**Capstone Project**

**Predicting Salaries of Baseball Players**

***February 19, 2024***

**Table of Contents**

Data Overview (Page 3)

Data Exploration (Page 5)

Model Development, Testing and Hyperparameter Optimization (Page 9)

1. Single Decision Tree
2. Random Forest
3. Boosting
4. Linear Regression
5. Ridge Regression
6. Lasso

Analysis (Page 19)

Reflections for Further Learning (Page 20)

**Data Overview:**

The goal of the project is to predict the salaries of Baseball players using the dataset “Hitters” which is Major League Baseball Data from the 1986 and 1987 seasons. The data is referenced in the book “Introduction to Statistical Learning (ISLR)” by James, Witten, Hastie and Tibshirani and downloaded from the website <https://rdrr.io/cran/ISLR/man/Hitters.html>.

This dataset was taken from the StatLib library which is maintained at Carnegie Mellon University. This is part of the data that was used in the 1988 ASA Graphics Section Poster Session. The salary data were originally from Sports Illustrated, April 20, 1987. The 1986 and career statistics were obtained from The 1987 Baseball Encyclopedia Update published by Collier Books, Macmillan Publishing Company, New York (source: above referenced website).

The data is available as a dataframe with 322 observations and 20 columns. This includes the “Salary” column that we would like to predict using the remaining variables as predictors.

