Task1 : Movie Genre Classification

We are tasked to design an Movie Genre classifier based on previous movie data in this jupyter notebook we are going to
create a machine learning model that can predict the genre of a movie based on its plot summary or other textual
information. we are going to use techniques like TF-IDF or word embeddings with classifiers such as Naive Bayes,
Logistic Regression, or Support Vector Machines.

Methodology:

1. Data Collection

2. Data Cleaning and Preprocessing

3. Exploratory Data Analysis (EDA)

4. Feature Engineering

5. Model Selection

6. Model Training and Evaluation

Import necessary libraries

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import re
from nltk.tokenize import word_tokenize
from nltk.stem import WordNetLemmatizer
from nltk.corpus import stopwords
from wordcloud import WordCloud
```

In [2]:
```
import nltk
nltk.download('punkt')
nltk.download('stopwords')
```

```
[nltk_data] Downloading package punkt to
[nltk_data]     C:\Users\Barcha\AppData\Roaming\nltk_data...
[nltk_data]   Package punkt is already up-to-date!
[nltk_data] Downloading package stopwords to
[nltk_data]     C:\Users\Barcha\AppData\Roaming\nltk_data...
[nltk_data]   Package stopwords is already up-to-date!
```

Out[2]:
```
True
```

In [3]:
```
import warnings

# Ignore all warnings
warnings.filterwarnings("ignore")
```

1. Data Collection

For the purpose of designing a Movie Genre classifier, we collected the dataset from Kaggle. The dat
aset consists of movie-related information sourced from the Internet Movie Database (IMDb). IMDb is

In [4]:
```
# Read the txt files using pandas

train_data = pd.read_csv("./Genre Classification Dataset/train_data.txt", delimiter=':::' ,header = None ,eng
test_data  = pd.read_csv("./Genre Classification Dataset/test_data.txt", delimiter=':::' ,header = None ,engi

test_data_solution  = pd.read_csv("./Genre Classification Dataset/test_data_solution.txt", delimiter=':::' ,h
```

In [ ]:
```
```

In [5]:
```
## View train data
print("shape",train_data.shape)
train_data.head()
```

```
shape (54214, 4)
```

Out[5]:

	0	1	2	3
0	1	Oscar et la dame rose (2009)	drama	Listening in to a conversation between his do...
1	2	Cupid (1997)	thriller	A brother and sister with a past incestuous r...
2	3	Young, Wild and Wonderful (1980)	adult	As the bus empties the students for their fie...
3	4	The Secret Sin (1915)	drama	To help their unemployed father make ends mee...
4	5	The Unrecovered (2007)	drama	The film's title refers not only to the un-re...

In [6]:
```
## View the test solution data
print("shape",test_data_solution.shape)
test_data_solution.head()
```

```
shape (54200, 4)
```

Out[6]:

	0	1	2	3
0	1	Edgar's Lunch (1998)	thriller	L.R. Brane loves his life - his car, his apar...
1	2	La guerra de papá (1977)	comedy	Spain, March 1964: Quico is a very naughty ch...
2	3	Off the Beaten Track (2010)	documentary	One year in the life of Albin and his family ...
3	4	Meu Amigo Hindu (2015)	drama	His father has died, he hasn't spoken with hi...
4	5	Er nu zhai (1955)	drama	Before he was known internationally as a mart...

In [7]:
```
## We will concat the test and train file

df = pd.concat((train_data ,test_data_solution))
df.columns = ["id" ,"Title","Genre","Description"]
df.head()
```

Out[7]:

	id	Title	Genre	Description
0	1	Oscar et la dame rose (2009)	drama	Listening in to a conversation between his do...
1	2	Cupid (1997)	thriller	A brother and sister with a past incestuous r...
2	3	Young, Wild and Wonderful (1980)	adult	As the bus empties the students for their fie...
3	4	The Secret Sin (1915)	drama	To help their unemployed father make ends mee...
4	5	The Unrecovered (2007)	drama	The film's title refers not only to the un-re...

In [8]:
```
## Check the size
df.shape
```

Out[8]:
```
(108414, 4)
```

2. Data Cleaning and Preprocessing

We will clean and preprocess the data
- Removing duplicates
- Removing Nan rows and column
- Preprocessing the Data

In [9]:
```
## Check for Duplicates and Remove them
df.duplicated().sum() ## Will give us a number of duplicates

df.drop_duplicates(inplace = True)  ## Will drops any duplicates
```

In [10]:
```
## Check for nan values

df.isna().sum()  # Will check for any duplicates

df.dropna( inplace = True ) ## Will drop any nan containing row if exists
```

In [11]:
```
## Check the size
df.shape
```

Out[11]: (108414, 4)

In [ ]:
```

```

In [12]:
```
## function to preprocess the data
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)
```

In [13]:
```
## Create list of words in discription column
df["Despcription_clean"] =  df["Description"].apply(preprocessing)
```

In [14]:
```
df.head()
```

Out[14]:

	id	Title	Genre	Description	Despcription_clean
0	1	Oscar et la dame rose (2009)	drama	Listening in to a conversation between his do...	listening conversation doctor parent 10yearold...
1	2	Cupid (1997)	thriller	A brother and sister with a past incestuous r...	brother sister past incestuous relationship cu...
2	3	Young, Wild and Wonderful (1980)	adult	As the bus empties the students for their fie...	bus empty student field trip museum natural hi...
3	4	The Secret Sin (1915)	drama	To help their unemployed father make ends mee...	help unemployed father make end meet edith twi...
4	5	The Unrecovered (2007)	drama	The film's title refers not only to the un-re...	film title refers unrecovered body ground zero...

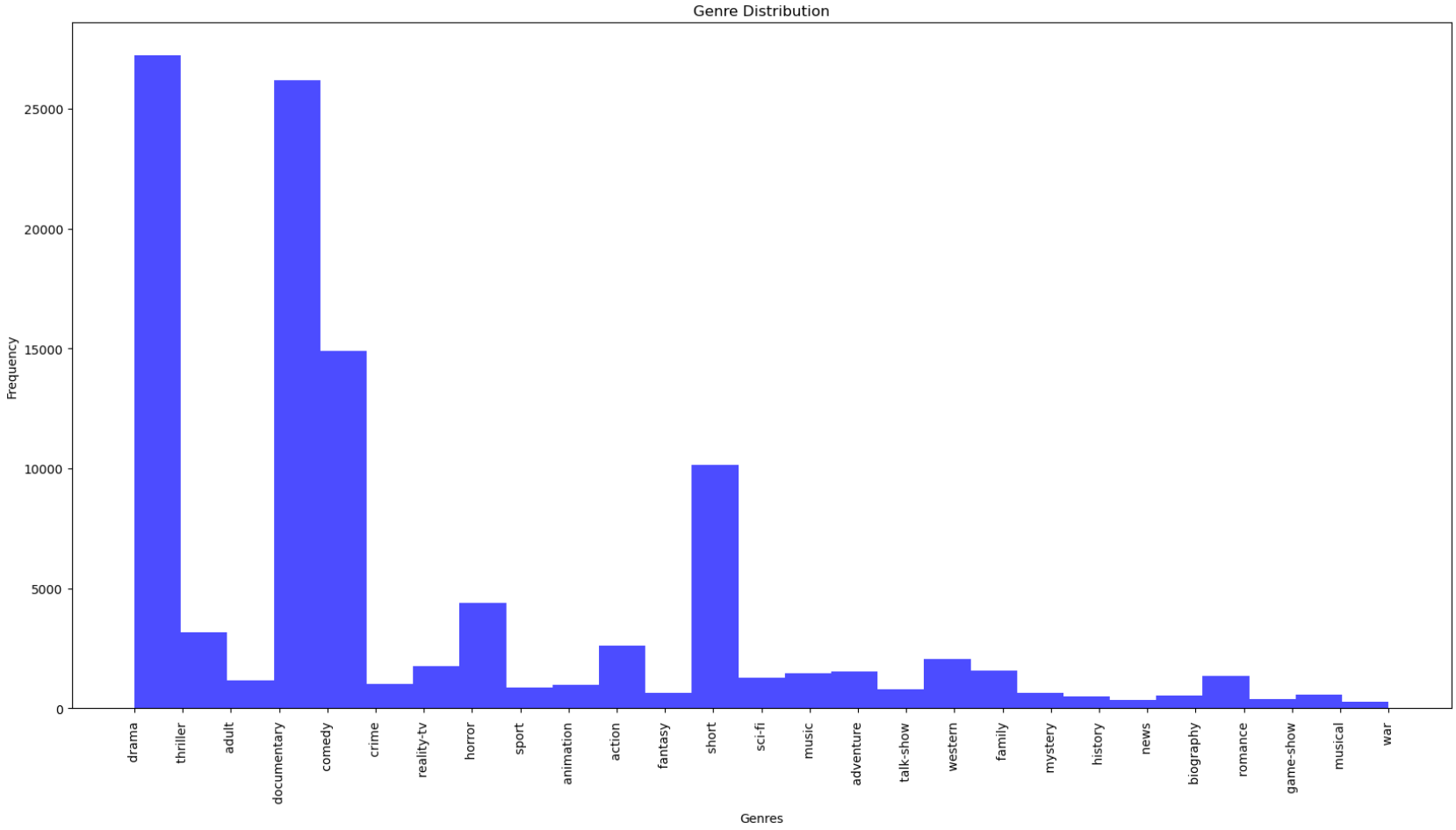
3. Exploratory Data Analysis (EDA)

Will try to analyse the data using Histogram and Bar Chart.
And will show word cloud for each genre.

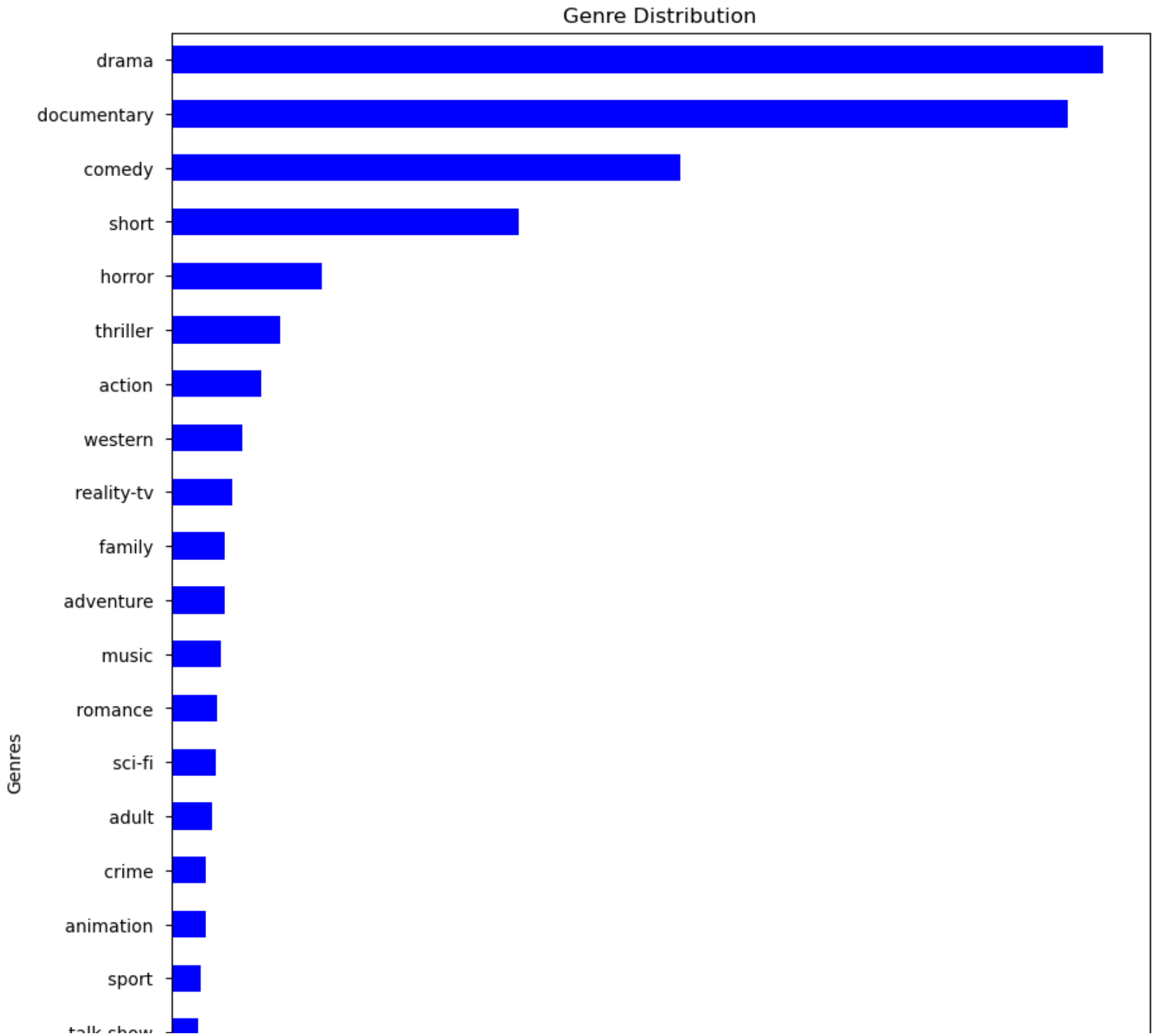
In [15]:
```
## Shows us the label counts
df["Genre"].value_counts()
```

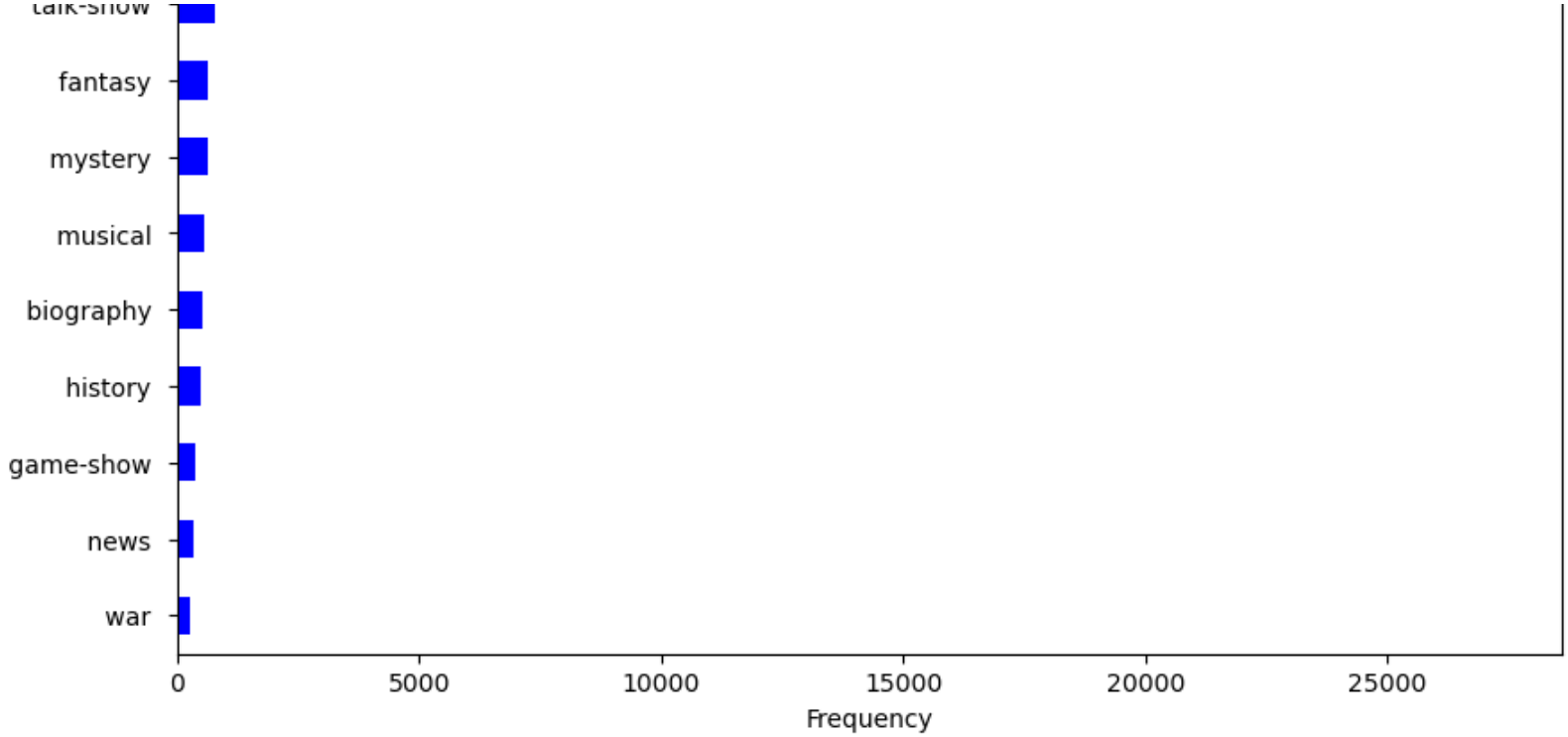
Out[15]:
```
 drama           27225
 documentary     26192
 comedy          14893
 short           10145
 horror           4408
 thriller         3181
 action           2629
 western          2064
 reality-tv       1767
 family           1567
 adventure        1550
 music            1462
 romance          1344
 sci-fi           1293
 adult            1180
 crime            1010
 animation         996
 sport             863
 talk-show         782
 fantasy           645
 mystery           637
 musical           553
 biography         529
 history           486
 game-show         387
 news              362
 war               264
Name: Genre, dtype: int64
```

In [16]:
```
# Create a histogram of genre distribution
plt.figure(figsize=(20, 10))
plt.hist(df["Genre"],bins =27 , color='blue', alpha=0.7)
plt.title("Genre Distribution")
plt.xlabel("Genres")
plt.ylabel("Frequency")
plt.xticks(rotation=90)
plt.show()
```



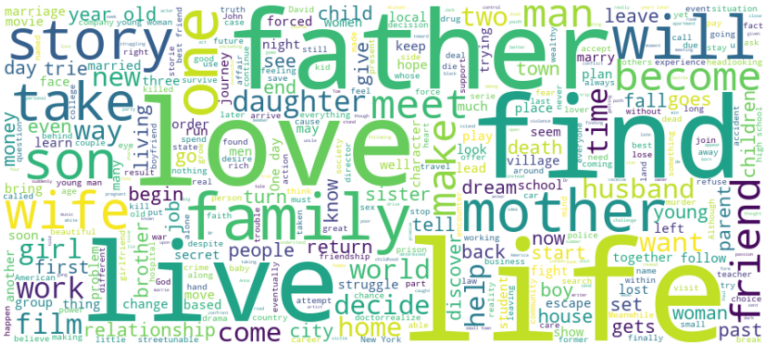
In [17]:
```
## View genre distribution on Horizontal graph

genre_counts = df["Genre"].value_counts()
sorted_genres = genre_counts.sort_values(ascending=True)
# Create a horizontal histogram of genre distribution
plt.figure(figsize=(10, 15))
sorted_genres.plot(kind='barh',color = "blue", alpha=1 )
plt.title("Genre Distribution")
plt.xlabel("Frequency")
plt.ylabel("Genres")
plt.show()
```


Genre Distribution
drama
documentary
comedy
short
horror
thriller
action
western
reality-tv
family
adventure
music
romance
sci-fi
adult
crime
animation
sport
talk-show
Genres

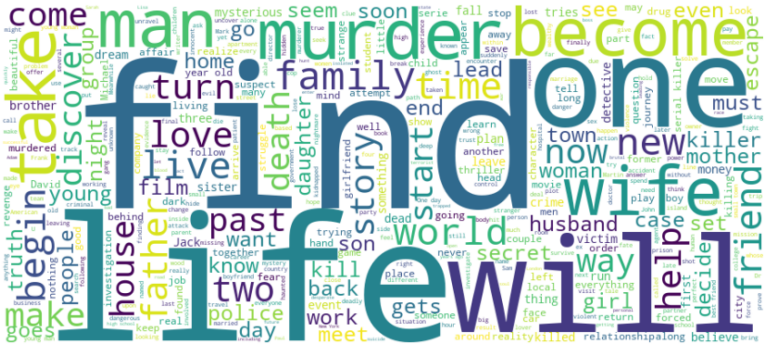

talk-show
fantasy
mystery
musical
biography
history
game-show
news
war
0
5000
10000
15000
20000
25000
Frequency

In [18]:
```
## Below code will generate wordcloud of each genre

# Create a list of unique genres from the dataset
genres = df['Genre'].unique()

# Set the figure size outside the loop
plt.figure(figsize=(20, 60))

# Iterate over each genre
for i, genre in enumerate(genres, 1):
    plt.subplot(14, 2, i)  # Assuming you have 14 rows and 2 columns for 28 genres
    text_subset = " ".join(list(df[df["Genre"] == genre]['Description']))
    wordcloud = WordCloud(max_words=400, width=900, height=400, background_color='white').generate(text_subse
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.title(f'Word Cloud for {genre} Genre',fontsize=30)
    plt.axis('off')


plt.tight_layout()
plt.show()
```

Word Cloud for drama Genre



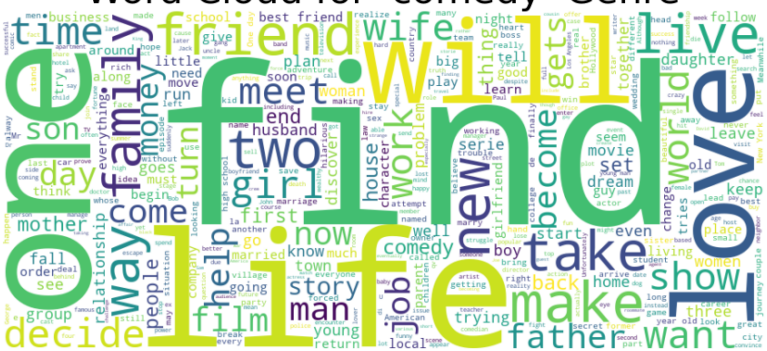
Word Cloud for thriller Genre



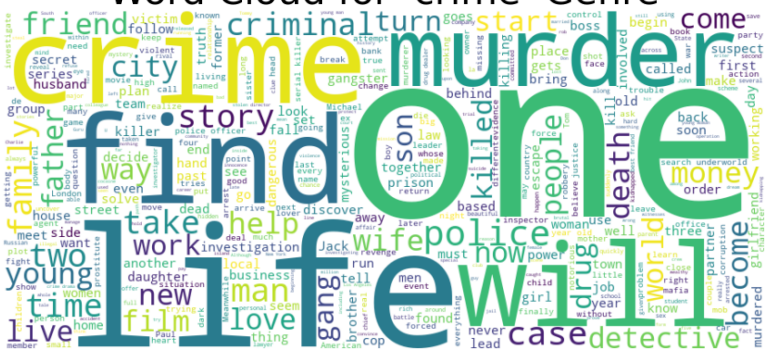
Word Cloud for adult Genre

Word Cloud for documentary Genre

Word Cloud for comedy Genre



Word Cloud for crime Genre

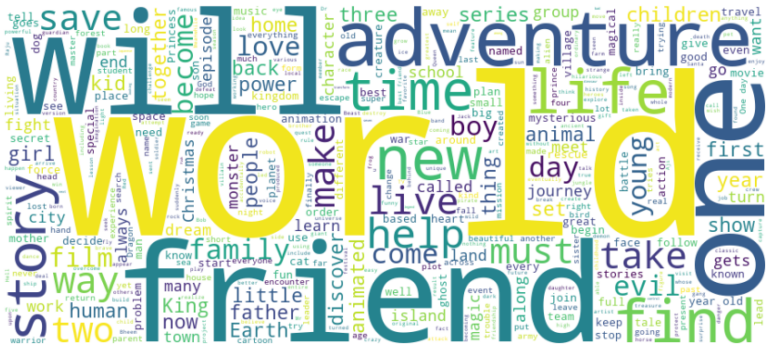


Word Cloud for reality-tv Genre

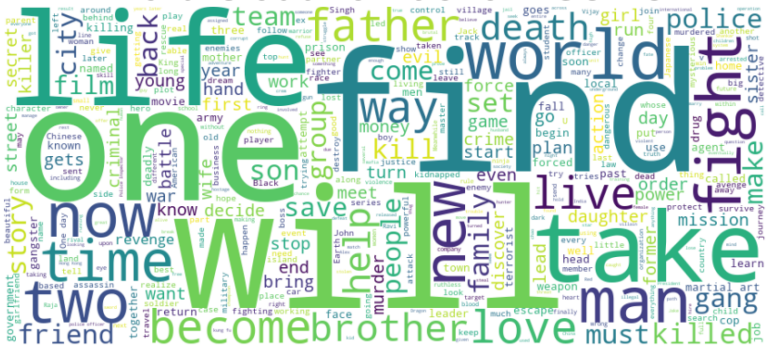
Word Cloud for horror Genre

Word Cloud for sport Genre

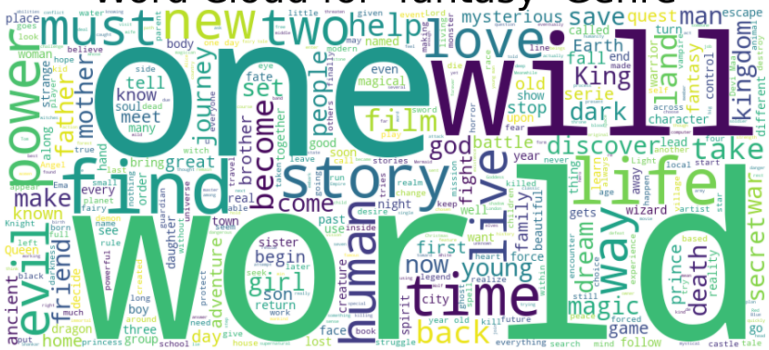
Word Cloud for animation Genre



Word Cloud for action Genre



Word Cloud for fantasy Genre



Word Cloud for short Genre

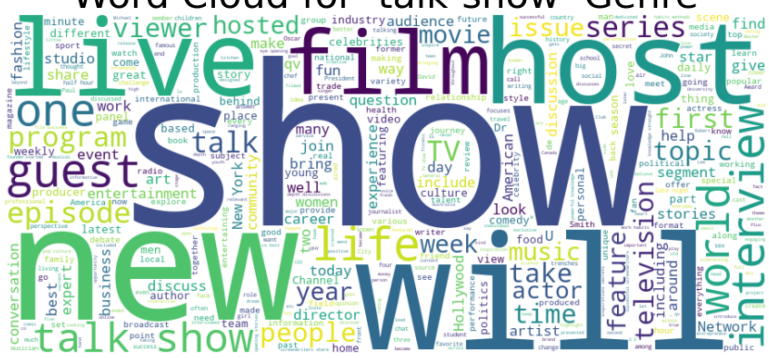
Word Cloud for sci-fi Genre

Word Cloud for music Genre

Word Cloud for adventure Genre

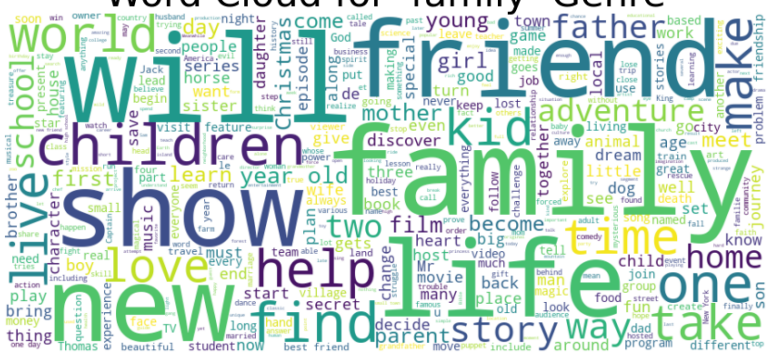
Word Cloud for talk-show Genre

Word Cloud for western Genre



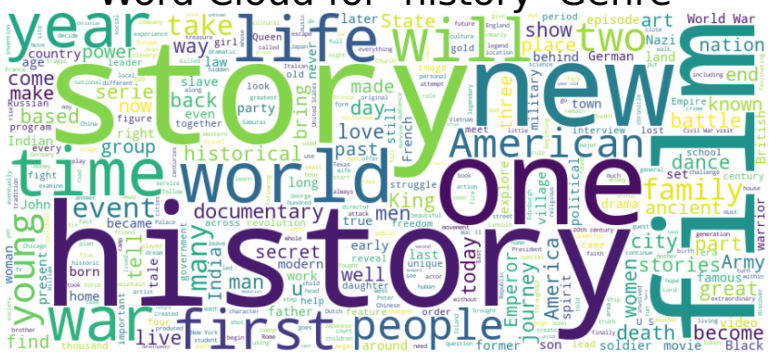
Word Cloud for family Genre

Word Cloud for mystery Genre

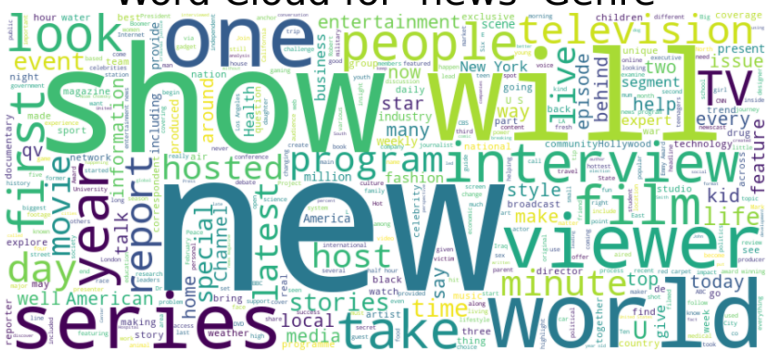


Word Cloud for history Genre

Word Cloud for news Genre



Word Cloud for biography Genre

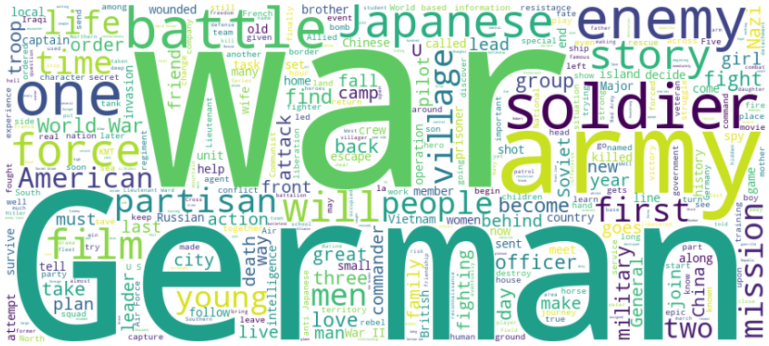


Word Cloud for romance Genre

Word Cloud for game-show Genre

Word Cloud for musical Genre

Word Cloud for war Genre



4. Feature Engineering

In this step we will remove unnecessary column which are not necessary for our model

In [19]:
```
## remove id column from head
data = df.drop(["Title","id"] , axis = 1) # will drop column
data.head()
```

Out[19]:

	Genre	Description	Despcription_clean
0	drama	Listening in to a conversation between his do...	listening conversation doctor parent 10yearold...
1	thriller	A brother and sister with a past incestuous r...	brother sister past incestuous relationship cu...
2	adult	As the bus empties the students for their fie...	bus empty student field trip museum natural hi...
3	drama	To help their unemployed father make ends mee...	help unemployed father make end meet edith twi...
4	drama	The film's title refers not only to the un-re...	film title refers unrecovered body ground zero...

5. Model Selection & Training

- In this section we will try to apply CountVectorizer and TF-IDF technique and try to predict accuracy on model.

Importing necessary libraries for model selection and training

In [20]:
```
## import necessary library for
from sklearn.linear_model import LogisticRegression
from sklearn.svm import LinearSVC
from sklearn.naive_bayes import MultinomialNB
from sklearn.ensemble import RandomForestClassifier

from sklearn.feature_extraction.text import CountVectorizer
from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.model_selection import train_test_split

from sklearn.metrics import classification_report,confusion_matrix,ConfusionMatrixDisplay
```

Converting Genre into Numerical form

In [21]:
```
#Convert sentiment labels to numerical values for modeling
from sklearn.preprocessing import LabelEncoder
label_encoder = LabelEncoder()
data['Genre_encoded'] = label_encoder.fit_transform(data['Genre'])
data['Genre_encoded']

class_names= list(label_encoder.classes_)
class_names
```

Out[21]:
```
[' action ',
 ' adult ',
 ' adventure ',
 ' animation ',
 ' biography ',
 ' comedy ',
 ' crime ',
 ' documentary ',
 ' drama ',
 ' family ',
 ' fantasy ',
 ' game-show ',
 ' history ',
 ' horror ',
 ' music ',
 ' musical ',
 ' mystery ',
 ' news ',
 ' reality-tv ',
 ' romance ',
 ' sci-fi ',
 ' short ',
 ' sport ',
 ' talk-show ',
 ' thriller ',
 ' war ',
 ' western ']
```

In [22]:
```
data.head()
```

Out[22]:

	Genre	Description	Despcription_clean	Genre_encoded
0	drama	Listening in to a conversation between his do...	listening conversation doctor parent 10yearold...	8
1	thriller	A brother and sister with a past incestuous r...	brother sister past incestuous relationship cu...	24
2	adult	As the bus empties the students for their fie...	bus empty student field trip museum natural hi...	1
3	drama	To help their unemployed father make ends mee...	help unemployed father make end meet edith twi...	8
4	drama	The film's title refers not only to the un-re...	film title refers unrecovered body ground zero...	8

Split the data to test and train

In [23]:
```
## Split the data
x = data["Despcription_clean"]
y = data["Genre"]

x_train ,x_test ,y_train ,y_test = train_test_split(x ,y ,test_size = 0.5)
```

Model training using CountVectorizer technique

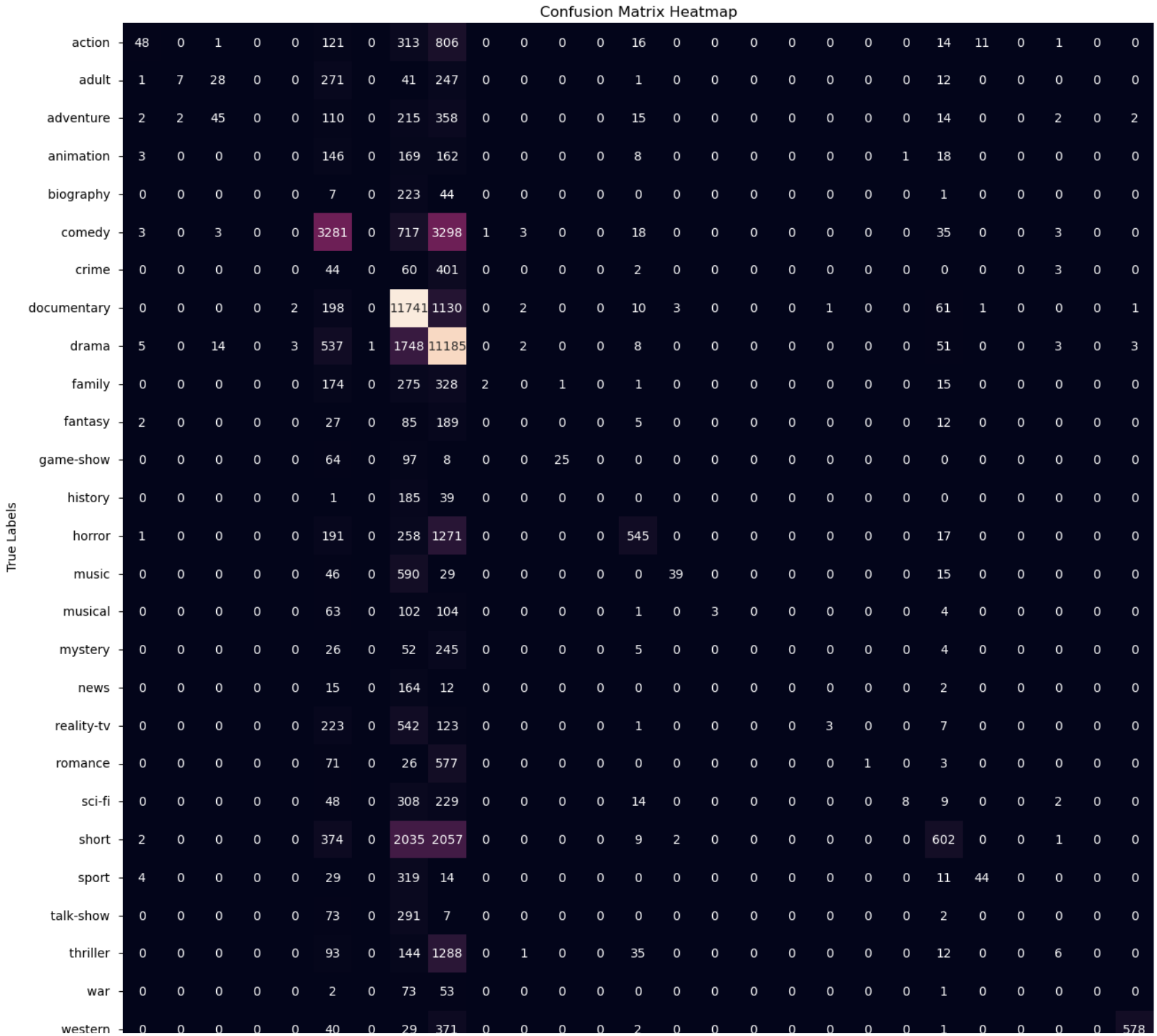
In [24]:
```
vectorize = CountVectorizer()
x_train1 = vectorize.fit_transform(x_train)
x_test1 =  vectorize.transform(x_test)
```

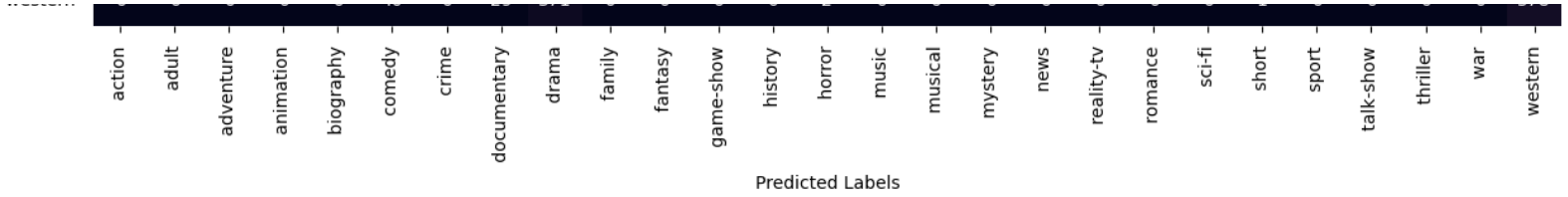
MultinomialNB

In [25]:
```
mnb = MultinomialNB()
mnb.fit(x_train1 ,y_train)
print("Model Score on Training data",mnb.score(x_train1 ,y_train))
print("Model Score on Training data",mnb.score(x_test1 ,y_test))
y_pred = mnb.predict(x_test1)

print(classification_report(y_pred ,y_test))

cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(15, 15))  # Adjust the figure size as needed
sns.heatmap(cm, annot=True, fmt='d', cbar=False,
            xticklabels=class_names, yticklabels=class_names)  # Replace 'class_names' with your class labels
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix Heatmap')
plt.show()
```

```
Model Score on Training data 0.6284797166417622
Model Score on Training data 0.51954544616009
```

	precision	recall	f1-score	support
action	0.04	0.68	0.07	71
adult	0.01	0.78	0.02	9
adventure	0.06	0.49	0.11	91
animation	0.00	0.00	0.00	0
biography	0.00	0.00	0.00	5
comedy	0.45	0.52	0.48	6275
crime	0.00	0.00	0.00	1
documentary	0.89	0.56	0.69	20802
drama	0.82	0.46	0.59	24575
family	0.00	0.67	0.01	3
fantasy	0.00	0.00	0.00	8
game-show	0.13	0.96	0.23	26
history	0.00	0.00	0.00	0
horror	0.24	0.78	0.37	696
music	0.05	0.89	0.10	44
musical	0.01	1.00	0.02	3
mystery	0.00	0.00	0.00	0
news	0.00	0.00	0.00	0
reality-tv	0.00	0.75	0.01	4
romance	0.00	1.00	0.00	1
sci-fi	0.01	0.89	0.03	9
short	0.12	0.65	0.20	923
sport	0.10	0.79	0.18	56
talk-show	0.00	0.00	0.00	0
thriller	0.00	0.29	0.01	21
war	0.00	0.00	0.00	0
western	0.57	0.99	0.72	584
accuracy			0.52	54207
macro avg	0.13	0.49	0.14	54207
weighted avg	0.78	0.52	0.60	54207


Confusion Matrix Heatmap
True Labels
action 48 0 1 0 0 121 0 313 806 0 0 0 0 16 0 0 0 0 0 0 0 14 11 0 1 0 0
adult 1 7 28 0 0 271 0 41 247 0 0 0 0 1 0 0 0 0 0 0 0 12 0 0 0 0 0
adventure 2 2 45 0 0 110 0 215 358 0 0 0 0 15 0 0 0 0 0 0 0 14 0 0 2 0 2
animation 3 0 0 0 0 146 0 169 162 0 0 0 0 8 0 0 0 0 0 0 1 18 0 0 0 0 0
biography 0 0 0 0 0 7 0 223 44 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
comedy 3 0 3 0 0 3281 0 717 3298 1 3 0 0 18 0 0 0 0 0 0 0 35 0 0 3 0 0
crime 0 0 0 0 0 44 0 60 401 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 3 0 0
documentary 0 0 0 0 2 198 0 11741 1130 0 2 0 0 10 3 0 0 0 1 0 0 61 1 0 0 0 1
drama 5 0 14 0 3 537 1 1748 11185 0 2 0 0 8 0 0 0 0 0 0 0 51 0 0 3 0 3
family 0 0 0 0 0 174 0 275 328 2 0 1 0 1 0 0 0 0 0 0 0 15 0 0 0 0 0
fantasy 2 0 0 0 0 27 0 85 189 0 0 0 0 5 0 0 0 0 0 0 0 12 0 0 0 0 0
game-show 0 0 0 0 0 64 0 97 8 0 0 25 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
history 0 0 0 0 0 1 0 185 39 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
horror 1 0 0 0 0 191 0 258 1271 0 0 0 0 545 0 0 0 0 0 0 0 17 0 0 0 0 0
music 0 0 0 0 0 46 0 590 29 0 0 0 0 0 39 0 0 0 0 0 0 15 0 0 0 0 0
musical 0 0 0 0 0 63 0 102 104 0 0 0 0 1 0 3 0 0 0 0 0 4 0 0 0 0 0
mystery 0 0 0 0 0 26 0 52 245 0 0 0 0 5 0 0 0 0 0 0 0 4 0 0 0 0 0
news 0 0 0 0 0 15 0 164 12 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0
reality-tv 0 0 0 0 0 223 0 542 123 0 0 0 0 1 0 0 0 0 3 0 0 7 0 0 0 0 0
romance 0 0 0 0 0 71 0 26 577 0 0 0 0 0 0 0 0 0 0 1 0 3 0 0 0 0 0
sci-fi 0 0 0 0 0 48 0 308 229 0 0 0 0 14 0 0 0 0 0 0 8 9 0 0 2 0 0
short 2 0 0 0 0 374 0 2035 2057 0 0 0 0 9 2 0 0 0 0 0 0 602 0 0 1 0 0
sport 4 0 0 0 0 29 0 319 14 0 0 0 0 0 0 0 0 0 0 0 0 11 44 0 0 0 0
talk-show 0 0 0 0 0 73 0 291 7 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0
thriller 0 0 0 0 0 93 0 144 1288 0 1 0 0 35 0 0 0 0 0 0 0 12 0 0 6 0 0
war 0 0 0 0 0 2 0 73 53 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
western 0 0 0 0 0 40 0 29 371 0 0 0 0 2 0 0 0 0 0 0 0 1 0 0 0 0 578

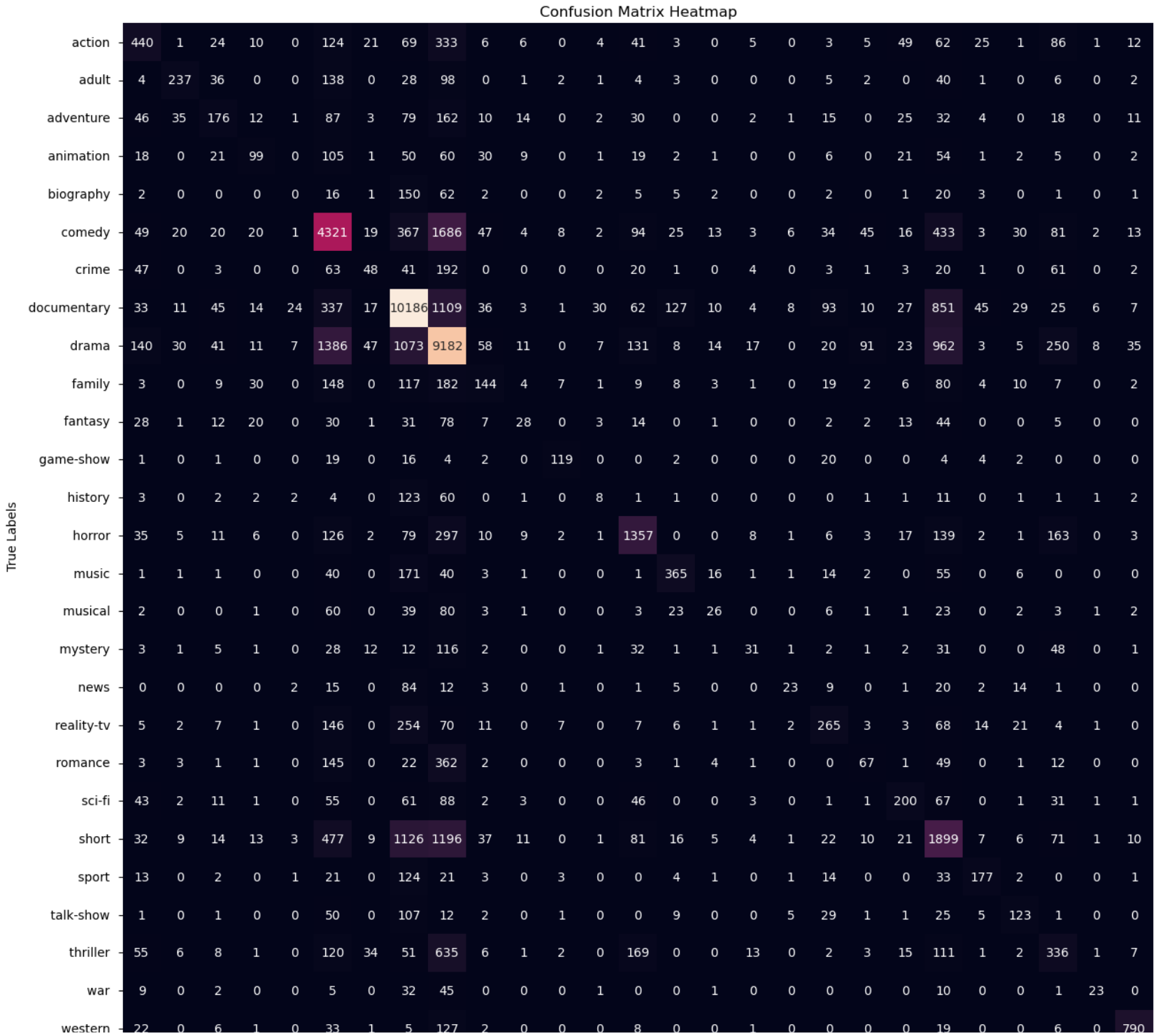

action
adult
adventure
animation
biography
comedy
crime
documentary
drama
family
fantasy
game-show
history
horror
music
musical
mystery
news
reality-tv
romance
sci-fi
short
sport
talk-show
thriller
war
western
Predicted Labels

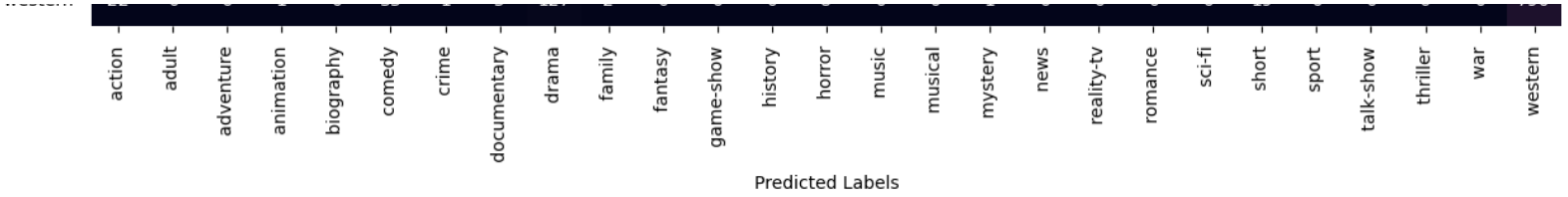
In [ ]:

LogisticRegression

In [26]:
```
## select Logistic regression for this
model = LogisticRegression()
model.fit(x_train1 ,y_train)
print("Model Score on Training data",model.score(x_train1 ,y_train))
print("Model Score on Training data",model.score(x_test1 ,y_test))
y_pred = model.predict(x_test1)
print(classification_report(y_pred ,y_test))


cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(15, 15))  # Adjust the figure size as needed
sns.heatmap(cm, annot=True, fmt='d', cbar=False,
            xticklabels=class_names, yticklabels=class_names)  # Replace 'class_names' with your class labels
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix Heatmap')
plt.show()
```

```
Model Score on Training data 0.9919383105502979
Model Score on Training data 0.5657940856346966
              precision    recall  f1-score   support

      action       0.33      0.42      0.37      1038
       adult       0.39      0.65      0.49       364
   adventure       0.23      0.38      0.29       459
   animation       0.20      0.41      0.26       244
   biography       0.00      0.00      0.00        41
      comedy       0.59      0.53      0.56      8099
       crime       0.09      0.22      0.13       216
 documentary       0.77      0.70      0.74     14497
       drama       0.68      0.56      0.61     16309
      family       0.18      0.34      0.24       428
     fantasy       0.09      0.26      0.13       107
   game-show       0.61      0.78      0.69       153
     history       0.04      0.12      0.06        65
      horror       0.59      0.63      0.61      2138
       music       0.51      0.59      0.55       615
     musical       0.09      0.26      0.14        99
     mystery       0.09      0.31      0.14        99
        news       0.12      0.46      0.19        50
  reality-tv       0.29      0.45      0.36       592
     romance       0.10      0.27      0.14       251
      sci-fi       0.32      0.45      0.38       447
       short       0.37      0.37      0.37      5162
       sport       0.42      0.59      0.49       302
   talk-show       0.33      0.47      0.39       259
    thriller       0.21      0.27      0.24      1223
         war       0.18      0.50      0.26        46
     western       0.77      0.87      0.82       904

    accuracy                           0.57     54207
   macro avg       0.32      0.44      0.36     54207
weighted avg       0.61      0.57      0.58     54207
```


Confusion Matrix Heatmap
True Labels
action 440 1 24 10 0 124 21 69 333 6 6 0 4 41 3 0 5 0 3 5 49 62 25 1 86 1 12
adult 4 237 36 0 0 138 0 28 98 0 1 2 1 4 3 0 0 0 5 2 0 40 1 0 6 0 2
adventure 46 35 176 12 1 87 3 79 162 10 14 0 2 30 0 0 2 1 15 0 25 32 4 0 18 0 11
animation 18 0 21 99 0 105 1 50 60 30 9 0 1 19 2 1 0 0 6 0 21 54 1 2 5 0 2
biography 2 0 0 0 0 16 1 150 62 2 0 0 2 5 5 2 0 0 2 0 1 20 3 0 1 0 1
comedy 49 20 20 20 1 4321 19 367 1686 47 4 8 2 94 25 13 3 6 34 45 16 433 3 30 81 2 13
crime 47 0 3 0 0 63 48 41 192 0 0 0 0 20 1 0 4 0 3 1 3 20 1 0 61 0 2
documentary 33 11 45 14 24 337 17 10186 1109 36 3 1 30 62 127 10 4 8 93 10 27 851 45 29 25 6 7
drama 140 30 41 11 7 1386 47 1073 9182 58 11 0 7 131 8 14 17 0 20 91 23 962 3 5 250 8 35
family 3 0 9 30 0 148 0 117 182 144 4 7 1 9 8 3 1 0 19 2 6 80 4 10 7 0 2
fantasy 28 1 12 20 0 30 1 31 78 7 28 0 3 14 0 1 0 0 2 2 13 44 0 0 5 0 0
game-show 1 0 1 0 0 19 0 16 4 2 0 119 0 0 2 0 0 0 20 0 0 4 4 2 0 0 0
history 3 0 2 2 2 4 0 123 60 0 1 0 8 1 1 0 0 0 0 1 1 11 0 1 1 1 2
horror 35 5 11 6 0 126 2 79 297 10 9 2 1 1357 0 0 8 1 6 3 17 139 2 1 163 0 3
music 1 1 1 0 0 40 0 171 40 3 1 0 0 1 365 16 1 1 14 2 0 55 0 6 0 0 0
musical 2 0 0 1 0 60 0 39 80 3 1 0 0 3 23 26 0 0 6 1 1 23 0 2 3 1 2
mystery 3 1 5 1 0 28 12 12 116 2 0 0 1 32 1 1 31 1 2 1 2 31 0 0 48 0 1
news 0 0 0 0 2 15 0 84 12 3 0 1 0 1 5 0 0 23 9 0 1 20 2 14 1 0 0
reality-tv 5 2 7 1 0 146 0 254 70 11 0 7 0 7 6 1 1 2 265 3 3 68 14 21 4 1 0
romance 3 3 1 1 0 145 0 22 362 2 0 0 0 3 1 4 1 0 0 67 1 49 0 1 12 0 0
sci-fi 43 2 11 1 0 55 0 61 88 2 3 0 0 46 0 0 3 0 1 1 200 67 0 1 31 1 1
short 32 9 14 13 3 477 9 1126 1196 37 11 0 1 81 16 5 4 1 22 10 21 1899 7 6 71 1 10
sport 13 0 2 0 1 21 0 124 21 3 0 3 0 0 4 1 0 1 14 0 0 33 177 2 0 0 1
talk-show 1 0 1 0 0 50 0 107 12 2 0 1 0 0 9 0 0 5 29 1 1 25 5 123 1 0 0
thriller 55 6 8 1 0 120 34 51 635 6 1 2 0 169 0 0 13 0 2 3 15 111 1 2 336 1 7
war 9 0 2 0 0 5 0 32 45 0 0 0 1 0 0 1 0 0 0 0 0 10 0 0 1 23 0
western 22 0 6 1 0 33 1 5 127 2 0 0 0 8 0 0 1 0 0 0 0 19 0 0 6 0 790

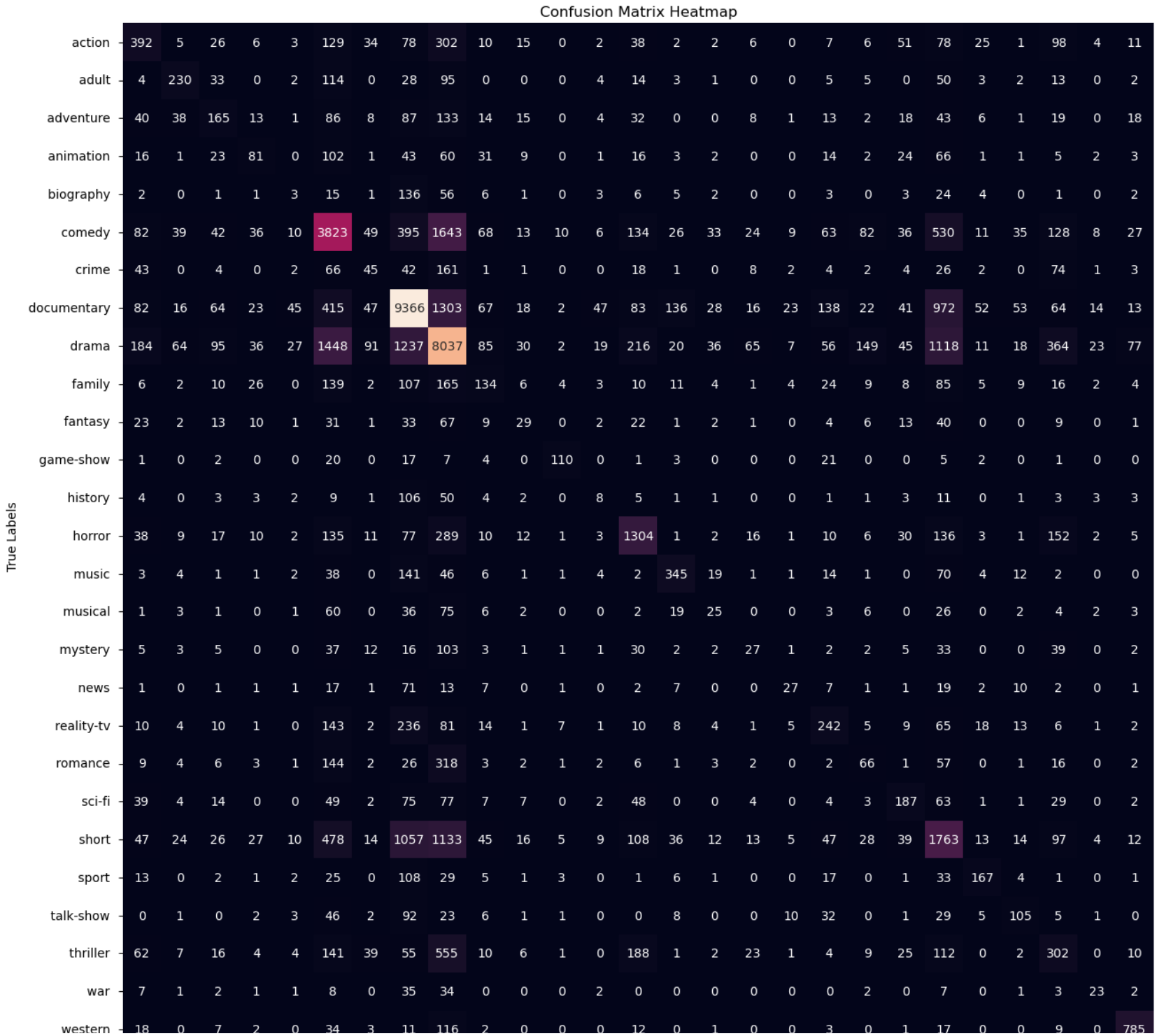

action
adult
adventure
animation
biography
comedy
crime
documentary
drama
family
fantasy
game-show
history
horror
music
musical
mystery
news
reality-tv
romance
sci-fi
short
sport
talk-show
thriller
war
western
Predicted Labels

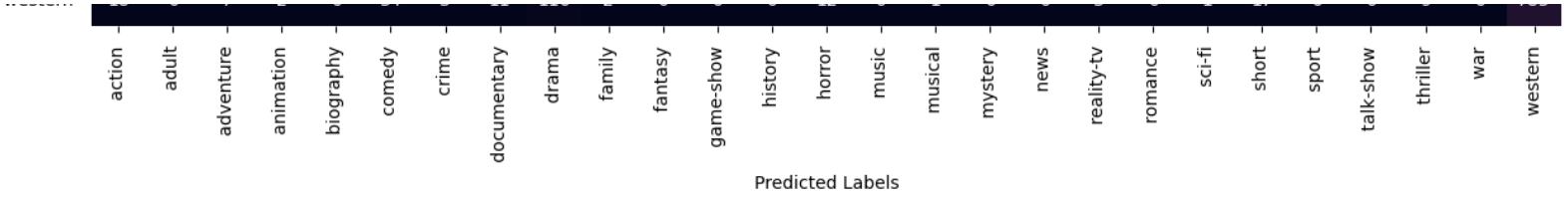
Support Vector Machine (SVC)

In [27]:
```
## Select SVC model

svm = LinearSVC()
svm.fit(x_train1 ,y_train)
print("Model Score on Training data",svm.score(x_train1 ,y_train))
print("Model Score on Training data",svm.score(x_test1 ,y_test))
y_pred = svm.predict(x_test1)
print(classification_report(y_pred ,y_test))
## As we can see from accuracy that the model the not performing well



cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(15, 15))  # Adjust the figure size as needed
sns.heatmap(cm, annot=True, fmt='d', cbar=False,
            xticklabels=class_names, yticklabels=class_names)  # Replace 'class_names' with your class labels
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix Heatmap')
plt.show()
```

```
Model Score on Training data 0.9994096703377793
Model Score on Training data 0.5126828638367739
```

	precision	recall	f1-score	support
action	0.29	0.35	0.32	1132
adult	0.38	0.50	0.43	461
adventure	0.22	0.28	0.24	589
animation	0.16	0.28	0.20	288
biography	0.01	0.02	0.02	123
comedy	0.52	0.49	0.51	7752
crime	0.09	0.12	0.10	368
documentary	0.71	0.68	0.70	13711
drama	0.59	0.54	0.56	14971
family	0.17	0.24	0.20	557
fantasy	0.09	0.15	0.11	189
game-show	0.57	0.73	0.64	150
history	0.04	0.07	0.05	123
horror	0.57	0.56	0.57	2308
music	0.48	0.53	0.51	646
musical	0.09	0.14	0.11	182
mystery	0.08	0.12	0.10	216
news	0.14	0.28	0.19	97
reality-tv	0.27	0.33	0.30	740
romance	0.10	0.16	0.12	415
sci-fi	0.30	0.34	0.32	546
short	0.35	0.32	0.33	5478
sport	0.40	0.50	0.44	335
talk-show	0.28	0.37	0.32	287
thriller	0.19	0.21	0.20	1462
war	0.18	0.26	0.21	90
western	0.77	0.79	0.78	991
accuracy			0.51	54207
macro avg	0.30	0.35	0.32	54207
weighted avg	0.53	0.51	0.52	54207


Confusion Matrix Heatmap
True Labels
action 392 5 26 6 3 129 34 78 302 10 15 0 2 38 2 2 6 0 7 6 51 78 25 1 98 4 11
adult 4 230 33 0 2 114 0 28 95 0 0 0 4 14 3 1 0 0 5 5 0 50 3 2 13 0 2
adventure 40 38 165 13 1 86 8 87 133 14 15 0 4 32 0 0 8 1 13 2 18 43 6 1 19 0 18
animation 16 1 23 81 0 102 1 43 60 31 9 0 1 16 3 2 0 0 14 2 24 66 1 1 5 2 3
biography 2 0 1 1 3 15 1 136 56 6 1 0 3 6 5 2 0 0 3 0 3 24 4 0 1 0 2
comedy 82 39 42 36 10 3823 49 395 1643 68 13 10 6 134 26 33 24 9 63 82 36 530 11 35 128 8 27
crime 43 0 4 0 2 66 45 42 161 1 1 0 0 18 1 0 8 2 4 2 4 26 2 0 74 1 3
documentary 82 16 64 23 45 415 47 9366 1303 67 18 2 47 83 136 28 16 23 138 22 41 972 52 53 64 14 13
drama 184 64 95 36 27 1448 91 1237 8037 85 30 2 19 216 20 36 65 7 56 149 45 1118 11 18 364 23 77
family 6 2 10 26 0 139 2 107 165 134 6 4 3 10 11 4 1 4 24 9 8 85 5 9 16 2 4
fantasy 23 2 13 10 1 31 1 33 67 9 29 0 2 22 1 2 1 0 4 6 13 40 0 0 9 0 1
game-show 1 0 2 0 0 20 0 17 7 4 0 110 0 1 3 0 0 0 21 0 0 5 2 0 1 0 0
history 4 0 3 3 2 9 1 106 50 4 2 0 8 5 1 1 0 0 1 1 3 11 0 1 3 3 3
horror 38 9 17 10 2 135 11 77 289 10 12 1 3 1304 1 2 16 1 10 6 30 136 3 1 152 2 5
music 3 4 1 1 2 38 0 141 46 6 1 1 4 2 345 19 1 1 14 1 0 70 4 12 2 0 0
musical 1 3 1 0 1 60 0 36 75 6 2 0 0 2 19 25 0 0 3 6 0 26 0 2 4 2 3
mystery 5 3 5 0 0 37 12 16 103 3 1 1 1 30 2 2 27 1 2 2 5 33 0 0 39 0 2
news 1 0 1 1 1 17 1 71 13 7 0 1 0 2 7 0 0 27 7 1 1 19 2 10 2 0 1
reality-tv 10 4 10 1 0 143 2 236 81 14 1 7 1 10 8 4 1 5 242 5 9 65 18 13 6 1 2
romance 9 4 6 3 1 144 2 26 318 3 2 1 2 6 1 3 2 0 2 66 1 57 0 1 16 0 2
sci-fi 39 4 14 0 0 49 2 75 77 7 7 0 2 48 0 0 4 0 4 3 187 63 1 1 29 0 2
short 47 24 26 27 10 478 14 1057 1133 45 16 5 9 108 36 12 13 5 47 28 39 1763 13 14 97 4 12
sport 13 0 2 1 2 25 0 108 29 5 1 3 0 1 6 1 0 0 17 0 1 33 167 4 1 0 1
talk-show 0 1 0 2 3 46 2 92 23 6 1 1 0 0 8 0 0 10 32 0 1 29 5 105 5 1 0
thriller 62 7 16 4 4 141 39 55 555 10 6 1 0 188 1 2 23 1 4 9 25 112 0 2 302 0 10
war 7 1 2 1 1 8 0 35 34 0 0 0 2 0 0 0 0 0 0 2 0 7 0 1 3 23 2
western 18 0 7 2 0 34 3 11 116 2 0 0 0 12 0 1 0 0 3 0 1 17 0 0 9 0 785

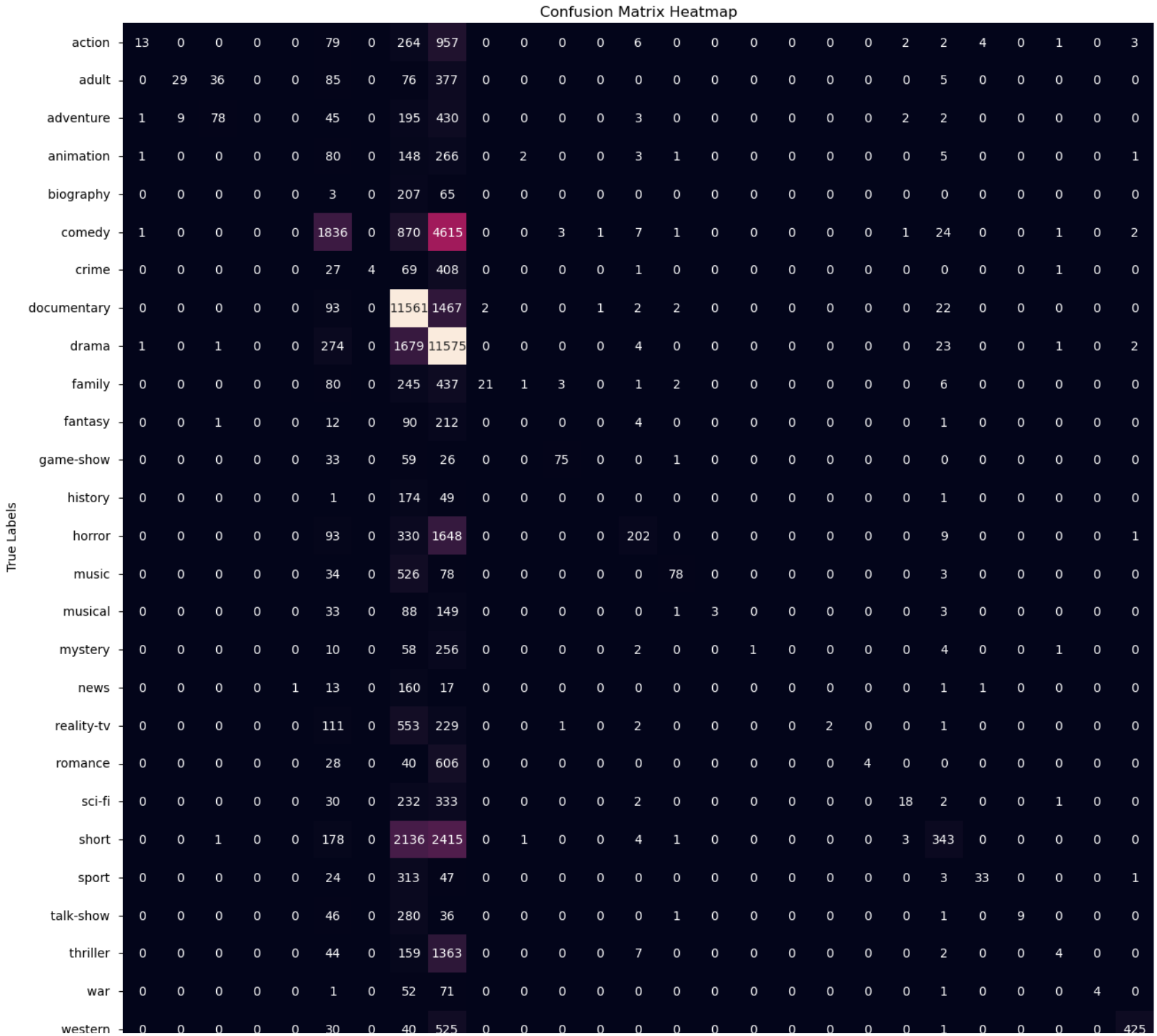

action
adult
adventure
animation
biography
comedy
crime
documentary
drama
family
fantasy
game-show
history
horror
music
musical
mystery
news
reality-tv
romance
sci-fi
short
sport
talk-show
thriller
war
western
Predicted Labels

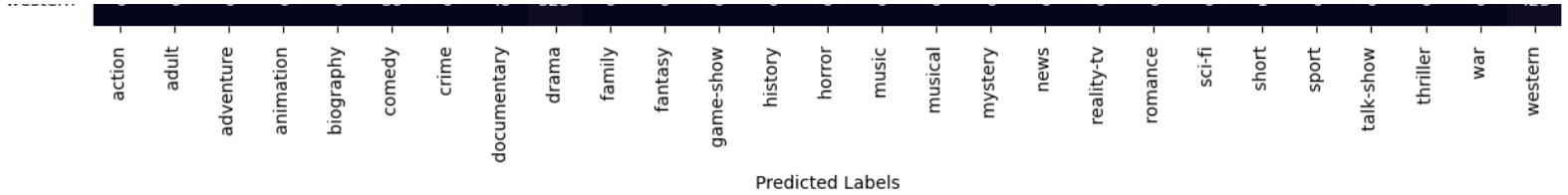
RandomForestClassifier

In [28]:
```
# Create a Random Forest model
random_forest = RandomForestClassifier()

# Fit the model with GridSearchCV
random_forest.fit(x_train1, y_train)
print("Random Forest - Train Score:",random_forest.score(x_train1, y_train))
print("Random Forest - Test Score:", random_forest.score(x_test1, y_test))

y_pred = random_forest.predict(x_test1)
print(classification_report(y_pred ,y_test))



cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(15, 15))  # Adjust the figure size as needed
sns.heatmap(cm, annot=True, fmt='d', cbar=False,
            xticklabels=class_names, yticklabels=class_names)  # Replace 'class_names' with your class labels
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix Heatmap')
plt.show()
```

