CodigoProyecto_Carreola_Silva_CarlosFranciscoJavier

September 20, 2023

[346]:
```
import pandas as pd
import numpy as np
import os
import re
import emoji
import unicodedata
import seaborn as sns
from sklearn.model_selection import train_test_split # pip install scikit-learn
from sklearn.feature_extraction.text import CountVectorizer, TfidfVectorizer
from sklearn.preprocessing import MinMaxScaler, StandardScaler
from sklearn.feature_selection import SelectKBest, f_classif
from PIL import Image
import cufflinks as cf
import stylecloud
from stylecloud import gen_stylecloud
import matplotlib.pyplot as plt
from sklearn.manifold import MDS
from sklearn.preprocessing import MinMaxScaler
from sklearn.manifold import MDS
from sklearn.decomposition import PCA
import seaborn as sns

from sklearn.neighbors import NearestNeighbors
from sklearn.cluster import DBSCAN#Reduccion Dimensionalidad
from sklearn.feature_selection import SelectKBest, f_classif
from sklearn.mixture import GaussianMixture
from sklearn.cluster import AgglomerativeClustering
import plotly.express as px
from sklearn.cluster import KMeans
import matplotlib.pyplot as plt
from scipy.stats import f_oneway
from statsmodels.stats.multicomp import pairwise_tukeyhsd
```

1 el objetivo de este trabajo sera separar por equipos ofensivos, defensivos y equilibrados

[347]:
```
club_games=pd.read_csv('club_games.csv')
club_games.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 122348 entries, 0 to 122347
Data columns (total 11 columns):
 #   Column                 Non-Null Count   Dtype
---  ------                 --------------   -----
 0   club_id                122348 non-null  int64
 1   game_id                122348 non-null  int64
 2   own_goals              122348 non-null  int64
 3   own_position           122348 non-null  int64
 4   own_manager_name       121026 non-null  object
 5   opponent_id            122348 non-null  int64
 6   opponent_goals         122348 non-null  int64
 7   opponent_position      122348 non-null  int64
 8   opponent_manager_name  121026 non-null  object
 9   hosting                122348 non-null  object
 10  is_win                 122348 non-null  int64
dtypes: int64(8), object(3)
memory usage: 10.3+ MB
```

[348]:
```
club_games
```

[348]:

	club_id	game_id	own_goals	own_position	own_manager_name	\
0	27	2229332	2	-1	Jupp Heynckes	\
1	131	2244388	3	-1	Tito Vilanova	
2	3709	2269557	0	-1	Luis García	
3	21322	2254432	1	-1	Pedro Buenaventura	
4	109	2221759	0	-1	Oscar Corrochano	
…	…	…	…	…	…	
122343	865	3828462	1	8	Albert Capellas	
122344	865	3828427	5	5	Henrik Jensen	
122345	1063	3828419	1	6	Jacob Friis	
122346	1177	3828414	2	6	Kent Nielsen	
122347	2414	3828447	0	7	Jens Berthel Askou	

	opponent_id	opponent_goals	opponent_position	opponent_manager_name	\
0	16	1	-1	Jürgen Klopp	\
1	418	2	-1	José Mourinho	
2	4032	0	-1	Claudio Barragán	
3	7077	0	-1	Pato	
4	27	4	-1	Jupp Heynckes	
…	…	…	…	…	

```
122343         2778                1                   1     Flemming Pedersen
122344          173                1                  12            Andreas Alm
122345          678                3                   5             Uwe Rösler
122346          369                2                   5   Freyr Alexandersson
122347          173                1                  11            Andreas Alm

       hosting  is_win
0         Home       1
1         Home       1
2         Home       0
3         Home       1
4         Home       0
…          …     …
122343    Away       0
122344    Away       1
122345    Away       0
122346    Away       0
122347    Away       0

[122348 rows x 11 columns]
```

[349]:
```
club_games=club_games[club_games['hosting']=='Home'].reset_index(drop=True)
```

[350]:
```
#vamos a crear variables que nos permitan analizar cada partido de mejor manera
```

[351]:
```
club_games['dif_goals_loc']=club_games['own_goals']-club_games['opponent_goals']
club_games['dif_goals_visit']=club_games['opponent_goals']-club_games['own_goals']
club_games['is_draw']=club_games['dif_goals_loc'].map(lambda x:1 if x==0 else 0)
club_games['is_win_visit']=club_games['dif_goals_loc'].map(lambda x:1 if x<0␣
 ↪else 0)
club_games['Total_goals']=club_games['own_goals']-club_games['opponent_goals']
# variable si el equipo gano por mas de 2, 3 y 4
club_games['is_win_over_2_local']=club_games['dif_goals_loc'].map(lambda x:1 if␣
 ↪x>2 else 0)
club_games['is_win_over_2_visit']=club_games['dif_goals_visit'].map(lambda x:1␣
 ↪if x>2 else 0)
club_games['is_win_over_3_local']=club_games['dif_goals_loc'].map(lambda x:1 if␣
 ↪x>3 else 0)
club_games['is_win_over_3_visit']=club_games['dif_goals_visit'].map(lambda x:1␣
 ↪if x>3 else 0)
club_games['is_win_over_4_local']=club_games['dif_goals_loc'].map(lambda x:1 if␣
 ↪x>4 else 0)
club_games['is_win_over_4_visit']=club_games['dif_goals_visit'].map(lambda x:1␣
 ↪if x>4 else 0)
# variable si el equipo perdio por mas de 2, 3 y 4
```

```
club_games['is_lost_over_2_local']=club_games['dif_goals_visit'].map(lambda x:1␣
 ↪if x>2 else 0)
club_games['is_lost_over_2_visit']=club_games['dif_goals_loc'].map(lambda x:1␣
 ↪if x>2 else 0)
club_games['is_lost_over_3_local']=club_games['dif_goals_visit'].map(lambda x:1␣
 ↪if x>3 else 0)
club_games['is_lost_over_3_visit']=club_games['dif_goals_loc'].map(lambda x:1␣
 ↪if x>3 else 0)
club_games['is_lost_over_4_local']=club_games['dif_goals_visit'].map(lambda x:1␣
 ↪if x>4 else 0)
club_games['is_lost_over_4_visit']=club_games['dif_goals_loc'].map(lambda x:1␣
 ↪if x>4 else 0)
```

[352]:
```
#agrupando variables del local
agg_col = 'club_id'
agg_funcs = {
    'own_goals': ['count','sum', 'min', 'max', 'mean','median'],
    'own_position':['min', 'max', 'mean','median'],
    'is_win':['mean'],
    'dif_goals_loc':['sum', 'min', 'max', 'mean','median'],
    'is_draw':'mean',
    'Total_goals':['sum', 'min', 'max', 'mean','median'],
    'is_win_over_2_local':'mean',
    'is_win_over_3_local':'mean',
    'is_win_over_4_local':'mean',
    'is_lost_over_2_local':'mean',
    'is_lost_over_3_local':'mean',
    'is_lost_over_4_local':'mean',

    }

team_local = club_games.groupby(agg_col).agg(agg_funcs)
team_local['diferent_manager']=club_games.groupby(agg_col)['own_manager_name'].
 ↪nunique()
team_local.sample(5)
```

[352]:
```
         own_goals                                  own_position
             count sum min max      mean median          min max  mean median
club_id
4313            11  17   0   5  1.545455    1.0           -1  -1 -1.00   -1.0  \
55686           20  31   0   5  1.550000    1.5           -1  12  7.25    8.0
22333            1   2   2   2  2.000000    2.0           -1  -1 -1.00   -1.0
5834             4   3   0   2  0.750000    0.5           -1  -1 -1.00   -1.0
13642           10  16   0   5  1.600000    1.0           -1  -1 -1.00   -1.0

          … Total_goals            is_win_over_2_local is_win_over_3_local
```

```
         …         max mean median                mean                mean
club_id  …
4313     …           4  0.0    0.0            0.181818            0.090909  \
55686    …           4 -0.7   -0.5            0.050000            0.050000
22333    …           2  2.0    2.0            0.000000            0.000000
5834     …          -3 -6.5   -4.0            0.000000            0.000000
13642    …           5  0.2    0.0            0.200000            0.100000

        is_win_over_4_local is_lost_over_2_local is_lost_over_3_local
                       mean                 mean                mean
club_id
4313                    0.0             0.090909                0.00  \
55686                   0.0             0.250000                0.10
22333                   0.0             0.000000                0.00
5834                    0.0             1.000000                0.75
13642                   0.1             0.100000                0.00

        is_lost_over_4_local diferent_manager
                        mean
club_id
4313                    0.00                7
55686                   0.10                4
22333                   0.00                1
5834                    0.25                0
13642                   0.00                3

[5 rows x 29 columns]
```

[353]:
```
#agrupando variables del local
agg_col = 'opponent_id'
agg_funcs = {
    'opponent_goals': ['count','sum', 'min', 'max', 'mean','median'],
    'opponent_position':['min', 'max', 'mean','median'],
    'is_win_visit':['mean'],
    'dif_goals_visit':['sum', 'min', 'max', 'mean','median'],
    'is_draw':'mean',
    'Total_goals':['sum', 'min', 'max', 'mean','median'],
    'is_win_over_2_visit':'mean',
    'is_win_over_3_visit':'mean',
    'is_win_over_4_visit':'mean',
    'is_lost_over_2_visit':'mean',
    'is_lost_over_3_visit':'mean',
    'is_lost_over_4_visit':'mean',

    }

team_visit = club_games.groupby(agg_col).agg(agg_funcs)
```

```
team_visit.sample(10)
```

[353]:

	opponent_goals						opponent_position	…	Total_goals							
	count	sum	min	max	mean	median	min	…	max	mean	median	…	min	max	mean	median
opponent_id								…				…				
41127	4	13	1	4	3.250000	4.0	-1	…	-1	-1.000000	-1.0	…	-3	2	-1.250000	-2.0
8322	4	1	0	1	0.250000	0.0	-1	…	-1	-1.000000	-1.0	…	1	2	1.250000	1.0
1423	38	31	0	3	0.815789	1.0	5	…	20	13.526316	13.0	…	-2	5	0.921053	1.0
371	282	588	0	9	2.085106	2.0	-1	…	10	0.652482	1.0	…	-9	7	-1.024823	-1.0
24898	5	13	0	6	2.600000	1.0	-1	…	-1	-1.000000	-1.0	…	-1	1	0.200000	1.0
19934	4	7	1	3	1.750000	1.5	-1	…	-1	-1.000000	-1.0	…	-1	2	0.250000	0.0
17980	1	1	1	1	1.000000	1.0	-1	…	-1	-1.000000	-1.0	…	2	2	2.000000	2.0
4603	92	70	0	4	0.760870	1.0	-1	…	16	8.217391	11.0	…	-4	5	0.934783	1.0
38405	1	1	1	1	1.000000	1.0	-1	…	-1	-1.000000	-1.0	…	1	1	1.000000	1.0
31276	3	7	1	4	2.333333	2.0	-1	…	-1	-1.000000	-1.0	…	-3	3	0.333333	1.0

	is_win_over_2_visit	is_win_over_3_visit	is_win_over_4_visit
	mean	mean	mean
opponent_id			
41127	0.250000	0.000000	0.000000
8322	0.000000	0.000000	0.000000
1423	0.000000	0.000000	0.000000
371	0.230496	0.099291	0.039007
24898	0.000000	0.000000	0.000000
19934	0.000000	0.000000	0.000000
17980	0.000000	0.000000	0.000000
4603	0.032609	0.010870	0.000000
38405	0.000000	0.000000	0.000000
31276	0.333333	0.000000	0.000000

	is_lost_over_2_visit	is_lost_over_3_visit	is_lost_over_4_visit
	mean	mean	mean
opponent_id			

```
41127                0.000000              0.000000              0.000000
8322                 0.000000              0.000000              0.000000
1423                 0.105263              0.052632              0.026316
371                  0.031915              0.017730              0.010638
24898                0.000000              0.000000              0.000000
19934                0.000000              0.000000              0.000000
17980                0.000000              0.000000              0.000000
4603                 0.173913              0.054348              0.010870
38405                0.000000              0.000000              0.000000
31276                0.333333              0.000000              0.000000

[10 rows x 28 columns]
```

[354]:
```
team_local.columns = [f"{col[0]}_{col[1]}" for col in team_local.columns]
team_visit.columns = [f"{col[0]}_{col[1]}" for col in team_visit.columns]
print(team_local.info())
print(team_visit.info())
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 2295 entries, 1 to 102251
Data columns (total 29 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   own_goals_count            2295 non-null   int64
 1   own_goals_sum              2295 non-null   int64
 2   own_goals_min              2295 non-null   int64
 3   own_goals_max              2295 non-null   int64
 4   own_goals_mean             2295 non-null   float64
 5   own_goals_median           2295 non-null   float64
 6   own_position_min           2295 non-null   int64
 7   own_position_max           2295 non-null   int64
 8   own_position_mean          2295 non-null   float64
 9   own_position_median        2295 non-null   float64
 10  is_win_mean                2295 non-null   float64
 11  dif_goals_loc_sum          2295 non-null   int64
 12  dif_goals_loc_min          2295 non-null   int64
 13  dif_goals_loc_max          2295 non-null   int64
 14  dif_goals_loc_mean         2295 non-null   float64
 15  dif_goals_loc_median       2295 non-null   float64
 16  is_draw_mean               2295 non-null   float64
 17  Total_goals_sum            2295 non-null   int64
 18  Total_goals_min            2295 non-null   int64
 19  Total_goals_max            2295 non-null   int64
 20  Total_goals_mean           2295 non-null   float64
 21  Total_goals_median         2295 non-null   float64
 22  is_win_over_2_local_mean   2295 non-null   float64
 23  is_win_over_3_local_mean   2295 non-null   float64
```

```
 24  is_win_over_4_local_mean   2295 non-null   float64
 25  is_lost_over_2_local_mean  2295 non-null   float64
 26  is_lost_over_3_local_mean  2295 non-null   float64
 27  is_lost_over_4_local_mean  2295 non-null   float64
 28  diferent_manager_          2295 non-null   int64
dtypes: float64(16), int64(13)
memory usage: 537.9 KB
None
<class 'pandas.core.frame.DataFrame'>
Index: 2042 entries, 2 to 102261
Data columns (total 28 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   opponent_goals_count       2042 non-null   int64
 1   opponent_goals_sum         2042 non-null   int64
 2   opponent_goals_min         2042 non-null   int64
 3   opponent_goals_max         2042 non-null   int64
 4   opponent_goals_mean        2042 non-null   float64
 5   opponent_goals_median      2042 non-null   float64
 6   opponent_position_min      2042 non-null   int64
 7   opponent_position_max      2042 non-null   int64
 8   opponent_position_mean     2042 non-null   float64
 9   opponent_position_median   2042 non-null   float64
 10  is_win_visit_mean          2042 non-null   float64
 11  dif_goals_visit_sum        2042 non-null   int64
 12  dif_goals_visit_min        2042 non-null   int64
 13  dif_goals_visit_max        2042 non-null   int64
 14  dif_goals_visit_mean       2042 non-null   float64
 15  dif_goals_visit_median     2042 non-null   float64
 16  is_draw_mean               2042 non-null   float64
 17  Total_goals_sum            2042 non-null   int64
 18  Total_goals_min            2042 non-null   int64
 19  Total_goals_max            2042 non-null   int64
 20  Total_goals_mean           2042 non-null   float64
 21  Total_goals_median         2042 non-null   float64
 22  is_win_over_2_visit_mean   2042 non-null   float64
 23  is_win_over_3_visit_mean   2042 non-null   float64
 24  is_win_over_4_visit_mean   2042 non-null   float64
 25  is_lost_over_2_visit_mean  2042 non-null   float64
 26  is_lost_over_3_visit_mean  2042 non-null   float64
 27  is_lost_over_4_visit_mean  2042 non-null   float64
dtypes: float64(16), int64(12)
memory usage: 462.6 KB
None
```

[355]:
```
team_local
```

[355]:

club_id	own_goals_count	own_goals_sum	own_goals_min	own_goals_max
1	10	25	0	8
2	10	24	0	8
3	139	199	0	7
4	58	72	0	6
5	260	431	0	6
…	…	…	…	…
101818	1	1	1	1
101819	1	0	0	0
102249	1	1	1	1
102250	1	0	0	0
102251	2	3	1	2

club_id	own_goals_mean	own_goals_median	own_position_min	own_position_max
1	2.500000	2.5	-1	-1
2	2.400000	2.0	-1	-1
3	1.431655	1.0	-1	18
4	1.241379	1.0	-1	18
5	1.657692	1.0	-1	15
…	…	…	…	…
101818	1.000000	1.0	-1	-1
101819	0.000000	0.0	-1	-1
102249	1.000000	1.0	-1	-1
102250	0.000000	0.0	-1	-1
102251	1.500000	1.5	-1	-1

club_id	own_position_mean	own_position_median	…	Total_goals_max
1	-1.000000	-1.0	…	2
2	-1.000000	-1.0	…	2
3	10.553957	11.0	…	6
4	12.620690	14.0	…	4
5	4.469231	4.0	…	6
…	…	…	…	…
101818	-1.000000	-1.0	…	-1
101819	-1.000000	-1.0	…	-7
102249	-1.000000	-1.0	…	-3
102250	-1.000000	-1.0	…	-3
102251	-1.000000	-1.0	…	1

club_id	Total_goals_mean	Total_goals_median	is_win_over_2_local_mean
1	-0.300000	1.0	0.000000
2	-0.600000	-1.0	0.000000
3	0.043165	0.0	0.071942

```
4              -0.362069                0.0                  0.086207
5               0.688462                1.0                  0.123077
…                    …                  …                       …
101818         -1.000000               -1.0                  0.000000
101819         -7.000000               -7.0                  0.000000
102249         -3.000000               -3.0                  0.000000
102250         -3.000000               -3.0                  0.000000
102251         -1.000000               -1.0                  0.000000

         is_win_over_3_local_mean  is_win_over_4_local_mean
club_id
1                        0.000000                  0.000000  \
2                        0.000000                  0.000000
3                        0.014388                  0.007194
4                        0.034483                  0.000000
5                        0.038462                  0.007692
…                           …                         …
101818                   0.000000                  0.000000
101819                   0.000000                  0.000000
102249                   0.000000                  0.000000
102250                   0.000000                  0.000000
102251                   0.000000                  0.000000

         is_lost_over_2_local_mean  is_lost_over_3_local_mean
club_id
1                         0.200000                   0.100000  \
2                         0.200000                   0.100000
3                         0.050360                   0.021583
4                         0.103448                   0.034483
5                         0.026923                   0.003846
…                            …                          …
101818                    0.000000                   0.000000
101819                    1.000000                   1.000000
102249                    1.000000                   0.000000
102250                    1.000000                   0.000000
102251                    0.500000                   0.000000

         is_lost_over_4_local_mean  diferent_manager_
club_id
1                         0.100000                  4
2                         0.100000                  7
3                         0.000000                  7
4                         0.017241                 10
5                         0.000000                 10
…                            …                     …
101818                    0.000000                  0
101819                    1.000000                  0
```

```
102249                    0.000000                 0
102250                    0.000000                 0
102251                    0.000000                 0

[2295 rows x 29 columns]
```

[356]:
```
team_visit
```

[356]:
```
             opponent_goals_count  opponent_goals_sum  opponent_goals_min
opponent_id
2                              12                  32                   0  \
3                             160                 232                   0
4                              66                  86                   0
5                             250                 396                   0
6                              17                  13                   0
…                             …                   …                    …
101808                         1                   2                   2
101815                         1                   1                   1
102243                         1                   0                   0
102249                         1                   3                   3
102261                         1                   1                   1

             opponent_goals_max  opponent_goals_mean  opponent_goals_median
opponent_id
2                             7             2.666667                   3.0  \
3                             9             1.450000                   1.0
4                             7             1.303030                   1.0
5                            11             1.584000                   1.0
6                             2             0.764706                   1.0
…                            …              …                          …
101808                        2             2.000000                   2.0
101815                        1             1.000000                   1.0
102243                        0             0.000000                   0.0
102249                        3             3.000000                   3.0
102261                        1             1.000000                   1.0

             opponent_position_min  opponent_position_max
opponent_id
2                               -1                     -1  \
3                               -1                     18
4                               -1                     18
5                               -1                     19
6                               13                     18
…                              …                      …
101808                          -1                     -1
101815                          -1                     -1
102243                          -1                     -1
```

```
102249                          -1                    -1
102261                          -1                    -1

             opponent_position_mean  opponent_position_median  …
opponent_id                                                   …
2                         -1.000000                      -1.0  …  \
3                          9.193750                      10.0  …
4                         11.090909                      14.0  …
5                          4.972000                       5.0  …
6                         16.882353                      17.0  …
…                               …                         …  …
101808                    -1.000000                      -1.0  …
101815                    -1.000000                      -1.0  …
102243                    -1.000000                      -1.0  …
102249                    -1.000000                      -1.0  …
102261                    -1.000000                      -1.0  …

             Total_goals_min  Total_goals_max  Total_goals_mean
opponent_id
2                         -7                4         -0.666667  \
3                         -8                6          0.337500
4                         -2                7          0.893939
5                         -7                5         -0.260000
6                         -1                4          1.000000
…                      …              …                …
101808                    -2               -2         -2.000000
101815                     3                3          3.000000
102243                     1                1          1.000000
102249                    -3               -3         -3.000000
102261                     1                1          1.000000

             Total_goals_median  is_win_over_2_visit_mean
opponent_id
2                          -1.0                     0.250  \
3                           0.0                     0.075
4                           1.0                     0.000
5                           0.0                     0.072
6                           1.0                     0.000
…                           …                       …
101808                     -2.0                     0.000
101815                      3.0                     0.000
102243                      1.0                     0.000
102249                     -3.0                     1.000
102261                      1.0                     0.000

             is_win_over_3_visit_mean  is_win_over_4_visit_mean
opponent_id
```

```
2                              0.166667                   0.083333  \
3                              0.050000                   0.037500
4                              0.000000                   0.000000
5                              0.020000                   0.004000
6                              0.000000                   0.000000
…                                   …                          …
101808                         0.000000                   0.000000
101815                         0.000000                   0.000000
102243                         0.000000                   0.000000
102249                         0.000000                   0.000000
102261                         0.000000                   0.000000

             is_lost_over_2_visit_mean  is_lost_over_3_visit_mean
opponent_id
2                              0.166667                   0.166667  \
3                              0.137500                   0.087500
4                              0.196970                   0.060606
5                              0.048000                   0.016000
6                              0.176471                   0.058824
…                                   …                          …
101808                         0.000000                   0.000000
101815                         1.000000                   0.000000
102243                         0.000000                   0.000000
102249                         0.000000                   0.000000
102261                         0.000000                   0.000000

             is_lost_over_4_visit_mean
opponent_id
2                              0.000000
3                              0.043750
4                              0.030303
5                              0.004000
6                              0.000000
…                                   …
101808                         0.000000
101815                         0.000000
102243                         0.000000
102249                         0.000000
102261                         0.000000

[2042 rows x 28 columns]
```

[357]:
```
team=pd.merge(team_local,team_visit,left_index=True, right_index=True,␣
↪how='inner')
team
```

[357]:

	own_goals_count	own_goals_sum	own_goals_min	own_goals_max	\
2	10	24	0	8	\
3	139	199	0	7	
4	58	72	0	6	
5	260	431	0	6	
6	17	20	0	3	
…	…	…	…	…	
101576	1	3	3	3	
101577	2	13	4	9	
101582	1	2	2	2	
101808	2	3	0	3	
102249	1	1	1	1	

	own_goals_mean	own_goals_median	own_position_min	own_position_max	
2	2.400000	2.0	-1	-1	\
3	1.431655	1.0	-1	18	
4	1.241379	1.0	-1	18	
5	1.657692	1.0	-1	15	
6	1.176471	1.0	12	18	
…	…	…	…	…	
101576	3.000000	3.0	-1	-1	
101577	6.500000	6.5	-1	-1	
101582	2.000000	2.0	-1	-1	
101808	1.500000	1.5	-1	-1	
102249	1.000000	1.0	-1	-1	

	own_position_mean	own_position_median	…	Total_goals_min_y	
2	-1.000000	-1.0	…	-7	\
3	10.553957	11.0	…	-8	
4	12.620690	14.0	…	-2	
5	4.469231	4.0	…	-7	
6	16.470588	17.0	…	-1	
…	…	…	…	…	
101576	-1.000000	-1.0	…	2	
101577	-1.000000	-1.0	…	-1	
101582	-1.000000	-1.0	…	4	
101808	-1.000000	-1.0	…	-2	
102249	-1.000000	-1.0	…	-3	

	Total_goals_max_y	Total_goals_mean_y	Total_goals_median_y	
2	4	-0.666667	-1.0	\
3	6	0.337500	0.0	
4	7	0.893939	1.0	
5	5	-0.260000	0.0	
6	4	1.000000	1.0	
…	…	…	…	
101576	2	2.000000	2.0	

```
101577                -1          -1.000000                  -1.0
101582                 4           4.000000                   4.0
101808                -2          -2.000000                  -2.0
102249                -3          -3.000000                  -3.0

        is_win_over_2_visit_mean  is_win_over_3_visit_mean
2                          0.250                  0.166667  \
3                          0.075                  0.050000
4                          0.000                  0.000000
5                          0.072                  0.020000
6                          0.000                  0.000000
…                              …                         …
101576                     0.000                  0.000000
101577                     0.000                  0.000000
101582                     0.000                  0.000000
101808                     0.000                  0.000000
102249                     1.000                  0.000000

        is_win_over_4_visit_mean  is_lost_over_2_visit_mean
2                       0.083333                   0.166667  \
3                       0.037500                   0.137500
4                       0.000000                   0.196970
5                       0.004000                   0.048000
6                       0.000000                   0.176471
…                              …                         …
101576                  0.000000                   0.000000
101577                  0.000000                   0.000000
101582                  0.000000                   1.000000
101808                  0.000000                   0.000000
102249                  0.000000                   0.000000

        is_lost_over_3_visit_mean  is_lost_over_4_visit_mean
2                        0.166667                   0.000000
3                        0.087500                   0.043750
4                        0.060606                   0.030303
5                        0.016000                   0.004000
6                        0.058824                   0.000000
…                              …                         …
101576                   0.000000                   0.000000
101577                   0.000000                   0.000000
101582                   1.000000                   0.000000
101808                   0.000000                   0.000000
102249                   0.000000                   0.000000

[1779 rows x 57 columns]
```

[358]:
```
team=team[team['own_goals_count']>15]
team=team[team['opponent_goals_count']>15]
team
```

[358]:

```
       own_goals_count  own_goals_sum  own_goals_min  own_goals_max  \
3                  139            199              0              7
4                   58             72              0              6
5                  260            431              0              6
6                   17             20              0              3
10                  47             55              0              6
…                  …              …              …              …
55686               20             31              0              5
60551               52            104              0              8
60949               61             72              0              5
61825               37             41              0              7
68608              100            114              0              4

       own_goals_mean  own_goals_median  own_position_min  own_position_max  \
3            1.431655               1.0                -1                18
4            1.241379               1.0                -1                18
5            1.657692               1.0                -1                15
6            1.176471               1.0                12                18
10           1.170213               1.0                -1                17
…                 …                 …                 …                 …
55686        1.550000               1.5                -1                12
60551        2.000000               2.0                -1                13
60949        1.180328               1.0                -1                10
61825        1.108108               1.0                -1                15
68608        1.140000               1.0                -1                16

       own_position_mean  own_position_median  …  Total_goals_min_y  \
3               10.553957                 11.0  …                 -8
4               12.620690                 14.0  …                 -2
5                4.469231                  4.0  …                 -7
6               16.470588                 17.0  …                 -1
10              10.808511                 15.0  …                 -6
…                     …                    …  …                  …
55686            7.250000                  8.0  …                 -2
60551            4.807692                  5.0  …                 -3
60949            5.557377                  6.0  …                 -3
61825            9.864865                 12.0  …                 -1
68608            8.370000                 10.0  …                 -3

       Total_goals_max_y  Total_goals_mean_y  Total_goals_median_y  \
3                      6            0.337500                   0.0
4                      7            0.893939                   1.0
5                      5           -0.260000                   0.0
```

```
6                    4            1.000000                   1.0
10                   6            0.555556                   1.0
…                  …                …                      …
55686                3            0.652174                   0.0
60551                4           -0.137255                   0.0
60949                5            0.360656                   0.0
61825                4            1.054054                   1.0
68608                6            0.637255                   0.0

       is_win_over_2_visit_mean  is_win_over_3_visit_mean
3                      0.075000                  0.050000  \
4                      0.000000                  0.000000
5                      0.072000                  0.020000
6                      0.000000                  0.000000
10                     0.066667                  0.044444
…                         …                       …
55686                  0.000000                  0.000000
60551                  0.098039                  0.000000
60949                  0.016393                  0.000000
61825                  0.000000                  0.000000
68608                  0.009804                  0.000000

       is_win_over_4_visit_mean  is_lost_over_2_visit_mean
3                      0.037500                   0.137500  \
4                      0.000000                   0.196970
5                      0.004000                   0.048000
6                      0.000000                   0.176471
10                     0.044444                   0.155556
…                         …                        …
55686                  0.000000                   0.043478
60551                  0.000000                   0.098039
60949                  0.000000                   0.131148
61825                  0.000000                   0.216216
68608                  0.000000                   0.127451

       is_lost_over_3_visit_mean  is_lost_over_4_visit_mean
3                       0.087500                   0.043750
4                       0.060606                   0.030303
5                       0.016000                   0.004000
6                       0.058824                   0.000000
10                      0.111111                   0.066667
…                          …                        …
55686                   0.000000                   0.000000
60551                   0.019608                   0.000000
60949                   0.049180                   0.016393
61825                   0.054054                   0.000000
68608                   0.068627                   0.029412
```

```
[488 rows x 57 columns]
```

[359]:
```
clubs=pd.read_csv('clubs.csv')
clubs
```

[359]:
```
     club_id            club_code                 name
0       1032           fc-reading           Fc Reading  \
1       2323             orduspor             Orduspor
2       1387       acn-siena-1904       Acn Siena 1904
3       1071       wigan-athletic       Wigan Athletic
4       2703  spartak-vladikavkaz  Spartak Vladikavkaz
..       …                   …                    …
406     3725        akhmat-grozny        Akhmat Grozny
407       13      atletico-madrid      Atletico Madrid
408      368           fc-sevilla           Fc Sevilla
409      940           celta-vigo           Celta Vigo
410      331           ca-osasuna           Ca Osasuna

    domestic_competition_id  total_market_value  squad_size  average_age
0                       GB1               33.66          26         25.9  \
1                       TR1                 NaN           0          NaN
2                       IT1                4.32          30         26.2
3                       GB1               12.38          29         26.5
4                       RU1                 NaN           1         20.0
..                      …                   …           …           …
406                     RU1                 NaN          30         25.8
407                     ES1                 NaN          22         28.5
408                     ES1                 NaN          24         28.7
409                     ES1                 NaN          23         27.0
410                     ES1                 NaN          21         28.0

     foreigners_number  foreigners_percentage  national_team_players
0                   12                   46.2                      6  \
1                    0                    NaN                      0
2                    6                   20.0                      2
3                   14                   48.3                      6
4                    0                    NaN                      0
..                   …                     …                      …
406                 10                   33.3                      2
407                 15                   68.2                     17
408                 18                   75.0                     11
409                 11                   47.8                      6
410                  4                   19.0                      3

                   stadium_name  stadium_seats net_transfer_record
0    Select Car Leasing Stadium          24161             +£8.37m  \
```

```
1              19 Eylül Stadyum          11024                +-0
2               Artemio Franchi          15373              £-6Th.
3                    DW Stadium          25133             £-140Th.
4    Republican Stadium Spartak          32464                +-0
..                           …              …                  …
406                Akhmat-Arena          30200              €-4.40m
407       Civitas Metropolitano          68456              €-9.25m
408       Ramón Sánchez-Pizjuán          43883             +€62.10m
409             Abanca Balaídos          29000              €-5.15m
410                   El Sadar           23576              €-2.00m

          coach_name                                                 url
0     Brian McDermott  https://www.transfermarkt.co.uk/fc-reading/sta…
1        Héctor Cúper  https://www.transfermarkt.co.uk/orduspor/start…
2         Serse Cosmi  https://www.transfermarkt.co.uk/acn-siena-1904…
3    Roberto Martínez  https://www.transfermarkt.co.uk/wigan-athletic…
4    Vladimir Gazzaev  https://www.transfermarkt.co.uk/spartak-vladik…
..                …                                                  …
406              NaN  https://www.transfermarkt.co.uk/akhmat-grozny/…
407              NaN  https://www.transfermarkt.co.uk/atletico-madri…
408              NaN  https://www.transfermarkt.co.uk/fc-sevilla/sta…
409              NaN  https://www.transfermarkt.co.uk/celta-vigo/sta…
410              NaN  https://www.transfermarkt.co.uk/ca-osasuna/sta…

[411 rows x 15 columns]
```

[360]:
```
team=pd.merge(team,clubs,left_index=True, right_on='club_id', how='inner')
team
```

[360]:
```
     own_goals_count  own_goals_sum  own_goals_min  own_goals_max
323              139            199              0              7  \
98                58             72              0              6
217              260            431              0              6
52                17             20              0              3
156               47             55              0              6
..               …              …              …              …
253               20             31              0              5
261               52            104              0              8
184               61             72              0              5
251               37             41              0              7
30               100            114              0              4

     own_goals_mean  own_goals_median  own_position_min  own_position_max
323        1.431655               1.0                -1                18  \
98         1.241379               1.0                -1                18
217        1.657692               1.0                -1                15
52         1.176471               1.0                12                18
```

```
156        1.170213               1.0                 -1                 17
..              …                 …                   …                  …
253        1.550000               1.5                 -1                 12
261        2.000000               2.0                 -1                 13
184        1.180328               1.0                 -1                 10
251        1.108108               1.0                 -1                 15
30         1.140000               1.0                 -1                 16

     own_position_mean  own_position_median  …  squad_size  average_age
323          10.553957                 11.0  …          35         24.3  \
98           12.620690                 14.0  …          28         25.3
217           4.469231                  4.0  …          31         26.6
52           16.470588                 17.0  …          27         24.0
156          10.808511                 15.0  …          25         25.0
..                 …                    …  …         …           …
253           7.250000                  8.0  …          24         25.5
261           4.807692                  5.0  …          29         25.8
184           5.557377                  6.0  …          26         26.2
251           9.864865                 12.0  …          26         26.3
30            8.370000                 10.0  …          28         24.8

     foreigners_number  foreigners_percentage  national_team_players
323                 10                   28.6                     6  \
98                   5                   17.9                     1
217                 25                   80.6                    20
52                  11                   40.7                     1
156                 16                   64.0                     5
..                 …                      …                     …
253                  2                    8.3                     0
261                  9                   31.0                     6
184                 16                   61.5                     2
251                  1                    3.8                     0
30                   4                   14.3                     0

             stadium_name  stadium_seats  net_transfer_record
323   RheinEnergieSTADION          50000              +€5.52m  \
98    Max-Morlock-Stadion          50000              +£1.58m
217       Giuseppe Meazza          75923             €-35.62m
52    Yeni Adana Stadyumu          33000             +£270Th.
156           SchücoArena          26515              £-8.45m
..                    …              …                    …
253         NSK Olimpisky          70050                  +-0
261               Avangard         10640               +€650k
184  Panthessaliko Stadio          22700              +€1.27m
251         Stadion Minaj           2000               +€100k
30     Estádio do Restelo          19980                  +-0
```

```
          coach_name                                                 url
323              NaN  https://www.transfermarkt.co.uk/1-fc-koln/star…
98   Michael Köllner  https://www.transfermarkt.co.uk/1-fc-nurnberg/…
217              NaN  https://www.transfermarkt.co.uk/ac-mailand/sta…
52    Engin Ipekoglu  https://www.transfermarkt.co.uk/adanaspor/star…
156   Marco Kostmann  https://www.transfermarkt.co.uk/arminia-bielef…
..               …                                                  …
253              NaN  https://www.transfermarkt.co.uk/metalist-1925-…
261              NaN  https://www.transfermarkt.co.uk/sk-dnipro-1/st…
184              NaN  https://www.transfermarkt.co.uk/volos-nps/star…
251              NaN  https://www.transfermarkt.co.uk/fk-minaj/start…
30      Lito Vidigal  https://www.transfermarkt.co.uk/cf-os-belenens…

[405 rows x 72 columns]
```

[361]:
```
ls_cont=[]
for i in team.columns:
    if team[i].nunique()>2:
        if team[i].dtype!='object':
            ls_cont.append(i)
ls_cont
```

[361]:
```
['own_goals_count',
 'own_goals_sum',
 'own_goals_max',
 'own_goals_mean',
 'own_goals_median',
 'own_position_min',
 'own_position_max',
 'own_position_mean',
 'own_position_median',
 'is_win_mean',
 'dif_goals_loc_sum',
 'dif_goals_loc_min',
 'dif_goals_loc_max',
 'dif_goals_loc_mean',
 'dif_goals_loc_median',
 'is_draw_mean_x',
 'Total_goals_sum_x',
 'Total_goals_min_x',
 'Total_goals_max_x',
 'Total_goals_mean_x',
 'Total_goals_median_x',
 'is_win_over_2_local_mean',
 'is_win_over_3_local_mean',
 'is_win_over_4_local_mean',
 'is_lost_over_2_local_mean',
```

```
 'is_lost_over_3_local_mean',
 'is_lost_over_4_local_mean',
 'diferent_manager_',
 'opponent_goals_count',
 'opponent_goals_sum',
 'opponent_goals_max',
 'opponent_goals_mean',
 'opponent_goals_median',
 'opponent_position_min',
 'opponent_position_max',
 'opponent_position_mean',
 'opponent_position_median',
 'is_win_visit_mean',
 'dif_goals_visit_sum',
 'dif_goals_visit_min',
 'dif_goals_visit_max',
 'dif_goals_visit_mean',
 'dif_goals_visit_median',
 'is_draw_mean_y',
 'Total_goals_sum_y',
 'Total_goals_min_y',
 'Total_goals_max_y',
 'Total_goals_mean_y',
 'Total_goals_median_y',
 'is_win_over_2_visit_mean',
 'is_win_over_3_visit_mean',
 'is_win_over_4_visit_mean',
 'is_lost_over_2_visit_mean',
 'is_lost_over_3_visit_mean',
 'is_lost_over_4_visit_mean',
 'club_id',
 'total_market_value',
 'squad_size',
 'average_age',
 'foreigners_number',
 'foreigners_percentage',
 'national_team_players',
 'stadium_seats']
```

[362]:
```
tad=team[ls_cont]
```

[363]:
```
# remocion de variables con alto contenido de missings (mayor al 10%)
def remover_variables_nulas(df, umbral=0.5):
    # Calcula el porcentaje de valores nulos por columna
    porcentaje_nulos = df.isnull().mean()

    # Obtiene las variables que superan el umbral de valores nulos
```

```
    variables_a_remover = porcentaje_nulos[porcentaje_nulos > umbral].index.
 ↪tolist()

    # Elimina las variables del DataFrame
    df.drop(variables_a_remover, axis=1, inplace=True)
    print(f"las variables que se removieron son: {variables_a_remover}")
```

[364]:
```
remover_variables_nulas(tad)
```

```
las variables que se removieron son: ['total_market_value']
```

[365]:
```
#Validamos ahora para inf variables
inf_values = np.isinf(tad).sum().items()
inf_variables = [var for var, n_inf in inf_values if n_inf > 0]
inf_variables # no hay variables con infinitos
```

[365]:
```
[]
```

[366]:
```
corr_matrix = tad.corr()
ls_checked = []
ls_correlated = []
for col in corr_matrix.columns:
    ls_checked.append(col)
    ls_correlated += corr_matrix[(corr_matrix[col] >= 0.99) & (~corr_matrix.
 ↪index.isin(ls_checked))].index.tolist()
ls_correlated = list(set(ls_correlated))
ls_correlated = [variable for variable in ls_correlated]
print(f"las variables con correlacion de 0.999 con respecto a otra son:␣
 ↪{ls_correlated}") #  tenemos 1 varibale con correlaicon 0.999 respecto a␣
 ↪otra, se eliminara
tad=tad.drop(ls_correlated,axis=1)
```

```
las variables con correlacion de 0.999 con respecto a otra son:
['opponent_goals_count', 'Total_goals_min_x', 'Total_goals_max_x',
'Total_goals_mean_x', 'Total_goals_median_x', 'Total_goals_sum_x']
```

[367]:
```
def remover_variables_unarias(df, umbral=0.98):
    # Calcula el valor más común en cada columna
    valores_comunes = df.mode().iloc[0]

    # Obtiene las variables que superan el umbral de igualdad
    variables_a_remover = []
    for columna in df.columns:
        valor_comun = valores_comunes[columna]
        porcentaje_igual = (df[columna] == valor_comun).mean()
        if porcentaje_igual > umbral:
            variables_a_remover.append(columna)
```

```
    # Elimina las variables del DataFrame
    print(f"variables unarias a remover: {variables_a_remover}")
    df.drop(variables_a_remover, axis=1, inplace=True)
```

[368]:
```
remover_variables_unarias(tad, umbral=0.98)
```

```
variables unarias a remover: []
```

[369]:
```
# tratamiento de outliers
def detect_outlier(serie, method):
    if method == "iqr":
        q1 = serie.quantile(.25)
        q3 = serie.quantile(.75)
        iqr = q3-q1
        upper_fence = q3 + 1.5*iqr
        lower_fence = q1 - 1.5*iqr
    elif method == "z-score":
        mean = serie.mean()
        std = serie.std()
        upper_fence = mean + 3*std
        lower_fence = mean - 3*std
    else:
        upper_fence = serie.quantile(.99)
        lower_fence = serie.quantile(.01)
    return ~serie.between(lower_fence, upper_fence, inclusive="both")
```

[370]:
```
Xp = pd.concat(map(lambda column: detect_outlier(tad[column], "other").
 ↪rename(f"{column}_ol"), tad.columns), axis=1)
```

[371]:
```
shape_old = tad.shape
shape_new=Xp[Xp.mean(axis=1)<0.2].shape #renglones que tengan menos del 30% de␣
 ↪variables detectadas como outlier
shape_new[0] / shape_old[0] # proporcion de tamaño del nuevo data frame, sin␣
 ↪valores extremos
```

[371]:
```
0.9901234567901235
```

[372]:
```
#vamos a eliminar los outliers
df2 =tad[Xp.mean(axis=1)<0.2].reset_index()
```

[373]:
```
df2.info() # no es necesario imputar valores nulos
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 401 entries, 0 to 400
Data columns (total 57 columns):
 #   Column                      Non-Null Count  Dtype
```

```
---  ------                      --------------  -----
 0   index                       401 non-null    int64
 1   own_goals_count             401 non-null    int64
 2   own_goals_sum               401 non-null    int64
 3   own_goals_max               401 non-null    int64
 4   own_goals_mean              401 non-null    float64
 5   own_goals_median            401 non-null    float64
 6   own_position_min            401 non-null    int64
 7   own_position_max            401 non-null    int64
 8   own_position_mean           401 non-null    float64
 9   own_position_median         401 non-null    float64
 10  is_win_mean                 401 non-null    float64
 11  dif_goals_loc_sum           401 non-null    int64
 12  dif_goals_loc_min           401 non-null    int64
 13  dif_goals_loc_max           401 non-null    int64
 14  dif_goals_loc_mean          401 non-null    float64
 15  dif_goals_loc_median        401 non-null    float64
 16  is_draw_mean_x              401 non-null    float64
 17  is_win_over_2_local_mean    401 non-null    float64
 18  is_win_over_3_local_mean    401 non-null    float64
 19  is_win_over_4_local_mean    401 non-null    float64
 20  is_lost_over_2_local_mean   401 non-null    float64
 21  is_lost_over_3_local_mean   401 non-null    float64
 22  is_lost_over_4_local_mean   401 non-null    float64
 23  diferent_manager_           401 non-null    int64
 24  opponent_goals_sum          401 non-null    int64
 25  opponent_goals_max          401 non-null    int64
 26  opponent_goals_mean         401 non-null    float64
 27  opponent_goals_median       401 non-null    float64
 28  opponent_position_min       401 non-null    int64
 29  opponent_position_max       401 non-null    int64
 30  opponent_position_mean      401 non-null    float64
 31  opponent_position_median    401 non-null    float64
 32  is_win_visit_mean           401 non-null    float64
 33  dif_goals_visit_sum         401 non-null    int64
 34  dif_goals_visit_min         401 non-null    int64
 35  dif_goals_visit_max         401 non-null    int64
 36  dif_goals_visit_mean        401 non-null    float64
 37  dif_goals_visit_median      401 non-null    float64
 38  is_draw_mean_y              401 non-null    float64
 39  Total_goals_sum_y           401 non-null    int64
 40  Total_goals_min_y           401 non-null    int64
 41  Total_goals_max_y           401 non-null    int64
 42  Total_goals_mean_y          401 non-null    float64
 43  Total_goals_median_y        401 non-null    float64
 44  is_win_over_2_visit_mean    401 non-null    float64
 45  is_win_over_3_visit_mean    401 non-null    float64
 46  is_win_over_4_visit_mean    401 non-null    float64
```

```
 47  is_lost_over_2_visit_mean  401 non-null    float64
 48  is_lost_over_3_visit_mean  401 non-null    float64
 49  is_lost_over_4_visit_mean  401 non-null    float64
 50  club_id                    401 non-null    int64
 51  squad_size                 401 non-null    int64
 52  average_age                374 non-null    float64
 53  foreigners_number          401 non-null    int64
 54  foreigners_percentage      363 non-null    float64
 55  national_team_players      401 non-null    int64
 56  stadium_seats              401 non-null    int64
dtypes: float64(32), int64(25)
memory usage: 178.7 KB
```

[374]:
```
df2=df2.drop(['index','average_age','foreigners_percentage','club_id'],axis=1).
↪reset_index(drop=True)
```

[375]:
```
# EDA post limpieza

for variable in df2.columns:
    fig = px.histogram(df2, x=variable, title=f"{variable}'s histogram")
    fig.update_layout(
        autosize=False,
        width=500,  # Ajusta el ancho según tus necesidades
        height=400,  # Ajusta la altura según tus necesidades
    )
    fig.show()
```

[376]:
```
#MDS
mm = MinMaxScaler()
Xmm = pd.DataFrame(mm.fit_transform(df2), columns=df2.columns)
```

[377]:
```
Xmm.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 401 entries, 0 to 400
Data columns (total 53 columns):
 #   Column                     Non-Null Count  Dtype
---  ------                     --------------  -----
 0   own_goals_count            401 non-null    float64
 1   own_goals_sum              401 non-null    float64
 2   own_goals_max              401 non-null    float64
 3   own_goals_mean             401 non-null    float64
 4   own_goals_median           401 non-null    float64
 5   own_position_min           401 non-null    float64
 6   own_position_max           401 non-null    float64
 7   own_position_mean          401 non-null    float64
 8   own_position_median        401 non-null    float64
 9   is_win_mean                401 non-null    float64
```

```
 10  dif_goals_loc_sum          401 non-null    float64
 11  dif_goals_loc_min          401 non-null    float64
 12  dif_goals_loc_max          401 non-null    float64
 13  dif_goals_loc_mean         401 non-null    float64
 14  dif_goals_loc_median       401 non-null    float64
 15  is_draw_mean_x             401 non-null    float64
 16  is_win_over_2_local_mean   401 non-null    float64
 17  is_win_over_3_local_mean   401 non-null    float64
 18  is_win_over_4_local_mean   401 non-null    float64
 19  is_lost_over_2_local_mean  401 non-null    float64
 20  is_lost_over_3_local_mean  401 non-null    float64
 21  is_lost_over_4_local_mean  401 non-null    float64
 22  diferent_manager_          401 non-null    float64
 23  opponent_goals_sum         401 non-null    float64
 24  opponent_goals_max         401 non-null    float64
 25  opponent_goals_mean        401 non-null    float64
 26  opponent_goals_median      401 non-null    float64
 27  opponent_position_min      401 non-null    float64
 28  opponent_position_max      401 non-null    float64
 29  opponent_position_mean     401 non-null    float64
 30  opponent_position_median   401 non-null    float64
 31  is_win_visit_mean          401 non-null    float64
 32  dif_goals_visit_sum        401 non-null    float64
 33  dif_goals_visit_min        401 non-null    float64
 34  dif_goals_visit_max        401 non-null    float64
 35  dif_goals_visit_mean       401 non-null    float64
 36  dif_goals_visit_median     401 non-null    float64
 37  is_draw_mean_y             401 non-null    float64
 38  Total_goals_sum_y          401 non-null    float64
 39  Total_goals_min_y          401 non-null    float64
 40  Total_goals_max_y          401 non-null    float64
 41  Total_goals_mean_y         401 non-null    float64
 42  Total_goals_median_y       401 non-null    float64
 43  is_win_over_2_visit_mean   401 non-null    float64
 44  is_win_over_3_visit_mean   401 non-null    float64
 45  is_win_over_4_visit_mean   401 non-null    float64
 46  is_lost_over_2_visit_mean  401 non-null    float64
 47  is_lost_over_3_visit_mean  401 non-null    float64
 48  is_lost_over_4_visit_mean  401 non-null    float64
 49  squad_size                 401 non-null    float64
 50  foreigners_number          401 non-null    float64
 51  national_team_players      401 non-null    float64
 52  stadium_seats              401 non-null    float64
dtypes: float64(53)
memory usage: 166.2 KB
```

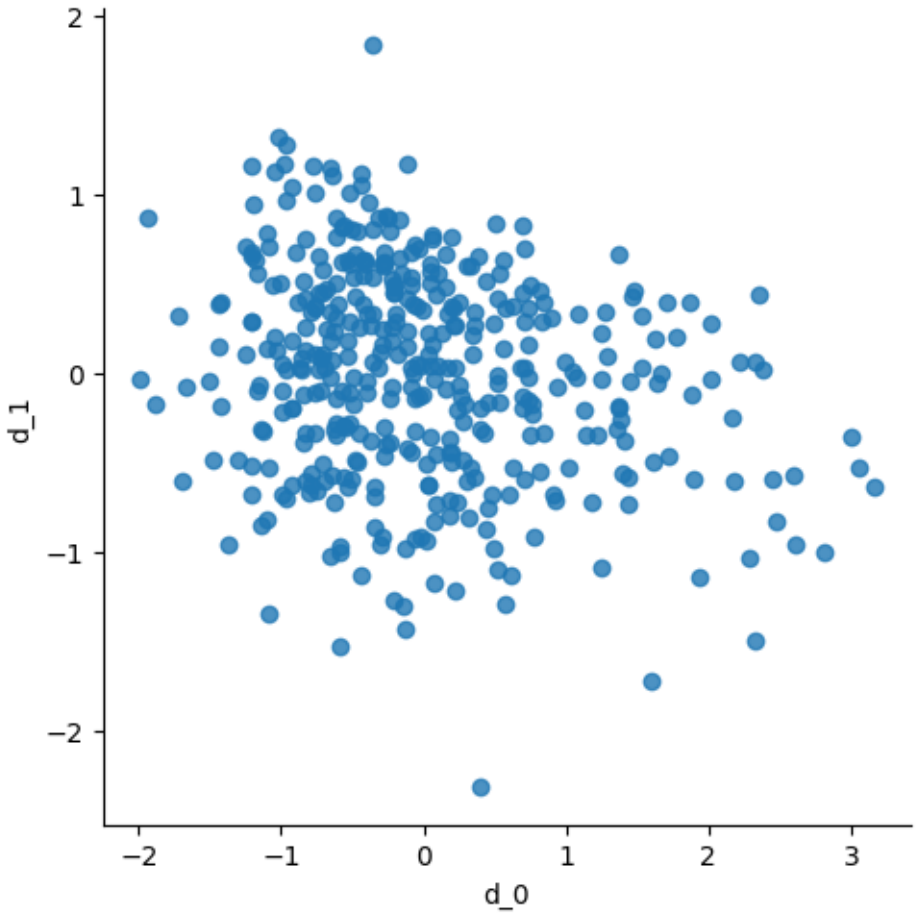
2 MDS

[378]:
```
j=3
mds2 = MDS(j)
Xmm_sample=Xmm
Xmds_sample = pd.DataFrame(mds2.fit_transform(Xmm_sample), columns=[f'd_{i}'␣
 ↪for i in range(j)])
Xmds_sample.head(3)
Xmds_modeling=Xmds_sample.copy()
```

```
/home/carlos/Documentos/diplomado/modulo 1/myenv/lib/python3.10/site-
packages/sklearn/manifold/_mds.py:299: FutureWarning:

The default value of `normalized_stress` will change to `'auto'` in version 1.4.
To suppress this warning, manually set the value of `normalized_stress`.
```

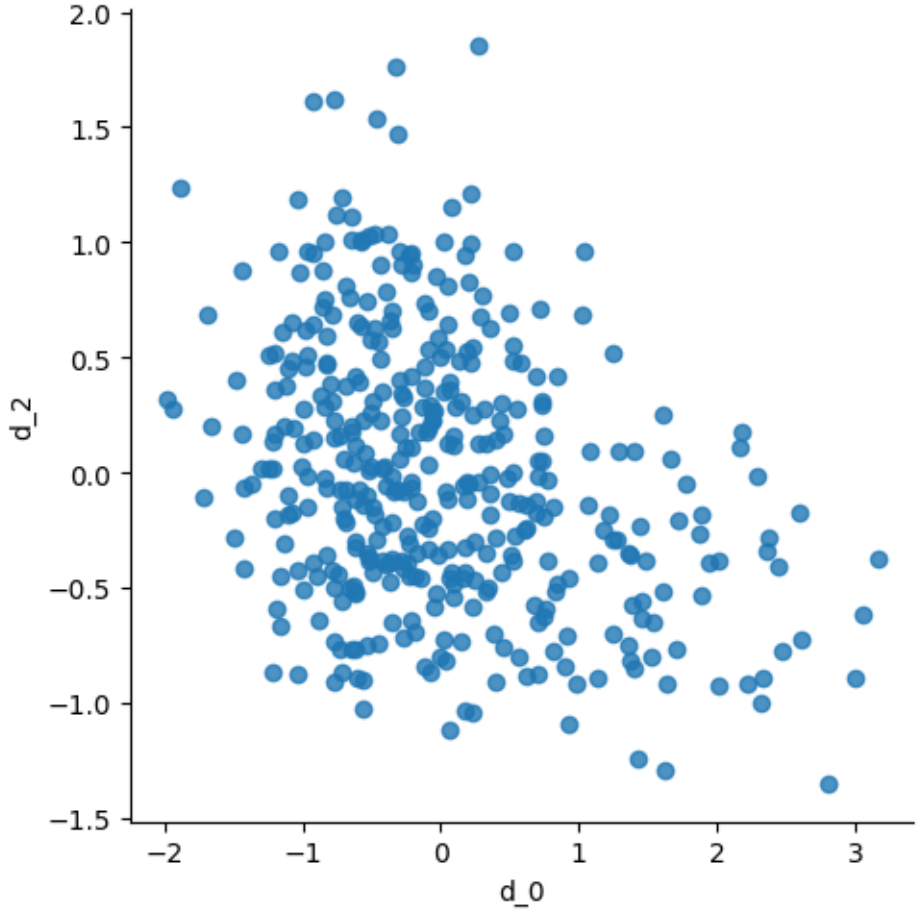
[379]:
```
sns.lmplot(data=Xmds_sample, x='d_0', y='d_1', fit_reg=False)
```

[379]:
```
<seaborn.axisgrid.FacetGrid at 0x7fdeba820580>
```



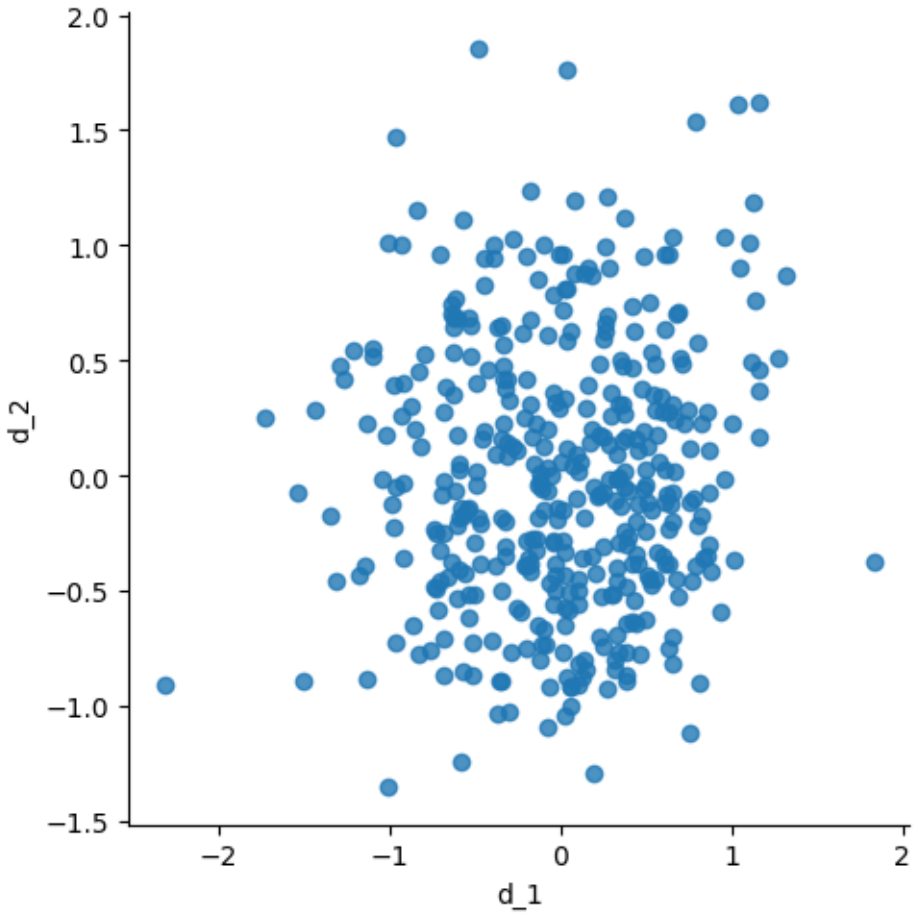
[380]:
```
sns.lmplot(data=Xmds_sample, x='d_0', y='d_2', fit_reg=False)
```

[380]: <seaborn.axisgrid.FacetGrid at 0x7fdebaa3b550>



[381]:
```
sns.lmplot(data=Xmds_sample, x='d_1', y='d_2', fit_reg=False)
```

[381]: <seaborn.axisgrid.FacetGrid at 0x7fdebaa3bca0>



3 Inercia

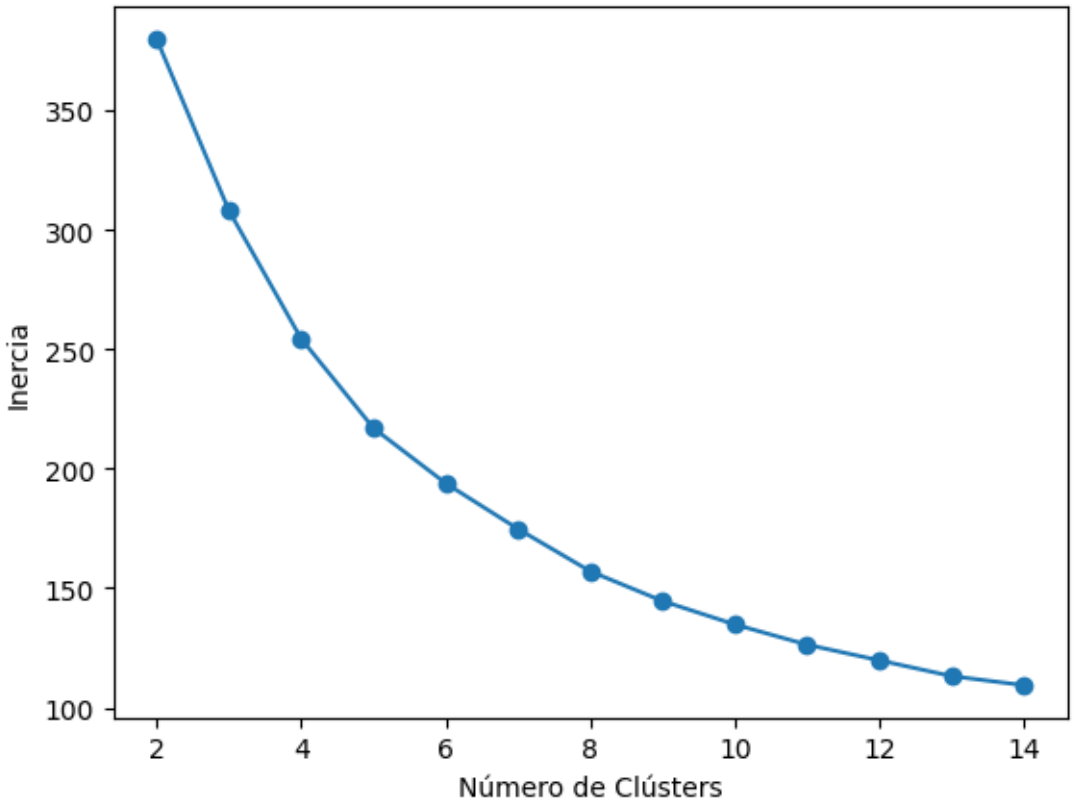
[382]:
```
inertia = []
for k in range(2, 15):
    kmeans = KMeans(n_clusters=k)
    kmeans.fit(Xmds_sample)
    inertia.append(kmeans.inertia_)

plt.plot(range(2, 15), inertia, marker='o')
plt.xlabel('Número de Clústers')
plt.ylabel('Inercia')
```

```
/home/carlos/Documentos/diplomado/modulo 1/myenv/lib/python3.10/site-
packages/sklearn/cluster/_kmeans.py:870: FutureWarning:

The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the
value of `n_init` explicitly to suppress the warning
```

```
/home/carlos/Documentos/diplomado/modulo 1/myenv/lib/python3.10/site-
packages/sklearn/cluster/_kmeans.py:870: FutureWarning:

The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the
value of `n_init` explicitly to suppress the warning

/home/carlos/Documentos/diplomado/modulo 1/myenv/lib/python3.10/site-
packages/sklearn/cluster/_kmeans.py:870: FutureWarning:

The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the
value of `n_init` explicitly to suppress the warning

/home/carlos/Documentos/diplomado/modulo 1/myenv/lib/python3.10/site-
packages/sklearn/cluster/_kmeans.py:870: FutureWarning:

The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the
value of `n_init` explicitly to suppress the warning

/home/carlos/Documentos/diplomado/modulo 1/myenv/lib/python3.10/site-
packages/sklearn/cluster/_kmeans.py:870: FutureWarning:

The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the
value of `n_init` explicitly to suppress the warning

/home/carlos/Documentos/diplomado/modulo 1/myenv/lib/python3.10/site-
packages/sklearn/cluster/_kmeans.py:870: FutureWarning:

The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the
value of `n_init` explicitly to suppress the warning

/home/carlos/Documentos/diplomado/modulo 1/myenv/lib/python3.10/site-
packages/sklearn/cluster/_kmeans.py:870: FutureWarning:

The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the
value of `n_init` explicitly to suppress the warning

/home/carlos/Documentos/diplomado/modulo 1/myenv/lib/python3.10/site-
packages/sklearn/cluster/_kmeans.py:870: FutureWarning:

The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the
value of `n_init` explicitly to suppress the warning

/home/carlos/Documentos/diplomado/modulo 1/myenv/lib/python3.10/site-
packages/sklearn/cluster/_kmeans.py:870: FutureWarning:

The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the
value of `n_init` explicitly to suppress the warning
```

```
/home/carlos/Documentos/diplomado/modulo 1/myenv/lib/python3.10/site-
packages/sklearn/cluster/_kmeans.py:870: FutureWarning:

The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the
value of `n_init` explicitly to suppress the warning

/home/carlos/Documentos/diplomado/modulo 1/myenv/lib/python3.10/site-
packages/sklearn/cluster/_kmeans.py:870: FutureWarning:

The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the
value of `n_init` explicitly to suppress the warning

/home/carlos/Documentos/diplomado/modulo 1/myenv/lib/python3.10/site-
packages/sklearn/cluster/_kmeans.py:870: FutureWarning:

The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the
value of `n_init` explicitly to suppress the warning

/home/carlos/Documentos/diplomado/modulo 1/myenv/lib/python3.10/site-
packages/sklearn/cluster/_kmeans.py:870: FutureWarning:

The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the
value of `n_init` explicitly to suppress the warning
```

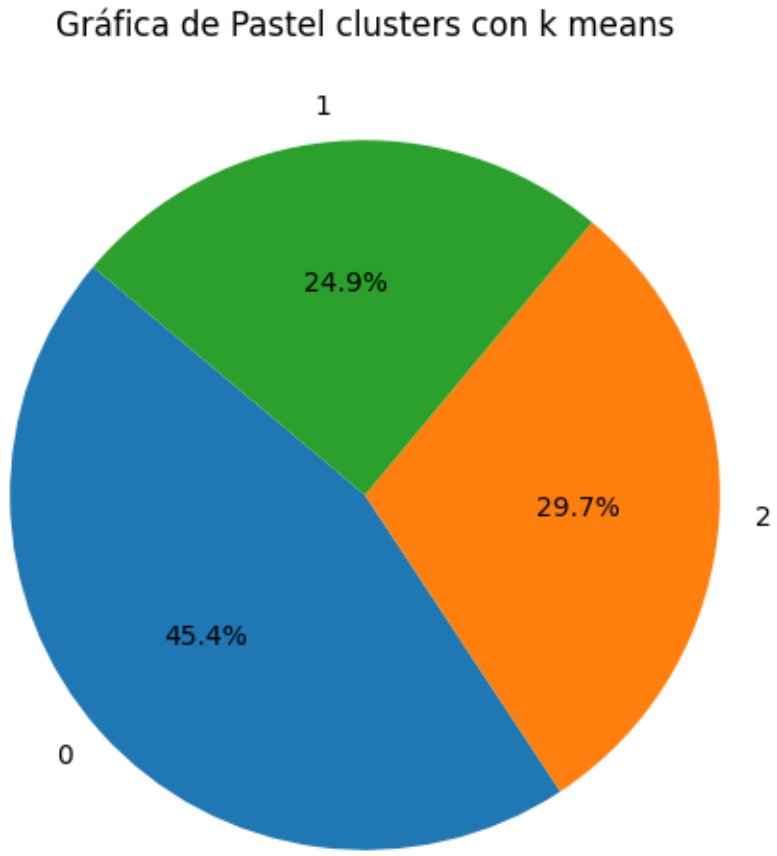
```
[382]: Text(0, 0.5, 'Inercia')
```



[383]:
```
# modelacion Kmeans
# K-Means (k=3) por visaulización:
kmeans_mds_3 = KMeans(3)
kmeans_mds_3.fit(Xmds_sample)

predictions = kmeans_mds_3.fit_predict(Xmds_sample)
Xmds_sample['kmeans_mds_3'] = predictions
```

```
/home/carlos/Documentos/diplomado/modulo 1/myenv/lib/python3.10/site-
packages/sklearn/cluster/_kmeans.py:870: FutureWarning:

The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the
value of `n_init` explicitly to suppress the warning

/home/carlos/Documentos/diplomado/modulo 1/myenv/lib/python3.10/site-
packages/sklearn/cluster/_kmeans.py:870: FutureWarning:

The default value of `n_init` will change from 10 to 'auto' in 1.4. Set the
value of `n_init` explicitly to suppress the warning
```

[384]:
```
conteo_clusters = Xmds_sample['kmeans_mds_3'].value_counts()
plt.figure(figsize=(6, 6))
plt.pie(conteo_clusters, labels=conteo_clusters.index, autopct='%1.1f%%',␣
 ↪startangle=140)
plt.title('Gráfica de Pastel clusters con k means')
plt.show()
```



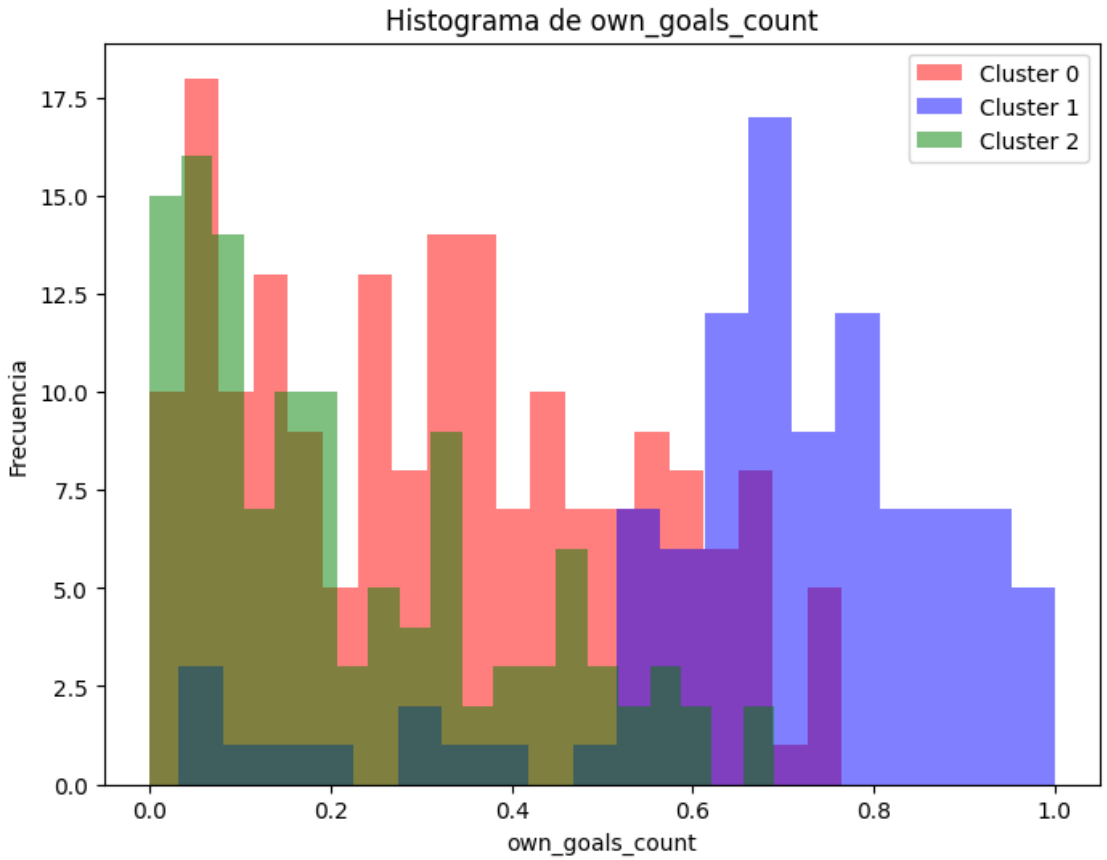
[385]:
```
%%time
grp = 'kmeans_mds_3'
ls_tukey=[]
for c_ in Xmm_sample.columns:
    #print(c_)
    tukey = pairwise_tukeyhsd(
        endog=Xmm_sample[c_],
        groups=Xmds_sample[grp],
        alpha=0.05
        )
```

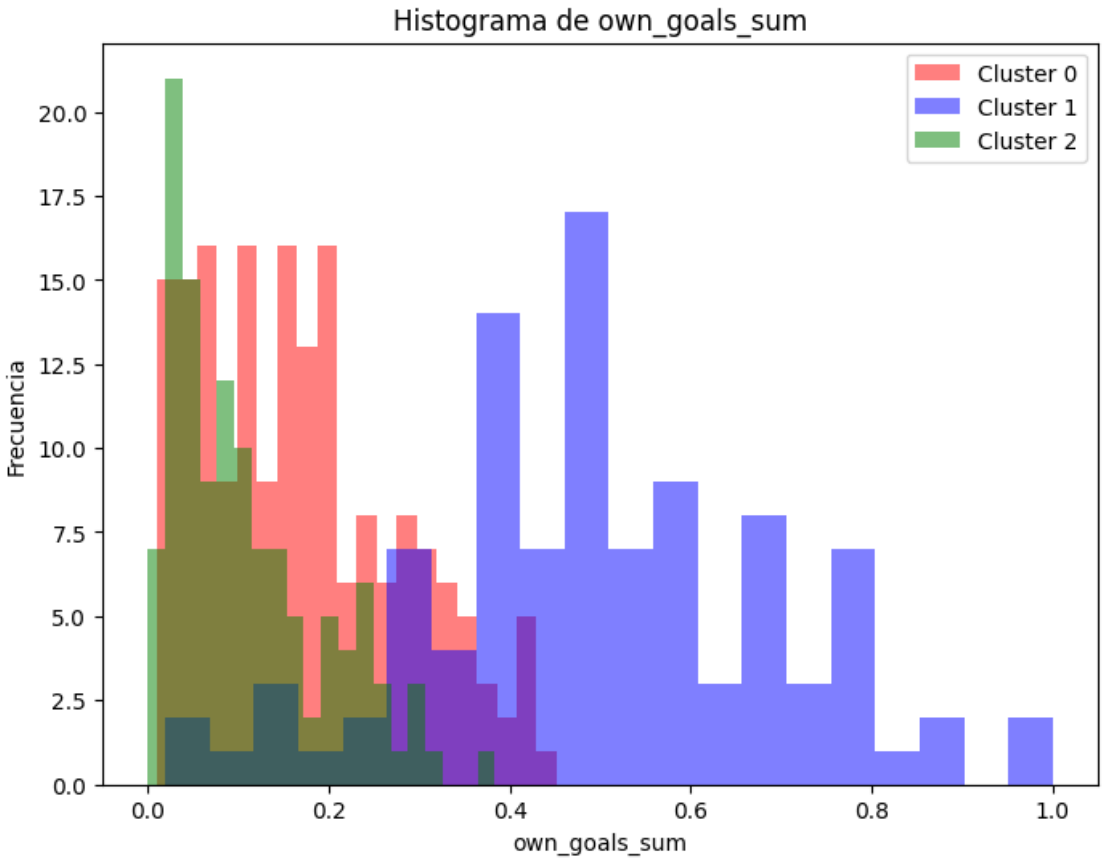
```
    tukey_df = pd.DataFrame(data=tukey._results_table.data[1:], columns=tukey.
↪_results_table.data[0])
    if tukey_df['reject'].mean()==1:
        print(f'variable con todos los valores de reject verdadero es {c_}')
        ls_tukey.append(c_)
```

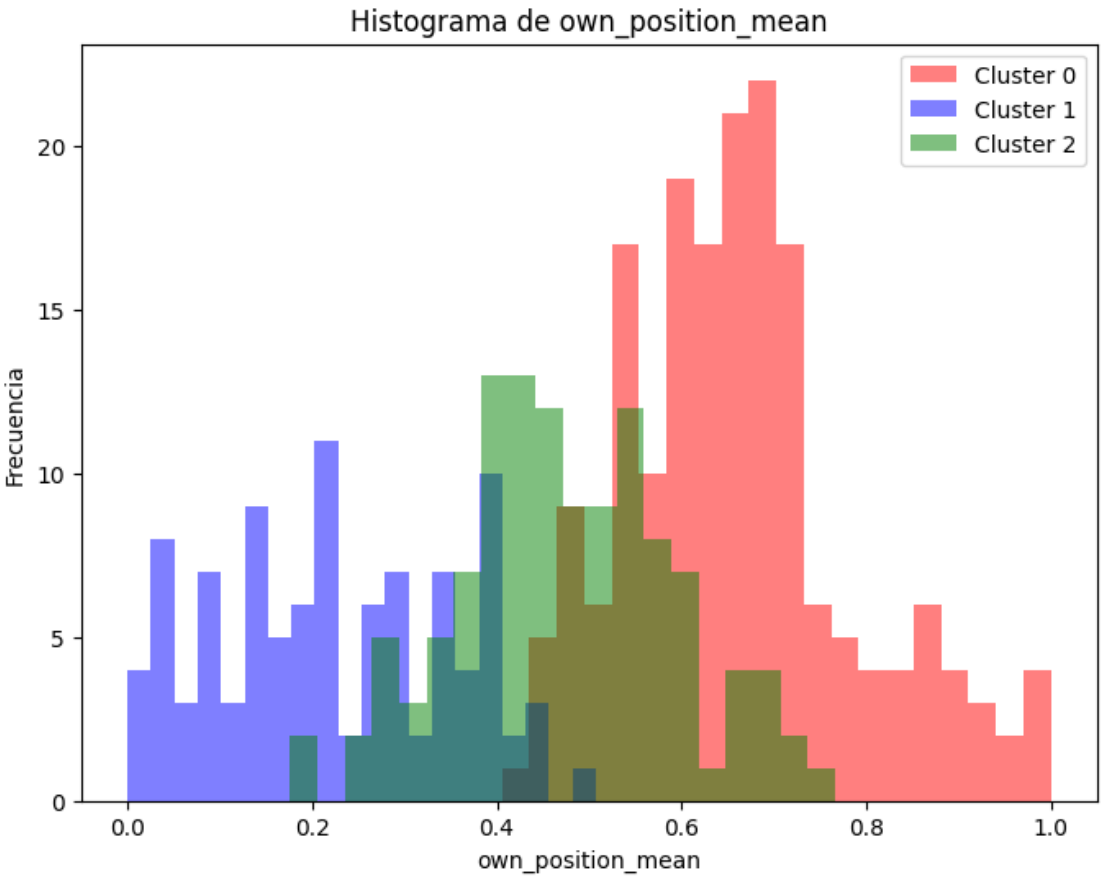
```
variable con todos los valores de reject verdadero es own_goals_count
variable con todos los valores de reject verdadero es own_goals_sum
variable con todos los valores de reject verdadero es own_position_mean
variable con todos los valores de reject verdadero es own_position_median
variable con todos los valores de reject verdadero es opponent_goals_sum
variable con todos los valores de reject verdadero es opponent_position_mean
variable con todos los valores de reject verdadero es opponent_position_median
variable con todos los valores de reject verdadero es is_win_visit_mean
variable con todos los valores de reject verdadero es dif_goals_visit_sum
variable con todos los valores de reject verdadero es dif_goals_visit_mean
variable con todos los valores de reject verdadero es dif_goals_visit_median
variable con todos los valores de reject verdadero es Total_goals_sum_y
variable con todos los valores de reject verdadero es Total_goals_mean_y
variable con todos los valores de reject verdadero es Total_goals_median_y
variable con todos los valores de reject verdadero es is_win_over_2_visit_mean
variable con todos los valores de reject verdadero es is_win_over_3_visit_mean
variable con todos los valores de reject verdadero es foreigners_number
variable con todos los valores de reject verdadero es national_team_players
variable con todos los valores de reject verdadero es stadium_seats
CPU times: user 9.44 s, sys: 128 ms, total: 9.57 s
Wall time: 9.53 s
```

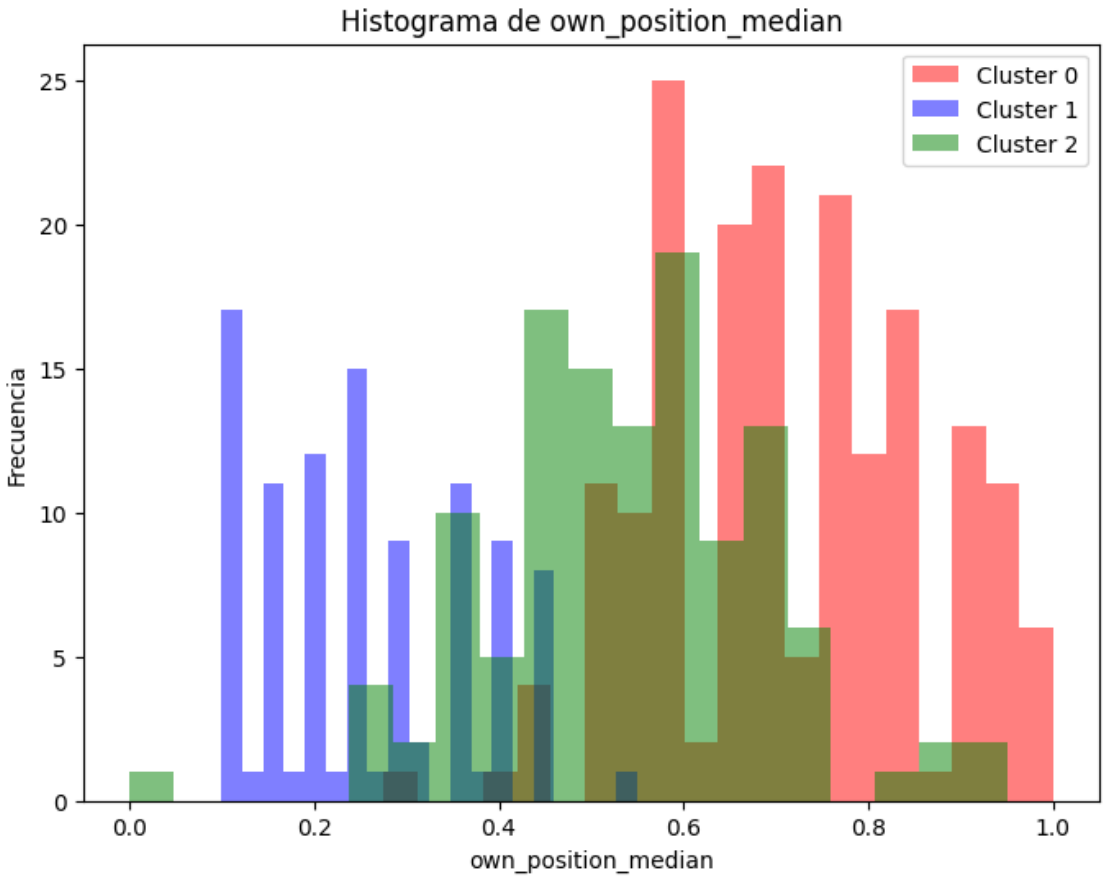
[386]:
```
unique_clusters = Xmds_sample['kmeans_mds_3'].unique()

# Color mapping para kmeans_mds_3
color_mapping = {0: 'red', 1: 'blue', 2: 'green'}

# Itera a través de las variables en ls_best
for variable in ls_tukey:
    # Crear un nuevo histograma para la variable actual
    plt.figure(figsize=(8, 6))  # Establece el tamaño de la figura (opcional)

    # Itera a través de los valores únicos de kmeans_mds_3
    for cluster_value in unique_clusters:
        # Restablece el índice del DataFrame Xmm_sample antes de la selección
        subset_data = Xmm_sample.
↪reset_index(drop=True)[Xmds_sample['kmeans_mds_3'] ==␣
↪cluster_value][variable]

        # Crea el histograma utilizando solo un color para este conjunto de␣
↪datos
```

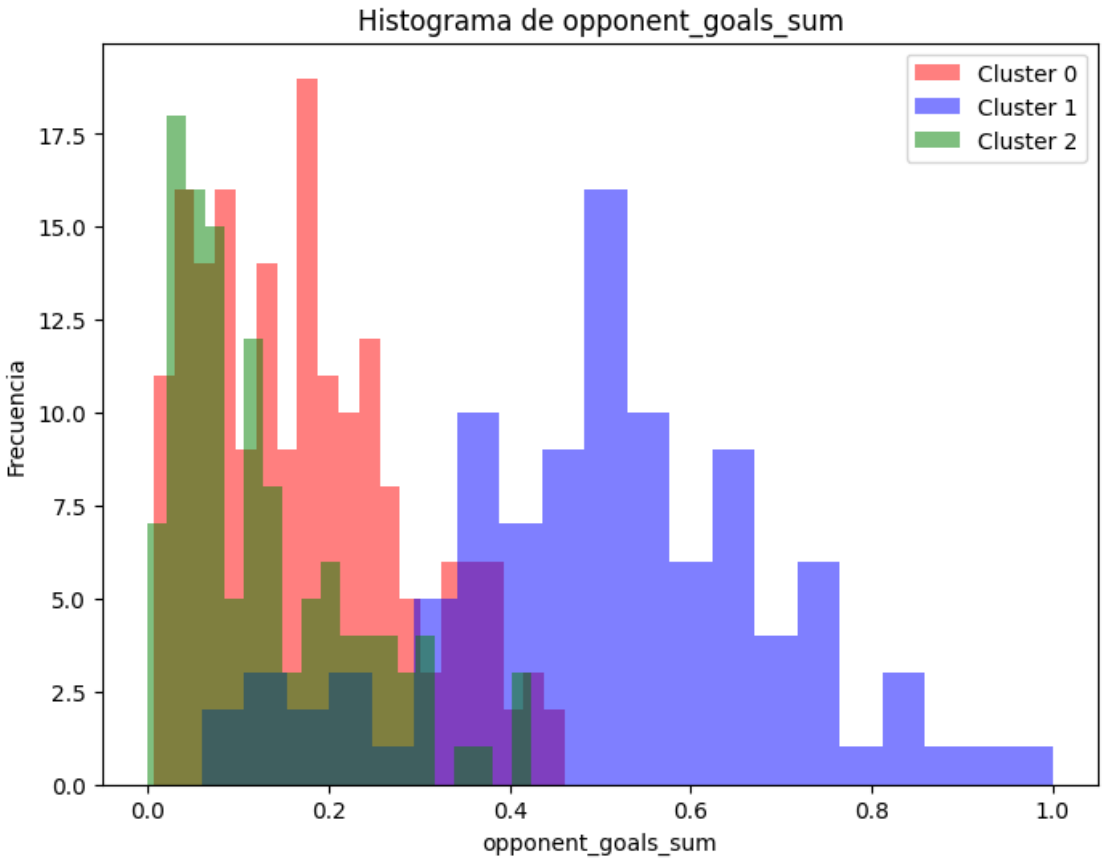
```
        plt.hist(subset_data, bins=20, color=color_mapping[cluster_value],␣
↪alpha=0.5, label=f'Cluster {cluster_value}')

    # Configura el título y etiquetas de los ejes
    plt.title(f'Histograma de {variable}')
    plt.xlabel(variable)
    plt.ylabel('Frecuencia')

    # Agrega una leyenda para identificar los clusters
    plt.legend()

    # Muestra la gráfica
    plt.show()
```

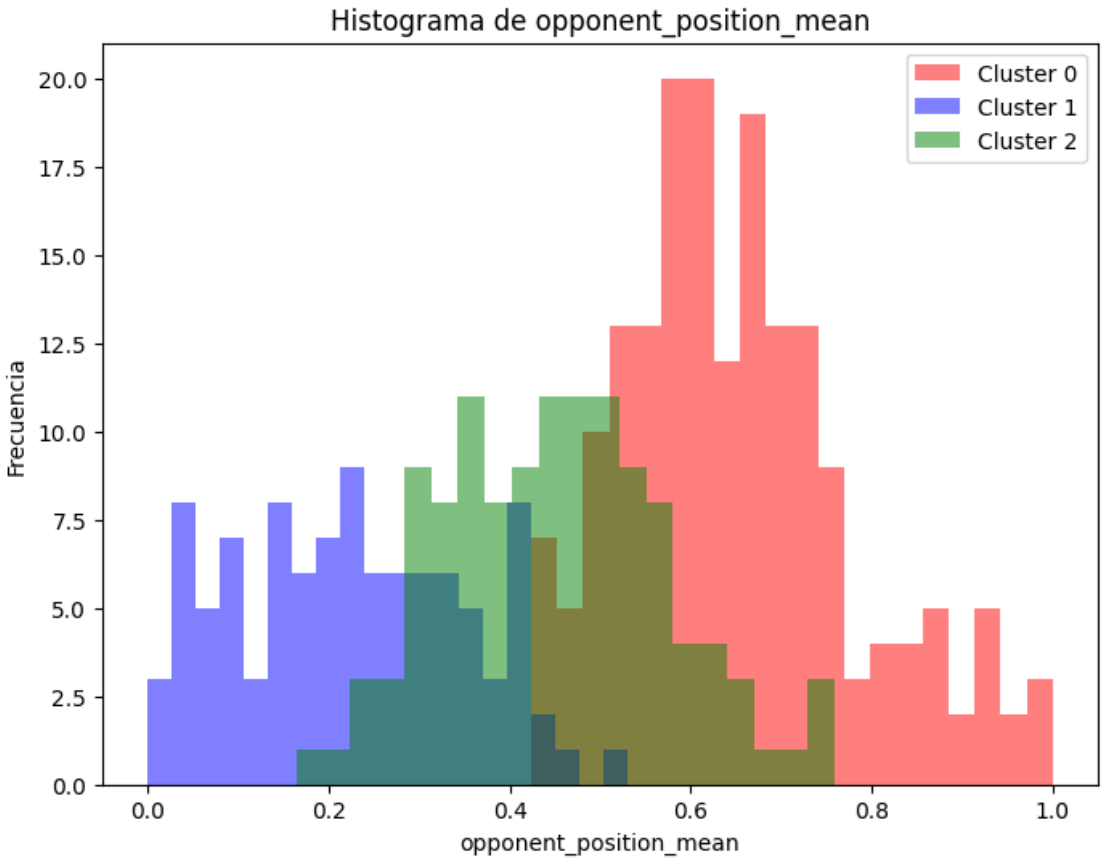


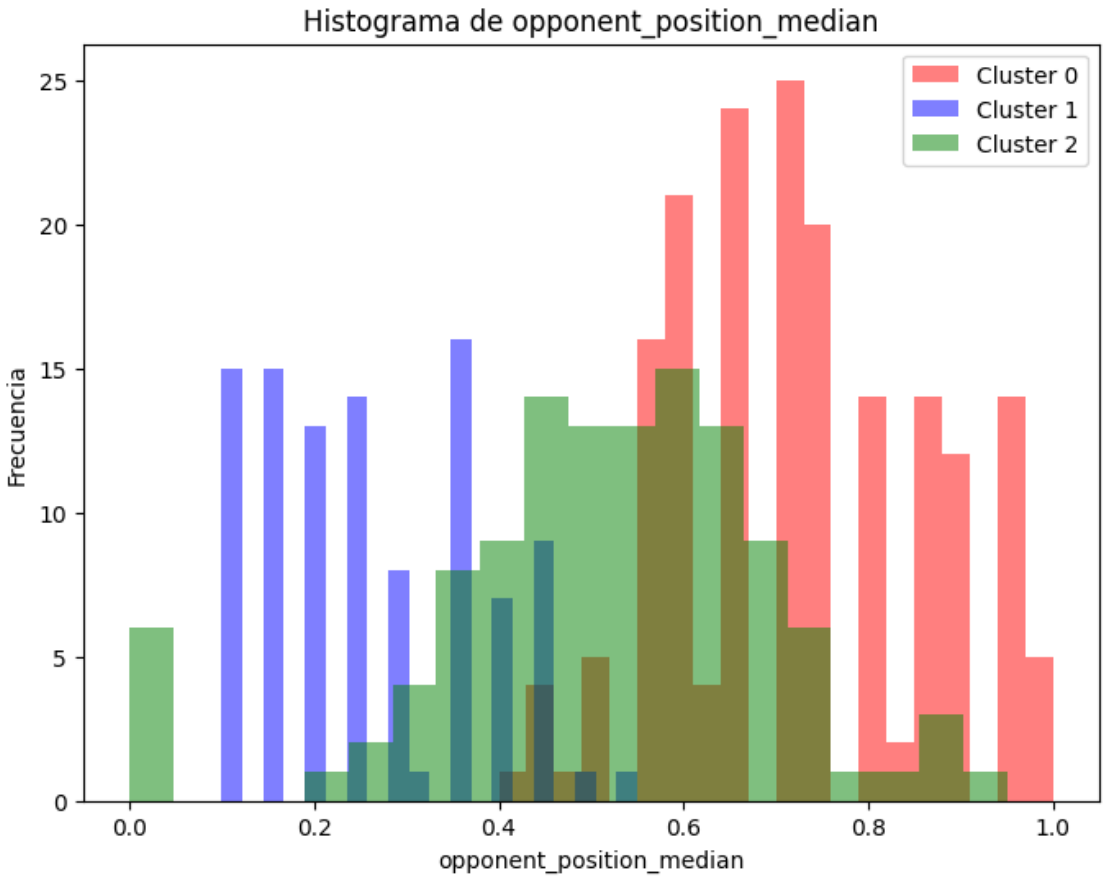

Histograma de own_goals_sum
Cluster 0
Cluster 1
Cluster 2
Frecuencia
own_goals_sum
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
0.0
0.2
0.4
0.6
0.8
1.0

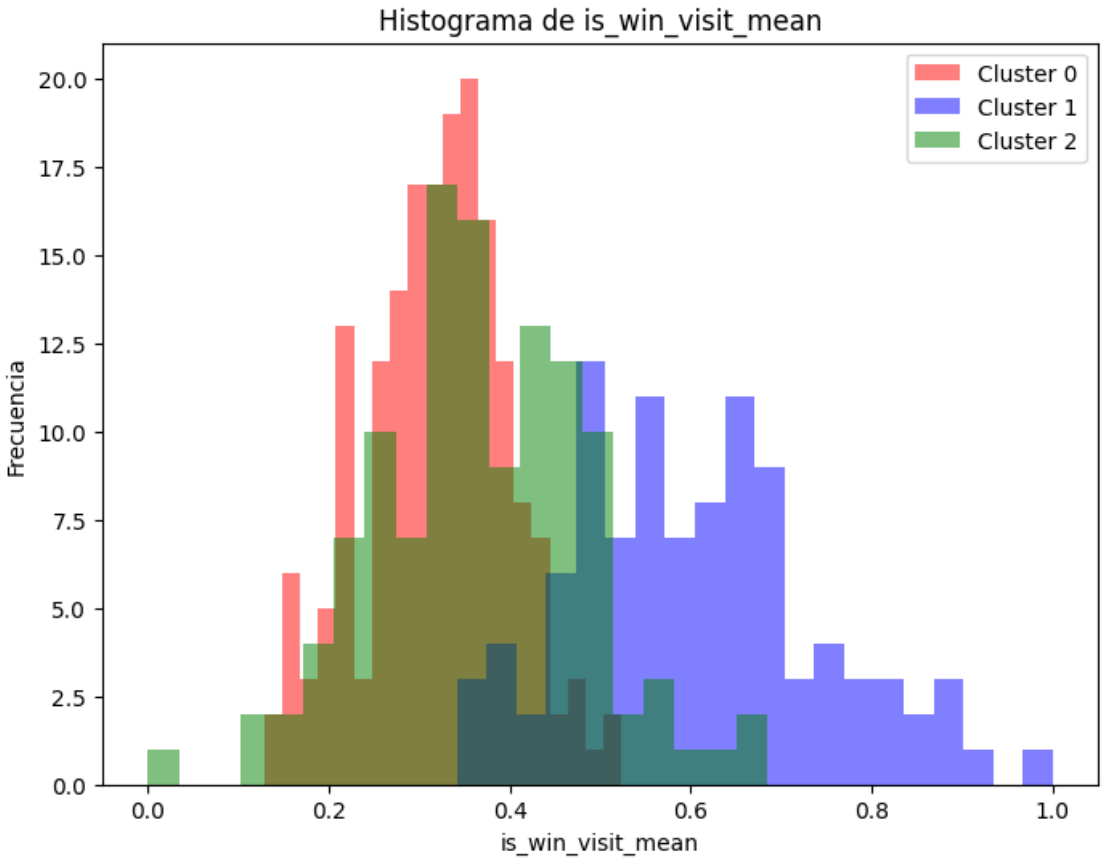

Histograma de own_position_mean
Cluster 0
Cluster 1
Cluster 2
Frecuencia
0
5
10
15
20
0.0
0.2
0.4
0.6
0.8
1.0
own_position_mean

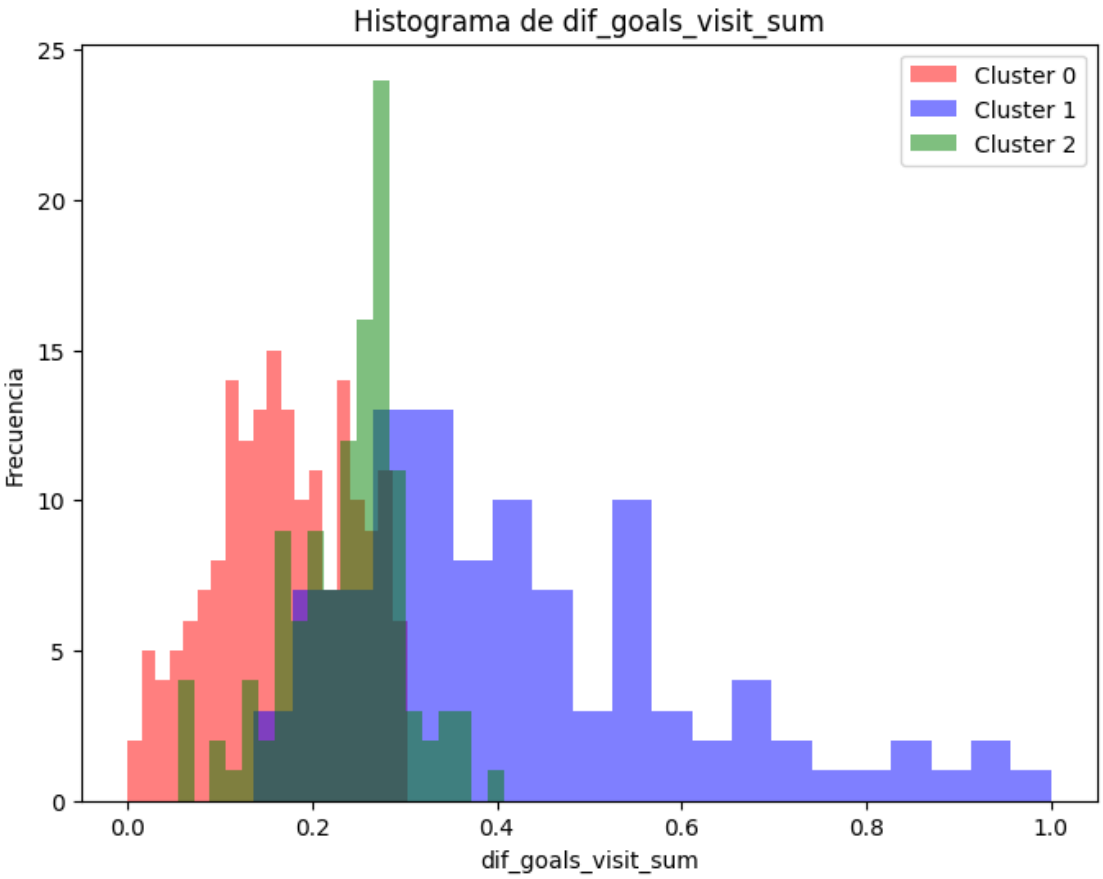

Histograma de own_position_median
Cluster 0
Cluster 1
Cluster 2
Frecuencia
own_position_median
0
5
10
15
20
25
0.0
0.2
0.4
0.6
0.8
1.0

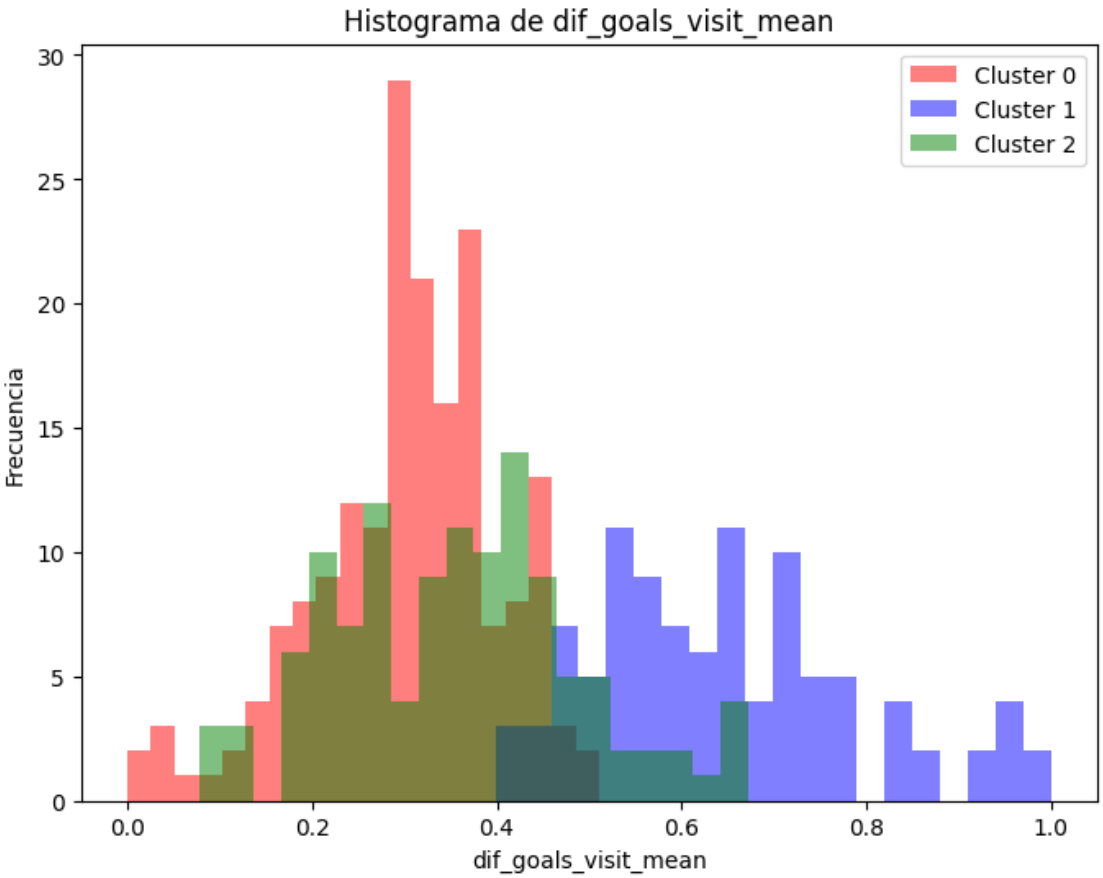

Histograma de opponent_goals_sum
Cluster 0
Cluster 1
Cluster 2
Frecuencia
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0
opponent_goals_sum

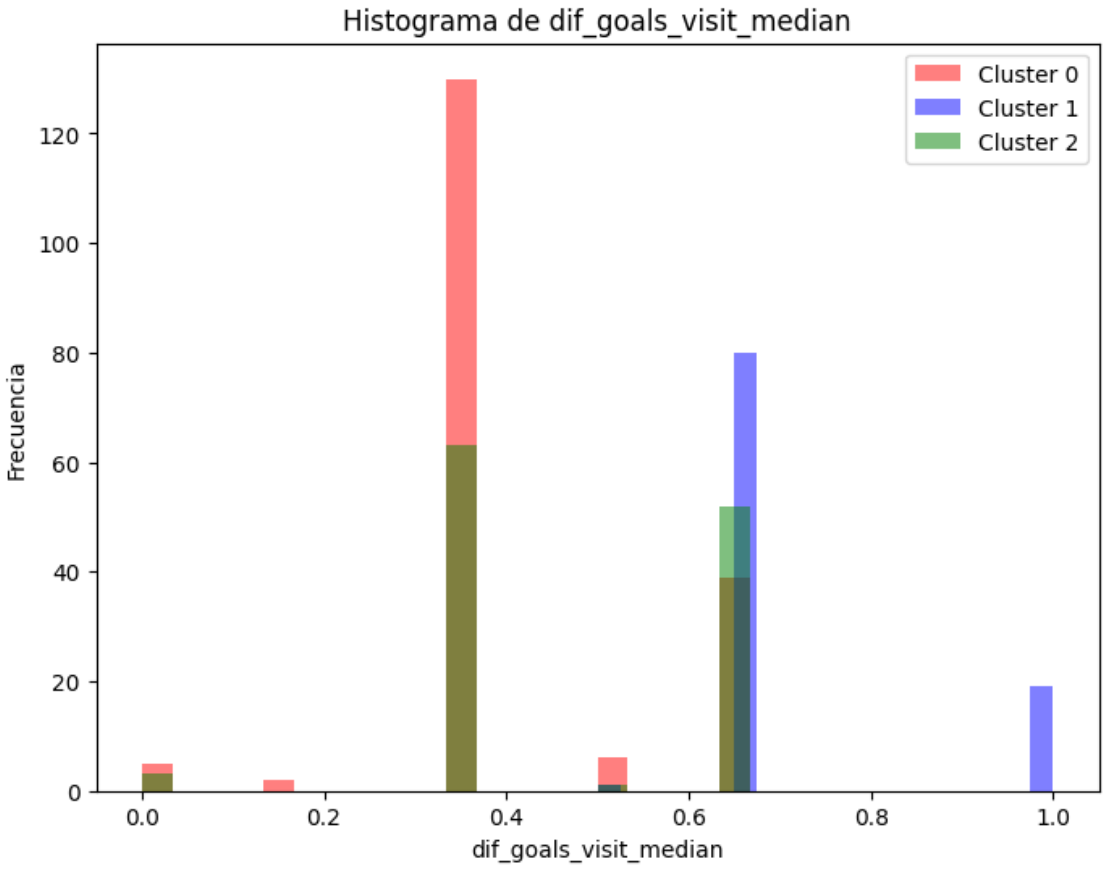

Histograma de opponent_position_mean
Cluster 0
Cluster 1
Cluster 2
Frecuencia
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0
opponent_position_mean

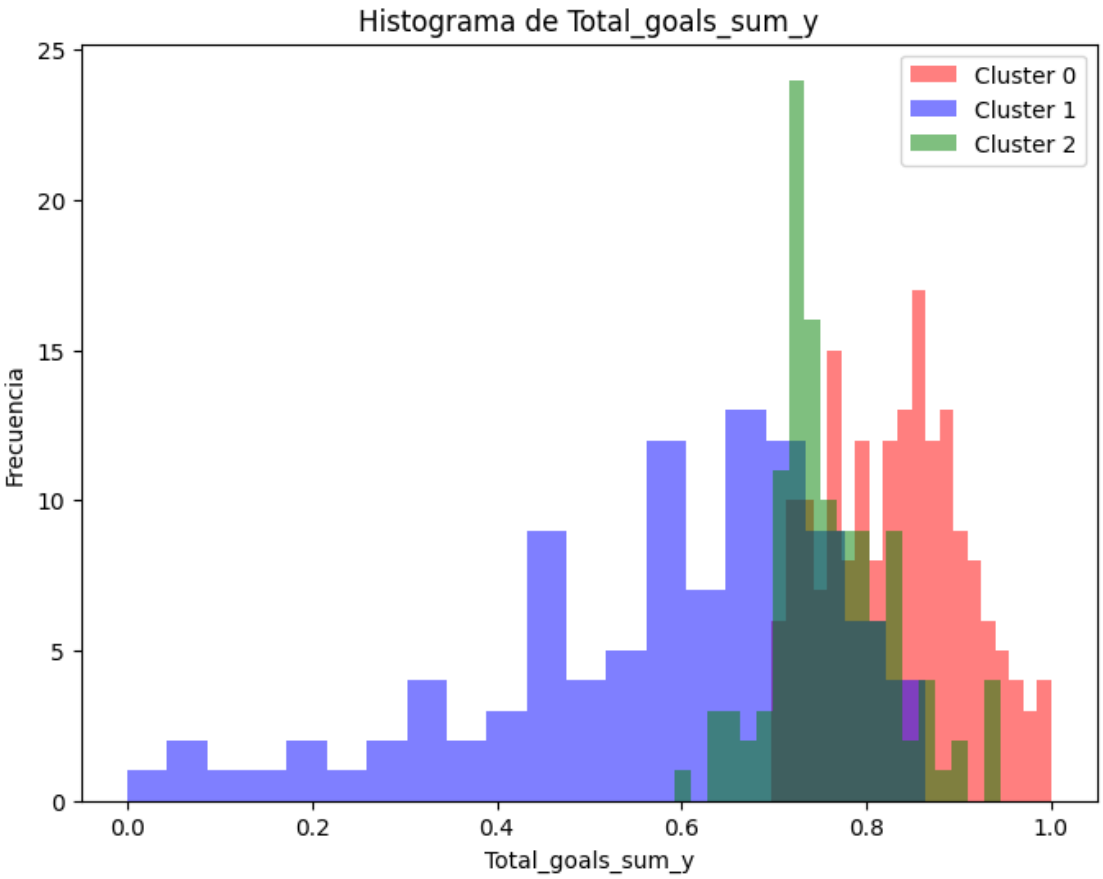

Histograma de opponent_position_median
Cluster 0
Cluster 1
Cluster 2
Frecuencia
0
5
10
15
20
25
0.0
0.2
0.4
0.6
0.8
1.0
opponent_position_median

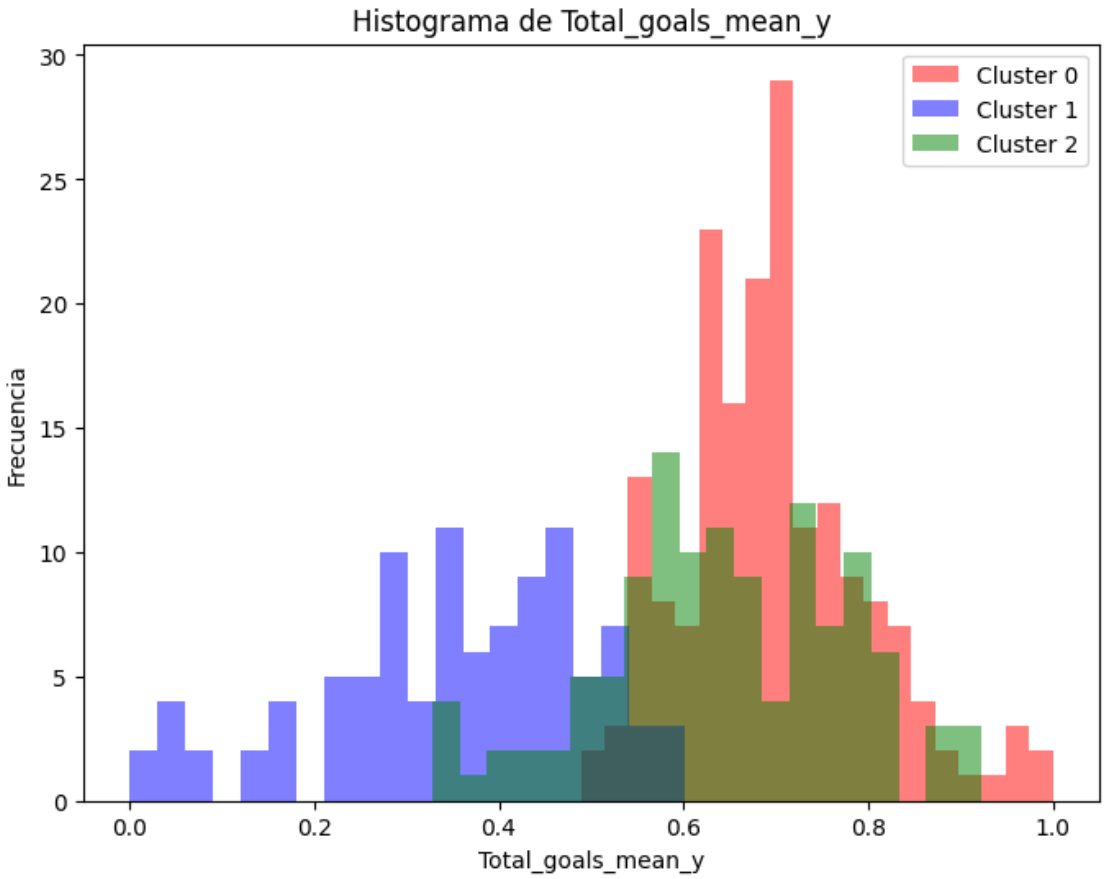

Histograma de is_win_visit_mean
Cluster 0
Cluster 1
Cluster 2
Frecuencia
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0
is_win_visit_mean

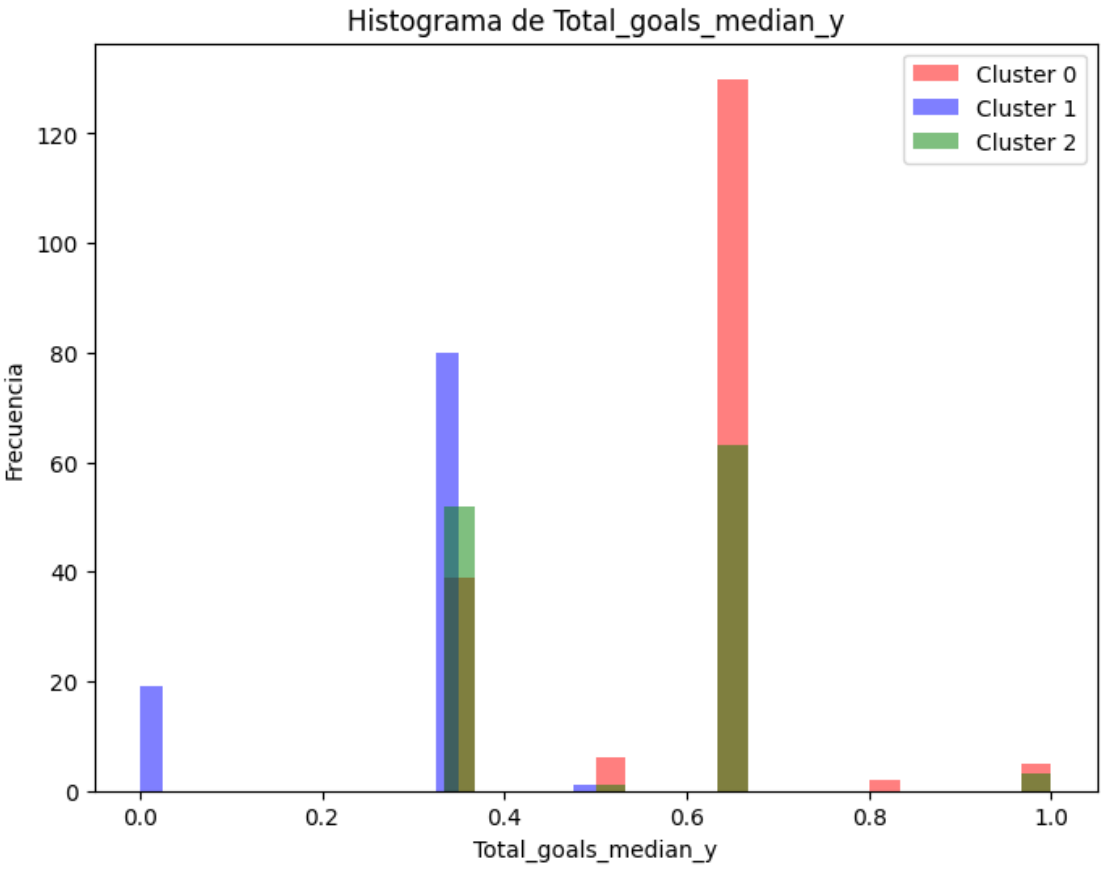

Histograma de dif_goals_visit_sum
Cluster 0
Cluster 1
Cluster 2
Frecuencia
dif_goals_visit_sum
0
5
10
15
20
25
0.0
0.2
0.4
0.6
0.8
1.0

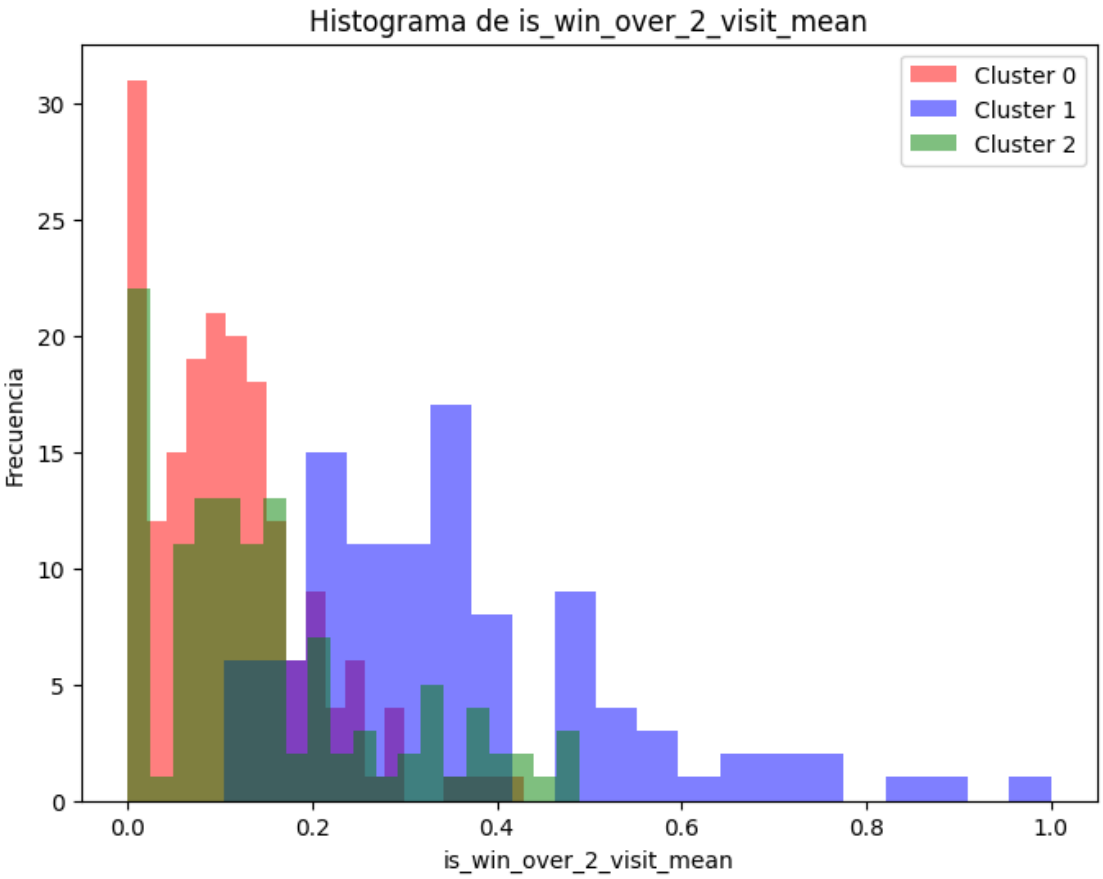

Histograma de dif_goals_visit_mean
Cluster 0
Cluster 1
Cluster 2
Frecuencia
dif_goals_visit_mean
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0

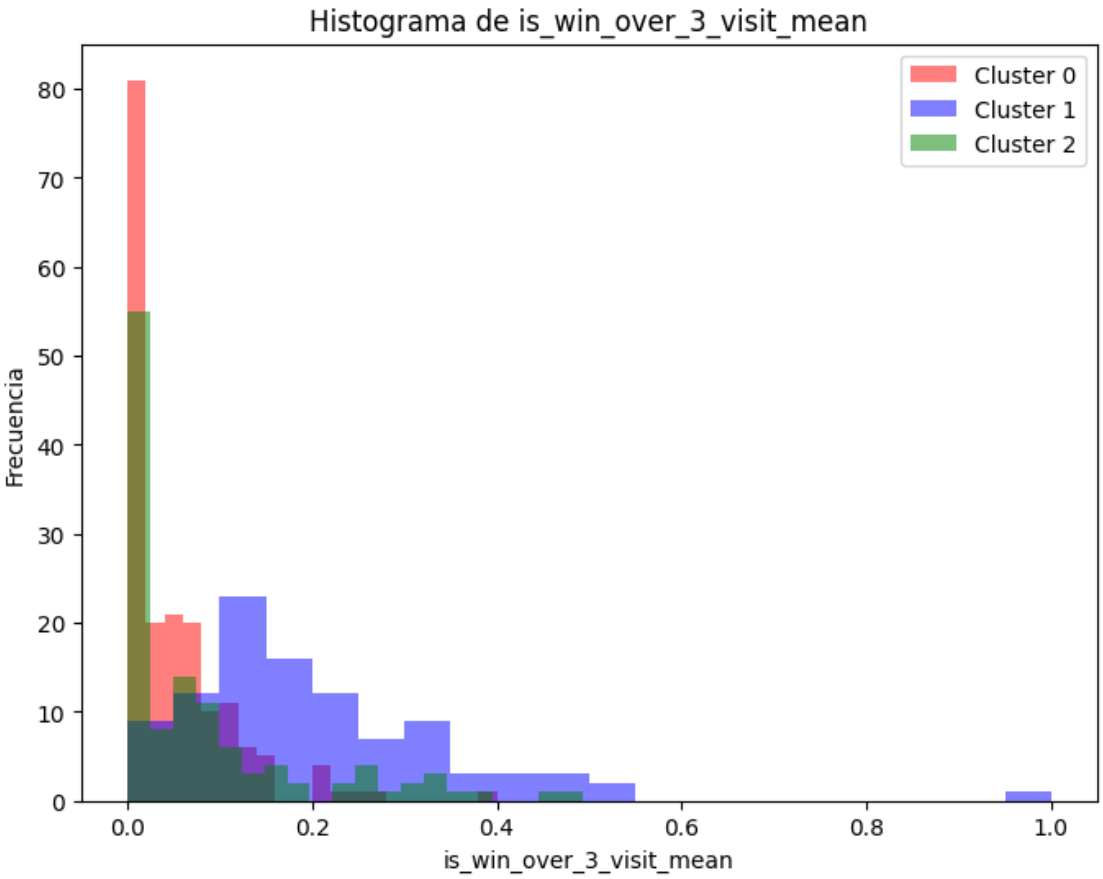

Histograma de dif_goals_visit_median
Cluster 0
Cluster 1
Cluster 2
Frecuencia
0
20
40
60
80
100
120
0.0
0.2
0.4
0.6
0.8
1.0
dif_goals_visit_median

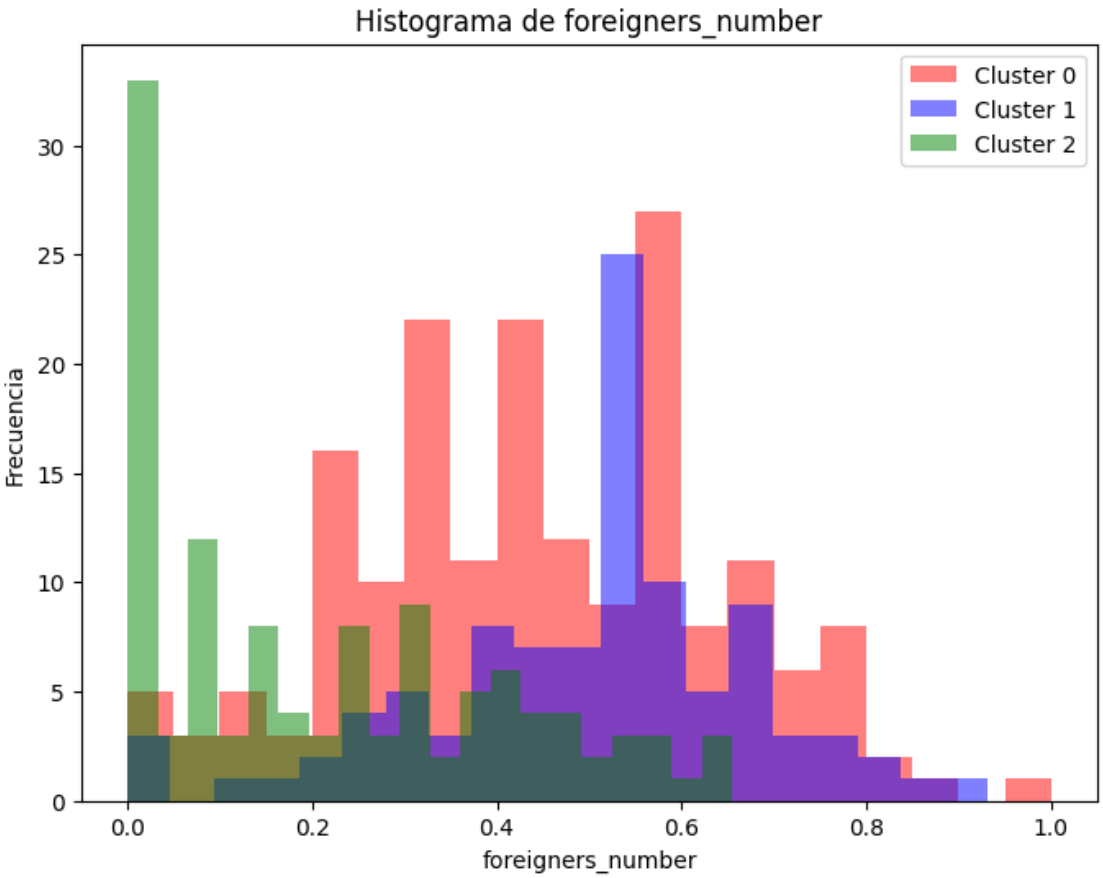

Histograma de Total_goals_sum_y
Cluster 0
Cluster 1
Cluster 2
Frecuencia
Total_goals_sum_y
0
5
10
15
20
25
0.0
0.2
0.4
0.6
0.8
1.0

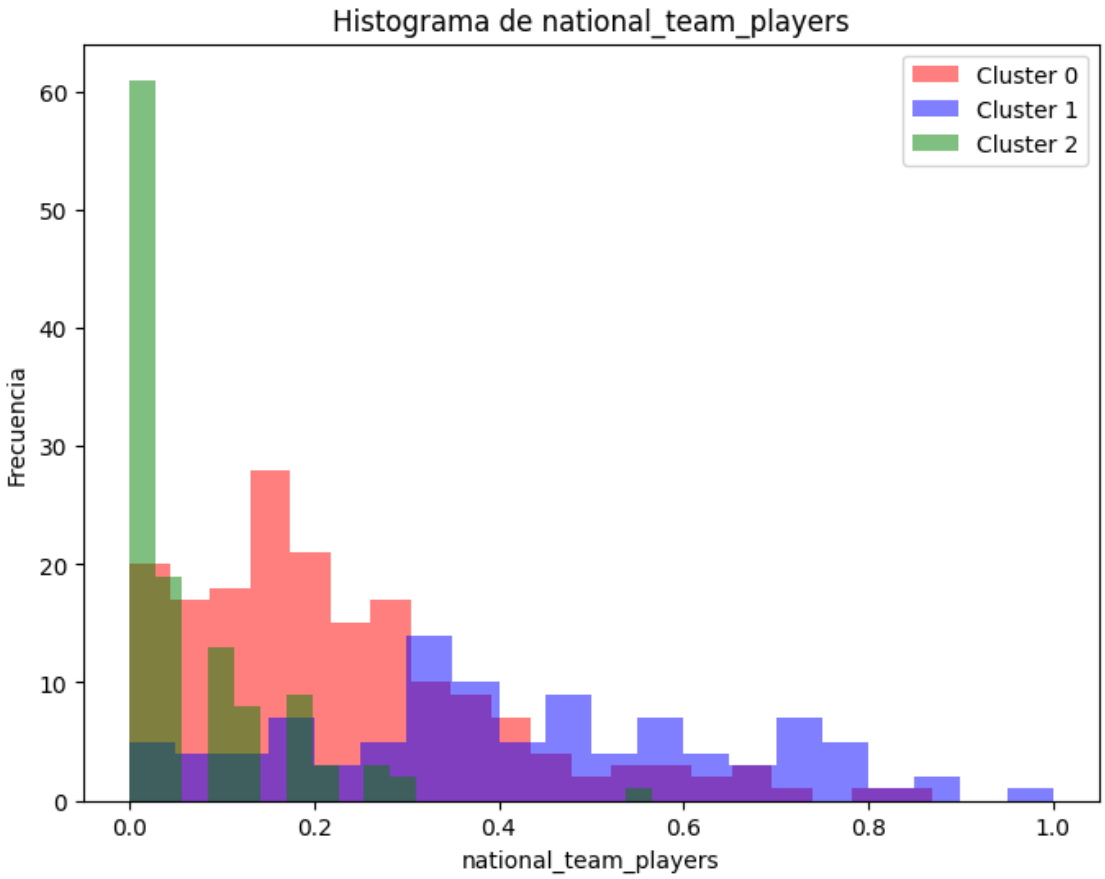

Histograma de Total_goals_mean_y
Cluster 0
Cluster 1
Cluster 2
Frecuencia
Total_goals_mean_y
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0

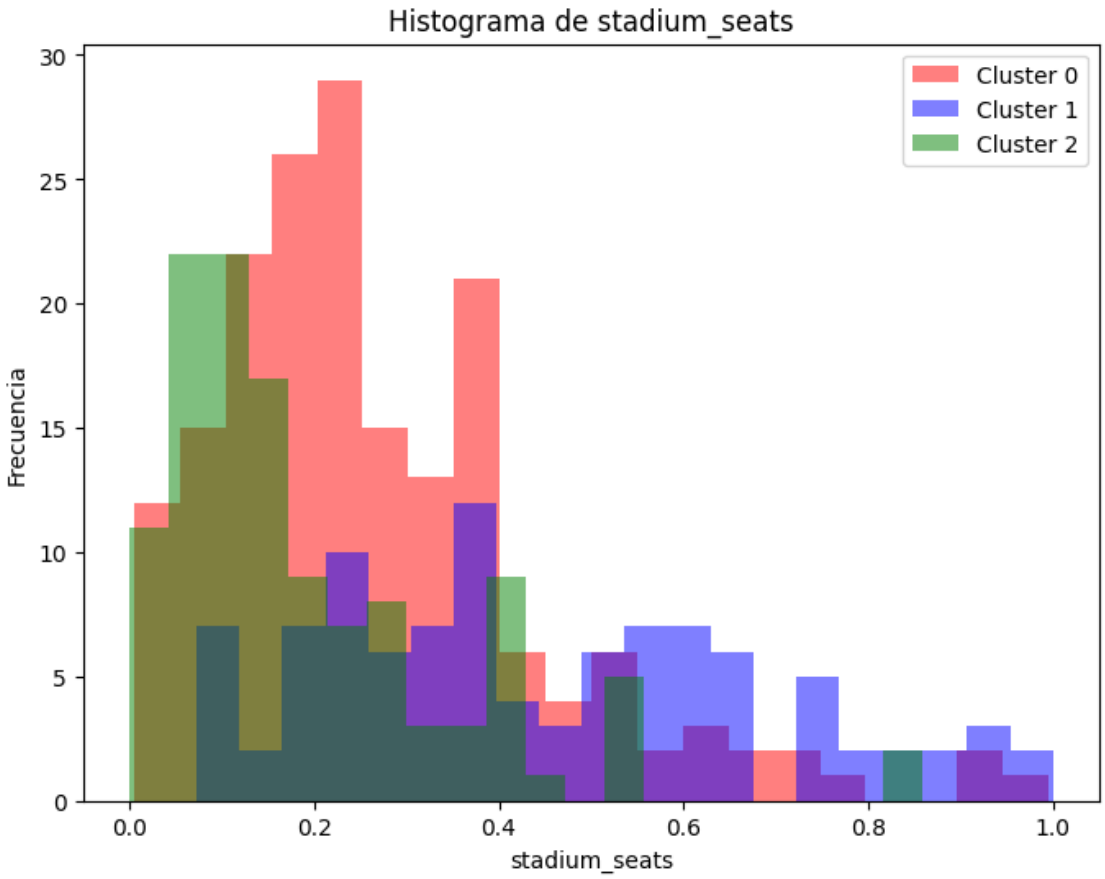

Histograma de Total_goals_median_y
Cluster 0
Cluster 1
Cluster 2
Frecuencia
0
20
40
60
80
100
120
0.0
0.2
0.4
0.6
0.8
1.0
Total_goals_median_y


Histograma de is_win_over_2_visit_mean
Cluster 0
Cluster 1
Cluster 2
Frecuencia
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0
is_win_over_2_visit_mean


Histograma de is_win_over_3_visit_mean
Cluster 0
Cluster 1
Cluster 2
Frecuencia
0
10
20
30
40
50
60
70
80
0.0
0.2
0.4
0.6
0.8
1.0
is_win_over_3_visit_mean

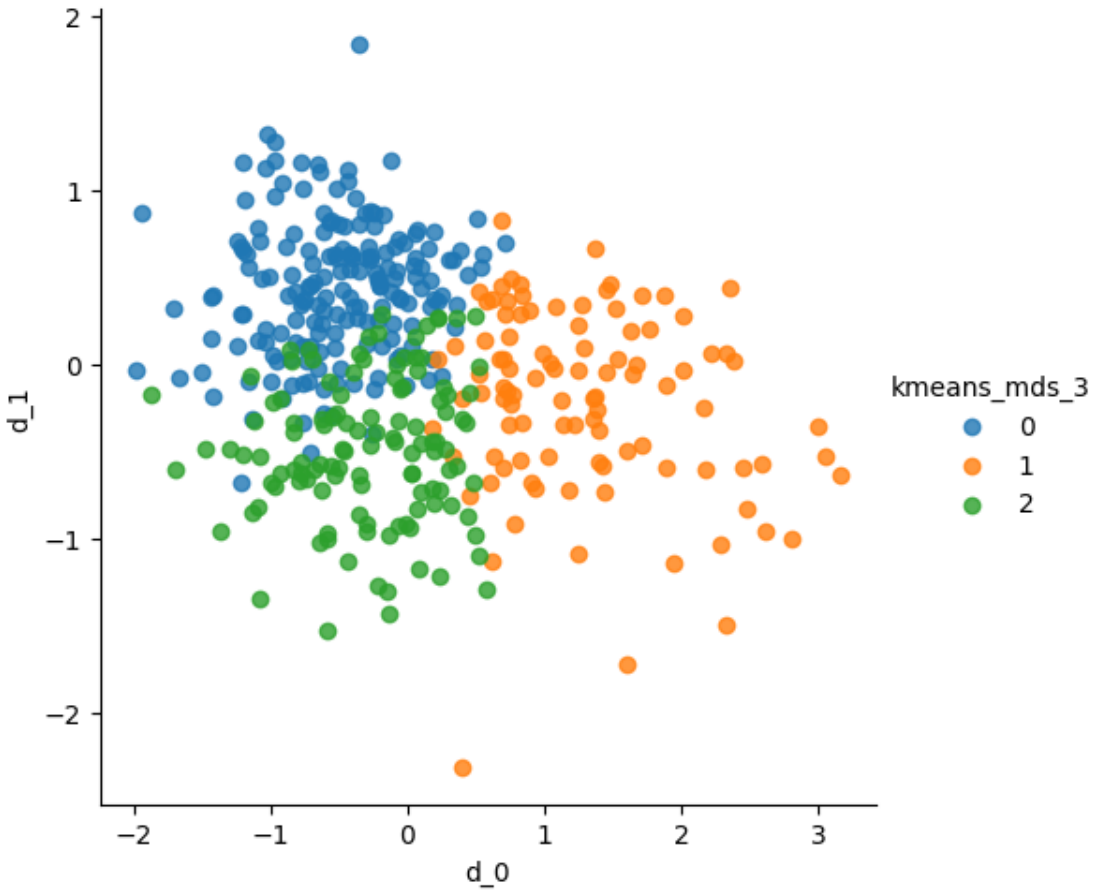

Histograma de foreigners_number
Cluster 0
Cluster 1
Cluster 2
Frecuencia
foreigners_number
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0


Histograma de national_team_players
Cluster 0
Cluster 1
Cluster 2
Frecuencia
national_team_players
0
10
20
30
40
50
60
0.0
0.2
0.4
0.6
0.8
1.0



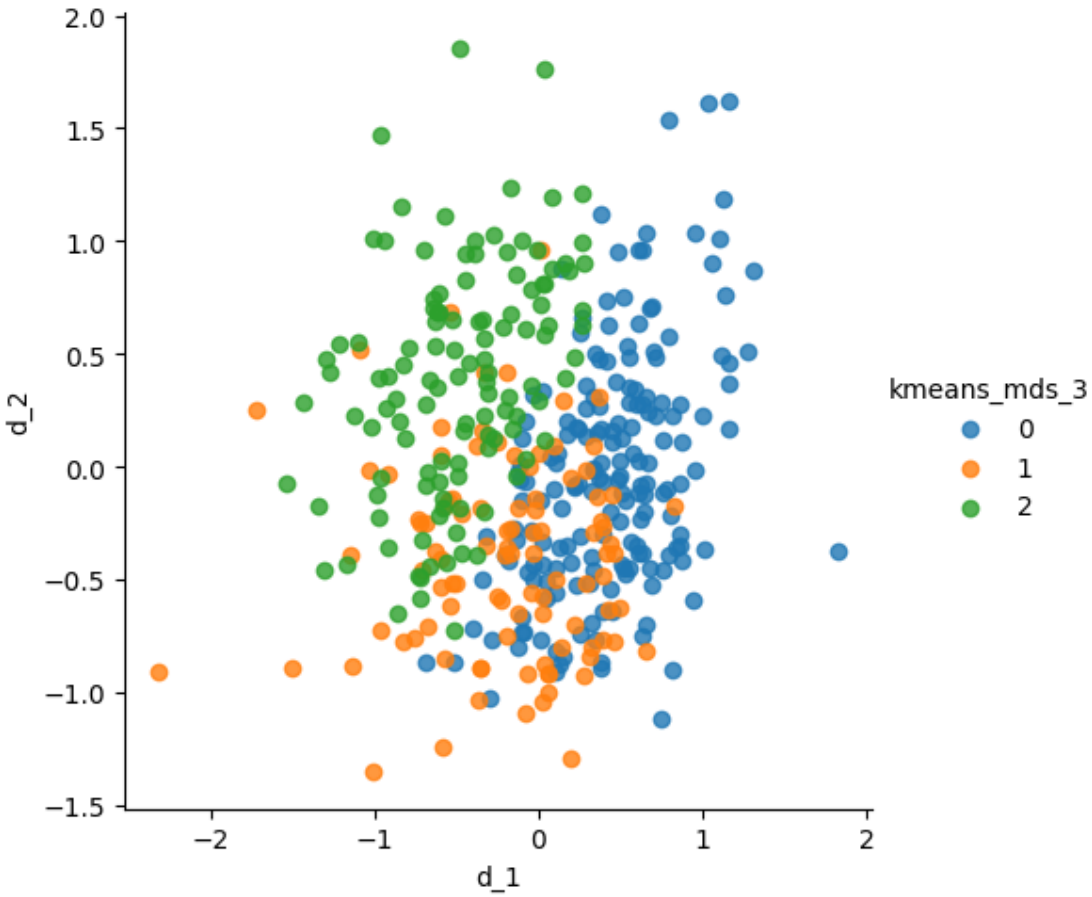
[387]:
```
sns.lmplot(data=Xmds_sample, x='d_0', y='d_1', fit_reg=False,␣
↪hue='kmeans_mds_3')
```

[387]: <seaborn.axisgrid.FacetGrid at 0x7fdeba5f64a0>



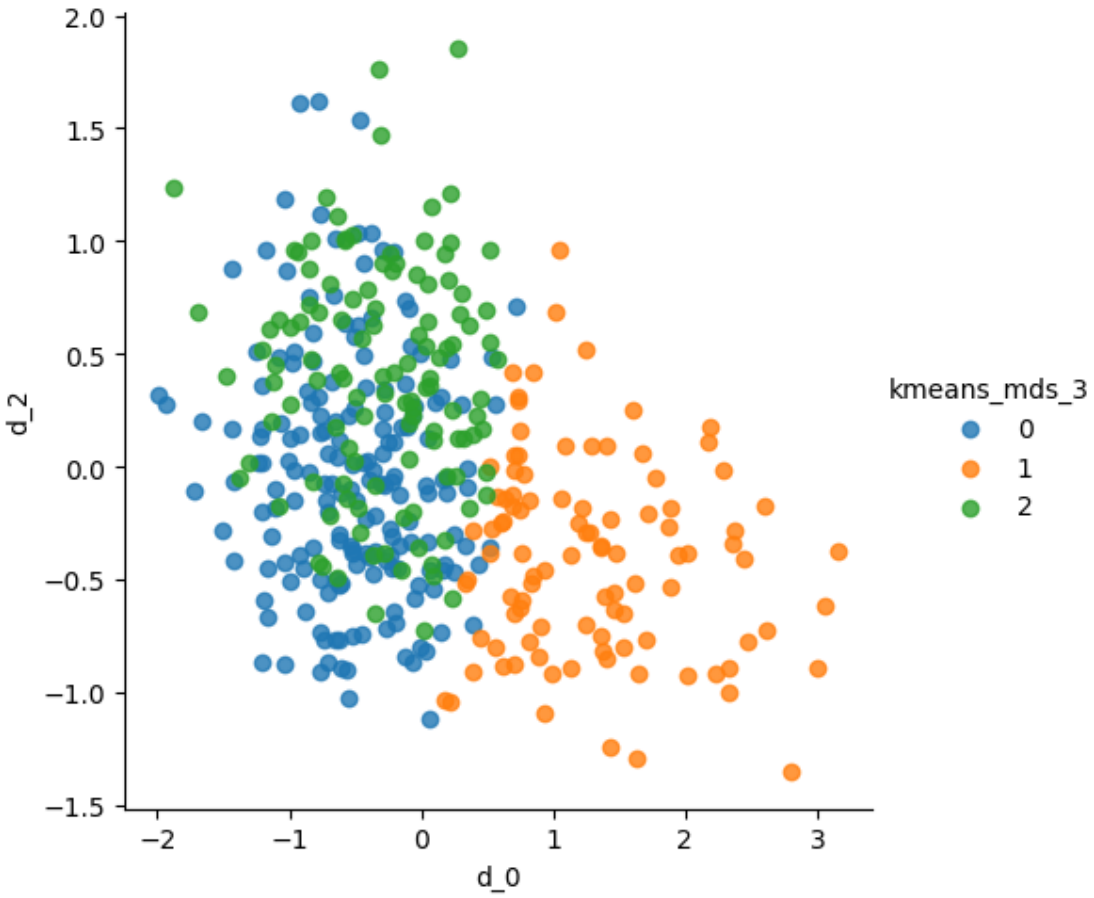
[388]:
```
sns.lmplot(data=Xmds_sample, x='d_1', y='d_2', fit_reg=False,␣
↪hue='kmeans_mds_3')
```

[388]: <seaborn.axisgrid.FacetGrid at 0x7fdeba5f7160>



[389]:
```
sns.lmplot(data=Xmds_sample, x='d_0', y='d_2', fit_reg=False,␣
↪hue='kmeans_mds_3')
```

[389]: <seaborn.axisgrid.FacetGrid at 0x7fdebb4e0820>



4 PCA

[390]:
```
sc = StandardScaler()
Xsc = pd.DataFrame(sc.fit_transform(df2), columns=df2.columns)
```

[391]:
```
from sklearn.decomposition import PCA
j = 3
pca = PCA(n_components=j)
```

[392]:
```
%%time
Xpca = pd.DataFrame(pca.fit_transform(Xsc), columns=[f'p_{i}' for i in␣
 ↪range(j)])
Xpca.head(2)
```

```
CPU times: user 21.5 ms, sys: 91.5 ms, total: 113 ms
Wall time: 37.5 ms
```

[392]:
```
        p_0       p_1       p_2
0  1.214334  1.968054 -2.561710
1 -3.728628  0.508560 -1.440877
```

[393]:
```
variance_explained = pca.explained_variance_ratio_

# Calcula la varianza explicada acumulativa
cumulative_variance_explained = np.cumsum(variance_explained)
print(cumulative_variance_explained)
```

```
[0.44135342 0.51616421 0.58537155]
```

[394]:
```
def pca_char_corr(X, Xp, pca):

    '''This function computes the correlation between each of the principal␣
 ↪components and the
       original variables of the data. The parameters are the next ones:
       1. X: pandas dataframe of original data.
       2. Xp: pandas dataframe of transformed data.
       3. pca: pca adjusted object.
    '''

    r = pd.DataFrame(
        data=[[np.corrcoef(X[c],Xp.loc[:,f'p_{n}'])[1,0] for n in range(pca.
 ↪n_components_)] for c in X],
        columns = [f'p_{i}' for i in range(pca.n_components_)],
        index = X.columns
        )

    return(r)
```

[395]:
```
pca_char_corr(df2, Xpca, pca)
```

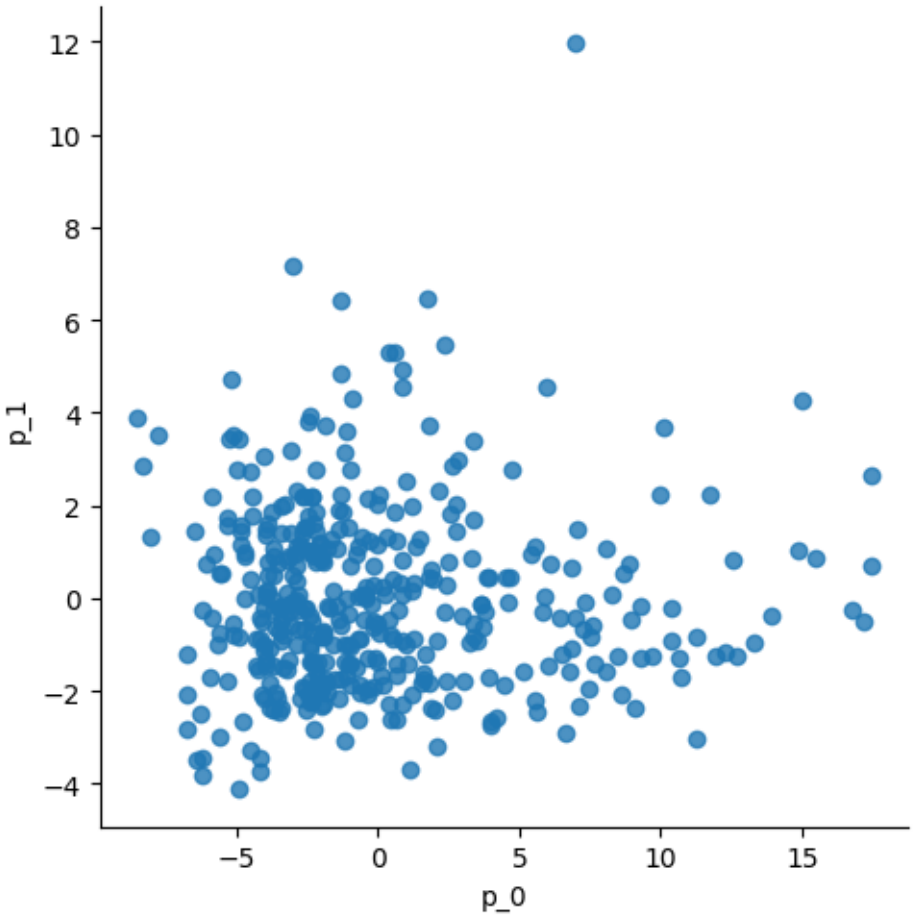
[395]:

	p_0	p_1	p_2
own_goals_count	0.729210	-0.273209	-0.442479
own_goals_sum	0.863581	-0.227763	-0.303107
own_goals_max	0.505395	-0.030745	-0.220229
own_goals_mean	0.848432	-0.155767	-0.042549
own_goals_median	0.753499	-0.152799	0.027193
own_position_min	-0.296923	-0.320223	0.281425
own_position_max	-0.504818	-0.466755	-0.401894
own_position_mean	-0.874906	-0.155881	-0.129746
own_position_median	-0.864322	-0.063950	-0.102629
is_win_mean	0.884078	-0.206783	-0.013236
dif_goals_loc_sum	0.905618	-0.180705	0.005978
dif_goals_loc_min	0.140705	-0.174318	0.403952
dif_goals_loc_max	0.716980	-0.181490	-0.235534

```
dif_goals_loc_mean         0.899555 -0.287379  0.007328
dif_goals_loc_median       0.791688 -0.237901 -0.008305
is_draw_mean_x            -0.275929 -0.207876 -0.082117
is_win_over_2_local_mean   0.852224 -0.130078 -0.011306
is_win_over_3_local_mean   0.823620 -0.095332  0.025729
is_win_over_4_local_mean   0.728119 -0.059928  0.080566
is_lost_over_2_local_mean -0.532865  0.389270 -0.137591
is_lost_over_3_local_mean -0.442660  0.394884 -0.035831
is_lost_over_4_local_mean -0.292840  0.306252 -0.081041
diferent_manager_          0.049440 -0.186833 -0.420802
opponent_goals_sum         0.876208 -0.115139 -0.313955
opponent_goals_max         0.398331  0.410252 -0.268674
opponent_goals_mean        0.814678  0.313810 -0.059354
opponent_goals_median      0.560482  0.024146 -0.060428
opponent_position_min     -0.299707 -0.310922  0.280746
opponent_position_max     -0.441275 -0.509369 -0.412391
opponent_position_mean    -0.859712 -0.245733 -0.128325
opponent_position_median  -0.836900 -0.137300 -0.130748
is_win_visit_mean          0.899937  0.188162  0.024660
dif_goals_visit_sum        0.804947  0.093636  0.334436
dif_goals_visit_min        0.113038 -0.222974  0.688993
dif_goals_visit_max        0.608124  0.394996 -0.303937
dif_goals_visit_mean       0.941700  0.062474  0.089887
dif_goals_visit_median     0.800396 -0.043147  0.092625
is_draw_mean_y             0.088296 -0.471068  0.079038
Total_goals_sum_y         -0.804947 -0.093636 -0.334436
Total_goals_min_y         -0.608124 -0.394996  0.303937
Total_goals_max_y         -0.113038  0.222974 -0.688993
Total_goals_mean_y        -0.941700 -0.062474 -0.089887
Total_goals_median_y      -0.800396  0.043147 -0.092625
is_win_over_2_visit_mean   0.810575  0.379807 -0.013849
is_win_over_3_visit_mean   0.699018  0.478425 -0.003976
is_win_over_4_visit_mean   0.482097  0.540205 -0.028379
is_lost_over_2_visit_mean -0.688325  0.291956 -0.154924
is_lost_over_3_visit_mean -0.560888  0.264368 -0.236960
is_lost_over_4_visit_mean -0.344800  0.362067 -0.340622
squad_size                 0.145852 -0.201838 -0.421553
foreigners_number          0.312977 -0.331212 -0.459986
national_team_players      0.601183 -0.306824 -0.359374
stadium_seats              0.487004 -0.211187 -0.224211
```

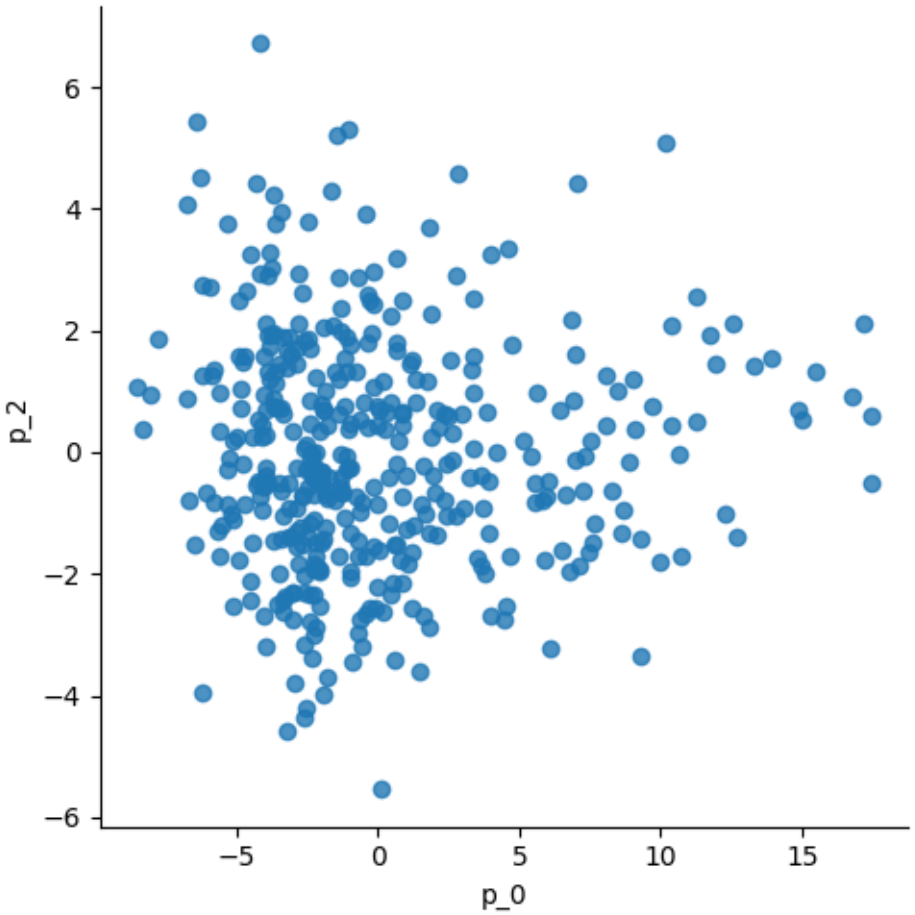
[396]:
```
sns.lmplot(data=Xpca, x='p_0', y='p_1', fit_reg=False)
```

[396]:
```
<seaborn.axisgrid.FacetGrid at 0x7fdebb4ec850>
```



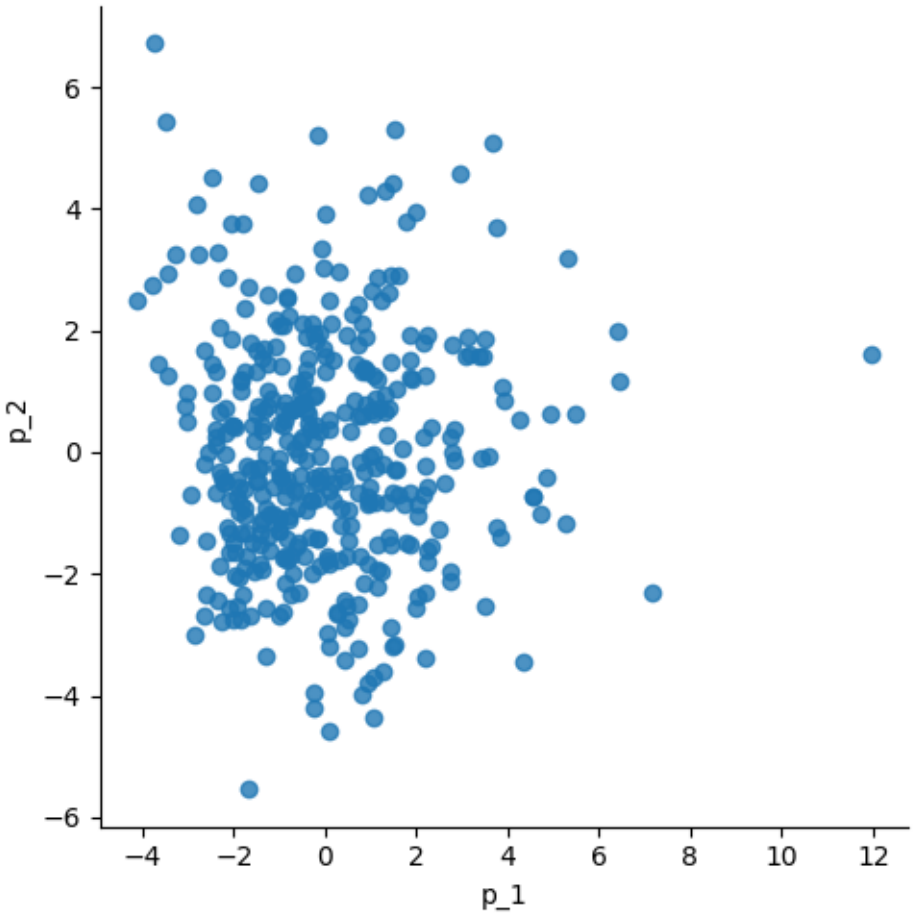
[397]:
```
sns.lmplot(data=Xpca, x='p_0', y='p_2', fit_reg=False)
```

