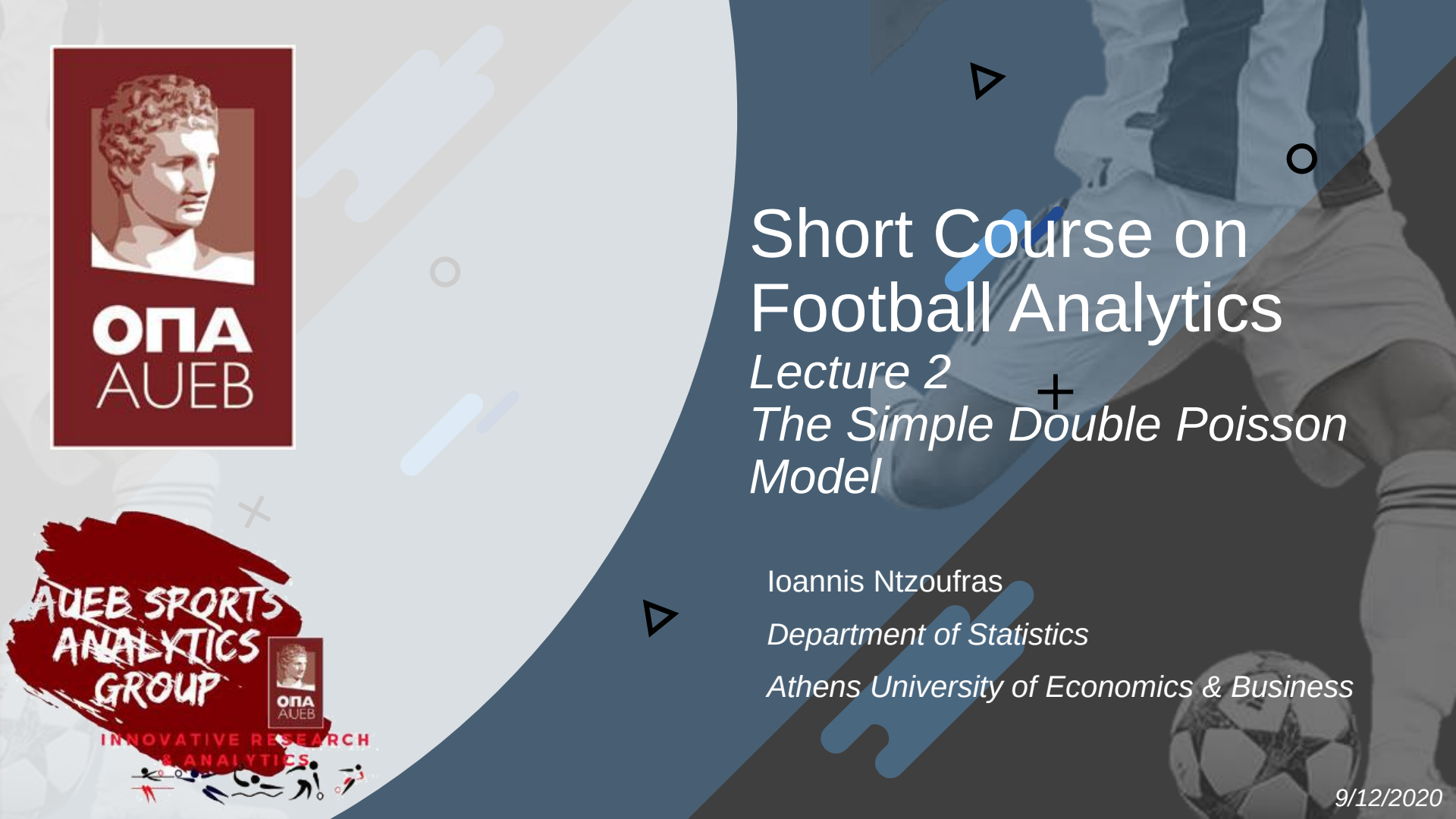


# Short Course on Football Analytics

## *Lecture 2*

## *The Simple Double Poisson Model*


Ioannis Ntzoufras

*Department of Statistics*

*Athens University of Economics & Business*


*9/12/2020*

# The Double Poisson Model


Here we focus on Poisson regression models for response variables defined in

- Such variables usually express the number of successes (visits, telephone calls,
- number of scored goals in football) within a fixed time interval.
- They are frequently called Poisson log-linear models because of the canonical link function (i.e. the log of the mean)

# The Double Poisson Model


Here we focus on Poisson regression models for response variables defined in

- Such variables usually express the number of successes (visits, telephone calls,
- number of scored goals in football) within a fixed time interval.
- They are frequently called Poisson log-linear models because of the canonical link function (i.e. the log of the mean)

# The vanilla model

## Background information and the vanilla model

✓ Simple and basic log-linear model originally introduced by Maher (1982).

✓Also used by other authors such as Lee (1997) and Karlis and Ntzoufras (2000).

We denote by

– $y_{i1}$ = goals scored by home team (HT) in the i-th game.

– $y_{i2}$ = goals scored by away team (AT) in the i-th game.

The model is given by

$$Y_{ij} \sim Poisson\,(\lambda_{ij})$$

$$\log(\lambda_{i1}) = \mu + home + att_{HT_i} + def_{AT_i}$$

$$\log(\lambda_{i2}) = \mu + att_{AT_i} + def_{HT_i}$$

where

- n = number of games,
- μ = constant parameter;
- home = home effect;
- $HT_i$ and $AT_i$ = home and away teams in i game;
- $att_k$ and $def_k$ = attacking and defensive effects–abilities of k team for k = 1, 2, . . . ,K; and
- K = number of teams in the data (here K = 20).

# The vanilla model

## Interpretation of the parameters

- ✓ $e^{\mu}$: Expected number of goals of the away team in a game between two teams of equal strength
- ✓ $e^{\mu+home}$: Expected number of goals of the home team in a game between two teams of equal strength
- ✓ $e^{home}$: Relative increase of the expected home goals in a game between two teams of equal strength

# The vanilla model


## Interpretation of the parameters

✓ *We use sum-to-zero constraints for attacking and defensive abilities*

$$\sum_{k=1}^{K} att_k = 0 \text{ and } \sum_{k=1}^{K} def_k = 0.$$

✓ *Thus, comparison is made with a team with an average (attacking of defensive) strength/ability*

✓ *In practice STZ dummies with* $att_1 = -\sum_{k=2}^{K} att_k$ *and* $def_1 = -\sum_{k=2}^{K} def_k$.

✓ *Positive values of attacking abilities -> better teams than average*

✓ *Negative values of defensive abilities -> better teams than average*

✓ $e^{att_k}$: *Relative increase/decrease in expected goals in comparison to an average team*

✓ $e^{def_k}$: *Relative decrease/expected in expected goals of the opponent team in comparison to an average team*

# The vanilla model

## Interpretation of the parameters

✓ *Note that*

$$\log\left(\frac{\lambda_{i1}}{\lambda_{i2}}\right) = home + \left(att_{HT_i} - def_{HT_i}\right) - \left(att_{AT_i} - def_{AT_i}\right)$$
$$= home + Ability_{HT_i} - Ability_{AT_i}$$

✓ *Hence, differences of attacking and defensive abilities can be considered as overall abilities*

# Basic Football Modelling Assumptions

**Important Assumptions that we need to be check**

1. Dependence between goals of the opponent teams
2. Over-dispersion
3. Excess of Draws
4. Dynamic Abilities

# Basic Football Modelling Assumptions

1. Dependence between goals of the opponent teams
   - ❑ Naturally, dependence should be added
   - ❑ Small dependence (but significant) has been observed
   - ❑ Possible remedies
     - ❑ Bivariate Poisson model (Karlis and Ntzoufras, 2003, JRSSD)
     - ❑ Poisson Difference/Skellam distribution (Karlis and Ntzoufras, 2008)
     - ❑ Random effects models – normal RE do not work very well for football data
     - ❑ Copula models

# Basic Football Modelling Assumptions

## 2. Over-dispersion

- ❑ Empirical evidence has shown over-dispersion in most leagues
- ❑ Negative binomial model is a good/standard choice
- ❑ Log-normal RE do not use very well (easy alternative)
- ❑ What about under-dispersion?

# Basic Football Modelling Assumptions

## 3. Excess of Draws

- ❑ Usually models for goals do not predict very well the draws
- ❑ In practice we have an excess of draws in comparison to the predicted one, especially for the 0–0 and 1–1 draws.
- ❑ Remedy: Draw inflation component
- ❑ What about cases where there is shortage of draws: Some exceptions in Premier League

# Basic Football Modelling Assumptions

## 3. Excess of Draws

- Remedy: Draw inflation component
  - Dixon and Coles (1997) → extra probabilities in these scores.
  - Karlis and Ntzoufras (2003) → diagonal inflated bivariate Poisson model
  - Karlis and Ntzoufras (2008) → zero inflated model for the goal differences.

# Basic Football Modelling Assumptions

## 4. Dynamic abilities

- This assumes that abilities are not constant across the season.
- Usually an AR(1) structure is used (hierarchical model)
- This is the modern trend in the literature
- More realistic. More complicated to fit.

# Using Covariates and additional info


## What about covariates?

- The vanilla usually serves as the base for comparisons
- Models using covariates can be further elaborated
- Different covariates for
  - Team/Player performance – Use box-scores at the end of the model
  - For prediction – Use similar covariates but summaries from previous games that are known before the beginning of the game
- Use variable selection or LASSO for selecting covariates

# Using Covariates and additional info


## Covariates vs Abilities?

- ❑ Attacking and defensive abilities are strongly confounded with candidate covariates
- ❑ We may leave them inside for controlling for other effects but usually will not be significant
- ❑ Prediction with additional covariates may be improved but not as much as we would expect because the abilities capture similar qualities

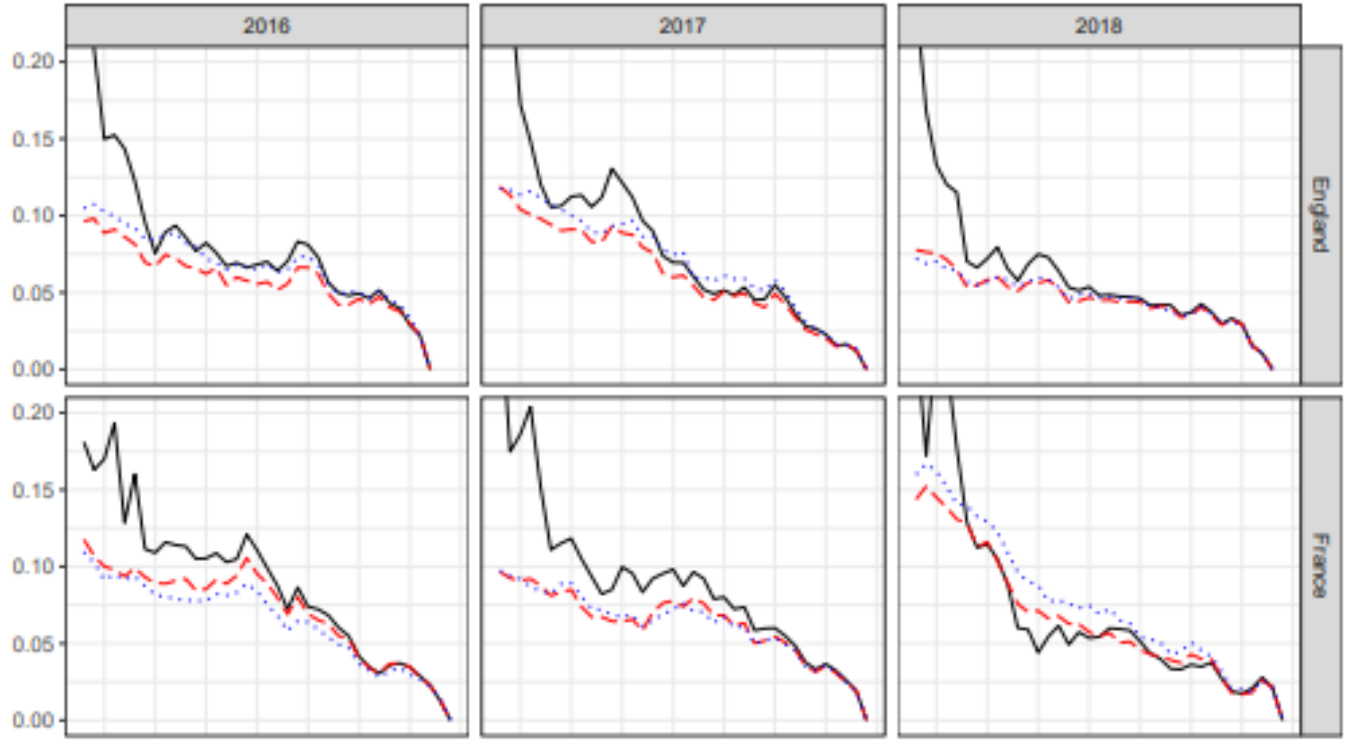
# Using Covariates and additional info

## Covariates vs Abilities?

- Covariates capture dynamic changes in abilities – and simpler to fit
- Models with covariates can use information from earlier seasons or they need lower number of games to provide good estimates
- Attacking and Defensive abilities need increase sample size to provide good prediction
- The mid-season is a good point for vanilla model to work efficient

# The vanilla model is not so bad!

## Covariates vs Abilities?



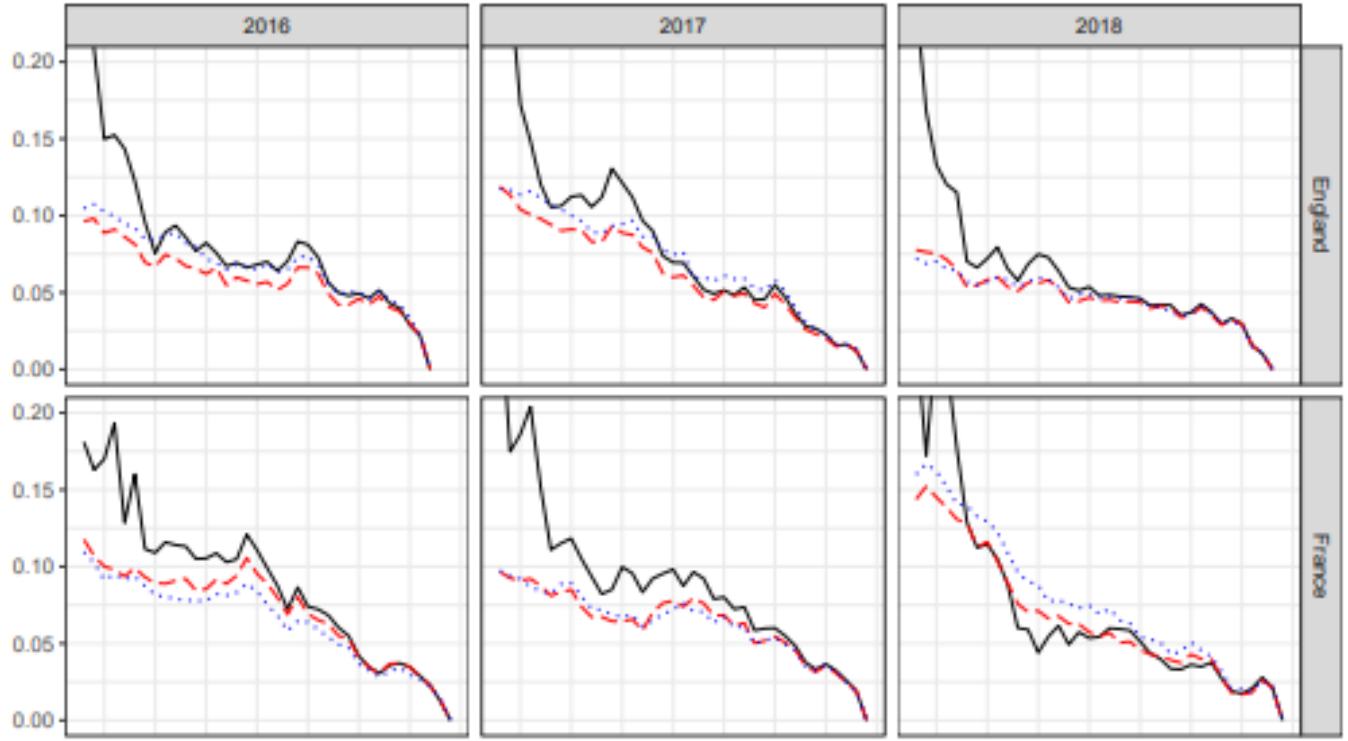

**Source:** Van Eetvelde, Hvattum & Ley (2021). *The Probabilistic Final Standing Calculator: a fair stochastic tool to handle abruptly stopped football seasons.*

See the opening presentation in SAW2021

Figures for 3 more leagues are available in the paper. The message is the same

# The vanilla model is not so bad!

## Covariates vs Abilities?




Lines in the plots:

1. **Black** -> simpler than vanilla with common abilities
2. **Red** -> with covariates and info from previous year
3. **Black dotted**: weighted version of vanilla – more weight to recent data – similar to dynamic – simpler to fit

# Models for other tournaments


## Models for European or International Tournaments?

- For tournaments that are not round-robin and the teams are separated in different groups then the vanilla model will not work until we have a cross-over between different groups
- Models with covariates are better
- We need also a covariate/index of the overall ability of the team such as the UEFA coefficient or ranking

# Football Modelling Protocol


## Protocol for a good fitted/prediction model

- ❑ Fit the vanilla model for reference
- ❑ Select possible covariates
- ❑ Implement variable selection for models with and without abilities
- ❑ Decide if you are going to include abilities with covariates
- ❑ Check the assumptions of the models
- ❑ Check the Goodness of Fit (in sample prediction)
- ❑ Check for the out of sample prediction

# Football Modelling Protocol

## Check the assumptions of the models

- ❑ Bivariate structure? Fit a bivariate model and test if this is preferable
- ❑ Overdispersion: Check for overdispersion. Use NB to see if the fit is improved
- ❑ Draws: Check the fit for the draws. Fit Draw inflated models and decide if they are better
- ❑ Dynamic models: Use dynamic abilities and see if the model is considerably improved.

# Football Modelling Protocol


## Goodness of fit

- ❑ Check the marginal distribution of goals
- ❑ Regenerate the full League
  - ❑ Rankings and expected points should be similar to the observed one
  - ❑ This means that the model describes well the final table
  - ❑ If can quantify the uncertainty of the final table
- ❑ Check for outliers/surprising games and see if
  - ❑ The model is justified for not picking this surprise
  - ❑ Whether there was some info not considered by the model that may influence the result

# Football Modelling Protocol

## Out-of-sample prediction (when valid)

- Not appropriate for performance analysis models using covariates from the [illegible] score at the end of each game
- Use some data for model fit (train) and some data for testing prediction (test)
- Usual approaches
  - Mid-season – enough info for the model to work well
  - Play-off or knock out prediction – when valid – usually surprises occur
  - One week ahead prediction
  - One stage ahead prediction (for knock-out competitions)

# Football Modelling Protocol


Out-of-sample prediction (when valid)

- ❑ Check the marginal distribution of goals
- ❑ Regenerate the full League
  - ❑ Rankings and expected points should be similar to the observed one
  - ❑ Make predictions for expected number of points and rankings
- ❑ Selection of the model is usually performed with in-sample-statistics or criteria (BIC, AIC, DIC) and validation using out-of-sample prediction
- ❑If prediction is the main focus, selection of the model can be based on the predictive ability of the model as in a machine learning approach

# Final League Table Fit & Prediction

## How to re-generate a league

Classic approach

- Use fitted values for each game to generate $Y_{ij}^{pred,(t)} \sim Poisson(\hat{\lambda}_{ij})$ for *i=1, …, n* and *j=1,2*
- Calculate the goal differences $D_{ij}^{(t)} = Y_{i1}^{pred,(t)} - Y_{i2}^{pred,(t)}$
- Calculate the points attributed in home and away teams $P_{i1}^{(t)} = 3 \times I\left(D_{ij}^{(t)} > 0\right) + I\left(D_{ij}^{(t)} = 0\right)$ and $P_{i2}^{(t)} = 3 \times I\left(D_{ij}^{(t)} < 0\right) + I\left(D_{ij}^{(t)} = 0\right)$
- Calculate the total points earned in each replication t as

$$P_k^{(t)} = \sum_{i=1}^{n} P_{i1}^{(t)} \times I(HT_i = k) + \sum_{i=1}^{n} P_{i2}^{(t)} \times I(AT_i = k)$$

# Final League Table Fit & Prediction

## How to re-generate a league

### Classic approach (contu'ed)

- Produce the team rankings $R_k^{(t)}$ according to the order of the total points
- Repeat for t=1, …, B replicates/leagues
- Use the re-generated scores, differences, points and rankings to produce summary statistics according to the model and compare with the observed table

# Final League Table Fit & Prediction

## How to re-generate a league

### Classic approach

- ❑ This is the plug-in approach.
- ❑ Additional variability can be considered about estimated λs using bootstrap

### Classic approach

- ❑ λs are calculated from a sample from the posterior predictive distribution
- ❑ Very simple within MCMC and related software
- ❑ No bootstrap or similar step is needed since the variability of the parameters is inherited from the posterior distribution

# Play-offs and Knock out games

## Play-off or knock out approach

- ❑ More difficult since we need to consider possible cross-overs
- ❑ Play-offs with multiple games/wins also need to be accounted with extra programming
- ❑ Simplification: We consider the observed cross-over in next knock-out phases (if known)

# Tools for fitting football models


- WinBUGS/OpenBUGS/Jags -> Details can be found in my wi… book (and in this presentation)
- footbayes R package by Leonardo Egidi (see course Day 2 and our future book on Football analytics)
  https://github.com/LeoEgidi/footBayes
- soccerAnimate: an R package to create 2D soccer animations
  https://www.datofutbol.cl/soccer-animate-r-package/
- regista R package -> Dixon and Coles model, xG?
  https://github.com/Torvaney/regista
- goalmodel R package: https://github.com/opisthokonta/goalmodel
- ffanalytics R Package for Fantasy Football Data Analysis
  https://fantasyfootballanalytics.net/2016/06/ffanalytics-r-package-fantasy-football-data-analysis.html

# R packages with Football data


- engsoccerdata: English and European Soccer Results 1871-2020
  https://github.com/jalapic/engsoccerdata
- worldfootballR: An R Package to Extract World Football (Soccer) Data from Fbref.com and transfermarkt.com
  https://github.com/JaseZiv/worldfootballR
- footballR: R package to obtain football (soccer) data from APIs
  https://github.com/dashee87/footballR

# Example: English Premier League 2006-7


An example

Modeling the English premiership football data 2006-2007

- ✓ We use the simple vanilla model
- ✓ Data are available in engsoccerdata package in R
- ✓ You can find details for the Bayesian implementation of this model using WinBUGS in my book:
  - ✓Ntzoufras (2009). *Bayesian modelling using WinBUGS* (Section 7.4.3, p. 249-257).

# Fitting the model in R using glm

## Fitting the model in R using glm

- An example of data

```
> head(chap07_ex2_soccer)
         team1 goals1 goals2         team2          ht            at z
1 Sheff Utd          1      1 Liverpool      Sheff Utd     Liverpool 0
2 Arsenal            1      1 Aston Villa      Arsenal   Aston Villa 0
3 Everton            2      1 Watford          Everton       Watford 1
4 Newcastle          2      1 Wigan          Newcastle         Wigan 1
5 Portsmouth         3      0 Blackburn     Portsmouth     Blackburn 3
6 Reading            3      2 Middlesbrough    Reading Middlesbrough 1
```

# Fitting the model in R using glm

**Fitting the model in R using glm**


Transform data for Poisson glm

- One response -> Goals= (goals1,goals2)
- Each game -> 2 lines in the dataset
- Add home team indicator

```
> head(chap07_ex2_soccer)
          team1 goals1 goals2          team2          ht                at z
1 Sheff Utd           1      1 Liverpool      Sheff Utd     Liverpool 0
2 Arsenal             1      1 Aston Villa      Arsenal   Aston Villa 0
3 Everton             2      1 Watford          Everton       Watford 1
4 Newcastle           2      1 Wigan          Newcastle         Wigan 1
5 Portsmouth          3      0 Blackburn     Portsmouth     Blackburn 3
6 Reading             3      2 Middlesbrough    Reading Middlesbrough 1
```

# Fitting the model in R using glm

## Fitting the model in R using glm

```
n<-nrow(chap07_ex2_soccer)

all(levels(chap07_ex2_soccer$ht)==levels(chap07_ex2_soccer$at))

goals <-c(chap07_ex2_soccer$goals1,chap07_ex2_soccer$goals2)
game <- c(1:n, 1:n )
home <- c( rep(1,n), rep(0,n)  )
att <- factor(c(chap07_ex2_soccer$ht, chap07_ex2_soccer$at) )
def <- factor(c(chap07_ex2_soccer$at, chap07_ex2_soccer$ht))

levels(att) <- levels(chap07_ex2_soccer$ht)
levels(def) <- levels(chap07_ex2_soccer$ht)
```

# Fitting the model in R using glm

## Fitting the model in R using glm

```
premier <- data.frame( game=game, att=att, def=def, home=home,
goals=goals)
head(premier)

i<-order(game)
premier<-premier[i,]
head(premier)
```

# Fitting the model in R using glm

## Fitting the model in R using glm

- An example of data

```
> head(chap07_ex2_soccer)
        team1 goals1 goals2        team2          ht          at z
1 Sheff Utd       1      1 Liverpool    Sheff Utd    Liverpool 0
2 Arsenal         1      1 Aston Villa  Arsenal      Aston Villa 0
3 Everton         2
4 Newcastle       2
5 Portsmouth      3
6 Reading         3
```

```
> head(premier)
    game         att         def home goals
1      1   Sheff Utd    Liverpool    1     1
381    1   Liverpool    Sheff Utd    0     1
2      2     Arsenal Aston Villa    1     1
382    2 Aston Villa      Arsenal    0     1
3      3     Everton      Watford    1     2
383    3     Watford      Everton    0     1
```

# Fitting the model in R using glm

## Fitting the model in R using glm

```
contrasts(premier$att)<-contr.sum(20)
contrasts(premier$def)<-contr.sum(20)
model <- glm( goals~home+att+def, family=poisson, data=premier )
summary(model)
round(summary(model)$coef,3)
```

# Fitting the model in R using glm

## Model parameters of DP in R

```
> round(summary(model)$coef,3)
```

| | Estimate | Std. Error | z value | Pr(>\|z\|) | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| (Intercept) | -0.075 | 0.053 | -1.413 | 0.158 | | | | | |
| home | 0.376 | 0.067 | 5.637 | 0.000 | | | | | |
| att1 | 0.329 | 0.124 | 2.646 | 0.008 | def1 | -0.231 | 0.164 | -1.409 | 0.159 |
| att2 | -0.047 | 0.149 | -0.314 | 0.753 | def2 | -0.095 | 0.152 | -0.627 | 0.531 |
| att3 | 0.159 | 0.136 | 1.165 | 0.244 | def3 | 0.191 | 0.134 | 1.428 | 0.153 |
| att4 | 0.055 | 0.143 | 0.385 | 0.701 | def4 | 0.147 | 0.136 | 1.083 | 0.279 |
| att5 | -0.261 | 0.166 | -1.566 | 0.117 | def5 | 0.276 | 0.127 | 2.169 | 0.030 |
| att6 | 0.333 | 0.123 | 2.693 | 0.007 | def6 | -0.608 | 0.197 | -3.092 | 0.002 |
| att7 | 0.138 | 0.136 | 1.015 | 0.310 | def7 | -0.216 | 0.162 | -1.332 | 0.183 |
| att8 | -0.149 | 0.158 | -0.945 | 0.345 | def8 | 0.281 | 0.127 | 2.204 | 0.027 |
| att9 | 0.220 | 0.130 | 1.688 | 0.091 | def9 | -0.498 | 0.186 | -2.680 | 0.007 |
| att10 | -0.438 | 0.180 | -2.440 | 0.015 | def10 | -0.040 | 0.147 | -0.274 | 0.784 |
| att11 | 0.597 | 0.110 | 5.438 | 0.000 | def11 | -0.469 | 0.186 | -2.520 | 0.012 |
| att12 | -0.015 | 0.147 | -0.100 | 0.921 | def12 | 0.084 | 0.140 | 0.603 | 0.546 |
| att13 | -0.164 | 0.158 | -1.039 | 0.299 | def13 | 0.036 | 0.143 | 0.251 | 0.802 |
| att14 | 0.000 | 0.146 | -0.001 | 1.000 | def14 | -0.069 | 0.150 | -0.459 | 0.646 |
| att15 | 0.151 | 0.136 | 1.106 | 0.269 | def15 | 0.052 | 0.143 | 0.362 | 0.717 |
| att16 | -0.327 | 0.171 | -1.910 | 0.056 | def16 | 0.187 | 0.133 | 1.408 | 0.159 |
| att17 | 0.251 | 0.130 | 1.923 | 0.055 | def17 | 0.197 | 0.134 | 1.471 | 0.141 |
| att18 | -0.421 | 0.180 | -2.345 | 0.019 | def18 | 0.254 | 0.128 | 1.977 | 0.048 |
| att19 | -0.233 | 0.164 | -1.418 | 0.156 | def19 | 0.260 | 0.128 | 2.030 | 0.042 |

# Fitting the model in R using glm

## Attacking and Defensive abilities in R

```
> round(abilities,3)
                Att SD-Att    Def SD-Def
Arsenal       0.329  0.124 -0.231  0.164
Aston Villa  -0.047  0.149 -0.095  0.152
Blackburn     0.159  0.136  0.191  0.134
Bolton        0.055  0.143  0.147  0.136
Charlton     -0.261  0.166  0.276  0.127
Chelsea       0.333  0.123 -0.608  0.197
Everton       0.138  0.136 -0.216  0.162
Fulham       -0.149  0.158  0.281  0.127
Liverpool     0.220  0.130 -0.498  0.186
Man City     -0.438  0.180 -0.040  0.147
Man Utd       0.597  0.110 -0.469  0.186
Middlesbrough -0.015  0.147  0.084  0.140
Newcastle    -0.164  0.158  0.036  0.143
Portsmouth    0.000  0.146 -0.069  0.150
Reading       0.151  0.136  0.052  0.143
Sheff Utd    -0.327  0.171  0.187  0.133
Tottenham     0.251  0.130  0.197  0.134
Watford      -0.421  0.180  0.254  0.128
West Ham     -0.233  0.164  0.260  0.128
Wigan        -0.177     NA  0.263     NA
```

```
> abilities <- matrix( nrow=20,ncol=4 )
> abilities[1:19,1:2] <- summary(model)$coef[2+1:19,1:2]
> abilities[1:19,2:3] <- summary(model)$coef[21+1:19,1:2]
> abilities[20,1] <- -sum(summary(model)$coef[2+1:19,1])
> abilities[20,3] <- -sum(summary(model)$coef[21+1:19,1])
>
> rownames(abilities)<-levels(premier$att)
> colnames(abilities)<-c( "Att", "SD-Att", "Def", "SD-Def" )
> abilities
```

# Fitting the model using WinBUGS

## Fitting the model in WinBUGS

chap07_ex2_1simple_poisson_model_UK2006

```
model{
    for (i in 1:n){
        # stochastic component
        goals1[i]~dpois(lambda1[i])
        goals2[i]~dpois(lambda2[i])
        # link and linear predictor
        log(lambda1[i])<-  mu + home + a[ ht[i] ] + d[ at[i] ]
        log(lambda2[i])<-  mu        + a[ at[i] ] + d[ ht[i] ]
    }
     # STZ constraints
    a[1]<-  -sum( a[2:20] )
    d[1]<-  -sum( d[2:20] )
    #
    # prior distributions
    mu~dnorm(0,0.001)
    home~dnorm(0,0.001)
    for (i in 2:K){
        a[i]~dnorm(0,0.01)
        d[i]~dnorm(0,0.01)
    }

}
```

# Example: English Premier League 2006-7


**Results:** **Important Findings based on Ability Parameters**

- Manchester United had the highest attacking parameter.
- Chelsea had the lowest (i.e., best) defensive parameter.
- In a game between two average teams: expected score = 1.3 – 0.9 .
- Home effect: Increases the expected number of goals by 46% .

# Example: English Premier League 2006-7

**Posterior summaries of the model parameters**

| | | | Posterior | | | | | Posterior | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | percentiles | | | | | percentiles | |
| | Team[a] | Node | Mean | SD | 2.5% | 97.5% | Node | Mean | SD | 2.5% | 97.5% |
| 1. | Arsenal | a[1] | 0.33 | 0.12 | 0.08 | 0.57 | d[1] | -0.23 | 0.16 | -0.57 | 0.08 |
| 2. | Aston Villa | a[2] | -0.05 | 0.15 | -0.34 | 0.23 | d[2] | -0.10 | 0.16 | -0.42 | 0.19 |
| 3. | Blackburn | a[3] | 0.16 | 0.14 | -0.13 | 0.42 | d[3] | 0.20 | 0.14 | -0.08 | 0.45 |
| 4. | Bolton | a[4] | 0.06 | 0.14 | -0.23 | 0.32 | d[4] | 0.15 | 0.14 | -0.13 | 0.41 |
| 5. | Charlton | a[5] | -0.26 | 0.16 | -0.61 | 0.04 | d[5] | 0.28 | 0.13 | 0.03 | 0.52 |
| 6. | Chelsea | a[6] | 0.33 | 0.12 | 0.08 | 0.56 | d[6] | -0.62 | 0.20 | -1.04 | -0.25 |
| 7. | Everton | a[7] | 0.14 | 0.14 | -0.13 | 0.41 | d[7] | -0.22 | 0.17 | -0.56 | 0.09 |
| 8. | Fulham | a[8] | -0.14 | 0.16 | -0.47 | 0.16 | d[8] | 0.29 | 0.13 | 0.02 | 0.53 |
| 9. | Liverpool | a[9] | 0.22 | 0.13 | -0.04 | 0.47 | d[9] | -0.51 | 0.19 | -0.90 | -0.16 |
| 10. | Man City | a[10] | -0.44 | 0.18 | -0.82 | -0.09 | d[10] | -0.04 | 0.15 | -0.34 | 0.24 |
| 11. | Man Utd | a[11] | 0.60 | 0.11 | 0.38 | 0.81 | d[11] | -0.47 | 0.19 | -0.85 | -0.12 |
| 12. | Middlesbrough | a[12] | -0.02 | 0.15 | -0.33 | 0.26 | d[12] | 0.09 | 0.14 | -0.20 | 0.35 |
| 13. | Newcastle | a[13] | -0.16 | 0.16 | -0.48 | 0.13 | d[13] | 0.03 | 0.14 | -0.26 | 0.31 |
| 14. | Portsmouth | a[14] | 0.00 | 0.15 | -0.29 | 0.29 | d[14] | -0.07 | 0.15 | -0.37 | 0.22 |
| 15. | Reading | a[15] | 0.15 | 0.14 | -0.13 | 0.40 | d[15] | 0.05 | 0.15 | -0.24 | 0.33 |
| 16. | Sheff Utd | a[16] | -0.33 | 0.17 | -0.68 | -0.01 | d[16] | 0.19 | 0.13 | -0.08 | 0.44 |
| 17. | Tottenham | a[17] | 0.26 | 0.13 | -0.00 | 0.51 | d[17] | 0.20 | 0.13 | -0.06 | 0.45 |
| 18. | Watford | a[18] | -0.42 | 0.18 | -0.78 | -0.07 | d[18] | 0.25 | 0.13 | -0.01 | 0.50 |
| 19. | West Ham | a[19] | -0.24 | 0.16 | -0.56 | 0.07 | d[19] | 0.27 | 0.13 | 0.01 | 0.51 |
| 20. | Wigan | a[20] | -0.19 | 0.16 | -0.51 | 0.11 | d[20] | 0.27 | 0.13 | 0.01 | 0.51 |
| | | home | 0.38 | 0.07 | 0.25 | 0.51 | $\mu$ | -0.10 | 0.05 | -0.20 | 0.003 |

# Example: English Premier League 2006-7

**Posterior summaries of the attacking parameters**

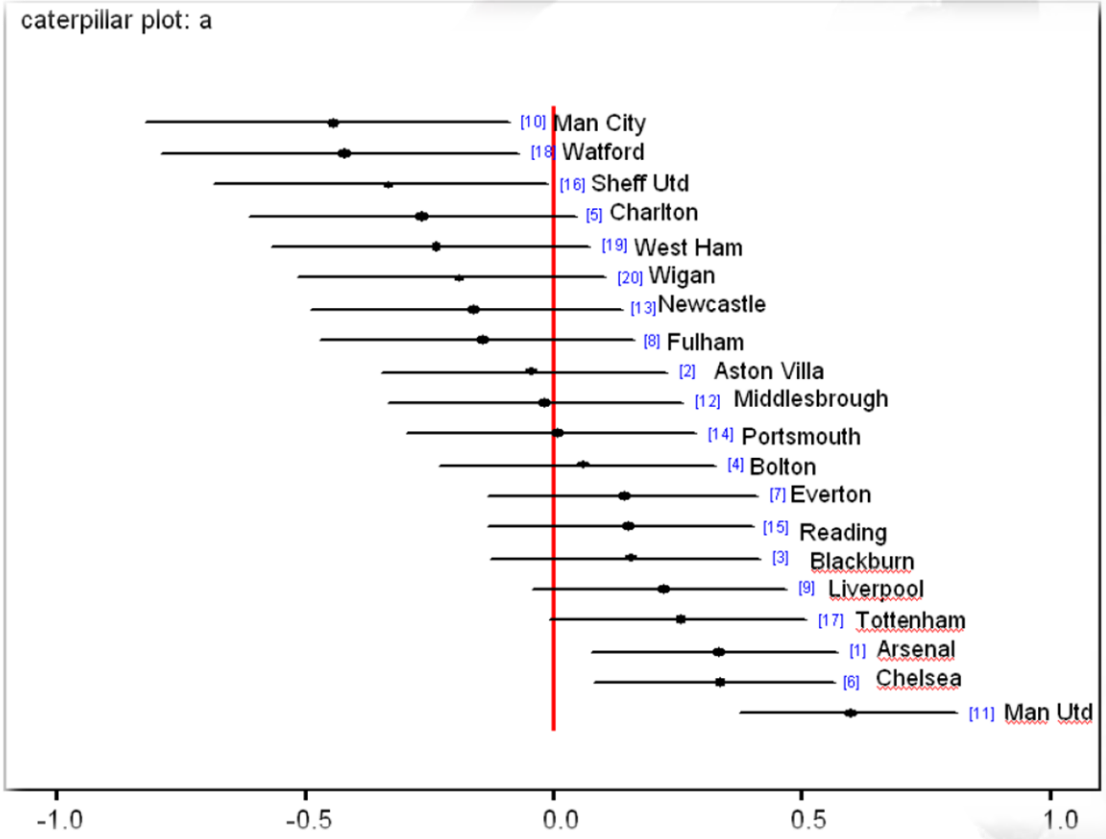



Figure: 95% posterior intervals for team attacking parameters for Premier League 2006-7

Abbreviations:

1. Man = Manchester;
2. Utd = United;
3. Sheff = Sheffield;
4. Ham = Hampshire.

# Example: English Premier League 2006-7

**Posterior summaries of the defensive parameters**

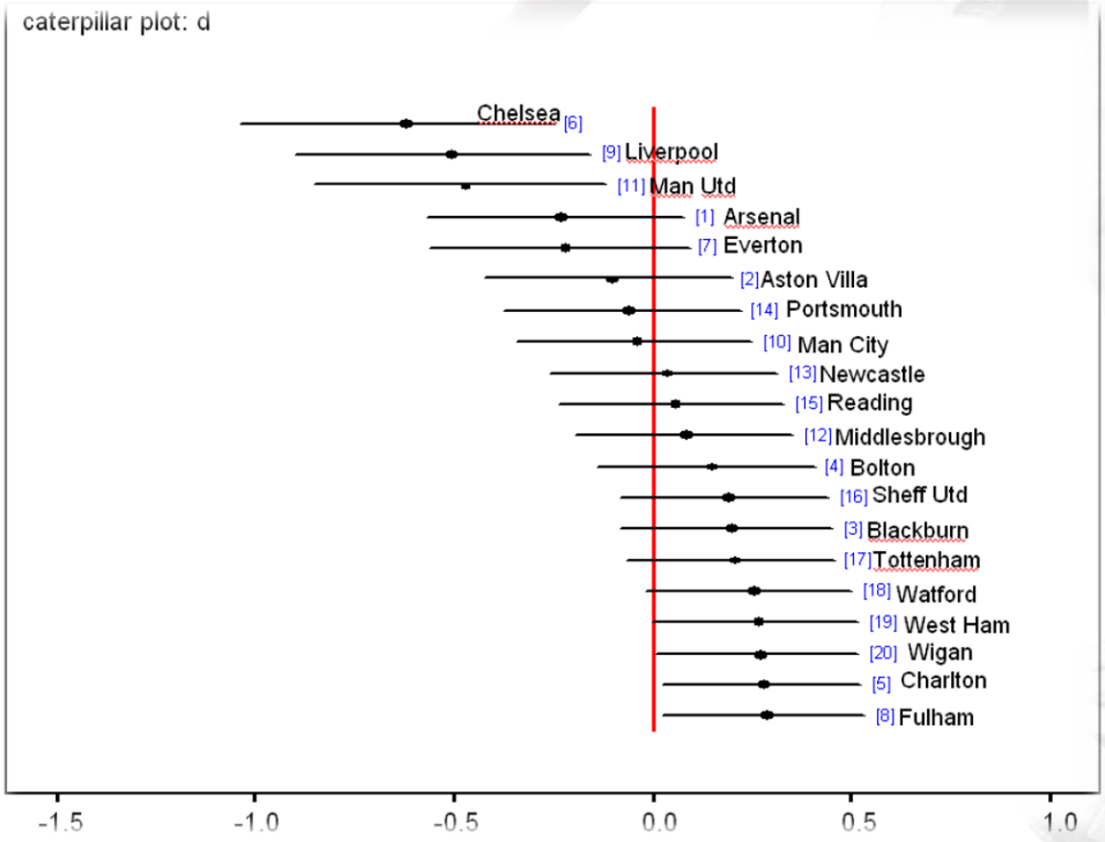



Figure: 95% posterior intervals for team defensive parameters for Premier League 2006-7

Abbreviations:

1. Man = Manchester;
2. Utd = United;
3. Sheff = Sheffield;
4. Ham = Hampshire.

# Example: English Premier League 2006-7


## Prediction of future games

- How to obtain predictions for (two) future games.
- The approach can be easily generalized for any games
- In the Bayesian approach
  - Any missing parameter is generated from the predictive distribution
  - In MCMC, we generate predicted goals of the game of interest by simply
    - calculating the expected goals for this game at the current observation, and then
    - generate predicted goals from the corresponding sampling distribution – here the Poisson
  - In WinBUGS, just we put NA at the goals of the game we want to predict

# Example: English Premier League 2006-7


## Prediction of future games

- In this example: we replaced with NA the goals of the last two games in the dataset
  - Tottenham – Manchester City and
  - Watford – Newcastle) by NA.
- WinBUGS automatically generates values for the missing goals from the predictive distribution and will provide estimates for each score by monitoring the nodes goals1 and goals2.

# Example: English Premier League 2006-7


## Prediction of future games

- In this example: we replaced with NA the goals of the last two games in the dataset
  - Tottenham – Manchester City and
  - Watford – Newcastle) by NA.
- WinBUGS automatically generates values for the missing goals from the predictive distribution and will provide estimates for each score by monitoring the nodes goals1 and goals2.

# Example: English Premier League 2006-7

## Prediction of future games

Posterior summaries of the expected scores for last two games

| | | | | | Posterior summaries of goal difference | | |
|---|---|---|---|---|---|---|---|
| | | Actual | Posterior | | | | |
| Home team | Away team | score | Median | Mean | Mean | SD | 95% CI |
| 379. Tottenham | Manchester City | 2–1 | 1–1 | 1.65 – 0.72 | 0.93 | 1.59 | (-2, 4) |
| 380. Watford | Newcastle | 1–1 | 1–1 | 0.93 – 1.02 | -0.09 | 1.43 | (-3, 3) |

# Example: English Premier League 2006-7

## Prediction of future games

Posterior probabilities of each game outcome for last two games

| | | | | Posterior Probability | | |
|---|---|---|---|---|---|---|
| | Home team | Away team | Actual score | Home wins | Draw | Away wins |
| 379. | Tottenham | Manchester City | 2–1 | 0.59 | 0.24 | 0.17 |
| 380. | Watford | Newcastle | 1–1 | 0.33 | 0.30 | 0.37 |

- Tottenham – Man City: 60% probability for Tottenham to win
- Watford – Newcastle: The two teams are of equal strength (~equal probabilities for the 3 outcomes)

# Example: English Premier League 2006-7


**Prediction of future games**

WinBUGS code for predictions

```
# calculation of the predicted differences
pred.diff[1] <- goals1[379]-goals2[379]
pred.diff[2] <- goals1[380]-goals2[380]
#
# probability of each game outcome (win/draw/loss)
for (i in 1:2){
    outcome[i,1] <- 1 - step( -pred.diff[i] )    #home wins (diff>0)
    outcome[i,2] <- equals( pred.diff[i] , 0.0 )#draw (diff=0)
    outcome[i,3] <- 1-step( pred.diff[i] )      #home loses (diff<0)
}
```

# Example: English Premier League 2006-7

## Prediction of future games

R code for predictions

```
i1<- premier$game<379
premier2<-premier[i1,]
model2 <- glm( goals~home+att+def, family=poisson, data=premier2 )
lambda<-exp(predict(model2, premier[!i1,]))
names(lambda)<-levels(premier$att)[c(17,10,18,13)]
lambda
np <- nrow(premier[!i1,])
B<-10000
G<-matrix(nrow=B, ncol=np)
for (i in 1:B){ G[i,]<-rpois(np, lambda)  }
i2<-seq(1,np,2)
diff <- G[,i2] - G[,i2+1]
apply(diff,2,mean)
apply(diff,2,sd)
t(apply(diff,2,quantile, probs=c(0.025,0.975)))
```

```
> lambda
Tottenham  Man City   Watford Newcastle
1.6444403 0.7181869 0.9155914 1.0163420
```

```
> apply(diff,2,mean)
[1]  0.9061 -0.0855
> apply(diff,2,sd)
[1] 1.571283 1.397063
> t(apply(diff,2,quan
     2.5% 97.5%
[1,]   -2     4
[2,]   -3     3
```

# Example: English Premier League 2006-7

## Prediction of future games

Posterior summaries of the expected scores for last two games

| Home team | Away team | Actual score | Posterior Median | Posterior Mean | Posterior summaries of goal difference Mean | SD | 95% CI |
|---|---|---|---|---|---|---|---|
| 379. Tottenham | Manchester City | 2–1 | 1–1 | 1.65 – 0.72 | 0.93 | 1.59 | (-2, 4) |
| 380. Watford | Newcastle | 1–1 | 1–1 | 0.93 – 1.02 | -0.09 | 1.43 | (-3, 3) |

```
> noquote(tabres)
    Home Team Away Team Score Median Expected  Exp.Diff SD Diff 95% CI Diff
379 Tottenham Man City  2-1   1-1    1.64-0.72 0.91     1.57    (-2,4)
380 Watford   Newcastle 1-1   1-1    0.92-1.02 -0.09    1.4     (-3,3)
```

# Example: English Premier League 2006-7

## Prediction of future games

R code for predictions

```
probs<-rbind(
                c(mean(diff[,1]>0), mean(diff[,1]==0), mean(diff[,1]<0)),
                c(mean(diff[,2]>0), mean(diff[,2]==0), mean(diff[,2]<0))
)
probs
```

```
> probs
        [,1]   [,2]   [,3]
[1,] 0.5869 0.2467 0.1664
[2,] 0.3238 0.3124 0.3638
```

# Example: English Premier League 2006-7

## Prediction of future games

Posterior probabilities of each game outcome for last two games

| | | | | Posterior Probability | | |
|---|---|---|---|---|---|---|
| | Home team | Away team | Actual score | Home wins | Draw | Away wins |
| 379. | Tottenham | Manchester City | 2–1 | 0.59 | 0.24 | 0.17 |
| 380. | Watford | Newcastle | 1–1 | 0.33 | 0.30 | 0.37 |

```
> noquote(tabres2)
    Home Team Away Team Score HomeWins Draw AwayWins
379 Tottenham Man City  2-1   0.59     0.25 0.17
380 Watford   Newcastle 1-1   0.32     0.31 0.36
```

# Example: English Premier League 2006-7

## Prediction of future games

Posterior probabilities of each game goal difference for last two games

| | Home Team | Away Team | Actual Score | Posterior Probability of goal difference[a] | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | $\leq -3$ | -2 | -1 | 0 | 1 | 2 | 3 | 4 | $\geq 5$ |
| 379. | Tottenham | Man City | 2–1 | 0.012 | 0.036 | 0.119 | 0.242 | **0.257** | 0.185 | 0.090 | 0.037 | 0.023 |
| 380. | Watford | Newcastle | 1–1 | 0.042 | 0.108 | 0.218 | **0.303** | 0.210 | 0.086 | 0.024 | 0.006 | 0.002 |

## Posterior mode

- Tottehnam vs. Man. City $\rightarrow$ One goal difference.
- Watford vs. Newcastle $\rightarrow$ Zero goal difference.

# Example: English Premier League 2006-7

## Prediction of future games

WinBUGS code for goal difference frequencies

```
# calculation of the probability of each difference
for (i in 1:2){
    pred.diff.counts[i,1]<- 1-step(pred.diff[i]+5) # less than -5
    # equal to k-7 (-5 to 5)
    for (k in 2:12){
        pred.diff.counts[i,k]<-equals(pred.diff[i],k-7)}
    pred.diff.counts[i,13]<-step(pred.diff[i]-6) # greater than 5
}
```

- ✓ In this syntax pred.diff.counts is again a matrix with binary elements indicating which difference appears in each MCMC iteration.
- ✓ Elements 2–12 denote differences from -5 to 5, while the first and last elements denote differences lower than -5 and higher than 5, respectively.

# Example: English Premier League 2006-7

## Prediction of future games

Posterior probabilities of each game goal difference for last two games

| | Home Team | Away Team | Actual Score | Posterior Probability of goal difference[a] | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | $\leq -3$ | -2 | -1 | 0 | 1 | 2 | 3 | 4 | $\geq 5$ |
| 379. | Tottenham | Man City | 2–1 | 0.012 | 0.036 | 0.119 | 0.242 | **0.257** | 0.185 | 0.090 | 0.037 | 0.023 |
| 380. | Watford | Newcastle | 1–1 | 0.042 | 0.108 | 0.218 | **0.303** | 0.210 | 0.086 | 0.024 | 0.006 | 0.002 |

```
> round(table(diff[,1])/B,3)

   -5    -4    -3    -2    -1     0     1     2     3     4     5     6     7     8
0.000 0.001 0.010 0.040 0.115 0.247 0.260 0.179 0.094 0.038 0.011 0.004 0.001 0.000
> round(table(diff[,2])/B,3)

   -6    -5    -4    -3    -2    -1     0     1     2     3     4     5     6
0.000 0.002 0.008 0.031 0.101 0.222 0.312 0.208 0.087 0.023 0.005 0.001 0.000
```

# Example: English Premier League 2006-7


## Regeneration of the full league

- Can be used as GOF
- Given that the model fits well, It can be used to quantify uncertainty for given quantities of interest
- So, we can obtain average, Median, SD, 95% Cis of
  - Points
  - Goals (not for multinomial models)
  - Goal differences
  - Rankings

# Example: English Premier League 2006-7

## Regeneration of the full league

| Predicted (actual) ranking[a] | Team | Actual points | Posterior summaries for total points | | | | | Posterior percentiles for points ranks[b] | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Mean | SD | 2.5% | Median | 97.5% | 2.5% | Median | 97.5% |
| 1 (1) | Man Utd | 89 | 84.7 | 8.1 | 68 | 85 | 99 | 16 | 20 | 20 |
| 2 (2) | Chelsea | 83 | 78.3 | 8.8 | 60 | 79 | 94 | 14 | 19 | 20 |
| 3 (3) | Liverpool | 68 | 72.6 | 9.2 | 54 | 73 | 90 | 12 | 18 | 20 |
| 4 (3) | Arsenal | 68 | 69.9 | 9.4 | 51 | 70 | 88 | 10 | 17 | 20 |
| 5 (6) | Everton | 58 | 62.8 | 9.4 | 44 | 63 | 81 | 7 | 15 | 19 |
| 6 (8) | Reading | 55 | 55.8 | 9.6 | 37 | 56 | 75 | 4 | 13 | 18 |
| 7 (5) | Tottenham | 60 | 56.0 | 9.6 | 36 | 55 | 73 | 4 | 12 | 18 |
| 8 (9) | Portsmouth | 54 | 54.3 | 9.6 | 36 | 54 | 73 | 3 | 12 | 18 |
| 9 (11) | Aston Villa | 50 | 53.4 | 9.7 | 34 | 53 | 73 | 3 | 12 | 18 |
| 10 (10) | Blackburn | 52 | 51.9 | 9.6 | 33 | 52 | 70 | 3 | 11 | 17 |

# Example: English Premier League 2006-7

## Regeneration of the full league

| Predicted (actual) ranking[a] | Team | Actual points | Posterior summaries for total points | | | | | Posterior percentiles for points ranks[b] | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | Mean | SD | 2.5% | Median | 97.5% | 2.5% | Median | 97.5% |
| 11 (7) | Bolton | 56 | 49.6 | 9.3 | 32 | 49 | 69 | 2 | 10 | 17 |
| 12 (12) | Middlesbrough | 46 | 49.2 | 9.4 | 32 | 49 | 68 | 2 | 10 | 17 |
| 13 (13) | Newcastle | 43 | 46.0 | 9.4 | 28 | 46 | 65 | 1 | 8 | 16 |
| 14 (14) | Man City | 42 | 40.8 | 9.0 | 24 | 40 | 59 | 1 | 6 | 14 |
| 15 (16) | Fulham | 39 | 39.3 | 8.9 | 22 | 39 | 57 | 1 | 5 | 13 |
| 16 (17) | Wigan | 38 | 38.8 | 9.1 | 22 | 39 | 57 | 1 | 5 | 13 |
| 17 (15) | West Ham | 41 | 37.5 | 8.7 | 21 | 37 | 55 | 1 | 4 | 13 |
| 18 (18) | Sheff Utd | 38 | 37.2 | 8.8 | 21 | 37 | 55 | 1 | 4 | 12 |
| 19 (19) | Charlton | 34 | 36.4 | 8.9 | 19 | 36 | 55 | 1 | 4 | 12 |
| 20 (20) | Watford | 28 | 33.2 | 8.6 | 17 | 33 | 51 | 1 | 3 | 11 |

[a] Predicted ranks are calculated using the median rank and then the mean points.
[b] *Ranks* here refers to the number of points in ascending order (e.g., 20 denotes the best team and 1, the worst team in terms of collected points).
*Abbreviations*: Man = Manchester; Utd = United; Sheff = Sheffield; Ham = Hampshire.

# Example: English Premier League 2006-7

## Regeneration of the full league

| Node | Posterior mean | Posterior SD | Posterior probability of each ranking[a] 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Man Utd | 1.7 | 1.1 | **60** | 23 | 10 | 4 | 1 | 1 | | | | | | | | | | | | | | |
| Chelsea | 2.6 | 1.6 | 25 | **33** | 21 | 11 | 6 | 2 | 1 | 1 | | | | | | | | | | | | |
| Liverpool | 3.7 | 2.0 | 10 | 21 | **25** | 18 | 11 | 7 | 3 | 3 | 1 | 1 | | | | | | | | | | |
| Arsenal | 4.3 | 2.3 | 6 | 15 | **21** | **21** | 13 | 9 | 6 | 4 | 2 | 1 | 1 | 1 | | | | | | | | |
| Everton | 6.2 | 2.9 | 1 | 5 | 10 | 16 | **17** | 13 | 10 | 8 | 6 | 5 | 3 | 2 | 1 | 1 | 1 | | | | | |
| Reading | 8.5 | 3.6 | | 2 | 4 | 6 | 10 | **12** | 11 | 11 | 9 | 9 | 7 | 6 | 4 | 3 | 3 | 2 | 1 | 1 | 1 | |
| Tottenham | 8.8 | 3.6 | | 1 | 3 | 6 | 10 | 11 | **12** | 10 | 10 | 9 | 7 | 6 | 4 | 4 | 3 | 2 | 1 | 1 | 1 | |
| Portsmouth | 9.0 | 3.7 | | 1 | 2 | 6 | 8 | 11 | 10 | **11** | 9 | 9 | 8 | 6 | 5 | 4 | 3 | 2 | 2 | 1 | 1 | |
| Aston Villa | 9.5 | 3.8 | | 1 | 2 | 5 | 7 | 9 | 10 | **11** | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 2 | 1 | |
| Blackburn | 9.9 | 3.8 | | 1 | 2 | 4 | 7 | 8 | 9 | 9 | **10** | **10** | 9 | 8 | 7 | 5 | 5 | 3 | 2 | 1 | 1 | 1 |
| Bolton | 10.8 | 3.8 | | | 1 | 2 | 4 | 6 | 7 | **9** | **9** | **9** | **9** | **9** | 8 | 7 | 6 | 4 | 3 | 2 | 2 | 1 |
| Middlesbrough | 11.0 | 3.9 | | | 1 | 2 | 4 | 6 | 7 | 8 | **9** | **9** | **9** | **9** | **9** | 7 | 5 | 5 | 4 | 3 | 2 | 1 |
| Newcastle | 12.3 | 3.9 | | | | 1 | 2 | 3 | 5 | 6 | 8 | 8 | 9 | 9 | **10** | 8 | 9 | 6 | 6 | 4 | 4 | 2 |
| Man City | 14.4 | 3.7 | | | | | 1 | 1 | 2 | 3 | 4 | 6 | 6 | 7 | 8 | 9 | 10 | **11** | 10 | 9 | 8 | 6 |
| Fulham | 14.9 | 3.6 | | | | | | 1 | 2 | 2 | 3 | 4 | 5 | 7 | 8 | 9 | 9 | 10 | **11** | **11** | 10 | 8 |
| Wigan | 15.1 | 3.6 | | | | | 1 | 1 | 2 | 2 | 4 | 4 | 5 | 5 | 8 | 10 | 9 | 10 | **11** | 10 | 10 | 10 |
| West Ham | 15.6 | 3.4 | | | | | | 1 | 1 | 1 | 2 | 3 | 4 | 6 | 7 | 8 | 9 | 11 | 11 | **12** | **12** | 11 |
| Sheff Utd | 15.8 | 3.3 | | | | | | | 1 | 2 | 3 | 3 | 4 | 5 | 7 | 8 | 9 | 10 | 12 | **13** | 12 | 12 |
| Charlton | 16.0 | 3.3 | | | | | | | 1 | 2 | 2 | 3 | 3 | 5 | 6 | 8 | 8 | 10 | 12 | 12 | 14 | **15** |
| Watford | 17.1 | 2.9 | | | | | | | | 1 | 1 | 2 | 2 | 3 | 4 | 5 | 8 | 8 | 10 | 14 | 18 | **24** |

Boldface indicates highest probability for each team. Posterior percentages were rounded to the closest integer, while percentages $< 0.5\%$ were omitted. Sums of probabilities for each column are slightly higher than 100%, due to ties.

*Abbreviations*: Man = Manchester; Utd = United; Sheff = Sheffield; Ham = Hamp

# Example: English Premier League 2006-7

## Regeneration of the full league

```
> noquote(summary.table2)
    Team           Exp-P SD-P 95%LB-P Med-P 95%UB-P 95%LB-R Med.R 95%UB-R
1   Man Utd        85.3  6.7  72      85     98      1       1     3
2   Chelsea        78.9  7    65      79     92      1       2     5
3   Liverpool      72.7  7.3  58      73     87      1       3     7
4   Arsenal        70.4  7.2  56      70     84      1       4     8
5   Everton        62.9  7.5  48      63     78      2       5     12
6   Reading        55.8  7.6  41      56     71      4       8     15
7   Tottenham      54.9  7.6  40      55     70      4       8     15
8   Portsmouth     54.3  7.5  40      54     69      4       8     16
9   Aston Villa    53.3  7.5  39      53     68      4       9     16
10  Blackburn      51.8  7.5  37      52     67      4       9     16
11  Bolton         49.7  7.5  35      50     64      5       11    18
12  Middlesbrough  49.1  7.5  35      49     64      5       11    18
13  Newcastle      46.1  7.3  32      46     61      6       12    19
14  Man City       40.6  7.1  27      40     55      8       15    20
15  Fulham         39    7.1  26      39     53      9       16    20
16  Wigan          38.8  7.1  25      39     53      9       16    20
17  West Ham       37.3  7.1  24      37     52      9       17    20
18  Sheff Utd      36.8  6.9  24      37     51      10      17    20
19  Charlton       36    7    23      36     50      10      17    20
20  Watford        32.7  6.8  20      33     47      12      18    20
```

# Example: English Premier League 2006-7

## Regeneration of the full league

```
> tabrankings
```

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Man Utd | 68.7 | 22.4 | 6.6 | 1.7 | 0.3 | 0.1 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Chelsea | 24.5 | 42.4 | 20.4 | 8.6 | 2.8 | 0.9 | 0.3 | 0.1 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Liverpool | 7.3 | 19.5 | 31.8 | 23.4 | 10.4 | 4.7 | 1.8 | 0.6 | 0.4 | 0.2 | 0.1 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Arsenal | 3.8 | 13.7 | 26.0 | 29.3 | 14.8 | 6.5 | 3.0 | 1.4 | 0.7 | 0.3 | 0.2 | 0.1 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Everton | 0.5 | 2.6 | 9.4 | 16.8 | 25.0 | 16.6 | 10.5 | 6.8 | 4.5 | 2.9 | 1.8 | 1.1 | 0.8 | 0.4 | 0.1 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 |
| Reading | 0.0 | 0.4 | 1.9 | 5.3 | 11.4 | 14.2 | 14.1 | 12.4 | 10.8 | 8.6 | 6.6 | 5.2 | 3.8 | 2.5 | 1.2 | 0.8 | 0.4 | 0.2 | 0.1 | 0.1 |
| Tottenham | 0.0 | 0.3 | 1.6 | 4.1 | 9.6 | 12.9 | 13.8 | 12.9 | 10.8 | 10.0 | 7.3 | 5.8 | 4.2 | 2.7 | 1.6 | 1.1 | 0.7 | 0.4 | 0.1 | 0.0 |
| Portsmouth | 0.0 | 0.3 | 1.4 | 3.8 | 8.1 | 12.5 | 12.8 | 12.7 | 11.7 | 9.8 | 8.8 | 6.2 | 4.4 | 3.0 | 2.0 | 1.2 | 0.7 | 0.4 | 0.2 | 0.0 |
| Aston Villa | 0.0 | 0.2 | 0.8 | 3.1 | 7.1 | 10.7 | 11.6 | 12.6 | 11.9 | 11.0 | 9.3 | 7.0 | 5.4 | 3.6 | 2.5 | 1.4 | 0.9 | 0.6 | 0.2 | 0.1 |
| Blackburn | 0.0 | 0.1 | 0.6 | 2.2 | 5.3 | 8.5 | 11.4 | 11.1 | 11.9 | 10.9 | 9.9 | 8.2 | 6.9 | 4.9 | 3.7 | 2.0 | 1.3 | 0.8 | 0.3 | 0.1 |
| Bolton | 0.0 | 0.0 | 0.3 | 1.2 | 3.1 | 6.4 | 8.2 | 9.4 | 10.5 | 10.8 | 11.0 | 10.2 | 8.8 | 6.7 | 5.0 | 3.3 | 2.3 | 1.8 | 0.8 | 0.3 |
| Middlesbrough | 0.0 | 0.0 | 0.3 | 1.0 | 3.1 | 5.0 | 7.6 | 9.2 | 10.2 | 11.0 | 10.6 | 10.8 | 8.7 | 7.3 | 5.6 | 3.6 | 2.6 | 1.7 | 1.1 | 0.5 |
| Newcastle | 0.0 | 0.0 | 0.1 | 0.4 | 1.4 | 2.9 | 4.0 | 5.5 | 7.7 | 9.6 | 10.1 | 11.3 | 10.8 | 10.0 | 8.2 | 6.7 | 5.1 | 3.0 | 2.1 | 1.0 |
| Man City | 0.0 | 0.0 | 0.0 | 0.0 | 0.2 | 0.6 | 1.0 | 2.0 | 2.8 | 3.9 | 5.9 | 7.5 | 9.4 | 11.2 | 11.8 | 11.7 | 11.3 | 8.9 | 7.0 | 4.8 |
| Fulham | 0.0 | 0.0 | 0.0 | 0.0 | 0.1 | 0.3 | 0.7 | 1.2 | 2.1 | 3.3 | 4.5 | 6.2 | 8.0 | 9.8 | 11.2 | 11.8 | 11.9 | 11.1 | 10.2 | 7.5 |
| Wigan | 0.0 | 0.0 | 0.0 | 0.0 | 0.1 | 0.2 | 0.8 | 1.2 | 1.9 | 3.1 | 4.2 | 5.9 | 8.0 | 9.2 | 11.3 | 12.4 | 12.3 | 11.2 | 9.9 | 8.2 |
| West Ham | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.2 | 0.4 | 0.9 | 1.4 | 2.0 | 3.4 | 4.8 | 6.2 | 8.0 | 10.9 | 11.7 | 12.8 | 12.8 | 13.0 | 11.5 |
| Sheff Utd | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.2 | 0.3 | 0.6 | 1.1 | 1.6 | 2.8 | 4.3 | 6.0 | 8.9 | 10.1 | 11.5 | 12.2 | 14.0 | 14.0 | 12.3 |
| Charlton | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.1 | 0.2 | 0.6 | 0.8 | 1.5 | 2.4 | 3.8 | 5.2 | 7.2 | 9.4 | 11.3 | 12.5 | 14.3 | 16.1 | 14.7 |
| Watford | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.1 | 0.2 | 0.3 | 0.6 | 0.8 | 2.0 | 2.8 | 4.1 | 6.0 | 8.5 | 10.7 | 14.1 | 20.1 | 29.7 |

# Example: English Premier League 2006-7

## Regeneration of the full league

### WinBUGS code for league re-generation

✓ In order to reconstruct the full table in WinBUGS, we need to replicate the full scores in a tabular K × K format and then calculate the number of points for each team.

```
for (i in 1:K){ for (j in 1:K){
    # replicated league
    goals1.rep[i,j]~dpois(lambda1.rep[i,j])
    goals2.rep[i,j]~dpois(lambda2.rep[i,j])
    # link and linear predictor
    log(lambda1.rep[i,j])<-  mu + home + a[ i ] + d[ j ]
    log(lambda2.rep[i,j])<-  mu        + a[ j ] + d[ i ]
    # replicated difference
    goal.diff.rep[i,j] <- goals1.rep[i,j]-goals2.rep[i,j]
}
```

# Example: English Premier League 2006-7

## Regeneration of the full league

## WinBUGS code for points calculation

✓ The total number of points is calculated using the following syntax:

```
for (i in 1:K){ for (j in 1:K){
    # points earned by each home team (i)
    points1[i,j]<-3 * (1-step(-goal.diff.rep[i,j]))
                    + equals(goal.diff.rep[i,j],0)
    # points earned by each away team (j)
    points2[i,j]<-3 * (1-step( goal.diff.rep[i,j]))
                    + equals(goal.diff.rep[i,j],0)
}}
# calculation of the total points for each team
for (i in 1:K){
    total.points[i] <- sum( points1[i,1:20] ) -  points1[i,i]
                     + sum( points2[1:20,i] ) -  points2[i,i] }
```

1. ***points1 and points2*** *= number of points in each game for the home and the away teams, respectively (arranged in two K × K matrices).*
2. ***Total points of i team =*** *points for team i in home games (sum of i row of node points1) plus points in away games (sum of j column of node points2).*
3. ***points1[i,i] and points2[i,i]*** *refer to each team playing against itself.*

# Example: English Premier League 2006-7


## WinBUGS files for you to play

| Name | |
|---|---|
| chap07_ex2_premier2006.R | Data |
| DNegBin_Model_UK2006.odc | Double Negative binomial model |
| DP1_Model_UK2006.odc | Double Poisson model (only) |
| DP2_Prediction_UK2006.odc | Double Poisson – prediction of last two games |
| DP3_League_UK2006.odc | Double Poisson – League regeneration |

# Example: English Premier League 2006-7


**R code files for you to play**

| | |
|---|---|
| 1_data_DP.r | Data and DP using glm function |
| 2_prediction.r | Code for prediction of the last two games |
| 3_league_table.r | Code for league re-generation |

# Summary of Lecture 2


- ✓ Basic vanilla DP model
  - ✓ Easy to fit
  - ✓ B.Sc. or M.Sc. Stats Graduate can fit it
- ✓ How to fit it Bayes and classic approach
- ✓ League regeneration for fit and prediction
- ✓ Assumptions for Football models
- ✓ Protocol for model-based Football analysis
- ✓ IMPORTANT NOTE: *Understanding the characteristics of football/sports is important in order to construct/fit a good model*