National Rugby League

Motivating Question

In team sports as long as a player has been able to make a decision; fans, coaches and players have wondered are they making the right decision. Assessing the decision has come a long way in recent years. Starting with the development of expected points [@carter1971operations] and its use on evaluating 4th down decision making [@carter1978note] and continuing today with [@yurko2018nflwar]. However when it comes to evaluating decisions in game by players and coaches evaluating the decisions is done without integrating the question of interest, even though with data accessibility this has become available.

Literature Review (NRL)

Studies for rugby league are overwhelmingly related to physiology. Studies consist of how to best prepare players for games, training and alike. While there has been work related on match outcomes, they tend to be somewhat limited. Limitations arise from the data availability and the methods used. Unfortunately more extensive datasets are not made available and researchers publish utilising only a few variables. Studies of prediction typically focus on scores only [@carbone2016rugby] or on events like Set, Tackle counts not tied to game specific events like kicks or passes[@kempton2016expected]. The other issue is that due to data availability studies are typically done based on summary statistics after the game [@woods2017explaining].

Expected Points

Expected points was developed to reflect the reality that not all yards in American football are created equally. We discuss the justification and interpretation of such models used in team sports such as American Football and Rugby League.

The first unit of analysis in American Football is a single play, in Rugby League it is a single tackle. To be able to objectively quantify onfield decisions and measure player performance each unit must be quantified to let coaches, players and analysts know if the play was a success. Typically this has been done in the NFL using expected points for over a decade [@WinNT], while in the NRL this has only been a recent research interest and hasn't received the same uptake.

Expected points was first developed for the National Football League (NFL) in 1971 [@carter1971operations] by Carter and Machol. Originally they took the play by play logs and entered the data into punch cards – 53 variables per play and 8373 punch cards later had the first iteration of expected points. For each play they determined which team scored next and how many points they scored, the average results gave the expected value of having the ball at different parts of the field. About 20 years later in the seminal book The Hidden Game of Football Carroll, Palmer and Thorn developed a linear model and came to roughly the same conclusion, -2 at your own line and +6 in the opponents end zone. In 2006 David Romer published one of the biggest contributions to applied football research by using expected points to evaluate a decision in game [@romer2006firms.] Romer evaluated forth down decisions with implications still being felt over a decade later [@yamlost].

In 2014, Brian Burke who runs the site advancedfootballanalytics extended the expected points concept to rate plays using expected points added. Expected points added is simply the change in expected points play to play. To evaluate play types, what people would typically do is based on an expected points model, take the EPA by play type, come up with the average of EPA by play type and whichever play type has the highest number becomes the best play [@berri2012measuring]. However, this fails to take into account, covariates of interest, such as where on the pitch the expected value might change, or how the value of the decision might change across the pitch.

It is here, that they originally talked about teams should go for it more on 4th down.

> More significantly, it appears to us that one technical decision is regularly being made incorrectly in professional football. Specifically the negative value of having the ball with first and ten very close to one's own goal line has been ignored. We are referring to the type of situation where a team has the ball, fourth and goal, in which case a field goal is routinely attempted.

Later in 1988 in the Hidden Game of Football a linear expected points model was applied using down and yards to go as covariates.

However this work was done when data wasn't as accessible as it is today. Recent work by [@yurko2018nflwar] in making play by play data readily accessible but they also fit their own expected points model provide this readily to users.

Today fans, analysts and media are concerned with tactical decisions being made on field by the players and coaches. The currency of these decisions is expected points added. Typically what people do is take the expected points added and group by the tactical decision of interest be it runs vs drop backs[1], comparing rushing vs passing [2] and who the quarterback is throwing the ball too [3] etc.

However, what teams, fans and media talk about this tactical decision being better, they do as an aggregated summary. This neglects and loses information around the other covariates that have been built using the model such as field position and downs.

A second issue, that fans encounter, is that these decisions are unable to be plotted across the field on a map with confidence intervals. This continuous production of point averages makes it hard for fans and analysts to tell if there is a significant difference.

This issue arises because of the way in which the expected points models have been built. That is the observations used in the model are not independent. This means that users of expected points are unable to create visual maps with standard errors across the field meaningfully.

A particular issue we consider are the issues around estimation, prediction and hypothesis testing. We suggest that current expected points literature while being able to tell if a play was successful based on expected points added. Currently to assess whether a decision between two play types is better or not, the user simply sums up the expected points added per play type and the one which has the higher EPA per play is considered the better option. However, averaging like this loses a lot of information such as where on the field these decisions might become better or worse and don't include variables which users might be interested in such as players and the decision the player is making. We propose using Generalized Additive Models (GAMS) which have the benefit which avoids problems where the true functional form is unknown. From GAMS we are able to put uncertainty around our estimates, integrate other variables easily and enable analysts to make decisions where on the field decisions have an advantage

Play by Play Data

While play by play data has been available for several decades for Baseball and over a decade for Ice Hockey. Only recently has play by play data been made available for the National Football League [@yurko2019nflwar]. and it is yet to be made publicly available for the National Rugby League.

Problems with current Expected Points Model

Linking the play by play data to events of interest is possible but not done typically in either academic or popular settings.

[1] https://fivethirtyeight.com/features/for-a-passing-league-the-nfl-still-doesnt-pass-enough/

[2] https://theathletic.com/1258367/2019/10/02/chiefs-have-struggled-to-stop-run-but-beating-them-with-run-game-is-suckers-bet/

[3] https://theathletic.com/1143546/2019/08/21/throwing-to-running-backs-the-latest-nfl-craze-that-doesnt-make-any-sense/

In academic settings such as journal articles, this makes more sense as typically the motivation is to provide a general framework for analysts to build upon. However in popular settings such as fivethirtyeight in this we have examples whereby either the author or the reader might be interested in things like the effect sizes or the difference in expected points added

Generalized Additive Models (GAMS)

Generalized Additive models (GAMS) have been used in the sports for win probability [@yurko2019nflwar], umpire bias studies [@lopez2019opportunistic], pitcher performance sievert2017using but not expected points.

Benefits of using GAMS is that they allow the relationship between points scored and the explanatory variables (Tackle, Distance) to vary according to smooth, non-linear functions. We can also plot these across the field allowing users to see where on the field certain decisions of interest have higher value.

Expected Points with Generalized Additive Models (methods)

Data was provided by the NRL.

Problems with current Australian studies applied to team tactics is that they often fail to be useable for coaches and analysts. Commonly with Australian sports studies such as AFL and NRL. The often researchers include variables that are considered mathematically related to the outcome measures and are unlikely to provide insights as to how a team needs to perform to have successful match outcomes [@young2019relationship]. Providing information that the team with more goal assists as a performance indicator doesn't help guide coaches with gameday strategy.

Here as an example we use the decision if a team should take the two after a penalty using expected points as the outcome variable.

