Exploring the representativeness of BA on MLB players' performances

Bo Lin (1329002), Ray Shi(1328997)
DATA 201B, TUFTS UNIVERSITY

1. Introduction

Entertainment Sports Programming Network (ESPN) releases the performance records of all Major League Baseball (MLB) players in each year. This information records the number of runs and hits the player has performed and games the player has been part of, et cetera. A batting average (BA) - the average number of hit at bats - is also calculated for each player. While many use the BA score to judge the performance of MLB players, it may not depict a comprehensive picture of players' capability and performances due to its simplistic calculation. This paper aims to explore if BA is significantly correlated with variables other than hits (H) and at-bat (AB) and how well does BA reveal the information about each MLB player's performance.

2. Data:

2.1 Definition:

Variable Name	Definition
YRS	Number of years played in the MLB; integer
G	Number of games played in one season; integer between [0,162]
AB	Number of times at-bat; integer
R	Number of runs scored; integer
H	Number of hits; integer
2B	Number of 2 base hits; integer
3B	Number of 3 base hits; integer
HR	Number of Homeruns; integer
RBI	Number of runs batted in; integer
BB	Number of four bad ball escorts; integer
SO	Number of strikeouts; integer
SB	Number of stolen bases; integer
CS	Number of times caught stealing the base; integer
BA	Batting Average, equals hits/at-bats; [0,1]

2.2 Summary Statistics:

Variable	Min	Mean	Max	St.Dev
YRS	0	4.971098	18	3.66563
G	56	118.5347	162	27.64255
AB	201	403.5751	681	126.3983
R	15	59.30347	135	25.42275
H	34	105.6792	206	40.09118
2B	4	21.57225	58	9.415789
3B	0	1.988439	10	2.047235
HR	0	17.63584	53	10.51791
RBI	12	57.24277	126	25.44967
BB	8	39.47977	119	21.48355
SO	24	97.73988	189	34.95076
SB	0	5.82948	46	7.483497
CS	0	2.092486	10	2.217872
BA	0.157	0.257549	0.344	0.032487

The above data is obtained from the ESPN website by using python web scraping functions and stored in a csv file. The detail information of the web scraping method being used could be found in the attaching python script file.

From the above summary statistics table, we can see that the average batting average (BA) for an MLB player is 0.257549, with a standard deviation of 0.032487. Also, on average, an MLB player produces17.6 home runs and 57.2 runs batted in per season. Since these three variables are commonly used by the baseball fans to determine if such player is a good hitter or not, we can conclude that an above-average hitter in MLB should at least has a BA of 0.257 with 18 home runs and 58 RBIs per season.

The data set is split into training and testing data groups following a 7 to 3 ratio.

3. Model

The model we build in order to test and analyze the relationship between BA and other variables is as follows:

$$BA \sim \beta_0 + \beta_1(YRS) + \beta_2(G) + \beta_3(R) + \beta_4(2B) + \beta_5(3B) + \beta_6(HR) + \beta_7(RBI) + \beta_8(BB) + \beta_9(SO) + \beta_{10}(SB) + u$$

The model is fitted with both the linear regression and the Ridge regression. The goodness-of-fit of the models is calculated by obtaining the in-sample and out-sample regression scores.

The purpose of going through a linear regression is to explore possible correlations between BA and other variables and to assess the effect of any possible collinearity, ridge regression analysis is performed subsequently.

4. Analysis

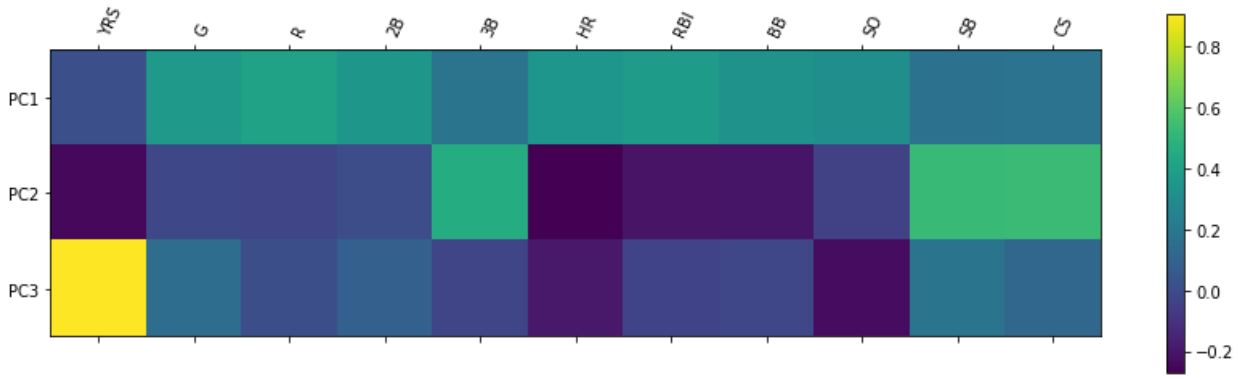
The following table shows the results of the prediction analysis using both linear regression and ridge regression.

Model	In sample score	Out of sample score
Linear Regression	0.5698	0.4622
Ridge Regression	0.5698	0.4623

The results suggest that either model captures at most 56.98% of the variation in BA with other variables. When fed with out of sample data, either model predicts with only 46.22% accuracy.

In attempts to explore the relationship between variables and BA, a high score would suggest a strong correlation between them and a low score suggests a weak or minimum correlation between. The lower than 0.5 score for both models in predicting out of sample BA suggests that BA is barely correlated with all of the variables.

To further assess the relations between variables and BA, a PCA analysis is done between BA and all other variables except AB and H. The contribution heatmap is shown below. With each principal component explains 49.08%, 18.34%, and 9.16% of the variance in BA respectively.



The PCA contribution analysis suggests how much each variable is correlated to BA and is commonly used in determining what variables keep to reduce the dimension of the data. As this PCA heatmap shows few variables have high contribution (contribution ratio greater than 0.6) and even the top principal component explains less than half of the variance in BA, it strongly suggests that there exists minimum correlation between BA and other variables excluding H and AB.

5. Conclusion

From linear regression and ridge regression, BA does not depict a full picture of the capability and performance of a MLB player. While BA may suggests a measurement of performance at a specific part of the game - number of hits per at-bat - there are other parts of the game where a player's ability and skills matter and make a difference in the game. Judging a player by the mere measurement of BA would thus likely to be bias and inaccurate as it represents less than 60% of the entire performance of the player in each game.

6. Appendix - Python code

The following code was executed on Python to perform the analysis:

```
#Bo Lin (1329002), Ray Shi(1328997)
#Linear and Ridge Regression, DATA 201B
#TUFTS UNIV, 2019-12-08
#----------------------------------------------------------------------

#import libraries
import pandas as pd
from sklearn.model_selection import train_test_split

#Read in data and drop unuseful col
data = pd.read_csv("PlayerData.csv")
data.drop(['Unnamed: 0'],axis=1,inplace=True)
data.drop(['PLAYER'],axis=1,inplace=True)
data.drop(['H'],axis=1,inplace=True)
data.drop(['AB'],axis=1,inplace=True)

#Split BA from others
Xs = data.drop(['BA'],axis=1)
y = data['BA'].values.reshape(-1,1)

#import library for ridge
from sklearn.model_selection import GridSearchCV
from sklearn.linear_model import Ridge

#Do ridge
ridge = Ridge()
parameters = {'alpha': [1e-15,1e-10,1e-8,1e-4,1e-3,1e-2,1,5,10,20]}
ridge_regressor = GridSearchCV(ridge,parameters,scoring = 'neg_mean_squared_error',cv=5)
ridge_regressor.fit(Xs,y)
#print out best params and the MSE
print(ridge_regressor.best_params_)
print(ridge_regressor.best_score_)

#Split data into test and training sets
X_train,X_test,y_train,y_test = train_test_split(Xs,y,test_size=0.3,
                                    random_state = 0)
#Based on best params=20, do a ridge with alpha=20; get scores
rr20 = Ridge(alpha=20)
rr20.fit(X_train,y_train)
Ridge_train_score = rr20.score(X_train,y_train)
Ridge_test_score = rr20.score(X_test,y_test)
```

```
print ("Ridge regression train score with alpha=20:", Ridge_train_score)
print ("Ridge regression test score with alpha=20:", Ridge_test_score)

#Do linear regression
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(X_train,y_train)
lr_test_score = lr.score(X_test,y_test)
lr_train_score = lr.score(X_train,y_train)
print ("Linear regression train score:", lr_train_score)
print ("Linear regression test score:", lr_test_score)

#Examine the PCA contribution by each variable
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler

pcaX = data.iloc[:,0:11].copy()
pcaY = data.iloc[:,11].copy()
pcaX_standard = StandardScaler().fit_transform(pcaX)
pca = PCA(n_components = 3)
PCs = pca.fit_transform(pcaX_standard)
projected_data = pd.concat([pd.DataFrame(data=PCs,columns =
["PC1","PC2","PC3"]),pcaY],axis= 1)

#Variance explained by top PCs
ExplainedRatio = pca.explained_variance_ratio_
for i in range(3):
    print("PC" + str(i+1) + " explains " + str(ExplainedRatio[i] *100)[0:6] + "% of variance in BA")


import matplotlib.pyplot as plt
plt.matshow(pca.components_,cmap='viridis')
plt.yticks([0,1,2],['PC1','PC2','PC3'],fontsize=10)
plt.colorbar()
plt.xticks(range(11),['YRS','G','R','2B','3B','HR','RBI','BB','SO','SB','CS'],rotation=65,ha='left')
plt.show()
```