SI 206 Final Project Report

Group Name: *The 707 Backlot Boyz*
Group Members: Adam Brenner & Cooper Drobnich

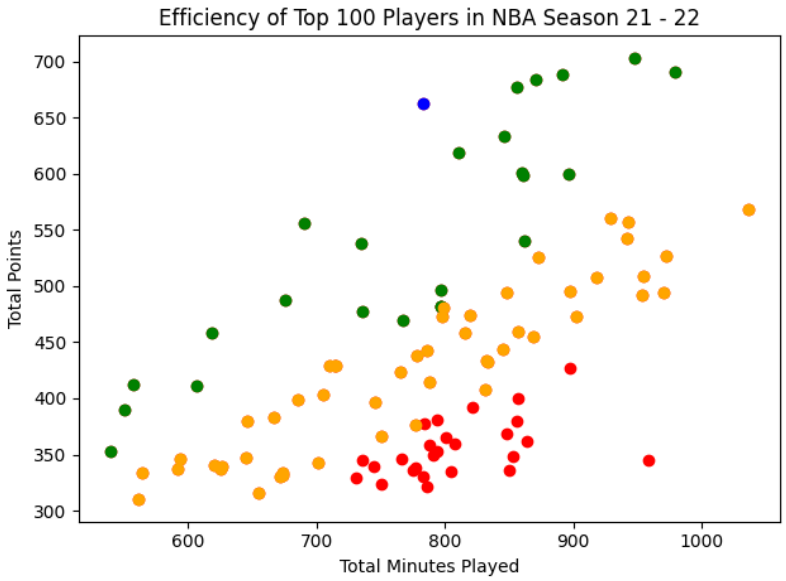
Goals:
Our first goal was to compare the stats between Lebron James and Michael Jordan. However, after beginning the project we would not have enough data to insert into the database for each player. Once this was assessed we decided to go down a different path. Instead, we decided to compare the top 100 players in the NBA based on points. From the 100 players our goal was to find our own way to measure the overall efficiency, turnover ratio and division counts for each of the players. Overall we wanted to find the player with the highest points per minute, the player with the highest turnovers per minute, and the division with the most top 100 players. Another goal was to be able to find a correlation between the data we selected and be able to visualize it so anyone can understand.

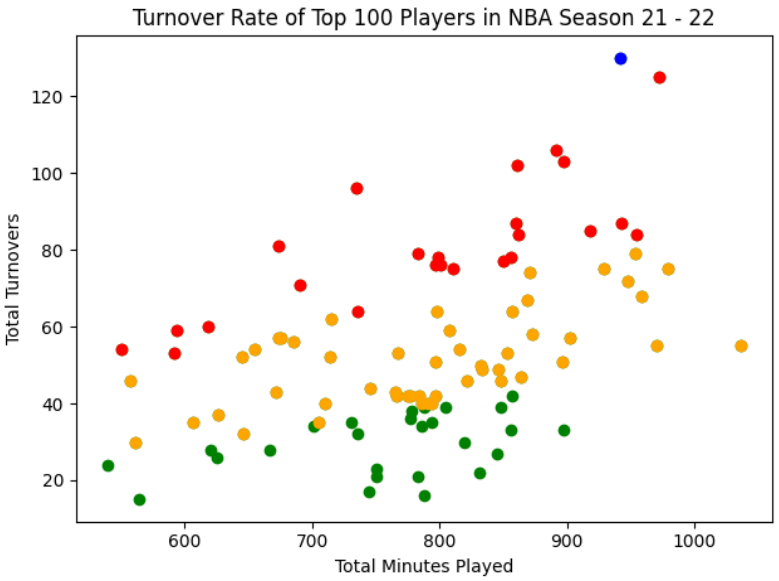
Goals Achieved:
Overall we achieved all of our goals for the project. We were able to calculate the average points per minute, turnovers per minute, and compare how many players were in each division for the top 100 players in the NBA. We were also able to go above what our goals were and visualize the data in a meaningful way. We were also able to represent the top 25, middle 25 and bottom 25 players on our graph and plots.

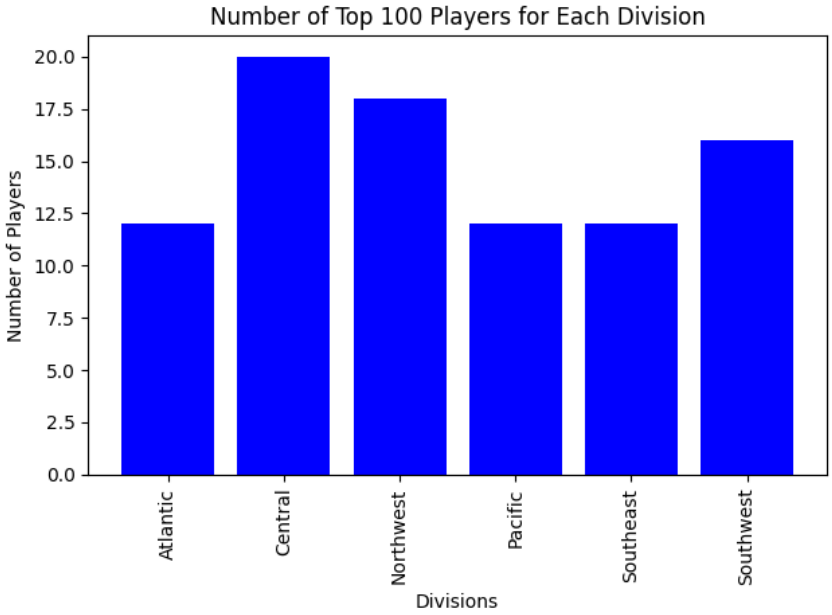
Problems:
Throughout the project we experienced many problems. One problem was we had difficulty with sorting our data dictionaries with the Basketball Reference website. Additionally, we had to deal with an issue arising from the NBA data to ensure it is properly live updating to add or remove players from our top 100 list as the season progresses in real time. We also ran into a problem where the API we were using had a certain amount of data per page. With the API we also ran into a problem where there was a limited amount of requests per minute. Sometimes we would run the code and it would show up with an error that was not from our code but because of the API request limit.

Calculation File:
The calculations that we made are in the file 'playerCalculations.txt' that can be found on our github repository here: https://github.com/Cooperdrobnich/FinalProject/blob/main/playerCalculations.txt

Visualizations:







Instructions:

Clear or delete the database and run the FinalProject.py four times. Next, run the Visualizations-Calculations.py to see the three visualizations and top players / division calculations. After the two files have been run correctly the database and calculation text file will have the data for each player in the NBA top 100 players.

Documentation:

FinalProject.py

Function: get_player_names
Input: None
Description: This function uses beautiful soup to parse through the urls table data. This function then returns a list of all the players in the current 21'-22' NBA season.
Output: ['Precious Achiuwa', 'Steven Adams', 'Bam Adebayo', 'Santi Aldama', 'LaMarcus Aldridge', 'Nickeil Alexander-Walker', …]

Function: get_team
Input: None
Description: This function uses beautiful soup to parse through the urls table data. This function then returns a list of all the team abbreviations correlated to the players from the function above in the current 21'-22' NBA season.
Output: ['TOR', 'MEM', 'MIA', 'MEM', 'BRK', 'NOP', 'MIL', 'CLE', 'NOP', …]

Function: get_minutes_played
Input: None
Description: This function uses beautiful soup to parse through the urls table data. This function then returns a list of the total minutes played correlated to the players from the first function in the current 21'-22' NBA season.
Output: [556, 673, 592, 148, 590, 825, 774, …]

Function: get_points
Input: None
Description: This function uses beautiful soup to parse through the urls table data. This function then returns a list of the total number of points correlated to the players from the first function in the current 21'-22' NBA season.
Output: [168, 188, 337, 55, 349, 396, 351, …]

Function: get_turnovers
Input: None
Description: This function uses beautiful soup to parse through the urls table data. This function then returns a list of the total number of turnovers correlated to the players from the first function in the current 21'-22' NBA season.
Output: [23, 47, 53, 4, 19, 43, 18, …]

Function: create_data_dict
Input: list of players, list of team abbreviations, list of minutes_played, list of total points, and list total turnovers
Description: This function takes in a list of players, team abbreviations, minutes_played, total points, and total turnovers and adds them to a dictionary called data_dict with player being the key and a dictionary as the value. Within the value dictionary it has the players specific stat titles as the keys and the stat as the value. The function then sorts the list from highest to lowest total points scored by players. Finally, it returns a dictionary of the top 100 NBA players stats by points.
Output: [('Kevin Durant', {'team': 'BRK', 'minutes_played': 912, 'points': 735, 'turnovers': 79}), ('Zach LaVine', {'team': 'CHI', 'minutes_played': 948, 'points': 703, 'turnovers': 72}), ('Jayson Tatum', {'team': 'BOS', 'minutes_played': 979, 'points': 691, 'turnovers': 75}), ('Trae Young', {'team': 'ATL', 'minutes_played': 891, 'points': 688, 'turnovers': 106}), …]

Function: get_id_team
Input: None
Description: This function uses the 'balldontlie' API to create a dictionary with the team abbreviation as the key and the correlating division for that specific team as the value. This function returns a dictionary in this format… Ex. {'ATL': 'Southeast', 'BOS': 'Atlantic', 'BKN': 'Atlantic', 'CHA': 'Southeast', ... }
Output: {'ATL': 'Southeast', 'BOS': 'Atlantic', 'BKN': 'Atlantic', 'CHA': 'Southeast', 'CHI': 'Central', 'CLE': 'Central', 'DAL': 'Southwest', 'DEN': 'Northwest', 'DET': 'Central', 'GSW': 'Pacific', 'HOU': 'Southwest', 'IND': 'Central', 'LAC': 'Pacific', …}

Function: create_database
Input: database file name
Description: This function creates a database called 'Top100nbaStats.db'
Output: Connection to database

Function: create_table
Input: The connection to the database (cur, conn)
Description: This function creates two tables within 'Top100nbaStats.db', 'PlayerStats' and 'TeamDivision'. The PlayerStats table has column headers name, team, minutes_played, points, and turnovers. The TeamDivision table has column headers team and division.
Output: None

Function: insert_player_data
Input: Dictionary from create_data_dict output, The connection to the database (cur, conn)
Description: This function takes in a dictionary from the create_data_dict() function and inserts the stats for 25 unique players into the PlayerStats table.
Output: None

Function: insert_team_data
Input: Dictionary from get_id_team output, The connection to the database (cur, conn)
Description: This function takes in a dictionary from the get_id_team() function and inserts the division for 25 unique teams into the TeamDivision table.
Output: None

Visualizations-Calculations.py

Function: get_database
Input: db file name
Description: connects to the 'Top100nbaStats.db' database
Output: Connection to database

Function: points_minutes_viz
Input: The connection to the database (cur , conn)
Description: This function selects points and minutes played from PlayerStats and creates a scatter plot with minutes played on the x-axis and total points on the y-axis. It plots the most efficient player (the player with the highest points per minute) in blue, the top 25 players in green, the middle 50 players in orange, and the bottom 25 players in red.
Output: Efficiency of Top 100 Players viz.png

Function: turnovers_minutes_viz
Input: The connection to the database (cur, conn)
Description: This function selects turnovers and minutes played from PlayerStats and creates a scatter plot with minutes played on the x-axis and total turnovers on the y-axis. It plots the player with the largest turnover ratio in blue, the top 25 turnover ratio players in green, the middle 50 turnover ratio players in orange, and the bottom 25 turnover ratio players in red.
Output: Turnover Rate of Top 100 Players viz.png

Function: get_most_efficient
Input: The connection to the database (cur , conn)
Description: This function selects name, points, and minutes played from PlayerStats and gets the player with the highest points per minute from the database table, PlayerStats by sorting all of the players into a list. This function returns the sorted efficiency list with the format [(player1, points, minutes), (player2, points, minutes), ...]. This function also prints the player with the highest efficiency.
Output: [('Giannis Antetokounmpo', 683, 815), ('Kevin Durant', 735, 912), ('Nikola Jokić', 556, 691), ('Stephen Curry', 677, 856), …], prints: The most efficient player in the NBA Top 100 is Giannis Antetokounmpo scoring 0.838 points per minute.

Function: get_highest_turnover_rate
Input: The connection to the database (cur, conn)
Description: This function selects turnovers and minutes played from PlayerStats and creates a scatter plot with minutes played on the x-axis and total turnovers on the y-axis. It plots the player with the largest turnover ratio in blue, the top 25 turnover ratio players in green, the middle 50 turnover ratio players in orange, and the bottom 25 turnover ratio players in red, in blue, the top 25 turnover ratio players in red, the middle 50 turnover ratio players in orange, and the bottom 25 turnover ratio players in green.
Output: [('James Harden', 130, 942), ('Luka Dončić', 96, 735), ('Russell Westbrook', 130, 1007), ('Cade Cunningham', 81, 674), …], prints: The player with the highest turnover rate in the NBA Top 100 is James Harden turning the ball over 0.138 times per minute.

Function: get_division_dict
Input: The connection to the database (cur, conn)
Description: This function joins PlayerStats and TeamDivision on PlayerStats.team = TeamDivision.team. It gets the count of players in the top 100 for each division and returns a dictionary with the division as the key and the number of players as the value.
Output: {'Atlantic': 12, 'Central': 19, 'Northwest': 18, 'Pacific': 12, 'Southeast': 12, 'Southwest': 16}

Function: division_viz
Input: Dictionary from get_division_dict
Description: This function creates a bar plot with division on the x-axis and number of players on the y-axis. It returns a dictionary that is sorted from largest to the smallest number of players for the divisions. It also prints the division with the most players in the NBA top 100.
Output: [('Central', 19), ('Northwest', 18), ('Southwest', 16), ('Atlantic', 12), ('Pacific', 12), ('Southeast', 12)], prints: The division with the most top 100 players in the NBA is the Central division with 19 players.

Function: write_calculations
Input: Sorted dictionary by efficiency, sorted dictionary by turnover ratio, sorted dictionary of division within top 100 players.
Description: This function writes all of the calculations for each player/division in a text file called 'playerCalculations.txt'. The text file has the ranks for the players with the top points per minute and turnovers per minute, as well as, the number of players in the top 100 for each division.
Output: None, text file 'playerCalculations.txt

Resources:

Date	Issue Description	Location of Resource	Result
12/7/21	Limiting the number of data inserted into database to 25 or fewer	https://stackoverflow.com/questions/11768127/use-limit-in-a-mysql-insert	Not Solved
12/7/21	Sorting dictionary based on key value	https://stackoverflow.com/questions/38218501/python-get-top-n-keys-with-value-as-dictionary	Problem Solved
12/8/21	Sort nested dictionary by keys	https://www.geeksforgeeks.org/python-sort-nested-dictionary-by-key/	Problem Solved
12/10/21	Sorting two lists based on the ratio between the two	https://stackoverflow.com/questions/45196229/sort-2-lists-in-python-based-on-the-ratio-of-individual-corresponding-elements-o	Problem Solved
12/10/21	Limiting float decimal places	https://www.kite.com/python/answers/how-to-limit-a-float-to-two-decimal-places-in-python	Problem Solved
12/11/21	Writing text into text file	https://www.geeksforgeeks.org/reading-writing-text-files-python/	Problem Solved
12/11/21	Creating one scatter plot with multiple colors	https://moonbooks.org/Articles/How-to-create-a-scatter-plot-with-several-colors-in-matplotlib-/	Problem Solved
12/11/21	Creating scatter plot	https://matplotlib.org/stable/plot_types/basic/scatter_plot.html#sphx-glr-plot-types-basic-scatter-plot-py	Problem Solved