Description of dataset

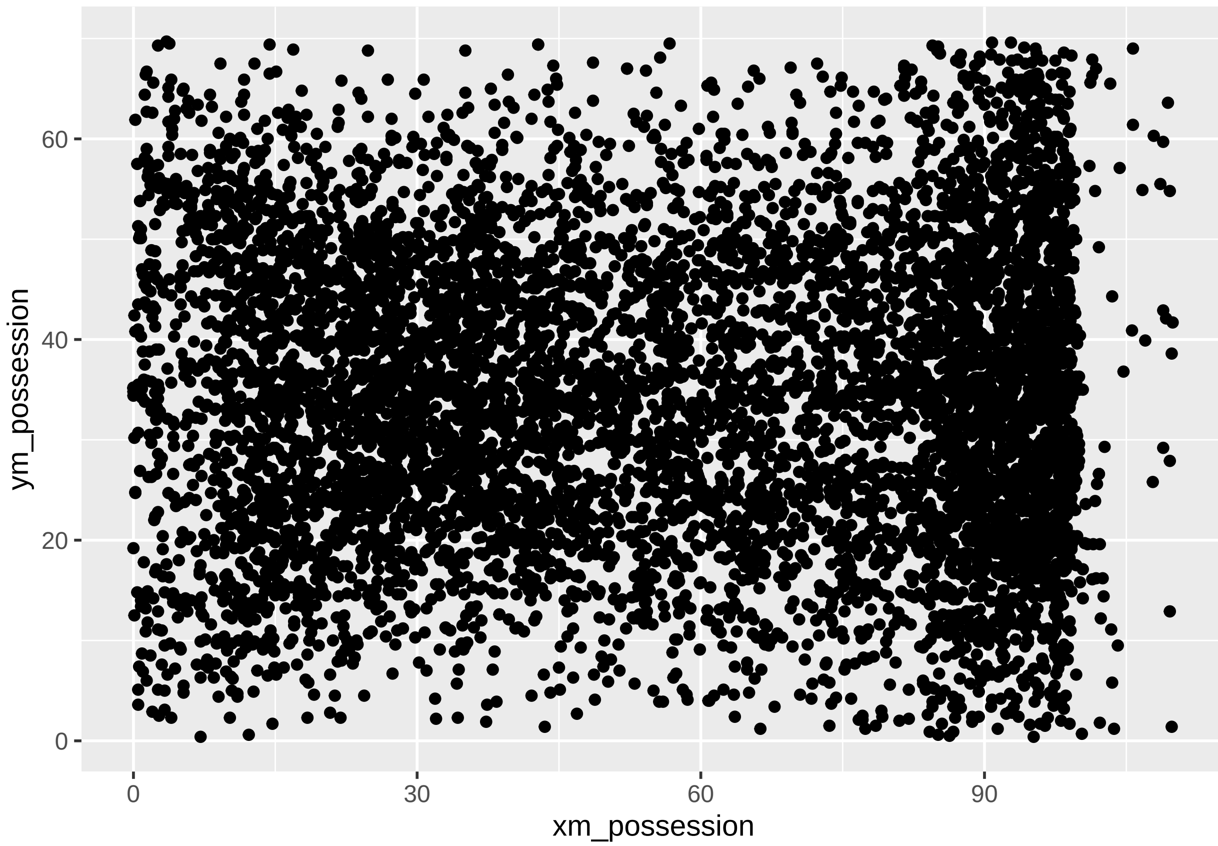
Data was used for the NRL seasons 2018 to 2020 as provided by the NRL. The dataset is event base with an xy location imputed manually for each event recorded.

Variable Definitions:

- match_id - unique identifier for each game
- xm_possession - the length of field location standardised between -10 and 110 with 100 representing the try line
- ym_possession - the width of the field location standardised between 0 and 70 with 70 representing the left side line and 0 representing the right sideline for the team in possession.

