K-Nearest Neighbor classification with Titanic dataset

```
import numpy as np
import pandas as pd
from sklearn.preprocessing import StandardScaler  #feature Scaling
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score,f1_score,precision_score,recall_score,confusion_matrix

#load the dataset
df=pd.read_csv('Titanic_dataset.csv')
df.head()
```

```
df.head()
```

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500
1	2	1	1	Cumings, Mrs. John Bradley (Florence	female	38.0	1	0	PC 17599	71.2833

```
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.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
```

```
df.dtypes
```

```
PassengerId      int64
Survived         int64
Pclass           int64
Name            object
```

```
Sex            object
Age           float64
SibSp           int64
Parch           int64
Ticket         object
Fare          float64
Cabin          object
Embarked       object
dtype: object
```

```
df.shape
```

```
(891, 12)
```

```
df.columns
```

```
Index(['PassengerId', 'Survived'], dtype='object')
```

```
#let 1:survived,0:not survived
x=df.drop('Survived',axis=1)    #features
y=df['Survived']
x          #labels
```

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

418 rows × 1 columns

```
#Split the dataset
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.2,random_state=34)
```

```
#feature scaling
scaler=StandardScaler()
x_train_scaled=scaler.fit_transform(x_train)
x_test_scaled=scaler.transform(x_test)
```

```
#Initialise the KNN model
knn_model=KNeighborsClassifier(n_neighbors=5)

#Train the model
knn_model.fit(x_train_scaled,y_train)
```

```
▾ KNeighborsClassifier
KNeighborsClassifier()
```

```
#Predictions on training dataset
y_pred=knn_model.predict(x_test_scaled)
```

```
y_pred
```

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

Calculations of Performance Metrics

```
from sklearn.metrics import accuracy_score,f1_score,precision_score,recall_score,confusion_matrix
accuracy=accuracy_score(y_test,y_pred)
precision=precision_score(y_test,y_pred)
recall=recall_score(y_test,y_pred)
f1=f1_score(y_test,y_pred)
conf_matrix=confusion_matrix(y_test,y_pred)

#print metrics
print("Accuracy:::",accuracy)
print("Precision:::",precision)
print("Recall:", recall)
print("F1-Score:", f1)
print("Confusion Matrix:\n", conf_matrix)
```

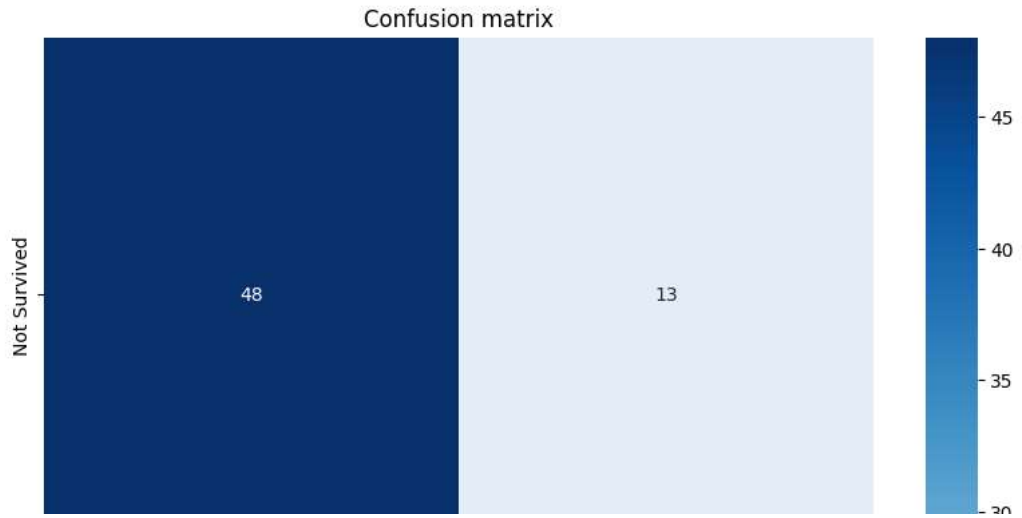
```
Accuracy::: 0.6785714285714286
Precision::: 0.4090909090909091
Recall: 0.391304347826087
F1-Score: 0.4
Confusion Matrix:
 [[48 13]
 [14  9]]
```

```
#Confusion matrix
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.metrics import confusion_matrix

subset_features=['Pclass',  'Sex',  'Age',  'Fare']

conf_matrix=confusion_matrix(y_test,y_pred)

plt.figure(figsize=(10,10))
sns.heatmap(conf_matrix, annot=True,  cmap='Blues', xticklabels=['Not Survived', 'Survived'],
            yticklabels=['Not Survived', 'Survived'])
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.title('Confusion matrix')
plt.show()
```



Visualising the Titanic Dataset

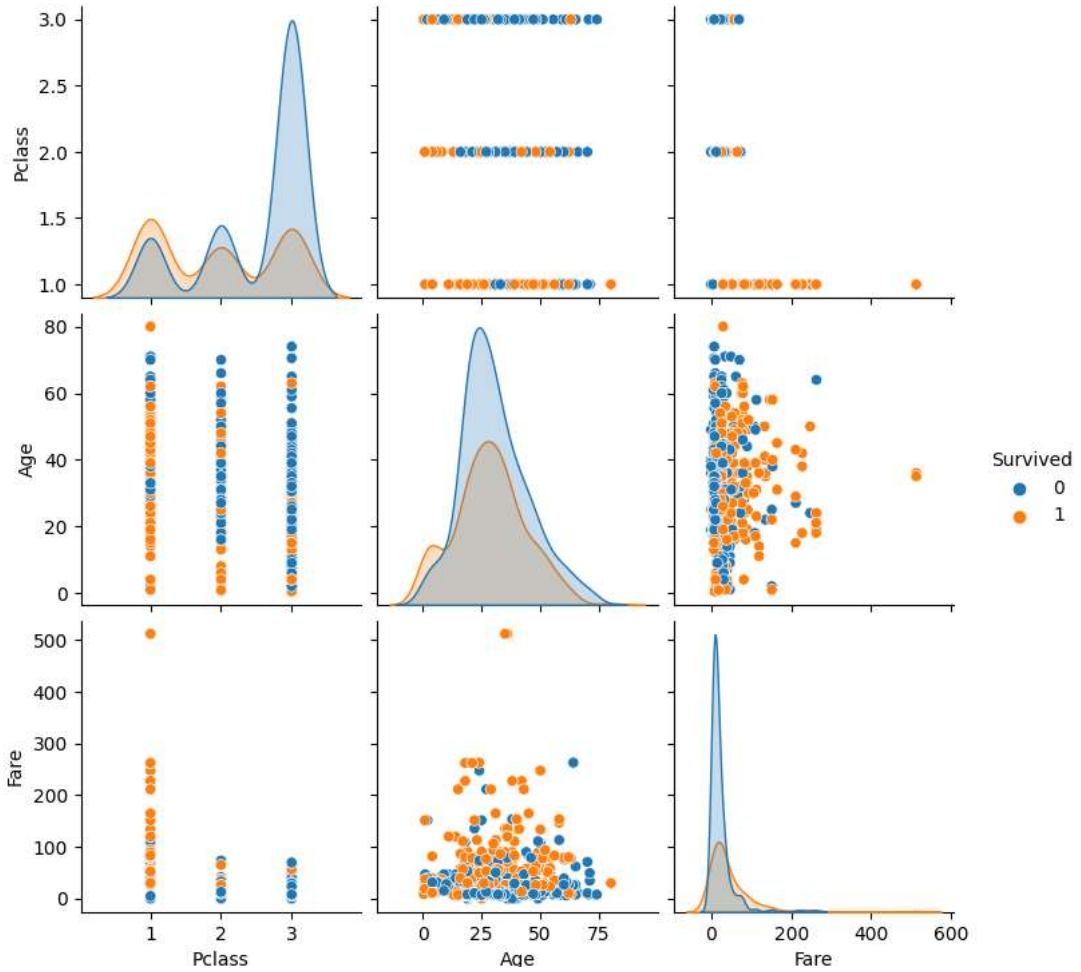
```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
```

```
df=pd.read_csv('titanic.csv')
df
```

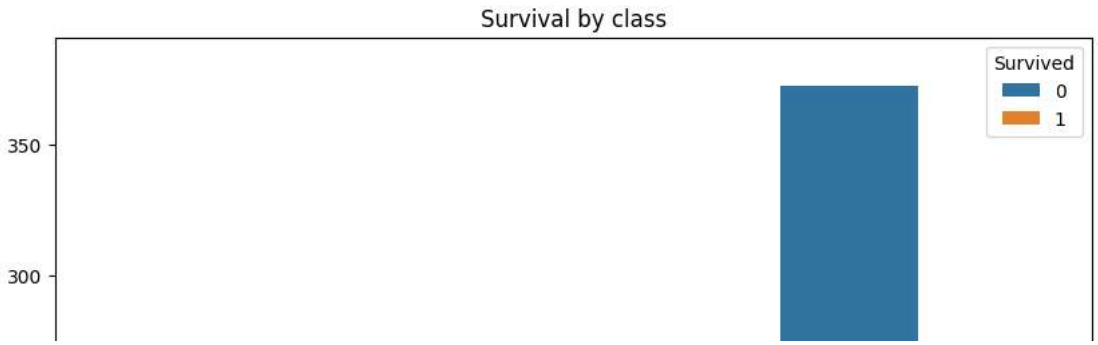
	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 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
...	...	...	...	...	...	...	...	...	...	...	...	...
886	887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S
887	888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	S
888	889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S
889	890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	C

```
subset_features=['Survived','Pclass',   'Sex',  'Age',  'Fare', 'Embarked']
```

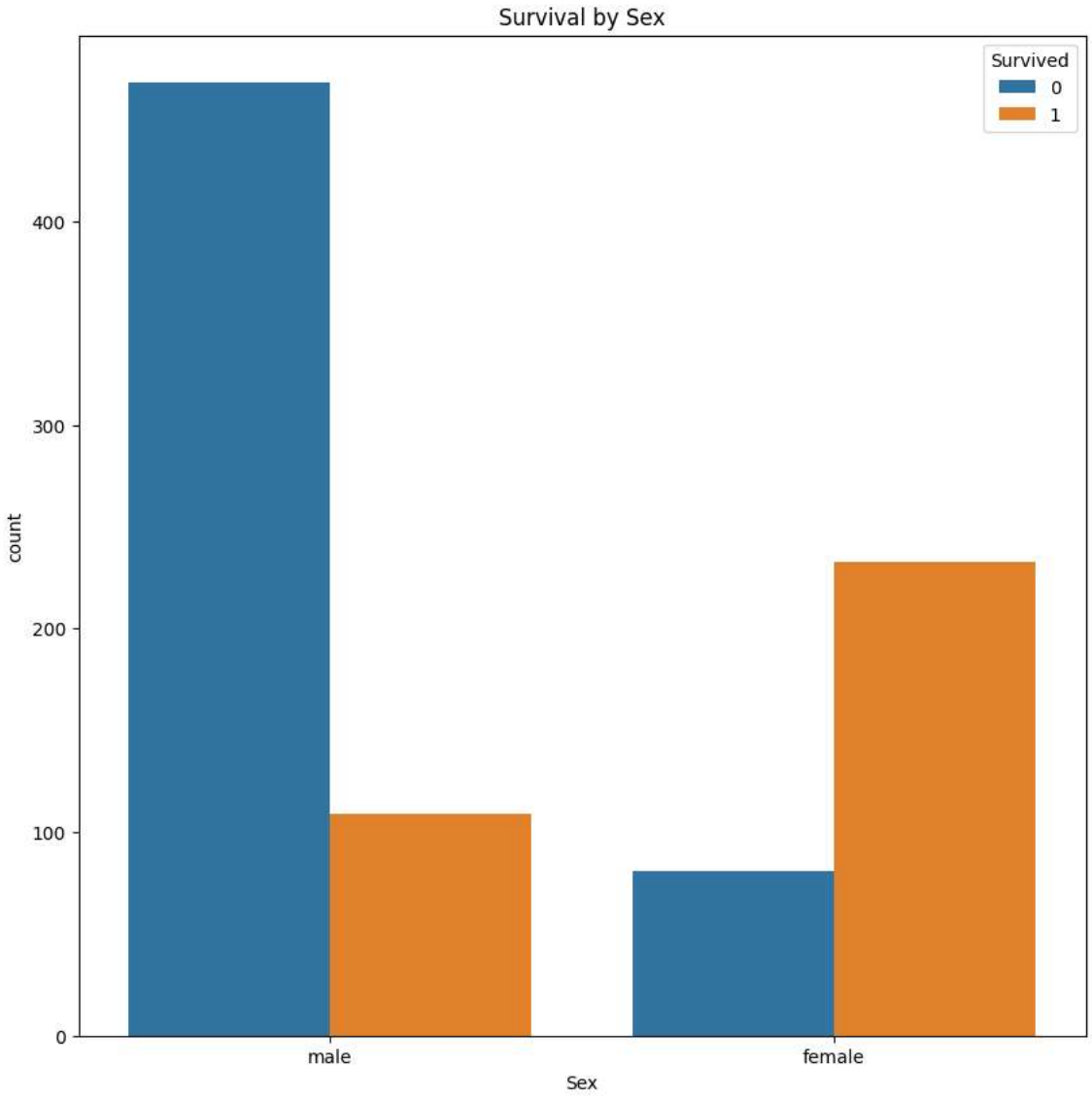
```
#create pairplot for selected features
sns.pairplot(df[subset_features],hue='Survived')
plt.show()
```



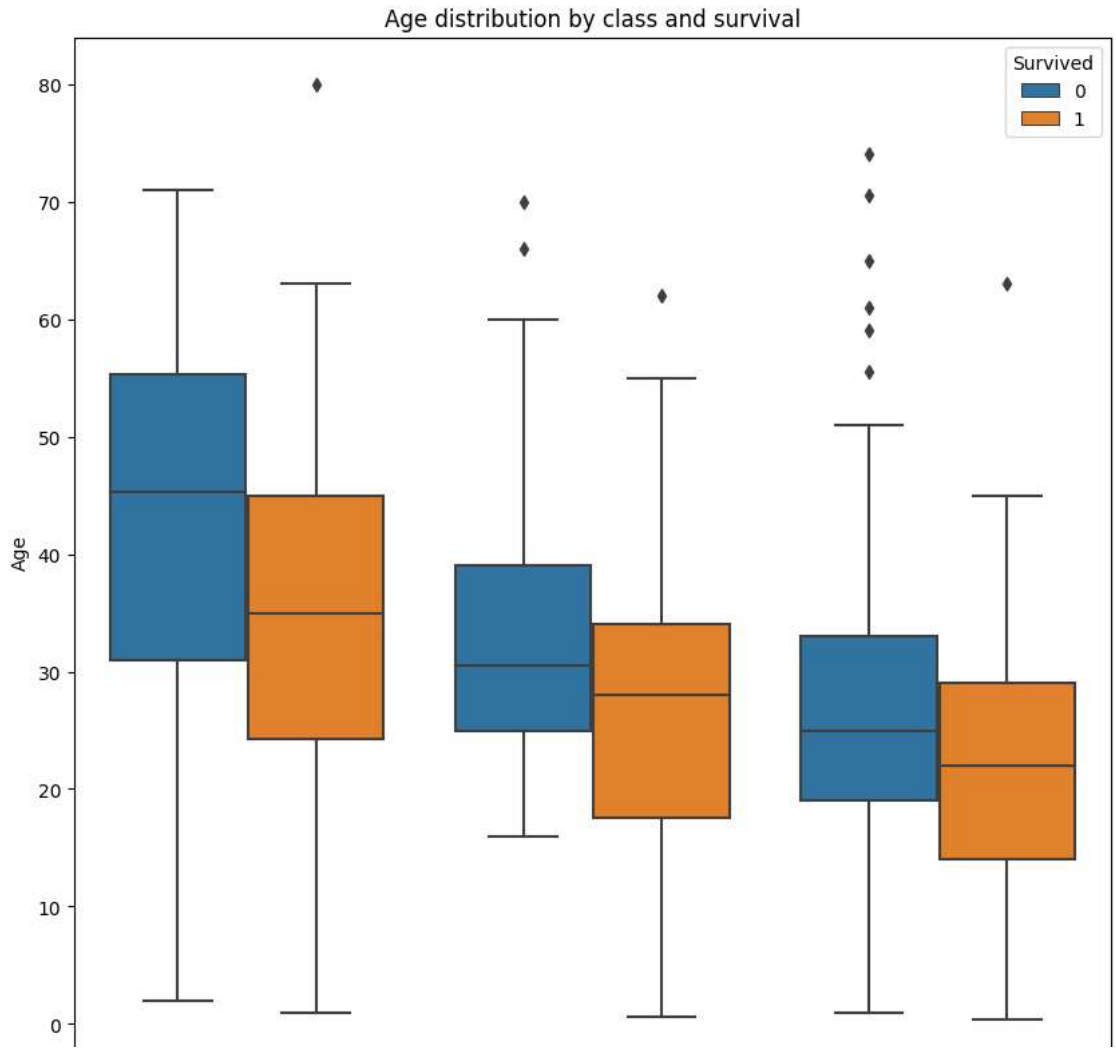

```
#visualise survival by class and sex using barplots
plt.figure(figsize=(10,10))
sns.countplot(data=df,x='Pclass',hue='Survived')
plt.title('Survival by class')
plt.show()
```



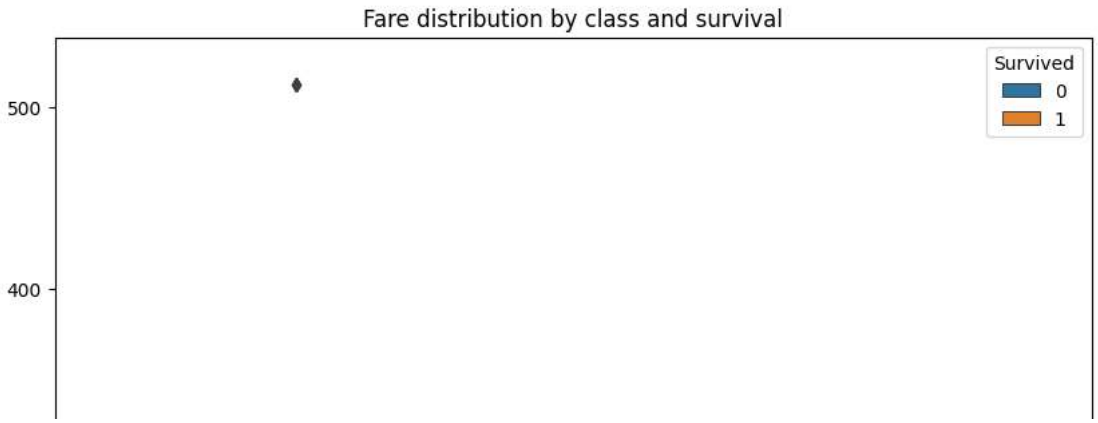

```
#countplot for sex
plt.figure(figsize=(10,10))
sns.countplot(data=df,x='Sex',hue='Survived')
plt.title('Survival by Sex')
plt.show()
```




```
#Visualise age distribution by class and survival
plt.figure(figsize=(10,10))
sns.boxplot(data=df,x='Pclass',y='Age',hue='Survived')
plt.title('Age distribution by class and survival')
plt.show()
```

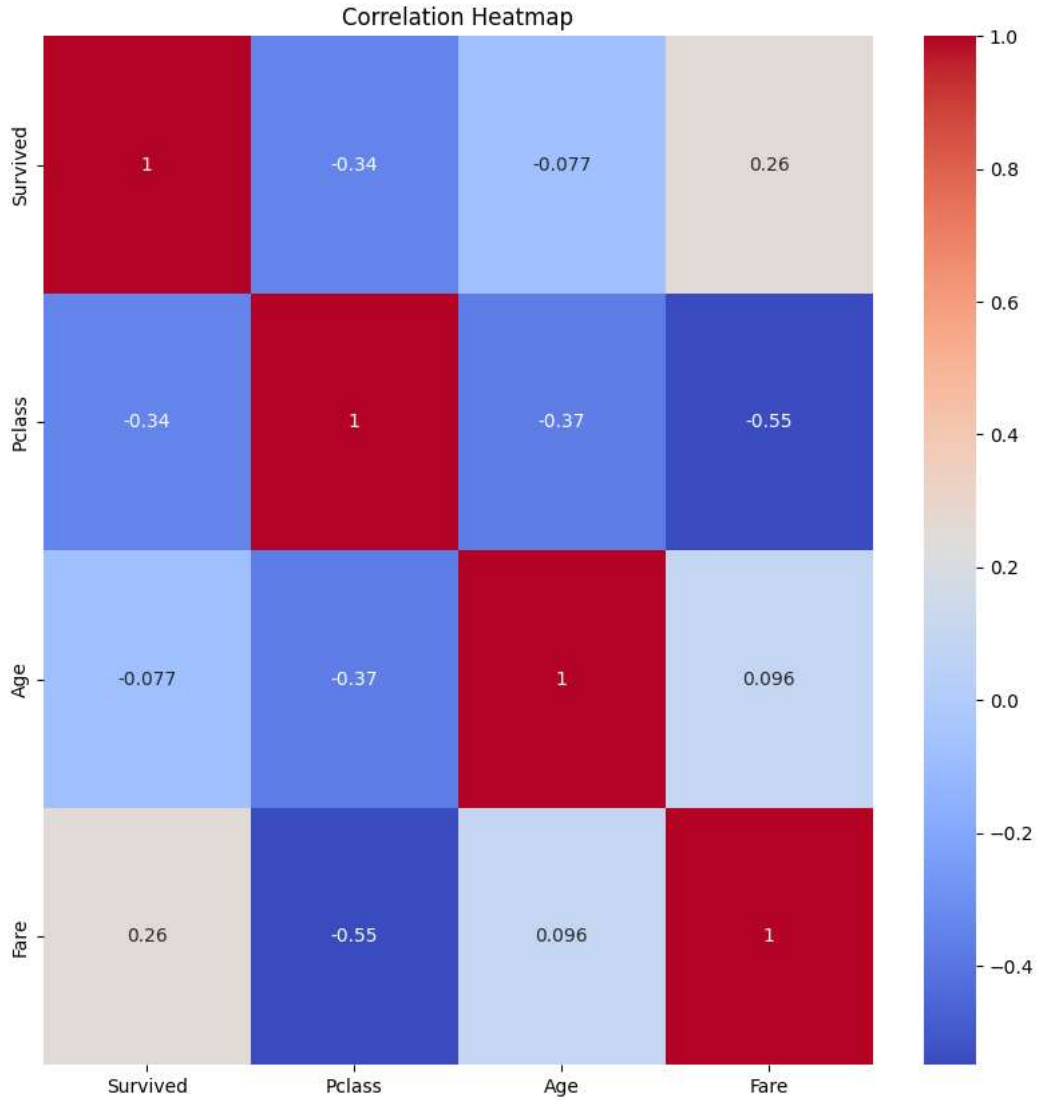



```
#Visualise Fare distribution by class and survival
plt.figure(figsize=(10,10))
sns.boxplot(data=df,x='Pclass',y='Fare',hue='Survived')
plt.title('Fare distribution by class and survival')
plt.show()
```

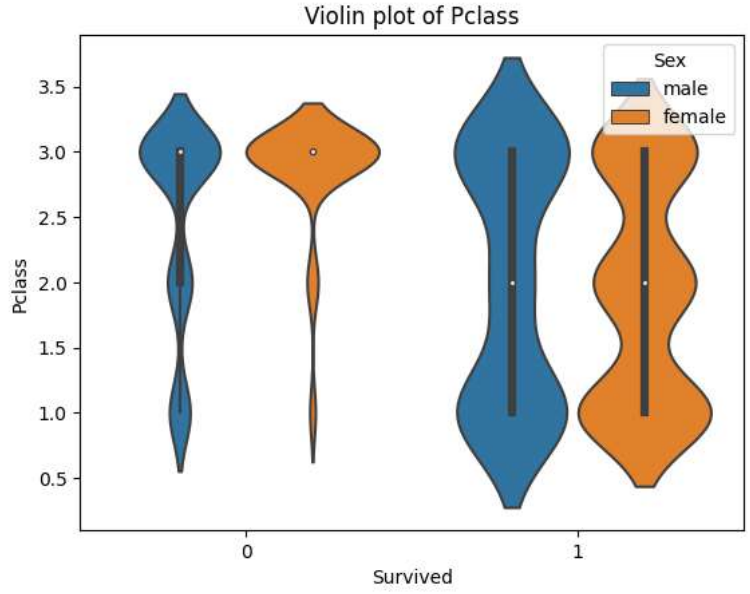



```
#Heatmap to visualise correlations among numerical features
corr_matrix=df[subset_features].corr()
plt.figure(figsize=(10,10))
sns.heatmap(corr_matrix,annot=True,cmap='coolwarm')
plt.title('Correlation Heatmap')
plt.show()
```

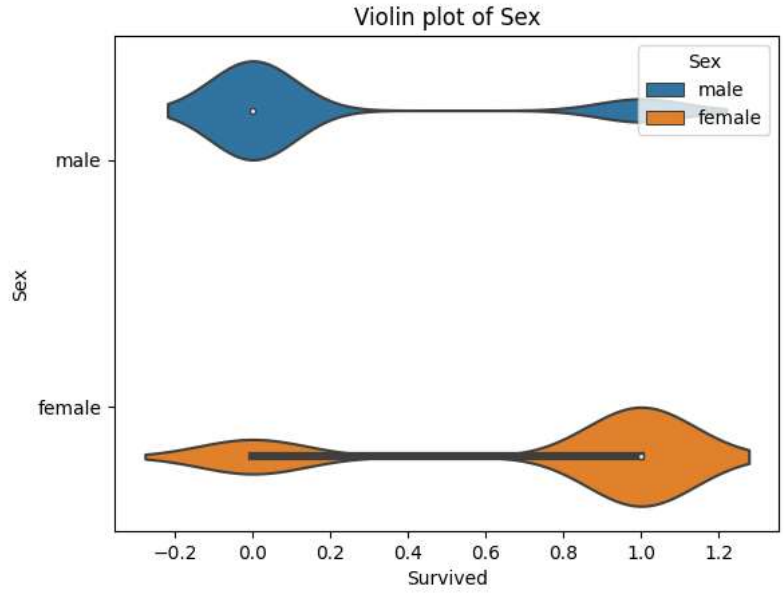
```
<ipython-input-18-0d91dbfabfd2>:2: FutureWarning: The default value of numeric_only in DataFrame.corr is deprecated. In a future version
  corr_matrix=df[subset_features].corr()
```



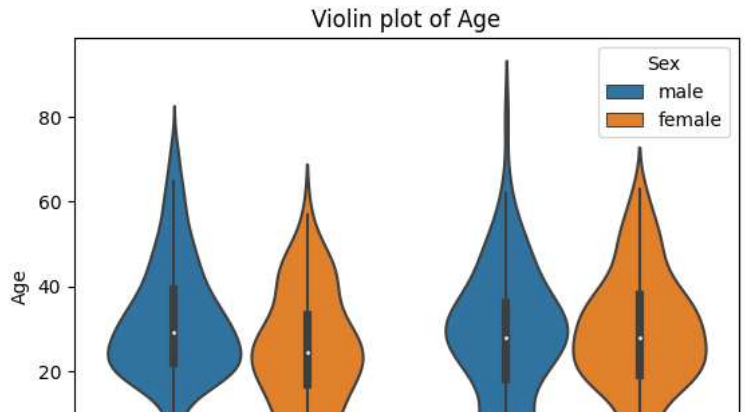
```
import seaborn as sns
#subset_features=['Survived','Pclass',  'Sex',  'Age',  'Fare', 'Embarked']
sns.violinplot(data=df, x='Survived', y='Pclass', hue='Sex')
plt.title('Violin plot of Pclass')
plt.show()
```



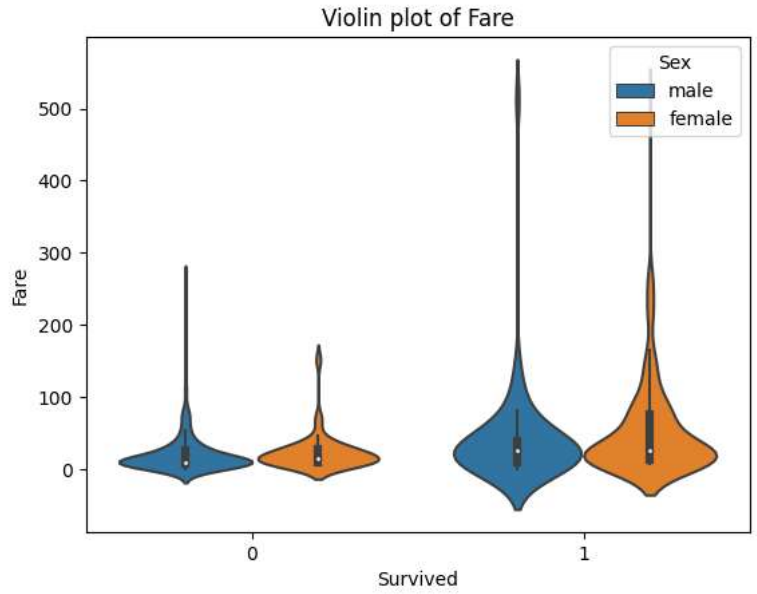

```
import seaborn as sns
#subset_features=['Survived','Pclass',  'Sex',  'Age',  'Fare', 'Embarked']
sns.violinplot(data=df, x='Survived', y='Sex', hue='Sex')
plt.title('Violin plot of Sex')
plt.show()
```



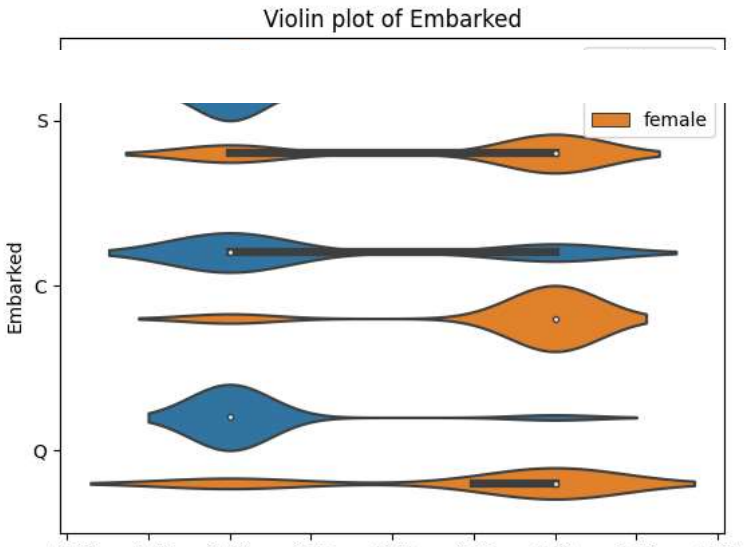

```
import seaborn as sns
#subset_features=['Survived','Pclass',  'Sex',  'Age',  'Fare', 'Embarked']
sns.violinplot(data=df, x='Survived', y='Age', hue='Sex')
plt.title('Violin plot of Age')
plt.show()
```




```
import seaborn as sns
#subset_features=['Survived','Pclass',  'Sex',  'Age',  'Fare', 'Embarked']
sns.violinplot(data=df, x='Survived', y='Fare', hue='Sex')
plt.title('Violin plot of Fare')
plt.show()
```




```
import seaborn as sns
#subset_features=['Survived','Pclass',  'Sex',  'Age',  'Fare', 'Embarked']
sns.violinplot(data=df, x='Survived', y='Embarked', hue='Sex')
plt.title('Violin plot of Embarked')
plt.show()
```


Violin plot of Embarked
female
Embarked
S
C
Q