Feature Importance Methods for CIN Feature Prediction: A Comparative Analysis

1. Gain (a.k.a. “training-loss reduction”)

What it is – Every time XGBoost splits on a gene, it records how much the split lowers the training loss; Gain is the sum of those reductions over all trees.

Pros

• Zero extra computation (comes “for free” from the booster).

• Gives a quick rough screen of which genes the model leans on.

Cons

• Inconsistent: if the model is retrained and now relies more on a gene, its Gain can still fall because the split moved nearer the root.

• Inflated by proxy genes when many correlated transcripts could do the same job.

• Unsigned: tells you nothing about whether high expression drives CIN up or down.

• Bias toward early splits: features that split nearer the root gather larger loss reduction and thus higher Gain, while equally useful features chosen deeper in the tree are systematically undervalued.

Suitability – Fine for debugging or early triage, but not trustworthy for the final biological ranking you need. Keep it as a development-time sanity check, not as evidence.

2. Cover

What it is – Counts (with sample weights) how many tumours fall through splits that use a given gene.

Pros

• Highlights genes whose effect is broad (affects many samples).

• No extra cost to compute.

Cons

• Ignores how much each split helped; a ubiquitous but weak effect can look big.

• Shares Gain’s inconsistency and bias toward lower tree levels.

• Still unsigned and blind to correlated-gene leakage.

Suitability – Offers a complementary “breadth” view, but cannot stand alone. Use it only to spot genes that act in many tumours, then confirm with a consistent metric.

3. LOFO (Leave-One-Feature-Out / Drop-column)

What it is – Retrain the learner with one gene removed and measure the drop in validation performance.

Pros

• Gold-standard for necessity; fully consistent by construction.

• Immune to proxy leakage: the model can adapt, so only unique information is scored.

• Produces an effect size (Δ-AUROC, Δ-RMSE) that wet-lab colleagues immediately grasp.

Cons

• Computationally heavy (retraining 327 genes × 56 tasks).

• Variance from re-optimisation unless you lock seeds/hyper-parameters.

• Impractical at scale unless you first narrow to ≈ 20 top genes or merge colinear transcripts into groups.

Suitability – Ideal as a final validation step for the handful of genes you are ready to test experimentally. Too costly for a full sweep, but unbeatable for credibility.

4. TreeSHAP (SHAP for trees)

What it is – An exact Shapley-value algorithm that decomposes each prediction into signed contributions from every gene; summing |SHAP| over samples yields a global importance.

Pros

• Consistent and complete (attributions add up to the model output).

• Fast (linear in number of leaves); scales to all 56 CIN learners.

• Signed scores show whether over- or under-expression pushes CIN up or down.

• Local explanations let you trace patient-specific mechanisms (means we can also break out accurate predictions from misclassified ones to assess the feature importance based only on correct predictions)

Cons

• Still reflects proxy overlap: interchangeable genes can split the credit.

• Memory-hungry if you keep per-sample matrices.

Suitability – The best first-line metric: rigorous enough for publication, fast enough for routine use, rich enough for mechanistic insight.

5. Permutation Importance

What it is – Shuffle one gene’s expression in the validation set, leave the model intact, and record the performance drop.

Pros

• Consistent under i.i.d. shuffles.

• No retraining – far cheaper than LOFO.

• Outputs the same familiar performance metric as LOFO.

Cons

• Proxy leakage: if a correlated gene stays untouched, the drop may be tiny.

• Requires multiple shuffles or grouped-gene permutations to stabilise scores.

• Not additive, so you cannot decompose the whole prediction.

Suitability – Excellent as a quick, faithful cross-check of TreeSHAP rankings, provided you cluster highly correlated heterochromatin genes before shuffling.

Workflow recipes

Scenario A TreeSHAP → LOFO

1. Run TreeSHAP on every CV fold of each CIN learner; aggregate mean|SHAP| to produce the global ranking.

2. Filter to genes that (i) rank in the top X for the CIN feature and (ii) are not physically located on that very aberrant region (per your NOTE2 plan).

3. Optionally cluster colinear transcripts (e.g. HP1 paralogues) to cut compute.

4. Apply LOFO only to the top 5–10 genes per target that pass the filter. Record Δ-metric with fixed seeds for reproducibility.

5. Interpret: genes that retain high TreeSHAP and large LOFO drops are prime candidates for wet-lab validation.

Scenario B TreeSHAP → Permutation Importance

1. Compute TreeSHAP exactly as above to get an initial, signed gene ranking.

2. Cluster correlated genes (PCA or hierarchical clustering) so that each cluster is permuted as a unit, reducing proxy leakage.

3. Run grouped permutation importance on the validation fold; repeat shuffling ≥ 5 times and average the Δ-metric.

4. Compare ranks: keep genes (or clusters) where permutation Δ agrees with high TreeSHAP values.

5. Optional LOFO spot-check the very top genes to assure reviewers that the shuffled results are not artefacts.

Both paths start with TreeSHAP for breadth and consistency, then apply a heavier second lens—LOFO for necessity, or permutation for faster effect-size confirmation—ensuring that your final gene list is both computationally vetted and biologically convincing.