In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import sweetviz as sv
import warnings
```

In [2]:
```
warnings.filterwarnings('ignore')
```

In [3]:
```
twitch_df_X = pd.read_pickle('twitch_df_wrng.pkl')
```

In [4]:
```
twitch_df_X.head()
```

Out[4]:

	Game	Month	Year	Hours_watched	Hours_streamed	Peak_viewers	Peak_channels	Strea
0	7 Days to Die	1	2016	269681	12131	4405	44	
1	Agar.io	1	2016	255617	20705	4183	74	
2	Age of Empires	1	2016	248884	232	107455	18	
3	Alien: Isolation	1	2016	264294	11799	9590	42	
4	American Truck Simulator	1	2016	314055	724	43089	48	

In [5]:
```
twitch_df = pd.read_pickle('twitch_df_og.pkl')
```

In [6]:
```
twitch_df.head()
```

Out[6]:

	Game	Month	Year	Hours_watched	Hours_streamed	Peak_viewers	Peak_channels	St
12	7 Days to Die	1	2016	269681	12131	4405	44	
258	Agar.io	1	2016	255617	20705	4183	74	
264	Age of Empires	1	2016	248884	232	107455	18	
422	Alien: Isolation	1	2016	264294	11799	9590	42	
477	American Truck Simulator	1	2016	314055	724	43089	48	

In [7]:
```
twitch_df_X[(twitch_df_X['Hours_watched_1mth'] == 0) & (twitch_df_X['Hours_wat
```

Out[7]:

	Game	Month	Year	Hours_watched	Hours_streamed	Peak_viewers	Peak_channe
94	Metroid Prime	1	2016	248704	1136	150677	
151	Super Mario Bros. 3	1	2016	426084	1161	206252	
155	Super Mario World	1	2016	270879	2888	150670	
156	Super Metroid	1	2016	576156	3905	191257	
166	The Elder Scrolls IV: Oblivion	1	2016	195502	1723	9447	
203	ArcheAge	1	2017	246054	14811	3868	

In [8]:
```
twitch_df_X[(twitch_df_X['Hours_watched_1mth'] == 0) & (twitch_df_X['Hours_wat
```

Out[8]:

	Month	Year	Hours_watched	Hours_streamed	Peak_viewers	Peak_channels
count	35.000000	35.000000	3.500000e+01	35.000000	35.000000	35.000000
mean	2.171429	2018.114286	4.944313e+05	13915.885714	55058.314286	107.285714
std	1.962677	1.966954	4.452449e+05	14098.685022	61231.169945	157.518920
min	1.000000	2016.000000	1.041330e+05	846.000000	2614.000000	7.000000
25%	1.000000	2017.000000	2.190645e+05	3521.000000	7725.500000	24.500000
50%	1.000000	2017.000000	3.448180e+05	9461.000000	27536.000000	57.000000
75%	2.000000	2019.500000	5.212395e+05	18359.500000	74112.500000	87.500000
max	8.000000	2022.000000	2.224158e+06	58877.000000	191257.000000	691.000000

In [9]:
```
report = sv.analyze(twitch_df_X)
report.show_notebook(  w=None,
                h=None,
                scale=None,
                layout='widescreen',
                filepath=None)
```

```
                                |          | [  0%]   00:00 ->
(? left)
```

In [10]:
```
report = sv.analyze(twitch_df)
report.show_notebook(  w=None,
                h=None,
                scale=None,
                layout='widescreen',
                filepath=None)
```

```
                                             |          | [  0%]   00:00 ->
(? left)
```

In [11]:
```
'''This library is actually kind of amazing. I found this when I was strugglin
A big thing is that the first time a game breaks into the top 200 is typically
01-01-2016, no surprise there as that is when the dataset starts by definition
However, second was 01-01-2017, third was 01-01-2018 etc.... Almost half of th
and 01-01-2016 only accounts for 10%. Do these games remain popular in past Ja
```

Out[11]:
```
'This library is actually kind of amazing. I found this when I was struggling
to get ydata_profiling to work properly. \nA big thing is that the first time
a game breaks into the top 200 is typically January. The most common date was
\n01-01-2016, no surprise there as that is when the dataset starts by definit
ion that has the max value of 200. \nHowever, second was 01-01-2017, third wa
s 01-01-2018 etc.... Almost half of the games break into the top 200 in Janua
ry \nand 01-01-2016 only accounts for 10%. Do these games remain popular in p
ast January or are these more likely to be fads?'
```

In [12]:

```
jan_df= twitch_df_X[twitch_df_X['Month'] == 1]
jan_df.head(10)
```

Out[12]:

	Game	Month	Year	Hours_watched	Hours_streamed	Peak_viewers	Peak_channels	Str
0	7 Days to Die	1	2016	269681	12131	4405	44	
1	Agar.io	1	2016	255617	20705	4183	74	
2	Age of Empires	1	2016	248884	232	107455	18	
3	Alien: Isolation	1	2016	264294	11799	9590	42	
4	American Truck Simulator	1	2016	314055	724	43089	48	
5	Ark: Survival Evolved	1	2016	1951875	93060	19486	241	
6	Arma 3	1	2016	2542838	86219	32132	275	
7	Azure Striker GUNVOLT	1	2016	197178	217	135933	14	
8	Banjo-Kazooie	1	2016	241250	2234	108131	28	
9	BattleBlock Theater	1	2016	332256	2041	152739	19	

In [13]:

```
report = sv.analyze(jan_df)
report.show_notebook(  w=None,
                h=None,
                scale=None,
                layout='widescreen',
                filepath=None)
```

```
                                        |          | [  0%]   00:00 ->
(? left)
```

In [14]:

```
x = len(twitch_df_X)
jan_list = []
for i in range(x):
    if twitch_df_X['Month'][i] == 1:
        jan_list.append(1)
    else:
        jan_list.append(0)
```

In [15]:
```
twitch_df_X['Jan_Debut_Month'] = jan_list
```

In [16]:
```
twitch_df_X.head()
```

Out[16]:

	Game	Month	Year	Hours_watched	Hours_streamed	Peak_viewers	Peak_channels	Strea
0	7 Days to Die	1	2016	269681	12131	4405	44	
1	Agar.io	1	2016	255617	20705	4183	74	
2	Age of Empires	1	2016	248884	232	107455	18	
3	Alien: Isolation	1	2016	264294	11799	9590	42	
4	American Truck Simulator	1	2016	314055	724	43089	48	

In [17]:
```
report = sv.analyze(twitch_df_X)
report.show_notebook(  w=None,
                h=None,
                scale=None,
                layout='widescreen',
                filepath=None)
```

```
                                     |          | [  0%]   00:00 ->
(? left)
```

In [18]:
```
'''Based on the EDA in this dataset it seems that if a game debuted in January
Let me remove 2016 and see if the correlation still holds
Also of note there is a .95 correlation bewteen if a game is popular in the fi
thus a prediction on hours watched 6 months from debut is not required'''
```

Out[18]:
```
'Based on the EDA in this dataset it seems that if a game debuted in January
it will be popular one month later. \nLet me remove 2016 and see if the corre
lation still holds\nAlso of note there is a .95 correlation bewteen if a game
is popular in the first 3 months (hours watched) and 6 months,\nthus a predic
tion on hours watched 6 months from debut is not required'
```

In [19]:

```
no_2016_df = twitch_df_X[twitch_df_X['Year']!= 2016]
no_2016_df.head()
```

Out[19]:

	Game	Month	Year	Hours_watched	Hours_streamed	Peak_viewers	Peak_channels
200	ARK: Survival Evolved	1	2017	2167646	192501	18756	483
201	ASTRONEER	1	2017	761112	21225	29721	72
202	Age of Empires II	1	2017	310965	5299	4129	16
203	ArcheAge	1	2017	246054	14811	3868	43
204	Assassin's Creed II	1	2017	341584	3049	33045	22

In [20]:

```
report = sv.analyze(no_2016_df)
report.show_notebook(  w=None,
                h=None,
                scale=None,
                layout='widescreen',
                filepath=None)
```

```
                                                    |          | [  0%]   00:00 ->
(? left)
```

In [21]:

```
'''the correlation still holds, this may be a useful feature, and a surprising
```

Out[21]:

```
'the correlation still holds, this may be a useful feature, and a surprising
one at that'
```

In [22]:

```
x = len(twitch_df_X)
top200_list = []
for i in range(x):
    if twitch_df_X['Hours_watched_1mth'][i] != 0:
        top200_list.append(1)
    else:
        top200_list.append(0)
```

In [23]:

```
twitch_df_X['Next_mth_200'] = top200_list
```

In [24]:
```
twitch_df_X.head()
```

Out[24]:

	Game	Month	Year	Hours_watched	Hours_streamed	Peak_viewers	Peak_channels	Strea
0	7 Days to Die	1	2016	269681	12131	4405	44	
1	Agar.io	1	2016	255617	20705	4183	74	
2	Age of Empires	1	2016	248884	232	107455	18	
3	Alien: Isolation	1	2016	264294	11799	9590	42	
4	American Truck Simulator	1	2016	314055	724	43089	48	

In [25]:
```
report = sv.analyze(twitch_df_X)
report.show_notebook(  w=None,
                h=None,
                scale=None,
                layout='widescreen',
                filepath=None)
```

```
                                        |          | [  0%]   00:00 ->
(? left)
```

In [27]:
```
twitch_df_X.to_pickle('twitch_df_wrng.pkl')
```

In [ ]: