Baseball Value

Matthew Helbig

2019-09-28

Introduction

The goal of this project is to identify the attributes of the players with the most value and the least value, as well as predicting which players will have the most and least value in the future.

Getting started

The first step we'll take is to identify which datasets we'll be working with. The Lahman Database has a lot of good information that we'll need, namely Player Value data, salary, and dates of birth.

We'll download that from http://www.seanlahman.com/baseball-archive/statistics/

It's possible to download the database as a .CSV file, and you could technically do everything needed for this project without using SQL, but it's a good practice to get used to working with databases, especially since a lot of the datasets you'd see in the real world will be bigger than the one we're currently working with.

Sean Lahman has posted the entire .sql file on his website, so importing it into MySQL is as simple as hitting 'Data Import' and loading that database into its own schema.

Once the Lahman database is all settled in its own schema, we want to start to work with that data. It's definitely possible to run all your SQL commands inside the MySQL environment itself, but RStudio has packages available for installation that make it easy to work with your database right from R. This has a few advantages, most notably the fact that you don't have to keep hopping back and forth between programs.

Talking to our Database

The first thing you'll do is make sure that the RMySQL package is installed. This is as simple as calling **install.packages("RMySQL")** once.

Next we'll load up our RMySQL library and get to work.

```
library(RMySQL)
```

```
## Loading required package: DBI
```

You'll need to define your new database object, called "mydb" here, using the following template:

mydb = dbConnect(MySQL(), user='user', password='password', dbname='dbname', host='host').

Then we'll run a quick command to look at the different tables inside the dataset we've loaded up.

```
dbListTables(mydb)
```

```
##  [1] "AllstarFull"          "Appearances"          "AwardsManagers"
##  [4] "AwardsPlayers"        "AwardsShareManagers" "AwardsSharePlayers"
##  [7] "Batting"              "BattingPost"          "Batting_Salary"
## [10] "Batting_post_2006"    "CollegePlaying"       "Fielding"
## [13] "FieldingOF"           "FieldingOFsplit"      "FieldingPost"
## [16] "HallOfFame"           "HomeGames"            "Managers"
## [19] "ManagersHalf"         "Master"               "Parks"
## [22] "Pitching"             "PitchingPost"         "Salaries"
## [25] "Salaries_post_2006"   "Schools"              "SeriesPost"
## [28] "Teams"                "TeamsFranchises"      "TeamsHalf"
## [31] "test1"
```

So the three that we're most interested in right now are "Batting," "Pitching," and "Salaries." The database has a ton of information that's interesting, but ultimately not super useful for our experiment. We're going to be mostly concerned with salary information from 2006 to 2016, since a lot of salary data before 2006 is incomplete and we're going to be running some predictive models on 2017 and 2018 data later.

We'll need to create a new table in SQL based on our criteria of only needing 2006 and later (this database ends in 2016).

```
CREATE TABLE Batting_post_2006
LIKE Batting
```

Now we've created a table with the same containers as our Batting table, and now we need to load the data into those containers.

```
INSERT INTO Batting_post_2006
SELECT *
FROM Batting;
```

Now we're going to trim our data from our new table in order to limit ourselves to only 2006 to 2016. This is where SQL is beneficial, since our original dataset of "Batting" is complete safe and untouched.

```
DELETE FROM Batting_post_2006
WHERE yearID < 2006
```

We're going to do the same thing with our Salaries table, and eventually our Pitching table. We won't do the Pitching table just yet, but we can follow these same steps in order to set that table up for export to R. For now, we'll work on the Salaries table.

Note to self, this is where I left off

```
batting_post_2006 <- read.csv("Batting_post_2006.csv", header = T)
salaries_post_2006 <- read.csv("Salaries_post_2006.csv", header = T)
total <- merge(batting_post_2006,salaries_post_2006,by="playerID")
newdata <- subset(total, yearID.x == yearID.y)
qualified <- subset(newdata, AB >= 300)
```

Read the data in and merged it in R because SQL was having a lot of problems dealing with 0 values and INNER JOIN wouldn't cooperate, and we can't ignore them in most cases (3B = 0, for example). For tomorrow (or Monday), an explanation of why we did it this way would be good, in addition to explaining where we left off with SQL.

We should get the player value data from baseball-reference and convert it to CSV, then we should see if we can combine the baseball-reference data with our current data. That'd be a good task for Sunday or Monday