decision tree is a type of machine learning algorithm that is mostly used in # #classification problems.
decision tree splits population data set into two or more homogeneous sets based on the most significant splitter in input varibles

predict the survival of a passenger based on the class, the sex, age,and fare in this titanic movie

In this file using following columns build a model to predict if person would survive or not,
Pclass
Sex
Age
Fare
Calculate score of your model

In [6]:
```
import pandas as pd
```

In [7]:
```
titanic_movie=pd.read_csv("titanic.csv")
```

In [8]:
```
titanic_movie.head()
```

Out[8]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabi
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	Na
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C8
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	Na
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C12
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	Na

cleaning the data

In [30]:
```
# filling the empty values with the median value of the age

median_Age=(titanic_movie.Age.median())
titanic_movie.Age = titanic_movie.Age.fillna(median_Age)

bool_series = pd.notnull(titanic_movie["Cabin"])

# filtering data
# displaying data only with experience = Not NaN


titanic_movie[bool_series]
titanic_movie.Cabin.fillna((0), inplace=True)
```

In [31]:
```
titanic_movie.head(10)
```

Out[31]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabi
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C8
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C12
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	
5	6	0	3	Moran, Mr. James	male	28.0	0	0	330877	8.4583	
6	7	0	1	McCarthy, Mr. Timothy J	male	54.0	0	0	17463	51.8625	E4
7	8	0	3	Palsson, Master. Gosta Leonard	male	2.0	3	1	349909	21.0750	
8	9	1	3	Johnson, Mrs. Oscar W (Elisabeth Vilhelmina Berg)	female	27.0	0	2	347742	11.1333	
9	10	1	2	Nasser, Mrs. Nicholas (Adele Achem)	female	14.0	1	0	237736	30.0708	

In [32]:
```
#next is to divide it d data between the target variable and the independent vari
independ = titanic_movie.drop('Survived', axis='columns')

#target variable also dependent variable "survived"
target=titanic_movie['Survived']

#dropping the target variable
```

In [33]:
```
#i will call the independent variable data frame as independ
independ
```

Out[33]:

	PassengerId	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Emba
0	1	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	0	
1	2	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	
2	3	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	0	
3	4	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	
4	5	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	0	
...	...	...	...	...	...	...	...	...	...	...	
886	887	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	0	
887	888	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	
888	889	3	Johnston, Miss. Catherine Helen "Carrie"	female	28.0	1	2	W./C. 6607	23.4500	0	
889	890	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	
890	891	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	0	

891 rows × 11 columns

In [34]:
```
#survived column
target
```

Out[34]:
```
0      0
1      1
2      1
3      1
4      0
      ..
886    0
887    1
888    0
889    1
890    0
Name: Survived, Length: 891, dtype: int64
```

In [35]:
```
#using label encoder to convert the sex columns to numbers
from sklearn.preprocessing import LabelEncoder
```

In [36]:
```
le_Sex=LabelEncoder()
```

In [37]:
```
#creating one more column in the independ data frame:Sex


independ['Sex_n']=le_Sex.fit_transform(independ['Sex'])
#this will create extra column to represent your data into numbers ie label encoc
independ.head()
```

Out[37]:

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

In [40]:
```
# droping the data column frame that are not needed

independ_n =independ.drop(['PassengerId','Name','Sex', 'SibSp','Parch', 'Ticket',
```

In [41]:
```
independ_n
# from this table, 1 =male, 0 =female
```

Out[41]:

	Pclass	Age	Fare	Sex_n
0	3	22.0	7.2500	1
1	1	38.0	71.2833	0
2	3	26.0	7.9250	0
3	1	35.0	53.1000	0
4	3	35.0	8.0500	1
...	...	...	...	...
886	2	27.0	13.0000	1
887	1	19.0	30.0000	0
888	3	28.0	23.4500	0
889	1	26.0	30.0000	1
890	3	32.0	7.7500	1

891 rows × 4 columns

great

In [42]:
```
from sklearn import tree
```

In [43]:
```
model=tree.DecisionTreeClassifier()
```

In [44]:
```
#training the model

model.fit(independ_n, target)
```

Out[44]:
```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                       max_depth=None, 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')
```

In [45]:
```
#to predict the score by supplying the number of dependent and independent variab
model.score(independ_n, target)
```

Out[45]:
```
0.9775533108866442
```

```
#good, for the prediction is accurate.
#now we predict a male with the fare, age, and sex
```

In [47]:
```
independ_n
```

Out[47]:

	Pclass	Age	Fare	Sex_n
0	3	22.0	7.2500	1
1	1	38.0	71.2833	0
2	3	26.0	7.9250	0
3	1	35.0	53.1000	0
4	3	35.0	8.0500	1
...	...	...	...	...
886	2	27.0	13.0000	1
887	1	19.0	30.0000	0
888	3	28.0	23.4500	0
889	1	26.0	30.0000	1
890	3	32.0	7.7500	1

891 rows × 4 columns

In [48]:
```
model.predict([[1,38.0,71.2833,0]])
```

Out[48]:
```
array([1], dtype=int64)
```

```
the output indicates that the person with the above data survived.
```