UNIVERSIDAD AMERICANA

Clase Practica 2

Inteligencia de Negocios

Entrega:

- Lester Alejandro Rodriguez Cuevas

Docente: Arlen Jeannette Lopez

16 de Septiembre

Managua, Nicaragua

El presente preprocesamiento de datos, se realizó en un archivo jupyter .ipynb, se utilizó un archivo de tipo .csv con el nombre de ‘matches’, donde se tiene toda la data e información de ventas de casas en un sector. Se subió el archivo a un repositorio de github y se mandó a llamar por medio de un url.

```python
import pandas as pd
import numpy as np
url = 'https://raw.githubusercontent.com/lard0503/premierleaguematches/main/matches.csv'
df = pd.read_csv(url)
print(df.head()) #primeros 5 registros de cada columna
```

✓ 0.8s

```
   Unnamed: 0        date          time            comp        round  day  \
0           0  2020-09-21  20:15 (21:15)  Premier League  Matchweek 2  Mon
1           2  2020-09-27  16:30 (17:30)  Premier League  Matchweek 3  Sun
2           4  2020-10-03  17:30 (18:30)  Premier League  Matchweek 4  Sat
3           5  2020-10-17  17:30 (18:30)  Premier League  Matchweek 5  Sat
4           7  2020-10-24  12:30 (13:30)  Premier League  Matchweek 6  Sat

   venue result  gf  ga  ...  match report  notes  sh  sot  dist fk pk pkatt  \
0   Away      W   3   1  ...  Match Report    NaN  13    8  21.1  2  1     1
1   Home      L   2   5  ...  Match Report    NaN  16    5  19.8  1  0     0
2   Away      D   1   1  ...  Match Report    NaN  23    1  18.2  1  0     0
3   Home      W   1   0  ...  Match Report    NaN  13    5  17.7  0  0     0
4   Away      D   1   1  ...  Match Report    NaN  14    7  20.9  1  0     0

   season            team
0    2024  Manchester City
1    2024  Manchester City
2    2024  Manchester City
3    2024  Manchester City
4    2024  Manchester City

[5 rows x 28 columns]
```

Se hace uso de pandas y numpy en Python para cargar y visualizar un archivo CSV. El archivo contiene datos de partidos de la Premier League, y se está leyendo desde una URL externa usando pd.read_csv(). Después de cargar el archivo en un DataFrame llamado df, se imprime la cabeza del DataFrame con df.head(), lo que muestra los primeros 5 registros.

Las columnas incluyen información sobre los partidos, como la fecha (date), la hora (time), la competición (comp), el resultado (result), goles a favor y en contra (gf, ga), así como estadísticas del partido, como tiros (sh), tiros a puerta (sot), y otras métricas como distancia (dist), tiros libres (fk), penales (pk y pkatt). También se muestran la temporada (season) y el equipo (team), con "Manchester City" en todos los registros mostrados. Este proceso es común para la exploración inicial de un dataset en análisis de datos.

```python
print(df.info()) #Número de registros y tipo de variables
print(df.describe()) #Estadísticas de las variables numéricas
```

[56] ✓ 0.1s

Se ejecutan dos funciones de pandas, df.info() y df.describe(), para obtener información detallada sobre el DataFrame df.

df.info() proporciona un resumen sobre el número de registros, las columnas disponibles y los tipos de datos de cada una, junto con la cantidad de valores no nulos en cada columna. Esto es útil para tener una visión general del tamaño del dataset y detectar si hay columnas con datos faltantes.

df.describe() genera estadísticas descriptivas para las columnas numéricas del DataFrame, como el promedio (mean), la desviación estándar (std), los valores mínimo, máximo, y los percentiles (25%, 50%, 75%). Esta función te permite entender la distribución de los datos numéricos en el dataset.

```
#venue,result,gf,ga,opponent,xg,xga,poss,sh,sot,fk,pk,pkatt,season,team
columnas_importantes = ['venue','result','gf','ga','opponent','xg','xga','poss','sh','sot','fk','pk','pkatt','season','team']
dataTrans = df[columnas_importantes] #creamos un nuevo dataframe con las columnas importantes
print(dataTrans.head()) #imprimimos las primeras 5 filas del nuevo dataframe para confirmar que esta bien
```

[57] ✓ 0.0s

```
  venue result  gf  ga        opponent   xg  xga  poss  sh  sot  fk  pk  \
0  Away      W   3   1         Wolves  1.9  0.6    65  13    8   2   1
1  Home      L   2   5  Leicester City  0.9  2.9    72  16    5   1   0
2  Away      D   1   1    Leeds United  1.2  2.4    49  23    1   1   0
3  Home      W   1   0         Arsenal  1.3  0.9    58  13    5   0   0
4  Away      D   1   1       West Ham  1.0  0.3    69  14    7   1   0

   pkatt  season            team
0      1    2024  Manchester City
1      0    2024  Manchester City
2      0    2024  Manchester City
3      0    2024  Manchester City
4      0    2024  Manchester City
```

Se crea un nuevo DataFrame llamado dataTrans a partir de un DataFrame original (df), seleccionando solo las columnas consideradas importantes, que están listadas en columnas_importantes. Estas incluyen información clave como el resultado del partido (result), los goles a favor (gf), goles en contra (ga), equipo oponente (opponent), y varias estadísticas del partido como posesión de balón (poss), tiros (sh), tiros a puerta (sot), penales (pk, pkatt), entre otras. Después de crear el nuevo DataFrame, se imprime con head() para verificar que las primeras 5 filas fueron seleccionadas correctamente.

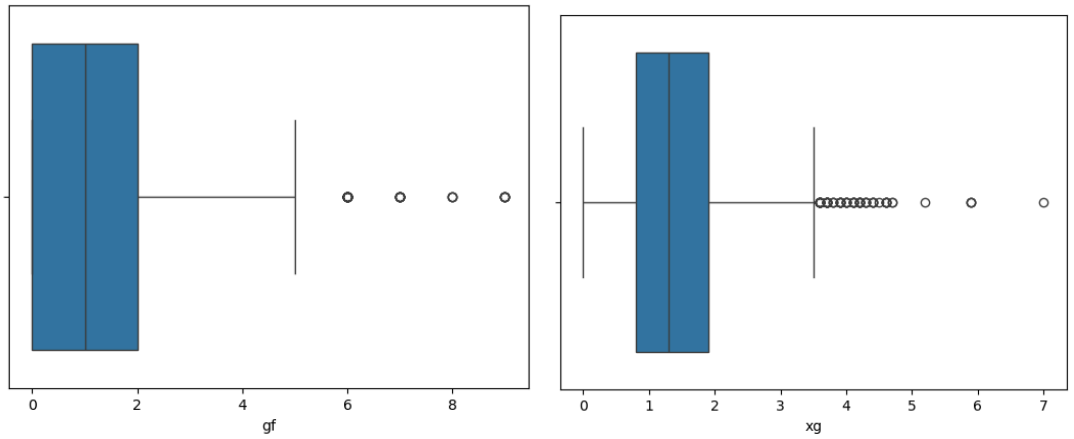
```
missing_values = dataTrans.isnull().sum() #contamos los valores nulos en cada columna
print(missing_values[missing_values >= 0]) #imprimimos la cantidad de valores nulos por columna
```

