

# Cas Kaggle: Pokémons

Ferran Martínez Reyes - 1565491

[https://github.com/Ferran04/Cas\\_Kaggle](https://github.com/Ferran04/Cas_Kaggle)

# Introducció

- — —
- Base de dades de Pokémons
- Pokémon amb habilitats en funció del tipus (Normal, Aigua, Foc, Elèctric, Gel...) → **N\_tipus: 18**
- Cada Pokémon té un HP diferent, disminuït per la quantitat de dany de l'atac del contrincant



# Entenent La Base De Dades

---

Mida BD: 20 Atributs, 576 Pokémons

Atributs:

- Nom del Pokémon
- Número del Pokémon a la Pokédex
- 18 **multiplicadors de dany** que el Pokémon en qüestió pot influir sobre rivals en funció del tipus

No tenim la informació de quin tipus és cada Pokémon

# Entenent La Base De Dades

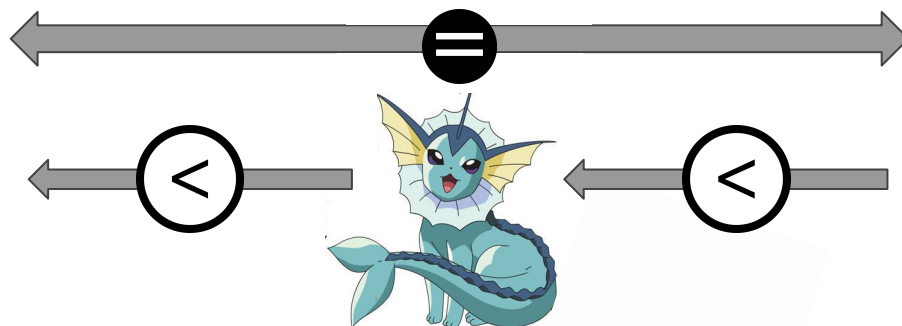
---

L'**efectivitat de l'atac** és un multiplicador sobre el dany de l'atac, **en funció del tipus** de Pokémon atacat i atacant.

Pokémon de foc (Flareon) més dèbil que Pokémon d'aigua (Vaporeon) i no més dèbils que els Pokémon de tipus elèctric (Jolteon)



Foc



Aigua



Elèctric

# Objectius

---

Per a què podem fer servir aquestes dades?

- ❌ • Donat un Pokémon saber per a quins rivals és més fort, més dèbil o igualat en força
- ❌ • Quins tipus de Pokémon són els més forts/débils en general?
- ✅ • Quins **tipus** de Pokémon són **els més similars entre si** en els atacs? → Útil per descartar similars i tenir un major abast d'atac en el nostre equip de Pokémons



**CLUSTERING**

# Analitzant i preparant les dades

```
pokemon_dataset.head()
```

	Name	Number	Normal	Fire	Water	Electric	Grass	Ice	Fighting	Poison	Ground	Flying	Psychic	Bug	Ro
0	Bulbasaur	#001	*1	*2	*0.5	*0.5	*0.25	*2	*0.5	*1	*1	*2	*2	*1	
1	Ivysaur	#002	*1	*2	*0.5	*0.5	*0.25	*2	*0.5	*1	*1	*2	*2	*1	
2	Venusaur	#003	*1	*2	*0.5	*0.5	*0.25	*2	*0.5	*1	*1	*2	*2	*1	
3	Charmander	#004	*1	*0.5	*2	*1	*0.5	*0.5	*1	*1	*2	*1	*1	*0.5	
4	Charmeleon	#005	*1	*0.5	*2	*1	*0.5	*0.5	*1	*1	*2	*1	*1	*0.5	

...