[397]: <seaborn.axisgrid.FacetGrid at 0x7fdebb2ac490>



[398]:
```
sns.lmplot(data=Xpca, x='p_1', y='p_2', fit_reg=False)
```

[398]: <seaborn.axisgrid.FacetGrid at 0x7fdebb3ca980>

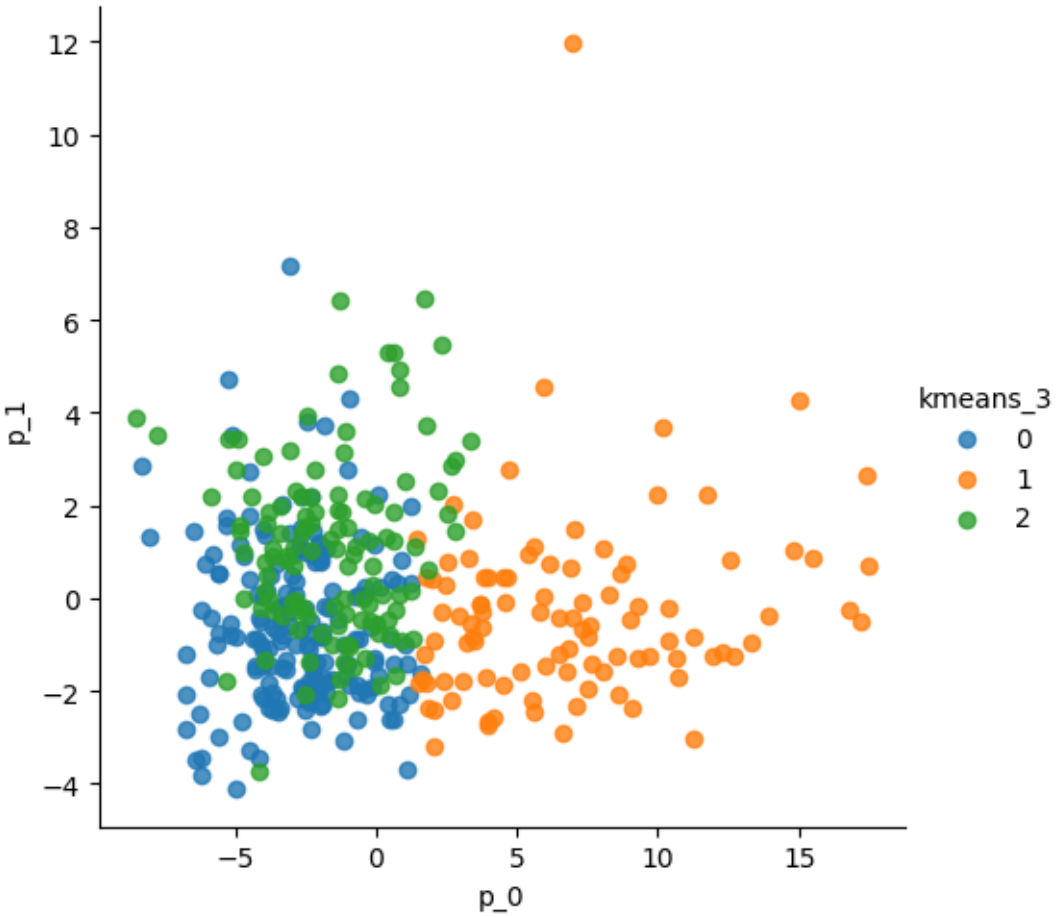


5 PCA para kmeans

[399]:
```
Xpca['kmeans_3']=Xmds_sample['kmeans_mds_3']
```

[400]:
```
sns.lmplot(data=Xpca, x='p_0', y='p_1', fit_reg=False, hue='kmeans_3')
```

[400]:
```
<seaborn.axisgrid.FacetGrid at 0x7fdeba698a30>
```



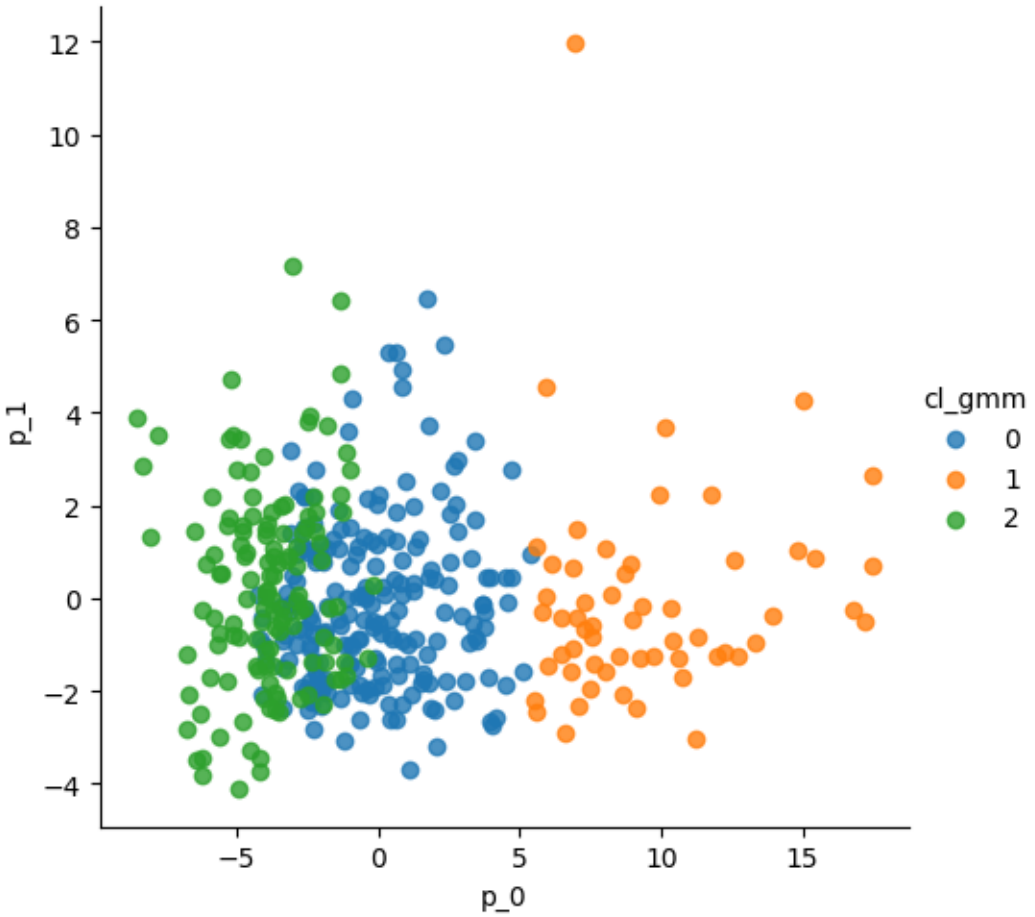
6 GAUSSIANOS MIXTOS

[404]:
```
from sklearn.mixture import GaussianMixture
n_clusters = 3
gmm = GaussianMixture(n_clusters)
gmm.fit(Xsc) # Tiene que ser el Estandarizado, porque recordemos que el␣
 ↪supuesto inicia con Gaussianas
predictions = gmm.predict(Xsc)
Xmds_sample['cl_gmm'] = predictions
```

[405]:
```
Xpca['cl_gmm']=predictions
```

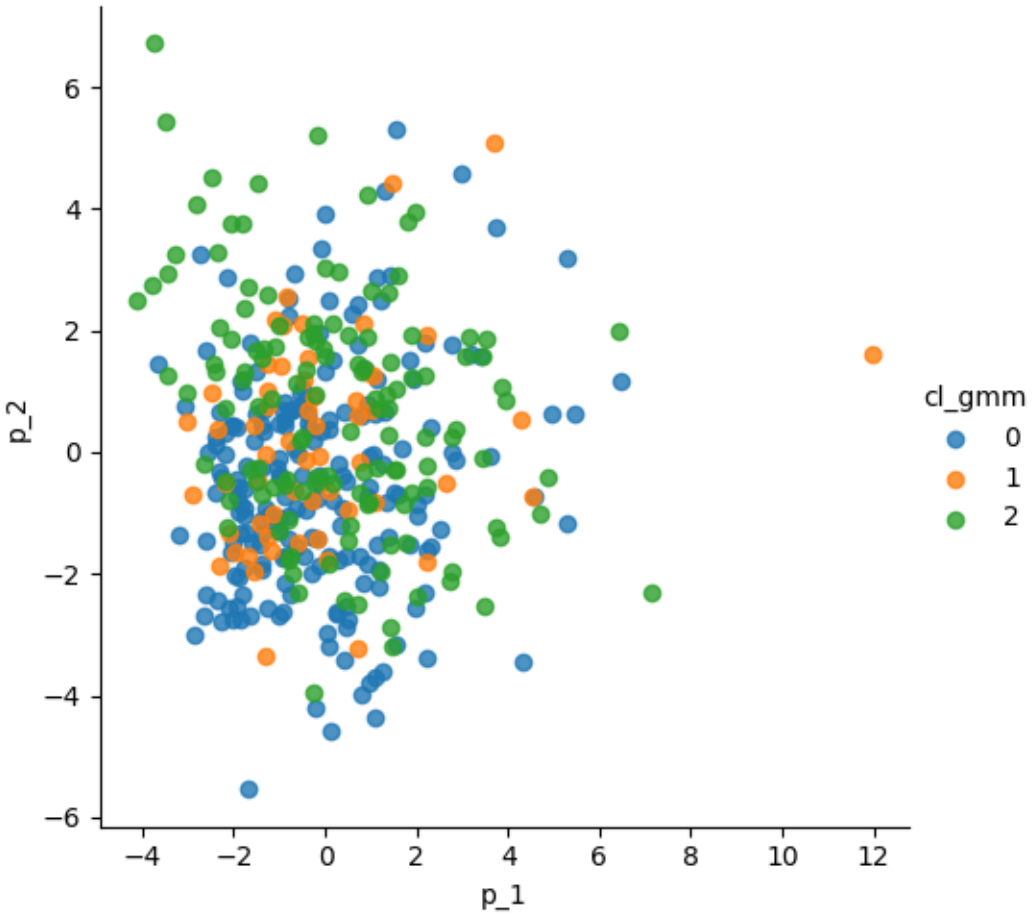
[406]:
```
sns.lmplot(data=Xpca, x='p_0', y='p_1', fit_reg=False, hue='cl_gmm')
```

[406]:
```
<seaborn.axisgrid.FacetGrid at 0x7fdeba59c1c0>
```



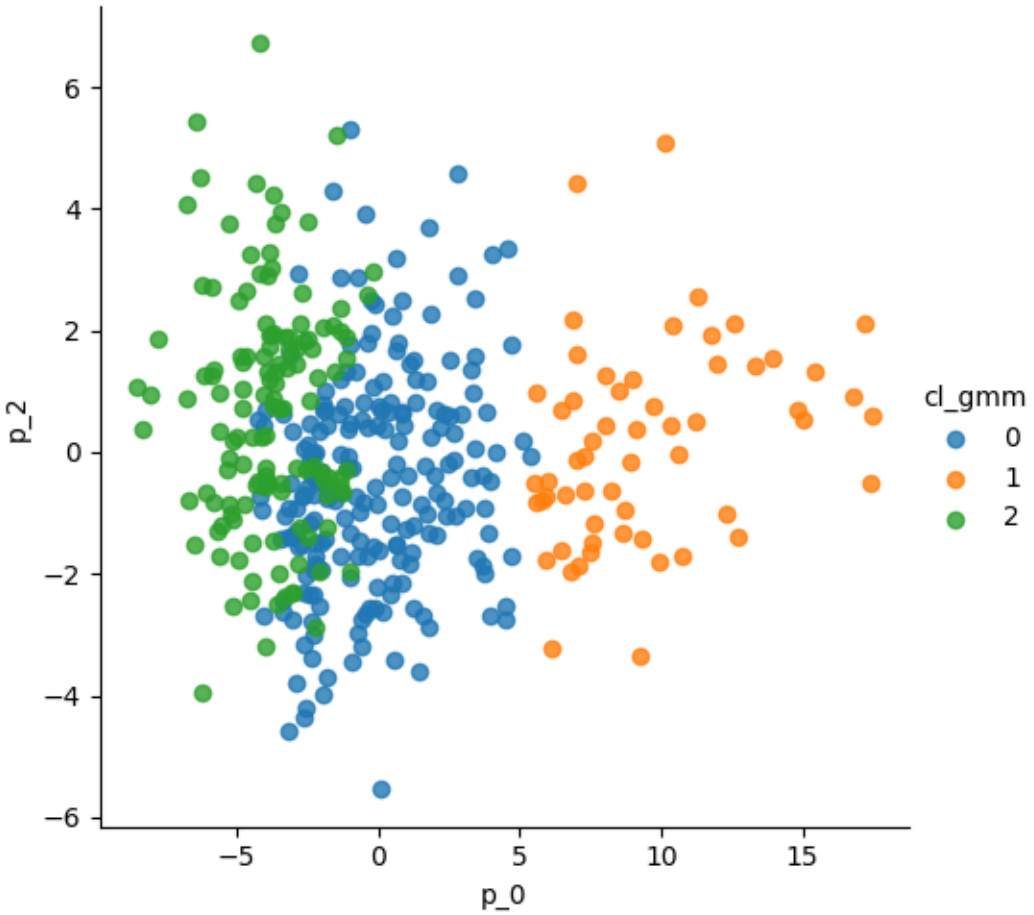
[407]:
```
sns.lmplot(data=Xpca, x='p_1', y='p_2', fit_reg=False, hue='cl_gmm')
```

[407]: <seaborn.axisgrid.FacetGrid at 0x7fdeba945600>

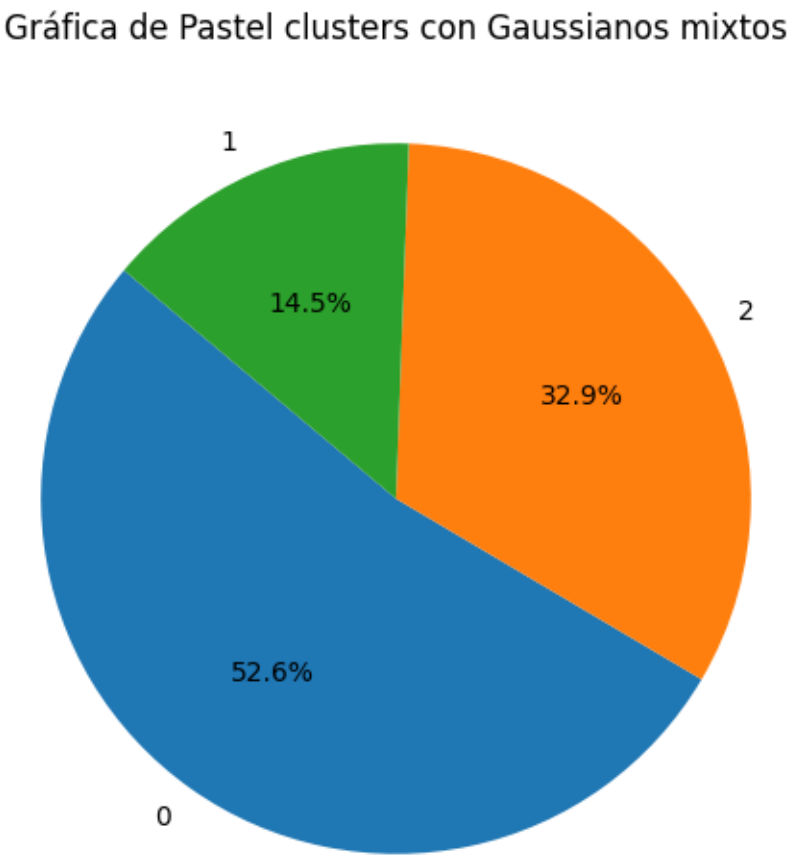


[408]:
```
sns.lmplot(data=Xpca, x='p_0', y='p_2', fit_reg=False, hue='cl_gmm')
```

[408]: <seaborn.axisgrid.FacetGrid at 0x7fdeba352410>



[409]:
```
conteo_clusters = Xmds_sample['cl_gmm'].value_counts()
plt.figure(figsize=(6, 6))
plt.pie(conteo_clusters, labels=conteo_clusters.index, autopct='%1.1f%%',␣
↪startangle=140)
plt.title('Gráfica de Pastel clusters con Gaussianos mixtos')
plt.show()
```



[410]:
```
%%time
grp = 'cl_gmm'
ls_tukey_gaussianos=[]
for c_ in Xmm_sample.columns:
    #print(c_)
    tukey = pairwise_tukeyhsd(
        endog=Xmm_sample[c_],
        groups=Xmds_sample[grp],
        alpha=0.05
        )
    tukey_df = pd.DataFrame(data=tukey._results_table.data[1:], columns=tukey.
 ↪_results_table.data[0])
    if tukey_df['reject'].mean()==1:
        print(f'variable con todos los valores de reject verdadero es {c_}')
        ls_tukey_gaussianos.append(c_)
```

```
variable con todos los valores de reject verdadero es own_goals_count
```

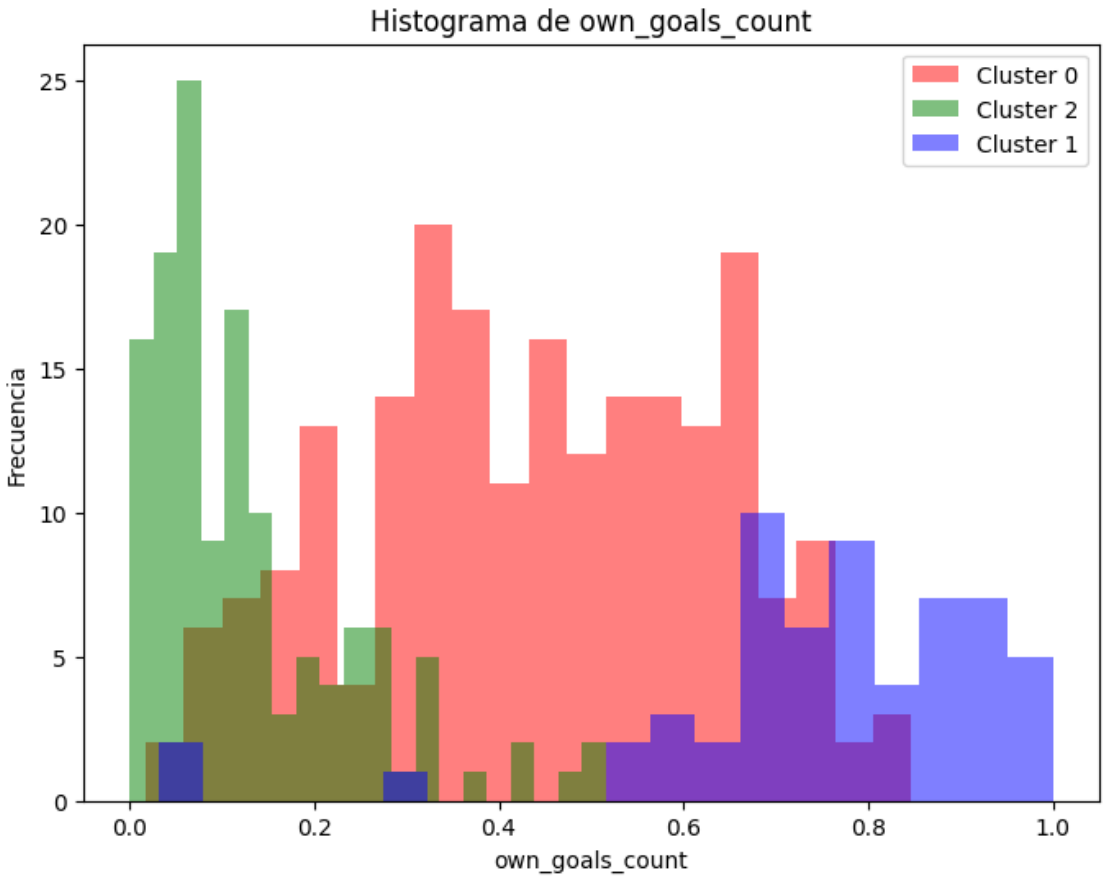
```
variable con todos los valores de reject verdadero es own_goals_sum
variable con todos los valores de reject verdadero es own_goals_max
variable con todos los valores de reject verdadero es own_goals_mean
variable con todos los valores de reject verdadero es own_position_max
variable con todos los valores de reject verdadero es own_position_mean
variable con todos los valores de reject verdadero es own_position_median
variable con todos los valores de reject verdadero es is_win_mean
variable con todos los valores de reject verdadero es dif_goals_loc_sum
variable con todos los valores de reject verdadero es dif_goals_loc_max
variable con todos los valores de reject verdadero es dif_goals_loc_mean
variable con todos los valores de reject verdadero es dif_goals_loc_median
variable con todos los valores de reject verdadero es is_draw_mean_x
variable con todos los valores de reject verdadero es is_win_over_2_local_mean
variable con todos los valores de reject verdadero es is_win_over_3_local_mean
variable con todos los valores de reject verdadero es is_lost_over_2_local_mean
variable con todos los valores de reject verdadero es is_lost_over_3_local_mean
variable con todos los valores de reject verdadero es is_lost_over_4_local_mean
variable con todos los valores de reject verdadero es opponent_goals_sum
variable con todos los valores de reject verdadero es opponent_goals_max
variable con todos los valores de reject verdadero es opponent_goals_mean
variable con todos los valores de reject verdadero es opponent_goals_median
variable con todos los valores de reject verdadero es opponent_position_max
variable con todos los valores de reject verdadero es opponent_position_mean
variable con todos los valores de reject verdadero es opponent_position_median
variable con todos los valores de reject verdadero es is_win_visit_mean
variable con todos los valores de reject verdadero es dif_goals_visit_sum
variable con todos los valores de reject verdadero es dif_goals_visit_max
variable con todos los valores de reject verdadero es dif_goals_visit_mean
variable con todos los valores de reject verdadero es dif_goals_visit_median
variable con todos los valores de reject verdadero es Total_goals_sum_y
variable con todos los valores de reject verdadero es Total_goals_min_y
variable con todos los valores de reject verdadero es Total_goals_mean_y
variable con todos los valores de reject verdadero es Total_goals_median_y
variable con todos los valores de reject verdadero es is_win_over_2_visit_mean
variable con todos los valores de reject verdadero es is_win_over_3_visit_mean
variable con todos los valores de reject verdadero es is_lost_over_2_visit_mean
variable con todos los valores de reject verdadero es is_lost_over_3_visit_mean
variable con todos los valores de reject verdadero es is_lost_over_4_visit_mean
variable con todos los valores de reject verdadero es foreigners_number
variable con todos los valores de reject verdadero es national_team_players
variable con todos los valores de reject verdadero es stadium_seats
CPU times: user 9.2 s, sys: 157 ms, total: 9.36 s
Wall time: 9.15 s
```

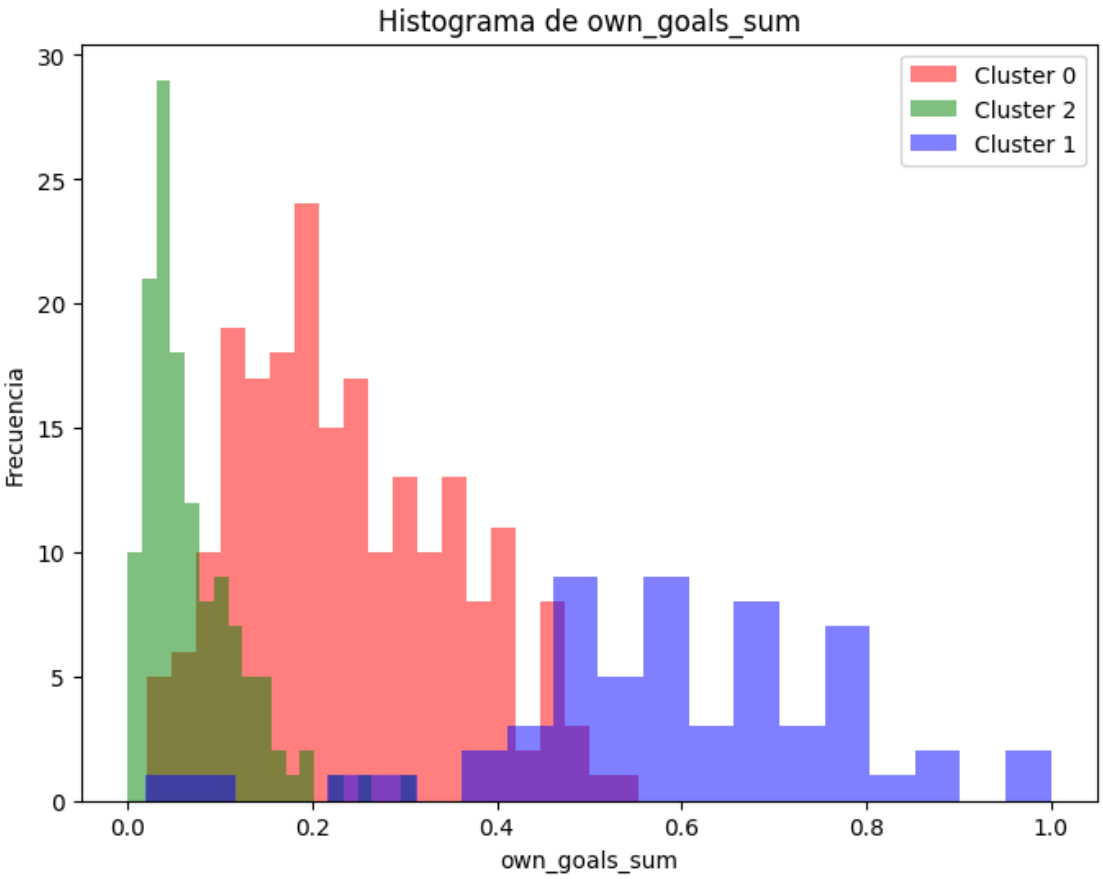
[411]:
```
ls_tukey_gaussianos
```

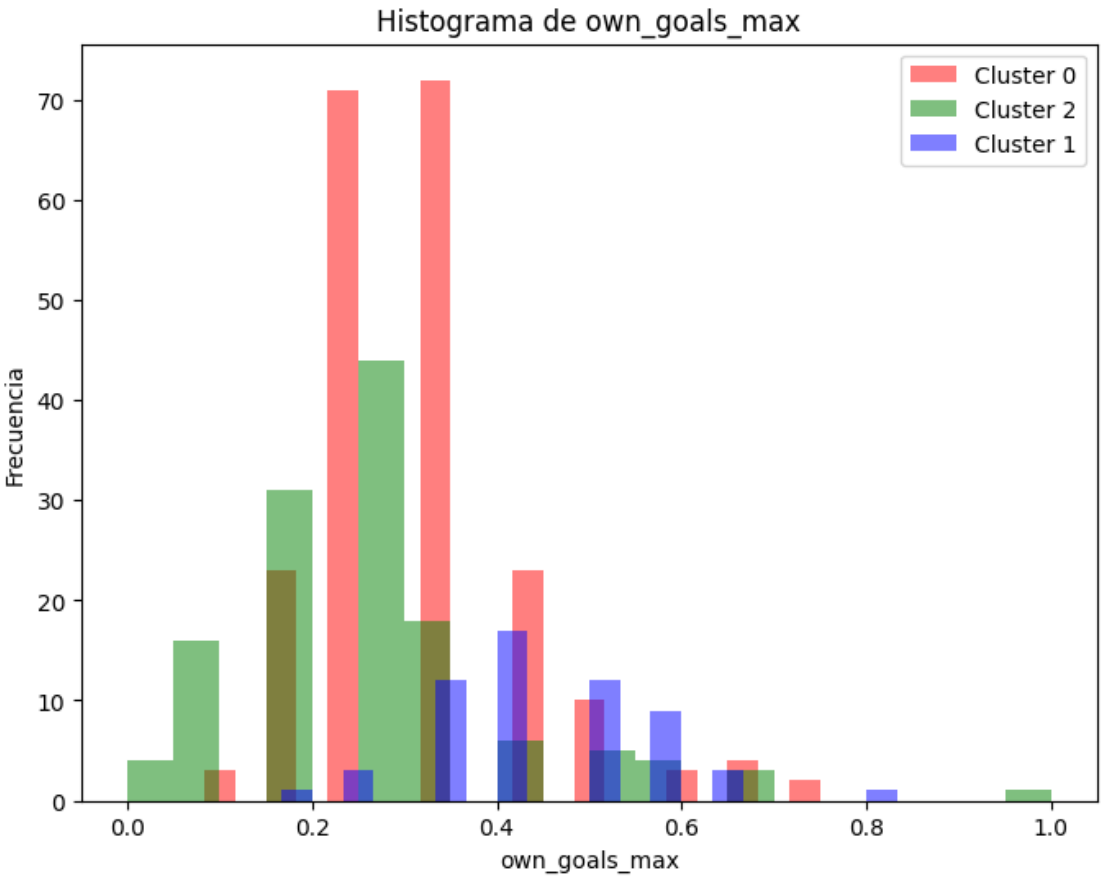
[411]:
```
['own_goals_count',
 'own_goals_sum',
 'own_goals_max',
 'own_goals_mean',
 'own_position_max',
 'own_position_mean',
 'own_position_median',
 'is_win_mean',
 'dif_goals_loc_sum',
 'dif_goals_loc_max',
 'dif_goals_loc_mean',
 'dif_goals_loc_median',
 'is_draw_mean_x',
 'is_win_over_2_local_mean',
 'is_win_over_3_local_mean',
 'is_lost_over_2_local_mean',
 'is_lost_over_3_local_mean',
 'is_lost_over_4_local_mean',
 'opponent_goals_sum',
 'opponent_goals_max',
 'opponent_goals_mean',
 'opponent_goals_median',
 'opponent_position_max',
 'opponent_position_mean',
 'opponent_position_median',
 'is_win_visit_mean',
 'dif_goals_visit_sum',
 'dif_goals_visit_max',
 'dif_goals_visit_mean',
 'dif_goals_visit_median',
 'Total_goals_sum_y',
 'Total_goals_min_y',
 'Total_goals_mean_y',
 'Total_goals_median_y',
 'is_win_over_2_visit_mean',
 'is_win_over_3_visit_mean',
 'is_lost_over_2_visit_mean',
 'is_lost_over_3_visit_mean',
 'is_lost_over_4_visit_mean',
 'foreigners_number',
 'national_team_players',
 'stadium_seats']
```

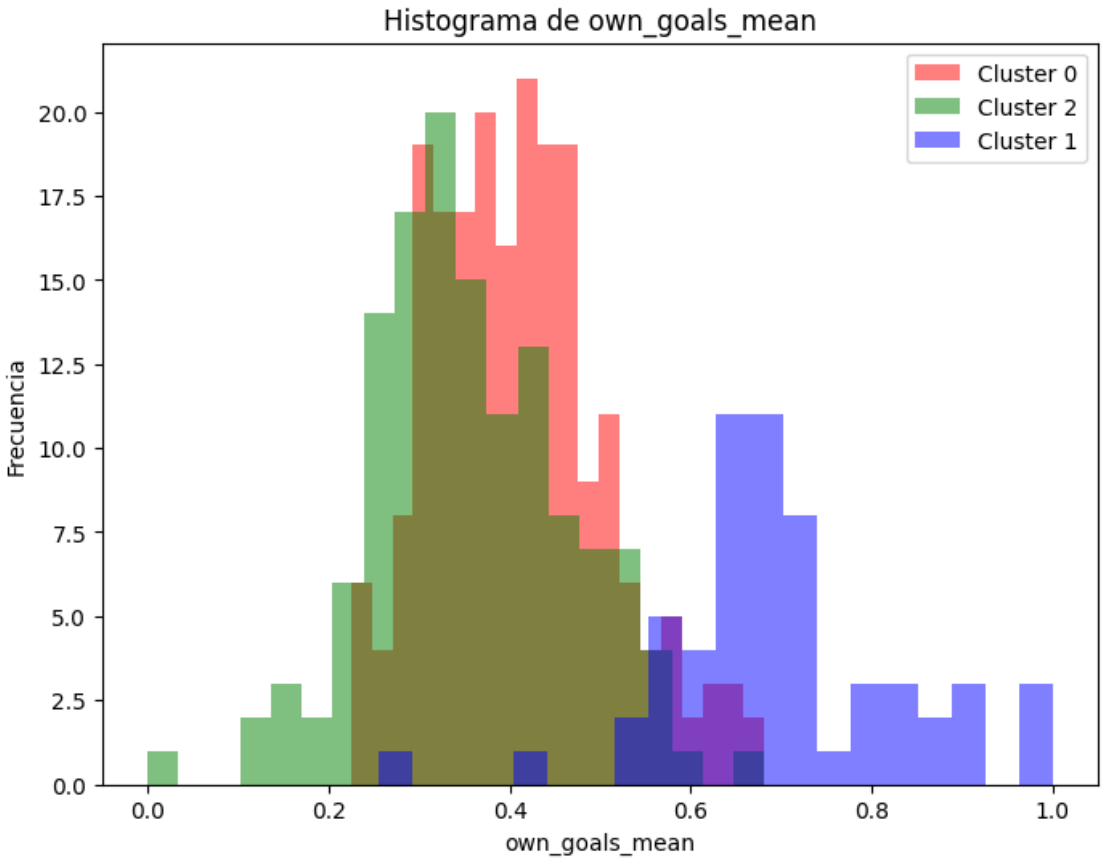
[412]:
```
unique_clusters = Xmds_sample['cl_gmm'].unique()

# Color mapping para kmeans_mds_3
color_mapping = {0: 'red', 1: 'blue', 2: 'green'}
```

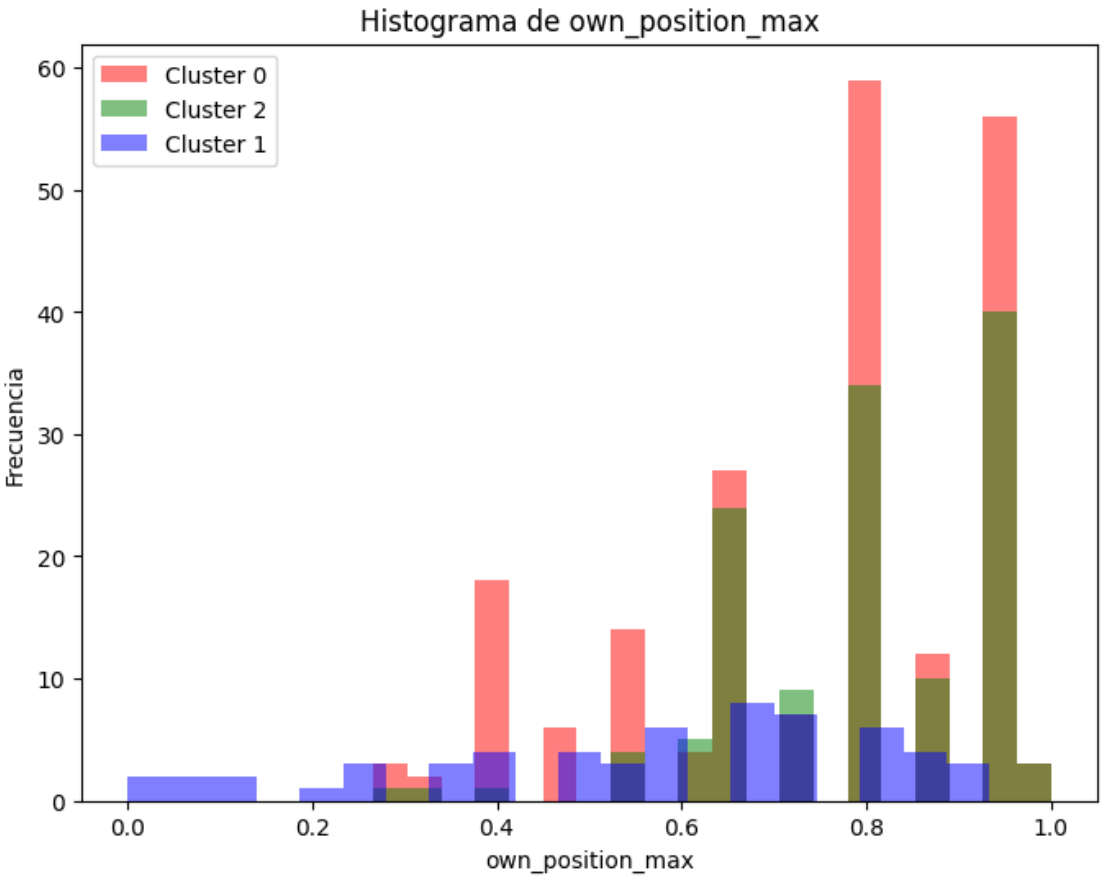
```
# Itera a través de las variables en ls_best
for variable in ls_tukey_gaussianos:
    # Crear un nuevo histograma para la variable actual
    plt.figure(figsize=(8, 6))  # Establece el tamaño de la figura (opcional)

    # Itera a través de los valores únicos de kmeans_mds_3
    for cluster_value in unique_clusters:
        # Restablece el índice del DataFrame Xmm_sample antes de la selección
        subset_data = Xmm_sample.reset_index(drop=True)[Xmds_sample['cl_gmm']
 ↪== cluster_value][variable]

        # Crea el histograma utilizando solo un color para este conjunto de
 ↪datos
        plt.hist(subset_data, bins=20, color=color_mapping[cluster_value],
 ↪alpha=0.5, label=f'Cluster {cluster_value}')

    # Configura el título y etiquetas de los ejes
    plt.title(f'Histograma de {variable}')
    plt.xlabel(variable)
    plt.ylabel('Frecuencia')

    # Agrega una leyenda para identificar los clusters
    plt.legend()

    # Muestra la gráfica
    plt.savefig(f'{variable}_hist.png')
    plt.show()
```

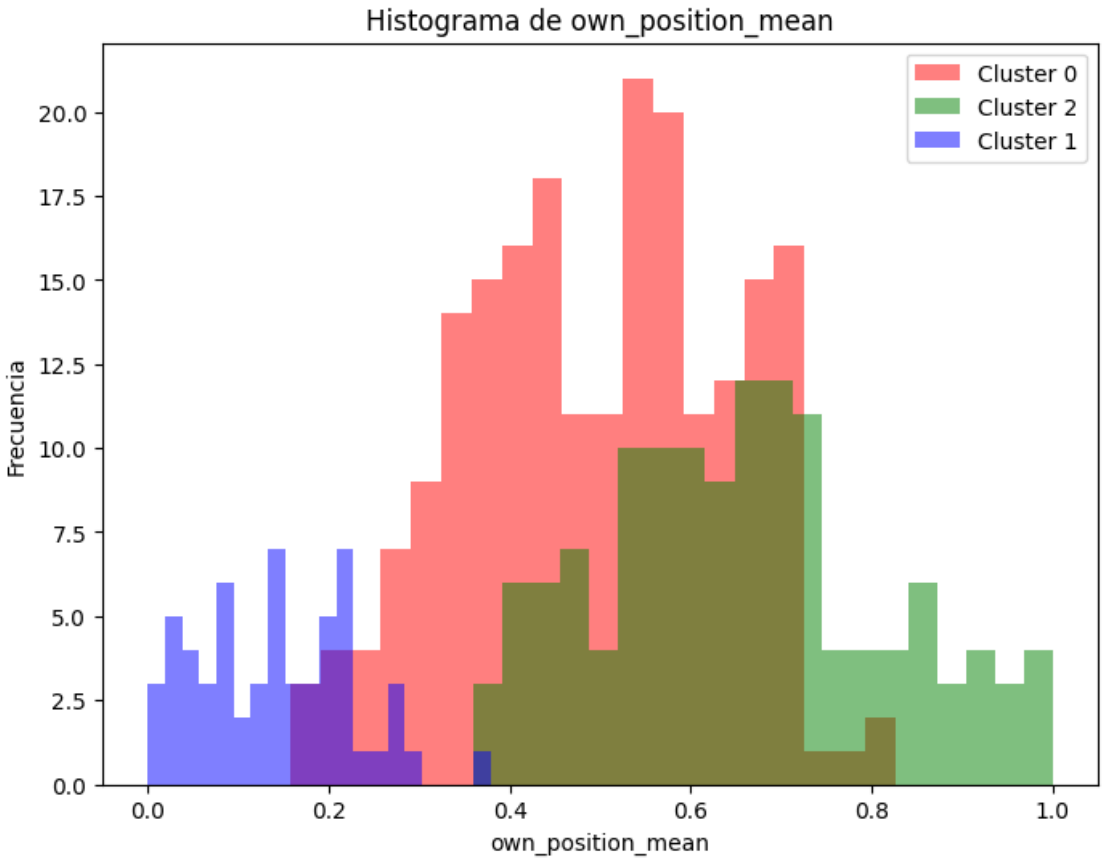

Histograma de own_goals_count
Cluster 0
Cluster 2
Cluster 1
Frecuencia
own_goals_count
0
5
10
15
20
25
0.0
0.2
0.4
0.6
0.8
1.0

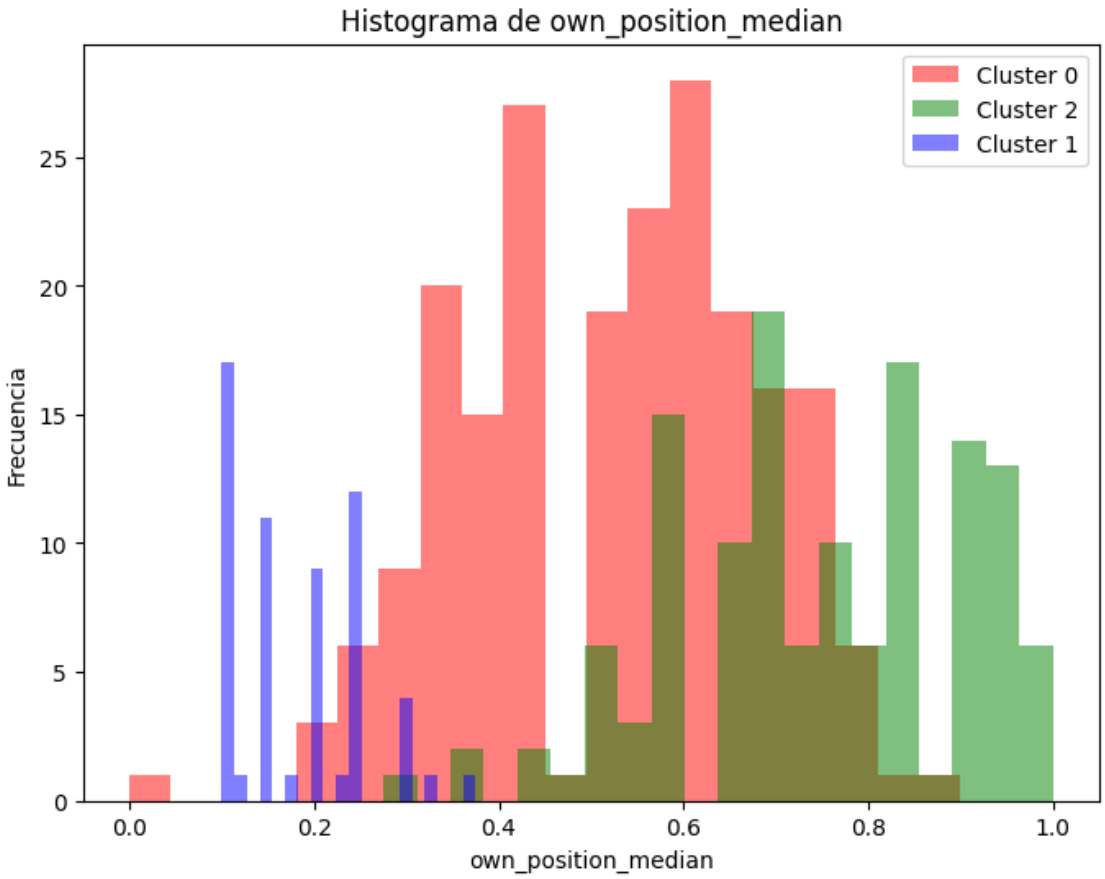

Histograma de own_goals_sum
Cluster 0
Cluster 2
Cluster 1
Frecuencia
own_goals_sum
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0

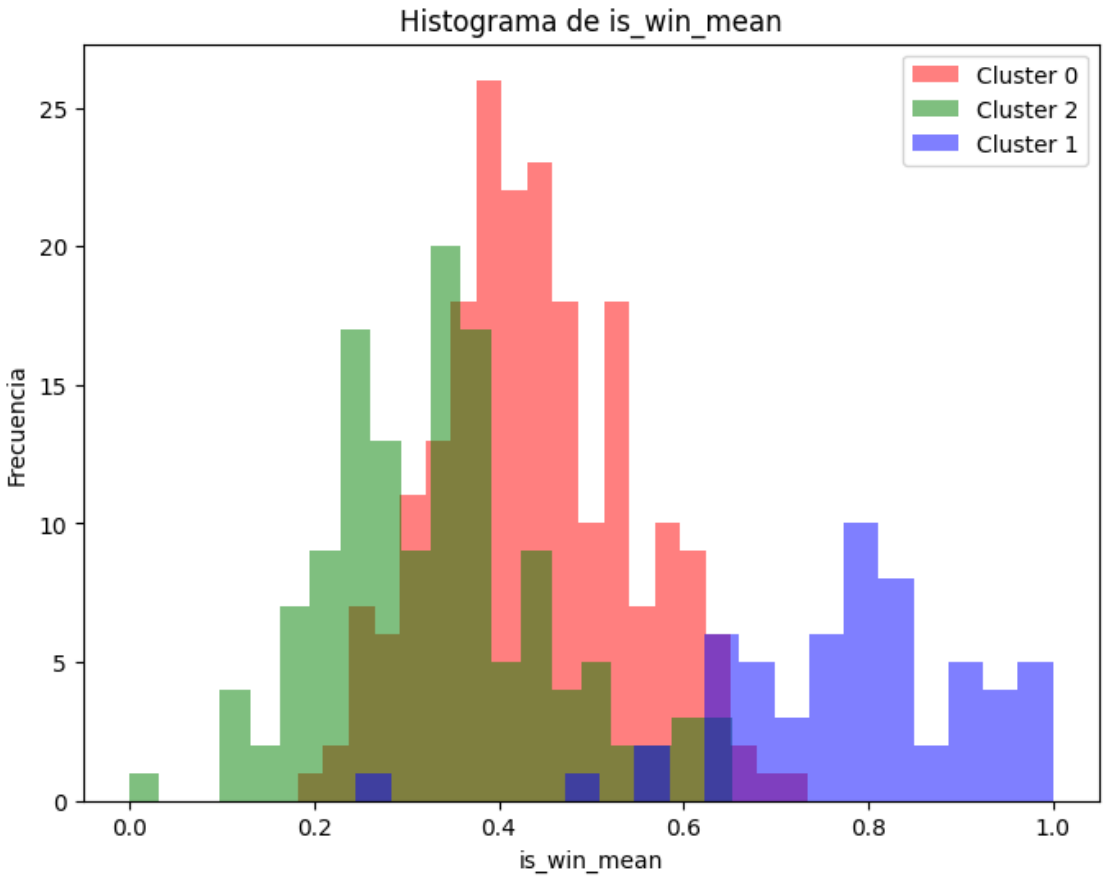

Histograma de own_goals_max
Cluster 0
Cluster 2
Cluster 1
Frecuencia
own_goals_max
0
10
20
30
40
50
60
70
0.0
0.2
0.4
0.6
0.8
1.0

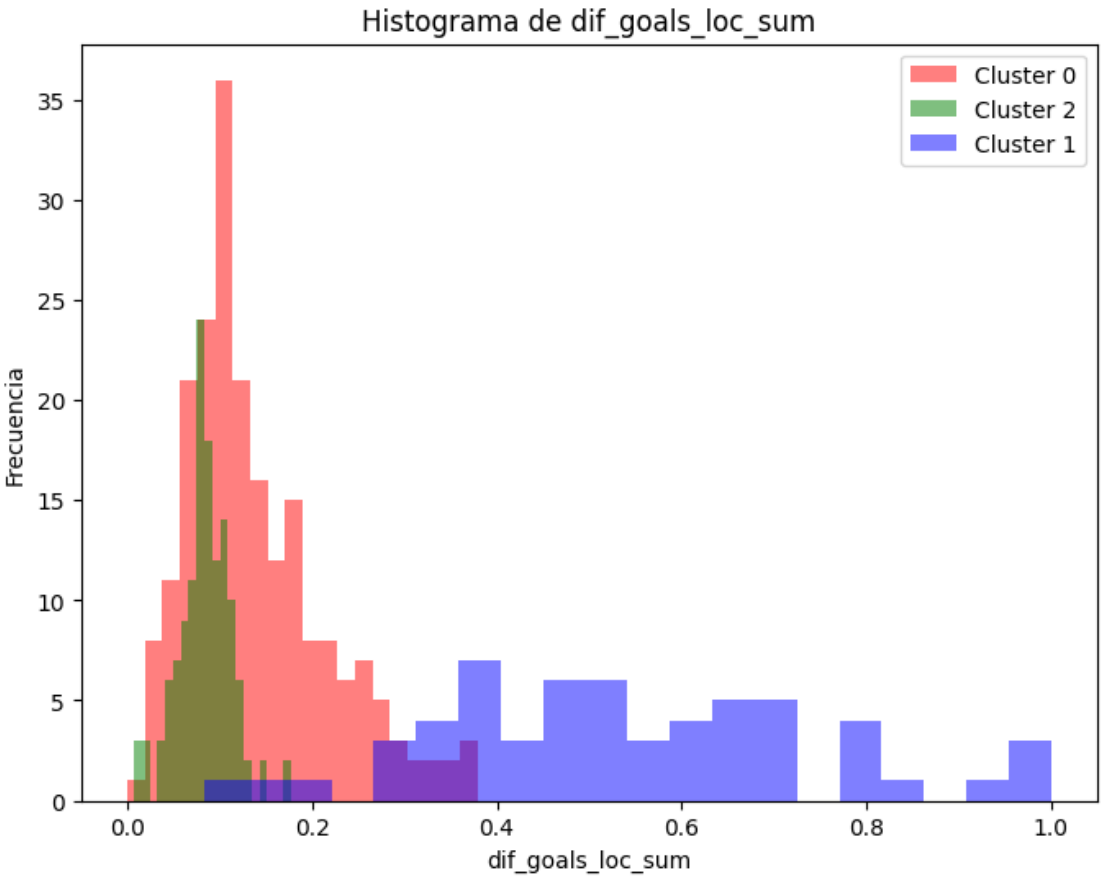

Histograma de own_goals_mean
Cluster 0
Cluster 2
Cluster 1
Frecuencia
own_goals_mean
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
0.0
0.2
0.4
0.6
0.8
1.0

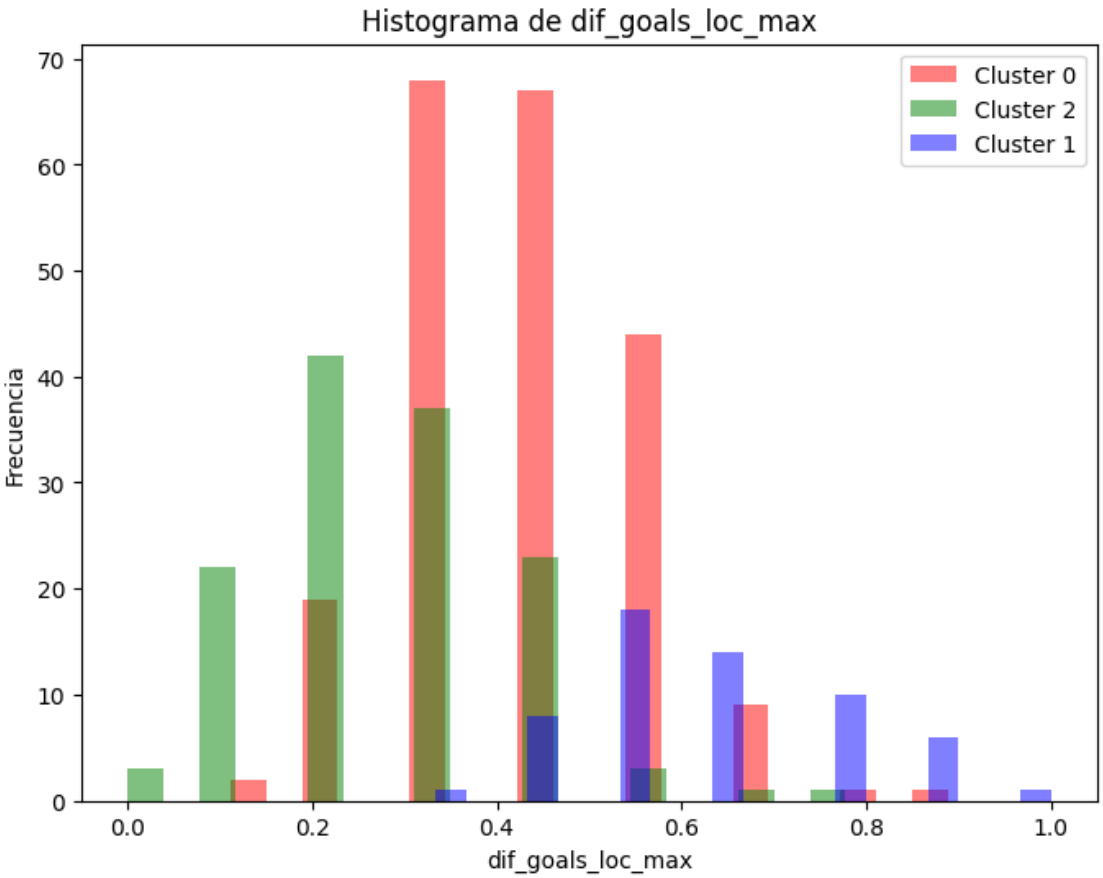

Histograma de own_position_max
Cluster 0
Cluster 2
Cluster 1
Frecuencia
own_position_max
0
10
20
30
40
50
60
0.0
0.2
0.4
0.6
0.8
1.0

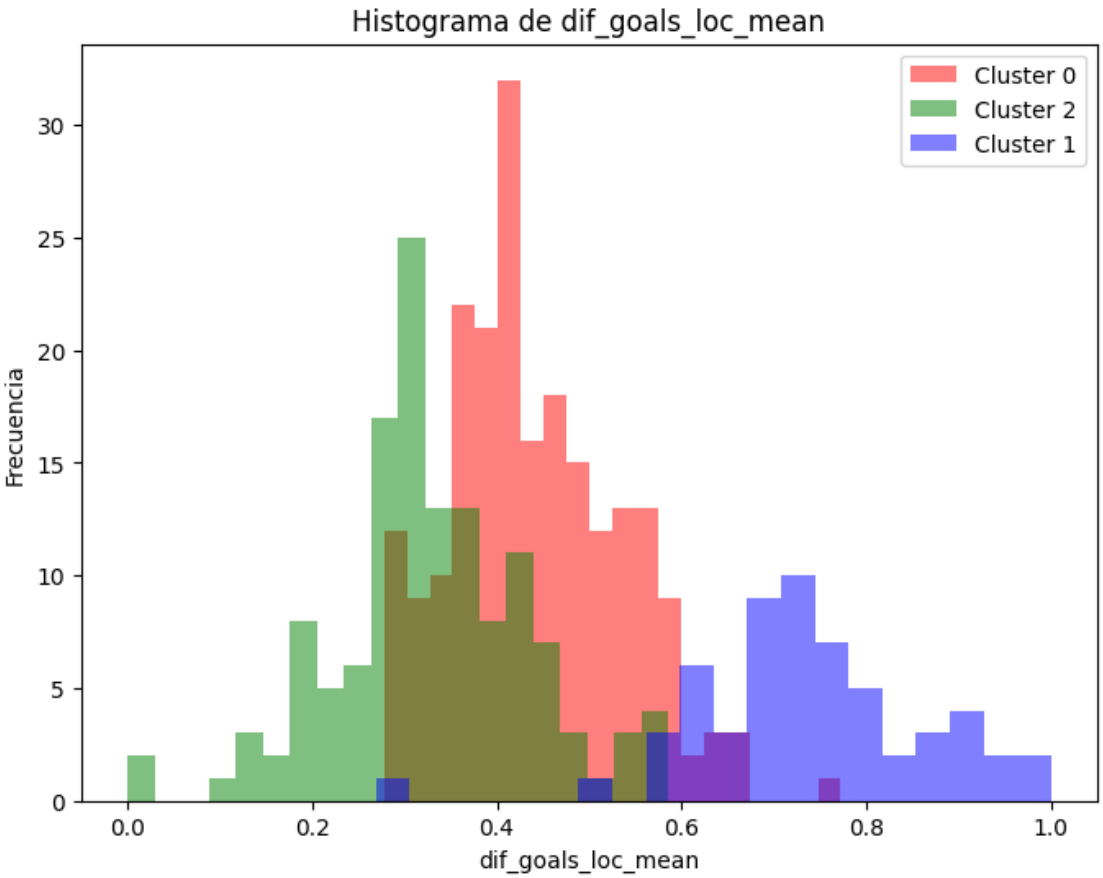

Histograma de own_position_mean
Cluster 0
Cluster 2
Cluster 1
Frecuencia
own_position_mean
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
0.0
0.2
0.4
0.6
0.8
1.0

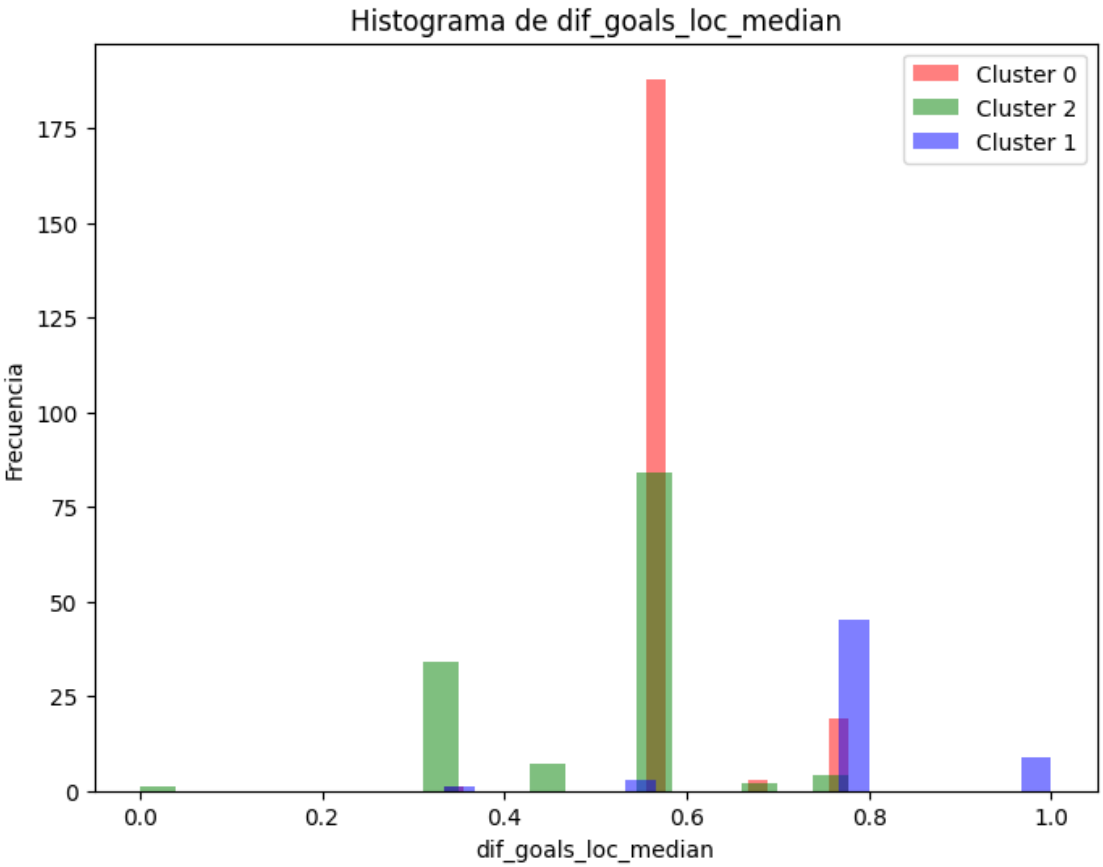

Histograma de own_position_median
Cluster 0
Cluster 2
Cluster 1
Frecuencia
0
5
10
15
20
25
0.0
0.2
0.4
0.6
0.8
1.0
own_position_median

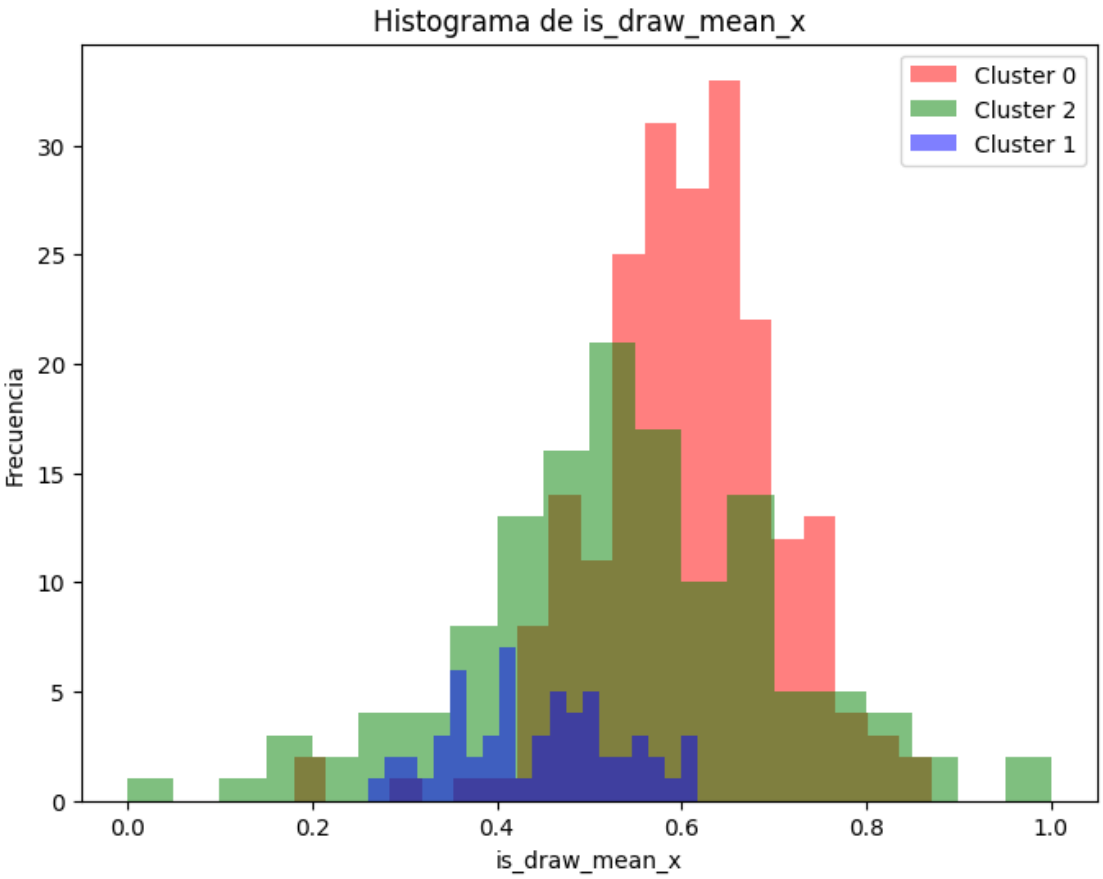

Histograma de is_win_mean
Cluster 0
Cluster 2
Cluster 1
Frecuencia
is_win_mean
0
5
10
15
20
25
0.0
0.2
0.4
0.6
0.8
1.0

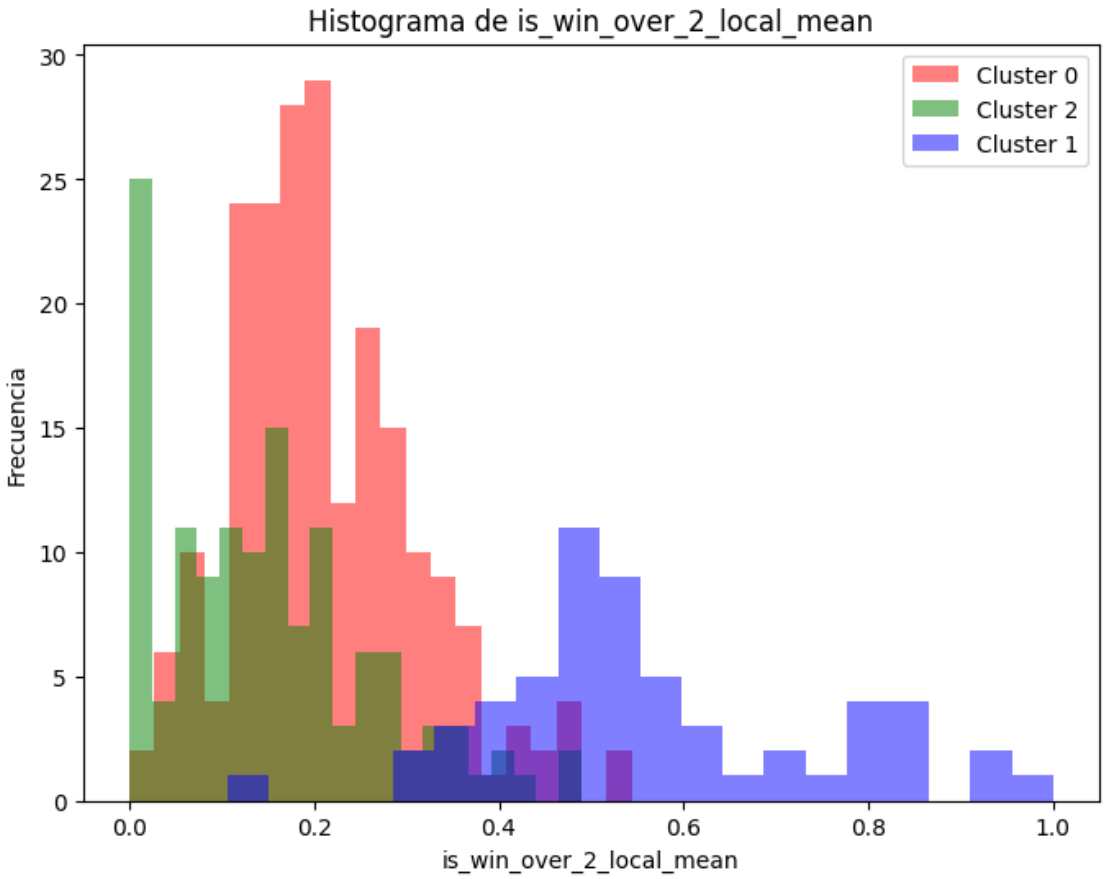

Histograma de dif_goals_loc_sum
Cluster 0
Cluster 2
Cluster 1
Frecuencia
dif_goals_loc_sum
0
5
10
15
20
25
30
35
0.0
0.2
0.4
0.6
0.8
1.0

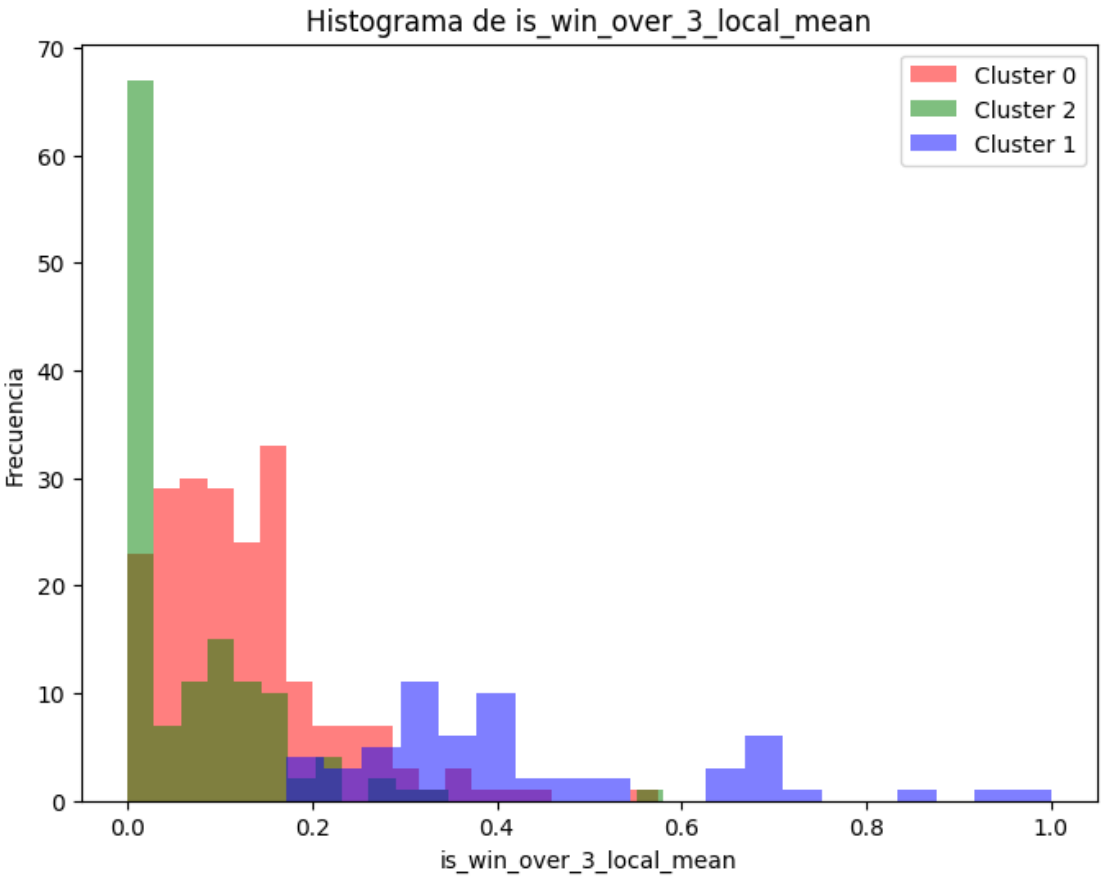

Histograma de dif_goals_loc_max
Cluster 0
Cluster 2
Cluster 1
Frecuencia
dif_goals_loc_max
0
10
20
30
40
50
60
70
0.0
0.2
0.4
0.6
0.8
1.0

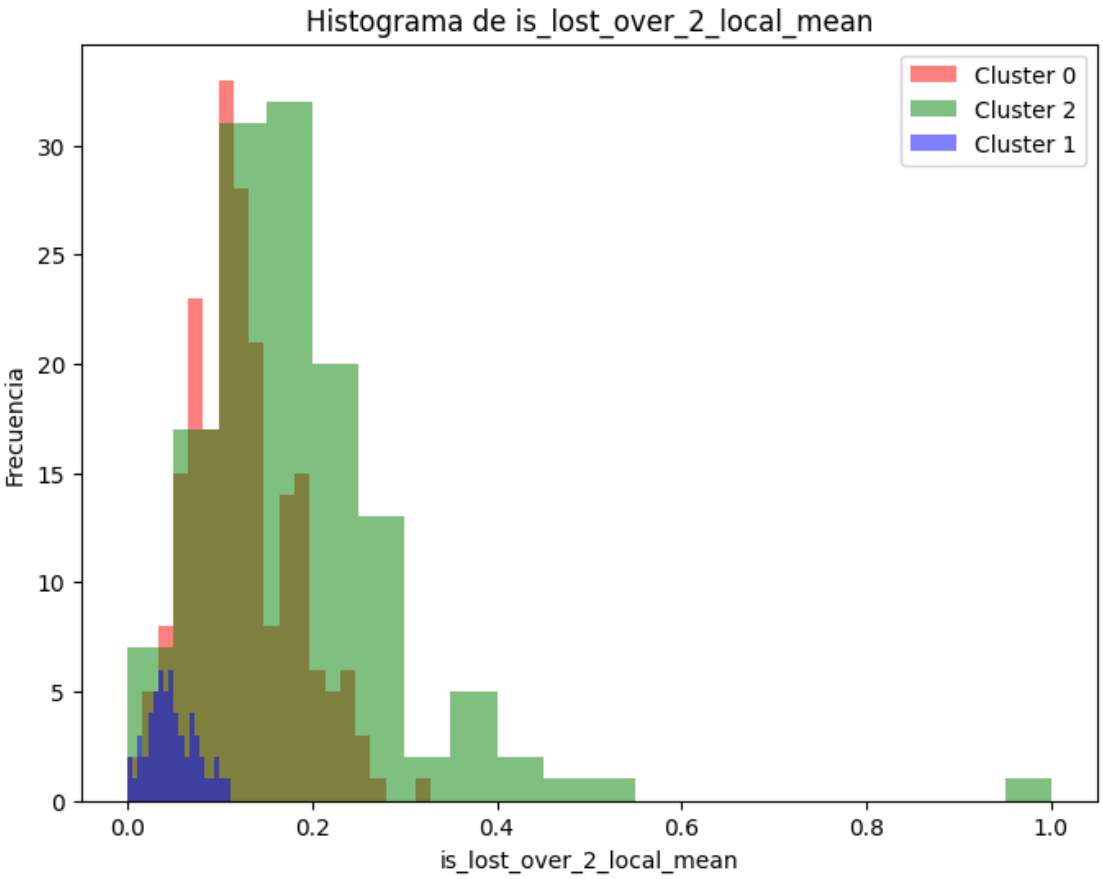

Histograma de dif_goals_loc_mean
Cluster 0
Cluster 2
Cluster 1
Frecuencia
dif_goals_loc_mean
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0

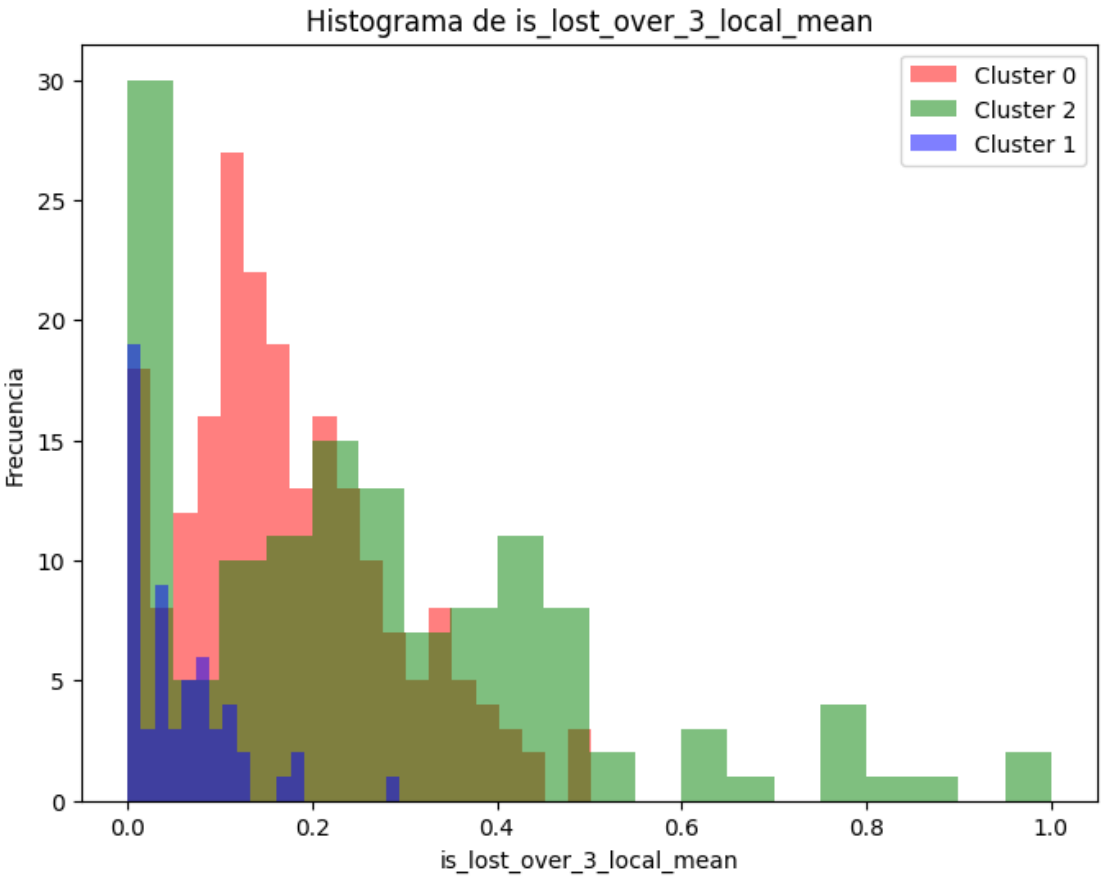

Histograma de dif_goals_loc_median
Cluster 0
Cluster 2
Cluster 1
Frecuencia
175
150
125
100
75
50
25
0
0.0
0.2
0.4
0.6
0.8
1.0
dif_goals_loc_median

