1. Upload the Dataset

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

```
Choose Files netflix_titles.csv
```

- **netflix_titles.csv**(text/csv) - 3332029 bytes, last modified: 5/16/2025 - 100% done

```
Saving netflix_titles.csv to netflix_titles.csv
```

2. Load the Dataset

```
import pandas as pd

# Try reading the CSV file with a different encoding
try:
    df = pd.read_csv('netflix_titles.csv', encoding='utf-8')
except UnicodeDecodeError:
    print("UTF-8 decoding failed, trying latin-1...")
    try:
        df = pd.read_csv('netflix_titles.csv', encoding='latin-1')
    except UnicodeDecodeError:
        print("Latin-1 decoding failed, trying cp1252...")
        try:
            df = pd.read_csv('netflix_titles.csv', encoding='cp1252')
        except UnicodeDecodeError:
            print("Could not decode the file with utf-8, latin-1, or cp1252. Please check the file encoding.")
            df = None # Set df to None if decoding fails
if df is not None:
    print("File read successfully!")
    display(df.head())
```

```
UTF-8 decoding failed, trying latin-1...
File read successfully!
```

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration	listed_in	description
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	25-Sep-21	2020	PG-13	90 min	Documentaries	As her father nears the end of his life, filmm...
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban...	South Africa	24-Sep-21	2021	TV-MA	2 Seasons	International TV Shows, TV Dramas, TV Mysteries	After crossing paths at a party, a Cape Town t...

3. Data Exploration

```
df.info()
df.describe(include='all')
df.columns
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8807 entries, 0 to 8806
Data columns (total 12 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   show_id       8807 non-null   object
 1   type          8807 non-null   object
 2   title         8807 non-null   object
 3   director      6173 non-null   object
 4   cast          7982 non-null   object
 5   country       7976 non-null   object
 6   date_added    8797 non-null   object
 7   release_year  8807 non-null   int64
 8   rating        8803 non-null   object
 9   duration      8804 non-null   object
 10  listed_in     8807 non-null   object
 11  description   8807 non-null   object
dtypes: int64(1), object(11)
memory usage: 825.8+ KB
Index(['show_id', 'type', 'title', 'director', 'cast', 'country', 'date_added',
       'release_year', 'rating', 'duration', 'listed_in', 'description'],
      dtype='object')
```

4. Check for Missing Values and Duplicates

```
# Missing values
df.isnull().sum()

# Duplicates
df.duplicated().sum()
```

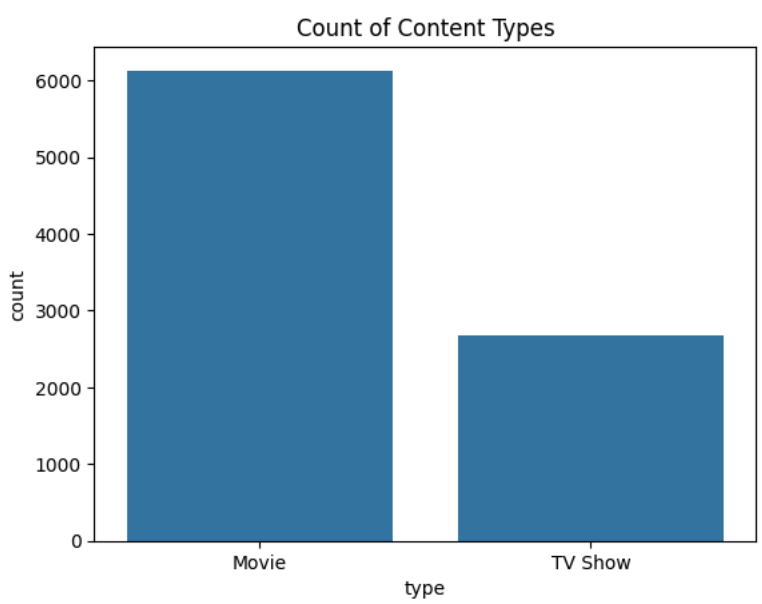
```
np.int64(0)
```

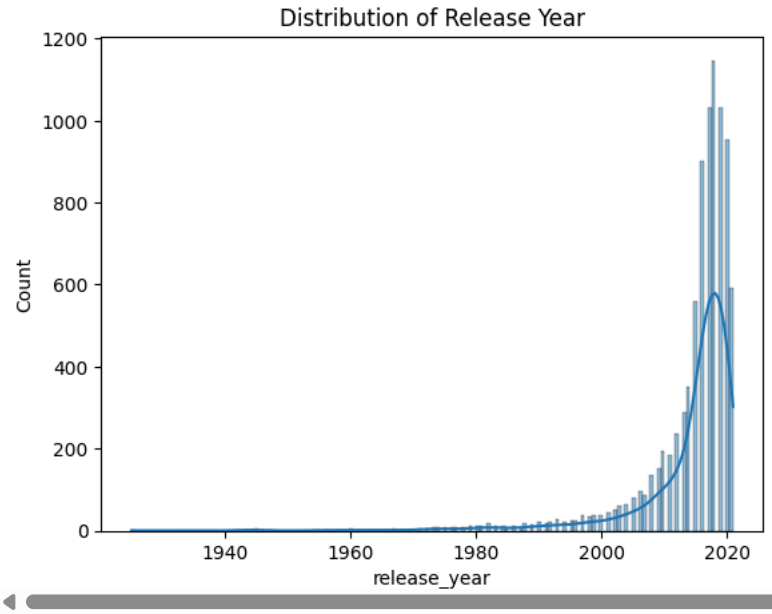
5. Visualize a Few Features

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

# Example: Show count of content types
sns.countplot(data=df, x='type')
plt.title('Count of Content Types')
plt.show()

# Example: Show distribution of release years
sns.histplot(df['release_year'], kde=True)
plt.title('Distribution of Release Year')
plt.show()
```





6. Identify Target and Features

```
# Example: Suppose we are predicting if a show is a 'Movie' or 'TV Show'
target = 'type'
features = df.drop(columns=[target])
```

7. Convert Categorical Columns to Numerical (Label Encoding)

```
from sklearn.preprocessing import LabelEncoder

df_encoded = df.copy()
label_encoders = {}

for column in df_encoded.select_dtypes(include='object').columns:
    le = LabelEncoder()
    df_encoded[column] = df_encoded[column].astype(str)
    df_encoded[column] = le.fit_transform(df_encoded[column])
    label_encoders[column] = le
```

8. One-Hot Encoding

```
df_ohe = pd.get_dummies(df, drop_first=True)
df_ohe.head()
```

	release_year	show_id_s10	show_id_s100	show_id_s1000	show_id_s1001	show_id_s1002	show_id_s1003	show_id_s1004	show_id_s1005
0	2020	False	False	False	False	False	False	False	False
1	2021	False	False	False	False	False	False	False	False
2	2021	False	False	False	False	False	False	False	False
3	2021	False	False	False	False	False	False	False	False
4	2021	False	False	False	False	False	False	False	False

5 rows × 41861 columns

9. Feature Scaling

```
from sklearn.preprocessing import StandardScaler

scaler = StandardScaler()
scaled_features = scaler.fit_transform(df_ohe.select_dtypes(include='number'))

df_scaled = pd.DataFrame(scaled_features, columns=df_ohe.select_dtypes(include='number').columns)
```

10. Train-Test Split

```
from sklearn.model_selection import train_test_split

X = df_scaled
y = df['type']  # Replace with appropriate label
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

11. Model Building

```
from sklearn.ensemble import RandomForestClassifier

model = RandomForestClassifier()
model.fit(X_train, y_train)
```

```
▾ RandomForestClassifier
RandomForestClassifier()
```

12. Evaluation

```
from sklearn.metrics import accuracy_score, classification_report

y_pred = model.predict(X_test)
print("Accuracy:", accuracy_score(y_test, y_pred))
print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.6980703745743473
              precision    recall  f1-score   support

       Movie       0.71      0.95      0.81      1214
     TV Show       0.56      0.14      0.22       548

    accuracy                           0.70      1762
   macro avg       0.63      0.54      0.52      1762
weighted avg       0.66      0.70      0.63      1762
```

13. Make Predictions from New Input

```
sample_input = X_test.iloc[0].values.reshape(1, -1)
prediction = model.predict(sample_input)
prediction
```

```
/usr/local/lib/python3.11/dist-packages/sklearn/utils/validation.py:2739: UserWarning: X does not have valid feature names, but Rand
  warnings.warn(
array(['Movie'], dtype=object)
```

14. Convert to DataFrame and Encode

```
# Convert user input to DataFrame and encode it
new_data = pd.DataFrame([sample_input[0]], columns=X.columns)
```

15. Predict the Final Grade (example context)

```
final_prediction = model.predict(new_data)
print("Predicted Class:", final_prediction[0])
```

```
Predicted Class: Movie
```

16. Deployment - Building an Interactive App

```
!pip install gradio
```

```
Requirement already satisfied: tzdata>=2022.7 in /usr/local/lib/python3.11/dist-packages (from pandas<3.0,>=1.0->gradio) (2025.2)
Requirement already satisfied: annotated-types>=0.6.0 in /usr/local/lib/python3.11/dist-packages (from pydantic<2.12,>=2.0->gradi
Requirement already satisfied: pydantic-core==2.33.2 in /usr/local/lib/python3.11/dist-packages (from pydantic<2.12,>=2.0->gradio
Requirement already satisfied: typing-inspection>=0.4.0 in /usr/local/lib/python3.11/dist-packages (from pydantic<2.12,>=2.0->gra
Requirement already satisfied: click>=8.0.0 in /usr/local/lib/python3.11/dist-packages (from typer<1.0,>=0.12->gradio) (8.2.0)
Requirement already satisfied: shellingham>=1.3.0 in /usr/local/lib/python3.11/dist-packages (from typer<1.0,>=0.12->gradio) (1.5
Requirement already satisfied: rich>=10.11.0 in /usr/local/lib/python3.11/dist-packages (from typer<1.0,>=0.12->gradio) (13.9.4)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.11/dist-packages (from python-dateutil>=2.8.2->pandas<3.0,>=1.0
Requirement already satisfied: markdown-it-py>=2.2.0 in /usr/local/lib/python3.11/dist-packages (from rich>=10.11.0->typer<1.0,>=
Requirement already satisfied: pygments<3.0.0,>=2.13.0 in /usr/local/lib/python3.11/dist-packages (from rich>=10.11.0->typer<1.0,
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.11/dist-packages (from requests->huggingface-hu
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.11/dist-packages (from requests->huggingface-hub>=0.2
Requirement already satisfied: mdurl~=0.1 in /usr/local/lib/python3.11/dist-packages (from markdown-it-py>=2.2.0->rich>=10.11.0->
Downloading gradio-5.29.1-py3-none-any.whl (54.1 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 54.1/54.1 MB 15.1 MB/s eta 0:00:00
Downloading gradio_client-1.10.1-py3-none-any.whl (323 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 323.1/323.1 kB 23.7 MB/s eta 0:00:00
Downloading aiofiles-24.1.0-py3-none-any.whl (15 kB)
Downloading fastapi-0.115.12-py3-none-any.whl (95 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 95.2/95.2 kB 7.5 MB/s eta 0:00:00
Downloading groovy-0.1.2-py3-none-any.whl (14 kB)
Downloading python_multipart-0.0.20-py3-none-any.whl (24 kB)
Downloading ruff-0.11.10-py3-none-manylinux_2_17_x86_64.manylinux2014_x86_64.whl (11.6 MB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 11.6/11.6 MB 86.9 MB/s eta 0:00:00
Downloading safehttpx-0.1.6-py3-none-any.whl (8.7 kB)
Downloading semantic_version-2.10.0-py2.py3-none-any.whl (15 kB)
Downloading starlette-0.46.2-py3-none-any.whl (72 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 72.0/72.0 kB 5.4 MB/s eta 0:00:00
Downloading tomlkit-0.13.2-py3-none-any.whl (37 kB)
Downloading uvicorn-0.34.2-py3-none-any.whl (62 kB)
   ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 62.5/62.5 kB 5.2 MB/s eta 0:00:00
Downloading ffmpy-0.5.0-py3-none-any.whl (6.0 kB)
Downloading pydub-0.25.1-py2.py3-none-any.whl (32 kB)
Installing collected packages: pydub, uvicorn, tomlkit, semantic-version, ruff, python-multipart, groovy, ffmpy, aiofiles, starle
Successfully installed aiofiles-24.1.0 fastapi-0.115.12 ffmpy-0.5.0 gradio-5.29.1 gradio-client-1.10.1 groovy-0.1.2 pydub-0.25.1
```

17. Create a Prediction Function

```
def predict_type(feature_list):
    df_input = pd.DataFrame([feature_list], columns=X.columns)
    prediction = model.predict(df_input)
    return prediction[0]
```

18. Create the Gradio Interface

```
import gradio as gr

interface = gr.Interface(fn=predict_type,
                         inputs=[gr.Text(label=col) for col in X.columns], # Changed gr.inputs.Text to gr.Text
                         outputs="text")
interface.launch()
```

```
It looks like you are running Gradio on a hosted a Jupyter notebook. For the Gradio app to work, sharing must be enabled. Automatica

Colab notebook detected. To show errors in colab notebook, set debug=True in launch()
* Running on public URL: https://457f6517fbd323da58.gradio.live

This share link expires in 1 week. For free permanent hosting and GPU upgrades, run `gradio deploy` from the terminal in the working
```