**- BabyNames**

pivot table을 활용해서 각 년도 별로 births가 10 미만인 이름들의 갯수 (column은 성별로)

\* pivot table이란?

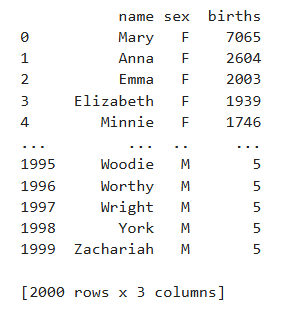
데이터를 행과 열로 구성하여 특정 조건에 따라 데이터를 집계하고 요약하는 기능을 제공하는 것

import numpy as np

import pandas as pd

babynames = pd.read\_csv('datasets/datasets/babynames/yob1880.txt', names=['name', 'sex', 'births'])

print(babynames)



→ read\_csv 함수를 이용해 데이터 읽어온 후, 확인하기

years = range(1880, 2011)

→ 1880년부터 2011년까지 연도 범위 설정

piv = [pd.read\_csv(f'datasets/datasets/babynames/yob{year}.txt', names=['name','sex','births']).assign(year=year) for year in years]

→ 각 연도에 대해 파일 경로를 문자열 포매팅(f-string)을 사용하여 생성

3개의 열이 있다고 가정하며, 순서대로 name, sex, births라는 이름을 지정

assign(year=year)는 데이터프레임에 새로운 열 year를 추가하고, 그 값은 현재 연도로

설정. 예를 들어, 1880년의 경우 모든 행의 year 값이 1880으로 설정

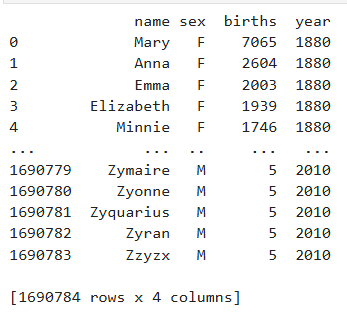
names = pd.concat(piv, ignore\_index=True)

print(names)

→ concat은 여러 데이터프레임을 하나의 데이터프레임으로 결합하는 함수

names에 piv 결합하여 새로운 데이터프레임 생성

ignore\_index = True는 기존 인덱스를 무시하고 0부터 시작하여 순차적으로 증가하라는 뜻



def sumbir(x):

if(x.births < 10):

return 1

elif(x.births >= 10):

return 0

→ sumbir이라는 함수를 지정하고, x의 births가 10보다 작으면 1을, 10보다 크면 0을 반환

names['sum'] = names.apply(lambda x: sumbir(x),axis='columns')

→ name에 sum이라는 새로운열 추가

axis=’columns’ 는 행 단위로 함수 적용한다는 의미

‘apply 함수는 데이터프레임의 각 행을 입력으로 받아 sumbir 함수를 적용하고, 반환된

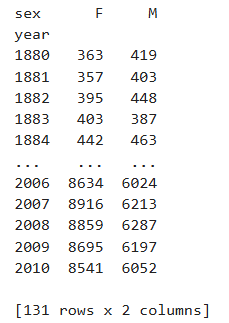
값을 새로운 열 sum에 저장

df = names.pivot\_table(values='sum', index='year', columns='sex', aggfunc=sum)

print(df)

→ sum 열을 활용해서 pivot table 생성

각 연도를 인덱스로 활용 / 성별은 각 열을 나타냄



import matplotlib.pyplot as plt

plt.figure(figsize=(14, 7))

plt.plot(df.index, df['F'], label='Female', color='blue')

plt.plot(df.index, df['M'], label='Male', color='red')

plt.xlabel('Year')

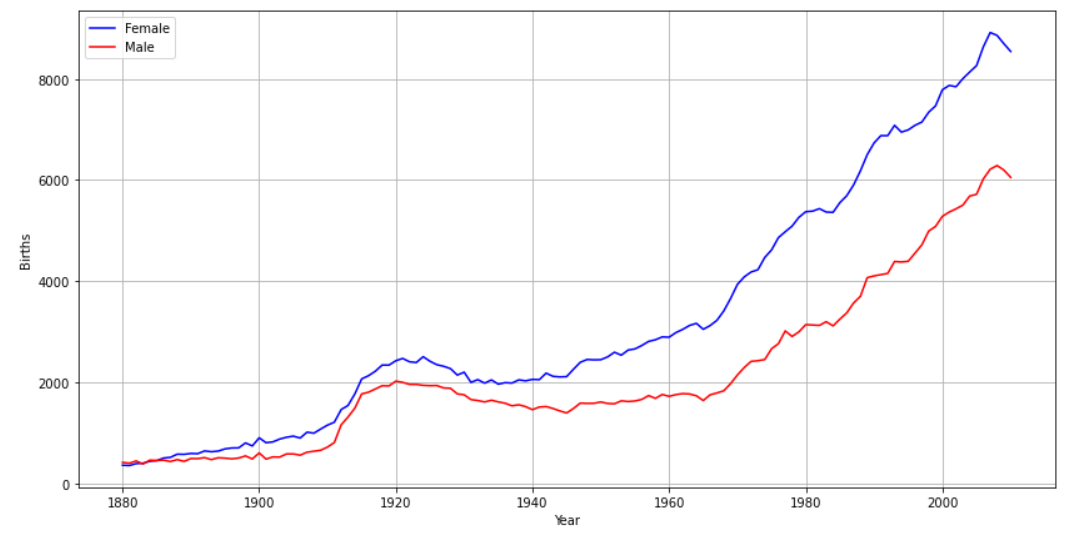
plt.ylabel('Births')

plt.legend()

plt.grid(True)

plt.show()

→ 시각화하기 위한 코드 / 성별로 색깔을 달리하여 표현



=> 결론

최근으로 올수록 birth가 10미만인 인구 수가 증가하고 있다.

**- MovieLen**

1.

- diff 값에 절댓값 씌우기

pd.options.display.float\_format = '{:.4f}'.format

unames = ['user\_id', 'gender', 'age', 'occupation', 'zip']

users = pd.read\_table('datasets/movielens/users.dat', sep='::', header=None, names=unames)

rnames = ['user\_id', 'movie\_id', 'rating', 'timestamp']

ratings = pd.read\_table('datasets/movielens/ratings.dat', sep='::', header=None, names=rnames)

mnames = ['movie\_id', 'title', 'genres']

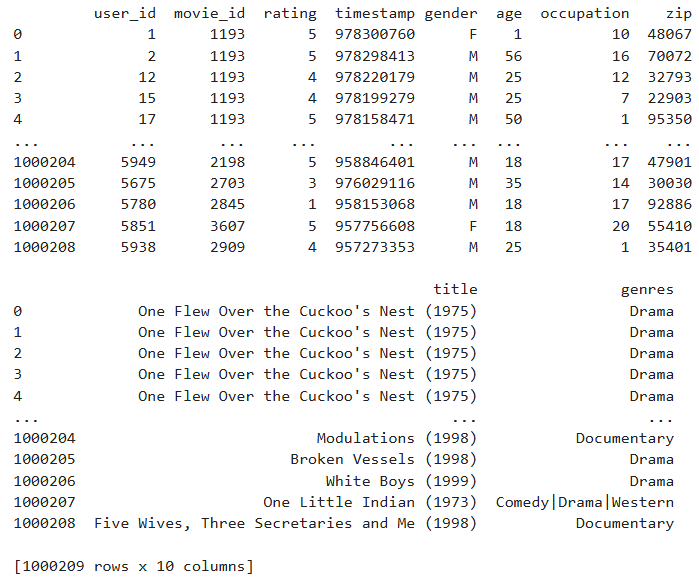
movies = pd.read\_table('datasets/movielens/movies.dat', sep='::', header=None, names=mnames)

data = pd.merge(pd.merge(ratings, users), movies)

print(data)

→ merge함수는 두 개의 데이터프레임을 병합할 수 있으므로 ratings와 users를 먼저 병합한

후, movies까지 병합하여 마무리

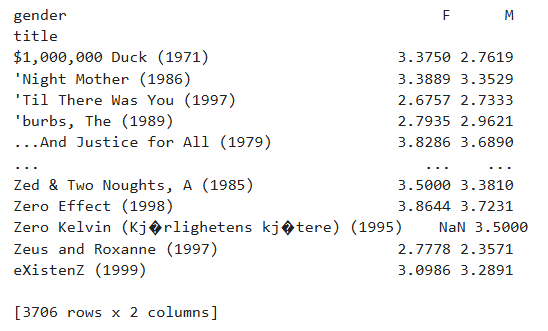


mean = data.pivot\_table('rating', index='title', columns='gender', aggfunc='mean')

print(mean)

→ rating값이 pivot table의 값으로 사용됨 / 인덱스는 title이고, gender로 열을 구성

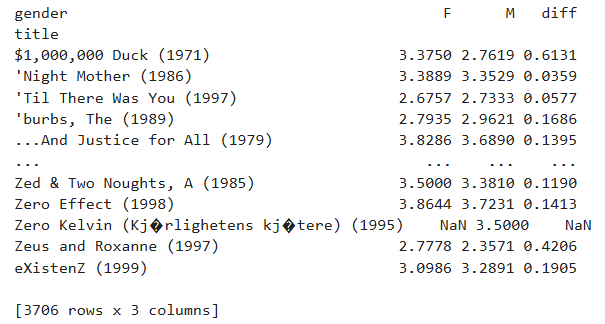
집계함수는 mean으로 평균을 계산함



mean['diff'] = abs(mean['M'] - mean['F'])

print(mean)

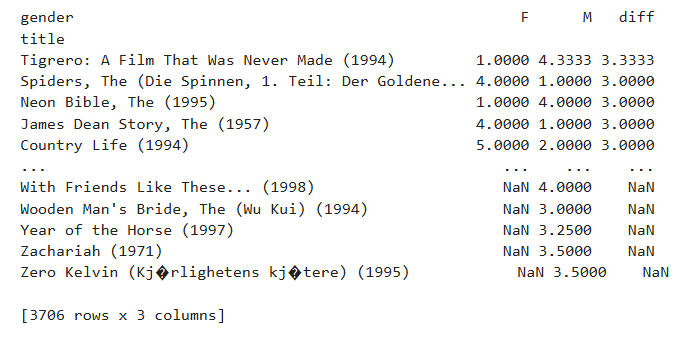
→ 각 성별간의 평점 평균 격차를 절댓값 abs 를 이용해 나타냄 / 계산 결과 diff 열에 추가



mean2 = mean.sort\_values(by='diff', ascending=False)

print(mean2)

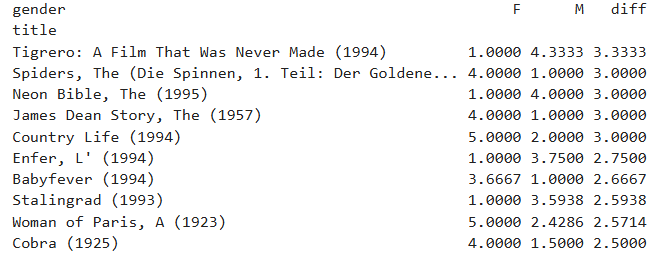
→ diff열을 기준으로 내림차순 정렬 (ascending = True 는 오름차순)



print(mean2[:10])

# mean2.head(10)

→ 상위 10개 데이터 출력 / 아래 # 코드를 사용하면 데이터프레임 형식으로 출력 가능

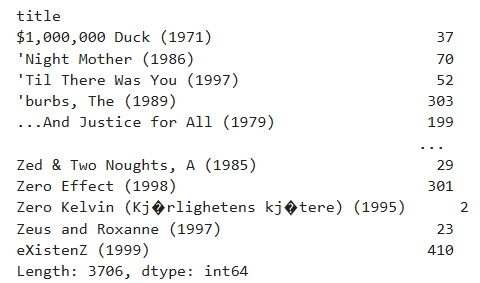


2. 전체 영화중 평점을 준 갯수가 100개 이상인 영화는 전체의 몇%인가

rating = data.groupby('title').size()

print(rating)

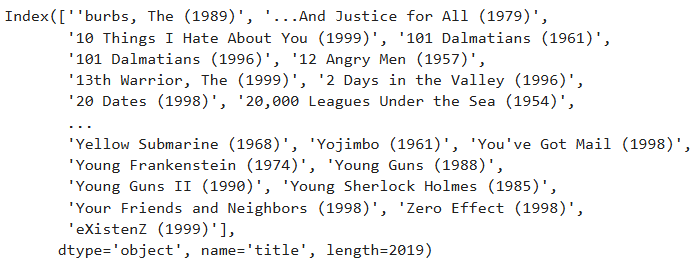
→ title 열을 기준으로 같은 제목을 가진 영화끼리 그룹화 / size 를 이용해 각 크기를 계산



rating2 = rating.index[rating >= 100]

print(rating2)

→ rating 중에 값이 100 이상을 만족하는 값들만 출력



total\_movies = len(movies.title)

rating3 = (len(rating2) / total\_movies) \* 100

print(rating3)

→ movies에서 title 열 길이를 계산하여 total\_movies 에 저장

비율을 계산하기 위해 rating2 를 total\_movies로 나눈 후, 100을 곱하여 결과 도출



3. timestamp와 평점이 상관이 있는지 확인해봅니다. 즉 timestamp가 높을수록 영화 평점이 높은지 아니면 이 2개는 관계가 있는건지 확인하고 이 상관관계를 matplot을 활용해 표현해 봅니다.

- timestamp 와 평점 사이의 상관관계를 알아보기 위해 상관계수 계산

corr = data['rating'].corr(data['timestamp'])

print('상관관계: %.4f' %(corr))

→ corr 을 이용하여 두 변수 간의 상관관계 계산 / 상관계수는 -1에서 1사이의 값이며 0에 가까울수록 서로 상관관계가 없음을 의미



- scatter 를 이용해 시각화

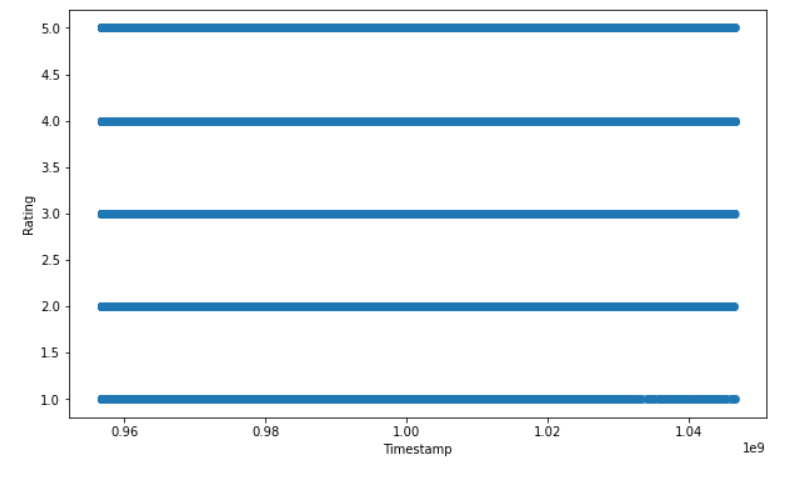
plt.figure(figsize=(10, 6))

plt.scatter(data['timestamp'], data['rating'], alpha=0.5)

plt.xlabel('Timestamp')

plt.ylabel('Rating')

plt.show()



→ 상관계수와 그래프 모두 서로 상관관계가 없음을 나타냄