Predicting the outcome of a basketball shot

Haitao Liu[a,1], Yang Liu [a,2], Jiawei Xue [a,3]
Instructor: Amir H Gandomi [a]

[a] Multivariate Data Analysis, 2018Fall
[1] hliu65@stevens.edu, (201)-616-8719
[2] yliu249@stevens.edu, (201)-985-4435
[3] jxue4@stevens.edu, (201)-985-4433

ARTICLE INFO

Keywords:
Classification
Predictive analysis
Multivariate analysis
Principal components analysis
Machine learning
Ensemble
XGBoost

ABSTRCT

A single basketball shot has two outcomes: made or missed. One of the methods that is used to predict the shooting result is the machine learning model. In this research, we intended to use this method to predict a basketball shot. The main work we did is as follows. First, we introduced the background of our research and came up with the research question. Second, we used 2014-2015 NBA season shot records as our dataset and evaluated the dataset. Third, we did six steps for data processing and preparation: One hot encoding, handling missing values, detecting the outliers, identifying the relationships between variables, standardizing the data and splitting the dataset. Then, we attempted to use principal components analysis to reduce dimension. In the next step, we built six models based on linear discriminant analysis, logistic regression, K nearest neighbors, naïve Bayes, XGBoost and ensemble algorithm. The model results indicated that the ensemble method showed a great enhancement over other simple algorithms, and the XGBoost had the best prediction performance.

1. Introduction

Basketball is one of the most popular sports in the world, especially NBA games. Different players enter the court, shoot the ball, and then score or not. One player may become a star and attract many fans if he is good at scoring. However, almost every player, whether he is an ordinary one or a star, will miss some shots during each game, even when they have "hot hands". This phenomenon happens every day and is very familiar to us. The fans are so interested in the shots that they will not be satisfied with just knowing which team is the winner, they buy tickets or turn on the TV to see whether a shot is made or missed.

As so many people spend money, time and energy to watch each play in game and the teams put effort to score more, a question arises. Can the outcome of a shot be predicted? In order to answer this question, we are going to build classification models using selected metrics to predict the shot result. A single basketball game contains many metrics that can be used to predict whether a shot is made. In this paper, first, we will identify the relationships between different features, and then based on that, we will build six classification models (linear discriminant analysis, logistic regression, K-nearest neighbors, naïve Bayes, xgboost and ensemble algorithm) to find out the most efficient prediction method.

This paper is structured as follows. The next section introduces the evolution of NBA and machine learning technology. Following that, we will evaluate our dataset. Then we will discuss data processing and preparation. Next, we will perform the principal component analysis for the model preparation. After that, we will build different classification models and compare the results. Finally, we will discuss the conclusion and future research.

2. Problem description

As we all know, different players have different shooting preferences. For example, some players like shooting after dribbling a lot, some players like just catching and shooting, some players like long-distance shots, and some players likes difficult shots, etc. But which one is best? What are the key factors that determine whether a shot is made or missed? If we figure out what key factors that affect the result of the shot are, we can use them to predict whether a shot is going to made or missed. The outcome of this prediction is, for the fans, that they can watch the game in a more rational way, not only based on intuition. On the other side, coaches and specialists can rely on the prediction model, instead of relying on judgment and hard stats, to support their decision-making process, such as how to make game strategies and how to train their players.

3. Evaluation of dataset

We retrieved our dataset from www.kaggle.com, the most famous data science competition website. The dataset contains all the records of all shots in the 2014-2015 NBA season. It has about 128,069 instances and each instance represent the record of one single shot from one player in one game. The reason that we chose to use the data from the 2014-2015 season instead of the latest is that we can verify our conclusion based on the classification model we create if we use the latest season's shots as new inputs. In this original dataset, there are a total of 21 columns that are the metrics of each game. Considering the fact that some of the metrics are obviously not the factors that affect the outcome of a shot, we deliberately selected nine features as the input variables and one feature as the output variable.

The input variables are the location (place where a game was held, indicates home or away), shot number (the number of a shot in order that one player take in a game, indicates equal or greater than one), period (the number of each quarter in a game when a shot is taken, indicates from first to fourth, with a few five to seventh, which are the overtime), game clock (time passed in each period, indicates from 00:00 to 12:00), shot clock (the remaining time of each offense time, indicates from 24 seconds to 0 seconds), dribbles (number of dribbles from catching the ball to shooting, indicates greater than 0), touch time (the time passed from catching to shooting, indicates greater than 0), shot distance(the distance from shooting position to the rim, indicates from 0 to 94 feet), and the closest defender distance(distance from shooter to defender, indicates from 0 to 94 feet). The output variable is FGM, "field goal made" (the outcome of a shot, which is made or missed, indicates 1 or 0).

There is another feature in our dataset, the player name (or player ID), which we selected as input variable at first, but then we chose to remove it. There are two reasons that support this decision.

Firstly, it is impossible to predict a new observation when this new observation has not already existed in our training set and testing set. The fact that a classification model contains players does not make sense. Just imagine if the final classification model had one variable called players' name (or players' ID) and its coefficient. It is not easy to interpret this. We cannot know how to use this feature to make any recommendation. Since the players' name is a nominal variable, even though players' ID is numeric, it will have no meaning in the classification model. Furthermore, because each player is unique, when we are going to predict a new player, for example, a rookie that enters the league in the next season, this player cannot be predicted because it is out of range of original player names that in the classification model.

Secondly, our goal is to get a general model that can be used for every player. We do not want an individual player to affect our prediction model, but we do want our prediction model to affect the players. As we know, different players have high variance in shooting performance. Using the player as an input variable would have a significant bias in the final classification model. What's more, the players nowadays are quite different from the players years ago. Nowadays, the players are smaller and faster, and even a center can shoot a 3-point. It's not because they can shoot a 3-point at the beginning of their career, but because the teams know a 3-point is more important than before and now the centers are required to practice 3-point shooting. Maybe this rule does not apply to some stars, but it does apply to ordinary players. We aim to find the general rule of the outcome of the shot so that we can make suggestions to the teams to make better game strategies and train their players more efficiently and correctly. In conclusion, using the players as an input variable is in conflict with our research goal, so we chose not to use it.

After variable selection, we moved to the next section, data processing and preparation.

4. Data processing and preparation

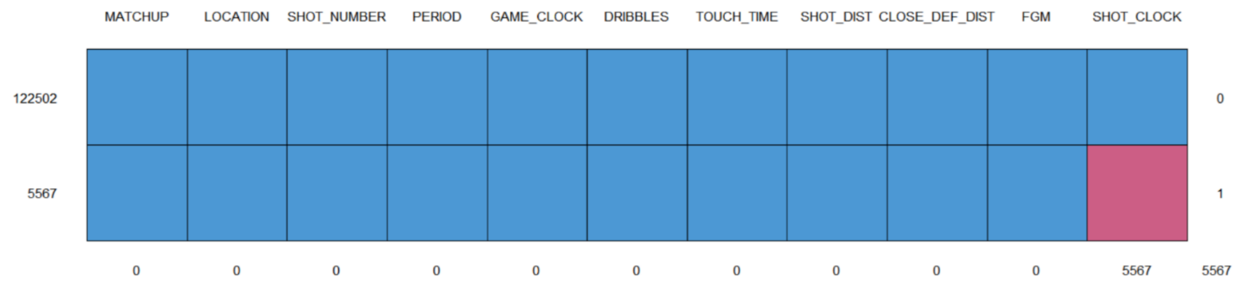
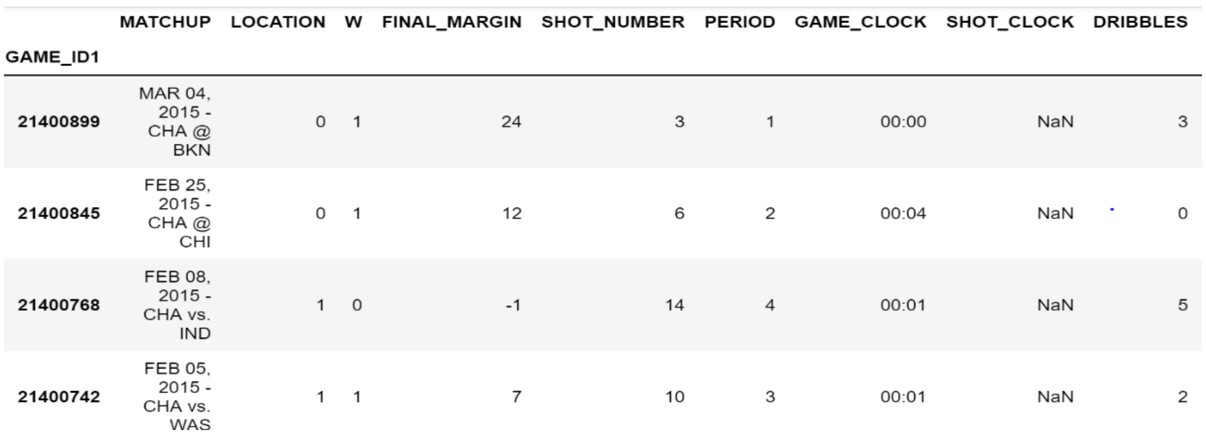


Figure 1 . The red cells represent missing values and the blue cells represent no missing values.



GAME_ID1	MATCHUP	LOCATION	W	FINAL_MARGIN	SHOT_NUMBER	PERIOD	GAME_CLOCK	SHOT_CLOCK	DRIBBLES
21400899	MAR 04, 2015 - CHA @ BKN	0	1	24	3	1	00:00	NaN	3
21400845	FEB 25, 2015 - CHA @ CHI	0	1	12	6	2	00:04	NaN	0
21400768	FEB 08, 2015 - CHA vs. IND	1	0	-1	14	4	00:01	NaN	5
21400742	FEB 05, 2015 - CHA vs. WAS	1	1	7	10	3	00:01	NaN	2

Figure 2 The missing values in shot clock are displayed as NaN.

4.1. One hot encoding

The first thing we did was checking whether each variable that we have selected was a categorical feature. It turned out the variable location is a nominal variable with two levels, home or away. In order to make it easier to analyze, we convert the location into a dummy variable-1 represents home and 0 represents away.

4.2. Handling missing values

Most of our work before building machine learning models started from identifying the missing values, the main component of the dirty data, which usually cause inaccurate models or difficult analyzing. Therefore, we check the whole dataset to identify the missing values. Unfortunately, there are 5567 rows that had missing values in the shock clock. (Figure 1)

Typically, there are two ways to handle missing values: imputation or removing data. But before we jumped to the methods of data imputation, we had to understand the reason why data goes missing.

(a)Missing at random (MAR): Missing at random means that the propensity for a data point to be missing is not related to the missing data, but it is related to some of the observed data.

(b)Missing Completely at Random (MCAR): The fact that a certain value is missing has nothing to do with its hypothetical value and with the values of other variables.

(c)Missing not at Random (MNAR): Two possible reasons are that the missing value depends on the hypothetical value (e.g. People with high salaries generally do not want to reveal their incomes in surveys) or missing value is dependent on some other variable's value (e.g. Assuming that females generally don't want to reveal their ages; here the missing value in the age variable is impacted by gender variable).

In the first two cases, it is safe to remove the data with missing values depending upon their occurrences, while in the third case removing observations with missing values can produce a bias in the model. Therefore, we need to be careful before removing observations. Note that imputation does not necessarily give better results.

After that, we needed to take a further a look at the missing value to see which type of missing value it belonged to.

From Figure 2, we can see that when there is a missing value of a shot clock, the game clock of that row is 0 or nearly 0, which means the shot is made at

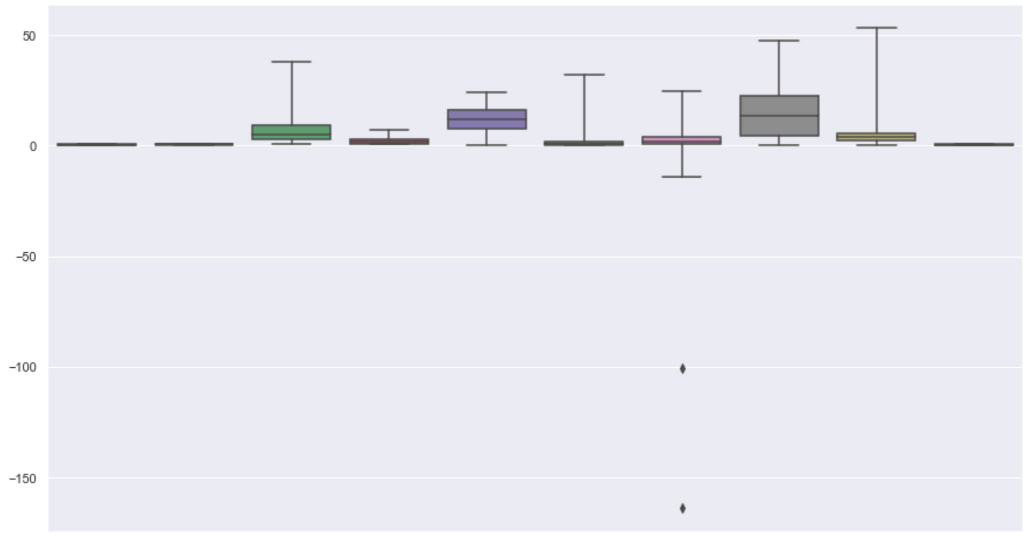


Figure 3. Two marks in the "Touch Time" are detected as outliers by box plot.

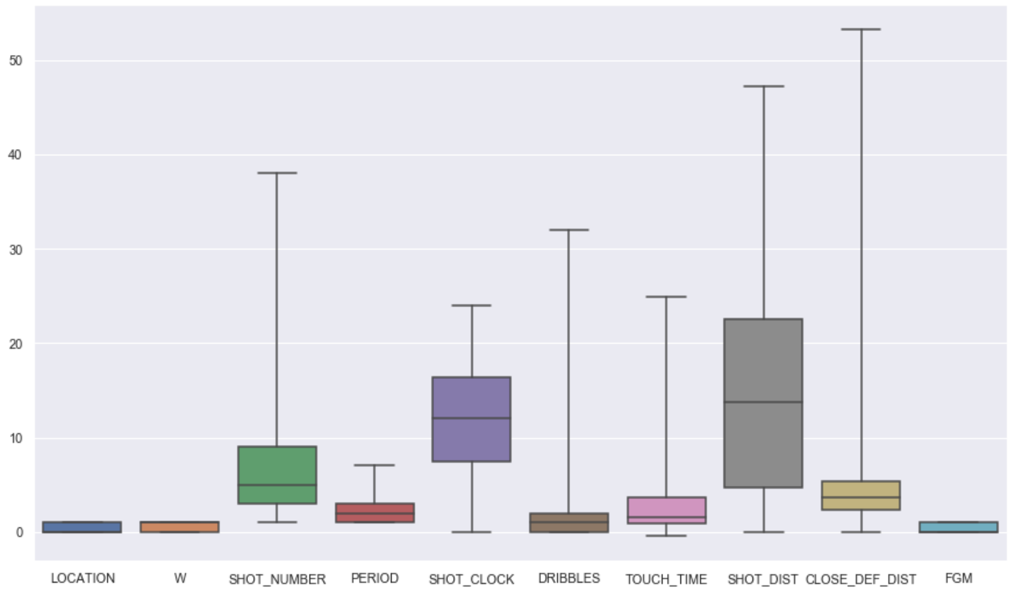


Figure 4. Box plot with no outliers after replacing outliers with the mean of "Touch Time"

the end of each period. Then we realized that the missing value exist is because no offense time left after that shot, what we usually called the buzzer beater. This is easy to understand because, at the end of each quarter, the games scores are often very close. The last seconds are so essential that everyone gets tense, including the recording staff. They may become too busy and nervous to record all the data at that time, and thus, cause them to miss recording the shot clock. Therefore, the missing values in shot clock belong to MNAR, not random missing values. Based on that, we decided to do imputation instead of just removing the missing values. Furthermore, as we discussed above, we figured out that a missing value represents zero time left after a shot, so we chose to replace the missing values with 0 instead of the mean of shot clock, which is another highly used method.

4.3. Detecting the outliers

An outlier is an observation point that is distant from other observations. Many classification algorithms are sensitive to the range and distribution of attribute values in the input data. Outliers in the input data can skew and mislead the training process of machine learning algorithms and thus result in longer training time, less accurate predict models and ultimately poorer results.

One of the simplest methods for detecting outliers is the use of box plots. A box plot is a graphical display for describing the distribution of the data. Box plots use the median and the lower and upper quartiles. We generated the box plots for both independent variables and dependent variable. (See Figure 3).

We carefully examined these outliers. Three hundred twelve data existed in these two areas. We found that the reason why they appeared here was that they all had negative values. Typically, the "TOUCH TIME" couldn't be negative. There must be some record errors or other random errors in these data.

At this point, a decision that whether we would alter the outliers with the mean of that variable or discard them should be made. Because both methods have some drawbacks. Replacing the outliers with the mean of other data in that variable would reduce the variance of that variable, which might cause underfitting, while simply removing the outliers would lose the precious data information in other variables. After deliberate discussion, instead of removing these values, we decided to retain these observations and replace them with the mean of "TOUCH TIME". There are two reasons supported our decision. First, we preferred to analyze the complete data set. Data is the "oil" of machine learning models, and the price of losing extra information would be huge if we discarded the instances just because one single variable contained outliers. There would be no other methods to compensate it if we discarded the outliers. Second, compared to the whole data set, the proportion of outliers was tiny, just 312 over 128069.

4.4. Identifying relationships between variables

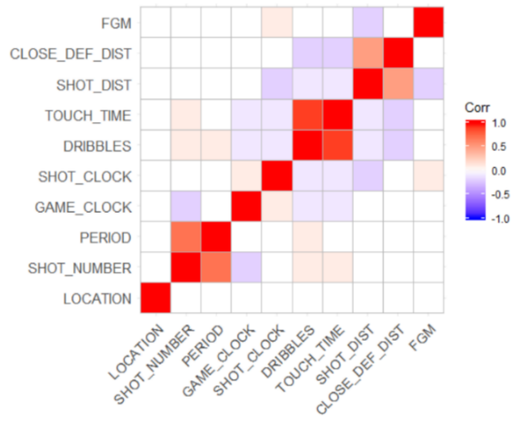


Figure 5. Correlation matrix between variables.

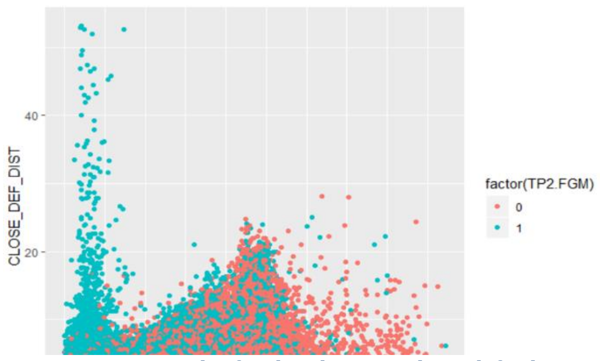


Figure 6. Scatter plot for shot distance, closet defender distance and FGM.

