investigate-a-dataset-movie

March 25, 2019

1 Project: Investigate a Dataset The Movie

1.1 Table of Contents

Introduction

In this project I use Movie Data. This dataset contains information about 10,000 movies collected from The Movie Database (TMDb). Contains data such as title, cast, director, runtime, budget, revenue, release year etc. - Certain columns, like 'cast' and 'genres', contain multiple values separated by pipe (|) characters. - The final two columns ending with "_adj" show the budget and revenue of the associated movie in terms of 2010 dollars, accounting for inflation over time.

Research Question Questions in the projects are as follows:

1.1.1 Part 1: General

Which movie earns the highest and lowest profit?
Which movie have the highest and lowest revenue?
Which movie have the highest and lowest budget?
Which movie have the longest and shortest runtime?
How much movie released year by year?
How distribution of profit in different popularity levels in recent ten years?
How distribution of profit in different vote average levels in recent ten years?

1.1.2 Part 2: Find Associate Variable Movie Genre with Movie Metric

What movie genre that associated with popularity level?
What movie genre that associated with revenue level?
What movie genre that associated with vote average level?
What movie genre that associated with profit level?

1.1.3 Part 3: Find Some Trend

What is the trend of the genre every 10 years?
Data Wrangling

In this section I will load the data and print the example of data so I know the data sample value

In [1]:
```
# import packages
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import re
from collections import Counter
%matplotlib inline
```

In [2]:
```
# Load data
df = pd.read_csv("tmdb-movies.csv")
# Print first row to see the example of data
df.head(1)
```

Out[2]:
```
       id    imdb_id  popularity     budget     revenue  original_title  \
0  135397  tt0369610   32.985763  150000000  1513528810  Jurassic World

                                                 cast  \
0  Chris Pratt|Bryce Dallas Howard|Irrfan Khan|Vi...

                        homepage          director             tagline  ...  \
0  http://www.jurassicworld.com/  Colin Trevorrow  The park is open.  ...

                                            overview runtime  \
0  Twenty-two years after the events of Jurassic ...     124

                                      genres  \
0  Action|Adventure|Science Fiction|Thriller

                                production_companies release_date vote_count  \
0  Universal Studios|Amblin Entertainment|Legenda...       6/9/15       5562

   vote_average  release_year    budget_adj   revenue_adj
0           6.5          2015  1.379999e+08  1.392446e+09
```

```
[1 rows x 21 columns]
```

In [3]:
```
# find shape of data
r,c = df.shape
print("Dataset Movie contains %d rows and %d columns" % (r,c))
```

```
Dataset Movie contains 10866 rows and 21 columns
```

1.1.4 Data Cleaning

In this section I will find column that unnecessary for this research, find duplicate data, find missing data, and change some format data that can make this research easier.

1. Find Missing Value and Unnecessary Columns

In [4]:
```
# print the information about count of not null data value and data types
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10866 entries, 0 to 10865
Data columns (total 21 columns):
id                      10866 non-null int64
imdb_id                 10856 non-null object
popularity              10866 non-null float64
budget                  10866 non-null int64
revenue                 10866 non-null int64
original_title          10866 non-null object
cast                    10790 non-null object
homepage                2936 non-null object
director                10822 non-null object
tagline                 8042 non-null object
keywords                9373 non-null object
overview                10862 non-null object
runtime                 10866 non-null int64
genres                  10843 non-null object
production_companies    9836 non-null object
release_date            10866 non-null object
vote_count              10866 non-null int64
vote_average            10866 non-null float64
release_year            10866 non-null int64
budget_adj              10866 non-null float64
revenue_adj             10866 non-null float64
dtypes: float64(4), int64(6), object(11)
memory usage: 1.7+ MB
```

From information above we know that column which missing values are: 1. imdb_id (object) 2. homepage (object) 3. tagline (object) 4. director (object) 5. overview (object) 6. keywords (object) 7. production_companies (object) 8. cast (object) 9. genres (object)

I don't need some columns such as imdb_id, homepage, tagline, and overview. So I will delete them.

1.1 Delete Unnecessary Columns

In [5]:
```
# list of unnecessary columns
col = ['imdb_id', 'homepage', 'tagline', 'overview']

# delete unnecessary columns
df.drop(col,axis=1,inplace=True)
```

In [6]:
```
# print statistical summary from numeric columns to find if any weird value
df.describe()
```

Out[6]:
```
                  id    popularity        budget       revenue       runtime  \
count   10866.000000  10866.000000  1.086600e+04  1.086600e+04  10866.000000
mean    66064.177434      0.646441  1.462570e+07  3.982332e+07    102.070863
std     92130.136561      1.000185  3.091321e+07  1.170035e+08     31.381405
min         5.000000      0.000065  0.000000e+00  0.000000e+00      0.000000
25%     10596.250000      0.207583  0.000000e+00  0.000000e+00     90.000000
50%     20669.000000      0.383856  0.000000e+00  0.000000e+00     99.000000
75%     75610.000000      0.713817  1.500000e+07  2.400000e+07    111.000000
max    417859.000000     32.985763  4.250000e+08  2.781506e+09    900.000000

         vote_count  vote_average  release_year    budget_adj   revenue_adj
count  10866.000000  10866.000000  10866.000000  1.086600e+04  1.086600e+04
mean     217.389748      5.974922   2001.322658  1.755104e+07  5.136436e+07
std      575.619058      0.935142     12.812941  3.430616e+07  1.446325e+08
min       10.000000      1.500000   1960.000000  0.000000e+00  0.000000e+00
25%       17.000000      5.400000   1995.000000  0.000000e+00  0.000000e+00
50%       38.000000      6.000000   2006.000000  0.000000e+00  0.000000e+00
75%      145.750000      6.600000   2011.000000  2.085325e+07  3.369710e+07
max     9767.000000      9.200000   2015.000000  4.250000e+08  2.827124e+09
```

From summary above we found some weird data about budget, revenue, and runtime because the minimal value is zero. Lets output 1 sample of row that have zero number to check that is really zero or just another missing value.

In [7]:
```
# print sample that have budget zero to know is this real zero or not
df.query('budget == 0').head(1)
```

Out[7]:
```
        id  popularity  budget   revenue original_title  \
30  280996    3.927333       0  29355203     Mr. Holmes

                                                   cast     director  \
```

```
30  Ian McKellen|Milo Parker|Laura Linney|Hattie M...  Bill Condon

                        keywords  runtime         genres  \
30  london|detective|sherlock holmes      103  Mystery|Drama

                                  production_companies release_date  \
30  BBC Films|See-Saw Films|FilmNation Entertainme...      6/19/15

    vote_count  vote_average  release_year  budget_adj   revenue_adj
30         425           6.4          2015         0.0  2.700677e+07
```

I found from google that Mr.Holmes budget is 10 million USD, so in that data I assumed zero number is mean missing value. Because of that I will change zero to NA so we can know there is some missing value from function 'info'

1.2 Change Zero Value to NA

In [8]:
```python
# change zero number to NA
zero_col = ['budget', 'revenue', 'runtime','budget_adj', 'revenue_adj']

# replace all zero value from to NAN in the list
df[zero_col] = df[zero_col].replace(0, np.NAN)
```

In [9]:
```python
# see the update info
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10866 entries, 0 to 10865
Data columns (total 17 columns):
id                      10866 non-null int64
popularity              10866 non-null float64
budget                  5170 non-null float64
revenue                 4850 non-null float64
original_title          10866 non-null object
cast                    10790 non-null object
director                10822 non-null object
keywords                9373 non-null object
runtime                 10835 non-null float64
genres                  10843 non-null object
production_companies    9836 non-null object
release_date            10866 non-null object
vote_count              10866 non-null int64
vote_average            10866 non-null float64
release_year            10866 non-null int64
budget_adj              5170 non-null float64
revenue_adj             4850 non-null float64
dtypes: float64(7), int64(3), object(7)
memory usage: 1.4+ MB
```

From updated info we find some columns with small missing value (>= 95% from all row or >= 10322 data) they are cast, director, runtime, and genres. I choose 95% as threshold because I didn't want to delete too much data

1.3 Remove Missing Value

```
In [10]: # list the column that we want to remove missing value
         col_mv = ['cast', 'director', 'runtime', 'genres']

         # remove missing value
         df.dropna(subset=col_mv, inplace=True)

         # see the update information
         df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10704 entries, 0 to 10865
Data columns (total 17 columns):
id                      10704 non-null int64
popularity              10704 non-null float64
budget                  5151 non-null float64
revenue                 4844 non-null float64
original_title          10704 non-null object
cast                    10704 non-null object
director                10704 non-null object
keywords                9294 non-null object
runtime                 10704 non-null float64
genres                  10704 non-null object
production_companies    9760 non-null object
release_date            10704 non-null object
vote_count              10704 non-null int64
vote_average            10704 non-null float64
release_year            10704 non-null int64
budget_adj              5151 non-null float64
revenue_adj             4844 non-null float64
dtypes: float64(7), int64(3), object(7)
memory usage: 1.5+ MB
```

2. Drop Duplicated

```
In [11]: # drop duplicate row in data
         df.drop_duplicates(inplace=True)

         # see the update info
         df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10703 entries, 0 to 10865
```

```
Data columns (total 17 columns):
id                      10703 non-null int64
popularity              10703 non-null float64
budget                  5150 non-null float64
revenue                 4843 non-null float64
original_title          10703 non-null object
cast                    10703 non-null object
director                10703 non-null object
keywords                9293 non-null object
runtime                 10703 non-null float64
genres                  10703 non-null object
production_companies    9759 non-null object
release_date            10703 non-null object
vote_count              10703 non-null int64
vote_average            10703 non-null float64
release_year            10703 non-null int64
budget_adj              5150 non-null float64
revenue_adj             4843 non-null float64
dtypes: float64(7), int64(3), object(7)
memory usage: 1.5+ MB
```

3. Change Data Type

from the update info we can see that columns release_date have type as object, so I will change it in type date.

In [12]:
```
# change string to date format
df.release_date = pd.to_datetime(df['release_date'])

# see the update info
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10703 entries, 0 to 10865
Data columns (total 17 columns):
id                      10703 non-null int64
popularity              10703 non-null float64
budget                  5150 non-null float64
revenue                 4843 non-null float64
original_title          10703 non-null object
cast                    10703 non-null object
director                10703 non-null object
keywords                9293 non-null object
runtime                 10703 non-null float64
genres                  10703 non-null object
production_companies    9759 non-null object
release_date            10703 non-null datetime64[ns]
```

```
vote_count              10703 non-null int64
vote_average            10703 non-null float64
release_year            10703 non-null int64
budget_adj              5150 non-null float64
revenue_adj             4843 non-null float64
dtypes: datetime64[ns](1), float64(7), int64(3), object(6)
memory usage: 1.5+ MB
```

Clean Data Information

```
In [13]: # the last data info
         df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10703 entries, 0 to 10865
Data columns (total 17 columns):
id                      10703 non-null int64
popularity              10703 non-null float64
budget                  5150 non-null float64
revenue                 4843 non-null float64
original_title          10703 non-null object
cast                    10703 non-null object
director                10703 non-null object
keywords                9293 non-null object
runtime                 10703 non-null float64
genres                  10703 non-null object
production_companies    9759 non-null object
release_date            10703 non-null datetime64[ns]
vote_count              10703 non-null int64
vote_average            10703 non-null float64
release_year            10703 non-null int64
budget_adj              5150 non-null float64
revenue_adj             4843 non-null float64
dtypes: datetime64[ns](1), float64(7), int64(3), object(6)
memory usage: 1.5+ MB
```

```
In [14]: # the last statistica summary from data
         df.describe()
```

Out[14]:

	id	popularity	budget	revenue	runtime	\
count	10703.000000	10703.000000	5.150000e+03	4.843000e+03	10703.000000	
mean	64904.988321	0.653818	3.084401e+07	8.933981e+07	102.736896	
std	91161.996308	1.005687	3.893782e+07	1.621546e+08	30.079331	
min	5.000000	0.000188	1.000000e+00	2.000000e+00	3.000000	
25%	10538.500000	0.211533	6.000000e+06	7.779664e+06	90.000000	
50%	20235.000000	0.388036	1.750000e+07	3.191160e+07	99.000000	
75%	73637.000000	0.722438	4.000000e+07	1.000000e+08	112.000000	

```
max    417859.000000     32.985763  4.250000e+08  2.781506e+09    900.000000

         vote_count  vote_average  release_year   budget_adj   revenue_adj
count  10703.000000  10703.000000  10703.000000  5.150000e+03  4.843000e+03
mean     220.333178      5.966112   2001.235355  3.701495e+07  1.152341e+08
std      579.481969      0.930155     12.825920  4.198674e+07  1.989424e+08
min       10.000000      1.500000   1960.000000  9.210911e-01  2.370705e+00
25%       17.000000      5.400000   1995.000000  8.210996e+06  1.048057e+07
50%       39.000000      6.000000   2006.000000  2.294283e+07  4.402879e+07
75%      149.000000      6.600000   2011.000000  5.024535e+07  1.317599e+08
max     9767.000000      9.200000   2015.000000  4.250000e+08  2.827124e+09
```

Exploratory Data Analysis

in this section I will answer the question was declare in introduction

I define some function that can help to answer the question
list function name: Function as_currency Function get_movie_info Function get_hi_low

In [15]:
```
# change numeric format to dollar format
def as_currency(amount):
    if amount >= 0:
        return '${:,.2f}'.format(amount) # return positive data in dollar version with
    else:
        return '-${:,.2f}'.format(-amount) # return negative data in dollar version wit
```

In [16]:
```
# get movie information
def get_movie_info(index, title):
    info = pd.DataFrame(df.loc[index]) # set some Data frame by index

    currency_col = ['budget','revenue','profit','budget_adj','revenue_adj','profit_adj'
    # for each currency in list above,  do the following procedure
    for idx in currency_col:
        info.loc[idx] = as_currency(info.loc[idx].item()) # change value number into cu

    info.columns = [title] # change column name so it the reader will be clear what tha
    return info
```

In [17]:
```
# get hihgest and lowes data information
def get_hi_low(column):
    # highest
    # get the index value of the highest number
    highest_idx = df[column].idxmax(skipna=True)
    ## get data from index before
    title = "Highest " + column
    highest_data = get_movie_info(highest_idx,title)

    # lowest
    ## get the index value of the highest number
```

```
        lowest_idx = df[column].idxmin(skipna=True)
        ## get data from index before
        title = "Lowest " + column
        lowest_data = get_movie_info(lowest_idx,title)

        #concatenating two dataframes
        hi_low_data = pd.concat([highest_data, lowest_data], axis = 1)

        return hi_low_data
```

General Question
1. Which movie earns the highest and lowest profit?

In [18]:
```
# add column profit in data
df['profit'] = df['revenue']-df['budget']
df['profit_adj'] = df['revenue_adj']-df['budget_adj']

# previewing the changes in the dataset
df.head(3)
```

Out[18]:
```
       id  popularity       budget       revenue      original_title  \
0  135397   32.985763  150000000.0  1.513529e+09      Jurassic World
1   76341   28.419936  150000000.0  3.784364e+08  Mad Max: Fury Road
2  262500   13.112507  110000000.0  2.952382e+08           Insurgent

                                                cast           director  \
0  Chris Pratt|Bryce Dallas Howard|Irrfan Khan|Vi...    Colin Trevorrow
1  Tom Hardy|Charlize Theron|Hugh Keays-Byrne|Nic...      George Miller
2  Shailene Woodley|Theo James|Kate Winslet|Ansel...  Robert Schwentke

                                            keywords  runtime  \
0  monster|dna|tyrannosaurus rex|velociraptor|island    124.0
1   future|chase|post-apocalyptic|dystopia|australia    120.0
2  based on novel|revolution|dystopia|sequel|dyst...    119.0

                                      genres  \
0  Action|Adventure|Science Fiction|Thriller
1  Action|Adventure|Science Fiction|Thriller
2         Adventure|Science Fiction|Thriller

                                production_companies release_date  vote_count  \
0  Universal Studios|Amblin Entertainment|Legenda...   2015-06-09        5562
1  Village Roadshow Pictures|Kennedy Miller Produ...   2015-05-13        6185
2  Summit Entertainment|Mandeville Films|Red Wago...   2015-03-18        2480

   vote_average  release_year   budget_adj   revenue_adj        profit  \
0           6.5          2015 1.379999e+08  1.392446e+09  1.363529e+09
1           7.1          2015 1.379999e+08  3.481613e+08  2.284364e+08
```

```
2           6.3         2015  1.012000e+08  2.716190e+08  1.852382e+08

     profit_adj
0  1.254446e+09
1  2.101614e+08
2  1.704191e+08
```

In [19]:
```
# check the update data, kolom profit and profit_adj will have na value because some re
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 10703 entries, 0 to 10865
Data columns (total 19 columns):
id                      10703 non-null int64
popularity              10703 non-null float64
budget                  5150 non-null float64
revenue                 4843 non-null float64
original_title          10703 non-null object
cast                    10703 non-null object
director                10703 non-null object
keywords                9293 non-null object
runtime                 10703 non-null float64
genres                  10703 non-null object
production_companies    9759 non-null object
release_date            10703 non-null datetime64[ns]
vote_count              10703 non-null int64
vote_average            10703 non-null float64
release_year            10703 non-null int64
budget_adj              5150 non-null float64
revenue_adj             4843 non-null float64
profit                  3849 non-null float64
profit_adj              3849 non-null float64
dtypes: datetime64[ns](1), float64(9), int64(3), object(6)
memory usage: 1.6+ MB
```

list used function: Function get_hi_low

In [20]:
```
# Find highest and lowest profit
get_hi_low('profit')
```

Out[20]:
```
                                                       Highest profit  \
id                                                              19995
popularity                                                    9.43277
budget                                                $237,000,000.00
revenue                                             $2,781,505,847.00
original_title                                                 Avatar
cast                    Sam Worthington|Zoe Saldana|Sigourney Weaver|S...
director                                                James Cameron
```

```
keywords              culture clash|future|space war|space colony|so...
runtime                                                            162
genres                        Action|Adventure|Fantasy|Science Fiction
production_companies  Ingenious Film Partners|Twentieth Century Fox ...
release_date                                       2009-12-10 00:00:00
vote_count                                                        8458
vote_average                                                       7.1
release_year                                                      2009
budget_adj                                             $240,886,902.89
revenue_adj                                          $2,827,123,750.41
profit                                               $2,544,505,847.00
profit_adj                                           $2,586,236,847.52

                                                         Lowest profit
id                                                               46528
popularity                                                     0.25054
budget                                                 $425,000,000.00
revenue                                                 $11,087,569.00
original_title                                         The Warrior's Way
cast                  Kate Bosworth|Jang Dong-gun|Geoffrey Rush|Dann...
director                                                   Sngmoo Lee
keywords              assassin|small town|revenge|deception|super speed
runtime                                                            100
genres                         Adventure|Fantasy|Action|Western|Thriller
production_companies                          Boram Entertainment Inc.
release_date                                       2010-12-02 00:00:00
vote_count                                                          74
vote_average                                                       6.4
release_year                                                      2010
budget_adj                                             $425,000,000.00
revenue_adj                                             $11,087,569.00
profit                                                -$413,912,431.00
profit_adj                                            -$413,912,431.00
```

from information above we know that the highest profit movie is Avatar(2009) which is \$2,544,505,847 and the lowest profit is The Warrior's Way(2010) which is -\$413,912,431.00, so that movie was loss money. from the information movie we know that columns ending with "_adj" show the budget and revenue of the associated movie in terms of 2010 dollars, accounting for inflation over time. So I also want to check the highest and lowest profit movie if we accounting for inflation over time

list used function: Function get_hi_low

In [21]:
```python
# Find highest and lowest profit_adj
get_hi_low('profit_adj')
```

