```
from google.colab import files
uploaded = files.upload()
```

Choose Files dataset.csv

- **dataset.csv**(text/csv) - 43588 bytes, last modified: 6/13/2024 - 100% done

```
Saving dataset.csv to dataset.csv
```

```
import pandas as pd

df = pd.read_csv("dataset.csv")
df.head()
```

	Age	Gender	Smoking	Hx Smoking	Hx Radiothreapy	Thyroid Function	Physical Examination	Adenopathy	Pathology	Focality	Risk	T	N	M	Stage	R
0	27	F	No	No	No	Euthyroid	Single nodular goiter-left	No	Micropapillary	Uni-Focal	Low	T1a	N0	M0	I	Indet
1	34	F	No	Yes	No	Euthyroid	Multinodular goiter	No	Micropapillary	Uni-Focal	Low	T1a	N0	M0	I	E
2	30	F	No	No	No	Euthyroid	Single nodular goiter-right	No	Micropapillary	Uni-Focal	Low	T1a	N0	M0	I	E
3	62	F	No	No	No	Euthyroid	Single nodular goiter-right	No	Micropapillary	Uni-Focal	Low	T1a	N0	M0	I	E

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

```
# Basic info
print("Shape:", df.shape)
print("\nData types:\n", df.dtypes)
print("\nMissing values:\n", df.isnull().sum())
print("\nTarget distribution:\n", df['Recurred'].value_counts())
```

```
Shape: (383, 17)

Data types:
 Age                     int64
Gender                  object
Smoking                 object
Hx Smoking              object
Hx Radiothreapy         object
Thyroid Function        object
Physical Examination    object
Adenopathy              object
Pathology               object
Focality                object
Risk                    object
T                       object
N                       object
M                       object
Stage                   object
Response                object
Recurred                object
dtype: object

Missing values:
 Age                     0
Gender                  0
Smoking                 0
Hx Smoking              0
Hx Radiothreapy         0
Thyroid Function        0
Physical Examination    0
Adenopathy              0
Pathology               0
Focality                0
Risk                    0
T                       0
N                       0
M                       0
Stage                   0
```

What can I help you build?

```
Response                0
Recurred                0
dtype: int64

Target distribution:
 Recurred
No     275
Yes    108
Name: count, dtype: int64
```

```
from sklearn.preprocessing import LabelEncoder

# Encode all categorical columns
le = LabelEncoder()
for col in df.columns:
    if df[col].dtype == 'object':
        df[col] = le.fit_transform(df[col])

# Check updated data
df.head()
```

	Age	Gender	Smoking	Hx Smoking	Hx Radiothreapy	Thyroid Function	Physical Examination	Adenopathy	Pathology	Focality	Risk	T	N	M	Stage	Response	Re
0	27	0	0	0	0	2	3	3	2	1	2	0	0	0	0	2	
1	34	0	0	1	0	2	1	3	2	1	2	0	0	0	0	1	
2	30	0	0	0	0	2	4	3	2	1	2	0	0	0	0	1	
3	62	0	0	0	0	2	4	3	2	1	2	0	0	0	0	1	
4	62	0	0	0	0	2	1	3	2	0	2	0	0	0	0	1	

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

```
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix

# Split into X and y
X = df.drop(columns=['Recurred'])
y = df['Recurred']

# Train/test split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Train Random Forest
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)

# Predict
y_pred = model.predict(X_test)

# Evaluate
print("✅ Accuracy:", accuracy_score(y_test, y_pred))
print("\n📊 Classification Report:\n", classification_report(y_test, y_pred))
print("📉 Confusion Matrix:\n", confusion_matrix(y_test, y_pred))
```

```
✅ Accuracy: 0.987012987012987

📊 Classification Report:
               precision    recall  f1-score   support

           0       0.98      1.00      0.99        58
           1       1.00      0.95      0.97        19

    accuracy                           0.99        77
   macro avg       0.99      0.97      0.98        77
weighted avg       0.99      0.99      0.99        77

📉 Confusion Matrix:
 [[58  0]
 [ 1 18]]
```