Baseball Data Analysis

1. Introduction

The data that we pulled had a lot of information. The file **readME2014.txt** has information about the CSV files and basic information of all the CSV files. The following analysis was not on all the CSV files. The files that was put at the centre of all the analysis was **Salary.csv** because a lot of the analysis is centered around the salary. The data cleaning operations were performed on what is present on the **Salary.csv** file. On the way of exploring the Baseball data, some other csv files are also explored, i.e,

- **Master.csv**
- **Teams.csv**
- **Batting.csv**
- **Pitching.csv**

All the CSVs are given with the project files. Download links of all the versions are provided in GitHub README in the form of **PDF, HTML, Markdown, iPython Notebook and JPEG images for plots .**

The code is strictly written in **Python 3.6** with a conda environment. It is recommended to run the code is **Python > 3.4** and using **Python2** is not at all recommended.

NOTE: The actual ipynb file may not have all the details in written and the styling may not kick in as it is in PDF or MarkDown. If the intention is to skim through the analysis only instead of running the code, the PDF version is recommended.

First get all the imports down the line. I am also configuring my own css file to improve the look and feel of the Jupyter Notebook. The CSS file is present in styles directory in the project.

Now before we start lets first see what are the questions that we can answer about the data set from the analysis. These question are of various type and varies from different range and different levels. **Hence after every question, the level of question is also mentioned in bold.** The questions are

1. How teams invested in players in terms of player's salary? **(Basic)**

2. How players received salary over their career? **(Basic)**

3. What are the most common salary ranges teams preferred to compensate their players? This reveals the salary standard of Baseball.**(Intermediate)**

4. Is there any abrupt changes (ups/downs) teams faced in terms of the salary they paid? This shows if any year was good or bad for Baseball players or was their any effect of recessions on players' salary.**(Intermediate)**

5. Is there any abrupt changes (ups/downs) in players' performance in any year and what could be the probable reason? **(Intermediate)**

6. How batters' home-runs (HR) or a pitchers' shutouts (SHO) is affected by their height and weight? **(Intermediate)**

7. How salary affects players' performance? **(Advanced)**

8. What makes a team secure first rank or at least be in top 3? **(Advanced)**

```
%config InlineBackend.figure_format = 'retina'
import seaborn as sns
import matplotlib.pylab as plt
import pandas as pd
import numpy as np
from IPython.display import display
from IPython.core.display import HTML
###################################
### We are supressing all warnings with an assurance that the
#  warnings that are supressed are are not substantial #
###########################################
import warnings
warnings.filterwarnings('ignore')
###################################
######### Our custom import #######
from core import configure
from core import _rc
```

Configuring our RC file is important to give plots my own customized look and feel. For more info on **rcParams** and visual parameter configuration,please click **Here**

```
configure(configType='css')
configure(configType='rc')
```

Now some basic path resolve according to the project directory. **Here** is the documentation for os module for python.

```
###### Resolve the path of the data source #######
ROOT = r'../res/baseball'
# -----  team data ----- #
from core import BaseballModel
model = BaseballModel(ROOT)
salary = model.salary
master = model.master
```

2. Let's start the Data cleaning !!

The data cleaning that we are going to perform would be based on **Salary.csv**. If we open the **Salary.csv**, we see no players' name but only their IDs. Now we would need players' name in furute. The players' name is only available in **Master.csv**. Hence we need to pullin the corresponding names of the players along with their IDs. Moreover in the salary csv file, entries are noted only from **1985** to **2014**. Salary info of any other year previous to this is not present, may be because the recodings of Salary is started from **1985** and not prior to that. Hence players who used to play prior to **1985** had no entry in the Salary.csv file. Hence we need to take only those entries whose record can be found in Salary.csv.

```
## Creating data_1 and pulling the fields form csvs that we need only
required_master_cols = ['playerID', 'nameFirst', 'nameLast',
                        'weight', 'height', 'bats', 'throws']
data_1 = salary.merge(master[required_master_cols], on='playerID',
                                                    how='inner')
data_1['fullName'] = data_1['nameFirst'] + '_' + data_1['nameLast']
```

