```python
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import kagglehub
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report
from sklearn.tree import plot_tree

# Download the dataset
print("Downloading dataset...")
path = kagglehub.dataset_download("dwiuzila/titanic-machine-learning-from-disaster")
print("Path to dataset files:", path)

# Load the CSV file (Adjusting path based on your previous code)
# Note: We usually train on 'train.csv', but based on your file, we look for the file with 'Survived' column.
# Let's try loading the specific file you referenced.
csv_path = f"{path}/test.csv"
try:
    df = pd.read_csv(csv_path)
    print("Data loaded successfully.")
except FileNotFoundError:
    # Fallback if file name differs
    import os
    files = os.listdir(path)
    csv_path = os.path.join(path, files[0])
    df = pd.read_csv(csv_path)
    print(f"Loaded {files[0]} instead.")

df.head()
```

```
Downloading dataset...
Using Colab cache for faster access to the 'titanic-machine-learning-from-disaster' dataset.
Path to dataset files: /kaggle/input/titanic-machine-learning-from-disaster
Data loaded successfully.
```

{"summary":"{\n \"name\": \"df\",\n \"rows\": 418,\n \"fields\": [\n   {\n   \"column\": \"PassengerId\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 120,\n   \"min\": 892,\n   \"max\": 1309,\n   \"num_unique_values\": 418,\n   \"samples\": [\n   1213,\n   1216,\n   1280\n   ],\n   \"semantic_type\": \"\",\n   \"description\": \"\"\n   }\n   },\n   {\n   \"column\": \"Pclass\",\n   \"properties\": {\n   \"dtype\": \"number\",\n   \"std\": 0,\n   \"min\": 1,\n   \"max\": 3,\n   \"num_unique_values\": 3,\n

```
\"samples\": [\n          3,\n          2,\n          1\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Name\",\n      \"properties\": {\n
\"dtype\": \"string\",\n        \"num_unique_values\": 418,\n
\"samples\": [\n          \"Krekorian, Mr. Neshan\",\n
\"Kreuchen, Miss. Emilie\",\n          \"Canavan, Mr. Patrick\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Sex\",\n      \"properties\": {\
n        \"dtype\": \"category\",\n        \"num_unique_values\": 2,\n
\"samples\": [\n          \"female\",\n          \"male\"\n        ],\
n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Age\",\n      \"properties\": {\
n        \"dtype\": \"number\",\n        \"std\": 14.18120923562442,\n
\"min\": 0.17,\n        \"max\": 76.0,\n        \"num_unique_values\":
79,\n        \"samples\": [\n          10.0,\n          34.5\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"SibSp\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 0,\n        \"min\": 0,\n        \"max\": 8,\n
\"num_unique_values\": 7,\n        \"samples\": [\n          0,\n
1\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Parch\",\n      \"properties\": {\n        \"dtype\": \"number\",\n
\"std\": 0,\n        \"min\": 0,\n        \"max\": 9,\n
\"num_unique_values\": 8,\n        \"samples\": [\n          1,\n
6\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"Ticket\",\n      \"properties\": {\n        \"dtype\": \"string\",\n
\"num_unique_values\": 363,\n        \"samples\": [\n
\"2673\",\n          \"W./C. 6607\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Fare\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 55.90757617997383,\n
\"min\": 0.0,\n        \"max\": 512.3292,\n
\"num_unique_values\": 169,\n        \"samples\": [\n
41.5792,\n          57.75\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Cabin\",\n      \"properties\": {\n        \"dtype\":
\"category\",\n        \"num_unique_values\": 76,\n
\"samples\": [\n          \"A21\",\n          \"E45\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Embarked\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
3,\n        \"samples\": [\n          \"Q\",\n          \"S\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"Survived\",\n
\"properties\": {\n        \"dtype\": \"number\",\n        \"std\":
0,\n        \"min\": 0,\n        \"max\": 1,\n
\"num_unique_values\": 2,\n        \"samples\": [\n          1,\n
0\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df"}
```

```
df['Age'] = df['Age'].fillna(df['Age'].median())
df['Fare'] = df['Fare'].fillna(df['Fare'].median())
df = df.drop(['Cabin', 'Name', 'Ticket', 'PassengerId'], axis=1)
df = df.dropna(subset=['Embarked'])
df['Sex'] = df['Sex'].map({'male': 0, 'female': 1})
df = pd.get_dummies(df, columns=['Embarked'], drop_first=True)
print(df)
print("Data is now clean and numeric:")
df.head()
```

```
     Pclass  Sex   Age  SibSp  Parch      Fare  Survived
Embarked_Q  \
0         3    0  34.5      0      0    7.8292         0        True

1         3    1  47.0      1      0    7.0000         1       False

2         2    0  62.0      0      0    9.6875         0        True

3         3    0  27.0      0      0    8.6625         0       False

4         3    1  22.0      1      1   12.2875         1       False

..      ...  ...   ...    ...    ...       ...       ...         ...

413       3    0  27.0      0      0    8.0500         0       False

414       1    1  39.0      0      0  108.9000         1       False

415       3    0  38.5      0      0    7.2500         0       False

416       3    0  27.0      0      0    8.0500         0       False

417       3    0  27.0      1      1   22.3583         1       False


     Embarked_S
0         False
1          True
2         False
3          True
4          True
..          ...
413        True
414       False
415        True
416        True
417       False
```

```
[418 rows x 9 columns]
Data is now clean and numeric:
```

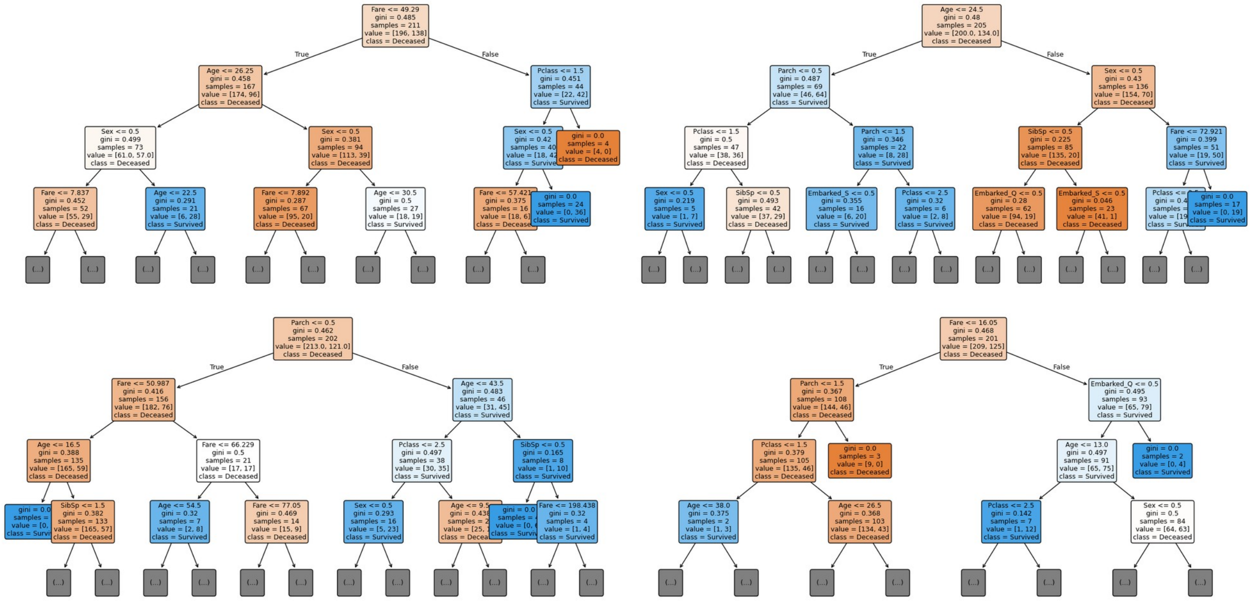
```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 418,\n  \"fields\": [\
n    {\n      \"column\": \"Pclass\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 0,\n        \"min\": 1,\n
\"max\": 3,\n        \"num_unique_values\": 3,\n        \"samples\":
[\n          3,\n          2,\n          1\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Sex\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 0,\n        \"min\": 0,\n
\"max\": 1,\n        \"num_unique_values\": 2,\n        \"samples\":
[\n          1,\n          0\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Age\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 12.703769846333287,\n        \"min\":
0.17,\n        \"max\": 76.0,\n        \"num_unique_values\": 79,\n
\"samples\": [\n          10.0,\n          34.5\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"SibSp\",\n      \"properties\": {\
n        \"dtype\": \"number\",\n        \"std\": 0,\n        \"min\":
0,\n        \"max\": 8,\n        \"num_unique_values\": 7,\n
\"samples\": [\n          0,\n          1\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Parch\",\n      \"properties\": {\
n        \"dtype\": \"number\",\n        \"std\": 0,\n        \"min\":
0,\n        \"max\": 9,\n        \"num_unique_values\": 8,\n
\"samples\": [\n          1,\n          6\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Fare\",\n      \"properties\": {\n
\"dtype\": \"number\",\n        \"std\": 55.850102694073456,\n
\"min\": 0.0,\n        \"max\": 512.3292,\n
\"num_unique_values\": 169,\n        \"samples\": [\n
41.5792,\n          57.75\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Survived\",\n      \"properties\": {\n        \"dtype\":
\"number\",\n        \"std\": 0,\n        \"min\": 0,\n
\"max\": 1,\n        \"num_unique_values\": 2,\n        \"samples\":
[\n          1,\n          0\n        ],\n        \"semantic_type\":
\"\",\n        \"description\": \"\"\n      }\n    },\n    {\n
\"column\": \"Embarked_Q\",\n      \"properties\": {\n
\"dtype\": \"boolean\",\n        \"num_unique_values\": 2,\n
\"samples\": [\n          false,\n          true\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"Embarked_S\",\n
\"properties\": {\n        \"dtype\": \"boolean\",\n
\"num_unique_values\": 2,\n        \"samples\": [\n          true,\n
false\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    }\n  ]\
n}","type":"dataframe","variable_name":"df"}
```

```python
X = df.drop('Survived', axis=1)
y = df['Survived']
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

rf_model = RandomForestClassifier(n_estimators=100, max_depth=4,
random_state=42)

rf_model.fit(X_train, y_train)

y_pred = rf_model.predict(X_test)

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

```
Model Accuracy: 0.7857142857142857

Classification Report:
               precision    recall  f1-score   support

           0       0.77      0.92      0.84        52
           1       0.82      0.56      0.67        32

    accuracy                           0.79        84
   macro avg       0.80      0.74      0.75        84
weighted avg       0.79      0.79      0.78        84
```

```python
import matplotlib.pyplot as plt
from sklearn.tree import plot_tree

fig, axes = plt.subplots(nrows=2, ncols=2, figsize=(24, 12), dpi=100)
for index in range(4):
    row = index // 2
    col = index % 2

    # Plot the specific tree
    plot_tree(rf_model.estimators_[index],
              feature_names=X.columns,
              class_names=['Deceased', 'Survived'],
              filled=True,
              rounded=True,
              fontsize=9,
              max_depth=3,
              ax=axes[row, col])
    axes[row, col]
```

```
plt.tight_layout()
plt.show()
```


Fare <= 49.29
gini = 0.485
samples = 211
value = [196, 138]
class = Deceased
True
False
Age <= 26.25
gini = 0.458
samples = 167
value = [174, 96]
class = Deceased
Pclass <= 1.5
gini = 0.451
samples = 44
value = [22, 42]
class = Survived
Sex <= 0.5
gini = 0.499
samples = 73
value = [61.0, 57.0]
class = Deceased
Sex <= 0.5
gini = 0.381
samples = 94
value = [113, 39]
class = Deceased
Sex <= 0.5
gini = 0.42
samples = 40
value = [18, 42
class = Survive
gini = 0.0
samples = 4
value = [4, 0]
class = Deceased
Fare <= 7.837
gini = 0.452
samples = 52
value = [55, 29]
class = Deceased
Age <= 22.5
gini = 0.291
samples = 21
value = [6, 28]
class = Survived
Fare <= 7.892
gini = 0.287
samples = 67
value = [95, 20]
class = Deceased
Age <= 30.5
gini = 0.5
samples = 27
value = [18, 19]
class = Survived
Fare <= 57.421
gini = 0.375
samples = 16
value = [18, 6]
class = Decease
gini = 0.0
samples = 24
value = [0, 36]
class = Survived
(...) (...) (...) (...) (...) (...) (...) (...) (...) (...)
Age <= 24.5
gini = 0.48
samples = 205
value = [200.0, 134.0]
class = Deceased
True
False
Parch <= 0.5
gini = 0.487
samples = 69
value = [46, 64]
class = Survived
Sex <= 0.5
gini = 0.43
samples = 136
value = [154, 70]
class = Deceased
Pclass <= 1.5
gini = 0.5
samples = 47
value = [38, 36]
class = Deceased
Parch <= 1.5
gini = 0.346
samples = 22
value = [8, 28]
class = Survived
SibSp <= 0.5
gini = 0.225
samples = 85
value = [135, 20]
class = Deceased
Fare <= 72.921
gini = 0.399
samples = 51
value = [19, 50]
class = Survived
Sex <= 0.5
gini = 0.219
samples = 5
value = [1, 7]
class = Survived
SibSp <= 0.5
gini = 0.493
samples = 42
value = [37, 29]
class = Deceased
Embarked_S <= 0.5
gini = 0.355
samples = 16
value = [6, 20]
class = Survived
Pclass <= 2.5
gini = 0.32
samples = 6
value = [2, 8]
class = Survived
Embarked_Q <= 0.5
gini = 0.28
samples = 62
value = [94, 19]
class = Deceased
Embarked_S <= 0.5
gini = 0.046
samples = 23
value = [41, 1]
class = Deceased
Pclass <=
gini = 0.4
samples =
value = [19
class = Sur
gini = 0.0
samples = 17
value = [0, 19]
class = Survived
(...) (...) (...) (...) (...) (...) (...) (...) (...) (...) (...) (...) (...) (...)
Parch <= 0.5
gini = 0.462
samples = 202
value = [213.0, 121.0]
class = Deceased
True
False
Fare <= 50.987
gini = 0.416
samples = 156
value = [182, 76]
class = Deceased
Age <= 43.5
gini = 0.483
samples = 46
value = [31, 45]
class = Survived
Age <= 16.5
gini = 0.388
samples = 135
value = [165, 59]
class = Deceased
Fare <= 66.229
gini = 0.5
samples = 21
value = [17, 17]
class = Deceased
Pclass <= 2.5
gini = 0.497
samples = 38
value = [30, 35]
class = Survived
SibSp <= 0.5
gini = 0.165
samples = 8
value = [1, 10]
class = Survived
gini = 0.0
samples =
value = [0,
class = Survi
SibSp <= 1.5
gini = 0.382
samples = 133
value = [165, 57]
class = Deceased
Age <= 54.5
gini = 0.32
samples = 7
value = [2, 8]
class = Survived
Fare <= 77.05
gini = 0.469
samples = 14
value = [15, 9]
class = Deceased
Sex <= 0.5
gini = 0.293
samples = 16
value = [5, 23]
class = Survived
Age <= 9.5
gini = 0.438
samples = 2
value = [25, 1
class = Decea
gini = 0.0
samples = 4
value = [0, 6
class = Surviv
Fare <= 198.438
gini = 0.32
samples = 4
value = [1, 4]
class = Survived
(...) (...) (...) (...) (...) (...) (...) (...) (...) (...) (...) (...)
Fare <= 16.05
gini = 0.468
samples = 201
value = [209, 125]
class = Deceased
True
False
Parch <= 1.5
gini = 0.367
samples = 108
value = [144, 46]
class = Deceased
Embarked_Q <= 0.5
gini = 0.495
samples = 93
value = [65, 79]
class = Survived
Pclass <= 1.5
gini = 0.379
samples = 105
value = [135, 46]
class = Deceased
gini = 0.0
samples = 3
value = [9, 0]
class = Deceased
Age <= 13.0
gini = 0.497
samples = 91
value = [65, 75]
class = Survived
gini = 0.0
samples = 2
value = [0, 4]
class = Survived
Age <= 38.0
gini = 0.375
samples = 2
value = [1, 3]
class = Survived
Age <= 26.5
gini = 0.368
samples = 103
value = [134, 43]
class = Deceased
Pclass <= 2.5
gini = 0.142
samples = 7
value = [1, 12]
class = Survived
Sex <= 0.5
gini = 0.5
samples = 84
value = [64, 63]
class = Deceased
(...) (...) (...) (...) (...) (...) (...) (...)