### Ime, priimek:

LV07: Priporočilni sistem, podobnost vsebin

Namen vaje:

* Spoznati podatke o ocenah filmov
* Algoritem izračuna podobnosti vsebin (filmov) na osnovi ocen
* Generiranje priporočila podobnih filmov za izbrani film

# Analiza podatkov

## Podatkovni set MovieLens

MovieLens data sets were collected by the GroupLens Research Project

at the University of Minnesota.

This data set consists of:

\* 100,000 ratings (1-5) from 943 users on 1682 movies.

\* Each user has rated at least 20 movies.

\* Simple demographic info for the users (age, gender, occupation, zip)

The data was collected through the MovieLens web site

(movielens.umn.edu) during the seven-month period from September 19th,

1997 through April 22nd, 1998. This data has been cleaned up - users

who had less than 20 ratings or did not have complete demographic

information were removed from this data set. Detailed descriptions of

the data file can be found at the end of this file.

## Nalaganje podatkov

Uvozimo podatke iz datoteke v data frame

import numpy as np

import pandas as pd

import sklearn.cross\_decomposition

# Uvozi za vizualizacijo

import matplotlib.pyplot as plt

import seaborn as sns

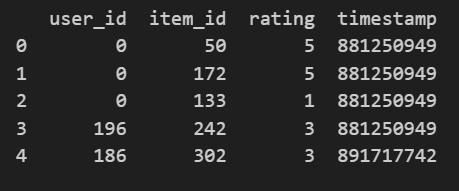
sns.set\_style('white')

%matplotlib inline

column\_names = ['user\_id', 'item\_id', 'rating', 'timestamp']

df = pd.read\_csv('u.data', sep='\t', names=column\_names)

Preglej in izpiši nekaj vrstic podatkov:



Preberemo naslove filmov, in jih dodamo v podatkovni objekt

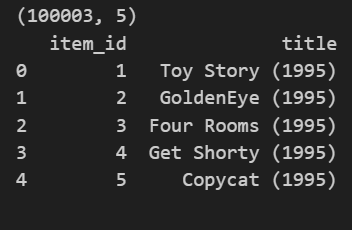
movie\_titles = pd.read\_csv("Movie\_Id\_Titles")

movie\_titles.head()

df = pd.merge(df,movie\_titles,on='item\_id')

print(df.shape)

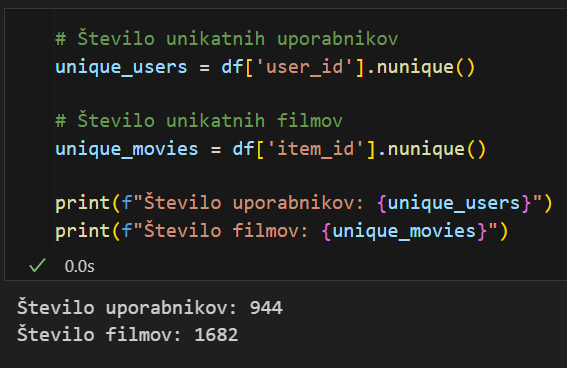
Izpiši nekaj vrstic podatkov:



## Vprašanje 1

Ugotovi, koliko uporabnikov in koliko filmov obsegajo podatki. Uporabi metodo nunique().

Koda in rezultat:

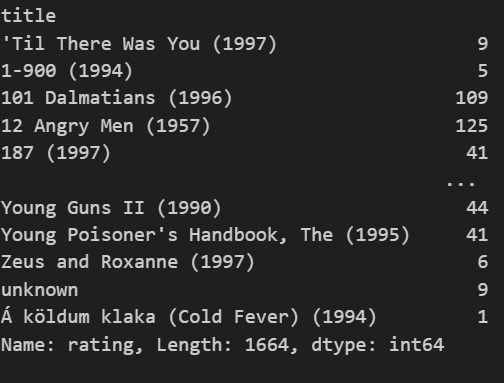
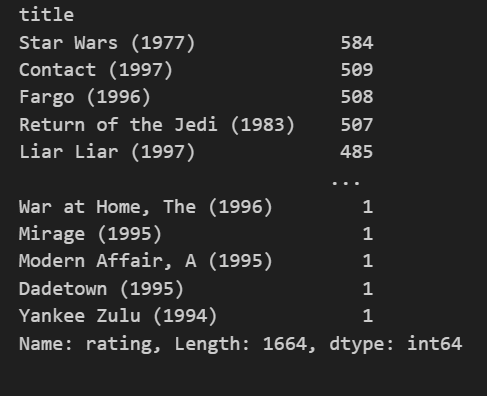


