HW2

Narek Sahakyan

12, October 2019

```
data("f_data_sm")
data("nba2009_2018")
nba2009_2018 <- nba2009_2018 %>%
  filter(home.WL %in% c("W","L"))
# removing indiana pacers cancelled game
# We don't have draw games in the data set so I did not include D
```

Winning Percentage:

Narek Sahakyan - La Liga Primera Division

2. library SportsAnalytics270 has a function final_table. It creates final league standing for the season. Use this function to create a dataframe with final standings of all seasons of your league. Combine seasons into 1 datafrmae. You need to get something like nba_east. You can use for loop here.

```
final_tables <- function(data = f_data_sm,country){
  result <- c()
  for(season in unique(data$SEASON)){
    season_table <- final_table(data, country, season)
    league <- data[data$SEASON == season &
                     data$COUNTRY == country,]$LEAGUE[1]
    season_table$SEASON = season
    season_table$COUNTRY = country
    season_table$LEAGUE = league
    result <- rbind(result, season_table)
  }
  cols <- colnames(result)
  len <- length(cols)
  result <- result[c("SEASON","COUNTRY","LEAGUE",cols[1 : (len-3)])]
  return(as.data.frame(result))
}

la_liga_tables <- final_tables(country = "Spain")
la_liga_tables$GD <- la_liga_tables$GF - la_liga_tables$GA
```

2.1 Create a variable with a winning percentage. Note, as there are draws in football, we are going to take draw as a half win - use for loop here. (2p)

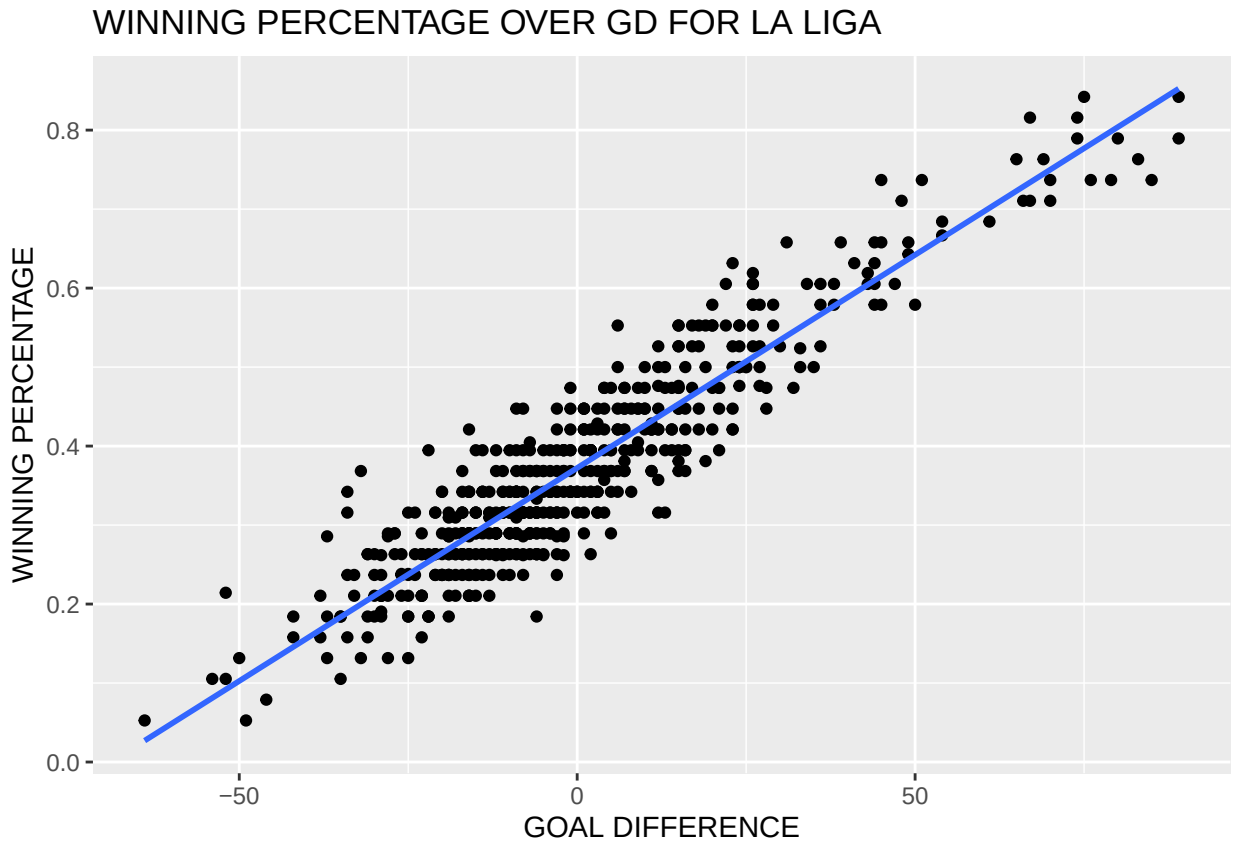
```
calculate_wpct <- function(data){
  result <- c()
  for(index in 1:nrow(data)){
    team <- data[index,]
    team$WPCT <- (team$W) / (team$M)
    team$WPCTv2 <- (team$W + team$D/2)/(team$M)
    team$WPCTv3 <- (team$W + round(team$D / 3)) / (team$M)
    result <- rbind(result,team)
```

```
  }
  return(as.data.frame(result))
}

la_liga_wpct <- calculate_wpct(la_liga_tables)
```

