Baseball Batting Orders

Optimization Using
Simulated Gameplay and Neural Networks

Claude Fried

2021

Problem Statement

In all sports, analytics are being used to leverage every possible advantage.
In modern baseball particularly, there is a wealth of data to study and model.

- Can the **outcome of an at-bat** be modeled and predicted?

How can these predictions be used to improve performance?

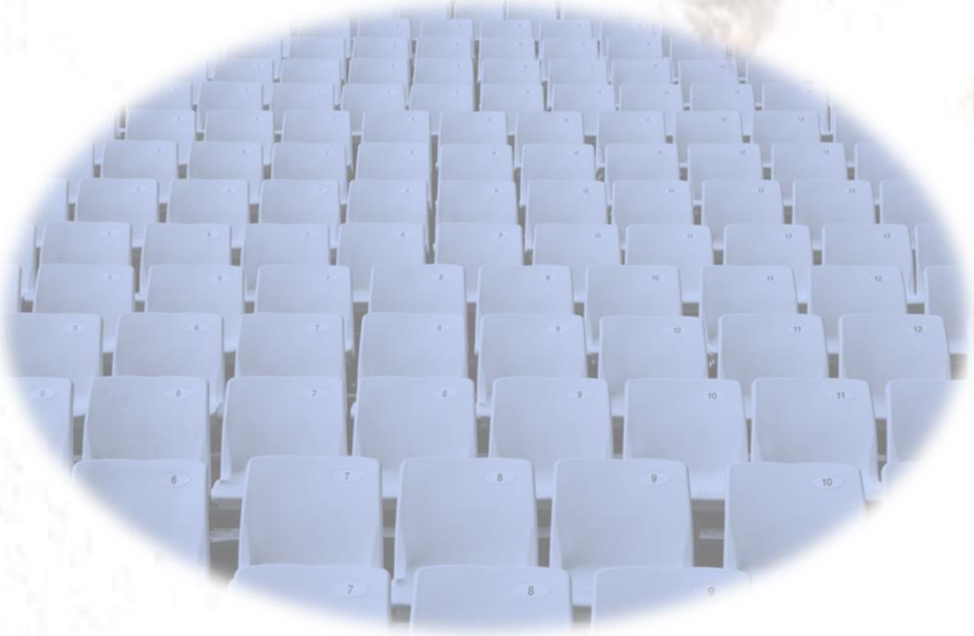
  - Does the **batting order** impact the expected runs scored?
  - Can team performance improve by setting an **optimal batting order**?

Business Value

- Finding the best batting order will improve the team's performance.

- By scoring more runs, the team is more likely to win more games.

- Increased team performance will lead to:
    1. Increased ticket sales.
    2. Increased merchandise sales.
    3. Increased publicity.



Methodology

1. **Gather** regular-season data for every at-bat since 1950.
2. **Engineer** *Players* to track career stats and Pitcher/Hitter interactions.

- Data was downloaded raw from *retrosheet.org* from 1950-2019.
- There ended up being 7.5 million at-bats in the Training Data and 1.6 million in both the Validation and Test Data.

Offensive Stats:

AB: At-Bats

K%: Strikeout Rate

OPS: On-Base plus Slugging

Defensive Stats:

IP: Innings Pitched

WHIP: Walks plus Hits per Inning Pitched

K/BB: Strikeout to Walk Ratio

Methodology

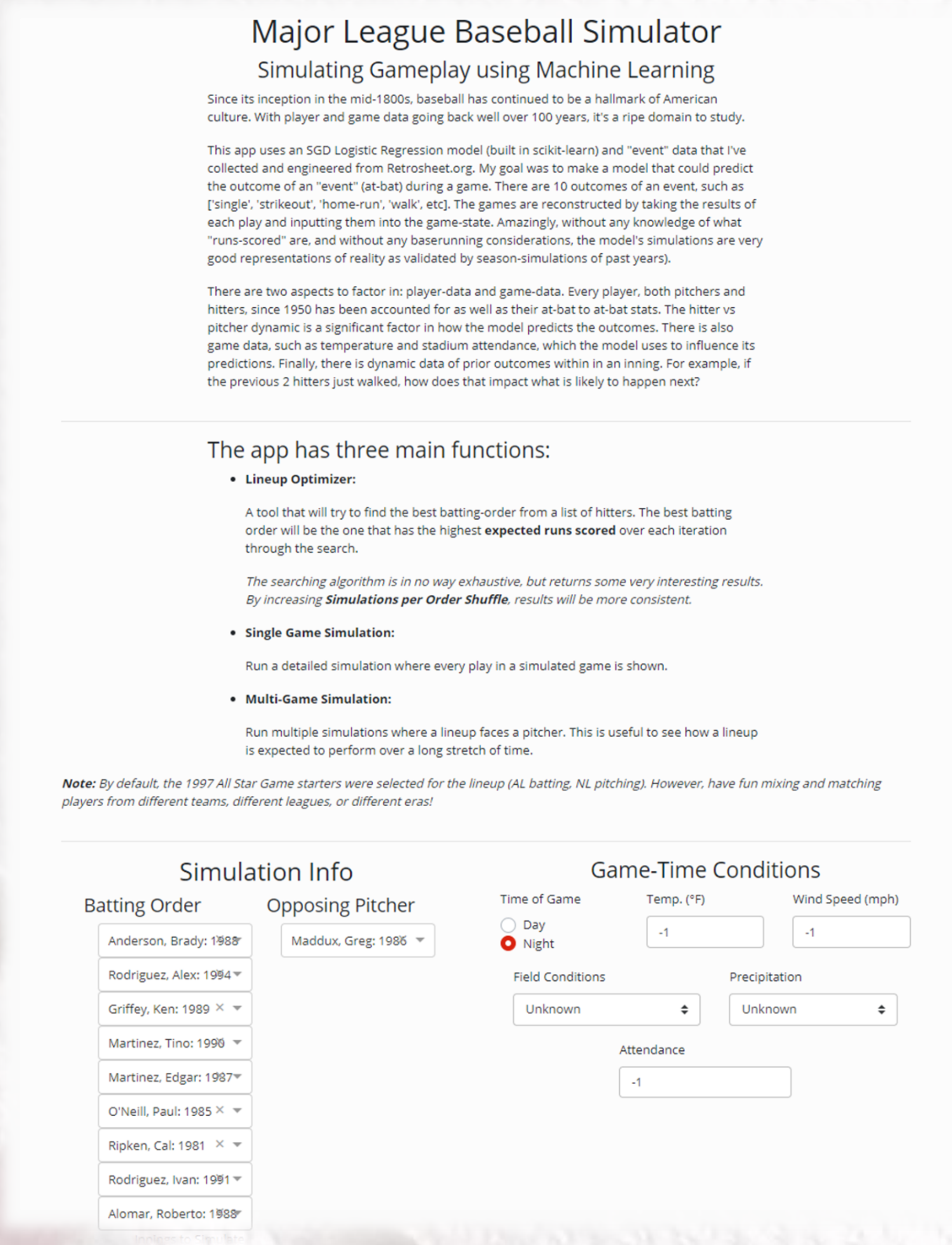
3. **Model** to predict the outcome of a given at-bat (probabilities).
4. Create and deploy ***Simulator*** to simulate games and optimize batting orders.

The Simulator is built using the model's real-time predictions.

The Simulator works by:

1. *Predicting the outcome of an at-bat.*
2. *Using whiffle-ball rules to determine how a given outcome impacts the game-state.*
3. *Updating the current game-state.*
4. *…and repeat!*

Baseball Simulator - App



Major League Baseball Simulator

Simulating Gameplay using Machine Learning

Since its inception in the mid-1800s, baseball has continued to be a hallmark of American culture. With player and game data going back well over 100 years, it's a ripe domain to study.

This app uses an SGD Logistic Regression model (built in scikit-learn) and "event" data that I've collected and engineered from Retrosheet.org. My goal was to make a model that could predict the outcome of an "event" (at-bat) during a game. There are 10 outcomes of an event, such as ['single', 'strikeout', 'home-run', 'walk', etc]. The games are reconstructed by taking the results of each play and inputting them into the game-state. Amazingly, without any knowledge of what "runs-scored" are, and without any baserunning considerations, the model's simulations are very good representations of reality as validated by season-simulations of past years).

There are two aspects to factor in: player-data and game-data. Every player, both pitchers and hitters, since 1950 has been accounted for as well as their at-bat to at-bat stats. The hitter vs pitcher dynamic is a significant factor in how the model predicts the outcomes. There is also game data, such as temperature and stadium attendance, which the model uses to influence its predictions. Finally, there is dynamic data of prior outcomes within in an inning. For example, if the previous 2 hitters just walked, how does that impact what is likely to happen next?

The app has three main functions:

- **Lineup Optimizer:**

  A tool that will try to find the best batting-order from a list of hitters. The best batting order will be the one that has the highest **expected runs scored** over each iteration through the search.

  *The searching algorithm is in no way exhaustive, but returns some very interesting results. By increasing **Simulations per Order Shuffle**, results will be more consistent.*

- **Single Game Simulation:**

  Run a detailed simulation where every play in a simulated game is shown.

- **Multi-Game Simulation:**

  Run multiple simulations where a lineup faces a pitcher. This is useful to see how a lineup is expected to perform over a long stretch of time.

Note: *By default, the 1997 All Star Game starters were selected for the lineup (AL batting, NL pitching). However, have fun mixing and matching players from different teams, different leagues, or different eras!*

Simulation Info

Batting Order

Anderson, Brady: 1988

Rodriguez, Alex: 1994

Griffey, Ken: 1989

Martinez, Tino: 1990

Martinez, Edgar: 1987

O'Neill, Paul: 1985

Ripken, Cal: 1981

Rodriguez, Ivan: 1991

Alomar, Roberto: 1988

Innings to Simulate

Opposing Pitcher

Maddux, Greg: 1986

Game-Time Conditions

Time of Game: ○ Day ● Night

Temp. (°F): -1

Wind Speed (mph): -1

Field Conditions: Unknown

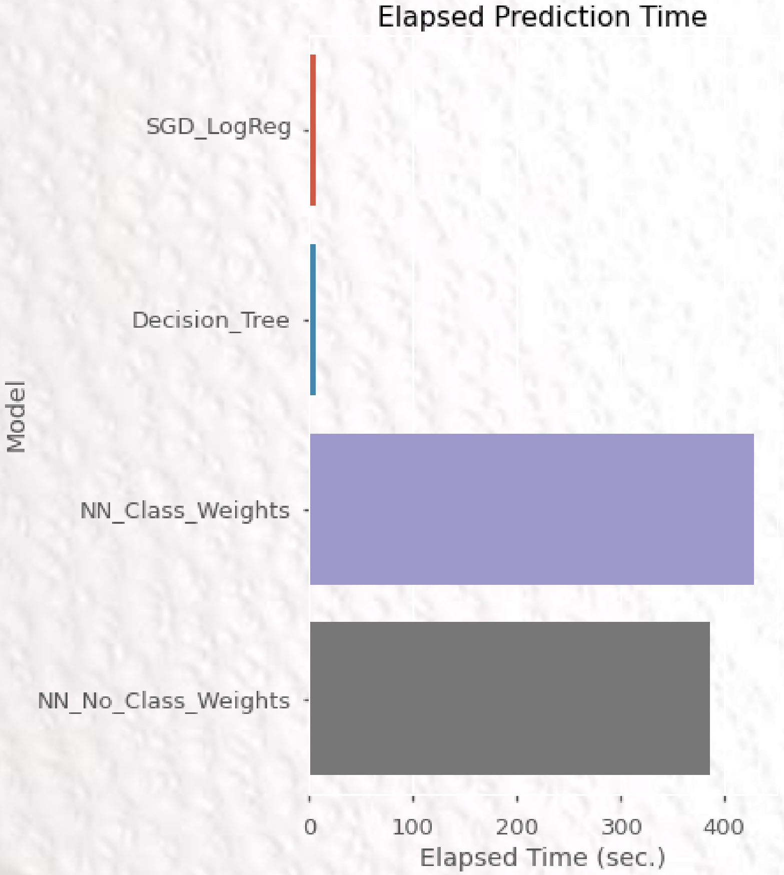
Precipitation: Unknown

Attendance: -1

Click HERE to use the app!

Modeling

- Model size and performance should be considered.
  - A stronger model with better predictions will be much bigger in size and slower to compute predictions and optimizations.



There is a massive difference in model performance. The fastest models were making predictions several hundred times faster than the slowest models.

Modeling Validation

Without having any prior knowledge about "runs scored", the models' predictions (compared to real past data) are quite good.

<u>Yankees:</u> Average Runs per Game

<u>Actual:</u>-----------*4.519*

<u>Simulated:</u>------*4.951*

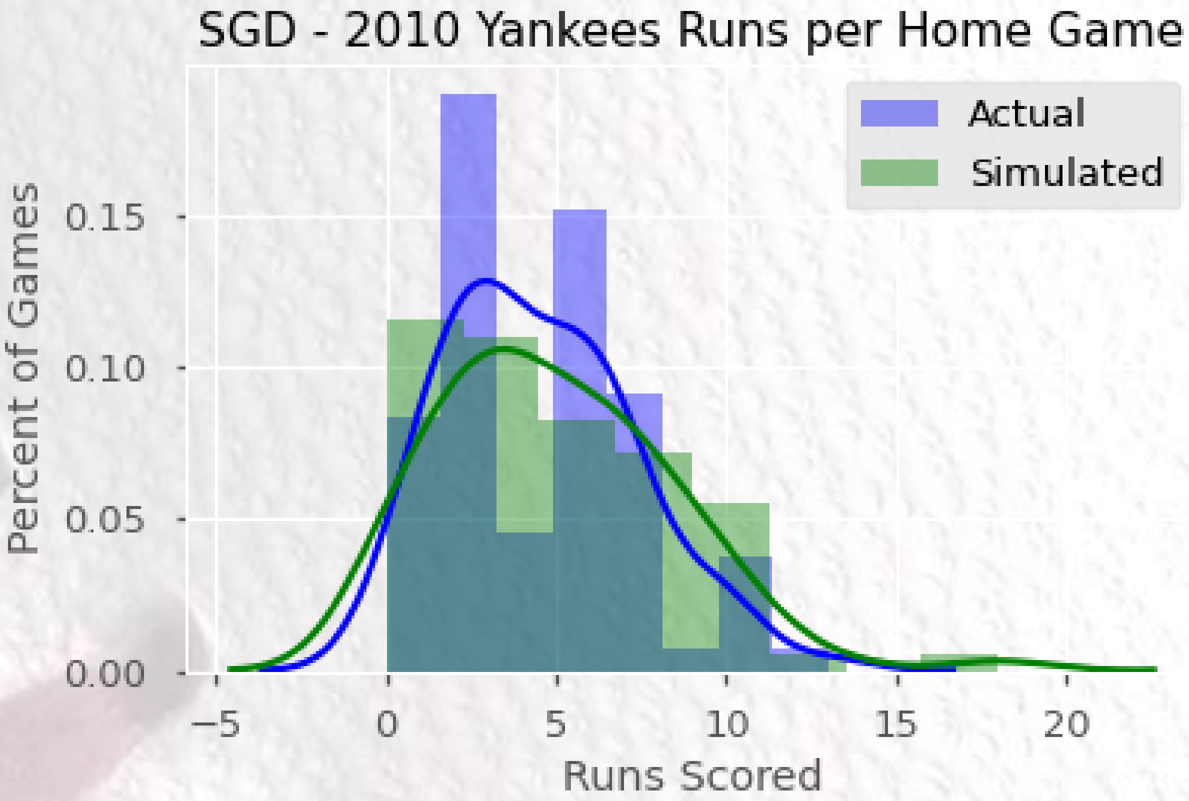
Mean Absolute Error:----*0.58 Runs/Game*

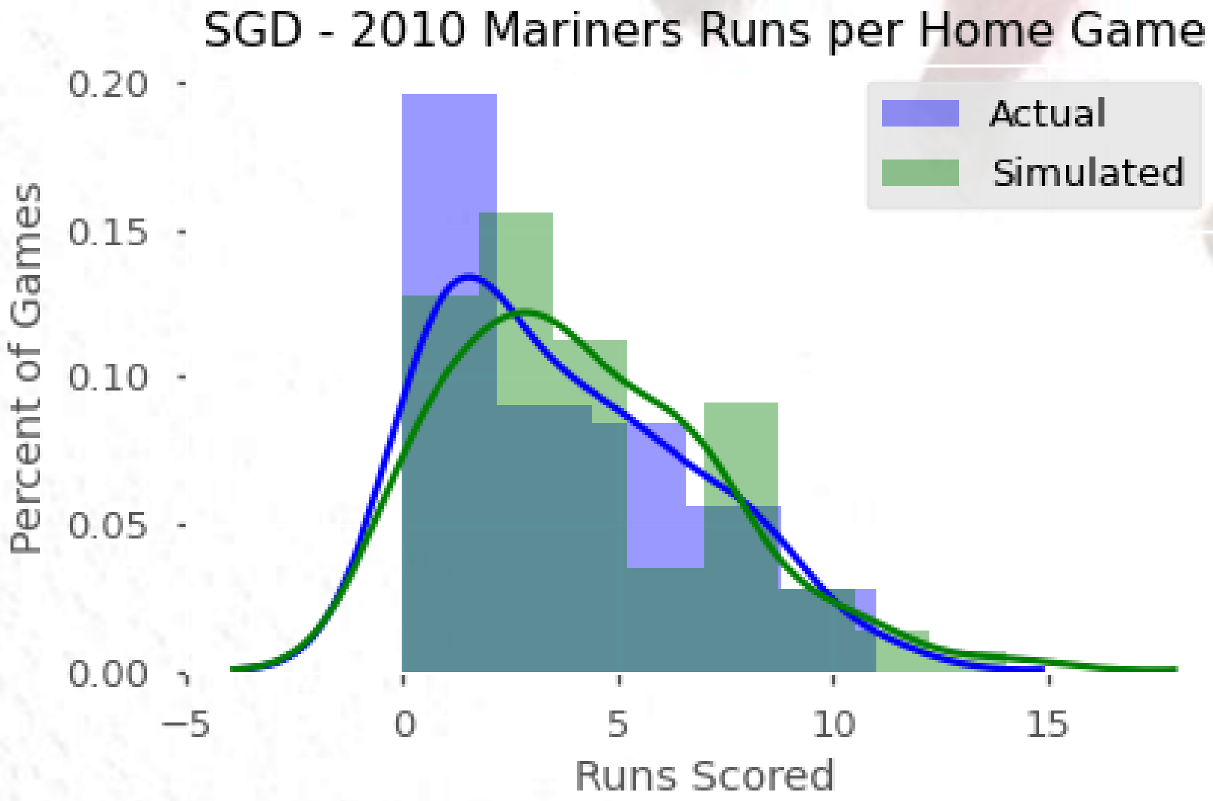
<u>Mariners:</u> Average Runs per Game

<u>Actual:</u>-----------*3.753*

<u>Simulated:</u>------*4.136*

Mean Absolute Error:---- *0.56 Runs/Game*



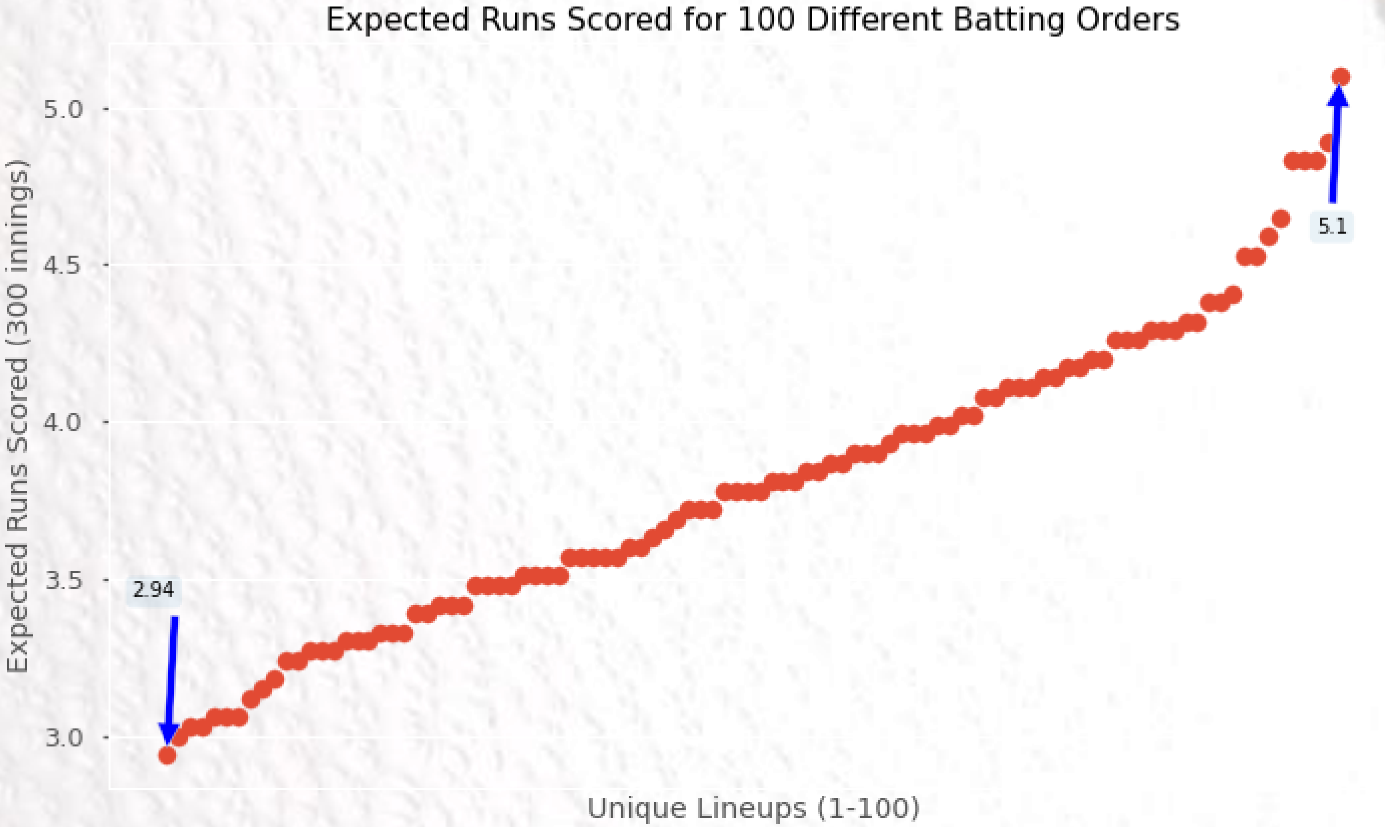


Baseball Recommendations

- **Batting Order Matters!**
  - When running simulations of different lineup-combinations against a given pitcher, the performance of these lineups were drastically different.

<u>Lowest Scoring Lineup:</u>

1. Curtis Granderson
2. Nick Johnson
3. Alex Rodriguez
4. Robinson Cano
5. Derek Jeter
6. Mark Teixeira
7. Jorge Posada
8. Nick Swisher
9. Brett Gardner

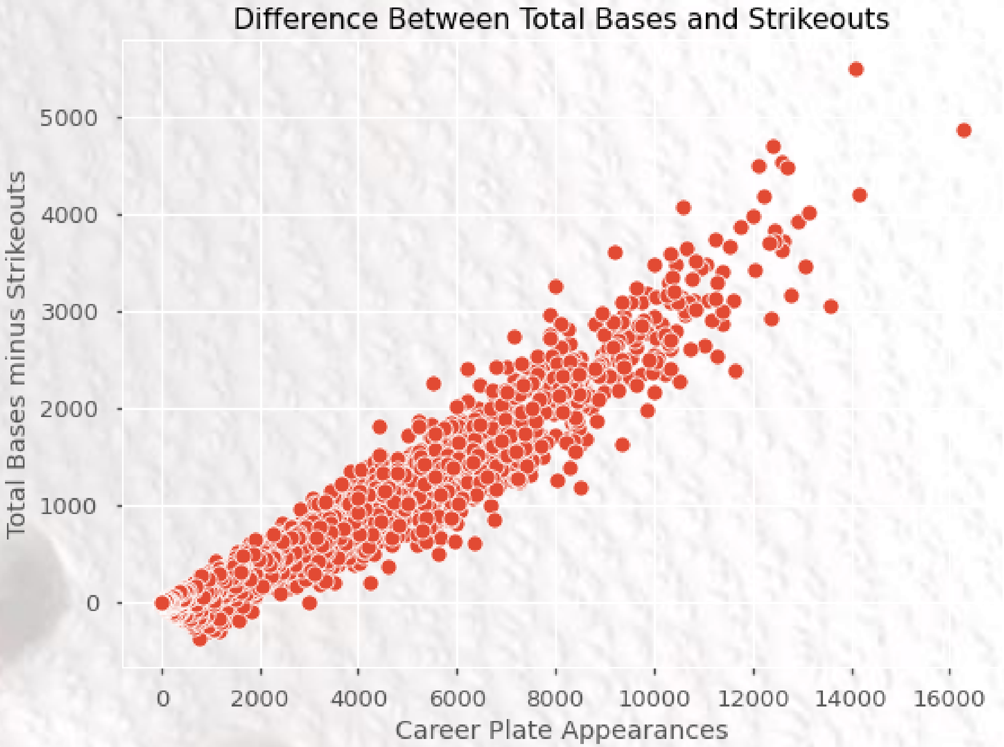


<u>Highest Scoring Lineup:</u>

1. Brett Gardner
2. Jorge Posada
3. Robinson Cano
4. Derek Jeter
5. Mark Teixeira
6. Curtis Granderson
7. Alex Rodriguez
8. Nick Johnson
9. Nick Swisher

Baseball Recommendations - Players

- When scouting, find young players with a big difference between *total-bases* and *strikeouts.*
  - There is a strong correlation between *number of career-at-bats* and *difference between total-bases and strikeouts*.



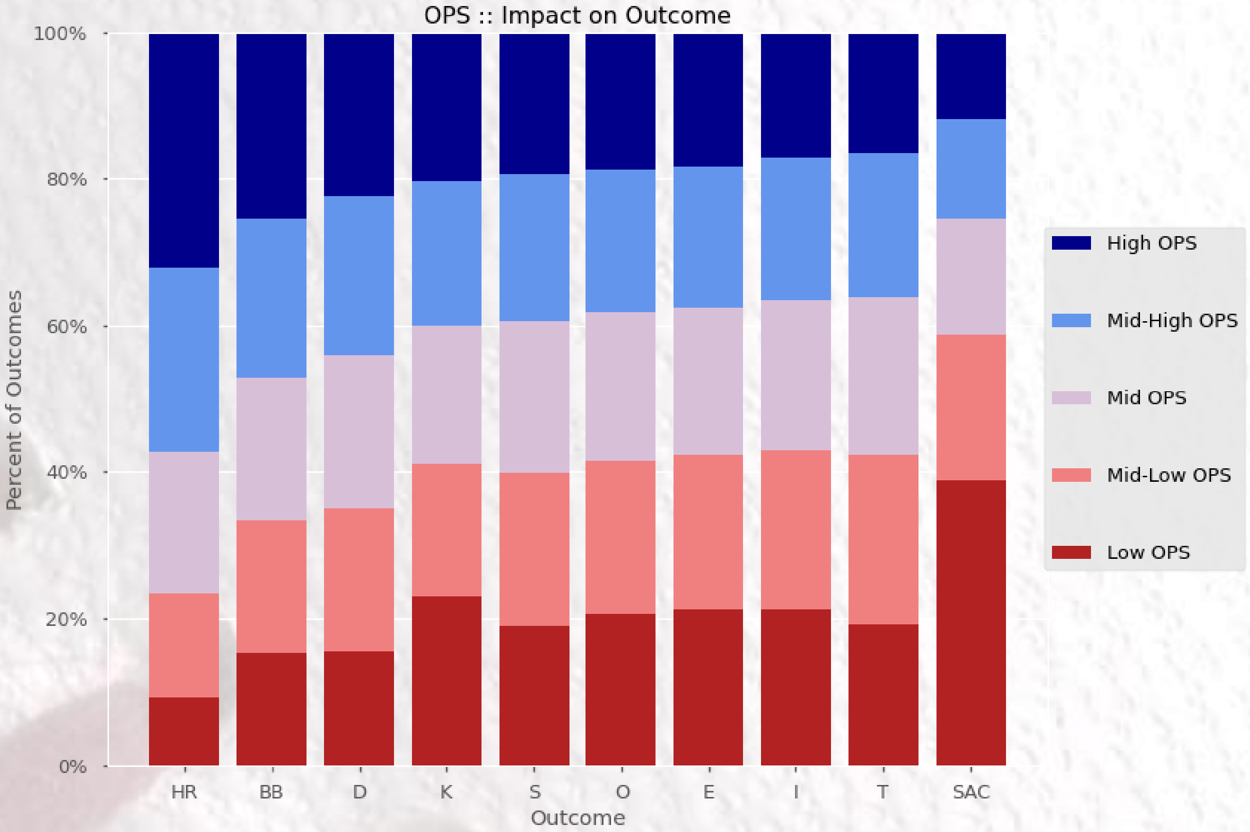
A hitter's ability to hit for a high total-bases and low strikeout total shows longevity in the league.

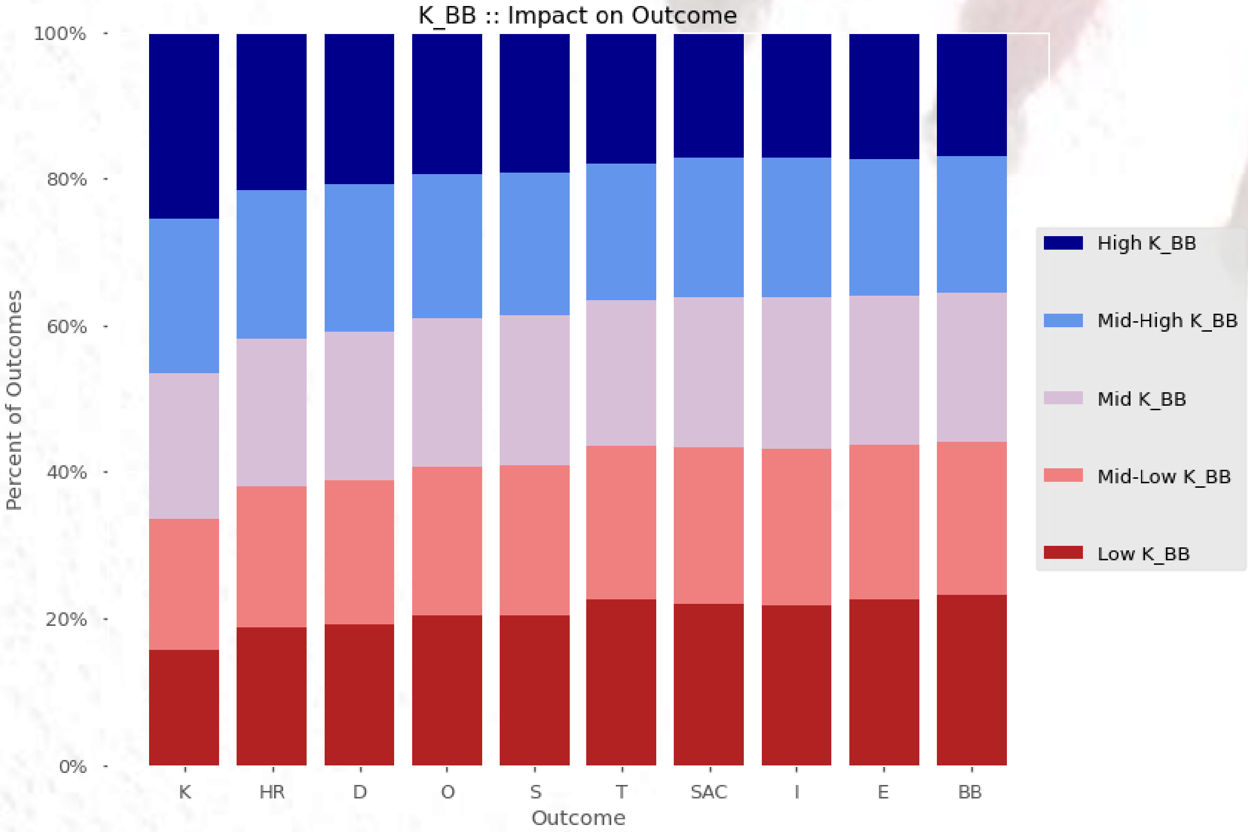
Player contracts can use this information to offer long-term deals for team-friendly money.

Baseball Recommendations - Outcomes

- For **acquiring players**, homeruns, walks, and doubles are achieved mostly by players with a high **OPS.**

- While **strikeouts** are most common outcome when pitchers have a high *strikeout-to-walk-ratio* (which is good for the pitcher)…
- **Homeruns** and **Doubles** are the second and third most common (which are very good for the hitters)!





Conclusion

- The Simulations showed conclusively that changing a batting order will change the expected results.
  - There was a huge shift in performance between 100 randomly-shuffled lineups.
  - Potentially, a team can improve its performance by several runs-per-game by simply changing the batting order.

- Using the **App**, different lineups can be compared.
  - Or, using the Optimizer, an optimized version of the lineup can be returned.

Future Work / Next Steps

- Experiment with different modeling architectures.
  - *E.g.: Looking at at-bats and the flow of the game as a time-series and using SARIMA or RNN models.*

- Engineer more features:
  - More stats for players.
  - *Hitter-vs.-Pitcher* interaction stats.

- Collect data and do research on the minor league system to acquire high-quality prospects to fit the team's roster.

Thank you!

Questions?

- Data:
  - Retrosheet.org

- Flatiron School – Data Science Bootcamp
  - Instructor: James Irving