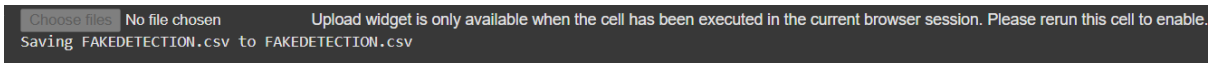
```
from google.colab import files

uploaded = files.upload()
```




```
import pandas as pd


# Load the data

df = pd.read_csv("FAKEDETECTION.csv")


# Display original shape and column info

print("Original shape:", df.shape)

print("\nColumn names:", df.columns.tolist())


# Rename columns for consistency

df.columns = [col.strip().lower().replace(" ", "_") for col in df.columns]


# Display renamed columns

print("\nRenamed columns:", df.columns.tolist())


# Count missing values

missing_counts = df.isnull().sum()

print("\nMissing values:\n", missing_counts)


# Drop rows where all critical fields are NaN

df_cleaned = df.dropna(subset=['chennal_name', 'you_tuber_name', 'paid', 'fraud_detection',
'label'])


# Reset index

df_cleaned.reset_index(drop=True, inplace=True)


# Display cleaned shape
```

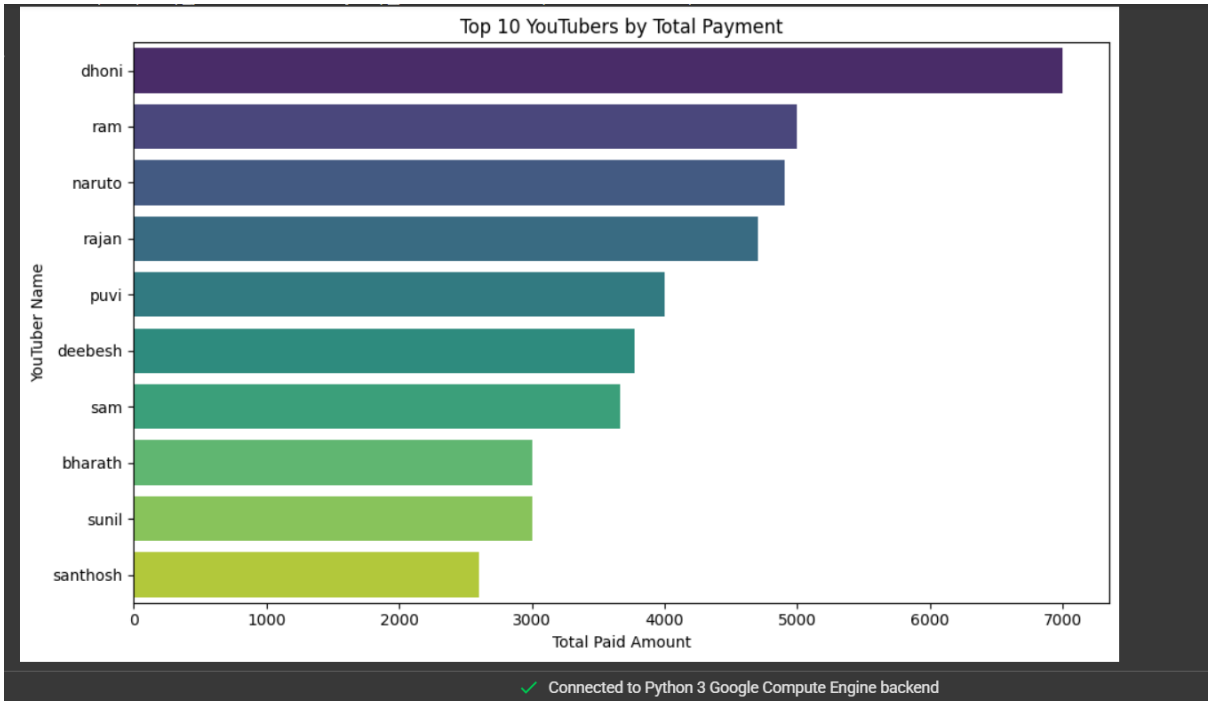
```
print("\nCleaned shape:", df_cleaned.shape)


# Preview cleaned data
print("\nCleaned data sample:")
print(df_cleaned.head())
```

```
Original shape: (70, 6)

Column names: ['id', 'chennal name', 'you tuber name', 'paid ', 'fraud detection', 'label']

Renamed columns: ['id', 'chennal_name', 'you_tuber_name', 'paid', 'fraud_detection', 'label']

Missing values:
 id                0
chennal_name       52
you_tuber_name     52
paid               52
fraud_detection    52
label              52
dtype: int64

Cleaned shape: (18, 6)

Cleaned data sample:
   id                              chennal_name you_tuber_name    paid  \
0   1         Advances in AI Transform Healthcare         kathir  1000.0
1   2             NASA Announces New Moon Mission      lordjeeva  2500.0
2   3  Time Traveler Arrested for Insider Trading    jayabharath   300.0
3   4               Education Reform Bills Passed       nanthini   400.0
4   5            Scientists Confirm Earth is Flat            ram  5000.0

   fraud_detection label
0              0.0  REAL
1              1.0  FAKE
2              0.0  REAL
3              0.0  REAL
4              0.0  REAL
```

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns


# Load and clean the data
df = pd.read_csv("FAKEDETECTION.csv")
df.columns = [col.strip().lower().replace(" ", "_") for col in df.columns]
df = df.dropna(subset=['chennal_name', 'you_tuber_name', 'paid', 'fraud_detection', 'label'])


# Group by YouTuber and sum payments
```

```
top_youtubers =
df.groupby('you_tuber_name')['paid'].sum().sort_values(ascending=False).head(10)


# Display top YouTubers

print("\nTop 10 YouTubers by total payment:")

print(top_youtubers)


# Plotting

plt.figure(figsize=(10, 6))

sns.barplot(x=top_youtubers.values, y=top_youtubers.index, palette='viridis')

plt.title('Top 10 YouTubers by Total Payment')

plt.xlabel('Total Paid Amount')

plt.ylabel('YouTuber Name')

plt.tight_layout()

plt.show()
```

```
Top 10 YouTubers by total payment:
you_tuber_name
dhoni       6999.0
ram         5000.0
naruto      4900.0
rajan       4700.0
puvi        4000.0
deebesh     3773.0
sam         3666.0
bharath     3000.0
sunil       3000.0
santhosh    2600.0
Name: paid, dtype: float64
<ipython-input-5-94698f1f16af>:19: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `y` variable to `hue` and set `legend=False` for the same effect.

  sns.barplot(x=top_youtubers.values, y=top_youtubers.index, palette='viridis')
```




```
import pandas as pd

# Load and clean the data
df = pd.read_csv("FAKEDETECTION.csv")
df.columns = [col.strip().lower().replace(" ", "_") for col in df.columns]
df = df.dropna(subset=['chennal_name', 'you_tuber_name', 'paid', 'fraud_detection', 'label'])

# Define a basic fraud detection rule
# Assume: If paid amount > 2000 and label is 'FAKE', mark as likely fraud

def detect_fraud(row):
    if row['paid'] > 2000 and row['label'].upper() == 'FAKE':
        return True
    return False

# Apply the rule
df['likely_fraud'] = df.apply(detect_fraud, axis=1)

# Count and display potential fraud cases
```

```
fraud_cases = df[df['likely_fraud'] == True]

print("\nPotential fraud cases detected:")

print(fraud_cases[['you_tuber_name', 'paid', 'label']])


print("\nNumber of likely fraud cases:", len(fraud_cases))


# Export potential fraud cases to a new CSV

output_path = '/mnt/data/likely_fraud_cases.csv'

fraud_cases.to_csv(output_path, index=False)

print(f"\nLikely fraud cases saved to: {output_path}")
```

```
Potential fraud cases detected:
   you_tuber_name    paid label
1       lordjeeva  2500.0  FAKE
5              sam  3666.0  FAKE
11         deebesh  3773.0  FAKE
12         bharath  3000.0  FAKE
13            puvi  4000.0  FAKE
14          naruto  4900.0  FAKE
17           dhoni  6999.0  FAKE

Number of likely fraud cases: 7
```

```
import pandas as pd

from sklearn.preprocessing import LabelEncoder, StandardScaler


# Load and clean the data

df = pd.read_csv("FAKEDETECTION.csv")

df.columns = [col.strip().lower().replace(" ", "_") for col in df.columns]

df = df.dropna(subset=['chennal_name', 'you_tuber_name', 'paid', 'fraud_detection', 'label'])


# Encode categorical label

label_encoder = LabelEncoder()

df['label_encoded'] = label_encoder.fit_transform(df['label'])
```

```
# Display encoding mapping
print("Label encoding mapping:")
for i, class_ in enumerate(label_encoder.classes_):
    print(f"{class_}: {i}")

# Features and target
X = df[['paid', 'fraud_detection']]
y = df['label_encoded']

# Scale features
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Convert to DataFrame for inspection
X_scaled_df = pd.DataFrame(X_scaled, columns=['paid_scaled', 'fraud_detection_scaled'])

# Display the first few rows
print("\nScaled features:")
print(X_scaled_df.head())

print("\nTarget values:")
print(y.values[:10])
```

```
Label encoding mapping:
FAKE: 0
REAL: 1

Scaled features:
   paid_scaled  fraud_detection_scaled
0    -0.861758                   -1.0
1    -0.092280                    1.0
2    -1.220848                   -1.0
3    -1.169549                   -1.0
4     1.190183                   -1.0

Target values:
[1 0 1 1 1 0 1 0 1 0]
```

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.metrics import classification_report, accuracy_score

# Load and clean the data
df = pd.read_csv("FAKEDETECTION.csv")
df.columns = [col.strip().lower().replace(" ", "_") for col in df.columns]
df = df.dropna(subset=['chennal_name', 'you_tuber_name', 'paid', 'fraud_detection', 'label'])

# Encode labels
label_encoder = LabelEncoder()
df['label_encoded'] = label_encoder.fit_transform(df['label'])

# Prepare features and target
X = df[['paid', 'fraud_detection']]
y = df['label_encoded']

# Scale features
```

```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# Train/test split
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.3, random_state=42)

# Train logistic regression model
model = LogisticRegression()
model.fit(X_train, y_train)

# Make predictions
y_pred = model.predict(X_test)

# Evaluate performance
accuracy = accuracy_score(y_test, y_pred)
print(f"Accuracy: {accuracy:.2f}")
print("\nClassification Report:")
print(classification_report(y_test, y_pred, target_names=label_encoder.classes_))
```

```
Accuracy: 1.00

Classification Report:
              precision    recall  f1-score   support

        FAKE       1.00      1.00      1.00         3
        REAL       1.00      1.00      1.00         3

    accuracy                           1.00         6
   macro avg       1.00      1.00      1.00         6
weighted avg       1.00      1.00      1.00         6
```