In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt

```

In [25]:
```
df = pd.read_csv(r"C:\Users\Vijay\Downloads\train (2).csv")
df.head()
```

	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 [26]:
```
df.shape
```

```
(891, 12)
```

In [53]:

```
df.dtypes

df['Age'] = df['Age'].astype(int)
df['Fare'] = df['Fare'].astype(int)
```

In [28]:

```
df.describe()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

In [29]:

```
df.columns
```

```
Index(['PassengerId', 'Survived', 'Pclass', 'Name', 'Sex', 'Age', 'SibSp',
       'Parch', 'Ticket', 'Fare', 'Cabin', 'Embarked'],
      dtype='object')
```

In [30]:
```
df.isnull().sum()
```

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age            177
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

In [33]:
```
df['Age'].fillna(df['Age'].mean(),inplace = True)
```

In [34]:
```
df.isnull().sum()
```

```
PassengerId      0
Survived         0
Pclass           0
Name             0
Sex              0
Age              0
SibSp            0
Parch            0
Ticket           0
Fare             0
Cabin          687
Embarked         2
dtype: int64
```

In [35]:
```python
from sklearn.preprocessing import LabelEncoder, OneHotEncoder, StandardScaler, MinMaxScaler

```

In [36]:
```python
df.head()
```

	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 [38]:
```python
label_encoder = LabelEncoder()
df['Sex']  = label_encoder.fit_transform(df['Sex'])
df['Embarked']  = label_encoder.fit_transform(df['Embarked'])

```

In [39]:
```
df.head()
```

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

In [56]:
```
from sklearn.model_selection import train_test_split,GridSearchCV
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

```

In [153]:

```
df_train = df.copy()

# Specify the columns to drop
columns_to_drop = ['PassengerId', 'Survived', 'Name', 'Ticket', 'Cabin','SibSp','Parch']

# Use square brackets for drop and pass the list of columns
X = df_train.drop(columns=columns_to_drop)

y = df_train['Survived']

# The rest of your code for train-test split goes here


X_train,X_test,y_train,y_test  = train_test_split(X,y,test_size=0.15,
    random_state=23)


rf_classifier = RandomForestClassifier(n_estimators=100, random_state=42,max_depth= 10,
                                       min_samples_split = 5,min_samples_leaf = 3,max_features = 'sqrt'


# Train the classifier on the training data
rf_classifier.fit(X_train, y_train)

# Make predictions on the testing data
y_pred = rf_classifier.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)*100
```

```
print(f'Accuracy: {accuracy:.2f}')

# Display classification report
print('Classification Report:\n', classification_report(y_test, y_pred))

```

```
Accuracy: 82.84
Classification Report:
               precision    recall  f1-score   support

           0       0.86      0.90      0.88        92
           1       0.76      0.67      0.71        42

    accuracy                           0.83       134
   macro avg       0.81      0.78      0.79       134
weighted avg       0.82      0.83      0.83       134
```

In [154]:

```
df_test.isnull().sum()
df_test['Age'].fillna(df['Age'].mean(),inplace = True)
df_test['Fare'].fillna(df['Fare'].mean(),inplace = True)

```

In [155]:

```
df_test.isnull().sum()
```

```
Pclass      0
Sex         0
Age         0
Fare        0
Embarked    0
dtype: int64
```

In [156]:
```
df_test = pd.read_csv(r"C:\Users\Vijay\Downloads\test (2).csv")
df_test.isnull().sum()
df_test['Age'].fillna(df['Age'].mean(),inplace = True)
df_test['Fare'].fillna(df['Fare'].mean(),inplace = True)

df_test_PId = df_test["PassengerId"]
df_test = df_test.drop(["PassengerId",'Name', 'Ticket', 'Cabin','SibSp','Parch'], axis=1)
label_encoder = LabelEncoder()
df_test['Sex']  = label_encoder.fit_transform(df_test['Sex'])
df_test['Embarked']  = label_encoder.fit_transform(df_test['Embarked'])

real_predictions = rf_classifier.predict(df_test)

output = pd.DataFrame({'PassengerId': df_test_PId, 'Survived': real_predictions})
```

In [157]:

```
output
```

	PassengerId	Survived
0	892	0
1	893	0
2	894	0
3	895	0
4	896	1
...	...	...
413	1305	0
414	1306	1
415	1307	0
416	1308	0
417	1309	0

418 rows × 2 columns

In [158]:

```
output.to_csv('submission.csv', index=False)
print("Your submission was successfully saved!")
```

```
Your submission was successfully saved!
```

In [141]:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, classification_report
from xgboost import XGBClassifier

X = df_train.drop(columns=columns_to_drop)

y = df_train['Survived']

# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.15, random_state=23)

# Initialize the XGBoost Classifier
xgb_classifier = XGBClassifier(learning_rate=0.1, n_estimators=100, max_depth=3, random_state=42)

# Train the classifier on the training data
xgb_classifier.fit(X_train, y_train)

# Make predictions on the testing data
y_pred = xgb_classifier.predict(X_test)

# Evaluate the model
accuracy = accuracy_score(y_test, y_pred)
print(f'Accuracy: {accuracy:.2f}')

# Display classification report
print('Classification Report:\n', classification_report(y_test, y_pred))
```

```
28
```

```
Accuracy: 0.82
Classification Report:
              precision    recall  f1-score   support

           0       0.84      0.91      0.87        92
           1       0.76      0.62      0.68        42

    accuracy                           0.82       134
   macro avg       0.80      0.77      0.78       134
weighted avg       0.82      0.82      0.82       134
```

In [133]:

```
Collecting xgboost
  Downloading xgboost-2.0.2-py3-none-win_amd64.whl.metadata (2.0 kB)
Requirement already satisfied: numpy in c:\users\vijay\anaconda3\envs\tensorflow_env\lib\site-packages (from xgboost) (1.26.0)
Requirement already satisfied: scipy in c:\users\vijay\anaconda3\envs\tensorflow_env\lib\site-packages (from xgboost) (1.11.3)
Downloading xgboost-2.0.2-py3-none-win_amd64.whl (99.8 MB)
   ---------------------------------------- 0.0/99.8 MB ? eta -:--:--
   ---------------------------------------- 0.1/99.8 MB 1.5 MB/s eta 0:01:06
   ---------------------------------------- 0.3/99.8 MB 3.2 MB/s eta 0:00:32
   ---------------------------------------- 0.7/99.8 MB 5.0 MB/s eta 0:00:20
    --------------------------------------- 1.3/99.8 MB 6.8 MB/s eta 0:00:15
    --------------------------------------- 2.4/99.8 MB 10.1 MB/s eta 0:00:10
   - -------------------------------------- 3.1/99.8 MB 11.1 MB/s eta 0:00:09
   - -------------------------------------- 4.2/99.8 MB 12.7 MB/s eta 0:00:08
   - -------------------------------------- 5.0/99.8 MB 13.8 MB/s eta 0:00:07
   -- ------------------------------------- 6.1/99.8 MB 14.4 MB/s eta 0:00:07
   -- ------------------------------------- 7.4/99.8 MB 15.7 MB/s eta 0:00:06
   --- ------------------------------------ 7.9/99.8 MB 15.3 MB/s eta 0:00:07
   --- ------------------------------------ 9.1/99.8 MB 16.6 MB/s eta 0:00:06
   ---- ----------------------------------- 10.5/99.8 MB 20.5 MB/s eta 0:00:05
   ---- ----------------------------------- 11.8/99.8 MB 22.6 MB/s eta 0:00:04
   ----- ---------------------------------- 12.7/99.8 MB 21.8 MB/s eta 0:00:04
   ----- ---------------------------------- 14.1/99.8 MB 23.4 MB/s eta 0:00:04
   ------ --------------------------------- 15.6/99.8 MB 24.2 MB/s eta 0:00:04
                                            16.7/99.8 MB 25.1 MB/s eta 0:00:04
```

In [146]:
```
df_test = pd.read_csv(r"C:\Users\Vijay\Downloads\test (2).csv")
df_test.isnull().sum()
df_test['Age'].fillna(df['Age'].mean(),inplace = True)
df_test['Fare'].fillna(df['Fare'].mean(),inplace = True)

df_test_PId = df_test["PassengerId"]
df_test = df_test.drop(["PassengerId",'Name', 'Ticket', 'Cabin','SibSp','Parch'], axis=1)
label_encoder = LabelEncoder()
df_test['Sex']  = label_encoder.fit_transform(df_test['Sex'])
df_test['Embarked']  = label_encoder.fit_transform(df_test['Embarked'])

real_predictions = xgb_classifier.predict(df_test)

output = pd.DataFrame({'PassengerId': df_test_PId, 'Survived': real_predictions})
```

In [147]:
```
output.to_csv('submission.csv', index=False)
print("Your submission was successfully saved!")
```

```
Your submission was successfully saved!
```

In [ ]:
```

```