Using SportVU Data and Neural Networks to Implement Dean Oliver's "Difficulty Theory"

Dan Loman

March 1, 2017

1 Introduction

One of the difficulties of player evaluation in the NBA is deciding how credit should be allocated for scoring on offense when more than one player is involved in making the play. Dean Oliver touched upon this topic in his 2007 book "Basketball on Paper" with his Difficulty Theory, which suggests that points should be credited proportional to the difficulty of their contribution leading to a score. For example, if Chris Paul finds DeAndre Jordan with a nice pass for an easy dunk, Paul's assist should take most of the credit, while if Stephen Curry makes a highly difficult shot after receiving a pass, Curry should be credited with the points. Of his theory, Oliver wrote: "All you have to do is watch every single game, estimate the chance of scoring as a consequence of every dribble, pass, pick and shot, and add things up". In 2007 this would have been impossible, but with STATS SportVU player tracking data, we can implement this theory in 2017.

The idea behind my project is to find the difference between the points a team is expected to score when a player first gets the ball and when the next player gets the ball, a shot is attempted or the ball is turned over. The player with the ball will get credited with the change in point expectancy, so he will be rewarded for doing things like passing to an open teammate which leads to a better look or shaking a defender, and penalized for putting the team in a worse position to score.

This paper details how I used a neural network to determine the point expectancy of 648,867 new states in the 2014-15 season based on the locations of the ball and player and time on the shot clock, then assigned appropriate credit to players based on how their actions increased or decreased the team's expected points per possession. This research will allow teams and fans to get a better grasp on individual offensive player contributions by adding context to each play and not penalizing players for their teammates' failure to make plays. It also has potential to identify team needs (ex: low point expectancy on shots indicates a need for better passing and ball movement) and allocate player resources (ex: a player who can both shoot and pass well should contribute more where his team lacks).

1.1 Definitions

Below are definitions of common terms I will be using throughout this paper:

Play - A period of time covered by a single play file within a game. A play can include one or several possessions and several hundred moments

Possession - The time a team gains offensive possession of the ball until there is a shot, turnover or free throw (note that I define a possession here differently than how it typically is defined in NBA analysis, where a possession ends when the opposing team gets the ball)

Moment - A moment in time captured by SportVU cameras. Each moment in SportVU data is .04 seconds apart

State - A moment in time where a new player gains possession of the ball via a pass, offensive rebound or start of a new possession

2 SportVU Data & Preprocessing

I spent a large percentage of my time preprocessing data from raw SportVU files to a consumable dataset for analysis and machine learning. My goal was to end up with a dataset where each row represents a new state, and includes spacial data - (x,y) location of each player and the ball - and temporal data - shot clock and game time information - to accurately represent the details of that state. This section details the preprocessing step in this project, which was done in python.

2.1 What is SportVU

STATS SportVU is a six-camera system installed in all 30 basketball arenas in the NBA that tracks the (x,y) positions of all players and the ball, 25 times per second. The data that results from SportVU player tracking allows teams and analysts to generate insights beyond box score statistics such as rim protection metrics, defender distance on shots, and player distance covered. SportVU player tracking stats are featured on stats.NBA.com/tracking.

2.2 Data Description

I received a data file 40 GB in size containing SportVU data for every NBA game from the preseason of the 2014-15 season until the end of the 2016 calendar year. Each game contains roughly 400-500 "plays", i.e. a chunk of the game that can include up to a few possessions. Each play file is in .json (key-value) format, is between roughly 20MB and 100MB in size and contains metadata about the game and player tracking information for each moment during the play. Each play file contains the following key fields:

1. *gameid* - The unique ID for the game

2. *gamedate* - The date of the game

3. *home* - Contains information on the home team and their players

4. *visitor* - Contains information on the away team and their players

5. *moments* - Included in the moments key are a list of moments, or snapshots in time, of a play. Each moment contains game time and (x,y) coordinates for each player on the court and the ball for each moment during the play. Moments are snapshots of the court every .04 sections, and are typically several hundred moments per play. Below is an example of a moment:

```
[1,
 1414541586032,
 720.0,
 24.0,
 None,
  [[-1, -1, 47.4393, 25.94672, 10.65305],
   [1610612740, 201569, 48.29735, 18.68403, 0.0],
   [1610612740, 201600, 58.46317, 20.60878, 0.0],
   [1610612740, 201950, 64.77708, 25.10907, 0.0],
   [1610612740, 201936, 47.45251, 33.84332, 0.0],
   [1610612740, 203076, 48.0291, 24.93867, 0.0],
   [1610612753, 202696, 46.04217, 26.94592, 0.0],
   [1610612753, 203124, 28.82642, 25.30571, 0.0],
   [1610612753, 202699, 45.89401, 34.01147, 0.0],
   [1610612753, 203901, 19.42869, 25.41476, 0.0],
   [1610612753, 203095, 45.82049, 18.77761, 0.0]]]
```

Each game folder also contains a file with play-by-play data for the game. I used this file to determine the outcome of each possession and which players were involved.

2.2.1 Data Issues

There were several issues I encountered with the raw data that should be noted. Here are the issues and how I addressed them:

1. Empty files - Some play files have no moments recorded. To resolve this I simply checked the number of moments for each play and skipped plays with no moments.

2. Duplicate files - Some play files are duplicates with the exact same information as a previous play file. To resolve this I compared the number of moments and first couple of moments in the current play file with the previous play file, and skipped the current play file if those items matched.

3. Duplicate moments - Most plays overlap to some degree with the previous play. I loaded all plays that weren't empty or duplicates and dropped duplicate rows at the end.

4. Missing players - There were a few rare cases where fewer than 10 players were tracked in a play or a player on the bench was being tracked. I discarded all of these plays.

5. Missing plays - Most games were missing plays at the end of the game; typically, the last play in a game was somewhere in the middle of the 4th quarter or earlier. There was nothing I could do to resolve this, so I worked under the assumption that player behavior was the same across all times of a game.

6. Missing shot clock - A few games had no shot clock values.

2.3 Preprocessing

While SportVU data is structured and organized, its format is far from consumable and required a lot of preprocessing to get my desired dataset, which is one state per row with points scored in the possession as an output variable.

First, I extracted all moments for all plays for a game, which are split up across hundreds of files, into one list, while excluding play files that were duplicates or empty (see "Data Issues" above). Then, I iterated through all of the moments in a game and extracted important information from each such as the game time, shot clock, quarter, ball position (x,y), and player positions. I calculated the euclidean distance between each player and the ball to determine which player was closest to the ball at (and how close they were). I used the metadata of the play to bring in player name, team and position.

Now that I had all the moments in a single data frame, I had to bring in play-by-play data to determine when important events occurred such as shots, turnovers and free throws. Parsing out the type of event, the player involved and resulting points for each event given its description required some text analysis; for example, "Davis 3' Putback Layup (10 PTS)" was parsed into a made shot, from Anthony Davis, for 2 points. I threw out events that wouldn't affect the model like rebounds and substitutions. With the play-by-play data cleaned up, I could then merge it to the moments data so that each moment could be associated with the events it led up to. Using the events from the play-by-play, I could exactly determine the team with possession and the eventual possession points for each moment. This is useful because in my final model, I'd need to know how many points each state helped result in.

Finally, I broke down all of the moments into just the moments that were new states. A new state can be one of two events: The ball is passed to a new player during the same possession, or a new possession begins. I only considered moments where the closest player was less than 2 feet from the ball, and then found moments where one of the two criteria were fulfilled, resulting in a dataset containing just new states.

2.3.1 Preprocessing difficulties

The preprocessing above covered approximately 95% of new states, but there were some special cases in the data that made it more difficult and required extra steps. Some of the difficulties I encountered are highlighted below:

1. Since the time precision of a moment is to the one-hundredth of a second and the time precision of an event in the play-by-play data is to the second, joining these two tables caused many rows of overlap which resulted in some issues. For example, plays that occurred too close together (i.e. steal then immediate turnover) were difficult to parse out, as were passes that occurred too close in time to a missed shot.

2. The ball occasionally ended up greater than two feet from the player with possession (sometimes in reality and sometimes as a glitch), so I took the first moment the player gained possession as the beginning of the state.

3. The ball might end within 2 feet of a teammate before a player's shot hits the basket or before the ball is turned over. Therefore it was important for me to link moments to their end result play based on the player involved in each.

2.4 Feature Engineering

The following features were created to be used for modeling this data. The goal is that these features will accurately depict each state:

- Ball distance from basket (feet)
- Ball angle from basket (with respect to baseline)
- Ball distance from closest defender - calculated in radians using the Pythagorean theorem
- Ball angle from closest defender (with respect to basket) - calculated in radians using the law of cosines
- Players 1-5 distance from basket
- Players 1-5 angle from basket (with respect to line perpendicular baseline)
- Players 1-5 distance from closest defender
- Players 1-5 angle from closest defender (with respect to basket)
- Shot Clock
- Possession Points (output variable) - the eventual points scored in the possession

3 Data Modeling & Machine Learning

The following section details how I calculated the point expectancy at the end of the possession of each state. All modeling was done in python.

3.1 Model selection

I tried six different machine learning models from the scikit-learn package in python to predict possession points given a state: Linear and Logistic Regression (*sklearn.linear_model*) module, Random Forest Regressor and Classifier (*sklearn.ensemblemodule*) and Neural Network Regressor and Classifier (*sklearn.neural_networkmodule*). The regression-based models simply predicted the possession points for a state, while the classification models output a set of predicted probabilities for potential point values (0, 1, 2, 3 and 4), which I combined into a single point prediction by taking a weighted average.

I used a 10-fold cross validation to pick the best model. The results of this validation, with error metrics, are shown in Table 1.

Using MAE and MSE as benchmarks I easily discarded the random forest models, which do far worse in both metrics. To choose among the remaining models I looked at how specific variables correlated with the predicted points output. I chose the three variables - distance between ball and

Table 1: 10-fold Validation Results for 6 models

Model Type	**MAE**	**MSE**
Linear Regression	1.060	1.264
Logistic Regression	1.060	1.264
Random Forest Regressor	1.066	1.393
Random Forest Classifier	1.068	1.393
Neural Network Regressor	1.057	1.262
Neural Network Classifier	1.057	1.261

basket, distance between ball and defender, and shot clock - which I felt were the most intuitive and easy to interpret, and compared the results to what I would expect based on my knowledge of the game to see which model yielded the closest result to reality. Results for Logistic Regression (Figure 1), Neural Network Classifier (Figure 2) and Neural Network Regressor (Figure 3) are highlighted.

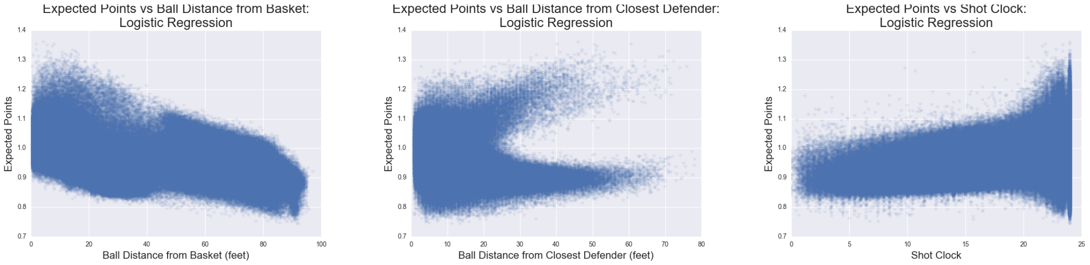

Figure 1: Logistic Regression Metrics

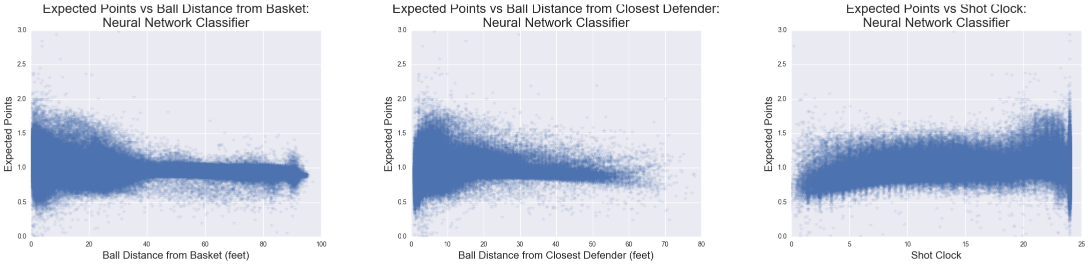

Figure 2: Neural Network Classifier Metrics

Based on these figures, I felt that the neural network classifier was the most optimal model to use going forward for four main reasons:

1. Expected points is highest the closer the ball is to the basket, then spikes again around 22-24 feet from the basket (3 point shots).

2. Expected points is lowest where ball-defender distance is close to 0 and highest when ball-defender distance 3-10 feet (which is where an open shot would be).

3. Expected points plummets when the shot clock is less than 5, and spikes at times when the shot clock is greater than 20 (likely for transition opportunities).

4. The variance in expected points from the neural network classifier is greater in all three graphs than the variance from the neural network regressor. This tells me the model is more influenced by the variables that were inputted.

3.2 Feature Selection

After settling on the neural network classifier to find the expected possession points for a state, I did feature selection to optimize the model. I used a "leave one out" approach where I iterated

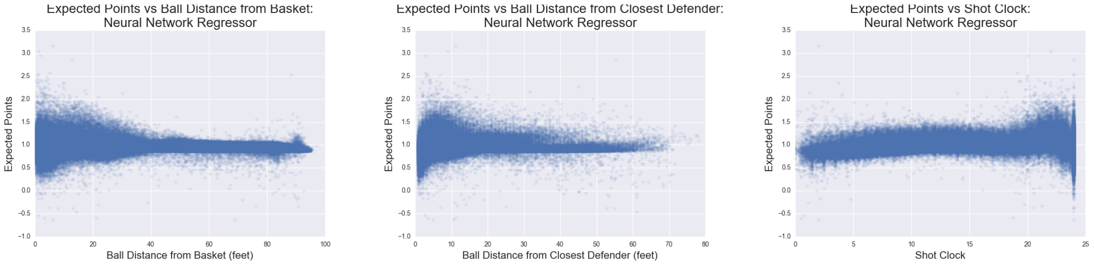

Figure 3: Neural Network Regressor Metrics

through each type of feature, omitted it from the model, and compared the model results via 5-fold cross validation. Results from this approach are shown in Table 2.

Table 2: Feature Selection results

Fields omitted	**MAE**	**MSE**
Players-Defender Distance	1.056	1.262
Players-Basket Angle	1.056	1.261
Ball-Defender Angle	1.057	1.262
Ball-Defender Distance	1.057	1.264
Players-Basket Distance	1.057	1.261
Shot Clock	1.057	1.263
None	1.057	1.263
Ball-Basket Angle	1.057	1.236
Players-Defender Angle	1.058	1.262
Ball-Basket Distance	1.600	1.264

Although the error metrics of the model were improved when several different features were omitted, the differences were so slight that I didn't feel there was enough evidence to support omitting any of the features. Since there is natural variation among neural network models I presumed that the error improvements were likely statistically insignificant, and the fact that omitting the shot clock feature, a feature I'd expect would improve the accuracy of the model, improved the error metrics led me to believe I was better off leaving all of the features in the final model.

4 Results

Once the model was finalized, I ran my neural network on the entire dataset to get the expected possession points of all states. I calculated the difference between the expected possession points of each state and the expected possession points of the next state (if the state was the last in a possession, the actual points value of the possession was used). This difference was assigned to the player that had the ball in the first state. Then, I aggregated by player and team and called the result "Expected Points Added" or EPA*.

*unrelated to Expected Points Added in football.

4.1 Player Results

Note that I used 25 games as the cutoff for inclusion in the following lists

Table 3 shows the top 10 players by EPA per game for the 2014-15 season. Most of these names are unsurprising - Kevin Durant, James Harden, Anthony Davis and Stephen Curry are all in the top 5, and sharpshooters Klay Thompson, JJ Redick, Kyle Korver and Kyrie Irving round out the top 10. George Hill at #2 definitely seems high, but Hill surprisingly finished 7th overall in ORPM in 2015 for a Paul George-less Pacers team. Wesley Johnson cracking the top 10 doesn't really have an explanation and seems to indicate a flaw in the model.

Table 4 shows the bottom 10 players by EPA per game for the 2015 season. The theme here is guards who can't shoot, led by Michael Carter-Williams and Rajon Rondo, who were also the bottom 2 point guards in ORPM that year. Given that Nerlens Noel and Lance Stephenson were also both in the top 10 in ORPM among all players, and no one else in the bottom 10 has a particularly stellar offensive reputation, the list appears to be fairly accurate.

Table 3: Top 10 Players - Most Expected Points Added

Player	**Position**	**Team**	**Games**	**Total Expected Points Added**	**Average Expected Points Added Per Game**
Kevin Durant	F	OKC	27	90.49	3.35
George Hill	G	IND	40	126.47	3.16
James Harden	G	HOU	75	228.12	3.04
Anthony Davis	F-C	NOP	65	163.65	2.52
Stephen Curry	G	GSW	77	183.33	2.38
Klay Thompson	G	GSW	73	171.24	2.35
JJ Redick	G	LAC	73	159.76	2.19
Wesley Johnson	F	LAL	75	157.97	2.11
Kyle Korver	G	ATL	71	129.3	1.82
Kyrie Irving	G	CLE	69	119.45	1.73

Table 4: Botton 10 Players - Least Expected Points Added

Player	**Position**	**Team**	**Games**	**Total Expected Points Added**	**Average Expected Points Added Per Game**
Michael Carter-Williams	G	PHI-MIL	63	-226.94	-3.6
Rajon Rondo	G	BOS-DAL	64	-196.05	-3.06
Lance Stephenson	G	CHA	58	-142.91	-2.46
Ish Smith	G	OKC-PHI	35	-70.4	-2.01
Elfrid Payton	G	ORL	78	-153.26	-1.96
Evan Turner	G-F	BOS	77	-142.77	-1.85
Joakim Noah	C	CHI	65	-119.81	-1.84
Lou Amundson	F	NYK	37	-60.21	-1.63
Nerlens Noel	C-F	PHI	73	-115.31	-1.58
Tony Wroten	G	PHI	29	-44.93	-1.55

4.2 Team Results

Table 5 shows the top 10 teams by EPA per game for the 2014-15 season, and Table 6 shows the bottom 10 teams. Of the top 10 teams, nine rated in the top 10 in offensive efficiency that year, while Houston rated 12th. Of the bottom 10 teams, eight ranked in the bottom 10 in offensive efficiency.

4.3 Limitations

1. The most obvious limitation for this research is that all of the credit for point expectancy changes goes to the ball-handler. So while a player is rewarded for getting the ball to the open man (something that there currently isn't a good metric for), the player receiving the ball doesn't get credit for getting open.

2. There's a strong correlation between players' expected points per touch and EPA per touch, which tells me that the features inputted in the model didn't affect the output as much as I would have hoped and thus the model's variance for expected points for a state is likely too low. Additionally, there's a high correlation between a player's shooting efficiency and EPA, which shows

Table 5: Top 10 Teams - Most Expected Points Added

Team	**Average Expected Points Added Per Game**
Golden State Warriors	5.39
Los Angeles Clippers	4.96
Cleveland Cavaliers	4.5
Toronto Raptors	4.31
San Antonio Spurs	3.56
Atlanta Hawks	2.68
New Orleans Pelicans	2.31
Dallas Mavericks	2.3
Houston Rockets	2.11
Portland Trail Blazers	1.52

Table 6: Bottom 10 Teams - Least Expected Points Added

Team	**Average Expected Points Added Per Game**
Philadelphia 76ers	-8.8
New York Knicks	-5.73
Charlotte Hornets	-3.48
Minnesota Timberwolves	-2.5
Orlando Magic	-2.26
Miami Heat	-1.8
Utah Jazz	-1.74
Milwaukee Bucks	-1.46
Los Angeles Lakers	-1.41
Boston Celtics	-1.25

that the model gives much stronger weight to made shots vs passes.

3. Due to the size of the dataset and of the court, I was unable to perform my desired method of machine learning, which was to simply feed the (x,y) coordinates of each player and the ball (and shot clock value) into a neural network to estimate expected possession points (similar idea to the classic neural network number classification example with the mnist dataset).

4. SportVU data issues still exist, so per-game EPA averages are deflated and don't cover entire 48 minute games. However, there isn't much variance among games for how much time is missing, so the final player rankings aren't affected much by this.

5 Conclusion/Next Steps

Overall, based on validation against more well-known statistics and reputations of NBA players, I believe my model did a decent job of ranking each players' offensive contributions by EPA. However, it's clear that the model's output for expected points is not sensitive enough to a state's attributes, so in the future I'd like to go back to the modeling phase and see how this could be corrected. If a more sensitive model is attained I'd like to be able to compare players by expected points per touch and per shot, to see which players are carrying the easiest and most difficult offensive loads on average.