DATA MINING PROJECT BASKETBALL SHOT PREDICTION

ADITYA SETHI [07]
CHETAN VERMA [18]
GAJENDER [62]

TIMELINE

- Problem Statement
- Data Source and Overview
- Data Preprocessing
- Feature Selection
- Models
- Insights
- Conclusion

PROBLEM STATEMENT

The national basketball association (NBA) is a professional basketball league in North America. Total number of 30 teams participate in this tournament with the player count of 600+. **It becomes hefty and time consuming task for coaches and managers of a particular team to evalutate performances of each and every player** therefore we can design certain models to predict the whether the player will be able to make a shot successfully or not based on certain features. Predicting the accuracy of shot would intern give us a brief of performance of player and hence the data can be utilized by team staff during selection of player.

(A PROJECT THAT AIMS TO TACKLE THE CHALLENGE OF PREDICTING A PLAYER'S SHOT ACCURACY TAKING IN CONSIDERATION CERTAIN FACTORS AFFECTING THE SHOT MAKING ABILITY.)

DATA SOURCE

Dataset contains shots taken during the 2014-2015 season. Dataset is scraped from NBA's REST API. Link for the dataset: https://www.kaggle.com/dansbecker/nba-shot-logs

Our dataset initially had 21 features out of which few were shortlisted based on their importance for prediction.

DATASET OVERVIEW

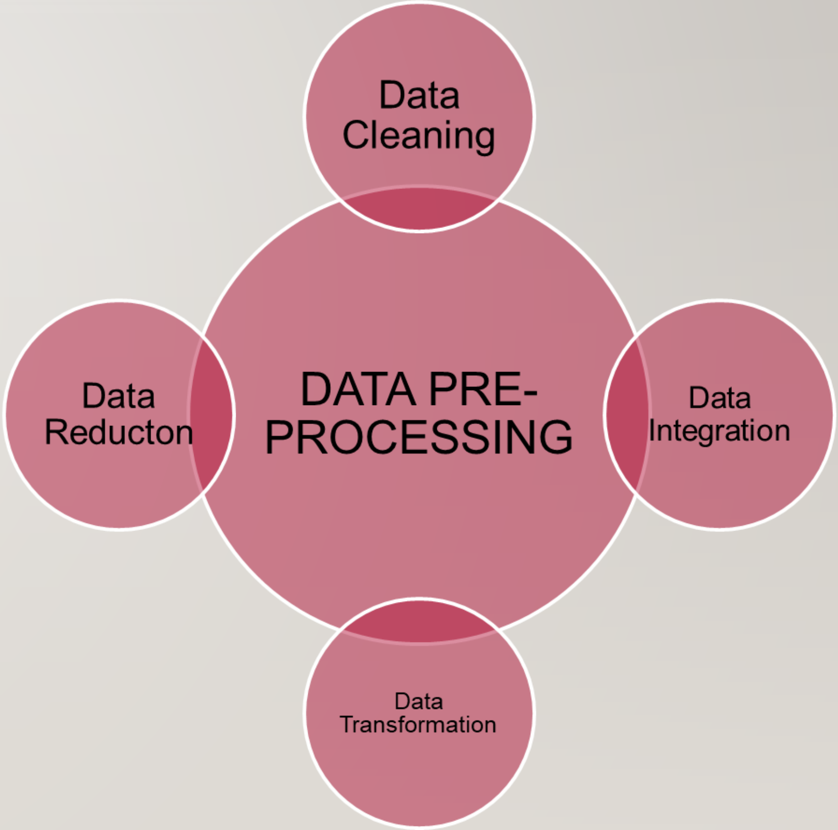
Our dataset as shown in the table initially has 21 attributes along with which number of observations are shown :

GAME_ID	904
MATCHUP	1808
HOME_AWAY	2
W	2
FINAL_MARGIN	88
SHOT_NUMBER	38
PERIOD	7
GAME_CLOCK	719
SHOT_CLOCK	242
DRIBBLES	33
TOUCH_TIME	238
SHOT_DIST	448
3PTS_SHOT	2
SHOT_RESULT	2
CLOSEST_DEFENDER	473
CLOSEST_DEFENDER_PLAYER_ID	474
CLOSE_DEF_DIST	299
FGM	2
PTS	3
PLAYER_NAME	281
PLAYER_ID	281

DATA PRE- PROCESSING

Data preprocessing is a crucial step in the data mining process, encompassing the cleaning and transformation of raw data to render it suitable for analysis. The goal of data preprocessing is to improve the quality of the data and to make it more suitable for the specific data mining task.

Data Preprocessing consists of 4 parts i.e. Data Cleaning, Data Reduction, Data Integration and Data Transformation



DATA CLEANING

Data cleaning is the process of fixing or removing incorrect, corrupted, incorrectly formatted, duplicate, or incomplete data within a dataset. In our project we performed Data Cleaning to improve the accuracy of our model.

We used isnull().sum() command to get the count of missing values in each column.

```
df.isnull().sum()
```

```
GAME_ID                        0
MATCHUP                        0
LOCATION                       0
W                              0
FINAL_MARGIN                   0
SHOT_NUMBER                    0
PERIOD                         0
GAME_CLOCK                     0
SHOT_CLOCK                  5567
DRIBBLES                       0
TOUCH_TIME                     0
SHOT_DIST                      0
PTS_TYPE                       0
SHOT_RESULT                    0
CLOSEST_DEFENDER               0
CLOSEST_DEFENDER_PLAYER_ID     0
CLOSE_DEF_DIST                 0
FGM                            0
PTS                            0
player_name                    0
player_id                      0
dtype: int64
```

The output indicates that the SHOT_CLOCK attribute contains 5567 missing values.

In order to remove the null values from the SHOT_CLOCK attribute, we replaced the null values with the mean of the attribute. We used the command :

```
df.SHOT_CLOCK = df.SHOT_CLOCK.fillna(df.SHOT_CLOCK.mean())
```

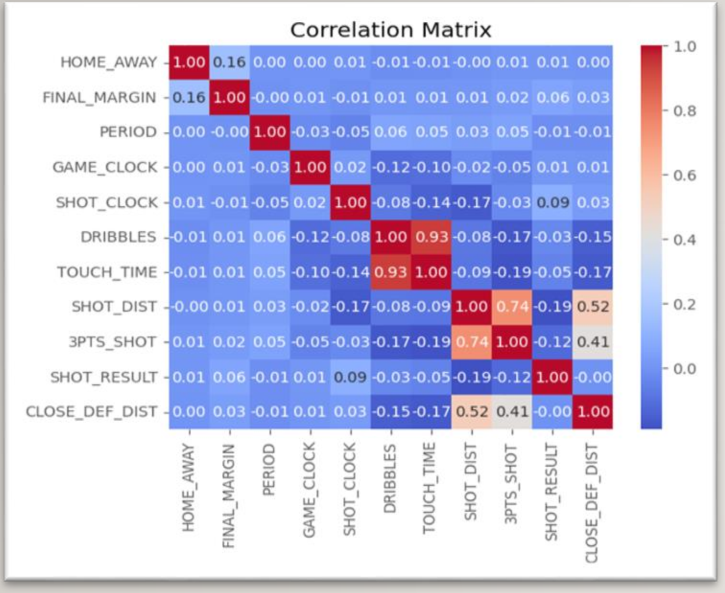
Apart from replacing null values with mean value in case of Shot Clock, we replaced the negative values in Touch Time by its mean. Furthermore, for values surpassing the maximum limit of 24 seconds, we set them to the maximum limit.

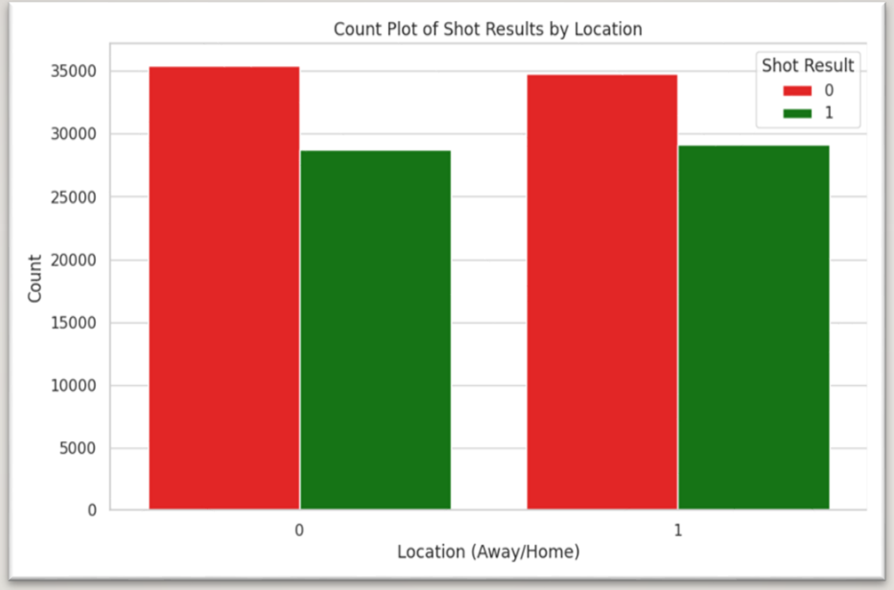
```
[ ]  len(df.TOUCH_TIME[df.TOUCH_TIME<0])

     312

[ ]  df.TOUCH_TIME[df.TOUCH_TIME<0] = df.TOUCH_TIME.mean()

[ ]  len(df.TOUCH_TIME[df.TOUCH_TIME>24.0])

     4

[ ]  df.TOUCH_TIME[df.TOUCH_TIME>24.0] = 24
```

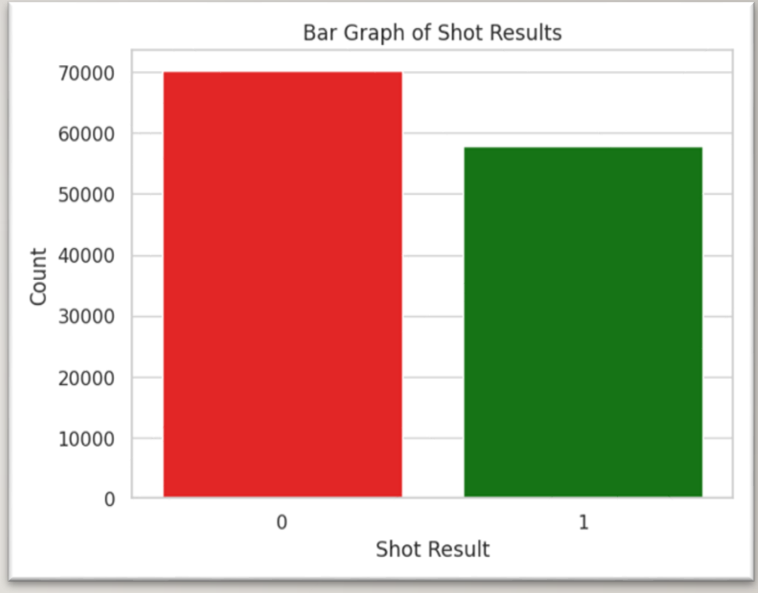
Along with this, Timestamp of Game clock was normalized. Earlier it was in the format of minutes preceded by seconds. Its values were changed into only seconds.

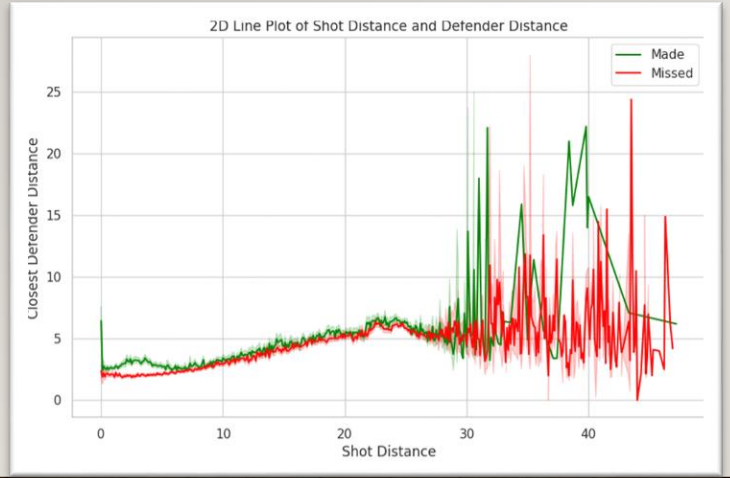
```
[ ]  df.GAME_CLOCK = df.GAME_CLOCK.apply(lambda x: int(x.split(":")[0])*60 + int(x.split(":")[1]))
```