In general, the correlation between different variables should be checked because the correlated features will make the learning algorithm slower, increase harmful bias and decrease the interpretability of the model. Therefore, we plotted the correlation matrix to see the relationships between variables (Figure 5). This correlation matrix showed us three pairs of variables with higher correlation. The first pair was PERIOD and SHOT_NUMBER. This pair is easy to understand because the SHOT_NUMBER is accumulative in a game. That is, if the number of the period increased, the shout number would increase. The second pair is TOUCH_TIME and DRIBBLES. Normally, a longer touch time enables a player to dribble more times. So, this pair also makes sense. It's a little tricky to understand the third pair—CLOSE_DEF_DIST and SHOT_DIST. So, we plotted another scatter plot for SHOT_DIST, CLOSE_DEF_DIST and FGM to take a further look at the relationships between these three variables. In Figure 6, the blue dot means that a shot was made, and the red dot means that a shot is missed. We noticed that in some cases the CLOSE_DEF_DIST and SHOT_DIST were correlated, but some cases are not. We could also find out an interesting relationship between these two variables. When the SHOT_DIST is less than 10, which means that shot distance was very close to the rim, the FGM almost all equaled to 1. However, the values of CLOSE_DEF_DIST varied a lot. Why did this happen? This happened may because most of the shot in this area were made in transitions, the fast breaks. That is, the closest defender is far behind the shooter. So, what does this mean to us? We should make as many fast breaks as possible to get easy points. This is also verified by today's basketball trend.

Typically, we should select fewer variables when some variables have a correlation with other variables because the correlated variables do not offer extra useful information but bring some drawbacks, for example, overfitting. But considering that our dataset just consisted a few variables and the problem of overfitting of prediction models may arise due to fewer variables, so we decided to remove any variables. We decided to use all the variables we selected at first to train our models.

4.5. Standardizing the data

The concept of standardization comes into picture when continuous independent variables are measured at different scales. With different scales, it would be difficult to measure each variable's contribution to analyzing and modeling. By standardizing the data, the variables would at the same scale and highly decrease the bias to models. The standardized method is to divide the difference between a observed value and its mean by its standard deviation. The idea is to make the variables have equal variance, but different means and ranges.

4.6. Splitting the dataset

Typically, the dataset will be split into two parts, a training dataset and a testing dataset. The training dataset is used to fit the model, while the testing dataset is used to provide an unbiased evaluation of a final model fit. Considering that our dataset may not large enough to supply substantial data to our model for training, we would optimize the split proportion for a larger training set. Therefore, we decided to split our dataset into the training set and testing set at a ratio of 4:1.

So far, we have discussed the data processing and model preparation part. In the next section, we will our models in detail.

5. Dimension Reduction

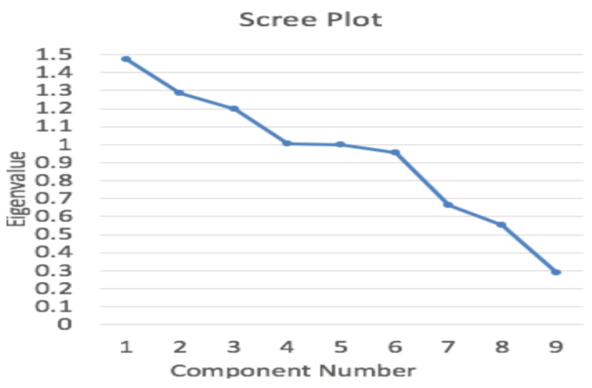


Figure 7 Scree plot for PCA

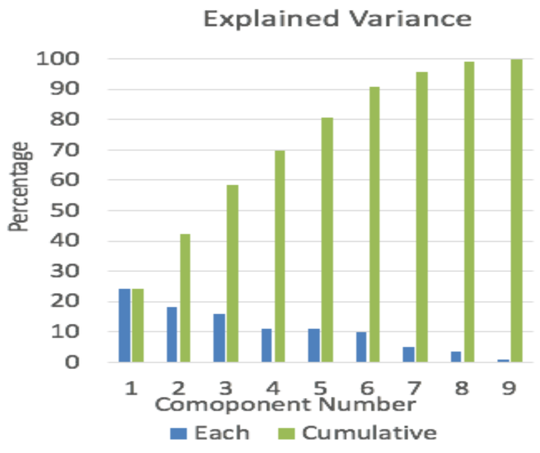


Figure 8 Explained variance plot for PCA

We wished to reduce the number of variables without losing much of the information. So, we conducted principal component analysis (PCA) with nine categories to achieve this goal. PCA is a statistical procedure that uses an orthogonal transformation to convert possibly correlated variables into a set of uncorrelated variables. In other words, PCA is a linear combination of original variables. In order to get the principal component, first, we calculated the correlation matrix of the original data first. Then calculate the eigenvalues of the correlation matrix and their corresponding eigenvectors. Finally, we got a principal component by the value of an eigenvector.

In Figure 7, the scree plot, nine principal components were plotted via the scree plot in decreasing order of variance.

In Figure 8, the blue columns represented the percentage of total variance explained by each principal component. Adding these percentages successively produced the cumulative percentages plotted in green columns.

When determining the retained number of components, at first, we adopted Kaiser's rule. That is, we selected the first four components whose eigenvalue were greater than one. These four components could explain nearly 69% of the total variance. However, we were not satisfied with this percentage. Therefore, we selected the first six components, which explained 90% percent of the total variance. As seen in Explained Variance Plot, the 6th point was the cutoff point where the two slopes met.

