#DataChallenge365

Predicción de ventas de videojuegos

Mayumy CH
Becaria #DataChallenge365 2020-2021

"Nunca te rindas, lo que es hoy es difícil, mañana es una conquista “ ☠

En el 2019, la industria de los videojuegos acumuló un total de ***120.1 mil millones*** *de dólares en ingresos en comparación con los ingresos de taquilla global del cine con 42.5 mil millones de dólares.*

Fuente: [Estadísticas del Mercado de los Juegos 2020/2021 - Ciberninjas]

1.

Comprensión del Negocio y los datos

Obtención de la data

Gregory Smith

Rush Kirubi

Carga de la data

	Name	Platform	Year_of_Release	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Critic_Score	Critic_Count	User_Score	User_Count	Developer	Rating
0	Wii Sports	Wii	2006.0	Sports	Nintendo	41.36	28.96	3.77	8.45	82.53	76.0	51.0	8	322.0	Nintendo	E
1	Super Mario Bros.	NES	1985.0	Platform	Nintendo	29.08	3.58	6.81	0.77	40.24	NaN	NaN	NaN	NaN	NaN	NaN
2	Mario Kart Wii	Wii	2008.0	Racing	Nintendo	15.68	12.76	3.79	3.29	35.52	82.0	73.0	8.3	709.0	Nintendo	E
3	Wii Sports Resort	Wii	2009.0	Sports	Nintendo	15.61	10.93	3.28	2.95	32.77	80.0	73.0	8	192.0	Nintendo	E
4	Pokemon Red/Pokemon Blue	GB	1996.0	Role-Playing	Nintendo	11.27	8.89	10.22	1.00	31.37	NaN	NaN	NaN	NaN	NaN	NaN

16719 registros / 16 columnas

Información

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16719 entries, 0 to 16718
Data columns (total 16 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   Name             16717 non-null  object
 1   Platform         16719 non-null  object
 2   Year             16450 non-null  float64
 3   Genre            16717 non-null  object
 4   Publisher        16665 non-null  object
 5   NA_Sales         16719 non-null  float64
 6   EU_Sales         16719 non-null  float64
 7   JP_Sales         16719 non-null  float64
 8   Other_Sales      16719 non-null  float64
 9   Global_Sales     16719 non-null  float64
 10  Critic_Score     8137 non-null   float64
 11  Critic_Count     8137 non-null   float64
 12  User_Score       10015 non-null  object
 13  User_Count       7590 non-null   float64
 14  Developer        10096 non-null  object
 15  Rating           9950 non-null   object
dtypes: float64(9), object(7)
memory usage: 2.0+ MB
```

Datos Nulos

	Nulos	Cantidad	%_Nulos	Tipo_Dato
User_Count	True	9129	54.60	float64
Critic_Score	True	8582	51.33	float64
Critic_Count	True	8582	51.33	float64
Rating	True	6769	40.49	object
User_Score	True	6704	40.10	object
Developer	True	6623	39.61	object
Year	True	269	1.61	float64
Publisher	True	54	0.32	object
Name	True	2	0.01	object
Genre	True	2	0.01	object

2.

Extracción y preparación de los datos

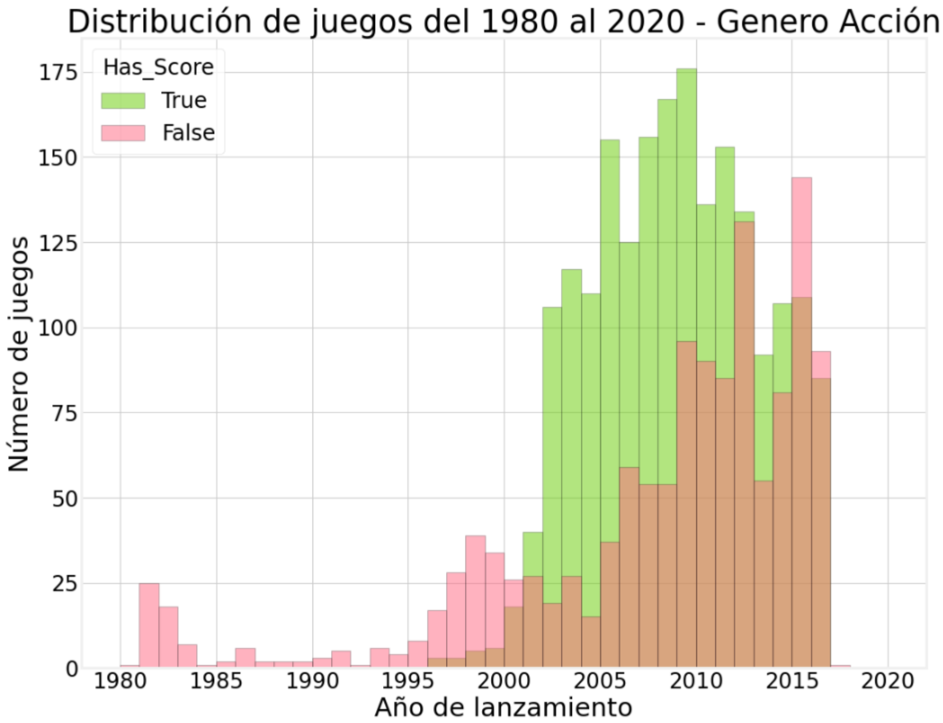
Extracción y preparación de los datos

Crear una variable "Has_Score" – Tener mapeado que registros tienen score ya sea por los criticos o por los usuarios

Tabla número de registros vs Genre vs Has_Score

