Using Machine Learning to Predict NBA Game Outcomes

Tim Cauley

Problem Summary

- Predict probability home team wins a game given net statistics between 2 teams in prior games
- Most prior research uses a games data to predict that game's outcome, which is easier but illogical
- Prior research uses team statistics rather than net statistics which is also illogical
- Industry standard models have relatively low accuracy (low-mid 60s) due to basketball being inherently difficult to predict
- Top of the line models have accuracy in low 70s, and use either an ELO rating system, or complex player-based models

Data Collection

- Scraped from NBA API using Python
- Collected data from 2017 through 2023 (~7000 games) from 3 different tables
  - Box Score, Advanced Box Score, Usage
  - Data consists of 85 columns across 3 tables

SEASON_ID	TEAM_ID	TEAM_ABBREVIATION	TEAM_NAME	GAME_ID	GAME_DATE	MATCHUP	WL	MIN	PTS	...
22017	1610612739	CLE	Cleveland Cavaliers	21700001	2017-10-17	CLE vs. BOS	1	239	102	...
22017	1610612738	BOS	Boston Celtics	21700001	2017-10-17	BOS @ CLE	0	241	99	...
22017	1610612744	GSW	Golden State Warriors	21700002	2017-10-17	GSW vs. HOU	0	241	121	...
22017	1610612745	HOU	Houston Rockets	21700002	2017-10-17	HOU @ GSW	1	239	122	...
22017	1610612764	WAS	Washington Wizards	21700006	2017-10-18	WAS vs. PHI	1	240	120	...

Data cleaning

- Joining tables into 1
- Created custom features
  - Key DNP: tracked # of key players on each team that rested for the given game
  - Rest: # of days since previous game
  - Back to Back: indicated whether team played on prior day
  - Game Number: linear count of games played each season
  - Games played prior 7: Counted # of games the team played in previous 7 days
  - Win percentage: % of games won in prior games
- Make columns for opponent stats for each game and make it 1 row per game rather than 2
- Make majority of variables net values, rather than 1 variable for each team

Data preprocessing

- Change values for each game to being averages of prior games, rather than the value resulted from that game
- Used both sliding window and exponentially weighted moving average (EWMA)
- Calculated best window length and degree of mixing rate by optimizing average correlation with game result
- Best game length: 50
- Best degree of mixing: 0.07
  - (7% most recent game, 93% moving average (doesn’t overreact to most recent game))
- Normalized values and balanced (training) data

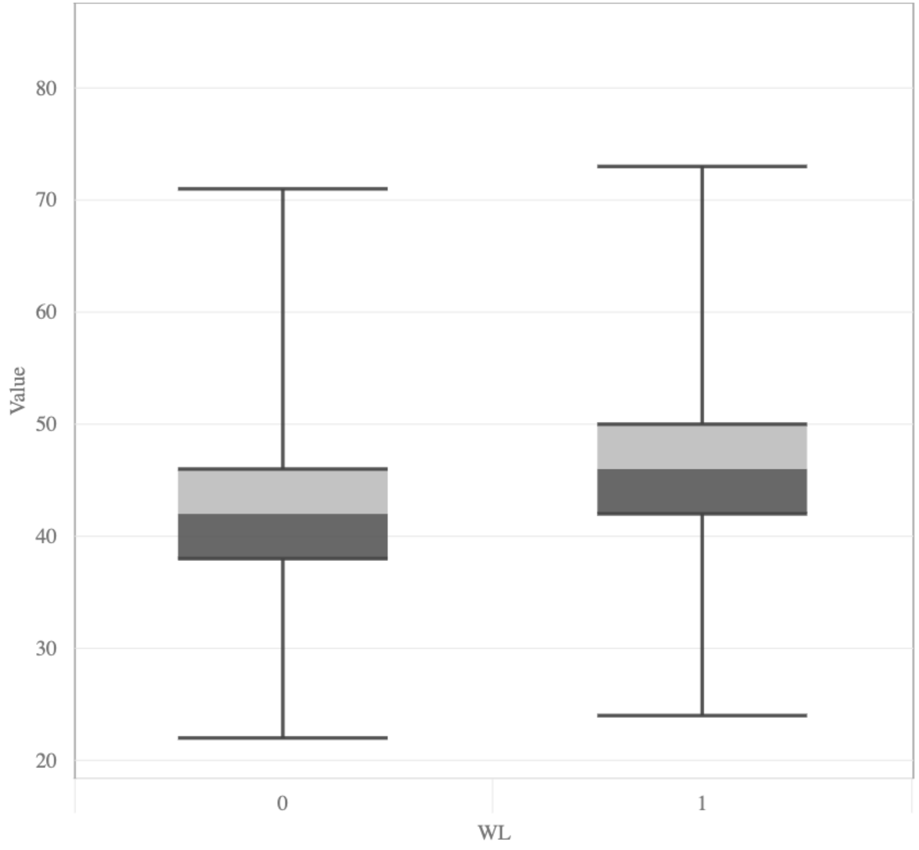
EDA

- Performed hypothesis testing on difference in means for variables between winning team and losing team
  - Almost every variable had statistically significant difference, with no surprises (ex: winning team averages 4 more rebounds than losing team)
- Did correlation analysis on variables vs outcome
  - When looking at game data, many variables were highly correlated with outcome, but when looking at sliding window / EWMA data no variable was highly correlated with outcome
- Summarized findings with a Power BI dashboard

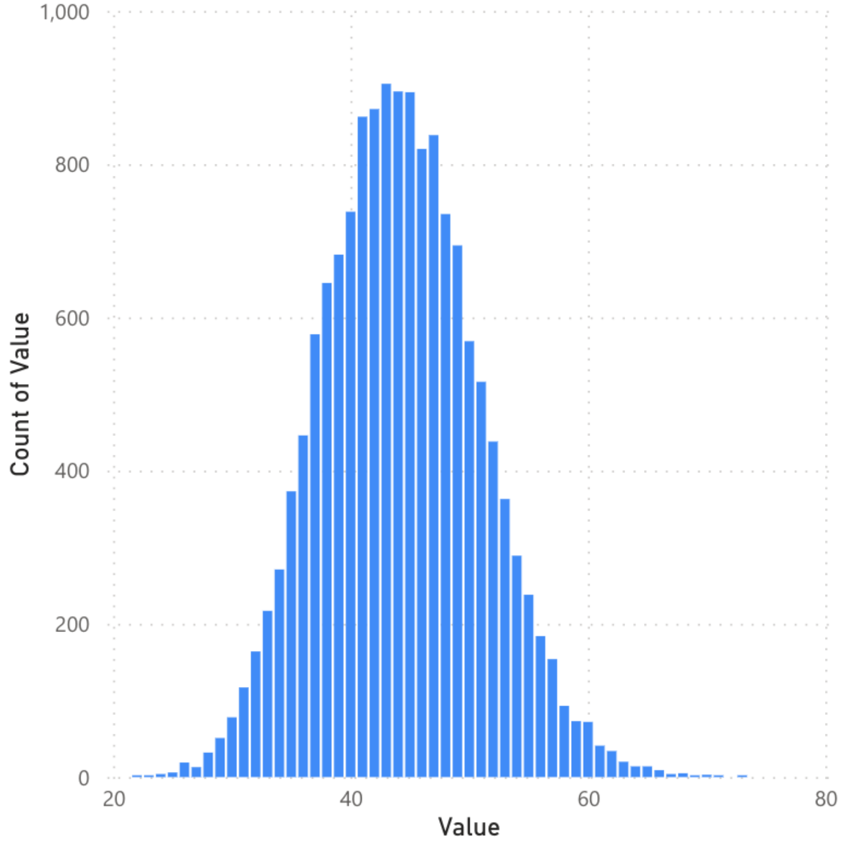
Attribute

REB

Box Plot of Values



Count of Value by Value



Maximum

73.00

Minimum

22.00

Average

44.21

Winning Average

46.18

Losing Average

42.24

Correlation with winning

0.30

Machine Learning

- 5 algorithms used: ELO Rating, Logistic Regression, Random Forest, Support Vector Machines, Feed Forward Neural Network
- 2 ELO models, and 8 models for each ML algorithm (1 using all features scaled and unscaled, 1 using feature selection scaled and unscaled for both sliding window and EWMA)
  - Used select K best for feature selection, finding optimal k (~50/70 features selected)
- 75/25 train test split using the most recent 25% as test set
- Calculated and compared accuracy, precision, recall, F1 score, and AUC for each model

ELO Rating System

$$p(win_{home}) = \frac{1}{1 + 10^{\frac{R_{away} - (R_{home} + 100)}{400}}}$$

$$MOV_{adjust} = \frac{(MOV + 3)^{0.8}}{7.5 + 0.006 * ((R_{home} + 100) - R_{away})}$$

$$R'_{home} = R_{home} + MOV_{adjust} * K(S_{home} - p(win_{home}))$$

Summarizing Average Model Accuracy

Algorithm Avg. Accuracies

ELO Model	Logistic Regression	FNN	Random Forest	SVM
67.07%	64.68%	64.23%	63.08%	62.67%

Avg. Accuracy by Sliding Window vs EWMA

Sliding Window	EWMA
63.64%	63.69%

Avg. Accuracy by scaled vs. unscaled

Unscaled	Scaled
63.83%	63.50%

Avg. Accuracy by features used

All Features	Feature Selection
63.59%	64.75%

Summarizing Best Model Accuracy

Algorithm Best Accuracy

ELO Model	Logistic Regression	FNN	Random Forest	SVM
67.17%	66.12%	65.64%	63.52%	63.46%

Best Accuracy by Sliding Window vs EWMA

Sliding Window	EWMA
66.12%	65.37%

Best Accuracy by scaled vs. unscaled

Unscaled	Scaled
66.12%	64.47%

Best Accuracy by features used

All Features	Feature Selection
65.01%	66.12%

Thoughts

- ELO models perform better because they are simply looking at who is the better team, while ML models are trying to generalize what makes a team better?
- Big success to be able to replicate model performance using data from prior games to predict game outcomes, rather than data from the game itself
  - Using net variables and custom variables (23/51 variables selected in feature selection were custom made)
- Low accuracy compared to other ML problems shows how difficult it is to predict sporting event outcomes
- Project continuation: player-based ML models