Python - NBA Shooting Project

Objective

Based on a large NBA dataset describing all plays from 2020-2021 season, the main objective was to **predict if a shot would be made** (or not) based on numerous plays features.

Scheme

Principal dataset is sent by a private third-party. After collecting the dataset, a **cleaning phase** was needed to eliminate bad quality data and **new independent variables were created** to include them as inputs in ML algorithms. All features can be found in the df_cleaned_shots.csv document on Line A. Their name makes it easy to understand the meaning of a variable.

A brief **Exploratory Analysis is conducted** before comparing predictions and accuracies of two different ML supervised binary classification algorithms (**Random Forest, Gradient Boosting**).

Data Description

The dataset received contains information regarding all plays from the 2020-2021 season (shots, fouls, rebounds, turnovers…). It consists in 541,348 samples (individuals) and 44 features (variables). The object of this research is to focus only on shooting plays (shot is made or not) and features having a potential impact on shooting results. Therefore, after filtering and eliminating bad quality and non-necessary data (missing values mainly), **the final dataset is reduced to 201,524 individuals**.

The **main features gather information on WHO the shooter is** (age, experience, position, points in the game before taking the shot, points in the season before the game), **WHEN the shooter shoots** (quarter, time of the shooting sequence, elapsed time since the beginning of the game) and **WHERE the shot is taken** (shot distance, shooting areas). After the feature engineering process, **the final dataset consists in 62 independent variables and one dependent variable (shot is made or not).**

It is to be noticed that defensive plays data (distance between defensive player and shooter, who is the direct defensive player, …) are not included which is unfortunate as they are main features to explain the difficulty of a shot taken.

Exploratory Data Analysis

Graph 1 shows the distribution of shots taken by positions. Clearly, **Guard (PG and SG) players are those who shoot the most and by far** despite the fact that they are **less efficient than others** (less than 45% for the Field Goal %). **Centers and Power Forward are the most efficient players** (with 55% of efficiency and 48% of accuracy respectively) **but their amount of shots taken are inferior**. In the meantime, it is to be underlined that Small Forwards (SF) are players who clearly shoot less than others, supposing that their main role is not on the scoring side. **Therefore, we can imagine that teams prefer shooting from three points spots or attacking the circle coming from outside the paint as Guards tend to take much more shots than players remaining close to the basket and in the paint.**

