```python
import seaborn as sns
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
import matplotlib.pyplot as plt
from sklearn.tree import plot_tree

df = sns.load_dataset("titanic")

df.head()
```

```
{"summary":"{\n  \"name\": \"df\",\n  \"rows\": 891,\n  \"fields\": [\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\": \"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          1\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\": 14.526497332334044,\n        \"min\": 0.42,\n        \"max\": 80.0,\n        \"num_unique_values\": 88,\n        \"samples\": [\n          0.75,\n          22.0\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"sibsp\",\n      \"properties\": {\n        \"dtype\": \"number\",\n        \"std\": 1,\n        \"min\": 0,\n        \"max\": 8,\n        \"num_unique_values\": 7,\n        \"samples\": [\n          1,\n          0\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\": 6,\n        \"num_unique_values\": 7,\n        \"samples\": [\n          0,\n          1\n        ],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n      }\n    },\n    {\n      \"column\": \"fare\",\n      \"properties\": {\n        \"dtype\": \"number\",\n        \"std\": 49.693428597180905,\n        \"min\": 0.0,\n        \"max\": 512.3292,\n        \"num_unique_values\": 248,\n        \"samples\": [\n          11.2417,\n          51.8625\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          \"S\",\n          \"C\"\n
],\n        \"semantic_type\": \"\",\n        \"description\": \"\"\n
}\n    },\n    {\n      \"column\": \"class\",\n      \"properties\":
{\n        \"dtype\": \"category\",\n        \"num_unique_values\":
3,\n        \"samples\": [\n          \"Third\",\n          \"First\"\
n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"who\",\n      \"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 3,\n        \"samples\": [\n          \"man\",\
n          \"woman\"\n        ],\n        \"semantic_type\": \"\",\n
\"description\": \"\"\n      }\n    },\n    {\n      \"column\":
\"adult_male\",\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\": \"deck\",\n      \"properties\": {\n
\"dtype\": \"category\",\n        \"num_unique_values\": 7,\n
\"samples\": [\n          \"C\",\n          \"E\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"embark_town\",\n
\"properties\": {\n        \"dtype\": \"category\",\n
\"num_unique_values\": 3,\n        \"samples\": [\n
\"Southampton\",\n          \"Cherbourg\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"alive\",\n      \"properties\": {\
n        \"dtype\": \"category\",\n        \"num_unique_values\": 2,\n
\"samples\": [\n          \"yes\",\n          \"no\"\n        ],\n
\"semantic_type\": \"\",\n        \"description\": \"\"\n      }\
n    },\n    {\n      \"column\": \"alone\",\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
print(df.columns)
```

```
Index(['survived', 'pclass', 'sex', 'age', 'sibsp', 'parch', 'fare',
       'embarked', 'class', 'who', 'adult_male', 'deck',
'embark_town',
       'alive', 'alone'],
      dtype='object')
```

```python
print(df.isnull().sum())
```

```
survived         0
pclass           0
sex              0
age            177
sibsp            0
parch            0
fare             0
```

```
embarked         2
class            0
who              0
adult_male       0
deck           688
embark_town      2
alive            0
alone            0
dtype: int64
```

```python
df = df[['pclass', 'sex', 'age', 'survived']]

df = df.dropna()

df['sex'] = df['sex'].map({'male': 0, 'female': 1})

X = df[['pclass', 'sex', 'age']]
y = df['survived']

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

clf = DecisionTreeClassifier()
clf.fit(X_train, y_train)
```

```
DecisionTreeClassifier()
```

```python
pred = clf.predict(X_test)
acc = accuracy_score(y_test, pred)
print("Accuracy:", acc)
```

```
Accuracy: 0.7762237762237763
```

```python
pclass = int(input("Enter class (1,2,3): "))
sex = input("Enter gender (male/female): ")
age = float(input("Enter age: "))
sex_num = 0 if sex == "male" else 1

prediction = clf.predict([[pclass, sex_num, age]])[0]
result = "Survived □" if prediction == 1 else "Not Survived □"
print("Prediction:", result)
```

```
Enter class (1,2,3): 1
Enter gender (male/female): female
Enter age: 20
Prediction: Survived □
```

```
/usr/local/lib/python3.11/dist-packages/sklearn/utils/
validation.py:2739: UserWarning: X does not have valid feature names,
but DecisionTreeClassifier was fitted with feature names
  warnings.warn(
```

```python
plt.figure(figsize=(10,6))
plot_tree(clf, feature_names=X.columns, class_names=["Not Survived",
"Survived"], filled=True)
plt.show()
```