NBA Game Prediction

Personal

Employ historical data to create and train a linear regression model that loosely predicts the winner of a random pair of home and away teams.

1. Introduction

   1.1 Background -

   As a lifelong fan of NBA basketball, I have always been fascinated with the game. Immersing myself and dedicating time to research and understanding not only the entertainment, but the make up of the game. This led me to dive into the world of data. With studying on my own the past two years, it was time to begin putting my newfound understanding to work. And what better way to do so than with NBA basketball?

   1.2 Objectives –

   The main goal of this project is to display my coding skills by way of developing a prediction model. Secondary objectives include learning about machine learning concepts and why they are important, practicing my data visualization skills and expanding my use of the expansive python libraries available.

2. How to go about it?

   2.1 Research –

   Understanding that correct and verified data is essential for precise machine learning techniques. I focused on the Basketball Reference website, a branch of Sports Reference LLC, a leading information and historical data source for countless sports.

   2.2 Key Points of Emphasis –

   While working on this project, I was able to name several points that were to be of focus when not only coding but also performing calculations. While Basketball Reference supplies an already well-structured database on their site, there were multiple points of vulnerability that easily could've altered the process. Team names were a key player in this way due to certain team names being similar. Overtime games posed another obstacle to maneuver.

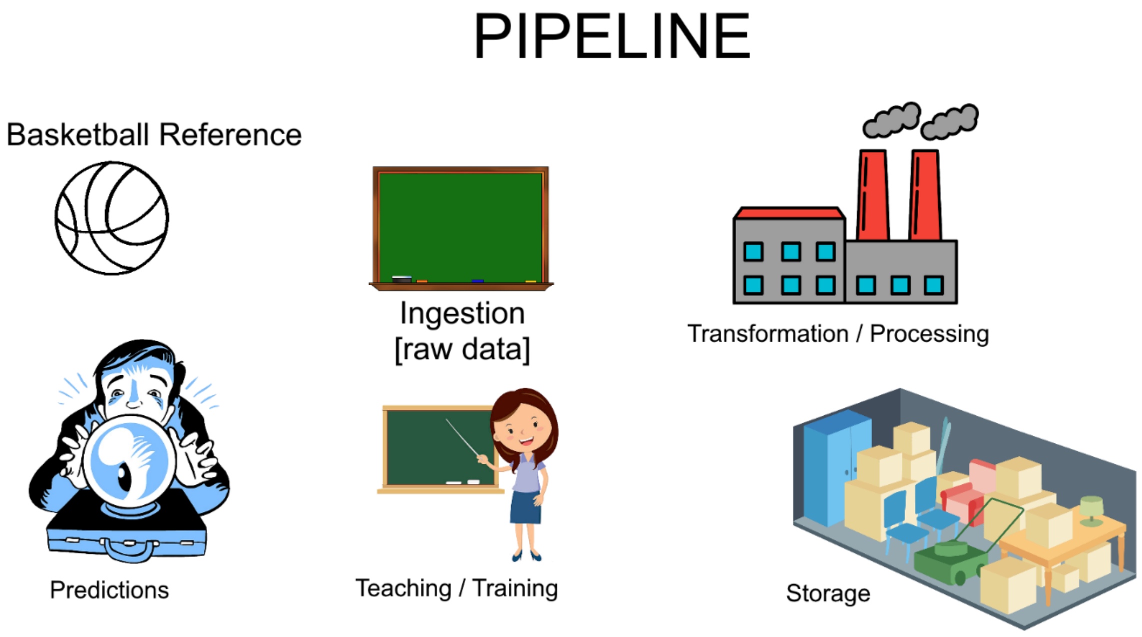
3. Toolbox -

   I handpicked each of the tools for specific reasons. Jupyter Notebook is the IDE used on Kaggle.com therefore using it on the site streamlined being able to make changed offline. For each of the python libraries used, each played a key role. BeautifulSoup allowed for web scraping to collect our raw dataset. Pandas, the point guard and floor general, has so many use cases; too many to count. Mainly we stabilized and transformed out into our new dataset. NumPy was a role player, who came in at clutch time to knock down free throws to put the game on ice with the efforts to create ranges for exploratory data. Our enforcer, statsmodels, then began rearranging the opposition updating to category codes and training models. Per usual, matplotlib spearheaded the color commentating visuals to present a better representation of what was happening.

A list of the tools and frameworks implemented in this project include:

1) Jupyter Notebook
2) Python
    a. BeautifulSoup
    b. statsmodels
    c. matplotlib / seaborn
    d. pandas
    e. NumPy

4. Flow –
   An Illustration of the data flow developed:



5. Where did it get you? –
   As stated above two of my biggest issues came when creating a dictionary used to create team / arena maps, along with some games going to overtime which threw off the web scraping results. One came when we saw that 2 teams share one arena, the Lakers and the Clippers. This created a dilemma when assigning the arena name. We had 30 teams available for selection but 29 arenas. We needed to explicitly edit the 2 teams'

arena name to specify who was at home to get an accurate read on the home team as well as correct the discrepancy of 29 arenas. As for the overtime games, we simply dropped the column of whether the game finished in regulation or not to uniform the data.

6. Results –
   We ended with a prediction of a randomly selected home team and randomly selected away team. Their final scores were calculated by their average points scored per game in the collect dataset. In our function, we included an if statement that reselects a team if both teams selected are the same one. In the case of 2 teams that scored the same average, meaning the game is predicted to go into overtime, a winner was randomly selected.

   A snippet of the prediction function is listed below:

```
def predictions():
    visitor_teams_random = random.sample(range(30), 15)
    home_teams_random = random.sample(range(30), 15)
    home_teams, visitor_teams, arena = [], [], []
    for i in range(15):
        while visitor_teams_random[i] == home_teams_random[i] and True:
            visitor_teams_random[i] = random.randint(0, 29)
            home_teams_random[i] = random.randint(0, 29)
```