Kobe Bryant Shot Prediction

Yi Zhang, Chia-Ju Chen, Minke Cheng, Meng Lin
Department of Electrical and Computer Engineering
The University of Texas at Austin
Email: yi.zhang.cn@utexas.edu

Abstract

The data mining revolution has recently come into basketball in a big way. In this work, we are interested in a 20-year shooting record of Kobe Bryant provided by a Kaggle competition and are motivated to build a model that is able to predict his shooting result with some associated information. First of all, we study Kobe's 20-year shooting records via advanced data visualization methods, based on which the feature selection is performed with the help of our domain knowledge in basketball. Afterwards, several practical feature transformation methods are proposed in this work to deal with categorical features and to reveal hidden information from existing features as well. Particularly, an accuracy-based feature clustering method is proposed to realize a more accurate partition of the basketball court and also to help reduce the dimensionality of categorical features.

Several classic classification models are presented, applied and compared, which consists of Logistic Regression, Decision Tree, Random Forest, Gradient Descent Boosting Tree, and XGBoost. It is shown that XGBoost outperforms other classifiers for our studied case by achieving a ROC score of 0.7167 and a Log Loss score of 0.5988, which ranks us top 17 among 1117 teams in this Kaggle competition. Beyond these, a more challenging prediction by considering the chronological order of the shooting records is further attempted and some interesting results are shown. Our work not only provides insights into the on-court performance of Kobe Bryant but also highlights some potential future directions for predictions about basketball players.

Index Terms

Basketball, shot prediction, Kobe Bryant, classification, feature transformation, XGBoost.

I. Introduction

A. Background

Not only a popular recreation, basketball is an on-going multi-billion dollar business. The data mining revolution has accordingly come into basketball and been engaging in its related business and recreation. With the new methodologies that have been developed for analyzing the statistic of basketball teams and players, modern data mining approaches help us to arrive at a deeper understanding of basketball games and players.

When it comes to basketball, the National Basketball Association (NBA) is unquestionably the most prestigious league in the world and its super stars are always attracting people's eyes. Data mining technique has been involved recently in NBA business. For example, the NBA draft in each summer is notorious for making the careers of general managers, dashing the hopes of fans, and creating future stars of the league. In [1], the NBA Draft has been dissected by data visualization methods and a neural network has been even implemented to predict how much a college player would contribute in his potential NBA career. In addition, some data scientists have developed a probabilistic forecast of what a current NBA players future might look like by identifying his similar players throughout NBA history [2]. In [3] and [4], some machine learning algorithms have been employed to predict the outcome of a game.

For the last two decades, the Los Angeles Laker shooting guard, Kobe Bryant, has been touted as the guy to take up the mantle of "best in the game". For almost twenty years, he lived up to the billing by achieving a 5-time NBA champion, 18-time All-Star and league MVP. Therefore, it is of great interest to deeply understand this legend by studying his on-court performance, i.e. the shooting records.

B. Investigated Problem

Motivated by the above observations, in this project, we are interested in Kobe's amazing shooting and are motivated to build a model that can predict the result of his single shot by leveraging the associated information. The organization of this report is summarized as below:

In Sec. II-A, the source and some basic information about the dataset of shots attempted by Kobe Bryant are provided. In Sec. II-B, our study firstly analyzes the dataset by visualizing the key parts of it. Afterwards, the feature engineering is performed with the help of our domain knowledge in basketball as well as the visualization results. In Sec. II-C, several feature transformation methods are proposed in this work to deal with certain categorical features as well as to reveal some hidden information from the

existing features. In Sec. II-D and II-E, the final predictors are summarized and the performance results of several classic classification models are presented and compared, which consists of Logistic Regression, Decision Tree, Random Forest, Gradient Descent Boosting Tree, and XGBoost. In Sec. III, we extend our work to a more challenging scenario, where the impact of the chronological order of shots is taken into consideration. Some primitive attempts are conducted alongside with several interesting results shown. In Sec. IV, we summarize the lessons learnt in this project and further provide some potential future directions of this work. In Sec. V, the conclusions are drawn.

Through this project, we hope to gain insights into Kobe Bryant's on-court performance, and to provide some guidances in using rigorous statistical methods for conducting predictions on basketball players or even for some other sports. The whole work to be presented in the following sections is reproducible with the public codes in our Github: https://github.com/Yimarc/Kobe_Shot_Prediction.

II. Empirical Results

In this section, the major work of the project is presented. We firstly introduce the source and background of the dataset on Kobe Bryant' shooting history. Then, a preliminary analysis of the dataset is presented via data visualization methods. Accordingly, some key informative features are selected or transformed to compose the final predictors for our predictive models. Finally, different classic classifiers are presented alongside with the comparisons of their performance.

A. Data Description

1) Data Source: After Kobe Bryant's career came to an end in April 2016, Kaggle released a dataset charting every single shot he took in his 20-year career by specifying the shooting result and the associated information of every field goal attempted by Kobe Bryant [5]. Therefore, this project is associated with a data mining competition in Kaggle.

2) Data Size: In the provided dataset, there are totally 30697 data units, i.e. the 30697 field goal attempted by Kobe Bryant. Each data unit consists of 24 features including the shooting position in the court, the type of shooting action token, the temporal information of the shot as well as some other associated information. The full list of the 24 features is given alongside with their explanations in Table I. The target variable is the shooting result, which is denoted by a binary number where 1 is for success and 0 is for failure.

3) Evaluation Metric: For the purpose of evaluation, Kaggle has removed the shooting result of 5000 data units, which is termed as test data in this report. The known 25697 data units would be denoted as the training data. In order to evaluate the performance of our models correctly, we acquire the complete shooting data of Kobe Byrant from the official website of NBA [6], of which the codes are also provided in our public Github mentioned before. In this work, the target is not limited to predict the binary value of the shooting result. We are expected to predict the probability of the event that a shot was successful. In this Kaggle competition, the evaluating metric is Log Loss, which would be further explained in the Sec. II-E.

4) Highlighted Challenge: Before exploring the dataset in Sec. II-B, the challenges in this studied problem are highlighted. Actually, the setting of this prediction problem seems to be easy at the first glance since it well fits an ordinary prediction problem. However, this does not simply imply that a perfect solution could be easily obtained. Some potential challenges and difficulties are summarized as below:

a. First of all, even though there are 24 features in the dataset, most of them are categorical features, which impedes us from directly applying some classic training models. Furthermore, there are multiple methods to convert categorical features into numerical ones, which poses us a question that which method would be the best one.
b. Secondly, we are somehow constrained by the size of the data and the number of features as well. The overall dataset is relatively small so that the distribution of Kobes shooting may not be accurately estimated. In addition, the number of distinct features is relatively small, which constraints us to devote lots of efforts in feature engineering. There are too many uncertainties in predicting the shooting result with limited information.
c. Thirdly, high correlations actually exist among features since many features are just categorical variables deriving from different perspectives of a common property of the shooting. The upcoming question is whether all of them should be used or not? If all the related features are kept, there would be redundant information. Otherwise, some critical information may be lost.
d. Lastly, there are peaks and troughs during Kobe's 20-year career. How this information could be employed for a better prediction?

The potential solutions we are going to use is to employ data visualization methods to initially judge the usefulness of every single feature with the help of our domain knowledge in basketball. Regarding

TABLE I
AN OVERVIEW OF DATASET

Feature Name	Explanation	Type
action_type	type of shooting action	categorical variables with 57 possibilities
combined_shot_type	combined type of shooting action	categorical variables with 6 possibilities
game_event_id	ID of an event in a game	integer
game_id	ID of a game	integer
lat	latitude of shooting position	float
lon	longitude of shooting position	float
loc_x	x location of shooting position	integer
loc_y	y location of shooting position	integer
game_date	date of a game	date (year-month-day)
season	game season	categorical variables with 20 possibilities
period	1 quarter in a single basketball game	categorical variables with 7 possibilities including 3 extra time
minutes_remaining	minutes remaining in the period	integer (0-11)
seconds_remaining	seconds remaining in the minutes	integer (0-59)
shot_distance	distance between shooting position and basket	integer
shot_type	2PT or 3PT	categorical variables with 2 possibilities
shot_zone_area	a partition of court	categorical variables with 6 possibilities
shot_zone_basic	a partition of court	categorical variables with 7 possibilities
shot_zone_range	a partition of court according to distance	categorical variables with 5 possibilities
match_up	home or visit	string
opponent	opponent	string
playoffs	regular or playoffs	binary
shot_id	ID of a shot	integer
team_id	ID of a team	sequence of integer
team_name	name of team	string
shot_made_flag	result of a shot	binary

categorical variables, several feature transformation methods are proposed. Besides, by the trial and error method, we decide a set of predictors for our predictive models.

B. Feature Visualization and Selection

In this subsection, we will conduct a preliminary analysis on the dataset on Kobe's shot history. With the domain knowledge and common senses in basketball games, the available features could be classified into 5 major types consisting of shot action types, spatial factors, temporal factors, uncertain factors and irrelevant factors. In the following, we will investigate each of them and try to extract the information conveyed.

1) Shot Action Type: We first consider the action type performed in a shot which includes combined_action_type and action_type two features. The feature action_type consists of 57 categories, which is shown in Fig. 1 where the x-axis is the action type performed and the y-axis denotes the corresponding accuracy according to the training data. It can be easily observed that there is an enormous variation among these 57 categories, which implies that the feature action_type is of great importance. For example, we can see that the accuracy of Kobe's jump shot is only 33% while his driving dunk shot accuracy is 97%, which is quite reasonable as dunk shots usually happen when a player is in a good situation (e.g. defense leak out) and the shooting result is usually successful. This suggests that when it comes to predicting the result of a dunk shot, it is always inclined to predict it to be a successful shot. Actually, our experiment results validate our conjecture that action_type serves as one of the most dominant factors for this shot prediction problem.

Besides, the feature combined_action_type consists of 6 categories and it is actually as a coarse classification of action_type, which in shown in Fig. 2.

2) Spatial features: Next, we will examine the position-related information on Kobe's shots. First of all, by comparing the features loc_x and loc_y with lat (latitude) and lon (longitude), we found out that they are almost one-to-one mapping to each other. Thus, we drop lat and lon to avoid the repeated information.

The features shot_zone_range, shot_zone_basic, shot_zone_area and shot_type are actually categorized variables based on different spatial partitions of a basketball court, which are shown in Fig. 3 to Fig. 6, respectively. By visualizing them, we gain insights on how shots are classified according to one of them. Actually, these four partitions are from the basketball domain knowledge. In particular, by comparing Fig. 5 with Fig. 6, we have one more observation that the feature shot_type is actually a more coarse partition of the court in comparison with shot_zone_range, where the feature shot_type only splits the court into 2PT and 3PT field.

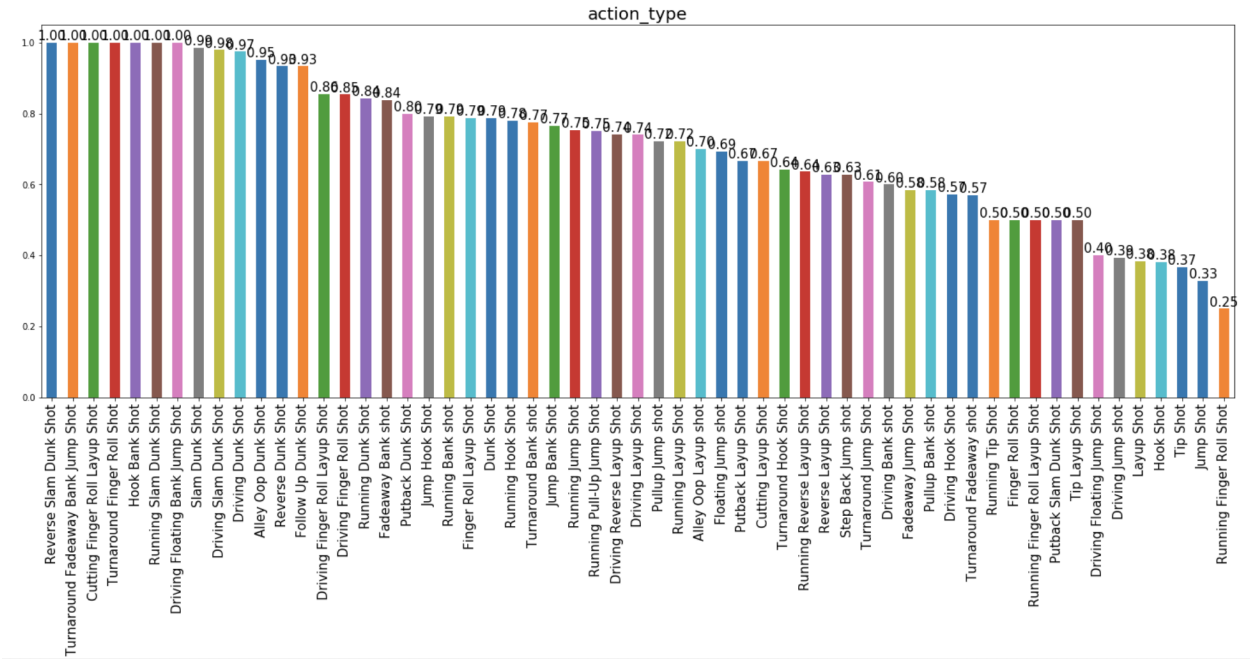

Fig. 1. Accuracy versus action_type

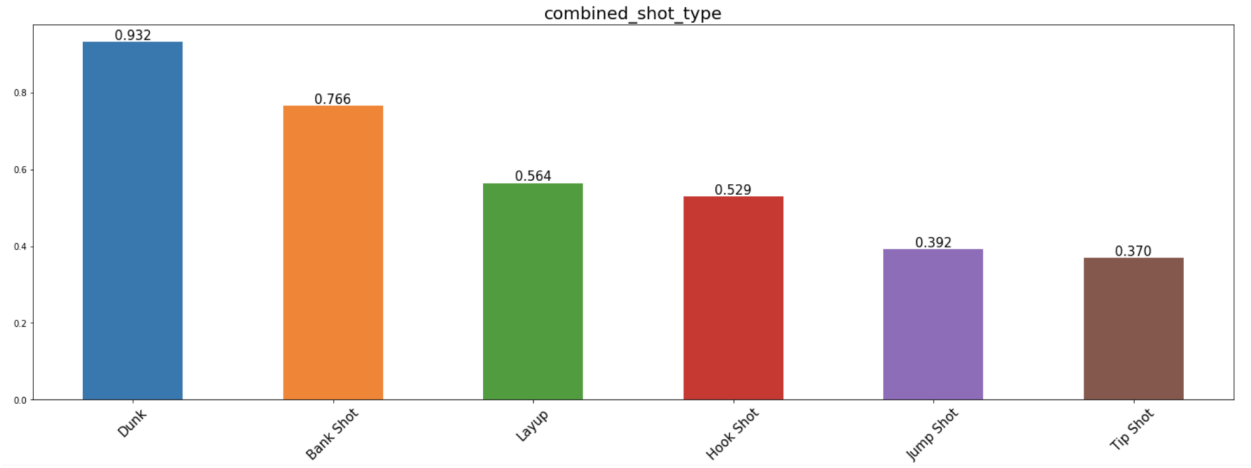

Fig. 2. Accuracy versus combined_shot_action_type

To better understand the potential usefulness of these 4 spatial features, the accuracy and the attempt number of each zone of the court are further provided in Fig. 7 and Fig. 8, respectively. Fig. 7 shows that all these four spatial features are informative. This is because we can easily see that, for either one of the 4 partitions of the court, there is a distinct variation of accuracies among the zones. For example, we could observe that the shots near the basket (restricted area or less than 8ft) maintain the highest accuracy in comparison with the other zones. Besides, the shots from backcourt are always with the lowest accuracy as it is too far from the basket. Furthermore, according to the total attempts shown in

