




P		NOM	PUNTS	J	G	E	P	F	C
1		JOVENTUT TEIA CLUB FUTBOL B	49	17	16	1	0	146	26
2		ROCAFONDA CLUB FUTBOL A	45	17	15	0	2	118	22
3		PREMIA DALT, C.D. C	42	16	14	0	2	109	30

G	G	G	G	N
G	G	G	P	N
G	P	G	P	N

... una liga cualquiera ...

- 7 jugadores. 60 x 40 m.
- 3 puntos (victoria) / 1 punto (empate)
- 60' (4 x 15')



**Info**

1. Descripción
2. Limpieza
3. Transformación

**Preparación datos**

404 non-null int64

404 non-null int64

total 14 columns

```
equipos.duplicated().sum()
```

0

**Descripción**

```
equipos['puntos'].describe().round(3)
```

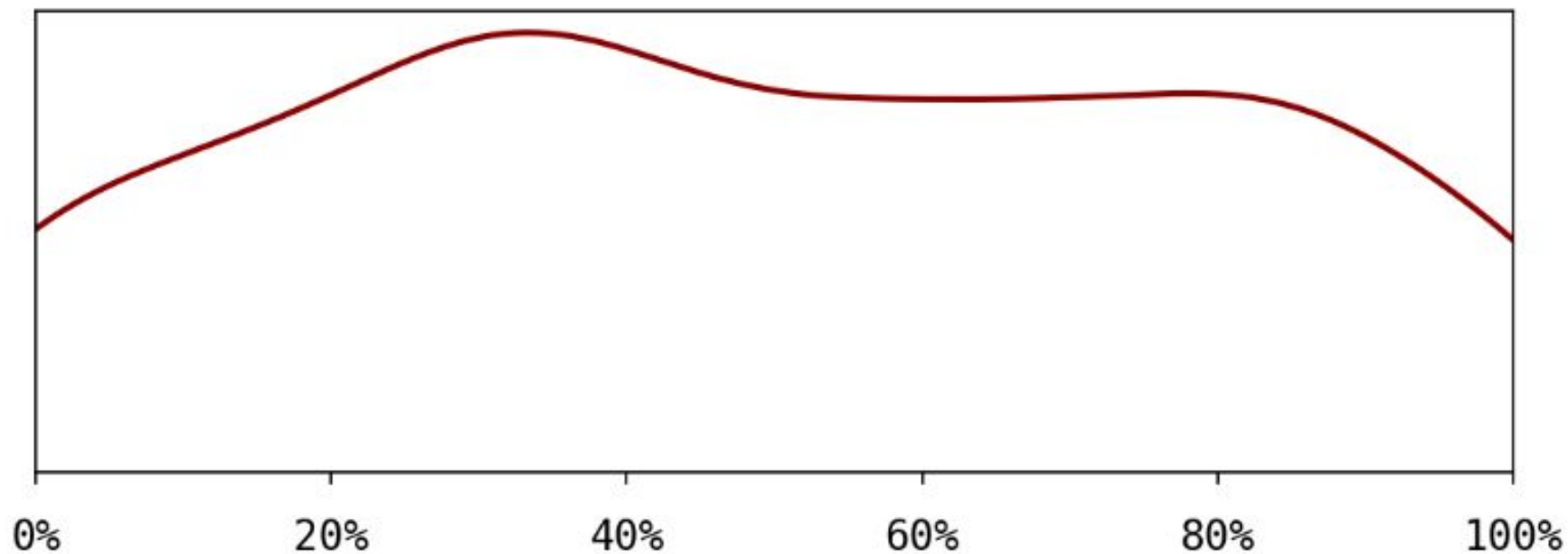
25%	0.773
50%	1.467
75%	2.200

**Descripción**

	mean	std
gan_loc	0.494	0.307
emp_loc	0.056	0.090
per_loc	0.449	0.307

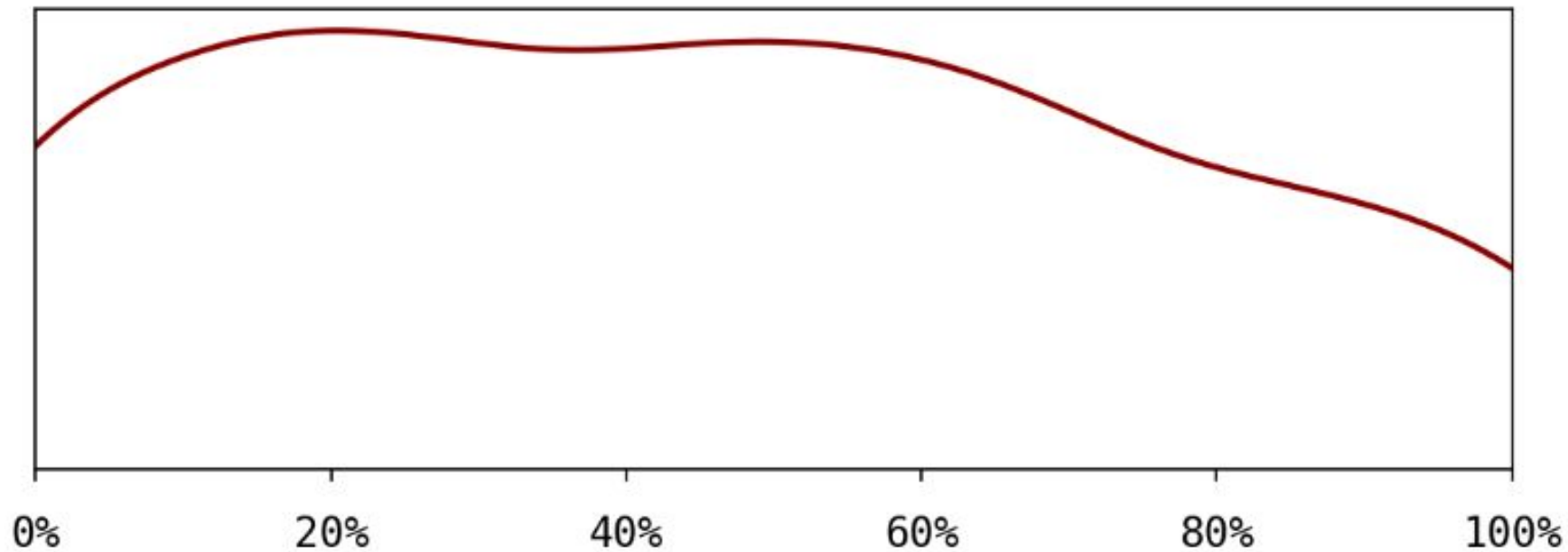
**Descripción**

## Ganados



Descripción

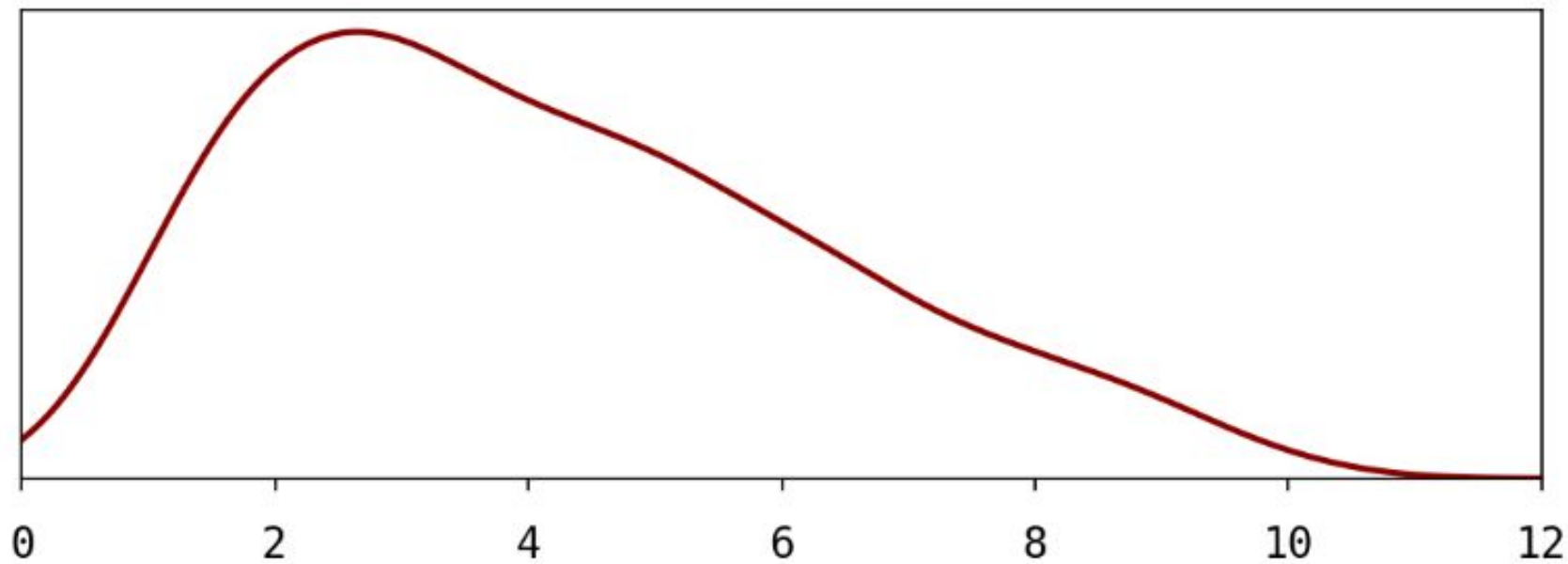
## Perdidos



Descripción



Goles



Descripción

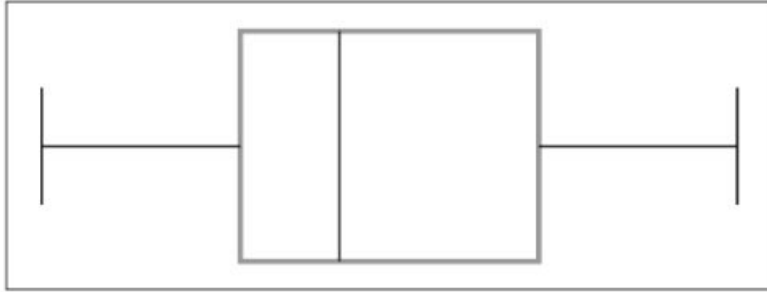
	nombre	dif_gol
284	ALMEDA, C.D. A	9.50
117	SABADELLENCA, UE. B	9.21
238	SINERA UNITED FUTBOL CLUB ASS. C	8.73

**Descripción**

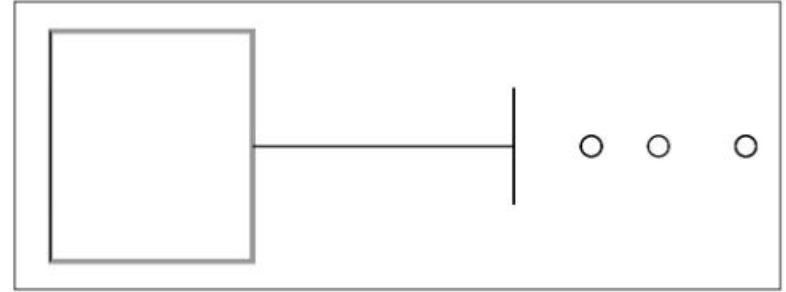
```
for l in columnas_loc:  
    equipos[l] = equipos[l] / equipos['jug_loc']  
  
for v in columnas_vis:  
    equipos[v] = equipos[v] / equipos['jug_vis']  
  
for g in columnas_gol:  
    equipos[g] = equipos[g] / (equipos['jug_loc'] + equipos['jug_vis'])
```

**Transformación**

gan\_loc



emp\_loc



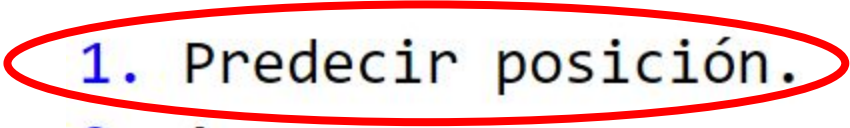
gan\_loc: No gaussiana -  $p = 0.000$

Outliers & Shapiro

J&J

Tipo de Dato	Condición	Transformación Recomendada
Datos con Distribución Gaussiana	-	StandardScaler
Datos con Distribución No Gaussiana	Sin Outliers	MinMaxScaler
Datos con Distribución No Gaussiana	Con Outliers	RobustScaler

**Transformación**



1. Predecir posición.

2. Agrupar.

**Objetivos**

Model	Adjusted R-Squared	R-Squared
HuberRegressor	0.86	0.90
OrthogonalMatchingPursuitCV	0.86	0.90
RANSACRegressor	0.86	0.90
LinearRegression	0.86	0.90

**Modelo**

gan_loc	1.0	-0.1	-1.0
emp_loc	-0.1	1.0	-0.2
per_loc	-1.0	-0.2	1.0
	gan_loc	emp_loc	per_loc

¿Colinealidad?



		VIF
0	const	8.134313
1	gan_loc	7.055540
2	gan_vis	6.314240
3	dif_gol	15.836779

¿Colinealidad?

```
X_train_esc.drop(['dif_gol', 'per_loc', 'per_vis'], axis=1, inplace=True)
```

```
X_train_esc['gan'] = (X_train_esc['gan_loc'] + X_train_esc['gan_vis'])/2
```

```
X_min_train_esc = X_train_esc.drop(['emp_loc', 'emp_vis'], axis=1)
```

