NBA_Win_Percentage

2025-11-13

```r
# Install Once in Console
# install.packages(c("tidyverse","janitor","lubridate","tidymodels","vip","finetune"))
# install.packages("hoopR")

library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   4.0.0     v tibble    3.3.0
## v lubridate 1.9.4     v tidyr     1.3.1
## v purrr     1.1.0
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
```

```r
library(janitor)
```

```
##
## Attaching package: 'janitor'
##
## The following objects are masked from 'package:stats':
##
##     chisq.test, fisher.test
```

```r
library(lubridate)
library(tidymodels)
```

```
## -- Attaching packages -------------------------------------- tidymodels 1.4.1 --
## v broom        1.0.10     v rsample      1.3.1
## v dials        1.4.2      v tailor       0.1.0
## v infer        1.0.9      v tune         2.0.1
## v modeldata    1.5.1      v workflows    1.3.0
## v parsnip      1.3.3      v workflowsets 1.1.1
## v recipes      1.3.1      v yardstick    1.3.2
## -- Conflicts ----------------------------------------- tidymodels_conflicts() --
## x scales::discard() masks purrr::discard()
## x dplyr::filter()   masks stats::filter()
## x recipes::fixed()  masks stringr::fixed()
## x dplyr::lag()      masks stats::lag()
## x yardstick::spec() masks readr::spec()
## x recipes::step()   masks stats::step()
```

```r
library(hoopR)
library(vip)
```

```
##
## Attaching package: 'vip'
##
## The following object is masked from 'package:utils':
##
##     vi
```

```r
set.seed(123)

df_raw <- load_nba_team_box(seasons = 2015:2024)

# Check
df_raw %>% count(season) %>% arrange(season)
```

```
## -- ESPN NBA Team Boxscores from hoopR data repository ----------- hoopR 2.1.0 --
## i Data updated: 2025-04-18 17:42:31 MDT

## # A tibble: 10 x 2
##    season     n
##     <int> <int>
##  1   2015  2624
##  2   2016  2634
##  3   2017  2618
##  4   2018  2626
##  5   2019  2628
##  6   2020  2290
##  7   2021  2242
##  8   2022  2648
##  9   2023  2642
## 10   2024  2640
```

```r
unique(df_raw$season)
```

```
##  [1] 2015 2016 2017 2018 2019 2020 2021 2022 2023 2024
```

```r
df_team_season <- df_raw %>%
  filter(season_type == 2) %>%        # 2 = Regular season
  group_by(season, team_id, team_abbreviation, team_name) %>%
  summarise(
    games = n(),
    wins  = sum(team_winner, na.rm = TRUE),    # TRUE = win
    win_pct = wins / games,

    # per-game averages of all numeric stats (box score + scores)
    across(where(is.numeric), mean, na.rm = TRUE),

    .groups = "drop"
  )
```

```
## Warning: There was 1 warning in `summarise()`.
## i In argument: `across(where(is.numeric), mean, na.rm = TRUE)`.
## i In group 1: `season = 2015`, `team_id = 1`, `team_abbreviation = "ATL"`,
##   `team_name = "Hawks"`.
## Caused by warning:
## ! The `...` argument of `across()` is deprecated as of dplyr 1.1.0.
## Supply arguments directly to `.fns` through an anonymous function instead.
##
```

```
##   # Previously
##   across(a:b, mean, na.rm = TRUE)
##
##   # Now
##   across(a:b, \(x) mean(x, na.rm = TRUE))
```

```r
df_team_season %>%
  summarise(
    n_teams  = n(),
    min_win  = min(win_pct),
    max_win  = max(win_pct),
    mean_win = mean(win_pct),
    sd_win   = sd(win_pct)
  )
```

```
## # A tibble: 1 x 5
##   n_teams min_win max_win mean_win sd_win
##     <int>   <dbl>   <dbl>    <dbl>  <dbl>
## 1     326       0       1    0.499  0.200
```

```r
#Quick Correlation for WIN PCT
num_vars <- df_team_season %>%
  select(where(is.numeric))

cor_with_win <- num_vars %>%
  summarise(across(everything(), ~ cor(.x, win_pct, use = "complete.obs"))) %>%
  pivot_longer(everything(), names_to = "variable", values_to = "cor_win_pct") %>%
  arrange(desc(abs(cor_win_pct)))
```

```
## Warning: There was 1 warning in `summarise()`.
## i In argument: `across(everything(), ~cor(.x, win_pct, use = "complete.obs"))`.
## Caused by warning in `cor()`:
## ! the standard deviation is zero
```