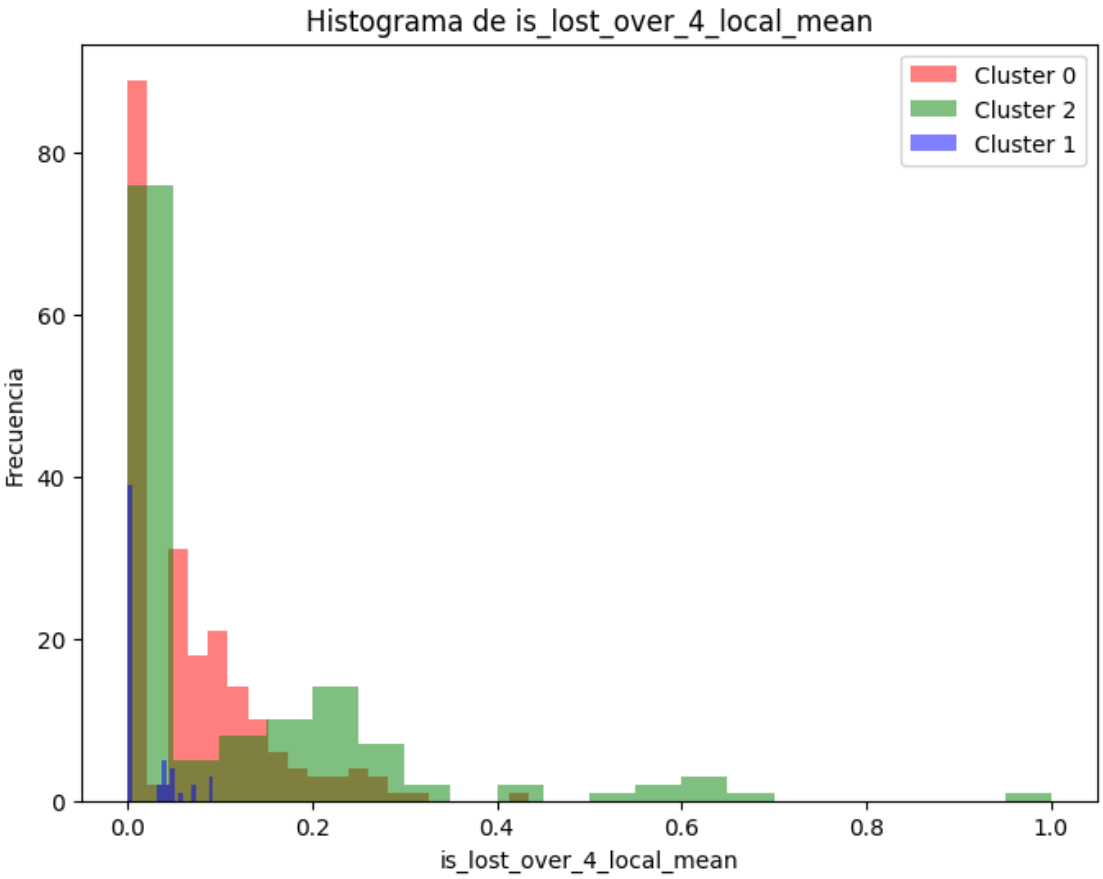

Histograma de is_draw_mean_x
Cluster 0
Cluster 2
Cluster 1
Frecuencia
is_draw_mean_x
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0

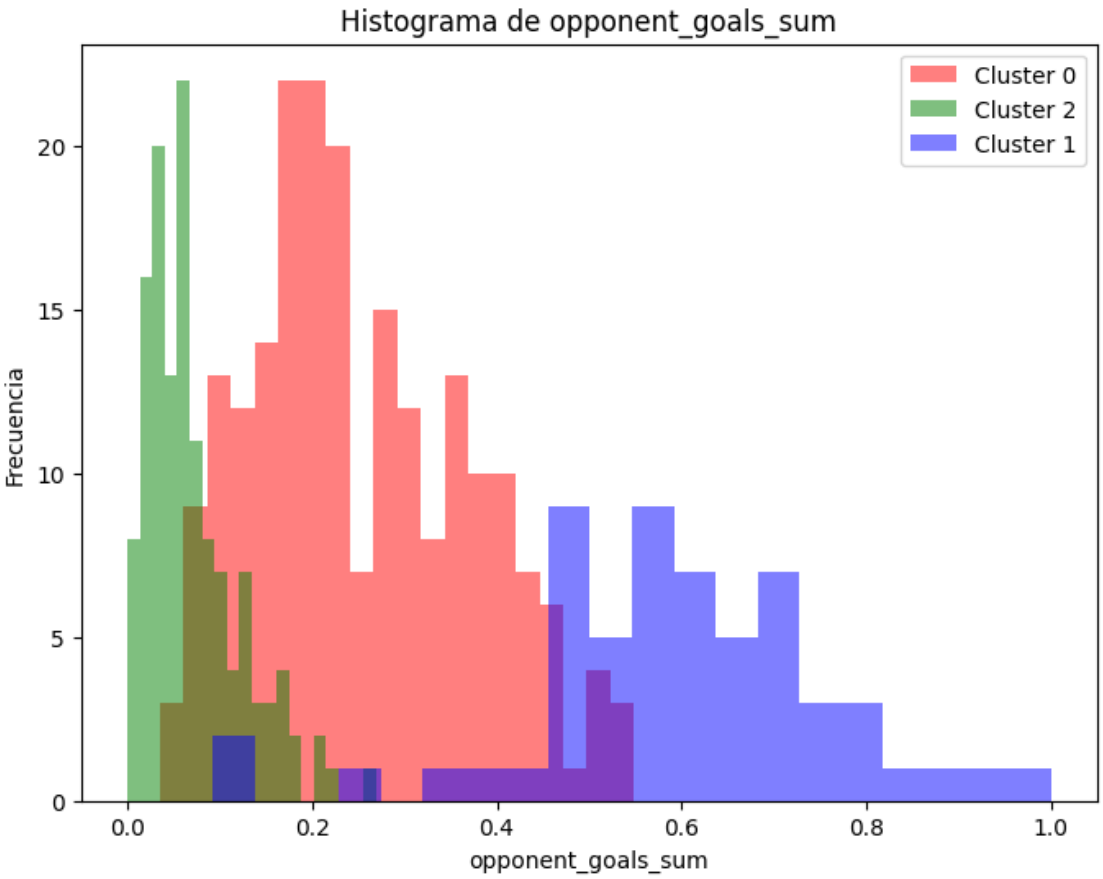

Histograma de is_win_over_2_local_mean
Cluster 0
Cluster 2
Cluster 1
Frecuencia
is_win_over_2_local_mean
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0

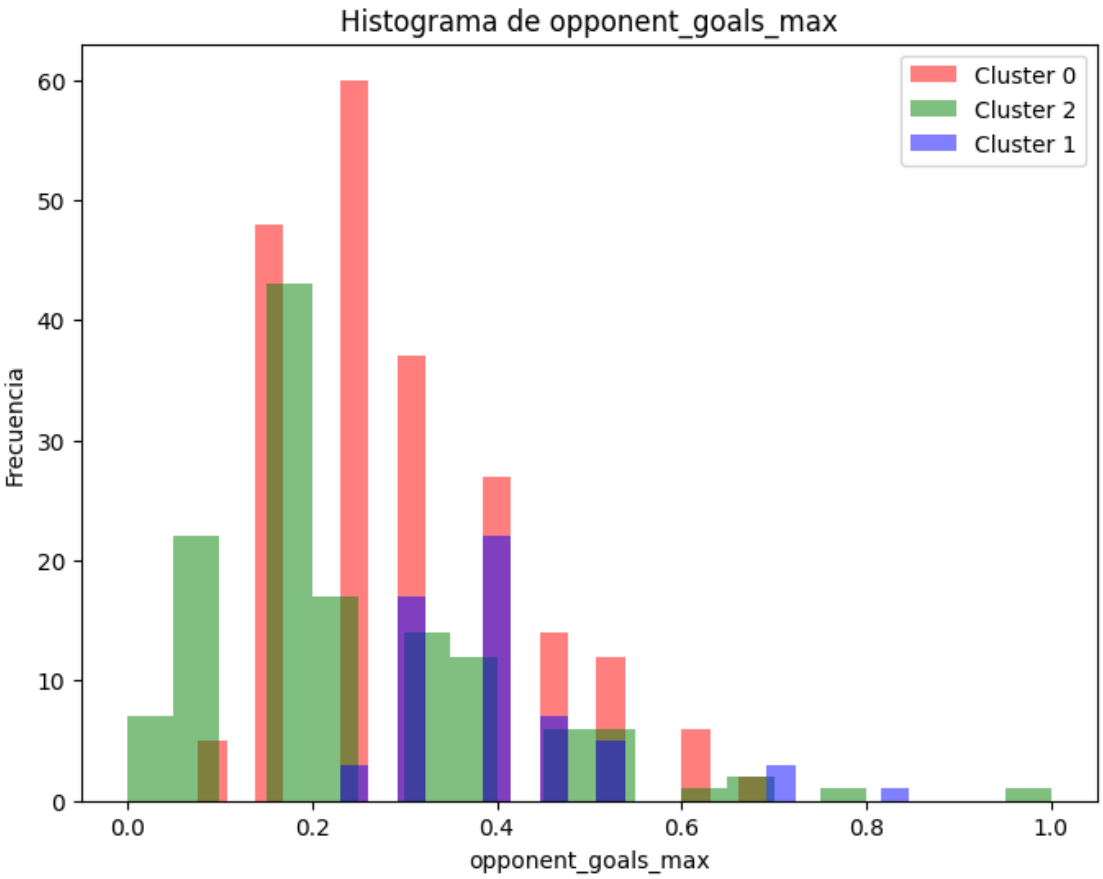

Histograma de is_win_over_3_local_mean
Cluster 0
Cluster 2
Cluster 1
Frecuencia
is_win_over_3_local_mean
0
10
20
30
40
50
60
70
0.0
0.2
0.4
0.6
0.8
1.0

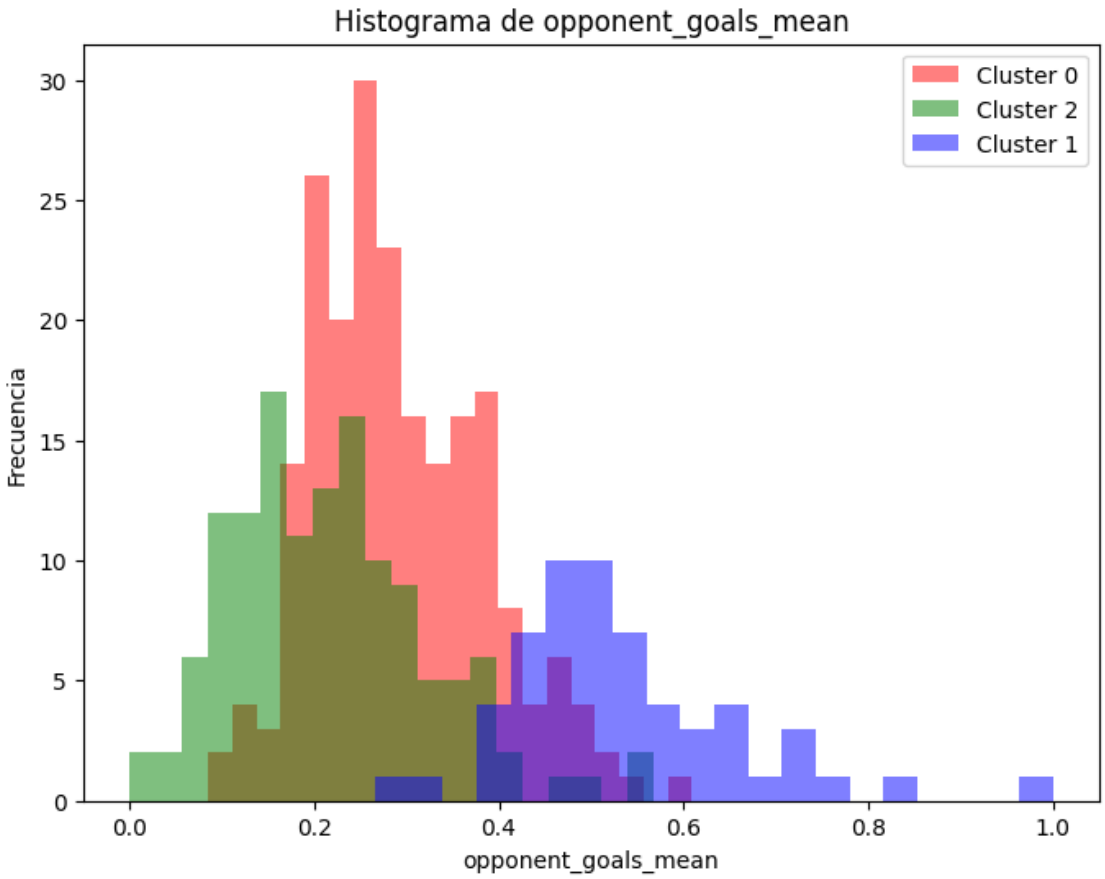

Histograma de is_lost_over_2_local_mean
Cluster 0
Cluster 2
Cluster 1
Frecuencia
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0
is_lost_over_2_local_mean

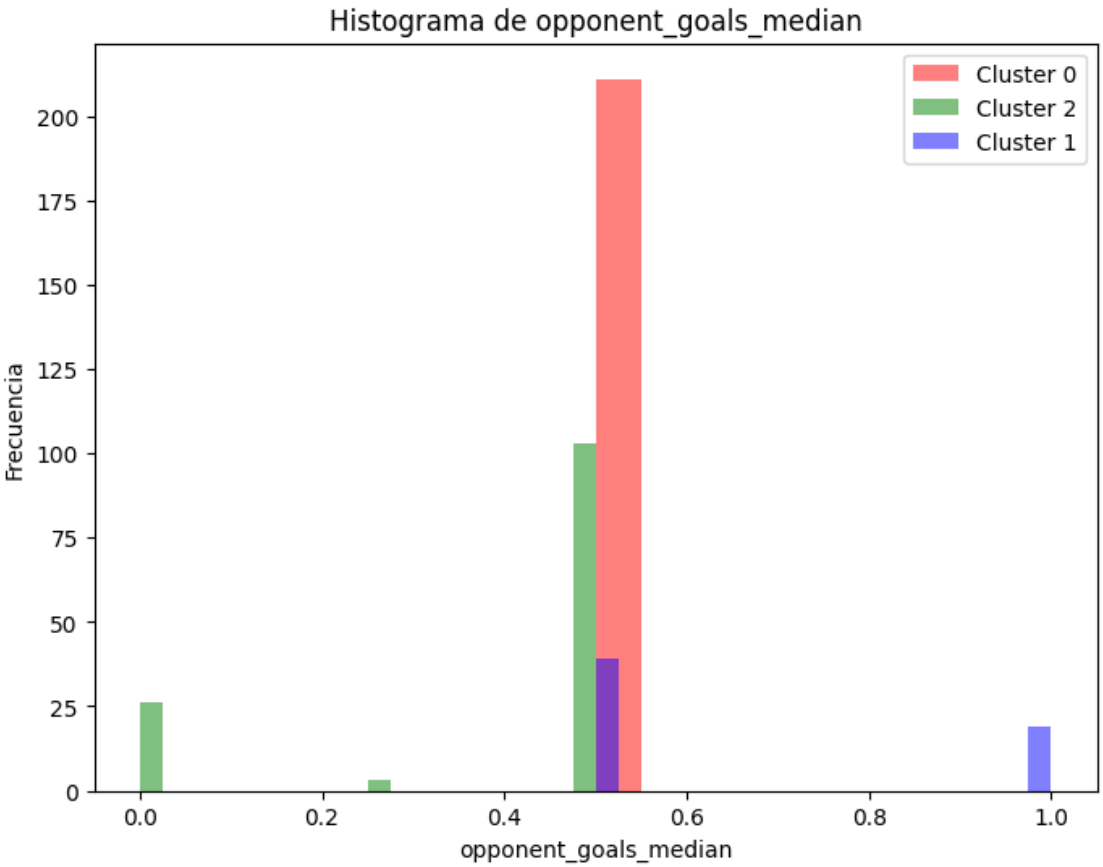

Histograma de is_lost_over_3_local_mean
Cluster 0
Cluster 2
Cluster 1
Frecuencia
is_lost_over_3_local_mean
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0

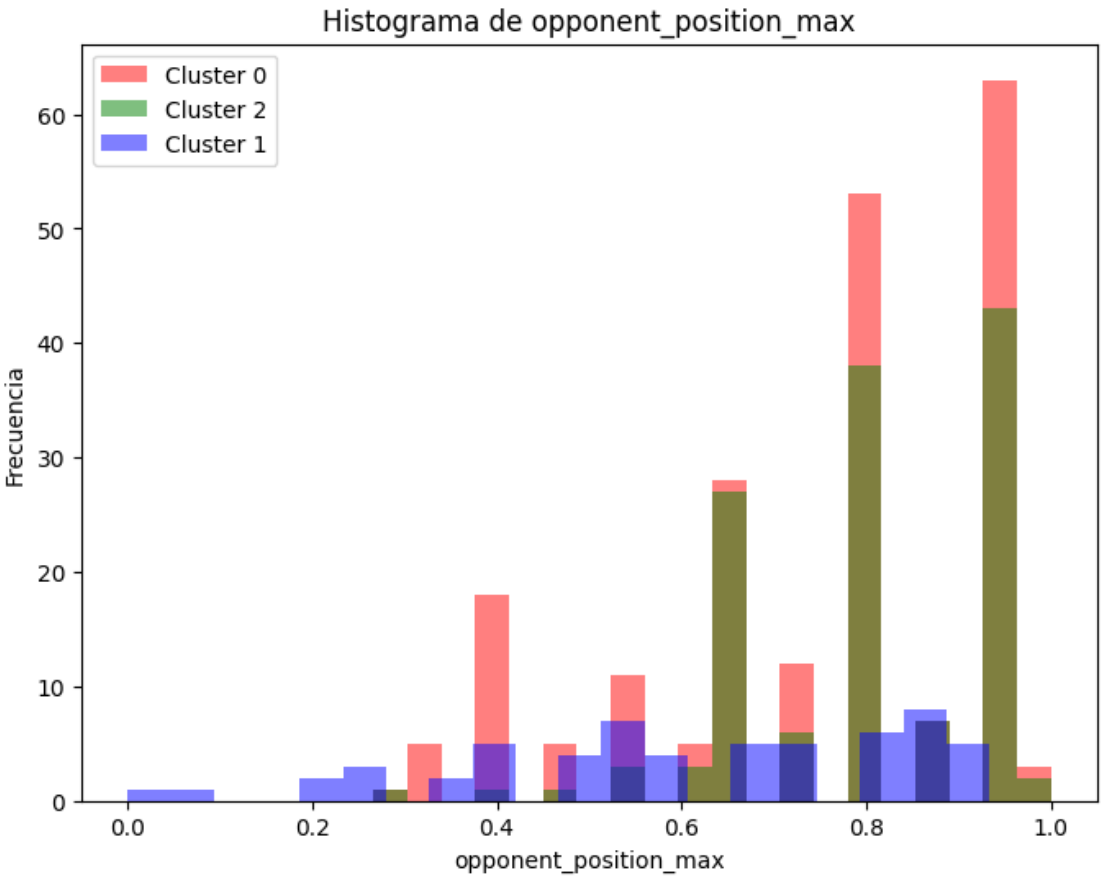

Histograma de is_lost_over_4_local_mean
Cluster 0
Cluster 2
Cluster 1
Frecuencia
0
20
40
60
80
0.0
0.2
0.4
0.6
0.8
1.0
is_lost_over_4_local_mean

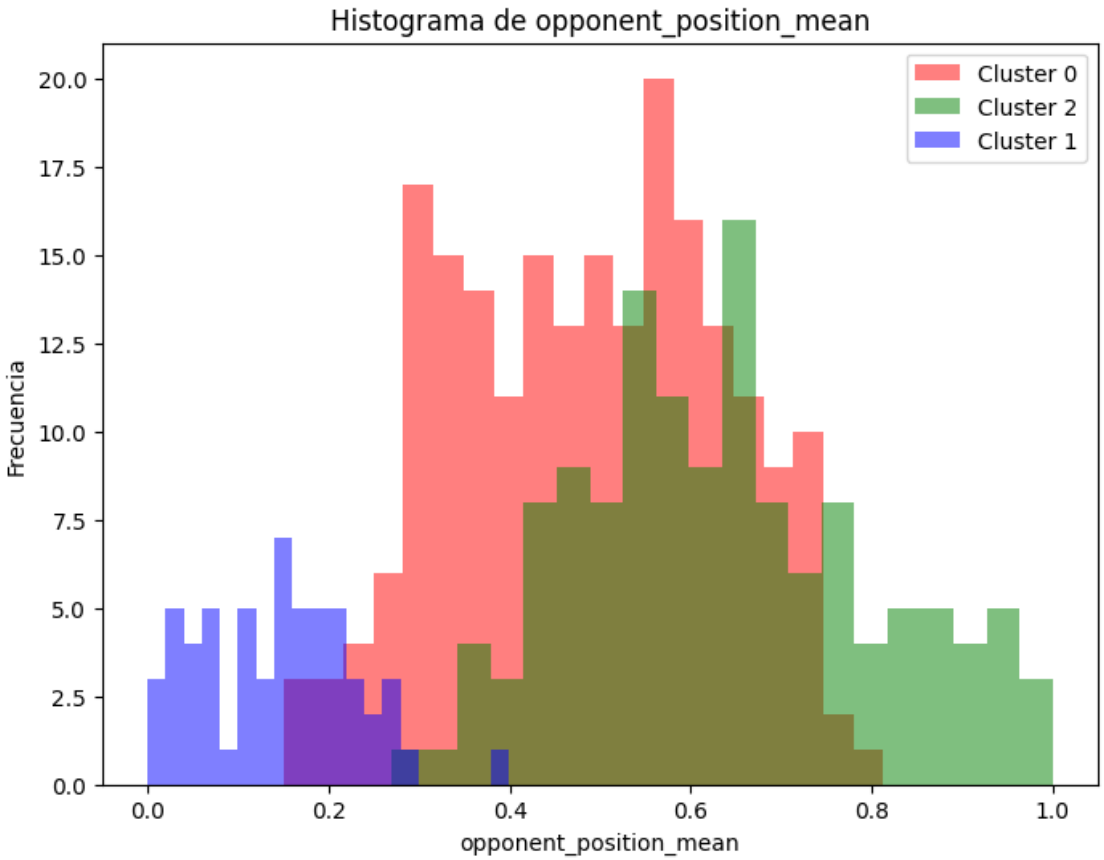

Histograma de opponent_goals_sum
Cluster 0
Cluster 2
Cluster 1
Frecuencia
0
5
10
15
20
0.0
0.2
0.4
0.6
0.8
1.0
opponent_goals_sum

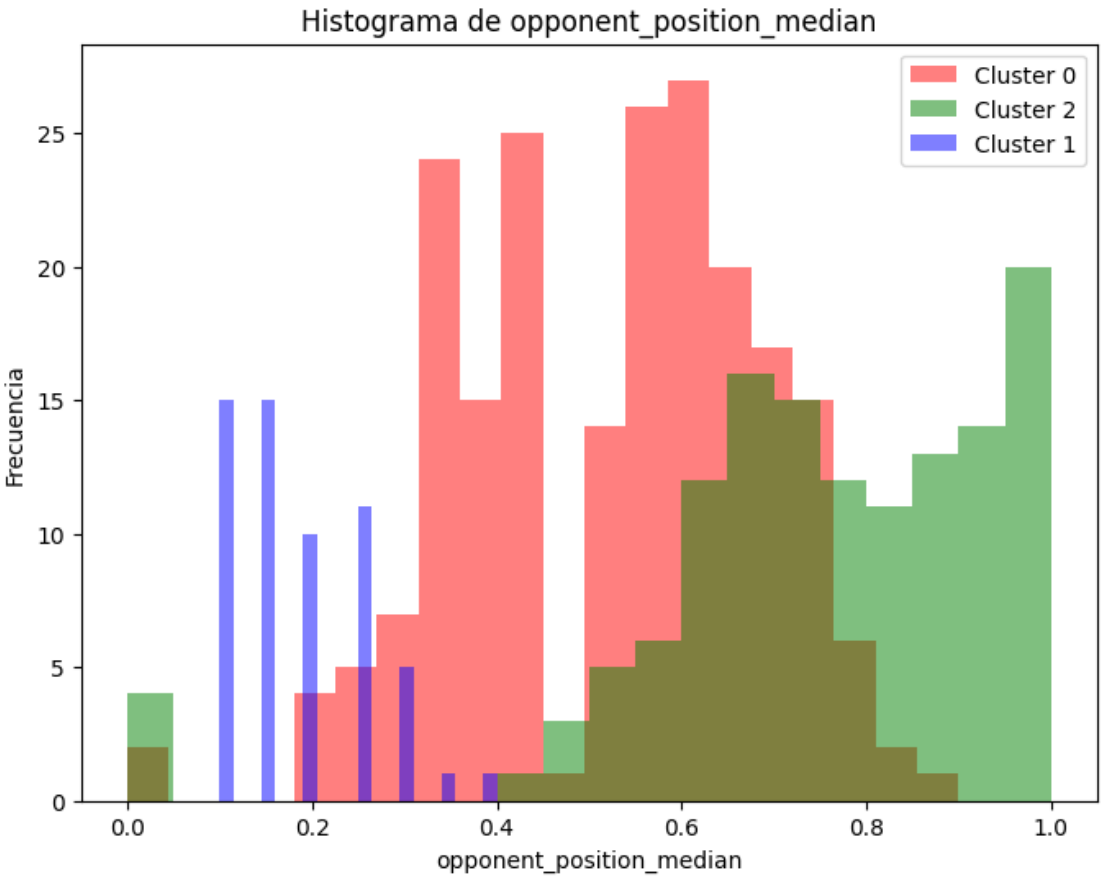

Histograma de opponent_goals_max
Cluster 0
Cluster 2
Cluster 1
Frecuencia
opponent_goals_max
0
10
20
30
40
50
60
0.0
0.2
0.4
0.6
0.8
1.0

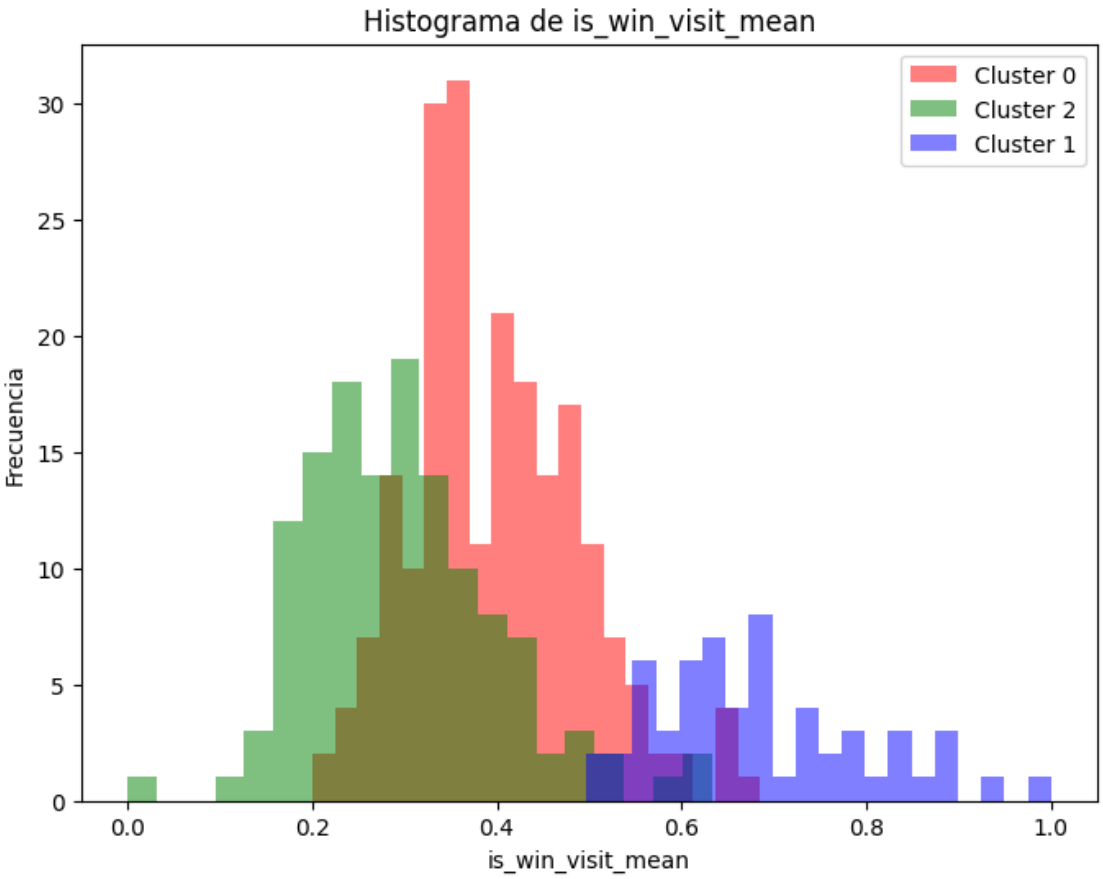

Histograma de opponent_goals_mean
Cluster 0
Cluster 2
Cluster 1
Frecuencia
opponent_goals_mean
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0

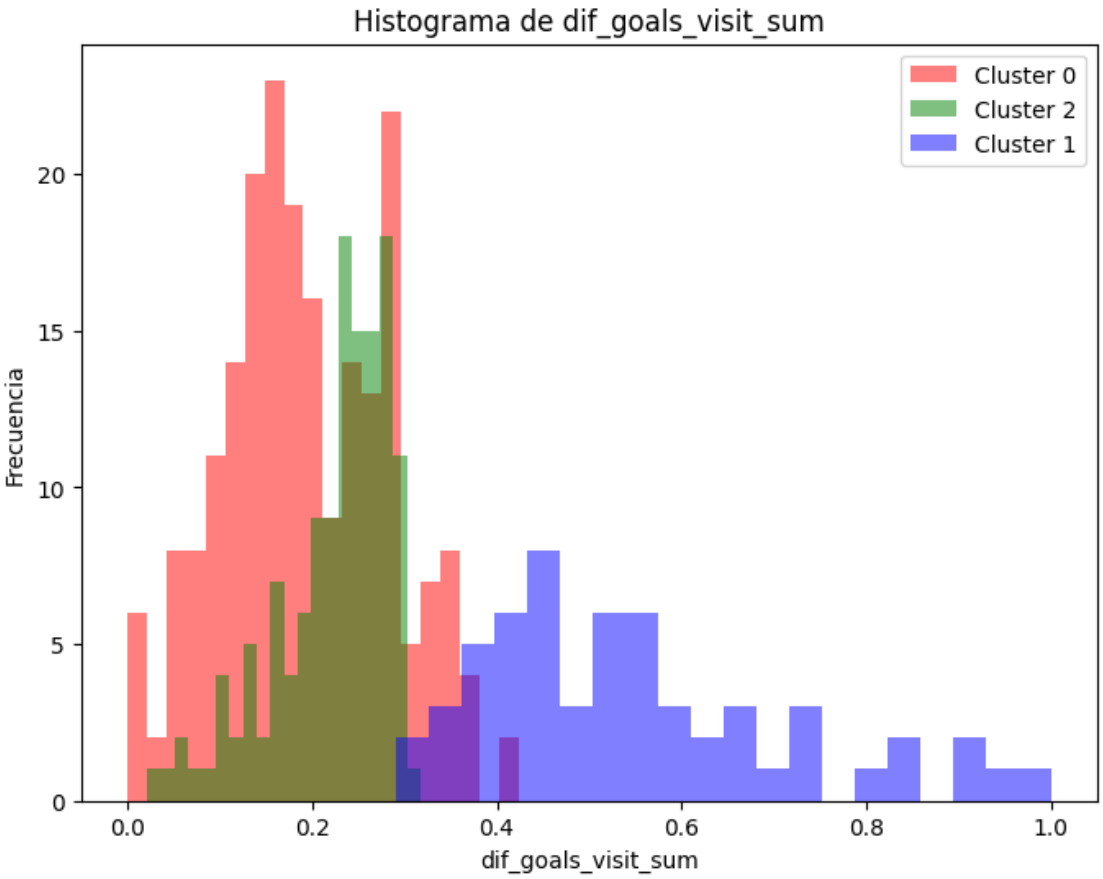

Histograma de opponent_goals_median
Cluster 0
Cluster 2
Cluster 1
Frecuencia
0
25
50
75
100
125
150
175
200
0.0
0.2
0.4
0.6
0.8
1.0
opponent_goals_median

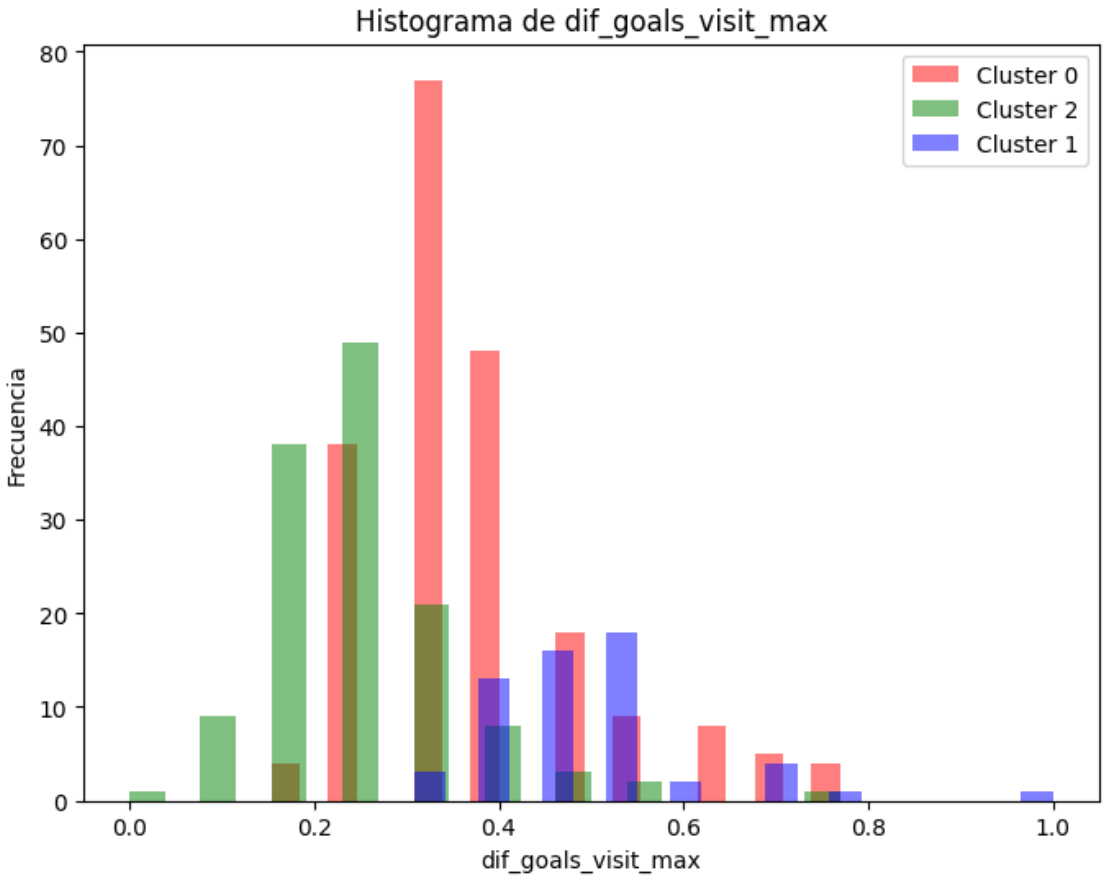

Histograma de opponent_position_max
Cluster 0
Cluster 2
Cluster 1
Frecuencia
0
10
20
30
40
50
60
0.0
0.2
0.4
0.6
0.8
1.0
opponent_position_max

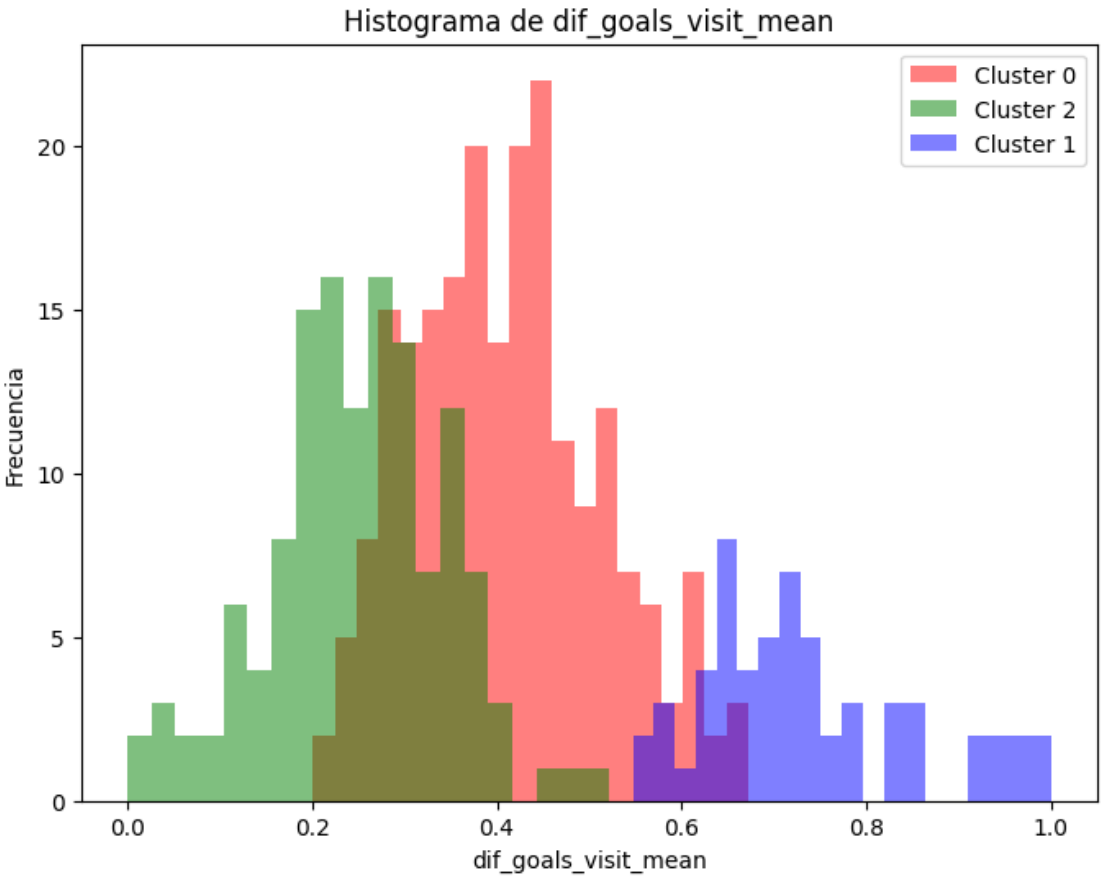

Histograma de opponent_position_mean
Cluster 0
Cluster 2
Cluster 1
Frecuencia
opponent_position_mean
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
0.0
0.2
0.4
0.6
0.8
1.0

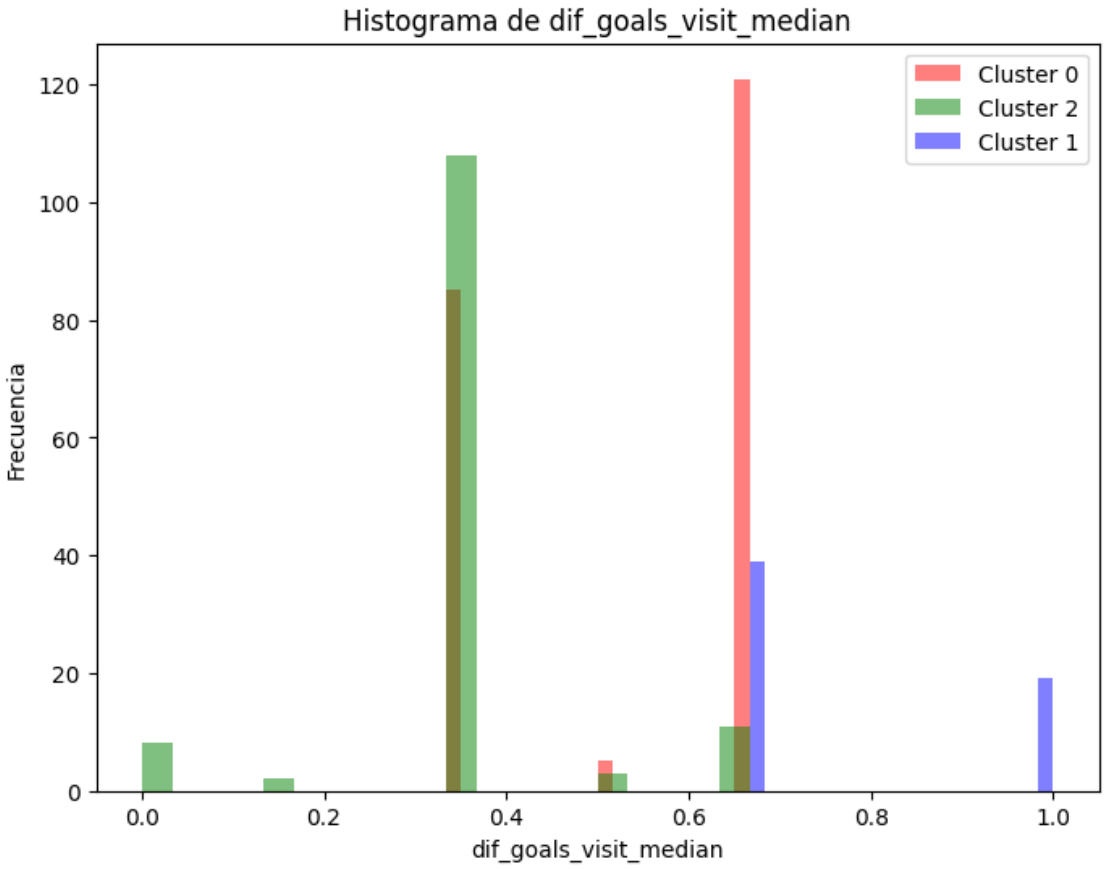

Histograma de opponent_position_median
Cluster 0
Cluster 2
Cluster 1
Frecuencia
0
5
10
15
20
25
0.0
0.2
0.4
0.6
0.8
1.0
opponent_position_median

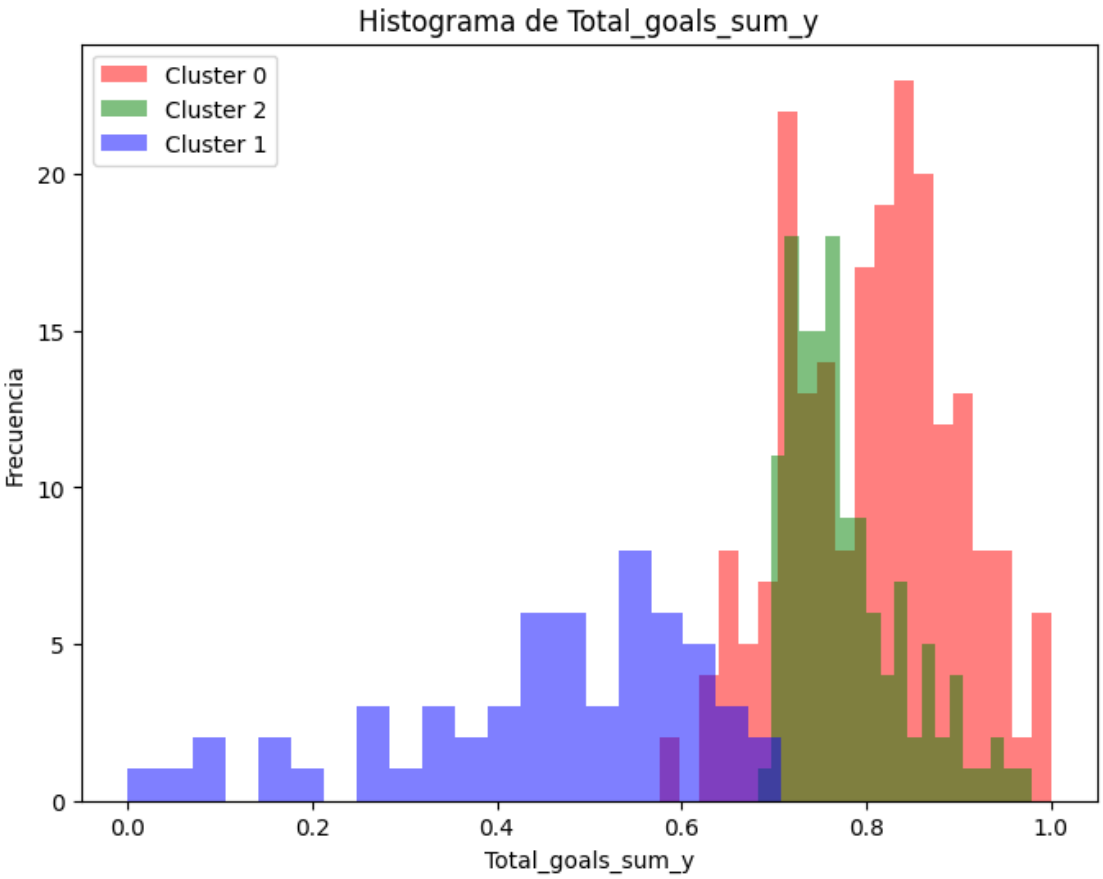

Histograma de is_win_visit_mean
Cluster 0
Cluster 2
Cluster 1
Frecuencia
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0
is_win_visit_mean

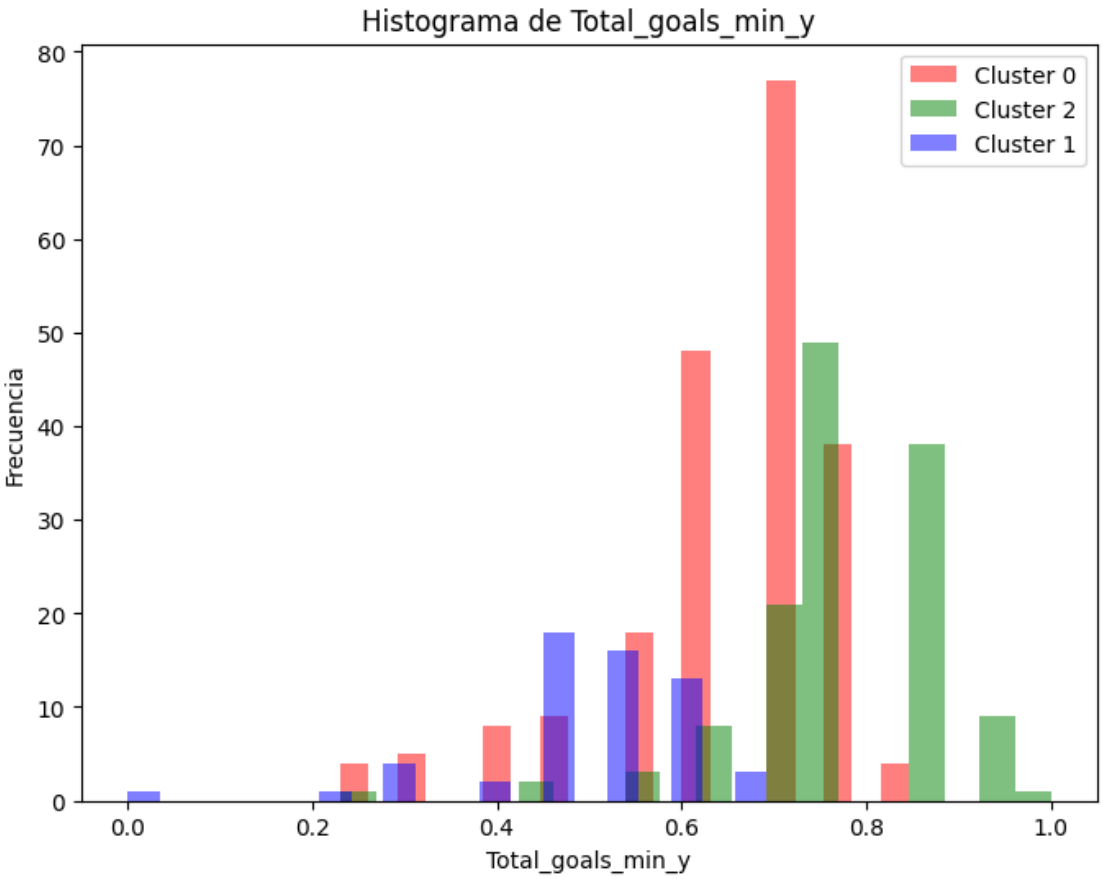

Histograma de dif_goals_visit_sum
Cluster 0
Cluster 2
Cluster 1
Frecuencia
0
5
10
15
20
0.0
0.2
0.4
0.6
0.8
1.0
dif_goals_visit_sum

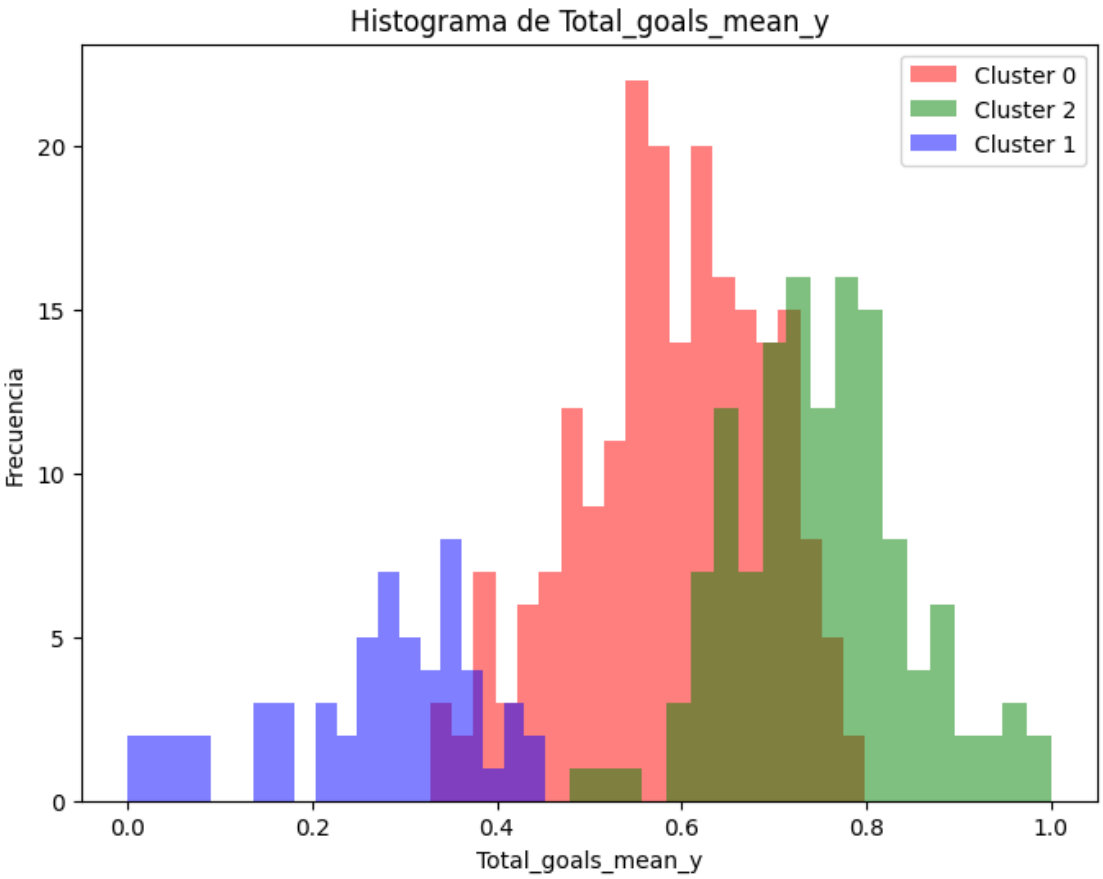

Histograma de dif_goals_visit_max
Cluster 0
Cluster 2
Cluster 1
Frecuencia
dif_goals_visit_max
0
10
20
30
40
50
60
70
80
0.0
0.2
0.4
0.6
0.8
1.0

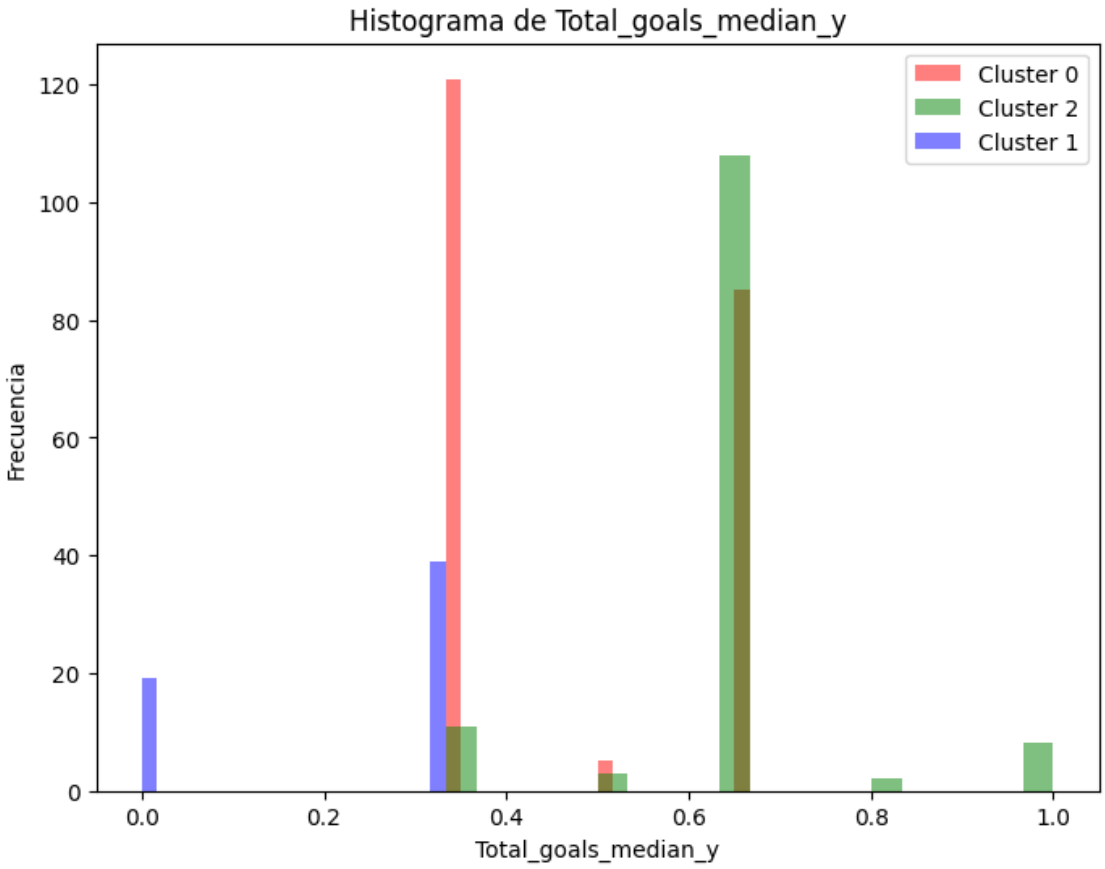

Histograma de dif_goals_visit_mean
Cluster 0
Cluster 2
Cluster 1
Frecuencia
0
5
10
15
20
0.0
0.2
0.4
0.6
0.8
1.0
dif_goals_visit_mean

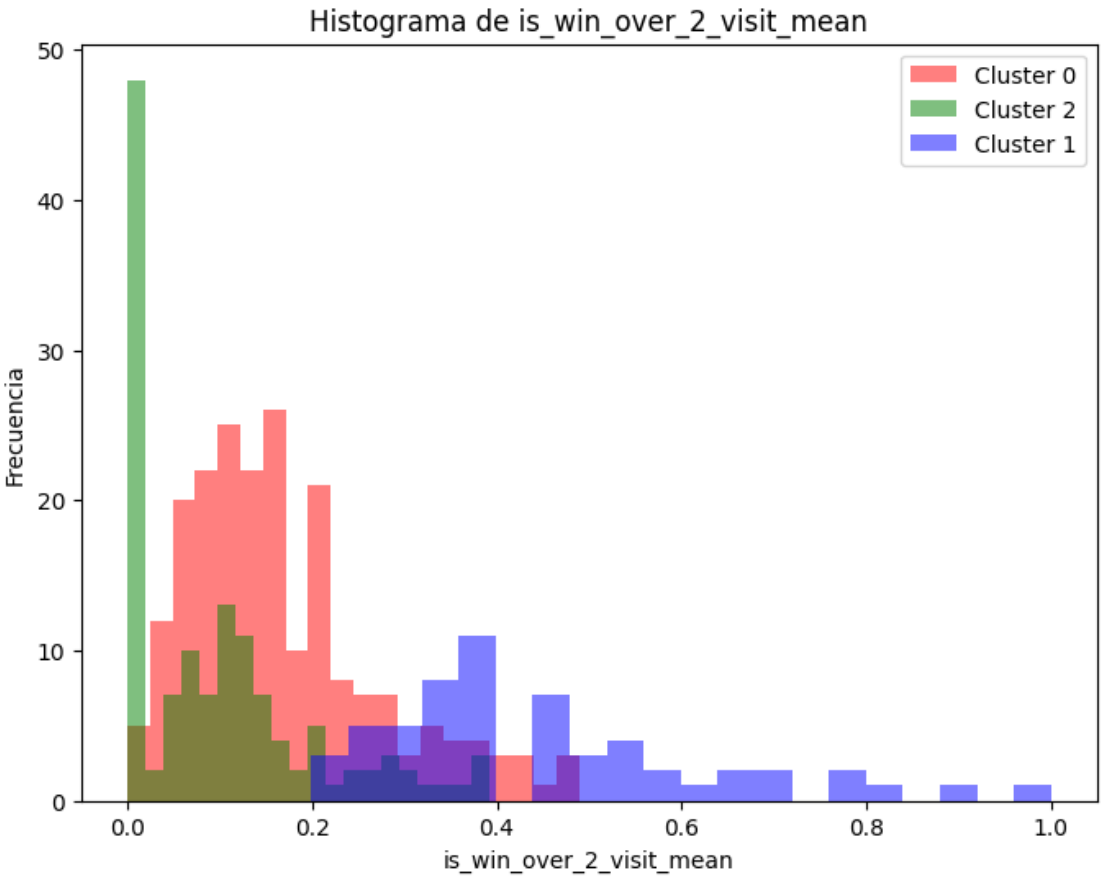

Histograma de dif_goals_visit_median
Cluster 0
Cluster 2
Cluster 1
Frecuencia
dif_goals_visit_median
0
20
40
60
80
100
120
0.0
0.2
0.4
0.6
0.8
1.0

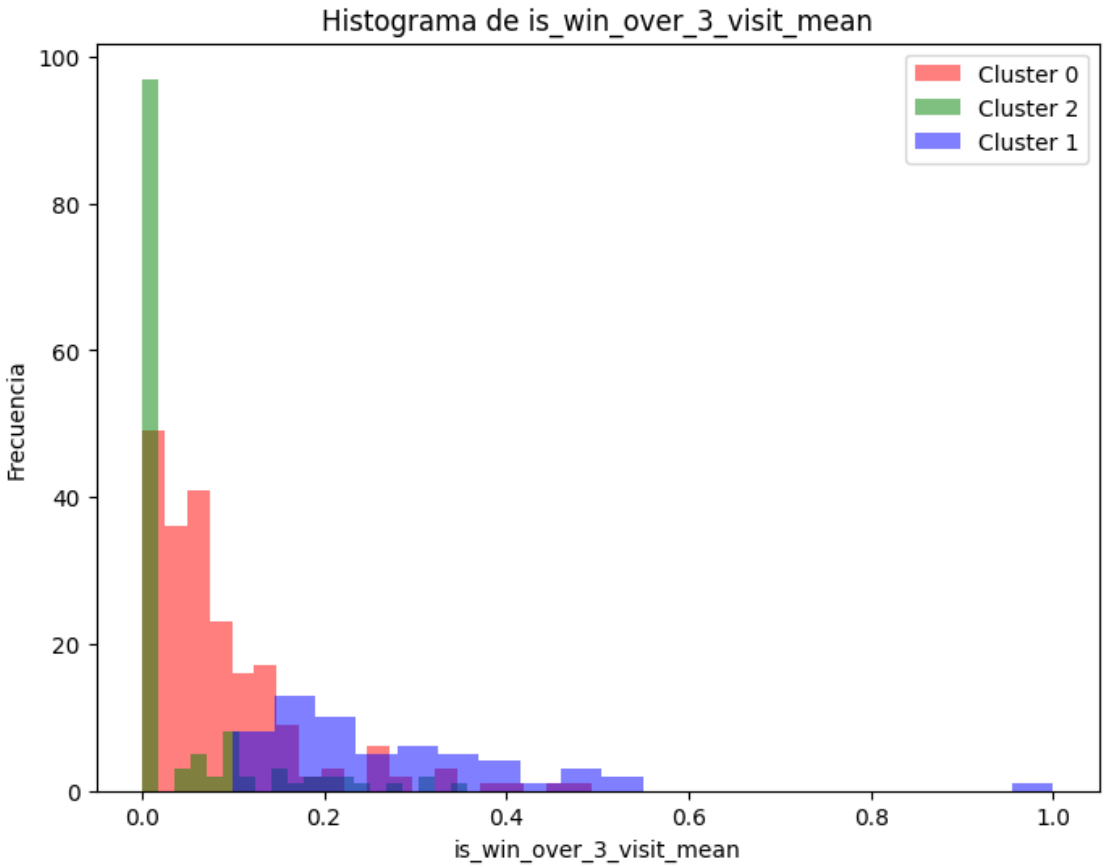

Histograma de Total_goals_sum_y
Cluster 0
Cluster 2
Cluster 1
Frecuencia
0
5
10
15
20
0.0
0.2
0.4
0.6
0.8
1.0
Total_goals_sum_y

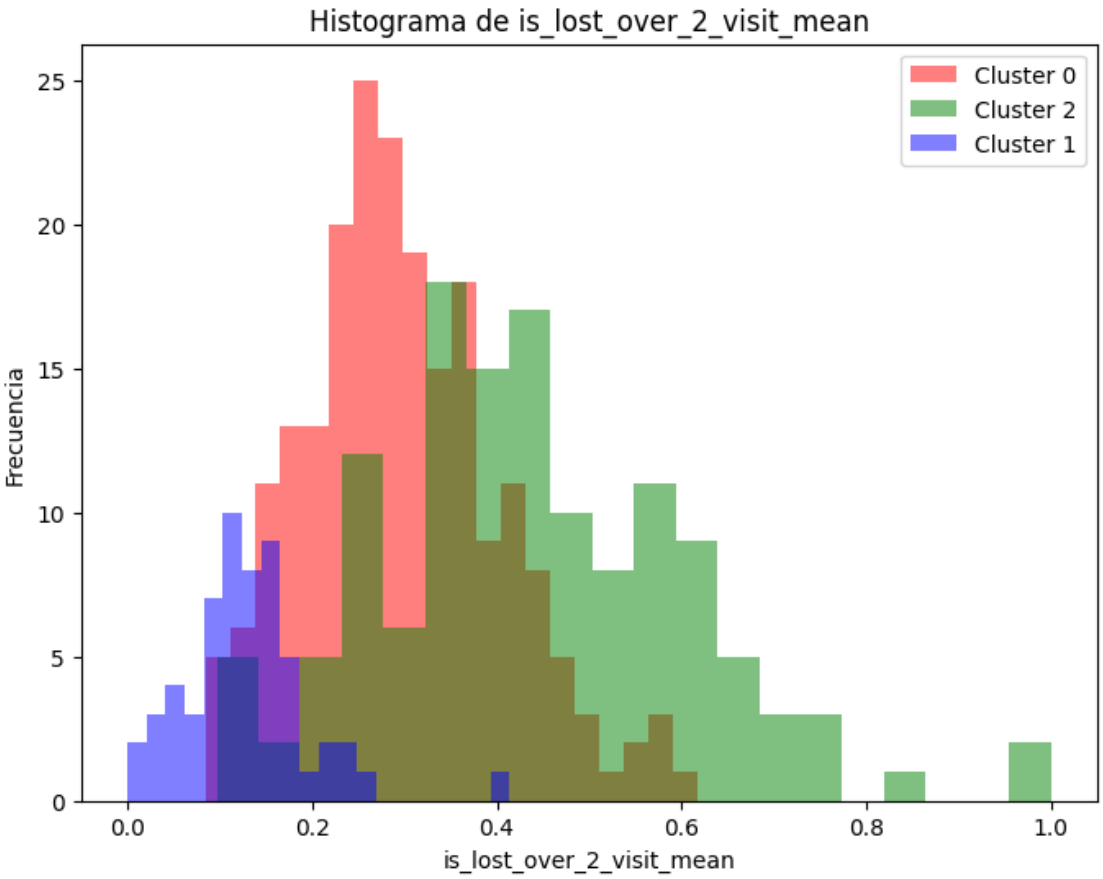

Histograma de Total_goals_min_y
Cluster 0
Cluster 2
Cluster 1
Frecuencia
Total_goals_min_y
0
10
20
30
40
50
60
70
80
0.0
0.2
0.4
0.6
0.8
1.0

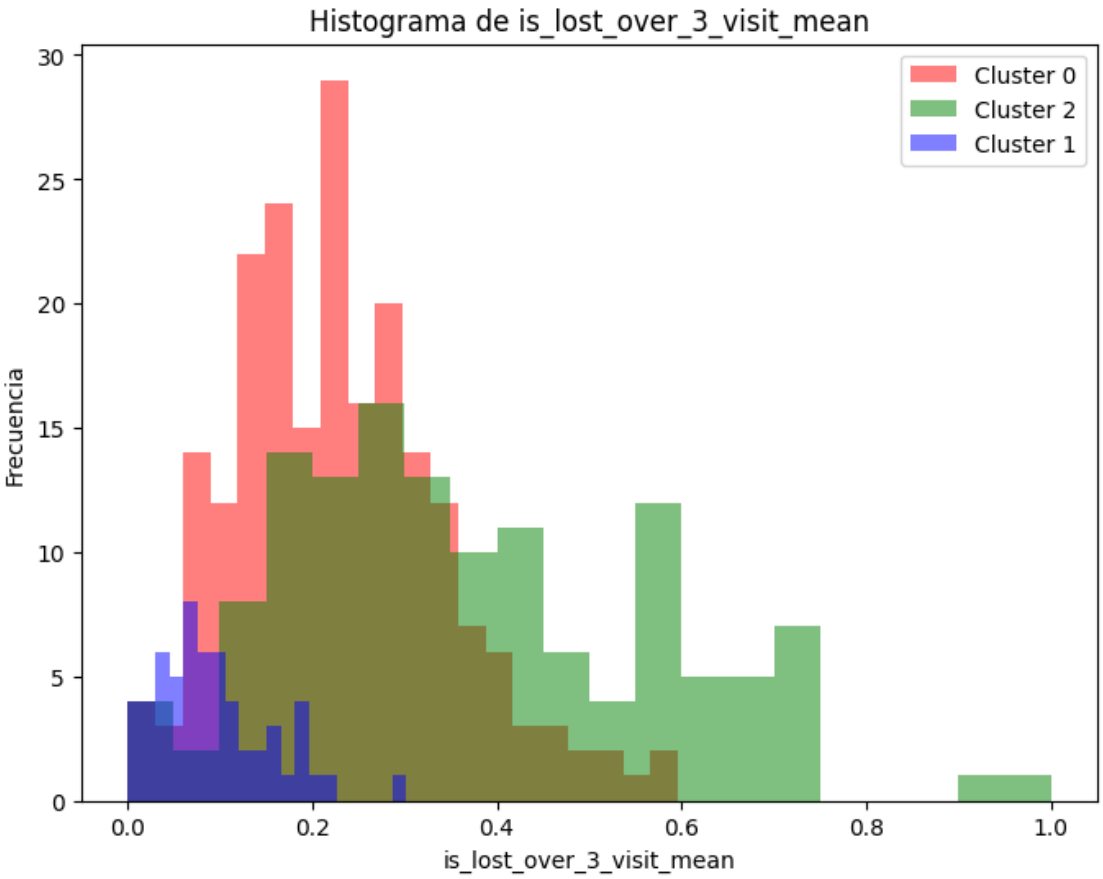

Histograma de Total_goals_mean_y
Cluster 0
Cluster 2
Cluster 1
Frecuencia
0
5
10
15
20
0.0
0.2
0.4
0.6
0.8
1.0
Total_goals_mean_y

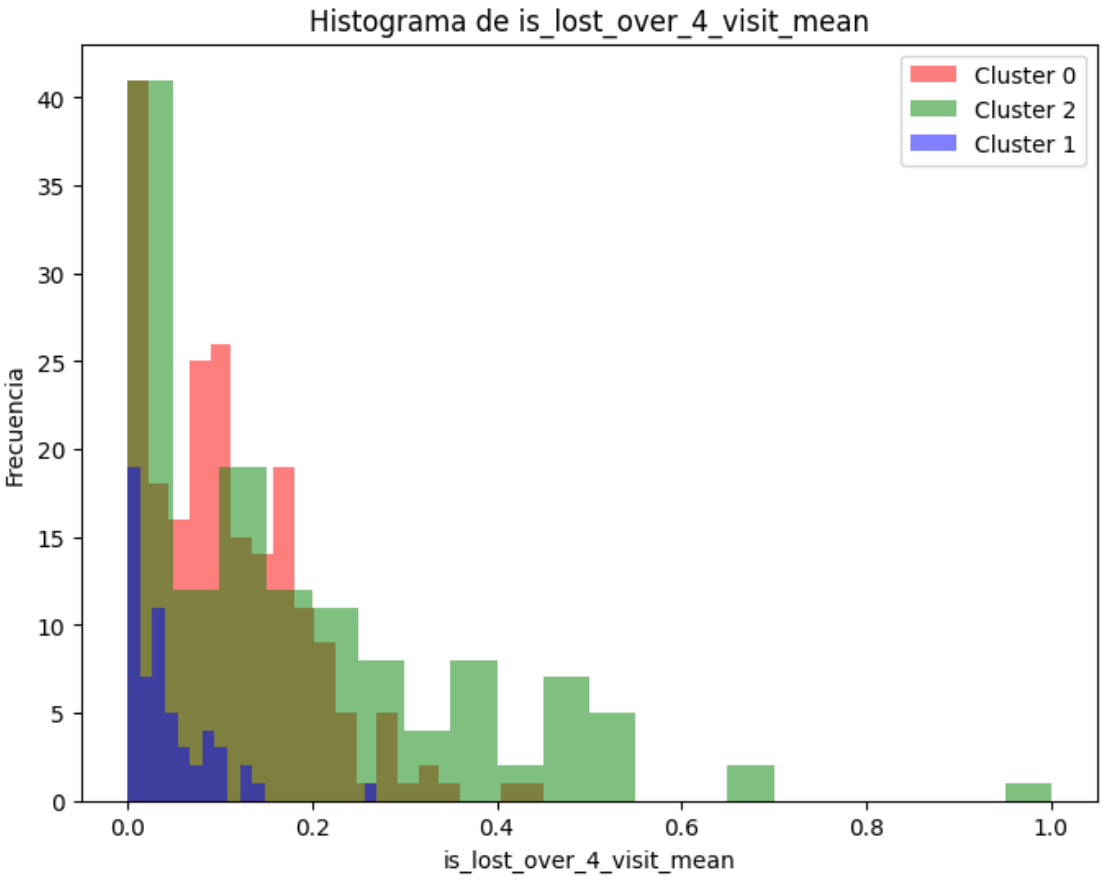

Histograma de Total_goals_median_y
Cluster 0
Cluster 2
Cluster 1
Frecuencia
Total_goals_median_y
0
20
40
60
80
100
120
0.0
0.2
0.4
0.6
0.8
1.0

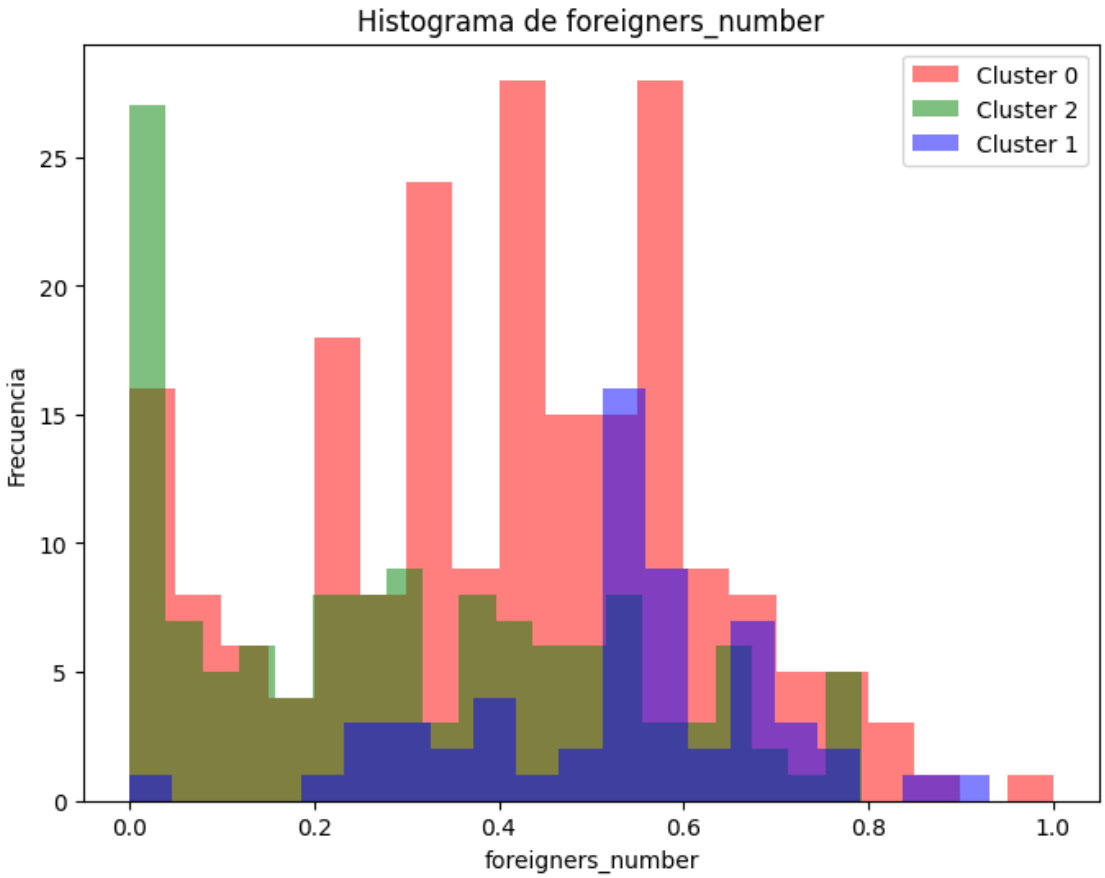

Histograma de is_win_over_2_visit_mean
Cluster 0
Cluster 2
Cluster 1
Frecuencia
is_win_over_2_visit_mean
0
10
20
30
40
50
0.0
0.2
0.4
0.6
0.8
1.0

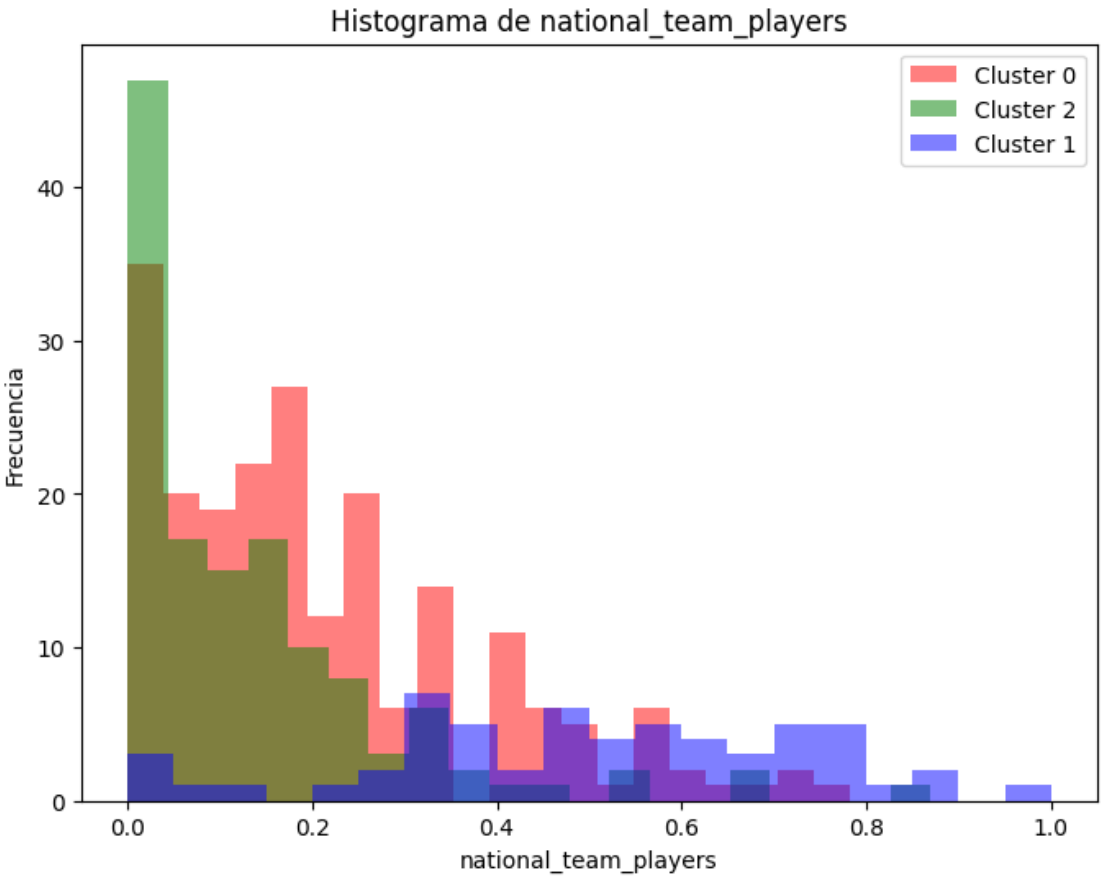

Histograma de is_win_over_3_visit_mean
Cluster 0
Cluster 2
Cluster 1
Frecuencia
100
80
60
40
20
0
0.0
0.2
0.4
0.6
0.8
1.0
is_win_over_3_visit_mean

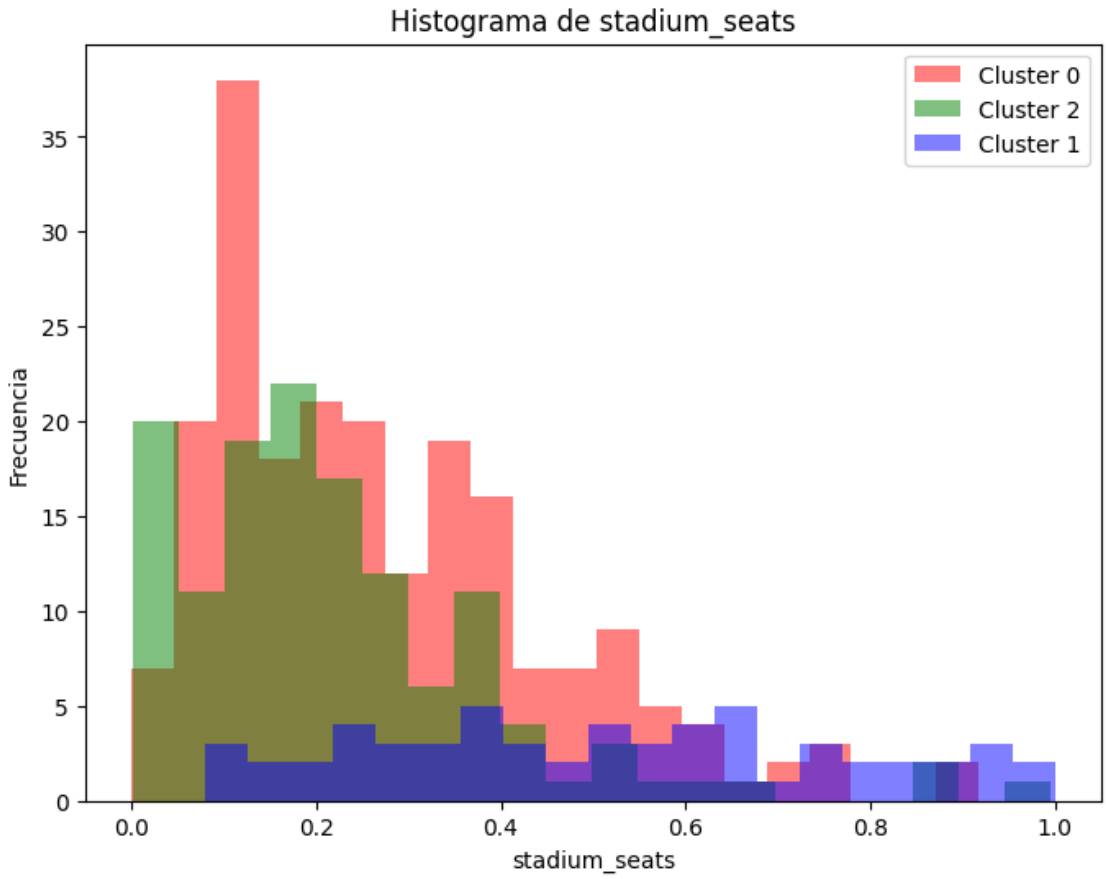

Histograma de is_lost_over_2_visit_mean
Cluster 0
Cluster 2
Cluster 1
Frecuencia
0
5
10
15
20
25
0.0
0.2
0.4
0.6
0.8
1.0
is_lost_over_2_visit_mean


Histograma de is_lost_over_3_visit_mean
Cluster 0
Cluster 2
Cluster 1
Frecuencia
is_lost_over_3_visit_mean
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0


Histograma de is_lost_over_4_visit_mean
Cluster 0
Cluster 2
Cluster 1
Frecuencia
0
5
10
15
20
25
30
35
40
0.0
0.2
0.4
0.6
0.8
1.0
is_lost_over_4_visit_mean


Histograma de foreigners_number
Cluster 0
Cluster 2
Cluster 1
Frecuencia
0
5
10
15
20
25
0.0
0.2
0.4
0.6
0.8
1.0
foreigners_number

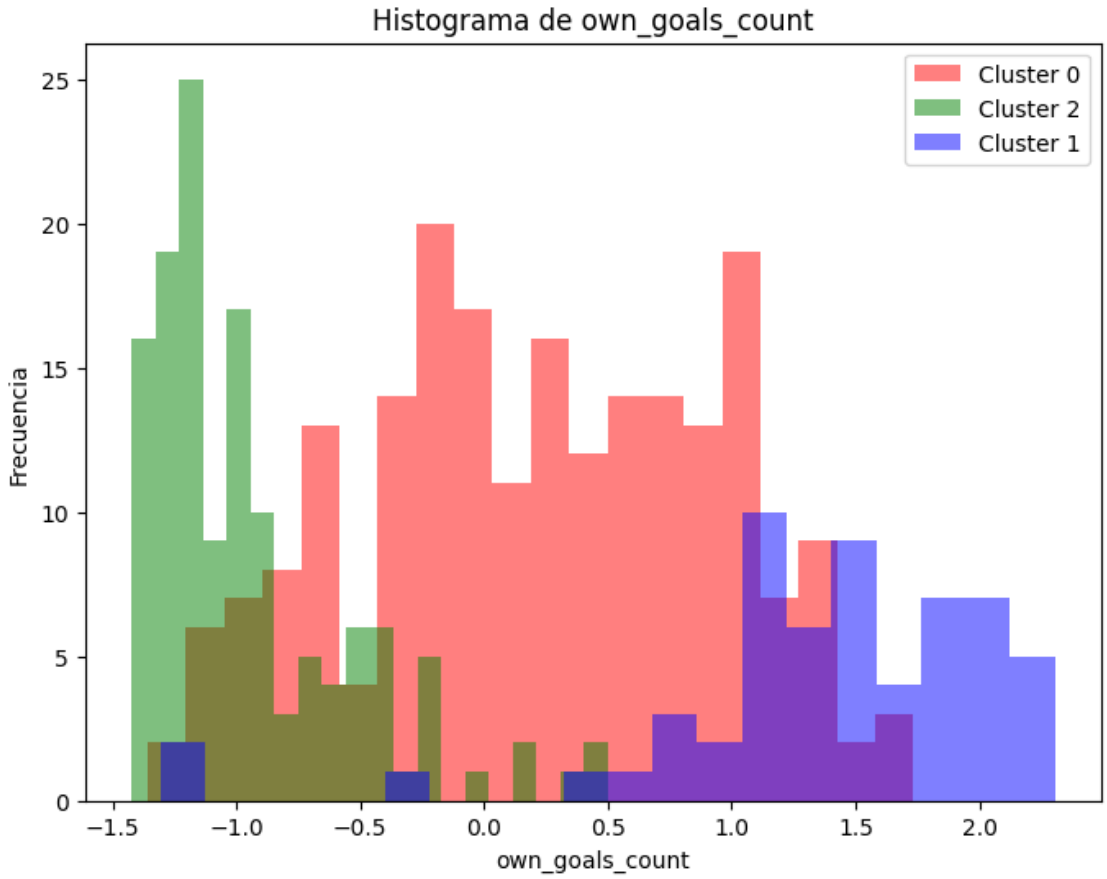

Histograma de national_team_players
Cluster 0
Cluster 2
Cluster 1
Frecuencia
0
10
20
30
40
0.0
0.2
0.4
0.6
0.8
1.0
national_team_players



[413]:
```
unique_clusters = Xmds_sample['cl_gmm'].unique()
Xsc['cl_gmm']=Xmds_sample['cl_gmm']

# Color mapping para kmeans_mds_3
color_mapping = {0: 'red', 1: 'blue', 2: 'green'}

# Itera a través de las variables en ls_best
for variable in ls_tukey_gaussianos:
    # Crear un nuevo histograma para la variable actual
    plt.figure(figsize=(8, 6))  # Establece el tamaño de la figura (opcional)

    # Itera a través de los valores únicos de kmeans_mds_3
    for cluster_value in unique_clusters:
        # Restablece el índice del DataFrame Xmm_sample antes de la selección
        subset_data = Xsc.reset_index(drop=True)[Xsc['cl_gmm'] ==␣
 ↪cluster_value][variable]

        # Crea el histograma utilizando solo un color para este conjunto de␣
 ↪datos
```

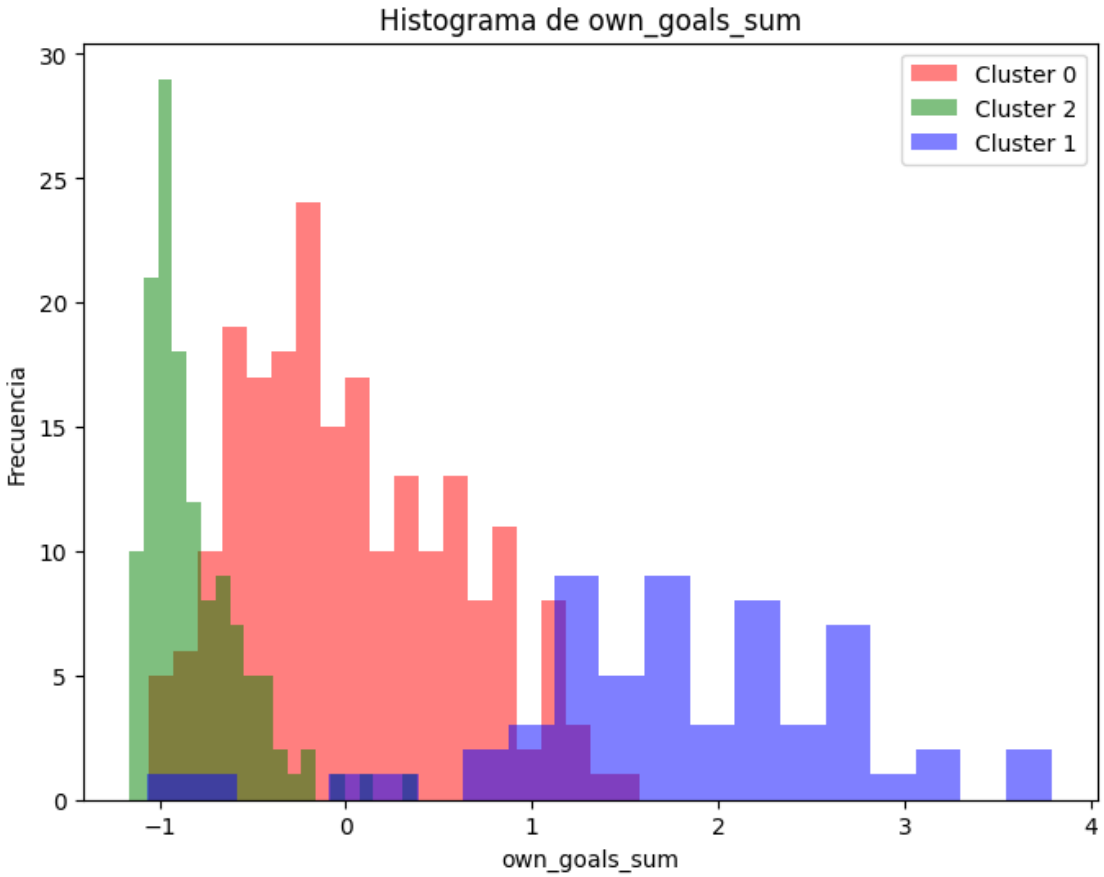
```
        plt.hist(subset_data, bins=20, color=color_mapping[cluster_value],␣
↪alpha=0.5, label=f'Cluster {cluster_value}')
        print('hola')
    # Configura el título y etiquetas de los ejes
    plt.title(f'Histograma de {variable}')
    plt.xlabel(variable)
    plt.ylabel('Frecuencia')

    # Agrega una leyenda para identificar los clusters
    plt.legend()

    # Muestra la gráfica
    plt.show()
```

```
hola
hola
hola
```

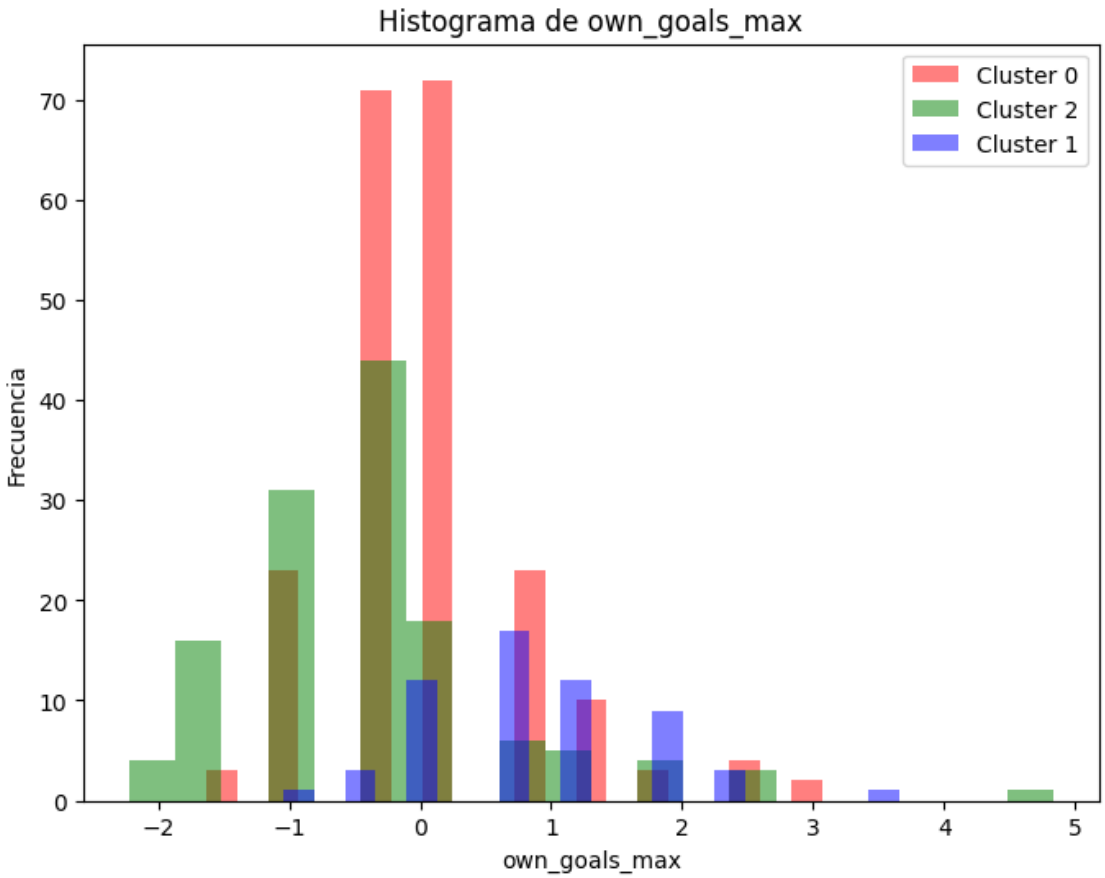



```
hola
hola
```

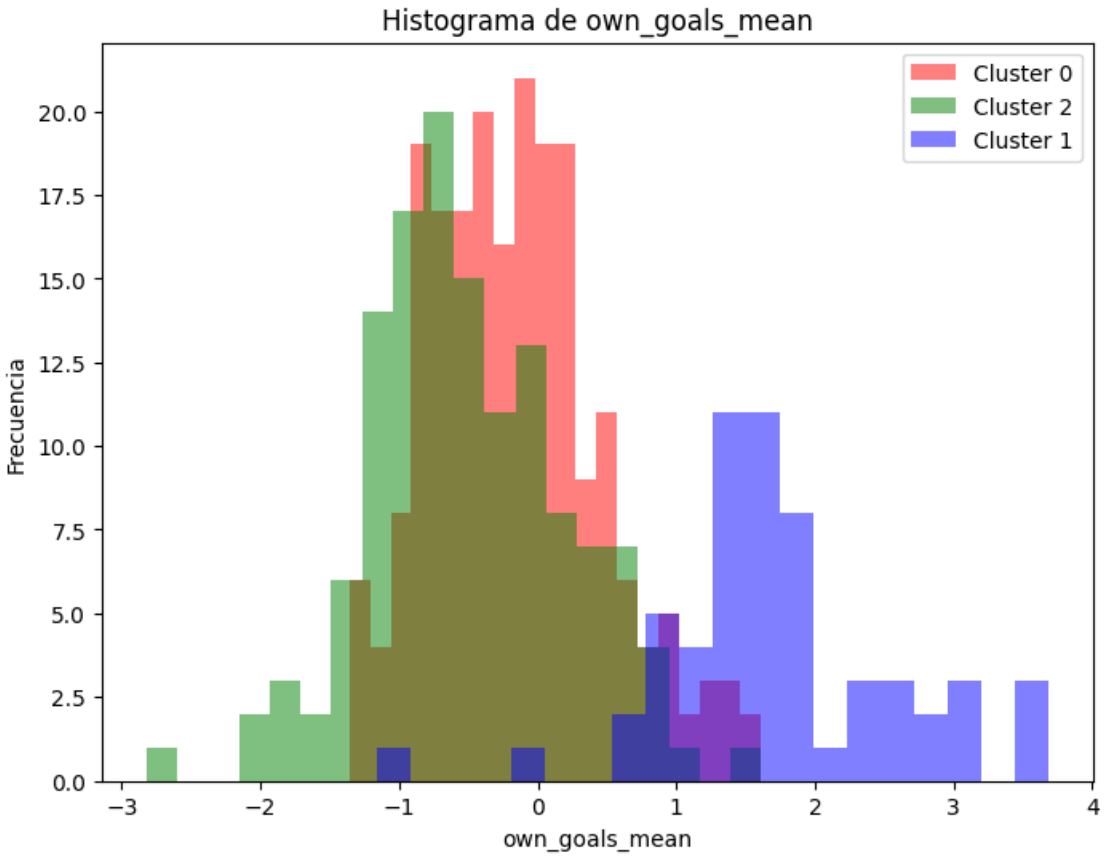
hola



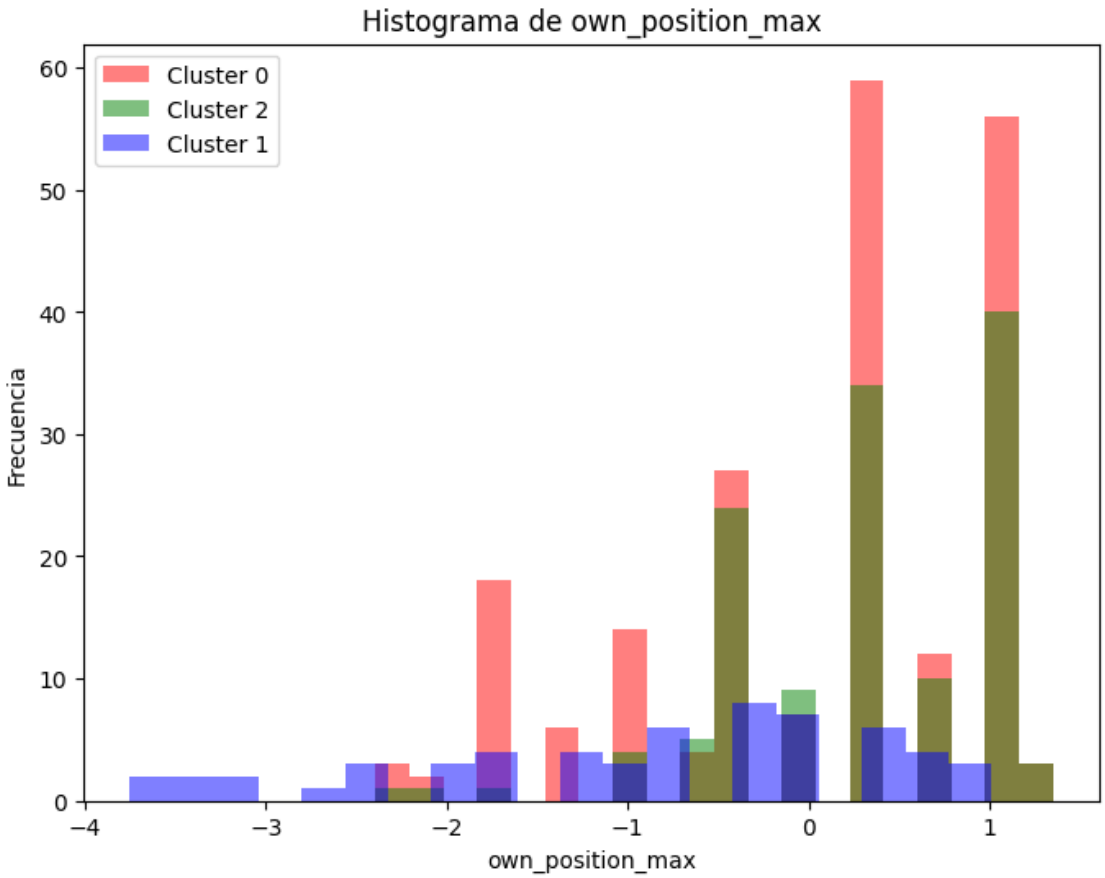
hola
hola
hola



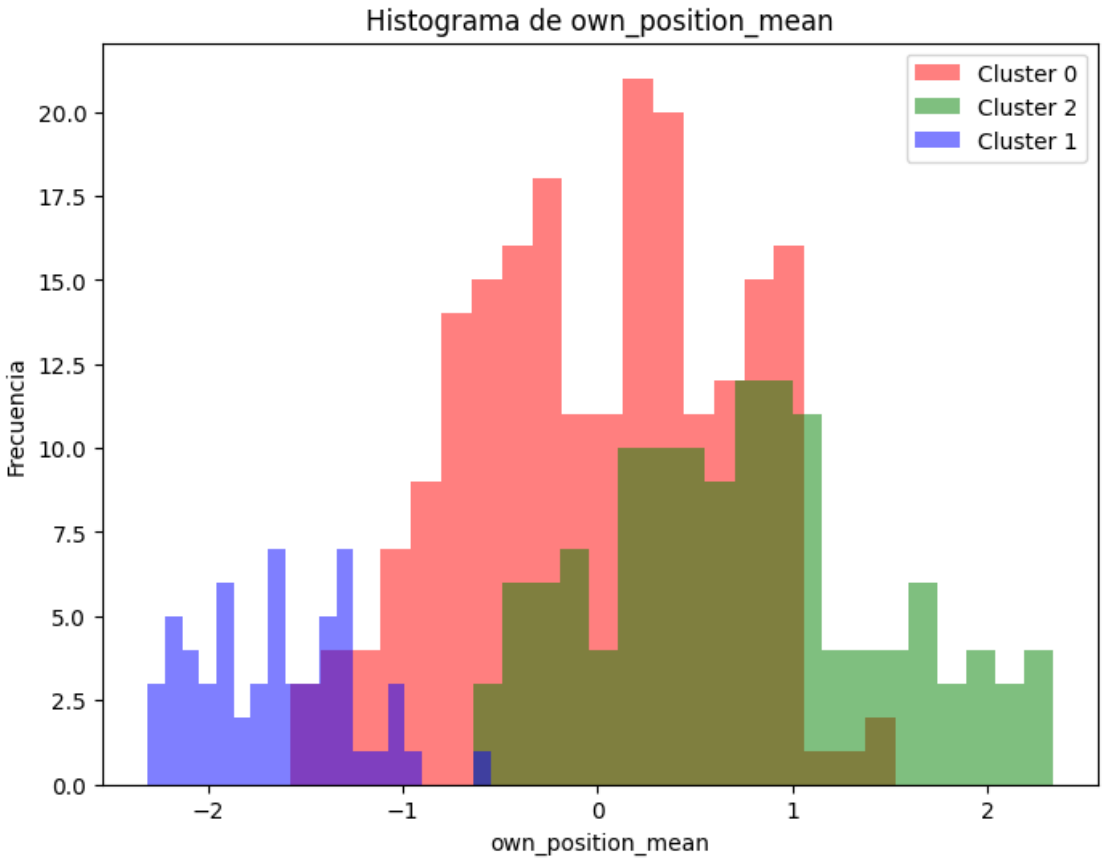

```
hola
hola
hola
```



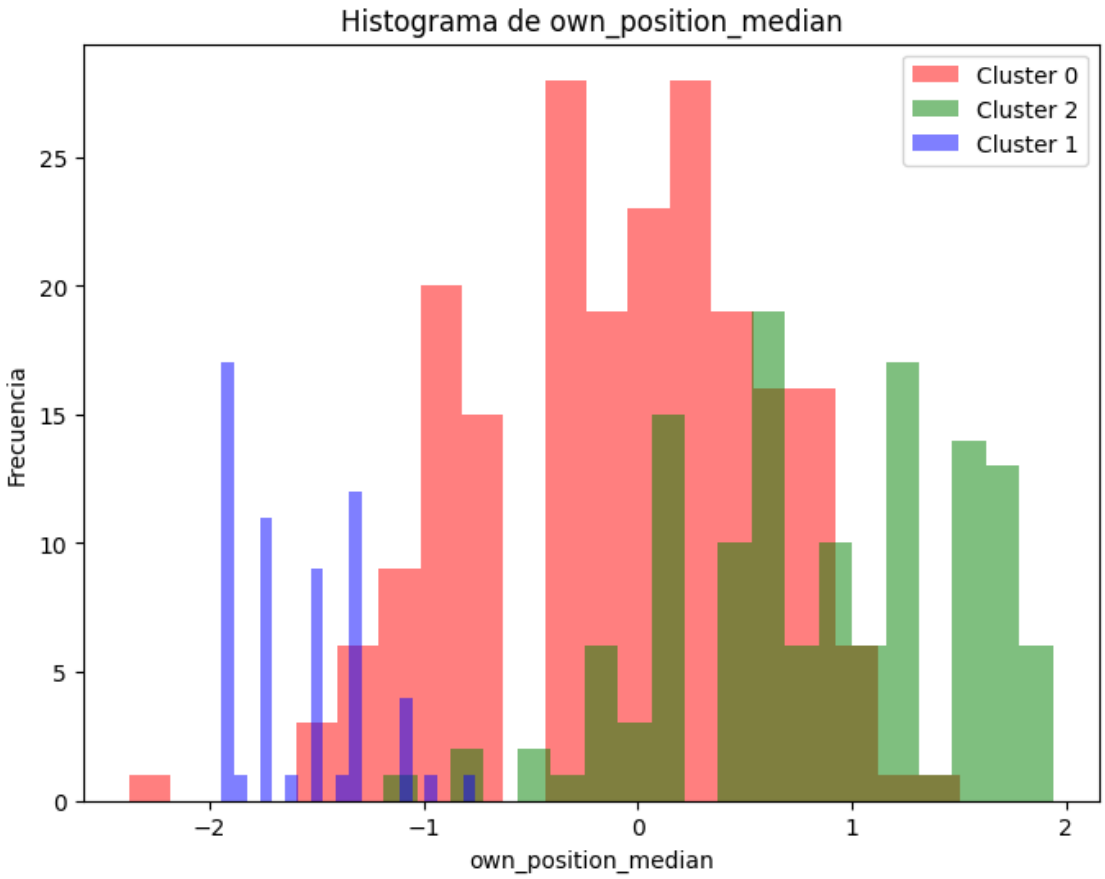

```
hola
hola
hola
```



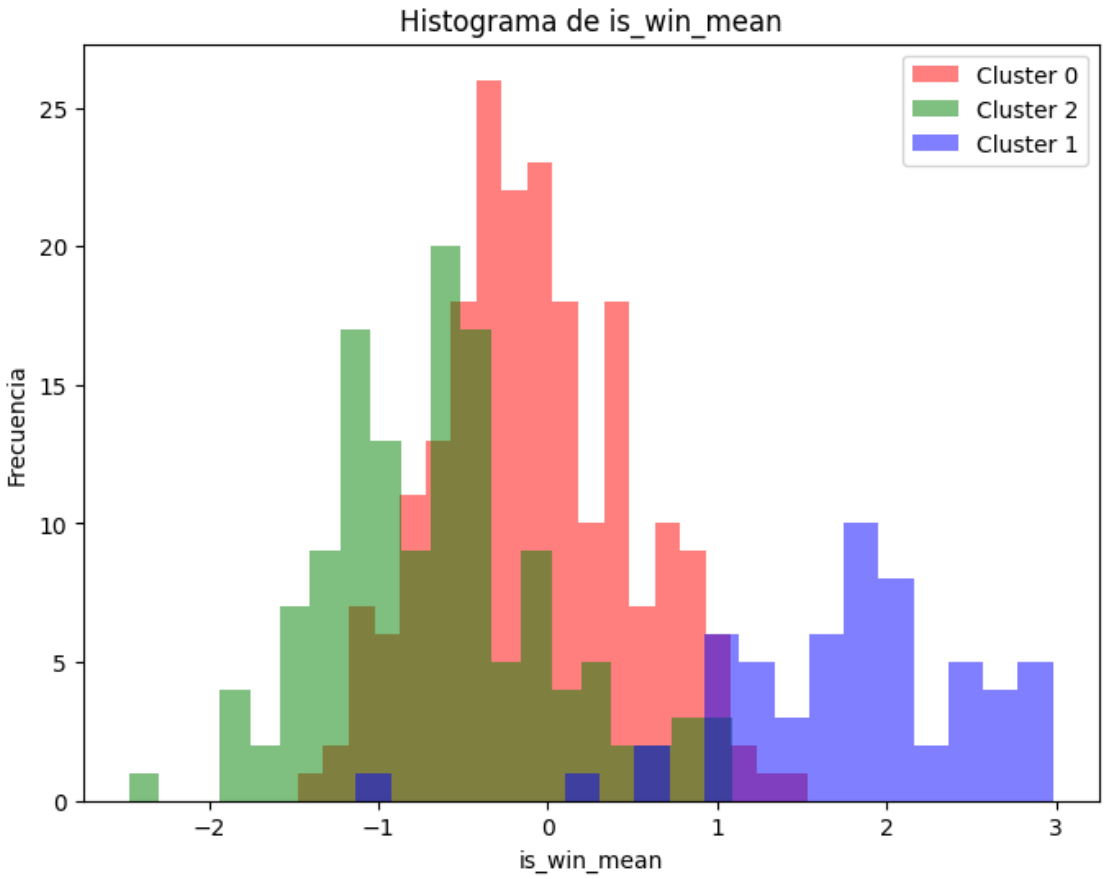

```
hola
hola
hola
```



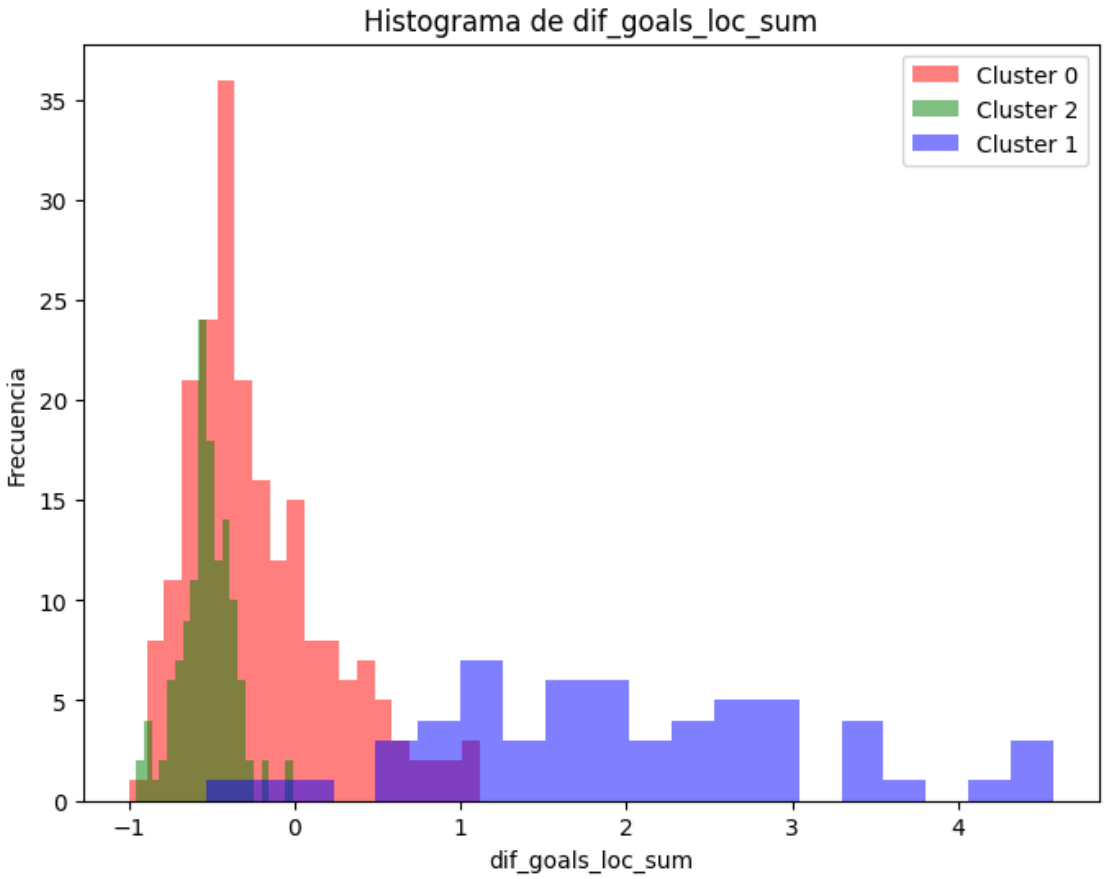

```
hola
hola
hola
```



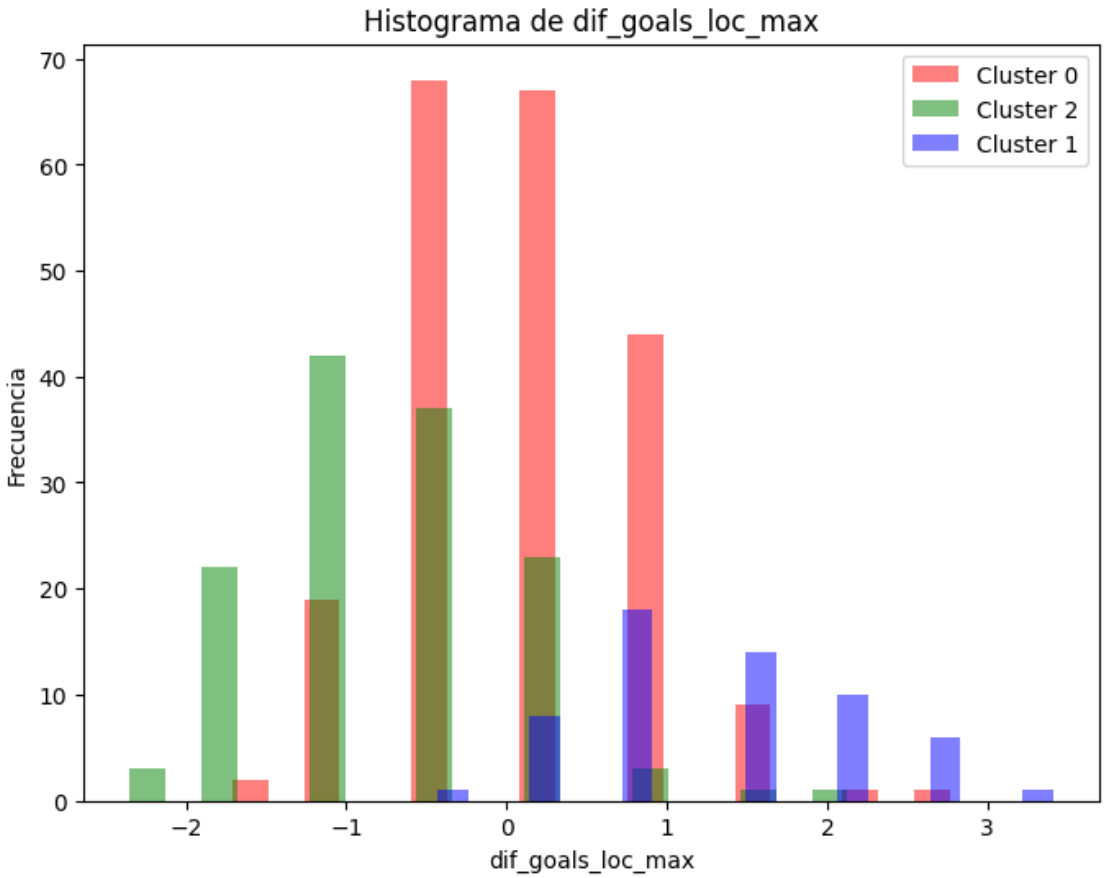

```
hola
hola
hola
```



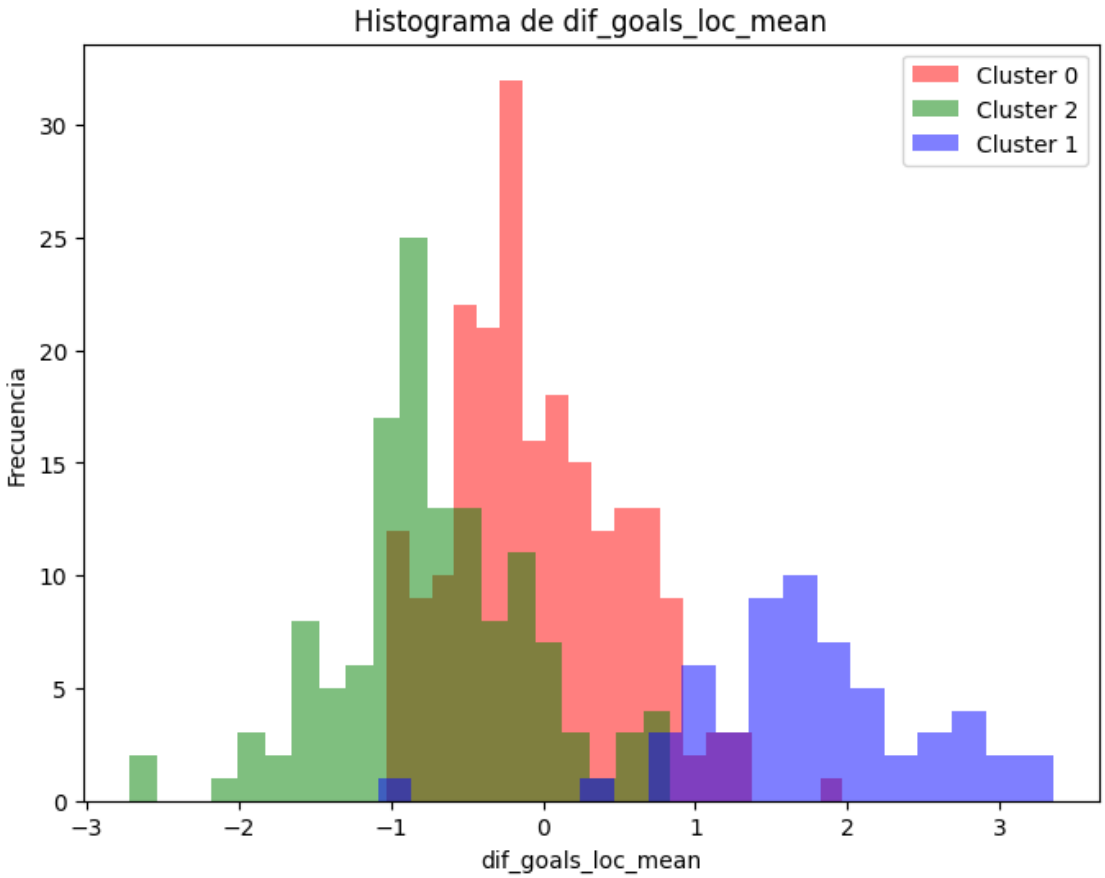

```
hola
hola
hola
```



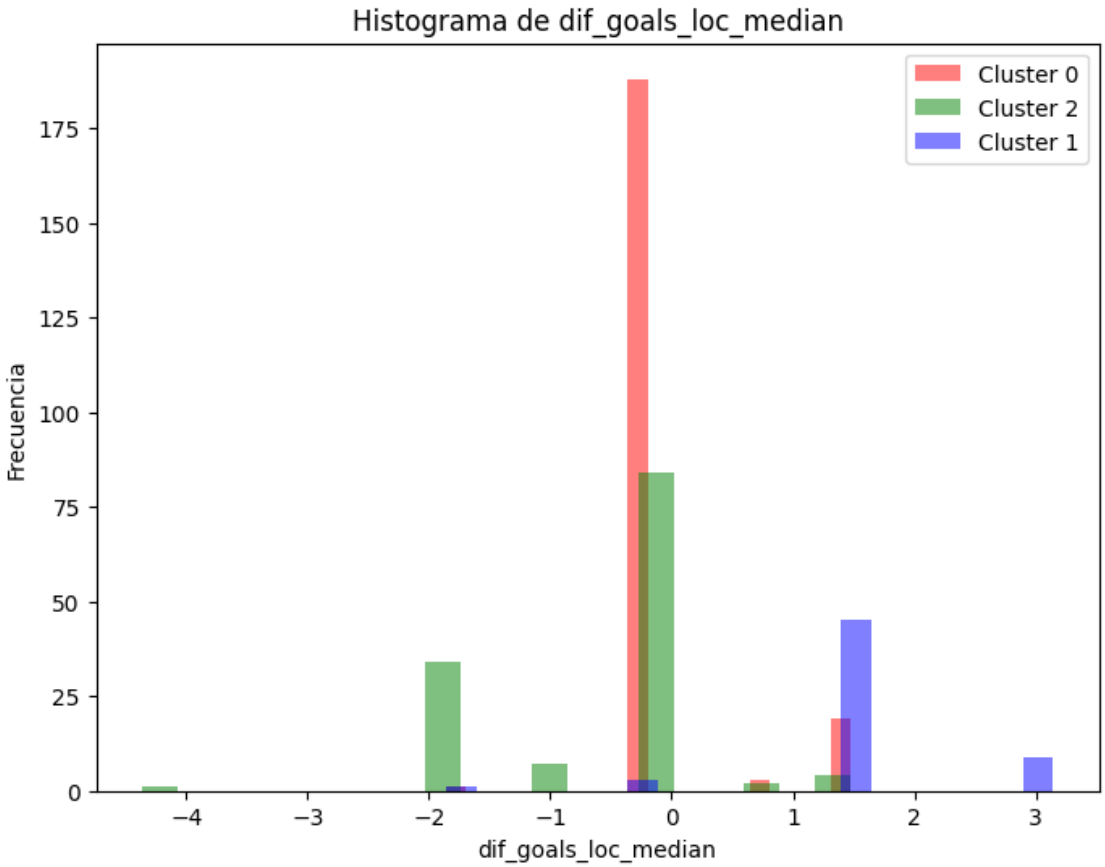

```
hola
hola
hola
```



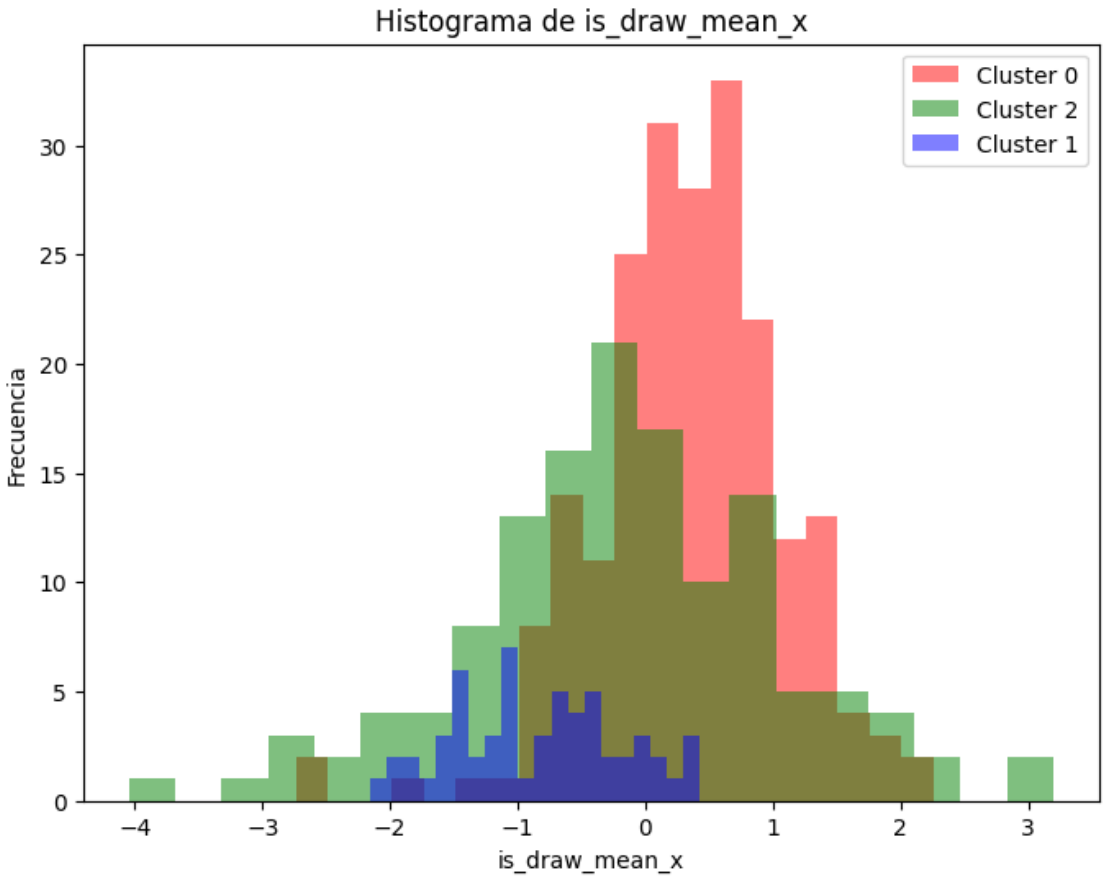

```
hola
hola
hola
```



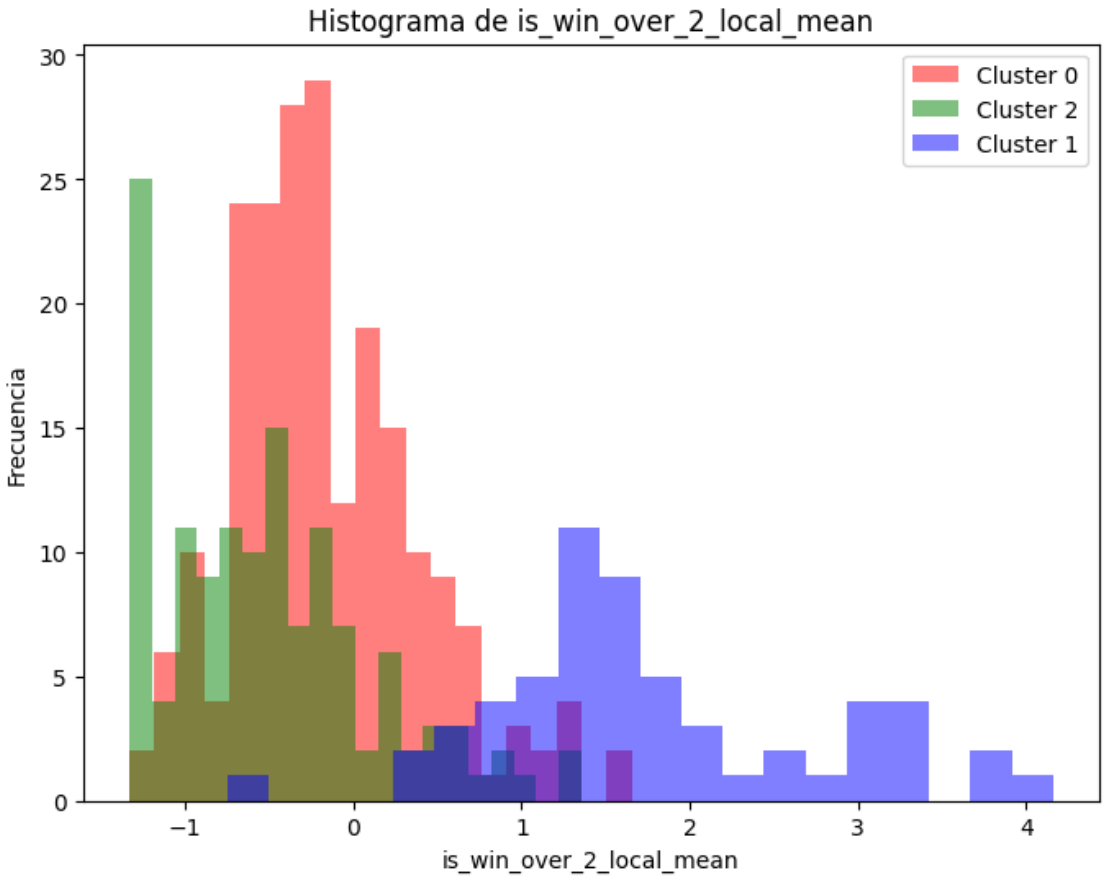

```
hola
hola
hola
```



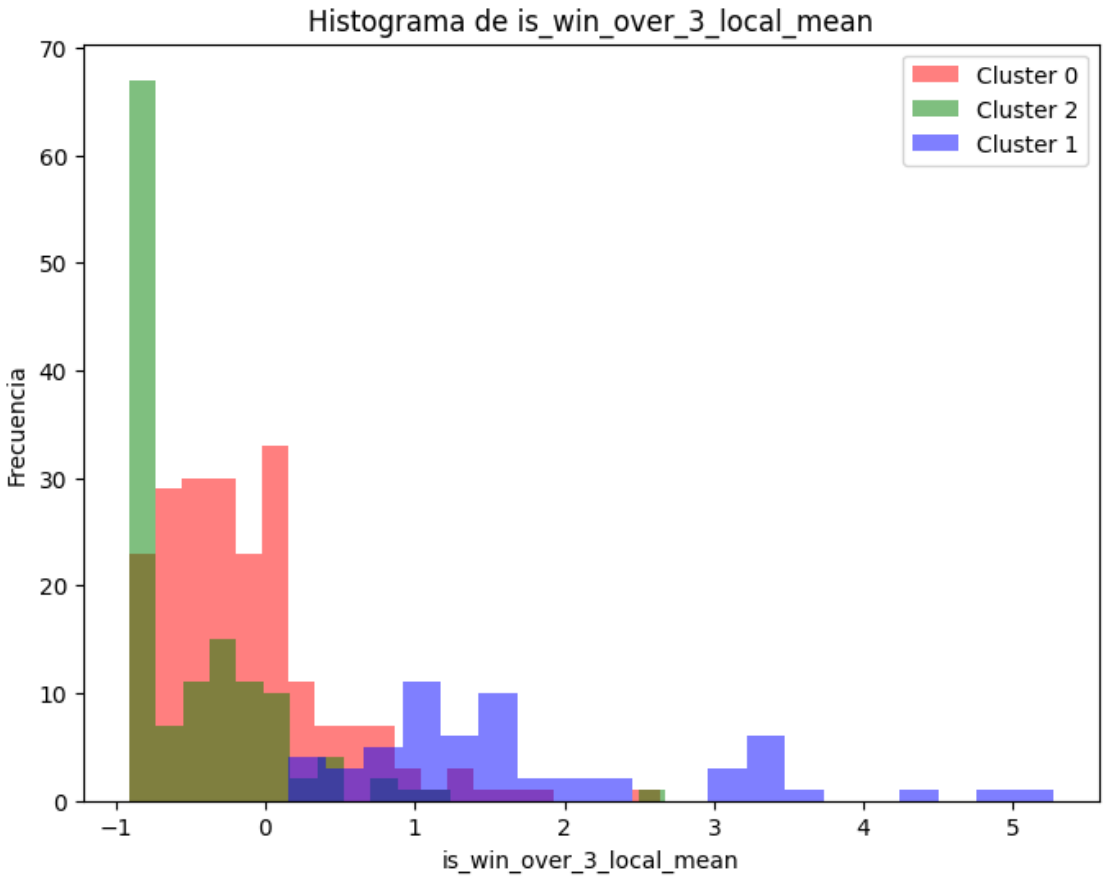

```
hola
hola
hola
```



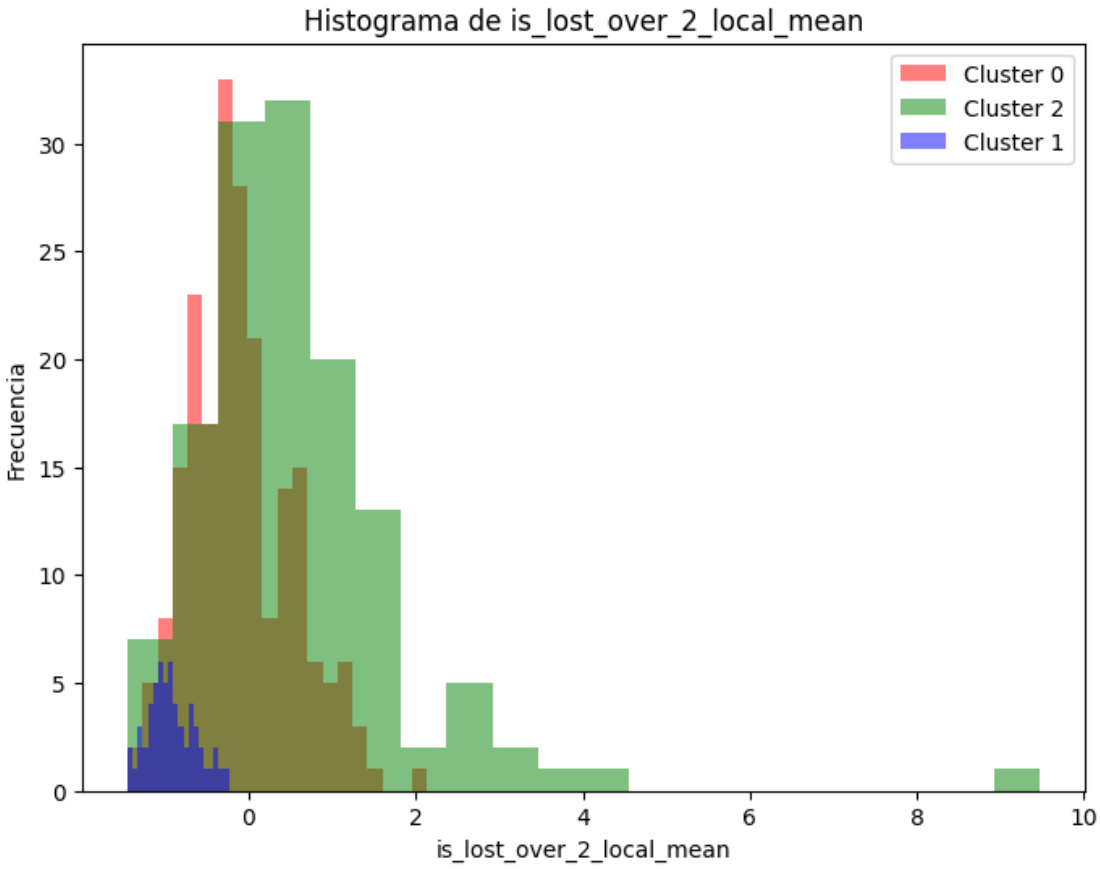

```
hola
hola
hola
```



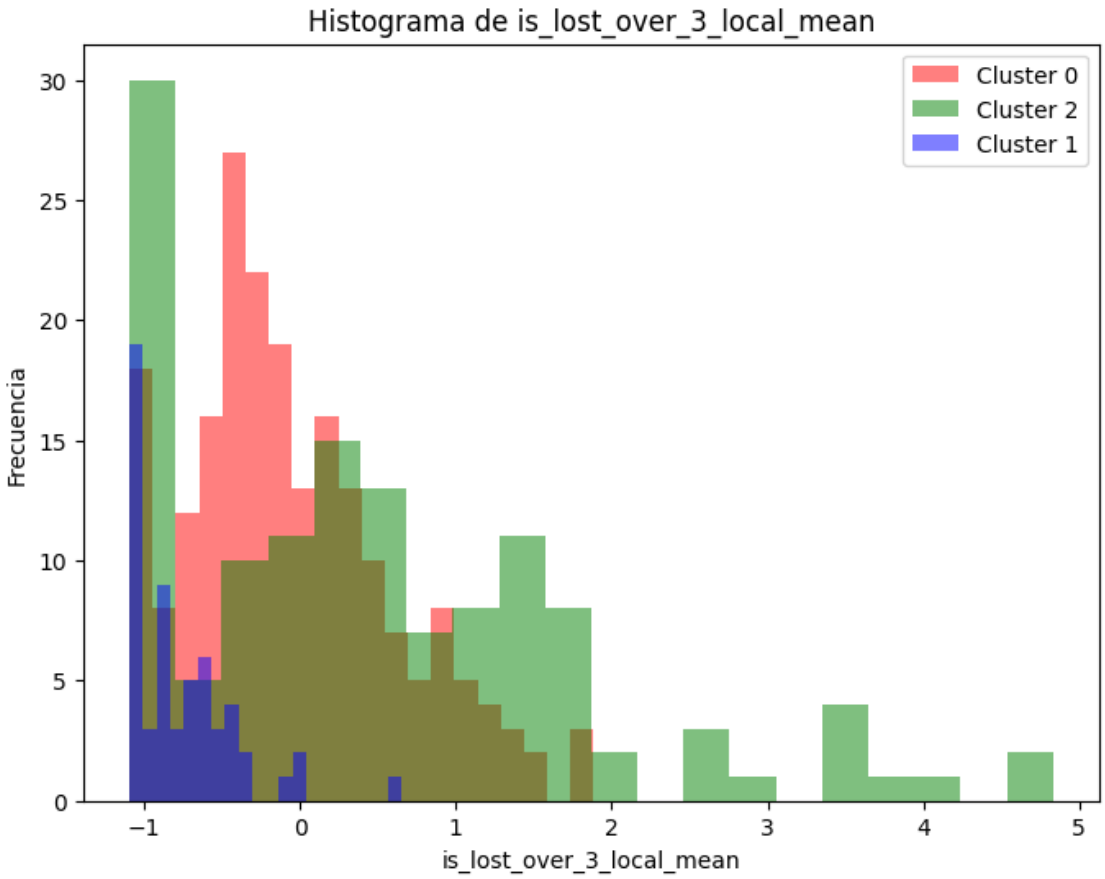

```
hola
hola
hola
```



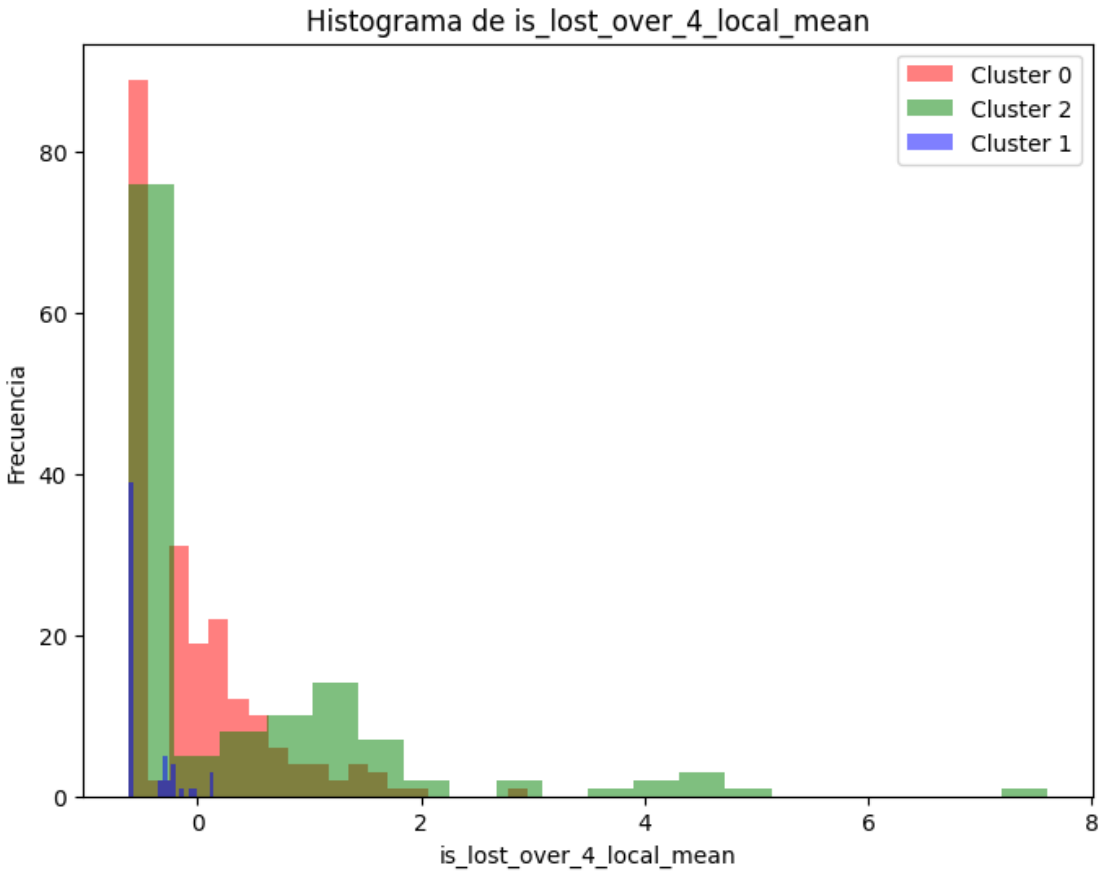

```
hola
hola
hola
```



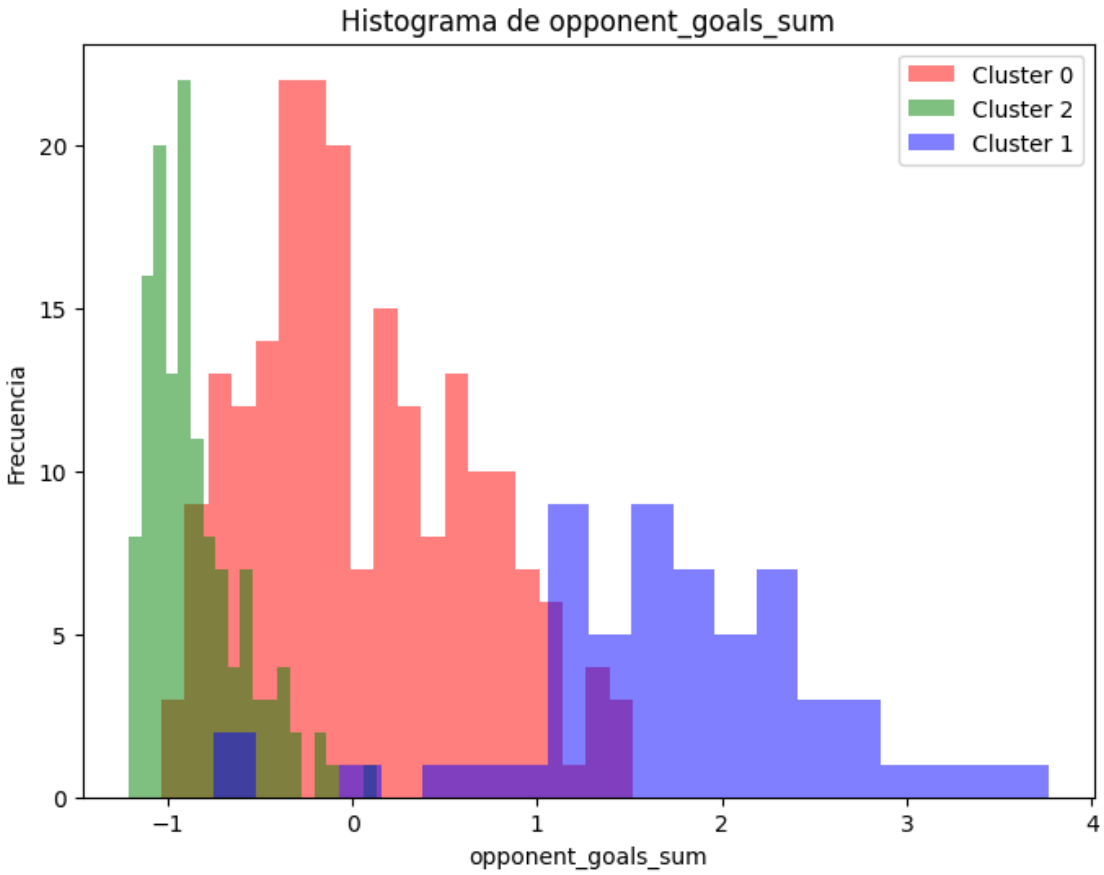

```
hola
hola
hola
```



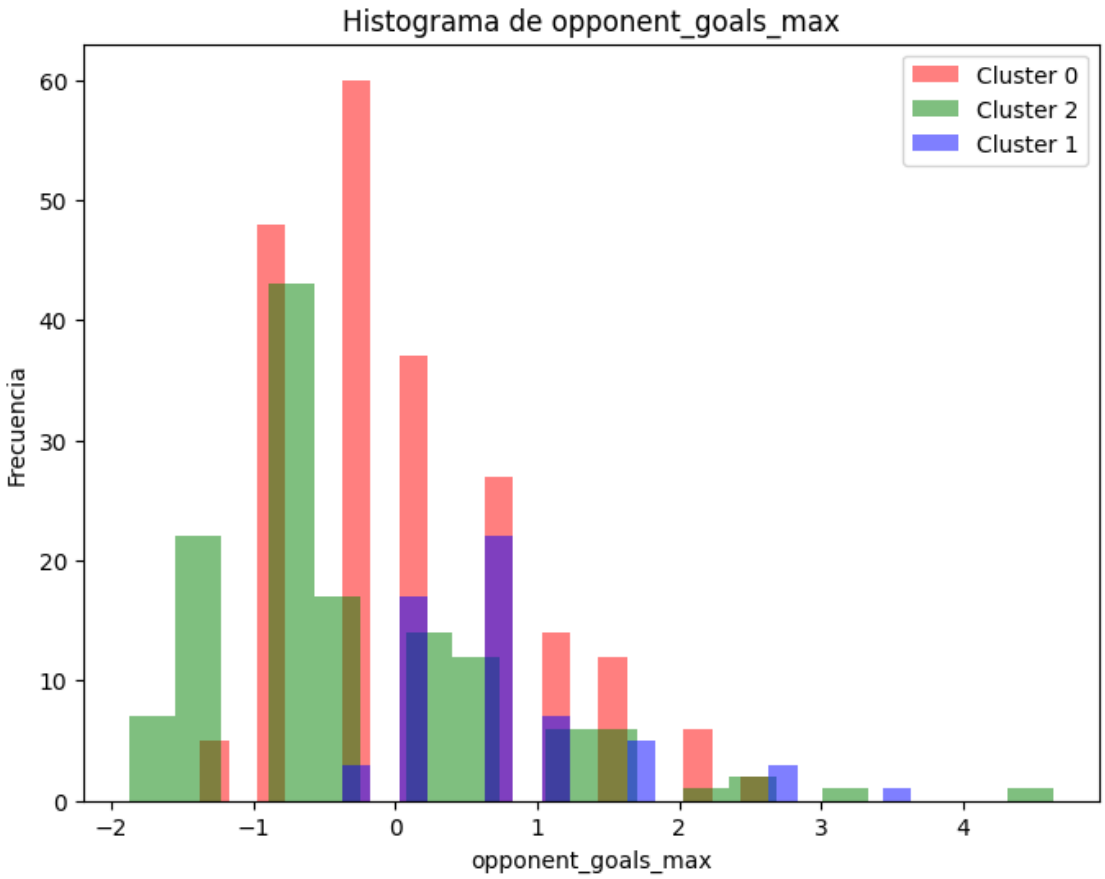

```
hola
hola
hola
```



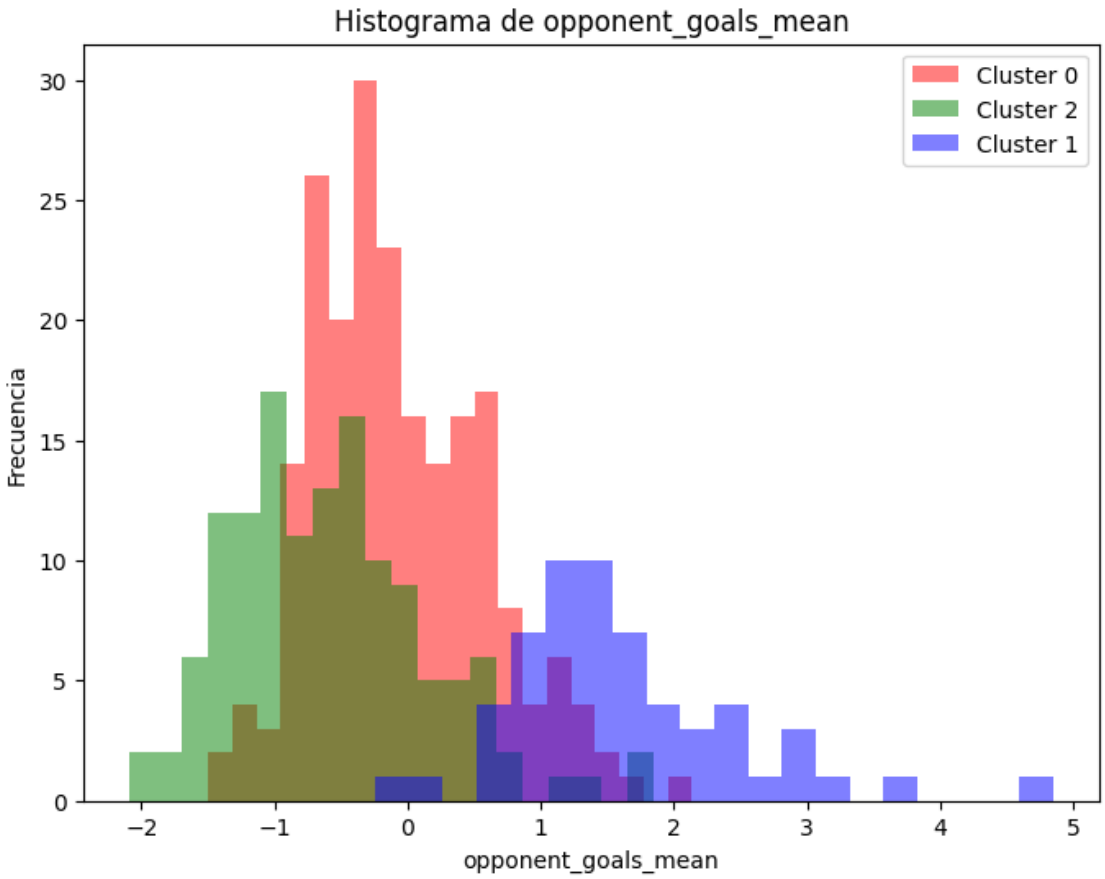

```
hola
hola
hola
```



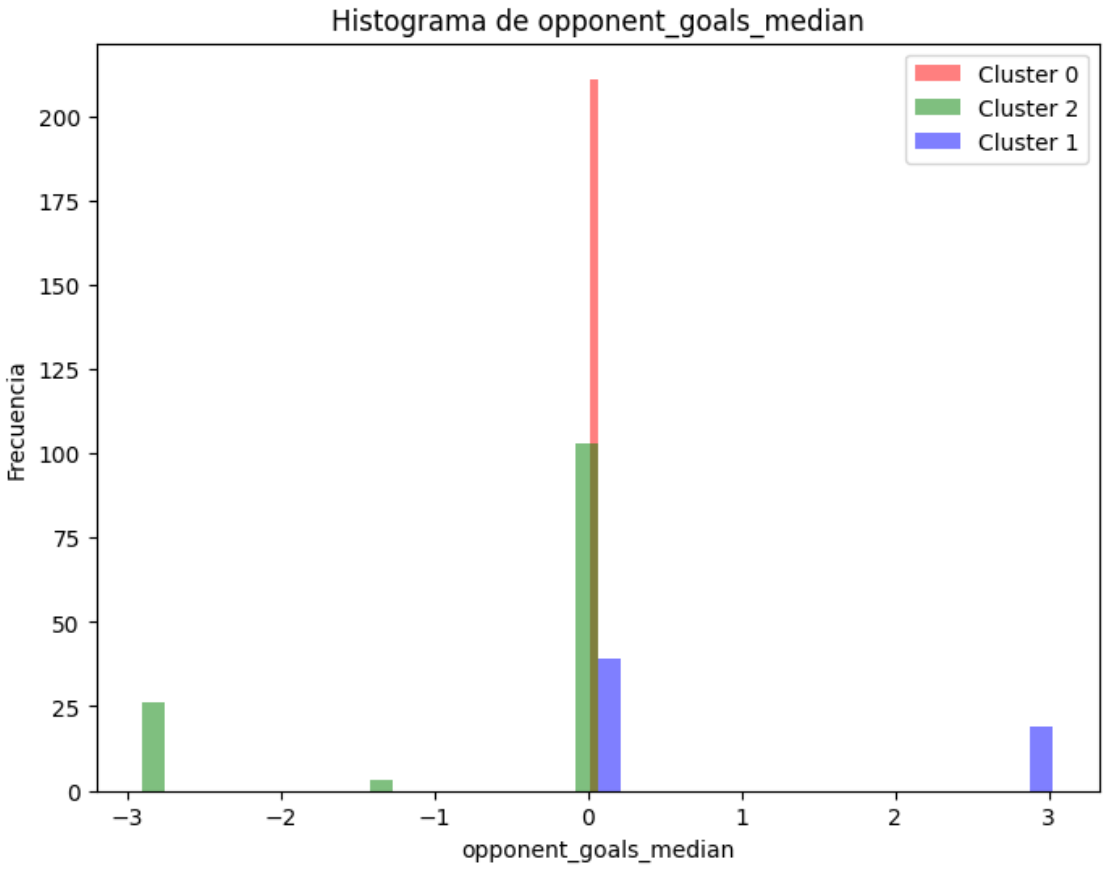

```
hola
hola
hola
```



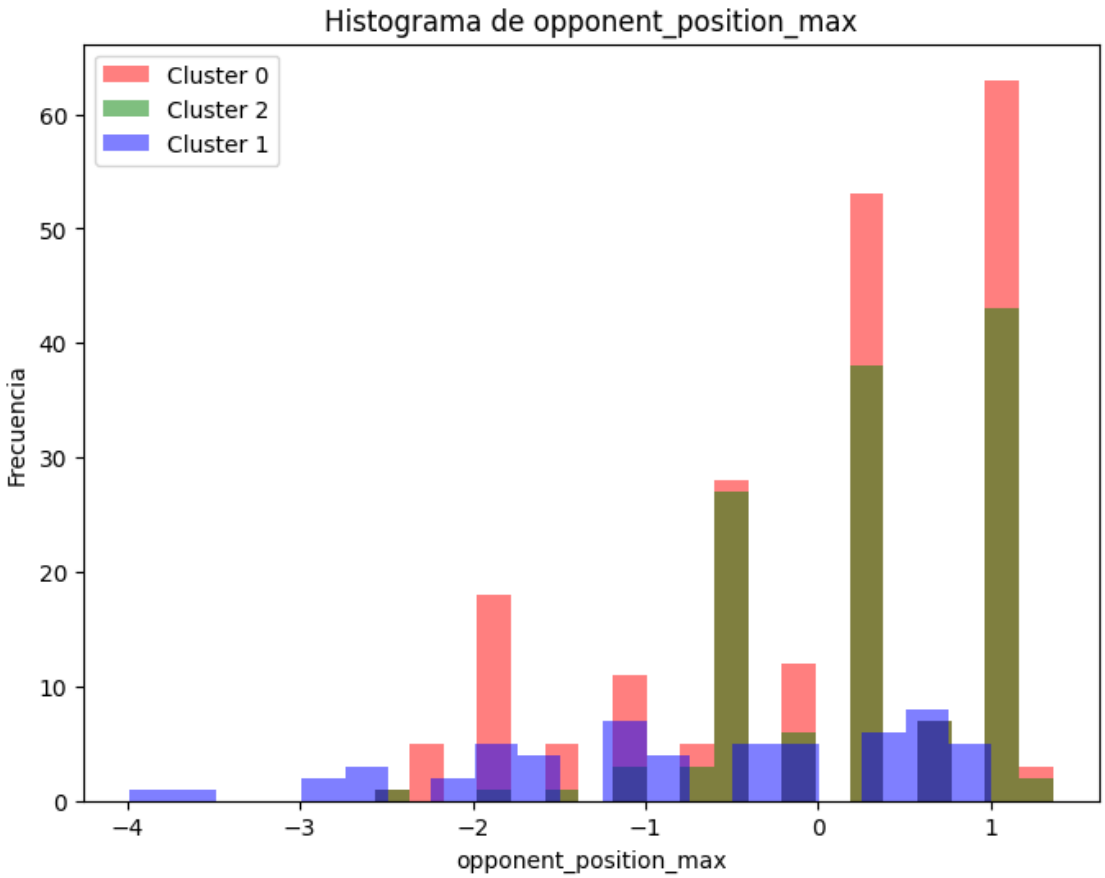

```
hola
hola
hola
```



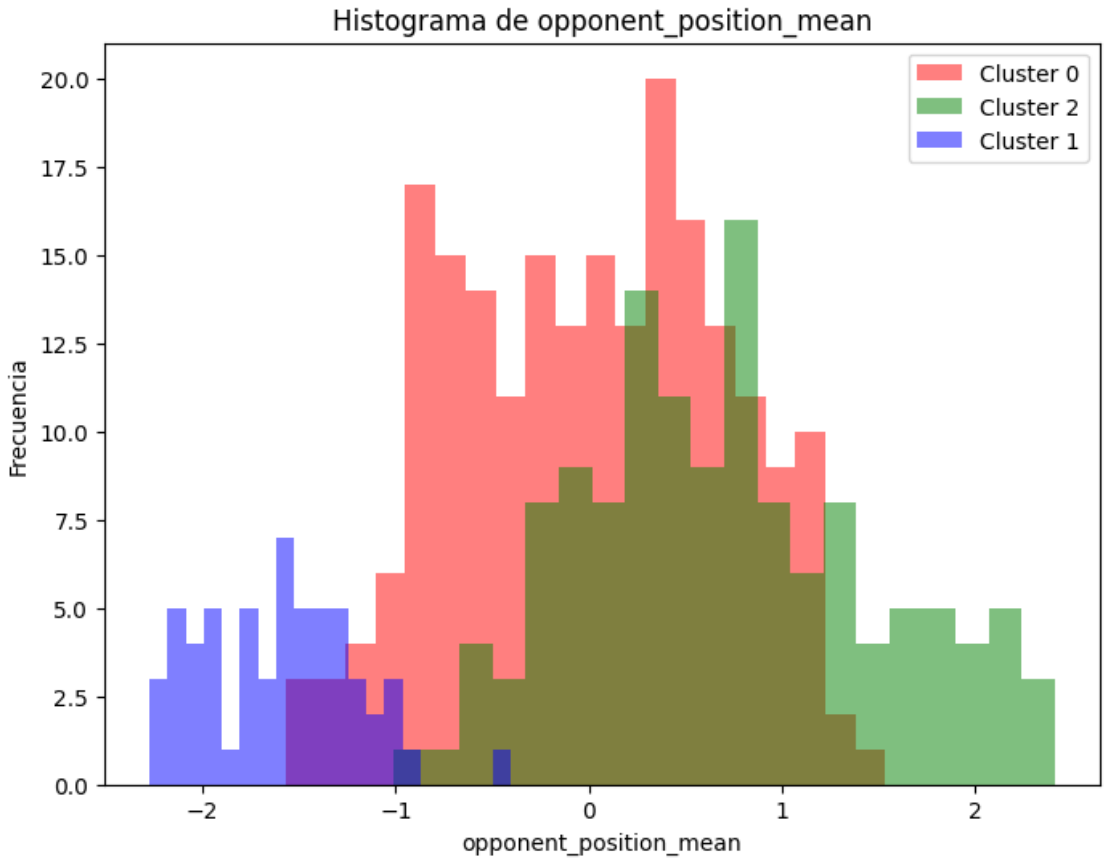

```
hola
hola
hola
```



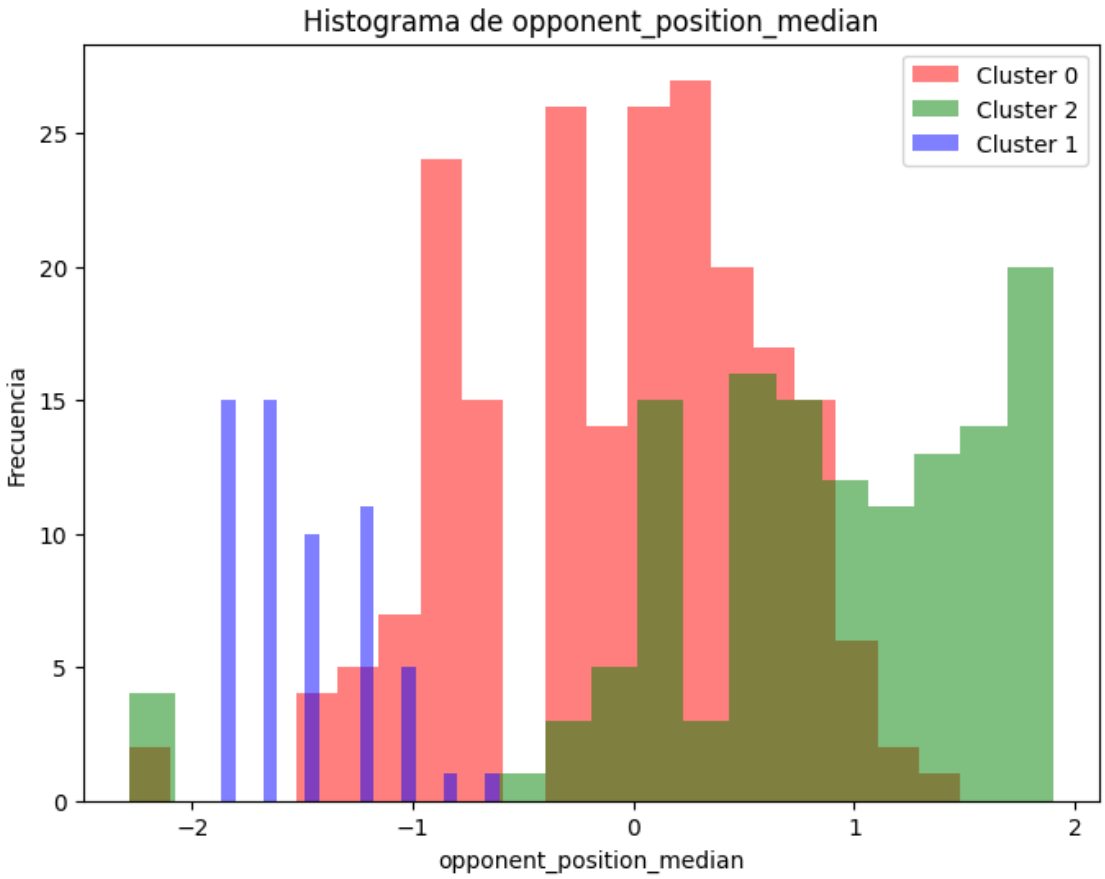

```
hola
hola
hola
```



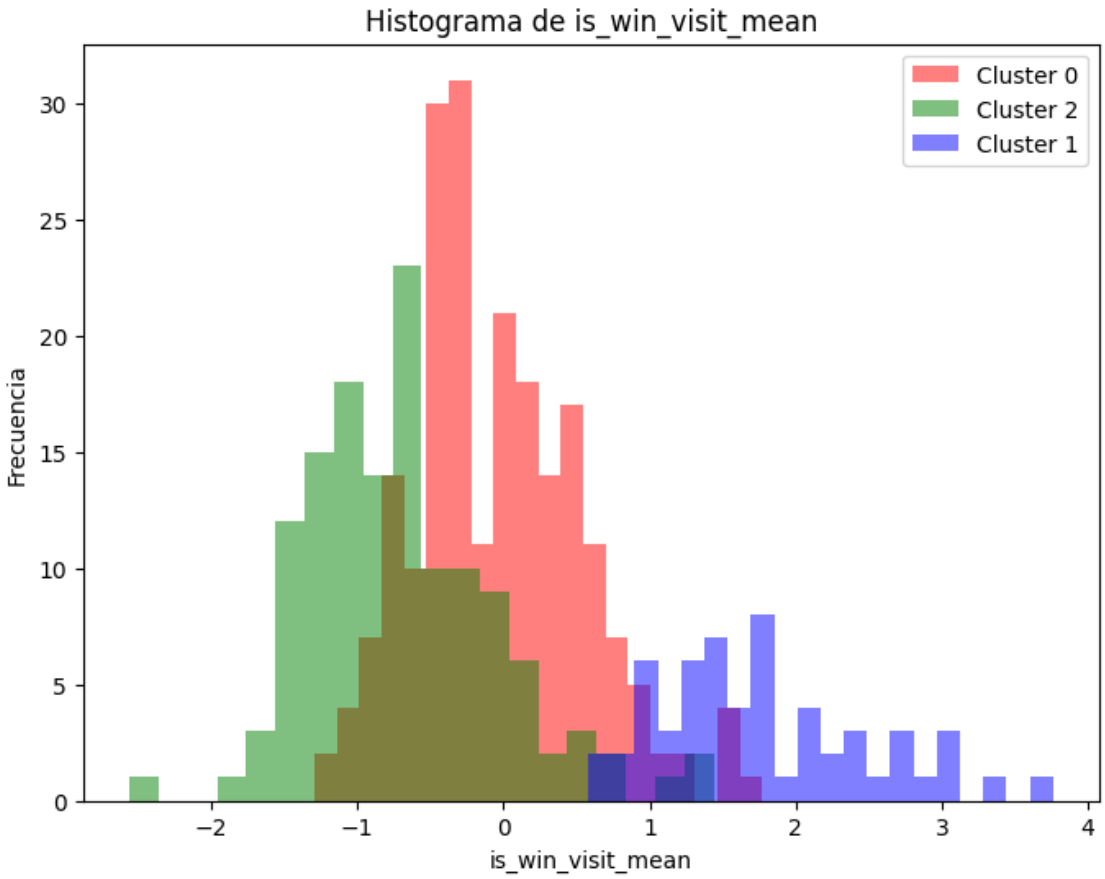

```
hola
hola
hola
```



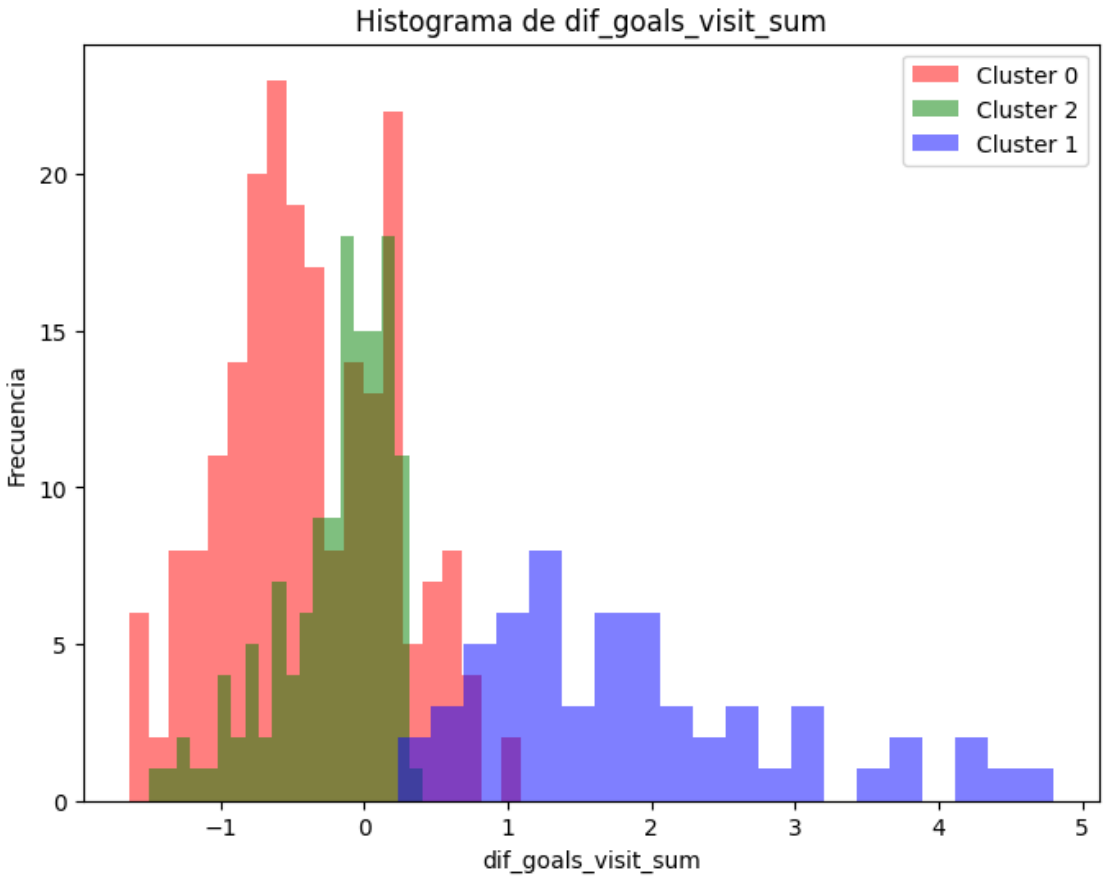

```
hola
hola
hola
```



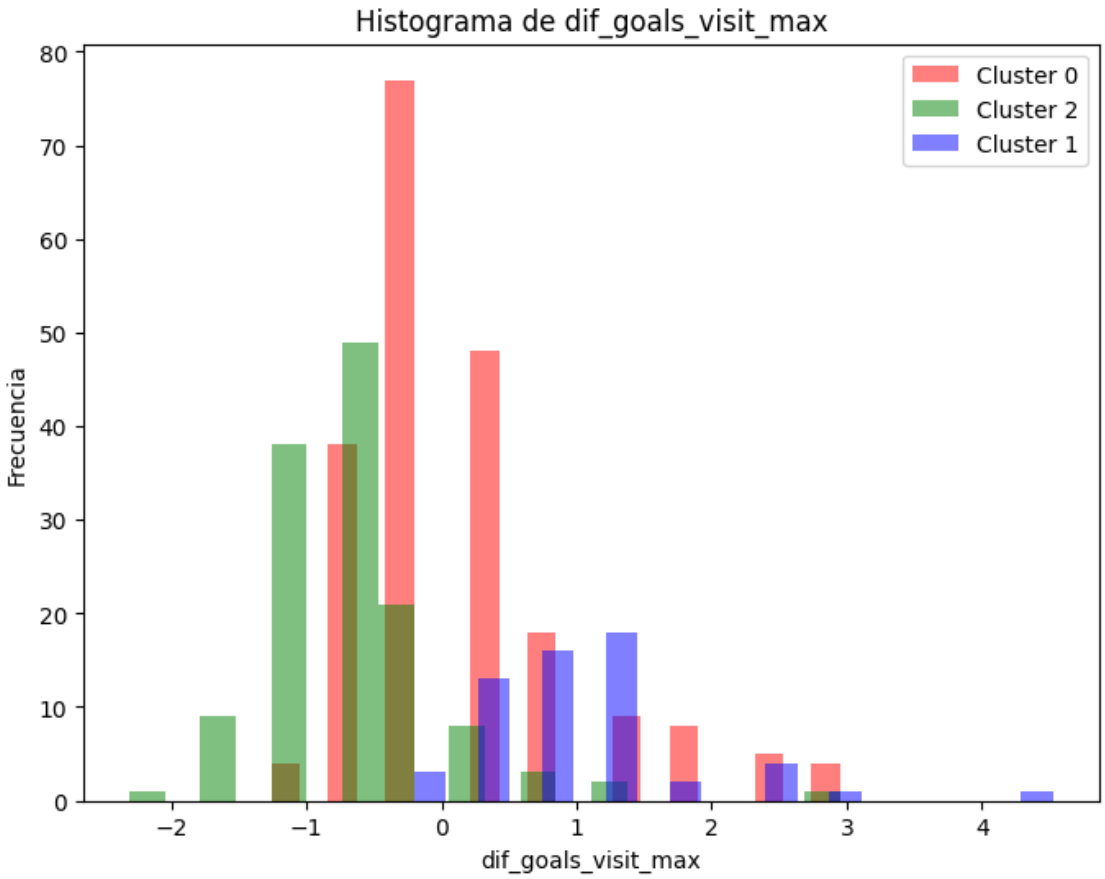

```
hola
hola
hola
```



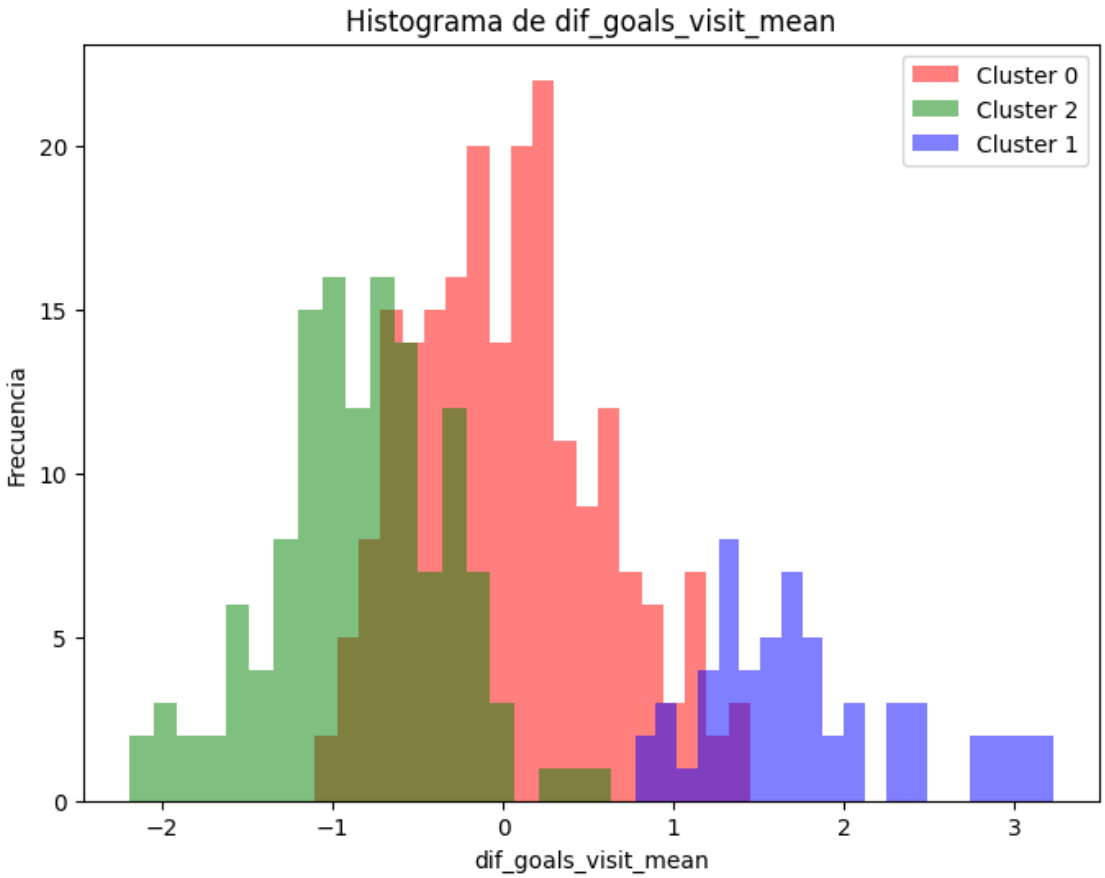

```
hola
hola
hola
```



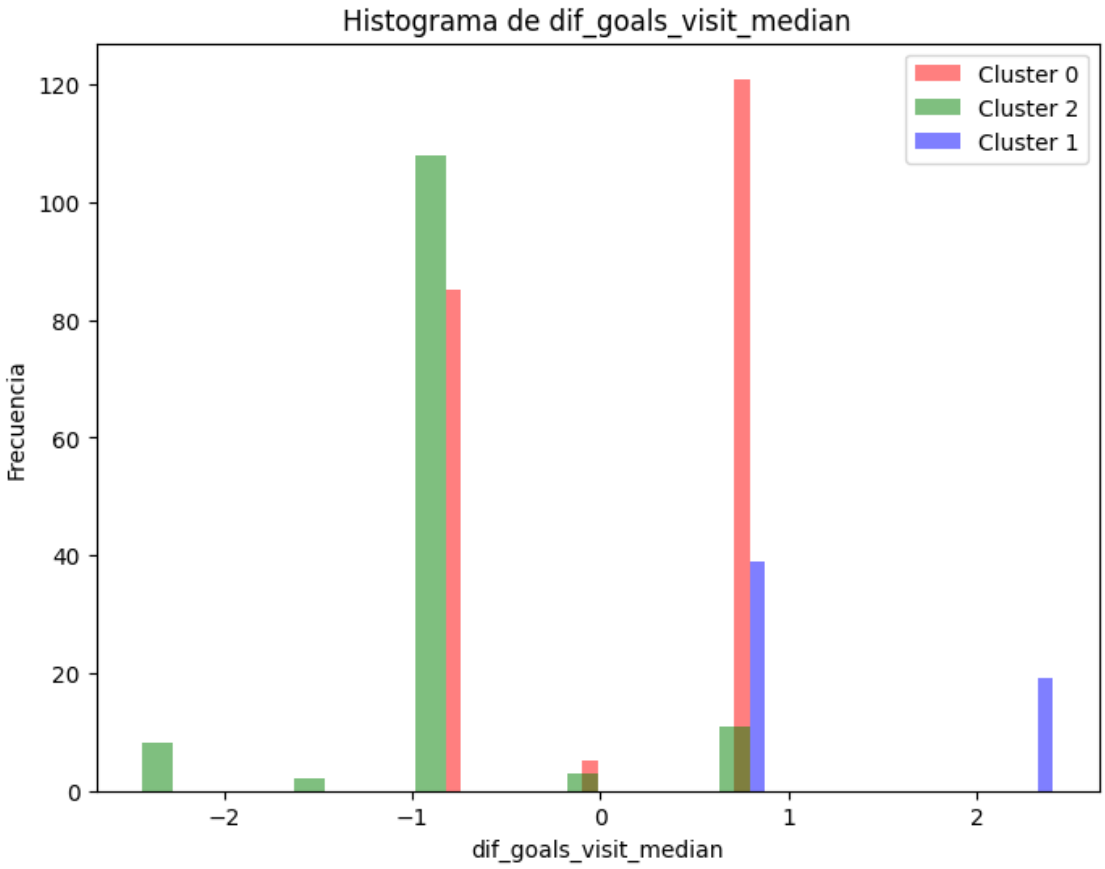

```
hola
hola
hola
```



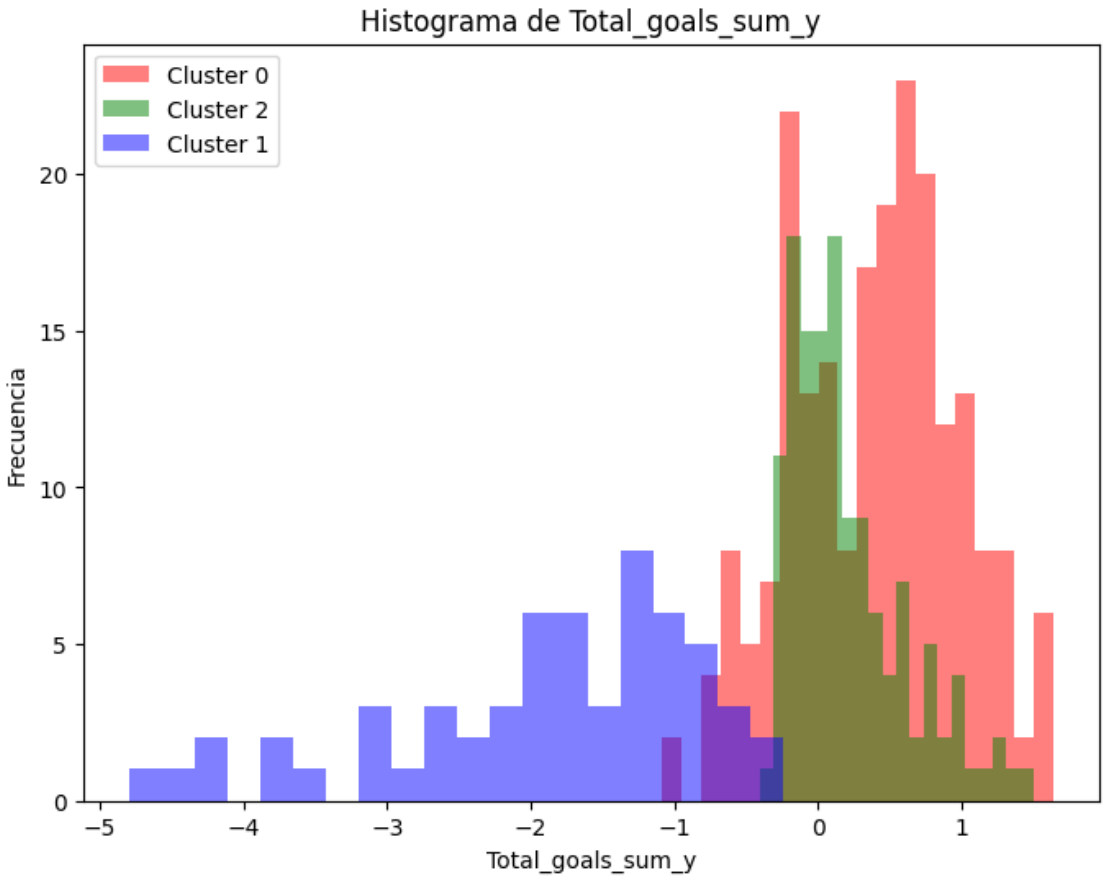

```
hola
hola
hola
```



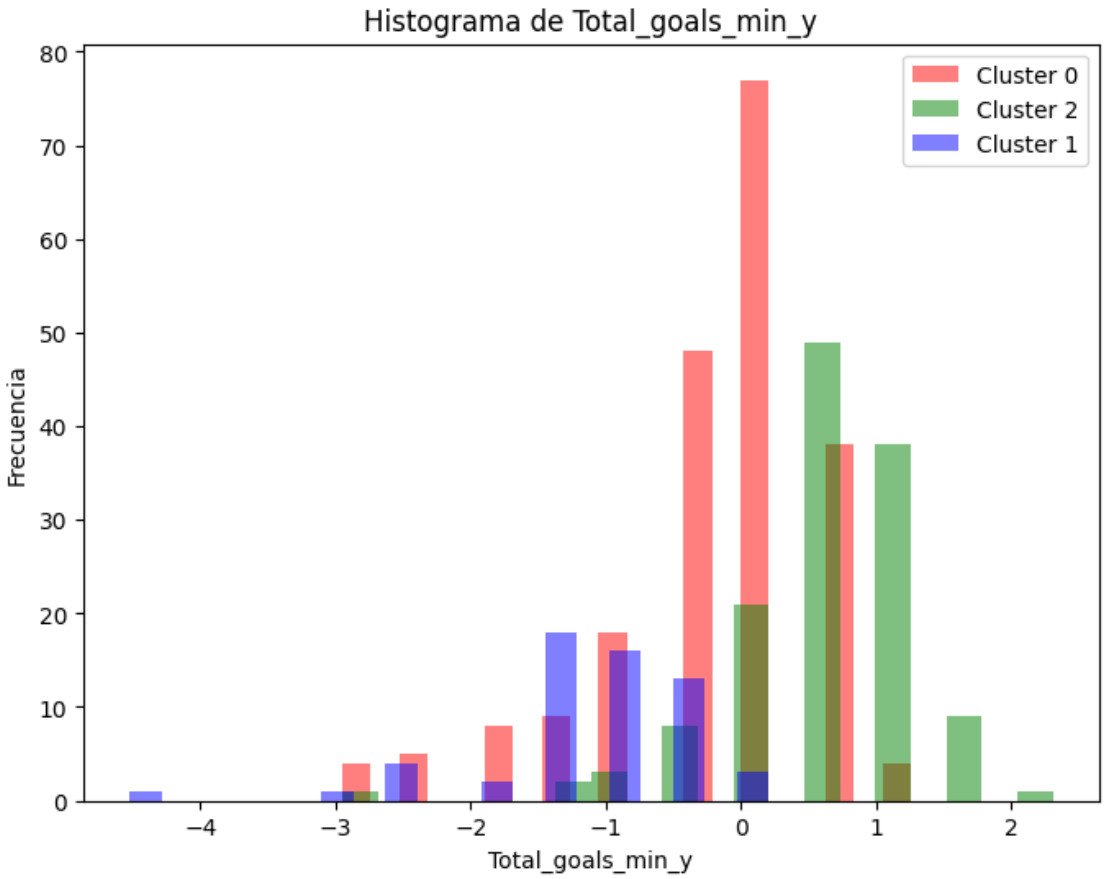

```
hola
hola
hola
```



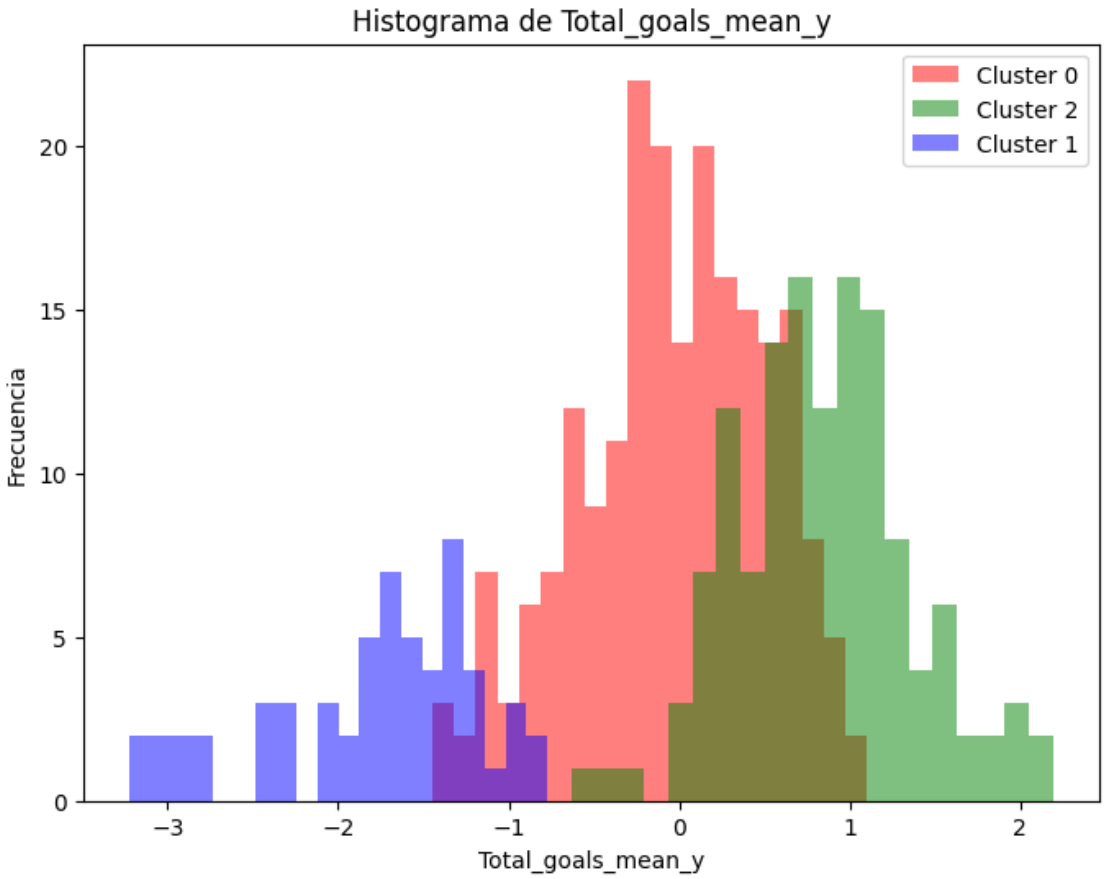

```
hola
hola
hola
```



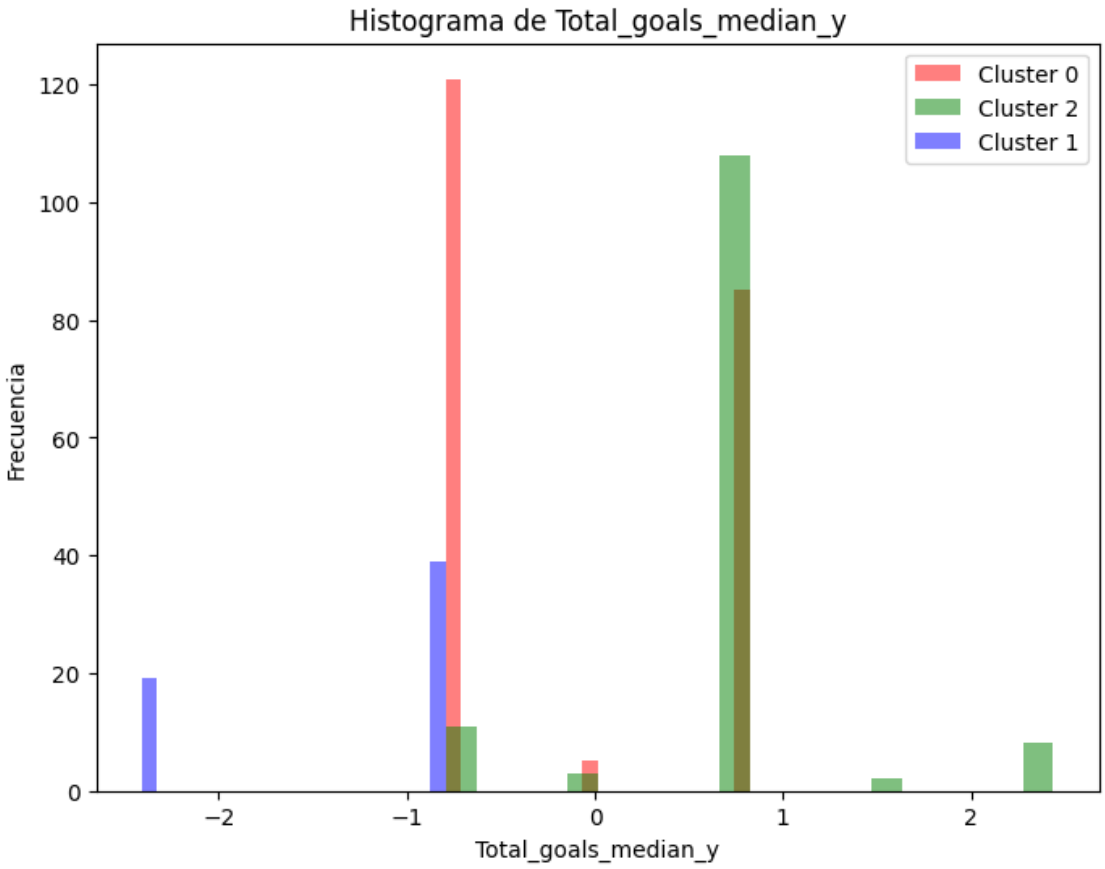

```
hola
hola
hola
```



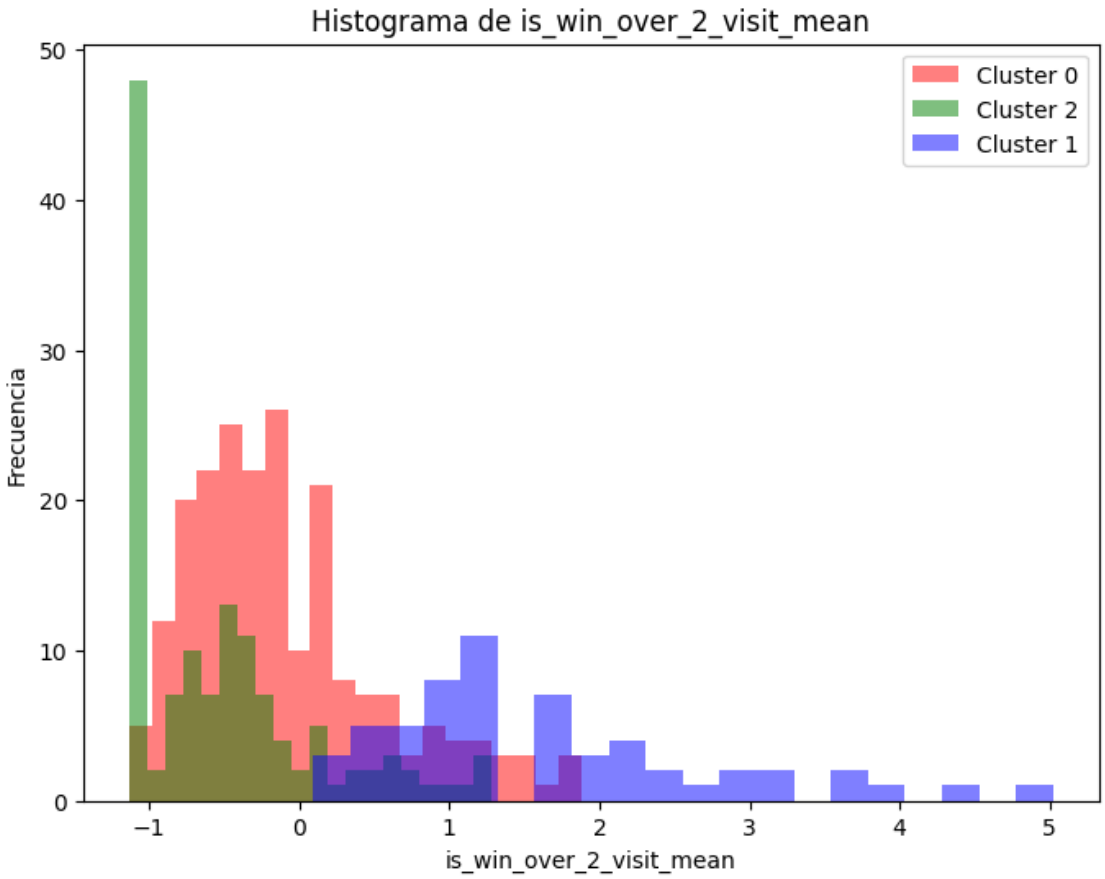

```
hola
hola
hola
```



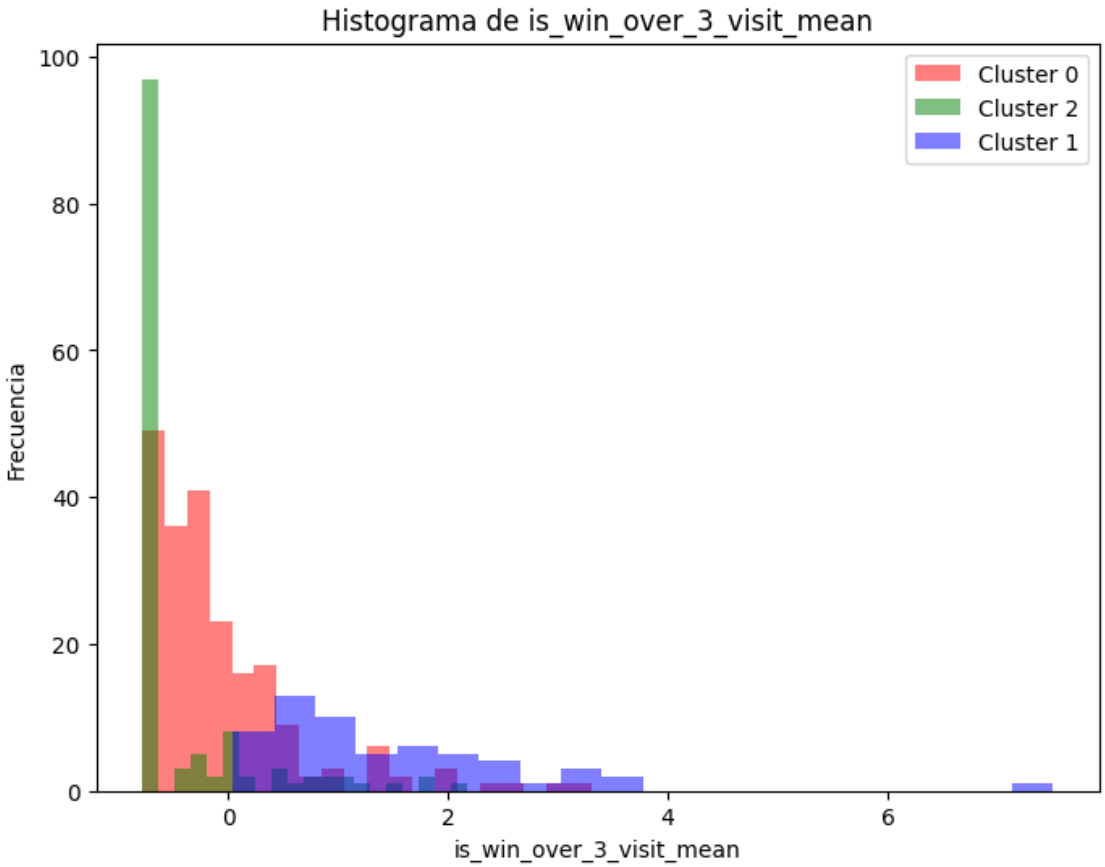

```
hola
hola
hola
```



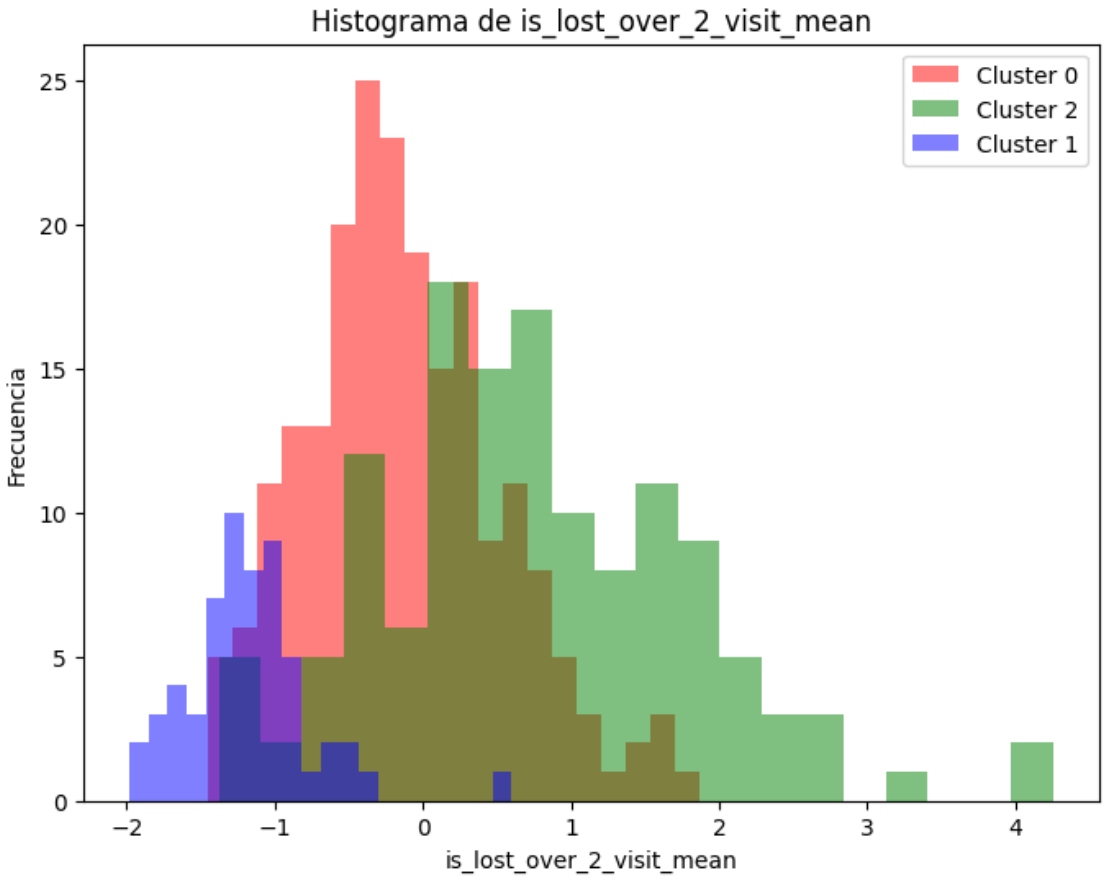

```
hola
hola
hola
```



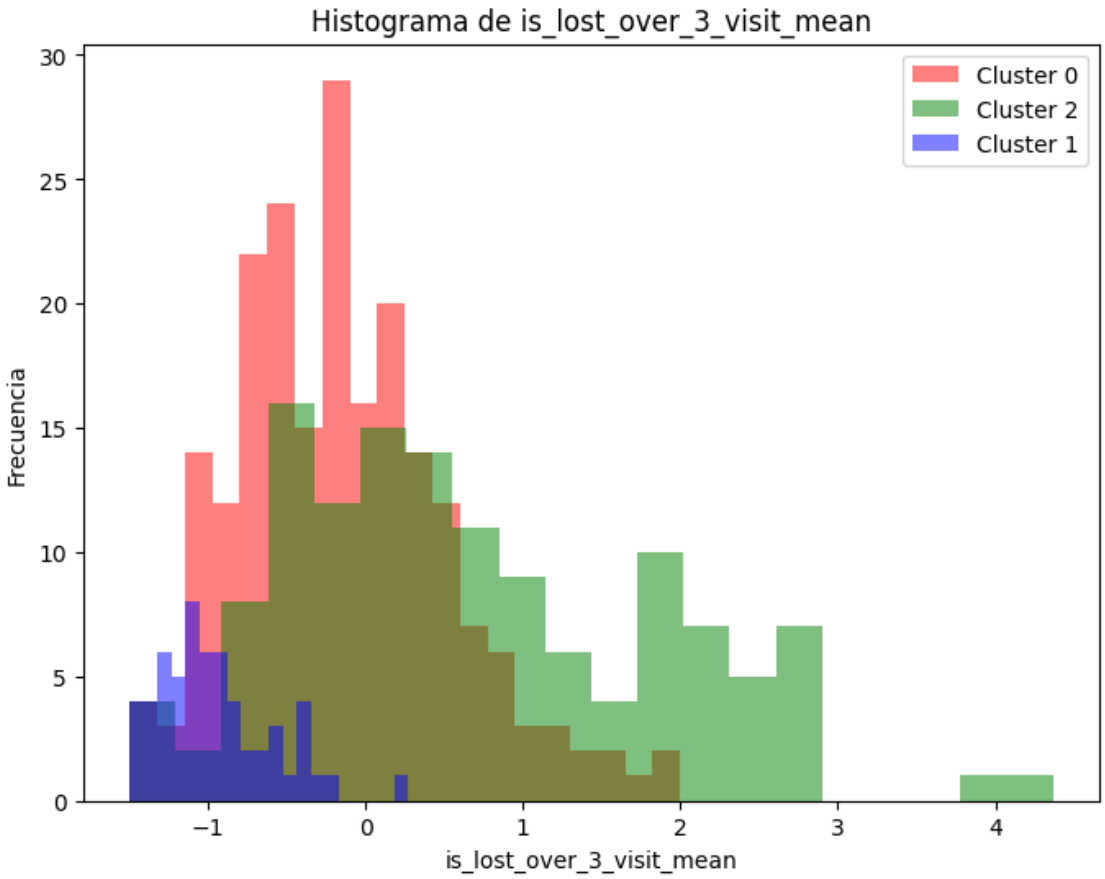

```
hola
hola
hola
```



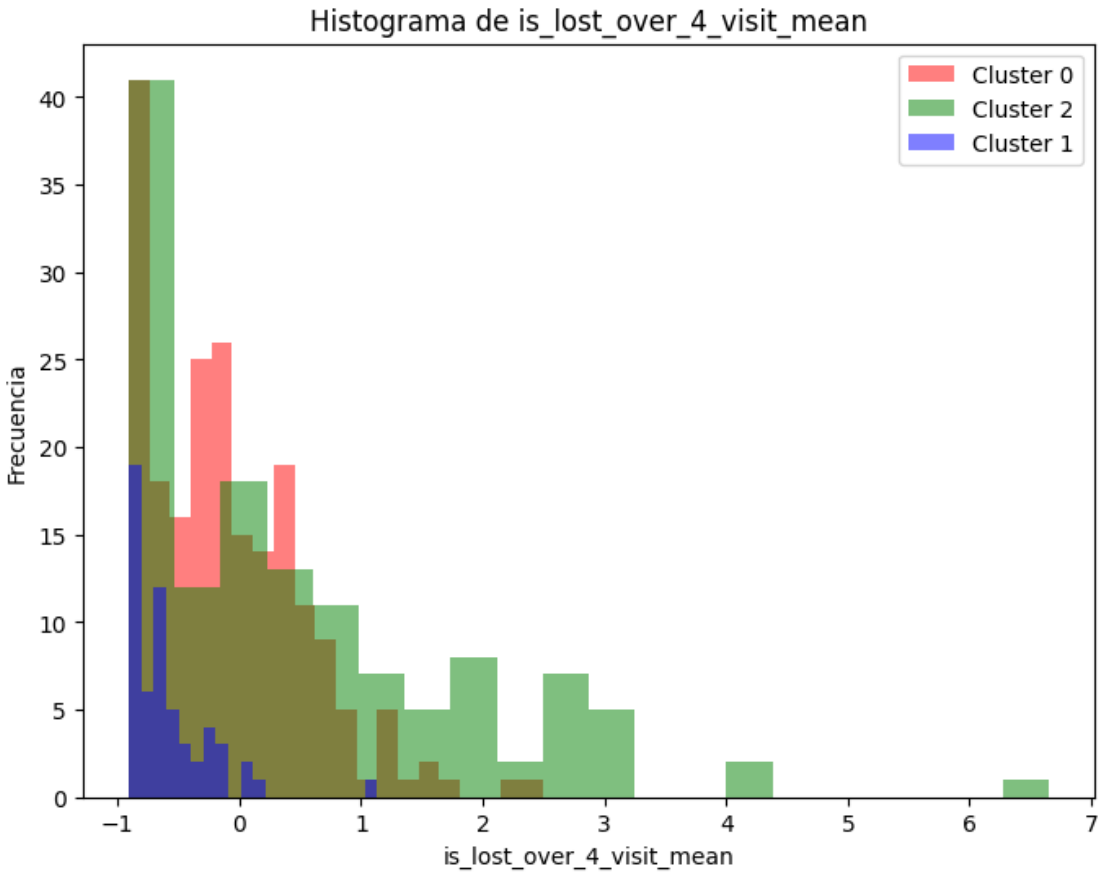

```
hola
hola
hola
```



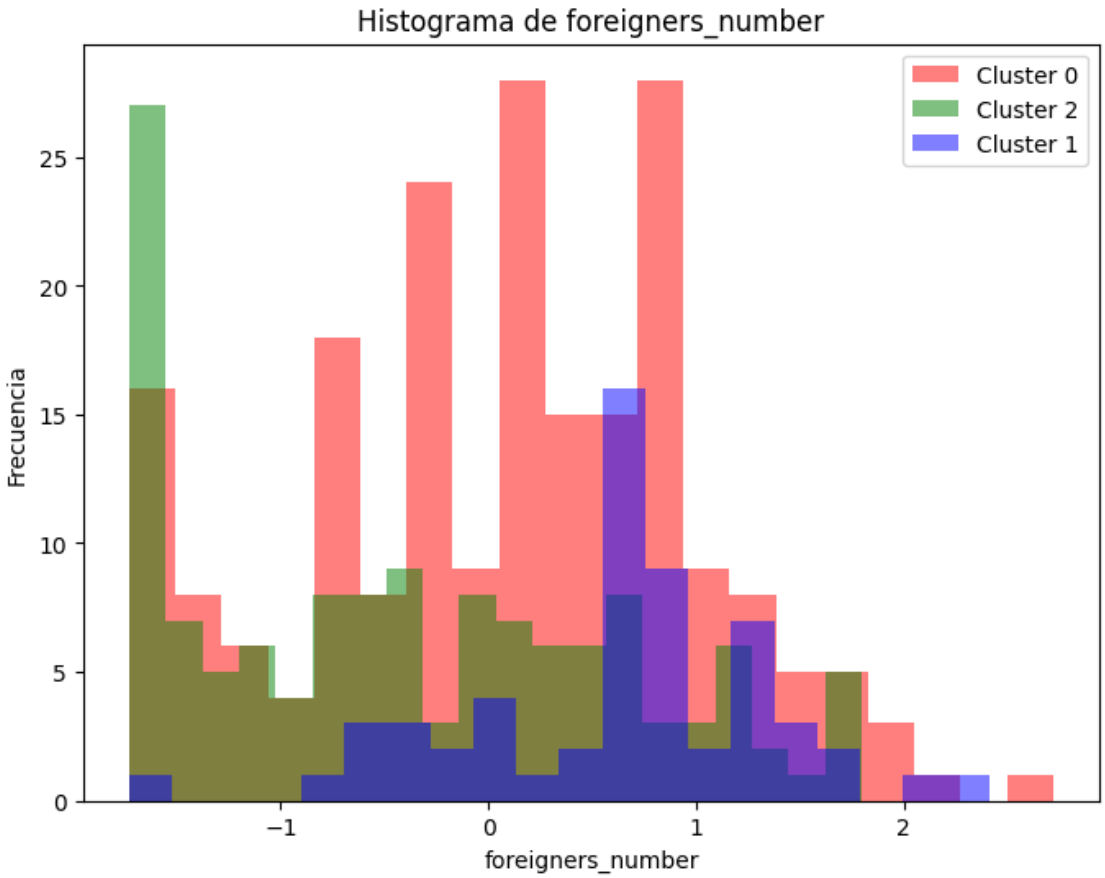

```
hola
hola
hola
```



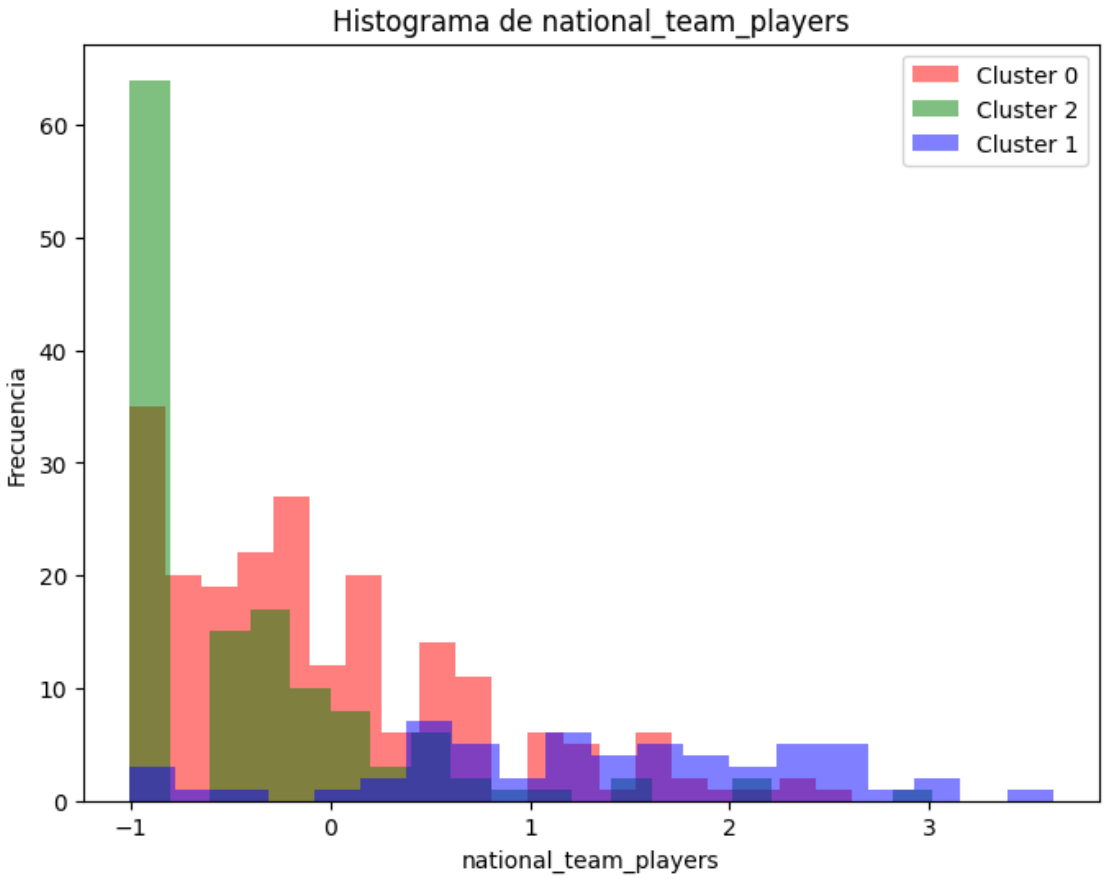

```
hola
hola
hola
```



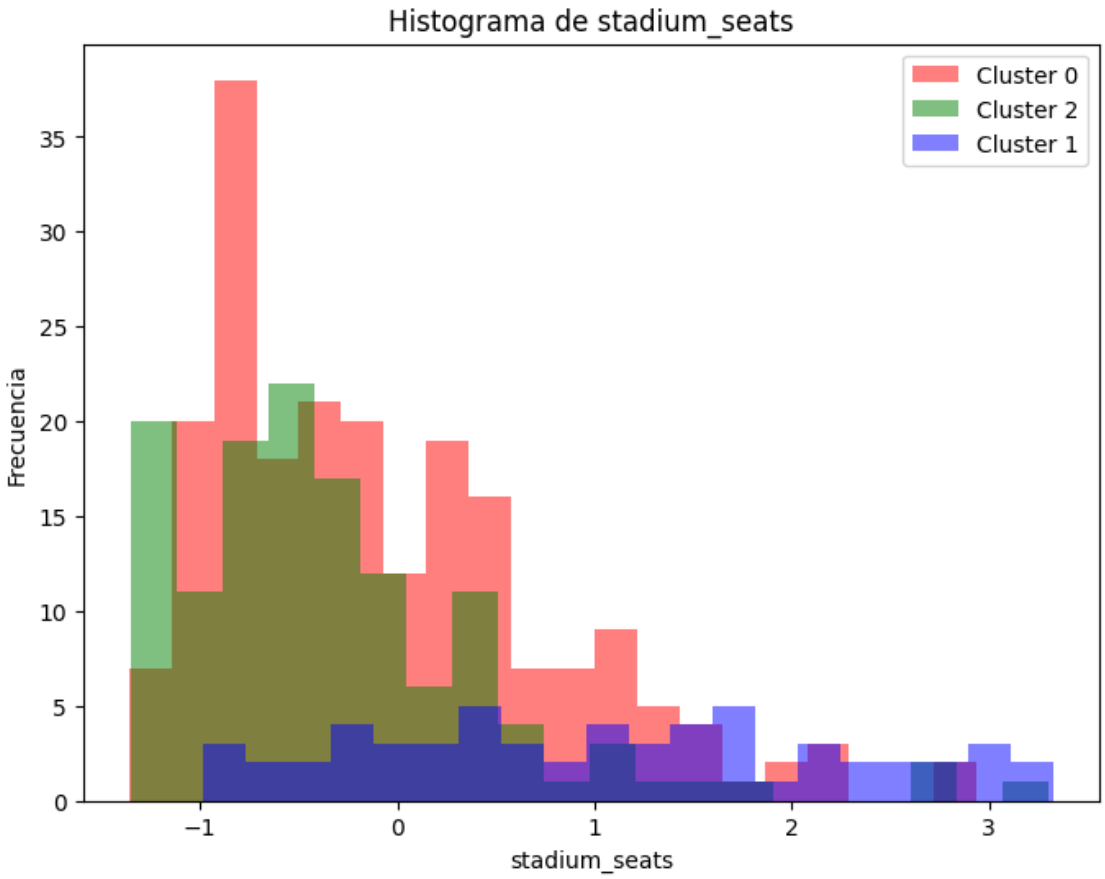

```
hola
hola
hola
```




```
hola
hola
hola
```




```
hola
hola
hola
```



7 ward

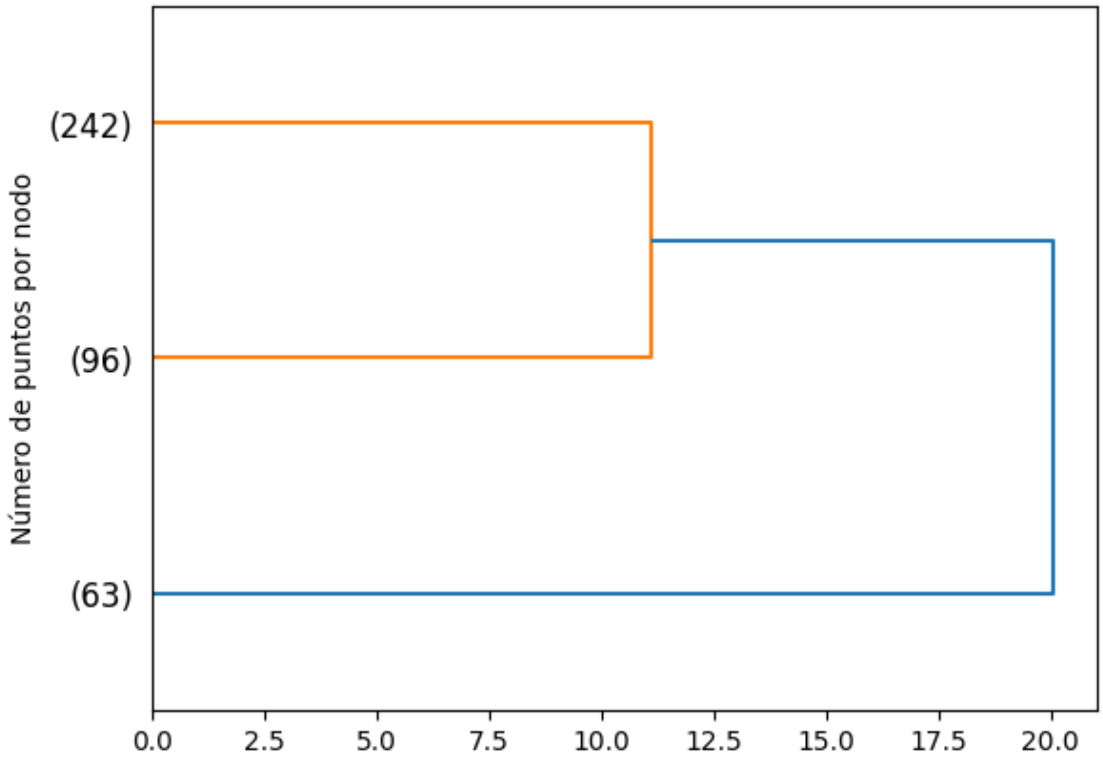
[414]:
```
# agg_model
ward_model = AgglomerativeClustering(distance_threshold=None, n_clusters=3,␣
↪linkage='ward')
ward_model_mm = ward_model.fit(Xmm)
predictions = ward_model_mm.fit_predict(Xmm)
Xmds_sample['ward_3'] = predictions
```

[415]:
```
Xpca['ward_3'] = predictions
```

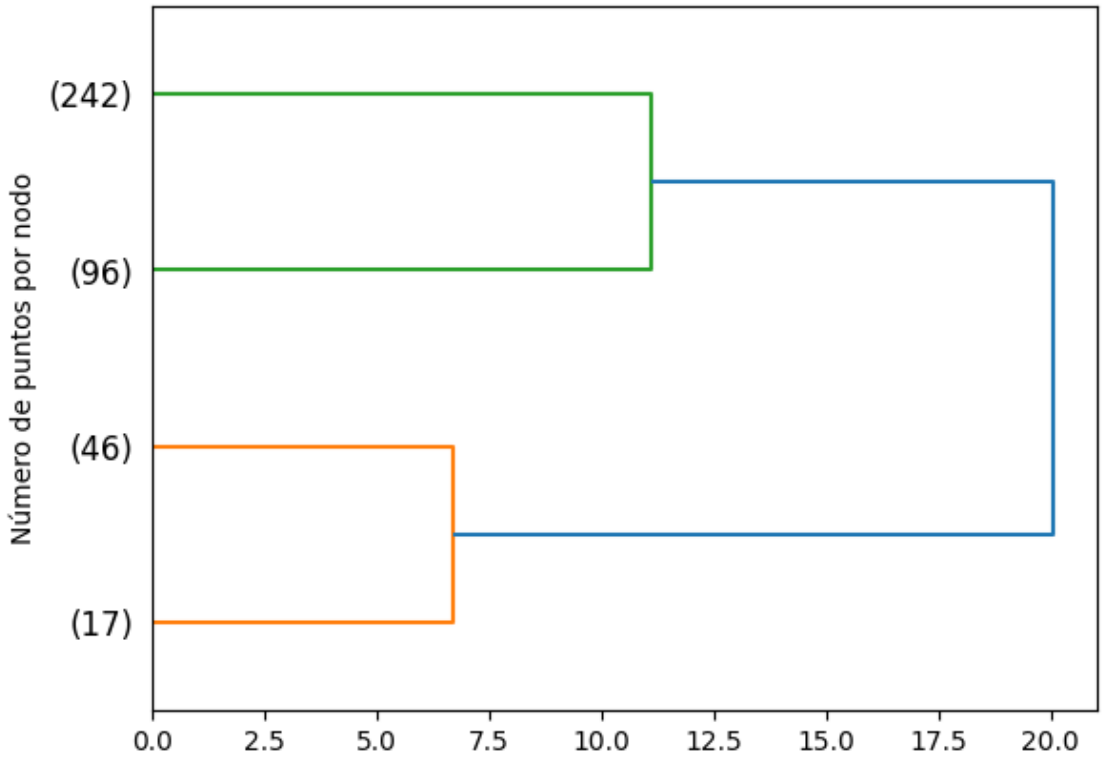
[416]:
```
from sklearn.cluster import AgglomerativeClustering


def plot_dendrogram(model, **kwargs):
    # Create linkage matrix and then plot the dendrogram

    # create the counts of samples under each node
```

```
    counts = np.zeros(model.children_.shape[0])
    n_samples = len(model.labels_)
    for i, merge in enumerate(model.children_):
        current_count = 0
        for child_idx in merge:
            if child_idx < n_samples:
                current_count += 1  # leaf node
            else:
                current_count += counts[child_idx - n_samples]
        counts[i] = current_count

    linkage_matrix = np.column_stack(
        [model.children_, model.distances_, counts]
    ).astype(float)

    # Plot the corresponding dendrogram
    dendrogram(linkage_matrix, **kwargs)
```

[417]:
```
# agg_model
ward_model = AgglomerativeClustering(distance_threshold=0, n_clusters=None,␣
 ↪linkage='ward')
ward_model_den = ward_model.fit(Xmm)
plot_dendrogram(ward_model_den, orientation='right', truncate_mode='lastp', p=3)
plt.ylabel('Número de puntos por nodo')
plt.show()
```

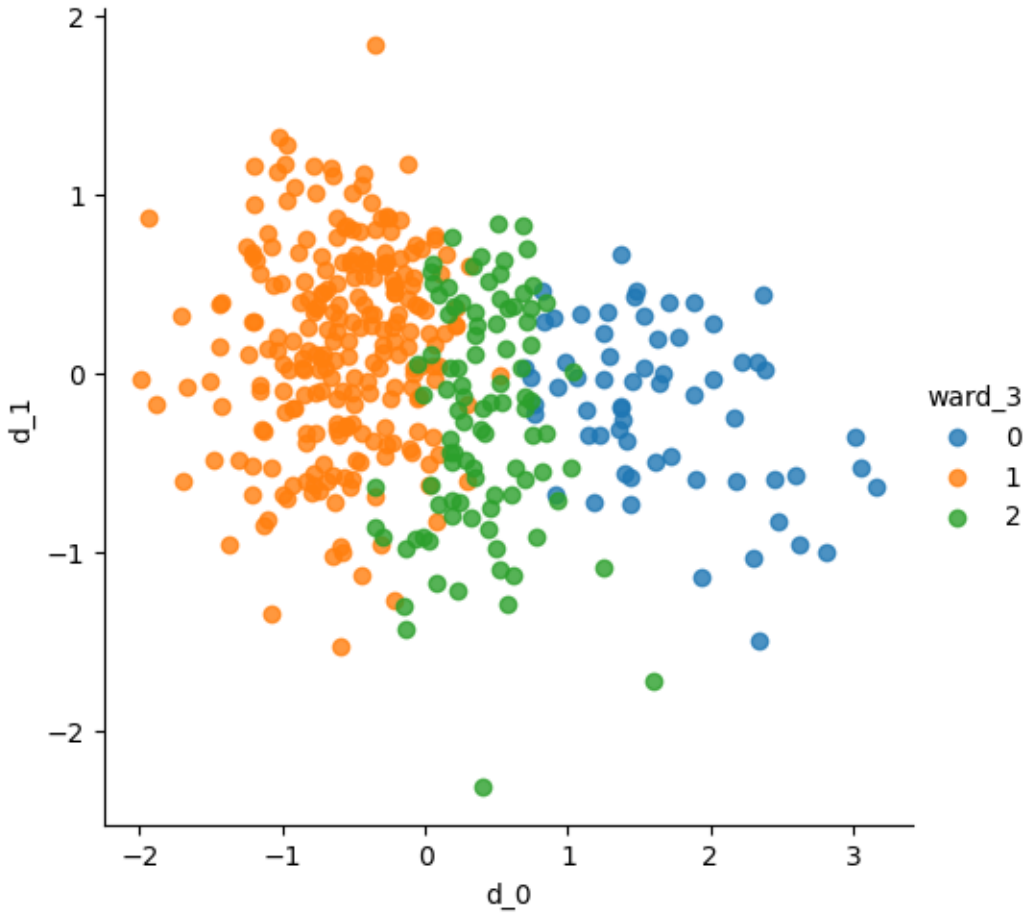


[418]:
```
plot_dendrogram(ward_model_den, orientation='right', truncate_mode='lastp', p=4)
plt.ylabel('Número de puntos por nodo')
plt.show()
```



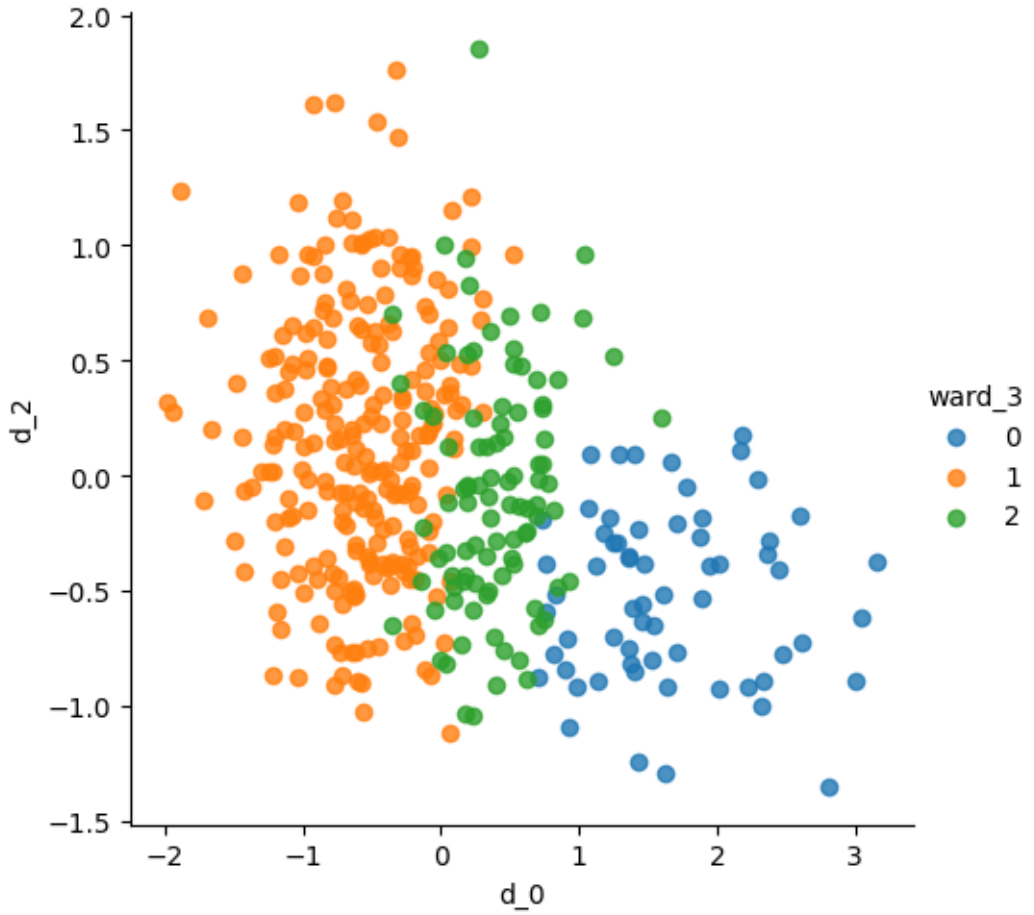
[419]:
```
sns.lmplot(data=Xmds_sample, x='d_0', y='d_1', fit_reg=False, hue='ward_3')
```

[419]: <seaborn.axisgrid.FacetGrid at 0x7fdeb8e68c10>



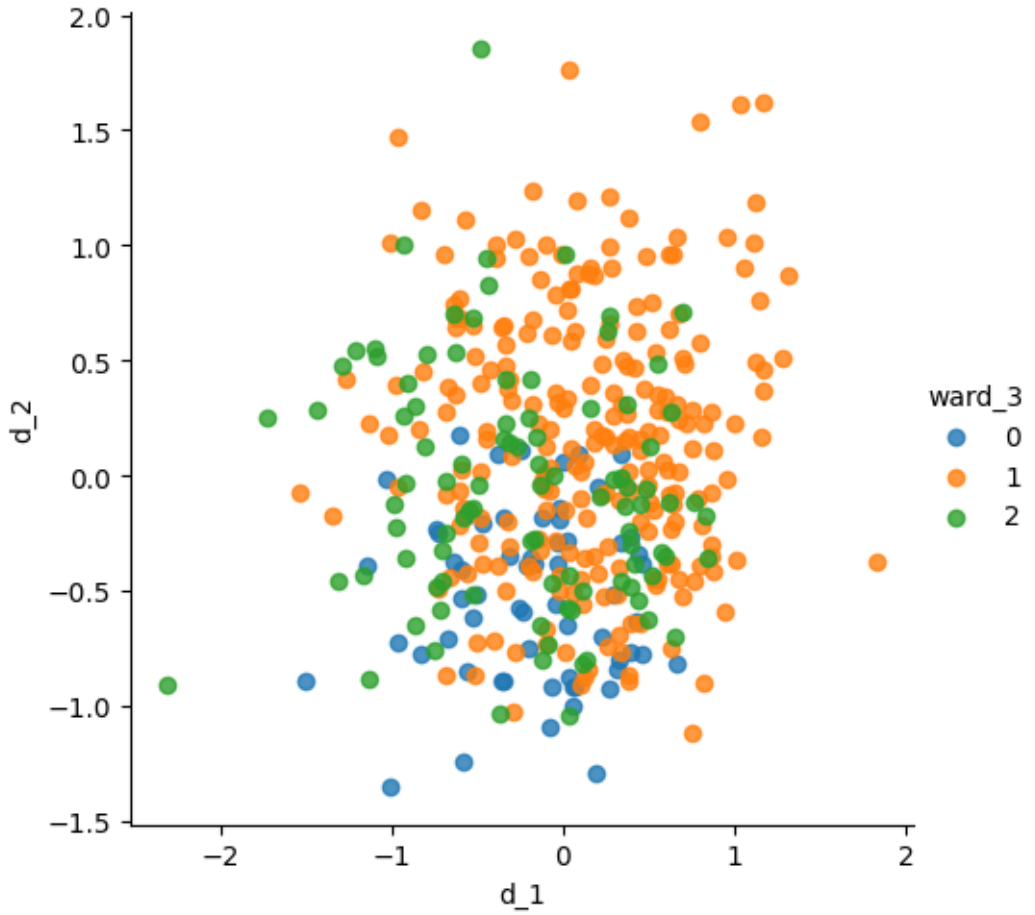
[420]:
```
sns.lmplot(data=Xmds_sample, x='d_0', y='d_2', fit_reg=False, hue='ward_3')
```

[420]: <seaborn.axisgrid.FacetGrid at 0x7fdeb8fa4a90>



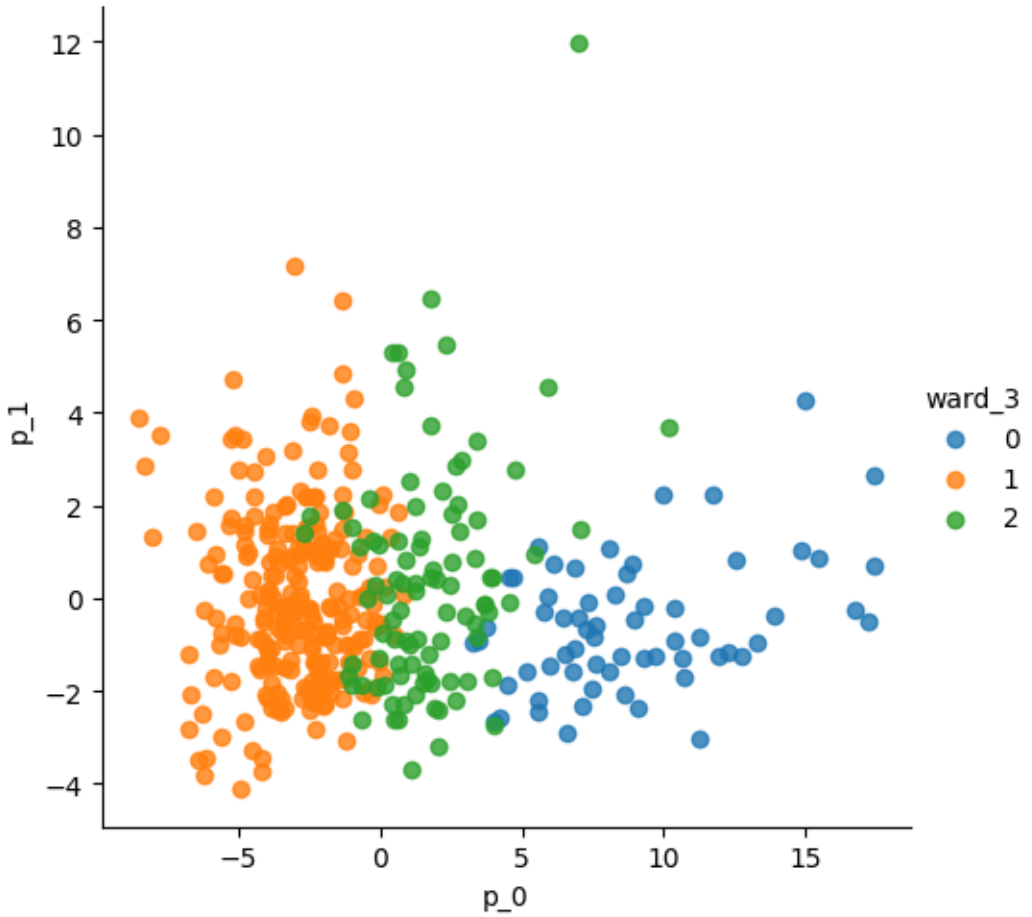
[421]:
```
sns.lmplot(data=Xmds_sample, x='d_1', y='d_2', fit_reg=False, hue='ward_3')
```

[421]: <seaborn.axisgrid.FacetGrid at 0x7fdeba5f7f10>



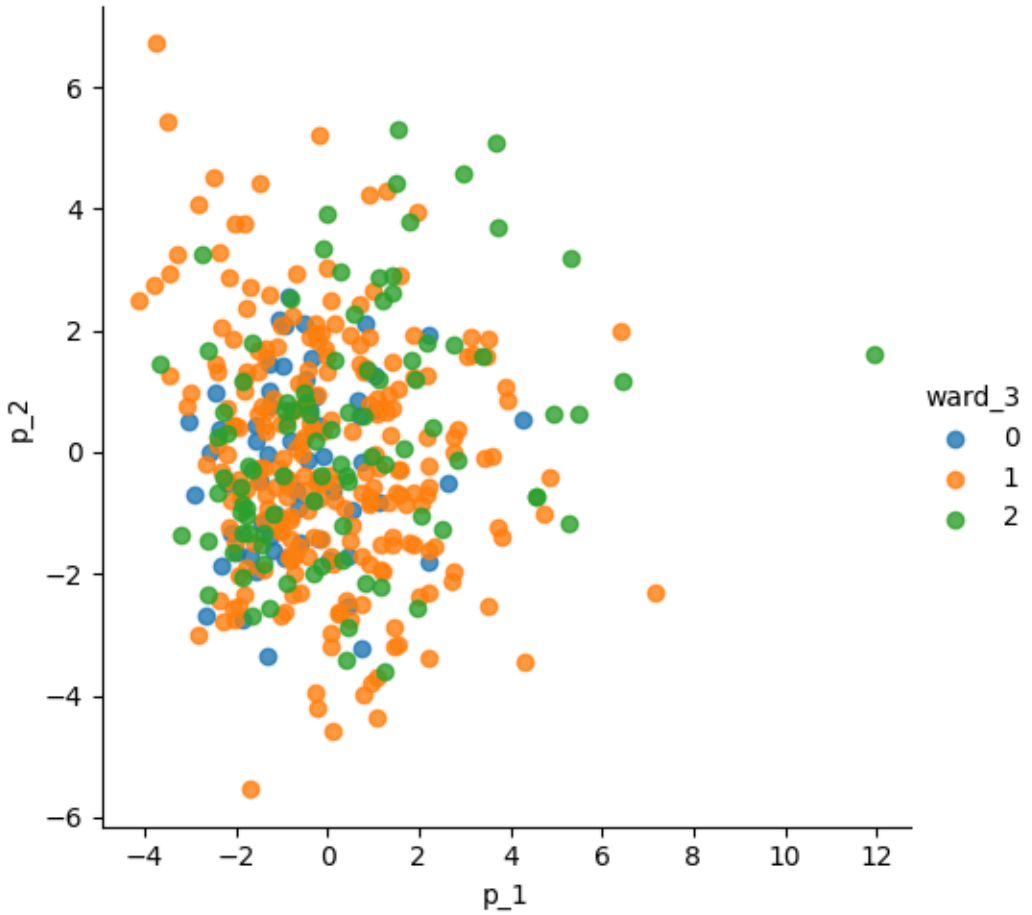
[422]:
```
sns.lmplot(data=Xpca, x='p_0', y='p_1', fit_reg=False, hue='ward_3')
```

[422]:
```
<seaborn.axisgrid.FacetGrid at 0x7fdeb91c6b00>
```



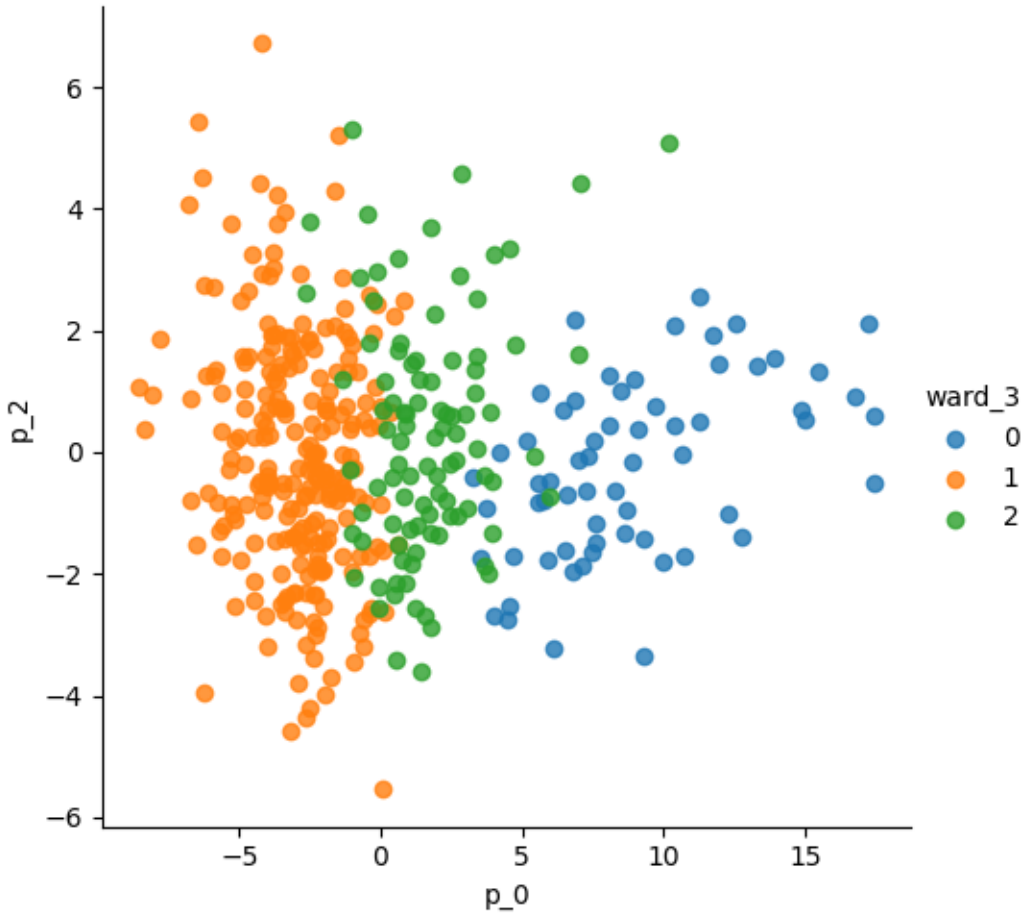
[423]:
```
sns.lmplot(data=Xpca, x='p_1', y='p_2', fit_reg=False, hue='ward_3')
```

[423]: <seaborn.axisgrid.FacetGrid at 0x7fdeb93b8ee0>

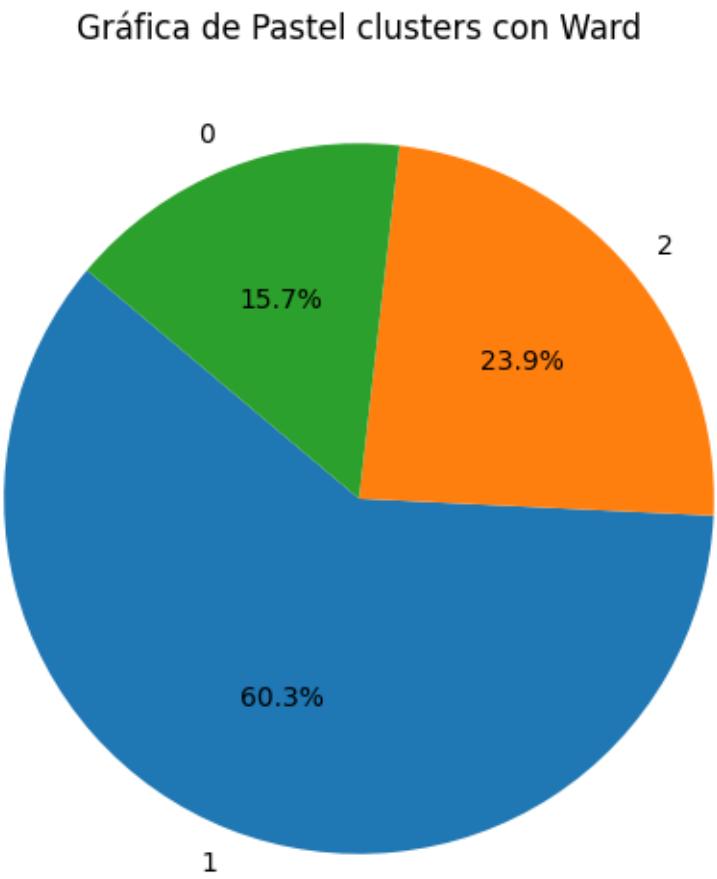


[424]:
```
sns.lmplot(data=Xpca, x='p_0', y='p_2', fit_reg=False, hue='ward_3')
```

[424]: <seaborn.axisgrid.FacetGrid at 0x7fdeb9c97280>



[425]:
```
conteo_clusters = Xmds_sample['ward_3'].value_counts()
plt.figure(figsize=(6, 6))
plt.pie(conteo_clusters, labels=conteo_clusters.index, autopct='%1.1f%%',␣
 ↪startangle=140)
plt.title('Gráfica de Pastel clusters con Ward')
plt.show()
```



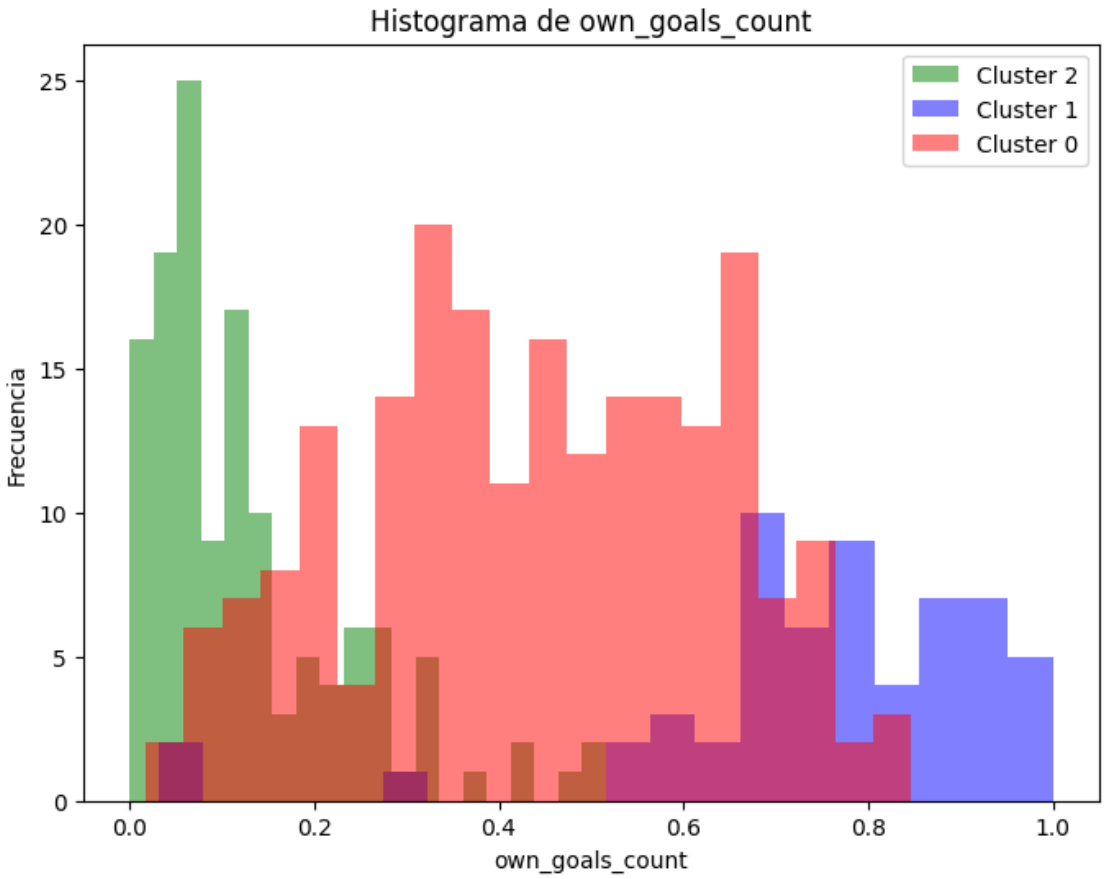
[426]:
```
%%time
grp = 'ward_3'
ls_tukey_ward=[]
for c_ in Xmm_sample.columns:
    #print(c_)
    tukey = pairwise_tukeyhsd(
        endog=Xmm_sample[c_],
        groups=Xmds_sample[grp],
        alpha=0.05
        )
    tukey_df = pd.DataFrame(data=tukey._results_table.data[1:], columns=tukey.
 ↪_results_table.data[0])
    if tukey_df['reject'].mean()==1:
        print(f'variable con todos los valores de reject verdadero es {c_}')
        ls_tukey_ward.append(c_)
```

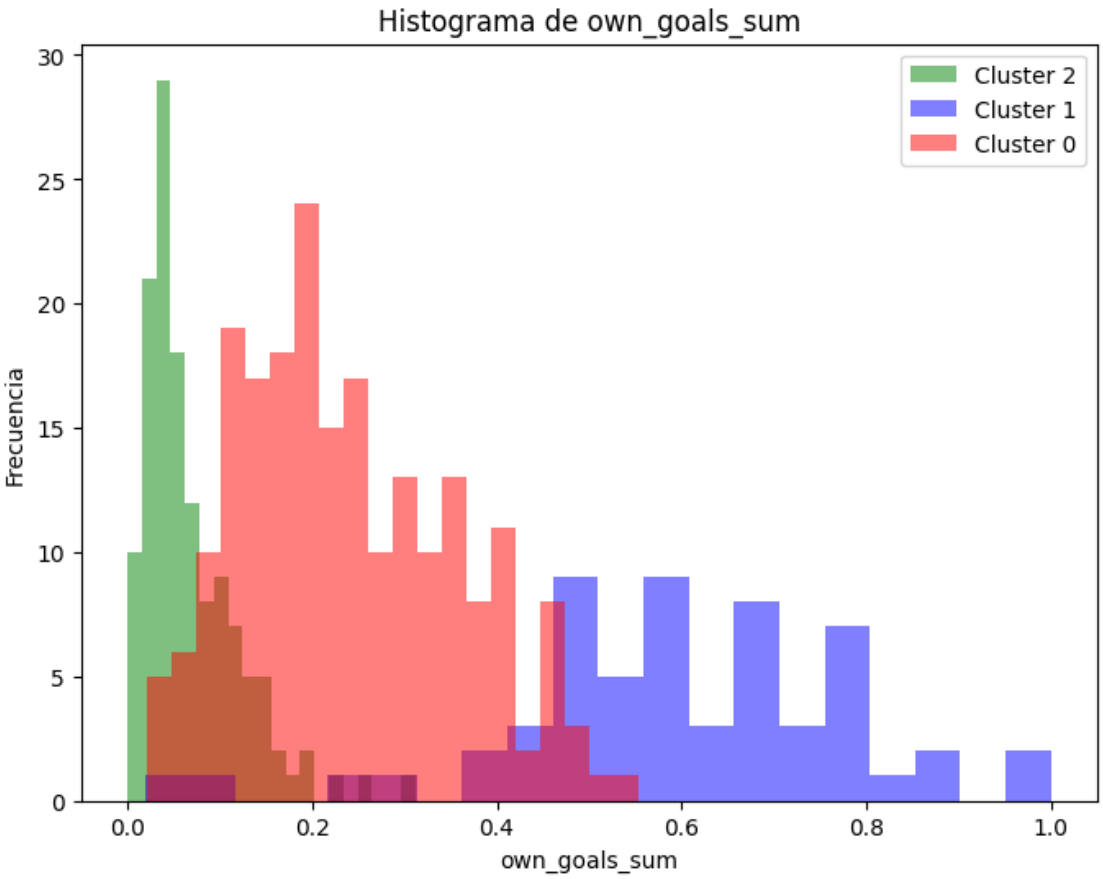
```
variable con todos los valores de reject verdadero es own_goals_count
```

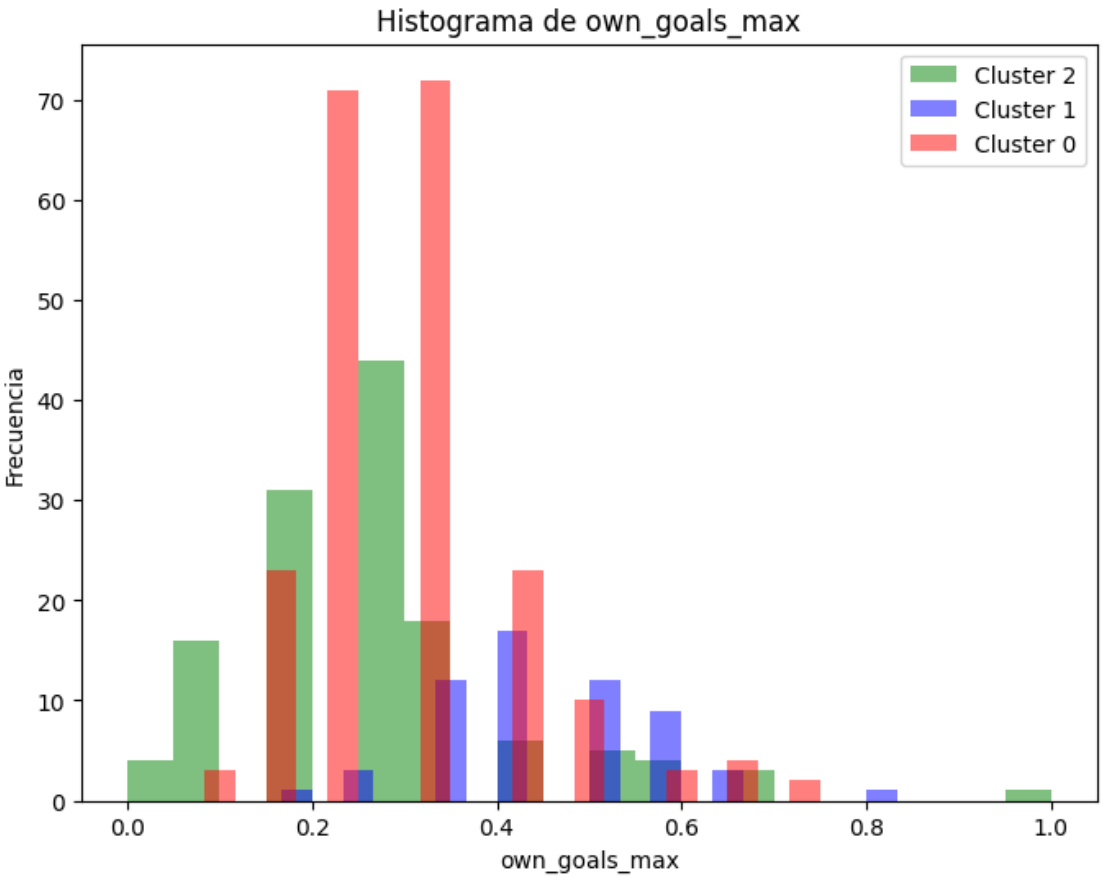
```
variable con todos los valores de reject verdadero es own_goals_sum
variable con todos los valores de reject verdadero es own_goals_max
variable con todos los valores de reject verdadero es own_goals_mean
variable con todos los valores de reject verdadero es own_goals_median
variable con todos los valores de reject verdadero es own_position_mean
variable con todos los valores de reject verdadero es own_position_median
variable con todos los valores de reject verdadero es is_win_mean
variable con todos los valores de reject verdadero es dif_goals_loc_sum
variable con todos los valores de reject verdadero es dif_goals_loc_max
variable con todos los valores de reject verdadero es dif_goals_loc_mean
variable con todos los valores de reject verdadero es dif_goals_loc_median
variable con todos los valores de reject verdadero es is_win_over_2_local_mean
variable con todos los valores de reject verdadero es is_win_over_3_local_mean
variable con todos los valores de reject verdadero es is_win_over_4_local_mean
variable con todos los valores de reject verdadero es is_lost_over_2_local_mean
variable con todos los valores de reject verdadero es is_lost_over_3_local_mean
variable con todos los valores de reject verdadero es opponent_goals_sum
variable con todos los valores de reject verdadero es opponent_goals_max
variable con todos los valores de reject verdadero es opponent_goals_mean
variable con todos los valores de reject verdadero es opponent_goals_median
variable con todos los valores de reject verdadero es opponent_position_mean
variable con todos los valores de reject verdadero es opponent_position_median
variable con todos los valores de reject verdadero es is_win_visit_mean
variable con todos los valores de reject verdadero es dif_goals_visit_sum
variable con todos los valores de reject verdadero es dif_goals_visit_max
variable con todos los valores de reject verdadero es dif_goals_visit_mean
variable con todos los valores de reject verdadero es dif_goals_visit_median
variable con todos los valores de reject verdadero es Total_goals_sum_y
variable con todos los valores de reject verdadero es Total_goals_min_y
variable con todos los valores de reject verdadero es Total_goals_mean_y
variable con todos los valores de reject verdadero es Total_goals_median_y
variable con todos los valores de reject verdadero es is_win_over_2_visit_mean
variable con todos los valores de reject verdadero es is_win_over_3_visit_mean
variable con todos los valores de reject verdadero es is_win_over_4_visit_mean
variable con todos los valores de reject verdadero es is_lost_over_2_visit_mean
variable con todos los valores de reject verdadero es is_lost_over_3_visit_mean
variable con todos los valores de reject verdadero es national_team_players
variable con todos los valores de reject verdadero es stadium_seats
CPU times: user 10.7 s, sys: 116 ms, total: 10.8 s
Wall time: 10.7 s
```

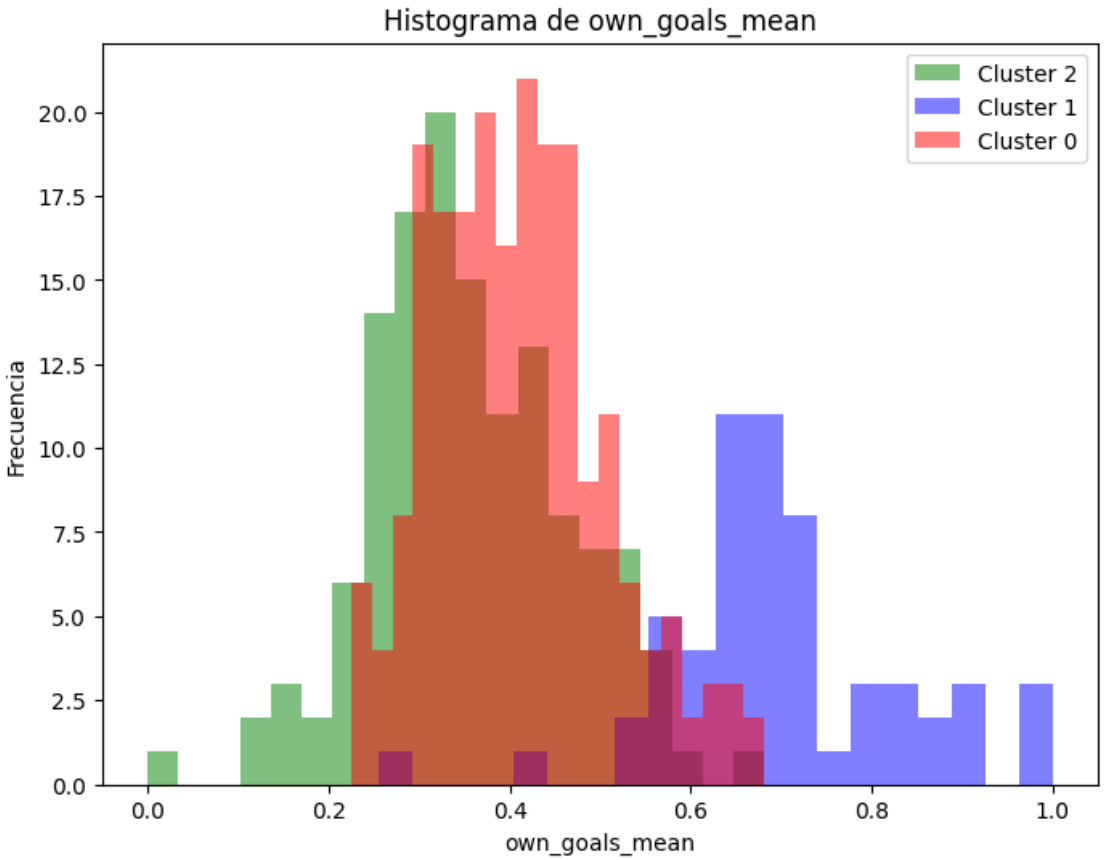
[427]:
```
unique_clusters = Xmds_sample['ward_3'].unique()

# Color mapping para kmeans_mds_3
color_mapping = {0: 'red', 1: 'blue', 2: 'green'}

# Itera a través de las variables en ls_best
```

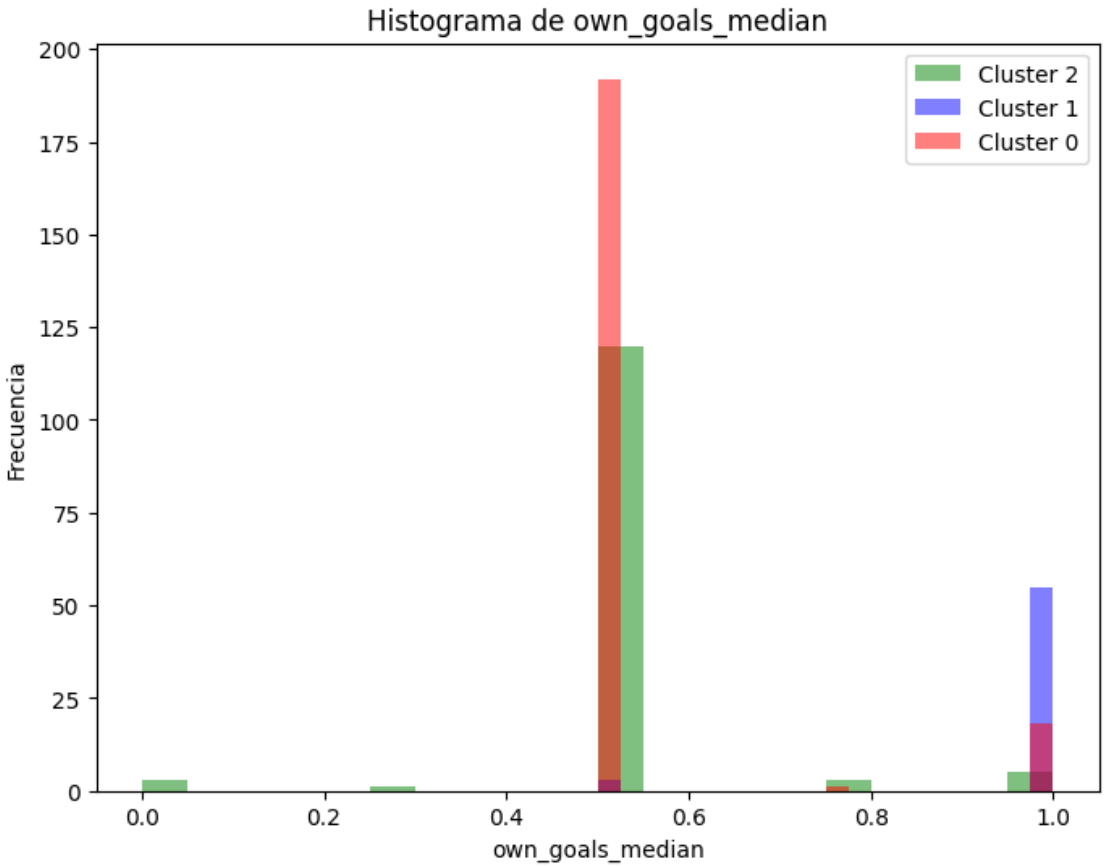
```
for variable in ls_tukey_ward:
    # Crear un nuevo histograma para la variable actual
    plt.figure(figsize=(8, 6))  # Establece el tamaño de la figura (opcional)

    # Itera a través de los valores únicos de kmeans_mds_3
    for cluster_value in unique_clusters:
        # Restablece el índice del DataFrame Xmm_sample antes de la selección
        subset_data = Xmm_sample.reset_index(drop=True)[Xmds_sample['cl_gmm']␣
 ↪== cluster_value][variable]

        # Crea el histograma utilizando solo un color para este conjunto de␣
 ↪datos
        plt.hist(subset_data, bins=20, color=color_mapping[cluster_value],␣
 ↪alpha=0.5, label=f'Cluster {cluster_value}')

    # Configura el título y etiquetas de los ejes
    plt.title(f'Histograma de {variable}')
    plt.xlabel(variable)
    plt.ylabel('Frecuencia')

    # Agrega una leyenda para identificar los clusters
    plt.legend()

    # Muestra la gráfica
    plt.show()
```

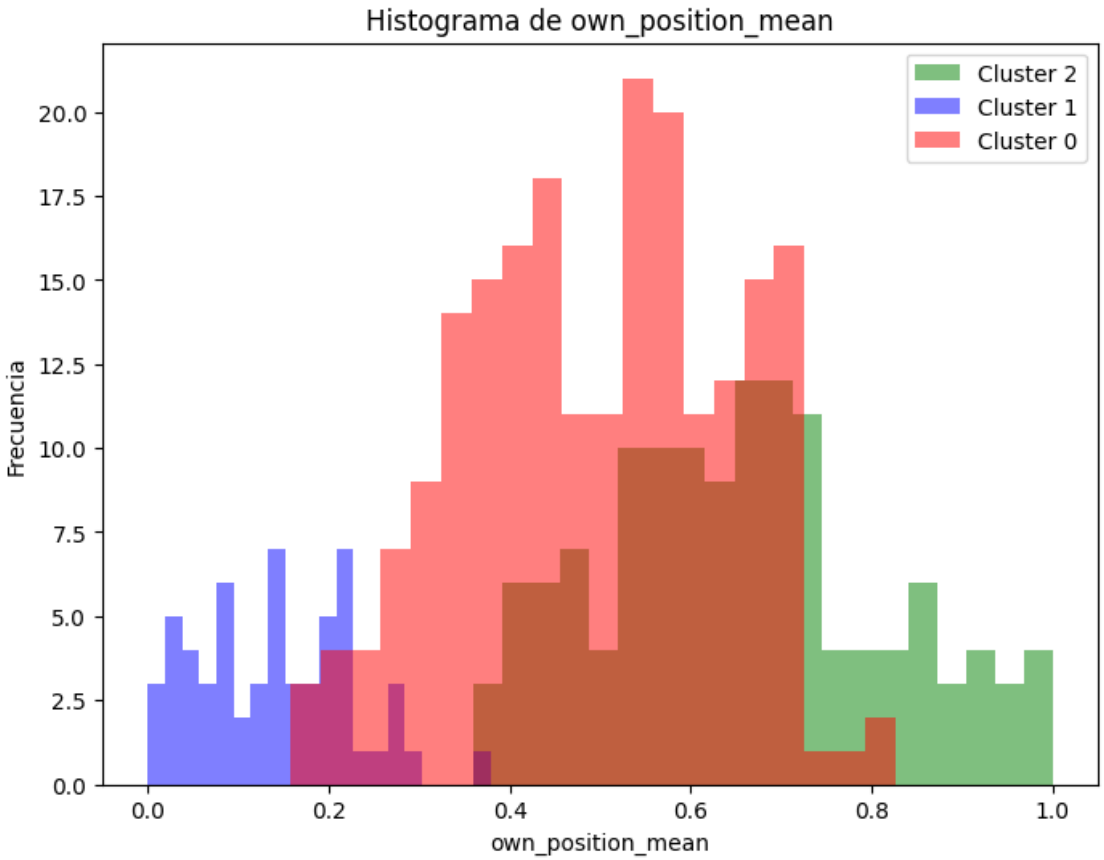

Histograma de own_goals_count
Cluster 2
Cluster 1
Cluster 0
Frecuencia
own_goals_count
0
5
10
15
20
25
0.0
0.2
0.4
0.6
0.8
1.0

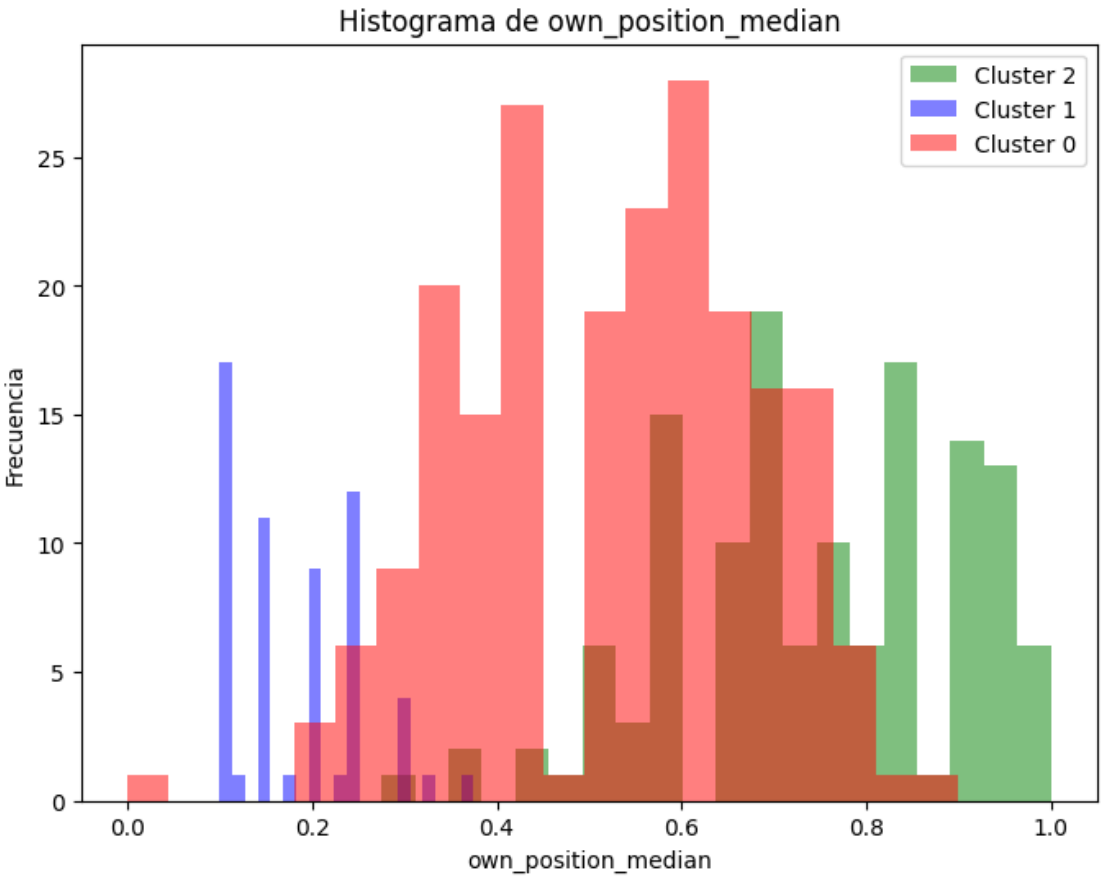

Histograma de own_goals_sum
Cluster 2
Cluster 1
Cluster 0
Frecuencia
own_goals_sum
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0

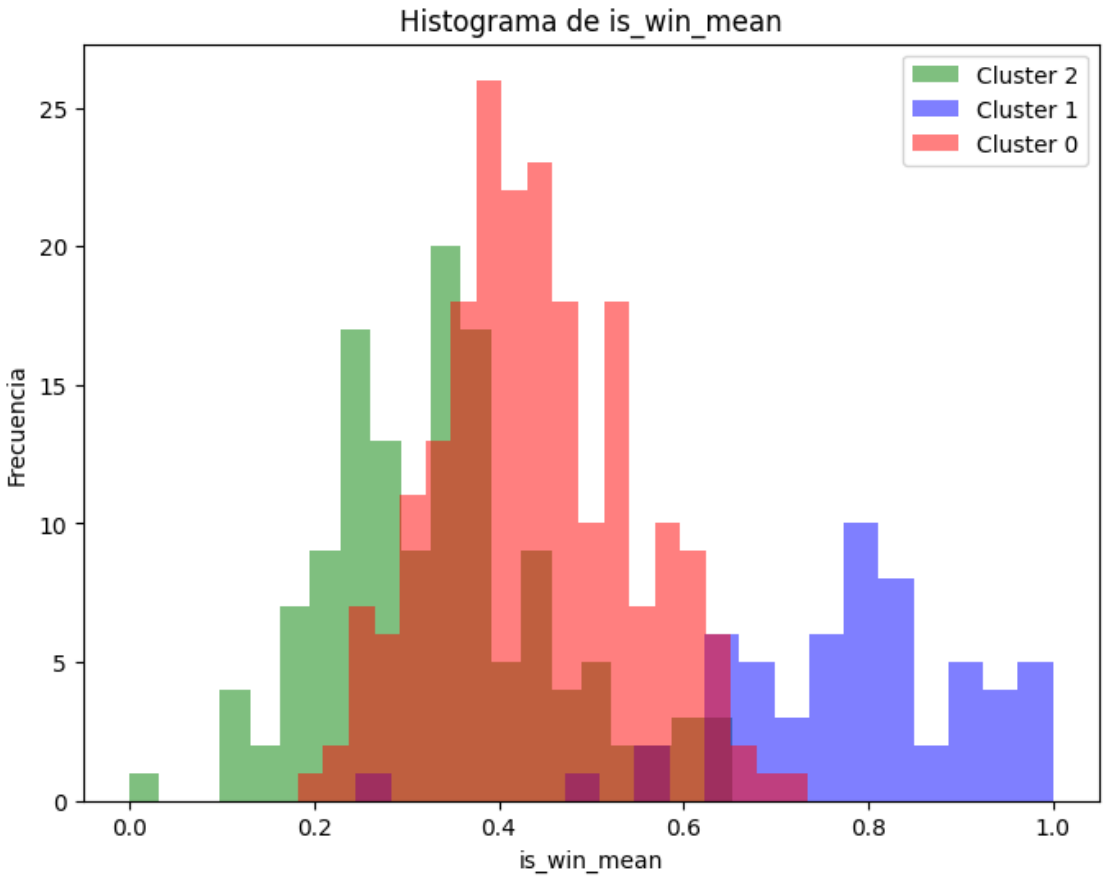

Histograma de own_goals_max
Cluster 2
Cluster 1
Cluster 0
Frecuencia
own_goals_max
0
10
20
30
40
50
60
70
0.0
0.2
0.4
0.6
0.8
1.0

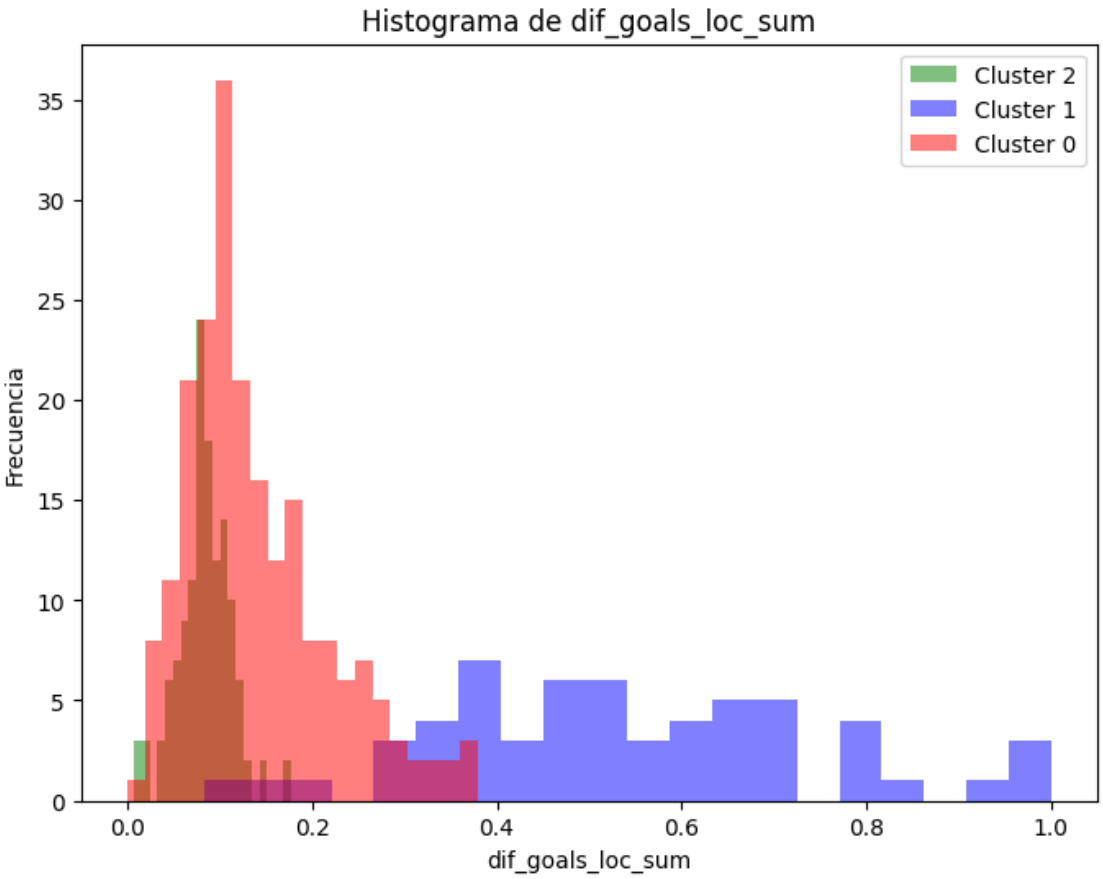

Histograma de own_goals_mean
Cluster 2
Cluster 1
Cluster 0
Frecuencia
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0
own_goals_mean

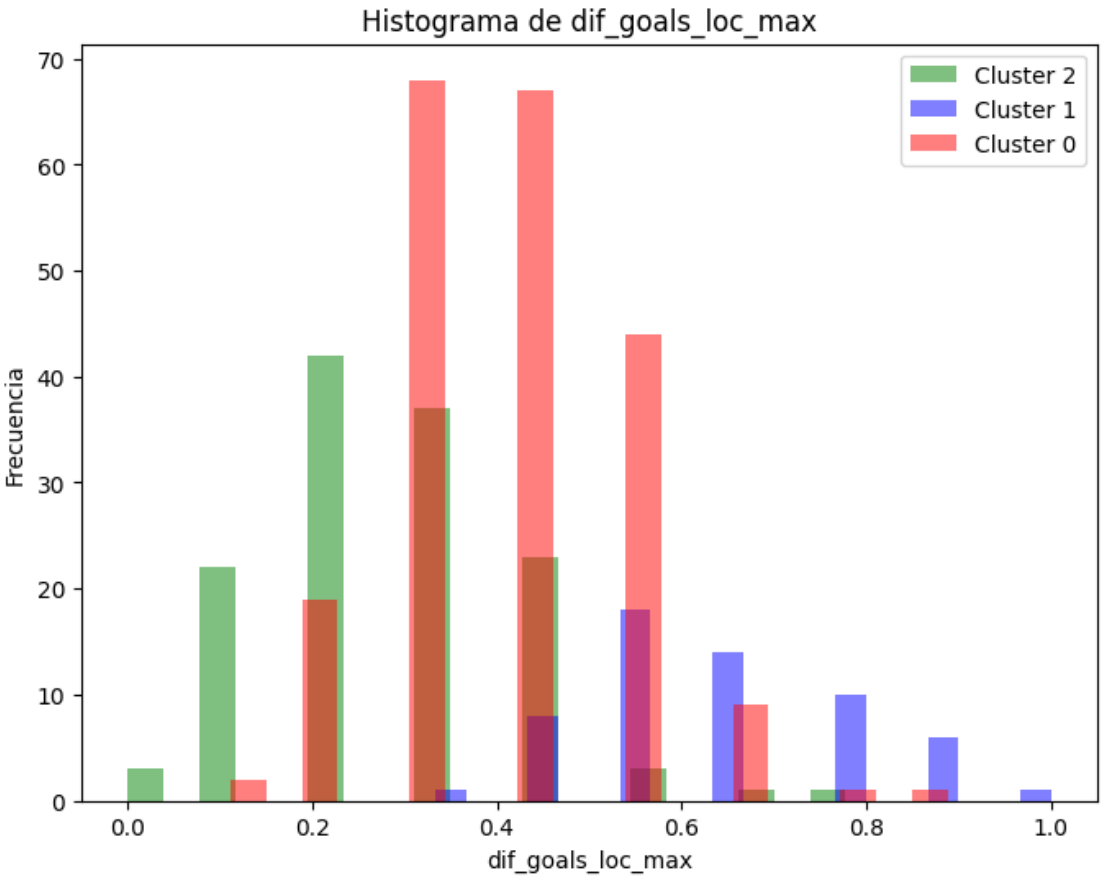

Histograma de own_goals_median
Cluster 2
Cluster 1
Cluster 0
Frecuencia
own_goals_median
200
175
150
125
100
75
50
25
0
0.0
0.2
0.4
0.6
0.8
1.0

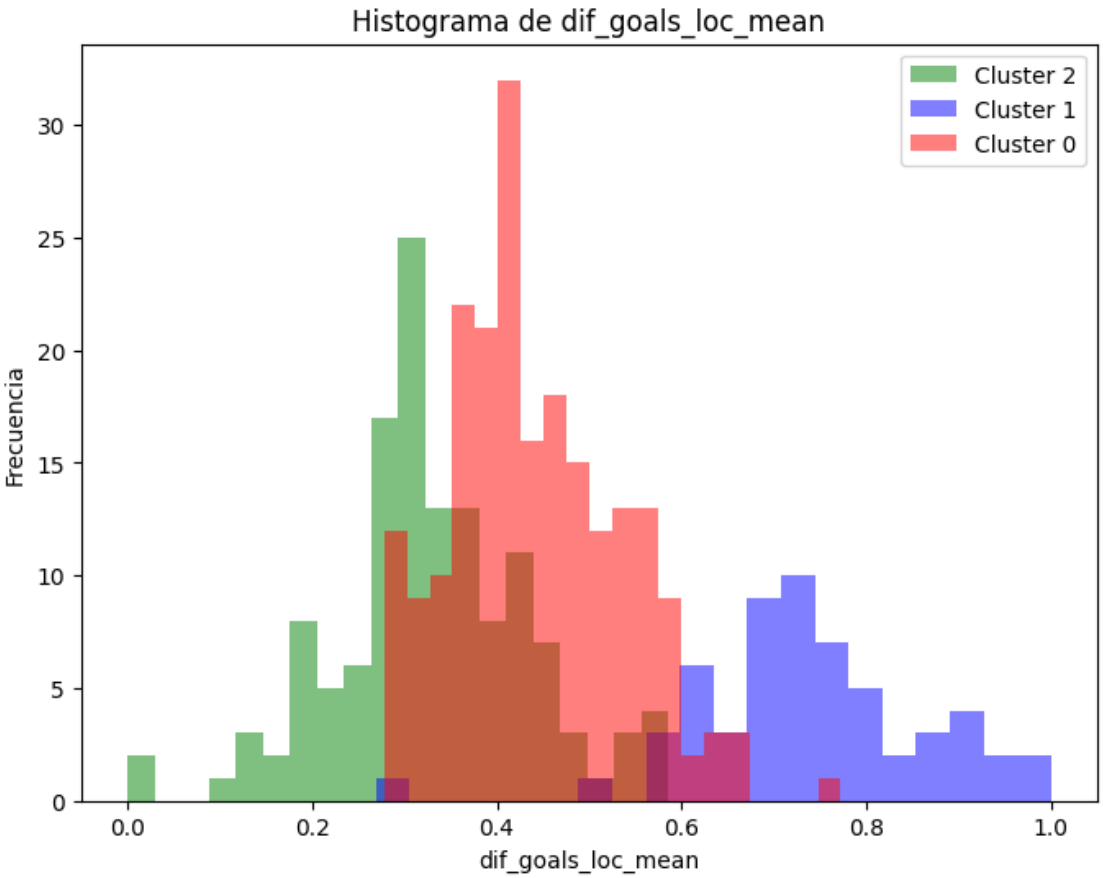

Histograma de own_position_mean
Cluster 2
Cluster 1
Cluster 0
Frecuencia
own_position_mean
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
0.0
0.2
0.4
0.6
0.8
1.0

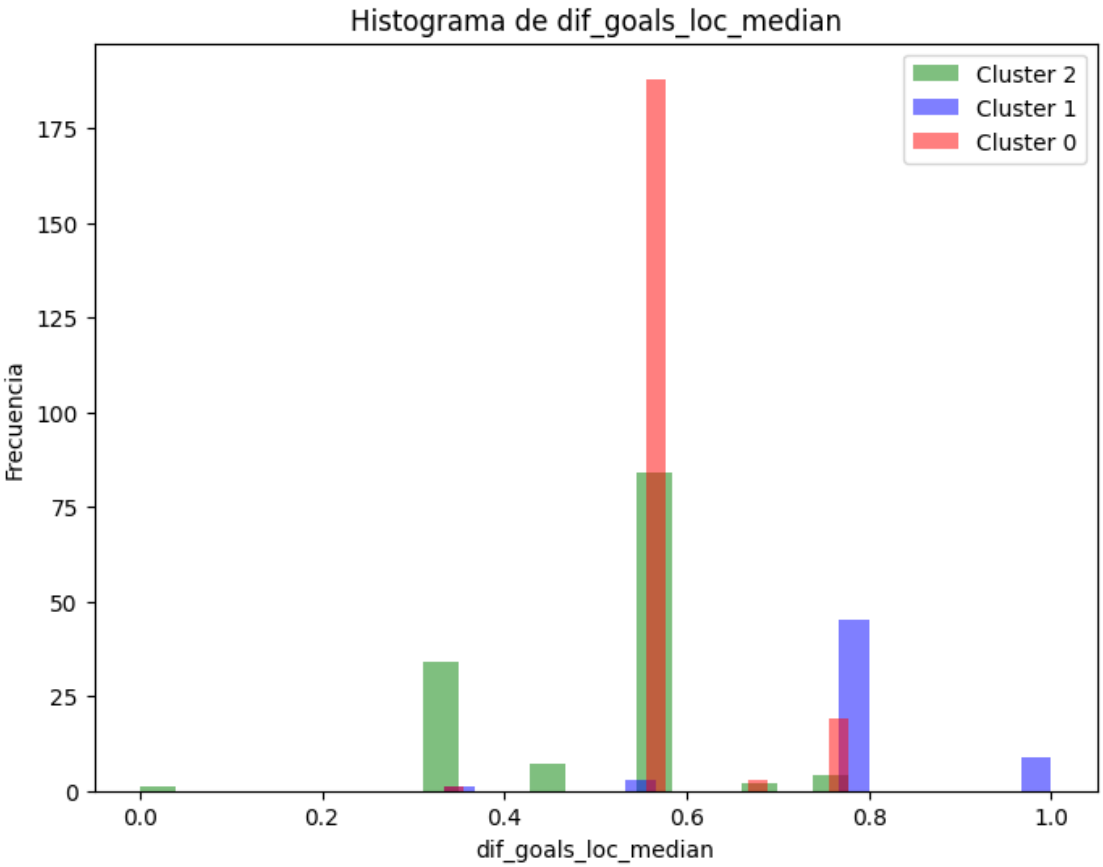

Histograma de own_position_median
Cluster 2
Cluster 1
Cluster 0
Frecuencia
0
5
10
15
20
25
0.0
0.2
0.4
0.6
0.8
1.0
own_position_median

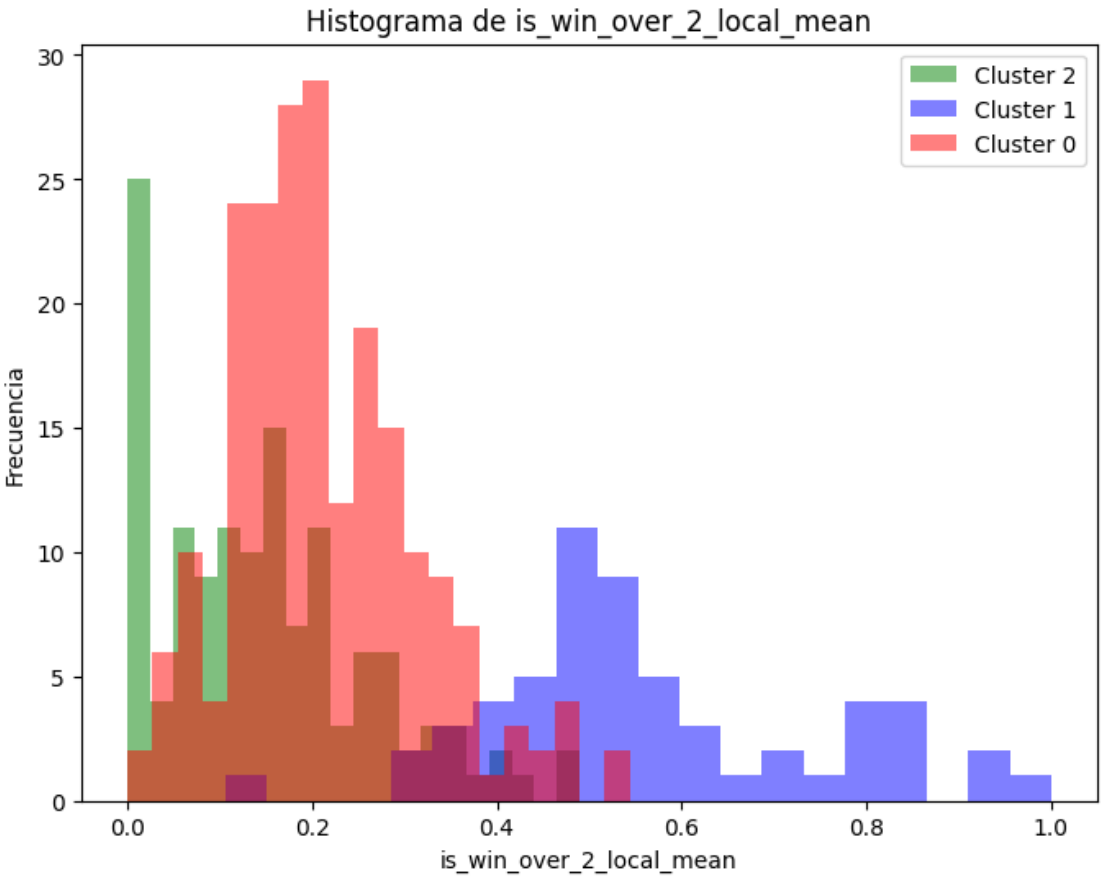

Histograma de is_win_mean
Cluster 2
Cluster 1
Cluster 0
Frecuencia
0
5
10
15
20
25
0.0
0.2
0.4
0.6
0.8
1.0
is_win_mean

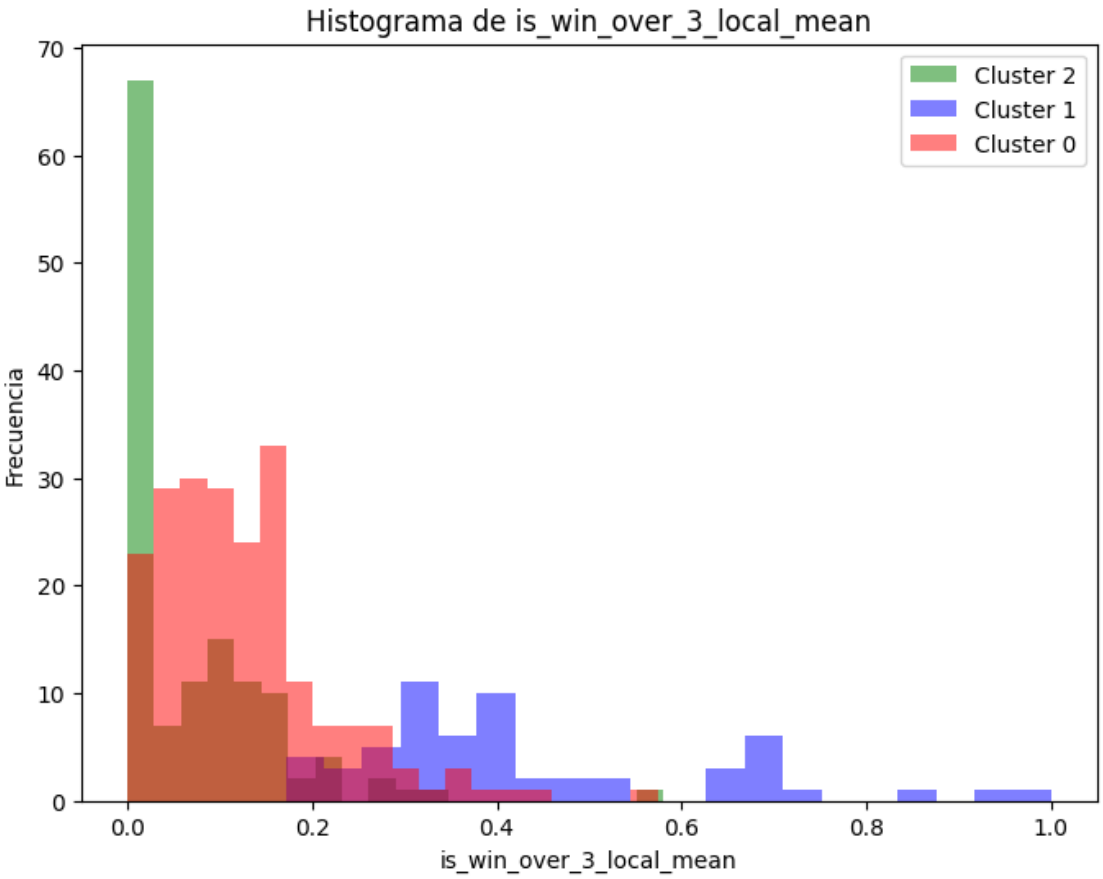

Histograma de dif_goals_loc_sum
Cluster 2
Cluster 1
Cluster 0
Frecuencia
dif_goals_loc_sum
0
5
10
15
20
25
30
35
0.0
0.2
0.4
0.6
0.8
1.0

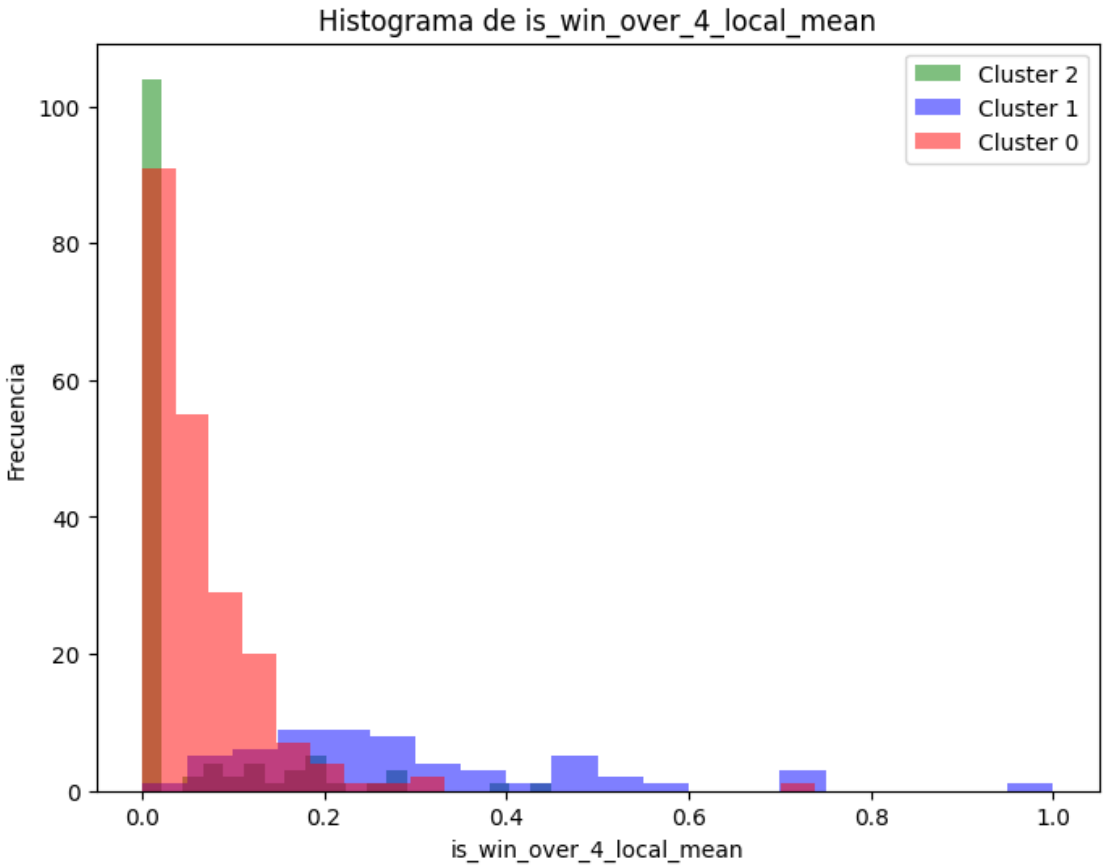

Histograma de dif_goals_loc_max
Cluster 2
Cluster 1
Cluster 0
Frecuencia
dif_goals_loc_max
0
10
20
30
40
50
60
70
0.0
0.2
0.4
0.6
0.8
1.0

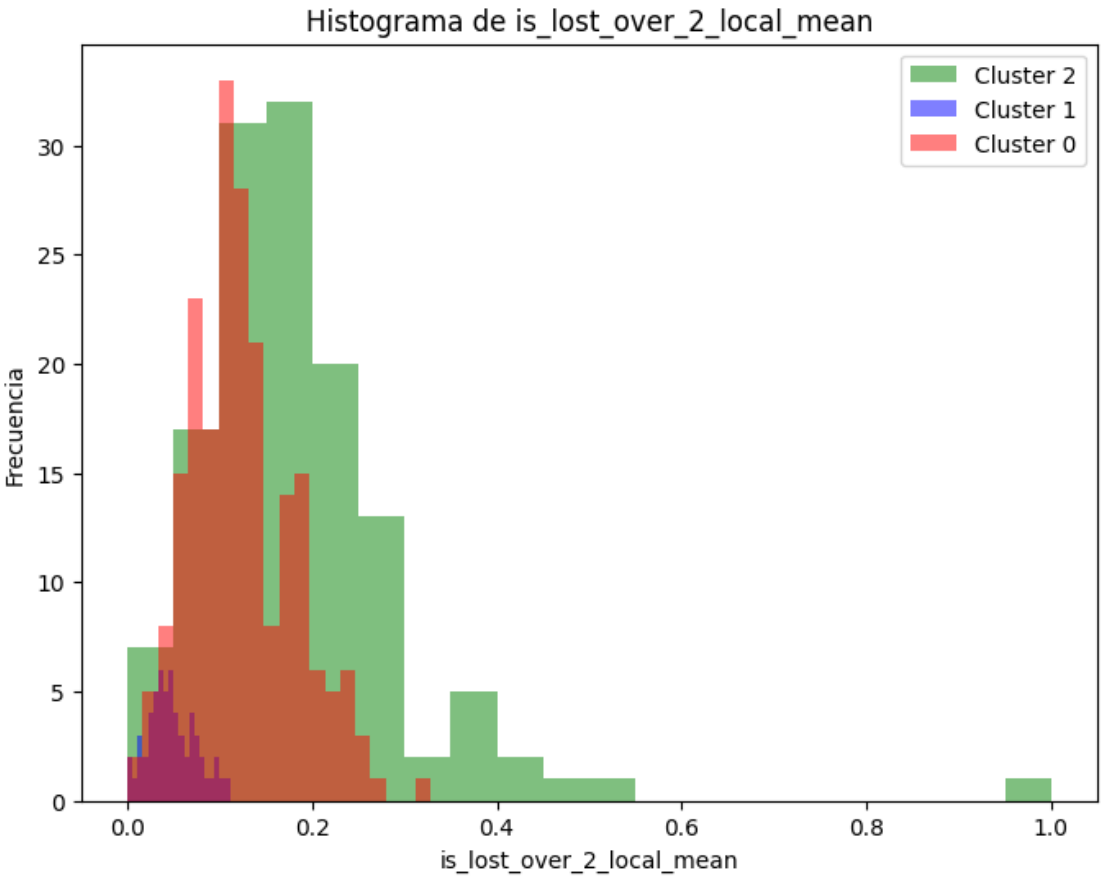

Histograma de dif_goals_loc_mean
Cluster 2
Cluster 1
Cluster 0
Frecuencia
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0
dif_goals_loc_mean

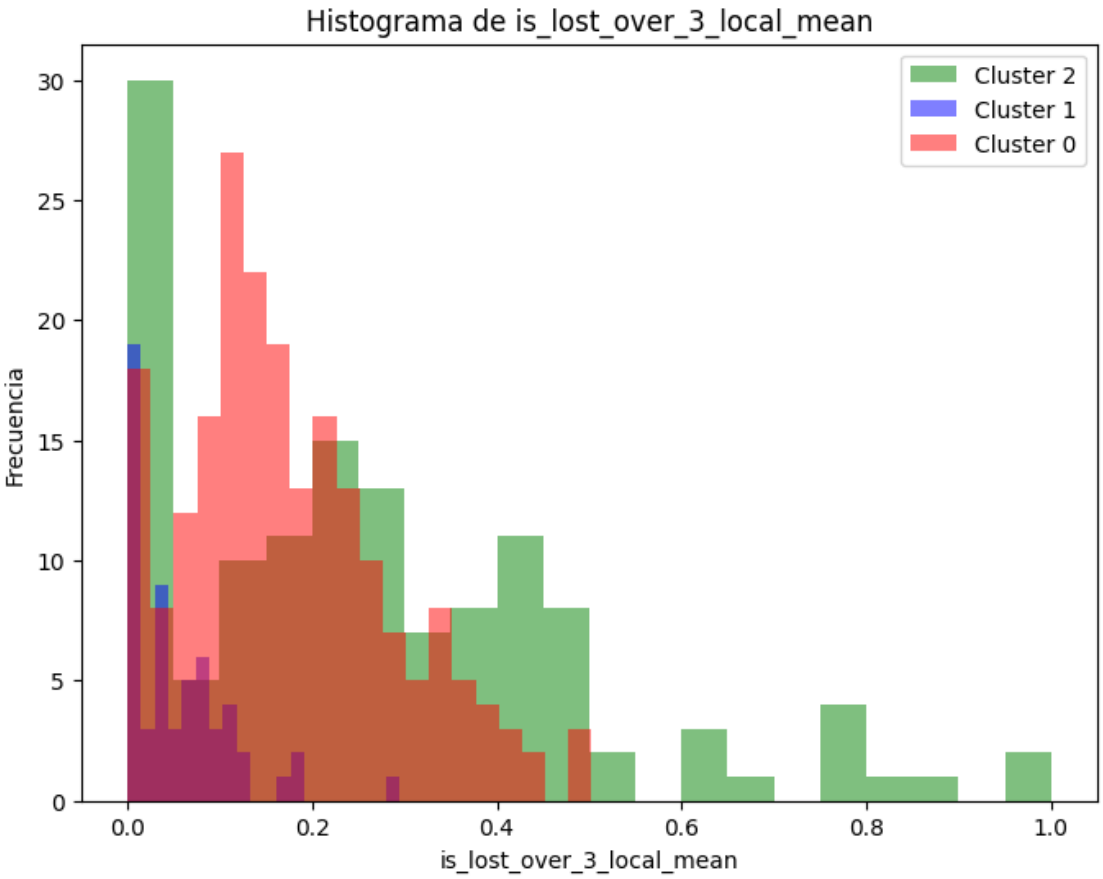

Histograma de dif_goals_loc_median
Cluster 2
Cluster 1
Cluster 0
Frecuencia
175
150
125
100
75
50
25
0
0.0
0.2
0.4
0.6
0.8
1.0
dif_goals_loc_median

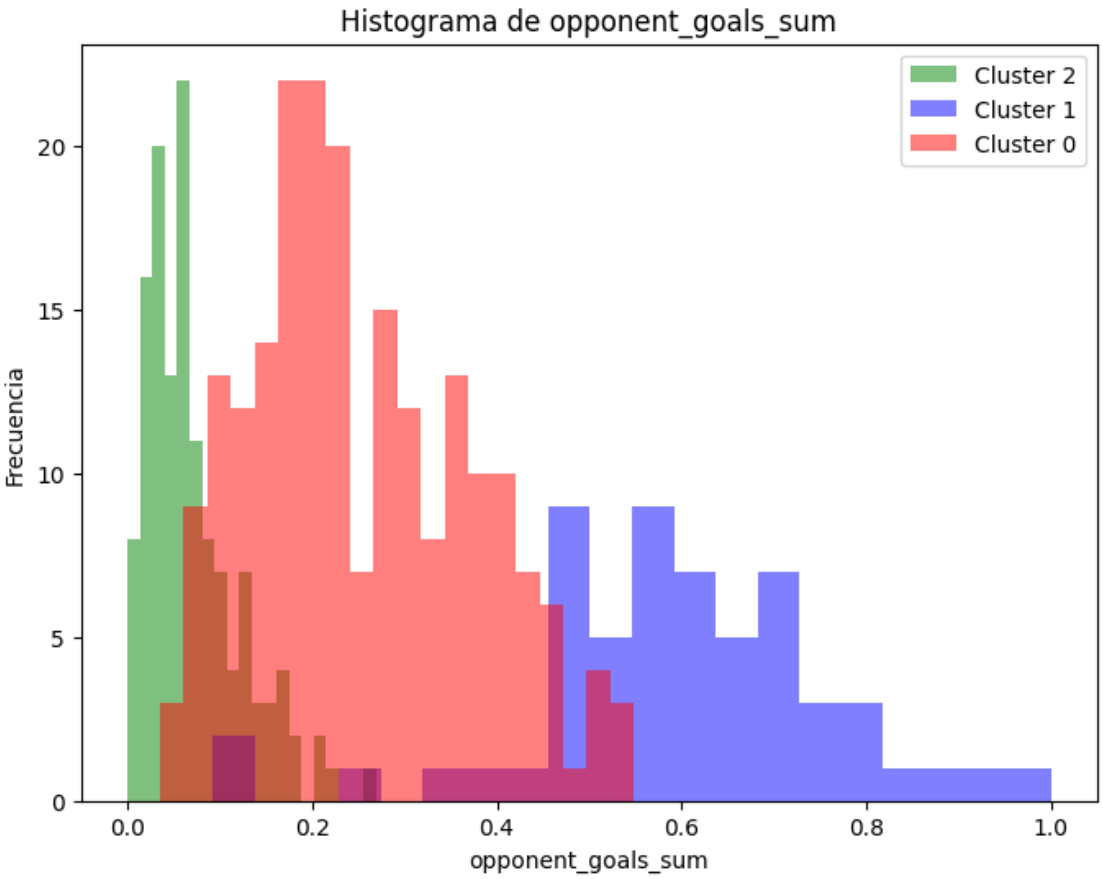

Histograma de is_win_over_2_local_mean
Cluster 2
Cluster 1
Cluster 0
Frecuencia
is_win_over_2_local_mean
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0

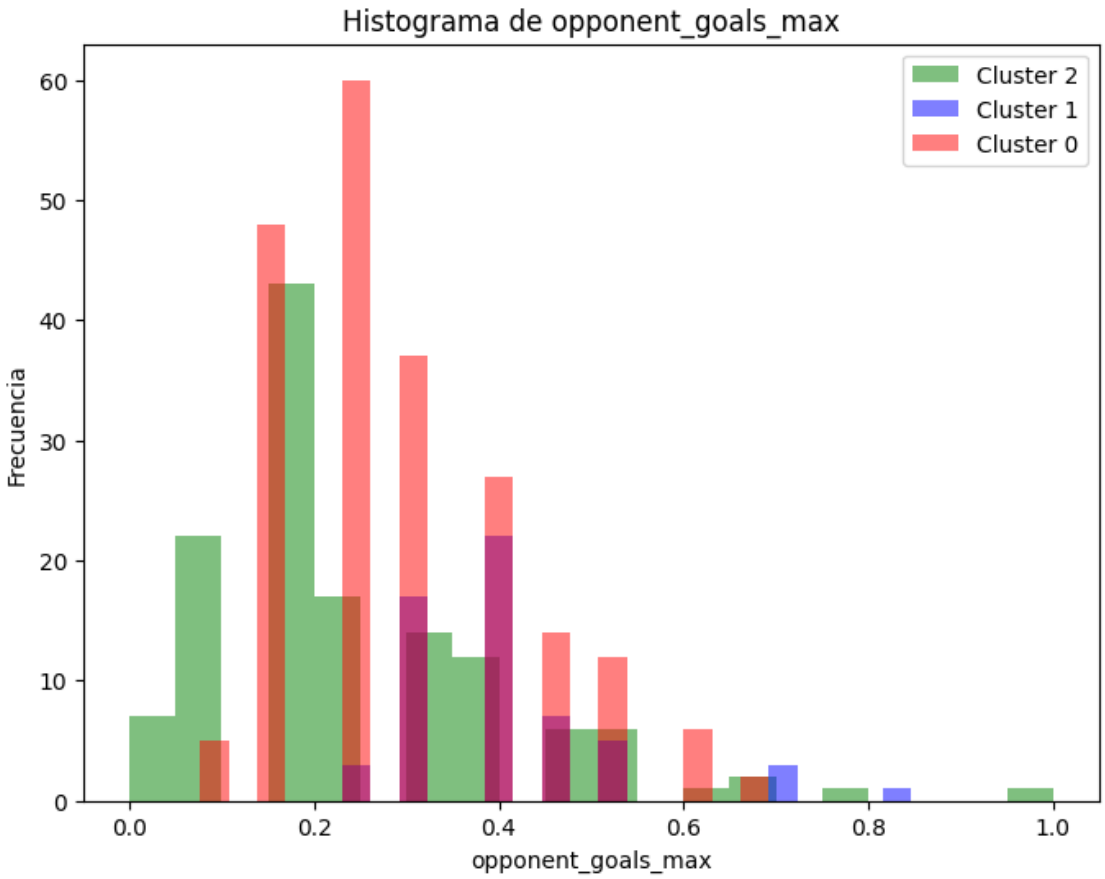

Histograma de is_win_over_3_local_mean
Cluster 2
Cluster 1
Cluster 0
Frecuencia
is_win_over_3_local_mean
0
10
20
30
40
50
60
70
0.0
0.2
0.4
0.6
0.8
1.0

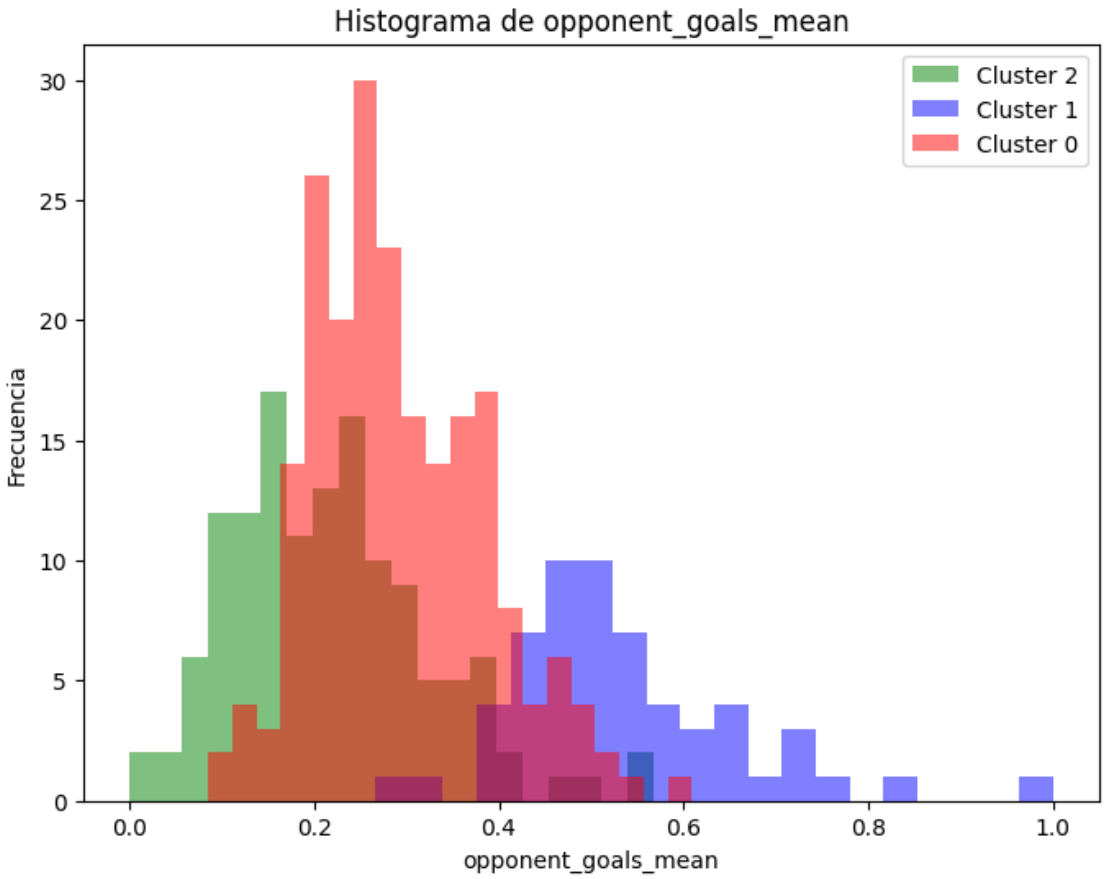

Histograma de is_win_over_4_local_mean
Cluster 2
Cluster 1
Cluster 0
Frecuencia
100
80
60
40
20
0
0.0
0.2
0.4
0.6
0.8
1.0
is_win_over_4_local_mean

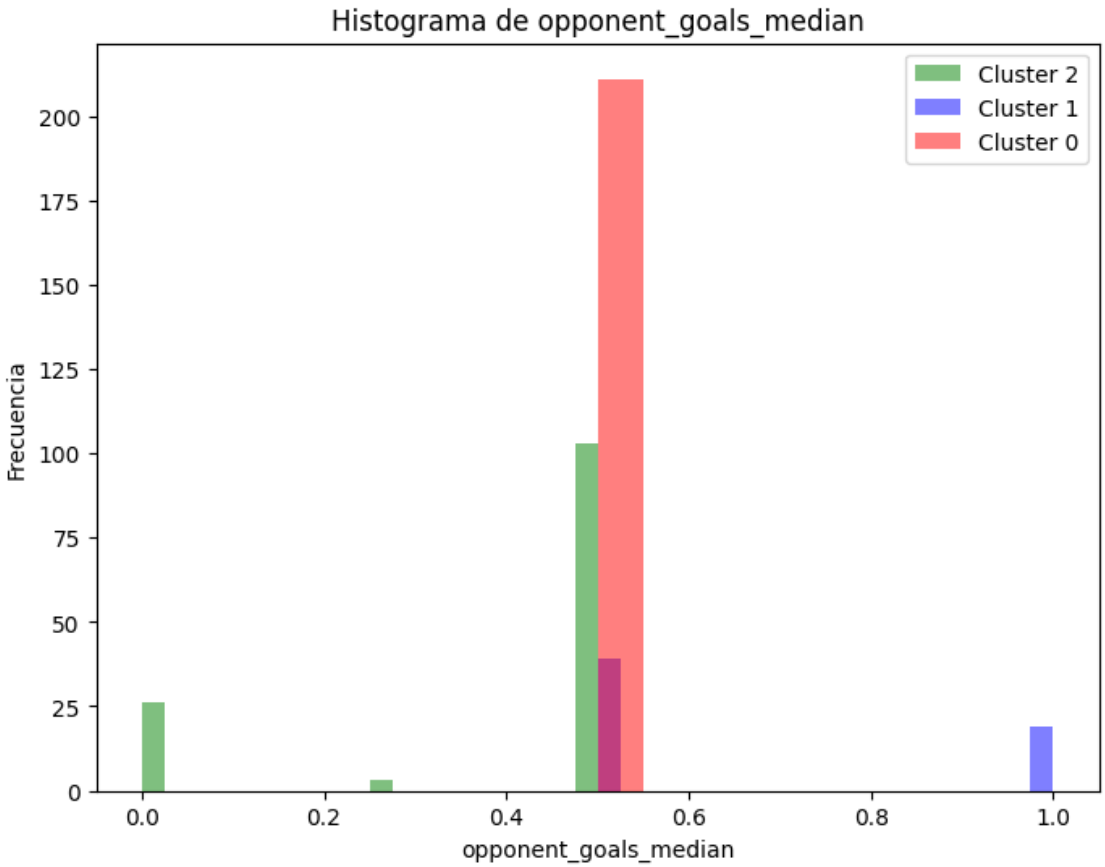

Histograma de is_lost_over_2_local_mean
Cluster 2
Cluster 1
Cluster 0
Frecuencia
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0
is_lost_over_2_local_mean

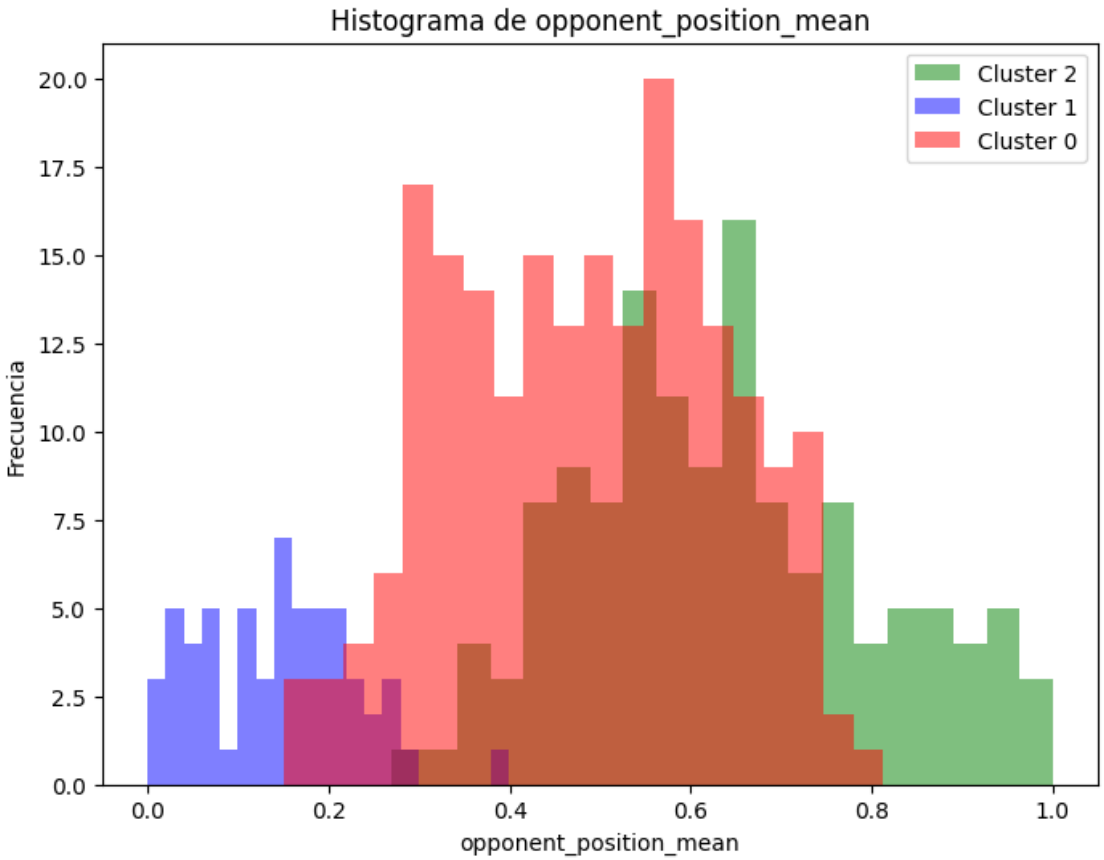

Histograma de is_lost_over_3_local_mean
Cluster 2
Cluster 1
Cluster 0
Frecuencia
is_lost_over_3_local_mean
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0

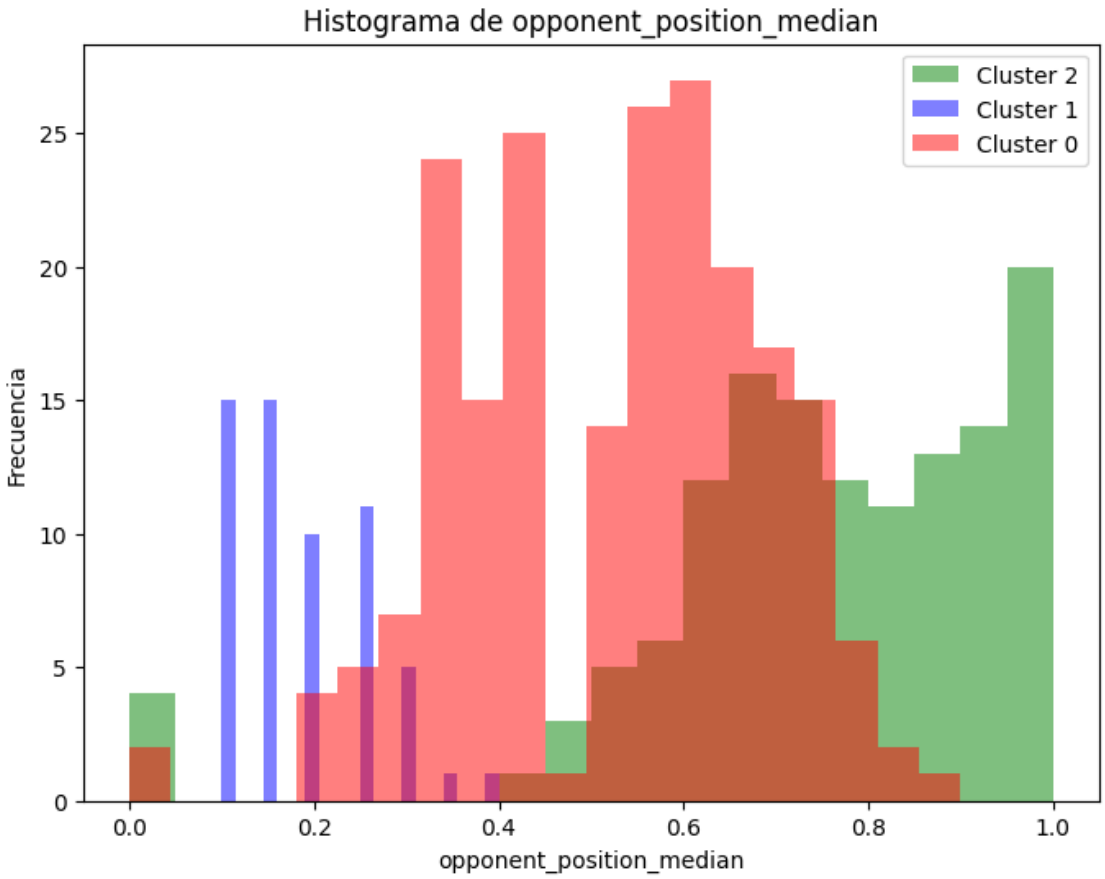

Histograma de opponent_goals_sum
Cluster 2
Cluster 1
Cluster 0
Frecuencia
0
5
10
15
20
0.0
0.2
0.4
0.6
0.8
1.0
opponent_goals_sum

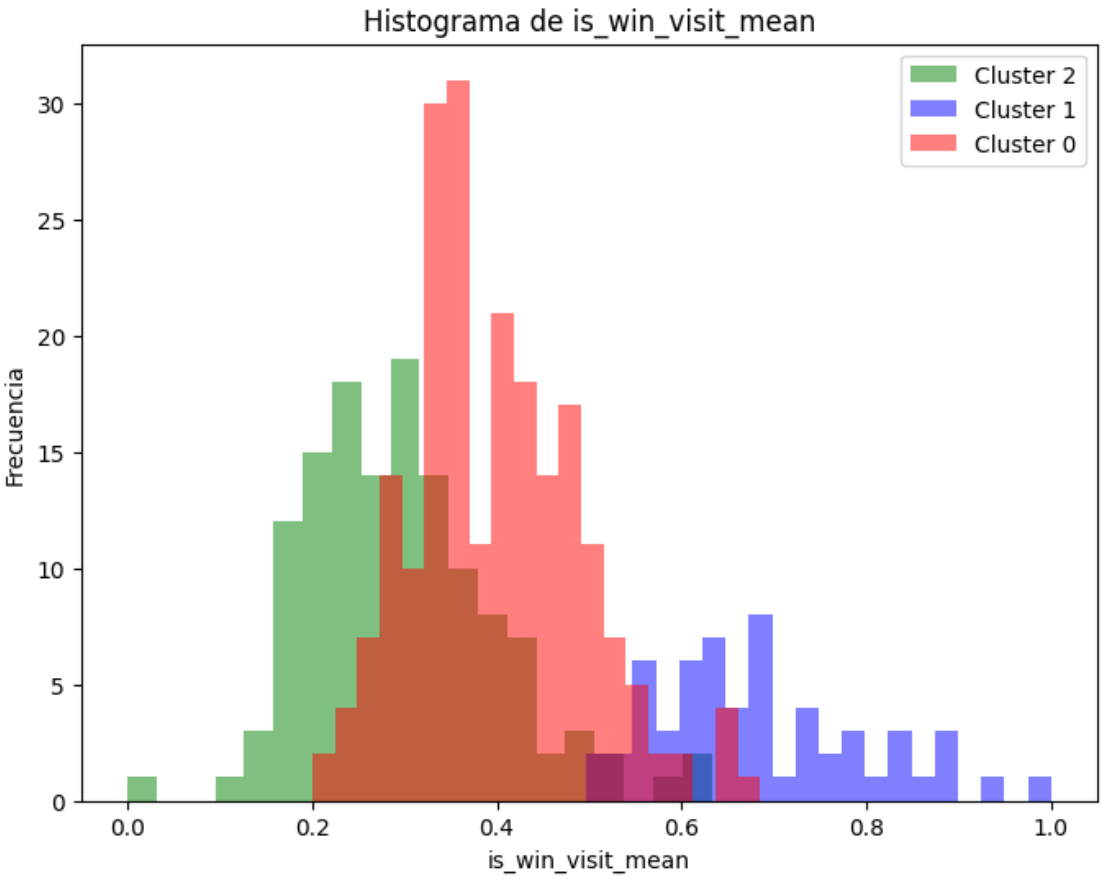

Histograma de opponent_goals_max
Cluster 2
Cluster 1
Cluster 0
Frecuencia
opponent_goals_max
0
10
20
30
40
50
60
0.0
0.2
0.4
0.6
0.8
1.0

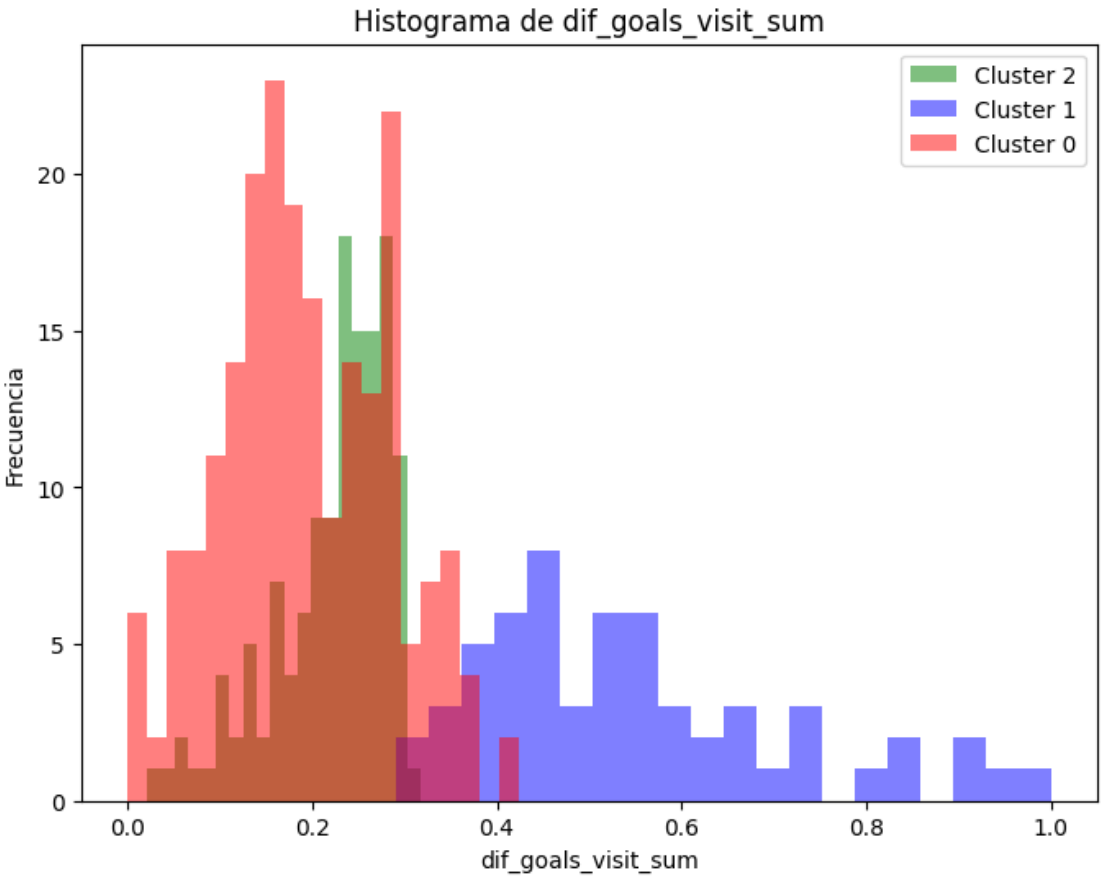

Histograma de opponent_goals_mean
Cluster 2
Cluster 1
Cluster 0
Frecuencia
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0
opponent_goals_mean

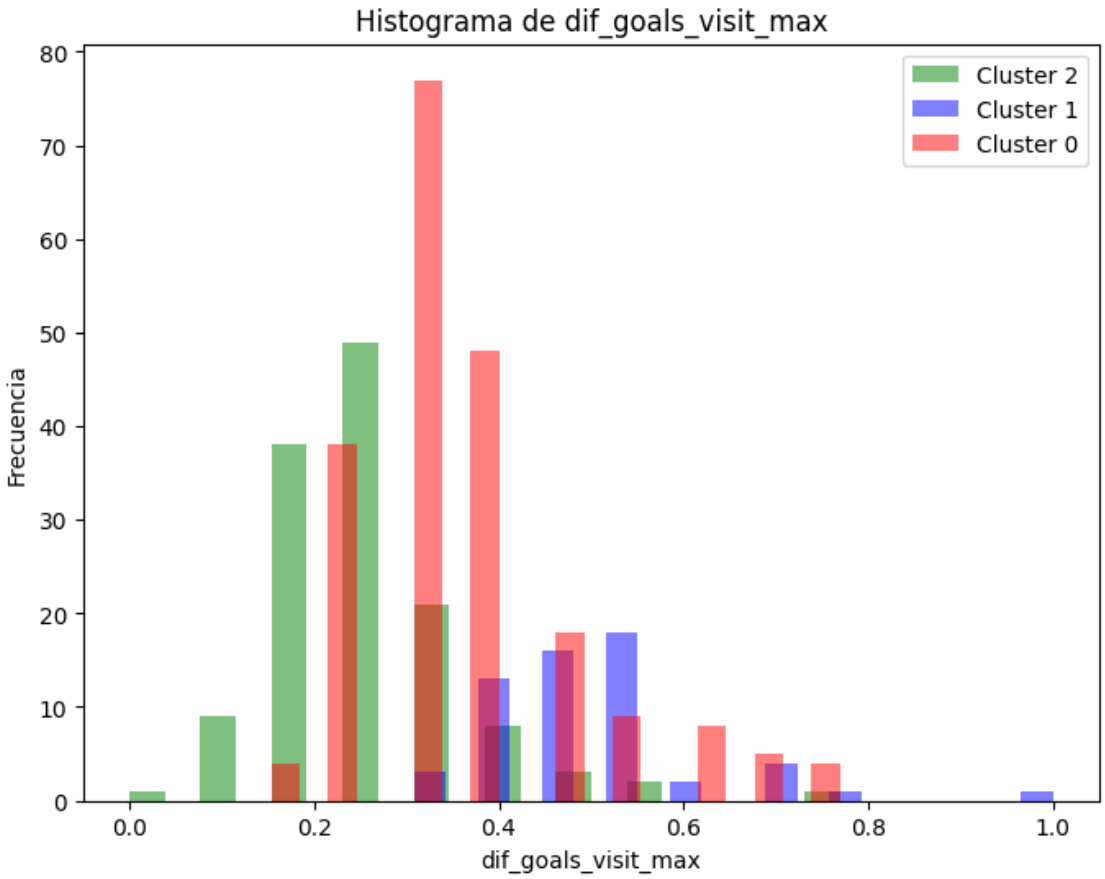

Histograma de opponent_goals_median
Cluster 2
Cluster 1
Cluster 0
Frecuencia
200
175
150
125
100
75
50
25
0
0.0
0.2
0.4
0.6
0.8
1.0
opponent_goals_median

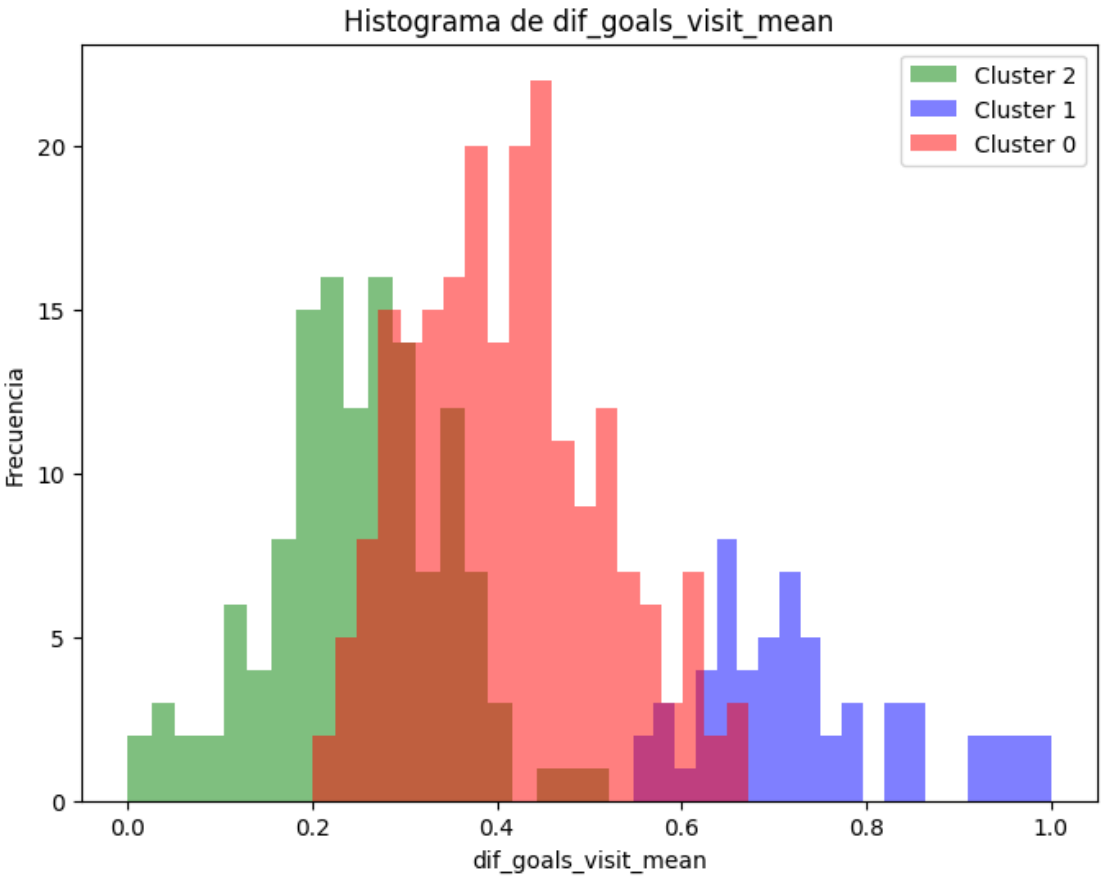

Histograma de opponent_position_mean
Cluster 2
Cluster 1
Cluster 0
Frecuencia
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
0.0
0.2
0.4
0.6
0.8
1.0
opponent_position_mean

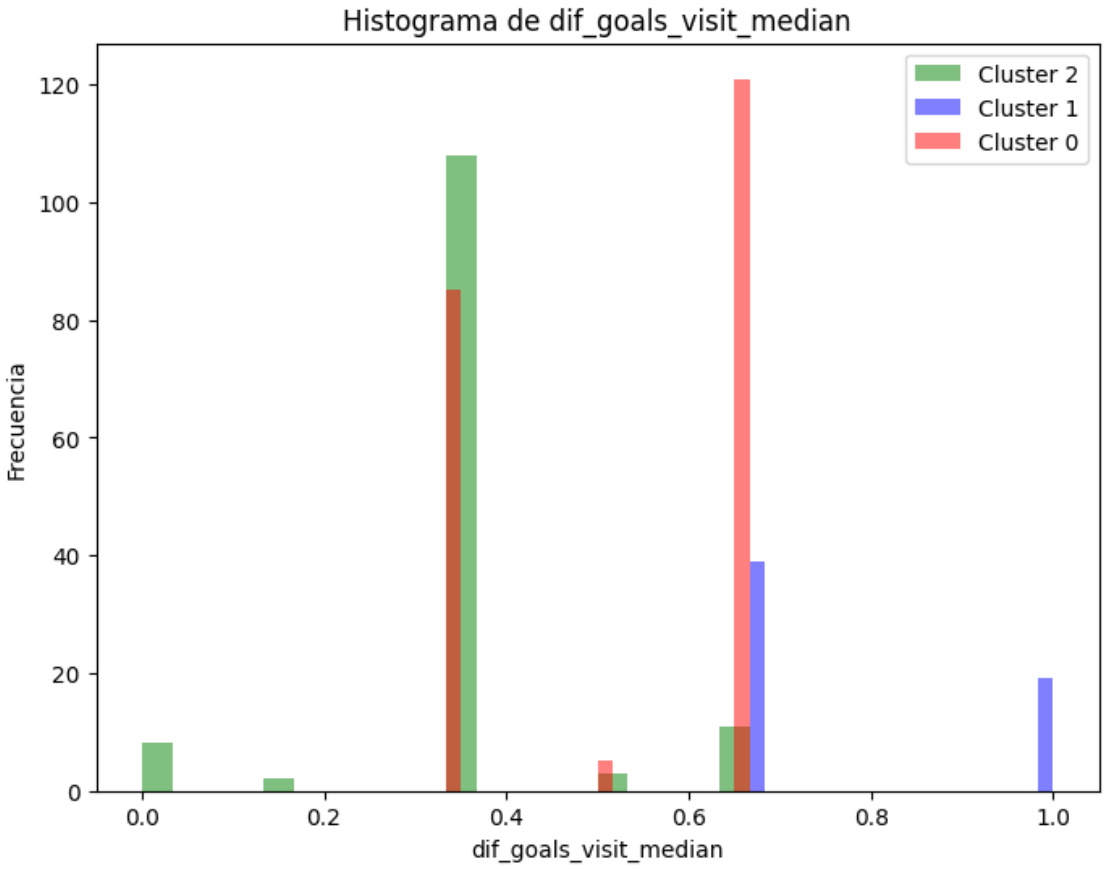

Histograma de opponent_position_median
Cluster 2
Cluster 1
Cluster 0
Frecuencia
0
5
10
15
20
25
0.0
0.2
0.4
0.6
0.8
1.0
opponent_position_median

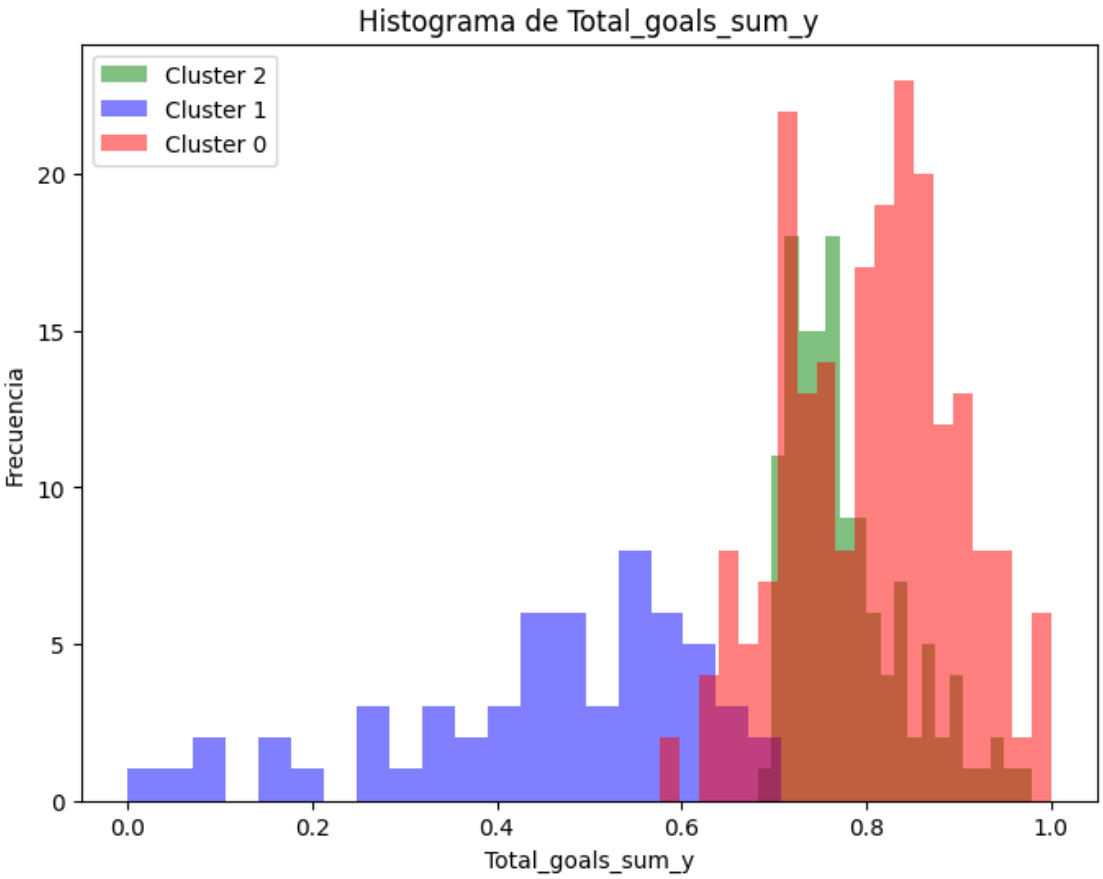

Histograma de is_win_visit_mean
Cluster 2
Cluster 1
Cluster 0
Frecuencia
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0
is_win_visit_mean

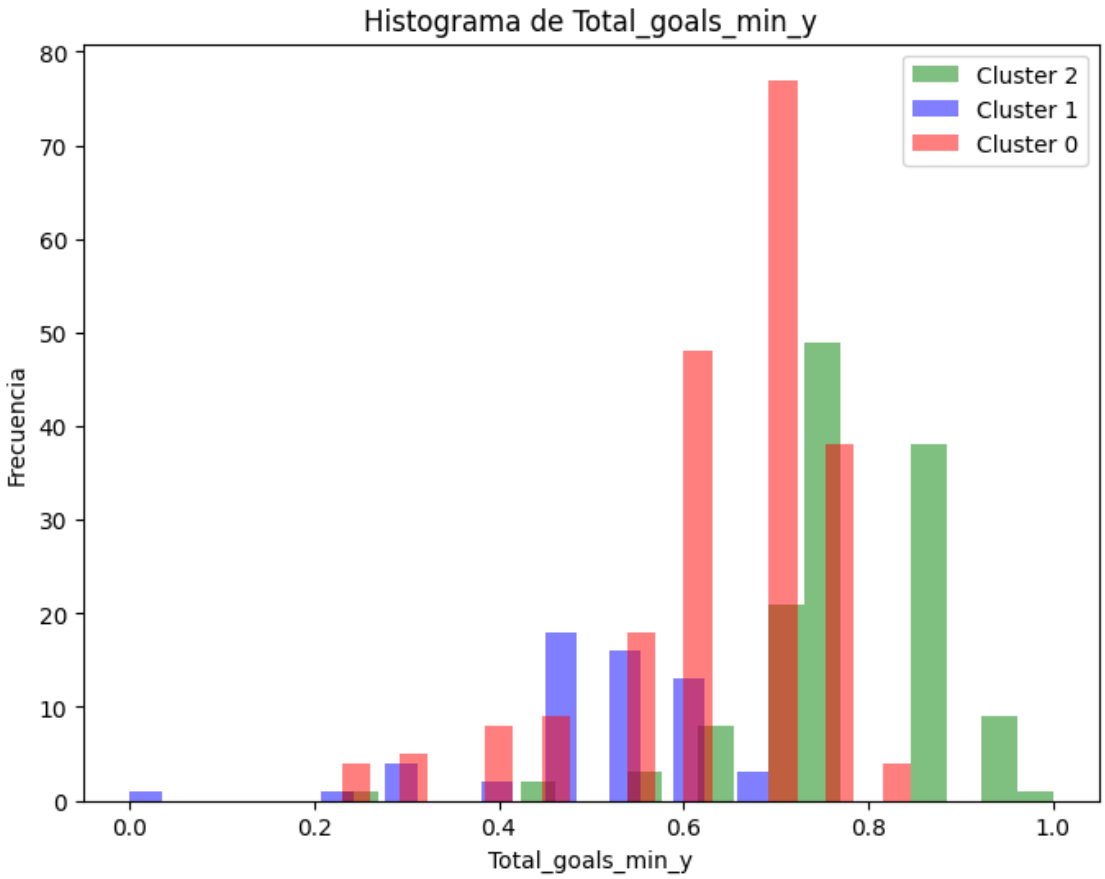

Histograma de dif_goals_visit_sum
Cluster 2
Cluster 1
Cluster 0
Frecuencia
0
5
10
15
20
0.0
0.2
0.4
0.6
0.8
1.0
dif_goals_visit_sum

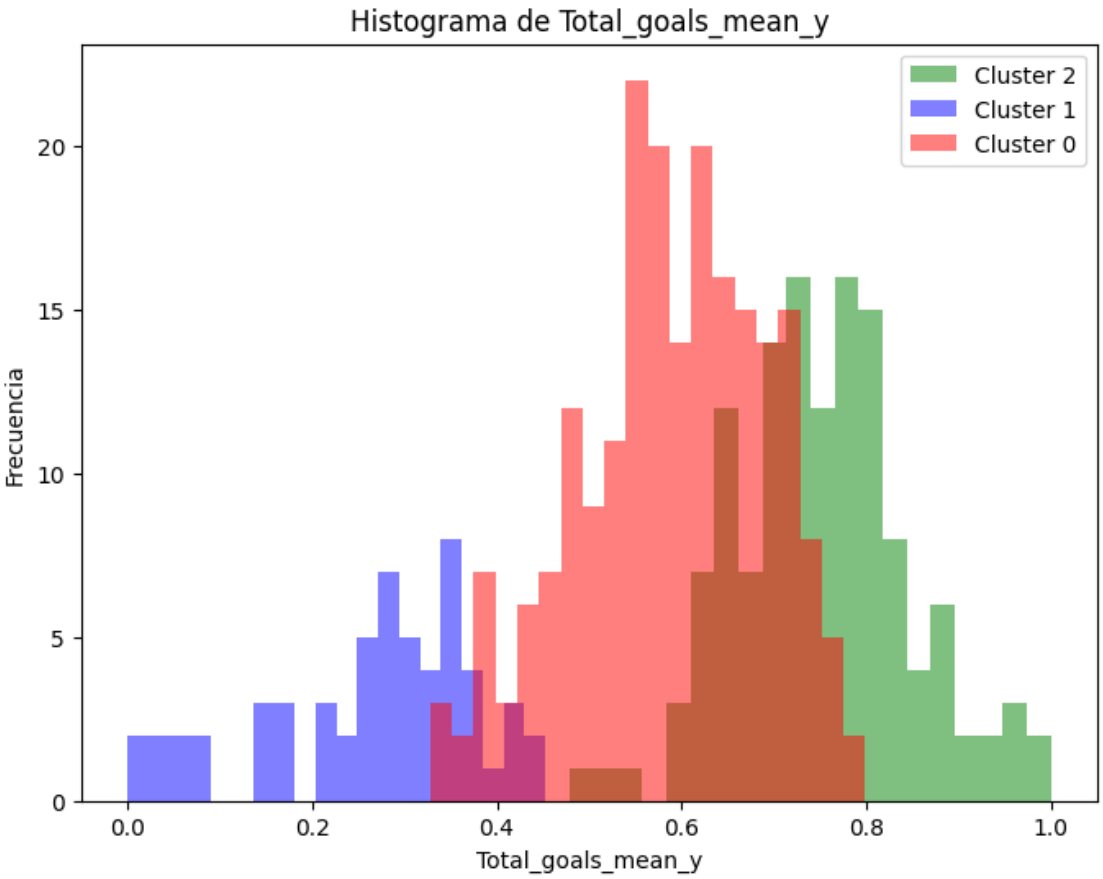

Histograma de dif_goals_visit_max
Cluster 2
Cluster 1
Cluster 0
Frecuencia
dif_goals_visit_max
0
10
20
30
40
50
60
70
80
0.0
0.2
0.4
0.6
0.8
1.0

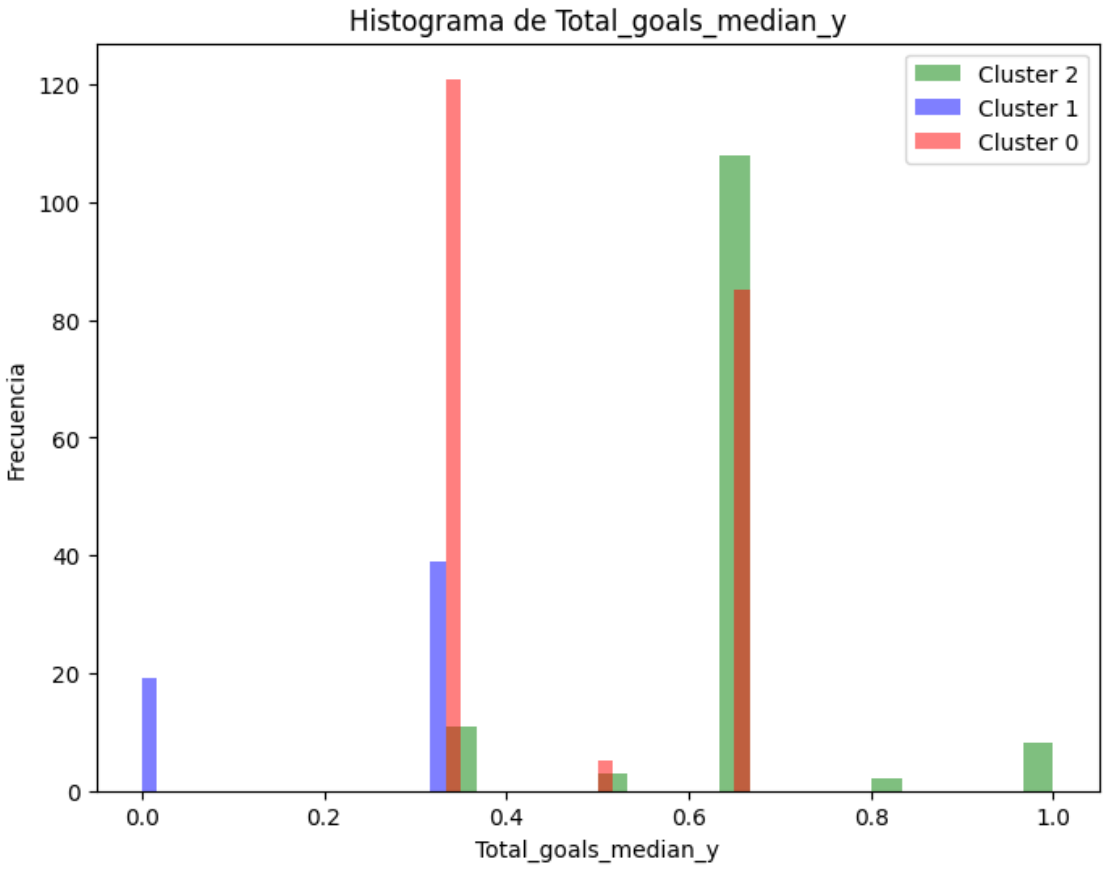

Histograma de dif_goals_visit_mean
Cluster 2
Cluster 1
Cluster 0
Frecuencia
0
5
10
15
20
0.0
0.2
0.4
0.6
0.8
1.0
dif_goals_visit_mean

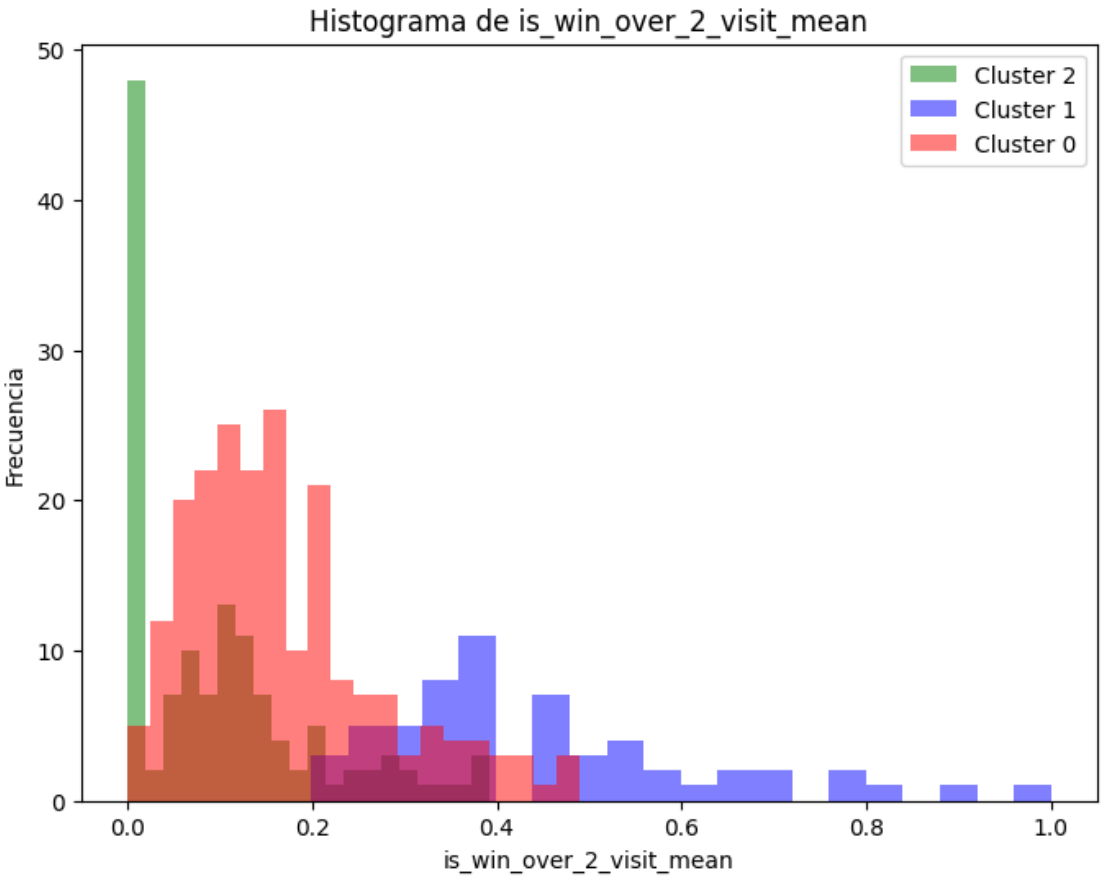

Histograma de dif_goals_visit_median
Cluster 2
Cluster 1
Cluster 0
Frecuencia
dif_goals_visit_median
0
20
40
60
80
100
120
0.0
0.2
0.4
0.6
0.8
1.0

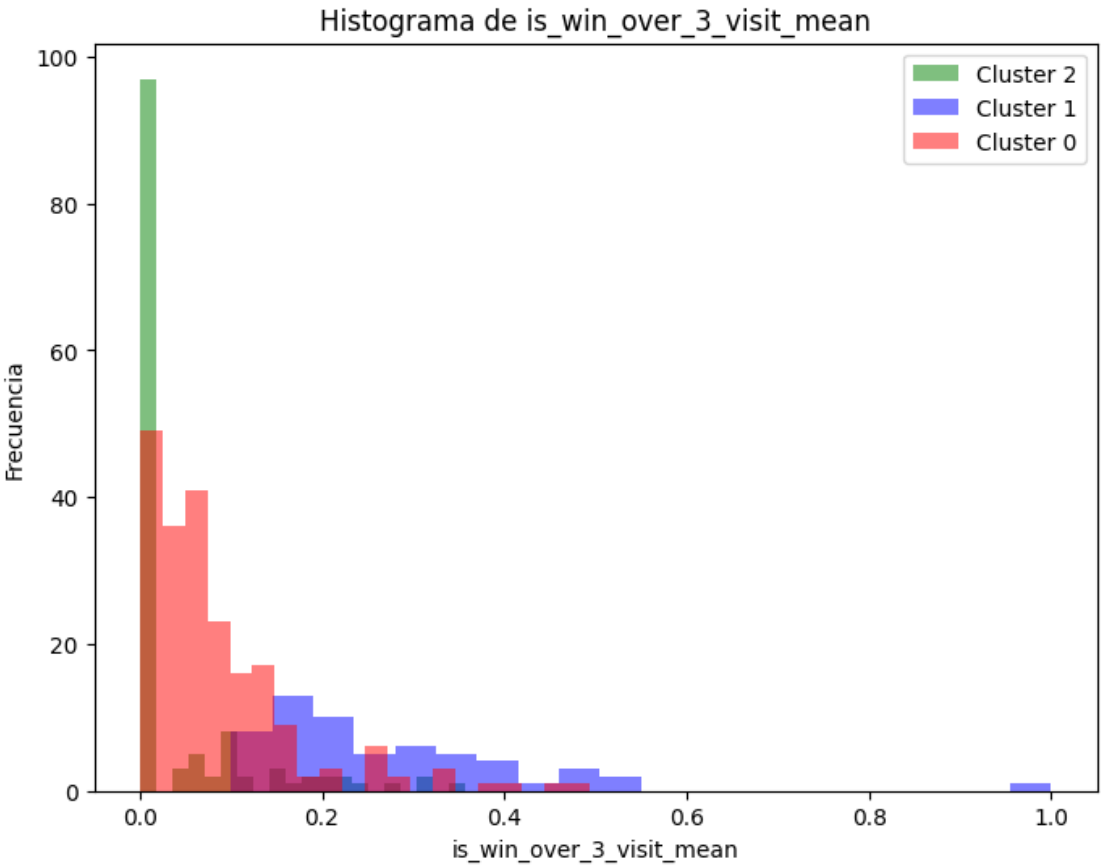

Histograma de Total_goals_sum_y
Cluster 2
Cluster 1
Cluster 0
Frecuencia
0
5
10
15
20
0.0
0.2
0.4
0.6
0.8
1.0
Total_goals_sum_y

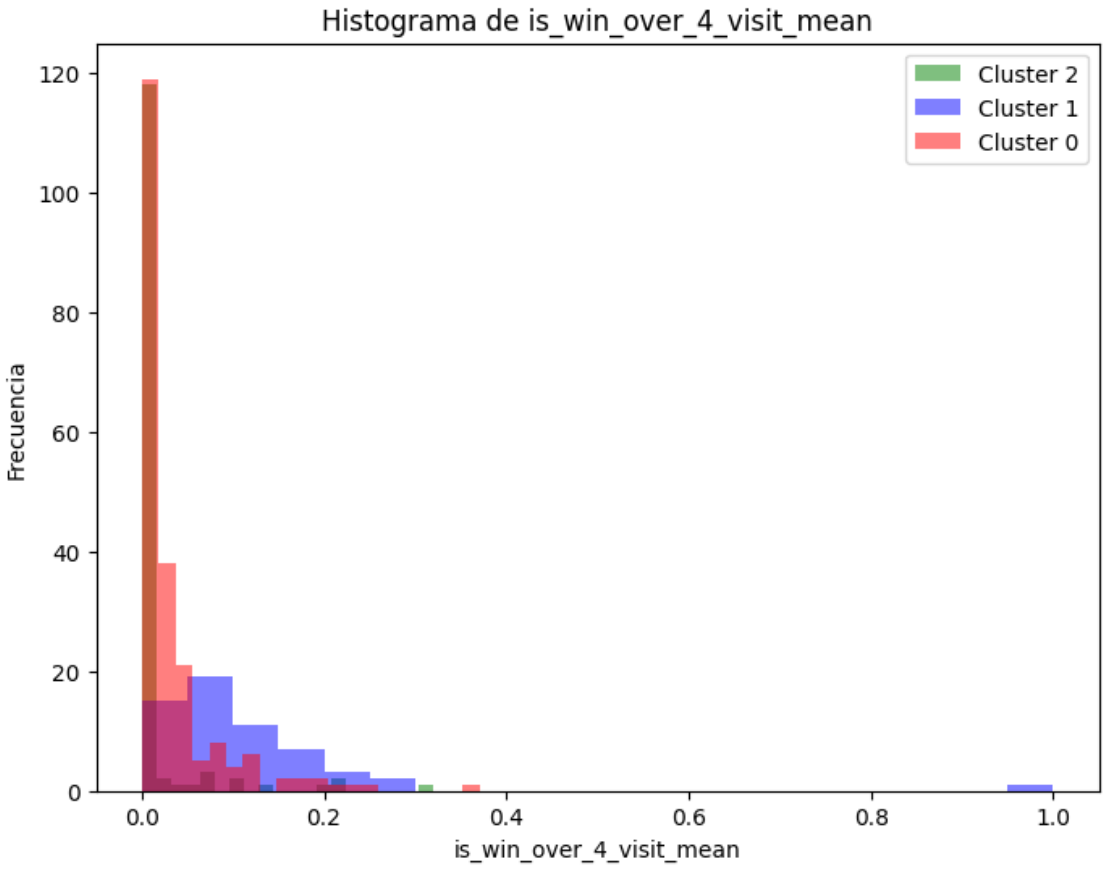

Histograma de Total_goals_min_y
Cluster 2
Cluster 1
Cluster 0
Frecuencia
Total_goals_min_y
0
10
20
30
40
50
60
70
80
0.0
0.2
0.4
0.6
0.8
1.0

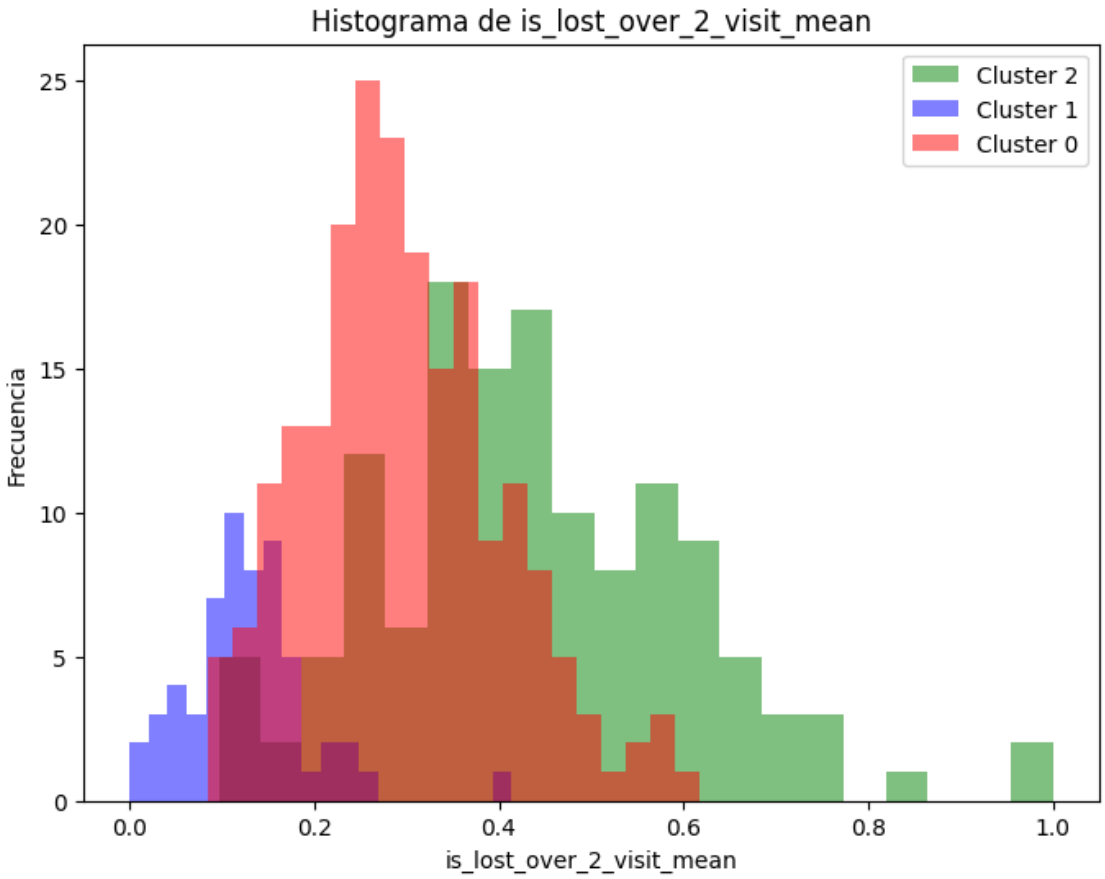

Histograma de Total_goals_mean_y
Cluster 2
Cluster 1
Cluster 0
Frecuencia
0
5
10
15
20
0.0
0.2
0.4
0.6
0.8
1.0
Total_goals_mean_y

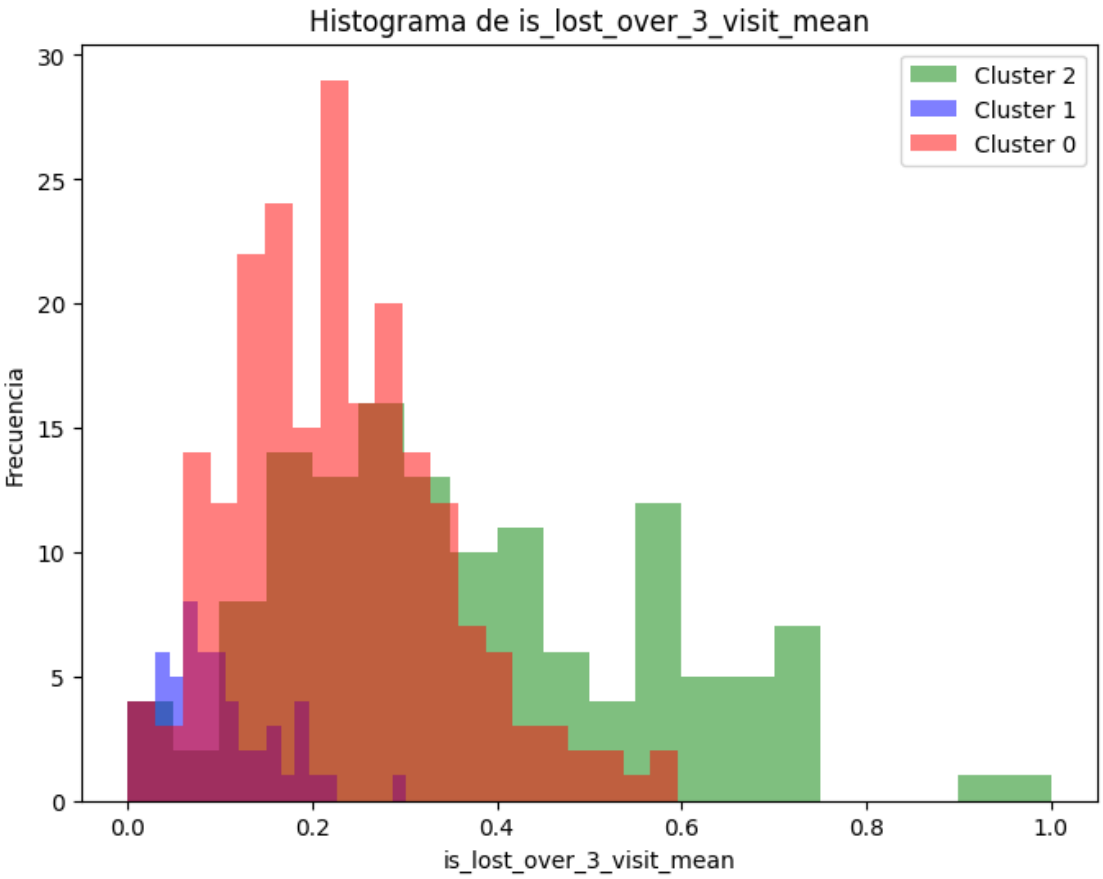

Histograma de Total_goals_median_y
Cluster 2
Cluster 1
Cluster 0
Frecuencia
120
100
80
60
40
20
0
0.0
0.2
0.4
0.6
0.8
1.0
Total_goals_median_y

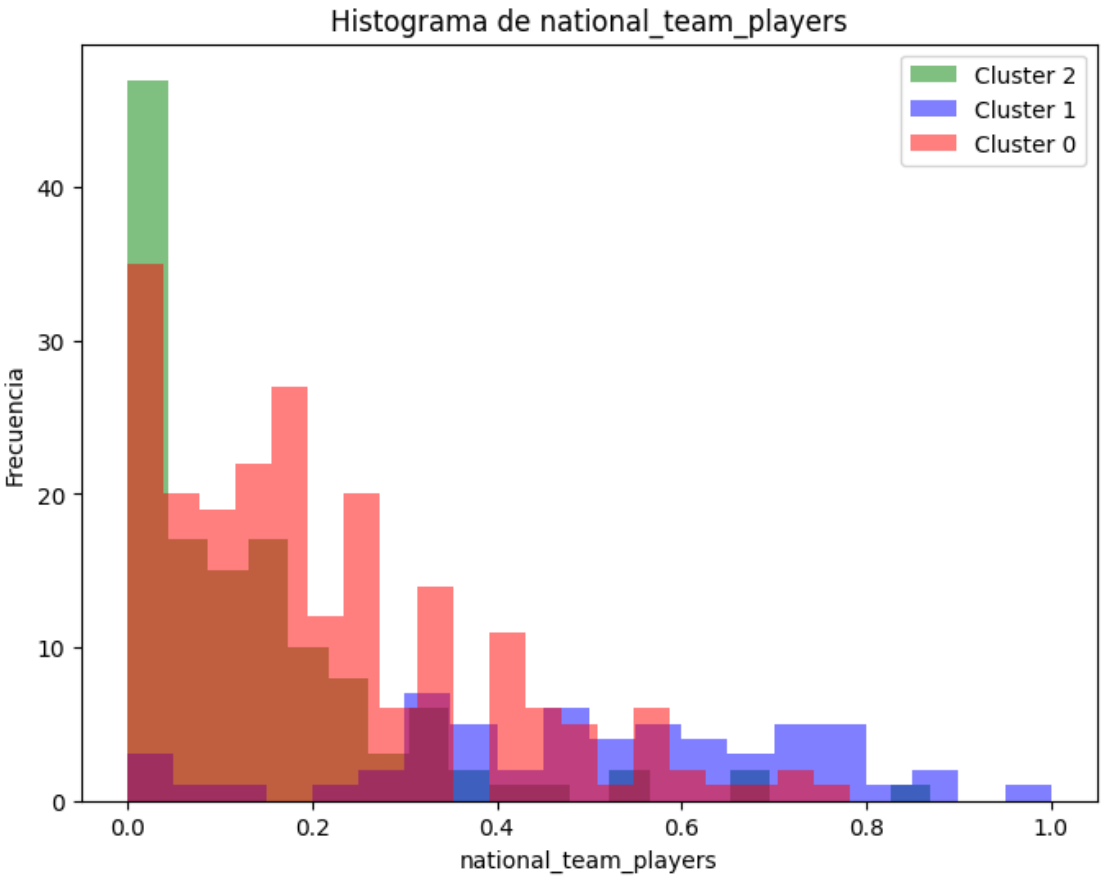

Histograma de is_win_over_2_visit_mean
Cluster 2
Cluster 1
Cluster 0
Frecuencia
is_win_over_2_visit_mean
0
10
20
30
40
50
0.0
0.2
0.4
0.6
0.8
1.0

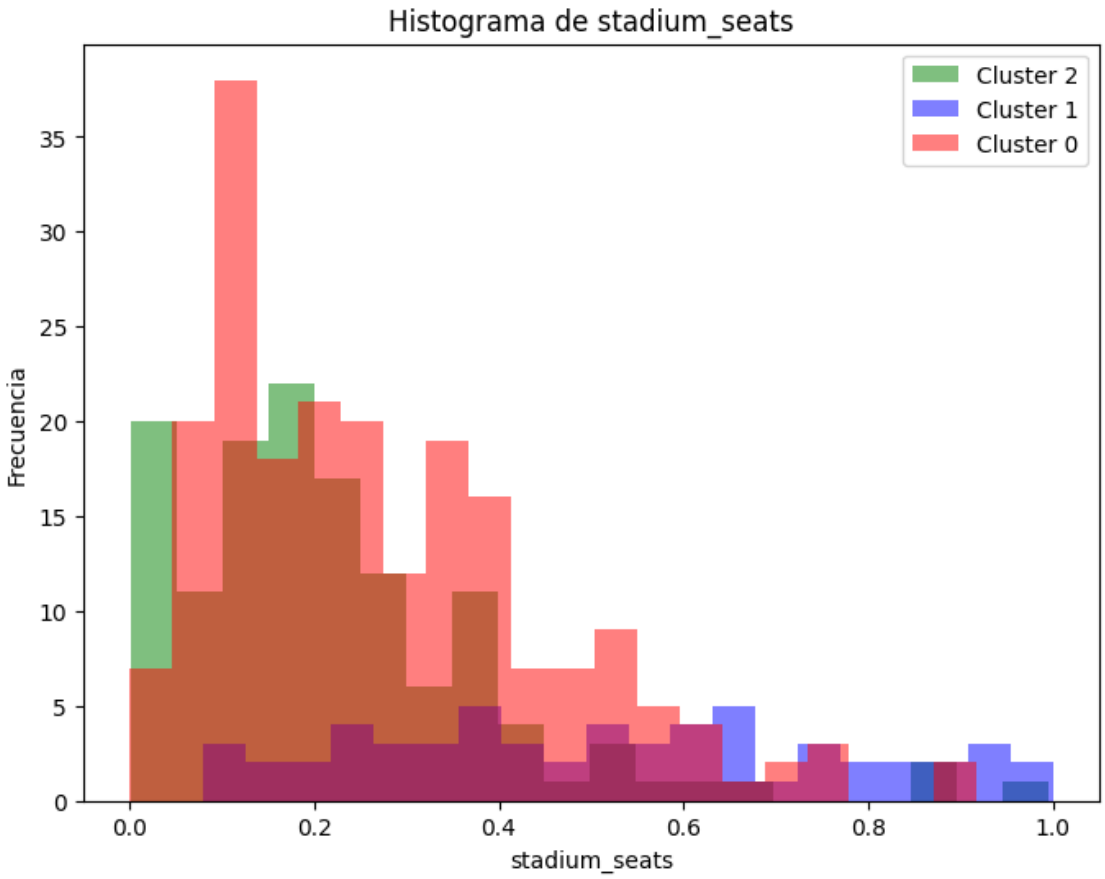

Histograma de is_win_over_3_visit_mean
Cluster 2
Cluster 1
Cluster 0
Frecuencia
100
80
60
40
20
0
0.0
0.2
0.4
0.6
0.8
1.0
is_win_over_3_visit_mean


Histograma de is_win_over_4_visit_mean
Cluster 2
Cluster 1
Cluster 0
Frecuencia
120
100
80
60
40
20
0
0.0
0.2
0.4
0.6
0.8
1.0
is_win_over_4_visit_mean


Histograma de is_lost_over_2_visit_mean
Cluster 2
Cluster 1
Cluster 0
Frecuencia
0
5
10
15
20
25
0.0
0.2
0.4
0.6
0.8
1.0
is_lost_over_2_visit_mean


Histograma de is_lost_over_3_visit_mean
Cluster 2
Cluster 1
Cluster 0
Frecuencia
is_lost_over_3_visit_mean
0
5
10
15
20
25
30
0.0
0.2
0.4
0.6
0.8
1.0


Histograma de national_team_players
Cluster 2
Cluster 1
Cluster 0
Frecuencia
0
10
20
30
40
0.0
0.2
0.4
0.6
0.8
1.0
national_team_players



[428]:
```
Xmds_sample
```

[428]:

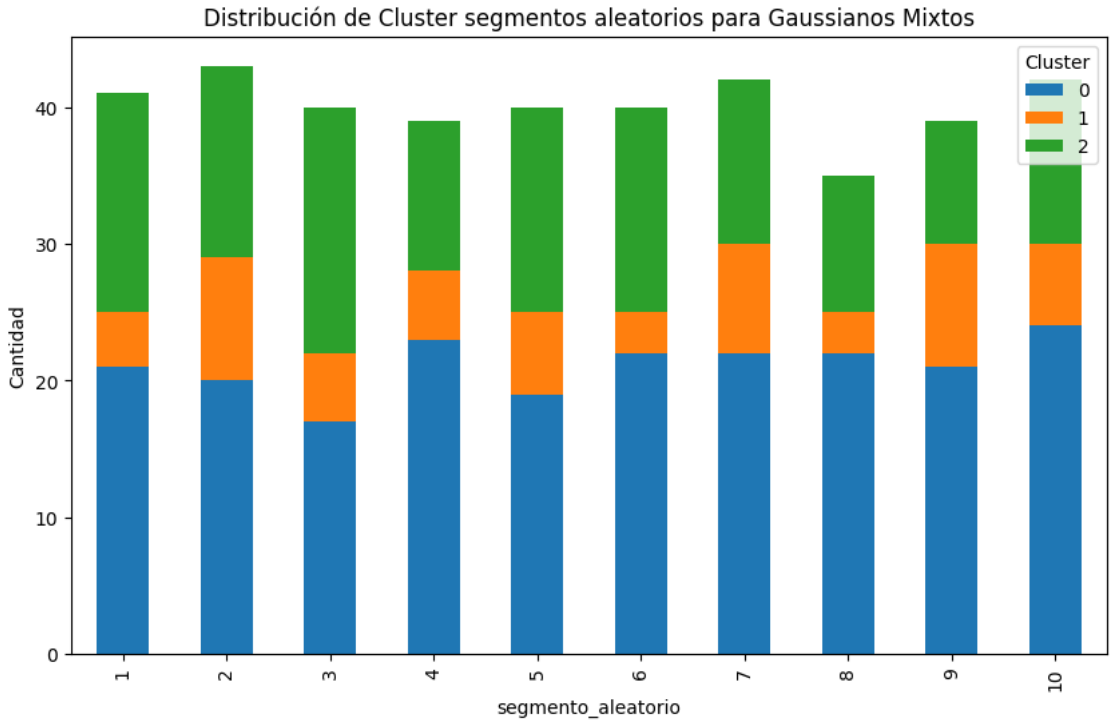
	d_0	d_1	d_2	kmeans_mds_3	cl_gmm	ward_3
0	-0.008888	-0.123359	-0.802850	0	0	2
1	-0.962083	0.013824	-0.153736	0	2	1
2	1.370340	0.660561	-0.816468	1	1	0
3	-1.042893	1.126775	1.189048	0	2	1
4	-0.920295	-0.200893	-0.389024	0	2	1
..	…	…	…	…	…	…
396	0.276149	-0.483708	1.855365	2	2	2
397	1.023684	-0.531060	0.685269	1	0	2
398	0.187596	-0.792615	0.528512	2	0	2
399	-0.838562	-0.388762	0.998001	2	2	1
400	-0.277489	-0.303818	0.324587	2	0	1

[401 rows x 6 columns]

8 reportes de estabilidad

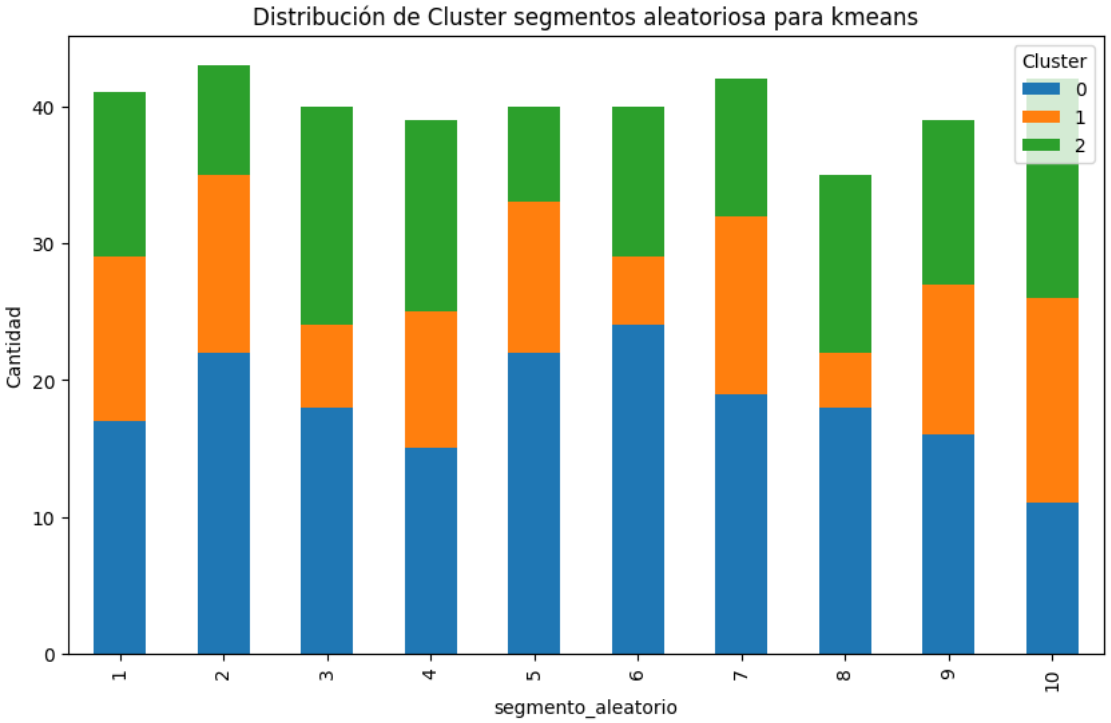
[429]:
```
aleatorio=[]
for i in range (len(Xmds_sample)):
    aleatorio.append(np.random.randint(1,11))
Xmds_sample['chunk']=aleatorio
```

[430]:
```
cluster_month_counts = Xmds_sample.groupby(['chunk','cl_gmm']).size().
 ↪unstack(fill_value=0)

# Luego, puedes crear un gráfico de barras apiladas.
cluster_month_counts.plot(kind='bar', stacked=True, figsize=(10, 6))
plt.xlabel('segmento_aleatorio')
plt.ylabel('Cantidad')
plt.title('Distribución de Cluster segmentos aleatorios para Gaussianos Mixtos')
plt.legend(title='Cluster', loc='upper right')
plt.show()
```

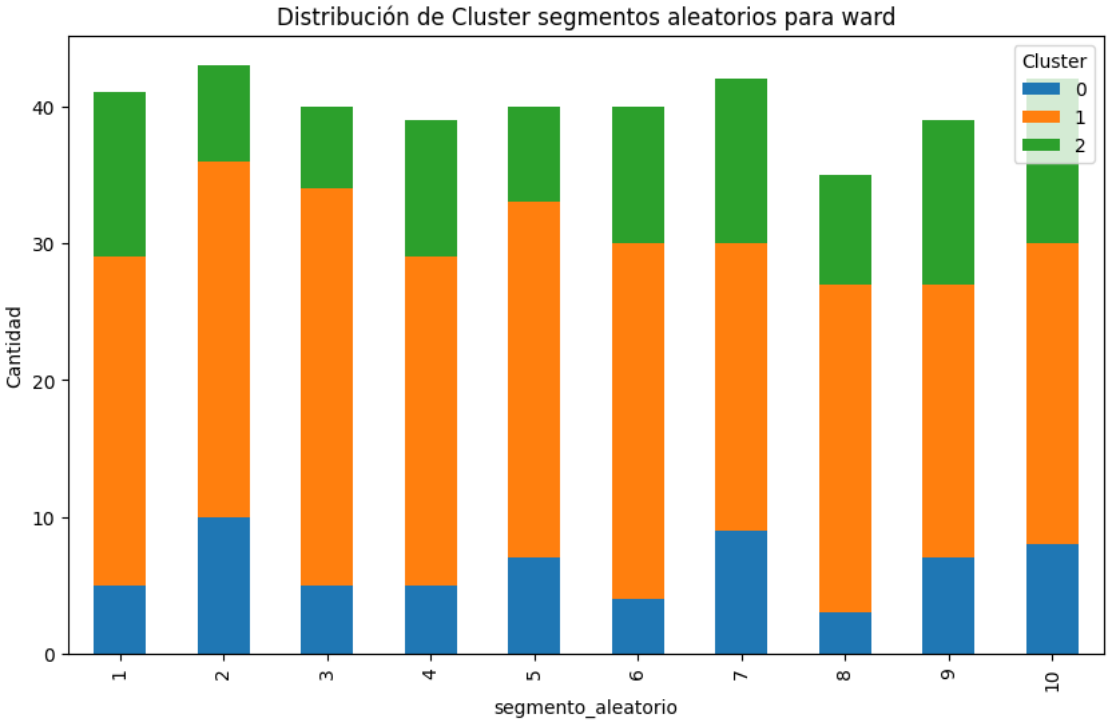


[431]:
```
cluster_month_counts = Xmds_sample.groupby(['chunk','kmeans_mds_3']).size().
 ↪unstack(fill_value=0)

# Luego, puedes crear un gráfico de barras apiladas.
cluster_month_counts.plot(kind='bar', stacked=True, figsize=(10, 6))
```

```
plt.xlabel('segmento_aleatorio')
plt.ylabel('Cantidad')
plt.title('Distribución de Cluster segmentos aleatoriosa para kmeans')
plt.legend(title='Cluster', loc='upper right')
plt.show()
```



[432]:
```
cluster_month_counts = Xmds_sample.groupby(['chunk','ward_3']).size().
 ↪unstack(fill_value=0)

# Luego, puedes crear un gráfico de barras apiladas.
cluster_month_counts.plot(kind='bar', stacked=True, figsize=(10, 6))
plt.xlabel('segmento_aleatorio')
plt.ylabel('Cantidad')
plt.title('Distribución de Cluster segmentos aleatorios para ward')
plt.legend(title='Cluster', loc='upper right')
plt.show()
```


Distribución de Cluster segmentos aleatorios para ward
Cluster
0
1
2
Cantidad
0
10
20
30
40
1
2
3
4
5
6
7
8
9
10
segmento_aleatorio

[ ]: