Big Data Pyspark Project

Vincent Bushong

2019-12-09

1 Phase 2: Linear Regression

To recalculate the coefficients using linear regression, I first rearranged the equation so it was three added terms, each with a coefficient. The equation became

$$RC = B * \frac{OnBase * TotalBases}{Opportunities} + C * \frac{OnBase * AdjustedWalks}{Opportunities} + D * \frac{OnBase * SacrificesSteals}{Opportunities}$$

where

- $OnBase = H + BB - CS + HBP - GIDP$,
- $TotalBases = 1B + 2 * 2B + 3 * 3B + 4 * HR$,
- $AdjustedWalks = BB - IBB + HBP$,
- $SacrificesSteals = SH + SF + SB$, and
- $Opportunities = AB + BB + HBP + SF + SH$.

Since we don't have a known "true" value for runs created (since that is the formula we are trying to construct) and because you can't use runs per player as the dependent variable (since the whole point of runs created is that a player's contribution is not just the runs they score), linear regression cannot be performed at the individual player level. However, if a player "creates" a run, it will appear as a run scored for their team, no matter which player individually scored it. Thus, the regression is performed on a team basis. The individual player stats for 2017 were grouped and aggregated by team, then fed into a PySpark `LinearRegression` object, with the intercept set to zero (since putting up zero of all the relevant stats means you didn't play at all, and so created zero runs). The resulting `LinearRegressionModel` was analyzed for its coefficients, and applied to the 2018 year as a test dataset.

The linear regression generated the coefficients $B \approx 1.00$, $C \approx 0.21$, and $D \approx 0.39$. The first constant is essentially the same while the second two are somewhat lower than the original coefficients. This suggests that including the park factor was causing runs to be overestimated when applied to the season as a whole, requiring the coefficients to be lowered. Indeed, when I calculated the total park factor-adjusted runs created for 2018 using the original coefficients by running the `phase1.py` script, it estimated 22376 runs created, compared to the season's actual 21630 runs (for an error of 4.3%). Using the new coefficients, the formula estimated 21396 runs created, slightly over-correcting the original overestimate (but clocking an error of only 1.1%).

The R^2 value of the model when applied to 2018 was 0.998607, suggesting the variance in the data is well-explained by the model. The $RMSE$ was 26.997677 for 2018, while for 2017 (the training data), it was a little lower at 21.440462. This suggests a certain amount of overfitting, although the model still seems to have good predictive value based on the result obtained from 2018.

2 Phase 3: Finding the Missing Data

For phase 3, I found the missing pitching data on the "Batting Against" page at Baseball Reference (`https://www.baseball-reference.com/leagues/MLB/2019-batting-pitching.shtml`). While there is no API for easy programmatic access to it, they make Javascript widgets available to embed on other websites. I made a simple HTML/jQuery scraper page that pulls in a widget for the user-chosen year, tweaks the resulting table to get it in a consistent format with Lahman's data (including renaming columns and changing some team ID's), and makes the data available for export as CSV using the DataTables jQuery plugin (`https://www.datatables.net/`).

I then used the scraper to pull CSVs for 2015-2018 and uploaded them to a GitHub repository. In the combiner program, I fetch those files via HTTP, append them as a string, then write it out as a single CSV file, which the report program uses as input.

Note that since for pitchers, lower runs created is better; therefore, I sort the results in ascending order. This by itself leads to effect that pitchers who e.g. pitched a single inning and didn't allow a single hit appear first in the order, so it isn't really a ranking of leading pitchers. Enforcing a minimum atbat value and sorting by RC27 can help alleviate this, but neither RC nor RC27 alone can give a meaningful ranking; they must be considered along with other stats.