In [6]:
```
import numpy as np
import pandas as pd
from scipy.stats import entropy
```

El siguiente código: utiliza la biblioteca pandas para leer un archivo CSV llamado "weather.nominal.csv" y almacenar su contenido en un objeto DataFrame de pandas llamado "tabla". El argumento "header=0" especifica que la primera fila del archivo CSV contiene los nombres de las columnas para el DataFrame.

In [7]:
```
tabla = pd.read_csv("weather.nominal.csv", header=0)
tabla
```

Out[7]:

	id	outlook	temperature	humidity	windy	play
0	1	sunny	hot	high	False	no
1	2	sunny	hot	high	True	no
2	3	overcast	hot	high	False	yes
3	4	rainy	mild	high	False	yes
4	5	rainy	cool	normal	False	yes
5	6	rainy	cool	normal	True	no
6	7	overcast	cool	normal	True	yes
7	8	sunny	mild	high	False	no
8	9	sunny	cool	normal	False	yes
9	10	rainy	mild	normal	False	yes
10	11	sunny	mild	normal	True	yes
11	12	overcast	mild	high	True	yes
12	13	overcast	hot	normal	False	yes
13	14	rainy	mild	high	True	no

El siguiente código: calcula la entropía de la variable objetivo (la última columna) en el DataFrame "tabla". La función "tabla.iloc[:, -1].value_counts(normalize=True)" selecciona la última columna del DataFrame "tabla" (accediendo a ella mediante el índice "-1"), cuenta la frecuencia de cada valor de esa columna utilizando el método "value_counts()", y normaliza las frecuencias para obtener las proporciones de cada valor utilizando el argumento "normalize=True". El resultado es una serie pandas que contiene las proporciones de cada valor de la última columna de "tabla". La función "entropy()" se utiliza para calcular la entropía de la serie pandas resultante. El primer argumento de "entropy()" es la serie de proporciones, y el segundo argumento especifica la base del logaritmo utilizado para calcular la entropía (en este caso, se utiliza una base de 2 para obtener la entropía en bits). El resultado de este código es el valor de la entropía de la variable objetivo en "tabla" en bits.

In [8]:
```
H = entropy(tabla.iloc[:, -1].value_counts(normalize=True), base=2)
H
```

Out[8]: 0.940285958670631

El siguiente código: ccalcula la ganancia de información de cada atributo en el DataFrame "tabla" con respecto a la variable objetivo (la última columna). El bucle "for atributo in tabla.columns[:-1]:" itera sobre todas las columnas de "tabla" excepto la última (la variable objetivo), y para cada columna realiza los siguientes pasos: Añade la entropía H de la variable objetivo (calculada previamente) a la lista G. Para cada valor v único en la columna actual, filtra las filas de "tabla" que contienen ese valor en la columna actual. Calcula la proporción de filas que contienen el valor v en la columna actual y la utiliza para ponderar la entropía de la variable objetivo condicionada a que la columna actual tenga el valor v. Esto se hace restando la entropía de la variable objetivo en el subconjunto de filas que contienen el valor v en la columna actual de la entropía H previamente calculada. La entropía condicionada se calcula utilizando la misma fórmula que en el código anterior. Resta la entropía ponderada del paso anterior a la entropía H y añade el resultado a la lista G. Al final del bucle, la lista G contendrá la ganancia de información de cada atributo en "tabla" con respecto a la variable objetivo. El resultado final es la lista G.

In [9]:
```
G = []
for atributo in tabla.columns[:-1]:
    G.append(H)
    for v in tabla[atributo].unique():
        filas = tabla.loc[tabla[atributo] == v]
        G[-1] -= (len(filas)/len(tabla))*entropy(filas.iloc[:, -1].value_co
G
```

Out[9]:
```
[0.940285958670631,
 0.246749819774439,
 0.029222565658954758,
 0.15183550136234159,
 0.04812703040826943]
```

El siguiente código: imprime la ganancia máxima de información (en bits) obtenida por los atributos en el DataFrame "tabla" con respecto a la variable objetivo (la última columna). La expresión "np.argmax(G[1:])+1" encuentra el índice del elemento máximo de la lista G a partir del segundo elemento (índice 1) hasta el final. El resultado se suma 1 para ajustar el índice al número de columna correspondiente en "tabla". Esto devuelve el índice del atributo que tiene la mayor ganancia de información con respecto a la variable objetivo. La expresión "tabla.columns[np.argmax(G[1:])+1]" utiliza el índice del atributo con mayor ganancia de información para obtener su nombre de columna correspondiente en "tabla". La expresión "G[np.argmax(np.array(G[1:]))+1]" utiliza el mismo índice del atributo con mayor ganancia de información para obtener su valor de ganancia de información correspondiente de la lista G. En conjunto, este código imprime el nombre del atributo con la mayor ganancia de información con respecto a la variable objetivo, seguido de su valor de ganancia de información en bits.

In [10]:
```
print("Ganancia MAXIMA en Información (Entropía) =    -", tabla.columns[np.
```

```
Ganancia MAXIMA en Información (Entropía) =    - outlook -  Correspondiente
al valor  0.246749819774439
```

El siguiente código: calcula el índice de Gini de cada atributo en el DataFrame "tabla" con respecto a la variable objetivo (la última columna). La variable "G_Gini" se inicializa como el valor del índice de Gini para la variable objetivo. El índice de Gini para una variable categórica es el índice de impureza mínimo que se puede alcanzar cuando se divide la población en subgrupos utilizando los valores de la variable categórica. En este caso, el índice de Gini para la variable objetivo se calcula como 1 menos la suma de los cuadrados de las proporciones de cada valor de la variable objetivo. El bucle "for atributo in tabla.columns[:-1]:" itera sobre todas las columnas de "tabla" excepto la última (la variable objetivo), y para cada columna realiza los siguientes pasos: Añade el valor de G_Gini a la lista Gi. Para cada valor v único en la columna actual, filtra las filas de "tabla" que contienen ese valor en la columna actual. Calcula la proporción de filas que contienen el valor v en la columna actual y la utiliza para ponderar el índice de Gini del subgrupo de filas que contienen el valor v en la columna actual. Esto se hace restando el índice de Gini del subgrupo del valor ponderado de G_Gini. El índice de Gini del subgrupo se calcula como 1 menos la suma de los cuadrados de las proporciones de cada valor de la variable objetivo en el subgrupo. Resta el índice ponderado del paso anterior a G_Gini y añade el resultado a la lista Gi. Al final del bucle, la lista Gi contendrá el índice de Gini de cada atributo en "tabla" con respecto a la variable objetivo. El resultado final es la lista Gi.

In [11]:
```
Gi = []
G_Gini = 1-sum(tabla.iloc[:,-1].value_counts(normalize=True)**2)
l = len(tabla)
for atributo in tabla.columns[:-1]:
    Gi.append(G_Gini)
    for v in tabla[atributo].unique():
        table = tabla.loc[tabla[atributo] == v]
        Gi[-1] -= (len(table)/l)*(1-sum(table.iloc[:,-1].value_counts(norma
Gi
```

Out[11]:
```
[0.4591836734693877,
 0.11632653061224485,
 0.018707482993197258,
 0.09183673469387743,
 0.030612244897959162]
```

El siguiente código: imprime la ganancia máxima de índice de Gini obtenida por los atributos en el DataFrame "tabla" con respecto a la variable objetivo (la última columna). La expresión "np.argmax(Gi[1:])+1" encuentra el índice del elemento máximo de la lista Gi a partir del segundo elemento (índice 1) hasta el final. El resultado se suma 1 para ajustar el índice al número de columna correspondiente en "tabla". Esto devuelve el índice del atributo que tiene el mayor índice de Gini con respecto a la variable objetivo. La expresión "tabla.columns[np.argmax(Gi[1:])+1]" utiliza el índice del atributo con mayor índice de Gini para obtener su nombre de columna correspondiente en "tabla". La expresión "Gi[np.argmax(np.array(Gi[1:]))+1]" utiliza el mismo índice del atributo con mayor índice de Gini para obtener su valor de índice de Gini correspondiente de la lista Gi. En conjunto, este código imprime el nombre del atributo con el mayor índice de Gini con respecto a la variable objetivo, seguido de su valor de índice de Gini.

In [12]:
```
print("Ganancia MAXIMA en Información (GINI) =    -", tabla.columns[np.argm
```

```
Ganancia MAXIMA en Información (GINI) =    - outlook - Correspondiente al v
alor  0.11632653061224485
```

In [35]:
```
# Crear la lista de diccionarios
atributos = tabla.columns[:-1]
datos = []
for i in range(len(atributos)):
    datos.append({
        "Atributo": atributos[i],
        "Entropía": "{:.4f}".format(H),
        "Ganancia de Información": "{:.4f}".format(G[i]),
        "Ganancia de Gini": "{:.4f}".format(Gi[i])
    })

# Crear el DataFrame
df = pd.DataFrame(datos)
df.drop([0], axis=0, inplace=True)
df
```

Out[35]:

	Atributo	Entropía	Ganancia de Información	Ganancia de Gini
1	outlook	0.9403	0.2467	0.1163
2	temperature	0.9403	0.0292	0.0187
3	humidity	0.9403	0.1518	0.0918
4	windy	0.9403	0.0481	0.0306