Import libraries

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report
```

Load the dataset (directly from GitHub or Kaggle link)

```
url = "https://raw.githubusercontent.com/datasciencedojo/datasets/master/titanic.csv"
data = pd.read_csv(url)
data
```

	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
890	891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	Q

891 rows × 12 columns

Next steps: Generate code with data | View recommended plots | New interactive sheet

Data Preprocessing

Fill missing values

```
data['Age'].fillna(data['Age'].median(), inplace=True)
data['Embarked'].fillna('S', inplace=True)
print(data)
```

```
     PassengerId  Survived  Pclass  \
0              1         0       3   
1              2         1       1   
2              3         1       3   
3              4         1       1   
4              5         0       3   
..           ...       ...     ...   
886          887         0       2   
887          888         1       1   
888          889         0       3   
889          890         1       1   
890          891         0       3   

                                                  Name     Sex   Age  SibSp  \
0                              Braund, Mr. Owen Harris    male  22.0      1   
1    Cumings, Mrs. John Bradley (Florence Briggs Th...  female  38.0      1   
2                               Heikkinen, Miss. Laina  female  26.0      0   
3         Futrelle, Mrs. Jacques Heath (Lily May Peel)  female  35.0      1   
4                             Allen, Mr. William Henry    male  35.0      0   
..                                                 ...     ...   ...    ...   
886                              Montvila, Rev. Juozas    male  27.0      0   
887                       Graham, Miss. Margaret Edith  female  19.0      0   
888           Johnston, Miss. Catherine Helen "Carrie"  female  28.0      1   
889                              Behr, Mr. Karl Howell    male  26.0      0   
890                                Dooley, Mr. Patrick    male  32.0      0   

     Parch            Ticket     Fare Cabin Embarked  
0        0         A/5 21171   7.2500   NaN        S  
1        0          PC 17599  71.2833   C85        C  
2        0  STON/O2. 3101282   7.9250   NaN        S  
3        0            113803  53.1000  C123        S  
4        0            373450   8.0500   NaN        S  
..     ...               ...      ...   ...      ...  
886      0            211536  13.0000   NaN        S  
887      0            112053  30.0000   B42        S  
888      2        W./C. 6607  23.4500   NaN        S  
889      0            111369  30.0000  C148        C  
890      0            370376   7.7500   NaN        Q  

[891 rows x 12 columns]
<ipython-input-13-9bf9d701e241>:1: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.

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

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

Drop irrelevant features

```
data = data.drop(['cabin', 'name', 'ticket'], axis=1, errors='ignore')
data.head()
```

	PassengerId	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	1	0	3	male	22.0	1	0	7.2500	S
1	2	1	1	female	38.0	1	0	71.2833	C
2	3	1	3	female	26.0	0	0	7.9250	S
3	4	1	1	female	35.0	1	0	53.1000	S
4	5	0	3	male	35.0	0	0	8.0500	S

Next steps: Generate code with data | View recommended plots | New interactive sheet

One-hot encoding for categorical variables

```
print("Columns before processing:", data.columns)
data.columns = data.columns.str.lower()
data['age'].fillna(data['age'].median(), inplace=True)
columns_to_drop = ['cabin', 'name', 'ticket']
for col in columns_to_drop:
    if col in data.columns:
        data.drop(col, axis=1, inplace=True)
if 'sex' in data.columns and 'embarked' in data.columns:
    data = pd.get_dummies(data, columns=['sex', 'embarked'], drop_first=True)
else:
    print("Columns 'sex' and 'embarked' are not found in the dataset.")
print(data)
```

```
Columns before processing: Index(['passengerid', 'survived', 'pclass', 'age', 'sibsp', 'parch', 'fare',
       'sex_male', 'embarked_q', 'embarked_s'],
      dtype='object')
Columns 'sex' and 'embarked' are not found in the dataset.
     passengerid  survived  pclass   age  sibsp  parch     fare  sex_male  \
0              1         0       3  22.0      1      0   7.2500      True
1              2         1       1  38.0      1      0  71.2833     False
2              3         1       3  26.0      0      0   7.9250     False
```

```
3              4         1       1  35.0      1      0  53.1000     False
4              5         0       3  35.0      0      0   8.0500      True
..           ...       ...     ...   ...    ...    ...      ...       ...
886          887         0       2  27.0      0      0  13.0000      True
887          888         1       1  19.0      0      0  30.0000     False
888          889         0       3  28.0      1      2  23.4500     False
889          890         1       1  26.0      0      0  30.0000      True
890          891         0       3  32.0      0      0   7.7500      True

     embarked_q  embarked_s
0         False        True
1         False       False
2         False        True
3         False        True
4         False        True
..          ...         ...
886       False        True
887       False        True
888       False        True
889       False       False
890        True       False

[891 rows x 10 columns]
<ipython-input-30-48250e006ad7>:3: FutureWarning: A value is trying to be set on a copy of a DataFrame or Series through chained assignment using an inplace method.
The behavior will change in pandas 3.0. This inplace method will never work because the intermediate object on which we are setting values always behaves as a copy.

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


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

Feature selection

```
X = data.drop('survived', axis=1)
y = data['survived']
```

Train-test split

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

Train the model

```
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)
```

```
▾     RandomForestClassifier
RandomForestClassifier(random_state=42)
```

Make predictions and evaluate

```
y_pred = model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print("\nClassification Report:\n", classification_report(y_test, y_pred))
```

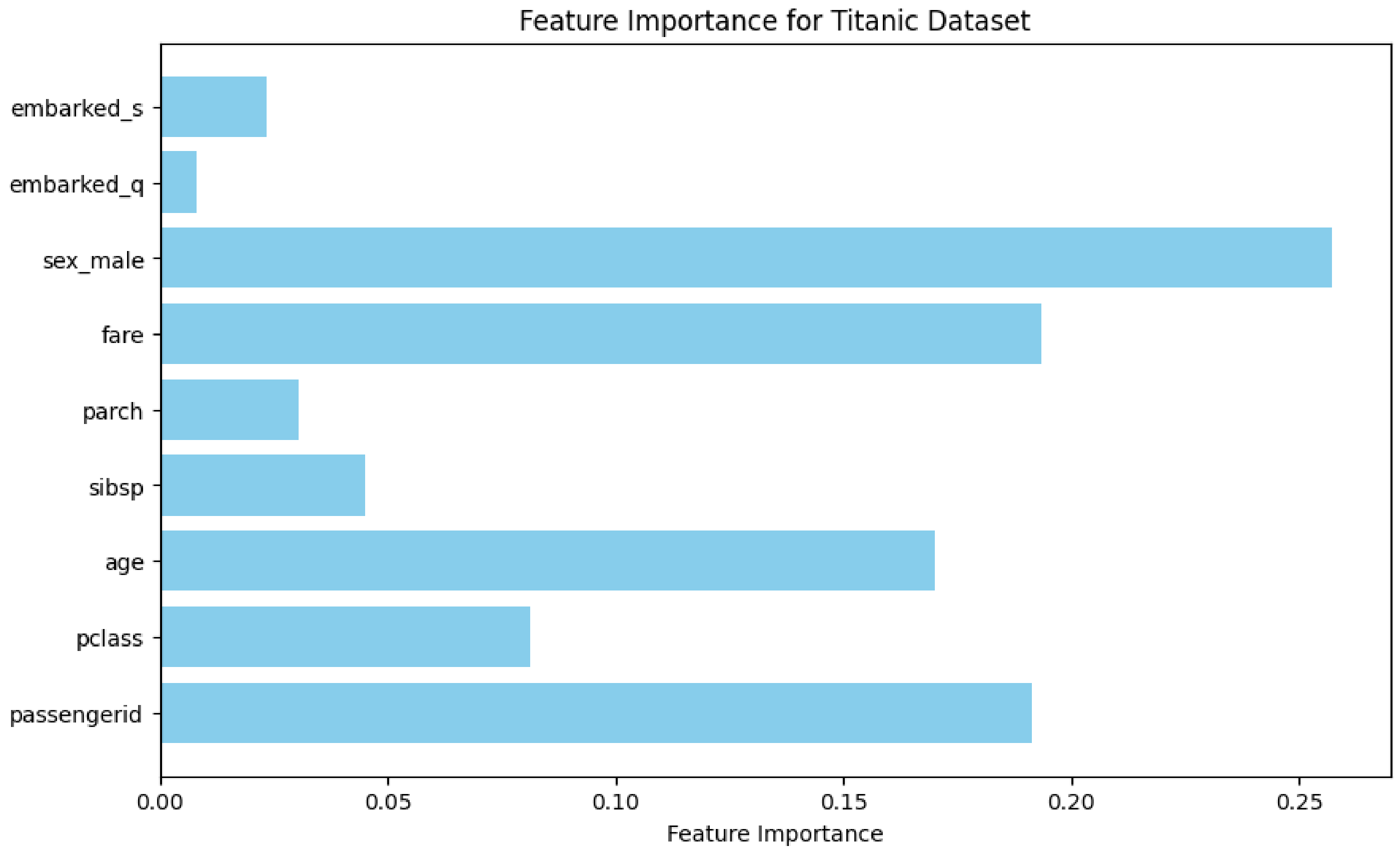
```
Accuracy: 0.8212290502793296

Classification Report:
               precision    recall  f1-score   support

           0       0.83      0.88      0.85       105
           1       0.81      0.74      0.77        74

    accuracy                           0.82       179
   macro avg       0.82      0.81      0.81       179
weighted avg       0.82      0.82      0.82       179
```

Feature Importance

```
import matplotlib.pyplot as plt
```

Plot feature importance

```
feature_importances = model.feature_importances_
features = X.columns
plt.figure(figsize=(10, 6))
plt.barh(features, feature_importances, color='skyblue')
plt.xlabel("Feature Importance")
plt.title("Feature Importance for Titanic Dataset")
plt.show()
```



Insights from the Predictions

The model's predictions determine whether each passenger in the test dataset survived or not.The evaluation metrics (accuracy, precision, recall) help assess how well the model performs this task.The feature importance plot provides insights into which factors were most influential in predicting survival, such as:

Gender (Sex_male): Being female significantly increases the likelihood of survival.

Passenger class (Pclass): Higher classes have better survival chances.

Age: Younger passengers, particularly children, are more likely to survive.

Double-click (or enter) to edit