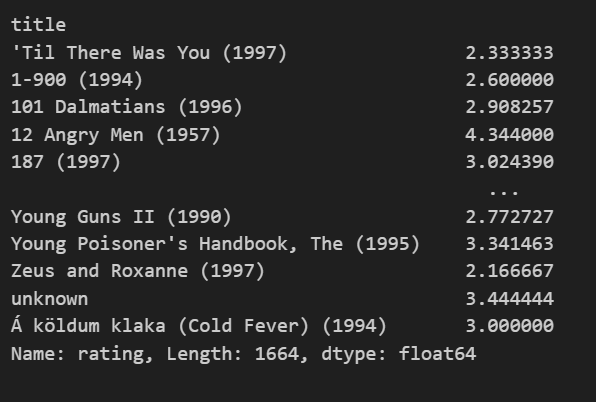
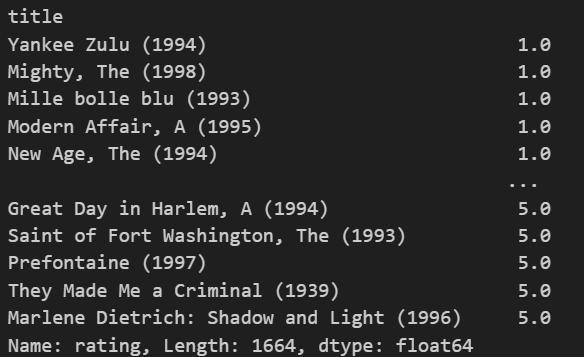
## Ogled podatkov

df.groupby('title').head()



Preskusi kodo in komentiraj rezultate:

Star wars ima najvec ocen, ker je najbolj popularen.

Pogost proble, je da če ne sortiramo po popularnosti in oceni hkrati ... Temveč samo po oceni, potem dobimo film »ki ga je ocenil samo 1 človek, a z oceno 5«

## Kaj naredimo, preizkusi ?

# Tabela filmov

#df.groupby('title')['rating'].count()

#df.groupby('title')['rating'].count().sort\_values(ascending=False)

#

#df.groupby('title')['rating'].mean()

#

#df.groupby('title')['rating'].mean().sort\_values()

## Podatki o filmih

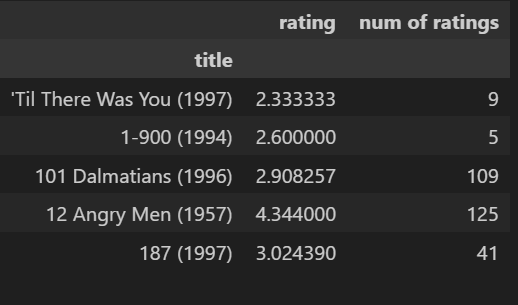
# tabela povpr ocen filmov

ratings = pd.DataFrame(df.groupby('title')['rating'].mean())

ratings.head()

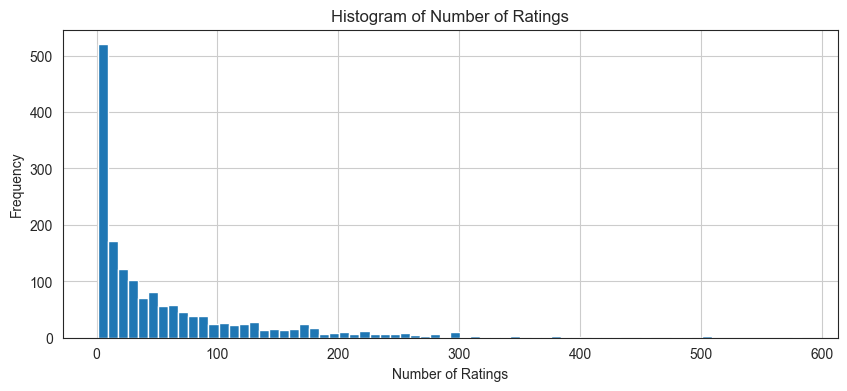
ratings['num of ratings'] = pd.DataFrame(df.groupby('title')['rating'].count())

ratings.head()

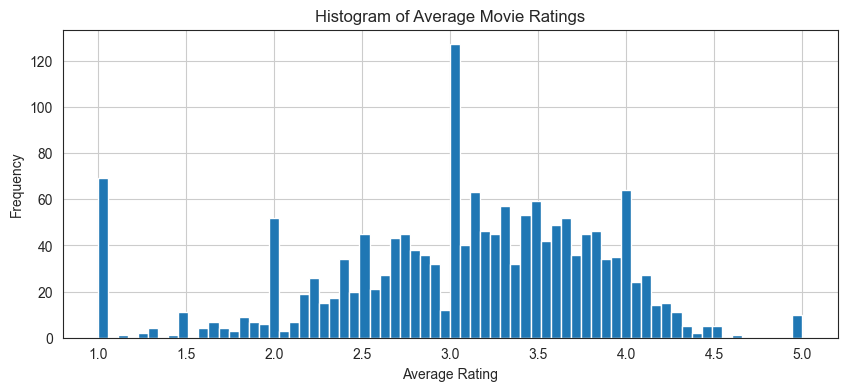


## Naloga : Izriši histograme

Izriši histogram števila ocen:



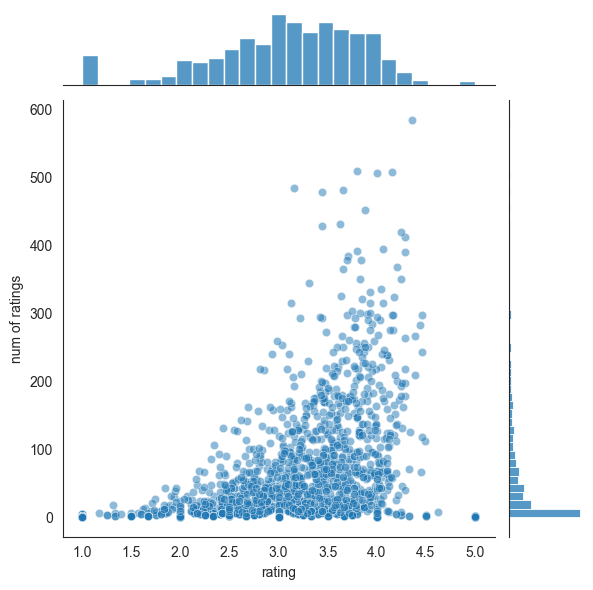
Izriši histogram povprečne ocene filmov (rating)



Izriši skupni histogram obeh:

sns.jointplot(x='rating',y='num of ratings',data=ratings,alpha=0.5)

Kaj nam skupni histogram pove o povprečnih ocenah filmov ?

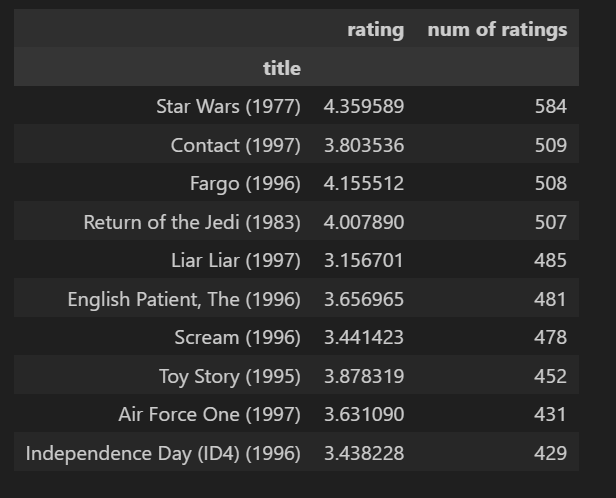


# Priporočanje podobnih filmov

moviemat = df.pivot\_table(index='user\_id',columns='title',values='rating')

moviemat.head()

Kaj so podatki v tej tabeli ?



ratings.sort\_values('num of ratings',ascending=False).head(10)

Kaj naredi koda, kaj predstavlja rezultat?

Rezultat bo natisnjen rezultat 10 najboljših filmov z najvišjim številom ocen v podatkovnem okviru ocen. To pomaga prepoznati najbolj ocenjene filme v zbirki podatkov

## Podobnost filmov

starwars\_user\_ratings = moviemat['Star Wars (1977)']

starwars\_user\_ratings.head()

similar\_to\_starwars = moviemat.corrwith(starwars\_user\_ratings)

similar\_to\_starwars

Kaj smo dobili, kaj vrne metoda corrwith, in kakšen algoritem uporablja za izračun?

Pojasnilo: metoda corrwith vrne parno korelacijo vrstic ali stolpcev dveh objektov DataFrame.

Privzeto uporablja Pearsonov korelacijski koeficient.

corr\_starwars = pd.DataFrame(similar\_to\_starwars,columns=['Correlation'])

corr\_starwars.dropna(inplace=True)

corr\_starwars.head()

corr\_starwars.sort\_values('Correlation',ascending=False).head(100)

Izpis: kaj smo dobili ?



## Obdelamo podatke o podobnosti filmov

corr\_starwars = corr\_starwars.join(ratings['num of ratings'])

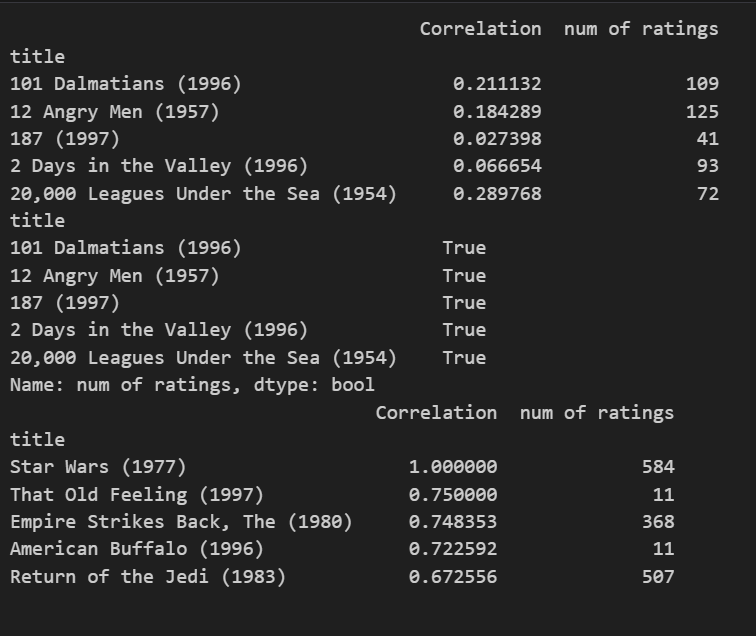
corr\_starwars.head()

index = corr\_starwars['num of ratings']>10

index.head()

corr\_starwars[index].sort\_values('Correlation',ascending=False).head()

Vstavi izpis:



Komentiraj, kaj smo naredili, in kaj smo dobili kot rezultat ?

Stolpec 'število ocen' smo združili z ratings DataFrame v corr\_starwars DataFrame.

Nato smo ustvarili logični indeks za filtriranje filmov z več kot 10 ocenami. Končno smo razvrstili filtrirani DataFrame po stolpcu 'Korelacija' v padajočem vrstnem redu in prikazali najboljše rezultate.

## Naloga

Ponovi postopek generiranja podobnih filmov za svoj izbrani film. Kopiraj ustrezno kodo iz primera.

