
- Default threshold:
  - Conservative chimera filter (high clean recall, high chimera precision)
  - Removes  $\sim 2/3$  of chimeras
- Features match PCR chimera biology
- Practical, interpretable pre-filter before mitochondrial assembly