Project: TMDB Data Analysis

Table of Contents

Introduction

> I have selected **The Movie Database(TMDB)** that contains data of around 10,000 movies including their votes, release date, score of popularity, budget, revenue and runtime.

With this dataset, we can ask following questions:

- Most popular movies from year to year.
- Movies with higher ratings from year to year.
- Which genres are most popular from year to year? (based on profit).
- What are the properties associated with profitable movies?
- Movies having longest and shortest runtime.
- Best Month to Release a Movie.

In [71]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sn
```

Data Wrangling

Gathering

In [72]:

```
df = pd.read_csv(r'C:\Users\Kami\Downloads\tmdb-movies.csv') #gathering the data, aka data acquisition
```

Assessing

- **Printing few lines of data**

In [73]:

```
df.head()
df.tail(50)
```

Out[73]:

	id	imdb_id	popularity	budget	revenue	original_title	cast	homepage	
10816	16378	tt0077147	0.064602	0	0	The Rutles: All You Need Is Cash	Eric Idle|John Halsey|Ricky Fataar|Neil Innes|...	NaN	Eric

	id	imdb_id	popularity	budget	revenue	original_title	cast	homepage	
10817	13963	tt0077838	0.064029	0	321952	The Last Waltz	Robbie Robertson|Rick Danko|Levon Helm|Richard...	http://www.mgm.com/#/our-titles/1092/The-Last-...	
10818	39995	tt0079482	0.047645	0	0	Long Weekend	John Hargreaves|Briony Behets|Mike McEwen|Roy ...	NaN	
10819	16214	tt0077696	0.044675	0	78000000	Hooper	Burt Reynolds|Robert Klein|Adam West|Jan-Micha...	http://en.wikipedia.org/wiki/Hooper_(film)	Hal
10820	13377	tt0060345	1.227582	315000	0	How the Grinch Stole Christmas!	Boris Karloff|June Foray|Thurl Ravenscroft|Dal...	NaN	J
10821	1714	tt0060390	0.929393	0	0	Fahrenheit 451	Oskar Werner|Julie Christie|Cyril Cusack|Bee D...	NaN	I
10822	396	tt0061184	0.670274	7500000	33736689	Who's Afraid of Virginia Woolf?	Elizabeth Taylor|Richard Burton|George Segal|S...	NaN	Mil
10823	3591	tt0060782	0.613444	0	0	One Million Years B.C.	Raquel Welch|John Richardson|Percy Herbert|Rob...	NaN	Dc
10824	2525	tt0060164	0.533292	18000000	0	The Bible: In the Beginning...	Michael Parks|Ulla Bergryd|Richard Harris|Fran...	NaN	Jol
10825	1052	tt0060176	0.509263	0	0	Blow-Up	David Hemmings|Vanessa Redgrave|Sarah Miles|Jo...	NaN	Mic
10826	874	tt0060665	0.418900	0	0	A Man for All Seasons	Paul Scofield|Wendy Hiller|Leo McKern|Robert S...	NaN	Z
10827	2661	tt0060153	0.410366	1377800	0	Batman	Adam West|Burt Ward|Cesar Romero|Burgess Mered...	NaN	
10828	5780	tt0061107	0.402730	3000000	13000000	Torn Curtain	Paul Newman|Julie Andrews|Lila Kedrova|HansjÃ¶...	NaN	
10829	6644	tt0061619	0.395668	4653000	6000000	El Dorado	John Wayne|Robert Mitchum|James Caan|Charlene ...	NaN	Howa
10830	4772	tt0060268	0.380321	0	0	Cul-de-sac	Donald Pleasence|FranÃ§oise DorlÃ©ac|Lionel St...	NaN	
10831	1888	tt0060424	0.529721	0	0	The Fortune Cookie	Jack Lemmon|Walter Matthau|Ron Rich|Judi West|...	NaN	E
10832	23030	tt0060121	0.358161	4800000	0	Arabesque	Gregory Peck|Sophia Loren|Alan Badel|Kieron Mo...	NaN	Stan
10833	3001	tt0060522	0.737730	0	0	How to Steal a Million	Audrey Hepburn|Peter O'Toole|Eli Wallach|Hugh ...	NaN	Will
10834	12639	tt0060897	0.310688	0	0	Return of the Seven	Yul Brynner|Robert Fuller|JuliÃ¡n Mateos|Warre...	NaN	Bur

Mateos|Warre...

	id	imdb_id	popularity	budget	revenue	original_title	cast	homepage	
10835	5923	tt0060934	0.299911	12000000	20000000	The Sand Pebbles	Steve McQueen|Richard Attenborough|Richard Cre...	NaN	Ro
10836	38720	tt0061170	0.239435	0	0	Walk Don't Run	Cary Grant|Samantha Eggar|Jim Hutton|John Stan...	NaN	
10837	19728	tt0060177	0.291704	0	0	The Blue Max	George Peppard|James Mason|Ursula Andress|Jere...	NaN	
10838	22383	tt0060862	0.151845	0	0	The Professionals	Burt Lancaster|Lee Marvin|Robert Ryan|Woody St...	NaN	
10839	13353	tt0060550	0.276133	0	0	It's the Great Pumpkin, Charlie Brown	Christopher Shea|Sally Dryer|Kathy Steinberg|A...	NaN	Bill
10840	34388	tt0060437	0.102530	0	0	Funeral in Berlin	Michael Caine|Paul Hubschmid|Oskar Homolka|Eva...	NaN	Guy
10841	42701	tt0062262	0.264925	75000	0	The Shooting	Will Hutchins|Millie Perkins|Jack Nicholson|Wa...	NaN	Mont
10842	36540	tt0061199	0.253437	0	0	Winnie the Pooh and the Honey Tree	Sterling Holloway|Junius Matthews|Sebastian Ca...	NaN	R
10843	29710	tt0060588	0.252399	0	0	Khartoum	Charlton Heston|Laurence Olivier|Richard Johns...	NaN	Dea
10844	23728	tt0059557	0.236098	0	0	Our Man Flint	James Coburn|Lee J. Cobb|Gila Golan|Edward Mul...	NaN	Da
10845	5065	tt0059014	0.230873	0	0	Carry On Cowboy	Sid James|Jim Dale|Angela Douglas|Kenneth Will...	NaN	
10846	17102	tt0059127	0.212716	0	0	Dracula: Prince of Darkness	Christopher Lee|Barbara Shelley|Andrew Keir|Fr...	NaN	Terer
10847	28763	tt0060548	0.034555	0	0	Island of Terror	Peter Cushing|Edward Judd|Carole Gray|Eddie By...	NaN	Terer
10848	2161	tt0060397	0.207257	5115000	12000000	Fantastic Voyage	Stephen Boyd|Raquel Welch|Edmond O'Brien|Donal...	NaN	
10849	28270	tt0060445	0.206537	0	0	Gambit	Michael Caine|Shirley MacLaine|Herbert Lom|Joh...	NaN	Rona
10850	26268	tt0060490	0.202473	0	0	Harper	Paul Newman|Lauren Bacall|Julie Harris|Arthur ...	NaN	Ja
10851	15347	tt0060182	0.342791	0	0	Born Free	Virginia McKenna|Bill Travers|Geoffrey Keen|Pe...	NaN	
10852	37301	tt0060165	0.227220	0	0	A Big Hand for the Little	Henry Fonda|Joanne Woodward|Jason	NaN	Fie

<table>
<tr><th></th><th>id</th><th>imdb_id</th><th>popularity</th><th>budget</th><th>revenue</th><th>original_title</th><th>cast</th><th>homepage</th><th></th></tr>
<tr><td></td><td></td><td></td><td></td><td></td><td></td><td>Lady</td><td>Robards|Paul</td><td></td><td></td></tr>
<tr><td>10853</td><td>15598</td><td>tt0060086</td><td>0.163592</td><td>0</td><td>0</td><td>Alfie</td><td>Michael Caine|Shelley Winters|Millicent Martin...</td><td>NaN</td><td>Lev</td></tr>
<tr><td>10854</td><td>31602</td><td>tt0060232</td><td>0.146402</td><td>0</td><td>0</td><td>The Chase</td><td>Marlon Brando|Jane Fonda|Robert Redford|E.G. M...</td><td>NaN</td><td>Ar</td></tr>
<tr><td>10855</td><td>13343</td><td>tt0059221</td><td>0.141026</td><td>700000</td><td>0</td><td>The Ghost & Mr. Chicken</td><td>Don Knotts|Joan Staley|Liam Redmond|Dick Sarge...</td><td>NaN</td><td>A</td></tr>
<tr><td>10856</td><td>20277</td><td>tt0061135</td><td>0.140934</td><td>0</td><td>0</td><td>The Ugly Dachshund</td><td>Dean Jones|Suzanne Pleshette|Charles Ruggles|K...</td><td>NaN</td><td>Norn</td></tr>
<tr><td>10857</td><td>5921</td><td>tt0060748</td><td>0.131378</td><td>0</td><td>0</td><td>Nevada Smith</td><td>Steve McQueen|Karl Malden|Brian Keith|Arthur K...</td><td>NaN</td><td></td></tr>
<tr><td>10858</td><td>31918</td><td>tt0060921</td><td>0.317824</td><td>0</td><td>0</td><td>The Russians Are Coming, The Russians Are Coming</td><td>Carl Reiner|Eva Marie Saint|Alan Arkin|Brian K...</td><td>NaN</td><td></td></tr>
<tr><td>10859</td><td>20620</td><td>tt0060955</td><td>0.089072</td><td>0</td><td>0</td><td>Seconds</td><td>Rock Hudson|Salome Jens|John Randolph|Will Gee...</td><td>NaN</td><td>Frank</td></tr>
<tr><td>10860</td><td>5060</td><td>tt0060214</td><td>0.087034</td><td>0</td><td>0</td><td>Carry On Screaming!</td><td>Kenneth Williams|Jim Dale|Harry H. Corbett|Joa...</td><td>NaN</td><td></td></tr>
<tr><td>10861</td><td>21</td><td>tt0060371</td><td>0.080598</td><td>0</td><td>0</td><td>The Endless Summer</td><td>Michael Hynson|Robert August|Lord 'Tally Ho' B...</td><td>NaN</td><td>Bru</td></tr>
<tr><td>10862</td><td>20379</td><td>tt0060472</td><td>0.065543</td><td>0</td><td>0</td><td>Grand Prix</td><td>James Garner|Eva Marie Saint|Yves Montand|Tosh...</td><td>NaN</td><td>Frank</td></tr>
<tr><td>10863</td><td>39768</td><td>tt0060161</td><td>0.065141</td><td>0</td><td>0</td><td>Beregis Avtomobilya</td><td>Innokentiy Smoktunovskiy|Oleg Efremov|Georgi Z...</td><td>NaN</td><td></td></tr>
<tr><td>10864</td><td>21449</td><td>tt0061177</td><td>0.064317</td><td>0</td><td>0</td><td>What's Up, Tiger Lily?</td><td>Tatsuya Mihashi|Akiko Wakabayashi|Mie Hama|Joh...</td><td>NaN</td><td>Wo</td></tr>
<tr><td>10865</td><td>22293</td><td>tt0060666</td><td>0.035919</td><td>19000</td><td>0</td><td>Manos: The Hands of Fate</td><td>Harold P. Warren|Tom Neyman|John Reynolds|Dian...</td><td>NaN</td><td></td></tr>
</table>

50 rows × 21 columns

- **Number of rows in data**
- **Number of columns in data**

In [74]:

```
num_rows , num_cols = df.shape
print("Number of rows = {}, Number of Columns = {}".format(num_rows,num_cols))
```

```
Number of rows = 10866, Number of Columns = 21
```

- **Duplicate rows in data**

- **Duplicate rows in data**

In [75]:

```
print('Number of duplicated row = ', df.duplicated().sum()) #we have one duplicated row.
#let's see duplicated row
duplicate_row_id = df [ df.duplicated() ].id.iloc[0]  #extracting duplicate row id.
print ('Duplicate row ID = ', duplicate_row_id)
df [ df.id == duplicate_row_id ] #taking look at duplicated rows.
```

```
Number of duplicated row =  1
Duplicate row ID =  42194
```

Out[75]:

	id	imdb_id	popularity	budget	revenue	original_title	cast	homepage	director	tagline	...	overview	runtime
2089	42194	tt0411951	0.59643	30000000	967000	TEKKEN	Jon Foo|Kelly Overton|Cary-Hiroyuki Tagawa|Ian...	NaN	Dwight H. Little	Survival is no game	...	In the year of 2039, after World Wars destroy ...	92
2090	42194	tt0411951	0.59643	30000000	967000	TEKKEN	Jon Foo|Kelly Overton|Cary-Hiroyuki Tagawa|Ian...	NaN	Dwight H. Little	Survival is no game	...	In the year of 2039, after World Wars destroy ...	92

2 rows × 21 columns

Issue No 1: Need to drop duplicated row.

- **Datatypes of each column**

In [76]:

```
df.dtypes
```

Out[76]:

```
id                       int64
imdb_id                 object
popularity             float64
budget                   int64
revenue                  int64
original_title          object
cast                    object
homepage                object
director                object
tagline                 object
keywords                object
overview                object
runtime                  int64
genres                  object
production_companies    object
release_date            object
vote_count               int64
vote_average           float64
release_year             int64
budget_adj             float64
revenue_adj            float64
dtype: object
```

Issue No 2: Change datatype of release_date from strings to datetime.

- **Looking for missing values in each column**

- **Looking for missing values in each column**

In [77]:

```
df.isnull().sum()
```

Out[77]:

```
id                        0
imdb_id                  10
popularity                0
budget                    0
revenue                   0
original_title            0
cast                     76
homepage               7930
director                 44
tagline                2824
keywords               1493
overview                  4
runtime                   0
genres                   23
production_companies   1030
release_date              0
vote_count                0
vote_average              0
release_year              0
budget_adj                0
revenue_adj               0
dtype: int64
```

Issue No 3: Need to address missing values for following columns
['imdb_id','cast','homepage','director','tagline','keywords','overview','genres','production_companies']

- **Non null unique values in each column**

In [78]:

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10866 entries, 0 to 10865
Data columns (total 21 columns):
 #   Column                Non-Null Count  Dtype
---  ------                --------------  -----
 0   id                    10866 non-null  int64
 1   imdb_id               10856 non-null  object
 2   popularity            10866 non-null  float64
 3   budget                10866 non-null  int64
 4   revenue               10866 non-null  int64
 5   original_title        10866 non-null  object
 6   cast                  10790 non-null  object
 7   homepage              2936 non-null   object
 8   director              10822 non-null  object
 9   tagline               8042 non-null   object
 10  keywords              9373 non-null   object
 11  overview              10862 non-null  object
 12  runtime               10866 non-null  int64
 13  genres                10843 non-null  object
 14  production_companies  9836 non-null   object
 15  release_date          10866 non-null  object
 16  vote_count            10866 non-null  int64
 17  vote_average          10866 non-null  float64
 18  release_year          10866 non-null  int64
 19  budget_adj            10866 non-null  float64
 20  revenue_adj           10866 non-null  float64
dtypes: float64(4), int64(6), object(11)
memory usage: 1.7+ MB
```

- **identifying rows with missing data**

In [79]:

In [79]:

```
df[ df.isnull().any(axis=1) ]
```

Out[79]:

	id	imdb_id	popularity	budget	revenue	original_title	cast	homepage	director	tagline
18	150689	tt1661199	5.556818	95000000	542351353	Cinderella	Lily James|Cate Blanchett|Richard Madden|Helen...	NaN	Kenneth Branagh	Midnight is just the beginning.
21	307081	tt1798684	5.337064	30000000	91709827	Southpaw	Jake Gyllenhaal|Rachel McAdams|Forest Whitaker...	NaN	Antoine Fuqua	Believe in Hope.
26	214756	tt2637276	4.564549	68000000	215863606	Ted 2	Mark Wahlberg|Seth MacFarlane|Amanda Seyfried|...	NaN	Seth MacFarlane	Ted is Coming, Again.
32	254470	tt2848292	3.877764	29000000	287506194	Pitch Perfect 2	Anna Kendrick|Rebel Wilson|Hailee Steinfeld|Br...	NaN	Elizabeth Banks	We're back pitches
33	296098	tt3682448	3.648210	40000000	162610473	Bridge of Spies	Tom Hanks|Mark Rylance|Amy Ryan|Alan Alda|Seba...	NaN	Steven Spielberg	In the shadow of war, one man showed the world...
...	...	...	...	...	...	...	...	...	...	...
10861	21	tt0060371	0.080598	0	0	The Endless Summer	Michael Hynson|Robert August|Lord 'Tally Ho' B...	NaN	Bruce Brown	NaN
10862	20379	tt0060472	0.065543	0	0	Grand Prix	James Garner|Eva Marie Saint|Yves Montand|Tosh...	NaN	John Frankenheimer	Cinerama sweeps YOU into a drama of speed and ...
10863	39768	tt0060161	0.065141	0	0	Beregis Avtomobilya	Innokentiy Smoktunovskiy|Oleg Efremov|Georgi Z...	NaN	Eldar Ryazanov	NaN
10864	21449	tt0061177	0.064317	0	0	What's Up, Tiger Lily?	Tatsuya Mihashi|Akiko Wakabayashi|Mie Hama|Joh...	NaN	Woody Allen	WOODY ALLEN STRIKES BACK!
10865	22293	tt0060666	0.035919	19000	0	Manos: The Hands of Fate	Harold P. Warren|Tom Neyman|John Reynolds|Dian...	NaN	Harold P. Warren	It's Shocking! It's Beyond Your Imagination!

8874 rows × 21 columns

- **Notice we have 8874 rows with missing data**

Cleaning

Issue No 1: Need to drop duplicated row.
Issue No 2: Change datatype of release_date from strings to datetime.
Issue No 3: Need to address missing values for following columns
['imdb_id','cast','homepage','director','tagline','keywords','overview','genres','production_companies']

Solving Issue No 1: Need to drop duplicated row.

In [80]:

In [80]:

```
df_clean = df.copy() #making copy so that original data set remains intact.
df_clean.drop_duplicates(inplace=True)
```

In [81]:

```
df_clean.duplicated().sum() #verifying that there should be now 0 duplicates.
df_clean.shape #Also notice , number of rows are now 10865 instead of 10866. So our Issue No 1 is resolved successfully.
```

Out[81]:

```
(10865, 21)
```

Solving Issue No 2: Change datatype of release_date from strings to datetime.

In [82]:

```
df_clean['release_date'] = pd.to_datetime(df_clean['release_date'])
```

In [83]:

```
print(df_clean.dtypes['release_date']) #verifying datetime conversion.
df_clean.tail(-10700)[['release_date','release_year']]
```

```
datetime64[ns]
```

Out[83]:

	release_date	release_year
10701	2065-08-06	1965
10702	2065-10-15	1965
10703	2065-06-22	1965
10704	2065-06-23	1965
10705	2065-04-01	1965
...	...	...
10861	2066-06-15	1966
10862	2066-12-21	1966
10863	2066-01-01	1966
10864	2066-11-02	1966
10865	2066-11-15	1966

165 rows × 2 columns

It seems like above conversion converted datatype to date time but there are some problems. Years like 1966,1965 are converted into 2066 and 2065 respectively, which is obviously not good enough. So we need to fix this somehow.

In [84]:

```
from datetime import datetime
df_clean[ df_clean['release_date'] >  datetime.today() ].shape #So we have total 362 rows with wrong year information.
```

Out[84]:

```
(362, 21)
```

In [85]:

```
#Now we will extract unique future years and type cast them as a list.
years_to_be_changed = list (df_clean[ df_clean['release_date'] >  datetime.today() ]['release_date'].dt.year.unique())
```

In [86]:

```
def change_year (date):
    if date.year in years_to_be_changed:
        #print(date)
        #print(date.year)
        date =  datetime.strftime(date,'%Y-%m-%d')
        date= date.replace('20','19')
        date = datetime.strptime(date,'%Y-%m-%d')
    return date



df_clean['release_date'] = df_clean['release_date'].apply(change_year) #applying function change_ye
ar to release_date column.
```

In [87]:

```
df_clean.tail(20)[['release_date','release_year']]
df_clean[ df_clean['release_date'] >  datetime.today() ]
#So now we don't have any rows with future dates,It means Issue No 2 is resolved successfully.
```

Out[87]:

id	imdb_id	popularity	budget	revenue	original_title	cast	homepage	director	tagline	...	overview	runtime	genres	production

0 rows × 21 columns

Solving Issue No 3: Need to address missing values for following columns ['imdb_id','cast','homepage','director','tagline','keywords','overview','genres','production_companies']

In [88]:

```
null_columns_mask = df_clean.isnull().sum()>0 #creating null columns mask.
print (df_clean.isnull().sum()[null_columns_mask].sort_values()) #Printing names of columns with
null values in ascending order.
```

```
overview                   4
imdb_id                   10
genres                    23
director                  44
cast                      76
production_companies    1030
keywords                1493
tagline                 2824
homepage                7929
dtype: int64
```

It looks like we don't have homepage,tagline,keywords, production_companies for many of the movies. I think deleting these columns will be a good solution as they won't be playing any significant role in terms of analyzing data.

In [89]:

```
df_clean.drop(['homepage','tagline','keywords','production_companies'],axis=1,inplace=True)
```

In [90]:

```
df_clean.isnull().sum()[null_columns_mask].sort_values() #now again taking look at remaining null
values (in ascending order).
```

Out[90]:

```
overview     4
imdb_id     10
genres      23
director    44
cast        76
```

```
dtype: int64
```

These columns can play important role in analyzing the data or trends,especially genres, director or even cast. So, deleting these won't be a good solution.
This time I would opt for deleting rows with missing values.

In [91]:

```
df_clean[df_clean.isna().any(axis=1)].shape
df_clean.dropna(inplace=True)
df_clean.shape
```

Out[91]:

```
(10724, 17)
```

In [92]:

```
df_clean.isnull().sum().sum() #Verifying if there any null value exists in any of the column. Sinc
e, it returns 0, means
#there are no more null or missing values left. It means our Issue No 3 is also resolved now. We a
re good to go for EDA.
```

Out[92]:

```
0
```

Exploratory Data Analysis

Research Question 1: Most popular movies from year to year.

- **we can answer this question by finding max popularity score for movie in each year.**

step 1: find max popularity score for each release_year.

In [93]:

```
popularity_score_year_wise =  list(df_clean.groupby(['release_year'])['popularity'].max())
df_popular_score_year_wise =  df_clean[df_clean.popularity.isin(popularity_score_year_wise)]
#df_popular_score_year_wise[['original_title','release_year','popularity','vote_count']]
```

step 2: sort by descending order of release year.

In [94]:

```
df_popular_score_year_wise.sort_values('release_year',ascending=False,inplace=True)
#df_popular_score_year_wise[['original_title','release_year','popularity','vote_count']]
```

```
C:\Users\Kami\anaconda3\lib\site-packages\ipykernel_launcher.py:1: 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-versus-a-copy
  """Entry point for launching an IPython kernel.
```

step 3: extract movie title and release year to be used as x axis labels.

In [95]:

```
xlabels = []
for i,j in df_popular_score_year_wise[['original_title','release_year']].iterrows():
       # print (j['original_title'],j['release_year'])
        xlabels.append(j['original_title'] + ' ' + str(j['release_year']))
```

```
#xlabels
```

step 4: extract popularity score to be plotted on y axis.

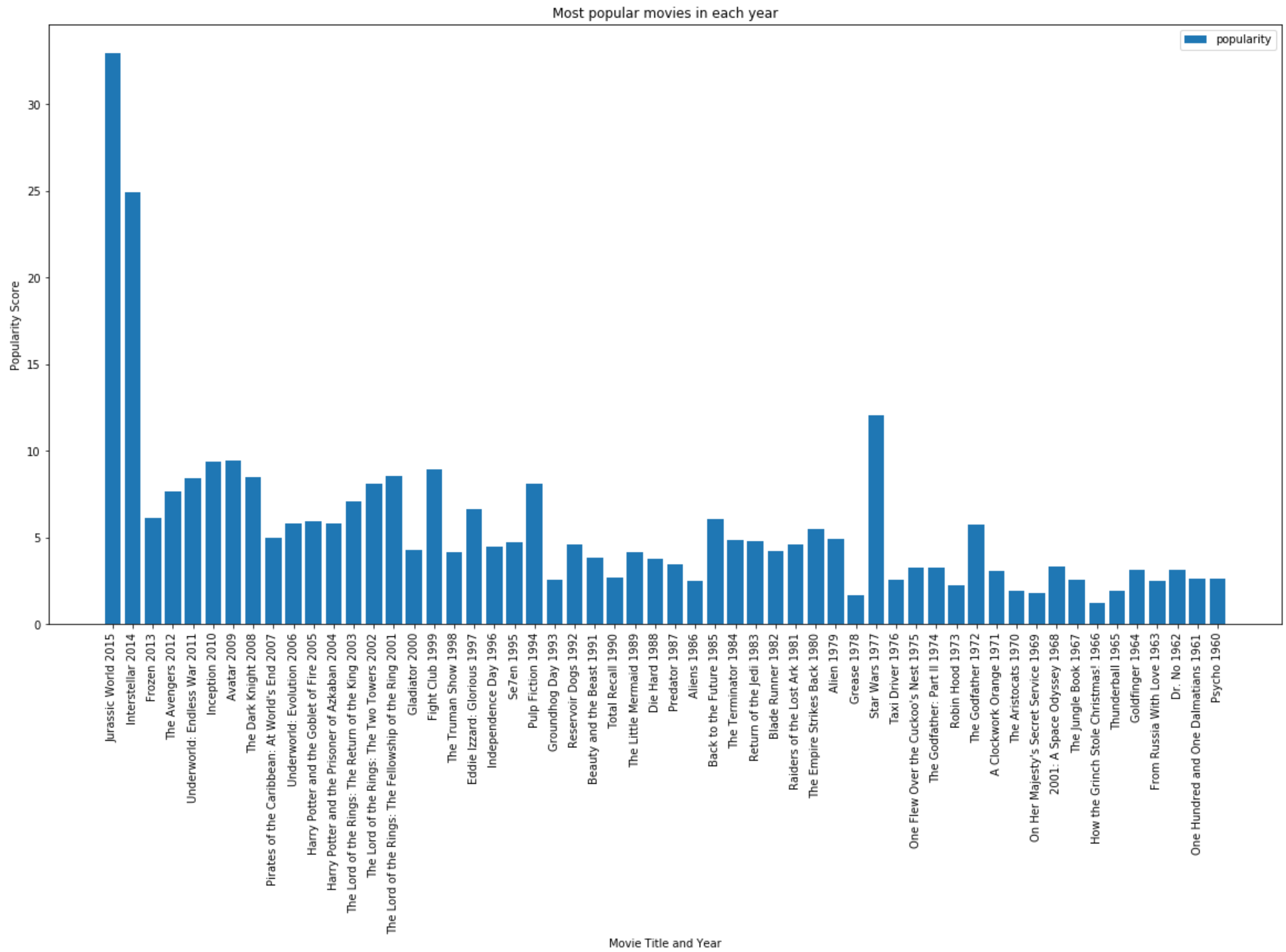
In [96]:

```
y_score = []
for i,j in df_popular_score_year_wise[['popularity']].iterrows():
    #print (j['popularity'])
    y_score.append(j['popularity'])
#y_score
```

step 5: finally plotting bar char to see graphically which movie was popular in each year along with its popularity score.

In [97]:

```
plt.subplots(figsize=(20,10))
plt.bar(x=xlabels,height=y_score,label='popularity')
plt.xticks(rotation=90);
plt.xlabel('Movie Title and Year')
plt.ylabel('Popularity Score')
plt.title("Most popular movies in each year")
plt.legend();
```



- **For movies released in year 2015, most popular movie was *"Jurasic World"* while *"Psycho"* received more popularity among movies released in 1960.**

Research Question 2: Movies with higher ratings from year to year.

- **we can answer this question by calculating the max vote_average and grouping it by release_year.**

step 1: first we will find movies with max average rating in each year.

In [65]:

```
vote_year_wise = dict (df_clean.groupby(['release_year'])['vote_average'].max())
cols_list = list(df_clean.columns)
df_vote_average = pd.DataFrame(columns=cols_list) #creating empty dataframe to store year wise mov
ies with max vote_average.

for key,value in vote_year_wise.items():
    #print (df_clean[(df_clean.release_year == key) & (df_clean.vote_average == value)])
    #print("\n\n")
    df_vote_average = df_vote_average.append(df_clean[(df_clean.release_year == key) & (df_clean.vo
te_average == value)])
```

step 2: extract movie title and release year to be used as x axis labels.

In [66]:

```
df_vote_average.sort_values('release_year',ascending=False,inplace=True)
xlabels = []
for i,j in df_vote_average[['original_title','release_year']].iterrows():
       # print (j['original_title'],j['release_year'])
        xlabels.append(j['original_title'] + ' ' + str(j['release_year']))
#xlabels
```

step 3: extract vote_average to be plotted on y axis.

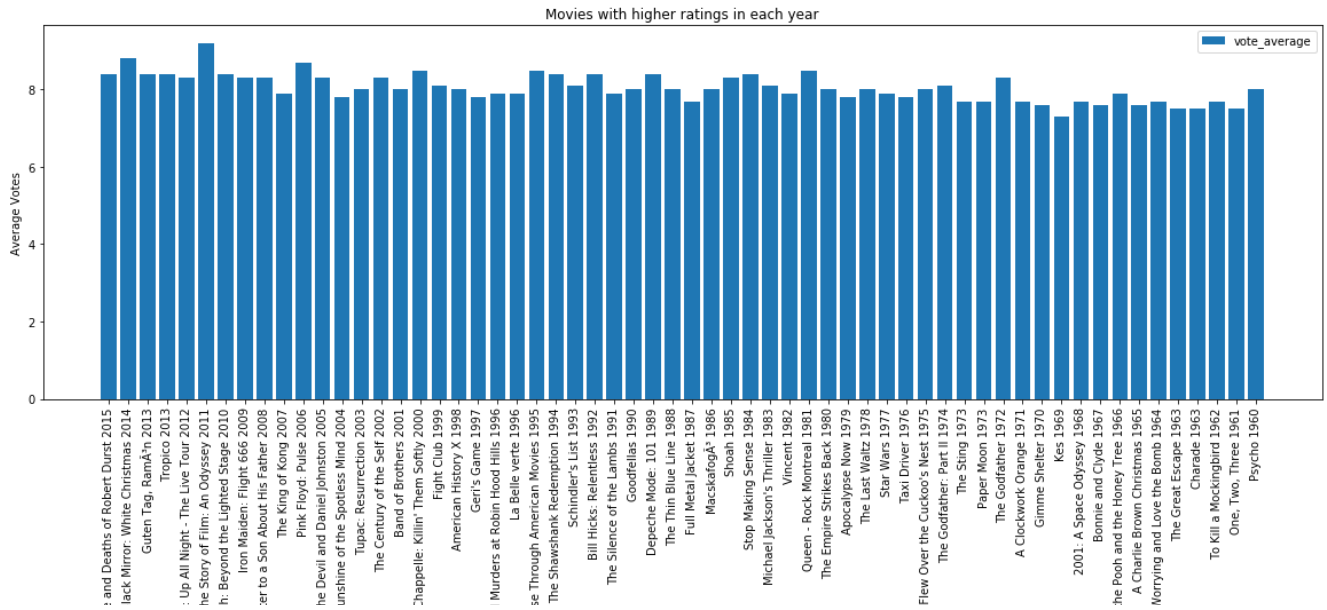
In [67]:

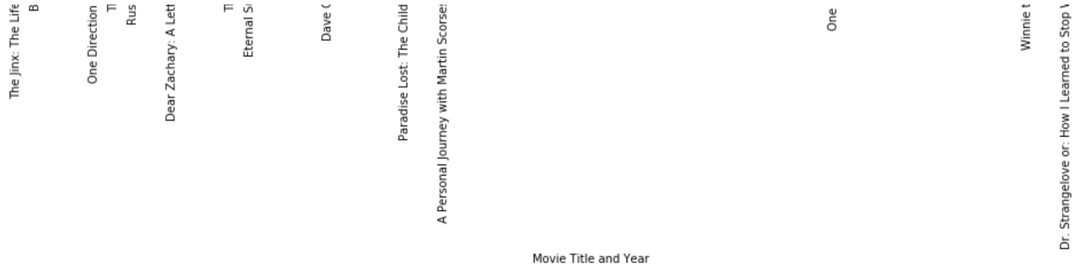
```
y_score = []
for i,j in df_vote_average[['vote_average']].iterrows():
    #print (j['popularity'])
    y_score.append(j['vote_average'])
#y_score
```

step 4: finally plotting bar char to see graphically which movie received highest average votes in each year.

In [68]:

```
plt.subplots(figsize=(20,6))
plt.bar(x=xlabels,height=y_score,label='vote_average')
plt.xticks(rotation=90);
plt.xlabel('Movie Title and Year')
plt.ylabel('Average Votes')
plt.title("Movies with higher ratings in each year")
plt.legend();
```





- **Among movies that were released in year 2015, the movie which received highest vote average was *"the jinx life and death of robert durst"* while *"Psycho"* received highest vote average among movies released in 1960.**

Research Question 3: Which genres are most popular from year to year? (based on profit)

- **To answer this question, we can find which genres contribute most to the profitable movie in each year.**

step 1: Calculating profit for each movie.

In [36]:

```
df_clean.insert(5,'profit',df_clean.revenue-df_clean.budget) #calculating profit of each movie.
```

step 2: Taking profitable movies into account.

In [37]:

```
df_profitable_movies =  df_clean[(df_clean.profit > 0) & (df_clean.budget > 0)] #Taking only profit
able movies in every year.
df_profitable_movies.shape
```

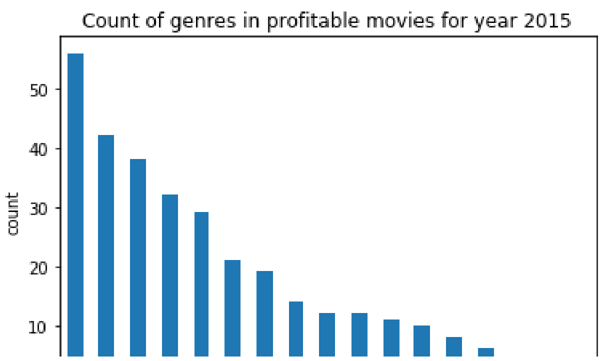
Out[37]:

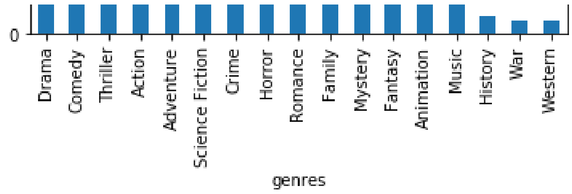
```
(2776, 18)
```

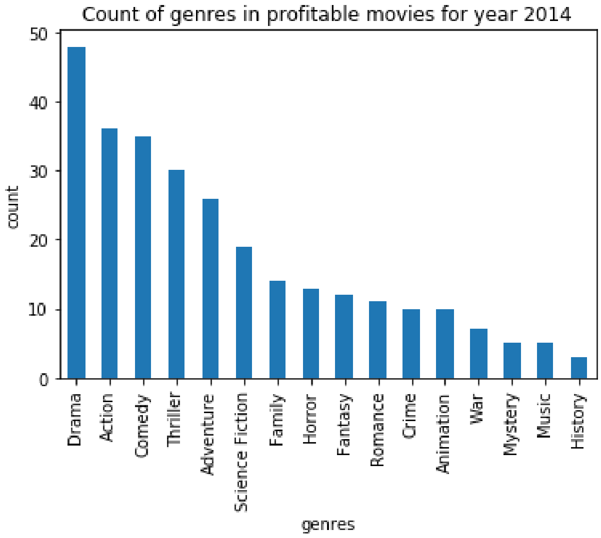
step 3: Now we will plot bar chart for each year separately to visualize which genres was most popular in profitable movies in each year.

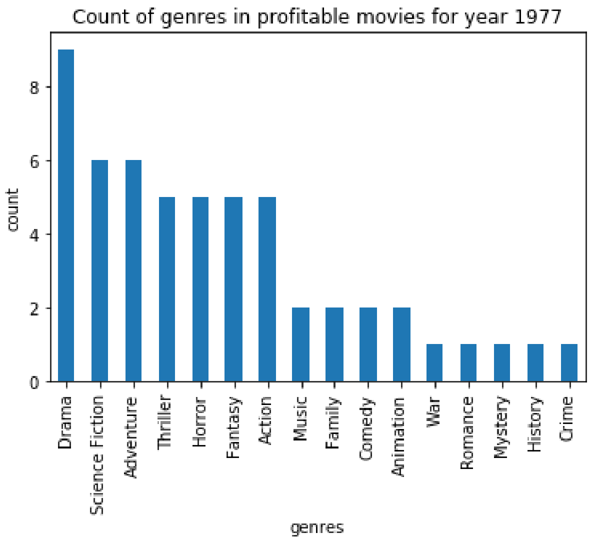
In [38]:

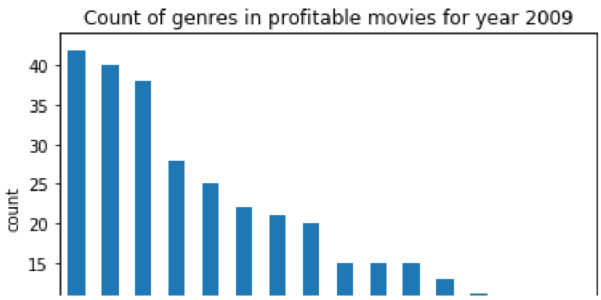
```
release_years = list (df_profitable_movies['release_year'].unique())

genres_dict = {}
for year in release_years:
    df_genres = df_profitable_movies[df_profitable_movies.release_year == year ]
    dummies_genres = df_genres.genres.str.get_dummies('|')
    dummies_genres.sum().sort_values(ascending=False).plot(kind='bar');
    plt.xlabel('genres')
    plt.ylabel('count')
    plt.title('Count of genres in profitable movies for year {}'.format(year))
    plt.show()
    print("\n\n\n")
```

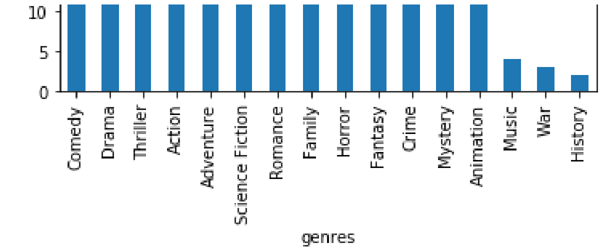



0
Drama
Comedy
Thriller
Action
Adventure
Science Fiction
Crime
Horror
Romance
Family
Mystery
Fantasy
Animation
Music
History
War
Western
genres

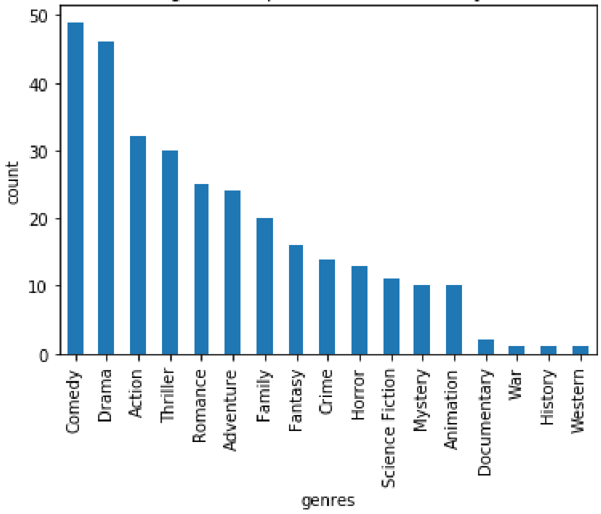

Count of genres in profitable movies for year 2014
count
0
10
20
30
40
50
Drama
Action
Comedy
Thriller
Adventure
Science Fiction
Family
Horror
Fantasy
Romance
Crime
Animation
War
Mystery
Music
History
genres


Count of genres in profitable movies for year 1977
count
0
2
4
6
8
Drama
Science Fiction
Adventure
Thriller
Horror
Fantasy
Action
Music
Family
Comedy
Animation
War
Romance
Mystery
History
Crime
genres

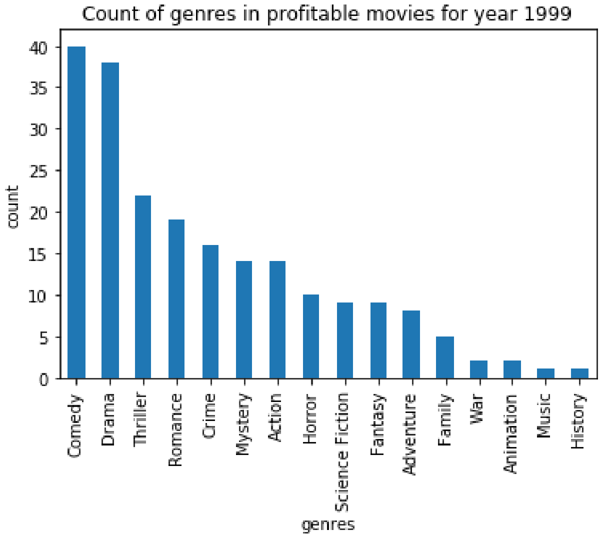

Count of genres in profitable movies for year 2009
count
15
20
25
30
35
40



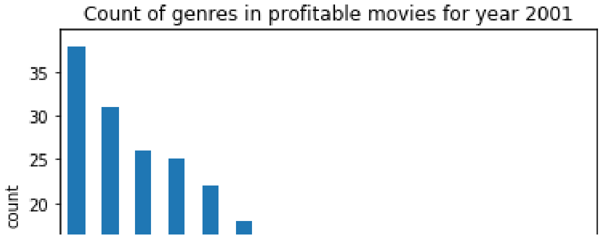
Count of genres in profitable movies for year 2010

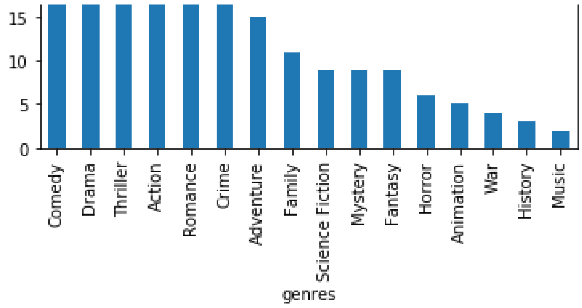


Count of genres in profitable movies for year 1999

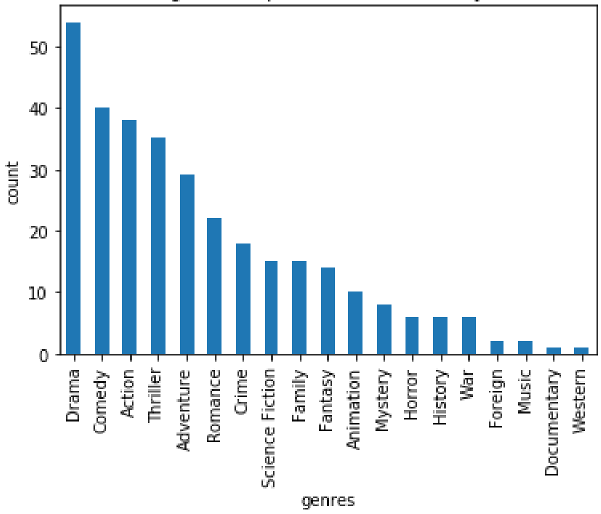


Count of genres in profitable movies for year 2001

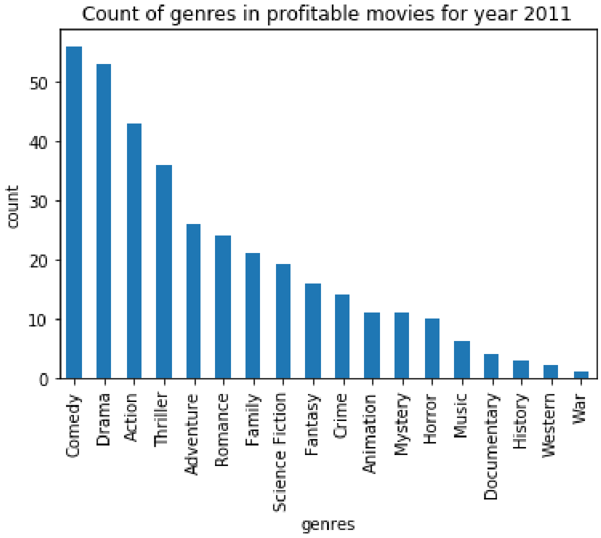




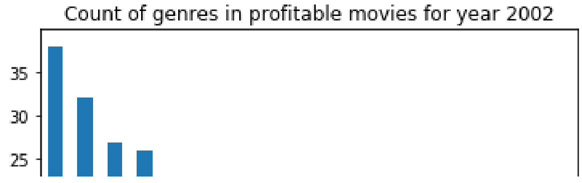
Count of genres in profitable movies for year 2008

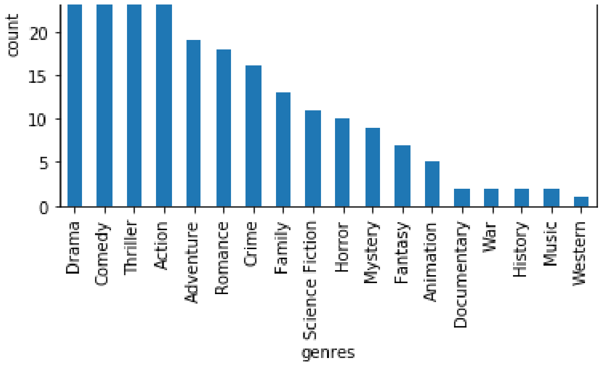


Count of genres in profitable movies for year 2011

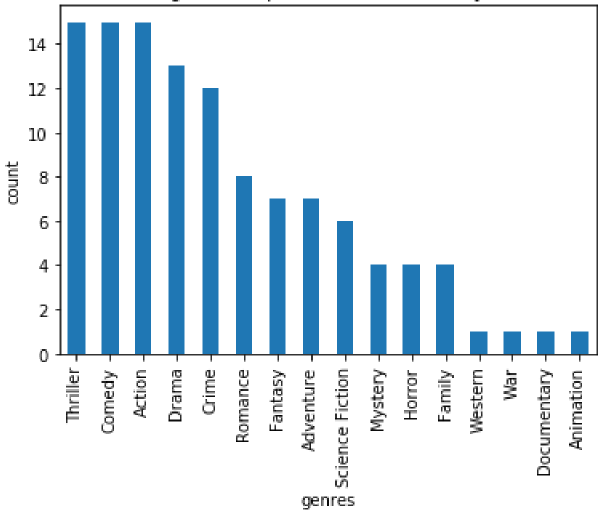


Count of genres in profitable movies for year 2002

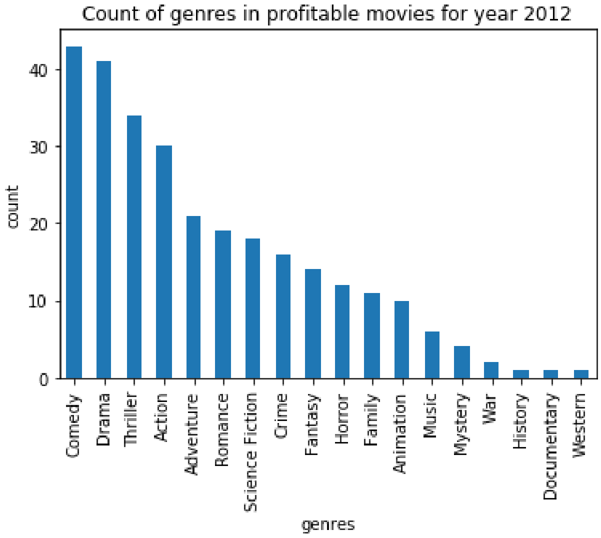




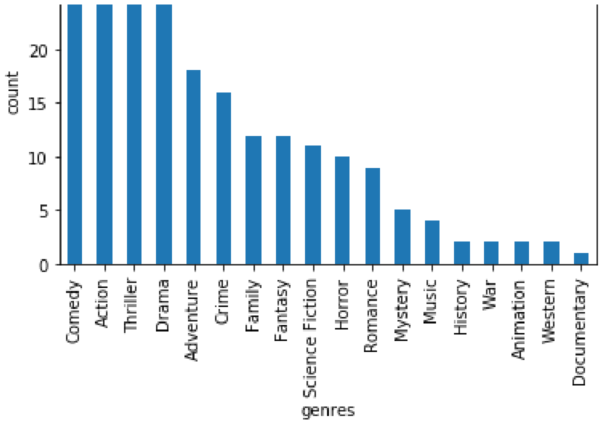
Count of genres in profitable movies for year 1994