* Izberi si svoj film (z najmanj 50 ocenami)
* Izriši histogram njegovih ocen
* Izpiši vsaj 10 najbolj podobnih filmov.

**Koda:**

#2.1

# DO NOT Get user ratings for Star Wars (1977)

starwars\_user\_ratings = moviemat['Toy Story (1995)']

print(starwars\_user\_ratings.head())

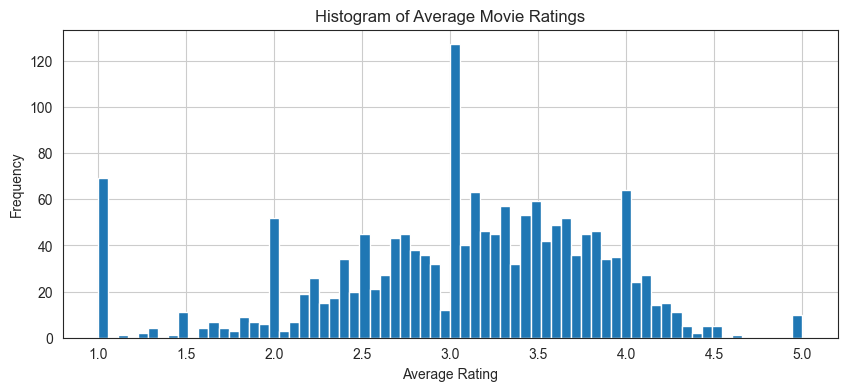
# Filter out movies with less than 10 ratings

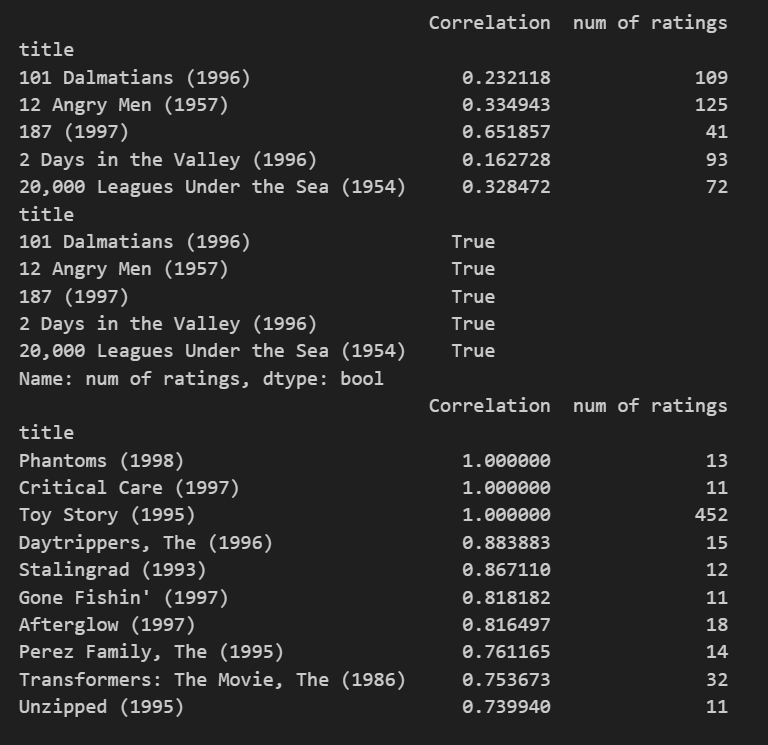
movie\_stats = ratings['num of ratings']

popular\_movies = movie\_stats[movie\_stats >= 10].index

filtered\_moviemat = moviemat[popular\_movies]

**Končni rezultat:**

****

****

**Komentar: Na čem temelji izračun podobnosti, in ali so rezultati smiselni glede na vsebino (naslov) filma, .. ?**

Izračun podobnosti temelji na edinstvenem številu uporabnikov in filmov ter oceni filmov. Če bi imeli več podatkov bi naredili boljši sistem npr dodali še žaner itd. (genre). Ob pomanjkanju podatkov smo primorani delati preprostejše modele.