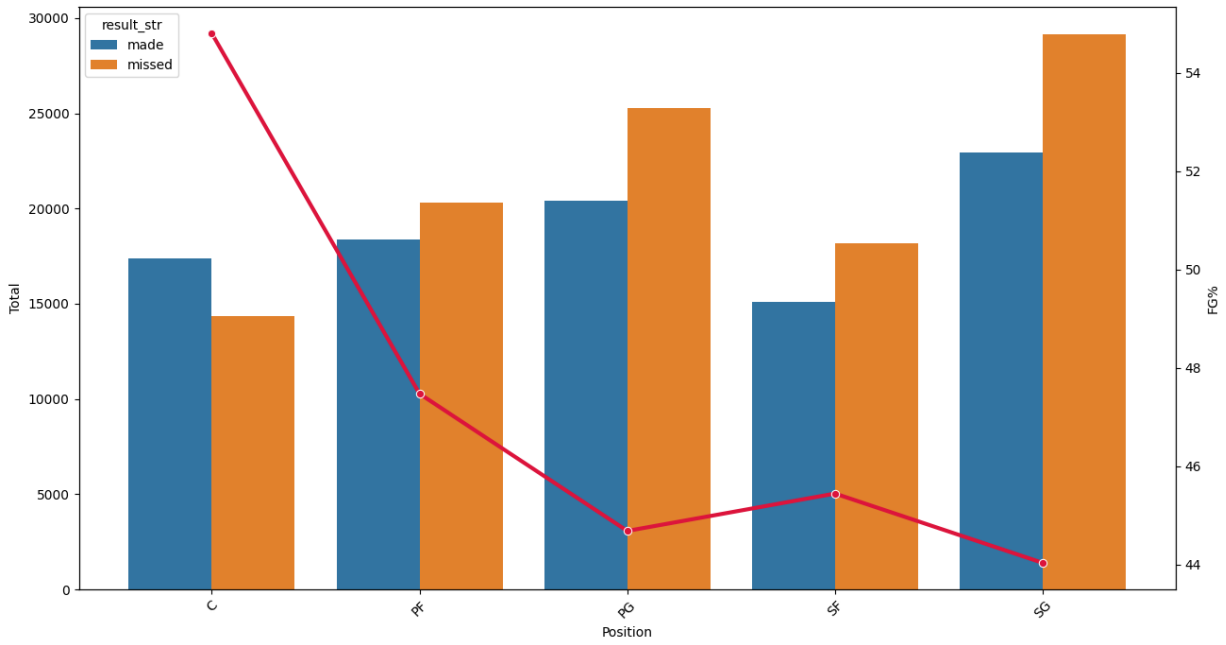

Fig 1 – Distribution of NBA shots by position

It was also important to study the role, through time, attributed to each type of player. I decided to conduct an analysis aiming at visualizing who make shots and when. The following graph 2 shows curves displaying numbers of made shots by position and through different time intervals. What can be highlighted through this graph is the volatile PG players' curve. Indeed, it contains four peaks, each of them appearing at the end of every quarter of an NBA game. **We can conclude that those players take their responsibilities and are central pieces when it comes to finish a quarter by making more shots**. While no important information is extracted from SG and PF curves, **it is to be noted that C and SF curves tend to decrease in an important way during the second half.**

It is clear that the most used areas are areas within the paint (under the circle and short paint shot) for all players combined, it was interesting to study the distribution of shots for less used areas such as 2pts and 3pts spots outside of the paint. **The heatmap fig 3-b demonstrates that, excluding Centers, the most used spots are those facing the basket circle and behind the 3pts line**. Indeed, PG, SG, SF, PF players tend to prefer shooting from 3pts Top Left, Top Right and Middle areas in comparison with other areas. As seen in fig 3-a, despite having a better accuracy percentage, **2 pts areas outside the paint are significantly underused, showing that nowadays, players focus on shooting from long distance.**

In the Jupyter scipt, an interactive tree map has been realized and indicates the FG% per areas and positions and highlights the overused areas or underused areas. In annex, a screenshot of such a graph has been pasted.

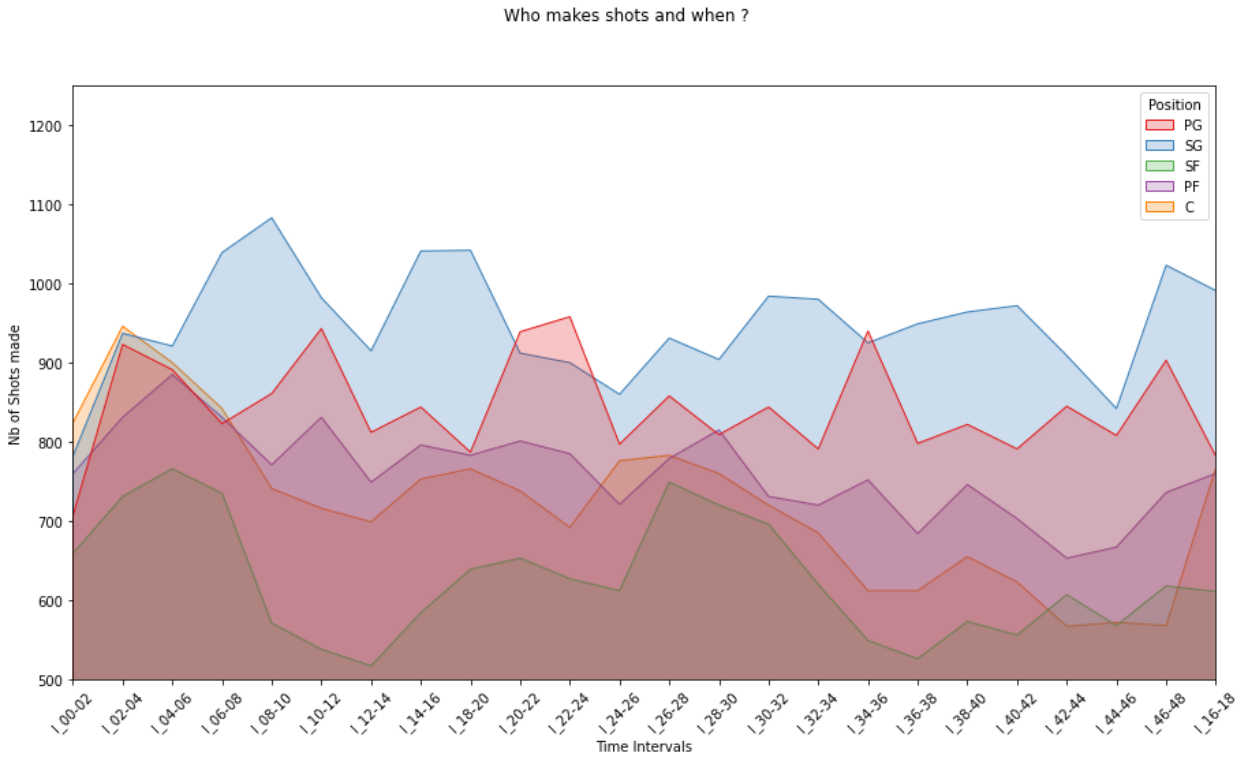

Fig 2 – Time series of MADE shots per position

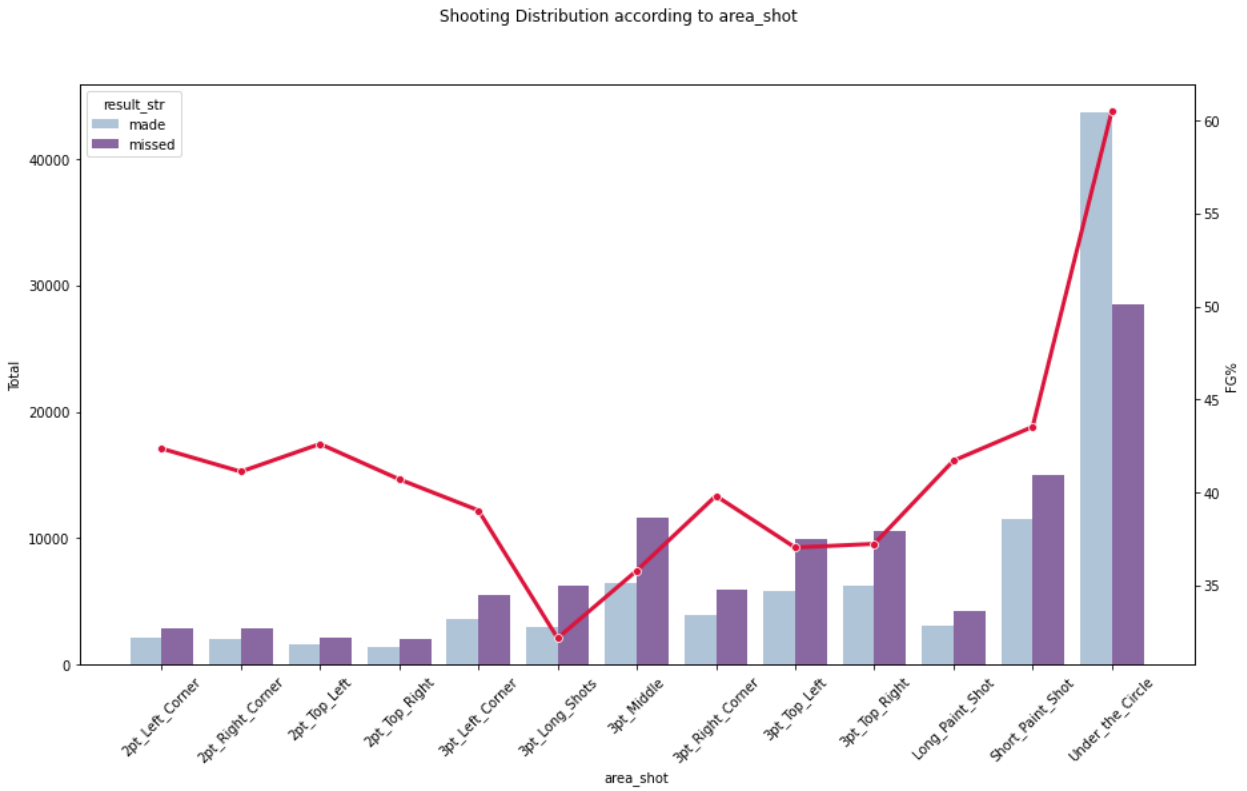

Fig 3-a – Shooting distribution and FG accuracy depending on shooting areas

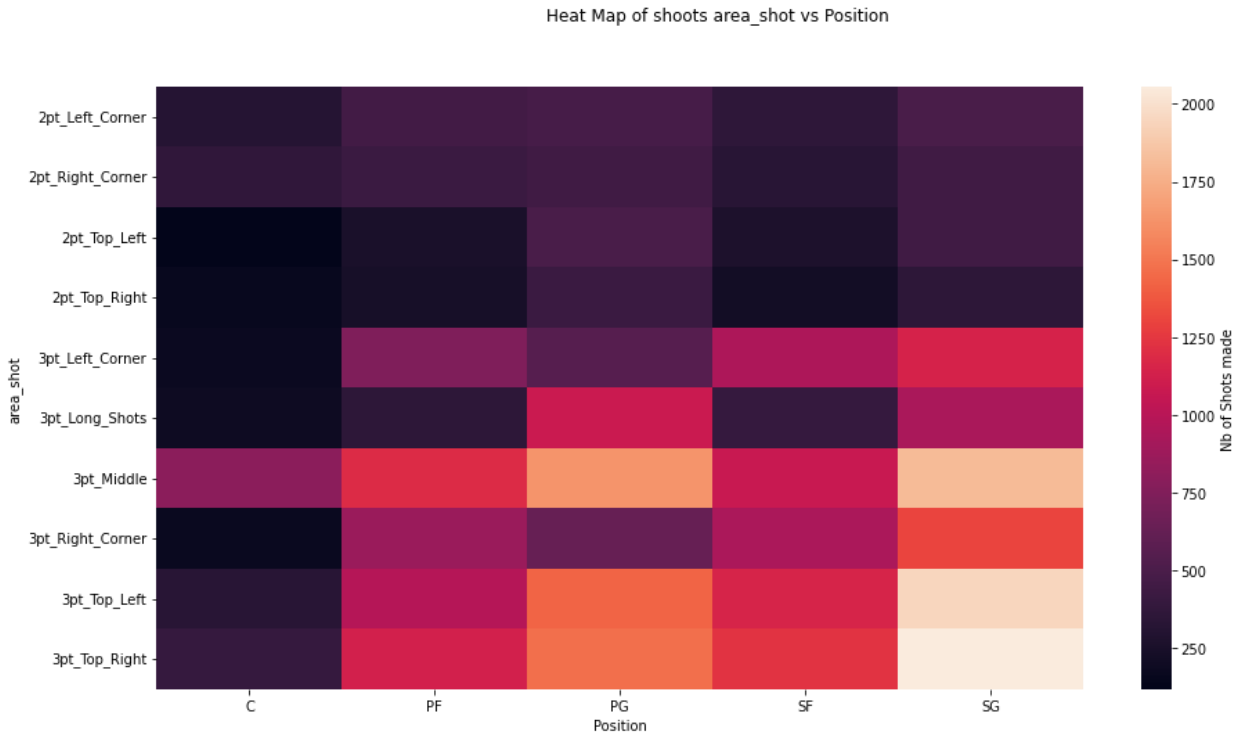

Fig 3-b – Distribution of total MADE shots by position & shooting area

Machine Learning Predictions

Before running an algorithm model, it is important to check if there is a high level of collinearity among numerical values. If we find a solid correlation (>0.7) between two variables, we can delete one of them as they display the same amount of information. Moreover, deleting one variable enables to reduce the dimensionality of the dataset, this latter being high here with more than 60 features.

We constate that the only strong correlation observable is related to the number of shots made in a game , the amount of total shots taken during the same game and the total number of points made. It is normal to find a strong relations among those variables and one can delete one of those variables. Nevertheless, I decided to keep all of them and will check the importance of those features later.

This correlation study excludes time intervals or Position as they consist of string variables converted into dummies variables later.

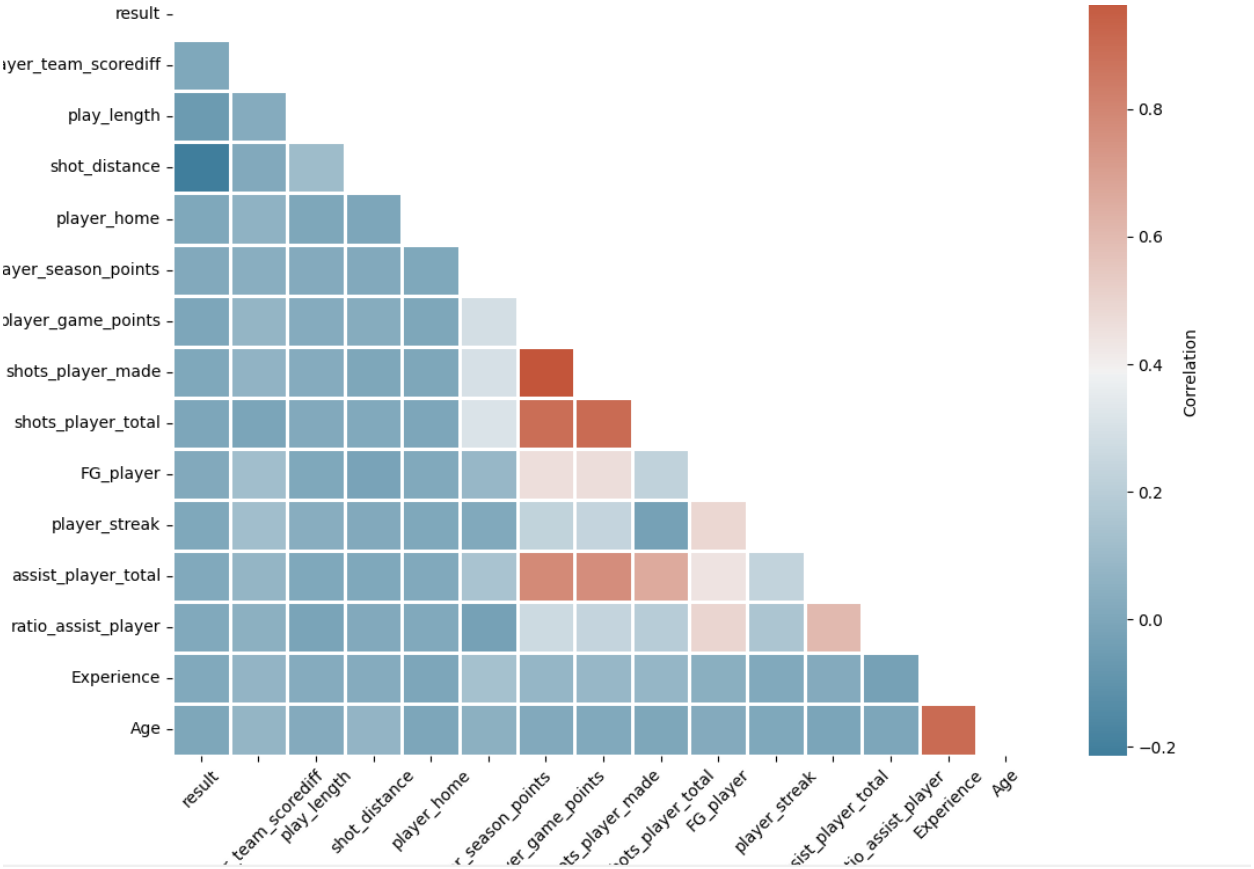

Fig 4 – Correlations among variables

Only two algorithms were considered to predict the result of a shot (binary prediction). Considering the number of features who is really high (62 including dummy variables), a **random forest** was chosen to reduce the variance by training on different samples of the data (bagging method). This algorithm is compared with the **XG Boosting algorithm**. Unlike the random forest taking into account decisions trees (estimators) independently from others, XG Boosting is still based on decisions trees but each estimator will be trained by learning from the predictions errors from the previous one. Therefore, it was interesting to compare those two methods. For each algorithms, optimizations of hyperparameters were done but results were almost the same that those using hyperparameters set by default. Optimization processes are accessible within Python code.

To compare those algorithms we use **confusion matrices,** classifying the actual and predicted values as 'made' or 'missed' shots. Results are shown in the below matrices.

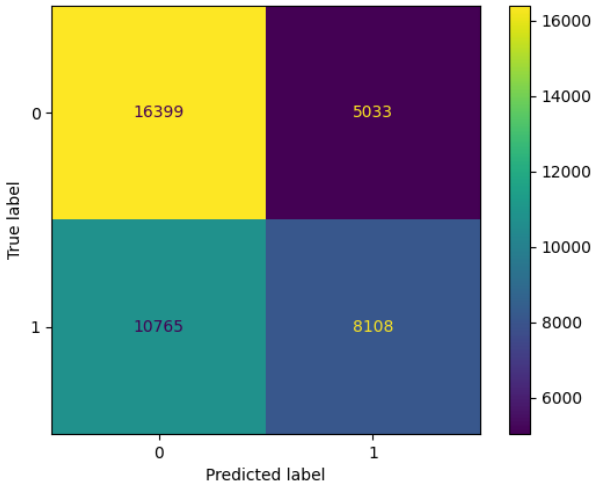

Fig 5-a – Confusion matrix of a Random Forest

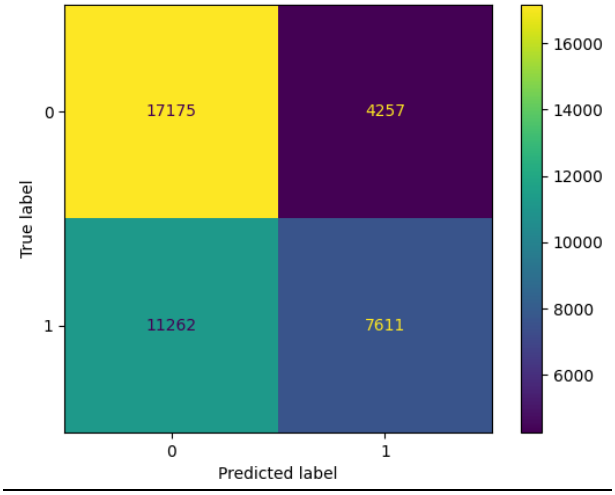

Fig 5-b – Confusion matrix of a XG Boost

MISSED SHOTS	Random Forest	XG Boost
Recall (TP / (TP + FN))	0.77	0.8
Precision (TP / (TP + FP))	0.6	0.6
F1 (2*Recall*Precision/(Recall+Precision)	0.67	0.69
MADE SHOTS	Random Forest	XG Boost
Recall (TP / (TP + FN))	0.43	0.4
Precision (TP / (TP + FP))	0.62	0.64
F1 (2*Recall*Precision/(Recall+Precision)	0.51	0.5

Fig 6 – Comparisons of predictive scoring

Conclusion

In addition to the above grid results, **the general accuracy scores are 60.8% for random forest and 61.2% for XG Boost making them relatively equivalent regarding the results**. Missed and made shots are relatively well balanced in the outcome tested (y_test in the code) and it seems that **models have a bigger**

issue regarding the classification of truly made shots while it classifies missed shots with much more accuracy (see fig 6).

Reducing the number of measures and finding collinearity using a dimension reduction analysis (PCA) might be a good project to test the algorithms with less complex dataset. On the opposite, we could add features as significant variables were not included in this project as they were missing within original dataset. But including features regarding the shooting defense (distance between shooter and defensive player, shooter is taller or smaller than defensive player, …) appear to be interesting to build a more stable, accurate and reliable algorithm…

Analysis of players efficiencies per area