Data information quality project

Computer Science and Engineering

Author(s): **Mattia Brianti**
Alex Hathaway

Academic Year: 2025-26

Contents

1 | Setup Choices

For this project we decided to use Colab Notebook to make it easier for us to collaborate. We relied mainly on pandas to load, inspect, clean, and reshape the dataset, since it provides an efficient and flexible DataFrame structure. To handle missing values and numerical operations we used numpy, especially for working with np.nan. Datetime conversions were managed through pandas, with occasional support from Python's built-in datetime module.

For exploration, we considered ydata-profiling to generate a quick and detailed HTML report. To visualize distributions and detect potential anomalies, we used seaborn for its statistical plots and matplotlib.pyplot for customization.

In the analysis phase, we relied on scikit-learn. We scaled features using RobustScaler to reduce the impact of outliers and applied LocalOutlierFactor to detect unusually different player profiles by comparing each record to its closest neighbors. For deduplication we used recordlinkage, which allowed us to identify potentially repeated player entries based on multiple attributes. Finally, we also experimented to predict the Overall feature, always using RobustScaler and a simple KNeighborsRegressor to show how data cleaning affects predictive performance. For evaluation, ShuffleSplit and cross_val_score helped ensure reliable cross-validation.

2 | Data preparation pipeline

The project consists of taking a dirty dataset and cleaning it using a structured data preparation pipeline. Starting with data profiling and data quality assessment, we first identify the main issues present in the raw data. We then apply a sequence of cleaning steps, standardization, error detection and correction, and finally deduplication to progressively improve the dataset before using it for a data analysis.

2.1. Data Exploration

We started by loading the FIFA dataset, which we discovered includes almost 19k players and 77 features. The dataset mixes numeric attributes (ratings, stats, IDs. . .) and categorical ones (names, nationality...), with a clear predominance of numeric columns. We also picked random players using the .sample() function to see if any alarming values would appear to us. Finally, we did a check on the two columns regarding the name of the players to see if one of the two could be dropped. Turns out that 261 players don't have their 'Name' inside their 'LongName' (e.g. Casemiro, Fabinho, Rodri...). This means that none of the two columns could be dropped. We decided not to drop 'LongName' column because some game features require it.

2.2. Data Profiling

For the data profiling phase, we used the ydata-profiling library, which produces a detailed interactive HTML report. This report allows us to explore every column in the dataset—its type, distribution, descriptive statistics, and more—through an intuitive interface. It also automatically highlights potential issues such as missing values, low uniqueness, and high correlations. To better understand the impact of our cleaning process, we generated two separate profiling reports: one for the original dirty dataset and one for the cleaned version.

2.3. Data Quality Assessment

A quick duplicate check confirmed that no rows are fully duplicated, so we are sure that each row is different from one another. To properly evaluate completeness, we first standardized common placeholder strings (like "Na", "–", "?", etc.) to pd.NA. After doing this, we computed the completeness of the dataset by dividing the number of non-null cells by the total number of cells, which showed that the dataset is 98.6% complete. The only two columns with notable gaps were expected cases, such as:

1. 'Loan Date End', which is mostly empty (about 5% complete) because most players aren't on loan.
2. 'Hits', with around 86% completeness, since not all players have been researched in the game database.

2.4. Data Cleaning

2.4.1. Data Transformation/Standardization

The initial state of the dataset contained numerous inconsistencies in formatting, units of measurement, and column nomenclature. To ensure semantic consistency and interoperability for downstream analysis, some transformations were implemented. We started with modifying the name of the columns to improve transparency for non-technical users. Ambiguous column names were renamed to descriptive labels (e.g., ↓OVA to ↓Overall, POT to Potential, DRI to Dribbling etc.). Furthermore, we made sure the dataset was sorted by the ↓Overall score in descending order to prioritize high-profile and known players during manual inspection.

Then we applied string manipulation techniques to clean textual data:

- Club Names: Leading whitespace characters (e.g., \n\n\n\nFC Barcelona) were stripped to ensure clean string matching.
- Attributes: Special characters, such as the star symbol ('⋆') in columns like Weak Foot, Skill Moves, and International Reputation, were removed to convert these features into pure integer types, for possible follow-up analysis.

After that we fixed the 'Contract' and 'Loan date end' columns. The 'Contract' column contained heterogeneous formats representing distinct player statuses ("Free", "On Loan", or date ranges like "2018 ∼ 2022"). A custom extraction logic was applied to decompose this field into three structured columns:

- Contract start and Contract End (Integers)
- Player Status (Categorical: 'Free Agent', 'On Loan', 'Permanent').

For "On Loan" players, the missing 'Contract End' date was imputed using the 'Loan Date End' column.

At the end of these manipulations we dropped the 'Loan Date End' and 'Contract' columns, which were substituted by: 'Contract Start', 'Contract End' and 'Player Status'.

The 'Joined' column was standardized to a datetime object.

Then, we proceeded to standardize the Height column, which contained inconsistent units of measurement. First, values expressed in feet were converted into centimeters; subsequently, the "cm" string was removed from the cells, and finally the column was renamed 'Height (cm)'.

The same procedure was applied to the 'Weight' column, using the appropriate unit conversions and measurements.

Subsequently, we addressed the financial columns 'Value', 'Wage' and 'Release Clause'. These columns contained strings with currency symbols (such as '€') and magnitude suffixes ('M' for millions and 'K' for thousands). All values were cleaned and converted into floating-point numbers, with Value and Release Clause expressed in millions of euros and Wage expressed in thousands of euros.

The last thing we did was to standardize the column 'Hits' which contained the magnitude suffix 'K'.

2.4.2. Error detection and correction of the missing values

The only columns were we had missing values were 'Contract Start', 'Contract End' and 'Hits'. For the first 2 we had null values for "Free Agent" players, so we replace them with "-1" as a sentinel value signaling that the player doesn't have a contract. We didn't drop the rows because these are still players that can be used in the game;

For what regards the 'Hits' column the null values represents players who had never been searched in the database. These were logically imputed with 0 since it means the player has never been searched.

2.4.3. Error detection and correction of outliers

During the data profiling phase, we observed a distinct disparity in numerical domains. While the vast majority of attributes (such as 'Finishing') are normalized within a standard 0–100 range, a select few columns exhibit significantly higher values (e.g., 'Attacking' ranges from 42 to 437). Given that the 0–100 scale is the predominant format for this dataset, we hypothesized that these larger columns were the sum of the previously mentioned ones. Consequently, we validated whether summary attributes—specifically Attacking, Skill, Movement, Power, Mentality, Defending, and Goalkeeping—were the exact mathematical sums of their underlying components.

The validation confirmed that all summary columns strictly matched the sum of their constituent parts (e.g., Attacking = Crossing + Finishing + ...), indicating no arithmetic errors in the raw data.

Consequently to continue addressing data anomalies we firstly applied the Local Outlier Factor algorithm to the whole dataset. It was applied to scaled numerical features to identify density-based outliers. While the algorithm correctly flagged extreme cases (e.g., Lionel Messi due to high stats, Kazuyoshi Miura due to age), manual inspection confirmed these were valid "superstars" or rare demographic instances rather than data entry errors.

Therefore, no rows were dropped.

A second outlier analysis was conducted after observing, during the data profiling phase, that several columns, such as 'Skill' and 'Attacking' exhibited a bimodal distribution, with the lower-value peak initially appearing anomalous. However, domain knowledge suggests that this behavior is not due to data errors but reflects a structural characteristic of football data, which is the distinction between goalkeepers and outfield players. This hypothesis was validated using Kernel Density Estimation (KDE) plots and by stratifying the dataset with a custom Player_Type flag (Goalkeeper vs. Outfield Player), which showed that each peak corresponds to a distinct player type rather than to data anomalies.

2.4.4. Data deduplication

To ensure that no player appeared multiple times in the dataset due to data entry redundancies, we employed a record linkage approach using the recordlinkage library. The methodology began with a SortedNeighbourhood indexer applied to the 'Long Name' column with a window size of 9, which efficiently generated candidate pairs of potentially similar records. Subsequently, we computed a detailed comparison vector involving the Jaro-Winkler distance for 'Long Name' and 'Club' (with a threshold of 0.8), exact matching for the 'Age' column, and non-exact numeric functions for physical and technical attributes to tolerate minor data entry variations. Pairs exceeding a similarity threshold of 8 out of 11 attributes were flagged as potential duplicates. However, a manual inspection of these high-confidence matches revealed that they were false positives; even if the flagged players shared some features we saw that they were completely different people. Consequently, no records were removed, ensuring the preservation of unique entities.

3 | Data analysis

To evaluate the effectiveness of the data quality pipeline, we carried out a comparative analysis to observe how the cleaning process influences predictive modeling performance. We set up a regression task to predict the 'Overall' rating of players based on their attributes, selecting a K-Nearest Neighbors (KNN) Regressor. The comparison involved the dirty dataset, which is the raw csv with the removal of some useless columns and the cleaned dataset. Regarding pre-processing, both datasets were treated with One-Hot Encoding for categorical variables and RobustScaler normalization to mitigate the impact of extreme values. We evaluated the models using ShuffleSplit Cross-Validation strategy, relying on the Root Mean Squared Error (RMSE) to determine if the cleaned dataset provided a more reliable result.