```
Random Forest - Train Score: 0.9994834615455569
Random Forest - Test Score: 0.4855092515726751
```

	precision	recall	f1-score	support
action	0.01	0.76	0.02	17
adult	0.05	0.76	0.09	38
adventure	0.10	0.67	0.18	117
animation	0.00	0.00	0.00	0
biography	0.00	0.00	0.00	1
comedy	0.25	0.55	0.34	3323
crime	0.01	1.00	0.02	4
documentary	0.88	0.56	0.69	20604
drama	0.85	0.40	0.55	28657
family	0.03	0.91	0.05	23
fantasy	0.00	0.00	0.00	4
game-show	0.39	0.91	0.54	82
history	0.00	0.00	0.00	2
horror	0.09	0.81	0.16	250
music	0.11	0.89	0.19	88
musical	0.01	1.00	0.02	3
mystery	0.00	1.00	0.01	1
news	0.00	0.00	0.00	0
reality-tv	0.00	1.00	0.00	2
romance	0.01	1.00	0.01	4
sci-fi	0.03	0.69	0.06	26
short	0.07	0.74	0.12	465
sport	0.08	0.87	0.14	38
talk-show	0.02	1.00	0.05	9
thriller	0.00	0.40	0.01	10
war	0.03	1.00	0.06	4
western	0.42	0.98	0.58	435
accuracy			0.49	54207
macro avg	0.13	0.66	0.14	54207
weighted avg	0.81	0.49	0.58	54207


Confusion Matrix Heatmap
True Labels
action 13 0 0 0 0 79 0 264 957 0 0 0 0 6 0 0 0 0 0 0 2 2 4 0 1 0 3
adult 0 29 36 0 0 85 0 76 377 0 0 0 0 0 0 0 0 0 0 0 0 5 0 0 0 0 0
adventure 1 9 78 0 0 45 0 195 430 0 0 0 0 3 0 0 0 0 0 0 2 2 0 0 0 0 0
animation 1 0 0 0 0 80 0 148 266 0 2 0 0 3 1 0 0 0 0 0 0 5 0 0 0 0 1
biography 0 0 0 0 0 3 0 207 65 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
comedy 1 0 0 0 0 1836 0 870 4615 0 0 3 1 7 1 0 0 0 0 0 1 24 0 0 1 0 2
crime 0 0 0 0 0 27 4 69 408 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0
documentary 0 0 0 0 0 93 0 11561 1467 2 0 0 1 2 2 0 0 0 0 0 0 22 0 0 0 0 0
drama 1 0 1 0 0 274 0 1679 11575 0 0 0 0 4 0 0 0 0 0 0 0 23 0 0 1 0 2
family 0 0 0 0 0 80 0 245 437 21 1 3 0 1 2 0 0 0 0 0 0 6 0 0 0 0 0
fantasy 0 0 1 0 0 12 0 90 212 0 0 0 0 4 0 0 0 0 0 0 0 1 0 0 0 0 0
game-show 0 0 0 0 0 33 0 59 26 0 0 75 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0
history 0 0 0 0 0 1 0 174 49 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
horror 0 0 0 0 0 93 0 330 1648 0 0 0 0 202 0 0 0 0 0 0 0 9 0 0 0 0 1
music 0 0 0 0 0 34 0 526 78 0 0 0 0 0 78 0 0 0 0 0 0 3 0 0 0 0 0
musical 0 0 0 0 0 33 0 88 149 0 0 0 0 0 1 3 0 0 0 0 0 3 0 0 0 0 0
mystery 0 0 0 0 0 10 0 58 256 0 0 0 0 2 0 0 1 0 0 0 0 4 0 0 1 0 0
news 0 0 0 0 1 13 0 160 17 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0
reality-tv 0 0 0 0 0 111 0 553 229 0 0 1 0 2 0 0 0 0 2 0 0 1 0 0 0 0 0
romance 0 0 0 0 0 28 0 40 606 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0
sci-fi 0 0 0 0 0 30 0 232 333 0 0 0 0 2 0 0 0 0 0 0 18 2 0 0 1 0 0
short 0 0 1 0 0 178 0 2136 2415 0 1 0 0 4 1 0 0 0 0 0 3 343 0 0 0 0 0
sport 0 0 0 0 0 24 0 313 47 0 0 0 0 0 0 0 0 0 0 0 0 3 33 0 0 0 1
talk-show 0 0 0 0 0 46 0 280 36 0 0 0 0 0 1 0 0 0 0 0 0 1 0 9 0 0 0
thriller 0 0 0 0 0 44 0 159 1363 0 0 0 0 7 0 0 0 0 0 0 0 2 0 0 4 0 0
war 0 0 0 0 0 1 0 52 71 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 4 0
western 0 0 0 0 0 30 0 40 525 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 425



In [ ]:

Now use TfidfVectorizer technique

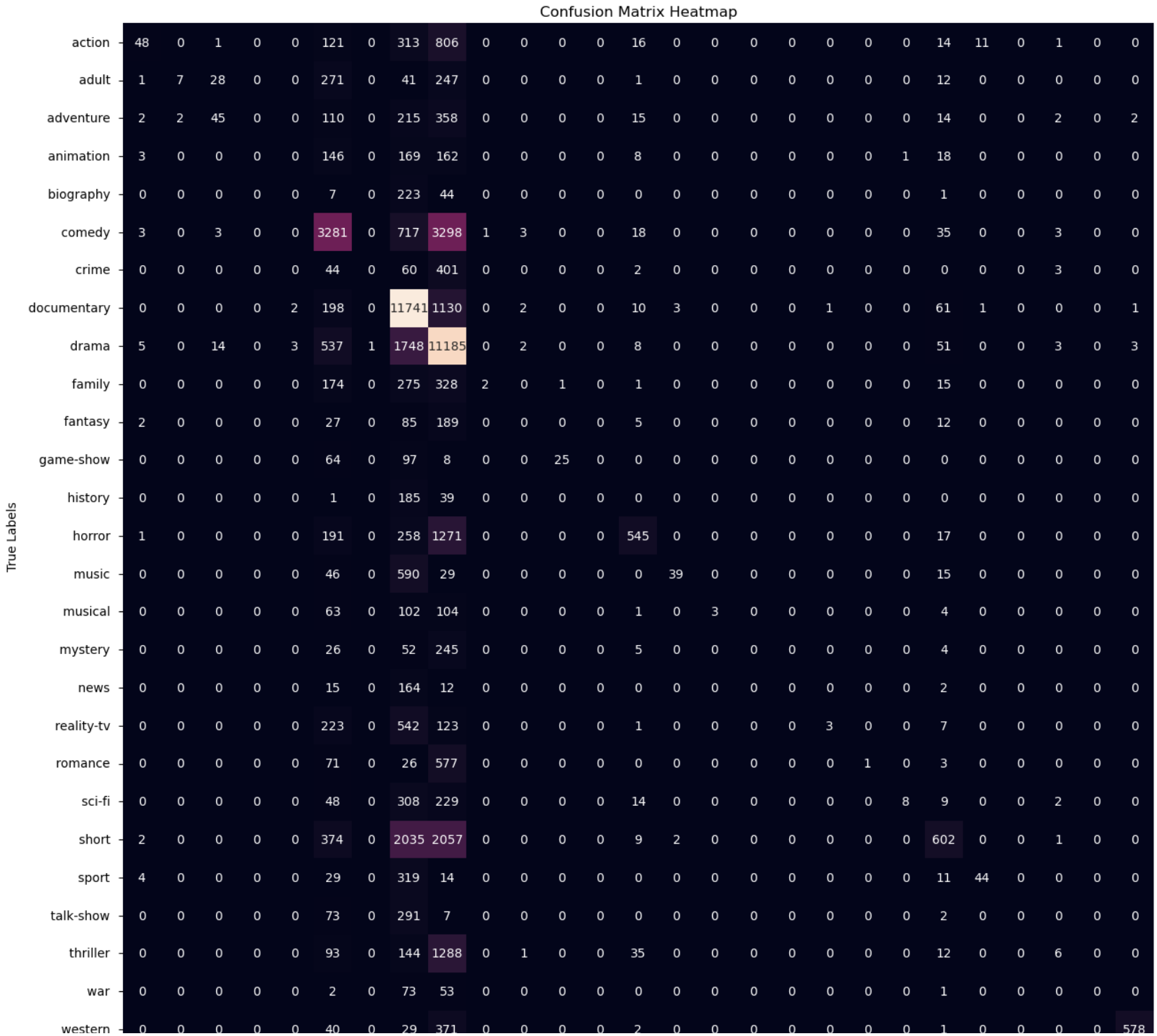
In [29]:
```
## Using TfidfVectorizer technique
vectorizer = TfidfVectorizer()
x_train2 = vectorize.fit_transform(x_train)
x_test2 =  vectorize.transform(x_test)
```

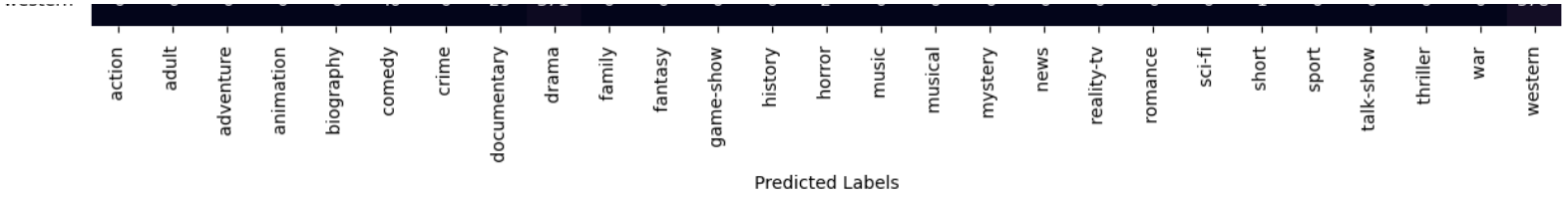
MultinomialNB with TfidfVectorizer

In [30]:
```
mnb = MultinomialNB()
mnb.fit(x_train2 ,y_train)
print("Model Score on Training data",mnb.score(x_train2 ,y_train))
print("Model Score on Training data",mnb.score(x_test2 ,y_test))
y_pred = mnb.predict(x_test2)
print(classification_report(y_pred ,y_test))


cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(15, 15))  # Adjust the figure size as needed
sns.heatmap(cm, annot=True, fmt='d', cbar=False,
            xticklabels=class_names, yticklabels=class_names)  # Replace 'class_names' with your class labels
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix Heatmap')
plt.show()
```

```
Model Score on Training data 0.6284797166417622
Model Score on Training data 0.51954544616009
```

	precision	recall	f1-score	support
action	0.04	0.68	0.07	71
adult	0.01	0.78	0.02	9
adventure	0.06	0.49	0.11	91
animation	0.00	0.00	0.00	0
biography	0.00	0.00	0.00	5
comedy	0.45	0.52	0.48	6275
crime	0.00	0.00	0.00	1
documentary	0.89	0.56	0.69	20802
drama	0.82	0.46	0.59	24575
family	0.00	0.67	0.01	3
fantasy	0.00	0.00	0.00	8
game-show	0.13	0.96	0.23	26
history	0.00	0.00	0.00	0
horror	0.24	0.78	0.37	696
music	0.05	0.89	0.10	44
musical	0.01	1.00	0.02	3
mystery	0.00	0.00	0.00	0
news	0.00	0.00	0.00	0
reality-tv	0.00	0.75	0.01	4
romance	0.00	1.00	0.00	1
sci-fi	0.01	0.89	0.03	9
short	0.12	0.65	0.20	923
sport	0.10	0.79	0.18	56
talk-show	0.00	0.00	0.00	0
thriller	0.00	0.29	0.01	21
war	0.00	0.00	0.00	0
western	0.57	0.99	0.72	584
accuracy			0.52	54207
macro avg	0.13	0.49	0.14	54207
weighted avg	0.78	0.52	0.60	54207

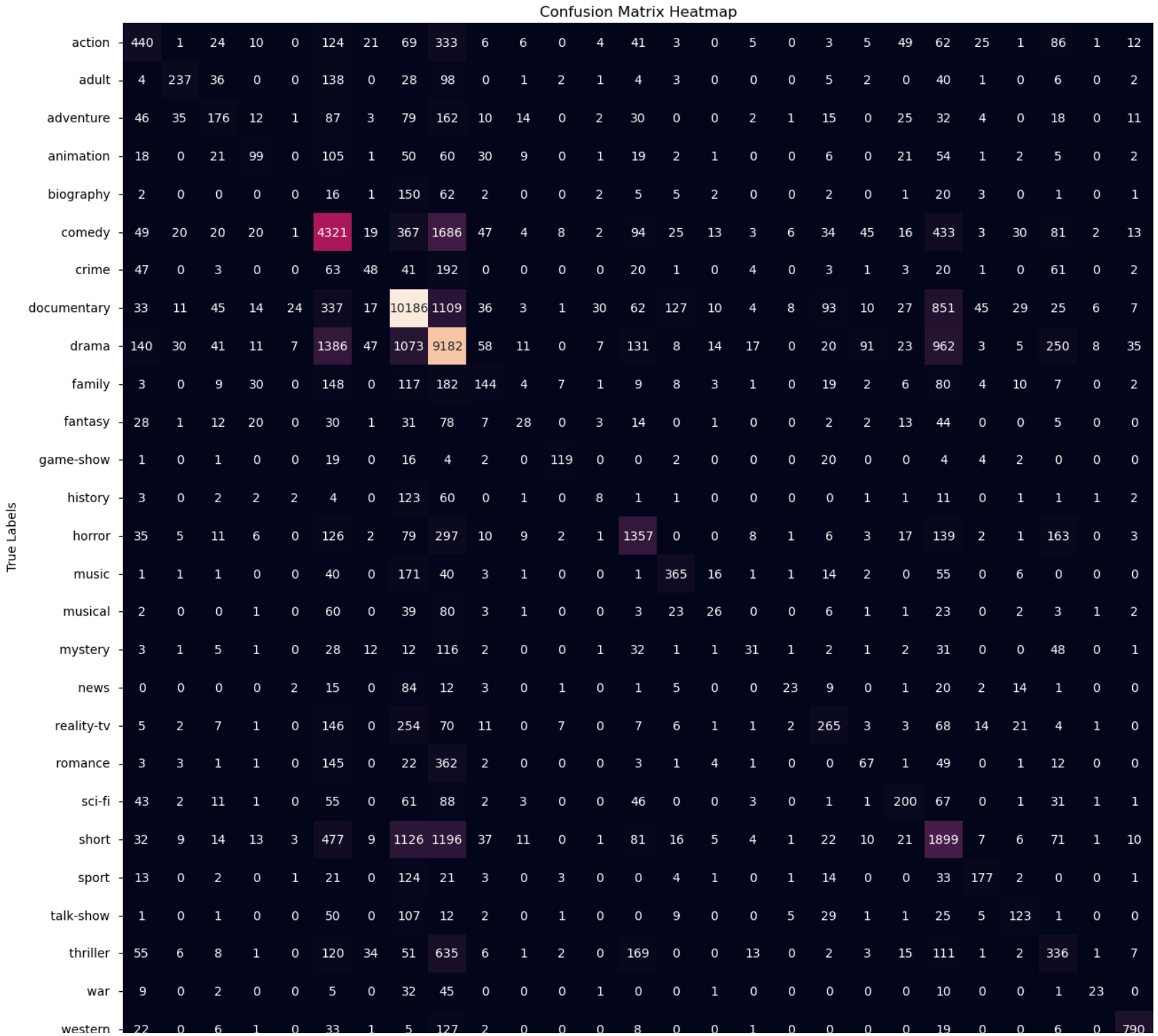

Confusion Matrix Heatmap
True Labels
action 48 0 1 0 0 121 0 313 806 0 0 0 0 16 0 0 0 0 0 0 0 14 11 0 1 0 0
adult 1 7 28 0 0 271 0 41 247 0 0 0 0 1 0 0 0 0 0 0 0 12 0 0 0 0 0
adventure 2 2 45 0 0 110 0 215 358 0 0 0 0 15 0 0 0 0 0 0 0 14 0 0 2 0 2
animation 3 0 0 0 0 146 0 169 162 0 0 0 0 8 0 0 0 0 0 0 1 18 0 0 0 0 0
biography 0 0 0 0 0 7 0 223 44 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
comedy 3 0 3 0 0 3281 0 717 3298 1 3 0 0 18 0 0 0 0 0 0 0 35 0 0 3 0 0
crime 0 0 0 0 0 44 0 60 401 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 3 0 0
documentary 0 0 0 0 2 198 0 11741 1130 0 2 0 0 10 3 0 0 0 1 0 0 61 1 0 0 0 1
drama 5 0 14 0 3 537 1 1748 11185 0 2 0 0 8 0 0 0 0 0 0 0 51 0 0 3 0 3
family 0 0 0 0 0 174 0 275 328 2 0 1 0 1 0 0 0 0 0 0 0 15 0 0 0 0 0
fantasy 2 0 0 0 0 27 0 85 189 0 0 0 0 5 0 0 0 0 0 0 0 12 0 0 0 0 0
game-show 0 0 0 0 0 64 0 97 8 0 0 25 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
history 0 0 0 0 0 1 0 185 39 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
horror 1 0 0 0 0 191 0 258 1271 0 0 0 0 545 0 0 0 0 0 0 0 17 0 0 0 0 0
music 0 0 0 0 0 46 0 590 29 0 0 0 0 0 39 0 0 0 0 0 0 15 0 0 0 0 0
musical 0 0 0 0 0 63 0 102 104 0 0 0 0 1 0 3 0 0 0 0 0 4 0 0 0 0 0
mystery 0 0 0 0 0 26 0 52 245 0 0 0 0 5 0 0 0 0 0 0 0 4 0 0 0 0 0
news 0 0 0 0 0 15 0 164 12 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0
reality-tv 0 0 0 0 0 223 0 542 123 0 0 0 0 1 0 0 0 0 3 0 0 7 0 0 0 0 0
romance 0 0 0 0 0 71 0 26 577 0 0 0 0 0 0 0 0 0 0 1 0 3 0 0 0 0 0
sci-fi 0 0 0 0 0 48 0 308 229 0 0 0 0 14 0 0 0 0 0 0 8 9 0 0 2 0 0
short 2 0 0 0 0 374 0 2035 2057 0 0 0 0 9 2 0 0 0 0 0 0 602 0 0 1 0 0
sport 4 0 0 0 0 29 0 319 14 0 0 0 0 0 0 0 0 0 0 0 0 11 44 0 0 0 0
talk-show 0 0 0 0 0 73 0 291 7 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 0 0
thriller 0 0 0 0 0 93 0 144 1288 0 1 0 0 35 0 0 0 0 0 0 0 12 0 0 6 0 0
war 0 0 0 0 0 2 0 73 53 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
western 0 0 0 0 0 40 0 29 371 0 0 0 0 2 0 0 0 0 0 0 0 1 0 0 0 0 578

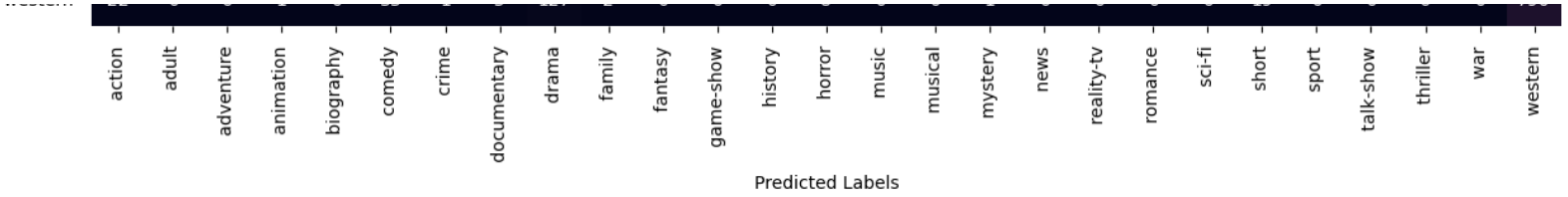

action
adult
adventure
animation
biography
comedy
crime
documentary
drama
family
fantasy
game-show
history
horror
music
musical
mystery
news
reality-tv
romance
sci-fi
short
sport
talk-show
thriller
war
western
Predicted Labels

LogisticRegression with TfidfVectorizer

In [31]:
```
## select Logistic regression for this
model = LogisticRegression()
model.fit(x_train2 ,y_train)
print("Model Score on Training data",model.score(x_train2 ,y_train))
print("Model Score on Training data",model.score(x_test2 ,y_test))
y_pred = model.predict(x_test2)
print(classification_report(y_pred ,y_test))


cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(15, 15))  # Adjust the figure size as needed
sns.heatmap(cm, annot=True, fmt='d', cbar=False,
            xticklabels=class_names, yticklabels=class_names)  # Replace 'class_names' with your class labels
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix Heatmap')
plt.show()
```

```
Model Score on Training data 0.9919383105502979
Model Score on Training data 0.5657940856346966
              precision    recall  f1-score   support

      action       0.33      0.42      0.37      1038
       adult       0.39      0.65      0.49       364
   adventure       0.23      0.38      0.29       459
   animation       0.20      0.41      0.26       244
   biography       0.00      0.00      0.00        41
      comedy       0.59      0.53      0.56      8099
       crime       0.09      0.22      0.13       216
 documentary       0.77      0.70      0.74     14497
       drama       0.68      0.56      0.61     16309
      family       0.18      0.34      0.24       428
     fantasy       0.09      0.26      0.13       107
   game-show       0.61      0.78      0.69       153
     history       0.04      0.12      0.06        65
      horror       0.59      0.63      0.61      2138
       music       0.51      0.59      0.55       615
     musical       0.09      0.26      0.14        99
     mystery       0.09      0.31      0.14        99
        news       0.12      0.46      0.19        50
  reality-tv       0.29      0.45      0.36       592
     romance       0.10      0.27      0.14       251
      sci-fi       0.32      0.45      0.38       447
       short       0.37      0.37      0.37      5162
       sport       0.42      0.59      0.49       302
   talk-show       0.33      0.47      0.39       259
    thriller       0.21      0.27      0.24      1223
         war       0.18      0.50      0.26        46
     western       0.77      0.87      0.82       904

    accuracy                           0.57     54207
   macro avg       0.32      0.44      0.36     54207
weighted avg       0.61      0.57      0.58     54207
```

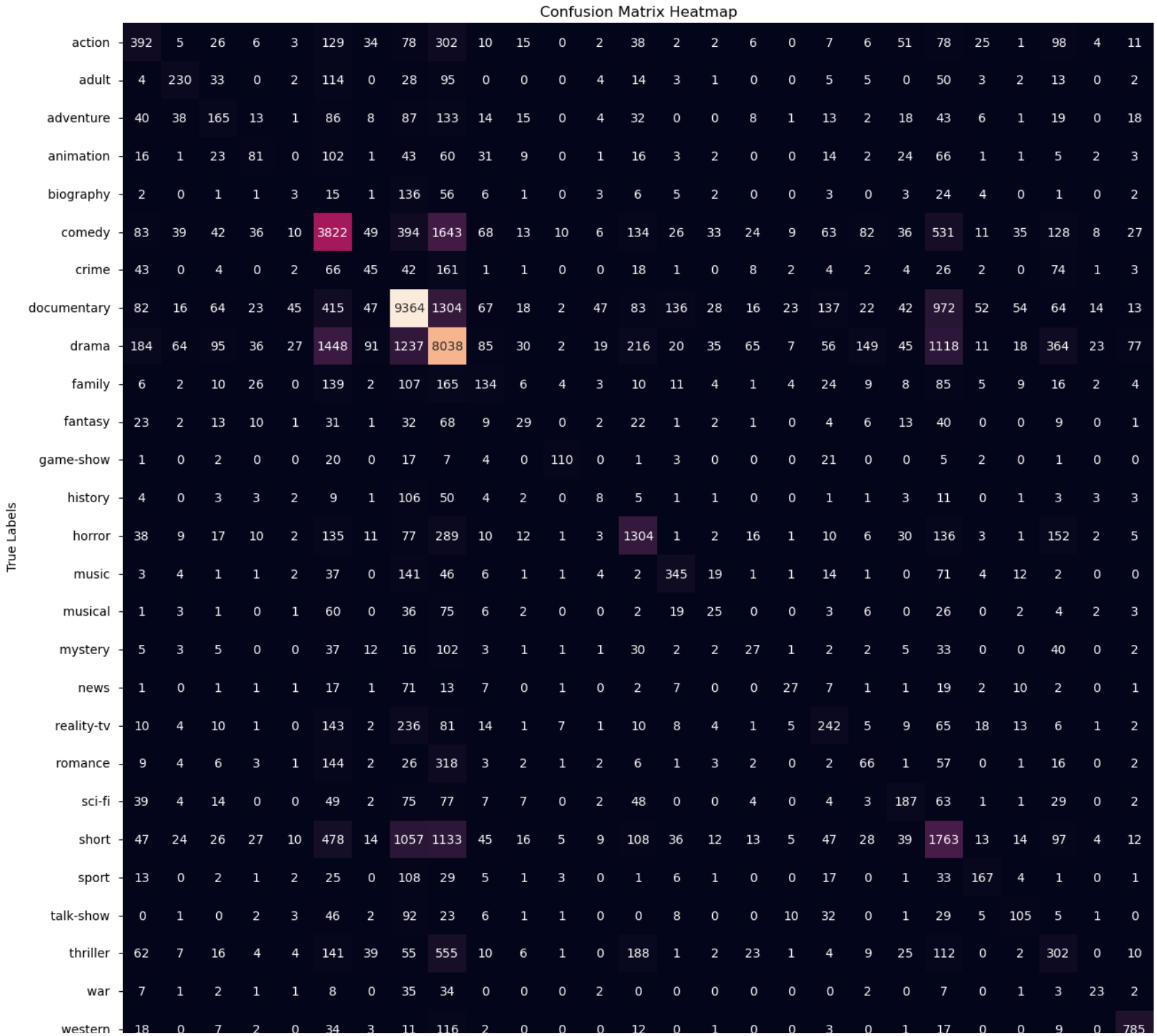

Confusion Matrix Heatmap
True Labels
action 440 1 24 10 0 124 21 69 333 6 6 0 4 41 3 0 5 0 3 5 49 62 25 1 86 1 12
adult 4 237 36 0 0 138 0 28 98 0 1 2 1 4 3 0 0 0 5 2 0 40 1 0 6 0 2
adventure 46 35 176 12 1 87 3 79 162 10 14 0 2 30 0 0 2 1 15 0 25 32 4 0 18 0 11
animation 18 0 21 99 0 105 1 50 60 30 9 0 1 19 2 1 0 0 6 0 21 54 1 2 5 0 2
biography 2 0 0 0 0 16 1 150 62 2 0 0 2 5 5 2 0 0 2 0 1 20 3 0 1 0 1
comedy 49 20 20 20 1 4321 19 367 1686 47 4 8 2 94 25 13 3 6 34 45 16 433 3 30 81 2 13
crime 47 0 3 0 0 63 48 41 192 0 0 0 0 20 1 0 4 0 3 1 3 20 1 0 61 0 2
documentary 33 11 45 14 24 337 17 10186 1109 36 3 1 30 62 127 10 4 8 93 10 27 851 45 29 25 6 7
drama 140 30 41 11 7 1386 47 1073 9182 58 11 0 7 131 8 14 17 0 20 91 23 962 3 5 250 8 35
family 3 0 9 30 0 148 0 117 182 144 4 7 1 9 8 3 1 0 19 2 6 80 4 10 7 0 2
fantasy 28 1 12 20 0 30 1 31 78 7 28 0 3 14 0 1 0 0 2 2 13 44 0 0 5 0 0
game-show 1 0 1 0 0 19 0 16 4 2 0 119 0 0 2 0 0 0 20 0 0 4 4 2 0 0 0
history 3 0 2 2 2 4 0 123 60 0 1 0 8 1 1 0 0 0 0 1 1 11 0 1 1 1 2
horror 35 5 11 6 0 126 2 79 297 10 9 2 1 1357 0 0 8 1 6 3 17 139 2 1 163 0 3
music 1 1 1 0 0 40 0 171 40 3 1 0 0 1 365 16 1 1 14 2 0 55 0 6 0 0 0
musical 2 0 0 1 0 60 0 39 80 3 1 0 0 3 23 26 0 0 6 1 1 23 0 2 3 1 2
mystery 3 1 5 1 0 28 12 12 116 2 0 0 1 32 1 1 31 1 2 1 2 31 0 0 48 0 1
news 0 0 0 0 2 15 0 84 12 3 0 1 0 1 5 0 0 23 9 0 1 20 2 14 1 0 0
reality-tv 5 2 7 1 0 146 0 254 70 11 0 7 0 7 6 1 1 2 265 3 3 68 14 21 4 1 0
romance 3 3 1 1 0 145 0 22 362 2 0 0 0 3 1 4 1 0 0 67 1 49 0 1 12 0 0
sci-fi 43 2 11 1 0 55 0 61 88 2 3 0 0 46 0 0 3 0 1 1 200 67 0 1 31 1 1
short 32 9 14 13 3 477 9 1126 1196 37 11 0 1 81 16 5 4 1 22 10 21 1899 7 6 71 1 10
sport 13 0 2 0 1 21 0 124 21 3 0 3 0 0 4 1 0 1 14 0 0 33 177 2 0 0 1
talk-show 1 0 1 0 0 50 0 107 12 2 0 1 0 0 9 0 0 5 29 1 1 25 5 123 1 0 0
thriller 55 6 8 1 0 120 34 51 635 6 1 2 0 169 0 0 13 0 2 3 15 111 1 2 336 1 7
war 9 0 2 0 0 5 0 32 45 0 0 0 1 0 0 1 0 0 0 0 0 10 0 0 1 23 0
western 22 0 6 1 0 33 1 5 127 2 0 0 0 8 0 0 1 0 0 0 0 19 0 0 6 0 790

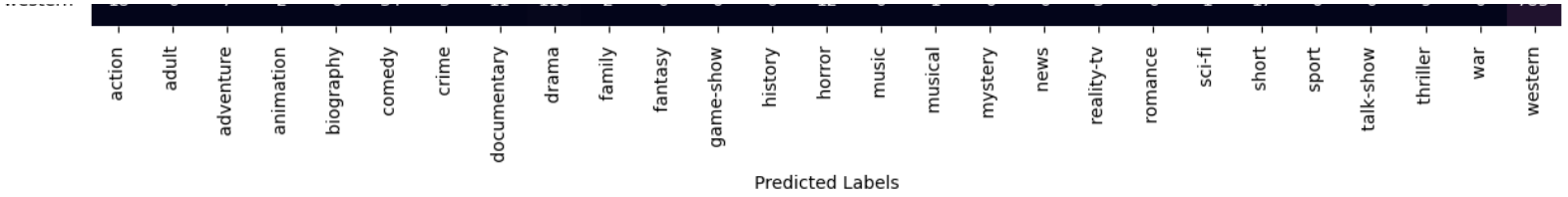

action
adult
adventure
animation
biography
comedy
crime
documentary
drama
family
fantasy
game-show
history
horror
music
musical
mystery
news
reality-tv
romance
sci-fi
short
sport
talk-show
thriller
war
western
Predicted Labels

Support Vector Machine (SVC) with TfidfVectorizer

In [32]:
```
## Select SVC model

svm = LinearSVC()
svm.fit(x_train2 ,y_train)
print("Model Score on Training data",svm.score(x_train2 ,y_train))
print("Model Score on Training data",svm.score(x_test2 ,y_test))
y_pred = svm.predict(x_test2)
print(classification_report(y_pred ,y_test))
## As we can see from accuracy that the model the not performing well



cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(15, 15))  # Adjust the figure size as needed
sns.heatmap(cm, annot=True, fmt='d', cbar=False,
            xticklabels=class_names, yticklabels=class_names)  # Replace 'class_names' with your class labels
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix Heatmap')
plt.show()
```

