Connor McDavid Pt Model

Tyrel Stokes

18/02/2022

Modelling Points at the Game Level

Quickly sharing some code to reproduce a version of my Connor McDavid Pt Model. This isn't a very practical model in that it won't scale well without considerable effort, but it is a fun model and might be useful for someone learning about Bayesian Modelling.

The goal is to model goals and assists jointly at the game level and mostly as an excercise allow the predicted points to vary by the interaction of season and opponenent. That is there is a parameter for all 31 teams in all 7 seasons for both goals and assists. We don't really have the information in the data to do this particularly well so the priors are quite strong in order to make the model feasible with the data. Here is the probabilistic model and then the stan code + some stuff to make fun graphs.

The original model also modelled time on ice allowing those parameters to vary by season. Conditioning on the season there seems to be almost no additional information that the time on ice provides for predicting points. There are some weird selection problems here likely at play. In blowout games, Connor both tends to have lots of points and low ice time for example. At the game level this causes there not to be much marginal correlation it seems. It would likely take even more granular data and clever modelling to get much information out of that. For clarity (and for model speed and to avoid divergences) I got rid of the TOI modelling.

Probabilistic Model

$$
\begin{aligned}
Goals_{s,tm,i} &\sim Poisson(\exp(\mu_s^{goals} + \mu_{s,tm}^{goals} && (1)\\
&\quad + home_i \beta_{home}^{goals})) && (2)\\
Assists_{s,tm,i} &\sim Poisson(\exp(\mu_s^{assists} + \mu_{s,tm}^{assists} && (3)\\
&\quad + home_i \beta_{home}^{goals} + Goals_{s,tm,i}\beta_{goal})) && (4)
\end{aligned}
$$

That's the likelihood. I add the goal term in the assist modelling to try and capture some within game dependence. If Connor scores, he can't get an assist on that goal which could induce some within game negative dependence. But otherwise goals and assists are model symmetrically.

The difficult part is setting up the priors to share enough information since this model is severely over-parametrized. For the season level means a random walk structure was chosen.

$$
\begin{aligned}
\mu_s^M &\sim N(\mu_{s-1}^M, \sigma_s^M),\\
&\quad s \in \{2, ..., 7\}\\
\mu_1^M &\sim N(\mu^M, \sigma_{mu}^M)\\
\sigma_s^M &\sim N(0, 0.25)T[0,]\\
M &\in \{Goals, Assists\}
\end{aligned}
$$

Where μ^M and σ^M_{mu} were chose fixed values corresponding to a somewhat informative prior centered near Connor's first season averages.

The team interactions were modelled with an $AR(1)$ process. This has a similar flavour to the random walk but pulls more information between the team-seasons. Since there are over 200 of these parameters per metric, we don't want to allow them to roam too freely.

$$\begin{aligned}
\mu_{s,tm} &\sim N(\mu^M_{s-1,tm} \times \rho_M, \sigma^M_{tm}) \\
&\quad s \in \{2,\ldots,7\}, \quad tm \in \{1,\ldots,31\} \\
\mu_{1,tm} &\sim N(0, 0.1) \\
\rho_M &\sim N(0, 0.3)T[0,1] \\
\sigma^M_{tm} &\sim N(0, 0.08)T[0,] \\
&\quad M \in \{Goals, Assists\}
\end{aligned}$$

Notice these priors are pretty tight, they likely could be relaxed some. Before I got rid of all the time on ice terms this tightness was essentially for avoiding divergences. I think that should be less of an issue here, but I have not tested it.

The remaining fixed effect priors were all weakly informative.

Code for the model

```
functions{
  real normal_lub_rng(real mu, real sigma, real lb, real ub) {
  real p_lb = normal_cdf(lb| mu, sigma);
  real p_ub = normal_cdf(ub| mu, sigma);
  real u = uniform_rng(p_lb, p_ub);
  real y = mu + sigma * Phi(u);
  return y;
}
}
data {
  int<lower=0> N;
  int Assists[N];
  int Goals[N];
  int opponent[N];
  int season[N];
  vector[N] home;
  int ns;
  int nt;
  int N2;
  int opponent_remain[N2];
  int home_remain[N2];
  int season_remain[N2];

  real cur_goals;
  real cur_assists;

}parameters {

  matrix[nt,ns] mu_assists_team_0;
  matrix[nt,ns] mu_goals_team_0;
  vector[ns] mu_assists_t_0;
  vector[ns] mu_goals_t_0;

  real<lower=0> sigma_assists_t;

  real<lower=0> sigma_goals_t;
  real<lower=0> sigma_assists_team;
  real<lower=0> sigma_goals_team;

  vector[3] b_home;

  real beta_goal;

  real<lower = 0, upper =1> rho_a;
  real<lower=0, upper =1> rho_g;
}

transformed parameters{


   matrix[nt,ns] mu_assists_team;
  matrix[nt,ns] mu_goals_team;
  vector[ns] mu_assists_t;
  vector[ns] mu_goals_t;

   mu_assists_t[1] = mu_assists_t_0[1];
  mu_goals_t[1] = mu_goals_t_0[1];

   for(i in 2:ns){
   mu_assists_t[i] = mu_assists_t[i-1] + mu_assists_t_0[i]*sigma_assists_t;
   mu_goals_t[i] = mu_goals_t[i-1] + mu_goals_t_0[i]*sigma_goals_t;
 }
 mu_assists_team[:,1] = mu_assists_team_0[:,1];
 mu_goals_team[:,1] = mu_goals_team_0[:,1];
for(i in 2:ns){
  for(j in 1:nt){
    mu_assists_team[j,i] = rho_a*mu_assists_team[j,(i-1)] + mu_assists_team_0[j,i]*sigma_assists_team;
    mu_goals_team[j,i] = rho_g*mu_goals_team[j,(i-1)] + mu_goals_team_0[j,i]*sigma_goals_team;
  }
}
}model {
vector[N] mn_a;
vector[N] mn_g;

for(i in 1:N){

  mn_a[i] = mu_assists_t[season[i]]+ mu_assists_team[opponent[i],season[i]]+ home[i]*b_home[2] + Goals[
  mn_g[i] = mu_goals_t[season[i]] + mu_goals_team[opponent[i],season[i]]+ home[i]*b_home[3];

}

  Assists ~ poisson_log(mn_a);
  Goals ~ poisson_log(mn_g);

  mu_assists_team_0[:,1] ~ normal(0,0.1);
  mu_goals_team_0[:,1] ~ normal(0,0.1);

  to_vector(mu_assists_team_0[:,2:ns]) ~ normal(0,1);
  to_vector(mu_goals_team_0[:,2:ns]) ~ normal(0,1);

   mu_assists_t_0[1] ~ normal(-0.3,.5);
  mu_goals_t_0[1] ~ normal(-1.05,.5);

  for(i in 2:ns){
  mu_assists_t_0[i] ~ normal(0,1);
  mu_goals_t_0[i] ~ normal(0,1);
  }

  sigma_assists_t ~ normal(0,0.25)T[0,];
  sigma_goals_t ~ normal(0,0.25)T[0,];
  sigma_assists_team ~ normal(0,0.08)T[0,];
  sigma_goals_team ~ normal(0,0.08)T[0,];

 b_home ~ normal(0,0.5);
  beta_goal ~ normal(0,0.5);
```

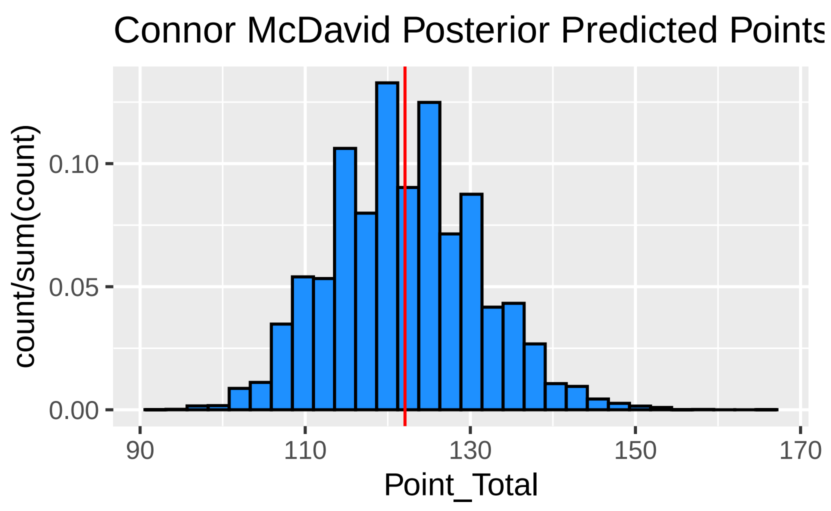
```
rho_a ~ normal(0,0.3)T[0,1];
rho_g ~ normal(0,0.3)T[0,1];


}generated quantities{

  vector[N2] pred_assists;
  vector[N2] pred_goals;

  real pred_total_assists;
  real pred_total_goals;
  real pred_total_points;

  real over_100;
  real over_200;
  real lb;
  lb = 0;

  for(i in 1:N2){
    pred_assists[i] = poisson_log_rng(mu_assists_t[season_remain[i]] + mu_assists_team[opponent_remain[i
    pred_goals[i] = poisson_log_rng(mu_goals_t[season_remain[i]] + mu_goals_team[opponent_remain[i],seas
}

  pred_total_assists = cur_assists + sum(pred_assists);
  pred_total_goals = cur_goals + sum(pred_goals);
   pred_total_points = pred_total_assists + pred_total_goals;

  if(pred_total_points >= 100){
    over_100 = 1.0;
  }else{
    over_100 = 0.0;
  }

  if(pred_total_points>= 200){
    over_200 = 1.0;
  }else{
    over_200 = 0.0;
  }


}
```

Now I load the data so we can fit the model. The data are cleaned game logs from every game connor has played to date and the remaining schedule for the oilers this season. We need the remaining schedule to make predictions since the goals and assist parameters depend on the opponent.

```
Mcdavid2 <- read.csv("McDavid_df.csv")
oilers <- read.csv("oilers_schedule_2122.csv")

names(oilers)[4] <- "Team"
oilers$Home <- as.numeric(grepl("vs",oilers$Vs))

tms <- unique(oilers$Team)

tms2 <- c("VAN", "CGY", "ANA","ARI","VGK","PHI","SEA","NSH","NYR","DET","BOS","BUF","STL","WPG","CHI","
          "NYI","OTT","FLA","MTL","WSH","TBL","COL")

oilers$Teams_2 <- plyr::mapvalues(oilers$Team, from = tms, to = tms2)


oilers$opp_int <- plyr::mapvalues(oilers$Teams_2, from = c(unique(Mcdavid2$opponent[1:412]),"SEA"), to

df_s7 <- Mcdavid2 %>% filter(season2 ==7)

N_games <- nrow(df_s7)## How many games have already past this season

curr_assists <- sum(df_s7$Assists)
curr_goals <- sum(df_s7$Goals)

oilers2 <- oilers[-c(1:N_games),]

mcdata3 <- list(N = nrow(Mcdavid2),Assists = Mcdavid2$Assists, Goals = Mcdavid2$Goals,
                opponent = as.numeric(Mcdavid2$opp_int), season = as.numeric(Mcdavid2$season2), home =
                ns = 7, nt = 31,N2 = nrow(oilers2),opponent_remain = as.numeric(oilers2$opp_int), seaso
                home_remain = oilers2$Home,  cur_goals = curr_goals,cur_assists = curr_assists)

fitt_s <-  mcdavid_pts_assists$sample(data = mcdata3, parallel_chains = 4, iter_warmup = 2500, iter_sam
```

The model takes about 50 seconds to sample 6500 x 4 parallel chains on my machine. Not bad for a model with 400 games of data and well over 400 parameters. But again, not likely to scale well. We certainly would not be able to run this model on more than a handful of players at a time without great sadness.

```
point_draws <- as_draws_df(fitt_s$draws(c("pred_total_points","pred_total_goals","pred_total_assists"))
names(point_draws)[1:3] <- c("Point_Total", "Goal_Total","Assist_Total")

p <- ggplot(point_draws, aes(x = Point_Total)) +geom_histogram(aes(y = stat(count / sum(count))),fill =
  geom_vline(xintercept = mean(point_draws$Point_Total), color = "red")+ ggtitle("Connor McDavid Poster
p
```



```
mean(point_draws$Point_Total)
```

```
## [1] 122.0877
```

As of February 18, 2022 the model expects McDavid to finish with roughly 122 points this season. His chances of a 150 + season have evaporated as well :(.

```
pg <- as.data.frame(point_draws$Goal_Total)
names(pg) <- "x"
pg$y <- "Goals"

pa <- as.data.frame(point_draws$Assist_Total)
names(pa) <- "x"
pa$y <- "Assists"

p2 <- rbind(pg,pa)

ci1 <- ci(pg$x,method = "HDI", ci = 0.93)
ci2 <- ci(pa$x,method = "HDI", ci = 0.93)

ci3 <- ci(point_draws$Point_Total, method = "HDI", ci = 0.93)
```

```
ci1
```

```
## 93% HDI: [35.00, 52.00]
```

The 93% (for nuge) highest posterior interval is 35-52 goals

```
ci2
```

```
## 93% HDI: [67.00, 93.00]
```

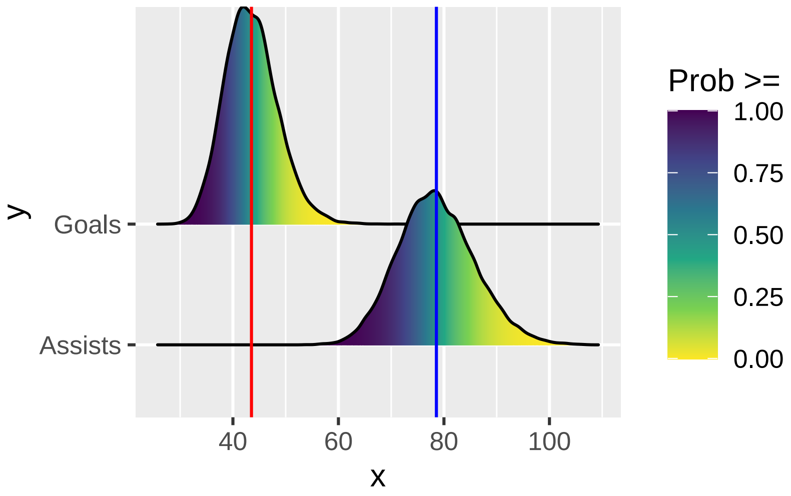
67-93 assists

```
ci3
```

```
## 93% HDI: [107.00, 138.00]
```

and 108-139 points. This all seems pretty reasonable. Below see a plot of the goals and assists densities. The shading on the densities indicate the likelihood of getting that many points or more.

```
pl2 <- ggplot(p2, aes(x=x,y = y, fill = (1-stat(ecdf)))) +stat_density_ridges(geom = "density_ridges_gra
  geom_vline(xintercept = mean(pg$x), color = "red") + geom_vline(xintercept = mean(pa$x), color = "blue
pl2
```



```
mean(pg$x)
```

```
## [1] 43.51456
```

```
mean(pa$x)
```

```
## [1] 78.57313
```

The red and blue lines denote the posterior predicted mean goals and assists, 43.5 and 78.6 respectively.