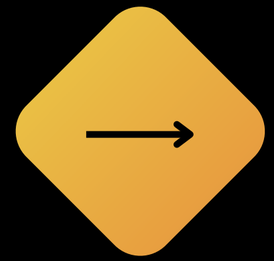
PROYECTO MÓDULO 2

Ciencia de Datos

Por Camila Plata



Características del dataset (Pokémon)

Datos y Registros

```
data.shape
```

```
(649, 15)
```

Nombre de las columnas

```
data.columns
```

```
Index(['nombre', 'indice_guia', 'resistencia', 'ataque', 'defensa',
       'tipo_primario', 'tipo_secundario', 'max_salud', 'tasa_captura',
       'tasa_escape', 'Weight', 'Height', 'Legendario', 'generacion',
       'fuerza_combate'],
      dtype='object')
```

De inglés a español

```
data.rename(columns= {"Weight": "peso", "Height": "altura", "Legendario": "legendario"}, inplace= True)
data
```

Eliminar columnas que no sirven

```
data.drop(columns = 'indice_guia', inplace=True)
```

nombre	resistencia	ataque	defensa	tipo_primario	tipo_secundario	max_salud	tasa_captura	tasa_escape	peso	altura	legendario	generacion	fuerza_combate

Limpieza de Datos

```
data.isna().sum()
```

```
nombre              0
resistencia         5
ataque              5
defensa             5
tipo_primario       5
tipo_secundario     5
max_salud           5
tasa_captura        5
tasa_escape         5
peso                0
altura              4
legendario          1
generacion          0
fuerza_combate      0
dtype: int64
```

```
data.dtypes
```

```
nombre              object
indice_guia          int64
resistencia        float64
ataque             float64
defensa            float64
tipo_primario       object
tipo_secundario     object
max_salud          float64
tasa_captura       float64
tasa_escape        float64
Weight             float64
Height             float64
Legendario          object
generacion           int64
fuerza_combate       int64
dtype: object
```

Datos NaN

Tipos de dato

Solución

```
nulo_legen = data["legendario"].isnull()
data.loc[nulo_legen]
```

peso	altura	legendario	generacion	fuerza_combate
21.2	0.71	NaN	2	677

```
data['legendario'].value_counts()
```

```
No    609
Sí     39
```

```
data.loc[nulo_legen, 'legendario'] = 'no'
data.loc[nulo_legen]
```

peso	altura	legendario	generacion	fuerza_combate
21.2	0.71	no	2	677

tasa_escape	peso	altura	legendario	generacion
0.09	40.6	NaN	No	1
0.15	10.0	NaN	No	1
0.10	2.3	NaN	No	3
0.10	60.5	NaN	No	4

```
data[data['altura'].isna()]
```

tasa_escape	peso	altura	legendario	generacion
0.09	40.6	NaN	No	1
0.15	10.0	NaN	No	1
0.10	2.3	NaN	No	3
0.10	60.5	NaN	No	4

```
altura_prom = data['altura'].mean()
print(altura_prom)

1.1505891472868215
```

```
nan = data['altura'].isna()
datos_nan = data.loc[nan]

display(datos_nan['altura'])

data.loc[nan, 'altura'] = altura_prom

data[nan]
```

tasa_escape	peso	altura	legendario	generacion
0.09	40.6	1.150589	No	1
0.15	10.0	1.150589	No	1
0.10	2.3	1.150589	No	3
0.10	60.5	1.150589	No	4

```
data[data['resistencia'].isna()]
```

nombre	resistencia	ataque	defensa	tipo_primario	tipo_secundario	max_salud	tasa_captura	tasa_escape	peso	altura	legendario	generacion	fuerza_combate
Togetic	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	3.2	0.61	No	2	1708
Lugia	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	216.0	5.21	Sí	2	3703
Cradily	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	60.4	1.50	No	3	2211
Garbodor	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	107.3	1.90	No	5	2345
Swanna	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	24.2	1.30	No	5	2088

```
data.isnull().sum()
```

```
nombre            0
resistencia       0
ataque            0
defensa           0
tipo_primario     0
tipo_secundario   0
max_salud         0
tasa_captura      0
tasa_escape       0
peso              0
altura            0
legendario        0
generacion        0
fuerza_combate    0
dtype: int64
```

```
data.dropna(inplace=True)
```

```
sns.countplot(data['legendario'])
data['legendario'].unique()
```

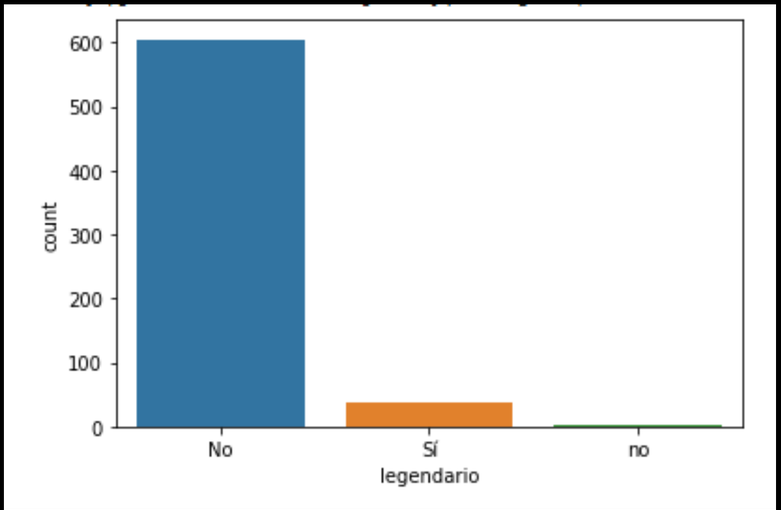


Gráfico de la columna 'legendario'

Cambiamos de palabras a numeros

```
data["legendario"].value_counts()
```

```
No    605
Sí     38
no      1
Name: legendario, dtype: int64
```

```
dict_legen={'Sí':1, 'No':0,'no':0}
data['legendario'].map(dict_legen)
```

```
data['legendario_bin'] = data['legendario'].map(dict_legen)
data.tail()
```

legendario	generacion	fuerza_combate	legendario_bin
Sí	5	3588	1
Sí	5	3575	1
No	5	3698	0
No	5	3972	0
No	5	3353	0

```
data.drop(columns=['legendario'], inplace=True)
data.head()
```

```
data["tipo_primario"].value_counts()
```

```
Water       99
Normal      89
Grass       61
Bug         60
Psychic     43
Fire        39
Electric    33
Rock        32
Ground      30
Poison      25
Dark        25
Fighting    22
Ice         21
Dragon      20
Ghost       19
Steel       18
Fairy        7
Flying       1
Name: tipo_primario, dtype: int64
```

```
data['tipo_secundario'].value_counts()
```

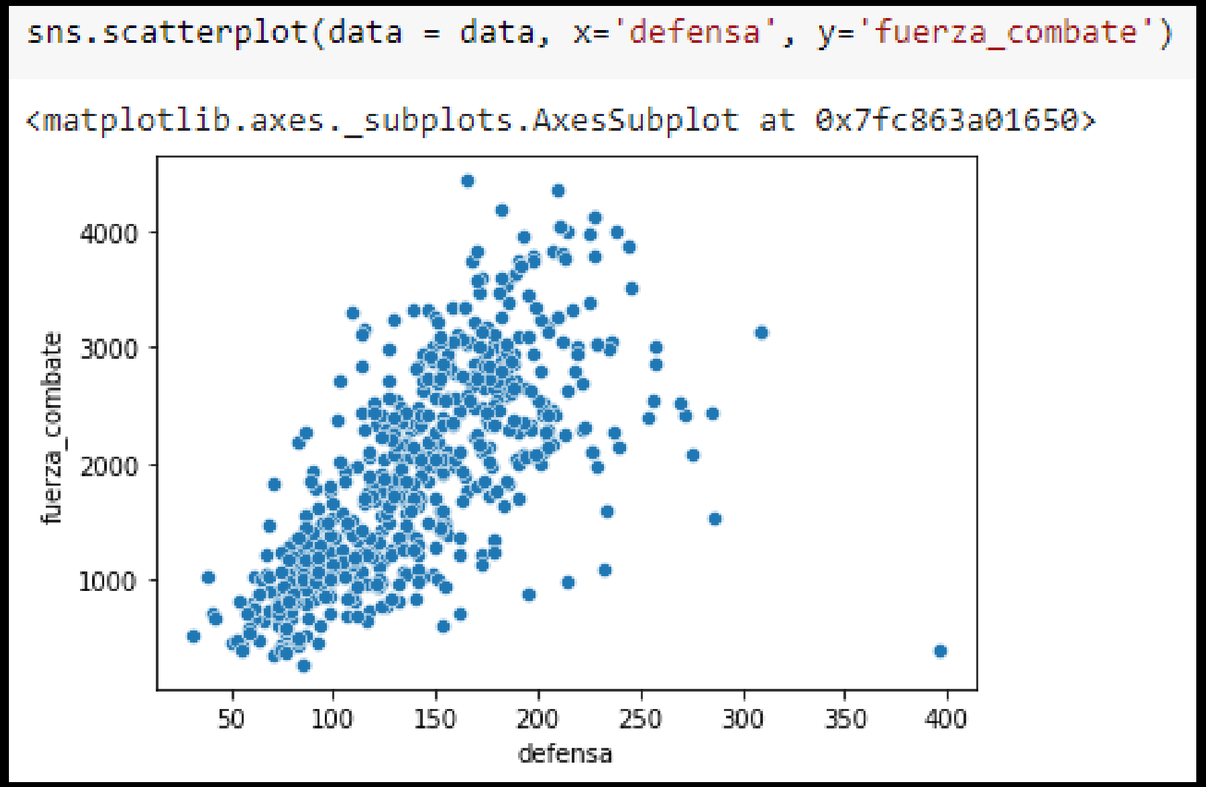
```
None        339
Flying       78
Poison       31
Ground       28
Psychic      24
Steel        19
Fighting     18
Dark         14
Fairy        14
Rock         14
Grass        13
Water         9
Fire          9
Dragon        9
Ghost         8
Ice           8
Electric      6
Bug           3
Name: tipo_secundario, dtype: int64
```

Regresion Lineal Simple

```
data_selec = data[['fuerza_combate', 'defensa']]
```

```
sns.scatterplot(data = data, x='defensa', y='fuerza_combate')
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fc863a01650>
```



Relación entre la columna 'fuerza_combate' y 'defensa'. Mientras más defensa hay, mayor es la fuerza de combate. A excepciones de algunos casos.

```python
from sklearn.model_selection import train_test_split
```

```python
X=data['defensa']
y=data['fuerza_combate']
```

```python
x_train, x_test, y_train, y_test = train_test_split(X,y, random_state = 24)
```

```python
x_train
```

```
316    159.0
125    154.0
618     98.0
626     97.0
42     112.0
       ...
145    181.0
404    156.0
346    100.0
193     66.0
421    105.0
Name: defensa, Length: 483, dtype: float64
```

```python
y = data['fuerza_combate']
X = data.drop(columns= ['nombre', 'tipo_primario', 'tipo_secundario', 'fuerza_combate'])
X_limpia = X

X.head()
```

	resistencia	ataque	defensa	max_salud	tasa_captura	tasa_escape	peso	altura	generacion	legendario_bin
0	128.0	118.0	111.0	113.0	0.20	0.10	6.9	0.7	1	0
1	155.0	151.0	143.0	134.0	0.10	0.07	13.0	1.0	1	0
2	190.0	198.0	189.0	162.0	0.05	0.05	100.0	2.0	1	0
3	118.0	116.0	93.0	105.0	0.20	0.10	8.5	0.6	1	0
4	151.0	158.0	126.0	131.0	0.10	0.07	19.0	1.1	1	0

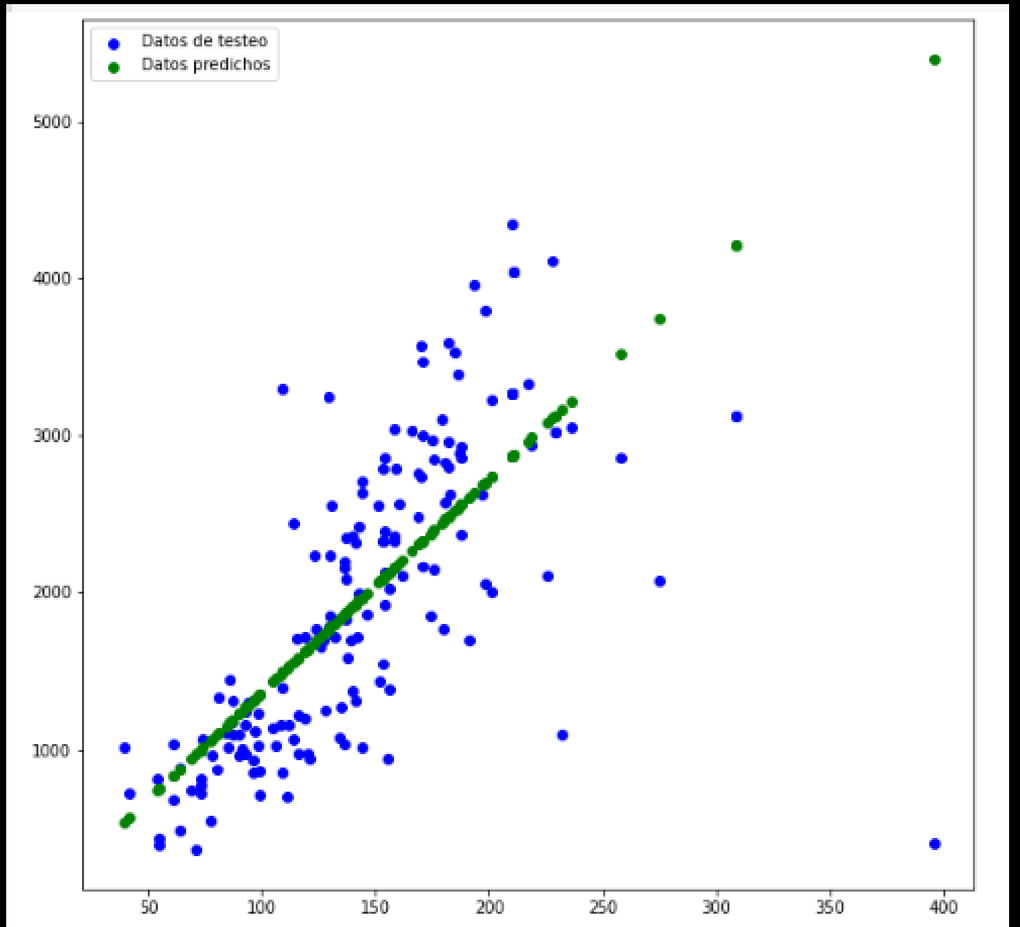
Regresion Lineal Múltiple

```python
from sklearn.metrics import r2_score


def regresion(X,y,columna):
  X = X[columna]
  x_train, x_test, y_train, y_test = train_test_split(X,y)
  x_train_o = x_train.copy()
  x_test_o = x_test.copy()
  if len(x_train.shape) == 1: # Esto es por si se recibe una sola columna en el x para que no haya problemas de formato
    x_train = x_train.to_numpy()
    x_train = np.matrix(x_train.reshape(len(x_train),1))
    x_test = x_test.to_numpy()
    x_test = np.matrix(x_test.reshape(len(x_test),1))
  modelo = LinearRegression(fit_intercept=True)
  modelo.fit(x_train,y_train)
  print (f'Hola, vamos a analizar la columna {columna}')
  print (f'Para la columna {columna} el coeficiente de regresion lineal dio {modelo.coef_[0]}')
  print (f'La ordenada al origen dio: {modelo.intercept_}')
```

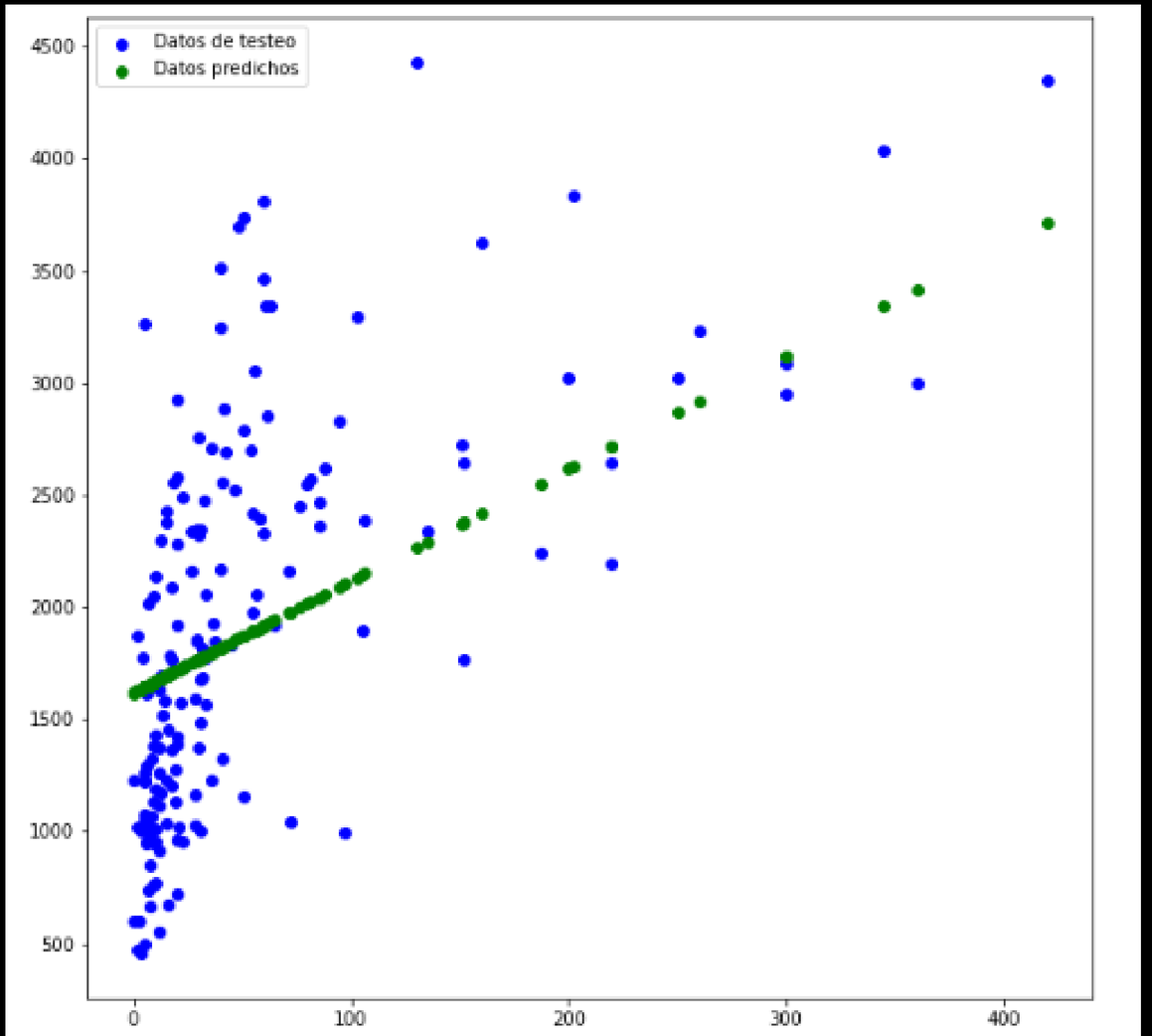
```python
for columna in X.columns:
  regresion(X,y,columna)
```

```
Hola, vamos a analizar la columna defensa
Para la columna defensa el coeficiente de regresion lineal dio 13.60357334148068
La ordenada al origen dio: 7.109620732998565
```




```
/usr/local/lib/python3.7/dist-packages/sklearn/utils/validation.py:598: FutureWar
  FutureWarning,
El R cuadrado para este ajuste dio: 0.438965815809077
```

```
Hola, vamos a analizar la columna peso
Para la columna peso el coeficiente de regresion lineal dio 4.993184372668278
La ordenada al origen dio: 1618.2314138892536
```




```
El R cuadrado para este ajuste dio: 0.3203452568084817
```

```python
x_train, x_test, y_train, y_test = train_test_split(X_limpia,y)
modelo = LinearRegression(fit_intercept=True)
modelo.fit(x_train,y_train)
print (modelo.coef_)
print (X_limpia.columns)
y_pred = modelo.predict(x_test)
r2 = r2_score(y_test,y_pred)
print (f'El R cuadrado para este ajuste dio: {r2}')
```

```
[-5.14552362e+00  1.07539845e+01  5.41696406e+00  1.16187942e+01
 -1.93047441e+00  5.86178856e+02  4.08610052e-01 -8.31583735e+00
 -1.46744051e+00  2.04568133e+02]
Index(['resistencia', 'ataque', 'defensa', 'max_salud', 'tasa_captura',
       'tasa_escape', 'peso', 'altura', 'generacion', 'legendario_bin'],
      dtype='object')
El R cuadrado para este ajuste dio: 0.9658486556986877
```

Gracias

Por Camila Plata