Machine Learning Algorithms for Football Prediction using statistics from Brazilian championship data

Matheus Kempa Severino[1]

Abstract

This article evaluated football/Soccer prediction in Brazilian Football Championship using various machine learning models based on real-world data from the real games. The models were tested recursively and average predictive results were compared. The results showed that logistic regression and support vectors machine yielded the best results, exhibiting superior average accuracy performance in comparison to others classifiers (KNN and Random Forest). In addition, a ranking of the features' relative importance was made to orient the use of Data.

*Corresponding author
Email address: `matheusks14@gmail.com`

1. Project Overview

Football/Soccer is a sport that is very present in people life's, people use to watch, play and also bet. Thinking on betting, we clearly can see that football is a very unpredictable sport, and it does not acquire a serious research to prove that. In premier league 2015/2016 season, we had a very unexpected champion, and their probability for title in the beginning of the season was one to five thousand Ruela (2016).

So, the prior objective of this project is about create a supervised machine learning algorithm that predicts the football matches results based at the statistics of the matches. Thus it will be possible to evaluate the difficult level of prediction.

2. Problem Statement

This project aims to:

1. Web scrapping robot to pick all the information of the matches
2. Automatize the process of Web scrapping to all the season matches
3. Create a supervised machine learning model to predict the outcome of the matches
4. Evaluate the models

3. Metrics

In classification problems, is common to use accuracy, as evaluation metric. As our outcome prediction is a multi-class problem, it's not going to be necessary to use other metrics.

$$\text{Accuracy} = \frac{TP+TN}{TP+FN+TN+FP}$$

Where TP are the true positives, FP are false positives, TN are the true negatives and FN are false negatives.

4. ETL and Data Exploration

4.1. Web-Scrapping

However before exploring the collected data, it's crucial to understand how this information were collected. So, this part will reach since the web-scrap robot developed to the final analysis of the whole database treated.

The fist and the second image bellow show how displays the page that the data was collected. So step number one is: Web-scrap the main page, picking the football data and creating a Data frame with all information combined.

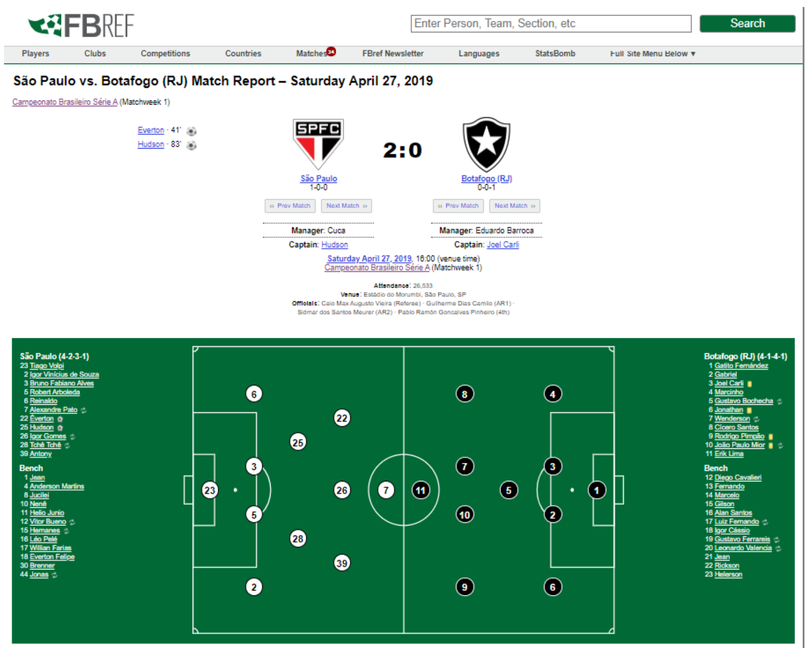

Figure 1: Example Match Page. FBREF (2019)

São Paulo Player Stats

Share & more ▾ Glossary

					Performance																
Player	#	Nation	Pos	Min	Gls	Ast	PK	PKatt	Sh	SoT	CrdY	CrdR	Fls	Fld	Off	Crs	TklW	Int	OG	PKwon	PKcon
Alexandre Pato	7	BRA	FW	75	0	0	0	0	1	0	0	0	2	0	2	1	0	1	0		
Jonas	44	BRA	MF	15	0	0	0	0	1	0	0	0	0	1	0	0	0	0	0		
Éverton	22	BRA	AM	90	1	0	0	0	1	1	0	0	4	0	0	0	0	0	0		
Igor Gomes	26	BRA	AM	63	0	0	0	0	1	0	0	0	1	2	0	1	1	0	0		
Hernanes	15	BRA	MF	27	0	1	0	0	1	1	0	0	0	0	0	0	0	0	0		
Antony	39	BRA	AM	90	0	1	0	0	0	0	0	0	0	2	0	3	1	1	0		
Hudson	25	BRA	DM	90	1	0	0	0	1	1	0	0	4	1	0	0	0	0	0		
Tchê Tchê	28	BRA	DM	86	0	0	0	0	1	1	0	0	2	1	0	0	0	1	0		
Vitor Bueno	12	BRA	MF	4	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
Reinaldo	6	BRA	LB	90	0	0	0	0	0	0	0	0	2	0	0	4	1	0	0		
Bruno Fabiano Alves	3	BRA	CB	90	0	0	0	0	0	0	0	0	3	0	0	0	0	0	0		
Robert Arboleda	5	ECU	CB	90	0	0	0	0	0	0	0	0	2	0	0	0	1	1	0		
Igor Vinícius de Souza	2	BRA	RB	90	0	0	0	0	0	0	0	0	3	0	0	2	0	1	0		
Tiago Volpi	23	BRA	GK	90	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0		
14 Players				**990**	**2**	**2**	**0**	**0**	**7**	**4**	**0**	**0**	**23**	**7**	**2**	**11**	**4**	**5**	**0**		

São Paulo Goalkeeper Stats

Share & more ▾ Glossary

			Shot Stopping			
Player	Nation	Min	SoTA	GA	Saves	Save%
Tiago Volpi	BRA	90	2	0	2	1.000

Botafogo (RJ) Player Stats

Share & more ▾ Glossary

					Performance																
Player	#	Nation	Pos	Min	Gls	Ast	PK	PKatt	Sh	SoT	CrdY	CrdR	Fls	Fld	Off	Crs	TklW	Int	OG	PKwon	PKcon
Erik Lima	11	BRA	FW	90	0	0	0	0	0	0	0	0	1	4	0	0	0	1	0		
Rodrigo Pimpão	9	BRA	LW	90	0	0	0	0	3	1	1	0	1	0	1	4	0	0	0		
João Paulo Mior	10	BRA	AM	76	0	0	0	0	2	0	1	0	1	3	0	2	2	1	0		
Leonardo Valencia	20	CHI	MF	14	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0		
Wenderson	7	BRA	AM	56	0	0	0	0	1	0	0	0	0	4	0	0	1	0	0		
Luiz Fernando	17	BRA	MF	34	0	0	0	0	0	0	0	0	0	2	0	0	0	0	0		
Cícero Santos	8	BRA	RW	90	0	0	0	0	2	1	0	0	0	2	0	0	0	0	0		
Gustavo Bochecha	5	BRA	DM	76	0	0	0	0	0	0	0	0	2	4	0	0	4	0	0		
Gustavo Ferrareis	19	BRA	MF	14	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0		
Jonathan	6	BRA	LB	90	0	0	0	0	0	0	1	0	1	0	0	2	0	1	0		
Gabriel	2	BRA	CB	90	0	0	0	0	1	0	0	0	0	0	0	0	2	1	0		
Joel Carli	3	ARG	CB	90	0	0	0	0	0	0	1	0	1	0	0	0	1	1	0		
Marcinho	4	BRA	RB	90	0	0	0	0	1	0	0	0	0	0	0	1	0	1	0		
Gatito Fernández	1	PAR	GK	90	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0		
14 Players				**990**	**0**	**0**	**0**	**0**	**11**	**2**	**4**	**0**	**7**	**20**	**1**	**10**	**11**	**6**	**0**		

Botafogo (RJ) Goalkeeper Stats

Share & more ▾ Glossary

			Shot Stopping			
Player	Nation	Min	SoTA	GA	Saves	Save%
Gatito Fernández	PAR	90	4	2	2	.500

Figure 2: Example Match Page Data - 2.FBREF (2019)

Now that we have the code ready to pick data from the matches it is necessary to create another code to collect all the matches URLs, so that it will be an automatized robot to do this task.

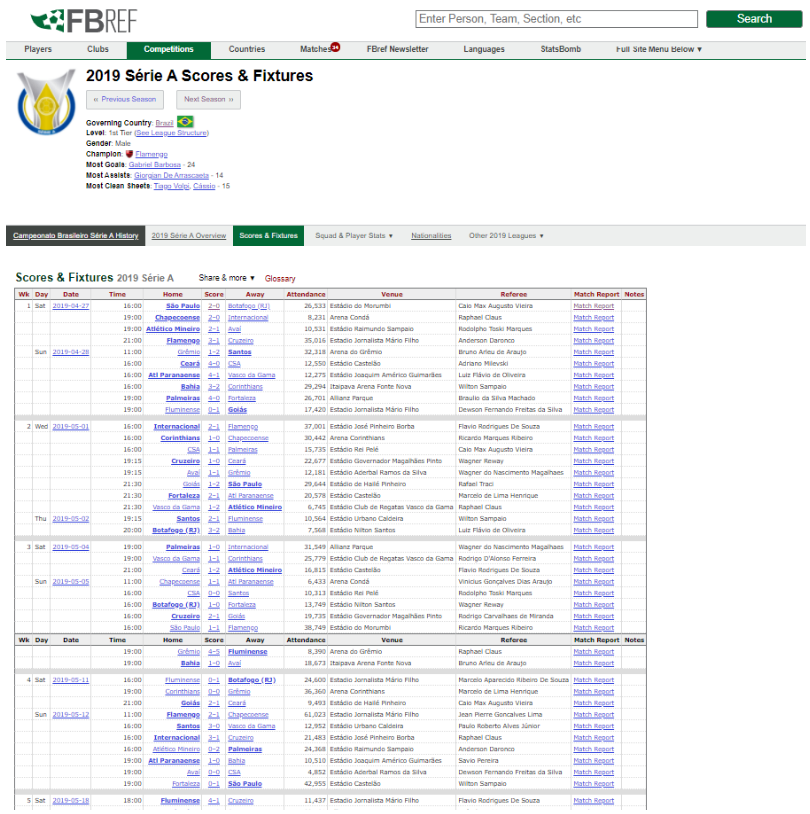

Figure 3: Example Match Page. FBREF (2019)

All the code will be attached to the Git repository:

- https://github.com/Matheuskempa/My_Udacity_Capstone

Before the data were clean and ready to be used for analysis, the bellow steps were done:

1. Select columns: Selected columns that hadn't had a lot number of null values.
2. Problems: The data collect came with some "problems" as the total of all players team's statistics, so it was needed that this lines were expelled from the data.
3. Group by match and team: Because the data collected were from the players of the match, it was necessary to group all the statistics of the player's team to matches and teams.

4. Append the Result: Because the data collected were from the players table, it did not brought the result of the match, so it was necessary to create a code that append this result to the Data frame.
5. Place: It's was necessary to create a code that show witch team played in home and away.

4.2. Data

Now it's time to evaluate the data collected so that it could be possible to create a prediction model to the results. These were the columns that our cleaned data had:

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 760 entries, 0 to 759
Data columns (total 21 columns):
 #   Column     Non-Null Count  Dtype
---  ------     --------------  -----
 0   Confronto  760 non-null    object
 1   Time       760 non-null    object
 2   Data       760 non-null    object
 3   Gls        760 non-null    int64
 4   Ast        760 non-null    int64
 5   PK         760 non-null    int64
 6   PKatt      760 non-null    int64
 7   Sh         760 non-null    int64
 8   SoT        760 non-null    int64
 9   CrdY       760 non-null    int64
 10  CrdR       760 non-null    int64
 11  Crs        760 non-null    int64
 12  Fls        760 non-null    int64
 13  TklW       760 non-null    int64
 14  Int        760 non-null    int64
 15  Fld        760 non-null    int64
 16  Torcida    760 non-null    float64
 17  OG         760 non-null    int64
 18  Off        760 non-null    int64
 19  Resultado  760 non-null    int64
 20  Place      760 non-null    object
dtypes: float64(1), int64(16), object(4)
memory usage: 150.6+ KB
```

Figure 4: Columns info

This is the meaning of all the variables:

- Confronto: Match
- Time: Team

- Data: Date of the match
- Gls: number of Goals in the match
- Ast : Assists
- PK : Penalty Kicks Made
- PKatt : Penalty Kicks Attempted
- Sh : Shots Total
- SoT : Shots on target
- CrdY : Yellow Cards
- CrdR : Red Cards
- Crs : Crosses
- Fls : Fouls Committed
- TklW : Tackles Won
- Int : Interceptions
- Fld : Fouls Drawn
- Torcida : Crowd
- OG : Own Goals
- Off : Offsides
- Resultado : Result of the match (Victory, Loss, Draw)
- Place : Home/ Away

** Observation: Every match had to lines one for the home team, and othe for the away team.

It's good to check how these variables relate with each other, so that it was created to approaches: Attack and Defense. In the code it was also provided a function that does this approaches by team and place played (Home/Away). In the code attached on Git Repository, will be provided a more complex the analysis of a team.

Figure 5: Attack Data

For the attack approach it was selected 4 variables: "SoT","Sh","Gls","Torcida". Looking at the data it's possible to see how difficult is to solve this problem because we can't find patterns looking at it. But it might be possible to conjecture some hypotheses different from the usual as:

- A high Torcida(football crowd) might not be related to a more significant number of goals.
- A low Torcida(football crowd) might be related to a high number of SoT(Shot on target).

Figure 6: Defense Data

With this graphs it's possible to see how our data is related, how the columns are related to each other, thus looking at 3 variables at the same time is possible to see the level of difficulty to separate this data. For the defense approach it was selected 4 variables: "TklW","Fls","CrdY","Torcida". And as in attack analysis it was not possible to find any patterns on the Defense Data either.

However looking at Figure **4**, there's more variables available than those showed on graphs. And it's clear that at least 17 collected variables, can influence on the performance of the prediction. But because football is a lot more complex, more variables were needed to have a good predict a match outcome, so in the next part it will be generate some other variables.

5. Methodology

5.1. Data Pre-processing

As it's not possible to have the statistic of the games before the result of the match, it's necessary to create some new variables that would be available before the games. So in order to solve this problem it will be generate a mean for all the variables, an this mean will contain all the games before the correspondent game, by this way when a team play in September 18, the code will provide a mean of all the variables available to all the games before this exactly game.

It's going to be created some variables that tries to show the computer the sequence of points for every team in the last 5 games, 3 games and for the last game also. For every victory the team the code summed 3 points, for a draw summed 1 point and for a loss summed no point, 0 points. By this way it would be possible to see if the team comes from a victories, draws and losses.

Since this treatments were done, it was found a way to inform the machine the "place" of the match. Because of this variable (place) and as the championship has 2 turns, the first turn may be in the Home of a team, and if the first game of the the first championship turn it was on their home, the second necessarily has to me Away, or in other words, not him their stadium. And in Football matches this is a very important variable. So the way designed to consider this variable was: Taking all data with all the variables created for the home teams(Place = Home) for every match, and then, subtract by the same dataset(the same variables) but from visiting teams (Place = Away). By this way it could be possible to generate a Data base that the data would basically say:

- Variables = Negative Values: The visiting team has had better performances in this variable in the past games than the visitant/opponent. It's possible to know that because, for example: picking the variable "Average of Shots", if the home team has a value P, then the visit team has X, if P is higher than X the output value of the subtraction of both (HOME - AWAY) will be positive, otherwise if P is lower than X the value will be negative, showing that the team does not have better performances on that variable in the past games played.

Recapping:

- Problem 1: The variables were only available after the match and for our model to perform it's necessary to have all data features data before the correspondent match, so that it was created a season averages for every variable of the season and more, some moving averages for each variable of the Dataset too.

- Problem 2: Insert the sequence variables: That problem was solved by summing all the points that the team had had in the past 3, 5 games and also the last game. For every victory the team summed 3 points, for a draw 1 point and for a loss 0 points.

- Problem 3: Show to the machine who is the home and the away team. To solve this we subtracted the home team variables results by the visitant variables results for every match. Showing by this way if the team home is in any way superior or inferior than the visitant team.

Thee final Database had 41 columns and 380 lines, and looked like this:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 380 entries, 0 to 379
Data columns (total 41 columns):
 #   Column                Non-Null Count  Dtype  
---  ------                --------------  -----  
 0   Confronto             380 non-null    object 
 1   Data_new              380 non-null    object 
 2   Time                  380 non-null    object 
 3   Time_Fora             380 non-null    object 
 4   avrg_Gls              380 non-null    float64
 5   last_3_avrg_Gls       380 non-null    float64
 6   avrg_Ast              380 non-null    float64
 7   last_3_avrg_Ast       380 non-null    float64
 8   avrg_PK               380 non-null    float64
 9   last_3_avrg_PK        380 non-null    float64
 10  avrg_PKatt            380 non-null    float64
 11  last_3_avrg_PKatt     380 non-null    float64
 12  avrg_Sh               380 non-null    float64
 13  last_3_avrg_Sh        380 non-null    float64
 14  avrg_SoT              380 non-null    float64
 15  last_3_avrg_SoT       380 non-null    float64
 16  avrg_CrdY             380 non-null    float64
 17  last_3_avrg_CrdY      380 non-null    float64
 18  avrg_CrdR             380 non-null    float64
 19  last_3_avrg_CrdR      380 non-null    float64
 20  avrg_Crs              380 non-null    float64
 21  last_3_avrg_Crs       380 non-null    float64
 22  avrg_Fls              380 non-null    float64
 23  last_3_avrg_Fls       380 non-null    float64
 24  avrg_TklW             380 non-null    float64
 25  last_3_avrg_TklW      380 non-null    float64
 26  avrg_Int              380 non-null    float64
 27  last_3_avrg_Int       380 non-null    float64
 28  avrg_Fld              380 non-null    float64
 29  last_3_avrg_Fld       380 non-null    float64
 30  avrg_Torcida          380 non-null    float64
 31  last_3_avrg_Torcida   380 non-null    float64
 32  avrg_OG               380 non-null    float64
 33  last_3_avrg_OG        380 non-null    float64
 34  avrg_Off              380 non-null    float64
 35  last_3_avrg_Off       380 non-null    float64
 36  pnts_lst_5            380 non-null    int64  
 37  pnts_lst_3            380 non-null    int64  
 38  pnts_lst_game         380 non-null    int64  
 39  Result                380 non-null    int32  
 40  N_Gols                380 non-null    int64  
dtypes: float64(32), int32(1), int64(4), object(4)
memory usage: 120.4+ KB
```

