

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
df=pd.read_csv("Titanic-Dataset.csv")
```

```
df=pd.read_csv("Titanic-Dataset.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

Next steps: [Generate code with df](#) [View recommended plots](#) [New interactive sheet](#)

```
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

```
df.shape
```

(891, 12)

```
df.info()
```

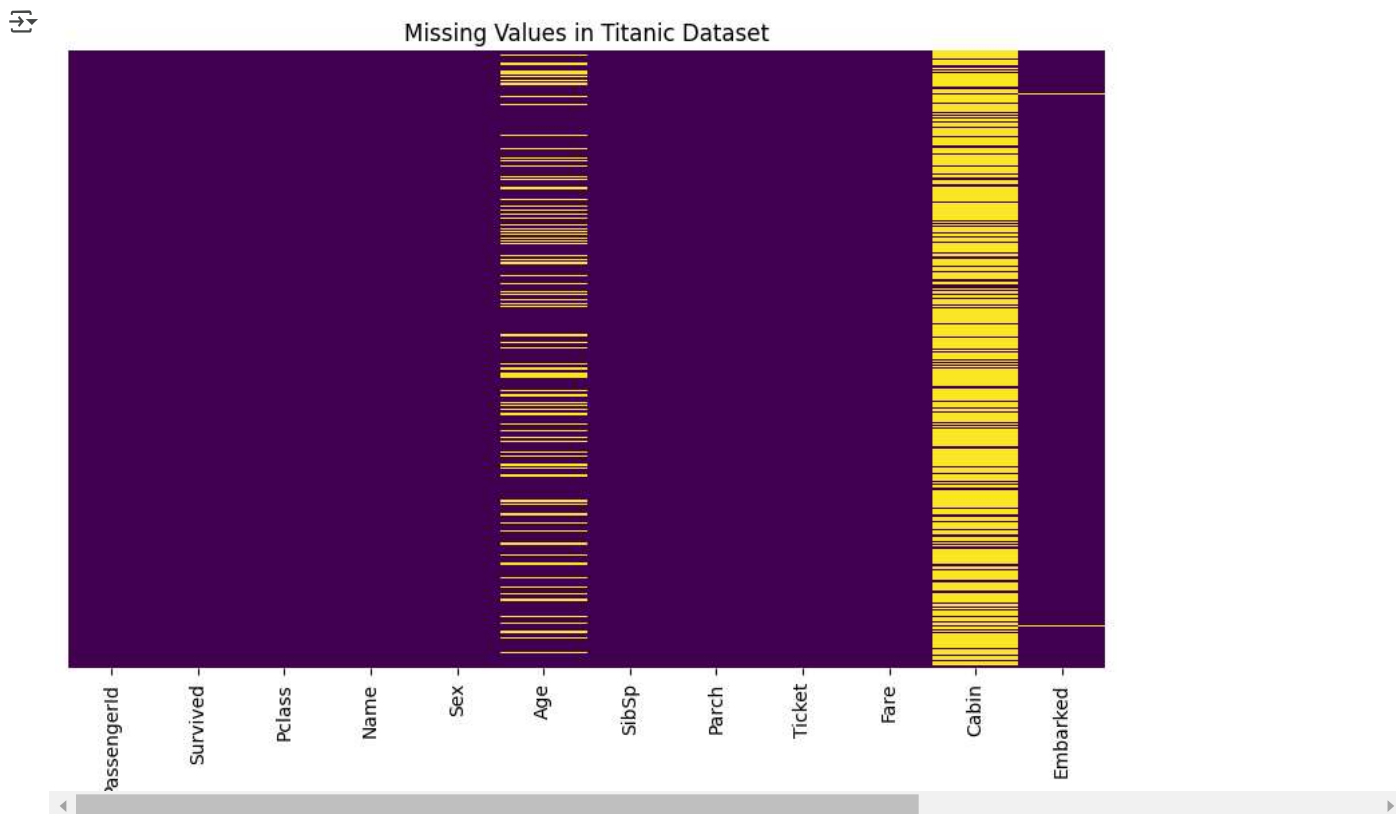
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
#   Column      Non-Null Count  Dtype
---  -
0   PassengerId  891 non-null    int64
1   Survived     891 non-null    int64
2   Pclass       891 non-null    int64
3   Name         891 non-null    object
4   Sex          891 non-null    object
5   Age          714 non-null    float64
6   SibSp        891 non-null    int64
7   Parch        891 non-null    int64
8   Ticket       891 non-null    object
9   Fare         891 non-null    float64
10  Cabin        204 non-null    object
11  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

```
print(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
```

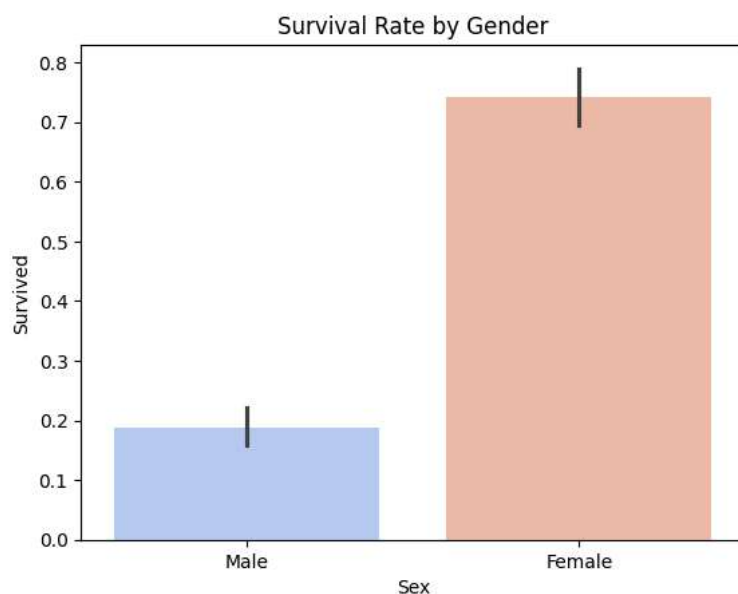
```
plt.figure(figsize=(10, 6))
sns.heatmap(df.isnull(), cmap='viridis', cbar=False, yticklabels=False)
plt.title("Missing Values in Titanic Dataset")
plt.show()
```



```
sns.barplot(x="Sex", y="Survived", data=df, palette="coolwarm")
plt.xticks([0, 1], ["Male", "Female"])
plt.title("Survival Rate by Gender")
plt.show()
```

`<ipython-input-65-d96d199a67b0>:1: FutureWarning:`
 Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and set `legend` to `True`.

```
sns.barplot(x="Sex", y="Survived", data=df, palette="coolwarm")
```



```
plt.figure(figsize=(10, 5))
sns.kdeplot(df[df["Survived"] == 1]["Age"], shade=True, label="Survived", color="green")
sns.kdeplot(df[df["Survived"] == 0]["Age"], shade=True, label="Did Not Survive", color="red")
```

```
plt.title("Age Distribution by Survival")
plt.legend()
plt.show()
```

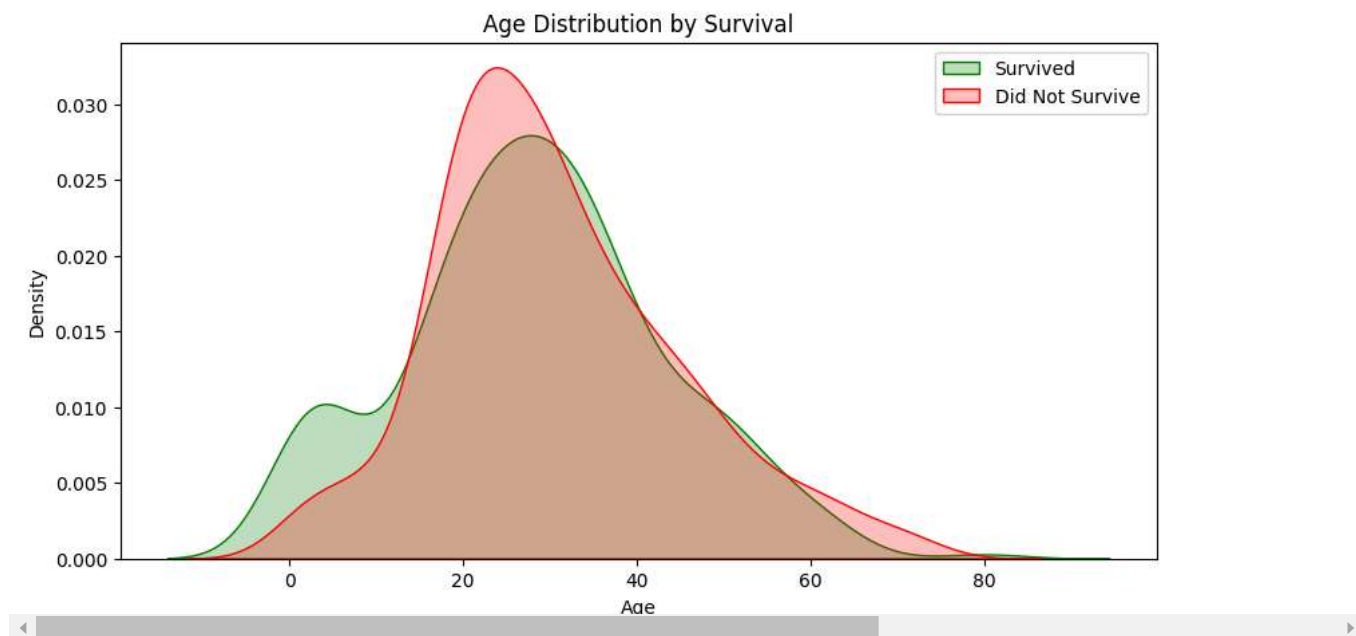
 <ipython-input-66-553fdd08e40c>:2: FutureWarning:

``shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.`

```
sns.kdeplot(df[df["Survived"] == 1]["Age"], shade=True, label="Survived", color="green")
<ipython-input-66-553fdd08e40c>:3: FutureWarning:
```


``shade` is now deprecated in favor of `fill`; setting `fill=True`.
This will become an error in seaborn v0.14.0; please update your code.`

```
sns.kdeplot(df[df["Survived"] == 0]["Age"], shade=True, label="Did Not Survive", color="red")
```




Start coding or [generate](#) with AI.

```
emb_unique=df["Embarked"].unique()
print(emb_unique)
```

 ['S' 'C' 'Q' nan]

```
ports=pd.get_dummies(df.Embarked,prefix='Embarked',dtype=int)
ports.head()
```



	Embarked_C	Embarked_Q	Embarked_S
0	0	0	1
1	1	0	0
2	0	0	1
3	0	0	1
4	0	0	1

Next steps:

[Generate code with ports](#)

[View recommended plots](#)

[New interactive sheet](#)

```
df=df.join(ports)
```

```
df.drop(['Embarked'],axis=1,inplace=True)
```

```
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked_C	Embarked_Q	Embarked_S
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	0	0	1
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	1	0	0
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	0	0	1
				Futrelle, Mrs										

Next steps:

[Generate code with df](#)

[View recommended plots](#)

[New interactive sheet](#)

```
df.Sex=df.Sex.map({'male':0,'female':1})
```

```
df['Sex']
```

	Sex
0	0
1	1
2	1
3	1
4	0
...	...
886	0
887	1
888	1
889	0
890	0

891 rows × 1 columns

```
#splitting the data
Y=df.Survived.copy()
X=df.drop(['Survived'],axis=1)
```

```
# Removing Unnecessary attributes
X.drop(['Cabin','Ticket','Name','PassengerId'],axis=1, inplace=True)
```

```
X.isnull().sum()
```

	0
Pclass	0
Sex	0
Age	177
SibSp	0
Parch	0
Fare	0
Embarked_C	0
Embarked_Q	0
Embarked_S	0

```
X.Age.fillna(X.Age.mean(),inplace=True)
```

↗ <ipython-input-77-d9dd57ff4da2>:1: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment. The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting

For example, when doing 'df[col].method(value, inplace=True)', try using 'df.method({col: value}, inplace=True)' or df[col] = df[col]

```
X.Age.fillna(X.Age.mean(),inplace=True)
```

```
X.isnull().values.any()
```

↗ False

```
#importing libraries for model building
from sklearn.model_selection import train_test_split
```

```
#splitting the data
X_train,X_test,Y_train,Y_test=train_test_split(X,Y,test_size=0.2,random_state=7)
```

```
#importing regression libraries
from sklearn.linear_model import LogisticRegression
```

```
#initializing a name
model = LogisticRegression()
```

```
#Fitting data to the mode
model.fit(X_train,Y_train)
```

↗ /usr/local/lib/python3.11/dist-packages/sklearn/linear_model/_logistic.py:465: ConvergenceWarning: lbfgs failed to converge (status= STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:

<https://scikit-learn.org/stable/modules/preprocessing.html>

Please also refer to the documentation for alternative solver options:

https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression

```
n_iter_i = _check_optimize_result(
```

```
    LogisticRegression ⓘ ?
```

```
LogisticRegression()
```

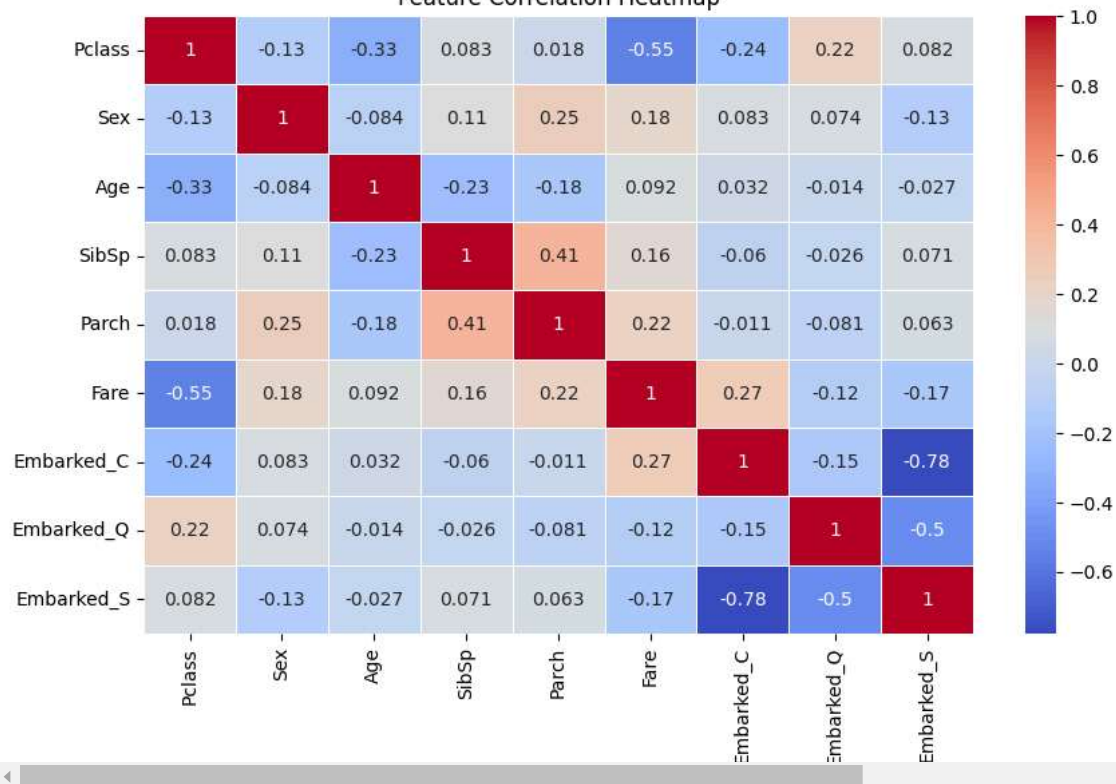
```
model.score(X_train,Y_train)
```

↗ 0.8089887640449438

```
plt.figure(figsize=(10, 6))
sns.heatmap(X.corr(), annot=True, cmap='coolwarm', linewidths=0.5)
plt.title("Feature Correlation Heatmap")
plt.show()
```



Feature Correlation Heatmap



```
#predict on the model
Y_pred_test=model.predict(X_test)
Y_pred_test
```



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

Y_test



Survived

725	0
861	0
528	0
46	0
627	1
...	...
809	1
305	1
538	0
825	0
874	1

179 rows × 1 columns

```
#importing metrics to check accuracy
from sklearn.metrics import accuracy_score
print('Model accuracy score: {0:0.4f}'.format(accuracy_score(Y_test,Y_pred_test)))
```



Model accuracy score: 0.7542

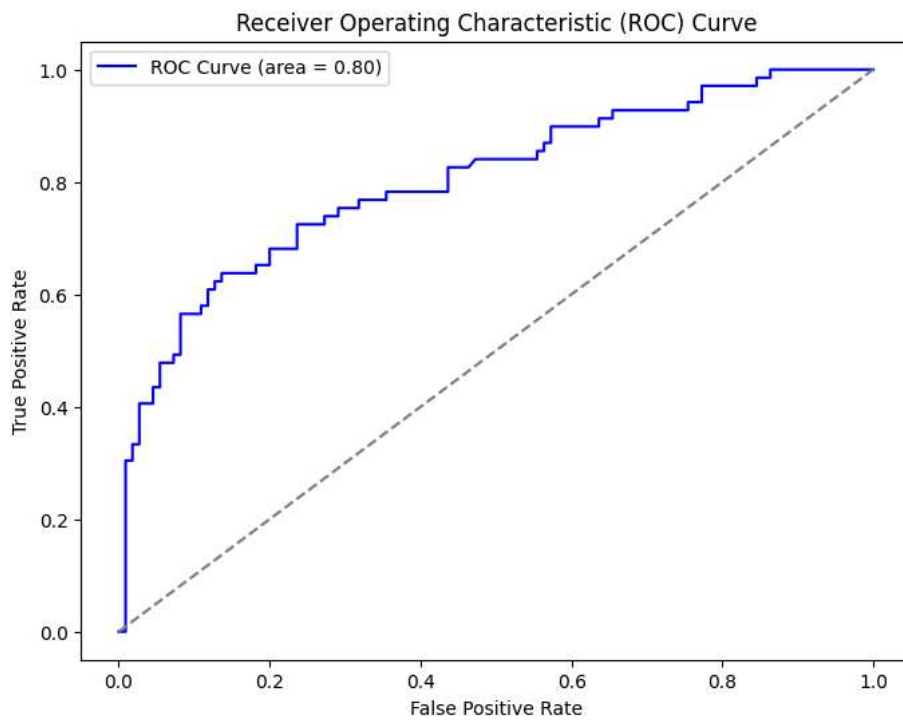
```

from sklearn.metrics import roc_curve, auc

y_prob = model.predict_proba(X_test)[: , 1]
fpr, tpr, _ = roc_curve(Y_test, y_prob)
roc_auc = auc(fpr, tpr)

plt.figure(figsize=(8,6))
plt.plot(fpr, tpr, label=f'ROC Curve (area = {roc_auc:.2f})', color='blue')
plt.plot([0, 1], [0, 1], linestyle='--', color='gray')
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('Receiver Operating Characteristic (ROC) Curve')
plt.legend()
plt.show()

```



Start coding or [generate](#) with AI.