This plot is the basic plot to show a bargraph of how the teams spent on player's salary. This also ensures the data cleaning was performed correctly.

```
###### First we want to see how teams invested in their player #########
##### Clean and extract the fields we require #########

plt_data = data_1.groupby('teamID', as_index=False)['salary'].sum()
plt_data.sort_values(['salary'], ascending=False, inplace=True)
plt_data = plt_data.reset_index().drop('index', axis=1)

#########################################################
#~~~~~~~~~~~~~~~~  NOW PLOT THE DATA ~~~~~~~~~~~~~~~~~~#
ax = sns.barplot(x="teamID", y="salary", data=plt_data, palette="Greens_d",
                 hatch='oo')
locs, labels = plt.xticks()

ax.set_xlabel('All Teams',
              {'weight': 'bold', 'fontsize': 22,
               'color':'darkgreen'})
ax.set_ylabel('Money Spent in Billions\n',
              {'weight': 'bold', 'fontsize': 22,
               'color':'darkgreen'})
sns.despine(left=True, bottom=True)

ticks = ax.get_yticks() / 1000000000
ticks = [*map(str, ticks)]
ticks = [*map(lambda e: '$ ' + e + 'B', ticks)]
ax.set_yticklabels(ticks,
                   {'fontsize': 16, 'color':'grey'})

ax.set_title("\nTotal Money spent by all teams for 1985 to 2014",
             {'weight': 'bold', 'fontsize': 30})

plt.setp(labels, rotation=45)
for p in ax.patches:
    height = p.get_height()
    ax.text(p.get_x() + p.get_width()/2., 1.01*height,
                '$ %.3f B' % (height / 1000000000),
                ha='left', va='bottom',
                weight='bold', color='green',
                size=12, rotation=45)
plt.show()
```

The avobe plot deals with all the teams from 1985 to 2014. Teams active prior to this is visible because of our data cleaning according to Salary.csv file.

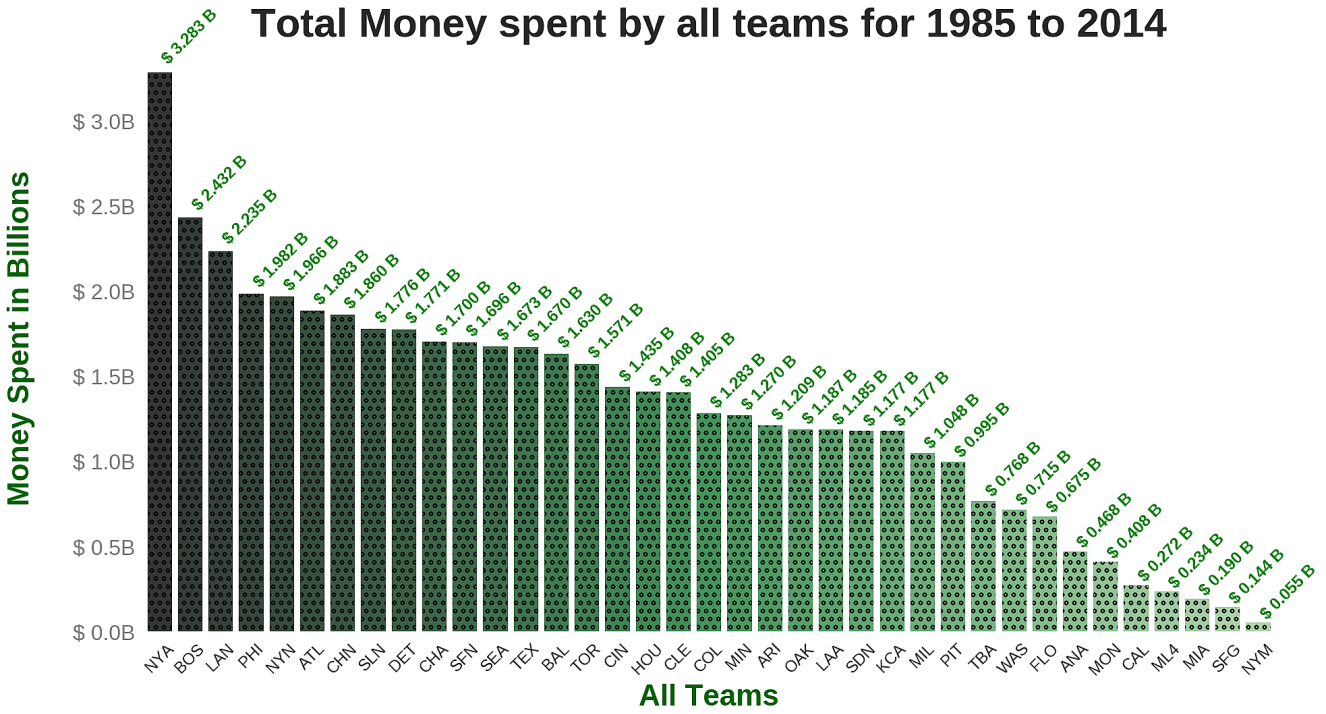


Figure 1: Team's Spending

Now we know that **NYA, BOS, LAN, PHI, NYN** are the top 5 teams in terms of spending money on their players. The next thing that we should do is to check what are top **players** in terms of salary and logically they should be from **these top teams**.

The following section shows the top 5 players in terms of total salary and the teams they played for throughout their career. This also proves the above data wrangling was performed correctly.

Note that **their_clubs** is not an **OrderedDict** and hence the 5 players mentioned are among to 5 but may not be in any ascending or descending order. To check top 5 players in descending order, please visit the following barchart of top 20 players.

Btw, here are the actal names of these top 5 teams

- **NYA** = New York Highlanders
- **BOS** = Boston Americans
- **LAN** = Los Angeles Dodgers
- **PHI** = Philadelphia Quakers
- **NYN** = New York Mets

```
plot_data = data_1.groupby(['fullName'], as_index=False)['salary'].sum()
plot_data.sort_values(['salary'], inplace=True, ascending=False)
five_most_expensive_players = plot_data.fullName.head(n=5)

###########  this part is important for later use   ##########
# It stores the top 10 players and their teams they played #

g=data_1[data_1.fullName.isin(five_most_expensive_players)].\
                            groupby(['fullName', 'teamID'])
tc = [k for k, gr in g]
from collections import defaultdict
their_clubs = defaultdict(list)
for player, team in tc:
    their_clubs[player].append(team)
their_clubs
#################################################################
```

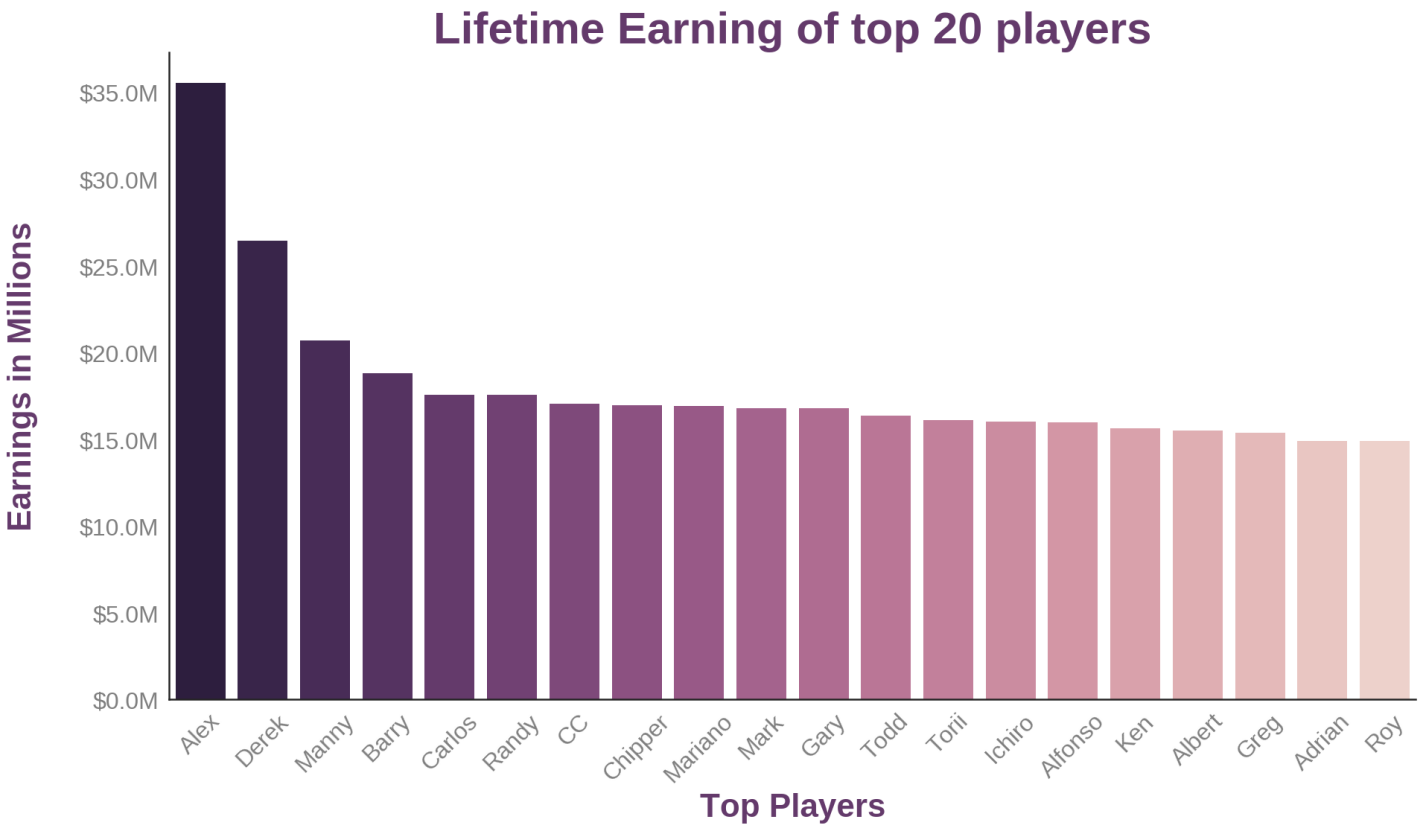
This prints,

```
defaultdict(list,
               {'Alex_Rodriguez': ['NYA', 'SEA', 'TEX'],
                'Barry_Bonds': ['PIT', 'SFN'],
                'Carlos_Beltran': ['KCA', 'NYA', 'NYN', 'SLN'],
                'Derek_Jeter': ['NYA'],
                'Manny_Ramirez': ['BOS', 'CLE', 'LAN', 'TBA']})
```

and here is the chart of top 20 players in terms of their career earnings.

```
# getting only the top 20 players
ten_most_expensive_players = plot_data.fullName.head(n=20)
plot_data = plot_data[plot_data.fullName.isin(ten_most_expensive_players)]
# fixing the structure for seaborn plot
plot_data = plot_data.reset_index().drop('index', axis=1)
# Plotting the top 10 players #
ax = sns.barplot(x="fullName", y="salary", data=plot_data,
                 palette=sns.cubehelix_palette(20, reverse=True),
                 saturation=1)
# printing only the firstName
xlabels = [*map(lambda e: e.split('_')[0], plot_data.fullName)]
ax.set_xticklabels(xlabels,
                   {'fontsize': 16, 'color':'grey'})

ticks = ax.get_yticks() / 10000000
ticks = [*map(str, ticks)]
```

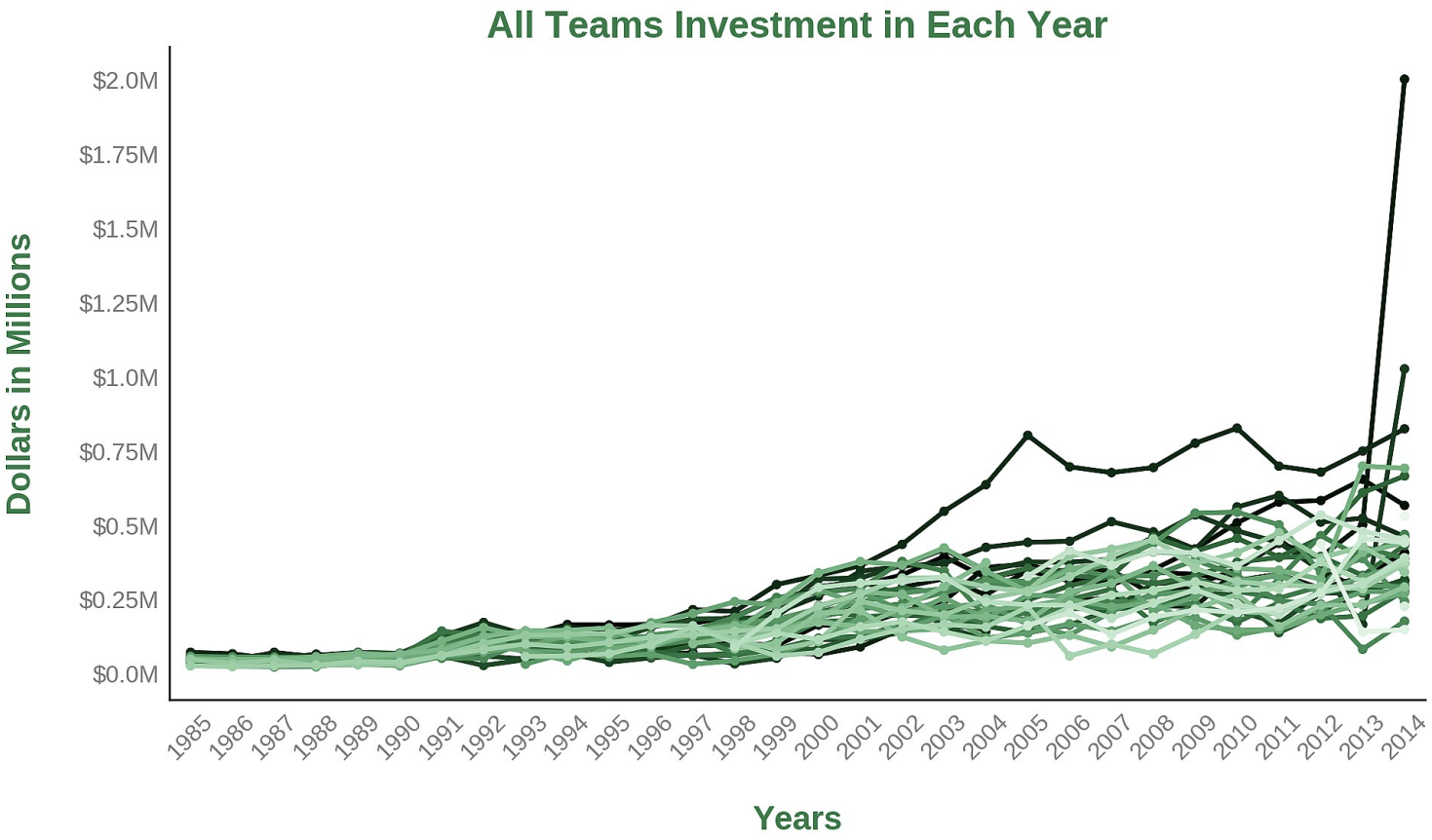
```
ticks = [*map(lambda e: '$' + e + 'M', ticks)]


ax.set_yticklabels(ticks,
                   {'fontsize': 16, 'color':'grey'})
ax.set_xlabel('Top Players',
              {'weight': 'bold', 'fontsize': 22,
              'color':sns.cubehelix_palette(20, reverse=True)[4]})
ax.set_ylabel('Earnings in Millions\n',
              {'weight': 'bold', 'fontsize': 22,
               'color':sns.cubehelix_palette(20, reverse=True)[4]})
ax.set_title("Lifetime Earning of top 20 players",
              {'weight': 'bold', 'fontsize': 30,
              'color':sns.cubehelix_palette(20, reverse=True)[4]})
plt.xticks(rotation=45)
sns.despine()
plt.show()
```



3. Were they all thrifty ??

The above distribution is skewed and it is the distribution of salary. This shows the most density distribution is located around *0 to 5000000*. This proves that high salaries were not so frequent. What does this shows?? This shows a common human nature; **most people are thrifty when it comes it money, at least the baseball teams were.**

```
rot = sns.cubehelix_palette(len(data_1.teamID.unique()), start=2,
                            rot=0, dark=0, light=.95, reverse=True)
ax = sns.pointplot(x="yearID", y="salary", data=data_1,
                   hue='teamID', palette=rot, ci=None,
                   markers='.', join=True)
ax.legend_.remove()
locs, labels = plt.xticks()
plt.setp(labels, rotation=45)
sns.despine()
ax.set_xticklabels(np.sort(data_1.yearID.unique()),
                   {'fontsize': 16, 'color':'grey'})
ax.set_xlabel('\nYears',
              {'weight': 'bold', 'fontsize': 22
              , 'color':rot[15]})
ax.set_ylabel('Dollars in Millions\n',
              {'weight': 'bold', 'fontsize': 22
              , 'color':rot[15]})
ticks = ax.get_yticks() / 10000000
ticks = [*map(str, ticks)]
ticks = [*map(lambda e: '$' + e + 'M', ticks)]
ax.set_yticklabels(ticks,
                   {'fontsize': 16, 'color':'grey'})
ax.set_title("All Teams Investment in Each Year",
             {'weight': 'bold', 'fontsize': 25, 'color':rot[15]})
plt.show()
```



4. Did teams think similar when it comes to money ??

The above timeseries shows how people liked to invest over the passage of time. Each teams represent a single line. The idea is not to show how each teams are investing over time, rather it shows what is the investment pattern of all the teams. We can see that upto *2001* all the teams invested almost similarly. Beyond *2001* to around *2010* the diffence between the investment of teams increased substantially with some team investing a lot than others. But beyong *2011* this difference got really big and we can also see the top 3 highest investments were made in the year of *2014*. For example, we can see the the one team at *2005* invested a lot more than others and kept on investing till the end. The following code shows this club name an the money that they invested.

```
data_1[data_1.yearID == 2005].groupby('teamID',
        as_index=False)['salary'].sum() \
        .sort_values('salary', ascending=False) \
        .reset_index().drop('index', axis=1).ix[0]
```

This prints,

```
teamID          NYA
salary    208306817
Name: 0, dtype: object
```

This is logical too because we have already seen that `NYA` invested most out of all the teams.

Interestingly, in the year of `2014`, `NYA` didn't remain at number 1 position and it changed to `LAN`. This code shows it below

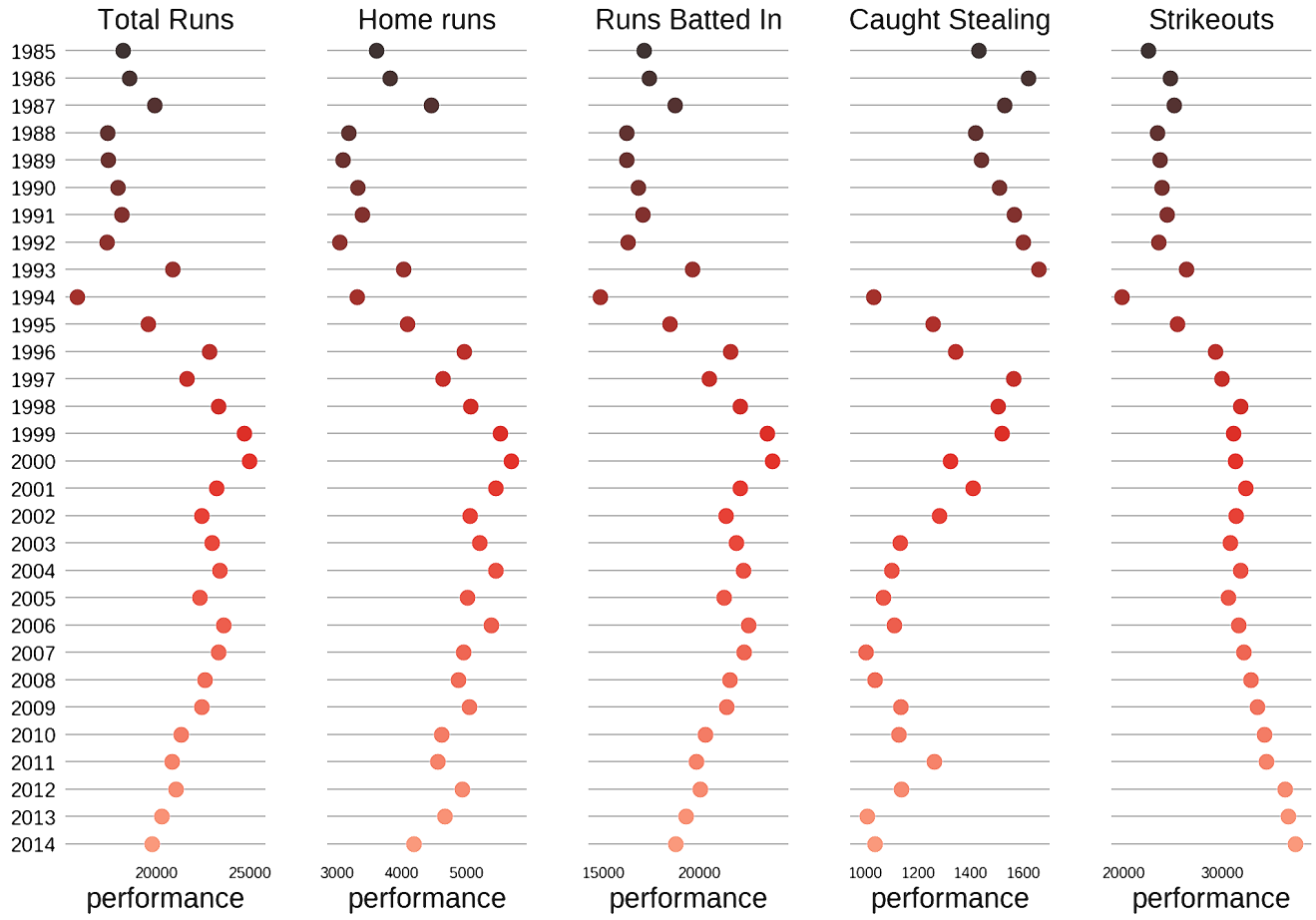
```
data_1[data_1.yearID == 2014].groupby('teamID',
        as_index=False)['salary'].sum() \
        .sort_values('salary', ascending=False) \
        .reset_index().drop('index', axis=1).ix[0:2]
```

this prints,

	teamID	salary
0	LAN	214014600
1	NYA	197543907
2	PHI	180944967

Howerver the trend that we have seen, we can say unless the team is an extraordinary, most teams spent likely over the passage of time.

```
batting = model.batting
BaseballModel.performance_burndown(batting)
```



Now in the above plot I have tried to show how different metric changed over time form 1985 to 2014. We can see for example that there is an abrupt decrese in the total runs scored in the year of 1994 as compared to other years. **Let's verify if the plot shows the right value?** If we execute ***plot_data.R.argmin()*** it returns ***9*** which is the row number of the ***plot_data*** table which is further derived from ***batting***. So to get the year we execute `***plot_data.ix[9][['yearID', 'R']]*** and indeed the year is 1994 and the run is 15752 which can also be verified from the scale. Evidently all other performance of Batters and Pitchers in the year of 1994 was poor as compared to other years. **SO I GUESS 1994 WAS A DARK YEAR FOR BASEBALL FANS..:)**