

# NBA Rolling Metrics Proposal

Kaleb Coleman

November 17, 2025

## Introduction

This proposal outlines an NBA game prediction study built on rolling advanced statistics for every regular-season matchup from seasons 2020-2025. Data is retrieved via the `hoopR` package, which wraps NBA Stats API endpoints, giving consistent team box scores across seasons. I use rolling metrics to smooth the game-to-game volatility that makes single-game box scores hard to interpret and align well with how bettors and analysts track recent form. I focus on constructing lagged efficiency, rebounding, and turnover measures, plus matchup-specific differentials that reflect each team's recent strengths heading into a game. Early data work highlights include handling first-game NAs (no prior history) and validating that rolling windows incorporate both home and away results without leakage of the current game outcome.

## Data Overview

The modeling table contains one row per game ( $n = 6,903$ ) with the binary outcome `home_win` and only pre-game predictors: home/away rolling efficiencies, matchup differentials, and prior games played counts. This is distilled from 13,824 raw team box-score rows down to 6,903 game-level rows (home vs away combined). Seasons range from 2020 through 2025, covering every team (30), and I can extend back to 2002 to add substantially more games if desired. Raw box scores are pulled directly from the NBA Stats API via `hoopR`, giving a reproducible link to the data source while avoiding local CSV downloads. Missing values occur exclusively for first games of each season where rolling history is unavailable (Table @ref(tab:missing)); these rows can be removed or imputed so every feature reflects completed historical performance.

Key modeling questions: (1) Which rolling matchup signals correlate most with home victory odds? (2) How much predictive lift do combined rolling metrics provide compared to simple baselines? These motivate the exploratory figures and model planning below.

## Exploratory Analysis

For proposal stage I am focusing on three core matchup signals—net rating differential, effective field goal% differential, and turnover% differential—each computed on a five-game rolling basis with lagging to avoid leakage. These are intuitive levers for both inference and prediction: net rating blends offense/defense, eFG% captures shot quality, and TOV% reflects possession security. Early smoothers show monotonic relationships with win odds, making them good candidates for parsimonious models and feature importance checks. The raw box scores (points, fgm/fga, 3pm/3pa, fta/ftm, rebounds, turnovers, assists) come straight from `hoopR::load_nba_team_box()`, so I can demonstrate end-to-end cleaning before deriving rolling features. The 5-game window is a baseline

choice; I can swap to 3- or 10-game windows in the same pipeline if the model fit suggests a better horizon.

## Modeling Plan

Inferential model: a logistic regression on the three matchup differentials (net rating, eFG%, TOV%) will supply coefficient estimates, odds ratios, and confidence intervals to quantify each pre-game factor's link to home-win odds. Predictive model: I will benchmark penalized logistic options (ridge and elastic net to handle many correlated predictors) against tree-based learners (random forest, XGBoost) using cross-validation; accuracy/AUC and calibration will guide model choice. Final analysis will incorporate cross-validation, feature scaling where needed, and potential interaction terms (e.g., net rating  $\times$  TOV%) if they materially improve out-of-sample fit. Cleaned modeling objects (`games_combined`, training/test splits, fitted models) will be saved to an `.RData` file to ensure reproducibility. Additional goals include documenting the treatment of early-season NA rows, comparing model lift over the baseline, and preparing a concise narrative for the final flash talk.

## Goals & Timeline

1. **Week 1 (current)**: finalize data pipeline, verify rolling features, produce summary tables and initial plots.
2. **Week 2**: complete logistic inferential model with confidence intervals; tune predictive models with cross-validation.
3. **Week 3**: consolidate results, create presentation-ready visuals, export `.RData` with cleaned tables and fitted objects.

Deliverables include the proposal PDF, cleaned data bundle, annotated modeling scripts, and draft presentation slides summarizing key findings.

## References

- hoopR package: Gill, K. (2023). *hoopR: Access Men's Basketball Data* (R package version 1.8.0).
- NBA Stats API documentation: <https://stats.nba.com> (accessed November 17, 2025).

## Tables

Table 1: Game coverage and date span for the modeling table.

Total games	Seasons covered	Teams	Median games/season	Date range
6903	6	30	1174	2019-10-22 to 2025-04-11

Table 2: Row/column counts for raw and modeling data (expandable back to 2002).

Dataset	Rows	Columns	Seasons
Raw team box (team_box_raw)	13,824	57	2020-2025
Modeling table (games_combined)	6,903	44	2020-2025

Table 3: Preview of available variables (many more predictors remain for lasso/ridge/elastic net).

Example columns
game_id
season
game_date
home_team_id
drb_roll5_diff
drb_roll5_ratio
tov_roll5_diff
tov_roll5_ratio

Table 4: Rolling features have 89–95 missing rows (first games only; caused by lagging). These rows will be dropped or imputed before modeling.

Min missing	Max missing
89	95

## Raw Data Snapshot

```
raw_meta <- tibble::tibble(
  Rows = nrow(team_box_raw),
  Columns = ncol(team_box_raw),
  `First 10 cols` = paste(head(names(team_box_raw), 10), collapse = ", "),
  `Last 10 cols` = paste(tail(names(team_box_raw), 10), collapse = ", ")
)

raw_preview <- team_box_raw %>%
  select(game_id, season, game_date, team_abbreviation, team_home_away,
         team_score, field_goals_made, field_goals_attempted,
         three_point_field_goals_made, three_point_field_goals_attempted,
         free_throws_made, free_throws_attempted,
         offensive_rebounds, defensive_rebounds, turnovers) %>%
  head()

raw_meta

## # A tibble: 1 x 4
##   Rows Columns `First 10 cols`                `Last 10 cols`
##   <int>   <int> <chr>                                <chr>
## 1 13824     57 game_id, season, season_type, game_date, game_da~ opponent_team~

raw_preview

## # A tibble: 6 x 15
##   game_id season game_date team_abbreviation team_home_away team_score
##   <int>   <int> <date>      <chr>                <chr>          <int>
## 1 401224790  2020 2020-08-14 PHI                    away            134
## 2 401224790  2020 2020-08-14 HOU                    home             96
## 3 401224792  2020 2020-08-14 OKC                    away            103
## 4 401224792  2020 2020-08-14 LAC                    home            107
## 5 401224791  2020 2020-08-14 MIA                    away             92
## 6 401224791  2020 2020-08-14 IND                    home            109
## # i 9 more variables: field_goals_made <int>, field_goals_attempted <int>,
## #   three_point_field_goals_made <int>,
## #   three_point_field_goals_attempted <int>, free_throws_made <int>,
## #   free_throws_attempted <int>, offensive_rebounds <int>,
## #   defensive_rebounds <int>, turnovers <int>
```

*Note:* Raw team box scores are pulled directly from `hoopR::load_nba_team_box()`; the full table has many more columns than shown in the preview, and the rolling feature NA counts arise only because lagged windows exclude each team's first game of a season.

## Figures

These three visuals correspond to the features emphasized in the modeling plan and will be reused in both the inferential write-up and the predictive section.

**Plot recipe:** jitter spreads overlapping 0/1 outcomes (points at the top are `home_win = 1`, bottom are `home_win = 0`, nudged vertically for visibility); `geom_smooth` with binomial glm traces win-probability curves with a confidence band and keeps predictions between 0 and 1; `scale_y_continuous(labels = percent_format())` formats the y-axis as intuitive percentages. Using `method = "lm"` would force a straight line, so binomial glm is preferred here.

### Matchup Net Rating Differential

```
games_combined %>%
  ggplot(aes(x = matchup_net_roll5, y = as.numeric(home_win))) +
  geom_jitter(height = 0.03, alpha = 0.2) +
  geom_smooth(method = "glm", method.args = list(family = binomial()), se = TRUE, color = "steelblue") +
  scale_y_continuous(labels = percent_format(accuracy = 1)) +
  labs(title = "Home Win Probability vs Rolling Net Rating Differential",
       x = "Home Net Rating (5) - Away Net Rating (5)",
       y = "Home Win Probability")
```

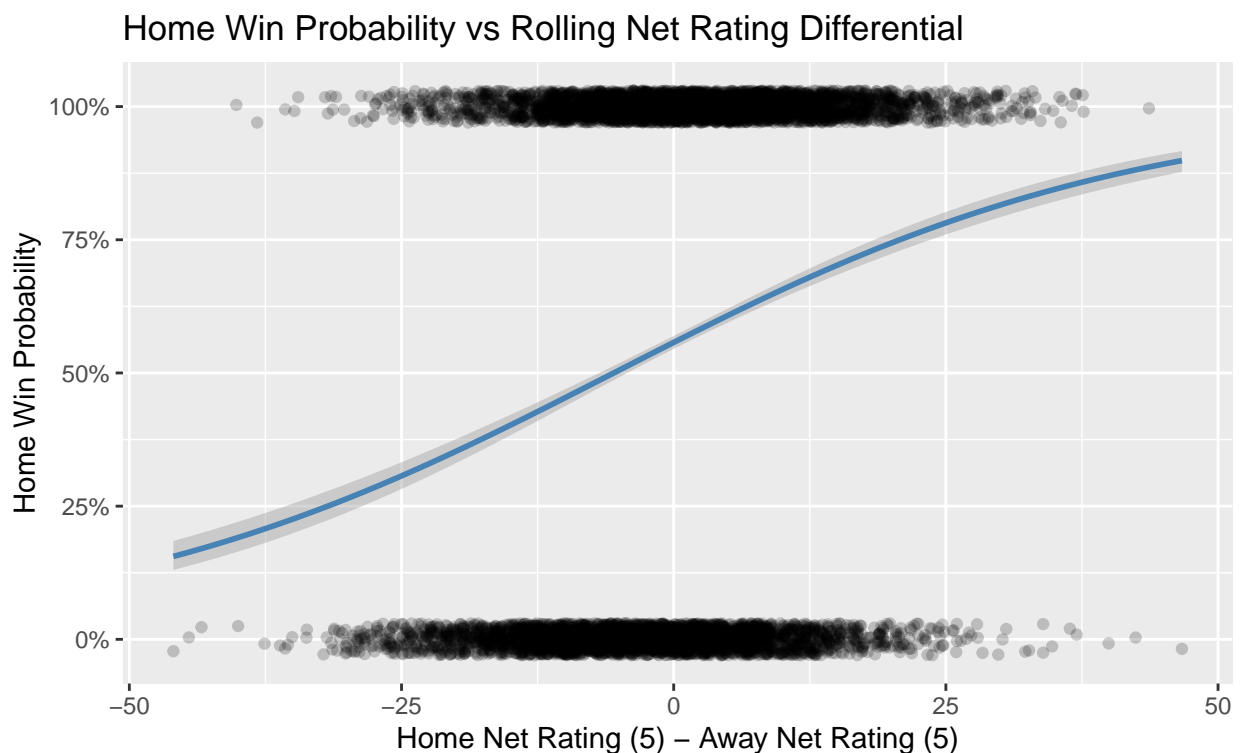


Figure 1: Home win probability versus net rating differential using logistic smoothing.

*Note:* The logistic smoother highlights how larger recent net rating gaps strongly increase the modeled chance of a home victory; the S-curve supports using this variable as a primary inferential predictor.

## eFG% Differential

```
games_combined %>%
  mutate(efg_diff = home_efg_roll5 - away_efg_roll5) %>%
  ggplot(aes(x = efg_diff, y = as.numeric(home_win))) +
  geom_jitter(height = 0.03, alpha = 0.2, color = "darkorange") +
  geom_smooth(method = "glm", method.args = list(family = binomial()), se = TRUE, color = "firebrick") +
  scale_y_continuous(labels = percent_format(accuracy = 1)) +
  labs(title = "Win Probability vs eFG% Differential",
       x = "Home eFG% (5) - Away eFG% (5)",
       y = "Home Win Probability")
```

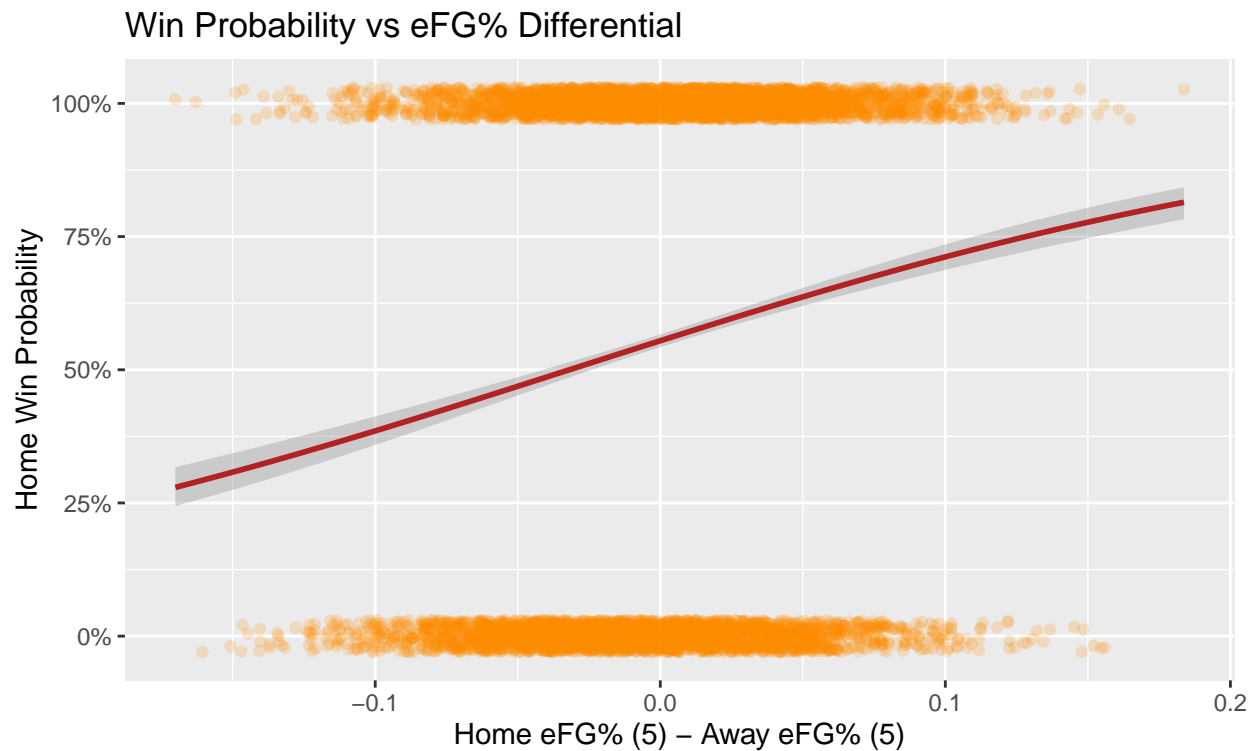


Figure 2: Home win probability versus 5-game rolling eFG% differential.

*Note:* Shot quality advantages (higher eFG%) translate into steadily rising win odds, providing a clean interpretation for the inferential model and a strong signal for the predictive model; the logistic smoother makes the monotone pattern visible despite noisy point-level jitter.

## Turnover% Differential

```
games_combined %>%
  mutate(tov_diff = home_tov_perc_roll5 - away_tov_perc_roll5) %>%
  ggplot(aes(x = tov_diff, y = as.numeric(home_win))) +
  geom_jitter(height = 0.03, alpha = 0.2, color = "steelblue") +
  geom_smooth(method = "glm", method.args = list(family = binomial()), se = TRUE, color = "navy") +
  scale_y_continuous(labels = percent_format(accuracy = 1)) +
  labs(title = "Win Probability vs Turnover% Differential",
       x = "Home TOV% (5) - Away TOV% (5)",
       y = "Home Win Probability")
```

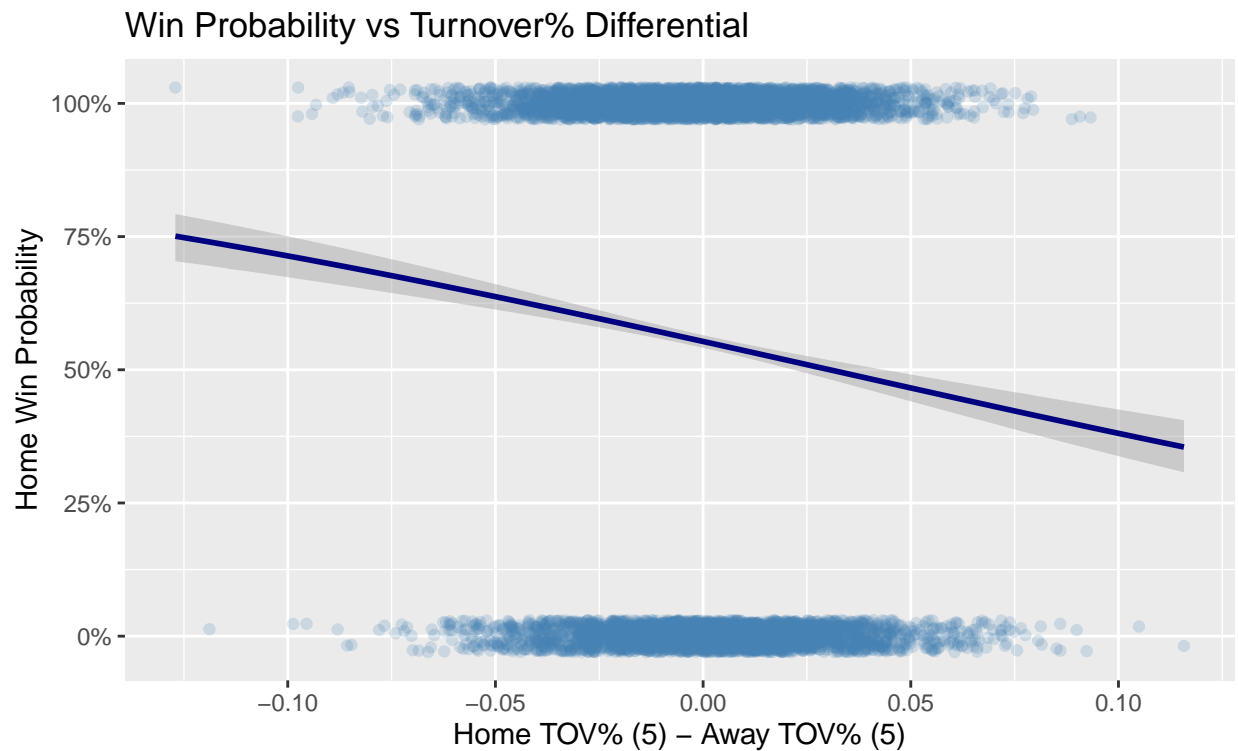


Figure 3: Home win probability versus 5-game rolling turnover percentage differential.

*Note:* Lower home turnover rates relative to the opponent drive a noticeable uptick in win probability, underlining the importance of protecting possessions; the negative slope will translate to a negative coefficient in the inferential model, while jitter prevents point pileups at 0/1 outcomes.