FDS Kaggle Competition

Pokémon Battles Prediction

Fundamentals of Data Science
Sapienza University of Rome
I Semester 2025-2026

TA: Leonardo Rocci

- B.Sc. in **Mathematics** @ SAPIENZA UNIVERSITÀ DI ROMA
- M.Sc. in **Data Science** (currently)
- My email: rocci.1922496@studenti.uniroma1.it

Welcome to the Arena

It’s not just a game

Have you ever watched a competitive match and tried to guess the winner early on? That's exactly what we're going to do.

- **Why Pokémon?** Behind the cute characters lies a deeply strategic and statistical system. A battle's outcome is decided by hundreds of interacting variables:
  - **Statistics**: Base stats, levels, and in-battle boosts.
  - **Strategy**: Team composition and type matchups.
  - **Momentum**: Turn-by-turn changes in health and status.

- **Your Mission:** is to train a machine learning model that can predict the winner of a battle using only the data from the first 30 turns.

Pokemon 101

From Game Mechanics to Model Features

To predict the winner, you need to understand the "features" of a battle. Here are the essentials:

- **Core Stats (The Raw Numbers):** Every Pokémon is defined by six base stats. These will be your primary numerical features.
  - **HP (Hit Points):** How much damage a Pokémon can take.
  - **ATK / DEF:** Physical Attack and Defense power.
  - **SPA / SPD:** Special Attack and Special Defense power.
  - **SPE (Speed):** Determines who attacks first each turn.
- **Types & Matchups (The Core Strategy):** This is where most of the strategy comes from.
  - Every Pokémon and every move has at least one **Type** (e.g., Fire, Water, Grass).
  - These types follow a rock-paper-scissors system. For example, a Water attack on a Fire Pokémon is "super effective" (2x damage).
  - You might consider quantifying type advantages across both teams.
- **Status Conditions (The Game Changers):**
  - Moves don't just do damage. They can apply status effects you will see in the data, which lowers speed and can cause a Pokémon to miss its turn.

*To learn more about battle mechanics, check out **Smogon's** Introduction to Competitive Pokémon.*

The challenge

A binary classification problem. Easy right?

Let’s make it all a bit more formal.

The Objective: Formally, your goal is to build a model that solves a **binary classification** task.

The Target Variable (y):

- You are predicting the ***player_won*** feature, which is missing in the test dataset.
- This is a boolean (true / false), which you should map to a numerical format: $y \in \{1,0\}$.

The Features (X): Your input data is everything else! We can group the features into two types:

- **Static Features (Pre-Battle Info):** Data that is known at turn 0.
  - ***p1_team_details***: The stats and types of all 6 Pokémon on Player 1's team.
  - ***p2_lead_details***: The stats and types of the **one** Pokémon the opponent starts with.
- **Dynamic Features (Live Battle Info):** Data that changes from turn to turn.
  - ***battle_timeline***: A sequence of HP percentages, status changes, and moves used over 30 turns.

So… is it easy?

The classification task is standard. The real challenge is mastering the data by parsing its complex structure, managing mixed types, and discovering powerful features. Ultimately, all that work needs to translate into a model that performs well and **climbs the leaderboard**.

About the Data...

Not your usual CSVs files

Both the train and test files are in the **.jsonl (JSON Lines)** format.

Don't be intimidated by the extension, it's simply a text file where each line is a complete JSON object.

What is JSON?

- JSON (JavaScript Object Notation) is a lightweight format for storing and transporting data. Its job is to store information in a structure very similar to a **Python dictionary**.
- It uses **key-value pairs**.
- Values can be other dictionaries or lists, which allows for a **nested structure**.
- This makes it ideal for storing the complex, layered information of a Pokémon battle all in one place.


```
{
  "DocumentType": 1,
  "No.": "S-ORD101001",
  "SellToCustNo": "10000",
  "PostingDate": "2023-04-02",
  "Lines": [
    {
      "LineNo": 10000,
      "Type": 2,
      "No": "1996-S",
      "Quantity": 12,
      "UnitPrice": 1397.3
    },
    {
      "LineNo": 20000,
      "Type": 2,
      "No": "1900-S",
      "Quantity": 4,
      "UnitPrice": 192.8
    }
  ]
}
```

Live Demo

Roadmap to your first submission

- Join the Competiton here using your university email (or the one you entered here)
- Make sure to be linked with your teammates
- You will find a starter notebook in the code section of the competition
- Alternatively, you can find it here

Validation

The problem of Optimism

- **The Trap**: If you evaluate your model on the same data it was trained on, its score will be misleadingly high because the model has already "seen the answers."

- **Optimistic Bias**: This effect has a technical name: Optimism. It's the measure of how much better your model performs on training data versus new, unseen data. This bias exists even with a perfect training process but gets much worse with overfitting.

- **The Practical Solution**:
  - While you can theoretically estimate and correct for optimism, this is an advanced technique that is often impractical and risky, as the correction itself can be unreliable.
  - To get an honest evaluation, the practical and sound way is to use a **Train/Test Split**. You train on one part and get your final, unbiased score on the other.

- **In This Competition**: This split is already done for you. You are given the Training Set (***train.jsonl***). For the Test Set, you are provided with the features, while Kaggle holds the "**ground truth**" (the actual ***player_won*** labels) for the final Private Leaderboard scoring.

Validation

Making Choices

- **A New Problem**: How do you make choices like feature/model selection or hyperparameter tuning?
- You can't use the final Test Set, as this would "leak" information and make your final score biased again.
- **The Solution: A Validation Set**. You must create your own "practice exam" by splitting your ***train.jsonl*** file. You use this validation set to compare models and make all your decisions.
- **The Kaggle Trap**: The Public Leaderboard is NOT your validation set. It's scored on a smaller, fixed portion of the test data (30%). If you make decisions based on this score, you will likely overfit to that small slice of data, and this risk gets higher with every choice you make.
- **The Good Practice**: Trust your local validation process. Use the Public Leaderboard only as a sanity check to see if your local scores are moving in the same general direction.

Validation

The Gold standard and its limits: Cross-Validation

- **The Best Practice**: Instead of one "practice exam," it's more reliable to create k of them and average the results. This is **k-fold Cross-Validation**. The average score is a much more robust estimate of your model's true performance.

- **When to Use It**: CV is the best practice for most models (e.g. Logistic Regression, Gradient Boosting). It should be your most trusted guide for this competition.

- **Practical Limits**: For computationally expensive models (e.g. Neural Networks), training k times can be too slow. In that case, you could fall back to using a single, fixed validation set.

The Rules

Fair play

- **Submission Limits**: You may submit up to **3 entry per day**. Before the final deadline, you must select **3 submissions** to be used for final scoring.

- **Public vs. Private Leaderboard**: The leaderboard you see during the competition is the **public leaderboard**, scored using **Accuracy** on 30% of the test data. Your final rank is determined by the **private leaderboard**, which uses the other 70%. This prevents "overfitting" to the leaderboard, so trust your validation process!

- **Data Usage**: You must train your models exclusively on the provided ***train.jsonl*** file. The use of external data or models pretrained for this task is strictly prohibited.

- **Collaboration**: Discussing ideas and strategies on the forum is encouraged. However, **private sharing of code between teams is not permitted**.

The Rules

Evaluation

Your final grade for this project is based on two key components:

- **Performance:** Your final performance on the private leaderboard will be a key factor in your evaluation. A higher Accuracy score will contribute positively towards your final grade.
- **Methodology:** A **Report** detailing your complete methodology. This is your chance to showcase the breadth and depth of your work, and it should cover areas such as:
  - **Feature Engineering & Selection:** Explain the features you created, why you believed they would be predictive, and the methods you used to select your final feature set.
  - **Model Experimentation and Refinement:** Discuss the variety of modeling approaches you explored. Detail your process for optimizing your models and the techniques you used to improve upon baseline performance, from simple models to potentially more complex structures.
  - **Validation:** Describe the validation strategy you used to make your choices and analyze its effectiveness in predicting your final performance on the public and private leaderboards.

Code Reproducibility is mandatory. After the deadline, you must submit your code. It must be able to reproduce the results of your final submissions. Well-commented code is appreciated.

Timeline

Key Dates

- **Competition Starts**: Monday, October 13
- **Competition Ends**: Friday, November 14

Some tips

How to climb the leaderboard

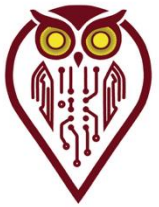
- **Understand the Game First:** Before diving into the code, take some time to understand the domain. The best features often come from a solid insight into Pokémon's core strategy and mechanics.

- **Build a Simple Baseline First**: Your initial goal should be to get a working submission pipeline and a score on the board. The provided starter notebook is one way to accomplish this, but you might want to start form scratch.

- **Iterate and Add Complexity:** Once the baseline works, improve it incrementally. This is where you should explore creative feature engineering and apply the new models and techniques as you learn them throughout the course.

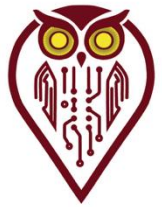
- **Trust Your Local Validation, not the LB:** Your local cross-validation score is your compass, the public leaderboard is just a sanity check.

SAPIENZA
UNIVERSITÀ DI ROMA



Any question?

SAPIENZA
UNIVERSITÀ DI ROMA



Best of luck!