Analisis Peluang Kemenangan Tim *English Premiere League* dengan Metode GLMM (*Generelized Linear Mixed Model*)

Kelompok 2 :

Muhammad Naufal Irham Ramdhani (10818019)

Senin, 17 Mei 2021

FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM

INSTITUT TEKNOLOGI BANDUNG

2021

Latar belakang

Pertandingan olahraga merupakan salah satu rekreasi yang disenanangi oleh banyak orang, baik tua maupun muda. Salah satu cabang olahraga dengan peminat terbanyak adalah sepak bola. Hampir setiap negara, memiliki liga sepak bola masing-masing. Namun, salah satu liga yang paling ditunggu dan paling kompetitif adalah liga sepakbola inggris yaitu *English Premiere League*.

Dalam sepakbola, banyak sekali pihak yang terlibat. Mulai dari para pendukung, pelatih, pemain, bahkan pebisnis sekalipun. Sudah pasti, mereka menginginkan tim yang mereka dukung menang. Banyak faktor yang dapat menentukan apakah suatu tim akan menang atau kalah, baik faktor dari internal tim maupun eksternal tim. Disinilah letak pentingnya analisis data dalam olahraga. Dengan menganalisis data olahraga, kita dapat menentukan peluang kemenangan suatu tim. Hal ini jelas sangat bermanfaat bagi pihak-pihak terkait, terutama pelatih. Sering kali pelatih kesuliatan untuk menentukan siapa yang akan bermain dalam suatu pertandingan, apalagi jika banyak pemain yang dibekap cedera. Maka dari itu, agar dapat menimbang keputusan terbaik dalam memilih pemain, pelatih dapat menggunakan prediksi apakah pemain-pemain yang ia pilih dapat memberikan kemungkinan terbesar untuk menang.Selain itu juga, pelatih dapat menentukan faktor apa yang masih kurang maksimal dalam timnya. Sehingga pelatih dapat meminta pemilik klub untuk meningkatkan fasilitas latihan atau bahkan membeli pemain baru.

Data yang akan dianalisis adalah data pertandingan *English Premiere League* dari musim 2010-2011 sampai musim 2020-2021. Data ini bersumber dari:

https://www.kaggle.com/pablohfreitas/all-premier-league-matches-20102021

Data tersebut diambil dengan metode *web scraping* atau mengekstrak langsung data dari website resmi *English Premiere League*. Data berisi 4070 pertandingan, dengan kolom sebanyak 114 kolom. Kolom-kolom tersebut berisi semua stastik sebelum dan sesudah pertandingan untuk masing-masing tim yang bertanding, seperti penguasaan bola, jumlah tembakan tepat sasaran, jumlah tekel dll. Dari data tersebut akan diprediksi berapa kemungkinan sebuah tim akan memenangi pertandingan. Hasil akhir pertandingan hanya ada menang, seri, dan kalah. Namun pada analisis data ini, seti dan kalah digabung menjadi "tidak menang". Hal ini untuk mempermudah analis data agar data yang diolah berdistribusi bernoulli, bukan trinomial. Karena data berdistribusi bernoulli, maka akan diggunakan *Generelized Linear Model*.

Metodologi

Import Library

Library yang akan digunakan dalam analisis data ini, yakni:

```
library(lme4)
library(dplyr)
library(ggplot2)
```

```
library(pROC)
library(caret)
library(ggridges)
```

Adapun fungsi dari masing-masing *library* adalah sebagai berikut:

- lme4 : Untuk melakukan analisis GLMM
- dplyr : Untuk mentransformasi *data set*
- ggplot : Untuk membuat berbagai macam plot
- pROC : Untuk membuat kurva ROC
- caret : Untuk membuat tabel kontingensi

Import Dataset

Berikut adalah cuplikan data yang akan dianalisis

```
Data = read.csv('D:/Memento/Project/df_full_premierleague.csv')
head(Data)[,2:5]
```

```
##                                link_match season       date         home_
team
## 1 https://www.premierleague.com/match/7186  10/11 2010-11-01         Black
pool
## 2 https://www.premierleague.com/match/7404  10/11 2011-04-11         Liver
pool
## 3 https://www.premierleague.com/match/7255  10/11 2010-12-13 Manchester Un
ited
## 4 https://www.premierleague.com/match/7126  10/11 2010-09-13        Stoke
City
## 5 https://www.premierleague.com/match/7350  10/11 2011-02-14            Fu
lham
## 6 https://www.premierleague.com/match/7096  10/11 2010-08-16 Manchester Un
ited
```

Dari cuplikan *data set* diatas, kita bisa lihat darimana sumber hasil pertandingan tersebut berasal. Ada banyak sekali informasi dalam data tersebut. Namun, tidak mungkin semua informasi tersebut dimasukkan kedalam model. Model harus sederhana namun tetap dapat menjelaskan fitur-fitur suatu pertandingan secara menyeluruh. Maka dari itu, hanya beberapa informasi yang akan diambil. Diantaranya adalah:

1. Nama tim yang bertanding
2. Menang atau tidak
3. Bermain sebagai tuan rumah atau tamu
4. Lawan tanding
5. Ekspektasi penguasaan bola oleh tim
6. Ekspektasi tembakan tepat sasaran oleh tim
7. Ekspektasi tekel oleh tim
8. Ekspektasi pelangagaran oleh tim

9. Ekspektasi penguasaan bola oleh lawan
10. Ekspektasi tembakan tepat sasaran oleh lawan
11. Ekspektasi tekel oleh lawan
12. Ekspektasi pelangagaran oleh lawan
13. *Exposure*

Penguasaan bola, tekel, tembakan tepat sasaran, pelanggaran dapat menjelaskan performa tim secara keseluruhan. Tim dengan penguasaan bola yang tinggi merupakan tim yang dapat menguasai jalannya pertandingan. Tim yang banyak melakukan tekel merupakan tim yang kuat secara fisik dan lebih bermain secara keras. Tim yang banyak melakukan tembakan tepat sasaran merupakan tim yang kuat dalam menyerang. Tentu atribut-atribut tersebut harus kita tinjau dari sisi tim yang kita perhatikan dan juga lawan. *Exposure* juga dimasukkan kedalam data set karena tiap musim terdapat tim-tim yang terdegradasi, sehingga setiap tim berlaga di *English Premiere League* dengan durasi yang berbeda

Bersihkan dataset

Berikut adalah proses untuk membersihkan dataset diatas. Perhatikan proses pembersihan dataset dibawah ini. Untuk kolom "Win" dan "H_or_A" tidak secara eksplisit terdapat pada dataset asli, sehingga perlu dibangun terlebih dahulu:

```
df = data.frame(Team       = character(),
                H_or_A     = character(),
                Win        = character(),
                Opponent   = character(),
                TPosession = numeric(),
                TShoot     = numeric(),
                TTackle    = numeric(),
                TFouls     = numeric(),
                OPosession = numeric(),
                OShoot     = numeric(),
                OTackle    = numeric(),
                OFouls     = numeric(),
                Exposure   = numeric())
Data = na.omit(Data)

for (Team in unique(Data$home_team)){
  for (i in seq(1,length(Data$home_team))){
    if (Team == Data$home_team[i]){

      new_row     = data.frame(Team,
                               "H",
                               as.character((Data$sg_match_ft[i] > 0)*1),
                               Data$away_team[i],
                               #Statistik tim
                               Data$possession_avg_home[i],
                               Data$shots_on_target_avg_home[i],
                               Data$tackles_avg_home[i],
```

```
                                Data$fouls_conceded_avg_home[i],
                                #Statistik Lawan
                                Data$possession_avg_away[i],
                                Data$shots_on_target_avg_away[i],
                                Data$tackles_avg_away[i],
                                Data$fouls_conceded_avg_away[i],
                                stringsAsFactors = T)

      names(new_row) = c("Team", "H_or_A", "Win", "Opponent",
                         'TPosession', 'Tshoot', 'TTackle','TFouls',
                         'OPosession', 'Oshoot', 'OTackle','OFouls')
      df = rbind(df, new_row)

    } else if (Team == Data$away_team[i]){

      new_row     = data.frame(Team,
                                "A",
                                as.character((Data$sg_match_ft[i] < 0)*1),
                                Data$home_team[i],
                                #Statistik tim
                                Data$possession_avg_away[i],
                                Data$shots_on_target_avg_away[i],
                                Data$tackles_avg_away[i],
                                Data$fouls_conceded_avg_away[i],
                                #Statistik tim
                                Data$possession_avg_home[i],
                                Data$shots_on_target_avg_home[i],
                                Data$tackles_avg_home[i],
                                Data$fouls_conceded_avg_home[i],
                                stringsAsFactors = T)
      names(new_row) = c("Team", "H_or_A", "Win", "Opponent",
                         'TPosession', 'Tshoot', 'TTackle','TFouls',
                         'OPosession', 'Oshoot', 'OTackle','OFouls')

      df = rbind(df, new_row)
    }
  }
}

df = na.omit(df)

#Masukkan exposure kedalam dataset
y   = as.data.frame(table(df$Team))

for (i in seq(1,nrow(df))){
  df$Exposure[i] = y$Freq[y$Var1 == df$Team[i]]
}

df$Exposure = as.numeric(df$Exposure)
```

```
head(df)
```

```
##        Team H_or_A Win             Opponent TPosession Tshoot TTackle TFou
ls
## 1 Blackpool      H   1 West Bromwich Albion       48.3    4.4    18.8   11
.3
## 2 Blackpool      H   0              Chelsea       49.8    4.2    19.9   11
.6
## 3 Blackpool      A   0      Manchester City       50.5    4.2    19.0   11
.1
## 4 Blackpool      A   1           Stoke City       50.3    4.3    18.7   11
.5
## 5 Blackpool      A   1     Newcastle United       48.1    4.0    21.0   10
.0
## 6 Blackpool      H   0          Aston Villa       50.5    4.6    19.4   11
.5
##   OPosession Oshoot OTackle OFouls Exposure
## 1       46.9    4.1    20.0   10.8       35
## 2       58.1    6.2    19.7   11.4       35
## 3       54.0    4.8    23.2   13.5       35
## 4       39.2    4.1    17.1   11.4       35
## 5       46.5    6.3    17.7   14.3       35
## 6       47.6    4.4    22.0   13.0       35
```

Sebelum dianalisis, struktur dataset perlu dicek kembali:

```
str(df)
```

```
## 'data.frame':    7682 obs. of  13 variables:
##  $ Team      : Factor w/ 37 levels "Blackpool","Liverpool",..: 1 1 1 1 1 1
1 1 1 1 ...
##  $ H_or_A    : Factor w/ 2 levels "H","A": 1 1 2 2 2 1 2 1 2 1 ...
##  $ Win       : Factor w/ 2 levels "1","0": 1 2 2 1 1 2 2 1 2 2 ...
##  $ Opponent  : Factor w/ 37 levels "AFC Bournemouth",..: 34 11 20 29 23 3
35 7 34 36 ...
##  $ TPosession: num  48.3 49.8 50.5 50.3 48.1 50.5 50.7 49.9 50.6 49.7 ...
##  $ Tshoot    : num  4.4 4.2 4.2 4.3 4 4.6 4.3 4.2 4.6 4.2 ...
##  $ TTackle   : num  18.8 19.9 19 18.7 21 19.4 18.8 20.6 19.5 20.2 ...
##  $ TFouls    : num  11.3 11.6 11.1 11.5 10 11.5 11.8 11.6 11 11.3 ...
##  $ OPosession: num  46.9 58.1 54 39.2 46.5 47.6 44.8 46.5 49.1 49.7 ...
##  $ Oshoot    : num  4.1 6.2 4.8 4.1 6.3 4.4 4.2 4.6 4 4.2 ...
##  $ OTackle   : num  20 19.7 23.2 17.1 17.7 22 21.9 24.9 19.8 23.9 ...
##  $ OFouls    : num  10.8 11.4 13.5 11.4 14.3 13 14.2 13.5 11.5 13.5 ...
##  $ Exposure  : num  35 35 35 35 35 35 35 35 35 35 ...
```

Dari keluaran R diatas, tampak semua kolom sudah memiliki struktur yang tepat. Selanjutnya cek *summary* data tersebut

```
summary(df)
```

```
##                 Team      H_or_A   Win                  Opponent
##  Manchester United: 386   H:3841   1:2897   Manchester United: 386
##  Manchester City  : 385   A:3841   0:4785   Arsenal          : 385
##  Arsenal          : 385                     Manchester City  : 385
##  Liverpool        : 384                     Everton          : 384
##  Everton          : 384                     Liverpool        : 384
##  Chelsea          : 383                     Chelsea          : 383
##  (Other)          :5375                     (Other)          :5375
##    TPosession       Tshoot          TTackle          TFouls        OPosess
ion
##  Min.   :26.8   Min.   : 0.000   Min.   : 5.50   Min.   : 4.50   Min.   :2
6.8
##  1st Qu.:44.8   1st Qu.: 3.500   1st Qu.:16.50   1st Qu.:10.20   1st Qu.:4
4.8
##  Median :49.2   Median : 4.200   Median :18.30   Median :11.20   Median :4
9.2
##  Mean   :50.0   Mean   : 4.379   Mean   :18.27   Mean   :11.21   Mean   :5
0.0
##  3rd Qu.:55.6   3rd Qu.: 5.200   3rd Qu.:20.00   3rd Qu.:12.10   3rd Qu.:5
5.6
##  Max.   :73.6   Max.   :11.000   Max.   :31.00   Max.   :19.00   Max.   :7
3.6
##
##      Oshoot          OTackle          OFouls         Exposure
##  Min.   : 0.000   Min.   : 5.50   Min.   : 4.50   Min.   : 24.0
##  1st Qu.: 3.500   1st Qu.:16.50   1st Qu.:10.20   1st Qu.:206.0
##  Median : 4.200   Median :18.30   Median :11.20   Median :312.0
##  Mean   : 4.379   Mean   :18.27   Mean   :11.21   Mean   :284.1
##  3rd Qu.: 5.200   3rd Qu.:20.00   3rd Qu.:12.10   3rd Qu.:384.0
##  Max.   :11.000   Max.   :31.00   Max.   :19.00   Max.   :386.0
##
```

"Team" dan "Opponent" pasti akan memiliki jumlah yang sama karena "Team" dan "Lawan" adalah dua hal yang sama namun posisinya berbeda. Selanjutnya, akan kita ubah level dari variabel-variabel yang berupa data kategorikal agar menghindari level *default* pada level yang "jarang terjadi" pada data.

```
df = df %>% mutate(Team = relevel(Team, ref = "Manchester City"))
df = df %>% mutate(Win = relevel(Win, ref = "0"))
df = df %>% mutate(Opponent = relevel(Opponent, ref = "Manchester City"))

str(df)
```

```
## 'data.frame':    7682 obs. of  13 variables:
##  $ Team      : Factor w/ 37 levels "Manchester City",..: 2 2 2 2 2 2 2 2 2
2 ...
##  $ H_or_A    : Factor w/ 2 levels "H","A": 1 1 2 2 2 1 2 1 2 1 ...
##  $ Win       : Factor w/ 2 levels "0","1": 2 1 1 2 2 1 1 2 1 1 ...
##  $ Opponent  : Factor w/ 37 levels "Manchester City",..: 34 12 1 29 23 4 3
5 8 34 36 ...
```

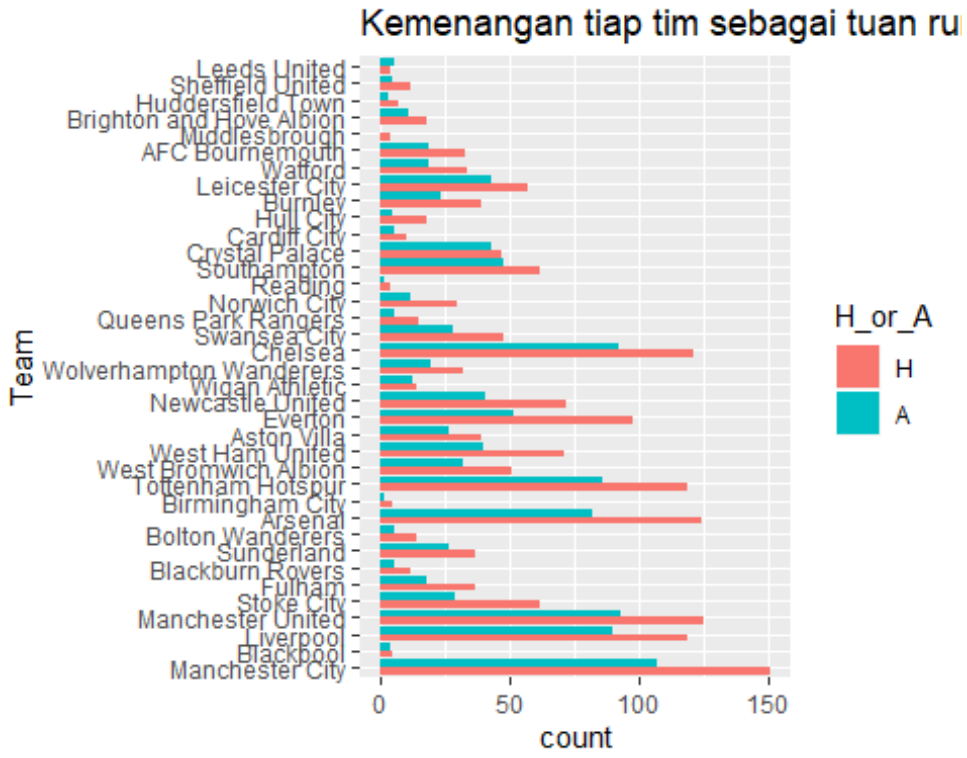
```
##  $ TPosession: num  48.3 49.8 50.5 50.3 48.1 50.5 50.7 49.9 50.6 49.7 ...
##  $ Tshoot    : num  4.4 4.2 4.2 4.3 4 4.6 4.3 4.2 4.6 4.2 ...
##  $ TTackle   : num  18.8 19.9 19 18.7 21 19.4 18.8 20.6 19.5 20.2 ...
##  $ TFouls    : num  11.3 11.6 11.1 11.5 10 11.5 11.8 11.6 11 11.3 ...
##  $ OPosession: num  46.9 58.1 54 39.2 46.5 47.6 44.8 46.5 49.1 49.7 ...
##  $ Oshoot    : num  4.1 6.2 4.8 4.1 6.3 4.4 4.2 4.6 4 4.2 ...
##  $ OTackle   : num  20 19.7 23.2 17.1 17.7 22 21.9 24.9 19.8 23.9 ...
##  $ OFouls    : num  10.8 11.4 13.5 11.4 14.3 13 14.2 13.5 11.5 13.5 ...
##  $ Exposure  : num  35 35 35 35 35 35 35 35 35 35 ...
```

Data tersebut akan lebih jelas dengan bantuan visualiasi data dibawah ini

1. Kemenangan tiap tim sebagai tuan rumah atau tamu

```
temp = df[df$Win == "1",]

ggplot(temp,
       aes(x = Team,
           fill = H_or_A)) +
  geom_bar(position = position_dodge(preserve = "single"))+
  coord_flip()+
  ggtitle("Kemenangan tiap tim sebagai tuan rumah/tamu")
```



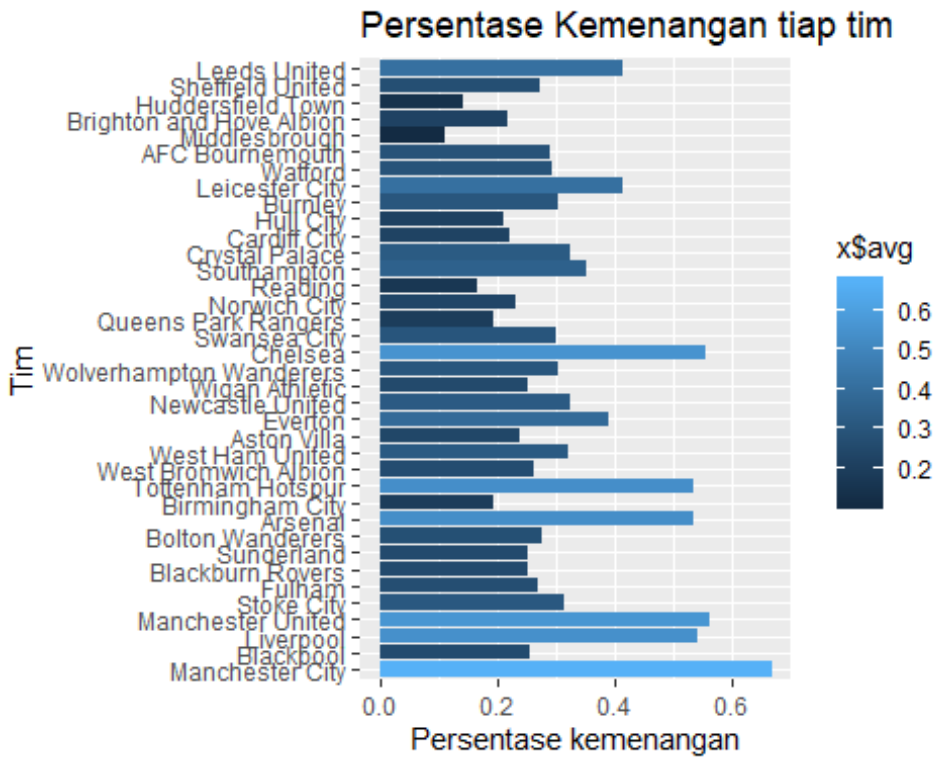
Perhatikan bar warna merah dan biru tiap tim. Bar warna merah mewakili banyakanya kemenangan saat bertanding sebagai tuan rumah sedangkan bar warna biru mewakili kemenangan saat bertanding sebagai tamu. Sekilas terlihat bahwa hampir semua tim memilki kemenangan terbanyak saat menjadi tuan rumah kecuali leeds united dan blackpool

yang cenderung sama secara kasat mata. Oleh karena itu, secara visual, kita perlu memasukkan faktor ini kedalam model.

2. Persentase Kemenangan tiap tim

```
x   = as.data.frame(table(temp$Team))
y   = as.data.frame(table(df$Team))
avg = x$Freq/y$Freq
x   = data.frame(x,y$Freq,avg)

ggplot(x,
       aes(x = x$Var1, y = x$avg, fill=x$avg)) +
  geom_bar(stat = "identity")+
  coord_flip()+
  ggtitle("Persentase Kemenangan tiap tim")+
  labs(y = "Persentase kemenangan", x ="Tim")
```



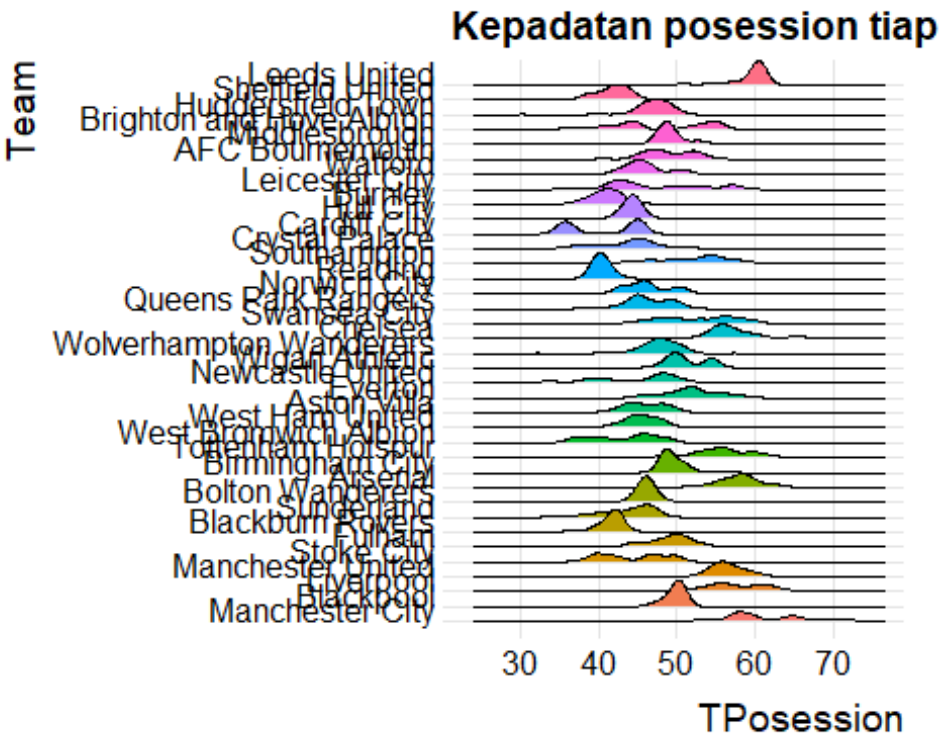
Dari plot diatas, sangat jelas bahwa tiap tim memiliki persentase kemanangan yang berbeda-beda. Ada tim yang sering menang (Contoh: Manchester City) dan ada juga tim yang jarang menang (Contoh: Middlesbrough).Dari sini, sudah sangat jelas bahwa tiap tim memiliki kualitas yang berbeda.

2. Kepadatan statistik lain tiap tim

```
#Plot untuk posession
ggplot(df,
       aes(x = TPosession,
           y = Team,
```

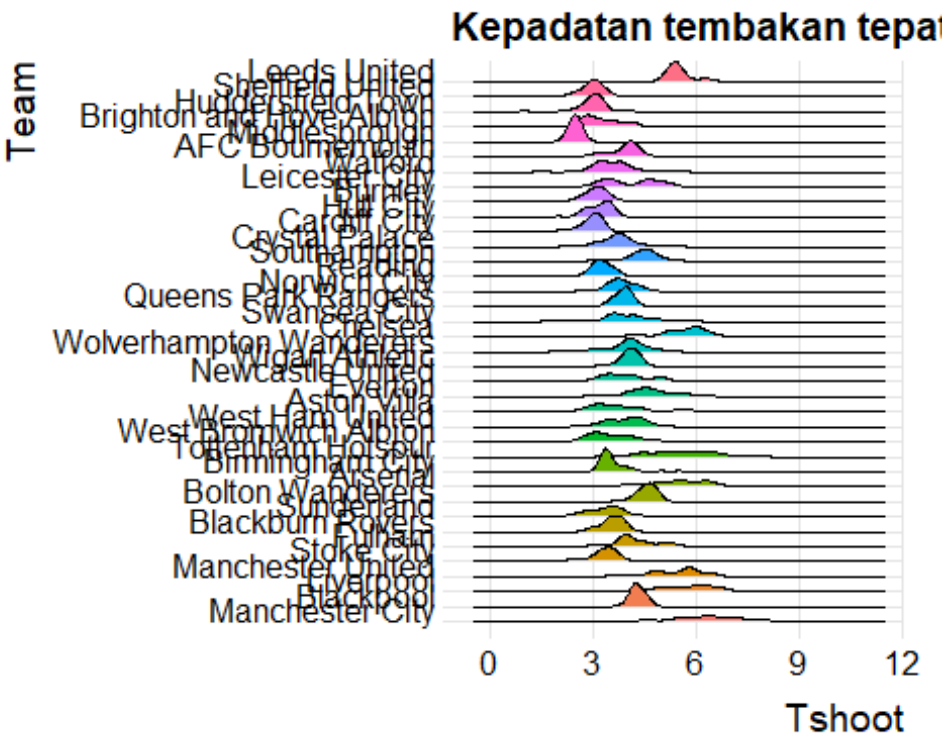
```
           fill = Team)) +
  geom_density_ridges() +
  theme_ridges() +
  ggtitle("Kepadatan posession tiap tim") +
  theme(legend.position = "none")
```

```
## Picking joint bandwidth of 0.933
```



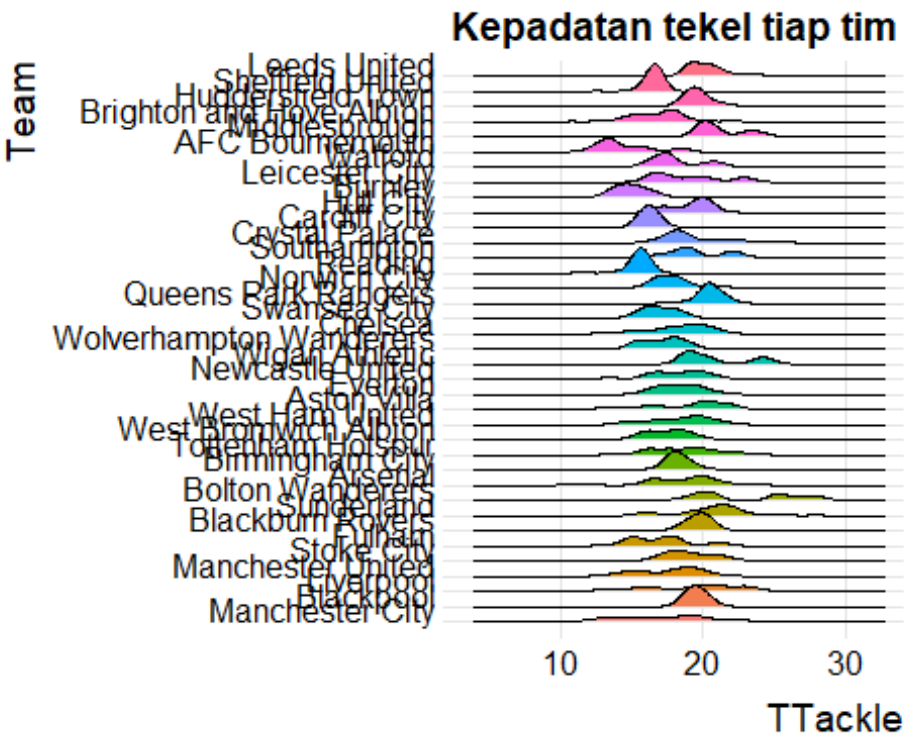

```
#Plot untuk shoot on target
ggplot(df,
       aes(x = Tshoot,
           y = Team,
           fill = Team)) +
  geom_density_ridges() +
  theme_ridges() +
  ggtitle("Kepadatan tembakan tepat sasaran tiap tim") +
  theme(legend.position = "none")
```

```
## Picking joint bandwidth of 0.16
```



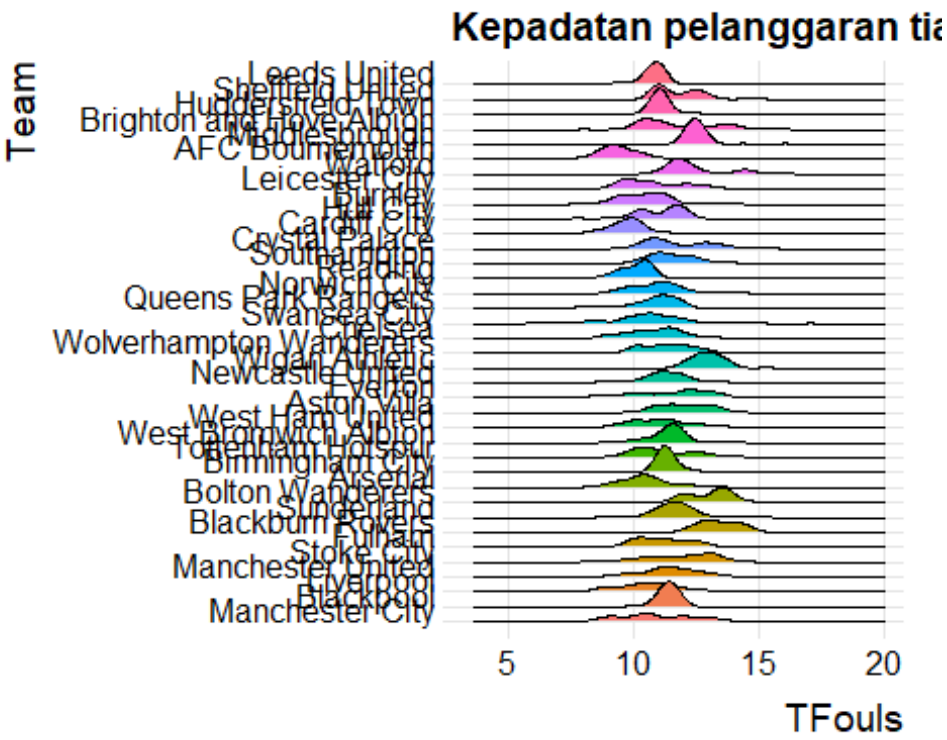

```
#Plot untuk tackel
ggplot(df,
       aes(x = TTackle,
           y = Team,
           fill = Team)) +
  geom_density_ridges() +
  theme_ridges() +
  ggtitle("Kepadatan tekel tiap tim") +
  theme(legend.position = "none")

## Picking joint bandwidth of 0.562
```




```
#Plot untuk fouls
ggplot(df,
       aes(x = TFouls,
           y = Team,
           fill = Team)) +
  geom_density_ridges() +
  theme_ridges() +
  ggtitle("Kepadatan pelanggaran tiap tim") +
  theme(legend.position = "none")

## Picking joint bandwidth of 0.314
```



Dari plot-plot diatas, secara visual, tampak ada tim yang memilki penguasaan bola dan tembakan tepat sasaran yang lebih besar secara signifikan daripada tim-tim lain (contoh: Leeds United, Manchester City, dan Chelsea). Untuk tekel, ada tim yang terbilang memiliki rata-rata tekel lebih rendah secara signifikan daripada tim-tim lain, seperti AFC Bournemouth. Sedangkan untuk pelanggaran, bisa dibilang tiap tim memiliki rata-rata yang sama

Dapat disimpulkan dari semua visualisasi diatas, tiap tim memilki ciri khas dan kualitas masing-masing. Hasil suatu pertandingan sangat mungkin untuk berkorelasi dengan klub yang bermain. Misal, klub yang kuat memiliki kemungkinan kemenangan yang lebih besar. Sedangkan klub yang lemah, memiliki kemungkinan kemenangan yang kecil. Dengan kata lain, setiap klub memiliki kemungkinan kemenangan yang berbeda-beda. Begitu pun juga hasil suatu kemenangan bergantung oleh lawan yang dihadapi. Lawan yang lebih kuat akan lebih sulit untuk dikalahkan, dan begitu juga sebaliknya. Maka dari itu, kita dapat membuat suatu *cluster*, dimana *cluster* tersebut berisikan klub-klub yang bermain. Atas dasar hal tersebut, maka model GLMM (Generelized Linear Mixed Models) adalah model yang tepat untuk menganalisis data ini karena variabel response memiliki korelasi.

Dalam anaslis data ini, Variabel respon dari data yang akan diolah adalah kemenangan suatu tim. Tim yang menang bernilai “1”, sedangkan tim yang kalah atau seri bernilai “0”. Maka dari itu, variabel respon memiliki distribusi bernoulli. *Link function* yang akan dipilih dalam menganalisis data ini adalah *logit link*.

Agar tetap sederhana, akan dicoba memodelkan GLMM dengan *Random intercept model* yang secara umum memiliki bentuk sebagai berikut:

$$ln\left(\frac{\pi}{1-\pi}\right) = \alpha + X'\beta, \qquad y \sim B(1,\pi), \qquad \alpha \sim N(0, v^2)$$

α pada model diatas nilainya berbeda-beda untuk setiap *cluster* (Tim), namun sama untuk setiap *case* di dalam *cluster* (Pertandingan untuk setiap tim). β pada model diatas sudah termasuk $\beta_0, \beta_1, \ldots$.

Metode *Maximum likelihood* digunakan untuk menentukan parameter pada model, namun perhitungannya tidak bisa dalam bentuk tertutup. sehingga perlu integrasi secara numerikal. Pada package R, perhitungan *maximum likelihood* dihitung dengan *laplace estimation*

Analisis Data

Dalam menganalisis data, data dibagi menjadi dua bagian terlebih dahulu. Yang pertama untuk melatih model dan yang kedua untuk menguji model

```
set.seed(20)
n_train  = nrow(df)*0.75

train_df = sample_n(df, n_train)
test_df  = anti_join(df, train_df)

## Joining, by = c("Team", "H_or_A", "Win", "Opponent", "TPosession", "Tshoot
", "TTackle", "TFouls", "OPosession", "Oshoot", "OTackle", "OFouls", "Exposur
e")
```

Pada program R diatas, data yang digunakan sebagai data untuk melatih model sebanyak 75% dari data keseluruhan, sedangkan 25% dari data sisanya dijadikan data untuk menguji prediksi.

Karena pada program R tidak ada stepwise regression untuk model GLMM. Maka akan dicari model terbaik secara manual. Model yang pertama kali dicoba adalah model lengkap yaitu:

Model1 : Win ~ H_or_A + TPosession + Tshoot + TTackle + TFouls + OPosession + Oshoot + OTackle + OFouls + (1 | Team) + (1 | Lawan)

```
set.seed(20)
Model1 <- glmer(as.numeric(paste(Win)) ~
                  H_or_A + TPosession + Tshoot + TTackle + TFouls +
                  OPosession + Oshoot + OTackle + OFouls + (1 | Team) + (1 |
Opponent),
        data = train_df, family = binomial,
        control = glmerControl(),
        start = NULL)
(modelsummary1 = summary(Model1))

## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
```

```
## Formula: as.numeric(paste(Win)) ~ H_or_A + TPosession + Tshoot + TTackle +
##     TFouls + OPosession + Oshoot + OTackle + OFouls + (1 | Team) +
##     (1 | Opponent)
##    Data: train_df
## Control: glmerControl()
##
##      AIC      BIC   logLik deviance df.resid
##   6758.9   6838.8  -3367.4   6734.9     5749
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.1080 -0.7053 -0.4407  0.8733  4.2834
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  Team     (Intercept) 0.08553  0.2925
##  Opponent (Intercept) 0.09580  0.3095
## Number of obs: 5761, groups:  Team, 37; Opponent, 37
##
## Fixed effects:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)  0.010952   0.657114   0.017   0.9867
## H_or_AA     -0.834765   0.060358 -13.830  < 2e-16 ***
## TPosession   0.013086   0.007546   1.734   0.0829 .
## Tshoot       0.290562   0.040823   7.118 1.10e-12 ***
## TTackle      0.019442   0.013498   1.440   0.1498
## TFouls      -0.039336   0.023588  -1.668   0.0954 .
## OPosession  -0.012974   0.007754  -1.673   0.0943 .
## Oshoot      -0.239806   0.041786  -5.739 9.53e-09 ***
## OTackle     -0.028540   0.014082  -2.027   0.0427 *
## OFouls       0.020273   0.023836   0.851   0.3950
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##             (Intr) H_r_AA TPsssn Tshoot TTackl TFouls OPsssn Oshoot OTackl
## H_or_AA     -0.044
## TPosession -0.525 -0.008
## Tshoot      -0.041 -0.041 -0.337
## TTackle     -0.155 -0.007  0.011 -0.020
## TFouls      -0.347  0.021  0.128  0.048 -0.217
## OPosession -0.508  0.003 -0.002  0.014 -0.011  0.001
## Oshoot      -0.020  0.040  0.018 -0.061 -0.023 -0.021 -0.357
## OTackle     -0.137  0.007  0.007 -0.053 -0.282 -0.023 -0.043 -0.064
## OFouls      -0.336 -0.008 -0.010 -0.009 -0.031 -0.114  0.126  0.040 -0.227
## convergence code: 0
## Model failed to converge with max|grad| = 0.00821884 (tol = 0.002, compone
nt 1)
## Model is nearly unidentifiable: very large eigenvalue
##  - Rescale variables?
```

Dari *fitting* model diatas, tampak bahwa “TTackle”, “TFouls” dan “OFouls” tidak signifikan pada model tersebut. Sehingga akan dicoba kembali *fitting* model namun tanpa “TTackle”, “TFouls” dan “OFouls”.

Model2 : Win ~ H_or_A + TPosession + Tshoot + OPosession + Oshoot + OTackle + (1 | Team) + (1 | Lawan)

```
set.seed(20)
Model2 <- glmer(as.numeric(paste(Win)) ~
                  H_or_A + TPosession + Tshoot  +
                  OPosession + Oshoot + OTackle +
                  (1 | Team) + (1 | Opponent),
        data = train_df, family = binomial,
        control = glmerControl(),
        start = NULL)
(modelsummary2 = summary(Model2))
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: as.numeric(paste(Win)) ~ H_or_A + TPosession + Tshoot + OPosessio
n +
##     Oshoot + OTackle + (1 | Team) + (1 | Opponent)
##    Data: train_df
## Control: glmerControl()
##
##      AIC      BIC   logLik deviance df.resid
##   6757.4   6817.4  -3369.7   6739.4     5752
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -3.1151 -0.7081 -0.4406  0.8779  4.4191
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  Team     (Intercept) 0.08200  0.2864
##  Opponent (Intercept) 0.09405  0.3067
## Number of obs: 5761, groups:  Team, 37; Opponent, 37
##
## Fixed effects:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.006328   0.537353  -0.012   0.9906
## H_or_AA     -0.833103   0.060313 -13.813  < 2e-16 ***
## TPosession   0.014262   0.007469   1.910   0.0562 .
## Tshoot       0.294450   0.040808   7.215 5.38e-13 ***
## OPosession  -0.013570   0.007690  -1.765   0.0776 .
## Oshoot      -0.241111   0.041667  -5.787 7.18e-09 ***
## OTackle     -0.022011   0.012968  -1.697   0.0896 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
##
## Correlation of Fixed Effects:
##             (Intr) H_r_AA TPsssn Tshoot OPsssn Oshoot
## H_or_AA     -0.049
## TPosession -0.578 -0.010
## Tshoot      -0.037 -0.042 -0.345
## OPosession -0.567  0.004 -0.002  0.014
## Oshoot      -0.023  0.041  0.021 -0.062 -0.365
## OTackle     -0.411  0.006  0.025 -0.059 -0.017 -0.070
## convergence code: 0
## Model is nearly unidentifiable: very large eigenvalue
##  - Rescale variables?
```

Pada model kedua, terlihat bahwa model kedua memilki AIC yang tidak berbeda jauh dengan Model pertama walaupun parameter yang digunakan lebih sedikit. Walaupun begitu, akan dicoba kembali model tanpa "TPosession", "OPosession" dan "OTackle"

Model3 : Win ~ H_or_A + Tshoot + Oshoot + (1 | Team) + (1 | Lawan)

```
set.seed(20)
Model3 <- glmer(as.numeric(paste(Win)) ~
                  H_or_A + Tshoot + Oshoot +
                  (1 | Team) + (1 | Opponent),
        data = train_df, family = binomial,
        control = glmerControl(),
        start = NULL)
(modelsummary3 = summary(Model3))
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: as.numeric(paste(Win)) ~ H_or_A + Tshoot + Oshoot + (1 | Team) +
##     (1 | Opponent)
##    Data: train_df
## Control: glmerControl()
##
##      AIC      BIC   logLik deviance df.resid
##   6761.1   6801.1  -3374.6   6749.1     5755
##
## Scaled residuals:
##     Min      1Q  Median      3Q     Max
## -2.9956 -0.7089 -0.4415  0.8745  4.4671
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  Team     (Intercept) 0.1074   0.3278
##  Opponent (Intercept) 0.1091   0.3303
## Number of obs: 5761, groups:  Team, 37; Opponent, 37
##
## Fixed effects:
```

```
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept) -0.33993    0.23423  -1.451    0.147
## H_or_AA     -0.83236    0.06031 -13.802  < 2e-16 ***
## Tshoot       0.31651    0.03933   8.047 8.45e-16 ***
## Oshoot      -0.27282    0.03977  -6.860 6.89e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Correlation of Fixed Effects:
##         (Intr) H_r_AA Tshoot
## H_or_AA -0.114
## Tshoot  -0.625 -0.049
## Oshoot  -0.633  0.046 -0.062
```

Ternyata, model ketiga juga memilki AIC yang tidak begitu berbeda jauh dengan model kedua walaupun parameternya lebih sedikit daripada model kedua. Mayoritas parameter di model ketiga juga signifikan di dalam model tersebut dengan.

Sebagai pembanding, akan dilakukan *Fitting* dengan model 2 model, yakni:

1. Model yang persis seperti model ketiga namun menganggap tim sebagai *fixed effect*.
   Model 4: win ~ H_or_A + Tshoot + Oshoot + Team + (1 | Opponent)
2. Model yang persis seperti model ketiga namun tidak ada *random effect*
   Model 5: win ~ H_or_A + Tshoot + Oshoot + Team + Opponent

```
set.seed(20)
Model4 <- glmer(as.numeric(paste(Win)) ~
                  H_or_A + Tshoot + Oshoot + Team + (1 | Opponent),
        data = train_df, family = binomial,
        control = glmerControl(),
        start = NULL)
modelsummary4 = summary(Model4)


Model5 = glm(as.numeric(paste(Win)) ~
                  H_or_A + Tshoot + Oshoot + Team + Opponent,
             family = binomial(link = "logit"),
             data = train_df)
modelsummary5 = summary(Model5)
```

Dengan alasan parameter untuk model keempat dan kelima terlalu banyak, hasil keluaran R untuk model tersebut tidak ditampilkan. Berikut adalah rangkuman hasil AIC dari keenam model diatas adalah:

```
cat("AIC untuk Model 1:", modelsummary1$AICtab[1])

## AIC untuk Model 1: 6758.887

cat("AIC untuk Model 2:", modelsummary2$AICtab[1])

## AIC untuk Model 2: 6757.427
```

```r
cat("AIC untuk Model 3:", modelsummary3$AICtab[1])

## AIC untuk Model 3: 6761.137

cat("AIC untuk Model 4:", modelsummary4$AICtab[1])

## AIC untuk Model 4: 6744.3

cat("AIC untuk Model 5:", modelsummary5$aic)

## AIC untuk Model 5: 6721.751
```

Dari hasil keluaran R diatas, untuk *random intercept model*, didapat bahwa model pertama memiliki AIC terkecil, namun sebenarnya tidak berbeda jauh dengan model *random intercept* lainnya. Jika dibandingkan dengan model tanpa *random effect* (Model kelima), ternyata model kelima memiliki AIC yang terkecil diantara semua model yang ada. Namun jelas bahwa model kelima memilki parameter yang jauh lebih banyak. Lebih lanjut lagi, akan dibuktikan apakah model dengan *random effect* dan model tanpa *random effect* memiliki perbedaan yang signifikan atau tidak.

```r
anova(Model3, Model4, Model5)

## Data: train_df
## Models:
## Model3: as.numeric(paste(Win)) ~ H_or_A + Tshoot + Oshoot + (1 | Team) +
## Model3:     (1 | Opponent)
## Model4: as.numeric(paste(Win)) ~ H_or_A + Tshoot + Oshoot + Team + (1 |
## Model4:     Opponent)
## Model5: as.numeric(paste(Win)) ~ H_or_A + Tshoot + Oshoot + Team + Opponen
t
##        npar    AIC    BIC  logLik deviance  Chisq Df Pr(>Chisq)
## Model3    6 6761.1 6801.1 -3374.6   6749.1
## Model4   41 6744.3 7017.3 -3331.2   6662.3 86.836 35  2.701e-06 ***
## Model5   76 6721.8 7227.8 -3284.9   6569.8 92.550 35  4.293e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Dari hasil keluaran R diatas, baik p-value yang didapat dari perbandingan antara model ketiga dengan keempat dan juga perbandingan model ketiga dan kelima, memiliki p-value yang kurang dari $\alpha = 0.001$ bahkan mendekati nol sehingga bisa dikatakan dengan mengganti tim menjadi *fixed effect* dan juga model tanpa *random effect* akan mengubah model secara signifikan. Dengan kata lain, *random effect* pada model ketiga tidak dapat diabaikan. Untuk mengetahui mana model yang terbaik dalam menghasilkan prediksi, dibawah ini, akan dihasilkan kurva ROC untuk masing-masing model.

```r
pred1 = predict(Model1, type = "response")
roccurve1 = roc(train_df$Win ~ pred1)

pred2 = predict(Model2, type = "response")
roccurve2 = roc(train_df$Win ~ pred2)
```

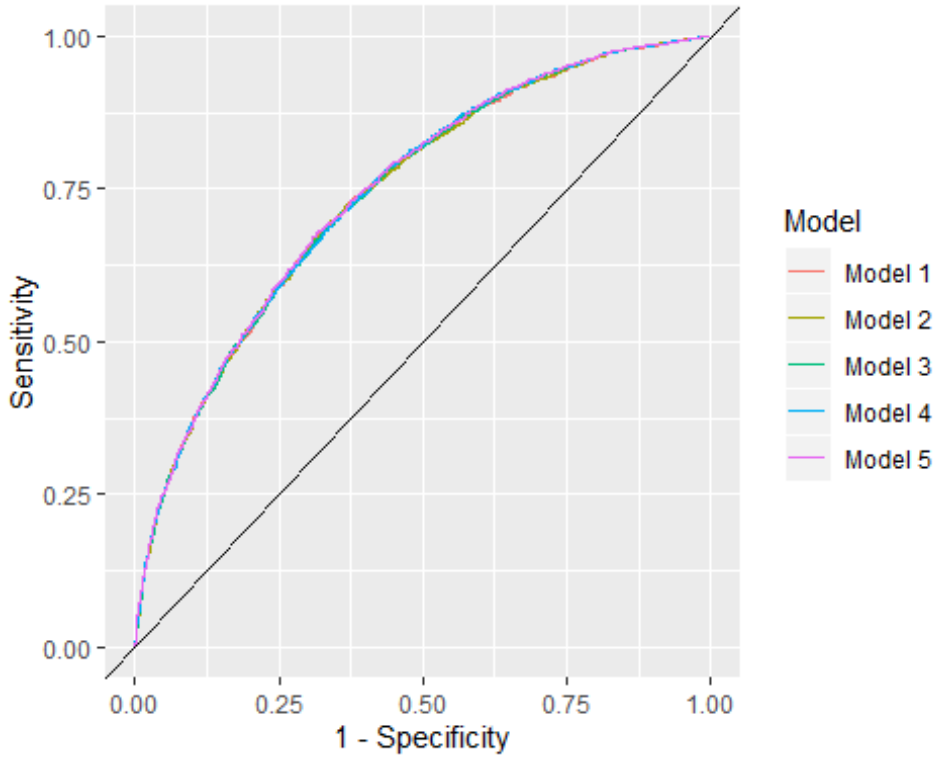
```
pred3 = predict(Model3, type = "response")
roccurve3 = roc(train_df$Win ~ pred3)

pred4 = predict(Model4, type = "response")
roccurve4 = roc(train_df$Win ~ pred4)

pred5 = predict(Model5, type = "response")
roccurve5 = roc(train_df$Win ~ pred5)

roclist = list("Model 1" = roccurve1,
               "Model 2" = roccurve2,
               "Model 3" = roccurve3,
               "Model 4" = roccurve4,
               "Model 5" = roccurve5)

ggroc(roclist, aes = "colour", legacy.axes = T)+
  geom_abline(intercept = 0, slope = 1)+
  labs(x = "1 - Specificity",
       y = "Sensitivity",
       colour = "Model")
```



Secara sekilas, dapat dilihat bahwa ternyata hampir semua kurva ROC dari setiap model memiliki kurva yang sama. Maka dari itu, agar lebih detail akan dihitung luas dibawah kurva ROC untuk masing-masing model.

```
cat("Luas dibawah kurva ROC untuk Model 1 :", auc(roccurve1))
```

```
## Luas dibawah kurva ROC untuk Model 1 : 0.743294
```

```
cat("Luas dibawah kurva ROC untuk Model 2 :", auc(roccurve2))
```

```
## Luas dibawah kurva ROC untuk Model 2 : 0.74252
```

```
cat("Luas dibawah kurva ROC untuk Model 3 :", auc(roccurve3))
```

```
## Luas dibawah kurva ROC untuk Model 3 : 0.7430289
```

```
cat("Luas dibawah kurva ROC untuk Model 4 :", auc(roccurve4))
```

```
## Luas dibawah kurva ROC untuk Model 4 : 0.7445247
```

```
cat("Luas dibawah kurva ROC untuk Model 5 :", auc(roccurve5))
```

```
## Luas dibawah kurva ROC untuk Model 5 : 0.7459111
```

Dari hasil perhitungan luas kurva ROC diatas, terlihat bahwa semua model memiliki perbedaan yang sangat tipis. Namun begitu, karena model ketiga merupakan model yang paling sederhana. Dengan memegang prinsip parsimony, Maka model ketiga adalah model yang diterima. Berikut formula dari model ketiga

```
Model3
```

```
## Generalized linear mixed model fit by maximum likelihood (Laplace
##   Approximation) [glmerMod]
##  Family: binomial  ( logit )
## Formula: as.numeric(paste(Win)) ~ H_or_A + Tshoot + Oshoot + (1 | Team) +
##     (1 | Opponent)
##    Data: train_df
##       AIC       BIC    logLik  deviance  df.resid
##  6761.137  6801.090 -3374.568  6749.137      5755
## Random effects:
##  Groups   Name        Std.Dev.
##  Team     (Intercept) 0.3278
##  Opponent (Intercept) 0.3303
## Number of obs: 5761, groups:  Team, 37; Opponent, 37
## Fixed Effects:
## (Intercept)      H_or_AA       Tshoot       Oshoot
##     -0.3399      -0.8324       0.3165      -0.2728
```

$$ln(\frac{\pi}{1-\pi})$$
$$= -0.3399 - 0.8324x_{HorA} + 0.3165x_{TShoot} - 0.2728x_{OShoot} + \alpha_{Team} + \alpha_{Opponent}$$

dengan,

$$\alpha_{Team} \sim N(0, 0.3278^2)$$

$$\alpha_{Opponent} \sim N(0, 0.3303^2)$$

Contoh interpretasi yang dapat diambil dari hasil fitting diatas adalah setiap kenaikan 1 satuan ekspektasi tembakan tepat sasaran oleh tim maka akan meningkatkan *odds* kemenangan sebesar 37.23% karena $e^{0.3165} = 1.3723$. Sedangkan setiap kenaikan 1 satuan ekspektasi tembakan tepat sasaran oleh lawan maka akan menurunkan *odds* kemenangan sebesar 23.88% karena $e^{-0.2728} = 0.7612$ Jika tim bertanding sebagai tim tamu, maka *odds* kemenangan akan menurun sebesar 56.50% karena $e^{-0.8324} = 0.43500$ dibandingkan jika tim tersebut bermain sebagai tuan rumah. Untuk random intercept, nilai dari random intercept berbeda-beda tiap *cluster* namun sama untuk semua pertandingan dalam *cluster* tersebut. Hal ini berlaku baik *random intercept* tim ataupun lawan

Selanjutnya, akan dibangun tabel kontingensi dari model ketiga

```
pred = factor(ifelse(predict(Model3) < 0.4, 0, 1))
mat  = confusionMatrix(pred, as.factor(train_df$Win))
mat
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction    0    1
##          0 3331 1554
##          1  237  639
##
##                Accuracy : 0.6891
##                  95% CI : (0.677, 0.7011)
##     No Information Rate : 0.6193
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.2544
##
##  Mcnemar's Test P-Value : < 2.2e-16
##
##             Sensitivity : 0.9336
##             Specificity : 0.2914
##          Pos Pred Value : 0.6819
##          Neg Pred Value : 0.7295
##              Prevalence : 0.6193
##          Detection Rate : 0.5782
##    Detection Prevalence : 0.8479
##       Balanced Accuracy : 0.6125
##
##        'Positive' Class : 0
##
```

Dari tabel kontingensi diatas, ada beberapa hal dapat diambil, yaitu:

1. Tingkat akurasi model terhadap *train data* adalah 68.91% (Accuracy)
2. 93.36% Pertandingan yang berakhir seri atau kalah dapat diprediksi dengan benar (Sensitivity)

3. 29.14% Pertandingan yang berakhir kemenangana dapat diprediksi dengan benar (Specificity)
4. 68.19% Hasil prediksi kemenangan sesuai dengan data sesungguhnya
5. 72.95% Hasil prediksi kekalahan atau seri sesuai dengan data sesungguhnya
6. 61.93% data yang ada adalah pertandingan yang berakhir seri atau kalah(Prevalance)
7. 57.82% data yang ada dideteksi sebagai seri atau kekalahan (Detection Rate)
8. 84.79% data yang ada dideteksi sebagai seri atau kekalahan baik prediksinya salah ataupun benar (Detection Prevelance)
9. Rata-rata kebenaran model dalam meprediksi adalah 61.25%

Dari hasil prediksi diatas terhadap *train data*, didapat bahwa hasil akurasi masih kurang baik. Berdasakan *detection prevelance&*, terlalu banyak prediksi yang menyatakan bahwa hasil akhir seri atau kalah. Sehingga, terbukti mengapa *sensitivity*-nya jauh lebih besar daripada *specificity*-nya

Prediksi

Cuplikan hasil prediksi model dengan data test adalah sebagai berikut

```
prob=round(predict(Model3, test_df, type='response'),2)

predtest=factor(ifelse(prob < 0.4 , 0, 1))

print(sample_n(data.frame(prob, FittedValue=predtest, test_df), 10))
```

```
##      prob FittedValue                 Team H_or_A Win                 Oppo
nent
## 166  0.56           1    Manchester United      A   1             Swansea
City
## 1215 0.24           0              Chelsea      A   0          Manchester
City
## 1899 0.34           0    Huddersfield Town      H   0          AFC Bournem
outh
## 1912 0.19           0     Sheffield United      A   0          West Ham Un
ited
## 1666 0.29           0              Burnley      A   0 Brighton and Hove Al
bion
## 761  0.26           0 West Bromwich Albion      H   0                   Eve
rton
## 1890 0.06           0    Huddersfield Town      A   0                   Ars
enal
## 541  0.65           1              Arsenal      A   1           Crystal Pa
lace
## 990  0.42           1              Everton      H   0                   Che
lsea
## 1726 0.13           0       Leicester City      A   0                Liver
pool
##      TPosession Tshoot TTackle TFouls OPosession Oshoot OTackle OFouls Exp
```

```
osure
## 166        55.6    5.9    18.5   13.3       49.0    4.5    18.0   12.2
386
## 1215       55.7    5.5    18.4   11.3       58.6    6.1    18.7   12.6
383
## 1899       48.8    3.2    18.9   10.8       45.6    4.1    13.1    8.7
70
## 1912       41.3    3.0    16.6   12.4       42.0    4.3    13.9   10.5
62
## 1666       42.5    3.1    14.6    8.9       43.5    2.8    15.1   10.5
206
## 761        46.5    3.6    17.4   11.5       55.8    5.7    19.3   10.0
314
## 1890       45.7    2.6    20.4   11.2       60.2    6.1    15.9   11.2
70
## 541        54.3    6.6    18.6   11.0       43.7    2.8    24.6    9.6
385
## 990        52.4    5.9    17.4   12.7       55.2    5.6    19.2   11.3
384
## 1726       44.7    3.6    16.4    9.7       59.8    6.6    16.9    9.6
240
```

```
confusionMatrix(predtest, as.factor(test_df$Win))
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   0   1
##          0 830 239
##          1 387 465
##
##                Accuracy : 0.6741
##                  95% CI : (0.6527, 0.6951)
##     No Information Rate : 0.6335
##     P-Value [Acc > NIR] : 0.0001084
##
##                   Kappa : 0.328
##
##  Mcnemar's Test P-Value : 4.221e-09
##
##             Sensitivity : 0.6820
##             Specificity : 0.6605
##          Pos Pred Value : 0.7764
##          Neg Pred Value : 0.5458
##              Prevalence : 0.6335
##          Detection Rate : 0.4321
##    Detection Prevalence : 0.5565
##       Balanced Accuracy : 0.6713
##
```

```
##        'Positive' Class : 0
##
```

Dari tabel kontingensi diatas, ada beberapa hal dapat diambil, yaitu:

1. Tingkat akurasi model terhadap *test data* adalah 67.41% (Accuracy)
2. 68.20% Pertandingan yang berakhir seri atau kalah dapat diprediksi dengan benar (Sensitivity)
3. 66.05% Pertandingan yang berakhir kemenangan dapat diprediksi dengan benar (Specificity)
4. 77.64% Hasil prediksi kemenangan sesuai dengan data sesungguhnya (PPV)
5. 54.58% Hasil prediksi kekalahan atau seri sesuai dengan data sesungguhnya (NPV)
6. 63.35% data yang ada adalah pertandingan yang berakhir seri atau kalah(Prevalance)
7. 43.21% data yang ada dideteksi sebagai seri atau kekalahan (Detection Rate)
8. 55.65% data yang ada dideteksi sebagai seri atau kekalahan baik prediksinya salah ataupun benar (Detection Prevelance)
9. Rata-rata kebenaran model dalam meprediksi adalah 67.13%

Dari hasil prediksi diatas, didapat bahwa hasil akurasi masih kurang baik. Namun *sensitivity* dan *specificity*-nya cukup berimbang. Dari sini, dapt diketahui bahwa model yang dibangun masih kurang baik.

Ada beberapa kemungkinan yang dapat memengaruhi hasil model tersebut, yaitu:

1. Informasi yang diambil dari data kurang menjelaskan pertandingan yang akan diprediksi
2. Variabel penjelas yang perlu dimasukkan kedalam model tidak sesuai
3. Sepakbola terlalu kompleks untuk diprediksi dengan GLM, sehingga perlu metode yang lebih *advance* seperti *deep learning*

Kesimpulan

1. Model terbaik yang dapat memprediksi kemenangan tim *English Premiere League* adalah $ln(\frac{\pi}{1-\pi}) = -0.3399 - 0.8324x_{HorA} + 0.3165x_{TShoot} - 0.2728x_{OShoot} + \alpha_{Team} + \alpha_{Opponent}$
2. Tingkat akurasi dari model yang didapat adalah:
   - Akurasi dengan *train data* adalah 68.91%
   - Akurasi dengan *test data* adalah 67.41%
3. Model yang didapat masih kurang baik dalam memprediksi kemenangan suatu tim.
4. Beberapa kemungkinan yang dapat mempengaruhi hasil model, diantaranya informasi yang kurang menjelaskan pertandingan, variabel penjelas yang tidak sesuai, dan model yang kurang tepat

Referensi

De jong, piet, and Gillian Z. Heller. 2008. *Generalized linear models for insurance data*. Cambridge: Cambridge University Press.

Walpole et all, 2011, *Probability And Statistics For Engineers And Scientists*, ninth edition, Prentice Hall: Boston.