Modelling tjStuff+ v1.0

Thomas Nestico 12 min read · Jan 22, 2024

You can find "Modelling tjStuff+ v2.0" here

You can find "Modelling tjStuff+ v3.0" here

Introduction

I have been fascinated by pitching ever since I started watching baseball. The way a baseball moves throughout space seems supernatural. Some pitches are 100 mph laser beams that seem to defy gravity, while other pitches can move in such a way that feels physics breaking. The way the baseball moves is explained by its physical characteristics such as velocity, spin rate, and spin axis. Major League Baseball is ahead of the curve in regard to motion tracking in sports and has made these pitch characteristics publicly accessible since 2015.

These metrics are the basis to almost every public pitch model. Whether it be Eno Sarris' Stuff+ Model (available on fangraphs.com) or Jeremy

Maschino Stuff+ Model (available on mlbpitchprofiler.com), they all employ these features to quantify the value of a pitch.

In this article, I explain my process in creating a machine learning model which aims to predict how effective a pitch is at limiting runs by using the physical characteristics of the pitch.

Data Selection

The most important step in building a machine learning model is ensuring the data which the model is trained from is accurate. Through the use of a scraper I wrote, I was able to download pitch data starting from the 2020 MLB Season through to the 2023 MLB Season. The data provided by MLB is all in the same format, which minimizes the time required for cleaning and processing.

My next step was to decide which data to include in the training of the model. My initial plan was to train on data from 2019 to 2022, and then I could use the model to predict the best pitches in the 2023 season. After further research, I learned that prior to the 2020 season, MLB switched from Trackman to Hawk-Eye to conduct pitch tracking. I decided to train on data from 2020 to 2022 as I would be more confident that the tracked data was more precise. Accounting for all pitches during these seasons, there were over 1.4 Million pitches in the data set.

Data Preparation

The physical characteristics of a pitch are well-defined and accurately measured, however these measurements are not normalized between pitchers of different handedness. This means that metrics such as Horizontal Release Point and Horizontal Break for left-handed pitchers would be scaled by a factor of -1 compared to right-handed pitchers. We can normalize these “mirrored” metrics so that during training, pitches thrown from either hand are on the same scale, which should improve performance.

Feature Selection

With the data selection and preparation complete, it is now time to decide which physical characteristics, or features, should be used to train the model. This process was iterative. I selected a handful of characteristics which I believed were important in characterizing a pitch, and then proceeded with training and testing the model and assessed the performance. The features that I ultimately decided on were as follows:

start_speed

- The speed of a pitch as it is released from the pitcher’s hand, measured in miles per hour

spin_rate

- The rotation per minute of a pitch as it travels through the air

extension

- The release extension of a pitch measured in feet

hb

- Horizontal movement in inches from the catcher’s perspective *ivb*

- Induced Vertical movement in inches from the catcher's perspective

x0

- Horizontal Release Position of the ball measured in feet from the catcher's perspective

z0

- Vertical Release Position of the ball, measured in feet from the catcher's perspective.

spin_axis

- The Spin Axis in the 2D X-Z plane in degrees from 0 to 360, such that 180 represents a pure backspin fastball and 0 degrees represents a pure topspin (12–6) curveball

fb_max_velo_diff

- For any given pitcher, the difference between release speed and their fastball with the greatest release speed.

fb_max_ivb_diff

- For any given pitcher, the difference between induced vertical break and their fastball with the greatest induced vertical break.

fb_max_hb_diff

- For any given pitcher, the difference between horizontal break and their fastball with the greatest horizontal break.

My reasoning for including metrics related to a pitcher's fastball stems from the importance of sequencing in pitching. How different one's fastball is from their off-speed and braking pitches allows the pitcher to approach certain scenarios differently, and adds a layer of deception which makes it difficult for the batter to adapt.

Target Selection

In order to train a regression machine learning model, the data must have a target. In this case, I would like to use a pitch's physical characteristics to predict the effectiveness of a pitch. Some metrics which can quantify this effectiveness include Whiff%, CSW%, Strike%, and Expected Run Value. I decided to proceed with Expected Run Value (xRV) as I believe it more clearly captures a pitch's effectiveness.

Not all xRV are the same. For example, a strike on a 3–2 count with the bases loaded, and 2 outs would be considered significantly more favourable for the pitcher than a strike to start an inning. Since xRV are heavily influenced by more than just the pitch outcome, taking an average of the xRV by outcome and count and using that as the target would assist in neutralizing the effect of runners when looking at individual pitches. I decided to consider the impact of the count in the target, as it would value pitches in a variety of situations differently. For example, a pitch which generated a swinging strike should be treated more valuable if it can generate a strikeout than one that leads to a 0–1 count.

Read more about my approach for grouping run values like this in my article about modelling batter decision values.

Figures 1 and 2 show the breakdowns for the xRV I used as the target in the training of this model.

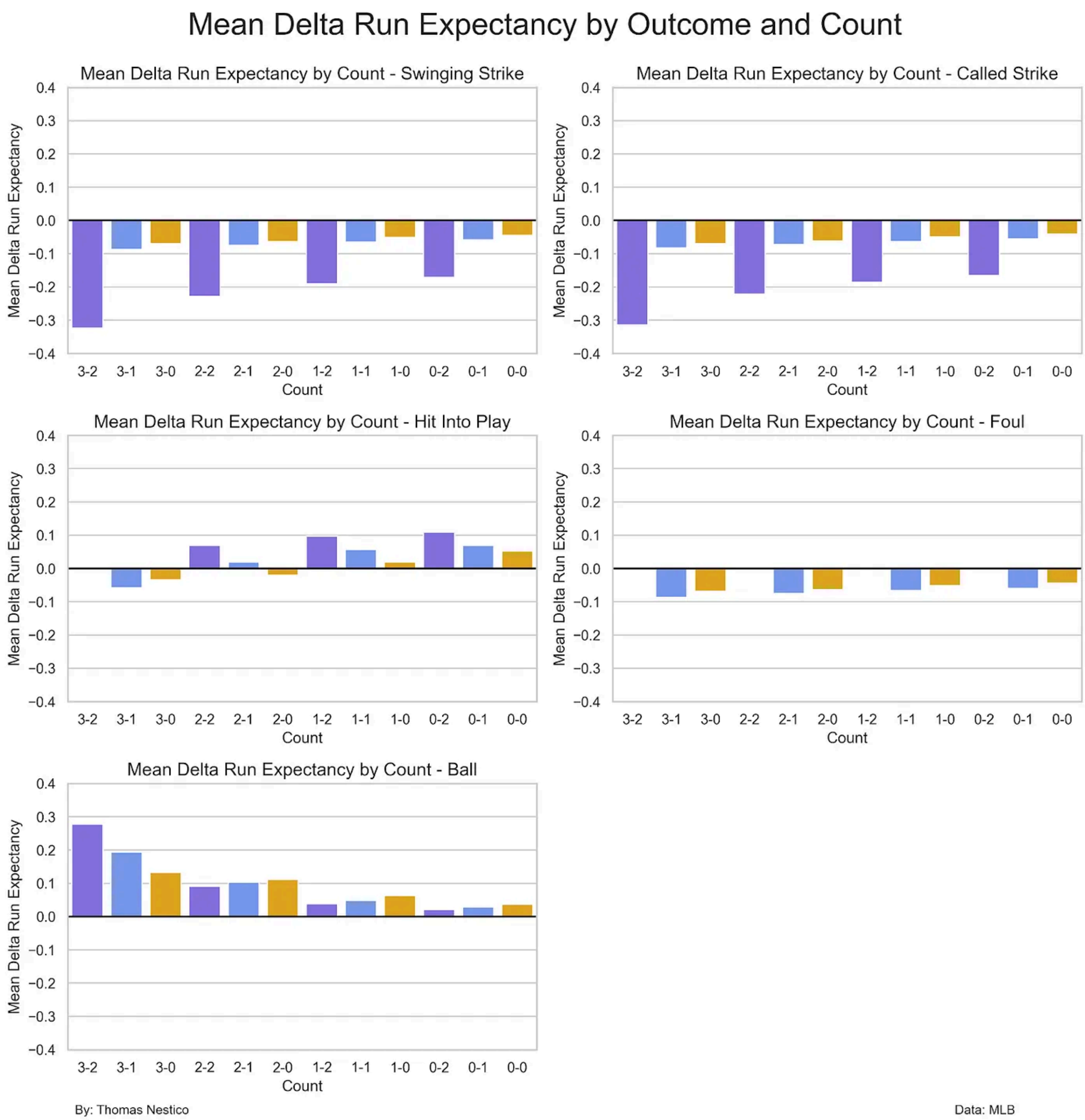


Figure 1: Pitch Outcome Mean Delta Run Expectancy by Count

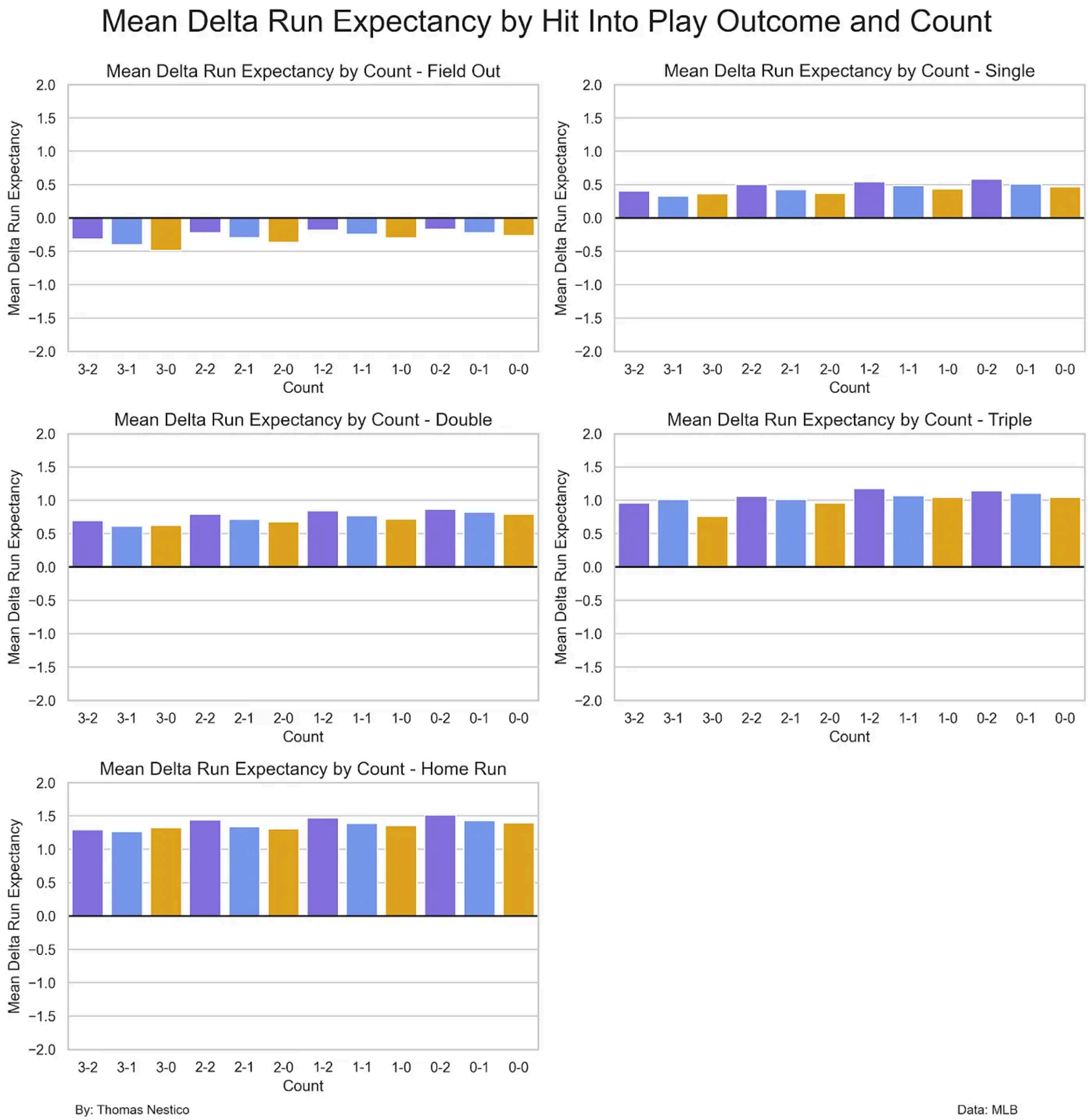


Figure 2: Hit Into Play Outcome Mean Delta Run Expectancy

Model Selection

An XGBoosted decision tree regression model was used to model the run expectancy of a pitch based on its physical characteristics. A decision tree mimics human-decision making and with clearly defined features and targets, the model can accureately predict the run expectancy. Gradient boosting was implemented to improve the performance and efficiency of the model.

I decided to train a single model instead of multiple models for different pitch types (Fastball, Breaking, Off-Speed) because I wanted to capture the effectiveness of any pitch given the specified features compared to all other pitches rather than pitches in specified groups. I believe this would assist in capturing the value of a pitch and comparing what makes a pitch more effective than another pitch. Additionally, after multiple iterations, a single model returned better performance metrics than multiple models.

Feature Importance

Figure 3 ssummarizes the importance of each feature in the model.

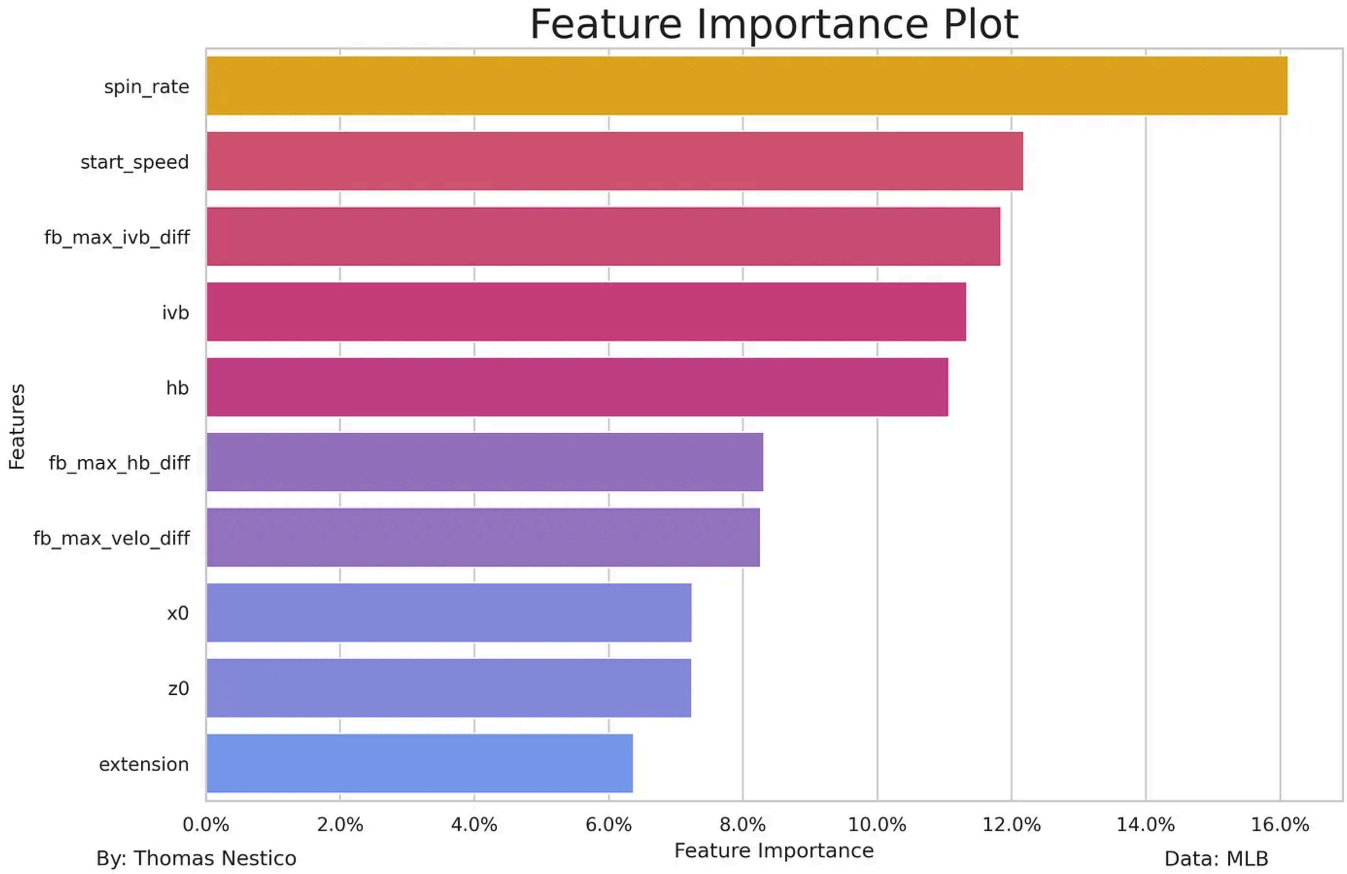


Figure 3: Feature Importance Plot

Initially, we can see that pitch movement, speed, and spin rate related features have the greatest importance in the model, with metrics related to how the pitcher is releasing the ball are ranked as least important. This result is intuitive, as a pitch's movement through the air should be the most important aspect of a pitch's ability to limit runs. The model seems to agree in this regard.

Calculating tjStuff+

The output of my model is expected run value, which means that for any given pitch, the model can predict how effective that pitch is at limiting runs based on its physical characteristics. We can use a standardization technique to assist in comparing pitchers and pitches to one another. This is where the calculation of tjStuff+ arises.

tjStuff+ is similar to the prospect tool grade scale. The prospect tool grade is a normal distribution which uses 50 as the average and 10 as the standard deviation. This means that a prospect with a "60 Grade" hit tool, has a hit tool 1 standard deviation above the mean, which would slot them approximately into the 84th percentile. Increase that to a "70 Grade" hit tool, and now the prospect sits at the 97th percentile of hit tools. Figure 4 illustrates this concept well.

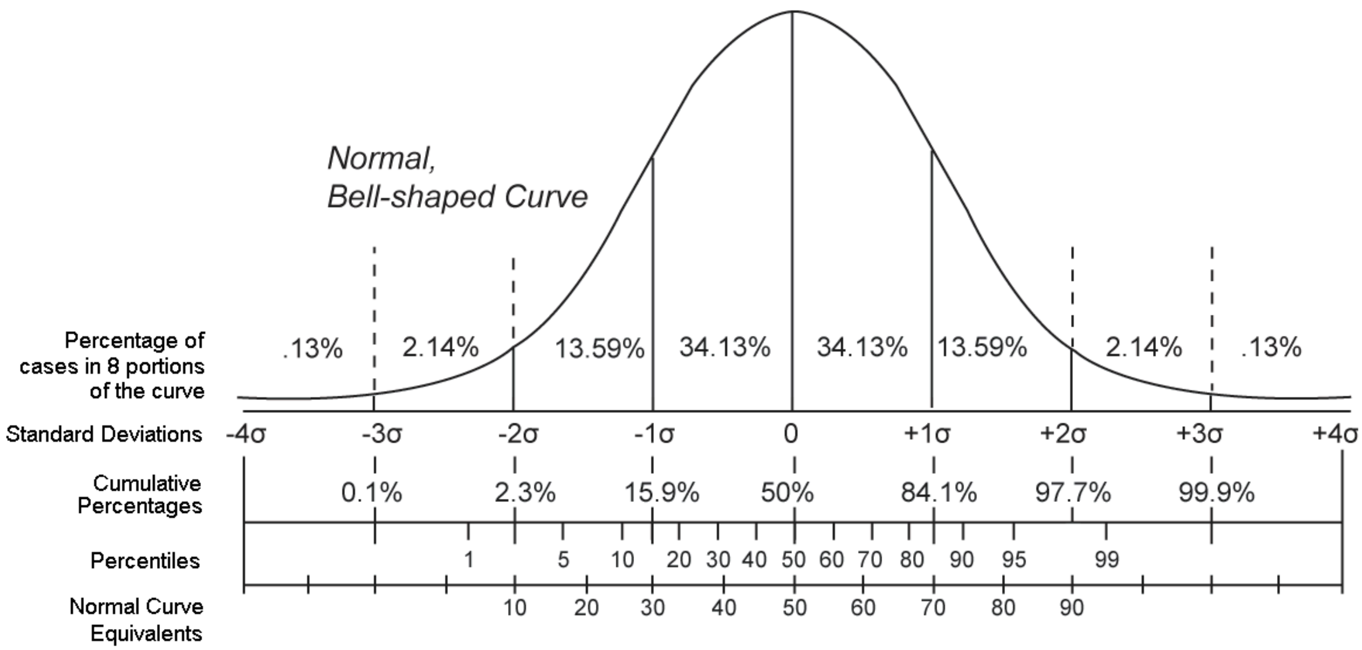


Figure 4: Normal Distribution

tjStuff+ follows this same standardization, but uses 100 as the mean and 10 as the standard deviation.

Model Performance

This goal of the model is to assess a pitcher's ability to limit runs based on the physical characteristics of their pitches. Earned Run Average (ERA) is the measurement of how many earned runs a pitcher allows per 9 innings on average. ERA is the most commonly used metric when assessing the performance of a pitcher, after all, if a pitcher cannot limit the amount of runs they allow, they are failing at their goal. Unfortunately, ERA is not a great indicator of a pitcher's true skill because there are a lot of factors which influence ERA, such as team defence. A pitcher's ERA is also not a strong predictor when trying to predict their future ERA.

When assessing my model, I wanted to test both its descriptiveness and predictiveness regarding run prevention. To test its descriptive power, I applied the model to all data since 2020 and calculated the correlation between my metric and the same season's ERA. To test its predictive power, I conducted a similar process, but instead of applying the model to current season data, I applied it to previous season data and calculated the correlation between my previous seaosn metric and current season ERA. Following the calculation of the correlations, I compared the results to a commonly used predictive and descriptive metric.

Descriptiveness

In terms of descriptiveness, the correlation between ERA and tjStuff+ in the same season is -0.38. A commonly used descriptive statistic when quantifying the true value of a pitcher in Fielding Independent Pitching. When looking at the relationship between ERA and FIP, the correlation jumps up to 0.77. From this, it looks like Stuff+ has a moderately strong relationship with ERA, but its power as a descriptive variable lags behind traditional metrics such as FIP.

These relationships are illustrated in Figure 5 and 6.

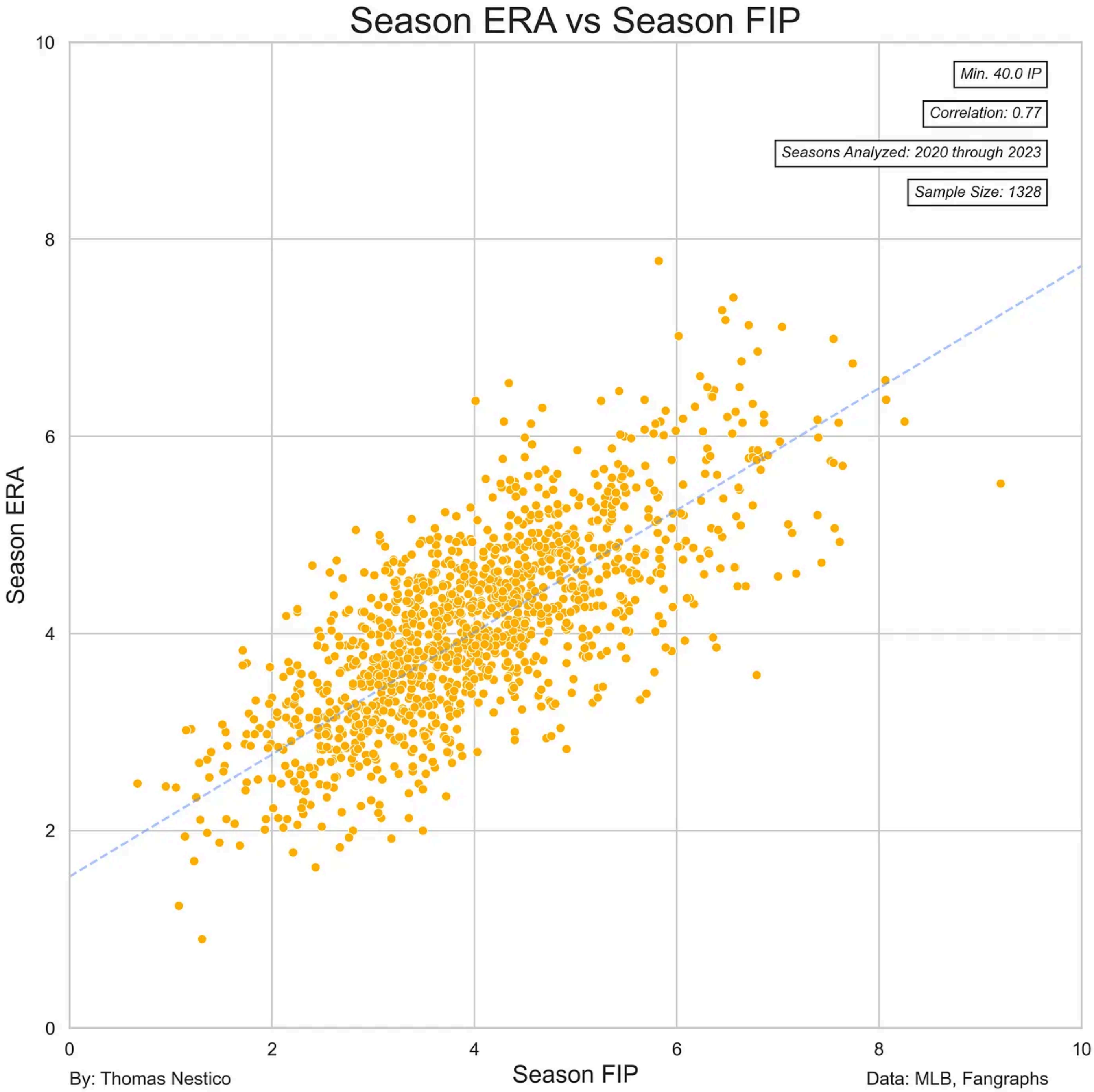
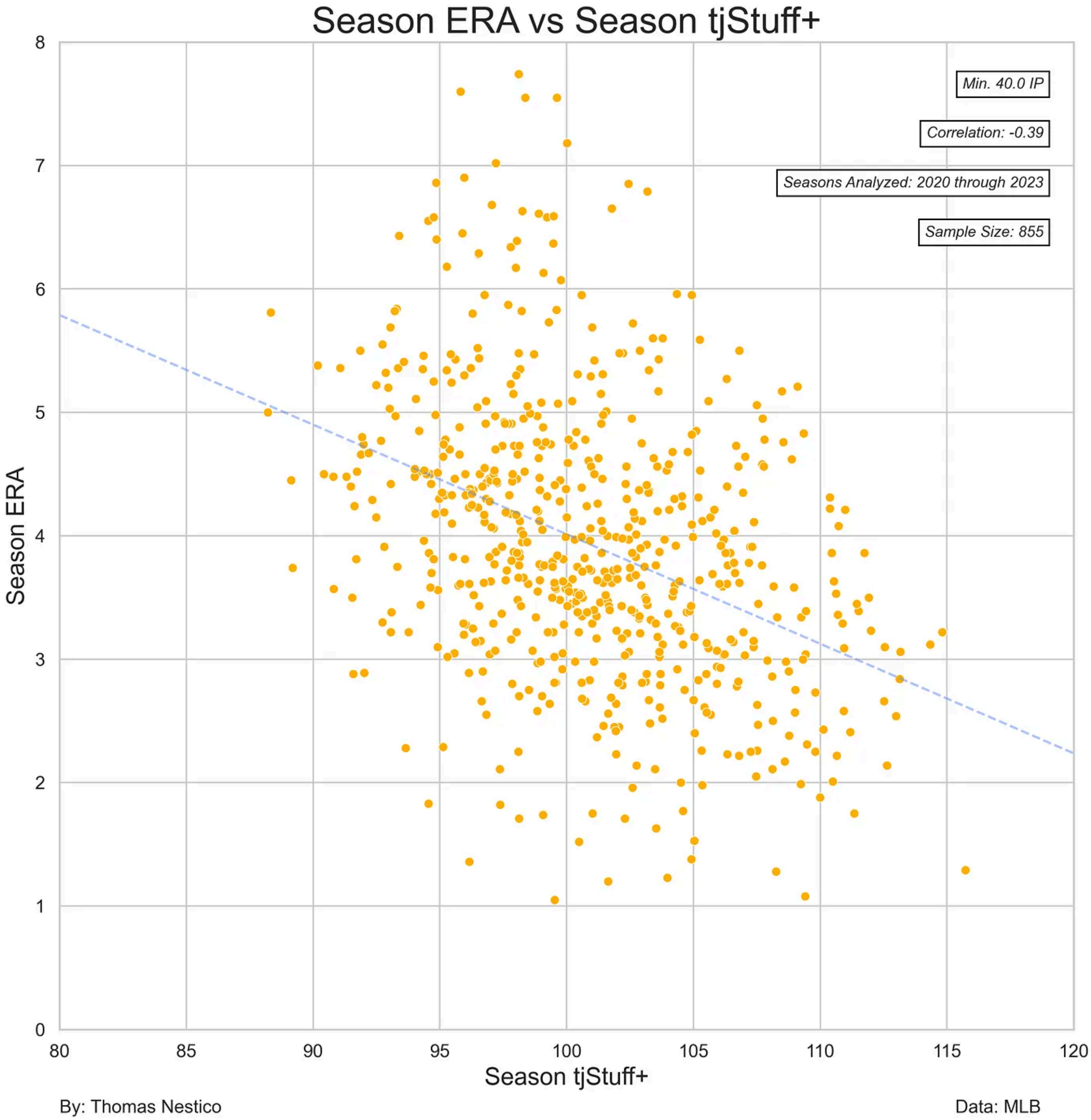


Figure 5: Season Era vs Season FIP



It is understandable that FIP works better as a descriptive statistic than tjStuff+ because it uses results. As Tom Tango writes:

> *"FIP is a DESCRIPTIVE stat, not predictive. It uses actual data and simply weights it and scales it."*

FIP is designed to remove the influence of events out of the pitcher's control, such as fielding and luck. It is solely meant to be descriptive, and it does a fantastic job at doing so. The next step is to test the predictiveness of tjStuff+.

Predictiveness

Since 2020, the correlation between season ERA and previous season ERA was 0.20. This indicates that while there is a positive relationship between ERA of different seasons, but it is not a strong relationship.

Figure 7 illustrates the weak relationship between ERA and previous season ERA.

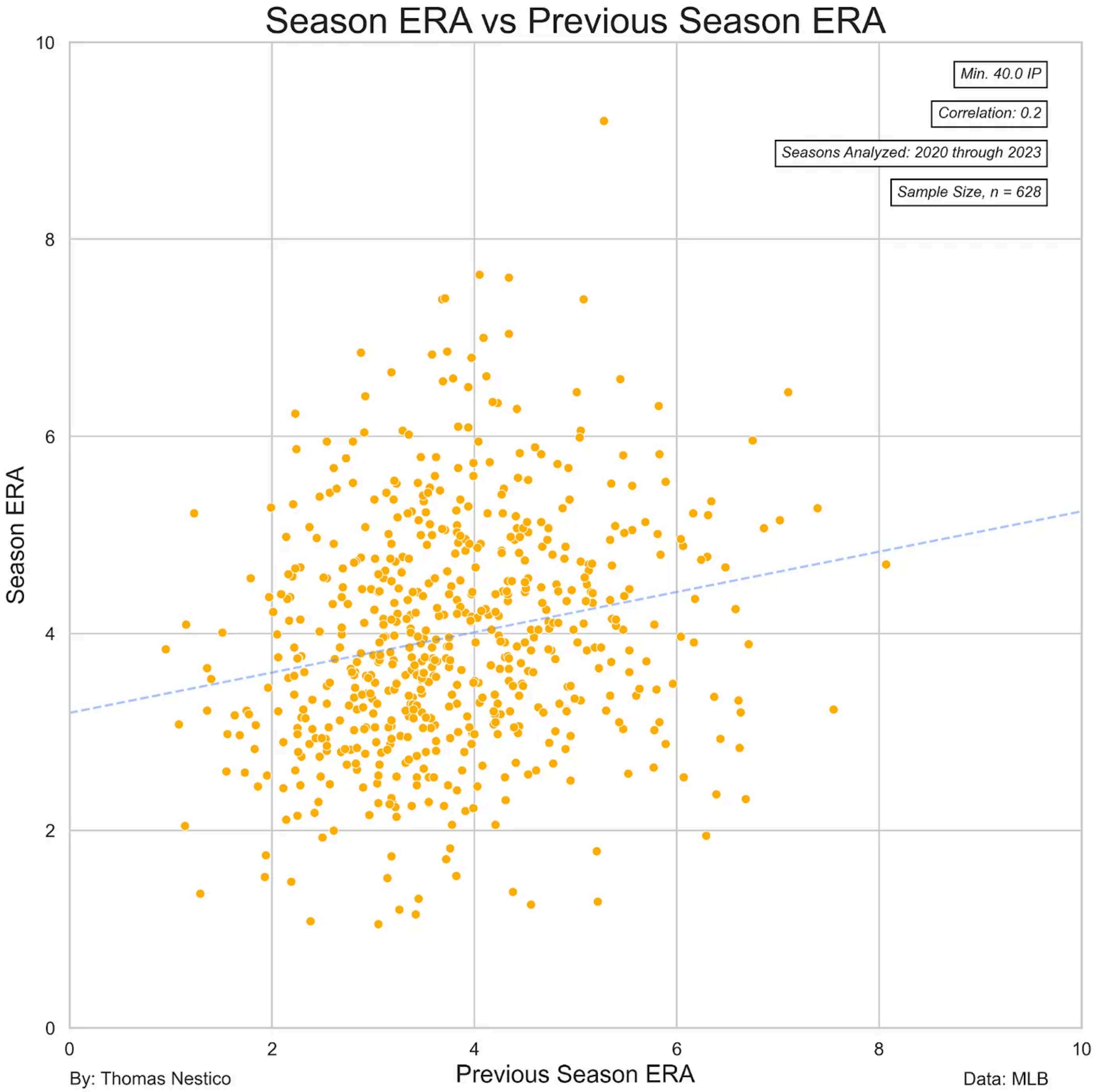


Figure 7: Season ERA vs Previous Season ERA

Fielding Independent Pitching is a descriptive statistic, however, analysts have created a metric which is based off of FIP, but normalizes a pitcher's home run rate based on the league average home run to fly ball ratio. This metric is called Expected Fielding Independent Pitching (xFIP). xFIP is considered to be more of a predictive statistic. To gauge the performances of tjStuff+ as predictive model, we will compare it to xFIP. The relationship between current season ERA vs previous season xFIP is shown in Figure 8.

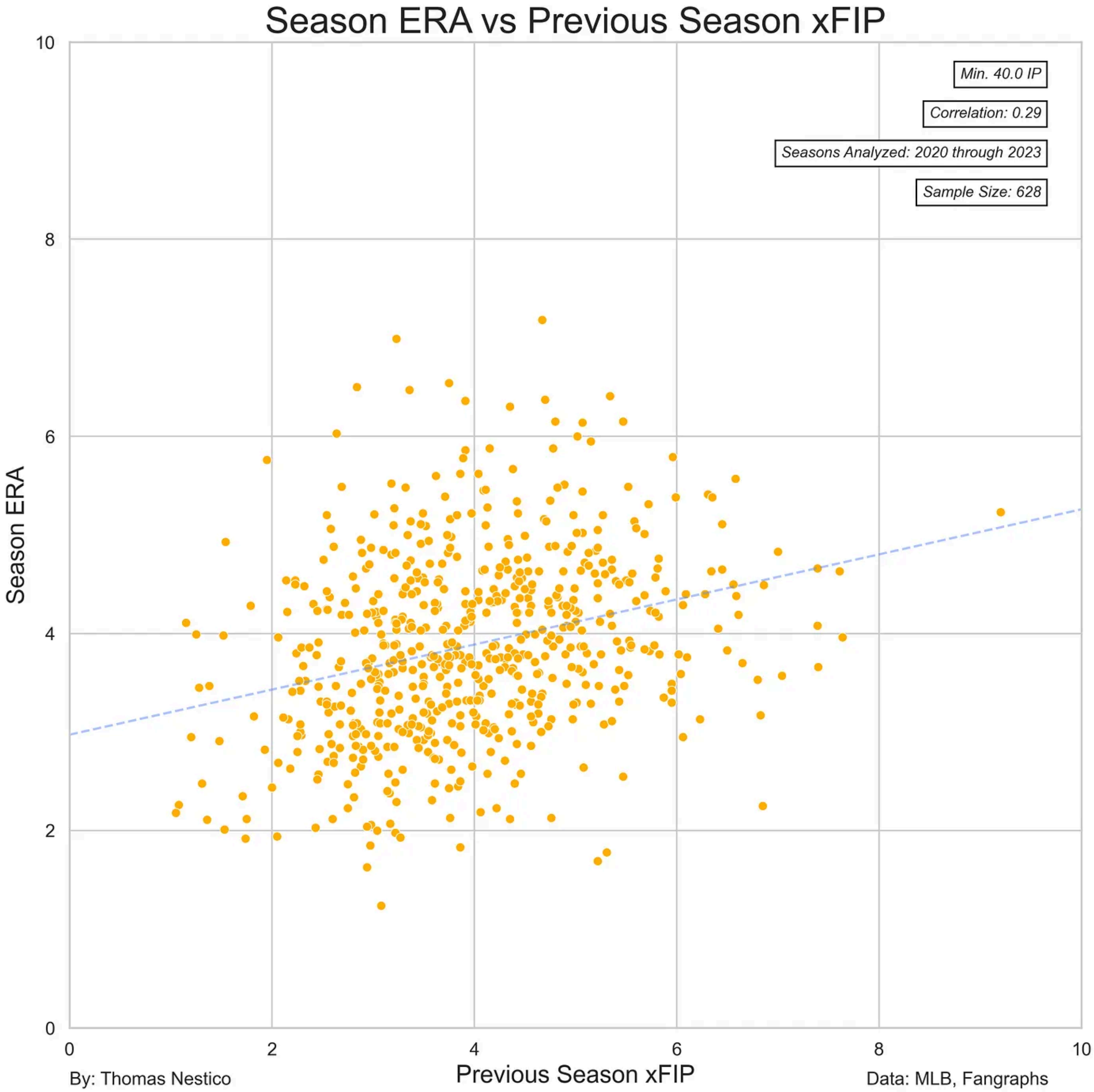
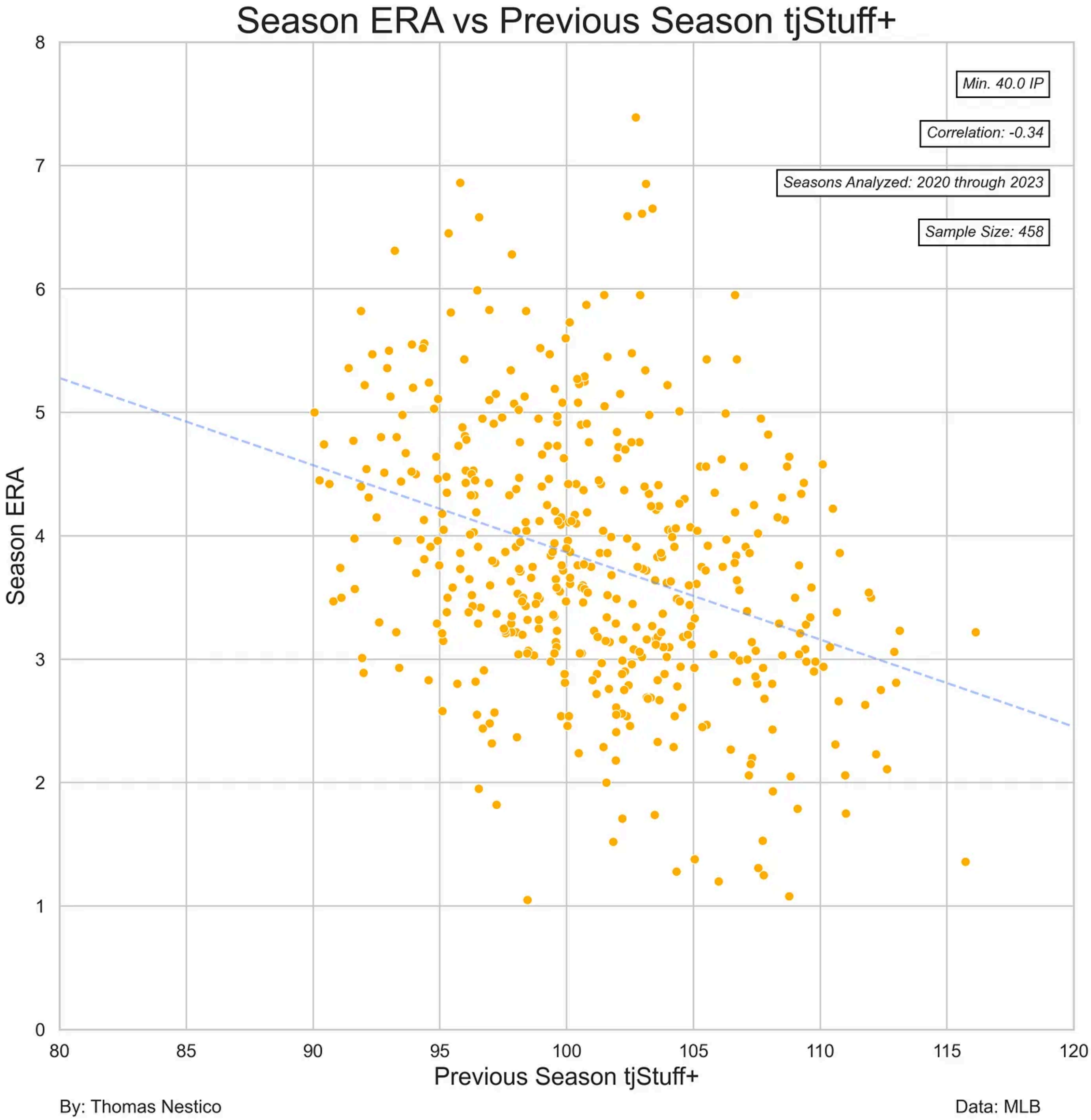


Figure 8: Season ERA vs Previous Season xFIP

The correlation of 0.29 between ERA and previous season xFIP indicates that xFIP is a better predictive statistic than simply using previous season ERA, however, it is still not a strong predictor.



The relationship between current season ERA and previous Season Stuff+ is illustrated in Figure 9. The correlation between ERA and previous season tjStuff+ is -0.34. It is important to note that the absolute value of this correlation is greater than the 0.22 when simply looking at ERA and also the 0.29 from xFIP. This indicates that previous season tjStuff+ does a better job as a predictive metric than previous season ERA and xFIP.

Stickiness

An important aspect of a predictive statistic is that it is “sticky” year over year. Stickiness is the property of a statistic to exhibit consistency over time. We can calculate the statistics stickiness by calculating the coefficient of deterimnation (R2) of the statistics between two consecutive seasons. For this example, we are looking at 2022 tjStuff+ vs 2023 tjStuff+, which is illustrated in Figure 10.

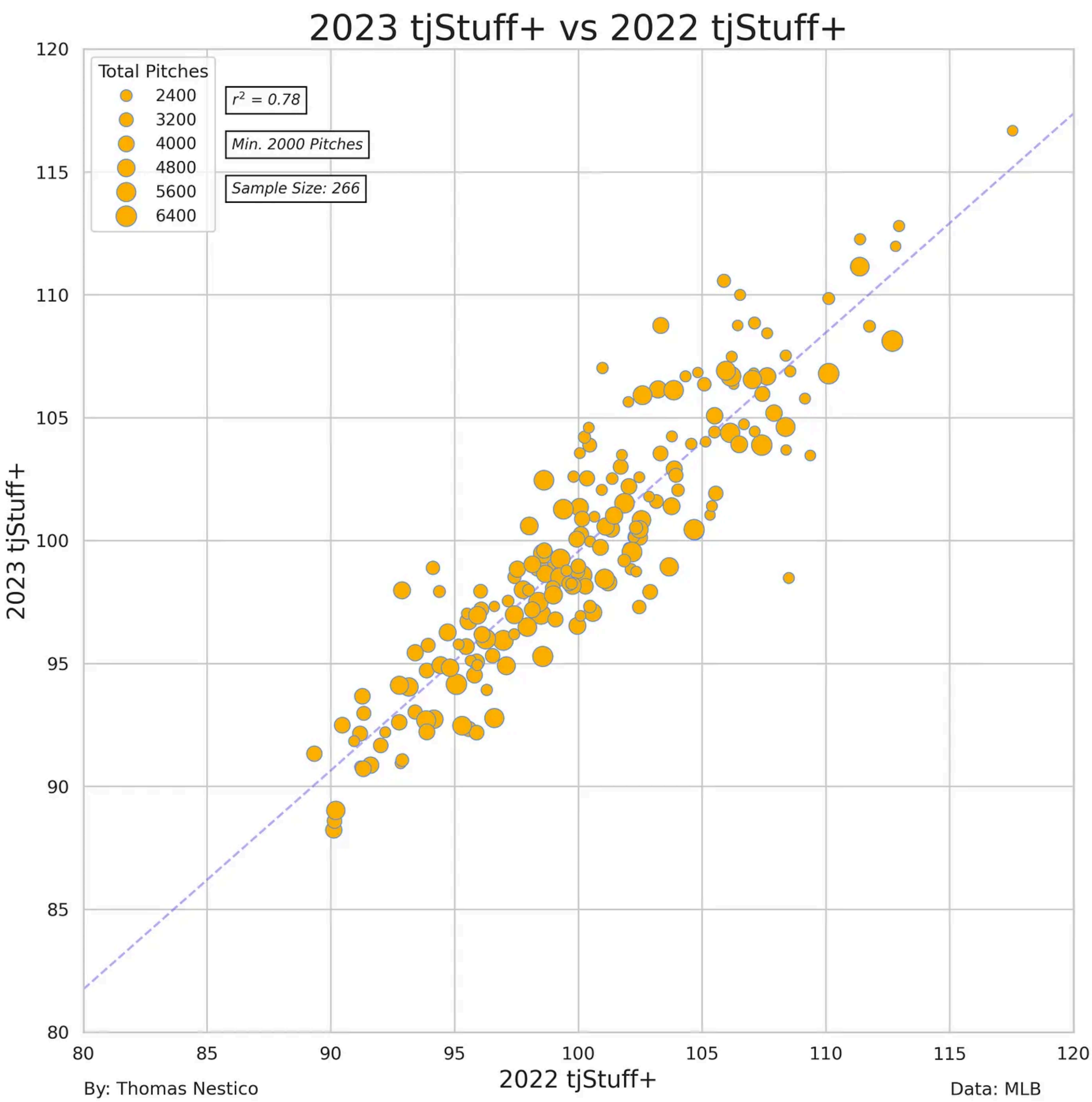


Figure 10: 2023 tjStuff+ vs 2022 tjStuff+

From 2022 to 2023, tjStuff+ has an R2 of 0.78. This value indicates that tjStuff+ is a very sticky statistic. The stickiness of tjStuff+ is desirable as it means that a player likely to attain a similar tjStuff+ in consecutive seasons, which supports the use of tjStuff+ as a predictive statistic.

Results

This is a link to a spreadsheet with the tjStuff+ Leaderboards for the 2023 MLB season

Notes:

- tjStuff+ is a metric which calculates the Expected Run Value (xRV) of a pitch
- tjStuff+ normally distriubted, where 100 is considered average, and the Standard Deviation is 10.
- tjStuff+ is provided for individual pitch types (min. 10 pitches), as well as the pitcher as a whole (min. 10 pitches)

Figure 11 is a plot illustrating the distribution of tjStuff+ during the 2023 season.

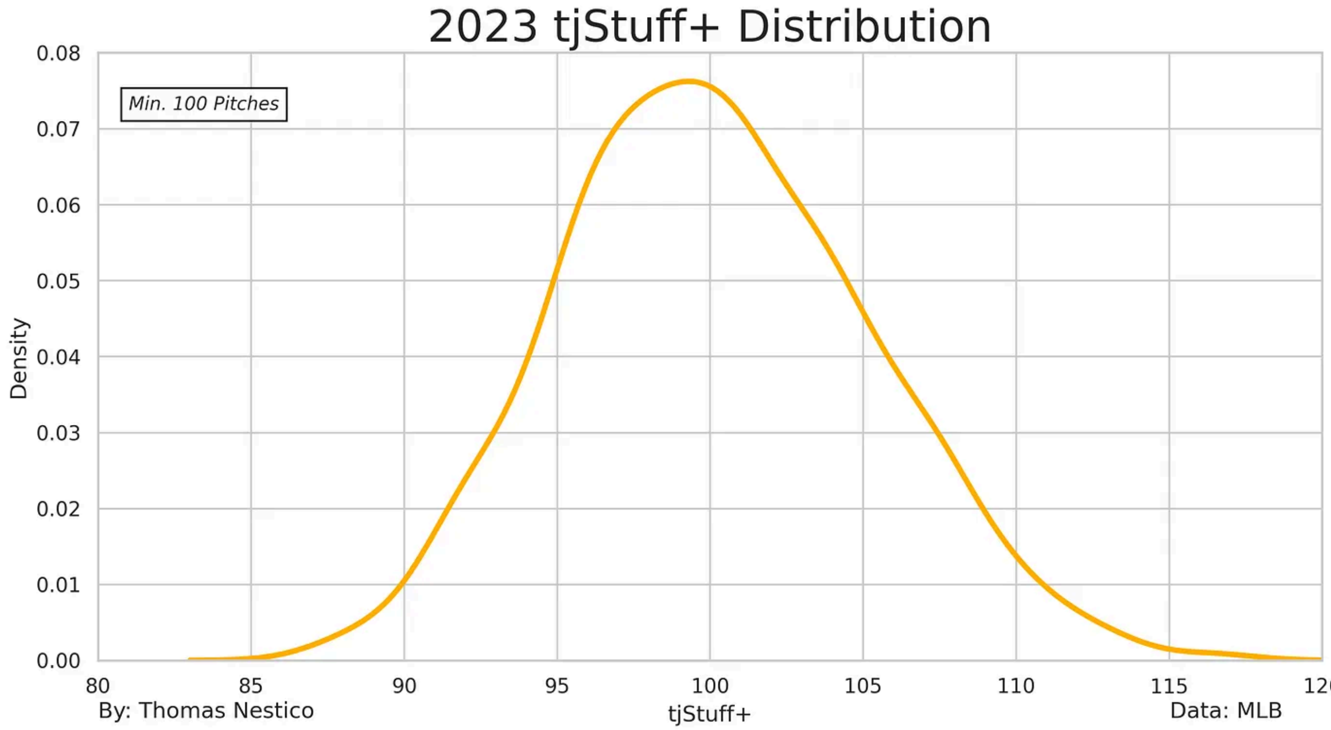


Figure 11: 2023 tjStuff+ Distribution

Figure 12 is a plot illustrates the distribution of tjStuff+ for pitch types during the 2023 season.

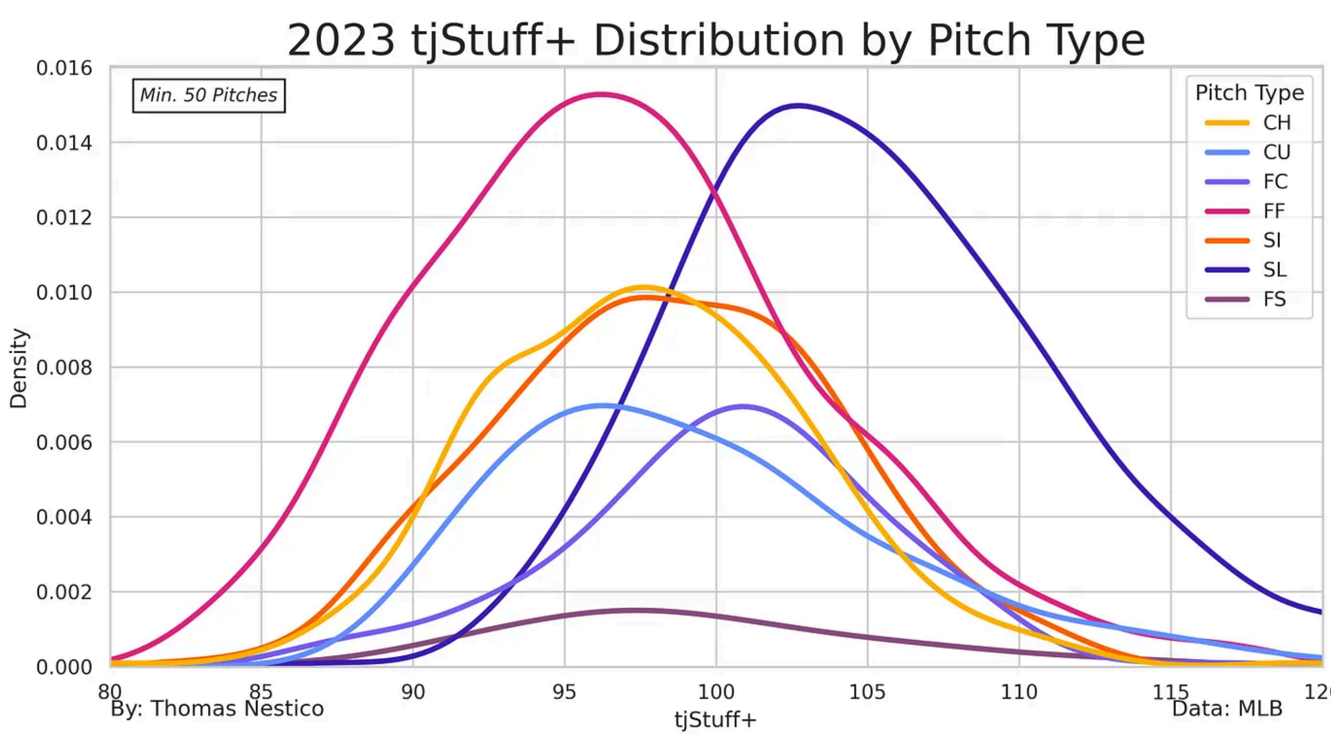


Figure 12: 2023 tjStuff+ Distribution by Pitch Type

Limitations

When creating a model that looks at a pitch's physical characteristics, a lot of context is lost in each event. For example, pitch location is not considered while training, despite pitch location being a very large driver in pitch outcomes. Another aspect which is not captured is the importance of both sequencing and deception.

Sequencing

In figure 12 it is apparent sliders generally have a greater tjStuff+ than other pitches. This discrepancy may stem from a multitude of factors, but a large one would be the situational usage of sliders. Figure 12 shows how the usage of sliders changes depending on the count.

		Balls			
		0	1	2	3
Strikes	0	21%	20%	15%	3%
	1	24%	23%	20%	12%
	2	28%	27%	25%	21%

Right away, we see that slider usage increases when there are more strikes, and more importantly, increases when the batter is in a "pitcher's" count. When outlining the target for this project, we saw how a pitcher inducing a strike with 2 strikes is more impactful than in other scenarios. Combine a slider's innate ability to induce swinging strikes with their usage in strikeout-potential situations, sliders tend to grade out very favourably in pitch models.

So what does this mean? Sliders are likely being overvalued by tjStuff+ due to the limitations imposed on the model. Without contextual information such as count and previous pitches thrown, the model cannot capture how impactful sequencing is when it comes to evaluating a pitch. Implementing these features may be the next step as I continue to refine my methodology.

Conclusion

Pitch modelling is a fascinating concept which continues to intrigue me. Understanding what makes a pitch effective is important while analyzing a pitcher. After all, a pitcher is only as good as the pitches they throw. Creating a model which quantified the effectiveness of a pitch was a very interesting task which challenged my problem-solving skills and allowed me to dig deeper into the factors that drive pitch effectiveness. Quantifying pitch effectiveness using only physical pitch characteristics strips back a lot of context, but it helps to capture what makes a "good" pitch and summarizes which pitchers are the most prolific at throwing competitive pitches.

Baseball Machine Learning Pitching

T

Written by Thomas Nestico

109 followers · 5 following

Follow

Responses (3)

Write a response

What are your thoughts?

Syedarizvi
Jul 8, 2024

Absolutely fascinating breakdown of pitch mechanics and the modeling process! It's incredible how much depth there is in understanding pitch movement and effectiveness. Your use of physical characteristics to build a machine learning model for... more

Reply

Steven Ruby
Feb 26, 2024

love this stuff.. the tjStuff+ name still makes my elbow hurt though.

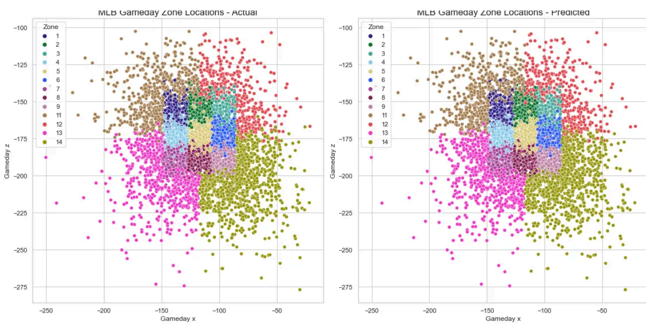
Reply

Caldwellgluesing
Jan 22, 2024

This is fantastic work! Following along on Twitter has been a great learning experience. The fb_max metrics totally blew my mind and it's so cool to see the results of your work! Thanks for sharing

1 reply Reply

More from Thomas Nestico

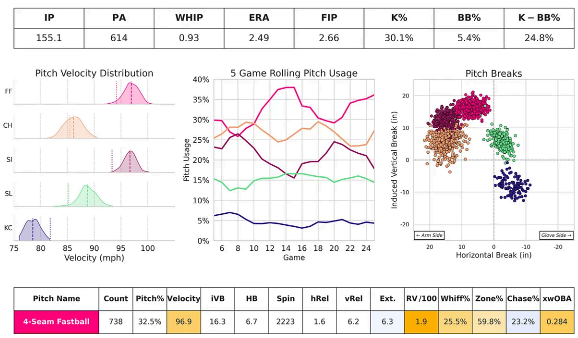


Thomas Nestico

Classifying MLB Pitch Zones and Predicting MiLB Zones

Statcast Pitch Classification

Nov 17, 2023 9

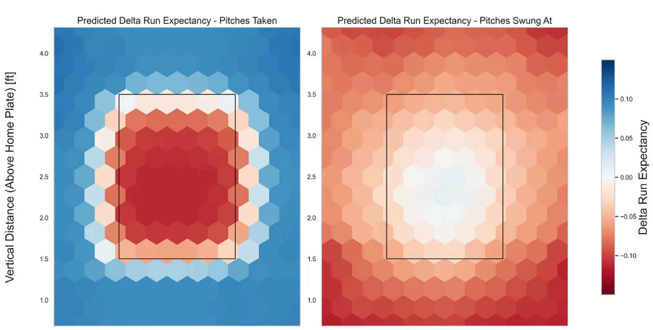


Thomas Nestico

Creating the Perfect Pitching Summary

Introduction

Aug 22, 2024 18 2

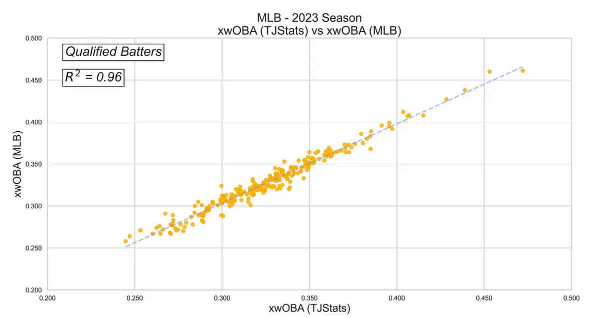


Thomas Nestico

Modelling Batter Decision Value

Introduction

Dec 1, 2023 11



Thomas Nestico

Modelling xwOBA (With KNN)

Introduction

Feb 13, 2024 54 1

See all from Thomas Nestico

Recommended from Medium

Kyle Jones

Tracking Houston's Green Cover with Satellite NDVI using Remote...

Change often happens in ways we can't quite see—until we pull back far enough. From hig...

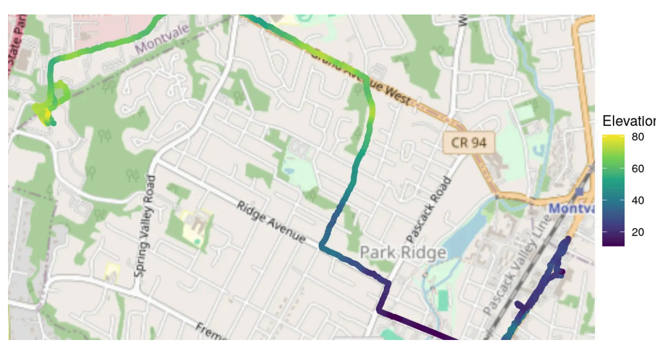
Apr 21 81

Trimmbor

sleep deprived musings from an elderly retirement home

As I lay in my ornate rich deep burgundy rotting mahogany wheelchair, sipping on my...

Sep 10



Nicole Venglovicova

Hockey Champions League Predictions — 7 September 2025

Get the stats, picks, predictions and analysis

Sep 6 29

Brian Carey

GPS Mapping with R

A comparison with Python

Mar 29 1

In Long. Sweet. Valuable. by Ossai Chinedum

I'll Instantly Know You Used Chat Gpt If I See This

Trust me you're not as slick as you think

May 16 24K 1433

kaiku

Anomaly Detection

Anomaly Detection Using Lime (Code Focus)

May 22

See more recommendations