

นายณัฐวุฒิ อกนิษฐ์ศักดิ์ 61070296

Task 1: API Chosen

ชื่อムกุลภาพยนตร์ภาษาเย็นใช้ The Movie Database (TMDB) ซึ่งเป็นเว็บที่เก็บข้อมูลของภาพยนตร์แต่ละเรื่องไว้ประกอบไปด้วย ค่าความเรียบ ประมาณหน่วย รันタイム เรื่องนี้ เป็นเรื่องเดียว อยู่ในกลุ่มภาษาญี่ปุ่นและตั้งให้ได้จากภาษาญี่ปุ่นเท่านั้น ไม่สามารถรับรู้ความเรียบง่ายได้ในช่วงเวลาโดยรวมผลลัพธ์จะแสดงเป็นรูปแบบแนวน้ำด้วย

- API Provider in <https://developers.themoviedb.org/3/getting-started/introduction>

Import Packages Etc.

```
In [2]: import numpy as np  
import pandas as pd  
import matplotlib.pyplot as plt  
import seaborn as sns  
import requests  
import json  
import urllib  
import csv
```

Define Web API key

```
In [3]: api_key = "7a9375b362fd6eaccb0c036bfa16b282"
```

Task 2: Collect data from chosen API(s)

ทำการกรอกข้อมูลจาก API โดยข้อมูลที่ได้จะเป็นรายละเอียดของภาษาญี่ปุ่นแต่จะเริ่งโดยมีการ Filter เลพาะภาษาญี่ปุ่นที่ใช้ภาษา en-US. เรียงค่าความนิยมจากมากสู่น้อย และทำการถอดแท้ในปี 2013 ที่มา

```
In [4]: url = "https://api.themoviedb.org/3/discover/movie?api_key=" + api_key + "&language=en-US&sort_by=popularity.desc&page=1&primary_release_date.gte:2010-01-01"
response = urllib.request.urlopen(url)
raw_json = response.read().decode("utf-8")
data = json.loads(raw_json)
data
```

```
Out[4]: {'page': 1,
 'total_results': 10000,
 'total_pages': 500,
 'results': [{"popularity": 2356.996,
  'vote_count': 1523,
  'video': False,
  'poster_path': '/ri1nlsq2kf1AWoGm80jQW5dLkp.jpg',
  'id': 497582,
  'adult': False,
  'backdrop_path': '/kMe4TKMDNXTKptQPAdOFoZHq3V.jpg',
  'original_language': 'en',
  'original_title': 'Enola Holmes',
  'genre_ids': [80, 18, 9648],
  'title': 'Enola Holmes',
  'vote_average': 7.7,
  'overview': "While searching for her missing mother, intrepid teen Enola Holmes uses her sleuthing skills to outsmart big brother Sherlock and help a run away lord.",
  'release_date': '2020-09-23'},
 {"popularity": 1792.051,
```

ทำการเก็บข้อมูลจาก API โดยข้อมูลที่ได้เป็นรายชื่อของประเทศภายใต้ ID ประเทศภายใต้รั้วบ้าน

```
In [5]: url = "https://api.themoviedb.org/3/genre/movie/list?api_key=" + api_key + "&language=en-US"
response = urllib.request.urlopen(url)
raw_json = response.read().decode("utf-8")
genres = json.loads(raw_json)
genres
```

```
Out[5]: {'genres': [{"id": 28, "name": "Action"}, {"id": 12, "name": "Adventure"}, {"id": 16, "name": "Animation"}, {"id": 35, "name": "Comedy"}, {"id": 80, "name": "Crime"}, {"id": 99, "name": "Documentary"}, {"id": 18, "name": "Drama"}, {"id": 10751, "name": "Family"}, {"id": 14, "name": "Fantasy"}, {"id": 36, "name": "History"}, {"id": 27, "name": "Horror"}, {"id": 10402, "name": "Music"}, {"id": 9648, "name": "Mystery"}, {"id": 10749, "name": "Romance"}, {"id": 878, "name": "Science Fiction"}, {"id": 10770, "name": "TV Movie"}, {"id": 53, "name": "Thriller"}, {"id": 10752, "name": "War"}, {"id": 37, "name": "Western"}]}
```

