Predicting Momentum Shifts in NBA Games

Whitney LaRow, Brian Mittl, Vijay Singh

Stanford University — December 11, 2015

1 Introduction

During the first round of the 2015 NBA playoffs, the eventual champions, the Golden State Warriors, found themselves pitted against the New Orleans Pelicans. In Game 3 of their matchup, the Pelicans opened the second quarter with a stretch in which they scored 19 points and the Warriors scored none. They opened the third quarter with a similar 10-0 run, and, despite comeback efforts by the Warriors, the Pelicans were up 20 points by the start of the fourth quarter. Against all odds, the Warriors managed to take momentum at this point, outscoring the Pelicans 39-19 in the quarter and eventually winning the game in overtime.

These types of momentum shifts are not atypical in basketball. The media often attributes these momentum shifts and streaks to “players getting distracted” or “not wanting it enough.” Our project hopes to delve past this rhetoric and determine effective methods to predict momentum shifts in these games. The ability to predict momentum shifts is valuable as it grants coaches the ability to counter momentum shifts in real time. We postulate that streaks are the factors that make or break games; thus, we define a streak as a point where one team scores 8 or more points while limiting its opponent to 0 points. We decided on this definition of a streak after an in-depth literature review (see References). There are several useful data points that can be used as features to reasonably predict the likelihood of a team going on a streak at any given point in an NBA game. While traditional NBA stats like fouls, turnovers, and rebounds are good features, other non-traditional stats, such as frequency and recency of timeouts, changes in lineups, and overall team offensive rating, have proven insightful in predicting streaks as well.

For our project, we use a “window” approach to predict whether or not a streak is about to occur; in other words, we look at the previous 20 events in a game (timeouts, turnovers, shots made or missed, rebounds, etc.) to predict whether or not a streak will occur in the next 20 events. We found that on average about 500 events occur in one game. The input to our algorithm is then a combination of statistics from the past 20 events along with some metadata about the teams. We then use two models, logistic regression and an SVM, to classify whether or not a streak will occur over the course of the next 20 events.

The inherent difficulty of this problem is tied to the high amount of variance in between basketball games. Every game has its own unique set of players, coaches, and referees, and so many different events can occur in a given game at any point in time. Our goal is to find features that transcend this high amount of variance and can serve as useful predictors over several unique basketball games.

2 Related Work

Related work falls primarily in a similar yet separate category: assessing the validity of momentum as an important factor in sports games. These studies typically analyze the relationship between successive outcomes in sports games, and many attempt to debunk the notion that momentum is influential in the outcome of a game. Two studies in particular evaluate winning streaks in sports with different outcomes. First, a study published in the Journal of Sport Behavior attempts to debunk the myth of momentum in sports and shows that both participants and observers place an unjustified level of importance on momentum in the result of games [9]. This study, done by hand, assesses the psychological elements of how momentum influences a game; however, another study published in the Journal of Quantitative Analysis in Sports uses neural networks to predict the outcomes of NBA games [8]. This state-of-the-art study shows that there are predictable factors including momentum that influence the outcome of games, allowing for a successful prediction rate 74.33% of the time, over 5% higher than experts.

Still, other studies attempt to analyze the effect of momentum on in-game statistics, which is closer to what this project strives to predict. A study done jointly at Cornell and Stanford University found that there is no positive correlation between the outcome of successive shots, which it attributes to the misperception that short random sequences are indicative of the ultimate generating process [7]. Thus, the notion of a “hot hand” or “streak shooting” in basketball is false according to this study. Along these same lines, a second study done at Harvard University compares the idea of a “hot hand,” or the belief that the success of one shot in a game will lead to the success of another, with the idea of “gambler’s fallacy,” or the belief that recent success must soon end [6].

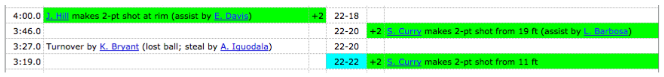
This study ultimately shows that it depends on the perceived intentionality of the streak's agent; in other words, the prediction of a streak depends on whether or not the predictor believes the streak is intentional.

