IMPLEMENTACIÓN ALGORITMO ID3

Jose María Perez Bravo y Pablo Gutiérrez Ruiz

Importar librerias

In [62]:

```
import pandas as pd
import queue
import numpy as np
```

Leer dataset

In [63]:

```
farmacos = pd.read_csv("datosTabla.csv", sep = ";")
farmacos = farmacos.iloc[:,1:7]
farmacos
```

Out[63]:

	Presion Arterial	Azucar en Sangre	Indice de Colesterol	Alergia a Antibioticos	Otras Alergias	Administrar Farmaco F
0	Alta	Alto	Alto	No	No	Si
1	Alta	Alto	Alto	Si	No	Si
2	Baja	Alto	Bajo	No	No	Si
3	Media	Alto	Alto	No	Si	No
4	Media	Bajo	Alto	Si	Si	No
5	Baja	Bajo	Alto	Si	Si	Si
6	Alta	Bajo	Alto	Si	No	Si
7	Alta	Bajo	Bajo	No	Si	Si
8	Alta	Alto	Bajo	Si	Si	No
9	Baja	Bajo	Alto	Si	Si	Si
10	Media	Bajo	Bajo	Si	Si	Si
11	Alta	Bajo	Alto	Si	Si	No
12	Baja	Alto	Alto	Si	Si	Si
13	Baja	Alto	Bajo	No	No	Si

In [64]:

```
datos = [['Rainy','Hot','High','False','No'], ['Rainy', 'Hot','High','True','No'], ['Overca
            ['Sunny', 'Mild','High','False','Yes'], ['Sunny', 'Cool', 'Normal', 'False', 'Y
               ['Sunny', 'Cool', 'Normal', 'True', 'No'], ['Overcast', 'Cool', 'Normal', 'T
                 ['Rainy', 'Mild', 'High', 'False', 'No'], ['Rainy', 'Cool', 'Normal', 'Fal
                     ['Sunny', 'Mild', 'Normal', 'False', 'Yes'], ['Rainy', 'Mild', 'Normal
                         ['Overcast', 'Mild', 'High', 'True', 'Yes'], ['Overcast', 'Hot', '
                             ['Sunny', 'Mild', 'High', 'True', 'No']]

play_golf = pd.DataFrame(datos, columns = ['Outlook', 'Temp', 'Humidity', 'Windy', 'Play_Go
play_golf
```

Out[64]:

	Outlook	Temp	Humidity	Windy	Play_Golf
0	Rainy	Hot	High	False	No
1	Rainy	Hot	High	True	No
2	Overcast	Hot	High	False	Yes
3	Sunny	Mild	High	False	Yes
4	Sunny	Cool	Normal	False	Yes
5	Sunny	Cool	Normal	True	No
6	Overcast	Cool	Normal	True	Yes
7	Rainy	Mild	High	False	No
8	Rainy	Cool	Normal	False	Yes
9	Sunny	Mild	Normal	False	Yes
10	Rainy	Mild	Normal	True	Yes
11	Overcast	Mild	High	True	Yes
12	Overcast	Hot	Normal	False	Yes
13	Sunny	Mild	High	True	No

Estructura del árbol

In [65]:

```
class Arbol:
    def __init__(self, raiz):
        self.hijos = []
        self.atributos = []
        self.raiz = raiz

    def agregarElemento(self, nodo, nodoPadre, atributo = ""):
        subarbol = self.buscarSubarbol(nodoPadre)
        subarbol.hijos.append(Arbol(nodo))
        subarbol.atributos.append(atributo)


    def buscarSubarbol(self, nodo):
        if self.raiz == nodo:
            return self
        for subarbol in self.hijos[::-1]:
            arbolBuscado = subarbol.buscarSubarbol(nodo)
            if (arbolBuscado != None):
                return arbolBuscado
        return None


    def profundidad(self):
        if len(self.hijos) == 0:
            return 1
        else:
            maxprof = 0;
            for subarbol in self.hijos:
                prof = subarbol.profundidad()
                if prof > maxprof:
                    maxprof = prof
            return maxprof + 1


    def grado(self, num):
        num += 1
        for hijo in self.hijos:
            grado(hijo, num)


    def printArbolProfundidad(self, atributo = "", level = 0):
        print("|" + "-" * level * 20 + " " + atributo + " --> " + self.raiz)
        for hijo in self.hijos:
            hijo.printArbolProfundidad(self.atributos[self.hijos.index(hijo)], level+1)


    def printArbolAnchura(self, cola = queue.Queue()):
        print(self.raiz)
        if(len(self.hijos) > 0):
            for hijo in self.hijos:
                cola.put(hijo)
        if(cola.qsize() != 0):
            cola.get().printArbolAnchura(cola)
```

In [66]:

```
abuela = "Jacqueline Gurney"
marge = "Marge Bouvier"
patty = "Patty Bouvier"
selma = "Selma Bouvier"
bart = "Bart Simpson"
lisa = "Lisa Simpson"
maggie = "Maggie Simpson"
ling = "Ling Bouvier"

arbol = Arbol(abuela)
arbol.agregarElemento(patty, abuela)
arbol.agregarElemento(selma, abuela)
arbol.agregarElemento(ling, selma)
arbol.agregarElemento(marge, abuela)
arbol.agregarElemento(bart, marge)
arbol.agregarElemento(lisa, marge)
arbol.agregarElemento(maggie, marge)

#print(arbol.profundidad())
arbol.printArbolProfundidad()
#arbol.printArbolAnchura()
```

```
|  --> Jacqueline Gurney
|--------------------  --> Patty Bouvier
|--------------------  --> Selma Bouvier
|---------------------------------------  --> Ling Bouvier
|--------------------  --> Marge Bouvier
|---------------------------------------  --> Bart Simpson
|---------------------------------------  --> Lisa Simpson
|---------------------------------------  --> Maggie Simpson
```

ID3

In [67]:

```
def id3(df, columna, arbol = None):
    ncol = len(df.columns)        ## Numero de atributos
    ganancias = []                ## Aqui guardo las ganancias

    col_class = df[columna]     ## Esta es la columna a predecir
    etiqueta_class = col_class.value_counts();   ## Numero de elementos de cada tipo en la
    entropy_class = np.array(etiqueta_class) / len(col_class.index)
    entropy_class = entropy_class * np.log2(entropy_class)
    entropy_class = -np.sum(entropy_class)  ## entropia de la columna a predecir


    for i in range(ncol):   ## Por cada Atributo

        col = df.iloc[:,i]  ## Cojo la columna
        if col.name != columna: ## Si no es la que queremos predecir
            semicalculo = []    ## No hagas mucho caso a esto por ahora
            numero_etiquetas = df.iloc[:,i].value_counts()  ## Esto es el numero de element
            etiquetas = numero_etiquetas.index ## Esto son los distintos elementos, los nom
            class_values = df[columna].dropna().unique()

            for j in etiquetas:  ## Por cada elemento distinto en la columna
                number = []  ## Variable auxiliar
                for z in class_values:  ## Por cada elemento distinto en la columna a prede
                    number.append(sum((df[col.name] == j) & (df[columna] == z)))

                number = np.array(number) / numero_etiquetas[j]
                number_log2 = [0 if x==0 else np.log2(x) for x in number] ##Para evitar que
                number = number * number_log2
                entropia = -np.sum(number)  ## Entropia
                semicalculo.append(numero_etiquetas[j] / len(col_class.index) * entropia) #

            ganancias.append(entropy_class - np.sum(semicalculo))  ## ganancia

    return ganancias

id3(play_golf, 'Play_Golf')
```

Out[67]:

```
[0.24674981977443933,
 0.02922256565895487,
 0.15183550136234159,
 0.04812703040826949]
```

Entrenamiento

In [72]:

```
def aprendizaje(df, columna, arbol = None, padre = "", atributo = ""):

    if(len(df[columna].value_counts()) > 1 and len(df.columns) > 1): ## Si hay más de una c
        ganancias = id3(df, columna, arbol)  ## Obtener las ganancias del dataframe pasado
        columna_ganadora = df.columns[np.argmax(ganancias)] ## Seleccionar la columna del a

        arbol.agregarElemento(columna_ganadora, padre, atributo) ## Añadir esa columna al a

        new_df = df.drop(columna_ganadora, axis = 1) ## Obtener el nuevo df eliminando la c

        numero_etiquetas = df.loc[:,columna_ganadora].value_counts() ## Contar las diferent

        etiquetas = numero_etiquetas.index ## Obtener una lista con las diferentes etiqueta
        #print(etiquetas)
        for i in etiquetas: ## Iteramos por las etiquetas de la columna ganadora
            indices_delete = (df[columna_ganadora] == i).tolist() ## Obtener los indices de
            new_new_df = new_df.iloc[indices_delete] ## Obtener el nuevo df se´gun los índi

            aprendizaje(new_new_df, columna, arbol, padre = columna_ganadora, atributo = i)

    else: ## Si solo queda una clase para predecir o no hay más columnas
        #arbol.agregarElemento(df.iloc[0][columna], padre, atributo) ## Agregar la predicci
        arbol.agregarElemento(np.argmax(df[columna].value_counts()), padre, atributo)


decision_tree = Arbol("")
aprendizaje(play_golf, 'Play_Golf', arbol = decision_tree)
decision_tree = decision_tree.hijos[0]
decision_tree.printArbolProfundidad()

print("\n")

decision_tree2 = Arbol("")
aprendizaje(farmacos, 'Administrar Farmaco F', arbol = decision_tree2)
decision_tree2 = decision_tree2.hijos[0]
decision_tree2.printArbolProfundidad()
```

```
|  --> Outlook
|-------------------- Rainy --> Humidity
|---------------------------------------- High --> No
|---------------------------------------- Normal --> Yes
|-------------------- Sunny --> Windy
|---------------------------------------- False --> Yes
|---------------------------------------- True --> No
|-------------------- Overcast --> Yes


|  --> Presion Arterial
|-------------------- Alta --> Otras Alergias
|---------------------------------------- Si --> Alergia a Antibioticos
|------------------------------------------------------------ Si --> No
|------------------------------------------------------------ No --> Si
|---------------------------------------- No --> Si
|-------------------- Baja --> Si
|-------------------- Media --> Indice de Colesterol
|---------------------------------------- Alto --> No
|---------------------------------------- Bajo --> Si
```

Predicción

In [79]:

```
def predict_one(arbol, datos):
    while arbol.hijos != [] :

        atr = datos[arbol.raiz]
        idx = arbol.atributos.index(atr)
        arbol = arbol.hijos[idx]

    return arbol.raiz

predict_one(decision_tree, play_golf.iloc[0])
```

Out[79]:

```
'No'
```

In [80]:

```
def predict(arbol, df, name):
    df[name] = df.apply(lambda row: predict_one(arbol, row), axis=1)
```

Accuracy

In [81]:

```
def accuracy(original, prediccion):
    return np.sum(original == prediccion) / len(original)
```

Prueba de predicción

In [82]:

```
print("***** DATOS_ORIGINALES ******")
print(play_golf.iloc[range(5),range(5)])
new_df = play_golf.iloc[range(5),range(4)]
print("******** PREDICCION *********")
predict(decision_tree, new_df, "Play_Golf")
print(new_df)
```

```
***** DATOS_ORIGINALES ******
    Outlook  Temp Humidity  Windy Play_Golf
0     Rainy   Hot     High  False        No
1     Rainy   Hot     High   True        No
2  Overcast   Hot     High  False       Yes
3     Sunny  Mild     High  False       Yes
4     Sunny  Cool   Normal  False       Yes
******** PREDICCION *********
    Outlook  Temp Humidity  Windy Play_Golf
0     Rainy   Hot     High  False        No
1     Rainy   Hot     High   True        No
2  Overcast   Hot     High  False       Yes
3     Sunny  Mild     High  False       Yes
4     Sunny  Cool   Normal  False       Yes
```

In [83]:

```
print("***** DATOS_ORIGINALES ******")
print(farmacos.iloc[1:10,0:6])
new_df2 = farmacos.iloc[1:10,0:5]
print("******** PREDICCION *********")
predict(decision_tree2, new_df2, "Administrar Farmaco F")
print(new_df2)
```

```
***** DATOS_ORIGINALES ******
  Presion Arterial Azucar en Sangre Indice de Colesterol  \
1             Alta             Alto                  Alto
2             Baja             Alto                 Bajo
3            Media             Alto                  Alto
4            Media             Bajo                  Alto
5             Baja             Bajo                  Alto
6             Alta             Bajo                  Alto
7             Alta             Bajo                 Bajo
8             Alta             Alto                 Bajo
9             Baja             Bajo                  Alto

  Alergia a Antibioticos Otras Alergias Administrar Farmaco F
1                     Si             No                    Si
2                     No             No                    Si
3                     No             Si                    No
4                     Si             Si                    No
5                     Si             Si                    Si
6                     Si             No                    Si
7                     No             Si                    Si
8                     Si             Si                    No
9                     Si             Si                    Si
******** PREDICCION *********
  Presion Arterial Azucar en Sangre Indice de Colesterol  \
1             Alta             Alto                  Alto
2             Baja             Alto                 Bajo
3            Media             Alto                  Alto
4            Media             Bajo                  Alto
5             Baja             Bajo                  Alto
6             Alta             Bajo                  Alto
7             Alta             Bajo                 Bajo
8             Alta             Alto                 Bajo
9             Baja             Bajo                  Alto

  Alergia a Antibioticos Otras Alergias Administrar Farmaco F
1                     Si             No                    Si
2                     No             No                    Si
3                     No             Si                    No
4                     Si             Si                    No
5                     Si             Si                    Si
6                     Si             No                    Si
7                     No             Si                    Si
8                     Si             Si                    No
9                     Si             Si                    Si
```

Demo 1

Entrenar el modelo

In [84]:

```
decision_tree = Arbol("")
aprendizaje(play_golf, 'Play_Golf', arbol = decision_tree)
decision_tree = decision_tree.hijos[0]
decision_tree.printArbolProfundidad()
```

```
|  --> Outlook
|-------------------- Rainy --> Humidity
|---------------------------------------- High --> No
|---------------------------------------- Normal --> Yes
|-------------------- Sunny --> Windy
|---------------------------------------- False --> Yes
|---------------------------------------- True --> No
|-------------------- Overcast --> Yes
```

