Leak-Free Greyhound Race Forecasting with Gradient Boosting and Sequence-Attention Ensembles

From Production CatBoost v6 to Research Ensemble v11

Jon Jovinsson

2026-03-31

Abstract

We present a leak-aware forecasting pipeline for greyhound racing in which pre-race feature snapshots and post-race result labels are deliberately separated during model development. The work began with a production CatBoost baseline (v6) built from aggregated form descriptors over the four most recent valid runs, and progressed through a series of controlled research iterations including a failed weather ablation (v7), race-level ranking models (v8), track-distance specialization (v9), and finally a two-branch ensemble model (v11) that combines gradient-boosted tabular learning with a sequence-and-field neural network. The neural branch encodes up to eight historical runs per dog with a bidirectional LSTM, fuses the sequence embedding with static features, and performs cross-runner self-attention across the field before race-normalized prediction. A logistic meta-learner stacks the neural and CatBoost branches. On the clean March 17, 2026 to March 30, 2026 test window, v6 achieves 30.15% top-1 accuracy and +20.53 flat-stake units (ROI 2.72%), while the v11 ensemble reaches 32.93% top-1 accuracy and +50.59 units (ROI 6.46%). Short-price stability analysis further suggests that the most robust edge lies below roughly odds of 1.60 rather than in more weakly calibrated value bands. The paper is both a technical account of model design decisions and a practical case study in building a reproducible wagering model under real-world data quality constraints.

Table of contents

1 Introduction

Forecasting greyhound races is an unusually demanding applied machine learning problem. Each event is a small structured field, the label space is highly imbalanced at the runner level, and the useful information is distributed across multiple temporal scales:

- long-horizon career priors
- short-horizon form cycles
- track-distance specialization
- intra-race interactions such as pace and box pressure
- market information that is informative but operationally unstable close to jump

The project documented here started as a practical attempt to build a deployable greyhound selection engine, but the resulting research problem is broader: how should one design a model family that is simultaneously

- statistically disciplined
- leak resistant
- operationally deployable
- and strong enough to survive direct profit-and-loss evaluation rather than classification accuracy alone

The engineering lesson that shaped the entire research program is simple: evaluation quality is determined first by *data protocol*, then by model complexity. Early live testing exposed a class of failures where predictions were produced before form histories were fully collected, and later evaluation sometimes relied on provisional result blocks embedded in pre-race files. Those issues motivated a hard separation between two research data stores:

- `data/prerace/`: feature snapshots exactly as seen before the race
- `data/postRaceContaminated/`: post-race files used only as the source of labels

From that point onward, the core design objective became *leak-free race modeling*: every feature must be computable from information available before the scheduled start, while all labels, prices, and post-race outcomes are read from a separate source of truth.

This paper makes four main contributions.

1. It formalizes the leak-free evaluation protocol used in the repository.
2. It documents the `v5 -> v11` model progression, including negative results.
3. It presents a strong production baseline (`v6`) and a stronger research challenger (`v11`) under a shared clean test window.
4. It introduces a bankroll-aware short-price analysis that reframes model quality in terms of stability and compounding potential, not only ROI.

The document is intentionally written to do double duty. It is designed to be read as a legitimate technical preprint, but also as a hiring artifact that shows end-to-end ownership of data curation, model design, evaluation hygiene, and deployment tradeoffs.

2 Problem Formulation

Let race r contain n_r runners. For runner i in race r, define a pre-race feature vector

$$\mathbf{x}_{r,i} \in \mathbb{R}^d$$

possibly augmented by categorical fields and an ordered form-history sequence

$$\mathbf{S}_{r,i} = (\mathbf{s}_{r,i,1}, \ldots, \mathbf{s}_{r,i,T}), \qquad T \leq 8.$$

The learning task is to estimate a race-normalized win distribution

$$p_{r,i} = P(i \text{ wins race } r \mid \mathbf{x}_{r,1:n_r}, \mathbf{S}_{r,1:n_r}), \qquad \sum_{i=1}^{n_r} p_{r,i} = 1.$$

The primary operational action is to select the top-ranked runner

$$\hat{i}_r = \arg\max_i p_{r,i}.$$

This is not a generic i.i.d. binary classification problem. It is a structured race-relative ranking problem in which the score assigned to a dog should depend not only on that dog's own form, but also on the composition of the field. The project therefore evaluates models at three distinct levels.

2.1 Ranking Metrics

For each race, we record:

- **Top-1 accuracy**: whether the highest-probability runner is the winner.
- **Winner rank**: the predicted rank position of the actual winner.
- **Mean reciprocal rank (MRR)**:

$$\text{MRR} = \frac{1}{R} \sum_{r=1}^{R} \frac{1}{\text{rank}_r(\text{winner})}.$$

MRR is especially useful in this setting because it measures not only whether the winner was ranked first, but how close it was to the top when not first.

2.2 Flat-Stake Betting Metrics

For top-pick wagering, let the observed starting price for the selected runner in race r be o_r and the event outcome be $y_r \in \{0, 1\}$. A one-unit flat-stake return is

$$g_r = \begin{cases} o_r - 1, & y_r = 1 \\ -1, & y_r = 0. \end{cases}$$

Profit and loss over R bets is

$$\text{P/L} = \sum_{r=1}^{R} g_r,$$

with ROI

$$\mathrm{ROI} = \frac{1}{R}\sum_{r=1}^{R} g_r.$$

Flat ROI remains useful, but this project increasingly treats it as a *partial* summary rather than the final decision criterion.

