**Practicum I - Statistical analysis and prediction of Major League Baseball earnings**

**Overview**

As a lifelong sports fan the business and analytical sides have also been of interest to me. For this project I plan to analyze baseball batting and fielding statistics and attempt to predict a player’s combined earnings based on their individual statistics. Machine learning (ML) techniques will be utilized in multiple stages of the project, including data preparation and salary predictions. Performance metrics will then be reported for each model.

**Table of contents**

1. Description of data
2. Research notes
3. Data cleaning and preparation
4. Exploratory data analysis and visualization
5. Feature importance and selection
6. Machine learning modeling & evaluation
7. Final analysis

**Description of data**

The data used in this project is the ‘The Lahman Baseball Database 1871-2021’, and was downloaded from [seanlahman.com archive](https://www.seanlahman.com/baseball-archive/statistics). It includes complete batting and pitching statistics from 1871 to 2022, plus fielding statistics, standings, team stats, managerial records, post-season data, and more. For this project, I used the batting, salaries, fielding, and names files.

**Research Notes**

General

The target variable is earnings, and the features are all other statistics as well as any engineered features. Players will be assigned a single position based on their most frequently played position and ‘designated hitter’ was not an option in the dataset. Team affiliation will not be considered.

Scoring metrics

For each ML model utilized, a R-squared (R^2) value and Mean Absolute Error (MSE) or Negative Mean Absolute Error (NMAE) will be provided:

* R^2 is equal to Pearson’s r, squared. While Pearson’s r is a measure of the correlation between the features & target, R^2 is a measure of how well the model fits the data. It represents the proportion of the variance in the target variable that is explained by the ML model. It ranges from 0 to 1, with higher values indicating a better fit
* MAE will be the primary scoring metric for this project. It measures the average absolute difference between the predicted and actual earnings totals. It will provide insight as to how far off the predictions are on average, regardless of whether it’s an over or under prediction
* NMAE is the negative value of MAE. Certain models required this be used instead of MAE

**Data cleaning and preparation**

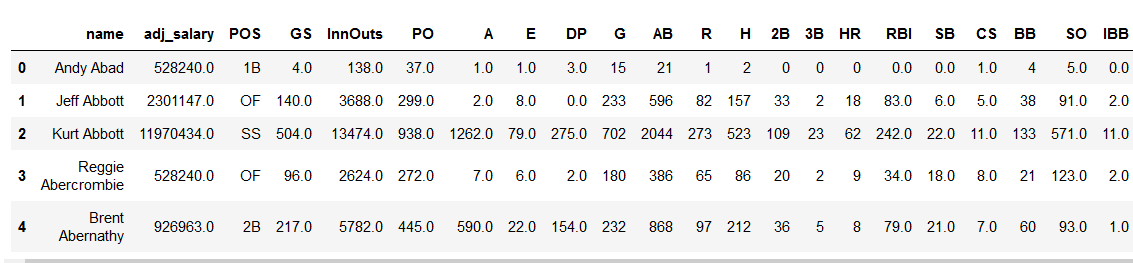
An initial inspection of the data revealed the following:

* Salaries are available from 1985 to 2016 only
* Missing data is prevalent in the batting, pitching, fielding, and names files but not the salaries file
* Each dataset has a 'playerID' that can be used for merging dataframes

Data cleaning process:

* Set timeframes for all statistical files to 1985-2016, to coincide with salaries file
* Group statistical files by player and calculate combined values
* Feature engineering
  + Defensive positions – the fielding statistics were analyzed to obtain each player’s most commonly played position
  + Adjusted earnings – inflation rate based on percentage change per year relative to 2016 total
  + Analytical statistics – plate appearances, slugging percentage, on-base percentage
* Merge dataframes – clean statistical data merged with clean salary data
* Filter positions – pitchers were removed from batting/fielding stats; fielders who occasionally pitched were removed from the pitching stats
* Missing values – rows with missing salaries were dropped as the corresponding statistics were deemed insignificant; the 7 missing positions were entered manually
* Add full names and reposition columns for visual purposes

Clean dataframe:



**Exploratory data analysis and visualization**

Multiple visuals indicated strong correlation among many variables, including the target variable (adj\_salary)

Correlation matrix:

Chart, treemap chart

Description automatically generated

Notably, two of the engineered features (OBP, SLG) had low correlation with the target variable:

Chart, line chart

Description automatically generated

RBIs & runs had the highest correlation with the target variable:

Chart, scatter chart

Description automatically generated

In this pairplot of defensive statistics, the impact of position is apparent:

Diagram

Description automatically generated

* The position-effect on correlation among offensive statistics is not as obvious:

A picture containing background pattern

Description automatically generated

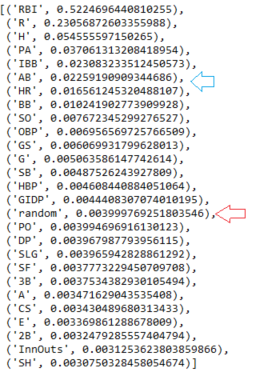
**Feature importance and selection**

With the demonstrated effect of position on defensive statistics, the position column was encoded with dummy variables and Pearson’s correlation was again used to determine correlation with the target variable. The newly encoded defensive columns showed to have the smallest correlation with the target variable, and they were removed from the dataset.

Chart

Description automatically generated

To quantify the importance of each variable, a `RandomForestRegressor()’ ML algorithm was implemented and return a list of features sorted by `feature\_importances\_` values as determined by the model. A random number column was included to identify useless columns. As indicated in the graphic, there was a steep drop after the first few columns. Although many features appeared above the 'random' column (red arrow), the threshold was set to .02 (blue arrow) and columns that fell below were dropped.



To address multicollinearity within the dataset, I checked the Variance Inflation Factor (VIF), which measures the correlation and strength of correlation between the dependent variables in regression models. The feature with the highest VIF value, plate appearances, was dropped.

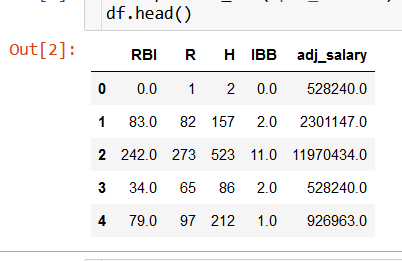
Table

Description automatically generated

**Machine learning modeling**

Supervised ML techniques were chosen to make the earnings predictions since the data is labeled and is being used to make predictions about new, unseen data. In this instance I am predicting the career earnings based on career offensive statistics.

Finalized dataframe:



* Simple linear regression – this method of linear regression attempted to predict earnings based on a single feature (RBI) and performed well despite the limited amount of data used, with a R^2 of .775 on the test data. Predictions were off by an average of about 13.7M. The model was overfit as the train score exceeded the test score by about 5%.
  + Results:

Text

Description automatically generated with medium confidenceGraphical user interface, text

Description automatically generated

Chart, scatter chart

Description automatically generated

* Multiple linear regression – this next linear regression technique used all features to predict earnings. This slightly outperformed the prior model, with a R^2 of .811 on the test data. Predictions improved and were off by an average of 12.7M. Overfitting remained with the train score exceeding the test score by about 5%.
  + Results:

Text

Description automatically generated

Chart, scatter chart

Description automatically generated

* K nearest neighbors (regression) – in the initial KNN run, the number of neighbors (k value) was set to n=3. Upon implementation the R^2 for the training data jumped to .92 but the test data dropped to .802, increasing the overfitting to an almost 12% difference. Interestingly, predictions improved again and were off by an average of 10M.
  + Results:

Text

Description automatically generated

Chart, scatter chart

Description automatically generated

* KNN tuned – to determine the optimum number of neighbors, k, I experimented with the KNN regressor algorithm and checked R^2 values for 2-29 neighbors, and found the maximum at k=26. This resulted in the train score dropping to .86 but test score increasing to .83, thus reducing the overfitting to about a 3.5% difference. Predictions improved once again, now off by an average of 9.4M.
  + Results:

Text

Description automatically generated

Chart, scatter chart

Description automatically generated

* K-Fold Cross-Validation – with the encouraging improvement thus far, I really wanted to see how far it could go. K-fold cross-validation was set to 5 folds and implemented multiple regression techniques including Linear Regression, Lasso, LassoCV, RidgeCV, ElasticNet, and KNN. Then the dataset was scaled with the `StandardScaler()` and the same set of algorithms were implemented again. Finally, boosting and bagging ensemble algorithms were implemented on both the scaled and unscaled data. Boosting methods used were Ada Boost Regressor (AB) and Gradient Boosting Regressor (GBM), and bagging methods used were Random Forest Regressor (RF) and Extra Trees Regressor (ET).