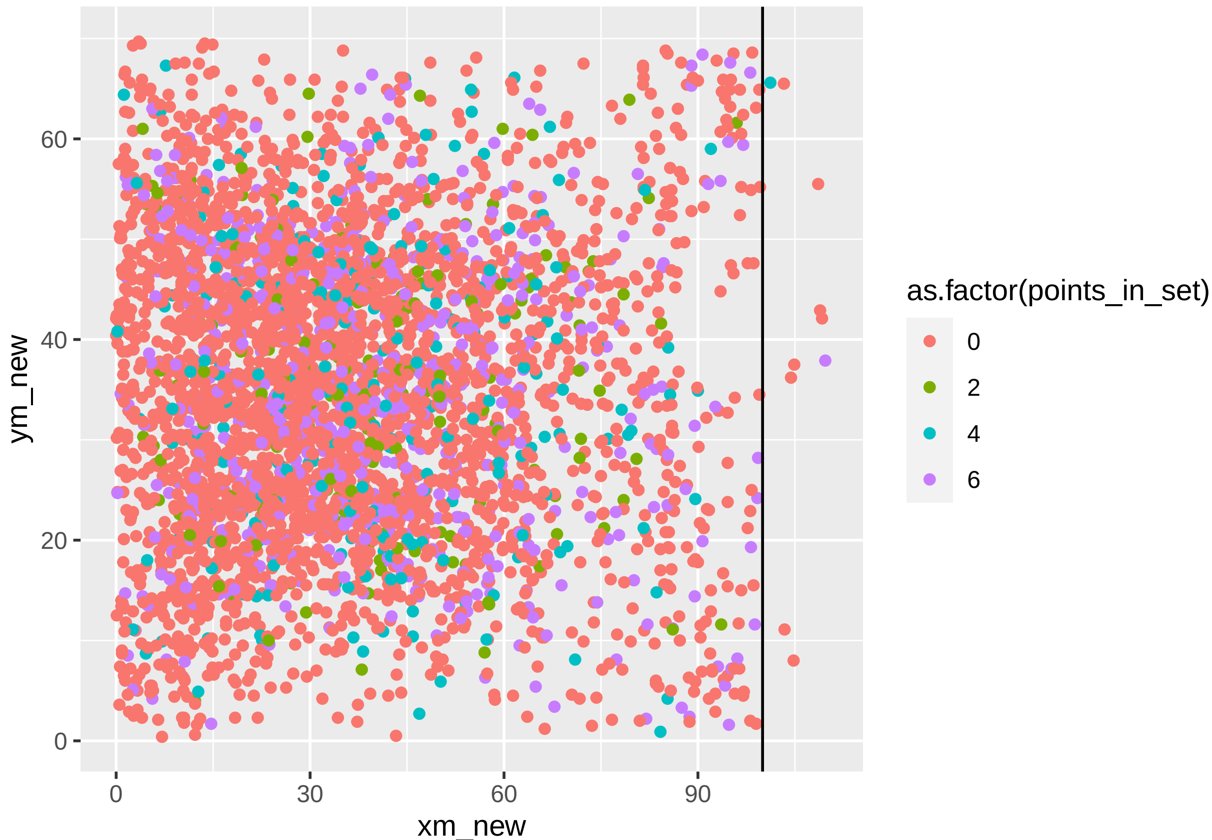
Analysis

```
## Warning: Removed 39 rows containing missing values (geom_point).
```



