Spring 2023

Web Scraping MLB Statistics to Predict Player Salaries Based on Performance

Alexander J. Schoessler , '23

Web Scraping MLB Statistics to Predict Player Salaries Based on Performance

Alex Schoessler

Advised by Allan Moser

Department of Engineering

Swarthmore College

ENGR 090

05/05/2023

Introduction

Baseball is one of the most popular sports in the world, referred to today as America’s pastime. Millions of fans follow Major League Baseball(MLB) year after year. Advancement in technology has allowed for the increasing availability of player data and statistics. Much player data for the MLB is located online, on websites such as ESPN or baseball-reference. Web scraping is a tool that allows for this data to be extracted and edited. With a recent rise in machine learning techniques, statistical analysis and predictive modeling can be used to make new observations. In this paper, the correlation between player performance and salary predictions will be investigated.

The goal of this paper is to display a web scraping process that allows for the creation of a dataset of MLB statistics for qualifying players, as well as the machine learning processes used to predict player salaries. These predicted salaries can thus be compared to the player's actual salaries. Players can be evaluated as overpaid, properly paid, or underpaid, which may occur for a variety of reasons. It is likely that MLB front offices have begun this type of analysis already. However, this type of analysis could influence how organizations view players and shape roster decisions.

Behind the Statistics

The statistics used in the dataset were chosen for their significance in regards to measuring a player's performance. Each of the statistics are relevant to a player's ability in a different way. This section is dedicated to describing each of the statistics used in the data set in detail [1].

Pitcher Statistics

- *Games Played(GP)*: The number of games that the pitcher appeared in.
- *Games Started(GS)*: The number of games in which the pitcher starts the game, meaning that they throw the first pitch of the game for their team to the first opposing batter.
- *Innings Pitched (IP)*: Measures a pitcher's appearance in the game by the number of outs recorded. 1 out recorded designates 0.1 IP, 2 outs designates 0.2 IP, and 3 outs designates 1 IP.
- *Wins(W)*: A pitcher can record a win when he is the pitcher of record in which their team takes the lead for the remainder of the game. If the pitcher is the starting pitcher, they must throw at least 5.0 innings to qualify. Otherwise, the official scorer can determine which relief pitcher will earn the win based on their performance.
- *Losses(L)*: A pitcher can record a loss when they give up what is deemed to be the "winning" run, in which case a run is given up giving the other team the lead that they do not give up.
- *Win Percentage(WP)*: The percentage of wins that a pitcher receives divided by the number of combined total wins and losses that pitcher has.
- *Earned Runs(ER)*: Defined as any run that a pitcher gives up that was not the result of either an error or passed ball.
- *Earned Run Average(ERA)*: The number of earned runs that a pitcher gives up per 9 innings.
- *Walks Plus Hits Per Inning Pitched(WHIP)*: This statistic is essentially the number of baserunners that a pitcher gives up per inning pitched. It is calculated by adding walks and hits against, divided by the number of innings pitched.

- *Strikeouts(K, SO)*: A strikeout is when a pitcher records 3 strikes on a hitter in a single at-bat. A strikeout must end on a strike swinging or looking, and a foul ball can not be the final pitch of the at-bat that results in a strikeout.
- *Walks(BB, Base on Balls)*: A walk, also known as Base on Balls, occurs when a pitcher throws four balls to a batter in a single at bat. This results in the batter being automatically awarded first base.
- *Strikeout to Walk Ratio(K/BB)*: The number of strikeouts a pitcher has recorded divided by the number of walks.
- *Saves(S)*: A save is defined as the pitcher of record who finishes the game for his team. A pitcher can not record both a win and a save. There are certain conditions in which a pitcher can record a save, including:
  1. A pitcher can enter a game, with their team leading by 3 runs or less, and record 1 inning pitched.
  2. A pitcher can enter the game, with the tying run either on base, at-bat, or on deck.
  3. A pitcher can record at least 3 innings pitched, while still finishing the game for their team.
- *Holds(H)*: A hold can be recorded if a pitcher enters a save situation, records at least 1 out, leaves the game without recording a save, and the team wins the game without relinquishing the lead.
- *Blown Saves(BS)*: Occurs when a player enters the game in a save situation and gives up the tying run.

