Predicting Soccer Match Outcomes

Theyab Alkhoori, Jasmine Moskowitz, Max Treusein

2025-12-02

Introduction

Around the world, soccer is a popular sport. Millions of people enjoy using each team's data to predict match outcomes. Our project aims to predict match results of the 2017 season. We will predict the 2017 season using the statistics from the 2008 to 2016 seasons, then compare our predictions with the actual outcomes. In the end we hope to know if the home team wins, ties, or loses and well as the number of goals scored during the game. Once complete we can then compare our results with the actual results of the game.

References

We plan to execute our desired tasks by using the **European Soccer Database** from Kaggle (https://www.kaggle.com/datasets/hugomathien/soccer/data). This source includes information from more than 25,000 games, more than 10,000 players, and eleven European nations. The data contains match statistics, results, and other information all contained in csv files. We will split the data into two parts: train and test. The training data will contain data from the season 2008 to 2016. The test data will contain data from the 2017 season.

Methods

Task 1: Data Preparation

Our first task is to clean out the data from the csv files. We want to extract all the necessary data to make it easier to complete our tasks.

Task 2: Classification

Our second task is using classification methods to predict match outcomes (home win, tie, home loss)

Task 3: Regression

Our third task is to predict the number of goals scored in a match.

Results

1. If the home team wins, ties, or loses.
2. The number of goals scored during a game.

=======

We used the **European Soccer Database** from Kaggle, which includes 11 leagues, 10,000 players, and more than 25,000.

```
library(tidymodels)
```

```
## -- Attaching packages --------------------------------------- tidymodels 1.3.0 --
```

```
## v broom        1.0.8     v recipes      1.3.1
## v dials        1.4.0     v rsample      1.3.0
## v dplyr        1.1.4     v tibble       3.3.0
## v ggplot2      4.0.0     v tidyr        1.3.1
## v infer        1.0.9     v tune         1.3.0
## v modeldata    1.4.0     v workflows    1.2.0
## v parsnip      1.3.2     v workflowsets 1.1.1
## v purrr        1.1.0     v yardstick    1.3.2
```

```
## -- Conflicts ------------------------------------------ tidymodels_conflicts() --
## x purrr::discard() masks scales::discard()
## x dplyr::filter()  masks stats::filter()
## x dplyr::lag()     masks stats::lag()
## x recipes::step()  masks stats::step()
```

```
library(discrim)
```

```
##
## Attaching package: 'discrim'
```

```
## The following object is masked from 'package:dials':
##
##     smoothness
```

```
library(ISLR)
library(ggplot2)
library(dplyr)
library(readr)
```

```
##
## Attaching package: 'readr'
```

```
## The following object is masked from 'package:yardstick':
##
##     spec
```

```
## The following object is masked from 'package:scales':
##
##     col_factor
```

```
library(lubridate)
```

```
##
## Attaching package: 'lubridate'
```

```
## The following objects are masked from 'package:base':
##
##     date, intersect, setdiff, union
```

```
set.seed(445)

## Load raw data ----
match <- read.csv("Match.csv")
team <- read.csv("Team.csv")
team_attr <- read.csv("Team_Attributes.csv")
```

```r
## Team-season attributes ----
team_attr |>
  mutate(
    date = as.Date(date),
    season = year(date)
  ) |>
  filter(season >= 2010, season <= 2015) |>
  select(-id, -date) |>
  group_by(team_api_id, season) |>
  summarise(
    across(where(is.numeric), ~ mean(.x, na.rm = TRUE)),
    .groups = "drop"
  ) -> team_season_attr

team |>
  select(team_api_id, team_long_name, team_short_name) |>
  right_join(team_season_attr, by = "team_api_id") -> team_season_attr

num_cols <- sapply(team_season_attr, is.numeric)

team_season_attr[num_cols] <- lapply(team_season_attr[num_cols], function(x) {
  if (all(is.na(x))) return(x)
  x[is.na(x)] <- mean(x, na.rm = TRUE)
  x
})

## Build match-level outcome and season ----
match |>
  mutate(
    date = as.Date(date),
    season = year(date),
    home_result = case_when(
      home_team_goal > away_team_goal ~ "Win",
      home_team_goal == away_team_goal ~ "Draw",
      home_team_goal < away_team_goal ~ "Loss"
    ),
    home_result = factor(home_result, levels = c("Win", "Draw", "Loss"))
  ) |>
  filter(season >= 2010, season <= 2015) |>
  select(
    match_api_id,
    season,
    date,
    home_team_api_id,
    away_team_api_id,
    home_team_goal,
    away_team_goal,
    home_result
  ) -> matches_base

## Home-team attributes ----
team_season_attr |>
  rename(
```

```
    home_team_api_id = team_api_id
  ) |>
  rename_with(
    ~ paste0("home_", .x),
    .cols = -c(home_team_api_id, season)
  ) -> home_attr

## Away-team attributes ----
team_season_attr |>
  rename(
    away_team_api_id = team_api_id
  ) |>
  rename_with(
    ~ paste0("away_", .x),
    .cols = -c(away_team_api_id, season)
  ) -> away_attr

## Join attributes onto matches ----
matches_base |>
  left_join(home_attr, by = c("season", "home_team_api_id")) |>
  left_join(away_attr, by = c("season", "away_team_api_id")) -> soccer_matches

## Final train / test splits ----
train_matches <- soccer_matches |> filter(season <= 2014)
test_matches  <- soccer_matches |> filter(season == 2015)

head(train_matches)
```

```
##   match_api_id season       date home_team_api_id away_team_api_id
## 1       665626   2010 2010-02-03             8635             8342
## 2       665630   2010 2010-02-04             9986             9985
## 3       665634   2010 2010-02-02             8203             9993
## 4       665665   2010 2010-01-30             8342            10001
## 5       665666   2010 2010-01-17             9985             8635
## 6       665667   2010 2010-01-16             9997             9994
##   home_team_goal away_team_goal home_result home_team_long_name
## 1              3              2         Win      RSC Anderlecht
## 2              2              3        Loss  Sporting Charleroi
## 3              1              0         Win         KV Mechelen
## 4              2              1         Win      Club Brugge KV
## 5              0              4        Loss   Standard de Liège
## 6              2              1         Win   Sint-Truidense VV
##   home_team_short_name home_team_fifa_api_id home_buildUpPlaySpeed
## 1                  AND                   229                    50
## 2                  CHA                   670                    40
## 3                  MEC                110724                    65
## 4                  CLB                   231                    35
## 5                  STL                   232                    53
## 6                  STT                   680                    50
##   home_buildUpPlayDribbling home_buildUpPlayPassing home_chanceCreationPassing
## 1                  48.60451                      35                         70
## 2                  48.60451                      50                         45
## 3                  48.60451                      60                         50
## 4                  48.60451                      40                         45
```

```
## 5                  48.60451                     40                          55
## 6                  48.60451                     50                          50
##   home_chanceCreationCrossing home_chanceCreationShooting home_defencePressure
## 1                          50                          60                    70
## 2                          43                          60                    70
## 3                          40                          50                    60
## 4                          50                          45                    60
## 5                          55                          65                    70
## 6                          50                          50                    50
##   home_defenceAggression home_defenceTeamWidth away_team_long_name
## 1                     50                    70      Club Brugge KV
## 2                     70                    70   Standard de Liège
## 3                     70                    60        Beerschot AC
## 4                     70                    70        KVC Westerlo
## 5                     70                    65      RSC Anderlecht
## 6                     50                    50    Sporting Lokeren
##   away_team_short_name away_team_fifa_api_id away_buildUpPlaySpeed
## 1                  CLB                   231                    35
## 2                  STL                   232                    53
## 3                  BAC                   675                    35
## 4                  WES                   681                    45
## 5                  AND                   229                    50
## 6                  LOK                  2007                    60
##   away_buildUpPlayDribbling away_buildUpPlayPassing away_chanceCreationPassing
## 1                  48.60451                     40                          45
## 2                  48.60451                     40                          55
## 3                  48.60451                     35                          45
## 4                  48.60451                     60                          40
## 5                  48.60451                     35                          70
## 6                  48.60451                     60                          50
##   away_chanceCreationCrossing away_chanceCreationShooting away_defencePressure
## 1                          50                          45                    60
## 2                          55                          65                    70
## 3                          40                          50                    70
## 4                          45                          55                    70
## 5                          50                          60                    70
## 6                          40                          50                    65
##   away_defenceAggression away_defenceTeamWidth
## 1                     70                    70
## 2                     70                    65
## 3                     70                    70
## 4                     70                    70
## 5                     50                    70
## 6                     65                    70
```

```
head(test_matches)
```

```
##   match_api_id season       date home_team_api_id away_team_api_id
## 1      1717979   2015 2015-02-03             9987             8573
## 2      1717985   2015 2015-01-16             8342             8203
## 3      1717986   2015 2015-01-18             8635             9989
## 4      1717987   2015 2015-01-17             8573             8571
## 5      1717988   2015 2015-01-17             9986             9984
## 6      1717989   2015 2015-01-17           274581             9991
##   home_team_goal away_team_goal home_result  home_team_long_name
```

```
## 1              1              1        Draw             KRC Genk
## 2              1              1        Draw       Club Brugge KV
## 3              3              0         Win       RSC Anderlecht
## 4              1              7        Loss          KV Oostende
## 5              0              2        Loss   Sporting Charleroi
## 6              1              3        Loss Royal Excel Mouscron
##   home_team_short_name home_team_fifa_api_id home_buildUpPlaySpeed
## 1                  GEN                   673                    58
## 2                  CLB                   231                    42
## 3                  AND                   229                    52
## 4                  OOS                   682                    52
## 5                  CHA                   670                    60
## 6                  MOP                111560                    50
##   home_buildUpPlayDribbling home_buildUpPlayPassing home_chanceCreationPassing
## 1                        52                      38                         30
## 2                        52                      45                         42
## 3                        46                      51                         60
## 4                        49                      54                         45
## 5                        47                      53                         48
## 6                        50                      50                         50
##   home_chanceCreationCrossing home_chanceCreationShooting home_defencePressure
## 1                          69                          56                    36
## 2                          57                          47                    51
## 3                          53                          47                    53
## 4                          42                          53                    42
## 5                          50                          48                    39
## 6                          50                          50                    45
##   home_defenceAggression home_defenceTeamWidth away_team_long_name
## 1                     57                    70         KV Oostende
## 2                     50                    57         KV Mechelen
## 3                     50                    61           Lierse SK
## 4                     47                    44         KV Kortrijk
## 5                     48                    49   KSV Cercle Brugge
## 6                     45                    50            KAA Gent
##   away_team_short_name away_team_fifa_api_id away_buildUpPlaySpeed
## 1                  OOS                   682                    52
## 2                  MEC                110724                    52
## 3                  LIE                   239                    69
## 4                  KOR                100081                    42
## 5                  CEB                  1750                    53
## 6                  GEN                   674                    50
##   away_buildUpPlayDribbling away_buildUpPlayPassing away_chanceCreationPassing
## 1                        49                      54                         45
## 2                        42                      40                         47
## 3                        44                      58                         56
## 4                        40                      47                         37
## 5                        49                      43                         52
## 6                        55                      37                         42
##   away_chanceCreationCrossing away_chanceCreationShooting away_defencePressure
## 1                          42                          53                    42
## 2                          53                          50                    42
## 3                          52                          61                    38
## 4                          67                          30                    46
## 5                          58                          42                    59
```

```
## 6                          66                          34                    49
##   away_defenceAggression away_defenceTeamWidth
## 1                     47                    44
## 2                     51                    52
## 3                     49                    65
## 4                     45                    52
## 5                     47                    62
## 6                     54                    52
```

```
#write.csv(train_matches, file = "train_matches.csv", row.names = FALSE)
#write.csv(test_matches, file = "test_matches.csv", row.names = FALSE)
```

```
# Prepare training and test data with predictors only
train_data <- train_matches |>
  mutate(
    buildUpPlaySpeed_diff        = home_buildUpPlaySpeed        - away_buildUpPlaySpeed,
    buildUpPlayPassing_diff      = home_buildUpPlayPassing      - away_buildUpPlayPassing,
    chanceCreationPassing_diff   = home_chanceCreationPassing   - away_chanceCreationPassing,
    chanceCreationCrossing_diff  = home_chanceCreationCrossing  - away_chanceCreationCrossing,
    chanceCreationShooting_diff  = home_chanceCreationShooting  - away_chanceCreationShooting,
    defencePressure_diff         = home_defencePressure         - away_defencePressure,
    defenceAggression_diff       = home_defenceAggression       - away_defenceAggression,
    defenceTeamWidth_diff        = home_defenceTeamWidth        - away_defenceTeamWidth,
    speed_shooting_int           = buildUpPlaySpeed_diff * chanceCreationShooting_diff,
    defence_press_agg_int        = defencePressure_diff * defenceAggression_diff
  ) |>
  select(
    home_result,
    buildUpPlaySpeed_diff,
    buildUpPlayPassing_diff,
    chanceCreationPassing_diff,
    chanceCreationCrossing_diff,
    chanceCreationShooting_diff,
    defencePressure_diff,
    defenceAggression_diff,
    defenceTeamWidth_diff,
    speed_shooting_int,
    defence_press_agg_int
  ) |>
  drop_na()

test_data <- test_matches |>
  mutate(
    buildUpPlaySpeed_diff        = home_buildUpPlaySpeed        - away_buildUpPlaySpeed,
    buildUpPlayPassing_diff      = home_buildUpPlayPassing      - away_buildUpPlayPassing,
    chanceCreationPassing_diff   = home_chanceCreationPassing   - away_chanceCreationPassing,
    chanceCreationCrossing_diff  = home_chanceCreationCrossing  - away_chanceCreationCrossing,
    chanceCreationShooting_diff  = home_chanceCreationShooting  - away_chanceCreationShooting,
    defencePressure_diff         = home_defencePressure         - away_defencePressure,
    defenceAggression_diff       = home_defenceAggression       - away_defenceAggression,
    defenceTeamWidth_diff        = home_defenceTeamWidth        - away_defenceTeamWidth,
    speed_shooting_int           = buildUpPlaySpeed_diff * chanceCreationShooting_diff,
    defence_press_agg_int        = defencePressure_diff * defenceAggression_diff
  ) |>
  select(
```

```
    home_result,
    buildUpPlaySpeed_diff,
    buildUpPlayPassing_diff,
    chanceCreationPassing_diff,
    chanceCreationCrossing_diff,
    chanceCreationShooting_diff,
    defencePressure_diff,
    defenceAggression_diff,
    defenceTeamWidth_diff,
    speed_shooting_int,
    defence_press_agg_int
  )

## Drop predictors that are entirely NA ----
na_all_cols <- names(train_data)[sapply(train_data, function(x) all(is.na(x)))]

na_all_cols
```

```
## character(0)
```

```
train_data <- train_data |>
  select(-all_of(na_all_cols))

test_data <- test_data |>
  select(-all_of(na_all_cols))

## Also drop any remaining rows with NA in predictors or outcome
train_data <- train_data |>
  drop_na()

test_data <- test_data |>
  drop_na()
```

```
## Logistic Regression ----

# Specify multinomial logistic regression
logistic_spec <- multinom_reg(mode = "classification") |>
  set_engine("nnet")

# Fit model on training data
logistic_spec |>
  fit(home_result ~ ., data = train_data) -> m0.fit

# Training confusion matrix
cat("=== MULTINOMIAL LOGISTIC RESULTS ===\n")
```

```
## === MULTINOMIAL LOGISTIC RESULTS ===
```

```
m0.fit |>
  augment(new_data = train_data) |>
  conf_mat(truth = home_result, estimate = .pred_class)
```

```
##           Truth
## Prediction  Win Draw Loss
##       Win  6607 3539 3854
##       Draw    0    0    0
```

```
##       Loss  369  283  438
```

```
# Training error rate
m0.fit |>
  augment(new_data = train_data) |>
  accuracy(truth = home_result, estimate = .pred_class) |>
  mutate(error = 1 - .estimate) |>
  pull(error)
```

```
## [1] 0.5330683
```

```
# Test confusion matrix and error rate
logistic_spec |>
  fit(home_result ~ ., data = train_data) |>
  augment(new_data = test_data) -> m0.test_res

m0.test_res |>
  conf_mat(truth = home_result, estimate = .pred_class)
```

```
##           Truth
## Prediction  Win Draw Loss
##       Win  1337  743  850
##       Draw    0    0    0
##       Loss   68   63   92
```

```
m0.test_res |>
  accuracy(truth = home_result, estimate = .pred_class) |>
  mutate(error = 1 - .estimate) |>
  pull(error)
```

```
## [1] 0.5467808
```

```
### LDA ----

cat("=== LDA RESULTS ===\n")
```

```
## === LDA RESULTS ===
```

```
# Specify LDA model
lda_spec <- discrim_linear()

# Fit on training data
lda_spec |>
  fit(home_result ~ ., data = train_data) -> lda.fit

# Training confusion matrix
lda.fit |>
  augment(new_data = train_data) |>
  conf_mat(truth = home_result, estimate = .pred_class)
```

```
##           Truth
## Prediction  Win Draw Loss
##       Win  6613 3544 3858
##       Draw    0    0    0
##       Loss  363  278  434
```

```
# Training error rate
lda.fit |>
  augment(new_data = train_data) |>
```

```
  accuracy(truth = home_result, estimate = .pred_class) |>
  mutate(error = 1 - .estimate) |>
  pull(error)
```

```
## [1] 0.533002
```

```
# Test confusion matrix + test error
lda.fit |>
  augment(new_data = test_data) -> lda.test_res

lda.test_res |>
  conf_mat(truth = home_result, estimate = .pred_class)
```

```
##           Truth
## Prediction  Win Draw Loss
##       Win  1337  745  847
##       Draw    0    0    0
##       Loss   68   61   95
```

```
lda.test_res |>
  accuracy(truth = home_result, estimate = .pred_class) |>
  mutate(error = 1 - .estimate) |>
  pull(error)
```

```
## [1] 0.5458294
```

```
### QDA ----

cat("=== QDA RESULTS ===\n")
```

```
## === QDA RESULTS ===
```

```
qda_spec <- discrim_quad()

# Fit on training data
qda_spec |>
  fit(home_result ~ ., data = train_data) -> qda.fit

# Training confusion matrix
qda.fit |>
  augment(new_data = train_data) |>
  conf_mat(truth = home_result, estimate = .pred_class)
```

```
##           Truth
## Prediction  Win Draw Loss
##       Win  5986 3134 3343
##       Draw   35   31   30
##       Loss  955  657  919
```

```
# Training error
qda.fit |>
  augment(new_data = train_data) |>
  accuracy(truth = home_result, estimate = .pred_class) |>
  mutate(error = 1 - .estimate) |>
  pull(error)
```

```
## [1] 0.5404904
```

```r
# Test predictions
qda.fit |>
  augment(new_data = test_data) -> qda.test_res

qda.test_res |>
  conf_mat(truth = home_result, estimate = .pred_class)
```

```
##           Truth
## Prediction  Win Draw Loss
##       Win  1214  648  704
##       Draw    5    3    0
##       Loss  186  155  238
```

```r
qda.test_res |>
  accuracy(truth = home_result, estimate = .pred_class) |>
  mutate(error = 1 - .estimate) |>
  pull(error)
```

```
## [1] 0.5385347
```

```r
### KNN ----

cat("=== KNN (K=1) RESULTS ===\n")
```

```
## === KNN (K=1) RESULTS ===
```

```r
## K = 1
knn1_spec <- nearest_neighbor(mode = "classification", neighbors = 1)

knn1_spec |>
  fit(home_result ~ ., data = train_data) -> knn1.fit

knn1.fit |>
  augment(new_data = test_data) -> knn1.test_res

knn1.test_res |>
  conf_mat(truth = home_result, estimate = .pred_class)
```

```
##           Truth
## Prediction Win Draw Loss
##       Win  681  361  421
##       Draw 356  201  241
##       Loss 368  244  280
```

```r
knn1.test_res |>
  accuracy(truth = home_result, estimate = .pred_class) |>
  mutate(error = 1 - .estimate) |>
  pull(error)
```

```
## [1] 0.6314621
```

```r
## K = 3

cat("=== KNN (K=3) RESULTS ===\n")
```

```
## === KNN (K=3) RESULTS ===
```

```r
knn3_spec <- nearest_neighbor(mode = "classification", neighbors = 3)

knn3_spec |>
  fit(home_result ~ ., data = train_data) -> knn3.fit

knn3.fit |>
  augment(new_data = test_data) -> knn3.test_res

knn3.test_res |>
  conf_mat(truth = home_result, estimate = .pred_class)
```

```
##           Truth
## Prediction Win Draw Loss
##       Win  678  359  420
##       Draw 364  199  245
##       Loss 363  248  277
```

```r
knn3.test_res |>
  accuracy(truth = home_result, estimate = .pred_class) |>
  mutate(error = 1 - .estimate) |>
  pull(error)
```

```
## [1] 0.6339994
```

```r
## K = 5

cat("=== KNN (K=5) RESULTS ===\n")
```

```
## === KNN (K=5) RESULTS ===
```

```r
knn5_spec <- nearest_neighbor(mode = "classification", neighbors = 5)

knn5_spec |>
  fit(home_result ~ ., data = train_data) -> knn5.fit

knn5.fit |>
  augment(new_data = test_data) -> knn5.test_res

knn5.test_res |>
  conf_mat(truth = home_result, estimate = .pred_class)
```

```
##           Truth
## Prediction Win Draw Loss
##       Win  738  390  429
##       Draw 327  189  227
##       Loss 340  227  286
```

```r
knn5.test_res |>
  accuracy(truth = home_result, estimate = .pred_class) |>
  mutate(error = 1 - .estimate) |>
  pull(error)
```

```
## [1] 0.615287
```

```r
### Validation Set Approach ----

goals_train <- train_matches |>
```

```
  mutate(
    buildUpPlaySpeed_diff        = home_buildUpPlaySpeed        - away_buildUpPlaySpeed,
    buildUpPlayPassing_diff      = home_buildUpPlayPassing      - away_buildUpPlayPassing,
    chanceCreationPassing_diff   = home_chanceCreationPassing   - away_chanceCreationPassing,
    chanceCreationCrossing_diff  = home_chanceCreationCrossing  - away_chanceCreationCrossing,
    chanceCreationShooting_diff  = home_chanceCreationShooting  - away_chanceCreationShooting,
    defencePressure_diff         = home_defencePressure         - away_defencePressure,
    defenceAggression_diff       = home_defenceAggression       - away_defenceAggression,
    defenceTeamWidth_diff        = home_defenceTeamWidth        - away_defenceTeamWidth,
    speed_shooting_int           = buildUpPlaySpeed_diff * chanceCreationShooting_diff,
    defence_press_agg_int        = defencePressure_diff * defenceAggression_diff
  ) |>
  select(
    home_team_goal,
    buildUpPlaySpeed_diff,
    buildUpPlayPassing_diff,
    chanceCreationPassing_diff,
    chanceCreationCrossing_diff,
    chanceCreationShooting_diff,
    defencePressure_diff,
    defenceAggression_diff,
    defenceTeamWidth_diff,
    speed_shooting_int,
    defence_press_agg_int
  )

goals_test  <- test_matches |>
  mutate(
    buildUpPlaySpeed_diff        = home_buildUpPlaySpeed        - away_buildUpPlaySpeed,
    buildUpPlayPassing_diff      = home_buildUpPlayPassing      - away_buildUpPlayPassing,
    chanceCreationPassing_diff   = home_chanceCreationPassing   - away_chanceCreationPassing,
    chanceCreationCrossing_diff  = home_chanceCreationCrossing  - away_chanceCreationCrossing,
    chanceCreationShooting_diff  = home_chanceCreationShooting  - away_chanceCreationShooting,
    defencePressure_diff         = home_defencePressure         - away_defencePressure,
    defenceAggression_diff       = home_defenceAggression       - away_defenceAggression,
    defenceTeamWidth_diff        = home_defenceTeamWidth        - away_defenceTeamWidth,
    speed_shooting_int           = buildUpPlaySpeed_diff * chanceCreationShooting_diff,
    defence_press_agg_int        = defencePressure_diff * defenceAggression_diff
  ) |>
  select(
    home_team_goal,
    buildUpPlaySpeed_diff,
    buildUpPlayPassing_diff,
    chanceCreationPassing_diff,
    chanceCreationCrossing_diff,
    chanceCreationShooting_diff,
    defencePressure_diff,
    defenceAggression_diff,
    defenceTeamWidth_diff,
    speed_shooting_int,
    defence_press_agg_int
  )
```

```r
## Drop predictors that are entirely NA ----
na_all_cols2 <- names(goals_train)[sapply(goals_train, function(x) all(is.na(x)))]

goals_train <- goals_train |>
  select(-all_of(na_all_cols2))

goals_test <- goals_test |>
  select(-all_of(na_all_cols2))

## Also drop any remaining rows with NA in predictors or outcome
goals_train <- goals_train |>
  drop_na()

goals_test <- goals_test |>
  drop_na()

## Prepare linear regression spec

linear_spec <- linear_reg()

## Base recipe: predict home_team_goal using the best predictors
linear_rec <- recipe(
  home_team_goal ~ ., data = goals_train)


## 4. Fit the linear model
linear_model <- workflow() |>
  add_model(linear_spec) |>
  add_recipe(linear_rec) |>
  fit(data = goals_train)

## Compute test MSE for linear model

linear_model |>
  augment(new_data = goals_test) |>
  mutate(resid2 = (.resid)^2) |>
  summarise(mse = mean(resid2, na.rm = TRUE)) |>
  mutate(model = "Linear Regression (Goals)") |>
  relocate(model)
```

```
## # A tibble: 1 x 2
##   model                       mse
##   <chr>                     <dbl>
## 1 Linear Regression (Goals)  1.64
```

```r
### Quadratic Model ----

## Add squared terms

quad_rec <- linear_rec |>
  step_mutate(
    buildUpPlaySpeed_diff2       = buildUpPlaySpeed_diff^2,
    buildUpPlayPassing_diff2     = buildUpPlayPassing_diff^2,
    chanceCreationPassing_diff2  = chanceCreationPassing_diff^2,
```

```
    chanceCreationCrossing_diff2 = chanceCreationCrossing_diff^2,
    chanceCreationShooting_diff2 = chanceCreationShooting_diff^2,
    defencePressure_diff2        = defencePressure_diff^2,
    defenceAggression_diff2      = defenceAggression_diff^2,
    defenceTeamWidth_diff2       = defenceTeamWidth_diff^2
  )

workflow() |>
  add_model(linear_spec) |>
  add_recipe(quad_rec) |>
  fit(data = goals_train) |>
  augment(new_data = goals_test) |>
  mutate(resid2 = (.resid)^2) |>
  summarise(mse = mean(resid2, na.rm = TRUE)) |>
  mutate(model = "Quadratic Regression (Goals)") |>
  relocate(model)
```

```
## # A tibble: 1 x 2
##   model                          mse
##   <chr>                        <dbl>
## 1 Quadratic Regression (Goals)  1.64
```

```
### Cubic Model ----

## Add cubic terms the same way

cubic_rec <- quad_rec |>
  step_mutate(
    buildUpPlaySpeed_diff3       = buildUpPlaySpeed_diff^3,
    buildUpPlayPassing_diff3     = buildUpPlayPassing_diff^3,
    chanceCreationPassing_diff3  = chanceCreationPassing_diff^3,
    chanceCreationCrossing_diff3 = chanceCreationCrossing_diff^3,
    chanceCreationShooting_diff3 = chanceCreationShooting_diff^3,
    defencePressure_diff3        = defencePressure_diff^3,
    defenceAggression_diff3      = defenceAggression_diff^3,
    defenceTeamWidth_diff3       = defenceTeamWidth_diff^3
  )

workflow() |>
  add_model(linear_spec) |>
  add_recipe(cubic_rec) |>
  fit(data = goals_train) |>
  augment(new_data = goals_test) |>
  mutate(resid2 = (.resid)^2) |>
  summarise(mse = mean(resid2, na.rm = TRUE)) |>
  mutate(model = "Cubic Regression (Goals)") |>
  relocate(model)
```

```
## # A tibble: 1 x 2
##   model                      mse
##   <chr>                    <dbl>
## 1 Cubic Regression (Goals)  1.63
```

```
### KNN ----
```

```r
goals_knn <- train_matches |>
  mutate(
    buildUpPlaySpeed_diff        = home_buildUpPlaySpeed        - away_buildUpPlaySpeed,
    buildUpPlayPassing_diff      = home_buildUpPlayPassing      - away_buildUpPlayPassing,
    chanceCreationPassing_diff   = home_chanceCreationPassing   - away_chanceCreationPassing,
    chanceCreationCrossing_diff  = home_chanceCreationCrossing  - away_chanceCreationCrossing,
    chanceCreationShooting_diff  = home_chanceCreationShooting  - away_chanceCreationShooting,
    defencePressure_diff         = home_defencePressure         - away_defencePressure,
    defenceAggression_diff       = home_defenceAggression       - away_defenceAggression,
    defenceTeamWidth_diff        = home_defenceTeamWidth        - away_defenceTeamWidth,
    speed_shooting_int           = buildUpPlaySpeed_diff * chanceCreationShooting_diff,
    defence_press_agg_int        = defencePressure_diff * defenceAggression_diff
  ) |>
  select(
    home_team_goal,
    buildUpPlaySpeed_diff,
    buildUpPlayPassing_diff,
    chanceCreationPassing_diff,
    chanceCreationCrossing_diff,
    chanceCreationShooting_diff,
    defencePressure_diff,
    defenceAggression_diff,
    defenceTeamWidth_diff,
    speed_shooting_int,
    defence_press_agg_int
  ) |>
  drop_na()

na_all_cols3 <- names(goals_knn)[sapply(goals_knn, function(x) all(is.na(x)))]

goals_knn <- goals_knn |>
  select(-all_of(na_all_cols3))

## Also drop any remaining rows with NA in predictors or outcome
goals_knn <- goals_knn |>
  drop_na()

## KNN k-fold CV function ----
k_fold_cv_err_knn <- function(k_fold = 10, knn, data, formula) {
  data.kfold <- vfold_cv(data, v = k_fold)

  knn_spec <- nearest_neighbor(mode = "regression", neighbors = knn)

  knn_rec <- recipe(formula, data = data) |>
    step_normalize(all_predictors())

  workflow() |>
    add_model(knn_spec) |>
    add_recipe(knn_rec) |>
    fit_resamples(data.kfold) |>
    collect_metrics() |>
    select(.metric, mean) |>
    pivot_wider(names_from = .metric, values_from = mean) |>
```

```
    mutate(mse = rmse^2) |>
    pull(mse)
}

## Evaluate KNN for K = 1, 5, 10, 20, 100 ----
res <- data.frame(knn = c(1, 5, 10, 20, 100))

for(i in seq_len(nrow(res))) {
  res[i, "mse"] <- k_fold_cv_err_knn(
    k_fold = 10,
    knn = res[i, "knn"],
    data = goals_knn,
    formula = home_team_goal ~ .
  )
}

## Inspect results and best K ----
res
```

```
##   knn      mse
## 1   1 2.986017
## 2   5 2.038056
## 3  10 1.826022
## 4  20 1.729794
## 5 100 1.663585
```

```
res[which.min(res$mse), ]
```

```
##   knn      mse
## 5 100 1.663585
```

```
# Build complete data set with engineered predictors and no missing values

goals_ridge <- train_matches |>
  mutate(
    buildUpPlaySpeed_diff        = home_buildUpPlaySpeed        - away_buildUpPlaySpeed,
    buildUpPlayPassing_diff      = home_buildUpPlayPassing      - away_buildUpPlayPassing,
    chanceCreationPassing_diff   = home_chanceCreationPassing   - away_chanceCreationPassing,
    chanceCreationCrossing_diff  = home_chanceCreationCrossing  - away_chanceCreationCrossing,
    chanceCreationShooting_diff  = home_chanceCreationShooting  - away_chanceCreationShooting,
    defencePressure_diff         = home_defencePressure         - away_defencePressure,
    defenceAggression_diff       = home_defenceAggression       - away_defenceAggression,
    defenceTeamWidth_diff        = home_defenceTeamWidth        - away_defenceTeamWidth,
    speed_shooting_int           = buildUpPlaySpeed_diff * chanceCreationShooting_diff,
    defence_press_agg_int        = defencePressure_diff * defenceAggression_diff
  ) |>
  select(
    home_team_goal,
    buildUpPlaySpeed_diff,
    buildUpPlayPassing_diff,
    chanceCreationPassing_diff,
    chanceCreationCrossing_diff,
    chanceCreationShooting_diff,
    defencePressure_diff,
    defenceAggression_diff,
```

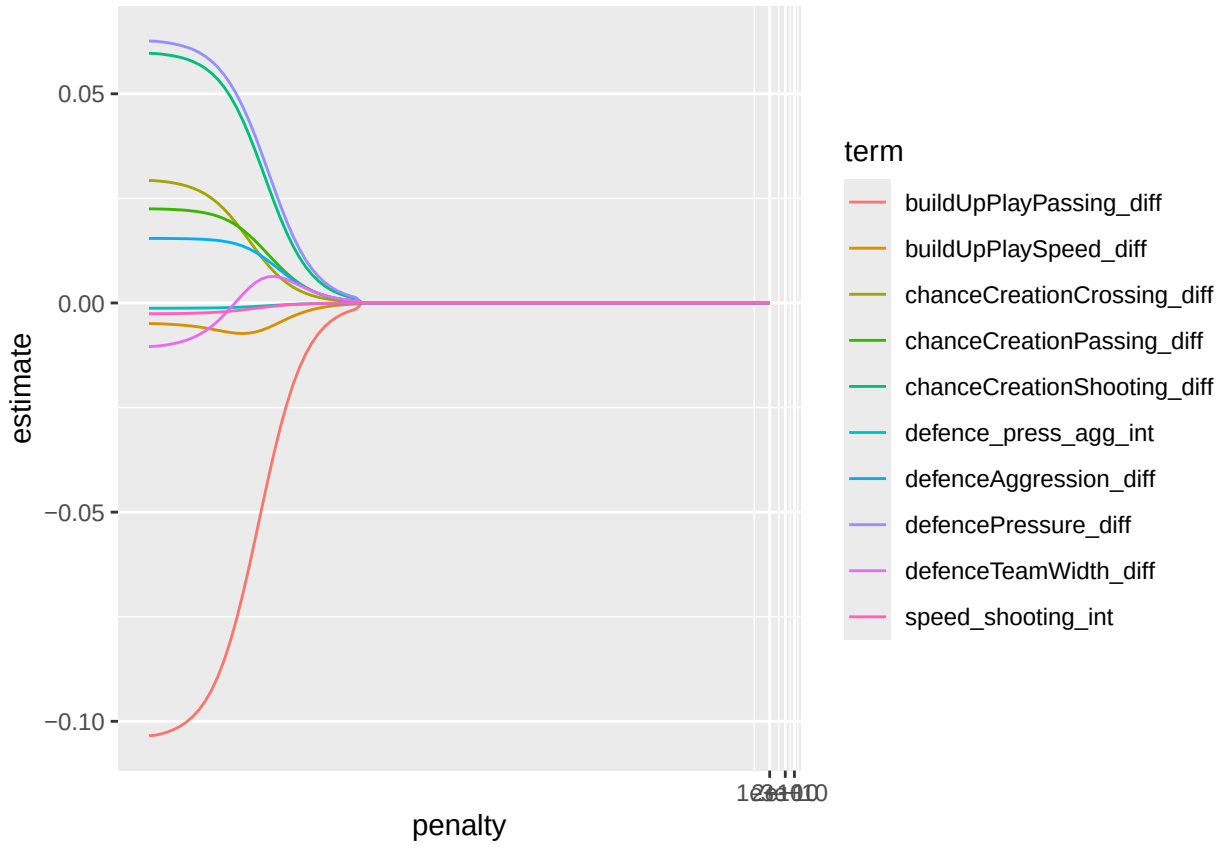
```
    defenceTeamWidth_diff,
    speed_shooting_int,
    defence_press_agg_int
  ) |>
  drop_na()

## Ridge Regression ----
# Lambda grid
lambda  <- 10^seq(-2, 10, length.out = 100)
tune_df <- data.frame(lambda = lambda)

# Recipe: standardize predictors
prep_data <- recipe(home_team_goal ~ ., data = goals_ridge) |>
  step_normalize(all_predictors())

# Fit ridge for each lambda and store coefficient paths
ridge_ests <- data.frame()
for(lam in lambda) {
  ridge_spec <- linear_reg(mixture = 0, penalty = lam) |>
    set_mode("regression") |>
    set_engine("glmnet")

  workflow() |>
    add_model(ridge_spec) |>
    add_recipe(prep_data) |>
    fit(goals_ridge) |>
    tidy() |>
    bind_rows(ridge_ests) -> ridge_ests
}
```

```
##
## Attaching package: 'Matrix'
```

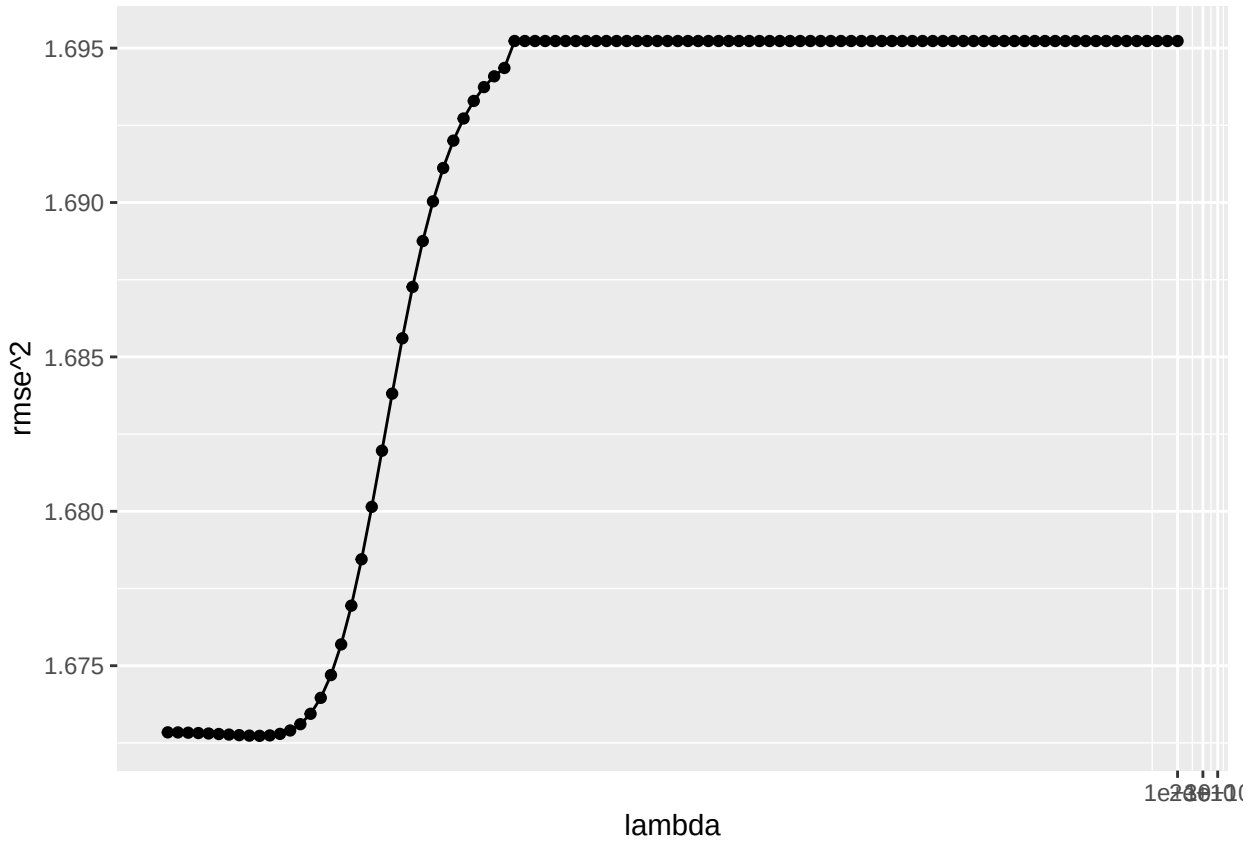
```
## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack
```

```
## Loaded glmnet 4.1-10
```

```
# Plot coefficient paths vs lambda
ridge_ests |>
  filter(term != "(Intercept)") |>
  ggplot() +
  geom_line(aes(penalty, estimate, group = term, colour = term)) +
  coord_transform(x = "log10")
```




```
## 10-fold CV to choose lambda ----
goals_ridge_10foldcv <- vfold_cv(goals_ridge, v = 10)

ridge_spec <- linear_reg(mixture = 0, penalty = tune("lambda")) |>
  set_mode("regression") |>
  set_engine("glmnet")

workflow() |>
  add_model(ridge_spec) |>
  add_recipe(prep_data) |>
  tune_grid(resamples = goals_ridge_10foldcv, grid = tune_df, metrics = metric_set(rmse)) -> ridge_tune

ridge_tune |>
  collect_metrics() |>
  select(lambda, .metric, mean) |>
  pivot_wider(names_from = .metric, values_from = mean) |>
  ggplot() +
  geom_line(aes(lambda, rmse^2)) +
  geom_point(aes(lambda, rmse^2)) +
  coord_transform(x = "log10")
```




```
## Best penalty (lambda) based on CV RMSE
show_best(ridge_tune, metric = "rmse", n = 1)
```

```
## # A tibble: 1 x 7
##   lambda .metric .estimator  mean     n std_err .config
##    <dbl> <chr>   <chr>      <dbl> <int>   <dbl> <chr>
## 1  0.123 rmse    standard    1.29    10  0.0128 Preprocessor1_Model010
```