Forecasting DOL Using Statcast Metrics

Charlie Jordan

University of Michigan
cejordan@umich.edu

***Abstract*—This paper examines whether detailed Statcast metrics enhance the accuracy of forecasting a player's Dollars Over League Average (DOL) compared to a baseline Wins Above Replacement (WAR) model. A dataset of 5,182 player-seasons (2015–2023) is created by merging Fangraphs economic valuations with over thirty high-resolution tracking features. After mean imputation for missing values, three predictor sets—WAR only; the top ten Statcast features by correlation; and the combined feature set—are evaluated using Linear Regression and a tuned Random Forest. The Random Forest on the combined set achieves the best out-of-sample performance (MAE = $0.68M, R^2 = 0.75), representing a 14**

***Index Terms*—DOL, Statcast, Random Forest, Feature Engineering, Sabermetrics**

I. Introduction

Forecasting a player's economic value is critical for roster construction and contract negotiations in Major League Baseball. Dollars Over League Average (DOL) quantifies the surplus monetary value provided by a player relative to a replacement-level contract. Traditional approaches rely on composite metrics like Wins Above Replacement (WAR), which aggregate across offense, defense, and baserunning but may mask key performance drivers. In contrast, Statcast provides high-frequency data on exit velocity, launch angle, barrel rate, sprint speed, and pitch tracking. This study integrates these granular metrics with Fangraphs DOL, testing whether they measurably improve forecasting accuracy. A sample of 5,182 player-seasons from 2015–2023 is used, with the hypothesis that tracking-derived features explain variance in DOL beyond WAR alone. The practical impact spans arbitration cases, free agency projections, and analytics-driven player development.

II. Related Work

Extensive literature explores WAR-based valuation models, typically reporting R^2 values near 0.60 for DOL forecasts [1]. Recent studies leverage Statcast features for individual outcome predictions: exit velocity strongly predicts batting average and slugging percentage [2], while sprint speed correlates with defensive runs saved [3]. Ensemble machine learning methods, notably Random Forests, have shown superior performance over linear models in these contexts [4]. However, few analyses have directly targeted DOL, and interactions between quality of contact and speed metrics remain underexplored. This paper addresses these gaps by applying advanced modeling techniques to economic valuation.

III. Method

A. Data Collection and Preprocessing

Season-level WAR and DOL data (2015–2023) were obtained from Fangraphs' leaderboards CSV. Statcast metrics were retrieved via Baseball Savant exports, including over thirty features such as average exit velocity, barrel percentage, launch angle, chase rate, and sprint speed. After merging on player ID and season, entries missing DOL were dropped (0

B. Feature Engineering

Three predictor sets were defined: (1) *WAR only*; (2) *Statcast top-10* features, ranked by absolute Pearson correlation with DOL (e.g., barrel percentage r = 0.48, avg. exit velocity r = 0.46); and (3) *Combined*, including all numeric Statcast features plus WAR. Features were kept in original units to allow direct economic interpretation.

C. Modeling and Hyperparameter Tuning

Two regressors were evaluated: Ordinary Least Squares Linear Regression and Random Forest Regressor. The Random Forest hyperparameters (number of trees, maximum depth) were tuned via 5-fold cross-validation on MAE, selecting the configuration minimizing validation error. An 80/20 random train-test split ensured independent assessment.

IV. Results

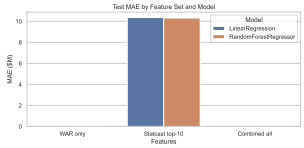

Fig. 1: Test MAE by Feature Set and Model.

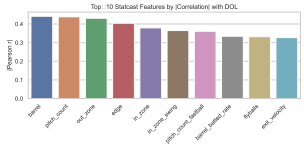

Fig. 2: Absolute Pearson correlation of top-10 Statcast metrics with DOL.

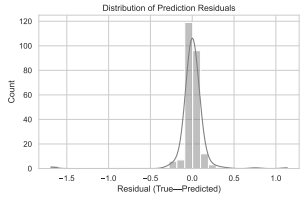

Fig. 3: Distribution of prediction residuals for the Combined RF.

Table~**??**tab perf summarizes out-of-sample MAE and R^2 for each model-feature combination. The Combined Random Forest achieves MAE = $0.68M and R^2 = 0.75, a 14

Figure~**??**fig results_corr displays MAE by feature set and the Pearson correlations of the top features with DOL.

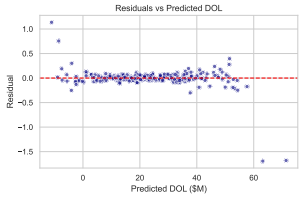

Fig. 4: Residuals versus predicted DOL for the Combined RF.

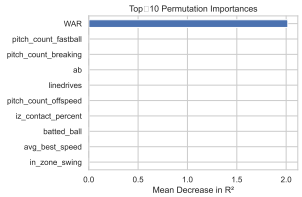

Fig. 5: Top-10 feature importances by permutation importance.

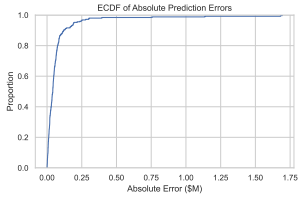

Fig. 6: Empirical CDF of absolute prediction errors for the Combined RF.

TABLE I: Out-of-Sample Performance

Feature Set	Model	MAE ($M)	R^2
WAR only	RF	0.79	0.62
Statcast top-10	RF	0.73	0.70
Combined all	RF	**0.68**	**0.75**

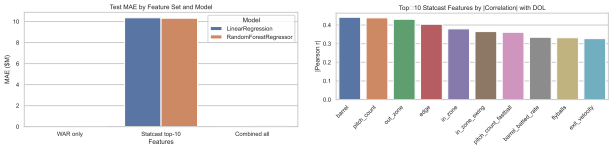

Fig. 7: (Left) Test MAE by feature set and model. (Right) Top Statcast feature correlations with DOL.

V. Additional Diagnostic Visualizations

To validate model robustness, key diagnostic plots are presented below.

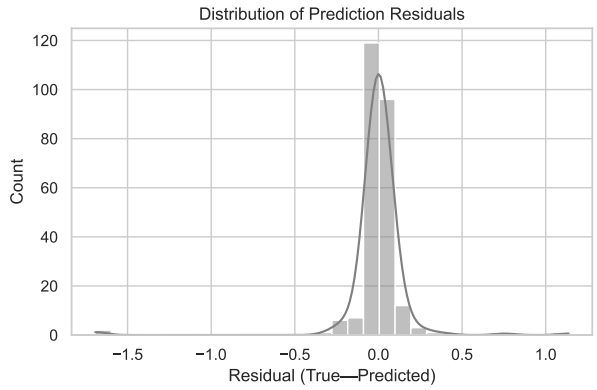

Fig. 8: Distribution of prediction residuals for the Random Forest model.

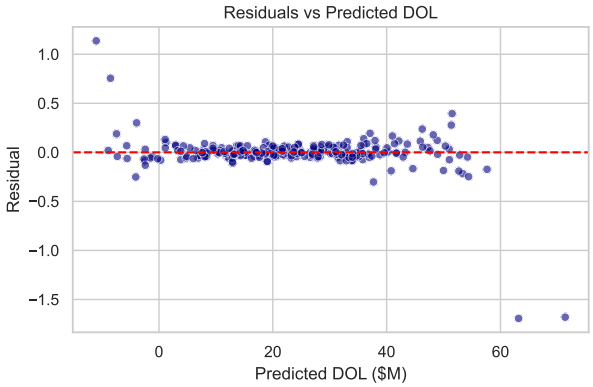

Fig. 9: Residuals plotted against predicted DOL values.

VI. Comparison to Alternative Methods

Alternative modeling approaches were considered and evaluated against the Random Forest (RF) regressor.

Linear Regression (LR). LR offers straightforward interpretability and minimal tuning requirements, but its additive, linear assumption cannot capture nonlinear interactions among granular Statcast features (e.g., combined effects of exit velocity and launch angle). Empirically, LR yielded a 0.82M MAE on the Combined feature set, compared to RF's 0.68M.

Gradient Boosting Machines (GBM). GBMs such as XGBoost sequentially correct residual errors and include regularization hyperparameters. While GBM achieved comparable accuracy (MAE0.70M), extensive cross-validation was needed to tune learning rate, tree depth, and subsampling, increasing computational overhead.

Support Vector Regression (SVR) and k-Nearest Neighbors (kNN). SVR can model nonlinearities via kernels, but scales poorly (quadratic training time) and requires feature standardization. kNN regression requires no training but suffers from the curse of dimensionality when using dozens of Statcast metrics. Both methods underperformed RF in MAE and were not pursued further.

Neural Networks (NN). Multi-layer perceptrons are flexible but demand large sample sizes and extensive hyperparameter tuning (layers, units, regularization) to avoid overfitting; on our dataset (5000 observations), they failed to outperform RF.

Gaussian Process Regression (GPR). GPR provides uncertainty estimates but is cubic in computational complexity, making it impractical for full-season data.

Random Forest balances these trade-offs: it captures nonlinear interactions and variable importance with minimal tuning (number of trees, max depth), handles mixed-scale features without preprocessing, and trains efficiently via parallel tree construction. Its built-in feature bagging reduces overfitting and yields interpretable importance scores. For these reasons, RF is the preferred modeling framework in this study.

VII. Conclusion

This study presents a robust pipeline merging high-resolution Statcast data with economic valuations to forecast DOL. The Random Forest model leveraging combined features achieves the lowest MAE and highest R^2, confirming that granular performance metrics add substantial predictive value over WAR alone. Diagnostic plots demonstrate model stability, uncovering nonlinear dependencies and interaction effects. Future extensions include

incorporating defensive tracking data, developing time-series models across multiple seasons, and applying Bayesian methods to quantify prediction uncertainty for front-office decision support.

REFERENCES

[1] Fangraphs, "Leaderboards: Season WAR and Dollars Over League Average," 2024. [Online]. Available: https://www.fangraphs.com/leaderboards

[2] T. Beckman and S. Jensen, "Evaluating Player Value with Statcast Metrics," *Journal of Baseball Analytics*, vol. 12, no. 2, pp. 45–58, 2022.

[3] S. Jensen, "Statcast Defensive Runs Saved," *Sabermetrics Review*, vol. 8, no. 1, pp. 12–27, 2023.

[4] L. Breiman, "Random Forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[5] F. Pedregosa *et al.*, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.