From a more psychological viewpoint, another study considers exactly what constitutes a streak in the predictor's mind and finds that three repeated events must occur to solidify the subjective belief of a streak [5]. Ultimately, most of these studies only consider the psychological effects of momentum and the resulting impact on belief. Although it has been argued that shooting streaks for a particular player do not occur, this is not the only way momentum can influence the outcome of a game. Rather, factors such as rebounds, assists, timeouts, substitutions, free throws, and more all combine to influence a streak of team points in a basketball game. Hence, this study intends to show that momentum is a factor in basketball games that can be predicted via machine learning techniques through a multitude of factors.

3 Dataset and Features

3.1 Data Extraction

We use the Python library Scrapy to scrape our data from Basketball-Reference.com [1], which contains play-by-play data for every NBA game from the 2000-01 season through the 2014-15 season. We start our crawlers at the Schedule & Results page for each of these 15 seasons [4], follow links through each of the game dates [2], and then follow links through the play-by-play pages for each of the games that occurred on that date [3]. We perform our actual data scraping from each of these play-by-play pages.



4:00.0	J. Hill makes 2-pt shot at rim (assist by E. Davis)	+2	22-18		
3:46.0			22-20	+2	S. Curry makes 2-pt shot from 19 ft (assist by L. Barbosa)
3:27.0	Turnover by K. Bryant (lost ball; steal by A. Iquodala)		22-20		
3:19.0			22-22	+2	S. Curry makes 2-pt shot from 11 ft

Figure 1: Sample unstructured play-by-play data.

We first collect metadata about the game, such as date, team names, final score, amount of time spent tied, number of lead changes, and other useful metadata. We then collect a list of plays, which has information about every play that occurred in the game, such as time, players involved, play type (shot, rebound, foul, timeout, turnover, substitution, etc.), score, and other details relevant to that specific type of play. Our Scrapy client then creates a JSON file with all of this information, where each game is represented as a dict-type structure:

```
{"awayTeam": "Los Angeles Lakers",
 "homeTeam": "Golden State Warriors",
 "date": "November 1, 2014",
 ...
 "plays": [{'awayScore': "22",
            'homeScore': "20",
            'team': "away",
            'time': "3:27.0",
            'playType': "turnover"},
          {'awayScore': "22",
           'homeScore': "22",
           'team': "home",
           'time': "3:19.0",
           'points': "2",
           'playType': "madeShot"},
         ...
         ]
}
```

This scraped data then becomes our database from which we derive our train and test data. We process one game at a time, first extracting metadata features like both teams' offensive rating and record, then using a window of the previous 20 in-game events to collect statistics like count of made shots, made free throws, offensive rebounds, team timeouts, turnovers, substitutions, etc. This window of only 20 events helped us to emphasize the importance of recency of events in our analysis. We combine these metadata features with the count features from the event window to create a feature vector for every in-game event we see, also including an intercept term $x_0 = 1$. We then use our streak definition to label our dataset by determining, at the given in-game event, whether or not a streak is about to occur (i.e. will the given team have outscored their opponent by 8 or more points at any moment in a window of 20 plays into the future). We output our feature vectors, along with these associated labels, to a CSV file, which we then used to train and test our models.

We ended up only using the game data from the 2014-15 NBA season because this had over 1,000 games, which translated into over 1,000,000 in-game events, which we decided would be enough data for both our train and test examples. We split our dataset 80-20, using about 800,000 examples for our training set (which we also broke up 80-20 for our validation set) and 200,000 examples for the final test set (the one on which our error rates are reported).

3.2 Feature Extraction

After examining our play-by-plays and observing streaks that occurred in actual NBA games, we built an exhaustive list of 35 features from our JSON data: 1. Home or Away Team, 2. Record Difference, 3. Difference in Field Goal Percentage, 4. Difference in Offensive Rating, 5. Difference in Turnover Percentage, 6. Difference in Free Throw Percentage, 7. Consecutive Points Scored, 8. Recent Points Differential, 9. Points Scored by Leading Scorer, 10. Team's Made Shots, 11. Team's Missed Shots, 12. Team's Made Three-Pointers, 13. Team's Missed Three-Pointers, 14. Team's Made Free Throws, 15. Team's Missed Free Throws, 16. Team's Offensive Rebounds, 17. Team's Defensive Rebounds, 18. Team's Assists, 19. Team's Personal Fouls, 20. Team's Turnovers,

21. Team's Timeouts, 22. Team's Substitutions, 23. Opponent's Made Shots, 24. Opponent's Missed Shots, 25. Opponent's Made Three-Pointers, 26. Opponent's Missed Three-Pointers, 27. Opponent's Made Free Throws, 28. Opponent's Missed Free Throws, 29. Opponent's Offensive Rebounds, 30. Opponent's Defensive Rebounds, 31. Opponent's Assists, 32. Opponent's Personal Fouls, 33. Opponent's Turnovers, 34. Opponent's Timeouts, 35. Opponent's Substitutions

We realized that including this many features could cause overfitting, and that many of the features were probably not useful, so we had to determine which features were the most effective in predicting streaks. To do this, we utilized the recursive feature elimination tool from the Python SciKit Learn library, which works essentially the same way as backward search. Backward search works by starting with the entire feature set and removing one at a time, tracking which of the resulting sets produces the best generalization error. It then repeats the process until it reaches the desired number of features.

We applied backward search to evaluate our feature set with respect to our Logistic Regression model. This produced a set of 12 optimal features (using the previous enumeration): 2, 4, 7, 8, 9, 10, 14, 21, 22, 24, 32, 33.

We also evaluated our features using our SVM, which produced an optimal feature set of 16 features: 1, 4, 6, 7, 8, 10, 15, 16, 17, 23, 24, 25, 26, 28, 29, 35.

4 Methods

4.1 Logistic Regression

The problem at hand is essentially a binary classification problem. At any given moment during a basketball game, we want to be able to predict whether or not a streak will occur. Given that our outcome variable is categorical, and our predictor variables may be either continuous or categorical, Logistic Regression is a natural starting point.

Logistic Regression models work by producing a weight vector, θ, which defines a hypothesis, $h_\theta(x)$, that is used to produce a probability score between 0 and 1. For a given feature vector, x, the hypothesis of the model produces the value:

$$h_\theta(x) = g(\theta^T x) = \frac{1}{1 + e^{-\theta^T x}}$$

This value necessarily falls between the values 0 and 1. Which values are classified as a streak and which are classified as a non-streak is determined by the decision (cutoff) boundary. Given a decision boundary, B, all prediction values $0 < p \leq B$ are classified as a non-streak and all prediction values $B < p < 1$ are classified as a streak.

To determine the optimal decision boundary, we implemented and ran 5-fold cross fold validation (splitting our test set 80-20 each time) on the range of values from 0.01 to 0.50, with 50 steps. This gave us an optimal decision boundary of 0.1, as it maximized the sum of the true positive and true negative rate (see graph below).

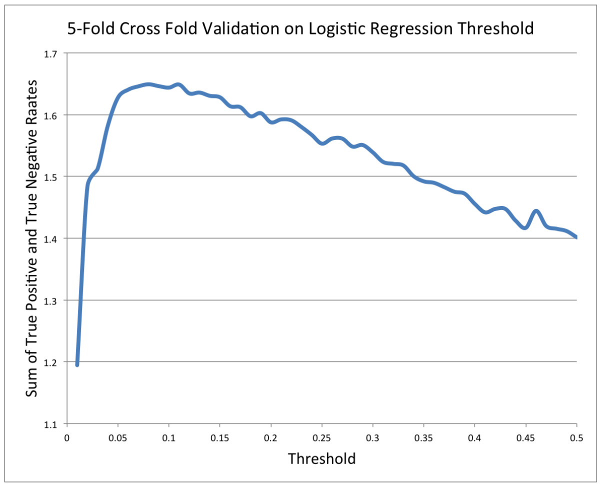


Figure 2: Graph showing the effect of changing the Logistic Regression decision boundary on test performance through the use of 5-fold cross fold validation.

To find the weight vector, θ, we fit the Logistic Regression model with training data. The model runs an algorithm like stochastic gradient ascent to maximize the likelihood of θ, using the update rule:

$$\theta_j := \theta_j + \alpha(y^{(i)} = h_\theta(x^{(i)}))x_j^{(i)}$$

We implemented our Logistic Regression model in Python using the Pandas library to handle and operate directly on our data and the Statsmodels library to create our actual Logistic Regression model.

4.2 Support Vector Machine

We also modeled our problem with a support vector machine to predict whether or not a streak will occur. To date, few supervised learning algorithms have outperformed SVMs; perhaps the most alluring property of SVMs is that we can utilize their symbiosis with kernels. This lets us create high or infinite dimensional feature vectors, allowing us to capture the subtle interactions between features, which is particularly important for our dataset because it contains so many intertwined and interdependent features. Thus, the SVM presents an ideal classification model that does not limit our ability to utilize a dense, high dimensional feature vector to determine the state of the game.

One disadvantage of SVMs, however, is that they are easily susceptible to overfitting because they work in such a high dimensional feature space. Since it is likely that the extracted data will become linearly separable, we run the danger of the model learning hidden attributes of our data rather than the more general trends we are looking for.

On the other hand, mapping features to a high dimensional space doesn't necessarily guarantee that the data becomes separable. We also don't know, in our case, if we want to find a separating hyperplane because it is likely that our data contains outliers (due to its highly variable nature), and we don't want to fit our optimal margin classifier to these confounding outliers. To make an effort to account for these issues, we also use ℓ_1 regularization, which incorporates an error term that allows for the disregard of outliers.

We optimize the resulting primal problem to find the optimal margin classifier, which defines our SVM:

$$\min_{\gamma,w,b} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{m}\xi_i$$
$$\text{s.t. } y^{(i)}(w^T x^{(i)} + b) \geq 1 - \xi_i, i = 1, \ldots, m$$
$$\xi_i \geq 0, i = 1, \ldots, m$$

SVMs constructed in this manner work to maximize the geometric margin for every point by constructing a separating hyperplane in high-dimensional space (as most data sets are not linearly separable) that classifies as many points correctly with the largest margin possible. Intuitively, geometric margin can be understood as the functional margin

$$\hat{\gamma}^{(i)} = y^{(i)}(w^T x^{(i)} + b)$$

scaled by $\|w\|$. The result is a classifier with the lowest possible generalization error, which is the ultimate goal of any classifier.

For our SVM, we used the Python library, SciKit Learn. We found that a linear kernel worked best on our dataset and features. We also implemented 5-fold cross fold validation (splitting our test set 80-20 each time) to determine other optimum values like $C = 1.2$, $\gamma = 1$, and 200 maximum iterations.

5 Results and Discussion

5.1 Final Results

Our final models produced the following success rates:

	True Positive Rate	True Negative Rate	Sum
Logistic Regression Train	0.785	0.862	1.647
Logistic Regression Test	0.787	0.860	1.647
SVM Train	0.644	0.678	1.322
SVM Test	0.638	0.681	1.319

The most important numbers are the sums of the true positive and true negative rates for the test set for both the SVM and the Logistic Regression model. We achieved an overall success rate of 1.319 for our SVM and 1.647 for our Logistic Regression model. Because these numbers are significantly higher than 1.0 (the success rate of a model that guesses randomly) we claim that our methods were successful in predicting scoring streaks in NBA games. A visual representation of the success of our models can be seen below:

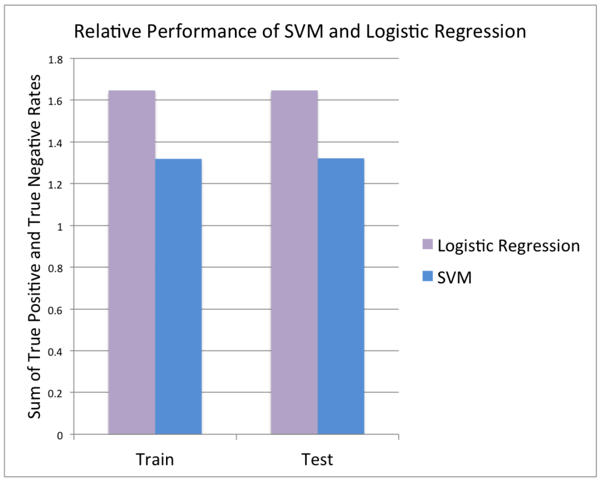


Figure 3: Final performance rates for our SVM and Logistic Regression.

While both our Logistic Regression and SVM models performed significantly well, the Logistic Regression model still outperformed the SVM. This could be for a variety of reasons. While SVMs can be very powerful and outperform most other models when appropriate, they are known to perform poorly under certain conditions. For example, if the dataset is susceptible to outliers, which ours may be, or if the high dimensionality of the examples causes the SVM to pick up trends seen only in the training data, the SVM could generalize poorly. Logistic Regression models, which are typically simpler, are not susceptible to these same pitfalls. It could be that the data followed trends which are more easily picked up by Logistic Regression models than SVMs. In the end, the results show that testing a variety of models helps produce the best results, as different models have different strengths and weaknesses, which aren't always apparent until tested on the data.

Interestingly enough, with our most successful model, Logistic Regression, not many traditional NBA statistics remained as useful features. For instance, both rebounding and assists, conventionally considered important stats, were not used in the predictions. This is probably because these events occur so frequently regardless of whether or not a streak about to occur. To illustrate, a rebound occurs at least once in every possession with a missed shot, and an assist usually occurs in a possession with a made shot.

Instead, the relevant features included stats about the team in general that did not have to do with the specific game (such as record difference and offensive rating), current momentum (such as recent points differential), and rarer in-game events (such as timeouts and substitutions). Despite the high variance in NBA games, these features consistently proved to be effective predictors across our entire large dataset, showing that there are similarities in how streaks occur across the NBA.

5.2 Look-Ahead Window Variation

We thought it would be interesting to look more into our original design ideas for our model, so we experimented with changing the look-ahead window size in consideration when training and testing on our dataset. Originally, we trained on the previous 20 events to predict whether or not a streak will occur in the next 20 events. What if we predict whether or not a streak will occur in the next 5 events instead? The next 35 events? This window controls how far into the future we are attempting to predict. How does changing this window size, making it larger and smaller, influence our results?

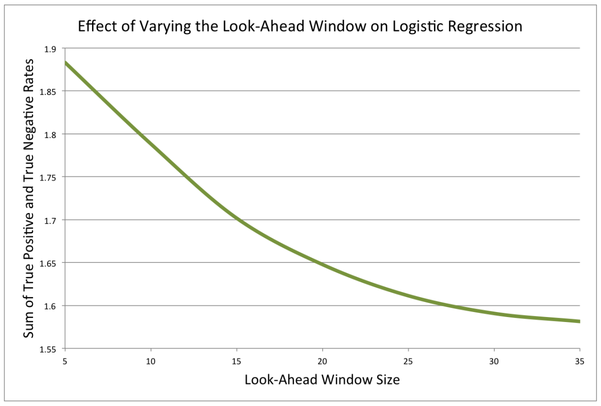


Figure 4: The effect of varying the look-ahead window size on the performance of our Logistic Regression model.

The results of this experiment make sense: our Logistic Regression model performs poorly when we ask it to look far into the future and well when we ask it to only predict the outcome of the next few plays. This shows that the recency of events in a game is important in predicting a scoring streak (events happening now have more effect on events in the near future than on events in the far future).

5.3 Single Feature Removal

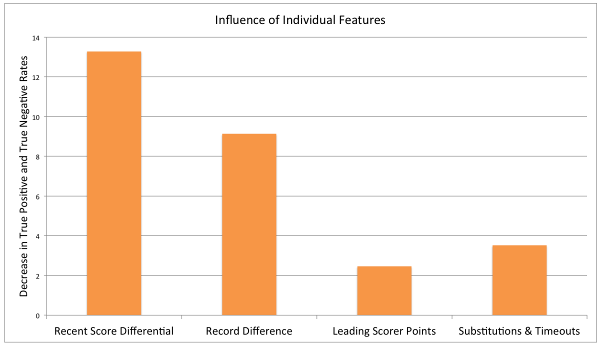


Figure 5: The effect of removing individual features from our feature set on the performance of our Logistic Regression model, as measured by decrease in true positive and true negative rates.

The above four features were some of the most important predictors in our Logistic Regression model, based on how our rates dropped when they were removed. By a wide margin, recent points differential was the most important predictor. This feature captures the difference in scoring between the two games over the last 20 events, indicating that momentum in NBA games builds on itself and can lead to streaks.

While the effect of the points scored by the leading scorer was not nearly as dramatic as the previous two features, it was still non-trivial given the size of our dataset. The importance of this feature as a predictor suggests that a team's leading scorer may often be the primary agent driving a streak. From a basketball perspective, star players often dominate the game, as bench players and role players usually score at a dramatically lower rate and are unlikely to produce a streak on their own.

6 Conclusions & Future Work

Our results showed that momentum shifts, as modeled by our streak definition, can be predicted with relatively high accuracy and are determined by a variety of factors. Ultimately, Logistic Regression produced better classification results as a model, but our SVM model was still able to correctly identify momentum streaks at around a 65% true positive and true negative rate. This shows that momentum is a factor of not just scoring but the overall state of the game. Given more time and resources, we would like to explore neural networks as a potential model for predicting momentum streaks in NBA games. Neural networks are promising for this project because of their ability to learn features. Since momentum is defined by a complex system of factors that are difficult to define by hand, neural networks could prove even more successful at predicting momentum shifts than Logistic Regression and SVMs.

Looking at other potential features, such as injuries on a team or data regarding the referees, could also prove to be useful predictors. Outside the scope of this project, future research into how coaches could use this data to change their in-game strategies would be very useful. For instance, if an opposing team seems likely to go on a streak, a coach could call a timeout or make substitutions to change the dynamic of the game. Looking more in-depth into the events that cause streaks to end would also be useful in this context.

Overall, this project successfully shows that NBA games are influenced by various factors that increase the likelihood of momentum shifts. This breakthrough is important as it could be used to influence in game decision making to optimize performance at the professional basketball level through machine learning and statistical analysis.

References

[1] Basketball-reference.com homepage. `http://www.basketball-reference.com` [Accessed: Nov 2015].

[2] Game date example page for october 31 2000. `http://www.basketball-reference.com/boxscores/index.cgi?month=10&day=31&year=2000` [Accessed: Nov 2015].

[3] Game play-by-play page for charlotte hornets at atlanta hawks. `http://www.basketball-reference.com/boxscores/pbp/200010310ATL.html` [Accessed: Nov 2015].

[4] Schedule & results example page for the 2000-01 nba season. `http://www.basketball-reference.com/leagues/NBA_2001_games.html` [Accessed: Nov 2015].

[5] Kurt A Carlson and Suzanne B Shu. The rule of three: How the third event signals the emergence of a streak. *Organizational Behavior and Human Decision Processes*, 104(1):113–121, 2007.

[6] Eugene M Caruso and Nicholas Epley. Hot hands and cool machines: Perceived intentionality in the prediction of streaks. In *Poster session presented at the 5th annual meeting of the society for personality and social psychology, Austin, TX, USA*, 2004.

[7] Thomas Gilovich, Robert Vallone, and Amos Tversky. The hot hand in basketball: On the misperception of random sequences. *Cognitive psychology*, 17(3):295–314, 1985.

[8] Bernard Loeffelholz, Earl Bednar, and Kenneth W Bauer. Predicting nba games using neural networks. *Journal of Quantitative Analysis in Sports*, 5(1), 2009.

[9] R Vergin. Winning streaks in sports and the mispreception of momentum. *Journal of Sport Behaviour*, 23(2):181–197, 2000.

Note: We used this project's data and infrastructure for another class project, with the permission of the instructor.