Prediction of NBA 2K Player Ratings

Aarsh Patel
Department of Computer Science
University of Massachusetts Amherst
Amherst, MA 01003
aarshpatel@umass.edu

Bhavik Jain
Department of Computer Science
University of Massachusetts Amherst
Amherst, MA 01003
bjain@umass.edu

1 Introduction

NBA 2K sports is a popular video game developed to emulate the sport of basketball. Player emulation is what really defines the great gameplay provided by NBA 2K. NBA 2K in recent years has tried very hard to make the game and the players as realistic as possible. However, perfecting player styles and player ratings in this video game can be really difficult. The problem we are tackling is generating NBA 2K ratings using previous years stats. Currently NBA 2K sports needs to go through a very tedious process to generate ratings for all NBA players at the end of every NBA season.

In order to help expedite this process we tested several different models such as linear regression, ridge regression and random forest regression to predict a player's NBA 2K rating based off of their previous years statistics. We wanted to see how accurately we could predict a player's rating based on how they did during the previous season. This is an interesting problem to deal with because we wanted to know which features of an NBA player makes the biggest differences in deciding their 2K rating.

Currently one of the main issues with 2K player ratings is that there is some bias in player rating generation. It seems as if 2K sports uses an eye test for many players, in which they judge player talent with their eyes instead of just through stats. Figure 1 depicts Rating vs. Games Started for all players in the dataset. From this figure, we were able to see a few outliers in the data. Looking at figure 1, we can see two major outliers. A player named Paul George started zero games the entire season, due to injury, and played in only six games that season as well as averaging only nine points per game but was rated as 90 in NBA 2K. Another player named Manu Ginobili started 0 games the entire season and played in 62 games that season as well as averaging about 11.75 points per game and was rated an 86 because of his previous accolades. These are outliers in the dataset because Paul George was only rated as a 90 because he is a super star and was given that because of his name as is the reason why Ginobli was rated so high. As a result, we can assume that NBA 2K factors in a form of bias for some players depending on their potential as well as their star status.

We felt that this project was creative because there isn't much work done with video games related machine learning. Everyday we hear about machine learning being applied to medicine, finance, computer vision and etc but we don't really see it being applied to video games. As a result, we thought using machine learning to predict player ratings would be interesting and unique.

We performed many different experiments to evaluate our models and our best results came from using Ridge Regression with hyperparameter optimization.

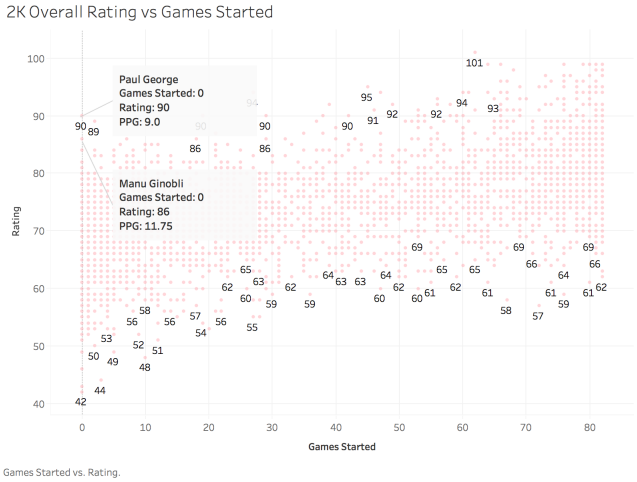

Figure 1: Graph showing 2K rating vs Games started. The graph depicts some outliers in the data where players didn't play that many games but had high 2K ratings.

2 Related Work

While NBA 2K is a very popular game, not many attempts have been made at building a machine learning model to predict a players' rating (1-99) based off of their season stats.

An attempt to predict 2K player ratings from seasonal stats was done by two college students, Christopher Kober and Shuai He [3]. They built a dataset which composed of just rudimentary regular season player stats combined with their NBA 2K ratings. Realizing that all of the features may not be critical in predicting the rating, they found some features (co-dependent and not helpful) they could remove from the dataset. Feature engineering their dataset helped them find the top eight features which directly impacted the 2K rating of a player and were able to use those eight features to train different types of regression models. After trying a variety of different models, their best results came from training and testing a Nearest Neighbor model which returned a MAE (Mean Absolute Error) score of 3.9135 which was able to beat their baseline model (predicts the mean of all player ratings) score of 7.92. Our work is similar to theirs in that we have the same goal in mind, but we are approaching it in a different way. We are using advanced statistics of these players whereas they are not. We strongly believe that incorporating these extra statistics into our models will outperform their models for 2K rating prediction.

Birlutiu et al. [2] attempted to use expected propagation (EP) for rating players in sports competitions specifically in tennis. EP is an approximation technique which tunes the parameters of a simpler distribution inorder to match the exact posterior distribution of the model parameters given the data. A Bayesian approach into rating tennis players using EP as an approximation technique is used. They take into account a player's strength as a probabilistic variable in the Bayesian framework and incorporate player information into the prior distribution. Thus, they computed the posterior distribution over the player's strength using Bayes' rule. They called this model EP-correlated because it takes into account the correlations between strengths of players. Their dataset consisted of tennis matches among 1000 players and the goal was to compute ratings for these players based on the match outcomes. They compared their results to another type of model that uses Assumed Density Filtering instead of EP. ADF is an approximation technique in which the terms of the posterior distribution are added one at a time. Combining EP with player strength did significantly better (using p-values) than the ADF model on the dataset. They concluded that this simple probabilistic model using EP as an approximation technique does a good job on predicting player ratings based on tennis matches and suggested that a more complex model could outperform EP on this task. This work is similar to ours in that they are trying to predict ratings for tennis players based on match outcomes, whereas we are trying to predict ratings for players in a video game. However, they are approaching this problem in a different way and are using more advanced methods such as bayesian statistics to help solve their problem.

Besides predicting ratings of players in video games or sports, rating prediction can also occur in the movie industry as well. Salakhutdinov et al. [5] were able to model user's ratings of movies, specifically using the Netflix dataset, using a class of two-layer undirected graphical model called Restricted Boltzmann Machine. The RBM is a stochastic artificial neural network that can learn a probability distribution over its input. They also tested out a Conditional RBM which extends the regular RBM by capturing temporal dependencies. In order to perform learning for these models, they use constrastive divergence which greatly reduces the variance of the estimates used for learning. It relies on an approximation of the gradient of the log-likelihood based on a short Markov chain. These models make predictions for a new movie by picking the rating with the maximum score (unnormalized) as the prediction. Using these models they were able to produce an error rate that is well over 6% better than Netflix's own baseline RMSE score of .9514. This work is related to ours in that they are also predicting ratings but for user rating for movies. It's also similar in that they also formalized their problem as a regression problem but they used a more advanced method, Restricted Boltzmann Machine, inorder to predict these ratings.

3 Datasets

In order to accurately predict a player's 2K rating for a particular year, we needed to gather their statistics for that year. Thus for each year, a player is represented by some feature vector (shown below) corresponding to their stats for that season and a target label corresponding to the 2K rating given to them.

For each year, we downloaded a CSV file from *http://www.basketball-reference.com*[1] that contains all basic stats of each player and a CSV file that contains all advanced stats for each player. We combined both CSV files to form one big CSV file that contains all player information (basic + advanced stats) for that year. We've collected these player statistics for years between 2000 - 2016.

We also needed to get 2K ratings for each player for each year. We used beautifulsoup and the requests library in python to scrape *operationsports.com*[4] for these player ratings. We were able to scrape all 2K player ratings from 2000 to 2016 except for years 2005, 2006 and 2007.

Thus our final dataset consisted of player data from 2000 to 2016, except years from 05-07. The dataset has 3980 data cases with 49 features. The features include a combination of continuous and discrete values. Some features that the dataset contains are *points_per_game*, *minutes_played*, *games_started*, *assists_per_game*, *blocks_per_game* and *field_goal_percentage* etc.

$$player^{(i)} = \left[x_1, x_2, x_3 ... x_{49}\right]$$

$$y^{(i)} = rating$$

4 Methodology

4.1 Libraries

For this project we used the following libraries: Scikit Learn [6], numpy, pandas, and matplotlib. We use Scikit Learn for machine learning, numpy for array manipulation, pandas for dataframe manipulation and matplotlib for graphing.

4.2 Data Preprocessing

As described in the section above, our data consists of player statistics and 2K player ratings. The raw data contains a lot of errors and inconsistencies. Some of the features contained missing data so we imputed values of 0 in that case. We realized that some features were actually linear combinations of other features. For example, *total_rebounds_per_game* is just a combination of *offensive_rebounds_per_game* and *defensive_rebounds_per_game*. Thus we just removed the *total_rebounds_per_game* feature from our dataset. Another feature we removed was the *total_win_shares* because it is a combination of *offensive_win_shares* and *defensive_win_shares*. We

also engineered a new feature, *game_started_percentage*, from *game_started* and *game_played*. We thought that having a continuous value representing the percent of games started for a player would be a good feature to include into our dataset. We also split up the dataset into a train and test set in which we used a 80% train and a 20% test split. We only used the test set at the end to see how well our modesl generalized to new unseen observations.

4.3 Feature Selection

We wanted to use feature selection on the dataset because it would help identify the most relevant features in predicting a player's overall 2K rating. We used mutual information as an implementation of feature selection. Mutual information measures the dependency between two random variables X and Y. More specifically, it tells us the amount of information obtained about one random variable through the other variable. In this problem, it measures the dependency between the NBA statistics and a player's 2K ratings. If this value is 0 then the two are independent of each other otherwise higher values mean higher dependency.

4.4 Evaluation Metrics

Since this is a regression problem we used MAE (mean absolute error) to evaluate how well our models were performing. MAE measures the average magnitude of the errors in the set of predictions (without considering direction). It sums the absolute values of the residuals divided by the number of observations. MAE is defined below:

$$MAE = \frac{1}{n}\sum_{j=1}^{n}|y_j - \hat{y}_j| \tag{1}$$

Our dataset contains numerous outliers and in that aspect we decided to use MAE because it is more robust to these outliers and thus gives us a good performance metric to evaluate our models.

4.5 Cross Validation & Hyperparameter Optimization

In order to evaluate our models, we use 10-fold cross validation using scikit learn's cross_val_score method. The main advantage of using cross validation than using a holdout set is that k-fold cross validation uses all of the data for training and for testing. The idea behind using this approach is that it will reduce the bias in the performance estimates by using more training data in contrast to setting aside a relatively large portion of the dataset as test data.

We also use cross validation to find the optimal hyperparameters for our regression models. We use scikit learn's GridSearchCV method to find the optimal hyperparameters for a given model. GridSearchCV does an exhaustive search over the parameter space while performing cross validation to figure out the model with the best hyperparameters.

4.6 Baseline Model

Before building our predictive models, we wanted to build a good baseline to give us a sense of what we are trying to achieve. Since this is a regression problem we decided to build a model that just predicts the mean of all of the 2K player ratings. We suspect that the baseline model will give us a good place to start trying new regression models and hope that these regression models beat the baseline model in terms of MAE.

4.7 Different Models

We wanted to try different regression models and see if any of these models would be better than the baseline model we created in the previous section. We tried a few different regression models such as Linear Regression, Ridge and Random Forest Regressor to see if these models would help improve prediction of player 2K ratings. For each of these models, we found best features using mutual information and optimal hyperparameters and then used the cross validation scoring function

(cross_val_score) from sklearn to return the MAE of the model. Below is a detailed description of the core models used to solve our problem.

4.7.1 Linear Regression

Linear Regression is a parametric regression method that assumes the relationship between y and **x** is a linear function. The Linear Regression function is defined as the following:

$$f_{lin}(\mathbf{x}) = \left(\sum_{d=1}^{D} w_d x_d\right) + b = \mathbf{x}\mathbf{w} + b \tag{2}$$

We assume that there is some linear relationship between the stats of a player for a season and the player's 2K rating for that season. As a result, we believe that linear regression is a good model to begin our experiments.

4.7.2 Ridge Regression

Ridge Regression is the name given to regularized least squares when the weights are penalized using the square of the ℓ_2 norm $\|\mathbf{w}\|_2^2 = \mathbf{w}^T\mathbf{w} = \sum_{d=1}^{d} w_d^2$. The equation used for Ridge Regression is:

$$\mathbf{w}^* = \underset{\mathbf{w}}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^{N} (y_i - \mathbf{x_i}\mathbf{w})^2 + \lambda \|\mathbf{w}\|_2^2 \tag{3}$$

We experimented with a Ridge Regression because this type of model is able to solve the problem of co-linear features. This is a big issue for the dataset we are using because all of the advanced stats are created using the normal seasonal stats of a player, hence most features are co-linear. The other reason for using this method was that the penalty term reduces overfitting.

4.7.3 Random Forest Regressor

Random Forest Regression is an ensemble regression method that works by averaging multiple regression trees during training time with the goal of reducing the variance of the model. Each tree constructed in the forest is trained on a different parts of the training set. Also these trees are grown very deep which tends to create overfitting problems. By averaging these deep trees, we reduce the variance while increasing the bias of the model slightly. Eventually, the performance of the model increases dramatically as compared to just one single regression tree. We experimented with Random Forest Regressor because it's very good at handling tabular data with numerical and categorical features. Also, Random Forests are able to capture the non-linearity between the features and the target, whereas Linear and Ridge Regression cannot.

5 Experiments & Results

5.1 Feature Correlation against 2K Ratings

One of the goals of the project was to figure out which features correlate well against 2K player ratings. Below is scatter plot of different features against 2K ratings. We showed these specific features because our intuition was that these would be highly correlated with a player's rating.

The scatter plot shows that there is some linear relationship between the features and the ratings. There are definitely some outliers but overall there is some underlying relationship and we expect our model to extract this.

We wanted to do a more in-depth analysis on the correlation between the features and ratings. The table below shows the top 10 features found using mutual information. Mutual information measures the dependency between the NBA statistics and the 2K ratings. From the graph above, we expected that PPG, Minutes Played, and Player Efficiency Ratings were important features in predicting a player's 2K rating. The mutual information shown in the table supports our hypothesis.

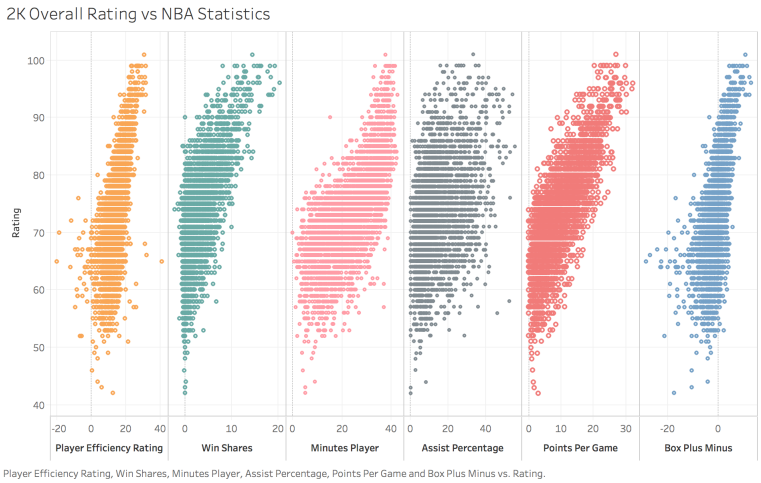

Figure 2: Different NBA Statistics against 2K Ratings

Feature Name	Mutual Information
Points Per Game	.467
Field Goals Per Game	.436
Field Goal Attempts Per Game	.433
Free Throw Attempts Per Game	.404
Minutes Played	.395
2 Point Field Goals Attempted Per Game	.394
2 Point Field Goals Per Game	.380
Player Efficiency Rating	.373
Free Throws Per Game	.371
Turnovers Per Game	.355

5.2 Hyperparameter Optimization with Feature Selection

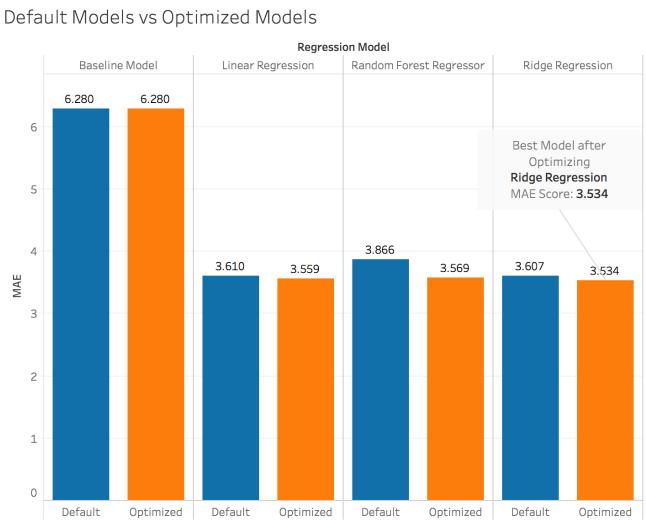

Figure 3: Default Regression Models vs Optimized Regression Models

We wanted to compare different regression models and see how well they could predict a player's 2K rating based off seasonal statistics. We built a pipeline that finds the best hyperparameters including the optimal number of features for a given estimator. In order to perform feature selection, we use mutual information in combination with sklearn's *SelectKBest* method that finds the top features according to mutual information. SelectKBest takes in an hyperparameter k which is the number of top features to select. For our previous experiment, we hypothesized that SelectKBest would remove some features be-

comes some have very low mutual information scores. For hyperparameter optimization we use sklearn's GridSearchCV which performs cross validation in order to find the best hyperparameters for a given estimator. For example, in order to optimize Ridge Regression we decided to optimize the number of top features *k* from 45 to 49 and *alpha* in the set {0, .01, .001, .1, 1, 10}. For this model we figured out that the optimal number of features *k* was 47 and the optimal alpha value was 10. The optimized Ridge Regression model scored a MAE of 3.534 on the test set which turned out to be our best performing model. The graph shown on the left depicts other models we tested out and how it compared to its counterpart with default hyperparameters. All models (default and optimized) ended up beating the baseline model and all optimized models were better than their default counterparts. From our experiments, we concluded that Ridge Regression was our best performing model.

5.3 Basic Stats vs Advanced Stats

We hypothesized that using all of the player statistics (regular and advanced) would yield better results than just using regular stats or just advanced stats for 2K rating prediction. Thus we conducted an experiment to prove our hypothesis. A combination of all features gives a clear picture of a players' impact/performance during a given season, removing a subset of this data does not allow us to see the full picture. We created a split on the dataset and created two subsets (basic stats and advanced stats). After training and testing four different models with the different subsets of the data we were able to compare the MAE scores for all of the different models and the datasets the model was run on (Figure 4). We can see that Ridge Regression was the best model when run on the entire dataset. The results of this experiment prove our hypothesis that the combination of basic stats and the advanced stats make a much bigger difference in determining the NBA 2K ratings for a player. It is imperative to keep all the features of the dataset because it helps show a better correlation of a player's season averages and how the player was ranked. Keeping the advanced stats is very important because these stats show certain characteristics of a player which a normal box score may not show.

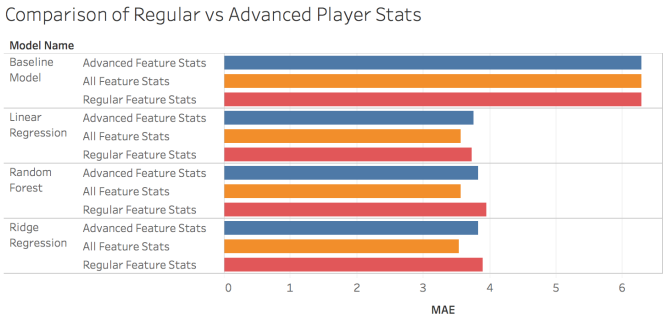

Figure 4: Comparison of Regular vs Advanced Stats

5.4 Offensive vs Defensive Statistics

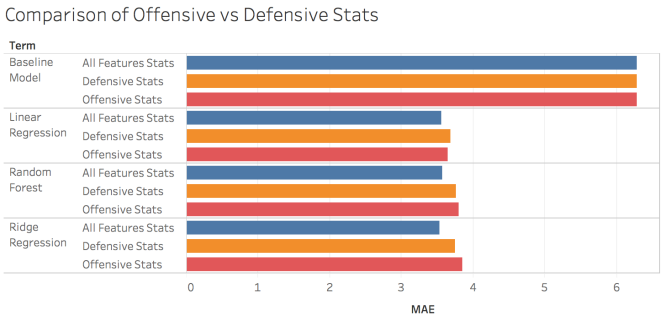

Figure 5: Comparison of Offensive vs Defensive Stats

We hypothesized that using only the offensive stats rather than all stats would return better results. To prove this theory, we conducted an experiment. We thought that the defensive stats are not as important in determining how a player would be rated because the biggest aspect to the game of basketball is being able to score (offense). We created a split on the dataset and created two subsets (offensive stats and defensive stats). After training and testing four different models with the different subsets of the data we

were able to compare the MAE scores for all of the different models and the datasets the model was run on (Figure 5). We can see that Ridge Regression was the best model when run on the entire dataset. The results of this experiment prove that our hypothesis was incorrect and that the combination of offensive stats and defensive stats make a much bigger difference in determining the NBA 2K ratings for a player. As important as offense is to the game of basketball, for accurate ratings of a player, the defensive stats are just as important because these show how the player is able to do on the other side of the court.

5.5 Final Results on Heldout Test Set

As described above, we held out 20% of the data as our final test to see how well our models generalize to new data. Below is a table depicting each of our models tested and its test MAE score.

Model	MAE Score
Linear Regression	3.70
Ridge Regression	3.69
Random Forest Regressor	3.79

From the table above, our models did infact overfit slightly on the training data. Ridge Regression with optimized hyperparameters (features and alpha value) turns out be our best model with an MAE score of 3.69, which slightly beats Linear Regression. This tell us that the underlying relationship between player statistics and 2K player is somewhat linear and that Ridge Regression could not have done much than Linear Regression.

6 Discussion & Conclusions

Based on the experimental results we draw the conclusion that Ridge Regression with optimized hyperparameters is well suited for this problem. This was no surprise because Ridge Regression is robust to co-linear features in the dataset. We had an intuition that Random Forest Regressor would perform well on this dataset because it's very powerful regressor that works well with tabular data, however, we didn't get the results we hoped to have achieved.

Our Ridge Regression model outperforms models tested by Kober and He [2] in terms of overall MAE. Their best model was a KNN with a 3.91 CV MAE whereas our best performing model was Ridge Regression with 3.69 CV MAE. One reason why we felt our model out performs theirs is because we take advanced statistics into consideration whereas they do not.

We would have liked to have tried to find super advanced stats from NBA.com i.e. experience, pace, pie, clutch rating, etc. These stats would have given much better insight about a player's performance throughout a season. We also felt that the simple regressors to model player statistics and information isn't enough and felt that using an advanced method such as Bayesian statistics would allow us to model the players in a correct manner. We could have encoded player information into some prior distribution which would give us some posterior distribution in which we can obtain player ratings. We expect that using more advanced models could outperform our models for 2K rating prediction.

References

[1] "Basketball Statistics and History." Basketball-Reference.com. N.p., n.d. Web. 02 May 2017.

[2] Birlutiu A., Heskes T. (2007) *Expectation Propagation for Rating Players in Sports Competitions.* In: Kok J.N., Koronacki J., Lopez de Mantaras R., Matwin S., Mladeni D., Skowron A. (eds) Knowledge Discovery in Databases: PKDD 2007. PKDD 2007. Lecture Notes in Computer Science, vol 4702. Springer, Berlin, Heidelberg

[3] Kober, Christopher. & He, Shuai. *NBA 2K Player Ratings Predictor.* NBA 2K Player Ratings Predictor. N.p., n.d. Web.

[4] Operation Sports: Dedicated To Sports Gaming. N.p., n.d. Web. 02 May 2017.

[5] Ruslan Salakhutdinov, Andriy Mnih, and Geoffrey Hinton. 2007. *Restricted Boltzmann machines for collaborative filtering.* In Proceedings of the 24th international conference on Machine learning (ICML '07), Zoubin Ghahramani (Ed.). ACM, New York, NY, USA, 791-798.

[6] Scikit-learn: Machine Learning in Python, Pedregosa et al., JMLR 12, pp. 2825-2830, 2011.