NBA Teams Season Standings Prediction

097209 Machine Learning Project Report

Omri Lidor - 301014536
Nadav Shai Oved - 200689768

Abstract

In recent years, there has been an increase in interest and capital investment in different sports fields worldwide, and particularly in the NBA. Naturally, this drove the level of competition in the NBA to much higher levels than before, and in conjunction with recent technological advancements, teams have begun adopting analytical thought processes to traditionally classical aspects of their organization, in order to make better decisions and achieve better basketball performance. The introduction of new training devices, statistical and computational methods allow teams to gather, process and analyze much more informative data than ever before.

In this paper we will attempt to predict an NBA team's next season standings according to current season performance metrics, using unstructured and structured machine learning models and algorithms.

We shall evaluate our different models according to prediction accuracy and custom score metrics and based on our results in each stage, implement variations and optimizations to our original learning models and algorithms. We found that the learning task at hand is possible yet harder than it seems at first, and we hope that our results and conclusions could lead to further analysis and improved learning models for NBA related questions and sports related data science.

Problem Definition

The fundamental question of what makes an NBA team successful is complex and affected by many different variables, on and off the basketball court, which doesn't seem likely to have a definite correct answer. Nevertheless, we believe there would be great value in being able to analyze NBA teams' performance metrics over time, in order to improve the ability to provide a prediction for teams measures of success. Our project's main goal is to analyze the ability of machine learning models and algorithms to predict the next season's final standings for a given NBA team, in relevance to playoff qualification from conference standing positions 1-4/5-8/9-15, based on teams' historical performance data. We decided to represent these team conference standings groups by labeling each team in a given season with the label "Standings_Bucket" - values {0,1,2}:

- 0 = Finals contender in current season (positions 1-4)
- 1 = Qualified to playoffs in current season (positions 5-8)
- 2 = Disqualified in current season (positions 9-15)

We constructed our dataset from statistical data gathered from basketball-reference.com, by collecting team season average performance metrics for all NBA teams dating back to the year 2000 (over 17 years past). We thought that 18 seasons is a sufficiently large baseline sample size (535 samples) for our learning models and algorithms in the basic part of the project. For the advanced part, we added data to create datasets of sizes 24 and 31 seasons, divided the data to decades (1987, 1997, 2007) and conferences (east, west) for further analysis.

The performance features we selected for our models are a subset of features from the following list:

Season - f.e. 2016-2017 (years in date format, index 1-17)
TeamID - A unique representation of a team/franchise (int 1-30)
E/W - Eastern or western conference (bool)
Conference Finalist - Did the team get to its conference finals ? (boolean)
W/L - Win/Loss ratio (float)
3PA - Team's average 3 points attempts per game (float)
2PA - Team's average 2 points attempts per game (float)
ORB - Team's average offensive rebounds per game (float)
DRB - Team's average defensive rebounds per game (float)
AST - Team's average assists per game (float)
STL - Team's average steals per game (float)
BLK - Team's average blocks per game (float)
TOV - Team's average turnovers per game (float)
PTS - Team average points per game (float)
Pace - An estimate of number of possessions per game (float)

Models and Algorithms

In order to examine the feasibility of reaching our main learning objectives, we initially decided to define 2 basic learning models for all teams in the dataset:

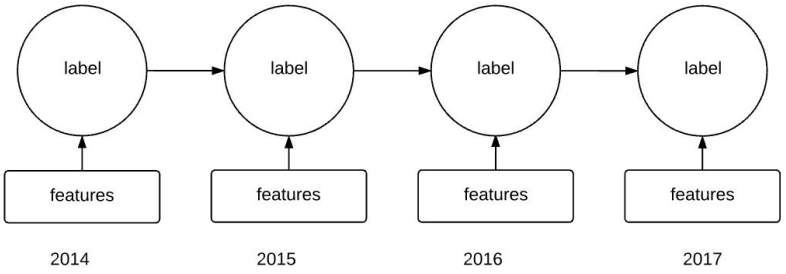
1. Learn current season standings bucket according to current season performance
2. Learn next season standings bucket according to current season standings and performance

The idea behind these models is to first verify our basic assumption about our learning algorithms' ability to infer a team's standings in the current season from its current performance data (simpler objective), and then use the same algorithms to tackle a harder objective of predicting the next season's standings according to current performance. For each data model we evaluated and compared between multiple learning algorithms, models and approaches. We first chose to compare between unstructured and structured learning approaches, since we believe that a learning model which expresses the dependency between seasons for all teams, will be able to learn a team's success pattern better than an unstructured model. For our unstructured approach, we implemented the Perceptron, SVM, Logistic Regression and Random Forest learning algorithms and compared their performances.

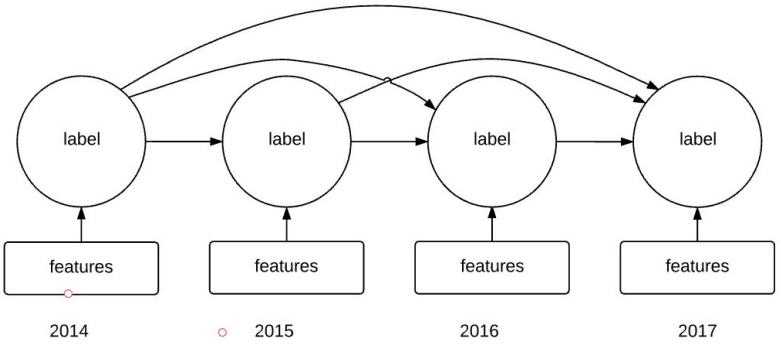
For our structured approach, we initially designed a linear chain CRF graphical model and chose to implement a Structured Perceptron algorithm, using Viterbi as our inference algorithm.

We constructed our graphical model according to the following method:

We generated a chain of nodes per team (divided into sub-chains according to sequence length - see figure below), where each node represents a prediction for a given season in chronological order.



We then concatenated each team's chain (in random team order) to form a long linear chain containing all teams. Clearly, the order of teams in the resulting chain could affect our results, we will elaborate on this in the Experiments and Results section. For the advanced part, we chose to implement the following variation to our graphical model: a chain CRF graph model where each node in the sub-chain is connected to all following nodes in its sequence. (see figure below)



We thought that a regular linear chain CRF would not express a more complex dependency between seasons and thus, might not be able to learn more sophisticated patterns we believe exist in the data.

For the creative part, we chose a different modelling approach and decided to build an independent chain CRF learning model for each team (30 models total). This approach comes in contrast to the basic structured and unstructured models we had originally implemented. It doesn't create a general prediction model for an NBA team by learning from all teams at once, instead it assumes independence between teams and aims to capture each team's unique characteristics and correlations individually. To achieve this, and to handle the significant reduction in amount of data each model receives now (31 seasons per model), we relaxed our original label to a binary label - Qualified/Disqualified (positions 1-8/9-15) for a given season's playoff.

Experiments and Results

Throughout the evaluation and analysis process for our models and algorithms, we used 2 main metrics:

1. Accuracy - percentage of correct labeling (higher is better)
2. Score - defined by the following calculation: $\frac{1}{n} \cdot \sum_{i=1}^{n} \left| p_i - r_i \right|$ (lower is better)

$\boldsymbol{n}$ = total number of predicted samples, $\boldsymbol{i}$ = predicted sample index, $\boldsymbol{p}$ = predicted label value, $\boldsymbol{r}$ = the true label value.

In order to evaluate our basic learning model 1 for the current season, we ran all our different algorithms on an 18 season dataset with a test size of ⅓, where the Standings_Bucket label refers to the current season's team's position, and got the following results:

Comparison of all basic algorithms for current season

Classifier	Score	Accuracy	# of Iterations
Perceptron	2.422	0.424	1
SVM	0.819	0.452	1
Logistic Regression	2.361	0.519	1
Random Forest	0.207	0.807	1
Structured Perceptron with Linear Chain CRF	0.609	0.557	30

We clearly see that the unstructured classifier Random Forest outperforms all other algorithms by a large margin in both Accuracy and Score, followed by our structured perceptron classifier. The unstructured models run much faster than our structured model. These results indicate, that even for the simpler learning objective, the data is not linearly separable since all classifiers (apart from Random Forest) have a very hard time to infer and predict correctly. We thought that due to the nature of Random Forest being a decision tree based classifier, it was able to perform well on this kind data, and would be able to learn the harder objective for next season (model 2). As for the Structured Perceptron, we believe some adjustments to the feature set, graph model and algorithm's parameters are required in order to improve its performance, which we will elaborate later on in this section.

We then ran the same evaluation on model 2 (next season label) objective, and got the following results:

Comparison of all basic algorithms for next season

Classifier	Score	Accuracy	# of Iterations
Perceptron	3.578	0.273	1
SVM	0.779	0.478	1
Logistic Regression	2.361	0.452	1
Random Forest	0.505	0.590	1
Structured Perceptron with Linear Chain CRF	0.747	0.482	30

We can see that all classifiers perform worse on this objective (as expected), yet Random Forest is the leading classifier, followed by Structured Perceptron. We conclude from this stage of our analysis, that our original learning objective is harder than we thought it would be, and decided to continue our analysis for Random Forest and Structured Perceptron in the following advanced section. We will attempt to improve our classifiers and models, while analyzing our feature set, the effects of different parameters per classifier and model variations.

Unstructured models analysis:

While analyzing our unstructured Random Forest classifier described in the Models and Algorithms section, we chose to focus on the following parameters:

- **Dataset size**
  Experimenting with different dataset sizes showed that the best results were received for 18 season dataset. Increasing the dataset size to 24 seasons decreases results in a 4% drop in accuracy and an 11% increase in score. Increasing it to 31 seasons results in a 1% drop in accuracy and a 2% increase in score. A possible hypothesis to explain this somewhat surprising result, would be that after the year 2000 there was a significant transition across the NBA in teams' style of play. It transitioned from a "big man" style in the 90's where most plays relied on tall players near the basket to a faster "small ball" game that relies on faster plays and outbound shooting in the 2010's. This transition could make it difficult for the learning algorithm to capture patterns over the entire dataset.
- **Important features**
  Our analysis showed that the three most important features for Random Forest, using the 18 seasons dataset, were BLK, PTS, ORB respectively. As we examined these features, a follow-up question arose: Are these important features significant in each conference and decade independently ?
  - **Decades**
    Our analysis showed that the most important feature in all 3 decades is BLK. The second most important feature in the 90's and in the 2010's was PTS, and in the 2000's it was AST.
  - **Conferences**
    The 2 most important features in both east and west conferences were BLK, followed by PTS.

  We've concluded that according to Random Forest, BLK is the most significant feature to predict a team's measure of success, and its significance does not decrease over the decades or conferences from which the team's data was gathered. Before this project, we believed that scoring the most points or passing the most assists were the best indicators for a team's success. This outcome is rather surprising to us, but interesting nonetheless. Our guess is that every successful team in every decade must have had a dominant tall player which increased the number of blocks per game, which exhibited his dominance on defense.

Our best run for Random Forest predicting next season's label (model 2) was for 18 seasons dataset on a subset of the original features with a score of 0.505 and accuracy of 0.631.

Structured models analysis:

While analyzing our structured models described in the Models and Algorithms section, we chose to focus on the following parameters:

- **Dataset size**
  Our analysis showed that changing the dataset size did not have a significant direct effect on our classifiers performance and that the best results were received for our 24 season dataset with a test size of ⅓, while 18 and 31 seasons were very close behind it. It is worth noting that a larger dataset (i.e. more seasons) allows us to choose from a greater range of sequence lengths.
- **Sub-chain sequence length (within Chain CRF models)**
  Each team's chain is constructed by several sub-chains of length X. This parameter proved to be very influential on our classifier's performance, since it directly affected the sequence fed to our inference method, i.e. how many seasons back will our inference algorithm try to fit for in each sub-chain. When the sequence length was large (above 10), we received very poor results, since the inference method was trying to predict a very long sequence of dependent seasons, which might be unrelated, and therefore smaller trends in those seasons were overlooked. When the sequence length was too small (1-3), we did not receive satisfying results, since the dependency structured between seasons could now be almost meaningless. We received the best results for sequence lengths of 4 on our 24 season dataset. We believe that this number also fits the dependency between seasons in reality, since most teams are affected by their performance in several recent seasons more than further back in history.

- **Order of teams in chain CRF models**
  When evaluating our structured model we encountered a fundamental issue of the order of teams in our linear chain. This proved to greatly affect our classifier's performance, since the order of teams in the chain yields a specific sequence for the classifier to infer from.
  Our proposed solution to this issue, is to randomly shuffle the order of the teams in the chain in each iteration on the dataset, and run a higher number of iterations to decrease the chances of our classifier being biased by a specific order of teams in the linear chain CRF. Moreover, we use the averaged weights version of the Structured Perceptron, to retrieve the averaged weights over all iterations.
- **Chain CRF Graphical Model Variation**
  While this variation of the chain CRF model we implemented, captures more dependencies between subsequent seasons within the sub-chains we found that the main effects of this model were a huge increase in runtime per iteration (over double), and a small but noticeable improvement (1-3%) on our classifier's performance for our larger datasets (24 and 31 seasons)
- **Number of iterations on dataset**
  We have already established that we need to run multiple iterations on the dataset in order to randomize the order of teams in the chain, yet we found that above 100 iterations there was almost no effect on our classifier's performance. The best results were received when this parameter was between 50-100.
- **Important features**
  Since our structured models contain joint features between data features and labels, we could derive which features were highly correlated to each label. Our structured models exhibited the following features as most significant: Label 0 - PTS and W/L, Label 1 - 2PA and Pace, Label 2 - PTS and Pace. As for label sequence features, the most significant in both CRF models were features exhibiting bigram and trigram sequences of the same label (i.e. (0,0), (2,2,2), etc…).
  In contrast to the unstructured models results, BLK didn't appear as a very significant feature.

Our best run for the Structured Perceptron predicting next season's label (model 2), was on our modified chain CRF graph model for 24 seasons dataset, sequence length of 6, 50 iterations, on a subset of our original features list: Averaged Test Accuracy of 0.572 and Score of 0.537.

Creative Section:

In this section we decided to implement a prediction model for each team as described in the end of Models and Algorithms section. In our experiments we used a 31 season dataset, trying both chain CRF graph models from the previous sections, with a sequence length of 3, a test size of 3 last sequences and 500 iterations over the dataset per model. This allowed us to analyze each team's model independently, and extract the significant features for each team, which varied across different teams. The interesting result we got throughout our experiments, was that some teams (f.e. San Antonio Spurs, Sacramento Kings) prediction models were consistently highly accurate, while other teams prediction models were not (f.e. LA Lakers, Chicago Bulls). This could be explained by the fact that in reality, some teams had been very unsuccessful for a long time and suddenly had a very successful streak and vice versa, while other teams succeeded (or failed) relatively consistently. To compare the results from this section to previous sections, we averaged accuracy and score measures over all 30 models. Our best results for this model were Average Test Accuracy of 0.643 and Average Score of 0.444

Comparison between our best models and news agencies for 2017 season

Model	Score	Accuracy
Random Forest	0.505	0.631
Structured Perceptron	0.537	0.572
Per Team Model	0.444	0.643
Average of News Agencies	0.307	0.727

Discussion and Conclusions

The main challenge we faced throughout all our experiments, was understanding the relationships between the different features and models, and constructing a model that accurately exhibits a structure which logically fits the data. We were faced with a constant trade-off between designing models which we thought would generalize well, and models which forced a specific structure to the data we had.

Our main conclusions from this project are, that while it is possible to learn for this task, its main objective revealed itself to be much harder for this level of machine learning methods than we thought. We think that in order to generate a higher quality prediction, a different, more complex approach is required, that incorporates in the learning model other factors, features and structures which are not necessarily performance related (i.e. player ratings, important trades, draft picks, player salaries, etc…).

We believe this learning task should be further explored and we would recommend to consider a more informative set of features, further refining the per team models, our modified chain CRF graphical model and experiment with other structured learning algorithms which we didn't explore. We feel that although Random Forest exhibited better accuracy, an unstructured model would be limited (for this task) only to its feature selection and might miss hidden structural patterns (f.e. time or team dependencies) in the data.

While we are somewhat disappointed with our overall prediction accuracy across our different models, we feel that we came to some interesting conclusions, explored different ideas and discovered some interesting results along the way. Overall, we enjoyed our learning process and would be eager to continue exploring such challenges in the future.