Predecir con el modelo

In [88]:

```
print("***** DATOS_ORIGINALES ******")
print(play_golf.iloc[range(5),range(5)])
new_df = play_golf.iloc[range(5),range(4)]
print("******** PREDICCION *********")
predict(decision_tree, new_df, "Play_Golf")
print(new_df)
```

```
***** DATOS_ORIGINALES ******
    Outlook  Temp Humidity  Windy Play_Golf
0     Rainy   Hot     High  False        No
1     Rainy   Hot     High   True        No
2  Overcast   Hot     High  False       Yes
3     Sunny  Mild     High  False       Yes
4     Sunny  Cool   Normal  False       Yes
******** PREDICCION *********
    Outlook  Temp Humidity  Windy Play_Golf
0     Rainy   Hot     High  False        No
1     Rainy   Hot     High   True        No
2  Overcast   Hot     High  False       Yes
3     Sunny  Mild     High  False       Yes
4     Sunny  Cool   Normal  False       Yes
```

Accuracy

In [89]:

```
print("Accuracy: " + str(accuracy(play_golf.iloc[0:5,4], new_df['Play_Golf'])))
```

```
Accuracy: 1.0
```

Demo 2

Entrenar el modelo

In [86]:

```
decision_tree_farmaco = Arbol("")
aprendizaje(farmacos, 'Administrar Farmaco F', arbol = decision_tree_farmaco)
decision_tree_farmaco = decision_tree_farmaco.hijos[0]
decision_tree_farmaco.printArbolProfundidad()
```

```
|  --> Presion Arterial
|-------------------- Alta --> Otras Alergias
|---------------------------------------- Si --> Alergia a Antibioticos
|------------------------------------------------------------ Si --> No
|------------------------------------------------------------ No --> Si
|---------------------------------------- No --> Si
|-------------------- Baja --> Si
|-------------------- Media --> Indice de Colesterol
|---------------------------------------- Alto --> No
|---------------------------------------- Bajo --> Si
```

Predecir con el modelo

In [90]:

```
print("***** DATOS_ORIGINALES ******")
print(farmacos.iloc[1:10,0:6])
#new_df_farmaco = farmacos.iloc[1:10,0:5]
new_df_farmaco = farmacos
print("******** PREDICCION *********")
predict(decision_tree_farmaco, new_df_farmaco, "Administrar Farmaco F")
print(new_df_farmaco)
```

```
***** DATOS_ORIGINALES ******
  Presion Arterial Azucar en Sangre Indice de Colesterol  \
1             Alta             Alto                  Alto
2             Baja             Alto                 Bajo
3            Media             Alto                  Alto
4            Media             Bajo                  Alto
5             Baja             Bajo                  Alto
6             Alta             Bajo                  Alto
7             Alta             Bajo                 Bajo
8             Alta             Alto                 Bajo
9             Baja             Bajo                  Alto

  Alergia a Antibioticos Otras Alergias Administrar Farmaco F
1                     Si             No                    Si
2                     No             No                    Si
3                     No             Si                    No
4                     Si             Si                    No
5                     Si             Si                    Si
6                     Si             No                    Si
7                     No             Si                    Si
8                     Si             Si                    No
9                     Si             Si                    Si
******** PREDICCION *********
   Presion Arterial Azucar en Sangre Indice de Colesterol  \
0             Alta             Alto                  Alto
1             Alta             Alto                  Alto
2             Baja             Alto                 Bajo
3            Media             Alto                  Alto
4            Media             Bajo                  Alto
5             Baja             Bajo                  Alto
6             Alta             Bajo                  Alto
7             Alta             Bajo                 Bajo
8             Alta             Alto                 Bajo
9             Baja             Bajo                  Alto
10           Media             Bajo                 Bajo
11            Alta             Bajo                  Alto
12            Baja             Alto                  Alto
13            Baja             Alto                 Bajo

   Alergia a Antibioticos Otras Alergias Administrar Farmaco F
0                     No             No                    Si
1                     Si             No                    Si
2                     No             No                    Si
3                     No             Si                    No
4                     Si             Si                    No
5                     Si             Si                    Si
6                     Si             No                    Si
7                     No             Si                    Si
8                     Si             Si                    No
9                     Si             Si                    Si
10                    Si             Si                    Si
```

11	Si	Si	No
12	Si	Si	Si
13	No	No	Si

Accuracy

In [91]:

```
print("Accuracy: " + str(accuracy(farmacos.iloc[:,5], new_df_farmaco.iloc[:,5])))
```

```
Accuracy: 1.0
```