UNIVERSIDAD MAYOR DE SAN ANDRÉS

FACULTAD DE CIENCIAS PURAS Y NATURALES

CARRERA INFORMÁTICA

PRIMER PARCIAL

PREGUNTAS PRIMER PARCIAL

MATERIA: INTELIGENCIA ARTIFICIAL (DAT-245)

DOCENTE: Ph.D. MOISES MARTIN SILVA CHOQUE

NOMBRE: UNIV. MOISES MARTIN CONDORI YUJRA

LA PAZ – BOLIVIA

2024

1. ¿Cuál es el área de su interés que le gustaría investigar multidisciplinariamente con IA? Describa en un párrafo las características del área, métodos y metodologías. (Ej. Cáncer de pulmón, comprende que es un cáncer de pulmón, como se detecta, que parámetros clínicos se toma en cuenta para su detección, factores biológicos, físico y otros).

Respuesta:

Área de Interés: **Análisis de Rendimiento y Predicción de Resultados en la Premier League usando IA**

El análisis del rendimiento deportivo en ligas de fútbol, como la Premier League, puede beneficiarse enormemente de la inteligencia artificial. Este campo abarca el estudio de jugadores, equipos y resultados a través de datos históricos y en tiempo real. Se podrían utilizar metodologías de **aprendizaje automático** y **análisis predictivo** para identificar patrones y tendencias en los datos, como predicción de resultados, evaluación del rendimiento de jugadores y análisis táctico de los equipos.

El dataset incluiría una variedad de características como:

- **Estadísticas de partidos**: número de goles, posesión, tiros a puerta, tarjetas amarillas/rojas, faltas cometidas, entre otros.
- **Rendimiento de jugadores**: datos individuales sobre goles, asistencias, pases completados, kilometraje recorrido, y rendimiento físico.
- **Características de los equipos**: alineaciones, tácticas, formaciones utilizadas, cambios realizados durante el partido.
- **Parámetros adicionales**: condiciones del clima, la localización del partido (local o visitante), estado del césped, y otros factores externos que puedan influir en los resultados.

Las columnas de mi datasets son:

Acerca del conjunto de datos
Descripción de las características
date : la fecha del juego
time : la hora del juego
comp : la competición del juego
round : la ronda del juego
day : el día de la semana del juego
venue : la sede del juego
result : el resultado del juego
golesloc : los goles del equipo local
golesvis : los goles del equipo visitante
opponent : el oponente del equipo local
golesesploc : los goles esperados para el equipo local
golesespvis : los goles esperados para el equipo visitante

poss : la posesión del equipo local
captain : el capitán del equipo local
formation : la formación del equipo local
referee : el árbitro del juego
tirosloc : los tiros del equipo local
tirosalarco : los tiros a puerta del equipo local
distpromtir : la distancia promedio de los tiros del equipo local
tiroslibloc : los tiros libres del equipo local
tirospenloc : los tiros penales del equipo local equipo
tirospenintloc: los tiros penales que ha intentado el equipo local
season: la temporada del año del partido
team: el equipo local

2. Selección un datatset tabular de al menos 1000 columnas, 14 filas. Si elige imágenes igualmente puede convertir la imagen en datos tabulares de NxM. De esta selección indique cual es la clase o si no tiene.

Complemente con lo siguiente:

a. Sin el uso de librerías en Python programe el percentil y cuartil de cada columna. Que distribución se puede aplicar en su caso normal, Bernoulli, gaussiana, poisson, otros. Indique la razón de su uso graficando con matplotlib.

```
import pandas as pd
import numpy as np

# librerias para graficar
import seaborn as sns
import matplotlib.pyplot as plt
import random

##########################################################################################
#################
# guardando en un array el dataset
url = 'https://raw.githubusercontent.com/moisesmartincy/DAT245---INTELIGENCIA-
ARTIFICIAL/refs/heads/main/DATASETS/matchesmod1.csv'
datos = pd.read_csv(url, encoding="unicode_escape", on_bad_lines='skip');
dataset = datos.to_numpy()
titles = ['id','date', 'time', 'comp', 'round', 'day', 'venue', 'result',
'golesloc', 'golesvis', 'opponent',
'golesesploc','golesespvis','poss','attendance','captain','formation','referee','ma
tch
report','notes','tirosloc','tirosalarco','distpromtir','tiroslibloc','tirospenloc',
'tirospenintloc','season','team']
#print(datos)
##########################################################################################
#################


################################################
```

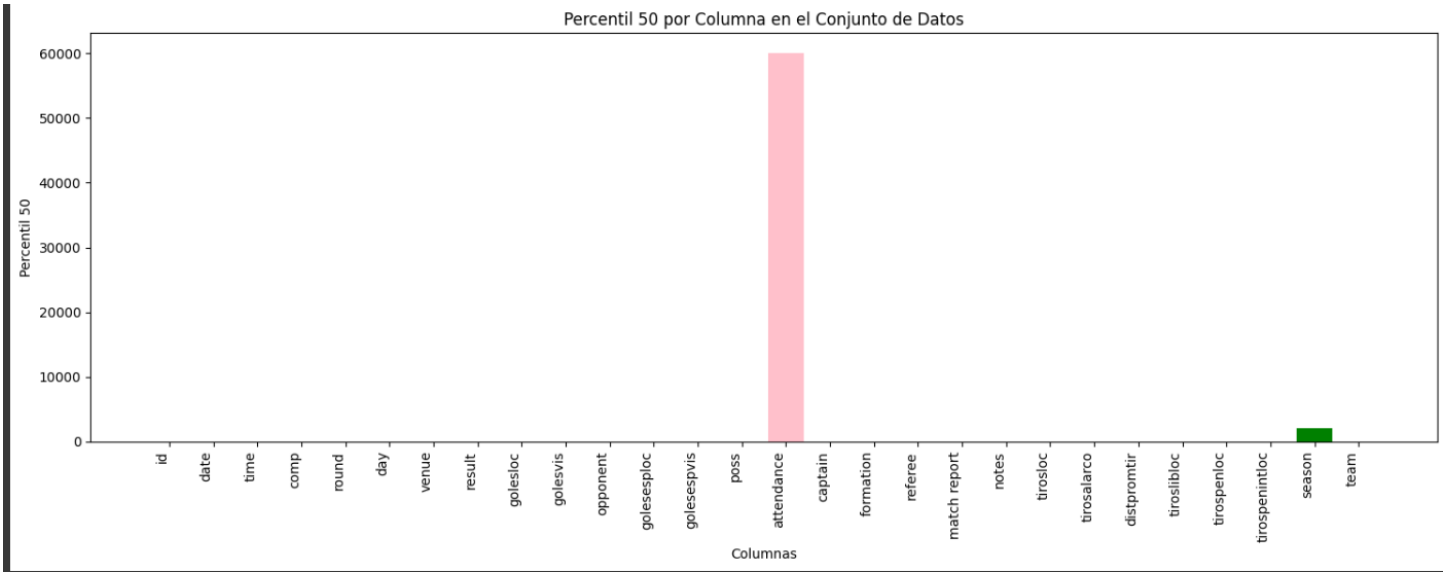
```
# FUNCIONES
def percentil(k):
  print("PERCENTIL ", k)
  percentiles = []
  for i in range(len(dataset[0])):
    sum = 0;
    if( type(dataset[0][i]) != str):
      vector = []
      for j in range(len(dataset)):
        vector.append(dataset[j][i])

      vector.sort()
      cur = k * (len(vector) + 1) / 100
      pos = int(cur)
      print(titles[i], end=  ' ')

      if(cur == pos):
        # quartil es exacto
        print(vector[pos - 1])
        percentiles.append(vector[pos - 1])
      else:
        x1 = vector[pos - 1]
        x2 = vector[pos]
        total = abs(x2 + x1) / 2
        print(total)
        percentiles.append(total)
    else:
      percentiles.append(0)
  print()

  string_tipo = "Percentil " + str(k)
  graficar(titles, percentiles, string_tipo)

def cuartil(k):
  print("QUARTIL ", k)
  quartiles = []
  for i in range(len(dataset[0])):
    sum = 0;
    if( type(dataset[0][i]) != str):
      vector = []
      for j in range(len(dataset)):
        vector.append(dataset[j][i])

      vector.sort()
      cur = k * (len(vector) + 1) / 4
      pos = int(cur)
      print(titles[i], end=  ' ')

      if(cur == pos):
        # quartil es exacto
```

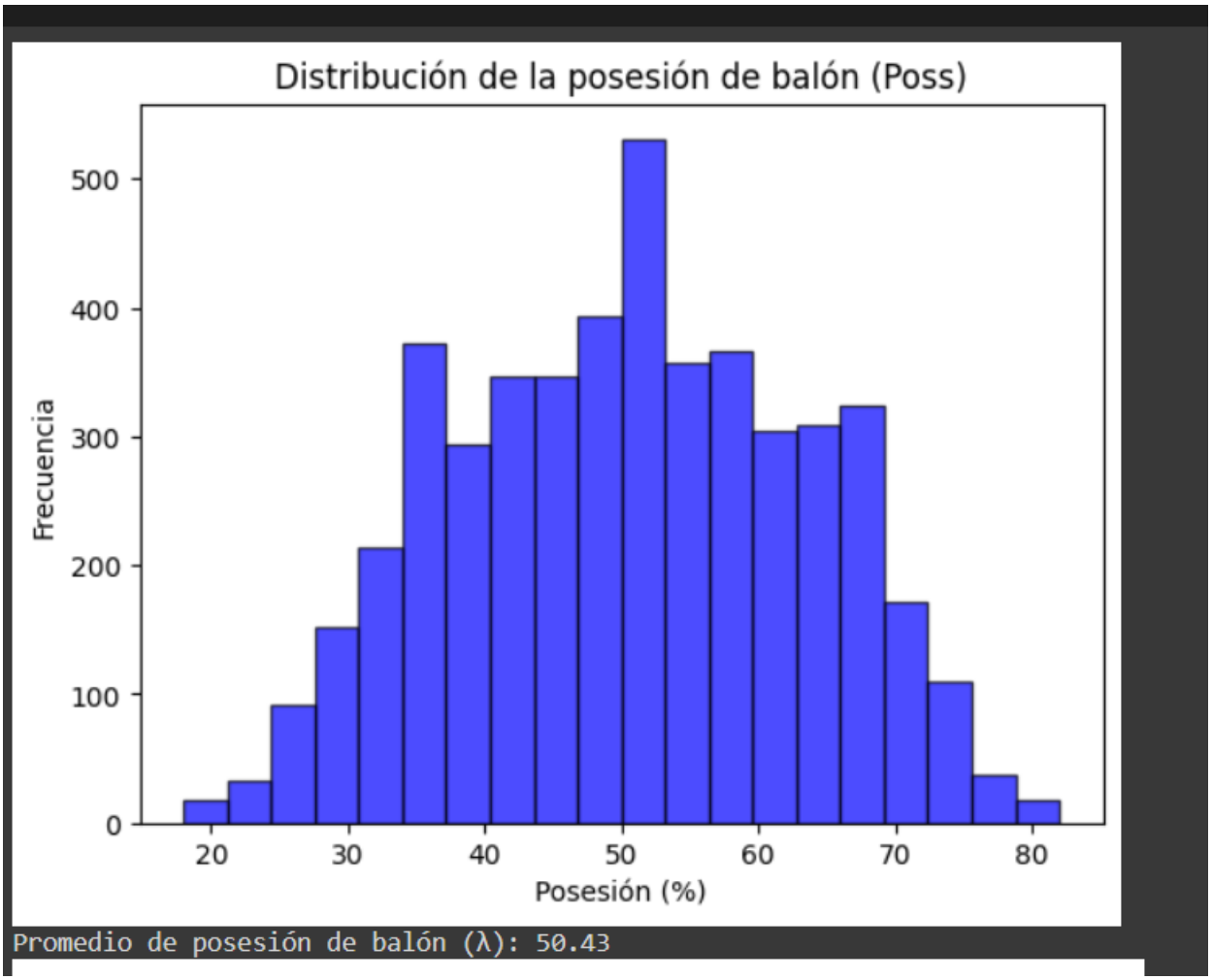
```
        print(vector[pos - 1])
        quartiles.append(vector[pos - 1])
      else:
        x1 = vector[pos - 1]
        x2 = vector[pos]
        total = abs(x2 + x1) / 2
        print(total)
        quartiles.append(vector[pos - 1])
    else:
      quartiles.append(0)
  print()
  string_tipo = "Quartil " + str(k)
  graficar(titles, quartiles, string_tipo)




def media():
  print("CALCULANDO EL PROMEDIO POR COLUMNAS DEL DATASET")
  promedios = []
  for i in range(len(dataset[0])):
    sum = 0;

    if(type(dataset[0][i]) != str):
      for j in range(len(dataset)):
        sum += dataset[j][i];
      sum /= len(dataset)

    promedios.append(sum)
    print(titles[i], sum)
  print()
  graficar(titles, promedios, "media")


def moda():
  print("CALCULANDO LA MODA POR COLUMNAS DEL DATASET")
  total_modas = []
  for i in range(len(dataset[0])):
    mapa = {}
    for j in range(len(dataset)):
      if(dataset[j][i] in mapa):
        mapa[dataset[j][i]] += 1;
      else:
        mapa[dataset[j][i]] = 1;

    lista = []
    for key, value in mapa.items():
      lista.append([value, key])
```