Task 3. Parse the collected data, and store it in an appropriate file format

เนื่องจากว่าก่อนหน้านี้ API ที่เข้าไปทำการเรียกเป็น Page รึไม่ได้จะต้องเป็น URL ที่มีพารามิเตอร์ที่ระบุตัวตนของหน้าที่ต้องการ เช่น `https://graph.facebook.com/v10.0/me/accounts?access_token=...`

```
In [6]: movie = []

for i in range(1,27):
    url = "https://api.themoviedb.org/3/discover/movie?api_key=' + api_key + '&language=en-US&sort_by=popularity.desc&page=' + str(i) + '&primary_release_date.gte=' + str(date)
    response = urllib.request.urlopen(url)
    raw_json = response.read().decode("utf-8")
    data = json.loads(raw_json)
    movie = movie + data["results"]
```

```

ทำการ Save ข้อมูลของภาพยนตร์ลง CSV

In [7]: csv_columns = ['popularity','vote_count','video','poster_path','id','adult','backdrop_path','original_language','original_title','genre_ids','title','vote
csv_file = "Movie_api.csv"
with open(csv_file, 'w', newline='', encoding="utf-8") as csvfile:
    writer = csv.DictWriter(csvfile, fieldnames=csv_columns)
    writer.writeheader()
    for data in movie:
        writer.writerow(data)

ทำการ Save ข้อมูลของ ID ที่มารายชื่อประเภทของภาพยนตร์ลง CSV

In [8]: csv_columns = ['id','name']
csv_file = "Genres_api.csv"
with open(csv_file, 'w', newline='', encoding="utf-8") as csvfile:
    writer = csv.DictWriter(csvfile, fieldnames=csv_columns)
    writer.writeheader()
    for data in genres['genres']:
        writer.writerow(data)

```

Task 4: Load and represent the data

ข้อมูลของแต่ละภาพยนตร์โดยที่ Column ต่อไปนี้

- ID รหัสของภาพยนตร์
- Popularity ค่าความนิยมของภาพยนตร์
- Vote Count จำนวนผู้ให้การโหวตภาพยนตร์
- Video ภาพเคลื่อนไหว Video ตัวอย่างหรือไม่
- Poster path ภาพ Poster
- Adult เนื้อหาของภาพยนตร์
- Backdrop path ภาพ Backdrop 1
- Backdrop path.1 ภาพ Backdrop 2
- Original language ภาษาของภาพยนตร์
- Original title ชื่อภาพยนตร์ในฉบับเดิม
- Genre ids ประเภทของภาพยนตร์
- Title ชื่อภาพยนตร์
- Vote average คะแนนของภาพยนตร์
- Overview ซึ่งย่อของภาพยนตร์
- Release date วันฉายของภาพยนตร์

```
In [9]: movies=pd.read_csv("Movie_api.csv",index_col="id")
movies.head(10)
```

```
Out[9]:
   popularity  vote_count  video          poster_path  adult  backdrop_path
id
497582      2356.996     1523   False  /mYlnsq2kf1AWoGm80JQW5dLk.jpg  False  /kMe4TKMDNXTKptQPAdOF0oZHq3V.jpg  /kMe4TKMDNXTKptQPAdOF0oZH
694919      1792.051      82   False  /6CoRTJTmhBLJTNUoSUNxZMEI.jpg  False  /pq0JSpwyt2URytdFG0euztQPAyR.jpg  /pq0JSpwyt2URytdFG0euztQP
623634      1448.288     39   False  /qOYf3L6eyCO027JoeM33pTJcNSR.jpg  False  /fbjzbWAUijs0fsQ6Hbls0NTAj.jpg  /fbjzbWAUijs0fsQ6Hbls0t
```

ข้อมูลของประเภทภาพยนตร์ที่สำคัญบนแปลง ID ของประเภทภาพยนตร์เป็นข้อมูลของภาพยนตร์โดยที่ Column ต่อไปนี้

- ID เป็นเลขรหัสของประเภทภาพยนตร์
- Name ชื่อของประเภทภาพยนตร์

```
In [10]: Genreslist=pd.read_csv("Genres_api.csv",index_col="id")
Genreslist.reset_index(inplace = True)
Genreslist
```

```
Out[10]:
   id      name
0   28      Action
1   12    Adventure
2   16     Animation
3   35     Comedy
4   80      Crime
5   99 Documentary
6   18      Drama
7  10751    Family
8   14      Fantasy
9   36      History
10  27      Horror
11  10402     Music
12  9648     Mystery
13  10749    Romance
14  878 Science Fiction
15 10770     TV Movie
16   53     Thriller
17  10752      War
18   37     Western
```

เคลียร์ Column ที่ไม่ได้นำมาใช้ในการวิเคราะห์และการแปลงประเภทของ Column

ทำการลบประเภทของแต่ละ Columns

```
In [11]: movies.dtypes
```

```
Out[11]: popularity      float64
vote_count       int64
video            bool
poster_path      object
adult            bool
backdrop_path    object
backdrop_path.1  object
original_language object
original_title   object
genre_ids        object
title            object
vote_average     float64
overview         object
release_date    object
dtype: object
```

เนื่องจาก Column release_date เป็น String จึงทำการแปลงเป็น datetime เพื่อใช้ในการสืบ ปี และ เดือน

```
In [12]: movies['datetime'] = pd.to_datetime(movies.release_date)
movies.drop(columns = 'release_date', inplace = True)
```

```
In [13]: movies.head(10)
```

	popularity	vote_count	video	poster_path	adult	backdrop_path	backdrop...
id							
497582	2356.996	1523	False	/nYlnsq2kf1AWoGm80JQW5dLkP.jpg	False	/kMe4TKMDNXTKptQPAadOF0oZHq3V.jpg	/kMe4TKMDNXTKptQPAadOF0oZH...
694919	1792.051	82	False	/6CoRTJTmhBLJTUNoVSUNxZMEI.jpg	False	/pq0JSpwyt2URytdFG0euztQPAyR.jpg	/pq0JSpwyt2URytdFG0euztQP...
623634	1448.288	39	False	/qOYf3L6eyCO027JoeM33pTjcNSR.jpg	False	/fbjzbWAUjs0fsQ6Hbls0NTAj.jpg	/fbjzbWAUjs0fsQ6Hbls0t...