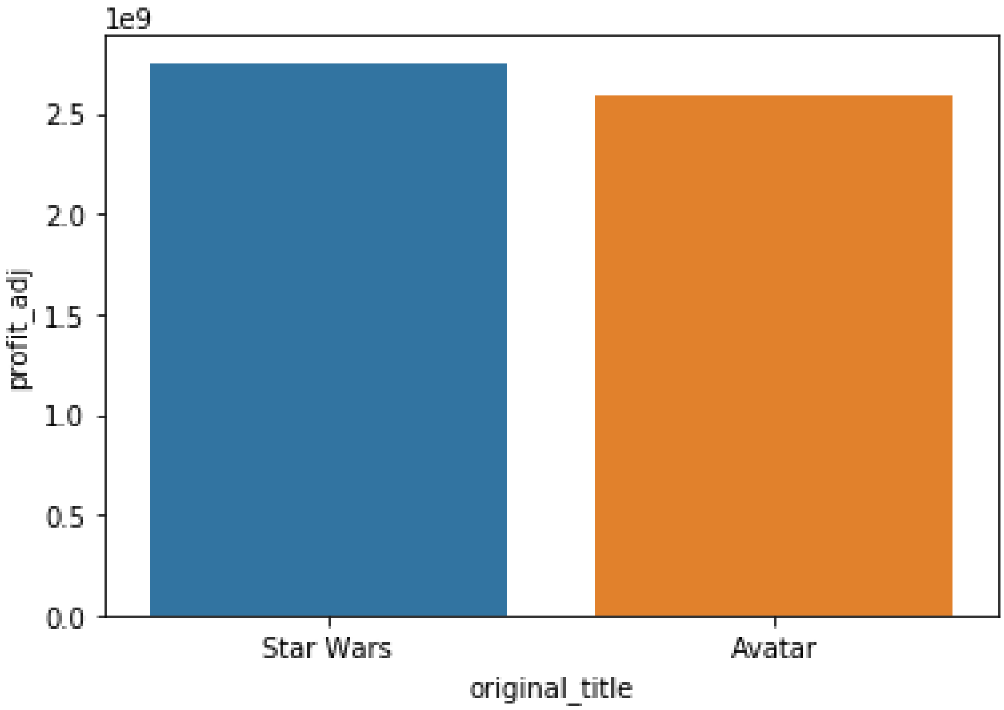
Out[21]:
```
                                                    Highest profit_adj  \
id                                                                   11
popularity                                                      12.0379
```

```
budget                                                   $11,000,000.00
revenue                                                 $775,398,007.00
original_title                                                Star Wars
cast                  Mark Hamill|Harrison Ford|Carrie Fisher|Peter ...
director                                                   George Lucas
keywords                     android|galaxy|hermit|death star|lightsaber
runtime                                                             121
genres                                 Adventure|Action|Science Fiction
production_companies   Lucasfilm|Twentieth Century Fox Film Corporation
release_date                                        1977-03-20 00:00:00
vote_count                                                         4428
vote_average                                                        7.9
release_year                                                       1977
budget_adj                                               $39,575,591.36
revenue_adj                                           $2,789,712,242.28
profit                                                  $764,398,007.00
profit_adj                                            $2,750,136,650.92

                                                     Lowest profit_adj
id                                                                46528
popularity                                                      0.25054
budget                                                  $425,000,000.00
revenue                                                  $11,087,569.00
original_title                                        The Warrior's Way
cast                  Kate Bosworth|Jang Dong-gun|Geoffrey Rush|Dann...
director                                                     Sngmoo Lee
keywords              assassin|small town|revenge|deception|super speed
runtime                                                             100
genres                         Adventure|Fantasy|Action|Western|Thriller
production_companies                           Boram Entertainment Inc.
release_date                                        2010-12-02 00:00:00
vote_count                                                           74
vote_average                                                        6.4
release_year                                                       2010
budget_adj                                              $425,000,000.00
revenue_adj                                              $11,087,569.00
profit                                                 -$413,912,431.00
profit_adj                                             -$413,912,431.00
```

from information above we know that the highest profit movie from all movie in our data is Star Wars(1977) which is \$2,750,136,650.92 that is the highest profit if we accounting for inflation over time. The profit from Start Wars is \$163,899,803.4 more than Avatar. The lowest movie profit is still The Warrior's Way(2010) which is -\$413,912,431.00

In [22]:
```
# visualize different profit_adj from Star Wars and Avatar
highest_profit = df[['original_title','profit_adj']].query("original_title in ['Star Wa
sns.barplot(x="original_title", y="profit_adj", data=highest_profit)
```

Out[22]: <matplotlib.axes._subplots.AxesSubplot at 0x7fdd5cc88588>



Answer Question General 1

- The highest profit movie is Avatar(2009), but if we check the inflation over time so the highest profit move is Star Wars(1977)
- The lowest profit movie is The Warrior's Way(2010)

Go To List Question
2. Which movie have the highest and lowest revenue?
list used function: Function get_hi_low

In [23]:
```
# Highest and lowest revenue
get_hi_low('revenue')
```

Out[23]:
```
                                                      Highest revenue  \
id                                                              19995
popularity                                                    9.43277
budget                                                $237,000,000.00
revenue                                             $2,781,505,847.00
original_title                                                 Avatar
cast                      Sam Worthington|Zoe Saldana|Sigourney Weaver|S...
director                                                James Cameron
keywords                  culture clash|future|space war|space colony|so...
runtime                                                           162
```

```
genres                                     Action|Adventure|Fantasy|Science Fiction
production_companies  Ingenious Film Partners|Twentieth Century Fox ...
release_date                                                2009-12-10 00:00:00
vote_count                                                                 8458
vote_average                                                                7.1
release_year                                                               2009
budget_adj                                                      $240,886,902.89
revenue_adj                                                   $2,827,123,750.41
profit                                                        $2,544,505,847.00
profit_adj                                                    $2,586,236,847.52

                                                                 Lowest revenue
id                                                                        13537
popularity                                                             0.462609
budget                                                            $6,000,000.00
revenue                                                                   $2.00
original_title                                                   Shattered Glass
cast                  Hayden Christensen|Peter Sarsgaard|ChloÃ« Sevi...
director                                                              Billy Ray
keywords                                                                    NaN
runtime                                                                      94
genres                                                            Drama|History
production_companies  Lions Gate Films|Cruise/Wagner Productions|Bau...
release_date                                                2003-11-14 00:00:00
vote_count                                                                   46
vote_average                                                                6.4
release_year                                                               2003
budget_adj                                                        $7,112,115.87
revenue_adj                                                               $2.37
profit                                                           -$5,999,998.00
profit_adj                                                       -$7,112,113.50
```

from information above we know that the highest revenue movie is Avatar(2009) which is \$2,781,505,847. The lowest movie revenue is Shattered Glass(2003) which is \$2

list used function: Function get_hi_low

In [24]:
```python
# Highest and lowest revenue_adj
get_hi_low('revenue_adj')
```

Out[24]:
```
                                                           Highest revenue_adj  \
id                                                                        19995
popularity                                                              9.43277
budget                                                          $237,000,000.00
revenue                                                       $2,781,505,847.00
original_title                                                           Avatar
cast                  Sam Worthington|Zoe Saldana|Sigourney Weaver|S...
director                                                          James Cameron
```

```
keywords              culture clash|future|space war|space colony|so...
runtime                                                             162
genres                          Action|Adventure|Fantasy|Science Fiction
production_companies  Ingenious Film Partners|Twentieth Century Fox ...
release_date                                        2009-12-10 00:00:00
vote_count                                                         8458
vote_average                                                        7.1
release_year                                                       2009
budget_adj                                              $240,886,902.89
revenue_adj                                           $2,827,123,750.41
profit                                                $2,544,505,847.00
profit_adj                                            $2,586,236,847.52

                                                    Lowest revenue_adj
id                                                                13537
popularity                                                     0.462609
budget                                                    $6,000,000.00
revenue                                                           $2.00
original_title                                           Shattered Glass
cast                  Hayden Christensen|Peter Sarsgaard|ChloÃ« Sevi...
director                                                      Billy Ray
keywords                                                            NaN
runtime                                                              94
genres                                                    Drama|History
production_companies  Lions Gate Films|Cruise/Wagner Productions|Bau...
release_date                                        2003-11-14 00:00:00
vote_count                                                           46
vote_average                                                        6.4
release_year                                                       2003
budget_adj                                                $7,112,115.87
revenue_adj                                                       $2.37
profit                                                   -$5,999,998.00
profit_adj                                               -$7,112,113.50
```

from information above we know that the highest revenue movie from all movie in our data is Avatar(2009) which is \$2,781,505,847 The lowest movie revenue is still Shattered Glass(2003) which is -\$2.37

Answer Question General 2

- The highest revenue movie is Avatar(2009)
- The lowest revenue movie is Shattered Glass(2003)

From answer question 1 we found that profit Avatar is lower than Star Wars so we can make conclusion that budget Star Wars is lower than Avatar because revenue Avatar is biger than Star Wars

Go To List Question

3. Which movie have the highest and lowest budget?

list used function: Function get_hi_low

```
In [25]: # Highest and lowest budget
         get_hi_low('budget')
```

```
Out[25]:                                                   Highest budget  \
         id                                                          46528
         popularity                                                0.25054
         budget                                            $425,000,000.00
         revenue                                            $11,087,569.00
         original_title                                    The Warrior's Way
         cast                  Kate Bosworth|Jang Dong-gun|Geoffrey Rush|Dann...
         director                                               Sngmoo Lee
         keywords              assassin|small town|revenge|deception|super speed
         runtime                                                       100
         genres                        Adventure|Fantasy|Action|Western|Thriller
         production_companies                       Boram Entertainment Inc.
         release_date                                  2010-12-02 00:00:00
         vote_count                                                     74
         vote_average                                                  6.4
         release_year                                                 2010
         budget_adj                                        $425,000,000.00
         revenue_adj                                        $11,087,569.00
         profit                                           -$413,912,431.00
         profit_adj                                       -$413,912,431.00

                                                            Lowest budget
         id                                                         287524
         popularity                                               0.177102
         budget                                                      $1.00
         revenue                                                     -$nan
         original_title                                        Fear Clinic
         cast                  Thomas Dekker|Robert Englund|Cleopatra Coleman...
         director                                              Robert Hall
         keywords                                        phobia|doctor|fear
         runtime                                                        95
         genres                                                     Horror
         production_companies  Dry County Films|Anchor Bay Entertainment|Movi...
         release_date                                  2014-10-31 00:00:00
         vote_count                                                     15
         vote_average                                                  4.1
         release_year                                                 2014
         budget_adj                                                  $0.92
         revenue_adj                                                 -$nan
         profit                                                      -$nan
         profit_adj                                                  -$nan
```

from information above we know that the highest budget movie is The Warrior's Way(2010) which is \$425,000,000 The lowest movie budget is Fear Clinic(2014) which is \$1.

list used function: Function get_hi_low

```
In [26]: # Highest and lowest budget_adj
         get_hi_low('budget_adj')
```

```
Out[26]:                                             Highest budget_adj  \
         id                                                       46528
         popularity                                             0.25054
         budget                                         $425,000,000.00
         revenue                                         $11,087,569.00
         original_title                                 The Warrior's Way
         cast                  Kate Bosworth|Jang Dong-gun|Geoffrey Rush|Dann...
         director                                           Sngmoo Lee
         keywords              assassin|small town|revenge|deception|super speed
         runtime                                                    100
         genres                        Adventure|Fantasy|Action|Western|Thriller
         production_companies                     Boram Entertainment Inc.
         release_date                                  2010-12-02 00:00:00
         vote_count                                                  74
         vote_average                                               6.4
         release_year                                              2010
         budget_adj                                     $425,000,000.00
         revenue_adj                                     $11,087,569.00
         profit                                         -$413,912,431.00
         profit_adj                                     -$413,912,431.00

                                                          Lowest budget_adj
         id                                                          287524
         popularity                                                0.177102
         budget                                                       $1.00
         revenue                                                      -$nan
         original_title                                         Fear Clinic
         cast                  Thomas Dekker|Robert Englund|Cleopatra Coleman...
         director                                               Robert Hall
         keywords                                        phobia|doctor|fear
         runtime                                                         95
         genres                                                      Horror
         production_companies  Dry County Films|Anchor Bay Entertainment|Movi...
         release_date                                   2014-10-31 00:00:00
         vote_count                                                      15
         vote_average                                                   4.1
         release_year                                                  2014
         budget_adj                                                   $0.92
         revenue_adj                                                  -$nan
         profit                                                       -$nan
         profit_adj                                                   -$nan
```

from information above we know that the highest budget_adj movie is The Warrior's

Way(2010) which is \$425,000,000 The lowest movie budget_adj is Fear Clinic(2014) which is \$1.