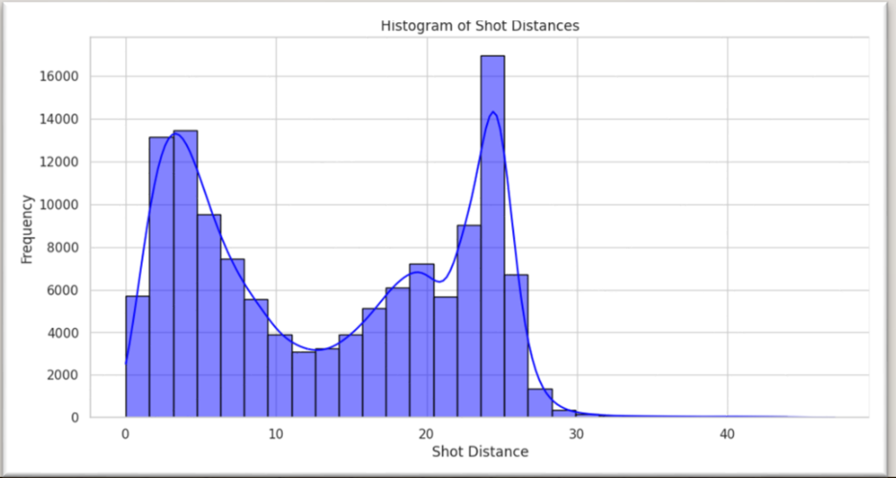
DATA VISUALIZATION

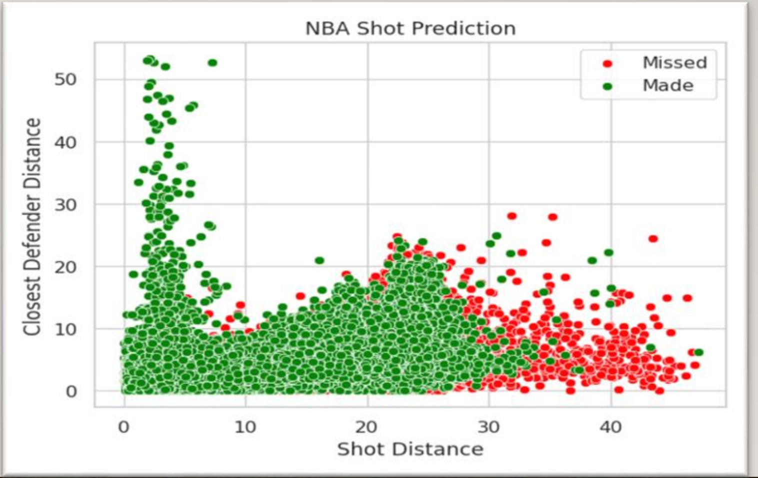




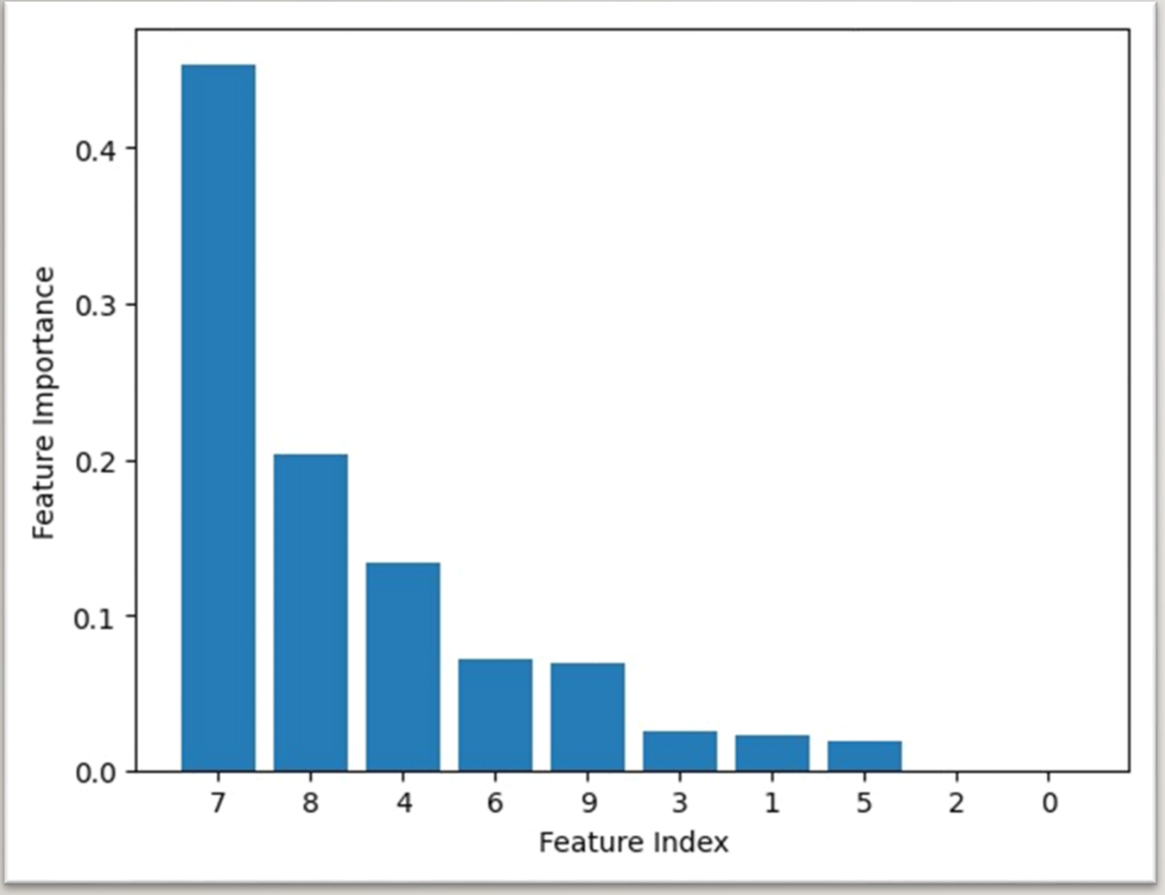








