Predicting MLB Hall of Fame Players

Nicholas Glaze
Rice University
nkg2@rice.edu

Artun Bayer
Rice University
ab116@rice.edu

Figure 1: Seattle Mariners at Spring Training, 2010.

ABSTRACT

Election to Major League Baseball's Hall of Fame is an honor reserved for only the greatest to ever play the game. In this paper, we present an evaluation of several machine learning models that predict whether a player will be elected to the Hall based on their batting statistics. In particular, we explore the advantages of using LSTM models to make predictions on samples formatted as time series, and extend on previous work by training models to predict Hall of Fame election probabilities for current players, in addition to those who have already retired.

KEYWORDS

machine learning, sports analytics, neural networks, time series, LSTM

1 INTRODUCTION

Each year, the Baseball Writers' Association of America (BBWAA) receives a ballot of recently-retired Major League Baseball (MLB) players who are eligible for election to the Hall of Fame (Hall, or HoF). Each writer votes "Yes" or "No" for each player, and any player receiving at least 75% "Yes" vote is elected to the Hall. Generally, votes are based on a player's statistics, awards, and other accomplishments throughout his career, but there are no set criteria for election to the Hall. [2]

In this paper, we assert that a machine learning model can be developed to effectively simulate the Hall of Fame election process. We train several models and report on their performance, ultimately delivering a model that can predict a retired player's Hall of Fame election with 96% accuracy, and any player's election with 89% accuracy. Furthermore, we explore how the use of a time series data format affects model performance, especially on players who are still early in their career.

Our model is very applicable to the real world: since it outputs a probability of whether a player will make the Hall of Fame, it can be used to produce a rough ranking of any set of current or former MLB players. Many sports writers make money by creating such rankings [4] [9] [10], and our model could supplement their work by offering novel insights which, unlike those of sports writers, are completely quantitative and relatively unbiased.

2 PROBLEM STATEMENT

Given a matrix X of dimension (n years, m statistics) representing a player's aggregate statistics per year, develop a function $\mathbf{y} = \mathbf{f(X)}$, $0 \leq y \leq 1$, where $\mathbf{y}$ represents the probability that the player with statistics X will be elected to the Hall of Fame after they retire.

If $y \geq 0.5$, classify the player as "Hall of Fame" ("1"). If $y < 0.5$, classify the player as "Not Hall of Fame" ("0").

This is a binary classification problem.

3 DATA PREPROCESSING

We use a database of baseball statistics (1871 - 2015) found on Kaggle for this project. [8] Tables from this database are referenced by their **.csv** filenames; **Batting.csv** is the primary statistics database, while **Appearances.csv** and **Hall_of_Fame.csv** are supplementary.

Generally, we chose statistics that are both considered standard [1] and present in most rows of the dataset. A glossary of the statistics we used can be found in Appendix **Section 10.1**.

3.1 NaN Removal

This dataset has many missing values. They appear for one of two reasons:

- The statistic was not counted in that year (only in later years)
- It is part of a collection of rows in the late 1900s in which all statistics are missing

DSCI 303: Machine Learning for Data Science, Fall 2020, Rice University

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

We eliminate these missing values with the following steps:

(1) Drop any rows with missing "AB" value
(2) Drop all rows with year < 1913
(3) Remove all players who, according to **Appearances.csv**, pitched in over half of their games (these are pitchers, who are not judged for HoF election based off their batting statistics

3.2 Normalization

When the BBWAA votes on a player's election to the Hall of Fame each year, they do so by comparing them to other players who played in the same years as the player. To properly simulate the BBWAA elections, our models must do the same. Thus, we normalize each row of **Batting.csv** by z-score against only rows with the same year. That is, we normalize each sample against only its competitors in that year, instead of over the entire table.

This is especially important since the average value of any statistic can drastically change over time (see appendix **Figure 6** for example). By normalizing per year, we account for these trends, ensuring that each player is compared only to their competitors when making predictions, just as the BBWAA does.

In the normalized table, if a player has a value of b for a statistic in a given year, that means they performed b standard deviations above that statistic's average value for that year.

After normalization, **Batting.csv** has around 40k samples, each with a player_id, year, and 13 statistics.

3.3 Sample Formatting

3.3.1 Windowing. After normalizing **Batting.csv**, we then create samples for modeling. To enable our model to make accurate predictions on partial career statistics (i.e. players who haven't finished playing), we use a windowing method to produce several samples from each player's stats. Given the statistics for a player that played for k years, we create k distinct samples by taking the first 1, 2, 3, ... k years of their career, respectively. That is, if a player played from 1940 - 1944, 5 samples would be created from his statistics: 1940 - 1940, 1940 - 1941, ..., 1940 - 1944.

From the 8,739 unique players in our normalized table, this windowing method produces **40,630 samples**.

3.3.2 Time series format. Finally, since our models should be invariant to time period and player identity, we drop the year and player_id columns from all samples, making each sample an (n-year, $(m-2)$-statistic) matrix. We then zero-pad the row axis of each matrix so they all have dimension (25 years, 13 statistics) This is the **time series** format.

3.3.3 Aggregate format. The **aggreagate** format samples are then created by taking the column-wise sum of each time series sample, and adding a feature *years_played*, the number of years of statistics present in this sample. Thus, each aggregate sample is a single $(m-1)$-statistic vector.

3.3.4 Labeling. We then create two separate sample sets from the same 40,630 samples: one in **time series format**, and another in **aggregate format**.

Each sample is then labeled with a "1" or "0" based on whether **Hall_of_Fame.csv** indicates that the corresponding player_id was elected to the HoF by the BBWAA. Players elected by the Veterans Committee or other groups are labeled as "0" for this project. Futhermore, some players had clearly HoF-caliber statistics, but were caught using steroids and thus not elected to the HoF. We set these players to "1", since, if it were solely based on statistics, they would surely be in the HoF.

3.3.5 Justification. We use these two formats to evaluate whether the preservation of **career trajectories** in time series modeling offers an increase in performance despite its greater complexity. For example, if a sample's statistics are steadily increasing each year so far in their career, will a time series model evaluate them differently than a sample with same aggregate statistics but constant trajectory, and can this differentiation improve performance? In the aggregate sample set, these two samples would be identical, so this relationship would be lost, but the models would be simpler and potentially less prone to overfitting.

3.3.6 Note: We exclude all players that had one or more rows removed during **Section 3.1 NaN Removal**. For example, if a player played from 1910 - 1920, the first 3 years of his career would have been dropped in **Step 2** of NaN removal. This player would thus be excluded from testing, since we designed our models to work only on samples starting at the beginning of a player's career.

3.4 Class imbalance

This dataset suffers from severe **class imbalance**: only around 5% of samples are "1"s. We address this issue in the following sections.

4 MODELS

We evaluate a variety of models in this paper. Our train / validation / test splits were 60%/10%/30% by class and player; that is, we partitioned the set of player_ids by label, then further partitioned these sets into train, validation, and test sets. We then added all samples for each player to their assigned set. We did this to ensure that the model did not "memorize" any of the test samples during training, which would occur if a player's samples were split between the sets. Since this is a classification problem, we train all models using cross-entropy loss.

The following is a list of the models we used, with relevant hyperparameters. "Balanced class weights" refers to a 0.05 weight ratio between the "0"s and "1"s. That is, classifying a "1" sample correctly is 20 times more important than classifying a "0" sample correctly.

4.1 Aggregate format models

4.1.1 Random Forest. We used a random forest model consisting of 100 trees, using balanced class weights.

4.1.2 Support Vector Machine. We used the default SVM of sklearn with the balanced class weights. The default SVM uses the *radial basis function* to perform the kernel trick.

4.1.3 Deep Neural Network. We trained a 3-layer fully connected neural network with 16 hidden units in each layer. We used a weighted cross entropy error that is described by the below equation

to overcome the imbalance in the classes:

$$Loss_i = \sum_k -y_k^{(i)} ln(\hat{y}_k^{(i)}) 0.25 - (1 - y_k^{(i)}) ln(1 - \hat{y}_k^{(i)})$$

The weighing coefficient we used above on the positive examples is 0.25, which decreases the false positive count for each class and increases the precision.

4.2 Time series format models

4.2.1 Random Forest. We used a random forest model consisting of 100 trees, using balanced class weights. We flattened each time series sample to a single vector of length (25 years) * (13 statistics) = 325 elements.

4.2.2 k-Nearest Neighbors. We trained a kNN model by using the Dynamic Time Warping distance metric and excluding 75% of the "0" samples. We applied no class weights to this model.

4.2.3 ROCKET. We used the ROCKET convolutional kernel transform [6] to transform each sample, then fit a logistic regression classifier to the data, using SGD and balanced class weights.

4.2.4 LSTM. Our LSTM model architecture consisted of a 20-unit LSTM layer, followed by a 50-unit dense layer with ReLU activation and an output layer with softmax activation. We trained it for 50 epochs with the Adam learning rate optimizer.

4.3 Ensemble model

We found that the LSTM and neural network were the best time series and aggregate models, respectively, so we created an ensemble model by averaging the outputs of these two models, with equal weights.

5 MODEL EVALUATION

5.1 Initial Accuracies

5.1.1 Methodology. Our first step in evaluating our models was to compute a variety of basic accuracies on the test set, as shown in **Table 1**. In addition to overall and class-specific accuracies, we computed class-specific accuracies based on **full-career samples** only; that is, samples that represent a player's complete career. The set of full-career samples is interesting because they are the samples that our real-world "classifier", the BBWAA, classifies on. Therefore, the theoretical maximum accuracy for classifying these samples is 100%. This is not the case for mid- or early-career samples, since the BBWAA will learn additional information (i.e. stats for the remainder of that player's career) before making a classification. We can also expect models to perform best on these full-career samples, since the circumstances under which these samples are classified are most similar to those for the BBWAA.

5.1.2 Interpretation. Based on **Table 1**, the LSTM and NN models appear to be the best (apart from the ensemble), since they significantly outperform the other models on "1" samples, while still performing well on "0"s. It is interesting to note that, while the LSTM model performs similarly on full-career "1" and "0" samples, the NN is much more polarized, with a 19% differences in accuracy between the classes.

Table 1: Basic Model Accuracies

Model	Overall	1s	0s	Full-career 1s	Full-career 0s
RF (agg)	**96**	31	99	68	99
SVM (agg)	94	21	99	11	**100**
NN (agg)	94	54	97	79	98
RF (ts)	95	12	**100**	18	**100**
ROCKET (ts)	93	14	97	26	97
KNN (ts)				18	93
LSTM (ts)	83	**77**	83	**92**	93
Ensemble (ts+agg)	89	71	90	**92**	96

The other models' poor performance is likely due to their inability to account for the extreme class imbalance in the data, despite our hyperparameter tuning. These models all perform very well on the "0" samples, while performing poorly on the "1" samples (many false negatives). This implies that these models are very predisposed to predict "0"; although this strategy results in high accuracy numbers, these models are not useful.

Although these results are generally useful, the extreme class imbalance necessitates the use of more advanced metrics to evaluate these models' overall performance. These are explored in the following section.

5.1.3 Note. The kNN model peformed poorly on this initial evaluation, and took incredibly long to train due to the large size of our dataset. Since we knew it would not be the best model, we excluded it from more advanced evaluations.

5.2 Advanced model metrics

As shown in **Figure 2**, we found that the most robust, elucidating model metric was the mean of several classification performances metrics: F_1 score, Jaccard score, ROC AUC score, and (1 - Brier score). See Appendix **Table 6** for numeric mean metric scores.

5.2.1 Figure 2 x-axis. The x-axis in **Figure 2**, "Years before end of career", partitions samples based on how many years before the end of a player's career a given sample ends. That is, if a player's career was from 1940-1945, the sample (1940 - 1942) would fall under x = 3, while (1940 - 1945) would fall under x = 0. Clearly, if a player played only 5 years, his samples would fall under the range $x \in \{0, 1, ..., 5\}$, so larger x values will have less samples, and every player's entire career will be represented in x = 0; thus, x = 0 represents the full-career samples from **Table 1**.

Essentially, this partition scheme causes smaller x-values to represent samples for which we know more about the player's ultimate full-career stats (what the BBWAA knows), while larger x-values represent samples for which we know less. Thus, it is expected that model scores will be higher for smaller x-values, since early-career samples are generally much less predictive than late-career ones. For example, if a player is on pace to make the HoF after 5 years, but then gets injured and never plays again, he would surely not make the Hall of Fame. However, if the player maintains HoF pace for 19 years, and then gets injured and retires, he would almost surely still make the HoF, since he had already accumulated the

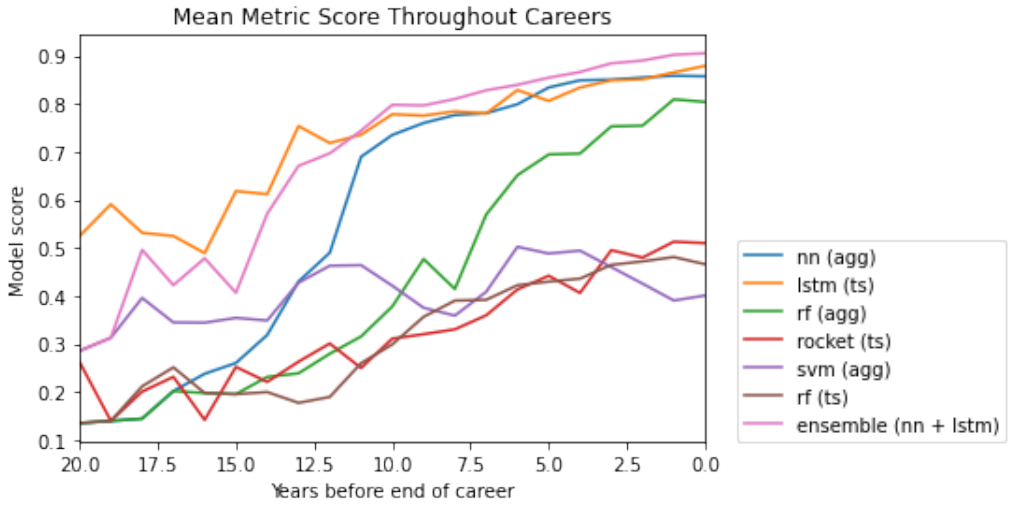

Figure 2: Means of F_1, Jaccard, ROC AUC, and (1 - Brier) scores for each model

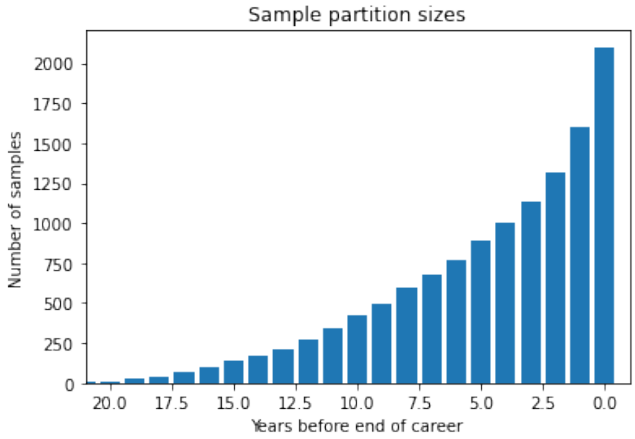

Figure 3

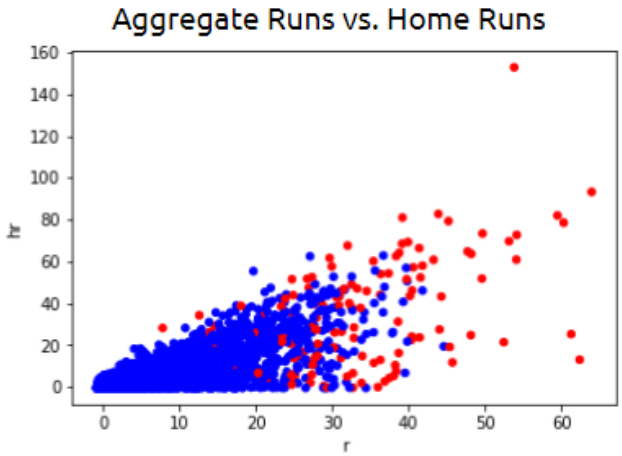

Figure 4: Blurry decision boundary; red: 1, blue: 0

statistics needed to be elected. Thus, even a perfect model likely would deliver mediocre performance on early-career samples.

5.2.2 Partition sizes. **Figure 3** shows how many samples are present at each x-value. Although scores for $x \geq 15$ might be less accurate due to low sample sizes, those for $x < 15$ are likely valid.

5.2.3 ROCKET, SVM, and time series RF. It is clearly noticeable in **Figure 2** that, by the mean metric, the ROCKET, SVM, and time series RF models generally perform poorly, with only the SVM giving reasonable performance on early-career samples. Thus, they are not the models of choice for this project. These models likely performed poorly due to their simple architectures; as evidenced in **Figure 4**, the decision boundary for this dataset between runs and home runs, two of the most important features, is not clean, so these more basic models likely underfit the data. Also, as seen in **Section 5.1.2**, these models produce lots of false negatives, which are properly penalized by this mean metric.

5.2.4 Aggregate RF. The aggregate format RF model, despite being a very simple model, performs relatively well on late-career samples, comparable to the performance of the LSTM and NN. However, its performance is much poorer on early- and mid-career samples. If this project's goal was quick, decent classification results on retired players (e.g. predicting the results of the 2020 HoF election), the aggregate random forest would be a decent choice; however, our goal is robust classification at any career stage, so this model is not the best choice.

5.2.5 LSTM and Neural Network. On late- and mid-career samples ($x \leq 10$), the LSTM (time series) and NN (aggregate) perform about equally well. From this, we can conclude that the preservation of career trajectories does not provide a significant performance

advantage when most of the information about a player's career is already known.

However, on early-career results (x > 10), the LSTM performs almost twice as well as any other model, while the NN performs incredibly poorly. This implies that the LSTM did successfully learn career trajectories, using them to make much more accurate early-career predictions than the NN (the best aggregate model). This is a very important result, as it shows that **the capturing of a career's trajectory is critical to effective early-career HoF prediction**.

The LSTM's high performance is relatively consistent across all values of x, all other models suffer a sharp decrease in performance as x increases, while the LSTM's is much more gradual. Although the LSTM's performance still decreases for large x, this is to be expected, as explained in **Section 5.2.1**. Therefore, we conclude that the LSTM, trained on time series-formatted samples, is the best standalone model for Hall of Fame prediction.

5.2.6 Ensemble. When HoF probabilites predicted by the LSTM and NN are averaged, the resulting ensemble model performs better than both single models on mid- and late-career samples (x *leq* 10). This is likely due to the fact that combining both individual models reduces much of the variance and overfitting present in each model; this results in a slight reduction of the resulting ensemble's bias.

On early-career samples, the ensemble performs worse than the LSTM, due to the NN's sharp decrease in performance for for x > 11. This is expected; if a high-performing model is averaged with a low-performing one, the result will be somewhere in between. Here, the NN is not accurate enough for any bias or overfitting reduction to have a net benefit against the large increase in bias when using the ensemble.

5.3 Selecting a model

For early-career samples, the LSTM is the clear best choice, and for late-career samples, the ensemble slightly beats the LSTM and NN. However, at times, it can be difficult to know whether a player has less than 10 years of his career left (Ensemble is better) or more than 10 (LSTM is better). This is another classification problem in and of itself, but we believe the following decision rule is a good approximation, based on our prior knowledge of the sport:

- If a player is retired, use the ensemble.
- If a player is not retired, use the LSTM.

In the real world, this strategy is feasible; players generally announce their retirement before the start of the next season, so it's easy to accurately categorize players as retired or active.

5.4 Final model performance

Table 2 and **Figure 5** show our test accuracy numbers, obtained by following the above rule. We are very satisfied with this performance, since we obtained very high accuracies on the full-career samples and relatively high accuracies overall.

Interestingly, 60% of the false positives predicted by the ensemble were on the Hall of Fame ballot; that is, they were close to the decision boundary between election and non-election. This is a good sign, as it shows that over half of our model's false positives were "reasonable" predictions; if these players had performed slightly better, they likely would have been "1"s instead of "0"s.

Table 2: Accuracies using Ensemble/LSTM Decision Rule

Overall	1s	0s	Full-career	Full-career 1s	Full-career 0s
89%	76%	89%	96%	92%	96%

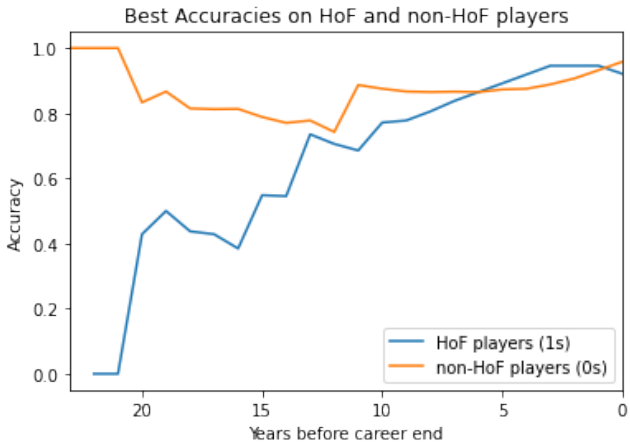

Figure 5: Using ensemble on retired players and LSTM on active players

5.5 Feature importance

The aggregate random forest was the best model we trained that computes feature importance, so we report these importances in **Table 5**. The learned importances generally align with what would be expected (based on our prior knowledge).

We experimented with using PCA or other techniques to reduce the number of statistics; however, these methods did not offer drastic reductions. This, combined with the messiness of our decision boundary, led us to decide to use all statistics in order to maximize the amount of variance available in our dataset.

6 COMPARISON TO RELATED WORK

We found two similar projects to ours [7] [11]. Our methodology differs from others in that (1) other models work only on retired players, while our model can make predictions at any career stage, and (2) others used simpler, non-deep learning models (RF, XGBoost) on aggregate samples, while ours used deep learning and multiple sample formats.

On full-career samples, our models outperformed those in other projects, as seen in **Table 3**. Furthermore, our model improves on this work by enabling predictions on current players. Other projects' models could only be used to, for example, predict who from the 2020 HoF class will be elected; our model can do that and predict whether any active player will eventually make the Hall of Fame at the end of their career.

Table 3: Comparison to Related Work on Full-Career Samples

Project	Model	ROC AUC	TP rate	FN Rate	FP Rate
Ours	NN+LSTM	**0.97**	**0.92**	**0.08**	**0.04**
"Using ML to..." [11]	RF	0.93			
"Baseball Analytics" [7]	Log Reg		0.88	0.12	0.06

Table 4: LSTM Predictions on Current Players

Player	Career stage	Expert opinion [9]	LSTM-predicted HOF probability (%)	Makes sense?
Mike Trout	Mid	Best active player	96 (HoF)	Yes
Mookie Betts	Mid	4th-best active player	93 (HoF)	Yes
Nolan Arenado	Mid	9th-best active player	2.4 (Not HoF)	Somewhat
Brett Gardner	Late	Solid player (but not great), long career	0.02 (Not HoF)	Yes
Fernando Tatis Jr.	Very early	Had MVP-caliber second season	76 (HoF)	Yes
Aaron Judge	Early	Young, but already top-20 player	86 (HoF)	Yes

Table 5: Feature Importances in Aggregate Random Forest

Feature	Importance
R	0.128
H	0.092
D	0.092
HR	0.089
RBI	0.082
SO	0.072
AB	0.064
G	0.062
T	0.061
BB	0.055
SH	0.052
Years Played	0.052
SB	0.049
HBP	0.049

7 CONCLUSION

7.1 Real-world example: Current players

To demonstrate our model, we normalized a database of batting statistics from 2010 - 2020 [3], selected several current players whose predictions we thought would be interesting, and used the LSTM to predict whether they will make the Hall of Fame once they retire. **Table 4** shows our results, along with a short, qualitative description of each player [9].

Generally, these results make sense. As expected, older players like Brett Gardner have more decisive predictions, since more information about their complete career is already known, while younger players like Aaron Judge have less decisive predictions. The only confusing prediction is Nolan Arenado's; he's a great player, and he will likely be on the HoF ballot; but was given almost zero chance of making. The decisiveness of the prediction makes sense, but we expected him to be projected as a Hall of Famer. Maybe the model will end up being right, though; we'll have to wait and see.

7.2 Final thoughts

In this paper, we presented a model that can predict whether any MLB player, current or retired, will be elected to the Hall of Fame. This model is incredibly useful, since it can be used to rank any set of MLB players across any time period, in terms their Hall of Fame election probability.

Furthermore, in creating this model, we analyzed several classification algorithms, and confirmed the important result that the trajectory relationships present in time series data can be leveraged to train superior classification models. Because of this, we were able to develop a model that improved significantly on existing work, and we currently have the most generalizable publicly available model to solve this problem.

8 CONTRIBUTIONS

(1) Artun Bayer: Aggregate format preprocessing, modeling, evaluation, and reporting
(2) Nicholas Glaze: Time series format preprocessing, modeling, evaluation, and reporting; report introduction and graphics

9 IMPLEMENTATION

Our code is available at the following Github repository. [5]

REFERENCES

[1] http://m.mlb.com/glossary/standard stats. [n.d.].
[2] https://baseballhall.org/hall-of-famers/rules/bbwaa-rules-for election. [n.d.].
[3] https://baseballsavant.mlb.com/leaderboard/statcast. [n.d.].
[4] https://bleacherreport.com/articles/2873529-ranking-mlbs-25-best-players-under-25-entering-the 2020-season. [n.d.].
[5] https://github.com/nglaze00/MLB-Hall-of Fame-Prediction. [n.d.].
[6] https://github.com/rh2835/Baseball Analytics/blob/master/Baseball%20Analytics-part%202.ipynb. [n.d.].
[7] https://github.com/rh2835/Baseball Analytics/blob/master/Baseball%20Analytics-part%202.ipynb. [n.d.].
[8] https://kaggle.com/seanlahman/baseball. [n.d.].
[9] https://www.espn.com/mlb/story/_/id/28839375/espn-mlb-rank-100-1-baseball-top-players 2020. [n.d.].
[10] https://www.mlb.com/news/mlb-young-star-power-rankings 2020-1. [n.d.].
[11] http://www.baseballdatascience.com/using-machine-learning-to-predict-baseball-hall-of famers/. [n.d.].

10 APPENDIX

10.1 Statistics Glossary

The following are the full names of each statistic from **Batting.csv** used for modeling:

- G: Games played
- AB: At Bats
- R: Runs scored
- H: Hits
- D: Doubles
- T: Triples
- HR: Home Runs
- RBI: Runs Batted In
- SB: Stolen Bases
- BB: Walks
- SO: Strikeouts
- HBP: Times Hit By Pitch
- SH: Sacrifice Hits

10.2 Home runs over time

The average value per player, per year, of many baseball statistics has changed drastically over time. For example (see **Figure 6**), during the 1900s, the number of home runs tripled as baseball manufacturing techniques improved, and balls were reused less during games. It peaked in the late 1900s, at the height of the steroids era, and has declined since then as steroid use has been more heavily regulated.

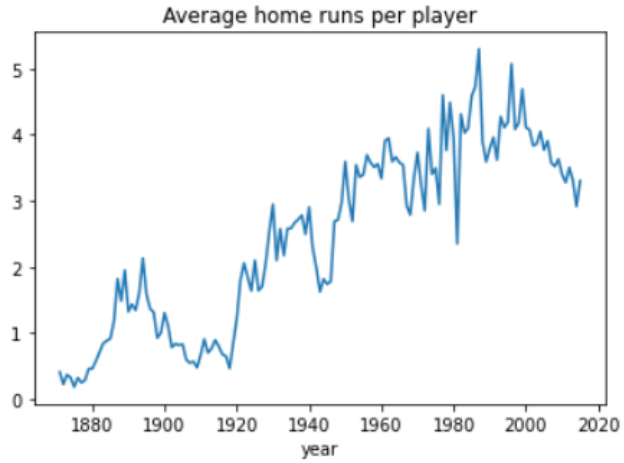

Figure 6: Yearly home run averages

10.3 Numeric mean metric scores

Table 6 shows mean metric scores of all models for x = 15 (early-career), x = 10 (mid-career), and x = 0 (full careers). The LSTM clearly outperforms other models for x = 15, while the ensemble is best for x = 10 and x = 0.

Table 6: Mean Metric Scores vs. Years Before End of Career

Model	x = 15	x = 10s	x = 0
RF (agg)	0.20	0.38	0.80
SVM (agg)	0.35	0.42	0.40
RF (ts)	0.20	0.30	0.47
ROCKET (ts)	0.25	0.31	0.51
NN (agg)	0.26	0.74	0.86
LSTM (ts)	**0.62**	0.78	0.88
Ensemble	0.41	**0.80**	**0.91**