ดูข้อมูลของ Column adult ซึ่งเป็นประเภท Boolean จะเห็นได้ว่ามีแต่ False จึงจะทำการลบ Column นี้ไป

```
In [14]: movies.groupby('adult').count()
```

```
Out[14]: popularity  vote_count  video  poster_path  backdrop_path  backdrop_path.1  original_language  original_title  genre_ids  title  vote_average  overview
adult
False      520        520       520        520        512        512        520        520        520        520        520        517
```

ดูข้อมูลของ Column video ซึ่งเป็นประเภท Boolean จะเห็นได้ว่ามีแต่ False จึงจะทำการลบ Column นี้ไป

```
In [15]: movies.groupby('video').count()
```

```
Out[15]: popularity  vote_count  poster_path  adult  backdrop_path  backdrop_path.1  original_language  original_title  genre_ids  title  vote_average  overview
video
False      520        520       520        520        512        512        520        520        520        520        520        517
```

ทำการลบ Column ที่ไม่ได้นำมาใช้ในการเริ่มคราฟ

```
In [16]: movies.drop(columns = 'adult', inplace = True)
movies.drop(columns = 'poster_path', inplace = True)
movies.drop(columns = 'video', inplace = True)
movies.drop(columns = 'backdrop_path', inplace = True)
movies.drop(columns = 'backdrop_path.1', inplace = True)
movies.drop(columns = 'overview', inplace = True)
movies.drop(columns = 'original_title', inplace = True)
movies.drop(columns = 'original_language', inplace = True)
```

```
In [17]: movies.head(10)
```

```
Out[17]: popularity  vote_count  genre_ids          title  vote_average  datetime
id
497582  2356.996  1523  [80, 18, 9648]  Enola Holmes  7.7  2020-09-23
694919  1792.051  82    [28]           Money Plane  5.9  2020-09-29
623634  1448.288  39    [18]          The Innocence  6.2  2020-01-10
621870  1258.996  100   [28, 12, 35, 14] Secret Society of Second Born Royals  7.2  2020-09-25
337401  1395.955  2257  [28, 12, 18, 14]          Mulan  7.4  2020-09-04
724989  1165.121  115   [28, 53]          Hard Kill  5.0  2020-08-25
539885  1045.033  418   [28, 80, 18, 53]          Ava  6.0  2020-07-02
734300  1042.148  123   [28]           Santana  5.6  2020-08-28
718444  931.488   284   [28]             Rogue  5.9  2020-08-20
438396  961.096   200   [18, 53]        Unknown Origins  6.2  2020-08-28
```

```
In [18]: movies.shape
```

```
Out[18]: (520, 6)
```

```
In [19]: movies.dtypes
```

```
Out[19]: popularity      float64
vote_count       int64
genre_ids        object
title            object
vote_average     float64
datetime         datetime64[ns]
dtype: object
```

หาก Null ของข้อมูล

```
In [20]: movies.isnull().sum()
```

```
Out[20]: popularity      0
          vote_count     0
          genre_ids      0
          title          0
          vote_average   0
          datetime       0
          dtype: int64
```

Calculate summary statistics for the dataset.

```
In [21]: movies.describe()
```

```
Out[21]:
```

	popularity	vote_count	vote_average
count	520.000000	520.000000	520.000000
mean	165.707292	2433.280769	6.563654
std	208.582412	4133.452306	1.267728
min	41.032000	0.000000	0.000000
25%	77.255500	59.500000	6.075000
50%	102.470000	348.000000	6.700000
75%	160.509500	3119.250000	7.300000
max	2356.996000	23844.000000	8.600000

ทำการ Save ข้อมูลเพื่อนการ Clear และมา Save เป็นไฟล์ CSV

```
In [22]: movies.to_csv ('Movie_Clean.csv', header=True)
```

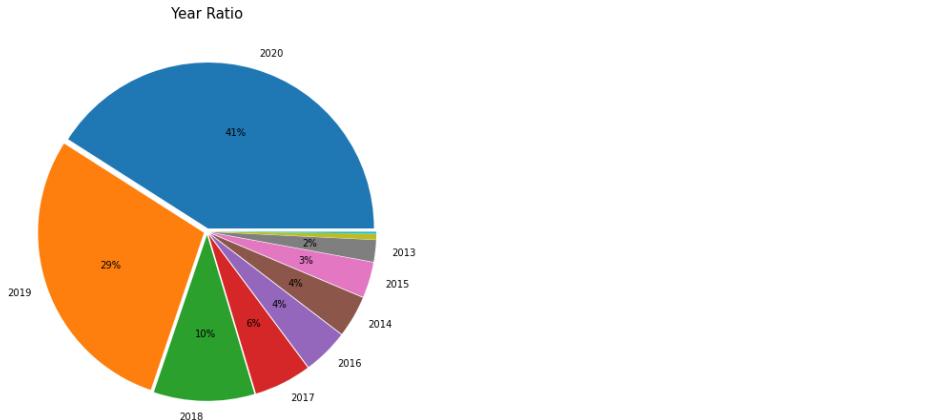
Task 5: Analyse and Summary

จาก API ที่นั่งดีอยู่มีความต้องการที่จะมองภาพโดยรวมต่อไปเรื่องจังหวัดที่มาจากว่าภาพนั้นจะเป็นประเทศใดเมืองใดที่สูงและมียอดหัวใจมากที่สุดโดยจะมีการแบ่งเป็นช่วง เวลาตามปีและค่าเฉลี่ย

จำนวนภาพนิทรรศ์ที่ทำการฉายในแต่ละปีโดยแบ่งเป็นสัดส่วน

โดยจำนวนที่ภาพนิทรรศ์ทำการฉายเพิ่มขึ้นชั่วปี 2013 ซึ่งจะเห็นได้ว่าจำนวนภาพนิทรรศ์ที่ฉายเพิ่มมากขึ้นทุกปีจนถึงปัจจุบัน

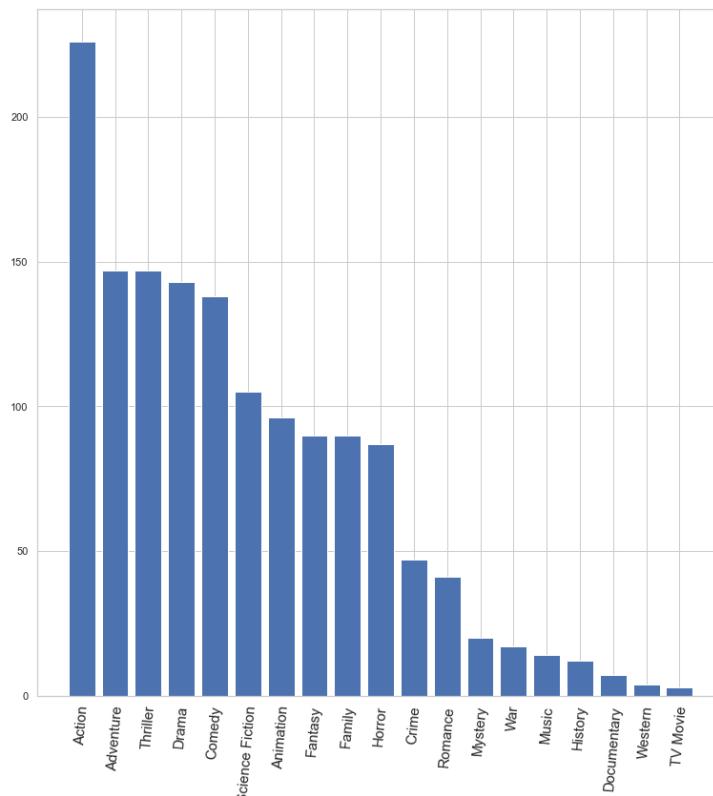
```
In [23]: count = movies.datetime.dt.year.value_counts(normalize=True)
count.reset_index()
labels = []
for i in count.index:
    if count[i] > 0.01: #ถ้าต่อส่วนของปีนั้นน้อยกว่า 0.01 จะไม่แสดงคลับบี้ในกราฟ
        labels.append(i)
    else:
        labels.append("0")
f, ax = plt.subplots(figsize=(8, 8))
plt.pie(count, labels=labels, autopct=lambda x:'{:1.0f}%'.format(x) if x > 1 else "", explode=[0.02, 0.02, 0.02, 0.02, 0.02, 0.02, 0.02, 0.02, 0.02], startangle=15)
plt.title('Year Ratio', fontsize=15)
plt.show()
```



จำนวนของภาพนิทรรศ์ในแต่ละประเภท

จากการลงทุนได้ว่าจำนวนของแต่ละประเภทของภาพนิทรรศ์รวมมากที่สุดคือประเภท Action ซึ่งจะเห็นได้ว่าภาพนิทรรศ์ประเภท Action นั้นมีมากที่สุดจากจำนวนประเภทภาพนิทรรศ์ที่มีจำนวนเยอะไปเรื่อยๆ คร่าวๆ ต่อ

```
In [36]: series = movies.genre_ids
sns.set(style="whitegrid")
d = {}
lis = []
for i in range(series.size):
    l=eval(series.loc[i])
    for x in l:
        d.setdefault(x,[])
        d[x].append(i)
        d[x].append(1)
        d[x].append(0)
for (k,v) in d.items():
    lis.append([(k, len(v))])#ทำการสรุป Len ที่มีจำนวนทั้งหมดของหนังประเภทนั้น
lis=sorted(lis, key=lambda x:x[1],reverse=True)
x = []
y = []
for i in lis: #นี้จะมาหาบทที่อยู่ใน Dataframe ของภาพนิทรรศ์ที่มี ID ซึ่งทำการแปลงเป็นชื่อประเภทของภาพนิทรรศ์ท่าน ID
    for j in range(Genreslist.shape[0]):
        if i[0] == Genreslist.id.iloc[j]:#ที่อยู่ที่ตรงกับประเภทของหนัง
            x.append(Genreslist.name.iloc[j])#ที่นี่จะขอประเภทของหนังใน List x
            y.append(i[1])
fig = plt.figure(figsize=(10, 10))
ax = fig.add_axes([0,0,1,1])
ax.bar(x,y)
plt.xticks(rotation=90, fontsize=15)
plt.show()
```

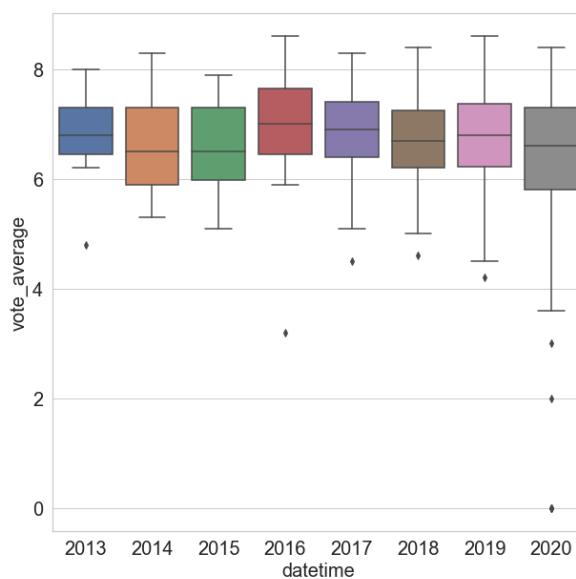


Vote Average ของแต่ละปี

โดยจะเห็นได้ว่าปี 2020 นั้นมียอด Vote ที่มีการกระจายมากที่สุดและมี Outliner ที่สูงที่สุดส่วนในปี 2013 นั้นเป็นปีที่ยอด Vote นั้นมีการกระจายน้อยที่สุดแต่ก็เป็นปีที่คลา Quartile ที่ 1 น้ำตกเท่ากับค่าเฉลี่ยของปี 2014 และปี 2015

```
In [37]: vote_ave = movies.loc[(movies.datetime.dt.year == 2020) | (movies.datetime.dt.year == 2019) |
                           | (movies.datetime.dt.year == 2018) | (movies.datetime.dt.year == 2017) |
                           | (movies.datetime.dt.year == 2016) | (movies.datetime.dt.year == 2015) |
                           | (movies.datetime.dt.year == 2014) | (movies.datetime.dt.year == 2013)] #ทำการน้ำตกข้อมูลตามปีที่ต้องการหากว่ารีเคราะห์
plt.figure(figsize=(10, 10))
plt.xlabel("", fontsize=20)
plt.ylabel("", fontsize=20)
plt.yticks(fontsize=20)
plt.xticks(fontsize=20)
sns.boxplot(x="vote_ave.datetime.dt.year", y="vote_average", data=vote_ave)
```

Out[37]: <matplotlib.axes._subplots.AxesSubplot at 0x1927d9c6190>



กราฟเปรียบเทียบระหว่างประเภทของภาพยนตร์ตามค่าความนิยม

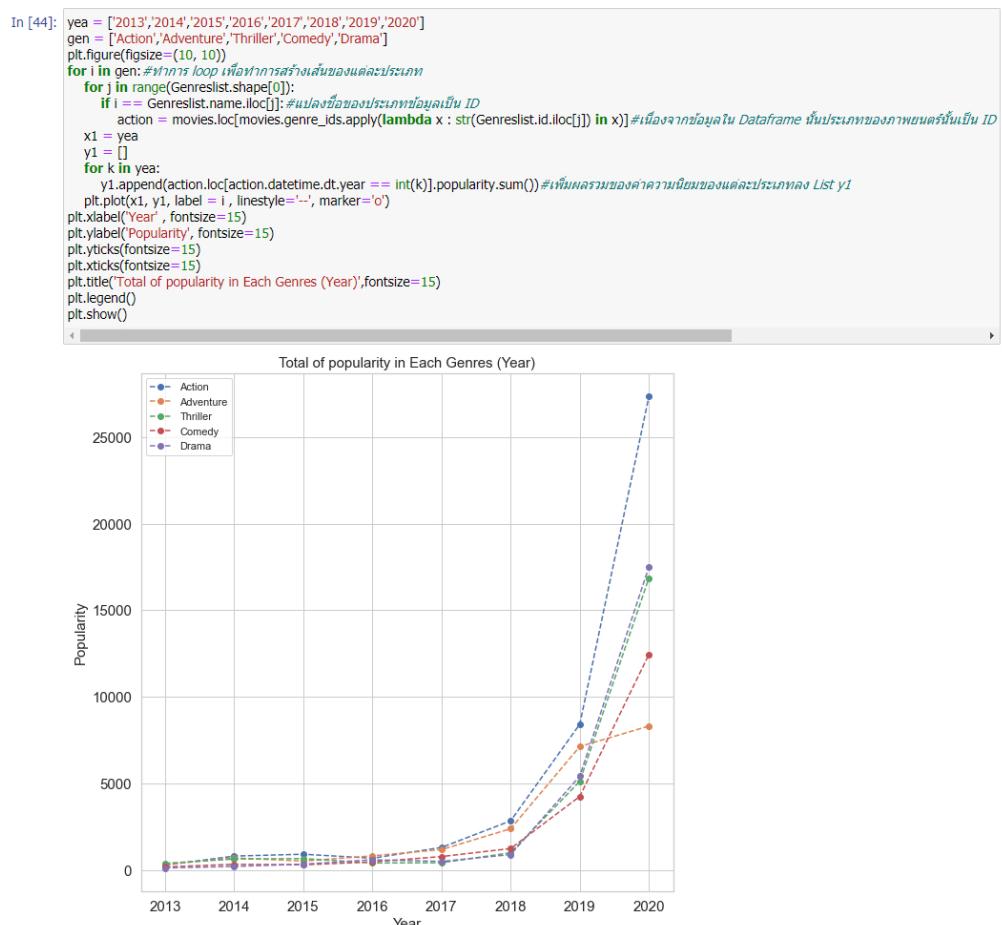
โดยจะเห็นว่าประเภท TV Movie นั้นค่าความนิยมต่ำที่สุดและมีค่ากลางที่สูงกว่าประเภทอื่นๆ เช่น Quatile ที่ 1 ที่สูงกว่าค่าเฉลี่ยของประเภทภาพยนตร์อื่นๆ และประเภท Action นั้นค่าความนิยมต่ำที่สุดแต่ค่าความนิยมที่สูงกว่าค่ากลางที่เทียบเท่ากับประเภทอื่นๆ และมี Outliner ที่สูงที่สุดเมื่อเทียบกับภาพยนตร์ประเภทอื่นๆ

```
In [38]: d={}
lis = []
for i in range(movies.genre_ids.size):
    l=eval(movies.genre_ids.iloc[i])
    for x in l:
        d.setdefault(x,[])
        d[x].append(i) #เพ้นท์ Id ของแต่ละประเภทเป็น Key โดยมี Value เป็น Index ของแต่ละเรื่องที่อยู่ในประเภทนั้นๆ
```



ผลรวมของค่าความนิยมของแต่ละประเภทตั้งแต่ปี 2013-2020

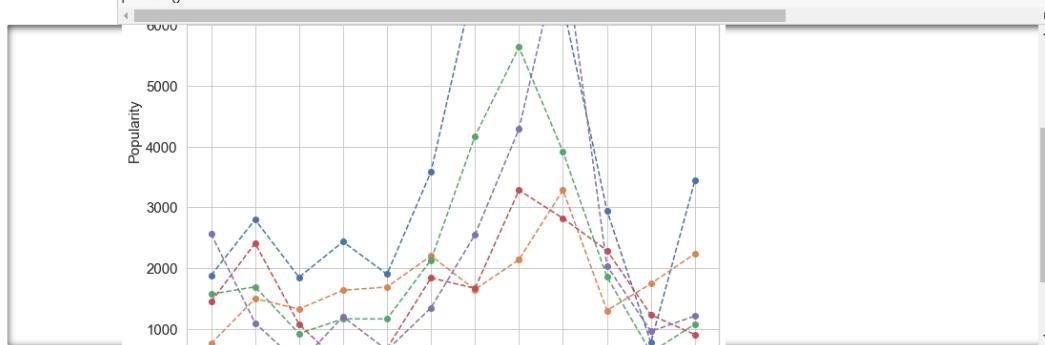
โดยประเภทของภาพยนตร์ที่มีประวัติ Action, Adventure, Thriller, Comedy และ Drama โดยเป็นประเภทที่มีจำนวนภาพยนตร์ที่สูงจากกราฟจะเห็นได้ว่าค่าความนิยมของแต่ละประเภทที่สูงขึ้นเรื่อยๆตามแนวโน้มที่ได้ไปในปี 2020 เป็นปีที่ค่าความนิยมสูงที่สุดและประเภทที่มีจำนวนภาพยนตร์น้อยที่สุดในปี 2019 ที่มีค่าความนิยมที่อยู่ก้าว Drama กับ Adventure และในปี 2020 กลับมีค่าความนิยมที่สูงกว่า 2 ประเภทเดียวกันประเภท Adventure ที่มีค่าความนิยมที่เพิ่มขึ้นอย่างสูงจาก 5 ประเภท



ผลรวมของค่าความนิยมของแต่ละประเภทในแต่ละเดือน

โดยจะเน้นไว้ในทุกๆเดือนนั้นกับภายนคร์ประเภท Action นั้นเมื่อความนิยมที่สูงกว่าประเภทอื่นๆทั้งหมดโดยในเดือนที่ 7 และ 8 นั้นมีภายนคร์ประเภท Thriller ที่มีความนิยมสูง และในเดือนที่ 9 ภาคภูมิประเภท Drama ก็มีความนิยมที่สูงเมื่อเทียบกับเดือนอื่นๆกับประเภทอื่นๆ

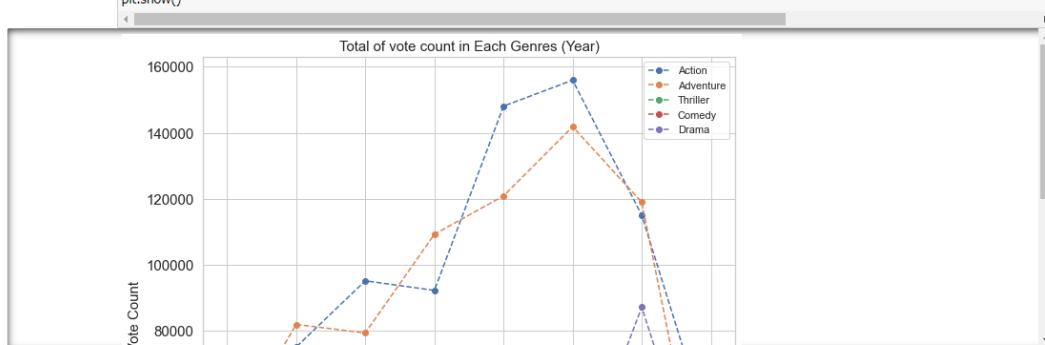
```
In [46]: yea = ['1','2','3','4','5','6','7','8','9','10','11','12']
gen = ['Action','Adventure','Thriller','Comedy','Drama']
plt.figure(figsize=(10, 10))
for i in gen: #ทำการ loop เพื่อทำการสร้างลิสต์ของแต่ละประเภท
    for j in range(Genreslist.shape[0]):#จะเลือกชื่อของประเภทที่อยู่ใน Genreslist
        if i == Genreslist.name.iloc[j]:#แปลงชื่อของประเภทที่อยู่ใน Genreslist
            action = movies.loc[movies.genre_ids.apply(lambda x : str(Genreslist.id.iloc[j]) in x)]#เมื่อจัดทำให้มูลใน Dataframe นั้นจะเก็บของภายนคร์ที่มีชื่อเป็น ID
            x1 = yea
            y1 = []
            for k in yea:
                y1.append(action.loc[action.datetime.dt.month == int(k)].popularity.sum())#เพิ่มผลรวมของความนิยมของแต่ละประเภทลง List y1
            plt.plot(x1, y1, label = i, linestyle='--', marker='o')
plt.xlabel('Month', fontsize=15)
plt.ylabel('Popularity', fontsize=15)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.title('Total of popularity in Each Genres (Month)', fontsize=15)
plt.legend()
plt.show()
```



ผลรวมยอดโหวตของแต่ละประเภทตั้งแต่ปี 2013-2020

โดยภายนคร์ประเภท Action ก็ยังมียอดโหวตมากกว่าประเภทอื่นๆ โดยในช่วงปีที่ผ่านมาหนึ่งปีประเภท Adventure ที่มียอดโหวตที่สูงมากเช่นกัน

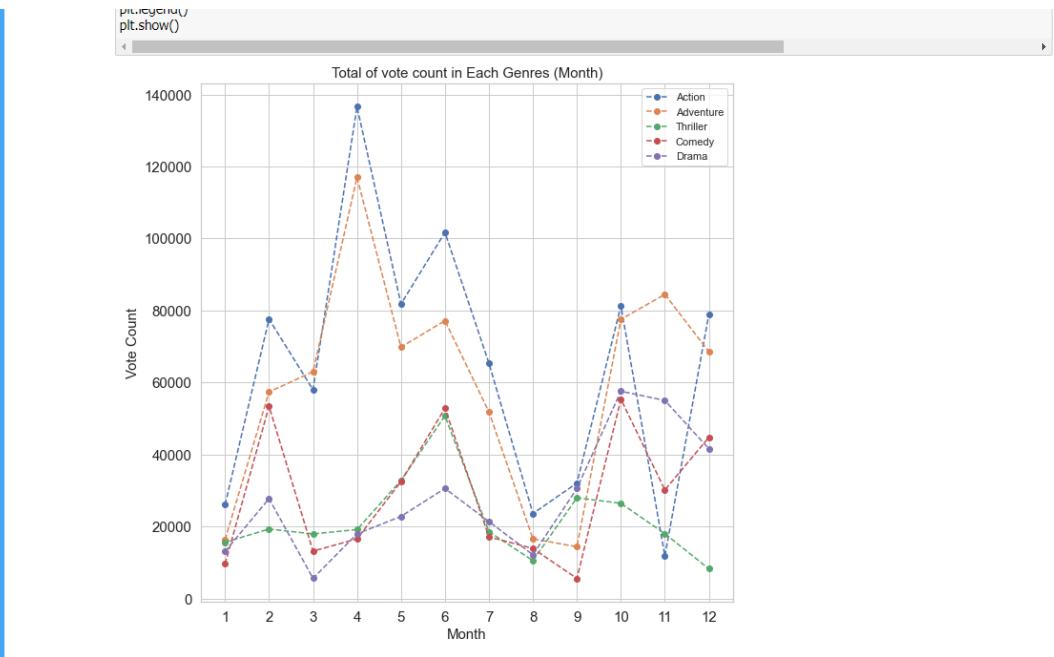
```
In [47]: yea = ['2013','2014','2015','2016','2017','2018','2019','2020']
gen = ['Action','Adventure','Thriller','Comedy','Drama']
plt.figure(figsize=(10, 10))
for i in gen: #ทำการ loop เพื่อทำการสร้างลิสต์ของแต่ละประเภท
    for j in range(Genreslist.shape[0]):#จะเลือกชื่อของประเภทที่อยู่ใน Genreslist
        if i == Genreslist.name.iloc[j]:#แปลงชื่อของประเภทที่อยู่ใน Genreslist
            action = movies.loc[movies.genre_ids.apply(lambda x : str(Genreslist.id.iloc[j]) in x)]#เมื่อจัดทำให้มูลใน Dataframe นั้นจะเก็บของภายนคร์ที่มีชื่อเป็น ID
            x1 = yea
            y1 = []
            for k in yea:
                y1.append(action.loc[action.datetime.dt.year == int(k)].vote_count.sum())#เพิ่มผลรวมของยอดโหวตของแต่ละประเภทลง List y1
            plt.plot(x1, y1, label = i, linestyle='--', marker='o')
plt.xlabel('Year', fontsize=15)
plt.ylabel('Vote Count', fontsize=15)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.title('Total of vote count in Each Genres (Year)', fontsize=15)
plt.legend()
plt.show()
```



ผลรวมยอดโหวตของแต่ละประเภทในแต่ละเดือน

โดยจะเน้นไว้ในเดือนที่ได้วางแผนกับภายนคร์ประเภท Action นั้นก็ยังมียอดโหวตที่สูงในแต่ละเดือนเมื่อเทียบกับประเภทอื่นๆ แต่ในเดือนที่ 11 ก็จะเป็นเดือนที่มียอดโหวตของภายนคร์ประเภท Action สำหรับเดือนที่ 5 ประเภทและก็เป็นประเภท Adventure ที่มียอดโหวตสูงที่สุด

```
In [48]: yea = ['1','2','3','4','5','6','7','8','9','10','11','12']
gen = ['Action','Adventure','Thriller','Comedy','Drama']
plt.figure(figsize=(10, 10))
for i in gen: #ทำการ loop เพื่อทำการสร้างลิสต์ของแต่ละประเภท
    for j in range(Genreslist.shape[0]):#จะเลือกชื่อของประเภทที่อยู่ใน Genreslist
        if i == Genreslist.name.iloc[j]:#แปลงชื่อของประเภทที่อยู่ใน Genreslist
            action = movies.loc[movies.genre_ids.apply(lambda x : str(Genreslist.id.iloc[j]) in x)]#เมื่อจัดทำให้มูลใน Dataframe นั้นจะเก็บของภายนคร์ที่มีชื่อเป็น ID
            x1 = yea
            y1 = []
            for k in yea:
                y1.append(action.loc[action.datetime.dt.month == int(k)].vote_count.sum())#เพิ่มผลรวมของยอดโหวตของแต่ละประเภทลง List y1
            plt.plot(x1, y1, label = i, linestyle='--', marker='o')
plt.xlabel('Month', fontsize=15)
plt.ylabel('Vote Count', fontsize=15)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.title('Total of vote count in Each Genres (Month)', fontsize=15)
```



Summary

โดยสรุปแล้ว ภาคบันยนต์ประเภท Action เป็นภาคบันยนต์ที่มีความนิยมและยอดโหวตมากที่สุดเมื่อเทียบกับประเภทอื่นๆ แต่ก็เป็นประเภทที่ภาคบันยนต์ล่ามมากเย็นไว้ด้วย ประเภท Adventure ก็เป็นอีกหนึ่งประเภทที่มีความนิยมที่สูงของลุมจากประเภท Action เท่ากันภาคบันยนต์ประเภท Action และ Adventureเป็นประเภทที่มีโอกาสประสบความสำเร็จมากที่สุดโดยควรทำการถ่ายในช่วงเดือน 7 ถึง 9 นี้จะจากเป็นเดือนที่มีความนิยมที่สูงที่สุดและต่อมาในเดือนที่ 10 ถึง 12

อีกประเภทภาคบันยนต์ที่หน้าสนใจการท่องเที่ยวประเภท Thriller เมื่อจากในช่วงปี 2019 ที่มีความนิยมสูงสันในที่น้อยกว่าประเภทอื่นๆ แต่กลับในปี 2020 กับเป็นประเภทที่มีความนิยมสูงเป็นอันดับ 2 รองจาก Action และในช่วงปี 2019 ไป 2020 ก็ยังมียอดโหวตที่สูงขึ้นเมื่อเทียบกับประเภทอื่นๆ