Análisis de Xbox Game Pass: Rentabilidad y Uso (2022)

Introducción

Este proyecto tiene como objetivo evaluar el uso y la rentabilidad del servicio Xbox Game Pass durante el año 2022. Se analiza si los usuarios dedican tiempo a jugar títulos populares y bien valorados por la crítica. Los datos utilizados provienen de datasets disponibles en **Kaggle**, que incluyen información sobre las horas de juego y las puntuaciones en **Metacritic**, una plataforma que integra valoraciones de jugadores y críticos.

Xbox Game Pass es un servicio de suscripción de videojuegos que permite a los usuarios acceder a una biblioteca de juegos de alta calidad para consolas Xbox, PC y la nube por una tarifa mensual dependiendo de cada plan. A través de Game Pass, los usuarios pueden jugar a títulos nuevos y populares sin necesidad de comprarlos individualmente. Además, existen otros servicios similares en otras plataformas de los cuales se estiman los siguientes números:

- Xbox Game Pass: 29 millones de suscriptores
- Playstation Plus: 47 millones de suscriptores
- EA Play: 13 millones de suscriptoresv
- Nintendo Switch Online: 36 millones de suscriptores

Con estos números, se puede interpretar que Xbox Game Pass se queda muy por detrás con su competencia direccta. Por eso mismo, se realiza este análisis para responder unas preguntas clave:

- ¿Están aprovechando suficientes jugadores las ventajas del servicio?
- ¿Qué factores afectan realmente el número de jugadores? ¿Es la popularidad de los juegos el único determinante o existen otros elementos importantes?
- ¿Debería Xbox Game Pass centrarse en cierto tipo de juegos para atraer más suscriptores?

Con esto, se presenta el análisis realiizado.

Imports

Primero, importamos las bibliotecas necesarias para el análisis y las funciones auxiliares desde la carpeta `utils`.

In [1]:
```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import seaborn as sns
import re
import sys
sys.path.append('src/utils')

import constants as cons
import funciones as funtils
```

Carga, tratamiento visualización de datos

Se cargan los datasets que se van a usar, se eliminan columnas innecesarias y se hace tratamiento de los valores en los registros de tal forma que los datos sean más sencillos de usar. Posteriormente, se unen los datasets para hacer el análisis

In [38]:
```
df_gamepass = pd.read_csv(cons.DF_GAMEPASS_SRC_MAIN)
df_gamepass = df_gamepass.drop(columns=[cons.RATIO, cons.TRUE_ACHIVEMENTS, cons.GAM
df_gamepass.head()
```

Out[38]:

	GAME	GAMERS	COMP %	TIME	RATING	ADDED
0	Mass Effect Legendary Edition	84,143	4.1	100-120 hours	4.8	06 Jan 22
1	The Elder Scrolls V: Skyrim Special Edition	213,257	8.0	80-100 hours	4.7	15 Dec 20
2	Mass Effect 2	221,178	9.6	50-60 hours	4.7	09 Nov 20
3	Stardew Valley	51,530	1.0	150-200 hours	4.7	02 Dec 21
4	It Takes Two	71,981	15.6	12-15 hours	4.7	03 Nov 21

Visualizamos la información general del DataFrame para entender mejor la estructura de los datos.

In [39]:
```
df_gamepass.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 455 entries, 0 to 454
Data columns (total 6 columns):
 #   Column  Non-Null Count  Dtype  
---  ------  --------------  -----  
 0   GAME    455 non-null    object 
 1   GAMERS  455 non-null    object 
 2   COMP %  455 non-null    float64
 3   TIME    421 non-null    object 
 4   RATING  452 non-null    float64
 5   ADDED   454 non-null    object 
dtypes: float64(2), object(4)
memory usage: 21.5+ KB
```

Existen nulos en algunas columnas y la columna GAMERS no es numerica. Además, es conveniente modificar la columna TIME de tal forma que sea el tiempo medio y así el análisis sea más sencillo.

Antes de hacer cambios, se revisan los nulos que existen en TIME

In [40]:
```
df_gamepass[df_gamepass[cons.TIME].isnull()].sort_values(by= cons.GAMERS, ascending
```

Out[40]:

	GAME	GAMERS	COMP %	TIME	RATING	ADDED
423	Bassmaster Fishing 2022	9,425	0.0	NaN	2.4	09 Oct 21
242	Fight Night Champion	84,000	0.0	NaN	3.6	09 Nov 20
345	Astria Ascending	8,446	0.4	NaN	3.3	18 Sep 21
431	Next Space Rebels	8,091	0.3	NaN	3.2	17 Nov 21
448	The Catch: Carp Coarse Fishing	8,084	0.0	NaN	2.7	18 May 21
436	Chinatown Detective Agency	738	0.0	NaN	2.6	07 Apr 22
444	Phoenix Point	7,388	0.0	NaN	3.6	01 Oct 21
91	Elite: Dangerous	66,416	0.0	NaN	4.1	24 Feb 21
437	RESEARCH and DESTROY	6,162	0.1	NaN	3.1	26 Apr 22
427	Unsouled	4,302	0.0	NaN	3.1	28 Apr 22
433	The Dungeon of Naheulbeuk: The Amulet of Chaos...	4,273	0.1	NaN	4.1	17 Mar 22
451	The Evil Within (JP)	388	12.1	NaN	3.8	16 Aug 21
429	Undungeon	386	0.0	NaN	2.5	19 Nov 21
442	Galactic Civilizations III (Windows)	348	0.0	NaN	3.1	16 Mar 22
438	Loot River	334	0.0	NaN	3.2	Yesterday
420	UNSIGHTED	3,684	0.5	NaN	3.6	10 Sep 21
336	NBA LIVE 19	29,473	0.1	NaN	3.3	09 Nov 20
421	Lemnis Gate	2,739	0.4	NaN	3.3	10 Sep 21
294	NBA 2K22	18,528	0.0	NaN	3.5	28 Apr 22
204	Train Sim World 2	17,494	0.0	NaN	3.8	19 Aug 21
428	Fae Tactics	166	0.0	NaN	3.0	18 Nov 21
414	Library Of Ruina	14,668	0.0	NaN	2.2	NaN
365	Edge of Eternity	14,114	0.0	NaN	3.2	31 Jan 22
393	Power Rangers: Battle for the Grid	13,851	0.4	NaN	2.9	26 Mar 20
384	Nuclear Throne	12,261	0.0	NaN	3.0	09 Sep 21

	GAME	GAMERS	COMP %	TIME	RATING	ADDED
368	Crown Trick	12,253	0.0	NaN	3.2	03 Sep 21
46	Stellaris: Console Edition	11,695	0.0	NaN	4.3	16 Dec 19
441	Crusader Kings III	11,456	0.0	NaN	3.8	14 Mar 22
419	NBA 2K22 (Xbox One)	11,311	0.0	NaN	3.4	28 Apr 22
449	Atomicrops	1,853	1.9	NaN	4.5	22 Jul 21
443	Dragon Ball FighterZ (Windows)	1,436	0.4	NaN	3.9	16 Mar 22
452	Shadowrun Returns	0	0.0	NaN	NaN	20 Apr 22
453	Shadowrun: Hong Kong - Extended Edition	0	0.0	NaN	NaN	20 Apr 22
454	Shadowrun: Dragonfall - Director's Cut	0	0.0	NaN	NaN	20 Apr 22

Al parecer algunos son juegos que en su momento fueron recien agregados, por lo que no hay suficientes datos y la mayoría no han sido commpletados o siquiera jugados. De momento, se eliminan los que fueron recien agregados y se buscará si los otros juegos son de algun tipo en particular y se continua con las modificaciones.

Convertimos la columna `GAMERS` a un formato numérico y eliminamos los juegos con 0 jugadores. Posteriormente, calculamos el tiempo medio para completar cada título.

In [41]:
```
df_gamepass[cons.GAMERS] = pd.to_numeric(df_gamepass[cons.GAMERS].str.replace(',',
df_gamepass = df_gamepass[df_gamepass[cons.GAMERS] != 0]
df_gamepass[cons.TIME] = df_gamepass[cons.TIME].apply(funtils.calculate_mean_time)
df_gamepass.head()
```

Out[41]:

	GAME	GAMERS	COMP %	TIME	RATING	ADDED
0	Mass Effect Legendary Edition	84143	4.1	110.0	4.8	06 Jan 22
1	The Elder Scrolls V: Skyrim Special Edition	213257	8.0	90.0	4.7	15 Dec 20
2	Mass Effect 2	221178	9.6	55.0	4.7	09 Nov 20
3	Stardew Valley	51530	1.0	175.0	4.7	02 Dec 21
4	It Takes Two	71981	15.6	13.5	4.7	03 Nov 21

In [42]:
```
df_gamepass.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 452 entries, 0 to 451
Data columns (total 6 columns):
 #   Column  Non-Null Count  Dtype
---  ------  --------------  -----
 0   GAME    452 non-null    object
 1   GAMERS  452 non-null    int64
 2   COMP %  452 non-null    float64
 3   TIME    414 non-null    float64
 4   RATING  452 non-null    float64
 5   ADDED   451 non-null    object
dtypes: float64(3), int64(1), object(2)
memory usage: 24.7+ KB
```

Ahora se carga el otro dataset de Metacritic para hacer el cruce. Posterior a esto, se hará el tratamiento de nulos. Pero antes, se eliminan las columnas innecesarias

In [43]:
```
df_metacritic = pd.read_csv(cons.DF_METACRITIC_SRC_MAIN)
df_metacritic = df_metacritic.drop(columns=[cons.ID, cons.PLATFORM, cons.SORT_NO, c
df_metacritic.head()
```

Out[43]:

	metascore	release_date	title	user_score
0	91	August 18, 2020	Microsoft Flight Simulator	7.1
1	91	December 8, 2022	Chained Echoes	8.7
2	91	November 7, 2005	Guitar Hero	8.5
3	91	November 13, 2008	World of Warcraft: Wrath of the Lich King	7.7
4	91	October 26, 2010	Rock Band 3	6.8

Visualizamos la información general del DataFrame de Metacritic.

In [44]:
```
df_metacritic.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20022 entries, 0 to 20021
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   metascore     20022 non-null  int64
 1   release_date  20022 non-null  object
 2   title         20022 non-null  object
 3   user_score    20022 non-null  object
dtypes: int64(1), object(3)
memory usage: 625.8+ KB
```

Se hace el cruce por título, pero se normalizan a minúsculas para asegurar el cruce. Normalizamos los títulos a minúsculas para hacer el cruce entre los datasets y realizamos el merge.

In [45]:
```
df_metacritic[cons.TITLE] = df_metacritic[cons.TITLE].str.lower()
df_gamepass[cons.GAME] = df_gamepass[cons.GAME].str.lower()
```

```
df_merged = pd.merge(df_metacritic, df_gamepass, left_on='title', right_on=cons.GAM
df_merged.head()
```

Out[45]:

	metascore	release_date	title	user_score	GAME	GAMERS	COMP %	TIME	RATIN
0	91	August 18, 2020	microsoft flight simulator	7.1	microsoft flight simulator	91677	0.0	NaN	4
1	91	September 27, 2019	dragon quest xi s: echoes of an elusive age - ...	8.6	dragon quest xi s: echoes of an elusive age - ...	52220	4.6	110.0	4
2	91	November 19, 2009	peggle	7.4	peggle	165096	0.6	45.0	3
3	91	November 9, 2021	forza horizon 5	6.7	forza horizon 5	301931	1.6	90.0	4
4	91	August 7, 2018	dead cells	8.1	dead cells	86081	0.1	27.5	4

In [46]:
```
df_merged.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 908 entries, 0 to 911
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   metascore     908 non-null    int64  
 1   release_date  908 non-null    object 
 2   title         908 non-null    object 
 3   user_score    908 non-null    object 
 4   GAME          908 non-null    object 
 5   GAMERS        908 non-null    int64  
 6   COMP %        908 non-null    float64
 7   TIME          862 non-null    float64
 8   RATING        908 non-null    float64
 9   ADDED         908 non-null    object 
dtypes: float64(3), int64(2), object(5)
memory usage: 78.0+ KB
```

Se revisan de nuevo los títulos con el campo TIME como nulo

In [47]:
```
df_merged[df_merged[cons.TIME].isnull()].sort_values(by= cons.GAMERS, ascending=Fal
```

Out[47]:

	metascore	release_date	title	user_score	GAME	GAMERS	COMP %	TIME
10	91	September 20, 2011	gears of war 3	8.1	gears of war 3	388495	0.7	NaN
819	67	March 20, 2018	sea of thieves	4.8	sea of thieves	344597	0.0	NaN
787	69	March 20, 2018	sea of thieves	5.3	sea of thieves	344597	0.0	NaN
322	82	September 10, 2019	gears 5	6.9	gears 5	302711	0.1	NaN
220	84	September 6, 2019	gears 5	8.2	gears 5	302711	0.1	NaN
740	71	April 4, 2014	the elder scrolls online	5.7	the elder scrolls online	204295	0.0	NaN
44	90	July 27, 2021	microsoft flight simulator	7.9	microsoft flight simulator	91677	0.0	NaN
0	91	August 18, 2020	microsoft flight simulator	7.1	microsoft flight simulator	91677	0.0	NaN
221	84	March 1, 2011	fight night champion	7.3	fight night champion	84000	0.0	NaN
159	86	March 1, 2011	fight night champion	7.4	fight night champion	84000	0.0	NaN
404	80	December 16, 2014	elite: dangerous	6.7	elite: dangerous	66416	0.0	NaN
534	77	June 27, 2017	elite: dangerous	6.7	elite: dangerous	66416	0.0	NaN
428	80	October 6, 2015	elite: dangerous	6.9	elite: dangerous	66416	0.0	NaN
253	83	May 14, 2021	subnautica: below zero	7.8	subnautica: below zero	41620	23.0	NaN
593	76	May 14, 2021	subnautica: below zero	4.5	subnautica: below zero	41620	23.0	NaN
301	82	May 14, 2021	subnautica: below zero	6.4	subnautica: below zero	41620	23.0	NaN
478	79	May 14, 2021	subnautica: below zero	5.8	subnautica: below zero	41620	23.0	NaN
634	75	September 7, 2018	nba live 19	6.3	nba live 19	29473	0.1	NaN

	metascore	release_date	title	user_score	GAME	GAMERS	COMP %	TIME
672	73	September 7, 2018	nba live 19	6.4	nba live 19	29473	0.1	NaN
591	76	September 10, 2021	nba 2k22	4.1	nba 2k22	18528	0.0	NaN
587	76	September 10, 2021	nba 2k22	3.6	nba 2k22	18528	0.0	NaN
818	67	June 8, 2021	edge of eternity	6.8	edge of eternity	14114	0.0	NaN
887	60	February 10, 2022	edge of eternity	6.3	edge of eternity	14114	0.0	NaN
878	62	March 26, 2019	power rangers: battle for the grid	6.3	power rangers: battle for the grid	13851	0.4	NaN
31	89	December 5, 2015	nuclear throne	7.8	nuclear throne	12261	0.0	NaN
284	82	December 5, 2015	nuclear throne	6.3	nuclear throne	12261	0.0	NaN
282	83	October 16, 2020	crown trick	7.7	crown trick	12253	0.0	NaN
399	80	October 16, 2020	crown trick	7.7	crown trick	12253	0.0	NaN
333	81	February 26, 2019	stellaris: console edition	7.6	stellaris: console edition	11695	0.0	NaN
541	77	February 26, 2019	stellaris: console edition	6.8	stellaris: console edition	11695	0.0	NaN
176	85	March 29, 2022	crusader kings iii	6.7	crusader kings iii	11456	0.0	NaN
258	83	March 29, 2022	crusader kings iii	6.2	crusader kings iii	11456	0.0	NaN
7	91	September 1, 2020	crusader kings iii	8.4	crusader kings iii	11456	0.0	NaN
873	62	September 30, 2021	astria ascending	7.3	astria ascending	8446	0.4	NaN
806	68	September 30, 2021	astria ascending	5.4	astria ascending	8446	0.4	NaN

	metascore	release_date	title	user_score	GAME	GAMERS	COMP %	TIME
871	63	September 30, 2021	astria ascending	6.4	astria ascending	8446	0.4	NaN
649	74	December 3, 2019	phoenix point	5.8	phoenix point	7388	0.0	NaN
247	84	September 30, 2021	unsighted	7.8	unsighted	3684	0.5	NaN
629	75	September 28, 2021	lemnis gate	7.2	lemnis gate	2739	0.4	NaN
485	78	September 28, 2021	lemnis gate	4.5	lemnis gate	2739	0.4	NaN
509	78	September 28, 2021	lemnis gate	6	lemnis gate	2739	0.4	NaN
531	78	May 28, 2020	atomicrops	7.4	atomicrops	1853	1.9	NaN
831	66	April 7, 2022	chinatown detective agency	4.5	chinatown detective agency	738	0.0	NaN
723	72	May 3, 2022	loot river	4.6	loot river	334	0.0	NaN
834	66	May 3, 2022	loot river	5.2	loot river	334	0.0	NaN
550	77	July 31, 2020	fae tactics	6.4	fae tactics	166	0.0	NaN

Antes de quitar los nulos, se observa que aún hay titulos repetidos. Por lo tanto, se van a quitar duplicados de tal forma que se conserven titulos con las calificaciones y porcentaje de completado más altos

In [48]:
```
df_merged = df_merged.sort_values(by=[cons.USER_SCORE, cons.COMP, cons.METASCORE],
df_merged.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 357 entries, 483 to 777
Data columns (total 10 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   metascore     357 non-null    int64  
 1   release_date  357 non-null    object 
 2   title         357 non-null    object 
 3   user_score    357 non-null    object 
 4   GAME          357 non-null    object 
 5   GAMERS        357 non-null    int64  
 6   COMP %        357 non-null    float64
 7   TIME          333 non-null    float64
 8   RATING        357 non-null    float64
 9   ADDED         357 non-null    object 
dtypes: float64(3), int64(2), object(5)
memory usage: 30.7+ KB
```

Ahora, se revisan de nuevo los nulos.

In [49]:
```
df_merged[df_merged[cons.TIME].isnull()].sort_values(by= cons.GAMERS, ascending=Fal
```

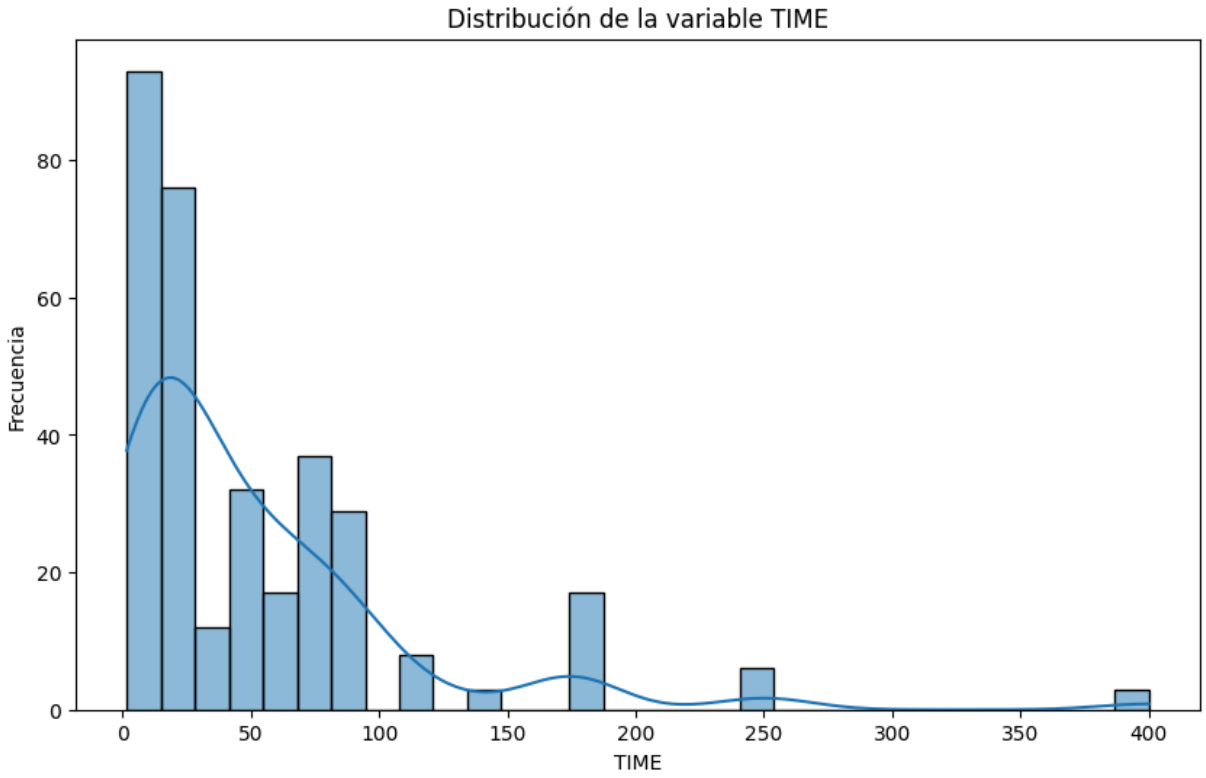
Out[49]:

	metascore	release_date	title	user_score	GAME	GAMERS	COMP %	TIME
10	91	September 20, 2011	gears of war 3	8.1	gears of war 3	388495	0.7	NaN
787	69	March 20, 2018	sea of thieves	5.3	sea of thieves	344597	0.0	NaN
220	84	September 6, 2019	gears 5	8.2	gears 5	302711	0.1	NaN
740	71	April 4, 2014	the elder scrolls online	5.7	the elder scrolls online	204295	0.0	NaN
44	90	July 27, 2021	microsoft flight simulator	7.9	microsoft flight simulator	91677	0.0	NaN
159	86	March 1, 2011	fight night champion	7.4	fight night champion	84000	0.0	NaN
428	80	October 6, 2015	elite: dangerous	6.9	elite: dangerous	66416	0.0	NaN
253	83	May 14, 2021	subnautica: below zero	7.8	subnautica: below zero	41620	23.0	NaN
672	73	September 7, 2018	nba live 19	6.4	nba live 19	29473	0.1	NaN
591	76	September 10, 2021	nba 2k22	4.1	nba 2k22	18528	0.0	NaN
818	67	June 8, 2021	edge of eternity	6.8	edge of eternity	14114	0.0	NaN
878	62	March 26, 2019	power rangers: battle for the grid	6.3	power rangers: battle for the grid	13851	0.4	NaN
31	89	December 5, 2015	nuclear throne	7.8	nuclear throne	12261	0.0	NaN
282	83	October 16, 2020	crown trick	7.7	crown trick	12253	0.0	NaN
333	81	February 26, 2019	stellaris: console edition	7.6	stellaris: console edition	11695	0.0	NaN
7	91	September 1, 2020	crusader kings iii	8.4	crusader kings iii	11456	0.0	NaN
873	62	September 30, 2021	astria ascending	7.3	astria ascending	8446	0.4	NaN

	metascore	release_date	title	user_score	GAME	GAMERS	COMP %	TIME
649	74	December 3, 2019	phoenix point	5.8	phoenix point	7388	0.0	NaN
247	84	September 30, 2021	unsighted	7.8	unsighted	3684	0.5	NaN
629	75	September 28, 2021	lemnis gate	7.2	lemnis gate	2739	0.4	NaN
531	78	May 28, 2020	atomicrops	7.4	atomicrops	1853	1.9	NaN
831	66	April 7, 2022	chinatown detective agency	4.5	chinatown detective agency	738	0.0	NaN
834	66	May 3, 2022	loot river	5.2	loot river	334	0.0	NaN
550	77	July 31, 2020	fae tactics	6.4	fae tactics	166	0.0	NaN

Para un mejor análisis y saber como se van a tratar los nulos, revisamos la distribución de TIME y los estadísticos de las variables numéricas.

In [50]:
```
plt.figure(figsize=(10, 6))
sns.histplot(df_merged[cons.TIME].dropna(), bins=30, kde=True)
plt.title('Distribución de la variable TIME')
plt.xlabel('TIME')
plt.ylabel('Frecuencia')
plt.show()

df_merged.describe()
```



Out[50]:

	metascore	GAMERS	COMP %	TIME	RATING
count	357.000000	357.000000	357.000000	333.000000	357.000000
mean	78.364146	92863.577031	5.957143	51.876877	3.786275
std	8.706942	93055.655763	10.100586	60.668476	0.454326
min	43.000000	166.000000	0.000000	1.500000	2.200000
25%	73.000000	23287.000000	0.600000	11.000000	3.500000
50%	79.000000	61345.000000	1.900000	27.500000	3.800000
75%	84.000000	128274.000000	6.100000	70.000000	4.100000
max	96.000000	455839.000000	62.900000	400.000000	4.800000

Revisando los juegos que siguen con valor nulo en tiempo, se puede notar que la mayoría son juegos online, por lo que no tienen un tiempo de juego definido. Sin embargo, aun existen algunos juegos offline que no cuentan con horas de juego. Por lo tanto, se propone lo siguiente:

- Se tomará un umbral de número de jugadores para descartar algunos titulos, en este caso se eliminaran los juegos con menos de 1000 jugadores
- Al revisar la distribución de valores de la variable `TIME` y viendo sus valores estadísticos, se decide que los valores nulos restantes serán reemplazados por la mediana. Esto porque lo usual es que los videojuegos duren alrededor de 30 horas actualmente.

Además, se eliminan las columnas added y release_date ya que en este punto no serán de mucha utilidad.

In [51]:
```
df_merged = df_merged[df_merged[cons.GAMERS] >= 1000]
median_time = df_merged[cons.TIME].median()
df_merged[cons.TIME].fillna(median_time, inplace=True)
df_merged = df_merged.drop(columns=[cons.ADDED, cons.RELEASE_DATE, cons.GAME])
df_merged.head()
```

```
C:\Users\PC ELITE\AppData\Local\Temp\ipykernel_30288\3247063364.py:3: FutureWarning:
A value is trying to be set on a copy of a DataFrame or Series through chained assig
nment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because
the intermediate object on which we are setting values always behaves as a copy.

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method
({col: value}, inplace=True)' or df[col] = df[col].method(value) instead, to perform
the operation inplace on the original object.


  df_merged[cons.TIME].fillna(median_time, inplace=True)
```

Out[51]:

	metascore	title	user_score	GAMERS	COMP %	TIME	RATING
483	78	olija	tbd	4380	14.2	4.5	3.3
802	68	gang beasts	tbd	135527	9.3	1.5	2.9
894	56	recompile	tbd	7808	6.1	7.0	3.2
852	65	moonglow bay	tbd	4060	2.9	45.0	3.5
699	72	lawn mowing simulator	tbd	33974	0.1	70.0	3.3

In [52]:
```
df_merged.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 354 entries, 483 to 777
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   metascore   354 non-null    int64
 1   title       354 non-null    object
 2   user_score  354 non-null    object
 3   GAMERS      354 non-null    int64
 4   COMP %      354 non-null    float64
 5   TIME        354 non-null    float64
 6   RATING      354 non-null    float64
dtypes: float64(3), int64(2), object(2)
memory usage: 22.1+ KB
```

Ahora surge otro problema: `user_score` tiene valores no numéricos y además hay registros con valor tbd. Antes de darles un tratamiento, se determina cuántos hay

In [53]:
```
non_numeric_user_score_count = df_merged[~df_merged[cons.USER_SCORE].apply(lambda x
print(f"Valores no numéricos en 'user_score': {non_numeric_user_score_count}")
```

```
Valores no numéricos en 'user_score': 5
```

Al ser pocos, se pueden reemplazar con 0 y la variable se convierte en numérica.

In [54]:
```
df_merged[cons.USER_SCORE] = df_merged[cons.USER_SCORE].apply(lambda x: 0 if not x.
df_merged[cons.USER_SCORE] = pd.to_numeric(df_merged[cons.USER_SCORE])
df_merged.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 354 entries, 483 to 777
Data columns (total 7 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   metascore   354 non-null    int64
 1   title       354 non-null    object
 2   user_score  354 non-null    float64
 3   GAMERS      354 non-null    int64
 4   COMP %      354 non-null    float64
 5   TIME        354 non-null    float64
 6   RATING      354 non-null    float64
dtypes: float64(4), int64(2), object(1)
memory usage: 22.1+ KB
```

Finalmente, se reemplazan los nombres de las columnas para una mejor comprensión y se normalizan las valoraciones. Para tener las valoraciones normalizadas, se cambian los campos `xbox_user_score` y `metacritic_user_score` a una escala de 0 a 100 y `metascore` a tipo de dato float.

In [55]:
```
df_merged.rename(columns={cons.RATING: cons.XBOX, cons.USER_SCORE: cons.METACRITIC_
df_merged['xbox_user_score'] = df_merged['xbox_user_score'] * 20
df_merged['metacritic_user_score'] = df_merged['metacritic_user_score'] * 10
df_merged['metascore'] = df_merged['metascore'].astype(float)
df_merged.reset_index(drop=True, inplace=True)
df_merged.head()
```

Out[55]:

	metascore	title	metacritic_user_score	GAMERS	COMP %	TIME	xbox_user_score
0	78.0	olija	0.0	4380	14.2	4.5	66.0
1	68.0	gang beasts	0.0	135527	9.3	1.5	58.0
2	56.0	recompile	0.0	7808	6.1	7.0	64.0
3	65.0	moonglow bay	0.0	4060	2.9	45.0	70.0
4	72.0	lawn mowing simulator	0.0	33974	0.1	70.0	66.0

Con esto, se puede empezar con el análisis.

Análisis

Para una mejor comprensión de cada variable, la siguiente tabla detalla de que va cada una

Columna/Variable	Descripción	Tipo	Importancia	Nota
metascore	Indica la valoración dada por los críticos avalados por Metacritic de cada título	Numérica	Alta	Nos ayuda a saber el veredicto de los "expertos".
title	Nombre de cada videojuego	Categórica	Media	Ayuda a identificar el nombre del juego, pero en este análisis damos más peso a sus valoraciones.
metacritic_user_score	Valoración por los usuarios/jugadores suscritos en Metacritic, pero que no tienen ese "plus" de ser críticos avalados por la plataforma	Numérica	Alta	Aunque hay expertos que dan valoraciones a los videojuegos, la comunidad también tiene voz y peso en sus opiniones.
GAMERS	Número de jugadores que han jugado el título	Numérica	Alta	Con el número de jugadores podemos determinar el impacto que tuvo el título desde que se añadió al catálogo.
COMP %	Porcentaje de jugadores que han completado el título	Numérica	Alta	Nos ayuda a saber cuántos jugadores han aprovechado el servicio para completar los títulos en el catálogo.
TIME	Tiempo medio para completar el título	Numérica	Media	Puede ayudarnos a saber cuánto tiempo tarda un jugador en promedio para completar el título y analizar si es un factor clave.

Columna/Variable	Descripción	Tipo	Importancia	Nota
xbox_user_score	Calificación del título dada por usuarios de Xbox	Numérica	Media	Aunque es una valoración dada por usuarios de Xbox, no nos da mucha certeza, ya que se restringe a usuarios de esta plataforma y son pocas las calificaciones.

Ahora, se harán análisis univariantes, bivariantes y multivariantes.

Análisis univariante

Empezamos haciendo un análisis univariante de cada variable numérica para entender su distribución (ya que son las que nos interesan en ese análisis).

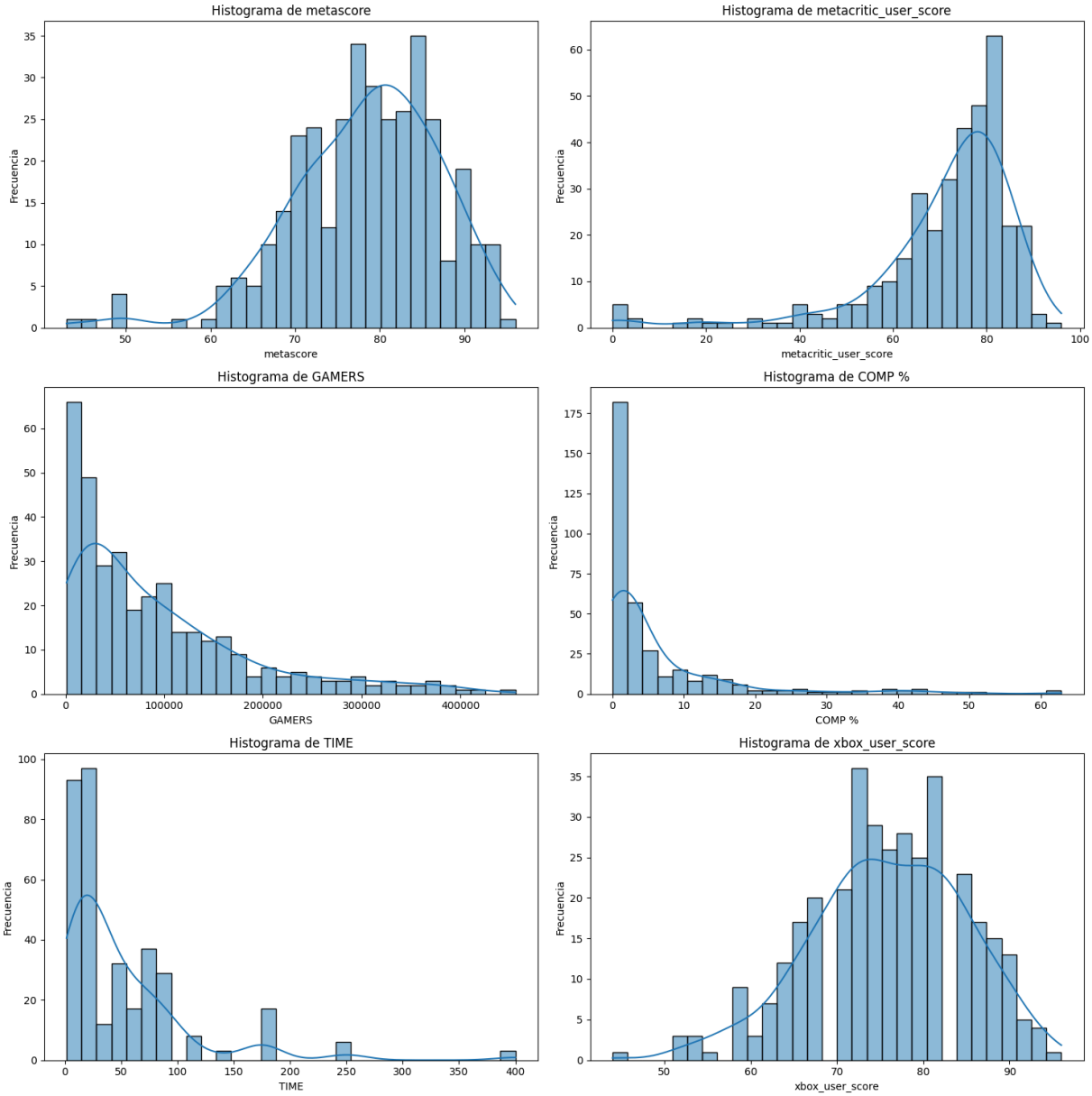
In [20]:
```
numeric_columns = df_merged.select_dtypes(include=[np.number]).columns
df_merged[numeric_columns].describe()
```

Out[20]:

	metascore	metacritic_user_score	GAMERS	COMP %	TIME	xbox_user_
count	354.000000	354.000000	354.000000	354.000000	354.000000	354.0
mean	78.437853	70.988701	93647.059322	6.007627	50.430791	75.8
std	8.693577	16.304291	93057.468284	10.128412	59.118143	8.9
min	43.000000	0.000000	1065.000000	0.000000	1.500000	44.0
25%	73.000000	67.000000	23690.250000	0.600000	13.500000	70.0
50%	79.500000	75.000000	62594.500000	1.900000	27.500000	76.0
75%	84.750000	81.000000	131071.500000	6.100000	70.000000	82.0
max	96.000000	96.000000	455839.000000	62.900000	400.000000	96.0

Graficamos histogramas de las variables numéricas.

In [21]:
```
funtils.plot_numerical_histograms(df_merged, numeric_columns, bins_list=[30,30,30,3
```



Revisando las gráficas, se pueden tener unas primeras conclusiones:

- En su debida proporción, las valoraciones dadas por criticos de Metacritic, usuarios de Metacritic y usuarios de Xbox en el valor máximo pero los estadísticos de las valoraciones de críticos de Metacritic y usuarios de Xbox se parecen mucho más.
- Como se vió antes, hay juegos que durán alrededor de 30 horas en su mayoría.
- Después de hacer el corte de número de jugadores en cada juego, se tiene una media significativa de alrededor de 100,000 jugadores
- Hay bastantes juegos que no han sido completados, ni siquiera llevan un 10% de completado. Habría que analizar que valoraciones tienen esos juegos.

Análisis bivariante

Para el análisis bivariante, se harán unas variables categóricas de cada variable de valoración teniendo 5 categorías para cada uno: "Muy positivas", "Positivas", "Variadas", "Negativas" y

"Muy negativas".

In [22]:
```
df_merged[cons.METACRITIC_USER_SCORE_CAT] = funtils.categorize_score(df_merged[cons
df_merged[cons.XBOX_USER_SCORE_CAT] = funtils.categorize_score(df_merged[cons.XBOX]
df_merged[cons.METASCORE_CAT] = funtils.categorize_score(df_merged[cons.METASCORE],
df_merged.head()
```

Out[22]:

	metascore	title	metacritic_user_score	GAMERS	COMP %	TIME	xbox_user_score	r
0	78.0	olija	0.0	4380	14.2	4.5	66.0	
1	68.0	gang beasts	0.0	135527	9.3	1.5	58.0	
2	56.0	recompile	0.0	7808	6.1	7.0	64.0	
3	65.0	moonglow bay	0.0	4060	2.9	45.0	70.0	
4	72.0	lawn mowing simulator	0.0	33974	0.1	70.0	66.0	

Con estas nuevas variables, será más sencillo el análisis con las valoraciones.

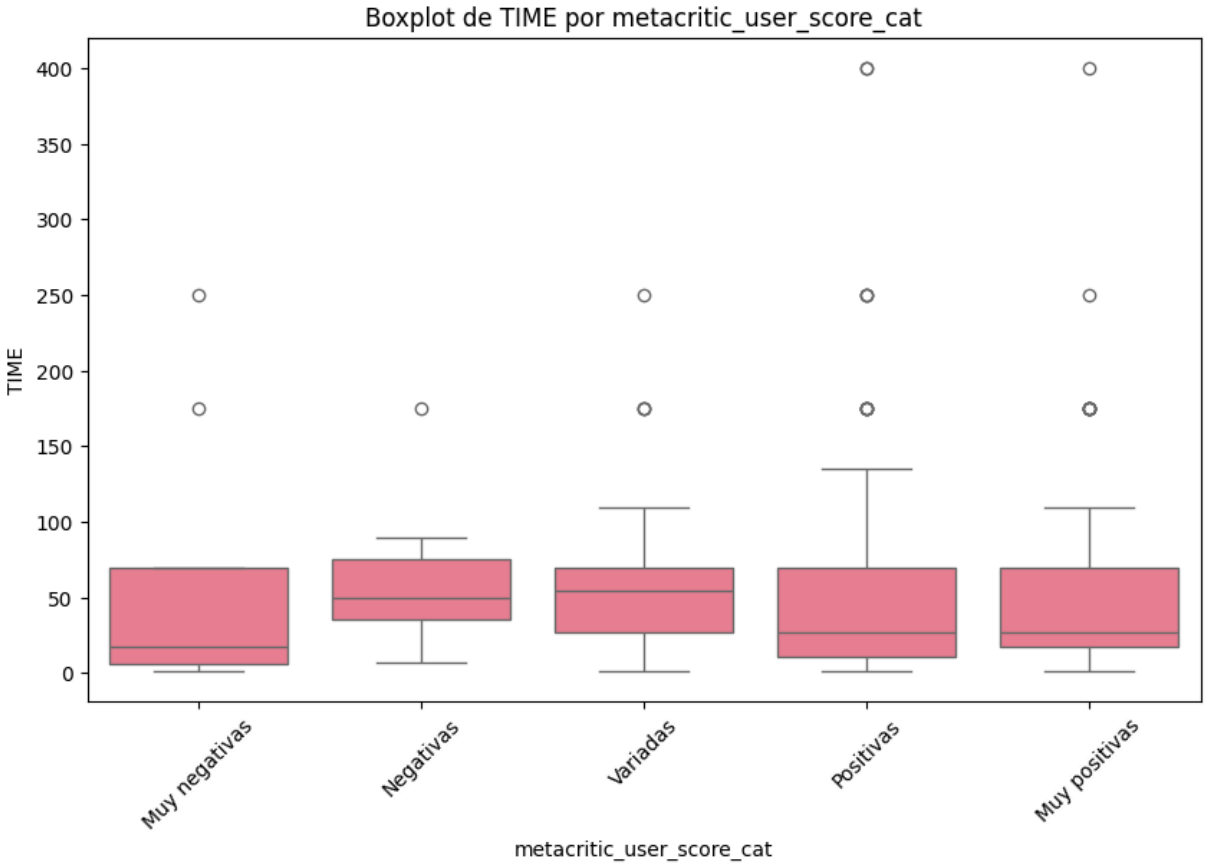
Análisis entre categóricas y numéricas

Para este análisis, se usarán boxplots (que sirve para interpretar los datos estadísticos) e histogramas (esto nos servirá para presentar resultados).

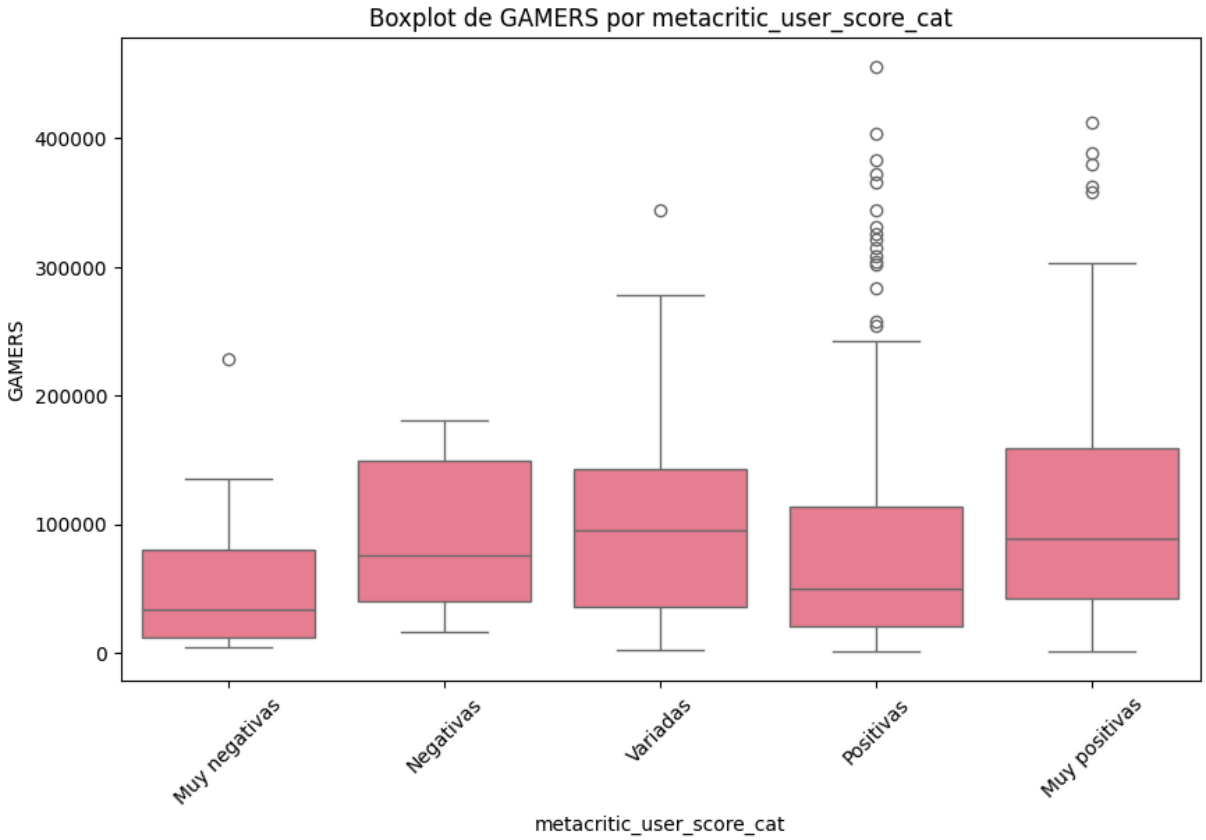
Empezamos con los boxplots.

In [23]:
```
categorical_columns = [cons.METACRITIC_USER_SCORE_CAT, cons.XBOX_USER_SCORE_CAT, co
numerical_columns_s = [cons.TIME, cons.GAMERS, cons.COMP]

funtils.plot_categorical_numerical_boxplots(df_merged, categorical_columns, numeric
```

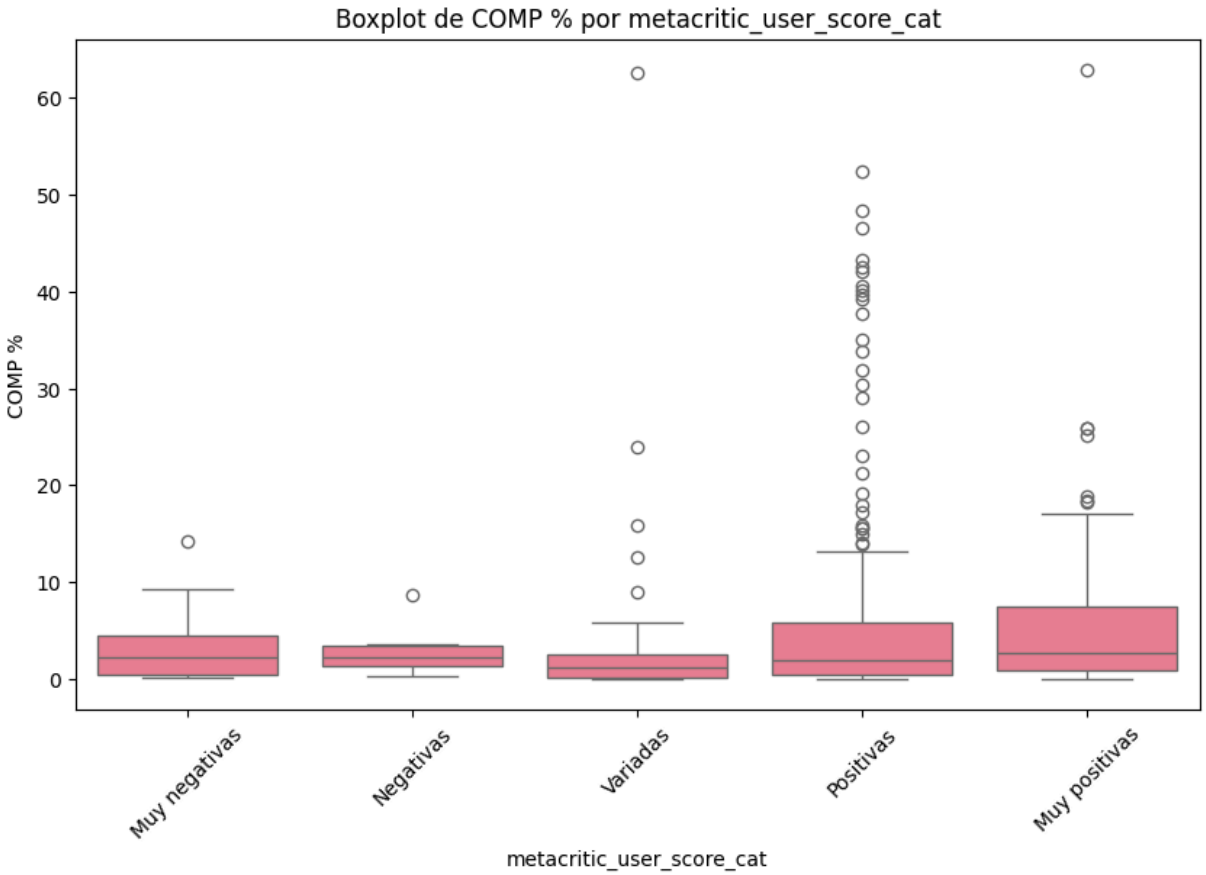
```
c:\Users\PC ELITE\Desktop\TheBridge\Proyectos\EjerciciosPersonales\EDA_Entrega\src/u
tils\funciones.py:59: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(x=df[cat_col], y=df[num_col], palette=[colors[color_idx]])
c:\Users\PC ELITE\Desktop\TheBridge\Proyectos\EjerciciosPersonales\EDA_Entrega\src/u
tils\funciones.py:59: UserWarning:
The palette list has fewer values (1) than needed (5) and will cycle, which may prod
uce an uninterpretable plot.
  sns.boxplot(x=df[cat_col], y=df[num_col], palette=[colors[color_idx]])
```



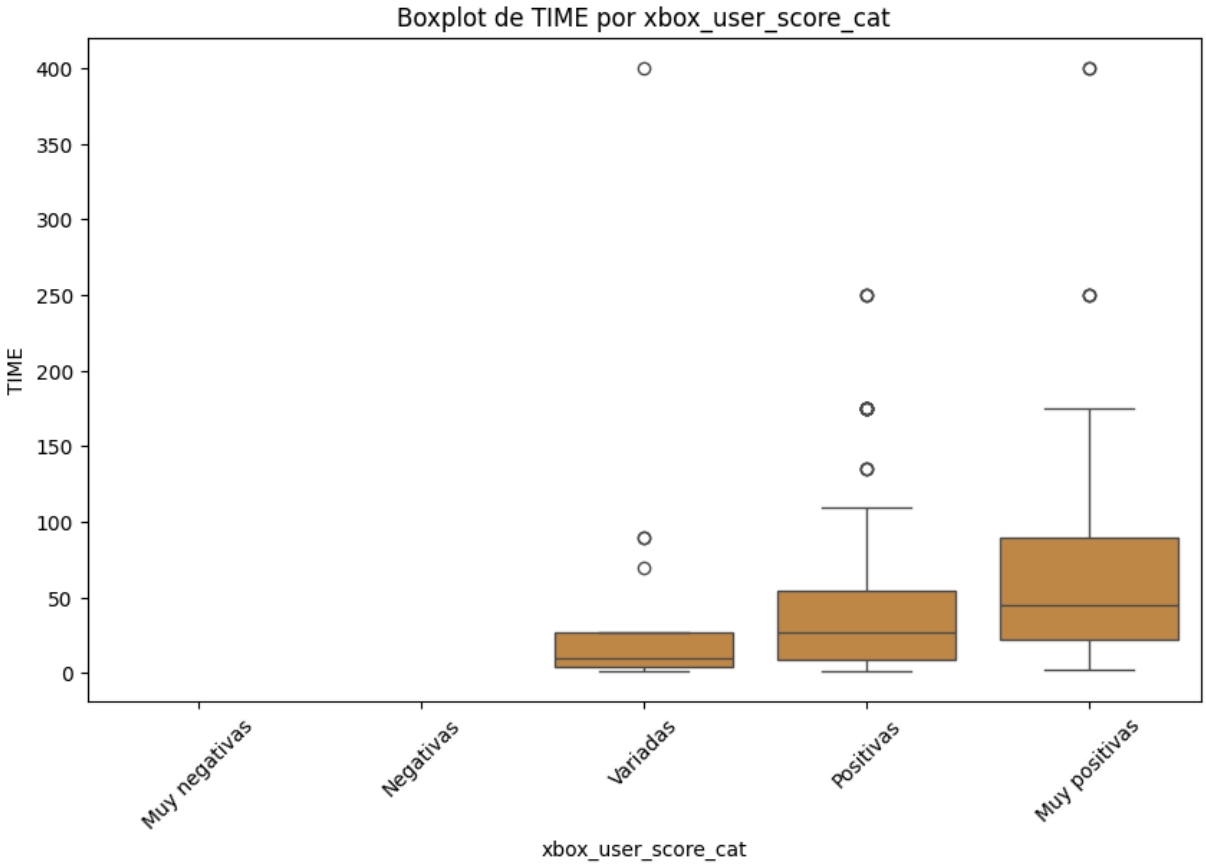

```
c:\Users\PC ELITE\Desktop\TheBridge\Proyectos\EjerciciosPersonales\EDA_Entrega\src/u
tils\funciones.py:59: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(x=df[cat_col], y=df[num_col], palette=[colors[color_idx]])
c:\Users\PC ELITE\Desktop\TheBridge\Proyectos\EjerciciosPersonales\EDA_Entrega\src/u
tils\funciones.py:59: UserWarning:
The palette list has fewer values (1) than needed (5) and will cycle, which may prod
uce an uninterpretable plot.
  sns.boxplot(x=df[cat_col], y=df[num_col], palette=[colors[color_idx]])
```



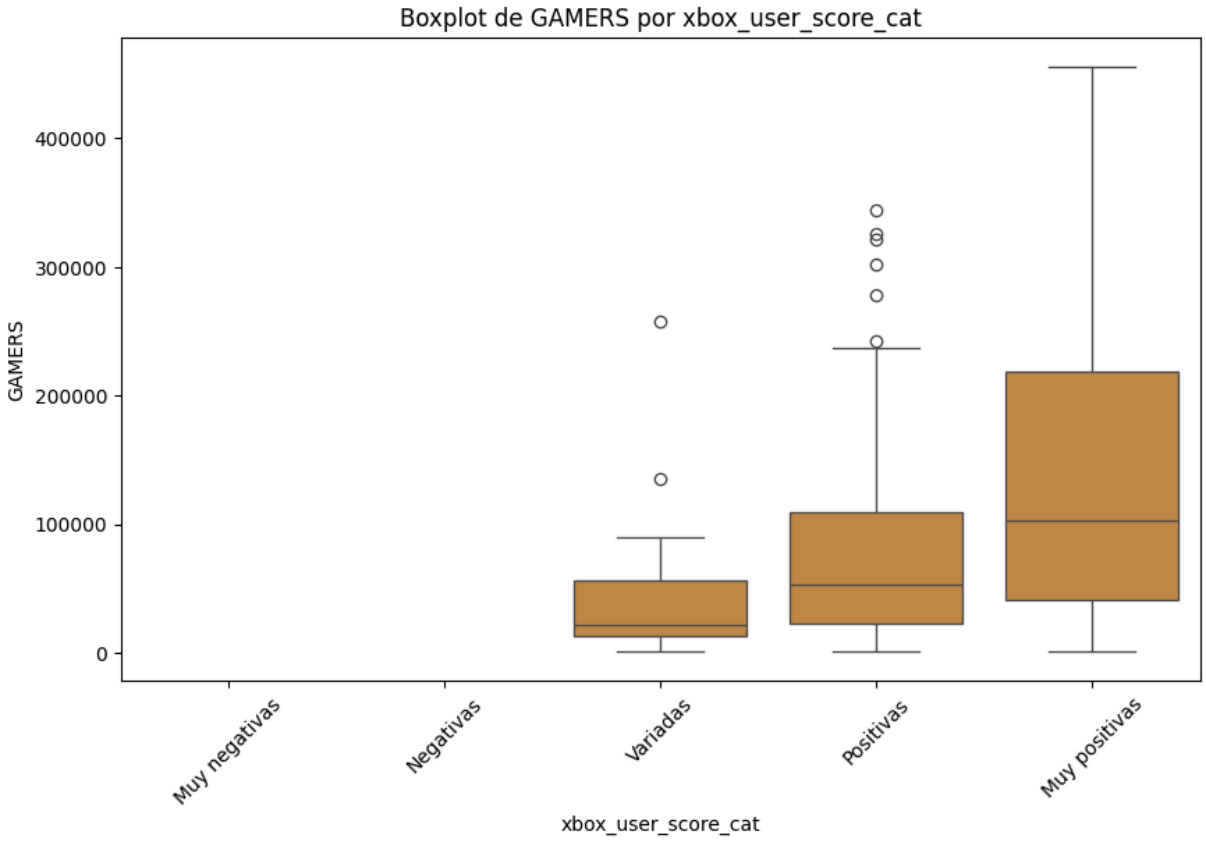

```
c:\Users\PC ELITE\Desktop\TheBridge\Proyectos\EjerciciosPersonales\EDA_Entrega\src/u
tils\funciones.py:59: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(x=df[cat_col], y=df[num_col], palette=[colors[color_idx]])
c:\Users\PC ELITE\Desktop\TheBridge\Proyectos\EjerciciosPersonales\EDA_Entrega\src/u
tils\funciones.py:59: UserWarning:
The palette list has fewer values (1) than needed (5) and will cycle, which may prod
uce an uninterpretable plot.
  sns.boxplot(x=df[cat_col], y=df[num_col], palette=[colors[color_idx]])
```



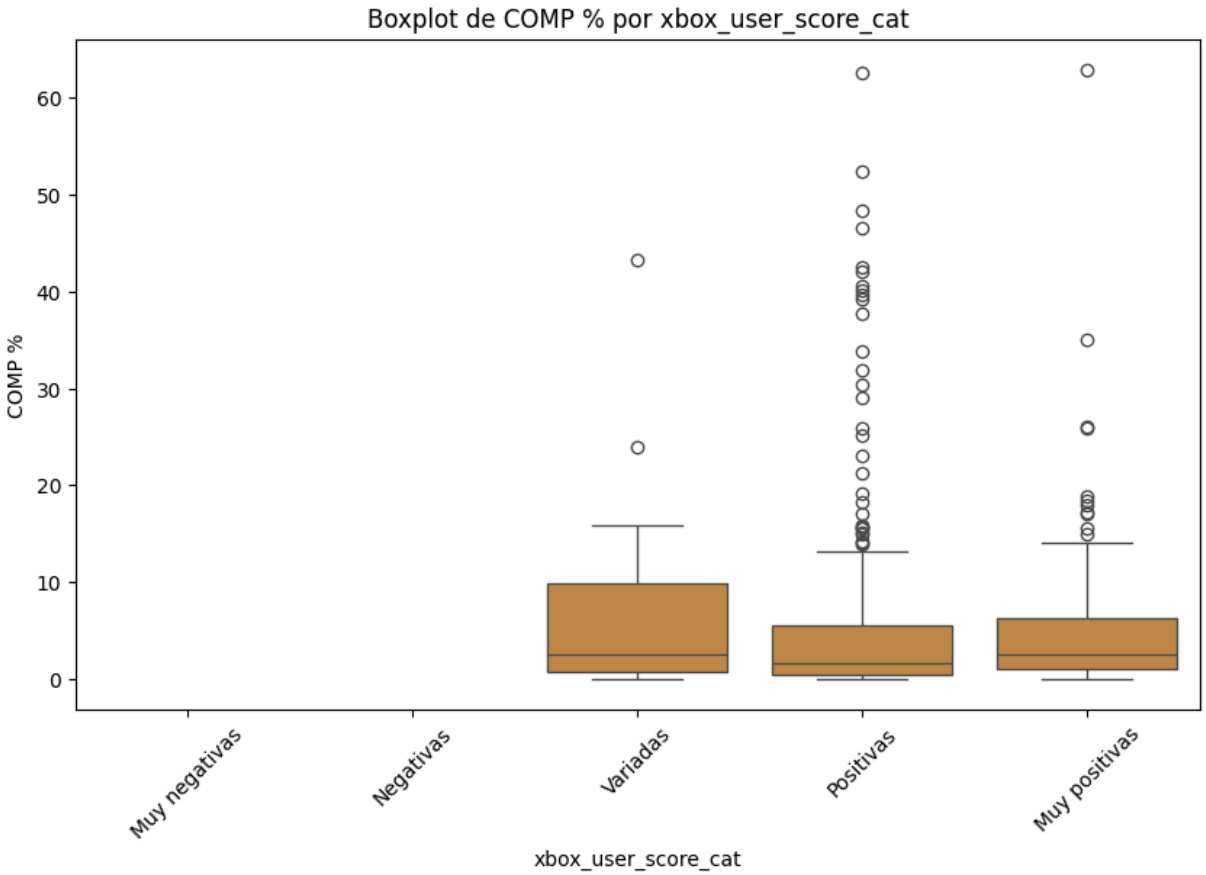

```
c:\Users\PC ELITE\Desktop\TheBridge\Proyectos\EjerciciosPersonales\EDA_Entrega\src/u
tils\funciones.py:59: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(x=df[cat_col], y=df[num_col], palette=[colors[color_idx]])
c:\Users\PC ELITE\Desktop\TheBridge\Proyectos\EjerciciosPersonales\EDA_Entrega\src/u
tils\funciones.py:59: UserWarning:
The palette list has fewer values (1) than needed (5) and will cycle, which may prod
uce an uninterpretable plot.
  sns.boxplot(x=df[cat_col], y=df[num_col], palette=[colors[color_idx]])
```



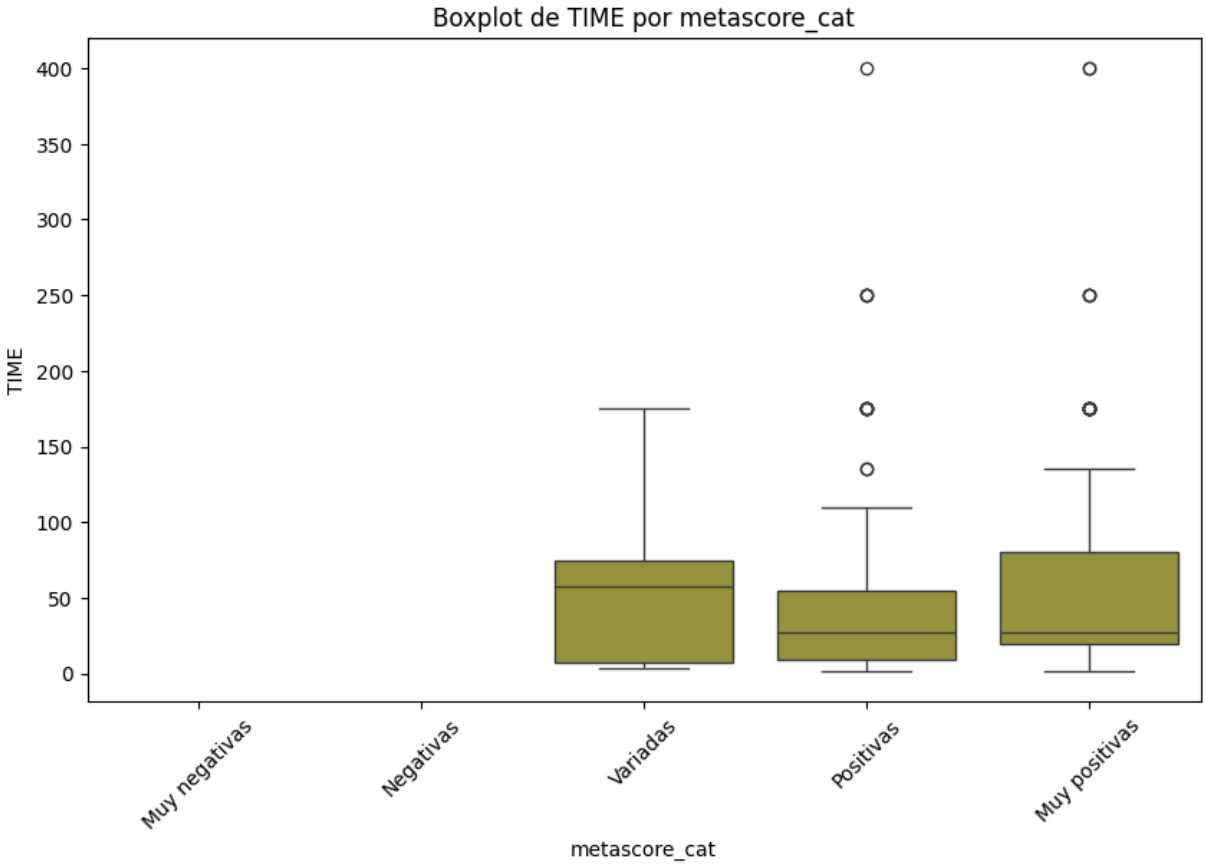

```
c:\Users\PC ELITE\Desktop\TheBridge\Proyectos\EjerciciosPersonales\EDA_Entrega\src/u
tils\funciones.py:59: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(x=df[cat_col], y=df[num_col], palette=[colors[color_idx]])
c:\Users\PC ELITE\Desktop\TheBridge\Proyectos\EjerciciosPersonales\EDA_Entrega\src/u
tils\funciones.py:59: UserWarning:
The palette list has fewer values (1) than needed (5) and will cycle, which may prod
uce an uninterpretable plot.
  sns.boxplot(x=df[cat_col], y=df[num_col], palette=[colors[color_idx]])
```



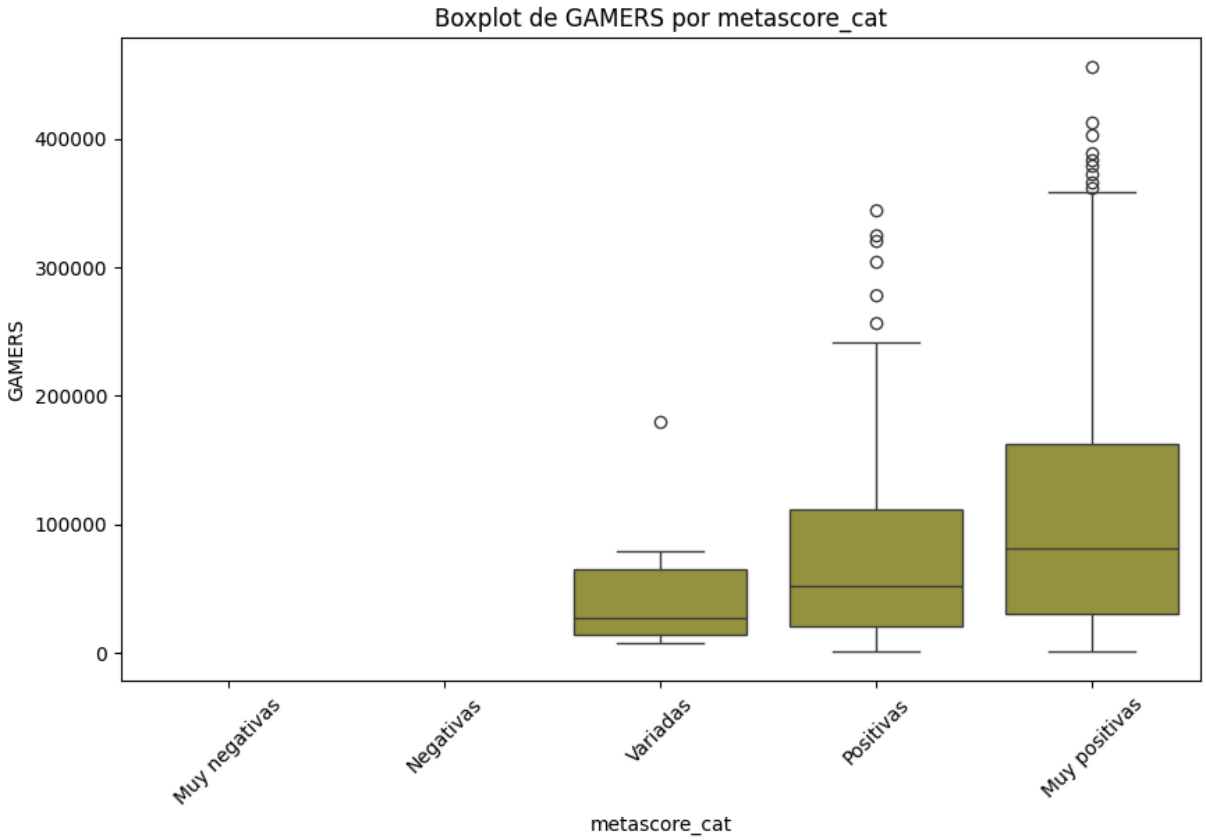

```
c:\Users\PC ELITE\Desktop\TheBridge\Proyectos\EjerciciosPersonales\EDA_Entrega\src/u
tils\funciones.py:59: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(x=df[cat_col], y=df[num_col], palette=[colors[color_idx]])
c:\Users\PC ELITE\Desktop\TheBridge\Proyectos\EjerciciosPersonales\EDA_Entrega\src/u
tils\funciones.py:59: UserWarning:
The palette list has fewer values (1) than needed (5) and will cycle, which may prod
uce an uninterpretable plot.
  sns.boxplot(x=df[cat_col], y=df[num_col], palette=[colors[color_idx]])
```



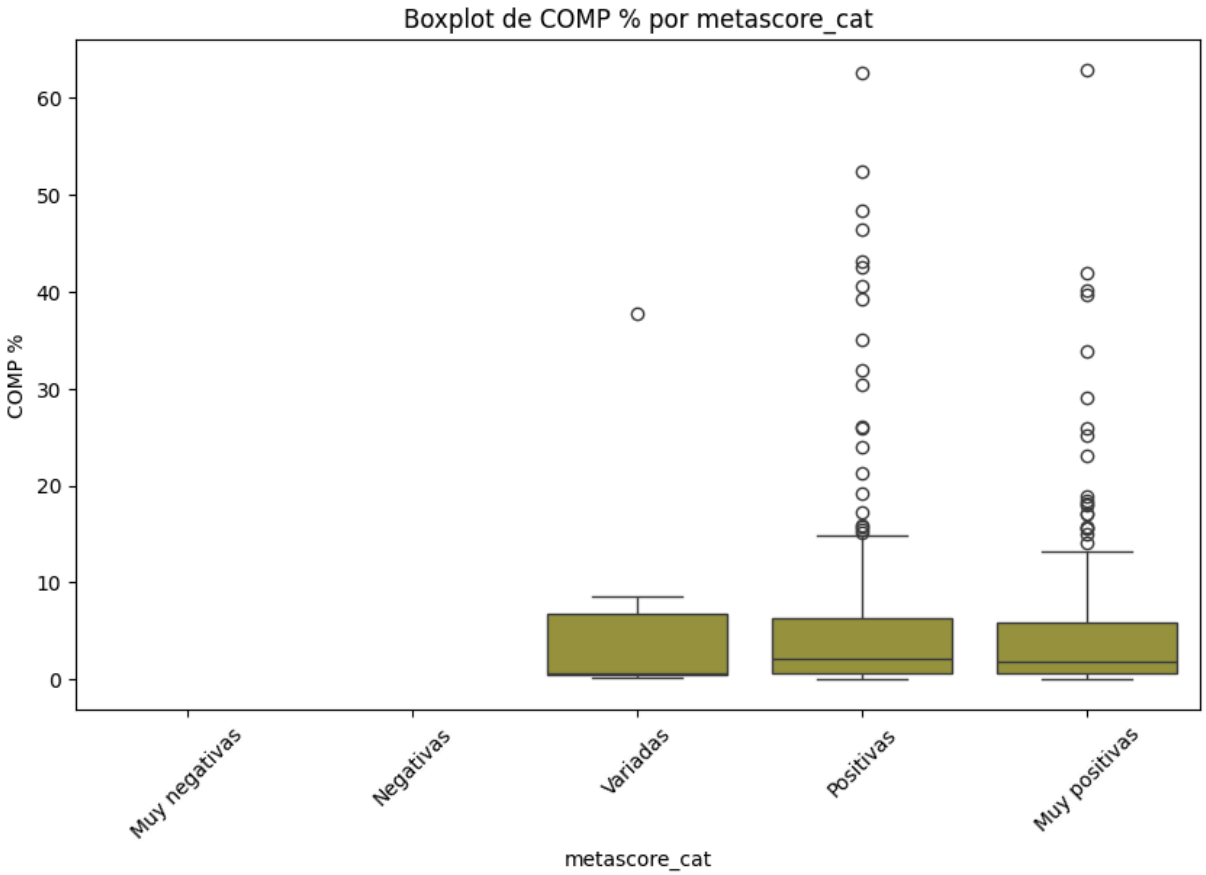

```
c:\Users\PC ELITE\Desktop\TheBridge\Proyectos\EjerciciosPersonales\EDA_Entrega\src/u
tils\funciones.py:59: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(x=df[cat_col], y=df[num_col], palette=[colors[color_idx]])
c:\Users\PC ELITE\Desktop\TheBridge\Proyectos\EjerciciosPersonales\EDA_Entrega\src/u
tils\funciones.py:59: UserWarning:
The palette list has fewer values (1) than needed (5) and will cycle, which may prod
uce an uninterpretable plot.
  sns.boxplot(x=df[cat_col], y=df[num_col], palette=[colors[color_idx]])
```




```
c:\Users\PC ELITE\Desktop\TheBridge\Proyectos\EjerciciosPersonales\EDA_Entrega\src/u
tils\funciones.py:59: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(x=df[cat_col], y=df[num_col], palette=[colors[color_idx]])
c:\Users\PC ELITE\Desktop\TheBridge\Proyectos\EjerciciosPersonales\EDA_Entrega\src/u
tils\funciones.py:59: UserWarning:
The palette list has fewer values (1) than needed (5) and will cycle, which may prod
uce an uninterpretable plot.
  sns.boxplot(x=df[cat_col], y=df[num_col], palette=[colors[color_idx]])
```

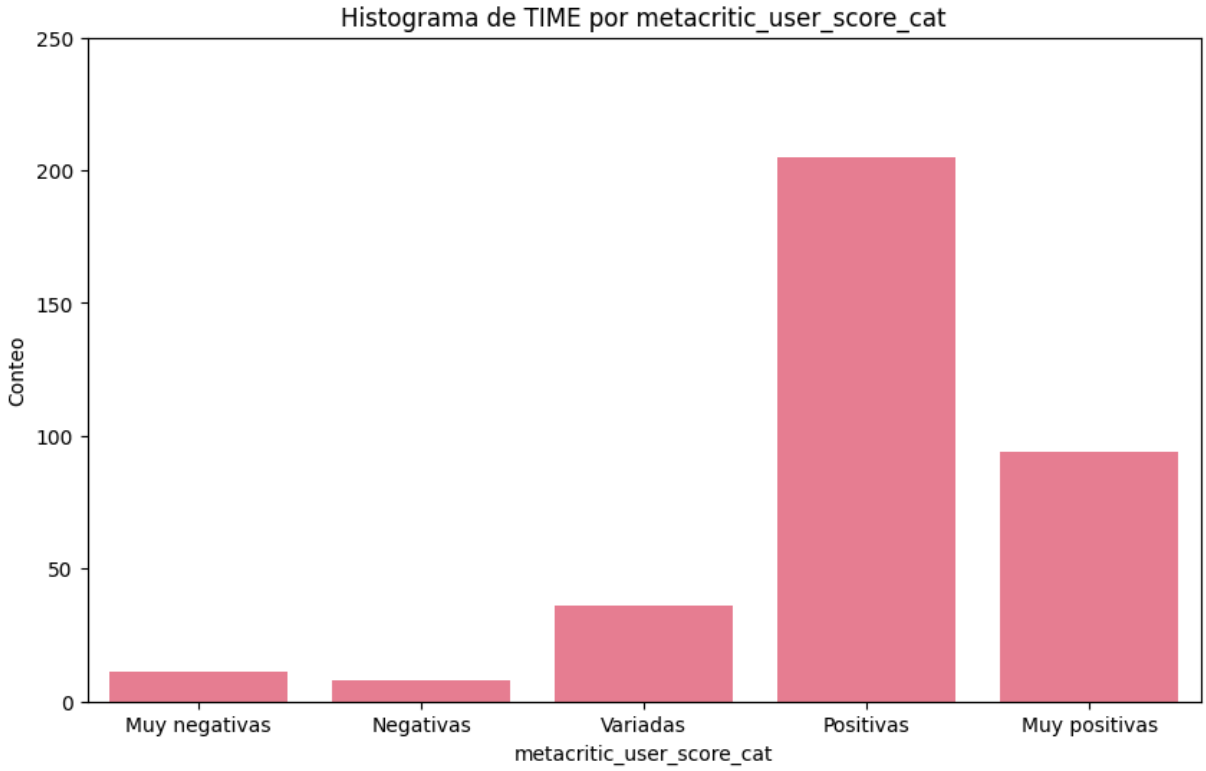


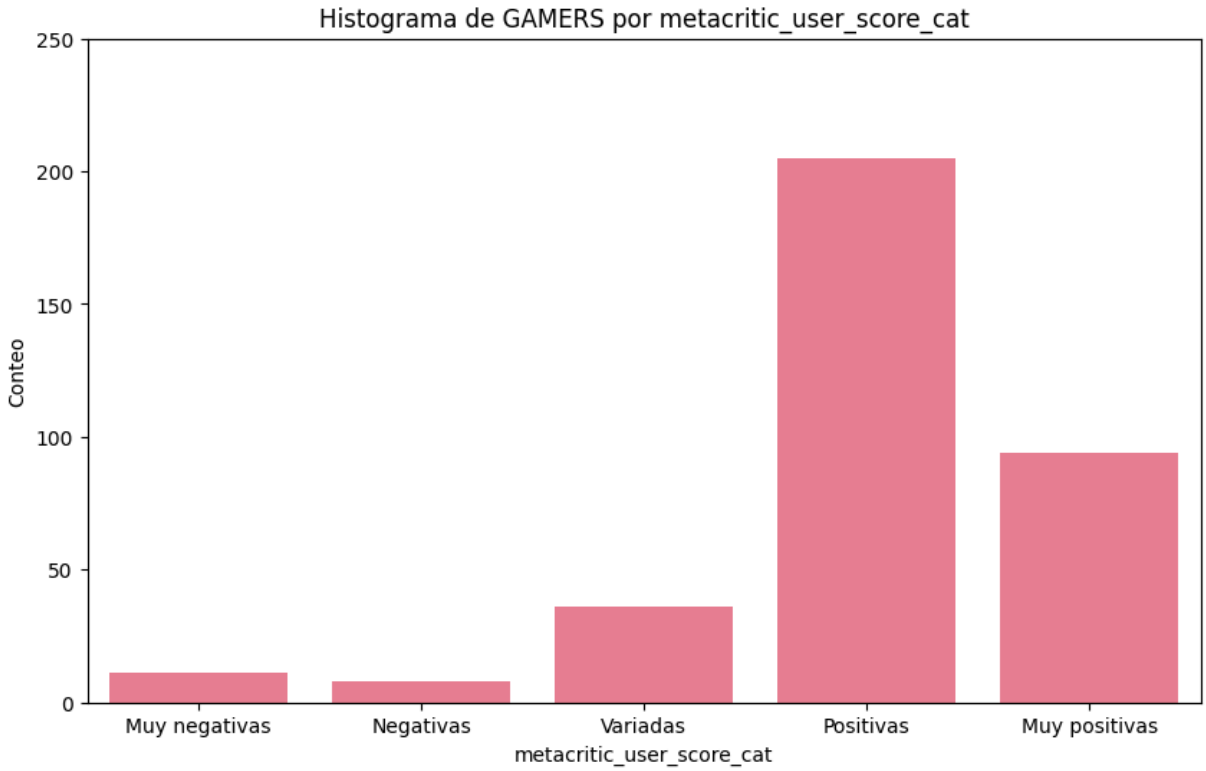

```
c:\Users\PC ELITE\Desktop\TheBridge\Proyectos\EjerciciosPersonales\EDA_Entrega\src/u
tils\funciones.py:59: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.1
4.0. Assign the `x` variable to `hue` and set `legend=False` for the same effect.

  sns.boxplot(x=df[cat_col], y=df[num_col], palette=[colors[color_idx]])
c:\Users\PC ELITE\Desktop\TheBridge\Proyectos\EjerciciosPersonales\EDA_Entrega\src/u
tils\funciones.py:59: UserWarning:
The palette list has fewer values (1) than needed (5) and will cycle, which may prod
uce an uninterpretable plot.
  sns.boxplot(x=df[cat_col], y=df[num_col], palette=[colors[color_idx]])
```

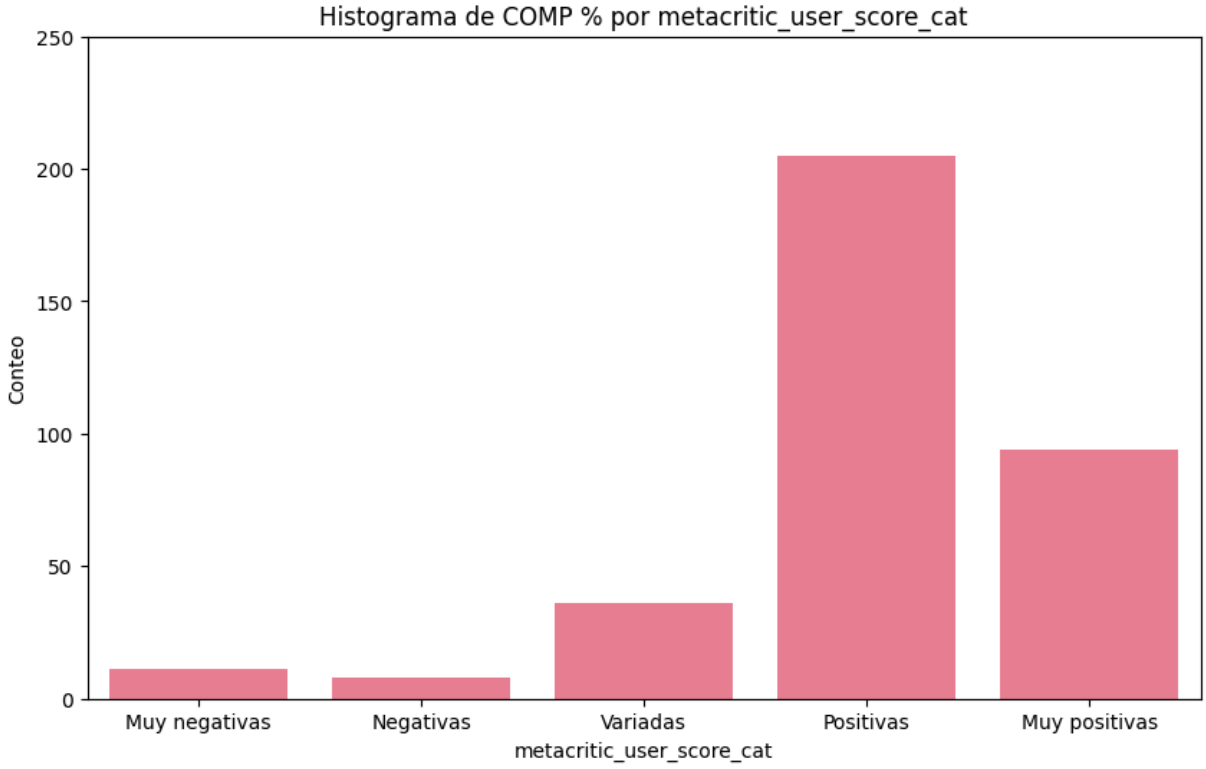


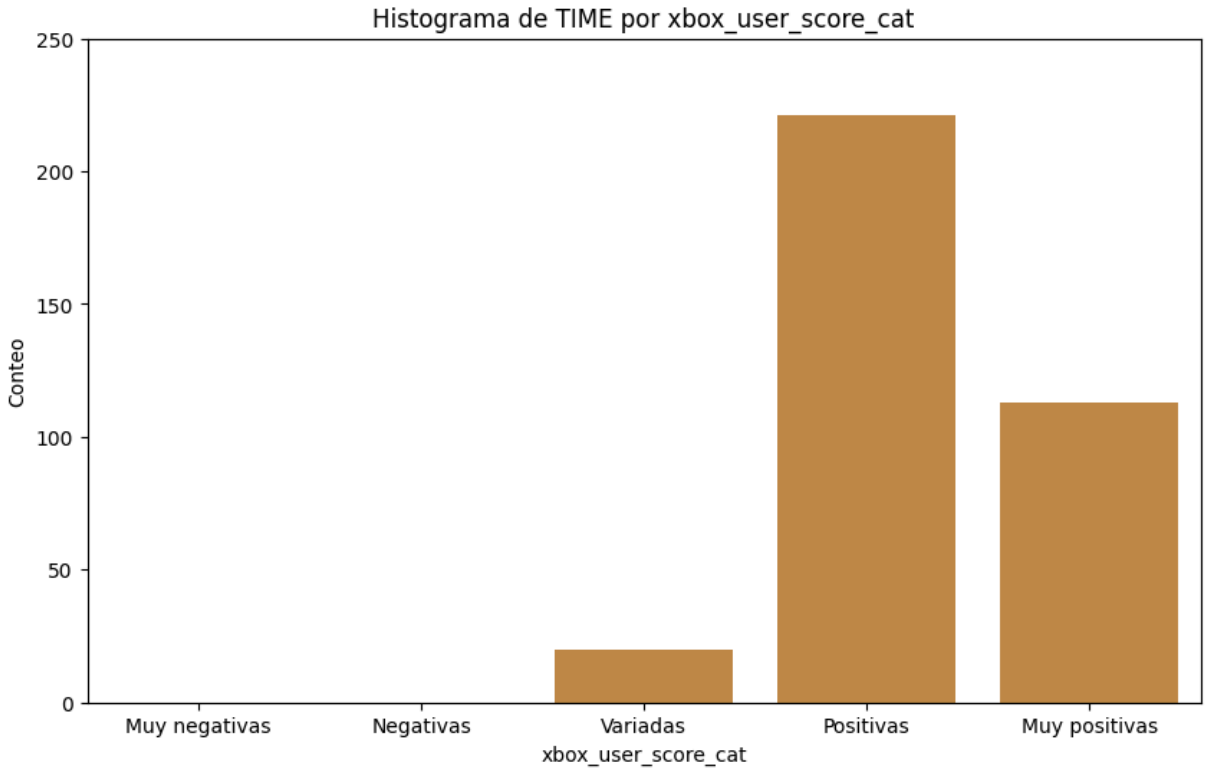
Continuamos con los histogramas.

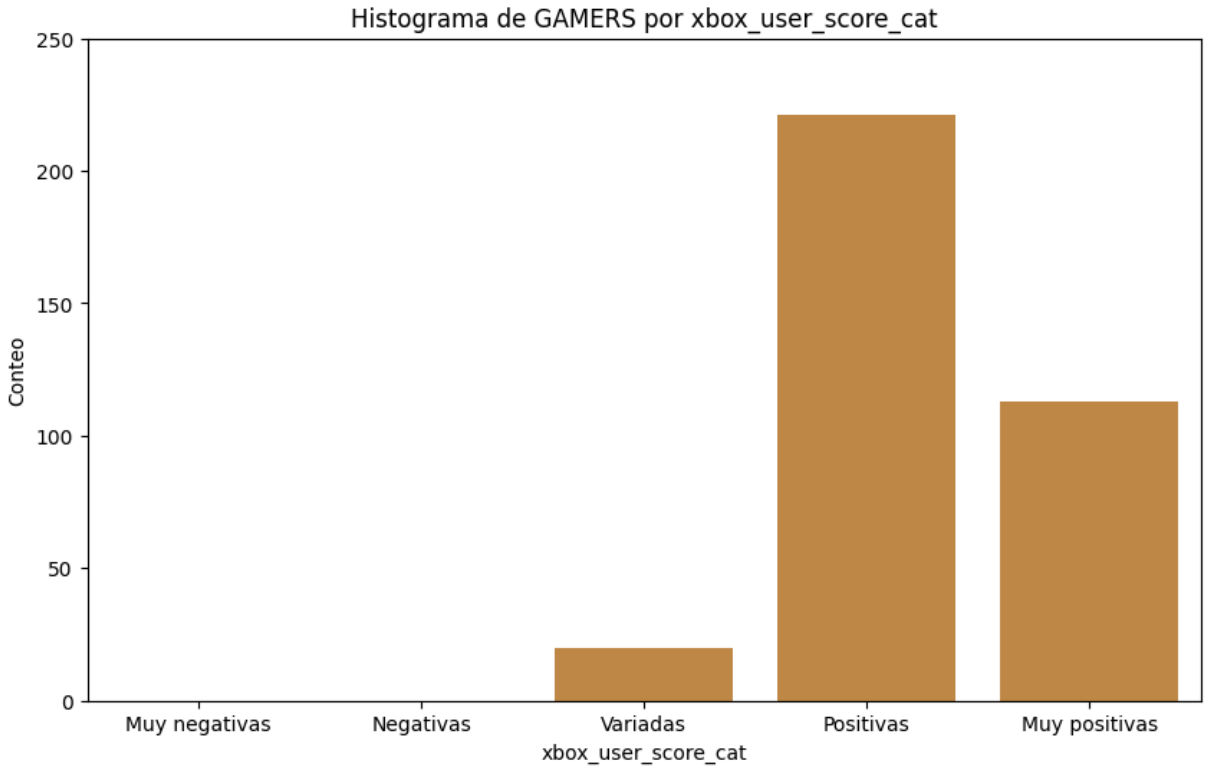
```
In [24]: funtils.plot_categorical_numerical_histograms(df_merged, categorical_columns, numer
```

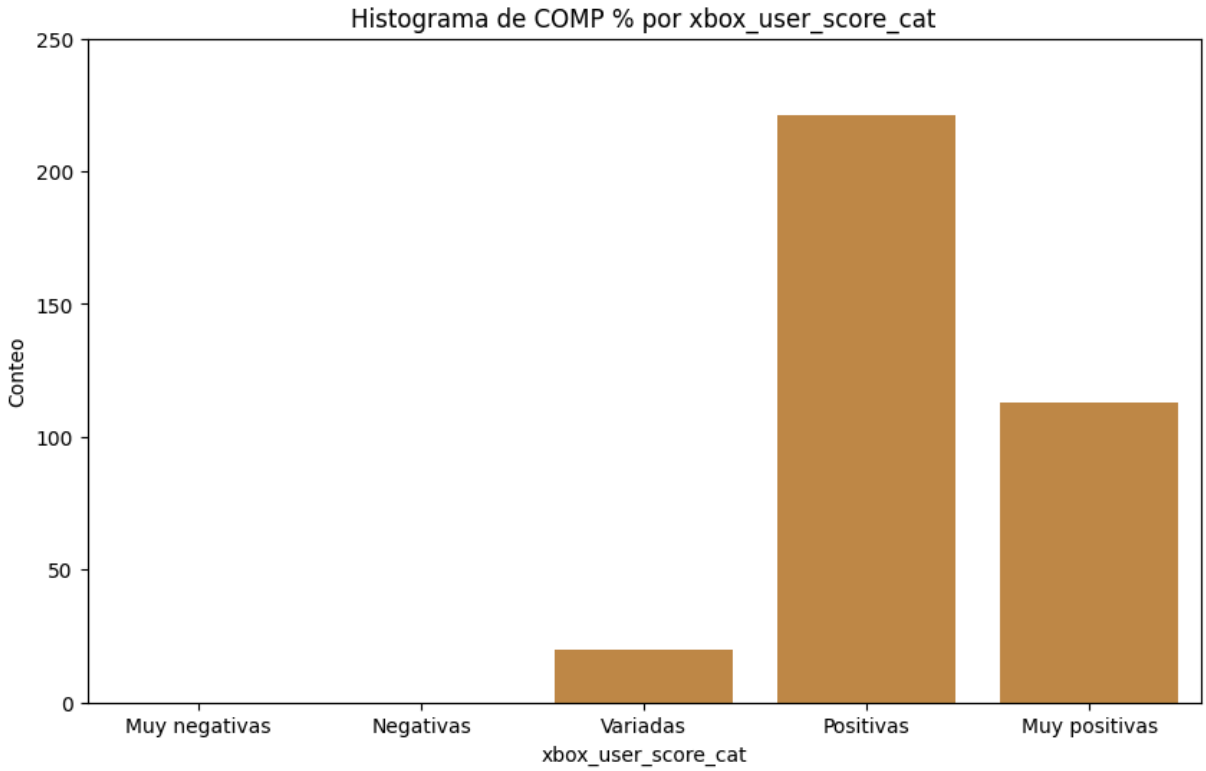


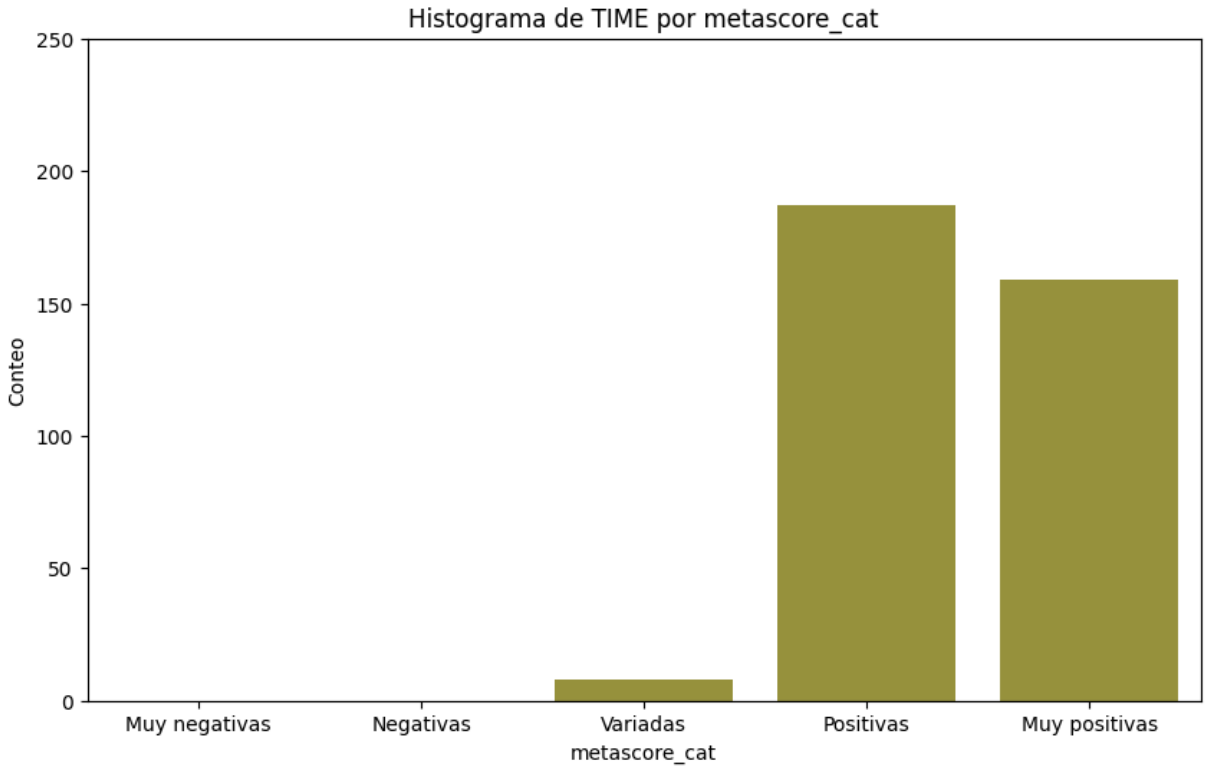

Histograma de GAMERS por metacritic_user_score_cat
250
200
150
100
50
0
Conteo
Muy negativas
Negativas
Variadas
Positivas
Muy positivas
metacritic_user_score_cat

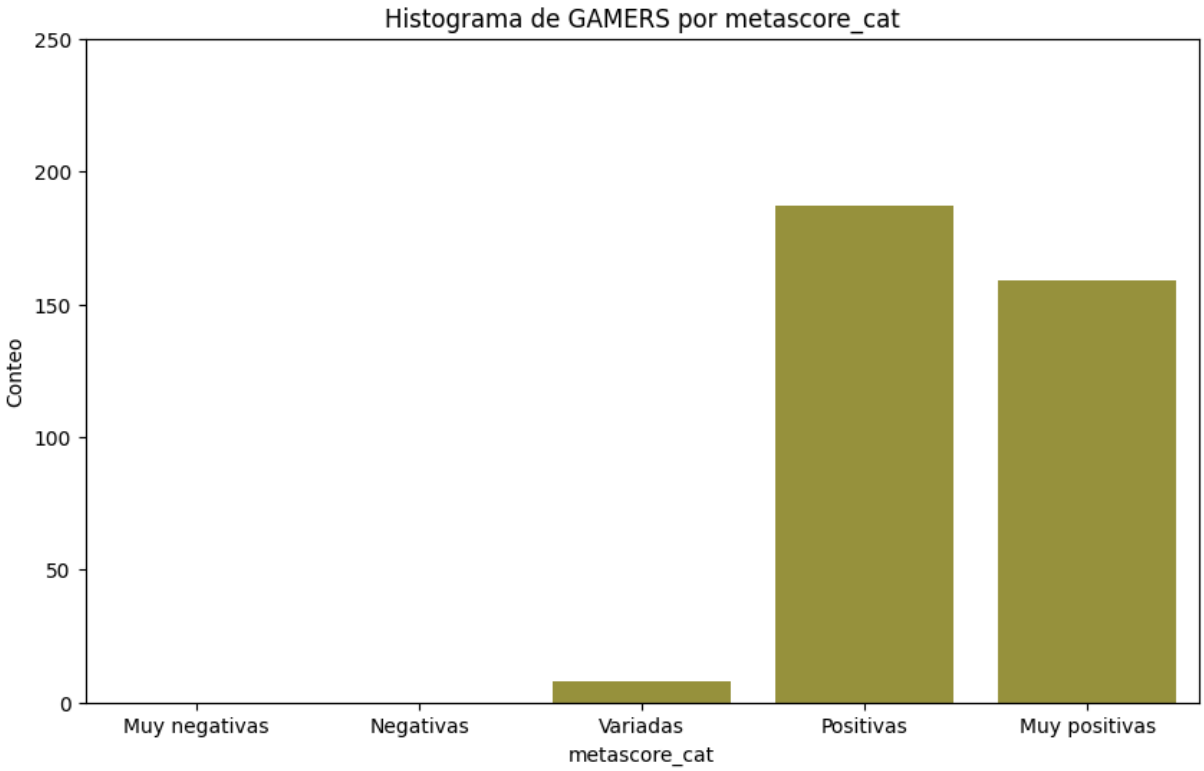

Histograma de COMP % por metacritic_user_score_cat
250
200
150
100
50
0
Conteo
Muy negativas
Negativas
Variadas
Positivas
Muy positivas
metacritic_user_score_cat

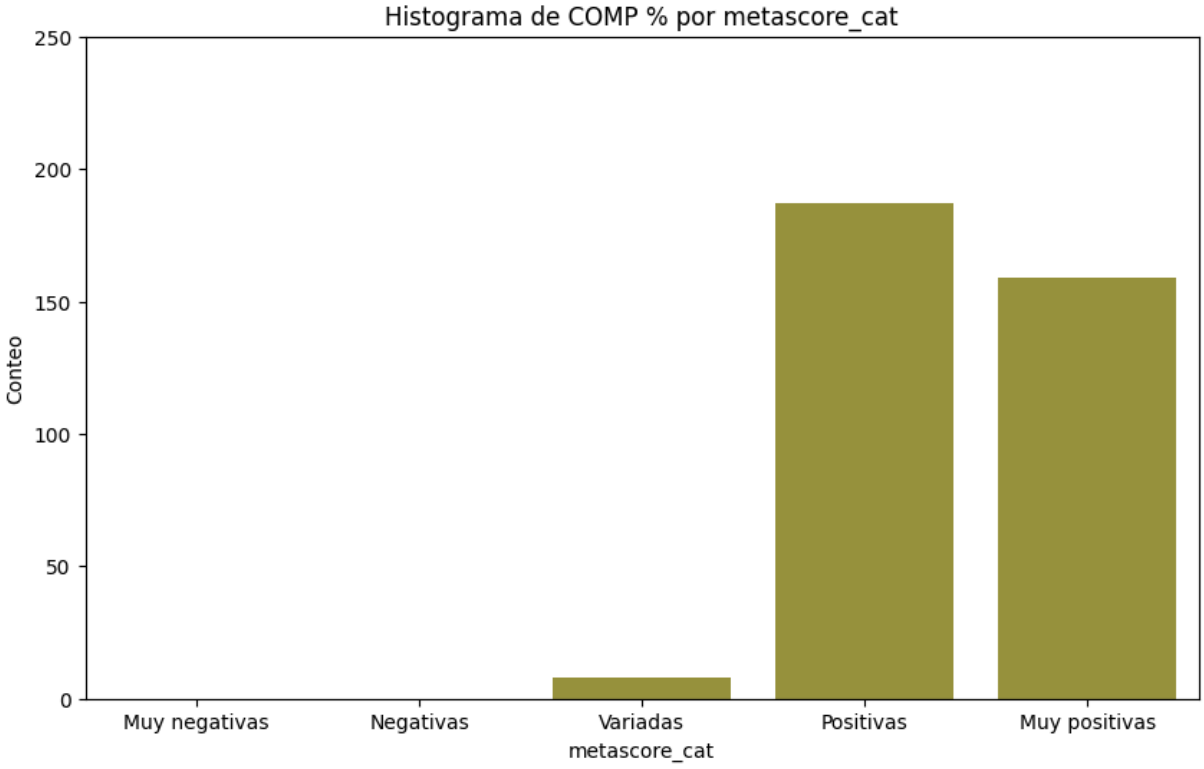

Histograma de TIME por xbox_user_score_cat
250
200
150
100
50
0
Conteo
Muy negativas
Negativas
Variadas
Positivas
Muy positivas
xbox_user_score_cat


Histograma de GAMERS por xbox_user_score_cat
250
200
150
100
50
0
Conteo
Muy negativas
Negativas
Variadas
Positivas
Muy positivas
xbox_user_score_cat


Histograma de COMP % por xbox_user_score_cat
250
200
150
100
50
0
Conteo
Muy negativas
Negativas
Variadas
Positivas
Muy positivas
xbox_user_score_cat


Histograma de TIME por metascore_cat
250
200
150
100
50
0
Conteo
Muy negativas
Negativas
Variadas
Positivas
Muy positivas
metascore_cat





Se revisarán algunos registros en los que `metacritic_user_score_cat` caen en la categoría 'Muy negativas' y ver si son valores igual a cero

In [25]:
```
df_merged[(df_merged[cons.METACRITIC_USER_SCORE_CAT] == 'Muy negativas')].sort_valu
```

Out[25]:

	metascore	title	metacritic_user_score	GAMERS	COMP %	TIME	xbox_user_score
0	78.0	olija	0.0	4380	14.2	4.5	66.0
1	68.0	gang beasts	0.0	135527	9.3	1.5	58.0
2	56.0	recompile	0.0	7808	6.1	7.0	64.0
3	65.0	moonglow bay	0.0	4060	2.9	45.0	70.0
4	72.0	lawn mowing simulator	0.0	33974	0.1	70.0	66.0
348	82.0	ea sports ufc 4	20.0	66145	0.5	17.5	68.0
349	66.0	star wars battlefront ii	19.0	228476	1.7	70.0	76.0
350	79.0	fifa 20	18.0	93669	0.3	175.0	66.0
351	77.0	fifa 21	15.0	38890	0.5	250.0	66.0
352	70.0	madden nfl 22	6.0	15217	2.2	3.5	54.0
353	69.0	madden nfl 21	5.0	22414	3.0	9.0	64.0

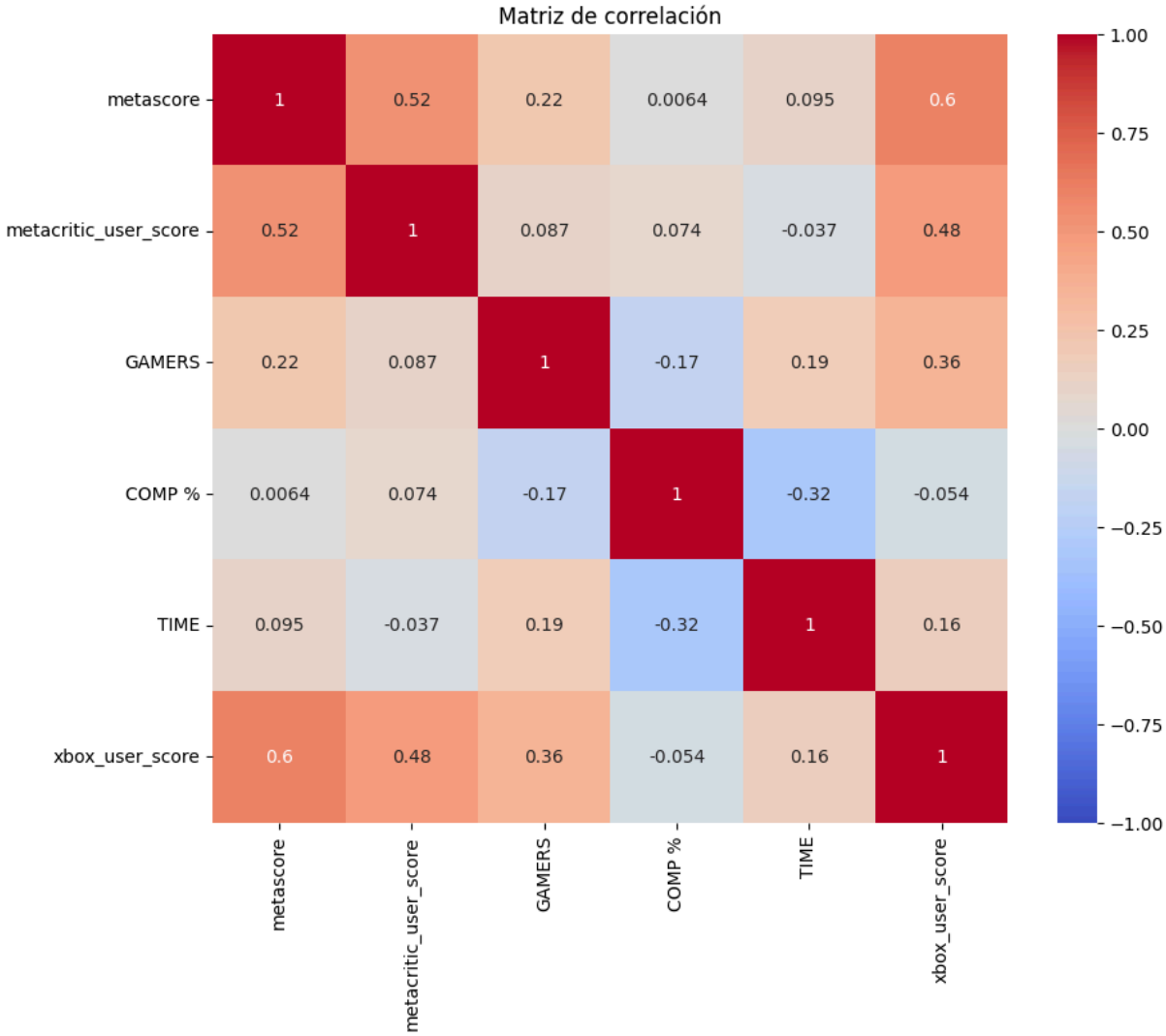
La mitad de registros que son catalogados como "Muy Negativo" en `metacritic_user_score_cat` tienen un valor de cero en su homólogo numérico. Puede significar que hay usuarios que no han calificado esos títulos.

Análisis entre numéricas

Para el análisis entre variables numéricas, se usará un mapa de calor para determinar si hay correlación entre variables. Así mismo se usará `pairplot` para revisar gráficos de dispersión y encontrar algún patrón.

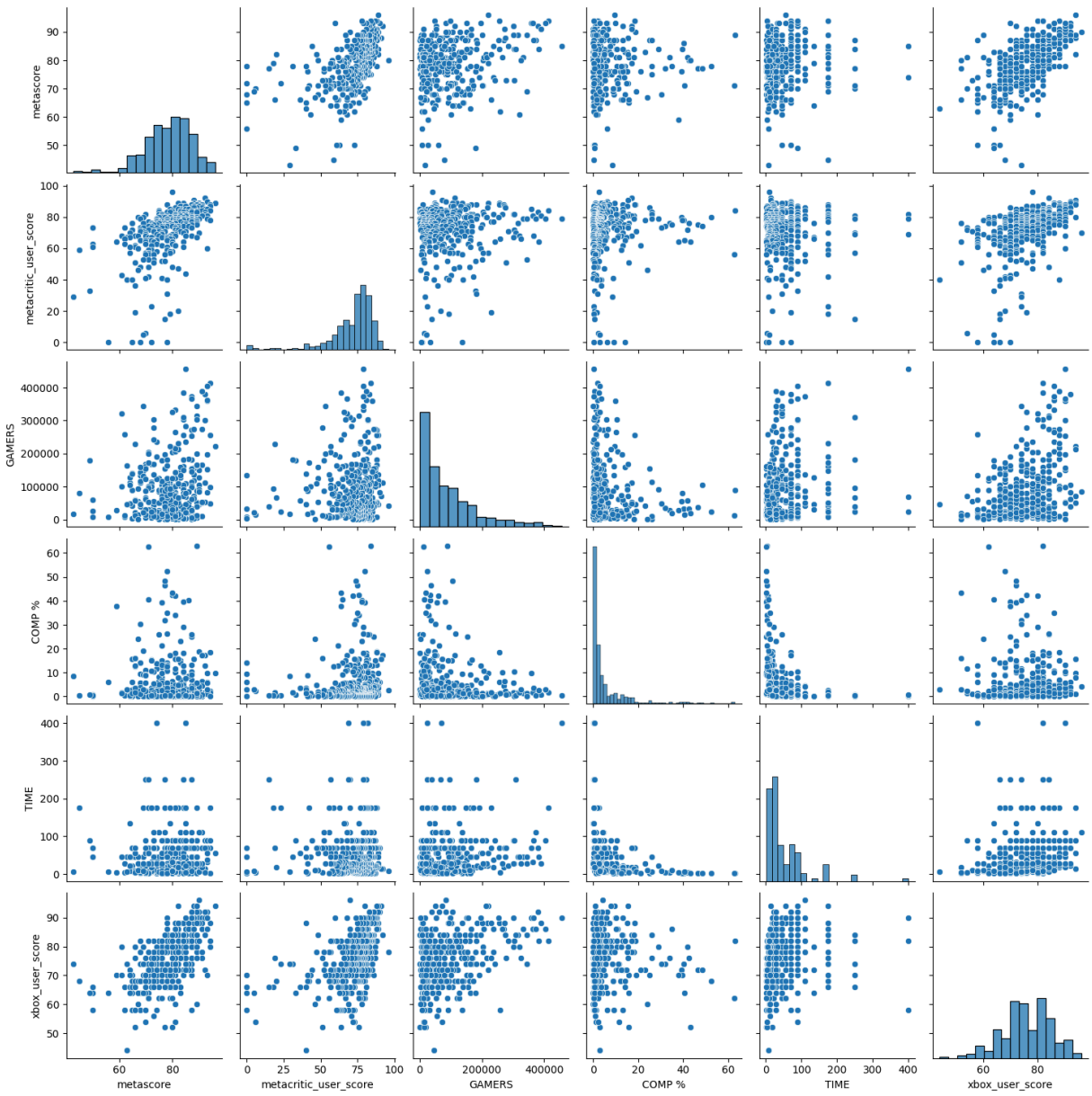
Empezamos con la matriz de correlación.

In [26]:
```
correlation_matrix = df_merged[numeric_columns].corr()

plt.figure(figsize=(10, 8))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', vmin=-1, vmax=1)
plt.title('Matriz de correlación')
plt.show()
```



Visualizamos gráficos de dispersión para cada par de variables numéricas.

In [27]:
```
sns.pairplot(df_merged[numeric_columns])
plt.show()
```



Del análisis bivariante, se puede concluir lo siguiente:

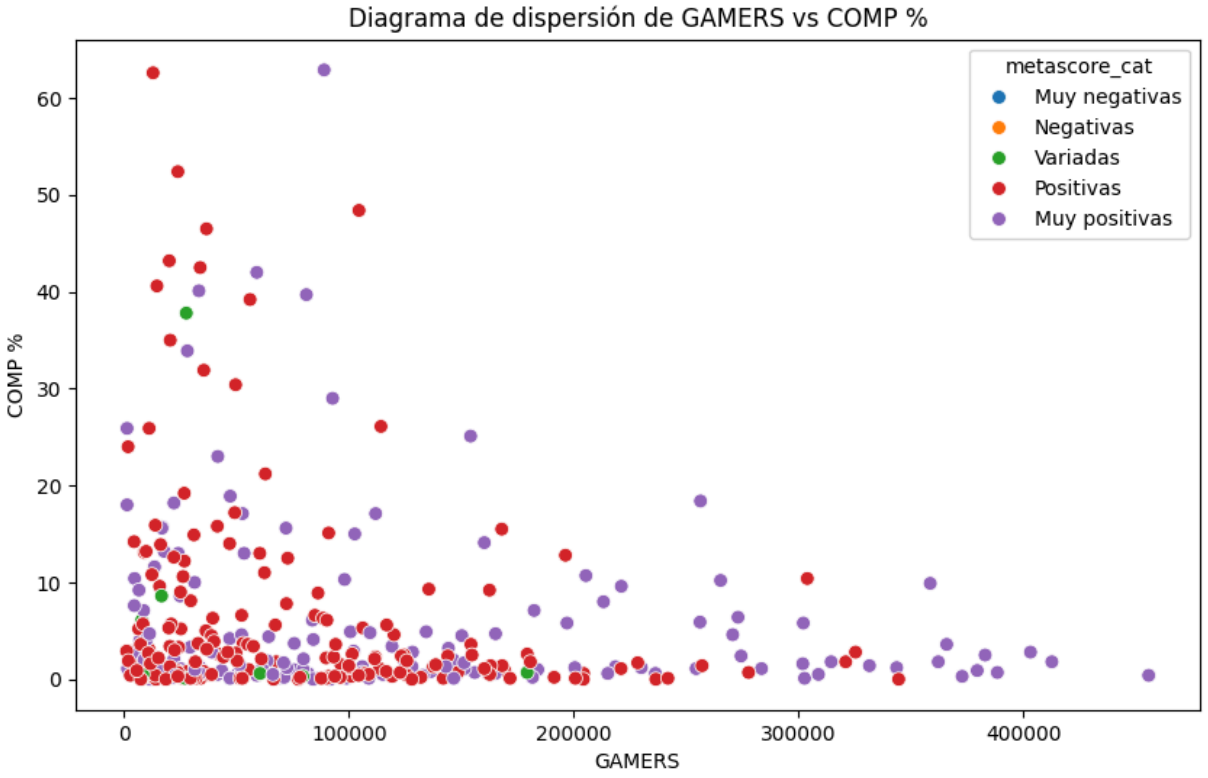
- No hay alguna correlación significativa entre las variables numéricas.
- Al hacer las variables de valoración a tipo categóricas, se puede observar tanto en los histogramas como en los boxplot que las valoraciones de los usuarios de metacritic son distintos a los de los usuarios de Xbox y críticos de Metacritic. Esto nos podría suponer que la opinión de usuarios de Metacritic no es muy confiable. Además, en un acercamiento a los registros con la categoría "Muy negativo" casi la mitad no tenían calificación así que se puede suponer que los usuarios no se han tomado el tiempo para poder calificar los títulos.
- Los títulos con valoraciones "Positivas" tienen grandes números de jugadores, porcentajes de jugadores que lo han completado y tiempo medio para completar. Entonces podemos tener una afirmación de que los jugadores aprovechan el servicio de Game Pass para jugar títulos muy bien valorados.

Análisis Multivariante

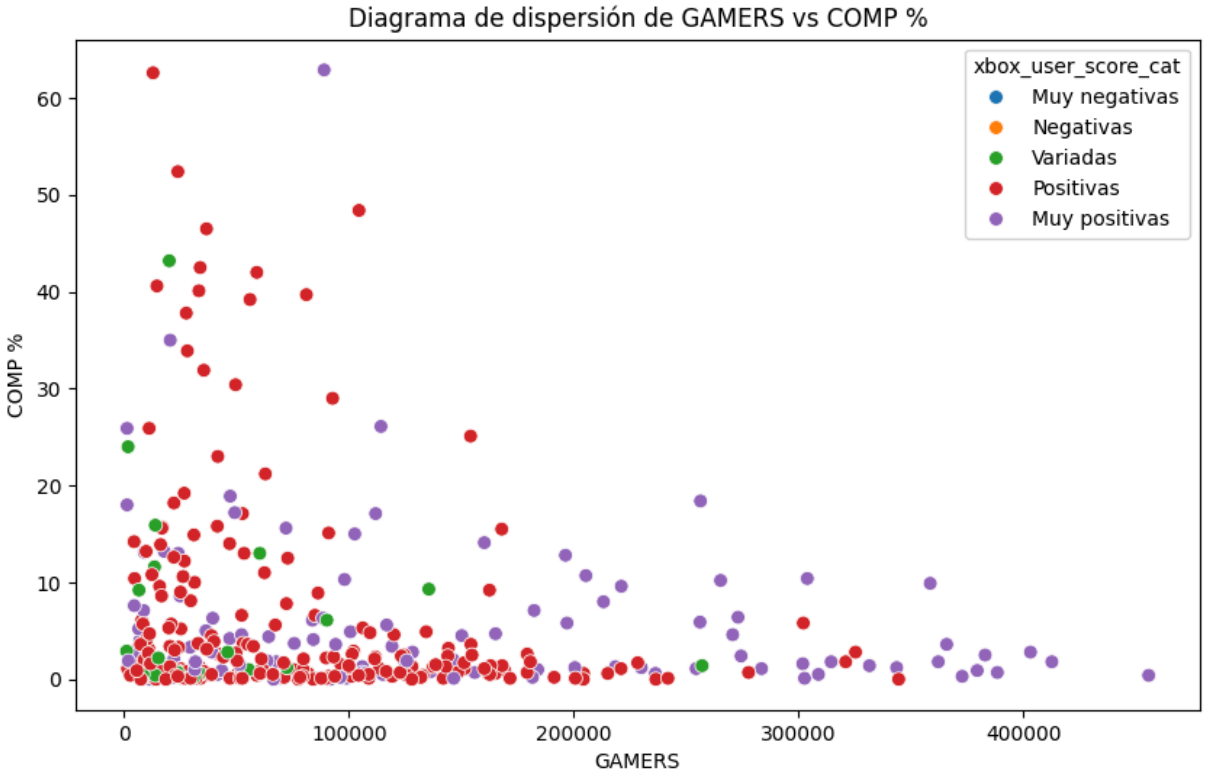
En base a los análisis previos, se hará analisis multivariante entre número de jugadores, porcentaje de completado y tiempo de cada título con cada valoración categórica.

Número de jugadores vs porcentaje de jugadores que han completado el título

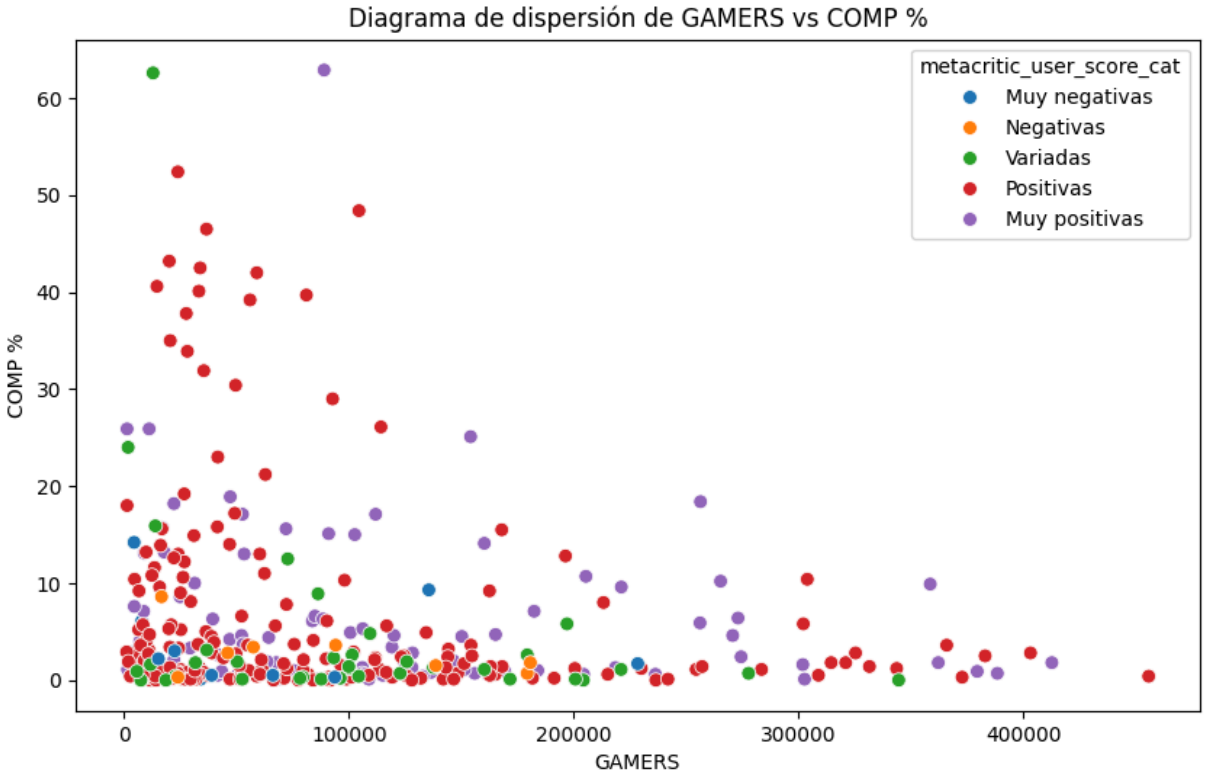
In [28]:
```
funtils.plot_scatter(df_merged, cons.GAMERS, cons.COMP, cons.METASCORE_CAT)
```



In [29]:
```
funtils.plot_scatter(df_merged, cons.GAMERS, cons.COMP, cons.XBOX_USER_SCORE_CAT)
```

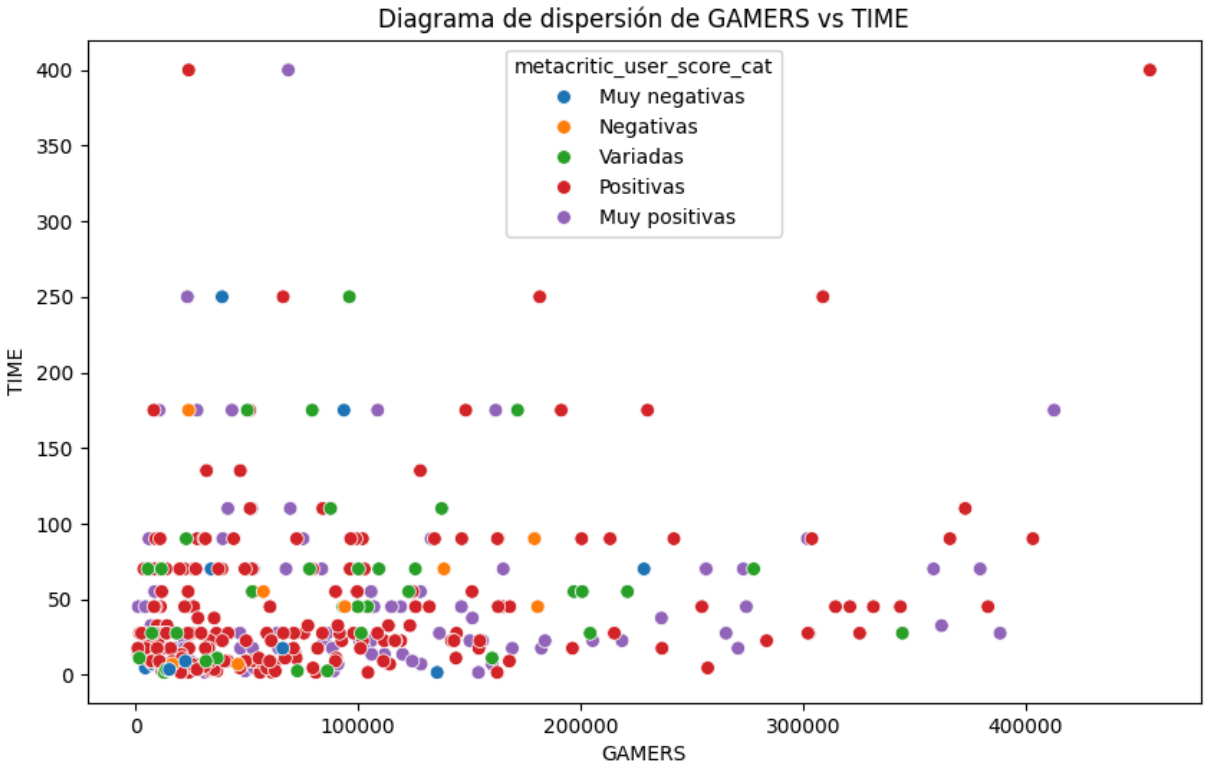


In [30]:
```
funtils.plot_scatter(df_merged, cons.GAMERS, cons.COMP, cons.METACRITIC_USER_SCORE_
```

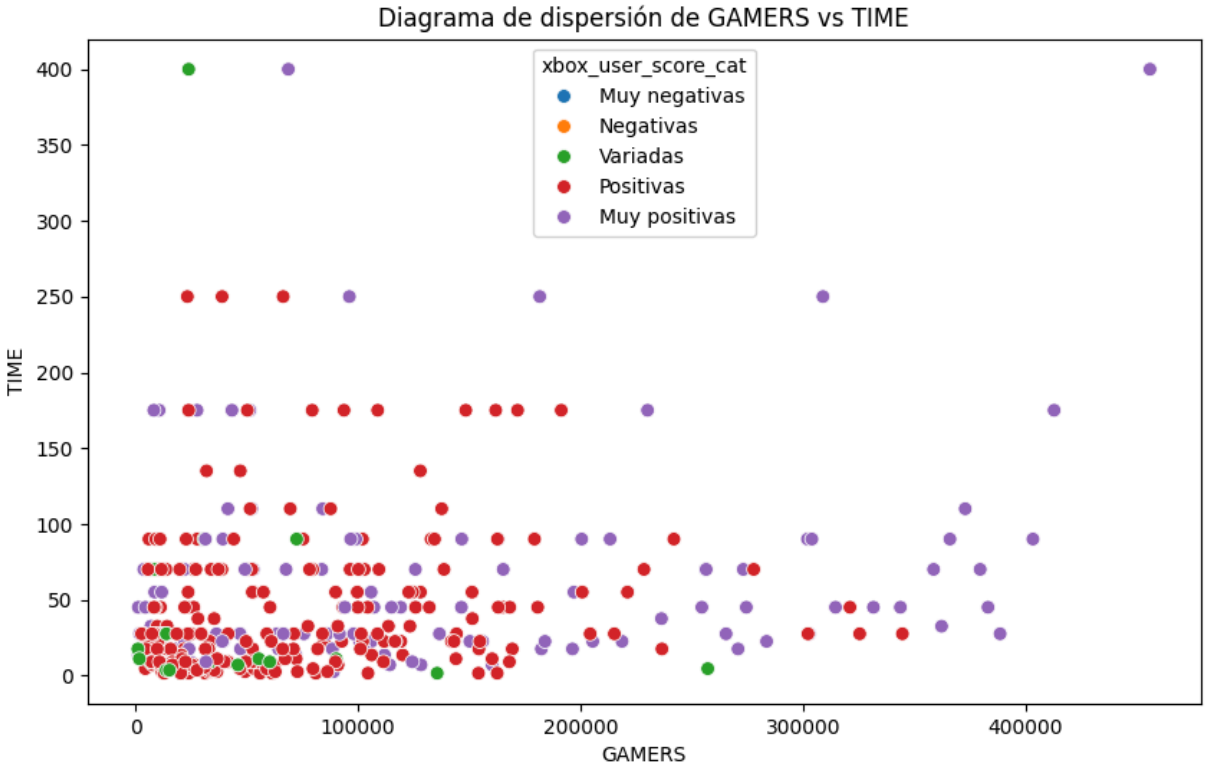


Número de jugadores vs tiempo medio para completar el título

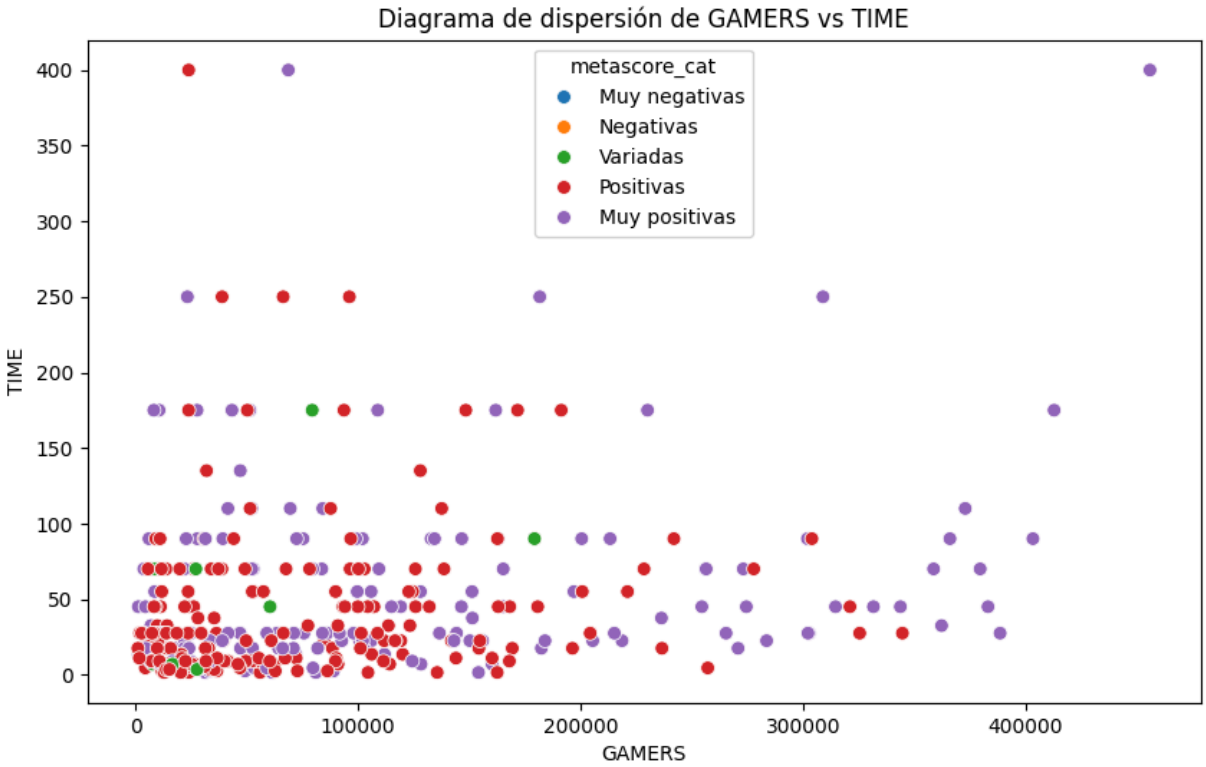
In [31]:
```
funtils.plot_scatter(df_merged, cons.GAMERS, cons.TIME, cons.METACRITIC_USER_SCORE_
```



In [32]:
```
funtils.plot_scatter(df_merged, cons.GAMERS, cons.TIME, cons.XBOX_USER_SCORE_CAT)
```

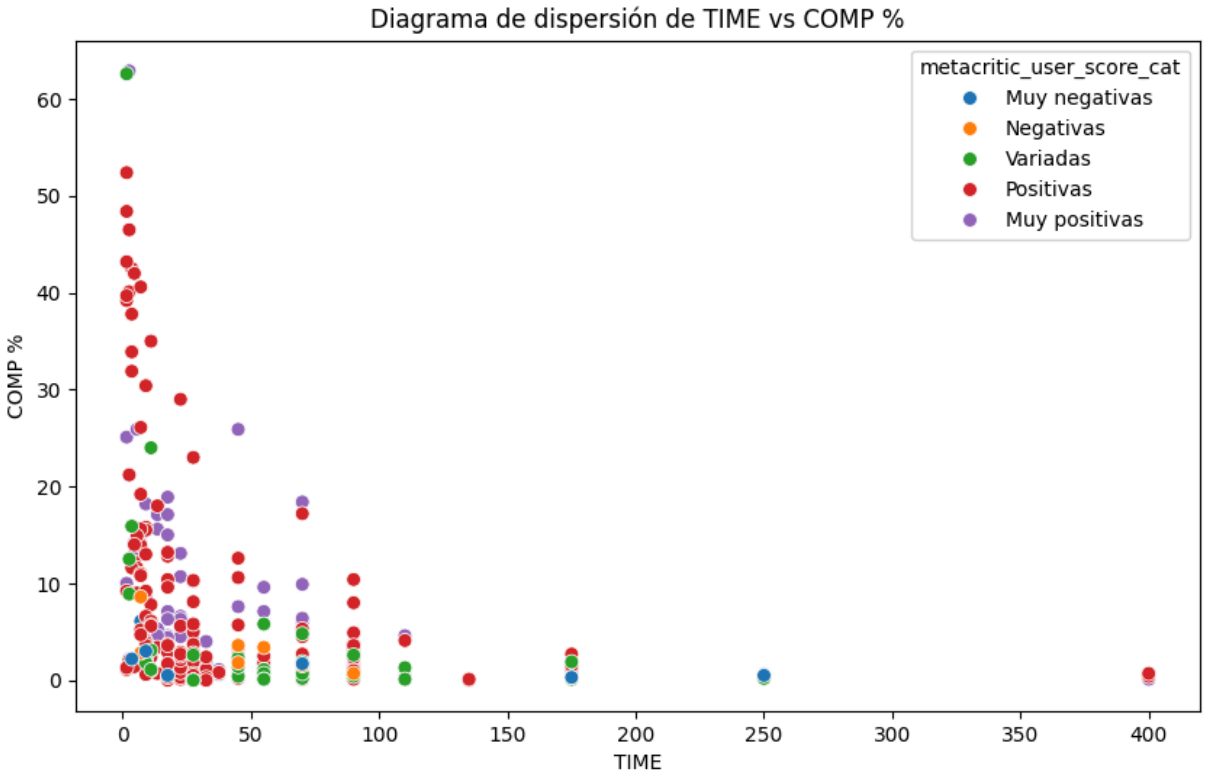


In [33]:
```
funtils.plot_scatter(df_merged, cons.GAMERS, cons.TIME, cons.METASCORE_CAT)
```

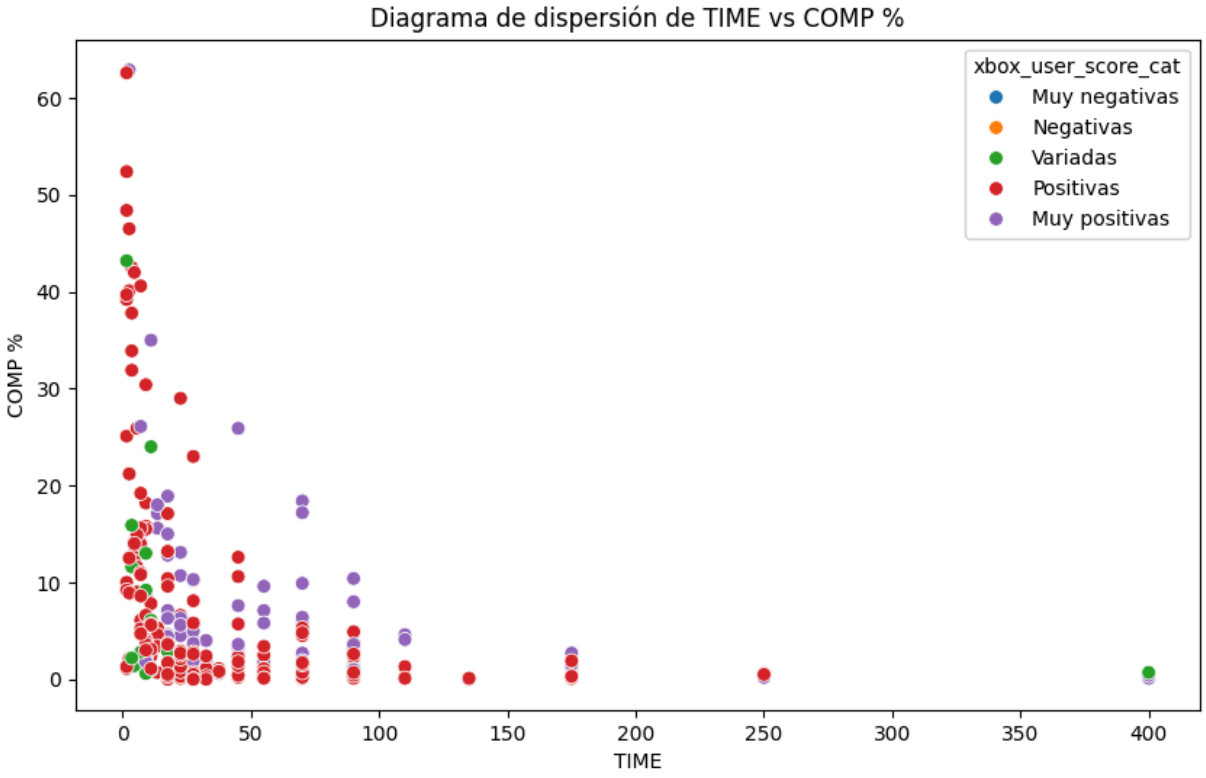


Tiempo medio para completar el título vs porcentaje para completarlo

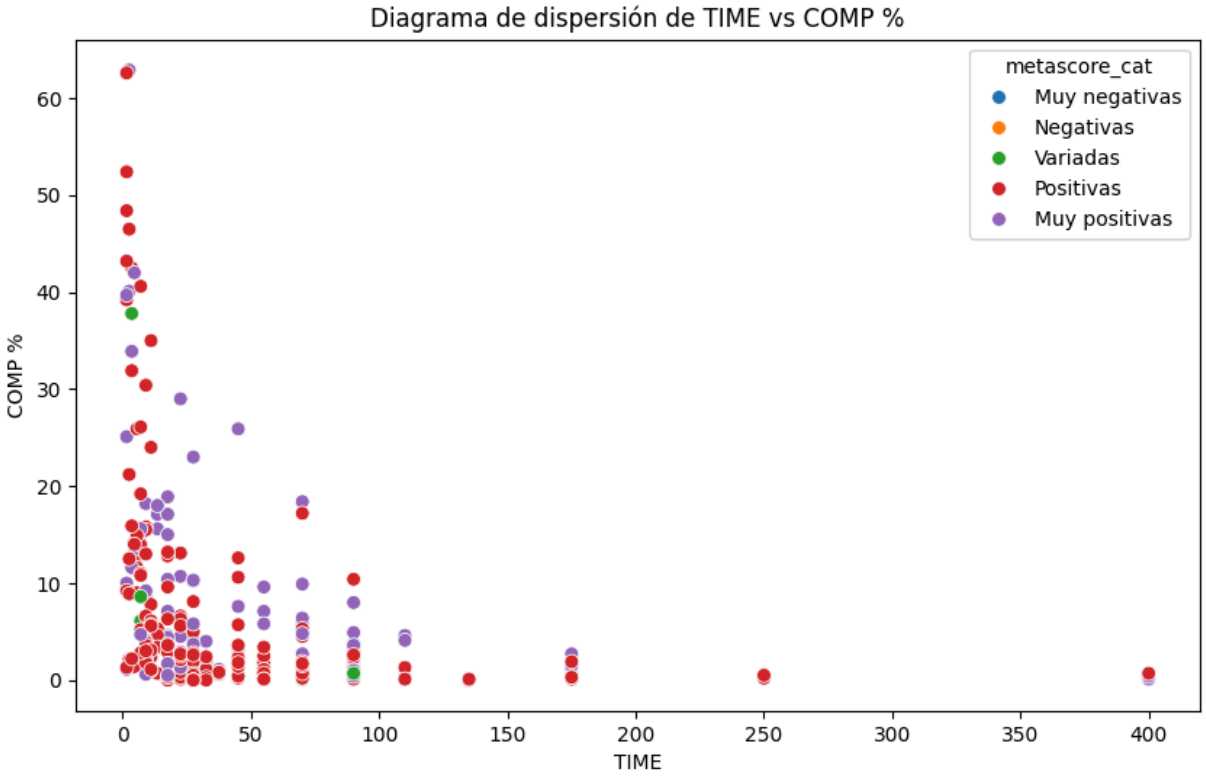
In [34]:
```
funtils.plot_scatter(df_merged,  cons.TIME, cons.COMP, cons.METACRITIC_USER_SCORE_C
```



In [35]:
```
funtils.plot_scatter(df_merged,  cons.TIME, cons.COMP, cons.XBOX_USER_SCORE_CAT)
```

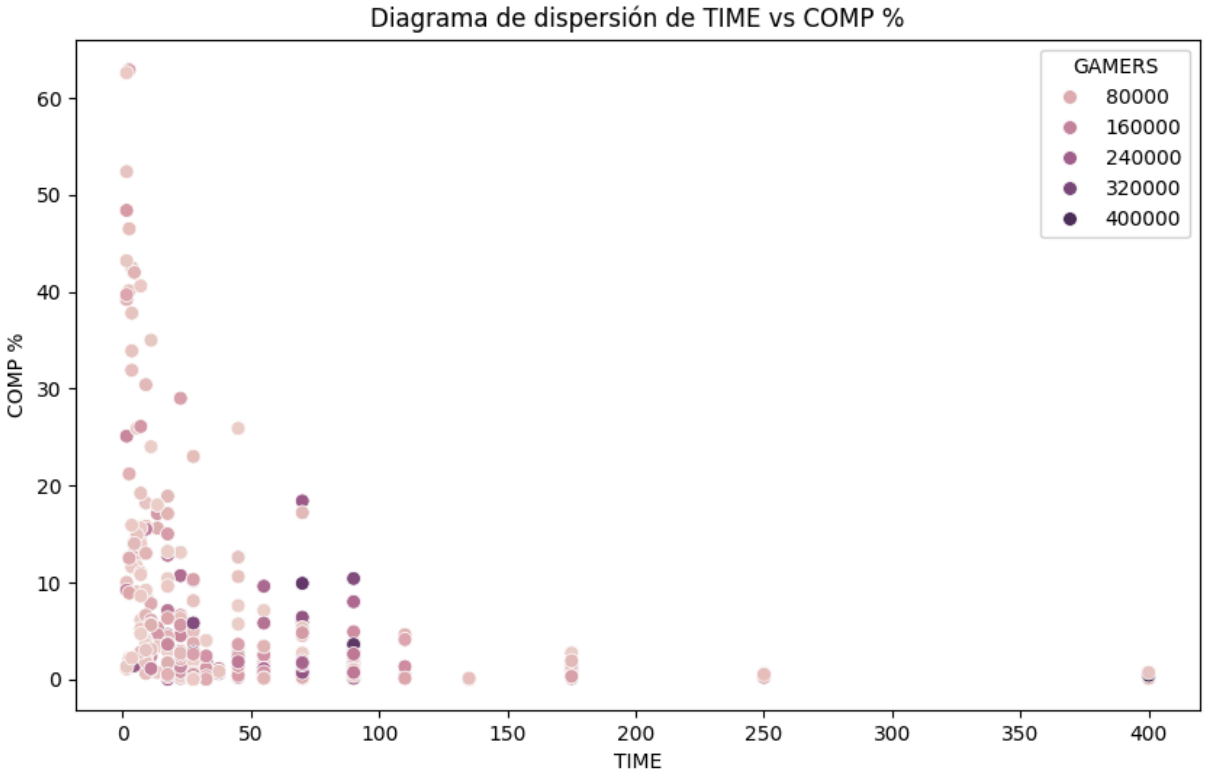


In [36]:
```
funtils.plot_scatter(df_merged,  cons.TIME, cons.COMP, cons.METASCORE_CAT)
```



Para tener el análisis completo, se hace el análisis entre las tres variables numéricas

In [37]:
```
funtils.plot_scatter(df_merged, cons.TIME, cons.COMP, cons.GAMERS)
```



Del análisis multivariante, se concluye:

- La valoración de usuarios de Xbox y críticos de Metacritic influye en el número de jugadores que juegan cada título, así como en el porcentaje de completado. Sin embargo, no necesariamentte significa que vayan a completar los títulos de larga duración.
- Los jugadores prefieren juegos que duren menos de 100 horas y eso influye en si los jugadores terminan o se acercan a completar el título.

Conclusiones

En base a los datos obtenidos:

- **Aprovechamiento del servicio:** Se comprueba la hipótesis de que los jugadores con la suscripción de Xbox Game Pass sí aprovechan el servicio para jugar títulos muy populares por la comunidad aunque no necesariamente los terminan en un 100%.
- **Relación tiempo y valoración:** Si bien hay buenos juegos con larga duración, los jugadores prefieren juegos con una duración media de 30 horas.
- **Finalización de cada título:** No todos los juegos son completados al 100% por los jugadores aunque sean títulos con altas valoraciones. Probablemente por el tiempo medio para completarlo, se enfocan en las historias principales (en el caso de juegos offline) o solo quieren jugar un rato.
- **Valoraciones y número de jugadores:** Las valoraciones de los usuarios de Xbox y los críticos de Metacritic influyen en el número de jugadores que juegan cada título. Sin

embargo, no necesariamente significa que vayan a completar los títulos de larga duración.
- **Duración del juego:** Los jugadores prefieren juegos que duren menos de 100 horas, lo cual influye en si los jugadores terminan o se acercan a completar el título.

Este análisis proporciona una visión clara sobre el uso y la rentabilidad del servicio Xbox Game Pass, ayudando a entender mejor el comportamiento de los jugadores y la relación entre la popularidad de los juegos y su calidad percibida.