In [8]:
```
# importing library
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [26]:
```
#uploading the dataset "mymoviedb file" in python by creating a variable and using pandas read function.
df = pd.read_csv('mymoviedb.csv', lineterminator='\n')
```

In [28]:
```
# using "head" fuction is used to view first five row of dataset
df.head()
```

Out[28]:

	Release_Date	Title	Overview	Popularity	Vote_Count	Vote_Average	Original_Language	Genre	Poster_Url
0	2021-12-15	Spider-Man: No Way Home	Peter Parker is unmasked and no longer able to...	5083.954	8940	8.3	en	Action, Adventure, Science Fiction	https://image.tmdb.org/t/p/original/1g0dhYtq4i...
1	2022-03-01	The Batman	In his second year of fighting crime, Batman u...	3827.658	1151	8.1	en	Crime, Mystery, Thriller	https://image.tmdb.org/t/p/original/74xTEgt7R3...
2	2022-02-25	No Exit	Stranded at a rest stop in the mountains durin...	2618.087	122	6.3	en	Thriller	https://image.tmdb.org/t/p/original/vDHsLnOWKl...
3	2021-11-24	Encanto	The tale of an extraordinary family, the Madri...	2402.201	5076	7.7	en	Animation, Comedy, Family, Fantasy	https://image.tmdb.org/t/p/original/4j0PNHkMr5...
4	2021-12-22	The King's Man	As a collection of history's worst tyrants and...	1895.511	1793	7.0	en	Action, Adventure, Thriller, War	https://image.tmdb.org/t/p/original/aq4Pwv5Xeu...

In [30]:
```
# Exploring dataset information by using "info" function, to check datatype, finding null or missing values in column.
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 9827 entries, 0 to 9826
Data columns (total 9 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Release_Date       9827 non-null   object
 1   Title              9827 non-null   object
 2   Overview           9827 non-null   object
 3   Popularity         9827 non-null   float64
 4   Vote_Count         9827 non-null   int64
 5   Vote_Average       9827 non-null   float64
 6   Original_Language  9827 non-null   object
 7   Genre              9827 non-null   object
 8   Poster_Url         9827 non-null   object
dtypes: float64(2), int64(1), object(6)
memory usage: 691.1+ KB
```

--> From the above dataset information we get to know that • This dataset has no null or missing values • Overview, Original_Language and Poster-Url wouldn't be useful during analysis so we will remove it for better understanding of the problems. • Datatype of Release_Date column need to be converted from object to date time, to extract only the year value.

In [34]:
```
# Exploring Genre column first 5 row by using "head" function
df['Genre'].head()
```

Out[34]:
```
0    Action, Adventure, Science Fiction
1              Crime, Mystery, Thriller
2                              Thriller
3    Animation, Comedy, Family, Fantasy
4      Action, Adventure, Thriller, War
Name: Genre, dtype: object
```

--> From the avove information from Genre column we get to know that---> • Genres are separated by commas with whitespaces

In [40]:
```
# Using "duplicate" function to check duplicate rows in dataset
# Using "sum" function to sum or total up the duplicate rows in the dataset[if available]
df.duplicated().sum()
```

Out[40]: 0

•From the above information we get to know that the dataset has no duplicate rows in the dataset.

In [42]:
```
# Using "describe" function to find the basic statistics of columns which  contains only numbers[Total count, Maximum, Minimum, Average etc.]
df.describe()
```

Out[42]:

	Popularity	Vote_Count	Vote_Average
count	9827.000000	9827.000000	9827.000000
mean	40.326088	1392.805536	6.439534
std	108.873998	2611.206907	1.129759
min	13.354000	0.000000	0.000000
25%	16.128500	146.000000	5.900000
50%	21.199000	444.000000	6.500000
75%	35.191500	1376.000000	7.100000
max	5083.954000	31077.000000	10.000000

Exploration Summary • This dataframe consisting of 9827 rows and 9 columns. • Our dataset looks a bit tidy with no null or duplicated values. • Release_Date column is in string datatype, it need to be conveted into date time forma, to extract only the year value. • Overview, Original_Languege and Poster-Url wouldn't be so useful during analysis , it will be removed. • There is noticable outliers in Popularity column • Vote_Average will be categorised in POPULAR, AVERAG, BELOW AVERAGE, NOT POPULAR for better understanding and analysis • Genre column has whitespaces after comma , it will be removed for better understanding.

DATA CLEANING OR PREPROCESSING

--> Changing Release_Date column from string to Date time format and extracting year values.

In [46]:
```
# using "head" fuction is used to view first five row of dataset
df.head()
```

Out[46]:

	Release_Date	Title	Overview	Popularity	Vote_Count	Vote_Average	Original_Language	Genre	Poster_Url
0	2021-12-15	Spider-Man: No Way Home	Peter Parker is unmasked and no longer able to...	5083.954	8940	8.3	en	Action, Adventure, Science Fiction	https://image.tmdb.org/t/p/original/1g0dhYtq4i...
1	2022-03-01	The Batman	In his second year of fighting crime, Batman u...	3827.658	1151	8.1	en	Crime, Mystery, Thriller	https://image.tmdb.org/t/p/original/74xTEgt7R3...
2	2022-02-25	No Exit	Stranded at a rest stop in the mountains durin...	2618.087	122	6.3	en	Thriller	https://image.tmdb.org/t/p/original/vDHsLnOWKl...
3	2021-11-24	Encanto	The tale of an extraordinary family, the Madri...	2402.201	5076	7.7	en	Animation, Comedy, Family, Fantasy	https://image.tmdb.org/t/p/original/4j0PNHkMr5...
4	2021-12-22	The King's Man	As a collection of history's worst tyrants and...	1895.511	1793	7.0	en	Action, Adventure, Thriller, War	https://image.tmdb.org/t/p/original/aq4Pwv5Xeu...

In [50]:
```
# Changing Release_date datatype from object to Date time format
df['Release_Date'] = pd.to_datetime(df['Release_Date'])
```

In [52]:
```
# Checking and confirming the datatype
print(df['Release_Date'].dtypes)
```

```
datetime64[ns]
```

In [54]:
```
# For analysis only month is needed, so removing Month and Date from Release_Date column
df['Release_Date'] = df['Release_Date'].dt.year
```

In [56]:
```
# Checking the datatype after the removing month and date from the Release_Date column
df['Release_Date'].dtypes
```

Out[56]: dtype('int32')

In [58]:
```
# Using "Head" function to View first 5 rows in the dataset
df.head()
```

Out[58]:

	Release_Date	Title	Overview	Popularity	Vote_Count	Vote_Average	Original_Language	Genre	Poster_Url
0	2021	Spider-Man: No Way Home	Peter Parker is unmasked and no longer able to...	5083.954	8940	8.3	en	Action, Adventure, Science Fiction	https://image.tmdb.org/t/p/original/1g0dhYtq4i...
1	2022	The Batman	In his second year of fighting crime, Batman u...	3827.658	1151	8.1	en	Crime, Mystery, Thriller	https://image.tmdb.org/t/p/original/74xTEgt7R3...
2	2022	No Exit	Stranded at a rest stop in the mountains durin...	2618.087	122	6.3	en	Thriller	https://image.tmdb.org/t/p/original/vDHsLnOWKl...
3	2021	Encanto	The tale of an extraordinary family, the Madri...	2402.201	5076	7.7	en	Animation, Comedy, Family, Fantasy	https://image.tmdb.org/t/p/original/4j0PNHkMr5...
4	2021	The King's Man	As a collection of history's worst tyrants and...	1895.511	1793	7.0	en	Action, Adventure, Thriller, War	https://image.tmdb.org/t/p/original/aq4Pwv5Xeu...

--> The above table is showig "Release_Date" Column is without Date and month

--> Dropping or Removing The columns which are not needed for the analysis

In [64]:
```
# Creating a list of the column which are not needed
cols = ['Overview', 'Original_Language', 'Poster_Url']
```

In [66]:
```
# Dropping The Columns
df.drop(cols, axis = 1, inplace = True)
```

In [69]:
```
# Confriming The Changes
df.columns
```

Out[69]:
```
Index(['Release_Date', 'Title', 'Popularity', 'Vote_Count', 'Vote_Average',
       'Genre'],
      dtype='object')
```

In [71]:
```
# Using "head" function to call the first 5 rows in the dataset
df.head()
```

Out[71]:

	Release_Date	Title	Popularity	Vote_Count	Vote_Average	Genre
0	2021	Spider-Man: No Way Home	5083.954	8940	8.3	Action, Adventure, Science Fiction
1	2022	The Batman	3827.658	1151	8.1	Crime, Mystery, Thriller
2	2022	No Exit	2618.087	122	6.3	Thriller
3	2021	Encanto	2402.201	5076	7.7	Animation, Comedy, Family, Fantasy
4	2021	The King's Man	1895.511	1793	7.0	Action, Adventure, Thriller, War

--> From the above information we can see that 'Overview','Original Language','Poster URL' Columns has been removed.

CATEGORIZING VOTE_AVERAGE COLUMN

Categorizing the value of VOTE_AVERAGE into 4 Categoris: POPULAR, AVERAGE, BELOW_AVERAGE, NOT_POPULAR

In [241...
```
# Creating a User Defined Function name - "categorize_col()" function
def categorize_col (df, col, labels):
    """
categorizes a certain column based on its quartiles
Args:
(df) df - dataframe we are proccesing
(col) str - to be catigorized column's name
(labels) list - list of labels from min to max
Returns:
(df) df - dataframe with the categorized col
"""

# Creating another variable named "edges" to cut the column accordingly example [Popular-> above 75%, Average-> 50% to 75%, Below_Average-> 25% to 50%,
# Not_Popular-> Under 25%
    edges = [df[col].describe()['min'],
            df[col].describe()['25%'],
            df[col].describe()['50%'],
            df[col].describe()['75%'],
            df[col].describe()['max']]

# Using "Cut" function which helps in categorization
    df[col] = pd.cut(df[col], edges, labels = labels, duplicates='drop')
    return df
```

In [245...
```
# Defining Lebels for the edges
labels = ['not_popular', 'below_avg', 'average', 'popular']

# Calling the user defined function and categorizing column based on labels and edges
#categorize_col (df,'Vote_Average', labels)******

# Checking or confirming changes
df['Vote_Average'].unique()
```

Out[245...
```
['popular', 'below_avg', 'average', 'not_popular', NaN]
Categories (4, object): ['not_popular' < 'below_avg' < 'average' < 'popular']
```

In [249...
```
# using "head" fuction is used to view first five row of dataset
df.head()
```

Out[249...

	Release_Date	Title	Popularity	Vote_Count	Vote_Average	Genre
0	2021	Spider-Man: No Way Home	5083.954	8940	popular	Action, Adventure, Science Fiction
1	2022	The Batman	3827.658	1151	popular	Crime, Mystery, Thriller
2	2022	No Exit	2618.087	122	below_avg	Thriller
3	2021	Encanto	2402.201	5076	popular	Animation, Comedy, Family, Fantasy
4	2021	The King's Man	1895.511	1793	average	Action, Adventure, Thriller, War

--> In the above Dataset we have converted the numerical value in column "Vote_Average" to Labels.

In [251...
```
# using "value.count()" function to check how many movies got popular, average, below average and below popular tags
df['Vote_Average'].value_counts()
```

Out[251...
```
Vote_Average
not_popular    2467
popular        2450
average        2412
below_avg      2398
Name: count, dtype: int64
```

In [253...
```
# Removing duplicate or null values in rows
df.dropna(inplace = True)

# Confriming or checking changes
df.isna().sum()
```

Out[253...
```
Release_Date    0
Title           0
Popularity      0
Vote_Count      0
Vote_Average    0
Genre           0
dtype: int64
```

In [255...
```
# using "head" fuction is used to view first five row of dataset
df.head()
```

Out[255...

	Release_Date	Title	Popularity	Vote_Count	Vote_Average	Genre
0	2021	Spider-Man: No Way Home	5083.954	8940	popular	Action, Adventure, Science Fiction
1	2022	The Batman	3827.658	1151	popular	Crime, Mystery, Thriller
2	2022	No Exit	2618.087	122	below_avg	Thriller
3	2021	Encanto	2402.201	5076	popular	Animation, Comedy, Family, Fantasy
4	2021	The King's Man	1895.511	1793	average	Action, Adventure, Thriller, War

Removing the whitespace and then spliting the Genre into lists, to have only one Genre per row for each movie

In [257...
```
df['Genre'] = df['Genre'].str.split(', ')

# Breaking the Genre column in diffrent lines
df = df.explode('Genre').reset_index(drop=True)

# using "head" fuction is used to view first five row of dataset
df.head()
```

Out[257...

	Release_Date	Title	Popularity	Vote_Count	Vote_Average	Genre
0	2021	Spider-Man: No Way Home	5083.954	8940	popular	Action
1	2021	Spider-Man: No Way Home	5083.954	8940	popular	Adventure
2	2021	Spider-Man: No Way Home	5083.954	8940	popular	Science Fiction
3	2022	The Batman	3827.658	1151	popular	Crime
4	2022	The Batman	3827.658	1151	popular	Mystery

In [259...
```
# Using "astype" function to cast column into category
df['Genre'] = df['Genre'].astype('category')

# Confirming changes and checking datatype
df['Genre'].dtypes
```

Out[259...
```
CategoricalDtype(categories=['Action', 'Adventure', 'Animation', 'Comedy', 'Crime',
                  'Documentary', 'Drama', 'Family', 'Fantasy', 'History',
                  'Horror', 'Music', 'Mystery', 'Romance', 'Science Fiction',
                  'TV Movie', 'Thriller', 'War', 'Western'],
, ordered=False, categories_dtype=object)
```

In [261...
```
# Exploring the dataset using "info" function
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 25552 entries, 0 to 25551
Data columns (total 6 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Release_Date  25552 non-null  int32
 1   Title         25552 non-null  object
 2   Popularity    25552 non-null  float64
 3   Vote_Count    25552 non-null  int64
 4   Vote_Average  25552 non-null  category
 5   Genre         25552 non-null  category
dtypes: category(2), float64(1), int32(1), int64(1), object(1)
memory usage: 749.6+ KB
```

In [ ]:
```
--> From the above information we can see that the number of data has increased, because we have split the Genre into Lists
```

In [263...
```
# Checking the unique values
df.nunique()
```

Out[263...
```
Release_Date     100
Title           9415
Popularity      8088
Vote_Count      3265
Vote_Average       4
Genre             19
dtype: int64
```

DATA VISUALIZATION

In [275...
