Weighting Schema for Regression Based Ratings and Predictions in Austrialian Rules Football

S&DS 312 Final Project

Luke Benz

12/13/2018

1) Introduction

Sports fans, regardless of the sport, will always argue about why Team A is better than Team B or vice versa, usually contending that the team they support is the superior squad. Televesion pundits and so-called experts frequently release weekly power-rankings, a subjective ranking of all teams in a given sport league. Perhaps the best, or at the very least, the most objective way to rank teams is mathematically with some statistical model. Many approaches to the sports rating problem utilize some form of linear regression, and are similar in nature to the Massey Method (Massey 1997).

Most approaches to sports ratings, however, are slighlty more complex than ordinary least squares regression. Techniques such as penalized regression (Masarotto and Varin 2012)(Mease 2003) and weighted least squares regression (Chartier et al. 2011) have been previously applied to rating problems in various sports in order to improve the predictive nature of such ratings. Because many ratings problems treat game score differential as the response variable in question, penalized regression methods, like ridge or LASSO regression can help prevent one blowout from disportionately skewing estimates of team stength. Similarly, it make take time over the course of a season until a team plays up to its full strength. Thus, if the main objective of a certain set of sport ratings is to best predict the outcome of future games, it might make sense decrease the weight of earlier season games, as teams evolve over the course of the season and early season games may be less informative as the season goes on.

Various weighted least squares methods have been used to increase predictive accuracy in NCAA Men's College Basketball (Chartier et al. 2011) (Benz 2018) among other sports, but to date, no known methods have been (publically) published for Austrialian Rules Football (AFL). It remains to be seen whether such methods will yield similar success in increasing predictive accuracy in AFL ratings for a few reasons. To begin with, the AFL season consists of only 22 games, roughly 60% of the length of a college basketball season and just over 25% of the length of an 82-game NBA season. It makes sense to think that 70 games into an NBA season, a team's most recent 15-20 games are probably more indicative of its future performace than what happened more than 50 games prior. With many fewer observations available per team over the course of an AFL season, however, each game is potentially much more informative, and as such, decreasing the weight of early season games would perhaps result in worse predictive accuracy as one would be throwing away a lot of usual information. Moreover, (Lopez, Matthews, and Baumer 2018) showed that of all North American sports, the NBA is the least random. Given that American football (NFL) is less random than basketball, it's not much of a strech to assume that so too is Australian Rules Football. Perhaps there is too much noise in AFL scores for any weighting schema to be of much ultity.

This project seeks to examine the utility of various weighting schema for determining the best possible predictive ratings for Austrialian Rules Football (also known as "Footy") and to test whether such weights are even nessecessary to begin with. **Section 2** reviews the theory of weighted least squares regression. **Section 3** presents and analyzes several weighting schema for AFL ratings. **Section 4** runs a simultation to determine whether the order in which games are played has any bearing on predicting out of sample games, and concluding remarking are presented in **Section 5**.

2) Weighted Least Squares Regression

In traditional least squares regression, one assumes that some vector of responses $\boldsymbol{Y}$ can be modeled as

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

where $\epsilon_i \overset{\text{iid}}{\sim} N(0, \sigma^2)$. The above model is also used in weighted least squares regression, but the assumption of iid error, or homoscedasticity, no longer holds. The assumption of independent errors is still made, but one assumes that errors are distributed as follows:

$$\epsilon_i \sim N(0, \sigma_i^2)$$

In weighted least sqaures regression, each observation is given weight

$$w_i = \frac{1}{\sigma_i^2}$$

Thus, under the assumption independently distributed error, the matrix of weights, $\boldsymbol{W}$ is simply the inverse covariance matrix $\boldsymbol{\Sigma}^{-1}$. That is,

$$\boldsymbol{W} = \begin{pmatrix} w_1 & 0 & \dots & 0 \\ 0 & w_2 & \dots & \vdots \\ \vdots & \dots & \ddots & \vdots \\ 0 & \dots & 0 & w_n \end{pmatrix} = \begin{pmatrix} \frac{1}{\sigma_1^2} & 0 & \dots & 0 \\ 0 & \frac{1}{\sigma_2^2} & \dots & \vdots \\ \vdots & \dots & \ddots & \vdots \\ 0 & \dots & 0 & \frac{1}{\sigma_n^2} \end{pmatrix} = \boldsymbol{\Sigma}^{-1}$$

Of course, one rarely knows $\sigma_1^2, ..., \sigma_n^2$, so rather than using $\sigma_1^2, ..., \sigma_n^2$ to set the weights of observations, it is usually the case that one chooses weights $w_1, ..., w_n$ which fixes the corresponding $\sigma_1^2, ..., \sigma_n^2$.

Unlike the traditional least squares estimator $\widehat{\boldsymbol{\beta}}_{OLS}$, which minimizes the residual sum of squares, the weighted least squares estimator $\widehat{\boldsymbol{\beta}}_{WLS}$ minimizes the sum of squared residuals multiplied by the correponding observation weight ("Weighted Least Squares," n.d.). That is,

$$\widehat{\boldsymbol{\beta}}_{WLS} = \arg\min_{\boldsymbol{\beta}} \sum_{i=1}^{n} w_i (y_i - \widehat{y}_i)^2$$

Some basic calculus yields,

$$\widehat{\boldsymbol{\beta}}_{WLS} = (\boldsymbol{X}^T \boldsymbol{W} \boldsymbol{X})^{-1} \boldsymbol{X} \boldsymbol{W} \boldsymbol{Y}$$

3) Weighting Schema

3.1) Methodology Outline

Data for organized Australian Rules Football are avaiable for each season since 1897 via the `fitzRoy` R package (Day 2018). This analysis will restrict itself to seasons from 1970 to present, as the length of the regular season (22 games) has been constant since that time. Moreover, this analysis will only consider regular season games. Not every team makes the Grand Finals (playoffs) and so we will ensure that each team contributes to an equal number of games on which a particular model is trained. The following methodology will be used to evaluate different weighting schema:

1. Select the first 16 rounds of games for a given AFL season. Note that since the 2013 season, the AFL schedule has adopted a 23 round schedule in which each team plays 22 games over the course of 23 rounds, with a single bye week for rest. All byes happen prior to round 17, and as such, selecting the first 6 games of the season for each team amounts to selecting all games played in rounds 1-17.

2. Fit weighted least squares regression model using provided weights $w_1, ..., w_n$, where n denotes the number of games in the given training set. The model response is `score_differential` (team score - opponent score), while covariates included are:

- `team`
- `opponent`
- `location` (home/away)

3. Predict the `score_differential` for each "future" game in the final 6 rounds of the regular season on which the model was not trained. Note that if predicted `score_differtial` is positive, that is equivalent to predicting that `team` will beat `opponent`, and if predicted `score_differential` is negative, that is equivalent to predicting that `opponent` will beat 'team.
4. Compute the mean squared error in predicted minus observed `score_differential`.
5. Compute the misclassification rate in predicted game winners.
6. Repeat steps 1-5 for each season between 1970-2018.

Unlike traditional cross validation methods, the training set above is not chosen at random. This is done intentionally in the spirit of how the data are collected. In practice, one tries to predict the outcome of future games and wouldn't have access to the results of games played after the date of games they are trying to predict. That is, there is a specific and important ordering in which the data are observed, and it makes most sense from theoretical standpoint to adhere to such a natural ordering. **Section 4** examines the counterfactual, "what if the games were not ordered?".

Below, I load in the data from the `fitzRoy` R package and clean it for analysis.

