

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 7, 2025

Outline

1 Train–Test Split and Validation

2 Model Zoo and Training

3 Metrics and Interpretation

4 Results

5 Discussion and Conclusion

Stratified Train–Test Split

- First step: **create a held-out test set** for final evaluation.
- Use `build_datasets.py`:
 - ① Combine clean and chimeric feature tables.
 - ② Attach labels (0 = clean, 1 = chimeric) if missing.
 - ③ Shuffle and perform **stratified** split:

Train : Test = 80% : 20%

with the same class proportions in each split.

- Output:
 - `train.tsv` (used for model selection and cross-validation).
 - `test.tsv` (kept untouched until the very end).

5-Fold Stratified Cross-Validation

- On the **training set only**, we perform:

5-fold stratified cross-validation

- Procedure:

- Split training data into 5 folds with balanced 0/1 labels.
- For each fold:
 - Train the model on 4 folds.
 - Evaluate on the remaining fold.
- Average metrics across the 5 folds:

mean F1 \pm std, mean accuracy \pm std

- This tells us:

- Typical performance** on unseen data.
- Stability** of each model (via standard deviation).
- Helps guide which algorithms are promising before going to the test set.

Model Zoo: Algorithms Compared

- We implemented a panel of 13 classifiers using scikit-learn and gradient boosting libraries:
 - **Baseline:** Dummy (always predicts most frequent class).
 - **Linear models:** Logistic regression (logreg_12), linear SVM with calibration.
 - **Tree ensembles:**
 - Random Forest, Extra Trees.
 - Gradient Boosting (sklearn).
 - XGBoost, LightGBM, CatBoost.
 - Bagging with decision trees.
 - **Others:** k-NN, Gaussian Naive Bayes, shallow MLP.
- All models use the same preprocessing pipeline:

Imputer (median) → StandardScaler → Classifier

Hyperparameter Tuning for Top Models

- For the 10 strongest families, we perform **RandomizedSearchCV** with 5-fold CV:
 - Logistic regression, linear SVM (calibrated).
 - Random Forest, Extra Trees, Gradient Boosting.
 - XGBoost, LightGBM, CatBoost.
 - Bagging (trees), MLP.
- Each search explores combinations of:
 - Tree depth, number of estimators, learning rate, subsample ratios, etc.
 - For MLP: hidden layer sizes, regularization (α), learning rate.
- Selection criterion:
 - Choose the hyperparameters with the best **cross-validated F1-score**.
 - Re-fit the best model on the **full training set**, then evaluate on the held-out test set.

Classification Metrics (Per-Read)

- For each model, on the test set we compute:

- Accuracy:**

$$\frac{\# \text{ correct predictions}}{\# \text{ all predictions}}$$

- Precision** (for chimeras):

$$\frac{TP}{TP + FP}$$

Of the reads we call “chimeric”, how many are truly chimeric?

- Recall** (for chimeras):

$$\frac{TP}{TP + FN}$$

Of all true chimeric reads, how many did we detect?

- F1-score** (for chimeras):

$$F1 = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}}$$

Harmonic mean: high only if both precision and recall are high.

Threshold-Free Metrics: ROC–AUC and PR Curves

- Our models output a **score** per read (probability of being chimeric).
- By sweeping a threshold on this score, we can draw:
 - **ROC curve:**
 - x-axis: False Positive Rate (FPR).
 - y-axis: True Positive Rate (TPR = recall).
 - **ROC–AUC** = area under the curve.
 - **Precision–Recall (PR) curve:**
 - x-axis: Recall.
 - y-axis: Precision.
 - **Average Precision (AP)** = area under PR curve.
- Intuition for ROC–AUC:

$\text{AUC} \approx 0.84 \Rightarrow 84\% \text{ chance a random chimera is scored higher than a random non-chimera}$

Overall Performance Across Models (Test Set)

Model	CV Acc	CV F1	Test Acc	Test F1	ROC-AUC
Dummy baseline	0.50	0.67	0.50	0.67	0.50
Logistic regression	0.79	0.75	0.79	0.74	0.82
Linear SVM (cal.)	0.79	0.75	0.79	0.74	0.82
Random Forest	0.80	0.77	0.79	0.75	0.83
Extra Trees	0.80	0.77	0.79	0.75	0.82
Gradient Boosting	0.81	0.78	0.80	0.77	0.84
XGBoost	0.81	0.77	0.80	0.76	0.84
LightGBM	0.81	0.77	0.80	0.76	0.84
CatBoost	0.81	0.78	0.80	0.77	0.84
k-NN	0.78	0.75	0.78	0.75	0.81
Gaussian NB	0.75	0.66	0.74	0.65	0.82
Bagging (trees)	0.80	0.77	0.79	0.76	0.84
MLP	0.79	0.75	0.79	0.75	0.82

Table: Summary of cross-validation and test performance (chimeric class F1).

ROC and PR Curves (Placeholder)

figures/roc_curves.pdf

figures/pr_curves.pdf

ROC curves (CatBoost, GBM, RF, logreg)

- ROC–AUC for top models: ≈ 0.84 .

• Curves pushed towards the top-left / top-right illustrate strong separation.

Confusion Matrix and Class-Wise Behaviour (CatBoost)

figures/cm_catboost.pdf

CatBoost (test set, illustrative)

clean: precision ≈ 0.73 , recall ≈ 0.95

chimeric: precision ≈ 0.92 , recall ≈ 0.66

overall accuracy ≈ 0.80

Placeholder: confusion matrix for CatBoost on test set

- **Clean reads:**

- Very high recall: most true clean reads are correctly kept.

- **Chimeric reads:**

Effect of Hyperparameter Tuning (F1 and ROC–AUC)

Model	F1 (base)	AUC (base)	F1 (tuned)	AUC (tuned)
CatBoost	0.767	0.839	0.769	0.844
Gradient Boosting	0.766	0.840	0.767	0.843
LightGBM	0.764	0.838	0.766	0.842
XGBoost	0.765	0.839	0.765	0.839
Random Forest	0.755	0.834	0.763	0.842
Bagging (trees)	0.760	0.837	0.763	0.842
Extra Trees	0.753	0.824	0.760	0.837
MLP	0.748	0.819	0.749	0.821
Logistic reg.	0.744	0.821	0.743	0.818
Linear SVM (cal.)	0.744	0.820	0.743	0.818

Table: Test F1 and ROC–AUC before vs after hyperparameter tuning.

- Tuning yields **modest but consistent gains** in F1 and ROC–AUC.
- Confirms that the initial defaults were already reasonable, but performance can be further refined.

Permutation Feature Importance (Placeholder)

figures/perm_importance_catboost.pdf

Placeholder: permutation importance for CatBoost

- Top features across CatBoost, GBM, RF:

- total_clipped_bases
- kmer_js_divergence, kmer_cosine_diff
- softclip_left, softclip_right
- mapq

- Interpretation:

- Chimeras are characterized by **large clipped segments and abrupt k-mer composition shifts.**
- Aligners are already “seeing” the breakpoint signal; the ML model learns to combine these signals into a chimera score.



Summary of Findings

- We built a per-read feature table capturing:
 - Alignment and clipping patterns.
 - Supplementary alignments and breakpoint distances.
 - Sequence-level k-mer divergence and microhomology.
- A broad panel of ML models was evaluated:
 - Tree-based ensembles (CatBoost, Gradient Boosting, Random Forest, LightGBM, XGBoost) achieved the **best performance**.
 - Test F1 for chimeras $\approx 0.76\text{--}0.77$, ROC-AUC ≈ 0.84 .
- Model behaviour:
 - Conservative on clean reads (high recall).
 - High precision on chimeric reads, moderate recall.

Implications for Mitochondrial Assembly

- The ML classifier can be used as a **pre-filter** before assembling mitochondrial genomes:
 - Remove high-confidence chimeric reads to reduce false junctions.
 - Retain the majority of clean reads to preserve coverage.
- Especially useful for:
 - Small, circular, and repetitive organellar genomes where chimeras are particularly harmful.
 - Scenarios without high-quality reference genomes or abundance information.
- The feature importance analysis provides biological insight:
 - Confirms the role of soft-clipping, supplementary alignments, and k-mer jumps as core signals of chimeric structure.

Limitations and Future Work

- Current study uses **simulated** chimeras and a single species:
 - Need to validate on real experimental datasets.
 - Extend to other organellar genomes and library preparations.
- Classifier currently treats each read independently:
 - Future work: incorporate read-pair information, local read depth, or graph features.
- Integration into practical pipelines:
 - Wrap as a command-line tool interfacing with standard BAM/FASTQ workflows.
 - Benchmark impact on final assembly quality (contiguity, misassemblies).

Conclusion

- We developed a **machine learning pipeline** that:
 - Learns from alignment- and sequence-based features.
 - Achieves strong separation between clean and chimeric reads.
- Tree-based gradient boosting models (CatBoost, GBM, RF) provide:
 - High test F1 and ROC-AUC.
 - Interpretable feature importance aligned with known chimera mechanisms.
- This framework is a step towards **reference-free chimera detection** tailored for organellar genomes and low-resource settings.

Thank You

Questions?