Projecting Major League Baseball Pitcher Success for the 2024 Season

Allen Zou, Ting-An Lu, Zachary Chao, and Casey Hild

May 12, 2024

Abstract

In the realm of Major League Baseball (MLB), the evaluation of pitcher performance is crucial for team success. This study examines the efficacy of analytics in forecasting pitchers' weighted on-base average (wOBA) for the 2024 MLB season. Leveraging a dataset encompassing player attributes and Statcast-derived metrics, we explore whether integrating advanced analytics into linear regression models enhances predictive accuracy compared to traditional metrics like expected wOBA (xwOBA). Our methodology involves feature selection, k-fold cross-validation, and grid search hyperparameter tuning, with a focus on Elastic Net regression. We then analyze the model's coefficients and discuss limitations and potential avenues for future research.

1 Introduction

In Major League Baseball (MLB), assembling a competitive team is challenged by the complexities of player evaluation and roster construction. With millions of dollars invested annually in player acquisitions, teams increasingly rely on data-driven approaches to make informed decisions. One important component of this endeavor is the evaluation of pitching talent, where the ability to prevent opposing batters from reaching base is paramount. Traditional metrics such as Earned Run Average (ERA) and Wins Above Replacement (WAR) have long served as yardsticks for pitcher performance. However, the availability of granular player data now offers insights beyond the confines of conventional statistics.

One such metric gaining prominence is weighted on-base average (wOBA), a composite measure that assigns differential weights to various offensive outcomes. Unlike ERA, which can be influenced by factors beyond a pitcher's control (e.g., fielding errors), wOBA accounts for the quality of contact allowed. In recent years, the proliferation of Statcast data – a sophisticated tracking technology capturing the details of every on-field action – has enabled the derivation of novel performance metrics and improved predictive power.

Amidst this backdrop, our study aims to investigate the efficacy of advanced analytics in projecting pitchers' wOBA for each MLB season. Leveraging a dataset encompassing a myriad of player attributes and Statcast-derived metrics, we seek to ascertain whether performance indicators derived from these analytics outperform traditional benchmarks in forecasting pitcher performance. Specifically, we hypothesize that the integration of Statcast data into linear regression models will yield more accurate projections of pitchers' wOBA compared to reliance solely on conventional metrics such as the previous year's expected wOBA (xwOBA).

This paper is organized as follows: We begin with a description of the dataset used in our study, along with details of the preprocessing steps undertaken. Following this, we provide an outline of our methodology, including the selection of models and the criteria for evaluation. Subsequently, we present the results of our analyses, followed by an interpretation of these results. We then engage in a discussion of the implications of our findings. Finally, we conclude with a consideration of the limitations of our study and suggestions for future research directions.

2 Dataset

The pitcher performance dataset used in this study comprises data sourced from Baseball Savant. The dataset was compiled by downloading data from the Baseball Savant website, spanning the seasons

from 2015 to 2024.

Each row within the dataset represents a unique pitcher's performance profile, capturing essential statistics and advanced analytics metrics across 14 columns. The first column, "player_id," serves as a unique identifier for each pitcher. The "year" column designates the specific season under consideration.

Numerous performance metrics are encapsulated in the dataset columns. These include plate appearances (pa), strikeout percentage (k_percent), walk percentage (bb_percent), and metrics such as barrel rate (barrel_batted_rate), sweet spot hits (sweet_spot_percent), and hard hit percentage (hard_hit_percent). Additionally, features like average speed (avg_best_speed) and hyper speed (avg_hyper_speed) provide insights into the velocity aspect of pitcher performance, while whiff percentage (whiff_percent) and swing percentage (swing_percent) quantify their swings and misses from opposing batter.

Below is a table summarizing these variables.

Variable	Description
pa	Plate appearances
k_percent	Strikeout percentage
bb_percent	Walk percentage
woba	Weighted On-Base Avg.
xwoba	Expected wOBA
sweet_spot_percent	Sweet spot hits
barrel_batted_rate	Barrel rate
hard_hit_percent	Hard hits
avg_best_speed	Average speed
avg_hyper_speed	Hyper speed
whiff_percent	Whiff percentage
swing_percent	Swing percentage

Table 1: Variables and Descriptions

2.1 Data Pre-processing

We preprocessed the data to prepare it for analysis. Initially, we loaded the data from a CSV file containing player statistics spanning the years 2015 to 2024. We excluded irrelevant columns, including player names, unique identifiers, and the year of observation. Subsequently, we split the dataset into training and testing sets using an 80-20 split ratio. The resulting training dataset contains 900 rows, while the testing dataset comprises 225 rows. Not including the wOBA and xwOBA columns, there are 10 columns remaining.

3 Methodology

In this section, we provide an overview of the methodologies used for model selection and evaluation, as well as the steps involved in developing predictive models. More detail about how we applied these methodologies to select our model will be provided in Section 4 (Model Selection). It should be noted that all the strategies employed here involve the use of scikit-learn's packages [4]. We employ a systematic approach, beginning with an explanation of why we chose the linear regression model, followed by feature selection, k-fold cross-validation, and grid search hyperparameter tuning. Each subsection outlines a aspect of our methodology, covering forward feature selection, hyperparameter optimization, cross-validation, and model evaluation. However, before delving into these methodologies, we will discuss why decision trees were chosen as our primary modeling technique.

3.1 Linear Regression Model

The choice of a linear regression model for this study is mainly due to its simplicity and interpretability. Linear regression assumes a linear relationship between the independent variables and the dependent variable. By estimating the coefficients of each predictor variable, linear regression provides insights into the strength and direction of their associations with the target variable.

Linear regression offers several advantages. They are computationally efficient and can handle large datasets with many predictors. They also provide easily interpretable coefficients for each predictor, allowing for insights into the direction and strength of their relationships with the target variable.

However, it's essential to acknowledge some limitations associated with linear regression. As described above, one key assumption is the linearity between predictors and the target variable, which might not hold in all cases. Additionally, linear regression is sensitive to outliers and multicollinearity among predictor variables. To address these challenges, techniques like regularization, such as Ridge and Lasso regression, were utilized to prevent overfitting and improve model generalization.

3.2 L1 and L2 Regularization

Regularization techniques in linear regression help prevent overfitting and improve model generalization. Two common forms of regularization are L1 (Lasso) and L2 (Ridge) regularization.

3.2.1 L1 Regularization (Lasso)

L1 regularization adds a penalty equal to the absolute value of the coefficients to the loss function. This form of regularization tends to produce sparse models, where some coefficients can become exactly zero. The L1 penalty is controlled by the hyperparameter alpha.

3.2.2 L2 Regularization (Ridge)

L2 regularization adds a penalty equal to the square of the coefficients to the loss function. Unlike L1 regularization, L2 regularization does not enforce sparsity but instead tends to shrink the coefficients towards zero. This helps in reducing the variance of the model. The L2 penalty is also controlled by the hyperparameter alpha.

3.2.3 Elastic Net Regularization

Elastic Net is a regularization technique that combines both L1 and L2 penalties. It introduces an additional hyperparameter, l1_ratio, which controls the balance between L1 and L2 regularization. When l1_ratio is set to 0, Elastic Net behaves like Ridge regression, and when set to 1, it behaves like Lasso regression. Intermediate values of l1_ratio provide a compromise between the two. In our study, we employ Elastic Net regression to take advantage of these combined properties, optimizing the hyperparameters alpha and l1_ratio through grid search, which we'll discuss in a later section, to achieve the best predictive performance.

3.3 Forward Selection Overview

Feature selection aims to identify the subset of relevant features that contribute most to the model's predictive performance while reducing dimensionality and computational complexity [2]. In our methodology, we employed a forward feature selection approach to iteratively select features based on their impact on model performance.

The forward feature selection process begins with an empty set of selected features and iteratively adds one feature at a time, evaluating the model's performance at each step. At each iteration, the algorithm identifies the feature that results in the greatest improvement in model performance, as measured by a chosen evaluation metric. This metric, in our case, was the mean squared error (MSE), a common measure of regression model accuracy.

To evaluate the performance of each feature subset, we utilized k-fold cross-validation, a robust technique for estimating the model's performance on unseen data. In our implementation, we employed a 5-fold cross-validation strategy, partitioning the dataset into 5 equally sized folds, training the model on 4 folds, and evaluating it on the remaining fold [3]. This process was repeated 5 times, with each fold serving as the validation set exactly once.

During each iteration of the forward feature selection process, we trained a linear regression model on the preprocessed training dataset using the selected features plus the feature under consideration. We then used grid search with cross-validation to tune the hyperparameters of the linear regression model.

Normally, the feature selection process continued until adding additional features no longer resulted in a significant improvement in model performance or began to degrade performance, but for the purpose of visualization, we continued to run forward selection until it depletes all features.

Upon completion of the feature selection process, we stored the selected features, their corresponding mean squared errors, and the hyperparameters of the best-performing models each iteration for further analysis and model evaluation.

3.4 K-Fold Cross-Validation

K-fold cross-validation is a widely used technique to assess the performance and generalization ability of a predictive model. The main idea behind k-fold cross-validation is to partition the dataset into k subsets, or "folds," of approximately equal size [3]. The model is then trained k times, each time using k-1 folds as the training set and the remaining fold as the validation set. This process allows each data point to be used for validation exactly once. The performance metric, such as mean squared error (MSE) for regression tasks or accuracy for

classification tasks, is computed for each fold, and the average performance across all folds is reported as the overall performance estimate of the model. K-fold cross-validation helps mitigate the risk of overfitting by providing a more reliable estimate of a model's performance on unseen data compared to a single train-test split. Common choices for the value of k include 5 or 10, although the choice may vary depending on the size and nature of the dataset.

We opted to use 5-fold cross-validation due to the characteristics of our preprocessed training dataset, which comprises 900 rows and 12 columns. With a moderate-sized dataset, 5-fold cross-validation strikes a balance between bias and variance in model evaluation. By partitioning the dataset into 5 equally sized folds, each fold contains 180 samples, ensuring that the training sets remain sufficiently large for model fitting while still allowing for comprehensive evaluation across multiple iterations.

3.5 Optimizing Hyperparameters with Grid Search

In each iteration of the feature selection process, hyperparameters are optimized using scikit-learn's grid search to fine-tune the linear regression model [4]. Grid search is a systematic method for tuning hyperparameters by exhaustively searching through a predefined grid of parameter values and selecting the combination that yields the best model performance [1]. For the linear regression model, hyperparameters such as alpha and l1_ratio are considered.

The grid search algorithm evaluates the model's performance using k-fold cross-validation as outlined in the previous subsection. By averaging the performance across all folds, grid search provides an estimate of the model's performance under various hyperparameter configurations. The hyperparameter values that yield the lowest mean squared error (MSE) are selected as the optimal configuration for the linear regression model.

4 Model Selection

In the Model Selection section, we explore the application of methodologies described in the Methodology section to identify the optimal predictive model for our housing price prediction task. First, we detail the process of constructing a parameter grid tailored to our linear regression model. Subsequently, we elaborate on how we leveraged the outcomes derived from grid search cross-validation to select hyperparameters and a subset of features.

4.1 Grid Search Parameter Range

The hyperparameters considered in this study are alpha and l1_ratio, which play crucial roles in regularization. As described in an earlier section, alpha controls the strength of the regularization, while l1_ratio determines the balance between L1 (Lasso) and L2 (Ridge) regularization. To explore a wide range of values, we defined the following parameter grid for grid search:

- **alpha**: 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1e-0
- **l1_ratio**: 1e-7, 1e-6, 1e-5, 1e-4, 1e-3, 1e-2, 1e-1, 1e-0

During the grid search process, each combination of alpha and l1_ratio values is evaluated using 5-fold cross-validation. This approach ensures a thorough examination of the parameter space, identifying the optimal hyperparameters that minimize the mean squared error (MSE). By evaluating the performance of each parameter combination, we aim to achieve a balance between overfitting and underfitting to improve model generalization.

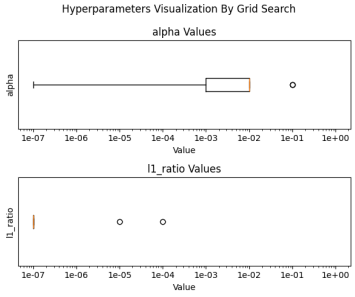

Figure 1: Hyperparameters Chosen by Grid Search

To ensure the effectiveness of our grid search process, we analyzed the best hyperparameters that grid search selected among all iterations, as depicted in figure 1. For the alpha hyperparameter, the results indicate that the range of values chosen seems to cover the correct area, as most of the values selected are around the middle of the range of values. However, for the l1_ratio values, the model

selected the lowest possible value of 1e-07 in almost every iteration. Although we observed this pattern, we decided against extending the range lower than 1e-07 due to limitations within scikit-learn, which can result in convergence errors at extremely low values. Despite this constraint, the chosen range of values for both alpha and l1_ratio still provids a comprehensive search within the limits of scikit-learn.

4.2 Feature Subset and Hyperparameter Selection

As described previously, the algorithm pinpoints the feature that yields the most substantial enhancement in model performance every iteration, gauged by MSE. Based on the feature subset identified with the best-performing feature, we accumulate all the MSE values from the grid search on that particular model using the optimal feature subset. These aggregated MSE values form the basis for constructing a box and whisker plot, with each plot representing the MSE values from grid search on the best-performing feature subset for that iteratoin.

In Figure 2, we present the box and whisker plots illustrating the MSE values of the model with the best-performing feature subset for all iterations of forward selection—comprising a total of 10 iterations, corresponding to the number of features in the preprocessed dataset. The initial iteration exhibits a markedly elevated box and whisker plot, indicating minimal improvement when incorporating a single feature.

Throughout all iterations, the majority of MSE values converge to roughly the same value as the mean, indicating mostly consistency in model performance as additional features are included. However, there are noticeable outliers high above all box and whisker plots, all of which hover around the 0.0011 mark. Upon further inspection, we discovered that these outliers likely occur when grid search selects the value of 1 for either alpha or l1_ratio.

This finding is evidenced by Figure 3, which shows the MSE values for the forward selection process, excluding the value of 1 in the grid search parameter grid. Specifically, the outliers disappear when rerunning the model selection process. Additionally, the figure visualizing the hyperparameters chosen by grid search back in Figure 1 supports this conclusion, as the best hyper-parameter selected was never 1 for either alpha or l1_ratio.

Given these observations, especially since the outliers correspond to higher MSE values, we chose to ignore the outliers when selecting the model. Additionally, we decided to focus our model selection based on the adjusted forward selection process because, in this scenario, most of the outliers also converged closer to the mean. Moreover, we will use the parameters from this adjusted forward selection process for our final model.

4.3 Final Hyperparameters and Features

We chose the model from iteration 7 of the forward selection process. This decision was driven by the observation that the MSE values for this model were generally the lowest across all iterations.

Hyperparameter	Value
alpha	1e-03
l1_ratio	1e-07

Table 2: Optimal hyperparameters for the decision tree model.

The hyperparameters listed in Table 2 represent the configuration determined by grid search to be the most optimal for the elastic net regression model chosen.

Table 3: Selected Features

Feature	Description
avg_hyper_speed	Hyper speed
k_percent	Strikeout percentage
bb_percent	Walk percentage
sweet_spot_percent	Sweet spot hits
barrel_batted_rate	Barrel rate
swing_percent	Swing percentage
pa	Plate appearances

The selected features, as shown in Table 3, were determined through the forward selection algorithm employed during the model selection process. These features represent a subset of the original set of predictors and were chosen by the end of iteration 7 based on their contribution to improving the model's predictive performance.

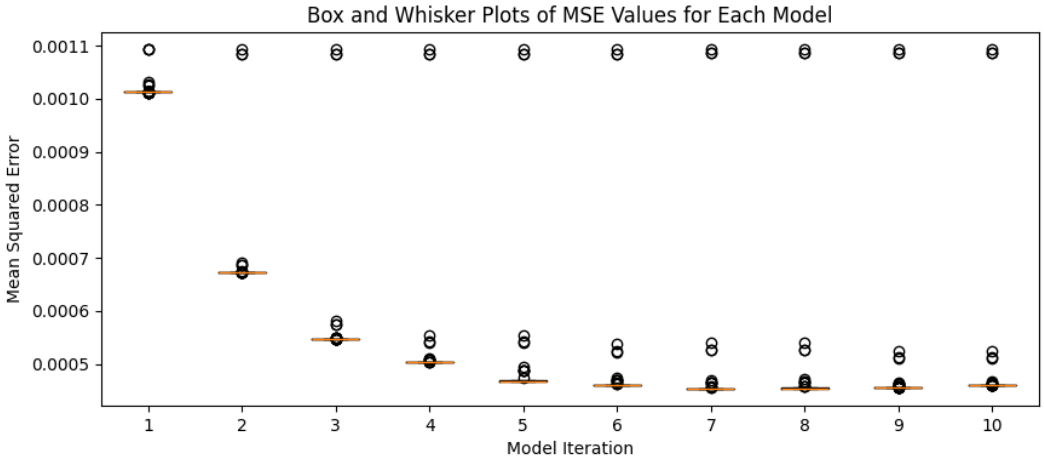

Figure 2: MSE values each iteration of grid search

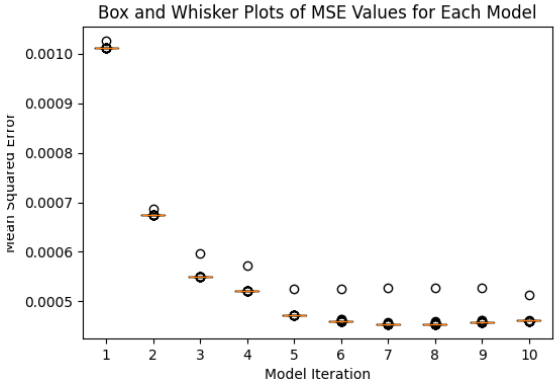

Figure 3: MSE values each iteration of grid search without values of 1 for each hyperparameter in the parameter grid

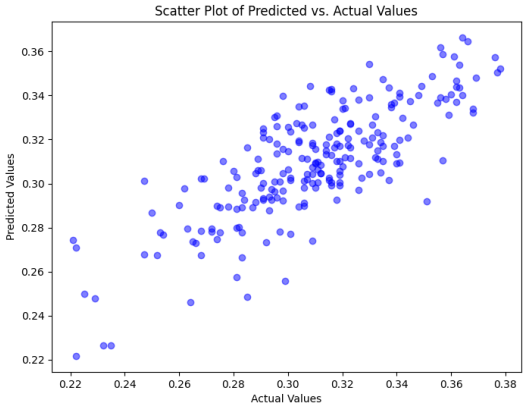

Figure 4: Scatterplot of predicted vs test values

5 Results

5.1 Performance Metrics

The performance of each model was evaluated using a common metric: Mean Squared Error (MSE). MSE calculates the average of the squares of the errors, giving higher weights to larger errors and penalizing outliers more heavily [1].

After retraining the elastic net regression model with the hyperparameters shown in Table 2, we evaluated its performance on the testing dataset. The resulting Mean Squared Error (MSE) was about 0.00037.

However, further examination is warranted. Upon inspecting the scatterplot in Figure 4, which illustrates the relationship between predicted and true prices, while there's a positive correlation, the scatter suggests that the predictions lack precision, as evidenced by the somewhat scattered distribution of data points. Even though the Pearson's correlation coefficient, calculated to be about 0.79, does indicate a positive linear relationship between predicted and true prices, the model's predictive accuracy may still be improved.

6 Interpretation and Analysis

In this section, we interpret the linear regression's coefficients associated with each feature. These coefficients represent the change in the target variable for a one-unit change in the corresponding feature, holding all other features constant. Higher magni-

tude coefficients indicate a stronger influence on the target variable. The feature coefficients table (Table 4) displays the coefficients of each variable sorted by magnitude.

Feature	Importance
avg_hyper_speed	0.007049
k_percent	-0.003764
bb_percent	0.003542
sweet_spot_percent	0.002649
barrel_batted_rate	0.002290
swing_percent	-0.0003238
pa	0.00001499

Table 4: Feature Coefficients from Linear Regression Model (Sorted By Magnitude)

Positive coefficients indicate a positive relationship, where an increase in the feature value corresponds to an increase in the predicted wOBA. Conversely, negative coefficients suggest a negative relationship, indicating that an increase in the feature value leads to a decrease in the predicted wOBA.

Among the features, "avg_hyper_speed" has the highest positive coefficient, indicating that higher average hyper speed is associated with higher predicted wOBA. Conversely, "k_percent" and "swing_percent" have negative coefficients, suggesting that a higher strikeout percentage and swing percentage correspond to lower predicted wOBA. Other features such as "bb_percent," "sweet_spot_percent," and "barrel_batted_rate" also exhibit positive coefficients, indicating positive relationships with the target variable. However, despite iteration 7's models generally performing better in terms of MSE, "pa" doesn't exhibit a noticeably strong correlation with wOBA.

7 Discussion

Based on our hypothesis and the results obtained from our analysis, we can draw conclusions regarding the effectiveness of integrating Statcast data into our linear regression model for forecasting pitcher performance. Our hypothesis posited that leveraging Statcast data would lead to more accurate projections of pitchers' wOBA compared to relying solely on conventional metrics such as the previous year's expected wOBA (xwOBA).

Upon examination of the Mean Squared Error (MSE) values, we find that the Elastic Net model, incorporating a diverse range of player attributes and Statcast-derived metrics, achieved an MSE of 0.000372, while a simple linear regression model trained solely with xwOBA as a predictor yielded an MSE of 0.000284.

It appears that the model utilizing solely xwOBA as a predictor outperforms the Elastic Net model in terms of predictive accuracy, as evidenced by the lower MSE value. This suggests that, contrary to our hypothesis, the integration of Statcast data into our linear regression model did not lead to more accurate projections of pitchers' wOBA compared to reliance solely on xwOBA. Additionally, the coefficient for xwOBA in the simple linear regression model is approximately 0.924, which has a much higher magnitude than the coefficients of the predictors in our linear regression model, indicating its strong predictive power.

Further analysis and exploration may be necessary to understand the reasons behind this discrepancy and to refine our approach to incorporating advanced analytics into predictive modeling for pitcher performance.

8 Research Limitations and Future Directions

Our analysis hinges on a specific dataset covering the years 2015 to 2024, raising concerns about the stability and consistency of pitcher performance metrics across different seasons. While this dataset offers valuable insights, training a model on data from all available years may not account for variations in player performance, rule changes, and other contextual factors between years. Future research could address these concerns by analyzing data on a year-by-year basis.

It's essential to acknowledge the assumptions and limitations inherent in linear regression modeling, which may impact the validity of model estimates and predictions. Linear regression assumes linearity, independence of observations, and homoscedasticity, among other assumptions. To address these limitations, future research could explore alternative modeling techniques. Additionally, investigating the convergence issues encountered during grid search, particularly concerning the selection of l1_ratio values near the lower bounds, suggests the need for further research into model convergence and optimization strategies.

Moreover, the discrepancies in the magnitude of coefficients between our model and a simple linear regression model trained solely on xwOBA merit additional investigation. The notably higher coefficient for xwOBA in the simple linear regression model suggests the need for deeper exploration into feature engineering and selection processes. Similarly, the presence of some outliers in the forward selection process converging around 0.0011 warrants further scrutiny to understand the underlying factors contributing to these anomalies.

9 Conclusion

In this study, we explored the integration of analytics, particularly Statcast data, into linear regression models for forecasting pitcher performance in Major League Baseball. Our analysis encompassed a dataset spanning player attributes and Statcast-derived metrics, with a focus on predicting weighted on-base average (wOBA) for each season.

While our initial hypothesis suggested that incorporating Statcast data would lead to more accurate predictions compared to traditional metrics like expected wOBA (xwOBA), our findings suggest otherwise. Despite employing advanced techniques such as Elastic Net regression and feature selection, our results indicated that a simple linear regression model trained solely on xwOBA yielded slightly better predictive accuracy.

The discrepancy in predictive accuracy raises questions about the factors influencing pitcher performance and the effectiveness of different modeling approaches. Further exploration into the reasons behind the observed discrepancies, such as the magnitude of coefficients and convergence issues encountered during hyperparameter tuning, could shed light on the nuances of player evaluation in MLB or any issues in our approach. Future research could also explore alternative modeling techniques, address convergence issues, and refine feature engineering and selection processes to enhance predictive accuracy.

In conclusion, our study underscores the complexities inherent in forecasting pitcher performance. While our findings may not fully validate our initial hypothesis, they pave the way for continued exploration and refinement of predictive modeling techniques in the dynamic realm of Major League Baseball.

References

[1] Aurélien Géron. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, 2017.

[2] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer, 2009.

[3] Max Kuhn and Kjell Johnson. *Applied Predictive Modeling*. Springer, 2013.

[4] Scikit-learn Contributors. Scikit-learn documentation, n.d.