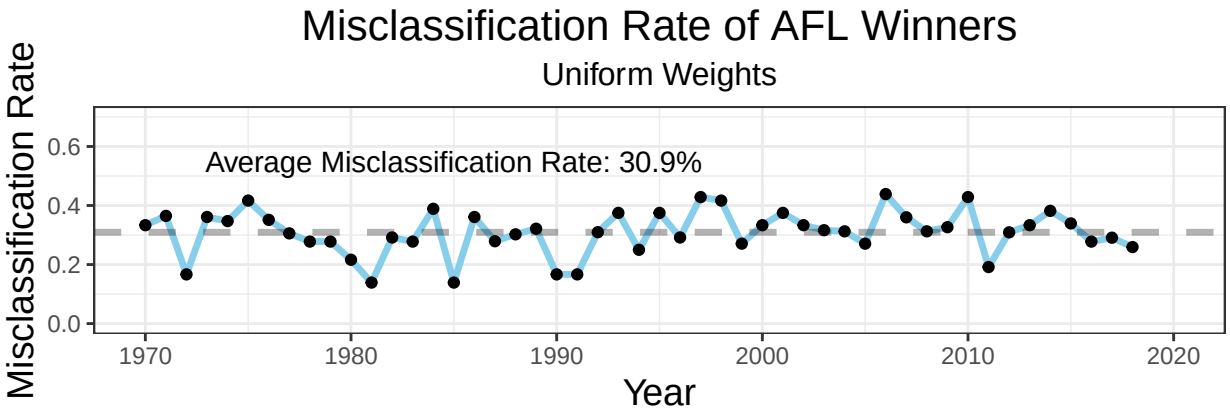
```
### Load Packages
library(dplyr)
library(tidyr)
library(ggplot2)
library(fitzRoy)
library(gridExtra)

### Pull and Clean AFL Data
results <- get_match_results()
results <- convert_results(results)
results <- filter(results, Season >= 1970, Round.Type == "Regular") %>%
  inner_join(select(results, Game, Team, Margin) %>% mutate(Margin = -Margin),
             by = c("Game", "Margin")) %>%
  rename("game_id" = Game, "date" = Date, "round" = Round,
         "venue" = Venue, "score_differential" = Margin, "season" = Season,
         "round_type" = Round.Type, "round_number" = Round.Number,
         "location" = Status, "behinds" = Behinds, "goals" = Goals,
         "points" = Points, "team" = Team.x, "opponent" = Team.y) %>%
  mutate(date = as.Date(date))
```

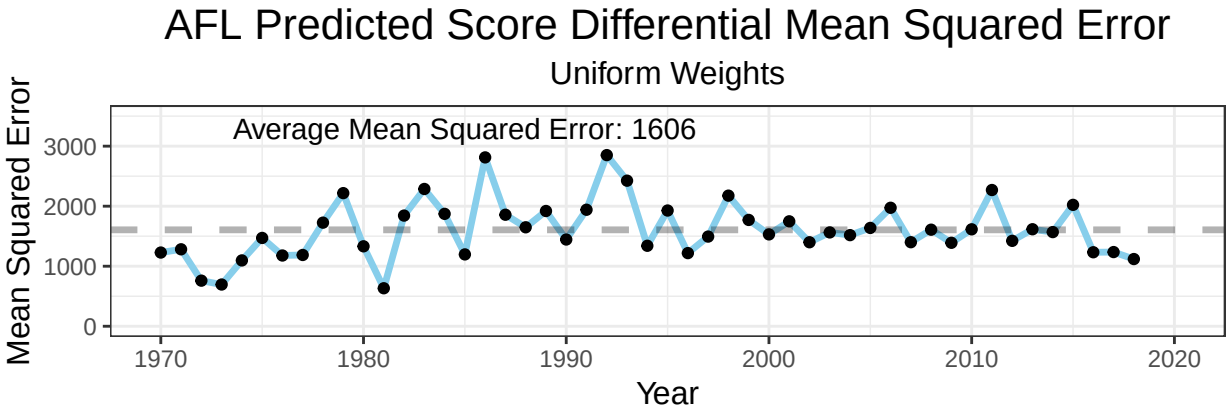
3.2) Uniform Weights

The first weighting schema we'll analyze is uniform weights, $w_i = 1$. This reduces weighted least squares regression to ordinary least squares regression, but it will be useful to establish a baseline against which to test other weighting schema.

```
for(year in 1970:2018) {
  ### Get data from given year and break into train/test sets
  year_data <- filter(results, season == year)
  train <- filter(year_data, round_number <= max(round_number) - 6) %>%
    mutate(weight = 1)
```

```
  test <- filter(year_data, round_number > max(round_number) - 6)

  ### Fit model with given weights
  lm.afl <- lm(score_differential ~ team + opponent + location,
               data  = train,
               weights = weight)

  ### Make test set predictions
  test <- mutate(test,
                 "pred_score_diff" = predict(lm.afl, newdata = test),
                 "method" = "uniform",
                 "error" = (pred_score_diff - score_differential)^2,
                 "misclassified" = sign(pred_score_diff) != sign(score_differential))

  ### Save (all) test sets
  if(year == 1970) {
    test_master <- test
  }else{
    test_master <- rbind(test_master, test)
  }
}
```





3.3) Linear Weights

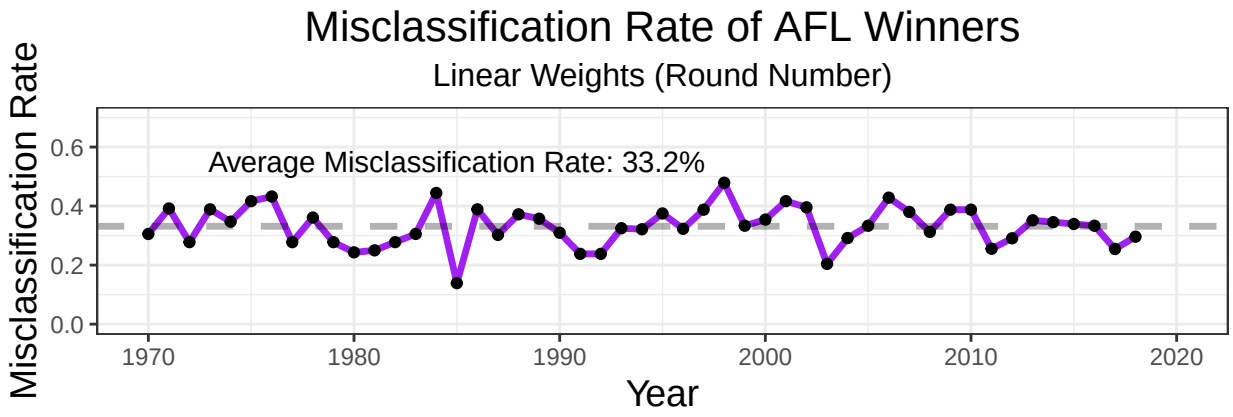
When creating futher weighting schema, there are two natural notions of time to consider. The first is `round_number` (i.e. how many games into the season a particular match is played), and the second is `date`.

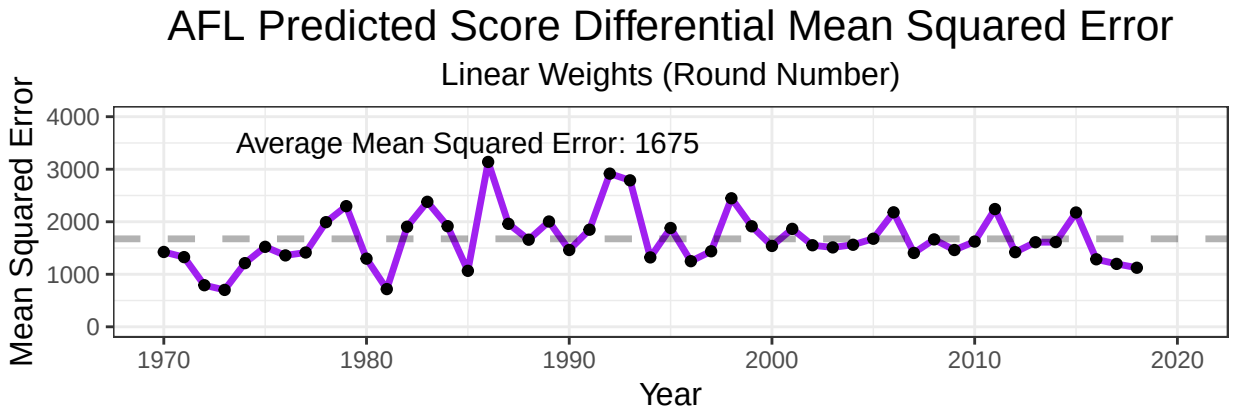
Each round of AFL matches is spread over the course of several days, and so a difference of 1 round can consist of anywhere between 6-10 days. In this section, we will examine linear weighting schema for both `round_number` and `date`. The first such schema sets $w_i = r_i$ where r_i denotes the `round_number` in which game i is played. The second such schema, modified from a schema utilized in college basketball by (Chartier et al. 2011) sets

$$w_i = \frac{t_i - t_0 + 1}{t_f - t_0 + 1}$$

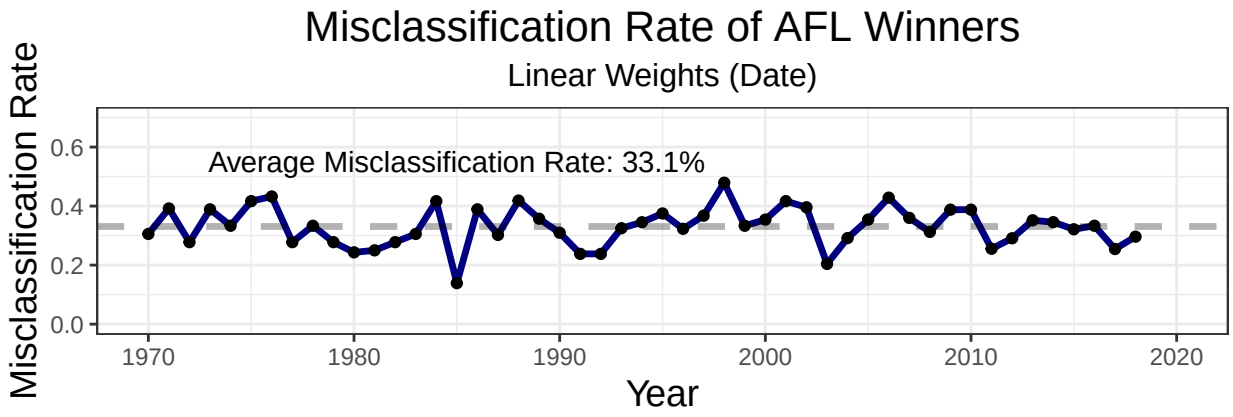
here t_i denotes the `date` of game i, t_0 denotes the `date` of the first game of the season, and t_f denotes the `date` of the final game of the season.

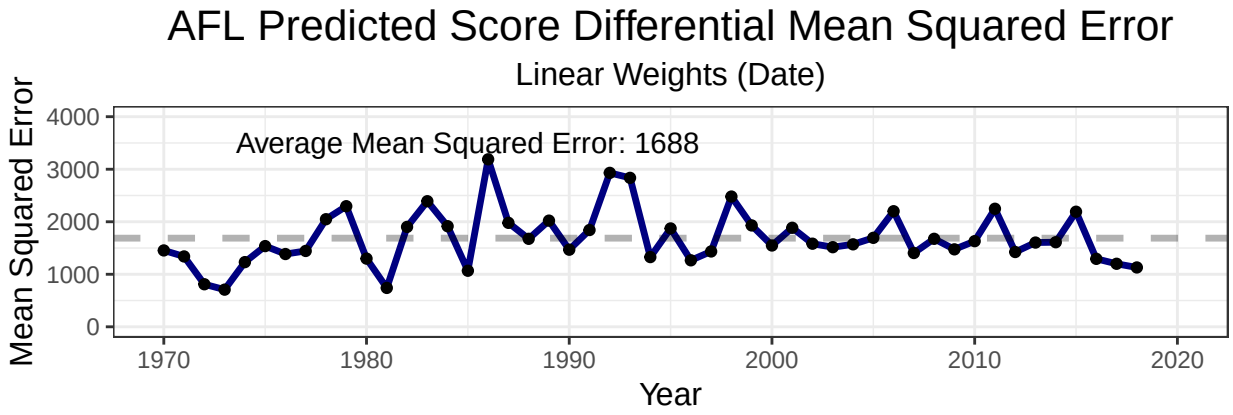
```
### Linear Weights (Round Number)
for(year in 1970:2018) {
  ### Get data from given year and break into train/test sets
  year_data <- filter(results, season == year)
  train <- filter(year_data, round_number <= max(round_number) - 6) %>%
    mutate(weight = round_number)
  test <- filter(year_data, round_number > max(round_number) - 6)

  ### Fit model with given weights
  lm.afl <- lm(score_differential ~ team + opponent + location,
               data  = train,
               weights = weight)

  ### Make test set predictions
  test <- mutate(test,
                 "pred_score_diff" = predict(lm.afl, newdata = test),
                 "method" = "round_number_linear",
                 "error" = (pred_score_diff - score_differential)^2,
                 "misclassified" = sign(pred_score_diff) != sign(score_differential))

  ### Save (all) test sets
  test_master <- rbind(test_master, test)

}
```






```
### Linear Weights (Date)
for(year in 1970:2018) {
  ### Get data from given year and break into train/test sets
  year_data <- filter(results, season == year)
  train <- filter(year_data, round_number <= max(round_number) - 6) %>%
    mutate(weight = as.numeric((date - min(date) + 1))/
             as.numeric((max(date) - min(date) + 1)))
  test <- filter(year_data, round_number > max(round_number) - 6)

  ### Fit model with given weights
  lm.afl <- lm(score_differential ~ team + opponent + location,
               data  = train,
               weights = weight)

  ### Make test set predictions
  test <- mutate(test,
                 "pred_score_diff" = predict(lm.afl, newdata = test),
                 "method" = "date_linear",
                 "error" = (pred_score_diff - score_differential)^2,
                 "misclassified" = sign(pred_score_diff) != sign(score_differential))

  ### Save (all) test sets
  test_master <- rbind(test_master, test)
}
```



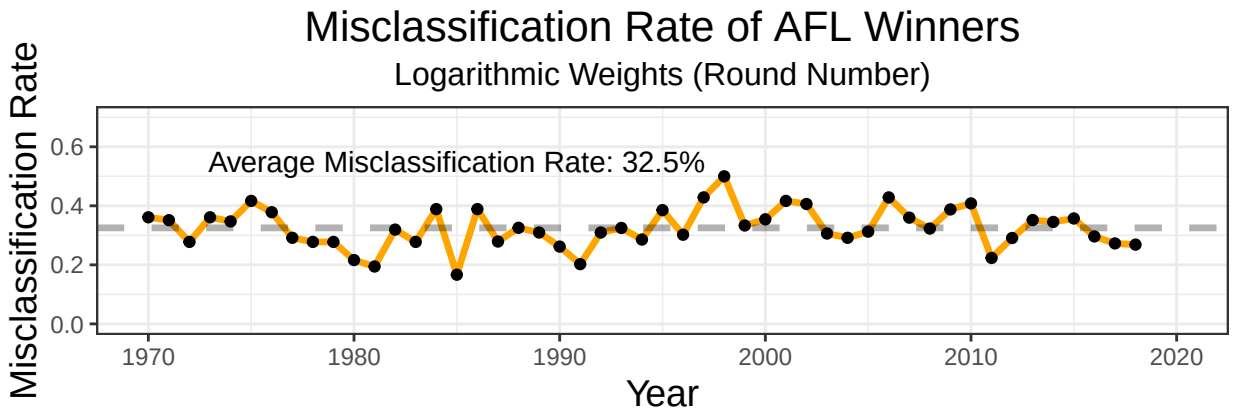


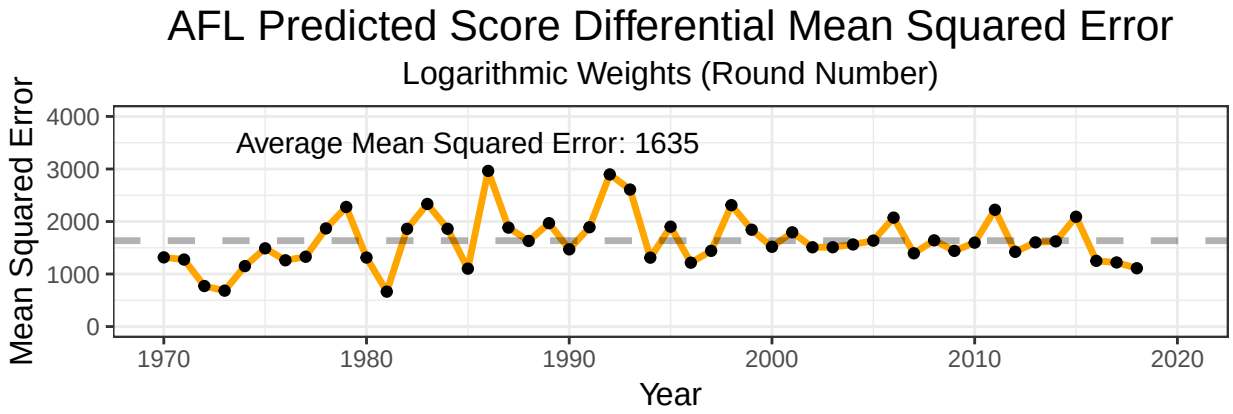
3.4) Logarithmic Weights

We see that these linear weighting schema for both `date` and `round_number` perform worse than the uniform weighting schema. This is likely because these schema treat games in round 16 as 16 times more important than games in round 1. It's reasonable to believe that more recent games should be treated as more imporant than games played earlier in the season, but perhaps a 16:1 ratio is too much. Perhaps a logarithmic weighting schema will perform better, as it weights recent games more heavily but not too heavily. That is, we set $w_i = \log(r_i + 1)$ where r_i denotes the `round_number` in which game i was played. Note that as shown in **Section 3.3** (and in additional experimentation not presented in this report), `date` and `round_number` weighting schema perform quite similar. Thus, in the interest of space and avoiding redundancy, I'll only present logarithmic weights for `round_number`.

```
### Log Weights (Round Number)
for(year in 1970:2018) {
  ### Get data from given year and break into train/test sets
  year_data <- filter(results, season == year)
  train <- filter(year_data, round_number <= max(round_number) - 6) %>%
    mutate(weight = log(round_number + 1))
  test <- filter(year_data, round_number > max(round_number) - 6)

  ### Fit model with given weights
  lm.afl <- lm(score_differential ~ team + opponent + location,
               data  = train,
               weights = weight)

  ### Make test set predictions
```

```
  test <- mutate(test,
                 "pred_score_diff" = predict(lm.afl, newdata = test),
                 "method" = "round_number_log",
                 "error" = (pred_score_diff - score_differential)^2,
                 "misclassified" = sign(pred_score_diff) != sign(score_differential))

  ### Save (all) test sets
  test_master <- rbind(test_master, test)

}
```





3.5) Sigmoidal Weights

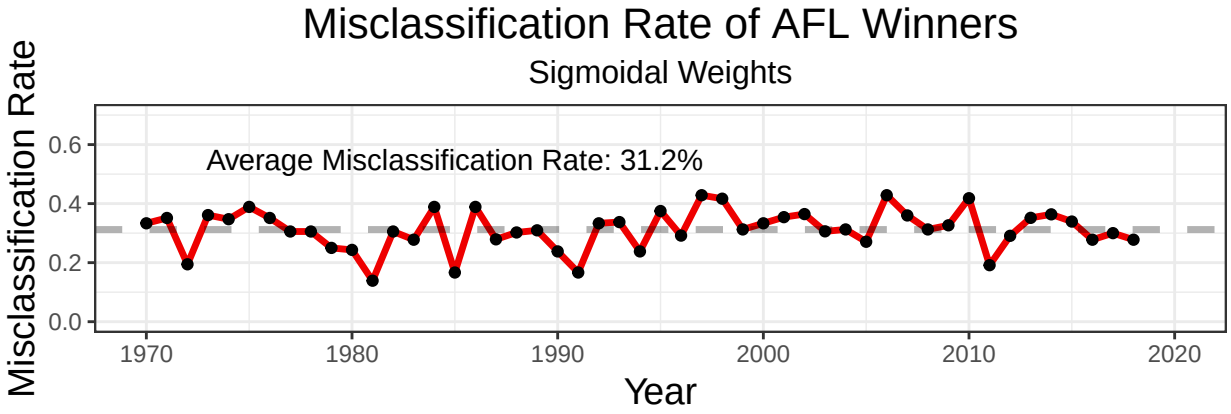
Another weighting schema to try is modified sigmoidal weights. Specifically, we try a variation of the weighting scheme used in (Benz 2018) for college basketball. For this weighting scheme, we set

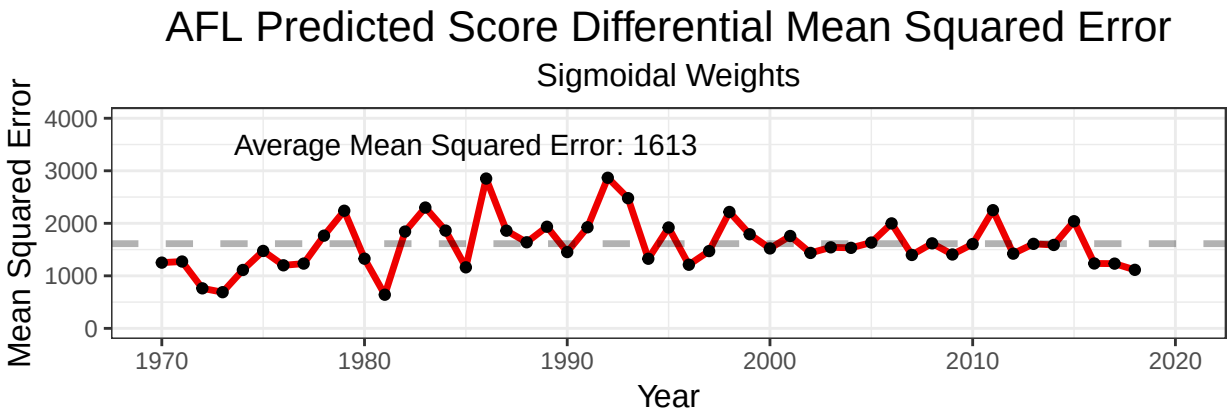
$$w_i = \frac{1}{1 + 0.5^{\sqrt{r_i}} \times e^{-r_i/16}}$$

where r_i denotes the round in which game i is played. This system of weights has the properties that $0.5 < w_i < 1$, and the weight increase in a concave down manner. That is, games very early in the season are downweighted but there isn't much difference.

```
### Sigmoidal Weights
for(year in 1970:2018) {
  ### Get data from given year and break into train/test sets
  year_data <- filter(results, season == year)
```

```r
  train <- filter(year_data, round_number <= max(round_number) - 6) %>%
    mutate(weight = 1/(1 + 0.5^sqrt(round_number) * exp(-round_number/(max(round_number) - 6))))
  test <- filter(year_data, round_number > max(round_number) - 6)

  ### Fit model with given weights
  lm.afl <- lm(score_differential ~ team + opponent + location,
               data  = train,
               weights = weight)

  ### Make test set predictions
  test <- mutate(test,
                 "pred_score_diff" = predict(lm.afl, newdata = test),
                 "method" = "sigmoid",
                 "error" = (pred_score_diff - score_differential)^2,
                 "misclassified" = sign(pred_score_diff) != sign(score_differential))

  ### Save (all) test sets
  test_master <- rbind(test_master, test)

}
```





3.6) Estimating σ_i^2

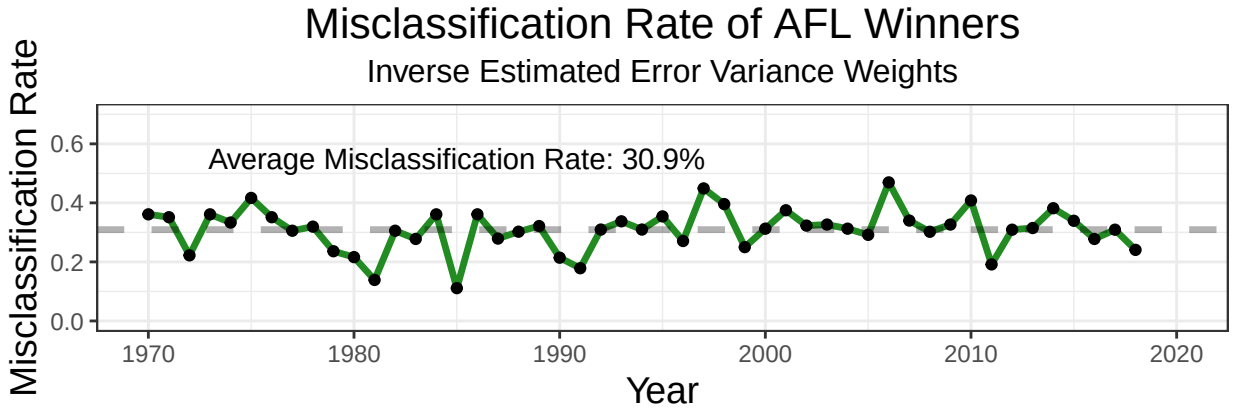
Recall from **Section 2** that $w_i = \frac{1}{\sigma_i^2}$ where $\epsilon_i \sim N(0, \sigma_i^2)$. While in practice it's often impossible in practice to know σ_i^2, a good estimate of σ_i^2 may provide a useful weighting scheme. We'll use the following steps to

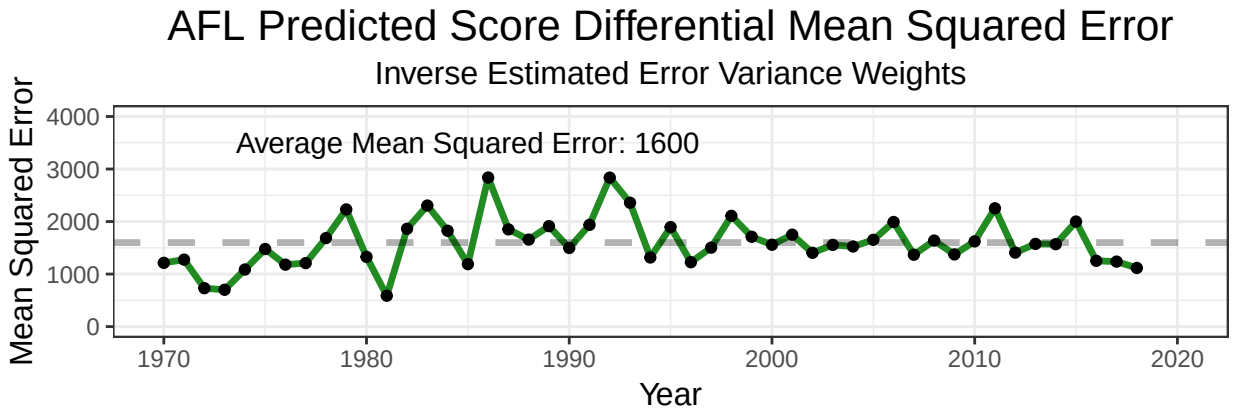
compute $\widehat{\sigma}_i^2$:

1. For a given year's set of games (all games per team), fit linear model with uniform weights, $w_i = 1$, as done in **Section 3.2**.
2. Predict the score differential for each training set game in the given year.
3. Compute the error for game i, $y_i - \widehat{y}_i$ (`score_differential - pred_score_diff`)
4. For each team, compute the variance of the errors of games in which the given team plays. Call such quantity $\widehat{\sigma}_{team}^2$
5. Estimate $\widehat{\sigma}_i^2 = \widehat{\sigma}_{teamA}^2 + \widehat{\sigma}_{teamB}^2$ where team A plays team B in game i.
6. Set
$$w_i = \frac{1}{\widehat{\sigma}_i^2} = \frac{1}{\widehat{\sigma}_{teamA}^2 + \widehat{\sigma}_{teamB}^2}$$
The intuition behind this weighting scheme is that the higher the variance in the prediction error of a team's games made using ordinary least squares regression, the less reliable any game in which that team plays is in predicting future games.

```
### Estimated Error Variance Weights

### Get Fitted Values by Year
for(year in 1970:2018) {
  train <- filter(results, season == year, round_number <= max(round_number) - 6)
  lm.afl <-   lm.afl <- lm(score_differential ~ team + opponent + location,
                           data  = train)

  ### Save predictions
  train <- mutate(train,
                  "pred_score_diff" = predict(lm.afl, newdata = train),
                  "method" = "variance_estimation",
                  "error" = (pred_score_diff - score_differential)^2,
                  "misclassified" = sign(pred_score_diff) != sign(score_differential))

  if(year == 1970) {
    train_master <- train
  }else{
    train_master <- rbind(train_master, train)
  }
}

### Estimate Team Sigma^2
sigmas <-
  group_by(train_master, team, season) %>%
  summarise("sigma_2" = var(score_differential - pred_score_diff)) %>%
  ungroup()

### Now make prediction
for(year in 1970:2018) {
  ### Get data from given year and break into train/test sets
  year_data <- filter(results, season == year)
  train <- filter(year_data, round_number <= max(round_number) - 6) %>%
    inner_join(sigmas, by = c("team", "season")) %>%
    inner_join(sigmas, by = c("opponent" = "team",
                              "season" = "season"), suffix = c("_team", "_opp")) %>%
    mutate(weight = 1/(sigma_2_team + sigma_2_opp))
```

```
  test <- filter(year_data, round_number > max(round_number) - 6)

  ### Fit model with given weights
  lm.afl <- lm(score_differential ~ team + opponent + location,
               data  = train,
               weights = weight)

  ### Make test set predictions
  test <- mutate(test,
                 "pred_score_diff" = predict(lm.afl, newdata = test),
                 "method" = "variance_estimation",
                 "error" = (pred_score_diff - score_differential)^2,
                 "misclassified" = sign(pred_score_diff) != sign(score_differential))

  ### Save (all) test sets
  test_master <- rbind(test_master, test)

}
```





3.7) Multiyear Weighting

Thus far, each weighting scheme has utiltized on games for the current season on which we seek to predict. This only allows a given model to be trained on 16 games per team over the course of the season. While a team's strength changes from year to year, the majority of changes are gradual. That is, very rarely does a team go from the best team in the league to the worst team the next year, or vice versa. Hence, there is likely some information one can glean about the current season by looking at the past season's results, and

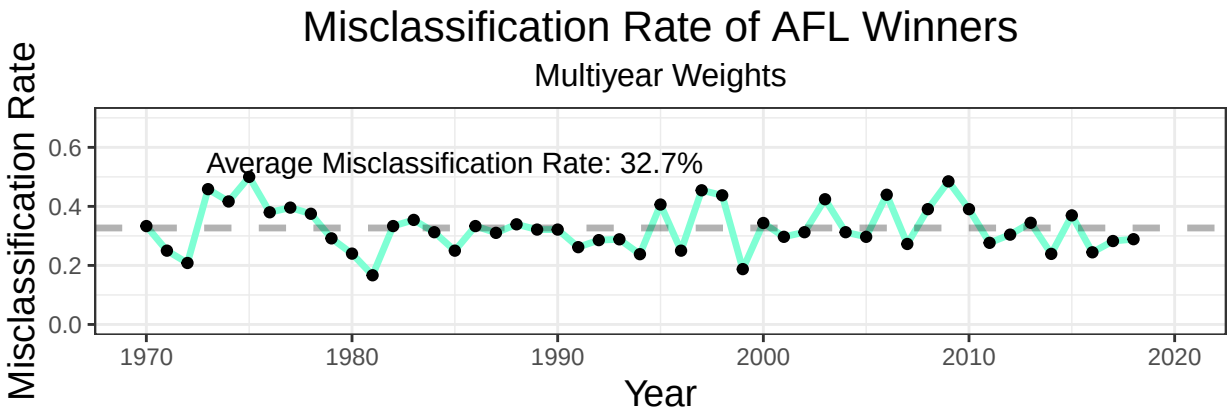
weighting them appropriately relative to games in the current season we are examining. For predicting the result of games in season j, we set the following weghts to

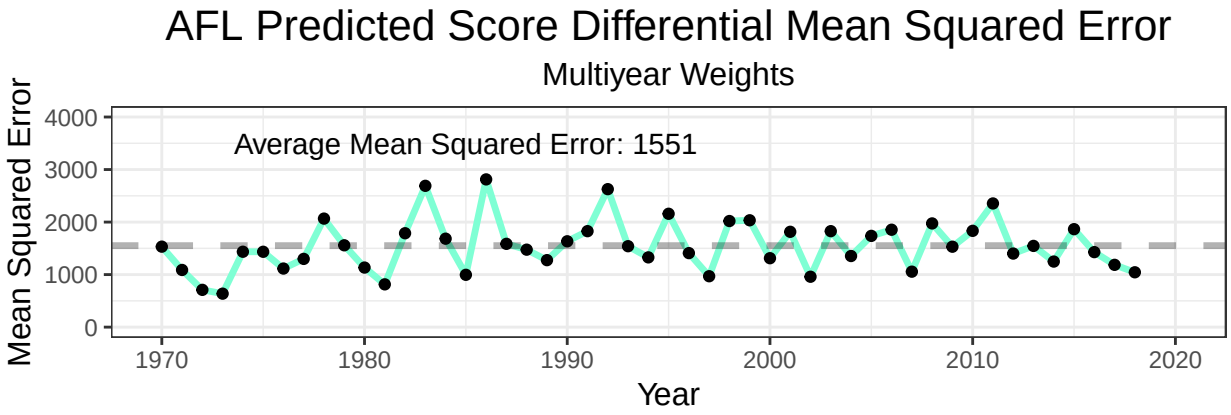
$$w_i = \frac{1}{e^{j-s_i} \times (\widehat{\sigma}^2_{teamA,s_i} + \widehat{\sigma}^2_{teamB,s_i})}$$

where $s_i \in \{j-1, j\}$ denotes the season in which game i was played between team A and team B.

```
### Get Fitted Values by Year
for(year in 1970:2018) {
  ### Make Predictions on Training Set for Given Year
  train <- filter(results, season == year, round_number <= max(round_number) - 6)
  lm.afl <-   lm.afl <- lm(score_differential ~ team + opponent + location,
                           data  = train)

  ### Save predictions
  train <- mutate(train,
                  "pred_score_diff" = predict(lm.afl, newdata = train),
                  "method" = "variance_estimation",
                  "error" = (pred_score_diff - score_differential)^2,
                  "misclassified" = sign(pred_score_diff) != sign(score_differential),
                  "sigma_type" = "short")


  full <- filter(results, season == year, round_number)
  lm.afl <-   lm.afl <- lm(score_differential ~ team + opponent + location,
                           data  = full)

  ### Save predictions
  full <- mutate(full,
                 "pred_score_diff" = predict(lm.afl, newdata = full),
                 "method" = "variance_estimation",
                 "error" = (pred_score_diff - score_differential)^2,
                 "misclassified" = sign(pred_score_diff) != sign(score_differential),
                 "sigma_type" = "long")

  if(year == 1970) {
    train_master <- rbind(train, full)
  }else{
    train_master <- rbind(train_master, train, full)
  }
}

### Estimate Team Sigma^2
sigmas <-
  group_by(train_master, team, season, sigma_type) %>%
  summarise("sigma_2" = var(score_differential - pred_score_diff)) %>%
  ungroup()

for(year in 1970:2018) {
  ### Get data from given year and break into train/test sets
  train <- filter(results, (round_number <= max(round_number) - 6 & season == year) |
                    season %in% c(year - 1)) %>%
    inner_join(filter(sigmas, sigma_type == "short"), by = c("team", "season")) %>%
    inner_join(filter(sigmas, sigma_type == "long"), by = c("opponent" = "team",
```

```
                                                    "season" = "season"),
              suffix = c("_team", "_opp")) %>%
    mutate(weight = exp(season - year)/(sigma_2_team + sigma_2_opp))

  test <- filter(results, round_number > max(round_number) - 6, season == year)

  ### Fit model with given weights
  lm.afl <- lm(score_differential ~ team + opponent + location,
               data  = train,
               weights = weight)

  ### Make test set predictions
  test <- mutate(test,
                 "pred_score_diff" = predict(lm.afl, newdata = test),
                 "method" = "multiyear",
                 "error" = (pred_score_diff - score_differential)^2,
                 "misclassified" = sign(pred_score_diff) != sign(score_differential))

  ### Save (all) test sets
  test_master <- rbind(test_master, test)

}
```





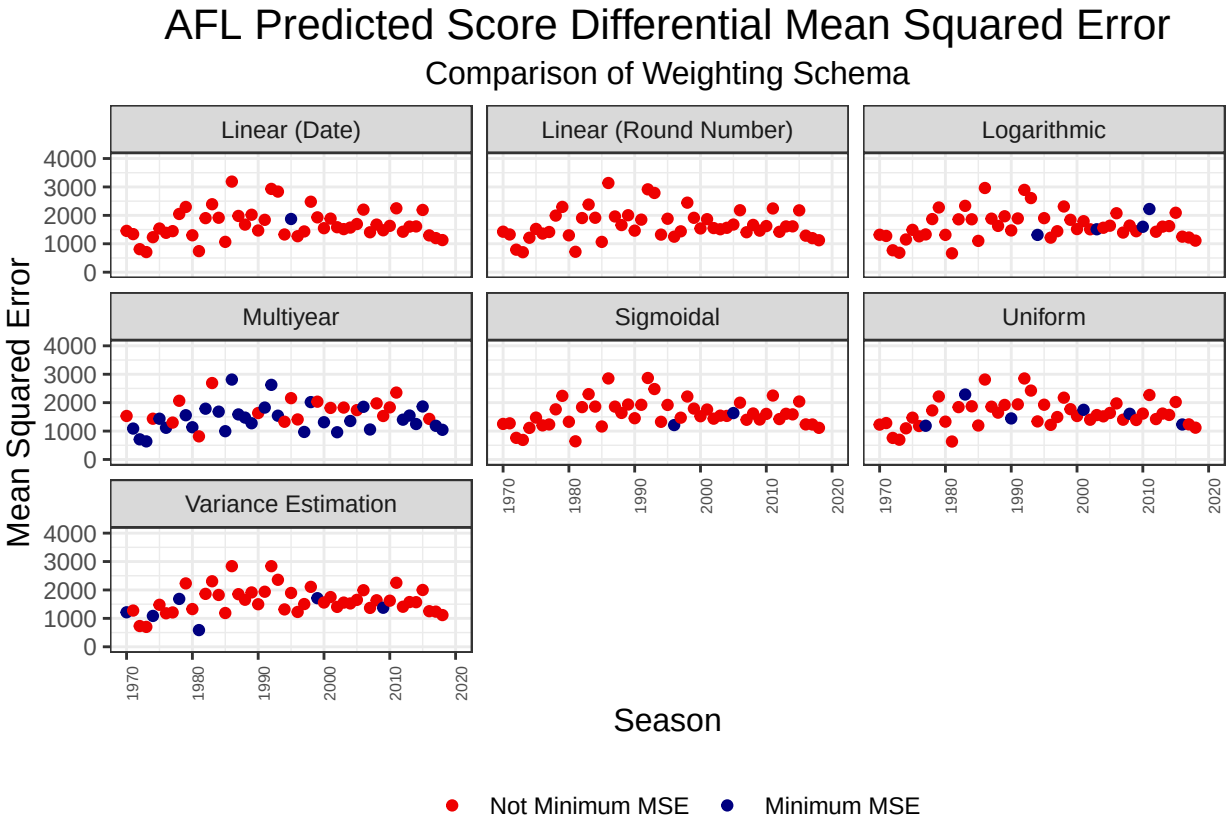
Weighting Method	Average MSE	Average Misclassification Rate	Lowest MSE	Lowest Misclassification Rate
Multiyear	1551	0.3269	30	12
Variance Estimation	1600	0.3093	6	18
Uniform	1606	0.3094	6	14
Sigmoidal	1613	0.3124	2	15
Logarithmic	1635	0.3254	4	9
Linear (Round Number)	1675	0.3316	0	10
Linear (Date)	1688	0.3308	1	13

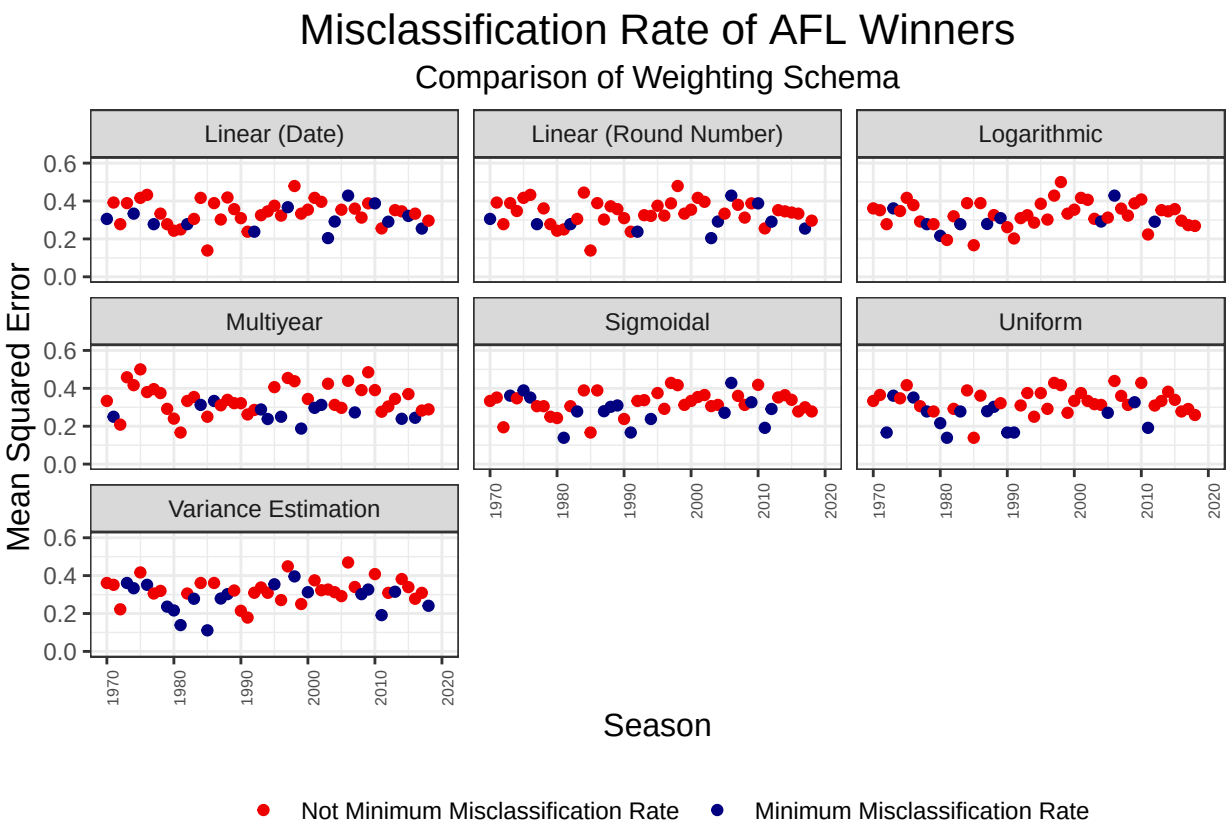
3.8) Analysis of Results

The below table contains aveage mean squared and average misclassification rates for each of the 7 weighting schema presented in this report. Additionally, the table also displays counts of the number of times each weighting method yielded the minimum mean squared error or misclassifcation rate in a given year. Note that the number of minimum misclassification rate sums to a total greater than the 49 seasons examined in this analysis, as many times two or more methods had the exact same misclassification rate in a particular year.

We see that sigmoidal, logarithmic, and both linear weighting schemes perform worse than the uniform weighting scheme. Both methods that involved estimating $\widehat{\sigma}_i^2$ outperform the baseline uniform weighting scheme. It seems that variance estimation over the course of two season (the multiyear weighting scheme) is the best for predicting `score_differential` of future matches. There is, however, a tradeoff in best predicting the winners of matches, and if one solely cares about predicting the winners of matching, a single year variance estimation is preferable. Despite tha fact that the uniform weighting scheme and the variance estimation weighting scheme had similar performance of the basis of both MSE and winner misclassification rate, the variance estimation scheme had lower misclassification rate more consistently than the uniform weighting scheme.

Additional experimentation not shown in this report due to space constraints suggest that the improved performance in the multiyear weighting scheme has a lot to do with its variance estimation component. With the multiyear year scheme, one is able to obtain more stable estimates of $\widehat{\sigma}_{team}^2$ over the course of 22 games per team as opposed to 16 games per team. Worse results were obtained with the variance estimation component was removed. Increasing the timeframe of prior data doesn't improve the weighting scheme for two reasons. To begin with, team strength can change a lot more over the course of 2-3 years than it can from season to season. Treating game results 2-3 years as equal to current results doesn't reflect current team strength, therby skewing predictions. In order to apply enough of a time-deacying component to remove such skew, one ends up setting weights of anything further back than a single year to weight ≈ 0.


AFL Predicted Score Differential Mean Squared Error
Comparison of Weighting Schema
Linear (Date)
Linear (Round Number)
Logarithmic
Multiyear
Sigmoidal
Uniform
Variance Estimation
Mean Squared Error
0
1000
2000
3000
4000
1970
1980
1990
2000
2010
2020
Season
Not Minimum MSE
Minimum MSE

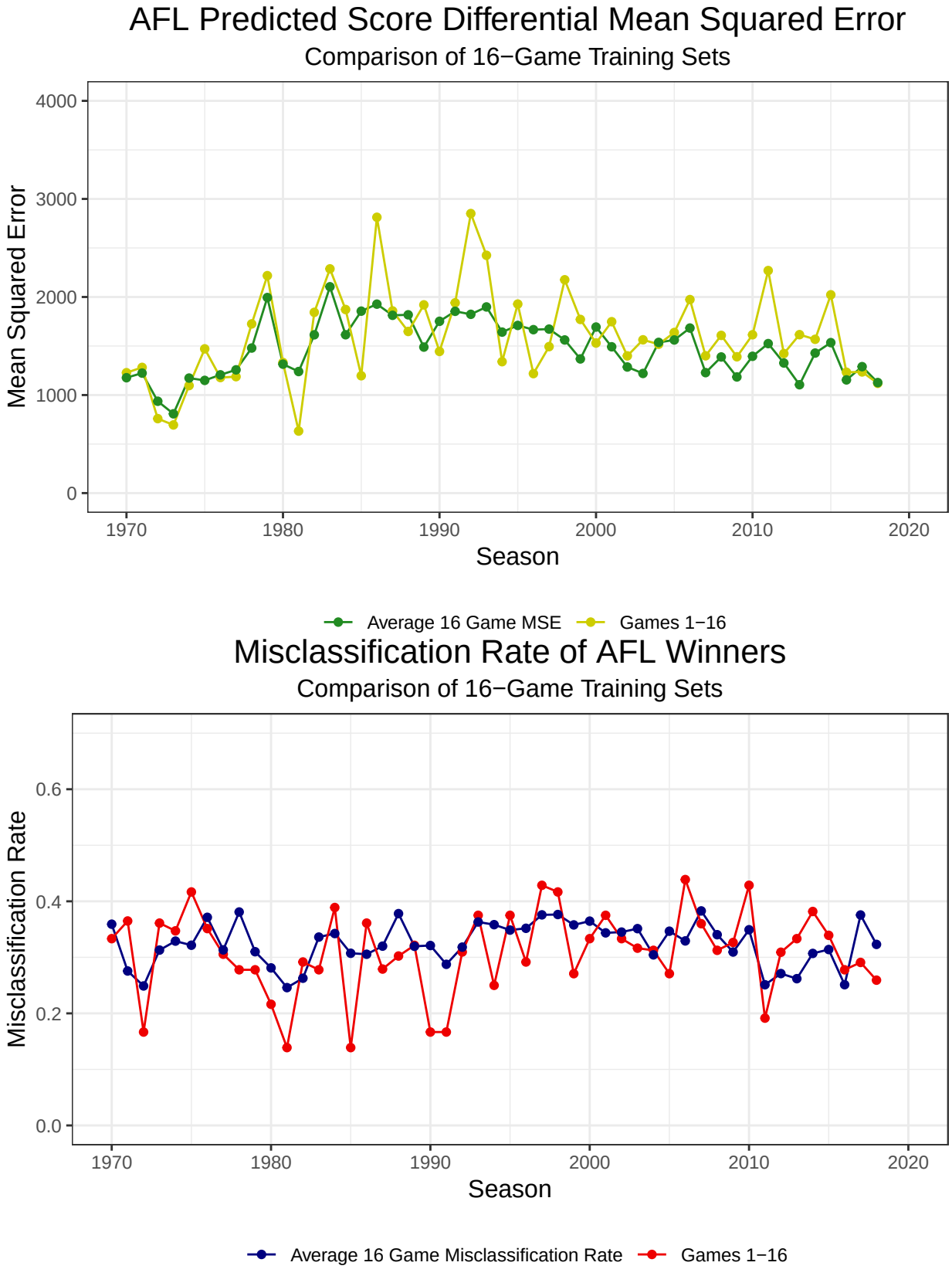

Misclassification Rate of AFL Winners
Comparison of Weighting Schema
Linear (Date)
Linear (Round Number)
Logarithmic
Multiyear
Sigmoidal
Uniform
Variance Estimation
Mean Squared Error
0.0
0.2
0.4
0.6
1970
1980
1990
2000
2010
2020
Season
Not Minimum Misclassification Rate
Minimum Misclassification Rate

4) Does the Order of Games Matter? A Simulation Study

Different weighting schema were utilized throughout this report in an attempt to increase the accuracy of predictions for AFL games. The primary reason that one would consider implementing some form of weighted least squares regression in the first place if there was reason to believe that team strength evolved over the course of the season. But does the order in which games are played even matter? That is, does knowing the results of the first 16 games of the season per team allow us to predict the next 6 held out games per team with higher accuracy than if we simply knew the results of 16 random rounds and used it to predict the results 6 random held out rounds? I devise the following simulation study to test the results.

1. For each given season select 16 rounds worth of games at random to be the training set. (For years 2013-18, we select 17 rounds at random as each team has a bye week). Note that rounds are chosen at random rather than individual games in order to ensure each team has an equal number of games in the training set.
2. Fit an ordinary least squares regression model on the training set (as in **Section 3.2**).
3. Predict the outcome of test set games.
4. Repeat 500 times for each season and get the average misclassification rate and mean squared error for each season, to compare with the results of **Section 3.2**.

```
### Ordering Simulation Study
sim_results <- data.frame("year" = 1970:2018,
                          "avg_mse" = NA,
                          "avg_misclass" = NA)
set.seed(73097)
for(year in 1970:2018) {
  year_data <- filter(results, season == year)
  rounds <- unique(year_data$round_number)
  mse <- rep(NA, 500)
  misclass <- rep(NA, 500)
  for(i in 1:500) {
    ### Select Random Rounds
    test_rounds <- sample(rounds, 6)
    train_rounds <- setdiff(rounds, test_rounds)

    ### Create Training and Test Set
    train <- filter(year_data, round_number %in% train_rounds)
    test <- filter(year_data, round_number %in% test_rounds)

    ### Fit Linear Model
    lm.afl <- lm(score_differential ~ team + opponent + location, data = train)

    ### Predict On Test Set
    test <- mutate(test,
                   "pred_score_diff" = predict(lm.afl, newdata = test),
                   "error" = (pred_score_diff - score_differential)^2,
                   "misclassified" = sign(pred_score_diff) != sign(score_differential))

    ### Get MSE and Misclassification Rate
    mse[i] <- mean(test$error)
    misclass[i] <- mean(test$misclassified)
  }
  sim_results$avg_mse[year - 1969] <- mean(mse)
  sim_results$avg_misclass[year - 1969] <- mean(misclass)
}
```



We observe that using games 1-16 to predict future games can yield better or worse results in both misclassification rate of winners than an average 16 game training setand mean squared error depending on the year.

What stands out most is that there is a much higher year to year variance in the MSE and misclassification rate of a particular 16 game set than the average 16 game training set. This suggests that ordering of games doesn't particiarly matter when predicting held out games. Of course, in practice, one will observe games in a particular ordering and as such while have to make due with the data available at the time of prediction, but these results offer further evidence that time-based weights are not of use with in a single AFL season.

5) Conclusion

This project has analyzed several different weighting schema for regression based ratings in Australian Rules Football. The results presented in **Section 3.8** suggest that multiyear weightings which also utilize weights inversly proportional to the average of estimations of team error variance yield the best rating system on the basis of mean squared prediction error. Most notably, time based weighting schema in terms of both `round_number` and `date` proved to be inferior than a simple uniform weighting scheme. The results of the simulation study in **Section 4** back this up; which 16 rounds of games are included in the training set doesn't significantly change MSE or misclassification rate, suggesting that the evoluation of a team's strength over the course of an AFL season is relatively constant over the course of the season.

Time based weightings that work well in sports like college basketball and the NBA don't seem to work well in Australian Rules Football for several reasons. To begin with, the AFL season is much much shorter than that over every North American sport besides the NFL. Moreover, the results of **Section 4** suggest that results in the AFL have a higher variance than those in certain other North American sports. (Lopez, Matthews, and Baumer 2018) rank the four major North American sports leagues in order of decreasing variance as follows:

1. MLB (Baseball)
2. NHL (Hockey)
3. NFL (Football)
4. NBA (Basketball)

I would hypothesize that AFL would fit into the above list between the NHL and NFL. That is to say, the AFL has significantly higher variance than American basketball and likely has higher/similar variance to that of American football. Reasons for such increased variance are likely as follows.

1. With only 18 teams, the AFL less than $\frac{2}{3}$ the size of any of the four major North American sports leagues. As such, in the same time frame, one observes many fewer games then they would in a different league.
2. There are two ways a team can score points: goals and behinds. A goal occurs when a player kicks the ball through the middle two posts among a set of 4 posts, and is worth 6 points. A behind occurs when a player kicks the ball through either of the outer pairs of posts and is worth a single point. While it's likely that the best teams are able to maintain higher goal to behind ratios then their opponents over the course a season, a few mis-hit kicks over the course of a single game can yield a huge difference in `score_differential`.
3. Games tend to be very high scoring compared with North American sports.

Given the seemingly high variance of AFL results, it makes sense that we are able to obtain good results by taking into account multiyear estimates of team variance and weighting games as inversely proportional to such estimates. After all, the method introduced in **Section 3.6** and expanded in **Section 3.7**, whereby our linear model is fit with weights $w_i = 1$ and then later updated is essentially the first set in an iteratively reweighted least squares procedure. My guess as to why even a single iteration of some iteratively reweighted least squares procedure is successful is that it acts as a variance reduction technique. Even slightly reducing variance can yield better prediction accuracy. Prediction in a higher variance sport like Australian Rules Football is certainly tricker than my experience building models for lower variance sports, like basketball and soccer. Thus, this work has shown that weighting schema which work well for one sport need not work well for another sport, and which weighting schema perform well depends a lot on the variability underlying the sport.

While the weighting scheme presented in **Section 3.7** performed best among methods tested in this project, I am by no means suggesting that is the best weighting scheme possible, nor claiming that weighted linear regression is the best way to model AFL games. Other methods, including those methods developed by (Corke 2018), would likely outperform the best model presented here. Moreover, all we are including in the model is the teams playing and the game location. Adding additional covariates might very well improve the quality of our model. That being said, I think the model presented here does quite well for it's simplificty and serves its purpose in learning about weighting schema and variance in Australian Rules Football.

5.1) Future Work

Using the results of this project, I hope to build out a comprehensive prediction model for Austrialian Rules Football and go toe to toe with some of the best publically avaible AFL models. I'd also love to build an in game win probability model for AFL, allowing me to estimate team's chances of winning a particular over the course of said game. Finally, I hope to examine how the AFL's unique playoff structure helps protect top seeds, and how such a tournament format might be applied to college basketball conference tournaments.

References

Benz, Luke. 2018. "NCAA Hoops Model Methodology." http://rpubs.com/lbenz730/ncaa_hoops_methodology.

Chartier, Timothy, Erich Kreutzer, Amy Langville, and Katherine Pedings. 2011. "Sports Ranking with Nonuniform Weighting." *Jornal of Quantitative Analysis in Sports* 7 (3).

Corke, Tony. 2018. "Matter of Stats." http://www.matterofstats.com/mafl-wagers-and-tips/2018-team-ratings-after-round-26.

Day, James. 2018. "FitzRoy: Easily Scrape Afl Data." https://github.com/jimmyday12/fitzRoy.

Lopez, Michael, Gregory Matthews, and Benjamin Baumer. 2018. "How Often Does the Best Team Win? A Unified Approach to Understanding Randomness in North American Sport." *Annals of Applied Statistics* 12 (4): 2483–2516.

Masarotto, Guido, and Cristiano Varin. 2012. "The Ranking Lasso and Its Application to Sports Tournaments." *The Annals of Applied Statistics* 6 (4): 1949–70.

Massey, Kenneth. 1997. "Statistical Models Applied to the Rating of Sports Teams." *Honors Theis, Bluefield College*.

Mease, David. 2003. "A Penalized Maximum Likelihood Approach for the Ranking of College Football Teams Independent of Victory Margin." *The American Statistician* 57 (4): 241–48.

"Weighted Least Squares." n.d. https://onlinecourses.science.psu.edu/stat501/node/352/.