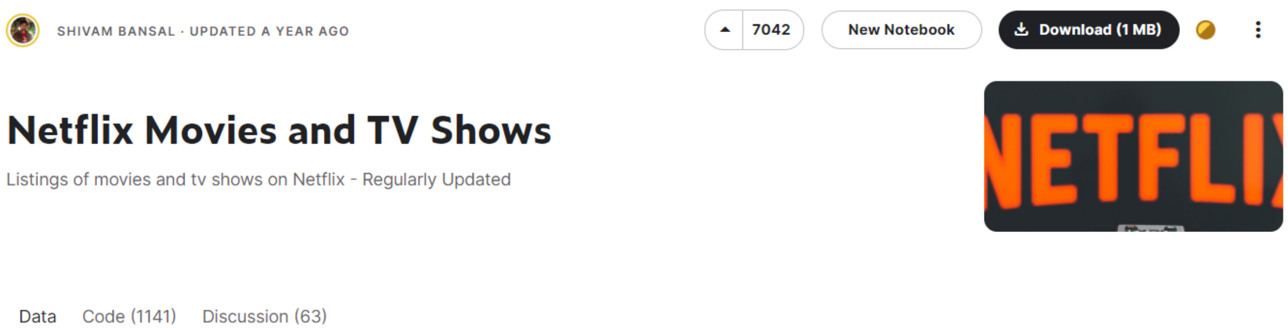
大數據分析期末報告

一、資料庫

名稱:Netflix Movies and TV Shows

簡介:Netflix 是最受歡迎的影片媒體平台之一，在全球擁有超過 2 億訂閱者， 平台上有超過 8000 部電影或電視節目。本資料集為 Netflix 上所有(～2021)的電影Movies和電視節目TV Shows、演員、導演、國家、影片風格、發行年份、片長等資訊。



Dataset: https://www.kaggle.com/datasets/shivamb/netflix-shows
Code:https://drive.google.com/drive/folders/1bDaESvjnrNgqkOci-ah62itXAVOYwbGK?usp=sharing
Netflix: https://www.netflix.com/tw/

組員：
M11007324 林政皓
M11010113 張菁蘭
M11015080 湯傑堯

二、摘要總覽

分析問題	關鍵數據	採用方法	實驗結果
近年片長與評分的關係	Type Date_added	分析工具: Percentage rating	依據影片在Netflix的上映年份與影片類型進行排序，找出每年被上架影片量的趨勢與類型占比
近年影片發行年份與影片風格的關係	Release_year Listed_in Type	分析工具: Sorting	計算近五年發行的影片各自在電影和節目兩大類的風格的總數，找出數量最多的前三種風格。
推薦系統—Content Based Filtering(基於電影屬性)	description listed_in	分析工具: FAISS TFIDF	使用將不同欄位組成電影關鍵字，利用不同的方式來進行處理，最後再做cosine similarity計算相似度，找出與所搜尋的內容相似度最高的前10個結果。

三、說明

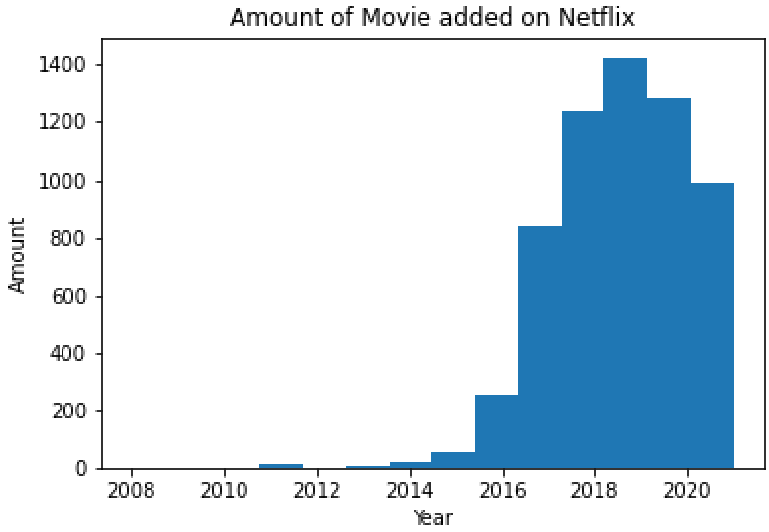
(一) 推薦系統—全體平均

Type：影片種類 (Movie, TV Show)

Date_added：影片被加入Netflix的時間

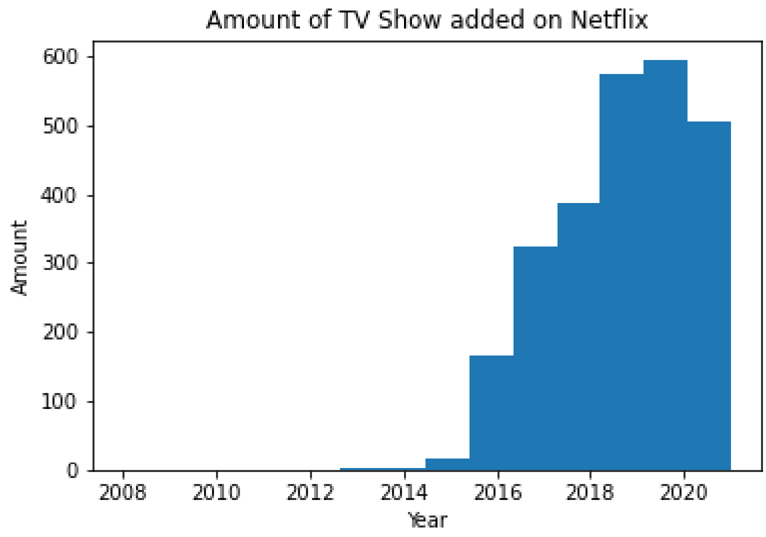
每年上架之 Movie 總數

```
a = df[df['type']=='Movie']['Year']
plt.hist(a.astype(int),bins=14)
plt.xlabel('Year')
plt.ylabel('Amount')
plt.title('Amount of Movie added on Netflix')
```



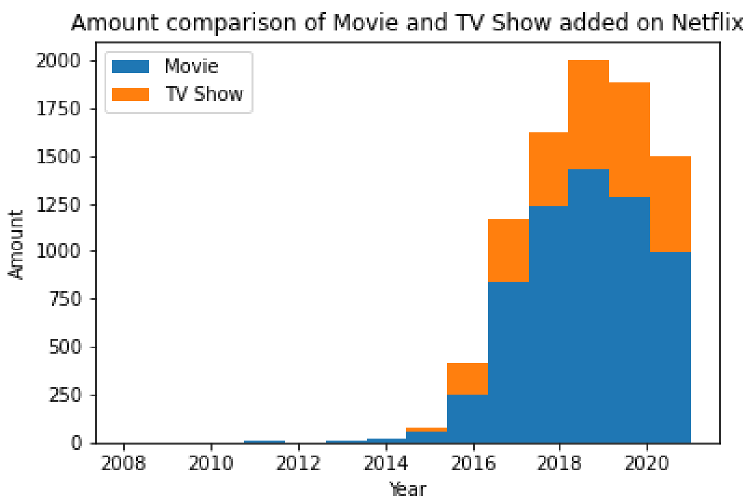
每年上架之 TV Show 總數

```
b = df[df['type']=='TV Show']['Year']
plt.hist(b.astype(int),bins=14)
plt.xlabel('Year')
plt.ylabel('Amount')
plt.title('Amount of TV Show added on Netflix')
```



每年上架之 Movie 與 TV Show 比較圖

```
c = [a.astype(int),b.astype(int)]
plt.hist(c, histtype='barstacked', bins=14)
plt.xlabel('Year')
plt.ylabel('Amount')
plt.ylabel('Amount')
plt.legend(['Movie','TV Show'])
plt.title('Amount comparison of Movie and TV Show added on Netflix')
```



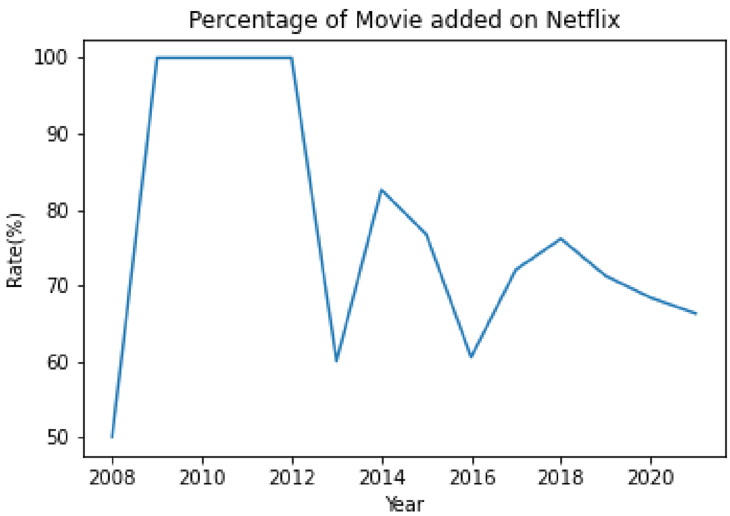
每年上架之 Movie 在全影片中的占比

分析工具:

使用 （上架Movie量）除以（總上架影片量）計算

每年上架之 Movie 在全影片中的占比

```
x = []

for i in range(2008,2022):
    df_year = df[df['Year']==str(i)]
    arr = df_year['type']=='Movie'
    x.append(sum(arr == 1)/(sum(arr == 1)+sum(arr == 0)))

plt.plot(range(2008,2022),np.array(x).astype(float)*100)
plt.xlabel('Year')
plt.ylabel('Movie')
plt.ylabel('Rate(%)')
plt.title('Percentage of Movie added on Netflix')
plt.savefig('Percentage of Movie.png')
```



(二) Netflix影片風格

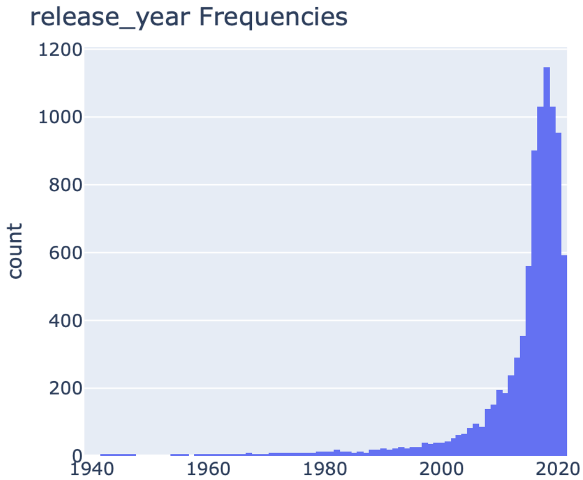
關鍵欄位:

Type 影片種類 (Movie, TV Show)

Release_year 影片的發行年份

Listed_in 影片風格

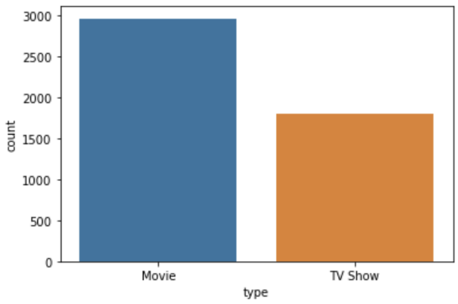
影片發行總數(依據發行年份)



近五年影片 (Movie, TV Show)發行總數

```
df_2017 =df[df['release_year'] >= 2017]
```

```
ax = sns.countplot(x="type", data = df_2017)
```



查看影片風格類型與總數

```
listed1 = [j for i in movies['listed_in'] for j in i.split(',')]
listed2 = [i.strip() for i in listed1]
listed_in = []
for i in tqdm(listed2):
    if i not in listed_in:
        listed_in.append(i)
```

```
100%|████████████████████████████████| 19323/19323 [00:00<00:00, 1484205.70it/s]
```

```
print(listed_in)
```

```
['Documentaries', 'International TV Shows', 'TV Dramas', 'TV Mysteries', 'Crime TV Shows', '
TV Action & Adventure', 'Docuseries', 'Reality TV', 'Romantic TV Shows', 'TV Comedies', 'TV
Horror', 'Children & Family Movies', 'Dramas', 'Independent Movies', 'International Movies',
'British TV Shows', 'Comedies', 'Spanish-Language TV Shows', 'Thrillers', 'Romantic Movies',
'Music & Musicals', 'Horror Movies', 'Sci-Fi & Fantasy', 'TV Thrillers', "Kids' TV", 'Action
& Adventure', 'TV Sci-Fi & Fantasy', 'Classic Movies', 'Anime Features', 'Sports Movies', 'A
nime Series', 'Korean TV Shows', 'Science & Nature TV', 'Teen TV Shows', 'Cult Movies', 'TV
Shows', 'Faith & Spirituality', 'LGBTQ Movies', 'Stand-Up Comedy', 'Movies', 'Stand-Up Comed
y & Talk Shows', 'Classic & Cult TV']
```

```
print(len(listed_in))
```

```
42
```

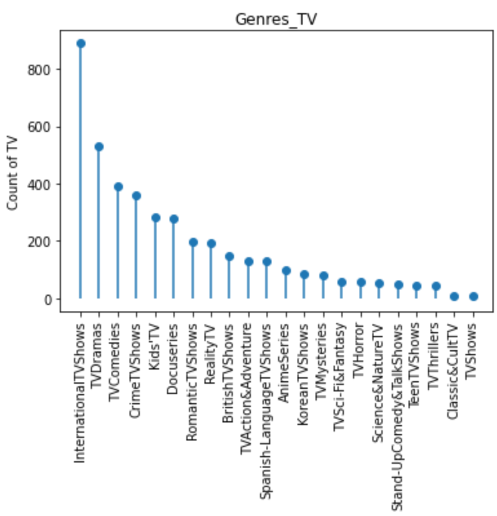
近五年Movie與TV Show個別發行數量排序

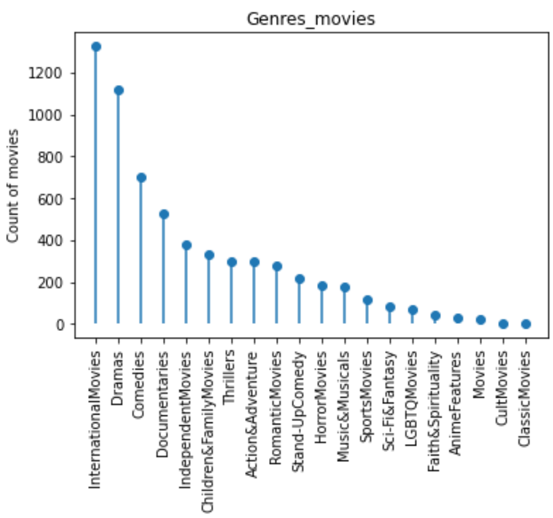
```python
genres_movie = list(netflix_Movie['listed_in'])
genres_TV = list(netflix_TV['listed_in'])
gen_movie=[]
gen_TV=[]

for i in genres_movie:
    i=list(i.split(','))
    for j in i:
        gen_movie.append(j.replace(' ',""))

for i in genres_TV:
    i=list(i.split(','))
    for j in i:
        gen_TV.append(j.replace(' ',""))
g_movie = Counter(gen_movie)
g_tv = Counter(gen_TV)
```

```python
text_movie = list(set(gen_movie))
text_tv = list(set(gen_TV))
```

```python
g_movie={k: v for k, v in sorted(g_movie.items(), key=lambda item: item[1], reverse= True)}
fig, ax = plt.subplots()

fig = plt.figure(figsize = (10, 10))
x=list(g_movie.keys())
y=list(g_movie.values())
ax.vlines(x, ymin=0, ymax=y)
ax.plot(x,y, "o")
ax.set_xticklabels(x, rotation = 90)
ax.set_ylabel("Count of movies")
ax.set_title("Genres_movies")
```

Movie與TV Show數量排序





結果:近五年Movie與TV Show個別發行數量最多的前三種風格。

```python
#TVshow一數量前三筆風格
first_three_TVshow= list(g_tv.keys())[:3]
print(first_three_TVshow)
```

```
['InternationalTVShows', 'TVDramas', 'TVComedies']
```

```python
first_three_Movies= list(g_movie.keys())[:3]
print(first_three_Movies)
```

```
['InternationalMovies', 'Dramas', 'Comedies']
```

(三) 推薦系統— Content Based Filtering(基於電影屬性)

根據 Movies或TV shows的內容風格(影片類別)推薦近十年的前部類似的影片

關鍵欄位：Release_year 影片的發行年份 , Listed_in 影片風格

分析工具:tf-idf, cosine similarity

```
movies_2011 =movies[movies['release_year'] > 2011]
```

篩選近十年的影片

透過listed_in的資訊進行分析後，可以得到每一個listed在整體資料集中每個字的重要性，之後在利用相似度的計算找到每一部電影或影集之間的關係，從而進行推薦。

結果:

輸入一部影片可以找到和該部影片風格相似的 10 部電影。

```
#Movies
get_recommendation('Ganglands')
```

```
11              Bangkok Breaking
543                   Undercover
734                        Lupin
1223                      Dealer
2676                       Fauda
3356                 Nowhere Man
3414                      Chosen
3976       The Eagle of El-Se'eed
4662                Monkey Twins
4752                     Smoking
```

```
#TVSHOWS
get_recommendation('Castle and Castle')
```

```
997      Life in Color with David Attenborough
1129          Secrets of Great British Castles
1603                              Alien Worlds
1885                 Bad Boy Billionaires: India
2476                               History 101
2742                     Sunderland 'Til I Die
3294           Greatest Events of WWII in Colour
4183      The World's Most Extraordinary Homes
5030                      Hitler's Circle of Evil
5251                 The Royal House of Windsor
```

關鍵欄位

description 描述

listed_in 電影風格

```python
features = ['description','listed_in']
for feature in features:
    netflix_movies[features] = netflix_movies[features].fillna('')
def combine_features(row):
    return row['description']+" "+row['listed_in']
netflix_movies["combined_features"] = netflix_movies.apply(combine_features, axis = 1)
```

先將 desription 及 listed_in合併成合併成新的一欄，名稱為 combined_feature。

```python
!pip install sentence_transformers ## Installs the library
##Extracting list of combined_features
sentences=netflix_movies['combined_features'].tolist()
from sentence_transformers import SentenceTransformer, util
##Loading the pretrained distil Roberta model
model = SentenceTransformer('paraphrase-distilroberta-base-v1')
## This model gives us 768 Dimension Vector
embeddings=model.encode(sentences) ## Extract the sentenceembeddings
```

載入 pretrained model，將我們新合併的一欄 combined_feature，丟到此 model 擷取 combined_feature 的 sentenceembeddings。

最後透過這個 searchFAISSIndex function，將我們的 query 先轉換成要進行 cosine_similarity 計算的 sentenceembedding。

透過將 query 轉換後的 sentenceembedding 以及剛剛先轉換的 sentenceembeddings 兩兩進行cosine_similarity 的計算，來得到哪些電影的描述是跟我們query較為相像。

為了將此方法與 TFIDF 做比較，我們使用相同的 query 來進行實驗。

並且在 TFIDF 中，一樣也選定相同欄位作為特徵。

```python
query="A seventeen-year-old aristocrat falls in love with a kind but poor artist"
```

以下三個是使用 FAISS 方法所得到的結果：

id	description	title	listed_in	cosine_sim
4722	A young artist falls for an aristocratic young...	Fitoor	Dramas, International Movies, Romantic Movies	0.685089
8509	When a young man leaves home to fulfill the wi...	The Sinking Of Van Der Wijck	Dramas, International Movies, Romantic Movies	0.631640
7243	A young woman talented at traditional dance fi...	Kuppivala	Dramas, International Movies, Romantic Movies	0.627530

index4722： A young artist falls for an aristocratic young woman whose bitter mother has trained her in the art of breaking hearts.
listed_in： Dramas, International Movies, Romantic Movies
cosine_sim： 0.6850886696038003

index8509： When a young man leaves home to fulfill the wishes of his late father, he meets and falls in love with a woman from a very different background.
listed_in： Dramas, International Movies, Romantic Movies
cosine_sim： 0.6316401164608294

index7243： A young woman talented at traditional dance finds her life changed when her love for a man clashes with the wishes of her father. **listed_in**： Dramas, International Movies, Romantic Movies
cosine_sim： 0.6275296646700141

以下三個是使用 IFIDF 方法所得到的結果：

index	listed_in	description	cosine_sim
7824	Dramas, International Movies, Romantic Movies	When a poor taxi driver falls in love with a w...	0.222315
7437	Dramas, International Movies, Romantic Movies	After recovering from a tragic experience, a y...	0.199183
3688	International TV Shows, Romantic TV Shows, TV ...	A kind computer repairman falls for a street-s...	0.188779

index7824： When a poor taxi driver falls in love with a wealthy young woman, he must stand up to her family and contend with his own insecurities.
listed_in： Dramas, International Movies, Romantic Movies
cosine_sim： 0.22231542739242505

index7437： After recovering from a tragic experience, a young woman makes a journey to her father's homeland and falls in love with a kind-hearted doctor.
listed_in： Dramas, International Movies, Romantic Movies
cosine_sim： 0.1991832451263083

index3688： A kind computer repairman falls for a street-smart graffiti artist whose multiple personality disorder worsens after she witnesses a double murder.
listed_in： International TV Shows, Romantic TV Shows, TV Dramas
cosine_sim 0.1887788165267757