CODE

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score,precision_score,recall_score,f1_score,classification_report
from sklearn.preprocessing import LabelEncoder
import seaborn as sns
import matplotlib.pyplot as plt

#load the dataset
data=pd.read_csv('titanic_sample.csv')
print(data)

#explore the dataset
print("Dataset Head:\n",data.head())
print("\nDataset Info",data.info())
print("\nMissing Values:\n",data.isnull().sum())

#Visualize some features
sns.countplot(x='Survived',data=data)
plt.title('Survival Count')
plt.show()

sns.countplot(x='Pclass',hue='Survived',data=data)
plt.title('Survival by ticket class')
plt.show()
```

```
sns.histplot(data[data['Survived']==1]['Age'],kde=True,label='Survived',color='green')
sns.histplot(data[data['Survived']==0]['Age'],kde=True,label='Not Survived',color='red')
plt.legend()
plt.title("Age Distribution by survival")
plt.show()

#data cleaning
#fill missing Age values with the median
data['Age'].fillna(data['Age'].median(),inplace=True)
print(data['Age'])

#filling missing Embarked values with the mode
data['Embarked'].fillna(data['Embarked'].mode()[0],inplace=True)
print(data['Embarked'])

#drop cabin(too many missin values)
data.drop('Cabin',axis=1,inplace=True)
print(data.drop)

#Drop irrelevant features
data.drop(['Name','Ticket','PassengerId'],axis=1,inplace=True)
print(data.drop)

#Encode categorical variables
encoder=LabelEncoder()
data['Sex']=encoder.fit_transform(data['Sex'])
data['Embarked']=encoder.fit_transform(data['Embarked'])
print(data['Sex'])
```

```
print(data['Embarked'])

#Feature Engineering
#Create familySize feature
data['FamilySize']=data['SibSp'] + data['Parch'] +1
print(data['FamilySize'])
data.drop(['SibSp','Parch'],axis=1,inplace=True)
print(data.drop)

#split the data into features and target
x=data.drop('Survived',axis=1)
print(x)
y=data['Survived']
print(y)

#train-test split
x_train, x_test, y_train, y_test=train_test_split(x,y,test_size=0.2,random_state=42)
print(x_train,x_test,y_train,y_test)

#Build and train models
#Logistic Regression
log_reg=LogisticRegression(max_iter=1000)
print(log_reg)
log_reg.fit(x_train,y_train)
print(log_reg.fit)

#Random Forest
rf=RandomForestClassifier(random_state=42)
rf.fit(x_train,y_train)
```

```
print(rf)

print(rf.fit)


from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score,
classification_report

from sklearn.ensemble import RandomForestClassifier

from sklearn.linear_model import LogisticRegression


# Assume x_train, x_test, y_train, y_test are already defined


# Define models

log_reg = LogisticRegression(random_state=42)

rf = RandomForestClassifier(random_state=42)


# Train models

log_reg.fit(x_train, y_train)

rf.fit(x_train, y_train)


# Evaluate models

models = {'Logistic Regression': log_reg, 'Random Forest': rf}

for name, model in models.items():

    y_pred = model.predict(x_test)

    print(f"\n{name} Metrics:")

    print("Accuracy:", accuracy_score(y_test, y_pred))

    print("Precision:", precision_score(y_test, y_pred))

    print("Recall:", recall_score(y_test, y_pred))

    print("F1 Score:", f1_score(y_test, y_pred))

    print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

```
# Optimization: Tune Random Forest

tuned_rf = RandomForestClassifier(n_estimators=100, max_depth=5, random_state=42)

tuned_rf.fit(x_train, y_train)

y_pred_tuned = tuned_rf.predict(x_test)  # Predict on x_test, not x_train


print("\nTuned Random Forest Metrics:")

print("Accuracy:", accuracy_score(y_test, y_pred_tuned))

print("Precision:", precision_score(y_test, y_pred_tuned))

print("Recall:", recall_score(y_test, y_pred_tuned))

print("F1 Score:", f1_score(y_test, y_pred_tuned))
```

OUTPUT

```
  PassengerId  Survived  Pclass                                               Name     Sex  Age  SibSp  Parch            Ticket
Fare Cabin Embarked

0           1         0       3                            Braund, Mr. Owen Harris    male   22      1      0         A/5 21171
7.2500   NaN        S

1           2         1       1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female   38      1      0          PC
17599  71.2833   C85        C

2           3         1       3                             Heikkinen, Miss. Laina  female   26      0      0  STON/O2. 3101282
7.9250   NaN        S

3           4         1       1       Futrelle, Mrs. Jacques Heath (Lily May Peel)  female   35      1      0            113803
53.1000  C123        S

4           5         0       3                           Allen, Mr. William Henry    male   35      0      0            373450   8.0500
NaN        S

Dataset Head:

   PassengerId  Survived  Pclass                                               Name     Sex  Age  SibSp  Parch            Ticket
Fare Cabin Embarked

0           1         0       3                            Braund, Mr. Owen Harris    male   22      1      0         A/5 21171
7.2500   NaN        S

1           2         1       1  Cumings, Mrs. John Bradley (Florence Briggs Th...  female   38      1      0          PC
17599  71.2833   C85        C
```

2 3 1 3 Heikkinen, Miss. Laina female 26 0 0 STON/O2. 3101282 7.9250 NaN S

3 4 1 1 Futrelle, Mrs. Jacques Heath (Lily May Peel) female 35 1 0 113803 53.1000 C123 S

4 5 0 3 Allen, Mr. William Henry male 35 0 0 373450 8.0500 NaN S

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 5 entries, 0 to 4

Data columns (total 12 columns):

```
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  5 non-null      int64
 1   Survived     5 non-null      int64
 2   Pclass       5 non-null      int64
 3   Name         5 non-null      object
 4   Sex          5 non-null      object
 5   Age          5 non-null      int64
 6   SibSp        5 non-null      int64
 7   Parch        5 non-null      int64
 8   Ticket       5 non-null      object
 9   Fare         5 non-null      float64
 10  Cabin        2 non-null      object
 11  Embarked     5 non-null      object
```

dtypes: float64(1), int64(6), object(5)

memory usage: 612.0+ bytes

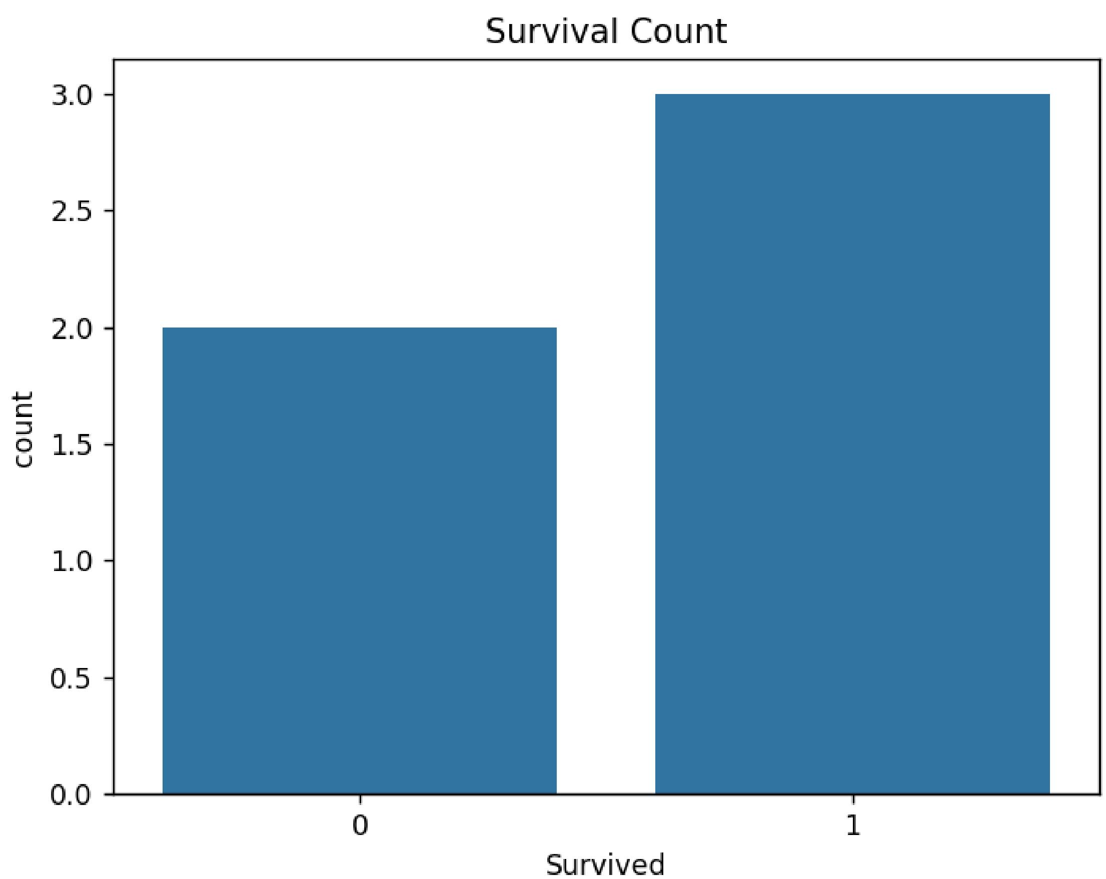
Dataset Info None

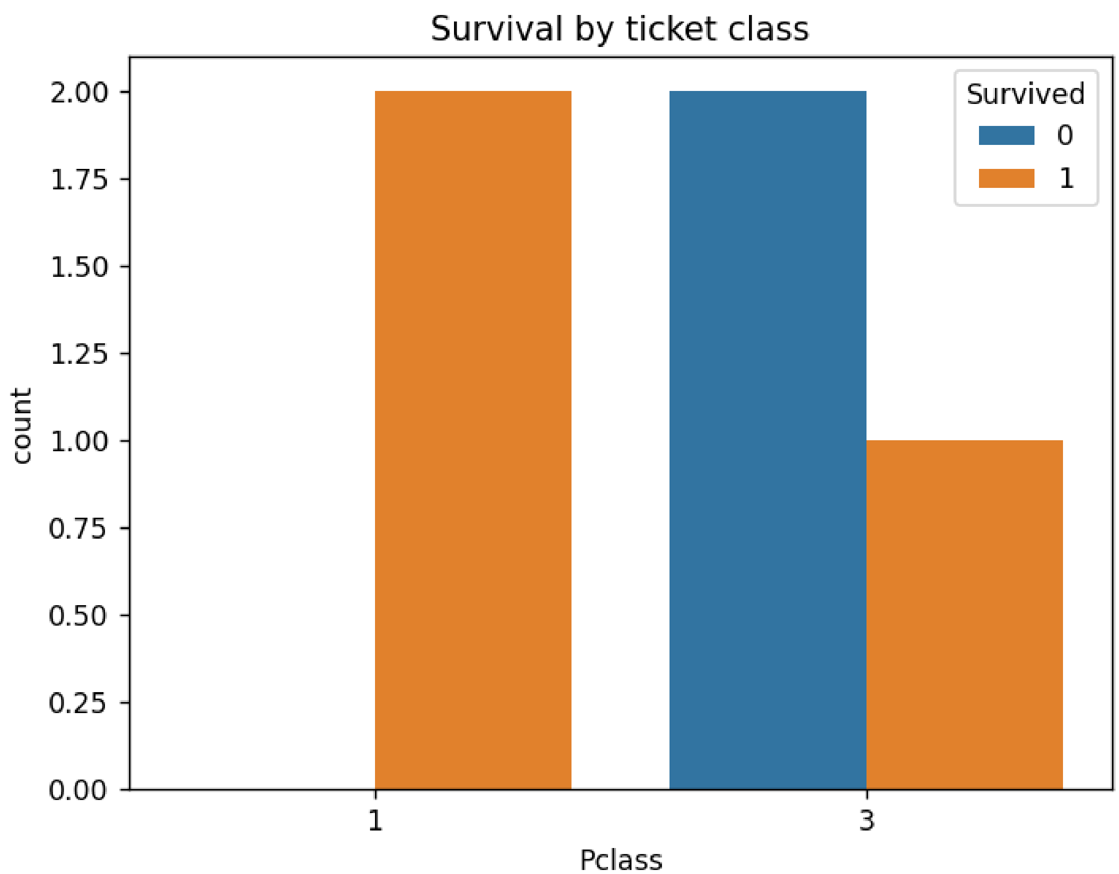
Missing Values:

PassengerId 0

Survived	0
Pclass	0
Name	0
Sex	0
Age	0
SibSp	0
Parch	0
Ticket	0
Fare	0
Cabin	3
Embarked	0

dtype: int64





data['Age'].fillna(data['Age'].median(),inplace=True)

0 22

1 38

2 26

3 35

4 35

Name: Age, dtype: int64

data['Embarked'].fillna(data['Embarked'].mode()[0],inplace=True)

0 S

1   C

2   S

3   S

4   S

Name: Embarked, dtype: object

<bound method DataFrame.drop of   PassengerId Survived Pclass

 Name   Sex Age SibSp Parch   Ticket   Fare Embarked

0   1   0   3   Braund, Mr. Owen Harris   male   22   1   0   A/5 21171
7.2500   S

1   2   1   1 Cumings, Mrs. John Bradley (Florence Briggs Th... female   38   1   0   PC
17599 71.2833   C

2   3   1   3   Heikkinen, Miss. Laina female   26   0   0 STON/O2. 3101282
7.9250   S

3   4   1   1   Futrelle, Mrs. Jacques Heath (Lily May Peel) female   35   1   0   113803
53.1000   S

4   5   0   3   Allen, Mr. William Henry   male   35   0   0   373450   8.0500
S>

<bound method DataFrame.drop of   Survived Pclass   Sex Age SibSp Parch   Fare Embarked

0   0   3   male   22   1   0   7.2500   S

1   1   1 female   38   1   0 71.2833   C

2   1   3 female   26   0   0   7.9250   S

3   1   1 female   35   1   0 53.1000   S

4   0   3   male   35   0   0   8.0500   S>

0   1

1   0

2   0

3   0

4   1

Name: Sex, dtype: int64

0   1

```
1    0
2    1
3    1
4    1
Name: Embarked, dtype: int64
0    2
1    2
2    1
3    2
4    1
Name: FamilySize, dtype: int64
<bound method DataFrame.drop of    Survived  Pclass  Sex  Age     Fare  Embarked  FamilySize
0         0       3    1   22   7.2500         1           2
1         1       1    0   38  71.2833         0           2
2         1       3    0   26   7.9250         1           1
3         1       1    0   35  53.1000         1           2
4         0       3    1   35   8.0500         1           1>
   Pclass  Sex  Age     Fare  Embarked  FamilySize
0       3    1   22   7.2500         1           2
1       1    0   38  71.2833         0           2
2       3    0   26   7.9250         1           1
3       1    0   35  53.1000         1           2
4       3    1   35   8.0500         1           1
0    0
1    1
2    1
3    1
4    0
Name: Survived, dtype: int64
```

```
   Pclass  Sex  Age    Fare  Embarked  FamilySize
4      3    1   35   8.050        1          1
2      3    0   26   7.925        1          1
0      3    1   22   7.250        1          2
3      1    0   35  53.100        1          2    Pclass  Sex  Age    Fare  Embarked  FamilySize
1      1    0   38  71.2833        0          2 4    0
2    1
0    0
3    1
Name: Survived, dtype: int64 1    1
Name: Survived, dtype: int64
LogisticRegression(max_iter=1000)
<bound method LogisticRegression.fit of LogisticRegression(max_iter=1000)>
RandomForestClassifier(random_state=42)
<bound method BaseForest.fit of RandomForestClassifier(random_state=42)>

Logistic Regression Metrics:
Accuracy: 1.0
Precision: 1.0
Recall: 1.0
F1 Score: 1.0

Classification Report:
              precision    recall  f1-score   support

           1       1.00      1.00      1.00         1

    accuracy                           1.00         1
   macro avg       1.00      1.00      1.00         1
```

weighted avg	1.00	1.00	1.00	1

Random Forest Metrics:

Accuracy: 1.0

Precision: 1.0

Recall: 1.0

F1 Score: 1.0

Classification Report:

	precision	recall	f1-score	support
1	1.00	1.00	1.00	1
accuracy			1.00	1
macro avg	1.00	1.00	1.00	1
weighted avg	1.00	1.00	1.00	1

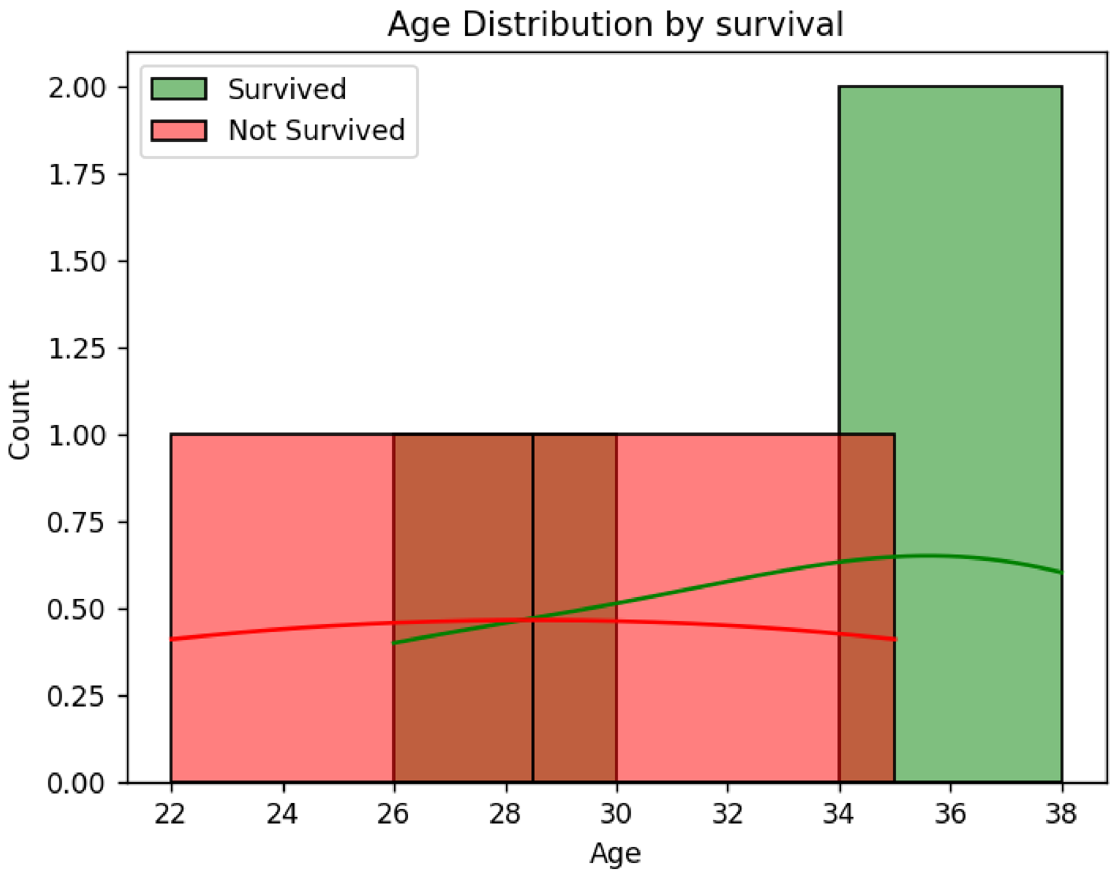
Tuned Random Forest Metrics:

Accuracy: 1.0

Precision: 1.0

Recall: 1.0

F1 Score: 1.0


Age Distribution by survival
Survived
Not Survived
Count
Age
2.00
1.75
1.50
1.25
1.00
0.75
0.50
0.25
0.00
22
24
26
28
30
32
34
36
38