A Novel Approach to Predicting the Results of NBA Matches

Omid Aryan
Stanford University
aryano@stanford.edu

Ali Reza Sharafat
Stanford University
sharafat@stanford.edu

***Abstract*—The current paper presents a novel approach to predicting the results of basketball matches played in the NBA league. As opposed to previous work that merely take into account a teams own performance and statistics in predicting a teams outcome in a match, our approach also uses the data known about the opposing team in our endeavor to make that prediction. Our findings show that utilizing the information about the opponent improves the error rate of observed in these predictions.**

I. Introduction

Statistics and data have become an ever more attractive component in professional sports in recent years, particularly with respect to the National Basketball Association (NBA). Massive amounts of data are collected on each of the teams in the NBA, from the number of wins and losses of each team to the number of field goals and three-point shots of each player to the average number of minutes each player plays. All the data that is collected from this great sport is truly intriguing, and given that it is usually interpreted by people in the sports profession alone, it is a fantastic and pristine field for people in the machine learning field (and other data-mining sciences) to apply their techniques and draw out interesting results. Thus, we plan to do just that and utilize the algorithms we learn in CS229 to predict the NBA matches.

The paper provides a novel approach to predicting these matches. As opposed to previous work done in this field, where the available data is directly fed to the algorithms, we intend to find a relationship between the data sets provided for the two teams in a match and modify the data accordingly before feeding it to our algorithms. Section II will describe the source and format of the dataset we will use to train our models. Section III will present an overview of the model we intend to implement along with a description of each of its components. Section IV will present the results of our implementation, and section V will conclude the paper.

II. Data Set

We collect our data from basketball-reference.com. For each game we get a set of 30 features for each team (e.g., http://www.basketball-reference.com/boxscores/201411160LAL.html). The features are listed in Table 1.

TABLE I: Features given for each team and each game in the season

Minutes Played	
Field Goals	Blocks
Field Goal Attempts	3-Point Field Goals
Field Goal Percentage	3-Point Field Goal Attempts
Turnovers	3-Point Field Goal Percentage
Personal Fouls	True Shooting Percentage
Points	Effective Field Goal Percentage
Free Throws	Offensive Rebound Percentage
Free Throw Attempts	Defensive Rebound Percentage
Free Throw Percentage	Total Rebound Percentage
Offensive Rebounds	Assist Percentage
Defensive Rebounds	Steal Percentage
Total Rebounds	Block Percentage
Assists	Turnover Percentage
Steals	Usage Percentage
Offensive Rating	Defensive Rating

We created a crawler that scrapes the website and collects the entire data set for all games for the past 5 seasons (2008-2013) and stores them in a csv file per season. We call this data set the game data. Each team plays on average 82 games per season, so we have about 2600 data points in each of those data sets. We primarily use this data set to predict features corresponding to teams playing an upcoming game.

We also collect seasonal data for each team (e.g.,

http://www.basketball-reference.com/leagues/NBA_2013.html). Here, there are 5 tables of features for each of the teams in the league. We combine 4 of these tables (Team Stats, Opponent Stats, Team Shooting, Opponent Shooting) into one table, resulting in 98 features per team. We call this data set the team data. We primarily use this data set to cluster teams in order to have more accurate feature predictions for upcoming games.

III. Model Overview

The previous Section described the data set with which we intend to train our models. Each training example is of the form $(\boldsymbol{x}, y)$, which corresponds to the statistics and output of a team in a particular match. $\boldsymbol{x}$ is an N dimensional vector containing the input variables and y indicates whether the team won ($y = 1$) or lost ($y = 0$) in that match.

Given such a data set, a naive approach would be to feed the training examples directly to the existing algorithms in machine learning (e.g., logistic regression, support vector machines, neural networks, etc.) to predict the outcome of a match. This approach is undertaken in the referenced work. However, this method merely gives a prediction of whether a team wins or loses *regardless* of which team it plays against.

In this paper we plan to tackle the intrinsic problem of the aforementioned method and take a different approach. Instead of directly utilizing the training examples in Section II to train our parameters and feed to our algorithms, we intend to modify them based on the matches they were obtained from as well as the relationship that the different features have with one another. For example, the statistics that a team is expected to have in each match depends on the team it is playing against. We plan to take this into account by clustering the teams into different groups and to predict their expected statistics based on: a)the relationship that the two clusters corresponding to the two teams have with one another and b)their average statistics from the previous matches. Moreover, once we have a prediction of how each team will perform with respect to each of the features, we will congregate the two feature sets of the two teams into a single feature set based on how those features relate to one another. Ultimately, this single feature set is fed to our algorithms in order to make a prediction.

An overview of our model is depicted in Figure 1. It is composed of the following three components:

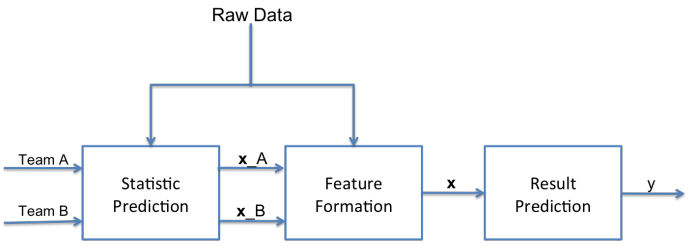

Fig. 1: Model Overview

1) **Statistic Prediction:** In this component the two teams to play in a match are given as input (e.g., Los Angeles Lakers and Boston Celtics) and the expected statistics of each of the teams ($\boldsymbol{x}_A$ and $\boldsymbol{x}_B$) in that match is predicted.
2) **Feature Formation:** This component forms a single set of input features $\boldsymbol{x}$ from $\boldsymbol{x}_A$ and $\boldsymbol{x}_B$ based on how the individual features (x_i's) are related to one another.
3) **Result Prediction:** This component contains the different machine learning algorithms we intend to utilize to predict the final outcome based on the input feature set $\boldsymbol{x}$ from the previous component.

A. Statistic Prediction

We use several different predictive models to predict the features for an upcoming game, using all the data points from the previous games. We describe all such models below:

1) Running average: In this model, the feature values of a team in an upcoming game are predicted using the running average of the features of that team in the previous games it has played in the season. This is the simplest prediction method one can use to predict upcoming games. Since this method relies on the team having played at least one game already, we do not make a prediction for the first game of teams in the season.

2) Exponentially decaying average: This is similar to the previous model, but with the difference that the previous games are weighed by a factor of $0 < \alpha < 1$. That is, if a given team has played n games already with g_i representing its features in the ith game, our prediction for the $n + 1$st game will be

$$g^*_{n+1} = \frac{1-\alpha}{1-\alpha^n} \sum_{i=1}^{n} \alpha^{n-i} g_i$$

The point of using a decaying average is to test

the hypothesis that results have temporal dependence. That is, more recent results are more relevant than older results. The lower the value of α, the higher the importance of the recent results.

3) Home and away average: The hypothesis we aim to test using this prediction is whether or not teams perform differently at home and away. We use two running averages for each team, for home and away games respectively. Then, if the upcoming game is a home game for a given team, the running home game average is our prediction. Similarly, we predict features if the game is away from home.

4) Cluster-based prediction: The aim of this method is to see if we can predict the behavior of a team, based on the type of team they play. In order to cluster the teams, we use our team data set, where for each team we have 98 features. We first perform PCA to reduce the number of features that go into clustering to n. Then, we perform k-means clustering to get k clusters, then we keep a running average of each team's features agains teams of each cluster. That is, we keep n running averages for each team. Then, if the upcoming game of a given team X is against a team which belongs to cluster i, our prediction of X's features is the running average of X's features against teams in cluster i.

