Titanic Survival Prediction using Python

In [8]:
```
import pandas as pd
import numpy as np
import sys
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, classification_report
```

In [9]:
```
import os
working_directory = os.getcwd()
print (working_directory)
```

```
/Users/ishu
```

Importing the Dataset and Data Preprocessing

In [10]:
```
path = working_directory + '/Desktop/tested.csv'
data = pd.read_csv (path)
data.head()
```

Out[10]:

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	892	0	3	Kelly, Mr. James	male	34.5	0	0	330911	7.8292	NaN	Q
1	893	1	3	Wilkes, Mrs. James (Ellen Needs)	female	47.0	1	0	363272	7.0000	NaN	S
2	894	0	2	Myles, Mr. Thomas Francis	male	62.0	0	0	240276	9.6875	NaN	Q
3	895	0	3	Wirz, Mr. Albert	male	27.0	0	0	315154	8.6625	NaN	S
4	896	1	3	Hirvonen, Mrs. Alexander (Helga E Lindqvist)	female	22.0	1	1	3101298	12.2875	NaN	S

In [11]:
```
data = data.drop(['PassengerId', 'Name', 'Ticket', 'Cabin', 'Embarked'], axis=1)
```

In [12]:
```
data.head()
```

Out[12]:

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare
0	0	3	male	34.5	0	0	7.8292
1	1	3	female	47.0	1	0	7.0000
2	0	2	male	62.0	0	0	9.6875
3	0	3	male	27.0	0	0	8.6625
4	1	3	female	22.0	1	1	12.2875

In [13]:
```
data['Age'].fillna(data['Age'].median(), inplace=True)
```

In [14]:
```
label_encoder = LabelEncoder()
data['Sex'] = label_encoder.fit_transform(data['Sex'])
```

In [15]:
```
data.replace([np.inf, -np.inf], np.nan, inplace=True)
```

In [19]:
```
print(data)
```

```
     Survived  Pclass  Sex   Age  SibSp  Parch      Fare
0           0       3    1  34.5      0      0    7.8292
1           1       3    0  47.0      1      0    7.0000
2           0       2    1  62.0      0      0    9.6875
3           0       3    1  27.0      0      0    8.6625
4           1       3    0  22.0      1      1   12.2875
..        ...     ...  ...   ...    ...    ...       ...
413         0       3    1  27.0      0      0    8.0500
414         1       1    0  39.0      0      0  108.9000
415         0       3    1  38.5      0      0    7.2500
416         0       3    1  27.0      0      0    8.0500
417         0       3    1  27.0      1      1   22.3583

[418 rows x 7 columns]
```

In [20]:
```
data.dropna(inplace=True)
```

In [22]:
```
#Split the data into features and target

X = data.drop('Survived', axis=1)
y = data['Survived']
```

In [23]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

In [24]:
```
clf = DecisionTreeClassifier(random_state=42)
clf.fit(X_train, y_train)
```

Out[24]:

```
DecisionTreeClassifier(random_state=42)
```

In [25]:
```
y_pred = clf.predict(X_test)
```

Evaluating the model

In [26]:
```
accuracy = accuracy_score(y_test, y_pred)
report = classification_report(y_test, y_pred)
```

In [29]:
```
print("Accuracy: {:.2f}%".format(accuracy * 100))
print("\n----------------by ishu-------------------")
print("Classification Report:\n", report)
```

```
Accuracy: 100.00%

----------------by ishu-------------------
Classification Report:
               precision    recall  f1-score   support

           0       1.00      1.00      1.00        50
           1       1.00      1.00      1.00        34

    accuracy                           1.00        84
   macro avg       1.00      1.00      1.00        84
weighted avg       1.00      1.00      1.00        84
```

In [31]:
```
print("ThankYou")
```

```
ThankYou
```

In [6]:

In [6]:

In [6]: