
A Leakage-Safe Framework for Formula One Race-Winner Prediction

Dadi Ishimwe

Amakuru Motorsport, Kigali, Rwanda

16 October 2026

ABSTRACT

A useful race-winner forecast is a probability distribution, not merely a selected driver. Formula One prediction is difficult because the number of competitors is small, observations are clustered by season and constructor, sporting regulations and competitive order change over time, and race outcomes are exposed to qualifying, reliability, weather, strategy and stochastic incidents. This paper specifies a reproducible, leakage-safe research programme for estimating pre-race driver-win probabilities. It defines a race-level target and forecast cutoff; develops a transparent grid-and-form multinomial-logit baseline; compares it with hierarchical ability and ranking models and calibrated tree ensembles; and proposes chronological evaluation using proper scoring rules, calibration diagnostics and season-block uncertainty. The literature includes Bayesian separation of driver and constructor effects, qualifying-centered prediction studies, and machine-learning comparisons, but published results do not establish a transferable accuracy level for a new site or era. Accordingly, this paper reports no new empirical accuracy claim. Its accompanying TypeScript scoring primitive requires coefficients estimated on earlier seasons and has no unvalidated defaults. The proposed use for Amakuru Motorsport is explanatory editorial context, with an explicit Rwanda lens, not betting advice or a promise of a correct winner.

Keywords Formula One · race winner prediction · probability calibration · chronological validation · Rwanda

Author profile: Dadi Ishimwe · blog.dadishimwe.com

1 Introduction

A Formula One Grand Prix has a compact field and a clear headline target—one official winner—but is not a simple classification problem. Qualifying establishes a strong, observable ordering; constructor performance and driver ability are entangled; reliability and race incidents introduce substantial variance; and performance relationships can shift between technical eras. A responsible system therefore needs to estimate uncertainty, respect when each feature becomes knowable, and demonstrate performance on future seasons rather than on a random sample of past race entries.

This paper sets out a technically reproducible framework for the Amakuru Motorsport site. It has three deliberate boundaries. First, the principal forecast is made after qualifying and steward penalties have determined the adjusted starting grid, but before the race begins. Second, probabilities are evaluated at the race level, with every entrant in the event normalized together. Third, no coefficients or accuracy claims are presented as trained or validated: the current site utility is a scoring foundation that must be fitted and tested before it can produce editorial forecasts.

The distinction matters. A model can select the winner often enough to sound plausible while assigning very poor probabilities to the field. Conversely, a calibrated distribution can communicate that the grid favorite is favored but far from certain. In an editorial product, transparent uncertainty is more useful than a confident-sounding tip.

2 Scope, target, and information cutoff

2.1 Prediction unit and target

Let r index a Grand Prix and d an eligible driver on its final adjusted grid E_r . The target is the official race winner in the final published classification: $y_r \in E_r$. The event is the Grand Prix, not a sprint. Sprint results can inform lagged form for later rounds, but sprint points are not a second winner target in this paper. If an event is cancelled, abandoned without an official winner, or has an unusable result record, its handling must be specified before fitting; it must not be quietly converted into a routine loss.

The unit of evaluation is one race. Driver rows within a race are not independent observations: exactly one entrant is the winner. Any training objective and any train/test split must preserve this group structure.

2.2 Forecast timing

The benchmark cutoff is the last information timestamp strictly before the race starts, after qualifying results, penalties and the final grid are confirmed. That setting permits adjusted grid position, qualifying measures and a genuine pre-race weather forecast. It excludes any race-lap timing, pit-stop information, in-race weather observations, final classification, fastest lap, or post-event explanation. A separate pre-qualifying model would be a different task with a different feature set and evaluation.

This timestamp is the operational meaning of the pre-race information cutoff (pre-race cutoff). A strict target leakage (data leakage) audit is essential: even a perfectly implemented classifier is invalid if its rolling form accidentally includes the target event, if a final grid is reconstructed using a later correction unavailable at forecast time, or if observed race weather is substituted for the forecast that was actually available.

2.3 Data and provenance

The current Amakuru charts use the Jolpica-F1 API, an Ergast-compatible public data feed, for schedule, standings, race classifications, sprint results and lap-position records. The API is a mutable service, not a frozen research archive. A training release should therefore preserve a dated, versioned snapshot of every response, record request URLs and retrieval times, and document corrections. A six-hour web cache is a presentation policy, not a reproducibility

strategy. The FastF1 documentation describes additional timing and telemetry data access, but availability, licence conditions and historical coverage must be checked for each proposed use rather than assumed from an endpoint name (Jolpica maintainers 2026; FastF1 maintainers 2025).

| Source layer | Candidate variables | Availability rule | |—|—|—| | Event and circuit | Circuit identifier, event order, race distance, layout and regulation-era tag | Known before the event; version changes and data definitions recorded | | Qualifying and grid | Adjusted grid, qualifying position, time gaps and teammate-relative pace | Use the final information actually published before race start | | Driver history | Prior starts, finishes, qualifying gaps, reliability and rolling performance | Strictly earlier events only; define window and shrinkage in advance | | Constructor history | Prior team pace and reliability, including the entrant’s constructor at the time | Strictly earlier events only; account for driver/constructor switches | | Context | Temperature and rain forecast, sprint weekend, known penalties or calendar context | Timestamped forecast/announcement available by the cutoff |

Feature family	Example	Leakage-safe construction
Grid	Adjusted start position	After qualifying and published penalties; before lights-out
Driver form	Prior qualifying / finish residual	Rolling or decayed history ending at the previous event
Constructor form	Prior team performance	Prior events only, with era-aware shrinkage
Weather	Rain probability	Archived forecast issued before the cutoff, not observed race weather

Table 1: Feature families and their information cutoffs for the post-qualifying benchmark.

2.4 Feature engineering

A transparent first feature vector for entrant d in race r is

$$x_{r,d} = (1, g_{r,d} - 1, z_{D,r,d}, z_{C,r,d}, z_{T,r,d}) \quad (1)$$

, where g is adjusted grid position; z_D is a standardized, lagged driver-form estimate; z_C is lagged constructor form; and z_T is the driver’s past performance relative to the contemporary teammate. Standardization parameters must be learned on the training portion only. Rookie drivers need a declared prior or a missing-history treatment; missing values must not be replaced with a target-derived statistic.

Form can be estimated from several historical outcomes, but it must be defined before looking at test-season results. For example, a decayed average of pre-race residual performance gives more weight to recent events while retaining older evidence. Constructor changes, driver transfers, regulation-era transitions and new circuits are potential distribution shifts, not permission to silently pool incompatible observations. Partial pooling can reduce unstable estimates, but assumptions and prior sensitivity should be reported.

3 Candidate models

3.1 A transparent multinomial-logit baseline

For coefficients β and a positive temperature τ , assign a linear score

$$\eta_{r,d} = \beta_0 - \beta_g(g_{r,d} - 1) + \beta_D z_{D,r,d} + \beta_C z_{C,r,d} + \beta_T z_{T,r,d} \quad (2)$$

. The winner distribution is the within-race softmax:

$$p_{r,d} = \frac{\exp(\frac{\eta_{r,d}}{\tau})}{\sum_{j \in E_r} \exp(\frac{\eta_{r,j}}{\tau})} \quad (3)$$

. This model is intentionally small. The sign convention makes a positive β_g penalize a worse grid slot; estimated signs should still be checked rather than imposed by storytelling. All entrant probabilities sum to one within each event. Temperature scaling can adjust sharpness, but τ must be selected from earlier out-of-fold predictions, never tuned on the final test season. A one-hot “pole sitter wins” rule can be reported as a top-one ranking baseline; it is not a useful probability baseline because it assigns zero probability to every other entrant and yields infinite log loss whenever the pole sitter loses.

A grouped negative log-likelihood objective is

$$L(\beta) = - \sum_r \log(p_{r,y_r}) + \frac{\lambda}{2} \|\beta_1\|_2^2 \quad (4)$$

. The intercept can be left unpenalized. The regularization strength λ is selected within the historical training window. With the temperature fixed, the gradient of the unregularized race term is $(\frac{1}{\tau})X_r^T(p_r - e_{y_r})$, where e_{y_r} is the one-hot winner vector. This makes the baseline auditable and straightforward to reproduce.

```
import numpy as np
```

```
def race_probabilities(X, beta, temperature=1.0):
    """X has one row per entrant; rows belong to one race only."""
    if temperature <= 0 or not np.isfinite(temperature):
        raise ValueError("temperature must be positive and finite")
    logits = X @ beta
    scaled = (logits - logits.max()) / temperature # stable softmax
    weights = np.exp(scaled)
    return weights / weights.sum()
```

3.2 Hierarchical driver–constructor ability

A richer Bayesian model can separate persistent driver and constructor contributions. Let a_d denote a driver effect and c_t a constructor effect. A simple score is $\eta_{r,d} = \beta^T x_{r,d} + a_d + c_{t(r,d)}$, with $a_d \sim \mathcal{N}(0, \sigma_D^2)$ and $c_t \sim \mathcal{N}(0, \sigma_C^2)$. A centered or otherwise identified parameterization is required because driver and constructor quality are only partially separable when a driver remains with one team. New drivers and teams can be partially pooled toward the population distribution instead of being assigned an extreme estimate from little data.

Van Kesteren and Bergkamp demonstrate the relevance of Bayesian modeling to disentangling driver skill and constructor advantage in Formula One results (Kesteren and Bergkamp 2023). Their work motivates an ability model; it does not, by itself, validate the feature set, cutoff, or

forecast accuracy proposed here. The likelihood must match the target. A categorical winner likelihood directly models the race winner; a finish-order likelihood models more information but must also account for retirement and official classification.

3.3 Ranking and machine-learning alternatives

A Plackett–Luce ranking model (Plackett–Luce) model assigns a probability to a complete order by selecting the first finisher in proportion to positive worth, then repeating among the remaining entrants. Pairwise comparisons can also estimate the probability that one driver finishes ahead of another. These approaches use more of each classification than a winner-only target and may be useful for joint finishing-order research. However, retirements, lapped cars and classification rules complicate the assumption that all finish positions arise from a single latent ability ordering. If the final product needs finish probabilities, a separate retirement or completion component should be evaluated, not hidden inside a winner score.

Random forests and gradient-boosted trees can capture nonlinearities and interactions among grid, circuit, form and conditions. They are worthwhile challengers, not automatic upgrades: small samples, season clustering and temporal drift make unconstrained flexibility dangerous. Feature attribution methods such as SHAP can help describe a fitted model, but attribution is not causal explanation and does not establish calibration. Existing F1 prediction theses and comparative studies explore machine-learning winner, championship, podium or qualifying-performance questions; differences in target, forecast timing, features and split design mean their reported results should not be compared as if they were one benchmark (Sicoie 2022; Weissbock and Mills 2025; Yathishkumar and Likhita 2025; Haber et al. 2025).

4 Training and evaluation protocol

4.1 Chronological backtest

Use chronological expanding-window evaluation (expanding-window evaluation): train only on seasons earlier than a held-out season, tune hyperparameters using earlier chronological folds, and evaluate on the next season. Repeat over several seasons where data coverage permits. Randomly splitting race-driver rows mixes adjacent events and often places a future season beside past rows from the same drivers and constructors; it is not the primary generalization test. If a feature has a publication timestamp, construct it as an as-of snapshot at the forecast cutoff. Fit scalars, imputation, shrinkage choices and calibration mappings inside the training folds.

The training target should contain one winner per completed Grand Prix. Each event’s feature matrix remains grouped. Historical rules, event formats, driver lineups and constructor identities should be versioned. Report the number of seasons, races and entrants, exclusions, missing-data rates and any change in API coverage. If a test season represents a new technical era, that is a substantive test of domain shift, not a nuisance to be erased.

4.2 Metrics

A useful evaluation reports both ranking and probability quality. Top-one accuracy measures whether $\arg\max_d p_{r,d}$ equals the winner, but discards uncertainty. The multiclass log loss is $-\frac{1}{R} \sum_r \log(p_{r,y_r})$. The multiclass Brier score is $\frac{1}{R} \sum_r \sum_{d \in E_r} (p_{r,d} - 1[d = y_r])^2$. Log loss

and Brier score are proper scoring rules: in expectation they reward honest probabilities, not merely a correct label. Also report calibration plots or reliability summaries, top-three winner coverage, and performance by grid band and season. Define binning and minimum sample sizes before presenting an expected calibration error; a single summary must not substitute for the calibration plot.

| Question | Primary measure | Interpretation | |—|—|—| | Did the selected driver win? | Top-one accuracy | Ranking only; does not assess probability sharpness | | Were probabilities useful? | Log loss and multiclass Brier | Penalize confident errors and reward honest distributions | | Did stated confidence mean what it claimed? | Reliability plot / calibration slope | Compare predicted win-probability bands with observed frequencies | | Did the model find contenders? | Top-three coverage | Whether the winner appears among the three largest probabilities | | Did results vary by context? | Season and grid-band slices | Diagnoses shift; small strata need uncertainty intervals |

Compare against at least: (i) a smoothed historical win-rate prior by grid position, (ii) a grid-only grouped softmax, and (iii) a simple prior-season constructor/driver form model. All probabilistic baselines must assign nonzero mass to every entrant. Select hyperparameters and calibrators using only the training history. Use season-level or season-block resampling for uncertainty intervals because races in one year share teams, regulations and calendar context. Report raw event counts with each interval; a narrow interval produced by treating every driver row as independent is misleading.

4.3 Reproducible feature snapshots

A feature pipeline should receive an explicit target race and cutoff timestamp. The following sketch emphasizes the boundary: the history table must contain event timestamps and race identifiers, and the `as_of` filter precedes all rolling statistics.

```
import pandas as pd

def prior_history(history, race_id, cutoff):
    past = history.loc[
        (history["race_id"] != race_id)
        & (history["available_at"] < cutoff)
    ].copy()
    return past.sort_values(["available_at", "race_id"])

# Fit scaling and missing-value policy on training seasons only.
# Persist the as-of feature snapshot with the model and forecast record.
```

The availability timestamp is as important as the event date: a corrected result published after the cutoff is not a valid historical feature unless the system can reconstruct the earlier version. Forecast records should preserve input snapshot, model version, coefficient version, cutoff timestamp and the full probability vector. This enables audits without retroactively rewriting what the site said before the race.

5 Amakuru implementation boundary

The accompanying `lib/f1-prediction.ts` implements only the multinomial-logit score and stable softmax defined above. It requires caller-supplied coefficients, standardized lagged features and adjusted grid positions; it rejects missing/non-finite inputs and duplicate driver

IDs. It intentionally contains no “best guess” weights. The next research stage must create a dated training snapshot, implement the grouped objective, run the chronological backtest, calibrate only from earlier seasons, and publish the baseline comparisons before any probabilities are displayed on the site. Until then, the page should describe this as a research foundation, not a validated predictor.

The editorial use is to help a reader understand why a pre-race favorite is favored and how uncertain that position remains. For Rwanda, the useful differentiator is context: explain qualifying, circuit characteristics, the route from regional motorsport to a future Grand Prix, and the meaning of each probability in plain language. When a Kigali event becomes a real scheduled event, its circuit and local context can enter a documented model update. The model should not manufacture predictions about a future host race without event-specific data. No betting odds, prize mechanics, fantasy leaderboard, account system or financial recommendation is required to offer an informative race preview.

The site remains a personal, non-commercial editorial project. Jolpica and any additional data service must be used within its current terms; attribution and retention should be reviewed before adding telemetry or weather feeds. The chart cache is a practical availability feature, not evidence that an API dataset is permanently available. The model itself should be versioned separately from the editorial prose, and every displayed probability should show its cutoff and evaluation status.

6 Limitations and research agenda

First, the sample is small relative to many generic machine-learning benchmarks: one target winner exists per race, and major regulation changes alter competitive structure. Second, the observed winner is the result of a coupled driver–constructor system with strategic and reliability variation. Third, the public data source is mutable and may omit historical forecast snapshots, incident context or the exact time a correction became available. Fourth, qualifying and grid position are powerful but their predictive value is conditional on era, circuit and the race’s stochastic conditions; an apparently strong result in one split may not transfer. Finally, a well-calibrated model can still be wrong on an individual Sunday.

The recommended empirical sequence is deliberately modest: freeze the data definition and cutoff; construct a dated as-of dataset; evaluate the smoothed grid baseline; fit the grouped softmax; add lagged driver/constructor features; test hierarchical and tree challengers; calibrate using past folds; and report chronological out-of-sample scores with season-block uncertainty. Publish a model card covering target, cutoff, data coverage, known exclusions, calibration, drift and limitations. Retrain only after a documented update rather than silently changing coefficients after a headline result.

7 Conclusion

A credible Formula One winner model begins with a defensible information cutoff and evaluation design, not a fashionable algorithm. This paper specifies a probability-based baseline, interpretable alternatives, race-grouped losses and a chronological test programme. Prior literature motivates these methods but does not confer accuracy on an untested implementation. Amakuru’s scoring code therefore has no unearned weights; the next step is a leakage-safe historical benchmark before any probabilities enter the editorial product.

8 Glossary

calibration – probability calibration: Agreement between predicted event probabilities and their long-run observed frequencies.

DNF – Did Not Finish: A race status indicating that a driver did not complete the event; official classification rules may still rank a retired car.

expanding-window evaluation – chronological expanding-window evaluation: A backtest in which each test season is later in time than its training data, and the training history grows at each step.

data leakage – target leakage: Use of information that would not have been available at the stated prediction time.

Plackett–Luce – Plackett–Luce ranking model: A probability distribution over complete rankings formed by sequentially selecting each next-ranked item with probability proportional to its positive worth.

pre-race cutoff – pre-race information cutoff: The timestamp before the race start after which no feature may be added to a forecast.

References

Bibliography

- [1] Jolpica maintainers, “Jolpica-F1 API: Ergast-Compatible Formula One Results and Timing Data.” [Online]. Available: <https://api.jolpi.ca/ergast/f1/>
- [2] FastF1 maintainers, “FastF1 Documentation: Introduction and Features.” 2025. [Online]. Available: <https://docs.fastf1.dev/>
- [3] E.-J. van Kesteren and T. Bergkamp, “Bayesian Analysis of Formula One Race Results: Disentangling Driver Skill and Constructor Advantage,” *Journal of Quantitative Analysis in Sports*, vol. 19, no. 4, pp. 273–293, 2023, doi: [10.1515/jqas-2022-0021](https://doi.org/10.1515/jqas-2022-0021).
- [4] H. Sicoie, “Machine Learning Framework for Formula 1 Race Winner and Championship Standings Predictor,” Bachelor's thesis in Cognitive Science and Artificial Intelligence, 2022. [Online]. Available: <https://arno.uvt.nl/show.cgi?fid=157635>
- [5] J. Weissbock and S. Mills, “Evaluating the Predictive Power of Qualifying Performance in Formula One Grand Prix,” *arXiv preprint arXiv:2507.10966 [stat.AP]*, 2025, [Online]. Available: <https://arxiv.org/abs/2507.10966>
- [6] S. Yathishkumar and Likhita, “Predicting Formula 1 Podium Finish Probabilities Using Machine Learning: A Comparative Study of Pre-Race and Within-Race Feature Scenarios,” Master's thesis in Data Analytics, 2025. [Online]. Available: <https://norma.ncirl.ie/9904/>
- [7] E. E. Haber, E. Sawaya, M. Attieh, A. Tannous, W. Ghazaly, and M. Owayjan, “Formula 1 Race Winner Prediction Using Random Forest and SHAP Analysis,” in *2025 International Conference on Control, Automation, and Instrumentation (IC2AI)*, 2025. doi: [10.1109/IC2AI62984.2025.10932140](https://doi.org/10.1109/IC2AI62984.2025.10932140).