Using Machine Learning to Predict MLB Playoff Teams

Sean Glaze
Computer Science and Software Engineering
Miami University
Oxford, US
glazesc@miamioh.edu

David Nwafor Tah
Computer Science and Software Engineering
Miami University
Oxford, US
nwafordu@miamioh.edu

Zongyao Qi
Computer Science and Software Engineering
Miami University
Oxford, US
qiz10@miamioh.edu

Alexander Kalb
Computer Science and Software Engineering
Miami University
Oxford, US
kalbaj@miamioh.edu

***Abstract*—Baseball has a big effect on American culture, seen in its huge popularity and global reach, with MLB teams being worth billions and players signing extremely valuable contracts, like Shohei Ohtani's $700 million deal. Many people very much look forward to the MLB post-season, and with legal sports betting becoming the norm, this popularity will only grow. Also, teams are increasingly utilizing advanced stats in scouting reports and the like. This paper aims to create a reliable machine learning model for forecasting MLB playoff teams using historical team statistics. We consider the statistics from all 30 teams over the 2020-2023 seasons. We employ various machine-learning-based classification models, including logistic regression, decision trees, random forest, gradient boosting, and support vector machines (SVM) in our model. We leverage findings from previous work to guide our data collection and preprocessing techniques. After training the 5 aforementioned models on the collected data, we implement a majority vote system to make a final prediction. We evaluate our model using 5 common metrics: accuracy, precision, recall, F1-Score, and area under the receiver operating characteristics (ROC) curve. Finally, we conclude that our model is the best-performing of its kind, and we identify the features most influential in its classifications.**

***Index Terms*—Machine Learning, Classification Algorithm, MLB, Major League Baseball, Baseball**

I. Introduction

According to USNews [6], American culture is ranked 3rd among 194 other countries in the world regarding its sheer worldly influence. The huge influence that America has on the world can be partially credited to the popularity of US sports. Baseball is placed second only behind American football in the list of most popular sports in the US. The most valuable sports franchises are extremely successful, having massive fan bases and prospects to attract amazing players and cunning coaches. These teams command billions of dollars flowing through their organizations each year. The New York Yankees, one of the most valuable sports teams in the world, leads the MLB with a valuation exceeding seven billion dollars. Additionally, player contracts increase in value every year, with overall contract values quickly approaching one billion dollars for a single player. For example, the current highest-paid player in Major League Baseball (MLB), Shohei Ohtani of the Los Angeles Dodgers, recently agreed to a $700 million 10-year contract. Given the lavish spending within the MLB, it's no surprise that the playoffs are so highly anticipated. It draws an immense amount of fan interest that generates significant income.

Thus, predicting the teams that will fill the 12 playoff spots is of high interest to both fans and stakeholders within the MLB. Since the birth of Major League Baseball, there have been countless efforts to predict regular season results. Made famous by the sports biopic Moneyball, Billy Beane, the general manager of the Oakland Athletics in 2001, used advanced statistics and data analytics to create a cost-effective but successful baseball team. This way of looking at the game was never seen before and ahead of its time. The success of this approach had started a trend around the league, as more and more teams sought to analyze advanced statistics in an effort to strike an optimal balance between team payroll and team success.

A. Research Questions

In this paper, we ask two research questions:

- **RQ1:** How well does our proposed model identify MLB playoff teams?
- **RQ2:** Which feature(s) best distinguish MLB playoff teams?

II. Related Work

A. Use of Machine Learning and Deep Learning to Predict the Outcomes of Major League Baseball Matches

There have been several efforts to apply machine learning techniques to Major League Baseball in recent years. Firstly, Mei-Ling Huang and Yun-Zhi Li published a paper titled "Use of Machine Learning and Deep Learning to Predict the Outcomes of Major League Baseball Matches" [7]. In

this paper, the authors performed a thorough feature selection process in an effort to reduce the number of considered input variables in their ML models. Once this was done, the authors employed three different machine-learning techniques to predict the outcomes of games. Specifically, they used a one-dimensional convolutional neural network (1DCNN), a traditional machine learning artificial neural network (ANN), and a support vector machine (SVM). Using each of these methods, they were able to achieve accuracies of 93.4%, 93.91%, and 93.90% respectively. Moreover, the data they used to train these models focused on the starting pitchers and available pitchers in each game – indicating that these features may be more influential. It is worth noting, however, that this worth focuses on the prediction of individual games, whereas in this paper we seek to identify playoff teams.

