Name: David Su

Student ID: 13020805

Introduction

(1) Define Problem: Our problem is about the basketball. There are five positions in basketball which are Point Guard, Shooting Guard, Small Forward, Power Forward and Centre respectively. Point Guard is the people who make the team's offense frequently by controlling the ball and move the ball to the right place to the right player at the right time. A Point Guard is also the most understandable people to the coach's gameplan, they control the pace of the game by like another coach in the court. So a Point Guard is significant to a team who want to win the most game in the season, or more, the championship.

(2) Justify the Significance

In this report we are going to classify all the Point Guard into five categories which can be benefit to the player and many people in the basketball team. For player, this classification report can be seen as an reference for them to better know what types of game they are playing and can be compared with the imagination of themselves in their mind. Then, players can make decisions to keeping going on his way of playing or make a change to create a new style to play a better game. For the basketball team, the coach can get the result of if his Point Guard adapt his gameplan or he need a new Point Guard. The scout may be the people who like this report most because it will be important for their job which is finding the appropriate player the boss and coach need.

Exploration

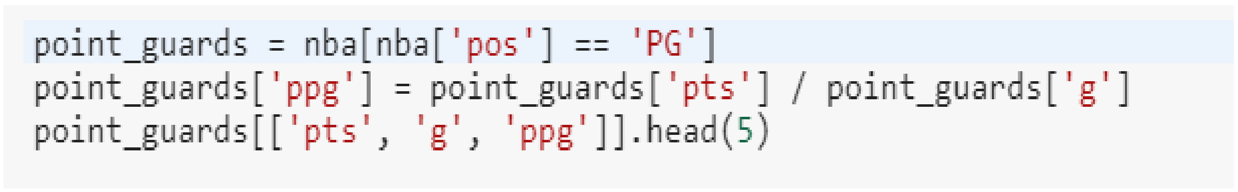
(1) Identify Challenges

The algorithm we are going to use is one of the most cluster algorithm called K-means. K-means clustering is a kind of algorithm that try to partition n data samples into k clusters. The way it partition the data samples is make each data sample belong to the cluster with the nearest mean, which is the centroid of each cluster. According to the definition of the K-means algorithm we know that the most difficult and important things is what the value is the k which will make big impression to the result. Specific it in our case is how many kind of player types we are going to classify and what is the standard of the classification.

(2) Design Data Structure

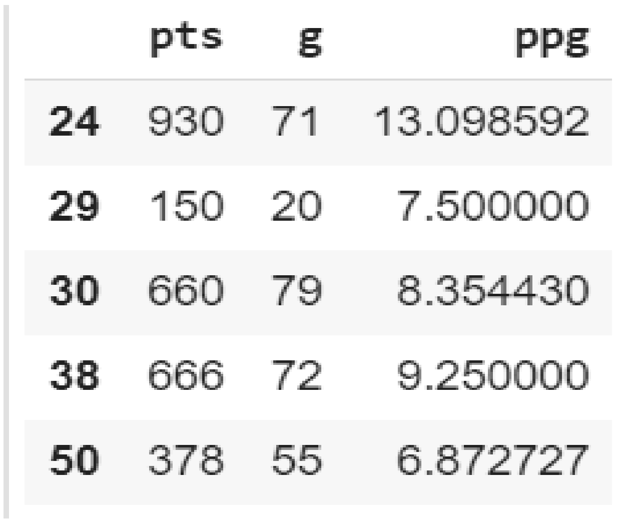
The two main data we need are Point Per Game and Assist Turnover Ratio respectively which are we are talking about in this section.