Answer Question General 3

- The highest budget movie is The Warrior's Way(2010)
- The lowest budget movie is Fear Clinic(2014) From answer question 1 we found that profit The Warrior's Way(2010) is lowest maybe because it is have a highest budget

Go To List Question
4. Which movie have the longest and shortest runtime?
list used function: Function get_hi_low

In [27]:
```
# Highest and lowest runtime
get_hi_low('runtime')
```

Out[27]:
```
                                                      Highest runtime  \
id                                                             125336
popularity                                                   0.006925
budget                                                          -$nan
revenue                                                         -$nan
original_title                         The Story of Film: An Odyssey
cast                      Mark Cousins|Jean-Michel Frodon|Cari Beauchamp...
director                                                 Mark Cousins
keywords                  cinema|nouvelle vague|hindi cinema|cinema novo...
runtime                                                           900
genres                                                    Documentary
production_companies                                              NaN
release_date                                      2011-09-03 00:00:00
vote_count                                                         14
vote_average                                                      9.2
release_year                                                     2011
budget_adj                                                      -$nan
revenue_adj                                                     -$nan
profit                                                          -$nan
profit_adj                                                      -$nan

                                                       Lowest runtime
id                                                             264170
popularity                                                   0.202776
budget                                                          -$nan
revenue                                                         -$nan
original_title                                   Batman: Strange Days
cast                                 Kevin Conroy|Brian George|Tara Strong
director                                                   Bruce Timm
keywords                  dc comics|superhero|based on comic book|noir|p...
runtime                                                             3
genres                                                   Action|Animation
production_companies                                        DC Comics
```

```
release_date                2014-04-09 00:00:00
vote_count                                   20
vote_average                                7.6
release_year                               2014
budget_adj                                -$nan
revenue_adj                               -$nan
profit                                    -$nan
profit_adj                                -$nan
```

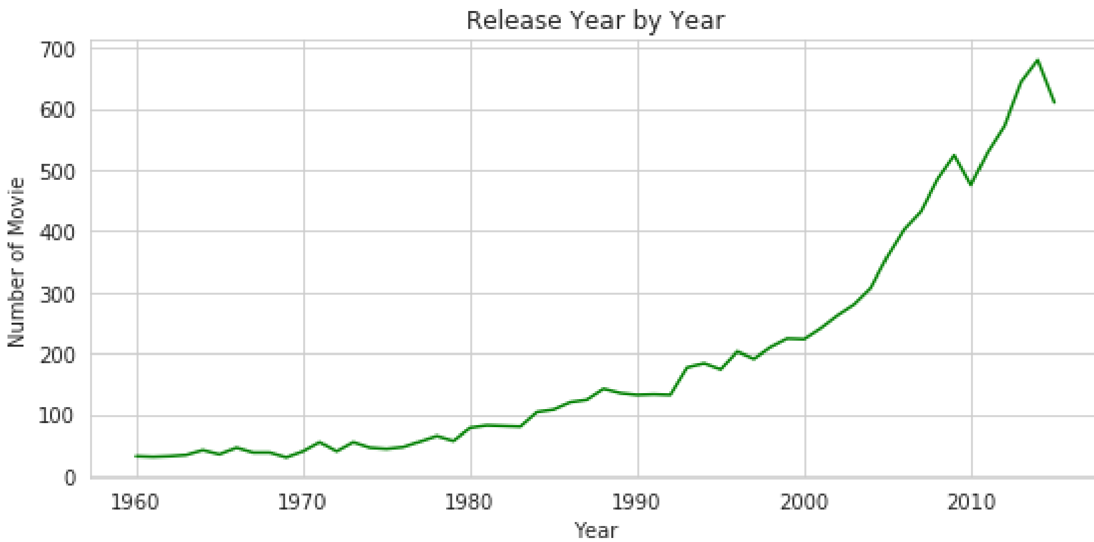
Answer Question General 4

- The longest runtime movie is The Story of Film: An Odyssey(2011) that is 900 minutes
- The shortest runtime movie is Batman: Strange Days(2014) that is 3 minutes

Go To List Question
5. How much movie released year by year?

```
In [28]: # get number of movie group by release year
         release = df.groupby('release_year').size()
         # get index number of movie group by release year
         release_idx = release.index
```

```
In [29]: # visualisation
         # set style
         sns.set_style('whitegrid')
         # set x, y axis data
         x, y = release_idx, release
         # set size
         plt.figure(figsize=(9, 4))
         # plot line chart for number of release
         plt.plot(x, y, color = 'g', label = 'mean')
         # set title and labels
         plt.title('Release Year by Year')
         plt.xlabel('Year')
         plt.ylabel('Number of Movie');
```



Answer Question General 5

from figure above we know that movie number always increassing every year, but in 2010 we have some slope.

Go To List Question

6. How distribution of profit in different popularity levels in recent ten years?

To help the next process, I will make some function

list function name: Function get_class Function plot_by_year

In [30]:
```
# make level from quantile to help categories column
def get_class(df, column):
    # find quantile to decide that class
    min_value = df[column].min()
    quantile_1 = df[column].describe()[4]
    quantile_2 = df[column].describe()[5]
    quantile_3 = df[column].describe()[6]
    max_value = df[column].max()

    # bin edges that will be used to "cut" the data into groups
    bin_level = [ min_value, quantile_1, quantile_2, quantile_3, max_value]
    # labels for the four budget level groups
    bin_name = [ 'Low', 'Medium', 'High', 'Very High']
    # creates budget_levels column
    name = '{}_levels'.format(column)
    df[name] = pd.cut(df[column], bin_level, labels=bin_name, include_lowest = True)
    return df
```

In [31]:
```
# plot data group by year of year
def plot_by_year(df,column,object_column,dfyear):
```

```
    # set the positions and width for the bars
    position = list(range(len(df.query('%s =="Low"' % column))))
    width = 0.2

    # plot the bars
    fig, ax = plt.subplots(figsize=(20,5))

    # create the bar with Low data, in position
    plt.bar(position,df.query('%s =="Low"' % column)[object_column], width, alpha=0.8,

    # create the bar with Medium data, in position pos + some width buffer
    plt.bar([p + width for p in position], df.query('%s =="Medium"'% column)[object_col
            color='#FFD700', label='Medium')

    # create the bar with High data, in position + some width buffer so they not inters
    plt.bar([p + width*2 for p in position], df.query('%s =="High"' % column)[object_co
            color='#ADFF2F', label='High')

    # create the bar with Very High data,
    # in position + some width buffer so they not intersect each other
    plt.bar([p + width*3 for p in position], df.query('%s =="Very High"'% column)[objec
            color='#008000', label='Very High')

    # set x axis and y axis
    ax.set_ylabel(object_column) # set the y-axis label
    ax.set_title('%s in different %s levels in recent 10 years' % (object_column,column
    ax.set_xticks([p + 1.5 * width for p in position]) # set the position of the x-tick
    ax.set_xticklabels(dfyear) # set the labels for the x ticks
    ax.set_ylim([min(df[object_column]),max(df[object_column])]) # set the y ticks

    # add the legend and showing the plot
    plt.legend( loc='upper left')
    plt.grid()
    plt.show()
```

Lets find distribution of profit in different popularity levels in recent ten years

list used function: Function get_class

```
In [32]: # get popularity level
         # choose the recent 10 years
         dfyear = np.sort(df.release_year.unique())[-10:]
         # creat a empty df to assign df with popularity levels
         df_popularity = pd.DataFrame()

         #for each year, do the following procedure
         for year in dfyear:
             df_temp = df.query('release_year == "%s"' % year).copy() # filter data with the sel
```

```
    df_temp = get_class(df_temp,'popularity') # get popularity level
    df_popularity = df_popularity.append(df_temp) # append to df_popularity
df_popularity.head(3)
```

Out[32]:

```
       id  popularity       budget       revenue  \
6554  834    5.838503   50000000.0  1.113408e+08
6555   58    4.205992  200000000.0  1.065660e+09
6556  920    3.941265  120000000.0  4.619831e+08

                                   original_title  \
6554                          Underworld: Evolution
6555  Pirates of the Caribbean: Dead Man's Chest
6556                                           Cars

                                                 cast  \
6554  Kate Beckinsale|Scott Speedman|Tony Curran|Sha...
6555  Johnny Depp|Orlando Bloom|Keira Knightley|Bill...
6556  Owen Wilson|Paul Newman|Bonnie Hunt|Larry the ...

                      director  \
6554               Len Wiseman
6555            Gore Verbinski
6556  John Lasseter|Joe Ranft

                                             keywords  runtime  \
6554               budapest|key|light|werewolf|evolution    106.0
6555  witch|fortune teller|bondage|exotic island|mon...    151.0
6556    car race|car journey|village and town|road|auto    117.0

                                         genres  \
6554  Fantasy|Action|Science Fiction|Thriller
6555                 Adventure|Fantasy|Action
6556         Animation|Adventure|Comedy|Family

                                  production_companies release_date  \
6554                Lakeshore Entertainment|Screen Gems   2006-01-12
6555  Walt Disney Pictures|Jerry Bruckheimer Films|S...   2006-06-20
6556       Walt Disney Pictures|Pixar Animation Studios   2006-06-08

      vote_count  vote_average  release_year    budget_adj   revenue_adj  \
6554        1015           6.3          2006  5.408346e+07  1.204339e+08
6555        3181           6.8          2006  2.163338e+08  1.152691e+09
6556        2336           6.4          2006  1.298003e+08  4.997129e+08

          profit    profit_adj popularity_levels
6554   61340801.0  6.635045e+07         Very High
6555  865659812.0  9.363575e+08         Very High
6556  341983149.0  3.699126e+08         Very High
```

```
In [33]: # lets find statistic summary from popularity to know how the quantile cut the levels,
         df_popularity.describe()
```

```
Out[33]:                  id   popularity        budget       revenue      runtime  \
         count    5354.000000  5354.000000  2.379000e+03  2.165000e+03  5354.000000
         mean   113436.690325     0.713095  3.475924e+07  1.031749e+08    99.186029
         std    107304.567574     1.219587  4.695805e+07  1.968944e+08    30.217657
         min        17.000000     0.000620  1.000000e+00  3.000000e+00     3.000000
         25%     19715.750000     0.206118  6.000000e+06  3.338228e+06    89.000000
         50%     71861.500000     0.390016  1.700000e+07  3.155486e+07    96.000000
         75%    201730.750000     0.771668  4.000000e+07  1.075972e+08   108.000000
         max    417859.000000    32.985763  4.250000e+08  2.781506e+09   900.000000

                 vote_count  vote_average  release_year    budget_adj   revenue_adj  \
         count  5354.000000   5354.000000   5354.000000  2.379000e+03  2.165000e+03
         mean    268.594135      5.891688   2010.928838  3.421956e+07  1.010225e+08
         std     681.641308      0.992524      2.831724  4.604534e+07  1.914991e+08
         min      10.000000      1.500000   2006.000000  9.210911e-01  3.038360e+00
         25%      18.000000      5.300000   2009.000000  5.816388e+06  3.171821e+06
         50%      42.000000      5.900000   2011.000000  1.682670e+07  3.038360e+07
         75%     182.000000      6.600000   2013.000000  4.065602e+07  1.055790e+08
         max    9767.000000      9.200000   2015.000000  4.250000e+08  2.827124e+09

                     profit    profit_adj
         count  1.712000e+03  1.712000e+03
         mean   8.334300e+07  8.133364e+07
         std    1.793768e+08  1.743710e+08
         min   -4.139124e+08 -4.139124e+08
         25%   -2.220506e+06 -2.191682e+06
         50%    2.152898e+07  2.145700e+07
         75%    9.400637e+07  9.068777e+07
         max    2.544506e+09  2.586237e+09
```

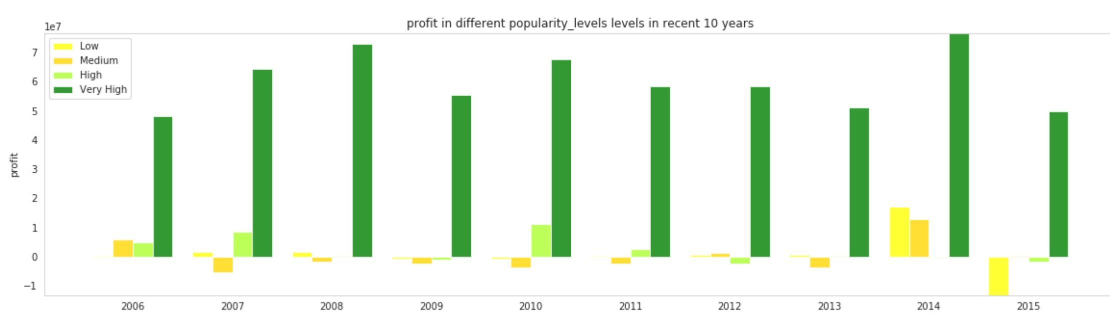
```
In [34]: # group the dataframe we created above with each popularity_levels in each year, find t
         # I choose median because it not have effect from outlier data
         df_popularity_by_year = df_popularity.groupby(['release_year','popularity_levels']).med
         df_popularity_by_year.head(8)
```

```
Out[34]:                                       id  popularity      budget     revenue  \
         release_year popularity_levels
         2006         Low                 14872.0    0.113193   7000000.0   4687766.0
                      Medium              12225.0    0.297434   8500000.0  11290263.5
                      High                 9806.5    0.546223  20000000.0  23629912.0
                      Very High            7551.0    1.182280  40000000.0  93161322.5
         2007         Low                 15117.5    0.139703  10000000.0  10337477.0
                      Medium              13517.5    0.298249  10000000.0   3478080.0
                      High                10966.0    0.519439  19000000.0  22179430.0
                      Very High            4748.0    1.188489  47500000.0  95652995.5
```

```
                                  runtime  vote_count  vote_average  \
release_year popularity_levels
2006         Low                     95.0        17.0          5.90
             Medium                  95.0        27.0          5.80
             High                   100.0        72.5          5.90
             Very High              106.0       306.0          6.30
2007         Low                     93.5        16.0          5.85
             Medium                  96.0        26.0          5.80
             High                    97.5        64.0          6.00
             Very High              105.0       453.0          6.20

                                    budget_adj   revenue_adj       profit  \
release_year popularity_levels
2006         Low                  7.571684e+06  5.070612e+06     166000.0
             Medium               9.194188e+06  1.221233e+07    5796544.0
             High                 2.163338e+07  2.555975e+07    4910153.5
             Very High            4.326677e+07  1.007697e+08   48197993.0
2007         Low                  1.051669e+07  1.087160e+07    1392364.0
             Medium               1.051669e+07  3.657787e+06   -5349184.5
             High                 1.998170e+07  2.332541e+07    8511656.5
             Very High            4.995426e+07  1.005953e+08   64373941.0

                                    profit_adj
release_year popularity_levels
2006         Low                  1.795571e+05
             Medium               6.269943e+06
             High                 5.311162e+06
             Very High            5.213428e+07
2007         Low                  1.464305e+06
             Medium              -5.625569e+06
             High                 8.951442e+06
             Very High            6.770005e+07
```

list used function: Function plot_by_year

In [35]:
```
# visualization
plot_by_year(df_popularity_by_year, 'popularity_levels', 'profit',dfyear)
```



Answer Question General 6

from figure above we found that the highest popularity didn't mean the highest profit, but for very high of popularity have highest profit. So keep the movie get very high popularity levels, with minimum popularity is 0.710151 to get high profit.

Go To List Question

7. How distribution of profit in different vote average levels in recent ten years?

list used function: Function get_class

```
In [36]: # lets make rating level just like question before
         # choose the recent 10 years
         dfyear = np.sort(df.release_year.unique())[-10:]
         # creat a empty df to assign df with vote average levels
         df_vote_average = pd.DataFrame()

         #for each year, do the following procedure
         for year in dfyear:
             df_temp = df.query('release_year == "%s"' % year).copy() # filter data with the sel
             df_temp = get_class(df_temp,'vote_average') # get vote average level
             df_vote_average = df_vote_average.append(df_temp) # append to df_popularity
         df_vote_average.head(3)
```

```
Out[36]:        id  popularity       budget       revenue  \
         6554  834    5.838503   50000000.0  1.113408e+08
         6555   58    4.205992  200000000.0  1.065660e+09
         6556  920    3.941265  120000000.0  4.619831e+08

                                            original_title  \
         6554                         Underworld: Evolution
         6555  Pirates of the Caribbean: Dead Man's Chest
         6556                                          Cars

                                                        cast  \
         6554  Kate Beckinsale|Scott Speedman|Tony Curran|Sha...
         6555  Johnny Depp|Orlando Bloom|Keira Knightley|Bill...
         6556  Owen Wilson|Paul Newman|Bonnie Hunt|Larry the ...

                             director  \
         6554             Len Wiseman
         6555          Gore Verbinski
         6556  John Lasseter|Joe Ranft

                                                    keywords  runtime  \
         6554              budapest|key|light|werewolf|evolution    106.0
         6555  witch|fortune teller|bondage|exotic island|mon...    151.0
         6556    car race|car journey|village and town|road|auto    117.0
```

```
                                         genres  \
6554  Fantasy|Action|Science Fiction|Thriller
6555                 Adventure|Fantasy|Action
6556        Animation|Adventure|Comedy|Family

                                   production_companies release_date  \
6554                Lakeshore Entertainment|Screen Gems   2006-01-12
6555  Walt Disney Pictures|Jerry Bruckheimer Films|S...   2006-06-20
6556       Walt Disney Pictures|Pixar Animation Studios   2006-06-08

      vote_count  vote_average  release_year    budget_adj   revenue_adj  \
6554        1015           6.3          2006  5.408346e+07  1.204339e+08
6555        3181           6.8          2006  2.163338e+08  1.152691e+09
6556        2336           6.4          2006  1.298003e+08  4.997129e+08

           profit    profit_adj vote_average_levels
6554   61340801.0  6.635045e+07                High
6555  865659812.0  9.363575e+08           Very High
6556  341983149.0  3.699126e+08                High
```

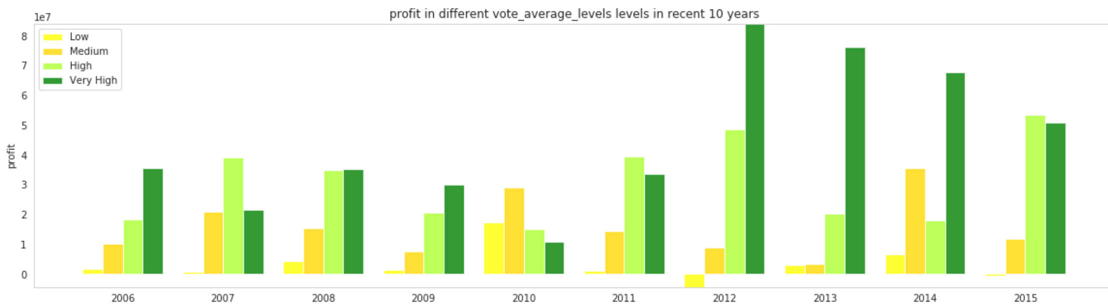
In [37]:
```
# lets find statistic summary from vote avg to know how the quantile cut the levels, an
df_vote_average.describe()
```

Out[37]:
```
                  id   popularity        budget       revenue      runtime  \
count    5354.000000  5354.000000  2.379000e+03  2.165000e+03  5354.000000
mean   113436.690325     0.713095  3.475924e+07  1.031749e+08    99.186029
std    107304.567574     1.219587  4.695805e+07  1.968944e+08    30.217657
min        17.000000     0.000620  1.000000e+00  3.000000e+00     3.000000
25%     19715.750000     0.206118  6.000000e+06  3.338228e+06    89.000000
50%     71861.500000     0.390016  1.700000e+07  3.155486e+07    96.000000
75%    201730.750000     0.771668  4.000000e+07  1.075972e+08   108.000000
max    417859.000000    32.985763  4.250000e+08  2.781506e+09   900.000000

        vote_count  vote_average  release_year    budget_adj   revenue_adj  \
count  5354.000000   5354.000000   5354.000000  2.379000e+03  2.165000e+03
mean    268.594135      5.891688   2010.928838  3.421956e+07  1.010225e+08
std     681.641308      0.992524      2.831724  4.604534e+07  1.914991e+08
min      10.000000      1.500000   2006.000000  9.210911e-01  3.038360e+00
25%      18.000000      5.300000   2009.000000  5.816388e+06  3.171821e+06
50%      42.000000      5.900000   2011.000000  1.682670e+07  3.038360e+07
75%     182.000000      6.600000   2013.000000  4.065602e+07  1.055790e+08
max    9767.000000      9.200000   2015.000000  4.250000e+08  2.827124e+09

             profit    profit_adj
count  1.712000e+03  1.712000e+03
mean   8.334300e+07  8.133364e+07
std    1.793768e+08  1.743710e+08
min   -4.139124e+08 -4.139124e+08
```

```
25%   -2.220506e+06 -2.191682e+06
50%    2.152898e+07  2.145700e+07
75%    9.400637e+07  9.068777e+07
max    2.544506e+09  2.586237e+09
```

list used function: Function plot_by_year

```
In [38]: # group the dataframe we created above with each popularity_levels in each year, find t
         # I choose median because it not have effect from outlier data
         df_vote_average_by_year = df_vote_average.groupby(['release_year','vote_average_levels'
         # visualization
         plot_by_year(df_vote_average_by_year, 'vote_average_levels', 'profit',dfyear)
```



Answer Question General 7

from figure above we found that the highest level of vote average not always mean the movie get the highest profit, especially to 2010 which medium vote have higher profit than high and very high vote average.

Go To List Question

Associated Question

1. What genre that associated with high popularity?

> Because I want to check the associate of genre and genre are multiple value with delimiter | , so I decide to devide the column into single value in each row so it will be a binner data with their popularity level to make the process easy, I make some function

list function name: Function remove_punctuation Function define_dict Function get_data_frame

```
In [39]: # remove punctuation, decimal, and space uneeded, so the string just containt character
         def remove_punctuation(sentence, chars: list = ['[^\w\s]', '_', '\d', '\n', '\r']):
             sentence = sentence.strip()
             sentence = re.sub("|".join(chars), " ", sentence)
             sentence = sentence.replace(' +', ' ')

             return sentence
```

```
In [40]: # define dictionary to help make dataframe with binner value,
         # the key of dictionary will save the column name, and the value will be number
         def define_dict(columns):
             d = {}
             for c in columns:
                 d[c] = 0
             return d

In [41]: # make binner dataframe with column we needed
         # metric are column level name
         def get_data_frame(old_df,column,metric):
             names = set() # make set to save what unique column we need it
             r = old_df.shape[0] # nrow of data
             c = old_df.shape[1] # nrow of columns

             # get set of data to generate column
             for i in range(0,r):
                 temp = str(old_df.iloc[i,old_df.columns.get_loc(column)]).split("|") # get arra
                 #for each word list temp, do the following procedure
                 for j in range(len(temp)):
                     last_name = remove_punctuation(temp[j].lower()).split(" ")[-1] # get last w
                     if last_name!='' and last_name!='nan': # if that is not missing value, save
                         names.add(last_name) # it will save all unic word split by | in column

             df_temp = pd.DataFrame(columns = names) # empty data frame to save the final data


             #for each data, do the following procedure
             for i in range(r):
                 dict_column = define_dict(names) # dictionary with name as key and zero as valu
                 dict_column[metric] = old_df.iloc[i,old_df.columns.get_loc(metric)] # add metri
                 temp = str(remove_punctuation(old_df.iloc[i,old_df.columns.get_loc(column)]).lo
                 #for each word in set names, do the following procedure
                 for name in names:
                     dict_column[name] += temp.count(name) # add number if the name exist in thi
                 df_add = pd.DataFrame(dict_column, index=[0]) # add the dict to dataframe so it
                 df_temp = df_temp.append(df_add,ignore_index=True,sort=False) # add to data fra

             return df_temp

In [42]: # make dataframe
         df_genres = get_data_frame(df_popularity,'genres','popularity_levels')

In [43]: # lets see the sample value of dataframe we make
         df_genres.head(3)

Out[43]:   mystery western romance action history thriller drama fantasy war fiction  \
         0       0       0       0      1       0        1     0       1   0       1
         1       0       0       0      1       0        0     0       1   0       0
```

```
2       0       0       0      0       0        0      0       0   0       0

   ... movie horror adventure documentary music crime animation family comedy  \
0  ...     0      0         0           0     0     0         0      0      0
1  ...     0      0         1           0     0     0         0      0      0
2  ...     0      0         1           0     0     0         1      1      1

  popularity_levels
0         Very High
1         Very High
2         Very High

[3 rows x 21 columns]
```

In [44]:
```
# lets find the sum in each column so we know what the genre most often used in all mov
df_genres.sum()
```

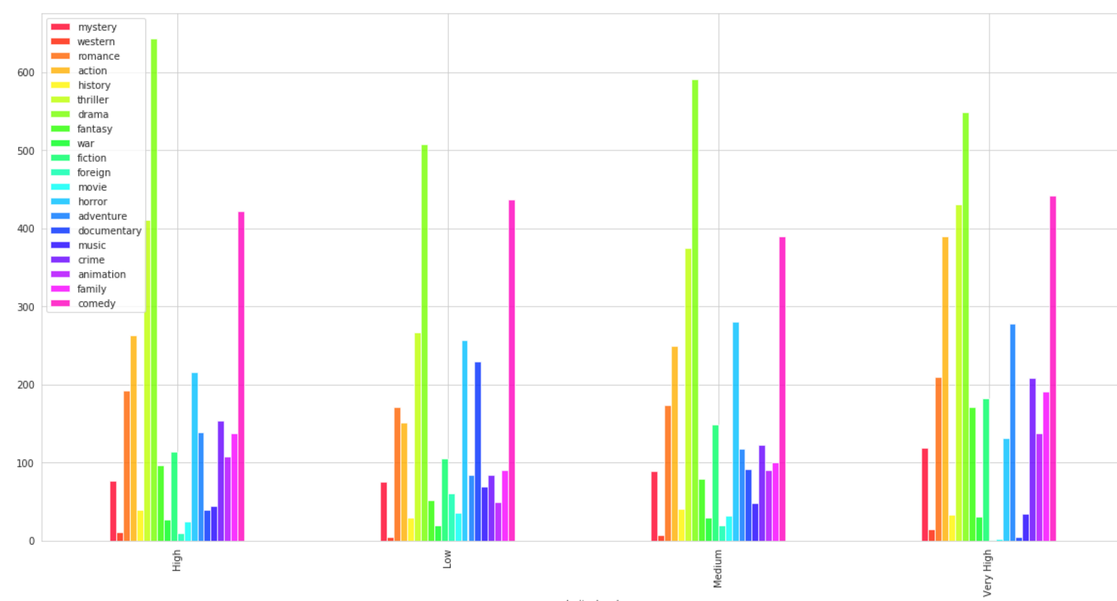
Out[44]:
```
mystery                                                                359
western                                                                 36
romance                                                                747
action                                                                1052
history                                                                141
thriller                                                              1482
drama                                                                 2291
fantasy                                                                398
war                                                                    106
fiction                                                                549
foreign                                                                 91
movie                                                                   95
horror                                                                 884
adventure                                                              618
documentary                                                            363
music                                                                  195
crime                                                                  569
animation                                                              383
family                                                                 519
comedy                                                                1690
popularity_levels    Very HighVery HighVery HighVery HighVery HighV...
dtype: object
```

In [45]:
```
# let's count the genre alomst use group by metric we decide before
df_genre_rank = df_genres.groupby(['popularity_levels']).sum()
df_genre_rank.head(8)
```

Out[45]:
```
                   mystery  western  romance  action  history  thriller  \
popularity_levels
High                    76       11      192     263       39       411
Low                     75        4      171     151       29       266
Medium                  89        7      174     249       40       375
```

```
Very High              119        14       210      389        33        430

                   drama  fantasy  war  fiction  foreign  movie  horror  \
popularity_levels
High                 643       96   27      114       10     25     216
Low                  508       52   19      105       60     36     257
Medium               591       79   29      148       20     32     280
Very High            549      171   31      182        1      2     131

                   adventure  documentary  music  crime  animation  family  \
popularity_levels
High                     139           39     44    154        107     138
Low                       84          229     69     84         49      90
Medium                   117           91     48    123         90     100
Very High                278            4     34    208        137     191

                   comedy
popularity_levels
High                  422
Low                   437
Medium                389
Very High             442
```

In [46]:
```
# plot sum of genre use in each level of popularity
df_genre_rank.plot(kind='bar',figsize=(20,10),colormap='gist_rainbow',alpha=0.8).legend
```



Answer Question Associate 1

from figure above we found that genre drama are high distributed in all popularity level movie which level "very high" popularity in genre movie, documentary, and foreign has the smallest amount. This means that not many movies have high popularity in that genres Go To List Question

2. What movie genre that associated with high revenue?

   This calculation just like the answer before, so lets make function to make it simple and reusable

list used function: Function get_class Function get_data_frame list function name: Function get_df_rank

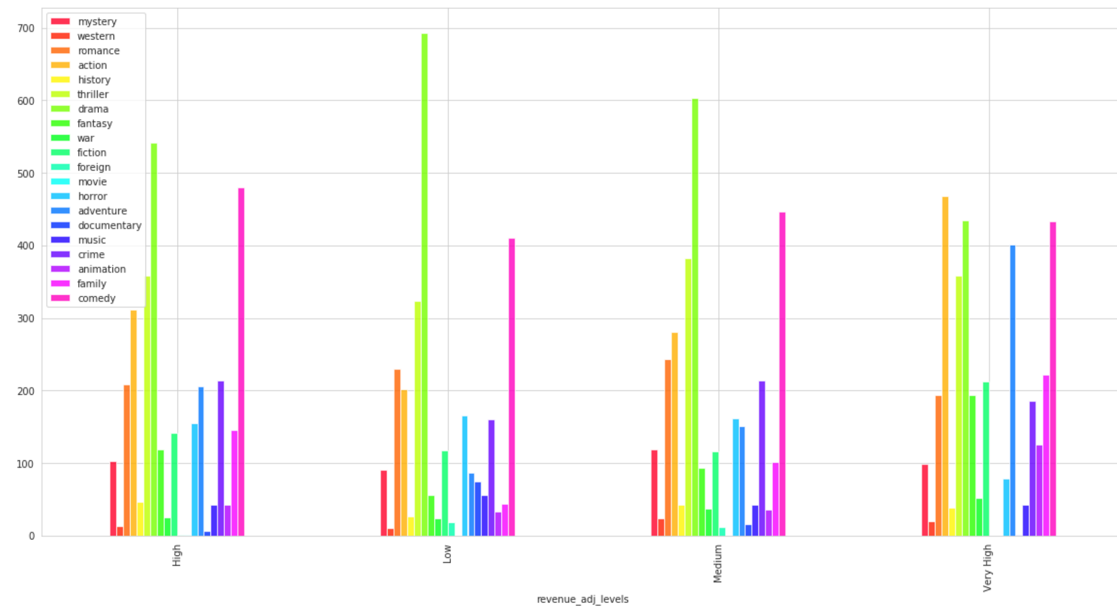
```
In [47]: def get_df_rank(df,column,metric):
             # make dataframe
             df_new = df.copy()
             # make class or level from metric
             df_new = get_class(df_new,metric)
             metric_name = '{}_levels'.format(metric) # declare the metric name to the next proc
             df_genre_new = get_data_frame(df_new.copy(),column,metric_name) # get the binary da

             # let's count the genre alomst use group by metric we decide before
             df_genre_new_rank = df_genre_new.groupby([metric_name]).sum() # make binary data to

             return df_genre_new_rank
```

```
In [48]: # lets call the function to get the df we want
         df_genre_revenue_rank = get_df_rank(df,'genres','revenue_adj')
         df_genre_revenue_rank.head(8)
```

```
Out[48]:                     mystery  western  romance  action  history  thriller  \
         revenue_adj_levels
         High                    102       13      209     311       47       358
         Low                      91       10      230     201       26       324
         Medium                  119       23      243     280       42       382
         Very High                98       19      194     468       38       358

                             drama  fantasy  war  fiction  foreign  movie  horror  \
         revenue_adj_levels
         High                  541      118   25      141        1      1     155
         Low                   693       56   23      117       18      0     166
         Medium                603       93   37      116       11      0     161
         Very High             434      194   52      212        0      0      79

                             adventure  documentary  music  crime  animation  family  \
         revenue_adj_levels
         High                      205            6     42    214         43     146
         Low                        86           74     56    160         33      44
         Medium                    151           16     43    214         35     101
```

```
Very High                          401              1       43      185           126       222

                        comedy
revenue_adj_levels
High                       480
Low                        411
Medium                     446
Very High                  433
```

In [49]:
```
# plot sum of genre use in each level of revenue
df_genre_revenue_rank.plot(kind='bar',figsize=(20,10),colormap='gist_rainbow',alpha=0.8
```



Answer Question Associate 2

from figure above we found that even always in all level o fpopularity (answer question associate 1) but in revenue genre that always appear in high distribution is horor. In very high revenue, genre documentary and foreign is not appear (or maybe too small) so its mean they dont have a big revenue. also in high level revenue, genre foreign is not appear but genre documentary is appear with small distribution. Go To List Question

3. What movie genre that associated with high vote average?
list used function: Function get_df_rank

In [50]:
```
# lets call the function to get the df we want
df_genre_vote_rank = get_df_rank(df,'genres','vote_average')
df_genre_vote_rank.head(8)
```

Out[50]:

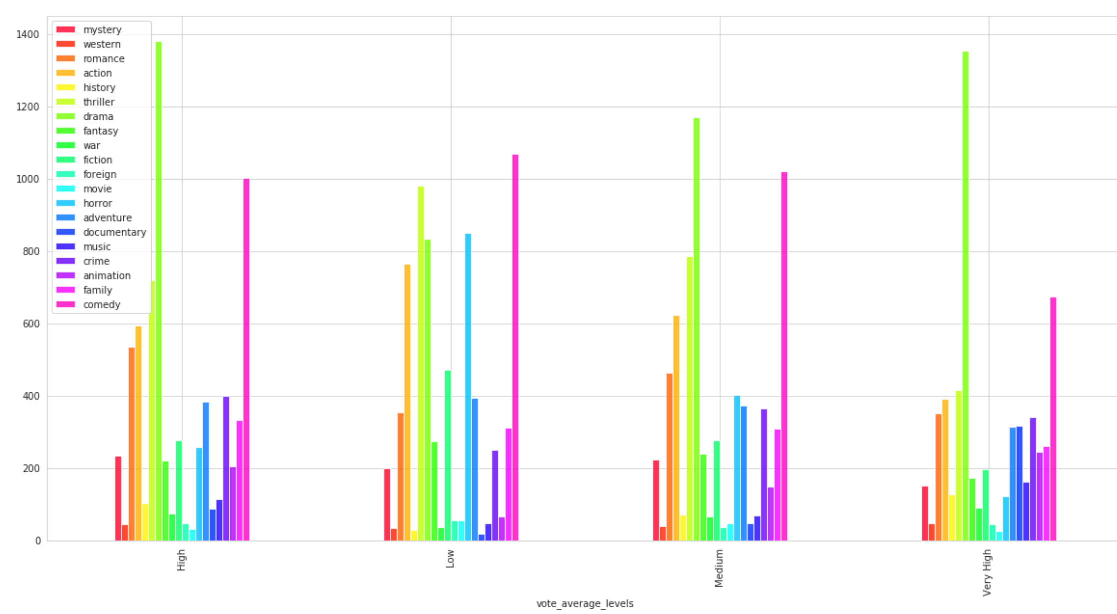
vote_average_levels	mystery	western	romance	action	history	thriller
High	233	45	534	595	104	719
Low	199	35	354	765	28	980
Medium	223	38	463	622	71	787
Very High	151	46	352	392	127	414

vote_average_levels	drama	fantasy	war	fiction	foreign	movie	horror
High	1381	220	75	276	47	31	257
Low	833	275	37	471	54	54	849
Medium	1170	239	65	277	37	47	401
Very High	1354	173	91	196	45	27	121

vote_average_levels	adventure	documentary	music	crime	animation	family
High	384	86	115	398	204	332
Low	393	18	48	249	67	311
Medium	373	47	68	364	149	308
Very High	314	316	163	341	244	261

vote_average_levels	comedy
High	1001
Low	1070
Medium	1020
Very High	674

In [51]:
```
# plot sum of genre use in each level of vote
df_genre_vote_rank.plot(kind='bar',figsize=(20,10),colormap='gist_rainbow',alpha=0.8).l
```



Answer Question Associate 3

from figure above we found that just like popularity level, in vote level drama still appear in all distribution. All genre drama have higher distribution except in low level. So its mean more drama movie have high vote. Just like popularity and revenue, comedy is in second place distribution in each vote level. Go To List Question

4. What movie genre that associated with high profit?
list used function: Function get_df_rank

In [52]:
```
# lets call the function to get the df we want
df_genre_profit_rank = get_df_rank(df,'genres','profit_adj')
df_genre_profit_rank.head(8)
```

Out[52]:

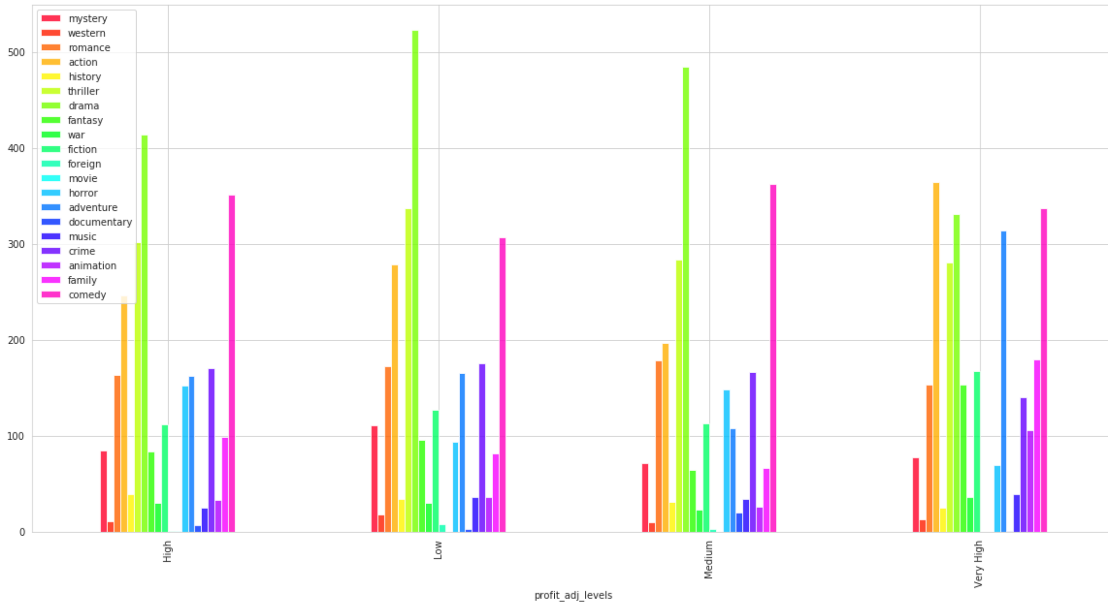
profit_adj_levels	mystery	western	romance	action	history	thriller
High	85	11	163	246	39	302
Low	111	18	172	278	34	337
Medium	71	10	178	197	31	284
Very High	77	13	153	364	25	280

profit_adj_levels	drama	fantasy	war	fiction	foreign	movie	horror
High	414	83	30	112	1	1	152
Low	523	96	30	127	8	0	94
Medium	485	64	23	113	3	0	148
Very High	331	153	36	167	0	0	69

profit_adj_levels	adventure	documentary	music	crime	animation	family
High	162	7	25	170	33	99
Low	165	3	36	175	36	81
Medium	108	20	34	166	26	66
Very High	314	1	39	140	106	179

profit_adj_levels	comedy
High	351
Low	307
Medium	362
Very High	337

In [53]:
```
# plot sum of genre use in each level of vote
df_genre_profit_rank.plot(kind='bar',figsize=(20,10),colormap='gist_rainbow',alpha=0.8)
```



Answer Question Associate 4

from figure above we found that just like the answer before, in profit level drama still appear in all distribution. All genre drama have higher distribution except in very high level. So its mean drama movie have good distribution in all profit level. Genre action have highest distribution in very high profit level, in another level that genre just in 4 posititon from higher distribution. Go To List Question

Trend Question
1. What is the trend of the genre every 10 years

In [54]:
```
# sort the movie release year list.
df_sub_year= df.release_year.unique()
df_sub_year= np.sort(df_sub_year)
df_sub_year
```

Out[54]:
```
array([1960, 1961, 1962, 1963, 1964, 1965, 1966, 1967, 1968, 1969, 1970,
       1971, 1972, 1973, 1974, 1975, 1976, 1977, 1978, 1979, 1980, 1981,
       1982, 1983, 1984, 1985, 1986, 1987, 1988, 1989, 1990, 1991, 1992,
       1993, 1994, 1995, 1996, 1997, 1998, 1999, 2000, 2001, 2002, 2003,
       2004, 2005, 2006, 2007, 2008, 2009, 2010, 2011, 2012, 2013, 2014,
       2015])
```

In [55]:
```
# make year list to make easy the next process
y1960s =df_sub_year[:10] # year list of 1960s
y1970s =df_sub_year[10:20] # year list of 1970s
y1980s =df_sub_year[20:30] # year list of 1980s
y1990s = df_sub_year[30:40] # year list of 1990s
y2000s = df_sub_year[40:50] # year list of afer 2000
y2010s = df_sub_year[50:] # year list of afer 2010
```

In [56]:
```
# year list devide by 10 years
times = [y1960s, y1970s, y1980s, y1990s, y2000s, y2010s]
# timesline name
names = ['1960s', '1970s', '1980s', '1990s', '2000s','2010s']
df['decade'] = np.nan # I make another metric to save the categorical data devide by de
for i in range(len(names)): # to fill the new metric "decade", do the following procedu
    index = df[df.release_year.isin(times[i])].index.values.tolist() # find list of idx
    for j in index: # for idx in list above, do the following procedure
        df.loc[j,'decade'] = names[i] # insert decade name to new metric we declare bef
df.head()
```

Out[56]:
```
       id  popularity       budget       revenue  \
0  135397   32.985763  150000000.0  1.513529e+09
1   76341   28.419936  150000000.0  3.784364e+08
2  262500   13.112507  110000000.0  2.952382e+08
3  140607   11.173104  200000000.0  2.068178e+09
4  168259    9.335014  190000000.0  1.506249e+09

                 original_title  \
0                Jurassic World
1            Mad Max: Fury Road
2                     Insurgent
3  Star Wars: The Force Awakens
4                     Furious 7

                                                cast          director  \
0  Chris Pratt|Bryce Dallas Howard|Irrfan Khan|Vi...   Colin Trevorrow
1  Tom Hardy|Charlize Theron|Hugh Keays-Byrne|Nic...     George Miller
2  Shailene Woodley|Theo James|Kate Winslet|Ansel...  Robert Schwentke
3  Harrison Ford|Mark Hamill|Carrie Fisher|Adam D...       J.J. Abrams
4  Vin Diesel|Paul Walker|Jason Statham|Michelle ...         James Wan

                                            keywords  runtime  \
0  monster|dna|tyrannosaurus rex|velociraptor|island    124.0
1   future|chase|post-apocalyptic|dystopia|australia    120.0
2  based on novel|revolution|dystopia|sequel|dyst...    119.0
3              android|spaceship|jedi|space opera|3d    136.0
4                car race|speed|revenge|suspense|car    137.0

                                      genres  \
0  Action|Adventure|Science Fiction|Thriller
1  Action|Adventure|Science Fiction|Thriller
2         Adventure|Science Fiction|Thriller
3   Action|Adventure|Science Fiction|Fantasy
4                      Action|Crime|Thriller

                                production_companies release_date  vote_count  \
0  Universal Studios|Amblin Entertainment|Legenda...   2015-06-09        5562
```

```
1  Village Roadshow Pictures|Kennedy Miller Produ...   2015-05-13        6185
2  Summit Entertainment|Mandeville Films|Red Wago...   2015-03-18        2480
3          Lucasfilm|Truenorth Productions|Bad Robot   2015-12-15        5292
4  Universal Pictures|Original Film|Media Rights ...   2015-04-01        2947

   vote_average  release_year    budget_adj   revenue_adj        profit  \
0           6.5          2015  1.379999e+08  1.392446e+09  1.363529e+09
1           7.1          2015  1.379999e+08  3.481613e+08  2.284364e+08
2           6.3          2015  1.012000e+08  2.716190e+08  1.852382e+08
3           7.5          2015  1.839999e+08  1.902723e+09  1.868178e+09
4           7.3          2015  1.747999e+08  1.385749e+09  1.316249e+09

     profit_adj decade
0  1.254446e+09  2010s
1  2.101614e+08  2010s
2  1.704191e+08  2010s
3  1.718723e+09  2010s
4  1.210949e+09  2010s
```

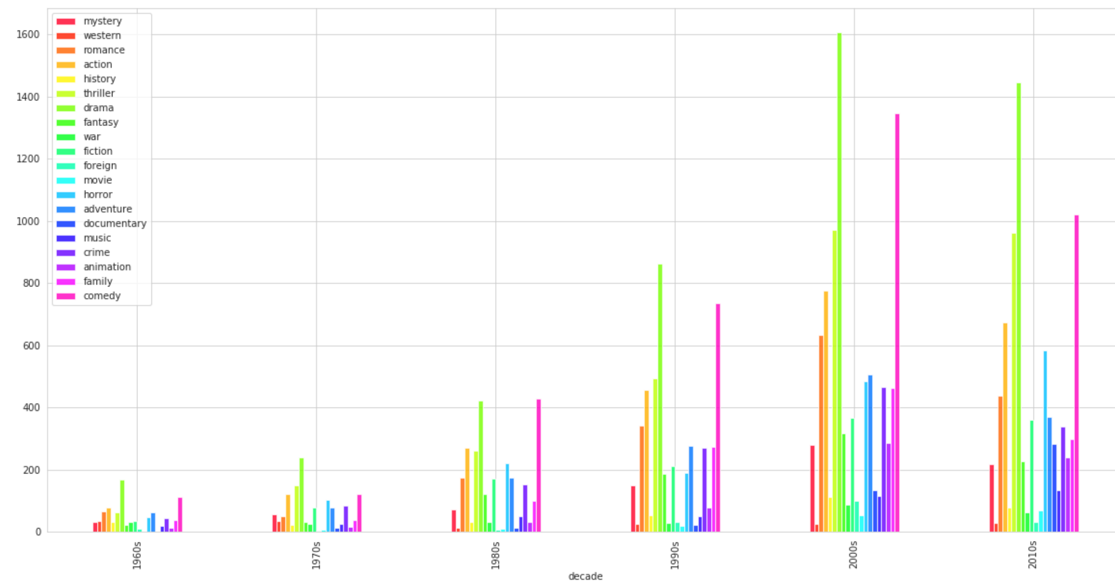
list used function: Function get_data_frame

In [57]:
```
df_genre_decade = get_data_frame(df.copy(),'genres','decade')
# let's count the genre alomst use group by metric we decide before
df_genre_decade_rank = df_genre_decade.groupby(['decade']).sum()
df_genre_decade_rank.head(8)
```

Out[57]:
```
        mystery  western  romance  action  history  thriller  drama  fantasy  \
decade
1960s        32       36       67      78       31        64    167       23
1970s        56       36       51     121       23       148    238       30
1980s        72       13      175     271       32       260    421      122
1990s       148       25      341     455       53       495    862      188
2000s       281       26      632     776      113       972   1605      318
2010s       217       28      437     673       78       961   1445      226

        war  fiction  foreign  movie  horror  adventure  documentary  music  \
decade
1960s    31       34        9      2      47         64            2     20
1970s    25       77        3      8     104         77           13     25
1980s    32      172        8      9     221        174           12     51
1990s    29      211       32     19     189        275           21     49
2000s    87      367       99     52     483        505          135    114
2010s    64      359       32     69     584        369          284    135

        crime  animation  family  comedy
decade
1960s      43         14      39     112
1970s      83         17      38     121
```

1980s	153	32	101	428
1990s	270	78	272	736
2000s	466	285	463	1346
2010s	337	238	299	1022

In [58]:
```
# plot sum of genre use in each decade
df_genre_decade_rank.plot(kind='bar',figsize=(20,10),colormap='gist_rainbow',alpha=0.8)
```



Answer Question Trend

from figure above we found that drama genre always have high distribution in every decade, genre western getting smaller in every decade. Genre foreign always have low distribution in every decade. Go To List Question

Limitation The limitation of this project are: 1. I assumed that Null and Zero value means missing value and other else is the right value From my mini research on Mr.Holmes movie, I found that zero value is not the real number. So I assumed that zero value in 'budget', 'revenue', 'runtime','budget_adj', and 'revenue_adj' means missing value and other else is the right value. Besides that, it still consists of a small and big value that maybe in there because of human error or other. Because of that maybe exist some bias in my research. 2. I choose to delete data with the count of row is not missing value is more than 95% from the real dataset I choose 95% as threshold because I didn't want to delete too much data. I don't want it because the closer number to the actual data is, the more analytical results must be closer also. 3. I bin the data level by quantile I choose quantile because I think it fairer, so movie is only compared to the fellow movie, not in the real range. Especially in popularity and vote average, I think that metric must have constant range but I don't know what the real range on that. 4. I'm not research about impact of the combined genre on metrics In my research, there is some bias because I do not calculate the effect of combined genre. So in some conclution maybe the large profit come from genre drama, but maybe it have large profit because that genre combines with fiction, or action, etc. 5. I'm not

counting people who vote that movie In my research, I just use vote average because I don't know the reason people want to vote or not. How if the people didn't vote because he/she agrees with another voter. Also, the vote count can not describe how much people see the movie.

Conclusions

The purpose of this research is to answer 3 parts of the question:

Part 1: General From this part we found that number of movie increasing every year. Movie with the highest profit is Avatar(2009), but if we check the inflation over time so the highest profit movie is Star Wars(1977) and the lowest profit movie is The Warrior's Way(2010). The Warrior's Way maybe get the lowest profit because it is movie with the highest budget. The lowest budget movie so far is Fear Clinic(2014). In this data we found the highest revenue movie is Avatar(2009), maybe it is reason that movie become the highest profit, but because the highest profit by inflation is Star Wars so we can conclude that budget Star Wars is bigger than Avatar (of course we assumed with inflation). The lowest revenue movie is Shattered Glass(2003). The longest runtime movie is The Story of Film: An Odyssey(2011) that is 900 minutes, its is make sense because it is documantary movie. The shortest runtime movie is Batman: Strange Days(2014) that is just run in 3 minutes. The highest popularity didn't mean the highest profit, but for level "very high" in popularity have highest profit. So if we want to make a highest profit movie we must make the movie get very high popularity levels, with minimum popularity is 0.710151. We also found that the highest level of vote average not always mean the movie get the highest profit, especially to 2010 which medium vote have higher profit than high and very high vote average.

Part 2: Find Associate Variable Movie Genre with Movie Metric From this part we found that genre drama are high distributed in all popularity level. Movies with genre "documentary", "movie", or "foreign" only few get "very high" popularity level. In revenue level, genre that always appear in high distribution is horor. In level very high revenue, genre documentary and foreign is not appear (or maybe too small) so its mean they don't have a big revenue. Also in high level revenue, genre foreign is not appear but genre documentary is appear with small distribution. In vote level, drama still appear in all distribution and have higher distribution except in low level vote. So its mean many drama movie have high vote. Just like popularity and revenue, comedy is in second place distribution in each vote level. In profit level, genre drama still appear in all distribution and also have higher distribution except in very high profit level. So its mean drama movie have good distribution in all profit level. Genre action have highest distribution in very high profit level, in another level that genre just in 4th positition from higher distribution.

Part 3: Find Some Trend From this part we found that drama genre always have high distribution in every decade, genre western getting smaller in every decade. Genre foreign always have low distribution in every decade.