AtBat

Number of times at bat in 1986

Hits

Number of hits in 1986

HmRun

Number of home runs in 1986

Runs

Number of runs in 1986

RBI

Number of runs batted in in 1986

Walks

Number of walks in 1986

Years

Number of years in the major leagues

CAtBat

Number of times at bat during his career

CHits

Number of hits during his career

CHmRun

Number of home runs during his career

CRuns

Number of runs during his career

CRBI

Number of runs batted in during his career

CWalks

Number of walks during his career

League

A factor with levels A and N indicating player's league at the end of 1986

Division

A factor with levels E and W indicating player's division at the end of 1986

PutOuts

Number of put outs in 1986

Assists

Number of assists in 1986

Errors

Number of errors in 1986

Salary

1987 annual salary on opening day in thousands of dollars

NewLeague

A factor with levels A and N indicating player's league at the beginning of 1987

**Data Exploration:**

A screenshot of a graph

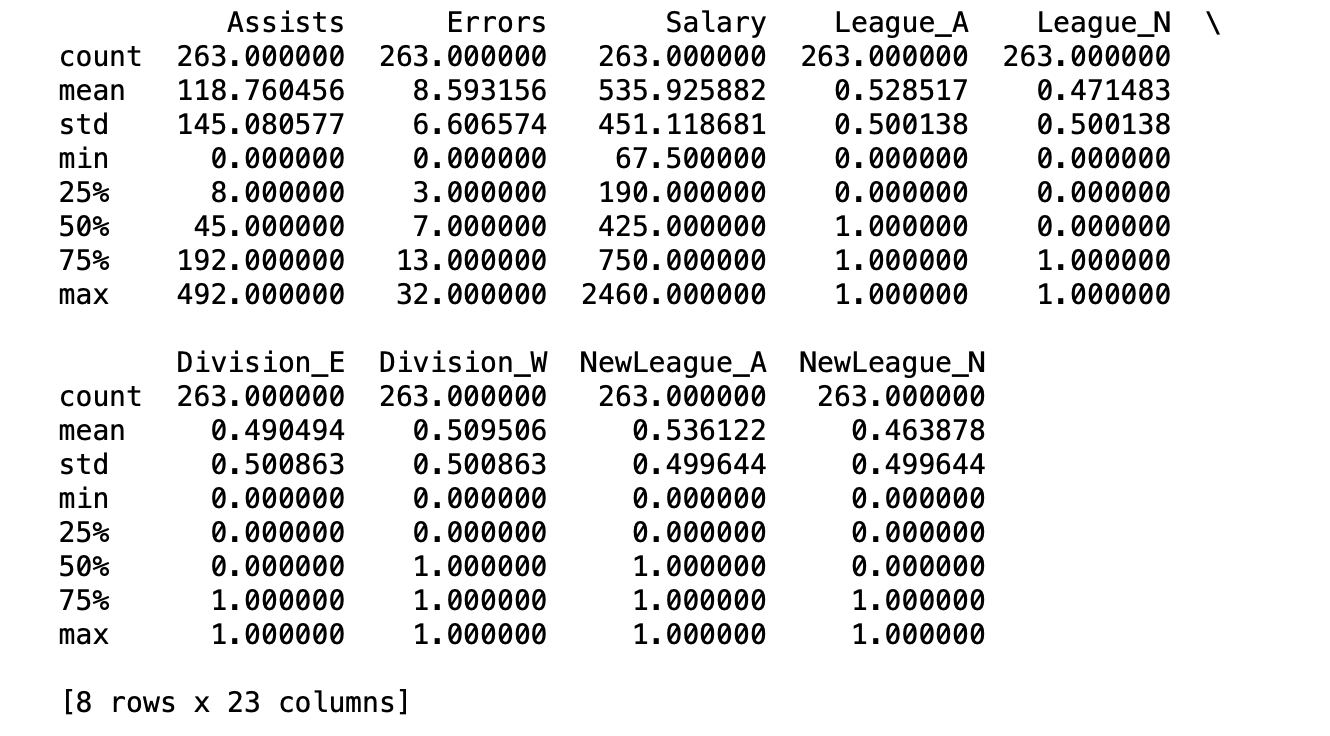
Description automatically generated

Variables like League, Division and NewLeague are categorical and are converted into numerical variables for further analysis. Also, all 59 rows with missing Salary values are removed. This leads to a new dataset with 263 rows and 23 columns. Excluding the predicted variable (Salary) and the categorical variables (League, Division and NewLeague), there are 19 numerical predictor variables.

Summary statistics of this dataset are as follows:

A screenshot of a computer

Description automatically generated



Next, we look at the distribution of the predicted variable, Salary, provided in the dataset in thousands of dollars:

A graph with blue columns

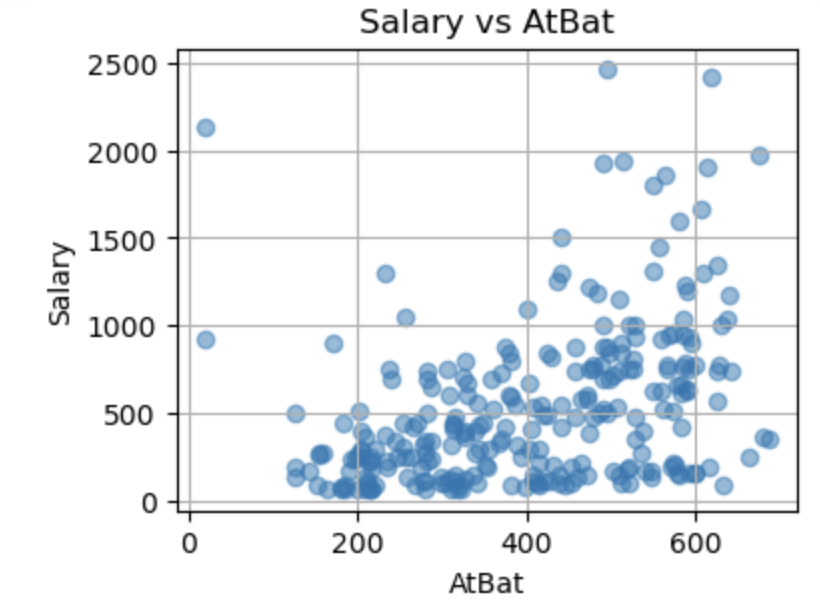
Description automatically generated

We then log-transform the Salary column so that its distribution is relatively bell-shaped.

A graph of a graph

Description automatically generated

We next plot bivariate scatter plots to explore the correlation between Salary and the 19 predictor variables. This will help shape our intuition about the relative importance of the different predictor variables when we build the regression models to predict Salary.

 A graph with blue dots

Description automatically generatedA graph with blue dots

Description automatically generatedA graph with blue dots

Description automatically generated

A graph with blue dots

Description automatically generatedA graph with blue dots

Description automatically generatedA graph with blue dots

Description automatically generatedA graph with blue dots

Description automatically generated

A graph with blue dots

Description automatically generatedA graph with blue dots

Description automatically generatedA graph with blue dots

Description automatically generatedA graph with blue dots

Description automatically generated

A graph with blue dots

Description automatically generatedA graph with blue dots

Description automatically generatedA graph with blue dots

Description automatically generatedA graph with numbers and numbers

Description automatically generated

A graph with numbers and lines

Description automatically generatedA graph with numbers and lines

Description automatically generatedA graph with numbers and lines

Description automatically generated

**Model Development, Testing and Hyperparameter Optimization:**

We now randomly split the data into 50% training data (131 rows), 25% validation data (65 rows) and 25% testing data (67 rows).

We now build six different regression models on the training data, using them to predict for the test data and calculate their accuracy relative to the actual salary using metrics like MSE and R^2.

1. Single Decision Tree
2. Random Forest
3. Boosting
4. Linear Regression
5. Ridge Regression
6. Lasso
7. Single Decision Tree: The model is built using sklearn’s DecisionTreeRegressor. The MSE outcomes on the Validation and Test data are as follows:

Validation Score: 0.5587164537125167

Validation MSE: 0.3787097926814133

Test Score: 0.7070098247833088

Test MSE: 0.2855998381359662

The plots of Actuals versus Predicted on the Validation data (65 rows) and Test data (67 rows) are as follows:

Validation Data: Test Data:

A graph with blue and orange lines

Description automatically generated A graph with blue and orange lines

Description automatically generated

Hyperparameter Optimization:

The hyperparameter, depth of the tree is optimized by iterating across different depths on the training data to identify the tree with the least MSE on the validation data. Accordingly, the tree with depth 3 is identified to be the best.

Depth 1: MSE = 0.44069589463984205

Depth 2: MSE = 0.4259261675693846

Depth 3: MSE = 0.3457787317105789

Depth 4: MSE = 0.37030165678935045

Depth 5: MSE = 0.43910202709082086

Depth 6: MSE = 0.4261640505552247

Depth 7: MSE = 0.37834128365797254

Depth 8: MSE = 0.4328382283248607

Depth 9: MSE = 0.3864998484114058

Depth 10: MSE = 0.3891024734307632

Depth 11: MSE = 0.3704849837134849

Depth 12: MSE = 0.44081624337355607

Depth 13: MSE = 0.40420925415507064

Depth 14: MSE = 0.40449280333537657

Depth 15: MSE = 0.40449280333537657

Best depth: 3

Best score on validation data: 0.5970886734149103

A graph with blue lines

Description automatically generated

The tree with optimal hyperparameter setting (depth 3) is as follows (analysis will be presented in a later section).

A diagram of a tree

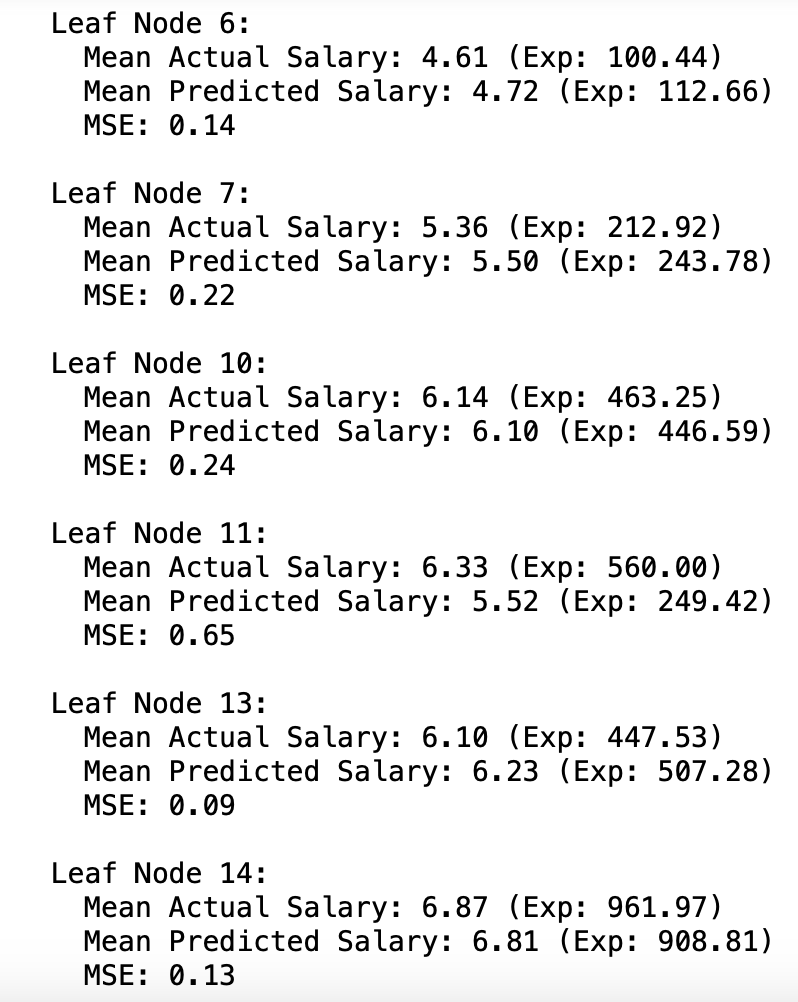
Description automatically generated

The plot of actual versus predicted Salary values on the Test data are as follows:

A graph with blue and orange lines

Description automatically generated

For each terminal node, the actual and predicted Salary, and the MSE of the log transform of Salary is shown below. In parentheses, the actual Salary in thousands of dollars is shown by exponentiating the log transform.



1. Random Forest Regressor:

Sklearn’s Random Forest Regressor is used to fit the tree on the training data. This is then used to calculate predictions for rows in the validation and test data.

Validation R^2: 0.6595974698501812

Validation MAE: 0.349897204246723

Validation MSE: 0.2921336467353416

Validation RMSE: 0.5404938914875371

Test R^2: 0.8712581992377905

Test MAE: 0.26619436061311325

Test MSE: 0.12549443827536638

Test RMSE: 0.35425194180888603

The chart of the actual versus predicted value (log transformed) for the test data is as follows. Clearly the fit and the MSE on the test sample is much lower for the Random Forest model than the previous single Decision Tree.

A graph with blue lines and white text

Description automatically generated

Mean of actual salary for test data: 5.785362424995543

Mean of predicted salary for test data: 5.766031025009343

(Note: the above are log transformed values of Salary)

Hyperparameter Optimization:

We now use the max\_features attribute to optimize the hyperparameter related to the number of parameters selected to build a tree at a time. The default is “sqrt” which considers the square root of the total number of parameters available. The other option is “none” which considers all parameters.

The R^2 comparisons are as follows:

Random Forest with max\_features=sqrt

train/validation/test:

0.9574675571663216 0.6757113279373217 0.8692502532325795

Random Forest with max\_features=None

train/validation/test:

0.9611367188451325 0.6498791880619909 0.8701076657616824

A comparison of the two versions of the Random Forest regression model (for the two hyperparameter settings) with the actual salary for the test data is as follows:

A graph of a graph with green and blue lines

Description automatically generated with medium confidence

1. Boosting Model:

A tree based Boosting regression model was built on the training data using AdaBoost. The model was scored on the validation and test data and the summary metrics are as follows.

Training R^2 Score: 0.8958072329267436

Training MSE: 0.06692818962220982

Validation R^2 Score: 0.6535811392930463

Validation MSE: 0.29729686507231323

Test R^2 Score: 0.8429333193625086

Test MSE: 0.1531048559339731

Mean of Actual Salary (Test Data): 5.785362424995543

Mean of Predicted Salary (Test Data): 5.820451392531291

(Note: the above are log transformed values of Salary)

The comparison of AdaBoost model predictions with the Random Forest model and Salary actuals (log transformed values) on the Test data is as follows:

A graph showing a graph of salary

Description automatically generated with medium confidence

1. Linear Regression:

As with the previous models, the model was fitted on the training data and then scored on the validation and testing data. The R^2 on the test data with linear regression is distinctly lower than the previous Single Decision Tree, Random Forest and Boosting regression models.

Random Forest Regression:

Validation Score: 0.660

Test Score: 0.871

AdaBoost Regression:

Validation Score: 0.645

Test Score: 0.845

Linear Regression:

Training Score: 0.464

Validation Score: 0.442

Test Score: 0.607

E and F. Ridge Regression and Lasso:

The linear model was modified using two shrinkage methods - Ridge Regression and Lasso. The R^2 remains low in comparison to the earlier Random Forest and Boosting regression models.

Lasso Regression:

Training Score: 0.39542608940321855

Validation Score: 0.4463604614947928

Test Score: 0.5539347145646234

Ridge Regression:

Training Score: 0.4644679866586633

Validation Score: 0.4417290978326277

Test Score: 0.606398624139123

Hyperparameter Optimization: The hyperparameter in this case is the tuning parameter used to calculate the shrinkage penalty in the minimization equations for Ridge Regression and Lasso.

Ridge Regression:

A math equations on a piece of paper

Description automatically generated

Lasso:

A math equation with a plus and a positive symbol

Description automatically generated with medium confidence

(Source: An Introduction to Statistical Learning by James, Witten, Hastie and Tibshirani)

Various values of the tuning parameter were tried but with no appreciable sloping in model accuracy for both Ridge Regression and Lasso.

Tuning Parameter Values: [0.00001, 0.001, 0.01, 0.1, 1, 10, 100, 1000]

Ridge Regression:

A number of a number

Description automatically generated with low confidence

Lasso:

A screenshot of a computer

Description automatically generated

A graph with a line and a blue line

Description automatically generated

**Analysis:**

The Random Forest regression model for the tuned hyperparameter had the best performance among all the predictive models that were attempted. The R^2 values are as follows:

Decision Tree Regression:

Validation Score: 0.5587164537125167

Test Score: 0.7070098247833088

Random Forest Regression:

Validation Score: 0.6595974698501812

Test Score: 0.8712581992377905

AdaBoost Regression:

Validation Score: 0.6450184594199913

Test Score: 0.8445658526697923

Linear Regression:

Training Score: 0.464

Validation Score: 0.442

Test Score: 0.607

Lasso Regression:

Training Score: 0.39542608940321855

Validation Score: 0.4463604614947928

Test Score: 0.5539347145646234

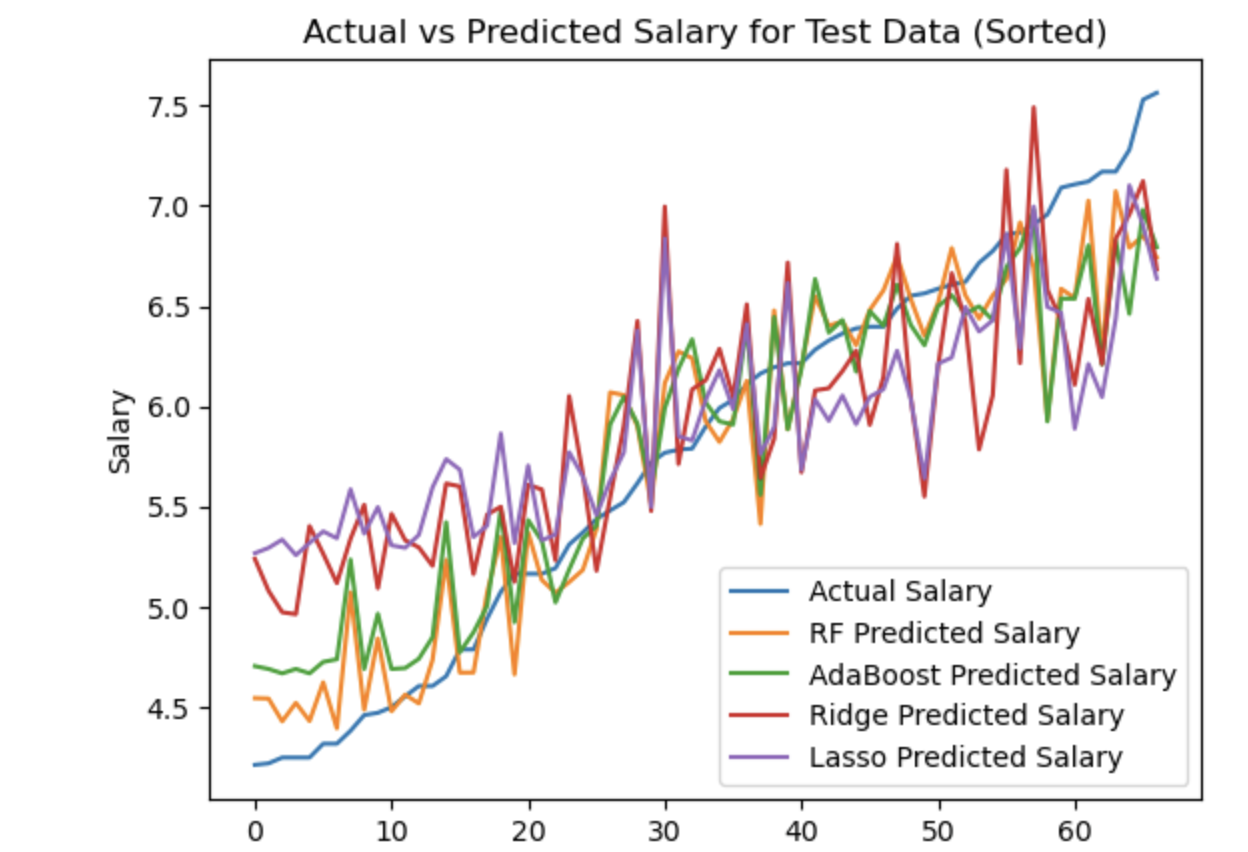
Ridge Regression:

Training Score: 0.4644679866586633

Validation Score: 0.4417290978326277

Test Score: 0.606398624139123

The decision tree regression models with tuned hyperparameters are doing better as a group compared to the variants of linear regression models that were attempted. This is also evident in the below line chart comparing predicted Salary (log transform values) to actuals values for the various regression models.



(RF: Random Forest)

**Reflections for Further Learning:**

Given more sample and time, I would have attempted to further fine tune the hyperparameters using k-fold cross-validation. I would also want to tune some of the other hyperparameters. I am surprised that the tuning parameter in Ridge Regression and Lasso did not produce more appreciable sloping of outcomes and suspect that this was because I did not cover a fuller range of possibilities. Finally, I noticed that even the best of the above regression models – Random Forest – underpredicts at the upper end of the Salary range and overpredicts at the lower end of the Salary range. I wonder if this is because an important predictor variable is missing from the available list. It could also be because all the hyperparameters are not sufficiently tuned.