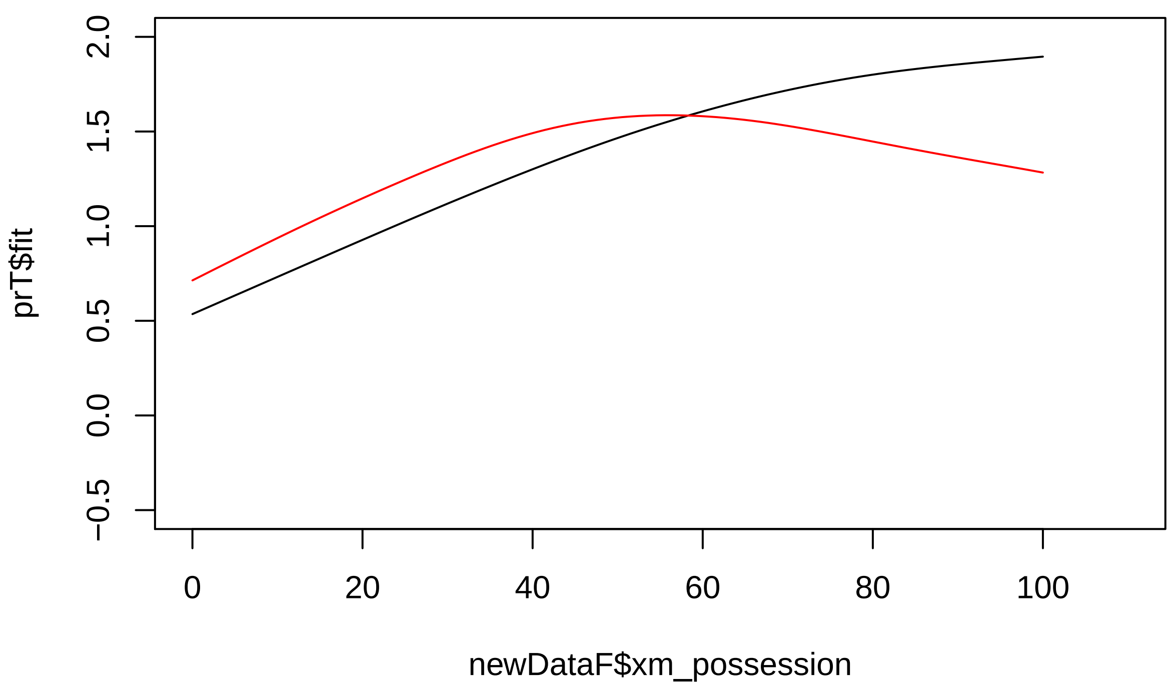

```
## # A tibble: 4,055 x 24
##    match_id player_id seq_number  half   set tackle.x club_id.x in_possession_c~
##       <int>     <int>      <int> <int> <int>    <int>     <int>            <int>
##  1 18111011    500622        242     1     3        3    500022           500011
##  2 18111011    500326        493     1     5        2    500011           500022
##  3 18111011    500622       1990     1    19        1    500022           500011
##  4 18111011    500337       5575     2    49        6    500022           500022
##  5 18111011    500517       5950     2    53        6    500011           500011
##  6 18111011    500459       7029     2    61        0    500022           500011
##  7 18111012    500872        281     1     4        1    500002           500003
##  8 18111012    500960       1189     1    11        2    500002           500003
##  9 18111012    500044       1680     1    16        0    500002           500003
## 10 18111012    501693       2444     1    22        1    500003           500002
## # ... with 4,045 more rows, and 16 more variables: event_name.x <chr>,
## #   xm_possession <dbl>, ym_possession <dbl>, qualifier2 <chr>, tackle.y <int>,
## #   event_name.y <chr>, club_id.y <int>, in_possession_club_id.y <int>,
## #   consecutive_set <dbl>, is_start_chain <dbl>, chain_count <dbl>,
## #   club_id <int>, in_possession_club_id <int>, event_name <chr>, xm_new <dbl>,
## #   ym_new <dbl>
```

```
## Warning: Removed 44 rows containing missing values (geom_point).
```