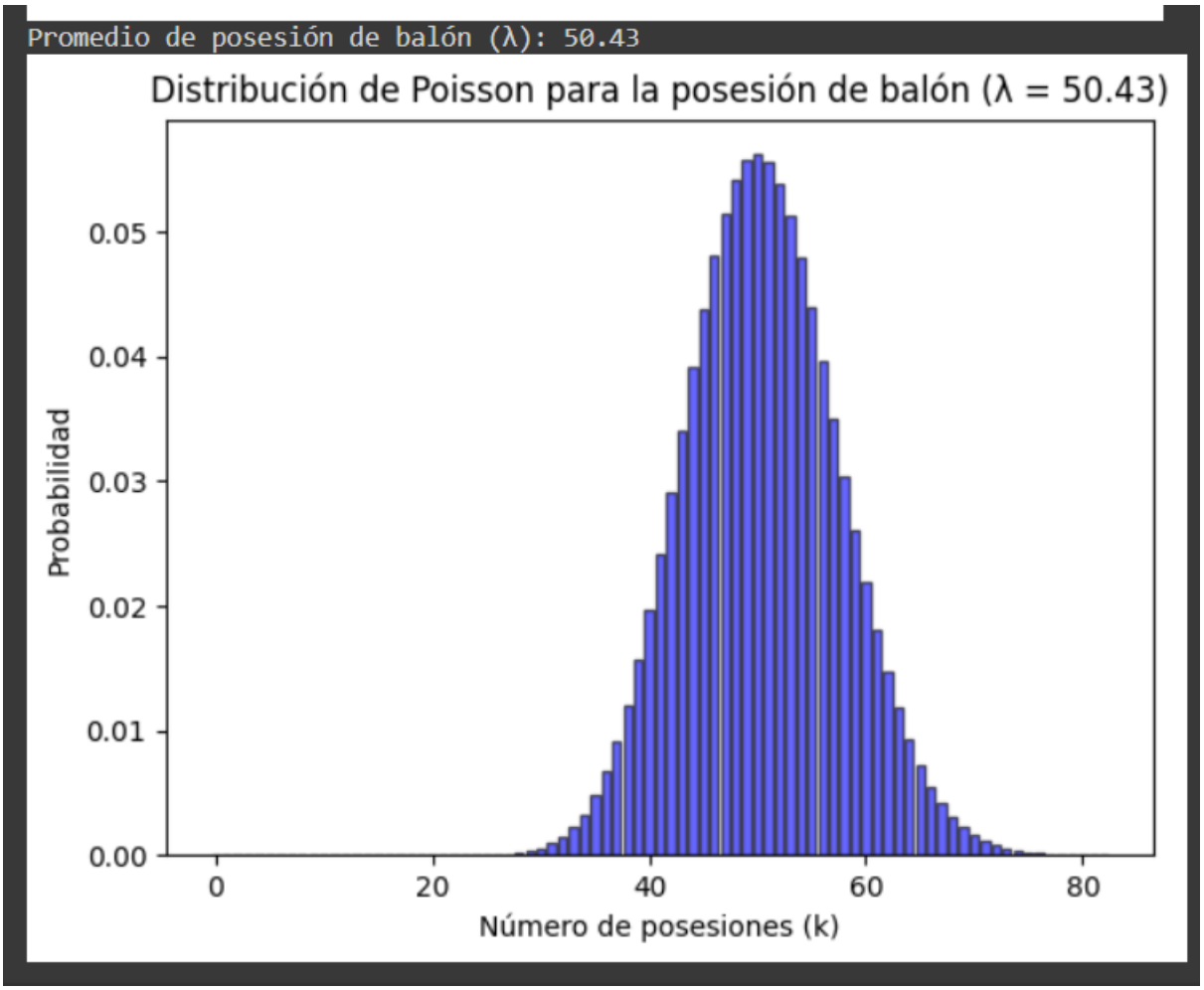
```
    lista.sort(reverse=True)
    print(titles[i], lista[0][1])
    if(type(lista[0][1]) == str):
      total_modas.append(0)
    else:
      total_modas.append(lista[0][1])
  print()
  graficar(titles, total_modas, "Moda")
# Función para calcular la mediana
def mediana():
    print("CALCULANDO LA MEDIANA POR COLUMNAS DEL DATASET")
    medianas = []
    for i in range(len(dataset[0])):
        if type(dataset[0][i]) != str:
            vector = [dataset[j][i] for j in range(len(dataset))]
            vector.sort()
            n = len(vector)
            if n % 2 == 0:
                mediana = (vector[n//2 - 1] + vector[n//2]) / 2
            else:
                mediana = vector[n//2]
            print(titles[i], mediana)
            medianas.append(mediana)
        else:
            medianas.append(0)
    graficar(titles, medianas, "Mediana")

def graficar(indices, valores, titulo):
    colores_aleatorios = [random.choice(['red', 'blue', 'green', 'purple',
'orange', 'pink', 'gray', 'brown']) for _ in indices]
    plt.figure(figsize=(15, 6))  # Aumenta la altura del gráfico para dar más
espacio
    plt.bar(indices, valores, color=colores_aleatorios)

    # Establecer etiquetas y título
    plt.xlabel('Columnas')
    plt.ylabel(titulo)
    plt.title('{} por Columna en el Conjunto de Datos'.format(titulo))

    # Rotar etiquetas del eje X
    plt.xticks(rotation=90, ha='right', fontsize=10)  # Rotar las etiquetas 90
grados y alinearlas a la derecha

    # Mostrar el gráfico
    plt.tight_layout()  # Ajustar el diseño para evitar superposición
    plt.show()
    print()
    print()
```

```
##################################################



############################## MENU PRINCIPAL
##################################################
#pregunta 2a
percentil(50)
percentil(80)
cuartil(1)
cuartil(2)
cuartil(3)

#pregunta 2b

media()
moda()
mediana()  # Llamada a la nueva función de mediana
```

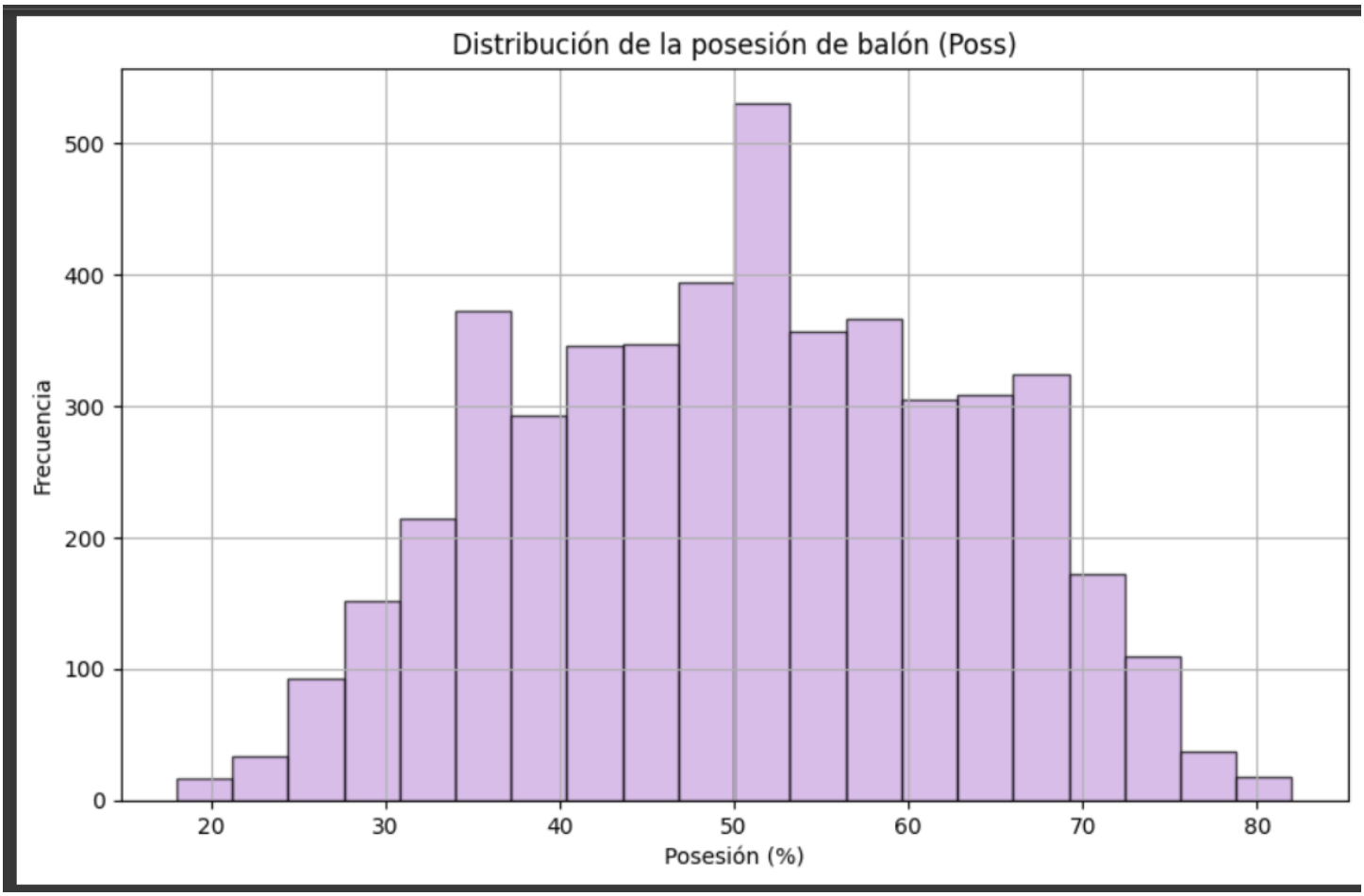
```
PERCENTIL  50
id 62.0
golesloc 1.0
golesvis 1.0
golesesploc 1.3
golesespvis 1.2
poss 51.0
attendance 60090.0
notes nan
tirosloc 12.0
tirosalarco 4.0
distpromtir 17.4
tiroslibloc 0.0
tirospenloc 0.0
tirospenintloc 0.0
season 2023.0
```

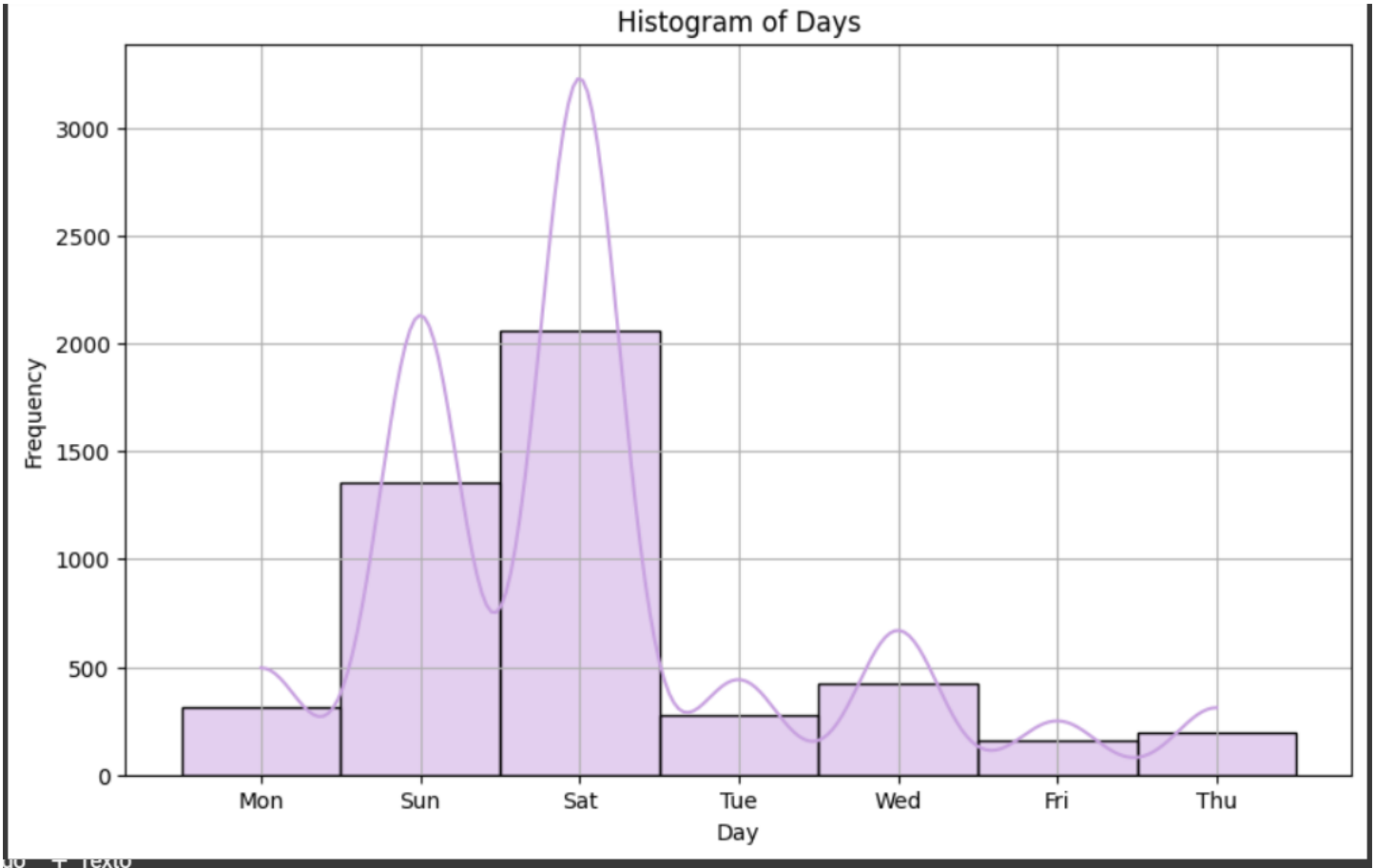

Percentil 50 por Columna en el Conjunto de Datos
Percentil 50
Columnas
0
10000
20000
30000
40000
50000
60000
id
date
time
comp
round
day
venue
result
golesloc
golesvis
opponent
golesesploc
golesespvis
poss
attendance
captain
formation
referee
match report
notes
tirosloc
tirosalarco
distpromtir
tirosliibloc
tirospenloc
tirospenintloc
season
team