First we extract 'PG' from all the five positions and define point_guard as 'PG'. We have no Point Per Game in our dataset but only sum points and game counts. So we define point_guard['ppg'] as point_guards['pts'] / point_guards['g']. Then we run the code and show if the result is right.



```
point_guards = nba[nba['pos'] == 'PG']
point_guards['ppg'] = point_guards['pts'] / point_guards['g']
point_guards[['pts', 'g', 'ppg']].head(5)
```

Diagram 1. Define the variable



```
     pts   g         ppg
24   930  71   13.098592
29   150  20    7.500000
30   660  79    8.354430
38   666  72    9.250000
50   378  55    6.872727
```

Diagram 2. Test the data

Second we go to ATR. There is a data statistic in NBA named Assist Turnover Ratio(ATR), it is a main standard to judge if a Point Guard is competent. Assist can reflect how good a Point Guard are in controlling the ball and pace, moving the ball to the right people in the right time, creating space for teammates to get scores. Turnover however can reflect how good a Point Guard is at the things we talk about above in another aspect. Because usually a Point Guard has the most time dominant the ball in the court, in other words, they have the biggest potential to make a turnover. So that is why the ATR can reflect if a Point Guard is competent, the more assists you get and the less turnovers you make, the more competent you are.

The formula of it is just like its' name. ATR= Assist/Turnover. Before we implement the formula into real data we need to clean the data which its' turnover is zero, because the player who has no turnover usually means they have no time to get in the court. So it is meaningless to analyse those data which the turnover is zero. The code we implement shows below.

```
point_guards = point_guards[point_guards['tov'] != 0]
point_guards['atr'] = point_guards['ast'] / point_guards['tov']
```

Diagram 3. Calculate the ATR

After that we finally can show what we want to show but can not in the beginning due to the lack of data. The x label represent ppg and the y label represent ATR.

```
plt.scatter(point_guards['ppg'], point_guards['atr'], c='y')
plt.title("Point Guards")
plt.xlabel('Points Per Game', fontsize=15)
plt.ylabel('Assist Turnover Ratio', fontsize=15)
```

```
Text(0, 0.5, 'Assist Turnover Ratio')
```

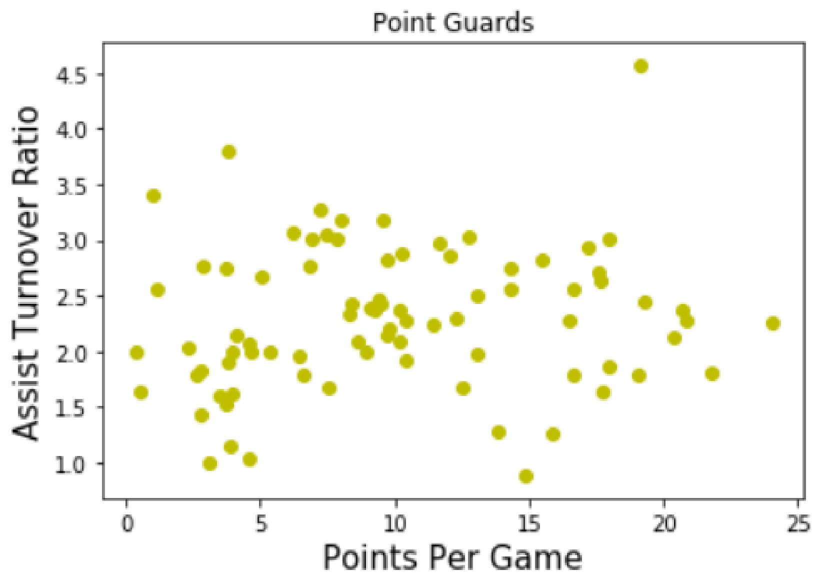


Diagram 4. Visualize the all the points

Methodology

(1)Implement Algorithms

As what algorithm we use can be called directly from the sklearn package so we try to show how K-means algorithm works in details in our own way, following is how we implement K-means algorithm using our code:

```
#Step 1:  Randomly generate five points

num_clusters = 5

random_initial_points = np.random.choice(point_guards.index, size=num_clusters)

centroids = point_guards.ix[random_initial_points]

# Visualize the initial cluster center, with the center in red and the other points in yellowplt.scatter(point_guards['ppg'], point_guards['atr'], c='yellow')

plt.scatter(centroids['ppg'], centroids['atr'], c='red')

plt.title("Centroids")

plt.xlabel('Points Per Game', fontsize=13)

plt.ylabel('Assist Turnover Ratio', fontsize=13)
```

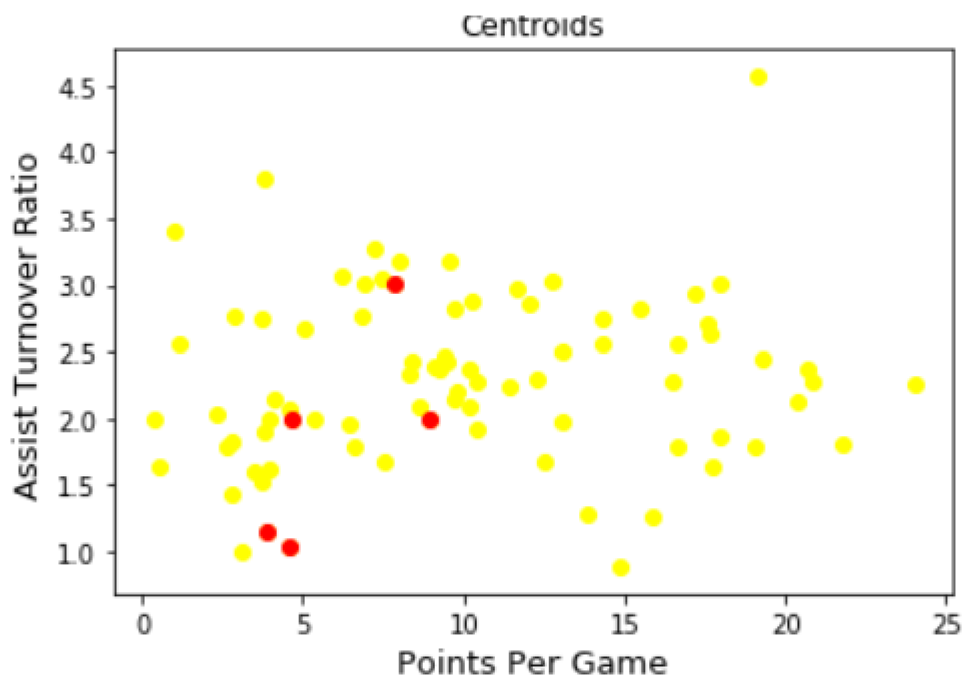


Diagram 5. Selected cetroids

```
# The center point is then converted into a dictionary format, where the key of the dictionary
is the name of the cluster, and the value of the dictionary is the information of the center
point( "ppg ", "atr " )

def centroids_to_dict(centroids):
    dictionary = dict()
    # Counter is incrementing as the dictionary key
    counter = 0
    # iterate a pandas data frame row-wise using .iterrows()
    for index, row in centroids.iterrows():
        coordinates = [row['ppg'], row['atr']] #list objective
        dictionary[counter] = coordinates
        counter += 1


    return dictionary

centroids_dict = centroids_to_dict(centroids)

# Then calculate the distance of each point to the cluster center and change the cluster
center of each point to the nearest cluster

import math

# A function of calculating the distance between two points

def calculate_distance(centroid, player_values): # The arguments are all list objects

    root_distance = 0

    for x in range(0, len(centroid)):

        difference = centroid[x] - player_values[x]

        squared_difference = difference**2

        root_distance += squared_difference


    euclid_distance = math.sqrt(root_distance)

    return euclid_distance

# Returns the key of the closest cluster to each point

def assign_to_cluster(row):
    lowest_distance = -1

    closest_cluster = -1

    for cluster_id, centroid in centroids_dict.items():

        df_row = [row['ppg'], row['atr']]

        euclidean_distance = calculate_distance(centroid, df_row)



        if lowest_distance == -1:

            lowest_distance = euclidean_distance

            closest_cluster = cluster_id

        elif euclidean_distance < lowest_distance:

            lowest_distance = euclidean_distance

            closest_cluster = cluster_id

    return closest_cluster

# Create a new attribute, cluster: stores the key values of the cluster to which each node belongs

point_guards['cluster'] = point_guards.apply(lambda row: assign_to_cluster(row), axis=1)

# The other points are assigned to these clusters according to the randomly initialized clusters and their key values. Visualize the first randomly initialized cluster diagram, with different clusters represented in different colors
```

```
def visualize_clusters(df, num_clusters):

    colors = ['b', 'g', 'r', 'c', 'm', 'y', 'k']



    for n in range(num_clusters):

        clustered_df = df[df['cluster'] == n]

        plt.scatter(clustered_df['ppg'], clustered_df['atr'], c=colors[n-1])

        plt.xlabel('Points Per Game', fontsize=13)

        plt.ylabel('Assist Turnover Ratio', fontsize=13)

visualize_clusters(point_guards, 5)
```

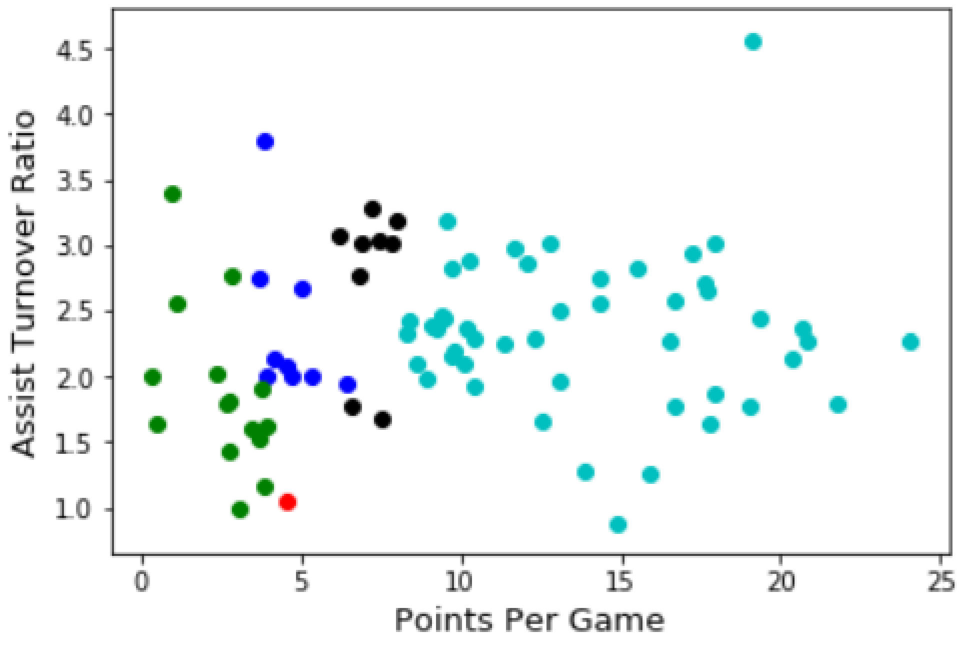


Diagram 6. First time iteration

#Step 2 : To allocate all the points to the nearest cluster, it is necessary to adjust the center of the cluster and recalculate the center point of the cluster

```
# Computing cluster center

def recalculate_centroids(df):

    new_centroids_dict = dict()



    for cluster_id in range(0, num_clusters):

        values_in_cluster = df[df['cluster'] == cluster_id]

        # Calculate new centroid using mean of values in the cluster

        new_centroid = [np.average(values_in_cluster['ppg']),
np.average(values_in_cluster['atr'])]

        new_centroids_dict[cluster_id] = new_centroid

    return new_centroids_dict



centroids_dict = recalculate_centroids(point_guards)
```

After we implement our algorithm we are going to talk something practical when we implmenting the algorithm regarding to our real project. The five cluster is selected randomly by the system in the beginning, but after several times iteration we finally get five typical point which are represent the five typical class of Point Guard in NBA. It means when we get some stable points as the centroid to represent every cluster, we can think every point in the same cluster are playing the same level or type of game in NBA.

Evaluation

(1) Report Execution on Data

We implement the K-means algorithm using our code and the visualize the execution on data. The first diagram represent the first point selection we choose randomly and in most situation it is no special meaning. As we know from the K-means algorithm we need many times of iteration to redefine the cluster centriods and recalculate the distance to get some stable clusters. And the we show our points and distance is visualiza them in a two label diagram which x label represent Point Per Game and y label represent Assist Turnover Ratio. So what we do is execute the code which implement K-means algorithm and visualize the cetroids and the cluster we get. Next we only need to do enough times of iteration and observe the diagram to judge if we have got all the clusters stable. In our report the iteration times is eight and we can clearly see that the points in every cluster do not change anymore from the seventh time iteration to the eighth time iteration.

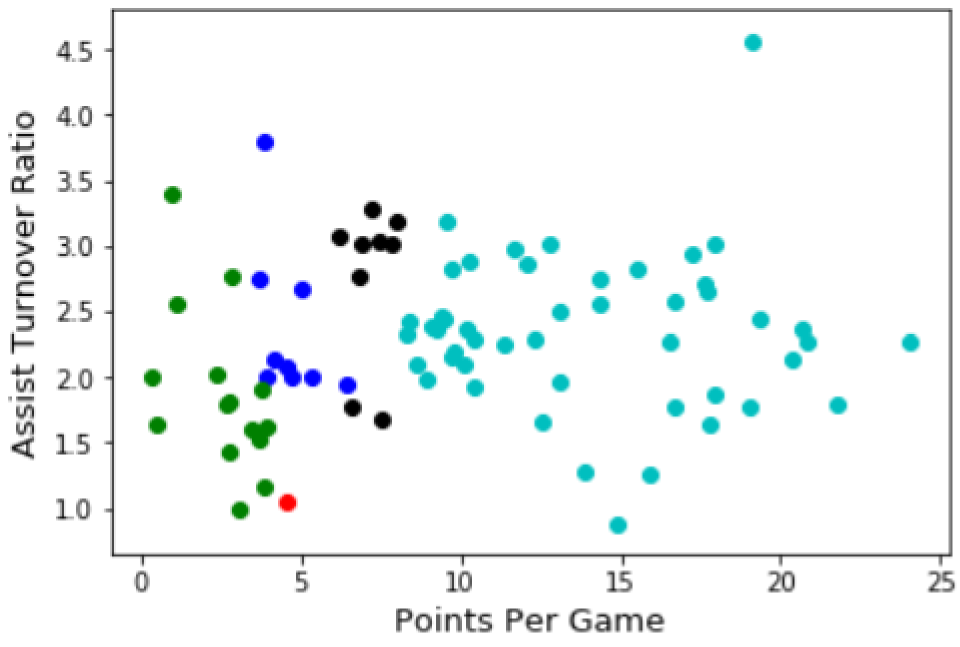


Diagram7. First time iteration

#Repeat Step 1(Second times)

point_guards['cluster'] = point_guards.apply(lambda row: assign_to_cluster(row), axis=1)

visualize_clusters(point_guards, num_clusters)

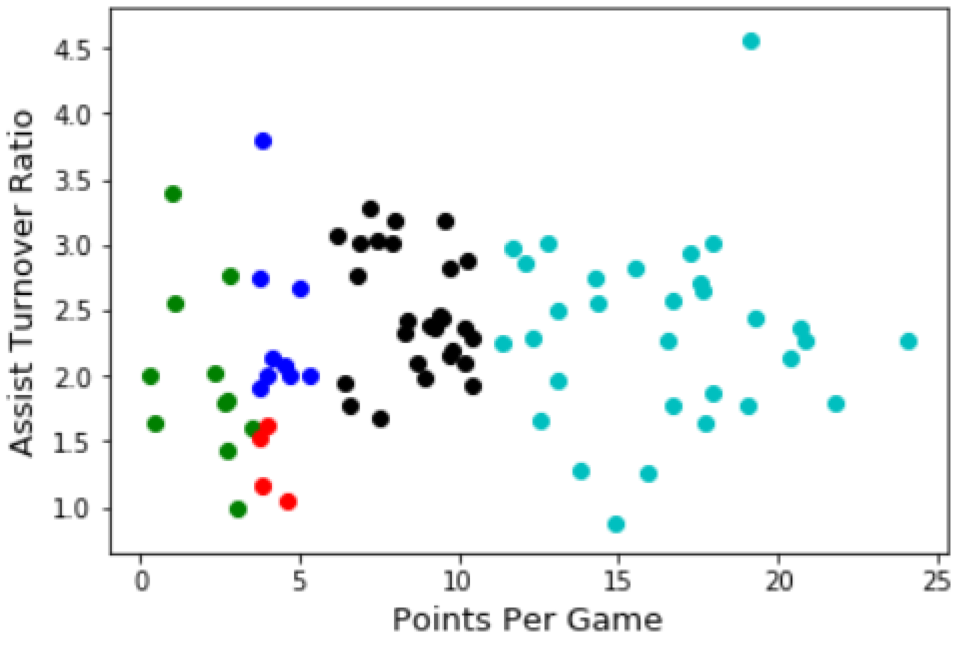


Diagram 8. Second time iteration

#Repeat Step 2 and Step 1(Third Times)

centroids_dict = recalculate_centroids(point_guards)

point_guards['cluster'] = point_guards.apply(lambda row: assign_to_cluster(row), axis=1)

visualize_clusters(point_guards, num_clusters)

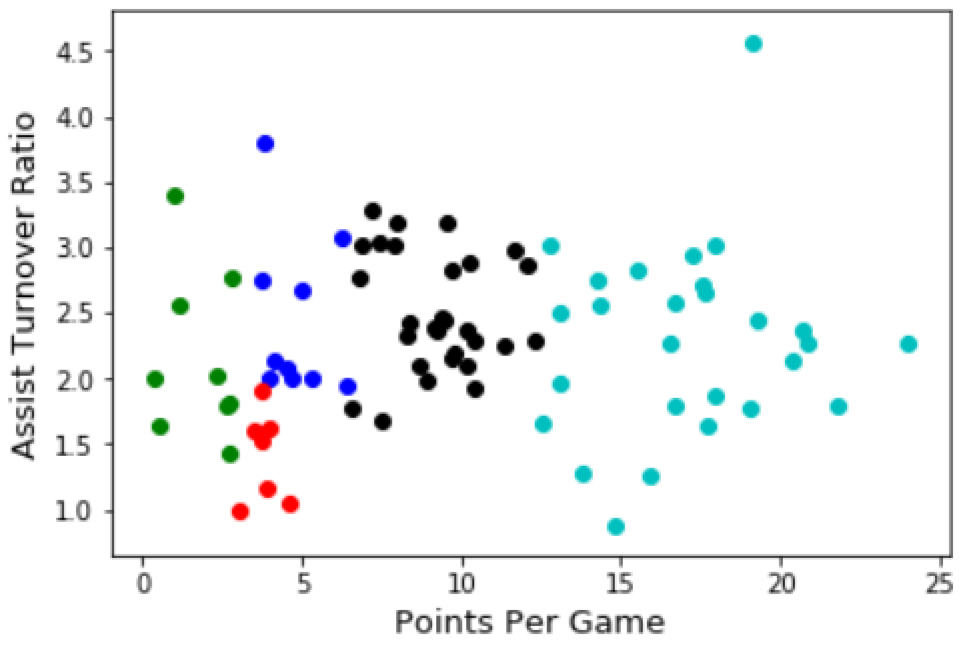


Diagram 9. Third time iteration

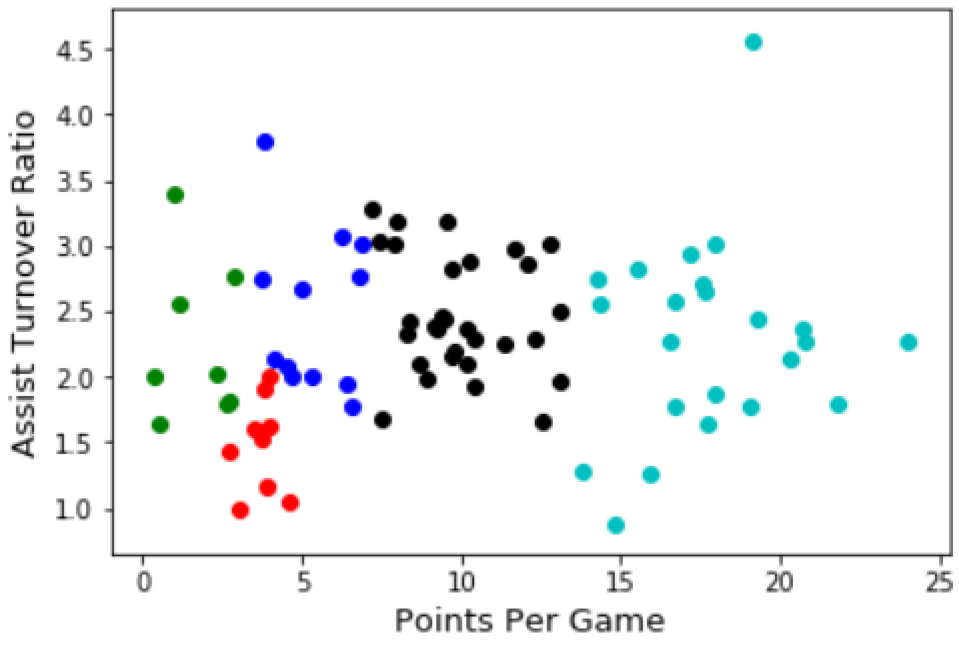


Diagram 10. Fourth time iteration

After more iterations, we finally get a stable cluster point at the eighth time iteration.

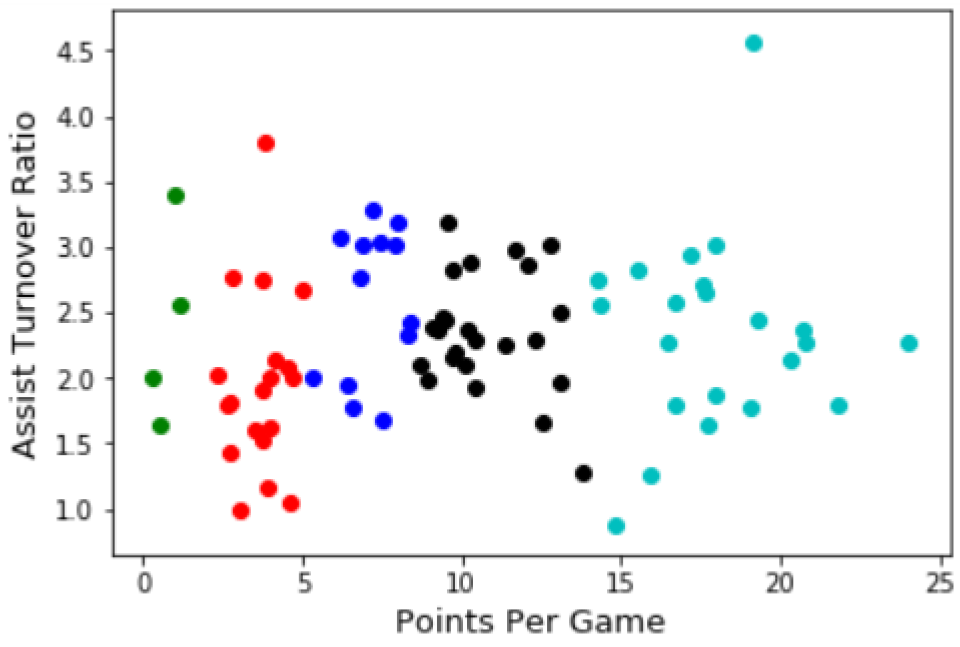


Diagram 11. Seventh time iteration

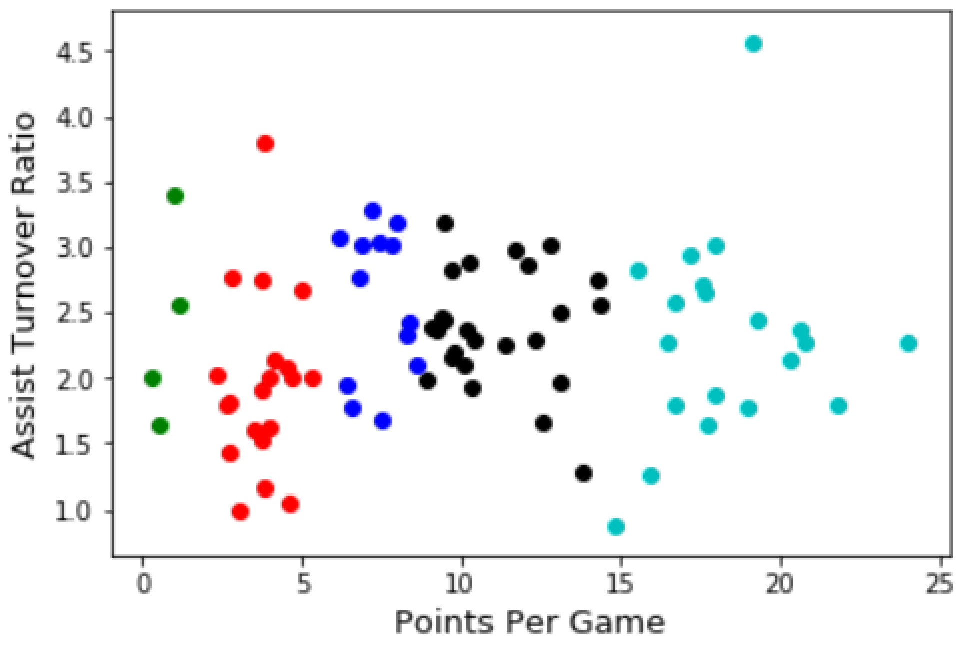


Diagram 12. Eighth time iteration

(2) Perform Efficiency Analysis

From the diagram we can see that the main feature that classify our clusters is Point Per Game. So we can see that in general we can classify the Point Guard in NBA into five categories which means the more points they get they bette they are. If we want to segment more detailedly we can see that for every class of Point Guard the more Assist Turnover Ratio they get they better they are. That comes to a conclusion which is if you see a point guard as a point in the diagram, the nearer they close to the top right the better they are.

Another point we get is if a Point Guard is close to the top right, all the Point Guard the same class with this player is deserved to pay attention to.

Conclusion

(1) In this report we classify Poing Guard into five categories by comparing their Point Per Game and Assist Turnover Ratio. It is no doubt that using more player information to classify Point Guard is also feasible. When you get stable clusters and then you try to get the player's information from this cluster you will find that most of the players in the same cluster are playing the same level of game. So if you find a player that is outsatnding, you will feel happy to pay more attention to the player in the same cluster.

(2) Propose possible improvement

In the process of implementing K-means algorithm we find that it is not very big for the change of the node it is mainly because:

a. In the iterative process, k-means algorithm will not cause great changes for each cluster, so this algorithm is always convergent and stable.

b. Due to the conservative iteration of k-means algorithm, the final result is closely related to the selected initial point

In order to solve these problems, the k-means implementation in the sklearn package makes some intelligent functions, such as repeated clustering and random selection of the center of mass each time, which is much less deviation than using only one selection of the center of mass. So next time we can just call the K-means algorithm from the sklearn package.

Ethical

My report in based on a most widely used cluster algorithm named K-means which will not infringe anyone's benefit because this algorithm has been existing for many years. And the data of our project is from the official website of NBA which are open to anyone who want so it is sure that will not make any player uncomfortable and do harm to them.

Our report is aim to analyze and classify the Point Guard playing in NBA in 2013-2014 season using Point Per Game and Assist Turnover Ratio which is a method that the professional NBA analyst using to do their job, so I can ensure the quality and the integrity of my research. All the research in done independtly by myself and I always hold an impartial attitude to both my research and the objective in my report. The algorithm using in my report is only one of the method that can be used to analuze the player, but it can not be guaranteed to be the best algorithm for this problem, I will be happy with other who can propose a new method to solve this problem.