B. Machine Learning Outperforms Regression Analysis to Predict Next-Season Major League Baseball Player Injuries: Epidemiology and Validation of 13,982 Player-Years From Performance and Injury Profile Trends, 2000-2017

Secondly, we take a look at a paper titled "Machine Learning Outperforms Regression Analysis to Predict Next-Season Major League Baseball Player Injuries: Epidemiology and Validation of 13,982 Player-Years From Performance and Injury Profile Trends, 2000-2017", authored by Karnuta, Et al. [8]. In this work, the authors analyzed the performance of 84 different machine learning algorithms and their effectiveness in predicting player injuries. They found that advanced machine learning models outperformed logistic regression in predicting publicly reportable next-season injuries.

C. Multimodal Machine Learning For Major League Baseball Playoff Prediction

Lastly, we consider a work titled "Multimodal Machine Learning For Major League Baseball Playoff Prediction" by Yaseen et al. [1]. Of any work that we were able to find in this field, this is most similar to what we look to accomplish. In this paper, the authors employed a number of supervised learning techniques to predict the playoff teams for the 2019 MLB season. They were able to achieve scores of 77% accuracy and 59% recall. This indicates to us that there is certainly work that can be done in this field. During our research, the authors provided us with several things that will benefit us during this project. First, they provide a comprehensive overview of their data collection and data cleaning approaches, giving us insight into what may be able to be improved in our approach. Also, they give us a straightforward way for us to evaluate our predictions. Namely, precision, recall, and F-1 score. They also provide us with their results using these metrics, a great way to compare our metrics to a similar project.

III. Background

There were several knowledge foundations that were necessary for the completion of this project. First, we needed to have a very good grasp of the basics of Python, and knowledge of the following libraries: NumPy, Pandas, MatPlotLib, Seaborn, and SkiKit-Learn. Second, we needed to have a basic understanding of each of the machine-learning-based classification algorithms that we used in this project. Namely, logistic regression, decision tree, random forest, gradient boosting, and support vector machine models. Finally, one must have a good grasp of the state of the field, and what benchmarks there are to compare this proposed model.

IV. Data Collection

A. Feature Selection

For each team's data that we considered, we collected 16 features. These features cover all three dimensions of baseball: *pitching*, *fielding*, and *batting*. We list the selected features below:

1) Pitching features:
    a) Win-Loss Percentage
    b) Earned-Run Average
    c) WHIP (Walks and Hits per 9 Innings)
    d) Hits Allowed per 9 Innings
    e) Home Runs Allowed per 9 Innings
    f) Walks Allowed per 9 Innings
    g) Strike Outs per 9 Innings
    h) Strike Outs / Walks
2) Fielding features:
    a) Runs Allowed per Game
    b) Defensive Efficiency
    c) Fielding Percentage
3) Batting features:
    a) Runs Scored per Game
    b) Team Batting Average
    c) Team On-Base Percentage
    d) Team Slugging Percentage
    e) Team OPS (On-Base Percentage Plus Slugging)

We selected these features due to their popularity among baseball fans and relative simplicity compared to other metrics.

1) Manual Data Entry: To collect our selected features, we use a very popular website that keeps track of MLB baseball statistics: https://www.baseball-reference.com/. We decided to collect data from the 2020-2023 MLB seasons – a total of 4 seasons. We chose this timeframe because we wanted to strike a balance between collecting data on as many years as possible while recognizing that the game of baseball is constantly changing, with trends in gameplay constantly evolving and minor tweaks to the rules of the game coming out each year.

For example, in 2023 the MLB instituted 2 major rule changes [5]:

1) The introduction of the *pitch timer*, limiting the amount of time a pitcher can take between delivering pitches.
2) The ban of the *defensive shift*, removing infielder's ability to play on the opposite side of the infield as they would traditionally.

After selecting the seasons for which we would collect data, we manually gathered each of the 16 selected features for all 30 MLB teams for each season over 2020-2023. For

example, each of the 16 features can be found for each team in the 2023 season on the following webpage: https://www.baseball-reference.com/leagues/majors/2023.shtml. We input each of these features into an Excel sheet for ease of processing. Figure I presents a sample of our manually gathered dataset.

2) Ground Truth: For each team considered in our dataset, we manually gathered whether they made the playoffs in that given year. Just like before, we gathered this data using baseball-reference. For instance, we can easily find the teams that made the 2023 playoffs on the following webpage: https://www.baseball-reference.com/leagues/majors/2023-standings.shtml#all_postseason. We appended the dataset presented in Table I with this label, called "Made Playoffs". This is exemplified in Table II.

TABLE II
SAMPLE OF MANUALLY GATHERED LABELS FOR EACH TEAM IN OUR DATASET.

Team	Made Playoffs
Arizona Diamondbacks 2023	TRUE
Atlanta Braves 2023	TRUE
Baltimore Orioles 2023	TRUE
Boston Red Sox 2023	FALSE
Chicago Cubs 2023	FALSE

B. Final Processing Steps

1) Removing Correlated Features: Correlated features may contribute to inaccurate classifications [3], [4]. In order to mitigate this issue, we used the Python libraries Pandas, Seaborn, and MatPlotLib to create and display a matrix with Pearson coefficients for each pair of features. We used this matrix to identify pairs of features with a Pearson coefficient greater than 0.7. This threshold is commonly used in previous literature [2]. Figure 1 presents a heatmap of the correlation between our selected features.

The below sequence describes our feature removal process:

1) Identify pairs of features with a Pearson correlation of greater than 0.7. Ex: $\{(f_0, f_1), (f_1, f_2), (f_2, f_3), ...\}$.
2) From this set of pairs of features, count how many times each feature is included in a pair. Ex: $f_0 : 7, f_1 : 5, f_2 : 3, ...$
3) Remove the feature with the highest frequency of being included in a pair in the set (Ties are resolved by removing the feature that appears first alphabetically). Using the above example, f_0 would be removed from the feature set.
4) Repeat steps 1-3 until no feature is involved in 4 or more pairs of features with greater than a 0.7 Pearson correlation coefficient.

The set of features that were removed following the above process are:

- Team WHIP
- Team Earned Run Average (ERA)
- Team Slugging Percentage (SLG)
- Team Batting Average (BA)
- Runs Allowed Per Game

Finally, we removed the minimal set of features that would leave us with a feature set where no two features have a Pearson correlation coefficient greater than 0.9. We chose this threshold so that we could still preserve most of our dataset. This resulted in the removal of only one additional feature, OPS. This left us with 10 features that were considered by our model.

V. METHODS

In this section, we will discuss which machine learning classifier algorithms we will utilize in our model, our hyperparameter tuning methodology, and how we put the output of each classifier algorithm we utilize to produce a singular prediction.

A. Machine Learning Classifiers

In our model, we utilize 5 machine-learning-based classifier algorithms:

1) Logistic Regression
2) Decision Trees
3) Random Forest
4) Gradient Boosting
5) Support Vector Machine (SVM)

We chose these classifiers due to their widespread usage across several applications and their ease of use through the SkiKit-Learn Python library.

B. Hyperparameter Fine-Tuning

We will now discuss what hyperparameters were chosen for each of the 5 classification algorithms we chose. Each hyperparameter that is referenced will refer to them by the way they are denoted as parameters in the SkiKit-Learn Python library.