2.3 Bankroll-Managed Return (BMR)

For stability analysis we also simulate a fixed-fraction bankroll process, motivated by the log-growth logic of Kelly-style betting (Kelly 1956). Let B_t denote bankroll before bet t and f a fixed stake fraction. Then

$$B_{t+1} = B_t(1 + f g_t).$$

From this sequence we compute:

- ending bankroll multiple
- maximum drawdown
- mean log growth per bet

This lens matters because a small positive ROI bucket can still be highly attractive if it has a high strike rate, shallow drawdown, and frequent compounding opportunities.

3 Data Protocol and Research Corpus

3.1 Two Sources of Truth

The cleaned research pipeline now treats the following directories as the only authoritative stores:

- `data/prerace/`
- `data/postRaceContaminated/`

The naming is intentionally explicit. The first directory contains frozen pre-race state; the second contains label-bearing post-race state. The historical term `raw` was abandoned precisely because it hid the semantic role of the files.

The fundamental rule is:

- **features come from `prerace`**
- **labels come from `postRaceContaminated`**

This prevents a subtle but extremely important contamination pattern: re-scraping a race after it has been run and then accidentally allowing post-race information to leak into the feature set.

3.2 Training / Test Split Logic

The current research setup uses:

- training source: `data/postRaceContaminated/`

- test feature source: `data/prerace/`
- test label source: `data/postRaceContaminated/`

For `v11`, the nominal training date range is January 1, 2022 to March 16, 2026, followed by a temporal split into:

- a fit-train block
- a holdout validation block
- a recent meta-training block for the stacker
- a clean prerace/post-race test window

The main clean comparison in this paper uses March 17, 2026 through March 30, 2026 as the evaluation horizon. The repository evaluator reports 824 prerace events loaded for that window, with 753 races contributing to the fully scored race-level comparison after matching and filtering.

3.3 Form Horizons

Two different temporal horizons are used by the main models:

- `v6`: up to 4 valid historical runs, aggregated into summary statistics
- `v11`: up to 8 valid historical runs, preserved as a sequence tensor for the neural branch while still supporting aggregated features for the tabular branch

This distinction is central. The `v6` representation is intentionally compact and stable. The `v11` representation explicitly models *trajectory*.

4 Model Evolution: v5 to v11

The project evolved through a sequence of increasingly ambitious model families. Not every step was an improvement, and documenting the failed branches is important because it explains why the final design looks the way it does.

Table 1: Chronology of the model family from v5 through v11, with artifact-backed quantitative notes where available.

Version	Core idea	Key result	Status
v5	4-run aggregate CatBoost	Established the tabular baseline	Historical baseline
v6	Leak-aware CatBoost baseline	30.15% top-1, +20.53u, +2.72%	Production baseline
v7	Weather CatBoost ablation	Weather worsened P/L by -4.10u	Rejected
v8	CatBoost/XGBoost rankers	XGBoost beat CatBoost by +10.28u	Useful side branch
v9	Track-distance and box-bias features	Older-corpus uplift of +8.90u	Feature family promoted
v10	Architecture staging	Staging branch; no promoted result	Not promoted
v11	LSTM-attention + CatBoost stack	32.93% top-1, +50.59u, +6.46%	Current research leader

Two points are worth emphasizing.

First, `v7` was useful precisely because it failed. It established that simply adding external weather covariates to the existing tabular stack did not create meaningful lift in this pipeline.

Second, `v8` and `v9` were not dead ends. `v8` demonstrated that race-relative ranking objectives were worthwhile, and `v9` supplied the track-distance normalization and box-bias feature family that later fed into `v11`.

5 v6: The Production Baseline

5.1 Feature Space

`v6` is a CatBoost classifier (Dorogush et al. 2018) built from 33 features:

- 27 numeric features
- 6 categorical features (`dog_name`, `trainer`, `sire`, `dam`, `track`, `grade`)

It uses a deliberately compressed summary of recent form. For each runner, the pipeline computes aggregates over the four most recent valid historical runs, including:

- career starts, win rate, and place rate
- track-distance starts and win rate
- best time and best split
- age in days
- number of valid form runs
- average finishing position
- recent win/place rates
- average margin
- average and best recent times
- average recent first split
- days since the most recent run
- field-relative win-rate advantages
- opponent-average strength summaries

The key design decision was to turn a variable-length historical record into a compact fixed-width vector with strong inductive bias and low operational complexity.

5.2 Objective and Hyperparameters

Given a runner-level label $y_{r,i} \in \{0, 1\}$ for whether runner i wins race r, CatBoost optimizes a standard logloss objective

$$\mathcal{L}_{\mathrm{CE}} = -\sum_{r,i} \left[y_{r,i} \log \hat{p}_{r,i} + (1 - y_{r,i}) \log(1 - \hat{p}_{r,i}) \right].$$

The repository configuration for `v6` uses:

- 3000 iterations
- depth 6
- learning rate 0.01
- L_2 leaf regularization 5
- balanced class weighting
- minimum leaf size 50
- early stopping after 200 non-improving rounds

Although the model is trained at the runner level, prediction is evaluated at the race level by taking the runner with highest posterior win probability.

5.3 Why v6 Still Matters

`v6` remains the production model not because it is the most sophisticated architecture in the repository, but because it combines three desirable properties:

1. strong live behavior
2. feature stability
3. low inference complexity

That combination is rare in practical ML systems. A baseline that is both accurate and reliably deployable deserves to survive even after stronger research challengers emerge.

6 v11: Sequence- and Field-Aware Modeling

`v11` was built to capture exactly the information that `v6` intentionally compressed away.

