FASE 1: Entendiendo el negocio

Contexto:

La empresa **GameInsights S.A.**, una compañía especializada en el análisis de datos para la industria de los videojuegos, ha sido contratada por Epic Games para ayudar a mejorar la experiencia de juego en Fortnite. El objetivo principal es utilizar los datos de rendimiento de los jugadores en modo solitario para desarrollar un modelo predictivo que pueda identificar a los jugadores con un alto potencial de rendimiento y aquellos que podrían estar en riesgo de abandonar el juego debido a un bajo desempeño.

Problematica:

Epic Games ha observado que la retención de jugadores en Fortnite es crucial para el éxito a largo plazo del juego. Sin embargo, algunos jugadores, especialmente aquellos con un bajo desempeño, tienden a abandonar el juego después de experimentar frustración por no lograr buenos resultados. La empresa quiere crear un sistema que pueda predecir el rendimiento futuro de los jugadores basándose en su historial de juegos en modo solitario.

Objetivo:

Desarrollar un modelo de predicción que, basado en las estadísticas actuales de un jugador (como el `Solo kd`, `Solo winRatio`, `Solo matches` y otras métricas), pueda predecir:

1. **El nivel de desempeño futuro** del jugador en términos de victorias y ratio de kills/deaths (KD).
2. **El riesgo de abandono** del jugador, definido como una disminución significativa en el número de partidas jugadas o un cambio abrupto en su rendimiento.

```
# Se Realiza las importanciones de las Librerias y del Dataset
import pandas as pd
import numpy as np
import seaborn as sb
import matplotlib.pyplot as plt
from sklearn.impute import KNNImputer
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import MinMaxScaler

#cargar anexo

df_completo = pd.read_csv('Fortnite_players_stats.csv', sep=",")

print(df_completo.columns)


Index(['Player', 'Solo score', 'Solo top1', 'Solo kd', 'Solo
winRatio',
```

```
       'Solo matches', 'Solo kills', 'Solo minutesPlayed', 'Duos
score',
       'Duos top1', 'Duos kd', 'Duos winRatio', 'Duos matches', 'Duos
kills',
       'Duos minutesPlayed', 'Trios score', 'Trios top1', 'Trios kd',
       'Trios winRatio', 'Trios matches', 'Trios kills', 'Trios
minutesPlayed',
       'Squads score', 'Squads top1', 'Squads kd', 'Squads winRatio',
       'Squads matches', 'Squads kills', 'Squads minutesPlayed', 'LTM
score',
       'LTM top1', 'LTM top3', 'LTM kd', 'LTM winRatio', 'LTM
matches',
       'LTM kills', 'LTM minutesPlayed'],
      dtype='object')

df = df_completo[['Player', 'Solo score', 'Solo top1', 'Solo kd',
                            'Solo winRatio', 'Solo matches', 'Solo
kills', 'Solo minutesPlayed']]
```

FASE 2: Data Understanding

Verificar columnas y tipos de datos

```
columnas = df.columns
for columna in columnas:
  if df[columna].dtype == int:
    print(f"Columna {columna} ------ Tipo de variable: Cuantitativa
discreta")
  elif df[columna].dtype == float:
    print(f"Columna {columna} ------ Tipo de variable: Cuantitativa
continua")
  elif df[columna].dtype == object:
    print(f"Columna {columna} ------ Tipo de variable: Cualitativa
nominal")
  elif df[columna].dtype == bool:
    print(f"Columna {columna} ------ Tipo de variable: Cualitativa
ordinal")

Columna Player ------ Tipo de variable: Cualitativa nominal
Columna Solo score ------ Tipo de variable: Cuantitativa discreta
Columna Solo top1 ------ Tipo de variable: Cuantitativa discreta
Columna Solo kd ------ Tipo de variable: Cuantitativa continua
Columna Solo winRatio ------ Tipo de variable: Cuantitativa continua
Columna Solo matches ------ Tipo de variable: Cuantitativa discreta
Columna Solo kills ------ Tipo de variable: Cuantitativa discreta
```

```
Columna Solo minutesPlayed ------ Tipo de variable: Cuantitativa discreta

# Se filtran las 10 primeras Filas
df.head(10)
```


```
{"column_count":8,"columns":
[{"dtype":"object","name":"Player","stats":{"categories":
[{"count":1,"name":"Prospering"},{"count":1,"name":"BH nixxxay"},
{"count":8,"name":"8 others"}],"nan_count":0,"unique_count":10}},
{"dtype":"int64","name":"Solo score","stats":{"histogram":
[{"bin_end":691496.4,"bin_start":15872,"count":3},
{"bin_end":1367120.8,"bin_start":691496.4,"count":2},
{"bin_end":2042745.2000000002,"bin_start":1367120.8,"count":0},
{"bin_end":2718369.6,"bin_start":2042745.2000000002,"count":2},
{"bin_end":3393994,"bin_start":2718369.6,"count":1},
{"bin_end":4069618.4000000004,"bin_start":3393994,"count":0},
{"bin_end":4745242.8,"bin_start":4069618.4000000004,"count":1},
{"bin_end":5420867.2,"bin_start":4745242.8,"count":0},
{"bin_end":6096491.600000001,"bin_start":5420867.2,"count":0},
{"bin_end":6772116,"bin_start":6096491.600000001,"count":1}],"max":"67
72116","min":"15872","nan_count":0,"unique_count":10}},
{"dtype":"int64","name":"Solo top1","stats":{"histogram":
[{"bin_end":653.4,"bin_start":6,"count":3},
{"bin_end":1300.8,"bin_start":653.4,"count":0},
{"bin_end":1948.1999999999998,"bin_start":1300.8,"count":5},
{"bin_end":2595.6,"bin_start":1948.1999999999998,"count":0},
{"bin_end":3243,"bin_start":2595.6,"count":0},
{"bin_end":3890.3999999999996,"bin_start":3243,"count":0},
{"bin_end":4537.8,"bin_start":3890.3999999999996,"count":0},
{"bin_end":5185.2,"bin_start":4537.8,"count":1},
{"bin_end":5832.599999999999,"bin_start":5185.2,"count":0},
{"bin_end":6480,"bin_start":5832.599999999999,"count":1}],"max":"6480"
,"min":"6","nan_count":0,"unique_count":10}},
{"dtype":"float64","name":"Solo kd","stats":{"histogram":
[{"bin_end":1.383,"bin_start":0.47,"count":3},
{"bin_end":2.296,"bin_start":1.383,"count":1},
{"bin_end":3.2089999999999996,"bin_start":2.296,"count":0},
{"bin_end":4.122,"bin_start":3.2089999999999996,"count":2},
{"bin_end":5.034999999999999,"bin_start":4.122,"count":1},
{"bin_end":5.9479999999999995,"bin_start":5.034999999999999,"count":0}
,
{"bin_end":6.860999999999999,"bin_start":5.9479999999999995,"count":1}
,
{"bin_end":7.773999999999999,"bin_start":6.860999999999999,"count":0},
{"bin_end":8.687,"bin_start":7.773999999999999,"count":0},
{"bin_end":9.6,"bin_start":8.687,"count":2}],"max":"9.6","min":"0.47",
"nan_count":0,"unique_count":10}},{"dtype":"float64","name":"Solo
winRatio","stats":{"histogram":
[{"bin_end":3.85,"bin_start":0.5,"count":3},
```

top1":1828,"Solo winRatio":18,"_deepnote_index_column":0},
{"Player":"BH nixxxay","Solo kd":8.71,"Solo kills":35895,"Solo
matches":5817,"Solo minutesPlayed":12732,"Solo score":439562,"Solo
top1":1694,"Solo winRatio":29.1,"_deepnote_index_column":1},
{"Player":"Ranger","Solo kd":9.6,"Solo kills":85481,"Solo
matches":13488,"Solo minutesPlayed":122171,"Solo score":4519465,"Solo
top1":4582,"Solo winRatio":34,"_deepnote_index_column":2},
{"Player":"Idk_Pi","Solo kd":0.84,"Solo kills":3005,"Solo
matches":3687,"Solo minutesPlayed":32453,"Solo score":752869,"Solo
top1":121,"Solo winRatio":3.3,"_deepnote_index_column":3},
{"Player":"Twitch.GryphonRB","Solo kd":6.32,"Solo kills":19591,"Solo
matches":4429,"Solo minutesPlayed":36245,"Solo score":1136282,"Solo
top1":1327,"Solo winRatio":30,"_deepnote_index_column":4},
{"Player":"Twitch Kayotica","Solo kd":3.23,"Solo kills":39131,"Solo
matches":13438,"Solo minutesPlayed":96777,"Solo score":2919037,"Solo
top1":1310,"Solo winRatio":9.7,"_deepnote_index_column":5},
{"Player":"FaZe Replays","Solo kd":3.84,"Solo kills":66161,"Solo
matches":18670,"Solo minutesPlayed":76258,"Solo score":2389537,"Solo
top1":1454,"Solo winRatio":7.8,"_deepnote_index_column":6},
{"Player":"NiteGamerYT 190k","Solo kd":0.47,"Solo kills":200,"Solo
matches":429,"Solo minutesPlayed":739,"Solo score":15872,"Solo
top1":6,"Solo winRatio":1.4,"_deepnote_index_column":7},
{"Player":"Bossン","Solo kd":1.39,"Solo kills":18610,"Solo
matches":19864,"Solo minutesPlayed":283835,"Solo score":6772116,"Solo
top1":6480,"Solo winRatio":32.6,"_deepnote_index_column":8},
{"Player":"CIUPEA 144.HZ","Solo kd":0.61,"Solo kills":1174,"Solo
matches":1938,"Solo minutesPlayed":2441,"Solo score":54479,"Solo
top1":9,"Solo winRatio":0.5,"_deepnote_index_column":9}]}

```
#Se Filtra el tipo de dato de la columnas
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1435 entries, 0 to 1434
Data columns (total 8 columns):
 #   Column              Non-Null Count  Dtype  
---  ------              --------------  -----  
 0   Player              1435 non-null   object 
 1   Solo score          1435 non-null   int64  
 2   Solo top1           1435 non-null   int64  
 3   Solo kd             1435 non-null   float64
 4   Solo winRatio       1435 non-null   float64
 5   Solo matches        1435 non-null   int64  
 6   Solo kills          1435 non-null   int64  
 7   Solo minutesPlayed  1435 non-null   int64  
dtypes: float64(2), int64(5), object(1)
memory usage: 89.8+ KB
```

Verificación de datos nulos y posibles errores en DataFrame.

```
# Verificación de datos nulos en el dataset
df.isnull().sum()
```

```
Player                0
Solo score            0
Solo top1             0
Solo kd               0
Solo winRatio         0
Solo matches          0
Solo kills            0
Solo minutesPlayed    0
dtype: int64
```

De momento los datos a utilizar no arrojan datos nulos, pero existe la posibilidad que hayan datos negativos que los interprete como datos no nulos por lo que se verificara con el siguiente código.

```
# Verificar si hay datos negativos
(df[['Solo score', 'Solo top1', 'Solo kd', 'Solo winRatio', 'Solo
matches', 'Solo kills', 'Solo minutesPlayed']] < 0).sum()
```

```
Solo score            0
Solo top1             0
Solo kd               0
Solo winRatio         0
Solo matches          0
Solo kills            0
Solo minutesPlayed    0
dtype: int64
```

Al no existir datos negativos y nulos, estamos listo para comenzar las medidas de posición.

Medidas de posición

```
#Resumen de Medidas de posición, para datos numéricos
print(df.describe())
```

```
         Solo score    Solo top1      Solo kd  Solo winRatio  Solo
matches  \
count  1.435000e+03  1435.000000  1435.000000    1435.000000
1435.000000
mean   1.088202e+06   328.909408     1.816300       4.853937
6911.060627
std    8.872639e+05   529.192568     1.156026       5.330249
5408.219523
min    4.453000e+03     0.000000     0.080000       0.000000
30.000000
25%    4.355170e+05    65.500000     1.050000       1.600000
2875.000000
50%    9.070010e+05   162.000000     1.480000       2.900000
```

```
5728.000000
75%    1.506555e+06   366.000000     2.200000       6.100000
9585.000000
max    6.772116e+06  6480.000000     9.600000      44.100000
36487.000000

         Solo kills  Solo minutesPlayed
count   1435.000000         1435.000000
mean   11781.843206        43517.135889
std    11417.340071        36314.877771
min       28.000000          210.000000
25%     3861.000000        16922.000000
50%     8452.000000        35762.000000
75%    15993.000000        59471.000000
max    89023.000000       283835.000000
```

```python
# Excluir columnas no numéricas
df_numeric = df.select_dtypes(include=[float, int])

# Calcular la matriz de correlación solo con las columnas numéricas
correlation_matrix = df_numeric.corr()

# Mostrar la matriz de correlación
print(correlation_matrix)
```

```
                    Solo score  Solo top1   Solo kd  Solo winRatio  \
Solo score            1.000000   0.598385  0.203172       0.245719
Solo top1             0.598385   1.000000  0.659677       0.768525
Solo kd               0.203172   0.659677  1.000000       0.854501
Solo winRatio         0.245719   0.768525  0.854501       1.000000
Solo matches          0.860244   0.361533  0.036813      -0.022918
Solo kills            0.715041   0.748277  0.613477       0.448650
Solo minutesPlayed    0.979049   0.484908  0.067514       0.133830

                    Solo matches  Solo kills  Solo minutesPlayed
Solo score              0.860244    0.715041            0.979049
Solo top1               0.361533    0.748277            0.484908
Solo kd                 0.036813    0.613477            0.067514
Solo winRatio          -0.022918    0.448650            0.133830
Solo matches            1.000000    0.705988            0.859965
Solo kills              0.705988    1.000000            0.601728
Solo minutesPlayed      0.859965    0.601728            1.000000
```

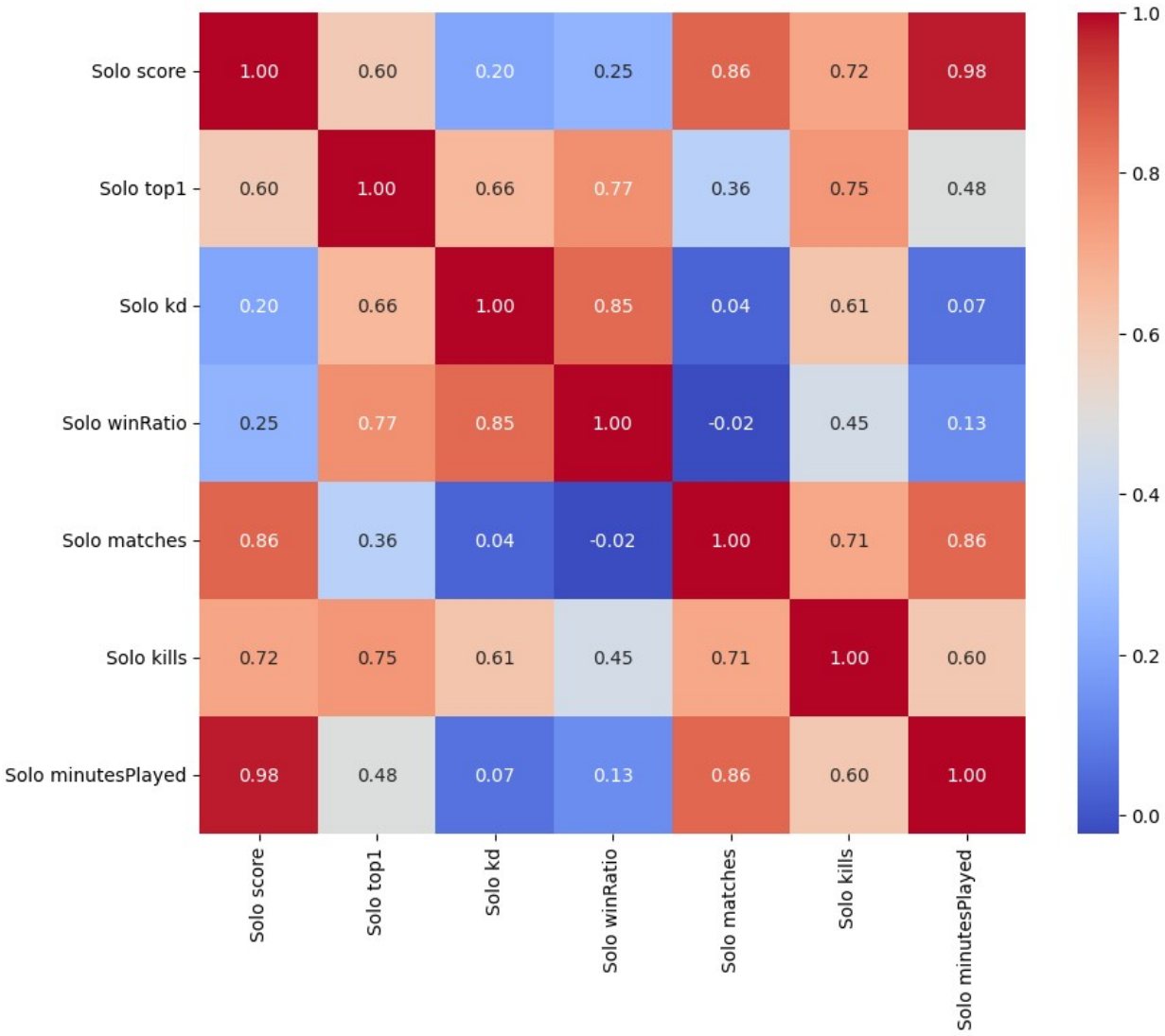
Matriz de correlación

Las variables Solo score, Solo matches y Solo minutesPlayed están fuertemente correlacionadas, podemos decir que el tiempo jugado y el número de partidas tienen un gran impacto en la puntuación total.

Solo winRatio está fuertemente correlacionado con la capacidad del jugador de estar en el Solo top1 y con su habilidad de combate (Solo kd).

No hay correlación significativa entre algunas variables, como Solo matches y Solo kd, lo que puede sugerir que jugar más partidas no mejora la relación de asesinatos/muertes de un jugador.

```
plt.figure(figsize=(10, 8))
sb.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt='.2f')
plt.show()
```

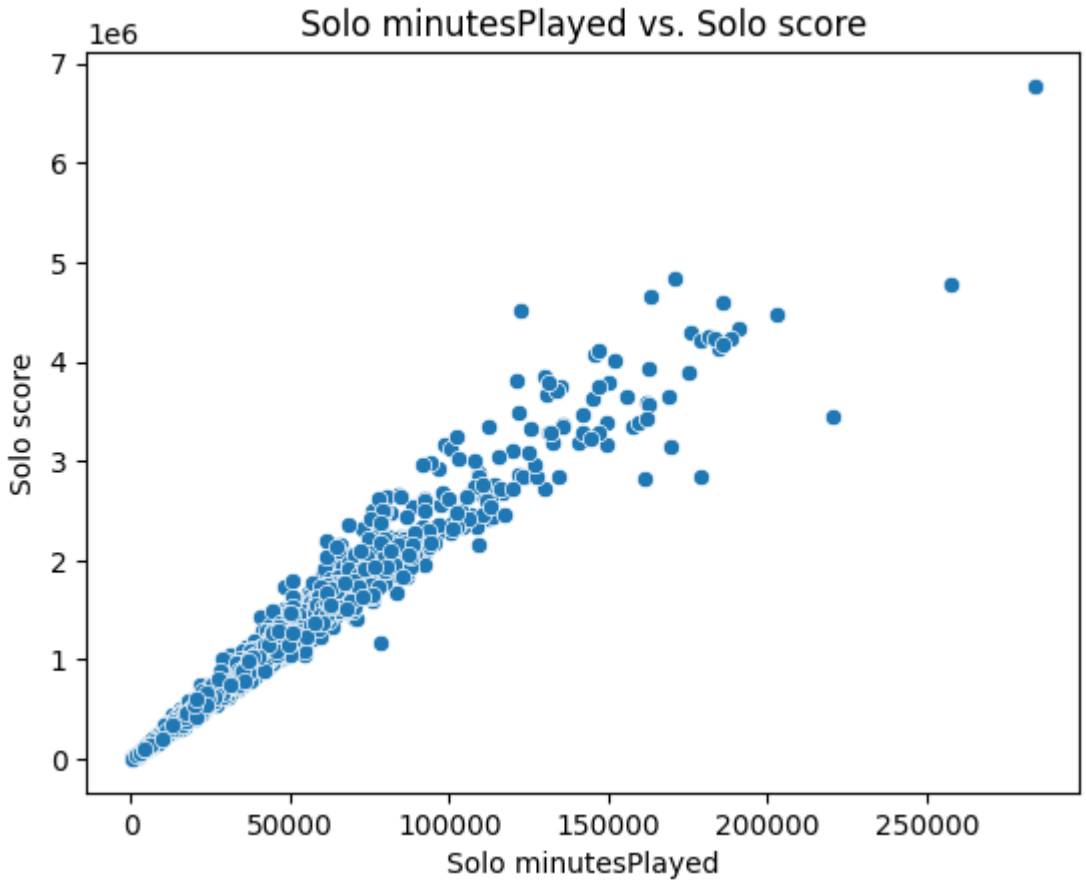


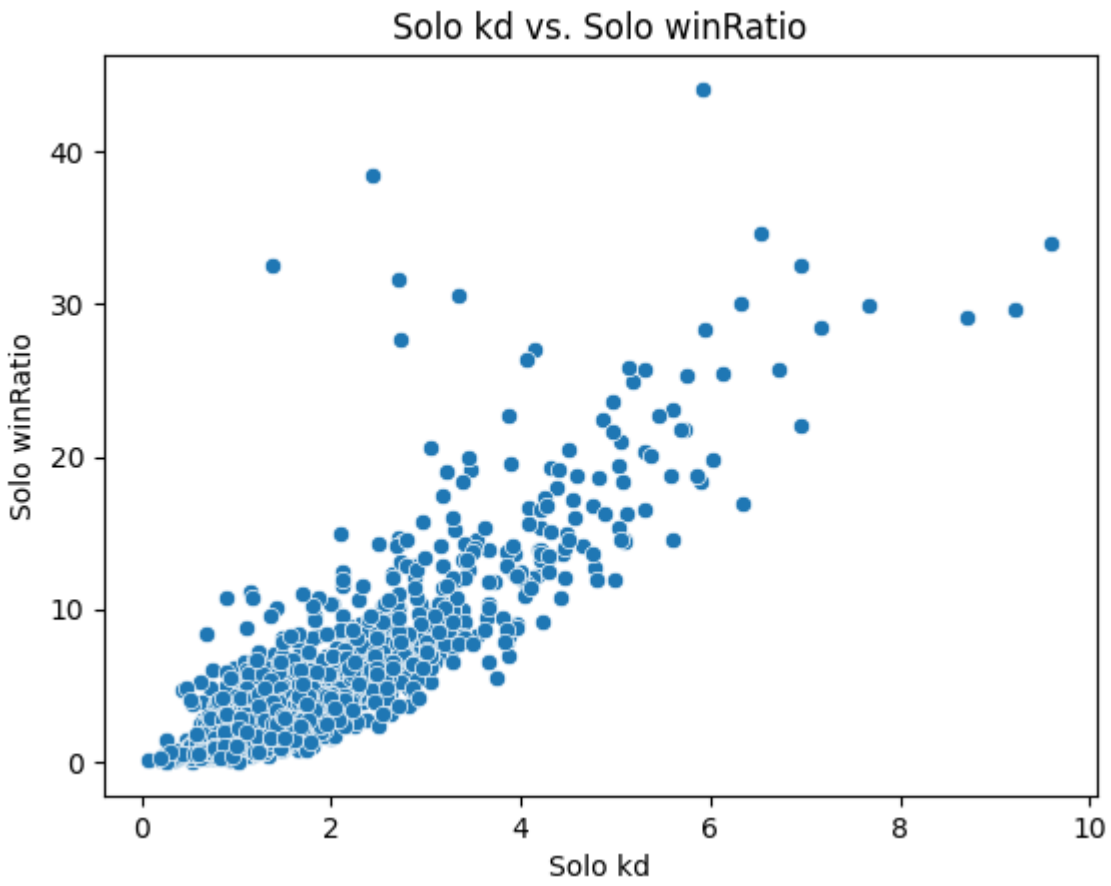
Scatter Plots (Diagramas de Dispersión):

Solo minutesPlayed vs. Solo score: Dado que existe una fuerte correlación entre el tiempo jugado y el puntaje, un scatter plot puede mostrar cómo el puntaje de los jugadores aumenta con el tiempo jugado. Solo kd vs. Solo winRatio: Con una correlación alta entre el ratio KD y el winRatio, un scatter plot puede ayudar a visualizar cómo se relacionan estas dos métricas.

```
sb.scatterplot(x='Solo minutesPlayed', y='Solo score', data=df)
plt.title('Solo minutesPlayed vs. Solo score')
plt.show()

sb.scatterplot(x='Solo kd', y='Solo winRatio', data=df)
plt.title('Solo kd vs. Solo winRatio')
plt.show()
```

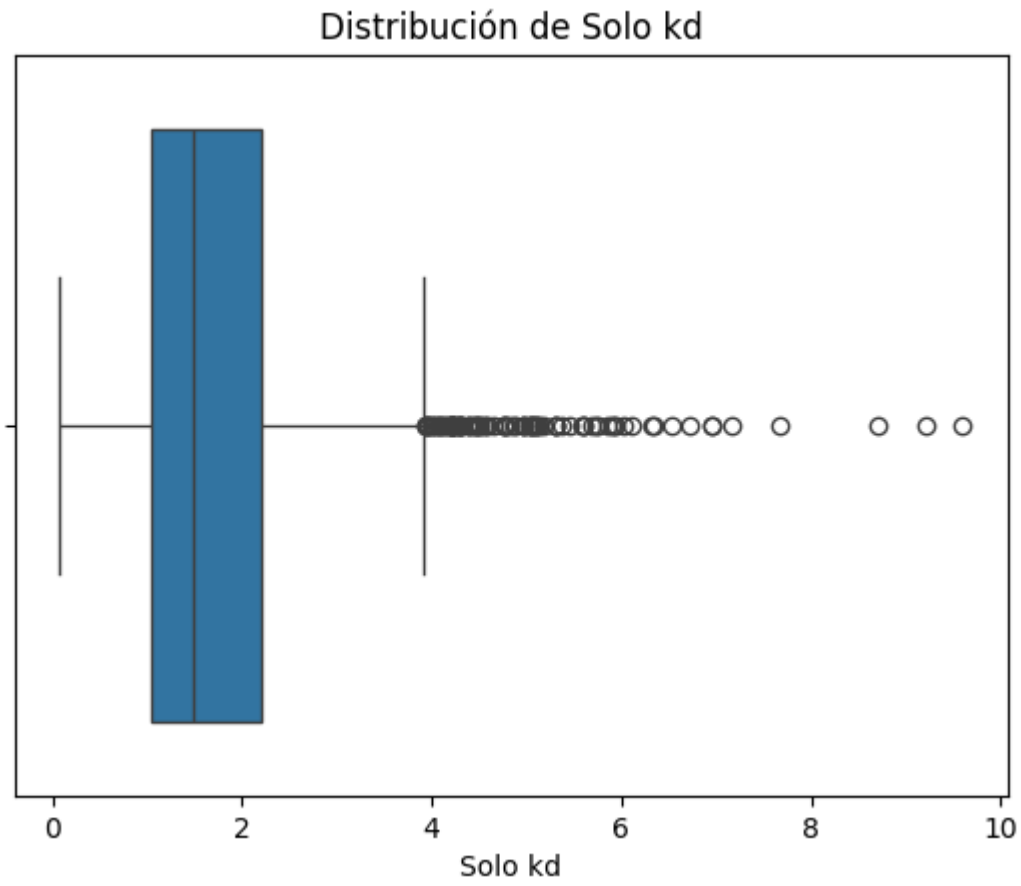


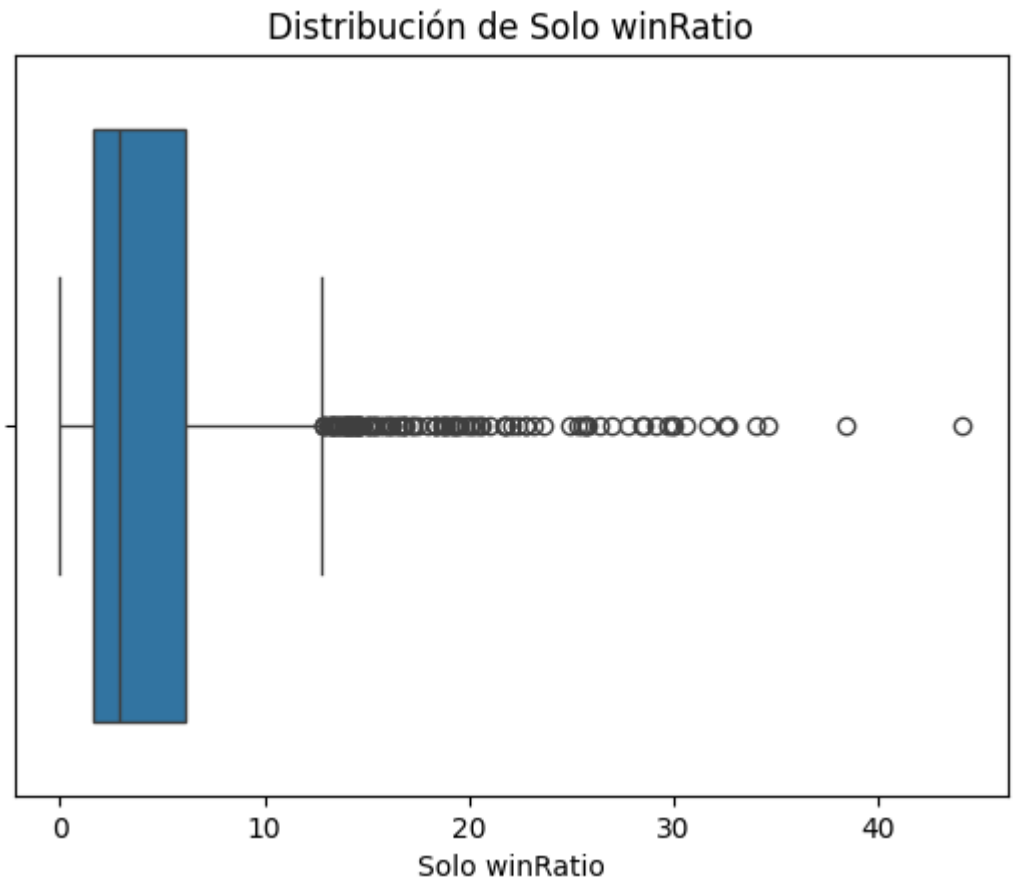


Box Plot

```
sb.boxplot(x='Solo kd', data=df)
plt.title('Distribución de Solo kd')
plt.show()

sb.boxplot(x='Solo winRatio', data=df)
plt.title('Distribución de Solo winRatio')
plt.show()
```


Distribución de Solo kd
0
2
4
6
8
10
Solo kd



Solo KD: La línea dentro de la caja representa la mediana del Solo kd, que parece estar alrededor de 1. Esto significa que el 50% de los jugadores tienen un ratio de eliminación-muerte menor o igual a 1.

Se observa un grupo considerable de outliers a la derecha del gráfico, es decir, jugadores que tienen un Solo kd significativamente mayor que el promedio. Algunos jugadores tienen valores de Solo kd que llegan hasta 10, pero son casos poco comunes.

Solo winRatio: La línea en la caja sugiere que la mediana del win ratio está entre 5% y 10%, lo que indica que el 50% de los jugadores ganan menos del 10% de las partidas que juegan.

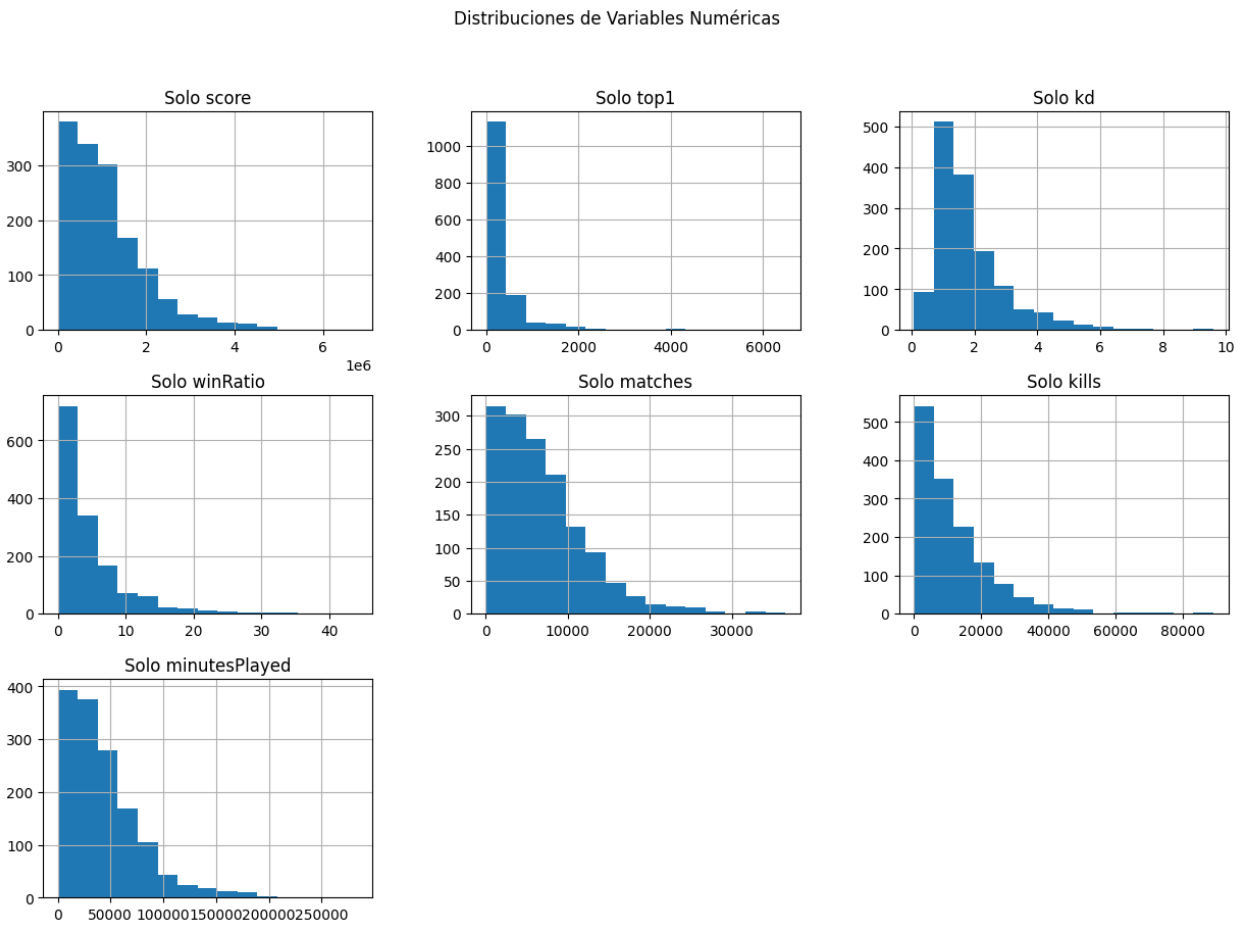
Nuevamente, se observan muchos outliers hacia la derecha, con algunos jugadores logrando ratios de victoria que superan el 40%. Estos jugadores son excepcionales en comparación con la mayoría.

Histogramas

Solo score: -Hay una concentración de valores bajos y una larga cola hacia la derecha, lo que nos indicaría que la mayoría de los jugadores tienen una puntuación baja, mientras que muy pocos tienen puntuaciones extremadamente altas. Solo top1: -La mayoría de los jugadores tienen pocas o ninguna victoria, mientras que solo unos pocos tienen un gran número de victorias. Solo kd: -La mayoría de los jugadores tienen un kd bajo, con pocos jugadores teniendo un kd

muy alto. -Tiene una distribución sesgada a la derecha, indicando que la mayoría de los jugadores no tienen una relación de asesinatos/muertes alta. Solo winRatio: -La mayoría de los jugadores tienen una proporción de victorias baja (menor a 10%), con solo unos pocos jugadores alcanzando porcentajes mayores. Solo matches: -Se tiene una distribución sesgada a la derecha, nos indica que es común tener pocos jugadores que juegan muchas partidas y que la mayoría han jugado un número relativamente bajo de partidas, mientras que unos pocos han jugado muchas más. Solo kills: -La mayoría de los jugadores tienen un número bajo de asesinatos, y pocos jugadores tienen un número muy alto. -Esta variable también muestra una distribución sesgada a la derecha. Solo minutesPlayed: -La mayoría de los jugadores tienen menos tiempo de juego, y hay algunos jugadores con una cantidad de tiempo de juego muy alta.

```
df_numeric.hist(bins=15, figsize=(15, 10), layout=(3, 3))
plt.suptitle('Distribuciones de Variables Numéricas')
plt.show()
```



Identificar los jugadores con el mejor rendimiento en cada modo

```
# Ordenar por rendimiento en el modo Solo
mejores_jugadores_solo = df.sort_values(by=['Solo score', 'Solo
```

```
winRatio', 'Solo kd'], ascending=False)

# Mostrar los jugadores con el mejor rendimiento en cada modo
print("Mejores jugadores en Solo:\n",
mejores_jugadores_solo[['Player', 'Solo score', 'Solo winRatio', 'Solo
kd']].head(10))
```

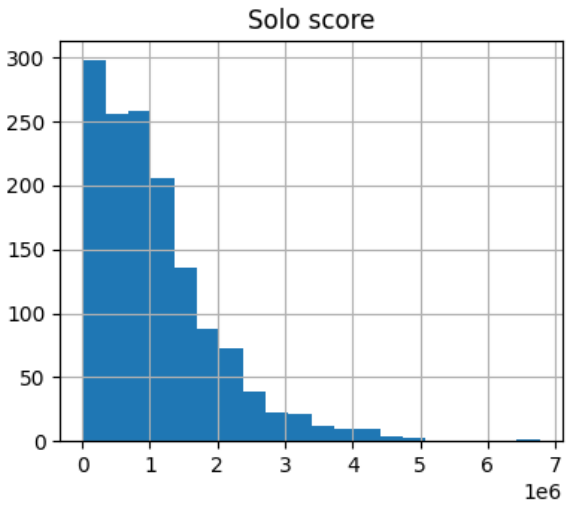
```
Mejores jugadores en Solo:
                Player  Solo score  Solo winRatio  Solo kd
8               ᴮᵒˢˢン  6772116           32.6     1.39
11     DarkAssaSSin._     4840719           30.6     3.34
110         TEBELKING     4784847            5.5     0.95
14            MachiTv     4659548           19.5     3.90
53          Ninjadad5     4595164            0.8     1.01
2              Ranger     4519465           34.0     9.60
144          RailCart     4479084            1.8     0.73
37               Ci2i     4323464            1.6     0.84
17          prOMancos     4300077            1.8     1.27
186  facu giorgetta29     4253032            7.0     1.36
```

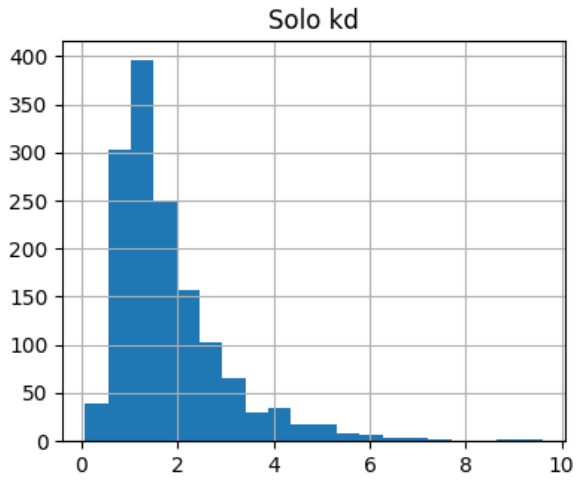
1-Relación entre tiempo jugado y rendimiento:

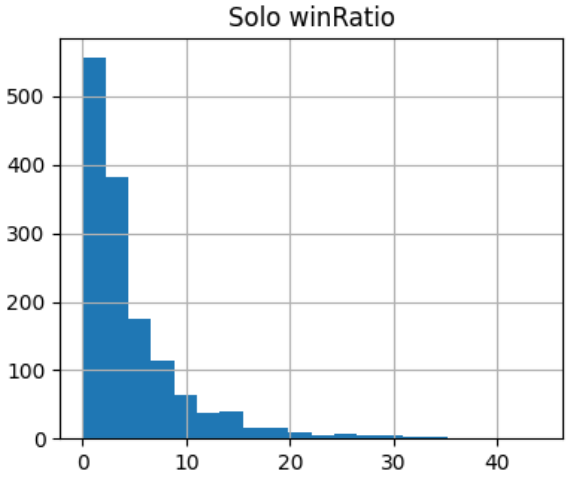
Los jugadores que dedican más minutos a jugar en un modo de juego específico como los modos de juegos (solo/duo/trios), tienden a tener un mayor kd (kill/death ratio) y una mayor proporción de victorias (winRatio) en ese modo. Lo que se quiere saber es que dependiendo el modo de juego, cual es el que tiene mas winrate entre los jugadores, y si existe una fuerte correlación entre la puntuación (score) y el número de victorias (top1) en los tres modos de juego.

Fase 3 Data Preparation

```
# Histograma para visualizar la distribución
df[['Solo score', 'Solo kd', 'Solo winRatio']].hist(bins=20,
figsize=(10, 8))
plt.show()
```







Solo score (Puntuación en Solitario): Este tipo de distribución sugiere que la mayor parte de los jugadores están concentrados en un rango bajo de puntuación, con un número reducido de jugadores que logran puntuaciones muy altas. Esto puede indicar una disparidad en el rendimiento, donde solo unos pocos jugadores son significativamente más exitosos que el promedio.

Solo kd (Relación K/D en Solitario): Esto indica que muchos jugadores tienden a mantener un rendimiento equilibrado (un número similar de bajas y muertes), mientras que solo algunos jugadores tienen un rendimiento mucho más alto (con un kd por encima de 2 o 3). Los valores extremos pueden ser jugadores con mucha habilidad o que se enfrentan a oponentes de menor nivel.

Solo winRatio (Porcentaje de Victorias en Solitario): La mayoría de los jugadores probablemente tengan dificultades para ganar en partidas en solitario. Solo un pequeño porcentaje logra un winRatio alto, lo cual es esperable, ya que las victorias son un resultado más exclusivo en un entorno competitivo como Fortnite.

Todas las variables muestran una distribución sesgada a la derecha, lo que sugiere que la mayoría de los jugadores tienen un rendimiento moderado, mientras que solo unos pocos logran resultados significativamente mejores.

Para mejorar el rendimiento del modelo vamos a normalizar los datos así obtener un mejor nivel predictivo de los datos.

Normalización de los datos

Primero seleccionamos las columnas que vamos a normalizar y utilizamos MinMaxScaler para escalar a valores entre 0 y 1.

```
columnasNormalizar = ['Solo score', 'Solo top1', 'Solo kd', 'Solo
winRatio', 'Solo matches', 'Solo kills', 'Solo minutesPlayed']
```

Columnas antes de la normalización

```
df.head(10)
```


```
{"column_count":8,"columns":
[{"dtype":"object","name":"Player","stats":{"categories":
[{"count":1,"name":"ᴮᵒˢˢン"},{"count":1,"name":"Ranger"},
{"count":8,"name":"8 others"}],"nan_count":0,"unique_count":10}},
{"dtype":"int64","name":"Solo score","stats":{"histogram":
[{"bin_end":691496.4,"bin_start":15872,"count":3},
{"bin_end":1367120.8,"bin_start":691496.4,"count":2},
{"bin_end":2042745.2000000002,"bin_start":1367120.8,"count":0},
{"bin_end":2718369.6,"bin_start":2042745.2000000002,"count":2},
{"bin_end":3393994,"bin_start":2718369.6,"count":1},
{"bin_end":4069618.4000000004,"bin_start":3393994,"count":0},
{"bin_end":4745242.8,"bin_start":4069618.4000000004,"count":1},
{"bin_end":5420867.2,"bin_start":4745242.8,"count":0},
{"bin_end":6096491.600000001,"bin_start":5420867.2,"count":0},
{"bin_end":6772116,"bin_start":6096491.600000001,"count":1}],"max":"67
72116","min":"15872","nan_count":0,"unique_count":10}},
{"dtype":"int64","name":"Solo top1","stats":{"histogram":
[{"bin_end":653.4,"bin_start":6,"count":3},
{"bin_end":1300.8,"bin_start":653.4,"count":0},
{"bin_end":1948.1999999999998,"bin_start":1300.8,"count":5},
{"bin_end":2595.6,"bin_start":1948.1999999999998,"count":0},
{"bin_end":3243,"bin_start":2595.6,"count":0},
{"bin_end":3890.3999999999996,"bin_start":3243,"count":0},
{"bin_end":4537.8,"bin_start":3890.3999999999996,"count":0},
{"bin_end":5185.2,"bin_start":4537.8,"count":1},
{"bin_end":5832.599999999999,"bin_start":5185.2,"count":0},
{"bin_end":6480,"bin_start":5832.599999999999,"count":1}],"max":"6480"
,"min":"6","nan_count":0,"unique_count":10}},
{"dtype":"float64","name":"Solo kd","stats":{"histogram":
[{"bin_end":1.383,"bin_start":0.47,"count":3},
{"bin_end":2.296,"bin_start":1.383,"count":1},
{"bin_end":3.2089999999999996,"bin_start":2.296,"count":0},
{"bin_end":4.122,"bin_start":3.2089999999999996,"count":2},
{"bin_end":5.034999999999999,"bin_start":4.122,"count":1},
```

{"bin_end":142287,"bin_start":113977.4,"count":1},
{"bin_end":170596.59999999998,"bin_start":142287,"count":0},
{"bin_end":198906.19999999998,"bin_start":170596.59999999998,"count":0
},{"bin_end":227215.8,"bin_start":198906.19999999998,"count":0},
{"bin_end":255525.4,"bin_start":227215.8,"count":0},
{"bin_end":283835,"bin_start":255525.4,"count":1}],"max":"283835","min
":"739","nan_count":0,"unique_count":10}},
{"dtype":"int64","name":"_deepnote_index_column"}],"row_count":10,"row
s":[{"Player":"Bossン","Solo kd":1.39,"Solo kills":18610,"Solo
matches":19864,"Solo minutesPlayed":283835,"Solo score":6772116,"Solo
top1":6480,"Solo winRatio":32.6,"_deepnote_index_column":8},
{"Player":"Ranger","Solo kd":9.6,"Solo kills":85481,"Solo
matches":13488,"Solo minutesPlayed":122171,"Solo score":4519465,"Solo
top1":4582,"Solo winRatio":34,"_deepnote_index_column":2},
{"Player":"Twitch Kayotica","Solo kd":3.23,"Solo kills":39131,"Solo
matches":13438,"Solo minutesPlayed":96777,"Solo score":2919037,"Solo
top1":1310,"Solo winRatio":9.7,"_deepnote_index_column":5},
{"Player":"Prospering","Solo kd":4.37,"Solo kills":36328,"Solo
matches":10150,"Solo minutesPlayed":81389,"Solo score":2476763,"Solo
top1":1828,"Solo winRatio":18,"_deepnote_index_column":0},
{"Player":"FaZe Replays","Solo kd":3.84,"Solo kills":66161,"Solo
matches":18670,"Solo minutesPlayed":76258,"Solo score":2389537,"Solo
top1":1454,"Solo winRatio":7.8,"_deepnote_index_column":6},
{"Player":"Twitch.GryphonRB","Solo kd":6.32,"Solo kills":19591,"Solo
matches":4429,"Solo minutesPlayed":36245,"Solo score":1136282,"Solo
top1":1327,"Solo winRatio":30,"_deepnote_index_column":4},
{"Player":"Idk_Pi","Solo kd":0.84,"Solo kills":3005,"Solo
matches":3687,"Solo minutesPlayed":32453,"Solo score":752869,"Solo
top1":121,"Solo winRatio":3.3,"_deepnote_index_column":3},
{"Player":"BH nixxxay","Solo kd":8.71,"Solo kills":35895,"Solo
matches":5817,"Solo minutesPlayed":12732,"Solo score":439562,"Solo
top1":1694,"Solo winRatio":29.1,"_deepnote_index_column":1},
{"Player":"CIUPEA 144.HZ","Solo kd":0.61,"Solo kills":1174,"Solo
matches":1938,"Solo minutesPlayed":2441,"Solo score":54479,"Solo
top1":9,"Solo winRatio":0.5,"_deepnote_index_column":9},
{"Player":"NiteGamerYT 190k","Solo kd":0.47,"Solo kills":200,"Solo
matches":429,"Solo minutesPlayed":739,"Solo score":15872,"Solo
top1":6,"Solo winRatio":1.4,"_deepnote_index_column":7}]}

Columnas después de la normalización

```
# Aplicar Min-Max Scaler
scaler = MinMaxScaler()
dfNormalizado = df.copy()  # Crear una copia del DataFrame
dfNormalizado[columnasNormalizar] =
scaler.fit_transform(df[columnasNormalizar])
dfNormalizado.head(10)
```

```
2,"Solo minutesPlayed":1.8651388276773907e-3,"Solo
score":1.6872885071257243e-3,"Solo top1":9.25925925925926e-4,"Solo
winRatio":3.1746031746031744e-2,"_deepnote_index_column":7}]}
```


Valor cercano a 0: El jugador tiene un bajo rendimiento en comparación con otros jugadores para esa métrica específica. Valor cercano a 1: El jugador tiene un alto rendimiento en comparación con otros jugadores para esa métrica específica.

"Boss" es el mejor jugador en términos de Solo score, Solo top1, Solo kd y Solo minutesPlayed, porque sus valores están en 1, lo que indica que está al tope de las métricas de rendimiento en comparación con los demás jugadores.

La normalización nos ayuda a estandarizar las escalas de las diferentes variables para que los algoritmos de machine learning puedan interpretar correctamente los datos, evitando que variables con mayor rango afecten de manera desproporcionada los resultados.

Tratamiento de outliers

```
# Calcular el IQR
Q1 = df['Solo kd'].quantile(0.25)
Q3 = df['Solo kd'].quantile(0.75)
IQR = Q3 - Q1

# Definir los límites inferior y superior
lim_inferior = Q1 - 1.5 * IQR
lim_superior = Q3 + 1.5 * IQR

# Filtrar datos dentro de los límites
df_solo_sin_outliers = df[(df['Solo kd'] >= lim_inferior) & (df['Solo
kd'] <= lim_superior)]

print("Número de filas antes de eliminar outliers:", len(df))
print("Número de filas después de eliminar outliers:",
len(df_solo_sin_outliers))
```

```
Número de filas antes de eliminar outliers: 1435
Número de filas después de eliminar outliers: 1343
```

```
# Verificar límites del IQR
print(f"Límite inferior: {lim_inferior}, Límite superior:
{lim_superior}")
```

```
Límite inferior: -0.675, Límite superior: 3.9250000000000003
```

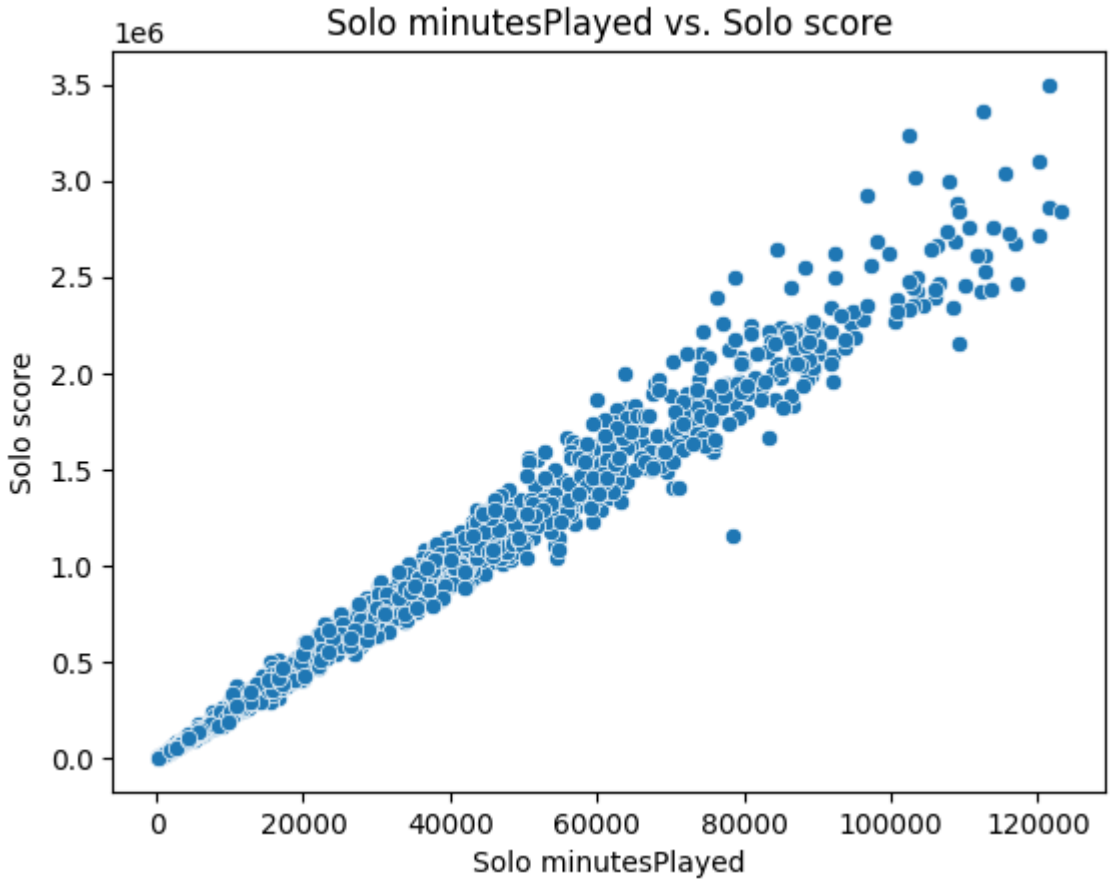
```
# Cálculo del IQR para Solo minutesPlayed
Q1_mp = df['Solo minutesPlayed'].quantile(0.25)
Q3_mp = df['Solo minutesPlayed'].quantile(0.75)
IQR_mp = Q3_mp - Q1_mp
```

```
# Límite inferior y superior para Solo minutesPlayed
lim_inferior_mp = Q1_mp - 1.5 * IQR_mp
lim_superior_mp = Q3_mp + 1.5 * IQR_mp

# Filtrar datos sin outliers en Solo kd y Solo minutesPlayed
df_solo_sin_outliers =
df_solo_sin_outliers[(df_solo_sin_outliers['Solo minutesPlayed'] >=
lim_inferior_mp) &

(df_solo_sin_outliers['Solo minutesPlayed'] <= lim_superior_mp)]

# Visualización final del scatterplot sin outliers
sb.scatterplot(x='Solo minutesPlayed', y='Solo score',
data=df_solo_sin_outliers)
plt.title('Solo minutesPlayed vs. Solo score')
plt.show()
```



El gráfico resultante muestra una relación positiva entre el tiempo jugado en modo solo (Solo minutesPlayed) y la puntuación obtenida en ese modo (Solo score). Es decir, a medida que los jugadores pasan más tiempo jugando, tienden a obtener puntuaciones más altas.

Esta relación es consistente y sigue una tendencia lineal, aunque se observan algunas dispersiones (pequeñas variaciones) a medida que aumenta el número de minutos jugados. Al

haber filtrado los outliers, se han eliminado valores atípicos que podrían haber distorsionado la interpretación de la tendencia general. El gráfico se concentra en los datos que representan el comportamiento típico de los jugadores.