Neural Network Report

Master's Degree in Computer Science and Artificial Intelligence

Football matches results predictor

by

Group: Anonimous2.0

Student Name

Karl Alwyn Sop Djonkam

Bruneau Antoine

Di Placido Anna

University of Nice Côte d'Azur

Under the supervision of Enrico Formenti

November 2020 9, 2023

Contents

Project Description

The goal of this project is to design a football match results predictor by using a machine learning algorithm. The predictor will have to predict the winner of a FIFA championship between national football teams. In order to do this, we have access to two data sets, the dataset "ranking" containing the team's statistics over years and the dataset "results" containing information on the matches played by teams.
During this project, our main goal was to learn, understand the different steps of a machine learning project. That's why, during this one, we didn't just follow the support sheet, but we explored, tried and did our own research on the subject. So there would be some references all along the documents on articles/tutorials we consulted.

Organization

Being new to the world of data science, we decided that everyone should be free to explore the approaches they think are best. That's why we didn't assign distinct tasks to everyone (except at the end). Despite everyone working on his side, we still work as a group. We talked about our progress, exchanged our discoveries and hypotheses, analyzed results and tests obtained by the other members in order to improve our model. At the end of the project, we committed and compared our results and performance in comparison with our data and our models. This enabled us to spot relevant features, parameters and models that we combined then. To have an overview about the major contributions of members, Anna took care of the tournament implementation, she also did the graphical interface. Karl used and optimized the MLP and RandomForestRegressor model. And to finish, Antoine discovered good features to add in our dataset and try different models such as SVM and Adaboost.

The Project

0.1 Analyze of both datasets

Like mentioned above, here is a description of both dataset we used : ,

Dataset 1: International Football Results (1872-2023)

This dataset contains information about international football matches from 1872 to 2023, including the following attributes:

- **date**: Date of the matches
- **home_team**: Name of the team playing at home
- **away_team**: Name of the second team
- **home_score**: Number of goals by the home team
- **away_score**: Number of goals by the away team
- **tournament**: Name of the tournament
- **city**: City where the match took place
- **country**: Country where the match took place
- **neutral**: True if one of the teams played at a neutral venue

This dataset includes records of international football matches from various confederations, starting from the inaugural official match in 1872 up to 2023.

Dataset 2: FIFA World Ranking (1992-2023)

- **rank**: Current country rank
- **country_full**: Country's full name (renamed as 'team' for resampling)
- **country_abrv**: Country abbreviation
- **total-points**: Current total points
- **previous_points**: Total points in the last rating
- **rank_date**: Date of the publication of the rank
- **rank_change**: How the rank has changed since the last publication
- **confederation**: FIFA confederation

This dataset indicates the performance and position of men's national football teams at different points in time, providing insight into fluctuations and trends in the FIFA world rankings.

Dataset 3: "Last_rank.csv" (2023/09/21 - 2023-10-26)

A dataset built by us to add the latest FIFA world rankings that are not in the original dataset (FIFA World Ranking). We decided to build this dataset because recents data are the most representative and so the most important data for prediction problems. In order to build it, we took the information on the Fifa's website and had to take the ranks from the 09/21 and the 10/26 because it was the only ones that weren't in the original dataset.

0.2 Data dropping

The first step was to explore our datasets after importing them using pandas. We start by computing different statistics on them: mean, standard deviation, min, max,

median, the four quartiles, ... We could obtain all those results with the command `dataframe_name.describe()`. . After that we checked if we had some rows with NA values. Hopefully, it wasn't the case. During our exploration we saw that the names of countries weren't consistent across the 2 dataset and even across the same dataset so we use the same name for countries across the 2 dataset to make them consistent. Also some countries no longer exist, they change their names or they splitted or merged in different countries. E.g: "Serbia and Montenegro" that became Serbia and Montenegro countries, "Netherlands Antilles" that merged in the Netherlands. For countries which changed their names we simply use the actual name: we replace all occurrences of the old name with the actual name. For all the countries that splitted or merged we delete their data from our 2 datasets. It represented 0.301% of the ranking dataset and 2.192% of the results dataset (the results dataset which is split on the date chosen). We also noticed that the ranking dataset starts at 31-12-1992 and the result dataset starts at 30-11-1872. It means that before 31-12-1992, we didn't have the FIFIA world ranks of the countries. So, we decided to drop every match in the results dataset that took place before this date. This was a lot of data, but these data weren't as useful because these old matches are not representative of the recent match we want to predict. Moreover, the championship we want to predict is organized by the FIFA and so, we don't need matches where one of the countries is not a member of the FIFA in 2023. That's why we drop these matches. The Fifa ranking dataset gives us the Fifa ranking of teams until 20-07-2023 and the result dataset has matches after the 20-07-2023 and since we don't want to get rid of this data we decide to find the new ranking on the Fifa's website. It is the file Last_rank.csv that we created and we import and append to the ranking dataset. With our 2 dataset consistent on the countries names, the date and without a null value, we combine them into one using the resampling method. We resample using a day as a resampling period and we fill the NA values with the last known value. In our result dataset we add the rank, the rank change, the total points and the previous points for both home and away teams. We add those columns for home team with the suffix _home and for away team with the suffix _away