Unscaled data, regression algorithms

R^2 Negative mean absolute error

Text

Description automatically generated Text, letter

Description automatically generated

Chart, box and whisker chart

Description automatically generated Chart, box and whisker chart

Description automatically generated

* + Evaluation – KNN scored better in both metrics among regression models but the lowest MSE remains with the tuned KNN model

Unscaled data, ensemble algorithms

R^2 Negative mean absolute error

Text

Description automatically generated Text, letter

Description automatically generated

Chart, box and whisker chart

Description automatically generated Chart, box and whisker chart

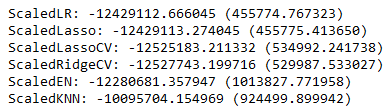
Description automatically generated

* + Evaluation – the gradient boosting regressor score better than all ensemble models and regression models in both metrics, however the lowest MSE remains with the tuned KNN model

Scaled data, regression algorithms

R^2 Negative mean absolute error

Text, letter

Description automatically generated 

Chart, box and whisker chart

Description automatically generated Chart, box and whisker chart

Description automatically generated

* + Evaluation – scaling the data did not have a significant impact on the outcomes aside from a reduction in the KNN R^2 value. Among k-fold regression algorithms, KNN scored best with scaled and unscaled data

Scaled data, ensemble algorithms

R^2 Negative mean absolute error

Text, letter

Description automatically generated Text

Description automatically generated

Chart, box and whisker chart

Description automatically generated Chart, box and whisker chart

Description automatically generated

* + Evaluation – scaling the data did not have a significant impact on the outcomes. Ada boost scored lowest in both metrics for scaled and unscaled data, while GBM scored highest for both metrics and datasets

**Final analysis**

The highest R^2 value was from the Kfold GBM model and the lowest MAE was from the tuned KNN model (k=26). The Kfold method (highlighted blue) had higher R^2 values and MAEs with the exception being the standalone KNN regressor.

After spending ample time working with this dataset, it is understandable why higher statistical totals translates to higher salaries, as I assume players who don’t put up significant numbers don’t last long in the league, and thus don’t repeatedly receive new contracts.

Machine learning results for all models:

|  |  |
| --- | --- |
| **R^2 (higher is better)** | **MAE (lower is better)** |
| GBM: 0.879 | KNN tuned: 9392546 |
| ScaledGBM: 0.877 | ScaledGBM: 9467963 |
| KNN: 0.864 | GBM: 9504770 |
| RF: 0.864 | KNN: 10095452 |
| ScaledRF: 0.863 | ScaledKNN: 10095704 |
| ET: 0.863 | KNN untuned: 10096430 |
| ScaledET: 0.861 | RF: 10245144 |
| LassoCV: 0.856 | ScaledET: 10250316 |
| ScaledKNN: 0.855 | ET: 10400949 |
| Elastic Net: 0.854 | ScaledRF: 10617952 |
| Linear Reg: 0.854 | ScaledEN: 12280681 |
| ScaledLR: 0.854 | Linear Reg: 12429112 |
| Lasso: 0.854 | ScaledLR: 12429112 |
| ScaledLasso: 0.854 | Lasso: 12429112 |
| RidgeCV: 0.854 | ScaledLasso: 12429113 |
| ScaledRidgeCV: 0.854 | RidgeCV: 12429123 |
| ScaledLassoCV: 0.852 | Elastic Net: 12429940 |
| ScaledEN: 0.838 | ScaledLassoCV: 12525183 |
| AB: 0.835 | ScaledRidgeCV: 12527743 |
| KNN tuned: 0.832 | LassoCV: 12641936 |
| ScaledAB: 0.829 | Multiple Linear Reg: 12655030 |
| Multiple Linear Reg: 0.811 | Simple Linear Reg: 13702250 |
| KNN untuned: 0.802 | AB: 15920825 |
| Simple Linear Reg: 0.775 | ScaledAB: 16666124 |