The winner of the next match

Simone Malesardi[978989]

Università Degli Studi di Milano

Abstract. The aim of the project is to define an automatic system to predict the outcome of the matches starting from the news concerning the teams involved in the match published in the days and weeks preceding the match. It was also possible to use other statistical data to make prediction regarding matches.

To predict an outcome of a match is a complex task which need different information coming from different sources.

There are different factors that can be used for predicting an outcome such as the number of scored goals and conceded goals, yellow and red cards, etc.

The outcome problem became a classification problem because an outcome match could belong to 3 different classes:

- home team win (away team lose)
- away team win (home team lose)
- draw

To build a model that allows predicting the outcome of matches, an initial analysis has been made and different materials has been used such as literal surveys, publications, lectures and papers found online and made available by the teacher.

Keywords: Football · Outcome prediction · Information retrieval

1 Data collection

In an initial phase, I tried to identify sources in order to find useful data to build the predicting model and, after different searches on the web, I found two different data sources:

- **FBref**: in this website statistics of various teams belonging to different leagues (male and female league) are published.
- **FootballPredictions**: in this website news of teams are published before the matchday

The competition taken into account, also for previous knowledge, is the Italian Serie A.

1.1 FBref

1.1.1 Obtaining the FBref data

FBref publishes different HTML tables (for each team) analysing statistics calculated over played matches by the team.

Matches used for the development of the project include all the matches played in seasons between the 2018-2019 one and the actual one.

Punteggi e partite 2022-2023 Juventus: Tutte le competizioni Condividere ed esportare ▾ Glossario

Data	Ora	Competizione	Girone	Giorno	Stadio	Risultato	Rf	Rs	Avversario	xG	xGA	Poss.	Spettatori	Capitano	Formazione	Arbitro	Report partita	Note
15-08-2022	20:45	Serie A	Giornata n. 1	Lun	Casa	V	3	0	Sassuolo	2.0	0.9	42	38.725	Leonardo Bonucci	4-4-2	Antonio Rapuano	Report partita	
22-08-2022	20:45	Serie A	Giornata n. 2	Lun	Ospiti	N	0	0	Sampdoria	0.8	0.9	52	27.573	Juan Cuadrado	4-3-3	Rosario Abisso	Report partita	
27-08-2022	18:30	Serie A	Giornata n. 3	Sab	Casa	N	1	1	Roma	0.6	1.3	57	41.003	Juan Cuadrado	4-2-3-1	Massimiliano Irrati	Report partita	
31-08-2022	20:45	Serie A	Giornata n. 4	Mer	Casa	V	2	0	Spezia	0.9	0.2	50	35.709	Juan Cuadrado	4-3-3	Andrea Colombo	Report partita	
03-09-2022	15:00	Serie A	Giornata n. 5	Sab	Ospiti	N	1	1	Fiorentina	0.9	2.1	40	36.550	Juan Cuadrado	4-3-3	Daniele Doveri	Report partita	
06-09-2022	21:00	Champions Lg	Fase a gironi	Mar	Ospiti	P	1	2	Paris S-G	1.0	1.7	43	47.415	Leandro Paredes	3-5-2	Anthony Taylor	Report partita	
11-09-2022	20:45	Serie A	Giornata n. 6	Dom	Casa	N	2	2	Salernitana	1.9	1.6	61	35.097	Leonardo Bonucci	4-3-3	Matteo Marcenaro	Report partita	
14-09-2022	21:00	Champions Lg	Fase a gironi	Mer	Casa	P	1	2	Benfica	0.8	2.2	45	34.015	Leonardo Bonucci	3-5-2	Felix Zwayer	Report partita	
18-09-2022	15:00	Serie A	Giornata n. 7	Dom	Ospiti	P	0	1	Monza	0.9	2.5	40	12.878	Danilo Luiz da Silva	4-3-3	Fabio Maresca	Report partita	
02-10-2022	20:45	Serie A	Giornata n. 8	Dom	Casa	V	3	0	Bologna	1.8	0.4	48	34.662	Leonardo Bonucci	4-4-2	Rosario Abisso	Report partita	

The above example shows data taken from FBref for the Juventus team in the 2022-2023 championship.

I extracted some features from FBref :

- goals
- goals_on_penalty
- fouls
- shots_on_target
- total_penalties
- yellow_cards
- total_shots
- percentage_possession
- red_cards

The datasets are extracted by making parsing of the HTML tables with the python library BeatifulSoup. Once data have been extracted, they are stored on csv files and every championship has its csv because.

download.py is the extraction script executable from the notebook download_and_merge_matches.ipynb.

At the end of this process a unique dataset "all_stats.csv" is obtained and it combines statistics of downloaded seasons.

1.1.2 Cleaning and pre-processing the FBref data

The dataset needs to be processed. Statistics referring a single match are composed by a pair of record because if in "17 December 2022" the match is "Lecce-Fiorentina" in the dataset appear one record representing the statistics of Lecce and one representing statistics of Fiorentina.

These values representing the same match must be combined.

According with an article of the largest professional league North America of American Fooball (NFL [2]) for creating the final dataset, different steps are followed:

- combination of matches
- statistics processing
- computation of ranking

Each pair of records referring the same match is merged into following 3 steps:

- filtering previous statistics of the 2 teams involved in the match, so I obtain two datasets (one referring the home statistics and one referring the away statistics)
- computing the average for each statistic (1.1.1) in the previous 5 matches
- subtracting features [avg(home features) – avg(away features)]

Another important feature is the ranking of the teams. Rankings are values that change from week to week or even from day to day and find these values for each match wasn't possible so I calculated for each match the number of direct confrontations won by home team and by away team in the last 6 seasons.

When all those feature are computed, the "cleaned_stats.csv" is saved.

1.2 FootballPredictions

1.2.1 Obtaining the FootballPredictions data

FootballPredictions (FP) is a website of football news.

FootballPredictions allow to search articles published before a match setting in the url the date of the match you want obtain news, e.g. https://footballpredictions.com/footballpredictions/?date=23-10-2022

From "cleaned_stats.csv" are taken all the dates on which matches have been played and for each data I obtain the FP link (as the above link) of the HTML page where I can find the list of all the matches in program in that data. Each element of this list has a link which take to another page where there is published the news about the match.

The time for obtaining all links is 15 minutes. To speed up the process I created a file "links.json" containing date as key and links found in the page date referred to Serie A that lead to a new page (the page with the news).


```
"2022-10-23": [
    "https://footballpredictions.com/footballpredictions/serieapredictions/udinese-vs-torino-prediction-4/",
    "https://footballpredictions.com/footballpredictions/serieapredictions/bologna-vs-lecce-prediction-2/",
    "https://footballpredictions.com/footballpredictions/serieapredictions/atalanta-vs-lazio-prediction-6/",
    "https://footballpredictions.com/footballpredictions/serieapredictions/as-roma-vs-napoli-prediction-5/"
],
```


Fig. 1. Example of object in links.json

For each link in the json is done scraping of the html page to obtain the news published and the prediction (a label that I used to build and evaluate the ML model, see section 2.1).

Team names read from football predictions aren't normalized, so they need to be processed. For example on FP there's "AS Roma" while on the dataset obtained from FBref there's "Roma". I created a list of team names (same names used by FBref) and each name got from FP is compared with the list and if there's an high match with one of the names, the FP name is replaced with the FBref one.

At the end of this process I obtained the "cleaned_news.csv" dataset composed by these features:

- date: date of the match
- name of the home team
- name of the away team
- description: news read
- prediction: made from FP and I treated it as a label (V, P, N)
- season: useful for the merge with the statistics dataset

The download of all news takes 36 minutes.

1.3 Final pre-processing

This phase merge the two datasets ("cleaned_stats.csv" and "cleaned_news.csv") by 3 parameters: **home team**, **away team**, **season**.

The merge isn't done by date because sometimes the date of a match in the first dataset is not equal to the same date of the same match in the second dataset.

Merging datesets a "completed_dataset.csv" is obtained.

2 Model building

As I told in introduction section, matches can be classified into 3 classes:

- home team winner (label **V** in csv files → label **1** used by the model)
- away team winner (label **P** in csv files → label **2** used by the model)
- draw (label **N** in csv files → label **0** used by the model)

To build the predictive model I used two different supervised classification models:

- **Decision Tree**: used to analyse the text read from football prediction and to get the label
- **Random Forest**: used to build the final model that predicts the final result of next matches

2.1 FootbalPrediction classification

Text analysis is an important task in the text mining and in order to do that I used a supervised approach. Before train the model, texts have to be tokenized through 4 phases:

- normalization of synonym: it is done because texts could be inconsistent: for example taking into account the the text of the match between *Sassuolo-Hellas Verona* in the 24 October 2022.

> *Monday's football game between **Sassuolo** and **Verona** will bring down the curtain on day 11 of the new Serie A campaign. The **Neroverdi** aim to make amends for back-to-back losses to Inter and Atalanta, and they are surely capable of matching the **Mastini** on home soil. [...]*
>
> ***Verona**, on the other hand, are on a five-game losing run in the Italian top flight, meaning that **Sassuolo** will not have a better chance than this to get back on course. [...] In their last encounter in the Serie A **Verona** outplayed **Sassuolo** 4-2, meaning that the hosts are hungry for revenge.*

  "Neroverdi" word means a nickname of "Sassuolo" that isn't known by the vocabulary. So, I replaced all nicknames found in the texts with "home team" or "away team" words. In order to do this I created a json (synonyms.json) where objects have as key the team name and as values all the possible nicknames related to the team.
- punctuation and numbers removing using regex
- tokenization: it allows to split each document into tokens. Nltk is the tokenization chosen because it reached the highest performance
- stop word filtering: words belonging to the "stopword set" are removed because they are considered noisy words
- stemming: it allows to get the root word of each token.

These tokens have to be transformed into vectors of numbers and in order to do that I compared two different approaches:

- **CountVectorizer**: convert a collection of text documents into a matrix of token counts. The ML model I used with it is Logistic Regression
- **TfIdfVectorizer**: convert a collection of raw documents into a matrix of TF-IDF features- The ML model I used with it is Decision Tree

The vectorization is done combining a trigrams approach which allow to reach an important accuracy rate. After that, documents in the vectorized matrix are shuffled and then splitted into training set (80%) and test set (20%).

To realise the model that allows to analyse and classify text, different algorithms (for both vectorizer) were used: Logistic Regression, Decision Tree, Random Forest, Support Vector Machine, Multinomial Naive Bayes.

There are 2 machine learning methods which have been used:

- fit: it allowed to train the model passing the train set with the relative labels (the predictive label published with the news of a match)
- predict: it allowed to predict the belonging class of a match passing only the test set without any label

The highest accuracy was reached with Logistic Regression combined with CountVectorizer with a score of 85.27% but considering the accuracy of the final classification 2.2, the model providing the best accuracy is Decision Tree combined with TfidfVectorizer with a 83.12% accuracy score.

Both of these results are so good, considering that there are 3 possible classes (0, 1, 2). Without using trigrams in both methods the accuracy drops under 70%.

The best reached results are showed in the figure below:

ML model	Vectorizer	Accuracy	Avg Precision (macro)	Avg Recall (macro)	Avg F1-score (macro)	Avg Precision (weighted)	Avg Recall (weighted)	Avg F1-score (weighted)
Decision Tree	TfidfVectorizer	0.834356	0.836946	0.809376	0.819919	0.841132	0.834356	0.834376
Logistic Regression	CountVectorizer	0.852761	0.866070	0.824695	0.839814	0.857762	0.852761	0.850975

Fig. 2. Tfidf with a Decision Tree and CountVectorizer with LogisticRegression

2.2 Final classification - Random Forest

The final classification is done using Random Forest with TfidfVectorizer. The aim of this point is to build a model which can be used to predict outcomes o future matches.

With the Decision Tree model I obtained labels that I combined with the relative matches in the statistics dataset.

Random Forest works with numerical features, so names of teams have to be transformed into numbers: it is done using a dictionary created during the download phase (1.1.1).

The final dataset has a total number of 1627 rows and 15 features.

	home	result	goals	away	total_shots	shots_on_target	goals_on_penalty	total_penalties	percentage_possession	fouls	yellow_cards	red_cards	rank_h	rank_a	prediction
1635	28.0	2.0	-0.4	5.0	-3.2	-2.2	-0.2	-0.4	-0.8	0.0	-0.2	2.0	9.0	1.0
1636	4.0	2.0	-0.4	0.0	2.4	1.2	-0.2	-0.2	-1.6	0.0	0.0	0.0	12.0	1.0
1637	22.0	0.0	-0.8	19.0	1.6	-1.6	0.2	0.4	-0.6	0.2	0.0	0.0	12.0	0.0
1638	26.0	2.0	0.4	1.0	2.0	2.0	-0.2	-0.2	1.0	0.2	2.0	9.0		1.0
1639	24.0	2.0	-0.2	14.0	-0.6	-0.4	-0.4	-0.4						

Fig. 3. Example of 5 records of the final dataset

In the above picture there's an example of the last 5 records: the first teams involded are "hellas verona" (id 24) and "roma"(id 14).

As for the previous classification, different models have been evaluated finding that the model with the best accuracy was Random Forest with a 58.28% score. Models used in this phase were the same of the previous classification without Multinomial Naive Bayes because it needs only positive features.

Model	Accuracy	Avg Precision (macro)	Avg Recall (macro)	Avg F1-score (macro)	Avg Precision (weighted)	Avg Recall (weighted)	Avg F1-score (weighted)
Logistic Regression	0.533742	0.505796	0.488472	0.434876	0.511845	0.533742	0.470308
Support Vector Machine	0.524540	0.517694	0.476463	0.409420	0.519018	0.524540	0.448334
Decision Tree	0.423313	0.409175	0.409334	0.409149	0.423020	0.423313	0.423056
Random Forest	0.527607	0.481086	0.487283	0.451237	0.492672	0.527607	0.481866
K-Nearest Neighbors	0.475460	0.456255	0.453503	0.450160	0.469638	0.475460	0.467519

Fig. 4. Evaluation of the final models

In the below confusion matrix it's shown how many matches have been correctly classified:

- 14 draws (0 class) over 78 total draws
- 106 wins (1 class) over 140 total wins
- 70 loses (2 class) over 108 total loses

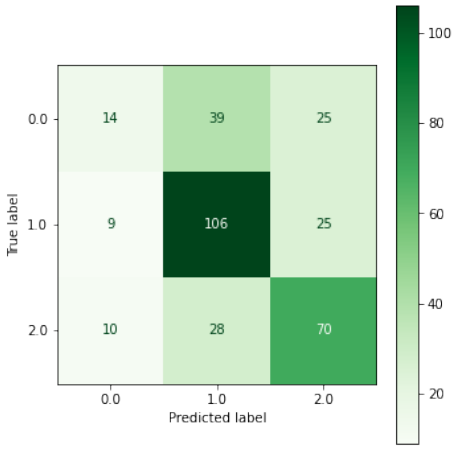

Fig. 5. Confusion matrix

The matrix shows how difficult it is to predict a draw outcome.

3 Outcome prediction

The last step is to get next matches and predict the outcome. Through the csv of the actual season is possible to keep track of the last played match, so it's easy to get the next matches. When these matches are obtained, different steps are followed:

- computation of the statistics as difference of the mean of the last 5 games of the two teams (1.1.2 section)
- computation of the rankings (direct confrontations won by two teams)
- scraping of the news of FootballPredictions (1.2.1 section)
- text vectorization (using **transform** method of TfidfVectorizer)
- news classification with Decision Tree (2.1 section)
- merge of statistics with the label classified from the previous step
- normalization of the team names (conversion of the names of the teams into numbers)
- final prediction with Random Forest model (2.2 section)

Taking into account matches of 8th matchday of Serie A, the model has correctly predicted 7 out of 10 games. The final result, really obtained in that matchday are:

Match	**Score**	**Result**	**Predicted**	**Right or Wrong result**
Empoli - Milan	1 - 3	2 (P)	2	Right
Inter - Roma	1 - 2	2 (P)	1	Wrong
Napoli - Torino	3 - 1	1 (V)	1	Right
Juventus - Bologna	3 - 0	1 (V)	1	Right
Lazio - Spezia	4 - 0	1 (V)	1	Right
Lecce - Cremonese	1 - 1	0 (N)	1	Wrong
Atalanta - Fiorentina	1 - 0	1 (V)	1	Right
Sassuolo - Salernitana	5 - 0	1 (V)	1	Right
Sampdoria - Monza	0 - 3	2 (P)	2	Wrong
Hellas Verona - Udinese	1 - 2	2 (P)	2	Right

References

1. Yangtuo Peng, Hui Jiang. *Leverage Financial News to Predict Stock Price Movements Using Word Embeddings and Deep Neural Networks* (2015). (link)
2. Dante Sblendorio. *Predicting NFL Games* (2021). (link)
3. President Sean Forman. *FBref* (2004) (link)
4. *FootballPredictions* (link)
5. Corentin Herbinet. *Predicting Football Results Using Machine Learning Techniques* (2018). (link)
6. Adalberto Tardelli, Meide S Anção, Abel L. Packer, Daniel Sigulem. *An implementation of the trigram phrase matching method for text similarity problems* (2004). (link)