After the unification of the dataset in a new dataset (that we will call "rera" in this document), we verify the correlation between each pair of features. The correlation

helps us to see if there exists a linear relation between two quantitative variables. If it is the case one feature could help to predict the other, so we should get rid of one and see how it affects the model. Here is the correlation heatmap between the features:

0.3 Data digitalisation

Then we had to digitize the left following features : “tournament”, “home_team”, “away_team” and “neutral”.

For the “tournament” feature, we decided to split these tournaments into 4 ordered groups. Here is the list of groups in the order : “Friendly and lesser-known tournaments”, “Continental Tournaments and Secondary Qualifications”, “Continental Tournaments and Major Qualifications”, and “Major Tournaments”. The objective is to evaluate the importance of those tournaments. For example a win in the “Fifa world cup” is not the same as a win in the “Dragon cup”. The point behind is that it isn’t the same stakes depending on the tournament and the teams do not have the same players for big tournaments than little tournaments.

For the “neutral” feature we just had to convert the True/False in binary 1/0.

And for the “home_team” and “away_team”, the group split into 2 directions : the one hot encoding method in one hand and in the other hand, drop those features.

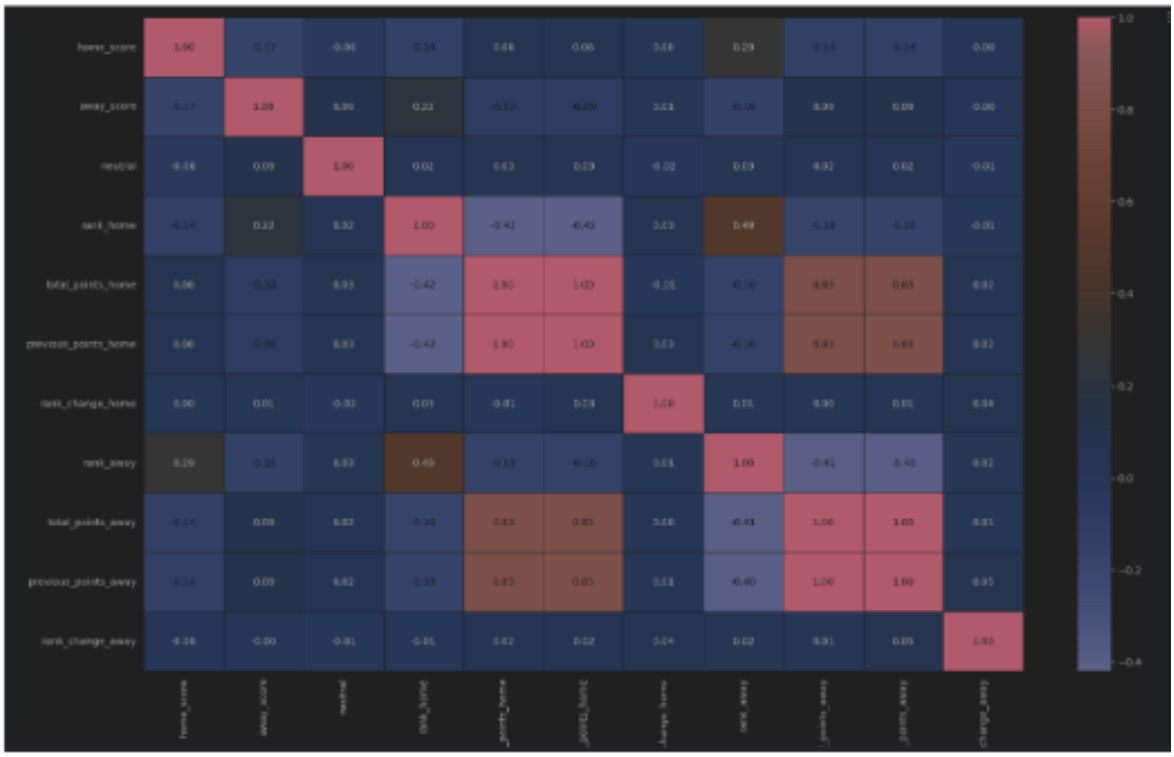
1st method :

For the digitization of datas we choose to use the One-hot-encoding method because in our case we don’t want to impose any order between all the teams like the label encoding method.
For example if we used the label encoding, it will give to France the label 1 and to Montenegro the label 2 which will falsificate the final results.

2nd method :

Drop the features “home_team” and “away_team” because the new features we add enable us to identify those teams without having to know their names.

Then, we verified the correlation between each pair of features. The correlation helps us to see if there exists a linear relation between two quantitative variables. If it is the case one feature could help to predict the other, so we should get rid of one and see how it affects the model. Here is the correlation heatmap between the features:



From this matrix we saw that the pair of features (previous_points_home, total_points_home), (previous_points_away, total_points_away) have a perfect correction $|r| = 1$. We delete one feature of each of these pairs.

0.4 Adding features

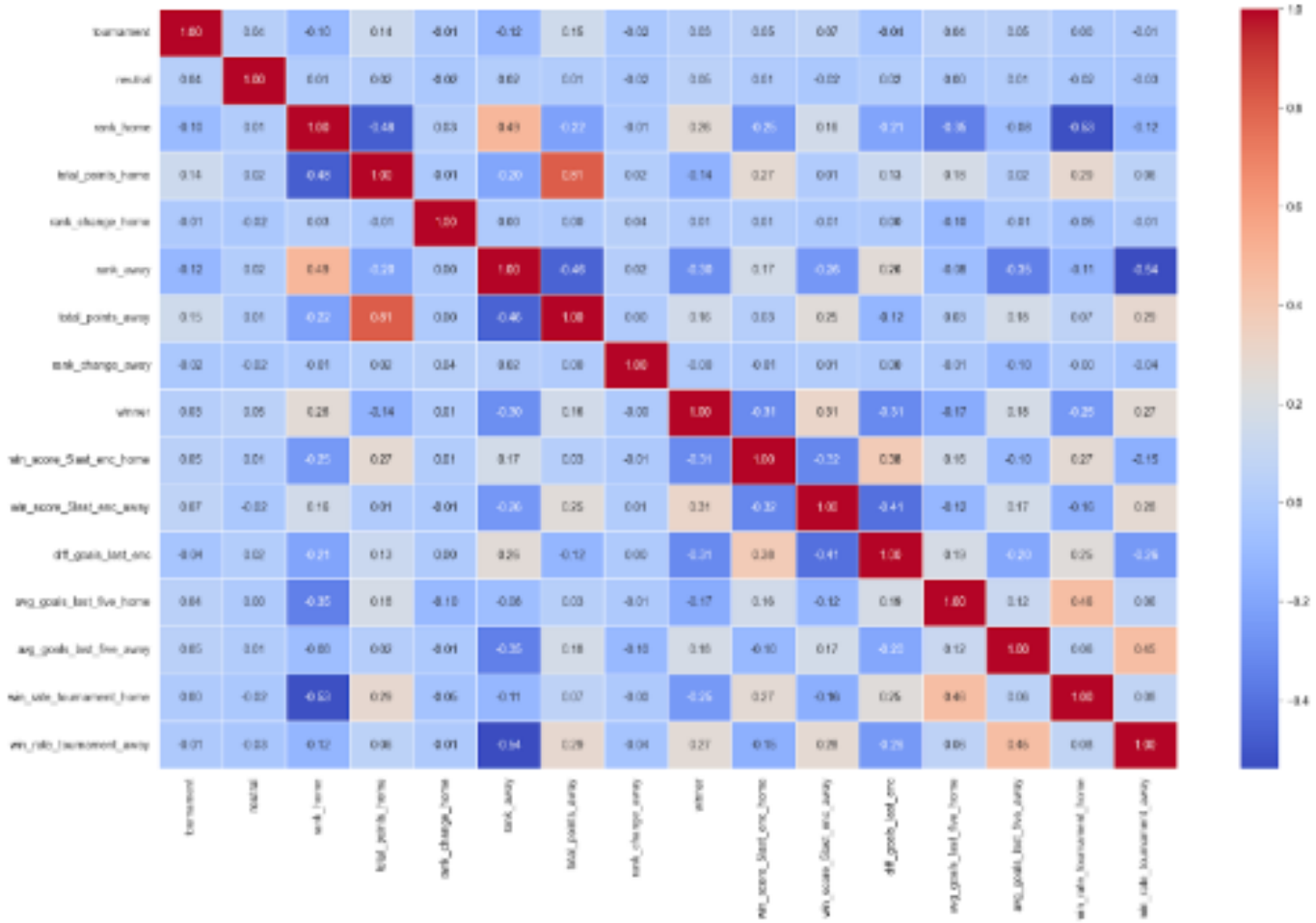
After that, each one decided to create his own new features to add in the dataset. This part is maybe one of the most important parts in terms of time and in view of the future results we will obtain after training the model. Here is the list of features we decided to implement:

- **avg_goals_last_five_home** : Average of the number of goals of the last five matches (whether the team is at home or away) of the home_team
- **avg_goals_last_five_away**: Average of the number of goals of the last five matches (whether the team is at home or away) of the away_team
- **Win_rate_tournament_home** :Win rate of home_team for this type of tournament on their five last matches
- **Win_rate_tournament_away**: Win rate of home_away for this type of tournament on their five last matches
- **Win_score_last5_enc_home** : Score for home_team calculated on the victory against away_team on the 5 last previous matches
- **Win_score_last5_enc_away**: Score for away_team calculated on the victory against home_team on the 5 last previous matches
- **Diff_goals_last_enc**: Difference of goals scored by both teams during their last match
- **Winner**: our target label, a binary class which represent who win

We chose the features "avg_goals_last_five" because this feature could be a good indicator of the team's performance lastly, and so, possibly give us an idea about the physical form and the motivation of the team.

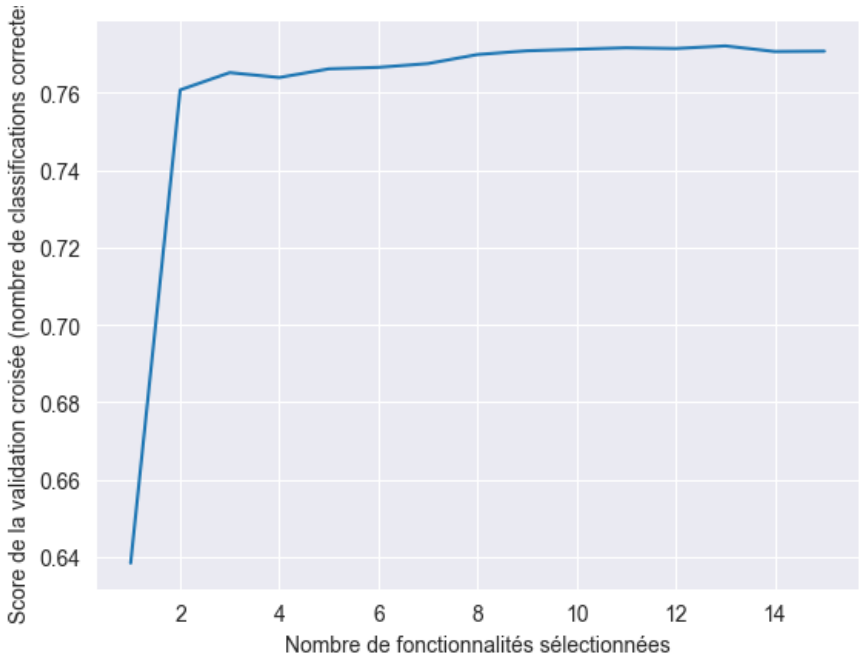
We also add the features "win_rate_tournament" because in our dataset, the tournament feature alone doesn't bring any information on the team's victory. Indeed, if we know for a team his win_rate for the tournament type in which they play, we will have more relevant information to know which one will win Then, we add "Win_score_last5_enc" which enables us to know which teams have been the best for their last five encounters. This feature is a score because we assign more points to the team with a more recent victory. We add "Diff_goals_last_enc" which represents the level difference in their last match together. And finally, the "winner" feature (target feature) that we calculate thanks to the score. If there is a draw, the winner will be the team with the best rank, because this is more probable.

With all those new features, here is the result with the correlation matrix :

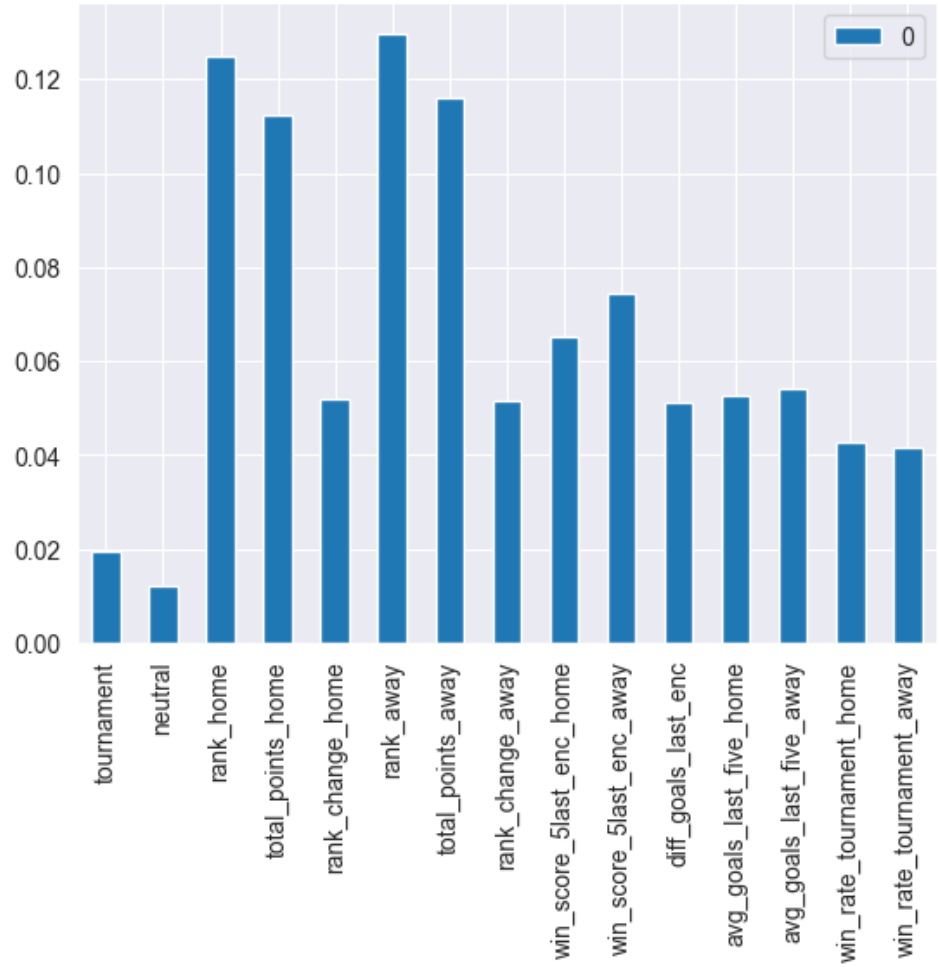


0.5 Features selection

Before training the model, we need features. Our dataset has some but in most cases we should create new ones from the existing ones and delete the ones that will not be used. Creating new features is “feature engineering”. We have removed the features between which there might be a linear relationship. The possibility of the existence of a linear relationship is indicated by a high correlation coefficient. The correlation between all the pairs of features are presented by the correlation matrix presented above. Also, to select the best features we use the attribute features_importance of scikit-learn models, which presents the percentage of explanation of the output for each feature. This method gives a result after training the model. We also have other scikit-learn classes SelectKBest and RFECV. SelectKBest will use a metric to evaluate the KBest features to use. RFECV will also use a metric but will train a model with all the features and gradually remove some of them until you find features that are not necessary. Below is a figure that show us how the RFECV works:



When doing feature selection, we also tried one Hot Encoded the country column. We should encode countries columns because Machine Learning model only works with numbers, so we should encode text into numbers. The figure below shows us the importance of each variable in the output. This graph is obtained by plotting the data of features_importances_ of our models. :



We exclude the OneHotEncode Columns because we had more than 422 features (211 for home team and 211 for away _team) with them. On this figure we see that the features explain more than 92% of the labels prediction and the other 8% are split between the 422 features (the countries names one hot encoded) remaining. We concluded that the one hot encoding wasn't useful and took too much time to train the model. That's why at the end, we decided to drop "away_team" and "hom_team" columns of our dataset. Here is a video we consulted on the feature selection : https://youtu.be/T4nZDuakYlU?si=6JZKTOoOOdJhRGtG

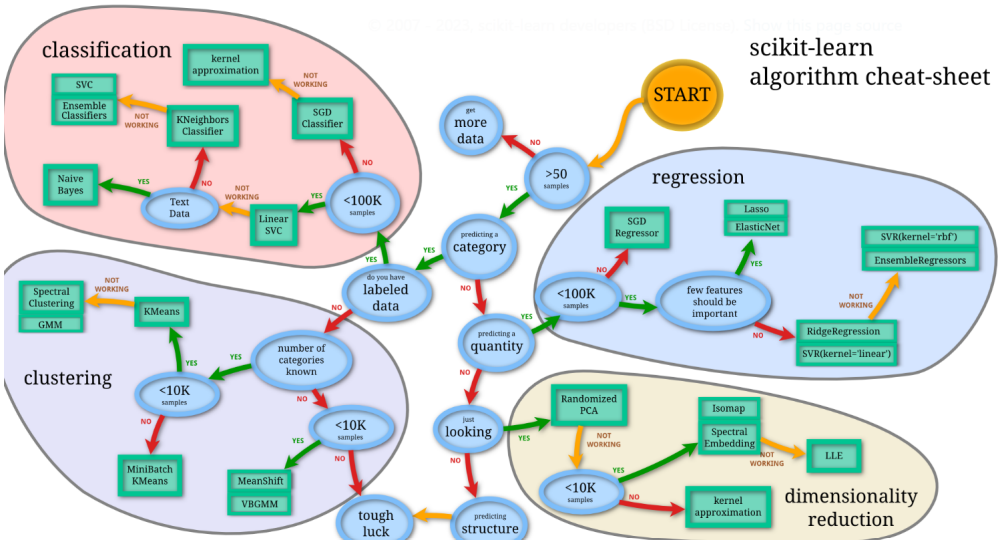
0.6 Last verification

Finally we checked if there was no more NA in our dataset in order to proceed with the rest of the project. We used normalization to scale our features to make them more consistent and comparable, to improve efficiency and convergence of our models and to reduce overfitting.

That is the end of the preprocessing phase. Our task is to predict the winner of a FIFA world cup if it were held now. We checked 2 different approaches, regression and classification. If we consider regression our task will be to predict the score of the home and away teams. With those scores we know who won, lost or if it was a draw. For classification we would have 2 classes: win or lose. During our training we assume that if the 2 teams have the same score, the team with the higher rank will win.

Model Training

In this part, we tested a lot of models in order to know which were the most suitable and accurate for our problem. Here is a diagram below provided by SKLearn which help us choosing models according to our situation :



Here are the different models we tested :

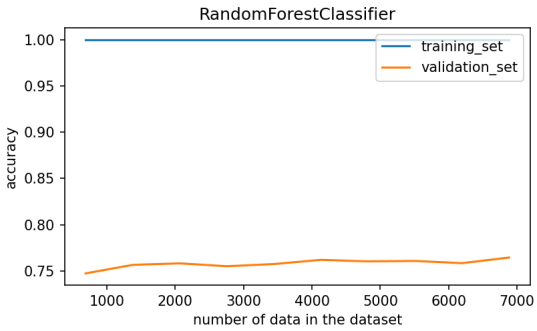
- **Random Forest Classifier}**
- **Adaboost**
- **SVM (Support Vector Machine)**
- **MLP (Multilayer Perceptron)**
- **Random Forest Regressor**

We selected those models based on the diagram above and thanks to other resources.

Here is an explanation of the models :

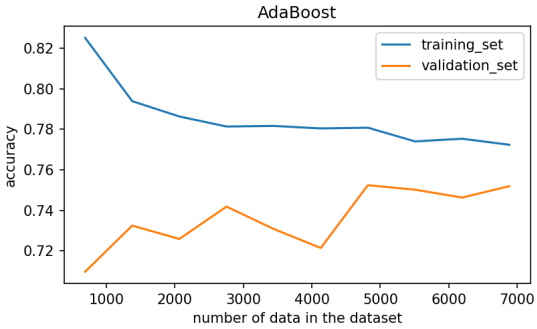
- **Random Forest Classifier: An ensemble model that builds multiple decision trees and combines their predictions for robust classification.**
- **Adaboost: An ensemble model that sequentially adjusts weak models by emphasizing misclassified examples, thereby improving overall performance.**
- **SVM (Support Vector Machine): A model that finds the optimal hyperplane to separate data into classes by maximizing the margin between points of different classes.**
- **MLP (Multilayer Perceptron): A neural network with a multi-layer architecture, used for classification or regression tasks by learning complex data representations.**
- **Random Forest Regressor: Similar to the Random Forest Classifier but used for regression tasks, predicting continuous values instead of discrete classes.**

For each model we decided to fit them with the default parameters and to evaluate their performance with a classification rapport and a method called “learning curve” of SkLearn. We decided to use this learning curve to have an overview on the models that could work well and those that don’t seem to work well for this project. Here are below the results of the classification report and the learning curve for each model :



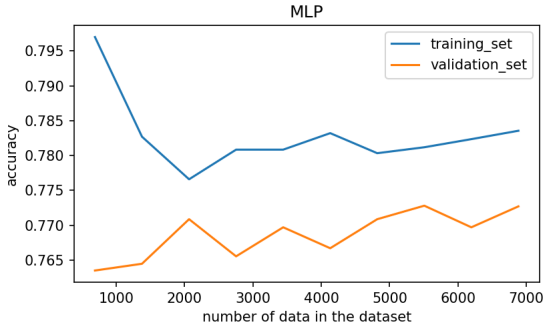
	precision	recall	f1-score	support
0	0.81	0.87	0.84	1572
1	0.77	0.68	0.72	1010
accuracy			0.80	2582
macro avg	0.79	0.77	0.78	2582
weighted avg	0.79	0.80	0.79	2582

(a) Here we see that the Random Forest have a good accuracy (80%), but it is in overfitting



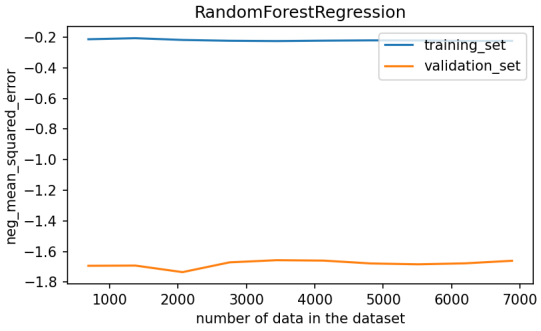
	precision	recall	f1-score	support
0	0.80	0.89	0.84	1572
1	0.79	0.65	0.71	1010
accuracy			0.79	2582
macro avg	0.79	0.77	0.78	2582
weighted avg	0.79	0.79	0.79	2582

(b) here we can see that the Adaboost have a lesser good accuracy than the Random Forest (79%) and seem to know to generalize(not in verfitting)



	precision	recall	f1-score	support
0	0.82	0.88	0.85	1572
1	0.79	0.69	0.74	1010
accuracy			0.81	2582
macro avg	0.81	0.79	0.79	2582
weighted avg	0.81	0.81	0.81	2582

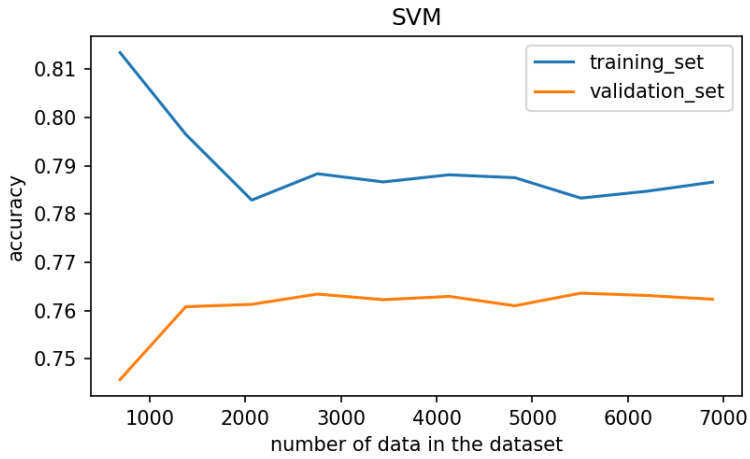
(c) here we can see that the MLP have a better accuracy than all the previous (81%) and seem also to know to generalize(not in overfitting)



{'mean_abs_error': 0.95, 'mean_sqr_error': 1.6, 'r2_score': 0.22}

(d) Same finding than the Random Forest Classifier, it have good result but a tendency to overfit)

A model that finds the optimal hyperplane to separate data into classes by maximizing the margin between points of different classes



	precision	recall	f1-score	support
0	0.80	0.88	0.84	1572
1	0.78	0.67	0.72	1010
accuracy			0.80	2582
macro avg	0.79	0.77	0.78	2582
weighted avg	0.80	0.80	0.79	2582

To conclude, we decided to eliminate the RFC and RFR due to their tendency to overfit. We also eliminate the Adaboost which have a lesser good accuracy. And finally, we only kept SVM because despite having the same accuracy as MLP, MLP is not recommended for this type of problem and it takes much more time than the SVM. After we choose our models, we fine-tune the hyperparameters using GridSearch Class of scikit-learn. We pass to this class the model and a dictionary where the key is the name of the model parameters and the value is a list of all the possible values we want to try for this param. GridSearch will try all the possible combinations using all those param values and will keep the best model's params. To evaluate the best model parameters, GridSearch will use a metric that we gave and cross validation. The best model's parameters are the parameters where the average of the metrics across the k-folds of the cross validation is the best.

For example with SVM, our main model, we obtain thanks to the gridSearch with the hyper param set below, the following params : 'C':700,'degree':1,'gamma': 0.001,'kernel': 'sigmoid'

```
hyper_params_SVM = {
    'C': [100, 400, 700],
    'kernel': ['linear', 'rbf', 'sigmoid', 'poly'],
    'gamma': ['scale', 'auto', 1e-3, 1e-4],
    'degree': [1, 2, 3, 4]
}
```

Championship Cup

We did functions that get recent features for the two teams selected for the match in a dataframe row which will be used for the prediction.

The group stage:

The program reads the csv file and plays the matches in each group where each team plays against each other. Teams are divided into n =

$$2^k$$

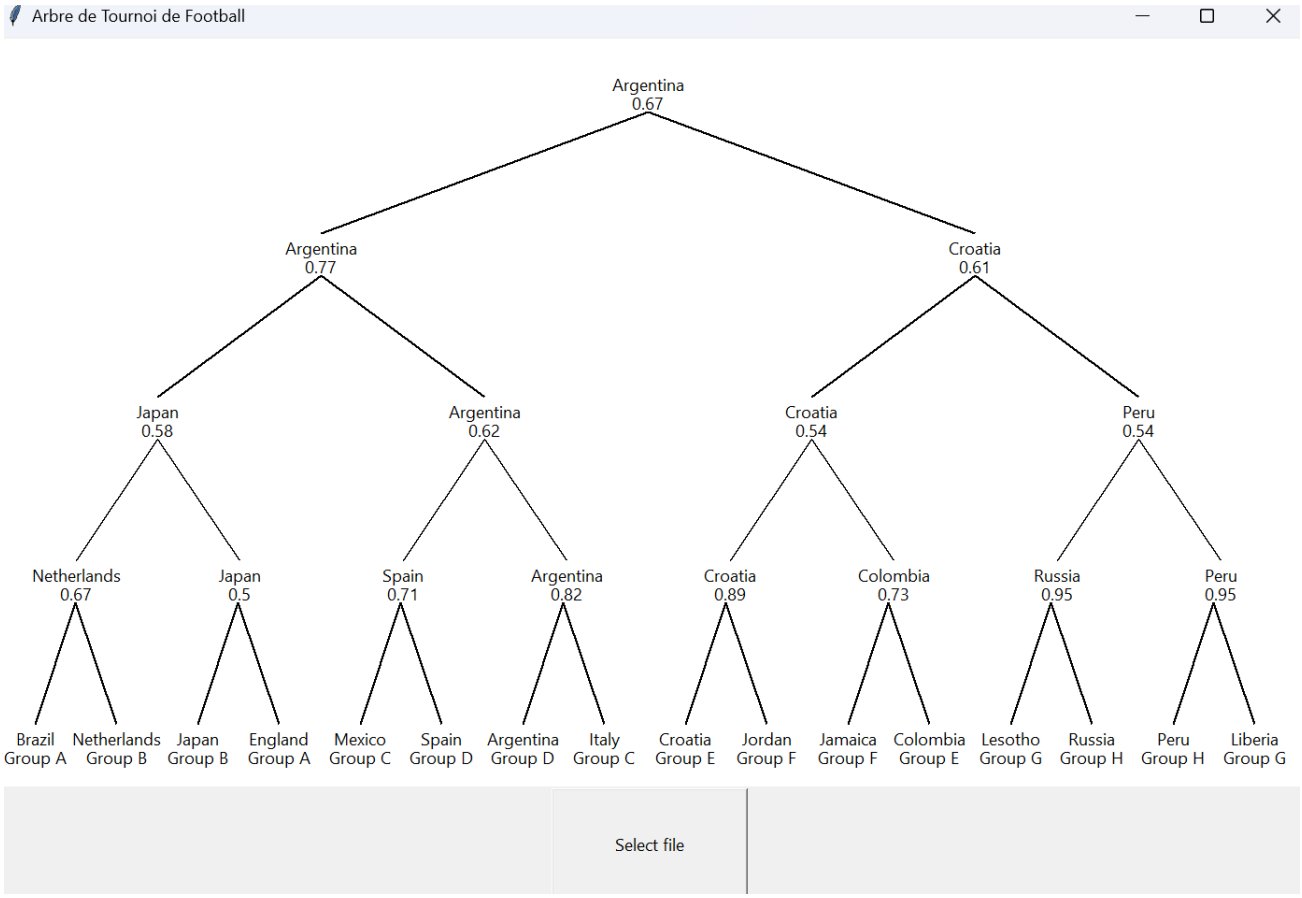
for k $x \in A$ N groups. We extract each feature for the two teams and take them into a dataset row so we can use them for the prediction. The results of the matches are predicted by our model and we can see the probabilities of each team to win. At the end of this stage for each group the two best teams are qualified according to the points they won : +3 points for the winner and +1 point if there is a draw.

The knockout stage:

Only the winners at level n will be admitted to level n + 1 according to their probabilities of winning. A draw is impossible.

GUI

We have created a graphical interface with the Tkinter library. The interface contains a button to load an excel file from File Explorer. After selecting the file, the tournament starts and there is a display in the form of a tree of the knockout stage with the odds and the names of each winning country



Conclusion

It was an interesting learning experience centered around data processing, model creation, and the pursuit of identifying the most accurate model for predicting data.

We have made many experiences to test our programs.

We have made the model which predicts the probabilities of the teams to win.