Batter Statistics:

- *Games Played(GP)*: See Pitcher Statistics
- *At-Bats(AB)*: Occurs when a player takes their turn to bat, discounting walks, being hit by a pitch, and sacrifice hits.
- *Runs(R)*: A player scores a run by crossing the plate.
- *Hits(H)*: A hit occurs when a player hits the ball and reaches a base safely, without an error or out being recorded.
- *Doubles(2B)*: Occurs when a player records a hit and reaches second base safely.
- *Triples(3B)*: Occurs when a player records a hit and reaches third base safely.

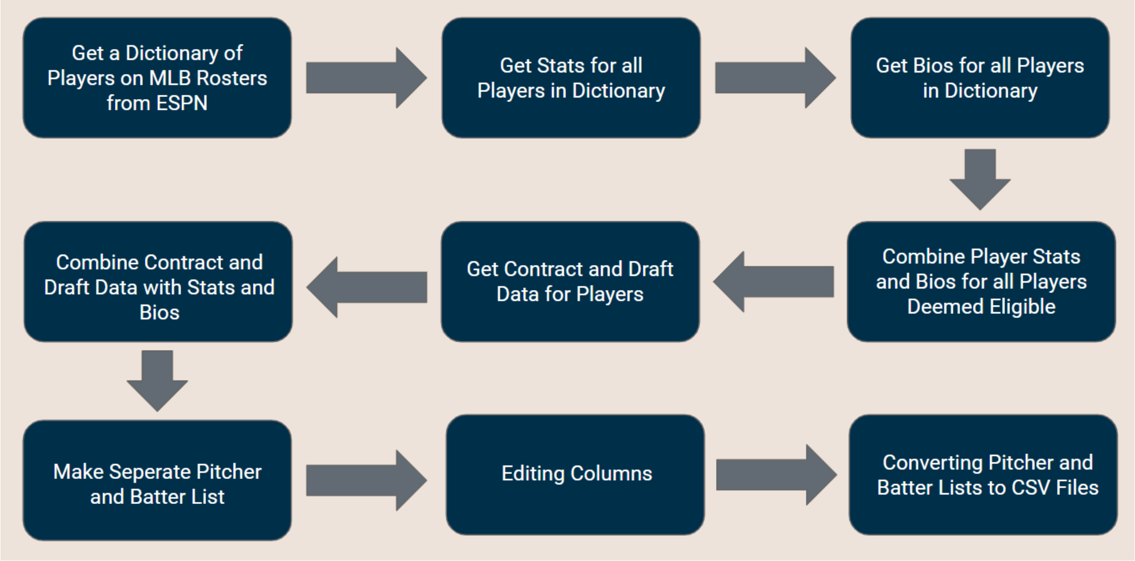
- *Home Runs (HR)*: Classified as a hit that allows the player hitting the ball to reach home plate and score a run, without an error. This typically occurs when a player hits the ball over the fence, or when the ball hits a foul pole in the air.
- *Runs Batted In(RBI)*: An RBI is recorded when a player causes a run to score. This can occur in many ways including a walk, a batter ball in play unaided by an error, a sacrifice play, a hit by pitch, a fielder's choice, or with less than 2 outs when a run scores on an error where a runner on 3rd base would have scored despite the error.
- *Walks(BB, Base on Balls)*: See Pitcher Statistics.
- *Hit-by-Pitch(HBP)*: When a player at-bat is hit with a thrown ball by the pitcher.
- *Strikeouts(K, SO)*: A strikeout is when a pitcher records 3 strikes on a hitter in a single at-bat. A strikeout must end on a strike swinging or looking, and a foul ball can not be the final pitch of the at-bat that results in a strikeout.
- *Stolen Bases(SB)*: Occurs when a player runs from one base to another successfully when the pitcher throws the ball to home plate.
- *Caught Stealing(CS)*: When a player runs from one base to another unsuccessfully when the pitcher throws the ball to home plate, and is thrown out.
- *Batting Average(AVG)*: Defined as the number of hits a player records divided by the number of at-bats the player records.
- *On-Base Percentage(OBP)*: The percentage of the time that a batter gets on base. This is the total number of hits, walks, and hit-by-pitches a player tallies divided by the total number of plate appearances.
- *Slugging Percentage(SLG)*: The total number of bases a batter achieves per at-bat, counting only the number of hits. A player tallies one base for a single, two for a double, three for a triple, and four for a home run.
- *On-Base Plus Slugging Percentage(OPS)*: Defined as the addition of a player's on-base percentage and their slugging percentage.

Creating the Datasets

One of the main goals of this report was to use web scraping techniques and other python methods to create datasets from scratch that could be used for analysis. The scripts used to complete this were created in python with the Spyder IDE. Throughout the creation of the datasets, multiple websites were used to scrape player statistics from. The first of these was baseball-reference. This site contains many different player statistics of major league baseball players. Their statistics include standard batting statistics, advanced batting statistics, defensive statistics, and more for all of the available players. After working with this information for a while, it became apparent that using this website would not be possible. Due to web scraping laws, a user can not violate the privacy policy of a given website. The baseball-reference website contains a request limit, meaning that a written program can only reach out to the website for information every so often. For the purposes of this project, this limit was slow enough to warrant errors in the code. Therefore, another website was chosen to scrape player statistics from.

This new website used was ESPN. The results with this site were much better than with baseball-reference. The figure below details a high-level overview of the structure of the code used to create the datasets.

Figure 1: Overview of the Structure of the Code Used to Create Datasets


Get a Dictionary of Players on MLB Rosters from ESPN
Get Stats for all Players in Dictionary
Get Bios for all Players in Dictionary
Combine Player Stats and Bios for all Players Deemed Eligible
Get Contract and Draft Data for Players
Combine Contract and Draft Data with Stats and Bios
Make Seperate Pitcher and Batter List
Editing Columns
Converting Pitcher and Batter Lists to CSV Files

The program begins with a function called get_players_dict(). This function starts by requesting information from the link for ESPN. Then, for every team found on the list of teams page and creates a link for the roster of every team. Then, a dictionary is created and for every link in this list, soup is created. The soup is essentially all of the information stored on the page which had been parsed with an HTML parse. Then, basic player information was found using the find.all method from BeautifulSoup. This method searched through all tags and then gathered all of the specified tags of the specified class, which contained the player information. Then for every player on every team, the player's name and the link to their individual statistics page was added to the dictionary. Using the pickle library, a file was created that stored the players dictionary for later use. This saved time throughout the course of the project, as this segment of the code would not have to be rerun.

The next function created in the program was called get_player_stats(). This function began by using the pickle library to read in the file containing the players dictionary stored in the previous function. Then, for every player on every team, soup was created for each player's stats page. The find.all method was used to search for each player's individual stats and add them to a list. Each player's stats were added to a new dictionary, where each player was also labeled as either a pitcher or a batter depending on their position. This dictionary containing player names, positions, and stats for each player was stored as a file using pickle.

Then, another function called get_player_bios() was created. In this function, the player statistics dictionary was read in, and string manipulation was used to change the link to every player's statistics link to a link to their bio page. This page contains other information such as a player's age, date of birth, hometown, and team. A similar process was used as in get_player_stats() to get bio information for each player. Soup was created for each player's bio page and using the find.all method, each player's individual bios were stored in a list. One of the items found on a player's bio page was experience, essentially the number of seasons a player had spent in the league. This information was incorrect for some players, labeling them as a "ROOKIE" even when this was not the case. Thus, experience was dropped from each player's individual bio list before a new dictionary was created that contained all of the bios for each player. This bio dictionary was stored as a file using pickle.

Next, a function called combine_stats_bios() was created to combine the statistics dictionary and the bio dictionary into a singular dictionary. This began by first determining which player's needed to be kept. It was decided that players who have played only a very small amount of time in Major League Baseball, less than a season, would be removed. This is because keeping these players would likely skew predictions to be made after the datasets were completed. Thus, a player was deemed eligible if they were a position player and had recorded at least 400 career at bats or if they were a pitcher and had recorded at least 50.0 career innings pitched. Then, a new dictionary was created that matched the players in the statistics dictionary with those in the bios dictionary. This new dictionary thus contained the bios and statistics for all of the players deemed eligible and was stored as a new file.

The following function used to create the datasets was make_spotrac_players_dict(). One of the main goals of this project was to use the acquired data to make player salary predictions based on player performance. Thus, contract information was needed for all eligible players. All of the previous functions had gathered information from ESPN. However, ESPN does not have player contract or draft information. Thus, another site, Spotrac, was used. This website contains salary information for players and organizations for many sports, including Major League Baseball. Thus, in the make_spotrac_players_dict() function, a new soup was created for a link to the Spotrac teams page. For every team, the link to that teams roster page was appended to a list of team links. Then for every link in this list, a similar dictionary was created to that made in get_players_dict(). This dictionary contained every player on every team and a link to the player's page on the Spotrac site. This new dictionary was saved with pickle as a file.

Next, get_draft_data() was written following the same format as get_player_stats(). For every player on every team in the Spotrac players dictionary, soup was created for the link to their individual player page. From this soup, a player's individual draft information and experience information was found with the find.all method in BeautifulSoup. This information was appended to individual lists and then combined into a dictionary that contained draft info and experience for each player. This dictionary was then saved with pickle. A new function called get_contract_data() was created. This new function followed the exact same procedure as get_draft_data. However, instead of gathering draft and experience data, the class found with the find.all method was altered to reflect that contract data was being scrapped instead. A dictionary that contained contract data for all players was created and stored with pickle.

The next function used in the creation of the datasets was combine_draft_contract_data(). This function first combined the dictionary that contained draft and experience information with the dictionary that contained contract information for all players. The output of this was a new dictionary that contained draft, experience, and contract information for all players from the Spotrac website. Now, all of the information from Spotrac was in a single dictionary and all of the information from ESPN was in another. Thus, these dictionaries were combined into a single dictionary. This final dictionary thus contains player statistics, bios, draft, contract, and experience information. Player names were matched across the previous dictionaries to include only eligible players pulled from the ESPN website in the new dictionary. This dictionary was saved using the pickle library.

The next function, make_pitcher_batter_lists(), sought to break up the dictionary made in combine_draft_contract_data() into lists of pitchers and batters. This was necessary for data analysis as pitchers and batters have different statistics, so they would need to be evaluated separately. Thus, if a player was a pitcher, he was moved to a list of pitchers and their statistics called eligible_pitcher_data. If a player was a batter, he was moved to a list of batters and their statistics called eligible_batter_data. Both of these lists were saved in pickle for later editing.

After making these lists, it became necessary to edit the lists for data analysis. Thus, the function edit_columns() was created. This function performed many different operations. Mainly, for each list, string manipulations were performed on various list elements. This was mainly necessary to remove unnecessary text in certain elements so those elements could be converted to floats or integers for data analysis. Additionally, some new columns were created. This included separating age from birthdate as well as contract length from total contract value. In this function, some elements were deleted entirely as they contained dashes or no information for many to all players, thus making them irrelevant. Finally, new columns were created for some statistics that were accumulated as totals over a player's career. These new columns were created as averages by taking the career total statistic and dividing it by the player's experience to get a 162 game average for that statistic for each player. The value 162 was chosen as this is the number of games in a full Major League Baseball season. Thus, the lists were extended to incorporate these new statistics. Therefore, the lists containing all pitcher and batter data were finalized and saved as files with the pickle library.

The final function used in this program was called convert_lists_to_csv(). Like it sounds, this list was used to convert the lists containing all pitcher and batter data into CSV files. For each list, the fields were labeled to reflect the order of information in each list. Then, a new data frame was created for each list. Each data frame was then converted to a CSV file with the given fields. Thus, the datasets were completed containing all of the scrapped information for pitcher and batters.

Writing Scripts to Predict Player Salaries

After constructing datasets to be analyzed, a new Python script was written in the Spyder IDE which sought to predict player salaries based on their performance. In this script, the main libraries used were pandas and sklearn. The library sklearn was chosen for its use in predictive modeling. Three separate scripts were written for analysis, one for batters, one for starting pitchers, and one for relief pitchers. Starting pitchers and relief pitchers were separated as relievers are paid much lower salaries than starters while often having better statistics on average. Relievers are generally paid less than starters due to recording a significantly lower amount of innings pitched. Thus, they were split up for different analyses.

For each of the three position categories, the relevant CSV file was read in. Then, certain features were dropped from the data frame due to their insignificance to the analysis or correlation with other features in the dataset. Player name was set as an index to thus be added back in later. An X variable was created that included the remaining data frame without the Average Annual Value(AAV) variable. AAV is the player's average salary over the length of their contract. A Y variable was created that was set equal to AAV. Then, a test-train split was done to train the model, where a linear regression model was used. The positions had a test size of 27% for batters, 20% for starting pitchers, and 20% for relief pitchers. This differed based on the number of features and players in each position group. The model was then fit to the training data and predictions were made using the .predict method. After this was completed, actual salary predictions were made on the data, again using the .predict method. The mean absolute error was found for each position group along with an r-squared score that evaluated the fit of each model. Finally, a new data frame was created for each position group that for each player included their name, actual AAV, and predicted AAV. Each of these data frames were exported as CSV files.

Results

Table 1: Mean Absolute Error and R-Squared Score for Each Predictive Model

Position Type	Mean Absolute Error	R-Squared Score
Batters	$ 4,287,044.09	0.5695
Starting Pitchers	$ 3,662,093.58	0.5929
Relief Pitchers	$ 1,280,407.98	0.5026

Table 2: Predicted Top 5 Highest Paid Batters

Player Name	Mike Trout	Miguel Cabrera	Aaron Judge	Mookie Betts	Juan Soto
AAV	$35,541,667.00	$31,000,000.00	$40,000,000.00	$30,416,667.00	$23,000,000.00
Predicted AAV	$30,274,644.39	$27,232,603.28	$24,980,298.53	$23,995,775.01	$23,773,083.80

Table 3: Predicted Top 5 Highest Paid Starting Pitchers

Player Name	Zack Greinke	Clayton Kershaw	Max Scherzer	Chris Sale	Charlie Morton
AAV	$8,500,000.00	$20,000,000.00	$43,333,333.00	$29,000,000.00	$20,000,000.00
Predicted AAV	$27,509,562.24	$25,257,790.33	$24,916,483.12	$21,227,231.58	$20,336,533.76

Table 4: Predicted Top 5 Highest Paid Relief Pitchers

Player Name	Kenley Jansen	Craig Kimbrel	Aroldis Chapman	David Robertson	Ian Kennedy
AAV	$16,000,000.00	$10,000,000.00	$3,750,000.00	$10,000,000.00	$2,250,000.00
Predicted AAV	$10,946,863.47	$10,472,217.59	$9,887,398.01	$9,417,314.48	$8,792,051.53

Table 5: Predicted Top 5 Most Undervalued Batters

Player Name	Nelson Cruz	Jason Heyward	Evan Longoria	Adley Rutschman	Andrew McCutchen
AAV	$1,000,000.00	$720,000.00	$4,000,000.00	$733,900.00	$5,000,000.00
Predicted AAV	$18,148,028.79	$15,875,867.45	$17,269,556.09	$13,796,241.53	$17,932,484.86
Salary Difference	$17,148,028.79	$15,155,867.45	$13,269,556.09	$13,062,341.53	$12,932,484.86

Table 6: Predicted Top 5 Most Overvalued Batters

Player Name	Dansby Swanson	Carlos Correa	Francisco Lindor	Corey Seager	Aaron Judge
AAV	$25,285,714.00	$33,333,333.00	$34,100,000.00	$32,500,000.00	$40,000,000.00
Predicted AAV	$ 8,380,663.50	$16,523,227.77	$18,068,783.50	$17,351,482.55	$24,980,298.53
Salary Difference	-$16,905,050.50	-$16,810,105.23	-$16,031,216.50	-$15,148,517.45	-$15,019,701.47

Table 7: Predicted Top 5 Most Undervalued Starting Pitchers

Player Name	Zack Greinke	Wade Miley	Rich Hill	Kenta Maeda	Corey Kluber
AAV	$ 8,500,000.00	$ 4,500,000.00	$ 8,000,000.00	$ 3,125,000.00	$10,000,000.00
Predicted AAV	$27,509,562.24	$15,785,160.04	$18,031,144.20	$11,069,167.79	$17,564,151.27
Salary Difference	$19,009,562.24	$11,285,160.04	$10,031,144.20	$7,944,167.79	$7,564,151.27

Table 8: Predicted Top 5 Most Overvalued Starting Pitchers

Player Name	Jacob deGrom	Max Scherzer	Gerrit Cole	Luis Castillo	Marcus Stroman
AAV	$37,000,000.00	$43,333,333.00	$36,000,000.00	$21,600,000.00	$23,666,667.00
Predicted AAV	$16,794,203.03	$24,916,483.12	$17,921,541.55	$10,375,096.48	$12,768,987.24
Salary Difference	-$20,205,796.97	-$18,416,849.88	-$18,078,458.45	-$11,224,903.52	-$10,897,679.76

Table 9: Predicted Top 5 Most Undervalued Relief Pitchers

Player Name	Ian Kennedy	Jesse Chavez	Aroldis Chapman	Jeurys Familia	Will Smith
AAV	$ 2,250,000.00	$ 1,200,000.00	$ 3,750,000.00	$ 1,500,000.00	$ 1,500,000.00
Predicted AAV	$ 8,792,051.53	$ 7,351,767.80	$ 9,887,398.01	$ 6,767,960.94	$ 6,695,345.65
Salary Difference	$ 6,542,051.53	$ 6,151,767.80	$ 6,137,398.01	$ 5,267,960.94	$ 5,195,345.65

Table 10: Predicted Top 5 Most Overvalued Relief Pitchers

Player Name	Ryan Pressly	Josh Hader	Rafael Montero	Nick Martinez	Taylor Rodgers
AAV	$15,000,000.00	$14,100,000.00	$11,500,000.00	$8,666,667.00	$11,000,000.00
Predicted AAV	$6,309,370.88	$6,808,596.70	$4,221,990.20	$2,866,655.56	$5,529,842.12
Salary Difference	-$8,690,629.12	-$7,291,403.30	$7,278,009.80	-$5,800,011.44	-$5,470,157.88

Discussion

Based on the results of the model, mean absolute errors were calculated for each of the three position groups. In this case, mean absolute error demonstrates the average absolute difference between the players actual salary and their predicted salary for all players across the entire dataset. Thus, for batters, the average difference between the predicted and actual salary was $4,287,044.09. For starting pitchers, the average difference between the predicted and actual salary was $3,662,093.58. For relief pitchers, the average difference between the predicted and actual salary was $1,280,407.98. R squared was used to measure the fit of the model for each position group. I would consider these scores to be moderate for a predictive model as they each fall in the 0.5-0.6 range. Thus, about 50-60% of the data is captured by the predictions made. However, regardless of the model used, there are many factors that determine player salaries. This model did not include many advanced statistics used to evaluate players, although it did include the main statistics. Additionally, this model did not include any defensive statistics. Furthermore, no predictive model can capture intangible qualities such as dedication to the sport or the qualities of a good teammate. No matter how good the model could be, there are other factors to consider when making roster decisions.

For every player in the model, a predicted salary was created and can be compared to the player's actual salary. The top 5 batters in terms of predicted salary are all players who have exemplary stats over a 162 game span. This trend continues for both starting pitchers and relief pitchers. However, when looking at the most overvalued and undervalued plates per the model, a different trend becomes apparent. When beginning this project, it was predicted that younger players who are being paid a low salary would be the most undervalued, while older players on high salaries would be the most overvalued. As it turns out, the model valued older players who have had strong careers as the most undervalued, and current players in their prime on high AAV salaries as the model overvalued. Therefore, the model likely did not incorporate age as much as expected. However, a motivation for this project was recent long term contracts with high AAV. Each of the top 5 batters deemed overvalued signed one of these contracts in the last two years. Thus, the model seems to predict that these players are not worth their current contracts, and teams overpaid these players to get them on their rosters. A similar trend extends beyond batters to both starting pitchers and relief pitchers. Older players with strong careers on low salaries were considered the most undervalued, while players in their prime on high salaries were considered the most overvalued. Thus, more experimentation with variable weighting as well as experimenting with other machine learning models could make this model more accurate However, if this model was to be shown to MLB organizations, one could still expect them to possibly reconsider the high prices they are paying for some players and find other players with better value.

Engineering Design

The following section is dedicated to answering the following questions in regards to how this project followed components of engineering design.

1. What constraints govern your design problem?

Some constraints that governed my design problem was the limited access to data that I had. Some websites, as discussed above, had request limits that prevented me from scraping these sites in a timely manner. As a result, I chose to move away from these sites and gather data from others. Additionally, I had very limited Python experience and I needed to learn how to use the BeautifulSoup library from scratch. I also had no prior experience with predictive modeling, which meant I needed to do a lot of research regarding which models would be successful for my project. Thus, it took a large amount of time to learn these topics and apply what I learned to my project.

2. What requirements did you develop?

I had developed requirements to complete a web scraping algorithm along with a statistical analysis program.

3. How did you evaluate your solution against the requirements?

My solution was considered a success as I was able to write a web scraping program that outputted two CSV files containing pitcher and batter data. I was also able to write a predictive modeling script that predicted player salaries given the statistics and other information compiled in the datasets. This model then outputted these salaries and allowed for comparison between actual and predicted salaries. Overall, the predictive modeling portion of this project could be refined and future work could be completed to make more accurate predictions.

4. What professional standards and codes, if any, govern your design (and if none, state why or that you were unable to find any).

The main standards that I needed to follow was to not infringe on the privacy policies of certain websites. Web scraping has recently become legal, and is allowed for publically available data on the internet. Since market share was not stolen, the data was publicly available, the scraper does not overburden the sites used, and the information found was factual in nature, this program was considered to be an ethical scraper and did not violate any laws[2].

Conclusion

Web scraping can be a valuable technique used to acquire statistical information. This powerful tool allows researchers to extract and edit data in a manner they deem appropriate. Web scraping was very valuable over the course of this project and allowed for the collection of data for Major League Baseball players. After this data was collected, predictive modeling allowed for salary predictions to be made for players in the datasets. Overall, the model was moderately accurate in predicting player salaries. Further experimentation with weighting variables and the use of other machine learning techniques could be used to conduct a more accurate prediction of player salaries. Future research on this topic could include scraping additional sites to gather more statistics for each player and create a larger dataset. Additionally, the data could be rearranged to include year over year statistics instead of career statistics. However, this would significantly complicate the structure of the data. Overall, the goals of this project were to create datasets that can be used for analysis and to use these datasets to engage in predictive modeling to predict player salaries based on performance. Each of these goals was achieved, and the work completed on this project can be used to analyze roster decisions made by Major League Baseball Organizations.

References

[1] "Glossary." *MLB.com*, https://www.mlb.com/glossary.

[2] Urban, Ondra. "Is Web Scraping Legal?" *Apify Blog*, Apify Blog, 27 Apr. 2023.

Appendix

The following link is for the code used to complete this report:
https://github.swarthmore.edu/aschoes1/Alex-Schoessler-E90-Project-Code

Acknowledgements

A special thank you to Professor Allan Moser for advising me on this project and his guidance throughout this process!