NBA Arbitrary Team P.K.: Build Your Dream Team and Fight!

Tian Lan
Department of Computer Science
Texas A&M University
College Station, Texas - 77840
tianlan@cs.tamu.edu

Yukun Gao
Department of Computer Science
Texas A&M University
College Station, Texas - 77840
gaoyukun@cs.tamu.edu

Yanming Zhou
Department of Computer Science
Texas A&M University
College Station, Texas - 77840
yanmingzhou@cs.tamu.edu

ABSTRACT

The National Basketball Association (NBA) is the pre-eminent men's professional basketball league in North America, and is widely considered to be the premier men's professional basketball league in the world. Many basketball fans choose their favorite teams mainly because the team has their favorite players. However, the truth is that many times people's favorite players cannot play for the same team, because of drafts order, financial issue and interest conflict, etc. This is exactly where our project aims at.

In our project, we endow users with the power to set up their own team by letting them choose whichever NBA player they wish. To make the project more exciting, we add the element of combating so that we turn it into a P.K. game where different personalized teams could fight with each other to determine which one is better. The decision is made based on the ranking score calculated by our algorithm using historical statistics of each player/team retrieved (crawled) from NBA stats websites.

Keywords

NBA, SVM, C tuning, Feature Selection, Crawler, Web Application

1. INTRODUCTION

National Basketball Association (NBA) is the men's professional basketball league in North America. The influence of NBA trespass its borders and have countless fans around the world.

As the league involves a lot of money and fans, not surprisingly, a lot of games have been developed to let players build up teams and play against each other. However, they typically not fully support game players to build the team at their will and their scoring systems are simply. Thus, we designed an online NBA game that supports free player chosen and has a more complicated scoring system which is more realistic.

Through the years a lot of data and statistics have been collected based on NBA and each day the data become richer. However, even with such rich data available, it is still very complex to analyze. In order to deal with that complexity and to achieve better scoring system. Machine learning method must be adapted to analysis the relationship between tens of players' (teams') features and their performance, rank, game results. We also need to weight them carefully to build up the scoring system.

The reminder of the paper is organized as follows. Section 2 describes the scoring system. Section 3 introduces the implementation details about the game. Section 4 includes the experiment result of scoring system and given the tutorial of web application.

2. SCORE SCHEME

2.1 Data Preprocess:

The data-set should be studied, selected and organized. The raw data are divided into two categories: Data of Teams, which includes features and see Table1 and Data of individual players, see Table2. It is important to mention that for team data only the past 3 years will be used, which means the study of correlation between team and its players are almost unchanged during the past 3 years.

For the individual players' data, we categorized them into 5 groups according to their position. That is because different positions value different features. In this way the features selection and weighting would be more realistic than treating all the players as one group.

Also, with respect to the individual players, we designed survey to collect the feedback of feature importance from real NBA fans.

2.2 Feature selection and weighting:

There are 26 raw features for individual player. And some of them are directly correlated with each other, such as the FG3_PCT which represent the 3-point Field Goal Percentage, is equal to FG3M divided by FG3A. These features should be treated as redundant features and should be pruned out. Otherwise, it could cause overweighting of one actual feature.

After features prune, only 6 raw features are stored, then we added 6 self-defined feature "". Then we label players' raw score, separated by their position, with the Holllinger Per (Player efficiency Rating) value.

$$PER = \left(uPER \times \frac{lgPace}{tmPace}\right) \times \frac{15}{lguPER} \quad (1)$$

If the per value exist in NBA official website, we assign them to players. If not, we calculate them by the equation [1]. Note that, we made some assumption when self-calculated the Per value in team factor, since the team information we collected are past 3 years which may have some information lose for players that changes the team or even league.

After labeling the players data, we feed them into the SVM. In the project, we chose SVM with linear kernel. The test sets are built up with TOP 10 players in each position from Fox NBA websites. Based on the accuracy, we chose penalty C carefully. And on average we get the 80% accuracy in each position. Finish the SVM ranking and weighting we get the ranked features together with their weights for the individual player.

Then we feed the teams information into the SVM, trying to select one extra features which is related to the place of game, say the "home" and "away". These data need not to be labeled, since they have win or lose result themselves. Thus we chose 60% as the training data and the left 40% as the testing data. Also, we tuned C carefully to get the good accuracy.

Since we have carried out the survey of features importance among the NBA fans, we collected the data and modify the features accordingly. In this way, we eliminate the bias raised by only using "cold" data to evaluate the player.

2.3 Scoring Scheme:

The final score mainly compose three parts, the individual players' contribution, the team place contribution, and another part, the Vector Space Similarity between the team built by a player and the real team. In this way, we try to match the self-build team with real team and fetch the score information and weight it into our final score.

The total score system is described in Fig.1.

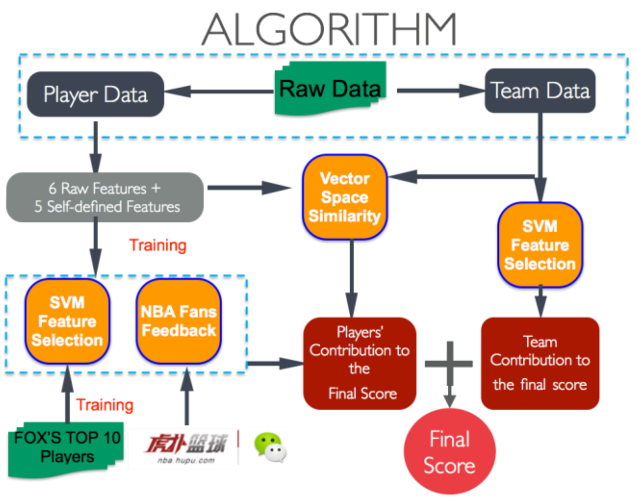


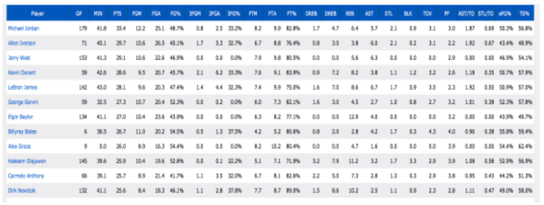
Figure 1. Scoring scheme

3. IMPLEMENTATION

3.1 Data preparation

3.1.1 Data source

Our data mainly comes from NBA official website [2], the office website of NBA, and "basketball reference" [3], a website about basketball statistics and history. We have get statics of 1100 historical players and 6000 Games for 30 teams in the league in the past three years. The media resource used in our project such as the headshot of players, logo of teams are also come from the above websites. Figure 1 show the data from NBA official website.



Player	GP	MIN	PTS	FGM	FGA	FG%	3FGM	3FGA	3FG%	FTM	FTA	FT%	OREB	DREB	REB	AST	STL	BLK	TOV	PF	AST/TO	STL/TO	eFG%	TS%
Michael Jordan	179	41.8	33.4	12.2	25.1	48.7%	0.8	2.5	33.2%	8.2	9.9	82.8%	1.7	4.7	6.4	5.7	2.1	0.9	3.1	3.0	1.87	0.69	50.3%	56.8%
Allen Iverson	71	45.1	29.7	10.6	26.5	40.1%	1.7	5.3	32.7%	6.7	8.8	76.4%	0.8	3.0	3.8	6.0	2.1	0.2	3.1	2.2	1.92	0.67	43.4%	48.9%
Jerry West	153	41.3	29.1	10.6	22.6	46.9%	0.0	0.0	0.0%	7.9	9.8	80.5%	0.0	0.0	5.6	6.3	0.0	0.0	0.0	2.9	0.00	0.00	46.9%	54.1%
Kevin Durant	59	42.6	28.6	9.5	20.7	45.7%	2.1	6.2	33.3%	7.6	9.1	83.9%	0.9	7.2	8.2	3.8	1.1	1.2	3.2	2.6	1.18	0.35	50.7%	57.9%
LeBron James	142	43.0	28.1	9.6	20.3	47.4%	1.4	4.4	32.3%	7.4	9.9	75.0%	1.6	7.0	8.6	6.7	1.7	0.9	3.5	2.3	1.92	0.50	50.9%	57.0%
George Gervin	59	32.5	27.3	10.7	20.4	52.3%	0.0	0.2	0.0%	6.0	7.3	82.1%	1.6	3.0	4.5	2.7	1.0	0.8	2.7	3.2	1.01	0.38	52.3%	57.8%
Elgin Baylor	134	41.1	27.0	10.4	23.6	43.9%	0.0	0.0	0.0%	6.3	8.2	77.1%	0.0	0.0	12.9	4.0	0.0	0.0	0.0	3.2	0.00	0.00	43.9%	49.7%
Billyray Bates	6	36.5	26.7	11.0	20.2	54.5%	0.5	1.3	37.5%	4.2	5.2	80.6%	0.8	2.0	2.8	4.2	1.7	0.3	4.3	4.0	0.96	0.38	55.8%	59.4%
Alex Groza	9	0.0	26.0	8.9	16.3	54.4%	0.0	0.0	0.0%	8.2	10.2	80.4%	0.0	0.0	4.7	1.6	0.0	0.0	0.0	3.9	0.00	0.00	54.4%	62.4%
Hakeem Olajuwon	145	39.6	25.9	10.4	19.6	52.8%	0.0	0.1	22.2%	5.1	7.1	71.9%	3.2	7.9	11.2	3.2	1.7	3.3	2.9	3.9	1.08	0.58	53.0%	56.9%
Carmelo Anthony	66	39.1	25.7	8.9	21.4	41.7%	1.1	3.5	32.0%	6.7	8.1	82.6%	2.2	5.0	7.3	2.8	1.3	0.3	2.9	3.8	0.95	0.43	44.2%	51.3%
Dirk Nowitzki	132	41.1	25.6	8.4	18.3	46.1%	1.1	2.8	37.8%	7.7	8.7	89.0%	1.5	8.6	10.2	2.5	1.1	0.9	2.3	2.8	1.11	0.47	49.0%	58.0%

Figure 2. NBA official website data table

3.1.2 Scrawl Tool - Scrapy

We use Scrapy to crawl data from the Internet. Scrapy is a web crawling framework with support for web scraping. It is open-source and written in Python. Scrapy is light and quick, it can get content of websites according to the elements' xpath, suitable for our project.

3.1.3 Data Processing

The data we get is in json format, which needs some processing to be better adapted for the Django framework. It's not hard, just some string manipulation. Also the filename of headshot need to be same with corresponding player's name. We make a modification on our crawl program's pipeline to achieve the goal.

Scrapy can easily get static data from websites, but some core data we need is dynamic, which is generated by javascript. Using common method we can get nothing. And the office document for Scrapy is stale, I spent many time on it but get nothing. To address this problem, at first we do it in the manual way, we call it "silly method". We copy the dynamic code of the website (They cannot be captured by Scrapy but can be seen by firebug), then paste them to my own server. So we can use Scrapy to get the data from my server easily. This method is simple but useful, especially when the data is not so large. For some data not suitable to get in this way, we use data in "basketball reference" instead. Content in that website is static. But then another difficulty appeared: some players' name are different between the two website (e.g. J.J. Redick and Jonathan Clay Redick). So we abandon this method. At last we address this problem by combine Scrapy with webkit, to simulate the behavior of a browser. By this way we could crawl the data we need.

3.2 Web Appliaction

3.2.1 Django – the web application framework:

As the algorithm and the data retrieval is written in Python, in order to better facilitate the implementation of the website, we selected to use the Django framework, which is a high-level Python Web framework that encourages rapid development and clean, pragmatic design[1]. Django has many advantages as a web framework. With object-relational mapper, we were able to represent all our data by three models: Player, Team, TeamGame and thus process data in a more elegant and efficient manner. Django provides an automatic admin interface which we used to add and change our data more conveniently. The elegant URL design feature of Django gives us the ability to map URL names to a reasonable page and the template system feature separate our HTML/CSS/JS files from the python code, which provides us a cleaner development environment.

3.2.2 SQLite database – the light-weight database:

SQLite database is built-in with the Django framework. SQLite provides an excellent development alternative for applications that are predominantly read-only or require a smaller installation footprint [1]. To load our crawled JSON data into the SQLite database, we utilized the data fixture feature of Django, where we provide initial data fixtures (JSON file) and Django framework will automatically load all the data from the fixtures into the SQLite database, which saves us a lot of time on dumping data.

3.2.3 Bootstrap – the front-end framework:

For the design of our website, we use the Bootstrap framework, a sleek, intuitive, and powerful mobile first front-end framework for faster and easier web development [2]. Bootstrap provides a lot of predefined HTML, CSS and JS tags and standards, following which we could write our code in a cleaner and more comprehensive way. Also noticeable is that Bootstrap has the self-adaptive ability such that our website is self-adaptable for mobile devices with smaller screens.

4. RESULT

4.1 The website

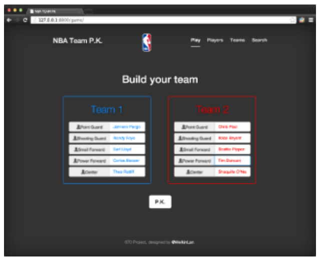
We built a website to implement our game “NBA Arbitrary Team P.K.”. The website contains a navigation bar of four main parts: “play”, “players”, “teams” and “search”.

The “play” part is where the game actually goes (Figure 3). The user is prompted to input 10 players, 5 for each team (blue and red). Then the user clicks the “P.K.” button to play the game (Figure 3(a)). After the data is verified, the website runs our algorithms to calculate the score for each team and display the result page to the user (Figure 3(b)).

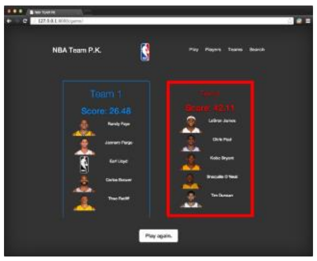
The “players” section is for the user to navigate through all 1100 NBA players to get ground truth of them (Figure 4). The user is able to filter the players by their positions (center, power forward, small forward, point guard, shooting guard) and sort the players by a list of futures (e.g. points, rebounds, assists). The pagination function is also included. (Figure 4(a)). By clicking a player’s name or avatar, the user is navigated to the player detail page, where the data (24 features) of that player is displayed and the score calculated by our algorithm is provided. (Figure 4(b))

The “teams” section is for the user to browse NBA team data (Figure 5). The main page contains all 30 teams from 6 different districts (Figure 5(a)). The user will be directed into the team detail page, where he could find all the game data (12 features) for each team in recent 3 years. Pagination is also provided here (Figure 5(b)).

The “search” part serves as a portal for the user to retrieve all the players (Figure 6). The user inputs the player name and hits the “Go!” button, and then he will be directed to the player detail page.

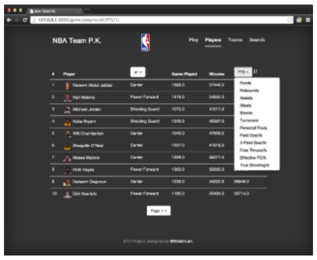


a. main page

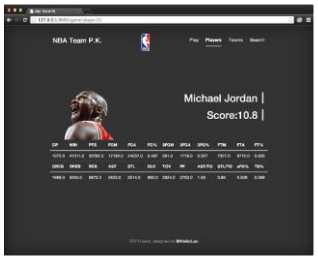


b. result page

Figure 3. “play” section

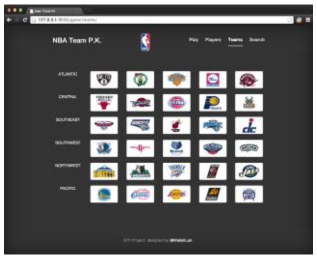


a. main page

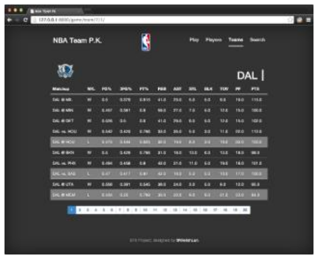


b. player detail page

Figure 4. “players” section



a. main page



b. team detail page

Figure 5. “teams” section

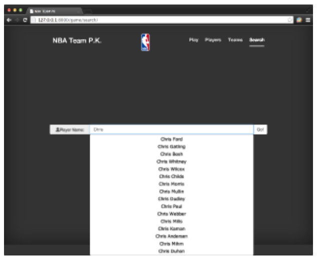
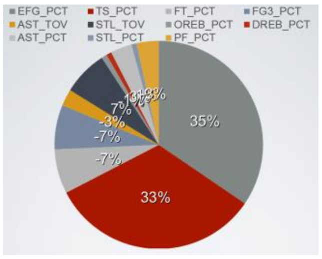
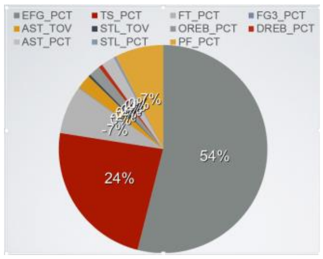


Figure 6. “search” section

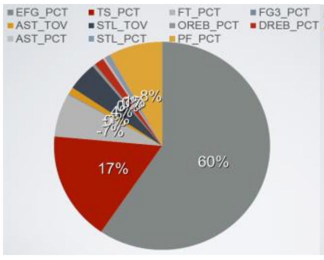
4.2 Feature weighted result



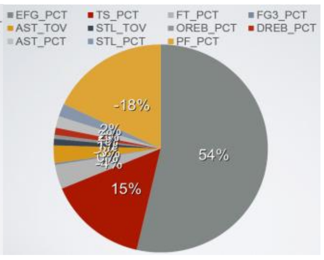
a. Shooting Guard



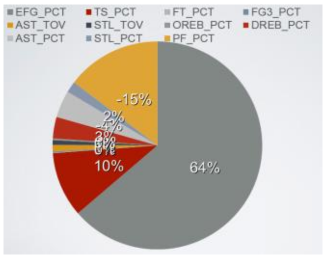
b. Point Guard



c.Small Forward



d.Power Forward



e. Center

Figure 7. Future weight

5. CHALLENGES

5.1 Data preparation

Scrapy can easily get static data from websites, but some core data we need is dynamic, which is generated by javascript. Using common method we can get nothing. And the office document for Scrapy is stale, I spent many time on it but get nothing. To address this problem, at first we do it in the manual way, we call it "silly method". We copy the dynamic code of the website (They cannot be captured by Scrapy but can be seen by firebug), then paste them to my own server. So we can use Scrapy to get the data from my server easily. This method is simple but useful, especially when the data is not so large. For some data not suitable to get in this way, we use data in "basketball reference" instead. Content in that website is static. But then another difficulty appeared: some players' name are different between the two website (e.g. J.J. Redick and Jonathan Clay Redick). So we abandon this method. At last we address this problem by combine Scrapy with webkit, to simulate the behavior of a browser. By this way we could crawl the data we need.

5.2 User Experience

To provide smooth user experience for our users, we designed the UI of our website to be precise and clean. Every section of the website has a primary function, which helps the user to faster locate his needs. Meanwhile, our website handles user input exceptions well. For example, in the "play" section, if the user haven't fill in all 10 player names or if he/her inputs a name that does not exists in our player pool, then our validation system will show proper prompts telling him/her what and where the problem is.

6. CONCLUSION AND FUTURE WORK

In this project, we develop a scoring scheme that is more realistic and build up a NBA arbitrary team web application. The result could achieve average 80% accuracy.

We plan to implement a new algorithm which finds the most similar existing teams with the teams our user builds, so that we could use the existing game data as ground truth. Also we may build a SNS website where users could register and set up their own user profile. They could share their results or compare with others, etc.

This project provide us a great chance to try out prevalent tools such as Scrapy, Django, SQLite Database, Boot-strap, python. Also our teamwork ability are cultivated and developed. Thanks again to Dr.Caverlee for guiding us during this great course.

7. REFERENCES

[1] Django framework, www.djangoproject.com/

[2] Basketball reference, www.basketball-reference.com

[3] NBA statistics website, stats.nba.com

[4] Bootstrap framework, getbootstrap.com/

[5] L Hoffman, A Multivariate Statistical Analysis of the NBA

[6] Osama K. Solieman, Data Mining in Sports: A Research Perspective.

[7] http://www.nbageneralmanagerthegame.com/

[8] http://scikit-learn.org/stable/