



SEI x Phillies Hackathon 2024

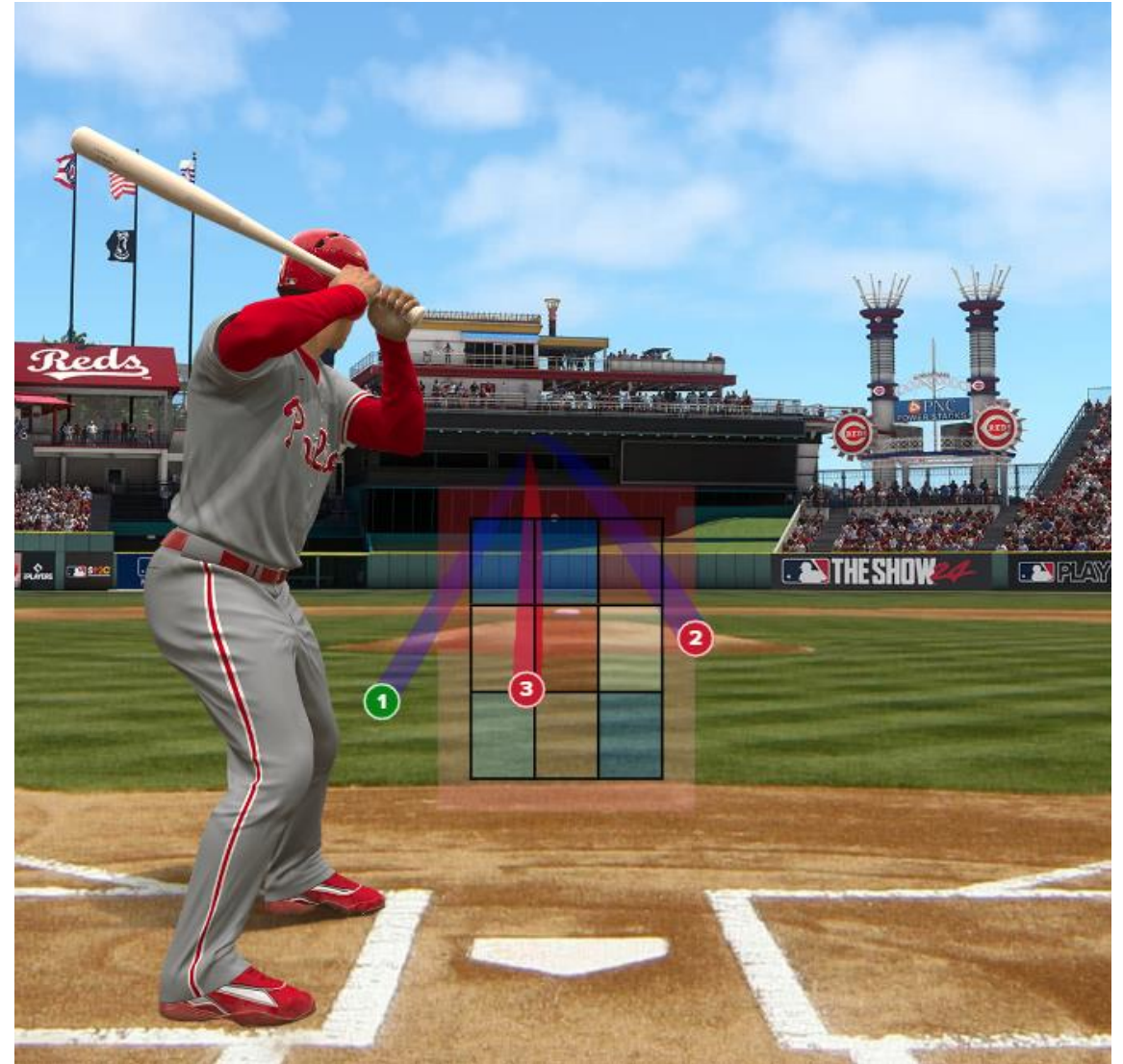
Villanova Sports Analytics Club

Stephen Cain, Amélie Devine, Chris
Galgano, Colin Hofmeister, Michael
Southwick



Problem/Prompt

- Develop a model to estimate the probability that a taken pitch will be called a strike
- Using strike probability model, develop a product to display predicted called strike probabilities



Objective/Early Thoughts

- Given pitch-by-pitch data from the 2022 MLB Season
- First objective was to model the probability that a pitch would be called a strike based on location, type of pitch, the pitch's releases speed, and game situation (count, number of outs, inning)
- Building out initial model to include various factors, such as catcher framing and umpire tendencies to make accurate adjustments



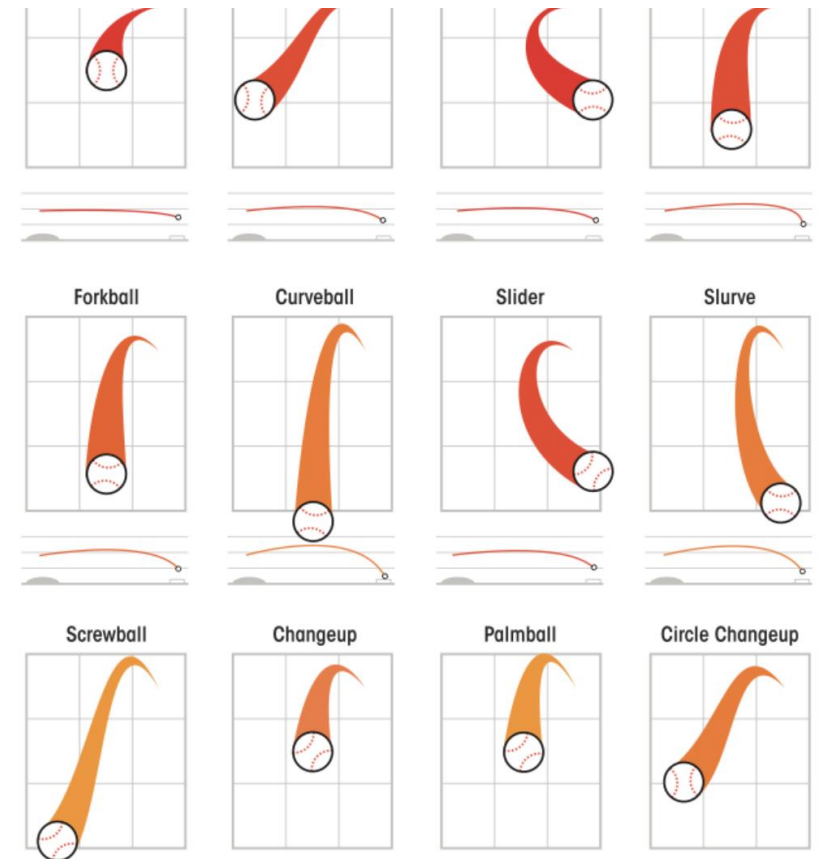


Data Processing & Cleaning



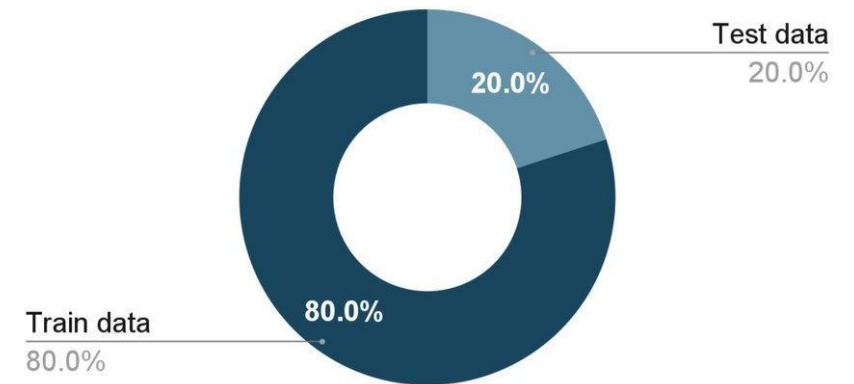
Current Data

- Filtered necessary columns of data from dataset (ex. IDs of the pitcher, batter, catcher, pitch result, 3D coordinates of the pitch)
- Created new variables to improve model's predictive power (ex. distance of each pitch from top and bottom of strike zone)
- Converted categorical/character-defined variables into binary representations (ex. delineate fastball vs breaking ball vs off-speed)



Additional Data

- Appended catcher and umpire data (ex. games played, umpire accuracy) for small sample baselines
- Divided data into two parts: 80% into training partition and 20% into testing partition



Additional Data Sourcing

- Baseball Savant pitch data from 2022 and 2021 to help create a predictive number for catcher framing
- Player ID data set to pair IDs with player names from Smart Fantasy Baseball
- 2022 Umpire Data from Umpire Scorecards to gather each umpire's mean accuracy



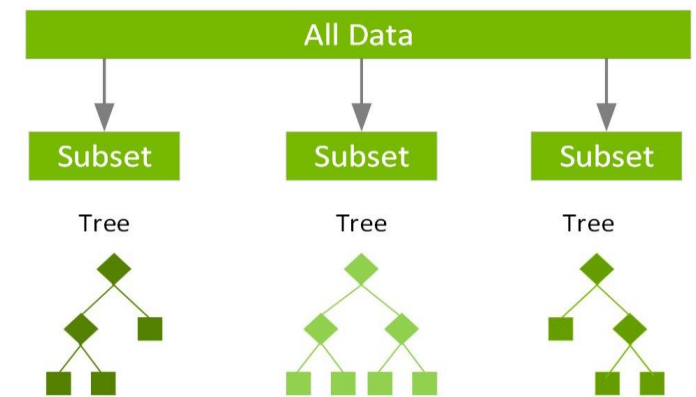


Modeling



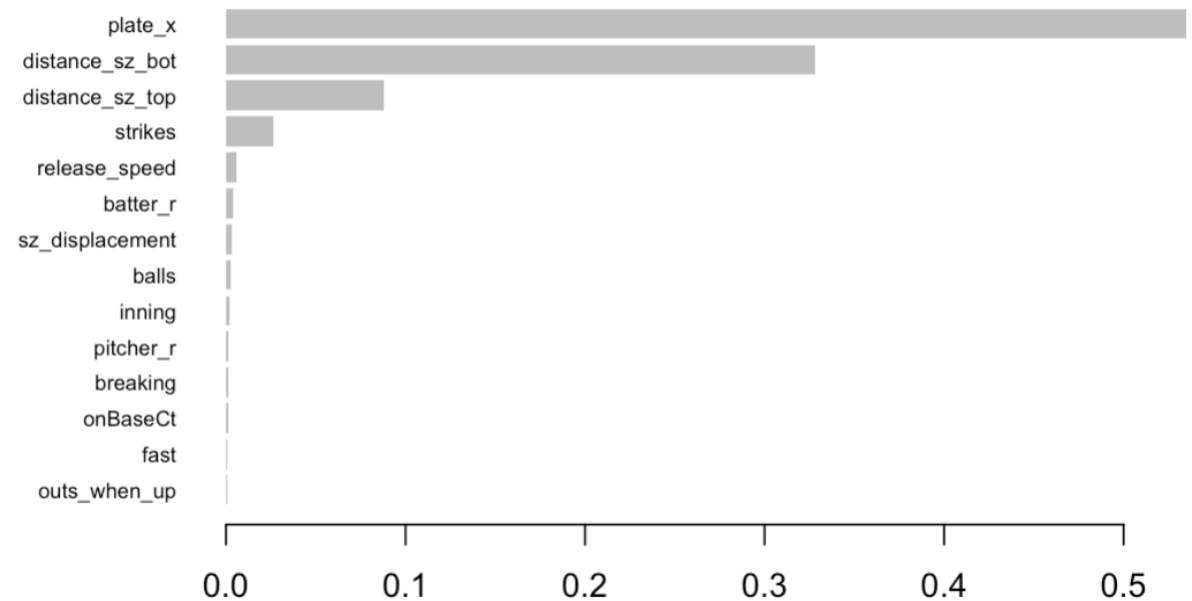
Methodology

- Used XGBoost Model in R to model probability of a pitch being called a strike
 - Powerful machine learning algorithm used for regression that builds a strong predictive model through decision trees
 - Decision trees take in all relevant data and predict whether the pitch will be a strike or not
 - Initially, the model makes mistakes, but each next decision tree learns from the previous one, and becomes a little more accurate each iteration at predicting whether the pitch was a strike or not
 - This process is repeated until the model minimizes what is known as a "log loss" function, lower "log loss" indicates that the model is very accurate



Details: Initial Model

- **Target Value:** Strike Probability
- Initial XGBoost Model had the following factors:
 - Horizontal location of the pitch
 - Vertical distance from top/bottom of hitter's median strike zone
 - # of balls/strikes prior to pitch
 - Handedness of batter/pitcher
 - Pitch type and velocity
 - Game state (inning, runners on base, outs)



Details: Improvements

- Used initial model to create catcher framing statistic
 - Used relationship between expected and actual strikes to determine positive framing impact
- Final model factors:
 - All initial model factors
 - Catcher Framing
 - Umpire Accuracy
- Final model had a test set accuracy of about 93% and an AUC of .9818

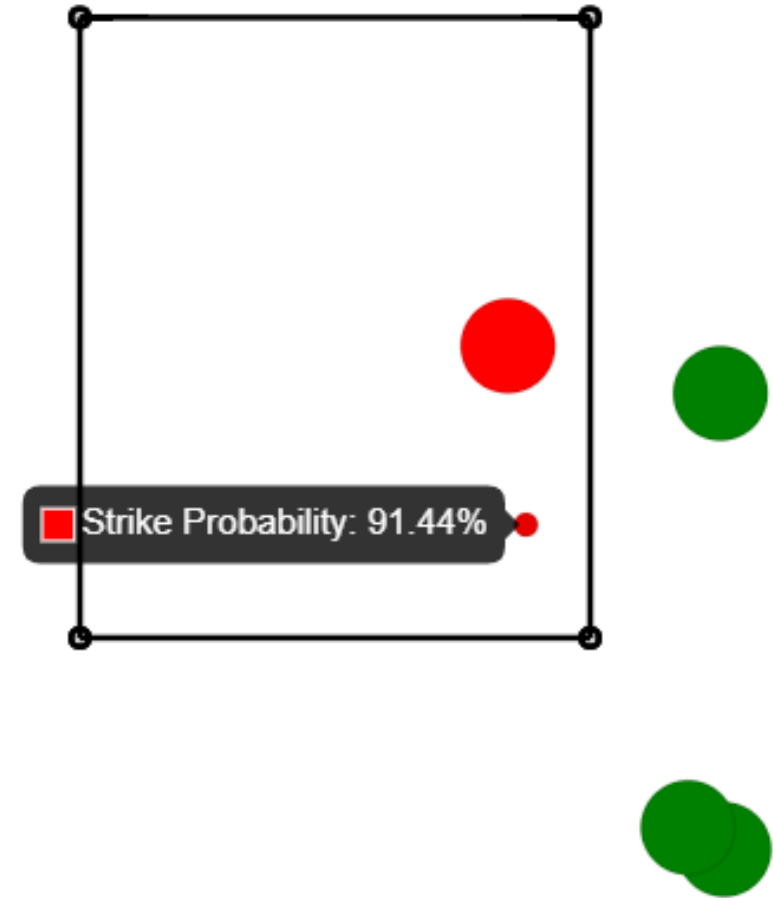
	▲ PLAYERNAME ▲	StkAx ▲
1	Jose Trevino	146.35792
2	Jonah Heim	122.79662
3	Max Stassi	119.99834
4	J.T. Realmuto	119.97990
5	Jonah Heim	102.43092
6	Tucker Barnhart	99.39381
7	Mike Zunino	94.62138
8	Sean Murphy	84.46384
9	Cal Raleigh	82.32321
10	Omar Narvaez	78.70770

Applications



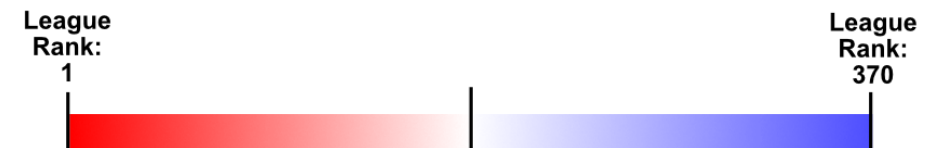
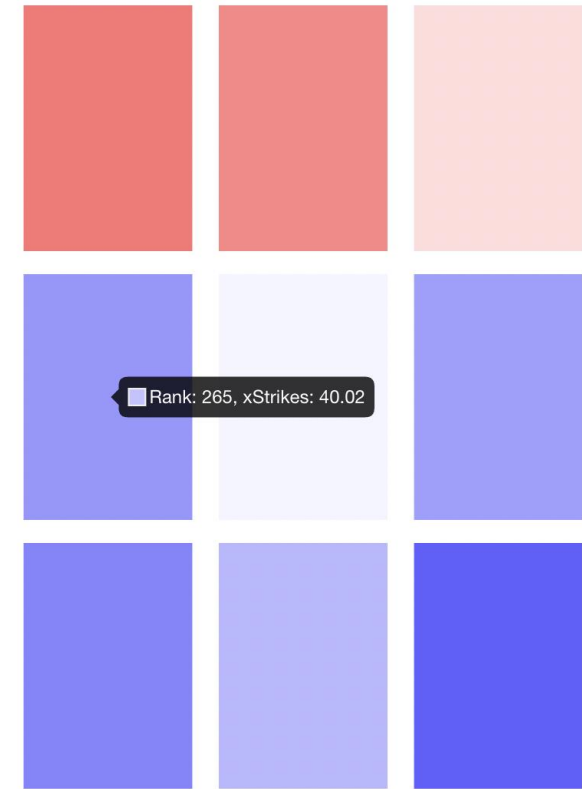
Strike Probability Visualization

- User chooses Pitcher, Game Date, Batter information and receives a graphic that displays the hitter's personalized strike zone, called strikes (red), called balls (green)
- Hovering over any of the pitches produces the strike probability based on our model
- Clicking on a pitch opens a link to the video of the pitch being thrown



Batter xStrikes Heatmap

- User chooses Batter and receives a heatmap that breaks that batter's pitches faced into 9 zones
- Also outputs league rank based on Expected Strikes
- The lower the rank the fewer expected strikes in that zone



Specific Applications

Batters

- Insight into a hitter's swing decisions
- Use expected strikes to adjust swing decisions based on strike probability trends broken up by zone

Pitchers

- Can help improve what type of pitch to throw in certain situations
- Can improve location decisions based on batters faced

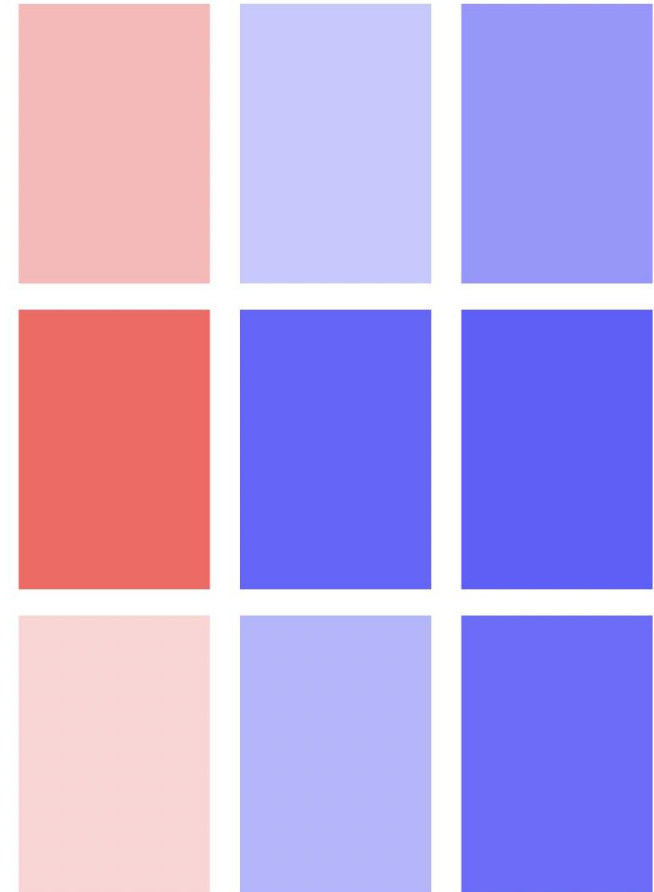


Example

Key: Players with imbalanced zones

Gleyber Torres

- 82nd in xStrks for the middle inside pitch (middle left quadrant)
- 350th for the middle away pitch (middle right quadrant)
- 245th for middle middle pitch (middle center quadrant)



Possible Improvements

Use a larger sample of pitches from other seasons for training the model

Having minor league data to better evaluate catcher framing for young/low MLB sample catchers

Breaking out umpire call tendencies into different zones

Evaluating pitch movement to see if it has meaningful impact on strike probability

Main Takeaways

XGBoost Model to predict strike probability

Most significant variables are vertical and horizontal pitch location

of strikes in count = next most important factor

Display the results of model through heatmap & plots

Our model and website together can provide insights towards catcher/umpire performance & help batters/pitchers make smarter decisions

Appendix



Model Parameters

```
#parameters for the XGBoost model
```

```
params <- list(  
  booster = "gbtree",  
  objective = "binary:logistic",  
  eta = 0.3,  
  gamma = 0,  
  max_depth = 3,  
  min_child_weight = 1,  
  subsample = 0.6,  
  colsample_bytree = 0.6  
)
```

```
#training the model
```

```
xgb_model <- xgb.train(params = params, data = dtrain, nrounds = 40,  
                      watchlist = list(eval = dtest, train = dtrain),  
                      early_stopping_rounds = 10)
```


Databases Utilized

```
` `` {r}  
df <- read.csv("k_prob.csv")  
df_detail <- read.csv("All_2022_Pitches.csv")  
df_detail_2021 <- read.csv("All_2021_Pitches.csv")  
catcher_info <- read.csv("SFBB_Player_ID_Map.csv")  
ump_data <- read.csv("2022_Umpire.csv")  
` ``
```

Libraries Utilized

```
` `` {r}  
library(tidyverse)  
library(readr)  
library(caret)  
library(xgboost)  
library(ROCR)  
library(lubridate)  
library(Lahman)  
library(dplyr)  
` ``
```