NBA Shot Prediction (’14-’15 Season)

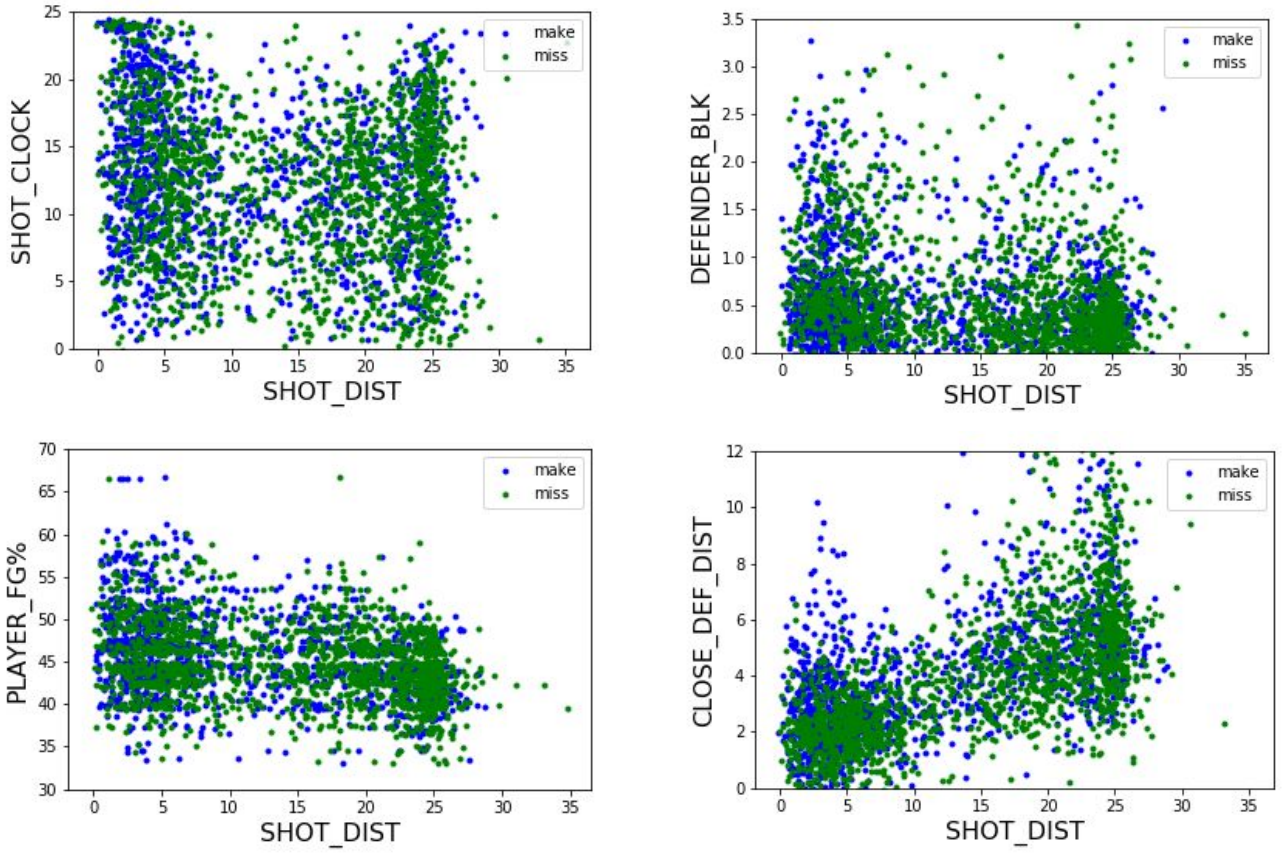
Irvin Tang & Kevin Xu

In the NBA, which team scores the most points matters the most at the end of the day, and being able to make shots is one of the most important things for both the players and the fans watching the sports. As NBA fans, we are interested to see if we can predict the outcome of a shot given data on the shot itself and the players involved with machine learning. Also, being able to predict every shot taken in the course of a game would be a really powerful analysing tool that can be applied to a large variety of aspects.

The goal of the project is to determine whether or not we can accurately predict if a shot was made. Two datasets were used for the project. The first one is data on every shot taken during the 2014-2015 NBA regular season, which included information like who took the shot, where on the floor was the shot was taken from, who was the nearest defender, how far away was the nearest defender, time on the shot clock, and much more. The second one is the full players’ stats from the 2014-2015 season plus personal details on the players such as height, weight, etc.

Data were preprocessed before creating our model. The two datasets were merged into one. The reasoning behind the merge was that we thought stats for both the shot taker and the defender of a shot can play a significant role when predicting if a shot was made or not. We encountered some problems when trying to merge the datasets. The datasets were designed to be merged by matching player names. However, the player names from the two datasets were of different formats, so the first thing we did was to convert the names into a unified first name-last name capitalized format. Then, we discovered that the names had some typos and different spellings such as abbreviations across the two datasets. The missing names were filtered out and fixed in order for the merge to be successful. Another major issue with our player stats was that some players were missing their height and weight data including some of the best players in the league. Therefore, the missing data fields cannot simply be ignored. To solve this, we manually entered every missing height and weight field with data found on www.basketball-reference.com, one of the best websites to find detailed basketball stats and references. We only selected certain features from the player stats dataset to merge into the shot log dataset because a lot of the data were irrelevant in our opinion. We included offensive stats such as points per game, field goal percentage, 3-point percentage of the shot takers, and defensive stats such as rebounds, blocks, steals, etc. for the closest defenders. Physical stats, height, and weight, were also included for both the shot taker and the defender. Data such as minutes played, age, birthday, and more were left out during the merge. The data preprocessing we have done at this point was for the early stage. Later in our analysis, more data will be excluded or manipulated to train the models.

After obtaining the preprocessed data in a single CSV file, we officially began our analysis on shot prediction. Since this kind of prediction is not very critical, we can simply use the error rate to evaluate the performance of our model. The first baseline we set for this project is to predict every shot within 5 feet of the basket as a make and anything else as a miss. Our simple baseline gave us an accuracy of 61.36% which was higher than what we expected. A second baseline we tried is to use the K-Nearest Neighbors algorithm. We ran the algorithm first without any normalization on the data, setting n_neighbors equal to 20. The accuracy of our model went up 2 percent to 63.19%. Then we normalized the data so that no one parameter exerts more influence on the result than others. However, after normalizing the data, the accuracy skyrocketed to near 100 percent (sometimes it was actually 100%). It turned out that FGM is not the total field goals made, but whether or not that individual shot was made or not, which means that the column is basically what we are trying to predict, so we had to drop FGM from our dataset. After dropping the column, the normalized data achieved an accuracy of 57.78% which was a decrease from what we had before. The accuracy for the normalized data using KNN stayed below 60% even with k increasing. We wanted to see if we could increase the baseline accuracy anymore so we created some visualizations for relationships between the features. These plots were only plotted with 3,000 samples as the size of the dataset would cause the plots to look like blobs of blue and green.



Because it wouldn't be feasible to plot every single pair of relationships of our dataset, we decided to pick four that we thought could give us some insight on any patterns that correlated with the results of a shot. For the plots, there seemed to be a higher concentration

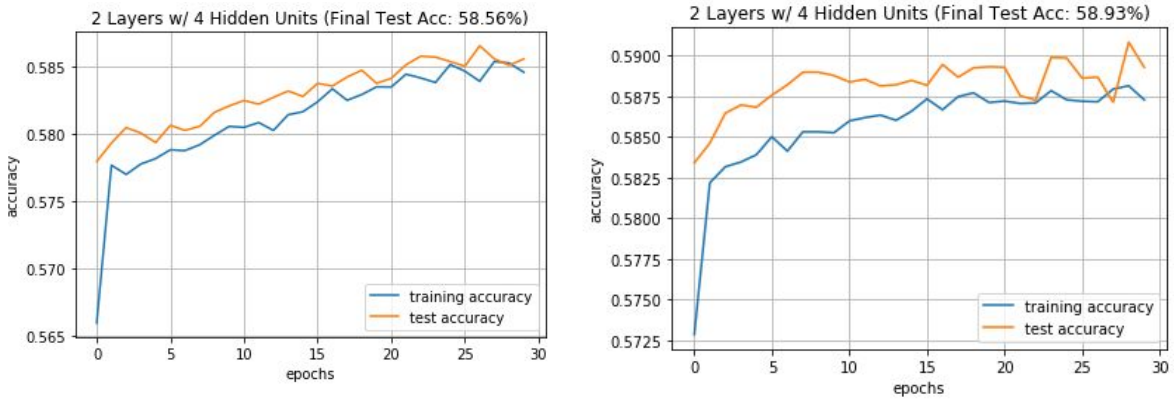
of made shots as the shot distance decreased. However, it's clear that there is no obvious trend in the data. We decided to just test out creating a new feature that represented the ratio between the shot distance and the defender's blocks per game. The intuition is that as the shot distance increases, a defender's blocks per game matters less. Using this feature transformation, our baseline accuracy increased to 60.14%. Additionally, since we had previously normalized the data, we decided to set some manual weights. The weights were not assigned using any particular scientific method. Rather, it was purely based on what we as basketball fans felt about the influence a particular feature had on the result of a shot. We assigned higher weights to Player PPG, Player FG%, Player 3P%, Closest Defender Distance, and Shot Distance. Running KKN on this newly weighted data, we achieved a 0.13%+ accuracy for total accuracy of 64.27%. The increase was minuscule, but the improvement was encouraging.

Seeing as how our manual weight assignments improved the baseline accuracy, even if it was minuscule, we felt that this was one of the keys to improving on our baseline. Instead of manually assigning weights to features, we decided to create a basic two-layer neural network with sigmoid activations so that the weights could be learned automatically. We tested neural networks with a different amount of hidden units.

The best accuracy we achieved was 62.36% using a two-layer neural network with sigmoid activations and 12 hidden units per layer and 30 epochs. We then decided to take this model and see if we could improve on it further by tuning some parameters. We performed cross-validation on various values for epoch, batch size as well as different kinds of weight initialization techniques. However, the best parameters that the grid search gave us were the exact same ones as the neural net that gave us 62.36% accuracy.

Feature selection and engineering seemed to have given us improvements. The neural nets we trained were able to learn some weights that allowed us to increase accuracy by 2%. However, it still wasn't that great of an increase. We decided that it would be a good idea to see if a neural net without irrelevant features could possibly improve on what our current neural net had. We used Adaptive Boosting just as a quick tool to show us which features had

the most influence. We then decided on an arbitrary threshold that the features' scores should cross in order for us to use them. We used two thresholds, one where the features' "score" had to be at least 0.05 and another where it had to be at least 0.03. Ultimately, neither neural network gave us better accuracy than what we had from our neural network trained on all the features we had initially selected.



7 features were used on the left model and 13 features were used on the right model

In conclusion, we find that problems of this nature are generally hard to predict since there are too many factors in sports that can impact the outcome of a shot or a game. We could maybe get close to 70 percent accuracy given more data and a better model, but that is about as high as we can achieve. So, if we were to continue our analysis in the future, we would look at the problem in a different way and explore the data in a different way. Instead of straightforwardly predicting the outcome of a shot, which by itself is not all that meaningful, we can ask some other questions such as does the player's shots follow a certain pattern, what could the player have done to improve his shot selection, can we predict the player's next shot given information on his last shot, etc. By asking these alternative questions, we can gain interesting but also applicable insights that could possibly be used towards managing the team, coaching, players' in-game decisions, and more.

Additionally, if we had more time, we would have maybe explored using convolutional neural networks since an advantage of CNNs is the feature engineering baked into the network. We also feel that further research into parameter tuning for neural networks would have proved fruitful in terms of gaining even one or two more percentage points in the accuracy of our model. Regarding our work with the baselines, one mistake we made was misinterpreting the data. I (Irvin) feel that if I had not mixed up the data on the plots, we could have maybe done some further feature transformations manually that could have matched the accuracy of our neural networks. Finally, with this kind of data, it seems that more is better. If it were feasible, we felt that including information on team statistics and specific player matchups would help boost the accuracy of our model since these are all relevant factors that influence the performance of any player.