NBA Data Analytics

Alex Ye, Darryl Vo, Tarrant Starck
CPE 369 Introduction to Distributed Computing

Introduction

Our project is a suite of Spark programs that compute several significant player/team analytical benchmarking algorithms. Each measurement are used within the NBA or widely accepted measurements within the basketball community.

Hadoop Technologies

For this project, we decided to use Spark instead of Map-Reduce solutions for our project. This is for several reasons. Since Spark loads to main memory, we can see a significant increase in speed of our computations when using our 4000+ record file containing player data. This allowed development to be far more rapid than if we had to run against a Map-Reduce cluster solution each time. We were also able to set up local environments on our machines that were able to run the code and handle the size of our dataset even with the data being loaded to main memory. Along with speed, the predictable nature of our data and the functional style of Scala gave us a lot more flexibility when it came to our code.

Dataset

Our data contains NBA data for the past 10 years. The data was programatically scraped from *basketball-reference.com* with Python's BeautifulSoup. The data is contained in directory `/data` and are plain `.txt` format of `.csv` files. Below are the headers for each file and a description of the datasets.

`data/player.txt`

`player.txt` contains data for all NBA players over the past 10 seasons. Each record is a single player's total stats for a single season. The key of the given dataset can be treated as `(name, year)`.

```
Player - Name of player
Pos - Position played
Age - Age
Team - Team played for
GP  - Games played
GS - Games started
```

```
MIN - Minutes
FGM - Field goals made
FGA - Field goals attempted
3PM - 3 point shots made
3PA - 3 points shots attempted
2PM - 2 point shots made
2PA - 2 point shots attempted
FTM - Free throws made
FTA - Free throws attempted
ORB - Offensive rebounds
DRB - Defensive rebounds
AST - Assists
STL - Steals
BLK - Blocks
TOV - Turnovers
PF - Personal Fouls
PTS - Points scored
Year - Tail end year of season (i.e. 2009-2010 season listed as 2010)
```

data/mvps.txt

`mvps.txt` contains data for all players receiving MVP awards over the past 10 seasons. Each record is the player and year the award was awarded.

```
Player - Name of player
Year - Tail end year of season (i.e. 2009-2010 season listed as 2010)
```

data/teams.txt

`Teams.txt` contains data of all teams and their respective geographic and name details.

```
abbrev - Abbreviation of team name
name -  Full name of team
city - Home city of team
state - Home state of team
```

data/champions.txt

`champions.txt` contains data for all teams winning championships over the past 10 seasons. Each record is the player and year the award was awarded.

```
Player - Name of player
Year - Tail end year of season (i.e. 2009-2010 season listed as 2010)
```

Program Internals and Functionality Log

Below is an overview of the main functionality of each of our programs do. We have additional methods within each of the programs that will compute more fine grained detail that we used to withdraw data for our presentation. Each method in the code contains a header that gives a short detail about what the method does.

Running the Code

Our code can be loaded into the spark shell and run from the command line. The code can be run locally without much hassle simply by entering the `‘spark-shell’` and using the command `‘:load <filename>.scala’`. In the case this is being run on a cluster, you must copy the files in `/data` to your HDFS cluster and modify the read locations in the code.

On successful loading of the scripts, each script contains an object with the corresponding statistic that it computes (i.e. `object tsp` is the object for True Shooting Percentage). Each object contains methods that compute the more fine grained calculations. Usage for each of the methods should also be documented in the code. These methods can be run by calling the method in the `‘spark-shell’` interpreter

Player Efficiency Rating (PER) - Alex Ye (~2hr)

Player Efficiency Rating is a rating that is tracked by the NBA to determine a rating that weighs both positive and negative effects of player stats. This is done by taking stats and adding a weight to each significant stat point that corresponds to the weight of impact to the game. The following is the whole formula for the PER calculation.

```
[ FGM x 85.910
+ Steals x 53.897
+ 3PTM x 51.757
+ FTM x 46.845
+ Blocks x 39.190
+ Offensive_Reb x 39.190
+ Assists x 34.677
+ Defensive_Reb x 14.707
- Foul x 17.174
- FT_Miss x 20.091
- FG_Miss x 39.190
- TO x 53.897 ]
x (1 / Minutes).
```

This data allows us to view players at a more holistic ranking other than by a single statistics. The top results from running it against our player dataset yielded predictable results that reflect the current NBA:

Top 10 Average PERs

Player	PER
LeBron James	37.27901966769465
Tim Duncan	32.79047687332209
Dwight Howard	32.25314923276986
Chris Paul	32.249495042425046
Dirk Nowitzki	32.2439429635584
Kevin Durant	31.33155356031596
Pau Gasol	30.274024998506466
Kevin Garnett	30.063141398950922
Dwyane Wade	29.901125550167308
Al Jefferson	29.654174428965728

The code for this can be found in `per.scala` in the root directory.

Effective Field Goal Percentage (EFG%) - Alex Ye (~1 hr)

Effective Field Goal Percentage is a formula to adjust the standard field goal percentage to account for 3 point field goal being worth more than a normal 2 point field goals. This adjustment reward players who are good at shooting beyond the three point line and is a better representation of a *pure shooting statistic*.

The formula is quite simple to calculate:
`(FGM + 0.5 * 3PM) / FGA`

Over all the players, we can see that the top EFG% players are mostly center position players since most of their plays are dunks or layups around the rim:

Top 5 EFG% Over All Players

Player	EFG%

DeAndre Jordan	0.670164675492412
Bo Outlaw	0.6666666666666666
Maceo Baston	0.6644736842105263
Montrezl Harrell	0.6444444444444445
Alan Henderson	0.6417910447761194

When we limit them to shooters (Point Guards and Shooting Guards), we get the following results:

Top 5 EFG% For PG and SG

Player	**EFG%**
Brent Barry	0.6038525963149078
Kyle Korver	0.5977553675992192
Danilo Gallinari	0.576
Steve Nash	0.574859747545582
Stephen Curry	0.5743111436235064

The code for this can be found in `efg.scala` in the root directory.

True Shooting Percentage (TSP%) - Alex Ye (~1hr)

True shooting percentage is a measure of shooting efficiency that takes into account field goals, 3-point field goals, and free throws. This provides a more holistic measurement of *overall shooting percentage*.

This formula is also quite simple to calculate:

```
TSP% = PTS / (2 * TSA)
where
TSA = FGA + 0.44 * FTA
```

This can give us interesting data. For instance, we are able to see that this statistic greatly reduces the field goal percentage for many center position players. Since many center positioned players are terrible free throw shooters, we can see the way this statistic works when we sort the output by difference in actual field goal percentage by by the true shooting percentage in the table below.

Players with Largest Difference in FG% to TSP%

Player	TSP%	FG%	FT%	Diff (FG% - TSP%)
Justin Williams	0.5660214310404	0.6136363636363	0.3650793650793	0.0476149325959
DeAndre Jordan	0.6227325338235	0.6700032289312	0.4207105719237	0.0472706951076
Clint Capela	0.5532418748979	,0.581863979848	0.3791469194312	0.0286221049509
Michael Ruffin	0.50713330361123	0.5322580645161	0.3968253968253	0.0251247609048
Joey Dorsey	0.51598084886128	0.5409836065573	0.3716216216216	0.0250027576960

The code for this can be found in `tsp.scala` in the root directory.

Usage Percentage (USG%) - Darryl Vo (~2 hrs)

Usage percentage is an estimate of the percentage of team plays used by a player while he was on the floor.

The formula for this calculation is the following:
`100 * ((FGA + 0.44 * FTA +TOV) * (Tm MP / 5)) / (MP * (Tm FGA + 0.44 * Tm FTA + Tm TOV)).`

Basically a stat for how much of a team's possessions is used by a player. I expected to see the top well known players to have the highest USG%, and our results support that hypothesis.

Top 5 avg USG% across their entire career

Player	USG%
Kobe Bryant	32.722217585187714
Carmelo Anthony	32.42367438948769
Dwyane Wade	32.357325868679126
LeBron James	31.875205033382297
Russell Westbrook	31.20551115599283

The code for this can be found in `usg.scala` in the root directory.

Per 48 - Tarrant Starck (~2hrs)

The Per48 Statistic is calculated by taking the stat, dividing by the minutes they played that season, and multiplying it by 48. 48 is used as it is the length of an NBA game. This stat can

allow comparisons of players with different amounts of minutes played. It also helps show how effective a player is when they play and how their effectiveness changes over time.

For example, here is

Stephen Curry's Per 48 For 3 Points Made Over Time

Year	**3PM per48**
2010	2.751
2011	2.91
2012	3.6
2013	4.37
2014	4.4
2015	5.25
2016	7.14

This Table makes it obvious that Curry's 3 Point Game has been improving over time. It shows that he is making over twice as many 3's per 48 minutes in 2016 than he did in 2010.

The code for this can be found in `Per48.scala` in the root directory.