

```
# Importing necessary Libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split
```

```
# Importing the dataset.
dataset=pd.read_csv("Titanic-Dataset.csv")
dataset.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
```

```
# Checking for Null Values.
dataset.isnull().any()
```

```
PassengerId    False
Survived        False
Pclass          False
Name            False
Sex             False
Age             True
SibSp           False
Parch           False
Ticket          False
Fare            False
Cabin           True
Embarked        True
dtype: bool
```

```
dataset.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

```

```
# Handling null values
```

```
# Null values are present in 3 columns - Age, Cabin and Embarked
```

```
# The 'Age' column contains some missing values, replacing those with mean/median of the dataset
dataset['Age'] = dataset['Age'].replace(np.NaN,dataset['Age'].median())
```

```
# As there are too many null values in the 'Cabin' column, removing the entire column is the best option
dataset = dataset.drop(['Cabin'], axis=1)
```

```
# As there are very few null values in 'Embarked' column, removing the corresponding rows is not a good idea
dataset.dropna(subset=['Embarked'],how='any',inplace=True)
```

```
dataset.isnull().sum()
```

```

PassengerId    0
Survived       0
Pclass         0
Name           0
Sex            0
Age           0
SibSp          0
Parch          0
Ticket         0
Fare           0
Embarked       0
dtype: int64

```

```
# Data Visualization.
```

```
# Heatmap
```

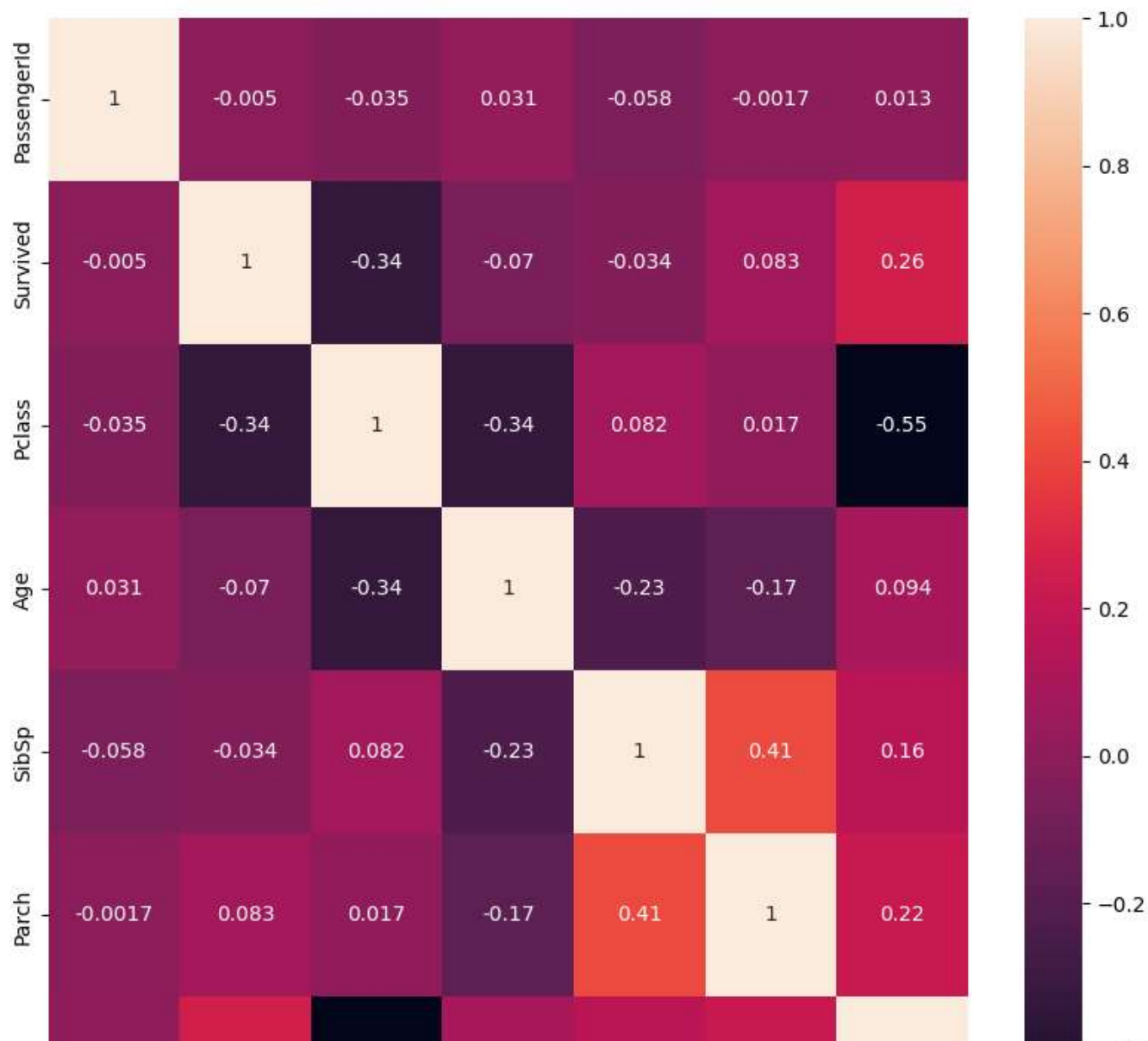
```
corr=dataset.corr()
```

```
plt.subplots(figsize=(10,10))
```

```
sns.heatmap(corr,annot=True)
```

```
<ipython-input-73-af9811d18692>:3: FutureWarning: The default value of numeric_only in [
corr=dataset.corr()
```

```
<Axes: >
```



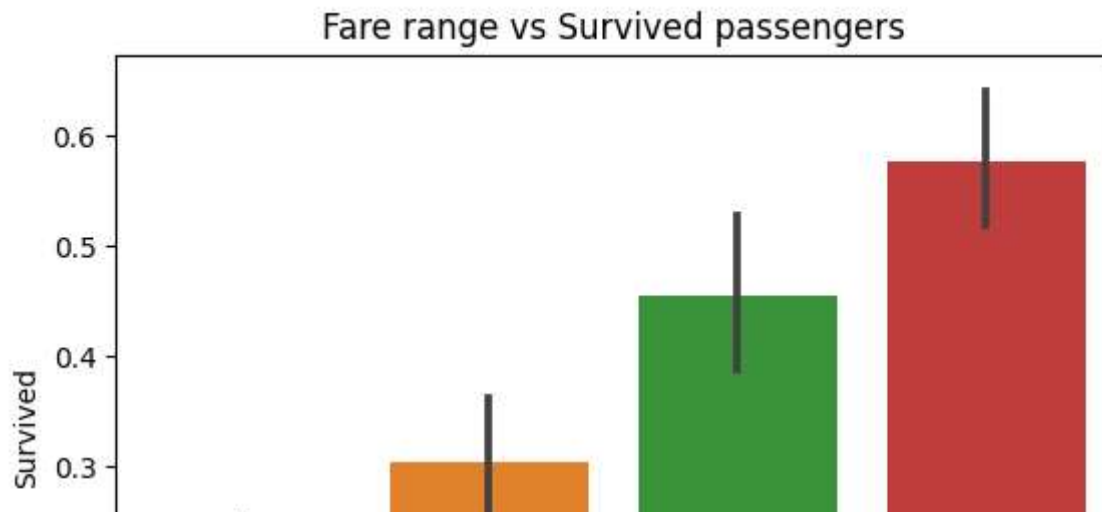
```
#Barplot
```

```
dataset['Fare_Range'] = pd.qcut(dataset['Fare'], 4)
```

```
plt.title('Fare range vs Survived passengers')
```

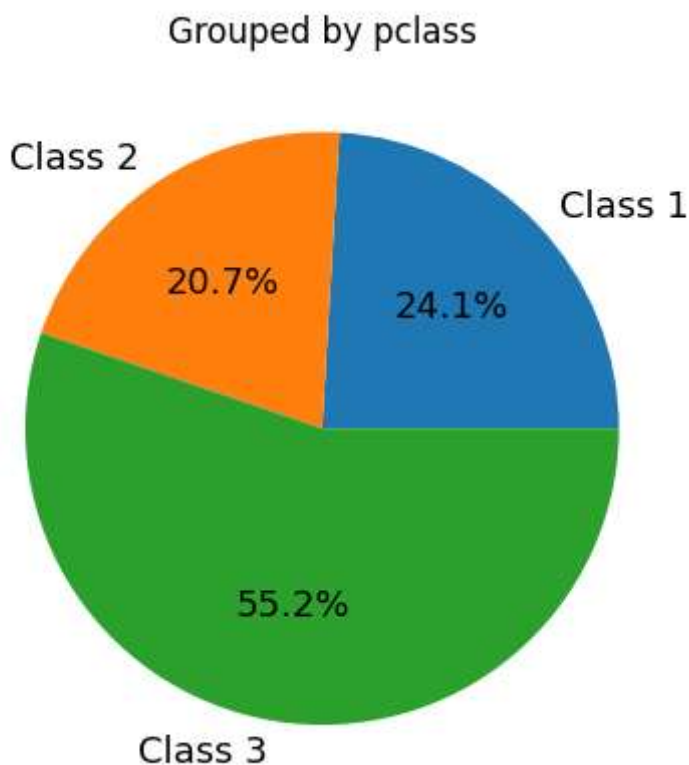
```
sns.barplot(x='Fare_Range', y='Survived', data=dataset)
```

```
<Axes: title={'center': 'Fare range vs Survived passengers'}, xlabel='Fare_Range',
ylabel='Survived'>
```



```
#Piechart
```