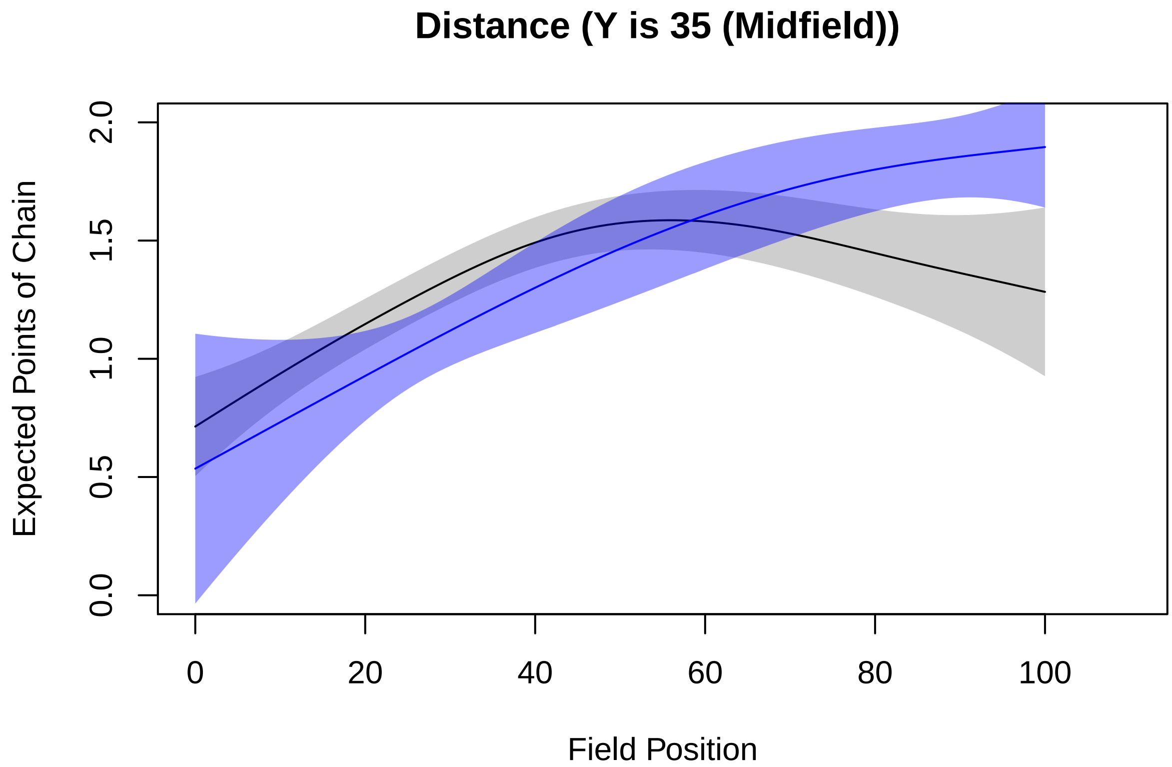


60
40
20
0
ym_new
0
30
60
90
xm_new
as.factor(points_in_set)
0
2
4
6


```
## [1] 939  11
```



2.0
1.5
1.0
0.5
0.0
−0.5
prT$fit
0
20
40
60
80
100
newDataF$xm_possession



How is it different?

One of the things talked about previously was that, one of the things that people don't seem to be doing with expected points, is putting the decision they want to evaluate into the model itself. This seems weird given the most common talked about scenario (run vs pass) has now been readily available for NFL fans for a few years now through data accessibilty in packages like nflscrapR and nflfastR

By not putting the decision that they want directly into the model, they are unable to plot expected points maps across the field. This would enable people to see if the tactical decision changes across the field.

To see why this isn't the case, we need to look at how the most popular versions of expected points in the NFL work.

```
# https://github.com/ryurko/nflscrapR-models/blob/master/R/init_models/init_ep_fg_models.R
# https://github.com/ryurko/nflscrapR-models/tree/master/R/init_models


library(tidyverse)

# Access nflWAR:
# install.packages("devtools")
# devtools::install_github("ryurko/nflWAR")

library(nflWAR)

# Load data from 2009 to 2016 from the nflscrapR-data repository using the
# get_pbp_data() function from the nflWAR package:
pbp_data <- get_pbp_data(2009:2016)
```

```r
# Remove error game from 2011 that is coded incorrectly in raw JSON data:
pbp_data <- pbp_data %>% filter(GameID != "2011121101")

nrow(pbp_data)
# 362263

#' Define a function that takes in a play-by-play data set and returns
#' what the type of next score is and the drive number only within same half.
#' @param pbp_dataset Play-by-play dataset with the following columns:
#' sp - scoring play indicator, PlayType - what type of play, qtr - quarter
#' of the game, Drive - drive number for the play, ReturnResult - indicates
#' what happened on return type plays, posteam - indicates the possession team
#' for the play, and columns for ReturnResult, FieldGoalResult, ExPointResult,
#' TwoPointConv, DefTwoPoint, Touchdown.
#' @return Data frame with two columns: Next_Score_Half denoting the type of
#' the next scoring event occurring within the half of each play and
#' Drive_Score_Half denoting the drive number of the next scoring play.

find_game_next_score_half <- function(pbp_dataset) {

  # Which rows are the scoring plays:
  score_plays <- which(pbp_dataset$sp == 1 & pbp_dataset$PlayType != "No Play")

  # Define a helper function that takes in the current play index,
  # a vector of the scoring play indices, play-by-play data,
  # and returns the score type and drive number for the next score:
  find_next_score <- function(play_i, score_plays_i,pbp_df) {

    # Find the next score index for the current play
    # based on being the first next score index:
    next_score_i <- score_plays_i[which(score_plays_i >= play_i)[1]]

    # If next_score_i is NA (no more scores after current play)
    # or if the next score is in another half,
    # then return No_Score and the current drive number
    if (is.na(next_score_i) |
        (pbp_df$qtr[play_i] %in% c(1, 2) & pbp_df$qtr[next_score_i] %in% c(3, 4, 5)) |
        (pbp_df$qtr[play_i] %in% c(3, 4) & pbp_df$qtr[next_score_i] == 5)) {

          score_type <- "No_Score"

          # Make it the current play index
          score_drive <- pbp_df$Drive[play_i]

      # Else return the observed next score type and drive number:
    } else {

      # Store the score_drive number
      score_drive <- pbp_df$Drive[next_score_i]

      # Then check the play types to decide what to return
      # based on several types of cases for the next score:
```

```r
    # 1: Return TD
    if (identical(pbp_df$ReturnResult[next_score_i], "Touchdown")) {

      # For return touchdowns the current posteam would not have
      # possession at the time of return, so it's flipped:
      if (identical(pbp_df$posteam[play_i], pbp_df$posteam[next_score_i])) {

        score_type <- "Opp_Touchdown"

      } else {

        score_type <- "Touchdown"

      }
    } else if (identical(pbp_df$FieldGoalResult[next_score_i], "Good")) {

      # 2: Field Goal
      # Current posteam made FG
      if (identical(pbp_df$posteam[play_i], pbp_df$posteam[next_score_i])) {

        score_type <- "Field_Goal"

        # Opponent made FG
      } else {

        score_type <- "Opp_Field_Goal"

      }

      # 3: Touchdown (returns already counted for)
    } else if (pbp_df$Touchdown[next_score_i] == 1) {

      # Current posteam TD
      if (identical(pbp_df$posteam[play_i], pbp_df$posteam[next_score_i])) {

        score_type <- "Touchdown"

      # Opponent TD
      } else {

        score_type <- "Opp_Touchdown"

      }
      # 4: Safety (similar to returns)
    } else if (pbp_df$Safety[next_score_i] == 1) {

      if (identical(pbp_df$posteam[play_i],pbp_df$posteam[next_score_i])) {

        score_type <- "Opp_Safety"

      } else {

        score_type <- "Safety"
```

```r
      }
      # 5: Extra Points
    } else if (identical(pbp_df$ExPointResult[next_score_i], "Made")) {

      # Current posteam Extra Point
      if (identical(pbp_df$posteam[play_i], pbp_df$posteam[next_score_i])) {

        score_type <- "Extra_Point"

        # Opponent Extra Point
      } else {

        score_type <- "Opp_Extra_Point"

      }
      # 6: Two Point Conversions
    } else if (identical(pbp_df$TwoPointConv[next_score_i], "Success")) {

      # Current posteam Two Point Conversion
      if (identical(pbp_df$posteam[play_i], pbp_df$posteam[next_score_i])) {

        score_type <- "Two_Point_Conversion"

      # Opponent Two Point Conversion