Hi ha “Numbers” de la Pokédex que es repeteixen, per tant, aquest atribut no ens serà gaire útil. Utilitzarem el “Name” com a identificador del Pokémon

# Analitzant i preparant les dades

---

- Treure símbols “\*” i “#” dels multipliers i els números
- Cap Valor NaN
- Repeticions Num\_ID en les files (1 mateix Pokémon amb formes diferents) -> Utilitzar el nom com a ID
- Creació d'un diccionari amb KEY=Name i VALUE=Multipliers
- Mirem tots els valors que poden prendre els multipliers i les freqüències d'aparició.

*0-----	3.71%	(INVULNERABLE)
*0.25-----	1.85%	(ATAC MOLT POC EFECTIU)
*0.5-----	19.99%	(ATAC POC EFECTIU)
*1-----	56.37%	(ATAC AMB EFECTIVITAT NORMAL)
*2-----	16.95%	(ATAC EFECTIU)
*4-----	1.12%	(ATAC MOLT EFECTIU)

# Clustering

— — —

- Mida màxima d'un equip: 6 Pokémons
- Objectiu: Formar equip amb Pokémons que no tinguin els multiplicadors molt semblants
- Resolució: Fer Clústers de Pokémons en funció dels seus multiplicadors, per seleccionar-los de clústers diferents



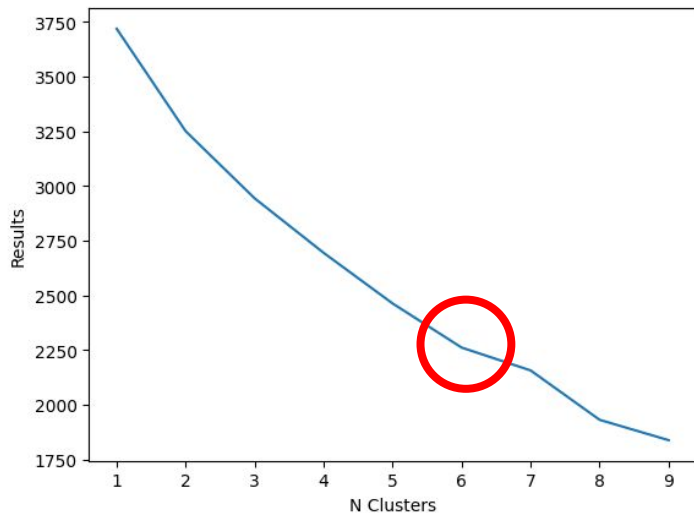
# Elbow Method

---

Anar provant amb diferent nombre de clústers i decidir-nos per aquella N on el pendent es ralenteix

```
from sklearn.cluster import KMeans
results = []
i = 1
for n_clusters in range(1, 10):
    model_kmeans = KMeans(n_clusters, random_state=0)
    model_kmeans.fit(pd.DataFrame(pokemon_data[:, 2:20]))
    results.append(model_kmeans.inertia_)
    print("provant amb ", i, " clusters")
    i += 1
```