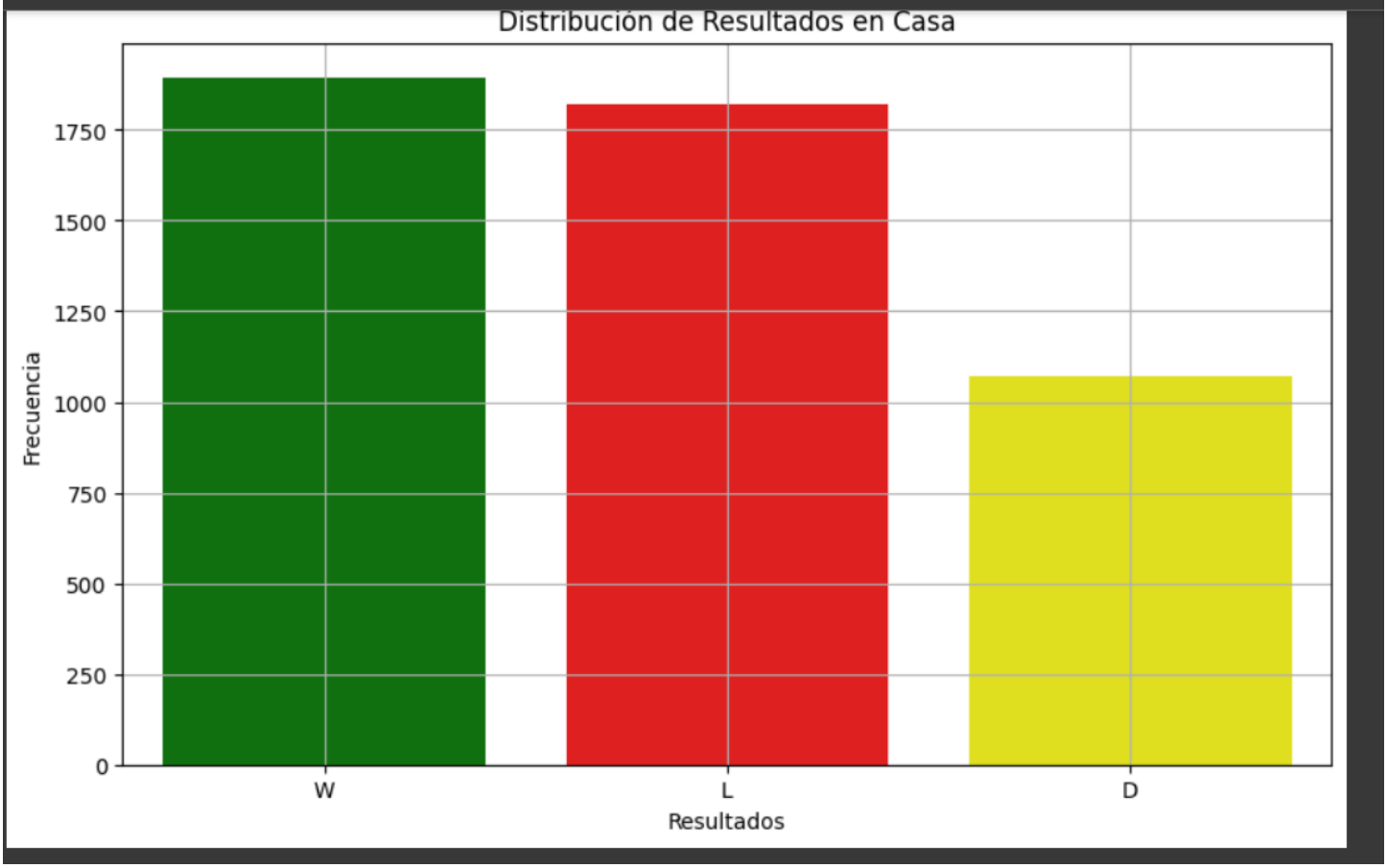

Distribución de la posesión de balón (Poss)
Frecuencia
Posesión (%)
0
100
200
300
400
500
20
30
40
50
60
70
80
Promedio de posesión de balón (λ): 50.43



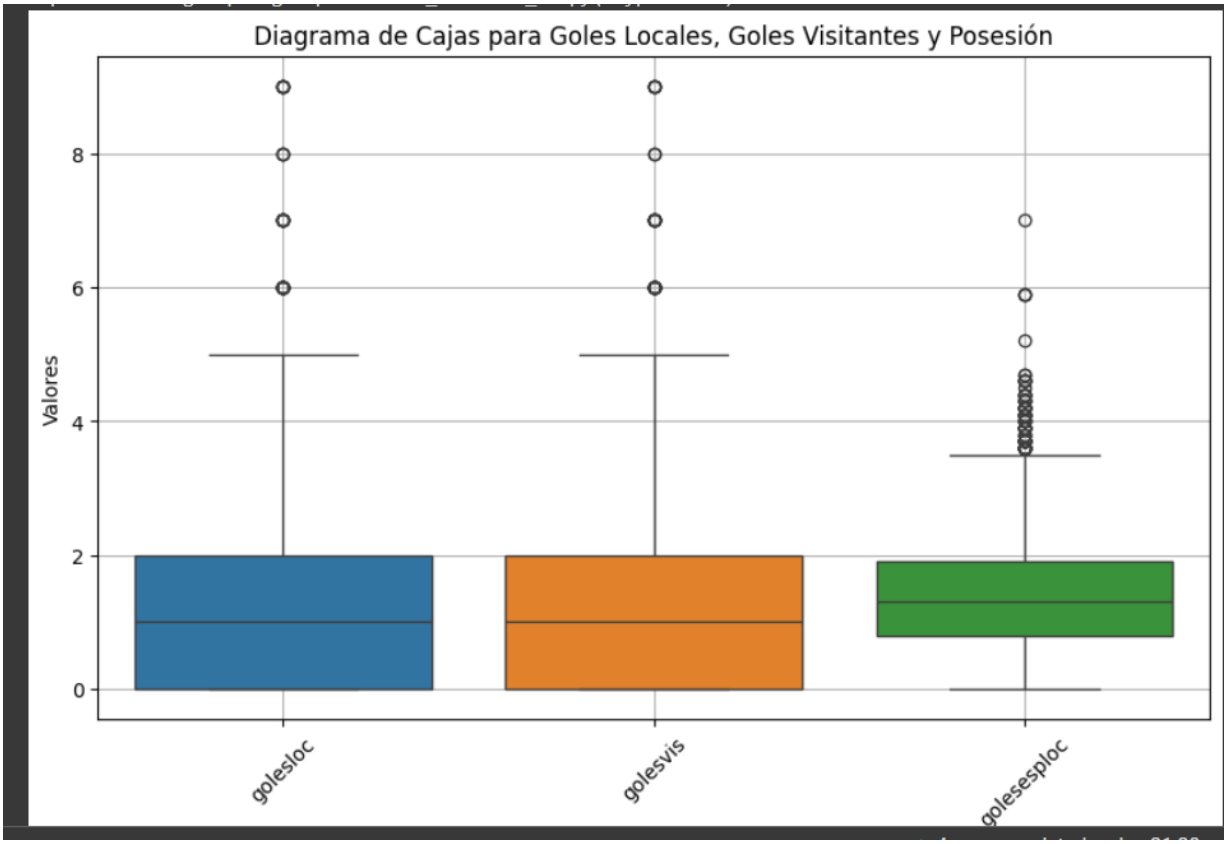
b. De al menos tres columnas seleccionadas por usted indique que datos son relevantes de estas, grafique la misma (puede ser dispersión o mapa de calor, otros), indique al menos 4 características por columna seleccionada.


Distribución de la posesión de balón (Poss)
Frecuencia
Posesión (%)
0
100
200
300
400
500
20
30
40
50
60
70
80

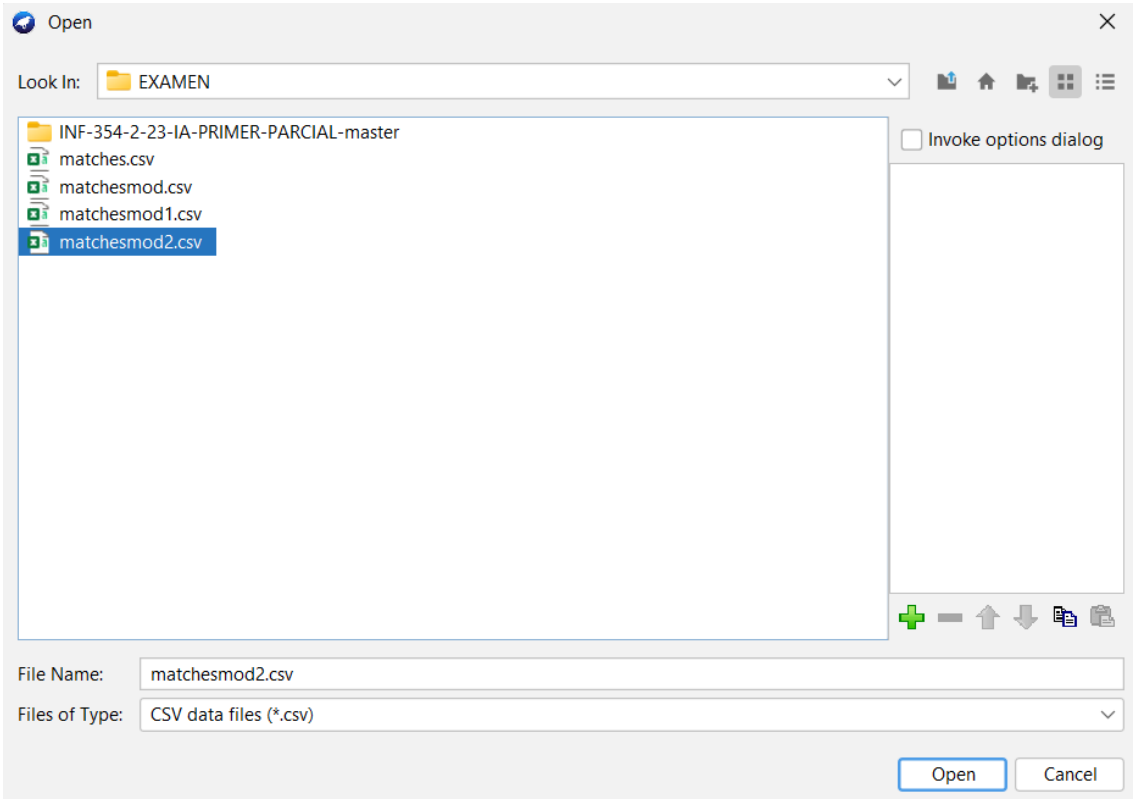

Histogram of Days
Frequency
3000
2500
2000
1500
1000
500
0
Mon
Sun
Sat
Tue
Wed
Fri
Thu
Day

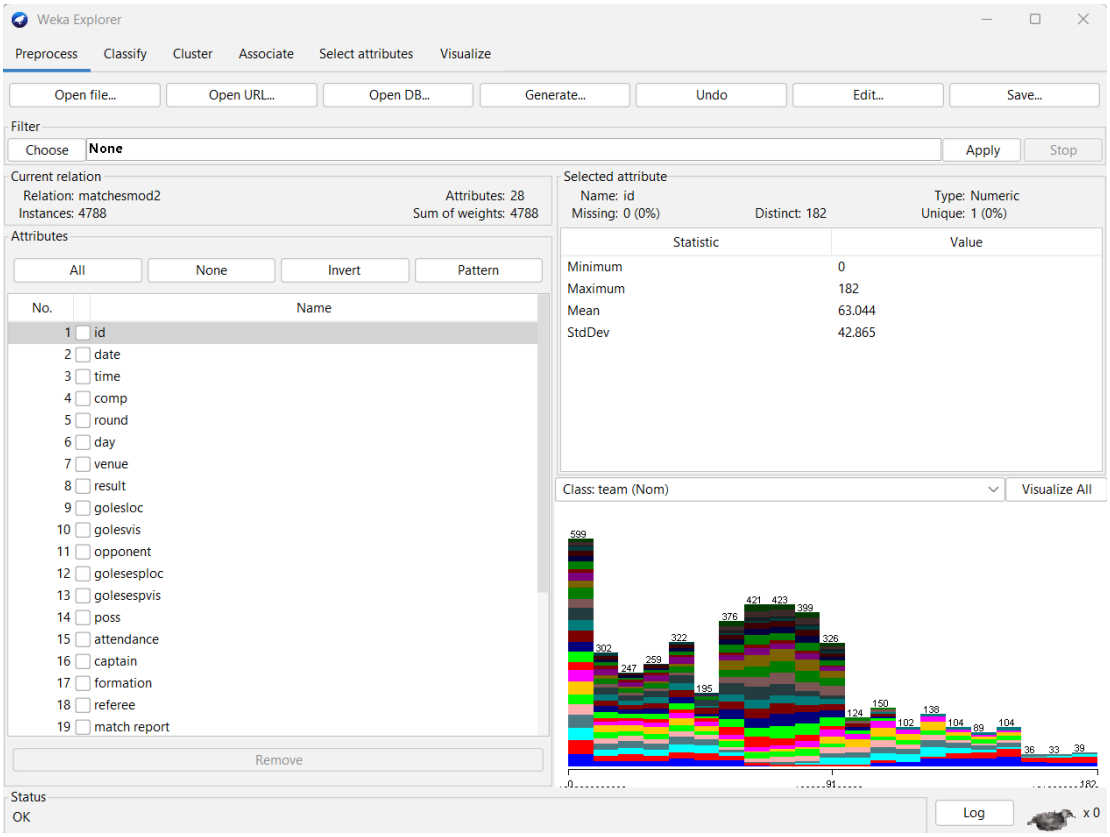

Distribución de Resultados en Casa
Frecuencia
1750
1500
1250
1000
750
500
250
0
W
L
D
Resultados

c. Obteniendo la media, mediana, moda con el uso de librerías, grafique un diagrama de cajas-bigote de al menos 3 columnas. Explique el resultado.



