# Machine Learning Project - Basketball Dataset

Christian Faccio

Course of A.A. 2024-2025 - Introduction to Machine Learning

***Abstract*— Machine learning is the process of enabling computers to learn and make decisions or predictions from data without being explicitly programmed. A learning technique can be applied to every kind of data, possibly helping humans to discover unseen patterns under the hood. In this project, I apply my knowledge in this field, tackling the challenge of predicting the ratio of wins to total games played by a team for a season, considering only the data available at the end of the previous one.**

## I. Problem Statement

The problem taken into account has the objective of predicting the ratio of wins to the total games played by a team during a season. This variable is continuous and will later be called the *label*. A list of NBA statistics is used to reach the goal, all referring to a single team and a single season, reported in Table I and later called the *predictors*. The data set is taken from Kaggle (games.csv) and contains only the *predictors*, while the *label* is engineered starting from the dichotomous variable HOME_TEAM_WINS, which is 1 if the team that played the match at home won and 0 otherwise. The *label* is basically the sum of the wins at home and abroad, divided by the total number of matches played during that season. Its domain is $[0, 1]$.

| NBA Statistic | Domain | Description |
|---|---|---|
| PTS | $\mathbb{N}$ | Number of points scored. |
| FG_PCT | $[0, 1]$ | Field goal percentage. |
| FT_PCT | $[0, 1]$ | Free Throw Percentage. |
| FG3_PCT | $[0, 1]$ | Three Point Percentage. |
| AST | $\mathbb{N}$ | Number of assists. |
| REB | $\mathbb{N}$ | Number of rebounds. |

TABLE I
PREDICTORS[1]

[1]To be precise, the statistics defined in the domain $\mathbb{N}$ have a smaller domain since the time of a match is not infinite, but is not well defined given that new records can always be estabilished.

The objective is then to learn some models using a learning function and using them in the prediction function to obtain the final predictions.

## II. Assessment and performance indexes

This is a regression problem, so the performance index consists of an error between each predicted value and the relative real one. There exist many such indexes, but for simplicity and clarity, I have chosen the Mean Absolute Error (MAE):

$$MAE = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|$$

The assessment procedure, instead, consists of a 10-fold cross-validation (CV) over the entire data set, giving as output an array of 10 MAE values for each learning technique used.

## III. Proposed solution

Since in this problem there is a *label* that can be used to compare the predictions, I trained a list of five supervised learning techniques on the pre-processed *predictors*:

- *Dummy Regressor*
- *Decision Tree Regressor*
- *Random Forest Regressor (RF)*
- *k-Nearest Neighbor Regressor (kNN)*
- *Support Vector Machines Regressor (SVR)*

The first always predicts the mean value on the set of *predictors* and has been used as a baseline for performance assessment. To guarantee a robust evaluation, the models that overcome the Dummy Regressor in terms of performance have been optimized using Grid-Search with different hyperparameter combinations. The procedure consisted of

a 10-fold cross-validation for each combination, giving as output the one with the lower MAE mean.

Since the domain of the variable is finite and the learning techniques used do not limit the range of the predictions, one should consider using a learning technique like the Logistic Regression or set an *if* condition that sets the value to the nearest boundary if predicted outside the range $[0, 1]$. This can happen if the real values of the *label* are near the extreme values $0, 1$ and are unlikely to happen. Anyway, for this problem, I set the *if* condition to avoid it, even if this could lead to a slight decrease in the error rate.

## IV. Experimental evaluation

### A. Data

The data used for this project can be found on Kaggle in the **games.csv** data set. Since the original dataset contains the statistics referred to every match disputed in every season, some feature engineering has been done on it, obtaining a "grouped data set" with end-of-season values (always for each team). The predictors introduced in table I are split in "home" statistic and "away" statistics, since for every match there is a team that played at home and the other away, obtaining the relative sums at the end of the season for each team. With respect to the *label*, it has been created started from the variable HOME_TEAM_WINS, which basically contains 1 if the home team won that match and 0 if it lost the match. By adding all the victories of a team for each season (including both those made at home and those made away) and dividing the obtained value by the number of matches played of the relative season, the final *label* can be obtained. In the data set, it is defined in the range $[0.117647, 0.839623]$. Moreover, since the objective is to predict the *label* of the following season, its values have been shifted back by 1, such that the learning techniques later applied would have learned to predict correctly. To be more precise and exaustive, prior to the shifting operation the data set has been ordered with respect to the year of the season and then grouped by the ID of the teams, such that the shift would have correctly been done.

### B. Procedure

After preprocessing the data, the learning techniques introduced in Section III have been applied using a 10-fold cross-validation. The learning techniques that overcame the Dummy Regressor in this step have been later optimized on their hyperparameters using the Grid-Search techniques, which basically tries all the possible combination of the hyperparameters given as input and outputs the one with the lowest error's mean. Using the best combinations obtained with the previous step, the best models have been trained and tested again with a 10-fold cross-validation.

However, it should be noted that performing a cross-validation on a time series creates difficulties in assessing the real performance of the learning technique used. This happens because in 9 out of 10 testing procedures, the training data comes after the testing data on a time basis, so intuitively it would not have been possible for me to do this procedure. But, as shown in [1], where Christoph Bergmeir and José M. Benítez analyzed the use of cross-validation and other procedures with time-series and stated that "Using standard 5-fold cross-validation, no practical effect of the dependencies within the data could be found, regarding whether the final error is under- or overestimated.", I have been able to use it.

### C. Results and discussion

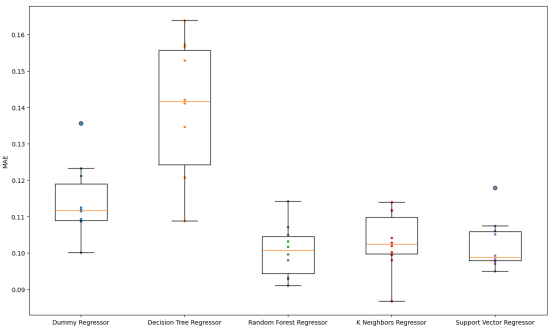


Fig. 1. Initial Results

In the first cross-validation, the learning techniques that outperformed the Dummy Regressor

were RF, KNN and SVM, while the Decision Tree performed poorly. The initial results are shown in Figure 1.

After hyper-parameter tuning, where the hyper-parameters of RF, KNN and SVM have been optimized, the final results have been computed, again doing the initial 10-fold cross-validation. The final results are shown with boxplots in Figure 2 and a slight upgrade can be observed in the performance of the three best learning techniques. The KNN had the best upgrade in performance, while the SVM remained the best learning technique for this problem, considering the lowest MAE and the lower variance compared to the others.

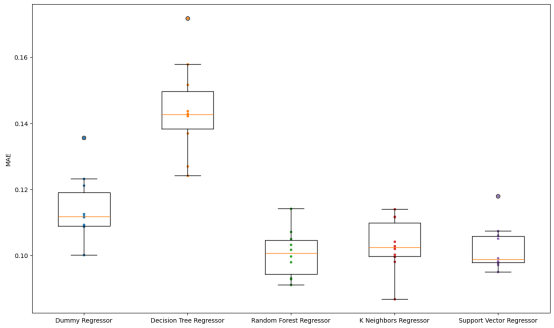


Fig. 2. Final results

## REFERENCES

[1] Christoph Bergmeir and José M Benítez. "On the use of cross-validation for time series predictor evaluation". In: *Information Sciences* 191 (2012), pp. 192–213.