

Analysis on movie names

GroupLens Research has collected and made available rating data sets from the MovieLens web site (<https://grouplens.org/datasets/movielens/> (<https://grouplens.org/datasets/movielens/>)).

The data sets were collected over various periods of time, depending on the size of the set.

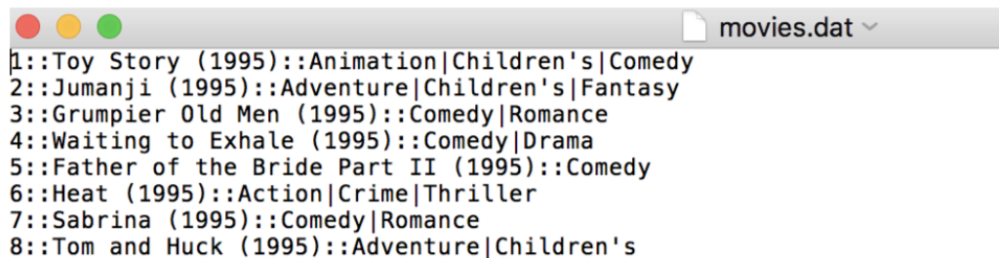
MovieLens 1M Dataset: collected from users of MovieLens in the late 1990s and early 2000s.

1 million ratings from 6000 users on 4000 movies.

It's spread across 3 tables: ratings, user information, and movie information. You can find users.dat, movies.dat, ratings.dat file in the ml-1m folder.

In this example we will work only on movies.dat file

Task: modify movies.dat file so that year information embedded in the title column is written as a separate column.



```
1::Toy Story (1995)::Animation|Children's|Comedy
2::Jumanji (1995)::Adventure|Children's|Fantasy
3::Grumpier Old Men (1995)::Comedy|Romance
4::Waiting to Exhale (1995)::Comedy|Drama
5::Father of the Bride Part II (1995)::Comedy
6::Heat (1995)::Action|Crime|Thriller
7::Sabrina (1995)::Comedy|Romance
8::Tom and Huck (1995)::Adventure|Children's
```



```
1::Toy Story ::Animation|Children's|Comedy::1995
2::Jumanji ::Adventure|Children's|Fantasy::1995
3::Grumpier Old Men ::Comedy|Romance::1995
4::Waiting to Exhale ::Comedy|Drama::1995
5::Father of the Bride Part II ::Comedy::1995
6::Heat ::Action|Crime|Thriller::1995
7::Sabrina ::Comedy|Romance::1995
8::Tom and Huck ::Adventure|Children's::1995
9::Sudden Death ::Action::1995
10::GoldenEye ::Action|Adventure|Thriller::1995
```

First read movies.dat file

In [1]:

```
moviefilename="movies.dat"
moviefile=open(moviefilename)
movies=moviefile.readlines()
```

- Normally when reading a file you don't have to specify encoding explicitly.
- However in our case there was a character which was not understood by the standart "utf-8" format.

try reading with encoding latin-1

In [2]:

```
moviefilename="movies.dat"
moviefile=open(moviefilename,encoding="latin-1")
movies=moviefile.readlines()
```

check how many lines you have

In [3]:

```
len(movies)
```

Out[3]:

3883

operation for all the movie names using for loop

In [4]:

```
import re
pattern=r"\\((\\d+)\\)"
outputfilename="movies-modified.dat"
outputfile=open(outputfilename,"w")
for movie in movies:
    movie_id,movie_title,movie_genre=movie.strip().split("::")
    year=re.findall(pattern,movie_title)[0]
    movie_title_modified=re.sub(pattern,"",movie_title) #Düzenlenmiş olarak bir string döndür
    print(movie_id,movie_title_modified,movie_genre,year,sep="::",file=outputfile)

outputfile.close()
```

Task: Find the year that has highest number of movies in movies-modified.dat file.

Movies-modify.dat dosyasında en fazla film bulunan yılı bulun.

First read the file and create a list for year information

In [5]:

```
moviefilename="movies-modified.dat"
moviefile=open(moviefilename)
movies=moviefile.readlines()
yearlist=[]

for movie in movies:
    movie_id,movie_title,movie_genre,year=movie.strip().split("::")
    yearlist.append(year)
```

In [6]:

```
yearlist
```

```
'1995',  
'1996',  
'1996',  
'1996',  
'1997',  
'1996',  
'1995',  
'1996',  
'1996',  
'1996',  
'1996',  
'1996',  
'1996',  
'1996',  
'1996',  
'1996',  
'1996',  
'1996',  
'1996',  
'1982',  
'1996',  
'1996',
```

in this yearlist, we should calculate the frequency of each year

In [7]:

```
from collections import Counter  
c=Counter(yearlist)
```

In [8]:

c

Out[8]:

```
Counter({'1995': 342,  
        '1994': 257,  
        '1996': 345,  
        '1976': 21,  
        '1993': 165,  
        '1992': 102,  
        '1988': 69,  
        '1967': 24,  
        '1964': 16,  
        '1977': 22,  
        '1965': 20,  
        '1982': 50,  
        '1962': 20,  
        '1990': 77,  
        '1991': 60,  
        '1989': 60,  
        '1937': 11,  
        '1940': 19,  
        '1969': 18,  
        '1981': 43,  
        '1973': 29,  
        '1970': 16,  
        '1960': 15,  
        '1955': 19,  
        '1956': 19,  
        '1959': 22,  
        '1968': 22,  
        '1980': 41,  
        '1975': 21,  
        '1986': 104,  
        '1948': 12,  
        '1943': 10,  
        '1963': 25,  
        '1950': 14,  
        '1946': 13,  
        '1987': 71,  
        '1997': 315,  
        '1974': 28,  
        '1958': 22,  
        '1949': 10,  
        '1972': 22,  
        '1998': 337,  
        '1933': 7,  
        '1952': 11,  
        '1951': 12,  
        '1957': 20,  
        '1961': 19,  
        '1954': 15,  
        '1934': 7,  
        '1944': 13,  
        '1942': 13,  
        '1941': 11,  
        '1953': 14,  
        '1939': 11,  
        '1947': 14,
```

```
'1945': 11,  
'1938': 6,  
'1935': 6,  
'1936': 8,  
'1926': 8,  
'1932': 7,  
'1930': 7,  
'1971': 26,  
'1979': 32,  
'1966': 12,  
'1978': 30,  
'1985': 65,  
'1983': 35,  
'1984': 60,  
'1931': 7,  
'1922': 2,  
'1927': 6,  
'1929': 3,  
'1928': 3,  
'1925': 6,  
'1923': 3,  
'1999': 283,  
'1919': 3,  
'2000': 156,  
'1920': 2,  
'1921': 1})
```

print the top-10 years having the highest number of movies

In [10]:

```
for year, freq in c.most_common(5):  
    print(year, ': ', freq)
```

```
1996 : 345  
1995 : 342  
1998 : 337  
1997 : 315  
1999 : 283
```

Task: Find the frequency of each genre type and print top-10 most frequent ones

- we will do similar operation for movie genres.
- Note that movie genres are separated by “|” character.

First read genres into a genrelist

In [11]:

```
moviefilename="movies-modified.dat"
moviefile=open(moviefilename)
movies=moviefile.readlines()
genrelist=[]
for movie in movies:
    movie_id,movie_title,movie_genre,year=movie.strip().split("::")
    movie_genres=movie_genre.split("|")
    for genre in movie_genres:
        genrelist.append(genre)
```

Now calculate the frequency

In [12]:

```
from collections import Counter
c=Counter(genrelist)
for genre, freq in c.most_common(10):
    print(genre, ': ', freq)
```

```
Drama : 1603
Comedy : 1200
Action : 503
Thriller : 492
Romance : 471
Horror : 343
Adventure : 283
Sci-Fi : 276
Children's : 251
Crime : 211
```

In []: