Capstone Project 1: Data Wrangling

BACKGROUND

For my capstone project 1, **pitch prediction**, I will apply machine learning to predict the next pitch thrown by a Major League Baseball (MLB) starting pitcher. Barring an injury, suspension, or other circumstance, a starting pitcher will generally pitch between 20-30 games a season and 5-7 innings per game. Overall, it is expected that a starting pitcher will pitch 2,000-3,000 pitches during the regular season. Also, each pitcher has their repertoire of 3-5 pitch types they like to throw with the four-seam fastball being the most utilized. While it may reason that with only 3-5 different pitch types to expect, a hitter might easily predict what the pitcher is going to throw next. However, as the average number of strikeouts increases and the average number of hits continues to decrease each year, this is obviously not the case.

DATA COLLECTION

In 2006 postseason (playoffs), MLB first began using a camera-based system to track the trajectory, speed, spin, break, and location of a pitched ball, called PITCHf/x. Such information about each pitch also allows for the pitch type to be determined. Early models for classifying pitch types, however, were not very accurate for pitches similar in speed and break and the early years data includes many misclassified pitch types. Since 2015, a more advanced system (Statcast) is being used which integrates doppler radar with high definition video to track all aspects of a game, including pitches, hits, and players. For the 2017 season, Trackman, a component of Statcast, officially replaced the previous PITCHf/x system.

Initially, the plan was to use the publicly available data collected by the PITCHf/x system along with additional resources such as MLB.com, BrooksBaseball.net (applied machine learning to correct misclassified pitch types by PITCHf/x), and Fangraphs.com to interpret the data. To access the PITCHf/x data, I used an **R** **package** called **PitchRx** and stored the downloaded XML files into an **SQLite** database, later converted to MySQL to play with during the SQL training modules. While PITCHf/x data was available from 2006 to current date, I quickly realized, using SQL queries, groupby’s and aggregations, that the data was missing a lot of important information such as player names that were associated to unique ID’s generated by the system so they could not be verified or accessed through other resources. To identify the missing player, it would require accessing the team roster on MLB.com and comparing it to the database to populate the missing information. As I looked into the best approach to clean the PITCHf/x data, I came across a collection of API’s provided by Sportradar.com for multiple sports, including MLB. Upon reviewing the documentation, it was apparent that the best approach for wrangling the data I needed was to use Python with the MLB v6.5 API from Sportradar.

**The remainder of this document describes the data wrangling and data cleaning steps applied to the MLB data collected using Python 3.6 and an API.** Considering the goal of the project, to predict the next pitch thrown by an MLB starting pitcher, I needed to first collect pitch data for each game where the pitcher of interest pitched at least one (1) inning. Regardless of which website or resource I used to collect information such as pitch types, speed, and location, the data was originally recorded by PITCHf/x (2006 – 2014) or Statcast (2015 – current). Because of the change in systems used to track pitches, I decided I would not consider any data prior to 2015 and for this project, I decided to collect data from the 2016 and 2017 seasons. The MLB data available through the Sportradar API not only includes pitch types, speed, and location but also includes play-by-play information that I would have had to wrangle and clean separately from other resources had I continued to use PITCHf/x.

It is important to note that the API key required by Sportradar to access the data is fee-based, thus, I am currently using a 90-day free trial with up to 1,000 calls per month. Because of this, I limited the number of pitchers of interest to twenty (20) who were considered “top” pitchers during the 2016 or 2017 or both seasons. The list of pitchers of interest are:

pitchers = ['aaron\_nola','carlos\_carrasco','carlos\_martinez','chris\_archer',

'chris\_sale','clayton\_kershaw','corey\_kluber','dallas\_keuchel',

'david\_price','gerrit\_cole','jacob\_degrom','jake\_arrieta',

'jose\_quintana','marcus\_stroman','justin\_verlander','max\_scherzer',

'michael\_fulmer','stephen\_strasburg','yu\_darvish','zack\_greinke']

Approximately half of the pitchers in the list play for the National League while the other half play for the American League with some switching teams during the off-season or getting traded mid-season. While there are only a couple of pitchers that switched teams and even less that moved to different leagues (e.g., American → National), it would be interesting to see if this impacted the pitchers pitch selection.

Once the source of data was determined, the first step was to use standard HTTP requests with API key to download the Sportradar Glossary in JSON format which provides descriptions for pitch types (e.g., FA = Fastball, CT = Cutter, SL = Slider) and pitch outcomes (e.g., aBK = Balk, kF = Foul Ball, oG0 = Ground Out). Next, I downloaded the League Schedules for 2016 and 2017 regular seasons in JSON format which includes the Team Names, ID’s and more importantly the Game ID’s. The Game ID’s are unique identifiers for every game played and required to download the Play-by-play data files.

Using MLB.com website, I manually downloaded a list of all the games each pitcher pitched for their respective teams. Because the game dates and times differed from the actual game dates and times provided by MLB.com, I had to update the Sportradar dates by offsetting them by -7 hours so they would sync with MLB schedules. With the dates updated, I was able to collect the specific Game ID’s for every game that each pitcher pitched in 2016 and 2017 and download the Play-by-play JSON files.

CHALLENGES

Convert dates

De-nest column with list of nested dictionaries as compared to column with nested dictionaries

Inconsistencies in columns (missing columns)

Missing pitching speeds and pitch types

Additional columns in 2017 not present in 2016

Accents in player names

Unwanted columns

Data type conversions to reduce memory usage

Handling double headers

To do so, it was necessary for me to collect play-by-play data for each game where the starting pitcher of interest pitched at least one (1) inning in the game. The type of play-by-play information required, includes