```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns


from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler


df = pd.read_csv("/content/Titanic-Dataset.csv")


from sklearn.impute import SimpleImputer
from sklearn.linear_model import LogisticRegression
from sklearn.compose import ColumnTransformer
from sklearn.preprocessing import OneHotEncoder


import warnings
warnings.filterwarnings('ignore')


df = df.drop(columns = ['PassengerId' , 'Name' , 'Ticket' , 'Cabin'])
df.head(5)
```

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

```
df.isnull().sum()
```

```
Survived      0
Pclass        0
Sex           0
Age         177
SibSp         0
Parch         0
Fare          0
Embarked      2
dtype: int64
```

```
df.describe()
```

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

```
df['Embarked'].value_counts()
```

```
S    644
C    168
Q     77
Name: Embarked, dtype: int64
```

```
df['Sex'].value_counts()
```

```
male      577
female    314
Name: Sex, dtype: int64
```

```
X = df.iloc[:,1:]
X.shape
```

```
(891, 7)
```

```
y = df.iloc[:,:1]
y.shape
```

```
(891, 1)
```

```
X_train , X_test , y_train , y_test = train_test_split(X,y , random_state = 0 , test_size = 0.3)
```

```
transformer  = ColumnTransformer(transformers = [
    ('tf1' , SimpleImputer(), ['Age'] ),
    ('tf2' , OneHotEncoder(sparse = False , drop = 'first' , dtype = np.int32) , ['Sex' , 'Embarked'] )
] , remainder = 'passthrough')
```

```
X_train_transformed = transformer.fit_transform(X_train)
X_test_transformed = transformer.transform(X_test)
```

```
X_train_transformed.shape , X_test_transformed.shape
```

```
((623, 9), (268, 9))
```

```
X_train_transformed = pd.DataFrame(X_train_transformed)
X_test_transformed = pd.DataFrame(X_test_transformed)
```

```
X_train_transformed.head()
```

	0	1	2	3	4	5	6	7	8
0	51.000000	1.0	0.0	1.0	0.0	1.0	0.0	0.0	26.5500
1	49.000000	0.0	0.0	0.0	0.0	1.0	1.0	0.0	76.7292
2	1.000000	1.0	0.0	1.0	0.0	3.0	5.0	2.0	46.9000
3	54.000000	1.0	0.0	1.0	0.0	1.0	0.0	1.0	77.2875
4	29.915339	0.0	0.0	0.0	0.0	3.0	1.0	0.0	14.4583

```
scaler = StandardScaler()
```

```
X_train_scaled = scaler.fit_transform(X_train_transformed)
X_test_scaled = scaler.transform(X_test_transformed)
```

```
X_train_scaled = pd.DataFrame(X_train_scaled)
X_test_scaled = pd.DataFrame(X_test_scaled)
```

```
np.round(X_train.describe() , 1)
```

	Pclass	Age	SibSp	Parch	Fare
count	623.0	502.0	623.0	623.0	623.0
mean	2.3	29.9	0.5	0.4	32.5
std	0.8	14.5	1.2	0.8	48.3
min	1.0	0.7	0.0	0.0	0.0
25%	1.5	21.0	0.0	0.0	7.9
50%	3.0	29.0	0.0	0.0	15.0
75%	3.0	38.0	1.0	0.0	31.4
max	3.0	80.0	8.0	6.0	512.3

```
np.round(X_train_scaled.describe() , 1)
```

	0	1	2	3	4	5	6	7	8
count	623.0	623.0	623.0	623.0	623.0	623.0	623.0	623.0	623.0
mean	-0.0	-0.0	0.0	0.0	-0.0	0.0	-0.0	-0.0	0.0
std	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
min	-2.2	-1.4	-0.3	-1.7	-0.1	-1.5	-0.5	-0.5	-0.7
25%	-0.5	-1.4	-0.3	-1.7	-0.1	-0.9	-0.5	-0.5	-0.5
50%	-0.0	0.7	-0.3	0.6	-0.1	0.8	-0.5	-0.5	-0.4
75%	0.5	0.7	-0.3	0.6	-0.1	0.8	0.4	-0.5	-0.0
max	3.8	0.7	3.2	0.6	17.6	0.8	6.4	6.7	10.0

```
model = LogisticRegression()
```

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

```
▾ LogisticRegression
LogisticRegression()
```

```
y_pred = model.predict(X_test_scaled)
y_pred
```

```
array([0, 0, 0, 1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 1, 0, 0, 0, 0, 0, 1,
       0, 0, 1, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0,
       1, 0, 0, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 0,
       1, 0, 1, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0,
       1, 1, 0, 0, 0, 1, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 0, 0, 1,
       0, 1, 0, 1, 0, 1, 1, 1, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0,
       0, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 1, 1, 1, 0, 1, 1, 0, 0, 1, 1, 0,
       1, 0, 1, 0, 1, 1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 0,
       1, 0, 0, 1, 0, 0, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 1,
       0, 0, 0, 1, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1,
       0, 1, 1, 0, 0, 0, 0, 1, 0, 0, 0, 1, 1, 1, 0, 0, 1, 1, 1, 0, 0, 1,
       0, 0, 1, 0, 1, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0,
       0, 0, 1, 1])
```

```
from sklearn.metrics import accuracy_score
```

```
print('Accuracy Score:' , accuracy_score(y_test , y_pred)*100)
```

```
Accuracy Score: 79.47761194029852
```