QRT Football Match Prediction — Methodology & Findings

Challenge: QRT (Qube Research & Technologies) × ENS *Challenge Data* — predict a match outcome (`HOME_WINS` / `DRAW` / `AWAY_WINS`) from pre-match statistics. **Metric:** 3-class accuracy. **Result: public leaderboard 0.4922, rank 19 / 960 (top 2%).** References: always-predict-home = 0.435, organizer benchmark ≈ 0.475, realistic ceiling ≈ 0.50.

1. The problem, and what makes it hard

Each match is described by team statistics (~140 per side) and player statistics (~300 for each of 15–25 players per side), as both season-to-date and last-5-match aggregates. Three properties shaped every decision:

- **The test set is anonymized.** Team / player / league names are present in train but stripped from test, so any identity-based feature (e.g. Elo) is unusable at inference and was excluded from the start.
- **All features are rank-scaled** to integers ~[0, 10], with missing values kept separate from 0. So `0` is a genuine value and raw magnitudes are gone.
- **Low intrinsic ceiling (≈ 0.50).** This is an *observed/practical* ceiling, not a hard bound, and it comes from three converging signals: 1. *Baselines are already high, leaving little headroom* — always-predict-home scores 0.435 and the organizer benchmark 0.475. 2. *Results empirically plateau there* — across this project, honest cross-validation topped out around 0.498–0.50 and the strongest public leaderboard entries clustered at ~0.492–0.494, with no technique breaking past it. When many independent attempts converge on the same number, that number is the effective ceiling. 3. *The domain caps it* — **draws (~26% of matches) are close to unpredictable**, so a large share of matches are near coin-flips; even bookmakers, with far richer data, reach only ~0.53–0.55 on 3-way football outcomes.

The consequence: the whole problem is a fight for fractions of a percent, which makes *validation discipline*, not model novelty, the deciding factor.

2. Validation: the backbone

Because gains live in the third decimal, the validation pipeline mattered more than any single model.

- **Repeated Stratified K-Fold** (5×3), one shared fold scheme across all models, so out-of-fold (OOF) predictions are mutually compatible for leak-free ensembling.
- **Nested early stopping.** Each gradient-boosting model carves an *inner* holdout from its training fold for early stopping, so the OOF fold is never used to choose iteration count.
- **Honest nested-CV for selection.** Blend and per-class decision weights are fit on training folds and scored on held-out folds, averaged over seeds. Naïve "fit on all OOF, score on all OOF" figures were

~0.001–0.002 optimistic and were never used to choose between near-tied configurations.

3. Feature engineering

- **Team block:** raw home/away stats plus explicit **home–away contrasts** (difference and a stabilized ratio) — relative strength predicts the result.
- **Player → team aggregation:** mean / sum / max / min / std of every player stat, overall and per position, with home–away contrasts. `max` captures the "best player" signal.

This produced ~6,000 features; a regularized model generalized as well or better, confirming the data is information-limited rather than feature-limited. Additional hand-crafted features (form deltas of last-5 vs season; "expected-goal-difference" style attack-vs-defence matchups; minutes-weighted XI quality) were each tested on honest CV and **did not beat the baseline** — gradient-boosted trees already recover those relationships from the raw aggregates. **Feature engineering is saturated on this dataset.**

4. Models

Diversity was favored over polishing any single algorithm:

Model	OOF accuracy (5×3)
LightGBM (Optuna-tuned)	0.4987
CatBoost	0.4970
XGBoost	0.4970
ExtraTrees	0.4966
LightGBM (default)	0.4939
HistGradientBoosting	0.4926
MLP	0.4906
DeepSets (player set)	0.4870
TabNet	0.4829
Set-Transformer (player set)	0.4822

Notable architectures: - **DeepSets / Set-Transformer over the raw player set** — permutation-invariant networks that learn the player→team aggregation end-to-end (the Set-Transformer adds self-attention so players model each other). Despite low standalone CV, DeepSets earned meaningful blend weight through decorrelation. - Hyperparameters tuned with **Optuna**. - Engineering note: PyTorch and LightGBM ship conflicting OpenMP runtimes and segfault in one process on macOS, and CatBoost can stall at full thread count. The pipeline runs NN and GBM training in **isolated processes** and caps CatBoost threads.

5. Ensembling

- OOF predictions combined by a **geometric-mean blend** (a log-opinion pool, more robust than arithmetic averaging and slightly better than a log-loss-weighted blend; rank-averaging was much worse).
- A final **per-class decision-weight** step (equivalent to per-class thresholds) tuned on OOF.

Final model: geometric-mean blend of {CatBoost, XGBoost, LightGBM-tuned, HistGB, ExtraTrees, DeepSets} + decision weights → honest CV ≈ 0.498, **LB 0.4922.**

A metric-specific finding: for accuracy the optimum **predicts DRAW almost never** — draws are close to unpredictable, so for nearly every match home or away is the higher-accuracy call. A draw-free 2-class model was slightly *worse* than keeping draw as a training target but rarely predicting it.

6. The leaderboard-noise analysis (central result)

The honest CV (~0.498) sat consistently above the leaderboard (~0.492). Crucially, improvements that were **robust on cross-validation did not transfer:**

Config	honest CV vs. base	leaderboard
geometric blend (final)	baseline	**0.4922**
drop ExtraTrees	+0.0008, robust 16/20 seeds	0.4920
+ Set-Transformer (7 models)	+0.0010, robust 17/20 seeds	0.4919

The two highest-CV, most-rigorously-validated configurations scored **lowest** on the leaderboard. This is not bad luck: the leaderboard's binomial sampling noise is **≈ ±0.003**, which is 3–4× larger than any honest CV edge we could find (≈ ±0.001). Past this point, cross-validation can no longer rank submissions — every change is a coin flip around the true value, and the original blend sits at the top of that noise band. Recognizing this was the disciplined stopping criterion.

7. What didn't help (all tested on honest CV)

- Extra feature generations (form deltas, attack-vs-defence matchups, minutes-weighted XI, player-rating distributions) — redundant with the trees.
- Seed-bagging the members for variance reduction — netted flat; bagging the dominant correlated boosters actually reduced blend diversity.
- Pseudo-labeling the test pool — confident predictions are heavily home-biased and draw-free, so aggressive use hurt and conservative use was within noise.
- Adding TabNet / MLP / a logistic member to the blend — they added noise.
- Caruana hill-climb selection, stacking, accuracy-weighted blending — all ≤ the geometric blend.

8. Key takeaways

1. **Validation discipline beats model novelty** on low-signal data. Honest nested-CV, and knowing when leaderboard noise exceeds CV resolution, mattered more than any single model.
2. **Diversity, then stop.** A geometric-mean blend of decorrelated models is the workhorse; once members agree, adding more only reduces variance within noise.
3. **Negative results are results.** Systematically proving what *doesn't* work defined the genuine ceiling of an anonymized, information-limited dataset.
4. **Final: 0.4922, rank 19 / 960 (top 2%)** — at the practical ceiling, reached entirely through evidence-driven decisions.

9. Repository

```
src/        config, data_loader, features, validation, ensemble, submission,
            tuning, player_tensors, models/{gbm, sklearn_models, nn_tabular,
            deepsets, set_transformer}
pipelines/  run_cv_model (GBMs), run_extras (ExtraTrees/LogReg), run_cv_nn
            (MLP/TabNet), run_deepsets, run_settransformer, run_tuning (Optuna),
            build_ensemble
artifacts/  oof/ (per-model OOF+test matrices), models/ (tuned params),
            submissions/ (final submission)
```

10. Reproduce

```
python -m venv .venv && source .venv/bin/activate
pip install -r requirements.txt
# place the challenge CSVs under data/ (Train_Data/, Test_Data/, benchmark_and_extras/)

python -m src.features                          # build & cache feature matrices
python -m pipelines.run_cv_model cat            # + xgb / histgb
python -m pipelines.run_cv_model lgbm --tuned   # Optuna-tuned LightGBM
python -m pipelines.run_extras et               # ExtraTrees
python -m pipelines.run_deepsets                # DeepSets member
python -m pipelines.build_ensemble              # ->
artifacts/submissions/submission_geomean.csv
```

The six final members' OOF matrices are committed, so `build_ensemble` reproduces the exact submission without retraining. Built and run on Apple Silicon (Python 3.12, MPS for the neural members).