6.1 Expanded Feature Engineering

The `v11` data layer incorporates all `v9` features and adds a new block of runner-level covariates:

- implied market probability
- PIR-based running style summaries
- weight delta and weight trend
- BON-gap summaries (`time - bon`)
- grade ordinal and grade delta
- time-of-day indicators
- trainer, sire, and dam win-rate priors
- extra card and placing descriptors

The main conceptual change is that `v11` no longer treats the last few runs as exchangeable observations. Instead, it keeps the order.

For each runner it constructs a form tensor of shape

$$\mathbf{S}_{r,i} \in \mathbb{R}^{8\times 15},$$

where each timestep contains:

- normalized finishing position
- field size
- time
- win time
- first split
- margin
- SP
- weight
- PIR
- BON gap
- days ago
- grade ordinal
- same-distance indicator

- same-track indicator
- win indicator

This turns the model from "feature summarization" into "trajectory modeling."

6.2 Neural Branch

The neural branch, `V11Net`, is a race-level model with five stages:

1. categorical embedding of static runner attributes
2. MLP encoding of static numeric + embedded categorical features
3. bidirectional LSTM over the raw form sequence (Hochreiter and Schmidhuber 1997)
4. fusion of static and temporal embeddings
5. multi-head self-attention across all runners in the race (Vaswani et al. 2017)

The architecture is conceptually close to a set-aware field model (Zaheer et al. 2017): each runner receives a local embedding, but the final score is conditioned on the other runners in the field.

Let $\mathbf{a}_{r,i}$ denote static features and $\mathbf{S}_{r,i}$ the form sequence. The model computes

$$\mathbf{h}_{r,i}^{\text{static}} = f_\theta(\mathbf{a}_{r,i}),$$

$$\mathbf{h}_{r,i}^{\text{seq}} = \text{BiLSTM}_\phi(\mathbf{S}_{r,i}),$$

$$\mathbf{u}_{r,i} = g([\mathbf{h}_{r,i}^{\text{static}}; \mathbf{h}_{r,i}^{\text{seq}}]).$$

The set of runner embeddings

$$U_r = \{\mathbf{u}_{r,1}, \dots, \mathbf{u}_{r,n_r}\}$$

is then passed through multiple cross-runner attention layers, producing contextualized race-aware embeddings $\mathbf{c}_{r,i}$. Final logits $z_{r,i}$ are converted to race probabilities via softmax:

$$p_{r,i} = \frac{\exp(z_{r,i})}{\sum_{j=1}^{n_r} \exp(z_{r,j})}.$$

The default neural hyperparameters in the repository are:

- hidden dimension 128
- sequence hidden dimension 64
- 2 attention layers
- 4 attention heads
- dropout 0.15
- AdamW with learning rate 5×10^{-4}
- cosine annealing schedule
- gradient clipping at 1.0
- early stopping patience 7

6.3 Loss Function

The neural model can train either with race-level cross-entropy or with a combined Plackett-Luce plus cross-entropy objective. The default path uses the combined loss:

$$\mathcal{L}_{\text{combined}} = \lambda \mathcal{L}_{\text{PL}} + (1 - \lambda)\mathcal{L}_{\text{CE}}, \qquad \lambda = 0.7.$$

The listwise Plackett-Luce component exploits partial finishing-order information rather than winner labels alone. If the ordered finish for a race is $(i_1, i_2, \ldots, i_m)$, then

$$P(i_1, i_2, \ldots, i_m) = \prod_{k=1}^{m-1} \frac{\exp(z_{i_k})}{\sum_{j=k}^{m} \exp(z_{i_j})}.$$

and the training loss is the negative log-likelihood of that ordered outcome. This is a principled way to give a model partial credit when it places the eventual winner second rather than eighth.

6.4 CatBoost Branch

The second branch of `v11` is a CatBoost classifier over the full `v11` feature set. It preserves the strong tabular inductive bias of `v6` while absorbing the new feature family learned from the `v7-v10` research cycle. The CatBoost parameters remain intentionally conservative:

- 3000 iterations
- depth 6
- learning rate 0.01
- L_2 leaf regularization 5
- balanced class weights
- minimum leaf size 50
- early stopping after 100 non-improving rounds

Predicted runner scores are race-normalized with a softmax layer before evaluation.

6.5 Stacked Ensemble

The final `v11` prediction is not a simple average. It uses a logistic meta-learner trained on a recent holdout block, following the stacked generalization framework of Wolpert (Wolpert 1992). If the neural and CatBoost branch probabilities are $p_{r,i}^{(N)}$ and $p_{r,i}^{(C)}$, the stacker learns

$$\Pr(y_{r,i} = 1) = \sigma(\beta_0 + \beta_1 p_{r,i}^{(N)} + \beta_2 p_{r,i}^{(C)}).$$

with race-level normalization applied after the meta-model.

This design matters because the two base models make different kinds of errors:

- CatBoost is strong on stable tabular priors and categorical interactions.
- The neural branch is better positioned to learn temporal form trajectories and field-relative interactions.

The ensemble is useful precisely when these error modes are not perfectly correlated.

7 Experimental Design

7.1 Evaluation Window

The main clean evaluation window spans March 17, 2026 to March 30, 2026. This window was chosen only after the missing post-race labels for March 27-30 were backfilled into `data/postRaceContaminated/`, so that test features remained pre-race and labels remained post-race.

7.2 Version Comparison Philosophy

Not all historical model reports are directly comparable. Some older branches were evaluated on noisier corpora before the current cleanup. Therefore this paper treats the development history in two layers:

- **historical direction finding**: `v7`, `v8`, `v9`
- **strict clean-window comparison**: `v6` vs `v11`

This distinction is important. A strong result on contaminated or partially labeled data can still be useful for hypothesis generation, but it should not be the sole basis for promotion.

