DATA608 Final Report

Justin Hink

Sunday, April 16, 2017

Abstract

The project explores predicting player level outcomes in professional basketball, specifically the National Basketball Association (NBA). Two distincly different models classes are explored in isolation for this task: Regression to the Mean based systems and Neural Network based systems. Finally, these two broad classes of prediction algorithms are combined into a composite model.

In total, four models are constructed, each with numerous tweakable parameters. Peformance is evaluated on a per model, per parameter basis and a recommendation is made as to which model should yield best predictive performance going forward.

Win Shares Per 48 Minutes is used as the target metric for the modeling effort thought the project. The technical framework built should allow similar work to be done against any other rate or counting statistic.

Key Words and Concepts

True Talent vs. Luck Each player in the NBA has an underlying, ever-changing talent level. This talent level is quite often different than statisically measured results (ex. FG%, 3 Point FG%, Defensive Rebound Rate, Win Shares Per Plying time) will present. A player's measured results and true talent are governed by the following high-level and simplified equation:

$$Outcomes_{Team} = Talent_{Team} + Luck$$

Regression To The Mean Also commonly referred to as "Regression Toward the Mean" or simply "Mean Reversion", is a general concept that extreme observataion tend to fall closer to the population mean on subsequent measurements. The concept is a pillar of in sports analytcs and most first generation projection systems incorporate the idea into their algorithms.

Projection System An algorithm, or set of algorithms, that attempt to predict future outcomes. These systems usually involve predicting player level statistics at various time scales (ie - season length projections, career length projections, game based projections). Examples:

1) Projection System A predicts Giannis Antetokounmpo to average 26.4 points next season
2) Projection System B predicts Mike Trout (baseball player) to finish his career with 116 Wins Above Replacement, making him a sure-fire Hall of Famer.

Marcel Originally developed for baseball but wholly relevant for other sports, Marcel is very simple but effective system developed and published by renowned baseball analyst, Tom Tango (an alias, his real name is not available in the public domain). The system gets it's name from a pet monkey named Marcel and the fact that the algorithm is so simple that a monkey should be able to develop and understand it. The mathematical form of the algorithm is as follows [cite blog post]:

$$\hat{p}_i = \frac{t^* x_i + \left(\frac{t^* n_0}{t^* n_1}\right) t^* D_{ni} p_0}{t^* n_i + t^* n_0}$$

While Tango published the algorithm, he does not want to take credit for the results it produces. Rather he prefers the algorithm to be used as a baseline for projection systems to use as a measuring stick.

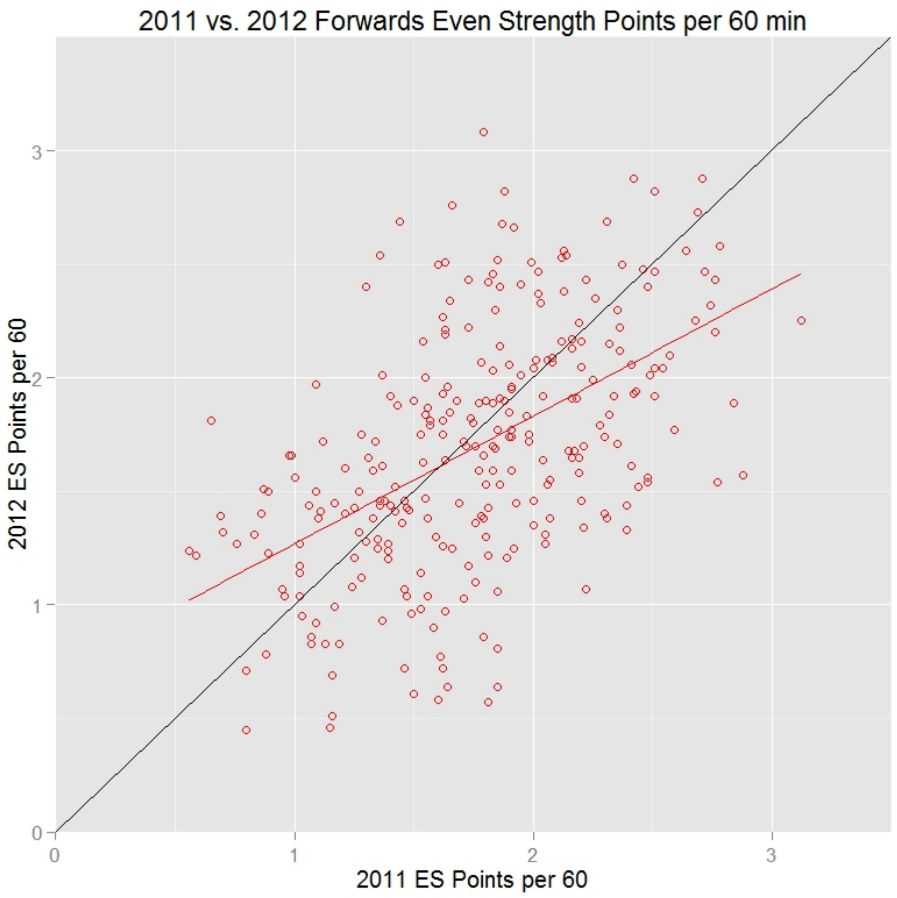
Win Shares A statistical concept originally developed by Bill James in the 1980s in his Baseball Abstracts, Win Shares attmpt to assign credit/blame to individual player for their team's performance.

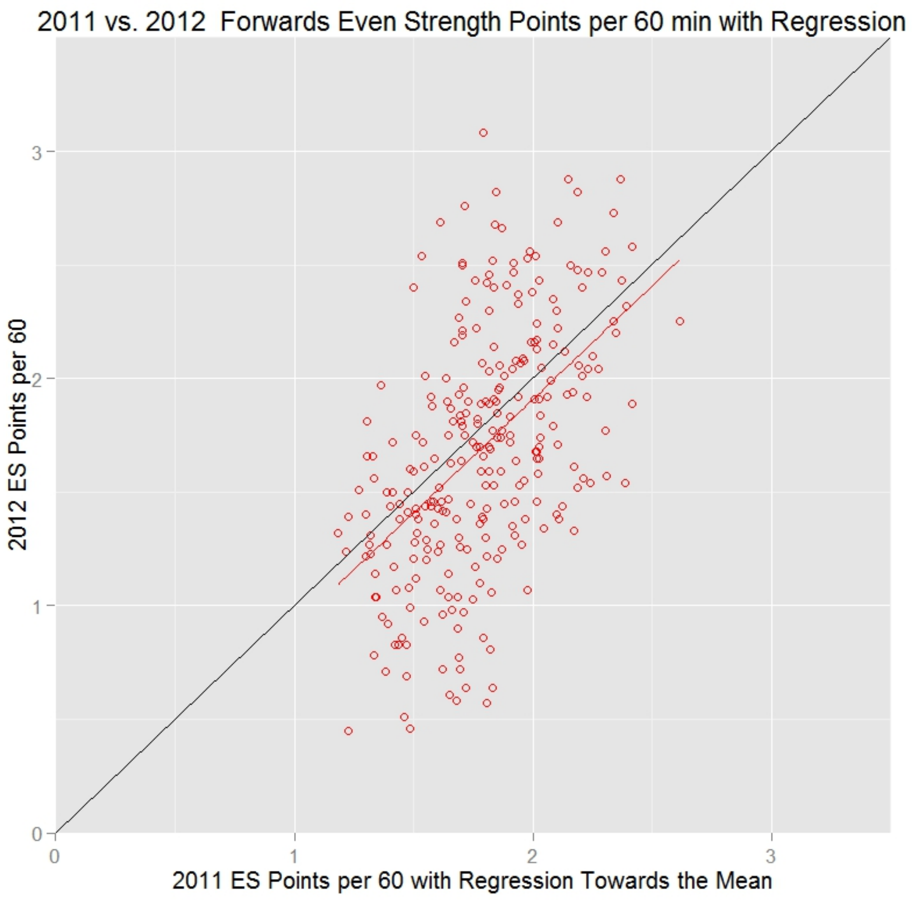
Neural Network A class of computation models that utilize a large collection of simple neural units working together in order to glean intelligence and insight for input data. Uses are broad but can include classification algorithms, regression algorithms and generalized prediction engines.

Deep Learning A class of machine learning algorithms that use many layers of nonlinear processing units for model feature extraction and transformation. Output from one layer of the network is fed into the next as input. High level model features are derivd from lower level features and create an overall hierarchical representation. [cite wiki]

Methodology

General Approach Description of Marcel





Data Source

Technology Used The code for the project was developed entirely in Python. The well established libraries for data wrangling (Pandas, numpy, TensorFlow), building neural networks and analyzing results made this a logical choice for coding up this sort of prototype.

There was no need for persistent storage other than saving simulation results. Simple CSV files were used for this purpose.

All training runs of the models in this project were built on a machine with the following specifications. Note, GPU information is not included as TensorFlow's GPU extensions were not utilized for this project. Re-running the models on a box with these extensions enabled will be left for future work.

Spec	Value
CPU Cores	4 physical, 8 logical
CPU Clock	2.8 GHz
CPU Model	Intel Core i7-4900MQ
RAM	24 GB
OS (Host)	Windows 8.1
OS (VM)	Ubuntu 16.04.1 LTS

Spec	Value
Hypervisor	VMWare Workstation

Results

TODO

Marcel

TODO

Eval Metric	Value
R-squared	0.7242
RMSE	0.0370
MAE	0.0291

Trained Marcel

TODO

Eval Metric	Value
R-squared	0.7236
RMSE	0.0375
MAE	0.0296

Pure Neural Network

TODO

Run	Use.League.Avg	Optimizer	Hidden.Units	Learning.Rate	Rsq	RMSE	MAE	TrainTime
1	No	Ftrl	1000:500:100	0.0100	0.6705	0.0401	0.0308	3602
2	No	RMSProp	1000:500:100	0.0100	0.6692	0.0389	0.0298	3711
3	No	RMSProp	1000:500:100	0.0010	0.6692	0.0389	0.0298	8832
4	No	RMSProp	1000:500:100:50	0.0010	0.6611	0.0461	0.0357	12432
5	Yes	RMSProp	1000:500:100	0.0100	0.6877	0.0398	0.0308	4123
6	Yes	ProxAda	1000:500:100	0.0100	0.6745	0.0389	0.0301	5522
7	Yes	ProxAda	1000:500:100	0.0010	0.6556	0.0487	0.0381	9973
8	Yes	Adam	1000:500:100	0.0100	0.6886	0.0381	0.0291	7824
9	Yes	Adam	125:64:12	0.0045	0.6906	0.0381	0.0289	233

Run	Use.League.Avg	Optimizer	Hidden.Units	Learning.Rate	Rsq	RMSE	MAE	TrainTime
10	Yes	Adam	32:16:03:1	0.0010	0.6906	0.0381	0.0288	41

Seeded Neural Network

TODO

Run	Hidden.Units	Learning.Rate	Rsq	RMSE	MAE
1	125:64:12	0.0045	0.7271	0.03610	0.0282
2	125:64:12	0.0010	0.7286	0.03595	0.0281
3	125:64:12	0.0005	0.7267	0.03598	0.0283
4	250:128:24	0.0010	0.7267	0.03610	0.0283
5	64:32:06:01	0.0010	0.7257	0.03630	0.0285
6	128:64:12:10:6:3:1	0.0010	0.7270	0.03600	0.2830

Future Work

The methods developed throughout this project should be general enough such that holistic measures of value other than Win Shares can be projected. For example, popular metrics such as Value Over Replacement Player (VORP) and Wins Above Replacement (WAR) should be projectable to a reasonable degree with both a pure Marcel approach and a Marcel/Deep Learning hybrid algorithm. Examining how effective the various methods are at projecting each metric would be an interesting project.

A deeper dive into creating a truly useful projection system for NBA players would involving projecting each individual component statistic that goes into the win share calculation. The first step down this path would be to project defensive (DWS) and offensive (OWS) win shares separately and then adding these together to create the overall Win Share projection. Logically, this can be extended further by first calculating individual components of DWS and OWS and then combining these projections into intermediate DWS and OWS which can then be added to give a final, more accurate (theoretically) overall Win Shares projection.

While this remains an untested hypothesis for basketball projections, this method of componentized projection has worked well for me in the past when building projection systems for other sports. For example, in hockey, projecting an individual's goal scoring rate (in goals per game or goals per minute) is easy enough to do directly. Using a Marcel based system, taking a weighted average of the target player's last 3 seasons goal rates and regressing that to the mean an appropriate amount will yield a decent amount of predictive power. However, large gains are made as soon as the goal rate projection is broken up into multiple sub projections. For goals, predicting a player's shot rate (the number of shots they take on goal per game or minute) and predicting their shooting percentage (rate of shots that become goals) separately and then combining together for an overall goal rate projection yields a substantial improvement. Breaking each projection down based on basic game context (Even Strength, Powerplay, Shorthanded) improves the method even further and does not meander into the realm of overfitting.

My professional work in baseball has shown similar results to the above hockey example. Projecting a measure like weighted on base average (wOBA) is simple enough and will provide a decent estimate of a player's overall batting skill. It is easily improved upon however, as soon as you start projecting component parts such as homerun per fly ball rate, ground ball rate, strikeout rate and walk rate separately.

Conclusion

This project set out to examine the efficacy of multiple algorithms for projecting future NBA player level performance. A total of 4 algorithms were examined with three of them heavily borrowing logic from the well established Marcel the Monkey projection system which originated in the baseball analytics community.

The most advanced system combined a pure Marcel approach (weighted time average plus regression to the mean) with a wide and deep neural network. This hybrid approach achieved superior performance compared to systems using Marcel or neural networks alone. The one significant downside of any of the algorithms leveraging a neural network was the training time and computing power needed when compared to a pure Marcel approach. The neural network based algorithms required orders of magnitude greater compute time to complete.

Ultimately, the choice of algorithm would depend on use case. If projection accuracy was the motivating decision variable, the hybrid system developed here provides that. But if the algorithm needs to be run and updated multiple times per minute, the amount of computing power required may become prohibitive (depending on budget of course). The Marcel system runs very quickly, has minimal code and library dependencies and is in the same realm accuracy wise as the more complicated systems. It may very well be the correct choice in some situations.

References