Surface and Time

```
library(TennisBT)
```

```
##
## Attaching package: 'TennisBT'

## The following object is masked from 'package:stats':
##
##     predict
```

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.3     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.4     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.0
## v purrr     1.0.2

## -- 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 error
```

```
library(BradleyTerry2)

data <- TennisTidyr(2013, 2022, 2023, 20)
train <- data$train
test <- data$test
players <- data$main_names
```

Set up the model as with in the surface experiment.

```
final_grass_clay <- 0.4
final_hard_clay <- 0.8
final_grass_hard <- 0.5
```

```
predict_with_surface <- function(time_weights) {

  weighting <- as.data.frame(diag(3))
  rownames(weighting) <- c("Grass", "Hard", "Clay")
  colnames(weighting) <- c("Grass", "Hard", "Clay")
  weighting['Grass', 'Hard'] <- final_grass_hard
  weighting['Hard', 'Grass'] <- final_grass_hard
  weighting['Clay', 'Hard'] <- final_hard_clay
  weighting['Hard', 'Clay'] <- final_hard_clay
  weighting['Clay', 'Grass'] <- final_grass_clay
  weighting['Grass', 'Clay'] <- final_grass_clay
  for (i in 1:nrow(train)) {
    train[i, 'grass_weight'] <- weighting[train[i, 'surface'], 'Grass']
    train[i, 'clay_weight'] <- weighting[train[i, 'surface'], 'Clay']
    train[i, 'hard_weight'] <- weighting[train[i, 'surface'], 'Hard']
  }

  btm_grass <- BTm(outcome = 1, factor(winner_name, levels=players), factor(loser_name, levels=players)
  btm_clay <- BTm(outcome = 1, factor(winner_name, levels=players), factor(loser_name, levels=players),
  btm_hard <- BTm(outcome = 1, factor(winner_name, levels=players), factor(loser_name, levels=players),

  # Create a dataframe with the Bradley-Terry estimates for each surface
  btm_grass_df <- as.data.frame(BTabilities(btm_grass))
  btm_clay_df <- as.data.frame(BTabilities(btm_clay))
  btm_hard_df <- as.data.frame(BTabilities(btm_hard))

  coeff_frames <- list('Grass' = btm_grass_df, 'Clay' = btm_clay_df,
                       'Hard' = btm_hard_df)

  draws <- test[, c("winner_name", "loser_name", "surface")]
  draws$pred <- NA

  # predict the test set using a different model based on the surface
  for (i in 1:nrow(draws)) {
    player1 <- draws$winner_name[i]
    player2 <- draws$loser_name[i]
    draw_surface <- draws$surface[i]

    pred <- predict(player1, player2, as.data.frame(coeff_frames[draw_surface]))
    draws$pred[i] <- pred[2]
  }

  return(draws)
}
```

Test this surface model with all of the time weighting parameters.

```
scores <- data.frame(matrix(0, ncol=5, nrow=10), row.names = seq(0.1,1,0.1))
colnames(scores) <- c("Accuracy", "Avg Prob", "Avg Prob SE", "Avg Log Prob", "Avg Log Prob SE")

row_count = 1
for (recency_weighting in seq(0.1,1,0.1)) {
  cat("w=",recency_weighting,"\n")
  weights <- get_recency_weights(train, recency_weighting)

  predictions <- predict_with_surface(weights)
  score <- score_predictions(predictions)

  scores[row_count, ]["Accuracy"] <- score$accuracy
  scores[row_count, ]["Avg Prob"] <- score$avg_probability
  scores[row_count, ]["Avg Prob SE"] <- score$avg_prob_se
  scores[row_count, ]["Avg Log Prob"] <- score$avg_log_probability
  scores[row_count, ]["Avg Log Prob SE"] <- score$abg_log_prob_se
  row_count <- row_count + 1
}
```

```
## w= 0.1
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 0.2
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 0.3
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 0.4
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 0.5
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 0.6
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## w= 0.7
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 0.8
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 0.9
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 1
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
scores
```