```
# Setting up seaborn library"sns" to set style
sns.set_style('whitegrid')
```

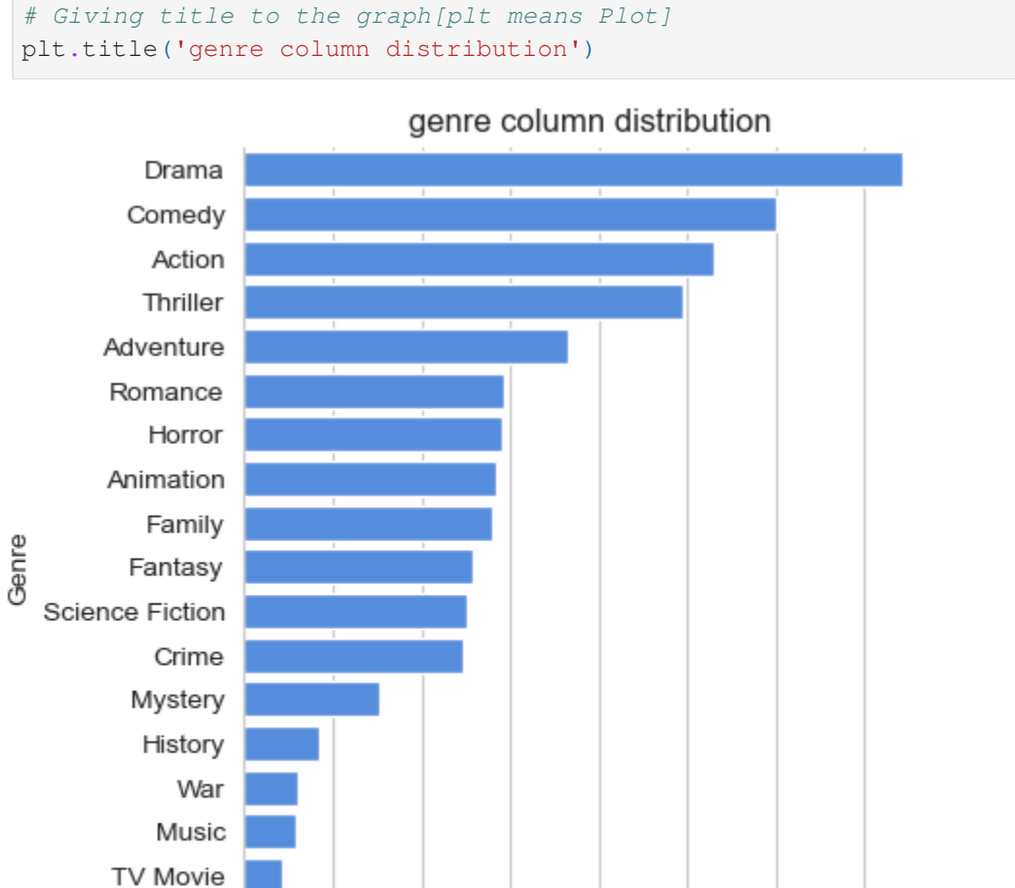
Task 1: What is the most frequent genre of movies released on Netflix?

In [279...
```
# Using "describe" function to show stats of the (Total count of genre,total unique Genre,Which is the top Genre and The frequency of the top Genre)
df['Genre'].describe()
```

Out[279...
```
count     25552
unique       19
top       Drama
freq       3715
Name: Genre, dtype: object
```

In [285...
```
# Using Seaborn library"sns" to Visualize the genre column
sns.catplot(y = 'Genre', data = df, kind = 'count',
           order = df['Genre'].value_counts().index,
           color = '#4287f5')

# Giving title to the graph[plt means Plot]
plt.title('genre column distribution')
```



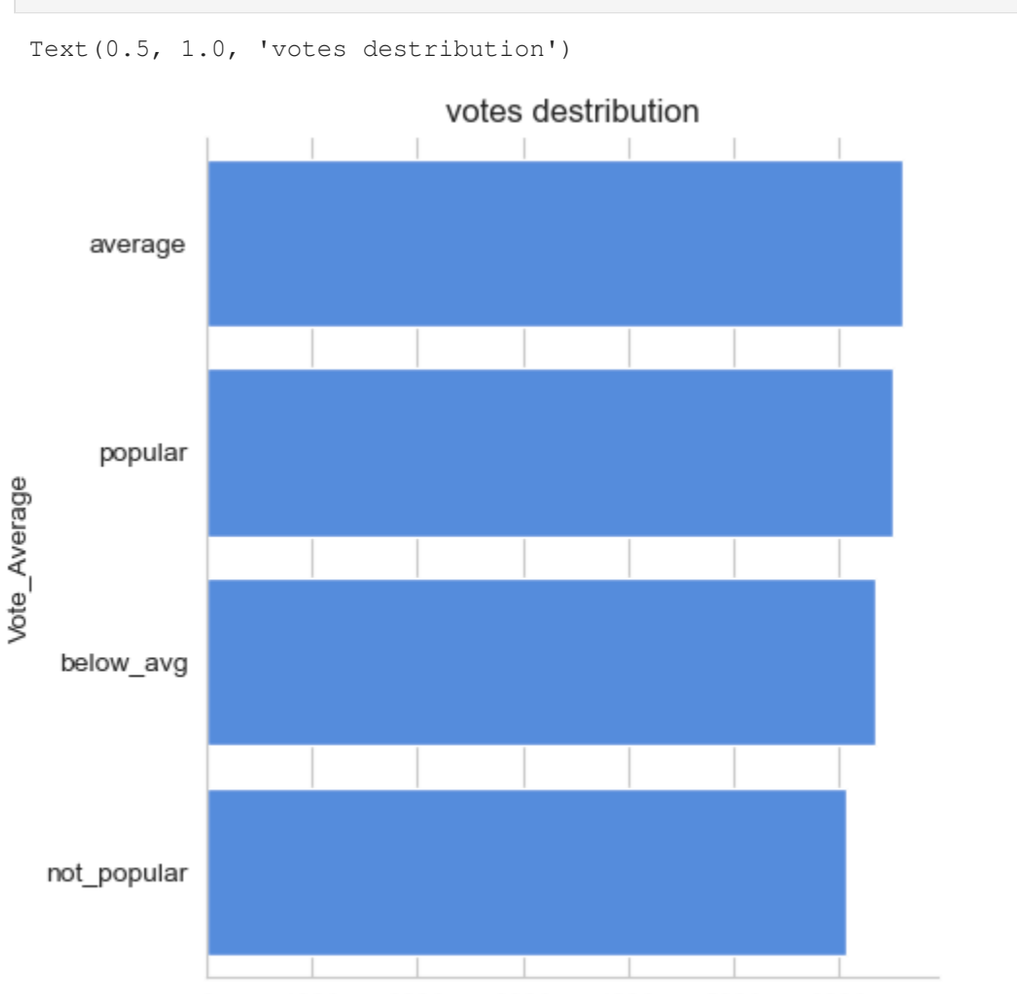
--> The above diagram shows Genre Column - from most frequent to lowest frequent released movies [**The most frequent released genre is "DRAMA" with count of 3715**]

Task 2: Which has the highest votes in Vote Average Column?

In [294...
```
# Using Seaborn library"sns" to Visualize the genre column
sns.catplot(y = 'Vote_Average', data = df, kind = 'count',
          order = df['Vote_Average'].value_counts().index,
          color = '#4287f5')

# Giving title to the graph
plt.title('votes destribution')
```

Out[294... Text(0.5, 1.0, 'votes destribution')



--> The above diagram showing that which type of movies have the highest votes or mostly available in the platform.[**Average are movies are most in the platform**]

Task 3:Which movie got the highest popularity? What is its genre?

In [302...
```
# Checking which movie has highest popularity and its genre
df[df['Popularity'] == df['Popularity'].max()]
```

Out[302...

	Release_Date	Title	Popularity	Vote_Count	Vote_Average	Genre
0	2021	Spider-Man: No Way Home	5083.954	8940	popular	Action
1	2021	Spider-Man: No Way Home	5083.954	8940	popular	Adventure
2	2021	Spider-Man: No Way Home	5083.954	8940	popular	Science Fiction

--> From the above diagram we find the "Spider-Man: No Way Home" has the highest popularity with "Action","Adventure","Science Friction" genre

Task 4:Which movie got the lowest popularity? What is its genre?

In [309...
```
# Checking which movie has lowest popularity and its genre
df[df['Popularity'] == df['Popularity'].min()]
```

Out[309...

	Release_Date	Title	Popularity	Vote_Count	Vote_Average	Genre
25546	2021	The United States vs. Billie Holiday	13.354	152	average	Music
25547	2021	The United States vs. Billie Holiday	13.354	152	average	Drama
25548	2021	The United States vs. Billie Holiday	13.354	152	average	History
25549	1984	Threads	13.354	186	popular	War
25550	1984	Threads	13.354	186	popular	Drama
25551	1984	Threads	13.354	186	popular	Science Fiction

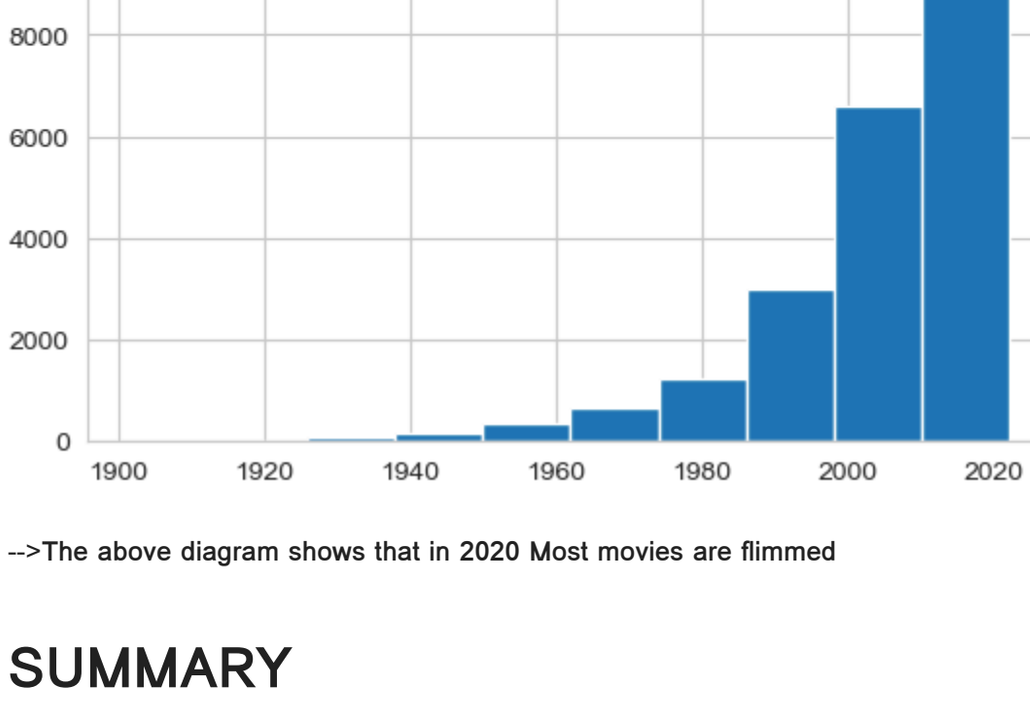
-->From the above diagram we find that "The United States vs. Billie Holiday" and "Threads" has the lowest popularity with "Music","Drama","History", "War","Drama" and "Science Friction" genre

Task 5: which year has the most flimmed movies?

In [315...
```
# Showing which year most movies are flimmed
df['Release_Date'].hist()

#Giving title to the graph
plt.title('Release Date column distribution')
```

Out[315... Text(0.5, 1.0, 'Release Date column distribution')



-->The above diagram shows that in 2020 Most movies are flimmed

SUMMARY

Task 1: What is the most frequent genre of movies released on Netflix?

Drama genre is the most frequent genre in our dataset.

Task 2: Which has the highest votes in Vote Average Column?

Average movies has the highest votes

Task 3:Which movie has got the highest popularity? What is its genre?

"Spider-Man: No Way Home" has the highest popularity with "Action","Adventure","Science Friction" genre

Task 4:Which Movie got the lowest popularity? what is its genre?

"The United States vs. Billie Holiday" and "Threads" has the lowest popularity with "Music","Drama","History", "War","Drama" and "Science Friction" genre

Task 5:Which year has the most flimmed movies?

In year 2020 most movies are flimmed

In [ ]: