In [1]:

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
```

In [2]:

```
data = pd.read_csv('C:/Users/Deep/Desktop/titanic.csv')
```

In [3]:

```
data.head()
```

Out[3]:

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

In [4]:

```
data.shape
```

Out[4]:

```
(891, 12)
```

In [5]:

```
input = data[['Age','Pclass','Sex','Fare','Survived']]
input.head()
```

Out[5]:

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

label encoding the sex into numbers

In [6]:

```
LE = LabelEncoder()
```

In [7]:

```
input['Sex_n'] = LE.fit_transform(input['Sex'])
```

```
C:\Users\Deep\anaconda3\lib\site-packages\ipykernel_launcher.py:1: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexi
ng.html#returning-a-view-versus-a-copy
  """Entry point for launching an IPython kernel.
```

In [8]:

```
input
```

Out[8]:

	Age	Pclass	Sex	Fare	Survived	Sex_n
0	22.0	3	male	7.2500	0	1
1	38.0	1	female	71.2833	1	0
2	26.0	3	female	7.9250	1	0
3	35.0	1	female	53.1000	1	0
4	35.0	3	male	8.0500	0	1
...	...	...	...	...	...	...
886	27.0	2	male	13.0000	0	1
887	19.0	1	female	30.0000	1	0
888	NaN	3	female	23.4500	0	0
889	26.0	1	male	30.0000	1	1
890	32.0	3	male	7.7500	0	1

891 rows × 6 columns

dropping the sex column

In [9]:

```
input.drop(axis=1,columns='Sex',inplace=True)
```

```
C:\Users\Deep\anaconda3\lib\site-packages\pandas\core\frame.py:3997: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexi
ng.html#returning-a-view-versus-a-copy
  errors=errors,
```

In [10]:

```
input.describe()
```

Out[10]:

	Age	Pclass	Fare	Survived	Sex_n
count	714.000000	891.000000	891.000000	891.000000	891.000000
mean	29.699118	2.308642	32.204208	0.383838	0.647587
std	14.526497	0.836071	49.693429	0.486592	0.477990
min	0.420000	1.000000	0.000000	0.000000	0.000000
25%	20.125000	2.000000	7.910400	0.000000	0.000000
50%	28.000000	3.000000	14.454200	0.000000	1.000000
75%	38.000000	3.000000	31.000000	1.000000	1.000000
max	80.000000	3.000000	512.329200	1.000000	1.000000

Handling Missing Values

In [11]:

```
#input = input.dropna(axis = 0)
```

In [12]:

```
input.shape
```

Out[12]:

```
(891, 5)
```

In [13]:

```
input = input.fillna(input.mean())
```

Dividing features and target

In [14]:

```
features = input[['Age','Pclass','Sex_n','Fare']]
features.shape
```

Out[14]:

```
(891, 4)
```

In [15]:

```
target = input[['Survived']]
target.shape
```

Out[15]:

```
(891, 1)
```

Splittng the data into train and test

In [16]:

```
X_train, X_test, y_train, y_test = train_test_split(features, target, test_size= 0.30, random_state= 12)
```

In [17]:

```
X_test.shape
```

Out[17]:

```
(268, 4)
```

Training the decision tree

In [18]:

```
model = DecisionTreeClassifier(criterion='gini', max_depth=7)
```

In [19]:

```
model.fit(X_train, y_train)
```

Out[19]:

```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                       max_depth=7, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=1, min_samples_split=2,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=None, splitter='best')
```

Checking the Accuracy

In [20]:

```
model.score(X_train, y_train)
```

Out[20]:

```
0.9165329052969502
```

In [21]:

```
model.score(X_test, y_test)
```

Out[21]:

```
0.7947761194029851
```

In [22]:

```
model.feature_importances_
```

Out[22]:

```
array([0.15412947, 0.15162644, 0.45544863, 0.23879545])
```

In [23]:

```
model.predict([[65,1,1,26]])
```

Out[23]:

```
array([0], dtype=int64)
```

In [24]:

```
model.predict([[30,1,0,52]])
```

Out[24]:

```
array([1], dtype=int64)
```

In [25]:

```
X_test
```

Out[25]:

	Age	Pclass	Sex_n	Fare
456	65.000000	1	1	26.5500
351	29.699118	1	1	35.0000
173	21.000000	3	1	7.9250
671	31.000000	1	1	52.0000
836	21.000000	3	1	8.6625
...	...	...	...	...
860	41.000000	3	1	14.1083
69	26.000000	3	1	8.6625
647	56.000000	1	1	35.5000
872	33.000000	1	1	5.0000
669	29.699118	1	0	52.0000

268 rows × 4 columns

In [26]:

```
y_test
```

Out[26]:

	Survived
456	0
351	0
173	0
671	0
836	0
...	...
860	0
69	0
647	1
872	0
669	1

268 rows × 1 columns

In [ ]:

```

```