Modeling High-Strike Red Sox Pitchers vs Yankees

Maysen Pagan

December 12, 2023

1 Introduction

The Red Sox-Yankees rivalry is known to be the oldest and most famous Major League Baseball (MLB) rivalry dating back to the early 1900s. How well do the Red Sox perform against their biggest rival? The all-time record of these two teams playing each other has the Yankees winning about 200 more games than the Red Sox in the regular season. In this project, I seek to analyze the performance of the Red Sox pitchers against the Yankees by modeling the probability of a pitcher throwing a high proportion of strikes against a particular batter. Using this model, it may be possible to determine what pitchers should pitch against the Yankees as well as what types of pitches should be thrown to certain batters.

It is widely known that statistics plays a huge role in optimizing MLB team performance. Many researchers use machine learning techniques and random forests to answer baseball questions. In one article, the author compared many models like random forest, neural network, and support vector machine to predict player strikeout rates which produced predictions with low error rates. In this analysis, I see how different logistic regression models compare.

Note: Throughout this report, I use the term high-strike and low-strike. High-strike or high strike proportion refers to an bat in which the proportion of strikes thrown by a Red Sox pitcher was greater than or equal to 0.5. Low strike refers to an at bat in which the proportion of strikes thrown by a Red Sox pitcher was less than 0.5.

2 Data Cleaning and Organization

The data used in this project was taken from the Baseball Savant website. Pitch by pitch data was aggregated for every Red Sox game against the Yankees from the 2023 season. This produces a data set with 3,809 rows representing 3,809 pitches and 16 columns. The first few rows of the data can be viewed in Table 1 and a description of the variables are below.

`pitcher`: name of the pitcher
`batter`: name of the batter
`game.pitch`: number of the pitch thrown in whole game
`pitcher.pitch`: number of the pitch thrown by corresponding pitcher in a game
`plate.app`: number plate appearance in a game
`inning`: inning pitch was thrown in
`pitcher.result`: result of pitch thrown
`pitch.type`: type of pitch thrown
`velo`: velocity of pitch thrown (in miles per hour)
`spin`: spin rate of pitch thrown (in revolutions per minute)
`vbreak`: vertical break of pitch thrown (in inches)
`vbreak.direc`: direction of vertial break of pitch (upward or downward)
`hbreak`: horizontal break of pitch thrown (in inches)
`hbreak.direc`: direction of horizontal break of pitch (left or right)

Table 1: First 6 rows of dataset

pitcher	batter	game.pitch	pitcher.pitch	plate.app	inning	pitcher.result
Kenley Jansen	Anthony Volpe	303	17	73	9	In play, out(s)
Kenley Jansen	Anthony Volpe	301	16	73	9	Foul
Kenley Jansen	Anthony Volpe	300	15	73	9	Foul
Kenley Jansen	Anthony Volpe	299	14	73	9	Ball
Kenley Jansen	Anthony Volpe	298	13	73	9	Ball
Kenley Jansen	Anthony Volpe	297	12	73	9	Swinging Strike

pitch.type	velo	spin	vbreak	vbreak.direc	hbreak	hbreak.direc
Cutter	96.7	2729	10	down	1	left
Cutter	95.5	2647	12	down	1	left
Cutter	94.9	2579	14	down	4	left
Cutter	94.6	2559	15	down	2	left
Cutter	95.1	2556	13	down	0	
Cutter	93.7	2554	14	down	3	left

To prepare the data for analysis and modeling, a few columns were added to the dataset. The column `team` was added taking on the value "Red Sox" or "Yankees" corresponding to the team of the pitcher. The data was then filtered to include only those rows where `team` was equal to "Red Sox" (only pitches thrown by the Red Sox). This new dataset now with 1,880 rows was then merged with two other datasets containing the wins above replacement (WAR) of the Red Sox pitchers (using FIP) and of the Yankees batters. These columns were labeled `pwar` and `bwar` for pitchers and batter respectively.

A new column `result` was mutated representing a binary variable that takes 1 if the pitch thrown was a strike and 0 otherwise. The pitches that were considered a strike were those pitches where the result was a swinging strike, called strike, foul, foul tip, foul bunt, and missed bunt.

The final step in preparing the data for modeling was to collapse the dataset so that every row represented a unique pitcher-batter combination. For each unique combination, the averages were taken of the variables `pwar`, `bwar`, `velo`, `spin`, `vbreak`, and `hbreak`. A new column `prop` was then created representing the proportion of pitches thrown by the pitcher that were considered a strike. This column was used to create the binary response variable of my analysis labeled `high`. If the proportion of strikes thrown was greater than 0.5, the binary response variable is 1 and is 0 if the proportion of strikes is less than 0.5.

3 Exploratory Data Analysis

Before modeling the probability Red Sox pitchers throw high strike proportions against Yankees batters, I conduct some exploratory data analysis to better understand the data.

I first look at a correlation heatmap to observe the linear relationships between all of the numerical variables. Figure 1 shows that darker tiles correspond with two variables that have strong negative or positive correlations. The most noticeable tile is that of the vertical break and velocity of the pitch thrown, suggesting the presence of redundant information and multicollinearity. As a result, the vertical break of each pitch will not be included as a variable in the modeling.

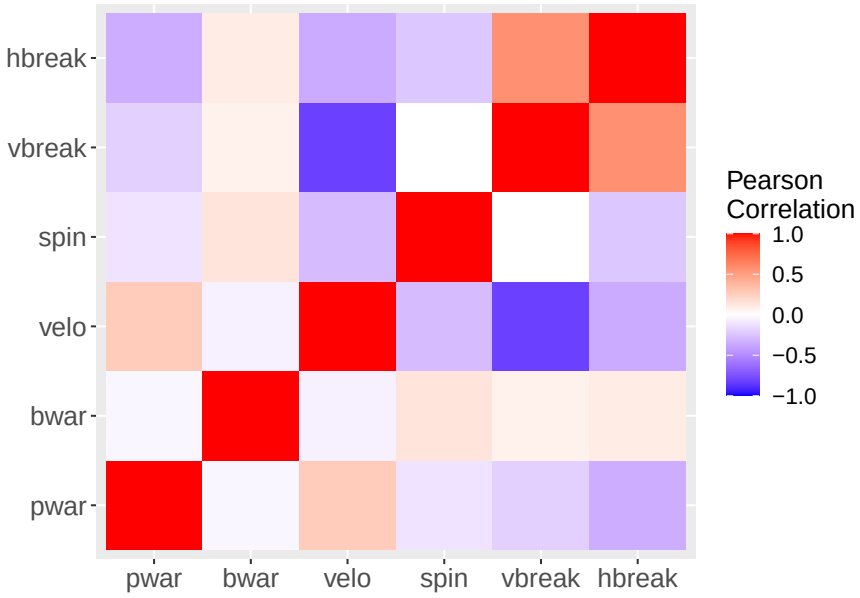

Figure 1: Correlation heatmap of numerical variables in data set.

Next, I look at the distribution of high-strike and low-strike at bats for different Yankees batters with different WARs. Figure 2 displays a violin plot to visualize this distribution. The distribution of pitchers who threw high strike proportions appears to take on the same shape as pitchers who threw low strike proportions. However, is this the case for all pitchers? I then looked closer at how high strike proportions varies across different Red Sox pitchers.

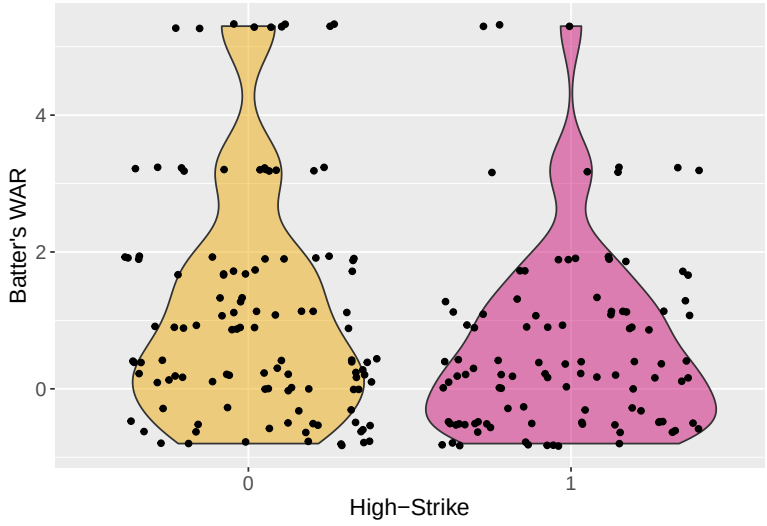

Figure 2: Violin plot of Yankees batters' WAR for high-strike and low-strike at bats.

Figure 3 now shows varying distributions of high-strike at bats. For some pitchers like Brandon Walter and Tanner Houck, fewer pitches that resulted in strikes are being thrown to Yankees batters with higher WARs. However, for other pitchers like Kenley Jansen and Zack Weiss, more pitches that resulted in strikes are being thrown to batters with higher WARs. It appears that the probability of a pitcher throwing high-strikes against a batter varies across different pitchers suggesting that a multilevel model with varying intercepts and slopes may be an appropriate fit to the data. This can also be seen in Figure 4 with the varying proportions of high-strikes for each pitcher.

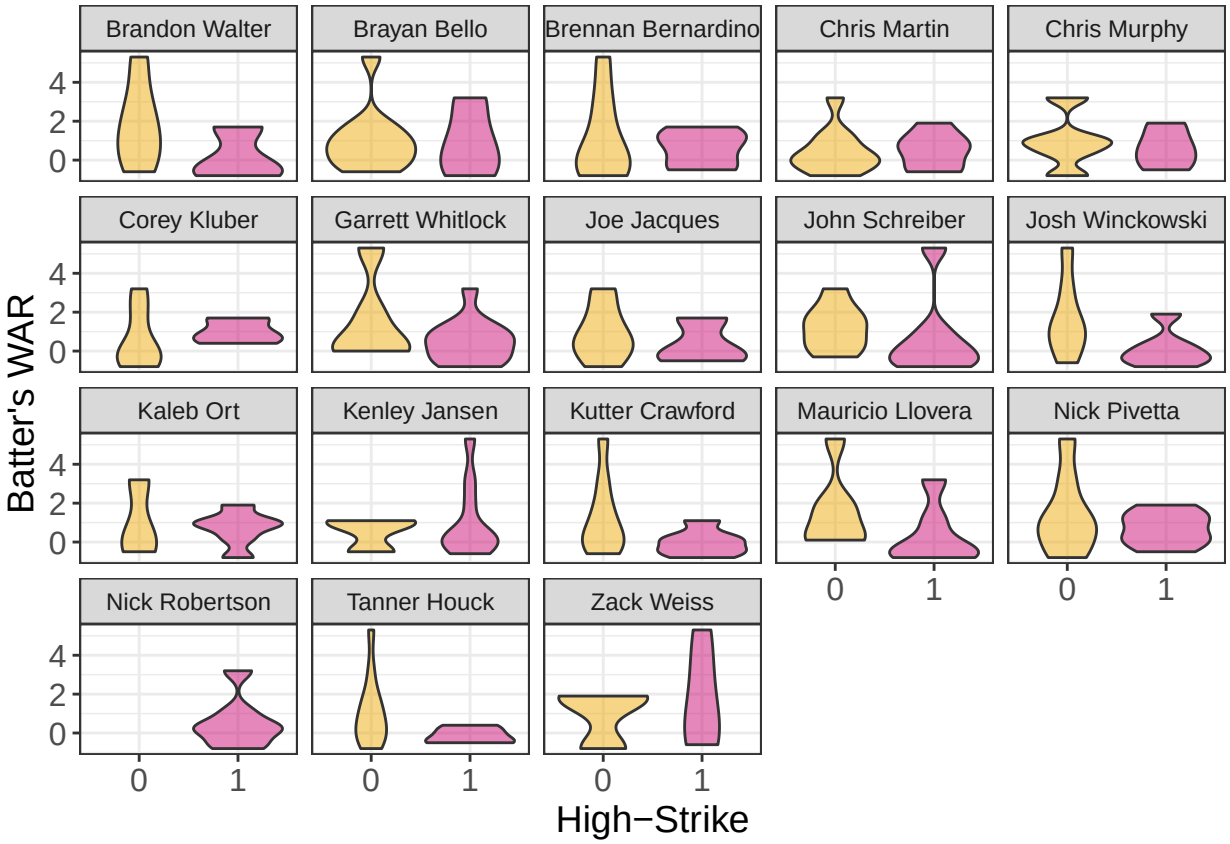

Figure 3: Violin plot of Yankees batters' WAR for high-strike and low-strike at bats across Red Sox pitchers.

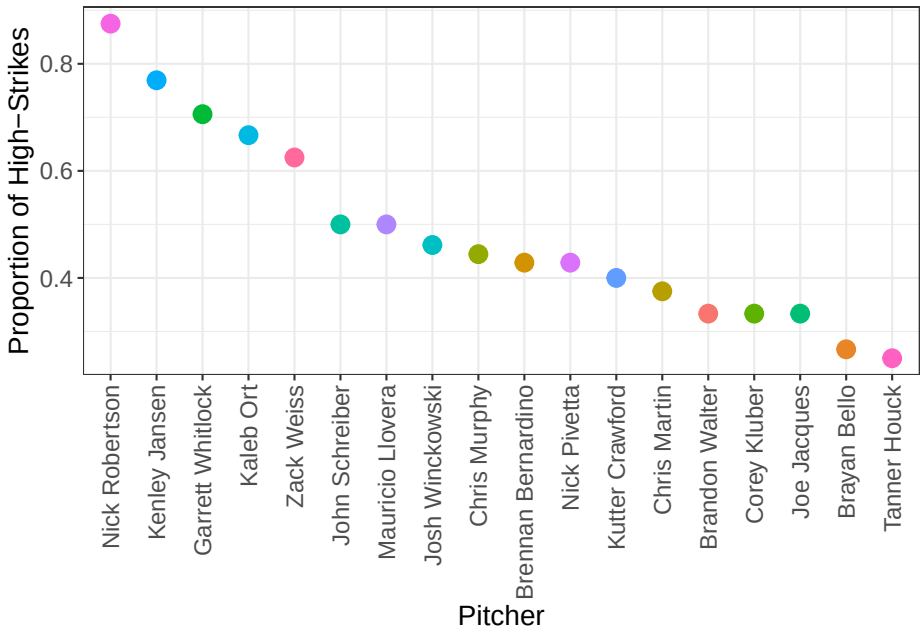

Figure 4: Proportion of times Red Sox pitchers threw high-strikes in the 2023 regular season against the Yankees.

4 Models

In this section, I fit and compare different models to the data to predict whether a high strike proportion is thrown to a particular batter while considering other variables. The data was randomly split into training (80%) and testing (20%) datasets where the models were produced from the training dataset and predictions were compared using the testing dataset. The first model I fit was that of the null logistic model, a model with no predictors. The output of the model is below.

```
## stan_glm
##  family:       binomial [logit]
##  formula:      high ~ 1
##  observations: 174
##  predictors:   1
## ------
##             Median  MAD_SD
## (Intercept) -0.2333  0.1547
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

The fitted values for this first model can be found from the following equation:

$$\frac{1}{1+e^{-(-0.2333)}} \approx 0.44$$

This value is average of the response variable in the training data set and so is the proportion of times a Red Sox pitcher had a high strike proportion against a Yankees batter.

The second model fit to the data was a logistic model with the predictors of the batter's WAR and characteristics of the pitches thrown to the batter. These variables are the average velocity, spin, and horizontal break of pitches thrown to the batter. This model does not consider the variability by pitcher and so this model is also the complete pooling model. The output of this model is below.

```
## stan_glm
##  family:       binomial [logit]
##  formula:      high ~ bwar + velo + spin + hbreak
##  observations: 174
##  predictors:   5
## ------
##             Median  MAD_SD
## (Intercept) -9.9064  5.9837
## bwar        -0.3256  0.1120
## velo         0.0706  0.0525
## spin         0.0015  0.0009
## hbreak       0.0286  0.0461
##
## ------
## * For help interpreting the printed output see ?print.stanreg
## * For info on the priors used see ?prior_summary.stanreg
```

The next model considered was that of the partial pooling model where we take into account the varying high strike proportions by pitcher. This model varies the intercepts and slopes. The coefficients can be viewed in Table 3.

Using the same partial pooling model, I consider adding a group-level predictor, `pwar`, the WAR of each Red Sox pitcher. The coefficients can be viewed in Table 4.

Briefly comparing the AIC and BIC of the partial pooling model without a group-level predictor and with the group-level predictor, the AIC only slightly decreases by one unit and the BIC actually increases. As a result, it does not appear that adding the group-level predictor adds to the prediction performance of the model.

```
## Without group-level AIC: 275.1175
```

```
## With group-level AIC: 306.6702
```

```
## Without group-level AIC: 328.8214
```

```
## With group-level AIC: 373.0104
```

The last model I consider is a no pooling model where a logistic model is created for each pitcher. The coefficients for each of the 18 models for 18 pitchers are placed in Table 5.

4.1 Comparing Models

In this section, I compare the four models (null, complete pooling, partial pooling with varying intercepts, and no pooling) by calculating and comparing the misclassification errors of each model used on the training

Table 2: Misclassification Errors for 4 Models

Model	Misclassification.Error
Null	0.4367
Complete Pooling	0.4886
Partial Pooling	0.3913
No Pooling	0.5217

dataset. The misclassification error for a model is defined as the proportion of false positives and false negatives generated the model. For each calculation the threshold of 0.40 was used to distinguish a high strike plate appearance from a low strike plate appearance. Therefore, if the probabilities from the predictions of the testing dataset were greater than 0.40, that observation would be considered as a high-strike result. The general formula for the misclassification error is then:

$$\frac{\sum_{i=1}^{n} \text{predicted}_i \neq \text{actual}_i}{n}$$

where n is the number of observations in the testing dataset.

For the null model and complete pooling model, the average misclassification error was calculated from the posterior predictive draws of the new data. A function was created to loop through the iterations of the posterior predictions and for each iteration, classify each observation as high-strike (1) or low-strike (0) and calculate the misclassification error. The average of these errors was then taken by dividing by the number of iterations.

The misclassification errors of each model can be viewed in Table 2.

```
## [1] 0.3913043
```

5 Discussion

Simply comparing the misclassification errors of the four models, the partial pooling model with no group-level predictor performs the best with a misclassification error of 39.13%. The model with the highest misclassification error is the no pooling model with over a 10% higher rate than the partial pooling model. The no pooling model creates a separate regression model for each pitcher and assumes that every plate appearance does not share similarities across pitchers. This model assumes no information is shared among the pitchers and each pitcher is considered to be independent from one another. Based on the misclassification error of the no pooling model, it appears that this analysis tends to overfit the data and overstate the variation for each pitcher.

On the other hand, the complete pooling model ignores the variation between pitchers and produces one model that groups all pitchers together to give one estimate of whether a plate appearance results in a high strike proportion. This model produces the second largest misclassificaiton error of 48.86% as it fails to account the variability of the response variable that was visualized in Figure 3 and Figure 4. Additionally, the objective of this analysis was to determine which Red Sox pitchers could throw a high strike proportion against Yankees batters with different WARs and so it would be beneficial to "pool away" this variable.

As a result, a compromise between the extremes of no pooling and complete pooling is the partial pooling model which takes into account the variation of high-strikes within and between pitchers. With the lowest misclassification error, the varying intercept and slopes of this model suggests that the Red Sox pitcher affects the high-strike proportion result and it also influences the effect of the average velocity, average spin, and average horizontal break of the pitches thrown during an at bat as well as the effect of the batter's WAR.

With this model, we can make predictions for new observations. For example, say we want to predict the probability that Red Sox pitcher Kenley Jansen throws a high strike proportion against Yankees player Kyle

Higashioka in the 2023 regular season. Kyle Higashioka has a WAR of 1.7 and say the average velocity, spin, and horizontal break of the pitches thrown to Higashioka are 93.0, 2300, and 5 respectively. Then the probability of Kenley Jansen throwing more pitches that result in strikes to Kyle Higashioka during one plate appearance can be determined from the following:

$$\begin{aligned} P(y=1) &= \frac{1}{1+e^{-X\beta}} \\ &= \frac{1}{1+e^{-[-7.19+(0.32*1.7)+(0.05*93)+(0.001*2300)+(0.02*5)]}} \\ &= \frac{1}{1+e^{-0.82}} \\ &\approx 0.6952 \end{aligned}$$

This result suggests that the estimated probability of Kenley Jansen throwing more pitches that result in strikes against Kyle Higashioka during a plate appearance is about 70%. We can compare this to another pitcher. What is the probability of Red Sox pitcher Chris Martin throwing a high-strike proportion against Higashioka with the same average stats? Now using the coefficients associated with Chris Martin,

$$\begin{aligned} P(y=1) &= \frac{1}{1+e^{-X\beta}} \\ &= \frac{1}{1+e^{-[-8.10+(-0.05*1.7)+(0.05*93)+(0.001*2300)+(0.04*5)]}} \\ &= \frac{1}{1+e^{-(-0.80)}} \\ &\approx 0.3091 \end{aligned}$$

This result suggests that the estimated probability of Chris Martin throwing more pitches that result in strikes against Kyle Higashioka is about 31%, about 40% less than if Kenley Jansen were to pitch.

Although this partial pooling model produces the lowest misclassification error rate, it is still much larger than those error rates seen from machine learning and random forest classification. For future work, more predictors that have stronger relationships with strike proportion could be added to this partial pooling logistic model. For example, one important factor that could be included is the handedness of the pitcher and batter. Lefty-lefty and righty-righty matchups are favorable to the pitcher as their breaking pitches will curve away from the batter. Other batter statistics could also be added to the model like On-base Plus Slugging (OPS) to improve predictions from increased opponent information.

6 Appendix

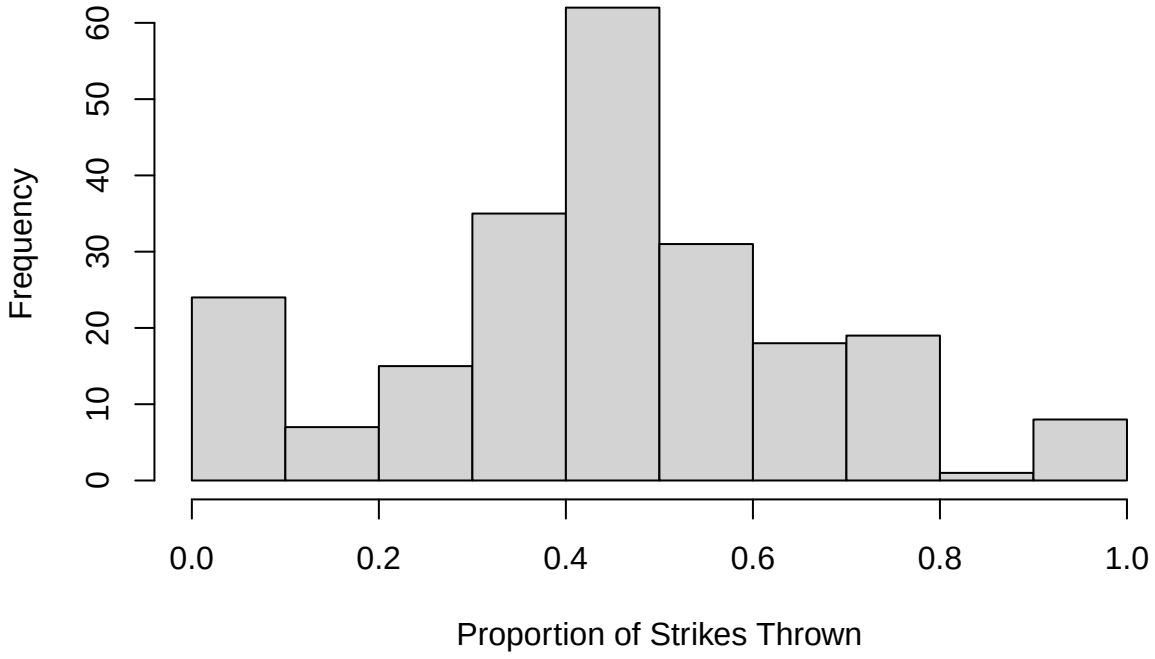

Figure 5: Histogram of proportion of strikes thrown by Red Sox pitchers against a Yankees batter. The distribution is approximately symmetrical. As a result, pitchers with a strike proportion greater than 0.5 will be labeled as having a high strike proportion against a Yankees batter.

Table 3: Coefficients for partial pooling models by pitcher

	(Intercept)	bwar	velo	spin	hbreak
Brandon Walter	-8.012501	-0.9585623	0.0533758	0.0010031	0.0326321
Brayan Bello	-9.011865	-0.1509241	0.0533886	0.0010196	0.0443013
Brennan Bernardino	-7.973247	-0.3480767	0.0538684	0.0009880	0.0420119
Chris Martin	-8.103947	-0.0455073	0.0532810	0.0009697	0.0383336
Chris Murphy	-7.110609	-0.1725026	0.0532724	0.0010199	0.0306592
Corey Kluber	-8.084531	-0.0241277	0.0531314	0.0009834	0.0810199
Garrett Whitlock	-5.752528	-0.8654767	0.0534715	0.0009846	0.0552558
Joe Jacques	-8.294636	-0.2648855	0.0529989	0.0010123	0.0381599
John Schreiber	-6.787995	-0.2501221	0.0533851	0.0009951	0.0484672
Josh Winckowski	-6.168195	-0.9998849	0.0535669	0.0009917	0.0394947
Kaleb Ort	-6.509611	-0.3157966	0.0537436	0.0010051	0.0396353
Kenley Jansen	-7.189097	0.3217366	0.0536081	0.0010303	0.0222872
Kutter Crawford	-7.621901	-0.7930677	0.0535605	0.0009869	0.0490562
Mauricio Llovera	-7.572934	-0.9573567	0.0531417	0.0010093	0.0441068
Nick Pivetta	-7.495080	-0.3694906	0.0533741	0.0010094	0.0591822
Nick Robertson	-5.472584	-0.7881973	0.0535913	0.0009974	0.0415692
Tanner Houck	-9.663864	-0.5010195	0.0532644	0.0010013	0.0401689
Zack Weiss	-7.660437	0.0496697	0.0536102	0.0009948	0.0366239

Table 4: Coefficients for partial pooling models with group-level predictor by pitcher

	(Intercept)	bwar	velo	spin	hbreak	pwar
Brandon Walter	-3.376557	-0.9101262	0.0343030	0.0000833	-0.0066495	-0.0453757
Brayan Bello	-4.895012	-0.0993601	0.0345583	0.0000735	0.0473697	-0.2239847
Brennan Bernardino	-4.391607	-0.3391128	0.0349687	0.0000586	0.0592340	-0.0831715
Chris Martin	-4.364910	-0.0597182	0.0344623	0.0000212	0.0277069	-0.0078934
Chris Murphy	-3.667538	-0.2655516	0.0343486	0.0001093	0.0752554	-0.0117143
Corey Kluber	-5.291397	-0.0934713	0.0340588	0.0000431	0.1850092	0.1763500
Garrett Whitlock	-3.473429	-0.8400188	0.0346432	0.0000789	0.1112424	-0.0431910
Joe Jacques	-4.728724	-0.1927529	0.0337644	0.0000789	0.0270639	-0.0367921
John Schreiber	-4.061662	-0.3336947	0.0344919	0.0000616	0.1050348	-0.0457477
Josh Winckowski	-3.189895	-0.9813637	0.0345116	0.0000785	0.1319716	-0.0257957
Kaleb Ort	-3.530101	-0.4597624	0.0347499	0.0000825	0.1406646	-0.0868312
Kenley Jansen	-3.734485	0.2407714	0.0349145	0.0001305	0.0581234	0.2290226
Kutter Crawford	-3.716963	-0.7370528	0.0346480	0.0000554	0.0956845	-0.1061300
Mauricio Llovera	-3.768792	-0.8809133	0.0340029	0.0000894	0.0578909	-0.0384228
Nick Pivetta	-4.201211	-0.4026158	0.0344070	0.0000875	0.2124442	-0.1201644
Nick Robertson	-2.735563	-0.8022297	0.0346883	0.0000735	0.2075029	-0.0801255
Tanner Houck	-4.393028	-0.4232486	0.0342819	0.0000864	-0.0162039	-0.1488185
Zack Weiss	-4.263689	0.0279623	0.0348112	0.0000692	0.0658494	-0.0671442

Table 5: Coefficients for no pooling models by pitcher

	(Intercept)	bwar	velo	spin	hbreak
Brandon Walter	13.511482	-0.7120976	-0.1349691	0.0061025	-1.4640676
Brayan Bello	-299.557746	-0.5210194	3.4037250	-0.0015994	-0.3394956
Brennan Bernardino	-25.899202	-0.4960056	0.2020482	0.0001400	0.4538184
Chris Martin	-8.755897	0.8659696	0.6965926	-0.0274250	-0.1772035
Chris Murphy	-68.468941	0.7146455	0.2978456	0.0232135	-1.9929335
Corey Kluber	121.827718	0.1693909	-1.4118893	-0.0019894	0.2825795
Garrett Whitlock	-43.769602	-1.5152778	0.1913802	0.0141915	0.0834745
Joe Jacques	96.572962	0.6378625	-1.1601957	-0.0056876	0.8010667
John Schreiber	42.568666	-0.1441725	-0.0288862	-0.0160049	0.0105264
Josh Winckowski	7.852150	-1.5977744	-0.3891728	0.0098452	1.2413003
Kaleb Ort	-217.087146	-0.3908051	1.7505058	0.0211987	0.2555543
Kenley Jansen	-63.332443	0.7368001	0.1440517	0.0196823	0.5875844
Kutter Crawford	-34.334797	-0.8423699	0.7923959	-0.0153188	-0.1115336
Mauricio Llovera	59.865997	-1.7436609	-0.8231860	-0.0108081	2.2279011
Nick Pivetta	-68.742572	-0.9160060	0.3383593	0.0105215	2.3366213
Nick Robertson	25.431353	-1.1740237	-0.2130540	-0.0023846	0.1550392
Tanner Houck	-29.877104	-0.5556074	0.2004679	0.0090200	-1.0257138
Zack Weiss	32.182977	0.8831726	1.3408858	-0.0596716	0.3182231