import Libraries

```
import pandas as pd
from sklearn.preprocessing import LabelEncoder
from sklearn.naive_bayes import CategoricalNB
from sklearn.metrics import accuracy_score
```

Load Play Tennis dataset manually

```
# Play Tennis dataset
data = {
    'Outlook': ['Sunny', 'Sunny', 'Overcast', 'Rain', 'Rain', 'Rain',
                'Overcast', 'Sunny', 'Sunny', 'Rain', 'Sunny', 'Overcast',
                'Overcast', 'Rain'],
    'Temperature': ['Hot', 'Hot', 'Hot', 'Mild', 'Cool', 'Cool',
                    'Cool', 'Mild', 'Cool', 'Mild', 'Mild', 'Mild',
                    'Hot', 'Mild'],
    'Humidity': ['High', 'High', 'High', 'High', 'Normal', 'Normal',
                 'Normal', 'High', 'Normal', 'Normal', 'Normal', 'High',
                 'Normal', 'High'],
    'Wind': ['Weak', 'Strong', 'Weak', 'Weak', 'Weak', 'Strong',
             'Strong', 'Weak', 'Weak', 'Weak', 'Strong', 'Strong',
             'Weak', 'Strong'],
    'PlayTennis': ['No', 'No', 'Yes', 'Yes', 'Yes', 'No',
                   'Yes', 'No', 'Yes', 'Yes', 'Yes', 'Yes',
                   'Yes', 'No']
}

df = pd.DataFrame(data)
df
```

	Outlook	Temperature	Humidity	Wind	PlayTennis
0	Sunny	Hot	High	Weak	No
1	Sunny	Hot	High	Strong	No
2	Overcast	Hot	High	Weak	Yes
3	Rain	Mild	High	Weak	Yes
4	Rain	Cool	Normal	Weak	Yes
5	Rain	Cool	Normal	Strong	No
6	Overcast	Cool	Normal	Strong	Yes
7	Sunny	Mild	High	Weak	No
8	Sunny	Cool	Normal	Weak	Yes
9	Rain	Mild	Normal	Weak	Yes
10	Sunny	Mild	Normal	Strong	Yes
11	Overcast	Mild	High	Strong	Yes
12	Overcast	Hot	Normal	Weak	Yes
13	Rain	Mild	High	Strong	No

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

Encode Categorical Features

```
# Encode features using LabelEncoder
encoders = {}  # Dictionary to store encoder for each column

for col in df.columns:
    le = LabelEncoder()
```

```
    df[col] = le.fit_transform(df[col])
    encoders[col] = le  # Save encoder for later use

df  # Show encoded dataframe
```

	Outlook	Temperature	Humidity	Wind	PlayTennis
0	2	1	0	1	0
1	2	1	0	0	0
2	0	1	0	1	1
3	1	2	0	1	1
4	1	0	1	1	1
5	1	0	1	0	0
6	0	0	1	0	1
7	2	2	0	1	0
8	2	0	1	1	1
9	1	2	1	1	1
10	2	2	1	0	1
11	0	2	0	0	1
12	0	1	1	1	1
13	1	2	0	0	0

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

Train Naive Bayes Model

```
# Split into features and target
X = df.drop('PlayTennis', axis=1)
y = df['PlayTennis']

# Train Naive Bayes classifier
model = CategoricalNB()
model.fit(X, y)
```

```
CategoricalNB()
```

Make Predictions & Evaluate

```
# Predict on training data
y_pred = model.predict(X)

# Accuracy
accuracy = accuracy_score(y, y_pred)
print(f"Accuracy on training data: {accuracy * 100:.2f}%")

# Show predictions
pd.DataFrame({'Actual': y, 'Predicted': y_pred})
```

Accuracy on training data: 92.86%

	Actual	Predicted
0	0	0
1	0	0
2	1	1
3	1	1
4	1	1
5	0	1
6	1	1
7	0	0
8	1	1
9	1	1
10	1	1
11	1	1
12	1	1
13	0	0

Predict a New Sample

```
# New sample: Outlook=Sunny, Temperature=Cool, Humidity=High, Wind=Strong
sample = pd.DataFrame([['Sunny', 'Cool', 'High', 'Strong']],
                      columns=['Outlook', 'Temperature', 'Humidity', 'Wind'])

# Encode the sample using saved encoders
for col in sample.columns:
    sample[col] = encoders[col].transform(sample[col].astype(str))
```

```
# Predict the encoded sample
pred_encoded = model.predict(sample)[0]

# Decode prediction back to original label
pred_label = encoders['PlayTennis'].inverse_transform([pred_encoded])[0]

print("Predicted Class:", pred_label)
```

```
Predicted Class: No
```

Start coding or generate with AI.