Figure 7: Final Data

Coming at this point our database has By now it's possible to run some models and check their performances

5.2. Implementation and Refinement

To run we dropped the first four variables and "Gls" variable either, also was used MinMaxScalar on all the feattures variables. As it can be seen on Sklearn (2020) this estimator scales and translates each feature individually such that it is in the given range on the training set, e.g. between zero and one.

Moreover were used all this 4 models:

- Support Vector Machine: is a supervised learning model, that always aims to increase the distance between the points so that it is possible to classify the classes, so, SVM separates data maximizing the margin between the classes.

- Random Forest: As the name says, random forest creates several decision trees and groups them into one “Forest” of trees, taking a sample with size m bootstrap of the columns that represent the explanatory variables when partitioning the tree in each node of it. The final decision vote will be given by the majority vote for classification problems (and the average, in a regression problem).

- KNN: is a simple model that performs classification based on the class of its k nearest points based on a distance metric;

- Logistic Regression: Basically, logistic regression is a multiple linear regression whose result is “squeezed” in the interval $[0, 1]$ using the sigmoid function.

In order to run all this models we split the Database randomly using the library ”train test split” from scikit-learn. For the test it was used 30% the Data. Moreover the algorithm was randomly played through 1000 times, for all four models. By this way it was possible to check the standard deviation and the accuracy of all models. All models were used by the default values, not exploring the parameters of each specifically model.

6. Results

6.1. Model Evaluation and Validation

As it can be seen in the table bellow, the results weren’t quite expressive. The algorithm achieved almost 50% of accuracy, maybe, considering that the random prediction probability of success is 33% (Victory/Draw/Loss), it is a good signal, showing us that the algorithm was able to identify some patterns after all.

Model	Accuracy
Random Forest	44.78% (4.10%)
KNN	45.65% (4.01%)
Logistic Regression	49.77% (4.02%)
SVM	47.15% (4.11%)

Table 1: Mean and standard deviation of performance

After all 1000 times the Logistic Regression had the best results, with one of the lowest standard deviation rates and the highest accuracy between all four models. SVM performed better than the others, but had the highest standard deviation, maybe showing that the model tends to vary more thant other models.

Also were created a feature ranking using Random Forest features importance. As it can be seen in the figure bellow, football game crowd from the last 3 matches, the yellow cards numbers and also football game crowd from all the season were influencing more on the prediction outcome.

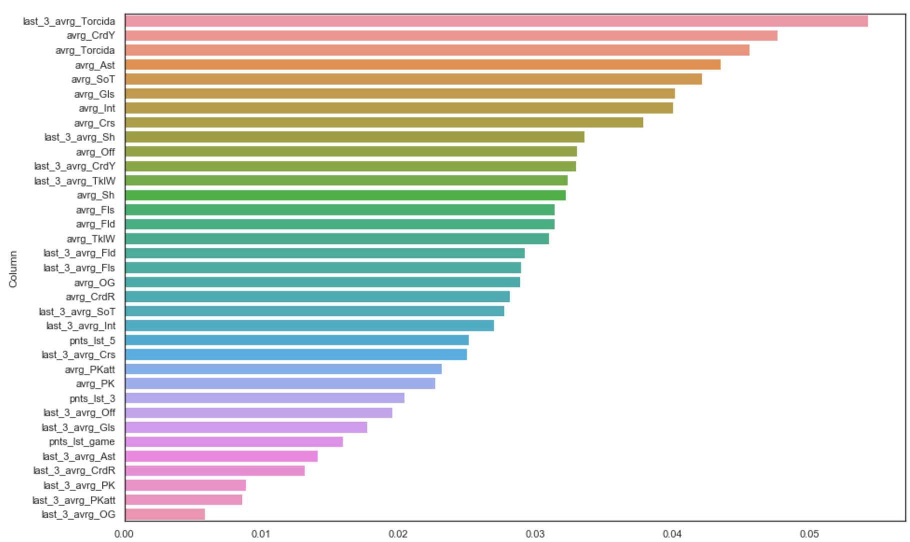

Figure 8: Features Ranking

6.2. Justification

It's weird to see logistic regression performed much better than the other models, mainly because as it was seen, data clearly it did not have patterns and it was a complex problem and surprisingly a very complex problem was solved, by a "linear" approach, maybe the simplest here. One of the reasons of this may be because the parameters of each model, that were not explored in this approach. Because SVM has a lot of kernels, trying different kernels it might be a solution to reach a better performance for this model.

Another weird point it is that the points sequence was one of the lasts variables to interfere on the prediction, maybe there are more ways to show this patterns to the machine.

7. Conclusion

7.1. Reflection

This articles proves that football prediction is still a very hard task, it still needs more variables to help on the prediction of the results. However we can see by this article that a machine learning algorithms can already "think" on witch team bet an can still be more accurate than people that does not know about the games having almost 17% of advantage in the prediction when comparing to the probability of a randomly prediction.

7.2. Improvement

For the future i suggest to investigate and find more variables that could be usefully, as injuries for example, or more details of the players of each team, maybe web-scrapping the fifa data to get this kind of information. Another thing that could be done for the future is on predicting the number of goals for of each team, this is more complex because it depends of the results predicted. So, maybe, this article can be a source of inspiration to the creation of better models in the future.

References

FBREF, 2019. São paulo vs. botafogo URL: https://fbref.com/en/matches/e1867e7b/Sao-Paulo-Botafogo-RJ-April-27-2019-Serie-A.

Ruela, J., 2016. Leicester campeão: a aposta mais improvável da história britânica URL: https://www.dn.pt/desporto/leicester-campeao-a-aposta-mais-improvavel-da-historia-britanica-5152206.html#:~:text=Segundo%20Joe%20Crilly%2C%20da%20William,paga%20mais%20alta%20na%20hist%C3%B3ria.

Sklearn, 2020. sklearn.preprocessing.minmaxscaler URL: https://scikit-learn.org/stable/modules/generated/sklearn.preprocessing.MinMaxScaler.html.