```
##      Accuracy  Avg Prob Avg Prob SE Avg Log Prob Avg Log Prob SE
## 0.1 0.6435331 0.7091272 0.004202919   -0.3615849     0.005925624
## 0.2 0.6460568 0.6998755 0.004051684   -0.3739154     0.005776137
## 0.3 0.6429022 0.6947137 0.003921399   -0.3803626     0.005622974
## 0.4 0.6384858 0.6907279 0.003803034   -0.3851870     0.005473270
## 0.5 0.6365931 0.6863991 0.003706887   -0.3908332     0.005362498
## 0.6 0.6365931 0.6822523 0.003619995   -0.3963448     0.005256236
## 0.7 0.6359621 0.6790322 0.003555305   -0.4006469     0.005172882
## 0.8 0.6416404 0.6749851 0.003531601   -0.4067163     0.005166466
## 0.9 0.6365931 0.6752287 0.003514150   -0.4061030     0.005138621
## 1   0.6403785 0.6742218 0.003528770   -0.4078452     0.005172765
```

Then we have that the best time weighting is between 0.1 and 0.3, so we can split it down further.

```r
scores <- data.frame(matrix(0, ncol=5, nrow=21), row.names = seq(0.1,0.3,0.01))
colnames(scores) <- c("Accuracy", "Avg Prob", "Avg Prob SE", "Avg Log Prob", "Avg Log Prob SE")

row_count = 1
for (recency_weighting in seq(0.1,0.3,0.01)) {
  cat("w=",recency_weighting,"\n")
  weights <- get_recency_weights(train, recency_weighting)

  predictions <- predict_with_surface(weights)
  score <- score_predictions(predictions)

  scores[row_count, ]["Accuracy"] <- score$accuracy
  scores[row_count, ]["Avg Prob"] <- score$avg_probability
  scores[row_count, ]["Avg Prob SE"] <- score$avg_prob_se
  scores[row_count, ]["Avg Log Prob"] <- score$avg_log_probability
  scores[row_count, ]["Avg Log Prob SE"] <- score$abg_log_prob_se

  row_count <- row_count + 1
}
```

```
## w= 0.1
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 0.11
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 0.12
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 0.13
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## w= 0.14
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 0.15
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 0.16
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 0.17
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 0.18
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 0.19
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!

## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 0.2
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## w= 0.21
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## w= 0.22
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## w= 0.23
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## w= 0.24
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## w= 0.25
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## w= 0.26
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
## w= 0.27
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 0.28
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 0.29
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## w= 0.3
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
## Warning in eval(family$initialize): non-integer #successes in a binomial glm!
```

```
scores
```

```
##       Accuracy  Avg Prob Avg Prob SE Avg Log Prob Avg Log Prob SE
## 0.1  0.6435331 0.7091272 0.004202919   -0.3615849     0.005925624
## 0.11 0.6441640 0.7078832 0.004185573   -0.3632684     0.005910398
## 0.12 0.6441640 0.7068907 0.004168515   -0.3645707     0.005893191
## 0.13 0.6454259 0.7055720 0.004152658   -0.3663960     0.005879804
## 0.14 0.6466877 0.7042927 0.004137977   -0.3681786     0.005868295
## 0.15 0.6460568 0.7036523 0.004121617   -0.3689582     0.005848987
## 0.16 0.6454259 0.7030486 0.004105596   -0.3696876     0.005829749
## 0.17 0.6460568 0.7020724 0.004091921   -0.3710224     0.005817424
## 0.18 0.6466877 0.7011207 0.004078866   -0.3723289     0.005805984
## 0.19 0.6447950 0.7009720 0.004063478   -0.3723647     0.005784452
## 0.2  0.6460568 0.6998755 0.004051684   -0.3739154     0.005776137
## 0.21 0.6422713 0.7003684 0.004033749   -0.3729319     0.005744984
## 0.22 0.6416404 0.6999032 0.004019722   -0.3734814     0.005727781
## 0.23 0.6416404 0.6992562 0.004006816   -0.3743270     0.005713892
## 0.24 0.6410095 0.6988240 0.003993009   -0.3748311     0.005696636
## 0.25 0.6422713 0.6978149 0.003982148   -0.3762646     0.005688916
## 0.26 0.6435331 0.6968232 0.003971297   -0.3776739     0.005680973
## 0.27 0.6435331 0.6962330 0.003958853   -0.3784427     0.005667116
## 0.28 0.6441640 0.6954648 0.003947170   -0.3794990     0.005655822
## 0.29 0.6435331 0.6950860 0.003934199   -0.3799346     0.005639307
## 0.3  0.6429022 0.6947137 0.003921399   -0.3803626     0.005622974
```

Therefore the best model is with surface weighting as defined above, and time weighting of $w = 0.14$.