Kobe Bryant Shot Selection

Zhi Wang

December 3, 2019

1030 Hands-on Data Science Final Project
Brown University
Master of Data Science
Github Link: https://github.com/u0820420/1030-Final-Project.git

1 Introduction

Data Science has play a significant role in NBA League and is changing the NBA landscape. Kobe Bryant has been one of the most dominant players in the NBA through out the last two decades. Shortly after Kobe Bryant retired a couple of years ago, Kaggle released a dataset containing 20 years worth of his shots. The data contained the location and circumstances of every field goal attempted by Kobe Bryant during his 20-year career. The task was to predict whether the basket went in (target variable: shot made flag).

The whole dataset size is 30697 rows * 25 columns. Description follows:

Variable	Info	Type	Grouping
season	Year span like 2000-01, 2015-16; 20 total	Categorical	Date
game_date	Date of the game	Date	Date
game_event_id	Numbered event in game	Integer	Game
game_id	Number assigned to each game	Integer	Game
playoffs	Regular or playoff game	Categorical	Game
minutes_remaining	Minutes remaining in quarter	Integer	Game Time
period	Period. Typically 1-4, but overtime 5,6,7	Categorical	Game Time
seconds_remaining	Seconds remaining in quarter	Integer	Game Time
shot_id	Sequential # for each shot	Integer	Index
lat	X location	Float	Location
loc_x	X location (0.1 ft)	Integer	Location
loc_y	Y location (0.1 ft)	Integer	Location
lon	Y location	Float	Location
shot_distance	Feet from basket, 0 is valid	Integer	Location
shot_zone_area	Left, right, center...6 levels	Categorical	Location
shot_zone_basic	7 levels: Above the Break 3; Backcourt; In The Paint (Non-RA - restricted area); Left Corner 3; Right Corner 3; Mid-Range; Restricted Area;	Categorical	Location
shot_zone_range	One of 5 zones: backcourt; 24+; 16-24 ft.; 8 to 16; less than 8;	Categorical	Location
shot_made_flag	Made/miss, this is what to predict	Categorical	Outcome
action_type	Detail shot type. 57 Levels: Reverse Layup Shot; Running Jump Shot; Jump Shot; Slam Dunk Shot...	Categorical	Shot type
combined_shot_type	More general shot type, 6 levele: Bank Shot; Dunk; Hook Shot; Jump Shot; Layup; Tip Shot	Categorical	Shot type
shot_type	2 or 3 point	Categorical	Shot type
team_id	Lakers	Integer	Team
team_name	Lakers	Categorical	Team
matchup	Opponent and home vs away	Categorical	Team
opponent	Opponent team	Categorical	Team

Figure 1.1: Data Set description

2 Exploratory Data Analysis

First, let's see all shots Kobe attempts under three locations category features with the Field Goal percentage. Note that there is a blank white line on the three points line, players should try to avoid such long two-point shots, because the expectation of getting two points from an approximate three-point shot percentage is very low.

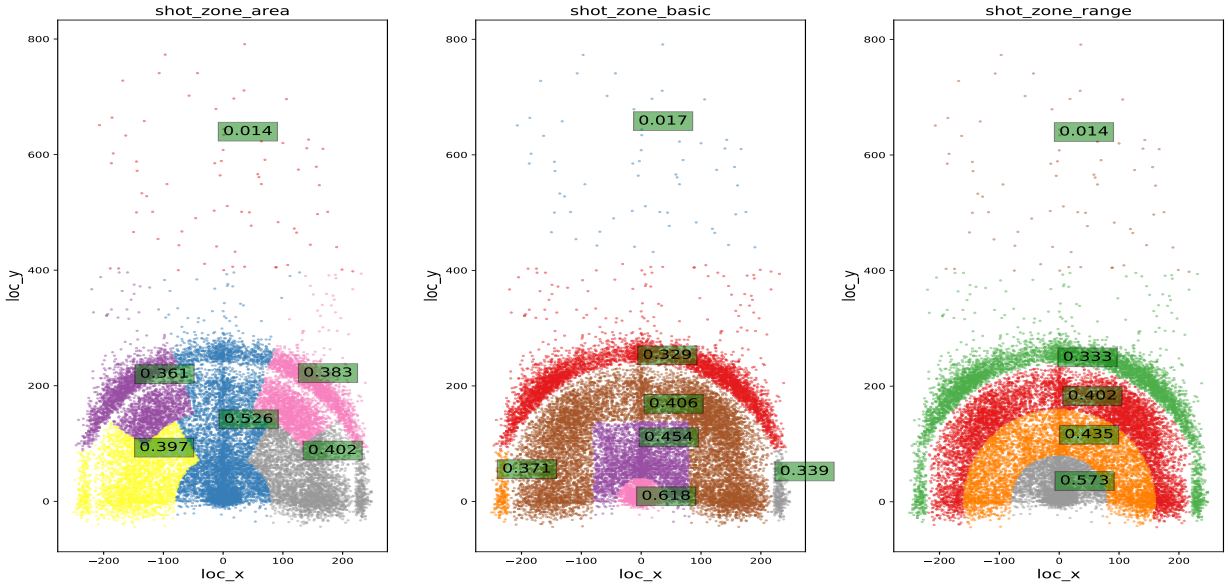


Figure 2.1: Shot Zone under three different locations with Field Goal percentage

Next, we trying to check whether Kobe has preference on attacking the ring, left or right? The Figure 2.2 tells us the distribution of x coordinate shots is almost symmetric, which means Kobe is evenly attack the ring.

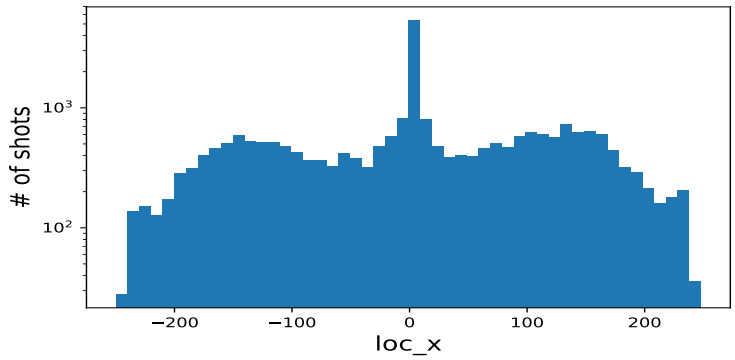
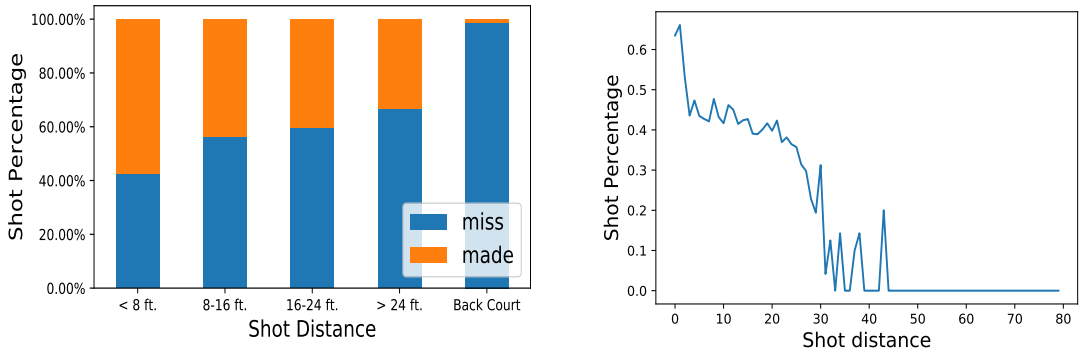


Figure 2.2: x coordinates histogram on Kobe's shots frequency

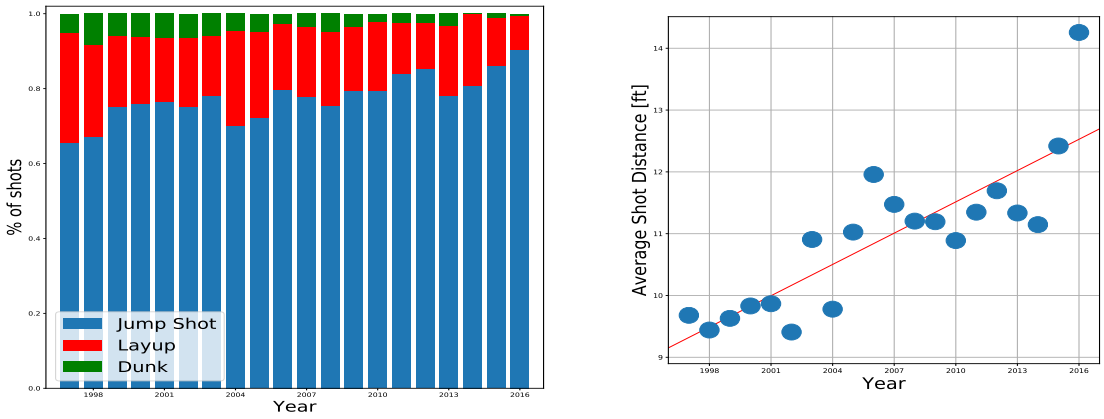
Let's examine how target variable distributes against some of the features. In the Figure 2.3(a) we can see the 'shot zone range' distribution. Apparently, the target variable is unevenly distributed across the subsets defined by the range feature. The shooting percentage decreases as the shooting distance increases. We can see this trend more intuitively in the Figure 2.3(b).



(a) (b)

Figure 2.3: (a) bar plot Shot percentage based on 5 distance range.
(b) line plot Shot percentage based on exact distance

Lastly, let's take a look at how Bryant's shooting style changes with age. In the Figure 2.4(a) we can discover that Bryant's tendency to shoot more jumpers than hits to the basket directly because of his injuries and declining athleticism, as most basketball players do. Figure 2.4(b) also indirectly proves this point, his shooting distance is getting further and further.



(a) (b)

Figure 2.4: (a) bar plot Shot Attempts based on three Shot Type.
(b) Linear regression between Average Shot Distance and Years

3 Models

3.1 Data Preprocess

1. I drop the rows in which shot make flag (target variable) has null values, because they are totally removed by random for compitetion purpose. 25697 remain and 5000 are dropped.

2. Since Kobe only play for Lakers in his career, so I drop the team id and team name and drop the shot id and game id, because they are not gonna help in the model. Drop lon and lat as well, because they are pairwise correlated 100% to the loc x and loc y.

3. Convert minutes remaining in seconds until the end of each period and add it to the dataframe. Drop the minutes remaining.

4. Continuous variables and Category Variables would be processed in the pipeline later.

3.2 Model Selection

The following models will be applied the data set, as well as Ensemble methods.

1. Basic Classification Model
   (a) Logistic Regression
   (b) Linear Discriminant Analysis
   (c) K-Neighbors Classifier
   (d) Decision Tree Classifier
   (e) Gaussian Naive Bayes
   (f) Support Vector Classifier

2. Ensemble Model
   (a) Ada Boost Classifier
   (b) Gradient Boosting Classifier
   (c) Random Forest Classifier
   (d) Extra Trees Classifier
   (e) XGBoost Classifier

3.3 Machine Learning Pipeline

I will explain each step I use in my ML pipeline.

1. Split the entire data set into 20% for the test and 80% for the other.

2. Define a 'StratifiedKFold' split other data set into 5 folders, 4 for Training data and 1 for Cross Validation. Set Random State as 50.

3. Applied Standard Scaler and Onehot Encoder for continuous and category variables perspectively. Using ColumnTransformer save them in preprocessor.

