Data Engineering Write Up

NBA Statistics Dashboard

Derek Call

Abstract

My goal in this project was to create an interactive dashboard where users could choose NBA players and compare them across a wide range of statistics. Where not all players are contemporary with one another I have designed the comparison to be from start of career to start of career. Users can compare basic box score statistics or different shooting statistics. Using Streamlit I was able to incorporate user input into the behind-the-scenes SQL queries and plot creations and then display them dynamically on the web page. A continuation of this project would include different windows of time for comparison, such as specific seasons or even individual game statistics.

Design

Streamlit was used to take user input and then using the inputs, display the correctly queried information in a line graph that compared the two chosen players and desired statistical output. The site has been configured in such a way that I can continue to add statistical categories or new statistics without changing the overall layout.

Data

I gathered the data using APIs provided by RapidAPI (<https://rapidapi.com/theapiguy/api/free-nba/>). Using these APIs I was able to directly access NBA player, team and game statistics for all players and games dating back to 1948 More than 20 individual statistics were gathered for each player, with there being over 1 million entries in the data set.

Algorithms

I used APIs to directly access NBA information and download into a Pandas Dataframe in Python. I then converted to a csv and from there read in the data into a SQL table. All further access of data is done using SQLAlchemy.

Models

This is a display/ of EDA and statistical graphs. There are no specific underlying models.

Findings

Streamlit was a great tool for this project, but it did have its limitations. On the input for player names, it requires an exact spelling as fuzzy matching or autocomplete are not Streamlit functionalities. As I continue with this project, I would like to incorporate fuzzy matching in the Python code to make the GUI more user friendly. I would also like to deploy this locally hosted application as a Heroku app once I have added more functionality.

Tools

* Python and API packages for data acquisition
* Sqlite3 and DB Browser for data storage and queries
* Pandas and SQLAlchemy Python packages for data cleaning and EDA
* Matplotlib for data visualization
* Streamlit for web app creation