Genre	Action	Adventure	Fighting	Misc	Platform	Puzzle	Racing	Role-Playing	Shooter	Simulation	Sports	Strategy	All
Sin Score	1327	945	424	1184	370	349	464	741	341	484	1024	354	8007
Con Score	2043	358	425	566	518	231	785	759	982	390	1324	329	8710
Total	3370	1303	849	1750	888	580	1249	1500	1323	874	2348	683	16717

Analizar los juegos con género de Acción



3370

Números de juegos con género Acción registrados del 1980 al 2020.

El 60.62% tienen algún score registrado.

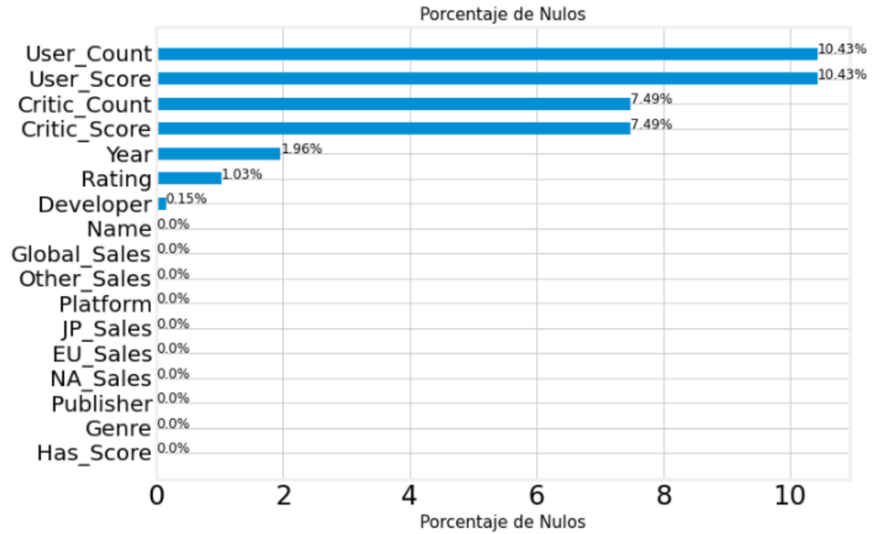
Datos de los juegos de Género Accion registrados

	Name	Platform	Year	Genre	Publisher	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Critic_Score	Critic_Count	User_Score	User_Count	Developer	Rating	Has_Score
16	Grand Theft Auto V	PS3	2013.0	Action	Take-Two Interactive	7.02	9.09	0.98	3.96	8.0	97.0	50.0	8.2	3994.0	Rockstar North	M	True
17	Grand Theft Auto: San Andreas	PS2	2004.0	Action	Take-Two Interactive	9.43	0.40	0.41	10.57	8.0	95.0	80.0	9.0	1588.0	Rockstar North	M	True
23	Grand Theft Auto V	X360	2013.0	Action	Take-Two Interactive	9.66	5.14	0.06	1.41	8.0	97.0	58.0	8.1	3711.0	Rockstar North	M	True
24	Grand Theft Auto: Vice City	PS2	2002.0	Action	Take-Two Interactive	8.41	5.49	0.47	1.78	8.0	95.0	62.0	8.7	730.0	Rockstar North	M	True

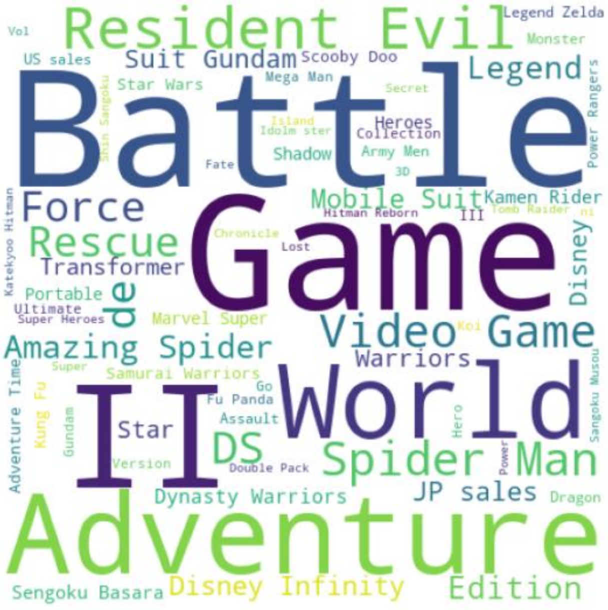
2043 registros / 17 columnas

Datos Nulos

	Nulos	Cantidad	%_Nulos	Tipo_Dato
User_Count	True	213	10.43	float64
User_Score	True	213	10.43	float64
Critic_Count	True	153	7.49	float64
Critic_Score	True	153	7.49	float64
Year	True	40	1.96	float64
Rating	True	21	1.03	object
Developer	True	3	0.15	object



Nombres de los Juegos con Score



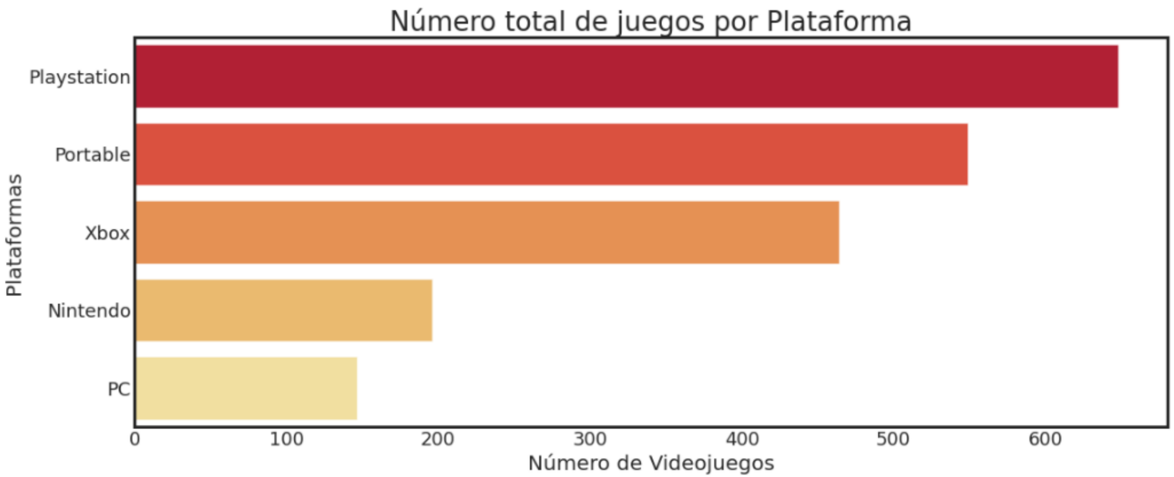
Nombres de los Juegos sin Score

Agrupar la variable “plataformas”

```
dic_platforms = {"Playstation" : ["PS", "PS2", "PS3", "PS4"],
                 "Xbox" : ["XB", "X360", "XOne"],
                 "PC" : ["PC"],
                 "Nintendo" : ["Wii", "WiiU"],
                 "Portable" : ["GB", "GBA", "GC", "DS", "3DS", "PSP", "PSV"]}
```

59.76%

De los juegos calificados fueron de playstation y portable

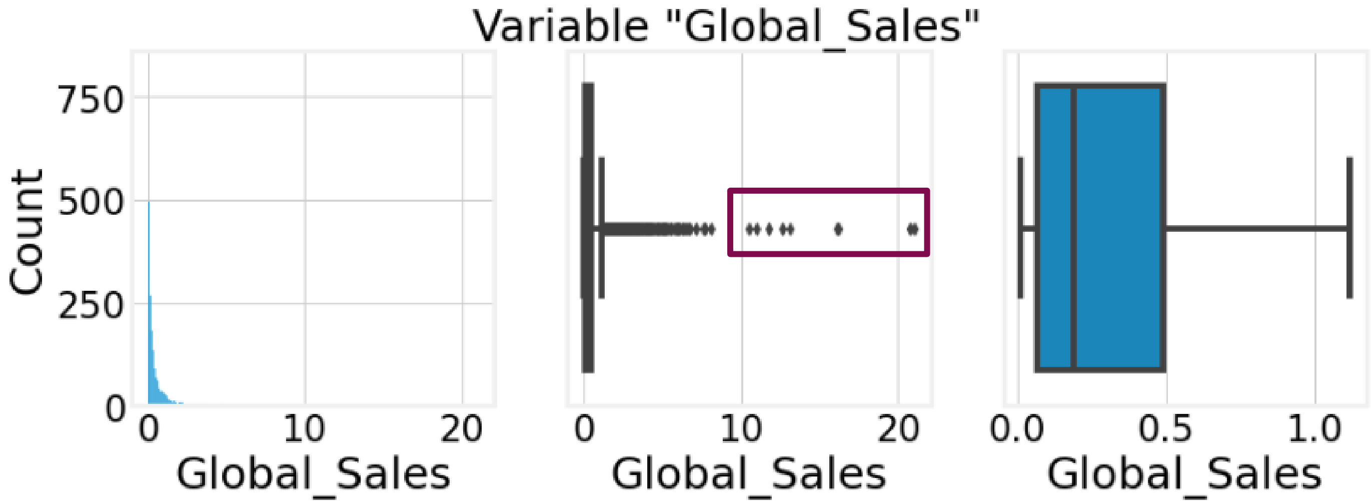


Descripción de variables Numericas

	Year	NA_Sales	EU_Sales	JP_Sales	Other_Sales	Global_Sales	Critic_Score	Critic_Count	User_Score	User_Count
count	2003.000000	2043.000000	2043.000000	2043.000000	2043.000000	2043.000000	1890.000000	1890.000000	1830.000000	1830.000000
mean	2008.428857	0.325174	0.209104	0.039310	0.078483	0.652428	66.629101	27.780952	7.054044	188.889617
std	4.255255	0.648005	0.486582	0.132171	0.298021	1.346726	14.206877	20.301921	1.425394	542.538499
min	1996.000000	0.000000	0.000000	0.000000	0.000000	0.010000	19.000000	4.000000	0.300000	4.000000
25%	2005.000000	0.050000	0.020000	0.000000	0.010000	0.100000	57.000000	12.000000	6.300000	11.000000
50%	2008.000000	0.130000	0.060000	0.000000	0.020000	0.260000	68.000000	22.000000	7.400000	28.000000
75%	2012.000000	0.330000	0.200000	0.010000	0.070000	0.635000	77.000000	39.000000	8.100000	103.500000
max	2016.000000	9.660000	9.090000	2.020000	10.570000	21.040000	98.000000	106.000000	9.500000	8003.000000

Variable Target tiene outlier

Variable Target

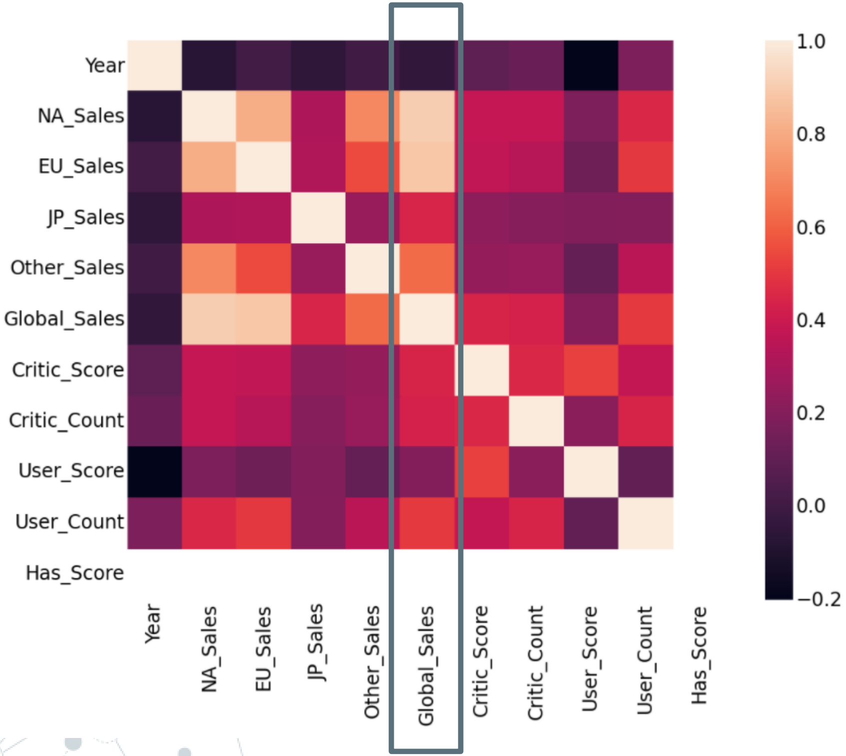


Limpieza de datos Nulos

```
# Features Numericos
df_actionScore = df_actionScore[df_actionScore["Year"].notnull()]
df_actionScore['Critic_Score'].fillna(df_actionScore['Critic_Score'].median(), inplace=True)
df_actionScore['Critic_Count'].fillna(df_actionScore['Critic_Count'].median(), inplace=True)
df_actionScore['User_Score'].fillna(df_actionScore['User_Score'].median(), inplace=True)
df_actionScore['User_Count'].fillna(df_actionScore['User_Count'].median(), inplace=True)

# Features Categoricos
df_actionScore['Rating'].fillna(df_actionScore['Rating'].mode()[0], inplace=True)
df_actionScore['Developer'].fillna(df_actionScore['Developer'].mode()[0], inplace=True)
```

Sin datos nulos

Analisis de Correlación




```
Year           -0.045150
User_Score      0.196443
Critic_Count    0.429609
Critic_Score    0.436868
JP_Sales        0.441947
User_Count      0.503288
Other_Sales     0.627949
EU_Sales        0.883294
NA_Sales        0.905475
Global_Sales    1.000000
Has_Score            NaN
Name: Global_Sales, dtype: float64
```

"NA_Sales" y **"EU_Sales "** son las **variables** que estan **mas correlacionadas** con el target.

Pero debido a que entre ellas estan correlacionadas solo me quedare con una de ellas.

3.

Modelamiento y Evaluación

Preparando los features a utilizar

```
# VARIABLE NUMERICA
df_num = df_actionScore.select_dtypes("number").drop(columns=["Year","Other_Sales","JP_Sales", "EU_Sales"])

# VARIABLE CATEGORICA
df_cat = df_actionScore[["GPlatforms", "Rating"]]
df_cat = pd.get_dummies(df_cat, drop_first=True)
```

```
dataModel = pd.concat([df_num, df_cat], axis = 1)
X = dataModel.drop(columns="Global_Sales")
y = dataModel["Global_Sales"]
```

Datos de Entrenamiento y Test

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3,
                                                    random_state=2020)  #Semilla para replicar el modelo
```

```
#Revisamos los tamaños de las pruebas de train y test
print("Tamaño del conjunto de datos Inicial:", dataModel.shape)
print("Tamaño del conjunto de características del entrenamiento:",X_train.shape)
print("Tamaño del conjunto de características de prueba:",X_test.shape)
print("Tamaño de la variable objetivo del entrenamiento:",y_train.shape)
print("Tamaño de la variable objetivo de prueba:",y_test.shape)
```

```
Tamaño del conjunto de datos Inicial: (2003, 14)
Tamaño del conjunto de características del entrenamiento: (1402, 13)
Tamaño del conjunto de características de prueba: (601, 13)
Tamaño de la variable objetivo del entrenamiento: (1402,)
Tamaño de la variable objetivo de prueba: (601,)
```

Modelo: Entrenar y Evaluar

```python
def mae(y_true, y_pred):
    return np.average(abs(y_true - y_pred))
```

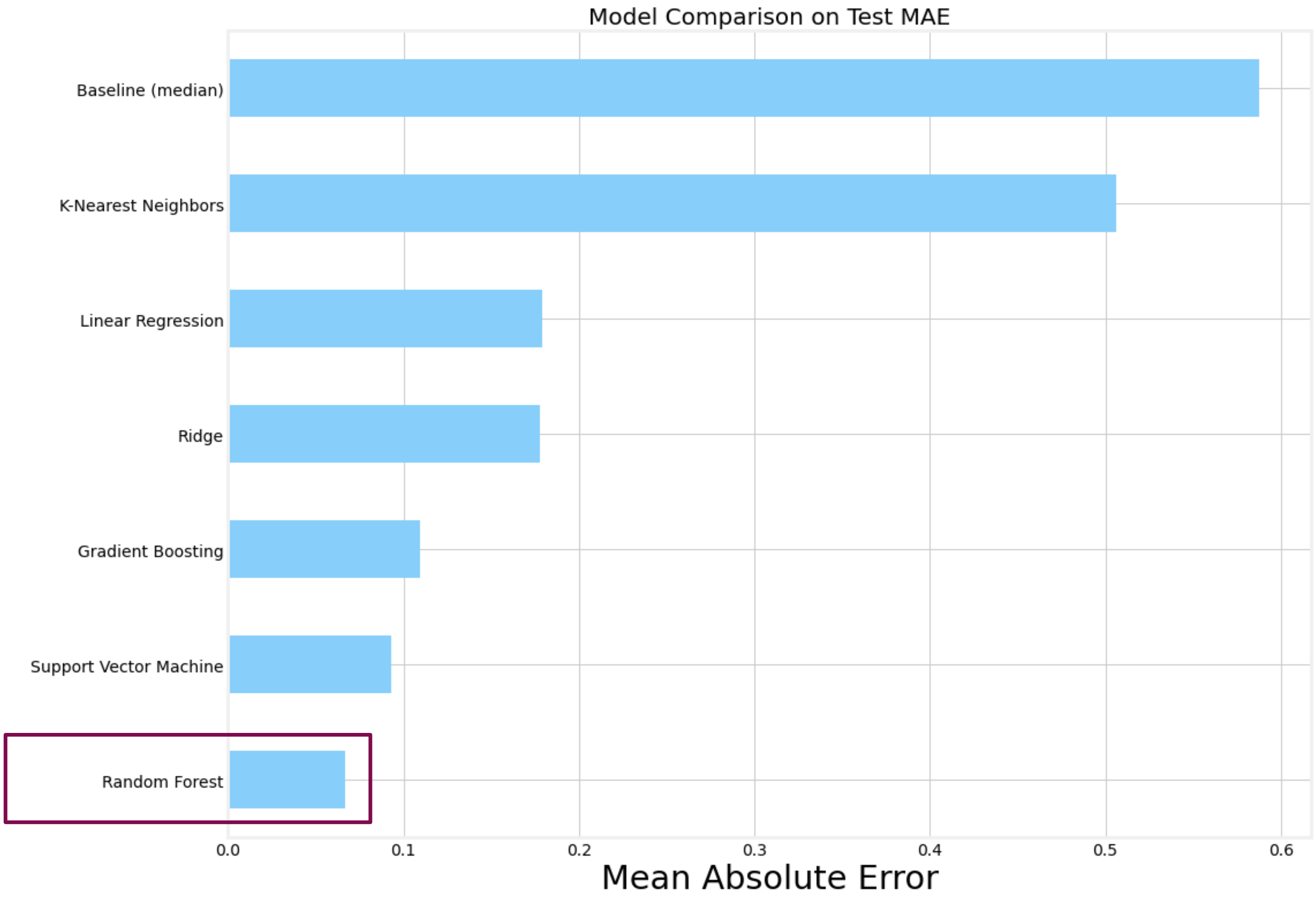
```python
def fit_and_evaluate(model):

    # Entrenar el modelo
    model.fit(X_train, y_train)

    # Predecir y evaluar
    model_pred = model.predict(X_train)
    model_mae = mae(y_train, model_pred)

    # Devuelve la métrica del Modelo
    return model_mae
```

```python
# MODELO DUMMY
baseline_guess = np.median(X_train)
basic_baseline_mae = mae(y_train, baseline_guess)
```

```python
# Regresion Lineal
lr = LinearRegression()
lr_mae = fit_and_evaluate(lr)
```

```python
# Random Forest
random_forest = RandomForestRegressor(random_state=60)
random_forest_mae = fit_and_evaluate(random_forest)
```

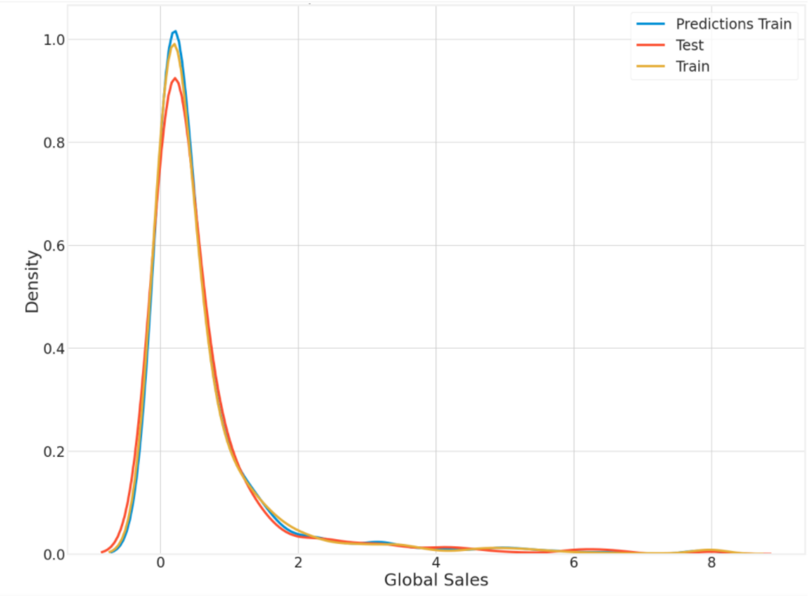
Comparando modelos con la métrica MAE



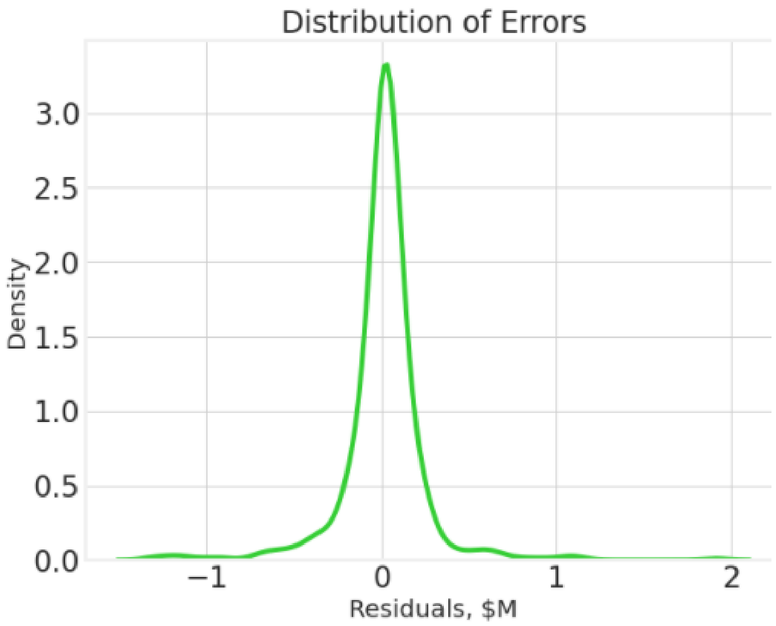
```
# Escogeremos el mejor modelo
# ----------------------------
from sklearn.linear_model import LinearRegression
from sklearn import metrics

random_forest = RandomForestRegressor(random_state=60)
random_forest.fit(X_train, y_train)

train_pred=random_forest.predict(X_train)
test_pred=random_forest.predict(X_test)

random_forest_mae = fit_and_evaluate(random_forest)
random_forest_mae
```

```
0.06036148359486446
```

Test, Train and Predictions



Video Games - Predicting Global Sales



¡ FELICIDADES A TODAS LAS BECAD@S !

Agradecimientos especiales:

DATA SCIENCIE FEM

TEAM
PaRsel

TEAM MARTES Y JUEVES
REFUERZO - FACTORED

Thanks!

Alguna pregunta?

- Redes: MayumyCH | Linktree
- Repositorio: MayumyCH/video_game_sales_with_ratings