Capstone 1 Milestone Report: Predicting MLB Players Salaries

Problem Statement

Throughout the years MLB teams have lost loads of money by overpaying for certain athletes. For example, let's say the New York Yankees signed player x for a multi-million dollar contract, and in the preceding years the player's performance was subpar. In this scenario the team would be at a huge loss because they invested millions into a player who is not performing at their perceived value.

Now, from a players viewpoint they also need to know their own worth. In the MLB there have been many contract signings where players have been undervalued. In this scenario the player should have just as much knowledge and power to know their value and be able to negotiate with these teams. In this project, I plan to build a model to predict player salaries based on their previous year stats which can be used both by sports franchises, to minimize their risk, and by players, to help them determine their value.

Data Wrangling

For this project I decided to create two separate dataframes, one consisting of pitchers and another for hitters/position players. This was due to the fact that pitchers and hitters each have their own set of recorded metrics. For example, a pitcher's measure of success is heavily weighted towards statistics such as 'W' Wins per season, 'K' Strikeouts per season, and 'ERA' Earned Run Average. Hitters are mostly focused on 'H' Hits per season, 'HR' Home Runs per season, and 'BAVG' Batting Average.The data will be scraped from 'http://www.thebaseballcube.com/', a baseball data warehouse. The metrics for each data set can be seen below…

Pitcher Dataframe

Hitter Dataframe

```
Data columns (total 41 columns):
playerName           7696 non-null object
salary               5266 non-null float64
adj_salary_filled    7696 non-null float64
flag                 7696 non-null int64
Age                  7696 non-null int64
HT                   7696 non-null object
WT                   7696 non-null int64
Bats                 7696 non-null object
Throws               7696 non-null object
year                 7696 non-null int64
teamName             7696 non-null object
posit                7696 non-null object
borndate             7696 non-null object
Place                7696 non-null object
LeagueAbbr           7696 non-null object
W                    7696 non-null int64
L                    7696 non-null int64
G                    7696 non-null int64
GS                   7696 non-null int64
CG                   7696 non-null int64
SHO                  7696 non-null int64
GF                   7696 non-null int64
SV                   7696 non-null int64
IP                   7696 non-null float64
H                    7696 non-null int64
HR                   7696 non-null int64
R                    7696 non-null int64
ER                   7696 non-null int64
BB                   7696 non-null int64
IBB                  7696 non-null int64
SO                   7696 non-null int64
WP                   7696 non-null int64
BK                   7696 non-null int64
ERA                  7696 non-null float64
h9                   7696 non-null float64
hr9                  7696 non-null float64
bb9                  7696 non-null float64
so9                  7696 non-null float64
WHIP                 7696 non-null float64
total_years_mlb      7696 non-null int64
minimum_year         7696 non-null int64
dtypes: float64(9), int64(23), object(9)
memory usage: 2.4+ MB
```

```
Data columns (total 38 columns):
playerName           6924 non-null object
salary               5047 non-null float64
adj_salary_filled    6924 non-null float64
flag                 6924 non-null int64
Age                  6924 non-null int64
HT                   6924 non-null object
WT                   6924 non-null int64
Bats                 6924 non-null object
Throws               6924 non-null object
posit                6924 non-null object
borndate             6924 non-null object
Place                6924 non-null object
teamName             6924 non-null object
LeagueAbbr           6924 non-null object
G                    6924 non-null int64
AB                   6924 non-null int64
R                    6924 non-null int64
H                    6924 non-null int64
Dbl                  6924 non-null int64
Tpl                  6924 non-null int64
HR                   6924 non-null int64
RBI                  6924 non-null int64
SB                   6924 non-null int64
CS                   6924 non-null int64
BB                   6924 non-null int64
IBB                  6924 non-null int64
SO                   6924 non-null int64
SH                   6924 non-null int64
SF                   6924 non-null int64
HBP                  6924 non-null int64
GDP                  6924 non-null int64
Bavg                 6924 non-null float64
Slg                  6924 non-null float64
obp                  6924 non-null float64
OPS                  6924 non-null float64
year                 6924 non-null int64
total_years_mlb      6924 non-null int64
minimum_year         6924 non-null int64
dtypes: float64(6), int64(23), object(9)
memory usage: 2.0+ MB
```

- **Missing Values**

One of the first steps I took towards cleaning my dataset was searching for any missing values. During my search I found a significant amount of null values in both of my datasets. The pitcher dataset had 16% of salaries missing and the hitter dataset had 13%. This was significant because all of the missing values were located in the target variable ‘Salary’. After some research i decided not to drop the missing values but to instead fill them. I filled each missing value with the minimum salary for that respective year. Due to the fact that the MLB has been increasing their minimum salary throughout the years, I figured I couldn't just fill the nulls with one single value.

In order to fill in the null values I created a dictionary with years as the key and minimum salary for that year as a value. I then created a new column called ‘salary_filled’, by using the map function to iterate through each row and fill any missing values with the salary dictionary.

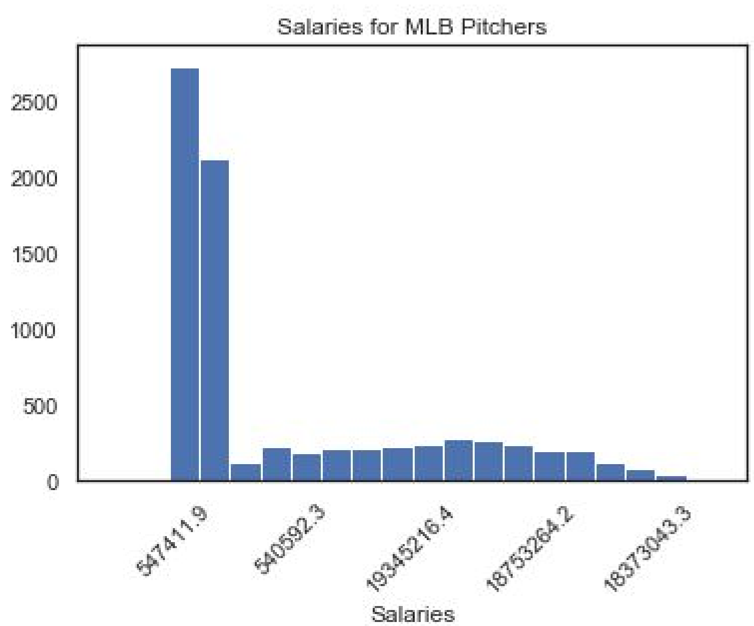
- **Feature Engineering**

After dealing with all the missing values and rearranging columns, I started to do some feature engineering. I needed to figure out how many years each player had been in the MLB. Both datasets were unorganized and only included players by groups for each season. Implementing this feature in each data set will help with organization and filtering going forward.

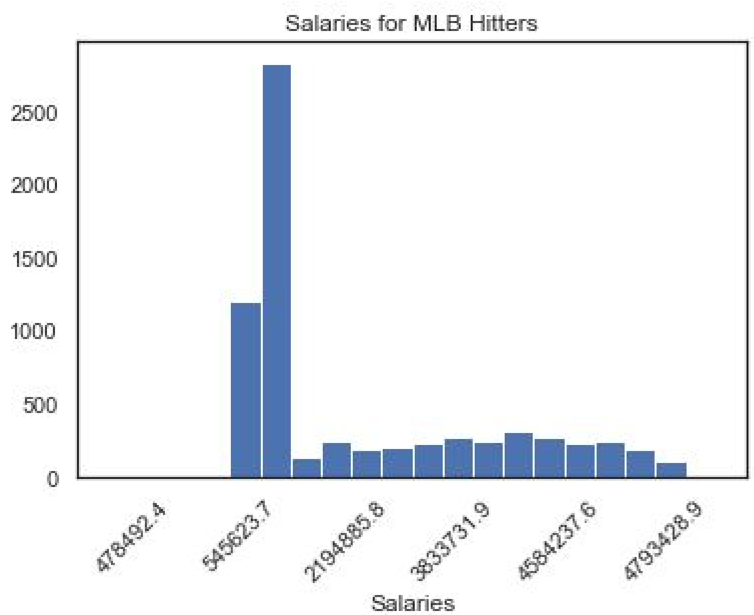
In order to create this new feature I sorted the entire dataset by columns ‘playerName’ and ‘year’. Next, I used the Pandas groupby function to split my dataset into groups for each player, and chained it with a cumulative count method. The new feature was called ‘total_years_mlb’, and returned the number of years in the MLB for each player . I also added another feature which returned the minimum salary for each year in ‘total_years_column’. This was built by using the apply method with a lambda function throughout each row in the ‘year’ column. The addition of these new features will help me gain a more valuable insight into my data.

Exploratory Data Analysis

- **Data Storytelling**
    - **Distribution of Salaries(Target Variable) for Pitchers and Hitters**

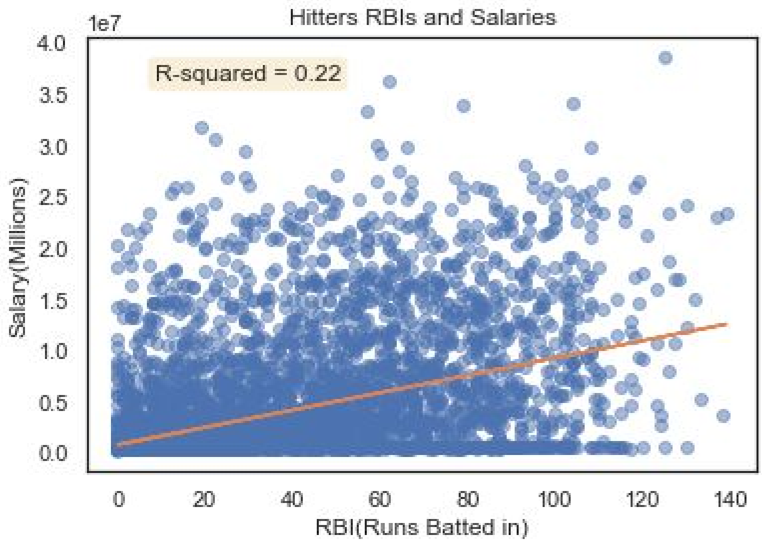


The chart above depicts the salary distribution for pitchers in the MLB. According to the chart most of the pitchers earn a salary around $550,000, which is close to the minimum salary amount. As we move to the right, we can see a drop off for players earning more than the league minimum. Towards the end of the chart there are less than 100 players earning a salary of $20 million or more.

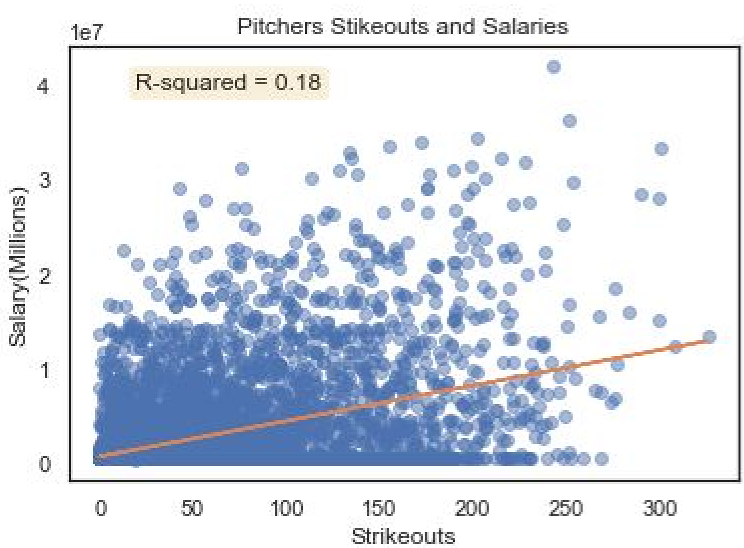


Similarly to the pitchers distribution of salaries, hitters too mostly earn the league minimum around $550,000. As we move to the right, we can see a drop off for players earning more than the league minimum. Towards the end of the chart we can see that the distribution of salaries for hitters goes all the way up $50 million.

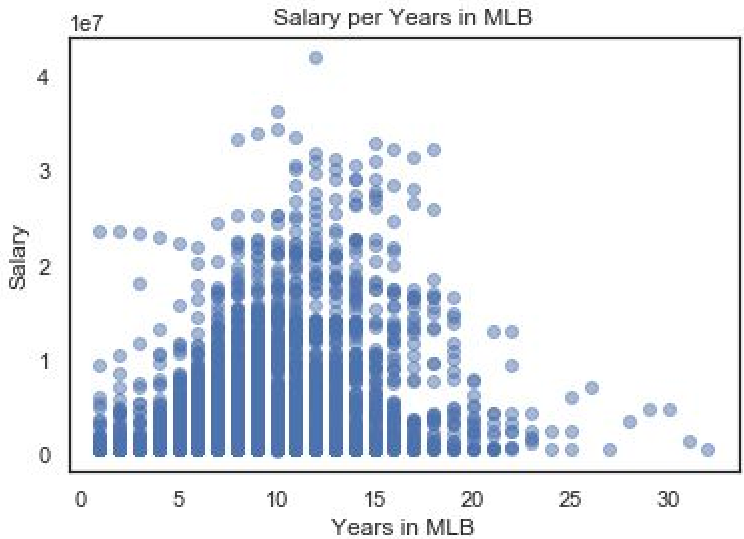
- **Data Storytelling**
    - **Independent Variables Vs. Dependant Variable**



The chart above represents a positive correlation between a player's RBI(runs batted in) and their salary. This feature also has the strongest level of correlation with the target variable within the hitter dataset. As the number of RBI's increases so does a player's salary. We can also see a couple of data points with high RBI totals and a low salary. This could be due to exceptionally good rookies just joining the league and being paid the minimum.

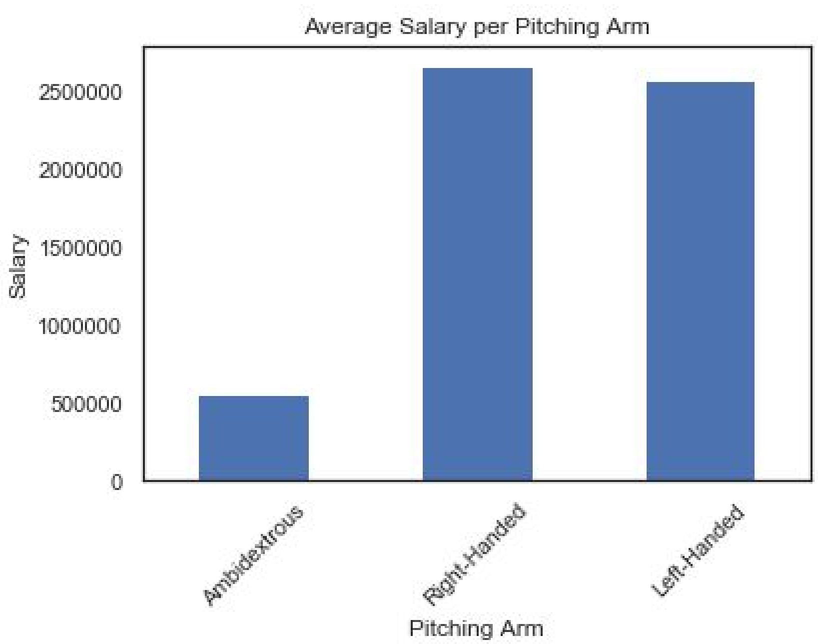


The chart above represents a positive correlation between a pitcher's strikeouts and their salary. This feature also has the strongest level of correlation with the target variable within the pitcher dataset. As the number of strikeouts increases so does a player's salary. The data includes different types of pitchers such as starting pitchers, relief pitchers, and closers. Number of strikeouts is one of the top measured pitching metrics per season.



The chart above depicts a weak or non-existent positive correlation between ‘Years in MLB’ and ‘Salary’. During the first couple of years in the MLB we can see a gradual increase in salary. We can also see that a player's maximum salary tends to peak after 7-10 years of MLB service. After 15 years of MLB service a player's salary begins to decrease rapidly. This could be due to aging and below average performance.

Statistical Inference



Is there a difference between the salaries paid towards right handed and left handed pitchers?

H_0 : The salaries for both left and right handed pitchers are the same.

H_1 : The salaries for left and right handed pitchers are different.

An independent-samples t-test was conducted to compare the salaries paid to left and right handed pitchers, with an alpha level set at .05. Results of the independent sample t-tests indicated that there were not significant differences in salaries paid to pitchers who threw with their left or right arm, ($t(7689) = -0.75$, $p = 0.77$). Specifically, our results suggest that pitchers are paid the same regardless of throwing arm.