DataBall

Predicting NBA Winners with Data

Kevin Lane

July 15, 2017

Outline

Background

- Sports analytics began in professional baseball, most notably with the work of Bill James[1]
  - James coined the term sabermetrics as "the search for objective knowledge about baseball"
  - James selected the name to honor the Society for American Baseball Research (SABR)
- Gained widespread adoption after Billy Beane implemented James' ideas and led the Oakland Athletics to a record winning streak[2]
- Analytics has since spread to other sports and its impact is evidenced by several examples:
  - MLB's increased attention to on-base percentage beginning in the Moneyball era of the early 2000s
  - The rise of the three-point shot and subsequent fall of the midrange jumper in the NBA
  - Increased use of short, high percentage passes in the NFL

[1] James 1985.

[2] Lewis 2003.

Background

What makes sports an attractive testbed for machine learning?

According to Nate Silver, "sports nerds have it easy."[3]

1. "Sports has awesome data."
2. "In sports, we know the rules."
3. "Sports offers fast feedback and clear marks of success."

Why the NBA?

- Easily the most deterministic of the major American sports
- The NBA provides a wealth of advanced stats and player tracking data on their website
- The season is long enough at 82 games that sample size is not as much of a concern as in the NFL, who claim to have parity, but also only play 16 regular season games

[3] Silver 2015.

Process

- I used several algorithms from the popular Python machine learning library scikit-learn[4] to predict NBA game winners
  - Logistic Regression
  - Support Vector Machine
  - Random Forest
  - Multilayer Perceptron
  - Naïve Bayes
- I used box score data from the 1990-91 season through 2015-16
- The games are split into training and test sets randomly, so games from future seasons are used to predict past games
  - This does not provide a realistic scenario for making predictions in real time, such as in betting
  - Provides a easy way to compare several algorithms and feature combinations, which will help inform future work
- All models are trained with season-averaged team stats

[4] Pedregosa et al. 2011.

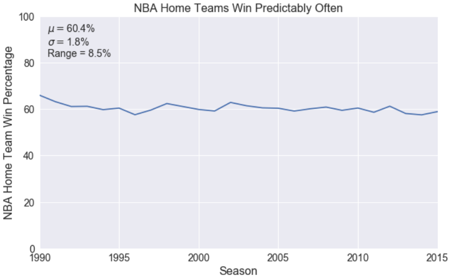
Data Wrangling

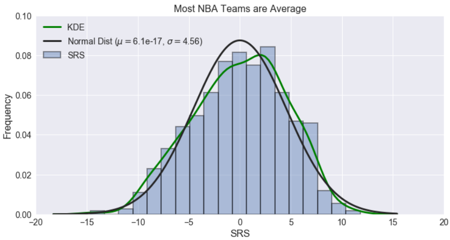
- I collected stats from the NBA's stats website `stats.nba.com`
  - The site exposes a wealth of information in JSON format through various web API endpoints
  - I utilized the GitHub project nba_py to format the URLs and collect stats into Pandas DataFrames
- I stored all collected stats to a SQLite database using Python's built-in SQLite support
- I used the basic box score stats to calculate more advanced stats
  - Offensive/defensive ratings (points scored/allowed per 100 possessions), which requires an estimate for the number of possessions
  - Simple Rating System (SRS), which is a team's average margin of victory adjusted for its strength of schedule
  - Oliver's four factors[5], which include effective FG%, TOV%, OREB%, and free throw rate
  - Weighted four factors, which is just sum of the four factors weighted according to Oliver's assigned weights

[5] Oliver 2004.

Data Exploration

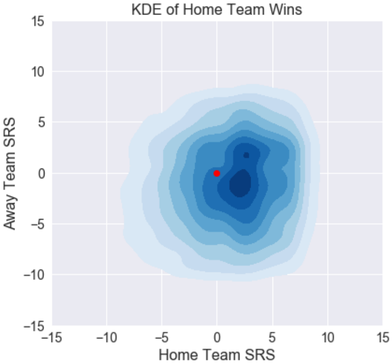
- The top figure shows that the home team winning percentage is remarkably consistent
  - Simply predicting the home team wins will yield about 60% accuracy
  - This provides a good baseline; any predictive model worth implementing should beat this
- The bottom figure shows that team performance (according to SRS) very closely resembles a normal distribution
  - An SRS of zero indicates an average team

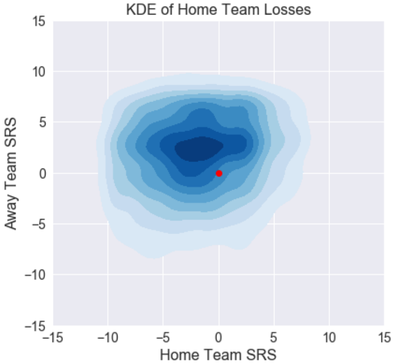




Data Exploration

- The plots below show kernel density estimations (KDE) of SRS split between home team wins and losses
- The dark region to the bottom right of the origin for home team wins shows above-average home teams tend to beat below-average visitors
- The opposite appears in the KDE of home team losses





Feature Selection

- The plots below show cross-validation ROC and precision/recall curves using home and away SRS
- The folds show little spread, so we are confident the cross-validation results in a good estimate of model performance

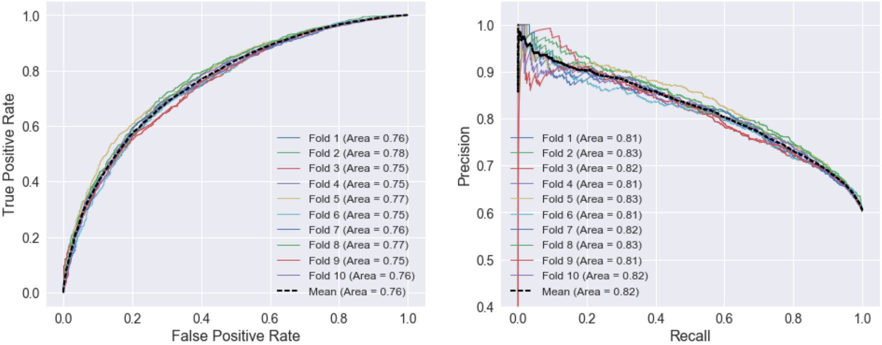


Figure 1: Cross-Validated ROC and Precision/Recall Curves (SRS)

Feature Selection

- The plots below show cross-validation ROC and precision/recall curves for various metrics
- All point-related metrics (SRS, Plus/Minus, etc.) are nearly identical
- The point-related metrics outperform the four factors metrics

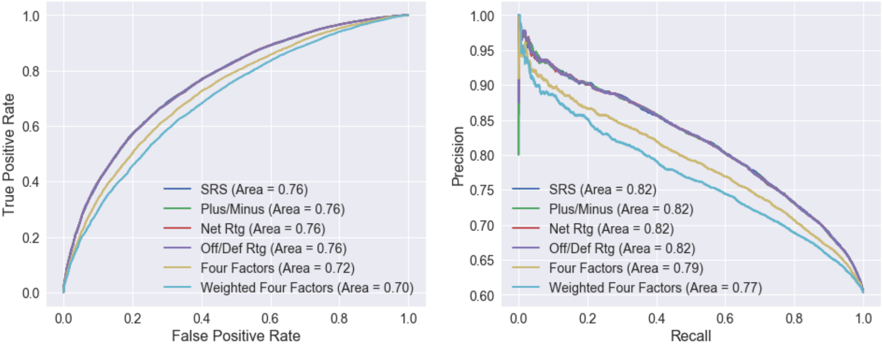


Figure 2: ROC and Precision/Recall Curve Feature Comparison

Parameter Tuning

- Parameter tuning did not yield models that performed noticeably better than the default models

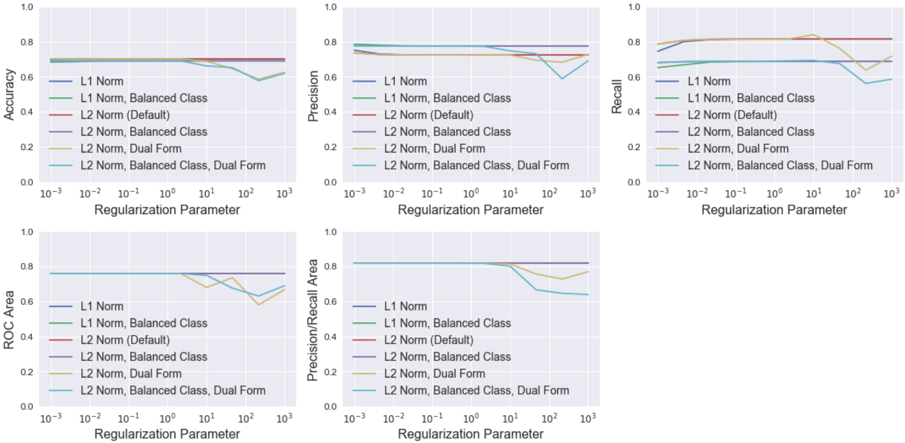


Figure 3: Logistic Regression Parameter Tuning

Model Performance

- Logistic regression does a great job at predicting home wins (81.5% accuracy), but struggles with home losses (about 50% accuracy)
- These general numbers hold true for all the models tested except the random forest model, which performed about 10% worse predicting home wins
- Additional effort should be focused on improving home loss predictions to improve overall model performance
    - One option is to down sample home wins to achieve a 50/50 split of the two classes

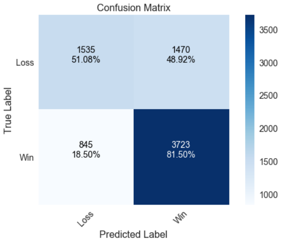


Figure 4: Logistic Regression CM

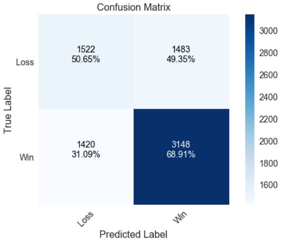


Figure 5: Random Forest CM

Model Performance

- All models performed about the same with the exception of the random forest model
- The random forest performed about the same with home losses, but had degraded performance predicting home wins

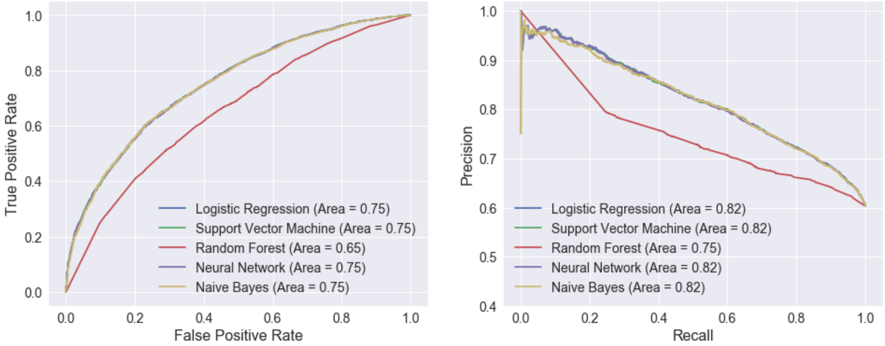


Figure 6: Model Performance Comparison

Comparison to Published Results

- The roughly 70% prediction accuracy is in line with published results
- Zimmermann[6] used algorithms in Weka[7] to predict NBA and NCAA game winners
  - He trained the models using data from previous games, making it a more realistic scenario
  - He correctly predicted about 57-68% of NBA games correctly, though most seasons were below 64% accuracy
  - He was much more successful in the NCAA, where the range in skill level is much larger than the NBA
- Loeffelholz et al.[8] used the MATLAB neural network toolbox to predict NBA game winners
  - They performed a cross-validation similar to what was shown here
  - They only examined part of one season (only 30 games used for testing)
  - They predicted approximately 74% of games correctly, but it is unclear if this generalizes well

[6] Zimmermann 2016.

[7] Witten, Frank, and Hall 2011.

[8] Loeffelholz, Bednar, and Bauer 2009.

Future Work

- Train models with prior games to permit “real time” predictions and update the models as each season progresses
- Incorporate player stats to adjust predictions as rosters fluctuate
- Predict winners against Vegas spreads
  - Track return on investment (ROI) if a prospective bettor were to bet on the model’s predicted winners
  - Incorporate a confidence threshold to investigate how ROI changes when bets are only made on games in which the model’s confidence exceeds the threshold

References I

[1] James, Bill. *The Bill James Historical Baseball Abstract*. Villard, 1985.

[2] Lewis, Michael. *Moneyball: The Art of Winning an Unfair Game*. W. W. Norton & Company, 2003.

[3] Silver, Nate. *Rich Data, Poor Data*. Feb. 2015. URL: https://fivethirtyeight.com/features/rich-data-poor-data/.

[4] Pedregosa, F. et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[5] Oliver, Dean. *Basketball on Paper: Rules and Tools for Performance Analysis*. Potomac Books, 2004.

[6] Zimmermann, Albrecht. "Basketball predictions in the NCAAB and NBA: Similarities and differences". In: *Statistical Analysis and Data Mining: The ASA Data Science Journal* 9 (2016), pp. 350–364.

References II

[7] Witten, Ian H., Frank, Eibe, and Hall, Mark A. *Data Mining: Practical Machine Learning Tools and Techniques*. 3rd ed. Morgan Kaufmann, 2011.

[8] Loeffelholz, Bernard, Bednar, Earl, and Bauer, Kenneth W. "Predicting NBA Games Using Neural Networks". In: *Journal of Quantitative Analysis in Sports* 5.1 (2009), pp. 1–17.