Decision Tree

En este archivo se va a analizar mediante el algoritmo del árbol de decisión los datos del dataset del Titanic

In [35]:
```
import warnings
warnings.filterwarnings('ignore')
import pandas as pd
```

In [36]:
```
df = pd.read_csv('titanic.txt', sep= ',')
```

In [64]:
```
df.head()
```

Out[64]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN

La columna 'Age' tiene algunos valores en blanco. Podemos rellenarlos con la mediana para poder seguir con el análisis:

In [82]:
```
df['Age'].median()
```

Out[82]:
```
28.0
```

In [97]:
```
df ['Age'] = df['Age'].fillna(df['Age'].median())
```

Vamos a centrarnos en las siguientes columnas:

- 'Survived': 0 No, 1 Yes.
- 'Pclass': Clase en la que viajaba cada pasajero.

- 'Sex': Sexo del pasajero.
- 'Age': Edad del pasajero.
- 'Fare': Tarifa que se le aplicó al pasajero.

In [98]:
```
df2 = df[['Survived', 'Pclass', 'Sex', 'Age','Fare']]
```

In [99]:
```
df2.head()
```

Out[99]:

	Survived	Pclass	Sex	Age	Fare
0	0	3	male	22.0	7.2500
1	1	1	female	38.0	71.2833
2	1	3	female	26.0	7.9250
3	1	1	female	35.0	53.1000
4	0	3	male	35.0	8.0500

Vamos a asignarle valores a las variables 'Sex' de manera que:

- Sex: 0 = male; 1 = female

In [100…
```
#Creamos la lista Sex_n para poder añadirla al dataframe como columna:
Sex_number = []
for i in df2['Sex']:
    if i == 'male':
        Sex_number.append(0)
    else:
        Sex_number.append(1)
```

In [101…
```
#la añadimos al dataframe df2 como columna 'Sex_n':
df2['Sex_n'] = Sex_number
```

In [102…
```
#Comprobamos que se ha añadido en nuestro dataframe:
df2.head()
```

Out[102…

	Survived	Pclass	Sex	Age	Fare	Sex_n
0	0	3	male	22.0	7.2500	0
1	1	1	female	38.0	71.2833	1
2	1	3	female	26.0	7.9250	1
3	1	1	female	35.0	53.1000	1
4	0	3	male	35.0	8.0500	0

Con los datos codificados así, podemos hacer nuestro árbol de decisión:

Lo primero que haremos será dividir nuestra muestra en dos. Una de 'Trainig' que servirá para ajustar el modelo y otra de 'Test' con la que obtendremos una estimación de la precisión de nuestro modelo.

In [103…

```
from sklearn.model_selection import train_test_split
```

In [173…
```
#Llamamos X e Y a las variables que vamos a usar como independientes y dependientes,
X = df2[['Pclass', 'Sex_n', 'Age', 'Fare']].values
y = df2['Survived'].values
```

In [174…
```
#Definimos en el programa la parte de la muestra destinada al Training, en nuestro c
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2)
```

In [176…
```
#Podemos ver qué escoge como X_test
#Cada vez que ejecutemos la celda anterior, cambiará nuestro X_test porque no le hem
#el argumento random state nos guardaría en memoria la muestra aleatoria que sacara
#X_test
```

Con los datos ya preparados, y el subconjunto de entrenamiento (X_train e y_train) y el subconjunto de validación/prueba (X_test, y_test) ya listos, ejecutamos el modelo:

In [177…
```
from sklearn import tree
```

In [178…
```
#Especificamos que queremos un modelo de Decision Tree:
modelo = tree.DecisionTreeClassifier()
```

In [179…
```
#Ajustamos el modelo con nuestros datos de Training
modelo.fit(X_train, y_train)
```

Out[179…
```
DecisionTreeClassifier()
```

In [180…
```
#Validamos el modelo con nuestros datos de Test
modelo.score(X_test, y_test)
#El método score hace las predicciones de los datos que pasemos como primer argument
# y compara esas predicciones con lo que le pasemos como segundo argumento (y_test e
```

Out[180…
```
0.7932960893854749
```

Al obtener la precisión del modelo, con varias muestras aleatorias (X_test, y_test, etc), obtenemos valores de precisión en un el intervalo aproximado de (0.74 , 0.83). En concreto, con la muestra aleatoria que se usó en la celda anterior, se ha obtenido un 0.79

In [200…
```
#Algunas predicciones:
#Pasajera de 2a clase, mujer de 23 años cuya tarifa fue de 32:
Pasajera = [[2, 1, 23.0, 32.0000]]
print(modelo.predict(Pasajera))
```

```
[1]
```

El modelo predice que sí sobrevivirá.

In [204…
```
#Pasajero de 3a clase, hombre de 53 años cuya tarifa fue de 12:
Pasajero = [[3, 0, 53.0, 12.0000]]
print(modelo.predict(Pasajero))
```

```
[0]
```

El modelo predice que el pasajero no sobrevivirá