SSW810_Final Project
Author: Yulong Yan
Github URL: https://github.com/yyan27/SSW810-Final_project

1 Background information and objectives

In recent years, as team teams such as the Rockets (2018-2019), The Suns (2020-21), and thunder (2019-20) have shined on the playoff stage, the traditional star lineup has not been able to lift the O'Brien Cup smoothly, and the Warriors have even lost in Toronto in the case of the Big Three. This phenomenon has led to a growing number of experts, scouts, and even academia focusing on the NBA. More and more people want to analyze what is causing traditional ball teams to perform less well than usual in the current league. One perspective is through the method of data analysis. A successful precedent is an American data analytics company called Spectrum Second, which modeled detailed data for NBA teams and players to analyze the true performance of each player. In fact, today's NBA in the statistics are also more and more professional, fine-grained, traditional scoring rebounds, assists and other surface data often can not express the real performance of a player, such as Drummond Green, plus and minus, scoring efficiency, assists to error ratio and other high-level statistics appear more and more in front of people's eyes.

With the rapid development of machine learning technology, statistics-based models seem to be able to play a role in today's increasingly large-scale NBA data. With this in mind, this experiment provides a detailed analysis of the actual performance of today's NBA players through data capture, data analysis, visualization and other processes.

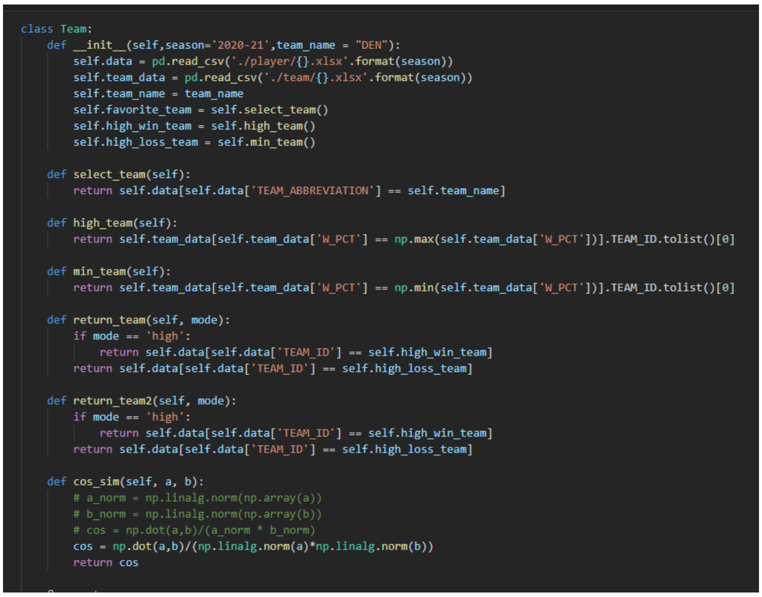
2 Data acquisition and pre-processing

This experiment captures the NBA's regular season data for the past three years by writing python, including the data of each player, some of the fields are as follows: player ID, player name, player belonging team, team ID, scoring, rebounding, assists, etc.; also includes the record of 30 NBA teams in the past three years.

The captured data is stored in a structured format in .csv format, and then stored in the sqlite3 database through Datagrip.

3 Data query and modeling

Write data classes through Python to load and query NBA data for each season.



```python
class Team:
    def __init__(self,season='2020-21',team_name = "DEN"):
        self.data = pd.read_csv('./player/{}.xlsx'.format(season))
        self.team_data = pd.read_csv('./team/{}.xlsx'.format(season))
        self.team_name = team_name
        self.favorite_team = self.select_team()
        self.high_win_team = self.high_team()
        self.high_loss_team = self.min_team()

    def select_team(self):
        return self.data[self.data['TEAM_ABBREVIATION'] == self.team_name]

    def high_team(self):
        return self.team_data[self.team_data['W_PCT'] == np.max(self.team_data['W_PCT'])].TEAM_ID.tolist()[0]

    def min_team(self):
        return self.team_data[self.team_data['W_PCT'] == np.min(self.team_data['W_PCT'])].TEAM_ID.tolist()[0]

    def return_team(self, mode):
        if mode == 'high':
            return self.data[self.data['TEAM_ID'] == self.high_win_team]
        return self.data[self.data['TEAM_ID'] == self.high_loss_team]

    def return_team2(self, mode):
        if mode == 'high':
            return self.data[self.data['TEAM_ID'] == self.high_win_team]
        return self.data[self.data['TEAM_ID'] == self.high_loss_team]

    def cos_sim(self, a, b):
        # a_norm = np.linalg.norm(np.array(a))
        # b_norm = np.linalg.norm(np.array(b))
        # cos = np.dot(a,b)/(a_norm * b_norm)
        cos = np.dot(a,b)/(np.linalg.norm(a)*np.linalg.norm(b))
        return cos
```

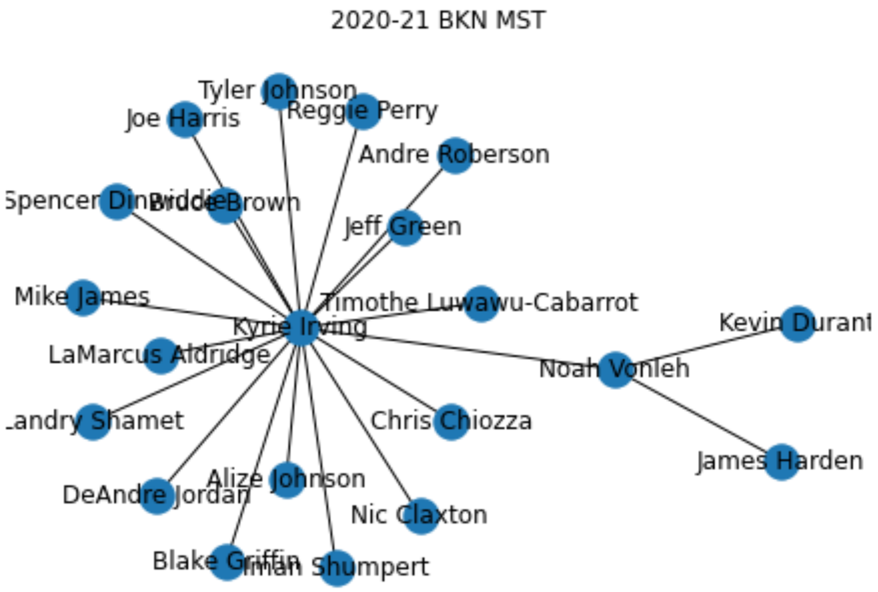
By instantiating a TEAM, users automatically get their favorite team, the best team, the worst team, and view detailed player performance through the corresponding class attributes. Note that the team's record here cannot be directly obtained through player data, such as Durant and Harden, who are also Nets players, because of the different number of appearances, the corresponding records of the two are not the same, which is contrary to the fact that a team has only one record. This experiment captures the team data and connects the two tables with the team ID to obtain the correct team record.

The class also defines a similarity matrix. Specifically, the performance data of players is mapped to the vector space as a dense vector by Euclidean distance, and the distance between any player is calculated using cosine similarity. Calling the corresponding method obtains a team's similarity matrix.

4 Data analysis

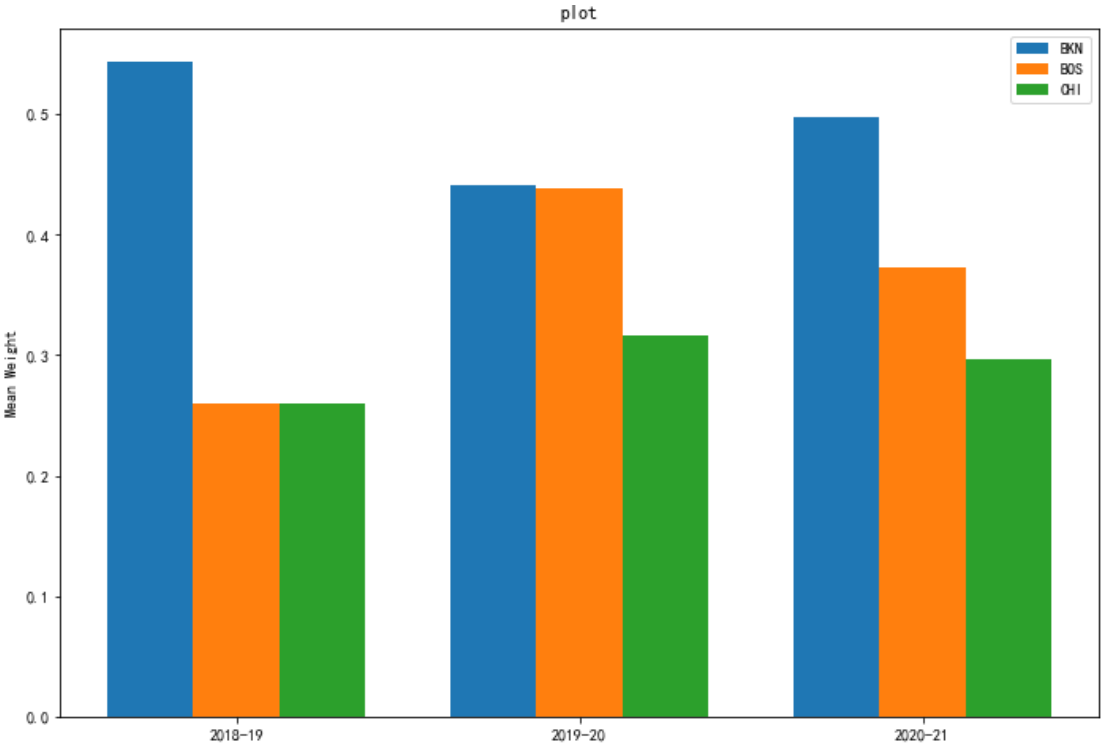
The kurstal algorithm is used to convert the adjacency matrix into a graph network, and then the minimum spanning tree is obtained. The meaning of the tree is that for any node, the node that is least similar to it is connected to an edge, that is, the connected graph becomes the non-connected graph. It reflects how different a team is, with teams with smaller average weights becoming less similar and stylistic differences. For this experiment, the Brooklyn Nets, Chicago Bulls, and Houston rockets were selected for minimal spanning tree analysis.

5 Data Visualization



Taking Brooklyn as an example, it can be seen that Kyrie Irving is at the center of the node, connecting most of the points, indicating that Irving is the least similar to other players, which is also in line with Irving's unique style of play.

The experiment then selected the Bulls, Celtics, and Nets to plot the average weights of the past three years, as shown in the following figure:



The Nets' weights are highest in three years, the Bulls are lowest, and the Nets perform best among the three teams, which means that the closer the team, the more teamwork is stronger and the easier it is to get good performances.