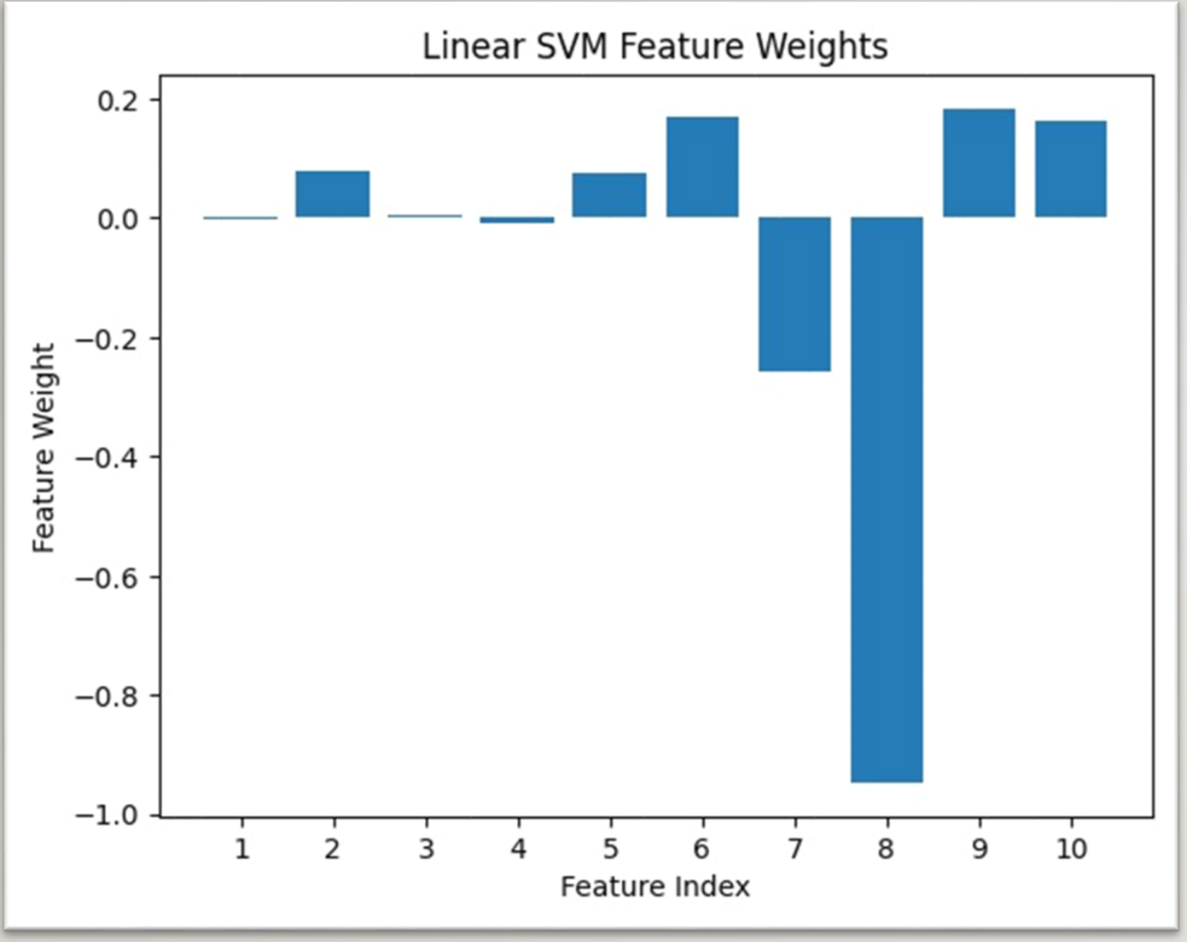
Upon using the inbuilt libraries of Python, we have computed feature importance for 3 different models :



Feature Importance of Random forest

We got the most important features using **<u>feature_importance()</u>** function for random forest and then trained the model with 5 most predictive features

7 denotes Shot Distance
8 denotes 3 Pts Shot
4 denotes Shot Clock
6 denotes Closest Defender’s Distance
9 denotes Touch Time

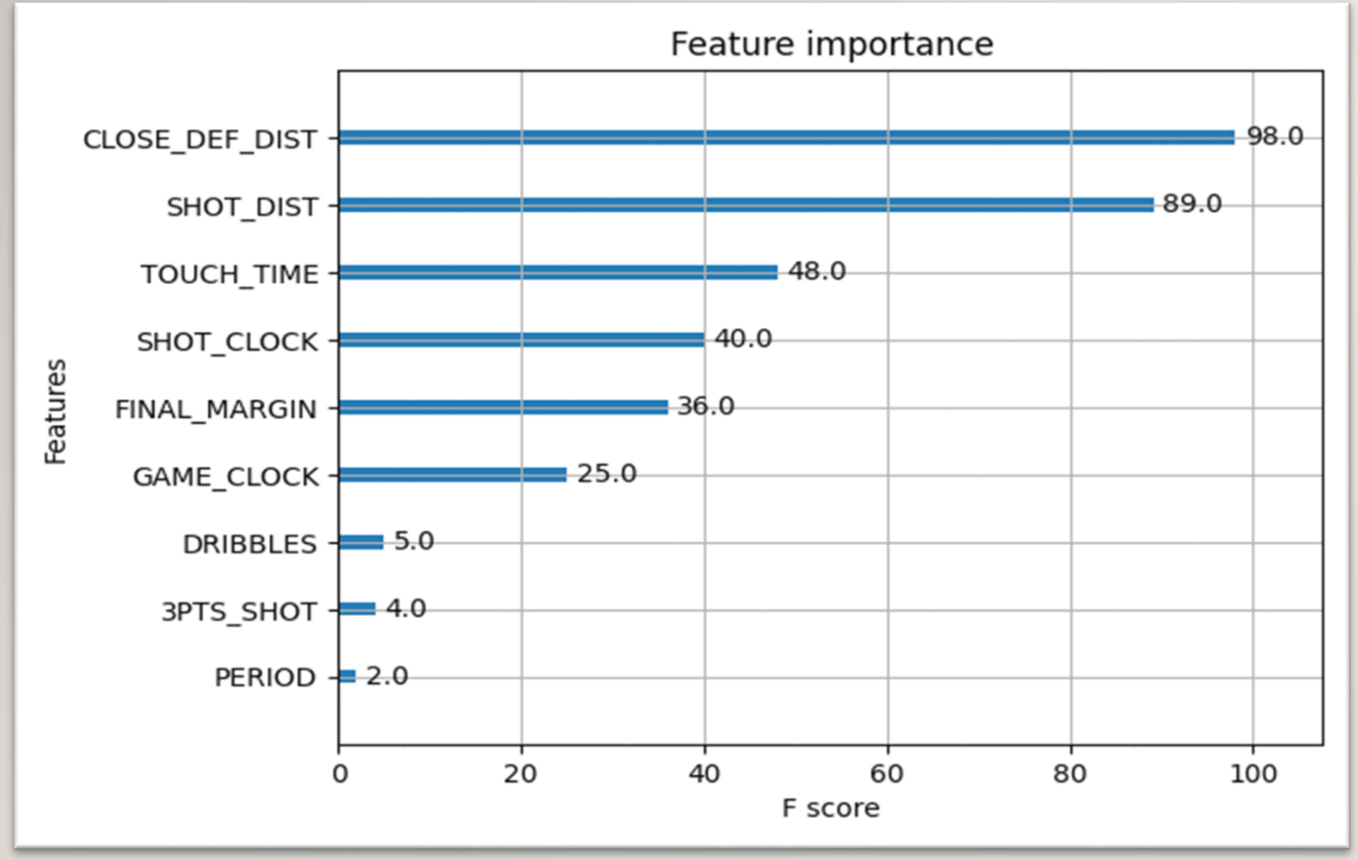


First, we trained our model using all the features then saw the importance of all the features **using coef attribute** and trained the model again with the five most predictive features (highlighted)

1 Denotes Home_away
2 Denotes FINAL MARGIN
3 Denotes PERIOD
4 Denotes GAME_CLOCK
5 Denotes SHOT CLOCK
6 Denotes DRIBBLES
7 Denotes TOUCH TIME
8 Denotes SHOT DISTANCE
9 Denotes 3 PTS SHOT
10 Denotes CLOSEST DEFENDER'S DISTANCE

Negative feature weight indicates an inversely proportional relationship between feature index and feature weight. For example, for feature no. 8 (Shot distance), increase in distance will result in decrease in success rate of shot made i.e. negative class (missed = 0) is more.

Feature importance for XGBoost



we visualized the importance of each feature using the **plot_importance() function** provided by the XGBoost library.

we decided to build our training and testing dataset using the five most predictive features which included:

Shot distance
Closest defender's distance
Shot clock
Final margin
Touch time.

Models used for making predictions

There is a long list of methods that can be used to predict accuracy of the shots attempted by a player that includes Logistic regression, SVM, Neural Network, Random Forest, Xgboost, Naive Bayes etc.

In our project, we have used 3 models to conduct a comparative study, aiming to analyze differences in accuracy. Models that we have used are

- Random Forest
- SVM
- XGBoost

Models were assessed by accuracy, processing time, confusion matrix and RoC Curve

Random Forest

The Random Forest model scored an accuracy of approximately 61%.

	RANDOM FOREST		
	Predicted P	Predicted N	
Actual P	11539	2597	14136
Actual N	7310	4168	11478
	18849	6765	

CONFUSION MATRIX FOR RANDOM FOREST

Method	Accuracy	Processing Time (in seconds)
Random Forest	0.61	2.9

XGBoost

XGBoost model outperformed other models by achieving an accuracy of approximately 62%, as indicated in the accompanying table.

	XGBOOST		
	Predicted P	Predicted N	
Actual P	12126	1977	14103
Actual N	7641	3870	11511
	19767	5847	

CONFUSION MATRIX FOR XGBOOST

Method	Accuracy	Processing Time (in seconds)
XGBoost	0.62	0.652

SVM

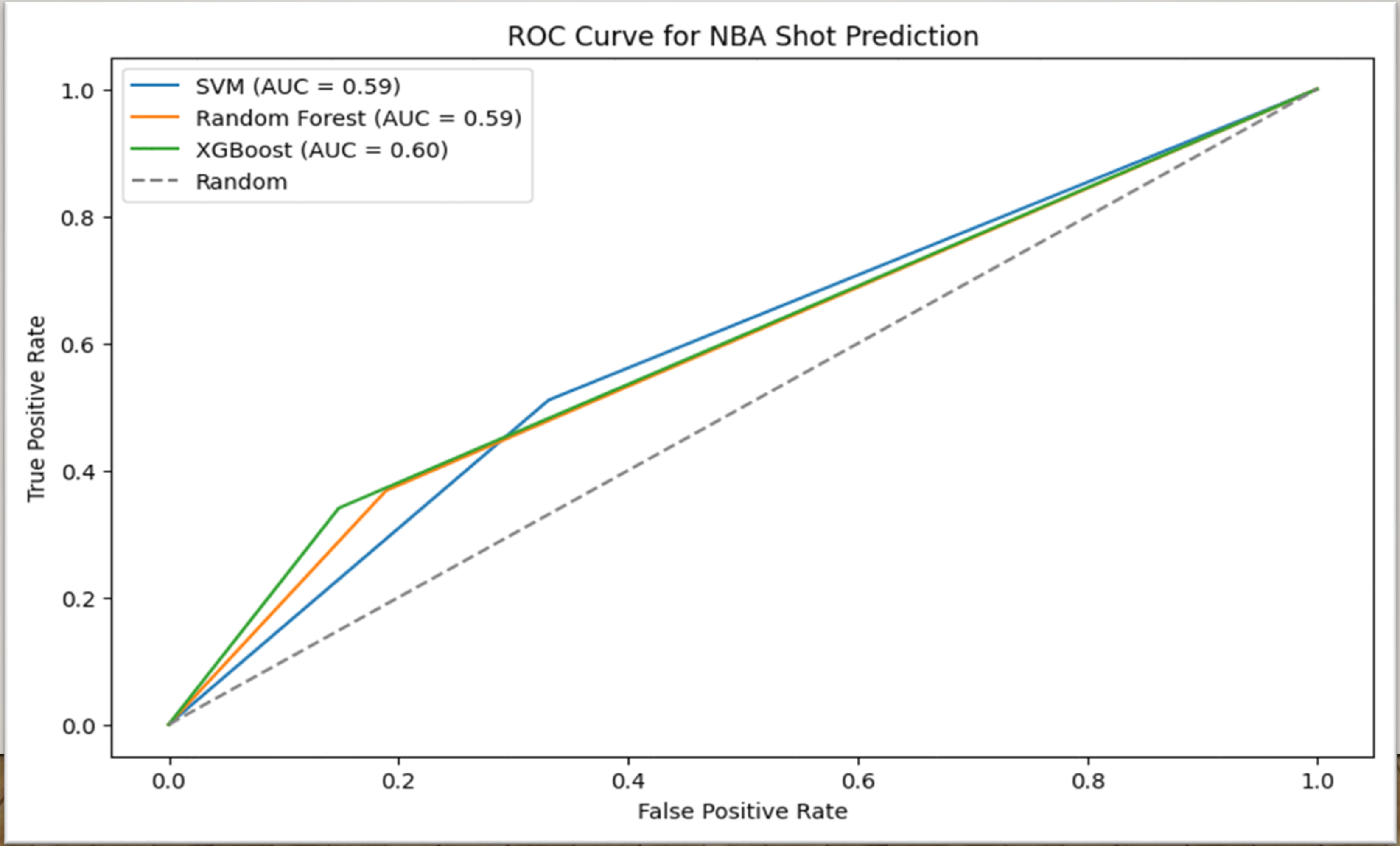
The SVM model scored an accuracy of approximately 59%.

	SVM		
	Predicted P	Predicted N	
Actual P	9250	4938	14188
Actual N	5413	6013	11426
	14663	10951	

CONFUSION MATRIX FOR SVM

Method	Accuracy	Processing Time (in seconds)
SVM	0.59	665

RoC Curve for SVM, Random Forest & XGBoost Models



CONCLUSION

In this investigation, SVM, Random Forest, and XGBoost models were utilized, with XGBoost emerging as the top-performing model, showcasing the highest accuracy rate. Nevertheless, the Random Forest model also demonstrated high efficacy as a classifier for the dataset. Considering the inherent challenges in analysing behavioural data and the limited available features, achieving an accuracy of approximately 60% is noteworthy. It's crucial to recognize the intricacies involved in the shooting process, where factors such as emotional states, subtle balance variations, or minor deviations can significantly influence the shot's outcome. Given these unpredictable elements, anticipating accuracy rates in the range of 80-90% appears unrealistic.

REFERENCES

[1] M. Harmon, P. Lucey, and D. Klabjan. Predicting shot making in basketball learnt from adversarial multiagent trajectories. https://www.researchgate.net/publication/356727108

[2] Maram Shikh Oughali, Sahar A. El Rahman, Mariah Bahloul. Analysis of NBA Players and Shot Prediction Using Random Forest and XGBoost Models. https://ieeexplore.ieee.org/abstract/document/8716412

[3] Brett Meehan, Stanford University. Predicting NBA Shots. https://cs229.stanford.edu/proj2017/final-reports/5132133

[4] https://www.geeksforgeeks.org/xgboost/
[5] https://www.geeksforgeeks.org/random-forest-classifier-using-scikit-learn/
[6] https://xgboost.readthedocs.io/en/latest/get_started.html#python