[데이터스토리] 데이터로 보는 개봉 영화 (1) 장르

0. 라이브러리 업그레이드 및 한글 사용 설정

0.1. Matplotlib, seaborn, pandas

In [ ]:

```
# Step 1. Matplotlib 업그레이드
!pip install matplotlib -U
!pip install seaborn -U
!pip install pandas -U

# Step 2. 한글 설치 및 사용 설정
!sudo apt-get install -y fonts-nanum
!sudo fc-cache -fv
!rm ~/.cache/matplotlib -rf

# Step 3. 셀 실행 후 런타임 재시작
```

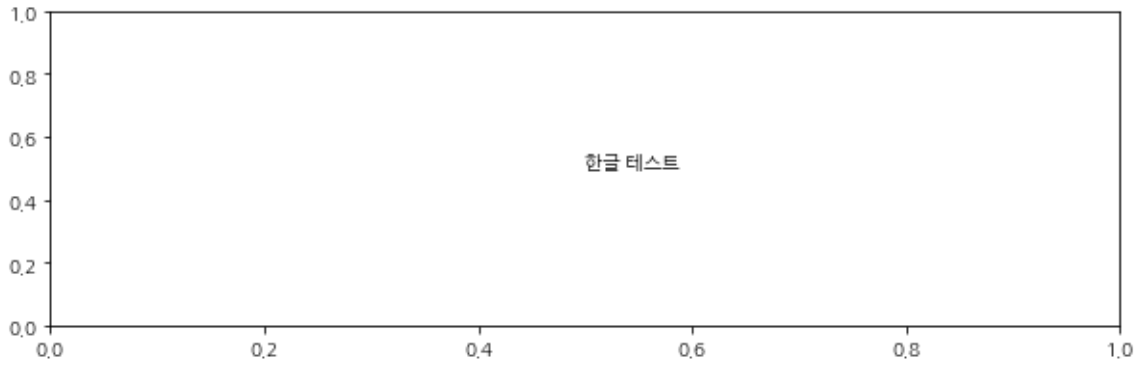
```
Requirement already satisfied: matplotlib in /usr/local/lib/python3.7/dist-package
s (3.2.2)
Collecting matplotlib
  Downloading matplotlib-3.4.3-cp37-cp37m-manylinux1_x86_64.whl (10.3 MB)
     |████████████████████████████████| 10.3 MB 29.3 MB/s
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-
packages (from matplotlib) (1.3.2)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.7/dist-pack
ages (from matplotlib) (7.1.2)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.7/di
st-packages (from matplotlib) (2.8.2)
Requirement already satisfied: numpy>=1.16 in /usr/local/lib/python3.7/dist-packag
es (from matplotlib) (1.19.5)
Requirement already satisfied: pyparsing>=2.2.1 in /usr/local/lib/python3.7/dist-p
ackages (from matplotlib) (2.4.7)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-packa
ges (from matplotlib) (0.10.0)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from
cycler>=0.10->matplotlib) (1.15.0)
Installing collected packages: matplotlib
  Attempting uninstall: matplotlib
    Found existing installation: matplotlib 3.2.2
    Uninstalling matplotlib-3.2.2:
      Successfully uninstalled matplotlib-3.2.2
ERROR: pip's dependency resolver does not currently take into account all the pack
ages that are installed. This behaviour is the source of the following dependency
 conflicts.
albumentations 0.1.12 requires imgaug<0.2.7,>=0.2.5, but you have imgaug 0.2.9 whi
ch is incompatible.
Successfully installed matplotlib-3.4.3
```

```
Requirement already satisfied: seaborn in /usr/local/lib/python3.7/dist-packages
(0.11.2)
Requirement already satisfied: numpy>=1.15 in /usr/local/lib/python3.7/dist-packag
es (from seaborn) (1.19.5)
Requirement already satisfied: matplotlib>=2.2 in /usr/local/lib/python3.7/dist-pa
ckages (from seaborn) (3.4.3)
Requirement already satisfied: scipy>=1.0 in /usr/local/lib/python3.7/dist-package
s (from seaborn) (1.4.1)
Requirement already satisfied: pandas>=0.23 in /usr/local/lib/python3.7/dist-packa
ges (from seaborn) (1.1.5)
Requirement already satisfied: pyparsing>=2.2.1 in /usr/local/lib/python3.7/dist-p
ackages (from matplotlib>=2.2->seaborn) (2.4.7)
Requirement already satisfied: python-dateutil>=2.7 in /usr/local/lib/python3.7/di
st-packages (from matplotlib>=2.2->seaborn) (2.8.2)
Requirement already satisfied: kiwisolver>=1.0.1 in /usr/local/lib/python3.7/dist-
packages (from matplotlib>=2.2->seaborn) (1.3.2)
Requirement already satisfied: pillow>=6.2.0 in /usr/local/lib/python3.7/dist-pack
ages (from matplotlib>=2.2->seaborn) (7.1.2)
Requirement already satisfied: cycler>=0.10 in /usr/local/lib/python3.7/dist-packa
ges (from matplotlib>=2.2->seaborn) (0.10.0)
Requirement already satisfied: six in /usr/local/lib/python3.7/dist-packages (from
cycler>=0.10->matplotlib>=2.2->seaborn) (1.15.0)
Requirement already satisfied: pytz>=2017.2 in /usr/local/lib/python3.7/dist-packa
ges (from pandas>=0.23->seaborn) (2018.9)
Requirement already satisfied: pandas in /usr/local/lib/python3.7/dist-packages
(1.1.5)
Collecting pandas
  Downloading pandas-1.3.4-cp37-cp37m-manylinux_2_17_x86_64.manylinux2014_x86_64.w
hl (11.3 MB)
     |████████████████████████████████| 11.3 MB 33.8 MB/s
Requirement already satisfied: pytz>=2017.3 in /usr/local/lib/python3.7/dist-packa
ges (from pandas) (2018.9)
Requirement already satisfied: numpy>=1.17.3 in /usr/local/lib/python3.7/dist-pack
ages (from pandas) (1.19.5)
Requirement already satisfied: python-dateutil>=2.7.3 in /usr/local/lib/python3.7/
dist-packages (from pandas) (2.8.2)
Requirement already satisfied: six>=1.5 in /usr/local/lib/python3.7/dist-packages
(from python-dateutil>=2.7.3->pandas) (1.15.0)
Installing collected packages: pandas
  Attempting uninstall: pandas
    Found existing installation: pandas 1.1.5
    Uninstalling pandas-1.1.5:
      Successfully uninstalled pandas-1.1.5
ERROR: pip's dependency resolver does not currently take into account all the pack
ages that are installed. This behaviour is the source of the following dependency
 conflicts.
google-colab 1.0.0 requires pandas~=1.1.0; python_version >= "3.0", but you have p
andas 1.3.4 which is incompatible.
Successfully installed pandas-1.3.4
```

```
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  fonts-nanum
0 upgraded, 1 newly installed, 0 to remove and 37 not upgraded.
Need to get 9,604 kB of archives.
After this operation, 29.5 MB of additional disk space will be used.
Get:1 http://archive.ubuntu.com/ubuntu bionic/universe amd64 fonts-nanum all 20170
925-1 [9,604 kB]
Fetched 9,604 kB in 1s (9,257 kB/s)
debconf: unable to initialize frontend: Dialog
debconf: (No usable dialog-like program is installed, so the dialog based frontend
cannot be used. at /usr/share/perl5/Debconf/FrontEnd/Dialog.pm line 76, <> line
1.)
debconf: falling back to frontend: Readline
debconf: unable to initialize frontend: Readline
debconf: (This frontend requires a controlling tty.)
debconf: falling back to frontend: Teletype
dpkg-preconfigure: unable to re-open stdin:
Selecting previously unselected package fonts-nanum.
(Reading database ... 155219 files and directories currently installed.)
Preparing to unpack .../fonts-nanum_20170925-1_all.deb ...
Unpacking fonts-nanum (20170925-1) ...
Setting up fonts-nanum (20170925-1) ...
Processing triggers for fontconfig (2.12.6-0ubuntu2) ...
/usr/share/fonts: caching, new cache contents: 0 fonts, 1 dirs
/usr/share/fonts/truetype: caching, new cache contents: 0 fonts, 3 dirs
/usr/share/fonts/truetype/humor-sans: caching, new cache contents: 1 fonts, 0 dirs
/usr/share/fonts/truetype/liberation: caching, new cache contents: 16 fonts, 0 dir
s
/usr/share/fonts/truetype/nanum: caching, new cache contents: 10 fonts, 0 dirs
/usr/local/share/fonts: caching, new cache contents: 0 fonts, 0 dirs
/root/.local/share/fonts: skipping, no such directory
/root/.fonts: skipping, no such directory
/var/cache/fontconfig: cleaning cache directory
/root/.cache/fontconfig: not cleaning non-existent cache directory
/root/.fontconfig: not cleaning non-existent cache directory
fc-cache: succeeded
```

In [ ]:

```
# Step 4. 한글 글꼴 설정
import matplotlib.pyplot as plt

plt.rcParams['font.family']=['NanumGothic', 'sans-serif']
plt.rcParams['axes.unicode_minus'] = False

# 한글 설정 확인
%matplotlib inline

fig, ax = plt.subplots(figsize=(10,3))
ax.text(0.5, 0.5, "한글 테스트")
plt.show()
```



1. 개발 환경 설정

1.1. 기본 라이브러리

In [ ]:

```
import numpy as np
import pandas as pd
import seaborn as sns

import requests
from copy import deepcopy

sns.set_context("talk")
sns.set_style("whitegrid")

pd.options.display.max_columns=50

# seaborn 설정에 의해 파괴되는 한글 설정을 재설정
plt.rcParams['font.family']=['NanumGothic', 'sans-serif']
plt.rcParams['axes.unicode_minus'] = False
```

1.2. KOBIS API Keys

In [ ]:

```
# KOBIS OPEN API 서비스에서 키 발급 필요 : http://www.kobis.or.kr/kobisopenapi/homepg/main/main.do
APIKEY = '본인의_API_KEY'
```

2. 데이터 확보

2.1. API 활용 KOBIS 영화 목록 데이터 다운로드

- 데이터 다운로드는 1960-2021년.
- 다운받은 후 1971-2020만 추출해서 사용

In [ ]:

```
# 영화 데이터 보관 디렉토리
!mkdir data
```

In [ ]:

```
%%time

# 영화 목록 다운로드
url = 'http://kobis.or.kr/kobisopenapi/webservice/rest/movie/searchMovieList.json'

cols_movielist = ['movieCd', 'movieNm', 'movieNmEn', 'prdtYear', 'openDt', 'typeNm', 'prdtStatNm', 'nationAlt', 'genreAlt', 'repNationNm', 'repGenreNm', 'directors', 'companys', 'openYear']
df_movielist_raw = pd.DataFrame(columns=cols_movielist)

for p in range(1000):
    # 영화개봉년도 1961~2021 검색. 한 페이지에 100개 아이템, 총 900페이지 = 최대 10만개 데이터
    r = requests.get(url, params={'key':APIKEY, 'openStartDt':'1960', 'openEndDt':'2021', 'itemPerPage':'100', 'curPage':str(p+1)})
    df_ = pd.DataFrame(r.json()['movieListResult']['movieList'])
    df_movielist_raw = pd.concat([df_movielist_raw, df_], axis=0)
    df_movielist_raw.to_pickle("./data/df_movielist_raw.pkl")

# 제작년도 결측치 처리, 타입 변환
df_movielist_raw.loc[df_movielist_raw["prdtYear"] == "", "prdtYear"] = np.nan
df_movielist_raw.loc[~df_movielist_raw["prdtYear"].isna(), "prdtYear"] = df_movielist_raw.loc[~df_movielist_raw["prdtYear"].isna()]["prdtYear"].astype(int)

# 개봉년도 추출, 타입 변환
df_movielist_raw["openYear"]=df_movielist_raw["openDt"].apply(lambda x: x[:4])
df_movielist_raw["openYear"]=df_movielist_raw["openYear"].astype(int)

# 개봉일자 타입 변환
df_movielist_raw["openDt"]=df_movielist_raw["openDt"].astype(int)

# 개봉일자 기준 정렬
df_movielist_raw.sort_values("openDt")
df_movielist_raw.reset_index(drop=True, inplace=True)

# 파일 저장
df_movielist_raw.to_pickle("./data/df_movielist_raw.pkl")
```

```
CPU times: user 2min 28s, sys: 8.45 s, total: 2min 37s
Wall time: 16min 12s
```

In [ ]:

```
# API 활용 다운로드에 15분 가량 소요 (Colab 기준)
# 본인 google drive 등을 활용해 데이터 백업 권장.
df_movielist_raw = pd.read_pickle("./data/df_movielist_raw.pkl")
```

2.2.영화 목록 데이터 정제 df_movielist

2.2.1. 범위 제한, 결측치 제거

In [ ]:

```
df_movielist_raw = pd.read_pickle("./data/df_movielist_raw.pkl")

# 1971년 이후로 범위 제한
df_movielist = df_movielist_raw.query("1971 <= openYear <= 2020")
# 개봉영화로 범위 제한
df_movielist = df_movielist.query("prdtStatNm == '개봉'")
print(df_movielist.shape)
df_movielist.head()
```

(23419, 14)

Out[ ]:

	movieCd	movieNm	movieNmEn	prdtYear	openDt	typeNm	prdtStatNm	nationAlt	g
10	20139221	그래비티	Gravity	2013	20131017	장편	개봉	미국	
23	20201002	조제	Josée	2020	20201210	장편	개봉	한국	
27	20010264	아멜리에	Amelie Of Montmartre	2001	20011019	장편	개봉	프랑스	
52	20030039	링0 - 버스데이	Ring0 - Birthday	2000	20030411	장편	개봉	일본	
63	20050265	부에나 비스타 소셜 클럽	Buena Vista Social Club	1999	20010301	장편	개봉	독일	

In [ ]:

```
# 대표국가 없는 영화 제거
df_movielist = df_movielist.drop(df_movielist.query("repNationNm == ''").index)
df_movielist.shape
```

Out[ ]:

(23409, 14)

In [ ]:

```
# 대표국가별 영화 수 확인
df_movielist["repNationNm"].value_counts()
```

Out[ ]:

```
한국          7418
미국          6207
일본          4295
홍콩           924
프랑스          916
           ...
모로코            1
아프카니스탄         1
몰타             1
쿠바             1
룩셈부르크          1
Name: repNationNm, Length: 75, dtype: int64
```

In [ ]:

```
# 데이터 백업
df_movielist.to_pickle("./data/df_movielist.pkl")
```

2.2.1. 제작국가별 데이터셋 구축 df_nations

In [ ]:

```
# 별도 데이터셋 작성
df_nations = df_movielist[["movieCd", "movieNm", "openYear", "repNationNm"]]
df_nations.dropna(subset=["repNationNm"], inplace=True)

# 영화 제작국가 정리
nations = np.unique(','.join(df_nations.loc[df_nations["repNationNm"] != ""]['repNationNm']).spl
it(','))
print(f"{len(nations)} Nations: {nations}")
```

```
75 Nations: ['그리스' '기타' '남아프리카공화국' '네덜란드' '노르웨이' '뉴질랜드'
'대만' '덴마크' '독일' '러시아' '루마니아'
 '룩셈부르크' '마케도니아' '말레이시아' '멕시코' '모로코' '몰타' '몽고' '미국' '베
네수엘라' '베트남' '벨기에'
 '보스니아' '부탄' '북한' '불가리아' '브라질' '서독' '세르비아' '스웨덴' '스위스'
'스페인' '슬로바키아'
 '싱가포르' '아랍에미리트연합국정부' '아르헨티나' '아이슬란드' '아일랜드' '아프카
니스탄' '에스토니아' '영국'
 '오스트리아 ' '우루과이' '우즈베키스탄' '우크라이나' '유고슬라비아' '이라크' '이
란' '이스라엘' '이탈리아' '인도'
 '인도네시아' '일본' '잉글랜드' '중국' '체코' '칠레' '카자흐스탄' '캐나다' '콜롬비
아' '쿠바' '크로아티아' '태국'
 '터키' '팔레스타인' '페루' '포르투갈' '폴란드' '프랑스' '핀란드' '필리핀' '한국'
'헝가리' '호주' '홍콩']

/usr/local/lib/python3.7/dist-packages/pandas/util/_decorators.py:311: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  return func(*args, **kwargs)
```

In [ ]:

```
# 국가별 one-hot encoding
for nation in nations:
    df_nations[f"N_{nation}"] = df_nations["repNationNm"].str.contains(nation).astype('int')

df_nations.tail()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:3: SettingWithCopyWar
ning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  This is separate from the ipykernel package so we can avoid doing imports until
```

Out[ ]:

	movieCd	movieNm	openYear	repNationNm	N_그리스	N_기타	N_남아프리카공화국	N_네덜란드	N_노르웨이	N_뉴질랜드	N_대만	N_덴마크	N_독일
25540	19860079	LA용팔이	1986	한국	0	0	0	0	0	0	0	0	0
25541	19910228	LA이야기	1991	미국	0	0	0	0	0	0	0	0	0
25542	19870065	Y의 체험	1987	한국	0	0	0	0	0	0	0	0	0
25696	19880146	원+씩스	1988	이탈리아	0	0	0	0	0	0	0	0	0
26169	19910096	칙칙이의 내일은 참 피온	1991	한국	0	0	0	0	0	0	0	0	0

5 rows × 79 columns

In [ ]:

```
# 해외영화를 따로 분류
df_nations["해외"] = 0
df_nations.loc[df_nations["repNationNm"] != "한국", "해외"] = 1
df_nations.drop("repNationNm", axis=1, inplace=True)

# 한국영화와 해외영화 수 확인
print("# 한국 : ", df_nations["N_한국"].sum())
print("# 해외: ", df_nations["해외"].sum())

# 개봉작 수 상위 20개국 추출
df_nations_top20 = df_nations.sum().drop(["openYear", "해외", "movieCd", "movieNm"]).sort_values
(ascending=False)[:20]
df_nations_top20
```

```
# 한국 :  7418
# 해외:  15991
```

Out[ ]:

```
N_한국       7418
N_미국       6207
N_일본       4295
N_홍콩        924
N_프랑스       916
N_영국        746
N_중국        429
N_이탈리아      358
N_독일        358
N_캐나다        249
N_스페인        239
N_러시아        152
N_호주        111
N_대만        101
N_덴마크         70
N_인도          59
N_스웨덴         58
N_태국          58
N_네덜란드        46
N_멕시코         44
dtype: object
```

In [ ]:

```
# 년도별 개봉 편 수 계산
df_nationsY = df_nations.groupby("openYear").sum()
df_nationsY.reset_index(inplace=True)
df_nationsY.head()
```

Out[ ]:

	openYear	N_그리스	N_기타	N_남아프리카공화국	N_네덜란드	N_노르웨이	N_뉴질랜드	N_대만	N_덴마크	N_독일	N_러시아	N_루마니아	N_룩셈부르크	N_마케도니아	N_말레이시아	N_멕시코	N_모로코	N_몰타	N_몽고
0	1971	0	0	0	0	0	0	6	0	0	0	0	0	0	0	2	0	0	0
1	1972	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0
2	1973	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0
3	1974	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
4	1975	1	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

5 rows × 77 columns

In [ ]:

```
# 데이터 백업
df_nations.to_pickle("./data/df_nations.pkl")
df_nationsY.to_pickle("./data/df_nationsY.pkl")
```

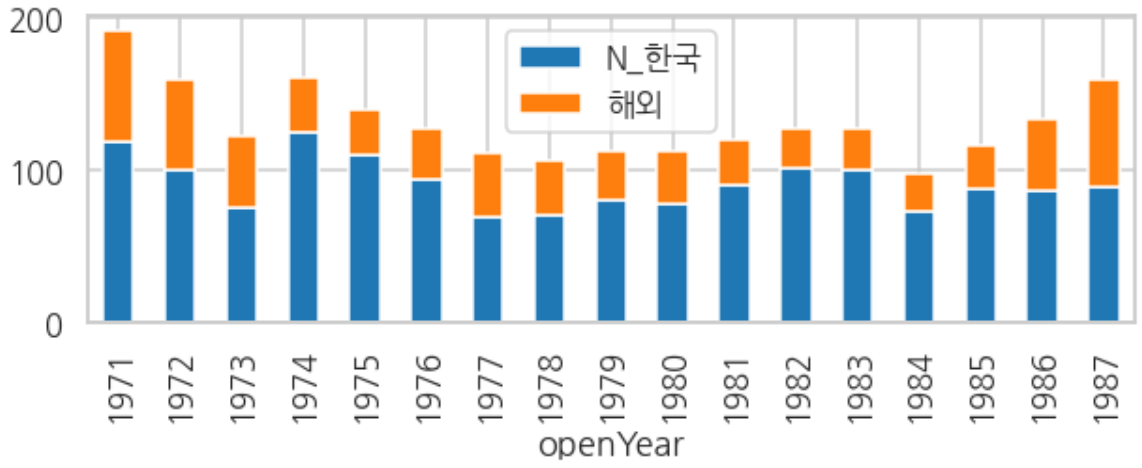
기초분석

In [ ]:

```
# 1987년까지 영화 개봉 현황
fig, ax = plt.subplots(figsize=(10, 3))
df_nationsY.query("openYear < 1988")[["openYear", "N_한국", "해외"]].plot.bar(x="openYear", stacked=True, ax=ax)
```

Out[ ]:

<AxesSubplot:xlabel='openYear'>



In [ ]:

```
# 1988-2010 개봉 한국영화
df_nations.query("1988 <= openYear <=2010")["N_한국"].sum()
```

Out[ ]:

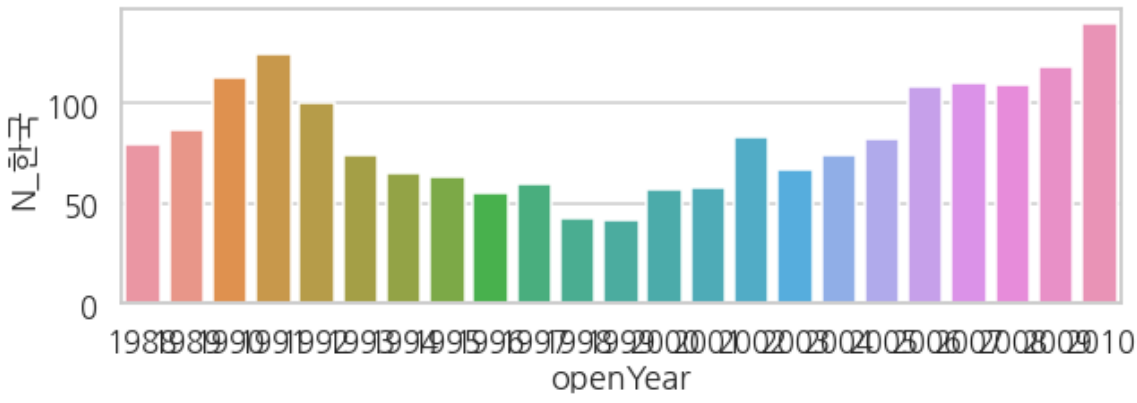
1913

In [ ]:

```
fig, ax = plt.subplots(figsize=(10, 3))
sns.barplot(x="openYear", y="N_한국", data=df_nationsY.query("1988 <= openYear <=2010"), ax=ax)
```

Out[ ]:

<AxesSubplot:xlabel='openYear', ylabel='N_한국'>



In [ ]:

```
# 1988-2010 개봉 한국영화 연평균
df_nations.query("1988 <= openYear <=2010")["N_한국"].sum()/23
```

Out[ ]:

83.17391304347827

In [ ]:

```
# 1988-2010 개봉 해외영화
df_nations.query("1988 <= openYear <=2010")["해외"].sum()
```

Out[ ]:

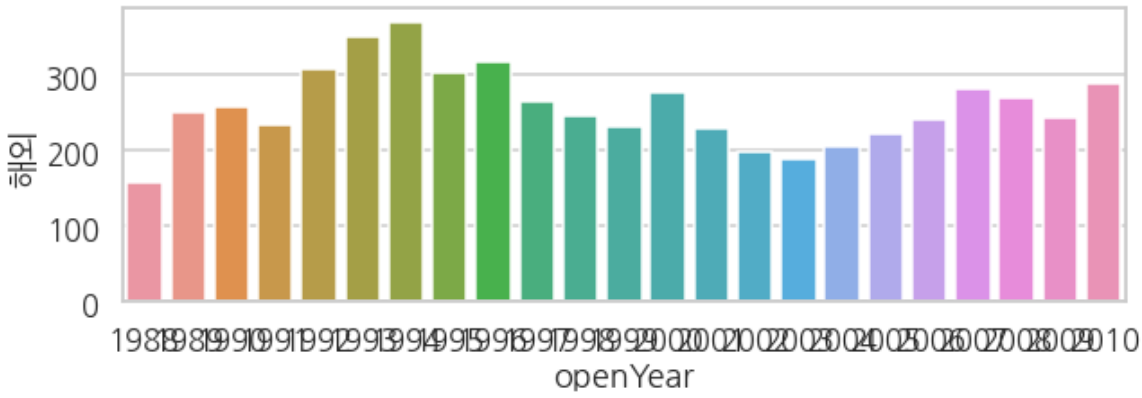
5932

In [ ]:

```
fig, ax = plt.subplots(figsize=(10, 3))
sns.barplot(x="openYear", y="해외", data=df_nationsY.query("1988 <= openYear <=2010"), ax=ax)
```

Out[ ]:

<AxesSubplot:xlabel='openYear', ylabel='해외'>



In [ ]:

```
# 1988-2010 개봉 홍콩영화
df_nations.query("1988 <= openYear <=2010")["N_홍콩"].sum()
```

Out[ ]:

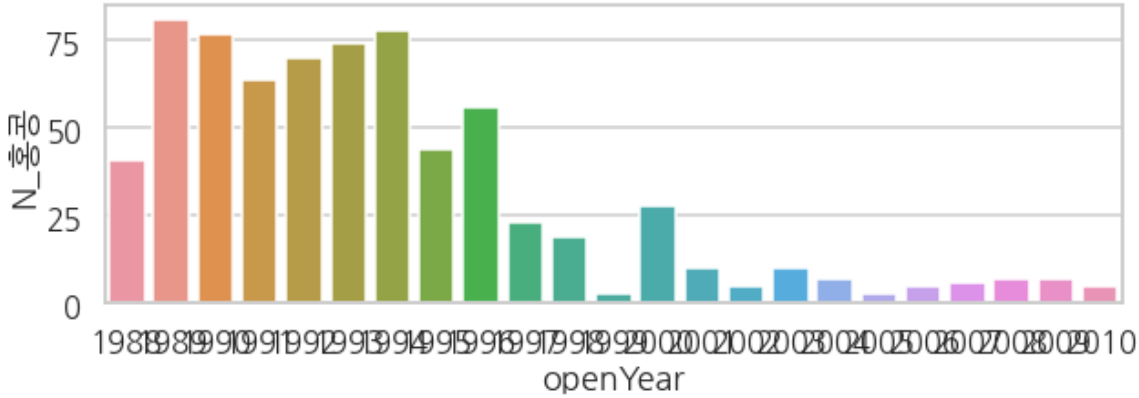
723

In [ ]:

```
fig, ax = plt.subplots(figsize=(10, 3))
sns.barplot(x="openYear", y="N_홍콩", data=df_nationsY.query("1988 <= openYear <=2010"), ax=ax)
```

Out[ ]:

<AxesSubplot:xlabel='openYear', ylabel='N_홍콩'>



In [ ]:

```
# 1988-2010 개봉 해외영화 연평균
df_nations.query("1988 <= openYear <=2010")["해외"].sum()/23
```

Out[ ]:

257.9130434782609

In [ ]:

```
# 1998년 한국영화 개봉작 수
df_nationsY.query("openYear == 1998")["N_한국"]
```

Out[ ]:

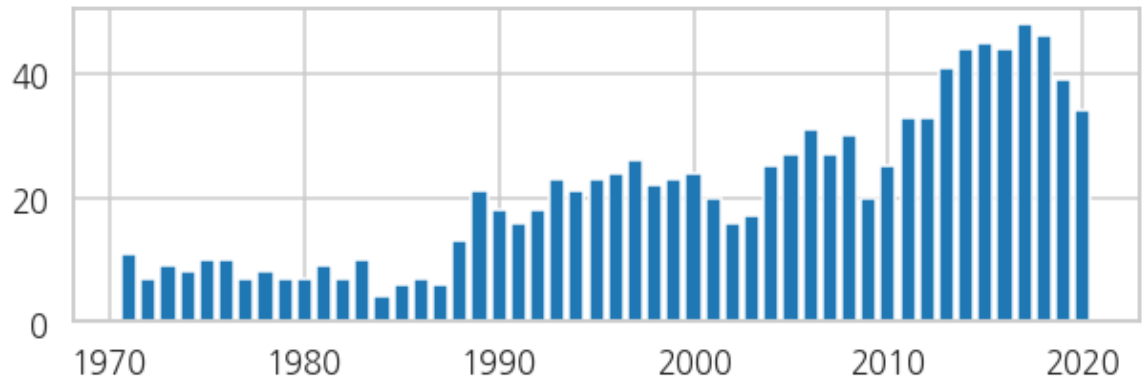
```
27    43
Name: N_한국, dtype: int64
```

In [ ]:

```
# 개봉 영화 제작 국가
fig, ax = plt.subplots(figsize=(10, 3))
ax.bar(df_nationsY["openYear"], (df_nationsY.filter(like="N_") > 0).sum(axis=1))
```

Out[ ]:

<BarContainer object of 50 artists>



In [ ]:

```
# 전체 범위에서 가장 다양한 나라의 영화가 개봉한 해
df_nationsY["openYear"][(df_nationsY.filter(like="N_") > 0).sum(axis=1).argmax()]
```

Out[ ]:

2017

In [ ]:

```
# 그 해에 개봉된 영화의 제작 국가 수
(df_nationsY.filter(like="N_") > 0).sum(axis=1).max()
```

Out[ ]:

48

In [ ]:

```
# 1987년 이전까지 가장 다양한 나라의 영화가 개봉됐을 때 제작 국가 수
(df_nationsY.query("openYear < 1989").filter(like="N_") > 0).sum(axis=1).max()
```

Out[ ]:

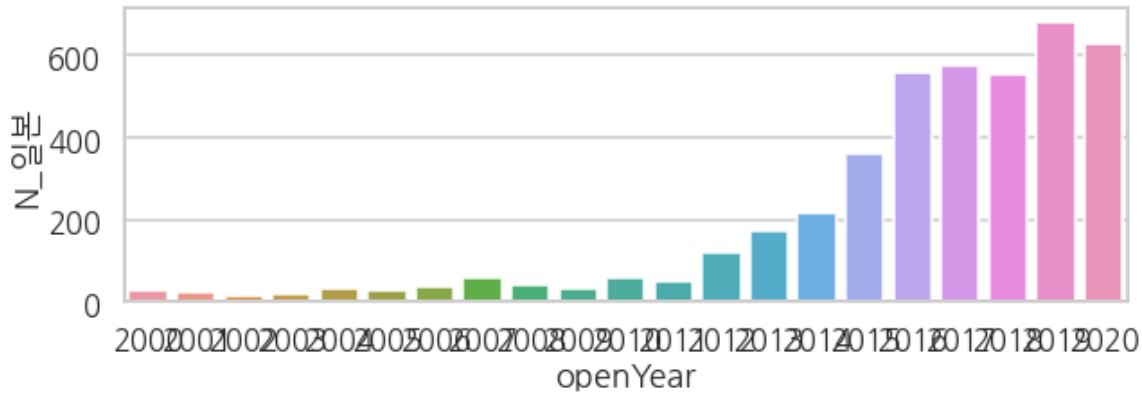
13

In [ ]:

```
# 일본영화 상영 편 수
fig, ax = plt.subplots(figsize=(10, 3))
sns.barplot(x="openYear", y="N_일본", data=df_nationsY.query("2000 <= openYear"), ax=ax)
```

Out[ ]:

<AxesSubplot:xlabel='openYear', ylabel='N_일본'>

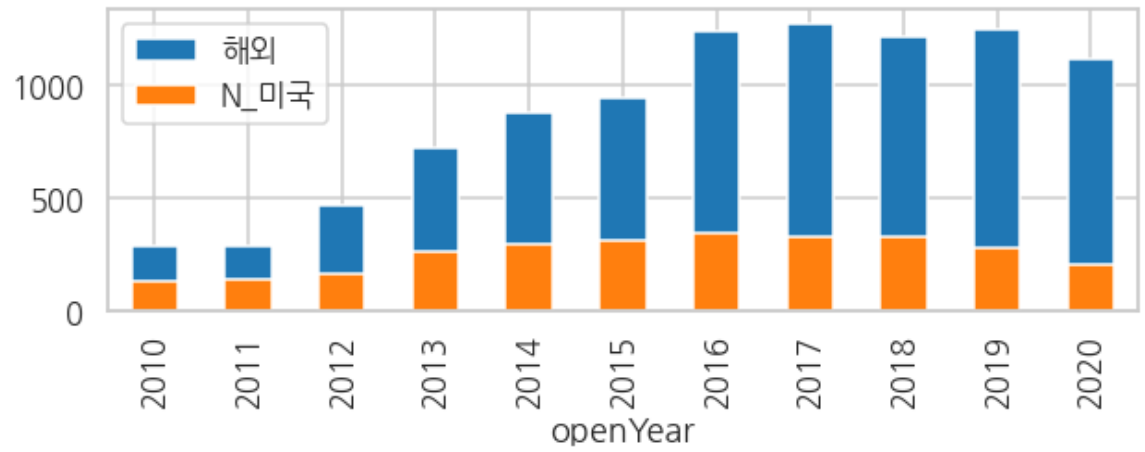


In [ ]:

```
# OTT 서비스 이후 해외와 미국 영화 상영 편 수
fig, ax = plt.subplots(figsize=(10, 3))
df_nationsY.query("2010 <= openYear ")[["openYear", "해외"]].plot.bar(x="openYear", ax=ax)
df_nationsY.query("2010 <= openYear ")[["openYear", "N_미국"]].plot.bar(x="openYear", ax=ax, color="C1")
```

Out[ ]:

<AxesSubplot:xlabel='openYear'>

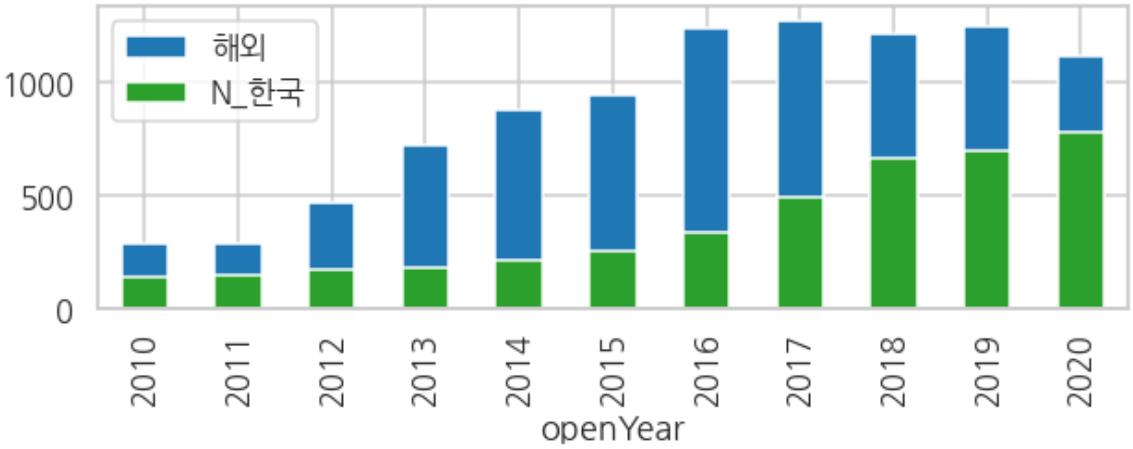


In [ ]:

```
# OTT 서비스 이후 해외와 한국 영화 상영 편 수
fig, ax = plt.subplots(figsize=(10, 3))
df_nationsY.query("2010 <= openYear ")[["openYear", "해외"]].plot.bar(x="openYear", ax=ax, color="C0")
df_nationsY.query("2010 <= openYear ")[["openYear", "N_한국"]].plot.bar(x="openYear", ax=ax, color="C2")
```

Out[ ]:

<AxesSubplot:xlabel='openYear'>



In [ ]:

```
# 설국열차, 미나리는 한국영화? : 미나리는 미국 영화. 2021년이라 범위 밖.
df_movielist_raw.query("movieNm == '미나리'")
```

Out[ ]:

	movieCd	movieNm	movieNmEn	prdtYear	openDt	typeNm	prdtStatNm	nationAlt
294	20205144	미나리	Minari	2020	20210303	장편	개봉	미국

In [ ]:

```
# 설국열차, 미나리는 한국영화? : 설국열차는 한국영화
df_nations.query("movieNm == '설국열차'")
```

Out[ ]:

	movieCd	movieNm	openYear	N_그리스	N_기타	N_남아프리카공화국	N_네덜란드	N_노르웨이	N_뉴질랜드	N_대만	N_덴마크	N_독일	N_러시아	N_루마니아	N_룩셈부르크
5716	20126674	설국열차	2013	0	0	0	0	0	0	0	0	0	0	0	0

1 rows × 79 columns

2.2.2. 장르별 데이터셋 구축 df_genres

In [ ]:

```
# 장르 데이터셋 구축
df_genres = df_movielist[["movieCd", "movieNm", "openYear", "genreAlt"]]
df_genres.dropna(subset=["genreAlt"], inplace=True)

# 영화 장르 파악
genres = np.unique(','.join(df_genres.loc[df_genres["genreAlt"] != ""]['genreAlt']).split(','))
genres = [g.split("(")[0] for g in genres]
print(f"{len(genres)} Genres: {genres}")
```

```
21 Genres: ['SF', '가족', '공연', '공포', '기타', '다큐멘터리', '드라마', '멜로/로
맨스', '뮤지컬', '미스터리', '범죄', '사극', '서부극', '성인물', '스릴러', '애니메
이션', '액션', '어드벤처', '전쟁', '코미디', '판타지']

/usr/local/lib/python3.7/dist-packages/pandas/util/_decorators.py:311: SettingWith
CopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stabl
e/user_guide/indexing.html#returning-a-view-versus-a-copy
  return func(*args, **kwargs)
```

In [ ]:

```
# one-hot encoding

for genre in genres:
    df_genres[f"G_{genre}"] = df_genres["genreAlt"].str.contains(genre.split("(")[0]).astype('int')
df_genres.drop("genreAlt", axis=1, inplace=True)
df_genres.tail()
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4: SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  after removing the cwd from sys.path.
```

Out[ ]:

	movieCd	movieNm	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬	G_미스터리	G_범죄
25540	19860079	LA용팔이	1986	0	1	0	0	0	0	1	0	0	0	0
25541	19910228	LA이야기	1991	0	0	0	0	0	0	0	0	0	0	0
25542	19870065	Y의 체험	1987	0	0	0	0	0	0	1	0	0	0	0
25696	19880146	원+씩스	1988	0	0	0	0	0	0	0	0	0	0	0
26169	19910096	칙칙이의 내일은 참 피온	1991	0	0	0	0	0	0	0	0	0	0	0

In [ ]:

```
# 기타 장르 영화 수 확인
df_genres["G_기타"].sum()
```

Out[ ]:

124

In [ ]:

```
# 아무 장르도 없는 영화를 기타로 처리
df_genres.loc[df_genres.filter(like="G_").sum(axis=1) == 0, "G_기타"] = 1
df_genres["G_기타"].sum()
```

Out[ ]:

305

In [ ]:

```
# 장르가 2개 이상인 영화를 복합장르로 지정
# 복합장르
df_genres["복합장르"] = 0
df_genres.loc[df_genres.filter(like="G_").sum(axis=1) > 1, "복합장르"] = 1
```

In [ ]:

```
df_genres.iloc[df_genres.filter(like="G_").sum(axis=1).argmax()]
```

Out[ ]:

```
movieCd      20090501
movieNm          9:나인
openYear         2009
G_SF                1
G_가족              0
G_공연              0
G_공포              0
G_기타              0
G_다큐멘터리        0
G_드라마            0
G_멜로/로맨스       0
G_뮤지컬            0
G_미스터리          1
G_범죄              0
G_사극              0
G_서부극            0
G_성인물            0
G_스릴러            1
G_애니메이션        1
G_액션              1
G_어드벤처          1
G_전쟁              0
G_코미디            0
G_판타지            1
복합장르            1
Name: 18893, dtype: object
```

In [ ]:

```
# 장르 순위 데이터셋
df_genres_top21 = df_genres.filter(like="G_").sum().sort_values(ascending=False)[:21]
df_genres_top21
```

Out[ ]:

```
G_드라마          9690
G_멜로/로맨스      5860
G_액션           4451
G_코미디          3063
G_스릴러          2707
G_성인물          2091
G_공포           1465
G_범죄           1371
G_애니메이션        1164
G_어드벤처         1126
G_SF           1040
G_판타지           882
G_미스터리          699
G_다큐멘터리         659
G_가족            630
G_사극            490
G_전쟁            464
G_기타            305
G_공연            243
G_뮤지컬           161
G_서부극            91
dtype: int64
```

In [ ]:

```
# 년도별 장르 데이터셋 구축
df_genresY = df_genres[["movieCd", "openYear"]].groupby("openYear").count().reset_index().merge(
df_genres.iloc[:,2:].groupby("openYear").sum().reset_index())
df_genresY.drop("movieCd", axis=1, inplace=True)
df_genresY.fillna(0, inplace=True)
df_genresY.head()
```

Out[ ]:

	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬	G_미스터리	G_범죄	G_사극	G_서부극	G_성인물	G_스릴러	G_애니메이션	G_
0	1971	1	3	0	4	25	3	54	51	6	2	4	12	17	0	3	0	4
1	1972	0	4	0	3	20	3	54	43	4	4	10	8	8	0	8	0	2
2	1973	0	3	0	3	8	6	38	30	4	3	12	12	1	0	6	2	2
3	1974	0	2	0	9	10	0	69	48	3	3	13	8	1	0	9	0	4
4	1975	1	3	0	7	11	2	46	42	3	2	3	8	1	0	6	0	3

In [ ]:

```
# 데이터 백업
df_genres.to_pickle("./data/df_genres.pkl")
df_genresY.to_pickle("./data/df_genresY.pkl")
```

2.2.2.1. 한국영화 장르 데이터

In [ ]:

```
# 영화 데이터
df_genres_kr = df_genres.loc[df_nations["N_한국"]==1]
df_genres_kr.head()
```

Out[ ]:

	movieCd	movieNm	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬	G_미스터리	G_범죄	G
23	20201002	조제	2020	0	0	0	0	0	0	1	1	0	0	0	
75	20050355	음란서생	2006	0	0	0	0	0	0	0	0	0	0	0	
105	19860031	변강쇠	1986	0	0	0	0	0	0	1	1	0	0	0	
106	19820012	애마부인	1982	0	0	0	0	0	0	1	1	0	0	0	
152	20101233	평범한 날들	2011	0	0	0	0	0	0	1	0	0	0	0	

In [ ]:

```
# 년간 데이터
df_genresY_kr = df_genres_kr[["movieCd", "openYear"]].groupby("openYear").count().reset_index().
merge(df_genres_kr.iloc[:,2:].groupby("openYear").sum().reset_index())
df_genresY_kr.drop("movieCd", axis=1, inplace=True)
df_genresY_kr.fillna(0, inplace=True)
df_genresY_kr.head()
```

Out[ ]:

	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬	G_미스터리	G_범죄	G_사극	G_서부극	G_성인물	G_스릴러	G_애니메이션	G_액션
0	1971	0	0	0	3	25	0	23	45	0	1	0	8	0	0	1	0	2
1	1972	0	0	0	2	18	0	23	34	0	2	0	6	0	0	0	0	1
2	1973	0	0	0	2	8	0	14	26	1	2	2	6	0	0	1	0	1
3	1974	0	0	0	7	9	0	46	45	1	1	0	4	0	0	1	0	2
4	1975	0	0	0	4	11	0	26	40	1	2	0	6	0	0	0	0	2

In [ ]:

```
# 데이터 백업
df_genres_kr.to_pickle("./data/df_genres_kr.pkl")
df_genresY_kr.to_pickle("./data/df_genresY_kr.pkl")
```

2.2.2.2. 해외영화 장르 데이터

In [ ]:

```
# 영화 데이터
df_genres_nkr = df_genres.loc[df_nations["해외"]==1]
df_genres_nkr.head()
```

Out[ ]:

	movieCd	movieNm	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬	G_미스터리	G_범죄	G_사극
10	20139221	그래비티	2013	1	0	0	0	0	0	1	0	0	0	0	0
27	20010264	아멜리에	2001	0	0	0	0	0	0	0	0	0	0	0	0
52	20030039	링0 - 버스데이	2003	0	0	0	1	0	0	0	0	0	0	0	0
63	20050265	부에나 비스타 소셜 클럽	2001	0	0	0	0	0	1	0	0	0	0	0	0
73	19990317	링	1999	0	0	0	1	0	0	0	0	0	1	0	0

In [ ]:

```
# 년간 데이터
df_genresY_nkr = df_genres_nkr[["movieCd", "openYear"]].groupby("openYear").count().reset_index
().merge(df_genres_nkr.iloc[:,2:].groupby("openYear").sum().reset_index())
df_genresY_nkr.drop("movieCd", axis=1, inplace=True)
df_genresY_nkr.fillna(0, inplace=True)
df_genresY_nkr.head()
```

Out[ ]:

	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬	G_미스터리	G_범죄	G_사극	G_서부극	G_성인물	G_스릴러	G_애니메이션	G_
0	1971	1	3	0	1	0	3	31	6	6	1	4	4	17	0	2	0	2
1	1972	0	4	0	1	2	3	31	9	4	2	10	2	8	0	8	0	1
2	1973	0	3	0	1	0	6	24	4	3	1	10	6	1	0	5	2	1
3	1974	0	2	0	2	1	0	23	3	2	2	13	4	1	0	8	0	1
4	1975	1	3	0	3	0	2	20	2	2	0	3	2	1	0	6	0	

In [ ]:

```
# 데이터 백업
df_genres_nkr.to_pickle("./data/df_genres_nkr.pkl")
df_genresY_nkr.to_pickle("./data/df_genresY_nkr.pkl")
```

기초분석

In [ ]:

```
# 한국 영화 편당 평균 장르 수
df_genres.filter(like="G_").sum(axis=1).loc[df_nations["N_한국"]==1].sum()/df_nations["N_한국"].
sum()
```

Out[ ]:

1.461714747910488

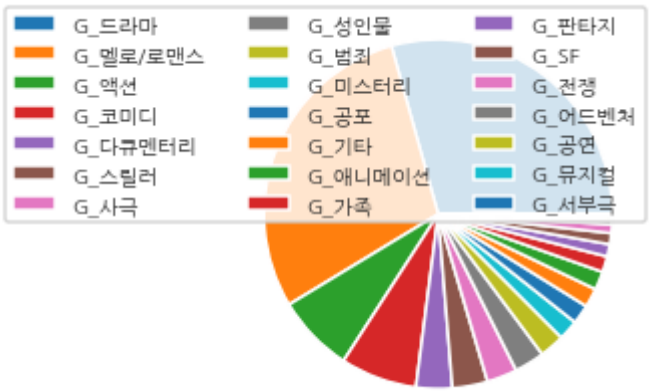
In [ ]:

```
# 한국 영화 장르별 비중
genres_portions_kr = df_genres.loc[df_nations["N_한국"]==1].filter(like="G_").sum().sort_values(
ascending=False)/df_genres.loc[df_nations["N_한국"]==1].filter(like="G_").sum().sum()
patches, texts = plt.pie(genres_portions_kr)
plt.legend(patches, labels=genres_portions_kr.index, ncol=3, fontsize="xx-small")
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:3: MatplotlibDeprecat
ionWarning: normalize=None does not normalize if the sum is less than 1 but this b
ehavior is deprecated since 3.3 until two minor releases later. After the deprecat
ion period the default value will be normalize=True. To prevent normalization pass
normalize=False
  This is separate from the ipykernel package so we can avoid doing imports until
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4: UserWarning: You h
ave mixed positional and keyword arguments, some input may be discarded.
  after removing the cwd from sys.path.
```

Out[ ]:

```
<matplotlib.legend.Legend at 0x7f4abd3069d0>
```



In [ ]:

```
# 해외 영화 편당 평균 장르
df_genres.filter(like="G_").sum(axis=1).loc[df_nations["해외"]==1].sum()/df_nations["해외"].sum
()
```

Out[ ]:

```
1.7390407103995997
```

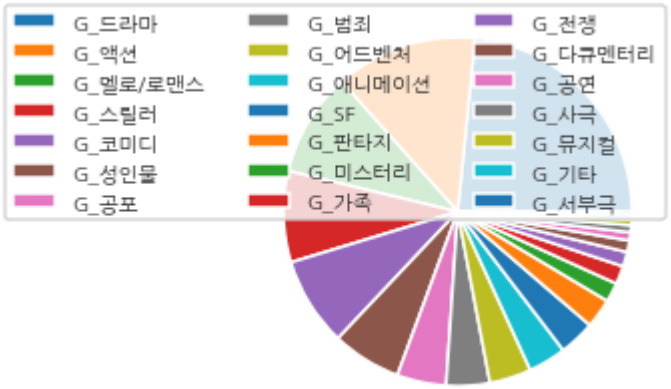
In [ ]:

```
# 해외 영화 장르별 비중
genres_portions_kr = df_genres.loc[df_nations["해외"]==1].filter(like="G_").sum().sort_values(as
cending=False)/df_genres.loc[df_nations["해외"]==1].filter(like="G_").sum().sum()
patches, texts = plt.pie(genres_portions_kr)
plt.legend(patches, labels=genres_portions_kr.index, ncol=3, fontsize="xx-small")
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:3: MatplotlibDeprecat
ionWarning: normalize=None does not normalize if the sum is less than 1 but this b
ehavior is deprecated since 3.3 until two minor releases later. After the deprecat
ion period the default value will be normalize=True. To prevent normalization pass
normalize=False
  This is separate from the ipykernel package so we can avoid doing imports until
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:4: UserWarning: You h
ave mixed positional and keyword arguments, some input may be discarded.
  after removing the cwd from sys.path.
```

Out[ ]:

```
<matplotlib.legend.Legend at 0x7f4aa51ea050>
```



In [ ]:

```
# 해외 영화 장르별 비중
df_genres.loc[df_nations["해외"]==1].filter(like="G_").sum().sort_values()/df_genres.loc[df_nations["해외"]==1].filter(like="G_").sum().sum()
```

Out[ ]:

```
G_서부극          0.003200
G_기타           0.004099
G_뮤지컬          0.005034
G_사극           0.006653
G_공연           0.007552
G_다큐멘터리        0.010644
G_전쟁           0.013593
G_가족           0.016937
G_미스터리         0.017836
G_판타지          0.027293
G_SF          0.033406
G_애니메이션        0.035204
G_어드벤처         0.038836
G_범죄           0.040275
G_공포           0.045705
G_성인물          0.064260
G_코미디          0.082455
G_스릴러          0.084577
G_멜로/로맨스      0.096372
G_액션           0.131972
G_드라마          0.234097
dtype: float64
```

In [ ]:

```
# 성인영화 장르가 드라마로 되어 있진 않은지: '애마부인 2016'은 성인물로 분류되지 않음.
df_genres.loc[df_genres["movieNm"].str.contains("애마")]
```

Out[ ]:

	movieCd	movieNm	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬	G_미스터리	G_범죄
106	19820012	애마부인	1982	0	0	0	0	0	0	1	1	0	0	0
11541	20163323	애마부인 2016	2016	0	0	0	0	0	0	0	1	0	0	0
17545	19910099	겨울애마, 봄	1991	0	0	0	0	0	0	0	0	0	0	0
20448	19950019	애마와 백수건달	1995	0	0	0	0	0	0	0	0	0	0	0
20875	19940002	드라큐라 애마	1994	0	0	0	0	0	0	1	0	0	0	0
21095	19950030	애마와 변강쇠	1995	0	0	0	0	0	0	0	0	0	0	0
21537	19950039	애마부인 11	1995	0	0	0	0	0	0	0	0	0	0	0
22060	19930046	애마부인 9	1993	0	0	0	0	0	0	0	1	0	0	0
22444	19920046	애마부인 7	1992	0	0	0	0	0	0	1	1	0	0	0
23026	19930024	애마부인 8	1993	0	0	0	0	0	0	1	1	0	0	0
23330	19910120	애마부인 6	1992	0	0	0	0	0	0	0	0	0	0	0
23412	19910001	애마부인 5	1991	0	0	0	0	0	0	0	0	0	0	0
24127	19850061	애마부인 3	1985	0	0	0	0	0	0	1	1	0	0	0
24984	19940042	애마부인 10	1994	0	0	0	0	0	0	0	1	0	0	0
24985	19840012	애마부인 2	1984	0	0	0	0	0	0	1	0	0	0	0
24986	19900093	애마부인 4	1990	0	0	0	0	0	0	1	1	0	0	0
25254	19900081	짚시애마	1990	0	0	0	0	0	0	0	1	0	0	0
25373	19880037	파리애마	1988	0	0	0	0	0	0	0	1	0	0	0

In [ ]:

```
# 아동용 애니메이션은 포함되어 있는지
df_genres.loc[df_genres["movieNm"].str.contains("영구")]
```

Out[ ]:

	movieCd	movieNm	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬	G_미스터리	G_범죄
18640	19930005	영구와 공룡 쮸쮸	1993	0	0	0	0	0	0	1	0	0	0	0
20045	19908586	영구 람보	1990	0	0	0	0	1	0	0	0	0	0	0
22446	19920044	영구와 흡혈귀 드라큐라	1992	0	0	0	1	0	0	1	0	0	0	0
23332	19910119	영구와 황금박쥐	1992	1	0	0	0	0	0	0	0	0	0	0
23434	19910018	영구와 땡칠이 4탄-홍콩할매귀신	1991	0	0	0	0	0	0	0	0	0	0	0
25059	19890038	영구와 땡칠이	1989	0	0	0	0	0	0	1	0	0	0	0
25060	19890083	영구와 땡칠이 소림사가다	1989	0	1	0	0	0	0	1	0	0	0	0
25061	19900090	영구와 땡칠이 3탄-영구람보	1990	0	0	0	0	0	0	0	0	0	0	0

In [ ]:

```
# 아동용 애니메이션은 포함되어 있는지
df_genres.loc[df_genres["movieNm"].str.contains("우뢰매")]
```

Out[ ]:

	movieCd	movieNm	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬	G_미스터리	G_범죄
13114	19860068	우뢰매 : 외계에서 온 우뢰매	1986	0	0	0	0	0	0	0	0	0	0	0
19694	19860087	외계에서 온 우뢰매 2	1986	1	1	0	0	0	0	0	0	0	0	0
19695	19870044	외계에서 온 우뢰매 전격 쓰리 작전	1987	1	1	0	0	0	0	0	0	0	0	0
19696	19870085	우뢰매 4 탄 썬더브이출동	1987	1	1	0	0	0	0	0	0	0	0	0
22411	19920002	우뢰매 7 : 돌아온 우뢰매	1992	1	0	0	0	0	0	1	0	0	0	0

3. 데이터 시각화

In [ ]:

```
# 이미지 저장 폴더
!mkdir images
```

3.1. 국가, 장르 overview

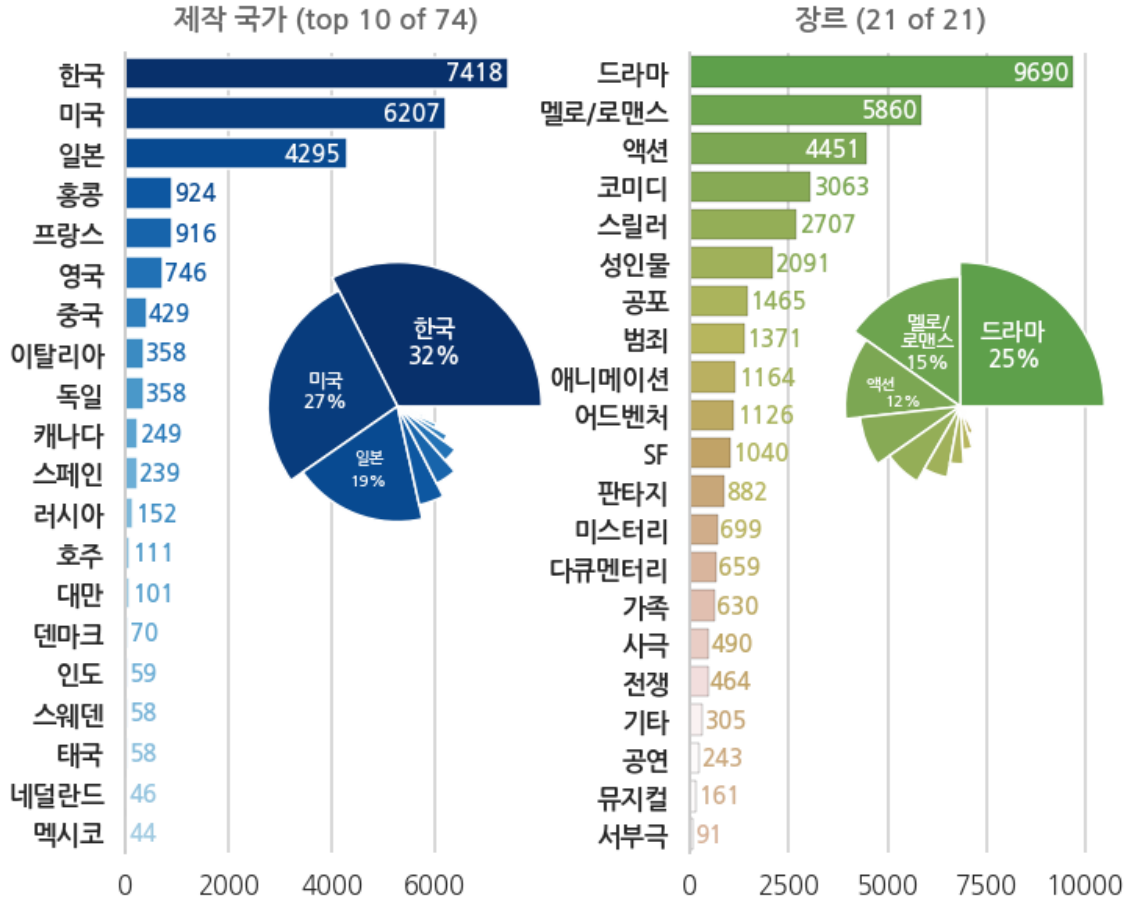
In [ ]:

```
# 국가별 색상코드
c_kr = "darkblue"       # 한국
c_etc = "0.7"           # 기타, 해외
c_it = "g"              # 이탈리아
c_cn = "darkred"        # 중국
c_fr = "gold"           # 프랑스
c_hk = "orangered"      # 홍콩
c_gb = "orchid"         # 영국
c_jp = "thistle"        # 일본
c_us = "mediumpurple"   # 미국
```

In [ ]:

```
fig, axs = plt.subplots(ncols=2, figsize=(10, 8), constrained_layout=True)

# 국가
blues_r = plt.get_cmap("Blues_r")
sns.barplot(x=df_nations_top20.values, y=df_nations_top20.index, ax=axs[0])
for i, p in enumerate(axs[0].patches):
    c = blues_r(i/20)
    p.set_facecolor(c)

for i, y in enumerate(df_nations_top20.index):
    val = df_nationsY[y].sum()
    if val > 2000:
        axs[0].text(val-100, i, str(val), c="w", fontsize="small", fontweight="bold", ha="right"
, va="center")
    else:
        axs[0].text(val+30, i, str(val), c=blues_r(i/30), fontsize="small", fontweight="bold", h
a="left", va="center")

axs[0].set_yticklabels([n.split("_")[1] for n in df_nations_top20.index], fontweight="bold")
font_title = {"fontweight":"bold", "color":"0.4"}
axs[0].set_title("제작 국가 (top 10 of 74)", fontdict=font_title, pad=16)

# pie plot
ax_pie0 = fig.add_axes([0.15, 0.35, 0.4, 0.4])
ax_pie0.pie(df_nations_top20.values)
for i, (p, v) in enumerate(zip(ax_pie0.patches, df_nations_top20.values)):
    # 안수빈, 황준원, 김영도님 의견 반영하여 수정. 감사합니다 :)
    r = 1-0.1*i if i < 10 else 0
    p.set_radius(r)
    p.set_facecolor(blues_r(i/20))

ax_pie0.text(0.25, 0.3, f"한국₩n{df_nations_top20['N_한국']/df_nations_top20.sum()*100:.0f}%",
             fontsize="small", color="w", fontweight="bold", ha="center")
ax_pie0.text(-0.5, 0, f"미국₩n{df_nations_top20['N_미국']/df_nations_top20.sum()*100:.0f}%",
             fontsize="x-small", color="w", fontweight="bold", ha="center")
ax_pie0.text(-0.2, -0.38, f"일본",
             fontsize="xx-small", color="w", fontweight="bold", ha="center")
ax_pie0.text(-0.2, -0.55, f"{df_nations_top20['N_일본']/df_nations_top20.sum()*100:.0f}%",
             fontsize="xx-small", color="w", fontweight="bold", ha="center")

# 장르
gist_earth = plt.get_cmap("gist_earth")
sns.barplot(x=df_genres_top21.values, y=df_genres_top21.index, ax=axs[1])
for i, p in enumerate(axs[1].patches):
    c = gist_earth(i/35+0.5)
    p.set_facecolor(c)
    p.set_edgecolor("k")
    p.set_linewidth(0.2)

for i, y in enumerate(df_genres_top21.index):
    val = df_genres_top21[y].sum()
    if val > 4000:
        axs[1].text(val-100, i, str(val), c="w", fontsize="small", fontweight="bold", ha="right"
, va="center")
    else:
        axs[1].text(val+50, i, str(val), c=gist_earth(i/55+0.5), fontsize="small", fontweight="b
old", ha="left", va="center")

axs[1].set_yticklabels([n.split("_")[1] for n in df_genres_top21.index], fontweight="bold")
```

```
axs[1].set_title("장르 (21 of 21)", fontdict=font_title, pad=16)

# spines
for ax in axs:
    ax.spines[["top", "right", "bottom"]].set_visible(False)

# pie plot
ax_pie1 = fig.add_axes([0.65, 0.35, 0.4, 0.4])
ax_pie1.pie(df_genres_top21.values)
for i, (p, v) in enumerate(zip(ax_pie1.patches, df_genres_top21.values)):
    r = 1 - 0.1*i if i < 10 else 0
    p.set_radius(r)
    p.set_facecolor(gist_earth(i/35+0.5))

ax_pie1.text(0.37, 0.28, f"드라마₩n{df_genres_top21['G_드라마']/df_genres_top21.sum()*100:.0f}%"
,
             fontsize="small", color="w", fontweight="bold", ha="center")
ax_pie1.text(-0.23, 0.27, f"멜로/₩n로맨스₩n{df_genres_top21['G_멜로/로맨스']/df_genres_top21.su
m()*100:.0f}%",
             fontsize="x-small", color="w", fontweight="bold", ha="center")
ax_pie1.text(-0.55, 0.13, f"액션",
             fontsize="xx-small", color="w", fontweight="bold", ha="center")
ax_pie1.text(-0.4, 0, f"{df_genres_top21['G_액션']/df_genres_top21.sum()*100:.0f}%",
             fontsize="xx-small", color="w", fontweight="bold", ha="center")


fig.savefig("./images/overview.png", dpi=200)
```

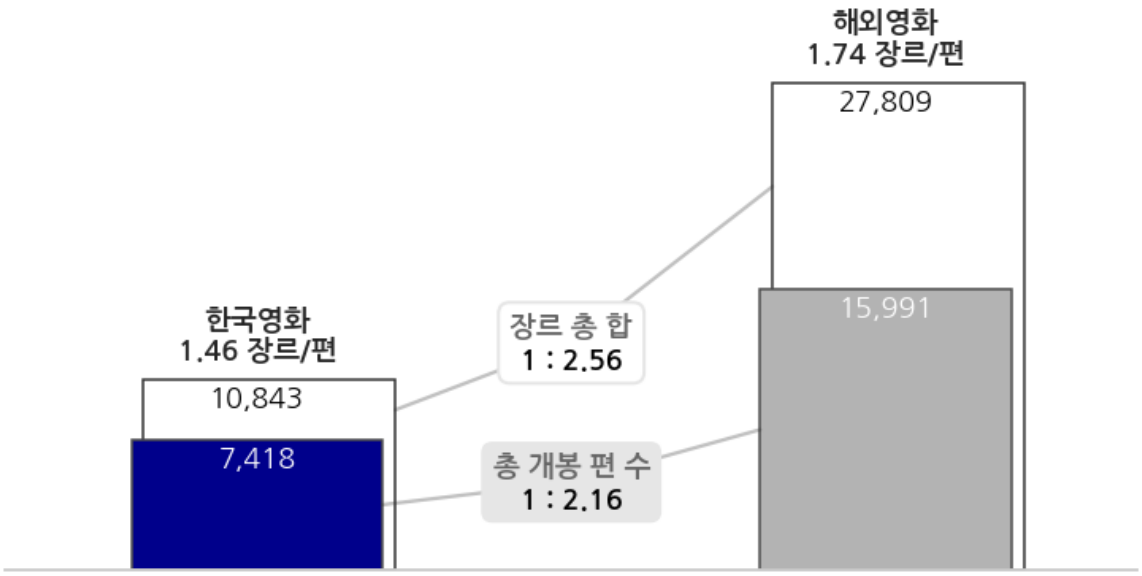


3.2. 한국영화 vs 해외영화 전체

In [ ]:

```
fig, ax = plt.subplots(figsize=(10, 5), constrained_layout=True)

# 장르 수
n_genres_kr = df_genres.filter(like="G_").sum(axis=1).loc[df_nations["N_한국"]==1].sum()
n_genres_nkr = df_genres.filter(like="G_").sum(axis=1).loc[df_nations["해외"]==1].sum()
ax.bar(0.02, n_genres_kr,
       width=0.4, fc="w", ec="0.2")
ax.bar(1.02, n_genres_nkr,
       width=0.4, fc="w", ec="0.2")

# 영화 수
n_movies_kr = df_nations["N_한국"].sum()
n_movies_nkr = df_nations["해외"].sum()
ax.bar(0, n_movies_kr,
       width=0.4, fc=c_kr, ec="0.3")
ax.bar(1, n_movies_nkr,
       width=0.4, fc=c_etc, ec="0.3")

# text
offset = 300
ax.text(0, n_genres_kr-offset, format(n_genres_kr, ","),
        fontsize="medium", ha="center", va="top", color="k")
ax.text(1, n_genres_nkr-offset, format(n_genres_nkr, ","),
        fontsize="medium", ha="center", va="top", color="k")
ax.text(0, n_movies_kr-offset, format(n_movies_kr, ","),
        fontsize="medium", ha="center", va="top", color="w")
ax.text(1, n_movies_nkr-offset, format(n_movies_nkr, ","),
        fontsize="medium", ha="center", va="top", color="w")

ax.text(0, n_genres_kr+offset*3, f"한국영화₩n{n_genres_kr/n_movies_kr:.2f} 장르/편",
        fontweight="bold", ha="center", va="bottom")
ax.text(1, n_genres_nkr+offset*3, f"해외영화₩n{n_genres_nkr/n_movies_nkr:.2f} 장르/편",
        fontweight="bold", ha="center", va="bottom")

### legend
p = 0.7

# 총 개봉 편수
ratio_num = n_movies_nkr/n_movies_kr
h_num = p*n_movies_kr/2 + (1-p)*n_movies_nkr/2
ax.text(0.5, h_num,
        f"총 개봉 편 수₩n", color="0.4", fontweight="bold", ha="center", va="center", zorder=2,
        bbox={"boxstyle":"round", "pad":0.4, "facecolor":'0.9', "edgecolor":'none', "linewidth":
0})
ax.text(0.5, h_num-1000,
        f"1 : {ratio_num:.2f}", color="k", fontweight="bold", ha="center", va="center", zorder
=2)

# 장르 총 합
ratio_genre = n_genres_nkr/n_genres_kr
h_genre = p*((n_genres_kr-n_movies_kr)/2 + n_movies_kr) + (1-p)*((n_genres_nkr-n_movies_nkr)/2 +
n_movies_nkr)
ax.text(0.5, h_genre,
        "장르 총 합₩n", color="0.4", fontweight="bold", ha="center", va="center", zorder=2,
        bbox={"boxstyle":"round", "pad":0.4, "facecolor":'w', "edgecolor":'0.9', "linewidth":2})
ax.text(0.5, h_genre-1000,
        f"1 : {ratio_genre:.2f}", color="k", fontweight="bold", ha="center", va="center", zord
er=2)
```

```
# line
ax.plot([0.2, 0.5, 0.8],
        [n_movies_kr/2, p*n_movies_kr/2 + (1-p)*n_movies_nkr/2, n_movies_nkr/2],
        c="0.2", alpha=0.3)
ax.plot([0.22, 0.5, 0.82],
        [(n_genres_kr-n_movies_kr)/2 + n_movies_kr,
         p*((n_genres_kr-n_movies_kr)/2 + n_movies_kr) + (1-p)*((n_genres_nkr-n_movies_nkr)/2 +
n_movies_nkr),
         (n_genres_nkr-n_movies_nkr)/2 + n_movies_nkr],
        c="0.2", alpha=0.3)

ax.spines[["left", "top", "right"]].set_visible(False)
ax.set_xlim(-0.4, 1.4)
ax.set_xticks([0, 1])
ax.set_xticklabels([])
ax.set_yticks([])
# ax.set_yticks([0, 10000, 20000, 30000])
ax.grid(axis="x")

fig.savefig("./images/genres_movie_ratio.png", dpi=200)
```



3.3. 년도별 제작국가

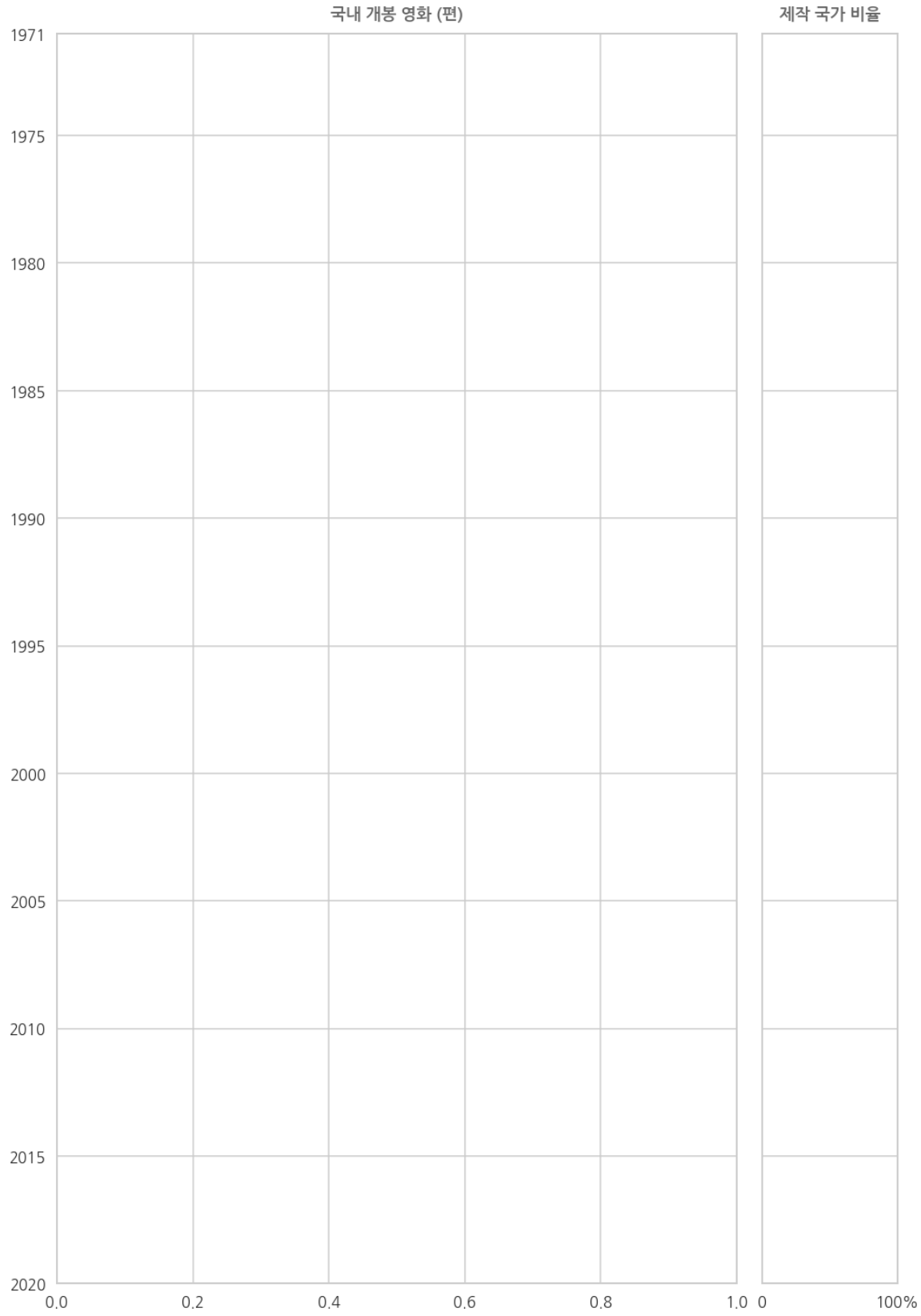
In [ ]:

```
# 틀 만들기
fig, axs = plt.subplots(ncols=2, gridspec_kw={"width_ratios":[5,1]},
                        sharey=True,
                        figsize=(14, 20), constrained_layout=True)

axs[0].set_title("국내 개봉 영화 (편)", fontdict=font_title, pad=16)
axs[1].set_title("제작 국가 비율", fontdict=font_title, pad=16)

portion_aspect0 = axs[0].get_position().height/axs[0].get_position().width
portion_aspect1 = axs[1].get_position().height/axs[1].get_position().width
print(portion_aspect0)

yticks = [1971] + list(range(1975, 2025, 5))
axs[0].set_ylim(2020, 1971)
axs[0].set_yticks(yticks)
axs[0].set_yticklabels(yticks)

axs[1].set_xticks([0, 1])
axs[1].set_xticklabels([0, "100%"])
```

1.285935483870968

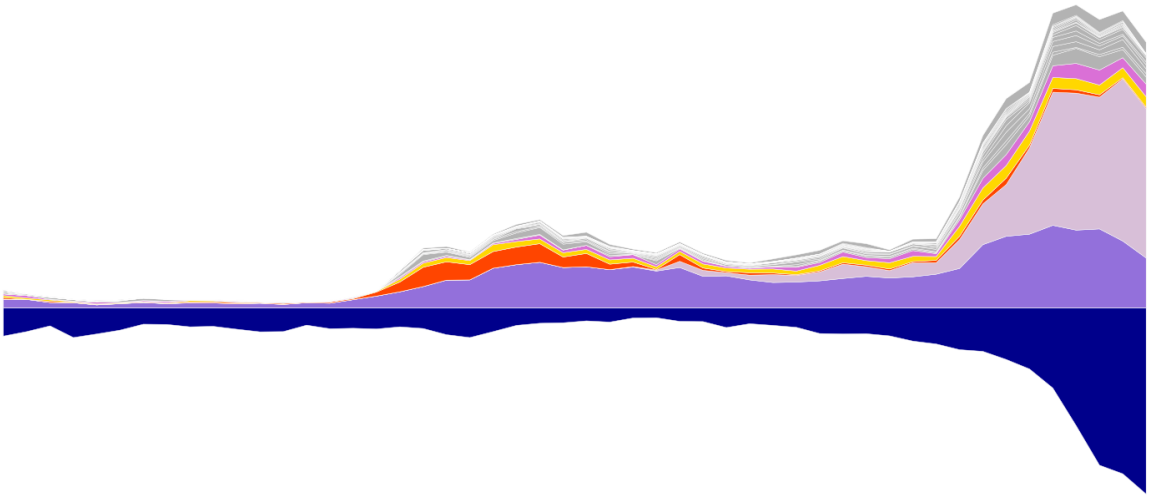
Out[ ]:

[Text(0, 0, '0'), Text(1, 0, '100%')]



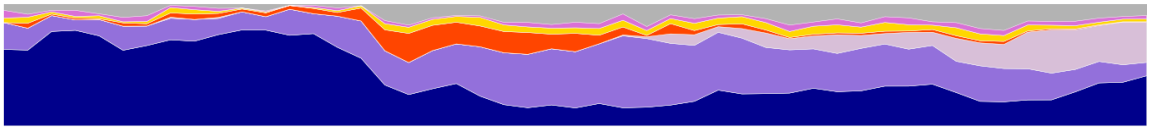
In [ ]:

```
# 국내 개봉 영화 편 수

fig_p0, ax_p0 = plt.subplots(figsize=(axs[0].get_position().height * 20, axs[0].get_position().w
idth * 14), constrained_layout=True)

ax_p0.stackplot(df_nationsY["openYear"], -df_nationsY["N_한국"], colors=c_kr, ec="w", lw=0.5)
ax_p0.stackplot(df_nationsY["openYear"], df_nationsY["해외"], colors=c_etc, ec="w", lw=0.5)

nationsY_top20_foreign = []
for n in df_nations_top20.index[1:]:
    nationsY_top20_foreign.append(df_nationsY[f'{n}'])

ax_p0.stackplot(df_nationsY["openYear"],
                nationsY_top20_foreign,
                colors=[c_us, c_jp, c_hk, c_fr, c_gb] + [c_etc]*14, ec="w", lw=0.5)

ax_p0.set_ylim(-1300, 1300)
ax_p0.set_xlim(1971, 2020)
ax_p0.axis(False)

fig_p0.savefig("./images/portion_year0.png", dpi=200, pad_inches=0)
```



In [ ]:

```
# 영화 편 수 비율
fig_p, ax_p = plt.subplots(figsize=(axs[1].get_position().height * 20, axs[1].get_position().wid
th * 14), constrained_layout=True)

ax_p.stackplot(df_nationsY.index,
               df_nationsY["N_한국"]/df_nationsY.filter(like="N_").sum(axis=1),
               df_nationsY["N_미국"]/df_nationsY.filter(like="N_").sum(axis=1),
               df_nationsY["N_일본"]/df_nationsY.filter(like="N_").sum(axis=1),
               df_nationsY["N_홍콩"]/df_nationsY.filter(like="N_").sum(axis=1),
               df_nationsY["N_프랑스"]/df_nationsY.filter(like="N_").sum(axis=1),
               df_nationsY["N_영국"]/df_nationsY.filter(like="N_").sum(axis=1),
             colors=[c_kr, c_us, c_jp, c_hk, c_fr, c_gb], ec="w", lw=0.5)
ax_p.set_facecolor(c_etc)
ax_p.set_xlim(1, df_nationsY.shape[0]-1)
ax_p.set_ylim(0, 1)
ax_p.set_xticks([])
ax_p.set_yticks([])
ax_p.spines[["top", "bottom", "left", "right"]].set_visible(False)
fig_p.savefig("./images/portion_year1.png", dpi=200, pad_inches=0)
```



In [ ]:

```
# 역사적 사건들
def plot_history(year, text, text_x_shift=50, text_y=None, text_c="green", text_size="medium",
text_fc="w", text_align="center",
                 c_h0 = "limegreen", c_h1 = "palegreen", alpha_h1=0.7, ax=axs):
    if not text_y:
        text_y = year-0.5

    y_line = [year] * 100

    if np.array(axs == None).any():
        ax = plt.gca()
        x0_line = np.linspace(ax.get_xbound()[1], ax.get_xbound()[0], 100)
        x1_line = 0
        ax_0 = ax
    elif isinstance(ax, np.ndarray):
        x0_line = np.linspace(axs[0].get_xbound()[1], axs[0].get_xbound()[0], 100)
        x1_line = np.linspace(axs[1].get_xbound()[1], axs[1].get_xbound()[0], 100)
        ax_0 = ax[0]
    else:
        x0_line = np.linspace(axs[0].get_xbound()[1], axs[0].get_xbound()[0], 100)
        x1_line = 0
        ax_0 = ax

    # axs[0]
    for i in range(99):
        ax_0.plot(x0_line[i:i+2], y_line[i:i+2], c=c_h0,
                  solid_capstyle='butt', alpha=np.power(np.sin(i/100),6)*2, zorder=15)
    ax_0.text(ax_0.get_xbound()[0]+text_x_shift, text_y, text, c=text_c, fontsize=text_size,
              multialignment=text_align, ha="left",
              bbox={"boxstyle":"square", "pad":0.4, "facecolor":text_fc, "edgecolor":"none", "li
newidth":1})

    # axs[1:]
    if isinstance(x1_line, np.ndarray):
        for ax_ in ax[1:]:
            for i in range(99):
                ax_.plot(x1_line[i:i+2], y_line[i:i+2], c=c_h1,
                         solid_capstyle='butt', alpha=alpha_h1, zorder=15)
```

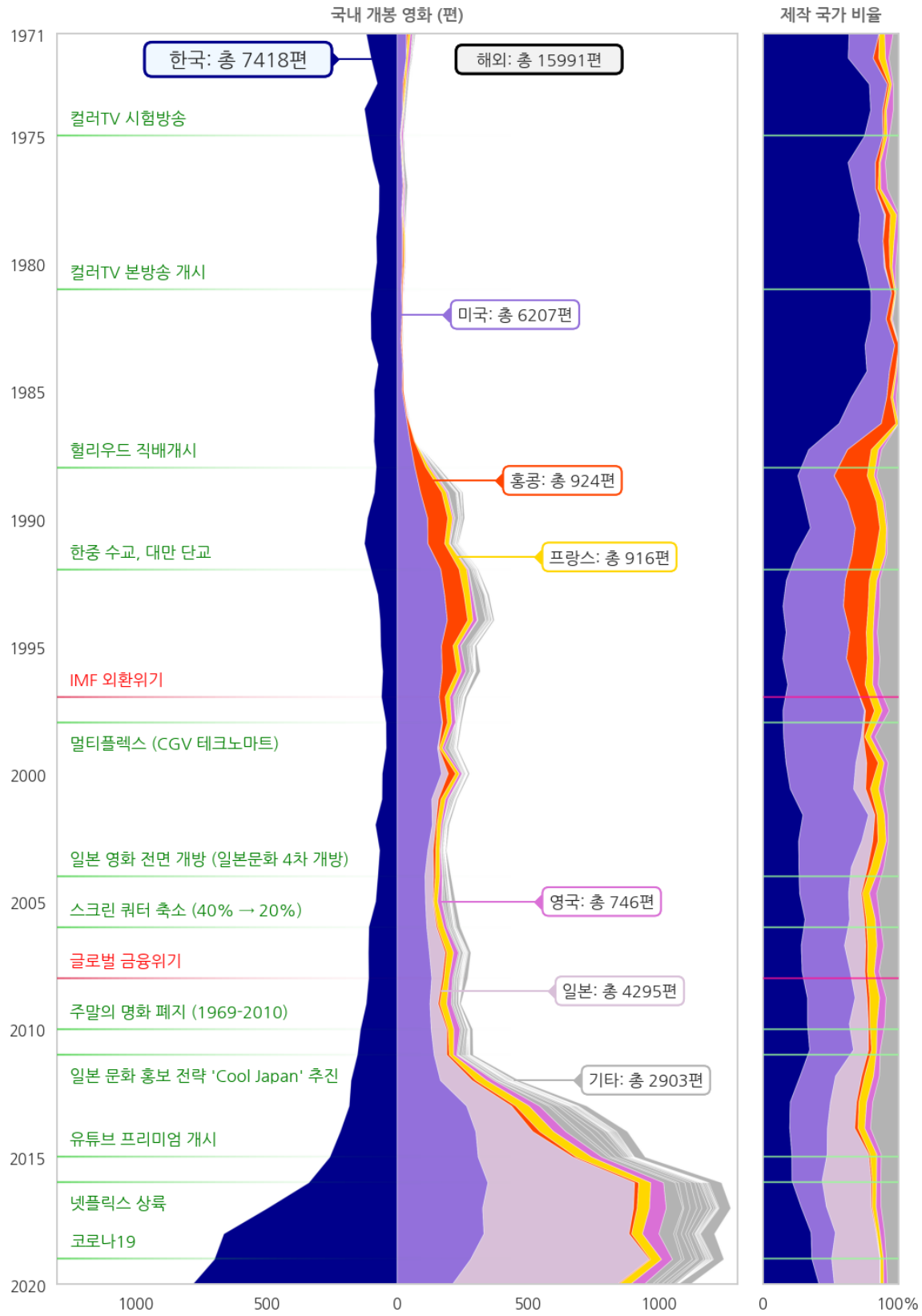
In [ ]:

```
## 조립

# axs[0]
fig0_img = plt.imread("./images/portion_year0.png")
fig0_img = fig0_img.swapaxes(0, 1)[:,::-1, :][10:-10,10:-10,:]
print(fig0_img.shape)
x0, x1 = -1300, 1300
y1, y0 = 1971, 2020

axs[0].imshow(fig0_img, extent=[x0, x1, y0, y1])
axs[0].set_aspect(abs(x1-x0)/abs(y1-y0)*portion_aspect0 *20/14)

xticks = [-1000, -500, 0, 500, 1000]
axs[0].set_xticks(xticks)
axs[0].set_xticklabels([abs(x) for x in xticks])
for y in yticks:
    axs[0].axhline(y, c="lightgray", alpha=0.2, zorder=-1)

axs[0].grid(False)

# plot_stem
def plot_stem(x, y, x0=0, s=100, ls="-", lw=2, c="cyan", marker="D",
              text="sample", text_size="medium", position="right", bbox_lw=2, bbox_fc="w", ax=No
ne, **kwargs):
    ax.plot([x0, x], [y, y], ls=ls, lw=lw, c=c)
    ax.scatter(x, y, s=s, c=c, marker=marker, **kwargs)
    if text:
        offset = 30
        text_pos = x+offset if position == "right" else x-offset
        ha = "left" if position == "right" else "right"
        va = "center"
        ax.text(text_pos, y, text, ha=ha, va=va, fontsize=text_size,
                bbox={"boxstyle":"round", "pad":0.4,
                      "facecolor":bbox_fc, "edgecolor":c, "linewidth":bbox_lw},
                zorder=20)

### Legends
handles, labels = axs[0].get_legend_handles_labels()

# 한국
plot_stem(-250, 1972, x0=-100, c=c_kr, bbox_lw=3, text=f"   한국: 총 {df_nations['N_한국'].sum
()}편   ",
          text_size="large", bbox_fc="aliceblue", position="left", ax=axs[0])

# 미국
plot_stem(200, 1982, x0=10, c=c_us, text=f"미국: 총 {df_nations_top20['N_미국']}편", ax=axs[
0])

# 일본
plot_stem(600, 2008.5, x0=170, c=c_jp, text=f"일본: 총 {df_nations_top20['N_일본']}편", ax=a
xs[0])

# 프랑스
plot_stem(550, 1991.5, x0=220, c=c_fr, text=f"프랑스: 총 {df_nations_top20['N_프랑스']}편",
ax=axs[0])

# 홍콩
plot_stem(400, 1988.5, x0=95, c=c_hk, text=f"홍콩: 총 {df_nations_top20['N_홍콩']}편", ax=ax
s[0])
```

```
# 영국
plot_stem(550, 2005, x0=170, c=c_gb, text=f"영국: 총 {df_nations_top20['N_영국']}편", ax=axs
[0])

# 기타
plot_stem(700, 2012, x0=450, c=c_etc, text=f"기타: 총 {df_nations['해외'].sum()-df_nations_top2
0.iloc[1:6].sum()}편",
          ax=axs[0])

axs[0].text(250, 1972, f"   해외: 총 {df_nations['해외'].sum()}편   ", ha="left", va="center",
            bbox={"boxstyle":"round", "pad":0.4,
                  "facecolor":"0.95", "edgecolor":"k", "linewidth":3},
            zorder=20)

# axs[1]
fig1_img = plt.imread("./images/portion_year1.png")
fig1_img = fig1_img.swapaxes(0, 1)[:,::-1, :][10:-10,10:-10,:]
x0, x1 = 0, 1
y1, y0 = 1971, 2020

axs[1].imshow(fig1_img, extent=[x0, x1, y0, y1])
axs[1].set_aspect(abs(x1-x0)/abs(y1-y0)*portion_aspect1 * 20/14)
axs[1].grid(False)


# 컬러TV 방송
plot_history(1975, "컬러TV 시험방송")
plot_history(1981, "컬러TV 본방송 개시")

# 헐리우드 직배영화
plot_history(1988, "헐리우드 직배개시")

# 한중수교
plot_history(1992, "한중 수교, 대만 단교")

# IMF
plot_history(1997, "IMF 외환위기", text_c="r", c_h0="crimson", c_h1="deeppink")

# 멀티플렉스
plot_history(1998, "멀티플렉스 (CGV 테크노마트)", text_y=1999)

# 일본 영화 전면 개방
plot_history(2004, "일본 영화 전면 개방 (일본문화 4차 개방)")

# 스크린쿼터
plot_history(2006, "스크린 쿼터 축소 (40% → 20%)")

# 글로벌 금융위기
plot_history(2008, "글로벌 금융위기", text_c="r", c_h0="crimson", c_h1="deeppink")

# 주말의 명화 폐지
plot_history(2010, "주말의 명화 폐지 (1969-2010)")

# 일본 쿨 재팬 전략
plot_history(2011, "일본 문화 홍보 전략 'Cool Japan' 추진", text_y=2012)

# 유튜브 프리미엄 개시
plot_history(2015, "유튜브 프리미엄 개시")

# 넷플릭스 상륙
```

```
plot_history(2016, "넷플릭스 상륙", text_y=2017)

# 코로나19
plot_history(2019, "코로나19")

axs[1].set_xlim(0, 1)

fig.savefig("./images/num_year.png", dpi=200)
display(fig)
```

(3800, 2057, 4)


국내 개봉 영화 (편)
제작 국가 비율
한국: 총 7418편
해외: 총 15991편
미국: 총 6207편
홍콩: 총 924편
프랑스: 총 916편
영국: 총 746편
일본: 총 4295편
기타: 총 2903편
1971
1975
1980
1985
1990
1995
2000
2005
2010
2015
2020
컬러TV 시험방송
컬러TV 본방송 개시
헐리우드 직배개시
한중 수교, 대만 단교
IMF 외환위기
멀티플렉스 (CGV 테크노마트)
일본 영화 전면 개방 (일본문화 4차 개방)
스크린 쿼터 축소 (40% → 20%)
글로벌 금융위기
주말의 명화 폐지 (1969-2010)
일본 문화 홍보 전략 'Cool Japan' 추진
유튜브 프리미엄 개시
넷플릭스 상륙
코로나19
1000
500
0
500
1000
0
100%

3.4. 년도별 장르

In [ ]:

```
# 기타 제외
genres_noetc = [g.split("_")[1] for g in df_genres_top21.index if "기타" not in g]
genres_noetc
```

Out[ ]:

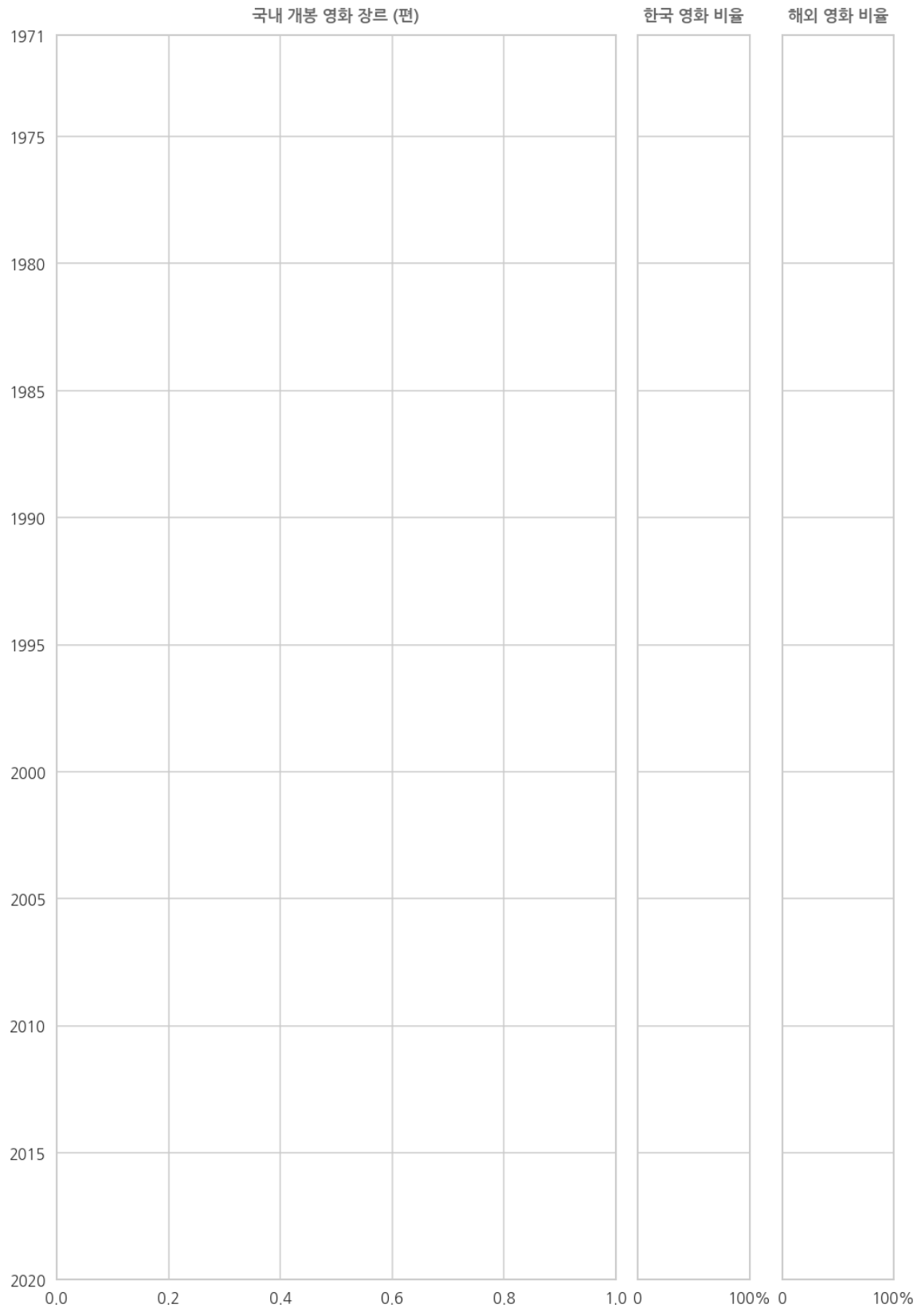
```
['드라마',
 '멜로/로맨스',
 '액션',
 '코미디',
 '스릴러',
 '성인물',
 '공포',
 '범죄',
 '애니메이션',
 '어드벤처',
 'SF',
 '판타지',
 '미스터리',
 '다큐멘터리',
 '가족',
 '사극',
 '전쟁',
 '공연',
 '뮤지컬',
 '서부극']
```

In [ ]:

```
# 장르별 색상
cmap = plt.get_cmap("tab20")

c_drama = cmap(0/20)        # 드라마
c_romance = cmap(1/20)      # 멜로/로맨스
c_action = cmap(2/20)       # 액션
c_comedy = cmap(3/20)       # 코미디
c_thriller = cmap(12/20)    # 스릴러
c_ero = cmap(13/20)         # 성인물
c_horror = cmap(6/20)       # 공포
c_crime = cmap(7/20)        # 범죄
c_ani = cmap(8/20)          # 애니메이션
c_adv = cmap(9/20)          # 어드벤처
c_sf = cmap(10/20)          # SF
c_fantasy = cmap(11/20)     # 판타지
c_mistery = cmap(4/20)      # 미스터리
c_docu = cmap(5/20)         # 다큐멘터리
c_family = cmap(14/20)      # 가족
c_history = cmap(15/20)     # 사극
c_war = cmap(16/20)         # 전쟁
c_play = cmap(17/20)        # 공연
c_musical = cmap(18/20)     # 뮤지컬
c_western = cmap(19/20)     # 서부극

c_genres = [c_drama, c_romance, c_action, c_comedy, c_thriller, c_ero, c_horror, c_crime, c_ani,
c_adv, c_sf, c_fantasy, c_mistery, c_docu, c_family, c_history, c_war, c_play, c_musical, c_west
ern, c_etc]
```

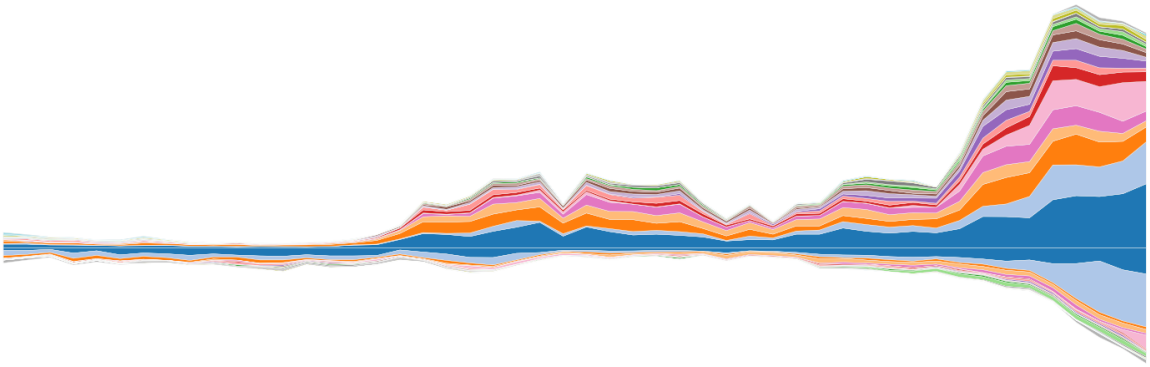
In [ ]:

```
# 틀 만들기
fig, axs = plt.subplots(ncols=3, gridspec_kw={"width_ratios":[5,1,1]},
                        sharey=True,
                        figsize=(14, 20), constrained_layout=True)

axs[0].set_title("국내 개봉 영화 장르 (편)", fontdict=font_title, pad=16)
axs[1].set_title("한국 영화 비율", fontdict=font_title, pad=16)
axs[2].set_title("해외 영화 비율", fontdict=font_title, pad=16)

portion_aspect0 = axs[0].get_position().height/axs[0].get_position().width
portion_aspect1 = axs[1].get_position().height/axs[1].get_position().width
portion_aspect2 = axs[2].get_position().height/axs[2].get_position().width

yticks = [1971] + list(range(1975, 2025, 5))
axs[0].set_ylim(2020, 1971)
axs[0].set_yticks(yticks)
axs[0].set_yticklabels(yticks)

for ax in axs[1:]:
    ax.set_xticks([0, 1])
    ax.set_xticklabels([0, "100%"])
```


국내 개봉 영화 장르 (편)
한국 영화 비율
해외 영화 비율
1971
1975
1980
1985
1990
1995
2000
2005
2010
2015
2020
0.0
0.2
0.4
0.6
0.8
1.0
0
100%
0
100%

In [ ]:

```
# 년도별 장르
sns.set_palette("tab20")

fig_p0, ax_p0 = plt.subplots(figsize=(axs[0].get_position().height * 20, axs[0].get_position().w
idth * 14), constrained_layout=True)

stack_ys_kr = []
stack_ys_nkr = []
for g in genres_noetc:
    stack_ys_kr.append(eval(f"-df_genresY_kr['G_{g}']"))
    stack_ys_nkr.append(eval(f"df_genresY_nkr['G_{g}']"))

ax_p0.stackplot(df_genresY_kr["openYear"], *stack_ys_kr,
                ec="w", lw=0.4, colors=c_genres[:20])
ax_p0.stackplot(df_genresY_nkr["openYear"], *stack_ys_nkr,
                ec="w", lw=0.4, colors=c_genres[:20])

# G_기타
ax_p0.stackplot(df_genresY_kr["openYear"],
                -df_genresY_kr.filter(like="G_").sum(axis=1),
                ec="w", lw=0.4, zorder=-1)
ax_p0.stackplot(df_genresY_nkr["openYear"],
                df_genresY_nkr.filter(like="G_").sum(axis=1),
                ec="w", lw=0.4, zorder=-1)
ax_p0.get_children()[40].set_facecolor(c_etc)
ax_p0.get_children()[41].set_facecolor(c_etc)

ax_p0.set_xlim(1971, 2020)
ax_p0.set_ylim(-2200, 2200)
ax_p0.axis(False)
fig_p0.savefig("./images/genres_year0.png")
```

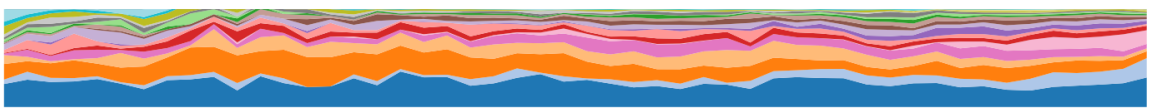


In [ ]:

```
# 한국영화 장르 비율

stack_ys_kr.append(-df_genresY_kr['G_기타'])

stack_ys_kr_p = np.array(stack_ys_kr)/np.array(stack_ys_kr).sum(axis=0)
stack_ys_kr_p.shape

fig_p1, ax_p1 = plt.subplots(figsize=(axs[1].get_position().height * 20, axs[1].get_position().w
idth * 14), constrained_layout=True)

ax_p1.stackplot(df_genresY_kr["openYear"], *stack_ys_kr_p, colors=c_genres,
                ec="none", lw=0.5)
ax_p1.get_children()[20].set_facecolor(c_etc)
ax_p1.set_xlim(1971, 2020)
ax_p1.set_ylim(0, 1)
ax_p1.set_xticks([])
ax_p1.set_yticks([])
ax_p1.spines[["top", "bottom", "left", "right"]].set_visible(False)
fig_p1.savefig("./images/genres_year1.png", dpi=200)
```

In [ ]:

```
# 해외영화 장르 비율

stack_ys_nkr.append(-df_genresY_nkr['G_기타'])

stack_ys_nkr_p = np.array(stack_ys_nkr)/np.array(stack_ys_nkr).sum(axis=0)
stack_ys_nkr_p.shape

fig_p2, ax_p2 = plt.subplots(figsize=(axs[2].get_position().height * 20, axs[2].get_position().w
idth * 14), constrained_layout=True)

ax_p2.stackplot(df_genresY_nkr["openYear"], *stack_ys_nkr_p, colors=c_genres,
                ec="none", lw=0.5)
ax_p2.get_children()[20].set_facecolor(c_etc)
ax_p2.set_xlim(1971, 2020)
ax_p2.set_ylim(0, 1)
ax_p2.set_xticks([])
ax_p2.set_yticks([])
ax_p2.spines[["top", "bottom", "left", "right"]].set_visible(False)
fig_p2.savefig("./images/genres_year2.png", dpi=200)
```

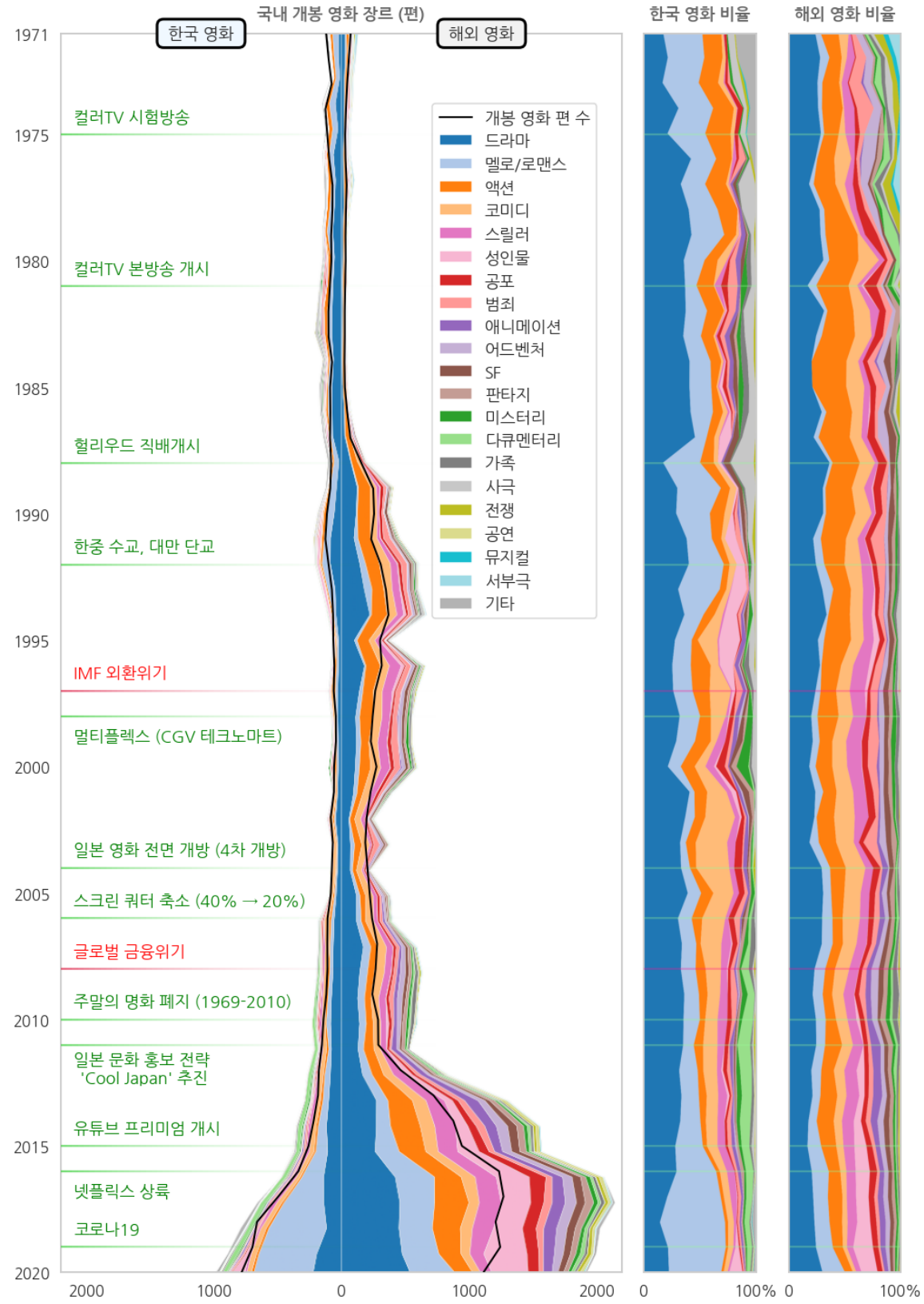


In [ ]:

```
## 조립

# axs[0]
fig0_img = plt.imread("./images/genres_year0.png")
fig0_img = fig0_img.swapaxes(0, 1)[:,::-1, :][10:-10,10:-10,:]
x0, x1 = -2200, 2200
y1, y0 = 1971, 2020

axs[0].imshow(fig0_img, extent=[x0, x1, y0, y1])
axs[0].set_aspect(abs(x1-x0)/abs(y1-y0)*portion_aspect0*20/14)

axs[0].plot(-df_nationsY["N_한국"], df_nationsY["openYear"],
            c="k", lw=2, zorder=19, label="개봉 영화 편 수")
axs[0].plot(df_nationsY["해외"], df_nationsY["openYear"],
            c="k", lw=2, zorder=19)
axs[0].grid(False)

axs[0].set_xlim(-2200, 2200)
xticks = [-2000, -1000, 0, 1000, 2000]
axs[0].set_xticks(xticks)
axs[0].set_xticklabels([abs(x) for x in xticks])
yticks = [1971] + list(range(1975, 2025, 5))
axs[0].set_yticks(yticks)
axs[0].set_yticklabels(yticks)

axs[0].text(-1100, 1971, f" 한국 영화 ", ha="center", va="center",
            bbox={"boxstyle":"round", "pad":0.4,
                  "facecolor":"aliceblue", "edgecolor":"k", "linewidth":3},
            zorder=20)


axs[0].text(1100, 1971, f" 해외 영화 ", ha="center", va="center",
            bbox={"boxstyle":"round", "pad":0.4,
                  "facecolor":"0.95", "edgecolor":"k", "linewidth":3},
            zorder=20)

# axs[1] 한국
fig1_img = plt.imread("./images/genres_year1.png")
fig1_img = fig1_img.swapaxes(0, 1)[:,::-1, :][10:-10,10:-10,:]
x0, x1 = 0, 1
y1, y0 = 1971, 2020

axs[1].imshow(fig1_img, extent=[x0, x1, y0, y1])
axs[1].set_aspect(abs(x1-x0)/abs(y1-y0)*portion_aspect1*20/14)
axs[1].grid(False)

# axs[2] 해외
fig2_img = plt.imread("./images/genres_year2.png")
fig2_img = fig2_img.swapaxes(0, 1)[:,::-1, :][10:-10,10:-10,:]
x0, x1 = 0, 1
y1, y0 = 1971, 2020

axs[2].imshow(fig2_img, extent=[x0, x1, y0, y1])
axs[2].set_aspect(abs(x1-x0)/abs(y1-y0)*portion_aspect2*20/14)
axs[2].grid(False)
### history

# 컬러TV 방송
plot_history(1975, "컬러TV 시험방송", text_x_shift=100, text_fc='none', alpha_h1=0.3, ax=axs)
```

```
plot_history(1981, "컬러TV 본방송 개시", text_x_shift=100, text_fc='none', alpha_h1=0.3, ax=axs)

# 헐리우드 직배영화
plot_history(1988, "헐리우드 직배개시", text_x_shift=100, text_fc='none', alpha_h1=0.3, ax=axs)

# 한중수교
plot_history(1992, "한중 수교, 대만 단교", text_x_shift=100, text_fc='none', alpha_h1=0.3, ax=ax
s)

# IMF
plot_history(1997, "IMF 외환위기", text_x_shift=100, text_fc='none', alpha_h1=0.3,
             text_c="r", c_h0="crimson", c_h1="deeppink", ax=axs)

# 멀티플렉스
plot_history(1998, "멀티플렉스 (CGV 테크노마트)", text_x_shift=100, text_fc='none', alpha_h1=0.3
,
             text_y=1999, ax=axs)

# 일본 영화 전면 개방
plot_history(2004, "일본 영화 전면 개방 (4차 개방)", text_x_shift=100, text_fc='none', alpha_h1=
0.3, ax=axs)

# 스크린쿼터
plot_history(2006, "스크린 쿼터 축소 (40% → 20%)", text_x_shift=100, text_fc='none', alpha_h1=
0.3, ax=axs)

# 글로벌 금융위기
plot_history(2008, "글로벌 금융위기", text_x_shift=100, text_fc='none', alpha_h1=0.3,
             text_c="r", c_h0="crimson", c_h1="deeppink", ax=axs)

# 주말의 명화 폐지
plot_history(2010, "주말의 명화 폐지 (1969-2010)",
             text_x_shift=100, text_fc='none', alpha_h1=0.3, ax=axs)

# 일본 쿨 재팬 전략
plot_history(2011, "일본 문화 홍보 전략 Wn'Cool Japan' 추진", text_x_shift=100, text_fc='none',
alpha_h1=0.3,
             text_y=2012.5, ax=axs)

# 유튜브 프리미엄 개시
plot_history(2015, "유튜브 프리미엄 개시", text_x_shift=100, text_fc='none', alpha_h1=0.3, ax=ax
s)

# 넷플릭스 상륙
plot_history(2016, "넷플릭스 상륙", text_x_shift=100, text_fc='none', alpha_h1=0.3, ax=axs, text
_y=2017)

# 코로나19
plot_history(2019, "코로나19", text_x_shift=100, text_fc='none', alpha_h1=0.3, ax=axs)

### legend
genres_order = genres_noetc + ["기타"]
for i, c in enumerate(c_genres):
    axs[0].bar(0, 1, label=genres_order[i], fc=c, zorder=-2)
axs[0].bar(0, 1, label="기타", zorder=-2)
axs[0].patches[20].set_facecolor(c_etc)

handles, labels = axs[0].get_legend_handles_labels()
axs[0].legend(handles=handles[:-1], labels=labels[:-1],
    fontsize="medium", bbox_to_anchor=(0.97, 0.95), loc="upper right")
```

```
fig.savefig("./images/genre_year.png", dpi=300)
display(fig)
```



3.5. 국내 개봉작 장르 비율

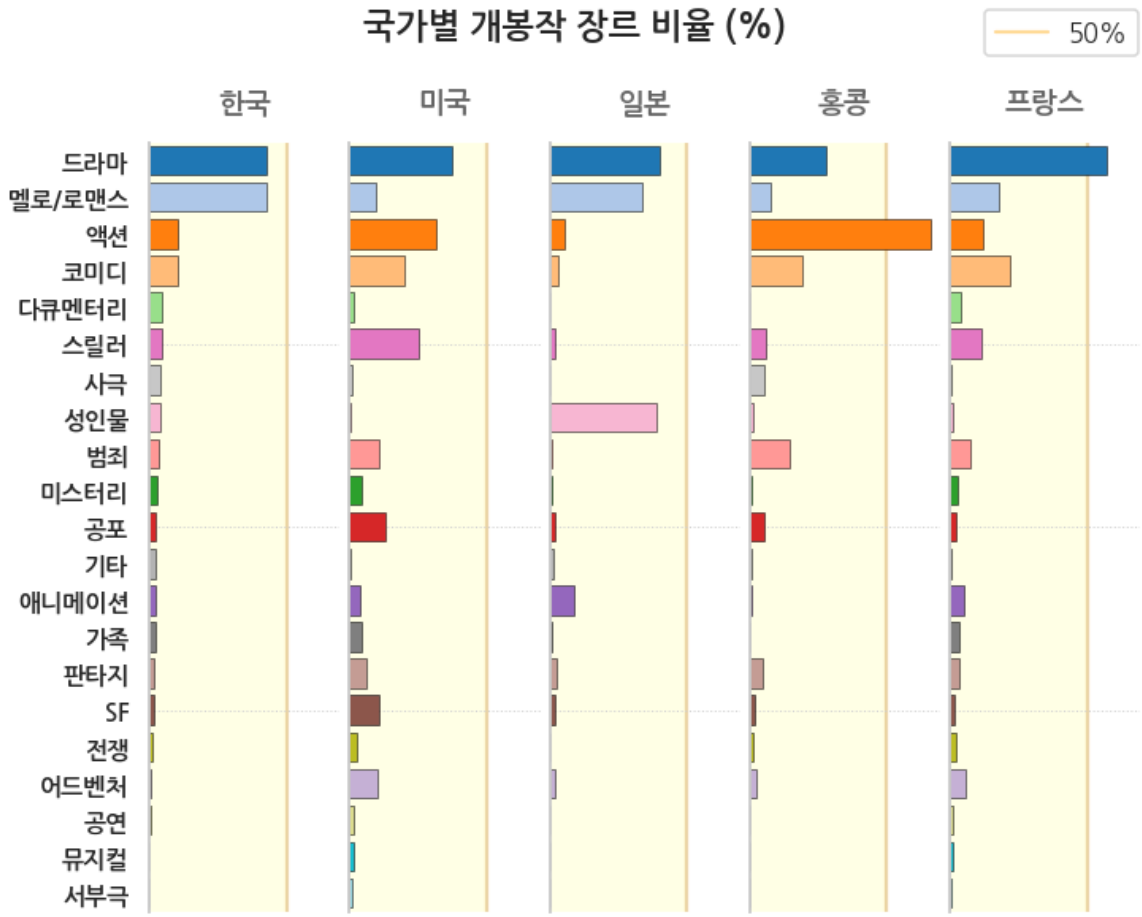
In [ ]:

```
#국가별 상위 장르
# 1. 한국
idx_movies_kr = df_nations.query("N_한국 == 1")[["movieCd", "movieNm", "openYear"]].index
genres_portion_kr = df_genres.loc[idx_movies_kr].filter(like="G_").sum().sort_values(ascending=T
rue)/len(idx_movies_kr)
genres_portion_kr_idx = genres_portion_kr.index
genres_portion_kr_v = genres_portion_kr.values

# 2. 미국
idx_movies_us = df_nations.query("N_미국 == 1")[["movieCd", "movieNm", "openYear"]].index
genres_portion_us = df_genres.loc[idx_movies_us].filter(like="G_").sum().sort_values(ascending=T
rue)/len(idx_movies_us)
genres_portion_us_v = genres_portion_us.loc[genres_portion_kr_idx]

# 3. 일본
idx_movies_jp = df_nations.query("N_일본 == 1")[["movieCd", "movieNm", "openYear"]].index
genres_portion_jp = df_genres.loc[idx_movies_jp].filter(like="G_").sum().sort_values(ascending=T
rue)/len(idx_movies_jp)
genres_portion_jp_v = genres_portion_jp.loc[genres_portion_kr_idx]

# 4. 홍콩
idx_movies_hk = df_nations.query("N_홍콩 == 1")[["movieCd", "movieNm", "openYear"]].index
genres_portion_hk = df_genres.loc[idx_movies_hk].filter(like="G_").sum().sort_values(ascending=T
rue)/len(idx_movies_hk)
genres_portion_hk_v = genres_portion_hk.loc[genres_portion_kr_idx]

# 5. 프랑스
idx_movies_fr = df_nations.query("N_프랑스 == 1")[["movieCd", "movieNm", "openYear"]].index
genres_portion_fr = df_genres.loc[idx_movies_fr].filter(like="G_").sum().sort_values(ascending=T
rue)/len(idx_movies_fr)
genres_portion_fr_v = genres_portion_fr.loc[genres_portion_kr_idx]
```

In [ ]:

```
fig, axs = plt.subplots(ncols=5, figsize=(10, 8), constrained_layout=True, sharey=True, sharex=
True)

for ax, n, c, p in zip(axs,
                       ["한국", "미국", "일본", "홍콩", "프랑스"],
                       [c_kr, c_us, c_jp, c_hk, c_fr],
                       [genres_portion_kr_v, genres_portion_us_v, genres_portion_jp_v, genres_po
rtion_hk_v, genres_portion_fr_v]):
    ax.barh(genres_portion_kr_idx, p,
            color=[c_western, c_musical, c_play, c_adv, c_war, c_sf, c_fantasy,
                   c_family, c_ani, c_etc, c_horror, c_mistery, c_crime, c_ero,
                   c_history, c_thriller, c_docu, c_comedy, c_action, c_romance, c_drama],
            height=0.8, ec="k", lw=0.5)

    ax.spines[["top", "right", "bottom"]].set_visible(False)

    ax.grid(axis="y", alpha=0)
    ax.grid(axis="x", alpha=0.5)
    ax.axvline(0.5, c="orange", alpha=0.4, zorder=-1)
    ax.axvspan(0, 0.5, fc="yellow", alpha=0.1, zorder=-1)
    ax.tick_params(color="w")

    xbound = ax.get_xbound()
    xticks = [x for x in np.linspace(0, 1, 3) if xbound[0] <= x <= xbound[1]]
    ax.set_xticks(xticks)
    ax.set_xticklabels([])
    yticklabels = [g.split("_")[1] for g in genres_portion_kr_idx]
    ax.set_yticklabels(yticklabels, size="small", fontweight="bold")

    ax.set_title(n, fontdict=font_title, pad=20)

    for y in [5, 10, 15]:
        ax.axhline(y, c="lightgray", ls=":", lw=1, zorder=-1)

axs[0].set_ylim(-0.5, len(genres_portion_kr)-0.5)
fig.legend(handles=[ax.get_children()[22]], labels=["50%"])
fig.suptitle("국가별 개봉작 장르 비율 (%)₩n", fontweight="bold")
fig.savefig("./images/genres_movie_ratio_nations.png", dpi=200)
```

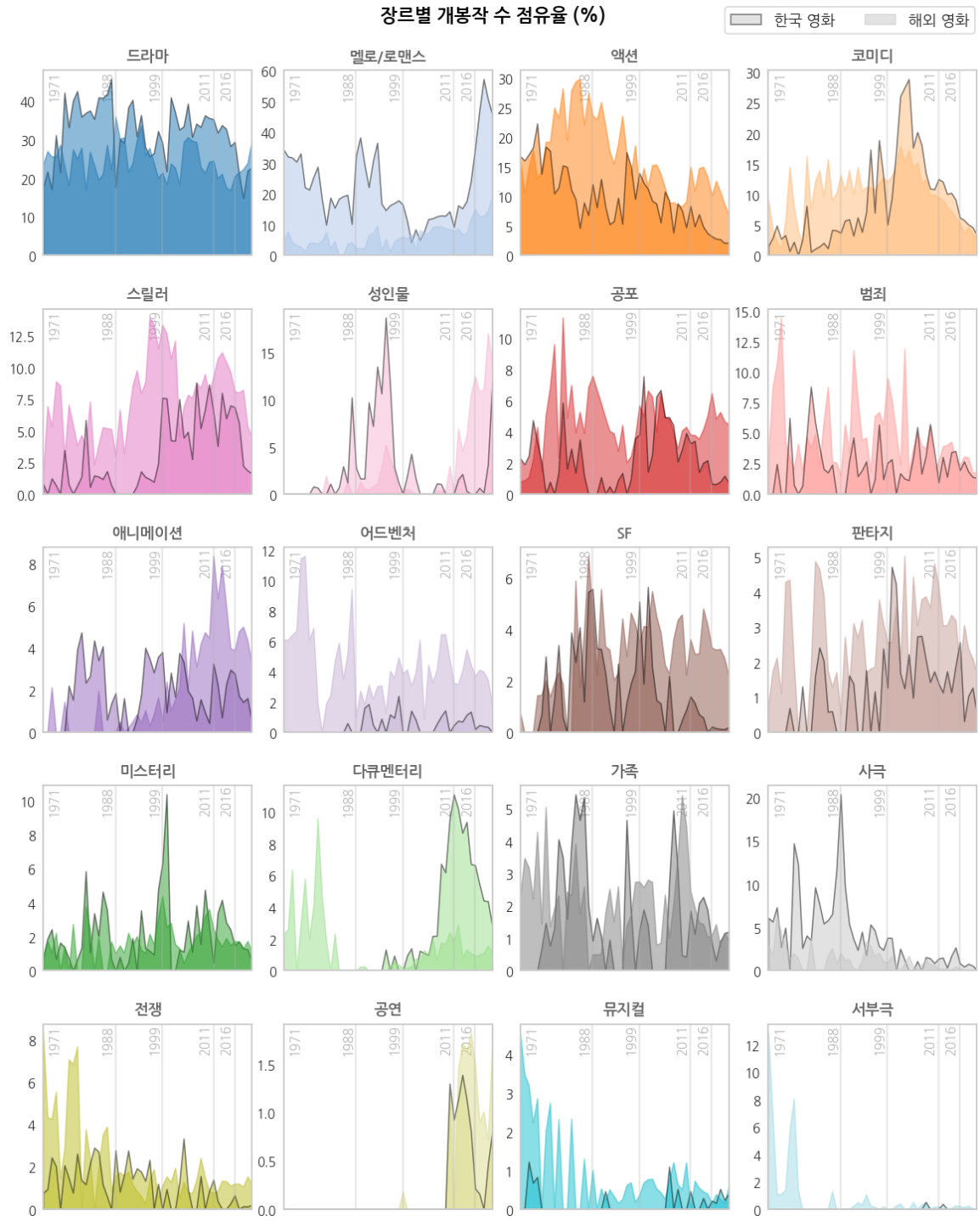
```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:26: UserWarning: Fixe
dFormatter should only be used together with FixedLocator
```



3.6. 시기별 대표 장르

In [ ]:

```
# 장르별 흐름
fig, axes = plt.subplots(ncols=4, nrows=5, figsize=(16, 20),
                         gridspec_kw={"hspace":0.1},
                         sharex=True, constrained_layout=True)
axs = axes.ravel()

z_krs = [1, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 1, 1, 1, 2, 2, 2, 2]

genres_order = genres_noetc + ["기타"]
for g, ax, c, z_kr in zip(genres_order[:20], axs, c_genres[:20], z_krs):
    genre_kr = df_genresY_kr[f"G_{g}"]/df_genresY_kr.filter(like="G_").sum(axis=1)
    genre_nkr = df_genresY_nkr[f"G_{g}"]/df_genresY_nkr.filter(like="G_").sum(axis=1)
    z_nkr = 2 if z_kr == 1 else 1

    ax.fill_between(df_genresY_kr["openYear"],
            genre_kr * 100, 0,
            fc=c, ec="k", alpha=0.5, label="한국 영화", zorder=z_kr)
    ax.fill_between(df_genresY_nkr["openYear"],
            genre_nkr * 100, 0,
            fc=c, ec=c, alpha=0.5, label="해외 영화", zorder=z_nkr)
#     ax.text(2000, 0, g, ha="center")
    ax.set_title(g, fontdict=font_title, fontsize="medium", pad=12)
    ax.set_xlim(1971, 2020)
    ax.set_ylim(0, )
    years = [1971, 1988, 1999, 2011, 2016]
    for year in years:
        ax.axvline(year, alpha=0.3, c=c_etc)
        text_x = year-0.5 if year!= 1971 else year+1
        ha = "right" if year != 1971 else "left"
        ax.text(text_x, ax.get_ybound()[1], year, va="top", ha=ha, fontsize="small", c=c_etc, ro
tation=90, zorder=-1)
    ax.set_xticks(years)
    ax.set_xticklabels([])
    ax.grid(False)
    ax.tick_params(labelsize="small", pad=0)

### label
handles, labels = axs[15].get_legend_handles_labels()
fig.legend(handles=handles[:2], labels=labels[:2], fontsize="medium", ncol=2)

fig.suptitle("장르별 개봉작 수 점유율 (%)₩n", fontweight="bold",  color="k")
fig.savefig("./images/genres_separate.png", dpi=200)
```



3.7. 국가별 시기별 개봉 장르

3.7.1. 데이터 & 시각화 함수

In [ ]:

```
# 국가별 장르 분할

# 미국
df_genres_us = df_genres.loc[df_nations["N_미국"]==1]
df_genresY_us = df_genres_us[["movieCd", "openYear"]].groupby("openYear").count().reset_index().
merge(df_genres_us.iloc[:,2:].groupby("openYear").sum().reset_index())
df_genresY_us.drop("movieCd", axis=1, inplace=True)
df_genresY_us.fillna(0, inplace=True)
df_genresY_us["openYear"] = df_genresY_us["openYear"].astype(int)

# 일본
df_genres_jp = df_genres.loc[df_nations["N_일본"]==1]
df_genresY_jp = df_genres_jp[["movieCd", "openYear"]].groupby("openYear").count().reset_index().
merge(df_genres_jp.iloc[:,2:].groupby("openYear").sum().reset_index())
df_genresY_jp.drop("movieCd", axis=1, inplace=True)
df_genresY_jp.fillna(0, inplace=True)
df_genresY_jp["openYear"] = df_genresY_jp["openYear"].astype(int)

# 홍콩
df_genres_hk = df_genres.loc[df_nations["N_홍콩"]==1]
df_genresY_hk = df_genres_hk[["movieCd", "openYear"]].groupby("openYear").count().reset_index().
merge(df_genres_hk.iloc[:,2:].groupby("openYear").sum().reset_index())
df_genresY_hk.drop("movieCd", axis=1, inplace=True)
df_genresY_hk.fillna(0, inplace=True)
df_genresY_hk["openYear"] = df_genresY_hk["openYear"].astype(int)

# 프랑스
df_genres_fr = df_genres.loc[df_nations["N_프랑스"]==1]
df_genresY_fr = df_genres_fr[["movieCd", "openYear"]].groupby("openYear").count().reset_index().
merge(df_genres_fr.iloc[:,2:].groupby("openYear").sum().reset_index())
df_genresY_fr.drop("movieCd", axis=1, inplace=True)
df_genresY_fr.fillna(0, inplace=True)
df_genresY_fr["openYear"] = df_genresY_fr["openYear"].astype(int)
```

In [ ]:

```
# 최다 5개 국가
N_top5 = ["한국", "미국", "일본", "홍콩", "프랑스"]
```

In [ ]:

```
# 장르별 색상
G_colors = dict(zip([f"G_{g}" for g in genres_order], c_genres))
```

In [ ]:

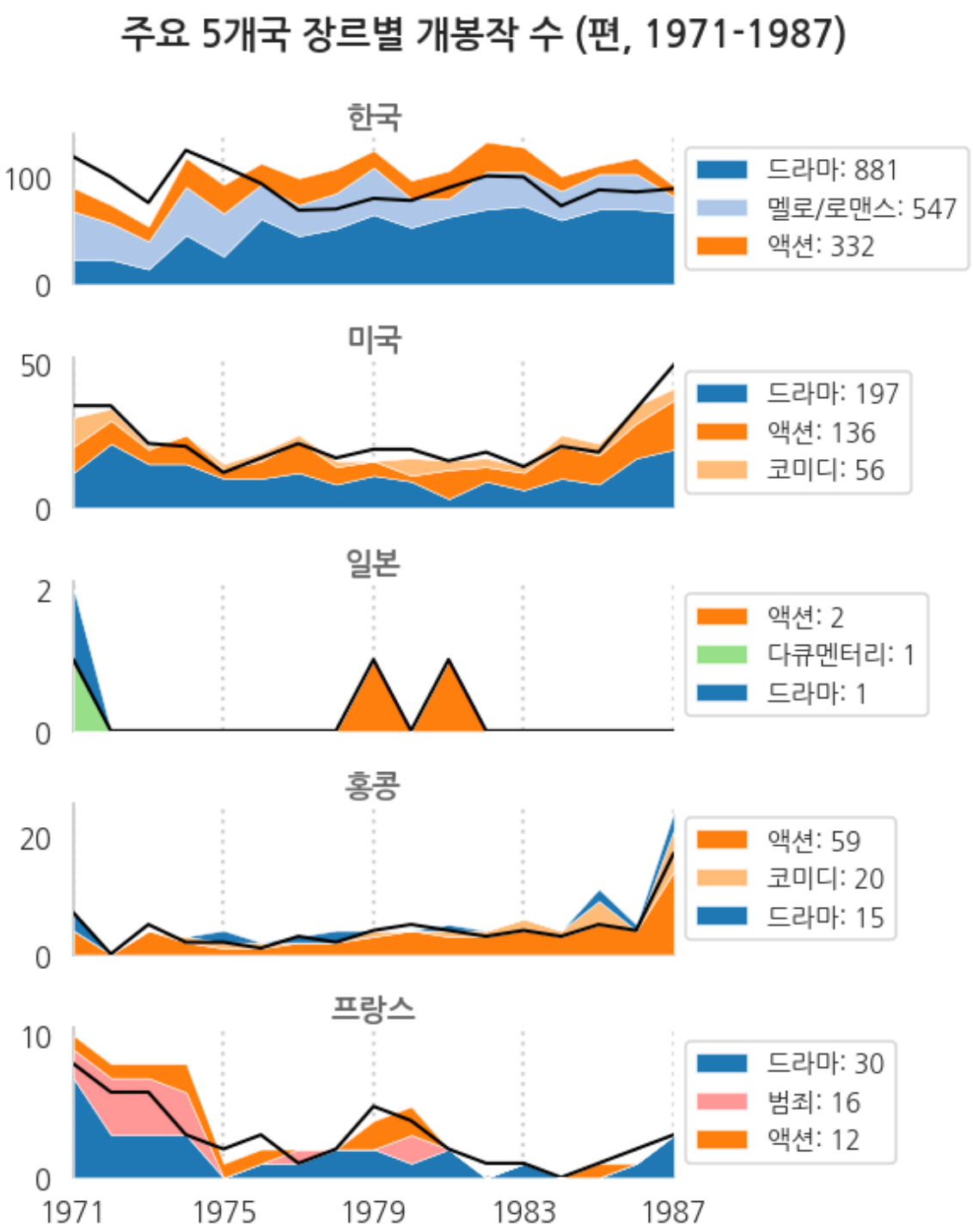
```
# 시각화 함수

def plot_time_GN_(nation, times, ax, genres=[], topN=3):
    time_init = times[0]
    time_fin = times[1]

    if nation == "한국":
        df = df_genresY_kr
    elif nation == "미국":
        df = df_genresY_us
    elif nation == "일본":
        df = df_genresY_jp
    elif nation == "홍콩":
        df = df_genresY_hk
    elif nation == "프랑스":
        df = df_genresY_fr

    # openYear 결측치 메우기
    year_zero = list(set(range(1971, 2021)) - set(df["openYear"].values))
    data_zero = np.zeros((len(year_zero), 23))
    data_zero[:,0] = year_zero
    df_zero = pd.DataFrame(data=data_zero, columns=df_genresY_jp.columns)
    df_zero["openYear"] = df_zero["openYear"].astype(int)
    df = pd.concat([df, df_zero], axis=0).sort_values("openYear").reset_index()

    # 국가별, 장르별 데이터 정리
    df_nationsY_time = df_nationsY.query(f"{time_init} <= openYear <= {time_fin}")
    df_genresY_ntime = df.query(f"{time_init} <= openYear <= {time_fin}")
    G_sort = df_genresY_ntime.filter(like="G_").sum().sort_values(ascending=False)
    G_sort_top = G_sort[:topN]

    if len(genres) == 0:
        G_names = list(G_sort_top.index)
        G_counts = list(G_sort_top.values)
    else:
        G_names = genres
        G_counts = G_sort.loc[genres].values

    # 해당 국가 개봉 편 수
    ax.plot(df_nationsY_time["openYear"], df_nationsY_time[f"N_{nation}"],
            c="k", lw=2, zorder=10, label="개봉 편 수")
    ax.stackplot(df_genresY_ntime["openYear"], [df_genresY_ntime[g] for g in G_names],
                 colors=[G_colors[g] for g in G_names], ec="w", lw=0.5, zorder=5,
                 labels=[f"{g.split('_')[1]}: {int(c)}" for g, c in zip(G_names, G_counts)])
    ax.set_xlim(times)
    time_delta = int((time_fin - time_init)/4)
    xticks = list(range(time_init, time_fin+time_delta, time_delta))
    ax.set_xticks(xticks)
    ax.set_ylim(0, )
    ax.grid(False)
    for x in xticks:
        ax.axvline(x, c="lightgray", ls=":")
    ax.spines[["top", "right"]].set_visible(False)
    ax.set_title(nation, fontdict=font_title)
    handles, labels = ax.get_legend_handles_labels()
    ax.legend(handles=handles[1:], labels= labels[1:], fontsize="small",
            #   title=nation, title_fontsize="small",
              loc="upper left", bbox_to_anchor=(1, 1), ncol=1)
```

```
    return handles, labels


def plot_time_GN(time_init, time_fin, topN=3):
    fig, axs = plt.subplots(nrows=5, figsize=(8, 10),
                            gridspec_kw={"hspace":0.1},
                            constrained_layout=True, sharex=True)
    for N, ax in zip(N_top5, axs):
        handles, labels = plot_time_GN_(N, (time_init, time_fin), ax, topN=topN)

    # fig.legend(handles=handles[:1], labels=["개봉작 (편)"],
    #            loc="upper right", fontsize="small")
    fig.align_ylabels(axs)
    fig.suptitle(f"주요 5개국 장르별 개봉작 수 (편, {time_init}-{time_fin})₩n", fontweight="b
old")
    fig.savefig(f"./images/genres_nations_{time_init}-{time_fin}.png", dpi=200)
```

3.7.2. 1971-1987

In [ ]:

```
plot_time_GN(1971, 1987)
```


주요 5개국 장르별 개봉작 수 (편, 1971-1987)
한국
100
0
드라마: 881
멜로/로맨스: 547
액션: 332
미국
50
0
드라마: 197
액션: 136
코미디: 56
일본
2
0
액션: 2
다큐멘터리: 1
드라마: 1
홍콩
20
0
액션: 59
코미디: 20
드라마: 15
프랑스
10
0
드라마: 30
범죄: 16
액션: 12
1971
1975
1979
1983
1987

In [ ]:

```
# 한국 영화 목록 (드라마)
df_genres.query("1971 <= openYear <= 1987").loc[df_nations["N_한국"]==1].query("G_드라마 == 1")[
"movieNm"].values
```

Out[ ]:

```
array(['변강쇠', '애마부인', '쥐띠부인', '앵무새 몸으로 울었다', '뻐꾸기도 밤에 우는가', '고래사냥',
       '옹기골 뽕녀', '젊은 밤 후회없다', '87 맨발의 청춘', '얄숙이들의 개성시대', '적도의 꽃',
       '내 마음의 풍차', '안개 기둥', '매일 죽는 남자', '바람불어 좋은 날', '삼포 가는 길', '쇠사슬을 끊어라',
       '바다의 사자들', '최후의 증인', '별들의 고향 ', '장사의 꿈', '짝코', '바보 선언', '금욕', '씨받이',
       '흐르는 강물을 어찌 막으랴', '부초', '아스팔트 위의 여자', '오계', '여기자 20년', '반금련',
       '어둠의 자식들', '과부춤', '실록 김두한', '사랑의 스잔나', '왕십리', "자유부인 '81",
       '겨울로 가는 마차', '마지막 찻잔', '길소뜸', '꼬방동네 사람들', '깊고 푸른 밤', '영웅연가',
       '먼 여행 긴 터널', '이브의 건넌방', '불행한 여자의 행복', '안녕하세요 하나님', '모정',
       '슬픔은 이제 그만', '사랑할 때와 죽을 때', '어미', '쌍둥이 꼬마신랑', '밤에도 뜨는 태양',
       '잊지 못할 모정', '내시의 아내', '장대를 잡은 여자', '탄야', '화조', '삐에로와 국화', '몸부림',
       '어느 여대생의 고백', '색깔있는 여자', '마음은 외로운 사냥꾼', '하와의 행방', '3일낮 3일밤', '이방인',
       '사람의 아들', '분례기', '만다라', '티켓', '나비 품에서 울었다', '안개마을', '우상의 눈물', '고백',
       '무녀도', '뽕', '낙동강은 흐르는가', '맨발의 눈길', '아내들의 행진', '왜 그랬던가', '증언',
       '둘째 어머니', '원한의 거리에 눈이 내린다', '13세 소년', '한강', '여수 407호', '속 여수 407호',
       '파계', '살인나비를 쫓는 여자', '느미', "화녀 '82", '자유처녀', '바보사냥', '육식동물',
       '영자의 전성시대', "미워도 다시 한번 '80", '난장이가 쏘아올린 작은 공', '야훼의 딸', '바람타는 남자',
       '정부', '슬픈계절에 만나요', '우요일', '지붕위의 남자', '호랑이 아줌마', '고교 깡돌이', '사대독자',
       '축 총각졸업', '아리송해', '아니 벌써', '난 모르겠네', '여자는 괴로워', '청춘을 뜨겁게',
       '82 바보들의 청춘', '여자가 더 좋아', '팔도주방장', '장남', '피막', '초분', '경찰관', '화녀촌',
       '물보라', "빙점 '81", '청춘 교사', '유관순', '별이 빛나는 밤에', '영광의 9회말',
       '기쁜 우리 젊은 날', '낮은데로 임하소서', '겨울나그네', '여왕벌', '땡볕', '족보', '불꽃',
       '도시에서 우는 매미', '깜보', '철인들', '오싱', '얄개행진곡', '불의 딸', '중광의 허튼소리',
       '심판자', '리칭의 여선생', '내가 버린 남자', '갯마을', '돛대도 아니달고', '우산속의 세여자', '애인',
       '당신만을 사랑해', '충녀', '산딸기 3', '산딸기 2', '이장호의 외인구단', '만추', '12인의 하숙생',
       '흙', '청녀', '들국화는 피었는데', '태양닮은 소녀', '겨울여자', '속 병태와 영자', '병태와 영자',
       '사랑의 조건', '진짜 진짜 좋아해', '진짜 진짜 잊지마', '진짜 진짜 미안해', '사망탑', '야시',
       '미워도 다시 한번(대완결편)', '웃음소리', '불', '0양의 아파트', '김두한과 서대문 1번지',
       '내사랑 짱구', '혼혈아 쥬리', '여자와 비', '가을비 우산속에', '죽음보다 깊은 잠', '꽃순이를 아시나요',
       '고교얄개', '목마와 숙녀', '강변부인', '춘희', '사랑의 찬가', '을화', '저하늘에도 슬픔이',
       '초대받은 사람들', '저 높은 곳을 향하여', '위기의 여자', '아름다운 팔도강
```

산', '은하전설 테라',
'꽃띠 여자', '갑자기 불꽃처럼', '가시를 삼킨 장미', '깃발없는 기수', '꽃밭에 나비',
'그 사랑 한이 되어', '꼭지 꼭지', '괴초 도사', '겨울사랑', '그여자 사람잡네', '그때 그 사람',
'들개', '마음은 짚시', '고래사냥2', '길고 깊은 입맞춤', '그것은 밤에 이뤄졌다', '고추밭에 양배추',
'깊고 깊은 그 곳에', '그대 있어야 할 자리에', '깊은 숲속 옹달샘', '그때 죽어도 좋았다', '야행',
'화려한 유혹', '화려한 외출', '화녀도정', '호기심', '가까이 더 가까이', '가슴깊게 화끈하게',
'가을을 남기고 간 사랑', '가을장미', '각설이 품바타령', '갈채', '개구장이 천사들', '거리의 악사',
'겨울사냥', '겨울여자 2부', '경아의 사생활', '고래섬 소동', '고속도로', '공중에 뜬 나의 맨발',
'과부 3대', '광동관 소화자', '구사일생', '그 어둠에 사랑이', '그대앞에다시 서리라',
'그들은 태양을 쏘았다', '금강선법', '김마리라는 부인', '깨소금과 옥떨메', '꽃잎이어라 낙엽이어라', '꿀맛',
'태', '단', '국제 경찰', '관계', '과부', '신의 아들', '초연', '창밖의 여자', '밤의 열기속으로',
'졸업여행', '이십육×365=0 (속)', '유혹', '유부녀', '낮과 밤', '오빠', '산중전기',
'사문의 승객', '혈육마방', '외팔이 권왕', '천하 제일권', '쌍용비객', '신 당산대형', '엄마결혼식',
'엄마결혼식', '아빠 안녕', '심장이 뛰네', '속 형', '7인의 말괄량이', '연인들', '외박',
'작은 꿈이 꽃필 때', '소명', '비련의 벙어리 삼룡', '작별', '아리랑', '해결사', '금지된 사랑',
'가슴을 펴라', '조용한 방', '석화촌', '비정지대', '흑야', '광화문통 아이', '내마음 나도몰라',
'너는 달 나는 해', '선생님 안녕', '내사랑 에레나', '이런 마음 처음이야', '성난능금', '학도의용군',
'청춘공화국', '독수리 전선', '용사왕', '주고 싶은 마음', '가위 바위 보', '사랑의 계절', '네가 좋아',
'말해버릴까', '신혼소동', '어머니', '우리에게 내일은 있다', '수제자', '영노', '방범대원 용팔이',
'내 갈 길을 묻지마', '배덕자', '대련의 해당화', '맨발의 억순이', '짝', '야성의 숲', '울면 바보야',
'돌아온 팔도강산', '용서받은 여인', '사나이 악수', '밀명객', '젊은 도시', '검은 띠의 후계자',
'빗 속의 여인들', '제7교실', '별하나 나하나', '정말 꿈이 있다구', '너무 너무 좋은 거야',
'위험한 여자', '졸업생', '여수 대탈옥', '대탈출', '아메리카 방문객', '여고시절', '원산공작',
"미워도 다시 한번 '80 제2부", '옥중녀', '소녀의 기도', '여자들만 사는 거리', '원무', '딸 삼형제',
'천의 얼굴', '왕룡', '그래 그래 오늘은 안녕', '불꺼진 창', '청춘을 이야기합시다', '보통여자',
'용비', '10년만의 외출', '황토', '병태의 감격시대', '탈출', '밀행', '소', '욕망',
'여고 졸업반', '새벽에 온 방문객', '인간 단지', '빛은 어디에', '타인의 숨결', '극락조', '작은 별',
'천하무적', '5천리 대도망', '70인의 여죄수', '어린 시절', '묘녀', '이중섭', '국회 푸락치',
'멋진 사나이들', '야녀', '토지', '여자 정신대', '진아의 편지', '애정이 꽃피는 계절', '꽃상여',
'속 눈물의 웨딩드레스', '토요일 밤에', '갈매기의 꿈',
'눈으로 묻고 얼굴로 대답하고 마음 속 가득히 사랑은 영원히', '하얀 수염', '암살 지령', '그대의 찬손',
'서로 좋아해', '신설', '나는 살아야 한다', '행운', '철면객', '소띠 아저씨',

'죽어서 말하는 여인',
'별난 장군과 팔도부하', '아빠의 이름은', '특별수사본부 배태옥 사건', '마음은 푸른 하늘', '캐서린의 탈출',
'야간 비행', '수선화', '비바리', '광복 20년과 백범 김구', '두 형제', '육군사관학교',
'황소 타고 시집 왔네', '두 사나이', '체포령', '특공외인부대', '멋진 인생', '항구의 등불', '해벽',
'판사부인', '우리의 팔도강산', '인생 우등생', '사랑하는 아들의 심판', '며느리', '목소리',
'경복궁의 여인들', '갑돌이와 갑순이', '나에게 조건은 없다', '지금은 남이지만', '낙도의 무지개',
'한 많은 두 여인', '속 두아들', '족보', '간다고 잊을소냐', '유쾌한 딸 7형제', '흐느끼는 두 여인',
'들개', '월남에서 돌아온 김상사', '두 남자', '떡국', '내일의 팔도강산', '말썽난 총각',
'엄마없는 하늘아래2', '소림통천문', '추하 내사랑', '나비소녀', '코메리칸의 낮과 밤', '내가 버린 여자',
'미스 양의 모험', '악어의 공포', '여고얄개', '돌아와요 부산항', '제3부두 고슴도치', '쌍면객',
'고교 유단자', '풍협과객', '표적', '삼인호객', '아내에게 바치는 노래', '우리들 세계', '불타는 소녀',
'별 3형제', '남궁동자', '산불', '쌍무지개 뜨는언덕', '소문난 고교생', '자! 지금부터야', '처녀의 성',
'둘빼기 셋', '엄마 없는 하늘 아래', '괴짜만세', '설국', '고교 꺼꾸리군 장다리군', '고교 우량아',
'첫눈이 내릴때', '이 다음에 우리는', '황금의 팔', '여수', '신권비객', '호국팔만대장경', '낙조',
'오빠가 있다', '오룡대협', '사랑과 죽음의 기록', '소림관문돌파', '절정', '상처', '병아리들의 잔칫날',
'너의 창에 불이 꺼지고', '소리치는 깃발', '철새들의 축제', '마지막 겨울', '하늘아래 슬픔이',
'십이대천왕', '고교고단자', '오륙도 이무기', '에너지 선생', '고교명랑교실', '관세음보살', '파천신권',
'개선문', '영아의 고백', '오대제자', '체험', '우리들의 고교시대', '청춘의 문', '정무신권',
'비목(碑木)', '너는 내 운명', '밤의 찬가', '모모는 철부지', '타인의 방', '태양을 훔친 여자',
'여자이기 때문에', '독신녀', '제3한강교', '순자야', '심봤다', '아침에 퇴근하는 여자',
'터질듯한 이 가슴을', '30일간의 야유회', '밤이면 내리는 비', '누가 이 아픔을', '나팔수', '청춘의 덫',
'너는 여자 나는 남자', '땅콩껍질속의 연가', '사랑이 깊어질 때', '0양의 아파트 2', '석양의 십번가',
'비색', '도시의 사냥꾼', '목마 위의 여자', '돌의 초상', '말띠 며느리', '맨발의 청춘',
'학을 그리는 여인', '물도리 동', '이십육×삼백육십오=0', '나녀', '0녀', '내일 또 내일',
'로맨스 그레이', '우리는 밤차를 탔읍니다', '일소일권', '최인호의 병태 만세', '요', '별명 붙은 사나이',
'춘자는 못말려', '두 여인', '여호신', '메아리', '이 세상 다 준다 해도', '야성의 처녀', '소림용문방',
'여자의 이 아픔을', '멋대로 해라', '조용히 살고 싶다', '불새', '외인들', '하얀 미소',
'하늘이 부를때까지', '땅울림', '미워할 수 없는 너', '아픈 성숙', '아낌없이 바쳤는데',
'화요일 밤의 여자', '복부인', '탈명비주', '해뜨는 집', '나를 보러와요', '달려라 풍선',
'바다로 간 목마', '마지막 밀애', '머저리들의 긴 겨울', '여자의 방', '휴가받은 여자',
'내가 버린 여자 2', '낯선 곳에서 하룻밤', '잊어야 할 그 사람', '오사까의 외로운 별', '화려한 경험',

'두아들 ', '어둠의 딸들', '대학얄개', '바람, 바람, 바람', '북치는 여자', '사랑하는 사람아',
'인간시장, 작은 악마 스물두살의 자서전', '물레방아', '몸 전체로 사랑을', '후궁한', '흑성마왕과 슈퍼왕자',
'해저탐험대 마린X', '그 해 겨울은 따뜻했네', '나는 할렐루야 아줌마였다', '여자, 여자', '빨간 앵두 2',
'열아홉살 쌩머리', '피조개 뭍에 오르다', '색깔있는 남자', '밤을 벗기는 독장미',
'인간시장2 - 불타는 욕망', '난 이렇게 산다우', '애마부인 3', '홍도경', '밤을 먹고 사는 여인',
'안녕 도오쿄', '돌아이', '대학별곡', '여자 정신대', '서울에서 마지막 탱고', '창밖에 잠수교가 보인다',
'뼈와 살이 타는 밤', 'W의 비극', '부나비는 황혼이 슬프다', '아가씨와 사관', '미스 김', '아가다',
'작년에 왔던 각설이', '불의 회상', '춤추는 청춘 대학', '추억의 빛', '여자의 성',
'훔친 사과가 맛이 있다', '이별없는 아침', '사슴사냥', '여신의 늪', '차이나 타운', '반노2', '철부지',
'먹다버린 능금', '달빛 멜로디', '사랑하는 사람아 3부', '밤마다 천국', '바람난 도시', '외계우뢰용',
'은하에서 온 별똥동자 1', '미미와 철수의 청춘스케치', '얼굴이 아니고 마음입니다', '타인의 둥지',
'키위새의 겨울', '나는 다시 살고 싶다', '난중일기', '날개달린 녀석들', '날마다 허물벗는 꽃뱀',
'남과 북', '납자루떼', '내 모든 것 빼앗겨도', '내 이름은 마야', '내 인생은 나의 것',
'내가 마지막 본 흥남', '내가 사랑했다', '내이름 쌍다리', '내일은 뭐 할거니', '내일은 야구왕',
'너무합니다', '너와 나의 비밀일기', '노상에서', '눈짓에서 몸짓까지', '늪에서 늪으로',
'다른 시간 다른 장소', '달빛 사냥꾼', '달빛타기', '당신은 나쁜 사람', '대학괴짜들', '대학들개',
'대학신입생 오달자의 봄', '덫 ', '도시로 간 처녀', '돌쇠바람', '돌아온 용팔이', '돌아이 2',
'돌아이 3', '동녀', '동반자', '됴화', '두 여자의 집', '따귀일곱대', '땜장이 아내',
'뛰는자 나는자', '뜨겁고 깊은 겨울', '뜸부기 새벽에 날다', '레테의 연가', '로보트왕 썬샤크',
'마계의 딸', '마카리안고', '만삭', '말괄량이 대행진', '목없는 여살인마', '못다부른 노래님', '몽녀한',
'몽마르트 언덕의 상투', '무녀의 밤', '무릎과 무릎사이', '무림걸식도사', '물망초', '미움의 세월',
'미워 미워 미워', '밀림의 대탈출', '밀명마상객', '바람부는 날에도 꽃은 피고', '바보스러운 여자', '반노',
'밤나비', '밤을 기다리는 해바라기', '밤의 요정', '밤의 천국', '밤이 무너질 때', '방황하는 별들',
'백구야 훨훨 날지마라', '버려진 청춘', '벽과 벽사이에', '별들의 고향 3', '복수는 내게 맡겨라',
'본전생각', '불바람', '불새의 늪', '불씨', '불춤', '불타는 신록', '비내리는 영동교', '비련',
'비창', '빙해', '빨간 하이힐', '빨간앵두', '사랑 그리고 이별', '사랑만들기', '사랑의 노예',
'사랑의 미로', '사랑하는 사람아2/사랑하는 사람아(속)', '사랑하는 자식들아', '사랑할 때와 헤어질 때',
'사룡사', '사향마곡', '사형사제', '사후세계', '산동 반점', '산동여자 물장수', '삼원녀',
'삿갓 쓴 장고', '새댁 나팔을 불기 시작했다', '서울손자병법', '서울의 탱고', '서울흐림 한때 비',
'설마/설마가 사람잡네', '성 리수일뎐', '세번은 짧게 세번은 길게', '소림사 주천귀동', '소림신방',
'속 변강쇠', '송골매의 모두 다 사랑하리', '수렁에서 건진 내 딸', '수렁에서

```
건진 내 딸 2',
       '술잔과 입술', '숲속의 바보', '스무해 첫째날', '스물 하나의 비망록', '스잔
나의 체험', '스타페리 불청객',
       '신입사원 얄개', '실연클럽', '심형래의 탐정큐', '십팔(18)통문방', '쌍배',
'아가씨 참으세요', '아내',
       '아벤고 공수군단', '악녀의 밀실', '안개는 여자처럼 속삭인다', '암사슴', '애
마부인 2',
       '야누스의 불꽃여자', '야망과 도전', '야생마', '약속한 여자', '어울렁 더울
렁', '엄마는 외출중', '엑스',
       '엘리베이터 올라타기', '여로', '여애권', '여자가 두번 화장할 때', '여자가
밤을 두려워 하랴',
       '여자가 울린 남자', '여자는 남자를 쏘았다', '여자는 비처럼 남자를 적신다',
'여자는 한번 승부한다',
       '여자의 대지에 비를 내려라', '여자의 반란', '여자의 함정', '연분홍치마',
'연인들의 이야기', '열병시대',
       '열아홉살의 가을', '열애', '영자의 전성시대 (속)', '영점구일칠', '오늘만은
참으세요',
       '오늘밤은 참으세요', '오마담의 외출', '오염된 자식들', '왜불러', '외출',
'요색유희', '욕망의 거리',
       '욕망의 늪', '용호의 사촌들', '울지않는 호랑이', '웅담부인', '웬일이니',
'유리의 성', '유정',
       '유혹시대', '이 깊은밤의 포옹', '이 한몸 돌이 되어', '이런 여자 없나요',
'이름없는 여자',
       '이상한 관계', '이혼 법정', '인사대전', '인사야투', '인생극장', '일송정 푸
른 솔은', '입을 연 석류',
       '잊혀진 계절', '자유부인 2', '작은 고추', '작은 사랑의 노래', '장미부인',
'장미와 도박사',
       '저녁에 우는 새', '젊은 시계탑', "정무문 '81", '정염의 갈매기/정념의 갈매
기', '젖은풀 젖은잎',
       '제2의 사춘기', '종군수첩', '종점', '죽으면 살리라', '지금 이대로가 좋아',
'지옥의 링',
       '진아의 벌레먹은 장미', '질투', '집을 나온 여인', '짧은 포옹 긴 이별', '차
라리 불덩이가 되리',
       '참새와 허수아비', '첫사랑은 못잊어', '청(블루스케치)', '청개구리들/5학년 3
반 청개구리들',
       '초대받은 성웅들', '초야에 타는 강', '최인호의 야색', '추적', '춤추는 달팽
이', '춤추는 딸',
       '친구 애인', '친구여 조용히 가다오', '캔디 캔디', '탄드라의 불', '토요일은
밤이 없다', '평양 박치기',
       '푸른계절의 열기', '푸른하늘 은하수', '풀잎처럼 눕다', '풍녀', '하녀의 방',
'하늘가는 밝은 길',
       '하늘나라 엄마별이', '학창보고서', '한쪽날개의 천사', '형', '홍병매', '화려
한 변신', '화순이',
       '화야', '화평의 길', '흑녀', '흑표비객', '英雄 돌아오다', '(무진) 흐린뒤 안
개',
       '(속) 여자가 밤을 두려워 하랴', '0번 아가씨', '0시의 호텔', '0점하의 자식
들', '13월의 연정',
       '2월30일생', '3색스캔들', '77단의 비밀', '87 영자의 전성시대', '88짝궁들',
'F학점의 천재들',
       'J에게', 'LA용팔이', 'Y의 체험'], dtype=object)
```

In [ ]:

```
# 미국 영화 목록 (액션)
df_genres.query("1971 <= openYear <= 1987").loc[df_nations["N_미국"]==1].query("G_액션 == 1")["movieNm"].values
```

Out[ ]:

```
array(['람보 ', '에이리언 2 ', '스트리트 오브 화이어', '스타워즈', '벤허', '탑건',
'데미트리아스',
       '군용열차', '사망유희', '스팔타카스', '타겟트', '페세이지', '최후의 총잡
이', '프렌치 커넥션', '록키',
       '로보캅', '매드맥스', '아우트 로', '캐논볼', '원스 어폰 어 타임 인 아메리
카', '더티 해리',
       '돌아오지 않는 강', '스카페이스 ', '터미네이터 ', '007 유어 아이스 온리',
'007 네버세이 네버어게인',
       '람보 2', '비도권운산', '젊은이의 양지', '타워링', '카산드라 크로스', '대장
부리바', '겟어웨이',
       '원 웨이 티켓', '고릴라 ', '킹 솔로몬 ', '캐논볼 2 ', '실버스트릭', '런어웨
이 ', '나바론 요새',
       '노머시', '신디케이트', '추방객', '유황도의 모래', '브레이크 아웃', '바람과
라이온', '매트 헌터 ',
       '도라,도라,도라', '이어 오브 드래곤 ', '7인의 독수리', '신밧드 대모험 호랑
이 눈알', '미드웨이',
       '굿바이 브루스 리', '클레오파트라 카지노정복', '지옥의 사자들', '슈퍼맨',
'더티 해리3-집행자', '스웜',
       '킹콩', '옥타곤', '스티브 맥퀸의 헌터', '쟈글라', '500만 달러의 모험', '쇼
군', '솔저',
       '알카포네', '써튼 퓨리', '메카닉', '전쟁이여 잘 있거라', '샤머스', '샤프
트', '크레므린 레터',
       '레드 선', '쟝글왕 타잔', '매드매드 대소동', '마제스틱', '솔저 블루', '4인
의 무뢰한', '엘 콘도',
       '전격후린트 특공작전', '봐이킹', '델타포스 ', '총알을 물어라', '해저세계일
주', '삼총사', '초인 사베지',
       '에어포오드 75', '스콜피오', '40인의 여도적', '이것이 법이다', '워킹톨',
       '007 제8편 : 죽느냐 사느냐', '용쟁호투', '나이트 호크', '대특명/부르다크',
'더블보더 ',
       '더티해리 4-써든 임팩트/서든 임팩트', '돌아온 터미네이터 ', '레모 ', '레이
더스', '레인저',
       '로맨싱 스톤 ', '롱 라이더스', '리쎌웨폰-인간병기', '매드맥스3', '베스트 키
드 ', '브레드레스 ',
       '블랙이글 ', '블루 썬더 ', '비버리힐스 캅', '비스마스터', '빅 트러블 ', '샤
키 머신', '쉬 ',
       '슈퍼맨 2', '스워드', '실버라도 ', '싸이렌스 ', '쓰리아미고 ', '어게인스트
', '에프 엑스 ',
       '엑스칼리버', '인디아나 존스 - 마궁의 사원, 저주받은 사원 ', '지옥의 7인 ',
'코난-바바리안',
       '코난2-디스트로이어', '코만도 ', '코브라 ', '폴리스 아카데미 긴급출동', '폴
리스아카데미 3-재훈련',
       '프레데터 ', '화이어폭스', '007 문레이커', '007 제13탄 - 옥토퍼시',
       '007 제14탄 - 뷰 투 어 킬', '48시간 '], dtype=object)
```

In [ ]:

```
# 미국 영화 목록 (코미디)
df_genres.query("1971 <= openYear <= 1987").loc[df_nations["N_미국"]==1].query("G_코미디 == 1")[
"movieNm"].values
```

Out[ ]:

```
array(['빽 투 더 퓨쳐', '졸업', '메리 포핀스', '당신에게 오늘 밤을', '딸라', '싼타
빅토리아의 비밀',
       '고스트바스터즈', '캐논볼', '블루스 브라더스', '애정의 조건 ', '7인의 신
부', '가브린', '구니스 ',
       '그렘린 ', '캐논볼 2 ', '실버스트릭', '스팅', '맛슈', '핑크 팬더', '위험한
여로', '럭키 레이디',
       '정오에서 3시까지', '키스미', '1941', '브레이킹 어웨이', '파울플레이', '메
인 이벤트', '썬번',
       '500만 달러의 모험', '프라이비트 레슨:개인교수', '캣트로', '털보 대소동',
'백작부인',
       '매드매드 대소동', '전격후린트 특공작전', '애로', '대탈옥', '솔티', '애인관
계', '후리비의 대난전',
       '연애대소동', '나인 투 화이브', '로맨싱 스톤 ', '마이튜터 ', '비버리힐스
캅', '빅 트러블 ',
       '쓰리아미고 ', '인디아나 존스 - 마궁의 사원, 저주받은 사원 ', '쿼터매인 ',
'텐', '투씨',
       '폴리스 아카데미 ', '폴리스아카데미 3-재훈련', '프라이빗 스쿨', '프리찌스
오너 ', '48시간 '],
      dtype=object)
```

In [ ]:

```
# 일본 영화 목록 (전체)
df_genres.query("1971 <= openYear <= 1987").loc[df_nations["N_일본"]==1]
```

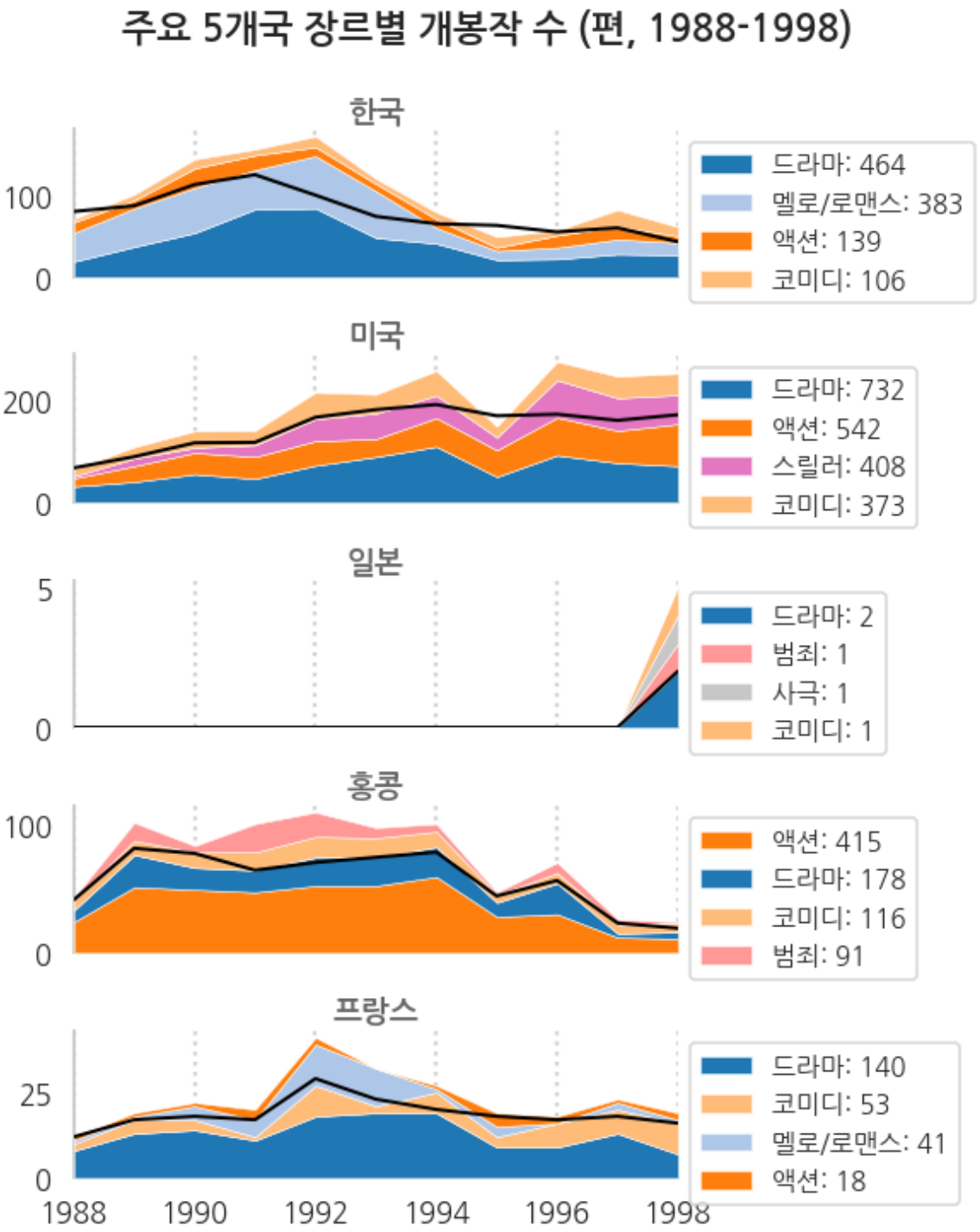
Out[ ]:

	movieCd	movieNm	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬	G_미스터리	G_범죄
18068	20108135	만국박람회	1971	0	0	0	0	0	1	1	0	0	0	0
19518	19790102	무적 가라데	1979	0	0	0	0	0	0	0	0	0	0	0
24185	19810054	격투왕	1981	0	0	0	0	0	0	0	0	0	0	0

3.7.3. 1988-1998

In [ ]:

```
plot_time_GN(1988, 1998, topN=4)
```



In [ ]:

```
# 미국 영화 목록 (드라마)
df_genres.query("1988 <= openYear <= 1998").loc[df_nations["N_미국"]==1].query("G_드라마 == 1")[
"movieNm"].values
```

Out[ ]:

```
array(['흐르는 강물처럼', '내 남자친구의 결혼식', '이보다 더 좋을 순 없다 ', '파리
의 여인', '죽은 시인의 사회',
       '늑대와 춤을', '시티 라이트', '써커스', '살인광시대', '라임라이트', '쇼걸',
'지옥의 묵시록',
       '불멸의 연인 ', '마돈나 진실 혹은 대담', '재키 브라운 ', '저수지의 개들 ',
'아름다운 비행 ',
       '프리 윌리 ', '알라딘', '의적 로빈후드', '쉰들러 리스트', '클리프 행어',
'비엔나 호텔의 야간배달부',
       '머나먼 여정', '트루먼 쇼', '보디가드', '드라이빙 미스데이지', '사랑의 기
적', '구름 속의 산책 ',
       '비포 선라이즈', '더티댄싱', '에비타 ', '환생', '칼리토 ', '실베스타스텔론
의 탈옥',
       '라스트맨 스탠딩 ', '에너미 오브 스테이트', '블레이드 러너: 파이널 컷', '데
이라잇', '태양의 제국',
       '벅시', '타이타닉', '콘돌', '굿 윌 헌팅', '사랑과 영혼', '포레스트 검프',
'사랑의 행로', '히트',
       '미녀와 야수', '라이온 킹', '닉슨', '사랑의 용기', '미스틱 피자', '빅 나이
트', 'K2',
       '트윈 픽스 ', '로스트 하이웨이 ', '데드 맨', '델마와 루이스 ', '천국보다 낯
선',
       '시애틀의 잠 못이루는 밤', '제리 맥과이어', '해리가 샐리를 만났을 때', '블
루 벨벳', '잉글리쉬 페이션트',
       '여인의 향기', '써머스비', '후라이드 그린 토마토', '로렌조 오일', '어느 멋
진날', '순수의 시대 ',
       '커리지 언더 파이어 ', '센스 앤 센서빌리티 ', '위험한 정사', '케이프 피어',
'코쿤', '쇼생크 탈출',
       '길버트 그레이프', '로리타', '흑우: 블랙레인', '로미오와 줄리엣', '가위손',
'알비노 앨리게이터',
       '헤븐스 프리즈너', '미키 루크의 F.T.W', '언터처블', '레인맨', '빅', '퍼블릭
에너미 NO 1',
       '죽음의 항로-그레이건맨', '필링 미네소타', '하베스트', '닌자 거북이', '굿바
이 마이 프렌드', '걸 식스',
       '꿈이 지나간 자리', '섹스 거짓말 그리고 비디오테이프', '마젠타의 숲', '에어
포스 원',
       '네 이웃의 아내를 탐하지마라', '불사신', '코코 뉴욕', '가정교사', '4월의 유
혹', '펜트하우스의 칼리귤라',
       '시티 오브 엔젤', '글라스 케이지', '아이스 스톰', '사랑의 블랙홀', '글렌게
리 글렌로스', '그리프터스',
       '고스트 앤 다크니스 ', '겟어웨이 - 끝없는 도주', '광란의 정사', '모스키토
코스트',
       '깊은밤 깊은 곳에 2', '그들만의 리그', '꼴레뜨', '글라디에이터', '가라데 키
드 ', '귀여운 악마',
       '골든 게이트 ', '글리머 맨 ', '꼬마천재 테이트', '길티', '고티 ', '겟 쇼티
', '애정의 숲',
       '그 아버지에 그 아들', '꿈의 구장', '광란의 시간', '흔들리는 영웅', '화이어
다운 ', '홈보이',
       '헨리의 이야기', '헤븐스 티어', '햄 스트리트의 육일간', '하바나', '프랜서',
'패리스 트라웃 ',
       '트루 로맨스 ', '투게더', '귀여운 여인', '그리고 신은 여자를 창조했다', '금
지된 춤', '터커',
       '광란의 사랑', '공룡시대', '아더는 영혼을 살해했다', 'L.A 여신 ', '여자가
사랑을 훔칠 때 ',
       '스나이퍼', '선체이서', '체인 리액션 ', '천국으로 가는 장의사', '백야의 연
인', '머홀랜드 폴스 ',
       '마담 엠마', '라스트 지고로', '드래곤', '붉은 전사', '펜타트론 ', '은밀한
게임', '블루라군',
       '엠마 ', '고인돌 가족 프린스톤 ', '에디', 'LA컨피덴셜', '컨스피러시 ', '왝
더 독 ', '쿨 러닝 ',
       '사이공', '폭풍의 질주', '육체의 문 ', '워킹걸', '브룩 쉴즈의 욕망의 거리',
```

'크래쉬',
'이스트 L.A 스토리', '집행자', '원나잇 스탠드 ', '스위트 룸 ', '비밀의 화원', '망각의 삶 ',
'론리 하트', '러브레터', '러브 렛슨 ', '악의 꽃 ', '해리슨 포드의 도망자', '포인트 블랭크',
'업클로즈 앤 퍼스널 ', '가타카', '히트맨', '나나', '질리안의 서른일곱번째 생일에', '쥬드 ',
'쥬다스 ', '좋은 친구들', '조이 럭 클럽 ', '이어 오브 더 코맷', '완전한 세상 ', '얼라이브',
'어둠속의 외침', '피터 웰라의 분노의 선택', '피고인', '프릭스 대모험 ', '프리덤 스트라이크',
'프라이버시 ', '프라이멀 피어 ', '펠리칸 브리프 ', '파리대왕', '티벳에서의7년 ', '크라임서치 ',
'섀터드 이미지', '찰리 채플린', '로드 앤드', '레더 쟈켓', '디 엣지 ', '돌로레스 클레이븐 ',
'F학점의 챔피언', '12 몽키즈', '보디 더블', '프리처스와이프', '줄리아 로버츠의 사랑게임', '불사조',
'택시 드라이버', '플레이 백 ', '디아볼릭 ', '넬', '파라다이스로드', '슬리퍼스 ', '캠퍼스 군단',
'조니 뎁의 돈 쥬앙 ', '영원한 사랑', '아이언 마스크 ', '아버지의 이름으로', '로 앤 크라임',
'부룩크린으로 가는 마지막 비상구', '헨리 밀러의 북회귀선 ', '미저리', '미드나잇 익스프레스 ', '머큐리 ',
'머더1600 ', '맬리시우스', '리얼 맥코이 ', '리벤지', '롱 키스 굿나잇 ', '로드 하우스',
'레이브 댄싱 ', '러시아 하우스', '라스베가스를 떠나며 ', '노웨이 아웃', '남자가 사랑할 때 ',
'남자가 여자를 사랑할 때', '남아있는 나날 ', '1492 콜럼버스', '쎄븐싸인', '미세스 다웃파이어 ',
'배신의 욕망', '마이키 이야기 3', '월트디즈니 삼총사 ', '레이지', '카드로 만든 집', '마더스 보이',
'밤 그리고 도시 ', '사선에서', '존 트라볼타의 황금사슬', '어져스터', '나이트 비지터',
'메이드 인 아메리카', '엄마의 노래', '은밀한 유혹', '이노쎈스', '트리플 임팩트', '사라피나',
'씨스터 액트', '죽음의 뺏지', '씨에스타', '용서받지 못한 자', '리즈', '살인추적', '리틀 빅 히어로',
'더 베스트', '라이브 와이어', '짐 캐리의 에이스 벤츄라 2 ', '차타레 부인', '주홍글씨 ',
'8초의 승부 ', '스모크 ', '위험한 아이들 ', '헨리와 후레디의 마지막 비상구', '라디오 인사이드 ',
'다크 댄서 ', '프리 윌리 2 ', '아폴로 13 ', '카멜롯의 전설 ', '숏컷 ', '크리스마스 인 코네티컷',
'토요일 밤의 남자', '브레이브 하트 ', '웰컴 홈', '신부는 왼손잡이', '보이즈 온 더 사이드 ',
'벤지 2 ', '파워플레이', '욕망의 교차로', '배반의 도시 ', '핸드 메이즈', '롭 로이 ',
'노스바스의 추억 ', '폴 타임 ', '더 서스피션 ', '황홀한 연인 ', '더블 헌터', '유혹의 함정 ',
'사랑의 선택', '센세이션', '블루진 캅 ', '트리플 플레이 ', '마녀 ', '킬라인', '마지막 쇼',
'드림 머신', '벨을 두 번 누른 여자 ', '불멸의 열정', '마이애미 열정 ', '비상구없는 캠퍼스',
'이너써클', '필살의 표적 ', '리틀 시스터', '빙고', '여자의 본능', '데저트 호크', '슈거힐 ',
'위험한 독신녀', '키스 굿나잇 ', '바톤 핑크', '사랑은 은반위에', '래피드 화이어', '은밀한 관계 ',
'육체의 법칙 ', '남의 돈', '빌리 배스게이트', '베이브', '작은 전쟁 ', '이면의 정사 ',
'에로스의 빨간 하이힐 ', '사랑의 본능', '헌팅', '패트리어트 게임', '뱃 인플루언스', '싸이렌서',

'아웃브레이크 ', '라스트 템테이션 ', '작은 아씨들 ', '위험한 유혹', '돈텔 맘', '프랭키와 쟈니',
'파워 오브 원', '스킨스 ', '퀴즈 쇼 ', '마이걸', '차이나 크라이', '브레이크업 ', '렛 잇 비 미',
'이집트 왕자', '조블랙의 사랑', '비공개', '사랑과 전쟁2', '외야의 천사들 ', '34번가의 기적 ',
'뱀파이어와의 인터뷰 ', '프랙티컬 매직', '헤모글로빈', "라스트 홍콩 '97", '유혹의 깊은 밤 ',
'용서받지 못할 관계 2 ', '코리나 코리나 ', '아웃 오브 더 레인', '토미 리 존스의 남자의 진실 ',
'모델, 작가 그리고 애인 ', '미세스 브라운 ', '위험한 질주 ', '내 노래를 들어라 ', '쥴리안 포',
'라이트 아웃 ', '세인트 오브 뉴욕 ', '네고시에이터', '데스퍼레이트 ', '에머렌즈의 첫경험',
'스페셜리스트 ', '퍼펙트 머더', '인피니티 ', '브룩 쉴즈의 7층 ', '디제스타', '사기 방정식 ',
'밤의 연인', '바스키아', '리치 리치 ', '아티카 ', '에로틱 테일즈1', '늑대의 그림자 ',
'라이언 일병 구하기', '의뢰인 ', '부기보이', '위험한 거래 ', '해리슨 포드의 긴급명령 ', '터치',
'레인메이커 ', '르네상스 맨', '브링크 ', '마이 라이프 ', '울프 ', '단판승부 ', '사이드 킥',
'데몰리션-U', '랜드 오브 프리', '사랑의 커피 ', '3 닌자 키드', '블랙아웃', '엑스파일',
'눈가리개 블라인드 폴드', '정글북 ', '툼스톤 ', '에덴의 마지막 날 ', '마이걸 2 ',
'투 문 정션 2', '앤지 ', '빌리 크리스탈의 토요일 밤의 남자', '당신에게 일어날 수 있는 일',
'러닝 타킷', '드림 러버 ', '패릴 ', '랜섬 ', '레드 로즈 화이트 로즈 ', '스토미 나이트 ',
'씨티 홀 ', '스트레인지 데이즈 ', '타임 투 킬 ', '배반의 음모', '바이브레이션 ', '잭 ',
'더 팬 ', '싸일런트 트리거 ', '저널 오브 머더 ', '톰베린저의 백색지대 ', '에이스', '사랑과 추억',
'미스터 존스 ', '원죄의 밤', '최종분석', '사랑과 우정', '내사랑 컬리수', '하드레인 ',
'나 홀로 집에 3 ', '아마겟돈', '어딕션', '도니 브래스코 ', '카피 캣 ', '위대한 유산 ',
'케이블 가이 ', '페노메논 ', '호스 위스퍼러 ', '조지아 ', '화이널 카운트', '에릭 로버츠의 저격자',
'틴 컵 ', '플루크 ', '앙드레 ', '에스케이프 48 ', '화이트 스콜 ', '스트라이킹 디스턴스 ',
'엑스트라 마리탈', '딥 임팩트', '스탤론의 리쎌 화이어 ', '롤러보이즈', '스트립티즈 ', '이자벨 에버하트',
'코요테 ', '트위스터 ', '비욘드 랭군 ', '화이트 샌드', '미스터 플라워 ', '어퓨굿맨', '이레이저 ',
'데인저 존 ', '바운티 헌터 ', '키스 오브 데드 ', '라스트 댄스 ', '매치 포인트', '더블 크로스 ',
'머팻 크리스마스 캐롤', '미드나잇 히쳐 ', '블라인드 키스 ', '워킹 앤 토킹', '주어러 ', '파이널 컷 ',
'흑백소동 ', '디스 보이스 라이프 ', '사랑에 미쳐서', '버드케이지 ', '후드럼 ', '암흑의 질주 ',
'아워 글래스 ', '포 룸 ', '데드 이노센트 ', '사랑을 기다리며 ', '나우 앤 덴 ', '비포 앤 애프터 ',
'블랙 데이 블루 나이트 ', '스텔스 98', '홀랜드 오퍼스 ', '카지노 ', '나이트 보디가드 4 ',
'예수 그리스도 ', '미스트 라이얼', '침묵의 암살자 ', '서든 데쓰 ', '로즈 앤 그레고리 ', '아미스타드',
'레드코너 ', '위험한 여인', '풀 메탈 쟈켓', '사브리나 ', '퓨너럴 ', '이완 맥그리거의 인질',
'에어버드 ', '살인키스 ', '붉은 유혹', '스네이크 아이', '말리 매틀린의 어두

```
워질때까지', '잠복근무2',
       '맨 트러블', '나이트 아이 2', '비터문', '야망의 함정', '스니커즈', '원죄적
본능', '드라큐라',
       '너에겐 내일이 없다', '자유시대', 'LA 리치걸', '어벤저2', '퍼시픽 하이츠',
'스톤콜드',
       '전자오락의 마법사', '늑대개', '분노의 역류', '핫스텝', '죽음전의 키스',
'실패한 성공', 'F.X. 2',
       '녹색카드', '베드룸 아이스2', '엑소시스트3', '사랑,모니터 그리고 비상구',
'록키 5', '체인댄스',
       '잭 더 베어', '천사의 침묵 ', '하늘과 땅 ', 'M 버터 플라이 ', '마지막 연인
',
       '퍼스트 레이디 특수경호대 ', '스냅 드래곤 ', '욕망의 볼레로 ', '덩크슛 ',
'필라델피아 ', '파우더',
       '맬리스 ', '마티니 ', '사랑의 동반자 ', '나의 청춘 워터랜드', '카프카 ',
'마미 마켓 ',
       '용서받지 못할 관계 ', '아이가 커졌어요', '8년만의 정사', '포이틱 저스티
스', '어느 목사의 부인',
       '토이즈', '반칙게임', '호파', '말콤 X', '콜드 나이트', '써가 ', '도어즈',
'와일드 오키드 3',
       '솔로몬의 딸', '비애', '햄릿', '와일드 오키드 2', '리버 피닉스의 아이다호',
'차타레 부인 2',
       '아빠셋 엄마하나', '사랑과 슬픔의 맨하탄', '적과의 동침', '하얀궁전', '뒤로
가는 남과 여', '의혹',
       '니키와 지노', '신묵시록', '3인의 약혼녀', '죽음의 키스', '마법사 피터의 모
험',
       '에밀리 브론테의 폭풍의 언덕', '데블스에드버킷', '로잔나포에버', '라스트도
그맨 ', '48시간의킬링게임 ',
       '로빈슨크루소 ', '원죄적 본능 3 ', '덴버 ', '위험한진실', '콘택트 ', '크로
싱가드 ',
       '키스 더 걸 ', '돈크라이마미 ', '더 웰 ', '머천트데쓰 ', '어느어머니의아들
', '빌로우유토피아 ',
       '블러드앤와인', '여인의초상 ', '스틸브리딩', '차이니즈박스', '나인하프위크
2 ', '마빈스룸',
       '지아이제인', '더블섹스', '인형의 집으로 오세요 ', '온더라인 ', '크루서블',
'바디체크 ', '크래프트',
       '마틸다 ', '페이스오프', '섹슈얼 인텐트', '래리플린트 ', '어거스트킹 ', '앱
솔루트 파워 ',
       '볼케이노 ', '육체의욕망 ', '셋잇오프 ', '아메리칸 퀼트', '폴링다운 ', '스
핏파이어그릴 ',
       '크래쉬 다이브 ', '단테스피크 ', '프랭키 더 플라이 ', '보거스 ', '에너미라
인 ', '트루 벤전스 ',
       '예견된폭로 ', '댓씽유두 ', '생존게임 ', '엠파이어레코드 ', '마이클 콜린스
', '씨크릿 오브 요란다',
       '광란의 춤 람바다', '그로잉 업 2', '나디아', '나이트클럽', '데스티니', '라
밤바', '라스트 버진',
       '라스트 타켓', '레이디', '로메로', '로아', '메이드인 헤븐', '문스트럭', '뮤
직박스', '배트 21',
       '베드룸 아이스', '보디 플레이', '볼륨을 높여라', '붉은 10월', '브로드캐스트
뉴스', '블랙 위도우',
       '블러드타이', '블레임 람바다', '사랑의 파도', '살바도르', '살사댄싱', '샤
넬', '세라자드', '스캐빈져',
       '스키 아카데미', '스텔라', '스트리트 워킹', '스트립 투킬', '슬램', '시덕
션', '시실리안', '심연',
       '얼굴없는 암살자', '열정의 람바다', '와일드 오키드', '운전면허', '워 앤드
러브', '웰 스트리트',
       '위험한 연인', '유혹은 밤그림자처럼', '이디아민', '잡초', '장미의 전쟁',
'쟈니핸섬',
       '적 그리고 사랑이야기', '죽는자를 위한 기도', '줄리아 줄리아', '지젤', '차
이나 걸', '카운터포스',
       '카프리의 깊은밤', '칵테일', '콜미', '크레이지보이 러브호텔', '크로스 컨츄
리', '크리미날로',
```

```
       '타이거 월샤', '투 문 정션', '트라이 앵글', '트라이엄프', '퍼스트 러브',
'프라하의 봄', '한나스 워',
       '핫타켓', '햄버거 힐', '헐크호건의 죽느냐 사느냐', '007 살인면허', '19번째
남자', '7월 4일생'],
      dtype=object)
```

In [ ]:

```
# 미국 영화 목록 (스릴러)
df_genres.query("1988 <= openYear <= 1998").loc[df_nations["N_미국"]==1].query("G_스릴러 == 1")[
"movieNm"].values
```

Out[ ]:

```
array(['나홀로 집에', '나는 네가 지난 여름에 한 일을 알고 있다 ', '재키 브라운 ',
'저수지의 개들 ',
       '터미네이터2 3D', '양들의 침묵', '황혼에서 새벽까지', '클리프 행어', '폭풍
속으로', '환생',
       '실베스타스텔론의 탈옥', '에너미 오브 스테이트', '숀 코네리의 함정 ', '가면
의 정사', '데이라잇',
       '불같은 약속', '콘돌', '세븐', '유주얼 서스펙트', '히트', '사랑의 용기',
'로스트 하이웨이 ',
       '블루 벨벳', '위험한 정사', '케이프 피어', '다크시티', '알비노 앨리게이터',
'데블스 오운',
       '헤븐스 프리즈너', '언터처블', '누가 로저래빗을 모함했나', '퍼블릭 에너미 N
0 1', '하베스트',
       '배트맨 앤 로빈', '007 제18편 : 네버 다이', '마젠타의 숲', '에어 포스 원',
'위험한 커플',
       '애증의 심판', '딥 다운', '거울속의 정사', '퍼팩트 타겟', '하드 타겟', '패
신져 57',
       '네 이웃의 아내를 탐하지마라', '불사신', '킬링 타켓', '페이 백', '섀도우 프
로그램', '메이데이',
       '에이리언3', '더 게임', '스카 시티', '리쎌웨폰 4 ', '블러드 라인', '글라스
케이지', '스피시즈 2',
       '프라퍼시 2', '파리의 늑대인간 ', '글렌게리 글렌로스', '고스트 앤 다크니스
', '광란의 정사',
       '계부 3', '가디안', '글리머 맨 ', '겟 쇼티 ', '광란의 시간', '개목걸이',
'화이어 다운 ',
       '공포의 수학열차', '크러쉬', '퀘스트 ', '아더는 영혼을 살해했다', '위험한
증거 ', '머홀랜드 폴스 ',
       '맥시멈 리스크 ', '마담 엠마', '일루션 ', '인베이더', '펜타트론 ', '다크 앤
젤', '다락방에 핀 꽃',
       '나이트 러닝맨 ', '은밀한 게임', '론머맨', '레비아탄', '프리티 마담',
       '그래서 나는 도끼부인과 결혼했다 ', 'LA컨피덴셜', '컨스피러시 ', 'JFK', '카
메론디아즈의 섹시블루 ',
       '레드 라인', '은밀한 본능', '크래쉬', '스위트 룸 ', '해리슨 포드의 도망자',
'가타카',
       '낯선 여자와 정사', '히트맨', '제수이트 죠', '져지 앤 쥬리 ', '얼라이브',
'피고인', '프라이트너 ',
       '프라이버시 ', '프라이멀 피어 ', '파리대왕', '특명델타포스 ', '터뷸런스',
'터미날 스피드 ',
       '스트레인저', '슈퍼 트릭', '섀터드 이미지', '이벤트호라이즌 ', '디 엣지 ',
'F학점의 챔피언',
       '12 몽키즈', '택시 드라이버', '제로 타겟 ', '조지클루니의 표적', '플레이 백
', '스트레인저 ',
       '디아볼릭 ', '블랙머니 게임', '슬리퍼스 ', '펄프 픽션 ', '캠퍼스 군단', '바
운드 ', '미저리',
       '미드나잇 익스프레스 ', '머큐리 ', '머더1600 ', '맬리시우스', '리얼 맥코이
', '롱 키스 굿나잇 ',
       '로닌 ', '레릭 ', '러시아 하우스', '나이트 무브', '쎄븐싸인', '배신의 욕
망', '마더스 보이',
       '야생 선인장', '론리 하트', '사선에서', '밤의 추적자', '레베카의 약점', '존
트라볼타의 황금사슬',
       '나이트 비지터', '양들의 반란', '위험한 노출', '카인의 두 얼굴', '흔들리는
유혹', '살인추적',
       '라이브 와이어', '플레이어', '과거 속의 살인 ', '네트 ', '다크 댄서 ', '야
성녀 아이비 2 ',
       '린다 해밀턴의 이중생활 ', '하이드어웨이 ', '악마의 유희', '칼리포니아 ',
'시고니 위버의 진실 ',
       '보 데릭의 욕망의 파편 ', '더 프리시디오', '파워플레이', '매니아 캅2-코델',
'원죄적 본능 2 ',
       '배반의 도시 ', '네미시스 2 ', '아틱 블루 ', '차이나 문 ', '이중노출 ', '황
홀한 연인 ',
```

'유혹의 함정 ', '블라이가의 미로 ', '사랑의 선택', '센세이션', '트리플 플레이 ', '마녀 ',
'드림 머신', '피어 인 사이드', '벨을 두 번 누른 여자 ', '붉은 사슴비', '은밀한 충동 ',
'비상구없는 캠퍼스', '언더씨즈', '여자의 본능', '슈거힐 ', '위험한 독신녀', '키스 굿나잇 ',
'래피드 화이어', '투명인간의 사랑', '사랑의 본능', '하이더 인더 하우스', '패트리어트 게임',
'뱃 인플루언스', '싸이렌서', '일급살인 ', '스텝파더3', '위험한 유혹', '카멜레온', '육체의 행로',
'론머맨2', '원초적 본능', '패트리어트', "라스트 홍콩 '97", '용서받지 못할 관계 2 ', '분노의 폭발',
'노웨이 런', '블레이드', '빅히트', '라이트 아웃 ', '네고시에이터', '데스퍼레이트 ', '스페셜리스트 ',
'화이트 레이븐', '퍼펙트 머더', '브룩 쉴즈의 7층 ', '디제스타', '베스트 캅 ', '아티카 ',
'의뢰인 ', '크로우 ', '부기보이', '위험한 거래 ', '비버리힐스 캅 3 ', '해리슨 포드의 긴급명령 ',
'트루 라이즈 ', '레인메이커 ', '스네이크 아이즈', '맥스 3000 ', '브레인 스캔 ', '폭력교실 2 ',
'브링크 ', '울프 ', '랜드 오브 프리', '사랑의 커피 ', '스피드 ', '엑스파일', '카운터 포스 ',
'눈가리개 블라인드 폴드', '스타 파이어', '폭로 ', '러닝 타킷', '드림 러버 ', '패릴 ', '랜섬 ',
'스토미 나이트 ', '크로우 2:도시의 천사 ', '씨티 홀 ', '언더월드 ', '커버미 ',
'스트레인지 데이즈 ', '타임 투 킬 ', '배반의 음모', '플레드 ', '더 팬 ', '러브 앤 에이 45 ',
'싸일런트 트리거 ', '저널 오브 머더 ', '스필', '최후의 나이트 메어 - 프레디의 죽음',
'전화로만 끝내 주세요', '살인전담 호미싸이드', '요람을 흔드는 손', '최종분석', '하드레인 ', '위너 ',
'레트로 액티브', '도니 브래스코 ', '모스트 원티드', '카피 캣 ', '케이블 가이 ', '화이어 스톰 ',
'에릭 로버츠의 저격자', '돈 룩 백 ', '행복했던 여자', '스트립티즈 ', '더 록 ', '트위스터 ',
'싸이버 벤젠스 ', '화이트 샌드', '어퓨굿맨', '이레이저 ', '볼티지', '화려한 유혹', '스카이 러너 ',
'루시퍼', '메리 라일리 ', '도망자 2 ', '세베지', '파이어스톰 ', '더블 크로스 ',
'나 홀로 집에 2', '미드나잇 히쳐 ', '블라인드 키스 ', '워 헤드 ', '주어러 ', '파이널 컷 ',
'브로큰 애로우 ', '스피어 ', '암흑의 질주 ', '데드 이노센트 ', '비포 앤 애프터 ', '네미시스 3 ',
'와일드 게임 ', '나이트 보디가드 4 ', '일렉트라 ', '미스트 라이얼', '침묵의 암살자 ', '싸일랜서',
'레드코너 ', '할로우 포인트 ', '블러드 파이터 ', '크라잉 프리맨 ', '리플레이스먼트 킬러', '살인키스 ',
'지옥의 슬로터', '헥시나', '나이트 보디가드', '나이트 아이 2', '무단침입', '도플갱어', '야망의 함정',
'육체의 증거', '원죄적 본능', '드라큐라', '퍼시픽 하이츠', '분노의 역류', '죽음전의 키스',
'실패한 성공', '베드룸 아이스2', '엑소시스트3', '사랑,모니터 그리고 비상구', '야곱의 사다리',
'총알탄 사나이 3 ', '에이스 벤츄라 ', '싸이파이터', '링거 ', '패스트 미드나잇', '슬리버', '스케치',
'콜드 나이트', '실루엣', '지옥의 일요일', '할리와 말보로맨', '살인개스', '죽음의 키스',
'데블스에드버킷', '자칼 ', '마스터마인드 ', '원죄적 본능 3 ', '탑오브더월드', '위험한진실',
'키스 더 걸 ', '피스메이커', '빌로우유토피아 ', '아메리칸북두권 ', '블러드

```
앤와인', '밤셸 ',
       '더블팀 ', '더블섹스', '브레이크다운', '바디체크 ', '레드엑트', '아나콘다
', '페이스오프',
       '섹슈얼 인텐트', '스피드 2 ', '콘에어 ', '쥬라기 공원 2 - 잃어버린 세계 ',
'앱솔루트 파워 ',
       '로미오이즈블리딩', '세인트 ', '용서받지 못할 관계 3 ', '셋잇오프 ', '미키
루크의 엑시트 인 레드 ',
       '메트로 ', '크래쉬 다이브 ', '비상탈출 ', '파고 ', '닥터모로의DNA ', '에너
미라인 ',
       '트루 벤전스 ', '예견된폭로 ', '생존게임 ', '나이트메어 4-꿈의 지배자', '다
크엔젤', '딥 식스',
       '비버리힐스 캅 2', '심령의 공포', '심연', '야성녀 아이비 ', '얼굴없는 암살
자', '엔젤 하트',
       '죽음의 찬미', '캣헌터', '크리미날로', '트로마스워', '프레데터2', '프린스
오브 다크니스', '헬 레이져',
       '환타즘 2', '007 살인면허'], dtype=object)
```

In [ ]:

```
# 미국 영화 목록 (애니메이션)
df_genres.query("1988 <= openYear <= 1998").loc[df_nations["N_미국"]==1].query("G_애니메이션 ==
 1")["movieNm"].values
```

Out[ ]:

```
array(['알라딘', '벅스 라이프', '미녀와 야수', '뮬란', '라이온 킹', '인어공주 ',
'토이 스토리 ',
       '난 이상한 사람과 결혼했다', '백조 공주 ', '톰과 제리', '공룡시대', '개미',
       '팀 버튼의 크리스마스 악몽', '포카혼타스 ', '정글북', '찰리의 하늘나라 대소
동', '푸른 골짜기',
       '이집트 왕자', '매직 스워드 ', '스페이스 잼', '월트디즈니의 노틀담의 꼽추
', '머팻 크리스마스 캐롤',
       '양배추인형의 클럽하우스 ', '아나스타샤', '월트디즈니의 헤라클레스 '], dtyp
e=object)
```

In [ ]:

```
# 홍콩 영화 목록 (액션)
df_genres.query("1988 <= openYear <= 1998").loc[df_nations["N_홍콩"]==1].query("G_액션 == 1")["movieNm"].values
```

Out[ ]:

```
array(['의본무언', '살수특급', '화평본위 ', '신정무문', '동성서취', '땡큐마담',
'천장지구 2', '종횡사해',
       '첩혈쌍웅', '천장지구', '영웅본색 3', '주윤발의 미스터 갱', '백발마녀전 2',
'패가자', '시티 헌터',
       '비룡맹장', '아비와 아기 ', '대도무문', '영웅본색 2', '대호출격', '소오강
호', '폴리스 스토리',
       '직격증인', '양리칭의 예스마담', '황비홍의서역웅사', '황비홍3-사왕쟁패',
'황비홍 2', '엽표행동',
       '무장원 소걸아', '도범2', '용형호제 2', '감옥풍운', '중환영웅', '황비홍',
'예스 마담2-황가전사',
       '열화전차', '첩혈가두', '폴리스 스토리3', '폴리스 스토리 2 - 구룡의 눈',
'소림사강호영패 ',
       '조수괴초', '첩혈남아 ', '지존본색', '백발 마녀전', '화소도', '가두패왕',
'광동오호',
       '철마류-가두살수', '마담 캅스', '성룡의 폴리스 스토리 4', '독망', '적나고양
2', '용호상조',
       '지존애림2', '동방불패 2', '땡큐마담 2', '독고구검', '흑금', '예스 마담3-중
화전사', '특경도룡',
       '공작왕', '칠복성', '모험왕 ', '성룡의 미라클', '용행천하', '동방불패', '영
웅 ', '동방삼협',
       '흑협', '기밀중안 ', '글로리아 입의 캣츠', '경천여장', '구마도장', '고혹녀
', '경망쌍웅', '가자왕',
       '귀호', '귀타여걸', '귀도귀', '귀경출사', '강시지존', '동사서독 ', '홍콩 여
복성', '패왕화',
       '경천12시', '공작왕 2 아수라', '광천용호투', '구복성', '군룡희봉', '수호전
지 영웅본색', '초류향',
       '진가락', '지존여장', '명월조천신사', '두호통집범 ', '도신불패 ', '도성대
형', '용소자',
       '대호출격 2', '와일드 ', '오호장', '영웅호한', '흑설', '유덕화의 도망자 ',
'소림오조 ', '비룡문',
       '화룡만가 ', '지존홍분', '지존애림', '정전자', '육지금마 ', '여아당자강',
'스트리트 파이팅',
       '벽혈신검 ', '살출군영', '천녀유혼', '도학위룡', '첩혈위룡(도학위룡 2) ',
'도신 2 ', '강호정',
       '성항기병 3 : 홍콩탈출', '지존소자', '자웅쌍혈', '천검절도', '태자전설',
'중안조', '비각칠',
       '재세추혼', '황비홍의 일대종사', '방세옥', '중원대결투', '서극의 칼 ', '블
러드 링 2 ', '자웅대도 ',
       '용호살수 ', '저격자 ', '무미신탐 ', '묘가 천장지구 ', '폭스 헌터 ', '성룡
의 썬더볼트 ',
       '이연걸의 탈출', '탈출', '폴리스 갱', '우주에서 온 사나이', '영웅대가', '무
림쌍걸', '폭풍소년',
       '검협전 ', '첩혈쌍웅 2 ', '흑창인', '뇌정행동 ', '이락부기안 ', '칠금강',
'폭풍의 눈 ',
       '폭열도시', '성항일호', '와저신산', '신협객행 ', '흑심귀', '정홍기하', '노
화광분', '무장원 철교삼 ',
       '쌍도여걸 ', '블랙캣', '선인장 ', '천사군단 ', '성향기병 IV 사자후 ', '영환
지존', '협도고비',
       '4인의 복수 ', '황비홍 소전', '녹야추격 ', '봉도필승', '반아종횡 ', '풍우동
로 ', '마역비룡 ',
       '신조협려2', '신용문객잔', '시티 보이즈 ', '사막의 드래곤 ', '추살자', '강
호영웅', '심사관',
       '신십이생초', '용호신풍운', '용등사해', '양리칭의 중출강호', '첩혈속집',
'월청', '대가대', '안락생포',
       '전격자', '풍운', '북경 예스마담2', '이연걸의 정무문 ', '대적성', '백련사신
', '아메리칸 소림',
       '맨하탄 잠복근무 ', '오룡체신', '벽혈남천 ', '독존자', '영춘권 ', '뇌전영
웅', '유민장원',
       '천여지 ', '도검소 ', '신영웅본색 ', '선학신침 ', '카멜레온 ', '흑표천하 ',
```

'황비홍-철계투오공 ',
'용쟁호투', '이연걸의 보디가드 ', '화룡풍운 ', '취권 3 ', '화급', '정전자 2 ', '적사판관 ',
'황비홍 5', '중금속 ', '화기소림 ', '동방대협 ', '성룡의 홍번구 ', '백장미', '요제연담지 오통신 ',
'의혈강호 ', '신 당산대형 ', '용호쟁투 ', '0양의 함정 ', '정무소영웅 ', '와일드 커플 ', '요마도',
'신 냉혈 13매', '마영정', '킬러의 3.5분 ', '상해탄 ', '홍등가의 혈투 ', '프로젝트 S',
'지존마담', '666 마귀부활 ', '무림객잔', '요수도시', '폭작령', '파호 2', '요괴도시',
'태극권 2 ', '라스트 예스마담 ', '맹룡과강 ', '포청천 칠협오의 ', '대삼원 ', '낭만풍폭 ',
'세븐 캅스 ', '현대 고혹자 ', '영웅불패 ', '신첩혈쌍웅 ', '메탈 캅스 ', '비호웅심2 ',
'아시안 커넥션 ', '호혈용천', '성룡의 CIA', '태극권', '무장원 장보자', '황비홍4-왕자지풍',
'방세옥 2 : 대도무문', '맹귀 패왕화', '혈염홍진', '절대쌍교', '묘가십이소', '영웅문', '오호사해',
'신조협려-무뢰한', '속 오복성', '동방노호', '의혈남아', '천라지망', '지존무상2', '추일', '열혈풍운',
'도성 2', '도협2', '사해교룡', '성시판관', '살수특경', '용봉타적', '블랙 캣 2',
'양리칭의 마담 시티헌터', '비호외전 ', '화소홍련사 ', '천룡팔부 ', '노화가두', '신용쟁호투',
'옥녀검 ', '취권 2 ', '차이나 폴리스', '대소비도', '일도경성', '철마류', '소도회', '지존불패',
'무림쌍절', '신 유성호접검', '의천도룡기 ', '동방추혼권 ', '협성 ', '전신', '뇌정소혈', '독호',
'의불용정', '뇌락정구', '서장소자', '탈명가두', '비응호제', '용혈미정', '신탐격전', '여걸쌍웅',
'천약유정', '기병', '경시종횡', '정고전가', '성전강호', '열혈천사', '봉황왕', '옥중룡',
'신투첩영 ', '비룡 ', '블러드문 ', '비호 ', '영앤데인저러스 ', '흑표풍운 ', '나이스가이 ',
'양자경의 스턴트우먼 ', '대형대부', '여경특공대', '구룡본색', '구룡신도', '대부양위', '도시의 아이들',
'로얄포리스 마담', '로즈킬러', '마고일장', '마담갱', '마담킬러', '마로영웅', '마피아 대 닌자',
'몽경영웅', '묘탐쌍웅', '무림지존', '무적쾌차', '무회행동', '벽력선봉', '복수의 만가', '빅보스',
'사대영웅', '살수천사', '상해 1920', '성룡의 일대영웅', '성전풍운', '성향기병 속집', '시티워',
'신룡쌍포', '쌍용번개', '여걸풍운', '여전사', '엽응계획', '영웅대도', '영웅대형', '영웅무언',
'영웅무적', '영웅복성', '영웅사저', '영웅우상', '영웅출격', '영웅투혼', '영환동자',
'예스 마담4-직격증인', '오야천사', '용지쟁패', '의본대형', '의혈쌍웅', '자유인', '자탄출조',
'재기풍운', '재전강호', '정무권', '중간인', '중화영웅', '지존계상', '지존무상', '천년귀인',
'철갑무적마리아', '첩혈강호', '첩혈전사', '청옥불', '최가박당5-신최가박당', '타이거맨', '포리스 마담',
'포리스 마담 2', '포리스 엔젤 마담', '포리스우먼', '폴리스 맨', '폴리스우먼 2', '풍운인물',
'프린세스 마담', '혈전영웅', '혈전자', '호담맹장', '홍콩엔젤킬러', '홍콩포리스 마담', '황가 특공대',
'황가마담', '황가여장', '흑전사', '흑해출격', '7 소복'], dtype=object)

In [ ]:

```
# 홍콩 영화 목록 (범죄)
df_genres.query("1988 <= openYear <= 1998").loc[df_nations["N_홍콩"]==1].query("G_범죄 == 1")["movieNm"].values
```

Out[ ]:

```
array(['메이드 인 홍콩 ', '종횡사해', '첩혈쌍웅', '영웅본색 2', '직격증인', '엽표행동', '용형호제 2',
       '첩혈가두', '폴리스 스토리 2 - 구룡의 눈', '첩혈남아 ', '화소도', '마담 캅스', '독망', '의담비봉',
       '노호광', '성룡의 미라클', '흑협', '경망쌍웅', '귀경출사', '흑사회', '지존여장', '도성대형',
       '살출군영', '성항기병 3 : 홍콩탈출', '자웅쌍혈', '중안조', '이연걸의 탈출', '폴리스 갱', '무림쌍걸',
       '이락부기안 ', '증인', '폭풍의 눈 ', '폭열도시', '정홍기하', '블랙캣', '성향기병 IV 사자후 ',
       '굉천 황가장', '용등사해', '쾌락적 소야계', '첩혈속집', '대적성', '맨하탄 잠복근무 ', '신영웅본색 ',
       '의혈강호 ', '와일드 커플 ', '킬러의 3.5분 ', '지존마담', '라스트 예스마담', '현대 고혹자 ',
       '신첩혈쌍웅 ', '메탈 캅스 ', '혈염홍진', '묘가십이소', '동방노호', '천라지망', '도성 2',
       '사해교룡', '성시판관', '파호', '살수특경', '와일드 엔젤', '노화가두', '뇌정소혈', '독호',
       '의불용정', '뇌락정구', '용혈미정', '용가호저', '신탐격전', '기병', '여도신', '경시종횡',
       '정고전가', '성전강호', '성시연가', '비호 ', '나이스가이 ', '색겁정겁', '구룡본색',
       '로얄포리스 마담', '마담갱', '빅보스', '여걸풍운', '엽응계획', '영웅사저', '용지쟁패', '의본대형',
       '포리스우먼', '프린세스 마담', '흑도영웅', '흑전사'], dtype=object)
```

In [ ]:

```
# 한국 영화 목록 (코미디)
df_genres.query("1988 <= openYear <= 1998").loc[df_nations["N_한국"]==1].query("G_코미디 == 1")[
"movieNm"].values
```

Out[ ]:

```
array(['조용한 가족', '마누라 죽이기', '3인조', '미지왕', '개그맨', '기막힌 사내
들', '결혼이야기',
       '넘버 3', '개같은 날의 오후', '아래층 여자와 위층 남자', '박봉곤 가출사건',
'닥터 봉', '할렐루야',
       '나의 사랑, 나의 신부', '패자부활전', '투캅스', '세상 밖으로', '누가 나를
미치게 하는가',
       '땡칠이와 쌍라이트', '진짜 사나이', '생과부 위자료 청구소송', '남자는 괴로
워', '투캅스 3', '투캅스 2',
       '아기공룡 둘리-얼음별 대모험', '계약 커플', '개그특공대 로봇트윈스', '체인
지', '비는 사랑을 타고',
       '커피 카피 코피', '억수탕(3pm Paradise)', '가진 것 없소이다', '가루지기',
'결혼이야기 2',
       '소쩍궁 탐정', '미끼', '장닭 고교얄개', '요절복통 신 양반뎐', '오색의 전
방', '똑바로 살아라',
       '돈을 갖고 튀어라', '헤어드레서', '오디션', '영구와 공룡 쮸쮸', '가보면 알
거야', '맨',
       '꼬리치는 남자', '도둑과 시인', '머저리와 등신들', '총잡이', '천재선언',
'내일은 참피온',
       '아빠는 보디가드', '가족시네마', '키스할까요', '엑스트라', '찜', '코르셋',
'토요일 오후2시',
       '죽이는 이야기', '소녀 18세', '사랑하고 싶은 여자 결혼하고 싶은 여자', '쿵
후보이 친미',
       '우뢰매 7 : 돌아온 우뢰매', '로맨스 황제', '미스터 맘마', '영구와 흡혈귀 드
라큐라', '정신나간 유령',
       '머저리와 도둑놈', '영구와 황금박쥐', '맹구와 북두신검', '삼중성', '토끼를
태운 잠수함',
       '영구와 땡칠이 4탄-홍콩할매귀신', '병팔이의 일기', '따봉수사대-밥풀떼기 형
사와 전봇대 형사', '인연',
       '일팔일팔(1818)', '스카이 닥터', '현상수배', '스커트 속의 드라마', '마지막
방위', '베이비 세일',
       '백수스토리', '산부인과', '박대박', '로케트는 발사됐다', '미스터 콘돔', '경
찰+칠득이와 너털도사',
       '밥풀때기 형사와 쌍라이트', '별난 두 영웅', '슈퍼 홍길동 3', '슈퍼홍길동',
       '슈퍼홍길동2-공초 도사와 슈퍼 홍길동', '심형래의 쫄병군단', '앗싸,호랑나
비', '영구와 땡칠이',
       '영구와 땡칠이 소림사가다', '영구와 땡칠이 3탄-영구람보', '외계인과 콩콩강
시', '짬뽕 홍길동',
       '쫄병수첩 2', '키스도 못하는 남자', '티라노의 발톱', '회장님 우리회장님',
'칙칙이의 내일은 참피온'],
      dtype=object)
```

In [ ]:

```
# 한국 영화 목록 (스릴러)
df_genres.query("1988 <= openYear <= 1998").loc[df_nations["N_한국"]==1].query("G_스릴러 == 1")[
"movieNm"].values
```

Out[ ]:

```
array(['올가미', '여고괴담', '장미의 나날', '태양 속의 남자', '가위여자', '천년환
생', '피아노맨', '손톱'],
      dtype=object)
```

In [ ]:

```
# 한국 영화 목록 (공포)
df_genres.query("1988 <= openYear <= 1998").loc[df_nations["N_한국"]==1].query("G_공포 == 1")["movieNm"].values
```

Out[ ]:

```
array(['301·302', '퇴마록', '올가미', '여고괴담', '구미호', '천년환생', '신의 이
방인',
       '영구와 흡혈귀 드라큐라', '천녀원앙'], dtype=object)
```

In [ ]:

```
# 한국 영화 목록 (여고괴담 시리즈)
df_genres.loc[df_genres["movieNm"].str.contains("여고")]
```

Out[ ]:

	movieCd	movieNm	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬	G_미스터리	G_범죄
6600	19990056	여고괴담 두번째 이야기	1999	0	0	0	1	0	0	0	0	0	0	0
10820	20030100	여고괴담 세번째 이야기: 여우계단	2003	0	0	0	1	0	0	0	0	0	0	0
11990	20141143	이지메-여고생의 혼잣말	2016	0	0	0	0	0	0	1	0	0	0	0
12244	19980035	여고괴담	1998	0	0	0	1	0	0	0	0	0	0	0
12586	20090327	여고괴담 5	2009	0	0	0	1	0	0	0	0	0	0	0
13392	20050252	신주쿠 여고생 납치 사건(완전한 사육)	2005	0	0	0	0	0	0	1	0	0	0	0
13616	20088594	소녀괴담 : 17살 여고생의 악몽	2015	0	0	0	1	0	0	0	0	0	0	0
14301	20000047	대학로에서 매춘하다가 토막살해당한 여고생 아직 대학로에 있다	2000	1	0	0	1	0	0	0	0	0	0	0
17723	20050142	여고괴담 4 : 목소리	2005	0	0	0	1	0	0	0	0	0	0	0
18918	19720052	여고시절	1972	0	0	0	0	0	0	1	0	0	0	0
18986	19750050	여고 졸업반	1975	0	0	0	0	0	0	1	0	0	0	0
19033	19740070	여고 교사	1974	0	0	0	0	0	0	0	1	0	0	0
19380	19770066	여고얄개	1977	0	0	0	0	0	0	1	1	0	0	0
20429	20040766	여고생 시집가기	2004	0	0	0	0	0	0	0	0	0	0	0

In [ ]:

```
# 한국 영화 목록 (드라마)
df_genres.query("1988 <= openYear <= 1998").loc[df_nations["N_한국"]==1].query("G_드라마 == 1")[
"movieNm"].values
```

Out[ ]:

```
array(['서울에비타', '빠담풍', '해가 서쪽에서 뜬다면', '접속', '조용한 가족', '마스카라', '나쁜 영화',
       '부활의 노래', '있잖아요 비밀이에요', '돼지가 우물에 빠진 날', '달마가 동쪽으로 간 까닭은?', '본 투 킬',
       '칠수와 만수', '축제', '초록물고기', '8월의 크리스마스', '파업전야', '마누라 죽이기', '영심이',
       '장군의 아들 2', '아름다운 청년 전태일', '우리들의 일그러진 영웅', '산딸기 5', '개그맨',
       '가슴에 돋는 칼로 슬픔을 자르고', '기막힌 사내들', '게임의 법칙', '사랑하기 좋은 날', '쫄병수첩',
       '여군외출', '아들나라', '세상은 살만큼 아름답다', '알바트로스', '밀월', '서편제', '백치애인',
       '걸어서 하늘까지', '불행한 아이의 행복', '복카치오 `93', '이혼녀', '무궁화 꽃이 피었습니다', '말미잘',
       '명자 아끼꼬 쏘냐', '마유미', '결혼이야기', '남부군', '칠삭동이의 설중매 ', '처녀들의 저녁식사',
       '세 친구', '키드캅', '정사', '달은...해가 꾸는 꿈', '두여자 이야기', '은마는 오지 않는다',
       '개같은 날의 오후', '학생부군신위', '미술관 옆 동물원', '편지', '별이 빛나는 밤에',
       '아래층 여자와 위층 남자', '장군의 아들 3', '장군의 아들', '올가미', '달콤한 신부들', '여자의 일생',
       '눈꽃', '행복은 성적순이 아니잖아요', '바람부는 날이면 압구정동에 가야 한다', '그대안의 블루',
       '아제아제 바라아제', '우리는 지금 제네바로 간다', '꽃잎', '너에게 나를 보낸다', '경마장 가는 길',
       '무소의 뿔처럼 혼자서 가라', '그들도 우리처럼', '강원도의 힘', '할렐루야', '나의 사랑, 나의 신부',
       '태백산맥', '투캅스', '누가 용의 발톱을 보았는가', '세상 밖으로', '하얀전쟁', '사의 찬미',
       '아스팔트 위의 동키호테', '화엄경', '크리스마스에 눈이 내리면', '젊은 남자', '내 안에 우는 바람',
       '장미빛 인생', '비 오는 날 수채화2', '가을여행', '이장호의 외인구단 2', '비 오는 날 수채화',
       '김의 전쟁', '시간은 오래 지속된다', '헐리우드 키드의 생애', '진짜 사나이', '바이 준', '유리',
       '생과부 위자료 청구소송', '단지 그대가 여자라는 이유만으로', '첫사랑', '겨울연가', '수잔브링크의 아리랑',
       '그 섬에 가고 싶다', '역사는 이렇게 시작됐다', '누가 붉은 장미를 꺾었나', '혼자도는 바람개비',
       '열아홉의 절망끝에 부르는 하나의 사랑노래', '소낙비', '계약 커플', '거꾸로 가는 여자', '애니깽',
       '그 여자, 그 남자', '가슴달린 남자', '체인지', '라이 따이한', '무거운 새', '비는 사랑을 타고',
       '어린 연인', '오사카의 푸른 밤', '커피 카피 코피', '태양 속의 남자', '한줌의 시간속에서', '씨내리',
       '억수탕(3pm Paradise)', '표류일기', '개벽', '나는 소망한다 내게 금지된 것을',
       '겨울꿈은 날지 않는다', '까', '꿈의 정사', '그대품에 아카시아 향기', '겨울만가', '공룡선생', '꾼사냥',
       '가위여자', '가출소년과 쌍코대작전', '갈마', '가진 것 없소이다', '겨울미리내',
       '끼있는 여자는 밤이슬을 좋아한다', '꽃그늘 속에 부는 바람', '검으나 땅에 희나 백성', '결혼이야기 2',
       '꼴찌부터 일등까지 우리반을 찾습니다', '투 타이어드 투 다이', '인생이 뭐 객관식 시험인가요', '캘리포니아',
       '굿모닝 대통령', '그래 가끔 하늘을 보자', '그리움엔 이유가 없다', '꼭지딴', '꽃지', '그들만의 세상',
       '산산이 부서진 이름이여', '사랑의 이름으로', '변신전사 트렌스토디', '채널
```

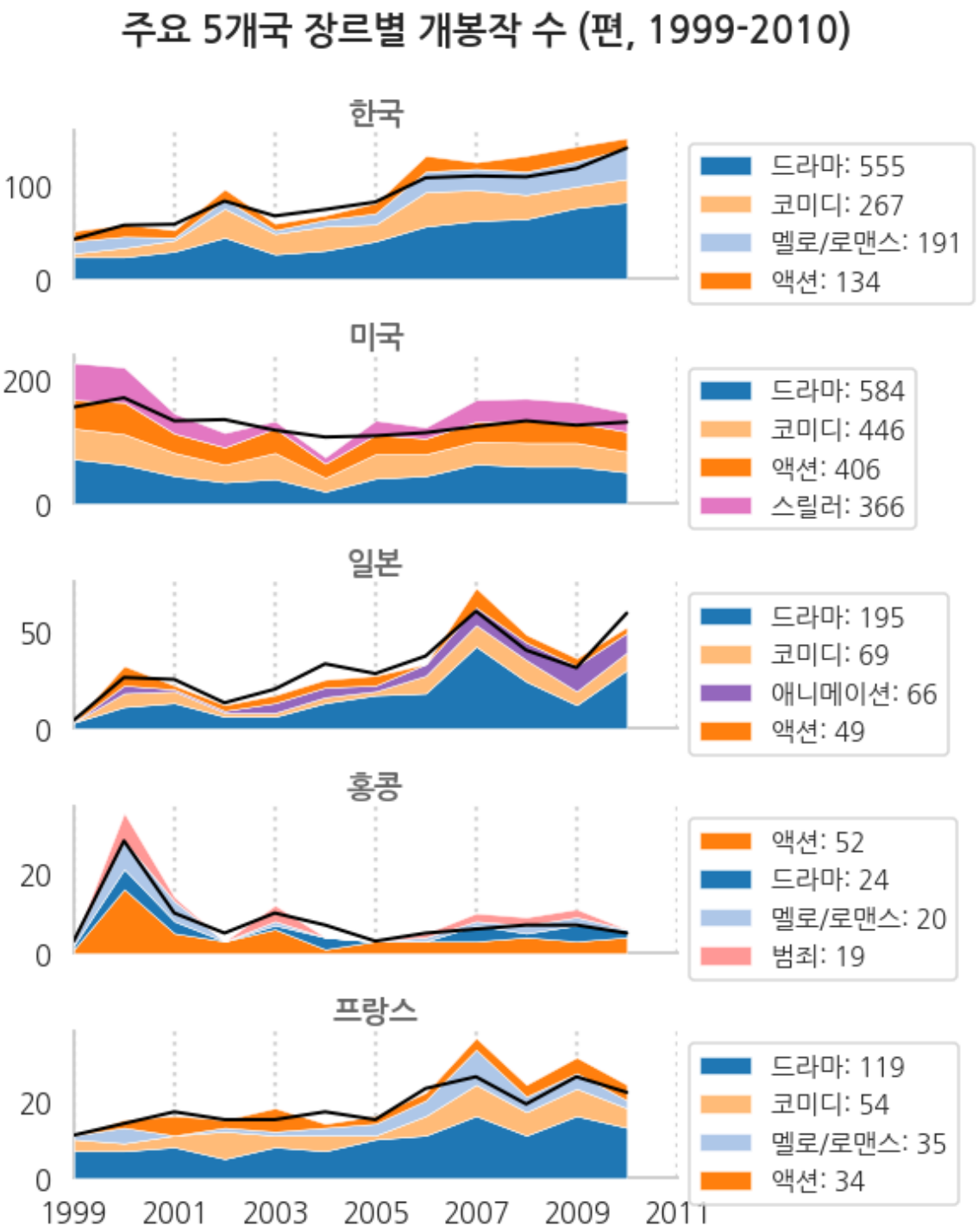
식스 나인', '배꼽위에 여자',
'미끼', '장닭 고교얄개', '이혼하지 않은 여자', '데카당스 37.2', '유리벽 속의 두 연인', '위험수위',
'요절복통 신 양반뎐', '오직 단 한번뿐인 내 인생인데', '열일곱살의 쿠데타', '나에게 오라',
'창(노는계집 창)', '며느리 밥풀꽃에 대한 보고서', '마리아와 여인숙', '여자의 시대는 끝나지 않는다',
'러브 러브', '아름다운 시절', '네온 속으로 노을지다', '내일로 흐르는 강', '어허 어이 어이 가리',
'눈물의 웨딩 드레스', '째즈빠 히로시마', '울고 싶어라', '실락원', '드래곤 투카', '노란집', '오디션',
'잘돼갑니다', '영구와 공룡 쮸쮸', '백한번째 프로포즈', '바리케이드', '웨스턴 애비뉴 ',
'에미 이름은 조센삐였다', '빨간 색깔의 여자', '혼자뜨는 달', '내가 성에 관해 알고 있는 몇가지 이야기들',
'삘구', '배꼽버스', '48+1', '깊은 슬픔', '빛은 내 가슴에', '파워킹', '천재 선언',
'어둔 밤 어둔 곳에', '시비시비', '비에 젖어서야 사랑하노라', '타악기의 계절', '마지막 남자의 모습',
'이조여인 숙명사 청상계', '용의 유혼', '내일은 참피온', '여자 배달부는 고장난 신호등', '서울의 달빛',
'매혹', '머나먼 사이공', '독재 소공화국', '에덴의 서쪽', '미녀사냥', '동춘별곡', '성인신고식',
'장미여관', '사랑은 지금부터 시작이야', '나를 보라 아메리카', '뽀식이와 꼬마특공단', '우주전사 불의 사나이',
'우리가 진정 가야할 곳은', '마당쇠와 언년이', '사랑받는 여자', '드라큐라 애마',
'립스틱 그리고 남자의 사랑', '하몽하몽 서울', '우리시대의 사랑', '휘모리', '아주 특별한 변신',
'헤드폰을 벗어라', '태권 파이터', '애수의 하모니카', '가족시네마', '짱', '파란대문', '키스할까요',
'엑스트라', '아름다운 게임', '이도백화', '7악동', '악어', '선유락', '유혹의 샘', '세븐틴', '찜',
'어른들은 청어를 굽는다', '코르셋', '토요일 오후2시', '피아노맨', '사라는 유죄', '정글스토리',
'남자이야기', '손톱', '카루나', '물랭루즈', '죽이는 이야기', '뜨거운 비', '소녀 18세', '도색부인',
'사랑하고 싶은 여자 결혼하고 싶은 여자', '쿵후보이 친미', '우뢰매 7 : 돌아온 우뢰매', '13월의 겨울',
'신혼여행 특공대', '짧은 여행의 끝', '서울에서 마지막 탱고 2', '마음의 파수꾼', '로맨스 황제',
'돌아오라 개구리소년', '미스터 맘마', '원색의 청춘', '무인도의 남과 여', '뜨거운 바다', '성애의 침묵',
'캉캉 69', '애마부인 7', '아들과 연인', '영구와 흡혈귀 드라큐라', '정신나간 유령',
'안개에 젖은 리오의 밤은 깊어', '우리사랑 이대로', '복카치오 92', '넝쿨속의 히야신스', '땅끝에 선 연인',
'나 이제 너를 잊으리', '새알각하', '백백교', '세상 끝의 향기', '스물일곱 송이 장미',
'모두가 죽이고 싶던 여자', '비황', '언제나 막차를 타고 오는 사람', '나는 너를 천사라고 부른다',
'변금련 2', '비처럼 음악처럼', '세나의 신혼일기', '여비서클럽', '숲속의 방', '황홀한 유혹',
'이별아닌 이별', '섬강에서 하늘까지', '매춘 3', '여자아리랑', '참견은 노~ 사랑은 오예~',
'내마음에는 악어가 산다', '아침이 오면 그대이름으로', '아담이 눈뜰 때', '사랑의 종합병원', '애마부인 8',
'모스크바에서 온 S여인', '오렌지나라', 'B타임의 정사 2', '스무살까지만 살고 싶어요', '탄드라부인',
'전갈군단', '서울야누스', '붉은 신호에 걷는 여자', '패배자', '맨발에서 벤츠까지', '첫날밤에 내린 비',

'미지의 흰새', '장미빛 여자', '뽕밭나그네', '천국의 계단', '비개인 오후를 좋아하세요',
'자전거를 타고온 연인', '사랑스런 이웃집 여자', '메리제인', '서울의 눈물', '테레사의 연인', '뻘',
'아그네스를 위하여', '제5의 사나이', '사랑과 눈물', '여자는 추억속에 집을 짓는다', '서글픈 사랑의 시작',
'1991 인간시장 3', '외길가게 하소서', '비너스 여름시', '나의 아내를 슬프게 하는 것들',
'핸드백속 이야기', '10대의 반항', '조금만 참아줘요', '아르바이트 여자', '불새의 춤', '하얀 비요일',
'지금 우리는 사랑하고 싶다', '잃어버린 너', '푸른 옷소매', '용궁에서 온 거북이와 이무기', '베를린 리포트',
'피와 불', '토끼를 태운 잠수함', '네 멋대로 해라', '하나님이 어디 있어요', '달빛타는 여자',
'15년 3시간 전', '미아리텍사스', '밀크초콜릿 1950-1990', '예수천당', '병팔이의 일기',
'내일은 비', '밤으로의 긴 여로', '실크커텐 저편', '이태원 밤하늘엔 미국달이 뜨는가', '불꽃 슛 통키',
'야생동물보호구역', '베이비 세일', '큐', '백수스토리', '산부인과', '아버지', '로케트는 발사됐다',
'용병이반', '아메리카 아메리카', '25불의 인간', '늑대의 호기심이 비둘기를 훔쳤다', '구로 아리랑',
'나는 날마다 일어선다', '난 깜짝놀랄 짓을 할거야', '남자시장', '내사랑 동키호테', '내친구 제제',
'누가 꽃밭에 불을 지르랴', '늪속의 여촌장', '단단한 놈', '달리는 도시', '담다디', '대남', '대물',
'대통령의 딸', '대학촌의 달빛', '도시의 상처', '독불장군', '돈아 돈아 돈아', '동경 아리랑',
'떠돌이 까치', '랏슈', '로티의 모험', '마리화나', '만무방', '물의나라', '미리마리 우리두리',
'미쓰코뿔소 미스터코란도', '바람의 아들', '반쪽 아이들', '발바리의 추억', '벌거벗은 분노',
'별난 두 영웅', '복카치오 94', '불여우', '블루하트', '빨강바다', '사랑의 낙서', '삼토스와 댕기똘이',
'새벽외출', '새앙쥐 상륙작전', '서울 무지개', '수탉', '슈퍼맨 일지매', '심형래의 쫄병군단',
'아리수변 꿈나무', '앗싸,호랑나비', '애마부인 4', '어둠의 딸', '어른들은 몰라요', '언제나 그 자리에',
'여인 파티', '영구와 땡칠이', '영구와 땡칠이 소림사가다', '오성군,한음군', '오세암(극영화)',
'용감한 사람들', '용호쿼', '우담바라', '우연한 여행', '유아독존', '이제 그 여자는 여기 살지 않는다',
'인신매매', '임소로의 부마동자', '잠자리에 들시간', '죄없는 병사들', '지금은 양지',
'지리산의 성난 토끼들', '쫄병수첩 2', '천국의 땅', '천국의 비밀', '청송으로 가는 길', '친구야 친구야',
'칠색조', '캠퍼스 연애특강', '캬바레부인', '코리안 커넥션', '키스도 못하는 남자', '햄버거 쟈니',
'휴거', '49일의 남자', '89 인간시장 오 하나님'], dtype=object)

3.7.4. 1999-2010

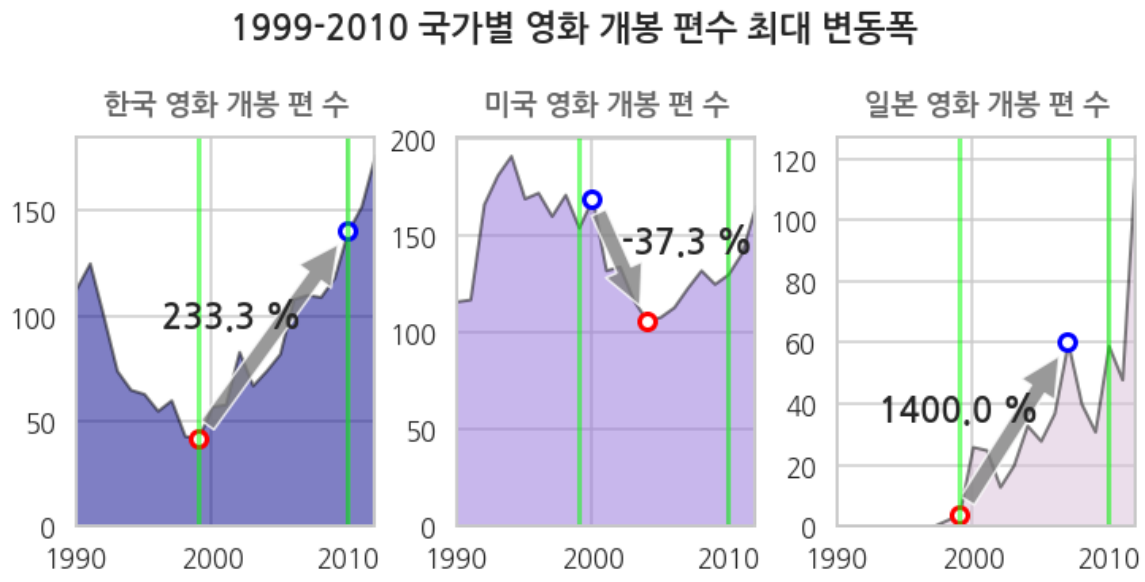
In [ ]:

```
plot_time_GN(1999, 2010, topN=4)
```



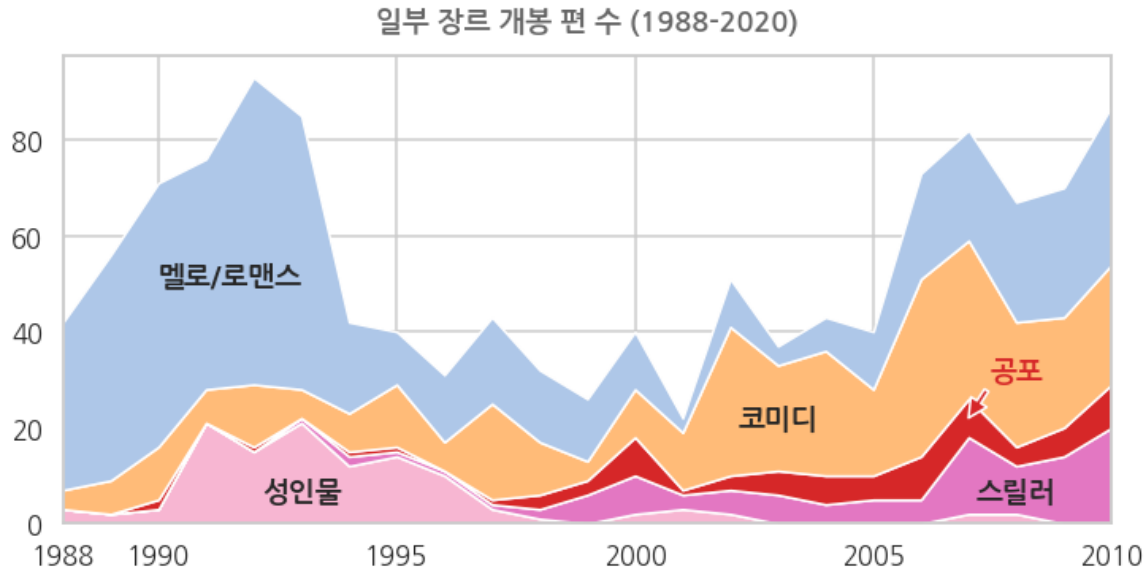
In [ ]:

```
# 한국, 미국, 일본 개봉 편 수 비교
from matplotlib.patches import FancyArrowPatch

fig, axs = plt.subplots(ncols=3, figsize=(10, 5), constrained_layout=True, sharex=True)

for (ax, n, c) in zip(axs, ["한국", "미국", "일본"], [c_kr, c_us, c_jp]):
    df = df_nationsY.query("1988 <= openYear <= 2012")
    ax.stackplot(df["openYear"], df[f"N_{n}"], colors=[c], ec="k", lw=2, alpha=0.5)
    min_x = df.query("1999 <= openYear <= 2010")["openYear"].iloc[df.query("1999 <= openYear <=
 2010")[f"N_{n}"].argmin()]
    min_y = df.query("1999 <= openYear <= 2010")[f"N_{n}"].min()
    max_x = df.query("1999 <= openYear <= 2010")["openYear"].iloc[df.query("1999 <= openYear <=
 2010")[f"N_{n}"].argmax()]
    max_y = df.query("1999 <= openYear <= 2010")[f"N_{n}"].max()
    ax.scatter(min_x, min_y, s=100, ec="r", lw=3, fc="w")
    ax.scatter(max_x, max_y, s=100, ec="b", lw=3, fc="w")

    arrow_start_x, arrow_end_x = (min_x, max_x) if min_x < max_x else (max_x, min_x)
    arrow_start_y, arrow_end_y = (min_y, max_y) if min_x < max_x else (max_y, min_y)
    arrow = FancyArrowPatch((arrow_start_x, arrow_start_y), (arrow_end_x, arrow_end_y),
                            shrinkA=10, shrinkB=10, mutation_scale=50, fc="gray", alpha=0.8)
    ax.add_artist(arrow)
    shift_x = 0
    if n == "한국":
        shift_x = -3
    elif n == "미국":
        shift_x = 5
    elif n == "일본":
        shift_x = -4
    ax.text((min_x + max_x)/2 +shift_x, (min_y + max_y)/2, f"{(arrow_end_y - arrow_start_y)/arro
w_start_y*100:.1f} %",
            fontsize="large", fontweight="bold", ha="center", va="bottom")

    ax.axvline(1999, lw=3, c="#00FF00", alpha=0.5)
    ax.axvline(2010, lw=3, c="#00FF00", alpha=0.5)
    ax.set_xlim(1990, 2012)
    ax.set_title(f"{n} 영화 개봉 편 수", fontdict=font_title, pad=16)

fig.suptitle("1999-2010 국가별 영화 개봉 편수 최대 변동폭₩n", fontweight="bold")
fig.savefig("./images/num_delta.png", dpi=200)
```



In [ ]:

```
# 한국 성인물 vs 액션

fig, ax = plt.subplots(figsize=(10,5), constrained_layout=True)

ax.stackplot(df_genresY_kr.query("1988 <= openYear <= 2010")["openYear"],
             [df_genresY_kr.query("1988 <= openYear <= 2010")["G_성인물"],
              df_genresY_kr.query("1988 <= openYear <= 2010")["G_스릴러"],
              df_genresY_kr.query("1988 <= openYear <= 2010")["G_공포"],
              df_genresY_kr.query("1988 <= openYear <= 2010")["G_코미디"],
              df_genresY_kr.query("1988 <= openYear <= 2010")["G_멜로/로맨스"]],
             colors=[c_ero, c_thriller, c_horror, c_comedy, c_romance], labels=["성인물", "스릴
러", "공포", "코미디", "멜로/로맨스"])
ax.set_xlim(1988, 2010)
ax.set_title("일부 장르 개봉 편 수 (1988-2020)", fontdict=font_title, pad=16)

ax.text(1993, 5, "성인물", fontweight="bold", ha="center")
ax.text(2008, 5, "스릴러", fontweight="bold", ha="center")
ax.annotate("공포", (2007, 22), (2008, 30),  c=c_horror, fontweight="bold", ha="center",
            arrowprops={"width":3, "fc":c_horror})
ax.text(2003, 20, "코미디", fontweight="bold", ha="center")
ax.text(1991.5, 50, "멜로/로맨스", fontweight="bold", ha="center")

xticks = [1988, 1990, 1995, 2000, 2005, 2010]
ax.set_xticks(xticks)
fig.savefig("./images/genre_change_1988.png", dpi=200)
```



In [ ]:

```
df_genresY_kr.query("1988 <= openYear <= 1998")["G_성인물"]/df_nationsY.query("1988 <= openYear
 <= 1998")["N_한국"]
```

Out[ ]:

```
17    0.037500
18    0.022989
19    0.026549
20    0.168000
21    0.150000
22    0.283784
23    0.184615
24    0.222222
25    0.181818
26    0.050000
27    0.023256
dtype: float64
```

In [ ]:

```
print(df_genres_kr.query("1988 <= openYear <= 1998").query("G_성인물==1")["movieNm"].values)
```

```
['산딸기 5' '산딸기 6' '산딸기 4' '복카치오 `93' '그 여자의 숨소리' '고금소총 2'
'고금소총 3'
 '거부하는 몸짓으로 저 하늘을' '금병풍월' '꿈의 정사' '꾼사냥' '갈마' '끼있는 여자
는 밤이슬을 좋아한다' '배꼽위에 여자'
 '데카당스 37.2' '겨울애마, 봄' '서울 엠마누엘' '러브호텔 비상구' '리허설' '안개
속에서 2분 더' '서울 미란다'
 '95 캉캉 69' '에로틱 걸' '애마와 백수건달' '매춘 6' '동춘별곡' 'B타임의 정사 4'
'빨간앵두 8'
 '짧은 시간 긴 시간' '드라큐라 애마' '립스틱 그리고 남자의 사랑' '하몽하몽 서울'
'자정이후' '비,여자 그리고 에로티시즘'
 '애마와 변강쇠' '미란다' '매춘 5' '애마부인 11' '숲속의 여자' '에로스 2' '96 옐로
우 하우스' '하얀 노을'
 '에로엑스' '물위의 하룻밤' '콜렉터' '립스틱 짙게 바르고' '여자가 타락하는 이유'
'홧김에?' '헬로우 변강쇠'
 '맥주가 애인보다 좋은 7가지 이유' '불륜의 바다' '애마부인 9' '도색부인' '뽕띠' 'B
타임의 정사' '봉성부인'
 '무인도의 남과 여' '뽕 3' '성애의 침묵' '캉캉 69' '소녀경' '애마부인 7' '안개에
젖은 리오의 밤은 깊어'
 '복카치오 92' '늪속의 불안개는 잠들지 않는다' '변금련 2' '매춘 3' '여자아리랑' 'B
타임의 정사 3'
 '연인들의 멜로디' '사라의 계절' '무엇에 쓰는 물건인고' '성인(Adult Movie)스' '애
마부인 8' 'B타임의 정사 2'
 '탄드라부인' '애마부인 6' '뽕밭나그네' '바람불면 일어서는 숲' "복카치오' 91" '성
숙한 외출'
 '엑스(X)성을 가진 여자' '팁주는 여자' '애마부인 5' '밥만 먹고 못살아' '변금련'
'어느 중년부인의 위기'
 '달빛타는 여자' '미아리텍사스' '후리꾼' '무릎위의 여자' '스커트 속의 드라마' '크
레이지 댄스' '사랑 그리고 죽는 연습'
 '매춘 2' '매춘 4' '변강쇠 3' '복카치오 94' '빨간앵두 5' '뽕 2' '살꼬지' '애마부인
10' '애마부인 4'
 '짚시애마' '파리애마']
```

In [ ]:

```
# 한국 영화 목록 (코미디)
df_genres.query("1999 <= openYear <= 2010").loc[df_nations["N_한국"]==1].query("G_코미디 == 1")[
"movieNm"].values
```

Out[ ]:

```
array(['음란서생', '당신이 잠든 사이에', '미녀는 괴로워', '마이 뉴 파트너', '국가대표', '나두야 간다',
       '광식이 동생 광태', '구미호 가족', '걸스카우트', 'YMCA야구단', '몽정기2', '그때 그 사람들',
       '여교수의 은밀한 매력', '엽기적인 그녀', '지구를 지켜라', '일단뛰어', '반칙왕', '아치와 씨팍',
       '브라보 마이 라이프', '마법경찰 갈갈이와 옥동자', '나는 곤경에 처했다!', '꿈은 이루어진다', '미쓰 홍당무',
       '청담보살', '원스어폰어타임', '울학교 이티', '어린 신부', '순정만화', '귀여워', '선생 김봉두',
       '대한민국 헌법 제1조', '헬로우 마이 러브', '낮술', '웰 또 그렇게까지', '이웃집 좀비', '화산고',
       '우린 액션배우다', '은하해방전선', '1번가의 기적', '최강로맨스', '헬로우 고스트', '아기와 나',
       '무림여대생', '펀치레이디', '구타유발자들', '주유소 습격사건 2', '방자전', '달콤한 거짓말',
       '내 눈에 콩깍지', '마이 캡틴 김대출', '굿모닝 프레지던트', '권순분여사 납치사건', '바르게 살자', '싸움',
       '슈퍼맨이었던 사나이', '차우', '누가 그녀와 잤을까?', '삼거리극장', '언니가 간다', '마파도2',
       '색즉시공 시즌2', '만남의 광장', '못말리는 결혼', '이장과 군수', '구세주', '예의없는 것들',
       'B형 남자친구', '연애술사', '좋지 아니한가', '동갑내기 과외하기 레슨 2', '투사부일체', '방과후 옥상',
       '모두들, 괜찮아요?', '귀신이 산다', 'S 다이어리', '여선생 VS 여제자', '달마야, 서울 가자',
       '돈 텔 파파', '달마야 놀자', '파송송 계란탁', '마파도', '연애의 목적', '박수칠 때 떠나라',
       '싸움의 기술', '불어라 봄바람', '황산벌', '위대한 유산', '영어완전정복', '맹부삼천지교', '돌려차기',
       '라이터를 켜라', '가문의 영광', '광복절 특사', '동갑내기 과외하기', '오! 해피데이', '정글쥬스',
       '플란다스의 개', '바람피기 좋은 날', '7급 공무원', '복면달호', '과속스캔들', '공포택시',
       '천하장사 마돈나', '백만장자의 첫사랑', '스카우트', '색즉시공', '경축! 우리 사랑', '방가? 방가!',
       '신라의 달밤', '용의주도 미스신', '어디선가 누군가에 무슨일이 생기면 틀림없이 나타난다 홍반장', '싱글즈',
       '주유소 습격사건', '두사부일체', '조폭마누라', '전우치', '해적 디스코왕 되다', '애자',
       '시라노; 연애조작단', '행복한 장의사', '품행제로', '그녀를 믿지 마세요', '내사랑 싸가지',
       '고독이 몸부림칠 때', '다찌마와리', '간첩 리철진', '김씨표류기', '다세포소녀', '목포는 항구다',
       '맨발의 기봉이', '이층의 악당', '내 깡패 같은 애인', '반가운 살인자', '육혈포 강도단', '킬러들의 수다',
       '간 큰 가족', '낭만자객', '정승필 실종사건', '가문의 위기(가문의 영광2)',
       '가문의 부활 - 가문의 영광3', '공필두', '미스터 주부퀴즈왕', '시실리 2km', '신석기 블루스',
       '야수와 미녀', '잠복근무', '조폭마누라3', '청춘만화', '쩨쩨한 로맨스', '누구나 비밀은 있다',
       '투 가이즈', '그녀를 모르면 간첩', '홍길동의 후예', '이대로, 죽을 순 없다(Short Time)',
       '로맨틱 아일랜드', '퀴즈왕', '쇼쇼쇼(Show Show Show)', '동해물과 백두산이', '가루지기',
       '내 남자의 순이', '남남북녀', '라듸오 데이즈', '페스티발', '판타스틱 자살소동',
       '김관장 대 김관장 대 김관장', '마강호텔', '쏜다', '두 얼굴의 여친', '마을금고 연쇄습격사건',
```

'흡혈형사 나도열', '뚝방전설', '원탁의 천사', '작업의 정석', '오! 브라더스', '라스트 갓파더',
'꽃미남 연쇄 테러 사건', '집 나온 남자들', '신장개업', '묻지마 패밀리', '색화동',
'갈갈이 패밀리와 드라큐라', '교도소 월드컵', '굳세어라 금순아', '긴급조치 19호', '걸프렌즈',
'그놈은 멋있었다', '공포특공대', '고추불패', '호박전', '마지막 늑대', '사사건건',
'첫사랑 사수 궐기 대회', '재밌는 영화', '죽어도 해피엔딩', '유감스러운 도시', '7인의 새벽', '무도리',
'역전의 명수', '생, 날선생', '카리스마 탈출기', '올드미스다이어리', '결혼식 후에', '바리바리 짱',
'구세주 2', '라이어', '어깨동무', 'Mr.로빈 꼬시기', '상사부일체', '불타는 내 마음',
'싸이보그 그녀', '약수터 부르스', '최후의 만찬', '은장도', '역전에 산다', '보리울의 여름',
'아빠가 여자를 좋아해', '노르웨이의 숲', '부킹쏘나타', '소녀X소녀', '후로거츠', '여고생 시집가기',
'휴머니스트', '휘파람공주', '철없는 아내와 파란만장한 남편 그리고 태권소녀', '유아독존', '몽정기',
'2424', '남자 태어나다', '마법의 성', '도둑맞곤 못살아', '우렁각시', '패밀리',
'좋은사람있으면 소개시켜줘', '서프라이즈', '뚫어야 산다', '네발가락', '울랄라씨스터즈', '아이언 팜',
'불량남녀', '젓가락', '미스터좀비', '킬 미', '물 좀 주소', '우리 집에 왜 왔니',
'애정결핍이 두 남자에게 미치는 영향', '잔혹한 출근', '잘 살아보세', '달려라 장미', '썬데이 서울',
'사랑방 선수와 어머니', '챔피언 마빡이', '산전수전', '하면 된다', '주노명 베이커리', '자카르타',
'이프', '신혼여행(身魂旅行)', '1724 기방난동사건', '쉿! 그녀에겐 비밀이에요', '잘못된 만남',
'흑심모녀', '날나리 종부전', '전투의 매너', '색다른 동거', '대한이, 민국씨',
'보스 상륙 작전(Boss X File)', '주글래 살래'], dtype=object)

In [ ]:

```
# 일본 영화 목록 (코미디)
df_genres.query("1999 <= openYear <= 2010").loc[df_nations["N_일본"]==1].query("G_코미디 == 1")[
"movieNm"].values
```

Out[ ]:

```
array(['뒤에서 앞에서', '혐오스런 마츠코의 일생', '안경', '요시노 이발관', '키사라
기 미키짱', '훌라걸스',
       '다케시즈', '기쿠지로의 여름', '모두 하고 있습니까?', '피쉬 스토리', '신부
의 수상한 여행가방',
       '이웃집 야마다군', '고양이의 보은', '폼포코 너구리 대작전', '괜찮아, 정말
괜찮아', '토일렛', '텐텐',
       '개구리중사 케로로', '남극의 쉐프', '구구는 고양이다', '개 달리다', '간장선
생',
       '명랑한 갱이 지구를 움직인다', '포켓몬스터 - 뮤츠의 역습', '춤추는 대수사선
2',
       '노다메 칸타빌레 Vol.1 ', '투엘디케이', '디트로이트 메탈 시티', '3-4X10월',
       '스즈미야 하루히의 소실', '사무라이픽션 - 적영', '마미야 형제', '눈물이 주
룩주룩', '포스트맨 블루스',
       '워터 보이즈', '퍼머넌트 노바라', '녹차의 맛', '우리들과 경찰아저씨의 700일
전쟁',
       '짱구는 못말려 극장판:태풍을 부르는 노래하는 엉덩이 폭탄', '원피스 - 기계태
엽성의 메카거병', '해피 플라이트',
       '워터스', '전차남', '도쿄 마블 초콜릿', '지옥갑자원', '자살관광버스', '린다
린다 린다', '스윙걸즈',
       '웰컴 미스터 맥도날드', '으랏차차 스모부', '열흘 밤의 꿈', '매직 아워', '춤
추는 대수사선',
       '아르헨티나 할머니', '상하이의 밤', '사무라이 픽션', '콰이어트 룸에서 만나
요',
       '케로로 더 무비:케로로 VS 케로로 천공대결전', '소림소녀', '극장판 도라에몽:
진구의 마계대모험 7인의 마법사',
       '달은 어디에 떠 있는가', '하나', '달려라! 타마코', '마츠가네 난사사건', '비
밀의 화원', '팬시댄스',
       '행복한 가족계획', '생일선물', '붕대클럽'], dtype=object)
```

In [ ]:

```
# 한국 영화 목록 (액션)
df_genres.query("1999 <= openYear <= 2010").loc[df_nations["N_한국"]==1].query("G_액션 == 1")["movieNm"].values
```

Out[ ]:

```
array(['마이 뉴 파트너', '국가대표', '청풍명월(淸風明月)', '소년은 울지 않는다',
'사생결단', '일단뛰어',
       '클레멘타인', '아치와 씨팍', '원더풀 데이즈', '한반도', '인정사정 볼 것 없
다', '말죽거리 잔혹사',
       '좋은 놈, 나쁜 놈, 이상한 놈 ', '죽거나 혹은 나쁘거나', '아저씨', '원스어폰
어타임', '추격자',
       '달콤한 인생', '의형제', '쉬리', '형사 Duelist', '짝패', '무적자', '용가
리', '하류인생',
       '이웃집 좀비', '숙명', '화산고', '마린보이', '2009로스트메모리즈', '우린 액
션배우다', '비열한 거리',
       '킬리만자로', '해운대', '미스터 소크라테스', '피도 눈물도 없이', '해결사',
'공공의 적', '황해',
       '무림여대생', '주유소 습격사건 2', '평행이론', '무방비도시', '신기전', '차
우', '인사동 스캔들',
       '거룩한 계보', '열혈남아', '강철중: 공공의 적 1-1', '구세주', '예의없는 것
들', '수', '강력 3반',
       '바람의 파이터', '조폭마누라2-돌아온 전설', '천년호', '아 유 레디?', '흑수
선', '정글쥬스',
       '7급 공무원', '리베라메', '유령', '디워', '주유소 습격사건', '거북이 달린
다', '무영검', '전우치',
       '사랑', '비상', '해바라기', '영화는 영화다', '다찌마와리', '무사', '아라한
장풍대작전',
       '우아한 세계', '태풍', '태양은 없다', '대한민국 1%', '육혈포 강도단', '눈에
는 눈 이에는 이',
       '야수', '잠복근무', '조폭마누라3', '천군', '바람', '홍길동의 후예', '아나키
스트', '비천무',
       '구르믈 버서난 달처럼', '보트', '튜브', '뚝방전설', '중천', '웨스트 32번
가', '컷 런스 딥',
       '묻지마 패밀리', '깡패수업 3', '이재수의 난', '양아치어조', '거칠마루', '광
시곡', '감자심포니',
       '강적', '게임오버', '깡패법칙', '건드레스', '건달의 법칙', '폭력써클', '유
감스러운 도시',
       '7인의 새벽', '도시락', '상사부일체', '미션바라바', '엘리시움', '한길수',
'건달본색', '카리스마',
       '흑우', '그리스 로마신화 - 올림포스 가디언 - 기간테스 대역습', '2424', '예
스터데이', '부산',
       '챔피언 마빡이', '율리시즈', '깡패수업 2', '철인 사천왕', '캅링크', '자카르
타', '싸이렌',
       '단적비연수', '1724 기방난동사건', '맨데이트: 신이 주신 임무', '스페어',
'런투유(RUN 2 U)'],
      dtype=object)
```

In [ ]:

```
# 한국 영화 목록 (스릴러)
df_genres.query("1999 <= openYear <= 2010").loc[df_nations["N_한국"]==1].query("G_스릴러 == 1")[
"movieNm"].values
```

Out[ ]:

```
array(['기담', '범죄의 재구성', '송어', '해피엔드', '섬', '제불찰씨 이야기', '살인
의 추억',
       '사람을 찾습니다', '케이티', '친절한 금자씨', '귀 鬼', '악마를 보았다', '추
격자', '복수는 나의 것',
       '궁녀', '마린보이', '작전', '이태원 살인사건', '하녀', '고死 두 번째 이야
기: 교생실습', '해결사',
       '무법자', '황해', '나의 친구, 그의 아내', '더 게임', '세븐 데이즈', '조용한
세상', '평행이론',
       '이끼', '백야행', '시크릿', '리턴', '우리 동네', '그놈 목소리', '극락도 살
인사건', '검은 집',
       '달콤, 살벌한 연인', '남극일기', '두뇌유희프로젝트, 퍼즐', '라라 선샤인',
'김복남 살인사건의 전말',
       '얼굴 없는 미녀', '혈의 누', 'GP 506', '심야의 F.M.', '그림자 살인', '올드
보이', '텔미썸딩',
       '홀리데이', '베스트셀러', '오로라 공주', '고死 : 피의 중간고사', '진실게
임', '용서는 없다', '10억',
       '브레인웨이브', '반가운 살인자', '여고괴담 5', '눈에는 눈 이에는 이', '죽이
고 싶은', '파괴된 사나이',
       '목두기 비디오', '주홍글씨', '핸드폰', '가면', '링', '여의도', '블러디쉐이
크', '뷰티풀 선데이',
       '엠', '이중간첩', '페티쉬', '배니싱 트윈', 'H(에이치)', '해부학 교실', '해
안선', '실종',
       '아티스트', '죽어도 해피엔딩', '쓰리', '써클', '엄지아빠', '플라스틱 트리',
'4인용 식탁', '외톨이',
       '파라다이스빌라', '세이예스', '이것이 법이다', '4교시 추리영역', '이브의 유
혹 - 그녀만의 테크닉',
       '이브의 유혹 - 키스', '이브의 유혹 - 엔젤', '해변으로가다', '삼양동 정육
점', '건축무한 육면각체의 비밀',
       '하피', '신혼여행(身魂旅行)', '사슬', '4요일: 자살을 부르는 요일', '맨데이
트: 신이 주신 임무',
       '트럭', '썸'], dtype=object)
```

In [ ]:

```
# 한국 영화 목록 (멜로/로맨스)
df_genres.query("1999 <= openYear <= 2010").loc[df_nations["N_한국"]==1].loc[df_genres["G_멜로/
로맨스"]==1]["movieNm"].values
```

Out[ ]:

```
array(['미녀는 괴로워', '연리지', '동감', '버스, 정류장', '해피엔드', '사과', '나는 곤경에 처했다!',
       '말죽거리 잔혹사', '너는 내 운명', '멋진 하루', '김종욱 찾기', '파주', '청담보살', '두 번째 사랑',
       '토끼와 리저드', '나는 행복합니다', '잘 알지도 못하면서', '해변의 여인', '신부수업', '쉬리',
       '순정만화', '모던보이', '형사 Duelist', '호우시절', '친구사이?', '헬로우 마이 러브', '하하하',
       '웰 또 그렇게까지', '미인도', '두 여자', '회오리 바람', '은하해방전선', '생활의 발견',
       '내 머리 속의 지우개', '최강로맨스', '불후의 명작', '하녀', '나도 아내가 있었으면 좋겠다',
       '내 마음의 풍금', '...ing', '어깨 너머의 연인', '무림여대생', '페어러브', '방자전',
       '달콤한 거짓말', '내 눈에 콩깍지', '백야행', '지금 사랑하는 사람과 살고 있습니까?', '싸움',
       '아내가 결혼했다', '불꽃처럼 나비처럼', '사랑할 때 이야기하는 것들', '구세주', '연애술사',
       '동갑내기 과외하기 레슨 2', '사랑을 놓치다', '로망스', '달콤, 살벌한 연인', '도마뱀',
       '연애, 그 참을 수 없는 가벼움', '연애의 목적', '내 생애 가장 아름다운 일주일', '러브토크', '천년호',
       '시월애', '바람피기 좋은 날', '사랑니', '초감각 커플', '싸이보그지만 괜찮아', '백만장자의 첫사랑',
       '내 사랑 내 곁에', '맛있는 인생', '사랑 따윈 필요 없어', '용의주도 미스신', '순애보',
       '조금만 더 가까이', '외출', '내 여자친구를 소개합니다', '카라', '황진이', '첫눈', '반두비',
       '가을로', '키친', '사랑', '비상', '결혼은, 미친짓이다', '시라노; 연애조작단', '청춘', '중독',
       '그해 여름', '6년째 연애중', '다세포소녀', '오! 수정', '그 남자의 책 198쪽', '내 깡패 같은 애인',
       '달려라 자전거', '펜트하우스 코끼리', '애인', '청춘만화', '쩨쩨한 로맨스', '누구나 비밀은 있다',
       '화이트 발렌타인', '주홍글씨', '비밀애', '사요나라 이츠카', '로맨틱 아일랜드', '참을 수 없는', '비몽',
       '옥희의 영화', '인터뷰', '쇼쇼쇼(Show Show Show)', '행복', '우리 만난 적 있나요',
       '폭풍전야', '두 얼굴의 여친', '엠', '데이지', '춤추는 동물원', '후회하지 않아', '동백꽃', '거짓말',
       '그녀에게 잠들다', '걸프렌즈', '정', '별빛 속으로', '여름, 속삭임', '죽어도 해피엔딩', '요술',
       '스물넷', '마요네즈', '사랑하니까, 괜찮아', '마들렌', '내 생애 최악의 남자', '천국의 우편배달부',
       '결혼식 후에', '낙원-파라다이스', '라이어', '노랑머리', 'Mr.로빈 꼬시기', '당시', '불타는 내 마음',
       '싸이보그 그녀', '카페 느와르', '내부순환선', '동거, 동락', '경의선', '이파네마 소년',
       '아빠가 여자를 좋아해', '주문진', '오감도', '첫사랑 열전', '서울', '여덟 번의 감정',
       '서서 자는 나무', '깃', '썸머 타임', '마법의 성', '좋은사람있으면 소개시켜줘', '서프라이즈',
       '오버 더 레인보우', '몽중인', '패밀리마트', '사랑활동의 내구성', '나탈리', '킬 미',
       '지구에서 사는 법', '우리 집에 왜 왔니', '슬픔보다 더 슬픈 이야기', '달려라 장미', '내 사랑', '허스',
       '나의 스캔들', '허밍', '내 여자의 남자친구', '여름이 가기 전에', '세기말', '성춘향뎐',
       '삼양동 정육점', '러브(LOVE)', '댄스댄스', '연풍연가', '주노명 베이커리',
```

```
'미인(美人)',
       '단적비연수', '쉿! 그녀에겐 비밀이에요', '흑심모녀', '그녀는 예뻤다', '전투
의 매너', '색다른 동거',
       '기다리다 미쳐'], dtype=object)
```

In [ ]:

```
# 프랑스 영화 목록 (스릴러)
df_genres.query("1999 <= openYear <= 2010").loc[df_nations["N_프랑스"]==1].query("G_스릴러 == 1"
)["movieNm"].values
```

Out[ ]:

```
array(['유(U) - 571', '히든', '테이큰', '더 독', '서브웨이', '돌이킬 수 없는', '페
이지 터너',
       '유령작가', '13 자메티', '안소니 짐머', '크림슨 리버', '웨이트 오브 워터',
       '레지던트 이블 3 - 인류의 멸망', '라르고 윈치', '마터스: 천국을 보는 눈',
'프롬 파리 위드 러브',
       '클로이', '페이싱', '카운테스', '나는 비와 함께 간다', '하이 레인', '미로',
'바디 스내치',
       '엑시트', '뎀', '식스 팩', '멜로디의 미소', '더 시크릿', '보딩 게이트'], dt
ype=object)
```

In [ ]:

```
# 한국 영화 목록 (미국)
df_genres.query("1999 <= openYear <= 2010").loc[df_nations["N_미국"]==1].query("G_스릴러 == 1")[
"movieNm"].values
```

Out[ ]:

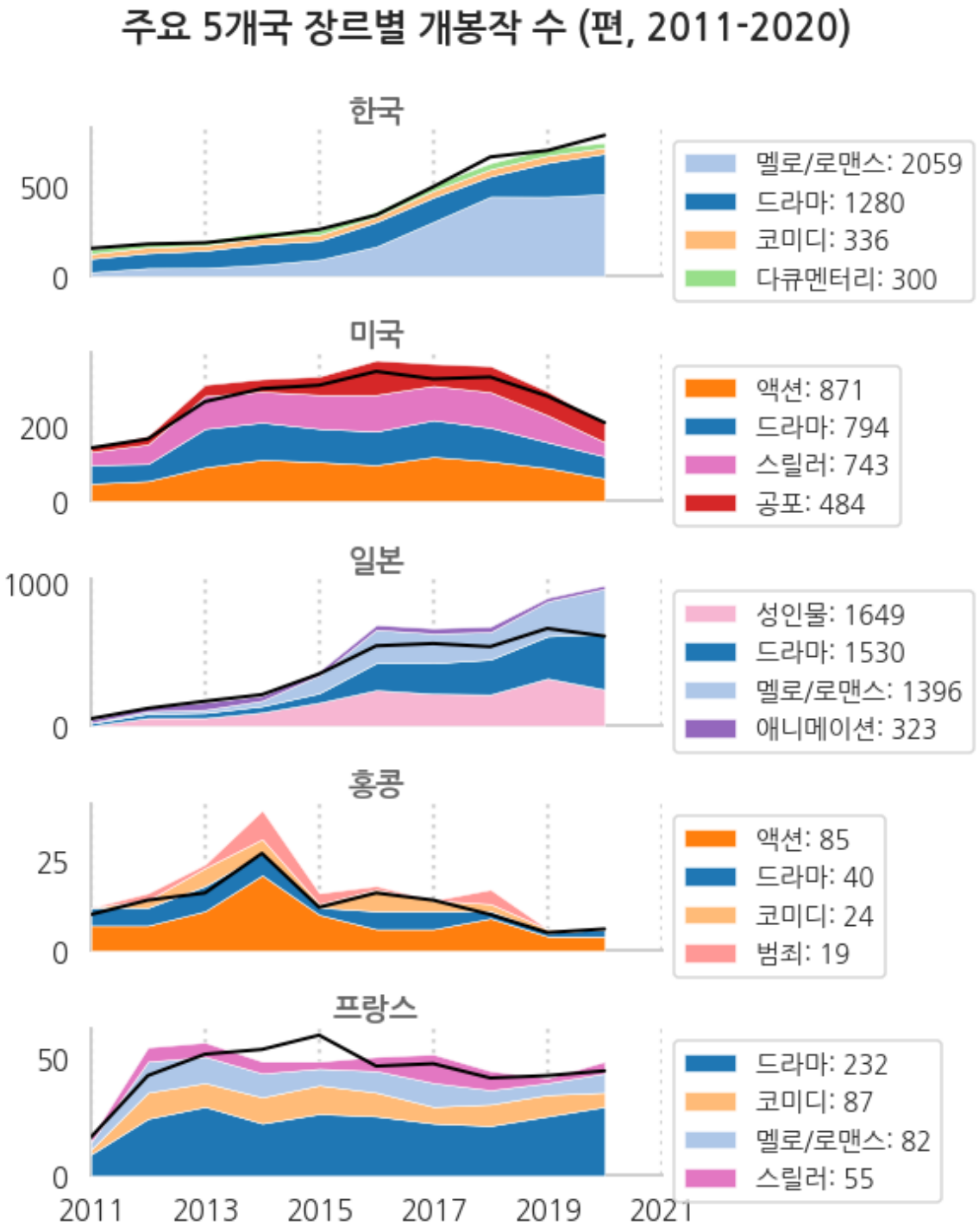
```
array(['시티 오브 갓', '메멘토', '리플리', '스크림', '목요일', '28일 후', '콰이어트 맨',
       '퍼펙트 스트레인저', '노인을 위한 나라는 없다', '언스토퍼블', '데어 윌 비 블러드',
       '람보 4 : 라스트 블러드', 'K-19', '그랜 토리노', '당신이 사랑하는 동안에', '콘스탄틴',
       '파이트 클럽 ', '워', '본 얼티메이텀', '악마가 너의 죽음을 알기전에', '이스턴 프라미스', '더 클럽',
       '미스터 브룩스', '원초적 본능 2', '조디악', '로드 오브 워', '우주전쟁', '파라노말 액티비티',
       '아이즈  와이드 셧', '6번째 날', '인사이더', '패닉 룸', '데블스 워닝', '다크 프린스',
       '언더 프레셔', '차일드 콜렉터', '우먼 오브 더 나잇', '언세드', '너스 베티', '비트 시티 ',
       '터뷸런스 2', '심플 플랜', '킹 코브라', '북 오브 섀도우', '레드 플래닛', '식스티 세컨즈',
       '레인디어 게임', '싸이퍼', '스테이트 오브 플레이', '월 스트리트: 머니 네버 슬립스', 'pm 11:14',
       '고스트 라이더', '그라운드 콘트롤', '미드나잇 미트 트레인', 'H2: 어느 살인마의 가족이야기 ',
       '글래스하우스', '그남자는 거기 없었다', '트리플 X2 : 넥스트 레벨', '그루지 3', '굿바이 러버',
       '쏘우 V', '더 야드', '위 오운 더 나잇', '브레이브 원', '트로미오와 줄리엣', '터치미', '넥스트',
       '씬 시티', '스톰캣처', '스위니 토드 : 어느 잔혹한 이발사 이야기', '보일러룸', '임포스터',
       '유주얼 서스펙트 2', '데스티네이션', '카오스팩터', '펠햄 123', '콰이어트 아메리칸', '글로리아',
       '뮌헨', '텍사스 전기톱 연쇄살인사건 : 0', '대통령을 죽여라', '힛쳐', '실베스타스탤론의 디-톡스',
       '리미츠 오브 컨트롤', '이노센스', '이클립스', '헨드릭스', '할리우드랜드', '함정', '엑소시즘',
       '애스트로넛', '테일러 오브 파나마', '더블타겟', '록 스탁 앤 2 스모킹 배럴즈', '디플로메틱 시즈',
       '기프트', '굿셰퍼드', '9:나인', '캐리2 ', '플래시드 ', '블레어 윗치', '진저브래드맨',
       '처키의 신부 ', '퍼펙트 겟어웨이', '게이머', '이퀄리브리엄', '방콕 데인저러스', '노크:낯선 자들의 방문',
       '에코', '얼터드', '100 피트', '4.4.4', '데드 걸', '데쓰 프루프', '88분', '엘라의 계곡',
       '크레이지', '핑거프린트', '일라이', '킬러 인사이드 미', '엑스페리먼트', '피라냐', '라스트 엑소시즘',
       '스카이 라인', '새벽의 저주', '프리퀀시', '캠퍼스 레전드', '스크림 2 ', '바이러스', '미믹 ',
       '나인스 게이트', '나비효과 2', '써티 데이즈 오브 나이트', '러시 아워 3', '트론: 새로운 시작',
       '투어리스트', '드리프트우드', '오픈 워터', '쏘우', '페이백', '아이덴티티', '어둠의 저주', '리크루트',
       '더 코어', '지퍼스 크리퍼스2', '컷', '웨이 오브 더 건', '레저렉션 2', '스토커', '위험한 유혹',
       '몬스터 볼', '프레일티', '인썸니아', '싸인', '쓰리 데이즈', '캔디 케인', '소울 서바이버',
       '머더 바이 넘버', '하이 크라임', '디스터번스', '모스맨', '돈 세이 워드', '마인드스톰',
       '바닐라 스카이', '아일랜드', '애프터 라이프', '스테이', '스켈리톤 키', '숨바꼭질', '스텔스',
       '맨츄리안 캔디데이트', '프레데터스', '랜드 오브 데드', '나이트 플라이트', '익스펜더블', '솔트',
       '여대생 기숙사', '나이트메어', '데드라인', '폰부스', '셔터 아일랜드', '데이
```

```
브레이커스',
       '미스터 & 미세스 스미스', '인투 더 썬', '포스 카인드', '잉카', '데이비드 게
일',
       '스네이크 온 어 플레인', '울프맨', '더 로드', '화이트 노이즈', '파이어월',
'쏘우 2', '호스티지',
       '쏘우 3', '할로우맨 2', '사일런트 힐', '프레스티지', '모범시민', '디스트릭
트 9', '퍼니 게임',
       '써로게이트', '파이널 데스티네이션 4', '페인터', '하쉬 타임', '언더월드: 라
이칸의 반란',
       '왼편 마지막 집', '포가튼', '오펀: 천사의 비밀', '메디엄', '블러디 발렌타
인', '드래그 미 투 헬',
       '엔드 오브 데이즈', '테이크다운 ', '유턴', '천사와 악마', '더블 스파이',
       '안나와 알렉스 : 두자매 이야기', '더블 크라임', '푸시', '13층', '언데드',
'인터내셔널', '셔틀',
       '옥시즌', '레저베이션 로드', '트랩', '작전명 발키리', '레드 드래곤', 'H20',
'더 헌팅',
       '식스 센스', '딥 블루 씨', '토마스 크라운 어페어 ', '엑시스텐즈', '낯선 사
람에게서 전화가 올 때',
       '하프 라이트', '트레인테러', '럭키 넘버 슬레븐', '프리덤랜드', '인 더 컷',
'블라인드 호라이즌',
       '테이킹 라이브즈', '레저렉션', '황혼에서 새벽까지2', '슬레이어 ', '도망자
3', '인스팅트 ',
       '포세이돈', '다빈치 코드', '오리지날 씬', '패컬티', '사랑보다 아름다운 유혹
', '아웃 오브 타임',
       '실종', '시리아나', '스파이더', '언브레이커블', '파이어웍', '아메리칸 싸이
코', '아트 오브 워',
       '왓쳐', '블레스 더 차일드', '옥토퍼스', '슬링 블레이드', '왓 라이즈 비니
스', '콜렉터',
       '미, 마이셀프 앤드 아이린', '샤프트', '할로우 맨', '지구가 멈추는 날', '수
퍼노바', '맥스 페인',
       '써스펙트', '미션 임파서블 2', '박쥐', '칠 팩터', '스컬스', '마이클 클레이
튼', '쏘우 4',
       '블랙독 ', '헌티드 힐', '카운트다운', '코드제로 ', '에어포스 21', '머더오브
크로우',
       '알렉볼드윈의 컨페션', '인드림스', '스터 오브 에코', '수어싸이드킹', '인 투
딥', '킹덤', '베이컨시',
       '인베이젼', '디스터비아', '브리치', '리턴 투 파라다이스 ', '펄스', '1408',
'폭력의 역사',
       '와일드씽', '저스티스', '하이잭', '슈터', '8미리', '베리배드씽', '저지먼트
데이', '인코그니토',
       '나는 아직도 네가 지난 여름에 한 일을 알고 있다', '밴티지 포인트', '바디 오
브 라이즈', '카운터메져',
       '이글 아이', '스트레이', '바빌론 A.D.', '인 블룸', '미러', '비치', '비상계
엄', '싸이코',
       '썸머 솔스티스', '캅 랜드', '비상근무', '본 콜렉터', '부트캠프', '원티드',
'룰스 오브 인게이지먼트',
       '디 아이', '레드 레터', '리핑 10개의 재앙', '선샤인', '넘버23', '스모킹 에
이스', '한니발 라이징',
       '블러드 다이아몬드', '에라곤', '패솔로지', '킬위드미', '스트리트 킹', '내가
숨쉬는 공기', '어웨이크',
       '디펜스', '디얼리 디보티드', '더 셀', '더 길티', '클로버필드', '포스 엔젤',
'데블 스네이크',
       '발렌타인', '팔콘 다운', '사보타지 2', '스워드피쉬', '미이라2', '아나토미',
'패스워드',
       '더 기프트', '스내치', '황혼에서 새벽까지3', '3000마일', '15분', '더 재킷',
'더 로드',
       '런어웨이', '메이', '스파이더 게임', '콜드 크릭'], dtype=object)
```

3.7.5. 2011-2020

In [ ]:

```
plot_time_GN(2011, 2020, topN=4)
```



In [ ]:

```
# 일본 영화 목록 (성인물)
print(df_genres.query("2011 <= openYear <= 2020").loc[df_nations["N_일본"]==1].query("G_성인물 ==
= 1")["movieNm"].values[:100])
```

['육덕녀의 황홀한 서비스' '대학선배에게 뺏긴 약혼녀' '나만의 판타지 섹스2' '나만의
판타지 섹스'
 '야간학교 : 음란한 누나들의 아찔한 수업' '흑심을 보인 시아버지' '첫키스부터 XX까
지 엄마친구' '여자마사지사는 정말 주는가'
 '술취한 직장후배와 원나잇 가능' '선넘은 F컵 물리치료사' '육덕 여직원과 러브호텔에
서...' '신인 AV 여배우의 첫 촬영'
 '피부좋은 육덕 새엄마' '첫사랑인 숙모에게 고백한 밤' '시아버지의 흑심에 불붙은 여
름밤' '너무나 커버린 새아들이 좋아'
 '남편몰래 들이대는 새아들' '매일 밤 엿보기: 대놓고 보여주는 불륜 유부녀' '야한 첫
경험' '시아버지의 여자 2'
 '낯선 남자와 청순 글래머' '아버지의 여자' '침대에서 만난 가정부' '소금맛이 나는
여자' '사촌누나의 성상담'
 '불륜축제-깊숙한 그곳에' '가정부가 야동처럼 보일때' 'G컵 베이글의 능숙한 몸놀림'
'항상 열려있는 육체'
 '적극적인 G컵 접대녀' '섹스 온천: 전설의 여관 여주인' '몸으로 빼앗은 여동생의 남
자친구'
 '불량 주부: 부하직원과 바람난 내 아내' '니가 사는 그 집: 불륜 전세 계약서' '동생
의 어린 아내-무삭제판'
 '자세가 좋은 제수씨' '내 여자가 된 선생님 2' '음란 택배기사와 거유 사모님'
 '한번 맛보면 참을 수 없는 육덕 유부녀 무삭제판' '사장에게 실제 흥분하는 육덕미인
무삭제판' '친구엄마의 승부속옷'
 '의동생의 팬티를 벗길 때' '유카타를 벗고 홀딱 마사지' '아들 여친이 샤워를 하고 있
다' '새엄마는 오늘도 오르가즘'
 '음란문학선생' '여중개인과 폭풍우 치는 밤' '핫바디-무삭제판' '아내의 남자친구-무
삭제판'
 '위에서 완벽한 나의 스페인 여자친구' '순종적인 음란한 여자' '농밀한 그녀의 서비
스' '가정교사의 참교육'
 'AV배우의 은밀한 사생활' '음탕함에 길들여진 유부녀' '음란 유부녀의 비밀 SNS' '신
입 여사원 길들이기'
 '술 취해 잠든 귀여운 여친 무삭제판' '자세가 좋았던 전 여친' '욕구불만 사모님'
 '회사직원과 남편 몰래 흥분하는 청순녀 무삭제판' '사장에게 유혹당하는 여배우지망생
무삭제판' '참지못해 터진 유부녀'
 '정말 있었던 보험계약' '봐주길 원한 친구엄마' '리얼 남편상사의 요구' '결혼직전 여
친엄마와' '회사 후배와 몰래하는 육덕미인'
 '이 여자는 나의 노예' '연하남과 첫불륜-인생절정' '아이미의 핫한 새엄마' '아내가
늦은 날 - 장모님'
 '실제로 흐느끼는 육덕 왕가슴 아내' '미즈노의 가터벨트' '부장의 젊은 아내-무삭제
판' '음란한 시아버지와 며느리'
 '내 첫경험은 옆집 아줌마' '여선생의 금지된 방문교육' '남편 부하에게 타락되어 젖는
육체' '배관공과 유부녀의 유혹'
 '내 여자가 된 선생님' '집주인 물건 빨아주는 세탁가정부' '음란한 과외 선생님' '유
부녀 집구멍으로 질속배달 택배기사'
 '욕망에 눈뜬 여관 아줌마' '음란착각 에미리짱- 성감젝크' '여상사와 단둘이 출장온
호텔' '숫처녀 같은 엄마친구'
 '새엄마의 극락목욕탕' '노골적인 폭유녀의 노브라공격' '숨도 못 쉬게 깊이 들어온 동
창' '여대생의 AV 데뷔기 2'
 '베이글녀의 진지한 데이트' '핫바디' '시아버지의 여자-무삭제판' '젊은 새엄마의 속
사정' '아내의 남자들'
 '전 남친에 빼앗긴 아내' '풍만 형수의 노예계약 - 음모론' '폭유도발클럽 - 제복편']

In [ ]:

```
# 한국 영화 목록 (성인물)
print(df_genres.query("2011 <= openYear <= 2020").loc[df_nations["N_한국"]==1].query("G_성인물 ==
= 1")["movieNm"].values)
```

['남편 친구' '네 번째 엄마' '스와핑 : 자매의 남자들 무삭제' '거짓말 : 친구엄마 무삭제' '애첩' '신입 여사원'
 '착한 유부녀들' '애인공유' '정사 : 타인의 아내' '애인의 가슴 큰 친구' '모델 박아라 감독판' '스와핑, 그 위험한 섹스'
 '불륜 동창회 3' '불륜여행 : 여친의 언니' '빚투: 부부교환 무삭제' '막장자매 클라쓰' '처제의 은밀한 맛' '캠핑촌 아내'
 '처제길들이기' '볼빨간 형수님' '볼빨간 여동생' '비뇨기과 여의사들 3' '착한 형수 : 두 부부가 이혼한 날 무삭제'
 '착한 형수 : 두 부부가 이혼한 날' '아는사이' '팽말숙 : 실전주의' '비키니 바 : 맛있는 서비스' '못말리는 사촌동생'
 '아들의 여친 아빠의 여친' '스와핑 : 두 부부의 이혼여행 무삭제' '형수에 반하다' '산부인과 남의사들'
 '불륜의 세계 무삭제판' '여동생과 은밀한 거래' '바람난 아내의 비밀' '내 아내가 아닌' '여대생의 은밀한 불륜' '페이섹스'
 '탐스러운 장모님' '무엇이든 도와드립니다' '주부남편의 발기찬 바깥생활 무삭제' '어린 와이프 무삭제' 'A급며느리'
 '빚투: 부부교환' '바람난 처제' '거짓말 : 친구엄마' '완벽한 스와핑' '애인이 생기면 하고 싶은 일' '엄마의 직업 2'
 '싱글의 세계' '여배우들: 섹스 오디션 무삭제' '아이돌 섹스: LA 교포녀 무삭제' '여대생 누드모델' '맛있는 너가 합격'
 '섹스가 좋은 그 여자' '해풍 : 바람난 여자' '위험한 과외' '섹스인듯 촬영인듯 섹스하는 너'
 '리얼섹스돌 : 사용설명서 무삭제' '착한 숙모 : 삼촌이 출장 간 날' '스와핑 : 자매의 남자들' '최면 무도 클럽'
 '오빠는 음식은 가려도 여자는 안가려 무삭제' '그녀의 모유는 백신' '친구엄마의 욕구불만'
 '욕구불만 자매들 : 형부가 출장 간 날' '착한 처형 : 마지막 여행' '스와핑 : 두 부부의 이혼여행' '19금 어린아내'
 '자매의 은밀한 동거 감독판' '예쁜식모 감독판' '예비며느리' '주부딜러 ' '여사장 후리꾼' '딴놈의 아내'
 '주부남편의 발기찬 바깥생활' '맛있는 과외누나2' '하이에나' '예민한 모녀' '위험한 정사-덫'
 '간기녀 - 간통을 기다리는 여자' '하녀엄마' '여관발이' '옆방에 일본여자가 싼다' '여배우들: 섹스 오디션'
 '세자매 은밀한 섹파' '관전, 섹스 과외' '성인 신고식' '맛있는 섹스 그리고 언니들' '담배가게 아가씨 : 나를 당겨 주세요'
 '아이돌 섹스: LA 교포녀' '싸와디캅 원정섹스2' '싸와디캅 원정섹스' '내연녀' '방콕 어린형수' '시누이의 맛'
 '스와핑 : 선배의 아내 ' '배달노출2 : 초대남과 좋아죽는 와이프' '오빠는 음식은 가려도 여자는 안가려' '그녀와의 스와핑'
 '여린여동생' '엄마의 비밀친구' '음란한 과외' '마사지 걸의 유혹' '환각섹스' '누드모델의 목적' '어린 관리사 무삭제'
 '사돈끼리 그거슨 아니지' '섹시한 가정부' '배달노출 : 알몸으로 유혹하기' '스와핑 부부' '불량탐정 : 리로드 무삭제판'
 '자매전쟁' '무한정사' '불량탐정: 먹이사슬' '어린 관리사 ' '잠 못 이루는 섹스2 무삭제판' '스와핑 동창회'
 '불여시:일본 여직원의 질투 ' '내 아내는 왜! 바람이 났을까?' '스토커 : 페르소나' '스토커 1/2'
 '보도드라이버: 풀코스 무삭제' '보도드라이버:떡마차 무삭제' '일반인 속초' '보도드라이버: 풀코스' '세부야놀자' '살맛'
 '보도드라이버: 풀코스 무삭제판' '아내의 음란한 유혹 ' '자매의 노예' '완벽한 타인 : 비밀의 스와핑 ' '형수의 엉덩이'
 '비가오면 하고싶다' '복상사 미망인을 탐하는 남자들' '내 아내를 과장이 후렸다' '보도드라이버: 떡마차 무삭제판'
 '내방에 일본여자가 있다 무삭제판' '화끈한 룸메이트 무삭제판' '여성친구 ' '긴섹충 ' '음란한 여자들의 수다'
 '일본식 아내 길들이기 ' '보도드라이버: 떡마차' '누나의 비디오' '리얼라이저' '만화방' '형수의 직업' '올리고당 더 무비'
 '썸 - 은밀한 이야기' '러브 슬레이트' '나는 야한 여자가 좋다' '플라스틱 섹스' '노랑머리 - 플라스틱 섹스'
 '전망 좋은 차- 맛있는 섹스' '맛있는 섹스 - 인연' '창녀의 성적유희' '마지막 중독'

```
 '참을 수 없는 성적유희 감독판'
 '48시간의 일탈' '야색야동']
```

In [ ]:

```
# 한국 영화 목록 (멜로/로맨스)
print(df_genres.query("2011 <= openYear <= 2020").loc[df_nations["N_한국"]==1].loc[df_genres["G_멜로/로맨스"] != 1]["movieNm"].values[:100])
```

```
['평범한 날들' '못' '캐리tv 러브콘서트 더 무비' '오! 문희' '이태원' '담보' '바람의
춤꾼' '국가대표 2'
 '카이: 거울 호수의 전설' '미씽: 사라진 여자' '동주' '미쓰 와이프' '신촌좀비만화'
'앨리스 죽이기' '마지막 휴가'
 '북촌방향' '부활: 그 증거' '2017 동명이인 프로젝트' '동명이인 프로젝트 시즌2' '감
쪽같은 그녀' '나를 구하지 마세요'
 '증발' '울지마 톤즈 2 : 슈크란 바바' '젊은이의 양지' '메기' '황무지 5월의 고해'
'런 보이 런' '아빠는 예쁘다'
 '검객' '타클라마칸' '바캉스' '우리 지금 만나' '어서오시게스트하우스' '소리꾼' '파
파로티'
 '극장판 샤이닝스타:새로운 루나퀸의 탄생!' '이웃사촌' '나는 고양이로소이다' '미스
터 주: 사라진 VIP' '마돈나' '카트'
 '돌멩이' '기도하는 남자' '루비' '남매의 여름밤' '남편 친구' '마이 리틀 히어로'
'잔칫날' '부활' '나만 없어 고양이'
 '어린 의뢰인' '미스터 보스' '열혈형사' '범털' '후쿠오카' '프랑스여자' '럭키 몬스
터' '욕창'
 '디아스포라의 노래 : 아리랑 로드 ' '헤로니모' '선샤인 패밀리' '기억의 전쟁' '웰컴
투 X-월드' '69세'
 '항거:유관순 이야기' '기묘한 가족' '프리즈너' '스웨그' '얼굴들' '세트플레이' '낯
손님' '마티나' '이별유예, 일주일'
 '정의심판' '용루각: 비정도시' '나의 노래는 멀리멀리' '나는보리' '목숨' '길위에서'
'위로공단' '뮤직 앤 리얼리티 '
 '개 같은 것들' '없던 일' '앙상블' '내언니전지현과 나' '겨울밤에' '담쟁이' '나의
특별한 형제' '기기괴괴 성형수'
 '내가 죽던 날' '국도극장: 감독판' '레슬러' '사라진 시간' '파수꾼' '스틸 플라워'
'우리집' '춘천, 춘천'
 '요가학원 : 죽음의 쿤달리니' '뺑반' '애비규환']
```

In [ ]:

```
# 다른 장르에 숨어있는, 성인물로 추정되는 영화들
df_genres.query("2011 <= openYear <= 2020").loc[df_nations["N_한국"]==1].loc[df_genres["G_성인물"] != 1].loc[df_genres["movieNm"].str.contains("섹스")]
```

Out[ ]:

	movieCd	movieNm	openYear	G_SF	G_가족	G_공연	G_공포	G_기타	G_다큐멘터리	G_드라마	G_멜로/로맨스	G_뮤지컬	G_미스터리	G_범죄
1689	20203543	섹스 소녀 10	2020	0	0	0	0	0	0	0	1	0	0	0
1695	20207755	여자 동창들 : 친구 오빠와 섹스한 날 무삭제	2020	0	0	0	0	0	0	0	1	0	0	0
1700	20203154	섹스 소녀 8	2020	0	0	0	0	0	0	0	1	0	0	0
1705	20203127	섹스 소녀 9	2020	0	0	0	0	0	0	0	1	0	0	0
1924	20203155	18 출사녀 화진의 미친 섹스	2020	0	0	0	0	0	0	0	1	0	0	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
16245	20134705	맛있는 사랑공식-섹스	2013	0	0	0	0	0	0	1	1	0	0	0
16742	20130248	거짓말 섹스가 좋아	2013	0	0	0	0	0	0	1	1	0	0	0
17356	20123344	전망 좋은 방-맛있는 섹스	2012	0	0	0	0	0	0	1	1	0	0	0
17911	20122449	맛있는 섹스, 맛있는 상상	2012	0	0	0	0	0	0	1	1	0	0	0
17912	20122025	금지된 섹스 불륜2	2012	0	0	0	0	0	0	1	1	0	0	0

140 rows × 25 columns

In [ ]: