Predicting Umpire Error in Baseball

John Russell
Liam Taylor
Radvilas Pelanis
Roshen Chatwal

CS109a Final Project

Harvard John A. Paulson School of Engineering and Applied Sciences

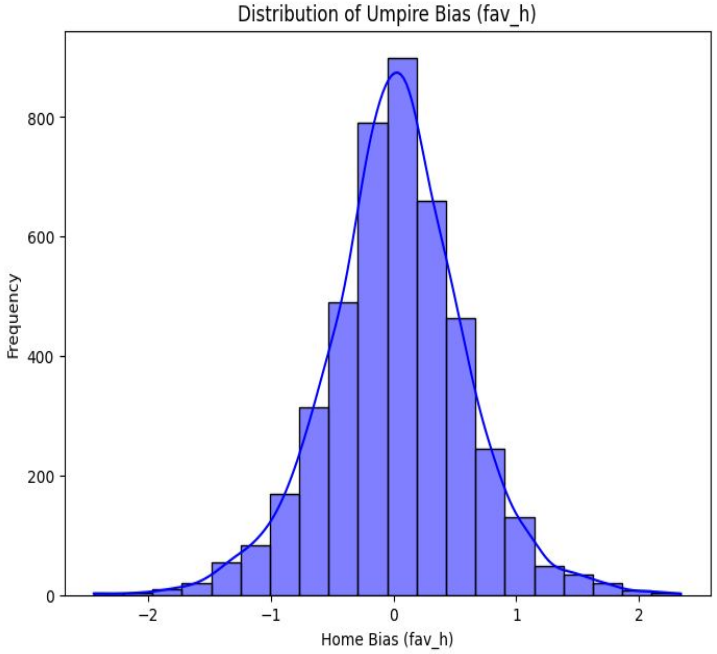
Problem Statement

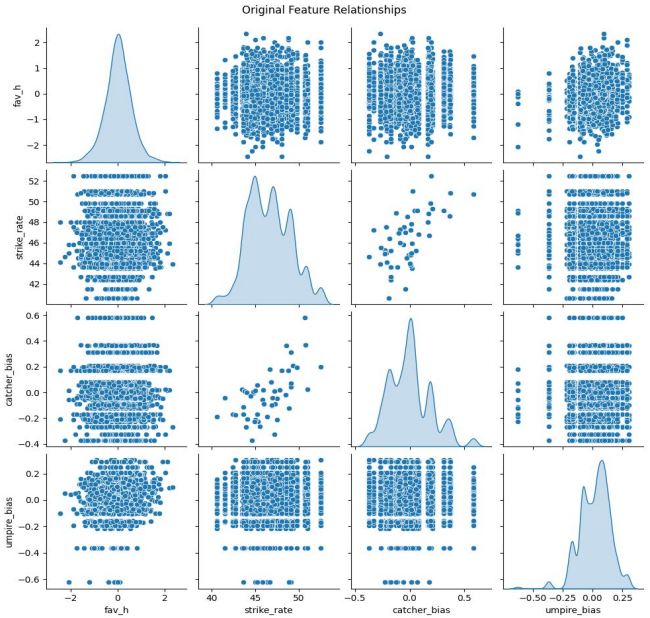
In this project, we attempt to determine the association of various factors (regarding the pitcher, catcher, and umpire) with umpire bias, and to develop a regression model which can predict the score impact of incorrect calls in a game based on the pitching, catching, and umpiring metrics from the game.

Research Questions

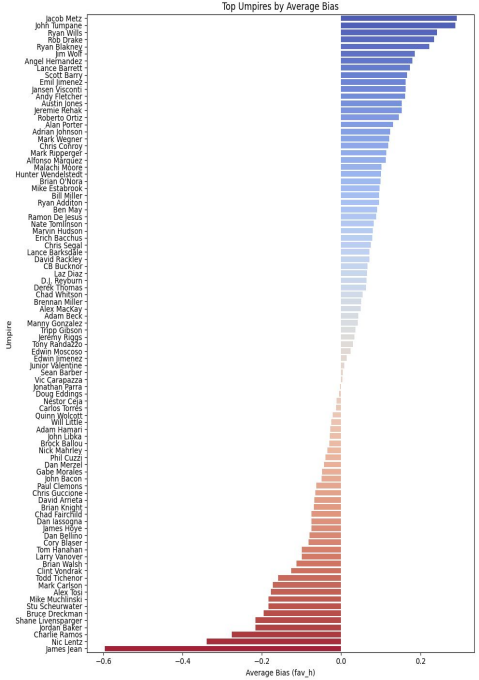
1. Based on past pitcher, catcher, and umpire data, can we effectively predict the projected impact of umpire error and pitch-framing in game outcomes?
2. Which factors have the biggest effect on umpire calls?

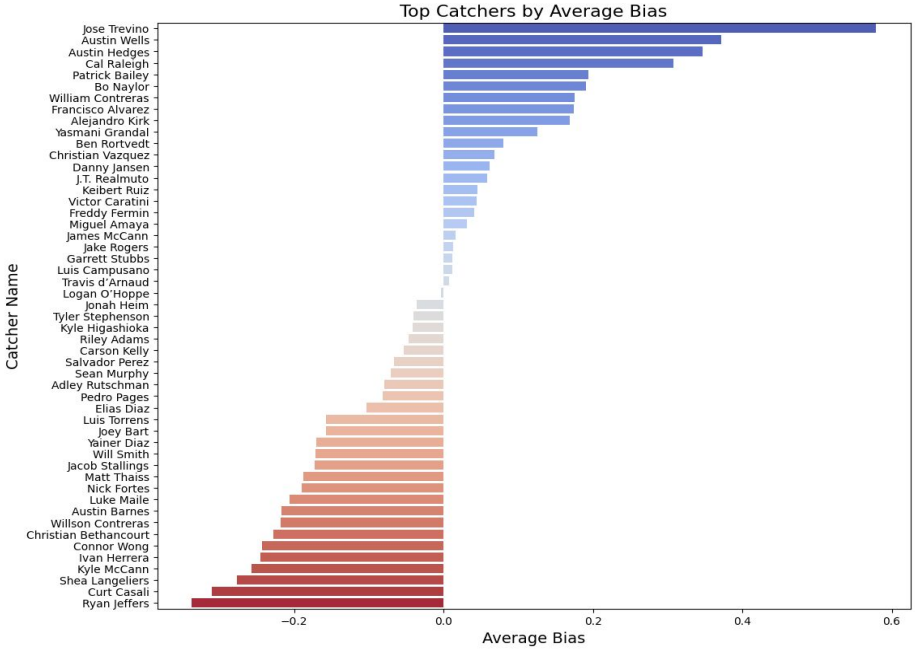
Exploratory Data Analysis (1)





Exploratory Data Analysis (2)





Final Model Pipeline

Our **predictor** variables are the following: 'catcher_bias', 'umpire_bias', 'interaction_home_umpire', 'strike_rate', 'stand_L' , 'stand_R', 'p_throws_L', 'p_throws_R'.

Our **target** variable: 'fav_h'.

We engineer **polynomial features** by setting degree equal to 2, implementing a **StandardScaler**, and defining our test_train_split such that our **test size is 0.3**, and random state is 42.

Baseline Model

Goal: to predict the average target value (fav_h) for all games. This will serve as a benchmark to understand if engineering features and more advanced models can improve predictive accuracy in the first place.

We use MSE and R^2 to measure baseline performance:

- test_**MSE = 0.3501**.
- train_**MSE = 0.3034.**
- **R^2 = -0.0007.**

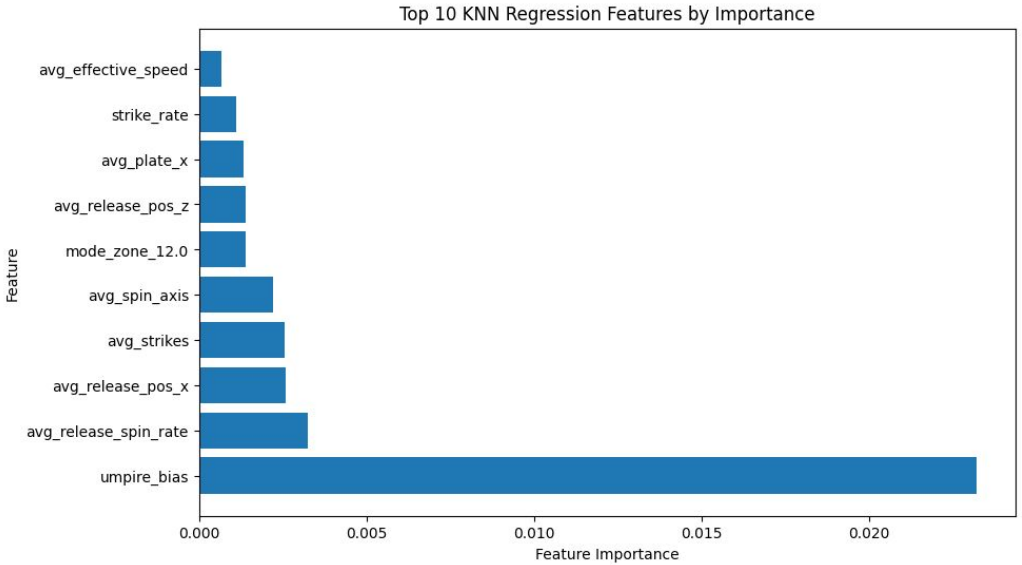
Model: kNN Regression with CV

For kNN regression, we test K values from 1 to 80, with K=67 identified as optimal.

At such model, we obtain:

- train_**MSE=0.2861**
- test_**MSE=0.3451**

Both of which show improvement compared to baseline model.

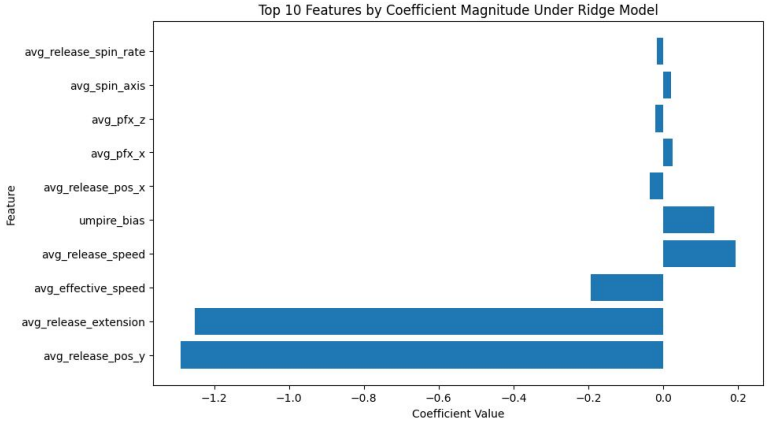


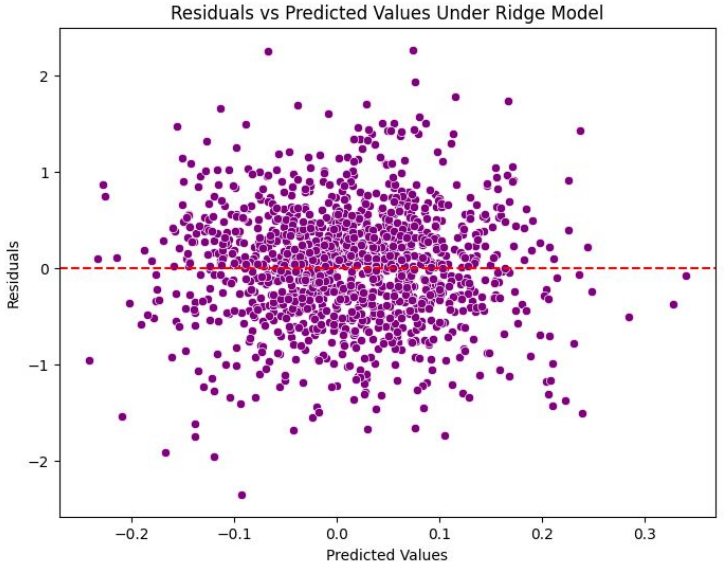
Training Time (with CV): < 1 min

Model: Ridge with CV

The Ridge regression cross-validation identified an optimal alpha of 0.01, minimizing mean cross-validated MSE. The **train MSE is 0.2825**, while **test MSE is 0.3406**, both an improvement compared to kNN Regression. However, the low **R^2 value (0.0256)** suggests the model explains very little variance in the test data.

Training Time (with CV): < 1 min

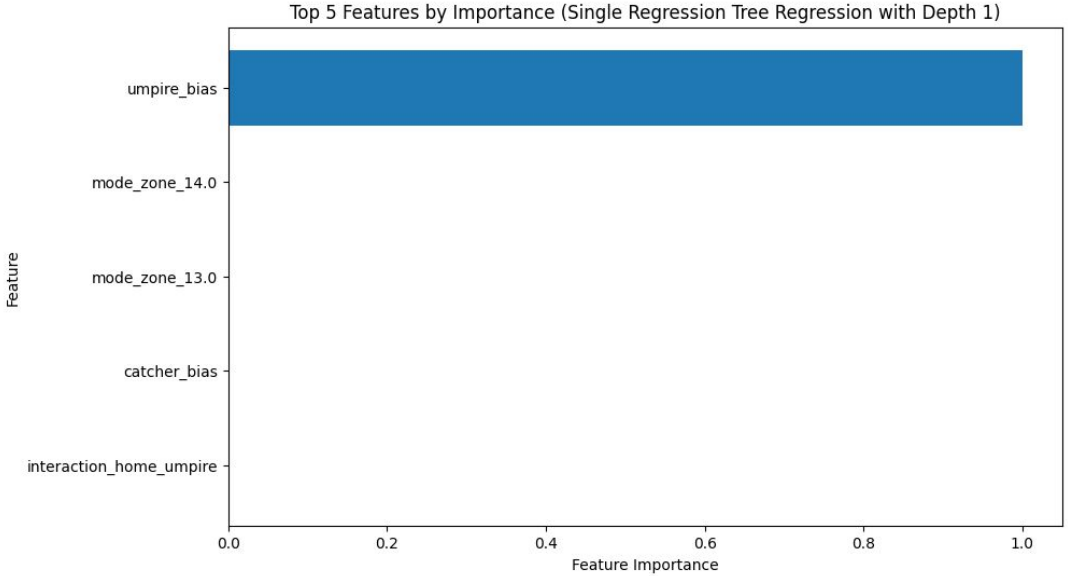




lsor
ing
s

Model: Regression Decision Tree

We run the model for max_depth from 1 to 9, identifying the optimal depth = 1. **Train MSE = 0.2887**, **Test MSE = 0.3527, Test R^2 = -0.008**, identifying a better performance in train but not test datasets compared to Ridge Regression.

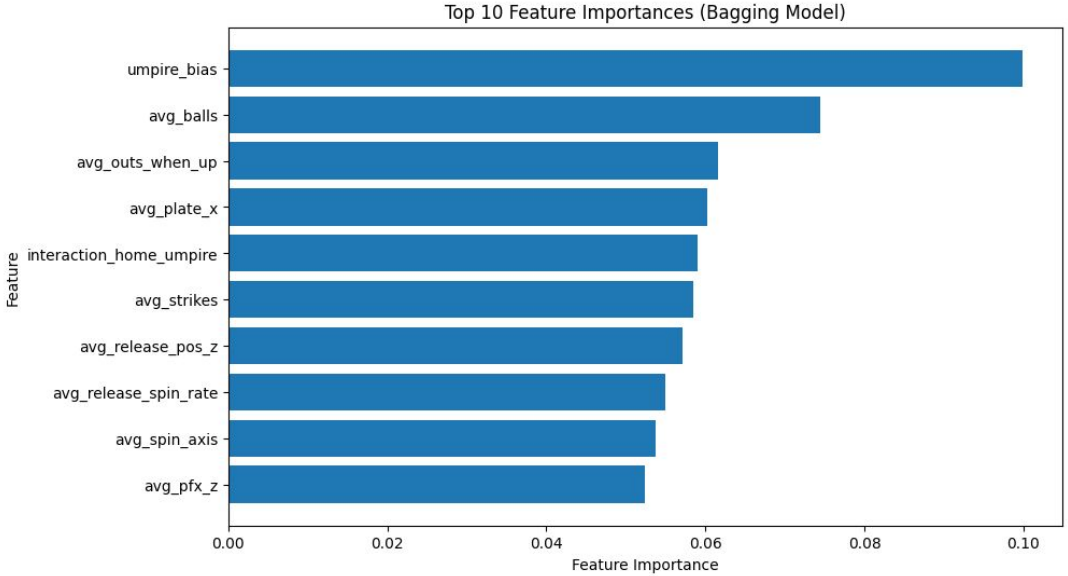


Training Time (with CV): < 1 min

Model: Bagging

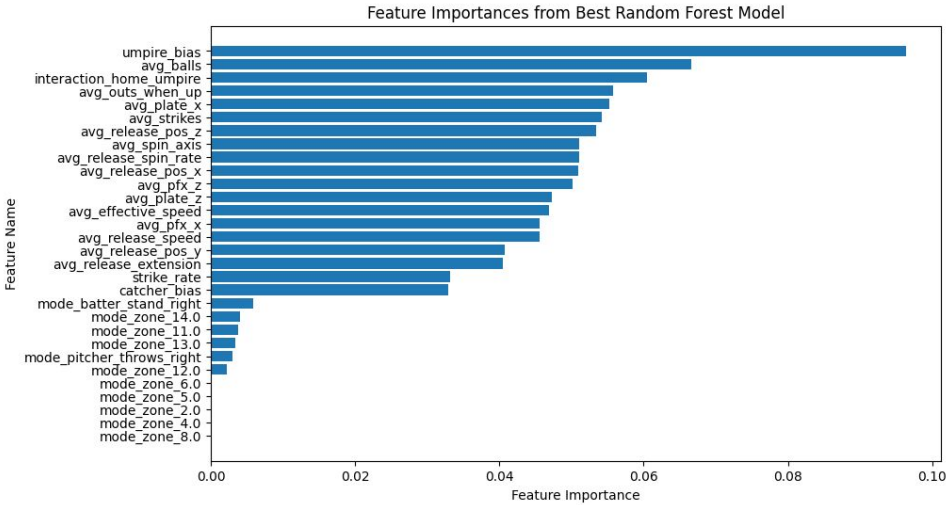
For bagging, we determine our optimal number of estimators to be 300, optimal subtree depth to be 35, and **train MSE = 0.0379, test MSE = 0.3315**, while our **R^2 = 0.0525.**

Training Time (with parameter grid): 5-10 min



Model: Random Forest

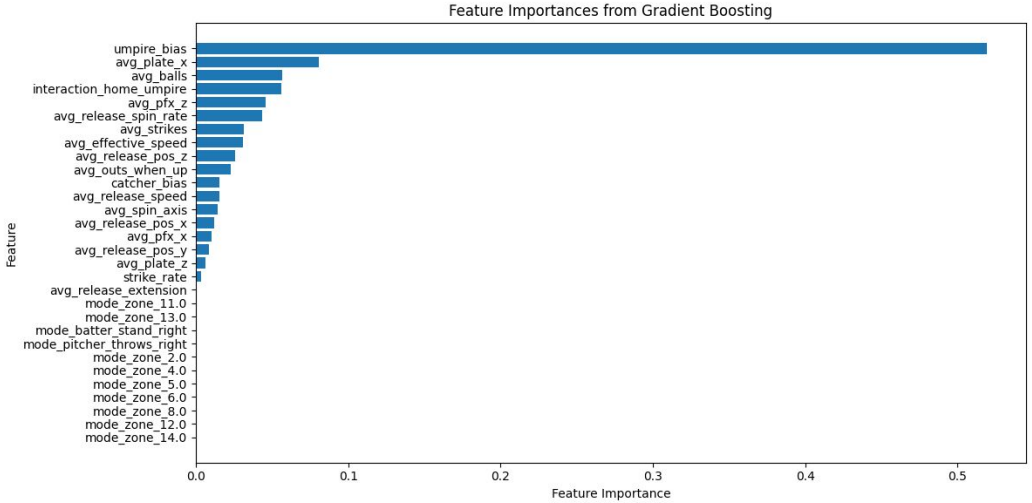
Trying out different hyperparameters such as max_depth=[15, 20,...], n_estimators = [50, 100, 200,...], yields the most optimal parameters (MSE-wise) to be max_depth=40, n_estimators = 250, resulting in most optimal **train MSE = 0.0377**, **test MSE = 0.3291**, and **R^2 = 0.0594.**



Training Time (with parameter grid): 5-10 min

Model: Boosting

Searching for most optimal hyperparameters, we find best max_depth=2, n_estimators=550, learning_rate=0.01 which result in **train_MSE=0.2687, test_MSE=0.3468, R^2 = 0.0089.**



Training Time (with parameter grid): > 20 min

Results & Conclusion: Random Forest!

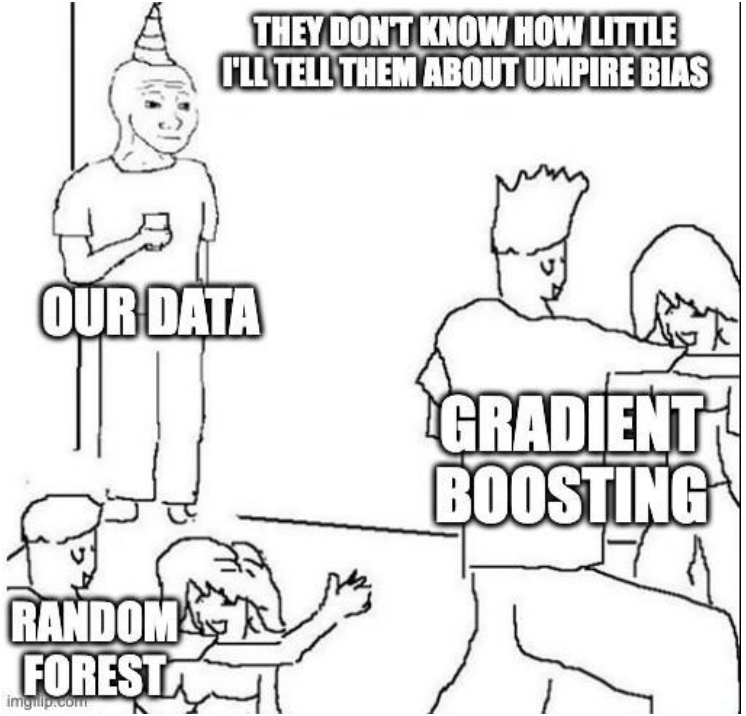
Model Performance Metrics

model_name	train_mse	test_mse	test_rmse	r-squared
Random Forest (subsets of n/3 features at each split, max depth 40, 250 estimators)	0.037743	0.329090	0.573663	0.059437
Bagging (subtree depth 35, 300 estimators)	0.037888	0.331524	0.575781	0.052481
Ridge Regression (Alpha = 0.01)	0.282485	0.340598	0.583607	0.026547
KNN (67 neighbors)	0.286081	0.345100	0.587452	0.013680
Gradient Boosting (550 iter, base learner depth 2, learning rate 0.01)	0.268687	0.346768	0.588870	0.008911
Naive Model (arithmetic average)	0.303358	0.350131	0.591718	-0.000699
Single Regression Tree (Depth = 1)	0.288750	0.352695	0.593882	-0.008029

Future Work

- Finding predictors more indicative of bias in baseball umpiring
  - Weather? Different pitch data?
- Learning and implementing better model types
- Researching the benefits of integrating robot umps in MLB as opposed to human umps (robo-umps already exist in the minors)

Thank you!


THEY DON'T KNOW HOW LITTLE
I'LL TELL THEM ABOUT UMPIRE BIAS
OUR DATA
GRADIENT
BOOSTING
RANDOM
FOREST
imgflip.com