```
Model Score on Training data 0.9994096703377793
Model Score on Training data 0.512645968232885
              precision    recall  f1-score   support

      action       0.29      0.35      0.32      1133
       adult       0.38      0.50      0.43       461
   adventure       0.22      0.28      0.24       589
   animation       0.16      0.28      0.20       288
   biography       0.01      0.02      0.02       123
      comedy       0.52      0.49      0.51      7750
       crime       0.09      0.12      0.10       368
 documentary       0.71      0.68      0.70     13707
       drama       0.59      0.54      0.56     14973
      family       0.17      0.24      0.20       557
     fantasy       0.09      0.15      0.11       189
   game-show       0.57      0.73      0.64       150
     history       0.04      0.07      0.05       123
      horror       0.57      0.56      0.57      2308
       music       0.48      0.53      0.51       646
     musical       0.09      0.14      0.11       181
     mystery       0.08      0.12      0.10       216
        news       0.14      0.28      0.19        97
  reality-tv       0.27      0.33      0.30       739
     romance       0.10      0.16      0.12       415
      sci-fi       0.30      0.34      0.32       547
       short       0.35      0.32      0.33      5480
       sport       0.40      0.50      0.44       335
   talk-show       0.28      0.36      0.32       288
    thriller       0.19      0.21      0.20      1463
         war       0.18      0.26      0.21        90
     western       0.77      0.79      0.78       991

    accuracy                           0.51     54207
   macro avg       0.30      0.35      0.32     54207
weighted avg       0.53      0.51      0.52     54207
```


Confusion Matrix Heatmap
True Labels
action 392 5 26 6 3 129 34 78 302 10 15 0 2 38 2 2 6 0 7 6 51 78 25 1 98 4 11
adult 4 230 33 0 2 114 0 28 95 0 0 0 4 14 3 1 0 0 5 5 0 50 3 2 13 0 2
adventure 40 38 165 13 1 86 8 87 133 14 15 0 4 32 0 0 8 1 13 2 18 43 6 1 19 0 18
animation 16 1 23 81 0 102 1 43 60 31 9 0 1 16 3 2 0 0 14 2 24 66 1 1 5 2 3
biography 2 0 1 1 3 15 1 136 56 6 1 0 3 6 5 2 0 0 3 0 3 24 4 0 1 0 2
comedy 83 39 42 36 10 3822 49 394 1643 68 13 10 6 134 26 33 24 9 63 82 36 531 11 35 128 8 27
crime 43 0 4 0 2 66 45 42 161 1 1 0 0 18 1 0 8 2 4 2 4 26 2 0 74 1 3
documentary 82 16 64 23 45 415 47 9364 1304 67 18 2 47 83 136 28 16 23 137 22 42 972 52 54 64 14 13
drama 184 64 95 36 27 1448 91 1237 8038 85 30 2 19 216 20 35 65 7 56 149 45 1118 11 18 364 23 77
family 6 2 10 26 0 139 2 107 165 134 6 4 3 10 11 4 1 4 24 9 8 85 5 9 16 2 4
fantasy 23 2 13 10 1 31 1 32 68 9 29 0 2 22 1 2 1 0 4 6 13 40 0 0 9 0 1
game-show 1 0 2 0 0 20 0 17 7 4 0 110 0 1 3 0 0 0 21 0 0 5 2 0 1 0 0
history 4 0 3 3 2 9 1 106 50 4 2 0 8 5 1 1 0 0 1 1 3 11 0 1 3 3 3
horror 38 9 17 10 2 135 11 77 289 10 12 1 3 1304 1 2 16 1 10 6 30 136 3 1 152 2 5
music 3 4 1 1 2 37 0 141 46 6 1 1 4 2 345 19 1 1 14 1 0 71 4 12 2 0 0
musical 1 3 1 0 1 60 0 36 75 6 2 0 0 2 19 25 0 0 3 6 0 26 0 2 4 2 3
mystery 5 3 5 0 0 37 12 16 102 3 1 1 1 30 2 2 27 1 2 2 5 33 0 0 40 0 2
news 1 0 1 1 1 17 1 71 13 7 0 1 0 2 7 0 0 27 7 1 1 19 2 10 2 0 1
reality-tv 10 4 10 1 0 143 2 236 81 14 1 7 1 10 8 4 1 5 242 5 9 65 18 13 6 1 2
romance 9 4 6 3 1 144 2 26 318 3 2 1 2 6 1 3 2 0 2 66 1 57 0 1 16 0 2
sci-fi 39 4 14 0 0 49 2 75 77 7 7 0 2 48 0 0 4 0 4 3 187 63 1 1 29 0 2
short 47 24 26 27 10 478 14 1057 1133 45 16 5 9 108 36 12 13 5 47 28 39 1763 13 14 97 4 12
sport 13 0 2 1 2 25 0 108 29 5 1 3 0 1 6 1 0 0 17 0 1 33 167 4 1 0 1
talk-show 0 1 0 2 3 46 2 92 23 6 1 1 0 0 8 0 0 10 32 0 1 29 5 105 5 1 0
thriller 62 7 16 4 4 141 39 55 555 10 6 1 0 188 1 2 23 1 4 9 25 112 0 2 302 0 10
war 7 1 2 1 1 8 0 35 34 0 0 0 2 0 0 0 0 0 0 2 0 7 0 1 3 23 2
western 18 0 7 2 0 34 3 11 116 2 0 0 0 12 0 1 0 0 3 0 1 17 0 0 9 0 785

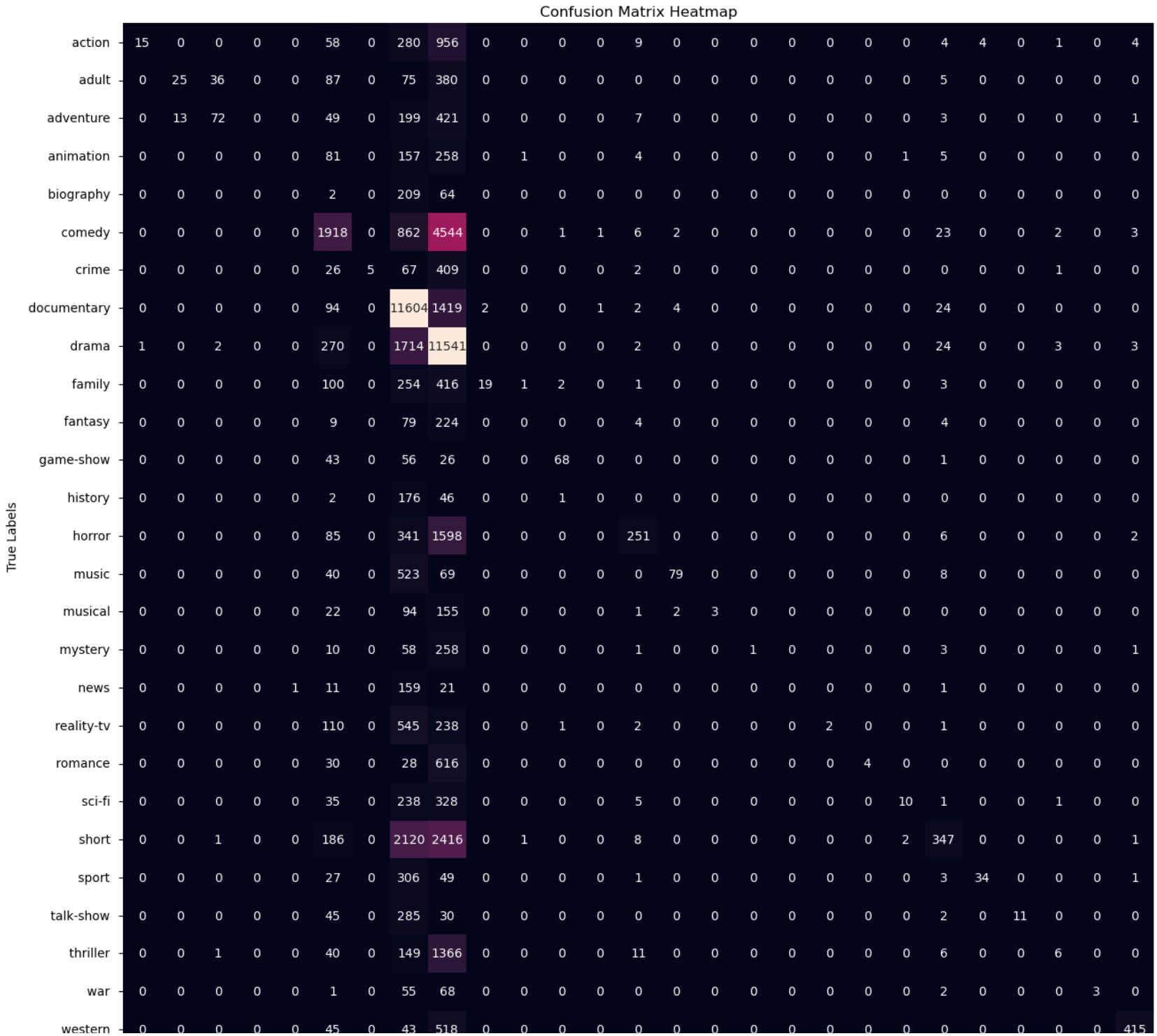

action
adult
adventure
animation
biography
comedy
crime
documentary
drama
family
fantasy
game-show
history
horror
music
musical
mystery
news
reality-tv
romance
sci-fi
short
sport
talk-show
thriller
war
western
Predicted Labels

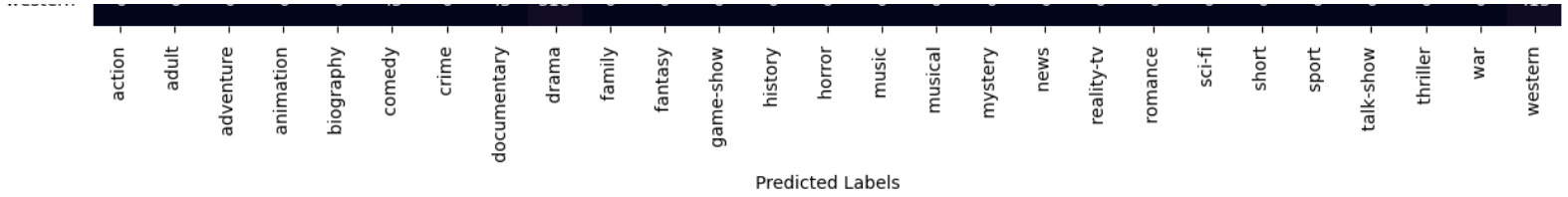
RandomForestClassifier with TfidfVectorizer

In [33]:
```
# Create a Random Forest model
random_forest = RandomForestClassifier()


# Fit the model with GridSearchCV
random_forest.fit(x_train2, y_train)


print("Random Forest - Train Score:",random_forest.score(x_train2, y_train))
print("Random Forest - Test Score:", random_forest.score(x_test2, y_test))

y_pred = random_forest.predict(x_test2)
print(classification_report(y_pred ,y_test))



cm = confusion_matrix(y_test, y_pred)

plt.figure(figsize=(15, 15))  # Adjust the figure size as needed
sns.heatmap(cm, annot=True, fmt='d', cbar=False,
            xticklabels=class_names, yticklabels=class_names)  # Replace 'class_names' with your class labels
plt.xlabel('Predicted Labels')
plt.ylabel('True Labels')
plt.title('Confusion Matrix Heatmap')
plt.show()
```

```
Random Forest - Train Score: 0.9994834615455569
Random Forest - Test Score: 0.48763074879628093
```

	precision	recall	f1-score	support
action	0.01	0.94	0.02	16
adult	0.04	0.66	0.08	38
adventure	0.09	0.64	0.16	112
animation	0.00	0.00	0.00	0
biography	0.00	0.00	0.00	1
comedy	0.26	0.56	0.36	3426
crime	0.01	1.00	0.02	5
documentary	0.88	0.56	0.69	20676
drama	0.85	0.41	0.55	28434
family	0.02	0.90	0.05	21
fantasy	0.00	0.00	0.00	3
game-show	0.35	0.93	0.51	73
history	0.00	0.00	0.00	2
horror	0.11	0.79	0.19	317
music	0.11	0.91	0.20	87
musical	0.01	1.00	0.02	3
mystery	0.00	1.00	0.01	1
news	0.00	0.00	0.00	0
reality-tv	0.00	1.00	0.00	2
romance	0.01	1.00	0.01	4
sci-fi	0.02	0.77	0.03	13
short	0.07	0.73	0.12	476
sport	0.08	0.89	0.15	38
talk-show	0.03	1.00	0.06	11
thriller	0.00	0.43	0.01	14
war	0.02	1.00	0.05	3
western	0.41	0.96	0.57	431
accuracy			0.49	54207
macro avg	0.13	0.67	0.14	54207
weighted avg	0.80	0.49	0.58	54207


Confusion Matrix Heatmap
True Labels
action 15 0 0 0 0 58 0 280 956 0 0 0 0 9 0 0 0 0 0 0 0 4 4 0 1 0 4
adult 0 25 36 0 0 87 0 75 380 0 0 0 0 0 0 0 0 0 0 0 0 5 0 0 0 0 0
adventure 0 13 72 0 0 49 0 199 421 0 0 0 0 7 0 0 0 0 0 0 0 3 0 0 0 0 1
animation 0 0 0 0 0 81 0 157 258 0 1 0 0 4 0 0 0 0 0 0 1 5 0 0 0 0 0
biography 0 0 0 0 0 2 0 209 64 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
comedy 0 0 0 0 0 1918 0 862 4544 0 0 1 1 6 2 0 0 0 0 0 0 23 0 0 2 0 3
crime 0 0 0 0 0 26 5 67 409 0 0 0 0 2 0 0 0 0 0 0 0 0 0 0 1 0 0
documentary 0 0 0 0 0 94 0 11604 1419 2 0 0 1 2 4 0 0 0 0 0 0 24 0 0 0 0 0
drama 1 0 2 0 0 270 0 1714 11541 0 0 0 0 2 0 0 0 0 0 0 0 24 0 0 3 0 3
family 0 0 0 0 0 100 0 254 416 19 1 2 0 1 0 0 0 0 0 0 0 3 0 0 0 0 0
fantasy 0 0 0 0 0 9 0 79 224 0 0 0 0 4 0 0 0 0 0 0 0 4 0 0 0 0 0
game-show 0 0 0 0 0 43 0 56 26 0 0 68 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
history 0 0 0 0 0 2 0 176 46 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
horror 0 0 0 0 0 85 0 341 1598 0 0 0 0 251 0 0 0 0 0 0 0 6 0 0 0 0 2
music 0 0 0 0 0 40 0 523 69 0 0 0 0 0 79 0 0 0 0 0 0 8 0 0 0 0 0
musical 0 0 0 0 0 22 0 94 155 0 0 0 0 1 2 3 0 0 0 0 0 0 0 0 0 0 0
mystery 0 0 0 0 0 10 0 58 258 0 0 0 0 1 0 0 1 0 0 0 0 3 0 0 0 0 1
news 0 0 0 0 1 11 0 159 21 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0
reality-tv 0 0 0 0 0 110 0 545 238 0 0 1 0 2 0 0 0 0 2 0 0 1 0 0 0 0 0
romance 0 0 0 0 0 30 0 28 616 0 0 0 0 0 0 0 0 0 0 4 0 0 0 0 0 0 0
sci-fi 0 0 0 0 0 35 0 238 328 0 0 0 0 5 0 0 0 0 0 0 10 1 0 0 1 0 0
short 0 0 1 0 0 186 0 2120 2416 0 1 0 0 8 0 0 0 0 0 0 2 347 0 0 0 0 1
sport 0 0 0 0 0 27 0 306 49 0 0 0 0 1 0 0 0 0 0 0 0 3 34 0 0 0 1
talk-show 0 0 0 0 0 45 0 285 30 0 0 0 0 0 0 0 0 0 0 0 0 2 0 11 0 0 0
thriller 0 0 1 0 0 40 0 149 1366 0 0 0 0 11 0 0 0 0 0 0 0 6 0 0 6 0 0
war 0 0 0 0 0 1 0 55 68 0 0 0 0 0 0 0 0 0 0 0 0 2 0 0 0 3 0
western 0 0 0 0 0 45 0 43 518 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 415


action
adult
adventure
animation
biography
comedy
crime
documentary
drama
family
fantasy
game-show
history
horror
music
musical
mystery
news
reality-tv
romance
sci-fi
short
sport
talk-show
thriller
war
western
Predicted Labels

In [ ]:

In [ ]:

In [ ]:

In [ ]: