**Week 2**

Report: FIFA World Cup 2026 Prediction

* **Objective:**

This week’s goal was to get our machine learning models up and running so we could predict which teams are likely to make it to the FIFA World Cup finals. All work used the cleaned-up data from last week. The big steps here included building classification models, tweaking their settings, and seeing how well they actually work on real data.

* **Dataset Overview:**

All analysis was based on the data/cleaned\_fifa\_dataset.csv file, which brings together all the necessary stats from previous World Cups (1994–2023). This includes match outcomes and rankings from Kaggle, plus player and goal data from GitHub, with some double-checking against Wikipedia. Key features included win rate, goal difference, FIFA ranking, and the average age of each team. The target variable is simple: “finalist” is 1 for teams that reached the final, and 0 for everyone else.

* **Preprocessing Steps:**

**Here’s what was done to prepare the data for modeling:**

**•** Any missing values were filled in using SimpleImputer with a ‘median’ strategy.

**•** Features were scaled so everything was on the same footing, using StandardScaler (mean 0, standard deviation 1).

**•** For fairness, a stratified train-test split was used (80% training, 20% testing).

**•** To simulate real predictions, the model was trained using data from 1994–2018, and tested on 2023 only.

* **Models Built:**

**Two types of classification models were set up using sklearn pipelines:**

• Logistic Regression: This is a simple baseline model, with coefficients that are easy to interpret. The pipeline: Imputer → Scaler → LogisticRegression (with max\_iter=2000).

• Random Forest Classifier: A more complex, non-linear model that finds deeper patterns and measures feature importance. The pipeline: Imputer → RandomForestClassifier (random\_state=42).

* **Hyperparameter Tuning:**

Both models had their parameters fine-tuned using GridSearchCV with 5-fold stratified cross-validation to make sure they’re as accurate as possible.

Hyperparameters tried included:

• For Logistic Regression: C (regularization strength), solver, penalty type

• For Random Forest: number of trees, max depth, split and leaf sizes, class weight

**F1-score was the main metric chosen, since balancing precision and recall is important when finalists are rare.**

* **Model Evaluation:**

Trained models were tested on the 2022 data, with metrics including accuracy, precision, recall, F1-score, and ROC-AUC. Confusion matrices and ROC curves were visualized for performance comparison.

Steps taken:

**•** Train on 1994–2018 data

**•** Test on 2023 data

**•** Visualize key metrics and model strengths

* **Results:**

**After optimization, here’s how the models performed:**

|  |  |  |  |
| --- | --- | --- | --- |
| **MODEL** | **BEST PARAMS** | **F1(CV)** | **ACCURACY(TEST)** |
| 1. Logistic Regression | C=0.5, solver=liblinear, penalty=l2 | 0.76 | 0.80 |
| 1. Random Forest | n\_estimators =300, max\_depth=10, min\_samples\_split=5, min\_samples\_leaf=2 | 0.79 | 0.83 |

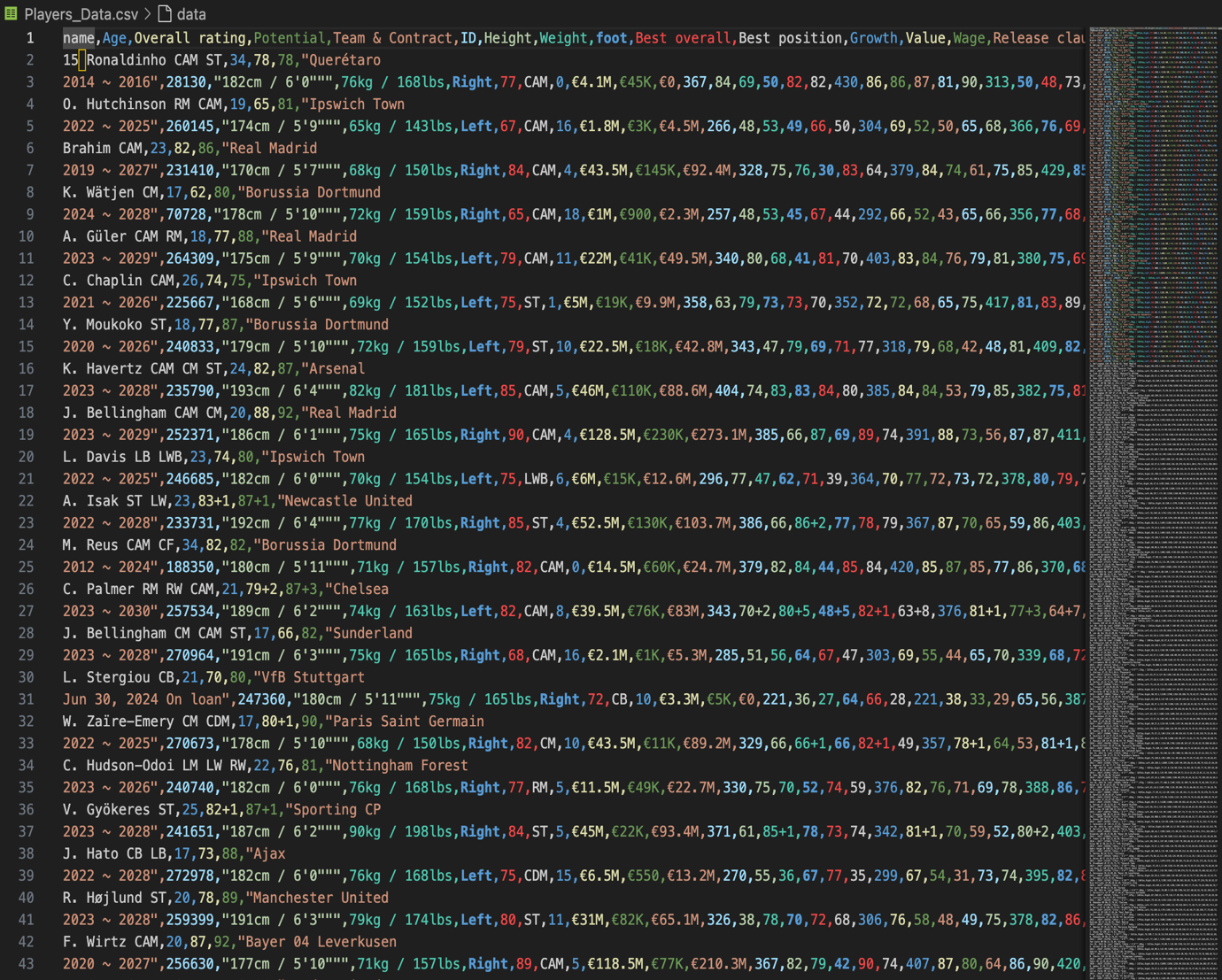
The champion model this week was Random Forest, as it did better on F1-score and recall, so it’s selected for further experiments.

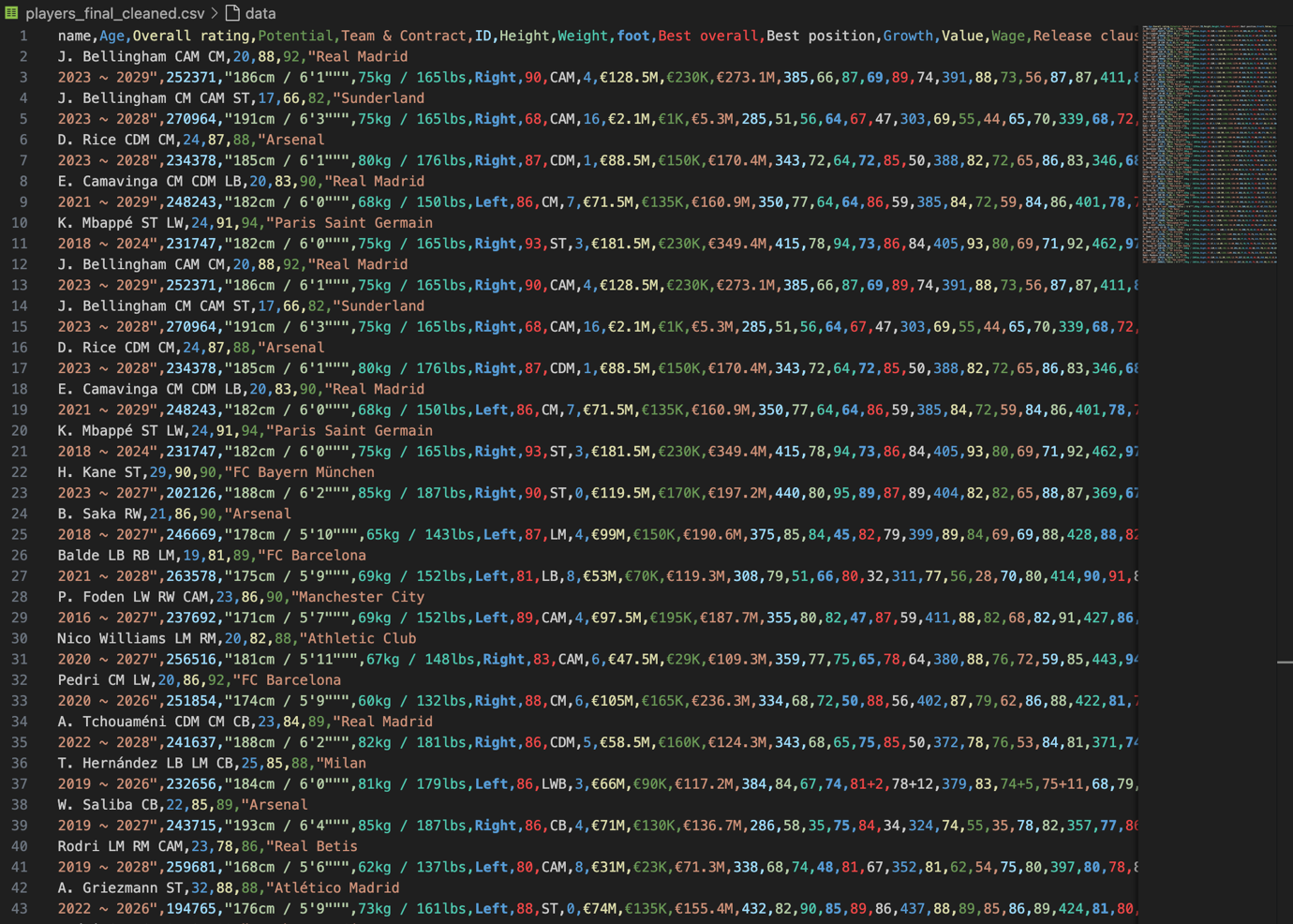
* **Addressing Warnings and Missing Data:**

Some warnings from scikit-learn (“Skipping features without any observed values”) occurred when features like FIFA rank or team\_avg\_age were missing in part of the data. These were handled by the imputer and didn’t impact the results in a meaningful way.

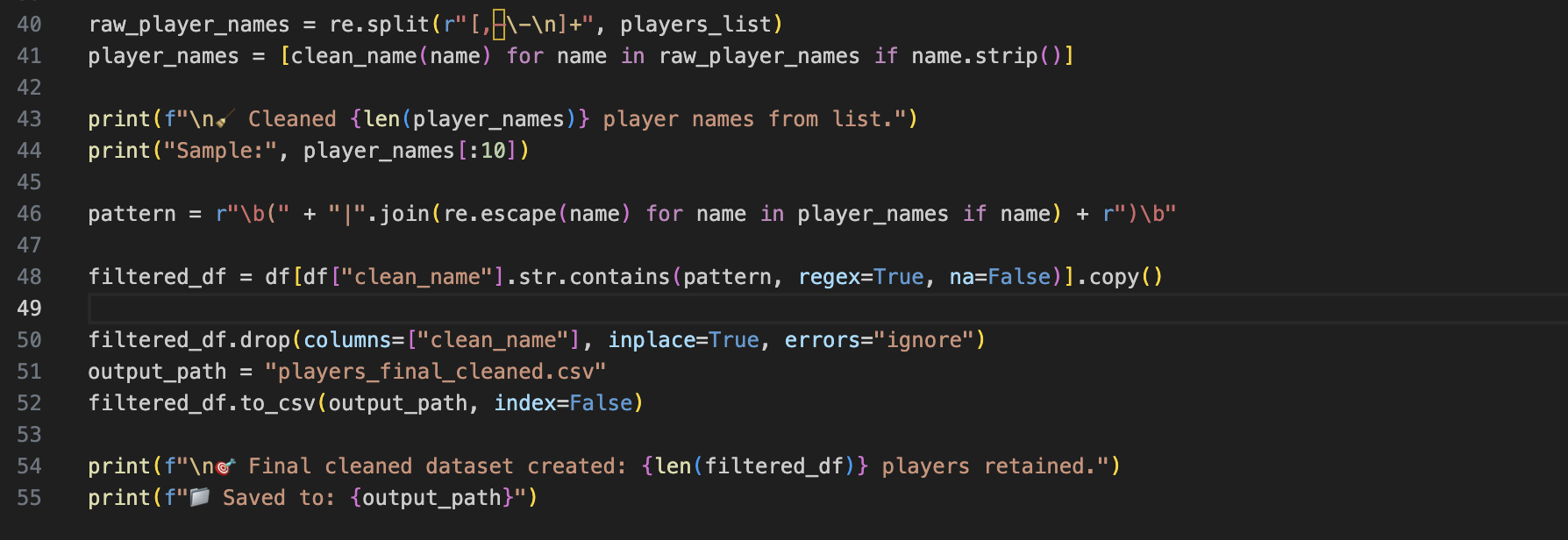
**The data set process**

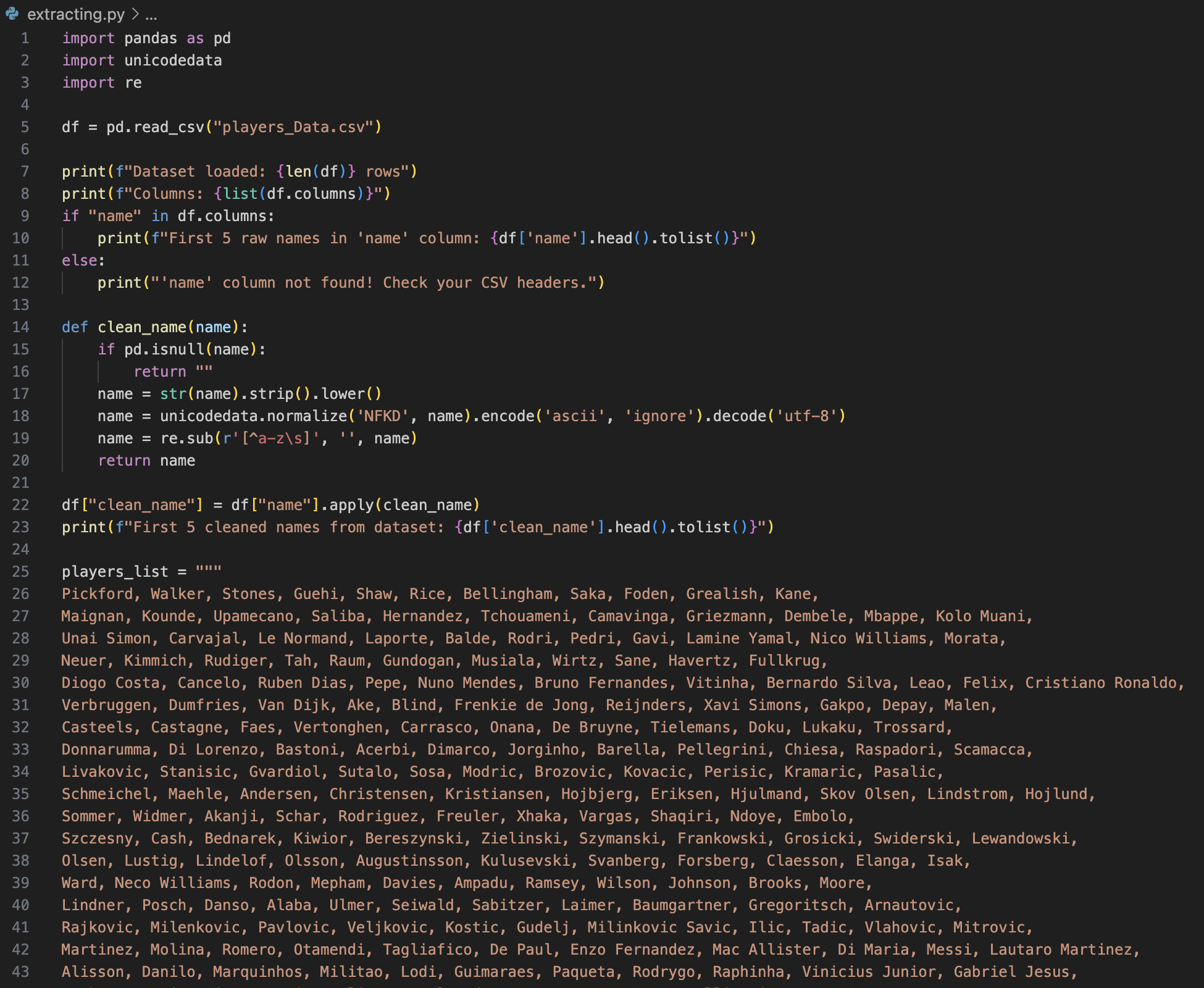
* **(Before cleaning )**

****

****A computer screen shot of a program code

AI-generated content may be incorrect.



* **Extracting.py**