✓ 0.0s

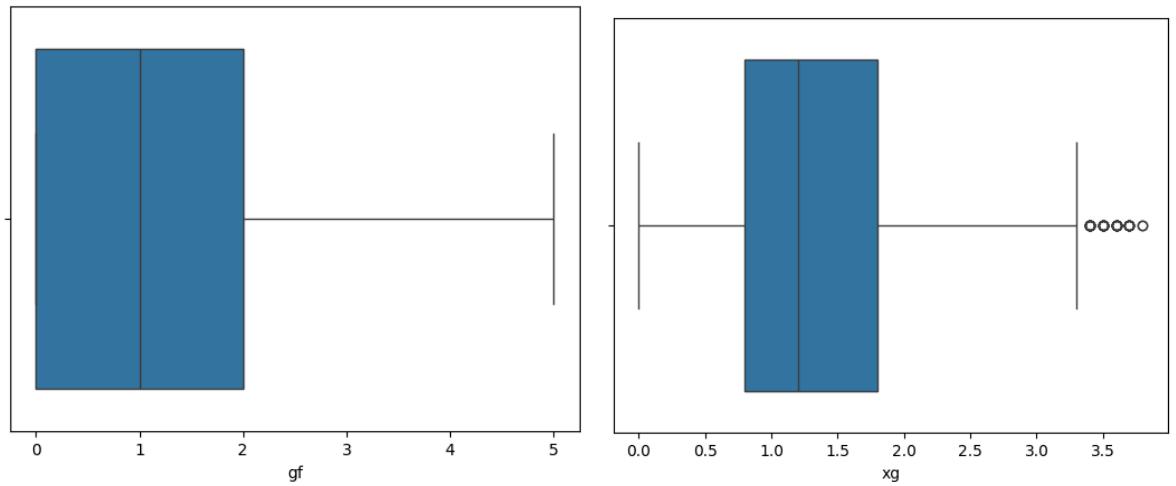
```
venue      0
result     0
gf         0
ga         0
opponent   0
xg         0
xga        0
poss       0
sh         0
sot        0
fk         0
pk         0
pkatt      0
season     0
team       0
dtype: int64
```

Se utiliza el método isnull() junto con sum() para contar el número de valores nulos en cada columna del DataFrame dataTrans. Luego, se imprime el resultado, que muestra el número de

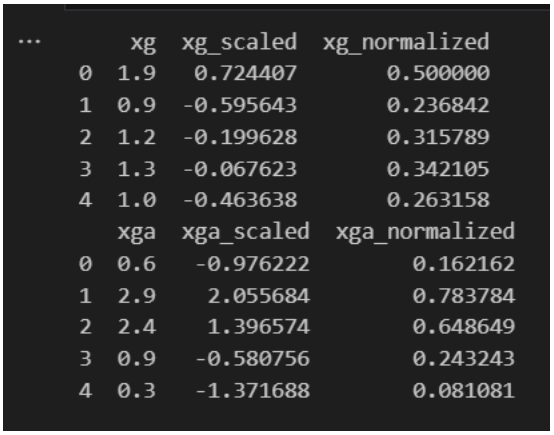
valores faltantes por columna. En este caso, todas las columnas (venue, result, gf, etc.) tienen cero valores nulos, lo que indica que no hay datos faltantes en las columnas seleccionadas.



Se utilizó Seaborn y Matplotlib para generar gráficos de caja (boxplots) de varias columnas numéricas del DataFrame, como gf (goles a favor) y xg (expected goals). Estos gráficos muestran la distribución de los datos, destacando los outliers (valores atípicos) que están por fuera del rango normal, como goles a favor superiores a 4 y expected goals superiores a 4.5. Los boxplots son útiles para identificar estos outliers, lo que permite tomar decisiones sobre su tratamiento en el análisis de datos. (Se realizaron más gráficos de boxplot pero para ejemplificar utilizamos dos nada mas)



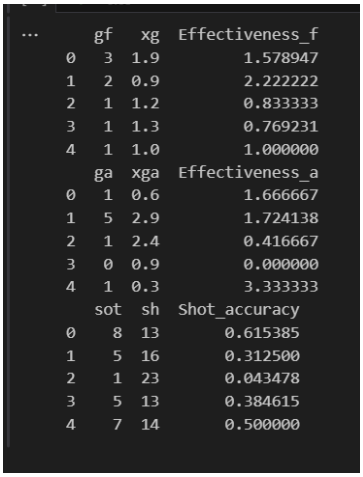
Se eliminaron outliers de varias columnas del DataFrame (dataTrans) usando el Z-score, que identifica y filtra valores fuera de 3 desviaciones estándar de la media. Para cada columna numérica como gf, ga, xg, y otras, se aplica el filtro para conservar sólo los valores dentro de ese rango. Después de eliminar los outliers, se generan nuevos gráficos de caja (boxplots) que muestran una distribución más ajustada de los datos sin valores extremos, como se observa en las gráficas de gf y xg, donde los outliers han sido reducidos, y las distribuciones se ven más compactas.



	xg	xg_scaled	xg_normalized
0	1.9	0.724407	0.500000
1	0.9	-0.595643	0.236842
2	1.2	-0.199628	0.315789
3	1.3	-0.067623	0.342105
4	1.0	-0.463638	0.263158

	xga	xga_scaled	xga_normalized
0	0.6	-0.976222	0.162162
1	2.9	2.055684	0.783784
2	2.4	1.396574	0.648649
3	0.9	-0.580756	0.243243
4	0.3	-1.371688	0.081081

Se utilizaron dos técnicas de preprocesamiento de datos a las columnas xg y xga: **Estandarización** (usando Z-score con StandardScaler) y **Normalización** (usando Min-Max Scaling con MinMaxScaler). La estandarización ajusta los datos para que tengan media 0 y desviación estándar 1, mientras que la normalización ajusta los valores dentro de un rango de 0 a 1. Esto es útil para preparar los datos para ciertos algoritmos de machine learning que son sensibles a la escala. Los resultados muestran las columnas originales junto con sus versiones escaladas y normalizadas.



	gf	xg	Effectiveness_f
0	3	1.9	1.578947
1	2	0.9	2.222222
2	1	1.2	0.833333
3	1	1.3	0.769231
4	1	1.0	1.000000

	ga	xga	Effectiveness_a
0	1	0.6	1.666667
1	5	2.9	1.724138
2	1	2.4	0.416667
3	0	0.9	0.000000
4	1	0.3	3.333333

	sot	sh	Shot_accuracy
0	8	13	0.615385
1	5	16	0.312500
2	1	23	0.043478
3	5	13	0.384615
4	7	14	0.500000

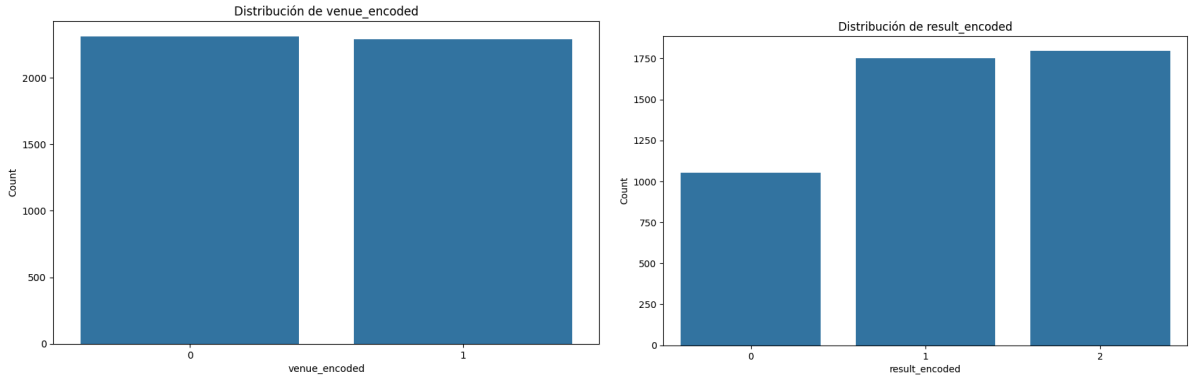
Creé nuevas variables en el DataFrame dataTrans para medir diferentes formas de **efectividad**. Se calculan tres métricas:

1. **Effectiveness_f**: Calcula la efectividad ofensiva dividiendo los goles anotados (gf) entre los goles esperados (xg).
2. **Effectiveness_a**: Calcula la efectividad defensiva dividiendo los goles recibidos (ga) entre los goles esperados en contra (xga).
3. **Shot_accuracy**: Calcula la precisión de los tiros dividiendo los tiros a gol (sot) entre los tiros totales (sh).

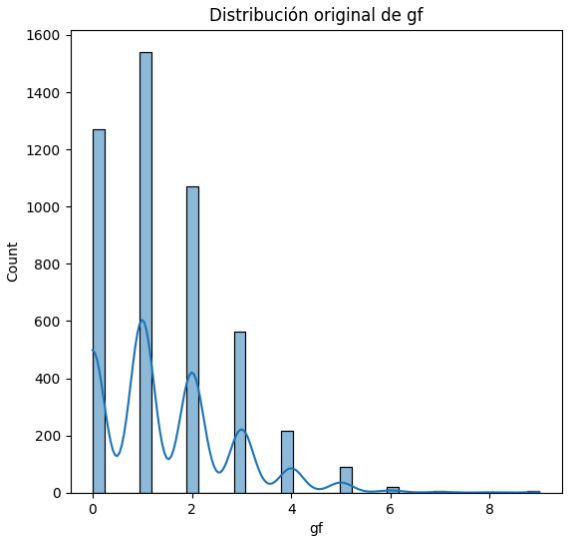
Estos cálculos permiten evaluar el rendimiento ofensivo, defensivo y la precisión de los equipos en los partidos analizados.

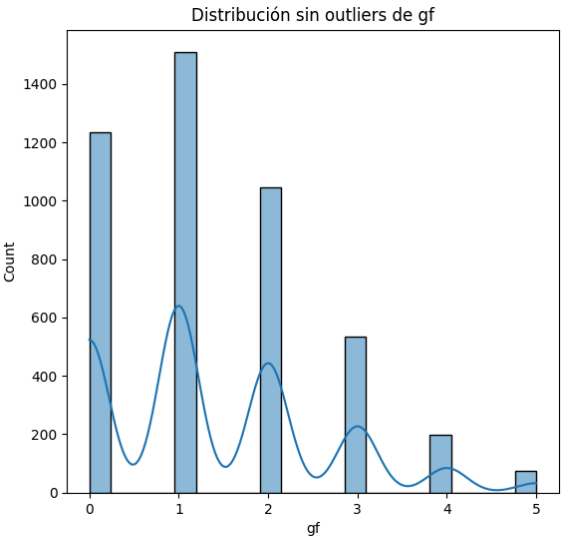
```
Simbología de la codificación de Venue
  venue  venue_encoded
0  Away              0
1  Home              1
  venue  venue_encoded
0  Away              0
1  Home              1
2  Away              0
3  Home              1
4  Away              0
Simbología de la codificación de Resultados
  result  result_encoded
2      D               0
1      L               1
0      W               2
  result  result_encoded
0      W               2
1      L               1
2      D               0
3      W               2
4      D               0
Simbología de la codificación de Team
                         team  team_encoded
266                   Arsenal             0
380               Aston Villa             1
1406              Bournemouth             2
1558                Brentford             3
570   Brighton and Hove Albion             4
608                   Burnley             5
114                   Chelsea             6
```

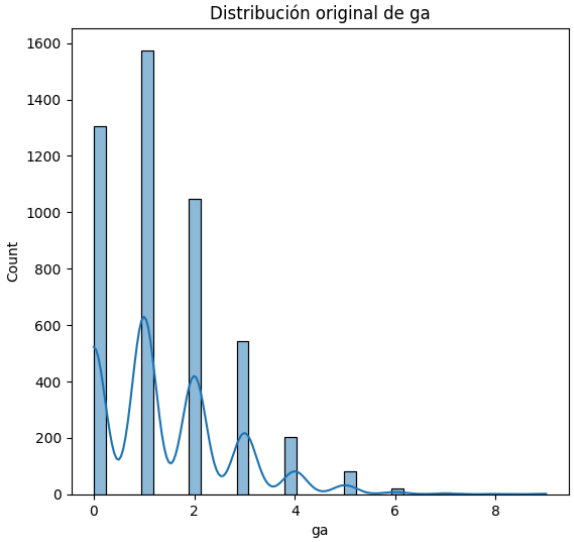
Utilice la tecnica LabelEncoder para convertir las columnas categóricas venue, result y team del DataFrame dataTrans en valores numéricos. Primero se aplica a la columna venue (visitante o local), luego a result (ganar, empatar o perder) y finalmente a team (nombres de equipos). Después de cada transformación, se imprime el head() que muestra la correspondencia entre los valores originales y sus equivalentes codificados. Esto es útil para preparar los datos categóricos para modelos de machine learning que requieren variables numéricas.

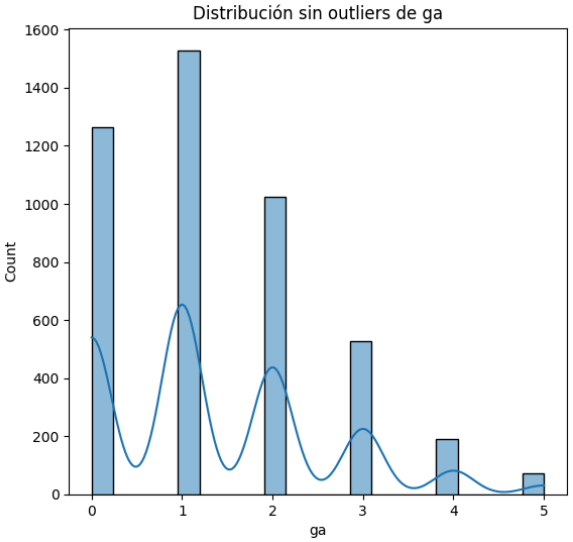


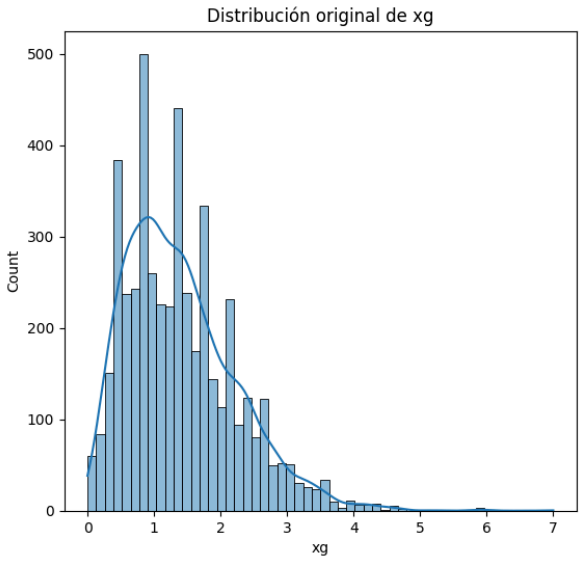
Realice gráficos para observar la distribución de las columnas codificadas, para ver la representación de las mismas sin sus datos categóricos, pero numéricos.

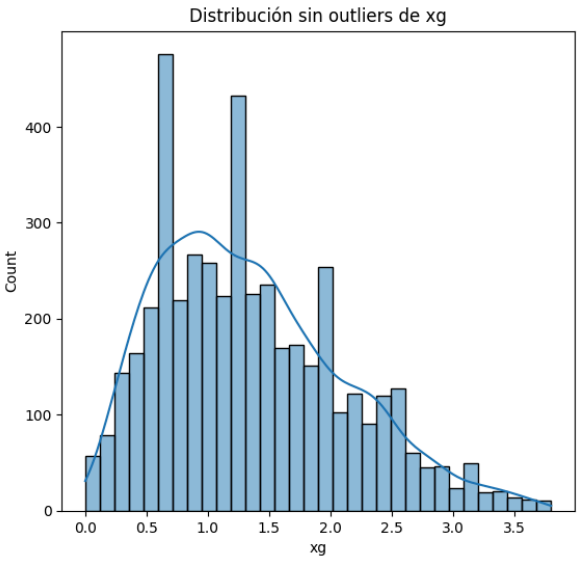






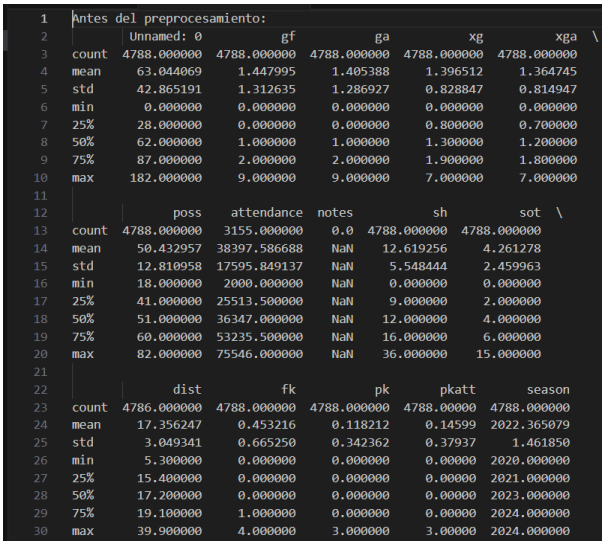






Genere gráficos comparativos de la distribución original y la distribución sin outliers para varias columnas del DataFrame, como gf (goles a favor), ga (goles en contra), xg (expected

goals), y otras. Use seaborn para crear histogramas con líneas KDE que muestran las distribuciones de los datos antes y después de eliminar los outliers. Al comparar las dos distribuciones, se puede observar cómo la eliminación de valores atípicos hace que los datos estén más centrados y ajustados, lo que facilita análisis más precisos sin la influencia de valores extremos.



Antes del preprocesamiento:

	Unnamed: 0	gf	ga	xg	xga
count	4788.000000	4788.000000	4788.000000	4788.000000	4788.000000
mean	63.044069	1.447995	1.405388	1.396512	1.364745
std	42.865191	1.312635	1.286927	0.828847	0.814947
min	0.000000	0.000000	0.000000	0.000000	0.000000
25%	28.000000	0.000000	0.000000	0.800000	0.700000
50%	62.000000	1.000000	1.000000	1.300000	1.200000
75%	87.000000	2.000000	2.000000	1.900000	1.800000
max	182.000000	9.000000	9.000000	7.000000	7.000000

	poss	attendance	notes	sh	sot
count	4788.000000	3155.000000	0.0	4788.000000	4788.000000
mean	50.432957	38397.586688	NaN	12.619256	4.261278
std	12.810958	17595.849137	NaN	5.548444	2.459963
min	18.000000	2000.000000	NaN	0.000000	0.000000
25%	41.000000	25513.500000	NaN	9.000000	2.000000
50%	51.000000	36347.000000	NaN	12.000000	4.000000
75%	60.000000	53235.500000	NaN	16.000000	6.000000
max	82.000000	75546.000000	NaN	36.000000	15.000000

	dist	fk	pk	pkatt	season
count	4786.000000	4788.000000	4788.000000	4788.00000	4788.000000
mean	17.356247	0.453216	0.118212	0.14599	2022.365079
std	3.049341	0.665250	0.342362	0.37937	1.461850
min	5.300000	0.000000	0.000000	0.00000	2020.000000
25%	15.400000	0.000000	0.000000	0.00000	2021.000000
50%	17.200000	0.000000	0.000000	0.00000	2023.000000
75%	19.100000	1.000000	0.000000	0.00000	2024.000000
max	39.900000	4.000000	3.000000	3.00000	2024.000000

Después del preprocesamiento:

	gf	ga	xg	xga	poss
count	4603.000000	4603.000000	4603.000000	4603.000000	4603.000000
mean	1.387791	1.362372	1.351227	1.340561	50.292418
std	1.210270	1.200786	0.757630	0.758681	12.642516
min	0.000000	0.000000	0.000000	0.000000	18.000000
25%	0.000000	0.000000	0.800000	0.700000	41.000000
50%	1.000000	1.000000	1.200000	1.200000	50.000000
75%	2.000000	2.000000	1.800000	1.800000	60.000000
max	5.000000	5.000000	3.800000	3.700000	82.000000

	sh	sot	fk	pk	pkatt	...
count	4603.000000	4603.000000	4603.000000	4603.000000	4603.000000	...
mean	12.383663	4.139909	0.451010	0.114491	0.140561	...
std	5.211820	2.297037	0.664442	0.337006	0.371770	...
min	0.000000	0.000000	0.000000	0.000000	0.000000	...
25%	9.000000	2.000000	0.000000	0.000000	0.000000	...
50%	12.000000	4.000000	0.000000	0.000000	0.000000	...
75%	16.000000	6.000000	1.000000	0.000000	0.000000	...
max	28.000000	11.000000	4.000000	3.000000	3.000000	...

	xg_normalized	xga_scaled	xga_normalized	Effectiveness
count	4603.000000	4.603000e+03	4603.000000	4598.000000
mean	0.355586	1.234921e-17	0.362314	1.101372
std	0.199376	1.000109e+00	0.205049	1.278751
min	0.000000	-1.767154e+00	0.000000	0.000000
25%	0.210526	-8.443997e-01	0.189189	0.000000
50%	0.315789	-1.852897e-01	0.324324	0.952381
75%	0.473684	6.056424e-01	0.486486	1.538462
max	1.000000	3.110261e+00	1.000000	20.000000

	Effectiveness_f	Effectiveness_a	Shot_accuracy	venue_encoded
count	4598.000000	4597.000000	4601.000000	4603.000000
mean	1.101372	1.078261	0.342396	0.497284
std	1.278751	1.170085	0.161082	0.500047
min	0.000000	0.000000	0.000000	0.000000
25%	0.000000	0.000000	0.235294	0.000000

Comparé las estadísticas descriptivas del DataFrame antes y después del preprocesamiento de datos, utilizando el método describe(). En las imágenes, se muestran cambios clave en métricas como la media (mean), desviación estándar (std) y valores máximo y mínimo en columnas como gf (goles a favor), ga (goles en contra) y xg (expected goals). Después del preprocesamiento, se observa una reducción en los valores extremos (outliers) y una disminución en las desviaciones estándar, lo que indica que los datos están más concentrados y ajustados, mejorando la calidad del análisis posterior.

Antes del preprocesamiento:

Este es el resumen estadístico del dataset original, donde las columnas principales (como 'gf', 'ga', 'xg', etc.) no han sido limpiadas ni transformadas. Los valores que se ven son:

count: El número de registros disponibles para cada columna. Por ejemplo, hay 4788 registros en la mayoría de las columnas, pero algunas, como 'attendance', tienen menos (3155).

mean: El promedio de los valores. Por ejemplo, el promedio de goles a favor ('gf') es 1.44 y el promedio de goles en contra ('ga') es 1.40.

std: Desviación estándar, que indica la dispersión de los datos respecto al promedio. Por ejemplo, la desviación estándar de 'gf' es 1.31, lo que significa que hay cierta variabilidad en el número de goles marcados.

min, 25%, 50%, 75%, max: Estos son los valores mínimos, percentiles (25%, 50%, 75%) y el valor máximo de cada columna. Estos te ayudan a entender la distribución de los datos. Por ejemplo, el 50% (mediana) de los goles a favor es 1, y el valor máximo es 9.

Después del preprocesamiento:

Los datos han sido procesados para eliminar outliers y realizar transformaciones adicionales. Las estadísticas después del preprocesamiento son ligeramente diferentes debido a estas acciones:

Reducción en la cantidad de datos: La cantidad de registros ('count') ha disminuido de 4788 a 4603 en la mayoría de las columnas. Esto indica que algunos valores extremos (outliers) fueron eliminados.

Reducción de las desviaciones estándar ('std'): Las desviaciones estándar son menores en el dataset preprocesado, lo que significa que los valores están más cerca del promedio. Por ejemplo, la desviación estándar de 'gf' baja de 1.31 a 1.21, lo que sugiere que el rango de los goles marcados es ahora más consistente.

Máximos más bajos: Algunos valores máximos han sido recortados. Por ejemplo, el valor máximo de 'gf' (goles a favor) pasa de 9 a 5, y en 'xg' (expected goals) pasa de 7 a 3.8. Esto se debe a la eliminación de outliers extremos.

Nuevas columnas añadidas tras el preprocesamiento:

xg_normalized y xga_normalized: Son versiones normalizadas de las columnas 'xg' y 'xga'. Los valores de estas columnas han sido escalados a un rango común (entre 0 y 1), lo que es útil para comparaciones y modelos de machine learning.

Effectiveness_f, Effectiveness_a: Estas columnas calculan la efectividad de goles a favor y goles en contra. El valor máximo de 'Effectiveness' es 20, lo que podría indicar un partido con una gran efectividad de parte de uno de los equipos.

Shot_accuracy: Representa la precisión de los tiros, con un valor medio de 34.2%. Los equipos con partidos más precisos tienen un valor de 1 (100% de precisión), mientras que los menos precisos están en 0.

result_encoded, team_encoded: Estas son columnas codificadas de las variables categóricas 'result' y 'team'. El 'result' y el 'team' han sido transformados en valores numéricos, donde:

result_encoded: 0 es 'derrota', 1 es 'empate', y 2 es 'victoria'.

team_encoded: Asigna un número a cada equipo en el dataset, entre 0 y 25.