```
!pip install nltk
```

```
Requirement already satisfied: nltk in /usr/local/lib/python3.10/dist-packages (3.8.1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from nltk) (8.1.7)
Requirement already satisfied: joblib in /usr/local/lib/python3.10/dist-packages (from nltk) (1.4.2)
Requirement already satisfied: regex>=2021.8.3 in /usr/local/lib/python3.10/dist-packages (from nltk) (2024.5.15)
Requirement already satisfied: tqdm in /usr/local/lib/python3.10/dist-packages (from nltk) (4.66.4)
```

```
import numpy as np
import pandas as pd
import nltk
import re
```

```
train_path = "/content/train_data.txt"
try:
    ds = pd.read_csv(train_path, sep=':::', names=['Title', 'Genre', 'Description'], engine='python')
    print(ds)
except FileNotFoundError:
    print("File not found. Please check the file path and ensure the file exists.")
```

```
                                              Title          Genre  \
1                      Oscar et la dame rose (2009)          drama
2                                      Cupid (1997)       thriller
3                  Young, Wild and Wonderful (1980)          adult
4                             The Secret Sin (1915)          drama
5                            The Unrecovered (2007)          drama
...                                             ...            ...
54210                                "Bonino" (1953)         comedy
54211                       Dead Girls Don't Cry (????)         horror
54212     Ronald Goedemondt: Ze bestaan echt (2008)    documentary
54213                          Make Your Own Bed (1944)         comedy
54214    Nature's Fury: Storm of the Century (2006)        history

                                             Description
1       Listening in to a conversation between his do...
2       A brother and sister with a past incestuous r...
3       As the bus empties the students for their fie...
4       To help their unemployed father make ends mee...
5       The film's title refers not only to the un-re...
...                                                  ...
54210   This short-lived NBC live sitcom centered on ...
54211   The NEXT Generation of EXPLOITATION. The sist...
54212   Ze bestaan echt, is a stand-up comedy about g...
54213   Walter and Vivian live in the country and hav...
54214   On Labor Day Weekend, 1935, the most intense ...

[54214 rows x 3 columns]
```

```
ds.head()
```

	Title	Genre	Description
1	Oscar et la dame rose (2009)	drama	Listening in to a conversation between his do...
2	Cupid (1997)	thriller	A brother and sister with a past incestuous r...
3	Young, Wild and Wonderful (1980)	adult	As the bus empties the students for their fie...
4	The Secret Sin (1915)	drama	To help their unemployed father make ends mee...

Next steps: Generate code with ds   View recommended plots

```
ds.isna().sum()
```

```
Title          0
Genre          0
Description    0
dtype: int64
```

```
ds.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 54214 entries, 1 to 54214
Data columns (total 3 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   Title        54214 non-null  object
 1   Genre        54214 non-null  object
 2   Description  54214 non-null  object
```

```
    dtypes: object(3)
    memory usage: 1.7+ MB
```

```
ds.duplicated().sum()
```

```
0
```

```
nltk.download('stopwords')
nltk.download('punkt')
nltk.download('wordnet')
from nltk.corpus import stopwords
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
```

```
[nltk_data] Downloading package stopwords to /root/nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
[nltk_data] Downloading package punkt to /root/nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package wordnet to /root/nltk_data...
[nltk_data]   Package wordnet is already up-to-date!
```

```
stopword = set(stopwords.words('english'))

def preprocessing(text):
    # Convert text to lowercase
    text = text.lower()

    # Remove punctuation using regular expressions
    text = re.sub(r'[^\w\s]', '', text)

    # Remove specific characters #, @, and $
    text = re.sub(r'[#@\$]', '', text)

    # tokenize and convert to list
    tokens = word_tokenize(text)

    ## Lemmatize it
    lemmatizer  = WordNetLemmatizer()

    ## lemmatize each token
    text = [lemmatizer.lemmatize(token) for token in tokens]

    text = [word for word in text if word not in stopword]


    return " ".join(text)
```

```
ds["Despcription_clean"] =  ds["Description"].apply(preprocessing)
```

```
ds.head()
```

	Title	Genre	Description	Despcription_clean
1	Oscar et la dame rose (2009)	drama	Listening in to a conversation between his do...	listening conversation doctor parent 10yearold...
2	Cupid (1997)	thriller	A brother and sister with a past incestuous r...	brother sister past incestuous relationship cu...
3	Young, Wild and Wonderful (1980)	adult	As the bus empties the students for their fie...	bus empty student field trip museum natural hi...

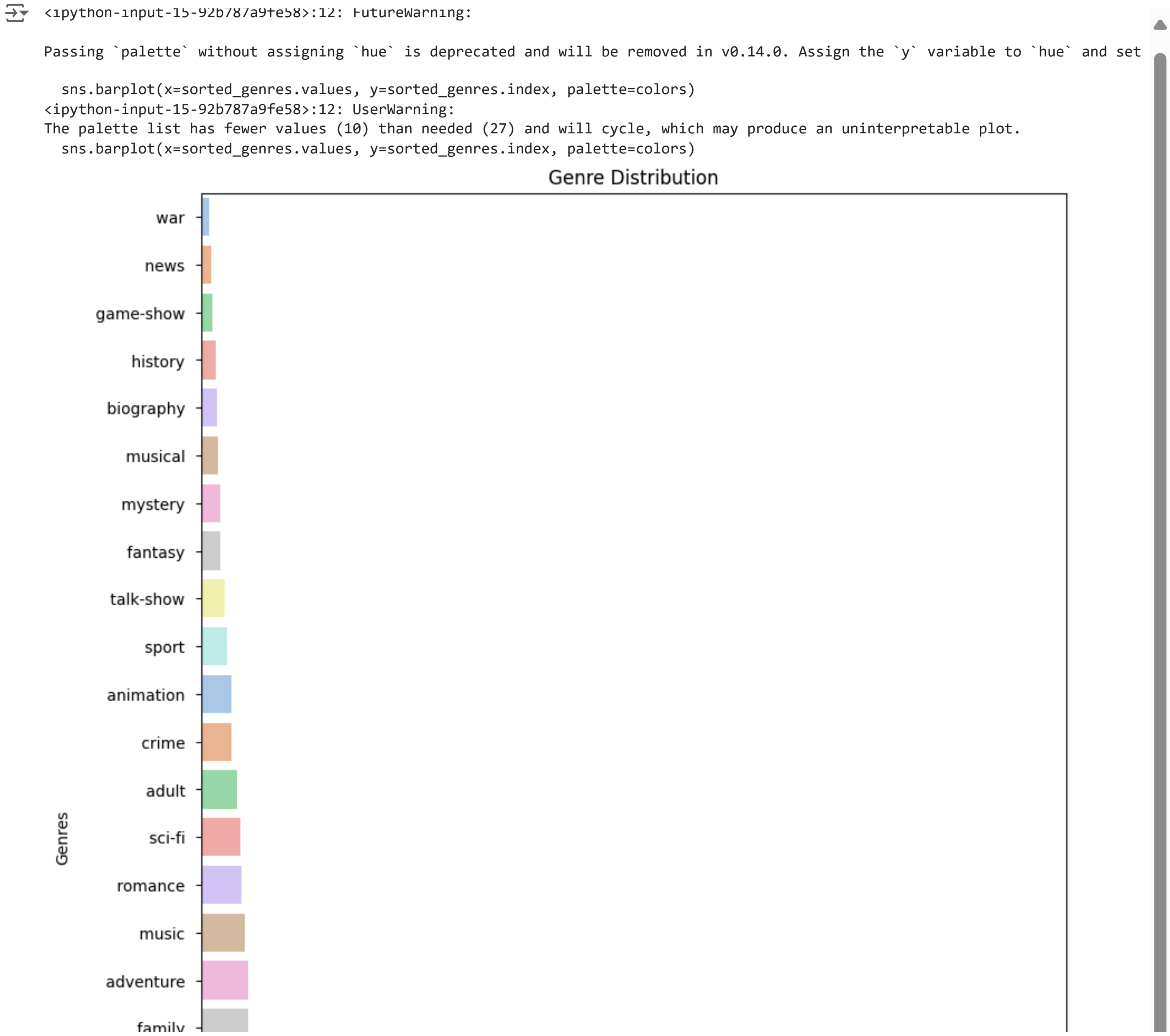
Next steps: Generate code with ds View recommended plots

```
ds['Genre'].value_counts()
```

```
Genre
 drama          13613
 documentary    13096
 comedy          7447
 short           5073
 horror          2204
 thriller        1591
 action          1315
 western         1032
 reality-tv       884
 family           784
 adventure        775
 music            731
 romance          672
```

```
      sci-fi            647
      adult             590
      crime             505
      animation         498
      sport             432
      talk-show         391
      fantasy           323
      mystery           319
      musical           277
      biography         265
      history           243
      game-show         194
      news              181
      war               132
     Name: count, dtype: int64
```

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


# Count the frequency of each genre
genre_counts = ds["Genre"].value_counts()

# Sort genres based on frequency
sorted_genres = genre_counts.sort_values(ascending=True)

# Set the color palette
colors = sns.color_palette("pastel")

# Create a horizontal bar chart with Seaborn for a stylish visualization
plt.figure(figsize=(10, 15))
sns.barplot(x=sorted_genres.values, y=sorted_genres.index, palette=colors)

plt.title("Genre Distribution")
plt.xlabel("Frequency")
plt.ylabel("Genres")
plt.show()
```



```
<ipython-input-15-92b787a9fe58>:12: FutureWarning:

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

  sns.barplot(x=sorted_genres.values, y=sorted_genres.index, palette=colors)
<ipython-input-15-92b787a9fe58>:12: UserWarning:
The palette list has fewer values (10) than needed (27) and will cycle, which may produce an uninterpretable plot.
  sns.barplot(x=sorted_genres.values, y=sorted_genres.index, palette=colors)
```