G B

ology@github

Search this website .

Inspecting the English Premier League Player Stats with R

2016-10-01 BY GENE

Being a soccer person and programmer, I wanted to inspect player statistics for myself. I finally found this excellent site for many leagues and primarily with player stats: whoscored.com. So, seeing that there was no download link, I determined to tediously copy/paste all the records for each player, for defensive, offensive, passing and summary categories, for last season, into four files (epl-player-stats-defensive-2015-16, epl-player-stats-offensive-2015-16, epl-player-stats-passing-2015-16 and epl-player-stats-summary-2015-16). Oof! All along the way, I thought about making little a web-page scraper… But in the end, I had my raw data. Here is the head of defense:

```
R
Name
Player     Apps      Mins      Tackles      Inter      Fouls      Offsides      Cl
1
Riyad Mahrez
Leicester, 25, AM(CLR)  36(1)   3058     1.4      1   0.5      -   0.4     1.9
2
Dimitri Payet
West Ham, 29, M(CLR)    29(1)   2573     0.8      0.8     0.4      -   0.1
```

```
3
Alexis Sánchez
Arsenal, 27, M(LR),FW   28(2)   2446    1.6     0.8     1.1     -   0.2
...
```

So I pressed on with the next task: Turning this into something that R can read without any pain.

Enter perl:

```
#!/usr/bin/env perl
# Program: player-stats.pl
use strict;
use warnings;

# Given a file to read
my $input = shift or die "Usage: perl $0 filename.txt";
# Open the file
open( my $in, '<', $input ) or die "Can't read < $input: $!";

# Set the maximum number of lines to read per row
my $max = 2;
# Set the initial line number
my $i = 1;
# Fill-up rows file lines per data point
my @row;

# Inspect each line...
while ( my $line = <$in> ) {
    # Strip leading or trailing whitespace
    $line =~ s/\A\s+//;
    $line =~ s/\s+\z//;

    # Split a line into fields (separated by more than one space)
    my @fields = split /\s{2,}/, $line;

    # Save the fields into the row
    push @row, @fields;

    # If we've seen max lines...
    if ( $i > $max ) {
        # Print out the row
        print join( "\t", @row ), "\n";
        # Reset the line counter and row
        $i = 1;
        @row = ();
    }
    else {
        # Increment our line read
        $i++;
    }
}

close $in;
```

Nothing tricky at all. Just read-in the file – three lines per row – and tab-separate the fields. With that in place, I say this on the command-line:

```
$ perl player-stats.pl EPL-Player-Stats-Defensive-2015-16.txt > Defensive-2
$ perl player-stats.pl EPL-Player-Stats-Offensive-2015-16.txt > Offensive-2
$ perl player-stats.pl EPL-Player-Stats-Passing-2015-16.txt > Passing-2015-
$ perl player-stats.pl EPL-Player-Stats-Summary-2015-16.txt > Summary-2015-
```

With those R-friendly processed files, I can now open R, import and explore the data. First, the importing:

```
# Set the R display width to the width of the screen
options( width = as.integer( Sys.getenv("COLUMNS") ) )

player = read.table("Defensive-2015-16-processed.txt", header = TRUE, na.st
offense = read.table("Offensive-2015-16-processed.txt", header = TRUE, na.s
passing = read.table("Passing-2015-16-processed.txt", header = TRUE, na.str
summary = read.table("Summary-2015-16-processed.txt", header = TRUE, na.str

player$Goals   = offense$Goals
player$Assists = offense$Assists
player$SpG     = offense$SpG
player$KeyP    = offense$KeyP
player$Drb     = offense$Drb
player$Fouled  = offense$Fouled
player$Off     = offense$Off
player$Disp    = offense$Disp
player$UnsTch  = offense$UnsTch

player$AvgP    = passing$AvgP
```

```
player$Pass    = passing$"PS."
player$Crosses = passing$Crosses
player$LongB   = passing$LongB
player$ThrB    = passing$ThrB

player$Yel         = summary$Yel
player$Red         = summary$Red
player$AerialsWon = summary$AerialsWon
player$MotM        = summary$MotM

rm(offense)
rm(passing)
rm(summary)

player$Club = factor( gsub( '(.*), \\d+, .*$', '\\1', player$Player ) )
player$Age = as.integer( gsub( '^.*, (\\d+), .*$', '\\1', player$Player ) )
player$Posn = factor( gsub( '^.*, \\d+, (.*)$', '\\1', player$Player ) )
player$Field = factor( gsub( '\\(.*?\\)', '', player$Posn ) )
player$Starts = as.integer( gsub( '\\(.*?\\)', '', player$Apps ) )
player$Subs = ifelse( grepl( '\\d+\\(\\d+?\\)', player$Apps ), as.integer(
player$AllApps = player$Starts + ifelse( is.na( player$Subs ), 0, player$Su

attach(player)
```

Next, the exploring:

```
R> subset(player, Goals > 15, select = c(Name, Goals))
             Name Goals
1    Riyad Mahrez    17
8      Harry Kane    25
9     Jamie Vardy    24
15  Sergio Agüero    24
49  Romelu Lukaku    18
74 Olivier Giroud    16
```

With SQL statements:

```
library(sqldf)

sqldf('select Age, count(*) as Number from player group by Age')
   Age Number
1   19      1
```

```
2   20      3
3   21      5
4   22     11
5   23     15
6   24     18
7   25     27
8   26     31
9   27     38
10  28     29
11  29     31
12  30     29
13  31     20
14  32     15
15  33      8
16  34      7
17  35      7
18  36      2
19  37      1
```

Better than average Forwards:

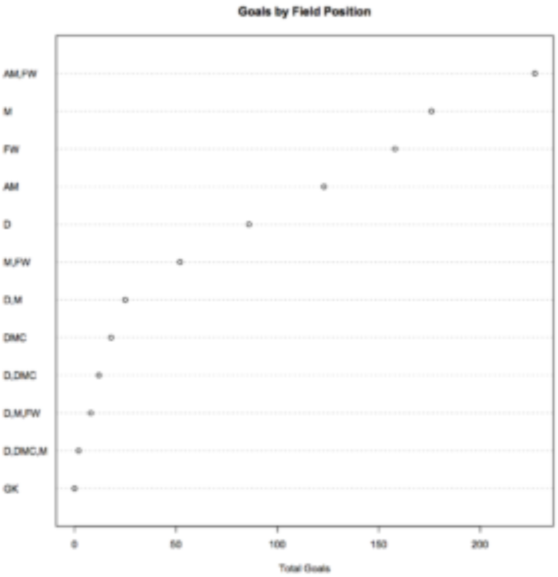
```
sqldf('select Name, Mins, Goals, SpG, Assists, Crosses, KeyP, AvgP, Pass fr
             Name Mins Goals SpG Assists Crosses KeyP AvgP Pass
1 Kevin De Bruyne 2004     7 2.0       9     1.9  3.2 45.1 78.3
2   Cesc Fàbregas 2899     5 1.2       7     0.6  1.8 76.5 84.0
```

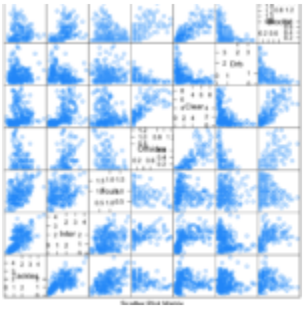
Goals by Field position:

```
sqldf('select Field, sum(Goals) as Total, count(*) as Number, cast(sum(Goal
     Field Total Number      Per
1    AM,FW   227     28 8.107143
2       FW   158     20 7.900000
3     M,FW    52      7 7.428571
4       AM   123     28 4.392857
5   D,M,FW     8      2 4.000000
6        M   176     68 2.588235
7    D,DMC    12      7 1.714286
8        D    86     75 1.146667
9      D,M    25     23 1.086957
10 D,DMC,M     2      2 1.000000
11     DMC    18     18 1.000000
12      GK    NA     20       NA
```

And here is a nice way of seeing goals by field position:

```
agg <- aggregate( player$Goals, by=list(Field=player$Field), FUN=sum )
ordered <- agg[order(agg$x),]
dotchart( ordered$x, labels=ordered$Field, cex=.7, main="Goals by Field Pos
```

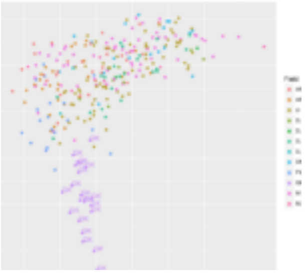




Ok. Let’s try to spot any strong relationships:

```
library(lattice)
splom(player[c(6,7,8,9,10,11,12)])
```

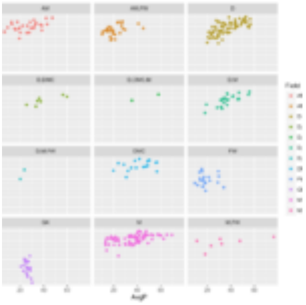
Hmm. Tackles x Interceptions looks like a sort-of linear relationship. So does Clearances x Blocks.



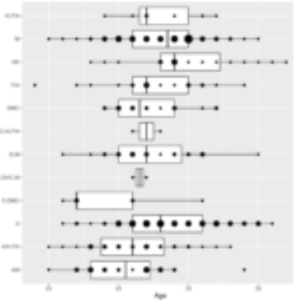
Ok. Time for some visualization. Here is average passes per game (AvgP) by pass completion percentage (Pass):

```
library(ggplot2)
```

```
ggplot( player, aes( AvgP, Pass, color = Field ) ) +
    geom_point() +
geom_text( aes( label = ifelse( Field == 'GK', 'GK', '' ) ), hjust = 0,
```

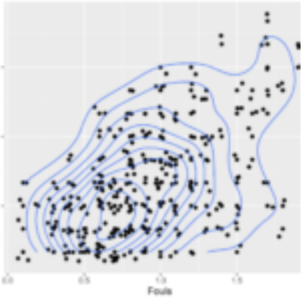


```
ggplot( player, aes( AvgP, Pass, color = Field ) )
    + geom_point() + facet_wrap( ~ Field, nrow = 4 )
```



What about ages by field position?

```
ggplot( data = player, mapping = aes( x = Field, y =
    + geom_boxplot()
    + geom_point()
    + geom_count()
    + coord_flip()
```



There should be a relationship between yellow cards and fouls:

```
ggplot( player, aes( Fouls, Yel ) ) + geom_point() +
```

We could go on and on, slicing, dicing and visualizing, but these are the tools that I reach for initially, to explore data.

FILED UNDER: DATA, SOFTWARE
TAGGED WITH: PERL, R, SOCCER, VISUALIZATION

Epistemologist-at-large

^ Top