15TH ANNUAL
MIT SLOAN
SPORTS ANALYTICS CONFERENCE
APRIL 8-9, 2021 | VIRTUAL
presented by ESPN 42 ANALYTICS

Baseball Predictions and Strategies Using Explainable AI

Joshua Silver (Singlearity, joshua.silver@singlearity.com)
Tate Huffman (Harvard University, thuffman@college.harvard.edu)

1. Abstract

Over the last decade, Major League Baseball has dramatically increased the amount of data it captures and makes available to the public. Meanwhile, there is a growing technology trend that applies AI to analyze massive amounts of data to build complex models that find relationships and meaning in the data. By marrying these two concepts together, we built a neural-network-based AI model called Singlearity-PA (pronounced single-arity-P-A) to solve one of the most fundamental questions in baseball: *How can we predict the outcome of a batter vs. pitcher plate appearance (PA)?*

We show how our model learned even the most subtle rules and strategies of the game, and is able to answer our batter vs. pitcher question with accurate and precise predictions. We demonstrate how to apply techniques to interpret and visualize Singlearity-PA's predictions to make the model's rationale understandable, and sharable for humans.

Finally, we provide open-source tools that allow for the readers' own experimentation and analyses.

2. Introduction and Motivation

Recently, new technologies and databases have rapidly increased the amount of available data in baseball. We now have many new statistics at our disposal. For instance, a single pitch at the major league level contains measurements for approximately 90 different parameters, with such varied data points as pitch information (including velocity, spin rates, and ball movement), batted-ball information (including exit velocity and launch angle) and fielder-position information. Baseball domain experts wishing to improve their abilities to forecast outcomes have pored over this data, looking to answer questions such as "How does a pitched ball spin rate affect pitcher performance [1]?", or "How can a batter's exit velocity predict the batter's future offensive output [2]?"

Now, there's a better way to put the data to work. In this paper, we show a methodology for effectively incorporating large amounts of data such that it yields improvements in accuracy compared to previous, more simplistic approaches. We demonstrate our model by predicting the outcome of a batter vs. pitcher matchup (although our methodology is generalizable and scalable for other types of predictions). We focus on this relationship as it drives a large portion of baseball strategy, affecting decisions on batting orders, pinch hitters, relief pitchers, base running, sacrifice bunting, intentional walks and much more. We also show how to use a technique called SHAP (Shapley Additive exPlanations) to produce easy-to-comprehend explanations and visualizations for how and why our model produced its results.

Armed with an accurate and precise model that understands the strategies and structure of the game, we show how we can more precisely model additional baseball outcomes and strategies that previously have relied on simplistic assumptions. To demonstrate one such improvement, we've built a Markov chain model that can better predict the effects of lineups on scoring probabilities compared to previous methods like RE24 that make batter and pitcher agnostic predictions.

3. Previous Techniques

One of the most popular models for head-to-head predictions is log5. log5, created by sabermetrics pioneer Bill James, uses a formula that incorporates the winning percentage of two competing teams to predict the winner of the matchup. James later extended log5 to predict a hit vs. no-hit result between a batter and a pitcher. Matt Haechrel then generalized James' work to predict the probability of seven types of PA outcomes for each PA. Haechrel's equation uses 365-day average rate-per-plate-appearance of each of seven possible outcomes: out, single, double, triple, home run, walk, and hit-by-pitch [3]. Haechrel's equation is:

$$P_{BP} = \frac{\frac{P_B * P_P}{P_L}}{\frac{(P_B * P_P)}{P_L} + \frac{(1-P_B)*(1-P_P)}{1-P_L}} \quad (1)$$

where:

P_{BP} = Probability of an event for a specific batter vs. pitcher PA

P_B = Probability of an event for the specific batter

P_P = Probability of an event for the specific pitcher

P_L = League average probability of an event

Haechrel's version of log5 suffers from the following limitations:

- **It performs accurately only on matchups between every day players.** Haechrel only showed results for batters and pitchers who had at least 502 plate appearances in the previous year. This limited its applicability since most PAs from 2011–2020 (82.1%) featured matchups where at least one of the players did not have at least 502 PAs. The current trends toward shorter outings for starting pitchers, and increased usage of role players mean that more PAs will feature players with reduced historical data.
- **It ignores many important factors shown to be critical in affecting outcomes,** including:
    - Left-handed vs. right-handed pitcher and batter matchups,(i.e., platoon effects)
    - Ballpark factors (e.g. in the last five years, Coors Field, home to the Colorado Rockies, has seen 40% more hits (on average) than the New York Mets' Citi Field in the same period, even after adjusting for different teams and players.)[1]
    - Weather factors including temperature, wind direction, and humidity

[1] See http://www.espn.com/mlb/stats/parkfactor

- Game context, such as number of outs and runners-on-base
- **Its predictions are not precise** (e.g. it can predict an out, but does not distinguish between a strikeout, a double play, or a sacrifice bunt).

4. Singlearity-PA Technique

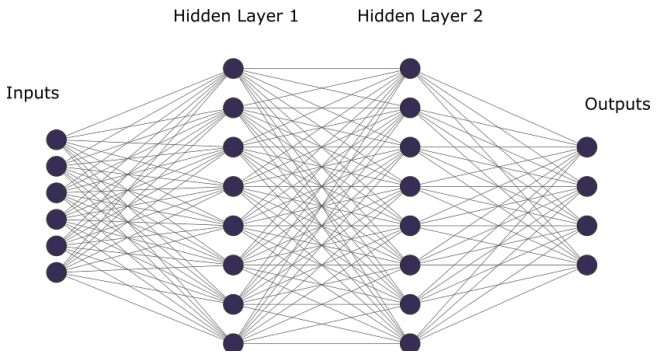
To address the limitations of log5, and to discover and reason about the complex relationship between the different variables that affect plate appearances, we turn to artificial neural networks.

4.1 Neural Network Introduction

An artificial neural network is a modelling technique that builds an internal representation that mimics the computation done in the human brain. Neural networks have been successfully used in a variety of applications, including image recognition, automated game playing, and natural language processing. The goal of a neural network is to predict output values based on input values. As a simple example, the input values of a consumer's income, marital status, and outstanding loans could produce the output predicting the probability that the consumer will default on a loan. A neural network trained on large amounts of data can build an internal representation that can predict default rates of new customers. Unlike more simplistic models like linear regression, neural networks can discover nuanced and complex relationships.

In our model, our inputs/features are the *a priori* statistics on a particular batter vs. pitcher plate appearance . We use 87 floating point inputs (details below). We have 21 outputs, where each output represents the probability of a distinct PA outcome as defined by MLB[2] (e.g., single, home run, fielder's choice, error, and hit-by-pitch, to name just a few).

Figure 1: Typical Architecture of a Neural Network



4.2 Feature Selection

Feature selection is the process of selecting the inputs to the model. We chose our input features by reviewing baseball research, and by trial and error. We then grouped our inputs as shown in Table 1 into commonly used baseball categories. Details on input features can be found in the Appendix in Table 5.

[2] See https://baseballsavant.mlb.com/statcast_search

Table 1: Inputs used by Singlearity-PA

Feature Group	# of Inputs	Details
Batter Historical	14	Batter last 365 days stats
Pitcher Historical	14	Pitcher last 365 days stats
Batter Recent	9	Batter last 21 days stats
Pitcher Recent	9	Pitcher last 21 days stats
Batter/Pitcher head-to-head	8	Batter-pitcher 3-year head-to-head stats
Park Factor	4	Offensive stats for games at this venue
Base State	4	Outs and occupied bases
Batter Position in Field	10	One-hot encoded value of fielding position of the batter (P, 1B, …, DH)
Platoon Statistics	5	Batter's and pitcher's 3-year platoon statistics
Exit Velocity	2	Batter's 3-year max and average exit velocity
Inning	1	Inning
Home or Away	1	Home or Away
Net Score	1	Batting score – Fielding score
Pitch Count	1	Pitcher's pitch count at the start of the PA
Weather	1	Game time temperature at the start of the game
Batter GB/FB Ratio	1	Batter's 3-year ratio of ground balls to fly balls
Pitcher GB/FB Ratio	1	Pitcher's 3-year ratio of ground balls to fly balls
Game Year	1	Game year
Game Day	1	Day of the season
Total	**87**	

Note that in this approach, we can incrementally decide to add additional features, such as minor league statistics, humidity and wind direction, fielding quality, or batter speed. Note also that an input such as "Batter Position in Field" could potentially be used by a well-trained network as a proxy for the batter's speed, and, in fact, results indicate Singlearity-PA used this input to determine the probability that the batter's plate appearance would result in a double play or a triple (see Figures 9 and 10).

4.3 Training

*Supervised Learning i*s the process of teaching the neural network to predict outcomes based on known "correct answers." We taught our network using Statcast's pitch-by-pitch data from regular season games from 2011-2020. This consisted of 1.72 million plate appearances played in 22,740 games over 1,694 different dates. Using best practices, we split the 1,694 calendar days so that 60% of the dates were used for training data, 20% for validation data, and 20% for test data. The architecture of the network was adjusted and finalized by analyzing the network's performance on the validation data, and then measured by using the test data.[3]

5. Results

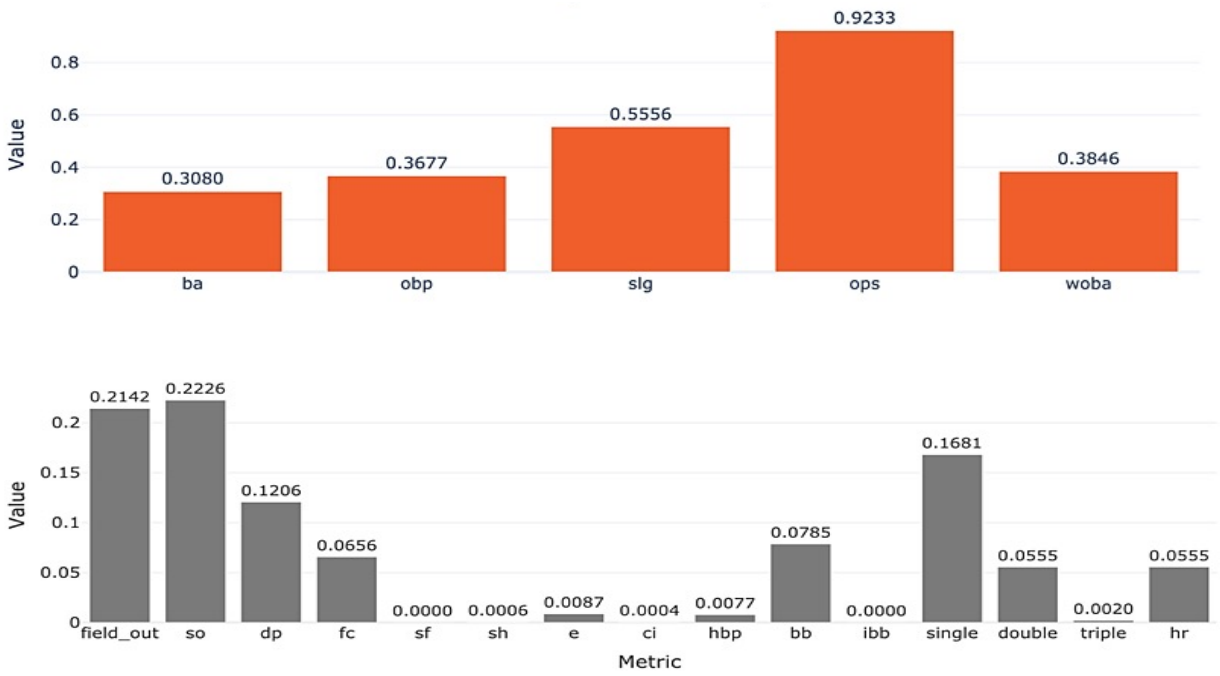
We begin to review results with a simple subjective review– a gut check– of Singlearity-PA's predictions. Figure 2 shows the expected results when the Yankee's Aaron Judge faced Houston's

[3] We used categorical cross-entropy as our loss function and used Tensorflow for our machine learning platform. We used 2 hidden layers of 80 nodes each, and validated that the model was not overfitting the training data by observing and plotting the training loss vs. the validation loss.

Wade Miley with the game tied 2-2 in the 6th inning on a 79°(F) evening on June 22, 2019 at Yankee Stadium. Wade Miley had thrown 75 pitches at the beginning of Judge's plate appearance. Judge's aggregated statistics seem reasonable, but it is important to note that our model has also predicted probabilities for outcomes such as double play and fielder's choice. It has learned when a double play or fielder's choice is likely to occur despite having never been taught the rules!

Note: You can create your own matchups and view the predicted outcomes using an online version of Singlelarity-PA.

Figure 2: Singlearity-PA predicted outputs for Aaron Judge vs. Wade Miley



We now do an evaluation in which we compare Singlearity-PA's predictions to those of log5. We also compare Singlearity-PA to another extremely simple model (LeagueAverage) where we assume probabilities for PA outcomes correspond to the league's average expected outcomes for the year. To determine whether Singlearity-PA's advantage was due to game context information (outs, baserunners, pitch count, inning), we created a new model, Singlearity-NoState, in which this context information was excluded from the inputs. For an aggregated statistic (like wOBA), we allowed the models to make the prediction of the precise PA outcome and then calculated the aggregated statistic for that plate appearance. We use RMSE to measure the quality of the different predictor's weighted on-base average (wOBA) stats and Categorical X-Entropy loss to evaluate the quality of HR and BB predictions.

Because log5 was designed only to work for plate appearances in which there were substantial statistical history for both the batter and the pitcher, we also examined how well different models performed when broken down for plate appearances with varied amounts of historical data available. We slotted each plate appearance into one of the following categories:

- **Everyday players:** Both batter and pitcher had >=502 PAs in the last 365 days
- **Infrequent players:** Either the batter or pitcher had <100 PAs in the last 365 days
- **Occasional players:** All other PAs

Table 2: Singlearity-PA error rate in predictive accuracy for PAs involving players with varying historical statistics (lower is better)

Type of PA Matchup	Model Name	wOBA	HR	BB
All	**Singlearity-PA**	**0.5091**	**0.05439**	**0.1149**
	Singlearity-NoState	0.5096	0.05507	0.1156
	log5	0.5151	0.05867	0.1208
	LeagueAverage	0.5111	0.05627	0.1184
"Everyday Players" (18% of all PAs)	**Singlearity-PA**	**0.5269**	**0.06100**	**0.1135**
	Singlearity-NoState	0.5279	0.06189	0.1138
	log5	0.5287	0.06240	0.1139
	LeagueAverage	0.5280	0.06250	0.1170
"Occasional Players" (61% of all PAs)	**Singlearity-PA**	**0.5098**	**0.05440**	**0.1155**
	Singlearity-NoState	0.5098	0.05490	0.1168
	log5	0.5120	0.05638	0.1163
	LeagueAverage	0.5111	0.05990	0.1185
"Infrequent Players" (21% of all PAs)	**Singlearity-PA**	**0.4925**	**0.04911**	**0.1145**
	Singlearity-NoState	0.4933	0.04970	0.1138
	log5	0.5120	0.06176	0.1383
	LeagueAverage	0.4968	0.05180	0.1192

We observe from Table 2 that Singlearity-PA performs better than both log5 and LeagueAverage for all types of predictions (wOBA, HR, BB) and all different types of plate appearances (everyday players, occasional players and infrequent players). Singlearity-NoState, while less accurate than Singlearity-PA, still generates significantly better predictive data than log5 and LeagueAverage for wOBA, and HR, but has lost some of its advantage over log5 in predicting BBs for all players except for infrequent players.

6. Explaining the Results using Shapley Values

Now that we've developed an accurate model, we're challenged with making the rationale for its predictions understandable to human beings. This process is referred to as making the model *explainable.* Only in this way can team managers and baseball analysts have the confidence to explain and defend their decisions to management, fans, players and the press. Explainability provides:

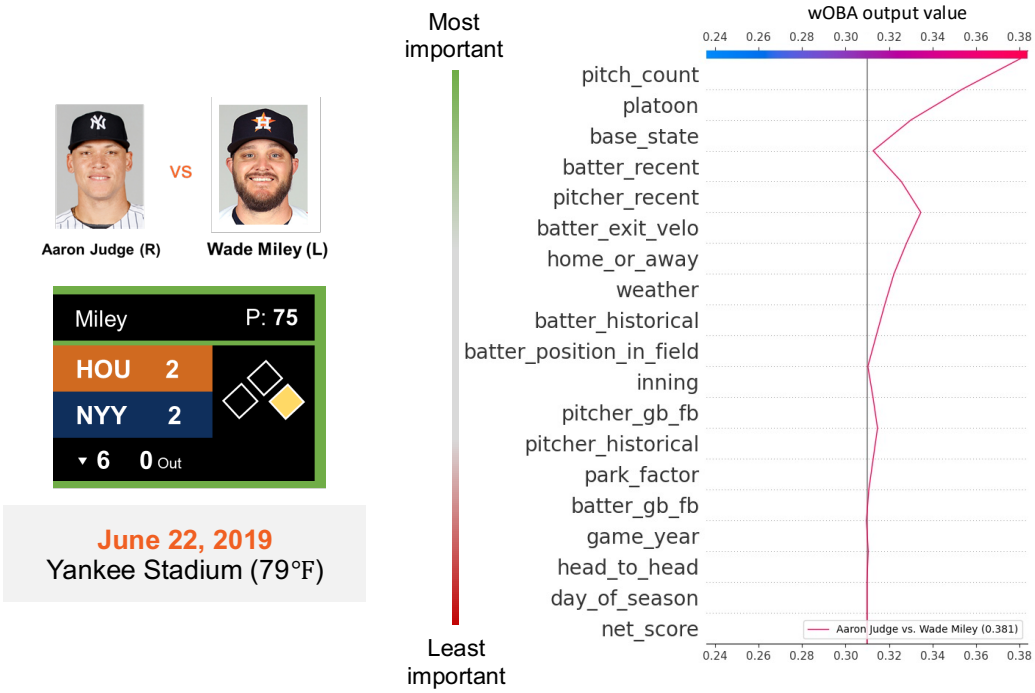
- **Trust** Ultimately, it is humans that make decisions based on the predictions of the model. These people need to understand the model's decision process in order to believe its outcomes.
- **Insight and Learning** Humans often have an intuition or rules-of-thumb about how things work. An explainable model can help us update our mental model.
- **Ability to debug** Occasionally, a model may produce a result that is clearly wrong. By being able to explain why the model predicted a result, we can understand and evaluate improvements or potential additions to our input features.

To make our model explainable, we use the SHAP method and Shapley values. Shapley values have a deep history in economics and game-theory, but have only rarely been applied to baseball[4]. Lloyd Shapley invented the concept in 1953, which led to a Nobel Prize in Economics in 1992. Shapley values were later applied to complex AI models to make them explainable[5]. The method gives us a mechanism to provide a set of simple metrics and visualizations. These show how each input feature contributed to the model's prediction, making it easy for non-AI experts to understand the basis for the predictions. The importance of an input feature may not be uniform across all data points, so Shapley values provide for both local feature importance (a ranking of the features that affect a single prediction) and global feature importance (a single ranking of how important the input feature is to all of the predictions).

6.1 Local Feature Importance

When making a single prediction, Shapley values explain why the prediction veered from an average predicted value. Let's look again at Aaron Judge's plate appearance against Wade Miley in Figure 3.

Figure 3: Shapley decision plot for predicted wOBA of Aaron Judge vs. Wade Miley



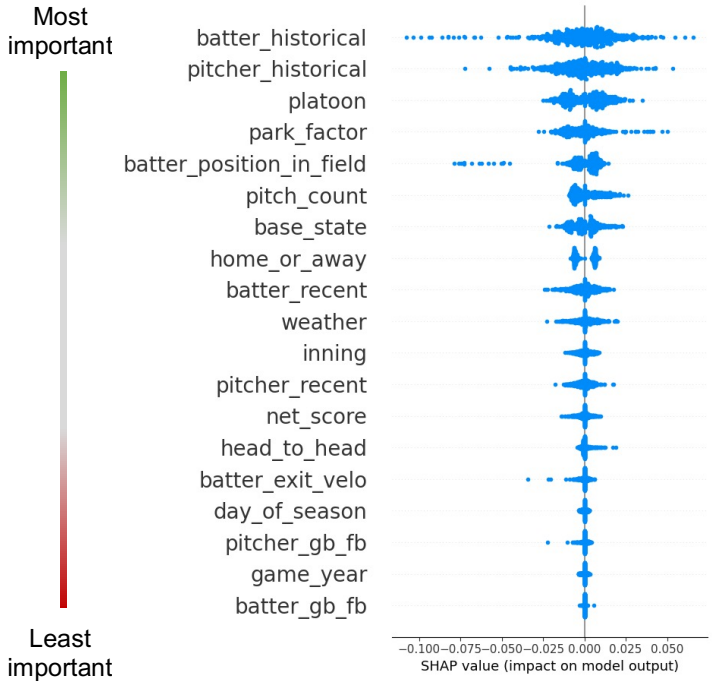
Reading the graph from the bottom up, we see how Aaron Judge's wOBA varies from the league average wOBA of just over 0.310. As we read up the graph, we see which groups of features had the largest effect on Judge's predicted wOBA. Let's examine some interesting features (from bottom to top):

- **Pitcher_historical**: Note that Wade Miley's historical stats do not cause a big change to the predicted wOBA of this PA. This doesn't mean that his stats are unimportant but that Miley's historical stats are fairly typical.
- **Batter_exit_velo:** Judge has historically hit the ball very hard. This causes Singlearity-PA to increase his projected wOBA relative to someone with similar offensive stats.
- **Batter_recent:** This is Judge's second game after returning from a 2-month stint on the IL. The fact that he has few recent PAs drives his predicted wOBA down.
- **Platoon:** Judge gets a big boost for the combination of the righty/lefty matchup, together with both Judge's and Miley's stats against opposite-handed opponents.

6.2 Global Feature Importance

Global feature importance refers to how important (on average) a feature is to make an accurate prediction. One way to visualize feature importance is to plot the Shapley values for a sample of predictions. Figure 4 shows the most important features impacting wOBA. From Figure 4, we can observe that, as expected, batter and pitcher historical stats have the largest impact on wOBA, followed by platoon effects and park factors. We also see that the "batter position in the field" row is important. The dots on the far left of that row represent players, likely pitchers, where the model decreases their expected wOBA based on its knowledge that pitchers are poor hitters. Figures 5-10 in the appendix show graphs of global feature importance for HR, SO, BB, HBP, DP. And triples.

Figure 4: Shapley global feature importance for wOBA prediction



7. Integrating Singlearity-PA with Markov Chains

Using Singlearity-PA's ability to generate precise outcomes, we now have the ability to predict more complex outcomes than that of a single plate appearance. For example, we can now predict results of a half-inning, allowing us to optimize lineups, or to choose a situation-appropriate relief pitcher, among other important decisions.

7.1 Previous work using RE24

Previous work for simulating runs scored in half-innings has focused on RE24, a methodology which uses historical statistics to calculate run expectancy from the start of a plate appearance to the end of a half-inning. For example, from 2010-2015, on average 0.481 runs scored with the bases empty and no one out, while 1.130 runs scored on average with 1 out and runners on first and third[6]. RE24 contains scoring expectancy for each of the 24 states (3 outs x 2^3 baserunners). However, RE24 does not take into account the batter, pitcher, lineup or other variables that can impact the expected offensive output - RE24 makes the same run expectancy prediction regardless of whether a pitcher is facing Murderer's Row or a lineup of "has-beens". There have been attempts to improve RE24 by introducing park factors[7] or current batter[8], but there has not been a holistic effort.

7.2 Markov Chains

To build a better run prediction, we coupled Singlearity-PA with Markov chains (we refer to the combined method as Singlearity-Markov). Markov chains are a method of constructing multi-state sequences using the transitions between unique states. In order to implement a Markov chain, we need to calculate a transition matrix that gives the transition probabilities from any state to every other state.

Extending this into baseball, we can think of state *i* as the starting base-out state before the PA, state *j* as the base-out state afterwards, and the probabilities in the transition matrix as the probabilities of transitioning from *i* to *j* for all *i* and *j* [9]. For instance, to compute the probability of going from a state with 0 outs and a runner on first base to 2 outs and no runners on base, we would need to know the probability of a double play for that PA.[4] We rely on Singlearity-PA's prediction to create these transition matrices. Because the outputs of plate appearances do not completely describe the movement of baserunners, we make some simplifying assumptions about how baserunners advance. For instance, we use historical stats to predict that a runner on 1B will go to 2B or 3B either 70% or 30% of time, respectively, on a single. A complete list of assumptions can be found in our Markov chain open source code.

By building up the transition matrix for each player in the starting lineup, we can calculate the expected run distribution for the entire half-inning. We rely on the algorithm described by Pankin [10] and Bukiet, et al. [11] to fully implement the Markov chain.

[4] Singlearity-PA makes probability predictions for each of the different types of double plays such as grounded_into_double_play, sac_fly_double_play, and strikeout_double_play. These are all included when calculating the transition matrix.

While Markov chains have previously been explored in baseball, they were of limited practical use due to the lack of "data... available to set up a transition matrix for all possible occasions [11]." By incorporating Singlearity-PA, we hope to alleviate some of these problems. Not only can we use as inputs a richer set of variables beyond specific batter, pitcher, and state information (themselves already an improvement on the default league-average probabilities), but we can go beyond standard hits and walks that typically populate these matrices, bringing events like sacrifice flies and errors into possible outcomes.

7.3. Methodology

We evaluate our approach by measuring our ability to predict the runs scored in the first inning of regular season games from 2017-2020. We use the first inning for these calculations because it avoids the complications of pinch-hitters, relief pitchers, or shortened games that may arise when using later innings. As a baseline to compare our results to, we use the RE24 value from that same state in all first innings of the prior season. As an example, the first inning RE24 matrix for 2019 can be viewed in the appendix in Table 6.

7.4. Experimental Results

Our first attempt at measuring Singlearity-Markov's ability to predict run expectation revealed that our predictions were consistently undershooting actual runs by 12-15%. This is consistent with Bukiet et al. who found that their predictions were 7% below actual runs scored [11]. Looking anecdotally at typical innings, we believe this underprediction comes in two primary forms:

1. **Baserunner advancement:** Singlearity-Markov does not account for some events which tend to favor the offense. This includes stolen bases, wild pitches, and advancing to second on a play at home.
2. **Multiple events:** We can only predict one outcome per plate appearance, so whereas real-life averages can incorporate these occurrences (e.g., a double and a throwing error resulting in the batter reaching third), we can not.

To counter this systemic underprediction, we scaled up Singlearity-Markov's predictions based on its undershoot in the previous year. For instance in 2016, actual run production was on average 15.3% higher than Singlearity-Markov. Therefore, in 2017, we increased each prediction from Singlearity-Markov by 15.3% to create the run expectancy in Singlearity-MarkovAdjusted.

Calculating the RMSE of our predictions vs. actual runs scored for each of the last five MLB seasons, we get the following results (the lower, the better):

Table 3: Singlearity vs. RE24 RMSE error rate (lower is better)

Year	RE24	Singlearity-Markov	Singlearity-MarkovAdjusted
2017	1.01307	1.01735	1.01199
2018	0.95021	0.94880	0.94593
2019	1.00832	1.01322	1.00684
2020	0.92545	0.92456	0.92357
Overall	**0.98412**	**0.98644**	**0.98192**

We observe that using RE24 from the previous season is superior to Singlearity-Markov without the adjustment to output. Once we adjust our output, Singlearity-MarkovAdjusted outperforms RE24. This indicates that we have a methodology capable of effectively accounting for batters and pitchers, as well as factors like park factors and weather, to improve upon RE24.

7.5 In-Game Example

One straightforward application of Singlearity-Markov is lineup optimization. In the sabermetric bible *The Book*, a perfect batting order may be worth up to 50 runs per season [12], and while the actual effect may not be quite that high, even half that could lead to an extra two or three wins each season. Using Singlearity-MarkovAdjusted, we are able to iterate over all 9-factorial permutations of a lineup to compute expected runs scored for the first inning in just over an hour on a modern lap top.

Let's use the example of the National League in the 2019 All-Star Game. Singlearity-PAs wOBA predictions for the NL starters vs. AL starting pitcher Justin Verlander are shown in Figure 11 in the appendix. NL manager Dave Roberts chose a lineup which projected for 0.51101 runs in the first inning according to Singlearity-MarkovAdjusted. However, when iterating over all the permutations of this lineup, we find an optimal lineup:

Table 4: 2019 NL All-Star lineup

Order	Actual Lineup	Optimal Lineup
1	Christian Yelich	Ronald Acuña Jr.
2	Javier Báez	Christian Yelich
3	Freddie Freeman	Cody Bellinger
4	Cody Bellinger	Josh Bell
5	Nolan Arenado	Nolan Arenado
6	Josh Bell	Ketel Marte
7	Willson Contreras	Freddie Freeman
8	Ketel Marte	Willson Contreras
9	Ronald Acuña Jr.	Javier Báez
Expected 1st inning runs	**0.51101**	**0.54493**

This lineup projects for 0.54493 runs in the inning, an increase of 6.7%. Some of the bigger changes in this lineup make intuitive sense: Ronald Acuña Jr., superstar in the making, leading off; Josh Bell batting cleanup after a first half with 27 home runs; and Javier Báez ninth with a .324 OBP and 29% strikeout rate to that point in the season.

8. Open Source Code and Demos

We've provided a set of open source software (under the MIT license) as well as additional tools and demos to aid in exploring, validating, and building upon our results. This includes:

- Extensive open source code in R with the following functionality:
    - Markov chain library for run expectations. This code is capable of using either Singlearity-PA predictions or other user-supplied PA predictions.
    - Command line tools and libraries for optimizing a batting order based on PA predictions and Markov chains.
    - Source code for replicating our Markov chain results including fetching historical lineups, generating run expectancy predictions, and calculating the accuracy of the results.
- Graphical online demo allowing you to access Singlearity-PA's predictions for plate appearances.
- Open source libraries in R and Python for programmatically accessing Singlearity-PA's predictions. (an API key is required).

9. Future Work

There are several areas for improvement and ongoing work:

- Singlearity-PA is still missing some input features that would undoubtedly yield additional accuracy predictions. This includes:
    - Minor league or international league stats: This would yield better predictions on players who are recent arrivals to the MLB.
    - Fielding quality: This would allow for more accurate PA predictions and allow building more strategies around defensive substitutions.
- There are also opportunities to leverage Singlearity-PA's predictions to do more than what was done for our first inning Markov.
    - Singlearity and our Markov model only accounted for the scorekeeping events associated with a batter's plate appearance. This meant that our Markov model was missing significant portions of offense such as wild pitches and stolen bases. We could build additional transitions into our Markov chain based on the players and the environment and thus have a more complete picture of a game.
    - We did not attempt to model how a game would evolve beyond the half-inning. If we could create the ability to incorporate bench players and likely substitutions, we could better model entire games, and eventually build out more complete strategies for player substitutions and ideal roster composition.

10. Conclusion

AI-based computing models have shown the ability to utilize massive amounts of data to improve our understanding and decision-making abilities. As the amount of readily available baseball data has skyrocketed, we introduce Singlearity-PA. This neural-network can successfully predict the outcome of plate appearances by using the data in a holistic manner. It can also serve as the building block to more complicated strategies, like those using Markov chains to build optimal lineups or to decide on pinch hitters and relief pitchers. Singlearity-PA provides a method to harness the power of AI to make baseball predictions that are easy to understand and share.

11. References

[1] Sawchik, Travis_. "Baseball's Top Staffs Have Come Around On The High-Spin Fastball." *FiveThirtyEight*, FiveThirtyEight, 5 Oct. 2018, fivethirtyeight.com/features/baseballs-top-staffs-have-come-around-on-the-high-spin-fastball/.

[2] Arthur, Rob. "Want to Predict a Hitter's Future? Sometimes It Takes Just a Single Batted Ball." *The Athletic*, 20 Apr. 2018, theathletic.com/321270/2018/04/19/want-to-predict-a-hitters-future-sometimes-it-takes-just-a-single-batted-ball.

[3] Haechral, Matt. "Matchup Probabilities in Major League Baseball." *Sabr.Org*, 2014, sabr.org/journal/article/matchup-probabilities-in-major-league-baseball.

[4] McBride, Michael. "Introducing SRC and OSWC: Using Game Theory to Assign Credit for Offensive Outcomes." *2020 SABR Analytics Conference*, 2020, sabr.org/latest/2020-sabr-analytics-conference-research-presentations.

[5] Lundberg, Scott, and Su-In Lee. "A Unified Approach to Interpreting Model Predictions." *ArXiv:1705.07874 [Cs, Stat]*, Nov. 2017. *arXiv.org*, http://arxiv.org/abs/1705.07874

[6] Tango, Tom. *Run Expectancy Matrix*, 1950-2015. http://tangotiger.net/re24.html. Accessed 14 Dec. 2020.

[7] *Exceeding Expectations: RE24 Leaders* |. http://www.highheatstats.com/2020/04/exceeding-expectations-re24-leaders/.

[8] Pemstein, Jonah. "Introducing the Batter-Specific Run-Expectancy Tool." *FanGraphs*. 12 May 2016, blogs.fangraphs.com/introducing-the-batter-specific-run-expectancy-tool.

[9] "Markov Chains." *Introduction to Probability*, Joseph K. Blitzstein and Jessica Hwang, CRC Press/Taylor & Francis Group, 2015, pp. 459.

[10] Pankin, Mark D. "Markov Models/Batting Order Optimization." *Markov Baseball Models*, 1987, www.pankin.com/markov/index.html.

[11] Bukiet, Bruce, et al. "A Markov Chain Approach to Baseball." *Operations Research*, vol. 45, no. 1, Jan.-Feb. 1997, pp. 14-23, doi:10.1287/opre.45.1.14.

[12] "Batting (Dis)Order." *The Book: Playing the Percentages in Baseball*, by Tom Tango et al., TMA Press, 2006, pp. 153.

12. Appendix

Table 5: Detailed list of input features for Singlearity-PA

Feature Group Name	#of Inputs	Details
Batter Historical	14	Batter 365 day moving average rates per PA: (1)# of PA (11)Rates per PA for: 1B, 2B, 3B, HR, BB, IBB, HBP, GDP, SO, SF, SH, wOBA (1) Average park factor of PAs[5] (1) Imputed[6]
Pitcher Historical	14	Pitcher 365 day moving average rates per PA against batters: (1)# of PA (11)Rates per PA for: 1B, 2B, 3B, HR, BB, IBB, HBP, GDP, SO, SF, SH (1) Average park factor of PAs (1) Imputed
Batter Recent	9	Batter 21 day moving average rates per PA: (1) # of PA (6) Rates per PA for: 1B, 2B, HR, BB, SO, wOBA (1) Average park factor of PAs (1) Imputed
Pitcher Recent	9	Pitcher 21 day moving average rates per PA against batters: (1) # of PA (6) Rates per PA for: 1B, 2B, HR, BB, SO, wOBA (1) Average park factor of PAs (1) Imputed
Batter/Pitcher head-to-head	8	Head-to-head over the last 3 years: (1) # of PA (6) Rates per PA for: 1B, 2B, HR, BB, SO, wOBA (1) Imputed
Park Factor	4	(4) Previous 3 year average ballpark factors of singles, doubles, triples, HR[7]
Base State	4	(1) Outs (3) 1B occupied, 2B occupied, 3B occupied
Position	10	(10) One-hot encoded value of batter's main position (9 positions + DH)
Platoon Statistics	5	(1) Lefty/Righty matchup (1) # of batter PA against this handed pitcher (1) Batter's wOBA against this handed pitcher

[5] To account for batter and pitcher stats being inflated or depressed due to playing previous games at offensive or defensive ballparks, respectively, we include the average ESPN park factor rating of the stadium where the batter or pitcher PAs occurred.

[6] In several categories, we include an imputed" input (of type Boolean) which indicates whether the number of PAs in this category met the minimum required value. The minimum values are 40 PAs for batter and pitcher historical data, 20 PAs for batter and pitcher recent data, and 10 PAs for batter and pitcher head-to-head data. If the minimum number of PAs are not met for a category, we impute all values in the category using league averages.

[7] http://www.espn.com/mlb/stats/parkfactor

		(1) # of pitcher PA against this handed batter (1) Pitcher's wOBA against this handed batter
Exit Velocity	2	Batter's 3 year exit velocity: (1) Max exit velocity of batter's PAs (1) Average exit velocity of batter's balls in play
Inning	1	(1) Inning

Figure 5: Shapley global feature importance for HR prediction

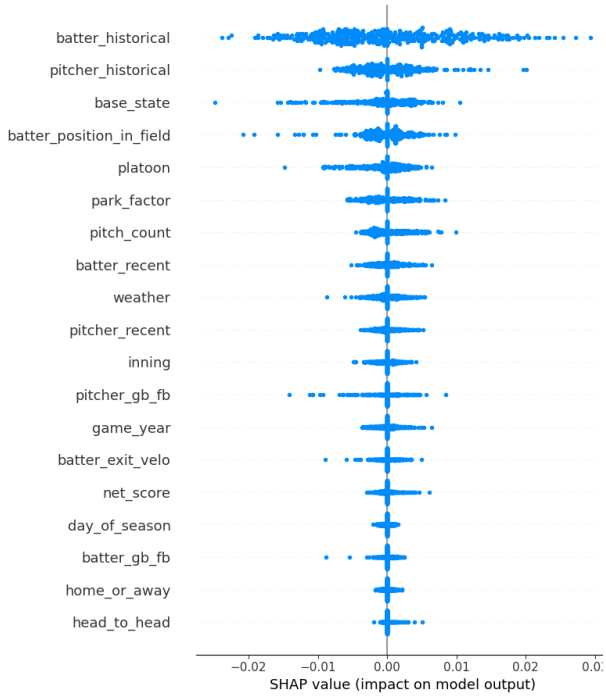


Figure 6: Shapley global feature importance for SO prediction

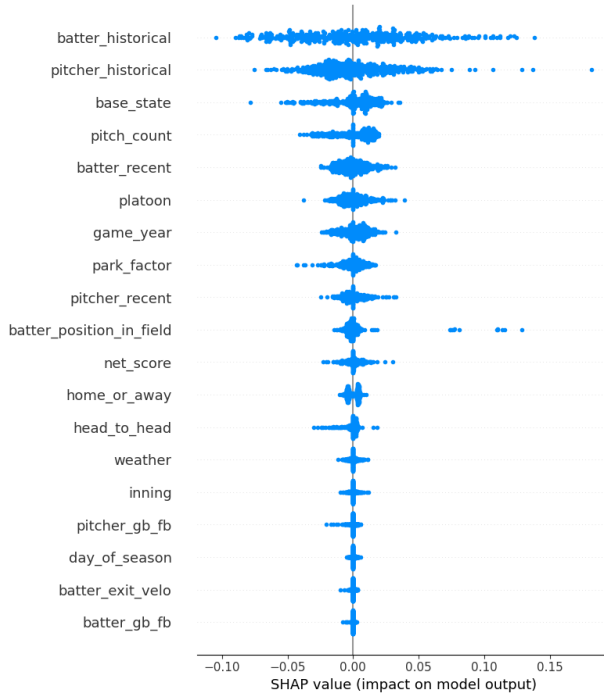


Figure 7: Shapley global feature importance for BB prediction

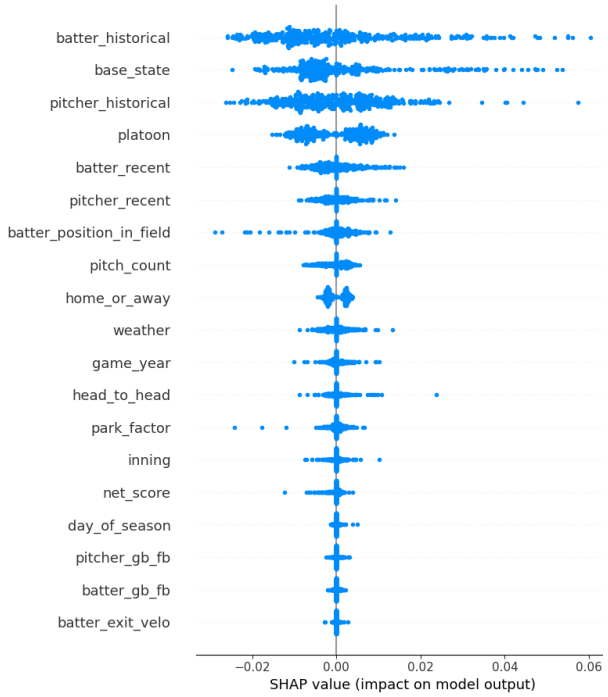


Figure 8: Shapley global feature importance for HBP prediction

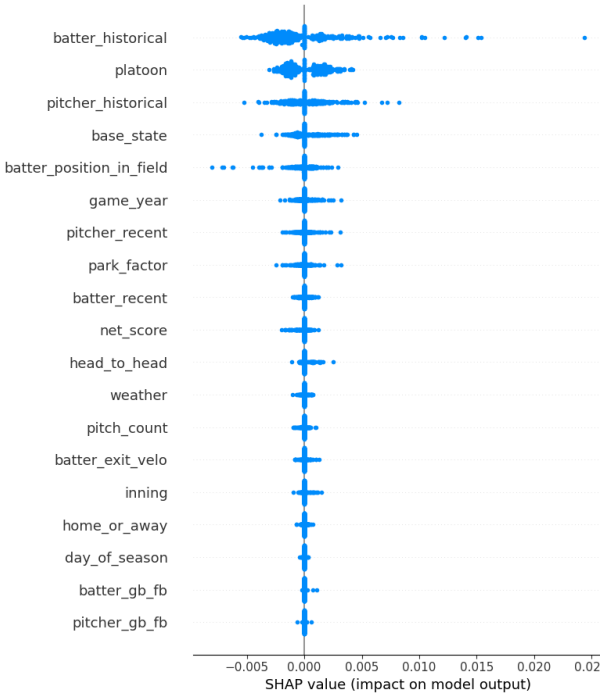


Figure 9: Shapley global feature importance for DP prediction

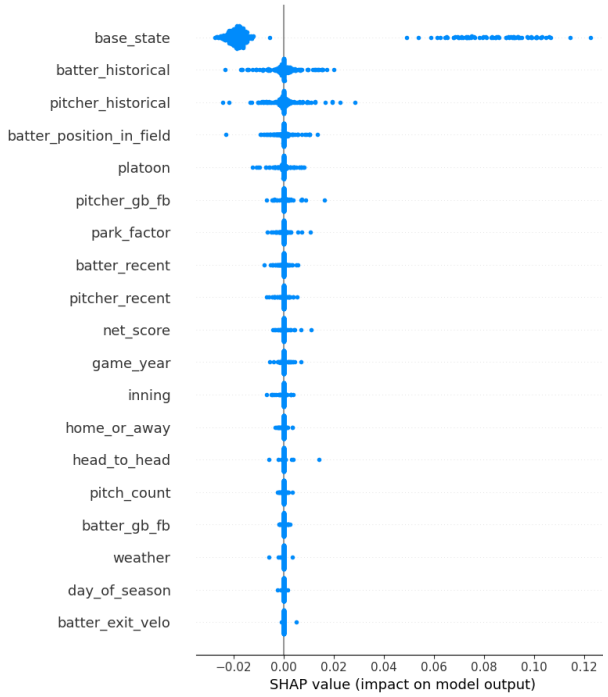


Figure 10: Shapley global feature importance for triple prediction

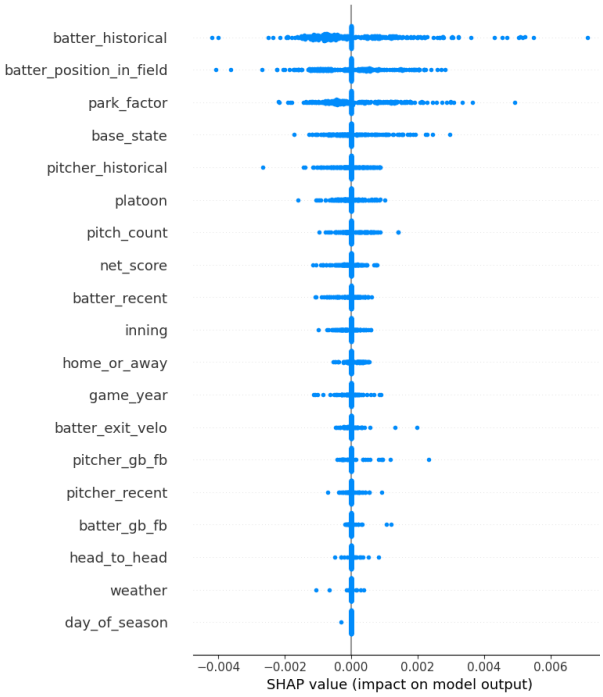


Table 6: 2019 1st inning RE24 matrix

Base Runners			Outs		
1B	2B	3B	0 outs	1 out	2 outs
-	-	-	0.544	0.298	0.115
1B	-	-	0.935	0.564	0.242
-	2B	-	1.147	0.713	0.339
1B	2B	-	1.537	0.979	0.467
-	-	3B	1.369	0.953	0.391
1B	-	3B	1.759	1.219	0.518
-	2B	3B	1.971	1.368	0.615
1B	2B	3B	2.361	1.634	0.743

Figure 11: Projected wOBA for NL starters vs. Justin Verlander in the 2019 All-Star game