B. Feature Formation

A distinctive aspect of our approach from the previous work is that we take into account the statistics of the opposing team before making a prediction. In other words, we create a training example (to be fed to our algorithms) for each *match* rather than for each *team*. This training example would be the output of this component, where the input would be the two (expected) training examples of the two teams (which is also derived with knowledge of the other team by the previous component). Hence, the task of the Feature Formation Component is to form an input variable vector for the match based on the expected input variables of each team.

For our implementation, we have carried out the most simplistic method of comparison, which is to simply take the difference of the predicted feature values of each team to form the final feature set, i.e.:

$$\boldsymbol{x} = \boldsymbol{x}_A - \boldsymbol{x}_B$$

An output of $y = 1$ would then indicate team A's victory, while $y = 0$ would indicate team B's victory (basketball matches have no draws and one team must win).

C. Result Prediction

Once we have the predicted feature set of a match, we can then predict the outcome of that match by means of any machine learning algorithm. The algorithms we utilized are linear regression, logistic regression, and support vector machines (SVM). For linear regression, the score difference of the two teams was used to train and test the data.

IV. Implementation and Results

The data was trained and tested for each season individually. A collection of five seasons were tested from 2008 to 2013. The foregoing results indicate the average of the results achieved from these seasons. The hold-out cross validation technique was used for training and testing, where 70% of the matches in a season were used to train the algorithms while the rest of the matches were used for testing purposes. The reported error rate was computed by taking the average of 100 iterations of this technique with different training and testing sets. Furthermore, the forward search feature selection algorithm was used to choose the most effective features for each of the algorithms.

Here we break down our results based on the Statistics Prediction method used.

1) Running average: The running average is the simplest predictor for the the features of an upcoming game. We simply train our classifiers on a 70/30 training/testing set split. The results are shown in Table II. We see that linear regression turns out to be the best classifier in this case. We note that training and test errors were generally very similar. One possible reason for that is that we used *predicted statistics* for both runs. The fact that we were using a prediction to train a classifier means that the noise from our predicted statistics cannot be subsumed during training.

2) Exponentially decaying average: We use the decaying averages and feed them into our 3 classifiers to predict the outcomes of the matches in the training and test sets. We experimented with various values of the decay parameter, α. The results are shown in Figure 2. Similar to the previous part, linear regression performs the best amongst the classifiers. Both training and test errors decrease as the value of α increases, which signifies that there is no temporal relationship between the

TABLE II: Training and test errors when using running averages as predictors

	Training error	Test error
Linear regression	0.3231	0.3198
Logistic regression	0.3281	0.3251
SVM	0.3317	0.3304

results (that is, the results from early in the season are as important as the latest results). The training and test errors for all values of α are still higher than those from running averages.

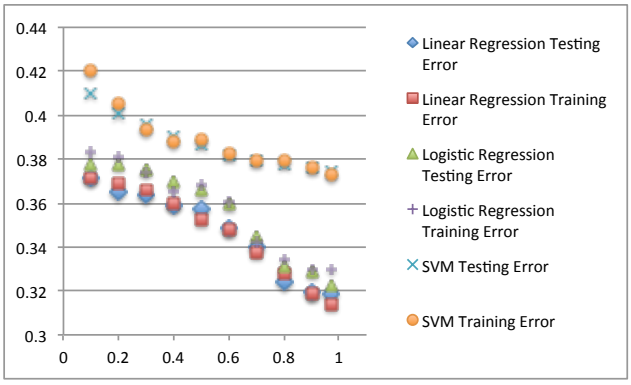

Fig. 2: The test training errors as a function of α (x-axis)

3) Home and away average: The classification part is very similar to the two previous sections. The training and test errors are shown in Table 3. We see that linear regression performs better in training and is marginally better in testing. The other two classifiers perform worse, however. This refutes our hypothesis that teams perform differently at home and away from home. We see that the running average (surprisingly) is still our best predictor.

TABLE III: Training and test errors when using running home vs away averages as predictors

	Training error	Test error
Linear regression	0.314	0.3174
Logistic regression	0.3371	0.3327
SVM	0.3412	0.3397

4) Cluster-based prediction: The classification part is similar to the previous parts. We experimented with different numbers of cluster and different PCA components during the prediction phase. The results are shown in Figure 3. We find that in general the training and test errors increase with the number of clusters and the number of components in PCA. We find that logistic regression performs best amongst our classifiers, just slightly better than how linear regression performed on the running averages.

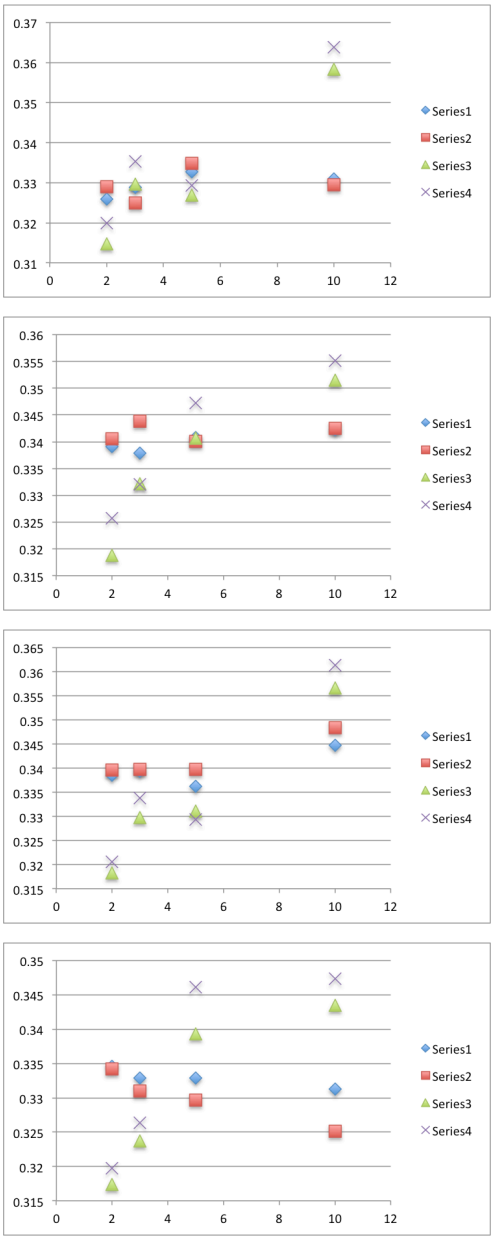

Fig. 3: Training and test error rates for various number of PCA components (4, 10, 20, 30 from top to bottom) and number of clusters (2, 3, 5, 10 x-axis).

V. Conclusion

We used a variety of predictors to predict features of teams in an upcoming game of basketball. We then used those predictions to form a single feature set corresponding to a single match. We then used that feature set to predict the outcome of the corresponding match. We found that running averages of the feature set was consistently the best predictor. We also found that linear regression and logistic regression performed the best when it came to classification, with SVM at a distant third. The reason for the poor performance of SVM is that our data was not nicely separable. Our training and test errors are in line with the literature referenced in the bibliography.

References

[1] Beckler, Matthew, Hongfei Wang, and Michael Papamichael. "Nba oracle." *Zuletzt besucht am* 17 (2013): 2008-2009.

[2] Bhandari, Inderpal, et al. "Advanced scout: Data mining and knowledge discovery in NBA data." *Data Mining and Knowledge Discovery* 1.1 (1997): 121-125.

[3] Cao, Chenjie. "Sports data mining technology used in basketball outcome prediction." (2012).

[4] Miljkovic, Dejan, et al. "The use of data mining for basketball matches outcomes prediction." *Intelligent Systems and Informatics (SISY), 2010 8th International Symposium on.* IEEE, 2010.