We applied the PCA to our four models (linear discriminant analysis, logistic regression, k nearest neighbors, naïve Bayes), which we will illustrate in the Methods section. Unfortunately, we found out that PCA had no positive impact on modeling after we performed, so we decided not to use PCA for modeling.

6. Methods and Results

As our goal was to build the model to predict whether a shot would be made or missed, a two-class classification, so we used six methods (linear discriminant analysis, logistic regression, k nearest neighbors, naïve Bayes, xgBoost, and ensemble) to perform the classification on our data. We will explain how it works and the result of each classification method.

6.1. Linear discriminate Analysis

Linear discriminate analysis is a simple method that can solve 2-class classification problems. The idea of this method is to maximize the difference between the means of each group and minimize the variation in each group. We chose the Euclidean distance to measure the distance of the predictors from a group centroid.

The fisher discriminate function is a combination of the predictors to predict group. The scaled estimates of discriminate function are shown in Figure 9.

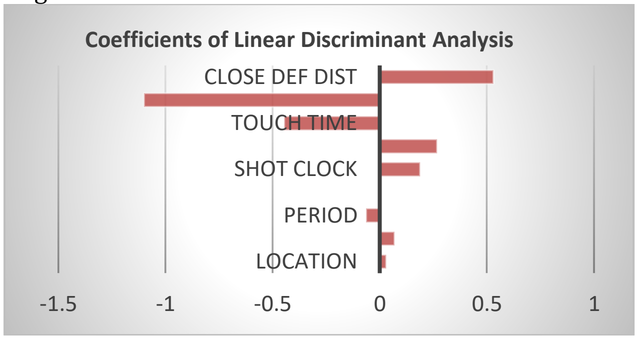


Figure 9 Linear discriminant analysis prediction result

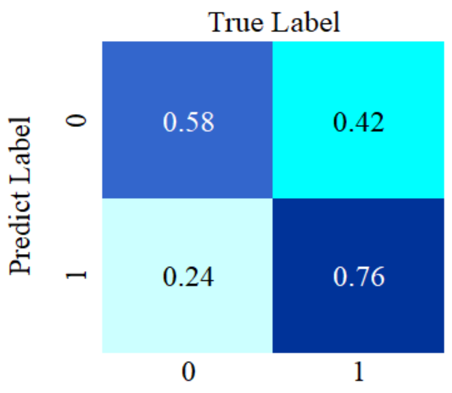


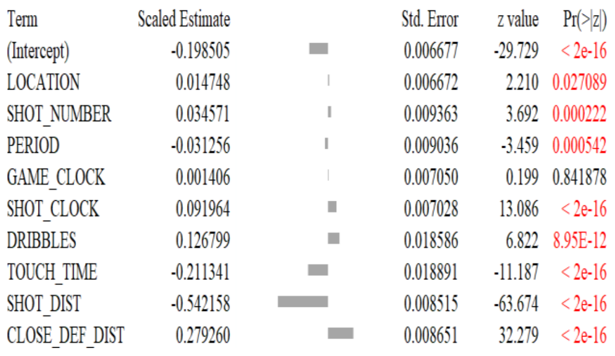
Figure 10 Confusion matrix of the LDA prediction

It can be seen from Figure 9 that SHOT DIST and CLOST DEF DIST have the largest impact on the shooting results. The performance of LDA made on the testing data is shown in Figure 10.

6.2. Logistic regression

Logistic regression is a method to explain the relationship between binary dependent variable and one or more independent variables. This method provides a probabilistic result so that we can have a better understanding of classification result predicted by our model.

Figure 11 shows the coefficients of the logistic regression function, and Figure 12 presents the confusion matrix of the result. The accuracy of the model is only 0.59 according to our result.



Term	Scaled Estimate	Std. Error	z value	Pr(>|z|)
(Intercept)	-0.198505	0.006677	-29.729	< 2e-16
LOCATION	0.014748	0.006672	2.210	0.027089
SHOT_NUMBER	0.034571	0.009363	3.692	0.000222
PERIOD	-0.031256	0.009036	-3.459	0.000542
GAME_CLOCK	0.001406	0.007050	0.199	0.841878
SHOT_CLOCK	0.091964	0.007028	13.086	< 2e-16
DRIBBLES	0.126799	0.018586	6.822	8.95E-12
TOUCH_TIME	-0.211341	0.018891	-11.187	< 2e-16
SHOT_DIST	-0.542158	0.008515	-63.674	< 2e-16
CLOSE_DEF_DIST	0.279260	0.008651	32.279	< 2e-16

Figure 11 Logistic regression prediction model

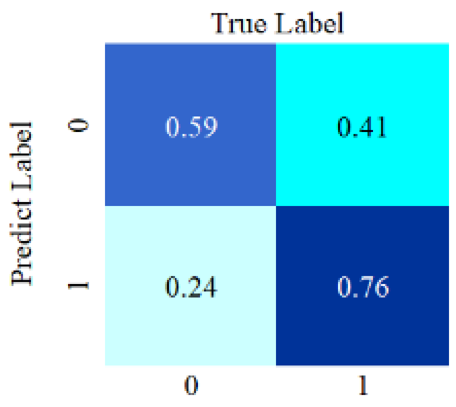


Figure 12 Confusion matrix of logistic regression.

6.3. KNN

The intention of KNN method is that a predictor is classified according to the majority vote of its k nearest neighbor. In this method, we chose Euclidean distance to calculate between each predictor. In order to find the optimal k value, we performed several times modeling training and testing. Figure 13 shows the relationships between the values of k and the corresponding accuracy. We can see that this model has a really low ability of prediction and the accuracy of model still remains in a low range.

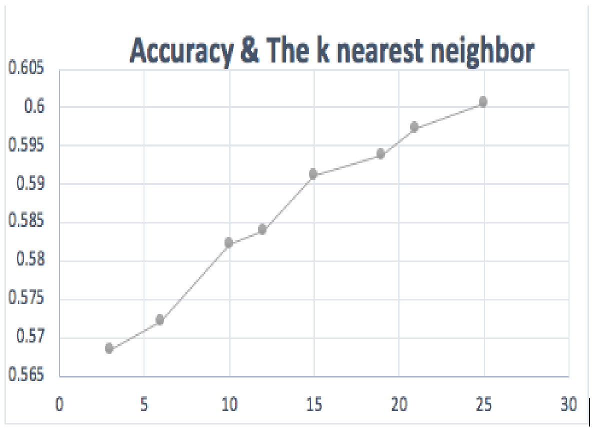


Figure 13 K nearest neighbor prediction result.

6.4. Naïve Bayes

Naïve Bayes is based on Bayes Rule of conditional probability and every pair of features being classified is independent of each other. The assumption of this method is that features are independent of each other and the attributes contribute equally to the outcome.

The accuracy of this model is only 0.594. Figure 14 is the corresponding ROC curve of the outcome.

6.5. Extreme Gradient Boosting

As the four models we had built have low accuracy, we wanted to enhance our model performance. We decided to use the extreme gradient boosting

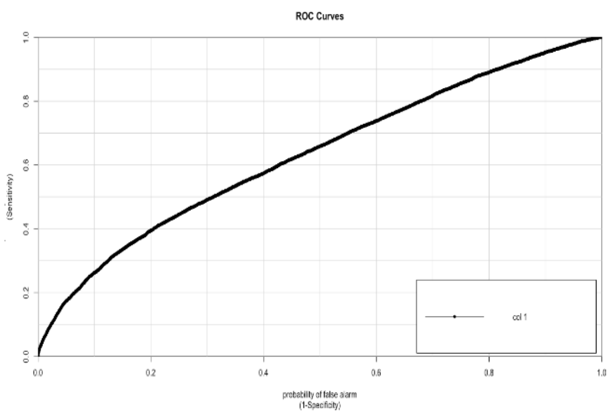


Figure 14 Receiver operating characteristic (ROC) cure for the prediction result of naïve Bayes

algorithm. The extreme gradient boosting (xgBoost) is similar to the gradient boosting framework, but xgBoost is more efficient, at least 10 times faster than the existing gradient boosting implementations. We chose the tree-based booster and logistic regression learner for this model, which can have a better effect on the result. The maximum depth of a tree in this model is 6, and the learning rate is 100%, while the max number of the boosting iteration is twice.

The importance matrix in Figure 15 shows the first six important variables in the model, and we can see that SHOT_DIST and CLOSE_DEF_DIST are the most important variables in this model. Figure16 is the confusion matrix of the testing result. From that figure, we could calculate the accuracy of this model equals to 63.13%.

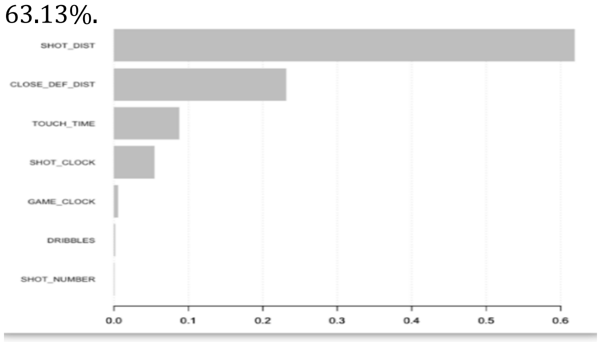


Figure 15 Importance matrix of xgBoost

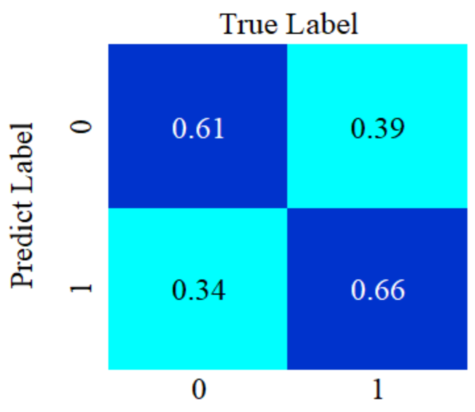


Figure 16 Confusion matrix of xgBoost prediction.