4. Make pipeline with preprocessor and classification methods.

5. Define the parameters we want to tune.

6. Put previous steps together into GridSearchCV function.

7. Fit the Other data set and return the Test data score

8. Run 10 times with different Random State values, return best parameters, test score for each time and return mean of these 10 scores and standard deviation round to 3 decimal.

3.4 Tune Parameters

I tune the different models with different parameters:

Models	Parameter 1	Parameter 2
Logistic Regression	penalty:['l1','l2']	C:np.logspace(-3,3,7)
Linear Discriminant Analysis	penalty:['svd','lsqr','eigen']	shrinkage:[0,0.25,0.5,0.75,1.0]
K-Neighbors Classifier	weights:['uniform','distance']	
Decision Tree Classifier	min samples split:[10,50,100,300,600]	max depth:[1,2,3]
Gaussian Naive Bayes	var smoothing: np.logspace(-11,-7,5)	
Support Vector Machine	kernel:['linear','poly','rbf','sigmoid']	C: np.logspace(-1,1,10)

Table 3.1: Parameter Tuning under Basic Classification Models

Models	Parameter 1	Parameter 2
Ada Boost Classifier	learning rate:[0.001,0.01,0.1,1]	
Gradient Boosting Classifier	min samples split:[10,50,100,300,600]	max depth:[1,2,3]
Random Forest Classifier	min samples split:[10,50,100,300,600]	max features:['sqrt', 'log2']
Extra Trees Classifier	min samples split:[10,50,100,300,600]	max features:['sqrt', 'log2']
XGBoost Classifier	max depth:[1,2,3]	learning rate:[0.001,0.01,0.1,1]

Table 3.2: Parameter Tuning under Ensemble Learning Models

I evaluate the models' performance by the mean and standard deviation of 10 accuracy scores on the given test data and labels with 10 different random state values. The problem here is a binary classification problem, such that predict accuracy would be a very convincing score.

Also, the standard deviation will show fluctuations in the accuracy of the models, and we expect the results of the model to be both accurate and stable.

4 Results

The mean and standard deviation of the accuracy scores from all the models show in the following table:

	Before Tune		After Tune		
Basic Model	mean	sd	mean	sd	improvement
Logistic Regression	0.681	0.003	0.682	0.003	0.15%
Linear Discriminant Analysis	0.681	0.004	0.683	0.004	0.3%
K-Neighbors Classifier	0.600	0.004	0.644	0.004	7.33%
Decision Tree Classifier	0.586	0.004	0.679	0.004	15.87%
Gaussian Naive Bayes	0.547	0.005	0.561	0.002	2.56%
Support Vector Machine	0.679	0.005	0.681	0.005	0.3%
Baseline:	0.5525				

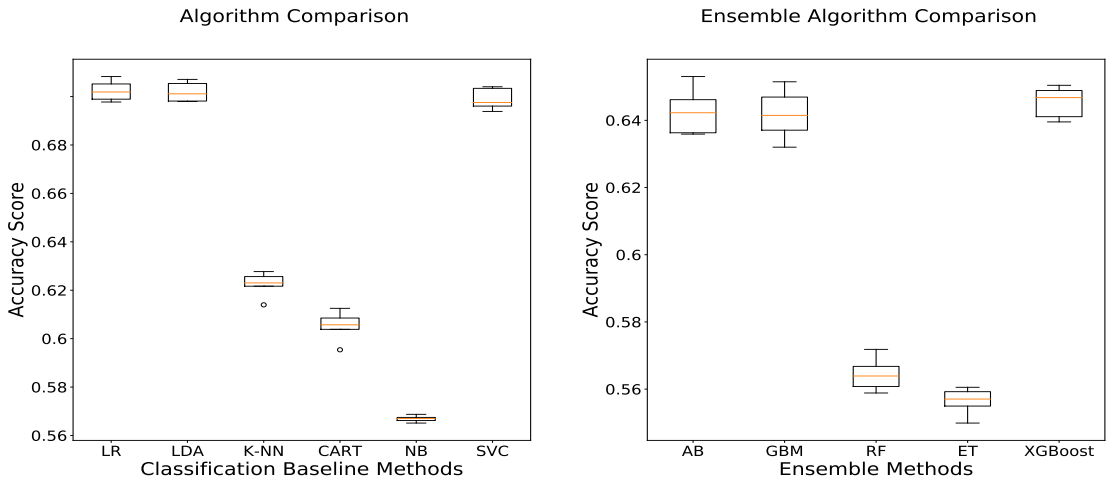
Table 4.1: Mean and Standard Deviation of the accuracy score before and after tuning parameters of basic classification models

	Before Tune		After Tune		
Ensemble Model	mean	sd	mean	sd	improvement
AdaBoost Classifier	0.681	0.003	0.682	0.004	0.15%
Gradient Boosting Classifier	0.681	0.003	0.681	0.004	0
Random Forest Classifier	0.647	0.002	0.68	0.004	5.1%
Extra Trees Classifier	0.638	0.005	0.675	0.006	5.8%
XGBoost Classifier	0.682	0.004	0.683	0.004	0.15%
Baseline:	0.5525				

Table 4.2: Mean and Standard Deviation of the accuracy score before and after tuning parameters of ensemble learning classification models

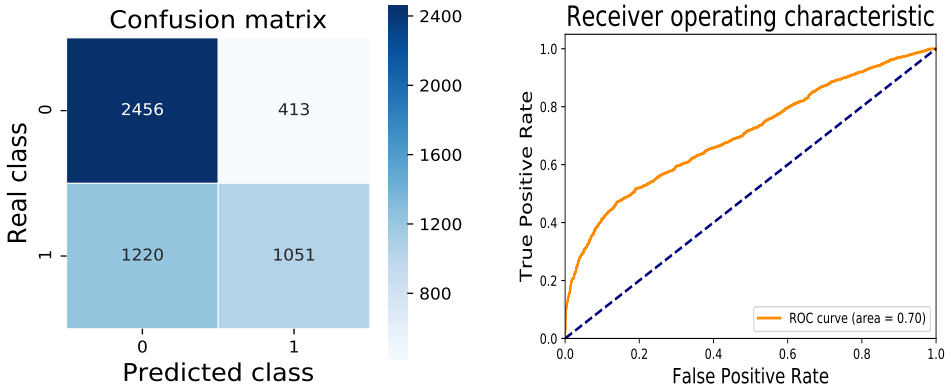
Most of the models have a significant improvement over the baseline which has an accuracy of 0.5525. Next, let's see box plot about tho classification groups performance:

From Table 4.2 and Figure 4.1, we found that the models that performed well are: LR, LDA, SVC, AB, GBM, XGBoost. Surprisingly, LR and LDA are perform impressive and stable. Of course, the result of the decision tree is also excellent, especially AdaBoosting, Gradient Boosting, XGBoosting. Next, I'll take a closer look at XGboosting.



(a) (b)

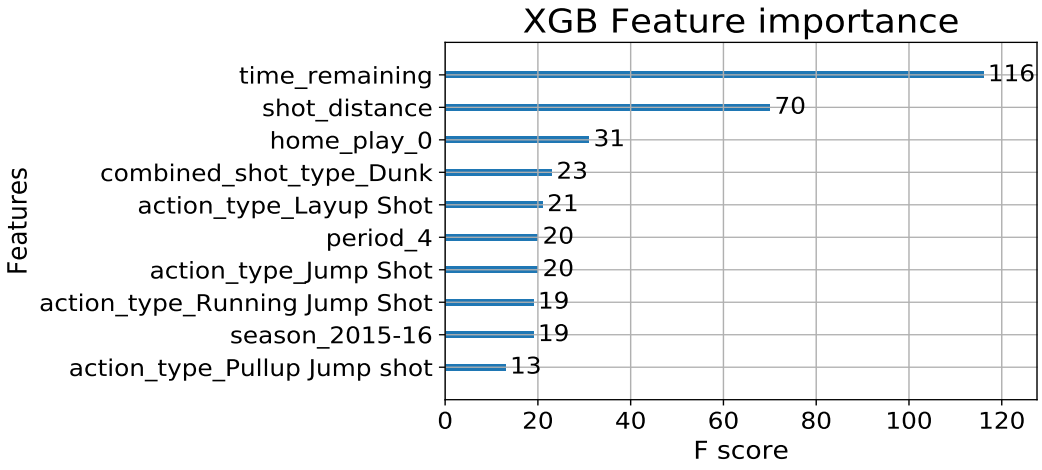
Figure 4.1: (a) Box plot Basic Classification methods performance.
(b) Box plot Ensemble Learning Classification methods performance.



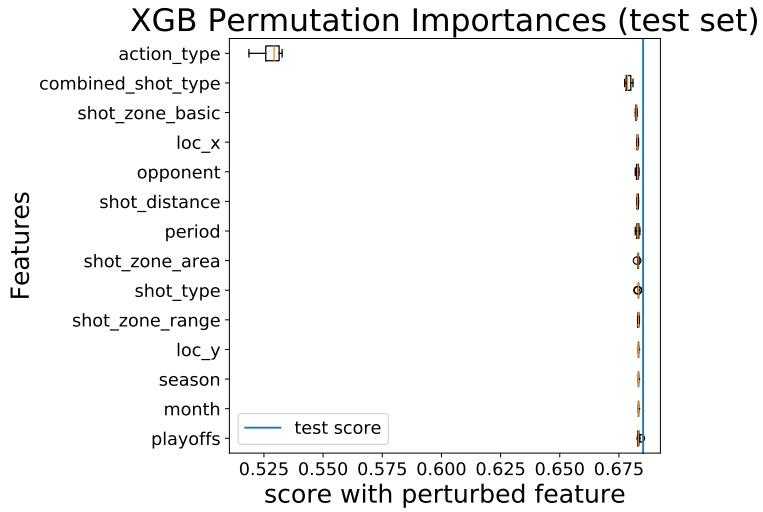
(a) (b)

Figure 4.2: (a) Confusion matrix on XGBoosting. (b) ROC curve on XGBoosting.

According to confusion matrix, we get a high number of false negatives(1220).



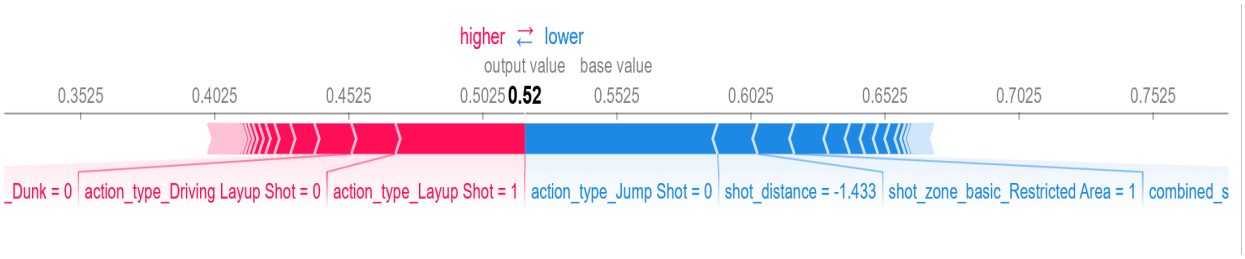
(a)



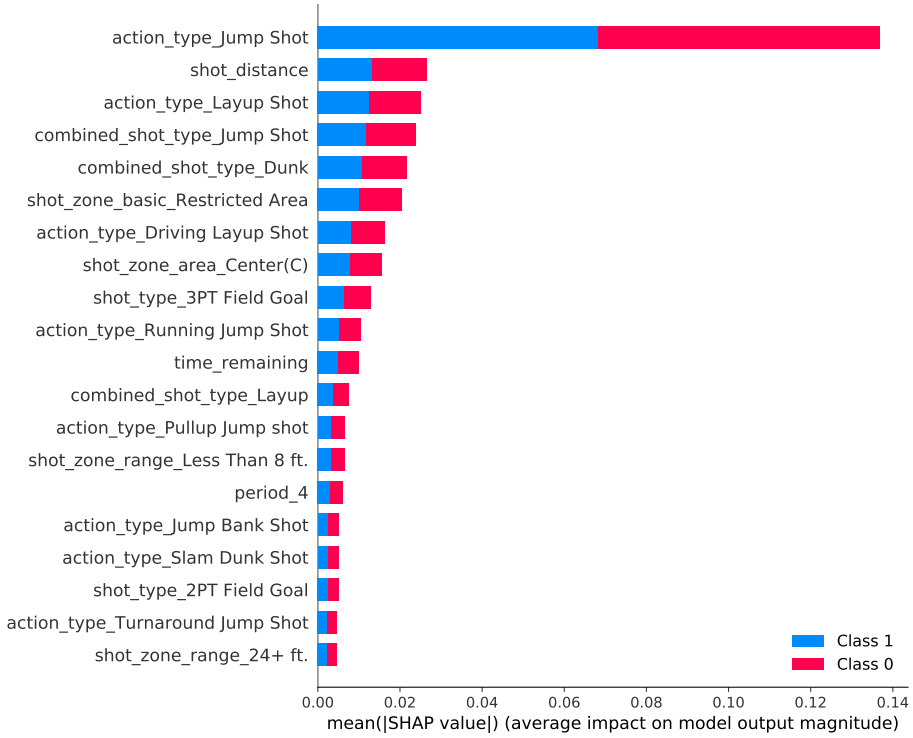
(b)

Figure 4.3: (a) XGBoosting feature importance based on F score.
(b) XGBoosting permutation importance based on score with perturbed feature.

Two results are quite different. Only two continuous rank top two on the F score test table. Action type occupies an absolutely important position, and all the remaining variables are not that important base on the permutation importance plot.



(a)



(b)

Figure 4.4: (a) Random Forest shap local on index 3 force plot
(b) Random Forest shap global features bar plot

By the Figure 4.4 (a), the output value is the prediction for that observation (index=3). Features that push the prediction higher (to the right) are shown in red, and those pushing the prediction lower are in blue. Figure 4.4 (b), the top variables contribute more to the model than the bottom ones and thus have high predictive power, and action type Jump Shot contribute way more than the other features, which is most powerful weapon why Kobe dominant the league.

5 Outlook

This prediction could still be improved in many ways. Aside from hyperparameter tuning of the models, there were other potential things to try:

- Test XGBLinear and Random Forest (caret) model.
- Test more parameter combination on XGBoosting model
- Add log loss into consideration to judge the models
- Find some data regarding defender since this dataset doesn't have variables explaining it.

6 References

- L. Breiman, J.H. Friedman, R.A. Olshen, and C.J Stone. Classification and Regression Trees. Wadsworth, Belmont, Ca, 1983.
- Danesi Ivan Luciano, Perugini Federica, Data Exploration for Data Science, EDUCatt, Milano, 2018.
- Eric Eaton, Classification: Decision Trees and Overfitting, Georgia Tech, 2016.
- Gareth James, Daniela Witten, Trevor Hastie and Robert Tibshirani, An Introduction to Statistical Learning, Springer, London, 2017.
- Kaggle, https://www.kaggle.com/c/kobe-bryant-shot-selection/data, Dataset, 2016.
- Terry Therneau, Beth Atkinson, Brian Ripley, Package 'rpart', CRAN, 2018.
- Terry Therneau, Beth Atkinson, An Introduction to Recursive Partitioning Using the RPART, routines, Mayo Foundation February, 2018.