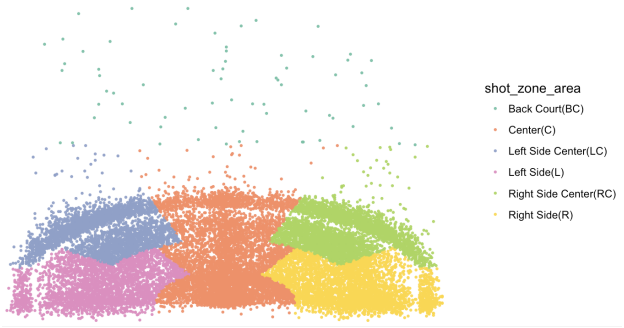

Fig. 3. Partition of the court by shot_zone_area

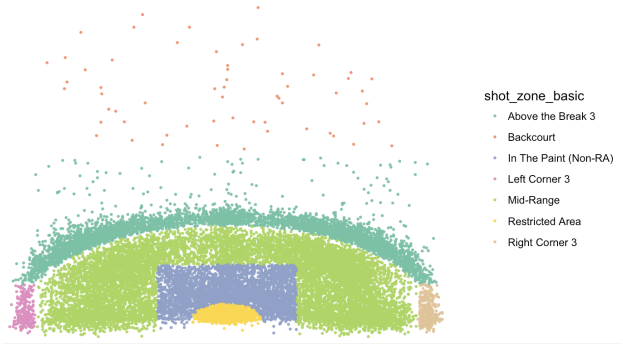

Fig. 4. Partition of the court by shot_zone_basic

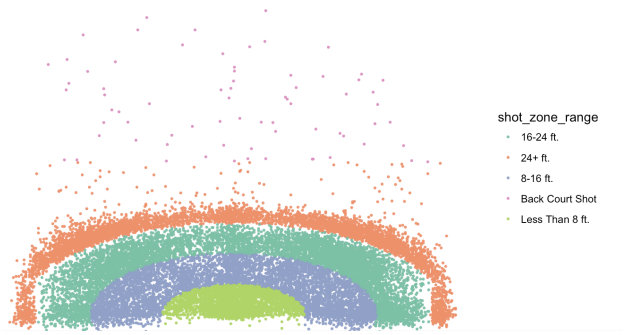

Fig. 5. Partition of the court by shot_zone_range

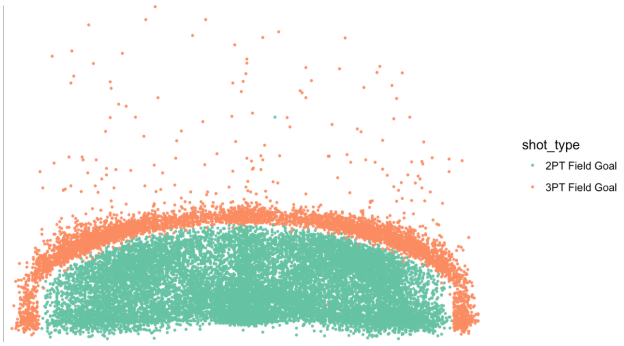

Fig. 6. Partition of the court by shot_type

Fig. 8, we could also observe that Kobe makes most shot in center region, and a slightly higher preference on the right side over the left side. In addition, even though the backcourt contains the lowest accuracy, the actual amount of attempts are low as well.

By looking into Fig. 7 and Fig. 8 jointly, we can see that Kobe makes more shots within 8 feet to the basket (or within the restricted area and the center area) and he also achieves a higher accuracy in these areas. This indicates that the closer the shooting distance to the basket is, the more attempts and the high accuracy are make by Kobe, which further implies that Kobe Bryant is aware of making shots in his “comfortable zones” so as to be efficient in the games.

Based on the above visualizations and analysis on the spatial features, we can conclude that almost all the position-related features convey useful information even though there are correlation and overlapping among them. In particular, these position-related features are either in partial overlapping or inclusion relationship with the others. As a result, what follows immediately is the question that how are we going to deal with these features? Are we supposed to eliminate the correlation (or overlapping) before using them, or are we going to include all of then regardless of the potential redundant information?

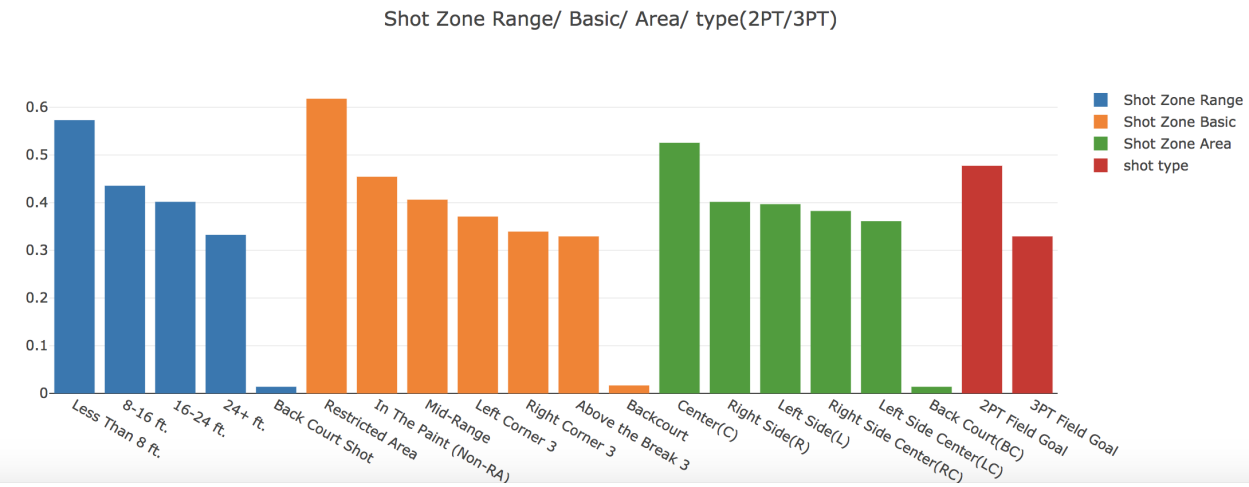

Fig. 7. Accuracy versus shot_zone_range, shot_zone_basic, shot_zone_area and shot_type

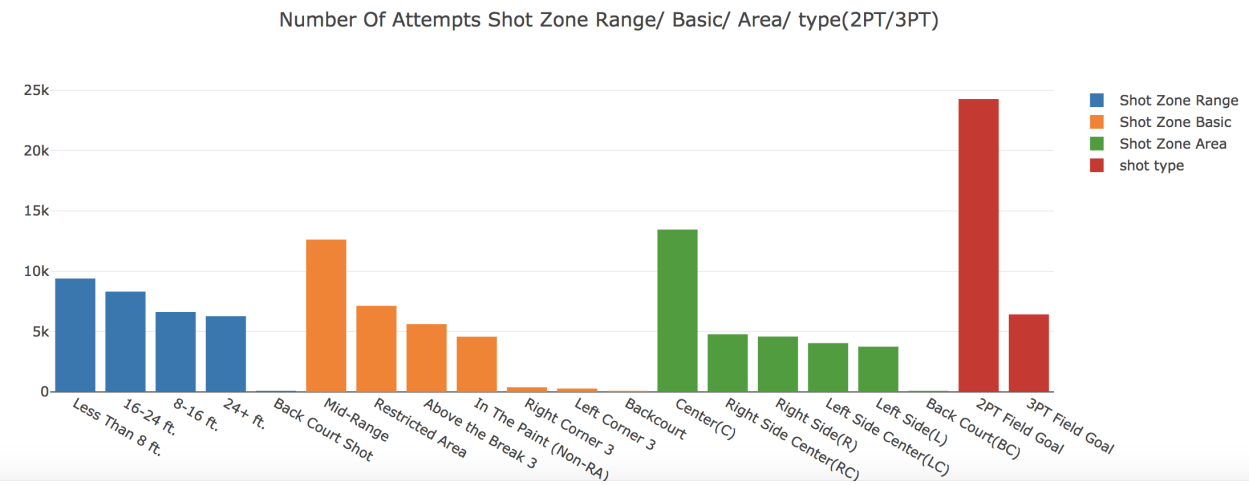

Fig. 8. Attempt number versus shot_zone_range, shot_zone_basic, shot_zone_area and shot_type

Actually, based on our experiment results, including most of them indeed achieves the best prediction performance. In particular, a more refined partition of the court is proposed in Sec. II-C to enhance the prediction performance. We have to admit that there must exist redundancy when multiple of these features are employed simultaneously. Nevertheless, they represent Kobe's shooting accuracy in the different zone of the court from different perspectives, which indeed helps to avoid the loss of information.

3) Temporal features: Aside from spatial features, one more critical impact is coming from the temporal factors. The most important one is unquestionably the game season. Fig. 9 depicts the variation of Kobe's accuracy versus his 20 game seasons. Actually, the shown results correspond well to some important events in Kobe's career. For example we could match the first fall of accuracy from 2002 to

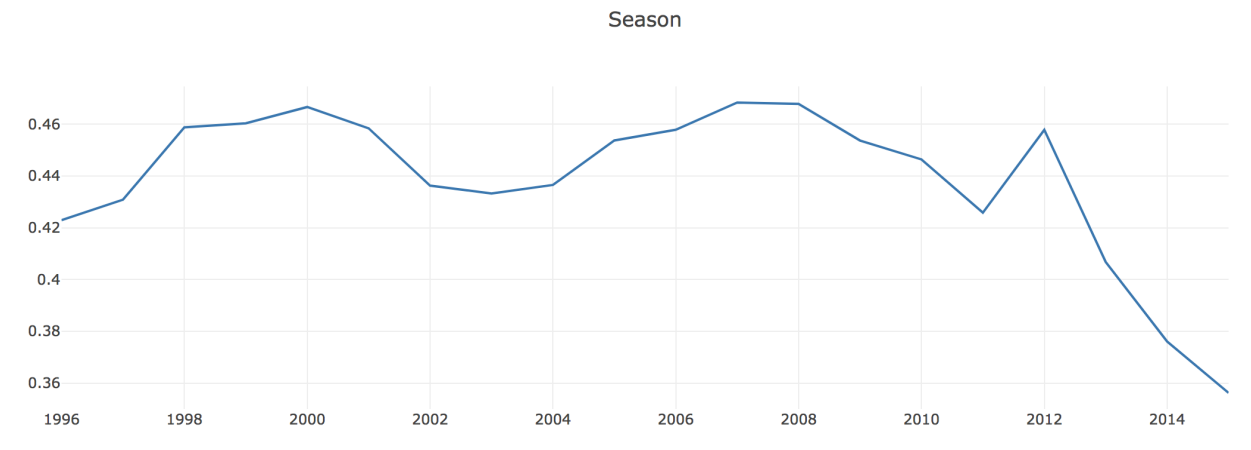

Fig. 9. Accuracy versus season

2004 with Kobe's sexual harassment scandal and the second performance fall from 2011 could be due to his injury on the Achilles' heel and that he was in the later period of the career as well.

