Import Libraries

```
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix
from sklearn.feature_selection import SelectKBest, chi2
```

Load Dataset

```
file_path = '/content/netflix_titles.csv'
data = pd.read_csv(file_path)
```

Data Preprocessing

```
data['director'].fillna('Unknown', inplace=True)
data['cast'].fillna('Unknown', inplace=True)
data['country'].fillna('Unknown', inplace=True)
data['date_added'].fillna('Unknown', inplace=True)
data['rating'].fillna('Not Rated', inplace=True)
data['duration'].fillna('Unknown', inplace=True)
```

```
data['date_added'] = pd.to_datetime(data['date_added'], errors='coerce')
```

```
data['duration_numeric'] = data['duration'].str.extract('(\d+)').astype(float)


print(data.info())
print(data.head())
```

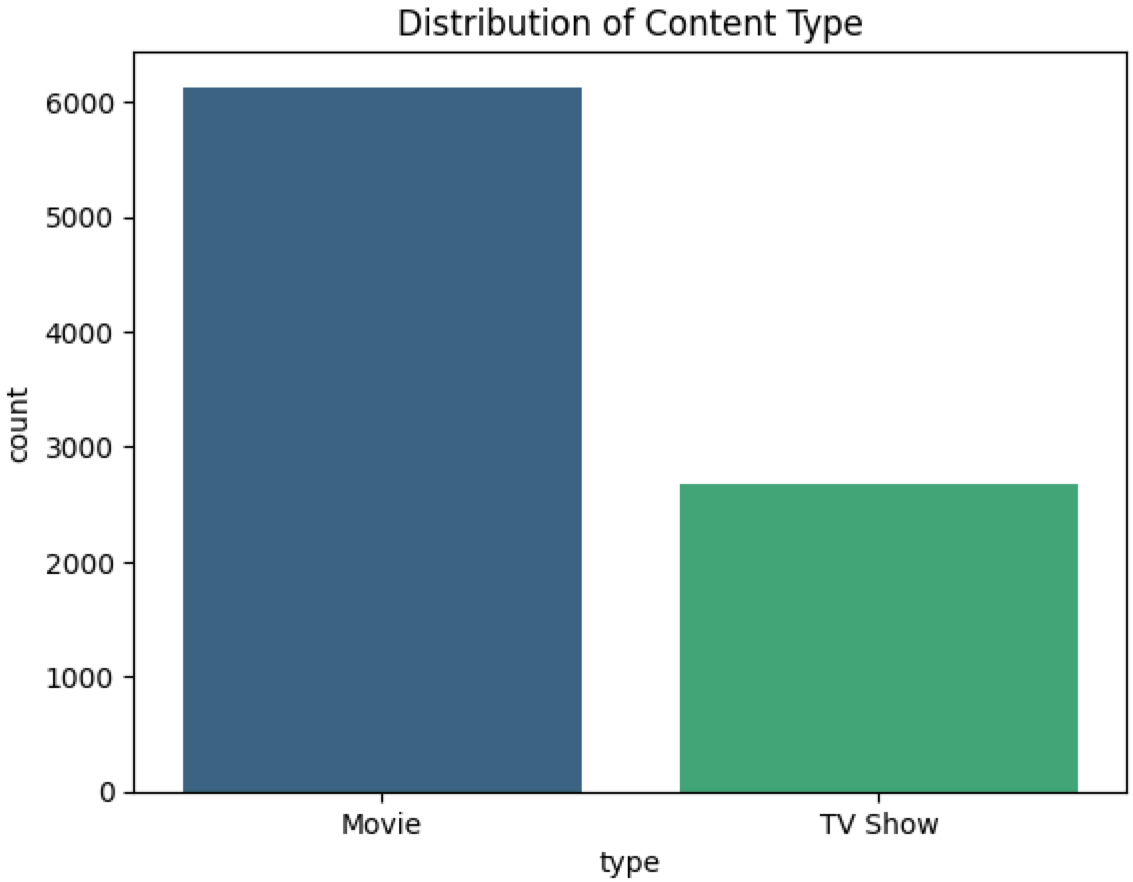
```
3                                            Unknown        Unknown
4  Mayur More, Jitendra Kumar, Ranjan Raj, Alam K...          India

  date_added  release_year rating   duration  \
0 2021-09-25          2020  PG-13     90 min
1 2021-09-24          2021  TV-MA  2 Seasons
2 2021-09-24          2021  TV-MA   1 Season
3 2021-09-24          2021  TV-MA   1 Season
4 2021-09-24          2021  TV-MA  2 Seasons

                                          listed_in  \
0                                     Documentaries
1    International TV Shows, TV Dramas, TV Mysteries
2  Crime TV Shows, International TV Shows, TV Act...
3                            Docuseries, Reality TV
4  International TV Shows, Romantic TV Shows, TV ...

                                        description  duration_numeric
0  As her father nears the end of his life, filmm...              90.0
1  After crossing paths at a party, a Cape Town t...               2.0
2  To protect his family from a powerful drug lor...               1.0
3  Feuds, flirtations and toilet talk go down amo...               1.0
4  In a city of coaching centers known to train I...               2.0
```

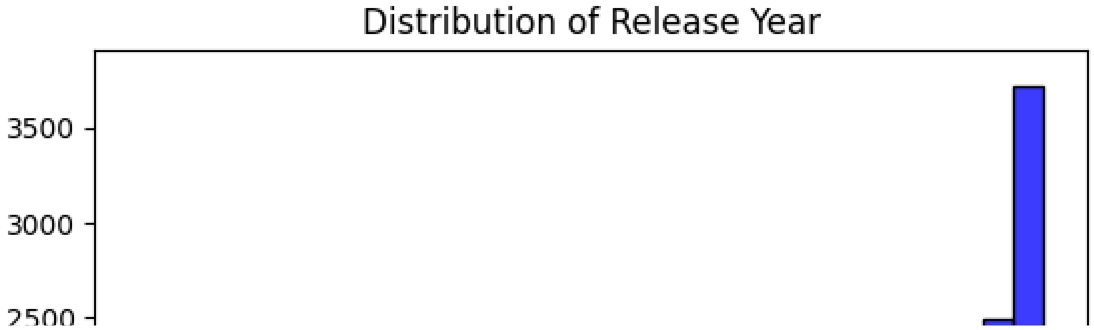
Exploratory Data Analysis (EDA)

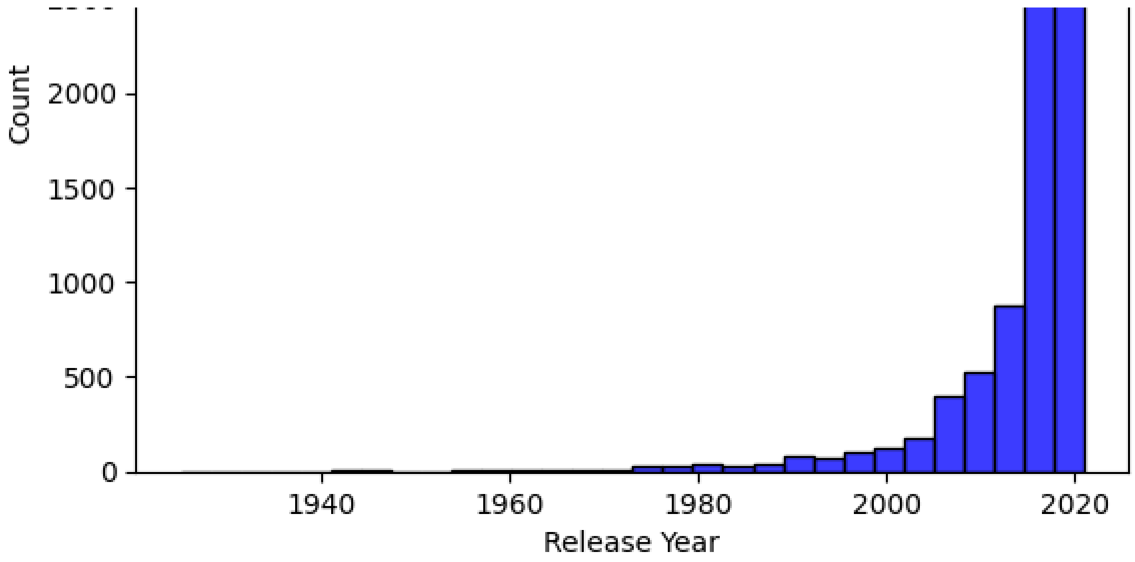
```
sns.countplot(x='type', data=data, palette='viridis')
plt.title('Distribution of Content Type')
plt.show()
```

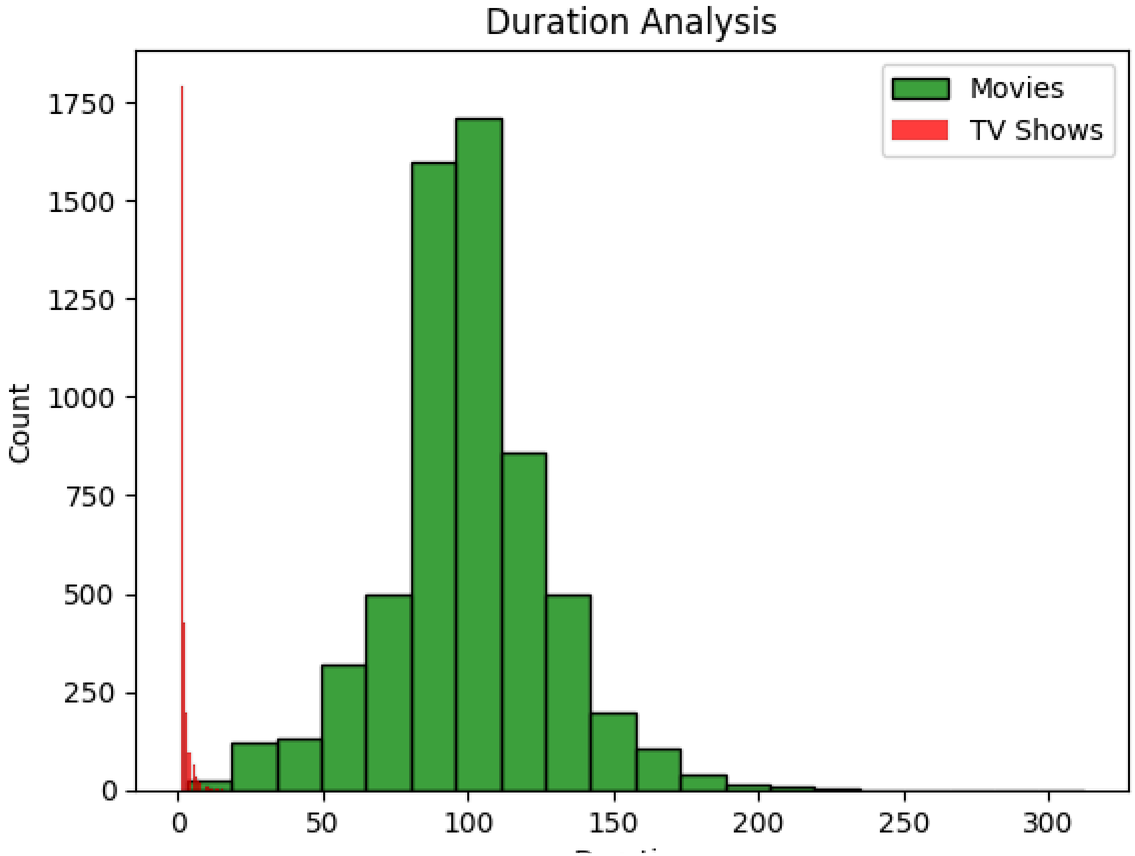
```
sns.histplot(data['release_year'], kde=False, bins=30, color='blue')
plt.title('Distribution of Release Year')
plt.xlabel('Release Year')
plt.ylabel('Count')
plt.show()


movies = data[data['type'] == 'Movie']
tv_shows = data[data['type'] == 'TV Show']

sns.histplot(movies['duration_numeric'].dropna(), bins=20, color='green', label='Movies')
sns.histplot(tv_shows['duration_numeric'].dropna(), bins=20, color='red', label='TV Shows')
plt.legend()
plt.title('Duration Analysis')
plt.xlabel('Duration')
plt.ylabel('Count')
plt.show()
```

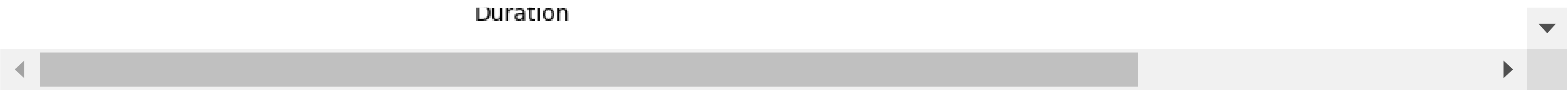
```
<ipython-input-4-12d33500fc5f>:2: FutureWarning:

Passing `palette` without assigning `hue` is deprecated and will be removed in v0.14.0. Assign the `x` variable to `hue` and

  sns.countplot(x='type', data=data, palette='viridis')
```






Count
2000
1500
1000
500
0
1940
1960
1980
2000
2020
Release Year


Duration Analysis
Movies
TV Shows
Count
1750
1500
1250
1000
750
500
250
0
0
50
100
150
200
250
300


Duration

Feature Selection

```
X = data[['release_year', 'duration_numeric']]
X.fillna(0, inplace=True)
y = data['type'].map({'Movie': 0, 'TV Show': 1})
```

```
selector = SelectKBest(chi2, k=2)
X_selected = selector.fit_transform(X, y)
print('Selected Features:', selector.get_support(indices=True))
```

```
Selected Features: [0 1]
<ipython-input-5-5bc2757e9d93>:3: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-ver
  X.fillna(0, inplace=True)
```

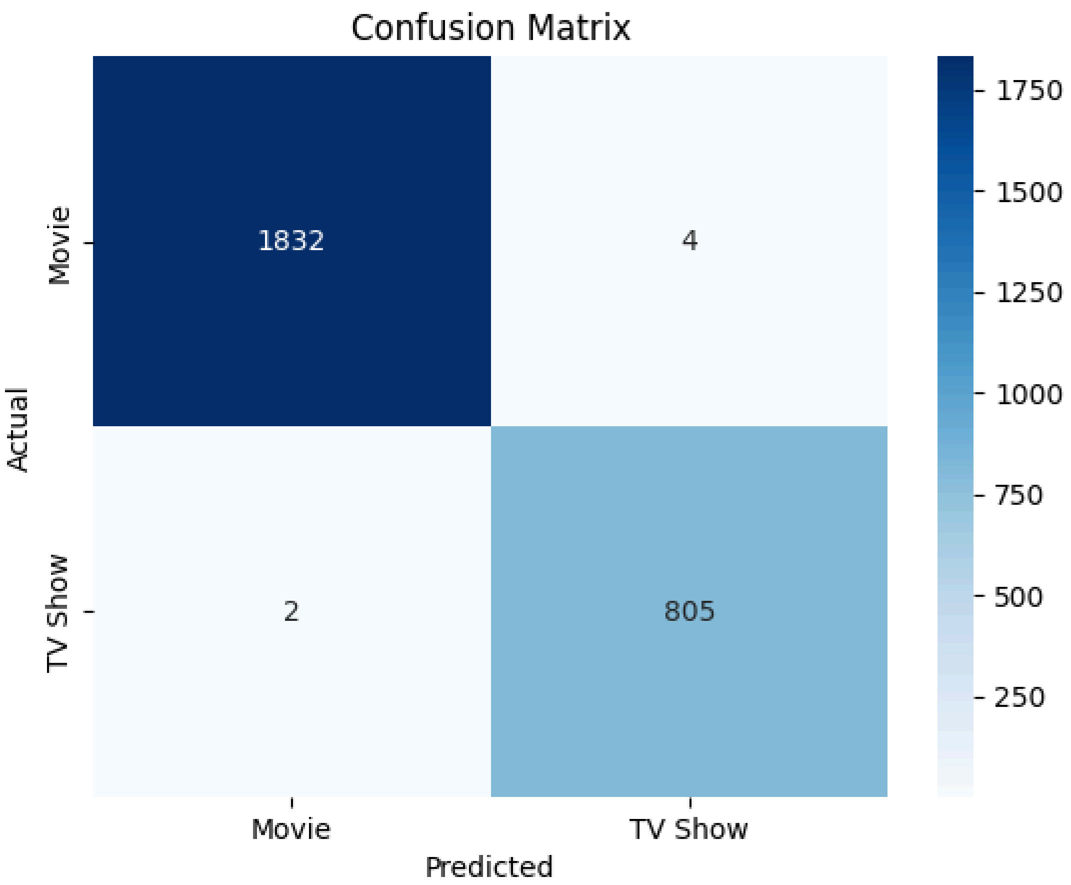
Modeling

```
X_train, X_test, y_train, y_test = train_test_split(X_selected, y, test_size=0.3, random_state=42)

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

# Save model for future use
import joblib
joblib.dump(model, '/content/random_forest_model.pkl')
```

```
['/content/random_forest_model.pkl']
```

Model Evaluation

```
# Predictions and evaluation
predictions = model.predict(X_test)

# Confusion matrix
cm = confusion_matrix(y_test, predictions)
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues', xticklabels=['Movie', 'TV Show'], yticklabels=['Movie', 'TV Show'])
plt.title('Confusion Matrix')
plt.xlabel('Predicted')
plt.ylabel('Actual')
plt.show()

# Classification report
print(classification_report(y_test, predictions))
```



```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      1836
           1       1.00      1.00      1.00       807

    accuracy                           1.00      2643
   macro avg       1.00      1.00      1.00      2643
weighted avg       1.00      1.00      1.00      2643
```

Logistic Regression

```
from sklearn.linear_model import LogisticRegression
model_lr = LogisticRegression(random_state=42)
model_lr.fit(X_train, y_train)
predictions_lr = model_lr.predict(X_test)
print(classification_report(y_test, predictions_lr))
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00      1836
           1       1.00      1.00      1.00       807

    accuracy                           1.00      2643
   macro avg       1.00      1.00      1.00      2643
weighted avg       1.00      1.00      1.00      2643
```

Support Vector Machine (SVM)

```
from sklearn.svm import SVC
model_svm = SVC(kernel='linear', random_state=42)
model_svm.fit(X_train, y_train)
predictions_svm = model_svm.predict(X_test)
print(classification_report(y_test, predictions_svm))
```