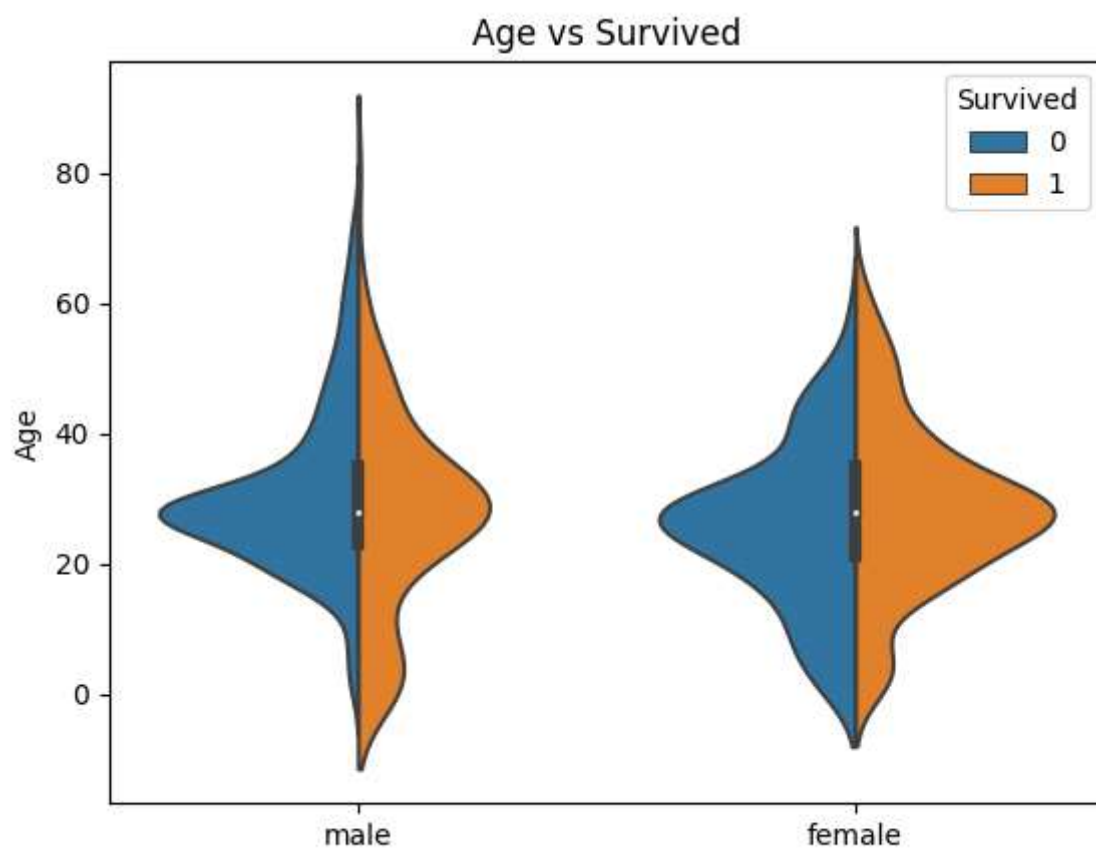
```
pclass_count = dataset.groupby('Pclass')['Pclass'].count()
plt.title('Grouped by pclass')
plt.pie(pclass_count.values, labels=['Class 1', 'Class 2', 'Class 3'], autopct='%1.1f%%', 1
plt.show()
```



```
# Violinplot
```

```
plt.title('Age vs Survived')
sns.violinplot(x="Sex", y="Age", hue="Survived", data = dataset, split = True)
```

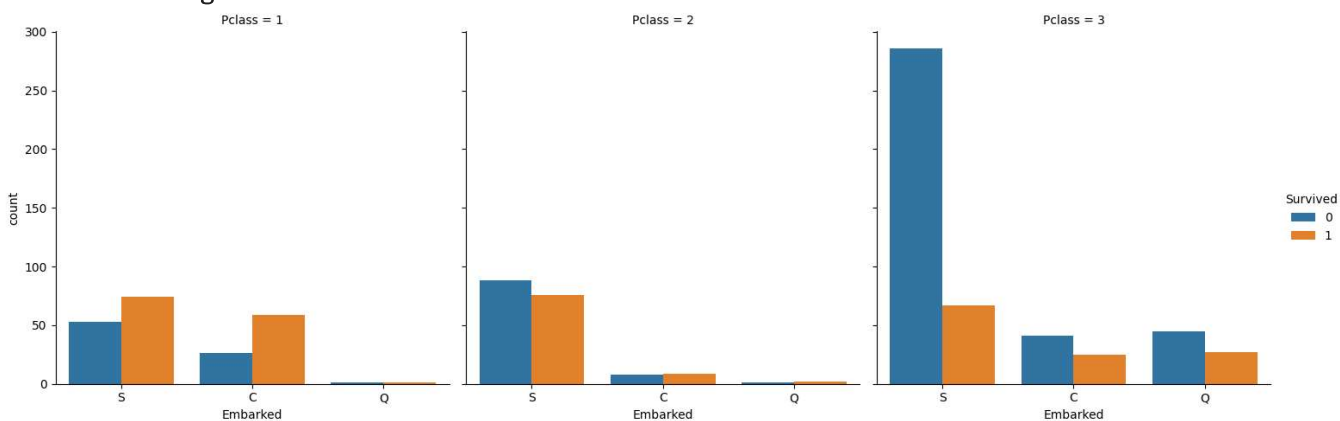
```
<Axes: title={'center': 'Age vs Survived'}, xlabel='Sex', ylabel='Age'>
```



```
# Countplot
```

```
sns.catplot(x='Embarked', hue='Survived', kind='count', col='Pclass', data=dataset)
```

```
<seaborn.axisgrid.FacetGrid at 0x79e5639a7490>
```



```
# Outlier Detection
```

```
sns.boxplot(dataset)
```

&lt;Axes: &gt;



```
# Outliers are present in Age, SibSp, Parch, Fare classes
fig, ax = plt.subplots(1, 4, figsize=(10, 6))
```

```
sns.boxplot(data=dataset['Age'], ax=ax[0], color='brown')
ax[0].set_xlabel('Age')
```

```
sns.boxplot(data=dataset['SibSp'], ax=ax[1], color='green')
ax[1].set_xlabel('Sibsp')
```

```
sns.boxplot(data=dataset['Parch'], ax=ax[2], color='yellow')
ax[2].set_xlabel('Parch')
```

```
sns.boxplot(data=dataset['Fare'], ax=ax[3], color='blue')
ax[3].set_xlabel('Fare')
```

Text(0.5, 0, 'Fare')



## Splitting Dependent and Independent variables

dataset

```
# Independent variables - Name, SibSp, Parch, Ticket
x = dataset.drop(['Name', 'SibSp', 'Parch', 'Ticket'], axis=1)
y = dataset['Survived']
```

dataset

```
# Perform Encoding
# Performing label encoding for Sex and Embarked columns
encoder = LabelEncoder()
x['Sex'] = encoder.fit_transform(x['Sex'])
x['Embarked'] = encoder.fit_transform(x['Embarked'])
```

dataset

```
x.head() # Values in Sex and Embarked columns into numerical values
```

	PassengerId	Survived	Pclass	Sex	Age	Fare	Embarked	Fare_Range	
0	1	0	3	1	22.0	7.2500	2	(-0.001, 7.896]	
1	2	1	1	0	38.0	71.2833	0	(31.0, 512.329]	
2	3	1	3	0	26.0	7.9250	2	(7.896, 14.454]	
3	4	1	1	0	35.0	53.1000	2	(31.0, 512.329]	
4	5	0	3	1	35.0	8.0500	2	(7.896, 14.454]	

```
x=x.drop(['Fare_Range'],axis=1)
# Feature Scaling
scaler = StandardScaler()
x_scaled = scaler.fit_transform(x)
```

x\_scaled

```
array([[ -1.73250451, -0.78696114,  0.82520863, ..., -0.56367407,
        -0.50023975,  0.58683958],
       [ -1.72861124,  1.27071078, -1.57221121, ...,  0.66921696,
         0.78894661, -1.93955453],
       [ -1.72471797,  1.27071078,  0.82520863, ..., -0.25545131,
        -0.48664993,  0.58683958],
       ...,
       [  1.72471797, -0.78696114,  0.82520863, ..., -0.10133993,
        -0.17408416,  0.58683958],
       [  1.72861124,  1.27071078, -1.57221121, ..., -0.25545131,
        -0.0422126 , -1.93955453],
       [  1.73250451, -0.78696114,  0.82520863, ...,  0.20688282,
        -0.49017322, -0.67635748]])
```

```
# Splitting Data into Train and Test
```

```
x_train,x_test,y_train,y_test = train_test_split(x_scaled,y,test_size=0.3,random_state=0)
```

```
print("Shape of x_train:",x_train.shape)
```

```
print("Shape of x_test:",x_test.shape)
```

```
print("Shape of y_train:",y_train.shape)
```

```
print("Shape of y_test:",y_test.shape)
```

```
Shape of x_train: (622, 7)
```

```
Shape of x_test: (267, 7)
```

```
Shape of y_train: (622,)
```

```
Shape of y_test: (267,)
```

---

[Colab paid products](#) - [Cancel contracts here](#)

✓ 0s completed at 10:47 PM