Besides long-term temporal feature like game season, we further investigate the short-term temporal features including seconds_remaining, minutes_remaining and period. From the visualization results shown in Fig. 10 to Fig. 12, it seems that some information is contained by these features but it is not easy to interpret any reasonable patterns, especially for the feature seconds_remaining. Nevertheless, we could calculate and reconstruct a new feature named 'seconds_from_game_start' from the previously mentioned three temporal features and this newly extracted feature is the actual elapsed time in a game. If we further take a look into the total attempt number and the corresponding accuracy versus 'seconds_from_game_start', we can have an observation that at the end of each period, there is a peak on number of attempts while the corresponding accuracy is relatively low. A possible explanation on this phenomenon is that due to pressure and eager of victory, Kobe tends to make more shots when the game is about to end, but those shots are actually risker. Actually, many of those shots are far away from the basket, which also results in a low accuracy. Were it not for the timing pressure, Kobe would manage to move into a more favorable position before shooting.

4) Uncertain Factors: We also suspect that some other factors might be useful for predicting the shooting results and they are opponent, match_up, playoffs. Fig. 14 shows the accuracy versus different opponents. As we can imagine, different opponents may put different levels of defense to Kobe, which would definitely impact his performance. For example, if it was a strong opponent, Kobe might have lower accuracy. From feature match_up, we could extract that whether it was a home game or not. In

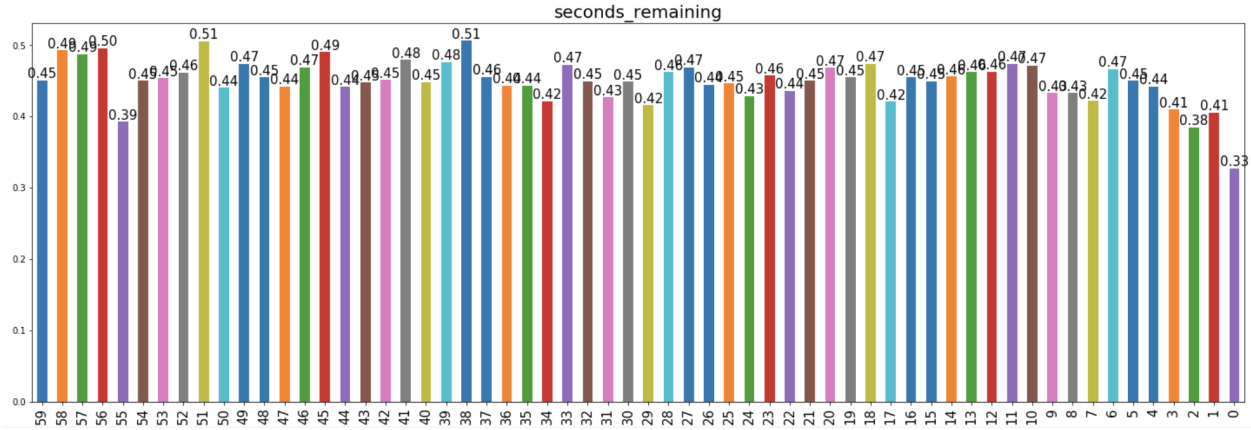

Fig. 10. Accuracy versus seconds_remaining

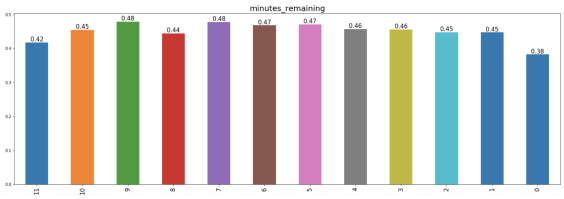

Fig. 11. Accuracy versus minutes_remaining

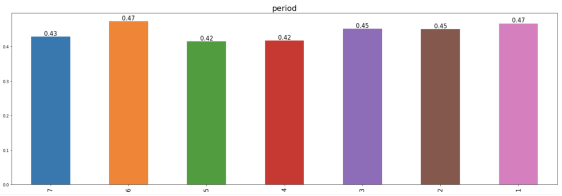

Fig. 12. Accuracy versus periods

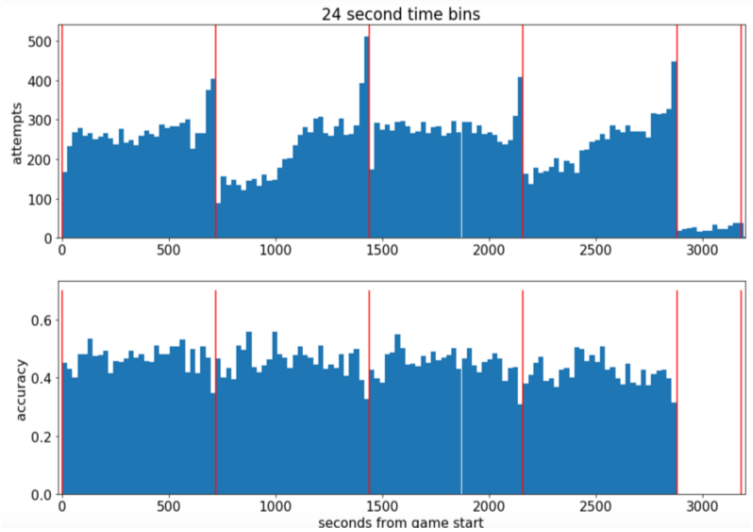

Fig. 13. Accuracy and attempt number versus seconds_from_game_start

the world of sport, we always suppose that home game provides somehow advantages. Accordingly, we denote this new feature as a binary variable ‘home_feild’. Regarding the feature playoffs, we suppose that playoffs would result in a lower accuracy in comparison with regular games since the opponents in playoffs are more competitive. However, the variation of overall accuracy is quite small for both playoffs

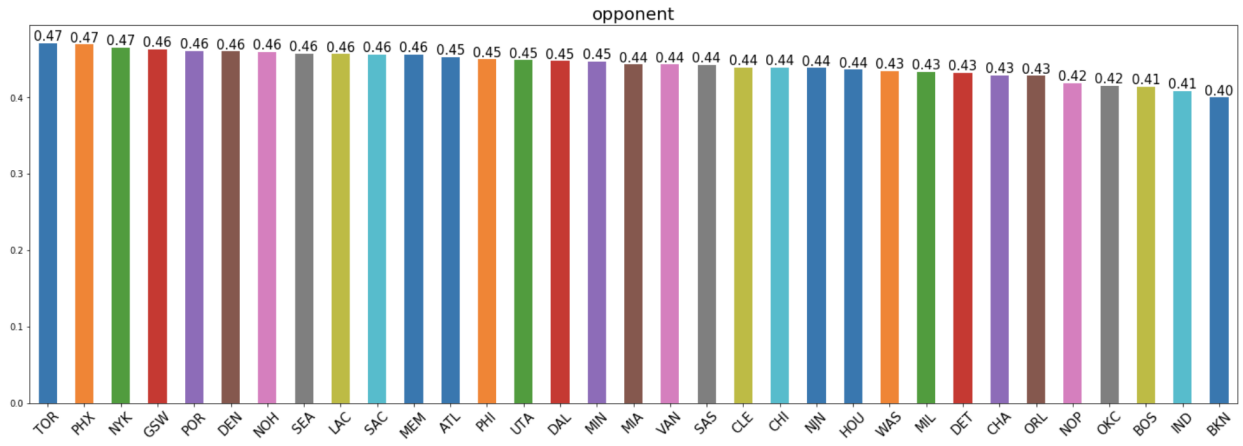

Fig. 14. Accuracy versus opponent

and home_field. But we still believe that they could be useful as the data mining training process may combine opponent, match_up and playoffs together to generate some useful hidden patterns.

5) Irrelevant Factors: We have dropped the following features as we regard them to be either uninformative or their information is captured by some other features. They are game_event_id, game_id, lat, lon, game_date, shot_id, team_id and team_name.

C. Proposed Feature Transformation Methods

From the analysis in previous sections, we can see that a lot of features are provided. Some of them might be useless while some of them are of great importance and hidden information. In order to capture as much as possible the useful information conveyed by this dataset, lots of time are spent on feature selection and extraction. Based on the feature visualization and a coarse selection performed in Sec. II-B, the filtered informative features can be divided into 4 classes summarized in Table II. We are going to present our dealing methods to each of them in order to extract the core information inside of them and provide our final predictors.

According to Table I and II, it is not hard to see that many of the features are categorical and can not be used directly. As a result, appropriate feature transformations are required to make them applicable. In this project, some transformation methods such as dummy coding, feature clustering and information representation are employed to deal with the features listed in Table II. The proposed processing methods are summarized as below and the final predictors are given in Sec. II-D.

TABLE II
INITIAL SELECTION OF USEFUL FEATURES

Type	Index	Feature Name
Spatial	1	loc_x
	2	loc_y
	3	shot_distance
	4	shot_zone_area
	5	shot_zone_basic
	6	shot_zone_range
	7	shot_type
Temporal	7	seconds_remaining
	8	minutes_remaining
	9	period
	10	season
Shot action type	11	action_type
	12	combined_action_type
Other	13	match_up
	14	opponent
	15	playoffs

1) Dummy coding: For spatial features such as shot_zone_area, shot_zone_basic, shot_zone_range, shot_type, temporal features season, and shot action type feature combined_action_type, action_type, dummy coding is used to transform these features into numeric forms. Simple dummy coding is good enough for these variables.

2) Accuracy-Based Feature Clustering: Based on the definition of the features shot_zone_range and shot_type. We have following two important observations: (1) The first observation is that shot_zone_range and shot_type are actually two different partitions of the basketball court in terms of the distance between shooting position and basket. To be specific, feature shot_zone_range partitions the court into 6 parts while feature shot_type partitions the court into 2 parts. According to the visualization results provided in Sec. II-B, especially by Fig. 8 and 7, both of them are informative. As a result, we believe that the distance is an important factor in determining the result of a shot. Motivated by this, some more sophisticated methods are promising to strengthen this factor. (2) The second observation is that we have another important feature named shot_distance. Although it should be a continuous numeric value in real world, in our studied problem it could be regarded as a categorical variable since the shot_distance is in the type of integer, which is therefore quantized and countable.

Based on the above two observations, we are inspired to create a more refined partition of the basketball court in terms of the distance by leveraging the feature shot_distance. This newly created partition is

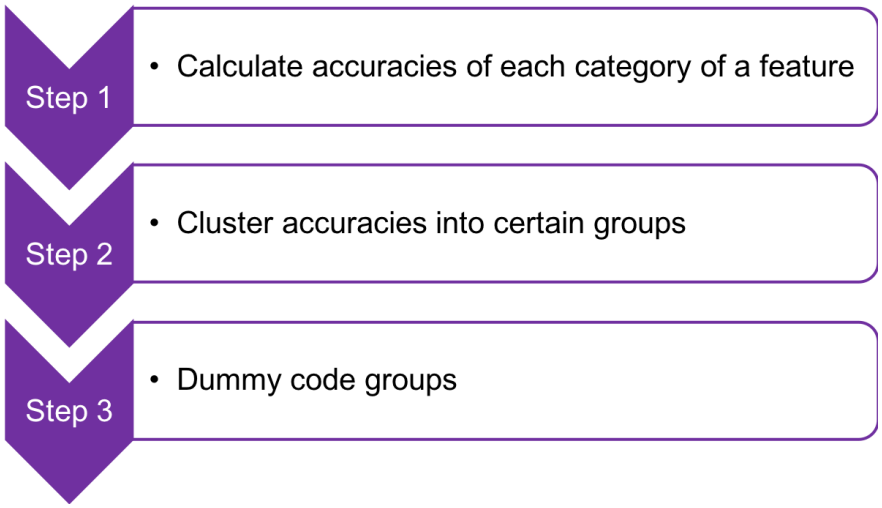

Fig. 15. Process of Accuracy-Based Feature Clustering

expected to serve as a more reliable feature to replace shot_zone_range and shot_type by achieving a better prediction performance. This proposed feature transformation method is termed as Accuracy-Based Feature Clustering, of which the process is shown in Fig. 15. The essential steps are illustrated in the following.

To be specific, first of all, we are able to compute the accuracies of each possibility of the distance since shot_distance is an integer in our case. Secondly, based on the calculated accuracies, we could use some clustering algorithm such as k-nearest neighbors (KNN) to group the similar accuracies. Lastly, we use dummy coding method to turn the newly created categorical variable into a numeric presentation. Actually, the nature of our proposed accuracy-based feature clustering method is to sort the accuracies of different categories and make similar accuracies into a more correlated group. The intuition behind is to provide a more accurate partition of the basketball court by taking the accuracy into considerations instead of simply using the distance. In our proposed predictive model in Sec. II-D, we have partitioned the accuracies of distance into 10 groups, which means that the basketball court is divided into 10 types of distance.

In addition, the proposed accuracy-based feature clustering could also be applied to other features for sake of reducing the dimensionality of the predictors. A simple illustration is provided in Fig. 16, where the feature season is used as an example. Kobe Bryant has a career of 20 years, thus there are 20 categories of this feature. In the clustering method, the accuracy that Kobe got for each season is firstly computed and it is shown in the graph below in Fig. 16. As we can see, there is a difference among

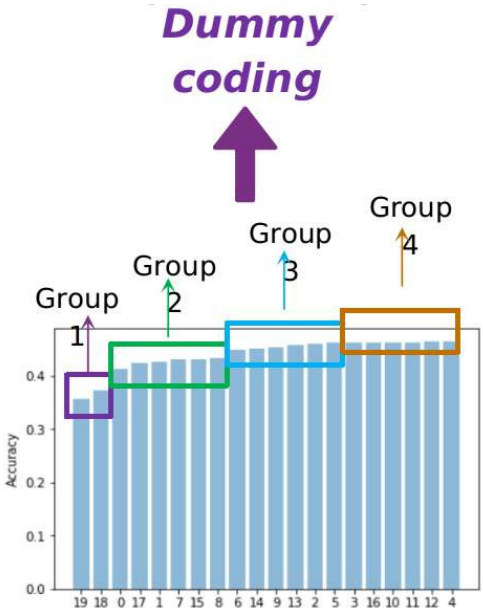

Fig. 16. Toy Example of Accuracy-Based Feature Clustering

different seasons. For example, in season 19 and 18, the accuracies are much lower than others because of the injure Kobe had and he really struggled in those two years. With the computed accuracies, we can further manually or use KNN algorithm to divide these game seasons into several groups. This toy example divides the season into 4 groups. This is only a 1-d partition problem and would be quite simple. Now 4 newly generated groups are given and the dummy coding method could be applied. In the end, a 20-categories feature is compressed into a 4-categories feature, which could help to ease the potential overfitting issue.

3) Category-to-Accuracy Method: It is also of great importance to make feature interpretable in data mining. Inspired by this principle, we propose the following feature transformation method: directly use the accuracy of each quantized or categorical level, namely that for each possibility of a countable feature, we calculate its accuracy rate, then this dictionary of probability would be used to provide a new numeric feature. The process is shown in Fig. 17. First of all, we need to quantize the feature if it is a continuous variable. Then the accuracies of each quantized level could be computed. Afterwards, we can directly use the calculated accuracy as a feature or apply the previously proposed accuracy-based feature clustering. In our final predictive model, we apply this method to the season feature. For example, If one wants to predict a shot that happens in game season 2002-2003 , then the accuracy of game season 2002-2003 is assigned to it as an extra numeric feature. Actually, this special processing of season makes sense

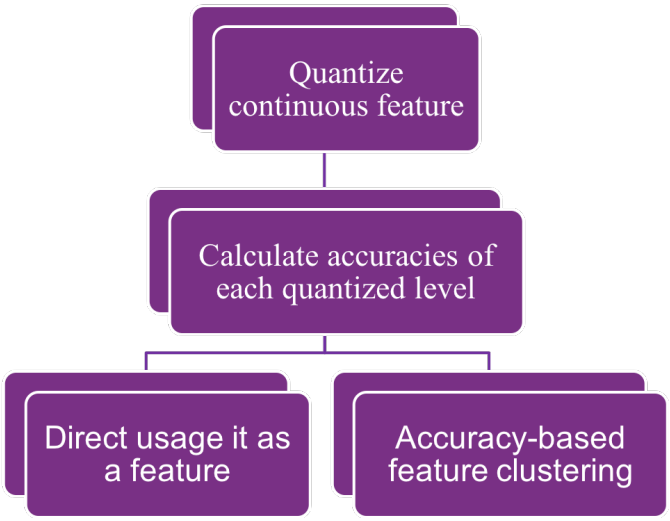

Fig. 17. Feature Quantization Process

because in the world of sports, players' career time has a close relation to their performance. Thereofre, the accuracy of each season would be informative.

4) Information Representation: In this part, we are discussing how we treat the other special features. First of all, let us consider the temporal features. Actually, seconds_remaining, minutes_remaining and period could be aggregated and represented by a new feature termed as seconds_from_game_start as mentioned before, which is the time from the beginning of the game in the unit of seconds. Actually, this newly created feature can be interpreted as the physical power of Kobe because when time goes by in a game, a player's energy is becoming less and less, which has a great impact on his or her performance. In addition, in Sec. II-B, we have revealed that Kobe has a relatively low accuracy at the end of a period. Accordingly, we created a binary feature termed as last_5_sec_in_period, which indicated whether the shot is performed in the last 5 seconds of a period. Besides, from match_up, a binary variable is used to indicate whether it was a home game or not. The opponent is transformed by dummy coding method.

In this subsection, we have provided some specific feature transformation methods to treat the categorical variables, continuous variables and etc. By proposing these methods, the original features have more viability and options to be interpreted and transformed. In the next subsection. Our final predictive model is built based on these transformation methods and the final predictor is given accordingly.

D. Predictive Modeling

Based on the feature selection and extraction techniques presented in Sec. II-B and II-C, we now are able to transform the original features in many different ways. On one hand, this diversity of feature transformation provides us with lots of possibilities of feature selection so as to alleviate the lost of information. On the other hand, it however makes the modeling process become more difficult due to multiple overlapping features which cause confusion and redundancy.

In this project, we have widely applied our proposed category-to-accuracy method and accuracy-based clustering methods to transform the informative original features. With trial and error method, we identify the potential effects of the original features and their corresponding transformed representations. A good combination of the features is picked by our best efforts even though it is not guaranteed to capture all the information contained in the dataset. Nevertheless, our performance results validate the viability and superiority of our picked predictors in the following part. To be specific, the finally selected predictors and their corresponding treating methods are summarized as below in Table III:

TABLE III
FINAL PREDICTORS FOR KOBE BRYANT SHOT PREDICTION

Index	Feature Name	Dealing Methods
1	action_type	dummy coding
2	combined_shot_type	dummy coding
3	loc_x	keep original integer value
4	loc_y	keep original integer value
5	shot_distance	keep original integer value
6	new_shot_distance_range	accuracy-based feature clustering and dummy coding
7	shot_zone_area	dummy coding
8	shot_zone_basic	dummy coding
9	season	dummy coding
10	season_accuracy	category-to-accuracy method
11	seconds_from_game_start	information representation method
12	home_field	0 for visit game and 1 for home game
13	opponent	dummy coding
14	last_5_sec_in_period	dummy coding

From Table III, we can see that 14 features are selected (or extracted). In particular, it is worth to point out that among these predictors, action_type, season, new_shot_distance_range are of greater importance in comparison with the others. This makes senses as the type of shooting, time of Kobe's career and shooting distance are unquestionably the most important factors for the shooting result.

In this project, many different classic classification models are applied to test their performance on such a prediction problem, which includes Logistic Regression, Decision Tree, Random Forest, AdaBoost, GBDT and XGBoost. In particular, we are primarily focusing on XGBoost which outperforms the others by achieving a much better prediction result in terms of Log Loss. The numeric results are further given in the next subsection.

E. Empirical Results and Comparisons

Employing the classic models mentioned above with the empirically selected features in Table III, we herein further display the final experimental results in Table IV. In this project, a virtual machine in Microsoft Azure with 16 CPUs and 64 GB memory are employed. The metrics we used include Log Loss, Accuracy and ROC Score, where Log Loss is the metric applied by this Kaggle competition and its definition is given as below:

$$\text{Log Loss} = -\sum_{i}^{N} \left(y \log(p) + (1-y)\log(1-p)\right), \tag{1}$$

where y is the binary indicator of that whether class label c is the correct classification for an observation o, p is the model's predicted probability for that the observation o is of class c, and N is the sample number.

TABLE IV
EVALUATION FOR EACH PREDICTIVE MODEL

Model	Log Loss	Accuracy	ROC Score	Rank in Kaggle (Log Loss)
Logistic Regression	0.609193	0.680800	0.697027	487/1117
Decision Tree	0.610539	0.678400	0.695851	531/1117
Random Forest	0.610914	0.671200	0.700825	540/1117
AdaBoost	0.642629	0.658800	0.649930	730/1117
GBDT	0.602738	0.680000	0.710514	258/1117
XGBoost	0.598824	0.680800	0.716694	17/1117

In particular, the parameters that we used in GBDT and XGBoost are given in Table V and Table VI, respectively:

To better understand the results shown in Table IV for these models, a brief review of these models and their applicability on our studied scenario are provided in the following.

TABLE V
PARAMETERS FOR GBDT CLASSIFIER

Index	Parameter Name	Value
1	n_estimators	950
2	max_depth	7
3	learning_rate	0.0038

TABLE VI
PARAMETERS FOR XGBOOST CLASSIFIER

Index	Parameter Name	Value
1	n_estimators	1025
2	max_depth	8
3	learning_rate	0.0039
4	min_child_weight	3
5	max_delta_step	1
6	colsample_bytree	0.7
7	objective	'binary:logistic'

*1) **Logistic Regression (LR)**:* Logistic Regression is a standard regression model where the dependent variable is categorical. It uses logistic function for prediction, which is defined as follows:

$$\sigma(t) = \frac{1}{1 + e^{-t}}, \tag{2}$$

where t is a linear function of a single or multiple explanatory variables $\mathbf{x}$ which can be regarded as a set of feature inputs. LR is fast and easy to train while preserving enough robustness, which makes it a good benchmark model.

In this project, we apply LR to help initially judge the usefulness of many features and its different transformations. However, LR under-performs when there are some nonlinear decision boundaries and complex relationships between dependent variables and independent variable. We assume that there do exist some nonlinear relationships in our situation, so we can see this model does not work well. Nevertheless, it provides us a baseline for comparison.

*2) **Decision Tree (DT)**:* Decision Tree is a tree-like model that is commonly used in classification. Each internal node represents a "test" on an attribute so as to split samples into to sub trees. Each branch represents the outcome of the "test" and each leaf node represents a class label. At each split, DT uses some criteria to decide which attribute to use. Two commonly used ones are entropy and entropy. One reason for choosing decision tree as one of our models is that it is as easy as logistic regression to understand and implement. Besides, it provides the basic idea for common ensemble methods, which we also tried in our project, so that it can help us better compare the models.

However, we know that the performance of DT is unsatisfactory to many application and it is not surprising for our studied problem. DT assumes some dependencies between features in order to achieve a good perform. In our situation, some features are actually irrelevant, which may account for its final

under-performance.

*3) **Random Forest (RF):*** Random Forest is a bootstrap aggregating (bagging) model which is commonly used in classification. Its basic idea is to build multiple decision trees and use voting or averaging to obtain the final prediction result. For each tree, a set of samples are selected based on bootstrapping method. We then use these samples to build the tree. At each splitting node, a random subset of the features is selected to find out the most suitable "test" attribute.

The advantages of RF can be, it helps reduce the variance of a single decision tree and achieves decoupling so as to make each tree own as much information as possible. However, since we do not have too many features, the maximum number of effective trees to be included is restricted as no more information can be dug. Another weakness of RF in our situation can be that all of the features in Table III are significant after feature engineering. This may cause low accuracy as each tree in RF lose some important information by randomly picking parts of the features to be split at each node. In all, we find that RF does not provide a satisfactory performance.

*4) **AdaBoost:*** Adaboost is a boosting ensemble method, which provides an idea to process weak learners and obtain a final strong estimator by procedurally editing the weights of sample points. With the training set $(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ..., (\mathbf{x}_m, y_m)$ as inputs. The basic idea is to decorate samples that have been misclassified in the previous step with higher weights, so that later step will focus more on these samples and try to classify them correctly in order not to get high loss. In the end, we collect all the models we have built with some weights to produce our final model and use it for prediction. Notice that later built models should have more weights.

In our case, the results show that AdaBoost did the worst among all the models we tried. Perhaps it is because part of the weak learners have accuracy lower than 0.5, which makes the following steps go in the opposite and finally produces a bad estimator.

*5) **Gradient Boosted Decision Trees (GBDT):*** GBDT is another form of boosting ensemble method. It allows the optimization of an arbitrary loss function. At each stage, GBDT improves the current weak learner by constructing a new model that adds an estimator to it. In order to find that estimator, the gradient boosting will fit it to the residual, and use a negative gradient on the loss function to find that estimator that corrects the current model.

From Table IV, GBDT is the second best model. While GBDT performs better than traditional boosting methods thanks to its gradient algorithm, it does have some defects, such as it is more prone to over-fitting

and we have to find good parameters in order to build a good model.

*6) **XGBoost**:* XGBoost is an extreme boosting model useful in machine learning. It is developed with both deep consideration in terms of system optimization and principles in machine learning and is not limited to a single algorithm. Some of the advantages that XGBoost possesses over other boosting method (such as GBDT) are listed below:

- XGBoost takes the Taylor expansion of the loss function up to the second order.
- XGBoost supports random features selecting for each learning, which reduces overfitting as well as training time.
- XGBoost uses a more regularized model formalization to control overfitting, which can usually give better performance.
- XGBoost has better support for multicore processing and parallelization.

Since XGBoost is an optimized ensemble method with these specific advantages, we believe it can help us better solve this Kobe Bryant' Shot prediction problem.

According to Table IV, we can find that XGBoost has the lowest log loss and highest ROC score, which validates our selection that it could be the best model among all the models we have tried. From the column of predicting accuracy, we can see none of these models have very high predicting accuracy, which implies the problem we are facing is not highly predictable and it is not so easy to find a perfect solution for our situation. In Table IV, we also list the rank of each model in this Kaggle competition based on its Log Loss on testing data. We can see that GBDT has a decent performance and XGBoost is even at top 38 among 1117 teams.

The ROC diagram is further provided in Fig. 18. We can see that no single model is highly distinguishable among all the models in terms of ROC score. But in all, XGBoost still works the best in our situation. Part of the reason may be, XGBoost has some particular advantages over other models. In conclusion, XGBoost is a good choice in our studied scenario.

III. Extended Prediction Based on Chronological Order of Shots

In this section, we extend our current prediction problem to a more challenging scenario, where the chronological order of shots is taken into consideration. In this case, we try to predict the result of a shot only based on the training data (shooting records) that happened prior to it. Therefore, an unavoidable challenge herein is that for those shots happened at the very beginning of Kobe's career, only a few

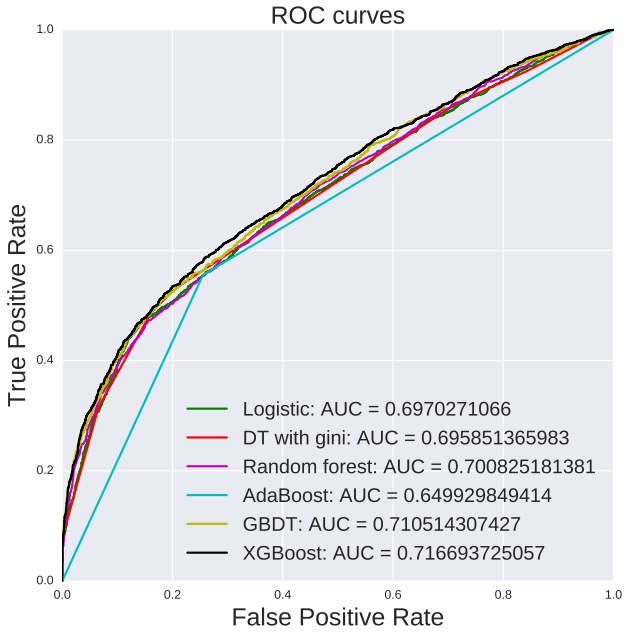

Fig. 18. RoC Curves of different models

training data are available. This will result in building weak learners. Accordingly, this setting with the chronological order of shots does pose a new challenge on the prediction of Kobe's shots.

Based on the above explanation, we are expected to train different models for different shots that need to be predicted (at least their training data is different). Unfortunately, this would be time-consuming as the complexity becomes $O(nT)$, where n is the number of test data unit and T is the complexity of building one single model, especially when XGBoost model is applied and T is relatively large.

In our investigated scenario, there are 5000 shots that need to be predicted in the test dataset, which means that 5000 predictive models need to be trained. By considering the complexity, we relax the constraint on training data to the case that we are allowed to use the shooting records that happened on the same day. For example, if we are going to predict a shot that happened on May 4, 2012, the records of shot with the game_date prior to and or equal to May 4, 2012 would be used as training data. Based on this relaxation, the number of models to be trained is reduced. Actually, the 5000 shots in the test dataset have 1457 different dates, which means that we only have to train 1457 models now.

One thing we would like to point out about the test dataset is that they are almost uniformly selected from Kobe's 20 season, which means that the date of shots to be predicted are approximately uniformly

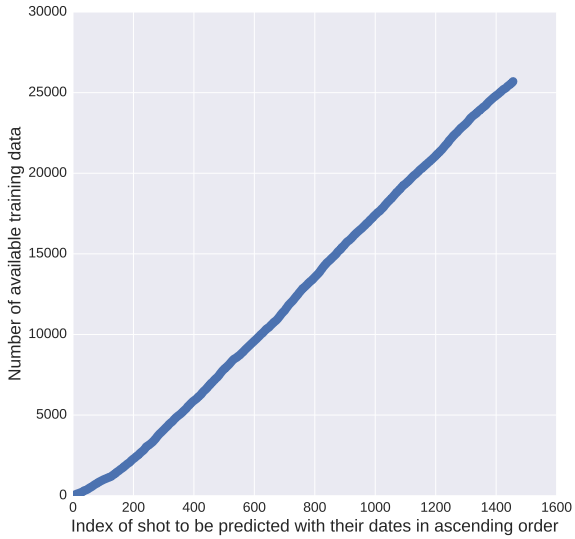

Fig. 19. Evolution of Number of Available Data Units

distributed from 1996 to 2016. To show this, Fig. 19 shows the number of available data units versus the index of shot to be predicted with their dates in ascending order.

For the initial attempt, we try to use Logistic Regression model because of its simplicity. The evolution of accuracy rate and Log Loss is shown in Fig. 20 and Fig. 21, respectively. The overall accuracy is 0.675600 and the overall Log Loss is 0.628203.

From both Fig. 20 and Fig. 21, we can see that the metrics have a dramatic improvement at the beginning of the evolution, which well corresponds to our previously mentioned concern that weak learners are built in predicting the shots happened at the very beginning of Kobe's career due to limited training data. Nevertheless, by jointly observing Fig. 19 to Fig. 21, we can figure out that when the number of training data is larger than around 5000, the predictive model applied is becoming stable. This may suggest that we do not need the whole training data set to achieve the same level of performance.

Furthermore, another observation we can make is that both the evaluation metrics converge fast. This suggests that we can build two sorts of model where one is for the prediction with very few training data and the other could be the model built in this project. Based on this concept, we believe that a better prediction could be achieved for this new challenge.

Now, by leveraging Microsoft Azure with 64 GPUs and 432 GB memory, we use several hours to further evaluate our XGBoost model on this new challenge. The evolution of accuracy rate and Log Loss

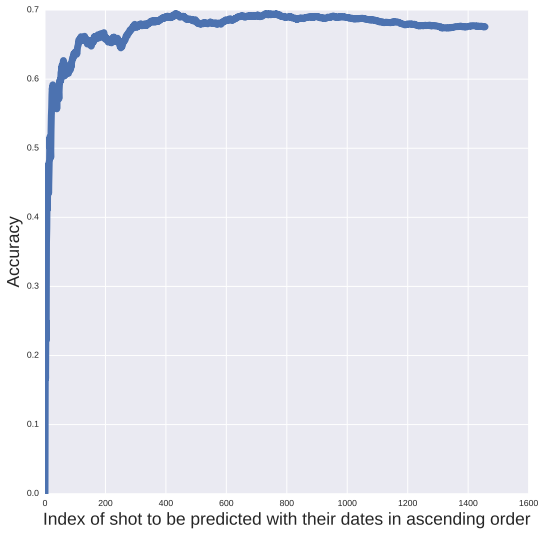

Fig. 20. Evolution of Accuracy Metric

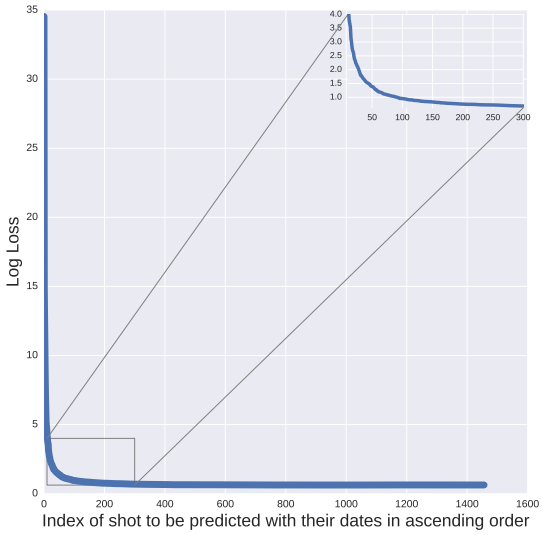

Fig. 21. Evolution of Log Loss Metric

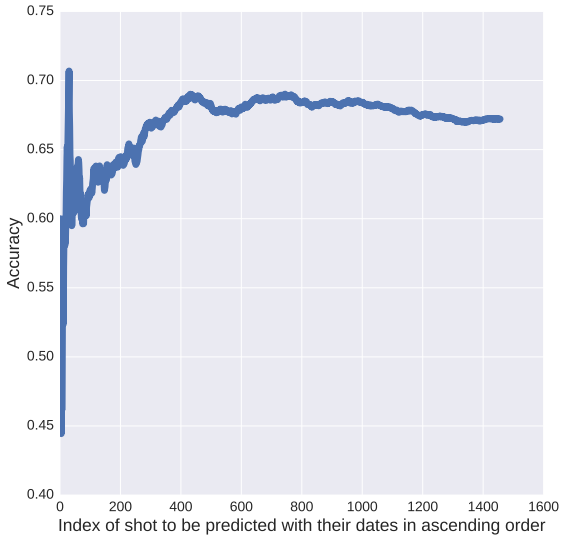

Fig. 22. Evolution of Accuracy Metric

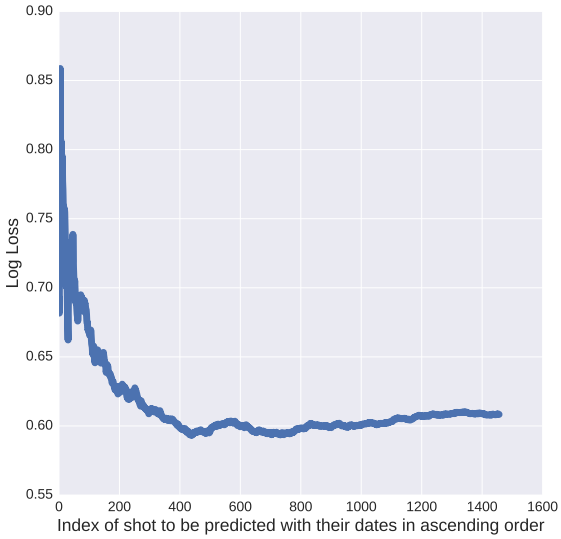

Fig. 23. Evolution of Log Loss Metric

is shown in Fig. 22 and Fig. 23, respectively. For XGBoost model, the overall accuracy is 0.672200 and the overall Log Loss is 0.608349. It is not surprising that we can obtain similar observations from Fig. 22 and Fig. 23, which well validates our intuition of this new prediction problem.

Due to the limited time for this project, only a preliminary trial is provided in this section. Actually,

inspired by what we discussed above, we can even partition the dataset by different resolution of time, such as in terms of week, of month, or even of year. Then the number of predictive models are further reduced. Different partitions of the training dataset may also help to reveal the hidden shooting patterns of Kobe Bryant and even provides more guidances in predicting the future performance of current NBA players.

IV. Lessons learnt and Future Directions

A. Lessons learnt

This an exciting project which not only leads us to discover lots of insight on Kobe Bryant's amazing career but more importantly guides us to analyze a sport and a player from the perspective of data mining. We have gained lots of experience and lessons from this project and they are summarized below:

1) Feature engineering is important (even than parameter tuning).
   - First of all, feature selection and extraction should always be one of the most important processes during a data mining project. A good set of features would provide a good start for training. Whenever we want to make predictions, either regression or classification on a large dataset, we should firstly aim at finding the most important and useful features. This is because they do contribute a lot in the final predicting result. Take the feature action_type as an example, without this feature, the Log loss of any models we tried is above 0.66, which is quite far from what we finally achieved, i.e., 0.5998.
   - There are multiple approaches to find out these useful features. One common and useful way is to visualize the available dataset. For example, by showing the relationship between game season and the corresponding shooting accuracy, we can easily confirm that the feature season does contain relevant information on Kobe's performance. Likewise, we have drawn a figure on loc_x (loc_y) and longitude (latitude), which apparently tells us the almost one-to-one mapping between them. Then we can select the feature that contributes the most and then drop the redundant ones if they could not provide any new information. Even though this approach does not always work as correlated features can provide extra information, it is still an easy and efficient way to select good features and be familiar with the dataset at the very beginning stage of this project.

- Another choice is to use some libraries that can help to label the importance level of each feature, such as Recursive Feature Elimination (RFE) and ET classifier. These libraries can help to select top features that provide the most information towards the target value. For example, RFE recursively trains estimator based on the current features and then discards features with the least importance until the number of features meets the requirements. In our project, we initially tried it to find out some useful features. However, we have many features dummy coded and it is not reasonable to discard part of one single feature. Besides, we have to set an appropriate feature number to use these libraries, which is somehow time-consuming. As we got useful features based on visualization and trial and error method, we did not involve this method too much in the end.

2) Dummy coding method is useful

   Dummy coding is always an intriguing approach to deal with categorical variables. If one variable can be set to multiple categories which are not numerical dependent, dummy code can help to transform the original feature into a numeric-style feature, which provides more flexibility for training a predictive model. In our situation, we have made lots of usage of dummy coding when dealing with the features such as opponent, action_type, season and etc. This process enlarged our feature size while decoupled values of a single categorical variable. It did help us to achieve lower log loss on various models, especially for our final XGBoost classifier.

3) Feature transformation provides new ideas on data pre-processing

   In this project, many informative features are highly correlated to each other as certain features are different interpretations of a common property of Kobe's shots, such as the partition of the basketball court. This phenomenon not only helps to reveal the usefulness of a property but also inspires us to create some new pre-processing methods to make better use of this property. Accordingly, we have proposed our accuracy-based feature clustering method to further explore some useful property of Kobe's shot. Based on available features and domain knowledge, feature extraction sometimes could help us to achieve a better predicting performance.

B. Future Directions

Even though we have tried a lot in order to make a good prediction, there still left much to explore due to the limited time. Some potential directions for future studies on this topic are also shown below:

1) Small sample categories

   According to our analysis, there are some shots belonging to one category with very high shooting accuracy while the number of samples in that category is quite small. For example, the number of samples with action_type as Turnaround Finger Roll Shot is 2 and both of them are successful attempts. We have tried some methods to deal with these "outliers" such as merge them into another similar category. Unfortunately, we did not obtain a better performance. In addition, we have tried Local Outlier Factor Method to detect outliers. However, we could not find good parameters that further reduces the Log Loss. But we are still thinking and believing that there exist some sophisticated approaches to deal with the situation related to unbalanced data categories. For example, to further consider the distribution of the samples and add appropriate weights to them might be a possible choice.

2) Combine multiple models

   We have actually built multiple models, such as Logistic Regression, Decision Trees, Random Forest, Adaboost, GBDT and XGBoost. While we currently only choose the best result of them as our final result, we may combine them to see whether we can get a better performance. In particular, as we can see, GBDT actually have obtained a decent result. Accordingly, one possible way is to use voting classifiers to ensemble XGBoost and GBDT.

3) Multi-Player Joint analysis

   Another direction is to conduct joint analysis on multiple players. For example, we can use one player's data to predict another player's shot accuracy, which may show us some similarities between these two players. Based on these similarities, our training data could be enlarged by incorporating multiple players' shooting history and the prediction performance may be enhanced. Furthermore, we can also try to find some features that contribute more to a higher (or lower) shot accuracy by analyzing datasets of multiple players.

V. Conclusions

In this work, we have investigated the application of data mining techniques in predicting Kobe Bryant's shooting results. In this project, the feature selection has been performed with the help of the domain knowledge in basketball and the advanced data visualization methods. Several feature transformation methods have been proposed to deal with certain categorical features and to reveal hidden information

from existing features. Specifically, an accuracy-based feature clustering method has been proposed to provide a more accurate partition of the basketball court and help to reduce the dimensionality of certain categorical features. It is shown that XGBoost has outperformed other classic classifiers for our studied problem by achieving a ROC score of 0.7167 and a Log Loss score of 0.5988, which ranks us top 17 among 1117 teams in this Kaggle competition. Beyond these, a more challenging prediction has been attempted by considering the chronological order of the shooting records and some primitive results have been shown. In the end, some potential directions on shot predictions have been summarized for future work.

REFERENCES

[1] A. Brahme, "Dissecting the NBA Draft", Online Available: https://towardsdatascience.com/dissecting-the-nba-draft-ee175d7aec31

[2] N. Silver and A. McCann, "CARMELO NBA Player Projections", Online Available: https://projects.fivethirtyeight.com/carmelo/

[3] Renato Amorim Torres, "Prediction of NBA games based on Machine Learning," Online available: https://homepages.cae.wisc.edu/ ece539/fall13/project/AmorimTorres_rpt.pdf

[4] Matthew Beckler, Hongfei Wang, "NBA Oracle," Online available: http://www.mbeckler.org/coursework/2008-2009/10701_report.pdf

[5] Kaggle, "Kobe Bryant Shot Selection", Online: https://www.kaggle.com/c/kobe-bryant-shot-selection

[6] https://stats.nba.com