1) Logistic Regression
   - *penalty = l2*. This specifies the regularization penalty – in our case, we opted for L2 regularization, the default option.
   - *C=1.0*. This specifies the regularization strength – in our case, we opted for the default strength.
2) Decision Trees
   - *max_depth = 5*. This specifies the maximum depth of the decision tree. The default value for this hyperparameter is None, however, we opted for a maximum depth of 5.
   - *min_sample_split = 2*. This specifies the minimum number of samples required to split an internal node – we chose the default value of 2 for this hyperparameter.
   - *min_samples_lead = 1*. This specifies the minimum number of samples required to be a leaf node – we chose the default value of 1 for this hyperparameter.
3) Random Forest

TABLE I
SAMPLE OF MANUALLY GATHERED DATA – NOT ALL COLUMNS ARE DISPLAYED FOR SPATIAL REASONS.

Team	Runs Allowed/Game	Defensive Efficiency	Fielding Percentage	ERA	WHIP	H9
Arizona Diamondbacks 2023	4.7	0.695	0.99	4.48	1.324	8.6
Atlanta Braves 2023	4.42	0.685	0.986	4.14	1.302	8.4
Baltimore Orioles 2023	4.19	0.696	0.988	3.89	1.243	8.3
Boston Red Sox 2023	4.79	0.677	0.982	4.52	1.338	8.9
Chicago Cubs 2023	4.46	0.694	0.984	4.08	1.283	8.3

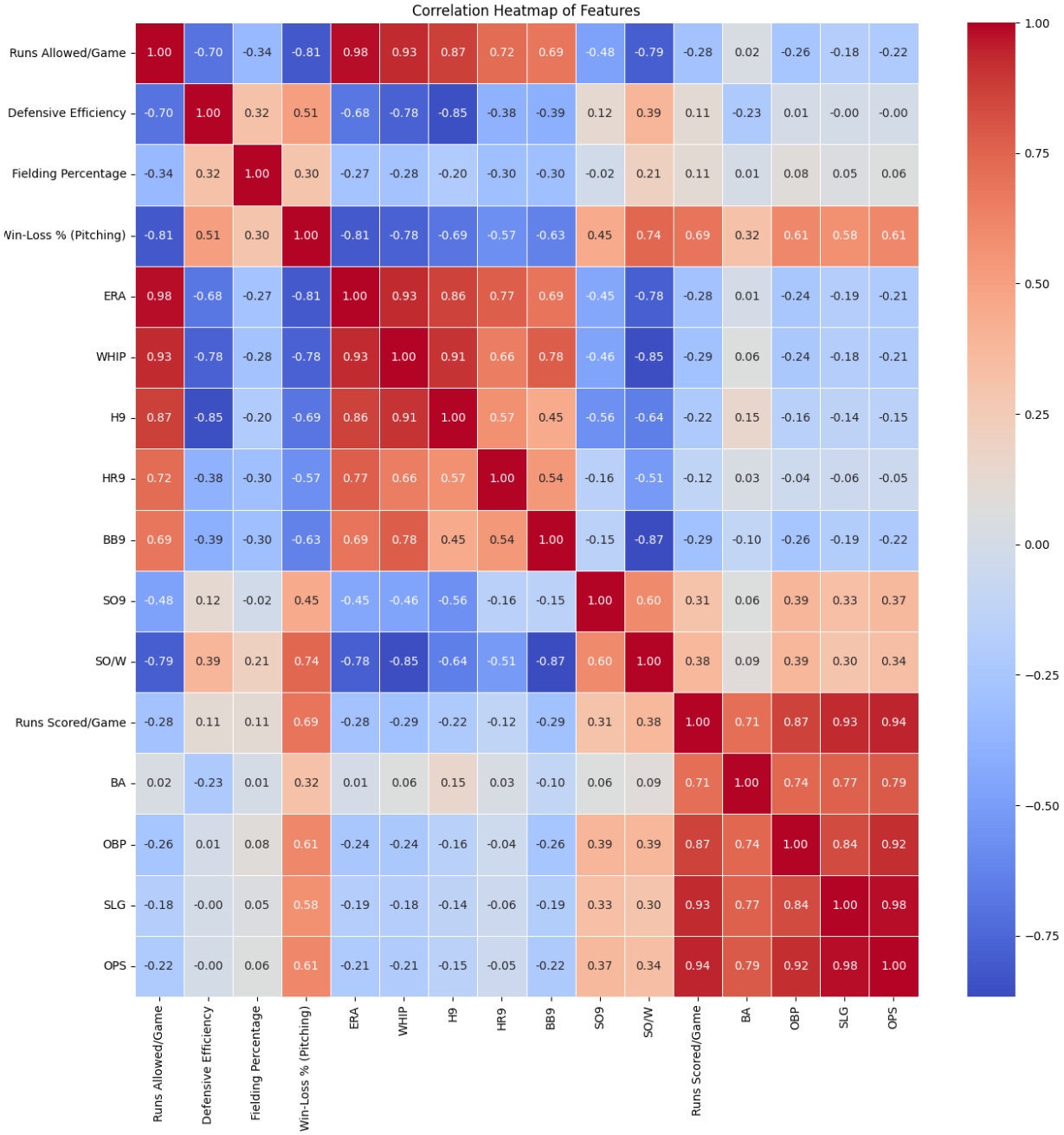

Fig. 1. Correlation Analysis of Selected Features.

- *n_estimators = 100*. This specifies the number of trees in the forest – we chose the default value of 100 for this hyperparameter.
- *max_features = "sqrt"*. This specifies the maximum number of features to consider when looking for the best split. In our case, we chose the square root of the total number of features.
- *max_depth = 10*. This specifies the maximum depth of the trees in the forest. In our case, we opted for a maximum depth of 10 instead of the default value of None for this hyperparameter.
- *min_sample_split = 2*. This specifies the minimum number of samples required to split an internal node – we chose the default value of 2 for this hyperparameter.
- *min_samples_lead = 1*. This specifies the minimum number of samples required to be a leaf node – we chose the default value of 1 for this hyperparameter.

4) Gradient Boosting
   - *n_estimators = 100*. This specifies the number of boosting stages – we chose the default value of 100 for this hyperparameter.
   - *learning_rate = 0.1*. This specifies the learning rate, which controls the contribution for each tree in the boosting algorithm. We chose the default value of 0.1 for this hyperparameter.
   - *max_depth = 3*. This specifies the maximum depth of the individual regression estimators in the boosting model – we chose the default value of 3 for this hyperparameter.
   - *min_sample_split = 2*. This specifies the minimum number of samples required to split an internal node – we chose the default value of 2 for this hyperparameter.
   - *min_samples_lead = 1*. This specifies the minimum number of samples required to be a leaf node – we chose the default value of 1 for this hyperparameter.
5) Support Vector Machine (SVM)
   - *C = 1.0*. This specifies the regularization parameter – used to prevent over-fitting by penalizing large coefficients. We chose the default value of 1.0 for this hyperparameter.
   - *kernel = rbf*. This specifies the kernel function that is used. In our case, we opted for the default kernel function: the Radial Basis Function (RBF). Other popular kernel functions include "Linear Kernel" and "Polynomial Kernel".
   - *gamma = scale*. This specifies the kernel coefficient used in the kernel function – i.e. the influence that individual training samples have on the decision boundary. In our case, we opted for the default kernel coefficient: scale.

C. Voting System

In order to take each of the 5 models' outputs into consideration in our model, we implemented a simple *voting system*, where our model's output is based on the majority vote between the 5 models. Since the number of models we consider is odd and we are considering a binary classification problem, calculating the majority vote is a very straightforward process, by giving each model exactly 1 vote in the outcome of our overall model.

VI. Results

First, we partitioned our dataset into an 80/20 split between training/test data. All 5 classification models were trained using the same training data. Then, we tested each model on the test dataset and computed a final prediction using the previously discussed voting system. Now, we discuss the metrics that we used to evaluate our model and its performance against these metrics.

