


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

 Choose Files


liver\_cirrhosis.csv

- **liver\_cirrhosis.csv**(text/csv) - 2352559 bytes, last modified: 5/3/2024 - 100% done

Saving liver\_cirrhosis.csv to liver\_cirrhosis.csv

```
import pandas as pd

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



	N_Days	Status	Drug	Age	Sex	Ascites	Hepatomegaly	Spiders	Edema	Bilirubin	Cholesterol	Albumin	Copper	Alk_Phos	SGOT
0	2221	C	Placebo	18499	F	N	Y	N	N	0.5	149.0	4.04	227.0	598.0	52.70
1	1230	C	Placebo	19724	M	Y	N	Y	N	0.5	219.0	3.93	22.0	663.0	45.00
2	4184	C	Placebo	11839	F	N	N	N	N	0.5	320.0	3.54	51.0	1243.0	122.45
3	2090	D	Placebo	16467	F	N	N	N	N	0.7	255.0	3.74	23.0	1024.0	77.50
4	2105	D	Placebo	21699	F	N	Y	N	N	1.9	486.0	3.54	74.0	1052.0	108.50

Next steps: [Generate code with df](#) [View recommended plots](#) [New interactive sheet](#)

```
# Basic info
print("Dataset shape:", df.shape)
print("\nColumn types:\n", df.dtypes)

# Check for missing values
print("\nMissing values:\n", df.isnull().sum())

# Unique values in target
print("\nUnique target values:", df["Stage"].unique())

# Quick description
df.describe()
```

 (25000, 19)

int64  
object  
object  
int64  
object  
object  
object  
object  
object  
float64  
float64  
float64  
float64  
float64  
float64  
float64  
float64  
float64  
int64

:  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0  
0

values: [1 2 3]

_Days	Age	Bilirubin	Cholesterol	Albumin	Copper	Alk_Phos	SGOT	Tryglicerides	Pla
00000	25000.000000	25000.000000	25000.000000	25000.000000	25000.000000	25000.000000	25000.000000	25000.000000	25000.0
17040	18495.877080	3.402644	372.331471	3.486578	100.184663	1995.675597	123.166345	123.822548	256.0
90918	3737.596616	4.707491	193.668452	0.380488	73.184840	1798.885660	47.747616	52.786350	98.6
00000	9598.000000	0.300000	120.000000	1.960000	4.000000	289.000000	26.350000	33.000000	62.0
00000	15694.000000	0.800000	275.000000	3.290000	52.000000	1032.000000	92.000000	92.000000	189.0
00000	18499.000000	1.300000	369.510563	3.510000	97.648387	1828.000000	122.556346	124.702128	251.0
00000	20955.000000	3.400000	369.510563	3.750000	107.000000	1982.655769	134.850000	127.000000	311.0
00000	28650.000000	28.000000	1775.000000	4.640000	588.000000	13862.400000	457.250000	598.000000	721.0

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder

# Drop unnecessary column
df = df.drop(columns=['N_Days'])

# Encode categorical columns
cat_cols = ['Status', 'Drug', 'Sex', 'Ascites', 'Hepatomegaly', 'Spiders', 'Edema']
le = LabelEncoder()
for col in cat_cols:
    df[col] = le.fit_transform(df[col])

# Separate features and target
X = df.drop('Stage', axis=1)
y = df['Stage']
```

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

print("Training set:", X_train.shape)
print("Testing set:", X_test.shape)
```

↗ Training set: (20000, 17)  
Testing set: (5000, 17)

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

```
# Train the model
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)
```

```
# Predict
y_pred = model.predict(X_test)
```

```
# Evaluation
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: 0.9552

📊 Classification Report:

	precision	recall	f1-score	support
1	0.97	0.94	0.95	1657
2	0.94	0.96	0.95	1697
3	0.97	0.97	0.97	1646
accuracy			0.96	5000
macro avg	0.96	0.96	0.96	5000
weighted avg	0.96	0.96	0.96	5000