Cricket Analytics with cricketr

Tinniam V Ganesh

Cricket analytics with cricketr

Tinniam V Ganesh

tvganesh.85@gmail.com

DEDICATION

This book is dedicated to all the cricketing fans of the world.

CONTENTS

PREFACE

Cricket has been the "national passion" of India for decades. As a boy I was also held in thrall by a strong cricketing passion like many. Cricket is a truly fascinating game! I would catch the sporting action with my friends as we crowded around a transistor that brought us live, breathless radio commentary. We also spent many hours glued to live cricket action on the early black and white TVs. This used to be an experience of sorts, as every now and then a part of the body of the players, would detach itself and stretch to the sides. But it was enjoyable all the same.

Nowadays broadcast technology has improved so much and we get detailed visual analysis of the how each bowler varies the swing and length of the delivery. We are also able to see the strokes of batsman in slow motion. Similarly computing technology has also advanced by leaps and bounds and we can analyze players in great detail with a few lines of code in languages like R, Python etc.

In 2015, I completed Machine Learning from Stanford at Coursera. I was looking around for data to play around with, when it suddenly struck me that I could do some regression analysis of batting records. In the subsequent months, I took the Data Science Specialization from John Hopkins University, which triggered more ideas in me. One thing led to another and I managed to put together an R package called 'cricketr'. I developed this package over 7 months adding and refining functions. Finally, I managed to submit the package to CRAN. During the development of the package for different formats of the game I wrote a series of posts in my blog.

This book is a collection of those cricket related posts. There are 6 posts based on my R package 'cricketr'. I have also included 2 earlier posts based on R which I wrote before I created my R package. Finally, I also include another 2 cricket posts based on Machine Learning in which I used the language Octave.

My ‘cricketr’ package is a first, for cricket analytics, howzzat! and I am certain that it won’t be the last. Cricket is often referred to, with a much clichéd phrase, as the game of ‘glorious uncertainties’. I hope my R package ‘cricketr’ brings a degree of certainty to the game of cricket.

Cricket is a wonderful pitch for statisticians, data scientists and machine learning experts. So you can expect some cool packages in the years to come.

I had a great time developing the package. I hope you have a wonderful time reading this book.

Feel free to get in touch with me anytime through email included below

Tinniam V Ganesh

tvganesh.85@gmail.com

February 3, 2016

Cricket Analytics with cricketr

1.1. Introducing cricketr! : An R package to analyze performances of cricketers

Yet all experience is an arch wherethro'

Gleams that untravell'd world whose margin fades

For ever and forever when I move.

How dull it is to pause, to make an end,

To rust unburnish'd, not to shine in use!

Ulysses by Alfred Tennyson

1.1.1. Introduction

This is an initial post in which I introduce a cricketing package 'cricketr' which I have created. This package was a natural culmination to my earlier posts on cricket and my finishing 10 modules of Data Science Specialization, from John Hopkins University at Coursera. The thought of creating this package struck me some time back, and I have finally been able to bring this to fruition.

So here it is. My R package 'cricketr'!!!'

This package uses the statistics info available in ESPN Cricinfo Statsguru. The current version of this package can handle all formats of the game including Test, ODI and Twenty20 cricket.

You should be able to install the package from GitHub (https://github.com/tvganesh/cricketr) and use many of the functions available in the package. Please be mindful of ESPN Cricinfo Terms of Use (http://www.espncricinfo.com/ci/content/site/company/terms_use.html)

The cricketr package

The cricketr package has several functions that perform several different analyses on both batsman and bowlers. The package has functions that plot percentage frequency runs or wickets, runs likelihood for a batsman, relative run/strike rates of batsman and relative performance/economy rate for bowlers are available.

Other interesting functions include batting performance moving average, forecast and a function to check whether the batsman/bowler is in in-form or out-of-form.

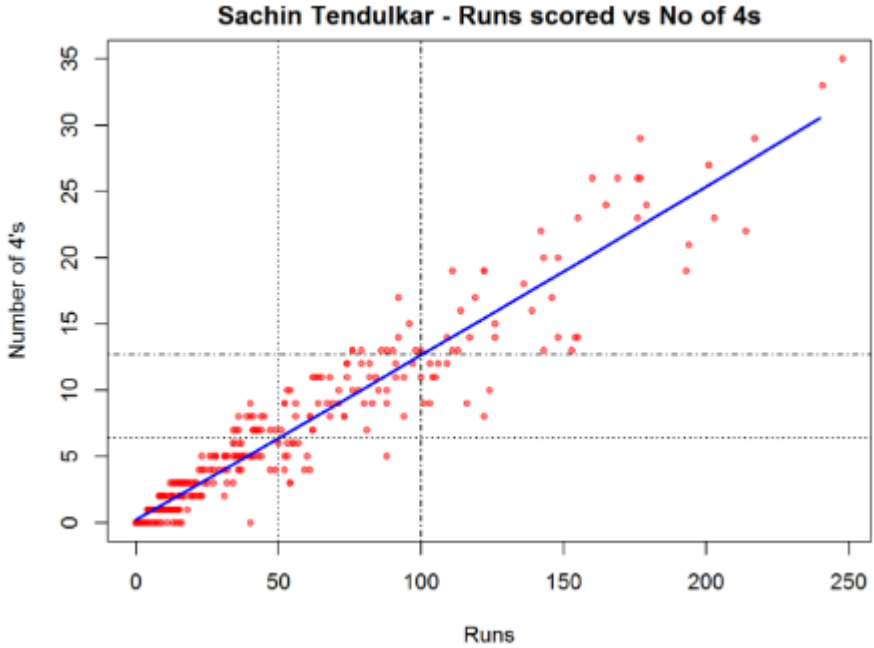
The data for a particular player can be obtained with the getPlayerData() function from the package. To do this you will need to go to ESPN CricInfo Player and type in the name of the player for e.g Ricky Ponting, Sachin Tendulkar etc. This will bring up a page which have the profile number for the player e.g. for Sachin Tendulkar this would be http://www.espncricinfo.com/india/content/player/35320.html. Hence, Sachin's profile is 35320. This can be used to get the data for Tendulkar as shown below

The cricketr package is now available from CRAN!!!. You should be able to install directly with

```
if (!require("cricketr")){
   install.packages("cricketr",lib = "c:/test")
}
library(cricketr)
```

The cricketr package includes some pre-packaged sample (.csv) files. You can use these sample to test functions as shown below

```
# Retrieve the file path of a data file installed with cricketr
pathToFile <- system.file("data", "tendulkar.csv", package = "cricketr")
batsman4s(pathToFile, "Sachin Tendulkar")

# The general format is pkg-function(pathToFile,par1,...)
batsman4s(<path-To-File>,"Sachin Tendulkar")
```



Alternatively, the cricketr package can be installed from GitHub with

```
if (!require("cricketr")){
   library(devtools)
   install_github("tvganesh/cricketr")
}
library(cricketr)
```

The pre-packaged files can be accessed as shown above.

To get the data of any player use the function getPlayerData()

```
tendulkar  <-
getPlayerData(35320,dir="..",file="tendulkar.csv",type="batting",homeOrAway=c(1,
2),
                                        result=c(1,2,4))
```

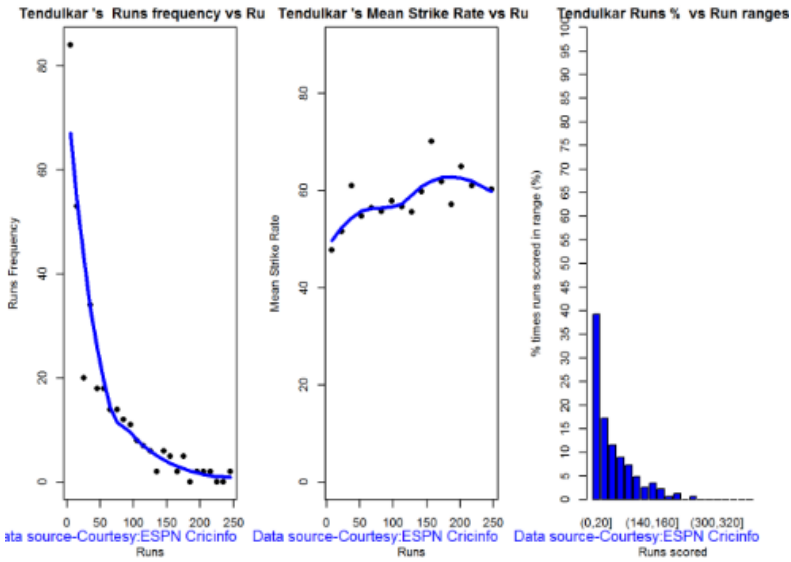
Important Note This needs to be done only once for a player. This function stores the player's data in a CSV file (for e.g. tendulkar.csv as above) which can then be reused for all other functions. Once we have the data for the players many analyses can be done. This post will use the stored CSV file obtained with a prior getPlayerData() for all subsequent analyses

1.1.2. Sachin Tendulkar's performance – Basic Analyses

The 3 plots below provide the following for Tendulkar

1. Frequency percentage of runs in each run range over the whole career
2. Mean Strike Rate for runs scored in the given range
3. A histogram of runs frequency percentages in runs ranges

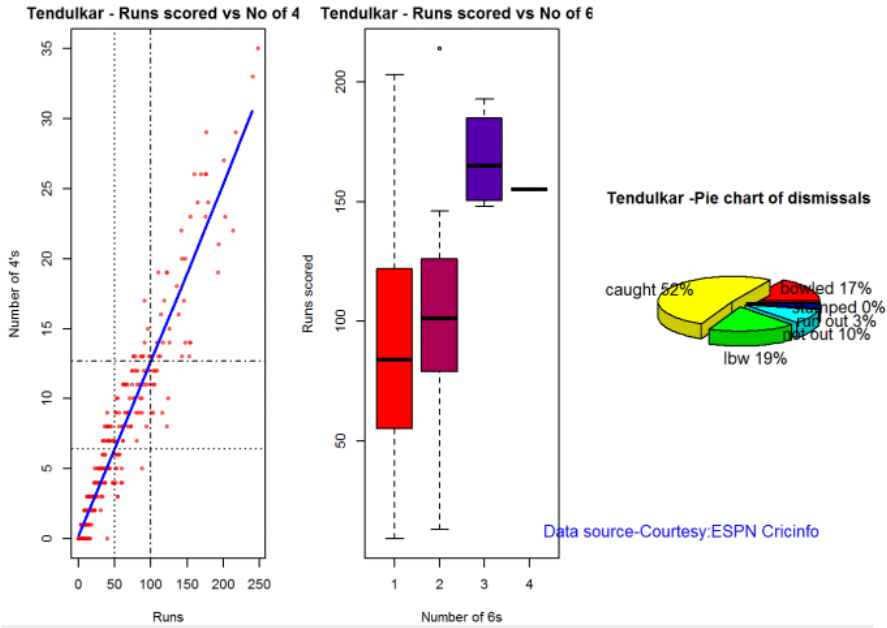
```
par(mfrow=c(1,3))
par(mar=c(4,4,2,2))
batsmanRunsFreqPerf("./tendulkar.csv","Sachin Tendulkar")
batsmanMeanStrikeRate("./tendulkar.csv","Sachin Tendulkar")

batsmanRunsRanges("./tendulkar.csv","Sachin Tendulkar")
```



1.1.3. More analyses

```
par(mfrow=c(1,3))
par(mar=c(4,4,2,2))
```

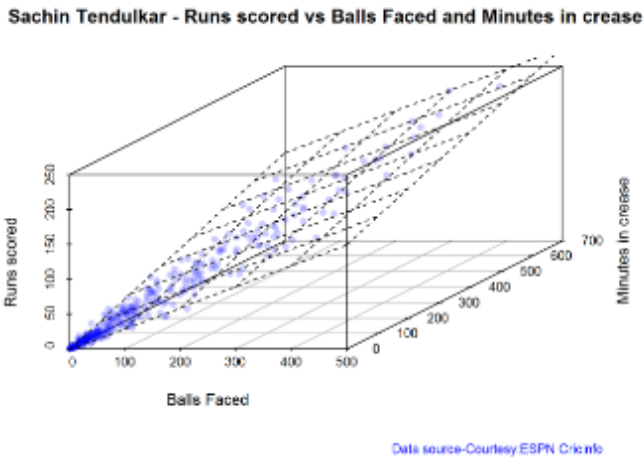
```
batsman4s("./tendulkar.csv","Tendulkar")
batsman6s("./tendulkar.csv","Tendulkar")

batsmanDismissals("./tendulkar.csv","Tendulkar")
```



1.1.4. 3D scatter plot and prediction plane

The plots below show the 3D scatter plot of Sachin's Runs versus Balls Faced and Minutes at crease. A linear regression model is then fitted between Runs and Balls Faced + Minutes at crease

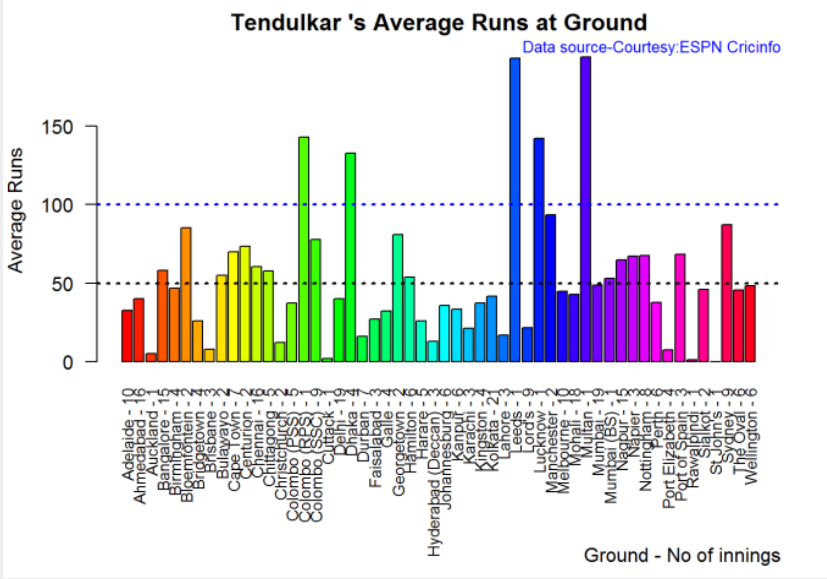
```
battingPerf3d("./tendulkar.csv","Sachin Tendulkar")
```



1.1.5. Average runs at different venues

The plot below gives the average runs scored by Tendulkar at different grounds. The plot also displays the number of innings at each ground as a label at x-axis. It can be seen Tendulkar did great in Colombo (SSC), Melbourne ifor matches overseas and Mumbai, Mohali and Bangalore at home

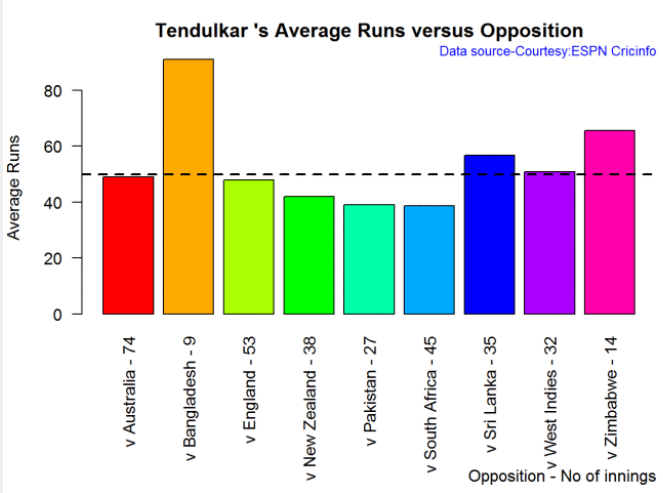
```
batsmanAvgRunsGround("./tendulkar.csv", "Sachin Tendulkar")
```



1.1.6. Average runs against different opposing teams

This plot computes the average runs scored by Tendulkar against different countries. The x-axis also gives the number of innings against each team

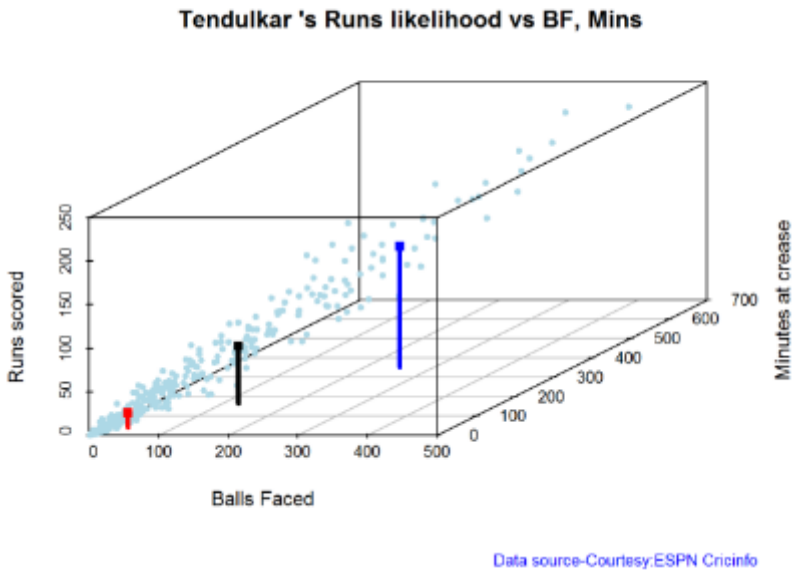
```
batsmanAvgRunsOpposition("./tendulkar.csv","Tendulkar")
```



1.1.7. Highest Runs Likelihood

The plot below shows the Runs Likelihood for a batsman. For this the performance of Sachin is plotted as a 3D scatter plot with Runs versus Balls Faced + Minutes at crease using. K-Means. The centroids of 3 clusters are computed and plotted. In this plot. Sachin Tendulkar's highest tendencies are computed and plotted using K-Means

```
batsmanRunsLikelihood("./tendulkar.csv","Sachin Tendulkar")
```




```
## Summary of  Sachin Tendulkar 's runs scoring likelihood
## ************************************************
##
## There is a 16.51 % likelihood that Sachin Tendulkar  will make
139 Runs in  251 balls over 353  Minutes
## There is a 58.41 % likelihood that Sachin Tendulkar  will make
16 Runs in  31 balls over  44  Minutes

## There is a 25.08 % likelihood that Sachin Tendulkar  will make

66 Runs in  122 balls over 167  Minutes
```

1.1.8. A look at the Top 4 batsman – Tendulkar, Kallis, Ponting and Sangakkara

The batsmen with the most hundreds in test cricket are

Sachin Tendulkar :**Average:53.78,100's – 51, 50's – 68**

Jacques Kallis : **Average: 55.47, 100's – 45, 50's – 58**

Ricky Ponting : **Average: 51.85, 100's – 41 , 50's – 62**

Kumara Sangakarra: **Average: 58.04 ,100's – 38 , 50's – 52**

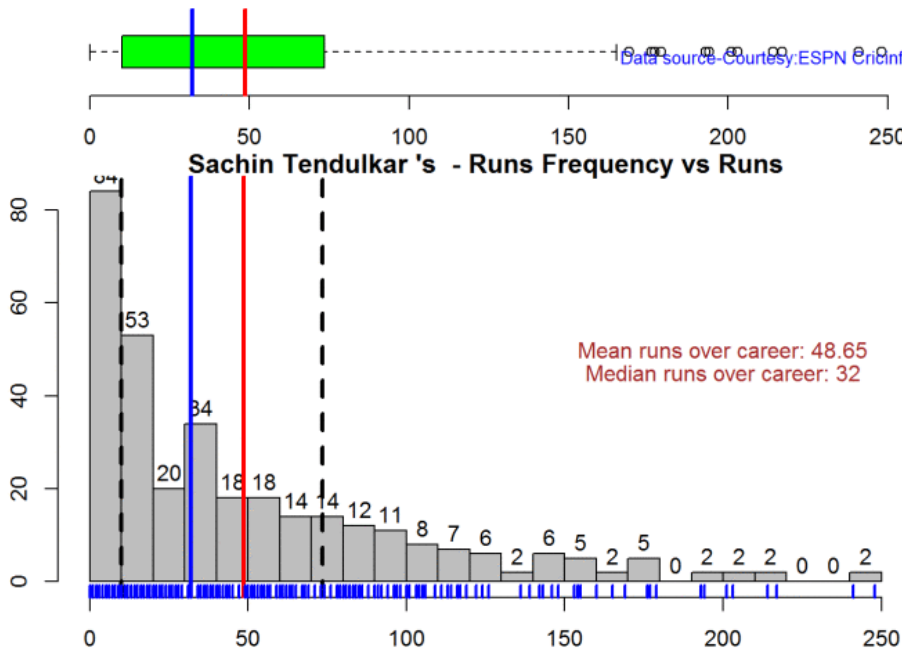
in that order.

The following plots take a closer at their performances. The box plots show the mean (red line) and median (blue line). The two ends of the boxplot display the 25th and 75th percentile.

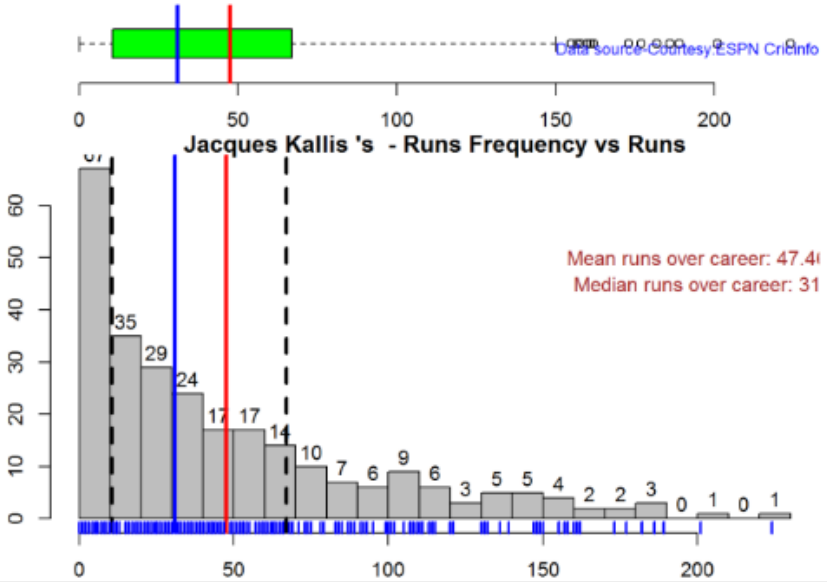
1.1.9. Box Histogram Plot

This plot shows a combined boxplot of the Runs ranges and a histogram of the Runs Frequency. The calculated Mean differ from the stated means possibly because of data cleaning. Also not sure how the means were arrived at ESPN Cricinfo for e.g. when considering not out..

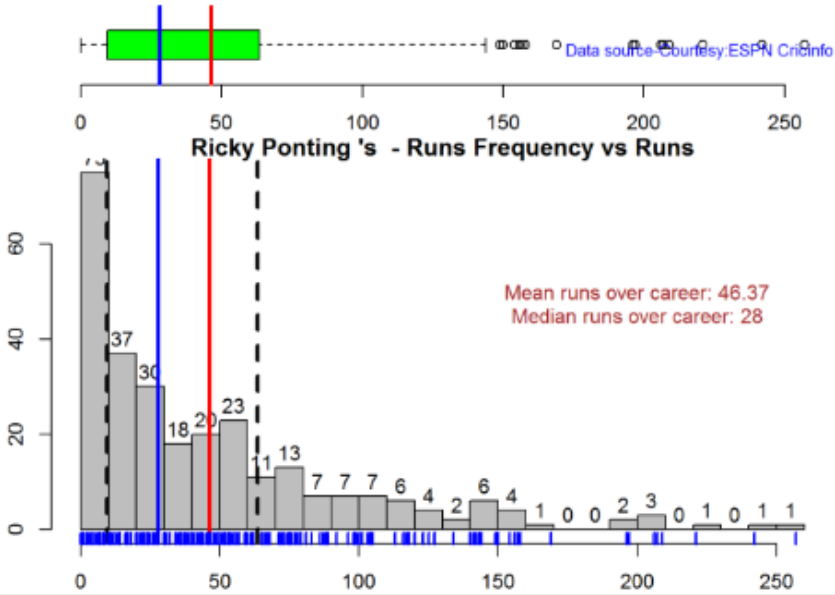
```
batsmanPerfBoxHist("./tendulkar.csv","Sachin Tendulkar")
```



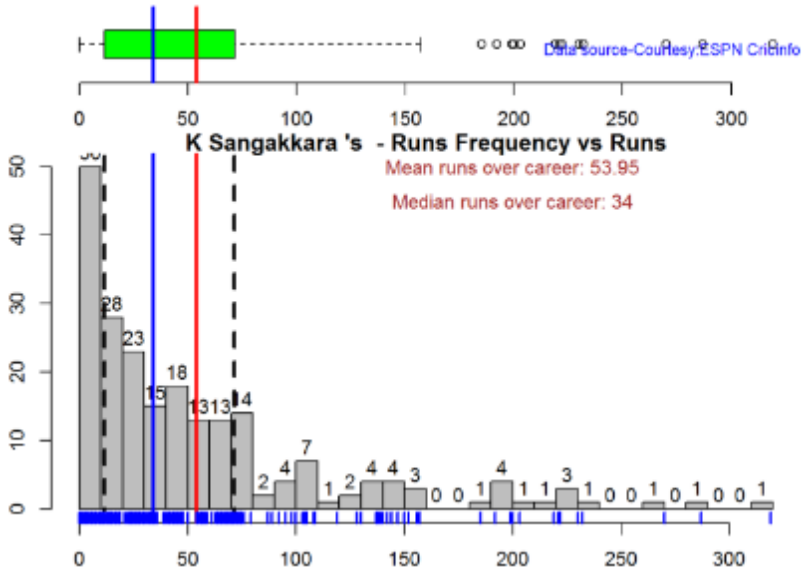

```
batsmanPerfBoxHist("./kallis.csv","Jacques Kallis")
```




```
batsmanPerfBoxHist("./ponting.csv","Ricky Ponting")
```




```
batsmanPerfBoxHist("./sangakkara.csv","K Sangakkara")
```



1.1.10. Contribution to won and lost matches

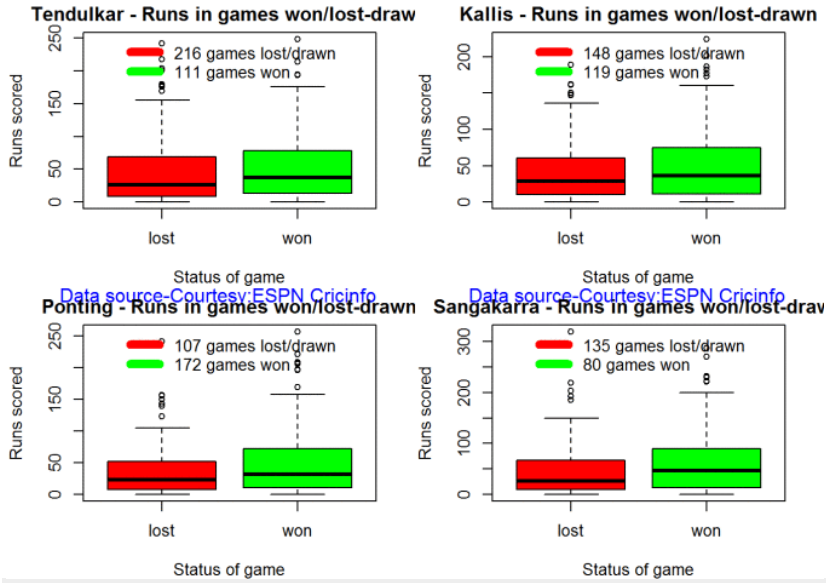
The plot below shows the contribution of Tendulkar, Kallis, Ponting and Sangakarra in matches won and lost. The plots show the range of runs scored as a boxplot (25th & 75th percentile) and the mean scored. The total matches won and lost are also printed in the plot.

All the players have scored more in the matches they won than the matches they lost. Ricky Ponting is the only batsman who seems to have more matches won to his credit than others. This could also be because he was a member of strong Australian team

For the next 2 functions below you will have to use the getPlayerDataSp() function. I have commented this as I already have these files

```
tendulkarsp <- getPlayerDataSp(35320,tdir=".",tfile="tendulkarsp.csv",ttype="batting")
kallissp <- getPlayerDataSp(45789,tdir=".",tfile="kallissp.csv",ttype="batting")
pontingsp <- getPlayerDataSp(7133,tdir=".",tfile="pontingsp.csv",ttype="batting")
sangakkarasp <-
getPlayerDataSp(50710,tdir=".",tfile="sangakkarasp.csv",ttype="batting")

par(mfrow=c(2,2))
par(mar=c(4,4,2,2))
batsmanContributionWonLost("tendulkarsp.csv","Tendulkar")
batsmanContributionWonLost("kallissp.csv","Kallis")
batsmanContributionWonLost("pontingsp.csv","Ponting")
```

```
batsmanContributionWonLost("sangakkarasp.csv","Sangakarra")
```



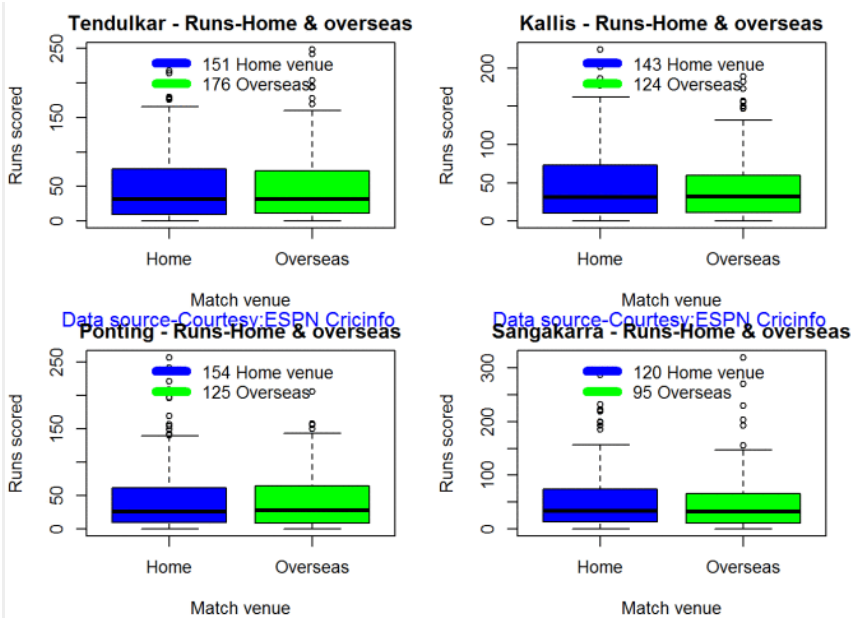

```
dev.off()
## null device
##           1
```

1.1.11. Performance at home and overseas

From the plot below it can be seen
Tendulkar has more matches overseas than at home and his performance is consistent in all venues at home or abroad. Ponting has lesser innings than Tendulkar and has an equally good performance at home and overseas. Kallis and Sangakkara's performance abroad is lower than the performance at home.

This function also requires the use of getPlayerDataSp() as shown above

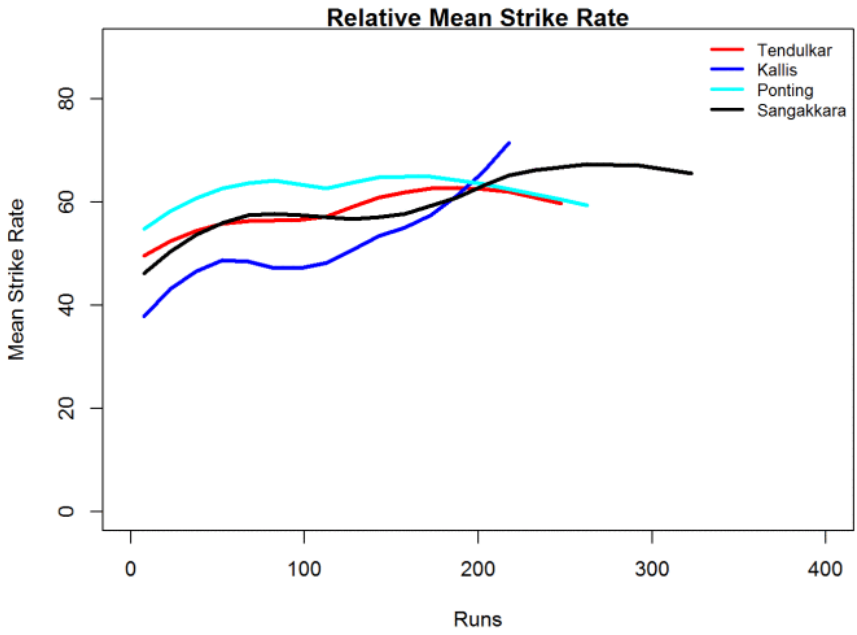
```
par(mfrow=c(2,2))
par(mar=c(4,4,2,2))
batsmanPerfHomeAway("tendulkarsp.csv","Tendulkar")
batsmanPerfHomeAway("kallissp.csv","Kallis")
batsmanPerfHomeAway("pontingsp.csv","Ponting")
batsmanPerfHomeAway("sangakkarasp.csv","Sangakarra")
dev.off()
```



1.1.12. Relative Mean Strike Rate plot

The plot below compares the Mean Strike Rate of the batsman for each of the runs ranges of 10 and plots them. The plot indicate the following Range 0 – 50 Runs – Ponting leads followed by Tendulkar Range 50 -100 Runs – Ponting followed by Sangakkara Range 100 – 150 – Ponting and then Tendulkar

```
frames <- list("./tendulkar.csv","./kallis.csv","ponting.csv","sangakkara.csv")
names <- list("Tendulkar","Kallis","Ponting","Sangakkara")

relativeBatsmanSR(frames,names)
```

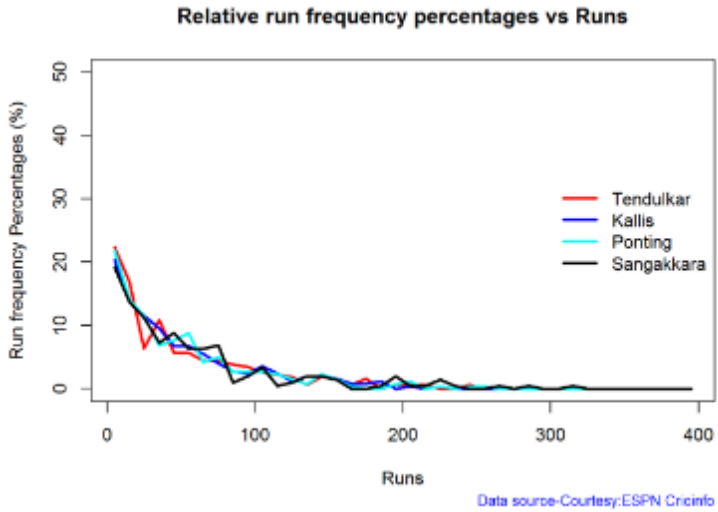


1.1.13. Relative Runs Frequency plot

The plot below gives the relative Runs Frequency Percentages for each 10 run bucket. The plot below show

Sangakkara leads followed by Ponting

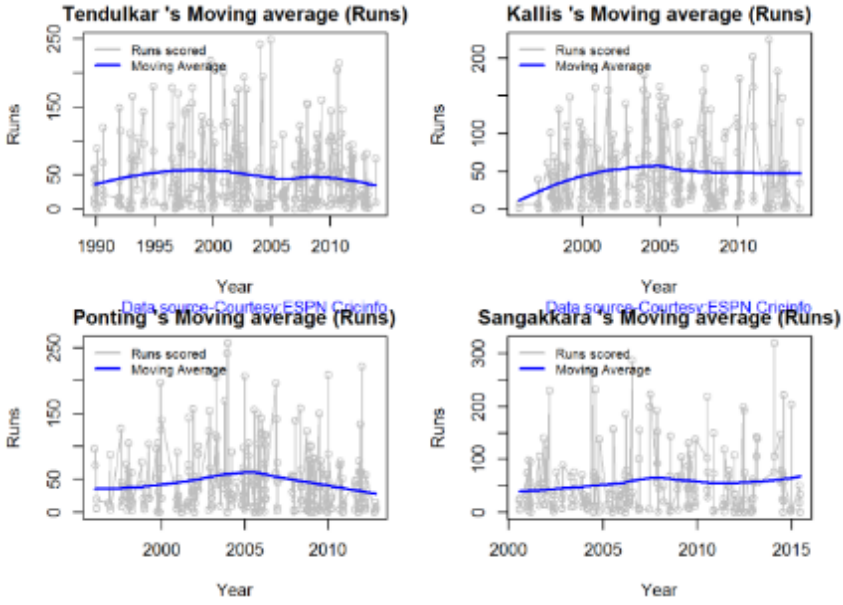
```
frames <- list("./tendulkar.csv","./kallis.csv","ponting.csv","sangakkara.csv")
names <- list("Tendulkar","Kallis","Ponting","Sangakkara")

relativeRunsFreqPerf(frames,names)
```



1.1.14. Moving Average of runs in career

Take a look at the Moving Average across the career of the Top 4. Clearly . Kallis and Sangakkara have a few more years of great batting ahead. They seem to average on 50. . Tendulkar and Ponting definitely show a slump in the later years

```
par(mfrow=c(2,2))
par(mar=c(4,4,2,2))
batsmanMovingAverage("./tendulkar.csv","Sachin Tendulkar")
batsmanMovingAverage("./kallis.csv","Jacques Kallis")
batsmanMovingAverage("./ponting.csv","Ricky Ponting")

batsmanMovingAverage("./sangakkara.csv","K Sangakkara")
```



1.1.15. Future Runs forecast

Here are plots that forecast how the batsman will perform in future. In this case 90% of the career runs trend is uses as the training set. the remaining 10% is the test set.

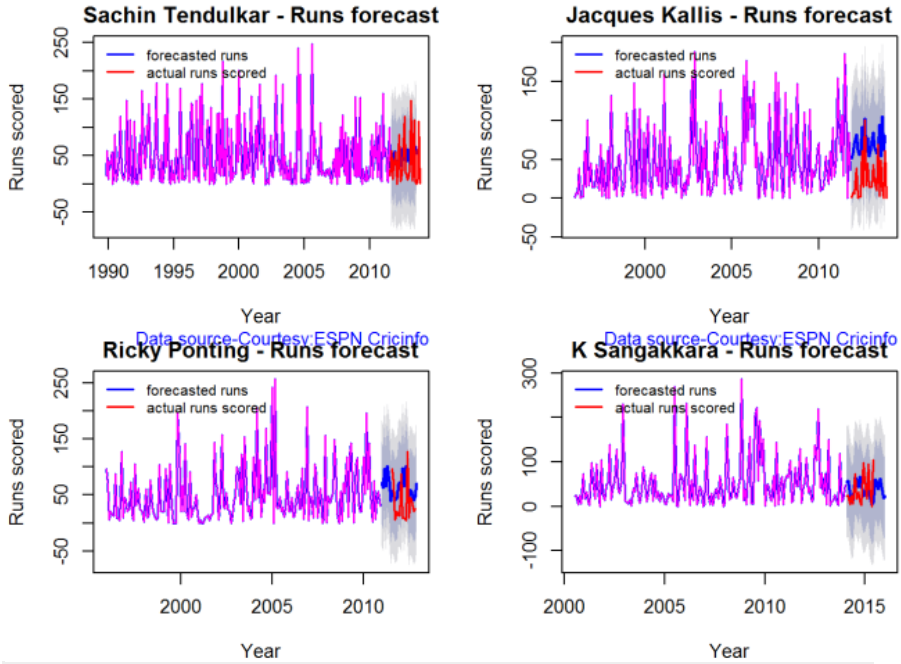
A Holt-Winters forecasting model is used to forecast future performance based on the 90% training set. The forecasted runs trend is plotted. The test set is also plotted to see how close the forecast and the actual matches

Take a look at the runs forecasted for the batsman below.

- Tendulkar's forecasted performance seems to tally with his actual performance with an average of 50
- Kallis the forecasted runs are higher than the actual runs he scored
- Ponting seems to have a good run in the future
- Sangakkara has a decent run in the future averaging 50 runs

```
par(mfrow=c(2,2))
par(mar=c(4,4,2,2))
batsmanPerfForecast("./tendulkar.csv","Sachin Tendulkar")
batsmanPerfForecast("./kallis.csv","Jacques Kallis")
batsmanPerfForecast("./ponting.csv","Ricky Ponting")
```

```
batsmanPerfForecast("./sangakkara.csv","K Sangakkara")
```




```
dev.off()
## null device
##           1
```

1.1.16. Check Batsman In-Form or Out-of-Form

The below computation uses Null Hypothesis testing and p-value to determine if the batsman is in-form or out-of-form. For this 90% of the career runs is chosen as the population and the mean computed. The last 10% is chosen to be the sample set and the sample Mean and the sample Standard Deviation are calculated.

The Null Hypothesis (H0) assumes that the batsman continues to stay in-form where the sample mean is within 95% confidence interval of population mean The Alternative (Ha) assumes that the batsman is out of

form the sample mean is beyond the 95% confidence interval of the population mean.

A significance value of 0.05 is chosen and p-value us computed If p-value >= .05 – Batsman In-Form If p-value < 0.05 – Batsman Out-of-Form

Note Ideally the p-value should be done for a population that follows the Normal Distribution. But the runs population is usually left skewed. So some correction may be needed. I will revisit this later

This is done for the Top 4 batsman

```
checkBatsmanInForm("./tendulkar.csv","Sachin Tendulkar")
##
************************************************************
************************
##
## Population size: 294  Mean of population: 50.48
## Sample size: 33  Mean of sample: 32.42 SD of sample: 29.8
##
## Null hypothesis H0 : Sachin Tendulkar 's sample average is
within 95% confidence interval
##         of population average
## Alternative hypothesis Ha : Sachin Tendulkar 's sample average
is below the 95% confidence
##         interval of population average
##
## [1] "Sachin Tendulkar 's Form Status: Out-of-Form because the
p value: 0.000713  is less than alpha=  0.05"

##

************************************************************

************************

checkBatsmanInForm("./kallis.csv","Jacques Kallis")
##
************************************************************
************************
##
```

```
## Population size: 240  Mean of population: 47.5
## Sample size: 27  Mean of sample: 47.11 SD of sample: 59.19
##
## Null hypothesis H0 : Jacques Kallis 's sample average is
within 95% confidence interval
##         of population average
## Alternative hypothesis Ha : Jacques Kallis 's sample average
is below the 95% confidence
##         interval of population average
##
## [1] "Jacques Kallis 's Form Status: In-Form because the p
value: 0.48647  is greater than alpha=  0.05"

##

************************************************************

************************
```

```
checkBatsmanInForm("./ponting.csv","Ricky Ponting")
##
************************************************************
************************
##
## Population size: 251  Mean of population: 47.5
## Sample size: 28  Mean of sample: 36.25 SD of sample: 48.11
##
## Null hypothesis H0 : Ricky Ponting 's sample average is within
95% confidence interval
##         of population average
## Alternative hypothesis Ha : Ricky Ponting 's sample average is
below the 95% confidence
##         interval of population average
##
## [1] "Ricky Ponting 's Form Status: In-Form because the p
value: 0.113115  is greater than alpha=  0.05"

##

************************************************************

************************
```

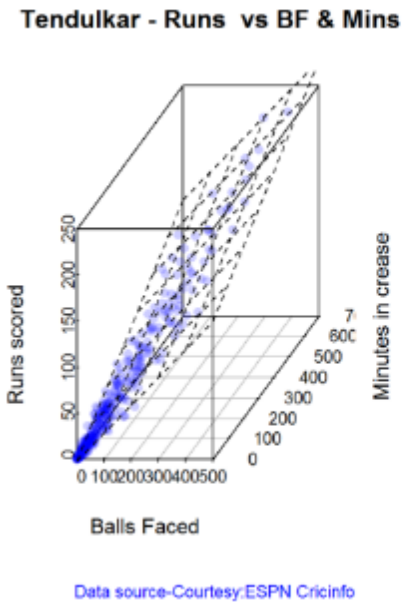
```
checkBatsmanInForm("./sangakkara.csv","K Sangakkara")
##
************************************************************
************************
##
## Population size: 193  Mean of population: 51.92
## Sample size: 22  Mean of sample: 71.73 SD of sample: 82.87
##
## Null hypothesis H0 : K Sangakkara 's sample average is within
95% confidence interval
```

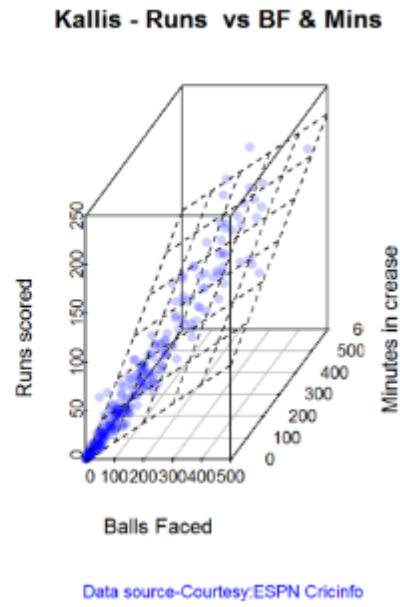
```
##         of population average
## Alternative hypothesis Ha : K Sangakkara 's sample average is
below the 95% confidence
##         interval of population average
##
## [1] "K Sangakkara 's Form Status: In-Form because the p value:
0.862862  is greater than alpha=  0.05"

##

*****************************************************************

*************************
```

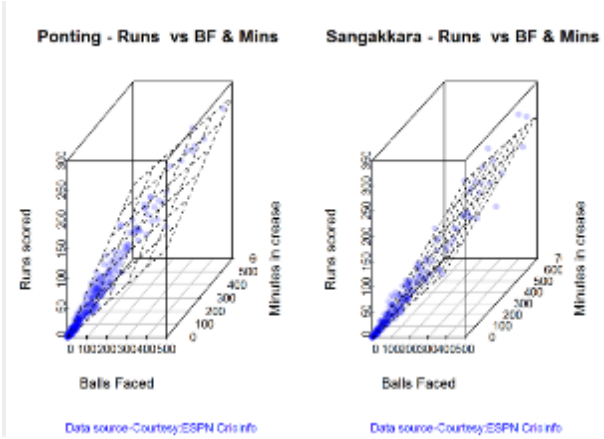
1.1.17. 3D plot of Runs vs Balls Faced and Minutes at Crease

The plot is a scatter plot of Runs vs Balls faced and Minutes at Crease. A prediction plane is fitted

```
par(mfrow=c(1,2))
par(mar=c(4,4,2,2))
battingPerf3d("./tendulkar.csv","Tendulkar")
battingPerf3d("./kallis.csv","Kallis")
par(mfrow=c(1,2))
par(mar=c(4,4,2,2))
battingPerf3d("./ponting.csv","Ponting")
battingPerf3d("./sangakkara.csv","Sangakkara")
```


Tendulkar - Runs vs BF & Mins
Runs scored
Balls Faced
Minutes in crease
Data source-Courtesy:ESPN Cricinfo


Kallis - Runs vs BF & Mins
Runs scored
Balls Faced
Minutes in crease
Data source-Courtesy:ESPN Cricinfo



1.1.18. Predicting Runs given Balls Faced and Minutes at Crease

A multi-variate regression plane is fitted between Runs and Balls faced +Minutes at crease. A sample sequence of Balls Faced(BF) and Minutes at crease (Mins) is setup as shown below. The fitted model is used to predict the runs for these values

```
BF <- seq( 10, 400,length=15)
Mins <- seq(30,600,length=15)
newDF <- data.frame(BF,Mins)
tendulkar <-
batsmanRunsPredict("./tendulkar.csv","Tendulkar",newdataframe=newDF)
kallis <- batsmanRunsPredict("./kallis.csv","Kallis",newdataframe=newDF)
ponting <- batsmanRunsPredict("./ponting.csv","Ponting",newdataframe=newDF)

sangakkara <-

batsmanRunsPredict("./sangakkara.csv","Sangakkara",newdataframe=newDF)
```

The fitted model is then used to predict the runs that the batsmen will score for a given Balls faced and Minutes at crease. It can be seen Ponting has the will score the highest for a given Balls Faced and Minutes at crease.

Ponting is followed by Tendulkar who has Sangakkara close on his heels and finally we have Kallis. This is intuitive as we have already seen that Ponting has a highest strike rate.

```
batsmen <-
cbind(round(tendulkar$Runs),round(kallis$Runs),round(ponting$Runs),round(sangakk
ara$Runs))
colnames(batsmen) <- c("Tendulkar","Kallis","Ponting","Sangakkara")
newDF <- data.frame(round(newDF$BF),round(newDF$Mins))
colnames(newDF) <- c("BallsFaced","MinsAtCrease")
predictedRuns <- cbind(newDF,batsmen)

predictedRuns
##    BallsFaced MinsAtCrease Tendulkar Kallis Ponting Sangakkara
## 1          10           30         7      6       9          2
## 2          38           71        23     20      25         18
## 3          66          111        39     34      42         34
## 4          94          152        54     48      59         50
## 5         121          193        70     62      76         66
## 6         149          234        86     76      93         82
## 7         177          274       102     90     110         98
## 8         205          315       118    104     127        114
## 9         233          356       134    118     144        130
## 10        261          396       150    132     161        146
## 11        289          437       165    146     178        162
## 12        316          478       181    159     194        178
## 13        344          519       197    173     211        194
## 14        372          559       213    187     228        210

## 15        400          600       229    201     245        226
```

1.1.19. Analysis of Top 3 wicket takers

The top 3 wicket takes in test history are

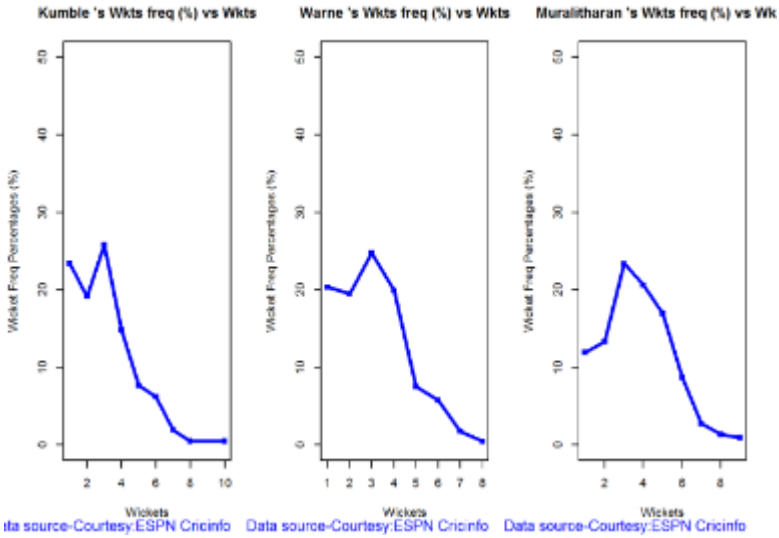
1. M Muralitharan: Wickets: 800, Average = 22.72, Economy Rate – 2.47
2. Shane Warne: Wickets: 708, Average = 25.41, Economy Rate – 2.65
3. Anil Kumble: Wickets: 619, Average = 29.65, Economy Rate – 2.69

How do Anil Kumble, Shane Warne and M Muralitharan compare with one another with respect to wickets taken and the Economy Rate. The next set of plots compute and plot precisely these analyses.

1.1.20. Wicket Frequency Plot

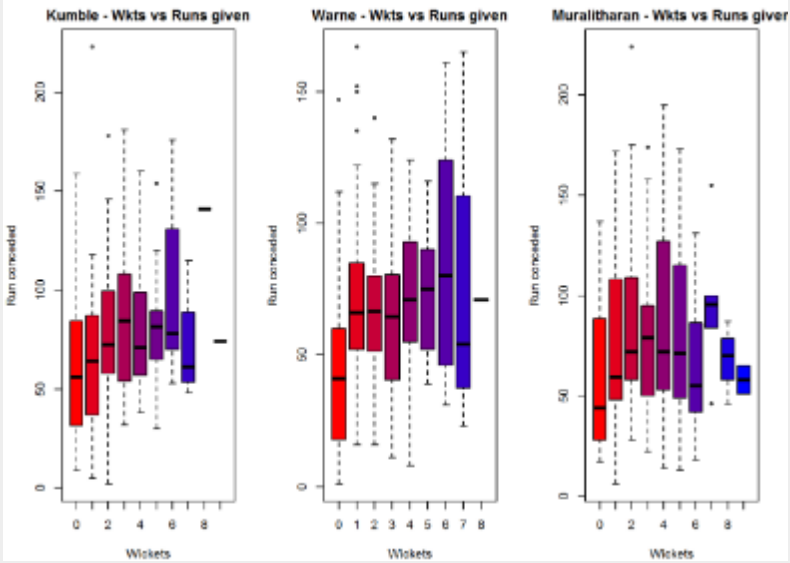
This plot below computes the percentage frequency of number of wickets taken for e.g. 1 wicket x%, 2 wickets y% etc. and plots them as a continuous line

```
par(mfrow=c(1,3))
par(mar=c(4,4,2,2))
bowlerWktsFreqPercent("./kumble.csv","Anil Kumble")
bowlerWktsFreqPercent("./warne.csv","Shane Warne")

bowlerWktsFreqPercent("./murali.csv","M Muralitharan")
```



1.1.21. Wickets Runs plot

```
par(mfrow=c(1,3))
par(mar=c(4,4,2,2))
bowlerWktsRunsPlot("./kumble.csv","Kumble")
bowlerWktsRunsPlot("./warne.csv","Warne")
bowlerWktsRunsPlot("./murali.csv","Muralitharan")
```

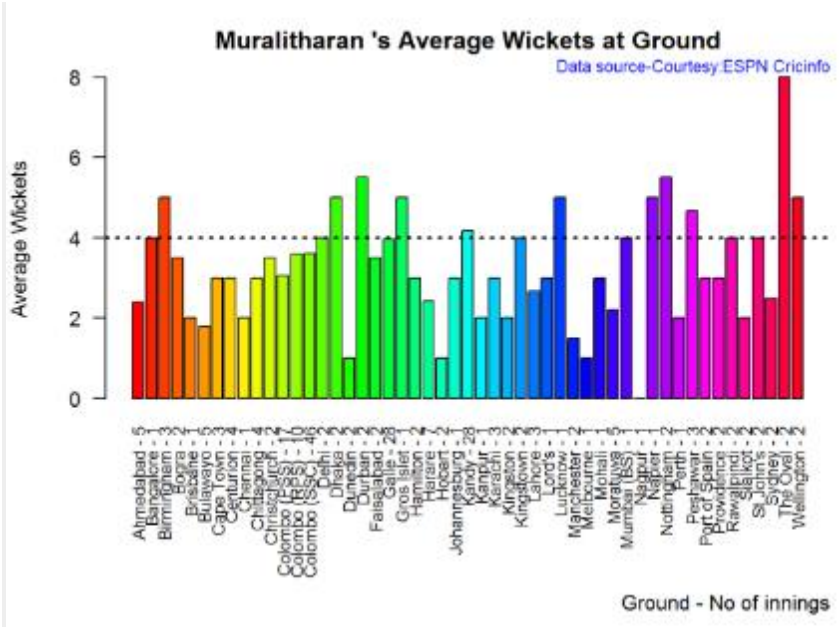



```
dev.off()
## null device

##           1
```

1.1.22. Average wickets at different venues

The plot gives the average wickets taken by Muralitharan at different venues. Muralitharan has taken an average of 8 and 6 wickets at Oval & Wellington respectively in 2 different innings. His best performances are at Kandy and Colombo (SSC)

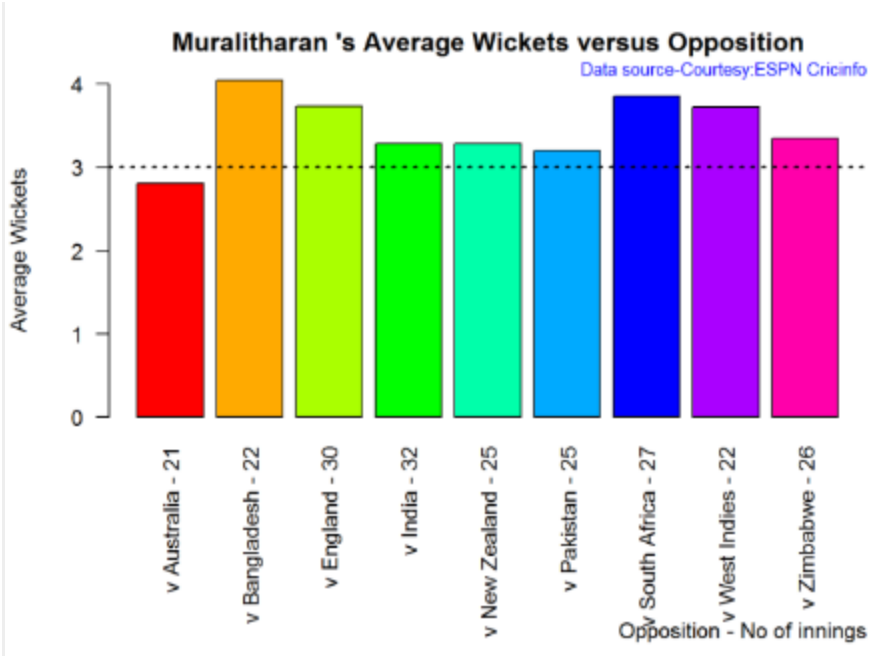
```
bowlerAvgWktsGround("./murali.csv","Muralitharan")
```



1.1.23. Average wickets against different opposition

The plot gives the average wickets taken by Muralitharan against different countries. The x-axis also includes the number of innings against each team

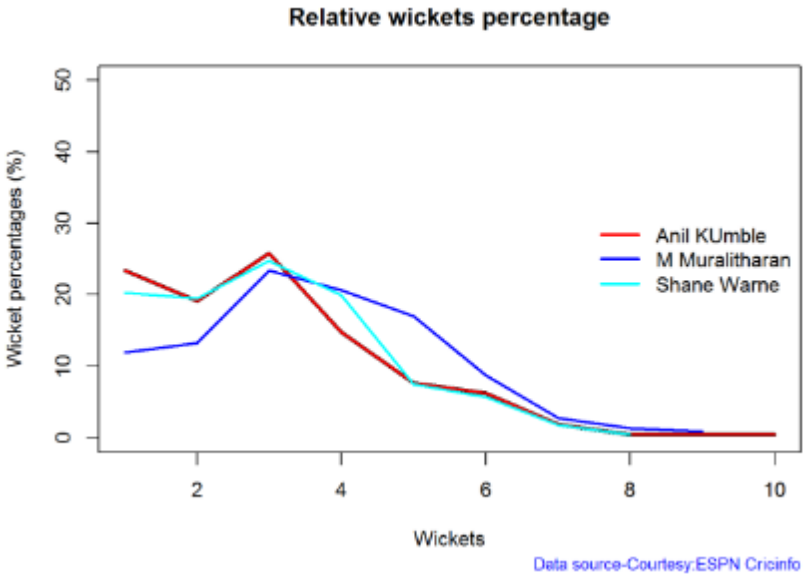
```
bowlerAvgWktsOpposition("./murali.csv","Muralitharan")
```



1.1.24. Relative Wickets Frequency Percentage

The Relative Wickets Percentage plot shows that M Muralitharan has a large percentage of wickets in the 3-8 wicket range

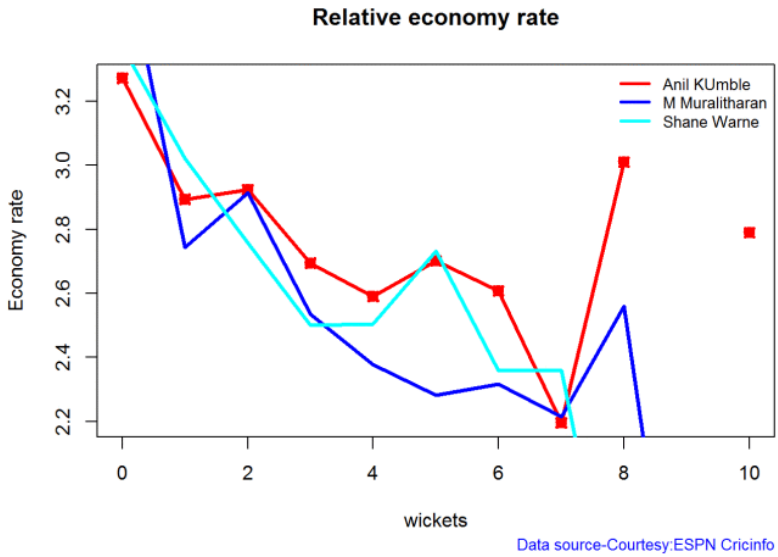
```
frames <- list("./kumble.csv","./murali.csv","warne.csv")
names <- list("Anil KUmble","M Muralitharan","Shane Warne")

relativeBowlingPerf(frames,names)
```



1.1.25. Relative Economy Rate against wickets taken

Clearly from the plot below it can be seen that Muralitharan has the best Economy Rate among the three

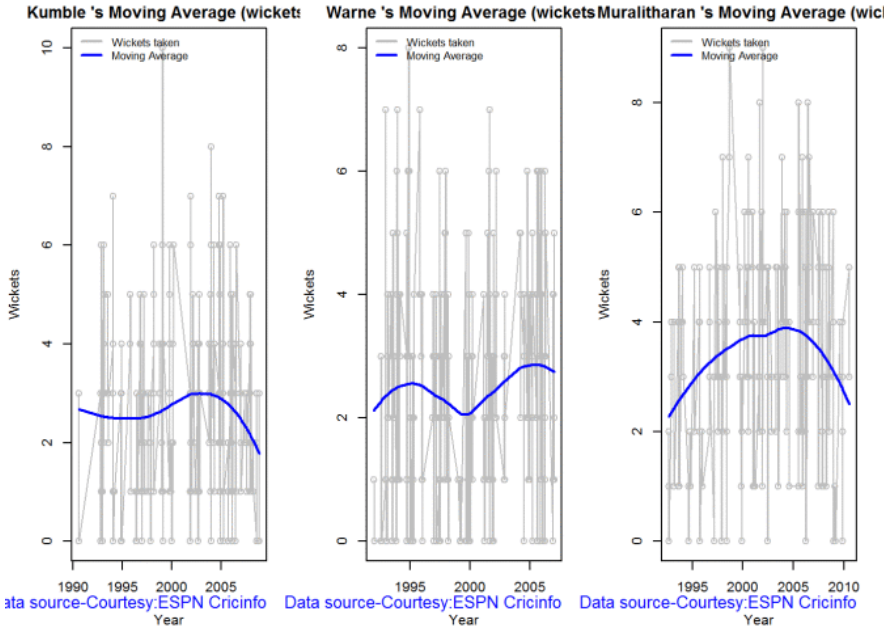
```
frames  <-  list("./kumble.csv","./murali.csv","warne.csv")
names  <-  list("Anil KUmble","M Muralitharan","Shane Warne")

relativeBowlingER(frames,names)
```



1.1.26. Wickets taken moving average

From the plot below it can be see 1. Shane Warne's performance at the time of his retirement was still at a peak of 3 wickets 2. M Muralitharan seems to have become ineffective over time with his peak years being 2004-2006 3. Anil Kumble also seems to slump down and become less effective.

```
par(mfrow=c(1,3))
par(mar=c(4,4,2,2))
bowlerMovingAverage("./kumble.csv","Anil Kumble")
bowlerMovingAverage("./warne.csv","Shane Warne")

bowlerMovingAverage("./murali.csv","M Muralitharan")
```



1.1.27. Future Wickets forecast

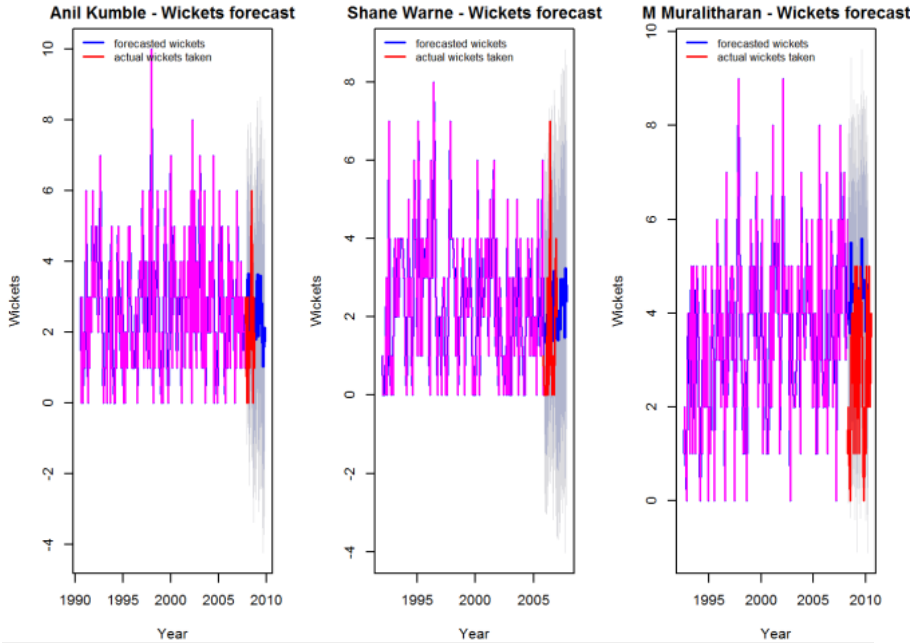
Here are plots that forecast how the bowler will perform in future. In this case 90% of the career wickets trend is used as the training set. the remaining 10% is the test set.

A Holt-Winters forecasting model is used to forecast future performance based on the 90% training set. The forecasted wickets trend is plotted. The test set is also plotted to see how close the forecast and the actual matches

Take a look at the wickets forecasted for the bowlers below. – Shane Warne and Muralitharan have a fairly consistent forecast – Kumble forecast shows a small dip

```
par(mfrow=c(1,3))
par(mar=c(4,4,2,2))
bowlerPerfForecast("./kumble.csv","Anil Kumble")
```

```
bowlerPerfForecast("./warne.csv","Shane Warne")

bowlerPerfForecast("./murali.csv","M Muralitharan")
```




```
dev.off()
## null device

##           1
```

1.1.28. Contribution to matches won and lost

The plot below is extremely interesting

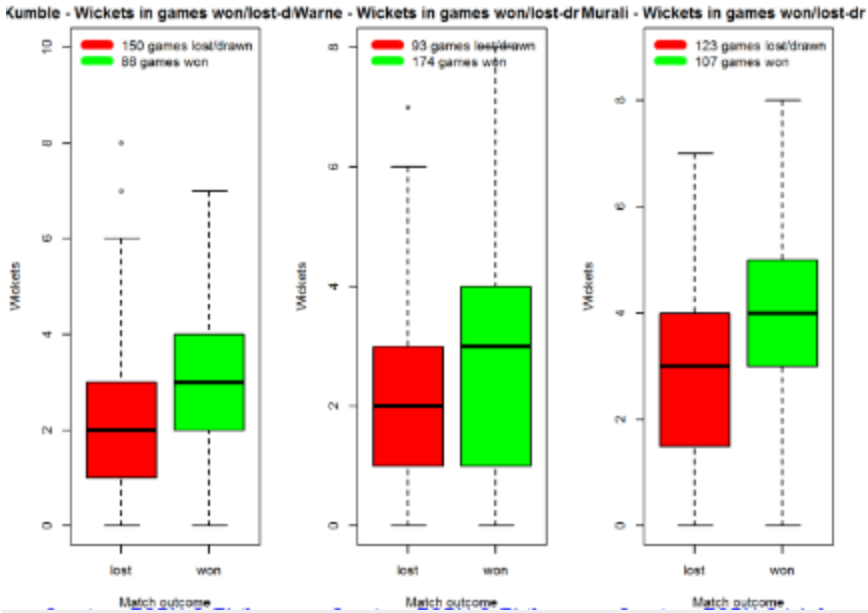
1. Kumble wickets range from 2 to 4 wickets in matches won with a mean of 3

2. Warne wickets in won matches range from 1 to 4 with more matches won. Clearly there are other bowlers contributing to the wins, possibly the pacers

3. Muralitharan wickets range in winning matches is more than the other 2 and ranges 3 to 5 and clearly had a hand (pun unintended) in Sri Lanka's wins

As discussed above the next 2 charts require the use of getPlayerDataSp()

```
kumblesp <- getPlayerDataSp(30176,tdir=".",tfile="kumblesp.csv",ttype="bowling")
warnesp <- getPlayerDataSp(8166,tdir=".",tfile="warnesp.csv",ttype="bowling")

muralisp <- getPlayerDataSp(49636,tdir=".",tfile="muralisp.csv",ttype="bowling")
par(mfrow=c(1,3))
par(mar=c(4,4,2,2))
bowlerContributionWonLost("kumblesp.csv","Kumble")
bowlerContributionWonLost("warnesp.csv","Warne")

bowlerContributionWonLost("muralisp.csv","Murali")
```

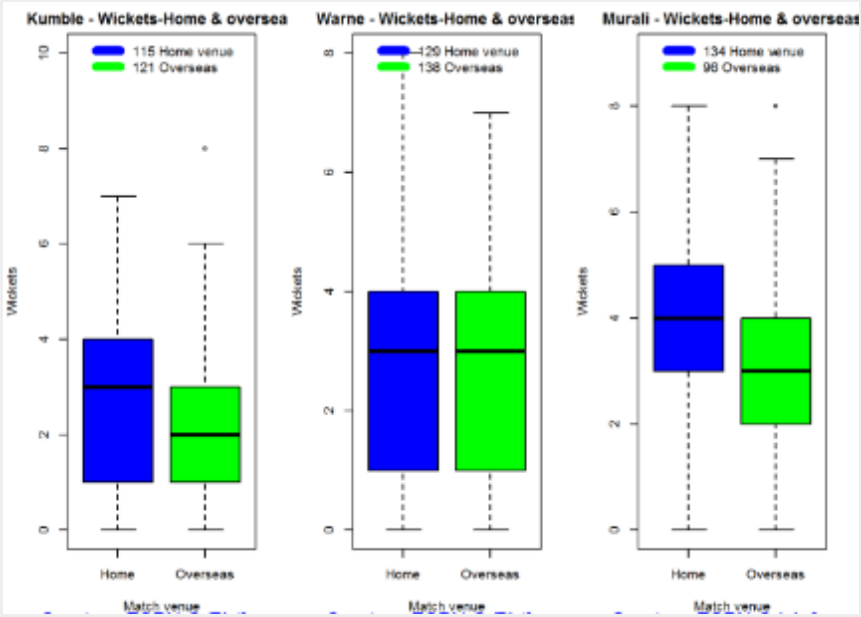



```
dev.off()
## null device
##           1
```

1.1.29. Performance home and overseas

From the plot below it can be seen that Kumble & Warne have played more matches overseas than Muralitharan. Both Kumble and Warne show an average of 2 wickers overseas, Muralitharan on the other hand has an average of 2.5 wickets overseas but a slightly less number of matches than Kumble & Warne

```
par(mfrow=c(1,3))
par(mar=c(4,4,2,2))
bowlerPerfHomeAway("kumblesp.csv","Kumble")
bowlerPerfHomeAway("warnesp.csv","Warne")
bowlerPerfHomeAway("muralisp.csv","Murali")
```




```
dev.off()
## null device
##           1
```

1.1.30. Check for bowler in-form/out-of-form

The below computation uses Null Hypothesis testing and p-value to determine if the bowler is in-form or out-of-form. For this 90% of the career wickets is chosen as the population and the mean computed. The last 10% is chosen to be the sample set and the sample Mean and the sample Standard Deviation are calculated.

The Null Hypothesis (H0) assumes that the bowler continues to stay in-form where the sample mean is within 95% confidence interval of

population mean The Alternative (Ha) assumes that the bowler is out of form the sample mean is beyond the 95% confidence interval of the population mean.

A significance value of 0.05 is chosen and p-value us computed If p-value >= .05 – Batsman In-Form If p-value < 0.05 – Batsman Out-of-Form

Note Ideally the p-value should be done for a population that follows the Normal Distribution. But the runs population is usually left skewed. So some correction may be needed. I will revisit this later

Note: The check for the form status of the bowlers indicate 1. That both Kumble and Muralitharan were out of form. This also shows in the moving average plot 2. Warne is still in great form and could have continued for a few more years. Too bad we didn't see his magic longer.

```
checkBowlerInForm("./kumble.csv","Anil Kumble")
##
************************************************************
************************
##
## Population size: 212  Mean of population: 2.69
## Sample size: 24  Mean of sample: 2.04 SD of sample: 1.55
##
## Null hypothesis H0 : Anil Kumble 's sample average is within
95% confidence interval
##         of population average
## Alternative hypothesis Ha : Anil Kumble 's sample average is
below the 95% confidence
##         interval of population average
##
## [1] "Anil Kumble 's Form Status: Out-of-Form because the p
value: 0.02549  is less than alpha=  0.05"
```

```
##
************************************************************
************************
checkBowlerInForm("./warne.csv","Shane Warne")
##
************************************************************
************************
##
## Population size: 240  Mean of population: 2.55
## Sample size: 27  Mean of sample: 2.56 SD of sample: 1.8
##
## Null hypothesis H0 : Shane Warne 's sample average is within
95% confidence interval
##         of population average
## Alternative hypothesis Ha : Shane Warne 's sample average is
below the 95% confidence
##         interval of population average
##
## [1] "Shane Warne 's Form Status: In-Form because the p value:
0.511409  is greater than alpha=  0.05"
##
************************************************************
************************
checkBowlerInForm("./murali.csv","M Muralitharan")
##
************************************************************
************************
##
## Population size: 207  Mean of population: 3.55
## Sample size: 23  Mean of sample: 2.87 SD of sample: 1.74
##
## Null hypothesis H0 : M Muralitharan 's sample average is
within 95% confidence interval
##         of population average
## Alternative hypothesis Ha : M Muralitharan 's sample average
is below the 95% confidence
##         interval of population average
##
## [1] "M Muralitharan 's Form Status: Out-of-Form because the p
value: 0.036828  is less than alpha=  0.05"
```

```
##
****************************************************************
************************
dev.off()
## null device
##           1
```

1.1.31. Key Findings

The plots above capture some of the capabilities and features of my **cricketr** package. Feel free to install the package and try it out. Please do keep in mind ESPN Cricinfo's Terms of Use.

Here are the main findings from the analysis above

1.1.32. Analysis of Top 4 batsman

The analysis of the Top 4 test batsman Tendulkar, Kallis, Ponting and Sangakkara show the following

1. Sangakkara has the highest average, followed by Tendulkar, Kallis and then Ponting.
2. Ponting has the highest strike rate followed by Tendulkar, Sangakkara and then Kallis
3. The predicted runs for a given Balls faced and Minutes at crease is highest for Ponting, followed by Tendulkar, Sangakkara and Kallis
4. The moving average for Tendulkar and Ponting shows a downward trend while Kallis and Sangakkara retired too soon
5. Tendulkar was out of form about the time of retirement while the rest were in-form. But this result has to be taken along with the moving average plot. Ponting was clearly on the way out.
6. The home and overseas performance indicate that Tendulkar is the clear leader. He has the highest number of matches played overseas and

his performance has been consistent. He is followed by Ponting, Kallis and finally Sangakkara

6.1.1. Analysis of Top 3 legs spinners

The analysis of Anil Kumble, Shane Warne and M Muralitharan show the following

1. Muralitharan has the highest wickets and best economy rate followed by Warne and Kumble
2. Muralitharan has higher wickets frequency percentage between 3 to 8 wickets
3. Muralitharan has the best Economy Rate for wickets between 2 to 7
4. The moving average plot shows that the time was up for Kumble and Muralitharan but Warne had a few years ahead
5. The check for form status shows that Muralitharan and Kumble time was over while Warne still in great form
6. Kumble's has more matches abroad than the other 2, yet Kumble averages of 3 wickets at home and 2 wickets overseas, while Warne . Muralitharan has played few matches but has an average of 4 wickets at home and 3 wickets overseas.

Final thoughts

Here are my final thoughts

6.1.2. Batting

Among the 4 batsman Tendulkar, Kallis, Ponting and Sangakkara the clear leader is Tendulkar for the following reasons

1. Tendulkar has the highest test centuries and runs of all time. Tendulkar's average is 2nd to Sangakkara, Tendulkar's predicted runs for a given Balls faced and Minutes at Crease is 2nd and is behind Ponting. Also Tendulkar's performance at home and overseas are consistent throughout despite the fact that he has a highest number of overseas matches

2. Ponting takes the 2nd spot with the 2nd highest number of centuries, 1st in Strike Rate and 2nd in home and away performance.
3. The 3rd spot goes to Sangakkara, with the highest average, 3rd highest number of centuries, reasonable run frequency percentage in different run ranges. However he has a fewer number of matches overseas and his performance overseas is significantly lower than at home
4. Kallis has the 2nd highest number of centuries but his performance overseas and strike rate are behind others

1. Finally Kallis and Sangakkara had a few good years of batting still left in them (pity they retired!) while Tendulkar and Ponting's time was up

1.1.1. Bowling

Muralitharan leads the way followed closely by Warne and finally Kumble.

The reasons are

1. Muralitharan has the highest number of test wickets with the best Wickets percentage and the best Economy Rate. Muralitharan on average gas taken 4 wickets at home and 3 wickets overseas
2. Warne follows Muralitharan in the highest wickets taken, however Warne has less matches overseas than Muralitharan and average 3 wickets home and 2 wickets overseas
3. Kumble has the 3rd highest wickets, with 3 wickets on an average at home and 2 wickets overseas. However Kumble has played more matches overseas than the other two. In that respect his performance is great. Also Kumble has played less matches at home otherwise his numbers would have looked even better.
4. Also while Kumble and Muralitharan's career was on the decline , Warne was going great and had a couple of years ahead.

Hope you have fun using the cricketr package as I had in developing it

1.2. Taking cricketr for a spin – Part 1

''Curiouser and curiouser!" cried Alice

"The time has come," the walrus said, "to talk of many things: Of shoes and ships – and sealing wax – of cabbages and kings"

"Begin at the beginning,"the King said, very gravely,"and go on till you come to the end: then stop."

"And what is the use of a book," thought Alice, "without pictures or conversation?"

Excerpt from Alice in Wonderland by Lewis Caroll

1.2.1. Introduction

This post is a continuation of my previous post "Introducing cricketr! A R package to analyze the performances of cricketers." In this post I take my package **cricketr** for a spin. For this analysis I focus on the Indian batting legends

– Sachin Tendulkar (Master Blaster)

– Rahul Dravid (The Will)

– Sourav Ganguly (The Dada Prince)

– Sunil Gavaskar (Little Master)

The package can be installed directly from CRAN

```
if (!require("cricketr")){
   install.packages("cricketr",lib = "c:/test")
}

library(cricketr)
```

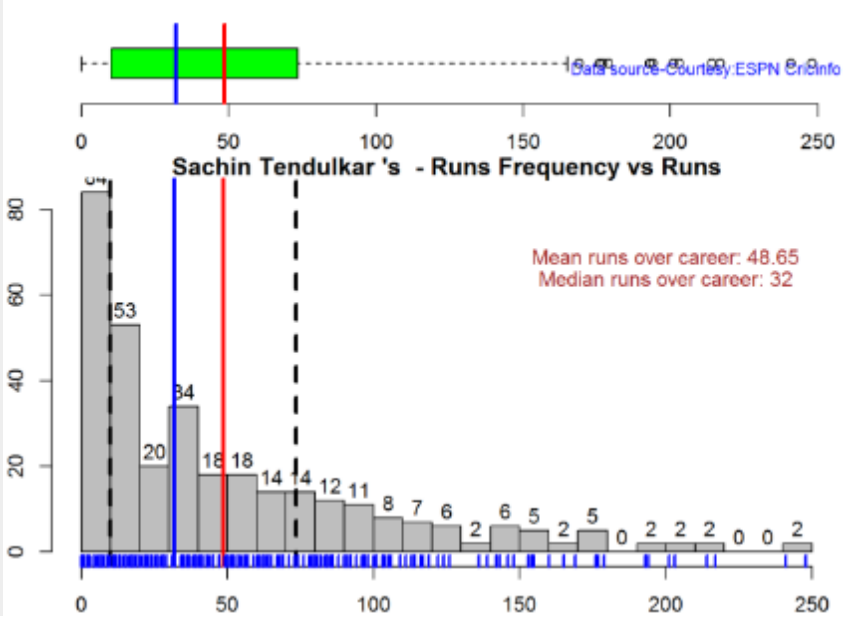
or from Github

```
library(devtools)
install_github("tvganesh/cricketr")
library(cricketr)
```

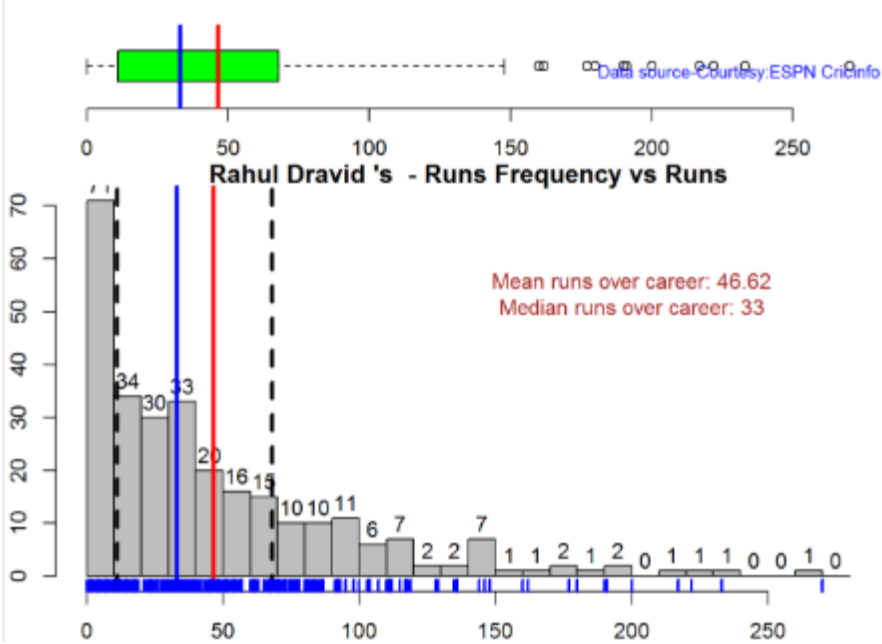
1.2.2. Box Histogram Plot

This plot shows a combined boxplot of the Runs ranges and a histogram of the Runs Frequency The plot below indicate the Tendulkar's average is the highest. He is followed by Dravid, Gavaskar and then Ganguly

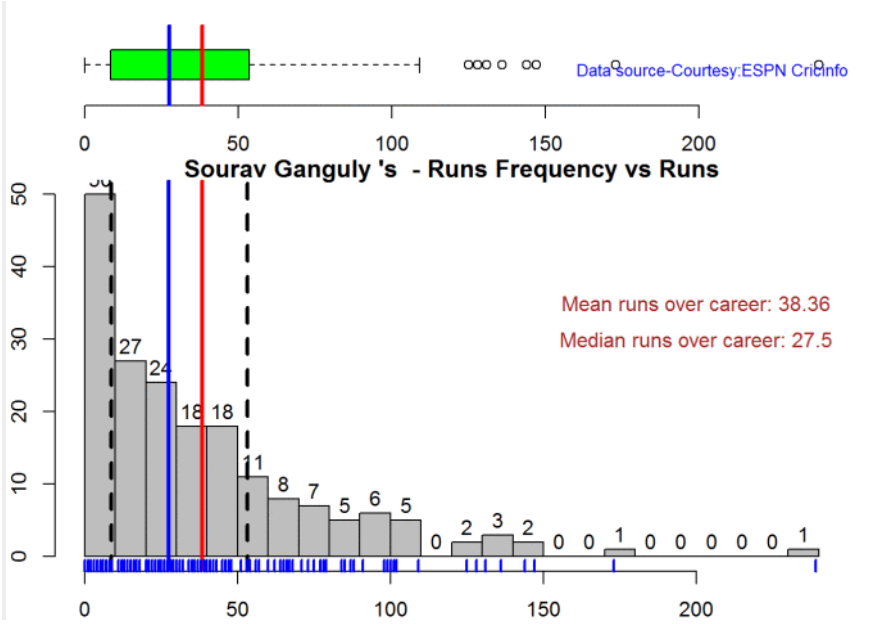
```
batsmanPerfBoxHist("./tendulkar.csv","Sachin Tendulkar")
```



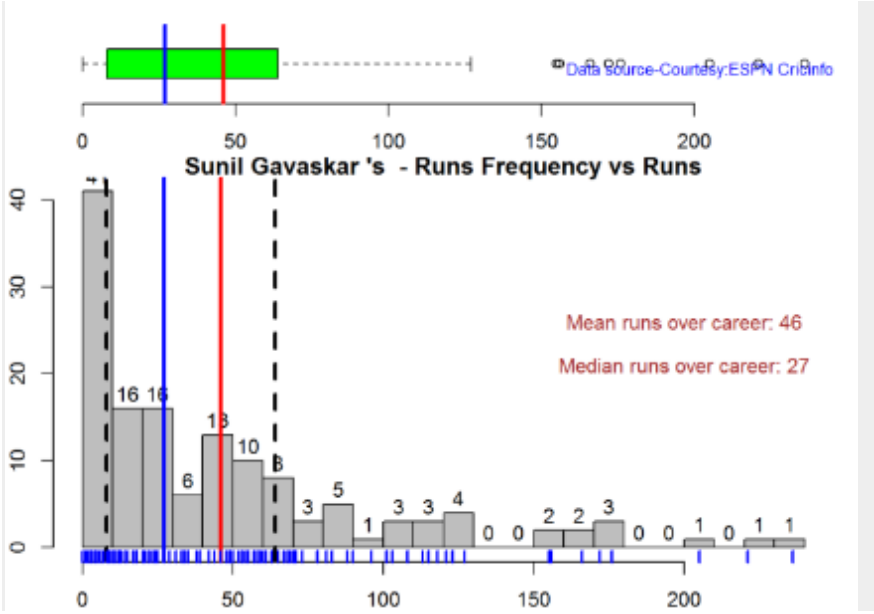

```
batsmanPerfBoxHist("./dravid.csv","Rahul Dravid")
```



batsmanPerfBoxHist("./ganguly.csv", "Sourav Ganguly")



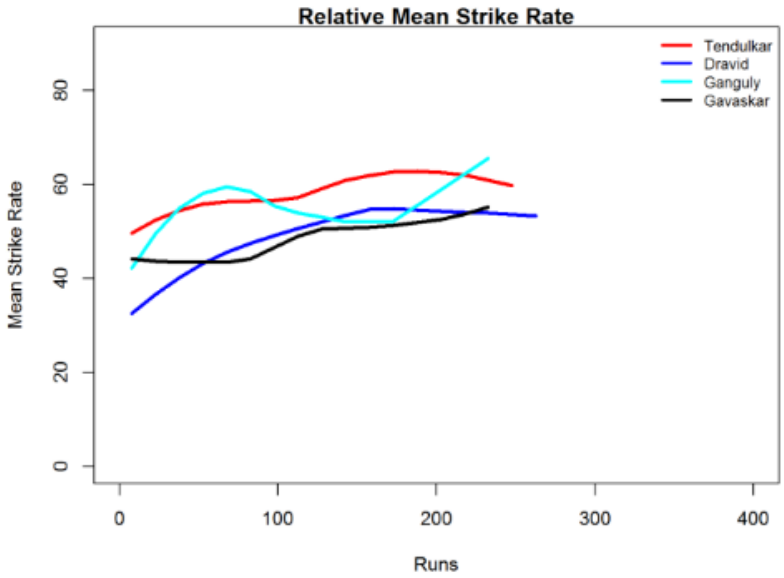

```
batsmanPerfBoxHist("./gavaskar.csv",,"Sunil Gavaskar")
```



1.2.3. Relative Mean Strike Rate

In this first plot I plot the Mean Strike Rate of the batsmen. Tendulkar leads in the Mean Strike Rate for each runs in the range 100- 180. Ganguly has a very good Mean Strike Rate for runs range 40 -80

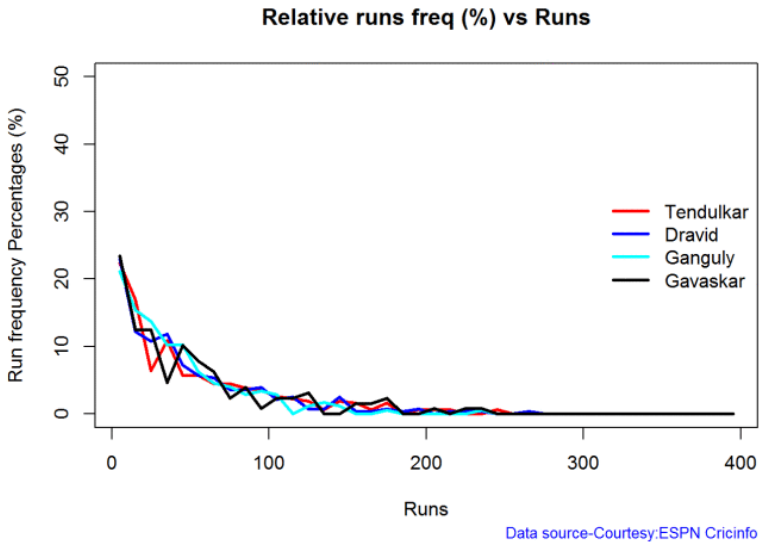
```
frames <- list("./tendulkar.csv","./dravid.csv","ganguly.csv","gavaskar.csv")
names <- list("Tendulkar","Dravid","Ganguly","Gavaskar")

relativeBatsmanSR(frames,names)
```



1.2.4. Relative Runs Frequency Percentage

The plot below show the percentage contribution in each 10 runs bucket over the entire career. The percentage Runs Frequency is fairly close but Gavaskar seems to lead most of the way

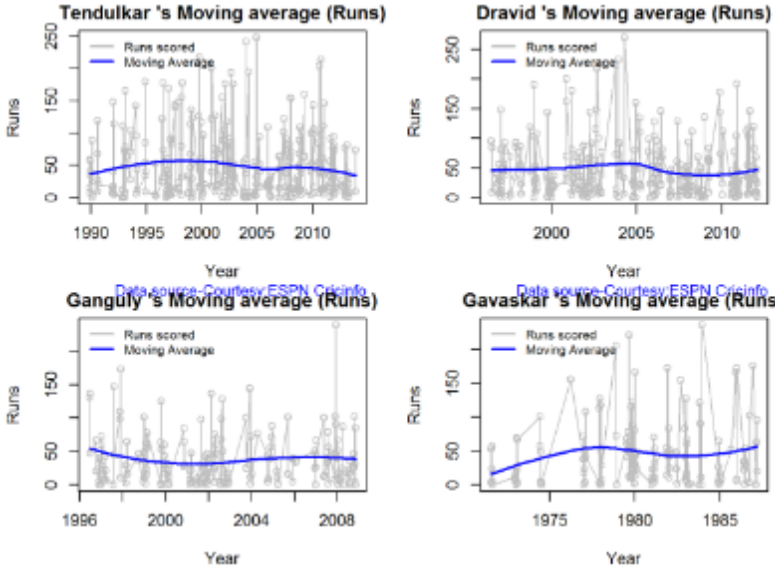
```
frames <- list("./tendulkar.csv","./dravid.csv","ganguly.csv","gavaskar.csv")
names <- list("Tendulkar","Dravid","Ganguly","Gavaskar")

relativeRunsFreqPerf(frames,names)
```



1.2.5. Moving Average of runs over career

The moving average for the 4 batsmen indicate the following – Tendulkar and Ganguly's career has a downward trend and their retirement didn't come too soon – Dravid and Gavaskar's career definitely shows an upswing. They probably had a year or two left.

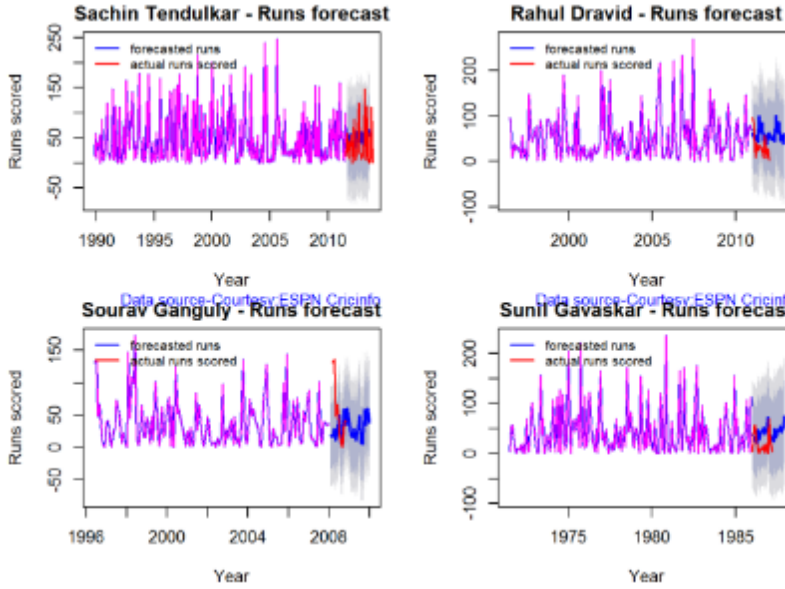
```
par(mfrow=c(2,2))
par(mar=c(4,4,2,2))
batsmanMovingAverage("./tendulkar.csv","Tendulkar")
batsmanMovingAverage("./dravid.csv","Dravid")
batsmanMovingAverage("./ganguly.csv","Ganguly")

batsmanMovingAverage("./gavaskar.csv","Gavaskar")
```



1.2.6. Runs forecast

The forecast for the batsman is shown below. The plots indicate that only Tendulkar seemed to maintain a consistency over the period while the rest seem to score less than their forecasted runs in the last 10% of the career

```
par(mfrow=c(2,2))
par(mar=c(4,4,2,2))
batsmanPerfForecast("./tendulkar.csv","Sachin Tendulkar")
batsmanPerfForecast("./dravid.csv","Rahul Dravid")
batsmanPerfForecast("./ganguly.csv","Sourav Ganguly")

batsmanPerfForecast("./gavaskar.csv","Sunil Gavaskar")
```



1.2.7. Check for batsman in-form/out-of-form

The following snippet checks whether the batsman is in-inform or ouyt-of-form during the last 10% innings of the career. This is done by choosing the null hypothesis (h0) to indicate that the batsmen are in-form. Ha is the alternative hypothesis that they are not-in-form. The population is based on the 1st 90% of career runs. The last 10% is taken as the sample and a check is made on the lower tail to see if the sample mean is less than 95% confidence interval. If this difference is >0.05 then the batsman is considered out-of-form.

The computation show that Tendulkar was out-of-form while the other's weren't. While Dravid and Gavaskar's moving average do show an upward trend the surprise is Ganguly. This could be that Ganguly was able to keep his average in the last 10% to with the 95$ confidence interval. It has to be noted that Ganguly's average was much lower than Tendulkar

```
checkBatsmanInForm("./tendulkar.csv","Tendulkar")
##
************************************************************
************************
##
## Population size: 294  Mean of population: 50.48
## Sample size: 33  Mean of sample: 32.42 SD of sample: 29.8
##
## Null hypothesis H0 : Tendulkar 's sample average is within 95%
confidence interval
##         of population average
## Alternative hypothesis Ha : Tendulkar 's sample average is
below the 95% confidence
##         interval of population average
##
## [1] "Tendulkar 's Form Status: Out-of-Form because the p
value: 0.000713  is less than alpha=  0.05"

##

************************************************************

************************

checkBatsmanInForm("./dravid.csv","Dravid")
##
************************************************************
************************
##
## Population size: 256  Mean of population: 46.98
## Sample size: 29  Mean of sample: 43.48 SD of sample: 40.89
##
## Null hypothesis H0 : Dravid 's sample average is within 95%
confidence interval
##         of population average
## Alternative hypothesis Ha : Dravid 's sample average is below
the 95% confidence
##         interval of population average
##
## [1] "Dravid 's Form Status: In-Form because the p value:
0.324138  is greater than alpha=  0.05"

##

************************************************************

************************

checkBatsmanInForm("./ganguly.csv","Ganguly")
##
************************************************************
************************
```

```
##
## Population size: 169  Mean of population: 38.94
## Sample size: 19  Mean of sample: 33.21 SD of sample: 32.97
##
## Null hypothesis H0 : Ganguly 's sample average is within 95%
confidence interval
##         of population average
## Alternative hypothesis Ha : Ganguly 's sample average is below
the 95% confidence
##         interval of population average
##
## [1] "Ganguly 's Form Status: In-Form because the p value:
0.229006  is greater than alpha=  0.05"

##

****************************************************************

************************
```

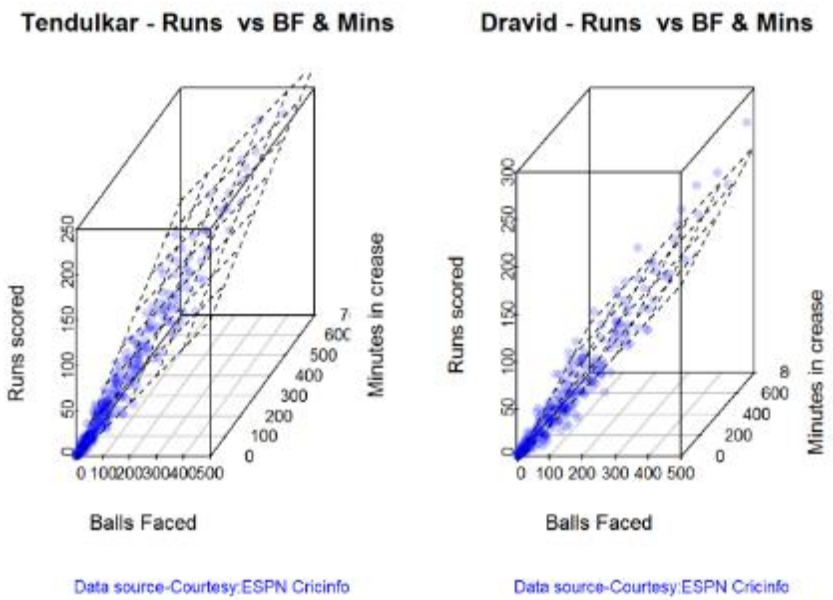
```
checkBatsmanInForm("./gavaskar.csv","Gavaskar")
```

```
##
****************************************************************
************************
##
## Population size: 125  Mean of population: 44.67
## Sample size: 14  Mean of sample: 57.86 SD of sample: 58.55
##
## Null hypothesis H0 : Gavaskar 's sample average is within 95%
confidence interval
##         of population average
## Alternative hypothesis Ha : Gavaskar 's sample average is
below the 95% confidence
##         interval of population average
##
## [1] "Gavaskar 's Form Status: In-Form because the p value:
0.793276  is greater than alpha=  0.05"

##

****************************************************************

************************
```

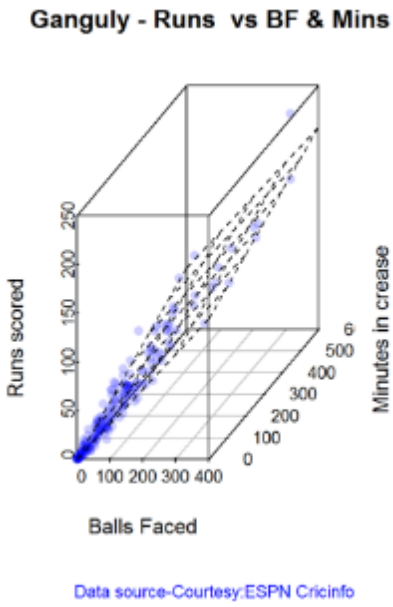
1.2.8. 3D plot of Runs vs Balls Faced and Minutes at Crease

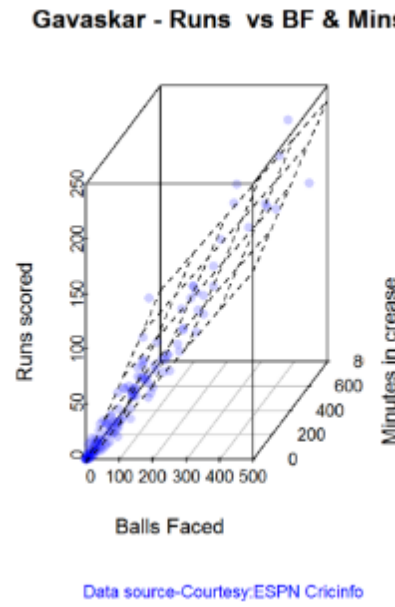
The plot is a scatter plot of Runs vs Balls faced and Minutes at Crease. A prediction plane is fitted

```
par(mfrow=c(1,2))
par(mar=c(4,4,2,2))
battingPerf3d("./tendulkar.csv","Tendulkar")

battingPerf3d("./dravid.csv","Dravid")
```




```
par(mfrow=c(1,2))
par(mar=c(4,4,2,2))
battingPerf3d("./ganguly.csv","Ganguly")

battingPerf3d("./gavaskar.csv","Gavaskar")
```





1.2.9. Predicting Runs given Balls Faced and Minutes at Crease

A multi-variate regression plane is fitted between Runs and Balls faced +Minutes at crease.

```
BF <- seq( 10, 400,length=15)
Mins <- seq(30,600,length=15)
newDF <- data.frame(BF,Mins)
tendulkar <-
batsmanRunsPredict("./tendulkar.csv","Tendulkar",newdataframe=newDF)
dravid <- batsmanRunsPredict("./dravid.csv","Dravid",newdataframe=newDF)
ganguly <- batsmanRunsPredict("./ganguly.csv","Ganguly",newdataframe=newDF)

gavaskar <- batsmanRunsPredict("./gavaskar.csv","Gavaskar",newdataframe=newDF)
```

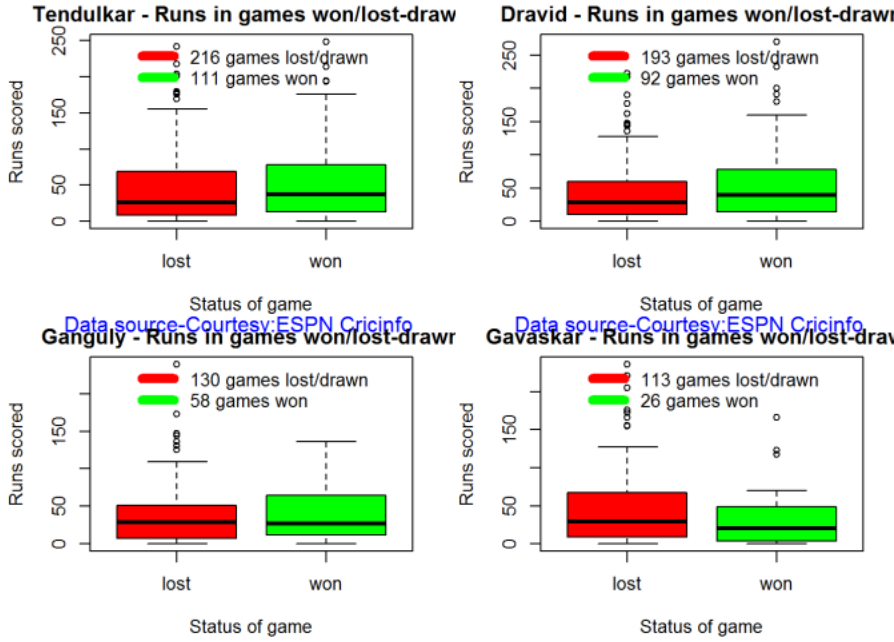
The fitted model is then used to predict the runs that the batsmen will score for a given Balls faced and Minutes at crease. It can be seen Tendulkar has a much higher Runs scored than all of the others.

Tendulkar is followed by Ganguly who we saw earlier had a very good strike rate. However it must be noted that Dravid and Gavaskar have a better average.

```
batsmen <-
cbind(round(tendulkar$Runs),round(dravid$Runs),round(ganguly$Runs),round(gavask
ar$Runs))
colnames(batsmen) <- c("Tendulkar","Dravid","Ganguly","Gavaskar")
newDF <- data.frame(round(newDF$BF),round(newDF$Mins))
colnames(newDF) <- c("BallsFaced","MinsAtCrease")
predictedRuns <- cbind(newDF,batsmen)

predictedRuns
##    BallsFaced MinsAtCrease Tendulkar Dravid Ganguly Gavaskar
## 1          10           30         7      1       7        4
## 2          38           71        23     14      21       17
## 3          66          111        39     27      35       30
## 4          94          152        54     40      50       43
## 5         121          193        70     54      64       56
## 6         149          234        86     67      78       69
## 7         177          274       102     80      93       82
## 8         205          315       118     94     107       95
## 9         233          356       134    107     121      108
## 10        261          396       150    120     136      121
## 11        289          437       165    134     150      134
## 12        316          478       181    147     165      147
## 13        344          519       197    160     179      160
## 14        372          559       213    173     193      173

## 15        400          600       229    187     208      186
```

1.2.10. Contribution to matches won and lost

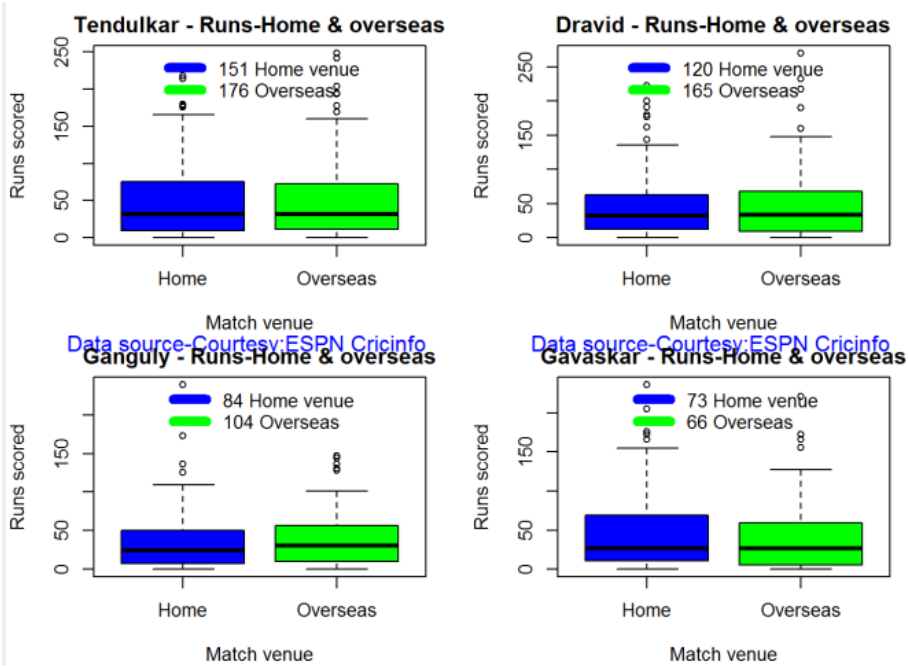
```
par(mfrow=c(2,2))
par(mar=c(4,4,2,2))
batsmanContributionWonLost(35320,"Tendulkar")
batsmanContributionWonLost(28114,"Dravid")
batsmanContributionWonLost(28779,"Ganguly")

batsmanContributionWonLost(28794,"Gavaskar")
```



1.2.11. Home and overseas performance

From the plot below Tendulkar and Dravid have a lot more matches both home and abroad and their performance has good both at home and overseas. Tendulkar has the best performance home and abroad and is consistent all across. Dravid is also consistent at all venues. Gavaskar played fewer matches than Tendulkar & Dravid. The range of runs at home is higher than overseas, however the average is consistent both at home and abroad. Finally we have Ganguly.

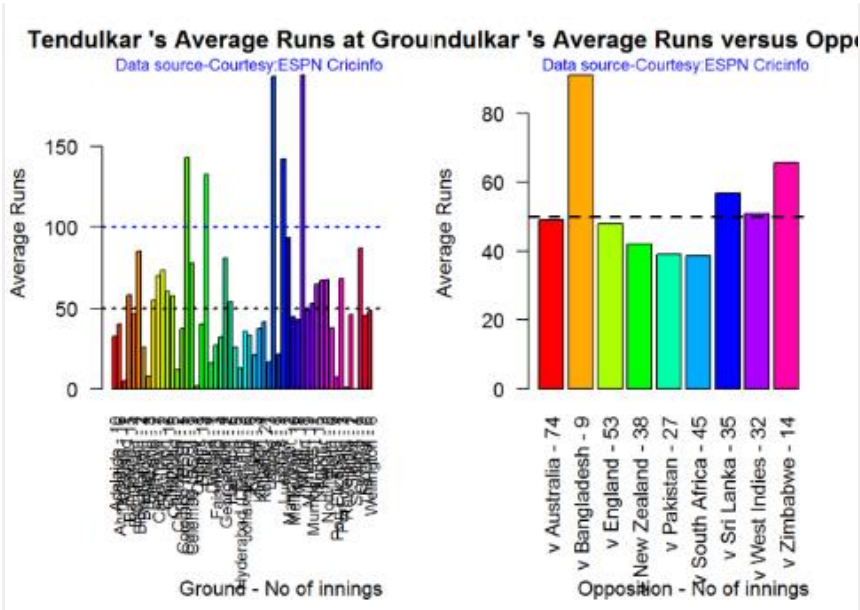
```
par(mfrow=c(2,2))
par(mar=c(4,4,2,2))
batsmanPerfHomeAway(35320,"Tendulkar")
batsmanPerfHomeAway(28114,"Dravid")
batsmanPerfHomeAway(28779,"Ganguly")
batsmanPerfHomeAway(28794,"Gavaskar")
```



1.2.12. Average runs at ground and against opposition

Tendulkar has above 50 runs average against Sri Lanka, Bangladesh, West Indies and Zimbabwe. The performance against Australia and England average very close to 50. Sydney, Port Elizabeth, Bloemfontein, Colombo are great hunting grounds for Tendulkar

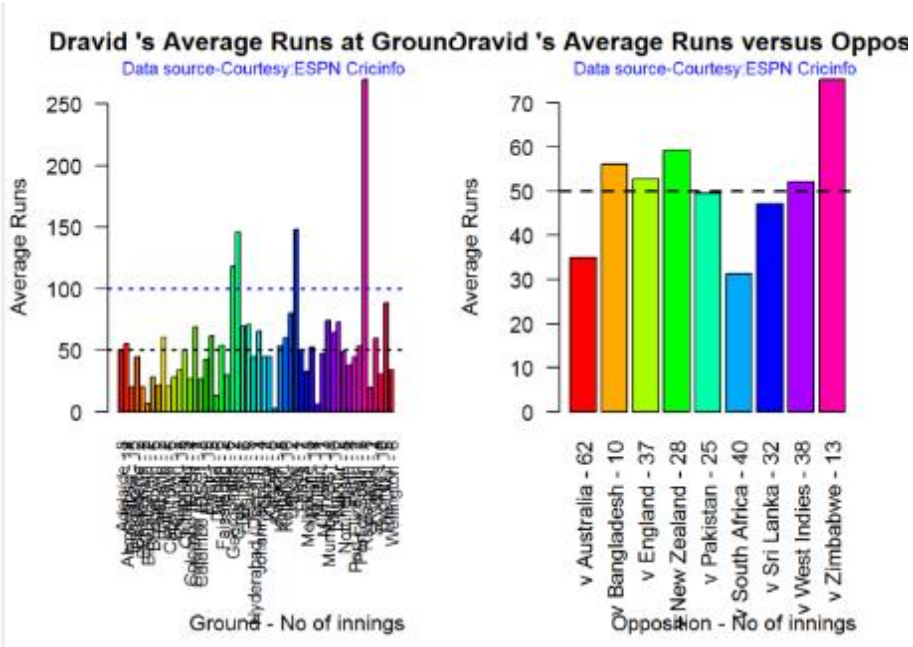
```
par(mfrow=c(1,2))
par(mar=c(4,4,2,2))
batsmanAvgRunsGround("./tendulkar.csv","Tendulkar")
batsmanAvgRunsOpposition("./tendulkar.csv","Tendulkar")
```



Dravid plundered runs at Adelaide, Georgetown, Oval, Hamilton etc.

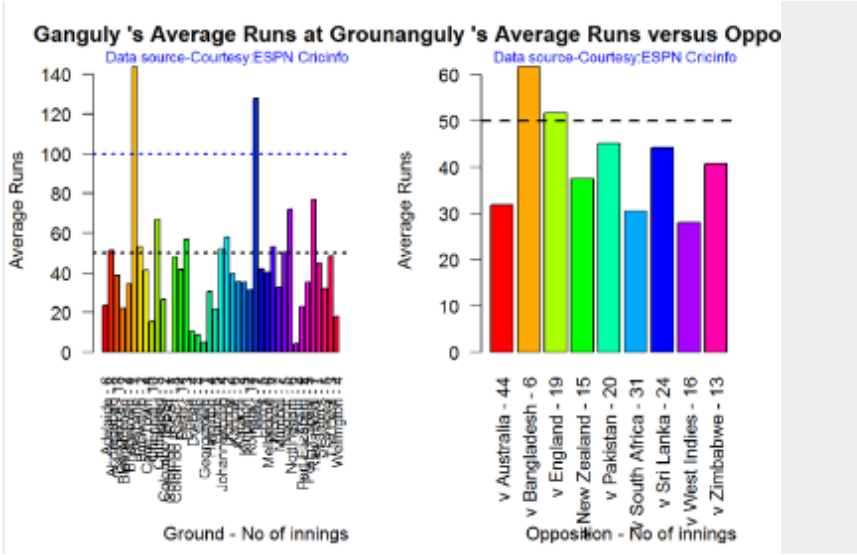
Dravid has above average against England, Bangladesh, New Zealand, Pakistan, West Indies and Zimbabwe

```
par(mfrow=c(1,2))
par(mar=c(4,4,2,2))
batsmanAvgRunsGround("./dravid.csv","Dravid")
batsmanAvgRunsOpposition("./dravid.csv","Dravid")
```



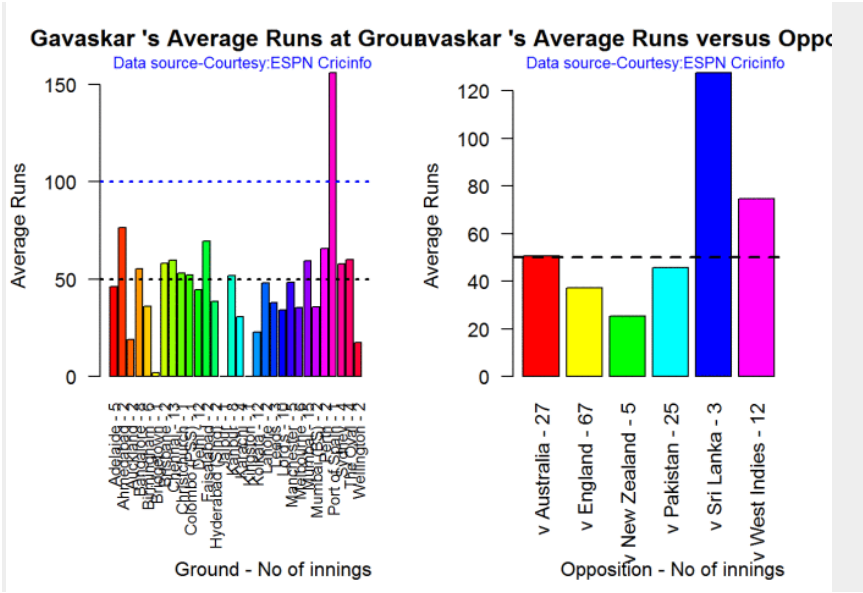
Ganguly has good performance at the Oval, Rawalpindi, Johannesburg and Kandy. Ganguly averages 50 runs against England and Bangladesh.

```
par(mfrow=c(1,2))
par(mar=c(4,4,2,2))
batsmanAvgRunsGround("./ganguly.csv","Ganguly")
batsmanAvgRunsOpposition("./ganguly.csv","Ganguly")
```



The Oval, Sydney, Perth, Melbourne, Brisbane, Manchester are happy hunting grounds for Gavaskar. Gavaskar averages around 50 runs Australia, Pakistan, Sri Lanka, West Indies.

```
par(mfrow=c(1,2))
par(mar=c(4,4,2,2))
batsmanAvgRunsGround("./gavaskar.csv","Gavaskar")
batsmanAvgRunsOpposition("./gavaskar.csv","Gavaskar")
```



1.2.13. Key findings

Here are some key conclusions

1. Tendulkar has the highest average among the 4. He is followed by Dravid, Gavaskar and Ganguly.
2. Tendulkar's predicted performance for a given number of Balls Faced and Minutes at Crease is superior to the rest
3. Dravid averages above 50 against 6 countries
4. West Indies and Australia are Gavaskar's favorite batting grounds
5. Ganguly has a very good Mean Strike Rate for the range 40-80 and Tendulkar from 100-180
6. In home and overseas performance, Tendulkar is the best. Dravid and Gavaskar also have good performance overseas.
7. Dravid and Gavaskar probably retired a year or two earlier while Tendulkar and Ganguly's time was clearly up

1.1.1. Final thoughts

Tendulkar is clearly the greatest batsman India has produced as he leads in almost all aspects of batting – number of centuries, strike rate,

predicted runs and home and overseas performance. Dravid follows Tendulkar with 48 centuries, consistent performance home and overseas and a career that was still green. Gavaskar has fewer matches than rest but his performance overseas is very good in those helmetless times. Finally we have Ganguly.

Dravid and Gavaskar had a few more years of great batting while Tendulkar and Ganguly's career was on a decline.

Note: It is really not fair to include Gavaskar in the analysis as he played in a different era when helmets were not used, even against the fiery pace of Thomson, Lillie, Roberts, Holding etc. In addition Gavaskar did not play against some of the newer countries like Bangladesh and Zimbabwe where he could have amassed runs. Yet I wanted to include him and his performance is clearly excellent

1.2. cricketr digs the Ashes!

1.2.1. Introduction

In some circles the Ashes is considered the 'mother of all cricketing battles'. But, being a staunch supporter of all things Indian, cricket or otherwise, I have to say that the Ashes pales in comparison against a India-Pakistan match. After all, what are a few frowns and raised eyebrows at the Ashes in comparison to the seething emotions and reckless exuberance of Indian fans.

Anyway, the Ashes are an interesting duel and I have decided to do some cricketing analysis using my R package **cricketr**. For this analysis I have chosen the top 2 batsman and top 2 bowlers from both the Australian and English sides.

Batsmen

1. Steven Smith (Aus) – Innings – 58 , Ave: 58.52, Strike Rate: 55.90
2. David Warner (Aus) – Innings – 76, Ave: 46.86, Strike Rate: 73.88
3. Alistair Cook (Eng) – Innings – 208 , Ave: 46.62, Strike Rate: 46.33
4. J E Root (Eng) – Innings – 53, Ave: 54.02, Strike Rate: 51.30

Bowlers

1. Mitchell Johnson (Aus) – Innings-131, Wickets – 299, Econ Rate : 3.28
2. Peter Siddle (Aus) – Innings – 104 , Wickets- 192, Econ Rate : 2.95
3. James Anderson (Eng) – Innings – 199 , Wickets- 406, Econ Rate : 3.05
4. Stuart Broad (Eng) – Innings – 148 , Wickets- 296, Econ Rate : 3.08

It is my opinion if any 2 of the 4 in either team click then they will be able to swing the match in favor of their team.

I have interspersed the plots with a few comments. Feel free to draw your conclusions!

The package can be installed directly from CRAN

```
if (!require("cricketr")){
   install.packages("cricketr",lib = "c:/test")
}
library(cricketr)
```

or from Github

```
library(devtools)
install_github("tvganesh/cricketr")
library(cricketr)
```

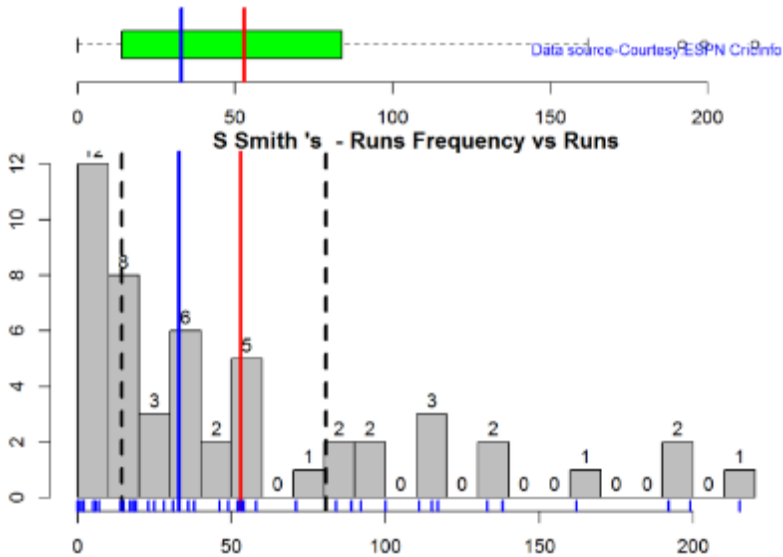
1.2.2. Analyses of Batsmen

The following plots gives the analysis of the 2 Australian and 2 English batsmen. It must be kept in mind that Cooks has more innings than all the rest put together. Smith has the best average, and Warner has the best strike rate

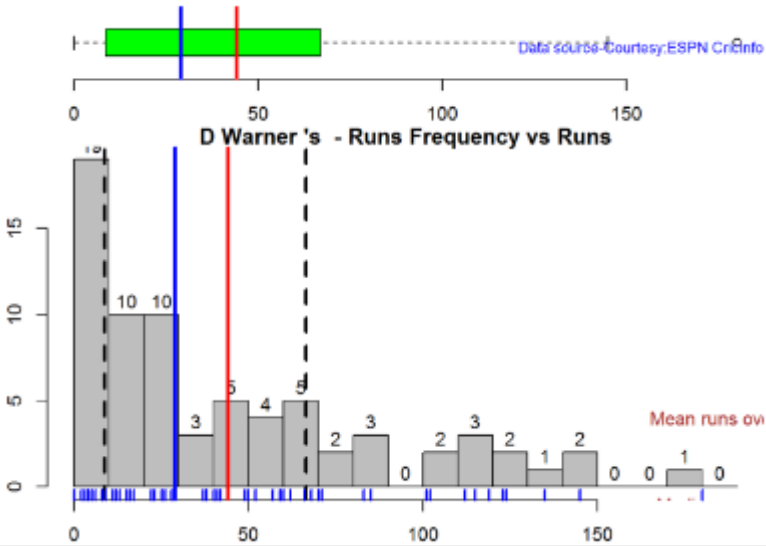
1.2.3. Box Histogram Plot

This plot shows a combined boxplot of the Runs ranges and a histogram of the Runs Frequency

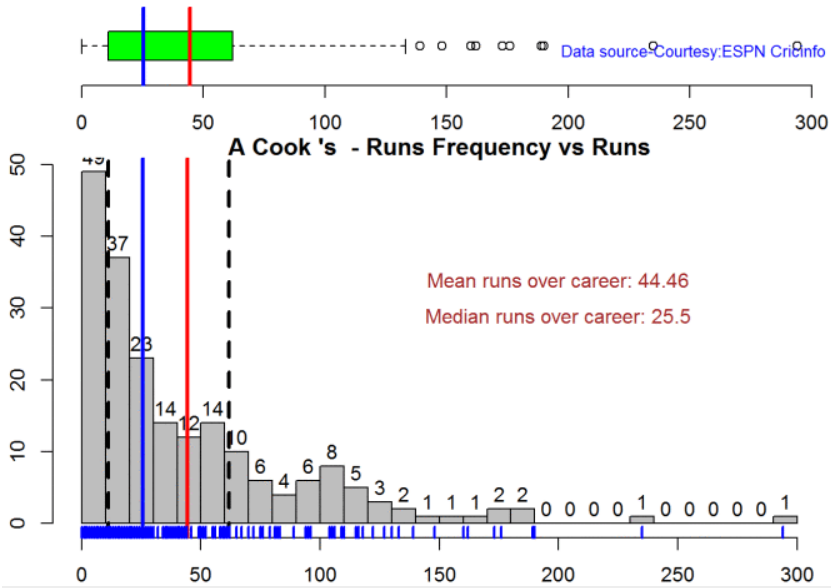
```
batsmanPerfBoxHist("./smith.csv",,"S Smith")
```



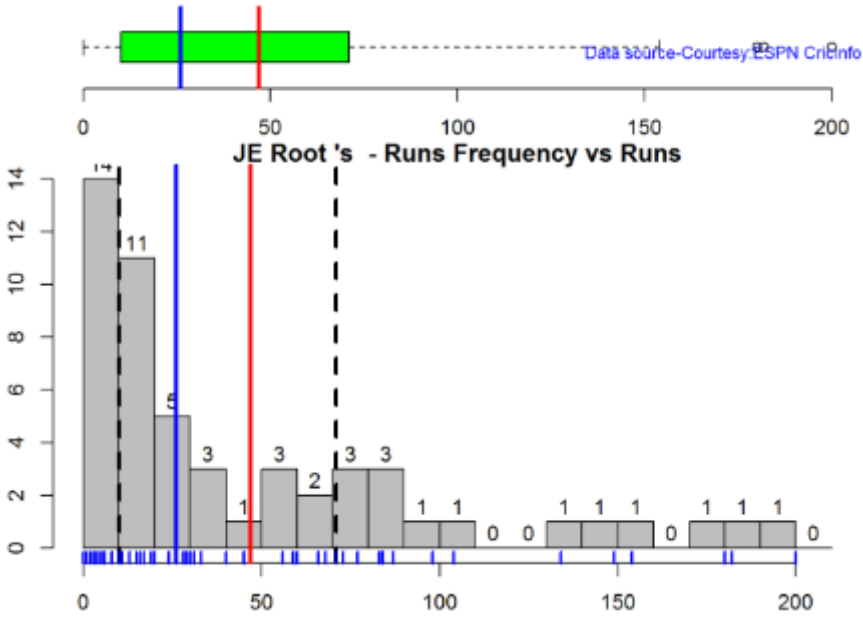

```
batsmanPerfBoxHist("./warner.csv",,"D Warner")
```


Data source:Courtesy:ESPN CricInfo
0
50
100
150
D Warner 's - Runs Frequency vs Runs
0
5
10
15
10 10
3
5
4
5
2
3
0
2
3
2
1
2
0
0
1
0
Mean runs ov
0
50
100
150

```
batsmanPerfBoxHist("./cook.csv","A Cook")
```



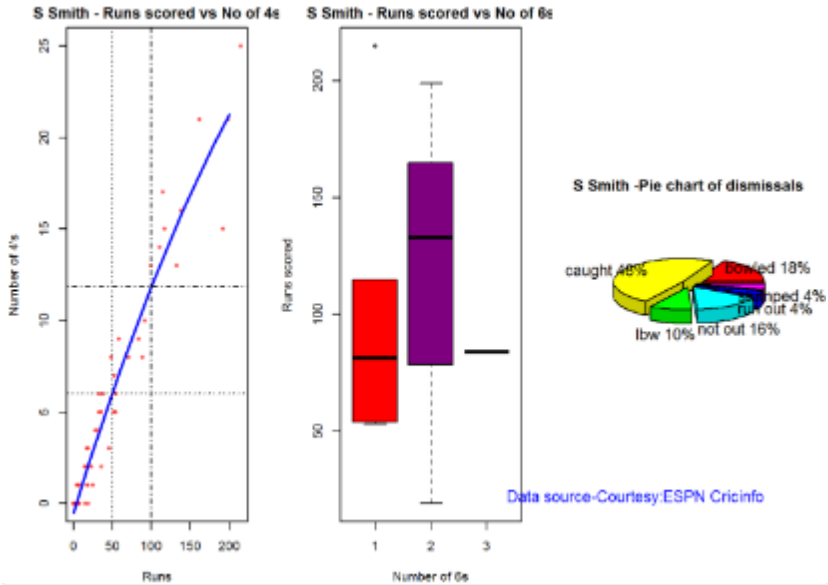

```
batsmanPerfBoxHist("./root.csv","JE Root")
```



1.2.4. Plot of 4s, 6s and the type of dismissals

A. Steven Smith

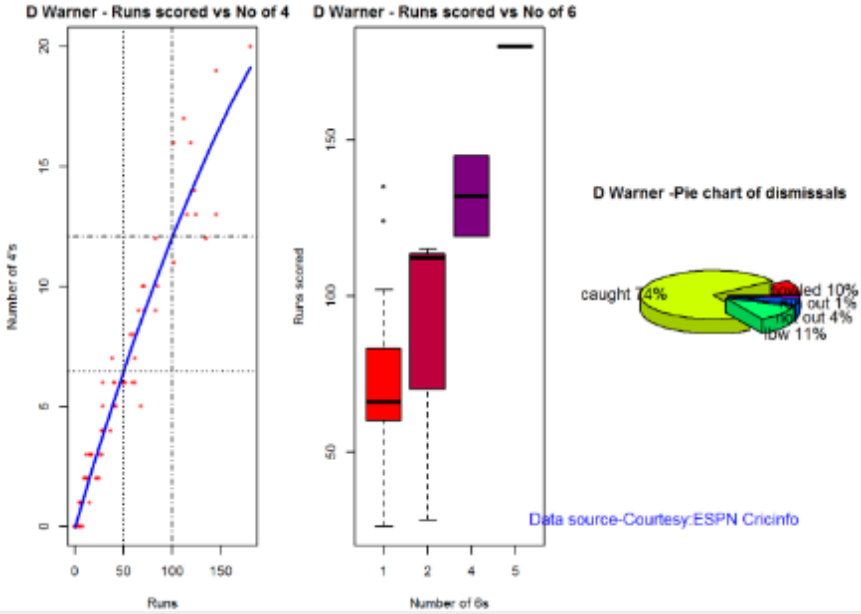
```
par(mfrow=c(1,3))
par(mar=c(4,4,2,2))
batsman4s("./smith.csv","S Smith")
batsman6s("./smith.csv","S Smith")
batsmanDismissals("./smith.csv","S Smith")
```




```
dev.off()
## null device
##           1
```

B. David Warner

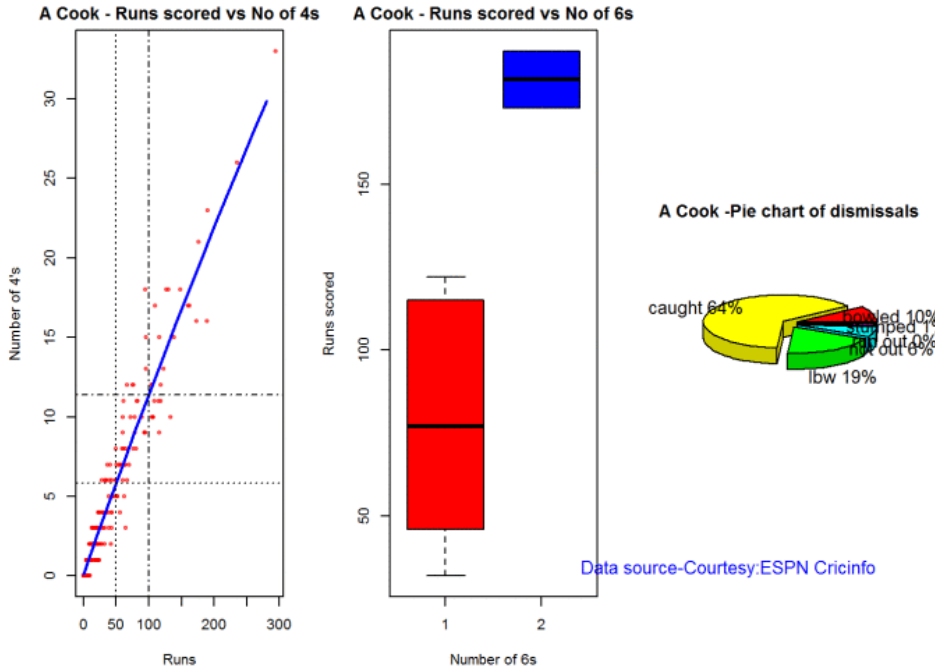
```
par(mfrow=c(1,3))
par(mar=c(4,4,2,2))
batsman4s("./warner.csv","D Warner")
batsman6s("./warner.csv","D Warner")
batsmanDismissals("./warner.csv","D Warner")
```




```
dev.off()
## null device
##           1
```

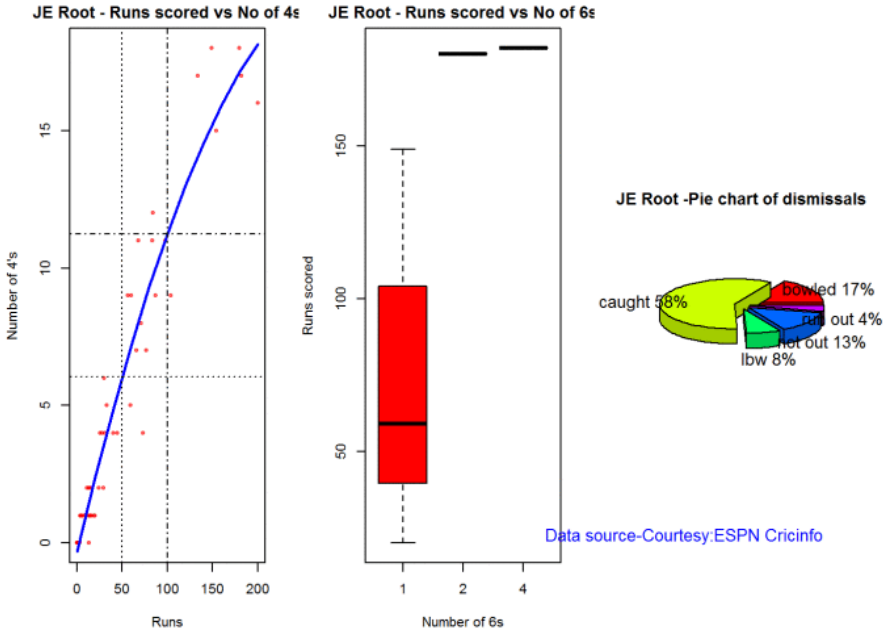
C. Alistair Cook

```
par(mfrow=c(1,3))
par(mar=c(4,4,2,2))
batsman4s("./cook.csv","A Cook")
batsman6s("./cook.csv","A Cook")
batsmanDismissals("./cook.csv","A Cook")
```



D. J E Root

```
par(mfrow=c(1,3))
par(mar=c(4,4,2,2))
batsman4s("./root.csv","JE Root")
batsman6s("./root.csv","JE Root")
batsmanDismissals("./root.csv","JE Root")
```



1.2.5. Relative Mean Strike Rate

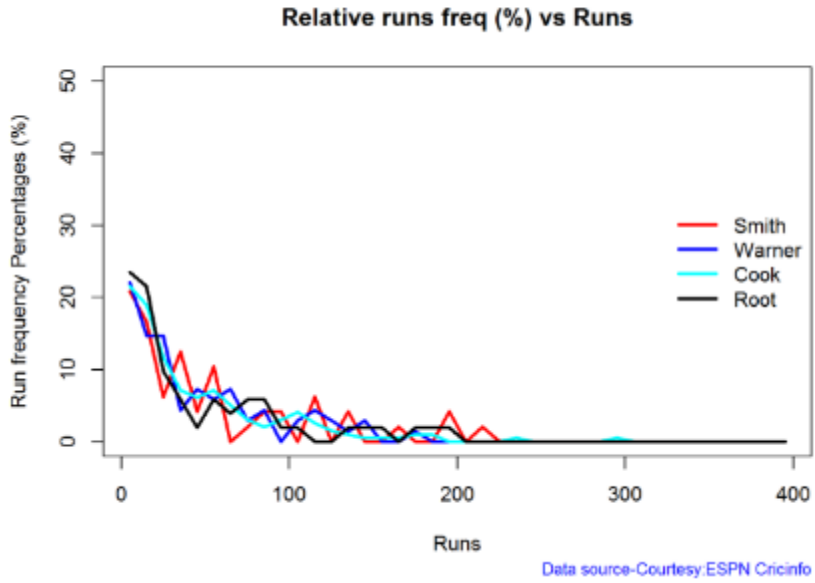
In this first plot I plot the Mean Strike Rate of the batsmen. It can be Warner's has the best strike rate (hit outside the plot!) followed by Smith in the range 20-100. Root has a good strike rate above hundred runs. Cook maintains a good strike rate.

```
par(mar=c(4,4,2,2))
frames <- list("./smith.csv","./warner.csv","cook.csv","root.csv")
names <- list("Smith","Warner","Cook","Root")
relativeBatsmanSR(frames,names)
```

1.2.6. Relative Runs Frequency Percentage

The plot below show the percentage contribution in each 10 runs bucket over the entire career. It can be seen that Smith pops up above the rest with remarkable regularity. Cook is consistent over the entire range.

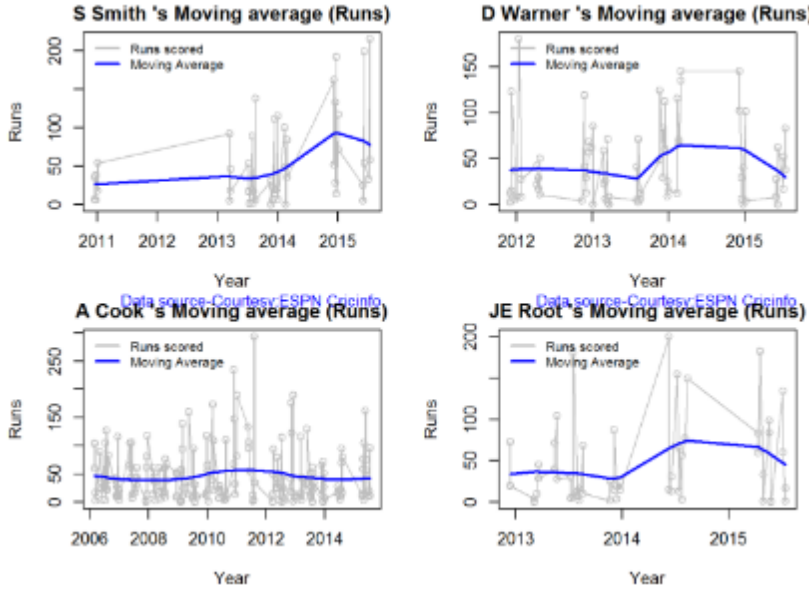
```
frames <- list("./smith.csv","./warner.csv","cook.csv","root.csv")
names <- list("Smith","Warner","Cook","Root")
relativeRunsFreqPerf(frames,names)
```



1.2.7. Moving Average of runs over career

The moving average for the 4 batsmen indicate the following 1. S Smith is the most promising. There is a marked spike in Performance. Cook maintains a steady pace and is consistent over the years averaging 50 over the years.

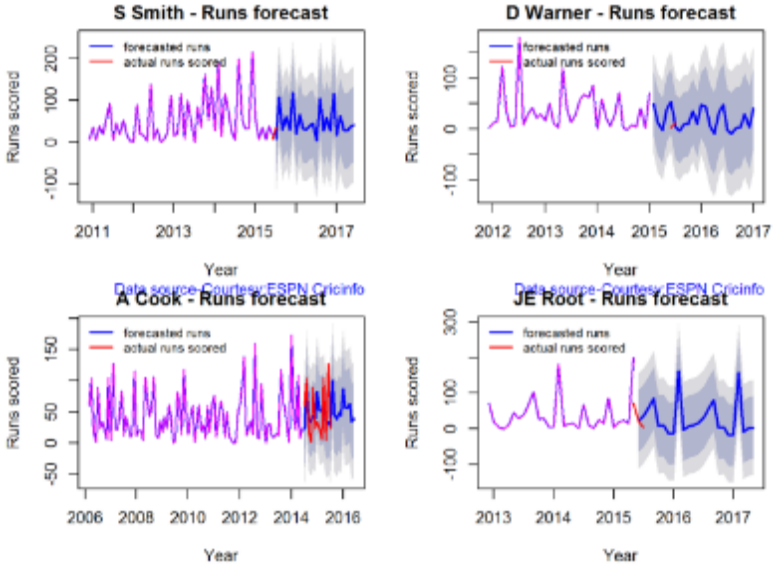
```
par(mfrow=c(2,2))
par(mar=c(4,4,2,2))
batsmanMovingAverage("./smith.csv","S Smith")
batsmanMovingAverage("./warner.csv","D Warner")
batsmanMovingAverage("./cook.csv","A Cook")
batsmanMovingAverage("./root.csv","JE Root")
```



1.2.8. Runs forecast

The forecast for the batsman is shown below. As before Cooks' performance is really consistent across the years and the forecast is good for the years ahead. In Cook's case it can be seen that the forecasted and actual runs are reasonably accurate

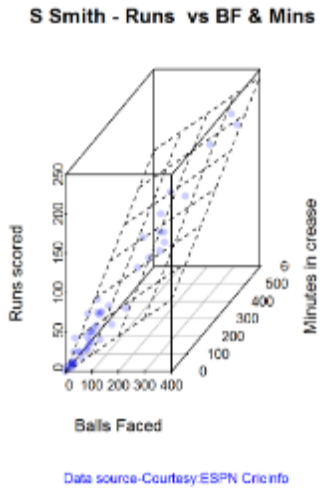
```
par(mfrow=c(2,2))
par(mar=c(4,4,2,2))
batsmanPerfForecast("./smith.csv","S Smith")
batsmanPerfForecast("./warner.csv","D Warner")
batsmanPerfForecast("./cook.csv","A Cook")
## Warning in HoltWinters(ts.train): optimization difficulties:
ERROR:
## ABNORMAL_TERMINATION_IN_LNSRCH
batsmanPerfForecast("./root.csv","JE Root")
```

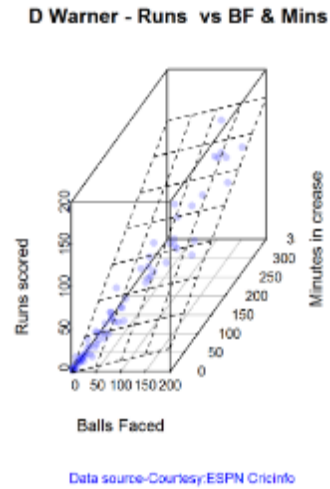


1.2.9. 3D plot of Runs vs Balls Faced and Minutes at Crease

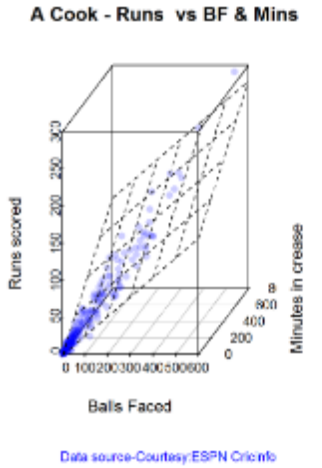
The plot is a scatter plot of Runs vs Balls faced and Minutes at Crease. A prediction plane is fitted

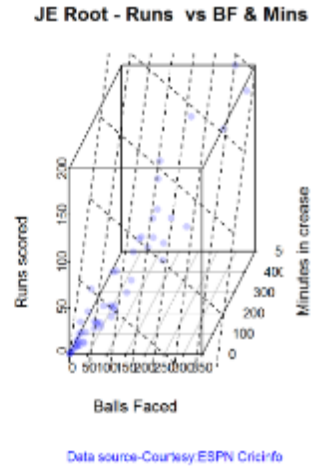
```
par(mfrow=c(1,2))
par(mar=c(4,4,2,2))
battingPerf3d("./smith.csv","S Smith")
battingPerf3d("./warner.csv","D Warner")
```






```
par(mfrow=c(1,2))
par(mar=c(4,4,2,2))
battingPerf3d("./cook.csv","A Cook")
battingPerf3d("./root.csv","JE Root")
```





1.2.10. Predicting Runs given Balls Faced and Minutes at Crease

A multi-variate regression plane is fitted between Runs and Balls faced +Minutes at crease.

```
BF <- seq( 10, 400,length=15)
Mins <- seq(30,600,length=15)
```

```
newDF <- data.frame(BF,Mins)
smith <- batsmanRunsPredict("./smith.csv","S Smith",newdataframe=newDF)
warner <- batsmanRunsPredict("./warner.csv","D Warner",newdataframe=newDF)
cook <- batsmanRunsPredict("./cook.csv","A Cook",newdataframe=newDF)
root <- batsmanRunsPredict("./root.csv","JE Root",newdataframe=newDF)
```

The fitted model is then used to predict the runs that the batsmen will score for a given Balls faced and Minutes at crease. It can be seen that Warner sets a searing pace in the predicted runs for a given Balls Faced and Minutes at crease while Smith and Root are neck to neck in the predicted runs

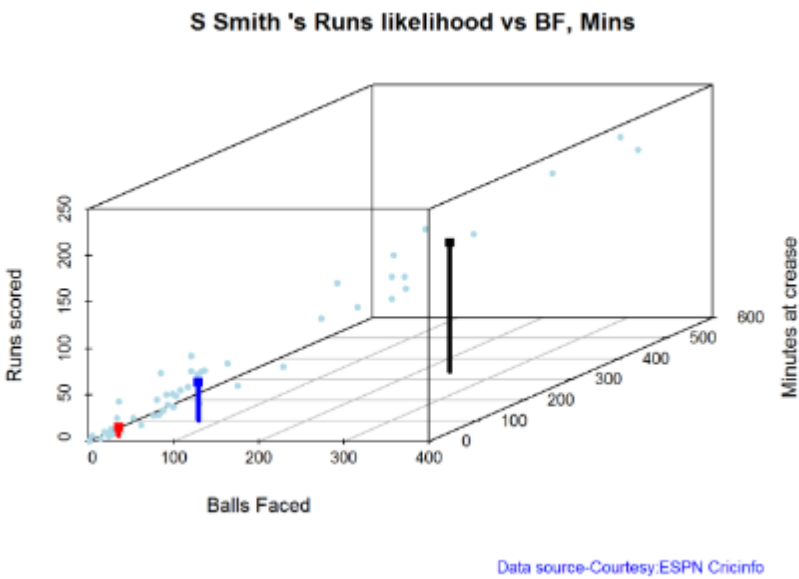
```
batsmen <-
cbind(round(smith$Runs),round(warner$Runs),round(cook$Runs),round(root$Runs))
colnames(batsmen) <- c("Smith","Warner","Cook","Root")
newDF <- data.frame(round(newDF$BF),round(newDF$Mins))
colnames(newDF) <- c("BallsFaced","MinsAtCrease")
predictedRuns <- cbind(newDF,batsmen)
predictedRuns
##    BallsFaced MinsAtCrease Smith Warner Cook Root
## 1          10           30     9     12    6    9
## 2          38           71    25     33   20   25
## 3          66          111    42     53   33   42
## 4          94          152    58     73   47   59
## 5         121          193    75     93   60   75
## 6         149          234    91    114   74   92
## 7         177          274   108    134   88  109
## 8         205          315   124    154  101  125
## 9         233          356   141    174  115  142
## 10        261          396   158    195  128  159
## 11        289          437   174    215  142  175
## 12        316          478   191    235  155  192
## 13        344          519   207    255  169  208
## 14        372          559   224    276  182  225
## 15        400          600   240    296  196  242
```

1.2.11. Highest runs likelihood

The plots below the runs likelihood of batsman. This uses K-Means. It can be seen Smith has the best likelihood around 40% of scoring around 41 runs, followed by Root who has 28.3% likelihood of scoring around 81 runs

A. Steven Smith

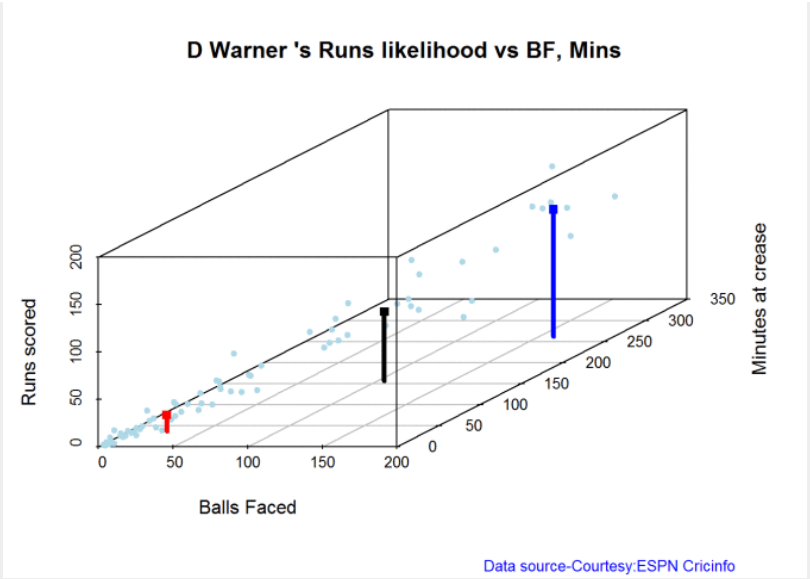
```
batsmanRunsLikelihood("./smith.csv","S Smith")
```




```
## Summary of  S Smith 's runs scoring likelihood
## ************************************************
##
## There is a 40 % likelihood that S Smith  will make  41 Runs in
73 balls over 101  Minutes
## There is a 36 % likelihood that S Smith  will make  9 Runs in
21 balls over  27  Minutes
## There is a 24 % likelihood that S Smith  will make  139 Runs
in  237 balls over 338  Minutes
```

B. David Warner

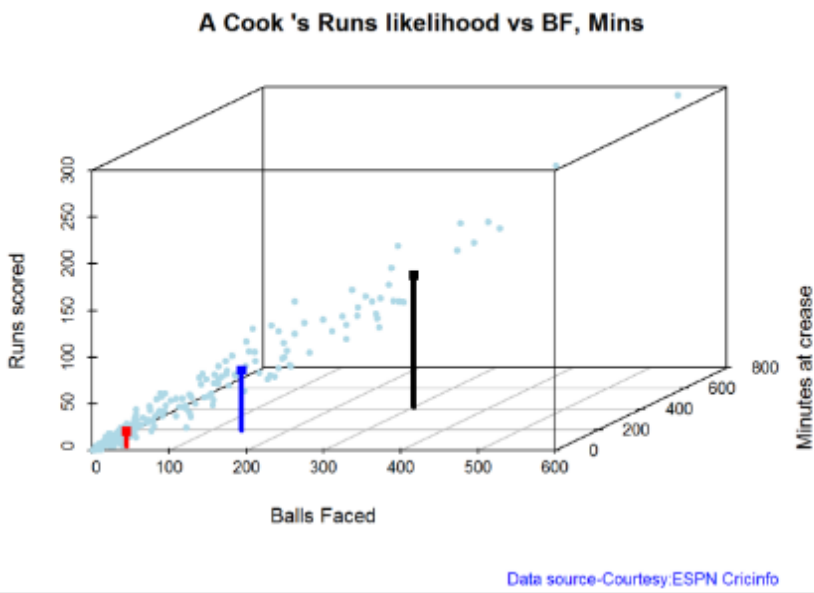
```
batsmanRunsLikelihood("./warner.csv","D Warner")
```




```
## Summary of  D Warner 's runs scoring likelihood
## ************************************************
##
## There is a 11.11 % likelihood that D Warner  will make  134
Runs in  159 balls over 263  Minutes
## There is a 63.89 % likelihood that D Warner  will make  17
Runs in  25 balls over  37  Minutes
## There is a 25 % likelihood that D Warner  will make  73 Runs
in  105 balls over 156  Minutes
```

C. Alastair Cook

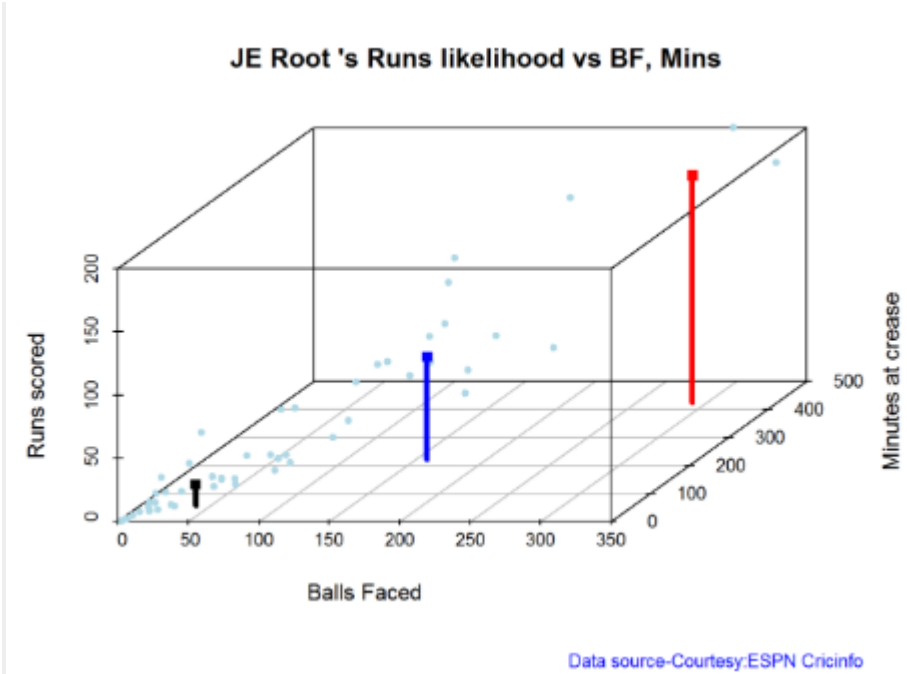
```
batsmanRunsLikelihood("./cook.csv","A Cook")
```




```
## Summary of  A Cook 's runs scoring likelihood
## ************************************************
##
## There is a 27.72 % likelihood that A Cook  will make  64 Runs
in  140 balls over 195  Minutes
## There is a 59.9 % likelihood that A Cook  will make  15 Runs
in  32 balls over  46  Minutes
## There is a 12.38 % likelihood that A Cook  will make  141 Runs
in  300 balls over 420  Minutes
```

D. J E Root

```
batsmanRunsLikelihood("./root.csv","JE Root")
```

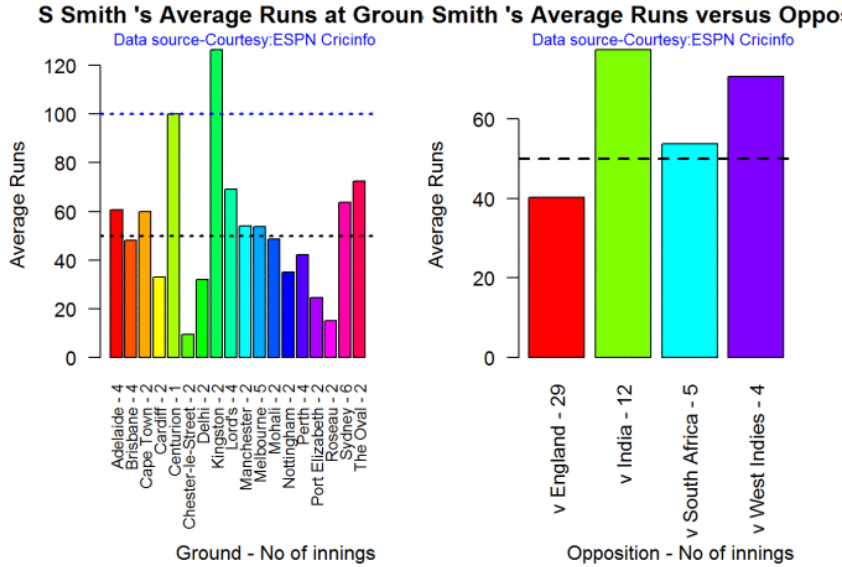



```
## Summary of  JE Root 's runs scoring likelihood
## ************************************************
##
## There is a 28.3 % likelihood that JE Root  will make  81 Runs
in  158 balls over 223  Minutes
## There is a 7.55 % likelihood that JE Root  will make  179 Runs
in  290 balls over  425  Minutes
## There is a 64.15 % likelihood that JE Root  will make  16 Runs
in  39 balls over 59  Minutes
```

1.2.12. Average runs at ground and against opposition

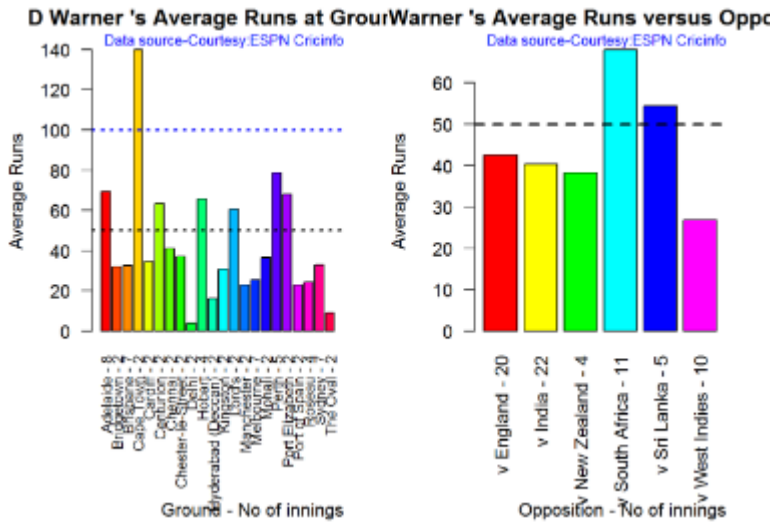
A. Steven Smith

```
par(mfrow=c(1,2))
par(mar=c(4,4,2,2))
batsmanAvgRunsGround("./smith.csv","S Smith")
batsmanAvgRunsOpposition("./smith.csv","S Smith")
```



B. David Warner

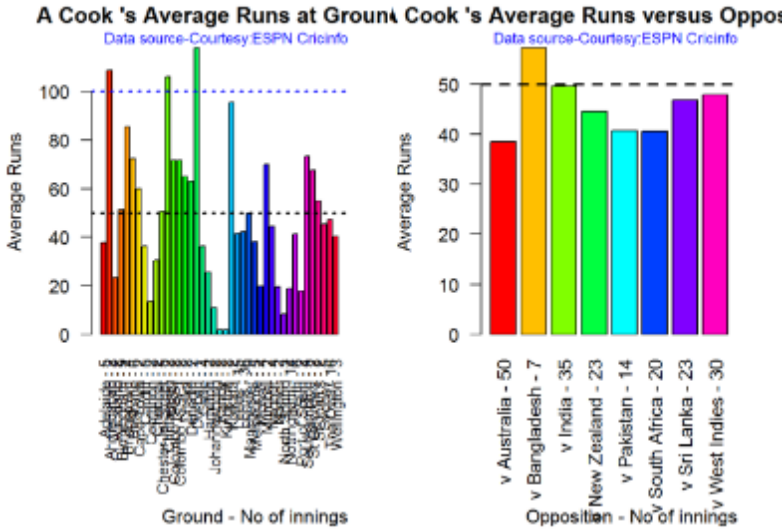
```
par(mfrow=c(1,2))
par(mar=c(4,4,2,2))
batsmanAvgRunsGround("./warner.csv","D Warner")
batsmanAvgRunsOpposition("./warner.csv","D Warner")
```



C. Alistair Cook

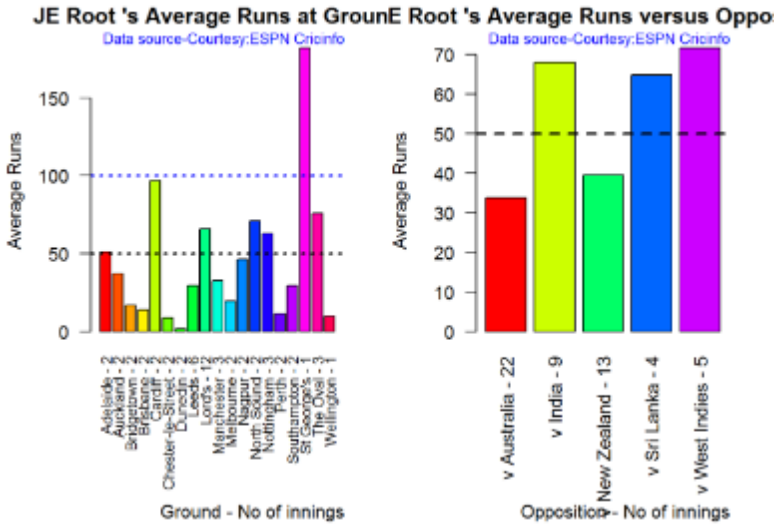
```
par(mfrow=c(1,2))
par(mar=c(4,4,2,2))
batsmanAvgRunsGround("./cook.csv","A Cook")
```

```
batsmanAvgRunsOpposition("./cook.csv","A Cook")
```



D. J E Root

```
par(mfrow=c(1,2))
par(mar=c(4,4,2,2))
batsmanAvgRunsGround("./root.csv","JE Root")
batsmanAvgRunsOpposition("./root.csv","JE Root")
```



1.2.13. Analysis of bowlers

1. Mitchell Johnson (Aus) – Innings-131, Wickets – 299, Econ Rate : 3.28

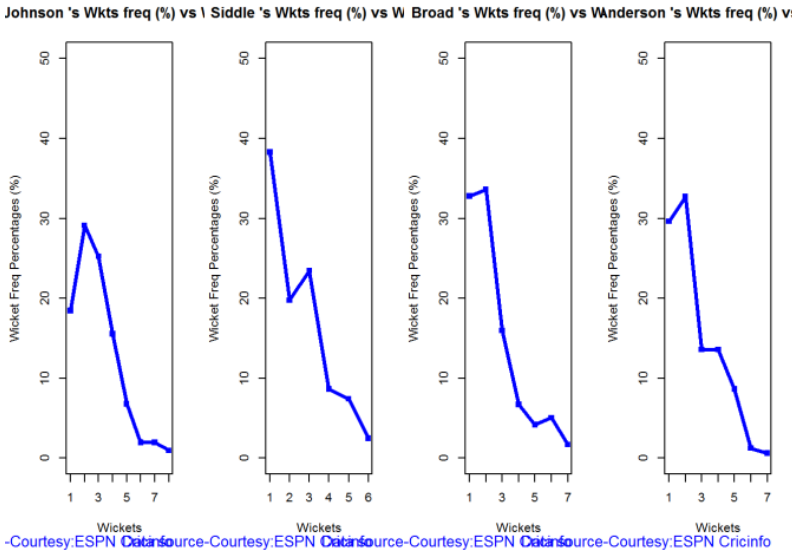
2. Peter Siddle (Aus) – Innings – 104 , Wickets- 192, Econ Rate : 2.95
3. James Anderson (Eng) – Innings – 199 , Wickets- 406, Econ Rate : 3.05
4. Stuart Broad (Eng) – Innings – 148 , Wickets- 296, Econ Rate : 3.08

Anderson has the highest number of inning and wickets followed closely by Broad and Mitchell who are in a neck to neck race with respect to wickets. Johnson is on the more expensive side though. Siddle has fewer innings but a good economy rate.

1.2.14. Wicket Frequency percentage

This plot gives the percentage of wickets for each wickets (1,2,3…etc.)

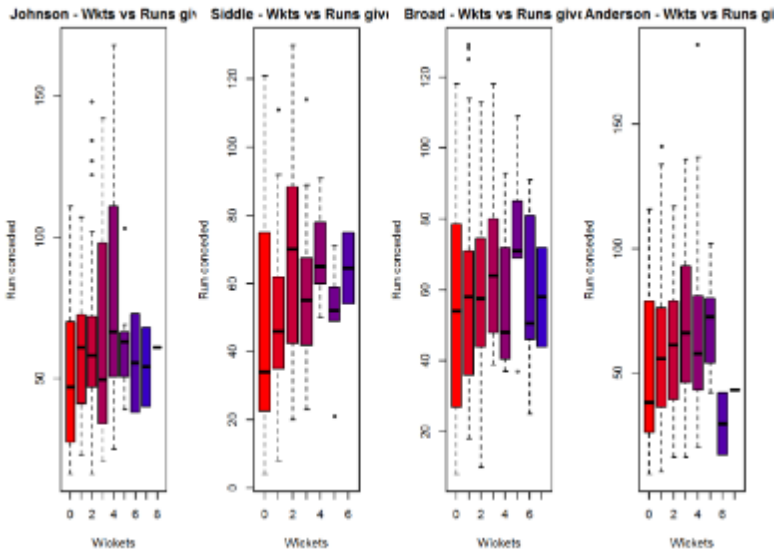
```
par(mfrow=c(1,4))
par(mar=c(4,4,2,2))
bowlerWktsFreqPercent("./johnson.csv","Johnson")
bowlerWktsFreqPercent("./siddle.csv","Siddle")
bowlerWktsFreqPercent("./broad.csv","Broad")
bowlerWktsFreqPercent("./anderson.csv","Anderson")
```



1.2.15. Wickets Runs plot

The plot below gives a boxplot of the runs ranges for each of the wickets taken by the bowlers

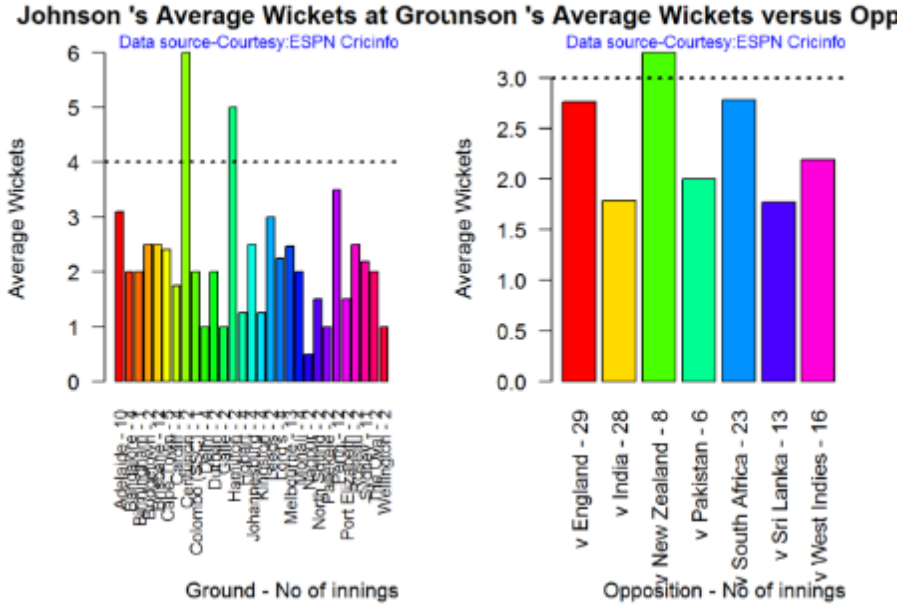
```
par(mfrow=c(1,4))
par(mar=c(4,4,2,2))
bowlerWktsRunsPlot("./johnson.csv","Johnson")
bowlerWktsRunsPlot("./siddle.csv","Siddle")
bowlerWktsRunsPlot("./broad.csv","Broad")
bowlerWktsRunsPlot("./anderson.csv","Anderson")
```



1.2.16. Average wickets in different grounds and opposition

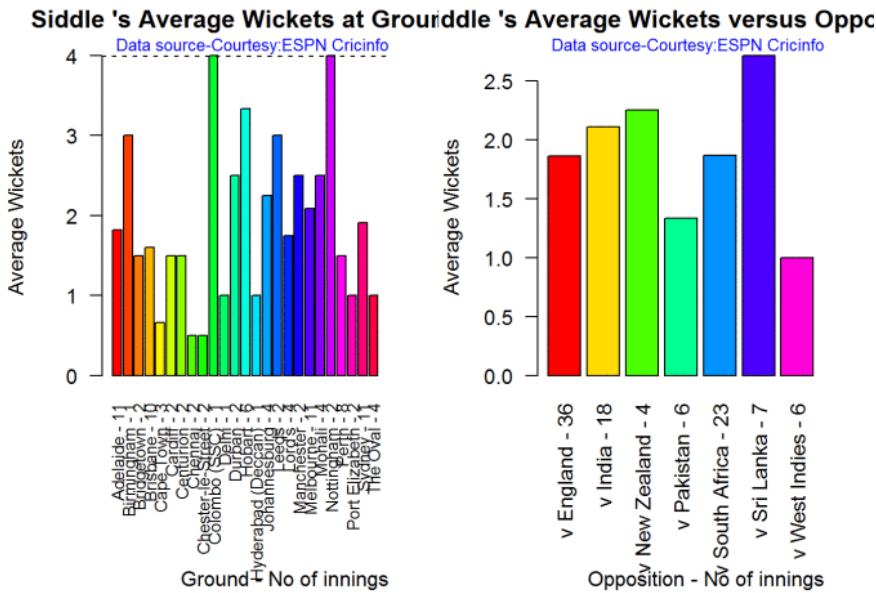
A. Mitchell Johnson

```
par(mfrow=c(1,2))
par(mar=c(4,4,2,2))
bowlerAvgWktsGround("./johnson.csv","Johnson")
bowlerAvgWktsOpposition("./johnson.csv","Johnson")
```



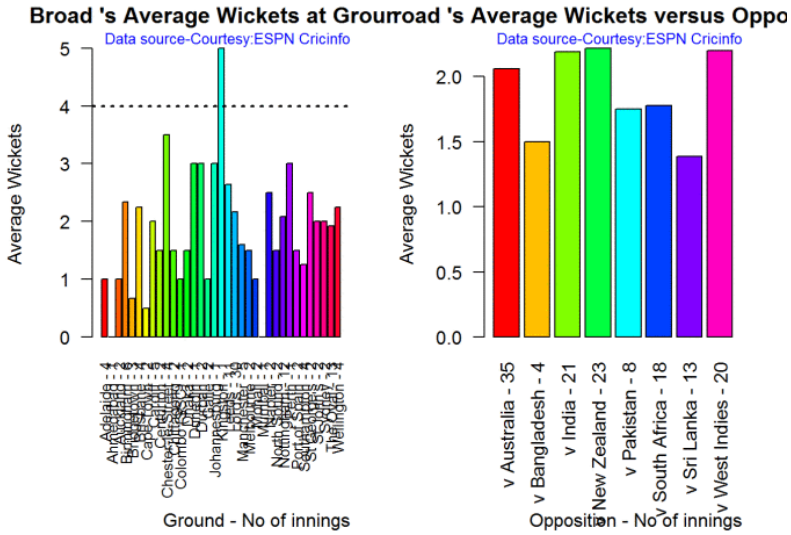
B. Peter Siddle

```
par(mfrow=c(1,2))
par(mar=c(4,4,2,2))
bowlerAvgWktsGround("./siddle.csv","Siddle")
bowlerAvgWktsOpposition("./siddle.csv","Siddle")
```



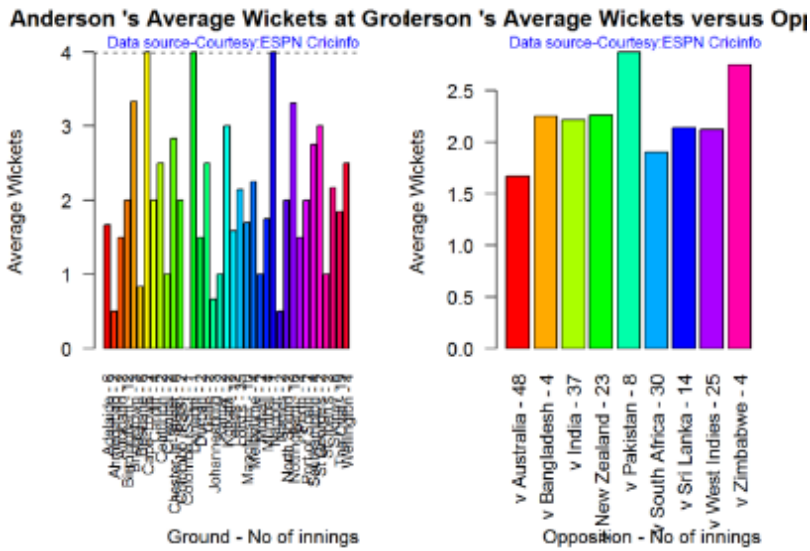
C. Stuart Broad

```
par(mfrow=c(1,2))
par(mar=c(4,4,2,2))
bowlerAvgWktsGround("./broad.csv","Broad")
bowlerAvgWktsOpposition("./broad.csv","Broad")
```



D. James Anderson

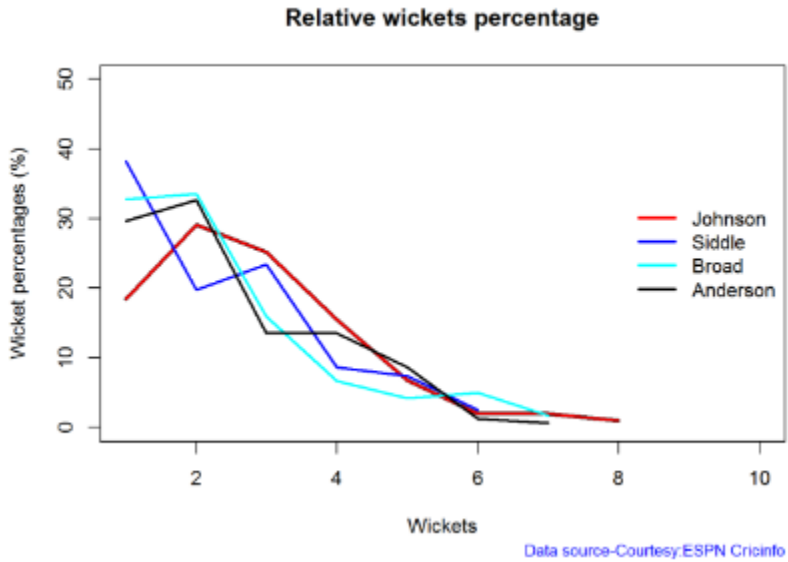
```
par(mfrow=c(1,2))
par(mar=c(4,4,2,2))
bowlerAvgWktsGround("./anderson.csv","Anderson")
bowlerAvgWktsOpposition("./anderson.csv","Anderson")
```



1.2.17. Relative bowling performance

The plot below shows that Mitchell Johnson is the most effective bowler among the lot with a higher wickets in the 3-6 wicket range. Broad and Anderson seem to perform well in 2 wickets in comparison to Siddle but in 3 wickets Siddle is better than Broad and Anderson.

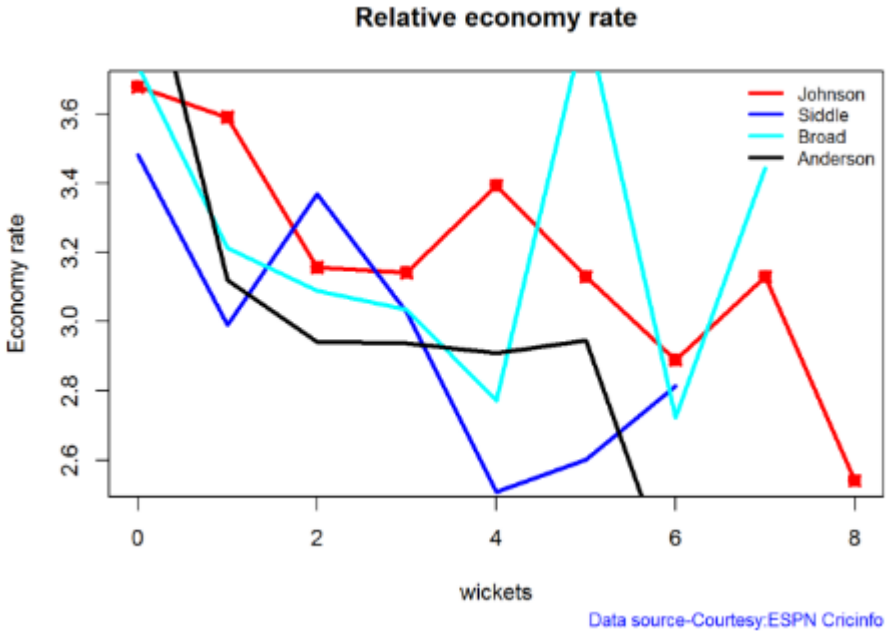
```
frames <- list("./johnson.csv","./siddle.csv","broad.csv","anderson.csv")
names <- list("Johnson","Siddle","Broad","Anderson")
relativeBowlingPerf(frames,names)
```



1.2.18. Relative Economy Rate against wickets taken

Anderson followed by Siddle has the best economy rates. Johnson is fairly expensive in the 4-8 wicket range.

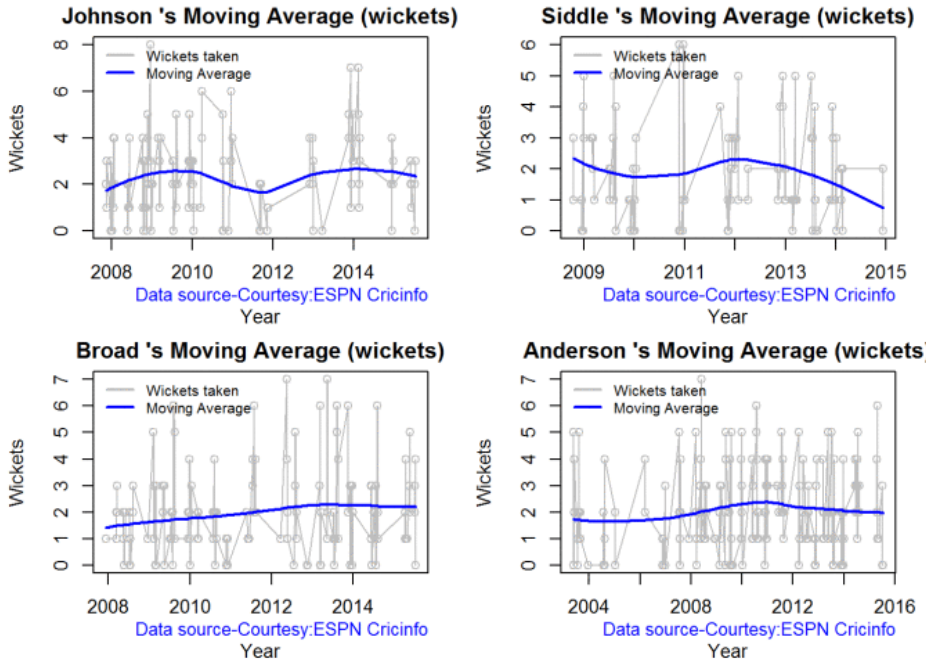
```
frames <- list("./johnson.csv","./siddle.csv","broad.csv","anderson.csv")
names <- list("Johnson","Siddle","Broad","Anderson")
relativeBowlingER(frames,names)
```



1.2.19. Moving average of wickets over career

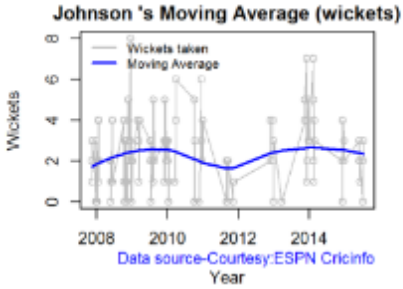
Johnson is on his second peak while Siddle is on the decline with respect to bowling. Broad and Anderson show improving performance over the years.

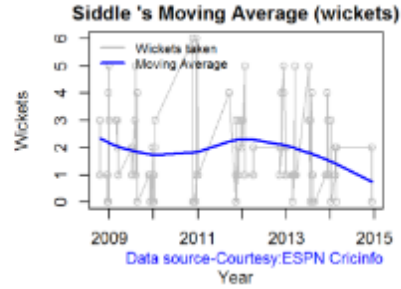
```
par(mfrow=c(2,2))
par(mar=c(4,4,2,2))
bowlerMovingAverage("./johnson.csv","Johnson")
bowlerMovingAverage("./siddle.csv","Siddle")
bowlerMovingAverage("./broad.csv","Broad")
bowlerMovingAverage("./anderson.csv","Anderson")
```

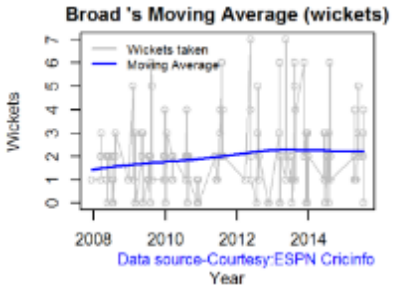


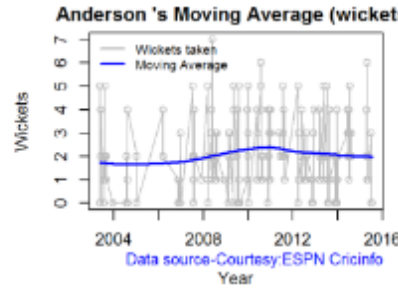
1.2.20. Wickets forecast

```
par(mfrow=c(2,2))
par(mar=c(4,4,2,2))
bowlerPerfForecast("./johnson.csv","Johnson")
bowlerPerfForecast("./siddle.csv","Siddle")
bowlerPerfForecast("./broad.csv","Broad")
bowlerPerfForecast("./anderson.csv","Anderson")
```









1.2.21. Key findings

Here are some key conclusions

1. Cook has the most number of innings and has been extremely consistent in his scores
2. Warner has the best strike rate among the lot followed by Smith and Root
3. The moving average shows a marked improvement over the years for Smith
4. Johnson is the most effective bowler but is fairly expensive
5. Anderson has the best economy rate followed by Siddle
6. Johnson is at his second peak with respect to bowling while Broad and Anderson maintain a steady line and length in their career bowling performance

1.3. cricketr plays the ODIs!

1.3.1. Introduction

In this post my package 'cricketr' takes a swing at One Day Internationals(ODIs). Like test batsman who adapt to ODIs with some innovative strokes, the cricketr package has some additional functions and some modified functions to handle the high strike and economy rates in ODIs. As before I have chosen my top 4 ODI batsmen and top 4 ODI bowlers.

Batsmen

1. Virender Sehwag (Ind)
2. AB Devilliers (SA)
3. Chris Gayle (WI)
4. Glenn Maxwell (Aus)

Bowlers

1. Mitchell Johnson (Aus)
2. Lasith Malinga (SL)
3. Dale Steyn (SA)
4. Tim Southee (NZ)

I have sprinkled the plots with a few of my comments. Feel free to draw your conclusions! The analysis is included below

The profile for Virender Sehwag is 35263. This can be used to get the ODI data for Sehwag. For a batsman the type should be "batting" and for a bowler the type should be "bowling" and the function is getPlayerDataOD()

The package can be installed directly from CRAN

```
if (!require("cricketr")){

   install.packages("cricketr",lib = "c:/test")
```

```
}

library(cricketr)
```

or from Github

```
library(devtools)

install_github("tvganesh/cricketr")

library(cricketr)
```

The One day data for a particular player can be obtained with the getPlayerDataOD() function. To do you will need to go to ESPN CricInfo Player and type in the name of the player for e.g Virender Sehwag, etc. This will bring up a page which have the profile number for the player e.g. for Virendar Sehwag this would behttp://www.espncricinfo.com/india/content/player/35263.html. Hence, Sehwag's profile is 35263. This can be used to get the data for Virender Sehwag as shown below

```
sehwag  <-  getPlayerDataOD(35263,dir="..",file="sehwag.csv",type="batting")
```

Analyses of Batsmen

The following plots gives the analysis of the 4 ODI batsmen

- Virender Sehwag (Ind) – Innings – 245, Runs = 8586, Average=35.05, Strike Rate= 104.33
- AB Devilliers (SA) – Innings – 179, Runs= 7941, Average=53.65, Strike Rate= 99.12
- Chris Gayle (WI) – Innings – 264, Runs= 9221, Average=37.65, Strike Rate= 85.11
- Glenn Maxwell (Aus) – Innings – 45, Runs= 1367, Average=35.02, Strike Rate= 126.69

1.3.2. Plot of 4s, 6s and the scoring rate in ODIs

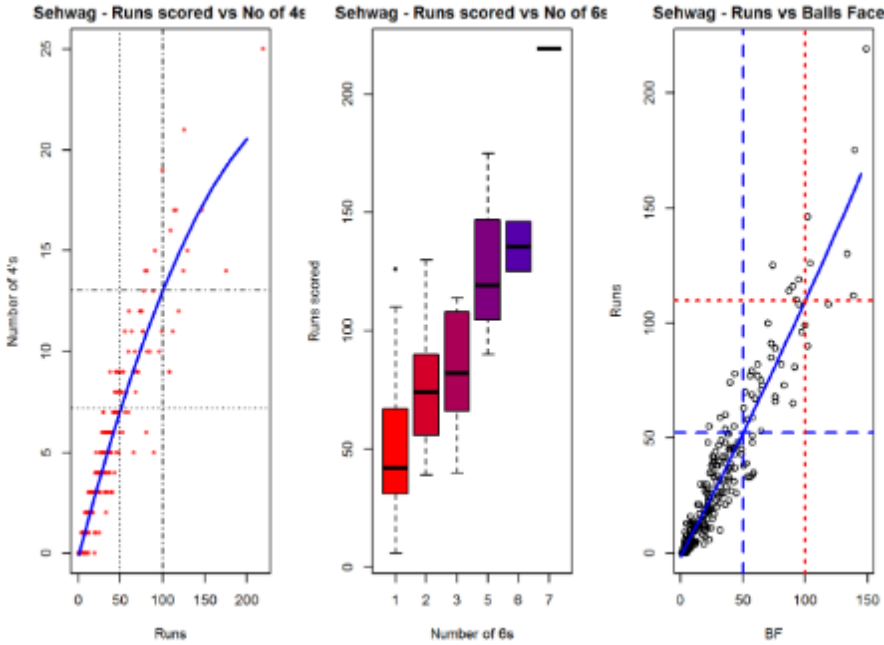
The 3 charts below give the number of

- 4s vs Runs scored
- 6s vs Runs scored
- Balls faced vs Runs scored

A regression line is fitted in each of these plots for each of the ODI batsmen

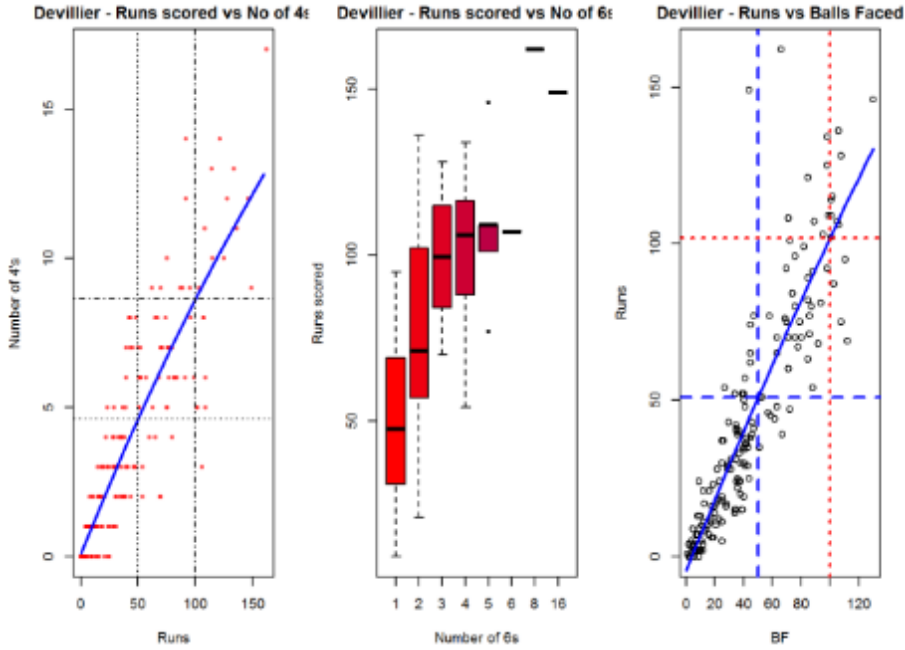
A. Virender Sehwag

```
par(mfrow=c(1,3))

par(mar=c(4,4,2,2))

batsman4s("./sehwag.csv","Sehwag")

batsman6s("./sehwag.csv","Sehwag")

batsmanScoringRateODTT("./sehwag.csv","Sehwag")
```



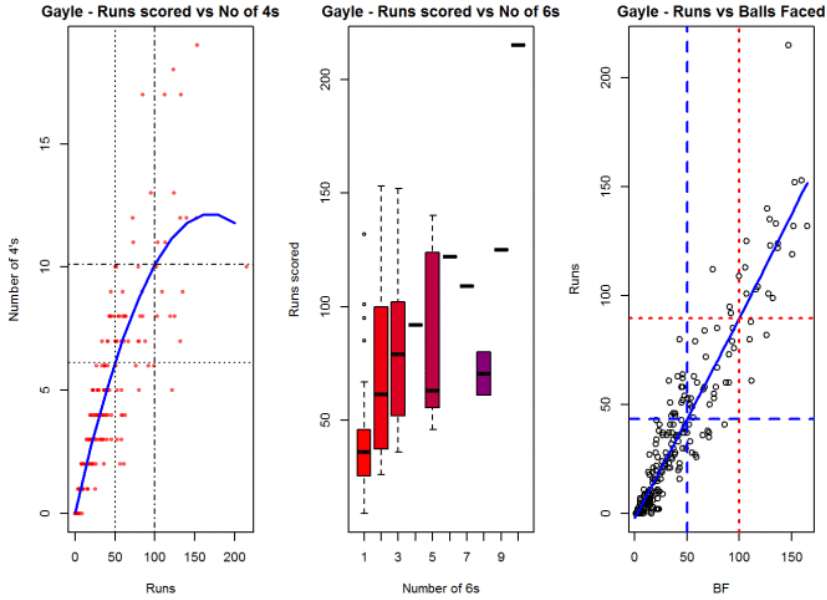
B. AB Devilliers

```
par(mfrow=c(1,3))

par(mar=c(4,4,2,2))

batsman4s("./devilliers.csv","Devillier")

batsman6s("./devilliers.csv","Devillier")

batsmanScoringRateODTT("./devilliers.csv","Devillier")
```



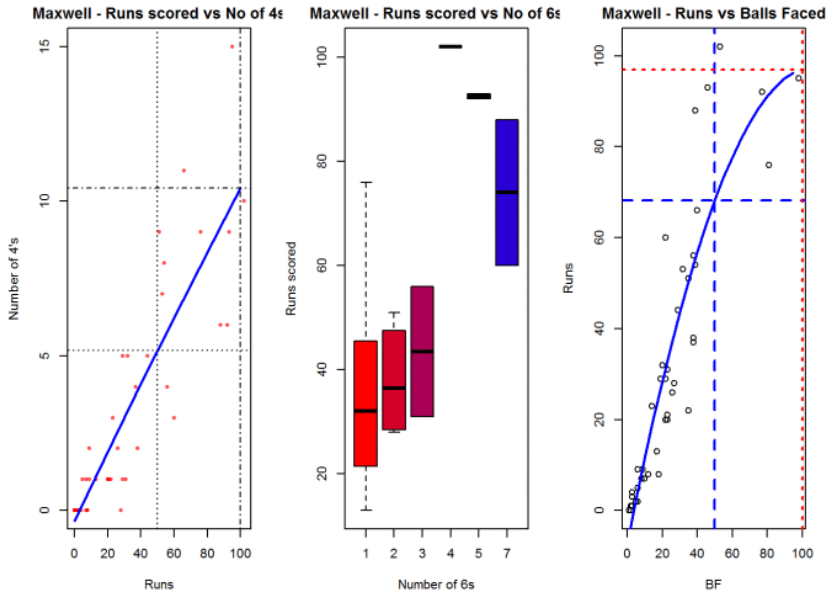
C. Chris Gayle

```
par(mfrow=c(1,3))

par(mar=c(4,4,2,2))

batsman4s("./gayle.csv","Gayle")

batsman6s("./gayle.csv","Gayle")

batsmanScoringRateODTT("./gayle.csv","Gayle")
```



D. Glenn Maxwell

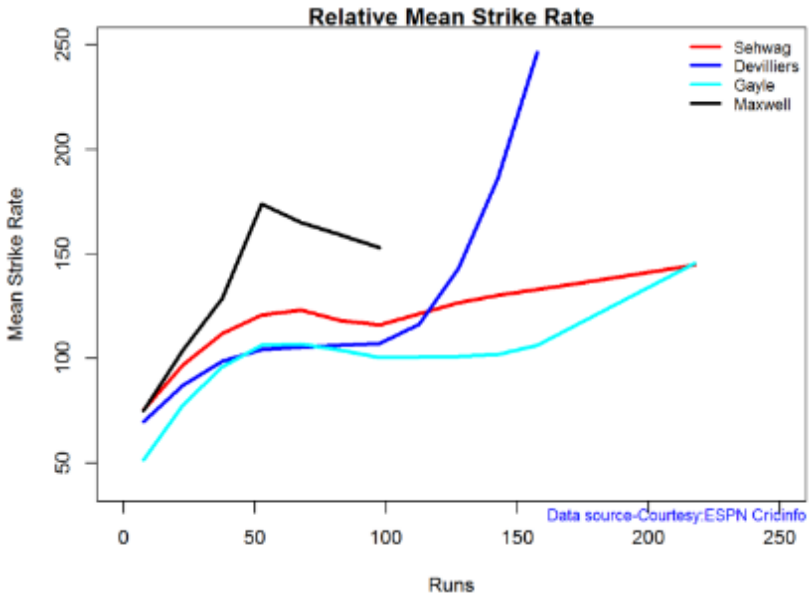
```
par(mfrow=c(1,3))

par(mar=c(4,4,2,2))

batsman4s("./maxwell.csv","Maxwell")

batsman6s("./maxwell.csv","Maxwell")

batsmanScoringRateODTT("./maxwell.csv","Maxwell")
```



1.3.3. Relative Mean Strike Rate

In this first plot I plot the Mean Strike Rate of the batsmen. It can be seen that Maxwell has a awesome strike rate in ODIs. However we need to keep in mind that Maxwell has relatively much fewer (only 45 innings) innings. He is followed by Sehwag who(most innings- 245) also has an excellent strike rate till 100 runs and then we have Devilliers who roars ahead. This is also seen in the overall strike rate in above

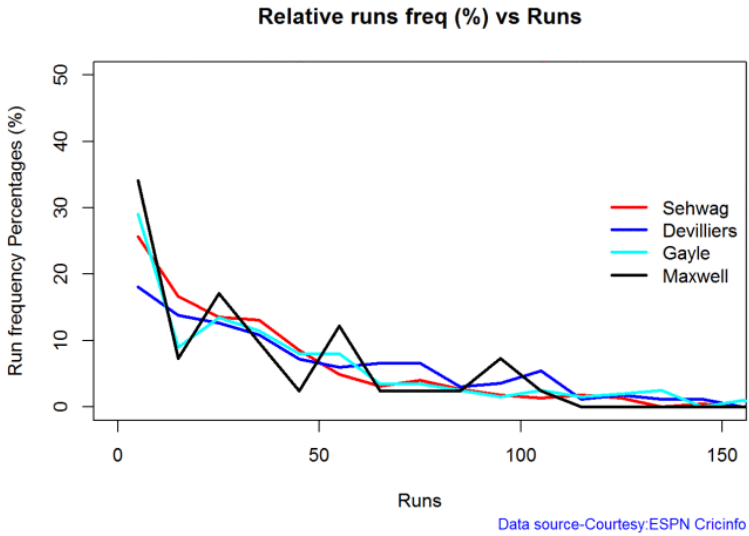
```
par(mar=c(4,4,2,2))

frames <- list("./sehwag.csv","./devilliers.csv","gayle.csv","maxwell.csv")

names <- list("Sehwag","Devilliers","Gayle","Maxwell")

relativeBatsmanSRODTT(frames,names)
```



1.3.4. Relative Runs Frequency Percentage

Sehwag leads in the percentage of runs in 10 run ranges upto 50 runs. Maxwell and Devilliers lead in 55-66 & 66-85 respectively.

```
frames <- list("./sehwag.csv","./devilliers.csv","gayle.csv","maxwell.csv")

names <- list("Sehwag","Devilliers","Gayle","Maxwell")

relativeRunsFreqPerfODTT(frames,names)
```



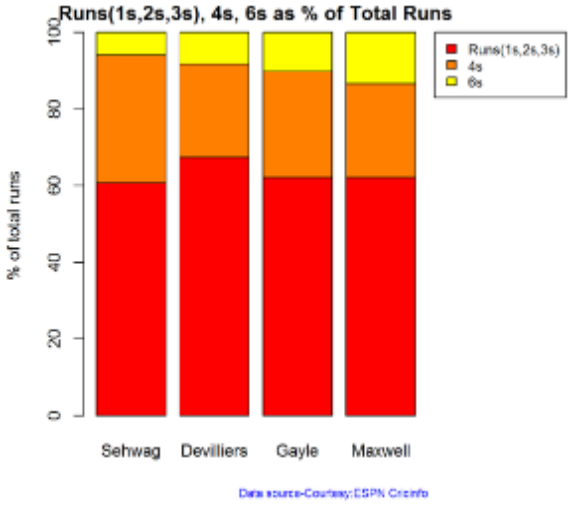
1.3.5. Percentage of 4s,6s in the runs scored

The plot below shows the percentage of runs made by the batsmen by ways of 1s,2s,3s, 4s and 6s. It can be seen that Sehwag has the highest percent of 4s (33.36%) in his overall runs in ODIs. Maxwell has the highest percentage of 6s (13.36%) in his ODI career. If we take the overall 4s+6s then Sehwag leads with (33.36 +5.95 = 39.31%),followed by Gayle (27.80+10.15=37.95%)

1.3.6. Percent 4's,6's in total runs scored

The plot below shows the contributions

```
frames <- list("./sehwag.csv", "./devilliers.csv", "gayle.csv", "maxwell.csv")

names <- list("Sehwag", "Devilliers", "Gayle", "Maxwell")

runs4s6s <-batsman4s6s(frames, names)
```




```
print(runs4s6s)
```

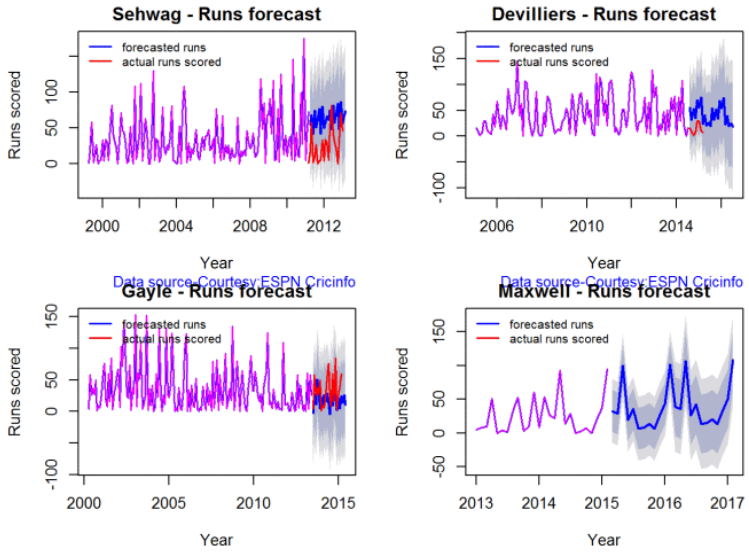
```
##                Sehwag Devilliers Gayle Maxwell
## Runs(1s,2s,3s)  60.69      67.39 62.05   62.11
## 4s              33.36      24.28 27.80   24.53
## 6s               5.95       8.32 10.15   13.36
```

1.3.7. Runs forecast

The forecast for the batsman is shown below.

```
par(mfrow=c(2,2))
par(mar=c(4,4,2,2))
batsmanPerfForecast("./sehwag.csv","Sehwag")
batsmanPerfForecast("./devilliers.csv","Devilliers")
batsmanPerfForecast("./gayle.csv","Gayle")
```

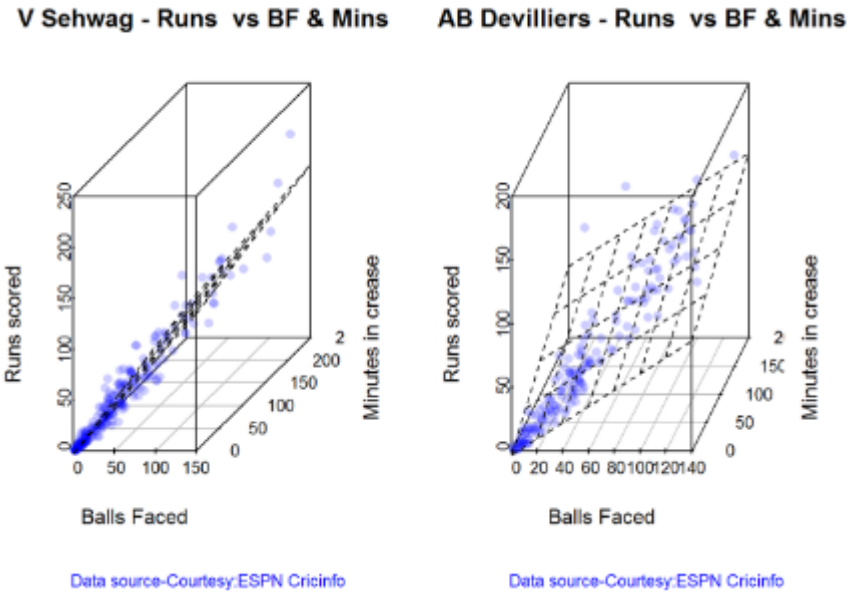
```
batsmanPerfForecast("./maxwell.csv", "Maxwell")
```



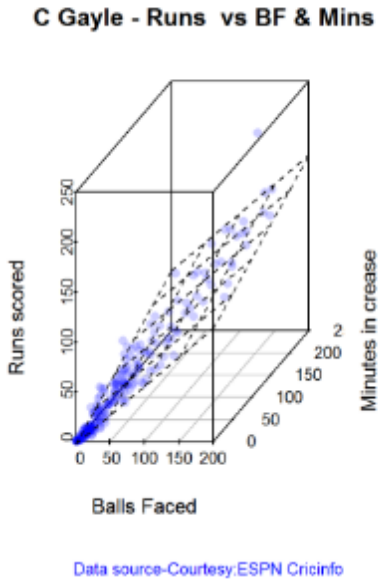
1.3.8. 3D plot of Runs vs Balls Faced and Minutes at Crease

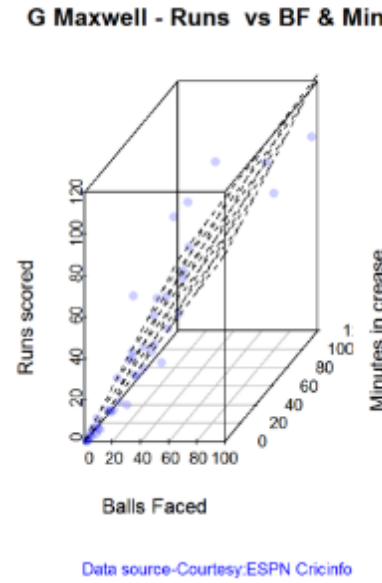
The plot is a scatter plot of Runs vs Balls faced and Minutes at Crease. A prediction plane is fitted

```
par(mfrow=c(1, 2))

par(mar=c(4, 4, 2, 2))

battingPerf3d("./sehwag.csv", "V Sehwag")

battingPerf3d("./devilliers.csv", "AB Devilliers")
```




```
par(mfrow=c(1,2))

par(mar=c(4,4,2,2))

battingPerf3d("./gayle.csv","C Gayle")

battingPerf3d("./maxwell.csv","G Maxwell")
```





1.3.9. Predicting Runs given Balls Faced and Minutes at Crease

A multi-variate regression plane is fitted between Runs and Balls faced +Minutes at crease.

```
BF <- seq( 10, 200,length=10)

Mins <- seq(30,220,length=10)

newDF <- data.frame(BF,Mins)



sehwag <- batsmanRunsPredict("./sehwag.csv","Sehwag",newdataframe=newDF)

devilliers <-
batsmanRunsPredict("./devilliers.csv","Devilliers",newdataframe=newDF)

gayle <- batsmanRunsPredict("./gayle.csv","Gayle",newdataframe=newDF)

maxwell <- batsmanRunsPredict("./maxwell.csv","Maxwell",newdataframe=newDF)
```

The fitted model is then used to predict the runs that the batsmen will score for a hypothetical Balls faced and Minutes at crease. It can be seen that Maxwell sets a searing pace in the predicted runs for a given Balls Faced and Minutes at crease followed by Sehwag. But we have to keep in mind that Maxwell has only around 1/5th of the innings of Sehwag (45 to Sehwag's 245 innings). They are followed by Devilliers and then finally Gayle

```
batsmen <-
cbind(round(sehwag$Runs),round(devilliers$Runs),round(gayle$Runs),round(maxwel
l$Runs))

colnames(batsmen) <- c("Sehwag","Devilliers","Gayle","Maxwell")

newDF <- data.frame(round(newDF$BF),round(newDF$Mins))

colnames(newDF) <- c("BallsFaced","MinsAtCrease")

predictedRuns <- cbind(newDF,batsmen)

predictedRuns
```

```
##    BallsFaced MinsAtCrease Sehwag Devilliers Gayle
Maxwell
## 1          10           30     11         12    11
18
## 2          31           51     33         32    28
43
## 3          52           72     55         52    46
67
## 4          73           93     77         71    63
92
## 5          94          114    100         91    81
```

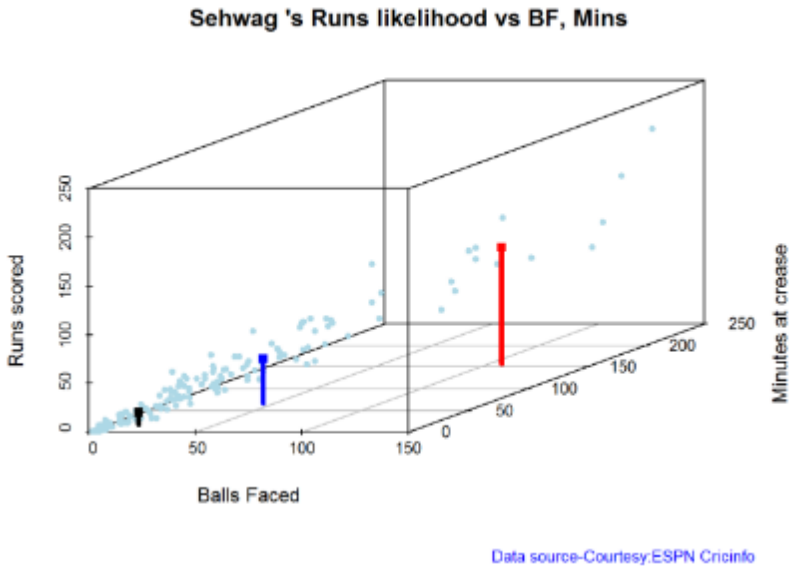
```
117

## 6          116          136     122          111     98
141

## 7          137          157     144          130    116
166

## 8          158          178     167          150    133
191

## 9          179          199     189          170    151
215

## 10         200          220     211          190    168
240
```

1.3.10. Highest runs likelihood

The plots below the runs likelihood of batsman. This uses K-Means It can be seen that Devilliers has almost 27.75% likelihood to make around 90+ runs. Gayle and Sehwag have 34% to make 40+ runs.

A. Virender Sehwag

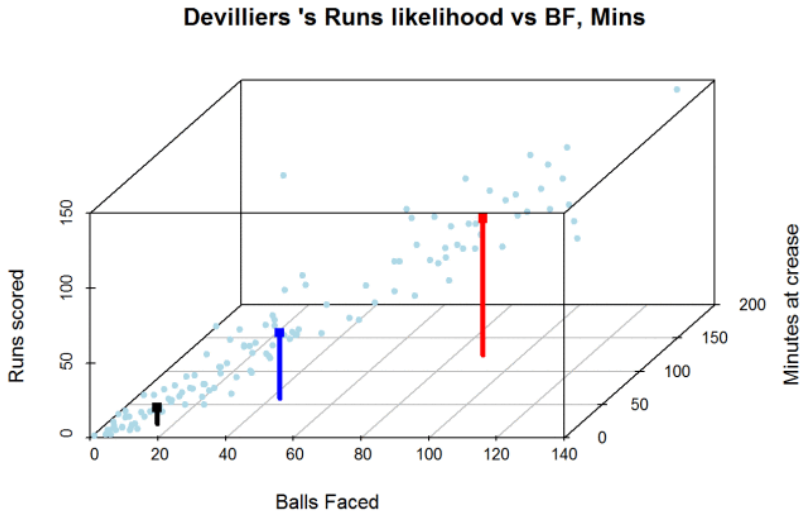
```
batsmanRunsLikelihood("./sehwag.csv","Sehwag")
```




```
## Summary of  Sehwag 's runs scoring likelihood

## ************************************************

##

## There is a 35.22 % likelihood that Sehwag  will
make  46 Runs in  44 balls over 67  Minutes

## There is a 9.43 % likelihood that Sehwag  will
make  119 Runs in  106 balls over  158  Minutes

## There is a 55.35 % likelihood that Sehwag  will
make  12 Runs in  13 balls over 18  Minutes
```

B. AB Devilliers

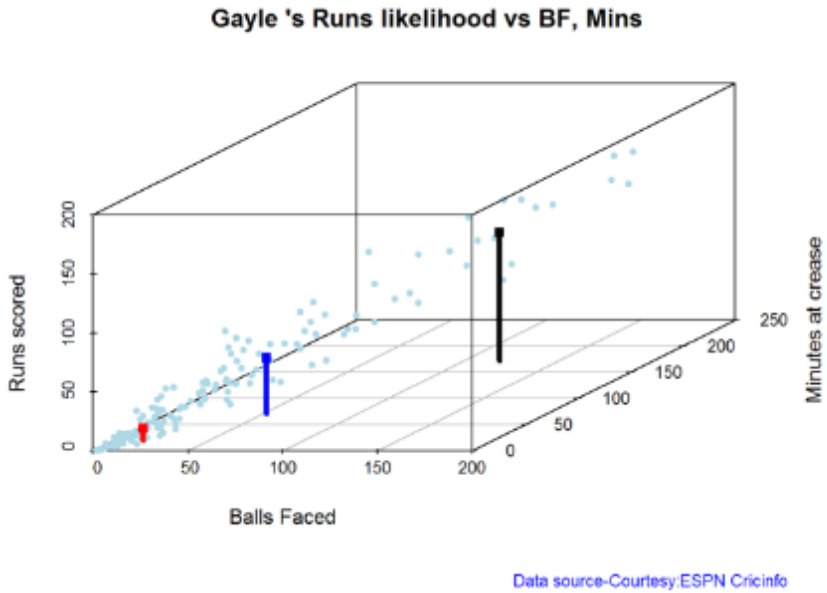
```
batsmanRunsLikelihood("./devilliers.csv", "Devilliers")
```




```
## Summary of  Devilliers 's runs scoring likelihood

## ************************************************

##

## There is a 30.65 % likelihood that Devilliers
will make  44 Runs in  43 balls over 60  Minutes

## There is a 29.84 % likelihood that Devilliers
will make  91 Runs in  88 balls over  124  Minutes

## There is a 39.52 % likelihood that Devilliers
will make  11 Runs in  15 balls over 21  Minutes
```

C. Chris Gayle

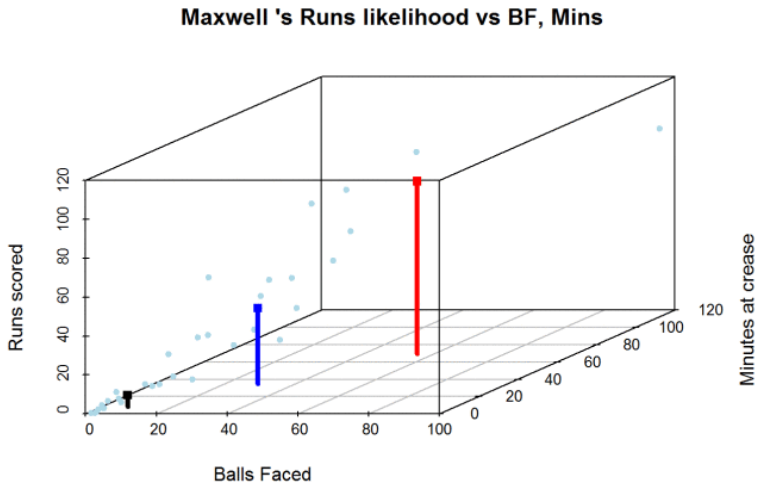
```
batsmanRunsLikelihood("./gayle.csv","Gayle")
```




```
## Summary of  Gayle 's runs scoring likelihood

## ************************************************

##

## There is a 32.69 % likelihood that Gayle  will
make  47 Runs in  51 balls over 72  Minutes

## There is a 54.49 % likelihood that Gayle  will
make  10 Runs in  15 balls over  20  Minutes

## There is a 12.82 % likelihood that Gayle  will
make  109 Runs in  119 balls over 172  Minutes
```

D. Glenn Maxwell

```
batsmanRunsLikelihood("./maxwell.csv", "Maxwell")
```



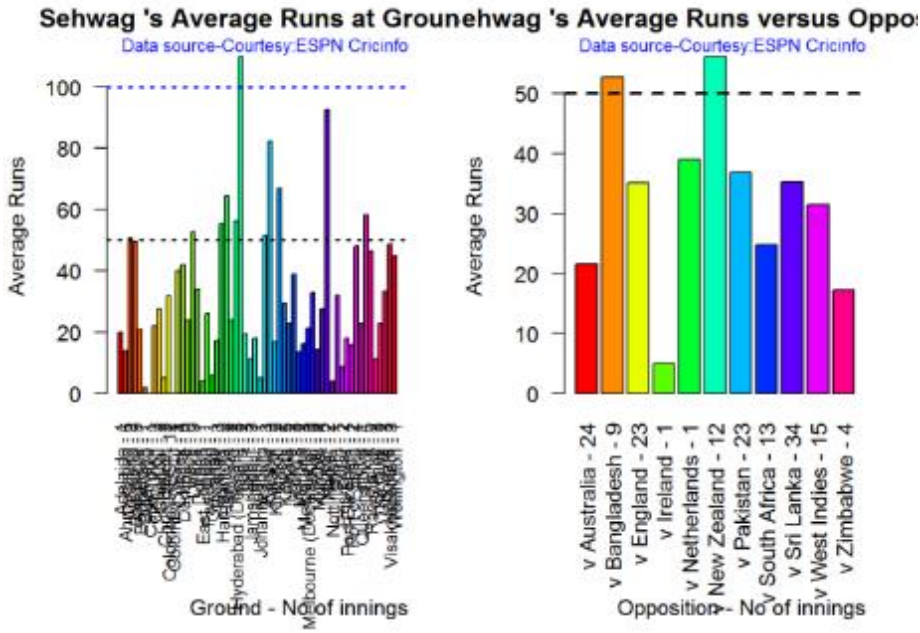

```
## Summary of  Maxwell 's runs scoring likelihood

## ************************************************

##

## There is a 34.38 % likelihood that Maxwell  will
make  39 Runs in  29 balls over 35  Minutes

## There is a 15.62 % likelihood that Maxwell  will
make  89 Runs in  55 balls over  69  Minutes

## There is a 50 % likelihood that Maxwell  will make
6 Runs in  7 balls over 9  Minutes
```

1.3.11. Average runs at ground and against opposition

A. Virender Sehwag

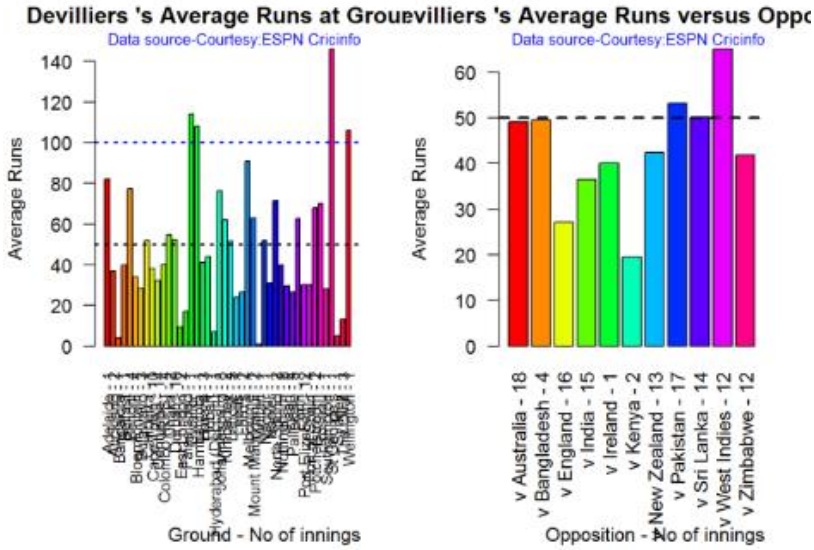
```
par(mfrow=c(1,2))

par(mar=c(4,4,2,2))
```

```
batsmanAvgRunsGround("./sehwag.csv", "Sehwag")

batsmanAvgRunsOpposition("./sehwag.csv", "Sehwag")
```



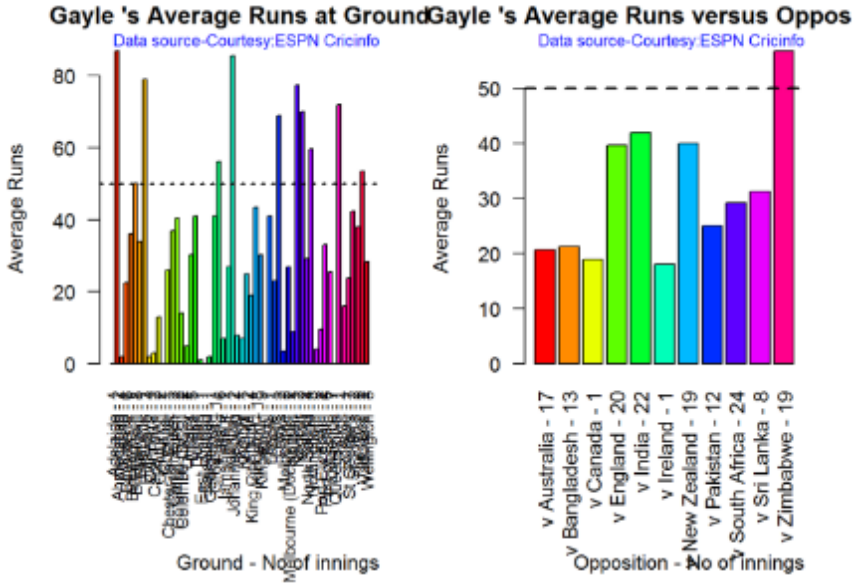
B. AB Devilliers

```
par(mfrow=c(1, 2))

par(mar=c(4, 4, 2, 2))

batsmanAvgRunsGround("./devilliers.csv", "Devilliers")

batsmanAvgRunsOpposition("./devilliers.csv", "Devilliers")
```



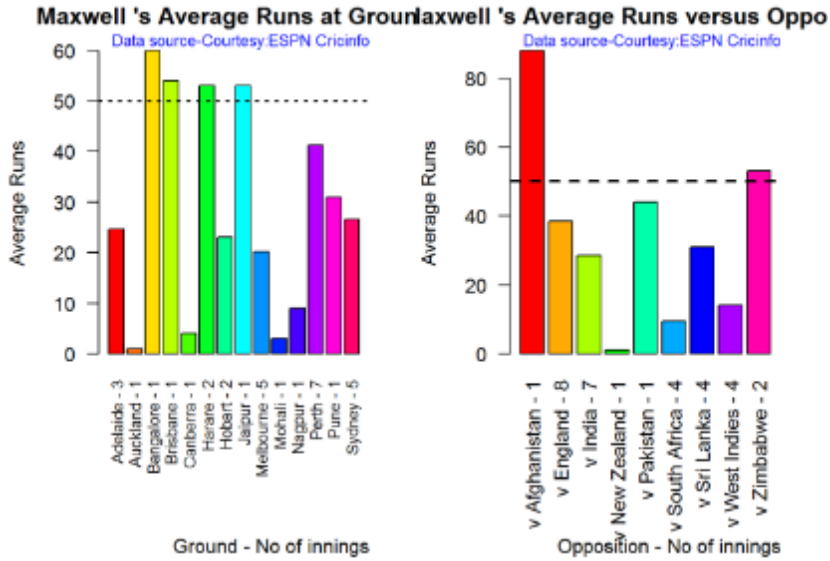
C. Chris Gayle

```
par(mfrow=c(1,2))

par(mar=c(4,4,2,2))

batsmanAvgRunsGround("./gayle.csv","Gayle")

batsmanAvgRunsOpposition("./gayle.csv","Gayle")
```



D. Glenn Maxwell

```
par(mfrow=c(1,2))

par(mar=c(4,4,2,2))

batsmanAvgRunsGround("./maxwell.csv","Maxwell")


batsmanAvgRunsOpposition("./maxwell.csv","Maxwell")
```

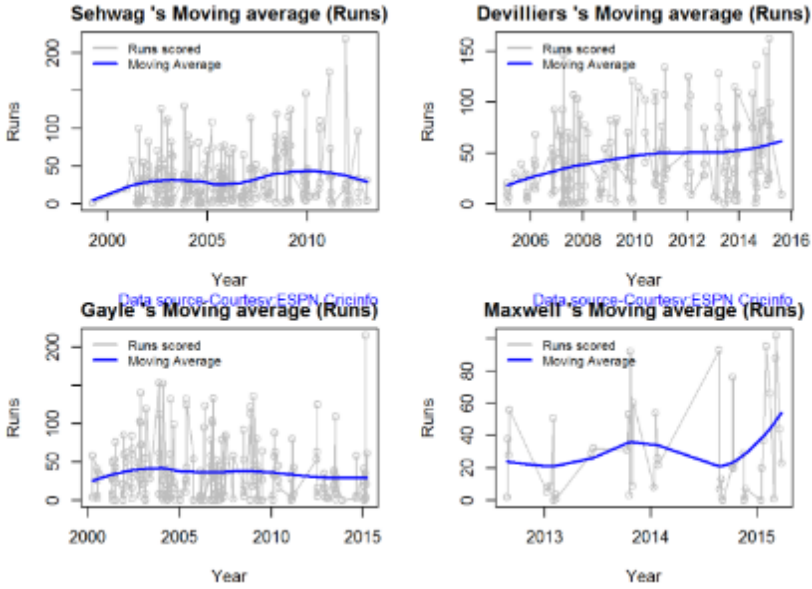


1.3.12. Moving Average of runs over career

The moving average for the 4 batsmen indicate the following

1. The moving average of Devilliers and Maxwell is on the way up.
2. Sehwag shows a slight downward trend from his 2nd peak in 2011
3. Gayle maintains a consistent 45 runs for the last few years

```
par(mfrow=c(2,2))

par(mar=c(4,4,2,2))

batsmanMovingAverage("./sehwag.csv","Sehwag")

batsmanMovingAverage("./devilliers.csv","Devilliers")

batsmanMovingAverage("./gayle.csv","Gayle")

batsmanMovingAverage("./maxwell.csv","Maxwell")
```



1.3.13. Check batsmen in-form, out-of-form

1. Maxwell, Devilliers, Sehwag are in-form. This is also evident from the moving average plot
2. Gayle is out-of-form

```
checkBatsmanInForm("./sehwag.csv","Sehwag")
```

```
##
**************************************************
**************************************

##

## Population size: 143  Mean of population: 33.76

## Sample size: 16  Mean of sample: 37.44 SD of
sample: 55.15

##

## Null hypothesis H0 : Sehwag 's sample average is
within 95% confidence interval
```

```
##         of population average

## Alternative hypothesis Ha : Sehwag 's sample
average is below the 95% confidence

##         interval of population average

##

## [1] "Sehwag 's Form Status: In-Form because the p
value: 0.603525  is greater than alpha=  0.05"

##
**************************************************
**********************************
```

checkBatsmanInForm("./devilliers.csv","Devilliers")

```
##
**************************************************
**********************************

##

## Population size: 111  Mean of population: 43.5

## Sample size: 13  Mean of sample: 57.62 SD of
sample: 40.69

##

## Null hypothesis H0 : Devilliers 's sample average
is within 95% confidence interval

##         of population average

## Alternative hypothesis Ha : Devilliers 's sample
average is below the 95% confidence

##         interval of population average
```

```
##

## [1] "Devilliers 's Form Status: In-Form because
the p value: 0.883541  is greater than alpha=  0.05"

##
**********************************************************
************************************
```

```
checkBatsmanInForm("./gayle.csv","Gayle")
```

```
##
**********************************************************
************************************

##

## Population size: 140  Mean of population: 37.1

## Sample size: 16  Mean of sample: 17.25 SD of
sample: 20.25

##

## Null hypothesis H0 : Gayle 's sample average is
within 95% confidence interval

##         of population average

## Alternative hypothesis Ha : Gayle 's sample
average is below the 95% confidence

##         interval of population average

##

## [1] "Gayle 's Form Status: Out-of-Form because the
p value: 0.000609  is less than alpha=  0.05"

##
```

```
**********************************************************
************************************

checkBatsmanInForm("./maxwell.csv","Maxwell")

##
**********************************************************
************************************

##

## Population size: 28  Mean of population: 25.25

## Sample size: 4  Mean of sample: 64.25 SD of
sample: 36.97

##

## Null hypothesis H0 : Maxwell 's sample average is
within 95% confidence interval

##         of population average

## Alternative hypothesis Ha : Maxwell 's sample
average is below the 95% confidence

##         interval of population average

##

## [1] "Maxwell 's Form Status: In-Form because the p
value: 0.948744  is greater than alpha=  0.05"

##
**********************************************************
************************************
```

1.3.14. Analysis of bowlers

1. Mitchell Johnson (Aus) – Innings-150, Wickets – 239, Econ Rate : 4.83
2. Lasith Malinga (SL)- Innings-182, Wickets – 287, Econ Rate : 5.26
3. Dale Steyn (SA)- Innings-103, Wickets – 162, Econ Rate : 4.81
4. Tim Southee (NZ)- Innings-96, Wickets – 135, Econ Rate : 5.33

Malinga has the highest number of innings and wickets followed closely by Mitchell. Steyn and Southee have relatively fewer innings.

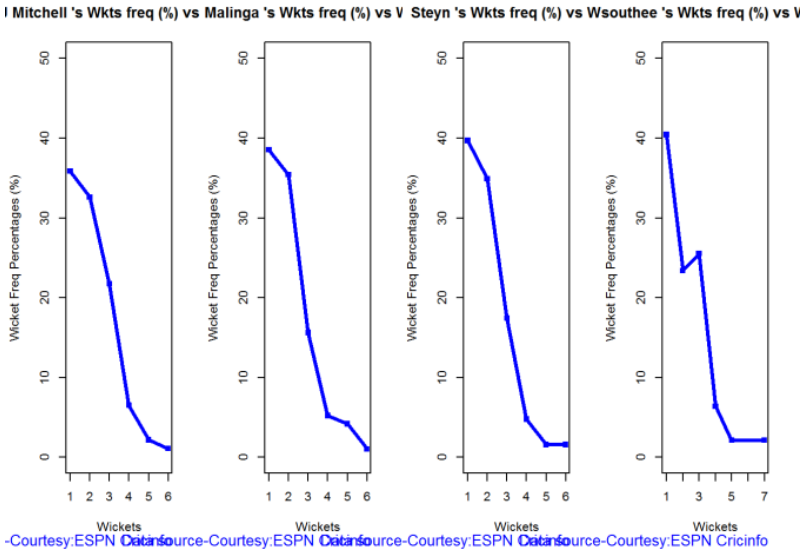
To get the bowler's data use

```
malinga <- getPlayerDataOD(49758,dir=".",file="malinga.csv",type="bowling")
```

1.3.15. Wicket Frequency percentage

This plot gives the percentage of wickets for each wickets (1,2,3…etc.)

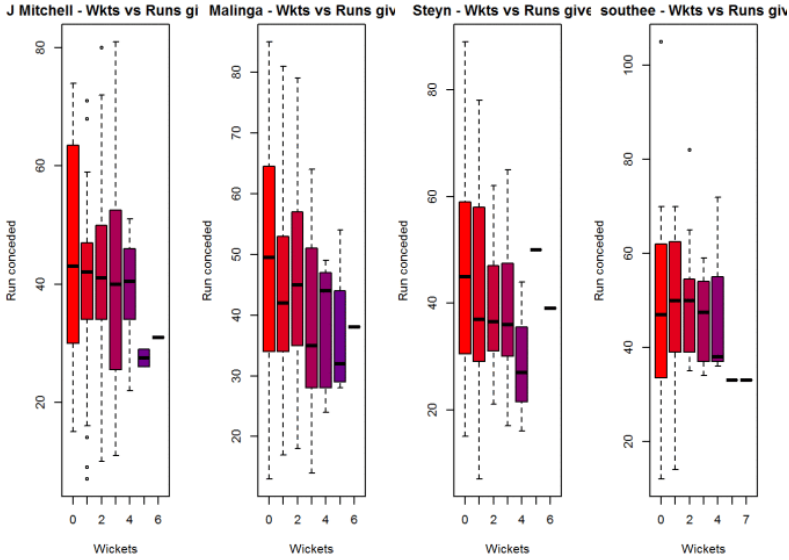
```
par(mfrow=c(1,4))

par(mar=c(4,4,2,2))

bowlerWktsFreqPercent("./mitchell.csv","J Mitchell")

bowlerWktsFreqPercent("./malinga.csv","Malinga")

bowlerWktsFreqPercent("./steyn.csv","Steyn")

bowlerWktsFreqPercent("./southee.csv","southee")
```



1.3.16. Wickets Runs plot

The plot below gives a boxplot of the runs ranges for each of the wickets taken by the bowlers. M Johnson and Steyn are more economical than Malinga and Southee corroborating the figures above

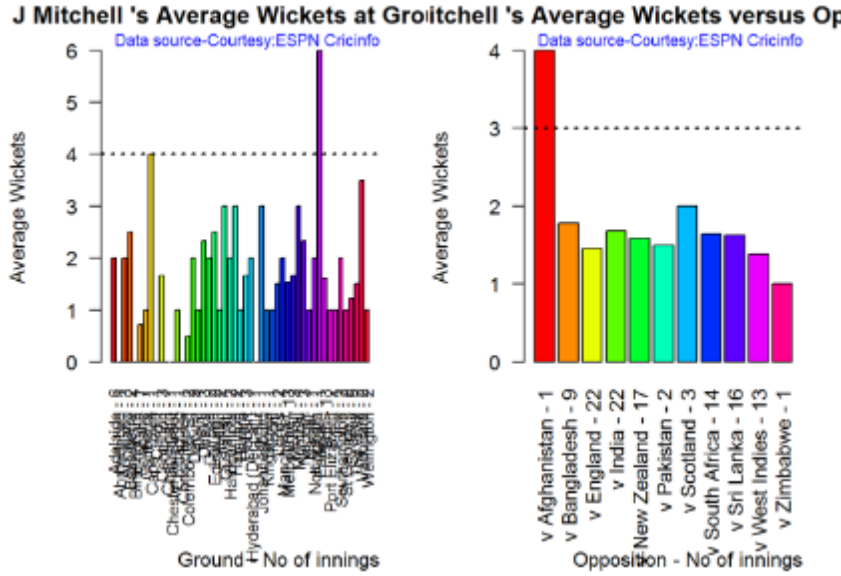
```
par(mfrow=c(1,4))

par(mar=c(4,4,2,2))



bowlerWktsRunsPlot("./mitchell.csv","J Mitchell")

bowlerWktsRunsPlot("./malinga.csv","Malinga")

bowlerWktsRunsPlot("./steyn.csv","Steyn")

bowlerWktsRunsPlot("./southee.csv","southee")
```



1.3.17. Average wickets in different grounds and opposition

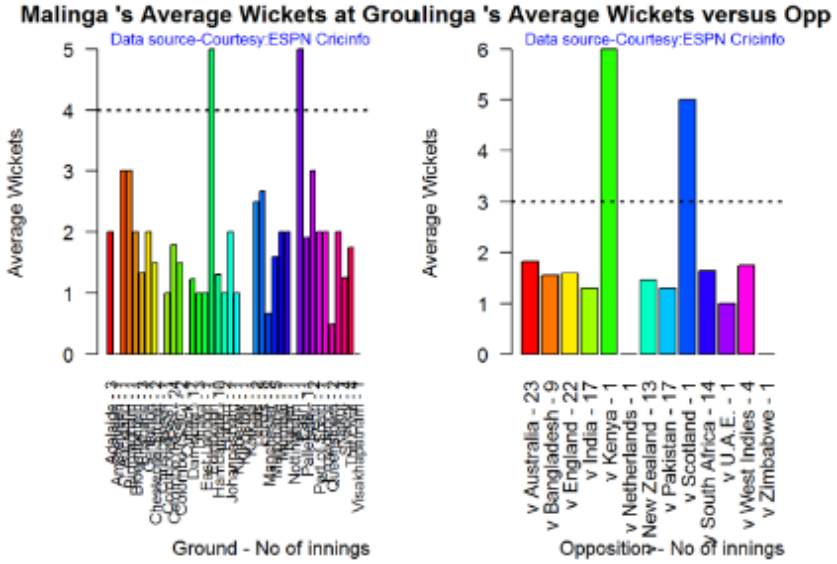
A. Mitchell Johnson

```
par(mfrow=c(1,2))

par(mar=c(4,4,2,2))

bowlerAvgWktsGround("./mitchell.csv","J Mitchell")

bowlerAvgWktsOpposition("./mitchell.csv","J Mitchell")
```



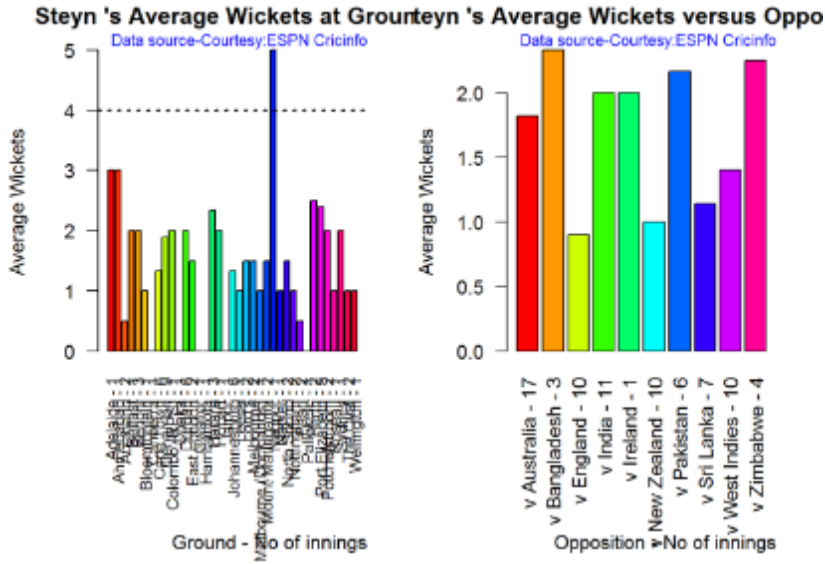
B. Lasith Malinga

```
par(mfrow=c(1,2))

par(mar=c(4,4,2,2))

bowlerAvgWktsGround("./malinga.csv","Malinga")

bowlerAvgWktsOpposition("./malinga.csv","Malinga")
```



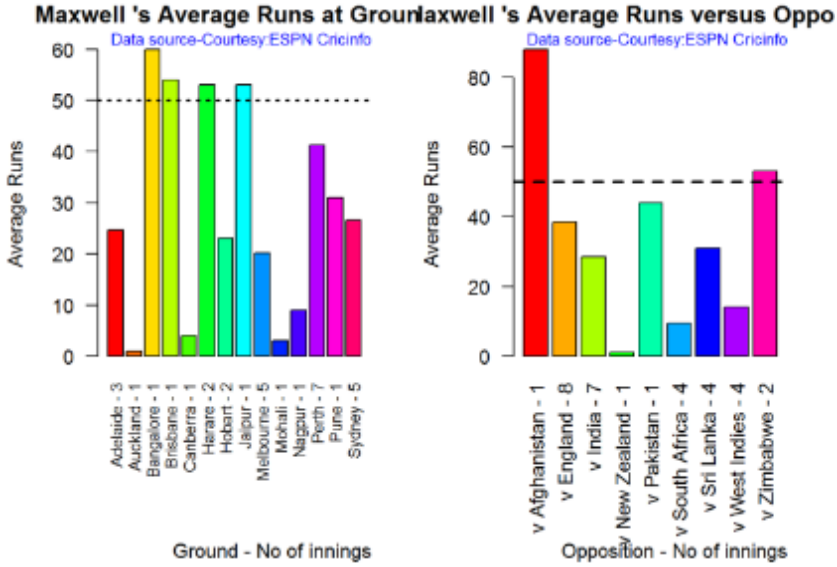
C. Dale Steyn

```
par(mfrow=c(1,2))

par(mar=c(4,4,2,2))

bowlerAvgWktsGround("./steyn.csv","Steyn")

bowlerAvgWktsOpposition("./steyn.csv","Steyn")
```



D. Tim Southee

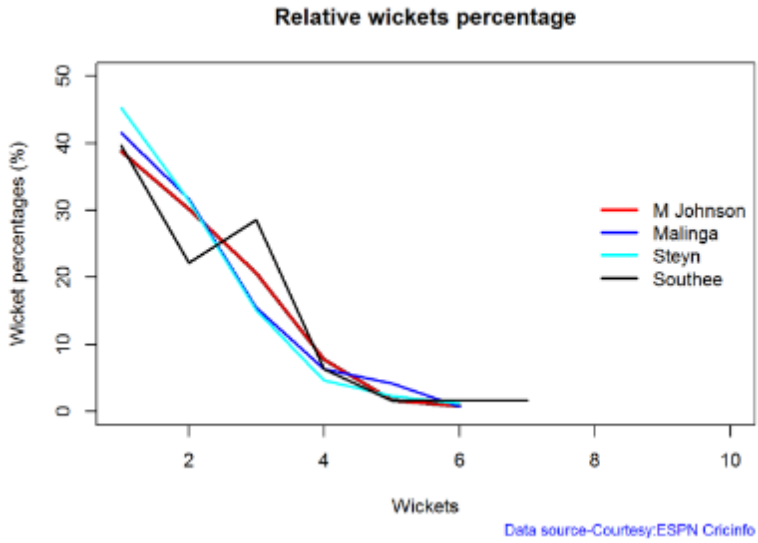
```
par(mfrow=c(1,2))

par(mar=c(4,4,2,2))

bowlerAvgWktsGround("./southee.csv","southee")


bowlerAvgWktsOpposition("./southee.csv","southee")
```



1.3.18. Relative bowling performance

The plot below shows that Mitchell Johnson and Southee have more wickets in 3-4 wickets range while Steyn and Malinga in 1-2 wicket range

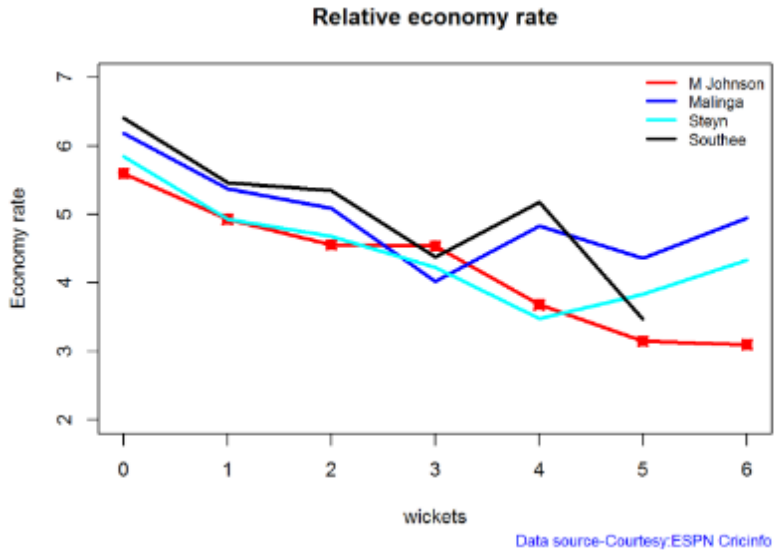
```
frames <- list("./mitchell.csv","./malinga.csv","steyn.csv","southee.csv")

names <- list("M Johnson","Malinga","Steyn","Southee")

relativeBowlingPerf(frames,names)
```



1.3.19. Relative Economy Rate against wickets taken

Steyn had the best economy rate followed by M Johnson. Malinga and Southee have a poorer economy rate

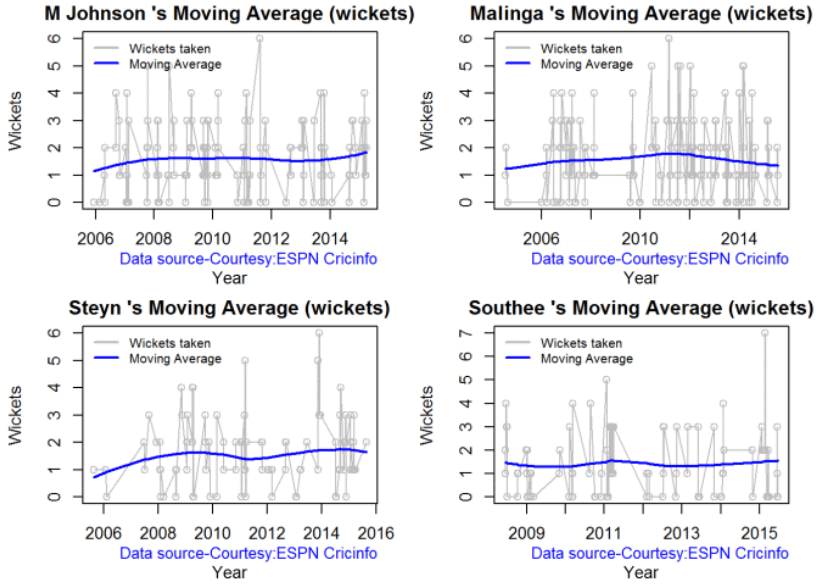
```
frames <- list("./mitchell.csv","./malinga.csv","steyn.csv","southee.csv")

names <- list("M Johnson","Malinga","Steyn","Southee")

relativeBowlingERODTT(frames,names)
```



1.3.20. Moving average of wickets over career

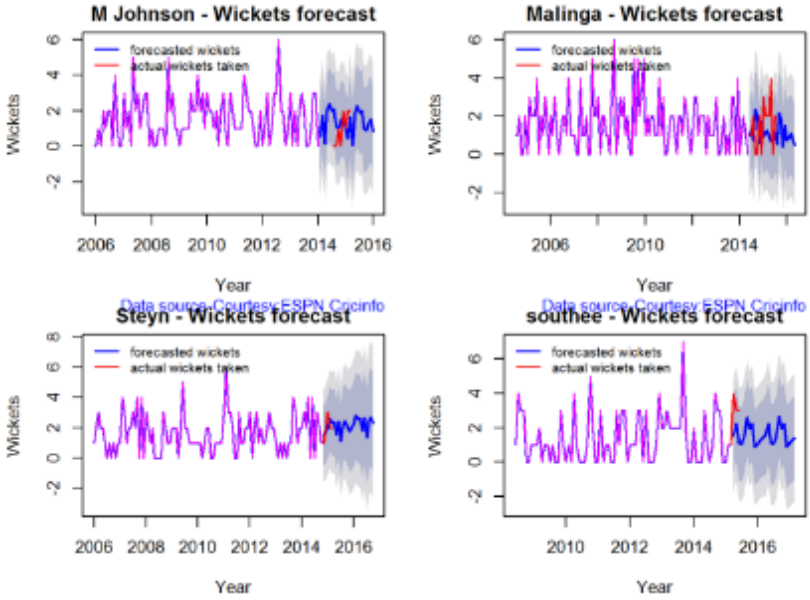
Johnson and Steyn career vs wicket graph is on the up-swing. Southee is maintaining a reasonable record while Malinga shows a decline in ODI performance

```
par(mfrow=c(2,2))

par(mar=c(4,4,2,2))

bowlerMovingAverage("./mitchell.csv","M Johnson")

bowlerMovingAverage("./malinga.csv","Malinga")

bowlerMovingAverage("./steyn.csv","Steyn")

bowlerMovingAverage("./southee.csv","Southee")
```



1.3.21. Wickets forecast

```
par(mfrow=c(2,2))

par(mar=c(4,4,2,2))

bowlerPerfForecast("./mitchell.csv","M Johnson")

bowlerPerfForecast("./malinga.csv","Malinga")

bowlerPerfForecast("./steyn.csv","Steyn")

bowlerPerfForecast("./southee.csv","southee")
```



1.3.22. Check bowler in-form, out-of-form

All the bowlers are shown to be still in-form

```
checkBowlerInForm("./mitchell.csv","J Mitchell")

##
**************************************************************
************************************

##

## Population size: 135  Mean of population: 1.55

## Sample size: 15  Mean of sample: 2 SD of sample:
1.07

##

## Null hypothesis H0 : J Mitchell 's sample average
is within 95% confidence interval
```

```
##         of population average

## Alternative hypothesis Ha : J Mitchell 's sample
average is below the 95% confidence

##         interval of population average

##

## [1] "J Mitchell 's Form Status: In-Form because
the p value: 0.937917  is greater than alpha=  0.05"

##
**************************************************
**********************************

checkBowlerInForm("./malinga.csv","Malinga")

##
**************************************************
**********************************

##

## Population size: 163  Mean of population: 1.58

## Sample size: 19  Mean of sample: 1.58 SD of
sample: 1.22

##

## Null hypothesis H0 : Malinga 's sample average is
within 95% confidence interval

##         of population average

## Alternative hypothesis Ha : Malinga 's sample
average is below the 95% confidence

##         interval of population average
```

```
##

## [1] "Malinga 's Form Status: In-Form because the p
value: 0.5  is greater than alpha=  0.05"

##
**********************************************************
************************************
```

checkBowlerInForm("./steyn.csv","Steyn")

```
##
**********************************************************
************************************

##

## Population size: 93  Mean of population: 1.59

## Sample size: 11  Mean of sample: 1.45 SD of
sample: 0.69

##

## Null hypothesis H0 : Steyn 's sample average is
within 95% confidence interval

##         of population average

## Alternative hypothesis Ha : Steyn 's sample
average is below the 95% confidence

##         interval of population average

##

## [1] "Steyn 's Form Status: In-Form because the p
value: 0.257438  is greater than alpha=  0.05"

##
```

```
**************************************************
************************************

checkBowlerInForm("./southee.csv","southee")

##
**************************************************
************************************

##

## Population size: 86  Mean of population: 1.48

## Sample size: 10  Mean of sample: 0.8 SD of sample:
1.14

##

## Null hypothesis H0 : southee 's sample average is
within 95% confidence interval

##         of population average

## Alternative hypothesis Ha : southee 's sample
average is below the 95% confidence

##         interval of population average

##

## [1] "southee 's Form Status: Out-of-Form because
the p value: 0.044302  is less than alpha=  0.05"

##
**************************************************
************************************

***************
```

1.3.23. Key findings

Here are some key conclusions ODI batsmen

1. AB Devilliers has high frequency of runs in the 60-120 range and the highest average
2. Sehwag has the most number of innings and good strike rate
3. Maxwell has the best strike rate but it should be kept in mind that he has 1/5 of the innings of Sehwag. We need to see how he progress further
4. Sehwag has the highest percentage of 4s in the runs scored, while Maxwell has the most 6s
5. For a hypothetical Balls Faced and Minutes at creases Maxwell will score the most runs followed by Sehwag
6. The moving average of indicates that the best is yet to come for Devilliers and Maxwell. Sehwag has a few more years in him while Gayle shows a decline in ODI performance and an out of form is indicated.

ODI bowlers

1. Malinga has the highest played the highest innings and also has the highest wickets though he has poor economy rate
2. M Johnson is the most effective in the 3-4 wicket range followed by Southee
3. M Johnson and Steyn has the best overall economy rate followed by Malinga and Steyn 4 M Johnson and Steyn's career is on the up-swing, Southee maintains a steady consistent performance, while Malinga shows a downward trend

1.4. cricketr adapts to the Twenty20 International!

1.4.1. Introduction

This should be last in the series of posts based on my R package cricketr. That is, unless some bright idea comes trotting along and light bulbs go on around my head.

In this post cricketr adapts to the Twenty20 International format. Now cricketr can handle stats from all 3 formats of the game namely Test matches, ODIs and Twenty20 International from ESPN Cricinfo. You should be able to install the package from GitHub and use the many of the functions available in the package.

Please be mindful of the ESPN Cricinfo Terms of Use

I have chosen the Top 4 batsmen and top 4 bowlers based on ICC rankings and/or number of matches played.

Batsmen

1. Virat Kohli (Ind)
2. Faf du Plessis (SA)
3. A J Finch (Aus)
4. Brendon McCullum (Aus)

Bowlers

1. Samuel Badree (WI)
2. Sunil Narine (WI)
3. Ravichandran Ashwin (Ind)
4. Ajantha Mendis (SL)

I have explained the plots and added my own observations. Please feel free to draw your conclusions!

The data for a particular player can be obtained with the getPlayerData() function. To do you will need to go to ESPN CricInfo Player and type in the name of the player for e.g Virat Kohli, Sunil Narine etc. This will bring up a page which have the profile number for the player e.g. for Virat Kohli this would be http://www.espncricinfo.com/india/content/player/253802.html.

The package can be installed directly from CRAN

```
if (!require("cricketr")){
   install.packages("cricketr",lib = "c:/test")
}

library(cricketr)
```

or from Github

```
library(devtools)
install_github("tvganesh/cricketr")

library(cricketr)
```

The data for a particular player can be obtained with the getPlayerData() function. To do you will need to go to ESPN CricInfo Player and type in the name of the player for e.g. Virat Kohli, Sunil Narine etc. This will bring up a page which have the profile number for the player e.g. for Virat Kohli this would be

http://www.espncricinfo.com/india/content/player/253802.html. Hence, Kohli's profile is 253802. This can be used to get the data for Virat Kohli as shown below

```
kohli <- getPlayerDataTT(253802,dir="..",file="kohli.csv",type="batting")
```

The analysis is included below

1.4.2. Analyses of Batsmen

The following plots gives the analysis of the 4 ODI batsmen

- Virat Kohli (Ind) – Innings-26, Runs-972, Average-46.28,Strike Rate-131.70
- Faf du Plessis (SA) – Innings-24, Runs-805, Average-42.36,Strike Rate-135.75
- A J Finch (Aus) – Innings-22, Runs-756, Average-39.78,Strike Rate-152.41
- Brendon McCullum (NZ) – Innings-70, Runs-2140, Average-35.66,Strike Rate-136.21

1.4.3. Plot of 4s, 6s and the scoring rate in ODIs

The 3 charts below give the number of

1. 4s vs Runs scored
2. 6s vs Runs scored
3. Balls faced vs Runs scored A regression line is fitted in each of these plots for each of the ODI batsmen

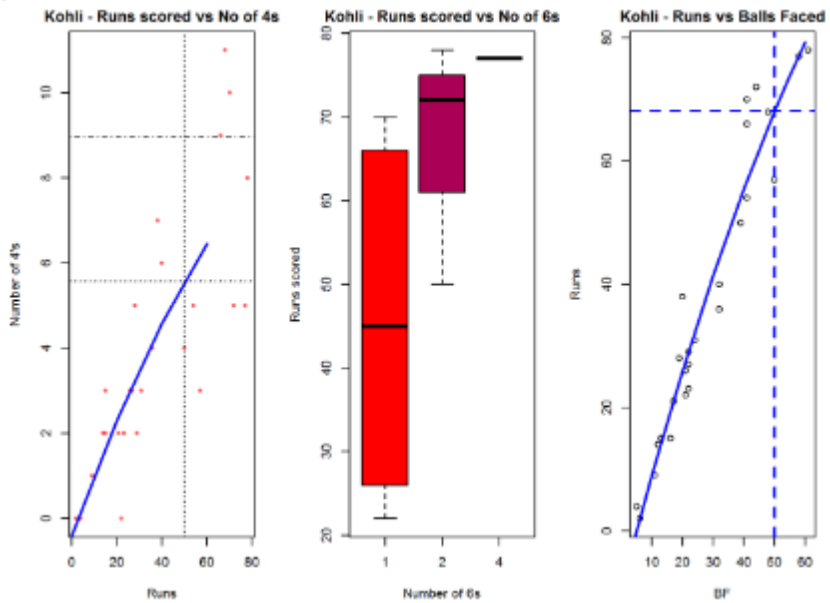
A. Virat Kohli

– The 1st plot shows that Kohli approximately hits about 5 4's on his way to the 50s

– The 2nd box plot of no of 6s and runs shows the range of runs when Kohli scored 1,2 or 4 6s. The dark line in the box shows the average runs when he scored those number of 6s. So when he scored 1 6 the average runs he scored was 45

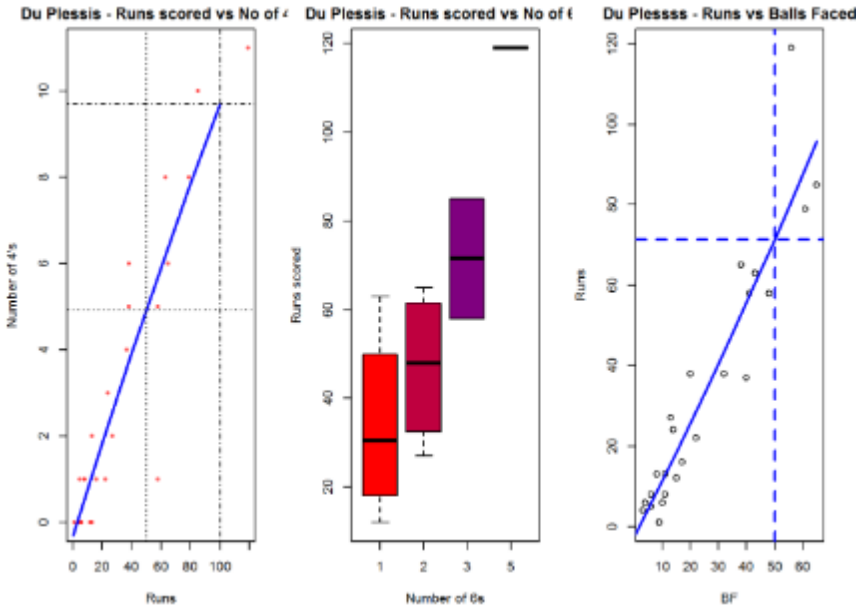
– The 3rd plot shows the number of runs scored against the balls faced. It can be seen when Kohli faced 50 balls he had scored around ~ 70 runs

```
par(mfrow=c(1,3))

par(mar=c(4,4,2,2))

batsman4s("./kohli.csv","Kohli")

batsman6s("./kohli.csv","Kohli")

batsmanScoringRateODTT("./kohli.csv","Kohli")
```



B. Faf du Plessis

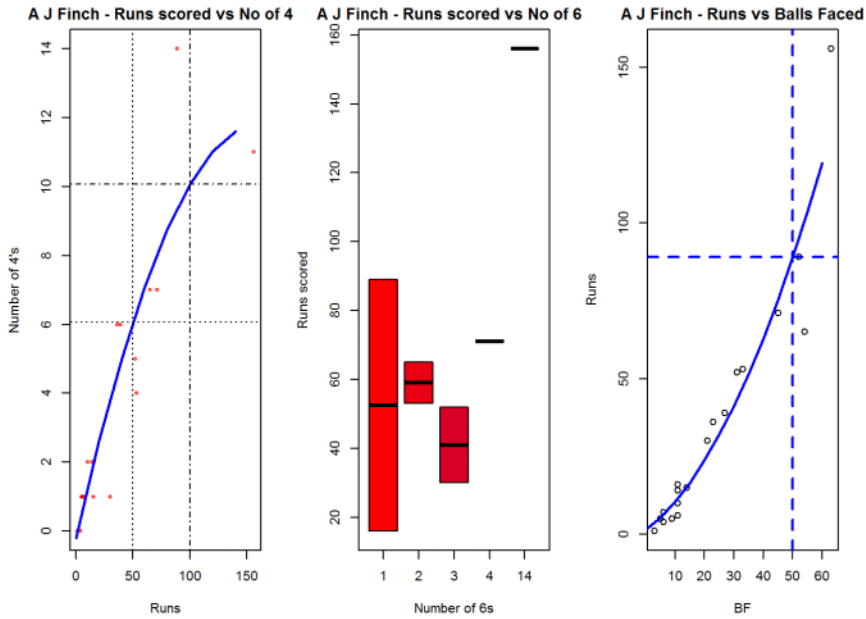
```
par(mfrow=c(1,3))

par(mar=c(4,4,2,2))

batsman4s("./plessis.csv","Du Plessis")

batsman6s("./plessis.csv","Du Plessis")

batsmanScoringRateODTT("./plessis.csv","Du Plessss")
```



C. A J Finch

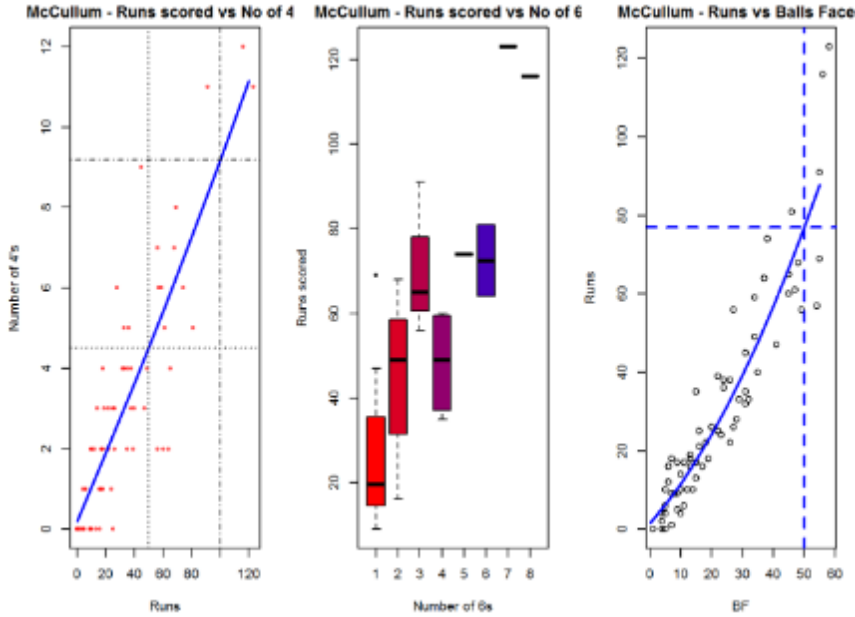
```
par(mfrow=c(1,3))

par(mar=c(4,4,2,2))

batsman4s("./finch.csv","A J Finch")

batsman6s("./finch.csv","A J Finch")
```

```
batsmanScoringRateODTT("./finch.csv", "A J Finch")
```



D. Brendon McCullum

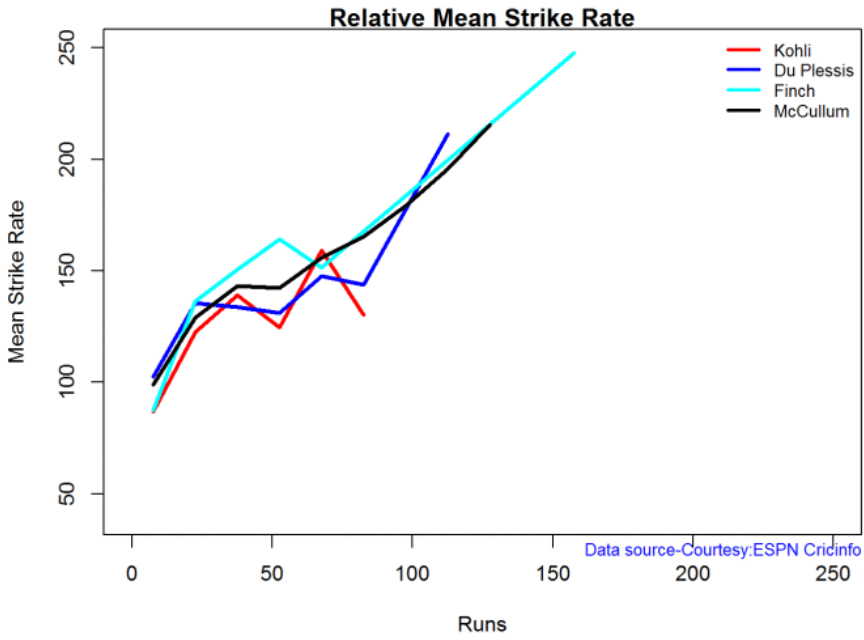
```
par(mfrow=c(1,3))

par(mar=c(4,4,2,2))

batsman4s("./mccullum.csv", "McCullum")

batsman6s("./mccullum.csv", "McCullum")

batsmanScoringRateODTT("./mccullum.csv", "McCullum")
```



1.4.4. Relative Mean Strike Rate

This plot shows the Mean Strike Rate of the batsman in each run range. It can be seen the A J Finch has the best strike rate followed by B McCullum.

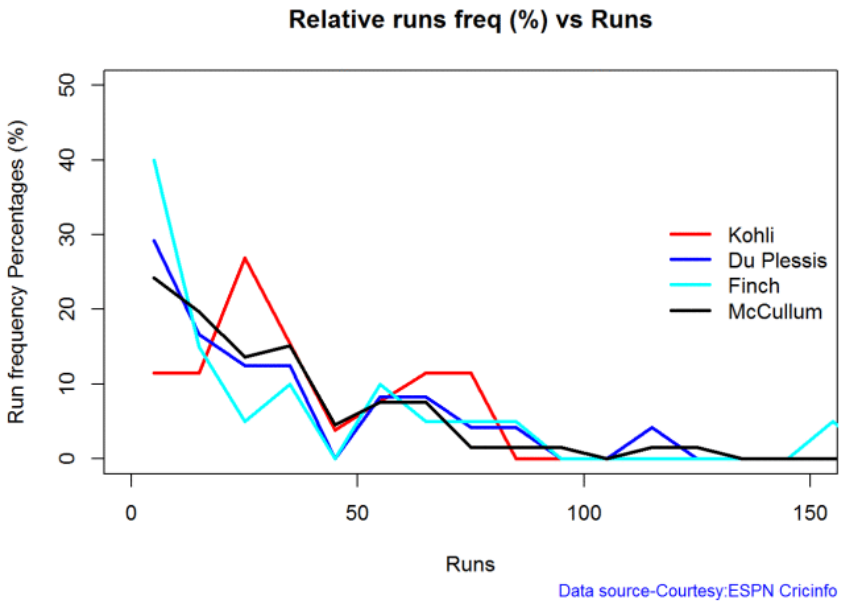
```
par(mar=c(4,4,2,2))

frames <- list("./kohli.csv","./plessis.csv","finch.csv","mccullum.csv")

names <- list("Kohli","Du Plessis","Finch","McCullum")

relativeBatsmanSRODTT(frames,names)
```



1.4.5. Relative Runs Frequency Percentage

The plot below provides the average runs scored in each run range 0-5,5-10,10-15 etc. Clearly Kohli has the most runs scored in most of the runs ranges. . This is also evident in the fact that Kohli has the highest average. He is followed by McCullum

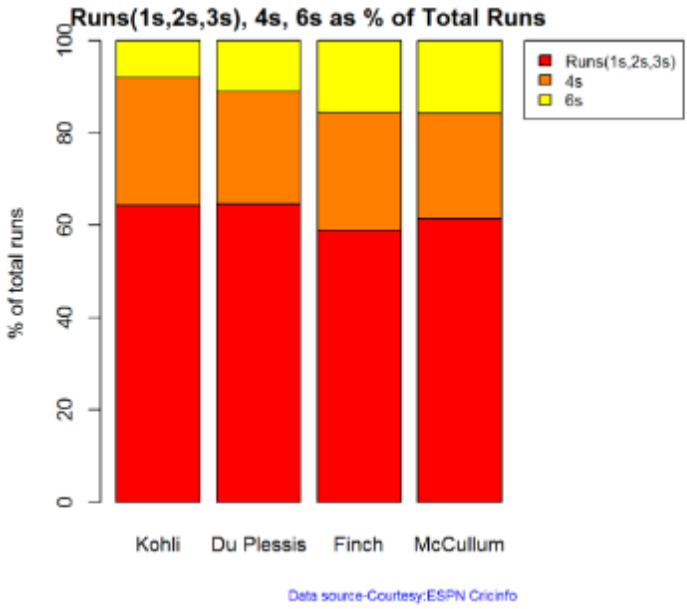
```
frames <- list("./kohli.csv","./plessis.csv","finch.csv","mccullum.csv")

names <- list("Kohli","Du Plessis","Finch","McCullum")

relativeRunsFreqPerfODTT(frames,names)
```



1.4.6. Percent 4's,6's in total runs scored

The plot below shows the percentage of runs scored by way of 4s and 6s for each batsman. Du Plessis has the highest percentage of 4s, McCullum has the highest 6s. Finch has the highest percentage of 4s & 6s – 25.37 + 15.64= 41.01%

```
rames  <-  list("./kohli.csv","./plessis.csv","finch.csv","mccullum.csv")

names  <-  list("Kohli","Du Plessis","Finch","McCullum")

runs4s6s  <-batsman4s6s(frames,names)
```




```
print(runs4s6s)
```

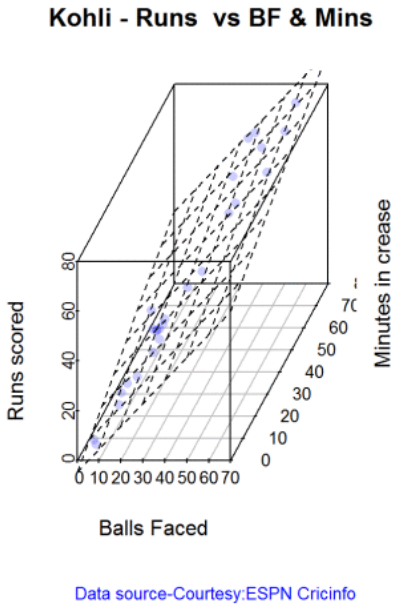
```
##                Kohli Du Plessis Finch McCullum
## Runs(1s,2s,3s) 64.29      64.55 58.99    61.45
## 4s             27.78      24.38 25.37    22.87
## 6s              7.94      11.07 15.64    15.69
```

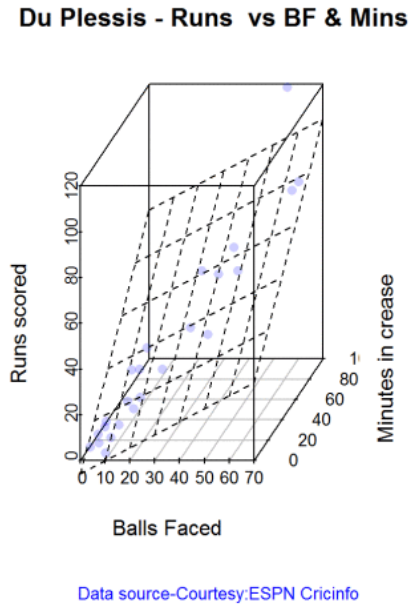
1.4.7. 3D plot of Runs vs Balls Faced and Minutes at Crease

The plot is a scatter plot of Runs vs Balls faced and Minutes at Crease. A prediction plane is then fitted based on the Balls Faced and Minutes at Crease to give the runs scored

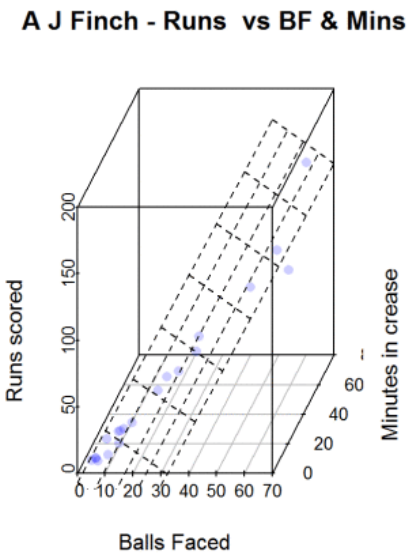
```
par(mfrow=c(1,2))

par(mar=c(4,4,2,2))
```

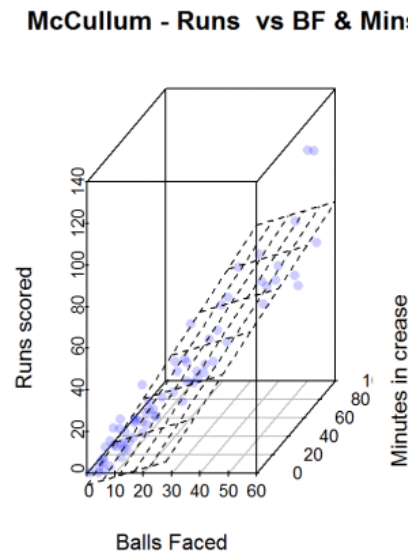
```
battingPerf3d("./kohli.csv", "Kohli")

battingPerf3d("./plessis.csv", "Du Plessis")
```






```
par(mfrow=c(1,2))

par(mar=c(4,4,2,2))

battingPerf3d("./finch.csv", "A J Finch")

battingPerf3d("./mccullum.csv", "McCullum")
```





1.4.8. Predicting Runs given Balls Faced and Minutes at Crease

A hypothetical Balls faced and Minutes at Crease is used to predict the runs scored by each batsman based on the computed prediction plane

```
BF <- seq( 5, 70,length=10)

Mins <- seq(5,70,length=10)

newDF <- data.frame(BF,Mins)



kohli <- batsmanRunsPredict("./kohli.csv","Kohli",newdataframe=newDF)

plessis <- batsmanRunsPredict("./plessis.csv","Du Plessis",newdataframe=newDF)

finch <- batsmanRunsPredict("./finch.csv","A J Finch",newdataframe=newDF)
```

```
mccullum <-
batsmanRunsPredict("./mccullum.csv","McCullum",newdataframe=newDF)
```

The predicted runs is displayed. As can be seen Finch has the best overall strike rate followed by McCullum.

```
batsmen <-
cbind(round(kohli$Runs),round(plessis$Runs),round(finch$Runs),round(mccullum$Runs))

colnames(batsmen) <- c("Kohli","Du Plessis","Finch","McCullum")

newDF <- data.frame(round(newDF$BF),round(newDF$Mins))

colnames(newDF) <- c("BallsFaced","MinsAtCrease")

predictedRuns <- cbind(newDF,batsmen)

predictedRuns
```

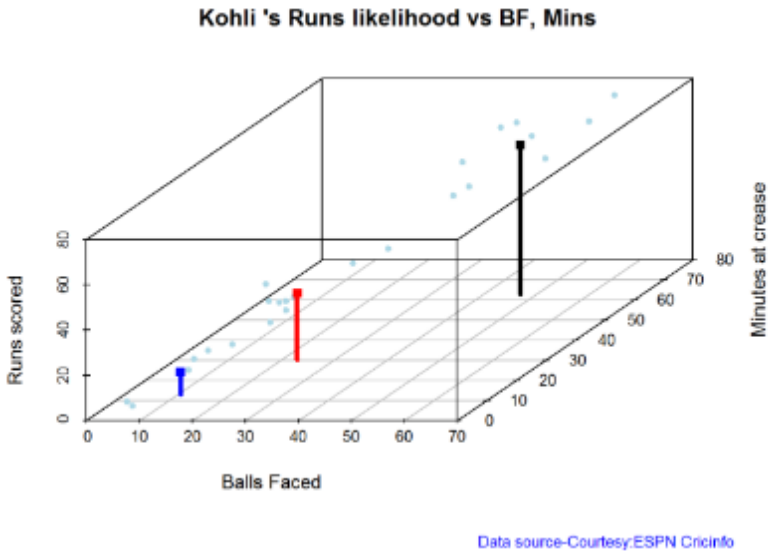
```
##    BallsFaced MinsAtCrease Kohli Du Plessis Finch McCullum
## 1           5            5     2          1     5        3
## 2          12           12    12         10    22       16
## 3          19           19    22         19    40       28
## 4          27           27    31         28    57       41
## 5          34           34    41         37    74       54
## 6          41           41    51         47    91
```

```
66

## 7          48          48    60          56   108
79

## 8          56          56    70          65   125
91

## 9          63          63    79          74   142
104

## 10         70          70    89          84   159
117
```

1.4.9. Highest runs likelihood

The plots below the runs likelihood of batsman. This uses K-Means Kohli has the highest likelihood of scoring runs 34.2% likely to score 66 runs. Du Plessis has 25% likelihood to score 53 runs, A. Virat Kohli

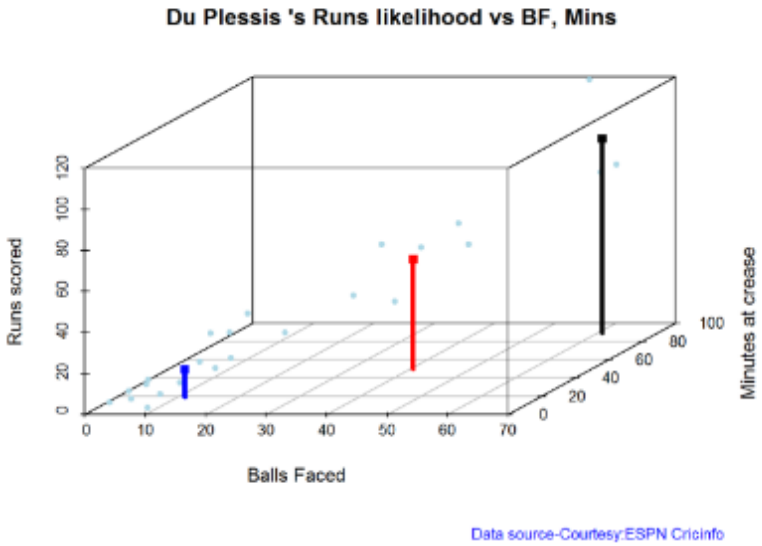
```
batsmanRunsLikelihood("./kohli.csv","Kohli")
```




```
## Summary of  Kohli 's runs scoring likelihood

## ************************************************

##

## There is a 23.08 % likelihood that Kohli  will
make  10 Runs in  10 balls over 13  Minutes

## There is a 42.31 % likelihood that Kohli  will
make  29 Runs in  23 balls over  30  Minutes

## There is a 34.62 % likelihood that Kohli  will
make  66 Runs in  47 balls over 63  Minutes
```

B. Faf Du Plessis

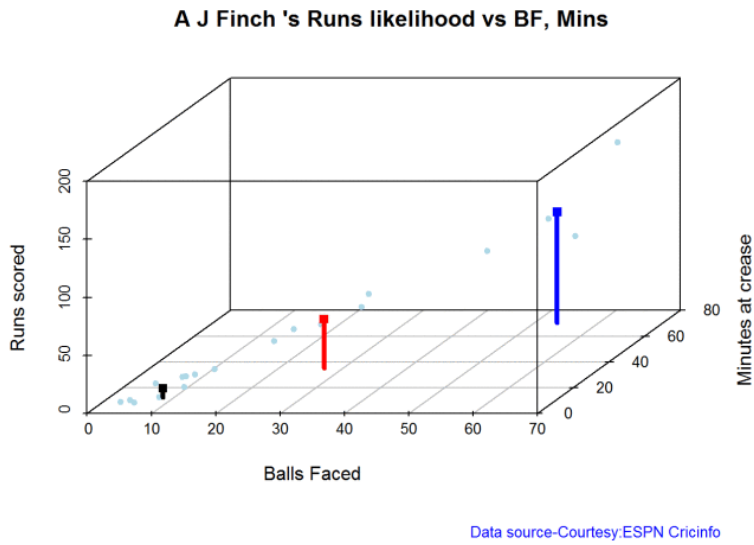
```
batsmanRunsLikelihood("./plessis.csv","Du Plessis")
```




```
## Summary of  Du Plessis 's runs scoring likelihood

## ************************************************

##

## There is a 62.5 % likelihood that Du Plessis  will
make  14 Runs in  11 balls over 19  Minutes

## There is a 25 % likelihood that Du Plessis  will
make  53 Runs in  40 balls over  50  Minutes

## There is a 12.5 % likelihood that Du Plessis  will
make  94 Runs in  61 balls over 90  Minutes
```

C. A J Finch

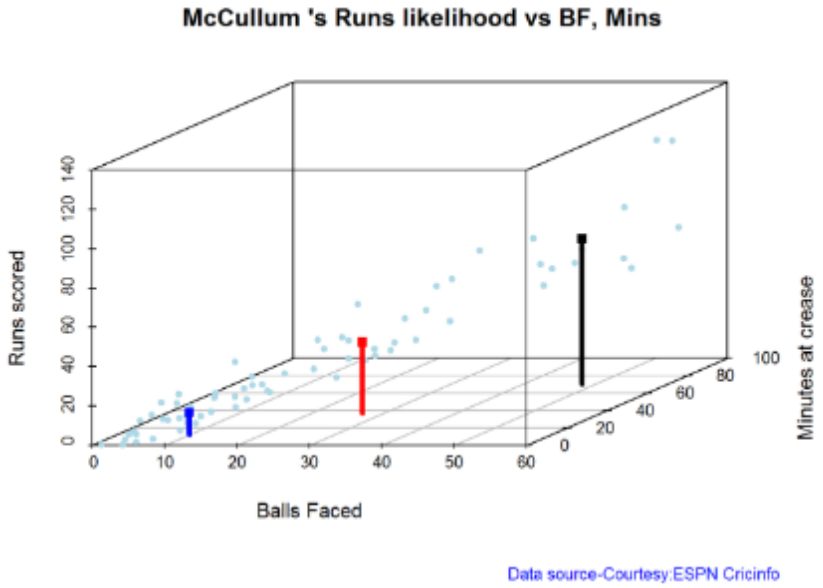
```
batsmanRunsLikelihood("./finch.csv","A J Finch")
```




```
## Summary of  A J Finch 's runs scoring likelihood

## ************************************************

##

## There is a 20 % likelihood that A J Finch  will
make  95 Runs in  54 balls over 70  Minutes

## There is a 25 % likelihood that A J Finch  will
make  42 Runs in  27 balls over  35  Minutes

## There is a 55 % likelihood that A J Finch  will
make  8 Runs in  8 balls over 12  Minutes
```

D. Brendon McCullum

```
batsmanRunsLikelihood("./mccullum.csv","McCullum")
```




```
## Summary of  McCullum 's runs scoring likelihood

## ************************************************

##

## There is a 50.72 % likelihood that McCullum  will
make  11 Runs in  10 balls over 13  Minutes

## There is a 28.99 % likelihood that McCullum  will
make  36 Runs in  27 balls over  37  Minutes

## There is a 20.29 % likelihood that McCullum  will
make  74 Runs in  48 balls over 70  Minutes
```

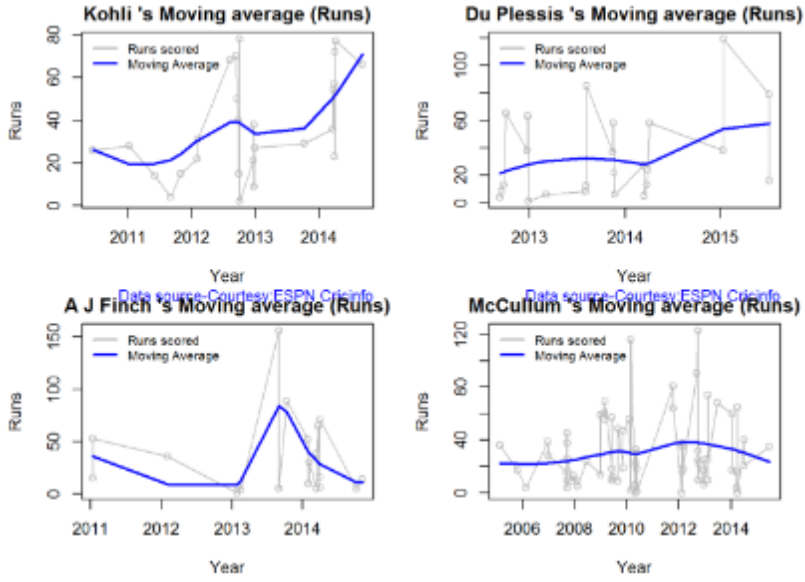
1.4.10. Moving Average of runs over career

The moving average for the 4 batsmen indicate the following. It must be noted that there is not sufficient data yet on Twenty20 Internationals. Kohli, Du Plessis and Finch average only 26 innings while McCullum has

close to 70. So the moving average while an indication will regress towards the mean over time.

1. The moving average of Kohli and Du Plessis is on the way up.
2. McCullum has a consistent performance while Finch had a brief burst in 2013-2014

```
par(mfrow=c(2,2))

par(mar=c(4,4,2,2))

batsmanMovingAverage("./kohli.csv","Kohli")

batsmanMovingAverage("./plessis.csv","Du Plessis")

batsmanMovingAverage("./finch.csv","A J Finch")

batsmanMovingAverage("./mccullum.csv","McCullum")
```



1.4.11. Analysis of bowlers

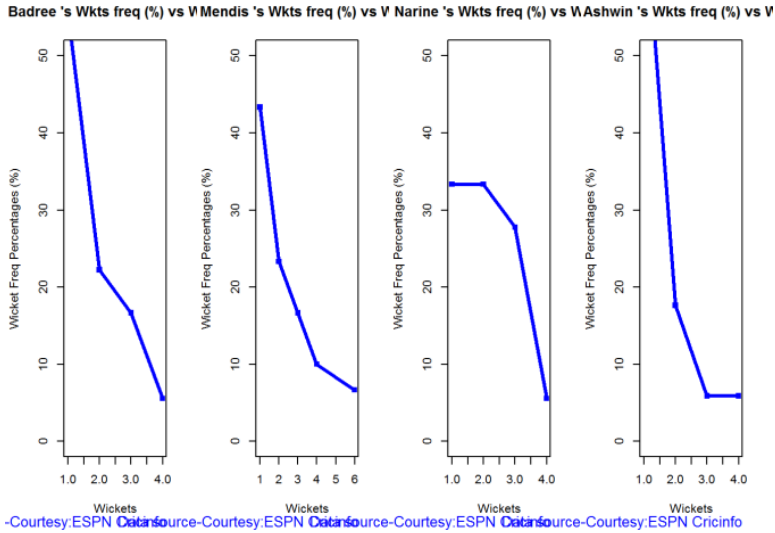
1. Samuel Badree (WI) – Innings-22, Runs -464, Wickets – 31, Econ Rate : 5.39
2. Sunil Narine (WI)- Innings-31,Runs-666, Wickets – 38 , Econ Rate : 5.70
3. Ravichandran Ashwin (Ind)- Innings-26, Runs- 732, Wickets – 25, Econ Rate : 7.32
4. Ajantha Mendis (SL)- Innings-39, Runs – 952,Wickets – 66, Econ Rate : 6.45

The plot shows the frequency with which the bowlers have taken 1,2,3 etc. wickets. The most wickets taken is by Ajantha Mendis (6 wickets)

1.4.12. Wicket Frequency percentage

This plot gives the percentage of wickets for each wickets (1,2,3…etc.)

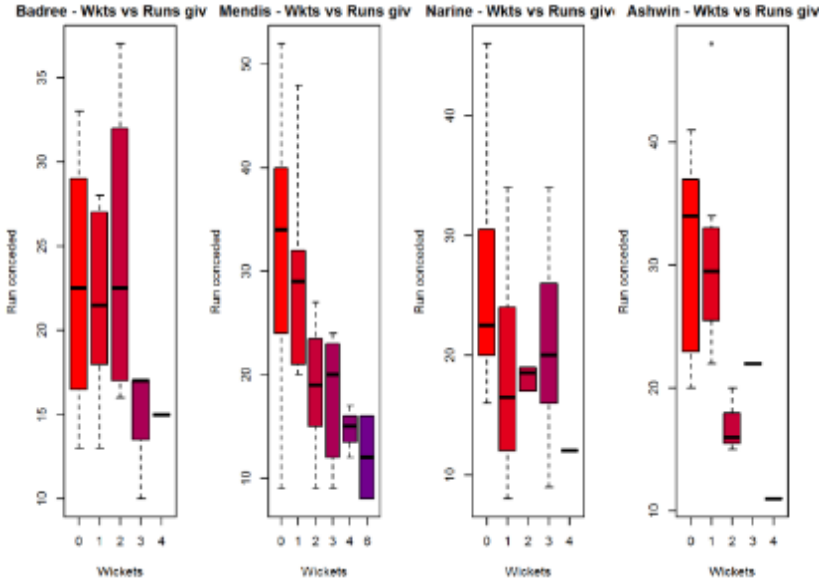
```
par(mfrow=c(1,4))

par(mar=c(4,4,2,2))

bowlerWktsFreqPercent("./badree.csv","Badree")

bowlerWktsFreqPercent("./mendis.csv","Mendis")

bowlerWktsFreqPercent("./narine.csv","Narine")

bowlerWktsFreqPercent("./ashwin.csv","Ashwin")
```



1.4.13. Wickets Runs plot

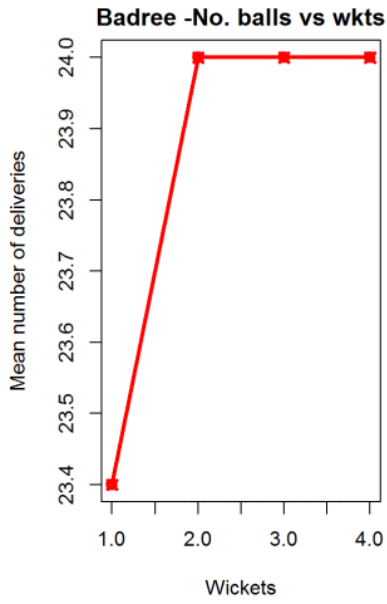
The plot below gives a boxplot of the runs ranges for each of the wickets taken by the bowlers. The ends of the box indicate the 25% and 75% percentile of runs scored for the wickets taken and the dark black line is the average runs conceded.

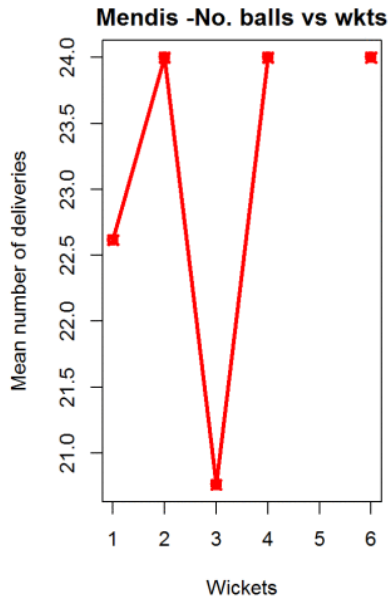
```
par(mfrow=c(1,4))

par(mar=c(4,4,2,2))

bowlerWktsRunsPlot("./badree.csv","Badree")

bowlerWktsRunsPlot("./mendis.csv","Mendis")

bowlerWktsRunsPlot("./narine.csv","Narine")

bowlerWktsRunsPlot("./ashwin.csv","Ashwin")
```



This plot below shows the average number of deliveries needed by the bowler to take the wickets (1,2,3 etc.)

```
par(mfrow=c(1,2))

par(mar=c(4,4,2,2))

bowlerWktRateTT("./badree.csv","Badree")

bowlerWktRateTT("./mendis.csv","Mendis")
```

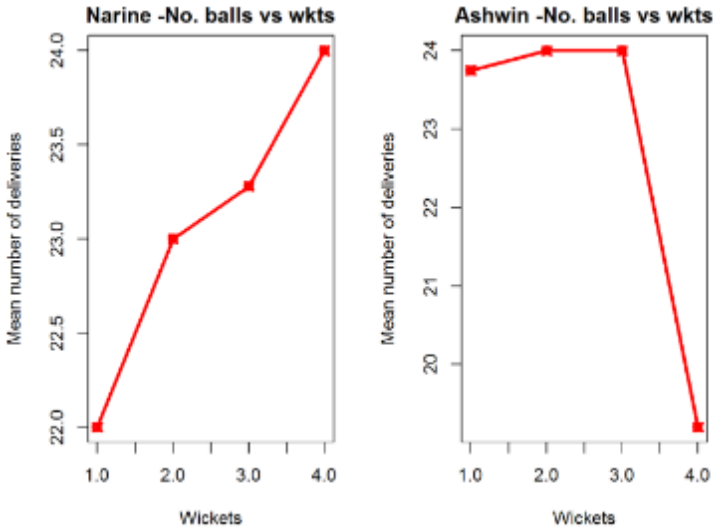





```
dev.off()
```

```
## null device

##           1
```

```
par(mfrow=c(1,2))

par(mar=c(4,4,2,2))

bowlerWktRateTT("./narine.csv","Narine")

bowlerWktRateTT("./ashwin.csv","Ashwin")
```



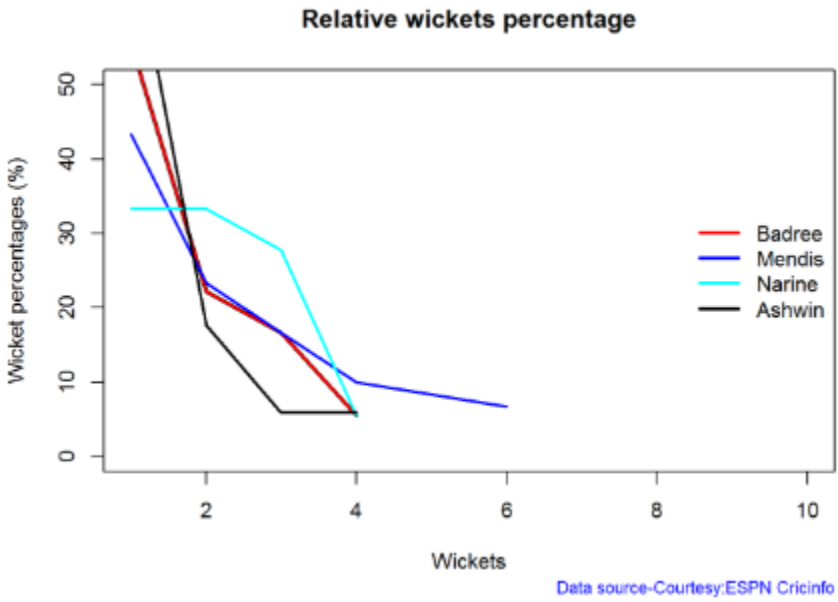

```
dev.off()
```

```
## null device

##           1
```

1.4.14. Relative bowling performance

The plot below shows that Narine has the most wickets in the 2 -4 range followed by Mendis

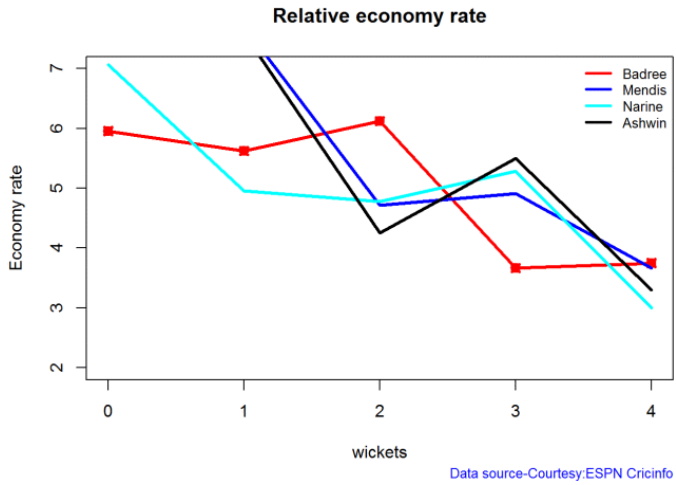
```
frames <- list("./badree.csv","./mendis.csv","narine.csv","ashwin.csv")

names <- list("Badree","Mendis","Narine","Ashwin")

relativeBowlingPerf(frames,names)
```



1.4.15. Relative Economy Rate against wickets taken

The economy rate can be deduced as follows from the plot below. Narine has a good economy rate around 1 & 4 wickets, Ashwin around 2 wickets and Badree around 3. wickets

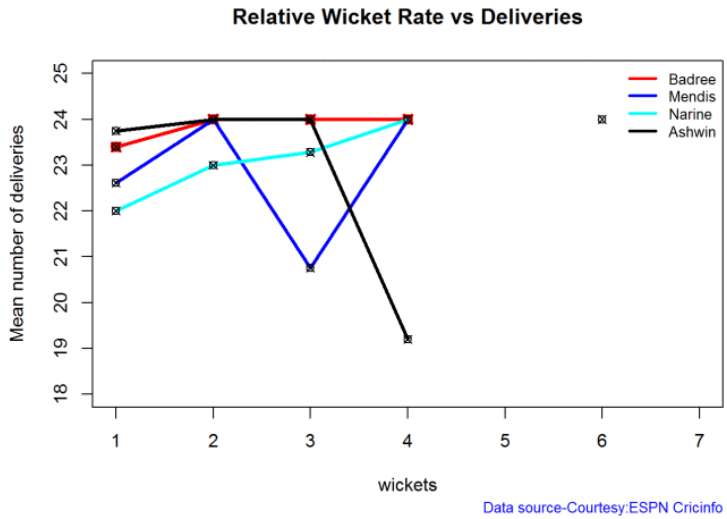
```
frames <- list("./badree.csv", "./mendis.csv", "narine.csv", "ashwin.csv")

names <- list("Badree", "Mendis", "Narine", "Ashwin")

relativeBowlingERODTT(frames, names)
```



1.4.16. Relative Wicket Rate

The relative wicket rate plots the mean number of deliveries needed to take the wickets namely (1,2,3,4). For e.g. Narine needed an average of 22 deliveries to take 1 wicket and 22.5,23.2, 24 deliveries to take 2,3 & 4 wickets respectively

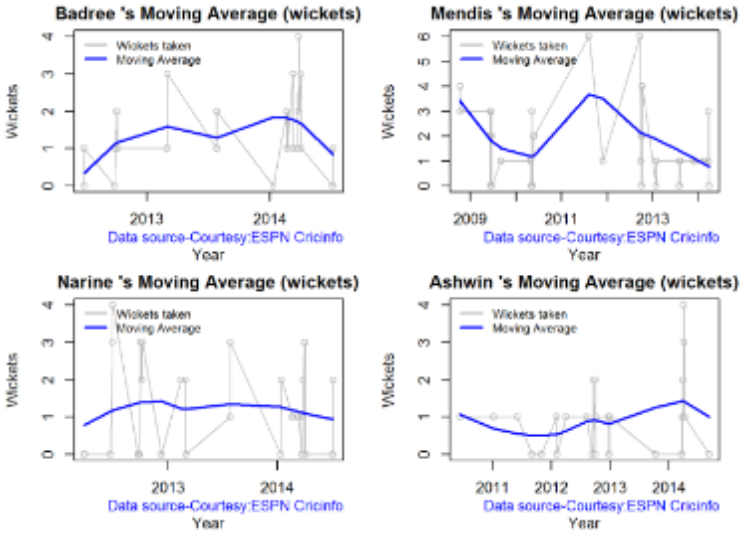
```
frames <- list("./badree.csv","./mendis.csv","narine.csv","ashwin.csv")

names <- list("Badree","Mendis","Narine","Ashwin")

relativeWktRateTT(frames,names)
```



Moving average of wickets over career

```
par(mfrow=c(2,2))

par(mar=c(4,4,2,2))

bowlerMovingAverage("./badree.csv","Badree")

bowlerMovingAverage("./mendis.csv","Mendis")

bowlerMovingAverage("./narine.csv","Narine")

bowlerMovingAverage("./ashwin.csv","Ashwin")
```

```
## null device

##           1
```



1.4.17. Key findings

Here are some key conclusions

1.4.18. Twenty 20 batsmen

1. Kohli has the a very consistent performance scoring high runs in the different run ranges. Kohli also has a 34.2% likelihood to score 6 runs. He is followed by McCullum for consistent performance
2. Finch has a best strike rate followed by McCullum.
3. Du Plessis has the highest percentage of 4s and McCullum has the percentage of 6s. Finch is superior in the percentage of runs scored in 4s and 6s
4. For a hypothetical balls faced and minutes at crease, Finch does best followed by McCullum
5. Kohli's & Du Plessis Twenty20 career is on a upswing. Can they maintain the momentum. McCullum is consistent

1.4.19. Twenty20 bowlers

1. Narine has the highest wickets percentage for different wickets taken followed by Mendis
2. Mendis has taken 1,2,3,4,6 wickets in 24 deliveries
3. Narine has the lowest economy rate for 1 & 4 wickets, Ashwin for 2 wickets and Badree for 3 wickets. Mendis is comparatively expensive
4. Narine needed the least deliveries to get 1 (22.5) & 2 (23.2) wickets, Mendis needed 20.5 deliveries and Ashwin 19 deliveries for 4 wickets

1.4.20. Key takeaways

If all the above batsmen and bowlers were in the same team we expect

1. Finch would be most useful when the run rate has to be greatly accelerated followed by McCullum
2. If the need is to consolidate, then Kohli is the best man for the job followed by McCullum
3. Overall McCullum is the best bet for Twenty20
4. When it comes to bowling Narine wins hands down as he has the most wickets, a good economy rate and a very good attack rate. So Narine is great bet for providing a vital breakthrough.

1.5. Sixer – R package cricketr's new Shiny avatar

In this post I create a Shiny App, Sixer, based on my R package cricketr. I had developed the R package cricketr, a few months back for analyzing the performances of batsman and bowlers in all formats of the game (Test, ODI and Twenty 20). This package uses the statistics info available in ESPN Cricinfo Statsguru. I had written a series of posts using the cricketr package where I chose a few batsmen, bowlers and compared their performances of these players. Here I have created a complete Shiny app with a lot more players and with almost all the features of the cricketr package. The motivation for creating the Shiny app was to

To show case the 'cricketr' package and to highlight its functionalities

Perform analysis of more batsman and bowlers

Allow users to interact with the package and to allow them to try out the different features and functions of the package and to also check performances of some of their favorite crickets

a) You can try out the interactive Shiny app Sixer at
https://tvganesh.shinyapps.io/Sixer/
b) The code for this Shiny app project can be cloned/forked from GitHub – Sixer
Note: Please be mindful of ESPN Cricinfo Terms of Use.

In this Shiny app I have 5 tabs which perform the following function

1.5.1. Analyze batsman

This tab analyzes batsmen based on different functions and plots the performances of the selected batsman. There are functions that compute and display batsman's run-frequency ranges, Mean Strike rate, No of 4's, dismissals, 3-D plot of Runs scored vs Balls Faced and Minutes at crease, Contribution to wins & losses, Home-Away record etc. The analyses can be done for Test cricketers, ODI and Twenty 20 batsman. I have included most of the Test batting giants including Tendulkar, Dravid, Sir Don Bradman, Viv Richards, Lara, Ponting etc. Similarly the ODI list includes Sehwag, Devilliers, Afridi, Maxwell etc. The Twenty20 list includes the Top 10 Twenty20 batsman based on their ICC rankings

1.5.2. Analyze bowler

This tab analyzes the bowling performances of bowlers, Wickets percentages, Mean Economy Rate, Wickets at different venues, Moving average of wickets etc. As earlier I have all the Top bowlers including Warne, Muralidharan, Kumble- the famed Indian spin quartet of Bedi, Chandrasekhar, Prasanna, Venkatraghavan, the deadly West Indies trio of Marshal, Roberts and Holding and the lethal combination of Imran Khan, Wasim Akram and Waqar Younis besides the dangerous Dennis Lillie and Jeff Thomson. Do give the functions a try and see for yourself the performances of these individual bowlers

1.5.3. Relative performances of batsman

This tab allows the selection of multiple batsmen (Test, ODI and Twenty 20) for comparisons. There are 2 main functions Relative Runs Frequency performance and Relative Mean Strike Rate

1.5.4. Relative performances of bowlers

Here we can compare bowling performances of multiple bowlers, which include functions Relative Bowling Performance and Relative Economy Rate. This can be done for Test, ODI and Twenty20 formats

1.5.5. Check for In-Form status of players

This tab checks the form status of batsman or bowler selected for all of the different formats of the game. The below computation uses Null Hypothesis testing and p-value to determine if the batsman is in-form or out-of-form. For this 90% of the career runs is chosen as the population and the mean computed. The last 10% is chosen to be the sample set and the sample Mean and the sample Standard Deviation are calculated. **Note**: The accuracy of the p-value test depends on the size of the population and

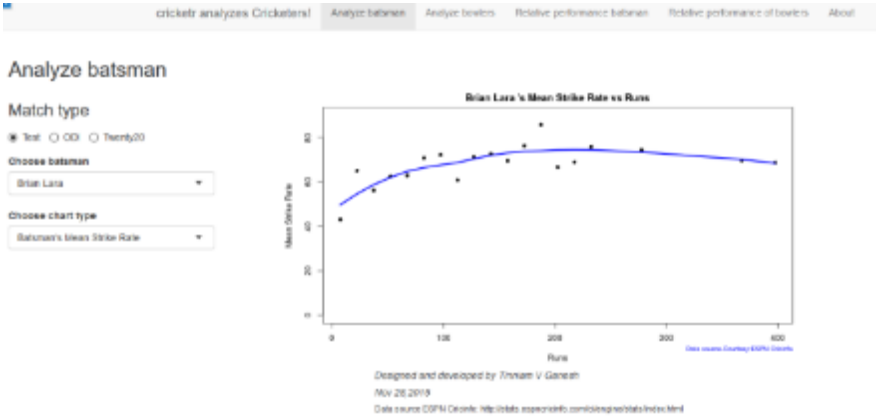
the size of the sample set. It may not be very significant for players with a few innings played.

You can clone the code from Github – https://github.com/tvganesh/Sixer

There is not much in way of explanation. The Shiny app's use is self-explanatory. You can choose a match type (Test,ODI or Twenty20), choose a batsman/bowler from the drop down list and select the plot you would like to seeHere a few sample plots

1.5.6. Analyze batsman tab

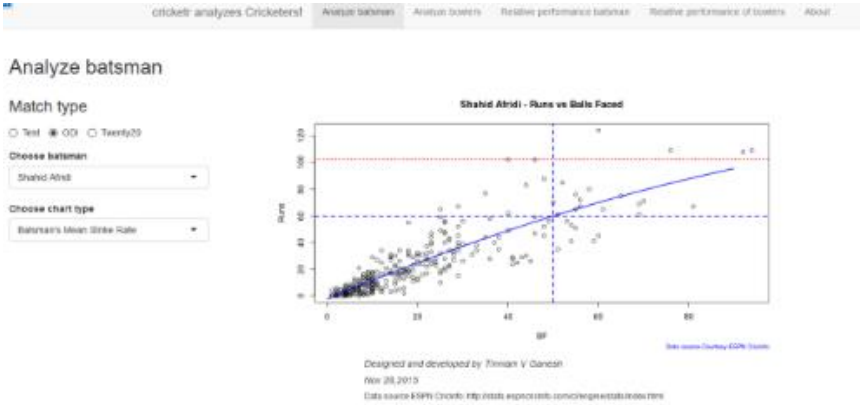
i) Batsman – Brian Lara , Match Type – Test, Function – Mean Strike Rate



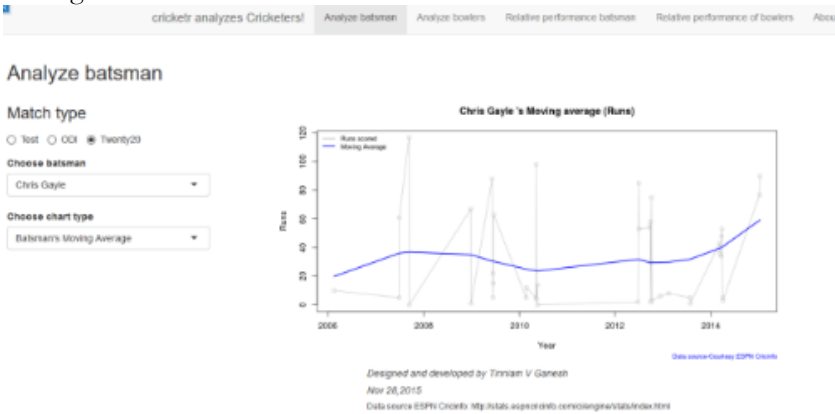
ii) Batsman – Shahid Afridi, Match Type – ODI, Function – Runs vs Balls faced

The plot below shows that if Afridi faces around 50 balls he is likely to

score around 60 runs in ODIs.

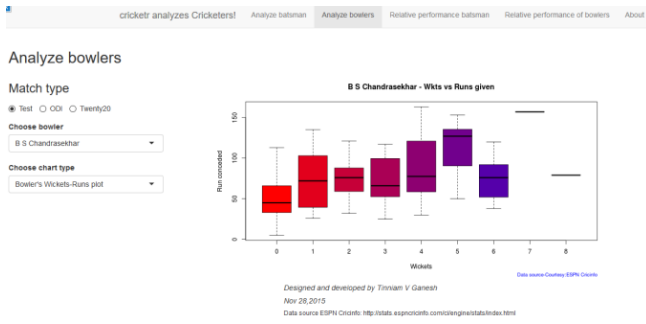


iii) Batsman – Chris Gayle, Match Type – Twenty20 Function – Moving Average

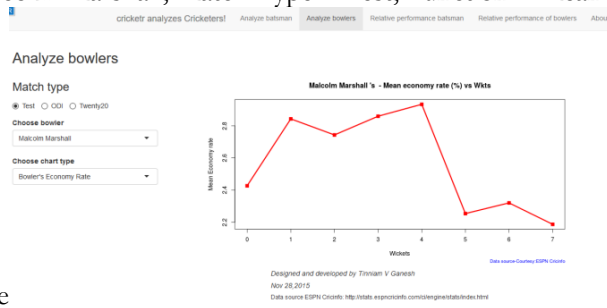


1.5.7. **Analyze bowler tab**

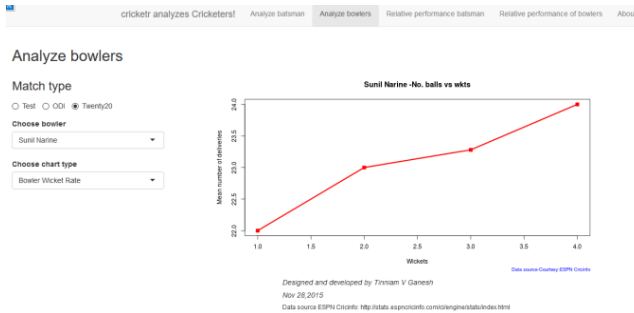
Bowler – B S Chandrasekhar, Match Type – Test, Function – Wickets vs Runs



ii) Bowler – Malcolm Marshall, Match Type – Test, Function – Mean Economy Rate

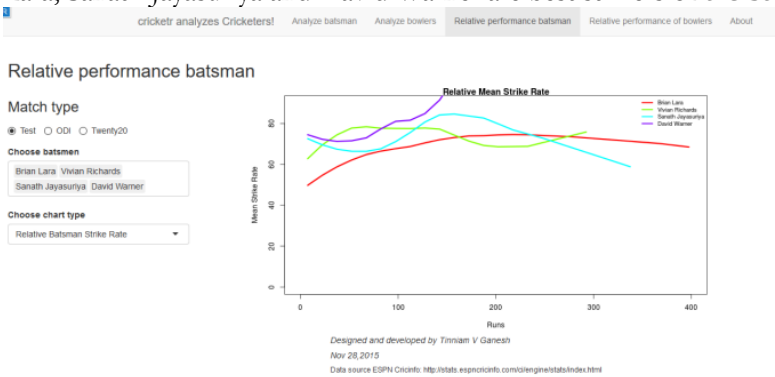


iii) Bowler – Sunil Narine, Match Type – Twenty 20, Function – Bowler Wicket Rate

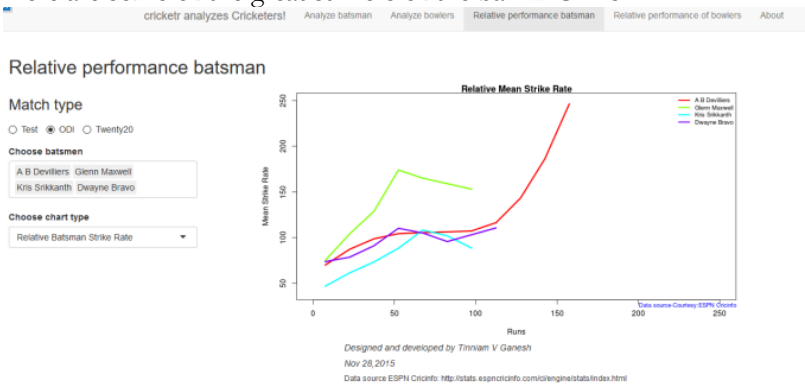


1.5.8. **Relative performance of batsmen (select more than 1)**

The below plot gives the Mean Strike Rate of batsman. Viv Richards, Brian Lara, Sanath Jayasuriya and David Warner are best strikers of the ball.



Here are some of the great strikers of the ball in ODIs

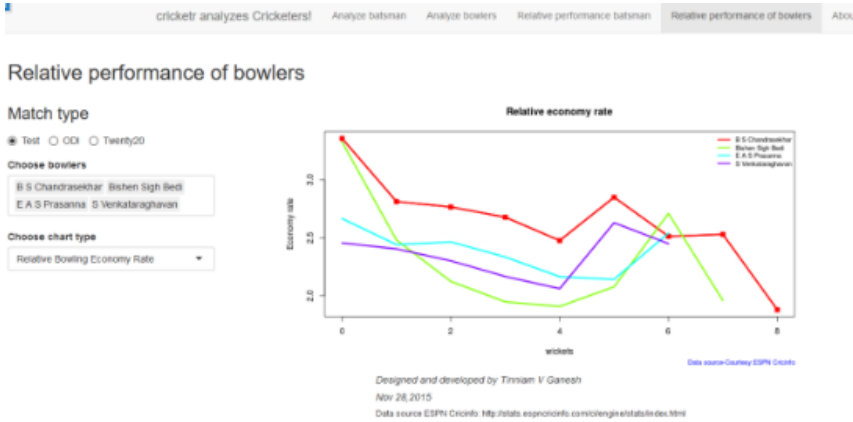


D.

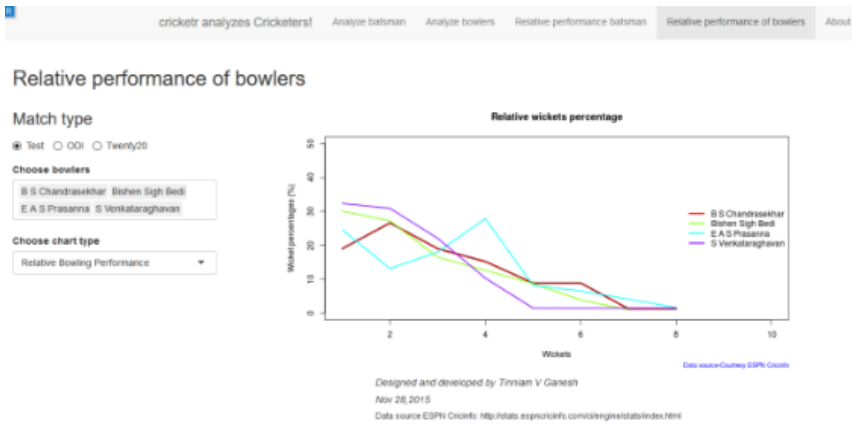
1.5.9. **Relative performance of bowlers (select more than 1)**

Finally a look at the famed Indian spin quartet. From the plot below it can be seen that B S Bedi & Venkatraghavan were more economical than

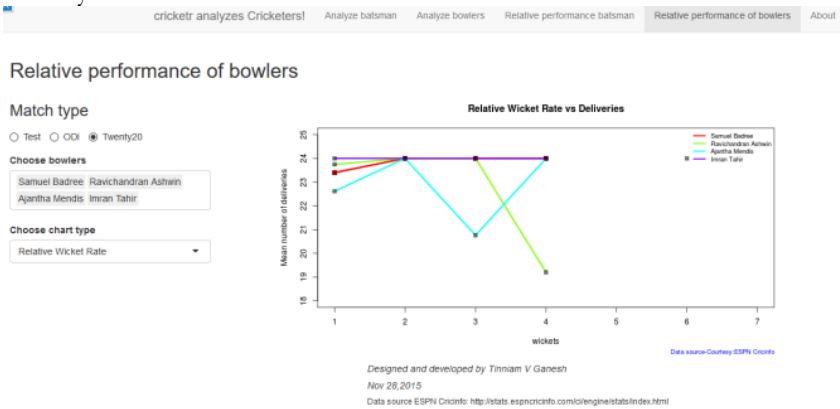
Chandrasekhar and Prasanna.



But the latter have a better 4-5 wicket haul than the former two as seen in the plot below



Finally a look at the average number of balls to take a wicket by the Top 4 Twenty 20 bowlers.



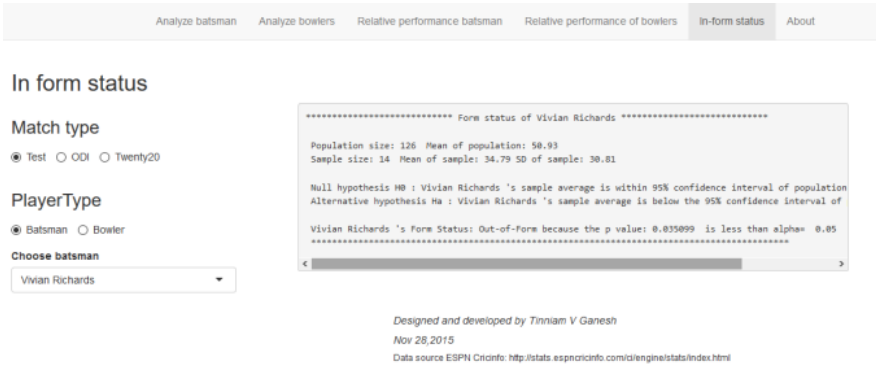
1.5.10. **Check in-form status of players**

Note: The accuracy of the p-value depends on the size of the population and the size of the sample set. It may not be very significant for smaller data sizes

i. Match Type – Test, Player Type – Batsman Name – Wickets v

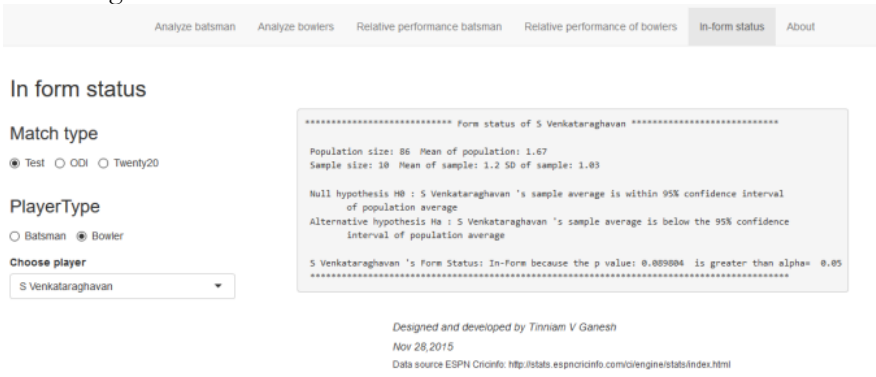
In this plot the in-form status of Viv Richards is checked. This shows that

Viv Richards was out-of-form



In the plot below the form status of S. Venkatraghavan is shown.

According to this at the time of his retirement S Venkat was still in-form



Do give the Shiny app Sixer a try.

2. Other cricket posts in R

2.1. Analyzing cricket's batting legends – Through the mirage with R

In this post I do a deep dive into the records of the all-time batting legends of cricket to identify interesting information about their achievements. In my opinion, the usual currency for batsman's performance like most number of centuries or highest batting average are too gross in their significance. I wanted something finer where we can pin-point specific strengths of different players

This post will answer the following questions.

– How many times has a batsman scored runs in a specific range say 20-40 or 80-100 and so on?

– How do different batsmen compare against each other?

– Which of the batsmen stayed well beyond their sell-by date?

– Which of the batsmen retired too soon?

– What is the propensity for a batsman to get caught, bowled run out etc?

For this analysis I have chosen the batsmen below for the following reasons

Sir Don Bradman : With a batting average of 99.94 Bradman was an obvious choice

Sunil Gavaskar is one of India's batting icons who amassed 774 runs in his debut against the formidable West Indies in West Indies

Brian Lara : A West Indian batting hero who has double, triple and quadruple centuries under his belt

Sachin Tendulkar: A prolific run getter, India's idol, who holds the record for most test centuries by any batsman (51 centuries)

Ricky Ponting:A dangerous batsman against any bowling attack and who can demolish any bowler on his day

Rahul Dravid: He was India's most dependable batsman who could weather any storm in a match single-handedly

AB De Villiers : The destructive South African batsman who can pulverize any attack when he gets going

The analysis has been performed on these batsmen on various parameters. Clearly different batsmen have shone in different batting aspects. The analysis focuses on each of these to see how the different players stack up against each other.

The data for the above batsmen has been taken from ESPN Cricinfo. Only the batting statistics of the above batsmen in Test cricket has been taken. The implementation for this analysis has been done using the R language.

The R implementation, datasets and the plots can be accessed at GitHub at analyze-batting-legends. Feel free to fork or clone the code. You should be able to use the code with minor modifications on other players. Also go ahead make your own modifications and hack away!

Key insights from my analysis below

a) Sir Don Bradman's unmatchable record of 99.94 test average with several centuries, double and triple centuries makes him the gold standard of test batting as seen in the 'All-time best batsman below'

b) Sunil Gavaskar is the king of batting in India, followed by Rahul Dravid and finally Sachin Tendulkar. See the charts below for details

c) Sunil Gavaskar and Rahul Dravid had at least 2 more years of good test cricket in them. Their retirement was premature. This is based on the individual batsmen's career graph (moving average below)

d) Brian Lara, Sachin Tendulkar, Ricky Ponting, Vivian Richards retired at a time when their batting was clearly declining. The writing on the wall was clear and they had to go (see moving average below)

e) The biggest hitter of 4's was Vivian Richards. In the 2nd place is Brian Lara. Tendulkar & Dravid follow behind. Dravid is a surprise as he has the image of a defender.

e) While Sir Don Bradman made huge scores, the number of 4's in his innings was significantly less. This could be because the ground in those days did not carry the ball far enough

f) With respect to dismissals Richards was able to keep his wicket intact (11%) of the times , followed by Ponting Tendulkar, De Villiers, Dravid (10%) who carried the bat, and Gavaskar & Bradman (7%)

A) Runs frequency table and charts

These plots normalize the batting performance of different batsman, since the number of innings played ranges from 89 (Bradman) to 348 (Tendulkar), by calculating the percentage frequency the batsman scores runs in a particular range. For e.g. Sunil Gavaskar made scores between 60-80 10% of his total innings

This is shown in a tabular form below

Name	0-20	20-40	40-60	60-80	80-100	100-120	120-140	140-160	160-180	180-200	200-220	220-240	240-260	260-280	280-300
Vivian Richards	33.7	20.3	11.6	**18**	2.3	**5.8**	**2.3**	1.7	1.2	1.2	0.6	0.6	0	0	0.6
AB De Villiers	35.7	20.4	**14.6**	**10.2**	**6.4**	5.1	1.9	1.3	3.2	0	0.6	0	0	0.6	0
Brian Lara	37.7	14.9	**15.8**	8.8	**7.4**	3.3	**2.3**	2.3	1.9	1.4	1.9	0.9	0	0.5	0
Sachin Tendulkar	39.5	17.2	11.5	8.9	7.3	4.8	**2.5**	3.5	2.2	0.6	1.3	0	0.6	0	0
Sunil Gavaskar	37.3	15.9	14.4	**10**	5.5	**7.5**	3	1.5	2.5	0.5	1	1	0	0	0
Rahul Dravid	35.3	22.7	12.9	9	**7.6**	4.7	1.4	2.9	1.1	1.1	0.4	0.7	0	0.4	0
Ricky Ponting	35.8	18.3	**16.4**	9.3	5.2	4.9	2.2	4.5	0.4	0.7	1.1	0.4	0.7	0	0
Sir Don Bradman	20.5	17.8	9.6	9.6	2.7	**6.8**	**6.8**	2.7	4.1	2.7	2.7	5.5	2.7	1.4	1.4

The individual charts for each of the players are shown below. The top performers after removing ranges 0-20 & 20-40 are

Between 40-60 runs – 1) Ricky Ponting (16.4%) 2) Brian Lara (15.8%) 3) AB De Villiers (14.6%)

Between 60-80 runs – 1) Vivian Richards (18%) 2) AB De Villiers (10.2%) 3) Sunil Gavaskar (10%)

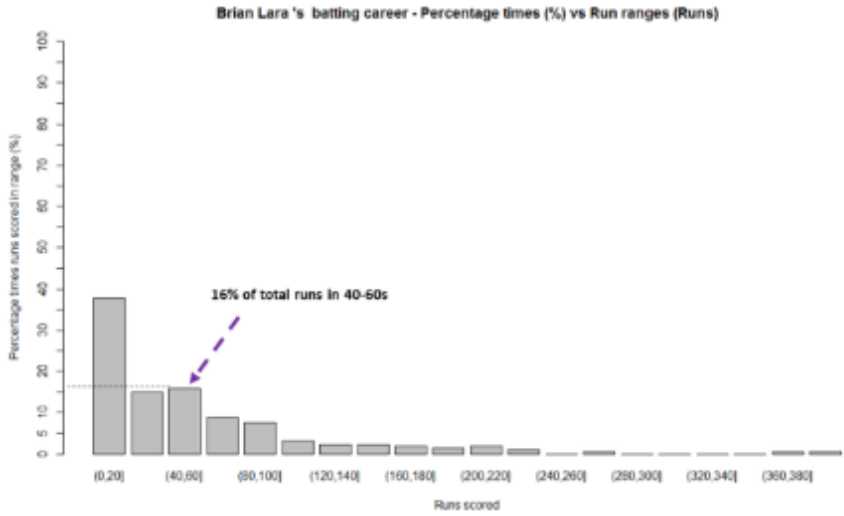
Between 80-100 runs – 1) Rahul Dravid (7.6%) 2) Brian Lara (7.4%) 3) AB De Villiers (6.4%)

Between 100 -120 runs – 1) Sunil Gavaskar (7.5%) 2) Sir Don Bradman (6.8%) 3) Vivian Richards (5.8%)

Between 120-140 runs – 1) Sir Don Bradman (6.8%) 2) Sachin Tendulkar (2.5%) 3) Vivian Richards (2.3%)

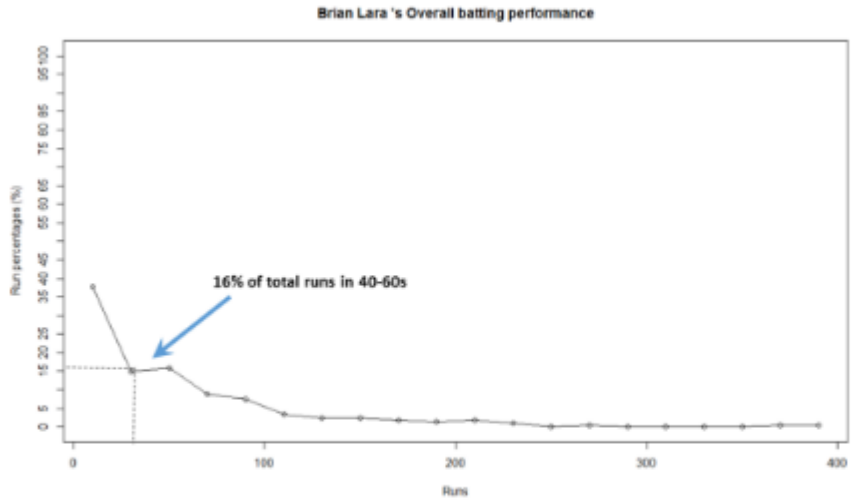
The percentage frequency for Brian Lara is included below

1) Brian Lara



The above chart shows out of the total number of innings played by Brian Lara he scored runs in the range (40-60) 16% percent of the time. The chart also shows that Lara scored between 0-20, 40% while also scoring in the ranges 360-380 & 380-400 around 1%.

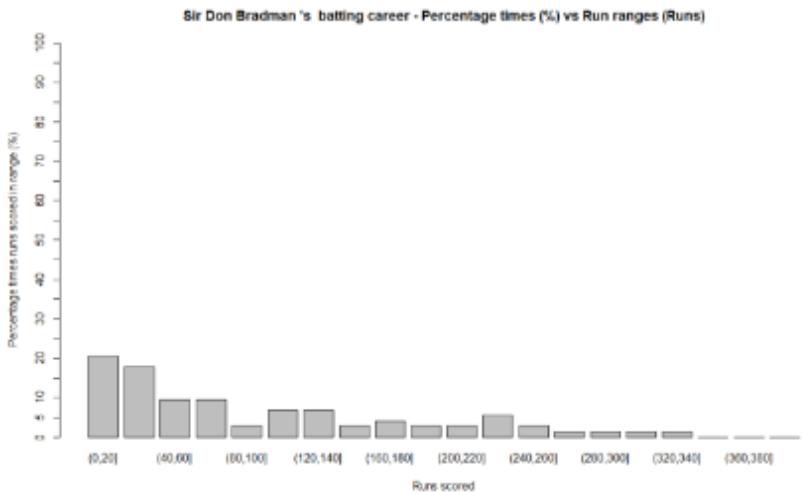
The same chart is displayed as continuous graph below



The run frequency charts for other batsman are

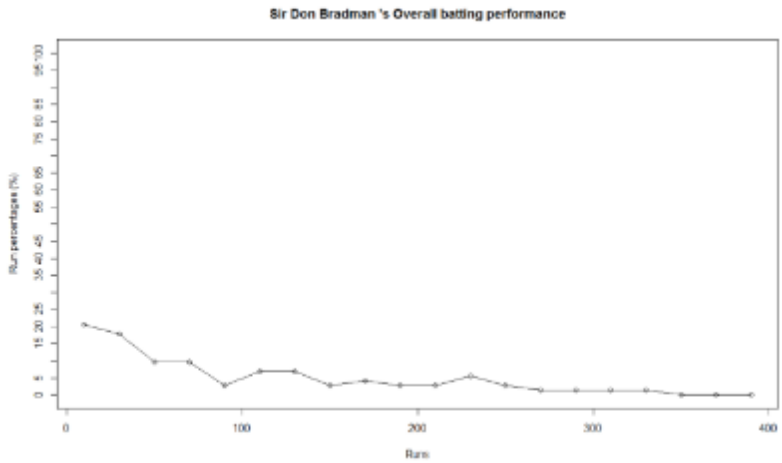
2) Sir Don Bradman

a) Run frequency



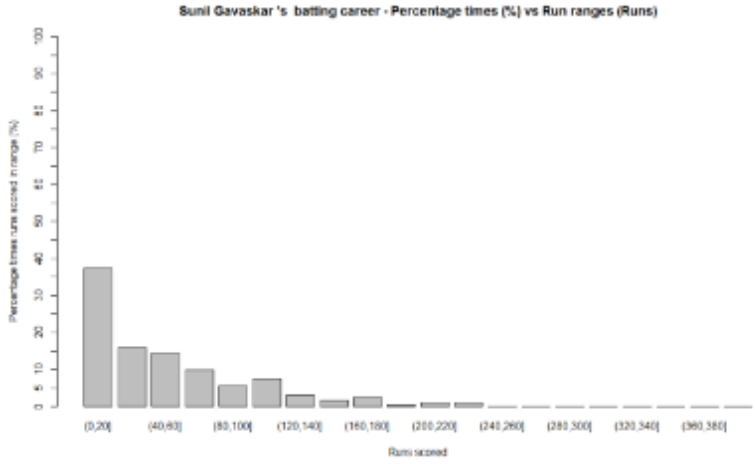
Note: Notice the significant contributions by Sir Don Bradman in the ranges 120-140,140-160,220-240,all the way up to 340

b) Performance

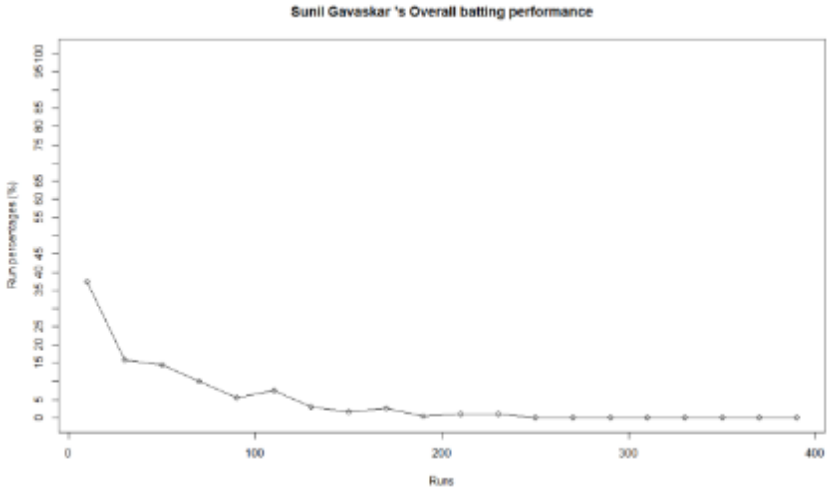


3) Sunil Gavaskar

a) Runs frequency chart

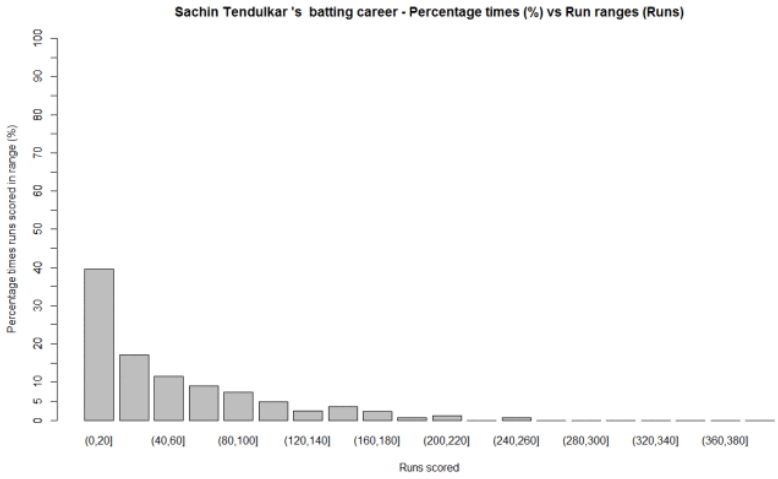


b) Performance chart

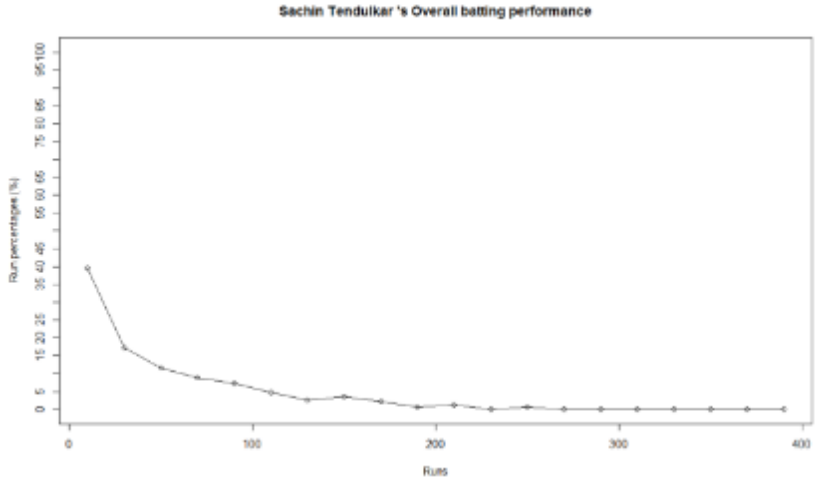


4) Sachin Tendulkar

a) Runs frequency chart

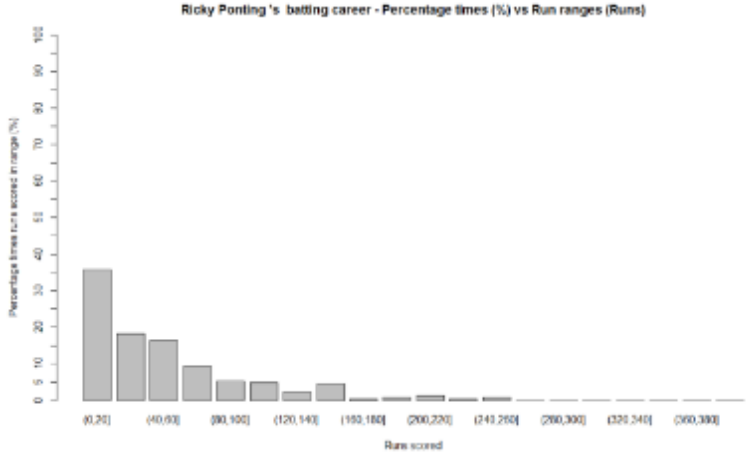


b) Performance chart

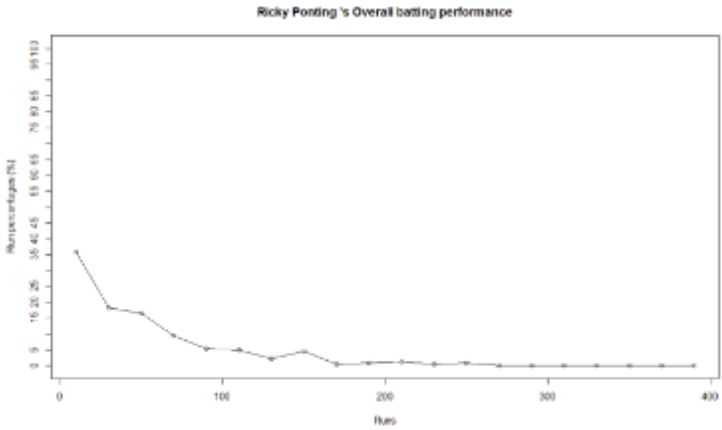


5) Ricky Ponting

a) Runs frequency

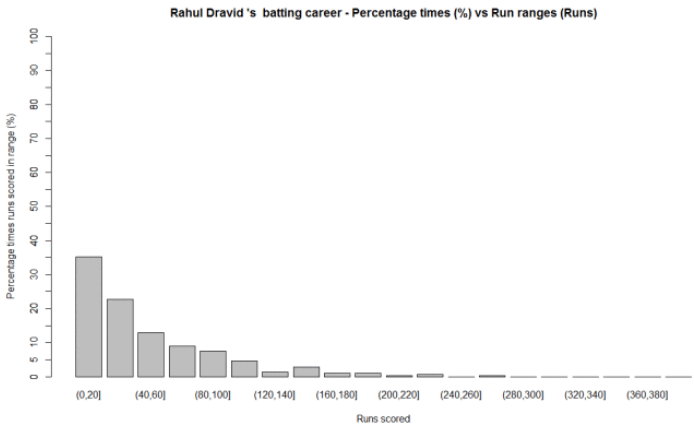


b) Performance

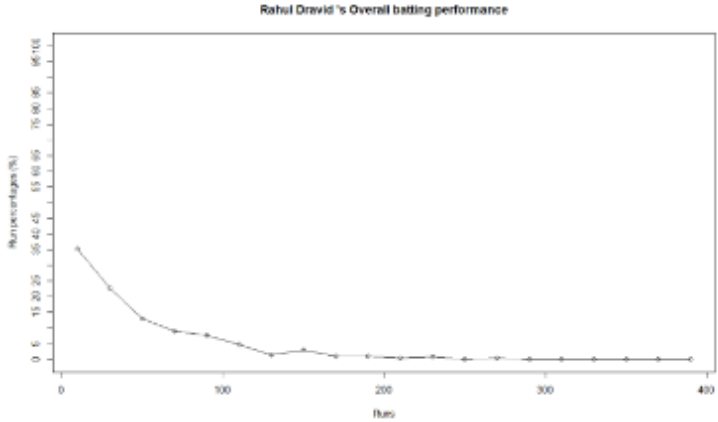


6) Rahul Dravid

a) Runs frequency chart

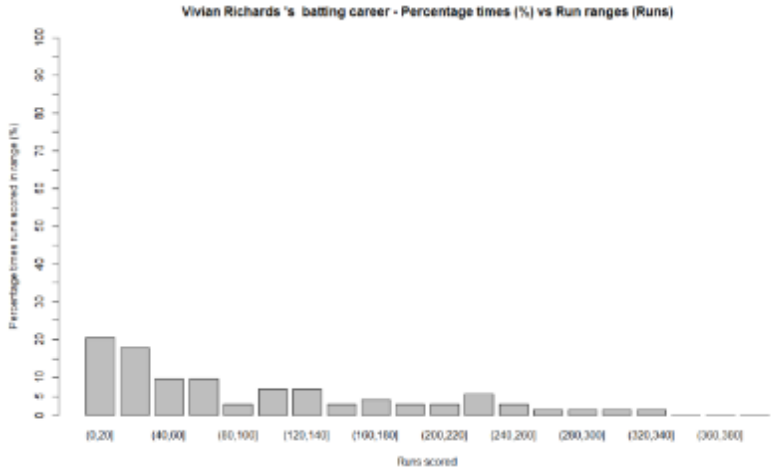


b) Performance chart

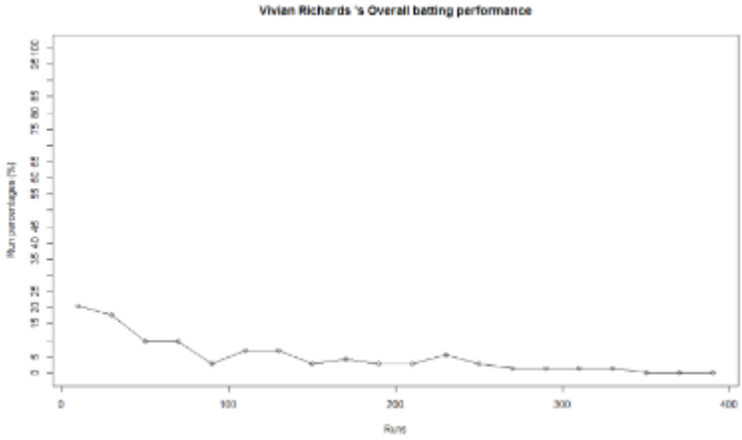


7) Vivian Richards

a) Runs frequency chart

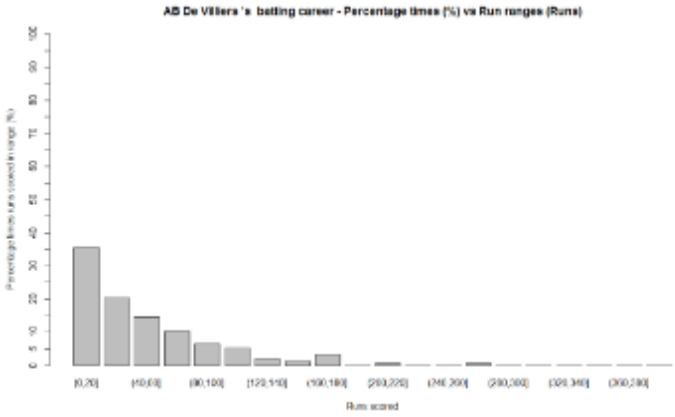


b) Performance chart

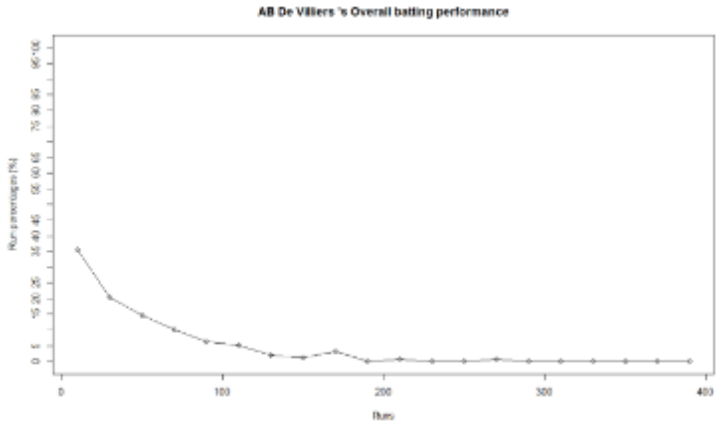


8) AB De Villiers

a) Runs frequency chart



b) Performance chart

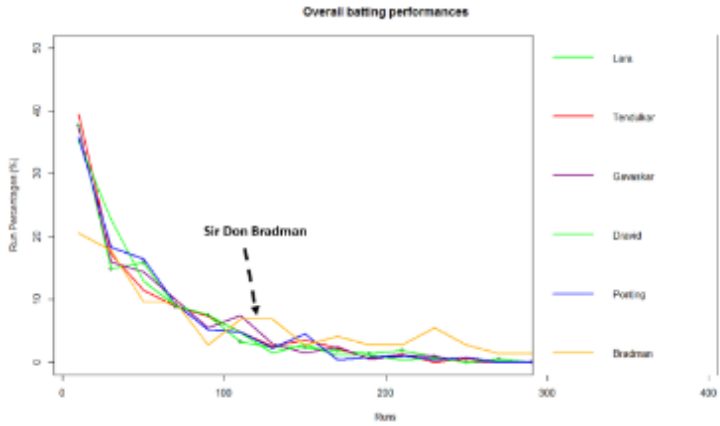


B) Relative performance of the players

In this section I try to measure the relative performance of the players by superimposing the performance graphs obtained above. You may say that "comparisons are odious!". But equally odious are myths that are based on gross facts like highest runs, average or most number of centuries.

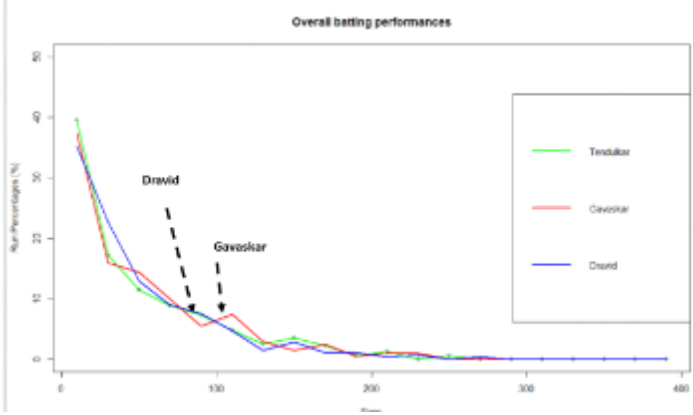
a) All-time best batsman

(Sir Don Bradman, Sunil Gavaskar, Vivian Richards, Sachin Tendulkar, Ricky Ponting, Brian Lara, Rahul Dravid, AB De Villiers)



From the above chart it is clear that Sir Don Bradman is the 'gold' standard in batting. He is well above others for run ranges above 100 – 350

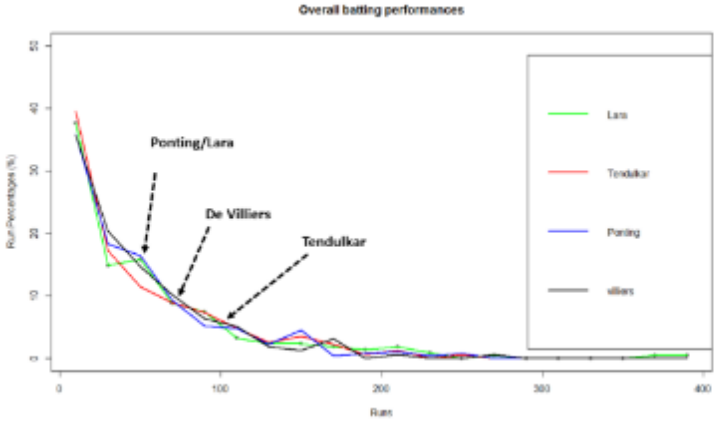
b) Best Indian batsman (Sunil Gavaskar, Sachin Tendulkar, Rahul Dravid)



The above chart shows that Gavaskar is ahead of the other two for key ranges between 100 – 130 with almost 8% contribution of total runs. This followed by Dravid who is ahead of Tendulkar in the range 80-120. According to me the all-time best Indian batsman is 1) Sunil Gavaskar 2) Rahul Dravid 3) Sachin Tendulkar

c) Best batsman -(Brian Lara, Ricky Ponting, Sachin Tendulkar, AB De Villiers)

This chart was prepared since this comparison was often made in recent times



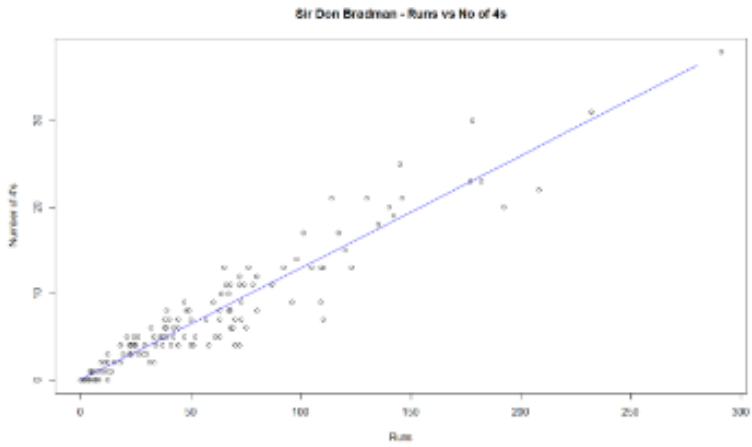
This chart shows the following ranking 1) AB De Villiers 2) Sachin Tendulkar 3) Brian Lara/Ricky Ponting

C) Chart of 4's

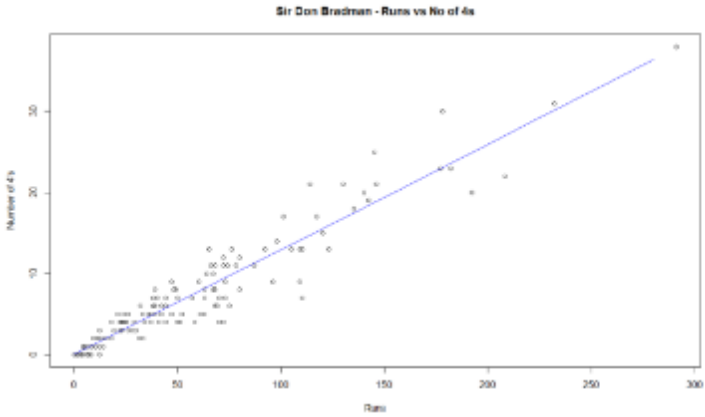
Name	0	20	40	60	80	100	120	140	160	180	200
Vivian Richards	0	3	5	8	10	13	16	18	21	23	26
AB De Villiers	0	3	5	7	10	12	13	15	17	18	19
Brian Lara	0	3	5	8	11	13	16	18	21	23	26
Sachin Tendulkar	0	3	5	8	10	13	15	18	20	23	25
Sunil Gavaskar	0	2	5	7	10	12	14	16	18	20	22
Rahul Dravid	0	2	5	8	10	13	15	18	20	23	25
Ricky Ponting	0	2	5	7	9	11	14	16	18	20	23
Sir Don Bradman	1	2	4	5	7	9	11	13	15	17	20

This chart is plotted with a 2nd order curve of the number of 4's versus the total runs in the innings

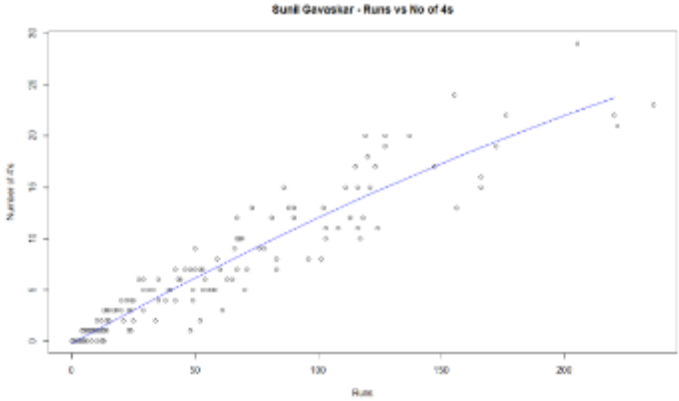
1) Brian Lara



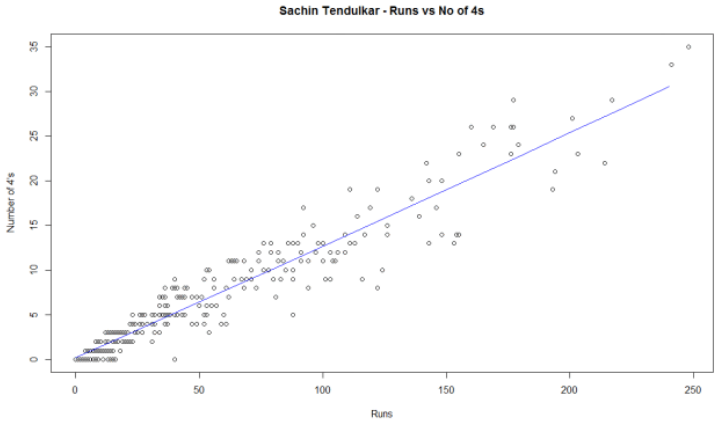
2) Sir Don Bradman



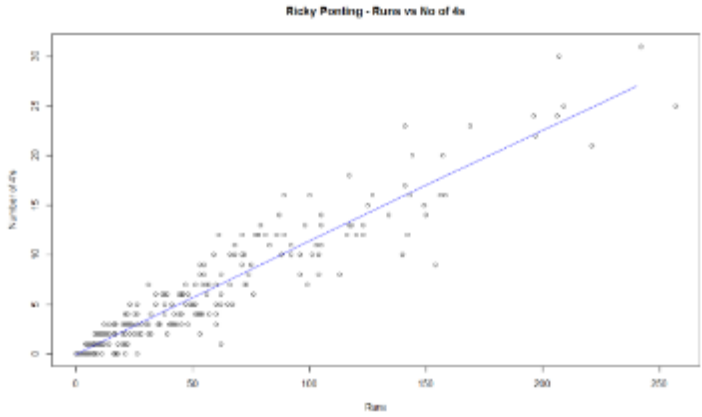
3) Sunil Gavaskar



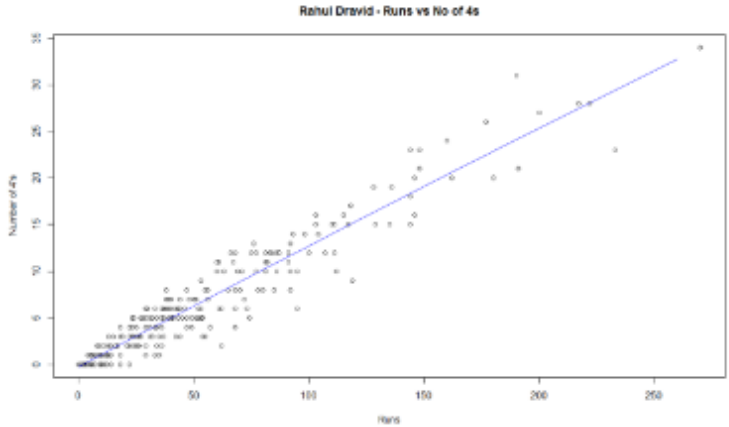
4) Sachin Tendulkar



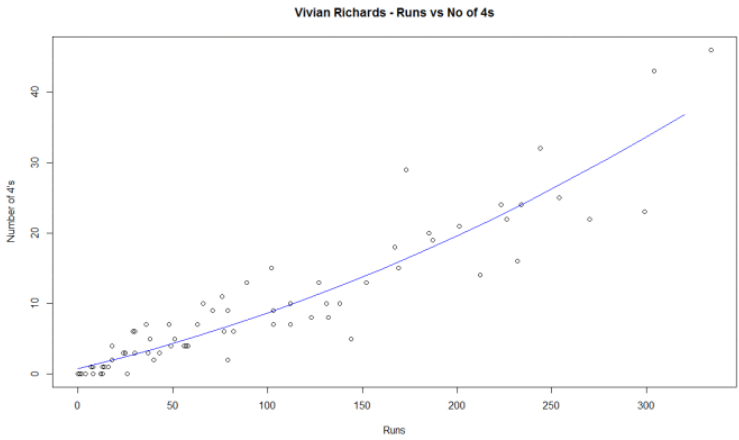
5) Ricky Ponting



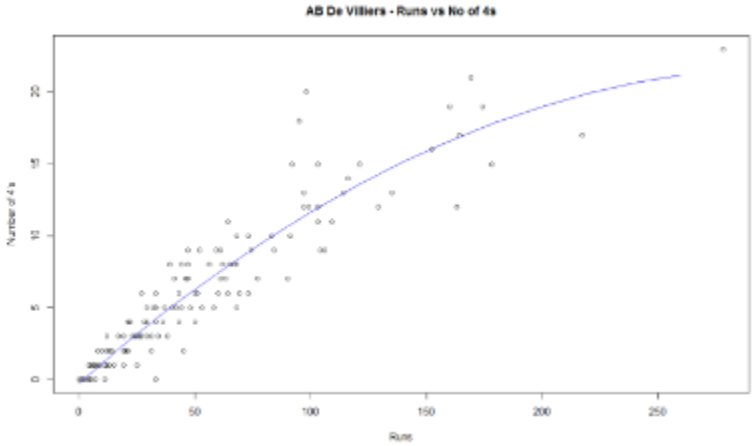
6) Rahul Dravid



7) Vivian Richards



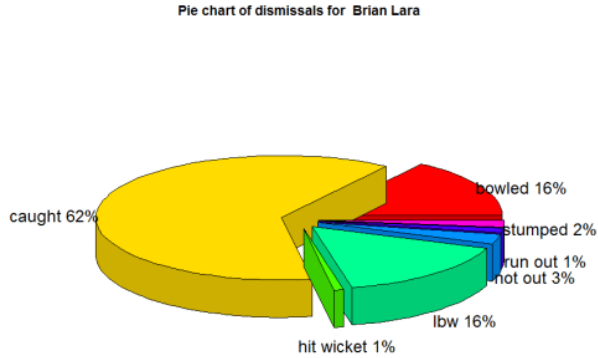
8) AB De Villiers



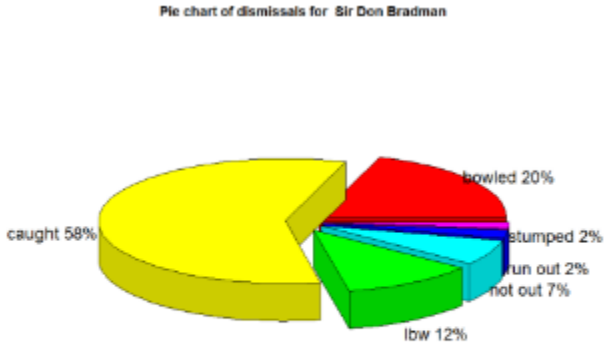
D) Proclivity for type of dismissal

The below charts show how often the batsman was out bowled, caught, run out etc

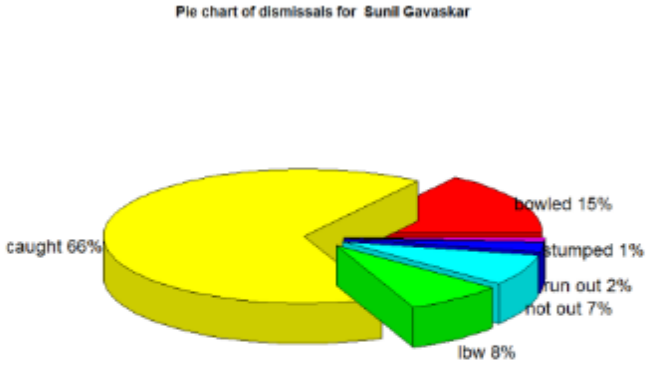
1) Brian Lara



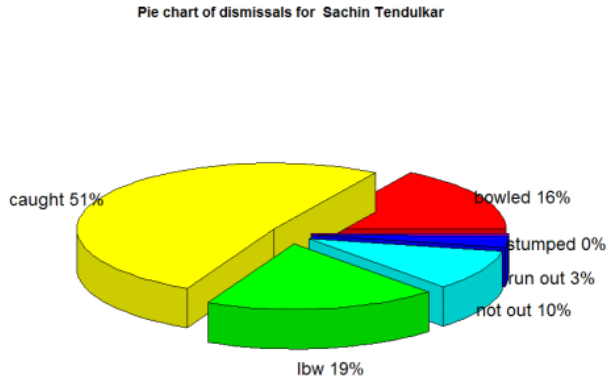
2) Sir Don Bradman



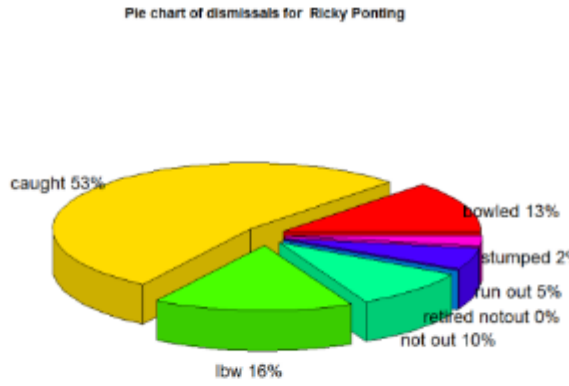
3) Sunil Gavaskar



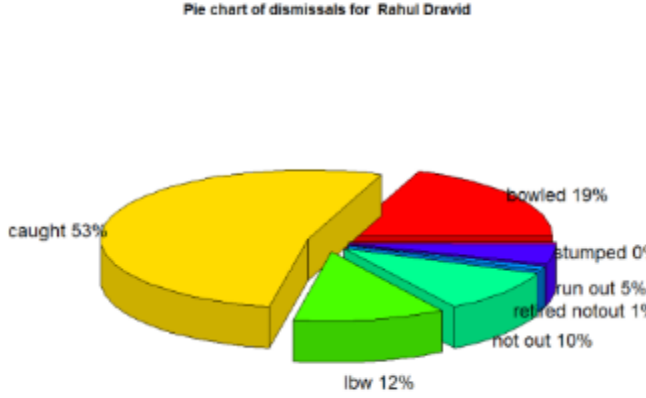
4) Sachin Tendulkar



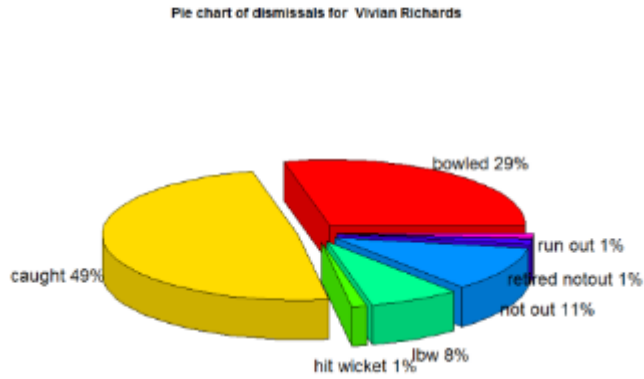
5) Ricky Ponting



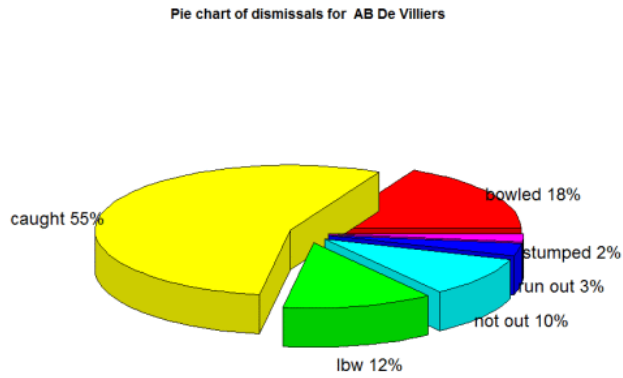
6) Rahul Dravid



7) Vivian Richard



8) AB De Villiers

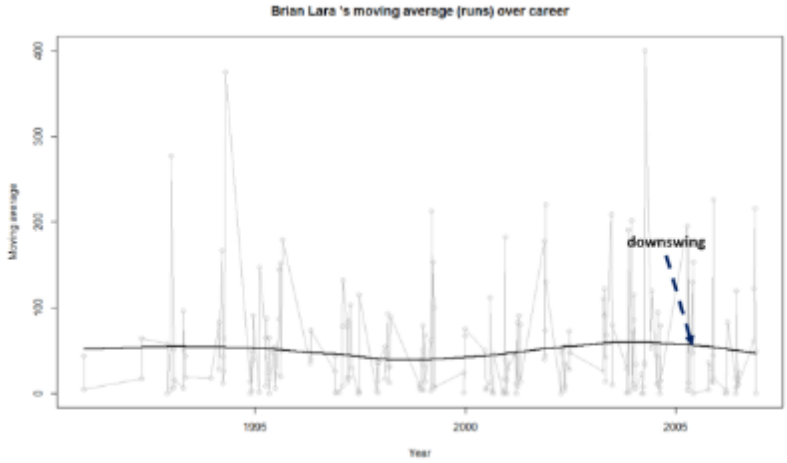


E) Moving Average

The plots below provide the performance of the batsman as a time series (chronological) and is displayed as the continuous gray lines. A moving average is computed using 'loess regression' and is shown as the dark line. This dark line represents the players performance improvement or decline.

The moving average plots are shown below

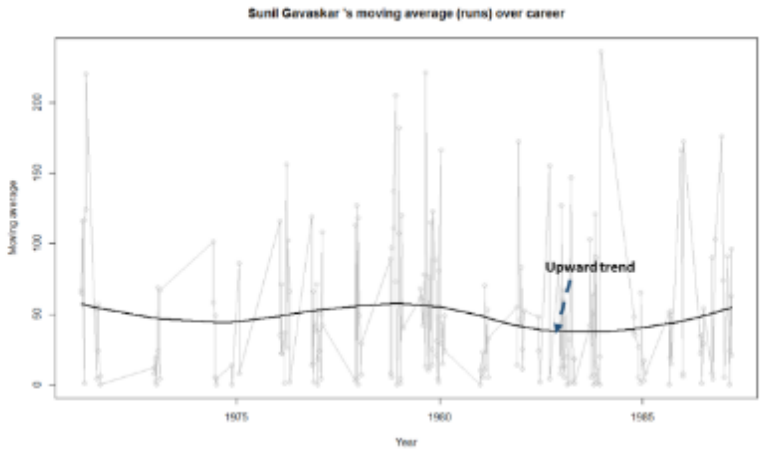
1) Brian Lara



2) Sir Don Bradman

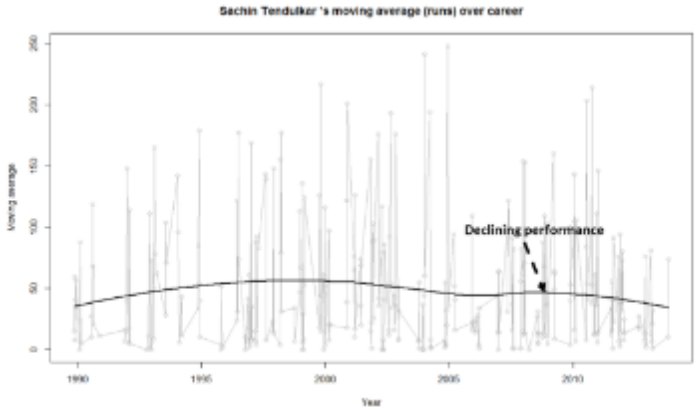
Sir Don Bradman's moving average shows a remarkably consistent performance over the years. He probably could have a continued for a couple more years

3)Sunil Gavaskar



Gavaskar moving average does show a good improvement from a dip around 1983. Gavaskar retired bowing to public pressure on a mistaken belief that he was under performing. Gavaskar could have a continued for a couple of more years

4) Sachin Tendulkar

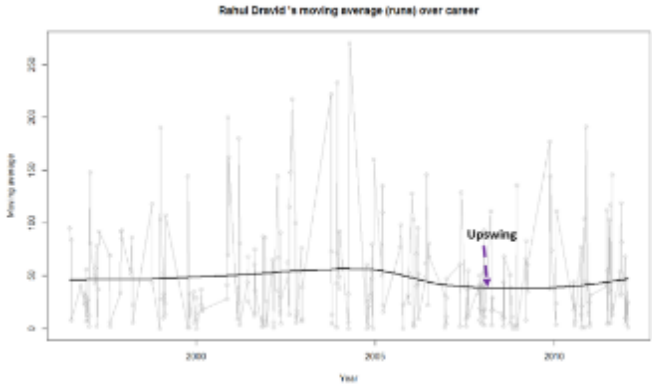


Tendulkar's performance is clearly on the decline from 2011. He could have announced his retirement at least 2 years prior

5) Ricky Ponting

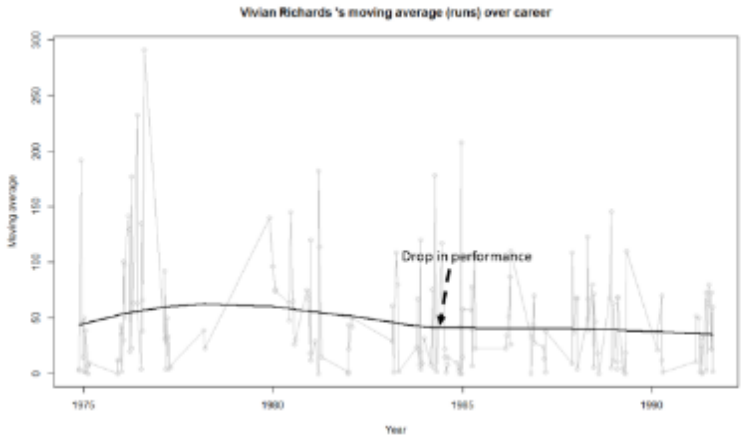
Ponting peak performance was around 2005 and does go steeply downward from then on. Ponting could have also retired around 2012

6) Rahul Dravid



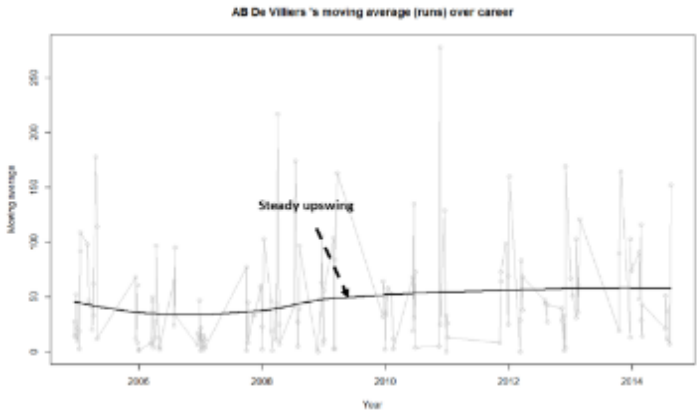
Dravid seems to have recovered very effectively from his poor for around 2009. His overall performance shows steady improvement. Dravid's announcement appeared impulsive. Dravid had another 2 good years of test cricket in him

7) Vivian Richards



Richard's performance seems to have dropped around 1984 and seems to remain that way.

8) AB De Villiers



AB De Villiers moving average shows a steady upward swing from 2009 onwards. De Villiers has at least 3-4 years of great test cricket ahead of him.

Finally as mentioned above the dataset, the R implementation and all the charts are available at GitHub at https://github.com/tvganesh/analyze-batting-legends. Feel free to fork and clone the code. The code should

work for other batsman as-is. Also go ahead and make any modifications for obtaining further insights.

Conclusion: The batting legends have been analyzed from various angles namely i) What is the frequency of runs scored in a particular range ii) How each batsman compares with others for relative runs in a specified range iii) How does the batsman get out? iv) What were the peak and lean period of the batsman and whether they recovered or slumped from these periods. While the batsman themselves have played in different time periods I think in an overall sense the performance under the conditions of the time will be similar.

Anyway feel free to let me know your thoughts. If you see other patterns in the data also do drop in your comment.

2.2. Mirror, mirror … the best batsman of them all?

"Full many a gem of purest serene
The dark oceans of cave bear."

Thomas Gray – Elegy in country churchyard

In this post I do a fine grained analysis of the batting performances of cricketing icons from India and also from the international scene to determine how they stack up against each other. I perform 2 separate analyses 1) Between Indian legends (Sunil Gavaskar, Sachin Tendulkar & Rahul Dravid) and another 2) Between contemporary cricketing stars (Brian Lara, Sachin Tendulkar, Ricky Ponting and A B De Villiers)

In the world and more so in India, Tendulkar is probably placed on a higher pedestal than all other cricketers. I was curious to know how much of this adulation is justified. In "Zen and the art of motorcycle maintenance" Robert Pirsig mentions that while we cannot define Quality (in a book, music or painting) we usually know it when we see it. So do the people see an ineffable quality in Tendulkar or are they intuiting his greatness based on overall averages?

In this context, we need to keep in mind the warning that Daniel Kahnemann highlights in his book, 'Thinking fast and slow'. Kahnemann suggests that we should regard "statistical intuition with proper suspicion and replace impression formation by computation wherever possible". This is because our minds usually detects patterns and associations even when none actually exist.

So this analysis tries to look deeper into these aspects by performing a detailed statistical analysis.

The data for all the batsman has been taken from ESPN Cricinfo. The data is then cleaned to remove 'DNB' (did not bat), 'TDNB' (Team did not bat) etc. before generating the graphs.

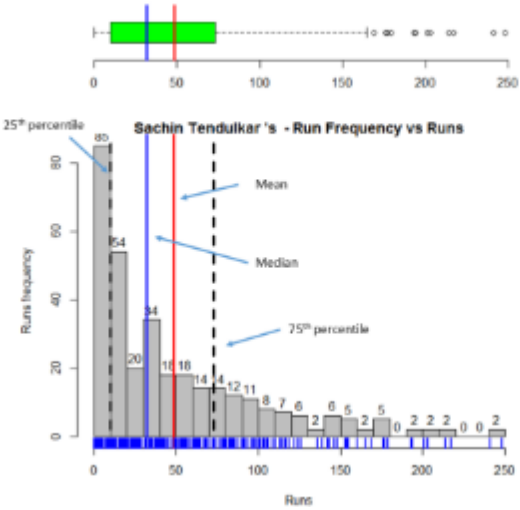
The code, data and the plots can be cloned, forked from Github at the following link https://github.com/tvganesh/bestBatsman. You should be able to use the code as-is for any other batsman you choose to.

Feel free to agree, disagree, dispute or argue with my analysis.

The batting performances of the each of the cricketers is described in 3 plots a) Combined boxplot & histogram b) Runs frequency vs Runs plot c) Mean Strike Rate vs Runs plot

A) Batting performance of Sachin Tendulkar

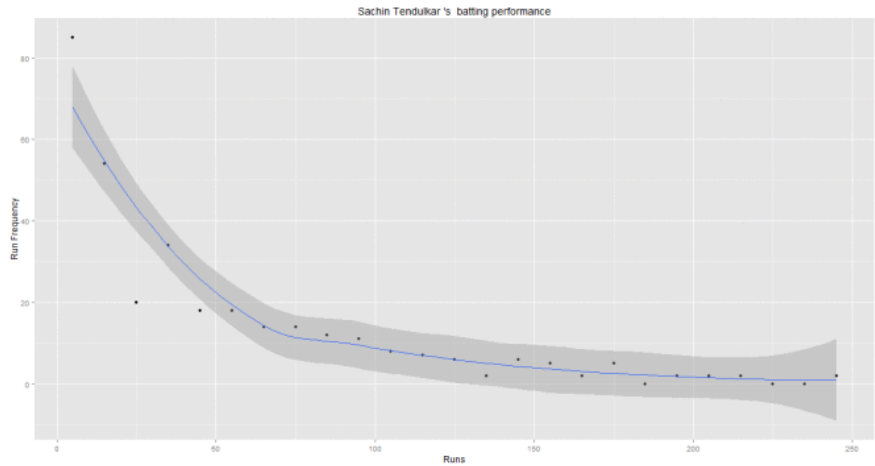
a) Combined Boxplot and histogram of runs scored



The above graph is combined boxplot and a histogram. The boxplot at the top shows the 1st quantile (25th percentile) which is the left side of the green rectangle, the 3rd quantile(75% percentile) right side of the green

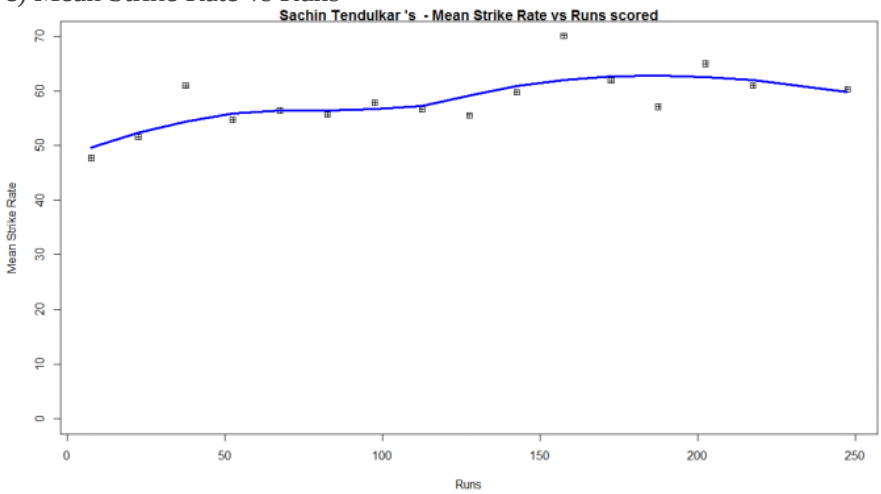
rectangle and the mean and the median. These values are also shown in the histogram below. The histogram gives the frequency of Runs scored in the given range for e.g (0-10, 11-20, 21-30 etc) for Tendulkar

b) Batting performance – Runs frequency vs Runs



The graph above plots the best fitting curve for Runs scored in the frequency ranges.

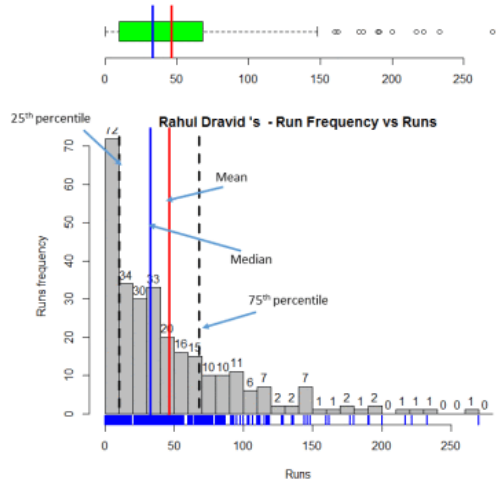
c) Mean Strike Rate vs Runs



This plot computes the Mean Strike Rate for each interval for e.g if between the ranges 11-21 the Strike Rates were 40.5, 48.5, 32.7, 56.8 then the average of these values is computed for the range 11-21 = (40.5 + 48.5 + 32.7 + 56.8)/4. This is done for all ranges and the Mean Strike Rate in each range is plotted and the loess curve is fitted for this data.

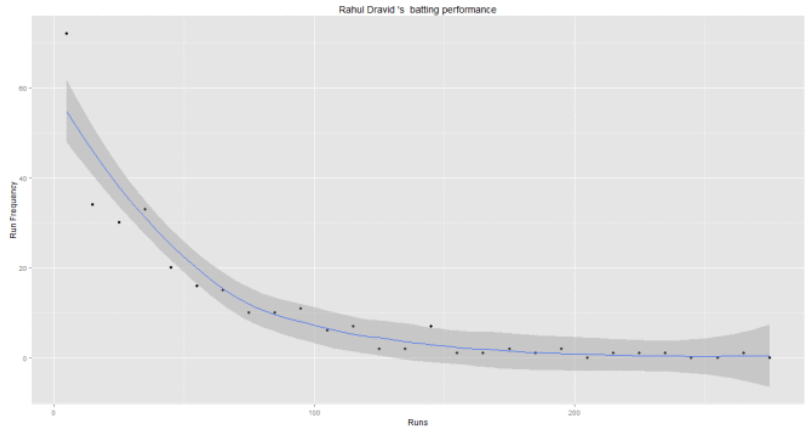
B) Batting performance of Rahul Dravid

a) Combined Boxplot and histogram of runs scored

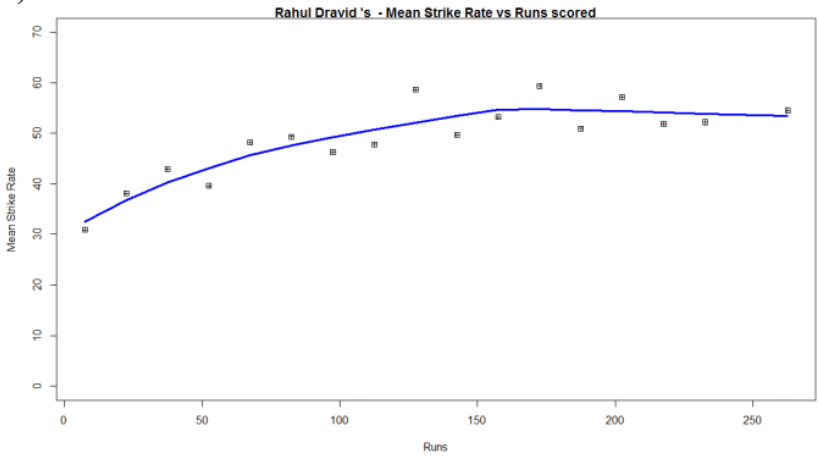


The mean, median, the 25th and 75 th percentiles for the runs scored by Rahul Dravid are shown above

b) Batting performance – Runs frequency vs Runs

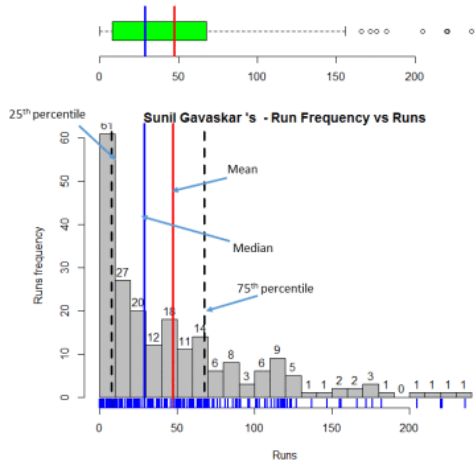


c) Mean Strike Rate vs Runs



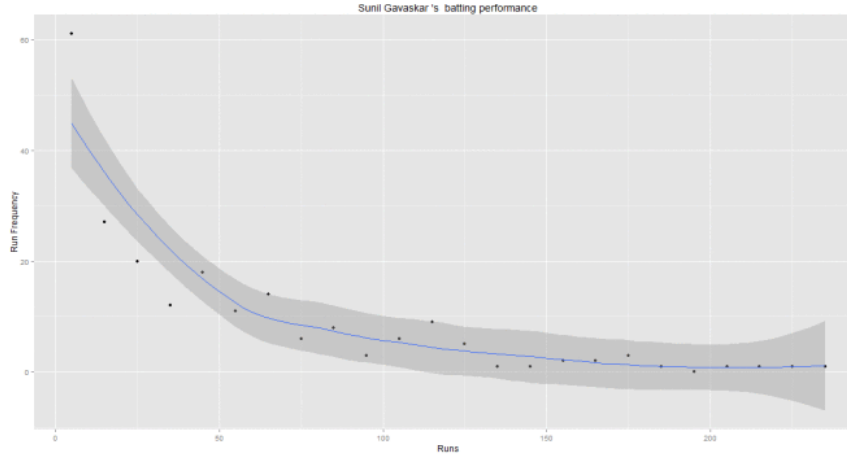
C) Batting performance of Sunil Gavaskar

a) Combined Boxplot and histogram of runs scored

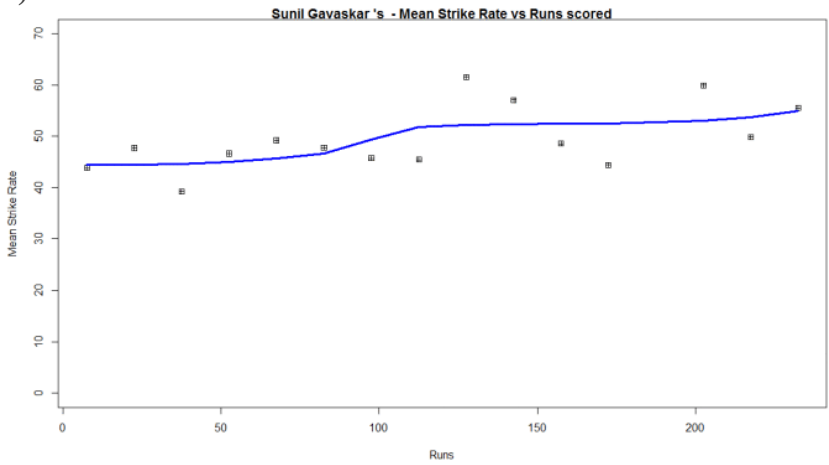


The mean, median, the 25th and 75 th percentiles for the runs scored by Sunil Gavaskar are shown above

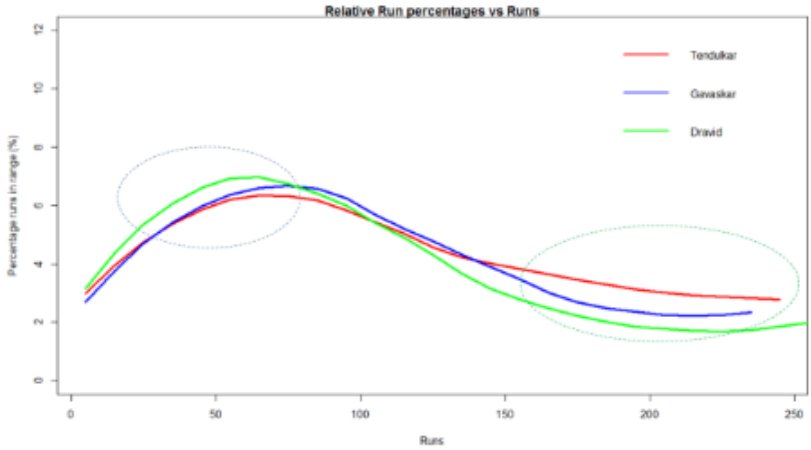
b) Batting performance – Runs frequency vs Runs



c) Mean Strike Rate vs Runs



D) Relative performances of Tendulkar, Dravid and Gavaskar



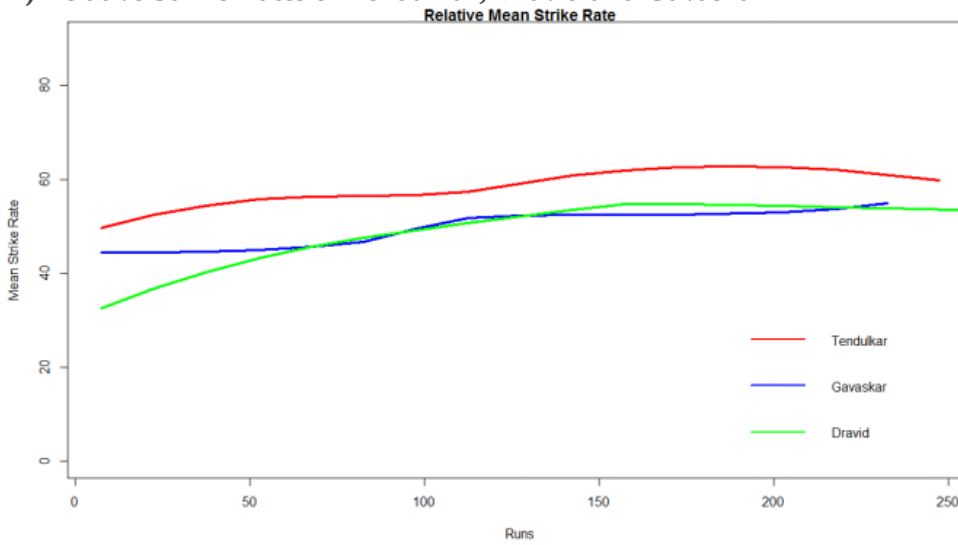
The above plot computes the percentage of the total career runs scored in a given range for each of the batsman.

For e.g. if Dravid scored the runs 23, 22, 28, 21, 25 in the range 21-30 then the

Range 21 – 20 => percentageRuns = (23 + 22 + 28 + 21 + 25)/ Total runs in career * 100

The above plot shows that Rahul Dravid's has a higher contribution in the range 20-70 while Tendulkar has a larger percentage in the range 150-230

E) Relative Strike Rates of Tendulkar, Dravid and Gavaskar



With respect to the Mean Strike Rate Tendulkar is clearly superior to both Gavaskar & Dravid

F) Analysis of Tendulkar, Dravid and Gavaskar

Name	Number of innings	Minimum runs	25th percentile	Median	Mean	75th percentile	Maximum runs	Skew	Kurtosis
Sachin Tendulkar	329	0	10	32	48.39	73	248	1.46	1.75
Sunil Gavaskar	214	0	8	29	47.3	67.75	236	1.42	1.72
Rahul Dravid	286	0	10.25	33	46.46	68	270	1.67	3.12

The above table captures the the career details of each of the batsman

The following points can be noted

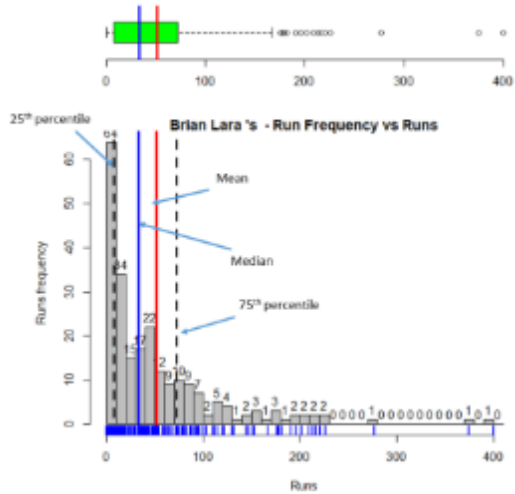
1) The 'number of innings' is the data you get after removing rows with DNB, TDNB etc

2) Tendulkar has the higher average 48.39 > Gavaskar (47.3) > Dravid (46.46)

3) The skew of Dravid (1.67) is greater which implies that there the runs scored are more skewed to right (greater runs) in comparison to mean

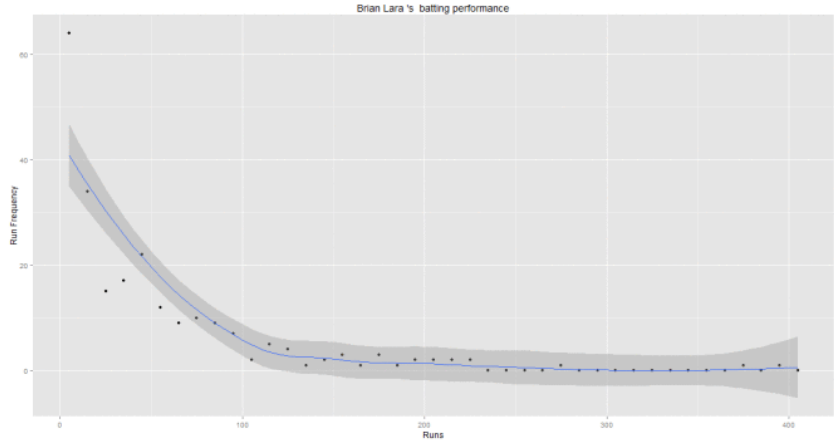
G) Batting performance of Brian Lara

a) Combined Boxplot and histogram of runs scored

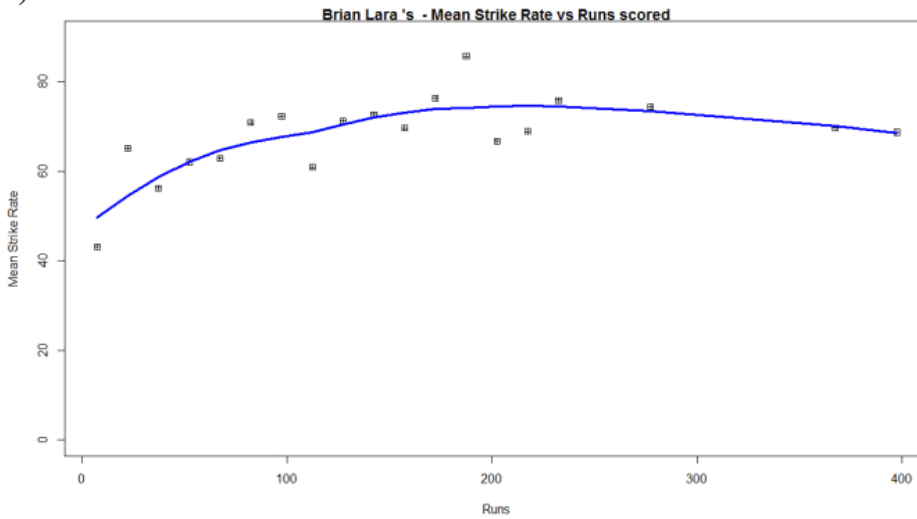


The mean, median, 1st and 3rd quartile are shown above

b) Batting performance – Runs frequency vs Runs

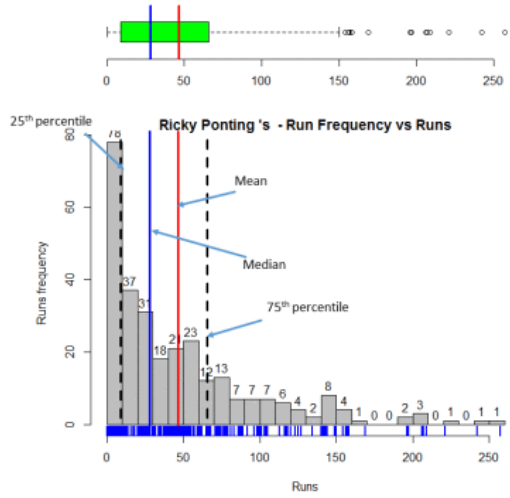


c) Mean Strike Rate vs Runs

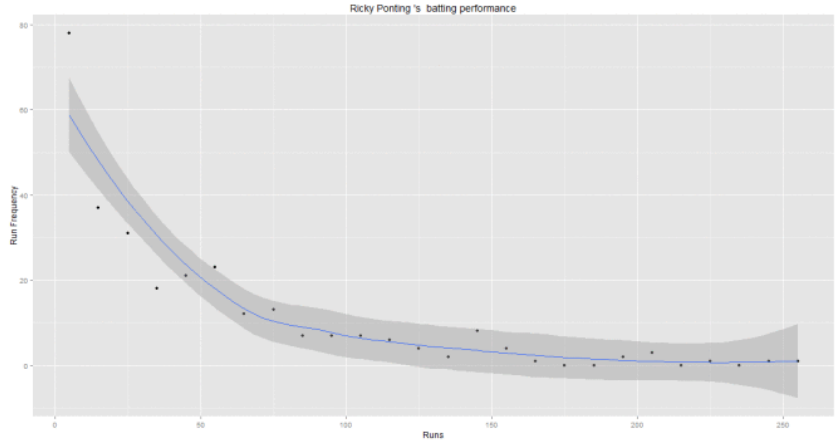


H) Batting performance of Ricky Ponting

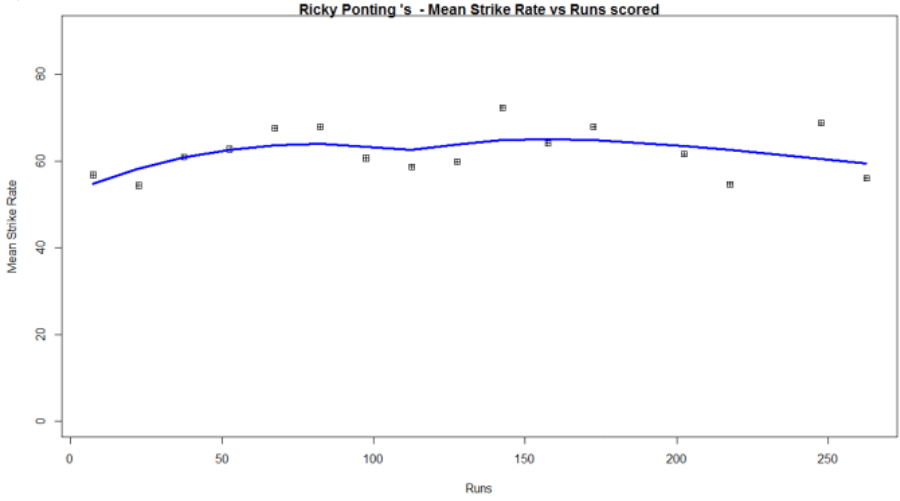
a) Combined Boxplot and histogram of runs scored



b) Batting performance – Runs frequency vs Runs

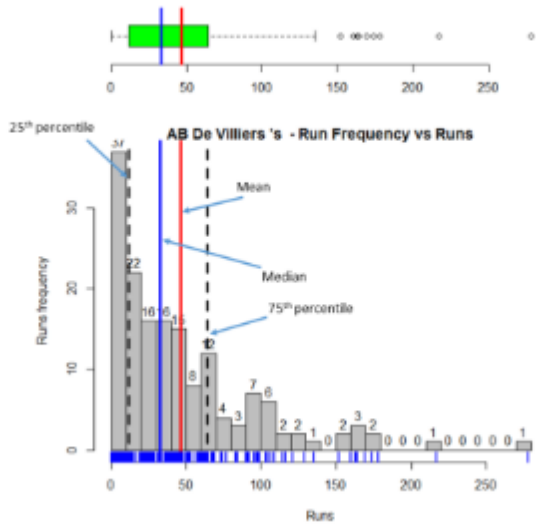


c) Mean Strike Rate vs Runs

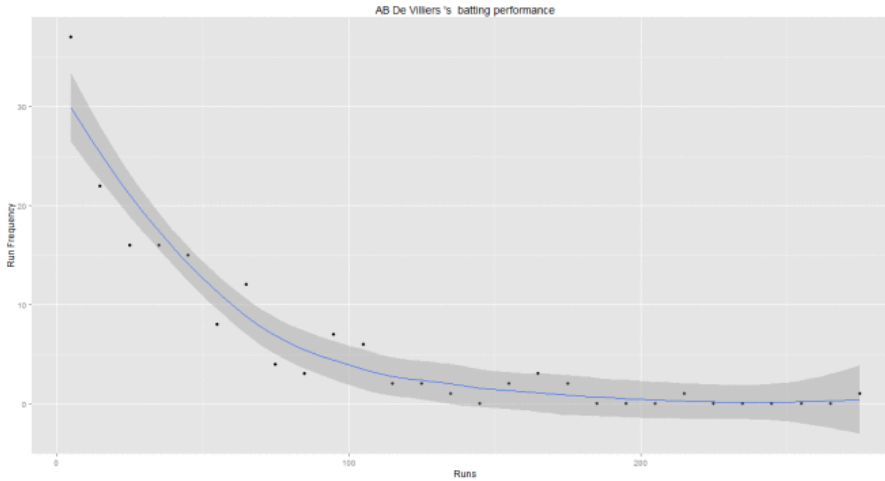


I) Batting performance of AB De Villiers

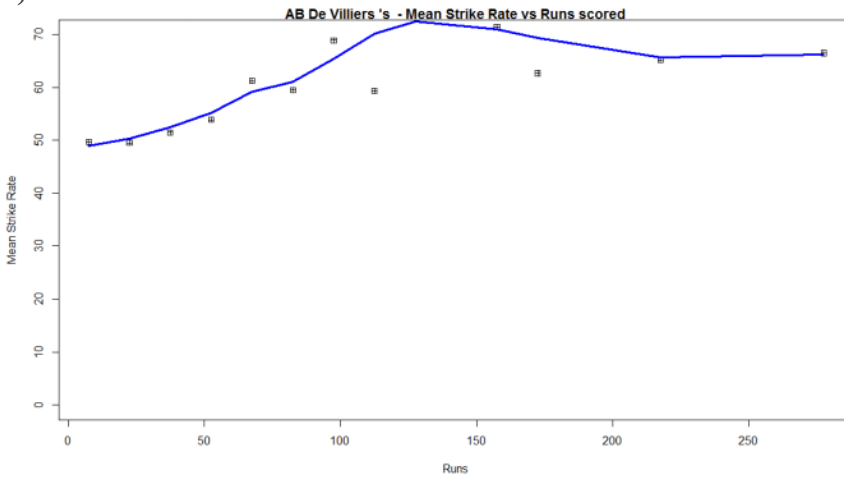
a) Combined Boxplot and histogram of runs scored



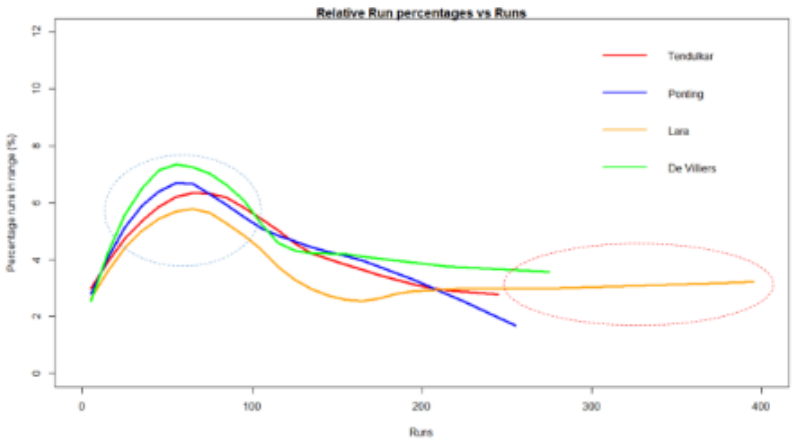
b) Batting performance – Runs frequency vs Runs



c) Mean Strike Rate vs Runs

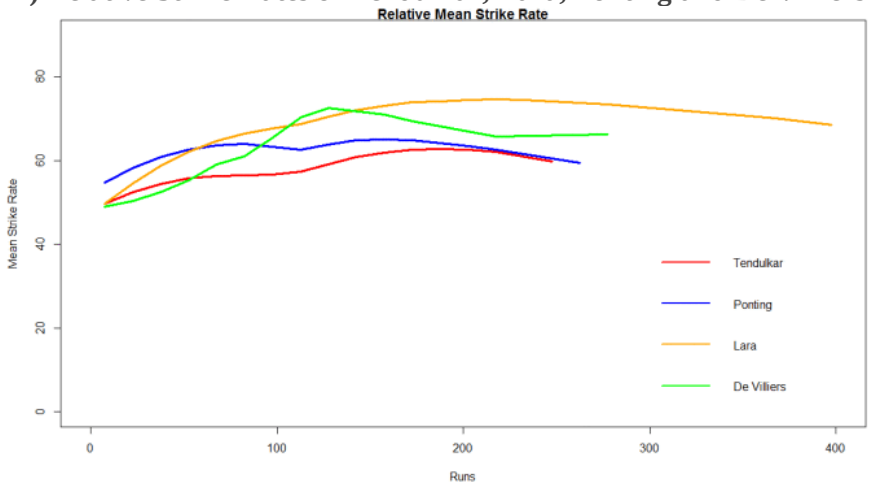


J) Relative performances of Tendulkar, Lara, Ponting and De Villiers



Clearly De Villiers is ahead in the percentage Runs scores in the range 30-80. Tendulkar is better in the range between 80-120. Lara's career has a long tail.

K) Relative Strike Rates of Tendulkar, Lara, Ponting and De Villiers



The Mean Strike Rate of Lara is ahead of the lot, followed by De Villiers, Ponting and then Tendulkar

L) Analysis of Tendulkar, Lara, Ponting and De Villiers

Name	Number of innings	Minimum runs	25th percentile	Median	Mean	75th percentile	Maximum runs	Skew	Kurtosis
Sachin Tendulkar	329	0	10	32	48.39	73	248	1.46	1.75
AB De Villiers	160	0	11.75	33	46.55	64.25	278	1.74	3.79
Brian Lara	232	0	8	33.5	51.52	72.25	400	2.31	7.2
Ricky Ponting	287	0	9	28	46.61	65.5	257	1.6	2.54

The following can be observed from the above table

1) Brian Lara has the highest average (51.52) > Sachin Tendulkar (48.39 > Ricky Ponting (46.61) > AB De Villiers (46.55)

2) Brian Lara also the highest skew which means that the data is more skewed to the right of the mean than the others

You can clone the code from Github at the following link https://github.com/tvganesh/bestBatsman. You should be able to use the code as-is for any other batsman you choose to.

3. Appendix

Cricket analysis with Machine Learning using Octave

3.1. Informed choices through Machine Learning – Analyzing Kohli, Tendulkar and Dravid

Having just completed the highly stimulating & inspiring Stanford's Machine Learning course at Coursera, by the incomparable Professor Andrew Ng I wanted to give my newly acquired knowledge a try. As a start, I decided to try my hand at analyzing one of India's fastest growing stars, namely Virat Kohli . For the data on Virat Kohli I used the 'Statistics database' at ESPN Cricinfo. To make matters more interesting, I also pulled data on the iconic Sachin Tendulkar and the Mr. Dependable, Rahul Dravid.

Based on the data of these batsmen I perform some predictions with the help of machine learning algorithms. That I have a proclivity for prediction, is not surprising, considering the fact that my Dad was an astrologer who had reasonable success at this esoteric art. While he would be concerned with planetary positions, about Rahu in the 7th house being in the malefic etc., I on the other hand focus my predictions on multivariate regression analysis and K-Means. The first part of my post gives the results of my analysis and some predictions for Kohli, Tendulkar and Dravid.

The second part of the post contains a brief outline of the implementation and not the actual details of implementation. This is ensure that I don't violate Coursera's Machine Learning' Honor Code.

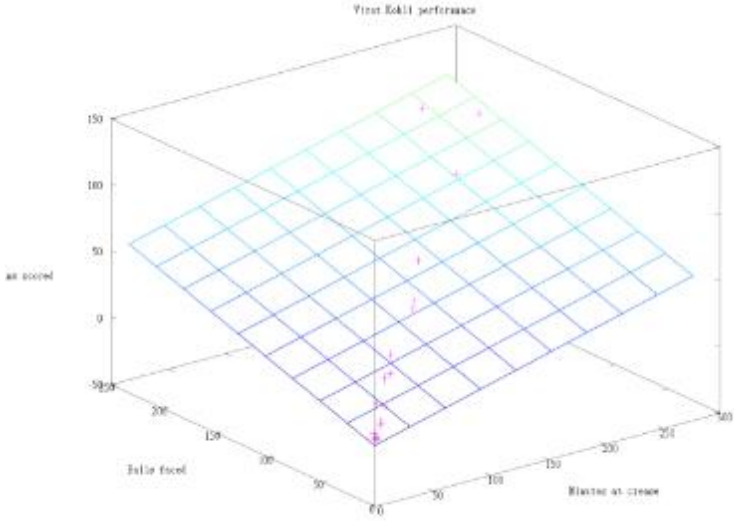
This code, data used and the output obtained can be accessed at GitHub at https://github.com/tvganesh/ml-cricket-analysis

Analysis and prediction of Kohli, Tendulkar and Dravid with Machine Learning As mentioned above, I pulled the data for the 3 cricketers Virat Kohli, Sachin Tendulkar and Rahul Dravid. The data taken from Cricinfo database for the 3 batsman is based on the following assumptions

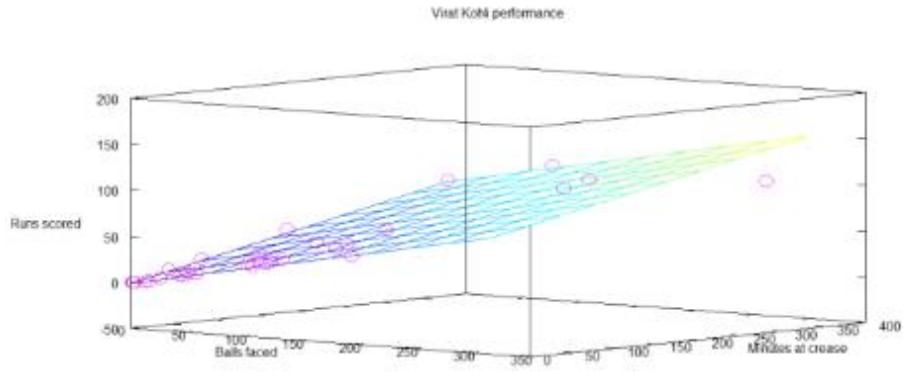
1. Only 'Minutes at Crease' and 'Balls Faced' were taken as features against the output variable 'Runs scored'
2. Only test matches were taken. This included both test 'at home' and 'away tests'
3. The data was cleaned to remove any DNB (did not bat) values
4. No extra weightage was given to 'not out'. So if Kohli made '28*' 28 not out, this was taken to be 28 runs

Regression Analysis for Virat Kohli There are 51 data points for Virat Kohli regarding Tests played. The data for Kohli is displayed as a 3D scatter plot where x-axis is 'minutes' and y-axis is 'balls faced'. The vertical z-axis is the 'runs scored'. Multivariate regression analysis was performed to find the best fitting plane for the runs scored based on the selected features of 'minutes' and 'balls faced'.

This is based on minimizing the cost function and then performing gradient descent for 400 iterations to check for convergence. This plane is shown as the 3-D plane that provides the best fit for the data points for Kohli. The diagram below shows the prediction plane of expected runs for a combination of 'minutes at crease' and 'balls faced'. Here are 2 such plots for Kohli



Another view of the prediction plane



Prediction for Kohli I have also computed the predicted runs that will be scored by Kohli for different combinations of 'minutes at crease' and 'balls

faced'. As an example, from the table below, we can see that the predicted runs for Kohli after being in the crease for 110 minutes and facing 135 balls is 54 runs.

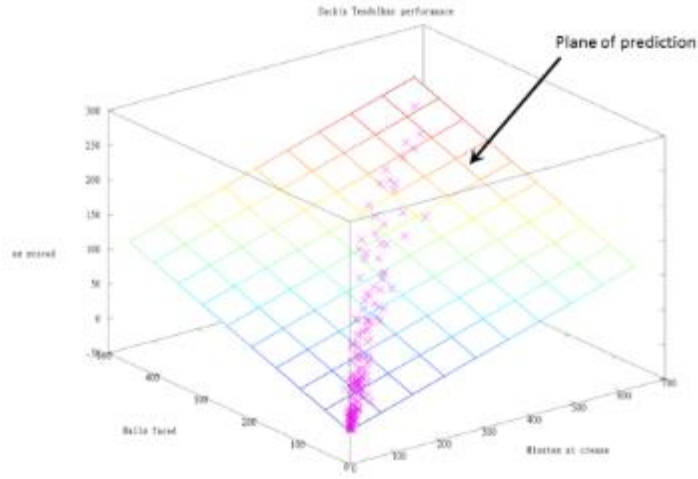
.

Virat Kohli's performance

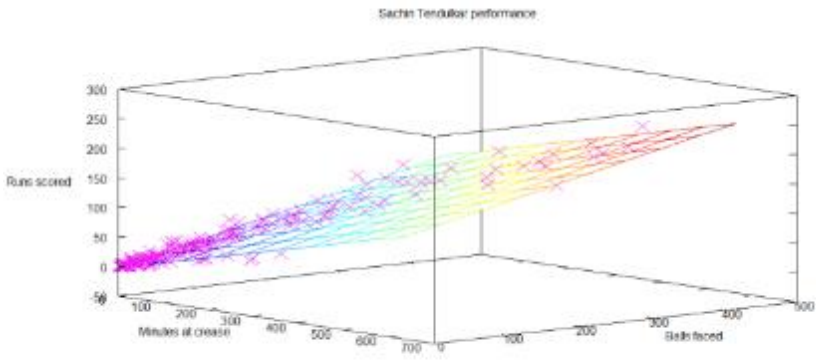
Balls Faced	10	35	60	85	110	135	160	185	210	235	260	285	310
Minutes at crease													
10	3.36	9.49	15.62	21.76	27.89	34.02	40.15	46.28	52.41	58.54	64.68	70.81	76.94
35	8.36	14.50	20.63	26.76	32.89	39.02	45.15	51.28	57.42	63.55	69.68	75.81	81.94
60	13.37	19.50	25.63	31.76	37.89	44.02	50.16	56.29	62.42	68.55	74.68	80.81	86.94
85	18.37	24.50	30.63	36.76	42.90	49.03	55.16	61.29	67.42	73.55	79.68	85.81	91.95
110	23.37	29.50	35.64	41.77	47.90	54.03	60.16	66.29	72.42	78.55	84.69	90.82	96.95
135	28.38	34.51	40.64	46.77	52.90	59.03	65.16	71.29	77.43	83.56	89.69	95.82	101.95
160	33.38	39.51	45.64	51.77	57.90	64.03	70.17	76.30	82.43	88.56	94.69	100.82	106.95
185	38.38	44.51	50.64	56.77	62.91	69.04	75.17	81.30	87.43	93.56	99.69	105.83	111.96
210	43.38	49.51	55.65	61.78	67.91	74.04	80.17	86.30	92.43	98.57	104.70	110.83	116.96
235	48.39	54.52	60.65	66.78	72.91	79.04	85.17	91.31	97.44	103.57	109.70	115.83	121.96
260	53.39	59.52	65.65	71.78	77.91	84.05	90.18	96.31	102.44	108.57	114.70	120.83	126.96
285	58.39	64.52	70.65	76.79	82.92	89.05	95.18	101.31	107.44	113.57	119.70	125.84	131.97
310	63.39	69.53	75.66	81.79	87.92	94.05	100.18	106.31	112.44	118.58	124.71	130.84	136.97

Regression analysis for Sachin Tendulkar There was a lot more data on Tendulkar and I was able to dump close to 329 data points. As before the 'minutes at crease', 'balls faced' vs 'runs scored' were plotted as a 3D scatter plot. The prediction plane is calculated using gradient descent and is shown

as a plane in the diagram below



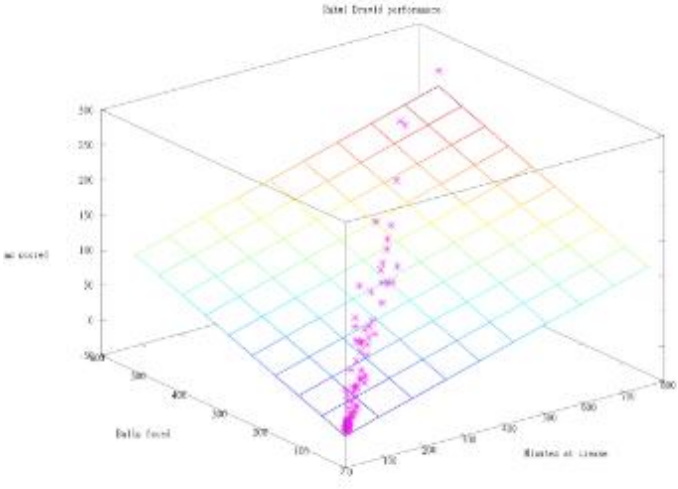
Another view of this is shown below

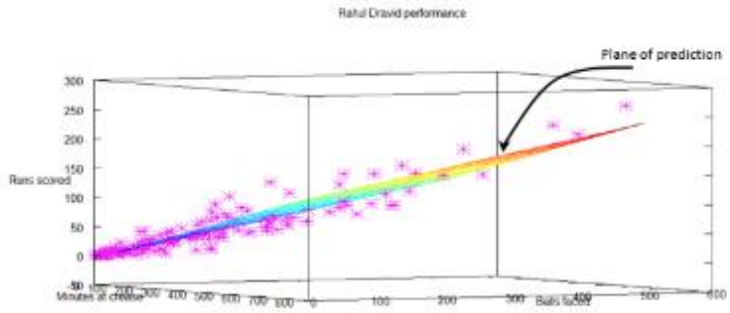


Predicted runs for Tendulkar The table below gives the predicted runs for Tendulkar for a combination of time at crease and balls faced. Hence, Tendulkar will score 57 runs in 110 minutes after facing 135 deliveries

Sachin Tendulkar's performance													
Balls Faced	10	35	60	85	110	135	160	185	210	235	260	285	310
Minutes at crease													
10	2.400126	9.391455	16.38279	23.37412	30.36544	37.35677	44.3481045	51.33943	58.33076	65.32209	72.31342	79.30475	86.29608
35	7.357864	14.34919	21.34052	28.33185	35.32318	42.31451	49.3058426	56.29717	63.2885	70.27983	77.27116	84.26249	91.25382
60	12.3156	19.30693	26.29826	33.28959	40.28092	47.27225	54.2635808	61.25491	68.24624	75.23757	82.2289	89.22023	96.21156
85	17.27334	24.26467	31.256	38.24733	45.23866	52.22999	59.2213189	66.21265	73.20398	80.19531	87.18664	94.17797	101.1693
110	22.23108	29.22241	36.21374	43.20507	50.1964	57.18773	64.1790571	71.17039	78.16172	85.15305	92.14438	99.13571	106.127
135	27.18882	34.18015	41.17148	48.16281	55.15414	62.14547	69.1367952	76.12813	83.11945	90.11078	97.10211	104.0934	111.0848
160	32.14655	39.13788	46.12921	53.12054	60.11187	67.1032	74.0945334	81.08586	88.07719	95.06852	102.0599	109.0512	116.0425
185	37.10429	44.09562	51.08695	58.07828	65.06961	72.06094	79.0522715	86.0436	93.03493	100.0263	107.0176	114.0089	121.0003
210	42.06203	49.05336	56.04469	63.03602	70.02735	77.01868	84.0100097	91.00134	97.99267	104.984	111.9753	118.9667	125.958
235	47.01977	54.0111	61.00243	67.99376	74.98509	81.97642	88.9677478	95.95908	102.9504	109.9417	116.9331	123.9244	130.9157
260	51.97751	58.96884	65.96017	72.9515	79.94283	86.93416	93.925486	100.9168	107.9081	114.8995	121.8908	128.8821	135.8735
285	56.93525	63.92658	70.9179	77.90923	84.90056	91.89189	98.8832241	105.8746	112.8659	119.8572	126.8485	133.8399	140.8312
310	61.89298	68.88431	75.87564	82.86697	89.8583	96.84963	103.840962	110.8323	117.8236	124.815	131.8063	138.7976	145.7889

Regression Analysis for Rahul Dravid The same was done for 'the Wall' Dravid. The prediction plane is below





Predicted runs for Dravid The predicted runs for Dravid for combinations of batting time and balls faced is included below. The predicted runs for Dravid after facing 135 deliveries in 110 minutes is 44 runs

.

Rahul Dravid's performance

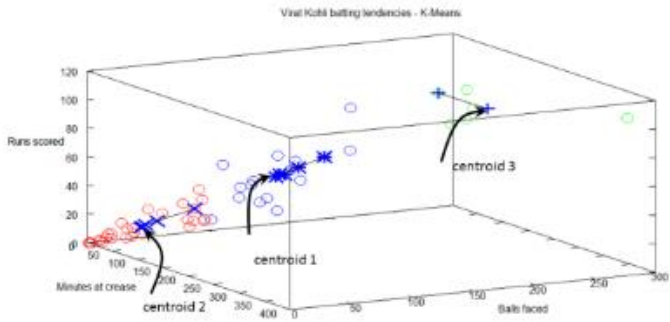
Balls Faced	10	35	60	85	110	135	160	185	210	235	260	285	310
Minutes at crease													
10	-1.07	4.80	10.67	16.55	22.42	28.29	34.17	40.04	45.92	51.79	57.66	63.54	69.41
35	3.00	8.87	14.75	20.62	26.49	32.37	38.24	44.12	49.99	55.86	61.74	67.61	73.48
60	7.07	12.94	18.82	24.69	30.57	36.44	42.31	48.19	54.06	59.94	65.81	71.68	77.56
85	11.14	17.02	22.89	28.77	34.64	40.51	46.39	52.26	58.14	64.01	69.88	75.76	81.63
110	15.22	21.09	26.96	32.84	38.71	44.59	50.46	56.33	62.21	68.08	73.96	79.83	85.70
135	19.29	25.16	31.04	36.91	42.79	48.66	54.53	60.41	66.28	72.15	78.03	83.90	89.78
160	23.36	29.24	35.11	40.98	46.86	52.73	58.61	64.48	70.35	76.23	82.10	87.98	93.85
185	27.44	33.31	39.18	45.06	50.93	56.81	62.68	68.55	74.43	80.30	86.17	92.05	97.92
210	31.51	37.38	43.26	49.13	55.00	60.88	66.75	72.63	78.50	84.37	90.25	96.12	102.00
235	35.58	41.46	47.33	53.20	59.08	64.95	70.83	76.70	82.57	88.45	94.32	100.19	106.07
260	39.65	45.53	51.40	57.28	63.15	69.02	74.90	80.77	86.65	92.52	98.39	104.27	110.14
285	43.73	49.60	55.48	61.35	67.22	73.10	78.97	84.84	90.72	96.59	102.47	108.34	114.21
310	47.80	53.67	59.55	65.42	71.30	77.17	83.04	88.92	94.79	100.67	106.54	112.41	118.29

Further analysis While the 'prediction plane' was useful, it somehow

does not give a clear picture of how effective each batsman is. Clearly the 3D plots show at least 3 clusters for each batsman. For all batsmen, the clustering is densest near the origin, become less dense towards the middle and sparse on the other end. This is an indication during which session during their innings the batsman is most prone to get out. So I decided to perform K-Means clustering on the data for the 3 batsman. This gives the 3 general tendencies for each batsman. The output is included below

K-Means for Virat The K-Means for Virat Kohli indicate the follow
Centroids found 255.000000 104.478261 19.900000
Centroids found 194.000000 80.000000 15.650000
Centroids found 103.000000 38.739130 7.000000

Analysis of Virat Kohli's batting tendency
Kohli has a 45.098 percent tendency to bat for 104 minutes, face 80 balls and score 38 runs
Kohli has a 39.216 percent tendency to bat for 19 minutes, face 15 balls and score 7 runs
Kohli has a 15.686 percent tendency to bat for 255 minutes, face 194 balls and score 103 runs
The computation of this included in the diagram below



K-means for Sachin Tendulkar

The K-Means for Sachin Tendulkar indicate the following

Centroids found 166.132530 353.092593 43.748691

Centroids found 121.421687 250.666667 30.486911

Centroids found 65.180723 138.740741 15.748691

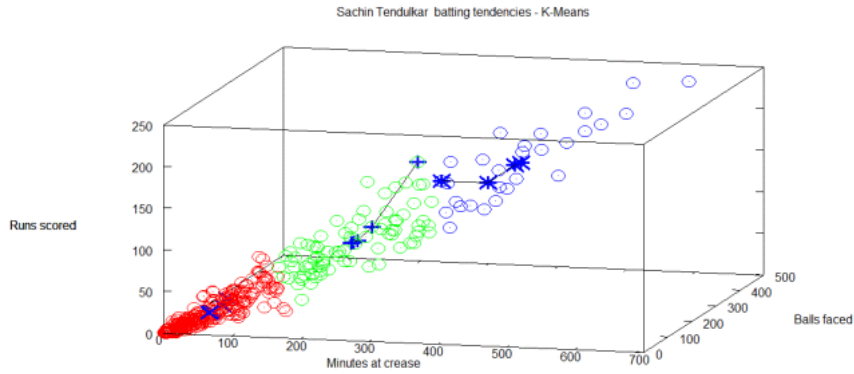
Analysis of Sachin Tendulkar's performance

Tendulkar has a 58.232 percent tendency to bat for 43 minutes, face 30 balls and score 15 runs

Tendulkar has a 25.305 percent tendency to bat for 166 minutes, face 121 balls and score 65 runs

Tendulkar has a 16.463 percent tendency to bat for 353 minutes,

face 250 balls and score 138 runs



K-Means for Rahul Dravid

Centroids found 191.836364 409.000000 50.506024

Centroids found 137.381818 290.692308 34.493976

Centroids found 56.945455 131.500000 13.445783

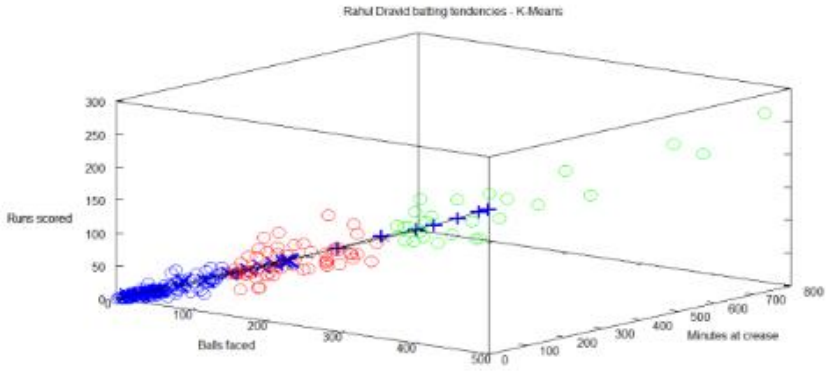
Analysis of Rahul Dravid's performance

Dravid has a 50.610 percent tendency to bat for 50 minutes, face 34 balls and score 13 runs

Dravid has a 33.537 percent tendency to bat for 191 minutes, face 137 balls and score 56 runs

Dravid has a 15.854 percent tendency to bat for 409 minutes,

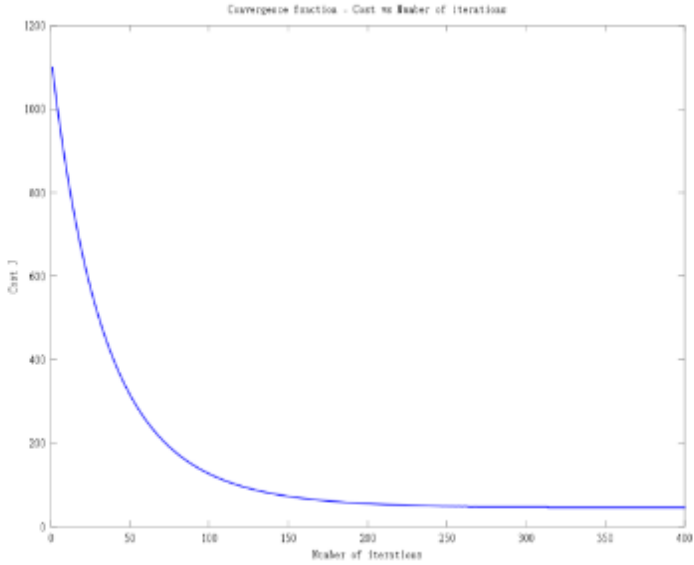
face 290 balls and score 131 runs



Some implementation details The entire analysis and coding was done with Octave 3.2.4. I have included the outline of the code for performing the multivariate regression. In essence the pseudo code for this

1. Read the batsman data (Minutes, balls faced versus Runs scored)
2. Calculate the cost
3. Perform Gradient descent

The cost was plotted against the number of iterations to ensure convergence while performing gradient descent



Plot the 3-D plane that best fits the data

The outline of this code, data used and the output obtained can be accessed at GitHub at ml-cricket-analysis

Conclusion: Comparing the results from the K-Means Tendulkar has around 48% to make a score greater than 60

Tendulkar has a 25.305 percent tendency to bat for 166 minutes, face 121 balls and score 65 runs

Tendulkar has a 16.463 percent tendency to bat for 353 minutes, face 250 balls and score 138 runs

And Dravid has a similar 48% tendency to score greater than 56 runs

Dravid has a 33.537 percent tendency to bat for 191 minutes, face 137 balls and score 56 runs

Dravid has a 15.854 percent tendency to bat for 409 minutes, face 290 balls and score 131 runs

Kohli has around 45% to score greater than 38 runs

Kohli has a 45.098 percent tendency to bat for 104 minutes, face 80 balls and score 38 runs

Also Kohli has a lesser percentage to score lower runs as against the other two

Kohli has a 39.216 percent tendency to bat for 19 minutes, face 15 balls and score 7 runs

The results must be looked in proper perspective as Kohli is just starting his career while the other 2 are veterans. Kohli has a long way to go and I am certain that he will blaze a trail of glory in the years to come!

3.2. Informed choices through Machine Learning-2 Pitting together Kumble, Kapil, Chandra

Continuing my earlier 'innings', of test driving my knowledge in Machine Learning acquired via Coursera, I now turn my attention towards the bowling performances of our Indian bowling heroes. In this post I give a slightly different 'spin' to the bowling analysis and hope I can 'swing' your opinion based on my assessment.
I guess that is enough of my cricketing 'double-speak' for now and I will get down to the real business of my bowling analysis!

As in my earlier post Informed choices through Machine Learning – Analyzing Kohli, Tendulkar and Dravid ,the first part of the post has my analyses and the latter part has the details of the implementation of the algorithm. Feel free to read the first part and either scan or skip the latter.

To perform this analysis I have skipped the data on our recent crop of new bowlers. The reason being that data is scant on these bowlers, besides they also seem to have a relatively shorter shelf life (hope there are a couple of finds in this Australian tour of Dec 2014). For the analyses I have chosen B S Chandrasekhar, Kapil Dev Anil Kumble. My rationale as to why I chose the above 3
B S Chandrasekhar also known as "Chandra' was one of the most lethal leg spinners in the late 1970's. He had a very dangerous combination of fast leg breaks, searing tops spins interspersed with

the occasional googly. On many occasions he would leave most batsmen completely clueless.

Kapil Nikhanj Dev, the Haryana Hurricane who could outwit the most technically sound batsmen through some really clever bowling. His variations were almost always effective and he would achieve the vital breakthrough outsmarting the opponent.

And finally Anil Kumble, I chose Kumble because in my opinion he is truly the embodiment of the 'thinking' bowler. Many times I have seen Kumble repeatedly beat batsmen. It was like he was telling the batsman 'check' as he bowled faster leg breaks, flippers, a straighter delivery or top spins before finally crashing into the wickets or trapping the batsmen. It felt he was saying 'checkmate dude!'

I have taken the data for the 3 bowlers from ESPN Cricinfo. Only the Test matches were considered for the analyses. All tests against all oppositions both at home and away were included

The assumptions taken and basis of the computation is included below

a. The data is based on the following 2 input variables a) Overs bowled b) Runs given. The output variable is 'Wickets taken'

b. To my surprise I found that in the late 1970's when BS Chandrasekhar used to bowl, an over had 8 balls for matches in

Australia. So, I had to normalize this data for Chandra to make it on par with the others. Hence for Chandra where the overs were made up of 8 balls the overs was calculated as follows

Overs (O) = (Overs * 8)/6

c. The Economy rate E was calculated as below

E = Overs/runs was chosen as input variable to take into account fewer runs given by the bowler

d. The output variable was re-calculated as Strike Rate (SR) to determine the 'bowling effectiveness'

Strike Rate = Wickets/Overs

(not be confused with a batsman's strike rate batsman strike rate = runs/ balls faced)

e. Hence the analysis is based on

f(O,E) = SR

An outline of the Octave code and the data used can be cloned from GitHub at ml-bowling-analyze

1. Surface of Bowling Effectiveness (SBE)

In my earlier post I was able to fit a 'prediction plane' based on the minutes at crease, balls faced versus the runs scored. But in this case a plane did not make sense as the wickets can only range from 0 – 10 and in most cases averaging between 3 and 5. So I plot the best fitting 3-D surface over the predicted hypothesis function. The steps performed are

1) The data for the different bowlers were cleaned with data which indicated (DNB – Did not bowl)

2) The Economy Rate (E) = Runs given/Overs and Strike Rate(SR)

= Wickets/overs were calculated.

3) The product of Overs (O), and Economy(E) were stored as Over_Economy(OE)

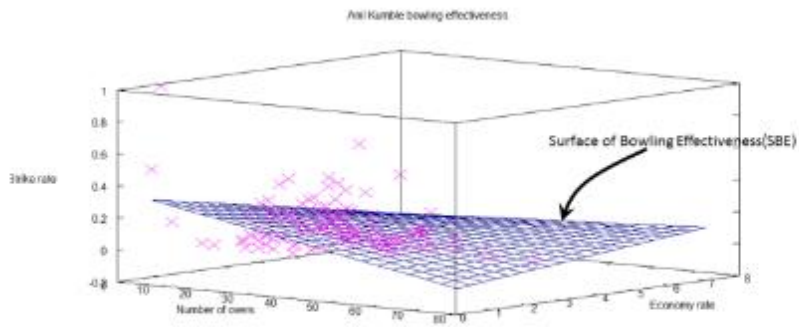
4) The hypothesis function was computed as h(O, E, OE) = y

5) Theta was calculated using the Normal Equation. The Surface of Bowling Effectiveness(SBE) was then plotted. The plots for each of the bowler is shown below

Here are the plots

A) Anil Kumble

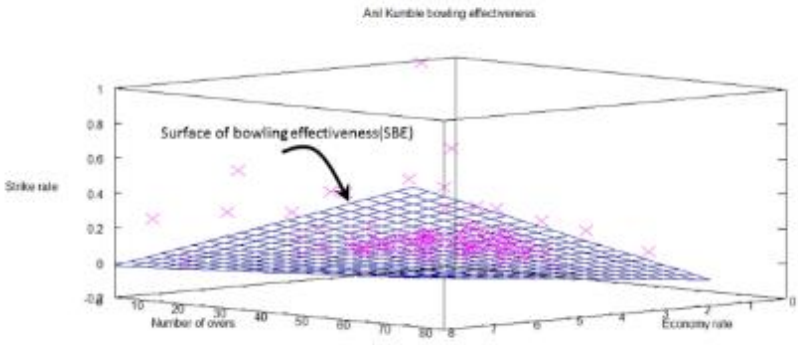
The data of Kumble, based on Overs bowled & Economy rate versus the Strike Rate is plotted as a 3-D scatter plot (pink crosses). The best fit as determined by solving the optimum theta using the Normal Equation is plotted as 3-D surface shown below.



The 3-D surface is what I have termed as 'Surface of Bowling Effectiveness (SBE)' as it depicts bowlers overall effectiveness as it

plots the overs (O), 'economy rate' E against predicted 'strike rate' SR.

Here is another view



The theta values obtained for Kumble are

Theta =

0.104208

-0.043769

-0.016305

0.011949

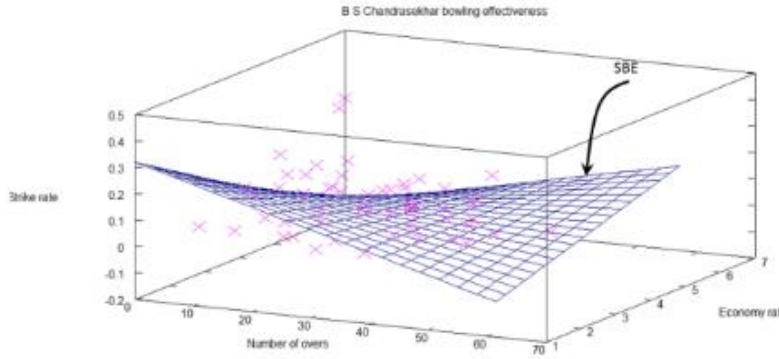
And the cost at this theta is

Cost Function **J = 0.0046269**

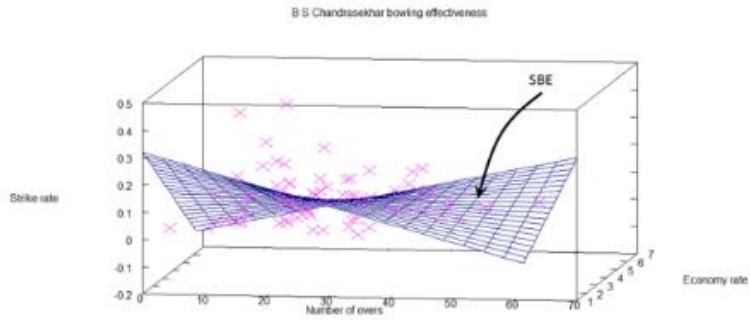
B) B S Chandrasekhar

Here are the best optimal surface plot for Chandra with the data on O,E vs SR plotted as a 3D scatter plot. Note: The dataset

for Chandrasekhar is smaller compared to the other two.



Another view for Chandra



Theta values for B S Chandrasekhar are

Theta =

0.095780

-0.025377

-0.024847

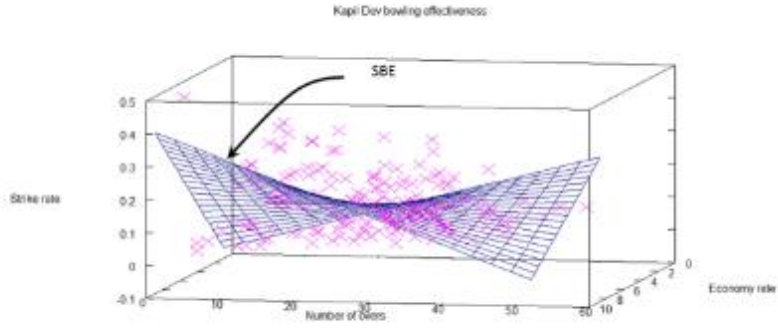
0.023415

and the cost is

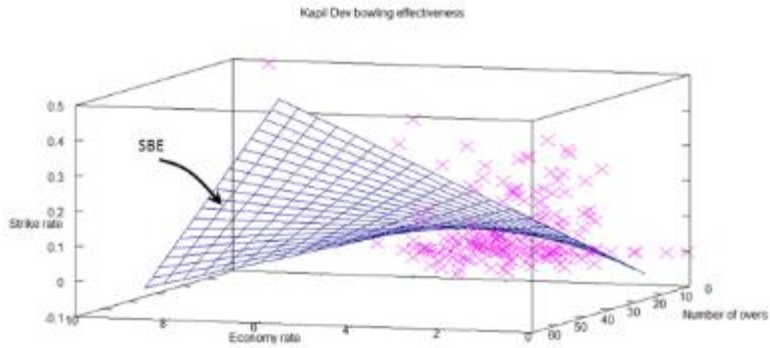
Cost Function J = 0.0032980

c) Kapil Dev

The plots for Kapil



Another view of SBE for Kapil



The Theta values and cost function for Kapil are

Theta =

0.090219

0.027725

0.023894

-0.021434

Cost Function J = 0.0035123

2. Predicting wickets

In the previous section the optimum theta with the lowest Cost Function J was calculated. Based on the value of theta, the wickets that will be taken by a bowler can be computed as the product of the hypothesis function and theta. i.e.

y= h(x) * theta => Strike Rate (SR) = [1 O E OE] * theta

Now predicted wickets can be calculated as

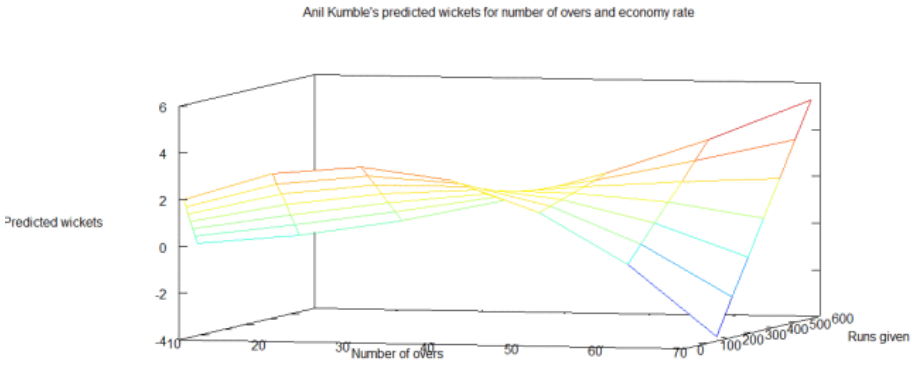
wickets = Strike rate(SR) * Overs(O)

This is done for Kumble, Chandra and Kapil for different combinations of Overs(O) and Economy(E) rate.

Here are the results

Predicted wickets for Anil Kumble

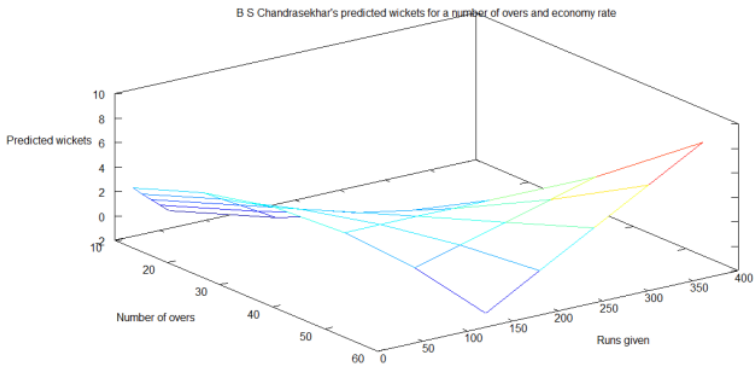
The plot of predicted wickets for Kumble is represented below



This can also be represented as a table

Predicted wickets for B S Chandrasekhar

	Anil Kumble's predicted wickets						
Economy rate	**2**	**3**	**4**	**5**	**6**	**7**	**8**
Number of overs							
10	2	2	1	1	1	0	0
20	3	3	2	2	1	1	0
30	3	3	2	2	2	1	1
40	3	3	2	2	2	2	1
50	1	2	2	2	2	2	2
60	-1	0	1	2	2	3	4
70	-4	-2	-1	1	2	4	5

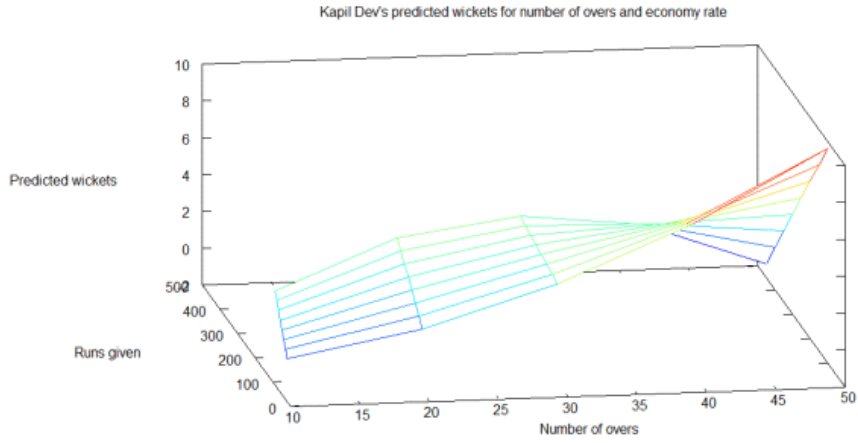


The table for Chandra

	B S Chandrasekhar's predicted wickets				
Economy	2	3	4	5	6
Number of overs					
10	2	1	1	0	-1
20	3	2	1	1	0
30	3	3	2	1	1
40	3	3	3	3	3
50	1	2	3	4	6
60	-1	2	4	7	9

Predicted wickets for Kapil Dev

The plot



The predicted table from the hypothesis function for Kapil Dev

	Kapil Dev's predicted wickets							
Economy rate	2	3	4	5	6	7	8	9
Number of overs								
10	0	1	1	1	2	2	3	3
20	1	2	2	3	3	4	4	4
30	3	3	4	4	4	4	4	4
40	6	6	5	4	4	3	3	2
50	10	8	7	5	3	2	0	-1

Observation: A closer look at the predicted wickets for Kapil, Kumble and B S Chandra shows an interesting aspect. The predicted number of wickets is higher for lower economy rates. With a little thought we can see bowlers on turning or pitches with a lot of

movement can not only be more economical but can also be destructive and take a lot of wickets. Hence the higher wickets for lower economy rates!

Implementation details

In this post I have used the Normal Equation to get the optimal values of theta for local minimum of the Gradient function. As mentioned above when I had run the 3D scatter plot fitting a 2D plane did not seem quite right. So I had to experiment with different polynomial equations first trying 2nd order, 3rd order and also the sqrt

I tried the following where 'O is Overs, 'E' stands for Economy Rate and 'SR' the predicated Strike rate. Theta is the computed theta from the Normal Equation. The notation in Matrix notation is shown below

i) A linear plane

SR = [1 O E] * theta

ii) Using the sqrt function

SR = [1 sqrt(O) sqrt(E)] * theta

iii) Using 2nd order plynomial

SR = [1 O^2 E^2] * theta

iv) Using the 3rd order polynomial

SR = [1 O^3 E^3] * theta

v) Before finally settling on
SR = [1 O E OE] * theta

where OE = O .* E

The last one seemed to give me the lowest cost and also seemed the most logical visual choice.

A good resource to play around with different functions and check out the shapes of combinations of variables and polynomial order of equation is at Wolfram Alpha: Plotting and Graphics
Note 1: The gradient descent with the Normal Equation has been performed on the entire data set (approx. 220 for Kumble & Kapil) and 99 for Chandra. The proper process for verifying a Machine Learning algorithm is to split the data set into (60% training data, 20% cross validation data and 20% as the test set). We need to validate the prediction function against the cross-validation set, fine tune it and finally ensure that it fits the test set samples well. However, this split was not done as the data set itself was very low. The entire data set was used to perform the optimal surface fit

Note 2: The optimal theta values have been chosen with a feature vector that is of the form
[1 x y x .* y] The Surface of Bowling Effectiveness' has been plotted above. It may appear that there is a 'high bias' in the fit and an even better fit could be obtained by choosing higher order polynomials

like

[1 x y x*y x^2 y^2 (x^2) .* y x .* (y^2)] or

[1 x y x*y x^2 y^2 x^3 y^3] etc.

While we can get a better fit we could run into the problem of 'high variance; and without the cross validation and test set we will not be able to verify the results, Hence the simpler option [1 x y x*y] was chosen

The Octave code outline and the data used can be cloned from GitHub at https://github.com/tvganesh/ml-bowling-analyze

Conclusion:

1) **Predicted wickets**: The predicted number of wickets is higher at lower economy rates

2) **Comparing performances**: There are different ways of looking at the results. One possible way is to check for a particular number of overs and economy rate who is most effective. Here is one way. Taking a small slice from each bowler's predicted wickets table for a Economy Rate=4.0 the predicted wickets are

Wickets taken for an Economy rate=4.0			
Overs	Chandra	Kumble	Kapil
20	1	2	2
30	2	2	4
40	3	2	5
50	3	2	7

From the above it does appear that Kapil is definitely more effective than the other two. However one could slice and dice in different ways, maybe the most economical for a given numbers and wickets

combination or wickets taken in the least overs etc. Do add your thoughts. comments on my assessment or analysis

Further reading

I would like to thank you for reading this book of mine. For more information or for similar articles please read my blog Giga thoughts - https://gigadom.wordpress.com/. You can also get in touch with me at tvganesh.85@gmail.com

Important Links

1. ESPN Cricinfo Statsguru - http://stats.espncricinfo.com/ci/engine/stats/index.html
2. ESPN Cricinfo Terms of Service - http://www.espncricinfo.com/ci/content/site/company/terms_use.html
3. Github link for R package cricketr - https://github.com/tvganesh/cricketr
4. My Github link - https://github.com/tvganesh
5. Interactive Shiny app Sixer - https://gigadom.wordpress.com/2015/11/29/sixer-r-package-cricketrs-new-avatar/
6. A short video tutorial on my R package cricketr - https://www.youtube.com/watch?v=q9uMPFVsXsI

ABOUT THE AUTHOR

Tinniam V Ganesh is a pioneer-architect-programmer with 27++ years' of experience in software. He is the author of '**cricketr**' R package, now available at CRAN. His interests span Telecom, Distributed Systems, AI, Machine Learning, Computer Vision, Cloud computing, Mobile apps, Data Science and Economics.

Tinniam V Ganesh did his B-Tech in Electronics from IIT-BHU, Varanasi, 1985 and Masters in Computer Science from Illinois Institute of Technology, Chicago in 2002. He recently completed the Data Science Specialization from John Hopkins University, Bloomberg School of Public Health at Coursera. He is currently working in IBM as a Cloud Architect in Cloud CoE.