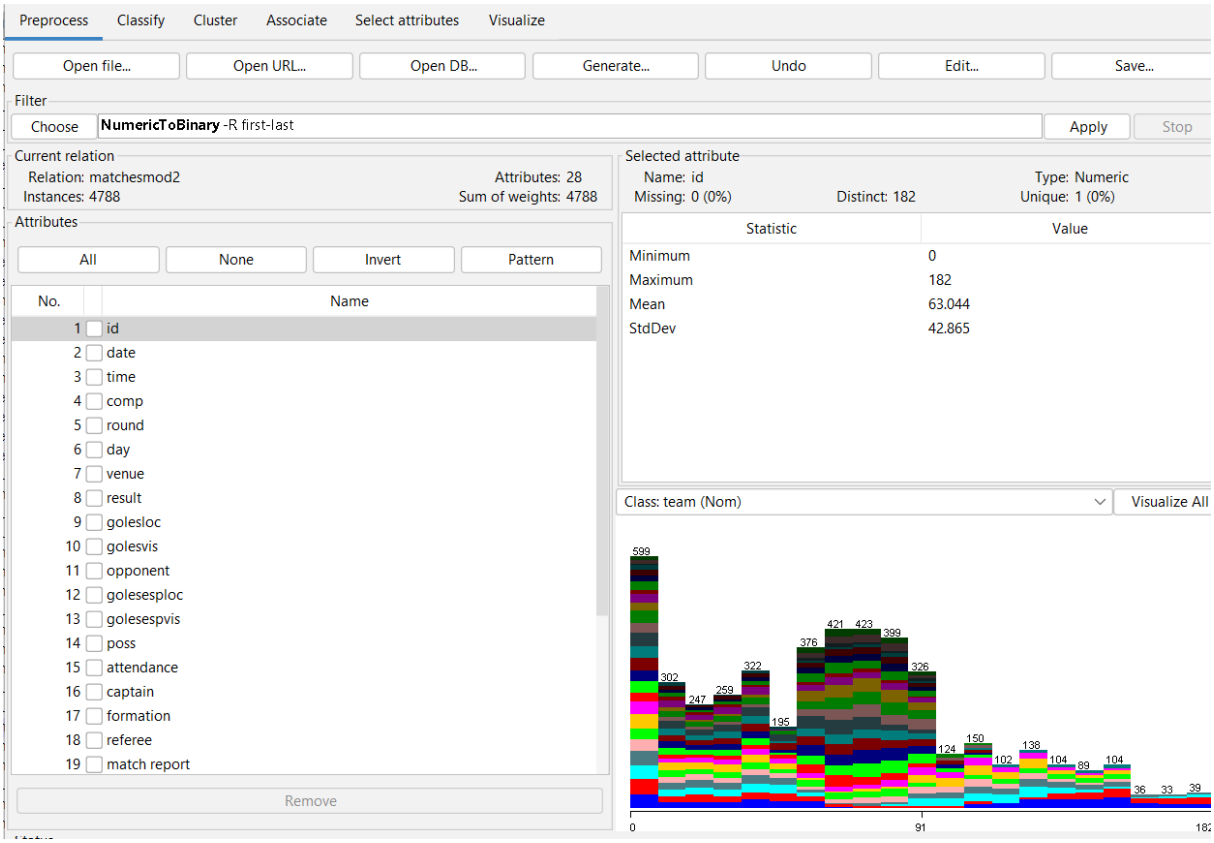
3. Para el preprocesamiento debe usted migrar su dataset de csv u otro formato a arff, una vez realizado ello. Realice un etiqueta onehotencoder, labelencoder, discretización y normalización.





OneHotEncoder:

- Ve a la pestaña “Preprocess”.
- Selecciona el atributo categórico y elige el filtro unsupervised.attribute.NominalToBinary



Preprocess | Classify | Cluster | Associate | Select attributes | Visualize

Open file... | Open URL... | Open DB... | Generate... | Undo | Edit... | Save...

Filter

Choose **NumericToBinary** -R first-last | Apply | Stop

Current relation

Relation: matchesmod2-weka.filters.unsupervised.attribute.Num... | Attributes: 28

Instances: 4788 | Sum of weights: 4788

Attributes

All | None | Invert | Pattern

No.	Name
10	golesvis_binarized
11	opponent
12	golesesploc_binarized
13	golesespvis_binarized
14	poss_binarized
15	attendance_binarized
16	captain
17	formation
18	referee
19	match report
20	notes
21	tirosloc_binarized
22	tirosalarco_binarized
23	distpromtir_binarized
24	tiroslibloc_binarized
25	tirospenloc_binarized
26	tirospenintloc_binarized
27	season_binarized
28	team

Remove

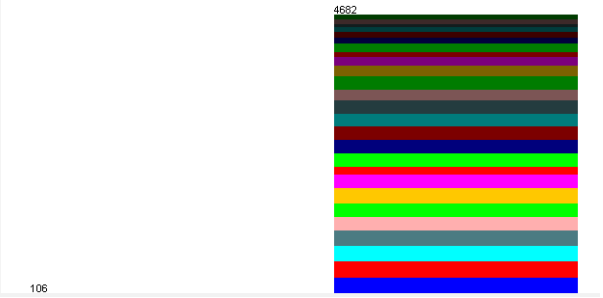
Selected attribute

Name: id_binarized | Type: Nominal

Missing: 0 (0%) | Distinct: 2 | Unique: 0 (0%)

No.	Label	Count	Weight
1	0	106	106
2	1	4682	4682

Class: team (Nom) | Visualize All



Preprocess | Classify | Cluster | Associate | Select attributes | Visualize

Classifier

Choose **NaiveBayesMultinomialText** -P 0 -M 3.0 -norm 1.0 -lnorm 2.0 -stopwords-handler weka.core.stopwords.Null -tokenizer "weka.core.tokenizers.WordTokenizer -delimiters \" \"

Test options

- Use training set
- Supplied test set | Set...
- Cross-validation | Folds | 10
- Percentage split | % | 66

More options...

(Nom) team

Start | Stop

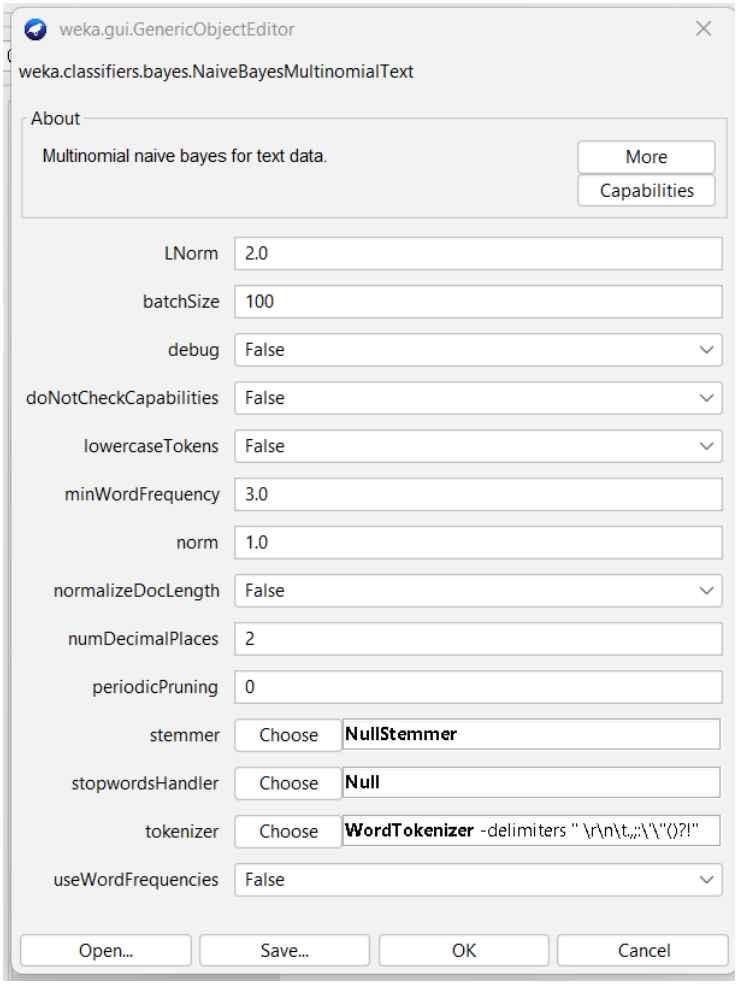
Result list (right-click for options)

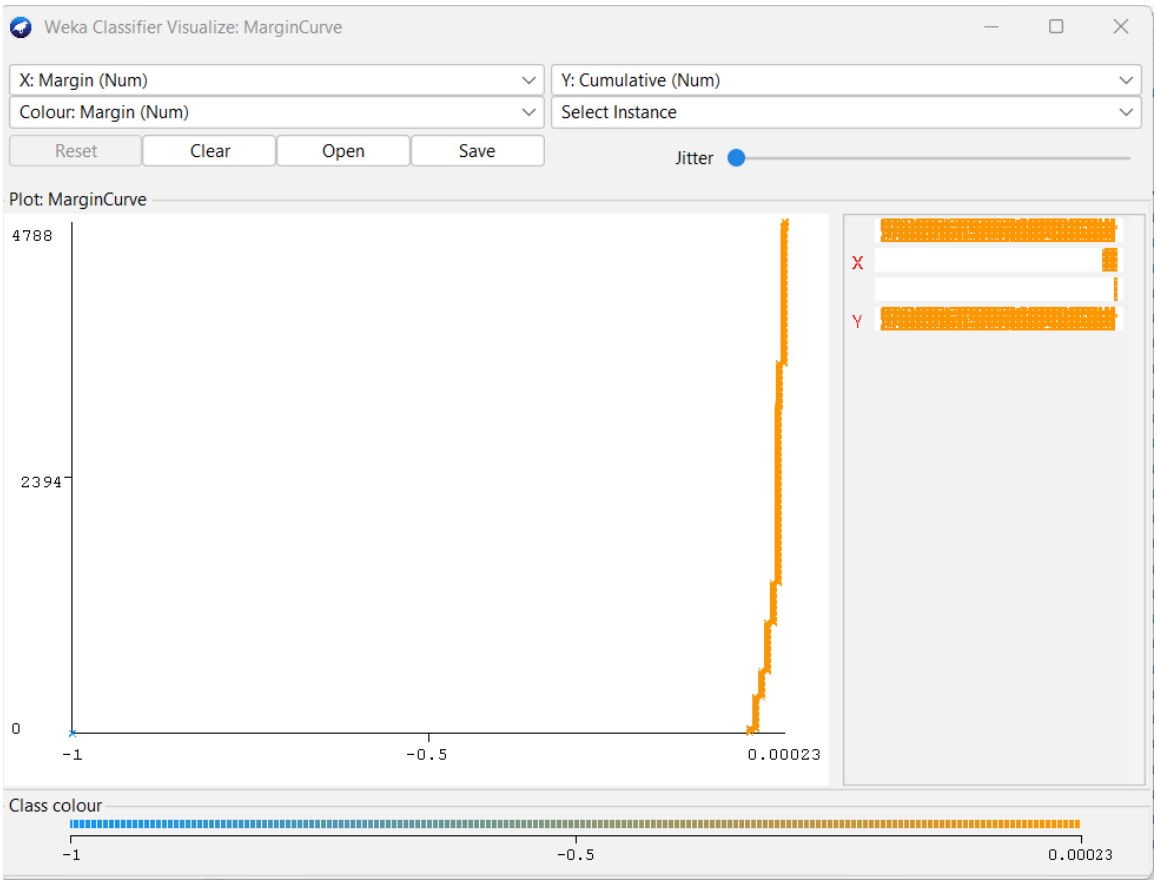
01:10:37 - bayes.NaiveBayesMultinomialText

01:10:44 - bayes.NaiveBayesMultinomialText

Classifier output

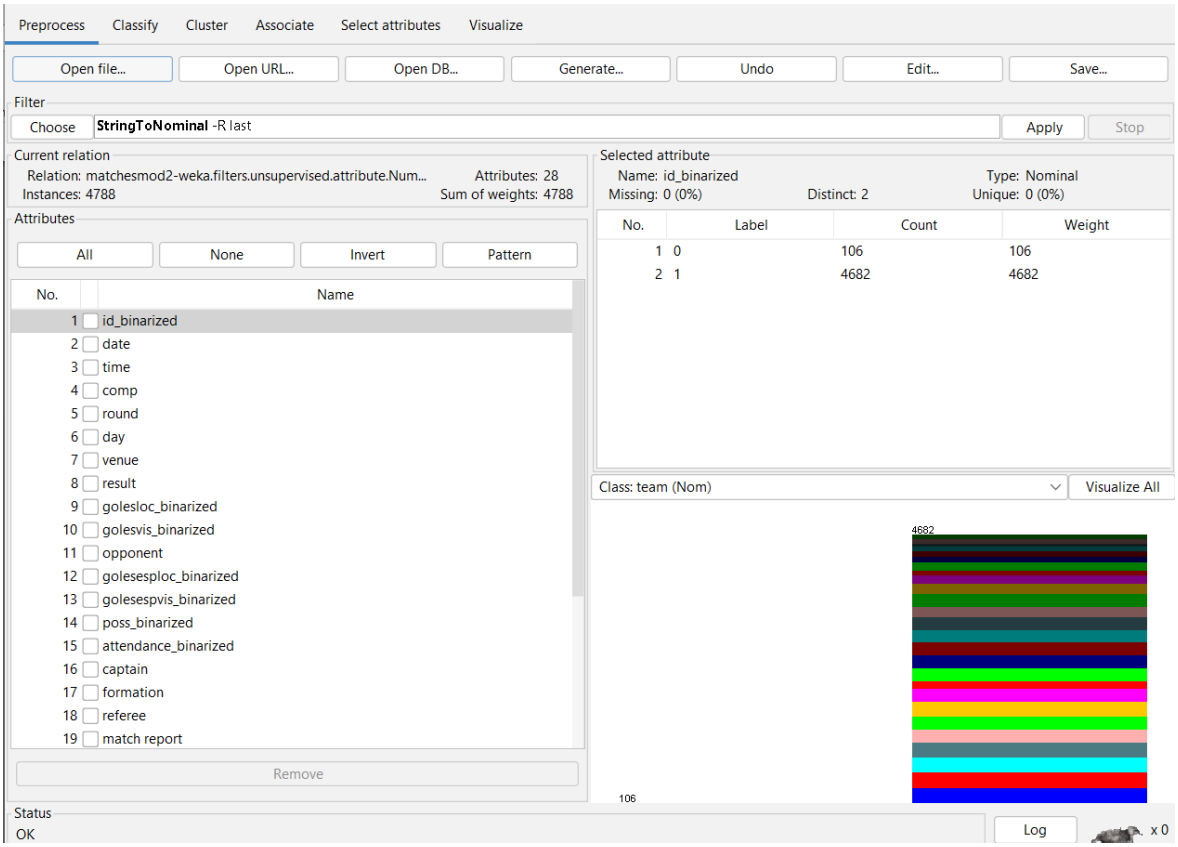
```
=== Run information ===

Scheme:       weka.classifiers.bayes.NaiveBayesMultinomialText -P 0 -M 3.0 -norm 1.0 -lnorm 2.0 -stop
Relation:     matchesmod2-weka.filters.unsupervised.attribute.NumericToBinary-Rfirst-last
Instances:    4788
Attributes:   28
              id_binarized
              date
              time
              comp
              round
              day
              venue
              result
              golesloc_binarized
              golesvis_binarized
              opponent
              golesesploc_binarized
              golesespvis_binarized
              poss_binarized
              attendance_binarized
              captain
              formation
              referee
              match report
              notes
              tirosloc_binarized
              tirosalarco_binarized
              distpromtir_binarized
              tiroslibloc_binarized
              tirospenloc_binarized
              tirospenintloc_binarized
              season_binarized
```

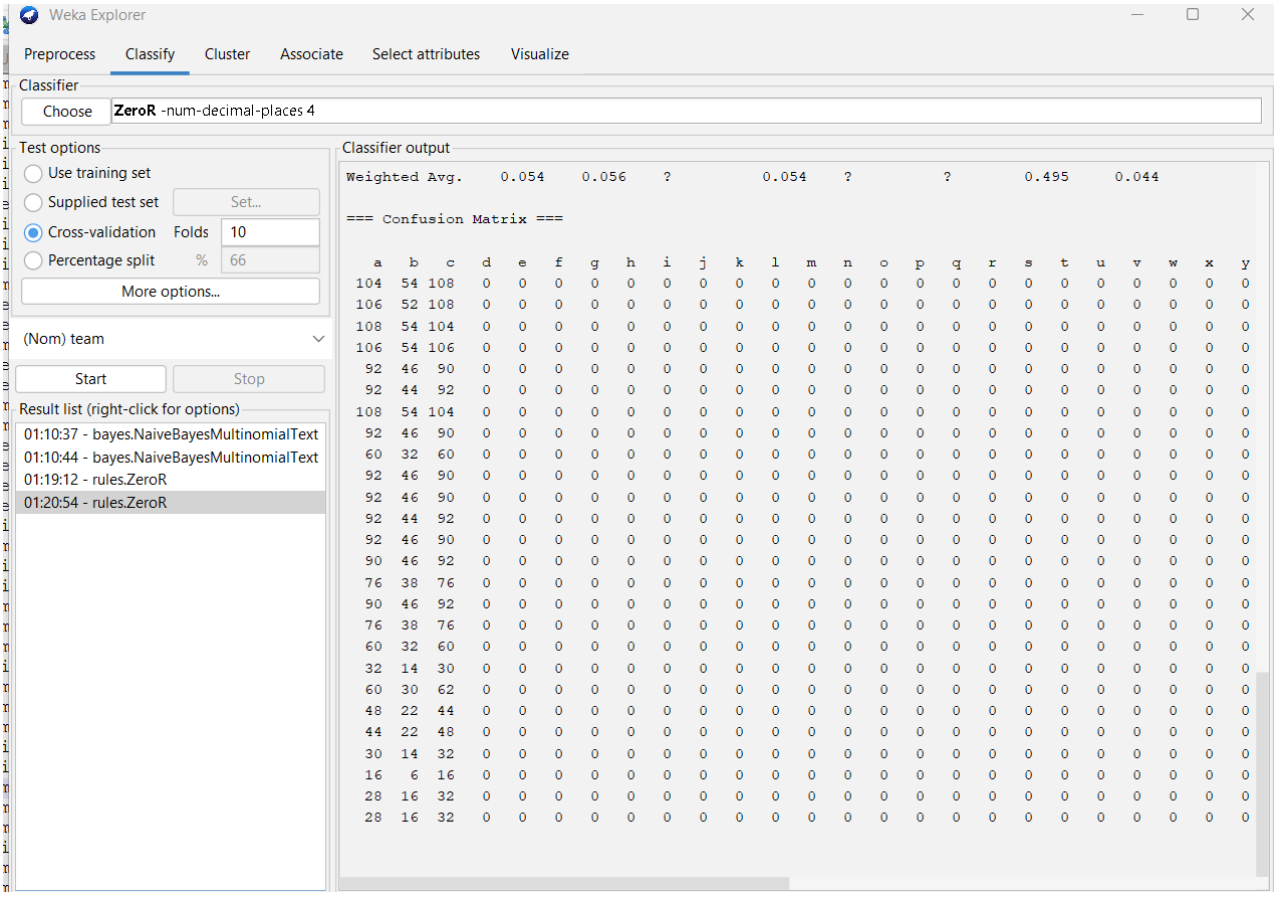

weka.gui.GenericObjectEditor
weka.classifiers.bayes.NaiveBayesMultinomialText
About
Multinomial naive bayes for text data.
More
Capabilities
LNorm
2.0
batchSize
100
debug
False
doNotCheckCapabilities
False
lowercaseTokens
False
minWordFrequency
3.0
norm
1.0
normalizeDocLength
False
numDecimalPlaces
2
periodicPruning
0
stemmer
Choose
NullStemmer
stopwordsHandler
Choose
Null
tokenizer
Choose
WordTokenizer -delimiters " \r\n\t.,;:\'\"()?!"
useWordFrequencies
False
Open...
Save...
OK
Cancel

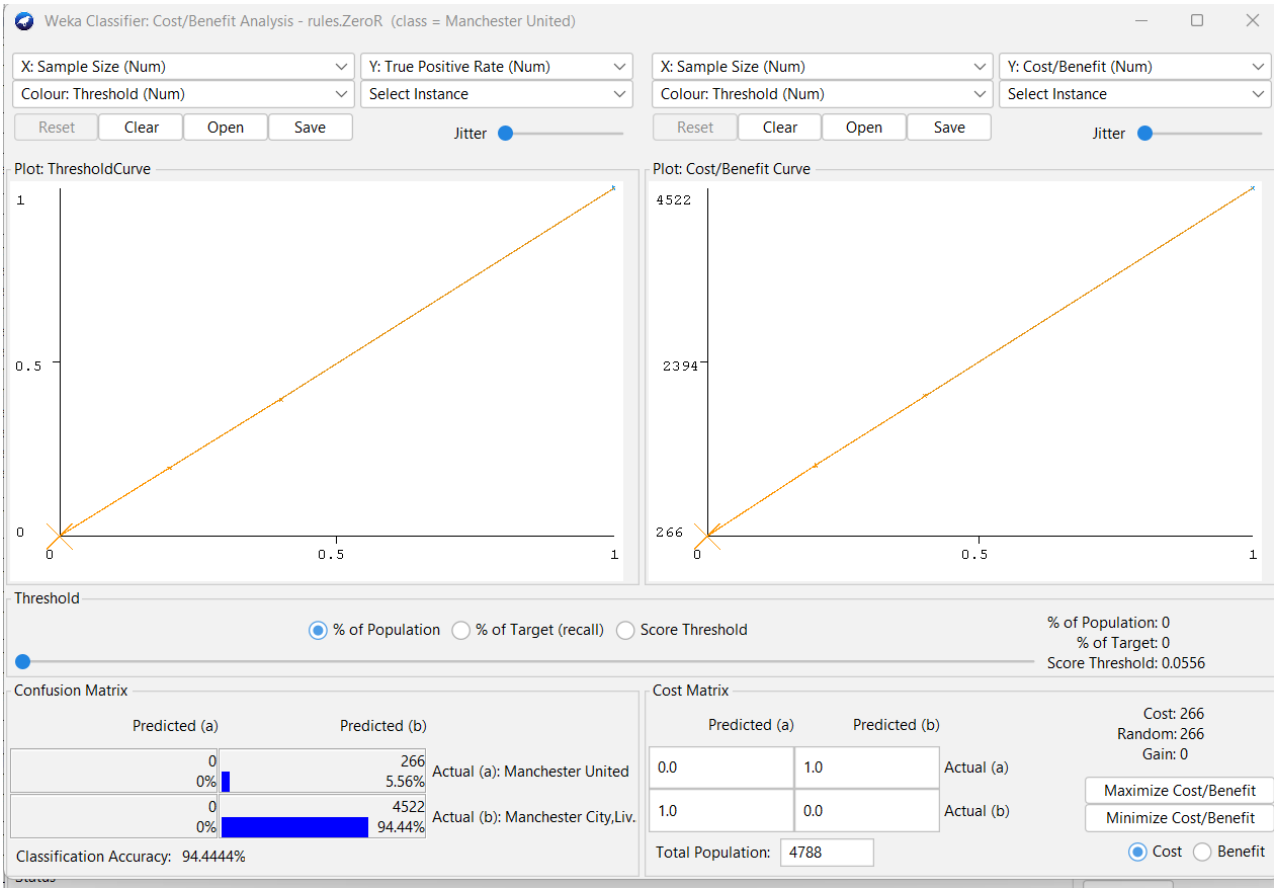

Weka Classifier Visualize: MarginCurve
X: Margin (Num)
Y: Cumulative (Num)
Colour: Margin (Num)
Select Instance
Reset
Clear
Open
Save
Jitter
Plot: MarginCurve
4788
2394
0
-1
-0.5
0.00023
X
Y
Class colour
-1
-0.5
0.00023

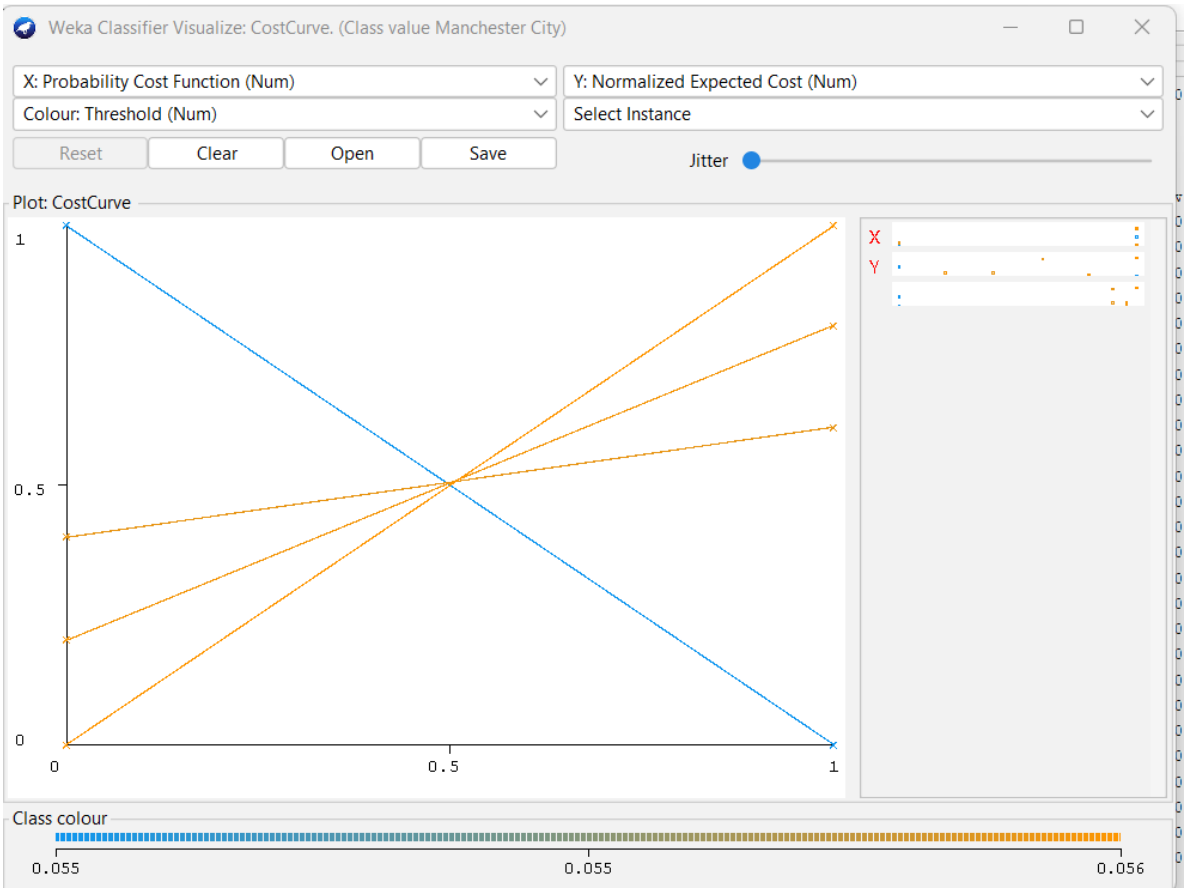
LabelEncoder:

- Aplica unsupervised.attribute.StringToNominal para convertir cadenas en números.



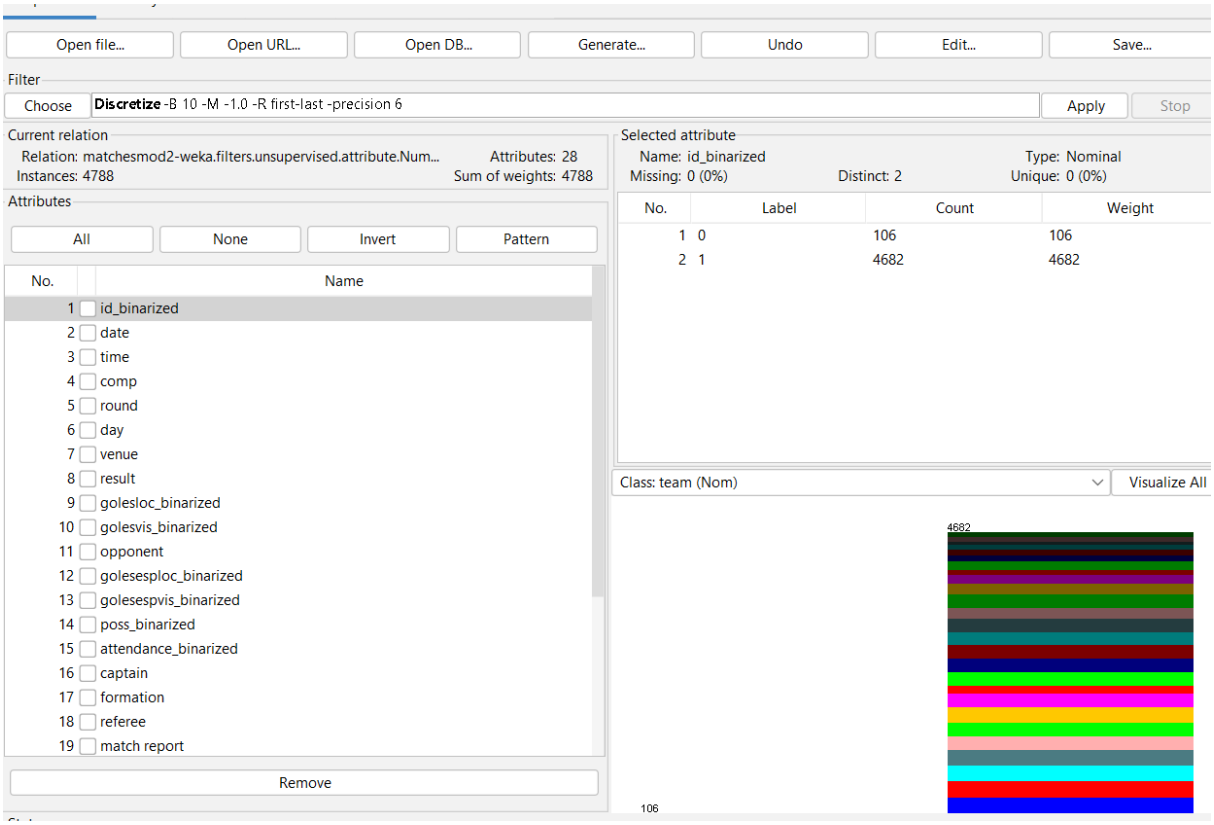


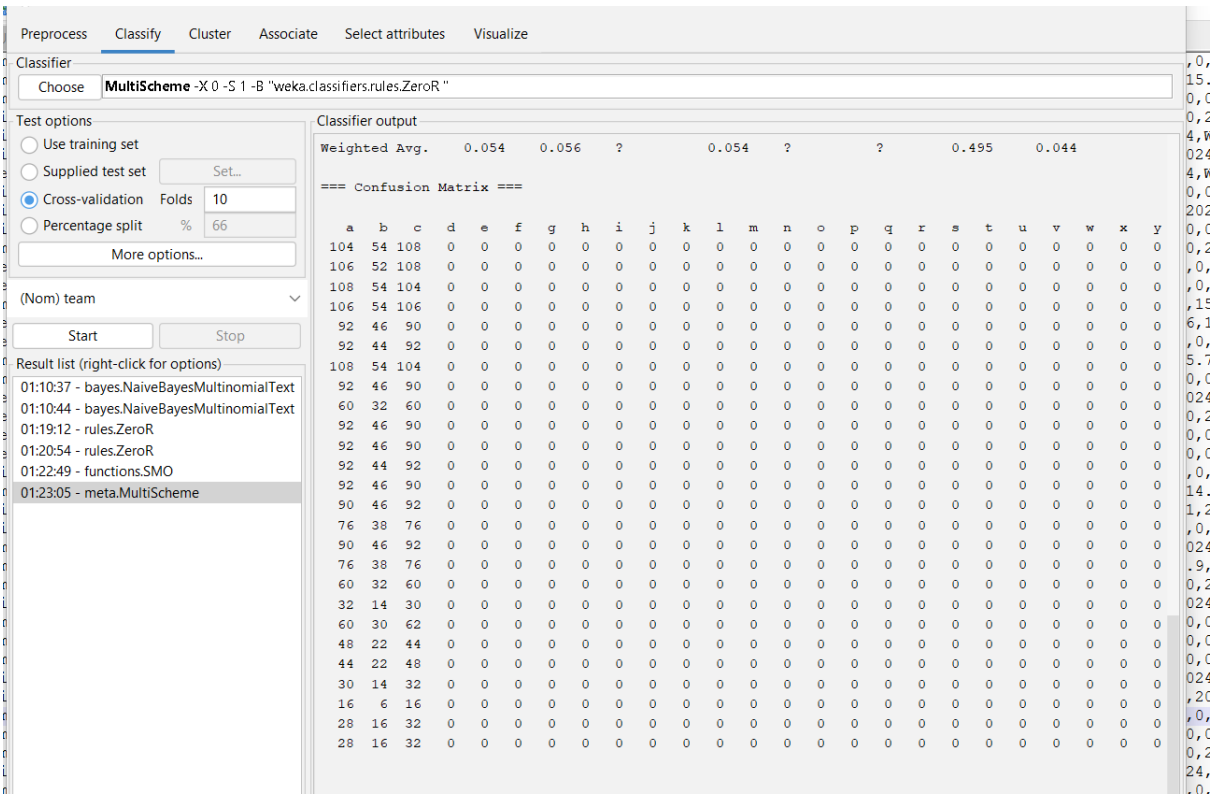


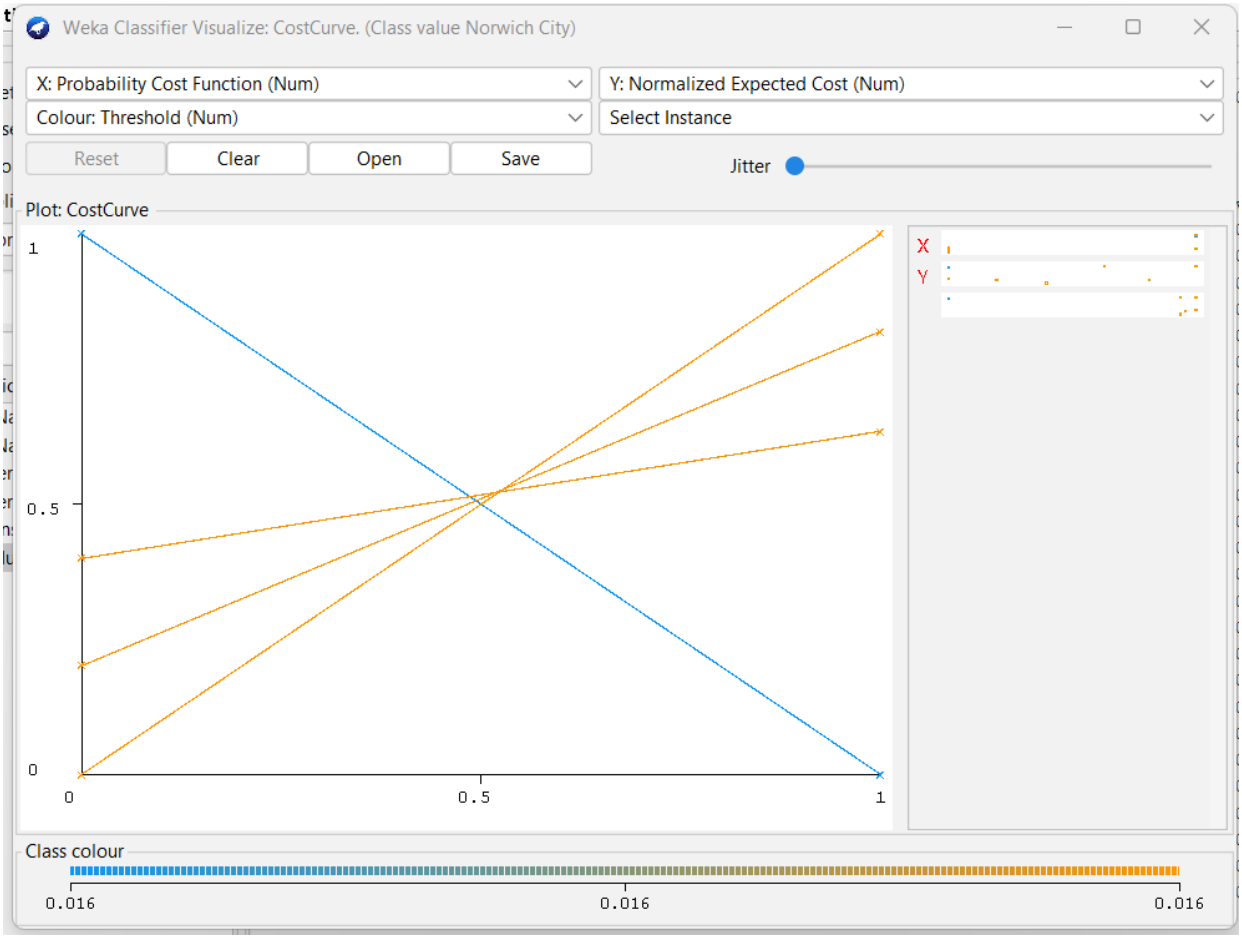


Discretización:

- Utiliza unsupervised.attribute.Discretize.







Normalización:

- Usa el filtro unsupervised.attribute.Normalize.

Weka Explorer

Preprocess | Classify | Cluster | Associate | Select attributes | Visualize

Open file... | Open URL... | Open DB... | Generate... | Undo | Edit... | Save...

Filter

Choose **Normalize** -S 1.0 -T 0.0 | Apply | Stop

Current relation

Relation: matchesmod2-weka.filters.unsupervised.attribute.Num... Attributes: 28

Instances: 4788 Sum of weights: 4788

Attributes

All | None | Invert | Pattern

No.	Name
1	id_binarized
2	date
3	time
4	comp
5	round
6	day
7	venue
8	result
9	golesloc_binarized
10	golesvis_binarized
11	opponent
12	golesesploc_binarized
13	golesespvis_binarized
14	poss_binarized
15	attendance_binarized
16	captain
17	formation
18	referee
19	match report

Remove

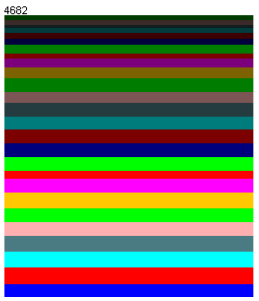
Selected attribute

Name: id_binarized Type: Nominal

Missing: 0 (0%) Distinct: 2 Unique: 0 (0%)

No.	Label	Count	Weight
1	0	106	106
2	1	4682	4682

Class: team (Nom) | Visualize All



Classifier

Choose **ZeroR**

Test options

Use training set

Supplied test set | Set...

Cross-validation Folds 10

Percentage split % 66

More options...

(Nom) team

Start | Stop

Result list (right-click for options)

01:10:37 - bayes.NaiveBayesMultinomialText

01:10:44 - bayes.NaiveBayesMultinomialText

01:19:12 - rules.ZeroR

01:20:54 - rules.ZeroR

01:22:49 - functions.SMO

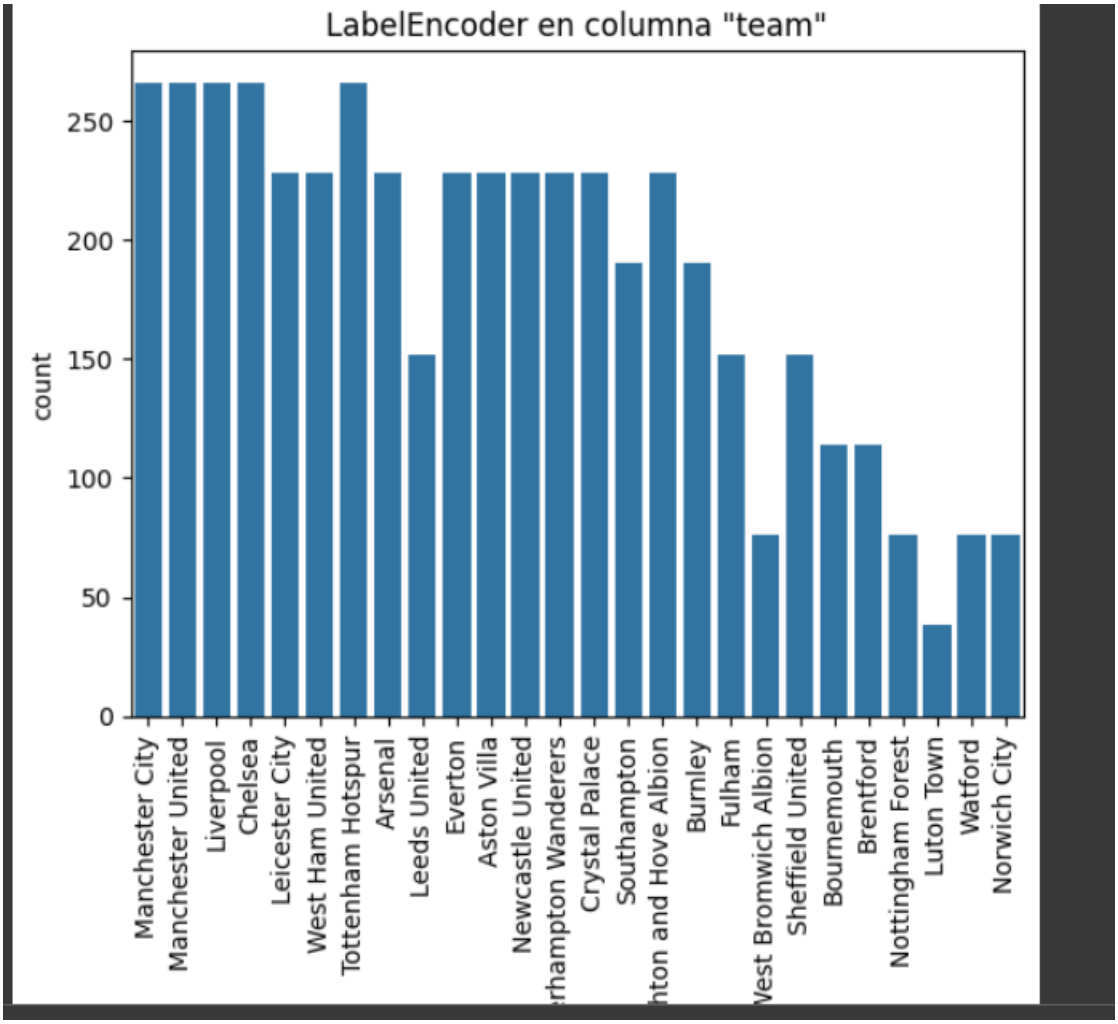
01:23:05 - meta.MultiScheme

01:26:48 - rules.ZeroR

Classifier output

```
                 0.000    0.000    ?          0.000    ?          ?          0.489    0.023    Brentf
                 0.000    0.000    ?          0.000    ?          ?          0.484    0.015    Nottin
                 0.000    0.000    ?          0.000    ?          ?          0.479    0.008    Luton
                 0.000    0.000    ?          0.000    ?          ?          0.484    0.015    Watfor
                 0.000    0.000    ?          0.000    ?          ?          0.484    0.015    Norwic
Weighted Avg.    0.054    0.056    ?          0.054    ?          ?          0.495    0.044

=== Confusion Matrix ===

   a   b   c   d   e   f   g   h   i   j   k   l   m   n   o   p   q   r   s   t   u   v   w   x   y
 104  54 108   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 106  52 108   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 108  54 104   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 106  54 106   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  92  46  90   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  92  44  92   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
 108  54 104   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  92  46  90   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  60  32  60   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  92  46  90   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  92  46  90   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  92  44  92   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  92  46  90   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  90  46  92   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  76  38  76   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  90  46  92   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  76  38  76   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  60  32  60   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  32  14  30   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  60  30  62   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  48  22  44   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  44  22  48   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
  30  14  32   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0
```

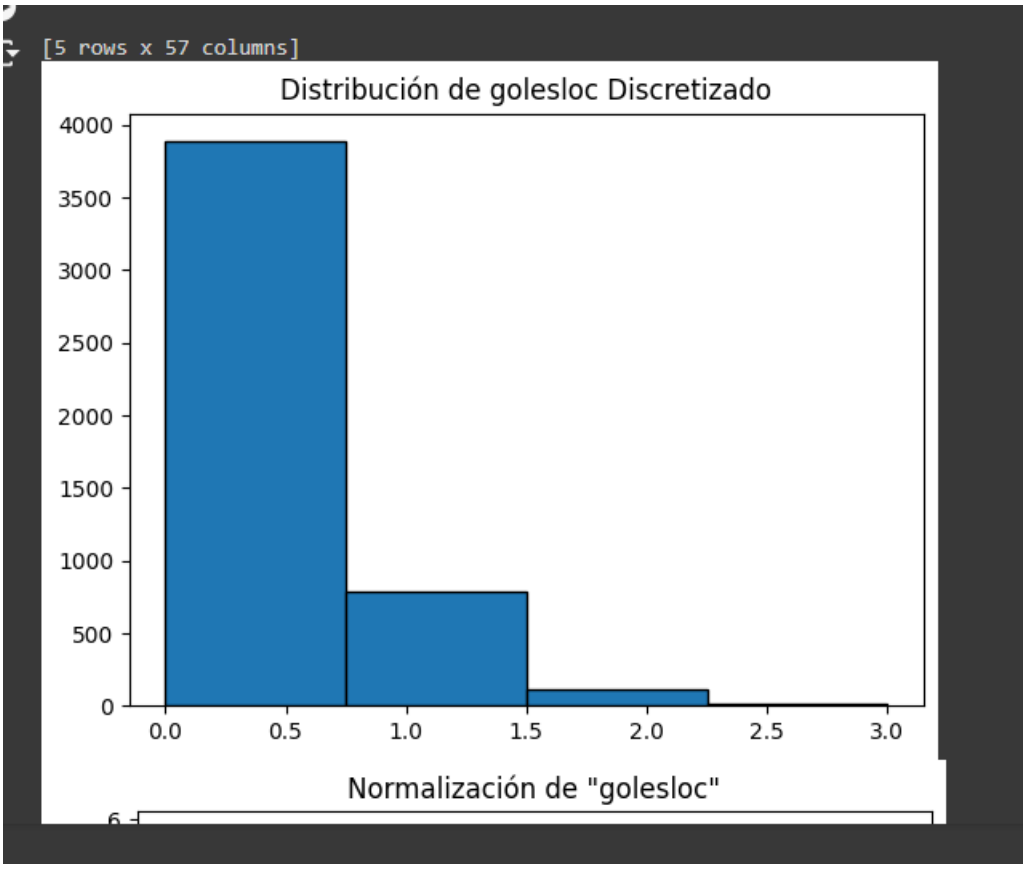
4. Con el uso de librerías realiza en Python los mismos preprocesamiento del punto 3.

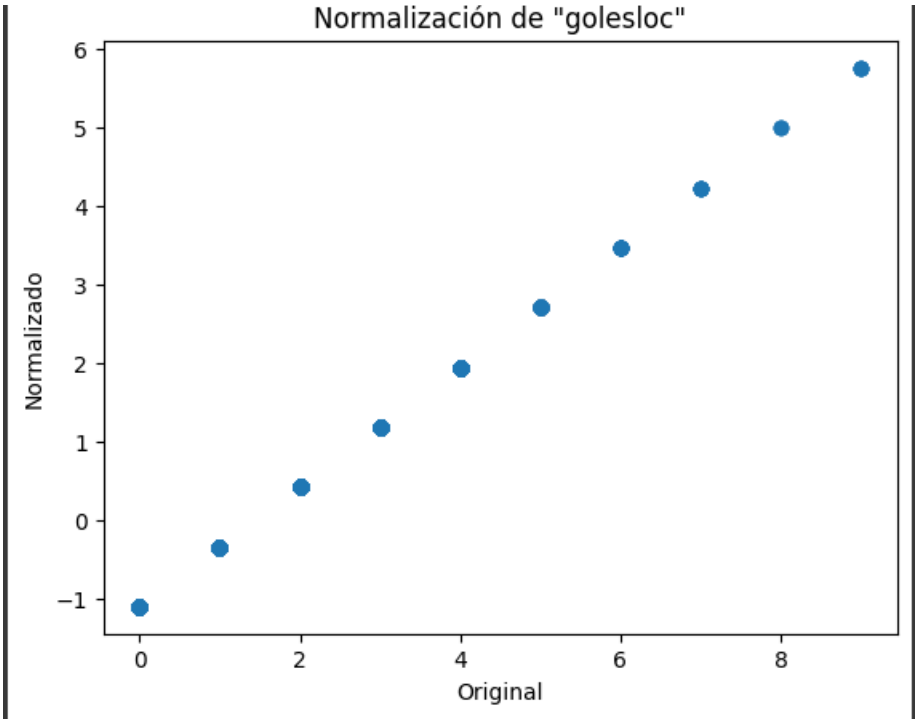


	id	date	time	comp	round	day	venue	\
0	0	9/21/2020	20:15 (21:15)	Premier League	Matchweek 2	Mon	Away	
1	2	9/27/2020	16:30 (17:30)	Premier League	Matchweek 3	Sun	Home	
2	4	10/3/2020	17:30 (18:30)	Premier League	Matchweek 4	Sat	Away	
3	5	10/17/2020	17:30 (18:30)	Premier League	Matchweek 5	Sat	Home	
4	7	10/24/2020	12:30 (13:30)	Premier League	Matchweek 6	Sat	Away	

	result	golesloc	golesvis	...	team_18	team_19	team_20	team_21	team_22	\
0	W	3	1	...	0.0	0.0	0.0	0.0	0.0	
1	L	2	5	...	0.0	0.0	0.0	0.0	0.0	
2	D	1	1	...	0.0	0.0	0.0	0.0	0.0	
3	W	1	0	...	0.0	0.0	0.0	0.0	0.0	
4	D	1	1	...	0.0	0.0	0.0	0.0	0.0	

	team_23	team_24	team_25	golesloc_discretizado	golesloc_normalizado
0	0.0	0.0	0.0	1.0	1.182482
1	0.0	0.0	0.0	0.0	0.420576
2	0.0	0.0	0.0	0.0	-0.341330
3	0.0	0.0	0.0	0.0	-0.341330
4	0.0	0.0	0.0	0.0	-0.341330





6. Con el uso de EXCEL, realice en el algoritmo genético de $f(x)=x^{2x}-1$. Al menos tres generaciones. Automatice el cálculo.

	1a Generacion													
	Selección			Cruce			Mutacion							
	Poblacion	funcion de evaluacion		Punto de division=4										
Original	x1	f(x)=(x^(2x))-1	fenotipo					Poblacion final		UMSA-DAT245 IA: UNIV. MOISES MARTIN CONDORI YUJRA				
10	13	9.17333E+28	00001101	'0000	'1101	'00001100	00001110	14						
13	12	7.94968E+25	00001100	'0000	'1100	'00001101	00001111	15						
3	11	8.14027E+22	00001011	'0000	'1011	'00001010	00001000	8						
4	10	1E+20	00001010	'0000	'1010	'00001011	00001001	9						
8	9	1.50095E+17	00001001	'0000	'1001	'00001000	00001010	10						
1	8	2.81475E+14	00001000	'0000	'1000	'00001001	00001011	11						
12	7	6.78223E+11	00000111	'0000	'0111	'00000110	00000100	4						
5	6	2176782335	00000110	'0000	'0110	'00000111	00000101	5						
7	5	9765624	00000101	'0000	'0101	'00000100	00000110	6						
6	4	65535	00000100	'0000	'0100	'00000101	00000111	7						
9	3	728	00000011	'0000	'0011	'00000001	00000011	3						
11	1	0	00000001	'0000	'0001	'00000011	00000001	1						
	2a Generacion													
	Selección			Cruce			Mutacion							
	Poblacion	funcion de evaluacion		Punto de division=4										
	x1	f(x)=(x^(2x))-1	fenotipo					Poblacion final						
	15	1.91751E+35	00001111	0000	1111	00001110	00001111	15						
	14	1.23477E+32	00001110	0000	1110	00001111	00001011	11						
	11	8.14027E+22	00001011	0000	1011	00001011	00001001	9						
	10	1E+20	00001010	0000	1010	00001011	00001110	14						
	9	1.50095E+17	00001001	0000	1001	00001000	00001000	8						
	8	2.81475E+14	00001000	0000	1000	00001001	00001101	13						
	7	6.78223E+11	00000111	0000	0111	00000110	00000110	6						
	6	2176782335	00000110	0000	0110	00000111	00000111	7						
	5	9765624	00000101	0000	0101	00000100	00000100	4						
	4	65535	00000100	0000	0100	00000101	00000101	5						
	3	728	00000011	0000	0011	00000001	00000001	1						
	1	0	00000001	0000	0001	00000011	00000010	2						
	2a Generacion													
	Selección			Cruce			Mutacion							
	Poblacion	funcion de evaluacion		Punto de division=4										
	x1	f(x)=(x^(2x))-1	fenotipo					Poblacion final						
15	15	1.91751E+35	00001111	0000	1111	00001110	00001111	15						
11	14	1.23477E+32	00001110	0000	1110	00001111	00001011	11						
9	13	9.17333E+28	00001101	0000	1101	00001011	00001001	9						
14	11	8.14027E+22	00001011	0000	1011	00001101	00001100	12						

7. Programe el anterior problema en Python con

   a. El uso de DEAP

```
import random
from deap import base, creator, tools, algorithms
import numpy as np

# 1. Definir el problema de maximización con la función objetivo f(x) = (x^(2x)) - 1
creator.create("FitnessMax", base.Fitness, weights=(1.0,))
creator.create("Individual", list, fitness=creator.FitnessMax)

# 2. Inicialización de individuos (población de números enteros aleatorios)
def init_individual():
    return [random.randint(0, 5)]  # Individuos entre 0 y 5

toolbox = base.Toolbox()
toolbox.register("individual", tools.initIterate, creator.Individual,
init_individual)
toolbox.register("population", tools.initRepeat, list, toolbox.individual)

# 3. Definir la función objetivo f(x) = (x^(2x)) - 1
def eval_function(individual):
    x = individual[0]
    # Asegurarse de que la función no cause desbordamientos numéricos
    try:
        result = (x**(2 * x)) - 1
```

```
    except OverflowError:
        result = float('inf')  # Penaliza el valor si hay un desbordamiento
    return (result,)  # Devuelve una tupla con un único valor numérico

# 4. Registrar las operaciones genéticas
toolbox.register("evaluate", eval_function)
toolbox.register("mate", tools.cxBlend, alpha=0.5)    # Operador de cruce (blend
crossover)
toolbox.register("mutate", tools.mutGaussian, mu=0, sigma=1, indpb=0.2)  # Operador
de mutación
toolbox.register("select", tools.selTournament, tournsize=3)  # Selección por torneo

# 5. Algoritmo genético con al menos 3 generaciones
def run_ga():
    # Inicializar la población
    population = toolbox.population(n=10)  # Población de 10 individuos

    # Configurar registros para guardar estadísticas
    stats = tools.Statistics(lambda ind: ind.fitness.values)
    stats.register("avg", np.mean)
    stats.register("std", np.std)
    stats.register("min", np.min)
    stats.register("max", np.max)

    # Hall of Fame para guardar los mejores individuos
    hof = tools.HallOfFame(1)

    # Ejecutar el algoritmo genético por 3 generaciones
    print("Iniciando algoritmo genético...\n")
    result = algorithms.eaSimple(population, toolbox, cxpb=0.5, mutpb=0.2, ngen=5,
stats=stats, halloffame=hof, verbose=True)

    # Mostrar el mejor individuo
    print("\nMejor individuo encontrado:", hof[0])
    print("Fitness del mejor individuo:", hof[0].fitness.values[0])

    # Mostrar resumen estadístico
    print("\nResumen estadístico por generación:")
    for gen, log in enumerate(result[1]):
        print(f"Generación {gen + 1}: Promedio = {log['avg']}, Mejor = {log['max']}")

run_ga()
```

```
Iniciando algoritmo genético...

gen     nevals  avg     std     min     max
0       10      13184.3 26176.2 0       65535
1       4       2.47812e+09     7.43414e+09     15      2.47805e+10
2       7       9.39101e+09     2.12388e+10     28192.5 6.91283e+10
3       5       1.8885e+10      2.63759e+10     65535   6.91283e+10
4       6       5.46216e+10     2.52507e+10     4.83979e+07     7.70317e+10
5       5       6.53737e+10     2.20653e+10     4.83979e+07     7.70317e+10

Mejor individuo encontrado: [6.627372060315224]
Fitness del mejor individuo: 77031696687.59828

Resumen estadístico por generación:
Generación 1: Promedio = 13184.3, Mejor = 65535.0
Generación 2: Promedio = 2478124468.5480804, Mejor = 24780540785.55872
Generación 3: Promedio = 9391007162.38507, Mejor = 69128295188.49797
Generación 4: Promedio = 18885026475.535004, Mejor = 69128295188.49797
Generación 5: Promedio = 54621568412.08329, Mejor = 77031696687.59828
Generación 6: Promedio = 65373722754.57212, Mejor = 77031696687.59828
```

b. Sin el uso de DEAP

```python
import random

# 1. Definir la función objetivo f(x) = (x^(2*x)) - 1
def eval_function(x):
    return (x**(2 * x)) - 1

# 2. Inicialización de la población
def init_population(size, lower_bound, upper_bound):
    population = [random.randint(lower_bound, upper_bound) for _ in range(size)]
    return population

# 3. Selección por torneo (escoge al mejor de un grupo aleatorio)
def tournament_selection(population, fitnesses, k=3):
    selected = random.sample(list(zip(population, fitnesses)), k)
    return max(selected, key=lambda ind_fit: ind_fit[1])[0]

# 4. Cruce (mezcla de genes de dos padres)
def crossover(parent1, parent2):
    alpha = 0.5
    return int(alpha * parent1 + (1 - alpha) * parent2)

# 5. Mutación (modificar el individuo ligeramente)
def mutate(individual, mutation_rate=0.1, lower_bound=0, upper_bound=10):
    if random.random() < mutation_rate:
        return random.randint(lower_bound, upper_bound)
    return individual
```

```python
# 6. Algoritmo Genético
def genetic_algorithm(generations=3, population_size=10, lower_bound=0,
upper_bound=5):
    # Inicializar población
    population = init_population(population_size, lower_bound, upper_bound)

    # Iterar a través de generaciones
    for gen in range(generations):
        print(f"\nGeneración {gen + 1}")

        # Evaluar aptitud de cada individuo
        fitnesses = [eval_function(ind) for ind in population]

        # Mostrar la población y sus aptitudes
        for i, (ind, fit) in enumerate(zip(population, fitnesses)):
            print(f"Individuo {i + 1}: {ind}, Fitness: {fit}")

        # Crear la nueva población
        new_population = []
        for _ in range(population_size // 2):
            # Seleccionar dos padres mediante torneo
            parent1 = tournament_selection(population, fitnesses)
            parent2 = tournament_selection(population, fitnesses)

            # Cruzar padres para crear descendencia
            child1 = crossover(parent1, parent2)
            child2 = crossover(parent2, parent1)

            # Mutar descendencia
            child1 = mutate(child1)
            child2 = mutate(child2)

            new_population.extend([child1, child2])

        # Reemplazar la población anterior con la nueva
        population = new_population

    # Evaluación final de la última generación
    final_fitnesses = [eval_function(ind) for ind in population]
    best_individual = population[final_fitnesses.index(max(final_fitnesses))]
    print(f"\nMejor individuo: {best_individual} con fitness:
{max(final_fitnesses)}")

# Ejecutar el algoritmo genético
genetic_algorithm()
```

```
Generación 1
Individuo 1: 3, Fitness: 728
Individuo 2: 0, Fitness: 0
Individuo 3: 1, Fitness: 0
Individuo 4: 5, Fitness: 9765624
Individuo 5: 2, Fitness: 15
Individuo 6: 1, Fitness: 0
Individuo 7: 5, Fitness: 9765624
Individuo 8: 5, Fitness: 9765624
Individuo 9: 2, Fitness: 15
Individuo 10: 2, Fitness: 15

Generación 2
Individuo 1: 3, Fitness: 728
Individuo 2: 3, Fitness: 728
Individuo 3: 3, Fitness: 728
Individuo 4: 3, Fitness: 728
Individuo 5: 7, Fitness: 678223072848
Individuo 6: 0, Fitness: 0
Individuo 7: 3, Fitness: 728
Individuo 8: 7, Fitness: 678223072848
Individuo 9: 2, Fitness: 15
Individuo 10: 1, Fitness: 0
```

```
Generación 3
Individuo 1: 5, Fitness: 9765624
Individuo 2: 5, Fitness: 9765624
Individuo 3: 5, Fitness: 9765624
Individuo 4: 5, Fitness: 9765624
Individuo 5: 3, Fitness: 728
Individuo 6: 3, Fitness: 728
Individuo 7: 5, Fitness: 9765624
Individuo 8: 5, Fitness: 9765624
Individuo 9: 5, Fitness: 9765624
Individuo 10: 5, Fitness: 9765624

Mejor individuo: 5 con fitness: 9765624
```

8. De las características de altura, peso y talla; realice su propio dataset determinando cuál será su clase. Realice el cálculo de la entropía y ganancia de información.

2 3 5 3 5 3 3 4050 4034 Como relleno?

UMSA-DAT245-IA-UNIV. CONDORI YUJRA MOISES MARTIN

X_1	X_2	X_3	X_4	X_5	X_6	X_7	y
Genero (Masculino, Femenino	**Altura** (Alto, Mediano,Bajo)	**Peso** (Gordo, Normal, Flaco)	**Talla** (S, M, L)	**Competencias** (Altas, Bajas, Medias)	**NiveldeActividadFisica** (Sedentario, activo, MuyActivo	**ColordeCabello** (Negro, Rubio, Castano)	**Deportista** (Si, No)
M	A	G	L	B	S	C	N
F	M	F	M	A	A	N	S
M	M	N	S	M	M	R	S
F	B	F	L	B	S	N	N
M	A	N	S	M	A	N	S
M	B	F	L	B	A	R	N
F	M	G	M	A	S	R	N
F	A	N	S	M	A	C	S
M	M	N	S	B	S	R	N
F	B	G	L	A	M	C	S
M	A	N	M	M	S	N	N
F	M	G	L	B	M	R	S
M	M	N	S	M	S	C	N
M	A	N	S	B	A	N	N
M	B	F	M	A	M	R	S
F	M	G	L	M	S	C	N

$$H(X) = -\sum_i p(x_i)\log_2 p(x_i) = \sum_i p(x_i)\log_2(1/p(x_i))$$

		Total	S	N	Entropia	GANANCIA DE INFORMACION
Deportista		16	7	9	0.98869941	
Genero	M	9	3	6	0.91829583	0.04112069
	F	7	4	3	0.98522814	
Altura	A	5	2	3	0.97095059	0.00424004
	M	7	3	4	0.98522814	
	B	4	2	2	1	
Peso	G	5	2	3	0.97095059	0.00424004
	N	7	3	4	0.98522814	
	F	4	2	2	1	
Talla	S	6	3	3	1	0.01933847
	M	4	2	2	1	
	L	6	2	4	0.91829583	
Competencia	A	4	3	1	0.81127812	0.16712147
	B	6	1	5	0.65002242	
	M	6	3	3	1	
NiveldeActFi	S	7	0	7	0	0.68527735 Manda el NiveldeEstadoFisico
	A	5	3	2	0.97095059	
	M	4	4	0	0	
ColordeCab	N	5	2	3	0.97095059	0.00685529
	R	6	3	3	1	
	C	5	2	3	0.97095059	

9. Usted se escapara al campo luego del examen de IA, deberá llevar varios artículos que no ingresan a su mochila. ¿Cómo optimizaría este problema permitiendo llevar la mayor cantidad de articulos?

El problema que describes se asemeja al clásico **Problema de la Mochila (Knapsack Problem)** en teoría de la computación y optimización combinatoria. En este problema, se busca maximizar el valor total de los artículos que se pueden llevar en una mochila, considerando un límite de peso (o volumen).

1.1.1 Definir el Problema

Primero, debes definir algunos parámetros clave:

- **Capacidad de la mochila (C)**: Es el límite de peso o volumen que la mochila puede llevar.
- **Artículos (A)**: Una lista de artículos que deseas llevar, donde cada artículo tiene un peso (o volumen) y un valor (o utilidad).

```
# Definición de los artículos
articulos = [
    {'nombre': 'CUADERNO 1', 'peso': 2, 'valor': 3},
    {'nombre': 'LIBROS 2', 'peso': 3, 'valor': 4},
    {'nombre': 'LAPTOP 3', 'peso': 10, 'valor': 9},
    {'nombre': 'BOLIGRAFOS 4', 'peso': 5, 'valor': 6},
]

def knapsack(articulos, capacidad):
    n = len(articulos)
    # Crear una tabla para almacenar el valor máximo para cada capacidad
    dp = [[0 for _ in range(capacidad + 1)] for _ in range(n + 1)]

    # Llenar la tabla de DP
```

```
    for i in range(1, n + 1):
        for w in range(1, capacidad + 1):
            if articulos[i - 1]['peso'] <= w:
                dp[i][w] = max(dp[i - 1][w], dp[i - 1][w - articulos[i - 1]['peso']] +
articulos[i - 1]['valor'])
            else:
                dp[i][w] = dp[i - 1][w]

    # Recuperar los artículos seleccionadosQ
    resultado = []
    w = capacidad
    for i in range(n, 0, -1):
        if dp[i][w] != dp[i - 1][w]:  # Significa que el artículo i fue incluido
            resultado.append(articulos[i - 1])
            w -= articulos[i - 1]['peso']

    return dp[n][capacidad], resultado

# Definir la capacidad de la mochila
capacidad_mochila = 8
valor_maximo, articulos_seleccionados = knapsack(articulos, capacidad_mochila)

# Imprimir resultados
print(f'El valor máximo que se puede llevar es: {valor_maximo}')
print('Artículos seleccionados:')
for articulo in articulos_seleccionados:
    print(f"{articulo['nombre']} - Peso: {articulo['peso']}, Valor:
{articulo['valor']}")
```

```
El valor máximo que se puede llevar es: 10
Artículos seleccionados:
BOLIGRAFOS 4 - Peso: 5, Valor: 6
LIBROS 2 - Peso: 3, Valor: 4
```

10. En Excel con el uso de formulas convierta un decimal en binario, octal y hexadecimal.

Convertir con macros VB con formulas de excel

```
Sub ConvertirDecimal()

    Dim ws As Worksheet

    Dim rng As Range
```

```
    Dim cell As Range
    Dim decimalNumber As Long
    Dim binario As String
    Dim octal As String
    Dim hexadecimal As String

    ' Cambia "Hoja1" al nombre de tu hoja de trabajo
    Set ws = ThisWorkbook.Sheets("Hoja1")

    ' Cambia "A4:A21" al rango que contiene tus números decimales
    Set rng = ws.Range("A4:A21")

    ' Itera a través de cada celda en el rango
    For Each cell In rng
        If IsNumeric(cell.Value) Then
            decimalNumber = cell.Value

            ' Convierte a diferentes bases
            binario = Application.WorksheetFunction.Dec2Bin(decimalNumber)
            octal = Application.WorksheetFunction.Dec2Oct(decimalNumber)
            hexadecimal = Application.WorksheetFunction.Dec2Hex(decimalNumber)

            ' Escribe los resultados en las columnas B, C y D
            cell.Offset(0, 1).Value = binario         ' Columna B
            cell.Offset(0, 2).Value = octal           ' Columna C
            cell.Offset(0, 3).Value = hexadecimal      ' Columna D
```

```
        Else

            ' Si no es un número, deja las celdas vacías

            cell.Offset(0, 1).Value = ""

            cell.Offset(0, 2).Value = ""

            cell.Offset(0, 3).Value = ""

        End If

    Next cell

End Sub
```

	A	B	C	D	E	F	G	H	I	J	K	L
1	10.-En Excel con el uso de formulas convierta un decimal en binario, octal y hexadecimal.								UNIV. CONDORI YUJRA MOISES MARTIN			
2	VISUALBASIC								FORMULAS			
3	DECIMAL	BINARIO	OCTAL	HEXADECIMAL					DECIMAL	BINARIO	OCTAL	HEXADECIMAL
4	40	101000	50	28					40	101000	50	28
5	1	1	1	1					1	1	1	1
6	2	10	2	2					2	10	2	2
7	5	101	5	5					5	101	5	5
8	8	1000	10	8					8	1000	10	8
9	7	111	7	7					7	111	7	7
10	9	1001	11	9					9	1001	11	9
11	1	1	1	1					1	1	1	1
12	10	1010	12	A					10	1010	12	A
13	2	10	2	2					2	10	2	2
14	20	10100	24	14					20	10100	24	14
15	30	11110	36	1E					30	11110	36	1E
16	3	11	3	3					3	11	3	3
17	5	101	5	5					5	101	5	5
18	4	100	4	4					4	100	4	4
19	40	101000	50	28					40	101000	50	28
20	50	110010	62	32					50	110010	62	32
21	100	1100100	144	64					100	1100100	144	64
22												
23												