6.6. Ensemble

The last model we created is stacking ensemble. Stacking ensemble is an ensemble method where the models are combined using another data mining technique. We built two layers model to perform stacking ensemble on the testing data. The bottom layer models which received the original input features from the original dataset and train data based on these features. Then we performed k-fold cross-validation using training data for these bottom layer models and got the predicted probabilities based on cross-validated results. Next, the top layer model took advantage of prediction results from the bottom layer models to do a further prediction. That is, the top layer model took the output of the bottom layer models as its input and predicted the final output.

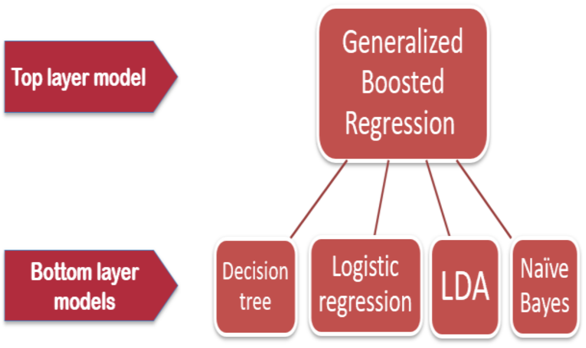


Figure 17 Ensemble

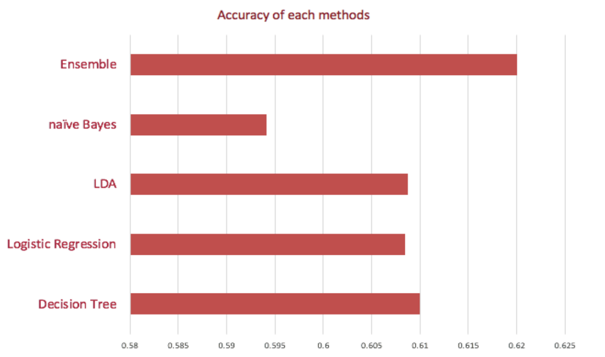


Figure 18 The prediction accuracy of ensemble and its layer models.

Figure 17 shows the methods that we chose for each layer model and Figure 18 presents the accuracy of each model. Except for the three models (LDA, logistic regression, naïve Bayes) we discussed previously, we added another model-decision tree to bottom layer models to get more inputs for the top layer model. From Figure 18 we found that ensemble did improve some. At the same time, decision tree had the best performance when we ran the bottom models separately at the beginning of the process.

7. Conclusion

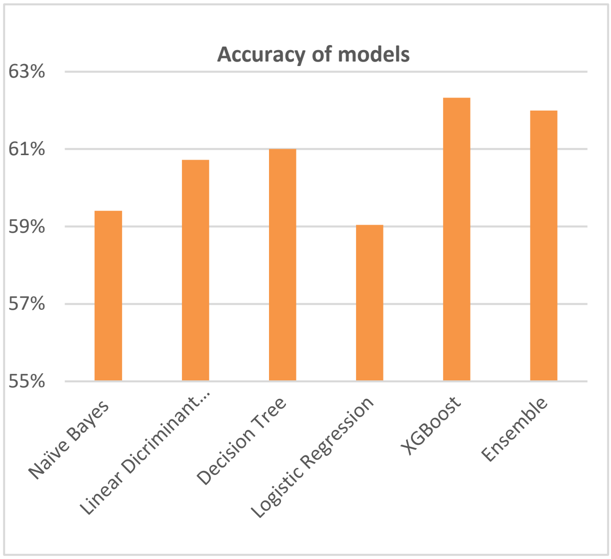


Figure 19 Comparisons of models' accuracy.

We used different kinds of methods to predict the outcome of a basketball shot, and it turns out that the XGBoost perform the highest accuracy which is 63.13% (Figure 19). After we performed the stacking ensemble, the accuracy is 62%, the higher than the accuracy of the bottom layer models. The models we used are not ideal enough but these models have a high potential to enhance their accuracy.

8. Future research

The accuracy of our models is not ideal enough, however, at the same time, have a great potential to enhance. The low accuracy may probably cause by small dataset, and, the limitation of few features. The basketball is a very complex teamwork, even though we just considered the shooting, there are also too many dimensions of data to measure one single, such as the shooting angle, shooting speed, type of shooting, the exact coordinate of the court, etc. In the future, we are going to directly retrieve the data from nba.com or basketball reference.com in order to add more features to our model. If this methodology still has little improvement on our models, we will consider using the game video, instead of CSV file, as the input data, build deep learning models using computer vision. That will be a long way to go.

Acknowledgments

We would like to express our special thanks to our professor Amir H Gandomi's for his great classes and guidance, especially for his high standard requirement for our presentation performance and the project paper.

Secondly, we would like to thank Haochen Liu, the teaching assistant, who helped us a lot in this project.

References

Abdelmonem, A., Susanne, M., & Virginia, A. C. (2012). Practical Multivariate Analysis. Fifth Edition

Paul, N., William, L C. & Betty, M. T. (2013). Statistics for Business and Economics

Zhihua, Z. (2016). Machine learning.

Manish Pathak, May 22nd, 2018, [online], available: https://www.datacamp.com/community/tutorials/categorical-data