Time Series Modeling of Baseball Performance

Kyle Andelin  
*Western Washington University*

Sam Kaplan  
*Western Washington University*

Follow this and additional works at: [https://cedar.wwu.edu/scholwk](https://cedar.wwu.edu/scholwk)

Part of the *Computer Sciences Commons*

[https://cedar.wwu.edu/scholwk/2015/Day_one/20](https://cedar.wwu.edu/scholwk/2015/Day_one/20)

This Event is brought to you for free and open access by the Conferences and Events at Western CEDAR. It has been accepted for inclusion in Scholars Week by an authorized administrator of Western CEDAR. For more information, please contact [westerncedar@wwu.edu](mailto:westerncedar@wwu.edu).
Overview

Motivation: Predicting upcoming player performance is vital to team management and a hot topic in sports media.

Goal: Greater understanding of recent trends impacting future outcomes and increased accuracy of predictions.

Approaches:
I. Use expectation maximization (EM) to identify most predictive past time periods
II. Predict next game performance based on season history using a recurrent neural network (RNN)

Background

Expectation Maximization
- The EM algorithm is a general iterative method to perform maximum likelihood estimation (MLE)
- Find MLE of mixture density parameters via EM

Our Model

Mixture Model
- Future performance as a function of past performance periods:
  \[ P_i(x) = \frac{w_1 P_{i,1}(x) + w_2 P_{i,2}(x) + w_3 P_{i,3}(x)}{\text{Past} \quad \text{Early} \quad \text{Recent}} \]
  where
  \[ P_{i,j}(x) = \begin{cases} 1 - \alpha & \text{if } x \neq \alpha \delta_j(x) \\ \alpha \delta_j(x) & \text{otherwise} \end{cases} \]
  \[ \delta_j \] - league average PMF for period \( j \)
  \[ \alpha \] - interpolation coefficient of league average PMF, \( 0 \leq \alpha \leq 1 \)
  \[ w_j \] - mixing weight for time period \( j \)

Experiments

Data
- Play-by-play data from Retrosheet.org
- 250+ players per season, years 2000-2013
- 6 statistics: strikeouts (K), walks (BB), singles (1B), doubles (2B), triples (3B), home runs (HR)

Training
- Tune \( \alpha \) to maximize log-likelihood on a held out data set
- Use EM to learn appropriate \( w_j \) weights to best predict future outcomes

Results

Optimal \( \alpha \) Model vs League Average Model

<table>
<thead>
<tr>
<th>Initial</th>
<th>BB</th>
<th>1B</th>
<th>2B</th>
<th>3B</th>
<th>K</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.2</td>
<td>0.25</td>
<td>0.35</td>
<td>0.6</td>
<td>0.45</td>
<td>0.1</td>
</tr>
<tr>
<td>0.1</td>
<td>0.25</td>
<td>0.35</td>
<td>0.6</td>
<td>0.45</td>
<td>0.1</td>
</tr>
</tbody>
</table>

The optimal \( \alpha \) is highly dependent on the statistic being considered

RNN

Data
- Stats per player, per game, over a season:
  \[ <\text{111120 ... 001100}> <\text{100010 ... 100001} \]
  \( x = \text{did not play} \quad \text{<}/\text{s}> = \text{end of season} \)

Model
- Recurrent Neural Network

\[ s(t) = f(Uw(t) + Ws(t-1)) \]
\[ y(t) = g(Vs(t)) \]
\[ f(z) = \frac{1}{1 + e^{-z}} \quad g(z_m) = \frac{e^{z_m}}{1 + e^{z_m}} \]

- Hidden layer, \( s(t-1) \), represents history
- Time on:
  - Size of hidden layer
  - Number of time steps to backpropagate error

Training
- Train on 60% of data - Tune on 20% - Test on remaining 20%
- Learn optimal \( U, W, V \) matrices
- Trained using backpropagation through time

Results
- To evaluate our methods, we feed held out data to our model in the above form and compute the metrics in the following table

Early Measures of Model Performance

<table>
<thead>
<tr>
<th>K</th>
<th>BB</th>
<th>1B</th>
<th>2B</th>
<th>3B</th>
<th>HR</th>
<th>Hits</th>
</tr>
</thead>
<tbody>
<tr>
<td>RNN MSE</td>
<td>0.64</td>
<td>0.29</td>
<td>0.53</td>
<td>0.16</td>
<td>0.02</td>
<td>0.09</td>
</tr>
<tr>
<td>Lg Avg</td>
<td>0.66</td>
<td>0.29</td>
<td>0.54</td>
<td>0.16</td>
<td>0.02</td>
<td>0.09</td>
</tr>
<tr>
<td>Run Avg</td>
<td>0.67</td>
<td>0.30</td>
<td>0.64</td>
<td>0.19</td>
<td>0.03</td>
<td>0.12</td>
</tr>
<tr>
<td>RNN MAE</td>
<td>0.68</td>
<td>0.44</td>
<td>0.62</td>
<td>0.28</td>
<td>0.04</td>
<td>0.16</td>
</tr>
<tr>
<td>Lg Avg</td>
<td>0.68</td>
<td>0.44</td>
<td>0.64</td>
<td>0.28</td>
<td>0.03</td>
<td>0.17</td>
</tr>
<tr>
<td>Run Avg</td>
<td>0.66</td>
<td>0.44</td>
<td>0.64</td>
<td>0.29</td>
<td>0.05</td>
<td>0.18</td>
</tr>
<tr>
<td>RNN % Corr</td>
<td>0.48</td>
<td>0.75</td>
<td>0.57</td>
<td>0.85</td>
<td>0.98</td>
<td>0.92</td>
</tr>
</tbody>
</table>

- Ongoing work: we anticipate more results soon

Ongoing work: we anticipate more results soon