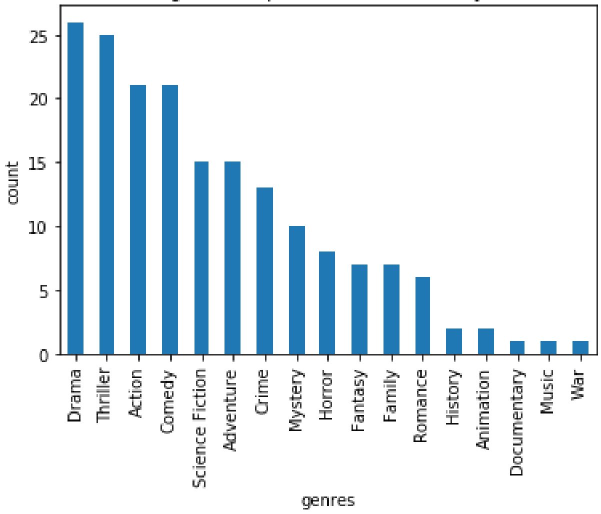
Count of genres in profitable movies for year 2012



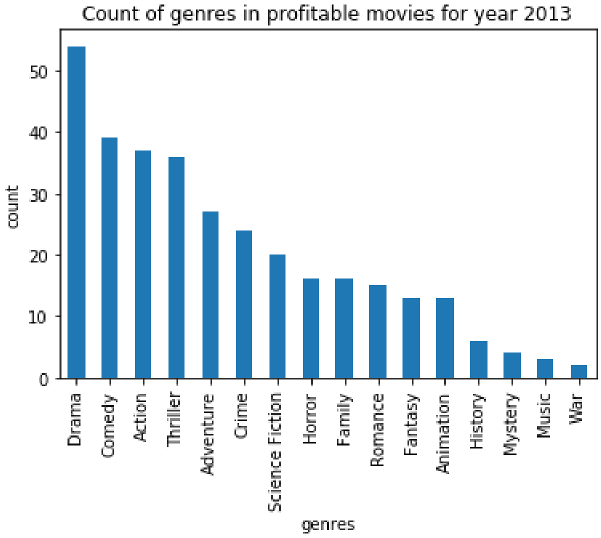
Count of genres in profitable movies for year 2003



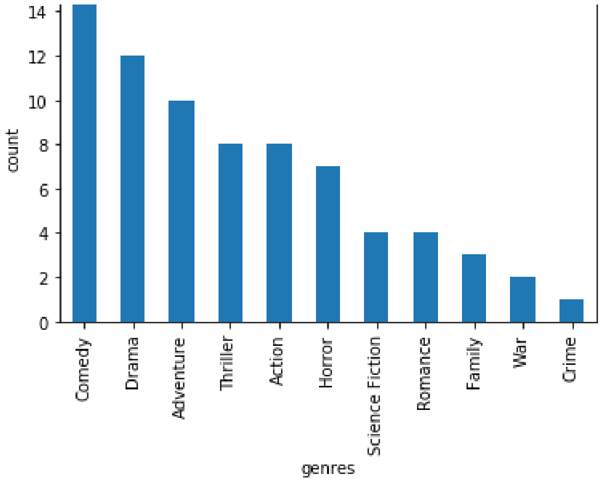
Count of genres in profitable movies for year 1997

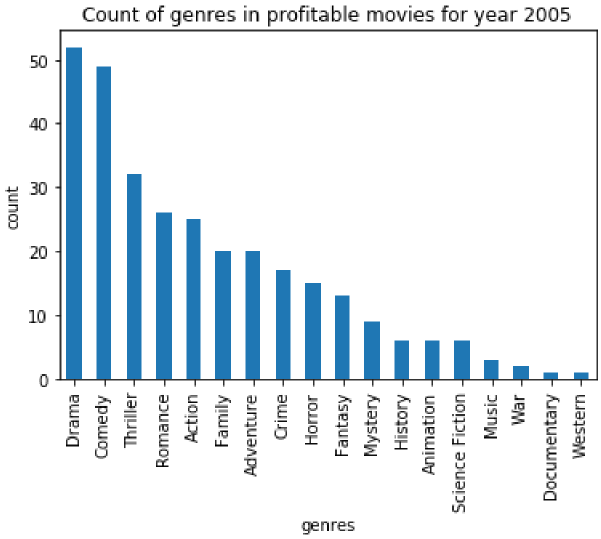


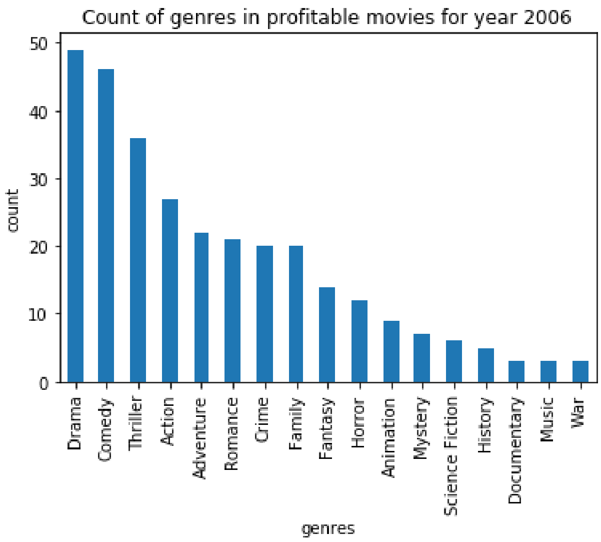
Count of genres in profitable movies for year 2013

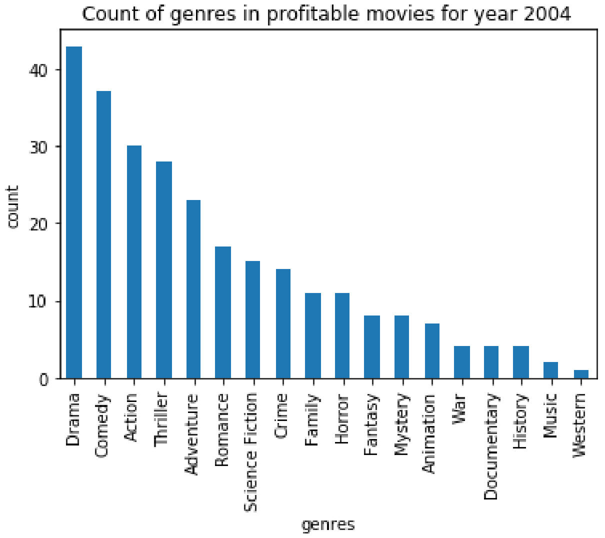


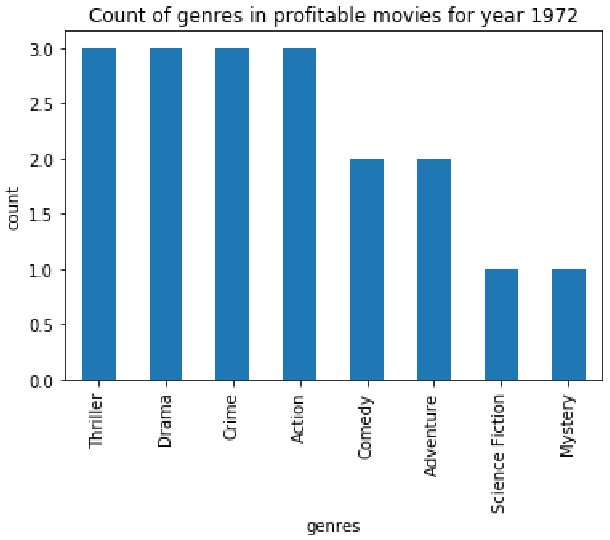
Count of genres in profitable movies for year 1985

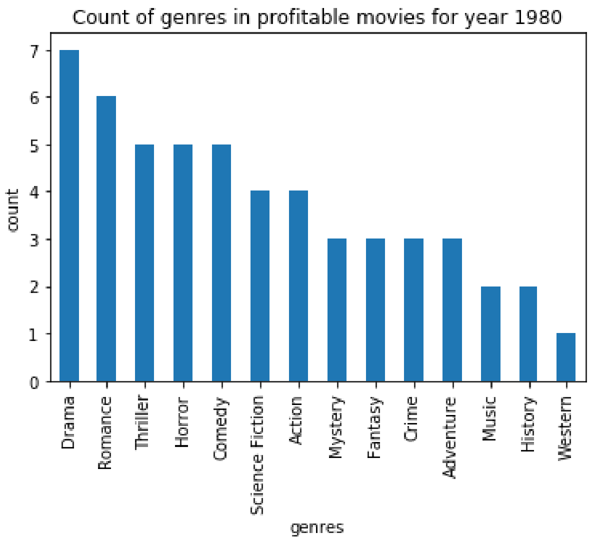

count
genres
Comedy
Drama
Adventure
Thriller
Action
Horror
Science Fiction
Romance
Family
War
Crime

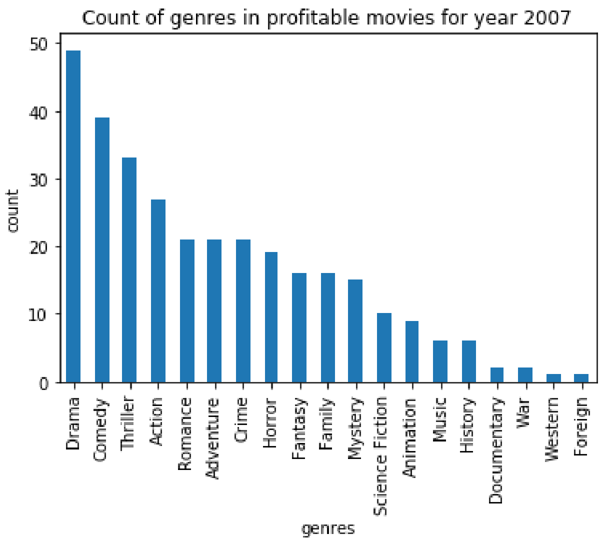

Count of genres in profitable movies for year 2005
count
genres
Drama
Comedy
Thriller
Romance
Action
Family
Adventure
Crime
Horror
Fantasy
Mystery
History
Animation
Science Fiction
Music
War
Documentary
Western


Count of genres in profitable movies for year 2006
count
genres
Drama
Comedy
Thriller
Action
Adventure
Romance
Crime
Family
Fantasy
Horror
Animation
Mystery
Science Fiction
History
Documentary
Music
War

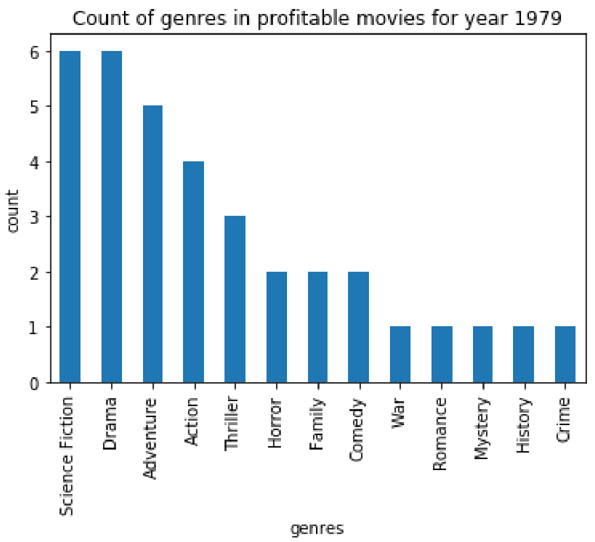

Count of genres in profitable movies for year 2004
count
genres
Drama
Comedy
Action
Thriller
Adventure
Romance
Science Fiction
Crime
Family
Horror
Fantasy
Mystery
Animation
War
Documentary
History
Music
Western


Count of genres in profitable movies for year 1972
count
genres
Thriller
Drama
Crime
Action
Comedy
Adventure
Science Fiction
Mystery

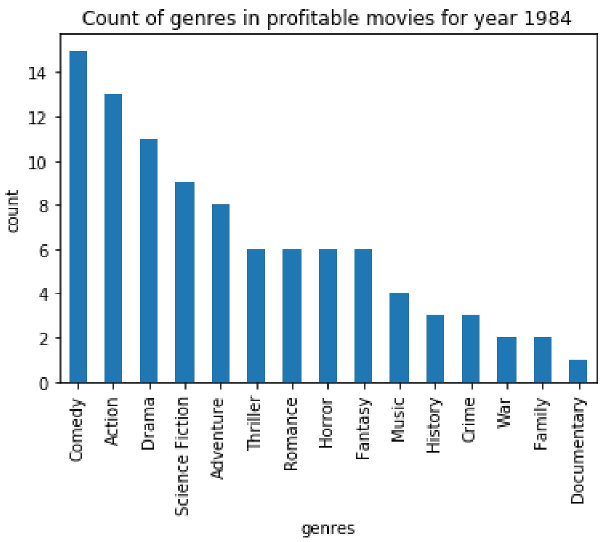

Count of genres in profitable movies for year 1980
count
genres
Drama
Romance
Thriller
Horror
Comedy
Science Fiction
Action
Mystery
Fantasy
Crime
Adventure
Music
History
Western



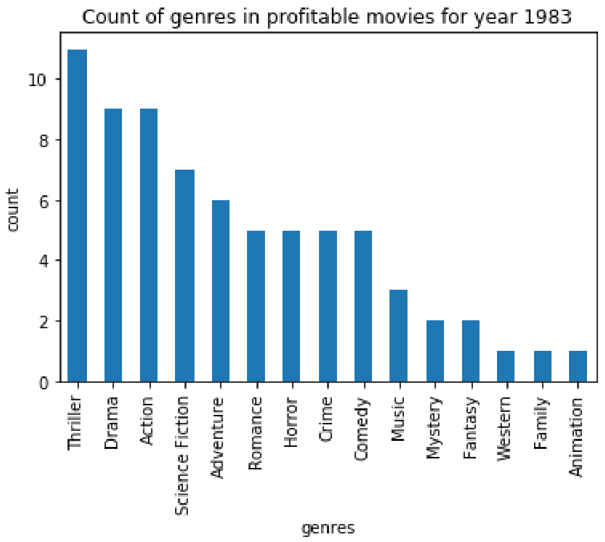
Count of genres in profitable movies for year 2007
count
50
40
30
20
10
0
Drama
Comedy
Thriller
Action
Romance
Adventure
Crime
Horror
Fantasy
Family
Mystery
Science Fiction
Animation
Music
History
Documentary
War
Western
Foreign
genres

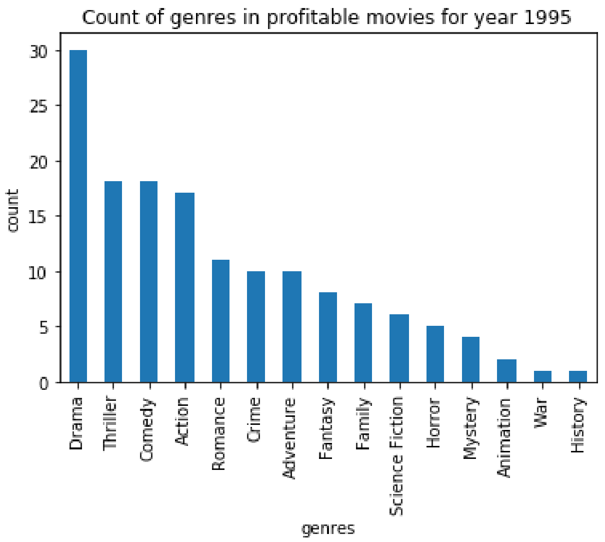


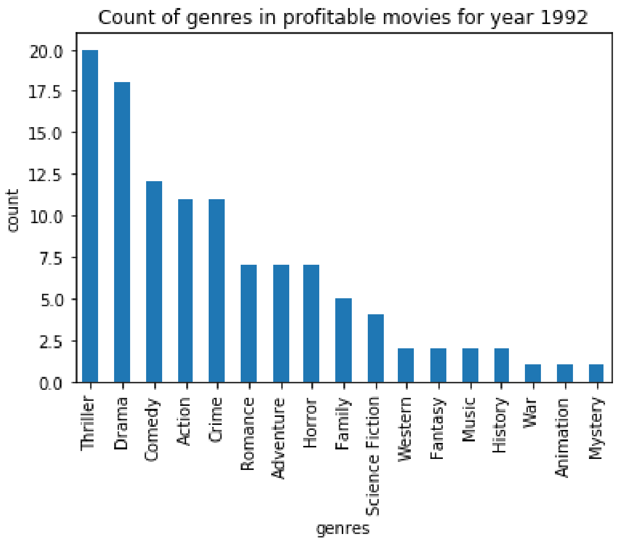
Count of genres in profitable movies for year 1979
count
6
5
4
3
2
1
0
Science Fiction
Drama
Adventure
Action
Thriller
Horror
Family
Comedy
War
Romance
Mystery
History
Crime
genres

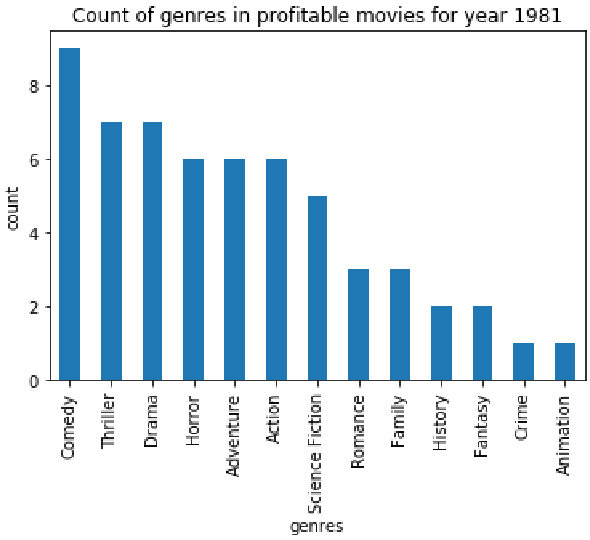


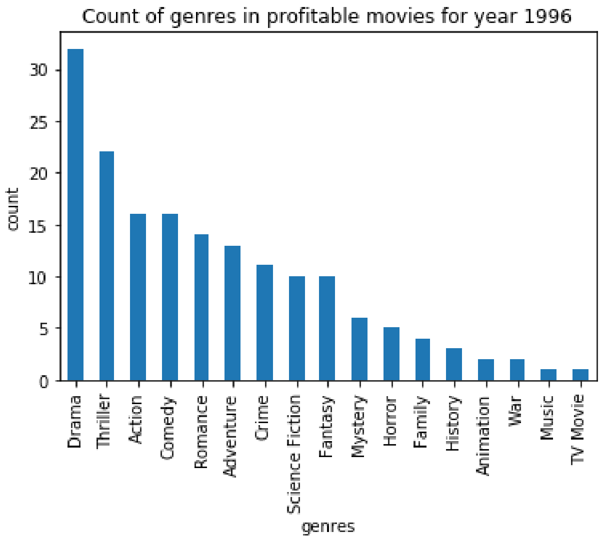
Count of genres in profitable movies for year 1984
count
14
12
10
8
6
4
2
0
Comedy
Action
Drama
Science Fiction
Adventure
Thriller
Romance
Horror
Fantasy
Music
History
Crime
War
Family
Documentary
genres

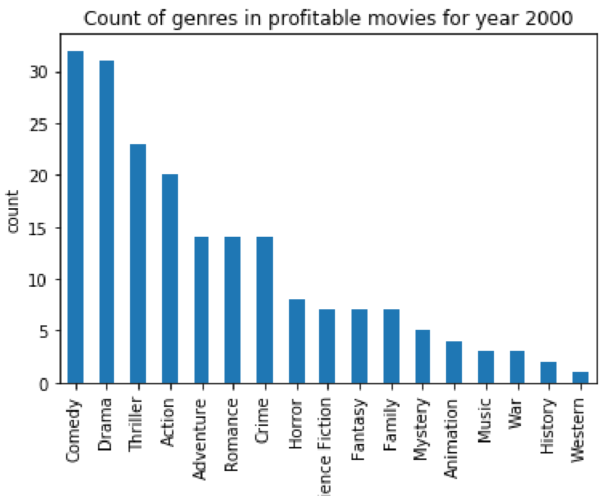

Count of genres in profitable movies for year 1983
count
0
2
4
6
8
10
Thriller
Drama
Action
Science Fiction
Adventure
Romance
Horror
Crime
Comedy
Music
Mystery
Fantasy
Western
Family
Animation
genres


Count of genres in profitable movies for year 1995
count
0
5
10
15
20
25
30
Drama
Thriller
Comedy
Action
Romance
Crime
Adventure
Fantasy
Family
Science Fiction
Horror
Mystery
Animation
War
History
genres

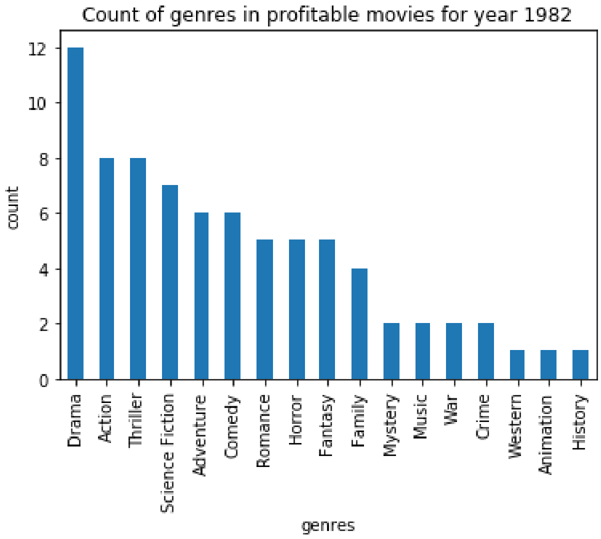

Count of genres in profitable movies for year 1992
count
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Thriller
Drama
Comedy
Action
Crime
Romance
Adventure
Horror
Family
Science Fiction
Western
Fantasy
Music
History
War
Animation
Mystery
genres

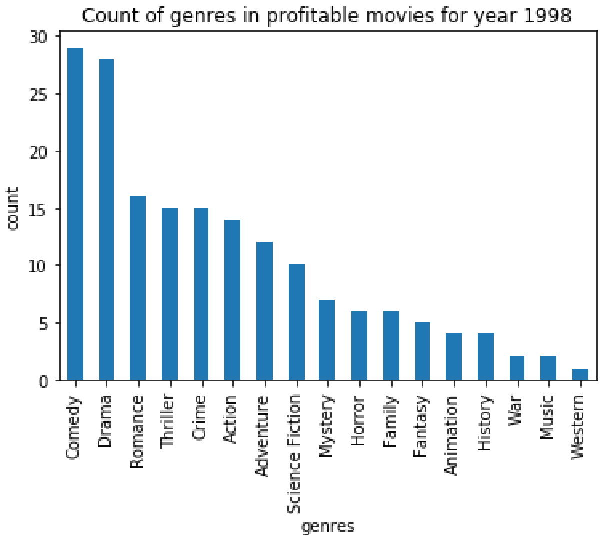

Count of genres in profitable movies for year 1981
count
0
2
4
6
8
Comedy
Thriller
Drama
Horror
Adventure
Action
Science Fiction
Romance
Family
History
Fantasy
Crime
Animation
genres

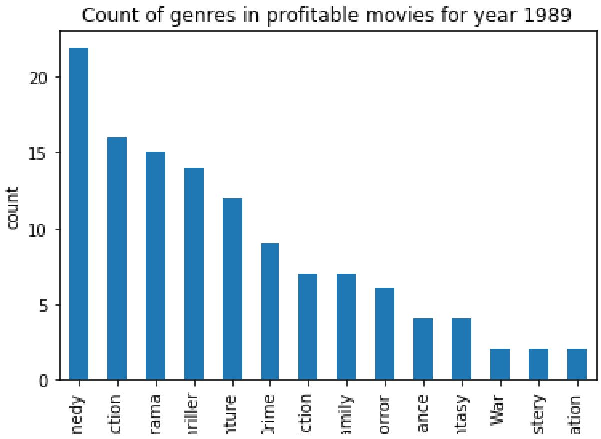

Count of genres in profitable movies for year 1996
count
0
5
10
15
20
25
30
Drama
Thriller
Action
Comedy
Romance
Adventure
Crime
Science Fiction
Fantasy
Mystery
Horror
Family
History
Animation
War
Music
TV Movie
genres

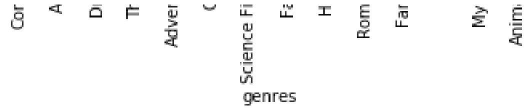

Count of genres in profitable movies for year 2000
count
0
5
10
15
20
25
30
Comedy
Drama
Thriller
Action
Adventure
Romance
Crime
Horror
Fantasy
Family
Mystery
Animation
Music
War
History
Western

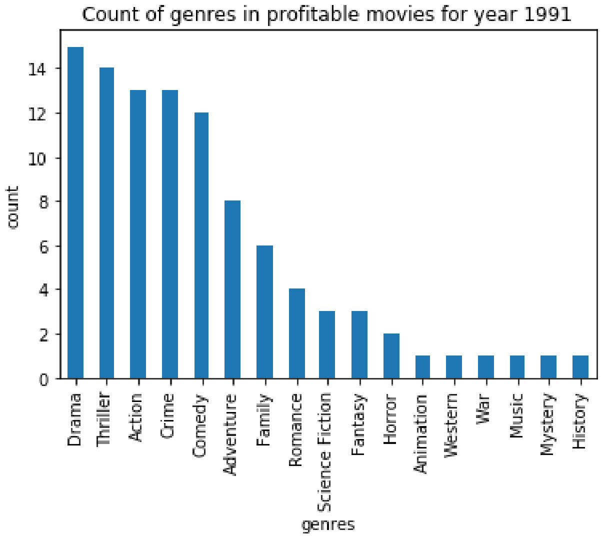

Sci
genres

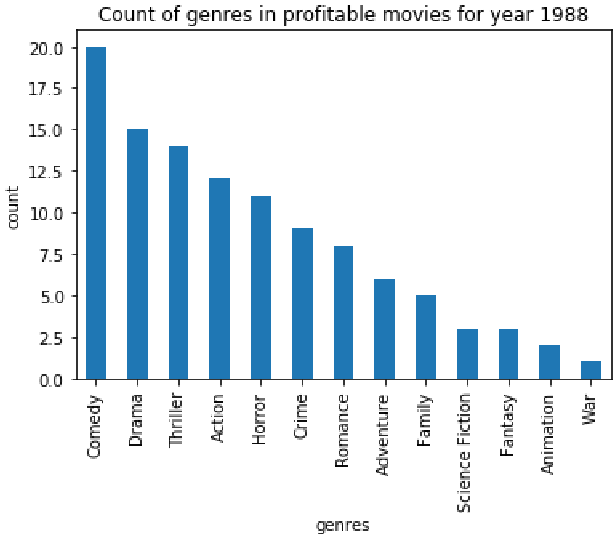

Count of genres in profitable movies for year 1982
count
0
2
4
6
8
10
12
Drama
Action
Thriller
Science Fiction
Adventure
Comedy
Romance
Horror
Fantasy
Family
Mystery
Music
War
Crime
Western
Animation
History
genres

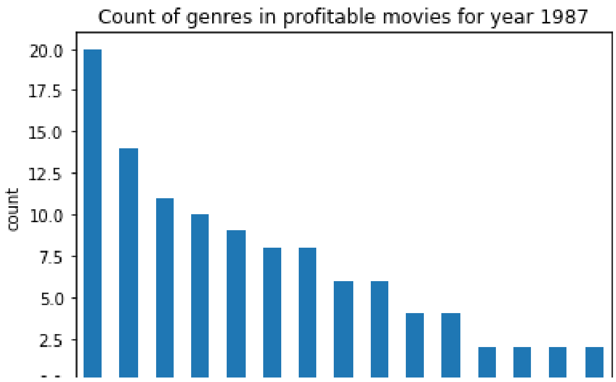

Count of genres in profitable movies for year 1998
count
0
5
10
15
20
25
30
Comedy
Drama
Romance
Thriller
Crime
Action
Adventure
Science Fiction
Mystery
Horror
Family
Fantasy
Animation
History
War
Music
Western
genres

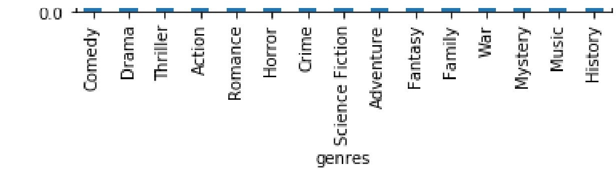

Count of genres in profitable movies for year 1989
count
0
5
10
15
20
War

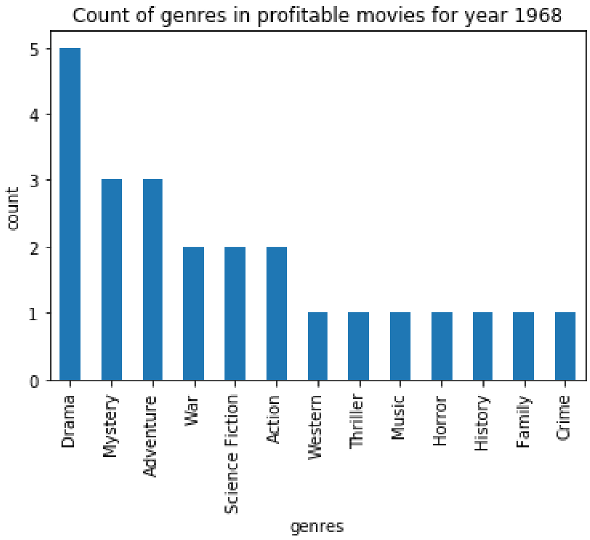

Cor
Adver
Science Fi
Rom
Far
My
Anim
genres

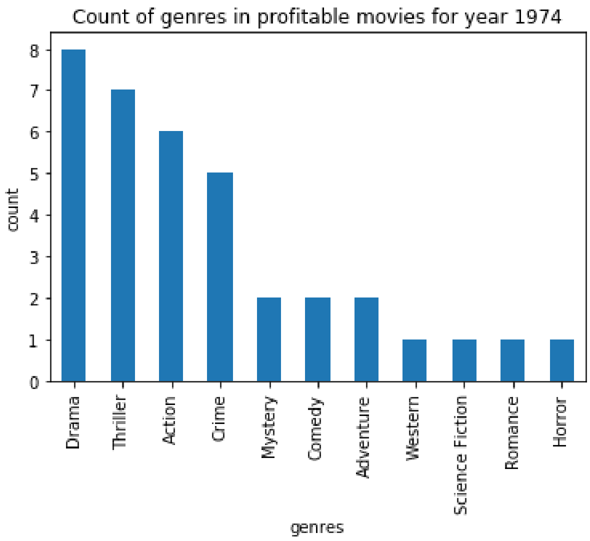

Count of genres in profitable movies for year 1991
count
genres
0
2
4
6
8
10
12
14
Drama
Thriller
Action
Crime
Comedy
Adventure
Family
Romance
Science Fiction
Fantasy
Horror
Animation
Western
War
Music
Mystery
History

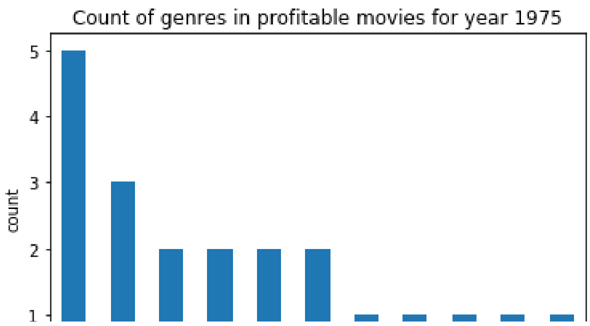

Count of genres in profitable movies for year 1988
count
genres
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
Comedy
Drama
Thriller
Action
Horror
Crime
Romance
Adventure
Family
Science Fiction
Fantasy
Animation
War

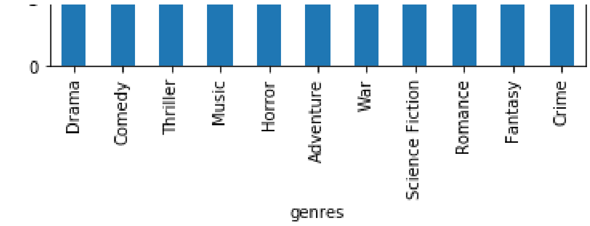

Count of genres in profitable movies for year 1987
count
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0

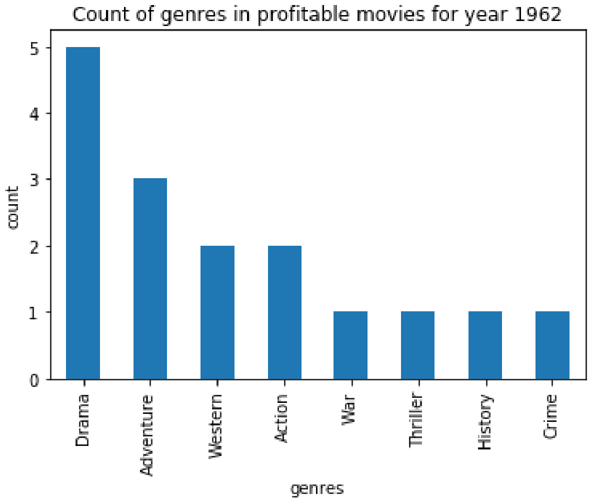

0.0
Comedy
Drama
Thriller
Action
Romance
Horror
Crime
Science Fiction
Adventure
Fantasy
Family
War
Mystery
Music
History
genres

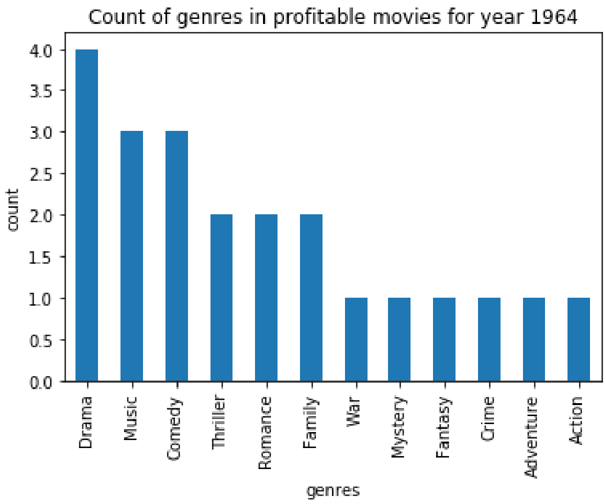

Count of genres in profitable movies for year 1968
count
0
1
2
3
4
5
Drama
Mystery
Adventure
War
Science Fiction
Action
Western
Thriller
Music
Horror
History
Family
Crime
genres

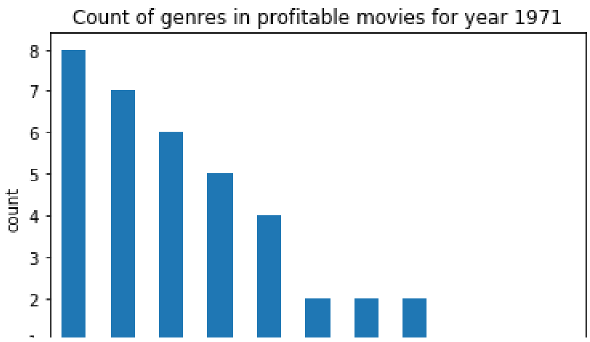

Count of genres in profitable movies for year 1974
count
0
1
2
3
4
5
6
7
8
Drama
Thriller
Action
Crime
Mystery
Comedy
Adventure
Western
Science Fiction
Romance
Horror
genres

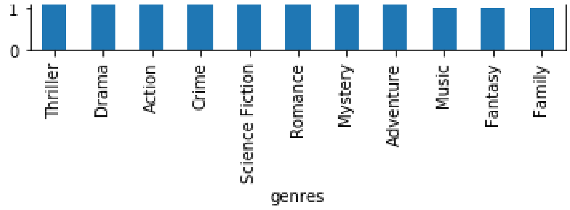

Count of genres in profitable movies for year 1975
count
1
2
3
4
5


0
Drama
Comedy
Thriller
Music
Horror
Adventure
War
Science Fiction
Romance
Fantasy
Crime
genres

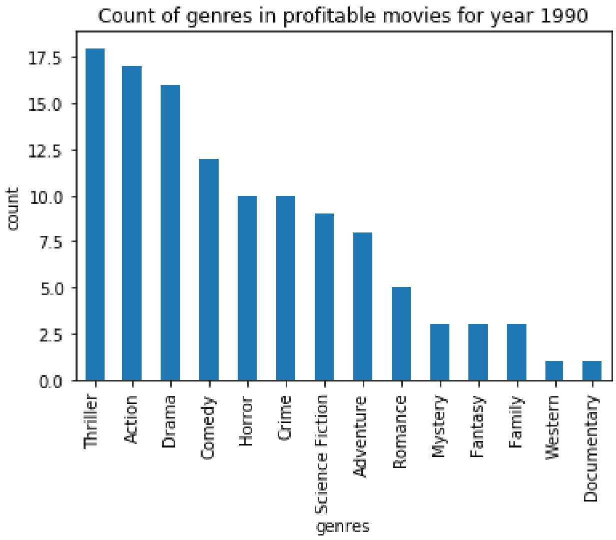

Count of genres in profitable movies for year 1962
count
0
1
2
3
4
5
Drama
Adventure
Western
Action
War
Thriller
History
Crime
genres


Count of genres in profitable movies for year 1964
count
0.0
0.5
1.0
1.5
2.0
2.5
3.0
3.5
4.0
Drama
Music
Comedy
Thriller
Romance
Family
War
Mystery
Fantasy
Crime
Adventure
Action
genres

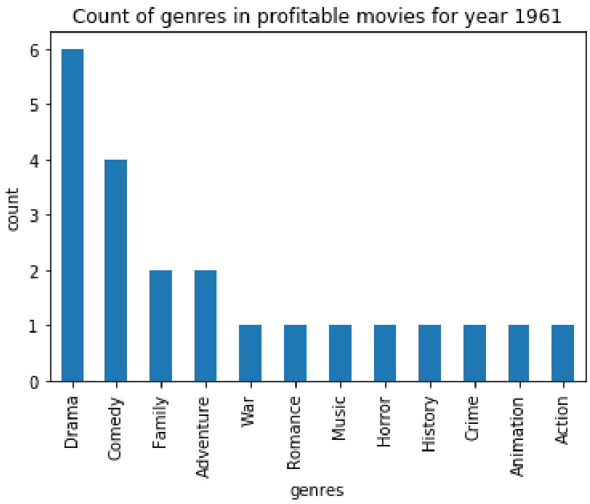

Count of genres in profitable movies for year 1971
count
2
3
4
5
6
7
8



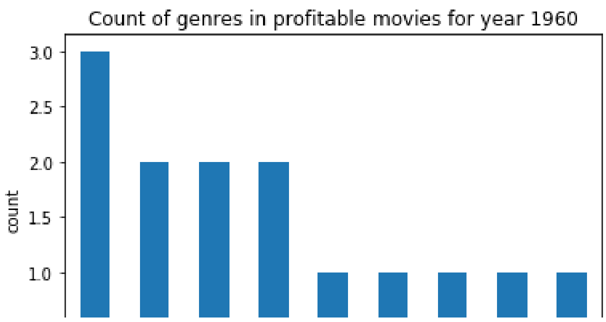
1
0
Thriller
Drama
Action
Crime
Science Fiction
Romance
Mystery
Adventure
Music
Fantasy
Family
genres



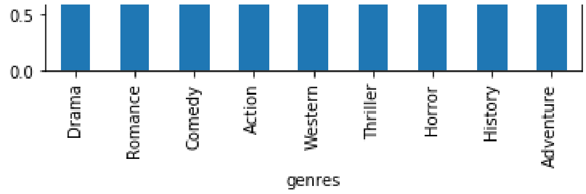
Count of genres in profitable movies for year 1990
count
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
Thriller
Action
Drama
Comedy
Horror
Crime
Science Fiction
Adventure
Romance
Mystery
Fantasy
Family
Western
Documentary
genres

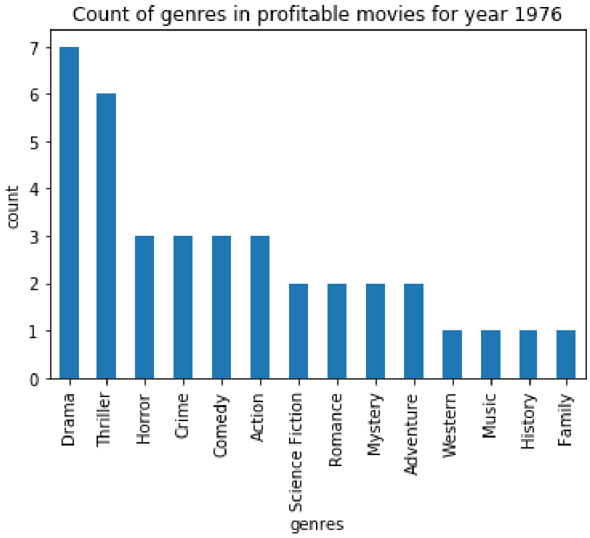


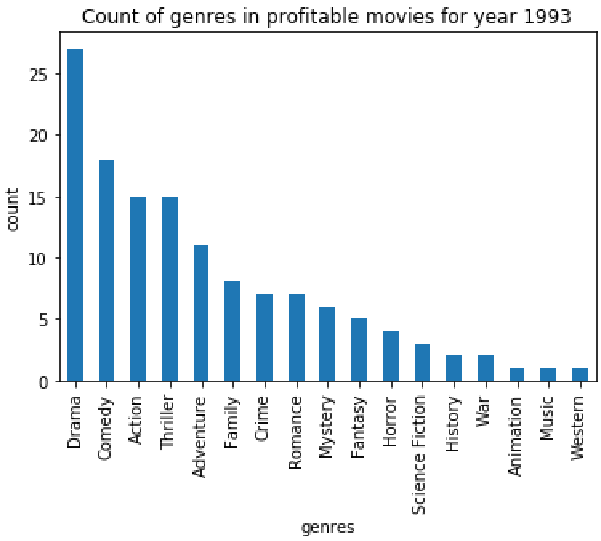
Count of genres in profitable movies for year 1961
count
6
5
4
3
2
1
0
Drama
Comedy
Family
Adventure
War
Romance
Music
Horror
History
Crime
Animation
Action
genres

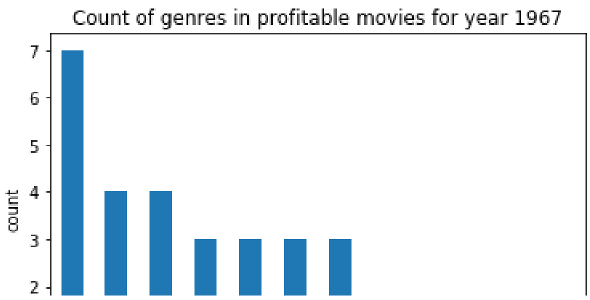


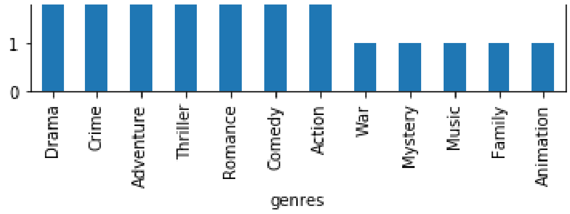
Count of genres in profitable movies for year 1960
count
3.0
2.5
2.0
1.5
1.0


0.5
0.0
Drama
Romance
Comedy
Action
Western
Thriller
Horror
History
Adventure
genres

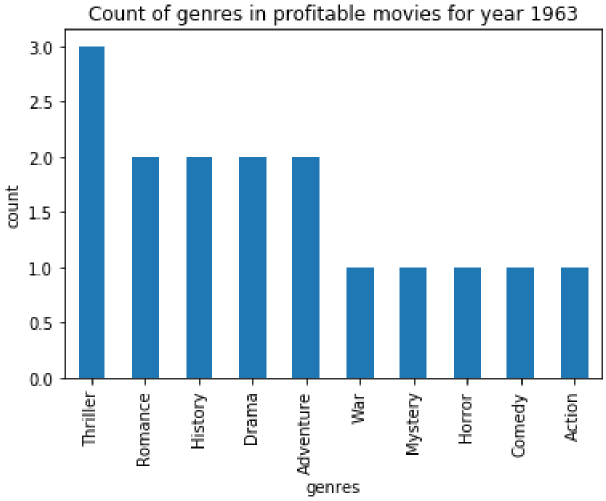

Count of genres in profitable movies for year 1976
count
genres
Drama
Thriller
Horror
Crime
Comedy
Action
Science Fiction
Romance
Mystery
Adventure
Western
Music
History
Family


Count of genres in profitable movies for year 1993
count
genres
Drama
Comedy
Action
Thriller
Adventure
Family
Crime
Romance
Mystery
Fantasy
Horror
Science Fiction
History
War
Animation
Music
Western

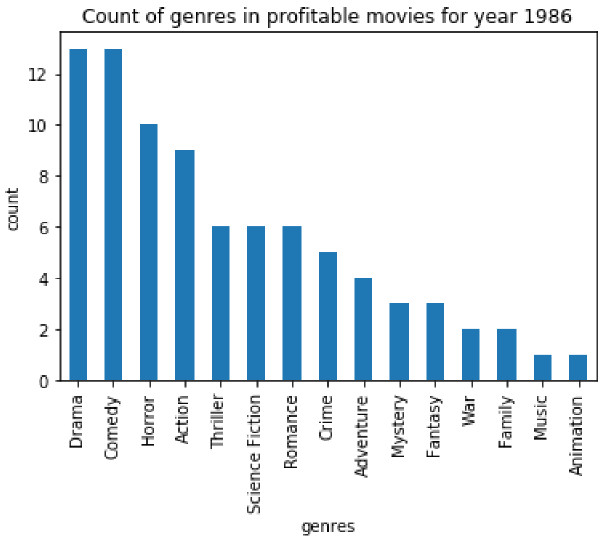

Count of genres in profitable movies for year 1967
count



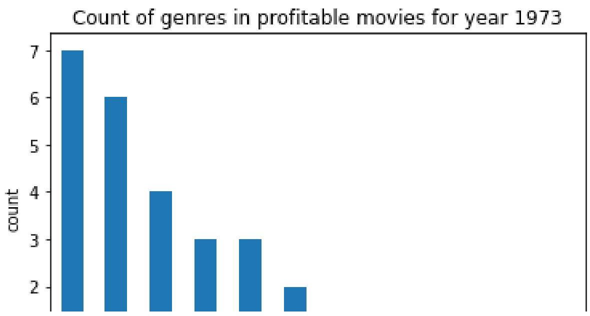
1
0
Drama
Crime
Adventure
Thriller
Romance
Comedy
Action
War
Mystery
Music
Family
Animation
genres



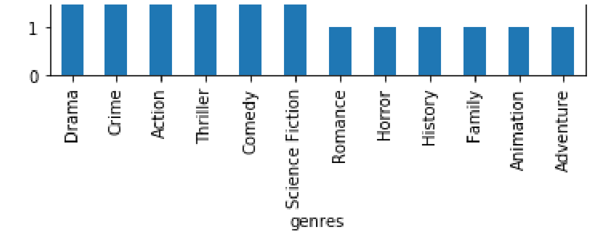
Count of genres in profitable movies for year 1963
count
0.0
0.5
1.0
1.5
2.0
2.5
3.0
Thriller
Romance
History
Drama
Adventure
War
Mystery
Horror
Comedy
Action
genres



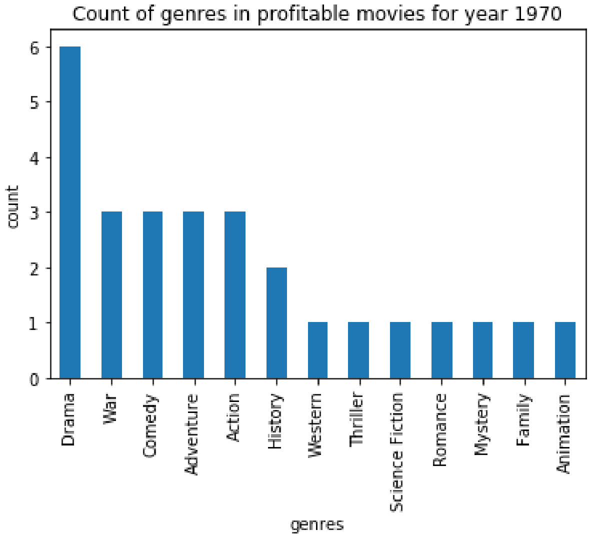
Count of genres in profitable movies for year 1986
count
0
2
4
6
8
10
12
Drama
Comedy
Horror
Action
Thriller
Science Fiction
Romance
Crime
Adventure
Mystery
Fantasy
War
Family
Music
Animation
genres



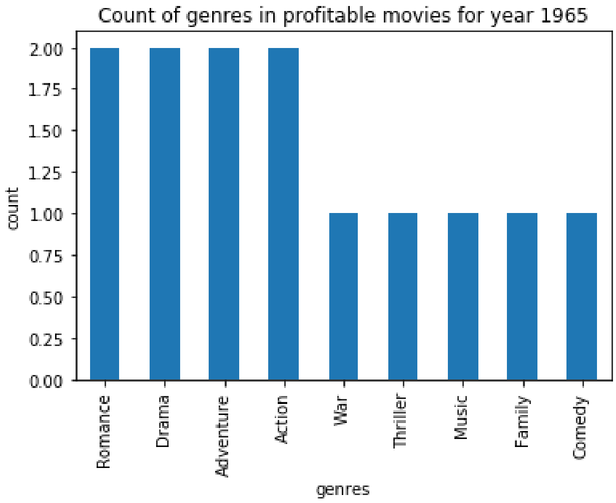
Count of genres in profitable movies for year 1973
count
2
3
4
5
6
7



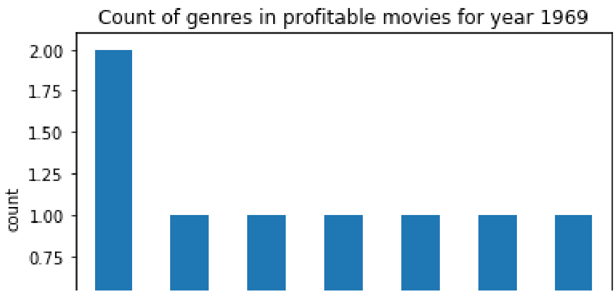
1
0
Drama
Crime
Action
Thriller
Comedy
Science Fiction
Romance
Horror
History
Family
Animation
Adventure
genres



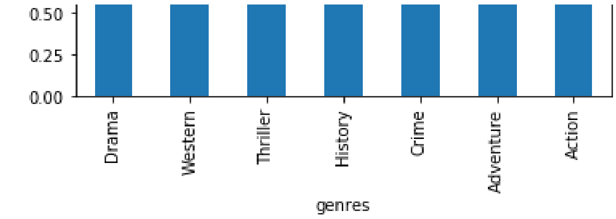
Count of genres in profitable movies for year 1970
count
0
1
2
3
4
5
6
Drama
War
Comedy
Adventure
Action
History
Western
Thriller
Science Fiction
Romance
Mystery
Family
Animation
genres

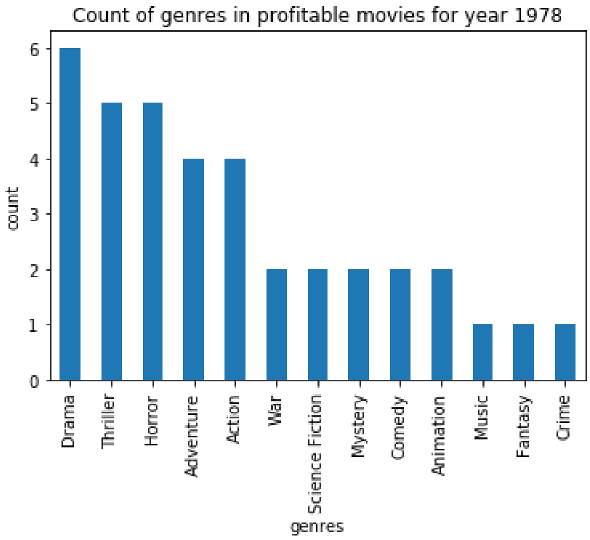


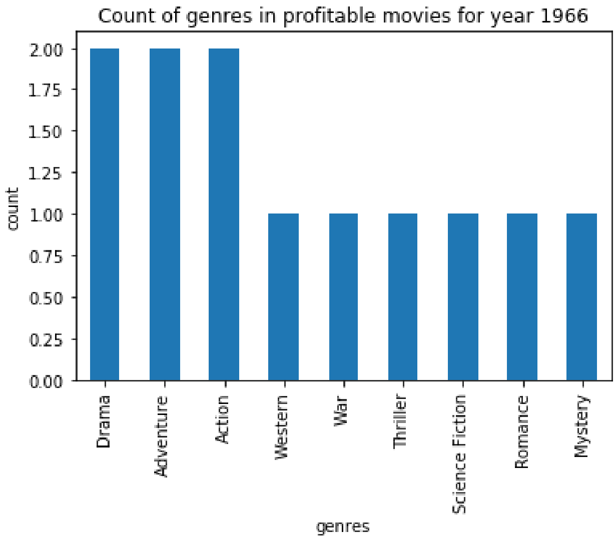
Count of genres in profitable movies for year 1965
count
0.00
0.25
0.50
0.75
1.00
1.25
1.50
1.75
2.00
Romance
Drama
Adventure
Action
War
Thriller
Music
Family
Comedy
genres



Count of genres in profitable movies for year 1969
count
0.75
1.00
1.25
1.50
1.75
2.00



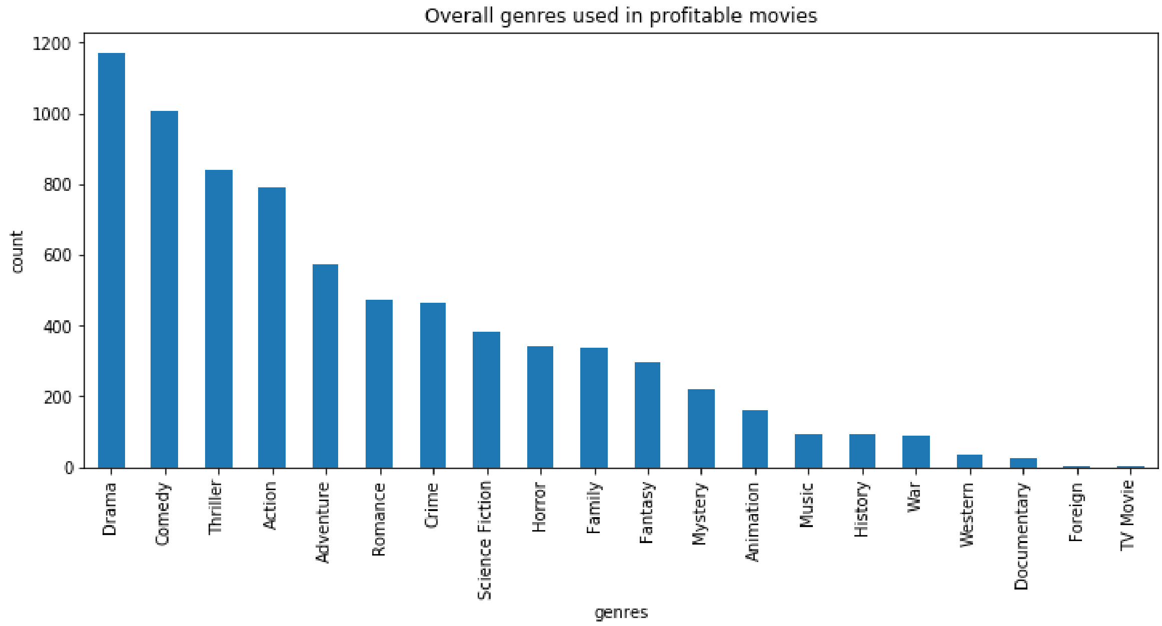




step 4: Now we will plot combined bar chart for all genres used in profitable movies in every year.

In [40]:

```
df_profitable_movies['genres'].str.get_dummies(sep='|').sum().sort_values(ascending=False).plot(kin
d='bar',figsize=(12,5));
plt.xlabel('genres');
plt.ylabel('count');
plt.title('Overall genres used in profitable movies');
```



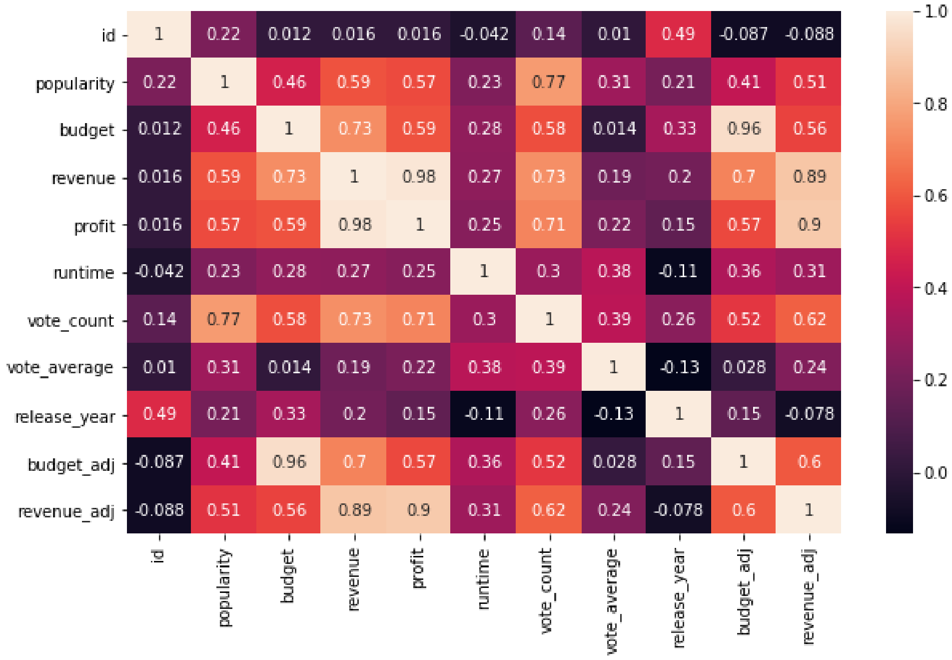
- **We can see clearly, Most of the profitable movies are based on the genre Drama followed by Comedy, Thriller and Action.**

Research Question 4: What are the properties associated with profitable movies?

step 1: Let's analyze correlation between profit and other features.

In [566]:

```
plt.subplots(figsize=(10,6))
sn.heatmap(df_profitable_movies.corr(),annot=True); #plotting heatmap to visualize graphically.
```



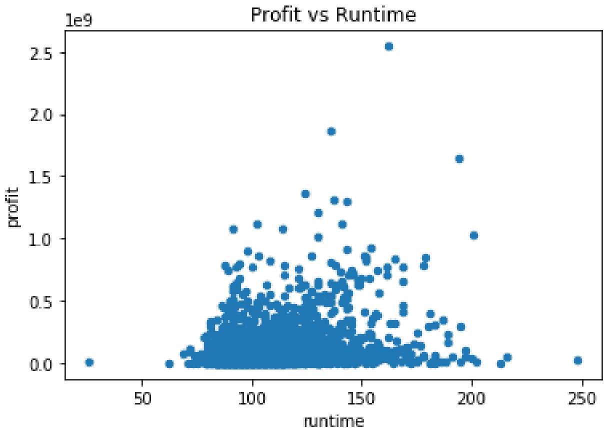
- **By plotting heatmap, we can clearly see there is a positive correlation of profit with features like runtime, budget and popularity.**

step 2: Let's analyze each of them one by one.

- **profit vs runtime**

In [41]:

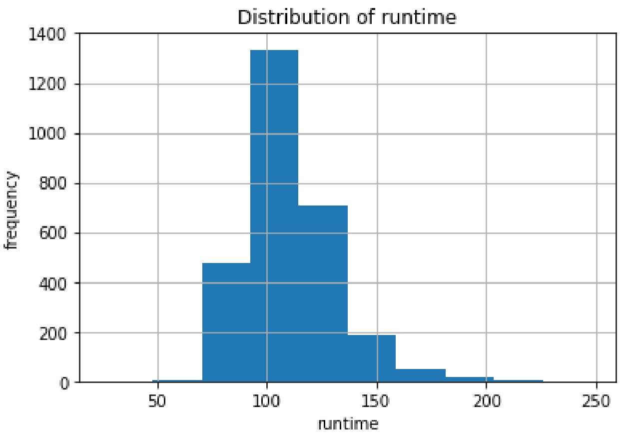
```
df_profitable_movies[['profit','runtime']].plot(kind='scatter',x='runtime',y='profit'); #plotting scatter plot
plt.title('Profit vs Runtime');
```



- **Most of the profitable movies have runtime greater than 50 mins but less than 200 mins. Only few movies are having runtime greater than 200 mins while only 1 movie having runtime less than 50 mins.**

In [42]:

```
df_profitable_movies['runtime'].hist();
plt.xlabel('runtime');
plt.ylabel('frequency');
plt.title('Distribution of runtime');
```



- **Most of the durations of profitable movies lie between 90 to 150 mins. Distribution is right or positive skewed.**

In [569]:

```
df_profitable_movies.runtime.mean()
```

Out[569]:

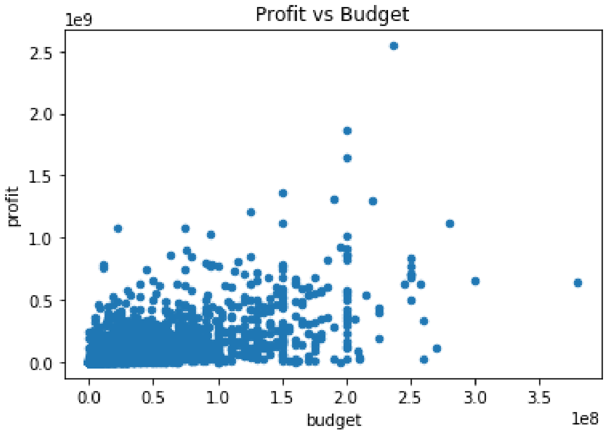
```
110.16750720461096
```

Average duration of profitable movie is 110 mins i.e. around (2 hours).

- **profit vs budget**

In [43]:

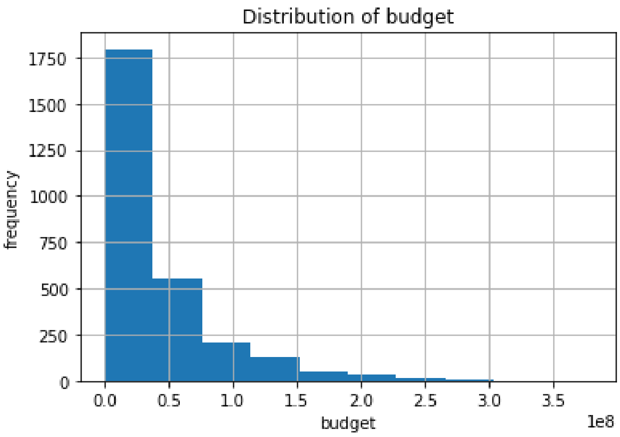
```
df_profitable_movies.plot(kind='scatter',x='budget',y='profit'); #plotting scatter plot
plt.title('Profit vs Budget');
```



- **Most of the movies are with budget range under 200,000,000 (20 crore).**

In [44]:

```
df_profitable_movies['budget'].hist();
plt.xlabel('budget');
plt.ylabel('frequency');
plt.title('Distribution of budget');
```



- **Budget for most of the porfitable movies lies between 1,00,00,000 (1 crore) to 120,000,000 (12 crore).**

In [572]:

```
df_profitable_movies.budget.mean()
```

Out[572]:

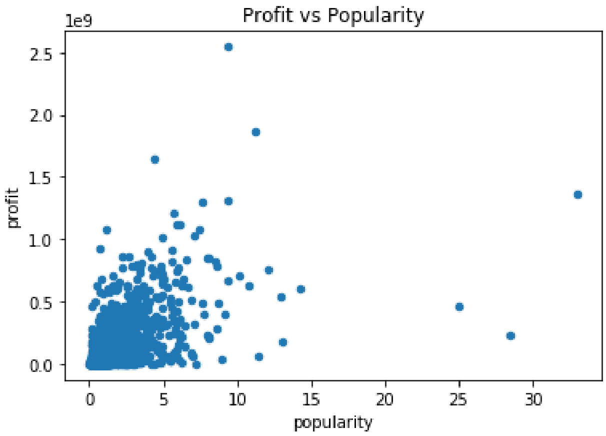
```
40301757.668227665
```

Average budget for profitable movie is around 4 crore.

- **profit vs popularity**

In [45]:

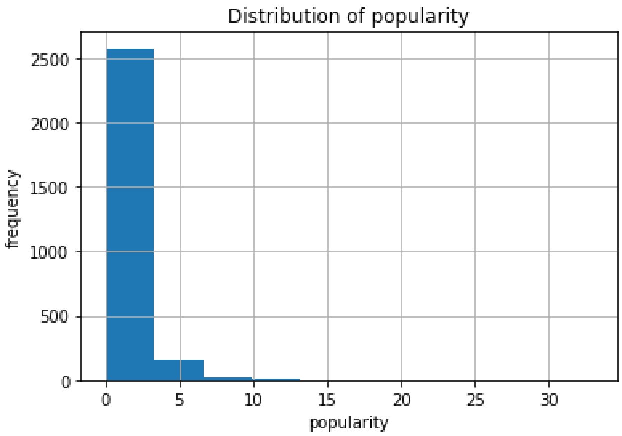
```
df_profitable_movies.plot(kind='scatter',x='popularity',y='profit'); #plotting scatter plot
plt.title('Profit vs Popularity');
```



- **Most of the popularity scores are between 0 to 10.**

In [46]:

```
df_profitable_movies['popularity'].hist();
plt.xlabel('popularity');
plt.ylabel('frequency');
plt.title('Distribution of popularity');
```



- **Now we can clearly see, popularity score for most of the profitable movies lies between 0 to 3.**

In [575]:

```
df_profitable_movies['popularity'].mean()
```

Out[575]:

```
1.4146774859510078
```

Average popularity score is 1.41.

step 3: Let's find cast associated with profitable movies.

In [48]:

```
profitable_movie_cast = df_profitable_movies['cast'].str.get_dummies(sep="|")
```

In [49]:

```
profitable_movie_cast.sum().sort_values(ascending=False)[:10] #Printing top 10 Artists associated
with porfitable movies.
```

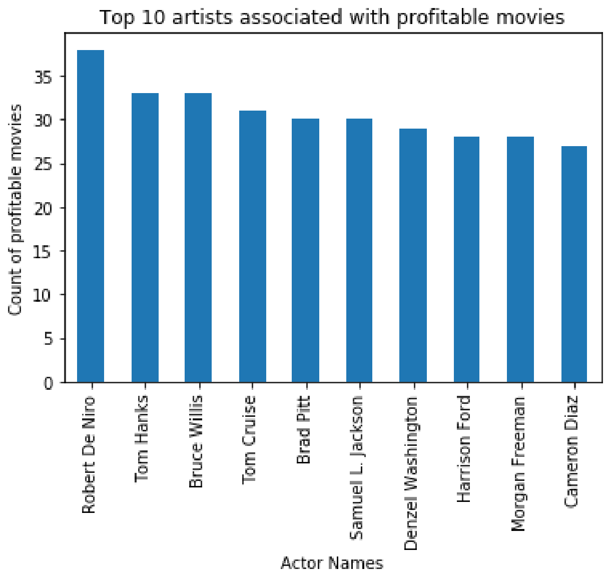
Out[49]:

```
Robert De Niro        38
Tom Hanks             33
Bruce Willis          33
Tom Cruise            31
Brad Pitt             30
Samuel L. Jackson     30
Denzel Washington     29
Harrison Ford         28
Morgan Freeman        28
Cameron Diaz          27
dtype: int64
```

visualizing top 10 artists associated with porfitable movies.

In [52]:

```
profitable_movie_cast.sum().sort_values(ascending=False)[:10].plot(kind='bar');
plt.xlabel('Actor Names')
plt.ylabel('Count of profitable movies')
plt.title('Top 10 artists associated with profitable movies');
```



- **Robert De Niro is an actor having 38 profitable movies followed by Tom Hanks and Bruce Willis both with count 33.**

step 4: Let's find directors associated with profitable movies.

In [55]:

```
profitable_movie_directors = df_profitable_movies['director'].str.get_dummies(sep="|")
```

In [56]:

```
profitable_movie_directors.sum().sort_values(ascending=False)[:10] #Printing top 10 directors asso
ciated with porfitable movies
```

Out[56]:

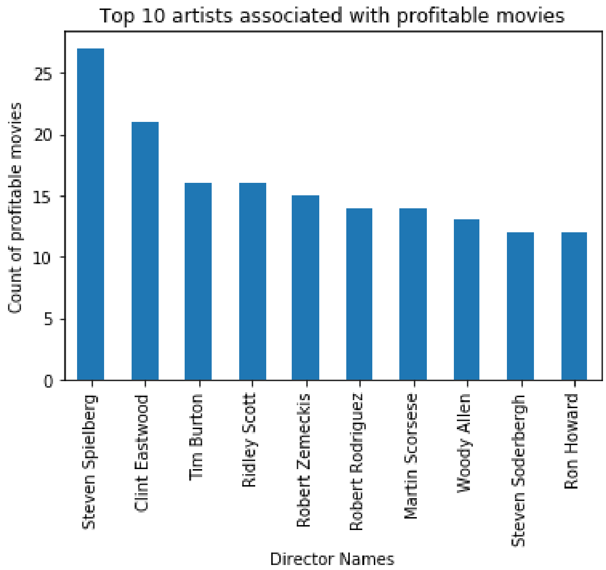
```
Steven Spielberg     27
Clint Eastwood       21
Tim Burton           16
Ridley Scott         16
Robert Zemeckis      15
```

```
Robert Rodriguez      14
Martin Scorsese       14
Woody Allen           13
Steven Soderbergh     12
Ron Howard            12
dtype: int64
```

Visualizing top 10 directors associated with porfitable movies.

In [57]:

```
profitable_movie_directors.sum().sort_values(ascending=False)[:10].plot(kind='bar');
plt.xlabel('Director Names')
plt.ylabel('Count of profitable movies')
plt.title('Top 10 artists associated with profitable movies');
```



- **So we can see, Steven Spielberg is the director having highest profitable movies count (27) while Steven Soderbergh Ron Howard share same profitable movies count i.e. 12.**

Research Question 5: Movies having longest and shortest runtime.

step 1: First let's find movie with longest runtime.

In [582]:

```
df_runtime = df_clean[df_clean.runtime > 0] #since duration cannot be 0 for any movie.
```

In [583]:

```
df_runtime [df_runtime.runtime == df_runtime.runtime.max()]
```

Out[583]:

	id	imdb_id	popularity	budget	revenue	profit	original_title	cast	director	overview	runtime	genres
3894	125336	tt2044056	0.006925	0	0	0	The Story of Film: An Odyssey	Mark Cousins|Jean-Michel Frodon|Cari Beauchamp...	Mark Cousins	The Story of Film: An Odyssey, written and dir...	900	Documentary

So the movie with longest runtime 900 mins is "The Story of Film: An Odyssey". It belongs to the Documentary genre.

step 2: Now let's find movie with shortest runtime.

step 2: Now let's find movie with shortest runtime.

In [584]:

```
df_runtime [df_runtime.runtime == df_runtime.runtime.min()]
```

Out[584]:

	id	imdb_id	popularity	budget	revenue	profit	original_title	cast	director	overview	runtime	
1112	264170	tt3643208	0.202776	0	0	0	Batman: Strange Days	Kevin Conroy|Brian George|Tara Strong	Bruce Timm	Celebrating Batmanâ€™s 75th anniversary, DC En...	3	
2232	55692	tt1791596	0.267950	0	0	0	Scrat's Continental Crack-Up	Chris Wedge|Simon Pegg	Steve Martino|Mike Thurmeier	You may think you know the history of continen...	3	
2830	26840	tt0307461	0.254157	0	0	0	Shrek in the Swamp Karaoke Dance Party	Mike Myers|Eddie Murphy	Vicky Jenson|Andrew Adamson	Shrek and his friends enjoy themselves with so...	3	
3298	140656	tt1315885	0.152615	0	0	0	MamÃ¡	Victoria Harris|Irma Monroig|Berta Ros	Andy Muschietti	Little Victoria is waken up by her sister Lili...	3	
3350	105759	tt1430144	0.037628	0	0	0	The Black Hole	Napoleon Ryan	Philip Sansom|Olly Williams	Charlie, a sleep-deprived office worker accide...	3	
3891	98857	tt2115386	0.028803	0	0	0	Scrat's Continental Crack-Up: Part 2	Chris Wedge	Steve Martino|Mike Thurmeier	This short film continues the adventures of th...	3	Anin
5399	43629	tt0411302	0.168542	0	0	0	Doodlebug	Jeremy Theobald	Christopher Nolan	A man is trying to catch some sort of bug runn...	3	
5993	259761	tt3605002	0.039953	0	0	0	Lights Out	Lotta Losten	David F. Sandberg	A woman prepares for bed, but realizes that so...	3	
8706	13930	tt0248808	0.811101	0	0	0	For the Birds	Ralph Eggleston	Ralph Eggleston	One by one, a flock of small birds perches on ...	3	A

So we have total 9 movies with shortest runtime i.e 3 mins.

Research Question 6: Best Month to Release a Movie.

we can answer this question by analyzing the release date of profitable movies.

step 1: Get release month from release date of each profitable movies

In [59]:

```
#adding new column release_month from release_date for each profitable movie.

df_profitable_movies.insert(13,'release_month' ,df_profitable_movies['release_date'].apply(lambda x
: x.month))
```

step 2: Now simply get count of each month.

In [60]:

```
release_months = df_profitable_movies['release_month'].value_counts().sort_index()
release_months
```

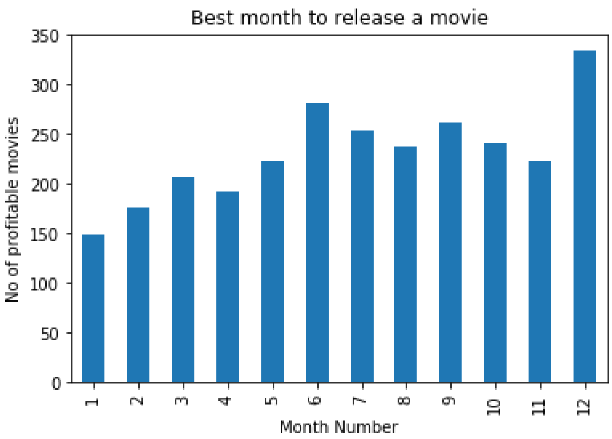
Out[60]:

```
1     148
2     176
3     207
4     192
5     223
6     281
7     253
8     237
9     262
10    240
11    223
12    334
Name: release_month, dtype: int64
```

step 3: Plotting bar chart to visualize graphically.

In [64]:

```
release_months.plot(kind='bar');
plt.xlabel('Month Number');
plt.ylabel('No of profitable movies');
plt.title('Best month to release a movie');
```



- **So it looks like most profitable month for movies is December followed by June.**

Conclusions

After the analysis of the given data set, we can provide information about following questions.

Research Question 1: Most popular movies from year to year.

By plotting bar chart of popular movies for each year, we can say that, for movies released in year 2015, most popular movie was *"Jurasic World"* while *"Psycho"* received more popularity among movies released in 1960.

Research Question 2: Movies with higher ratings from year to year.

By plotting bar chart of movies with higher rating for each year, we can say that, among movies that were released in year 2015, the movie which received highest vote average was once again *"the jinx life and death of robert durst"* while *"Psycho"* received highest vote average among movies released in 1960.

Research Question 3: Which genres are most popular from year to year? (based on profit)

By looking at the chart, we can say, top 4 genres of profitable movies were *Drama, Comedy, Thriller and Action.*

Research Question 4: What are the properties associated with profitable movies?

We have concluded following properties about porfitable movies.
Most of the durations of profitable movies lie between 90 to 150 mins with average duration being 110 mins i.e. around (2 hours).
Budget for most of the porfitable movies lies between 1,00,00,000 (1 crore) to 120,000,000 (12 crore) with average budget is around 4 crore.
Popularity score for most of the profitable movies lies between 0 to 3 with average popularity score being 1.41.
Top 10 Artists associated with profitable movies are following:

> - ***Robert De Niro***
> - ***Tom Hanks***
> - ***Bruce Willis***
> - ***Tom Cruise***
> - ***Brad Pitt***
> - ***Samuel L. Jackson***
> - ***Denzel Washington***
> - ***Harrison Ford***
> - ***Morgan Freeman***
> - ***Cameron Diaz***

Top 10 Directors associated with profitable movies are following:

> - ***Steven Spielberg***
> - ***Clint Eastwood***
> - ***Tim Burton***
> - ***Ridley Scott***
> - ***Robert Zemeckis***
> - ***Robert Rodriguez***
> - ***Martin Scorsese***
> - ***Woody Allen***
> - ***Steven Soderbergh***
> - ***Ron Howard***

Research Question 5: Movies having longest and shortest runtime.

The movie with longest runtime 900 mins is "The Story of Film: An Odyssey". It belongs to the Documentary genre.

On the other hand, we have total 9 movies with shortest runtime i.e 3 mins.

Research Question 6: Best Month to Release a Movie.

December followed by June are two most profitable months

Limitations

All the above presented analysis, insights and visualization about the data are totaly based on the information that we have gathered through the data without performing any statiscal inferences.

For question 1 and 2, we have drawn our conclusion on the basis of popularity score and average votes respectively. Popularity score for the movie "Jurasic World" is much higher than the popularity score for other movies, So, in this situation we might have applied some statistics to discover any outliers or there might be missing data in terms of popularity score for other movies. On the other hand we can see, Average votes are almost same for all the movies.

Question 3,4 and 6 are based on the analysis of profitable movies. Again this is not 100 percent true that if we follow all properties associated with profitable movies while making a movie, then that movie also makes profit, Because in our

dataset there are many movies having budget equal to 0. While analyzing the data, We have dropped all the rows where budget was 0. Movie having budget equal to 0 does not make sense at all because there is always some budget associated for movie, So, technically these are the missing values that need to filled before analyzing the data.