Predicting Baseball Hits

AI Boot Camp Project 2

https://github.com/ReptilianRex6/Project-2.git

Team Members: Ethan Bennett, Zac Baxter, Patrick Peters, Aleksandar Belic, & Seneca Moore

Project Overview

Project Purpose / Description

- Explore supervised machine learning prediction and modeling in the data-rich field of American major league baseball.
- We thought it would be fun to predict the World Series champion, but found it would take a lot more than a 2 week project to accomplish this. So we decided to predict hit's instead.
- Future work would be to make the model better to predict future outcomes based on historical data, team statistics, player performances, weather conditions, and even park factors, to predict the outcomes of future games.

Project Overview

Goals/Questions to be addressed

Primary goals utilizing historical performance data and advanced statistical modeling techniques:

- Aim 1: To create an optimized supervised machine learning model to predict baseball hits.
- Aim 2: To predict the number of hits baseball players will achieve in the upcoming year.

Project Overview

Approach taken to achieve goals

- Set up Github & project management
- Collected and evaluated data
- Formulated & revised research questions
- Cleaned data
- Supervised learning modeling
- Results
- Evaluation and discussion

Data Collection

Overview of data collection, cleanup and exploration process

- Sources: Pybaseball library, which extracts baseball data from Baseball Reference, Baseball Savant, and FanGraphs.
- Scope: Focused on players detailed batting stats with at least 50 plate appearances annually, spanning 2014-2019.
- In addition to batting stats, we integrated player speed data into our dataset to more holistically predict next season's hits.

Data Preparation

Overview of Data Cleanup and Preparation

Merged Data

- Combined batting stats with speed metrics to create a holistic dataset.

Cleaning:

- Removed Unnecessary Columns:
  - Excluded columns not relevant to our analysis goals.
  - Excluded data related to the 2014 season (speed metric start from 2015)
  - Drop players appearing in only one season to maintain consistency and relevance.

Preparation:

- Converted all data types to numeric (excluding object types) to facilitate analysis.
- Handled Missing Data: Identified and filled missing values, ensuring a complete dataset for modeling.
- Drop data that caused the data leakage

Key Features

Key Features for Model Training (Aim 1):

- **Batting Metrics:** HR (Home Run), RBI (Runs Batted In), SB (Stolen Bases), BB (Base on Balls, commonly known as Walks), SO (Strikeouts), OBP (On-Base Percentage), SLG (Slugging Percentage), OPS (On-base Plus Slugging).
- **Advanced Stats:** wRAA (Weighted Runs Above Average), Barrels (A batted ball event whose comparable hit types have led to a minimum .500 batting average and 1.500 slugging percentage), IFFB (Infield Fly Ball), wOBA (Weighted On-Base Average), WAR (Wins Above Replacement), BABIP (Batting Average on Balls in Play), LD% (Line Drive Percentage), GB% (Ground Ball Percentage), Contact% (Contact Percentage), SwStr% (Swinging Strike Percentage).
- **Physical Metrics:** Competitive Runs, Age
- **Target Variable:** Next Year Hits (next_year_hits)

Result/Conclusion 1.

Aim 1: Which model produces the best prediction for next year's hits?

Model	Training Results	Testing Results
GradientBoostingRegressor	MSE: 28.555884873542396 MAE: 3.94707028144996 R^2: 0.9894306570044707	MSE: 398.34215931956476 MAE: 13.657809975591451 R^2: 0.8506441710830508
RandomForestRegressor	MSE: 103.18634560737527 RMSE: 10.158068005648282 R^2: 0.961807806551634 8	MSE: 724.0826046420825 RMSE: 26.90878303903918 R^2: 0.728509887566525
XGBoost (with hyperparameter tuning)	MSE: 98.98159327621012 MAE: 7.461078889288994 R^2: 0.963284627036945 2	MSE: 400.07088433530623 MAE: 14.856202112374234 **R^2: 0.8514606534409036**
LinearRegression	MSE: 0.3566850316963296 MAE: 3.84707028366217 R^2 Score: 0.9439306570066241	MSE: 0.3514193381286068 MAE: 17.693551056337913 R^2 Score: 0.7506441749301639

Result/Conclusion 1 cont.

Aim 1: Feature Importance Analysis

Model	Feature Importance Analysis Top Predictors
GradientBoostingRegressor **RandomForestRegressor**	SO: Strikeouts CSW%: Called Strike plus Whiff Rate LD: Line Drives Events: Batted Ball Events wRAA: Weighted Runs Above Average Barrels: Barreled Balls RBI: Runs Batted In IFFB: Infield Fly Ball Rate R: Runs competitive_runs: Competitive Runs
XGBoost (with hyperparameter tuning)	Line Drive Percentage (LD%), Ground Ball Percentage (GB%), Batting Average on Balls In Play (BABIP), and On-Base Percentage (OBP)

Key Features

Key Features for Players' Next-Year Hits (Aim 2):

- Primary modeling dataset
- Age, GB (Ground ball), LD (Line Drive), and SwStr% (Swing Strike Percentage)

Result/Conclusion 2 cont.