A. ***RQ1:*** *How well does our proposed model identify MLB playoff teams?*

1) Evaluation Metrics: We evaluate our model using 5 common metrics used to measure the performance of a classification algorithm:

1) *Accuracy:*

$$\text{Accuracy} = \frac{\text{Number of Correct Predictions}}{\text{Total Number of Predictions}} \tag{1}$$

2) *Precision:*

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives + False Positives}} \tag{2}$$

3) *Recall:*

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives + False Negatives}} \tag{3}$$

4) *F1-Score:*

$$\text{F1-Score} = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}} \tag{4}$$

5) *AUC (Area Under the ROC (Receiver Operating Characteristic) Curve):*
   A measure of the classifier's ability to distinguish between the positive and negative classes across all possible thresholds. It is a value that ranges from 0 to 1. A score of 0.5 indicates random guessing, and a score of 1 indicates perfect classification.

2) Performance Against Metrics: Table III presents the performance of model, the ground truth, and our model's overall prediction for each team in our test dataset.

Table IV presents our model's performance against all metrics that we considered. Additionally, Figure 2 presents a confusion matrix that provides an easier digestion of our model's recall and precision performance, and Figure 3 presents a plot of our model's performance against the ROC metric.

TABLE III
EACH MODEL'S PREDICTION, OUR MODEL'S FINAL PREDICTION, AND GROUND TRUTH FOR EACH TEAM IN OUR TEST DATASET

Team	Logistic Regression	Decision Tree	Random Forest	Gradient Boosting	SVM	Ground Truth	Overall Prediction
Chicago Cubs 2023	TRUE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Texas Rangers 2020	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
San Diego Padres 2020	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Miami Marlins 2020	FALSE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE
Atlanta Braves 2022	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
San Francisco Giants 2023	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Philadelphia Phillies 2020	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Baltimore Orioles 2021	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Minnesota Twins 2023	TRUE	FALSE	TRUE	TRUE	TRUE	TRUE	TRUE
Chicago White Sox 2022	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Boston Red Sox 2022	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Colorado Rockies 2023	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Chicago White Sox 2023	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Arizona Diamondbacks 2023	FALSE	TRUE	FALSE	TRUE	FALSE	TRUE	FALSE
Los Angeles Dodgers 2023	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
New York Mets 2021	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Texas Rangers 2021	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Pittsburgh Pirates 2022	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Chicago Cubs 2020	TRUE	TRUE	TRUE	TRUE	FALSE	TRUE	TRUE
Los Angeles Angels 2021	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
Washington Nationals 2023	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
St. Louis Cardinals 2021	FALSE	TRUE	TRUE	FALSE	FALSE	TRUE	FALSE
Chicago White Sox 2021	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE	TRUE
Tampa Bay Rays 2022	TRUE	FALSE	FALSE	FALSE	FALSE	TRUE	FALSE

TABLE IV
OUR MODEL'S PERFORMANCE AGAINST ALL CONSIDERED METRI

Metric	Performance Against Metric
Accuracy	0.833
Precision	1.000
Recall	0.600
F1-Score	0.750
AUC	0.800

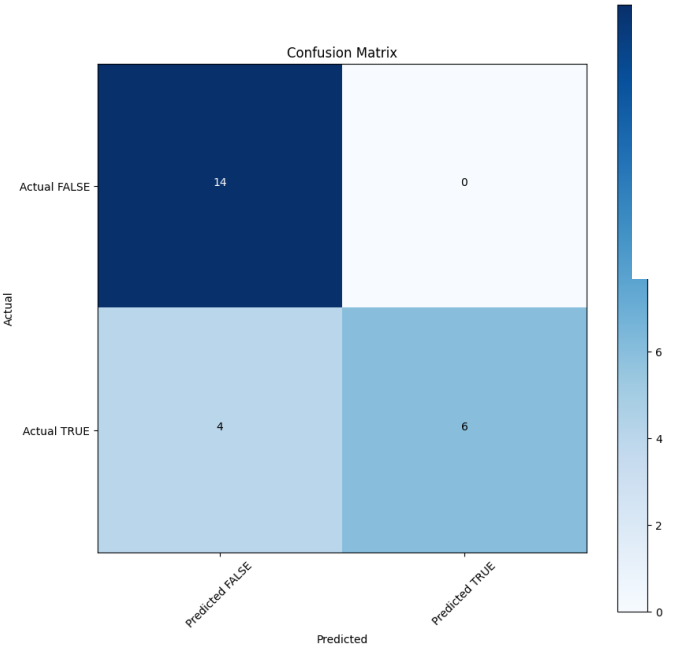

Fig. 2. Confusion Matrix that Presents our Model's Recall and Precision Scores.

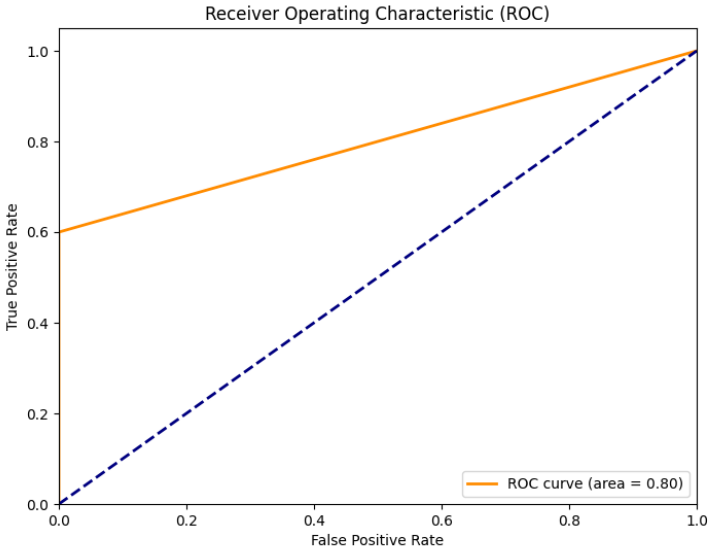

Fig. 3. Plot of ROC Curve.

Comparing our model's performance against Yaseen et al.'s [1] which reported a 0.77 precision score, a 0.59 recall score, and a 0.67 F-1 score in predicting MLB playoff teams for the 2018/2019 season, our model reports higher scores in all 3 metrics. To the best of our knowledge, this is the only published work that develops a model for predicting MLB playoff teams. Thus, we conclude that we have developed the most accurate model available for MLB playoff team prediction.

B. ***RQ2:*** *Which feature(s) best distinguish MLB playoff teams?*

In order to answer this research question, we figured out how much each of the models we used in our model was taking each feature into consideration in its prediction process. We did this by simply using attributes that are built into skikit-learn model objects. Specifically, we used the *feature_importance_* attribute of the gradient boosting, decision tree, and random forest models, and the *coef_* attribute of the logistic regress model to answer this question. Unfortunately, we are unable to see the importance of each feature in our SVM model's decision boundary, as this is only available on SVM models that use a linear kernel function. Table V presents the feature importance for our logistic regression, decision tree, random forest, and gradient boosting models.

From this presented data, we make several observations. First, we notice that a team's average win/loss percentage for their pitchers is the most important feature for all models other than logistic regression, where it is the 3rd highest weighted feature. Another observation we make is that in all models other than logistic regression, a team's runs scored per game statistic is hardly weighted into its model, if at all. This is very counter-intuitive, as intuitively, most would probably assume that the more runs a team scores on average, the more they would win, thus increasing their odds of making the playoffs. However, this observation appears to challenge that line of thinking. Though, it is worth noting that this feature is the highest weighted in our logistical regression model.

In summary, we observe through our presented data that a team's average win/loss percentage for their pitchers is more often than not, far and away the most influential statistic in a team's postseason odds.

VII. Conclusion and Future Work

In this paper, we investigated the usage of various machine-learning-based classification models (e.g. Logistic Regression, Decision Tree, Random Forest, Gradient Boosting, Support Vector Machine) in predicting MLB playoff teams using 10 features (i.e. team statistics) over 2020 to 2023. By combining the prediction from the 5 classification algorithms we considered using a majority vote, we achieved a 0.833 accuracy, 1.0 precision, 0.6 recall, 0.75 F1-Score, and 0.8 AUC – better than any other available model. Additionally, we conclude that a team's pitcher's average win/loss percentage appears to be the most influential feature in predicting whether a team will make the postseason.

As future work, we recommend (1) the investigation of including additional features in model training, (2) investigating the inclusion of additional machine-learning-based classification models, (3) a more thorough optimization of hyperparameters in the models that we considered in our model.

Lastly, all scripts that were used in data analysis as well as our dataset can be found in the following repository: https://github.com/scglaze/CSE432-532-Group-Project

References

[1] A. S. Yaseen, A. F. Marhoon, and S. A. Saleem, "Multimodal Machine Learning for Major League Baseball Playoff Prediction," Informatica, vol. 46, no. 6, Jul. 2022, Accessed: Mar. 04, 2024. [Online]. Available: https://www.informatica.si/index.php/informatica/article/view/3864/1786

[2] Lingfeng Bao, Zhenchang Xing, Xin Xia, David Lo, and Shanping Li. 2017. Who will leave the company? a large-scale industry study of developer turnover by mining monthly work report. In Proceedings of the 14th International Conference on Mining Software Repositories (MSR '17). IEEE Press, 170–181. https://doi.org/10.1109/MSR.2017.58

[3] Lei Yu and Huan Liu. 2003. Feature selection for high-dimensional data: a fast correlation-based filter solution. In Proceedings of the Twentieth International Conference on International Conference on Machine Learning (ICML'03). AAAI Press, 856–863.

[4] Zhihao Chen, Barry Boehm, Tim Menzies, and Daniel Port. 2005. Finding the Right Data for Software Cost Modeling. IEEE Softw. 22, 6 (November 2005), 38–46. https://doi.org/10.1109/MS.2005.151

[5] Anthony Castrovince, "Pitch timer, shift restrictions among announced rule changes for '23", https://www.mlb.com/news/mlb-2023-rule-changes-pitch-timer-larger-bases-shifts

[6] These countries have the most influential cultures — U.S. news, US News, Accessed: Mar. 4, 2024. [Online]. Available: https://www.usnews.com/news/best-countries/rankings/influential-culture.

[7] M.-L. Huang and Y.-Z. Li, "Use of Machine Learning and Deep Learning to Predict the Outcomes of Major League Baseball Matches," Applied Sciences, vol. 11, no. 10, p. 4499, May 2021, doi: https://doi.org/10.3390/app11104499.

[8] J. M. Karnuta et al., "Machine Learning outperforms regression analysis to predict next-season Major League Baseball player injuries: epidemiology and validation of 13,982 player-years from performance and injury profile trends, 2000-2017," Orthopaedic Journal of Sports Medicine, vol. 8, no. 11, p. 232596712096304, Nov. 2020, doi: https://doi.org/10.1177/2325967120963046.

TABLE V
FEATURE IMPORTANCE FOR LOGISTIC REGRESSION, DECISION TREE, RANDOM FOREST, AND GRADIENT BOOSTING MODELS

	Logistic Regression	**Decision Tree**	**Random Forest**	**Gradient Boosting**
Defensive Efficiency	0.0259	0.0420	0.0607	0.0900
Fielding Percentage	0.0177	0.0663	0.0552	0.0454
Win-Loss Percentage (Pitching)	0.6735	0.8007	0.3556	0.7621
Hits Allowed Per 9 Innings	-1.5383	0.0229	0.0756	0.0014
Homeruns Allowed Per 9 Innings	-0.7312	0.0138	0.0319	0.0109
Walks Allowed Per 9 Innings	-1.0292	0.0168	0.0811	0.0229
Strike Outs Per 9 Innings	0.0312	0.0000	0.0242	0.0068
Strike Outs / Walks (Pitching)	1.0312	0.0000	0.1455	0.0047
Runs Scored Per Game	1.8562	0.0000	0.0986	0.0160
Team On Base Percentage	0.0653	0.0374	0.0717	0.0399