3. Plot Goal differential agains Winning Percentage. 3.1 Add regression line and make your plot pretty. Do you see relationship ? (3p)

```
la_liga_wpct %>%
  ggplot(aes(x = GD, y = WPCT)) +
  geom_point() +
  geom_smooth(method = "lm", se = F) +
  labs(x = "GOAL DIFFERENCE", y = "WINNING PERCENTAGE") +
  ggtitle("WINNING PERCENTAGE OVER GD FOR LA LIGA")
```



3.2 Calculate Pearson correlation coefficient for Goal differential and Winning Percentage. What you think ? (1p)

```
cor.test(la_liga_wpct$GF, la_liga_wpct$WPCT)
```

```
##
##  Pearson's product-moment correlation
##
```

```
## data:  la_liga_wpct$GF and la_liga_wpct$WPCT
## t = 39.212, df = 522, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.8405845 0.8842437
## sample estimates:
##       cor
## 0.8640297
```

```
# The corellation of the variables is very close to 1, which
# means there is a strong association between them
# However I believe that nowadays, high goal difference is not
# always the key factor to success, as many teams with deffensive playing
# style succeed by barely winning a game with a goal difference of 1
# La Liga, includes one of these teams and maybe that is the reason
# that the correlation coeficient is not around 85%,
# Let's remove Atletico to the hell and see the changes
la_liga_no_atl <- la_liga_wpct %>%
  filter(TEAM != "Ath Madrid")
cor.test(la_liga_no_atl$GF, la_liga_no_atl$WPCT)
```

```
##
##  Pearson's product-moment correlation
##
## data:  la_liga_no_atl$GF and la_liga_no_atl$WPCT
## t = 39.752, df = 498, p-value < 2.2e-16
## alternative hypothesis: true correlation is not equal to 0
## 95 percent confidence interval:
##  0.8492426 0.8915154
## sample estimates:
##       cor
## 0.8719952
```

```
# It is not that significant, but I enjoyed the process of removing Atletico
```

4. Your goal is to build a regression model to estimate the k value from Pythagorean formula. Show all the steps, interpret results. (4p)

```
# Let's first build the model, based on combined goals scored and
# goals allowed differnces for all teams in all seasons
# As the draw is not a very rare event in football
# and it occurs too, we cannot use the model considering only
# wins and loses, so for the first model divide football events into
# wins(W) or not wins(D,L)
all_seasons_k <- function(data = la_liga_tables){
  gd_df <- data %>%
    group_by(TEAM) %>%
    summarise(GF.ALL = sum(GF), GA.ALL = sum(GA),
              W.ALL = sum(W), NW.ALL = sum(L) + sum(D)) %>%
    mutate(RATIO = (GF.ALL / GA.ALL)) %>%
    arrange(desc(RATIO))
  k_model <- lm(log(W.ALL / NW.ALL) ~ 0 + log(RATIO), data = gd_df)
```

```
  return(list(df = gd_df, model = k_model))
}
la_liga_all_k <- all_seasons_k()
k <- coefficients(la_liga_all_k$model)
# This model suggests that the predicted win percentage
# for the teams can be calculated by ((GF.ALL))^(2.1) / ((GF.ALL) ^ 2.1 + (GA.ALL) ^ 2.1 )
# Let's do the calculations for Real Madrid

the_greatest <- function(data = la_liga_tables, k) {
 real <- data %>%
  filter(TEAM == "Real Madrid") %>%
  summarise(WPCT = sum(W)/ (sum(M)),
            GF.ALL = sum(GF), GA.ALL = sum(GA)) %>%
  mutate(EXP.WPCT = (GF.ALL)^k /
           ((GF.ALL)^k + (GA.ALL)^k) )
 return(real)
}

real_madrid <- the_greatest(k = k)

# As we can see considering dividing the football events into
# win or not win leads to about 20 % error margin, so let's
# consider how we can improve it
```

```
# To improve our coefficient let's try to calculate the "quality" of the team
# using this formula QualityToWin = (W + D/3)/(L) = (GF)^k / (GA)^k
# as in terms of points three draws are equivalent to one win

all_seasons_k_v2 <- function(data = la_liga_tables){
  gd_df <- data %>%
    group_by(TEAM) %>%
    summarise(GF.ALL = sum(GF), GA.ALL = sum(GA),
              W.ALL = sum(W), D.ALL = sum(D), L.ALL = sum(L)) %>%
    mutate(RATIO = (GF.ALL / GA.ALL)) %>%
    arrange(desc(RATIO))
  k_model <- lm(log((W.ALL + (D.ALL / 3))/ L.ALL) ~ 0 + log(RATIO), data = gd_df)
  return(list(df = gd_df, model = k_model))
}
la_liga_all_k_v2 <- all_seasons_k_v2()
k_v2 <- coefficients(la_liga_all_k_v2$model)

real_madrid_v2 <- the_greatest(k = k_v2)
# As we can see the difference in the formula had a significant effect
# on the model, and our predicted wpct differs from the real one for real :)
# only by 5%, let's apply our calculations also to the worst team in the history
# of football
the_worst <- function(data = la_liga_tables, k){
  simeone <- data %>%
  filter(TEAM == "Ath Madrid") %>%
  summarise(WPCT = sum(W)/ (sum(M)),
            GF.ALL = sum(GF), GA.ALL = sum(GA)) %>%
  mutate(EXP.WPCT = (GF.ALL)^k /
           ((GF.ALL)^k + (GA.ALL)^k) )
```

```
 return(simeone)
}

atletico <- the_worst(k = k)
atletico_v2 <- the_worst(k = k_v2)
# We can see that the second version of k improved our model
# but for those deffending azerbaijan sponsored "Football" team,
# the second version's margin of error is about 10 % compared to almost 20 % of the first k
```

```
# Well, football standings are firstly decided by points
# We can change our calculations. Let's find the point differences
# of each team. For each team X we will calculate points earned by team X
# and the number of points earned by other teams in games with team X

all_seasons_k_v3 <- function(data = la_liga_tables){
  gd_df <- data %>%
    group_by(TEAM) %>%
    summarise(PF.ALL = sum(POINTS), PA.ALL = sum(M)*3 - PF.ALL,
              W.ALL = sum(W), D.ALL = sum(D), L.ALL = sum(L)) %>%
    mutate(RATIO = (PF.ALL / PA.ALL)) %>%
    arrange(desc(RATIO))
  k_model <- lm(log((W.ALL + D.ALL/3 ) / L.ALL) ~ 0 + log(RATIO), data = gd_df)
  return(list(df = gd_df, model = k_model))
}

la_liga_all_k_v3 <- all_seasons_k_v3()
k_v3 <- coefficients(la_liga_all_k_v3$model)

real_madrid_v3 <- the_greatest(k = k_v3)
atletico_v3 <- the_worst(k = k_v3)
# We can again see improvements in predicting the win percentage
# For Real Madrid it is almost the same, whereas for Atletico
# the iprovement is not significant but is visible
```

5. Using estimated k value, calculate Pythagorean Winning percentage and Pythagorean wins for each team. (1p)

```
# Let's first calculate the Pythagorean Winning percentage based on GD's k
la_liga_pyth <- la_liga_wpct %>%
  mutate(P.WP = (GF ^ k_v2) / ( (GF ^ k_v2) + (GA ^ k_v2) ),
         P.EW = round(M * P.WP))
# Now Let's calculate the Pythagorean Winning percentage based on PD's k
la_liga_pyth_v2 <- la_liga_wpct %>%
  mutate(P.WP = (POINTS ^ k_v3)/( (POINTS ^ k_v3) + (M*3 - POINTS) ^ (k_v3) ),
         P.EW = round(M * P.WP))

# The model seems to work relatively good for strong teams,
# but it is not very accurate for weak teams
```

6. Now find overperforming and underperforming teams. (1p)

```
overperforming <- la_liga_pyth %>%
  filter(W > P.EW)
underperforming <- la_liga_pyth %>%
  filter(W < P.EW)
justperforming <- la_liga_pyth %>%
  filter(W == P.EW)


overperforming_v2 <- la_liga_pyth_v2 %>%
  filter(W > P.EW)
underperforming_v2 <- la_liga_pyth_v2 %>%
  filter(W < P.EW)
justperforming_v2 <- la_liga_pyth_v2 %>%
  filter(W == P.EW)
```

Elo

Your goal is to build Elo rating model for NBA games. The data can be found - nba2009_2018

```
# Let's first build the model without considering seasons
# to later analyze seasononal change's factor

make_final_elos_df <- function(final_elos){
  final_elos <- as.data.frame(final_elos)
  final_elos$TEAM <- rownames(final_elos)
  colnames(final_elos) <- c("ELO", "TEAM")
  rownames(final_elos) <- 1:length(final_elos$TEAM)
  final_elos <- final_elos[c("TEAM", "ELO")]
  return(final_elos)
}


nba_elos_filtered <- function(data, home_factor, k_factor, def_elo = 1505){
  elos <- elo.run(score(home.PTS, away.PTS) ~
                            adjust(home.TEAM_NAME, home_factor) +
                            away.TEAM_NAME, data = data,
                          k = k_factor, initial.elos = def_elo)
  elos_df <- data.frame(season = data$SEASON_ID, elos)
  final_elos <- final.elos(elos)
  final_elos <- make_final_elos_df(final_elos)
  return(list(relative = elos_df, elos_df = final_elos))
}
nba_elos <- nba_elos_filtered(nba2009_2018, 100, 20)
```

In your model be sure that: You have adjusted elo ratings at the begining of each season. (5p)

Please follow the link for info here - https://fivethirtyeight.com/features/how-we-calculate-nba-elo-ratings/ . you need to use for loop here, if you have a trouble with it , let me know.

```
get_elos <- function(final_elos, def_elo = 1505){
  starting_elos <- c()
  if(length(final_elos) == 0){
```

```
    starting_elos = def_elo
  }
  else{
    starting_elos <- (def_elo * 0.25) + (final_elos * 0.75)
  }
  return(starting_elos)
}


nba_elos_relative <- function(data = nba2009_2018, home_factor, k_factor, def_elo = 1505){
  relative_df <- c()
  final_elos <- c()
  for(season in unique(data$SEASON_ID)){
    starting_elos <- get_elos(final_elos, def_elo)
    season_data <- data[data$SEASON_ID == season,]
    season_elos <- elo.run(score(home.PTS, away.PTS) ~
                             adjust(home.TEAM_NAME, home_factor) +
                             away.TEAM_NAME, data = season_data,
                           k = k_factor, initial.elos = starting_elos)
    season_final_elos <- final.elos(season_elos)
    season_elos <- data.frame(SEASON = season,season_elos)
    relative_df <- rbind(relative_df, season_elos)
    final_elos <- season_final_elos
  }
  final_elos <- make_final_elos_df(final_elos)
  return(list(relative = relative_df, elos_df = final_elos))
}
nba_rel_elos <- nba_elos_relative(nba2009_2018, 100, 20)
nba_rel_elos_df <- nba_rel_elos$elos_df
nba_rel_seasonal <- nba_rel_elos$relative
```

Try different values for home team advantage, K, anything else ?. (5p)

How good is your model in predicting the game results ? Use either Brier score or confusion matrix. (2p)

```
# Let's first make the confusion matrices based on recomended values
# As in our data set there are no draw games we will change the range of
# values for predicted outcome a little bit
#                         {
#                         1,p > 0.5,
#  Predicted Outcome =    0,p < 0.5,
#                         rnd, p = 0.5
#                         }
# As the draws are very random event in baseball, let the luck handle the case
# when the teams are expected to draw, so if the predicted outcome is exactly
# 0.5 we will call a function that will randomly pick 0 or 1

rand_0_1 <- function() {
  round(runif(n = 1, min = 0, max = 1))
}
nba_rel_seasonal <- nba_rel_seasonal %>%
  mutate(Predicted = ifelse(p.A > 0.5,1,
                            ifelse(p.A < 0.5, 0, rand_0_1())))
```

```
make_conf_matrix <- function(data = nba_rel_seasonal){
  rel_seasonal_conf <- table(Actual = data$wins.A,
                             Predicted = data$Predicted)
  colnames(rel_seasonal_conf) <- c("Win", "Lose")
  rownames(rel_seasonal_conf) <- c("Win", "Lose")

  accuracy <- sum(diag(rel_seasonal_conf))/sum(rel_seasonal_conf)
  return(list(table = rel_seasonal_conf, accuracy = accuracy))
}

calculate_brier <- function(data = nba_rel_seasonal){
  diff <- data$p.A - data$wins.A
  accuracy <- sum(diff^2) / nrow(data)
  return(accuracy)
}

conf_seasonal_custom <- make_conf_matrix()
conf_seasonal <- confusionMatrix(data = factor(nba_rel_seasonal$wins.A),
                                              reference = factor(nba_rel_seasonal$Predicted))

brier_seasonal_custom <- calculate_brier()
brier_seasonal_score <- BrierScore(pred = nba_rel_seasonal$p.A,
           resp = nba_rel_seasonal$wins.A)

brier_seasonal_score
```

```
## [1] 0.210655
```

In my opinion brier score is a better way for estimating your predictions, as the predicted outcome can take any value in the range of (0,1), whereas for confusion matrix we form some divisions of the predicted values so it takes only values 0 or 1. Let's try to identify the good values for k and home advantage by changing them , and see how does it effect the brier score

```
# Let's start by identifying a good value for k

nba_k_picker <- function(data = nba2009_2018){
  result <- c()
  for( k in seq(15,25,0.1)){
    elos_for_k <- nba_elos_relative(data = data,
                                    home_factor = 100,
                                    k_factor = k )
    br_scores <- elos_for_k$relative %>%
      group_by(SEASON) %>%
      summarise(BR.SCORE = BrierScore(pred = p.A,
                                      resp = wins.A),
                k = k)
    result <- rbind(result, br_scores)
  }
  model <- lm( BR.SCORE ~ 0 + k, data = result )
  return( list(model = model,
               df = as.data.frame(result)) )
}
```

```
brier_k <- nba_k_picker()
coefficients(brier_k$model)
```

```
##          k
## 0.01031682
```

```
# The model suggests us that BRIER_SCORE =~ 0.01 * K
# It means that with the increase of k brier_score increases too
# But the lower the brier the higher the precision
# So in general, after finding the good value of k, the increase in
# k will lead to decrease in precision
best_nba_k <- mean((brier_k$df %>%
  group_by(SEASON) %>%
  filter(BR.SCORE == min(BR.SCORE)) %>%
  ungroup())$k)
# Seems like 22.03 is the best option


# We found a good estimate for k, now let's
nba_home_adv_picker <- function(data = nba2009_2018){
  result <- c()
  for( h_f in seq(50,150, 1)){
    elos_for_k <- nba_elos_relative(data = data,
                                    home_factor = h_f,
                                    k_factor = best_nba_k )
    br_scores <- elos_for_k$relative %>%
      group_by(SEASON) %>%
      summarise(BR.SCORE = BrierScore(pred = p.A,
                                      resp = wins.A),
                H.FACTOR = h_f)
    result <- rbind(result, br_scores)
  }
  model <- lm( BR.SCORE ~ 0 + H.FACTOR, data = result )
  return( list(model = model,
               df = as.data.frame(result)) )
}

brier_home <- nba_home_adv_picker()
coefficients(brier_home$model)
```

```
##    H.FACTOR
## 0.001956323
```

```
# The same situation as described above,
# but as we took mostly hundreds as a measurement
# the coefficeint for home_adv is almost ten times smaller than the one for k
best_nba_ha <- mean((brier_home$df %>%
  group_by(SEASON) %>%
  filter(BR.SCORE == min(BR.SCORE)) %>%
  ungroup())$H.FACTOR)
```

Is it getting good or worst by season ? Are you consistently making errors for one team ? (3p)

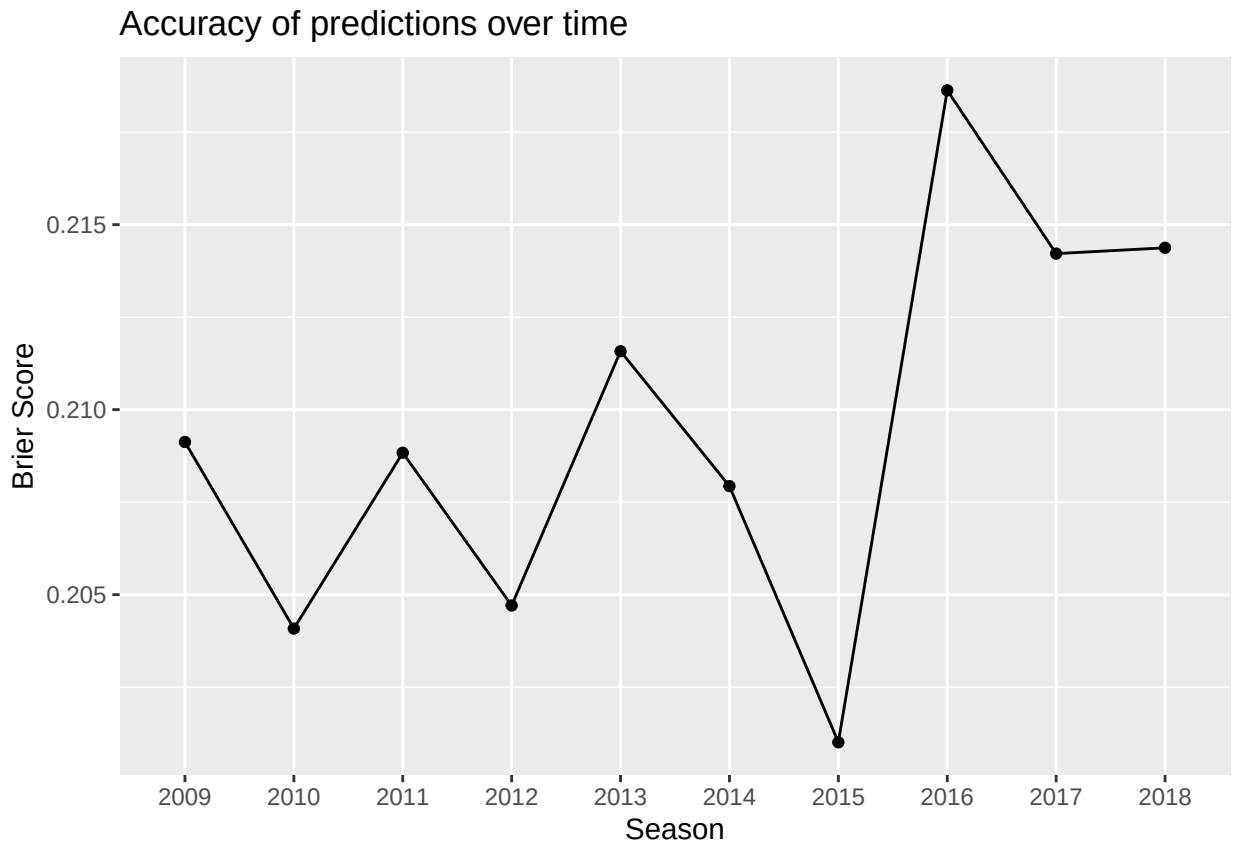
```
best_nba_relative <- nba_elos_relative(home_factor = best_nba_ha, k_factor = best_nba_k)$relative
best_nba_ratings <- nba_elos_relative(home_factor = best_nba_ha, k_factor = best_nba_k)$elos_df
brier_for_best <- calculate_brier(data = best_nba_relative)
brier_for_best
```

```
## [1] 0.209462
```

```
brier_over_seasons <- function(data = best_nba_relative){
  briers <- c()
  for( season in unique(data$SEASON) ){
    season_data <- data %>%
      filter(SEASON == season)
    season_brier <- calculate_brier(season_data)
    briers <- c(briers, season_brier)
  }
  return(data.frame(SEASON = unique(data$SEASON), BR.SCORE = briers))
}

options(scipen = 999, digits = 4)
briers <- brier_over_seasons()

briers$SEASON <- factor(briers$SEASON, levels = unique(briers$SEASON))
briers %>%
  ggplot(aes(x = factor(SEASON), y = BR.SCORE, group = BR.SCORE)) +
   geom_line(aes(group=1)) +
    geom_point() +
    labs(x = "Season", y = "Brier Score") +
    ggtitle("Accuracy of predictions over time")
```



As the visualization shows, the accurace changes every season, and most of the time it flips it's direction For example the accuracy got better in 2010 compared to 2009 but got worse in 2011, and the similar pattern was untill 2013. Starting from 2013 up to 2015 the accuracy was improved(brier score decreased) and in 2015 the accuracy was the highest, but then in the next year it became the lowest and then improved again in 2017, and stayed almost on the same level in 2018

```
nba_games <- read.csv("nba_games.csv")

get_teams_elo <- function(data = best_nba_ratings, team){
  return((data %>%
            filter(TEAM == team))$ELO
  )
}

nba_preds <- nba_games %>%
  rowwise() %>%
  mutate(wins.A = elo.prob(elo.A = get_teams_elo(team = team.A),
                           elo.B = get_teams_elo(team = team.B)),
         wins.B = 1 - wins.A)

head(nba_preds)
```

```
## Source: local data frame [3 x 4]
## Groups: <by row>
##
## # A tibble: 3 x 4
```

```
##   team.A               team.B            wins.A wins.B
##   <fct>                <fct>              <dbl>  <dbl>
## 1 Cleveland Cavaliers  Boston Celtics     0.188  0.812
## 2 New Orleans Pelicans San Antonio Spurs  0.256  0.744
## 3 Philadelphia 76ers   Orlando Magic      0.539  0.461
```

Look at the schedule of upcoming NBA games. Make predictions for the first 3 games. (3p)

The favorites to win

Boston Celtics (playing away) San Antonio Spurs (playing away) Philadelphia 76ers (playing home) I don't follow nba so I don't know how reailstic are my predictions, but I noted that for these games the favorites are mostly away teams, except Philadelphi 76ers game, whereas for that game the probabilites for both teams to win are almost the same.