8 Results

8.1 Historical Ablations and Branches

Three branch results were especially informative, and the current manuscript now pulls them directly from the saved JSON artifacts rather than hard-coding them.

Table 2: Historical branch experiments that materially shaped the v11 design.

Branch	Test	Delta top-1	Delta P/L	Delta loss	Takeaway
v7 weather	Weather vs baseline CatBoost	-1.33 pp	-4.10u	+0.00	Weather did not help
v8 rankers	XGBoostRanker vs CatBoostRanker	+1.29 pp	+10.28u	n/a	Ranking helped; XGBoost led
v9 specialization	Specialized CatBoost vs baseline	+1.86 pp	+8.90u	-0.01	Feature family fed into v11

These experiments were not wasted branches. They were hypothesis filters.

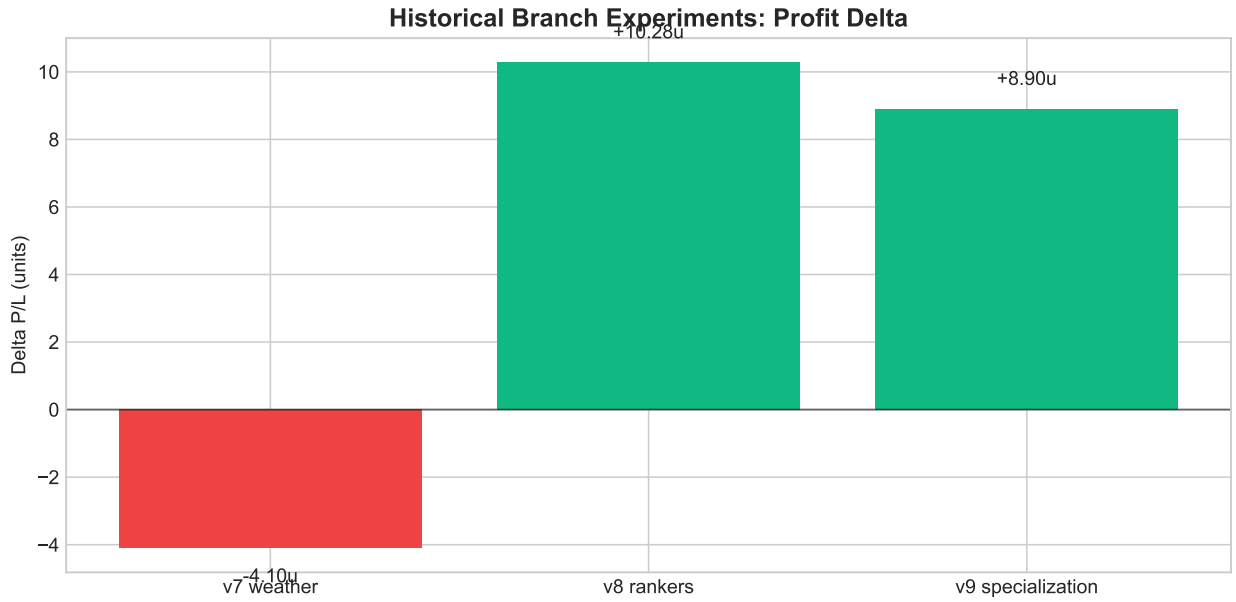
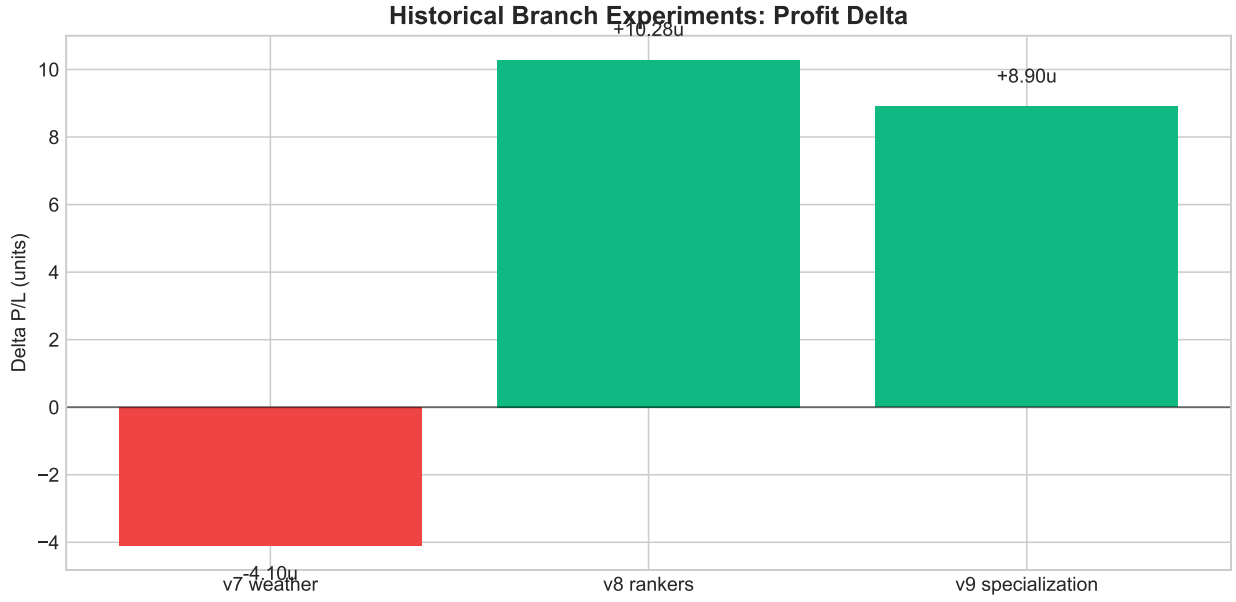


Figure 1: Historical branch experiments viewed through profit delta. Negative weather impact and positive ranking / specialization branches explain why the research program moved toward v11 rather than broader feature accretion.



8.2 Main Clean-Window Comparison

The primary result table is shown below.

Table 3: Main clean-window comparison on the March 17, 2026 to March 30, 2026 evaluation horizon.

Model	Races	Top-1	Win rank	MRR	Wins/Bets	SR	P/L	ROI
v6	753	30.15%	2.984	0.5217	226/756	29.89%	+20.53u	+2.72%
v11 neural	753	30.15%	3.159	0.5117	226/757	29.85%	+60.65u	+8.01%
v11 CatBoost	753	30.68%	3.117	0.5151	230/786	29.26%	+21.08u	+2.68%
v11 ensemble	753	32.93%	3.053	0.5319	247/783	31.55%	+50.59u	+6.46%

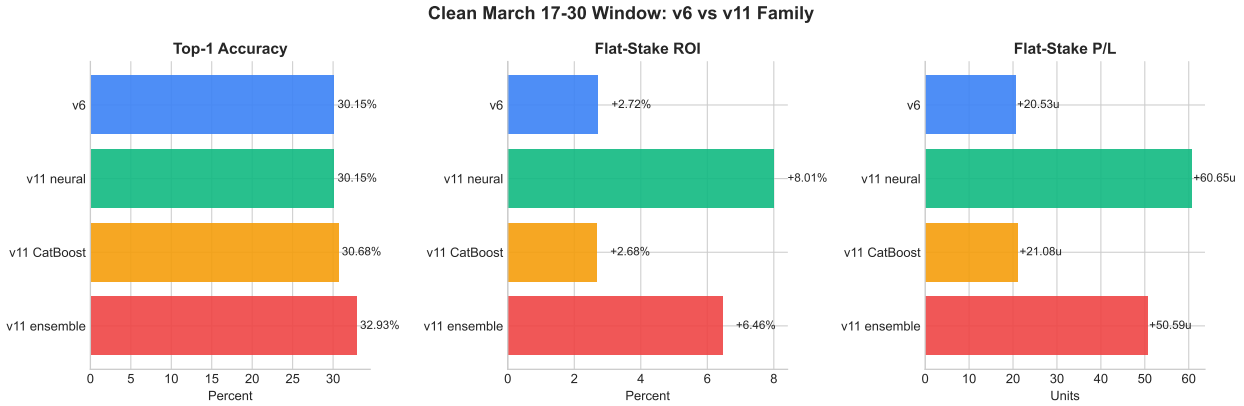
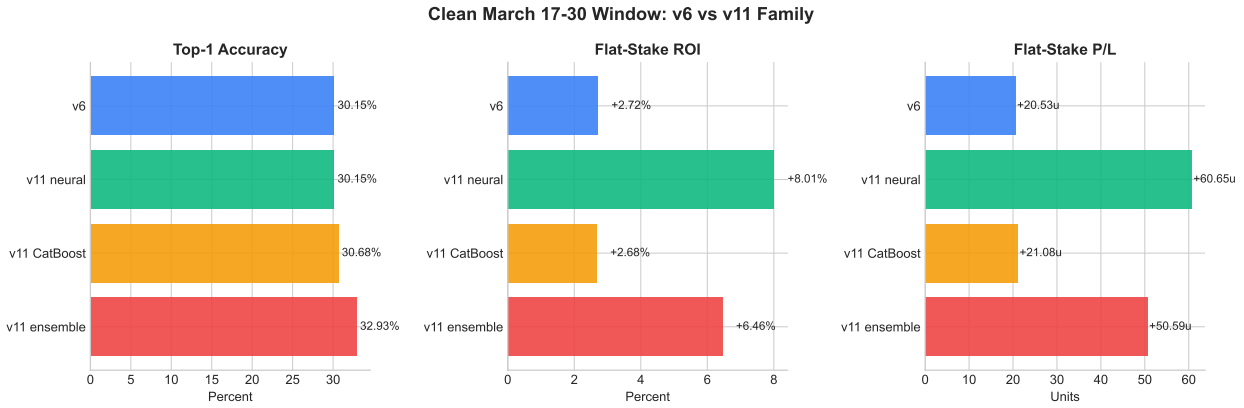


Figure 2: Clean-window trade-off across the production baseline and the v11 family. The ensemble leads on ranking accuracy and remains strongly profitable, while v6 stays competitive enough to justify its continued production role.



The interpretation is nuanced:

- `v11` ensemble is clearly best on top-1 accuracy, MRR, strike rate, and total flat-stake profit.
- `v11` neural is the strongest single base model in flat-stake ROI.
- `v6` remains competitive with `v11` CatBoost, which explains why it has remained viable in production.

In other words, `v11` wins because the *combination* is better than either branch alone, not because the tabular branch alone dominates the older system.

8.3 Short-Price Stability

The repository now includes a dedicated short-price/BMR analysis script that evaluates fine-grained odds buckets and cumulative price caps. The most important result is that the short-price edge appears to decay around odds of 1.60.

8.3.1 Cumulative caps

Table 4: Cumulative short-price caps at and below odds of 1.60, the regime that currently appears most stable under flat-stake and BMR views.

Model	Cap	Bets	Wins	SR	P/L	ROI	BMR	Max DD
v6	<= 1.60	63	49	77.78%	+3.58u	+5.68%	1.035x	4.30%
v11 CatBoost	<= 1.60	65	53	81.54%	+7.58u	+11.66%	1.078x	3.33%
v11 neural	<= 1.60	53	43	81.13%	+4.75u	+8.96%	1.048x	4.62%
v11 ensemble	<= 1.60	63	51	80.95%	+6.09u	+9.67%	1.062x	3.87%

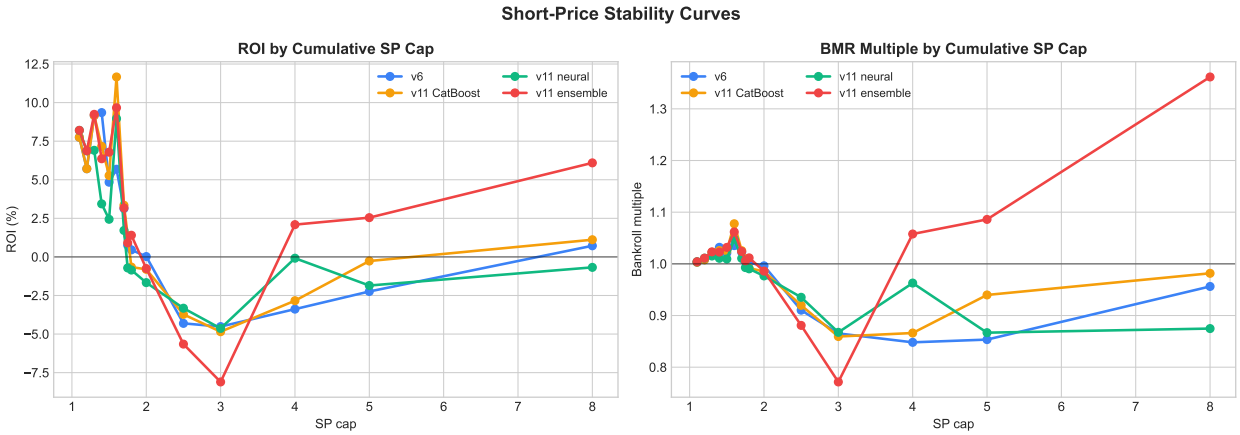
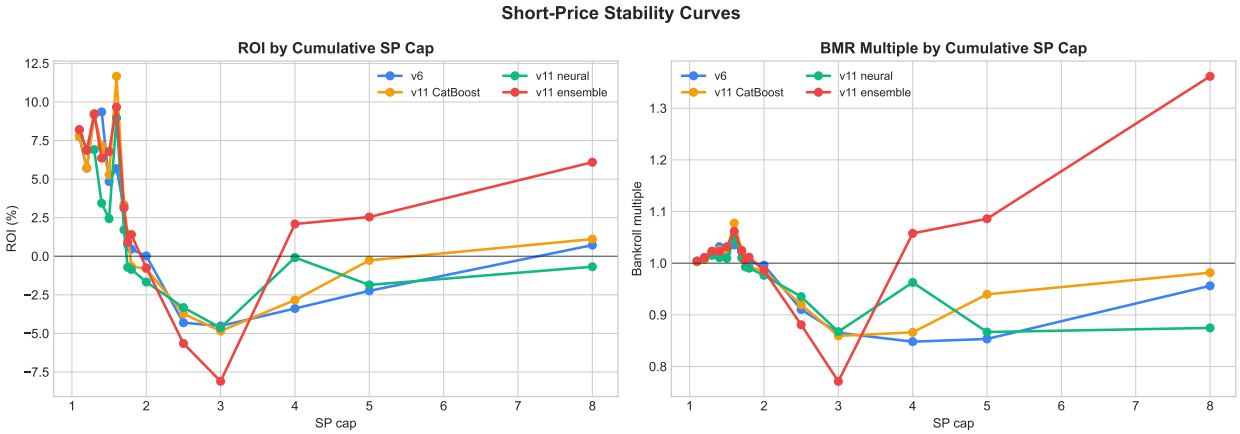


Figure 3: Short-price stability curves across cumulative SP caps. The visual break in performance beyond approximately odds of 1.60 supports a conservative deployment rule even when broader cumulative windows remain superficially positive.



8.3.2 Marginal buckets for the ensemble

Table 5: Disjoint v11 ensemble odds buckets showing where the short-price edge starts to fade.

Bucket	Bets	Wins	SR	P/L	ROI
1.41-1.50	11	8	72.73%	+0.90u	+8.18%
1.51-1.60	16	12	75.00%	+2.90u	+18.13%
1.61-1.70	17	8	47.06%	-3.55u	-20.88%
1.71-1.75	7	3	42.86%	-1.75u	-25.00%

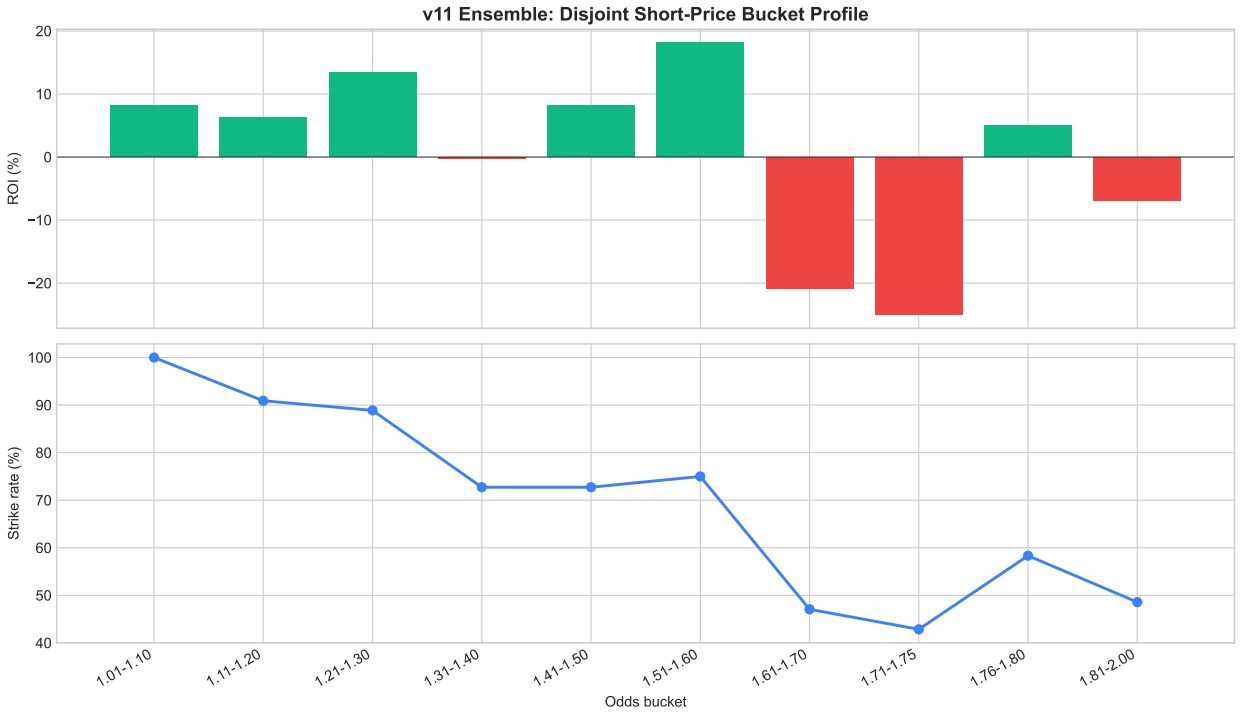
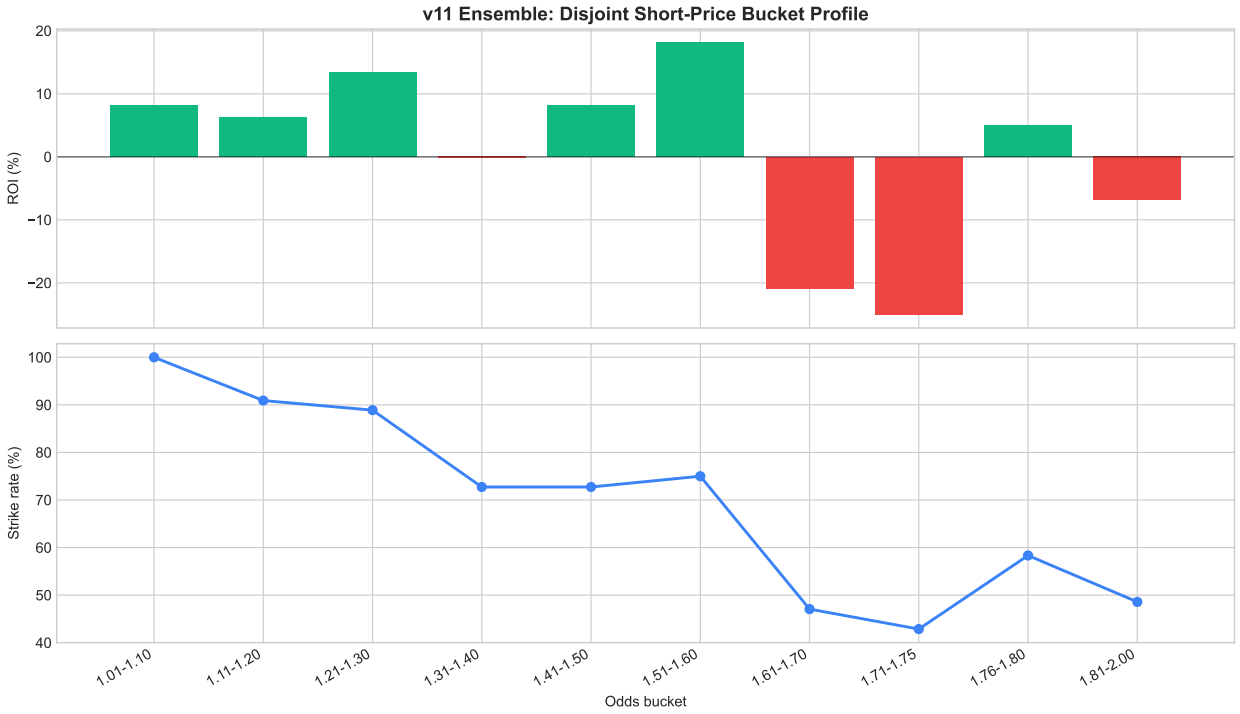


Figure 4: Disjoint v11 ensemble short-price buckets. The positive profile up to roughly odds of 1.60 and the abrupt deterioration above that range illustrate why cumulative caps should be interpreted alongside marginal bucket quality.



This pattern is more informative than a single aggregate ROI number. The cumulative `<= 1.70` and `<= 1.75` figures remain slightly positive only because the lower buckets carry them. The *marginal* buckets above about odds of 1.60 are meaningfully weaker.

That is a strong argument for stable rule design:

- train the model broadly
- deploy the short-price rule conservatively
- avoid relying on noisy last-tick odds changes to manufacture value

9 Discussion

9.1 Why v11 Works

The `v11` gain is not the result of a single magical feature. It emerges from three complementary changes.

9.1.1 1. Order information is preserved

`v6` treats recent form as a bag of summary statistics. `v11` preserves the sequence. That allows the model to distinguish trajectories such as "improving sharply" from "declining sharply" even if their averages are similar.

9.1.2 2. Race context is modeled explicitly

The cross-runner attention block allows each runner's score to depend on the rest of the field. This is much closer to how racing actually works. A strong front-runner in a clean field is not the same proposition as the same dog drawn against three other pace dogs.

9.1.3 3. The ensemble corrects branch-specific errors

The `v11` neural model is the stronger pure betting model on this clean window, but the ensemble is the stronger ranking model and the stronger overall decision system. This is classic stacked generalization behavior: a meta-model extracts signal from disagreement.

9.2 Why v6 Still Matters

If this paper were only about leaderboard maximization, the story would end at `v11`. But real systems do not live on leaderboards.

`v6` still matters because it has already survived live deployment pressure. That gives it a type of credibility that purely offline challengers do not yet have. A mature production process therefore should not be:

- train new model
- see better offline score
- replace production immediately

It should be:

- keep `v6` live
- run `v11` as challenger
- compare both under matched clean windows and shadow/live conditions
- only promote when research and operations agree

9.3 Lessons from Negative Results

The failed weather branch and the mixed value-betting slices are not embarrassing. They are informative.

The weather result suggests that simply broadening the feature set does not guarantee lift. The unstable value slices suggest that probability calibration and policy selection remain separate open problems even when ranking quality improves.

That points naturally toward the next research phase: not merely a larger winner model, but a more stable *bet selection* layer on top of `v11`.

10 Limitations

This study has several practical limitations.

1. Historical reports from `v7-v9` were not all produced under the final cleaned corpus, so the chronology table is informative rather than perfectly apples-to-apples.
2. The current live odds snapshot process can be unstable close to jump, which makes high-frequency market-edge research less trustworthy than coarse odds bucket analysis.
3. Some test-window race filtering still reflects the realities of missing or imperfectly matched post-race data.
4. Flat-stake P/L is only a partial decision metric; more work is needed on calibrated policy models and bankroll-aware deployment rules.

These limitations do not invalidate the core conclusion, but they do constrain how aggressively the results should be operationalized.

11 Future Work

The most promising next step is not `v12` as "just a bigger predictor." It is `v12` as a *policy model* on top of `v11`.

That system would take as input:

- `v11` probabilities
- model disagreement features
- rank margin over the second pick
- coarse odds buckets rather than noisy exact odds
- race and field context

and would learn a more stable decision boundary for bet/no-bet selection.

Other high-value directions include:

- adding a third diverse base model such as an `XGBoost` ranker
- explicit post-hoc calibration by odds bucket or track-distance bucket
- meeting-state or track-bias adjustments from earlier races on the same card

The common theme is stability. The future edge is unlikely to come from chasing the final 30 seconds of market movement. It is more likely to come from combining a strong base probability model with robust policy design.

12 Conclusion

This project began with a practical deployment problem and evolved into a serious applied research program in race-relative forecasting. The central lesson is that model sophistication only matters after data protocol is disciplined.

Within that disciplined setting, the results are strong:

- `v6` is a legitimate production baseline, not a disposable early model
- `v11` ensemble is the strongest research model built so far
- the most robust short-price edge appears to live below about odds of 1.60

From a research perspective, the project demonstrates that leak-free temporal evaluation, structured race modeling, and stacked ensembling can materially improve both ranking quality and betting outcomes in a difficult small-field domain.

From a hiring perspective, it shows something equally important: the ability to take ownership of the entire loop from messy data and failed ablations to clean experimental design, model architecture, evaluation, and production judgment.

13 Reproducibility Notes

The manuscript is backed by repository code rather than retrospective prose:

- `catboostResearch/shared/train_v6_catboost.py`
- `ml/models/v11/data.py`
- `ml/models/v11/neural.py`
- `ml/models/v11/catboost_model.py`

- `ml/models/v11/ensemble.py`
- `ml/models/v11/metrics.py`
- `scripts/analyze_short_price_buckets.py`

The paper is therefore not only descriptive; it is meant to stay synchronized with the actual research codebase.

References

Dorogush, Anna Veronika, Vasily Ershov, and Andrey Gulin. 2018. "CatBoost: Unbiased Boosting with Categorical Features." *arXiv Preprint arXiv:1810.11363*.

Hochreiter, Sepp, and Jurgen Schmidhuber. 1997. "Long Short-Term Memory." *Neural Computation* 9 (8): 1735–80. https://doi.org/10.1162/neco.1997.9.8.1735.

Kelly, John L. 1956. "A New Interpretation of Information Rate." *Bell System Technical Journal* 35 (4): 917–26. https://doi.org/10.1002/j.1538-7305.1956.tb03809.x.

Vaswani, Ashish, Noam Shazeer, Niki Parmar, et al. 2017. "Attention Is All You Need." *Advances in Neural Information Processing Systems*.

Wolpert, David H. 1992. "Stacked Generalization." *Neural Networks* 5 (2): 241–59. https://doi.org/10.1016/S0893-6080(05)80023-1.

Zaheer, Manzil, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander J. Smola. 2017. "Deep Sets." *Advances in Neural Information Processing Systems*.