**Transformación 2**

```
X_min_train_esc = X_train_esc[['gan']]
```

Regresión lineal, 1 variable independiente, R2: 0.897

**Rendimiento**

```
from sklearn.model_selection import cross_val_score
```

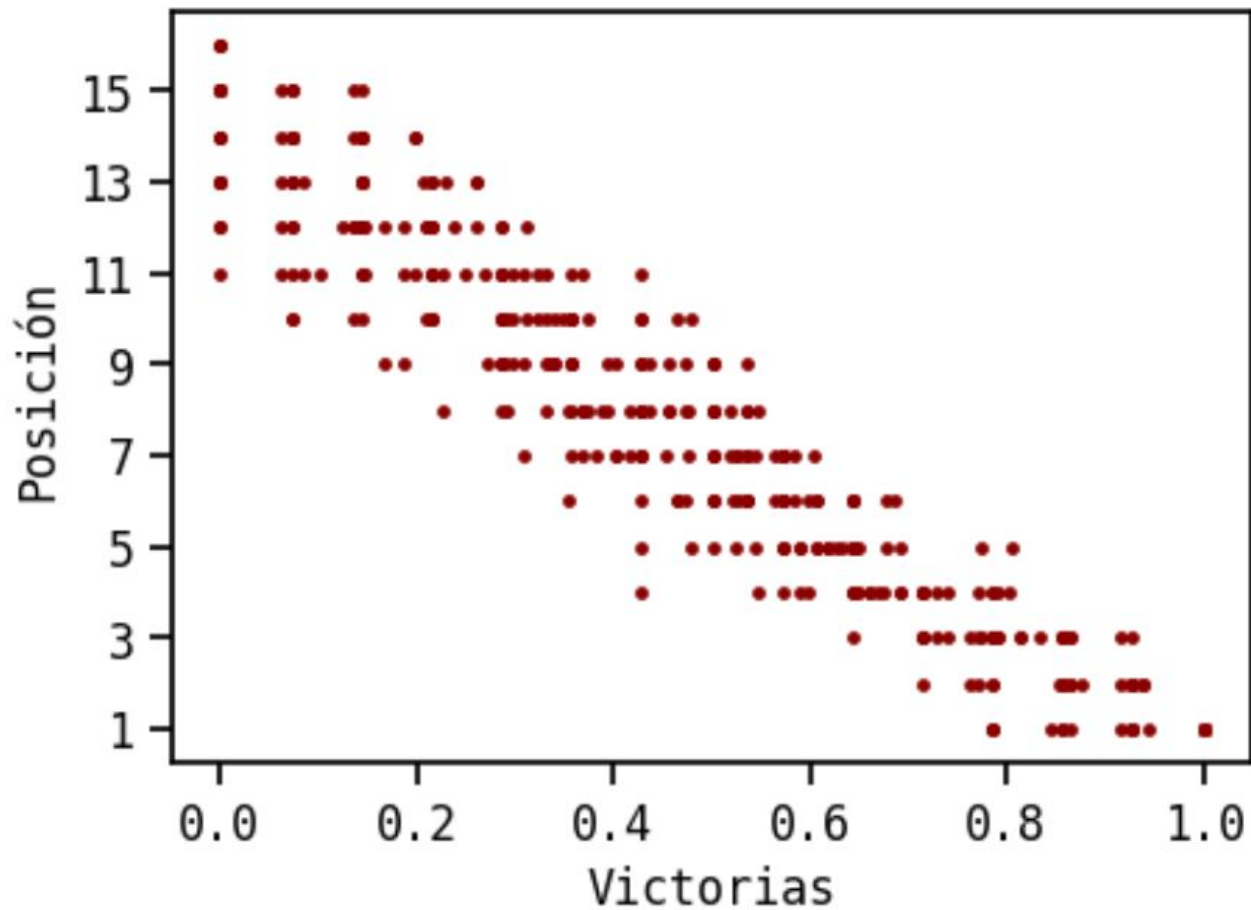
```
[0.90810109 0.90824179 0.93279942 0.9093283 0.89677121]
```

**+ Rendimiento**

1. Predecir posición.

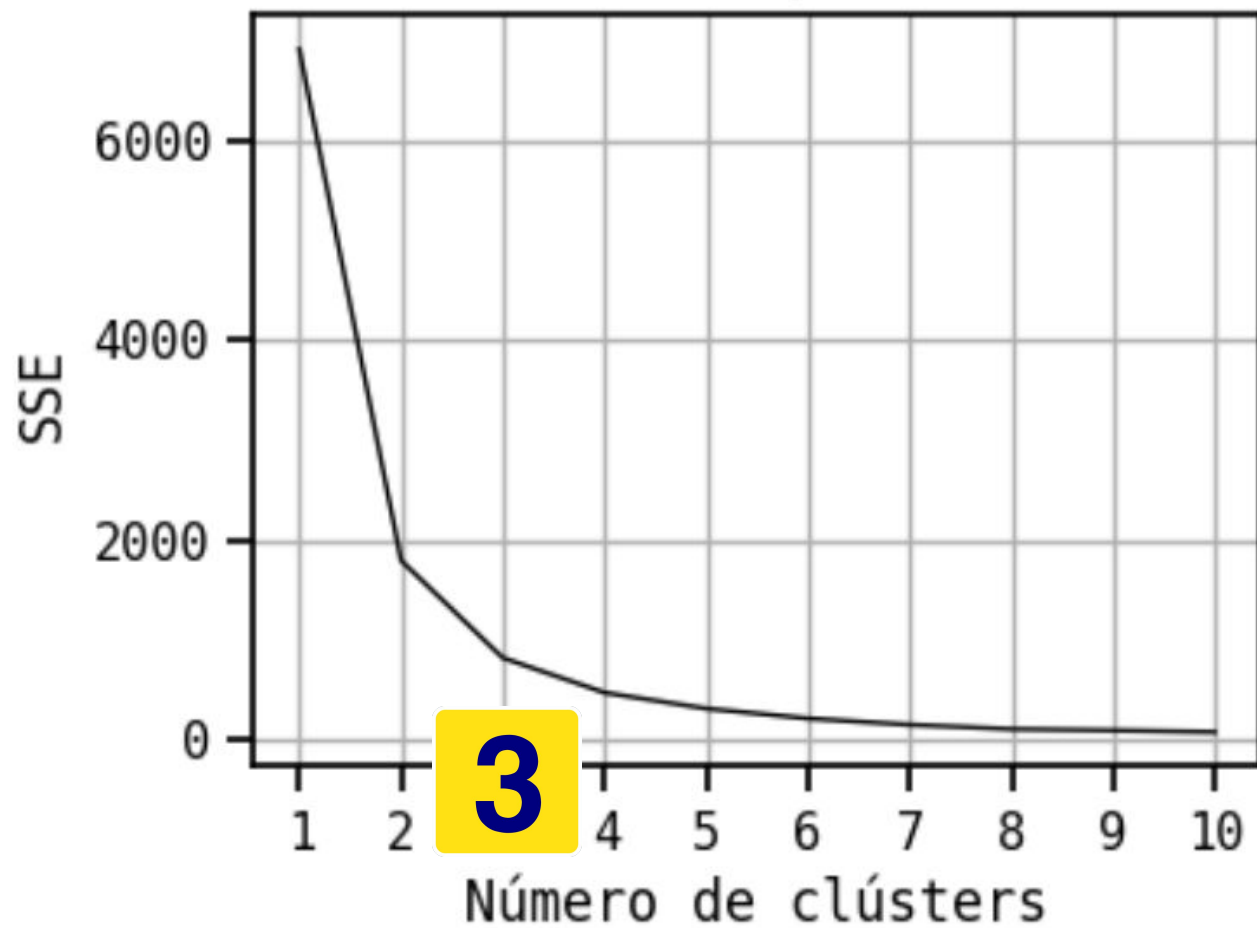
2. Agrupar.

**Objetivos**



Dispersión

K-means / Data



Clústers

**Posicion Cluster Conteo**

**4 5 0 29**

**5 6 1 29**

**6 7 1 29**

**7 8 1 29**

**8 9 1 29**

**9 10 1 29**

**10 11 2 29**

**Clústers**



```
score = silhouette_score(data, kmeans.labels_).round(3)
```

0.629

¿Fin?



Como conclusión, volvemos al principio. Hemos visto cómo prescindiendo de prácticamente todas las variables, incluyendo los puntos por partido, y solamente utilizando la media de victorias de los equipos, como local y como visitante, no sólo podemos obtener la clasificación de cada equipo de manera bastante precisa utilizando un sencillo modelo de regresión lineal, sino que además podemos agruparlos en nodos que los asimilan, de manera que podemos reconocer las diferencias de puntos por partido o incluso goles en función del nodo al que pertenece cada equipo, y esto utilizando un algoritmo con un rendimiento Silhouette de sólo un 60%. Con todo ello podemos ver la potencia de la estadística y el machine learning en la predicción de tendencias y patrones, y hasta qué punto podemos llegar a reducir el uso de datos, con lo que ello implica en mejoras de eficiencia. Ahora sí, fin. Gracias por la atención.

Data Science 

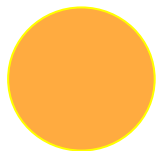
IT Academy 29/02/2024

Agradecimientos:

 Lucía Álvarez, profesora.

 Compis, por los buenos ratos y las complicidades ;)

Happy hour for ever!  



aar\_t 