Predicting the Outcomes of NBA Games

Frank Longueira
Electrical Engineering
The Cooper Union

Matthew Smarsch
Electrical Engineering
The Cooper Union

***Abstract*—This research project establishes a framework for determining the outcome of NBA games through the use of machine learning algorithms. By taking team statistics per season, assuming they are jointly Gaussian, and correlating them with winning, a model was created such that machine learning algorithms, like linear regression, k-nearest neighbors, support vector machines, and neural networks, could determine winners of games to be played at a high accuracy.**

I. Introduction

Being able to predict the outcome of a sporting event has plenty of obvious applications, from fantasy sports to gambling. As two avid sports fans and players, we thought that exploring the applications of various machine learning techniques in sports would be incredibly interesting. Thanks to its long, 82 game, seasons and team statistic contribution to overall success, basketball lends itself nicely to a classification study.

A. Related Works

We found the website b̈asketball-reference.comp̈articularly helpful in this study thanks to [1]. Additionally, we found the benchmark figures, which we valued our performance against, in [2]. Specifically, it was reported that ẗop, expert analystsc̈an hope to achieve 70% accuracy in their predictions for the outcome of NBA games. Apart from these two pieces of information, all of the other elements of this study (features, classifiers, etc.) were deliberate design choices in the construction of our model.

II. Feature Collection & Organization

We decided that we would train our models using the five seasons before the one which we were attempting to predict the outcomes for. For each game that we were either training or testing on, the feature space could be broken down into four categories:

1) Away Stats
2) Away Stats Allowed
3) Home Stats
4) Home Stats Allowed

The stats that we ultimately decided to use in our models were Win %, FG %, 3P%, FT%, ORB, DRB, TOT RB, AST, STL, BLK, TOV, PF, and PTS. Using the Beautiful Soup Python API, three scripts were used to get training data, testing data, and the game schedules from www.basketball-reference.com. For both the training and testing data, the aforementioned stats were gathered for each element of the list above. Additionally, these features were normalized by Z-score to remove any bias. The final refinement in features was the subtraction of corresponding feature sets (Away Stats - Home Stats and Away Stats Allowed - Home Stats Allowed) in order to get a relative measure of the teams' offensive and defensive prowess.

A. Training Data

The training data was collected for the 2008 through 2013 NBA seasons. For each year, each team's stats per game and stats allowed per game were collected. Additionally, the total league schedule was collected for each season. The league schedule allowed us to train our model by running down the list of games, grabbing the appropriate away and home features for the teams playing, subtracting the corresponding features, and inputting our final feature inputs.

B. Testing Data

Testing data was collected for the entire 2014-2015 season. Unlike the training data, this data had to be collected as a running average of stats per game for each team. This way, when we went down the season schedule, we would pull the stats for each team - the running average of all their stats through that game of the season. This distinction is significant because it ensures that the classifiers only have stats for past games, nothing for the game in question or any future games.

III. Classification Methods

Four common machine learning classifiers were used in this study and their performance can be found in the Results section:

1) Linear Regression
2) K-Nearest Neighbors
3) Support Vector Machine
4) Autoencoder

First, we applied linear regression and k-nearest neighbors classification to the data. They are simpler models and we figured they could give us quick insight into our problem. Our feature space was 25 dimensional, so sparsity can largely impact these classifiers. For this reason, we used Principal Component Analysis to reduce the dimensionality of the feature space. We took the eigenvectors that described 95% of the variance in the feature space. This variance calculation was found by taking the sum of the eigenvalues of these eigenvectors divided by the sum of all eigenvalues pertaining to the entire orthonormal basis set generated by PCA. This

reduced the feature space to 14 dimensions. As is common practice, the value of k for kNN was chosen to be the square root of the number of data points in our training set. Using the 2008-2013 seasons and this reduced feature space, we trained the linear regression and kNN models. The trained models were then applied to the 2014-15 season for testing. Linear regression produced a correct classification rate of 71.1% for 1194 NBA games, while kNN produced a correct classification rate of 69.3%.

Next, we looked at two, more advanced classification algorithms: SVM and an autoencoder implementation. The support vector machine was implemented using a radial basis function kernel. Based on prior research, this kernel function seems to generally be the most robust option. Accordingly, there were two free parameters, scaling factor and box constraint, within the model that had to be optimized for peak classification accuracy. This optimization process required using a gridsearch on the range between 10-5 and 105 for both parameters. The measure to be optimized for this gridsearch was the classification rate, of the generated SVM model, on a validation set constructed from the training set using 10-fold cross-validation. This procedure resulted in a scaling factor value of 12 and box constraint of 0.1. Using these optimal values, an SVM model was built using the same training set as before and applied to the 2014-15 test set. The SVM model correctly classified at a rate of 71.0%, quite comparable to the previous two models. Finally, a single, hidden layer autoencoder network with softmax output layer was constructed for classification. This network had multiple free parameters, including node size, and optimal values were found by taking a greedy algorithm approach towards maximizing the classification rate on a validation set taken from the training set. These parameters were found, such as a node size of two, and the network was trained. The training took place by first pre-training the autoencoder layer and softmax layer separately and then stacked together. Since there is a probabilistic nature in setting initial rates which can affect classification rates, the model was generated and applied on the test set 25 times and the average correct classification on the test set was 71.8%.

IV. Results

Classifier	*Accuracy*
Linear Regression	71.1%
K-Nearest Neighbors	69.3%
Support Vector Machine	71.0%
Autoencoder	71.8%

TABLE I
CLASSIFICATION RESULTS

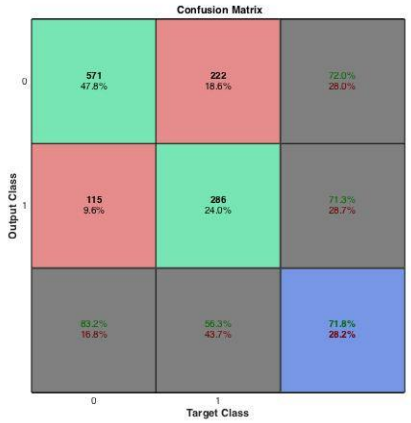

Fig. 1. Autoencoder Confusion Matrix

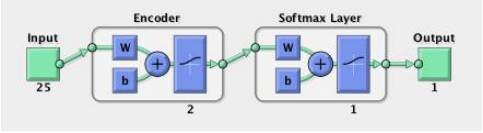

Fig. 2. Autoencoder Structure

V. Conclusion

As we expected at the beginning of our project, the autoencoder produced the most accurate classifications. However, as displayed by Table I, the performance of all of our classification models were within about 2% of the autoencoder accuracy.

This implies, taking into account computational complexity, that the linear regression model would be the best model to use in practice. All of our results, are at or above common benchmarks for predicting NBA games. An important point to make here is that our model predicted extremely well on a regular season, but would predict even better during the playoffs. This conclusion is due to the nature of how our model trains classifiers by taking into account the entire seasons stats, which during the playoffs we would have as prior knowledge.

Acknowledgment

We would like to thank Professor Keene for helping us come up with the idea for our project. It was a very fun and interesting project that we enjoyed completing, especially considering the strength of our results.

References

[1] Torres, Renato Amorim. P̈rediction of NBA Games Based on Machine Learning Methods(̈2013): n. pag. University of Wisconsin Madison, Dec. 2013. Web. http://homepages.cae.wisc.edu/ ece539/fall13/project/AmorimTorres_rpt.pdf.

[2] Cheng, Bryan, Kevin Dade, Michael Lipman, and Cody Mills. P̈redicting the Betting Line in NBA Games(̈2013): n. pag. Stanford University, 2013. Web. http://cs229.stanford.edu/proj2013/ChengDadeLipmanMills-PredictingTheBettingLineInNBAGames.pdf.