Data Engineering Project Write-Up

By: Colin Gallagher

Abstract:

A data storage and processing pipeline was engineered to provide a user an easy way to visualize the data made available on the PGA Tour stats website. Data was scraped from the PGA Tour website and each statistic grabbed was made into its own SQL table, and when complied altogether made up a large SQL database. Using Python, the data was pulled in and helpful visualizations are made and displayed by using a Streamlit web app. Users can input any PGA golfer's name and select stats available that they want to visualize. Clients that would benefit from using this web app are anyone that is interested in the game's statistics available, but want to see visuals that are not offered on the PGA website.

Design:

The purpose of this project is create a data storage and processing pipeline that is designed in an efficient, modularized, and maintainable manner. The problem that this data storage and processing pipeline solves is a way for anyone interested in the statistics of the PGA Tour to view the data in a more visually pleasing way. Currently, a user can only view stats on their website as a leaderboard with a few options to filter the data with no visualizations what so ever. This pipeline gives users the option to type in anyone's name, the years interested, and stats that they would like visualized.

Data:

The raw data that was used for this project was scraped from the PGA Tour website. A total of 36 statistics were scrapped that were made up of 37,119 webpages total. Each page was a group of players season stats at a given time of the season through a certain tournament. The 36 statistics were made into their own tables in the SQL database. When combining all the rows of each table in the database there are a total of 7,436,863 rows of data.

Algorithms:

To scrape the data, first a list of links was acquired and built for the statistics of interest. This initial list had to be manually selected, unless every statistic available wanted to be scraped. For this project, the first set of links consisted of 36 webpages for 36 statistics. Using these links, the links for the actual scraping of the data was built. Using the 36 first links, the link codes for stat, year, and tournament were found and every combination of these codes was used to build the master link of lists which was 37,119 links. These links were scraped and the tables were saved as a data frame during each iteration. This data frame would grow in size as long as the statistic was not changing. When the statistic changed, the data frame would be saved to a SQL database as a table for that statistic. After the creation of the database, the data can then be used to create visualizations using Python and Streamlit. To manage and handle this much data, the web app has users input a golfer's name, years, and select statistics of interest using on/off checkboxes. This helps prevent a user from loading in every single table and every row in that table. The most someone could load in would be enter a golfer who played the longest, view all their years, and all the stats at one time which would be no where near 7,436,863 rows of data. Using these inputs, Streamlit processes the data filtering and visualization on their cloud web app hosting service. Due to Github upload restrictions, the database used for demonstration purposes was reduced to only one table or statistic (SG: Total). This table still has 171,037 rows of data.

Tools:

Tools used for building the data storage and processing pipeline:

- **Python**: Coding language used to perform actions on the data with its various libraries.
- **Pandas**: Python library used to clean and place data into organized data frames.
- **BeautifulSoup**: Python library used to scrape and parse through HTML to get the data.
- **SQL Alchemy**: Python library used to create and load in the database.
- **Streamlit**: Python library and web app hosting used to create the web app and host it on its website.

Communication:

A five minute slide presentation will be given to explain the pipeline, how it was created, the results, conclusions, any future work that can be done to improve the project, and answer any questions.