The script fetches baseball statistics from the years 2014 to 2018 using the batting_stats function from pybaseball.

The train_model function is defined. This function takes in the baseball data and a machine learning model as inputs. It separates the data into features (X) and the target variable (y). The features are 'Age', 'GB', 'LD', and 'SwStr%', and the target variable is 'H' (Hits). It then splits the data into training and testing sets, trains the model on the training data, and returns the trained model along with the testing data.

```
# Load in the data for each model
data = batting_stats(2014,2018)

def train_model(data, model):
    X = data[['Age', 'GB', 'LD', 'SwStr%']]
    y = data['H']
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=0)
    model.fit(X_train, y_train)
    return model, X_test, y_test

rf_model_instance, _, _ = train_model(data, RandomForestRegressor(n_estimators=128, random_state=78))
gb_model_instance, _, _ = train_model(data, GradientBoostingRegressor(n_estimators=190, learning_rate=0.1, max_depth=8, random_state=42))
```

Result/Conclusion 2 cont.

```
# Get the data for the current year
data_current_year = batting_stats(2019)

# Select 5 random players
random_players = data_current_year['Name'].sample(5)
```

The script randomly selects five players from the data DataFrame.

```
# Iterate over the selected players
for player in random_players:
    # Get the player's data
    player_data = data_current_year[data_current_year['Name'] == player]

    # Remove the 'Name' column
    player_data = player_data.drop('Name', axis=1)

    # Select only those columns from player_data
    player_data_rf = player_data[['Age', 'GB', 'LD', 'SwStr%']]

    # Run the data through the models
    rf_prediction = rf_model_instance.predict(player_data_rf)
    gb_prediction = gb_model_instance.predict(player_data_rf)
```

For each selected player, the script extracts the player's data from the DataFrame, selects the features used to train the models, and uses both the Random Forest and Gradient Boosting models to predict the number of hits the player will make in the next year.

Result/Conclusion 2 cont.

The output prints the predicted number of hits from both models, as well as the actual number of hits the player made in 2019.

```
According to the Random Forest Model, Vladimir Guerrero Jr. will have around [129.296875] hits in 2019
According to the Gradient Boosting Model, he will have around [132.15728577] hits in 2019
Actual hits: 126

According to the Random Forest Model, Albert Pujols will have around [126.7421875] hits in 2019
According to the Gradient Boosting Model, he will have around [137.09431312] hits in 2019
Actual hits: 120

According to the Random Forest Model, Danny Santana will have around [135.640625] hits in 2019
According to the Gradient Boosting Model, he will have around [130.67534672] hits in 2019
Actual hits: 134

According to the Random Forest Model, Yuli Gurriel will have around [169.0625] hits in 2019
According to the Gradient Boosting Model, he will have around [169.47966012] hits in 2019
Actual hits: 168

According to the Random Forest Model, Hunter Dozier will have around [135.078125] hits in 2019
According to the Gradient Boosting Model, he will have around [134.05113251] hits in 2019
Actual hits: 146
```

Problems Encountered

- Data leakage and overfitting in our initial modeling
- Limited time and resources/What was manageable in the timeframe
  - First ideas were too ambitious and had to scale it down
  - Took some time and experimentation to get to the right target variable
- Aim 1: Discrepancies between training and testing results suggest room for model improvement.
  - Testing MSE is higher, indicating potential overfitting.
- High complexity of what goes into the target variable of a baseball hit, a “Holy Grail” target
  - Many contributing factors that are impossible to predict (weather, injuries, performance of a given pitcher on a given day, contextual decision-making that indicate high performance, such as “productive outs,” etc.)

Summary

Our project explored the potential of machine learning for predicting total hits in a baseball season. We utilized readily available data on hitters, but encountered challenges in acquiring and incorporating data on opposing pitchers and fielders, which would likely improve model accuracy.

Also, there are external factors like contract disputes, personal life and injury that can impact player performance that made ‘Hits’ as a target variable particularly difficult to predict. Perhaps, a similar metric that might be easier to model for would be Batting Average. Despite limitations in data scope, our initial models demonstrated the effectiveness of modeling based exclusively on a hitter-centric statistic performance in previous seasons.

The project highlights the exceptional amount of data available in baseball, while also underlining the potential challenges of properly coordinated datasets and feature selection in this data-rich domain.

Future Considerations

Additional Questions:

- Who will be the world series winners for 2024?

What we might research if more time was available:

- Identify top players and teams based on past performance indicators
- Expanding upon the larger dataset to fully utilize its potential
- Incorporate a live game feed for real-time prediction and modeling

Plan for future development:

- Focus on reducing overfitting and improving model generalization.
- Explore additional data sources and advanced modeling techniques.

Questions and Comments?