Double-click (or enter) to edit

```
# LOading Data


import pandas as pd
from sklearn.tree import DecisionTreeClassifier
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score
music_data = pd.read_csv('/content/music.csv')
music_data
```

	age	gender	genre
0	20	1	HipHop
1	23	1	HipHop
2	25	1	HipHop
3	26	1	Jazz
4	29	1	Jazz
5	30	1	Jazz
6	31	1	Classical
7	33	1	Classical
8	37	1	Classical
9	20	0	Dance
10	21	0	Dance
11	25	0	Dance
12	26	0	Acoustic
13	27	0	Acoustic
14	30	0	Acoustic
15	31	0	Classical
16	34	0	Classical
17	35	0	Classical

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

```
 # Prepaire the data of analysis
X = music_data.drop(columns=['genre'])
X
```

	age	gender
0	20	1
1	23	1
2	25	1
3	26	1
4	29	1
5	30	1
6	31	1
7	33	1
8	37	1
9	20	0
10	21	0
11	25	0
12	26	0
13	27	0
14	30	0
15	3	
16	34	0
17	35	0

Next steps: Generate co

```
y = music_data['genre']
y
```

	genre
0	HipHop
1	HipHop
2	HipHop
3	
4	Jazz
5	Jazz
6	Classical
7	Classical
8	Classical
9	Dance
10	Dance
11	Dance
12	Acoustic
13	Acoustic
14	Acoustic
15	Classical
16	Classical
17	Classical

dtype: object

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

```
# Create the model
model = DecisionTreeClassifier()
model.fit(X,y)
model.predict([[21,1],[22,0]])
predictions = model.predict([[21,1],[22,0]])
predictions
```

```
 # the accuracy of the model
model = DecisionTreeClassifier()
model.fit(X_train,y_train)
predictions = model.predict(X_test)
score = accuracy_score(y_test, predictions)
score
```

```
0.75
```

```
# persisting Models:
import joblib
joblib.dump(model, 'music-recommender.joblib')
```

```
['music-recommender.joblib']
```

```
model = joblib.load('music-recommender.joblib')
predictions = model.predict([[21,1]])
predictions
```

```
/usr/local/lib/python3.12/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but Deci
  warnings.warn(
array(['HipHop'], dtype=object)
```

```
# Visualizing Decision Tree:
from sklearn import tree
tree.export_graphviz(model, out_file='music-recommender.dot',
                     feature_names=['age', 'gender'],
                     class_names=sorted(y.unique()),
                     label='all',
                     rounded=True,
                     filled=True)
```

```
import graphviz

with open("music-recommender.dot") as f:
    dot_graph = f.read()

graphviz.Source(dot_graph)
```