**Spiegazione dei singoli dataset**

I dataset sono stati ricavati dalla piattaforma Kaggle, riguardano i pokemon e i loro combattimenti.

Dai dataset presi in considerazione possiamo ricavare le informazioni necessarie a scoprire quale pokemon può vincere in un combattimento con un altro in base alle sue statistiche.

I dataset utilizzati per il progetto sono 4:

1. pokemon.csv, che contiene l’elenco di tutti i pokemon e di tutte le loro statistiche;
2. combats.csv, che contiene alcuni dei possibili combattimenti tra i vari pokemon di cui conosciamo già il risultato (che verrà poi utilizzato per fare il training);
3. test.csv, contiene altri possibili combattimenti dei pokemon di cui non conosciamo ancora il risultato (che verrà poi utilizzato per fare il testing);
4. pokemonTypeComp.csv, che contiene le relazioni di forza e debolezza tra i possibili “tipi” di pokemon.

**Dettagli dei dataset**

|  |  |  |
| --- | --- | --- |
| **combats.csv** | | |
| **Attributi** | **Tipo dei dati** | **Contenuto attributo** |
| First\_pokemon | int | Id del primo pokemon rivale |
| Second\_pokemon | Int | Id del secondo pokemon rivale |
| Winner | Int | Id del vincitore tra i due rivali |

|  |  |  |
| --- | --- | --- |
| **Test.csv** | | |
| **Attributi** | **Tipo dei dati** | **Contenuto attributo** |
| First\_pokemon | int | Id del primo pokemon rivale |
| Second\_pokemon | Int | Id del secondo pokemon rivale |

|  |  |  |
| --- | --- | --- |
| **pokemon.csv** | | |
| **Attributi** | **Tipo dei dati** | **Contenuto attributo** |
| id | int | Id |
| Name | chr | Nome |
| Type.1 | chr | Primo tipo del pokemon tra i 18 possibili: Normal, Fire, Water, Electric, Grass, Ice, Fighting, Poison, Ground, Flying, Psychic, Bug, Rock, Ghost, Dragon, Dark, Steel, Fairy |
| Type.2 | chr | Secondo tipo del pokemon tra i 18 possibili: Normal, Fire, Water, Electric, Grass, Ice, Fighting, Poison, Ground, Flying, Psychic, Bug, Rock, Ghost, Dragon, Dark, Steel, Fairy |
| HP | int | Punti vita |
| Attack | int | Valore di attacco |
| Defense | int | Valore di difesa |
| Sp.Atk | int | Valore di attacco degli attacchi speciali |
| Sp.Def | int | Valore di difesa delle difese speciali |
| Speed | int | Valore della velocità |
| Generation | int | Generazione a cui appartiene il pokemon che può andare dalla prima all’ottava |
| Legendary | chr | Valore booleano(“False”o “True”) che indica se il pokemon è leggendario |

|  |  |  |
| --- | --- | --- |
| **pokemonTypeComp.csv** | | |
| **Attributi** | **Tipo dei dati** | **Contenuto attributo** |
| Attacking | char | Tipo del pokemon che attacca |
| Normal | int | Livello di efficienza di un “tipo” di pokemon rispetto ad un altro di tipo “Normal”. Il dominio dei valori è 2, 1, 0.5 e 0 |
| Fire | int | Livello di efficienza di un “tipo” di pokemon rispetto ad un altro di tipo “Fire”. Il dominio dei valori è 2, 1, 0.5 e 0 |
| Water | int | Livello di efficienza di un “tipo” di pokemon rispetto ad un altro di tipo “Water”. Il dominio dei valori è 2, 1, 0.5 e 0 |
| Electric | int | Livello di efficienza di un “tipo” di pokemon rispetto ad un altro di tipo “Electric”. Il dominio dei valori è 2, 1, 0.5 e 0 |
| Grass | int | Livello di efficienza di un “tipo” di pokemon rispetto ad un altro di tipo “Grass”. Il dominio dei valori è 2, 1, 0.5 e 0 |
| Ice | int | Livello di efficienza di un “tipo” di pokemon rispetto ad un altro di tipo “Ice”. Il dominio dei valori è 2, 1, 0.5 e 0 |
| Fighting | int | Livello di efficienza di un “tipo” di pokemon rispetto ad un altro di tipo “Fighting”. Il dominio dei valori è 2, 1, 0.5 e 0 |
| Poison | int | Livello di efficienza di un “tipo” di pokemon rispetto ad un altro di tipo “Poison”. Il dominio dei valori è 2, 1, 0.5 e 0 |
| Ground | int | Livello di efficienza di un “tipo” di pokemon rispetto ad un altro di tipo “Ground”. Il dominio dei valori è 2, 1, 0.5 e 0 |
| Flying | int | Livello di efficienza di un “tipo” di pokemon rispetto ad un altro di tipo “Flying”. Il dominio dei valori è 2, 1, 0.5 e 0 |
| Psychic | int | Livello di efficienza di un “tipo” di pokemon rispetto ad un altro di tipo “Psychic”. Il dominio dei valori è 2, 1, 0.5 e 0 |
| Bug | int | Livello di efficienza di un “tipo” di pokemon rispetto ad un altro di tipo “Bug”. Il dominio dei valori è 2, 1, 0.5 e 0 |
| Rock | int | Livello di efficienza di un “tipo” di pokemon rispetto ad un altro di tipo “Rock”. Il dominio dei valori è 2, 1, 0.5 e 0 |
| Ghost | Int | Livello di efficienza di un “tipo” di pokemon rispetto ad un altro di tipo “Ghost”. Il dominio dei valori è 2, 1, 0.5 e 0 |
| Dragon | int | Livello di efficienza di un “tipo” di pokemon rispetto ad un altro di tipo “Dragon”. Il dominio dei valori è 2, 1, 0.5 e 0 |
| Dark | Int | Livello di efficienza di un “tipo” di pokemon rispetto ad un altro di tipo “Dark”. Il dominio dei valori è 2, 1, 0.5 e 0 |
| Steel | int | Livello di efficienza di un “tipo” di pokemon rispetto ad un altro di tipo “Steel”. Il dominio dei valori è 2, 1, 0.5 e 0 |
| Fairy | int | Livello di efficienza di un “tipo” di pokemon rispetto ad un altro di tipo “Fairy”. Il dominio dei valori è 2, 1, 0.5 e 0 |

Nota: Il livelli di efficienza del “tipo” sono: 2 che corrisponde a “effective”, 1 che corrisponde a “normal”, 0.5 che corrisponde a “not too effective” e 0 che corrisponde a “no effect”.

**Integrazione dataset**

L’integrazione viene fatta sui dataset “pokemon.csv”,“combats.csv” e “pokemonTypeComp.csv” direttamente su quello dei combattimenti. Vogliamo infatti considerare la relazione tra le vittorie nei combattimenti, le statistiche dei vari pokemon e se il “tipo” possa influenzare in qualche modo il risultato.

Notare che gli id dei combattenti nel dataset “combats.csv”(e la stessa cosa vale per il dataset di training “test.csv”) sono gli stessi id utilizzati per identificare i pokemon nell’elenco completo, cioè il dataset “pokemon.csv”.

Comincio integrando in “combats.csv” il nome dei combattenti in base al loro id, poi considero le statistiche che identificano i pokemon e per ognuno degli sfidanti integro i parametri e ne calcolo le differenze. Dall’elenco dei pokemon ricavo anche il primo “tipo”, mentre non considero il secondo per il fatto che per alcuni pokemon non è presente il valore.

Aggiungo l’attributo “winner\_first\_label” il cui dominio è “yes” o “no” per sapere di ogni combattimento se il vincitore è il primo avversario e aggiungo l’ultimo parametro, utile per l’analisi successiva, cioè l’”advantage” che possiamo ricavare dal dataset “pokemonTypeComp.csv” considerando le resistenze e le debolezze tra i vari tipi di pokemon.

Infine il dataset integrato viene esportato nel file “integrated.csv”.

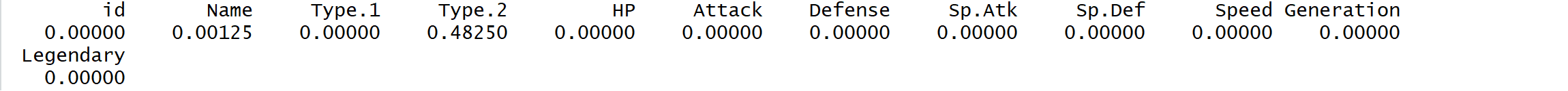
**Dimensioni di qualità per i dataset singoli**

Le dimensioni di qualità scelte per analizzare i dati sono: la completezza, che misura quanto un attributo è vuoto o ha valori nulli e l’unicità, che verifica se esistono duplicati all’interno dei dataset e sugli attributi.

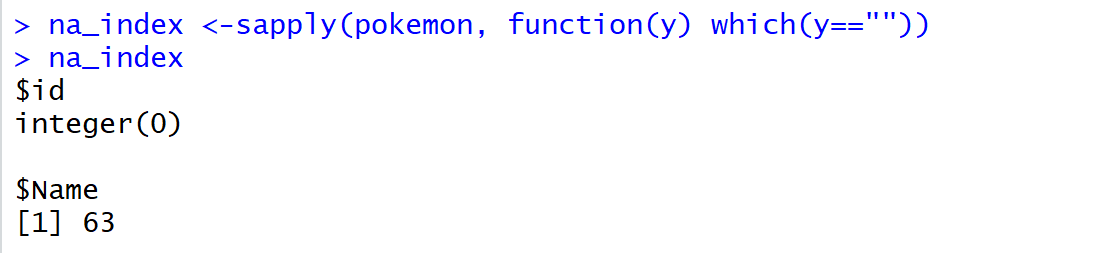
Per calcolare la completezza si fa il rapporto tra il numero di valori nulli e il numero di record per l’attributo considerato.

Calcoliamo quindi il valore di completezza per i singoli dataset: combats.csv, pokemon.csv, test.csv e pokemonTyperComp.csv.

Completezza pokemon.csv



Da questi valori si possono rilevare due fattori interessanti. Il primo è che all’interno del dataset c’è un mancanza, infatti nell’attributo “Name” non è presente il nome di 1 pokemon , per la precisione il 63, come si può ricavare dalla seguente funzione che mostra tutti i dati mancanti per ogni attributo.



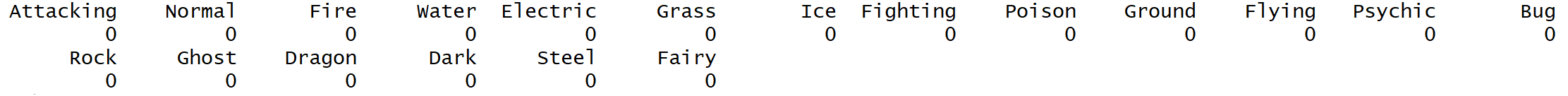
Il secondo è che per alcuni pokemon manca il secondo “tipo”, per questo è stato scelto di non considerarne l’influenza sui combattimenti quando vengono integrati i dataset.

Completezza combats.csv

In questo caso il dataset sui combattimenti non contiene alcun valore nullo.

Completezza test.csvPer il dataset per fare il testing non ci sono valori nulli.

Completezza pokemonTypeComp.csv



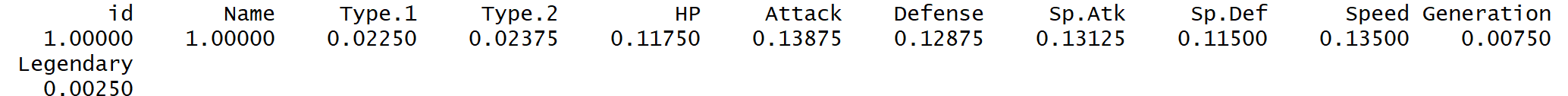
In questo caso i dati non mancano.

Invece per ottenere il valore di unicità per un dataset bisogna calcolare il rapporto tra il numero di valori unici e il numero dei record.

Calcoliamo quindi il valore di unicità per i singoli dataset: combats.csv, pokemon.csv, test.csv e pokemonTyperComp.csv.

C’è da sottolineare però che il valore di unicità ha senso sull’attributo identificativo del dataset o comunque su un attributo di cui valori devono essere unici. Per gli altri valori ha poco senso che venga calcolato.

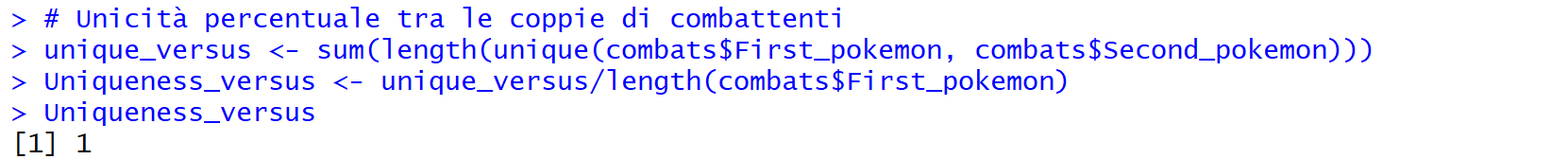
Unicità pokemon.csv

In questo caso è importante che sia l’attributo “id” che l’attributo “Name” abbiano valori unici essendo il dataset una lista completa di tutti i pokemon e come si può vedere dal risultato tutti i pokemon vengono identificati correttamente.

Unicità combats.csv

In questo caso il valore di unicità per ogni attributo non è molto utile perché dice solamente che alcuni pokemon effettuato più di una sola lotta, ma essendo il numero di records 50000 mentre quello dei pokemon 800 è naturale che si abbiamo delle ripetizioni sull’utilizzo dei combattenti.

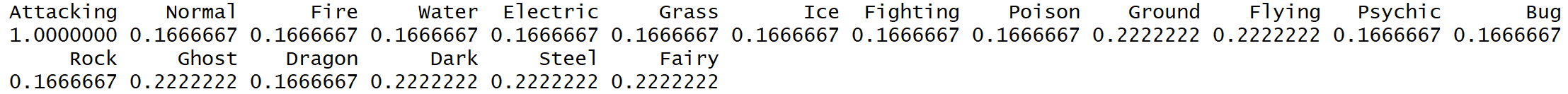
Però è molto più rivelante considerare l’unicità delle coppie di combattenti in modo da verificare che il nostro dataset non consideri più volte uno stesso combattimento.

Dal risultato ricavato dall’immagine si può notare che il valore di unicità per le coppie di combattenti è pari ad 1, quindi all’interno del nostro dataset tutti i combattenti vengono considerati una volta sola.

Unicità test.csv

L’analisi effettuata su “combats.csv” è la stessa che si deve considerare sul dataset per il testing.

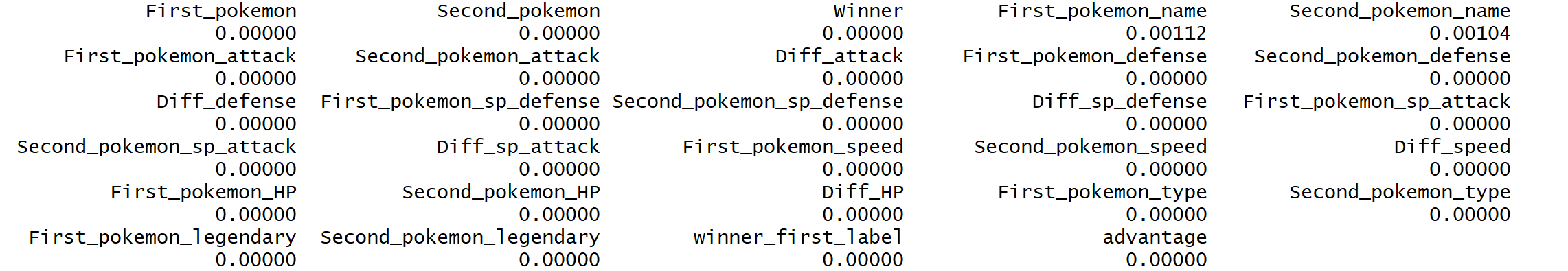
Unicità pokemonTypeComp.csv

In questo caso il valore di unicità è corretto per la colonna di Attacking, mentre ha poco senso considerare le altre colonne.

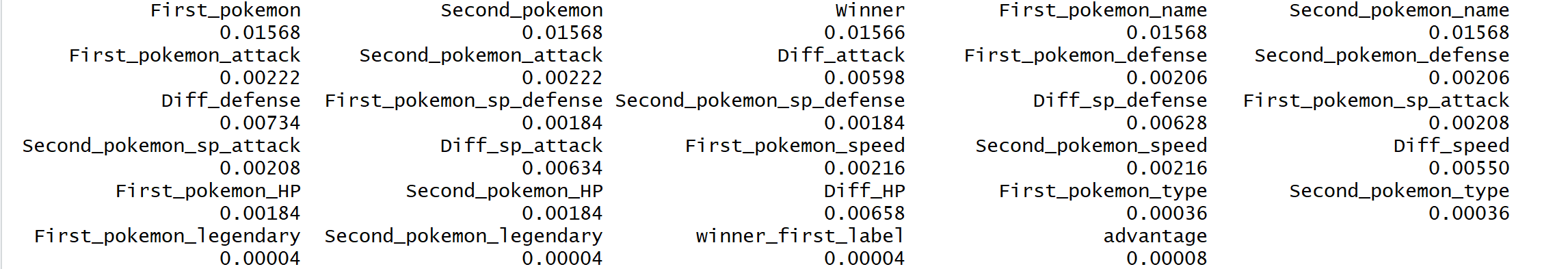
**Dimensioni di qualità su dataset integrato**

Anche per il dataset integrato sono stati considerati i valori di unicità e completezza.

Completezza per il dataset integrato

Come si può vedere dai risultati per il valore di completezza del dataset non è presente alcun valore vuoto o “NULL” per ogni attributo.

Unicità per il dataset integrato

In questo caso invece il calcolo dell’unicità ha poco senso farlo per tutti gli attributi, mentre ha senso calcolare, come si è fatto per il dataset “combats.csv”, l’unicità sulle coppie di combattenti.

Effettuando ancora il calcolo di unicità sulle coppie di pokemon si ottiene lo stesso risultato fatto su “combats.csv” ,cioè 1, quindi tutte le coppie sono uniche.