      } else {

        score_type <- "Opp_Two_Point_Conversion"

      }

      # 7: Defensive Two Point (like returns)
    } else if (identical(pbp_df$DefTwoPoint[next_score_i], "Success")) {

      if (identical(pbp_df$posteam[play_i], pbp_df$posteam[next_score_i])) {

        score_type <- "Opp_Defensive_Two_Point"

      } else {

        score_type <- "Defensive_Two_Point"

      }

      # 8: Errors of some sort so return NA (but shouldn't take place)
    } else {

      score_type <- NA

    }
  }

  return(data.frame(Next_Score_Half = score_type,
                    Drive_Score_Half = score_drive))
```

```r
  }

  # Using lapply and then bind_rows is much faster than
  # using map_dfr() here:
  lapply(c(1:nrow(pbp_dataset)), find_next_score,
         score_plays_i = score_plays, pbp_df = pbp_dataset) %>%
    bind_rows() %>%
    return
}

# Apply to each game (ignore the warning messages here):
pbp_next_score_half <- map_dfr(unique(pbp_data$GameID),
                               function(x) {
                                 pbp_data %>%
                                   filter(GameID == x) %>%
                                   find_game_next_score_half()
                               })

# Join to the pbp_data:
pbp_data_next_score <- bind_cols(pbp_data, pbp_next_score_half)

# Create the EP model dataset that only includes plays with basic seven
# types of next scoring events along with the following play types:
# Field Goal, No Play, Pass, Punt, Run, Sack, Spike

pbp_ep_model_data <- pbp_data_next_score %>%
  filter(Next_Score_Half %in% c("Opp_Field_Goal", "Opp_Safety", "Opp_Touchdown",
                                "Field_Goal", "No_Score", "Safety", "Touchdown") &
        PlayType %in% c("Field Goal", "No Play", "Pass", "Punt", "Run", "Sack",
                        "Spike") & is.na(TwoPointConv) & is.na(ExPointResult) &
        !is.na(down) & !is.na(TimeSecs))

nrow(pbp_ep_model_data)
# 304805

# Now adjust and create the model variables:
pbp_ep_model_data <- pbp_ep_model_data %>%

         # Reference level should be No_Score:
  mutate(Next_Score_Half = fct_relevel(factor(Next_Score_Half), "No_Score"),

         # Create a variable that is time remaining until end of half:
         # (only working with up to 2016 data so can ignore 2017 time change)
         TimeSecs_Remaining = as.numeric(ifelse(qtr %in% c(1,2), TimeSecs - 1800,
                                         ifelse(qtr == 5, TimeSecs + 900,
                                                TimeSecs))),

         # log transform of yards to go and indicator for two minute warning:
         log_ydstogo = log(ydstogo),
         Under_TwoMinute_Warning = ifelse(TimeSecs_Remaining < 120, 1, 0),

         # Changing down into a factor variable:
         down = factor(down),
```

```r
        # Calculate the drive difference between the next score drive and the
        # current play drive:
        Drive_Score_Dist = Drive_Score_Half - Drive,

        # Create a weight column based on difference in drives between play and next score:
        Drive_Score_Dist_W = (max(Drive_Score_Dist) - Drive_Score_Dist) /
          (max(Drive_Score_Dist) - min(Drive_Score_Dist)),
        # Create a weight column based on score differential:
        ScoreDiff_W = (max(abs(ScoreDiff)) - abs(ScoreDiff)) /
          (max(abs(ScoreDiff)) - min(abs(ScoreDiff))),
        # Add these weights together and scale again:
        Total_W = Drive_Score_Dist_W + ScoreDiff_W,
        Total_W_Scaled = (Total_W - min(Total_W)) /
          (max(Total_W) - min(Total_W)))

ep_model <- nnet::multinom(Next_Score_Half ~ TimeSecs_Remaining + yrdline100 +
                           down + log_ydstogo + GoalToGo + log_ydstogo*down +
                           yrdline100*down + GoalToGo*log_ydstogo +
                           Under_TwoMinute_Warning, data = pbp_ep_model_data,
                           weights = Total_W_Scaled, maxit = 300)
```

Looking at the most popular expected points model, we can see that each row in the dataset used to build the model is attached to a scoring event. The reason people aren't able to create expected points maps is because each observation isn't independent. [reference?]

To see how the results would change

- So here I have a couple of thoughts re comparisons between NRL and NFL.
- Take the first play of every drive only so the map produced is 1st and 10 pass vs run
- Doing NRL the NFL way - What makes sense for the Total_W? next_score_half ~ x, y, tackle, weights=?
- Is the weights thing a better choice than one row per score.

Main selling message - maps of expected points across the field is sort of an obvious use case - pivot tables with a range more handy compared to point estimates - can't do it under current set up, can do it in new set up.

Things to work on - Exactly what makes the best comparison