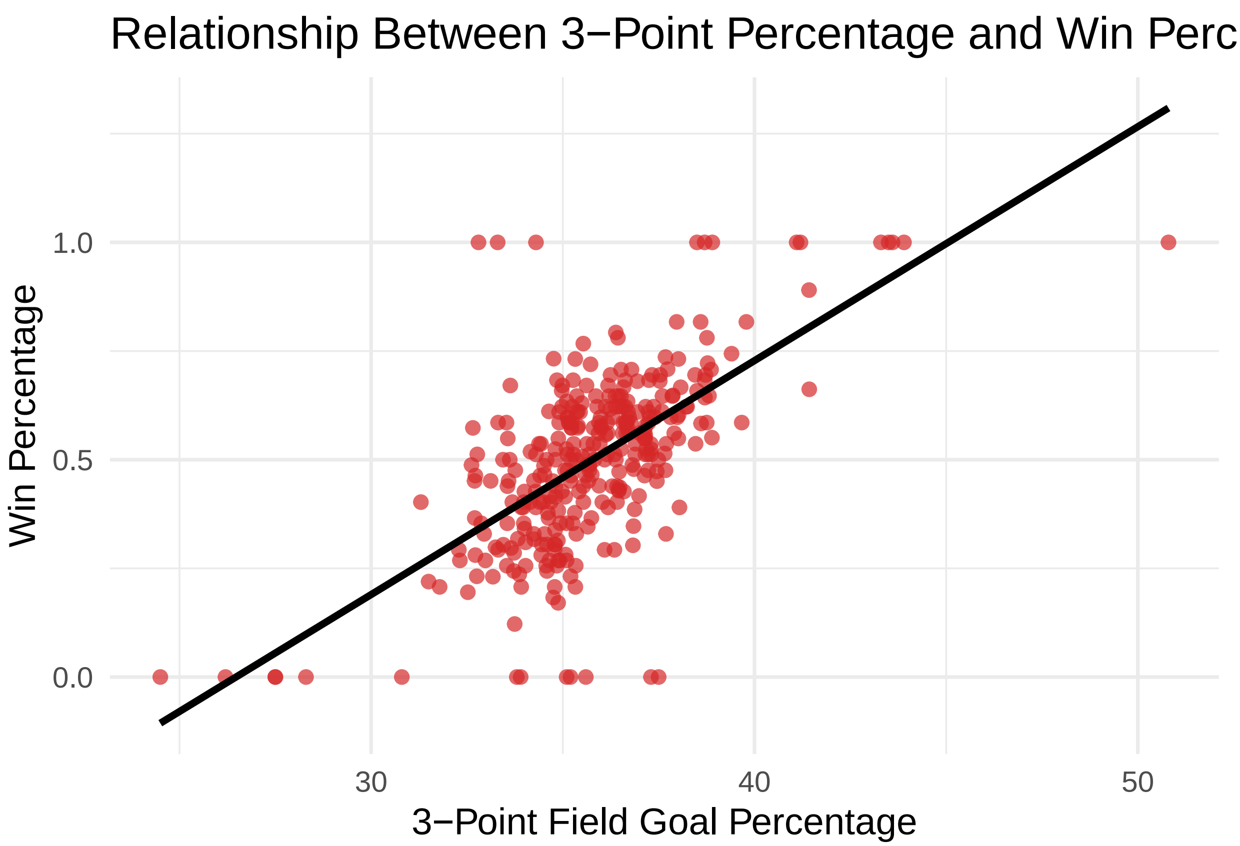
```r
head(cor_with_win, 20)
```

```
## # A tibble: 20 x 2
##    variable                          cor_win_pct
##    <chr>                                   <dbl>
##  1 win_pct                                1
##  2 three_point_field_goal_pct             0.644
##  3 wins                                   0.515
##  4 defensive_rebounds                     0.276
##  5 three_point_field_goals_made           0.265
##  6 team_score                             0.178
##  7 opponent_team_score                   -0.167
##  8 total_rebounds                         0.116
##  9 three_point_field_goals_attempted      0.114
## 10 field_goals_made                       0.112
## 11 field_goal_pct                         0.0773
## 12 technical_fouls                        0.0754
## 13 total_technical_fouls                  0.0754
## 14 turnovers                             -0.0683
## 15 total_turnovers                       -0.0682
## 16 blocks                                 0.0634
## 17 free_throw_pct                         0.0568
```

```
## 18 assists                                   0.0550
## 19 offensive_rebounds                       -0.0496
## 20 steals                                    0.0351
```

```
df_team_season %>%
  ggplot(aes(x = three_point_field_goal_pct, y = win_pct)) +
  geom_point(alpha = 0.7, color = "#d62728") +
  geom_smooth(method = "lm", se = FALSE, color = "black") +
  labs(
    title = "Relationship Between 3-Point Percentage and Win Percentage",
    x = "3-Point Field Goal Percentage",
    y = "Win Percentage"
  ) +
  theme_minimal(base_size = 14)
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



Add All Plots or Correlations here, leave RF Model at end

```
df_model <- df_team_season %>%
  # drop obvious leaks / not useful predictors
  select(
    -wins,
    -games,
  )

set.seed(123)
```

```r
nba_split <- initial_split(df_model, prop = 0.8, strata = win_pct)
nba_train <- training(nba_split)
nba_test  <- testing(nba_split)

nba_recipe <- recipe(win_pct ~ ., data = nba_train) %>%
  # Mark ID columns so they're not used as predictors
  update_role(season, team_id, team_abbreviation, team_name, new_role = "ID") %>%


  step_zv(all_predictors()) %>%
  step_impute_median(all_numeric_predictors()) %>%
  step_normalize(all_numeric_predictors())
```

```r
rf_spec <- rand_forest(
  mtry  = tune(),
  min_n = tune(),
  trees = 1000
) %>%
  set_mode("regression") %>%
  set_engine("ranger", importance = "impurity")

#Cross Validation
set.seed(123)
nba_folds <- vfold_cv(nba_train, v = 5, strata = win_pct)

# Workflow
rf_workflow <- workflow() %>%
  add_model(rf_spec) %>%
  add_recipe(nba_recipe)

#Small Tuning Grid
rf_grid <- grid_regular(
  mtry(range = c(5, 30)),
  min_n(range = c(2, 15)),
  levels = 5
)

#Tune
rf_tuned <- tune_grid(
  rf_workflow,
  resamples = nba_folds,
  grid = rf_grid,
  metrics = metric_set(rmse, rsq)
)
```

```
## > A | warning: ! 30 columns were requested but there were 26 predictors in the data.
##                i 26 predictors will be used.
```

```
## There were issues with some computations   A: x1There were issues with some computations   A: x3There
```

```r
collect_metrics(rf_tuned) %>%
  arrange(mean)
```

```
## # A tibble: 50 x 8
##     mtry min_n .metric .estimator  mean     n std_err .config
```

```
##    <int> <int> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
##  1    30     2 rmse    standard   0.145     5  0.0121 pre0_mod21_post0
##  2    30     8 rmse    standard   0.146     5  0.0124 pre0_mod23_post0
##  3    23     2 rmse    standard   0.146     5  0.0119 pre0_mod16_post0
##  4    30     5 rmse    standard   0.146     5  0.0124 pre0_mod22_post0
##  5    23     5 rmse    standard   0.146     5  0.0122 pre0_mod17_post0
##  6    23     8 rmse    standard   0.147     5  0.0121 pre0_mod18_post0
##  7    23    11 rmse    standard   0.147     5  0.0116 pre0_mod19_post0
##  8    30    11 rmse    standard   0.147     5  0.0120 pre0_mod24_post0
##  9    17     5 rmse    standard   0.148     5  0.0106 pre0_mod12_post0
## 10    30    15 rmse    standard   0.148     5  0.0118 pre0_mod25_post0
## # i 40 more rows
```

```r
#Pick Best and Refit
best_rf <- select_best(rf_tuned, metric = "rmse")

rf_final_wf <- finalize_workflow(rf_workflow, best_rf)

rf_fit <- fit(rf_final_wf, data = nba_train)
```

```
## Warning: ! 30 columns were requested but there were 26 predictors in the data.
## i 26 predictors will be used.
```

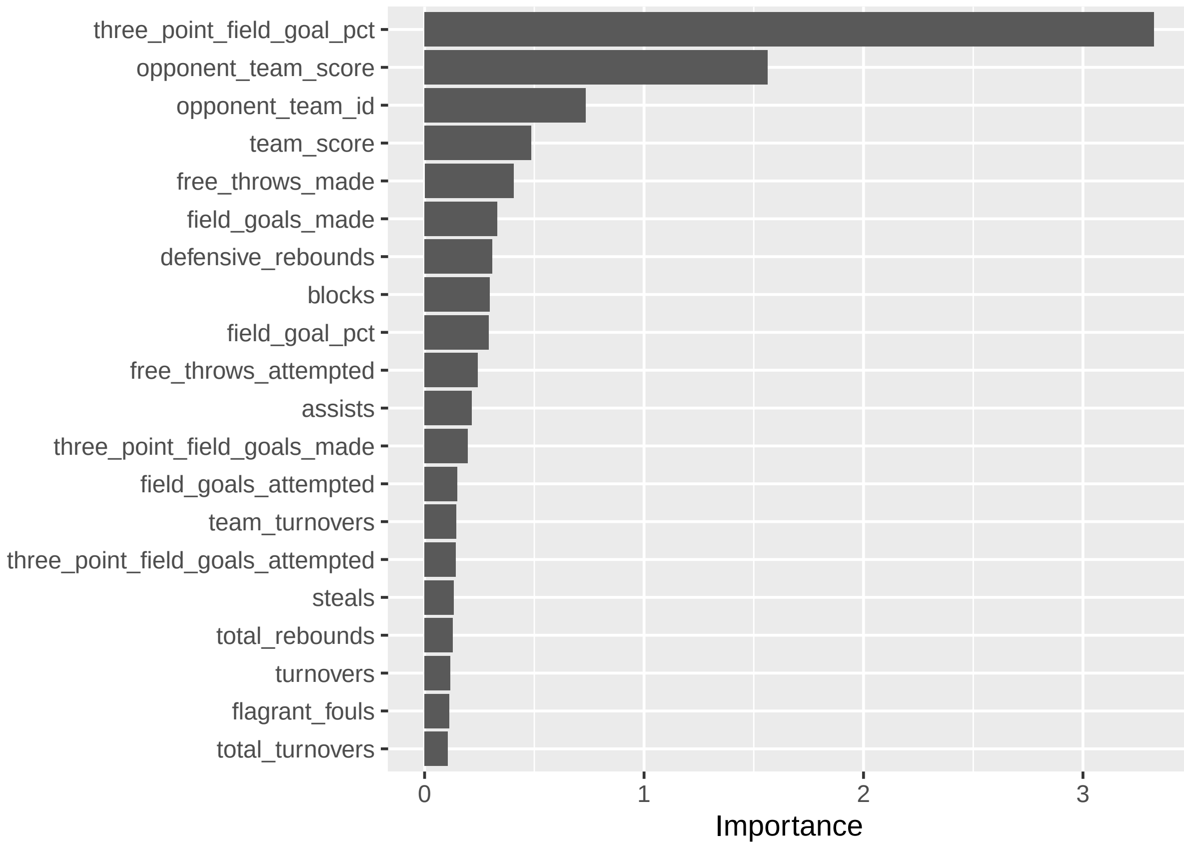
```r
rf_test_preds <- predict(rf_fit, new_data = nba_test) %>%
  bind_cols(nba_test %>% select(win_pct))

rf_test_preds %>%
  metrics(truth = win_pct, estimate = .pred)
```

```
## # A tibble: 3 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard       0.178
## 2 rsq     standard       0.322
## 3 mae     standard       0.102
```

```r
rf_fit_ranger <- rf_fit %>%
  extract_fit_parsnip() %>%
  pluck("fit")

vip(rf_fit_ranger, num_features = 20)
```




```r
var_imp <- rf_fit_ranger$variable.importance %>%
  sort(decreasing = TRUE)

head(var_imp, 30)
```

```
##        three_point_field_goal_pct               opponent_team_score
##                        3.32338798                        1.56244976
##                  opponent_team_id                        team_score
##                        0.73395888                        0.48652163
##                  free_throws_made                  field_goals_made
##                        0.40520673                        0.33047505
##                defensive_rebounds                            blocks
##                        0.30785477                        0.29801706
##                    field_goal_pct             free_throws_attempted
##                        0.29243095                        0.24257150
##                           assists      three_point_field_goals_made
##                        0.21606435                        0.19691119
##             field_goals_attempted                    team_turnovers
##                        0.14814133                        0.14494877
## three_point_field_goals_attempted                            steals
##                        0.14156217                        0.13268603
##                    total_rebounds                         turnovers
##                        0.12953774                        0.11662880
##                    flagrant_fouls                   total_turnovers
##                        0.11358668                        0.10675867
##                    free_throw_pct                offensive_rebounds
##                        0.10507568                        0.08991222
```

```
##                              fouls             total_technical_fouls
##                         0.08511494                        0.07220005
##                    technical_fouls                           game_id
##                         0.06590008                        0.05713184
```

```r
lm_spec <- linear_reg() %>%
  set_engine("lm")

lm_wf <- workflow() %>%
  add_model(lm_spec) %>%
  add_recipe(nba_recipe)

lm_fit <- fit(lm_wf, data = nba_train)

tidy(lm_fit) %>%
  arrange(desc(abs(estimate))) %>%
  head(20)
```

```
## # A tibble: 20 x 5
##    term                             estimate std.error statistic   p.value
##    <chr>                               <dbl>     <dbl>     <dbl>     <dbl>
##  1 field_goals_made                   1.33     9.57        0.139 8.90e-  1
##  2 team_score                        -0.977    9.92       -0.0985 9.22e-  1
##  3 (Intercept)                        0.496    0.00445  111.     1.94e-205
##  4 opponent_team_score               -0.423    0.0304   -13.9    1.57e- 32
##  5 three_point_field_goals_made       0.395    2.61        0.151 8.80e-  1
##  6 free_throws_made                   0.206    2.30        0.0893 9.29e-  1
##  7 fouls                              0.105    0.0235      4.47  1.23e-  5
##  8 field_goal_pct                     0.0957   0.0315      3.04  2.67e-  3
##  9 free_throws_attempted              0.0929   0.0377      2.46  1.45e-  2
## 10 free_throw_pct                     0.0712   0.0160      4.44  1.37e-  5
## 11 flagrant_fouls                     0.0697   0.0270      2.58  1.05e-  2
## 12 blocks                             0.0648   0.0186      3.48  6.06e-  4
## 13 assists                            0.0537   0.0177      3.03  2.74e-  3
## 14 steals                            -0.0468   0.0120     -3.90  1.28e-  4
## 15 field_goals_attempted              0.0368   0.0576      0.638 5.24e-  1
## 16 three_point_field_goals_attempted -0.0303   0.0525     -0.577 5.65e-  1
## 17 game_id                           -0.0225   0.0107     -2.09  3.73e-  2
## 18 offensive_rebounds                -0.0158   0.0116     -1.36  1.74e-  1
## 19 total_turnovers                   -0.0143   0.0232     -0.617 5.38e-  1
## 20 total_rebounds                    -0.0111   0.0127     -0.872 3.84e-  1
```

Summary

Interpretation of Variable Importance for Predicting Win Percentage

The random forest model identifies which team statistics are most strongly associated with overall winning percentage across all NBA seasons from 2015–2024. The most important predictor by far is three-point field goal percentage, followed by several offensive and defensive efficiency indicators such as scoring, opponent scoring, free throws made, and defensive rebounding.

Overall, the model suggests that efficient shooting—especially from three, limiting opponent scoring, and winning key possession battles (rebounds, blocks, turnovers) are the strongest predictors of a successful season.