Trobem Elbow a N = 6



# Assignar a cada Pokémon a un clúster

— — —

- Cada clúster té Pokémons amb multipliers molt semblants

```
# trobem el elbow a n = 6
kmeans_n6 = KMeans(n_clusters=6, random_state=0)
kmeans_n6.fit(pd.DataFrame(pokemon_data[:, 2:20]))
print(kmeans_n6.labels_)
```

```
[2 2 2 4 4 0 4 4 4 0 0 0 4 4 3 1 5 1 5 4 4 4 0 4 0 4 4 2 2 2 1 5 1 5 4 4 5
4 4 4 4 4 4 4 4 4 3 3 3 4 4 4 4 4 4 3 4 3 3 3 3 3 5 5 0 4 4 4 3 3 3 1 4 4
2 2 2 1 1 3 4 4 4 4 4 4 1 1 4 2 4 4 4 4 4 4 3 3 3 0 0 4 4 0 4 4 4 4 4 4
4 3 3 0 0 4 4 4 4 4 4 0 0 0 2 4 4 1 4 1 1 3 4 3 3 4 5 4 5 0 2 4 1 1 1 3 4
4 0 0 0 4 4 4 4 4 1 1 1 2 4 4 4 4 2 2 2 2 2 0 0 3 3 3 2 0 3 4 4 4 3
5 4 4 2 4 4 4 4 1 1 1 1 1 1 4 4 1 1 4 4 3 3 3 0 0 5 4 4 4 2 2 0 0 2 2
4 1 0 0 4 4 4 4 5 5 1 3 4 4 4 5 1 1 0 4 4 4 0 2 2 4 5 4 1 2 0 2 0 1 4 3 3
3 3 5 4 5 0 2 4 4 4 4 3 3 0 0 0 1 1 1 0 0 1 5 4 4 4 4 1 1 4 4 0 0 0 2 2
2 2 4 1 1 1 4 0 4 0 3 0 2 0 0 4 4 0 3 1 3 4 4 4 4 4 4 3 3 3 3 3 0 0 0 0
0 5 2 2 3 3 0 0 5 5 5 5 5 3 3 3 3 3 4 4 4 0 0 0 0 1 5 4 4 4 1 1 5 5 4 0 0
0 0 4 5 4 4 4 0 0 5 1 2 4 4 4 4 4 1 0 0 0 4 4 3 3 3 3 3 4 4 4 4 2 2 1 1 4
4 4 4 4 4 4 0 4 4 4 4 5 3 3 3 3 0 0 0 0 2 2 3 4 4 4 4 4 0 0 0 0 0 1 1 4
4 4 1 1 0 0 2 2 2 2 5 5 4 4 2 2 2 4 4 4 0 0 1 1 4 4 4 4 5 3 4 3 4 4 4 3 3
5 3 3 5 3 4 5 5 2 2 2 4 4 4 4 4 4 4 0 0 0 0 3 3 4 4 2 2 4 4 4 1 4 4 1 5
5 2 2 2 1 1 0 4 4 5 5 0 0 4 4 3 3 3 3 3 4 4 4 5 3 4 3 1 4 4 4 4 0 0 1 0
4 4 5 5 4 5 4 4 5 3 3 3 4 5 4 5 4 4 4 2]
```

# Conclusió

— — —

Pokemons al cluster 0

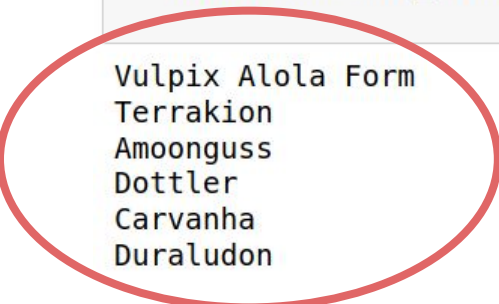
0	Bulbasaur
1	Ivysaur
2	Venusaur
5	Charizard
9	Caterpie
10	Metapod
11	Butterfree
27	Oddish
28	Gloom
29	Vileplume
57	Rapidash Galarian Form
64	Farfetch'd
74	Exeggcute
75	Exeggutor
76	Exeggutor Alola Form
88	Tentacool

Per fer el millor equip:  
seleccionar 6 Pokémons, un de  
cada clúster diferent

# Generador d'equips amb efectivitat d'atac màxima

— — —

```
#A continuacio el generador de equips  
import random  
for j in range(0, 6):  
    indexos = pd.DataFrame(pokemon_data_dict.values())[kmeans_n6.labels_ == j].index.values.astype(int)  
    index = indexos[random.randint(0, len(indexos) - 1)]  
    print(pokemon_data[index][0])
```



Vulpix Alola Form  
Terrakion  
Amoonguss  
Dottler  
Carvanha  
Duraludon

— — —

Ferran Martínez Reyes - 1565491

[https://github.com/Ferran04/Cas\\_Kaggle](https://github.com/Ferran04/Cas_Kaggle)